GaussDB

Product Description

Issue	01
Date	2023-03-30

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions

HUAWEI and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

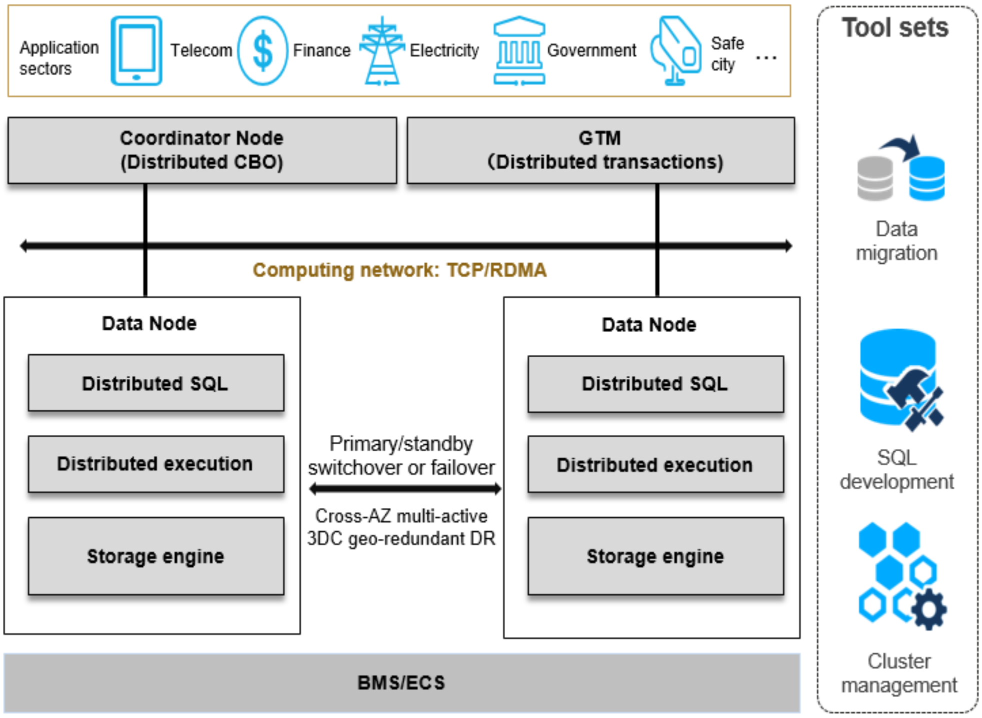
Contents

1 Introduction

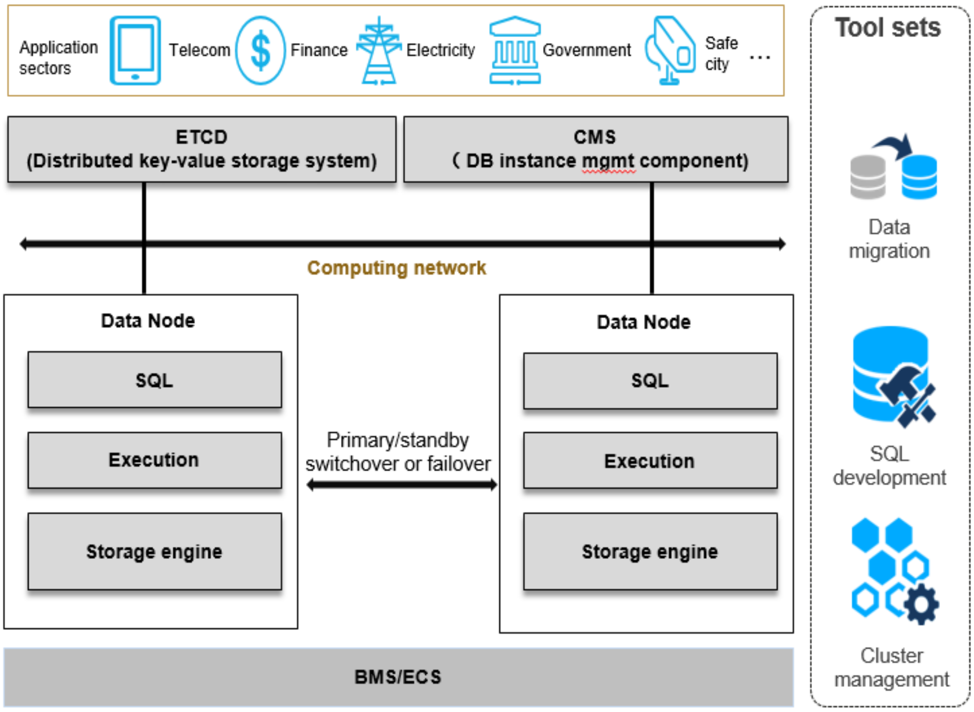
1.1 What Is GaussDB?

GaussDB is a distributed relational database from Huawei. It supports hybrid transactional/analytical processing (HTAP) workloads and intra-city cross-AZ deployment. With a distributed architecture, GaussDB supports petabytes of storage and contains more than 1,000 nodes per DB instance. It is highly available, secure, and scalable and provides services including quick deployment, backup, restoration, monitoring, and alarm reporting for enterprises.

The overall architecture of a distributed instance is as follows:



The overall architecture of a primary/standby instance is as follows:



1.2 Scenarios

- Transaction applications

  The distributed, highly scalable architecture of GaussDB makes it an ideal fit for highly concurrent online transactions containing a large volume of data from government, finance, e-commerce, O2O, telecom customer relationship management (CRM), and billing. GaussDB supports different deployment models.

- CDR query

  GaussDB can process petabytes of data and use the memory analysis technology to query massive volumes of data when data is being written to databases. Therefore, it is suitable for the Call Detail Record (CDR) query service in the security, telecom, finance, and Internet of things (IoT) sectors.

1.3 Software Architecture

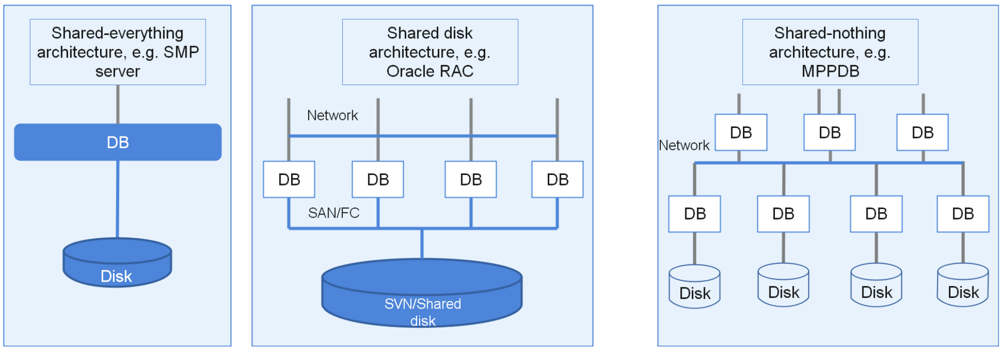
Scenario

GaussDB is designed based on the shared-nothing architecture. It consists of multiple independent logical nodes that do not share system resources, such as the CPU, memory, and storage resources. In such an architecture, service data is stored on multiple compute nodes. Data query tasks are executed on the nearest nodes. With coordination of CNs, a large amount of data can be processed concurrently, achieving quick data processing.

Shared-Nothing Architecture

Figure 1-1 shows the comparison between shared-nothing and other architectures.

Figure 1-1 Architecture comparison


Shared-everything architecture, e.g. SMP server
DB
Disk
Shared disk architecture, e.g. Oracle RAC
Network
DB
DB
DB
DB
SAN/FC
SVN/Shared disk
Shared-nothing architecture, e.g. MPPDB
Network
DB
DB
DB
DB
DB
DB
DB
Disk
Disk
Disk
Disk

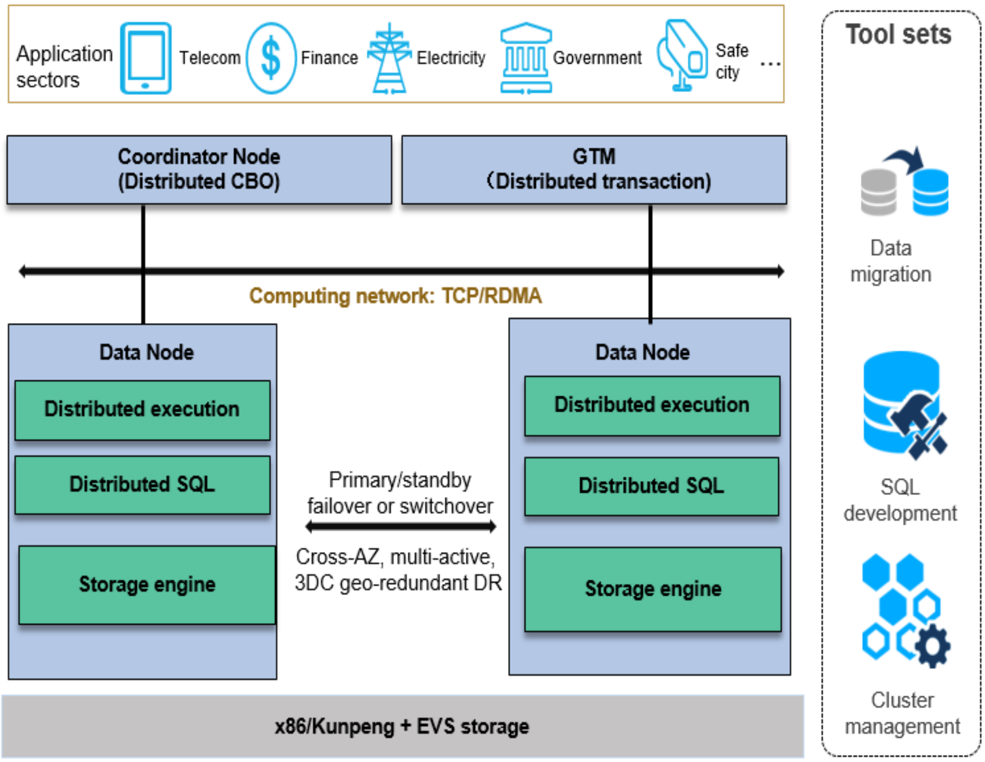
The shared-nothing architecture has the following advantages:

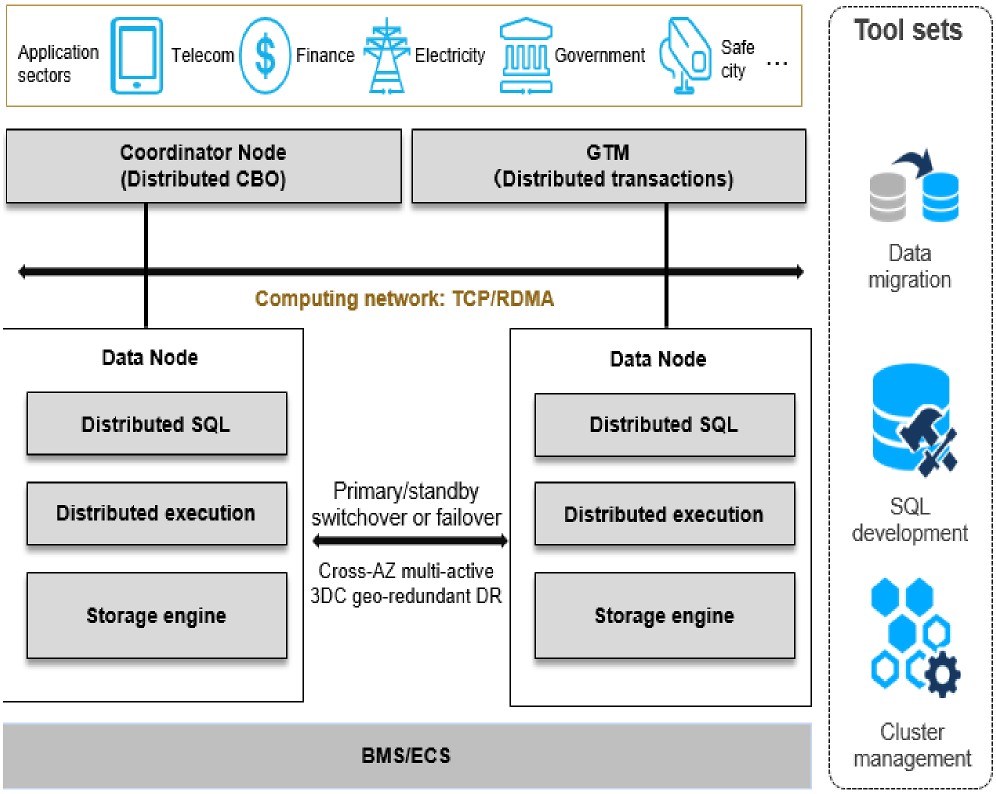
- Flexible scalability
  - Processes high concurrent requests and huge amounts of data during BI and data analysis.
  - Uses a parallel processing mechanism.
- Automated internal parallel processing
  - Allows you to load and access data as you would in a common database.
  - Distributes data on all parallel nodes.
  - Ensures that every node processes only partial data.
- Optimal I/O processing
  - Allows all nodes to process data in parallel.
  - Ensure that nodes share nothing with each other and have no I/O conflicts.
- High storage, query, and loading performance by adding nodes

Software Architecture

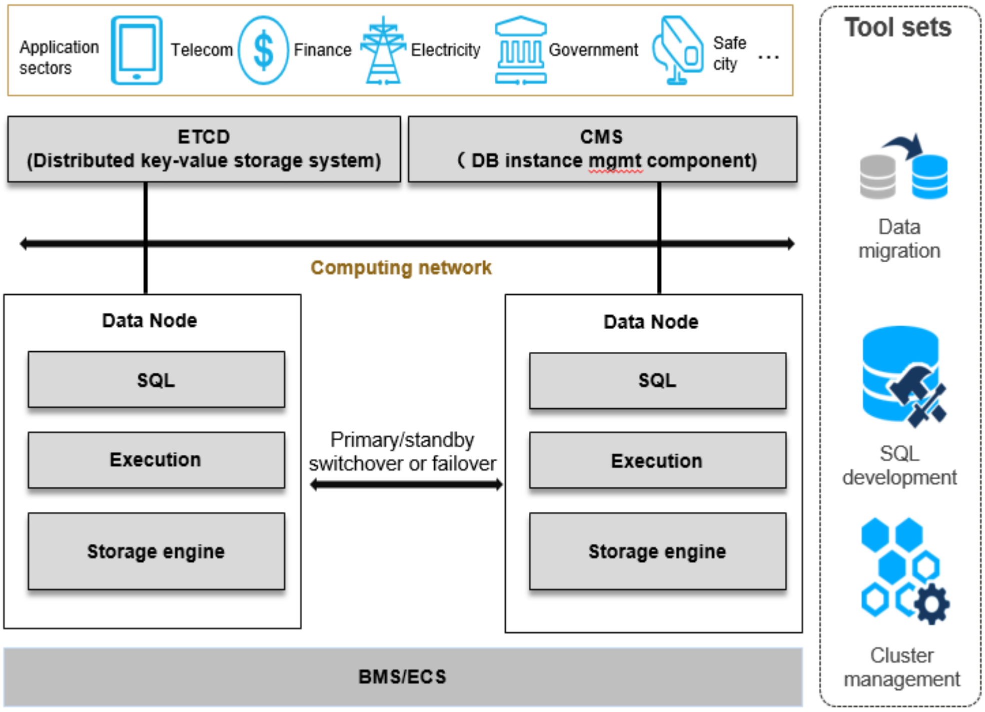
GaussDB is a distributed database based on the shared-nothing architecture.

The following figure shows the overall architecture of GaussDB in distributed model.





The overall architecture of a primary/standby instance is as follows:



GaussDB consists of Global Transaction Manager (GTM), Operation Manager (OM), Cluster Manager (CM), Coordinator Node (CN), and Data Node (DN). **Figure 1-2** and **Figure 1-3** show the distributed and primary/standby logical architectures, respectively. Not all compute nodes have GTMs and CNs shown in the figure. For details, see **Table 1-1**.

Figure 1-2 Distributed logical architecture

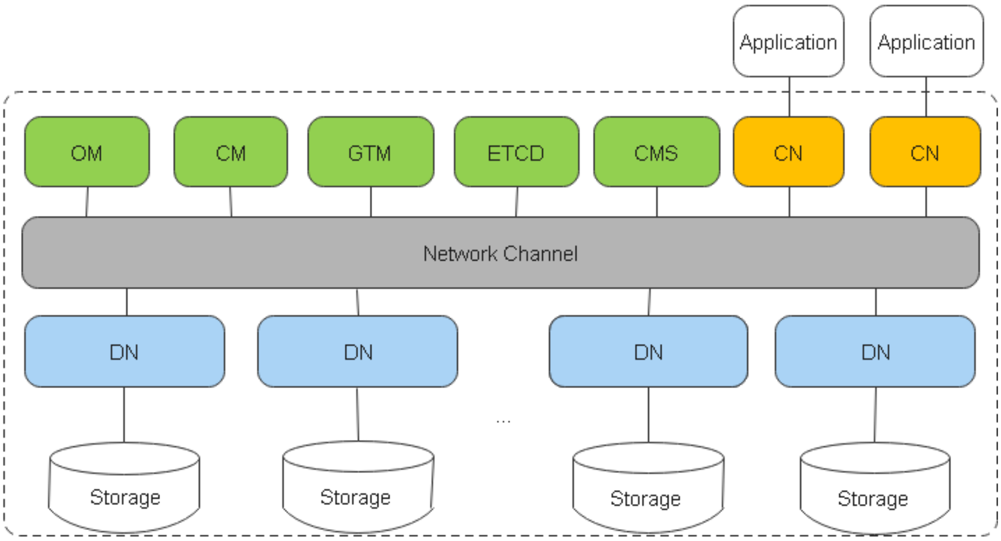


Figure 1-3 Primary/standby logical architecture

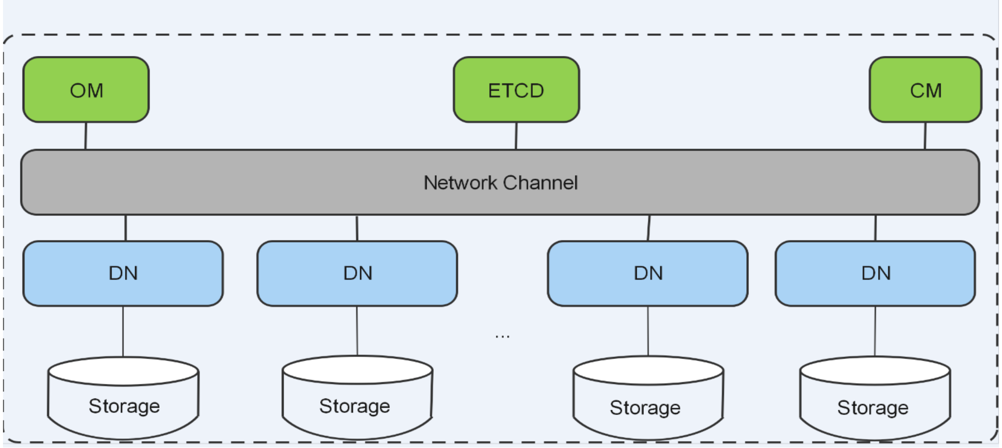


Table 1-1 Instance description

Instance	Description	Remarks
OM	Operation Manager. It provides management interfaces and tools for routine maintenance and configuration management of the cluster.	Different from instances (such as GTM, CM, CN, and DN), OM provides tools for managing the cluster.

Instance	Description	Remarks
CM	Cluster Manager. It manages and monitors the running status of functional units and physical resources in a distributed system, ensuring stable running of the entire system.	CM consists of CM Agent, OM Monitor, and CM Server. ● CM Agent monitors the running status of primary and standby GTMs, CNs, DNs on the servers and reports the status to CM Server. In addition, it executes the arbitration instruction delivered by CM Server. A CM Agent process runs on each host of the cluster. CM Server saves the cluster topology information in ETCD. ● OM Monitor monitors scheduled tasks on CM Agent and restarts CM Agent when CM Agent stops. If CM Agent cannot be restarted, the host cannot be used. In this case, you need to manually rectify this failure. **NOTE** CM Agent fails to be restarted in a low probability. The possible reason is that new processes cannot be started due to insufficient system resources. ● CM Server checks whether the CM status is normal according to the status reported by CM Agent. If the status is abnormal, CM Server delivers recovery commands to CM Agent. GaussDB supports the primary and standby CM Servers to ensure system high availability. CM Agent connects to the primary CM Server. If the primary CM Server is faulty, the standby CM Server is promoted to primary to prevent a single point of failure (SPOF).
GTM	Global Transaction Manager. It generates and maintains the global transaction IDs, transaction snapshots, time stamps, and sequences that must be unique globally.	The entire cluster has only one primary GTM and one or more standby GTMs.

Instance	Description	Remarks
CN	Coordinator Node. It receives access requests from applications and returns execution results to the client. A CN splits and distributes a task to different DNs for parallel processing.	A CN receives access requests from applications and returns execution results to clients. It also splits and distributes a task to different DNs for parallel processing. In a cluster, multiple CNs can be deployed on different compute nodes. CNs are peers to one another. When a DML statement is executed, the same result can be obtained by connecting to any CN.
DN	Data Node. It stores service data (by column, row, or hybrid store), performs data queries, and returns execution results to a CN.	DNs store service data, execute data query tasks, and return execution results to a CN. GaussDB supports one primary DN and multiple standby DNs for high reliability. In a cluster, the number of DNs can be configured in the configuration file. The principle is as follows: Data is replicated between the primary and standby DNs based on quorum. Data is stored on both the primary and standby DNs. For example, if there are three DNs (one primary and two standby DNs), three data copies are available. Even if one of the three DNs is faulty, the cluster still has two copies of data to ensure service continuity. In addition, any standby DN can be promoted to the primary. You are advised to deploy DNs on different compute nodes.
ETCD	Editable Text Configuration Daemon. It is used for shared configuration and service discovery (service registry and search).	ETCD provides functions such as service discovery, message publishing and subscription, load balancing, distributed notification and coordination, distributed lock, distributed queue, cluster monitoring, and leader election.

1.4 Basic Concepts

Instances

The smallest management unit of GaussDB is the instance. A DB instance is an isolated database environment on the cloud. You can create and manage instances on the management console. For details about instance statuses, instance specifications, storage types, and versions, see **DB Instance Description**.

Instance Versions

GaussDB 3.200 is supported.

Instance Types

GaussDB supports distributed and primary/standby instances. You can add nodes for distributed instances as needed to handle large volumes of concurrent requests. The primary/standby instances are suitable for scenarios with small and stable volumes of data, where data reliability and service availability are extremely important.

Instance Specifications

The instance specifications determine the computation (vCPUs) and memory capacity (in GB) of an instance. For details, see **Instance Specifications**.

Coordinator Nodes

Coordinator nodes (CNs) store database metadata, distribute and execute query tasks, and then return the query results from DNs to applications.

Data Nodes

Data nodes (DNs) store and query table data.

Automated Backups

When you create an instance, automated backup is enabled by default. After the instance is created, you can modify the backup policy. GaussDB will automatically create backups for instances based on your settings.

Manual Backups

Manual backups are user-initiated full backups of instances. They are retained until you delete them manually.

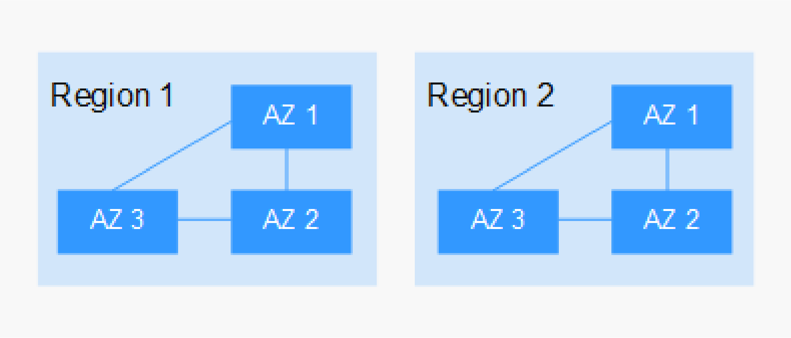
Regions and AZs

A region and availability zone (AZ) identify the location of a data center. You can create resources in a specific region and AZ.

- A region is a physical data center. Each region is isolated from the other regions, improving fault tolerance and stability. The region that is selected during resource creation cannot be changed after the resource is created.
- An AZ is a physical location using independent power supplies and networks. Faults in an AZ do not affect other AZs. A region contains one or more AZs that are physically isolated but interconnected through internal networks. Because AZs are isolated from each other, any fault that occurs in one AZ will not affect others.

Figure 1-4 shows the relationship between regions and AZs.

Figure 1-4 Regions and AZs



Resource Set

Resource sets are used to group and isolate underlying resources (such as compute, storage, and network resources). A resource set can be a department or a project team. Multiple resource sets can be created for one account.

1.5 Advantages

- **High Security**

  GaussDB provides a wide range of features to let you enjoy the security of top-level commercial databases at a low cost: dynamic data masking, transparent data encryption (TDE), row-level access control, and Always Encrypted.
- **Comprehensive Tools and Service-oriented Capabilities**

  GaussDB can be deployed in the Huawei Cloud Stack for commercial use and can work with ecosystem tools such as Data Admin Service (DAS) and Data Replication Service (DRS) to make database development, O&M, tuning, monitoring, and migration easy.
- **In-House, Full-Stack Development**

  GaussDB performance is always improved to meet ever-increasing demands in different scenarios.
- **Open-Source Ecosystem**

  The primary/standby version is available for you to download from the openGauss community.

1.6 DB Instance Description

1.6.1 Instance Statuses

Instance Statuses

The status of a DB instance reflects the health of the instance. You can use the management console to view the status of a DB instance.

Table 1-2 DB Instance statuses

Status	Description
Available	The instance is available.
Abnormal	The instance is unavailable.
Creating	The instance is being created.
Creation failed	The instance failed to be created.
Rebooting	The instance is being rebooted because of a user request or a modification that requires a reboot for the modification to take effect.
Scaling up	The storage space of the instance is being scaled up.
Adding nodes	The nodes are being added to the instance.
Changing instance specifications	The instance specifications are being changed.
Backing up	The backup is being created.
Restoring	The instance is being restored from a backup.
Restore failed	The instance failed to be restored.
Storage full	The storage space of the instance is full. No more data cannot be written to the databases on this instance.
Deleted	The instance has been deleted. Deleted instances will not be displayed in the instance list.
Upgrading	The engine version is being upgraded.
Parameters change. Pending reboot	A modification to a database parameter is waiting for a DB instance reboot before it can take effect.

Status	Description
Balancing the distribution of primary and standby nodes	The distribution of the primary and standby nodes is being balanced.
Observing version upgrade	The instance is in the observation period during the gray rolling upgrade.

Backup Statuses

Table 1-3 Backup statuses

Status	Description
Completed	The backup was successfully created.
Failed	The backup failed to be created.
Creating	The backup is being created.

1.6.2 Instance Specifications

Table 1-4 Instance Specifications

Specification Type	vCPUs	Memory (GB)	Storage Space (GB)	Maximum Connections (Default Value)
BMS (x86)	72	576	12 x 960 GB	Finance edition (data computing): Per CN: 1,000 Per DN: 5,000

Specification Type	vCPUs	Memory (GB)	Storage Space (GB)	Maximum Connections (Default Value)
	96	768	**NOTE** Storage space: 24 x 960 GB. For disks in a RAID 10 configuration, the available storage space is only 12 x 960 GB.	● Finance edition (standard): Per CN: 4,000 Per DN: 16,000 ● Enterprise edition: Per CN: 3,000 Per DN: 11,000 ● Finance edition (data computing): Per CN: 2,500 Per DN: 8,000 ● Primary/standby DB instance: 40,000
	96	1,024		● Finance edition (standard): Per CN: 6,000 Per DN: 21,000 ● Enterprise edition: Per CN: 4,000 Per DN: 15,000 ● Primary/standby DB instance: 55,000
	104	1,024		● Finance edition (standard): Per CN: 6,000 Per DN: 21,000 ● Enterprise edition: Per CN: 4,000 Per DN: 15,000 ● Primary/standby DB instance: 55,000
BMS (Arm)	64	512	12 x 960 GB	● Finance edition (standard): Per CN: 2,500 Per DN: 11,000 ● Enterprise edition: Per CN: 2,000 Per DN: 7,500 ● Primary/standby DB instance: 25,000

Specification Type	vCPUs	Memory (GB)	Storage Space (GB)	Maximum Connections (Default Value)
	96	768	**NOTE** Storage space: 24 x 960 GB. For disks in a RAID 10 configuration, the available storage space is only 12 x 960 GB.	- Finance edition (standard): Per CN: 4,000 Per DN: 16,000 - Enterprise edition: Per CN: 3,000 Per DN: 11,000 - Finance edition (data computing): Per CN: 2,500 Per DN: 8,000 - Primary/standby DB instance: 40,000
	128	1,024		- Finance edition (standard): Per CN: 6,000 Per DN: 21,000 - Enterprise edition: Per CN: 4,000 Per DN: 15,000 - Primary/standby DB instance: 55,000

Specification Type	vCPUs	Memory (GB)	Storage Space (GB)	Maximum Connections (Default Value)
General-enhanced II **NOTE** The general-enhanced II type is suitable for x86-powered instances in the ECS deployment.	4	32 **NOTE** This specification is not available for production environments. To use the specification, add **gaussdb_feature_support4U32G** to the trustlist.	Select it as required.	Distributed DB instance: Per CN: 100 Per DN: 100

Specification Type	vCPUs	Memory (GB)	Storage Space (GB)	Maximum Connections (Default Value)
	8	64 **NOTE** This specification is available for only primary/standby DB instances that run 2.6 or later versions.		Distributed DB instance: Per CN: 1,000 Per DN: 2,500 Primary/standby DB instance: 2,048
	16	128		Distributed DB instance: Per CN: 2,000 Per DN: 6,000 Primary/standby DB instance: 5,000
	32	256		Distributed DB instance: Per CN: 4,000 Per DN: 12,000 Primary/standby DB instance: 11,000

Specification Type	vCPUs	Memory (GB)	Storage Space (GB)	Maximum Connections (Default Value)
	64	512		Distributed DB instance: Per CN: 8,000 Per DN: 24,000 Primary/standby DB instance: 25,000
Kunpeng general-enhanced **NOTE** The Kunpeng general-enhanced is suitable for Arm-powered instances deployed in MCSs.	8	64 **NOTE** This specification is available for only primary/standby DB instances that run 2.6 or later versions.	Select it as required.	Distributed DB instance: Per CN: 1,000 Per DN: 2,500 Primary/standby DB instance: 2,048
	16	128		Distributed DB instance: Per CN: 2,000 Per DN: 6,000 Primary/standby DB instance: 5,000

Specification Type	vCPUs	Memory (GB)	Storage Space (GB)	Maximum Connections (Default Value)
	32	256		Distributed DB instance: Per CN: 4,000 Per DN: 12,000 Primary/standby DB instance: 11,000
	60	480		Distributed DB instance: Per CN: 8,000 Per DN: 24,000 Primary/standby DB instance: 24,000

Specification Type	vCPUs	Memory (GB)	Storage Space (GB)	Maximum Connections (Default Value)
Kunpeng general computing-plus **NOTE** The Kunpeng general computing-plus is suitable for Arm-powered instances deployed in ECSs.	4	32	Select it as required.	Distributed DB instance: Per CN: 100 Per DN: 100

Specification Type	vCPUs	Memory (GB)	Storage Space (GB)	Maximum Connections (Default Value)
		NOTE Only instances in the ECS deployment are supported. This specification is not available for production environments. To use the specification, add **gaussdb_feature_support4U32G** to the trustlist.		

Specification Type	vCPUs	Memory (GB)	Storage Space (GB)	Maximum Connections (Default Value)
	8	64 **NOTE** This specification is available for only primary/standby DB instances that run 2.6 or later versions.		Distributed DB instance: Per CN: 1,000 Per DN: 2,500 Primary/standby DB instance: 2,048
	16	128		Distributed DB instance: Per CN: 2,000 Per DN: 6,000 Primary/standby DB instance: 5,000
	32	256		Distributed DB instance: Per CN: 4,000 Per DN: 12,000 Primary/standby DB instance: 11,000

Specification Type	vCPUs	Memory (GB)	Storage Space (GB)	Maximum Connections (Default Value)
	60	480		Distributed DB instance: Per CN: 8,000 Per DN: 24,000 Primary/standby DB instance: 24,000

1.6.3 Instance Storage Types

The database system is generally an important system in the IT system and has high requirements on storage I/O performance. GaussDB supports SSDs in the BMS and MCS deployment modes and ultra-high I/O storage in the ECS deployment mode.

1.6.4 Instance Versions

GaussDB 3.200 Enterprise Edition is supported.

1.7 User Roles and Permissions

ManageOne Operation Portal (ManageOne Operation Management Portal in B2B scenarios) provides role management and access control functions for cloud services. Role management refers to management of users and user groups. Access control refers to management of their permissions.

ManageOne Operation Portal in B2B scenarios allows users to control access to GaussDB SQL resources. One or more of the permissions listed in **Table 1-5** can be assigned to a user to use GaussDB SQL.

Table 1-5 User roles and permissions

Role	Role Source	Permission	Description
GaussDB SQL administrator	VDC administrator	● VDC management permissions ● Management permissions on all cloud services	Users with the permissions can perform any operation on the GaussDB SQL resources.
	VDC operator	● VDC operator permissions ● Management permissions on all cloud services	

Role	Role Source	Permission	Description
	User-defined	● VDC query permissions ● Management permissions on all cloud services	
		● VDC management permissions, query permissions, or operator permissions ● GaussDB SQL management permissions	
GaussDB SQL read-only user	VDC read-only administrator	● VDC query permissions ● Query permissions on all cloud services	Users with the permissions can query the resource usage of GaussDB SQL. It means that users with the permissions can read GaussDB SQL databases.
	User-defined	● VDC management permission or operator permission ● Query permissions on all cloud services	

Table 1-6 lists the common operations supported by each system-defined policy of GaussDB SQL. Select the proper system-defined policies as required.

Table 1-6 Common operations supported by each system-defined policy or role

Operation	GaussDB FullAccess	GaussDB ReadOnlyAccess
Creating a GaussDB SQL instance	Supported	Not supported
Deleting a GaussDB SQL instance	Supported	Not supported
Querying a GaussDB SQL instance list	Supported	Supported

NOTE

- GaussDB FullAccess: administrator permissions of GaussDB SQL. By default, this role has all permissions to perform operations on GaussDB SQL.
- GaussDB ReadOnlyAccess: read-only permissions for GaussDB SQL. This role can also perform some custom operations on GaussDB SQL.
- To use other services, it is required to add the corresponding actions by referring to the **Remarks** column in **Table 1-7** and **Table 1-8**.

Table 1-7 lists common GaussDB SQL operations and corresponding actions. You can refer to this table to customize permission policies.

Table 1-7 Common operations and supported actions

Operation	Action	Remarks
Creating a DB instance	gaussdb:instance:create gaussdb:param:list	To select a VPC, subnet, and security group, configure the following actions: VPC Administrator Server Administrator Tenant Administrator
Rebooting a DB instance	gaussdb:instance:restart	N/A
Deleting a DB instance	gaussdb:instance:delete	N/A
Querying DB instances	gaussdb:instance:list	N/A
Querying instance details	gaussdb:instance:list	If the VPC, subnet, and security group are displayed in the DB instance list, you need to configure VPC administrator and server administrator. If the storage usage is displayed, you need to configure **VDC Readonly**.
Recycling a DB instance	gaussdb:instance:list	N/A
Changing a DB instance name	gaussdb:instance:modify	N/A
Querying instance parameter details	gaussdb:param:list	N/A

Operation	Action	Remarks
Creating a parameter template	gaussdb:param:create	N/A
Modifying a parameter template	gaussdb:param:modify	N/A
Obtaining parameter templates	gaussdb:param:list	N/A
Applying a parameter template	gaussdb:param:apply	N/A
Deleting a parameter template	gaussdb:param:delete	N/A
Creating a manual backup	gaussdb:backup:create	N/A
Deleting a manual backup	gaussdb:backup:delete	N/A
Obtaining backups	gaussdb:backup:list	N/A
Modifying the backup policy	gaussdb:instance:modifyBackupPolicy	N/A
Modifying the recycling policy	gaussdb:instance:modifyBackupPolicy	N/A
Changing the retention period of automated backups	gaussdb:instance:modifyBackupPolicy	N/A
Querying the restoration time range	gaussdb:instance:list	N/A
Restoring data to a new DB instance	gaussdb:instance:create	To select a VPC, subnet, and security group, configure the following actions: VPC Administrator Server Administrator
Restoring data to the original DB instance	gaussdb:instance:restoreInPlace	N/A
Scaling up storage space	gaussdb:instance:modifySpec	N/A
Changing vCPUs and memory of an instance	gaussdb:instance:modifySpec	N/A
Adding a node	gaussdb:instance:modifySpec	N/A

Operation	Action	Remarks
Resetting a password	gaussdb:instance:modify	N/A
Managing logs	gaussdb:instance:list	N/A
Downloading logs	gaussdb:instance:modify	N/A
Exporting DB instance information	gaussdb:instance:list	N/A
Downloading a driver	gaussdb:instance:list	N/A

Table 1-8 DR operations and supported actions

Operation	Action	Remarks
Querying instances that can establish a DR relationship with a primary instance	gaussdb:disasterRecovery: list	The feature trustlist **gaussdb_feature_supportDisasterApiGlobal** must be enabled. You need to configure VPC Administrator.
Checking DR operations	gaussdb:disasterRecovery: list	The feature trustlist **gaussdb_feature_supportDisasterApiGlobal** must be enabled. You need to configure VPC Administrator.
Querying instance DR status	gaussdb:disasterRecovery: list	The feature trustlist **gaussdb_feature_supportDisasterApiGlobal** must be enabled. You need to configure VPC Administrator.
Querying the DR relationship of instances	gaussdb:disasterRecovery: list	The feature trustlist **gaussdb_feature_supportDisasterApiGlobal** must be enabled.
Establishing a DR relationship	gaussdb:disasterRecovery:construct	The feature trustlist **gaussdb_feature_supportDisasterApiGlobal** must be enabled. You need to configure VPC Administrator.

Operation	Action	Remarks
Promoting the DR instance to primary	gaussdb:disasterRecovery:failover	The feature trustlist **gaussdb_feature_supportDisasterApiGlobal** must be enabled. You need to configure VPC Administrator.
Deleting a DR relationship	gaussdb:disasterRecovery:release	The feature trustlist **gaussdb_feature_supportDisasterApiGlobal** must be enabled. You need to configure VPC Administrator.
Switch roles of primary and DR instances	gaussdb:disasterRecovery:switchover	The feature trustlist **gaussdb_feature_supportDisasterApiGlobal** must be enabled. You need to configure VPC Administrator.
Performing a failback	gaussdb:disasterRecovery:construct	The feature trustlist **gaussdb_feature_supportDisasterApiGlobal** must be enabled. You need to configure VPC Administrator.
Performing a DR simulation	gaussdb:disasterRecovery:simulation	The feature trustlist **gaussdb_feature_supportDisasterApiGlobal** and **gaussdb_feature_supportDrSimulation** must be enabled. You need to configure VPC Administrator.
Data cache in progress	gaussdb:disasterRecovery:keeplog	The feature trustlist **gaussdb_feature_supportDrLogKeep** must be enabled. You need to configure VPC Administrator.

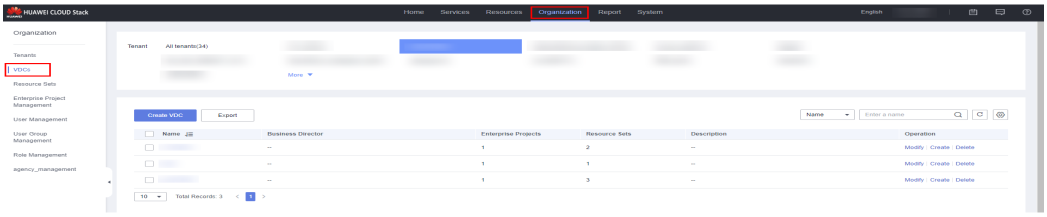
Creating a User Group and Assigning Permissions

Step 1 Use a browser to log in to ManageOne as a VDC administrator.

URL in non-B2B scenarios: **https://***Domain name of ManageOne Operation Portal*, for example, **https://console.demo.com**.

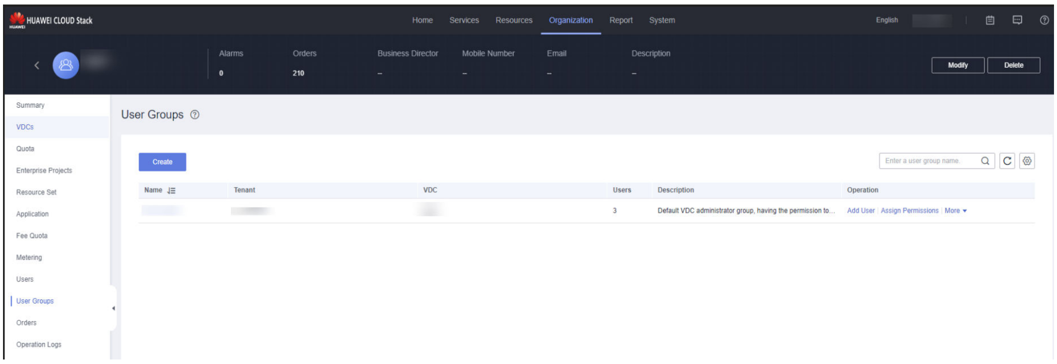
URL in B2B scenarios: **https://***Doman name for accessing ManageOne Tenant Portal*, for example, **https://tenant.demo.com**.

Step 2 Choose **Organization** > **VDCs**. On the displayed page, select the target VDC user and click the VDC name.



Name	Business Director	Enterprise Projects	Resource Sets	Description	Operation
	--	1	2	--	Modify | Create | Delete
	--	1	1	--	Modify | Create | Delete
	--	1	3	--	Modify | Create | Delete

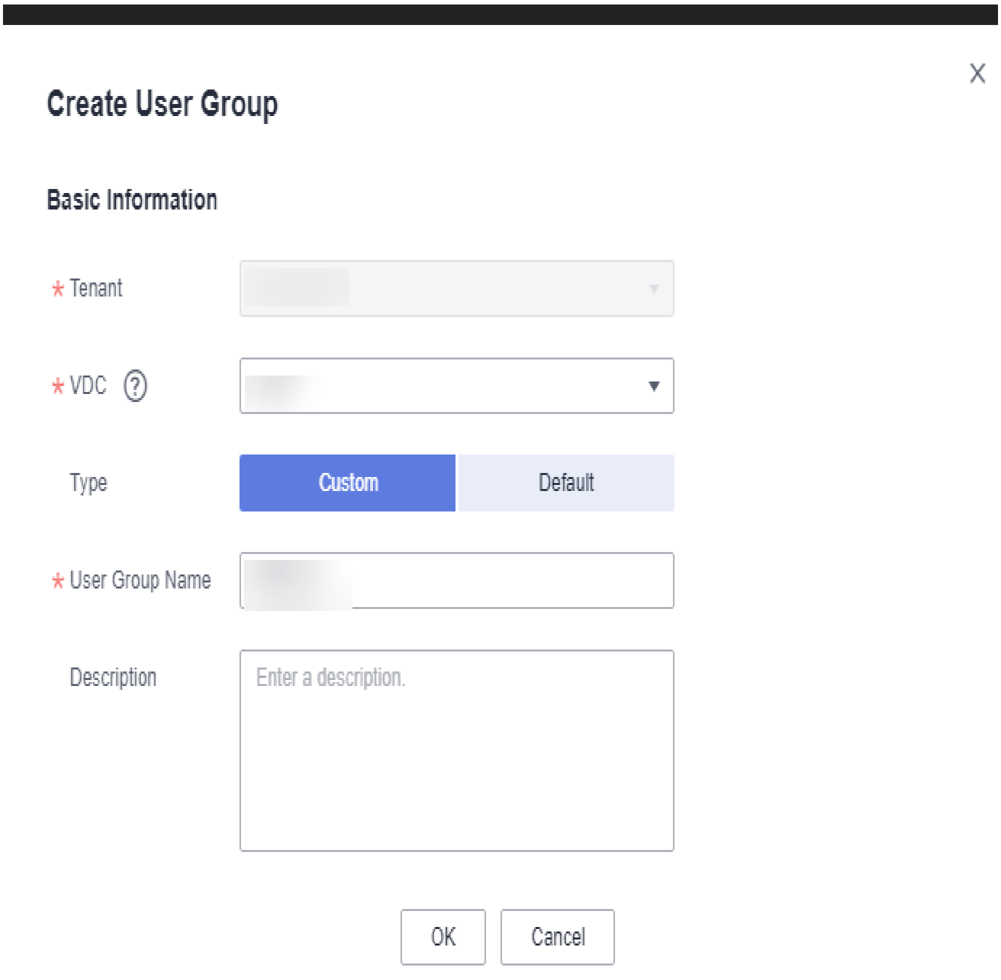
Step 3 In the navigation pane, click **User Groups**. Then, click **Create**.



Name	Tenant	VDC	Users	Description	Operation
			3	Default VDC administrator group, having the permission to...	Add User | Assign Permissions | More

Step 4 In the displayed dialog box, configure the required parameters and click **OK**.

- **Type**: Select **Custom**.
- **User Group Name**: The name contains 1 to 64 characters and cannot start with a digit. It can contain only letters, digits, hyphens (-), and underscores (_), and cannot be **admin**, **power_user**, or **guest**.
- **Description**: It contains 0 to 255 characters.

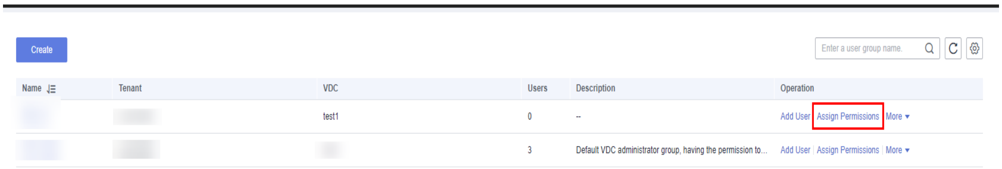


Create User Group

Basic Information

* Tenant	
* VDC ?	
Type	Custom / Default
* User Group Name	
Description	Enter a description.

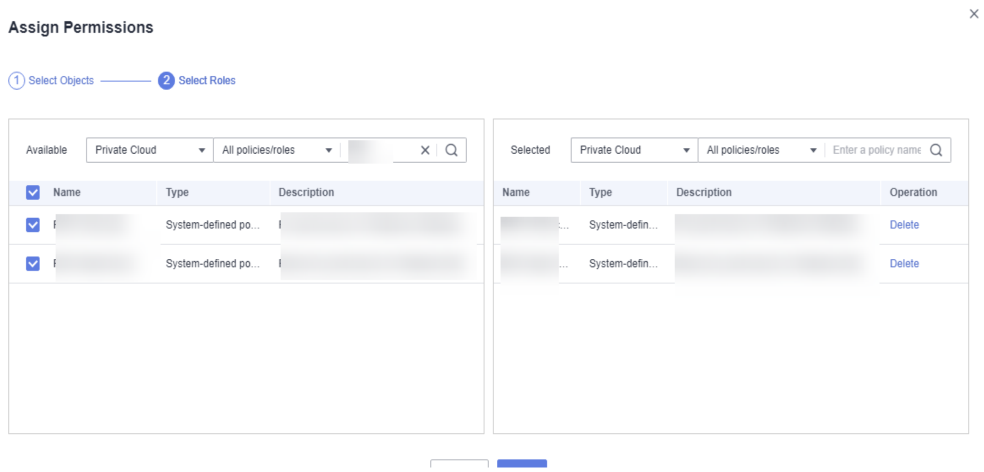
Step 5 After the creation is complete, click **Assign Permissions** in the **Operation** column.



Name	Tenant	VDC	Users	Description	Operation
		test1	0	--	Add User | Assign Permissions | More ▾
			3	Default VDC administrator group, having the permission to...	Add User | Assign Permissions | More ▾

Step 6 On the displayed page, select the object to be authorized and click **Next**.

Step 7 Select the required policies (system-defined policies or user-defined policies created in **Creating a Custom Policy**) and click **OK**.



☑	Name	Type	Description
☑		System-defined po...	
☑		System-defined po...	

Name	Type	Description	Operation
...	System-defin...		Delete
...	System-defin...		Delete

----End

Creating a Custom Policy

The service has multiple built-in operations. You can allow or deny some operations and apply custom policies to user groups.

Step 1 Use a browser to log in to ManageOne as an operation administrator.

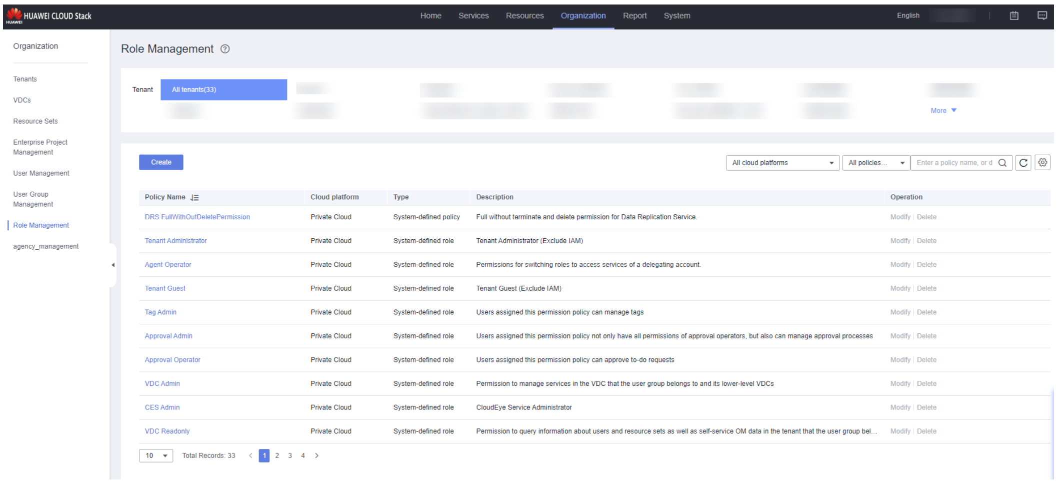
URL in non-B2B scenarios: **https://***Domain name of ManageOne Operation Portal*, for example, **https://console.demo.com**.

URL in B2B scenarios: https://*Domain name of accessing ManageOne Management Portal*, for example, **https://tenant.demo.com**.

Step 2 Choose **Organization** > **Role Management**.

Step 3 Click **Create** in the upper left corner of the page.

Figure 1-5 Role management

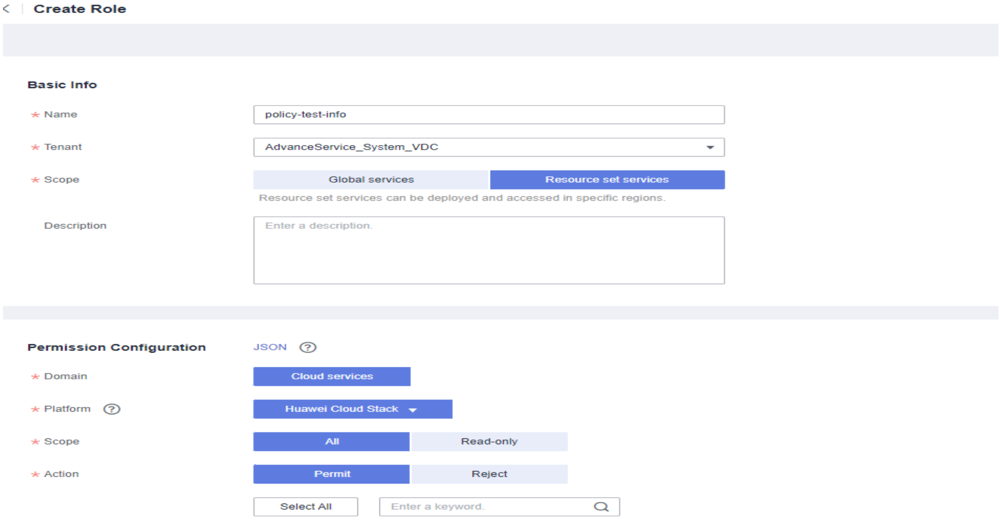


Policy Name	Cloud platform	Type	Description	Operation
DRS FullWithOutDeletePermission	Private Cloud	System-defined policy	Full without terminate and delete permission for Data Replication Service.	Modify | Delete
Tenant Administrator	Private Cloud	System-defined role	Tenant Administrator (Exclude IAM)	Modify | Delete
Agent Operator	Private Cloud	System-defined role	Permissions for switching roles to access services of a delegating account.	Modify | Delete
Tenant Guest	Private Cloud	System-defined role	Tenant Guest (Exclude IAM)	Modify | Delete
Tag Admin	Private Cloud	System-defined role	Users assigned this permission policy can manage tags	Modify | Delete
Approval Admin	Private Cloud	System-defined role	Users assigned this permission policy not only have all permissions of approval operators, but also can manage approval processes	Modify | Delete
Approval Operator	Private Cloud	System-defined role	Users assigned this permission policy can approve to-do requests	Modify | Delete
VDC Admin	Private Cloud	System-defined role	Permission to manage services in the VDC that the user group belongs to and its lower-level VDCs	Modify | Delete
CES Admin	Private Cloud	System-defined role	CloudEye Service Administrator	Modify | Delete
VDC Readonly	Private Cloud	System-defined role	Permission to query information about users and resource sets as well as self-service OM data in the tenant that the user group bel...	Modify | Delete

10 Total Records: 33 < 1 2 3 4 >

Step 4 On the displayed page, configure related parameters and click **OK**.

Figure 1-6 Creating a custom policy



< Create Role

Basic Info

* Name: policy-test-info

* Tenant: AdvanceService_System_VDC

* Scope: Global services | Resource set services

Resource set services can be deployed and accessed in specific regions.

Description: Enter a description.

Permission Configuration JSON

* Domain: Cloud services

* Platform: Huawei Cloud Stack

* Scope: All | Read-only

* Action: Permit | Reject

Select All | Enter a keyword.

Table 1-9 Parameter description

Parameter	Description
Name	The system provides a default policy name, for example, policy-GaussDB. You can change it.
Tenant	Select a tenant.
Scope	● **Global services**: Global services that can be directly deployed and accessed in any regions. ● **Resource set services**: Resource set services can be deployed and accessed in specific regions.
Description	(Optional) Describes the custom policy.
Permission Configuration	● **Platform**: Choose **HUAWEI CLOUD** > **GaussDB SQL (GaussDB)**. ● **Scope**: Operation permissions can be filtered as needed. ● **Action**: Select **Permit** or **Reject** as needed.

----End

1.8 Deployment Solutions

1.8.1 Distributed Deployment in Huawei Cloud Stack

GaussDB supports distributed deployment in a Huawei Cloud Stack environment.

Deployment	Replicas	AZ	Description	Deployment Method
Intra-city HA	4	3	Two service AZs and one quorum AZ. Two replicas are symmetrically deployed in each service AZ.	● MCS container + local disk storage ● ECS+Ultra-high I/O ● BMS (centralized gateway, the ratio of BMGW to BMS does not exceed 2:30) + local disk storage
	3	1	Three replicas deployed in one service AZ	● MCS container + local disk storage ● ECS+Ultra-high I/O ● BMS (centralized gateway, the ratio of BMGW to BMS does not exceed 2:30) + local disk storage
	3	3	Each replica for each service AZ	● MCS container + local disk storage ● ECS+Ultra-high I/O ● BMS (centralized gateway, the ratio of BMGW to BMS does not exceed 2:30) + local disk storage
	3	1	All components deployed in a single service AZ	ECS+Ultra-high I/O
	3	3	Three intra-city service AZs, with one replica each service AZ	ECS+Ultra-high I/O
Intra-city HA + remote DR	6	2	An intra-city service AZ with three replicas, a remote service AZ with three replicas and the same number of shards as the primary cluster	● MCS container + local disk storage ● ECS+Ultra-high I/O ● BMS (centralized gateway, the ratio of BMGW to BMS does not exceed 2:30) + local disk storage

Deployment	Replicas	AZ	Description	Deployment Method
	6	4	Three intra-city service AZs, with one replica each service AZ A remote service AZ, with three replicas and the same number of shards as the primary cluster.	• MCS container + local disk storage • ECS+Ultra-high I/O • BMS (centralized gateway, the ratio of BMGW to BMS does not exceed 2:30) + local disk storage
	6	4	Two intra-city service AZs (with two replicas in each service AZ) and one quorum AZ A remote service AZ with two replicas and the same number of shards as the primary cluster.	• MCS container + local disk storage • ECS+Ultra-high I/O • BMS (centralized gateway, the ratio of BMGW to BMS does not exceed 2:30) + local disk storage

Intra-city HA deployment

Intra-city HA scenario 1: intra-city 3-AZ 4-replica deployment

A complete intra-city active-active deployment solution consists of two service AZs and one quorum AZ. Two service AZs are deployed in peer-to-peer mode, and the equipment rooms in AZs access services. The quorum AZ is responsible for auxiliary quorum. It cannot access services, and can avoid single point of failure (SPOF). Any equipment room can achieve RPO of 0 and withstand network disconnections between equipment rooms. GaussDB also supports 2-AZ, 4-replica (one primary and three standby DNs), and 1-quorum AZ deployment solution. All primary roles are deployed in the primary AZ by default.

- AZ 1 and AZ 2 have complete data, and AZ 3 functions as the third-party quorum node.
- AZ 1 and AZ 2 can access services at the same time to implement dual-AZ active-active mode.
- AZ3 serves as the quorum AZ. If one AZ is faulty, the majority of ETCD nodes can survive, ensuring data consistency.
- In quorum-based replication between primary and standby DNs, there must be synchronous backup DNs across AZs and data will not be lost.

- If a standby DN is faulty, services are not interrupted. If the primary DN is faulty, a primary/standby switchover is automatically performed.
- This solution provides high availability for data center faults. If AZ1 or AZ2 is faulty, all services in the faulty AZ are automatically switched to the other AZ. After the switchover is complete, services can continue running.
- If any of AZ1 or AZ2 and the quorum AZ are faulty, you need to manually start the faulty AZs.

Figure 1-7 Intra-city three AZ, 4-replica BMS/MCS deployment (standard)

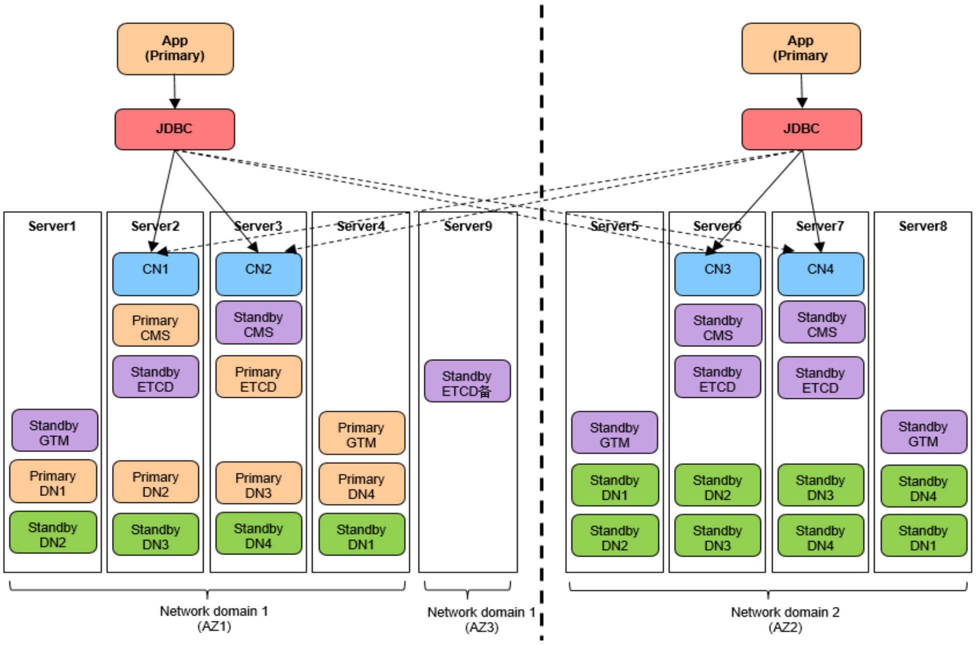
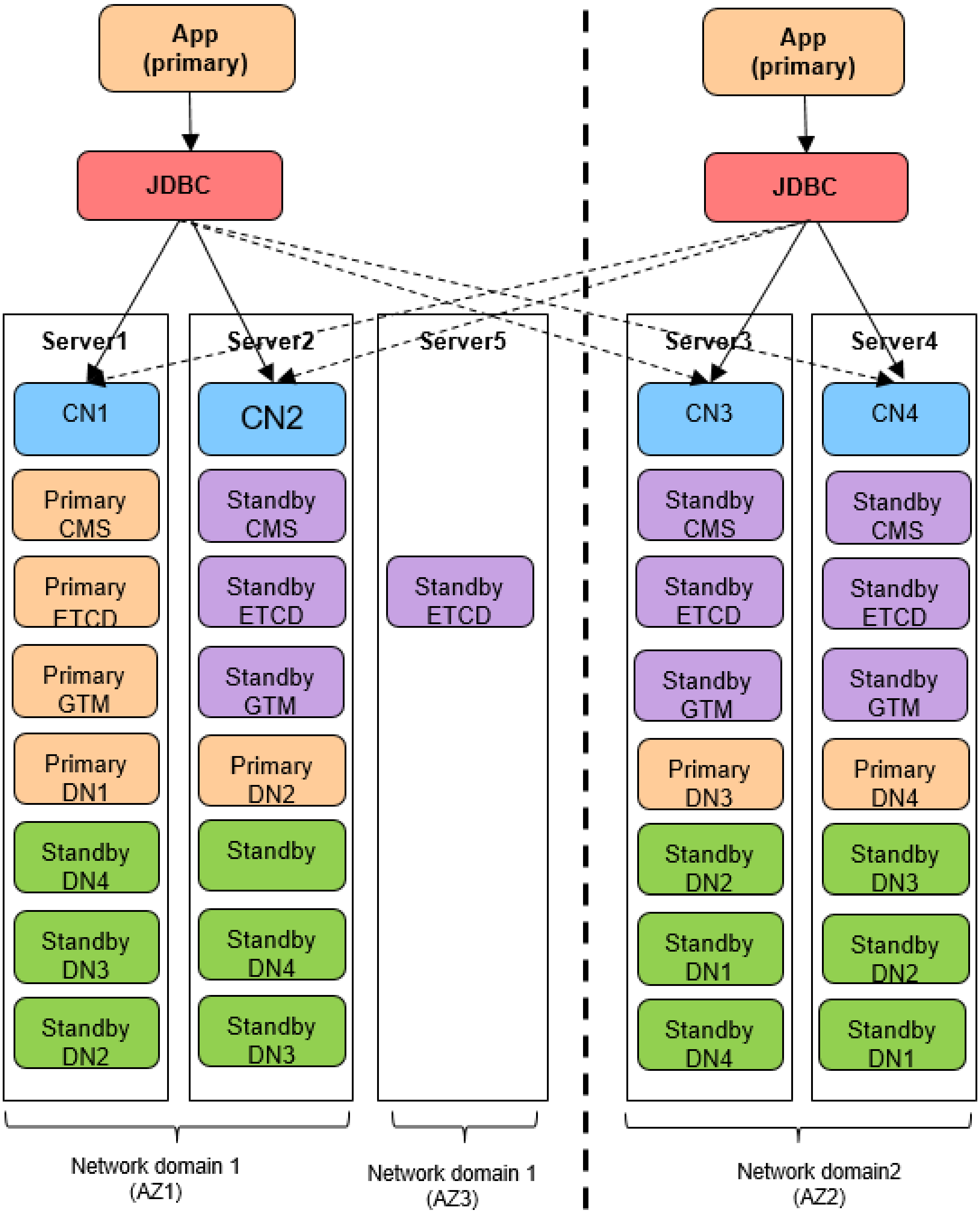


Figure 1-8 Intra-city 3-AZ 4-replica BMS deployment (data computing)



Intra-city HA scenario 2: intra-city 3-AZ 3-replica deployment

The intra-city, three-AZ deployment is supported. Three AZs are deployed in peer-to-peer mode and can access services. Any equipment room can achieve RPO of 0 and withstand network disconnections between equipment rooms.

- AZ 1, AZ 2, and AZ 3 have complete data and can access services at the same time to implement the three-active mode.

- In quorum-based replication between primary and standby DNs, there must be synchronous backup DNs across AZs and data will not be lost.
- If a standby DN is faulty, services are not interrupted. If the primary DN is faulty, a primary/standby switchover is automatically performed.
- This solution provides high availability for data center faults. If AZ1, AZ2 or AZ3 is faulty, all services in the faulty AZ are automatically switched to the other AZ. After the failover is complete, services become normal.

Figure 1-9 BMS/MCS deployment

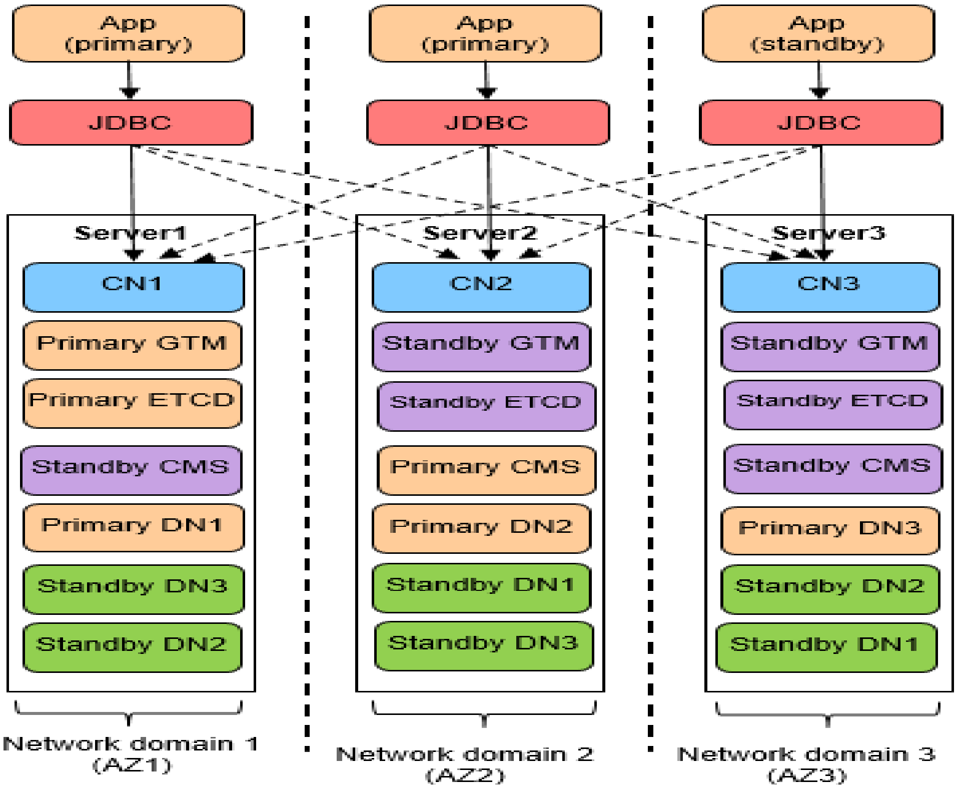
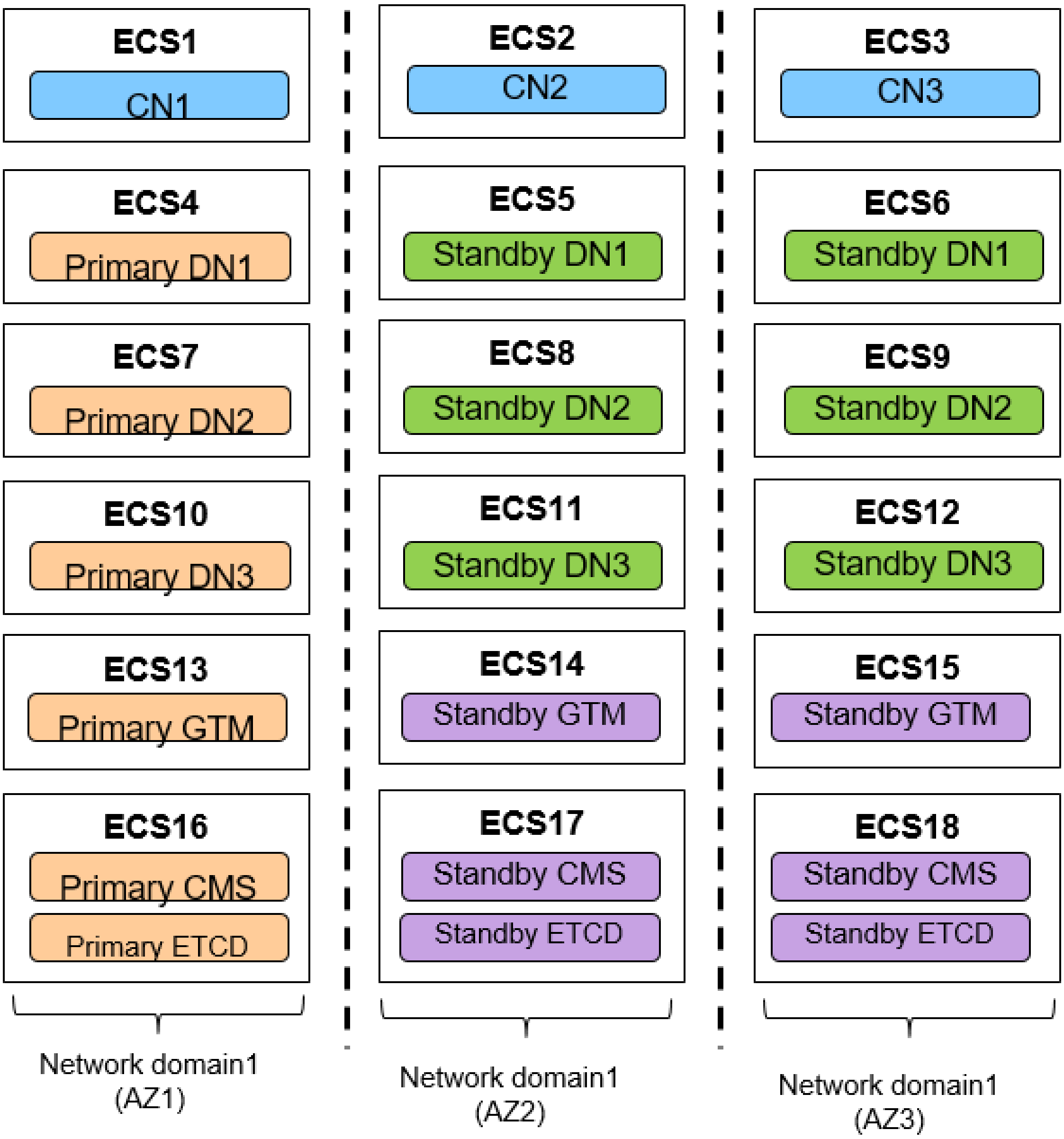


Figure 1-10 ECS deployment



Intra-city HA scenario 3: single-AZ 3-replica deployment

The single-AZ three-replica deployment helps defend against instance-level faults. This deployment is applicable to scenarios where data center DR is not required but some hardware faults need to be prevented.

A single AZ supports only three replicas. The reliability is 99.99% in three-replica or single-AZ scenarios. Therefore, in single-AZ scenarios, the reliability of the system will not be improved even if the number of replicas exceeds three.

- In primary/standby DN quorum replication, data is synchronized to at least one standby to ensure an RPO of zero.
- If a standby DN is faulty, services are not interrupted. If the primary DN is faulty, a primary/standby switchover is automatically performed.

- There are three copies of data. If one node is faulty, the system still has two copies of data. In addition, any standby node can be promoted to primary.
- The primary and standby DNs of a shard cannot be deployed on the same physical machine.

Figure 1-11 BMS/MCS deployment

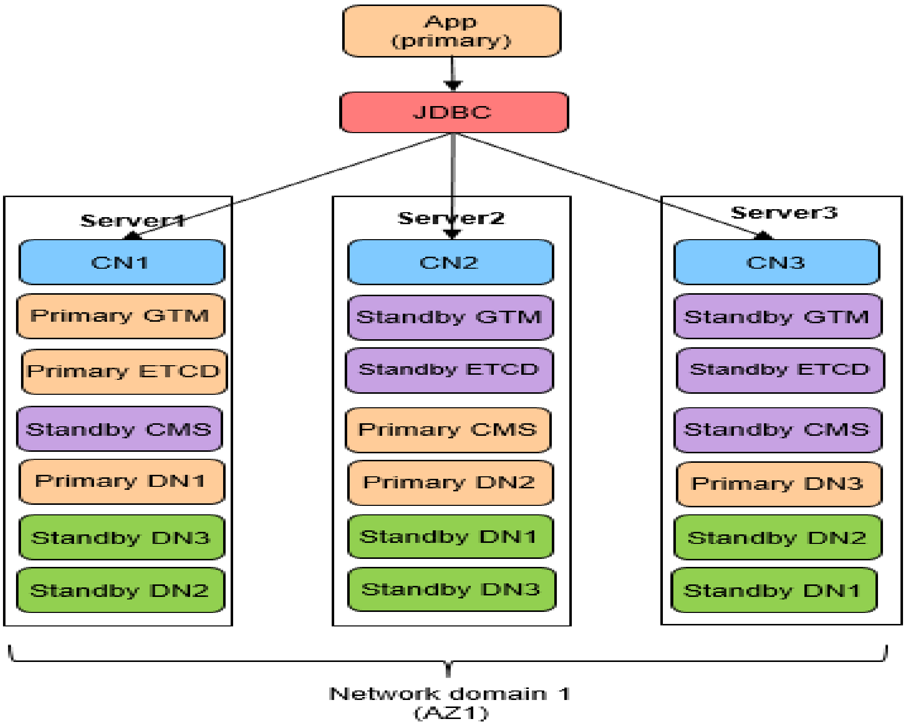
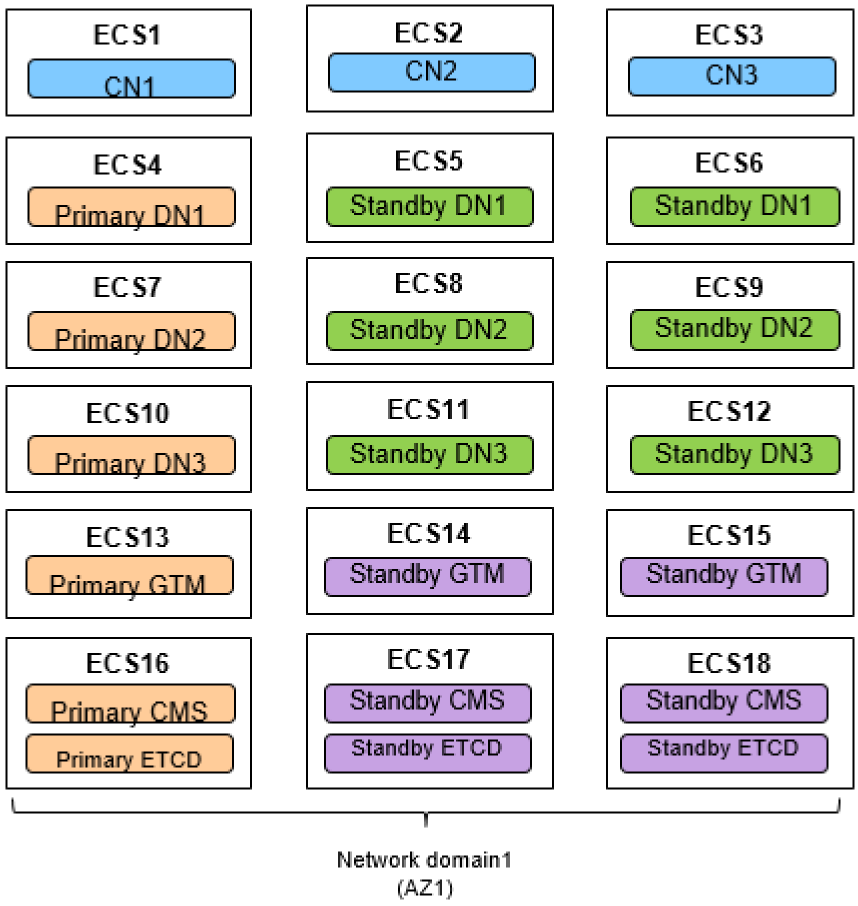


Figure 1-12 ECS deployment



Intra-city + remote DR

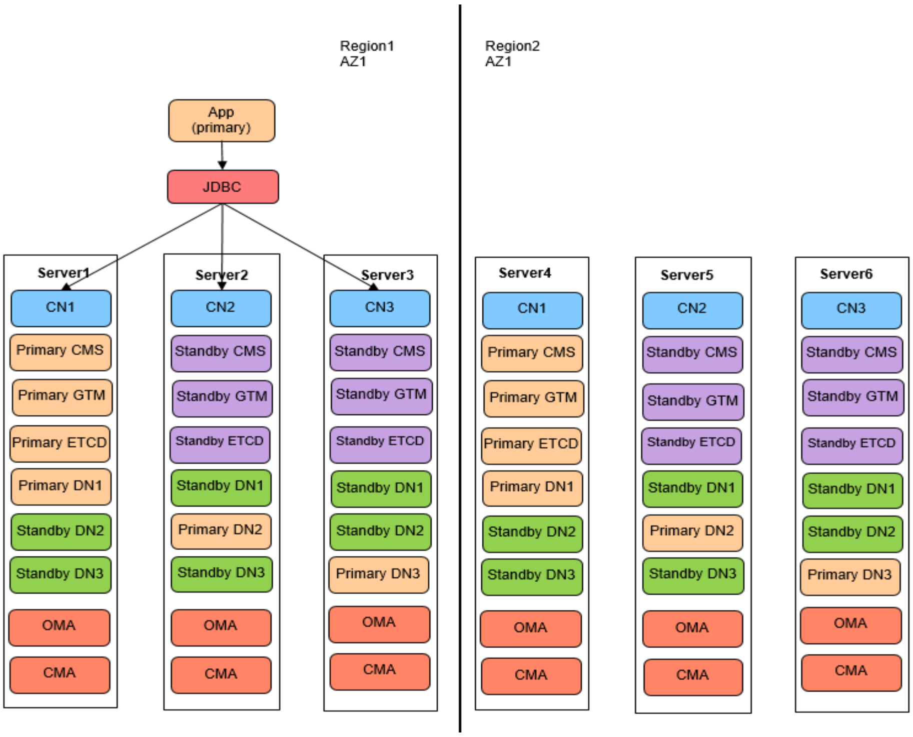
DR scenario 1: Intra-city 1-AZ and remote 1-AZ deployment

Two data centers are deployed in different cities and there are three replicas in each city. In this deployment, the intra-city data center can defend against instance-level faults and the cross-city data center can defend against region-level faults.

The reliability is 99.99% in three-replica or single-AZ scenarios. Therefore, in single-AZ scenarios, the reliability of the system will not be improved even if the number of replicas exceeds three.

- A complete database cluster is deployed in both the local and remote data centers.
- In primary/standby DN quorum replication, data is synchronized to at least one standby to ensure an RPO of zero.
- If a standby DN is faulty, services are not interrupted. If the primary DN is faulty, a standby DN is automatically promoted to the primary.
- There are three copies of data. If one node is faulty, the system still has two copies of data. In addition, any standby node can be promoted to primary.
- Cross-region DR requires manual switchover.

Figure 1-13 BMS/MCS deployment



DR scenario 2: Intra-city 3-AZ and remote 1-AZ deployment

Two data centers are deployed in the same city and one data center in another city. There are the same shards in city 1 and city 2, but city 1 supports four replicas and city 2 supports two replicas. A complete intra-city active-active deployment solution consists of two service AZs and one quorum AZ. Two service AZs are deployed in peer-to-peer mode, and every data center accesses services. The quorum AZ is responsible for auxiliary quorum to avoid SPOFs. It cannot access services. The deployment solution can achieve zero RPO and withstand network disconnections between data centers. GaussDB also supports 2-AZ, 4-replica (one primary and three standby DNs), and 1-quorum AZ deployment solution. Remote data center provides cross-region DR.

- A complete database cluster is deployed in both the local and remote data centers.
- In the same city, AZ1 and AZ2 have complete data. AZ3 serves as the quorum AZ. AZ1 and AZ2 can access services at the same time to implement dual-AZ active-active mode. If one AZ is faulty, the majority of ETCD nodes can survive, ensuring data consistency.
- In primary/standby DN quorum replication, data is synchronized to at least two standby DNs to ensure an RPO of zero.
- If a standby DN is faulty, services are not interrupted. If the primary DN is faulty, a standby DN is automatically promoted to the primary.
- There are four copies of data. If one node is faulty, the system still has three copies of data. In addition, any standby node can be promoted to primary.
- The intra-city DR provides high availability for data center faults. If AZ1, AZ2 or AZ3 is faulty, all services in the faulty AZ are automatically switched to the other AZ. After the switchover is complete, services can continue running. If

any of AZ1 or AZ2 and the quorum AZ are faulty, you need to manually start the faulty AZs.

- Cross-region DR requires manual switchover.

Figure 1-14 Intra-city 3-AZ, remote 1-AZ BMS/MCS deployment (standard)

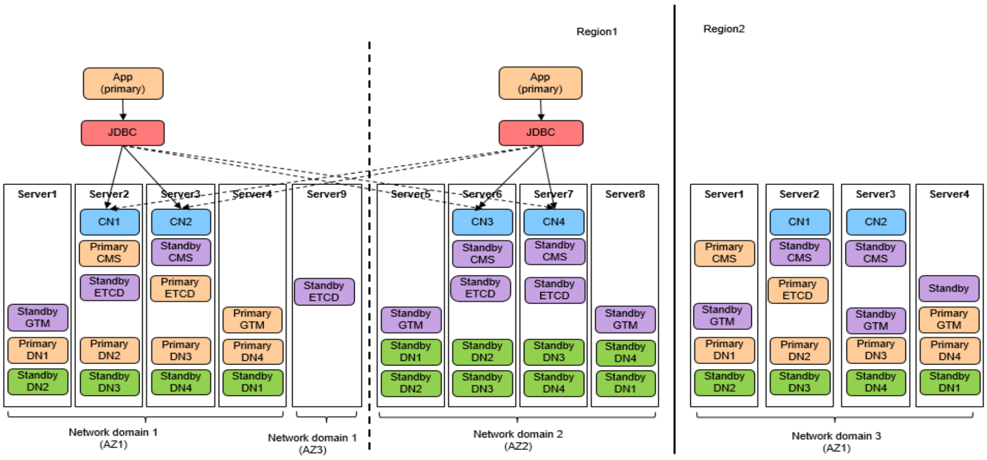
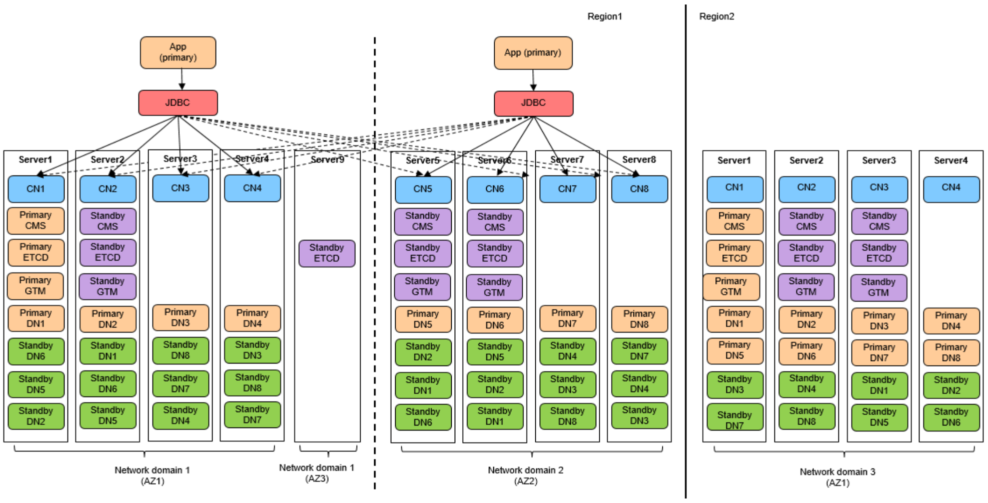


Figure 1-15 Intra-city 3-AZ and remote 1-AZ BMS deployment (data computing)



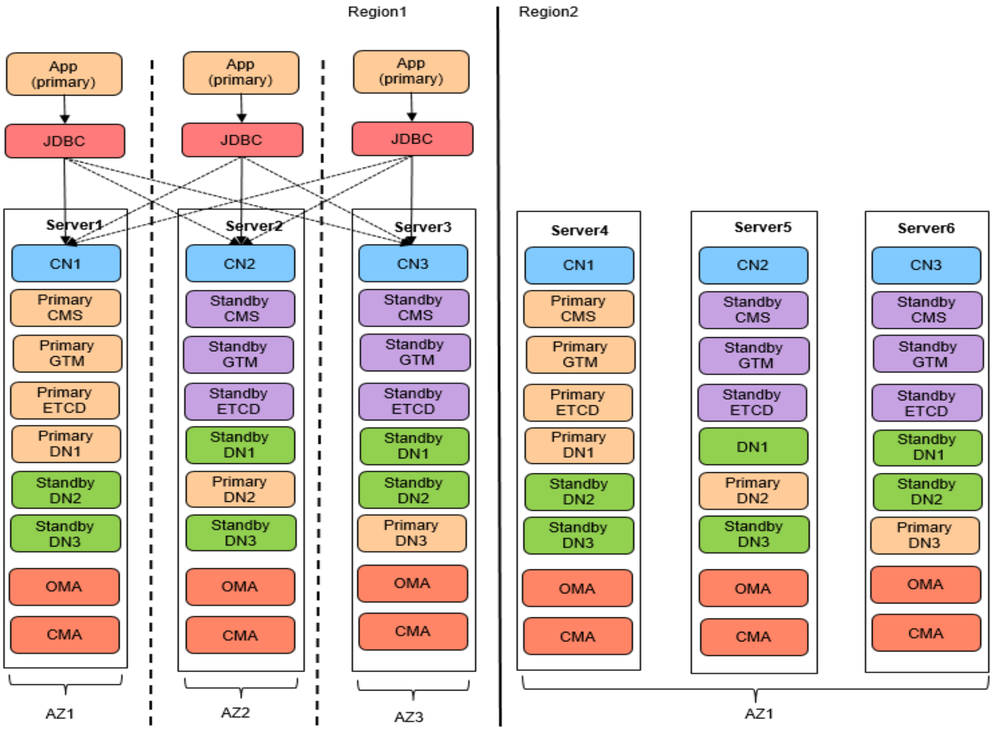
DR scenario 3: Intra-city 3-AZ and remote 1-AZ deployment

Among four data centers, three data centers are deployed in a city and a data center is deployed in another city. Three replicas (one primary and two DNs) are supported. In this deployment, the intra-city data centers can defend against instance-level faults and the cross-city data center can defend against region-level faults.

- A complete database cluster is deployed in both the local and remote data centers.
- In primary/standby DN quorum replication, data is synchronized to at least one standby to ensure an RPO of zero.
- If a standby DN is faulty, services are not interrupted. If the primary DN is faulty, a standby DN is automatically promoted to the primary.

- There are three copies of data. If one node is faulty, the system still has two copies of data. In addition, any standby node can be promoted to primary.
- The intra-city DR provides high availability for data center faults. If AZ1, AZ2 or AZ3 is faulty, all services in the faulty AZ are automatically switched to the other AZ. After the switchover is complete, services can continue running.
- Cross-region DR requires manual switchover.

Figure 1-16 BMS/MCS deployment



1.8.2 Primary/Standby Deployment in Huawei Cloud Stack

GaussDB supports primary/standby deployment in a Huawei Cloud Stack environment.

Deployment	Replicas	AZ	Description	Deployment Method
Intra-city HA	4	3	Two service AZs and one quorum AZ. Two replicas are symmetrically deployed in each service AZ.	● MCS container + local disk storage ● ECS+Ultra-high I/O ● BMS (centralized gateway, the ratio of BMGW to BMS does not exceed 2:30) + local disk storage

Deployment	Replicas	AZ	Description	Deployment Method
	3	1	Three replicas deployed in one service AZ	● MCS container + local disk storage ● ECS+Ultra-high I/O ● BMS (centralized gateway, the ratio of BMGW to BMS does not exceed 2:30) + local disk storage
	3	3	Each replica for each service AZ	● MCS container + local disk storage ● ECS+Ultra-high I/O ● BMS (centralized gateway, the ratio of BMGW to BMS does not exceed 2:30) + local disk storage
	3	1	All components deployed in a single service AZ	ECS+Ultra-high I/O
	3	3	Three intra-city service AZs, with one replica each service AZ	ECS+Ultra-high I/O
Intra-city HA + remote DR	6	2	One intra-city service AZ with three replicas One remote service AZ with three replicas	● MCS container + local disk storage ● ECS+Ultra-high I/O ● BMS (centralized gateway, the ratio of BMGW to BMS does not exceed 2:30) + local disk storage
	6	4	Three intra-city service AZs, with one replica each service AZ One remote service AZ with three replicas	● MCS container + local disk storage ● ECS+Ultra-high I/O ● BMS (centralized gateway, the ratio of BMGW to BMS does not exceed 2:30) + local disk storage

Deployment	Replicas	AZ	Description	Deployment Method
	8	6	Two intra-city service AZs (with two replicas in each service AZ) and one quorum AZ Two remote service AZs (with two replicas in each service AZ) and one quorum AZ	● MCS container + local disk storage ● ECS+Ultra-high I/O ● BMS (centralized gateway, the ratio of BMGW to BMS does not exceed 2:30) + local disk storage

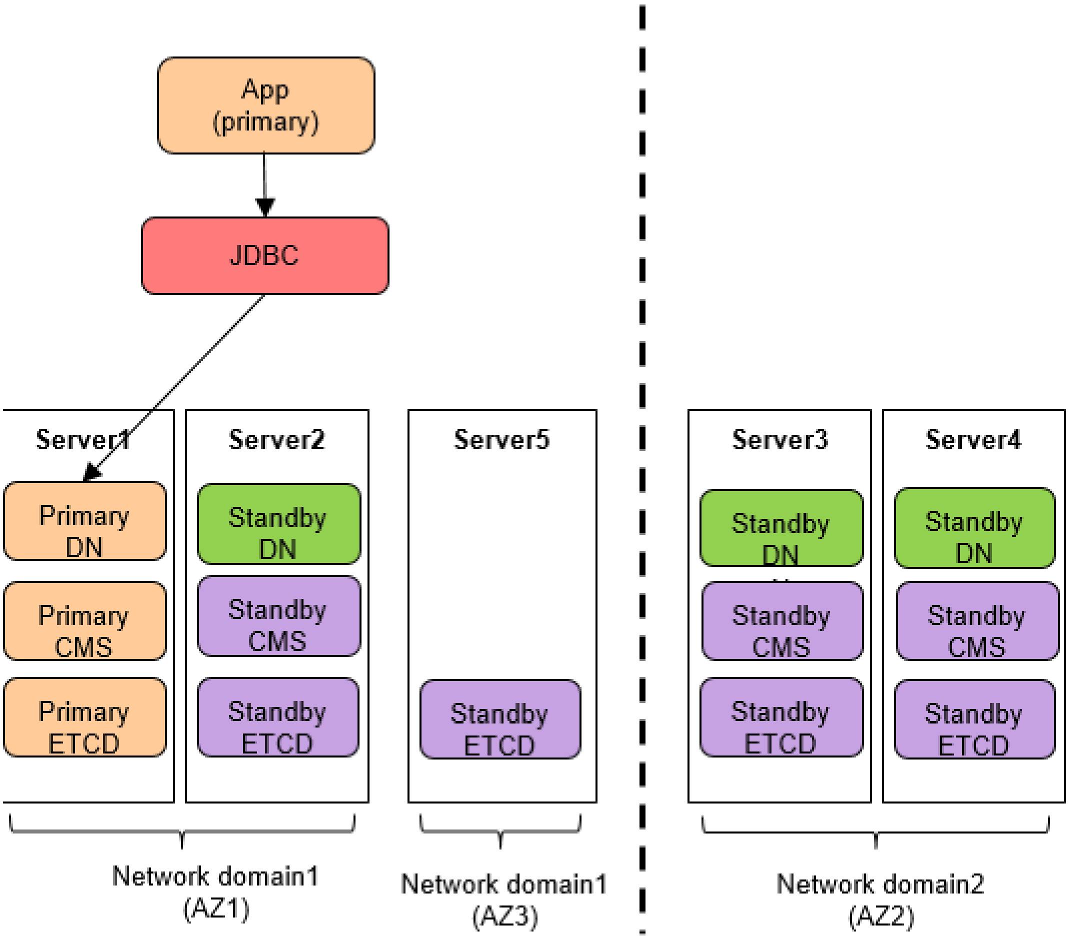
Intra-city HA deployment

Intra-city HA scenario 1: intra-city 3-AZ 4-replica deployment

A complete intra-city active-active deployment solution consists of two service AZs and one quorum AZ. Two service AZs are deployed in peer-to-peer mode, and the equipment rooms in AZs access services. The quorum AZ is responsible for auxiliary quorum to avoid SPOFs. It cannot access services. The deployment solution can achieve zero RPO and withstand network disconnections between data centers. GaussDB also supports 2-AZ, 4-replica (one primary and three standby DNs), and 1-quorum AZ deployment solution. All primary roles are deployed in the primary AZ by default.

- AZ 1 and AZ 2 have complete data, and AZ 3 functions as the third-party quorum node.
- AZ3 serves as the quorum AZ. If one AZ is faulty, the majority of ETCD nodes can survive, ensuring data consistency.
- In quorum-based replication between primary and standby DNs, there must be synchronous backup DNs across AZs and data will not be lost.
- If a standby DN is faulty, services are not interrupted. If the primary DN is faulty, a primary/standby switchover is automatically performed.
- This solution provides high availability for data center faults. If AZ1 or AZ2 is faulty, all services in the faulty AZ are automatically switched to the other AZ. After the switchover is complete, services can continue running.
- If any of AZ1 or AZ2 and the quorum AZ are faulty, you need to manually start the faulty AZs.

Figure 1-17 BMS/MCS deployment



Intra-city HA scenario 2: intra-city 3-AZ 3-replica deployment

The intra-city, three-AZ deployment is supported. Three AZs are deployed in peer-to-peer mode and can access services. The deployment solution can achieve zero RPO and withstand network disconnections between data centers.

1. AZ1, AZ2, and AZ3 have complete data.
2. In quorum-based replication between primary and standby DNs, there must be synchronous backup DNs across AZs and data will not be lost.
3. If a standby DN is faulty, services are not interrupted. If the primary DN is faulty, a primary/standby switchover is automatically performed.
4. This solution provides high availability for data center faults. If AZ1, AZ2 or AZ3 is faulty, all services in the faulty AZ are automatically switched to the other AZ. After the switchover is complete, services can continue running.

Figure 1-18 BMS/MCS deployment

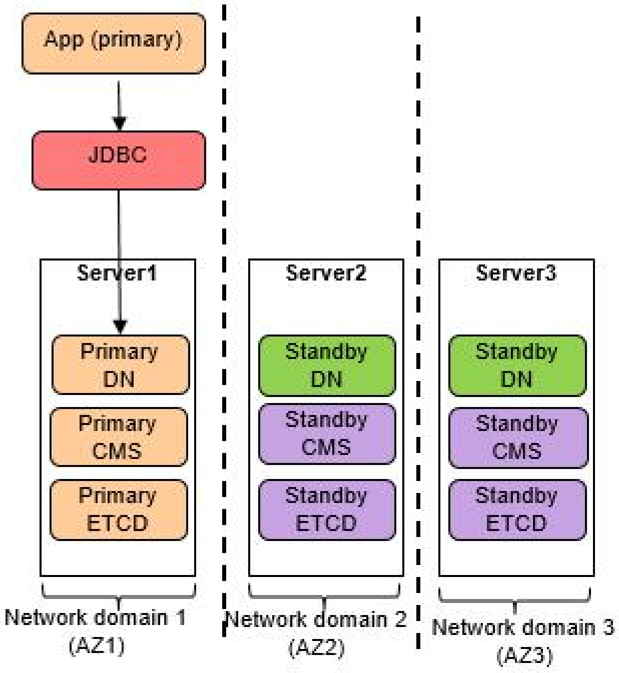
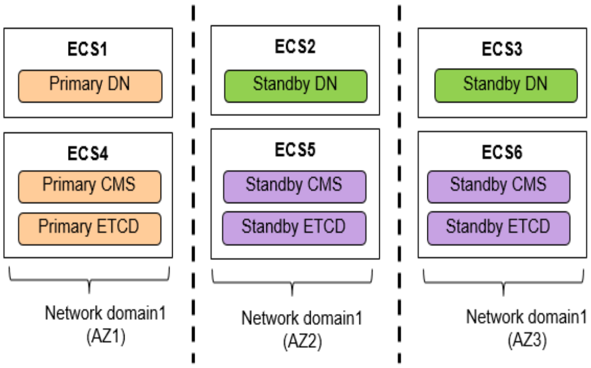


Figure 1-19 ECS deployment



Intra-city HA scenario 3: single-AZ 3-replica deployment

The single-AZ three-replica deployment helps defend against instance-level faults. This deployment is applicable to scenarios where data center DR is not required but some hardware faults need to be prevented.

A single AZ supports only three replicas. The reliability is 99.99% in three-replica or single-AZ scenarios. Therefore, in single-AZ scenarios, the reliability of the system will not be improved even if the number of replicas exceeds three.

- In primary/standby DN quorum replication, data is synchronized to at least one standby to ensure an RPO of zero.
- If a standby DN is faulty, services are not interrupted. If the primary DN is faulty, a primary/standby switchover is automatically performed.
- There are three copies of data. If one node is faulty, the system still has two copies of data. In addition, any standby node can be promoted to primary.

Figure 1-20 BMS/MCS deployment

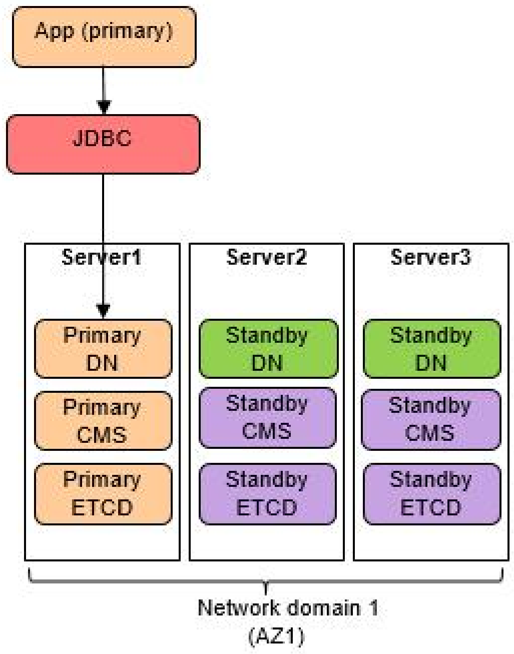
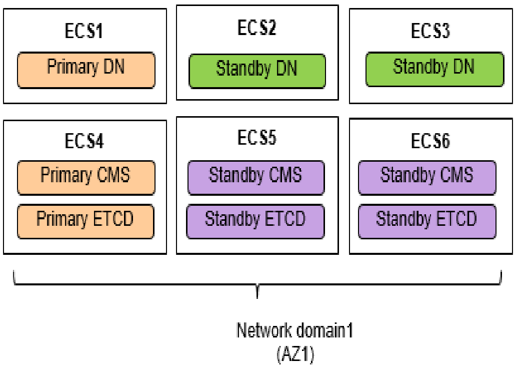


Figure 1-21 ECS deployment



Intra-city + remote DR

DR scenario 1: Intra-city 1-AZ and remote 1-AZ deployment

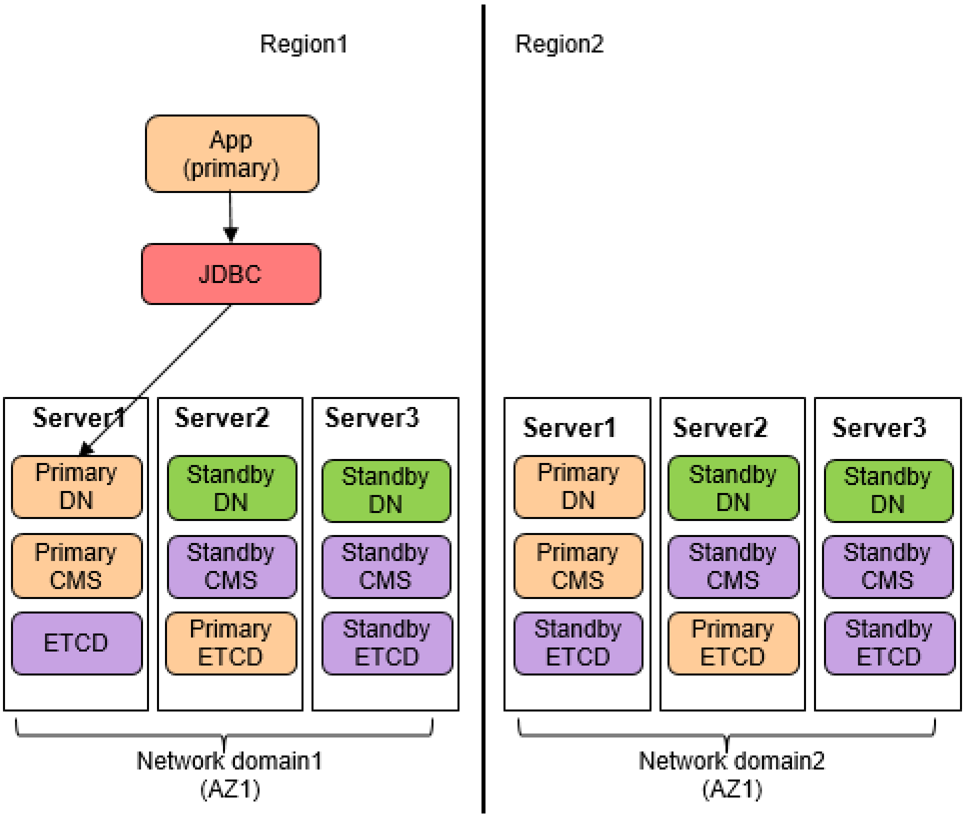
Two data centers are deployed in different cities and there are three replicas (one primary and two standby DNs) in each city. In this deployment, the intra-city data center can defend against instance-level faults and the cross-city data center can defend against region-level faults.

The reliability is 99.99% in three-replica or single-AZ scenarios. Therefore, in single-AZ scenarios, the reliability of the system will not be improved even if the number of replicas exceeds three.

- A complete database cluster is deployed in both the local and remote data centers.
- In primary/standby DN quorum replication, data is synchronized to at least one standby to ensure an RPO of zero.

- If a standby DN is faulty, services are not interrupted. If the primary DN is faulty, a primary/standby switchover is automatically performed.
- There are three copies of data. If one node is faulty, the system still has two copies of data. In addition, any standby node can be promoted to primary.
- Cross-region DR requires manual switchover.

Figure 1-22 BMS/MCS deployment



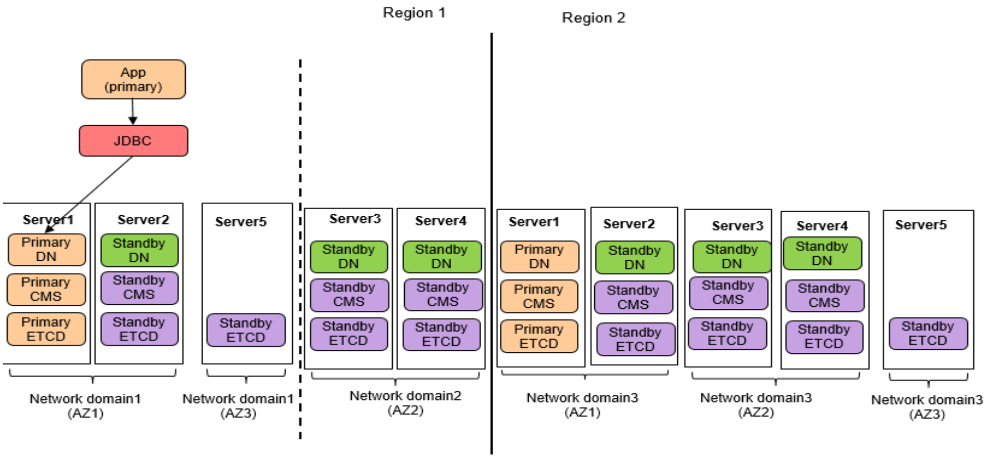
DR scenario 2: Intra-city 3-AZ and remote 3-AZ deployment

Two data centers are deployed in the same city and one data center in another city. Four replicas are supported in the two cities. A complete intra-city active-active deployment solution consists of two service AZs and one quorum AZ. Two service AZs are deployed in peer-to-peer mode, and every data center accesses services. The quorum AZ is responsible for auxiliary quorum to avoid SPOFs. It cannot access services. The deployment solution can achieve zero RPO and withstand network disconnections between data centers. GaussDB also supports 2-AZ, 4-replica (one primary and three standby DNs), and 1-quorum AZ deployment solution. Remote data center provides cross-region DR.

- A complete database cluster is deployed in both the local and remote data centers.
- In the same city, AZ1 and AZ2 have complete data. AZ3 serves as the quorum AZ. AZ1 and AZ2 can access services at the same time to implement dual-AZ active-active mode. If one AZ is faulty, the majority of ETCD nodes can survive, ensuring data consistency.

- In primary/standby DN quorum replication, data is synchronized to at least two standby DNs to ensure an RPO of zero.
- If a standby DN is faulty, services are not interrupted. If the primary DN is faulty, a standby DN is automatically promoted to the primary.
- There are four copies of data. If one node is faulty, the system still has three copies of data. In addition, any standby node can be promoted to primary.
- The intra-city DR provides high availability for data center faults. If AZ1, AZ2 or AZ3 is faulty, all services in the faulty AZ are automatically switched to the other AZ. After the switchover is complete, services can continue running. If any of AZ1 or AZ2 and the quorum AZ are faulty, you need to manually start the faulty AZs.
- Cross-region DR requires manual switchover.

Figure 1-23 BMS/MCS deployment



DR scenario 3: Intra-city 3-AZ and remote 1-AZ deployment

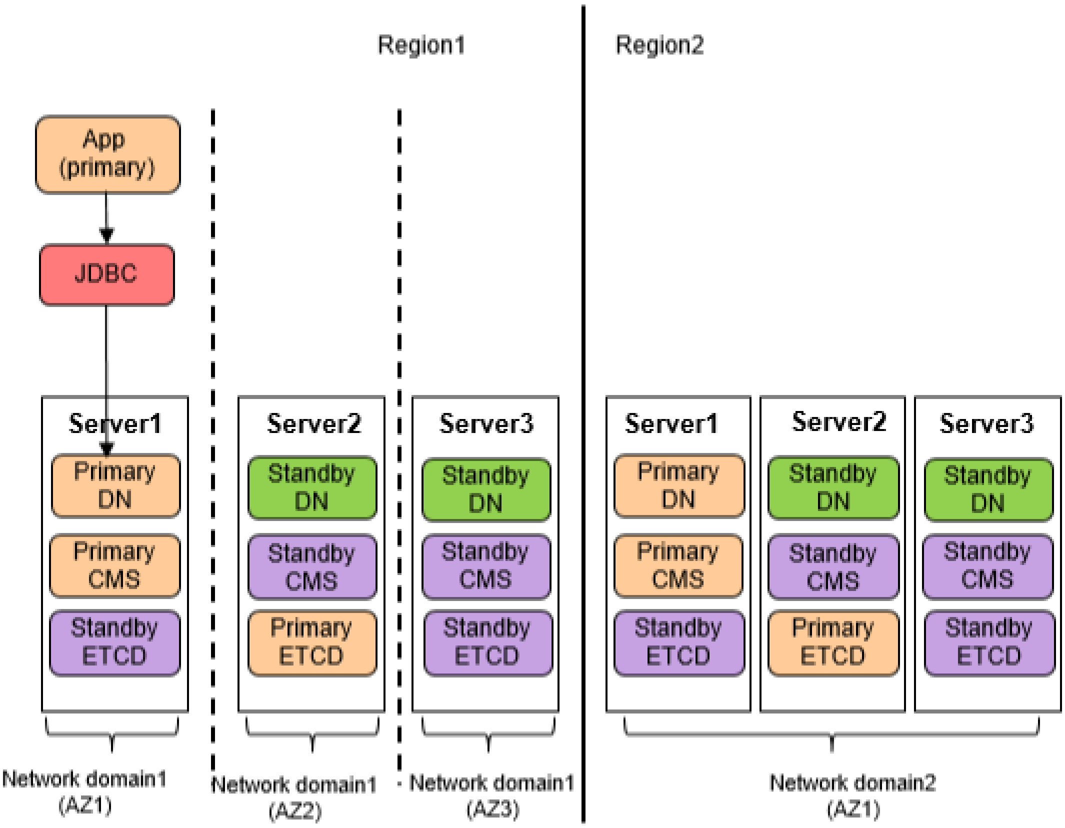
Among four data centers, three data centers are deployed in a city and a data center is deployed in another city. Three replicas (one primary and two DNs) are supported. In this deployment, the intra-city data centers can defend against instance-level faults and the cross-city data center can defend against region-level faults.

The reliability is 99.99% in three-replica or single-AZ scenarios. Therefore, in single-AZ scenarios, the reliability of the system will not be improved even if the number of replicas exceeds three.

- A complete database cluster is deployed in both the local and remote data centers.
- In primary/standby DN quorum replication, data is synchronized to at least one standby to ensure an RPO of zero.
- If a standby DN is faulty, services are not interrupted. If the primary DN is faulty, a primary/standby switchover is automatically performed.
- There are three copies of data. If one node is faulty, the system still has two copies of data. In addition, any standby node can be promoted to primary.

- The intra-city DR provides high availability for data center faults. If AZ1, AZ2 or AZ3 is faulty, all services in the faulty AZ are automatically switched to the other AZ. After the switchover is complete, services can continue running.
- Cross-region DR requires manual switchover.

Figure 1-24 BMS/MCS deployment



1.9 Core Database Technologies

1.9.1 Basic Functions Oriented to Application Development

Standard SQL

SQL is a standard computer language used to control the access to databases and manage data in databases. SQL standards are classified into core features and optional features. Most databases do not fully support SQL standards.

GaussDB supports most of the core features of SQL:2011 and some optional features.

The introduction of standard SQL provides a unified SQL interface for all database vendors, reducing the learning costs of users and application migration costs.

Standard Development Interfaces

Standard ODBC and JDBC interfaces are provided to ensure quick migration of user services to GaussDB.

Currently, the standard ODBC 3.5 and JDBC 4.0 interfaces are supported. The ODBC interface supports SUSE Linux, Windows 32-bit, and Windows 64-bit platforms. The JDBC interface supports all platforms.

Transactions

Transaction support refers to the system capability to ensure the atomicity, consistency, isolation, and durability (ACID) features of global transactions under the shared-nothing architecture. The two-phase commit (2PC) protocol is used to ensure transaction status consistency across nodes, avoiding the situation that a transaction is committed on some nodes but rolled back on others.

GaussDB supports globally strong-consistency distributed transactions to ensure the ACID feature of global transactions.

- Atomicity

  A transaction is an indivisible unit of work. Operations performed in a transaction must be all finished or have not been performed.
- Consistency

  Transactions must always keep the system in a consistent state.
- Isolation

  Transactions are isolated for execution, as if each of them is the only operation performed during the specified period planned by the system. If there are two transactions that are executed within the same period of time and performing the same function, the transaction isolation makes each of them regard itself as the only transaction using the system.
- Durability

  Durability indicates that after a transaction is complete, modifications made on the database by the transaction are durably saved in the database and will not be rolled back.

Global Transaction Manager (GTM) allocates a set of transaction IDs and commit transactions based on the 2PC protocol to ensure global transaction consistency. To submit a transaction, a pre-committing command is sent to each node. The transaction is formally committed after all the pre-committing commands succeed.

The supported transaction isolation levels are READ COMMITTED and REPEATABLE READ. The default isolation level is READ COMMITTED. ensuring no dirty data will be read.

Transactions are categorized into single-statement transactions and transaction blocks. Their basic interfaces are as follows:

- Start transaction
- Commit
- Rollback

Functions and Stored Procedures

Functions and stored procedures are important database objects. They encapsulate SQL statement sets used for certain functions so that the statements can be easily invoked.

A stored procedure is a combination of SQL, PL/SQL, and Java statements. Stored procedures can move the execution code from the application to the database. In this way, the code can be used by multiple programs.

1. Users are allowed to modularize program design and encapsulate SQL statement sets, easy to invoke.
2. The compilation results of stored procedures are cached to accelerate SQL statement set execution.
3. System administrators restrict the permission for executing a specific stored procedure and control access to the corresponding type of data. This prevents access from unauthorized users and ensures data security.
4. To process SQL statements, the stored procedure process assigns a memory fragment to store context association. Cursors are handles or pointers to context areas. With cursors, stored procedures can control alterations in context areas.
5. Six-level exception information is supported to facilitate the debugging of stored procedures.
6. Retry is a process in which the database tries again in case of a SQL statement or stored procedure (including anonymous block) execution failure, improving the execution success rate and user experience. The database checks the error codes and retry configuration to determine whether to retry.
7. The stored procedure provides seven advanced function packages, including DBMS_LOB, DBMS_RANDOM, DBMS_OUTPUT, UTL_RAW, DBMS_JOB, DBMS_SQL, and UTL_FILE.

GaussDB supports functions and stored procedures compliant with the SQL standards. The stored procedures are compatible with certain Oracle stored procedure syntax, improving their usability.

PostgreSQL API Compatibility

PostgreSQL ecosystem tools can seamlessly connect to PostgreSQL clients and are compatible with standard PostgreSQL interfaces.

SQL Hints

SQL hints are supported, which can override any execution plan and thus improve SQL query performance.

In plan hints, you can specify a join order; join, stream, and scan operations, the number of rows in a result, and redistribution skew information to tune an execution plan, improving query performance.

PL/Java

With the GaussDB PL/Java functions, you can choose your favorite Java IDE to write Java methods and install the JAR files containing these methods into the

GaussDB database before invoking them. GaussDB PL/Java is developed based on the open-source data PL/Java 1.5.2.

- Java UDFs can implement simple Java computing. However, do not encapsulate services in Java UDFs.
- Users are not advised to connect to a database in any way (for example, JDBC) in Java functions.
- Currently, only data types listed in **Table 1-10** are supported. Other data types, such as user-defined data types and complex data types (for example, Java array and its derived types) are not supported.
- Currently, UDAF and UDTF are not supported.

Table 1-10 PL/Java mapping for default data types

GaussDB	Java
BOOLEAN	boolean
"char"	byte
bytea	byte[]
SMALLINT	short
INTEGER	int
BIGINT	long
FLOAT4	float
FLOAT8	double
CHAR	java.lang.String
VARCHAR	java.lang.String
TEXT	java.lang.String
name	java.lang.String
DATE	java.sql.Timestamp
TIME	java.sql.Time (stored value treated as local time)
TIMETZ	java.sql.Time
TIMESTAMP	java.sql.Timestamp
TIMESTAMPTZ	java.sql.Timestamp

GDS

Gauss Data Service (GDS) is a fast data import and export service provided by GaussDB. It fully utilizes the computing and I/O capabilities of all nodes to achieve the maximum import and export speeds in parallel mode. GDS can manage data sources in various formats, including TEXT, CSV, and FIXED. It is suitable for scenarios where a large amount of data needs to be imported and exported, for example, cluster data migration.

Error Tolerance Mechanism of Copy Interface

GaussDB provides the encapsulated copy error tables for creating functions and allows users to specify error tolerance options when using the **Copy From** statement. In this way, errors related to parsing, data format, and character set during the execution of the **Copy From** statement are recorded in the error table instead of being reported and interrupted. Even if a small amount of data in the target file of **Copy From** is incorrect, the data can be imported to the database. Users can locate and rectify the fault in the error table later.

Data Export to OBS

After obtaining the OBS server, OBS bucket, OBS directory, access key, and secret access key, users can create a write-only foreign table using the GSOBS protocol and export the required data from the database to the corresponding location on the OBS server by using the **INSERT INTO SELECT** statement.

1.9.2 High Performance

Distributed CBO Optimizer

The GaussDB optimizer is a typical Cost-based Optimization (CBO). By using CBO, the database calculates the number of tuples and the execution cost for each step under each execution plan based on the number of table tuples, column width, null record ratio, and characteristic values, such as distinct, MCV, and HB values, and certain cost calculation methods. The database then selects the execution plan that takes the lowest cost for the overall execution or for the return of the first tuple.

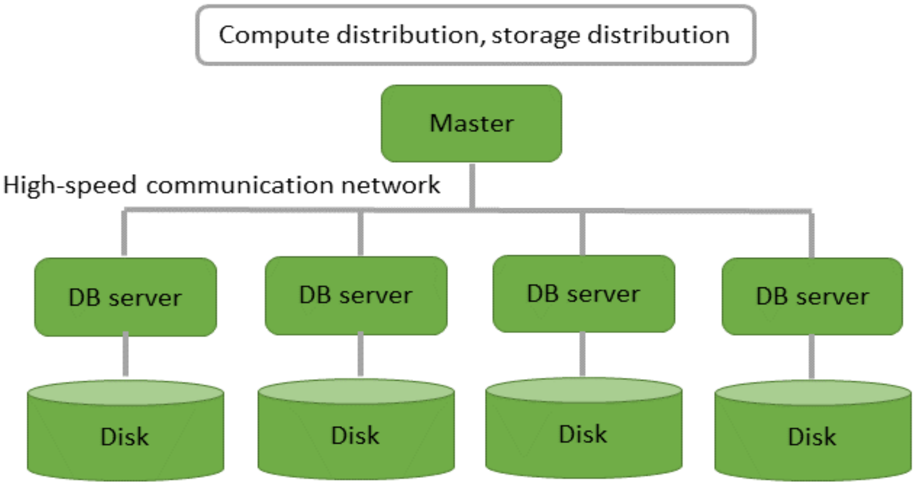
The GaussDB CBO optimizer can select the most efficient execution plan among multiple distributed plans based on the cost to meet customer service requirements to the maximum extent.

Fully Parallel Distributed Execution

GaussDB adopts the massively parallel processing (MPP) architecture. This architecture features parallel task execution, distributed data storage (localization), distributed computing, private resources (CPU, memory, disk, and network), scale-out, and shared nothing. Therefore, GaussDB can process large-scale parallel data.

In addition to the MPP architecture, GaussDB has the streaming computing framework to enhance the data exchange capability between all compute nodes. Currently, almost all database operations, such as data scanning, table joining, and data aggregation, can be executed in fully parallel distributed mode. In addition, GaussDB supports scale-out of up to 256 shards. Compared with traditional databases, it has great advantages in computing capabilities.

Figure 1-25 MPP architecture


Compute distribution, storage distribution
Master
High-speed communication network
DB server
DB server
DB server
DB server
Disk
Disk
Disk
Disk

High-Performance Transaction Processing

Consistency and performance are the two cores of distributed transaction processing. Based on different service application scenarios, GaussDB optimizes the distributed transaction processing performance and consistency to the maximum content. Currently, the following transaction processing modes are supported:

- In GTM-Lite mode, distributed transactions are strongly consistent. GTM is lightweight to the maximum extent and provides good performance in OLTP scenarios.
- In GTM-free mode, distributed transactions are finally consistent. The central transaction management node does not participate in transaction management, eliminating the GTM single-point bottleneck and achieving higher transaction processing performance. However, in terms of consistency, external read consistency is ensured after all transactions are executed. Strong consistency read of distributed transactions is not supported. Transaction consistency that depends on query results, such as insert into select * from, is not supported. Write operations that are split into multiple statements are not supported. Write operations that involve multiple nodes are not supported. The GTM is not connected. For perfect OLTP sharding scenarios, the performance is the best when the central node is removed.

High-Speed Parallel Data Loading

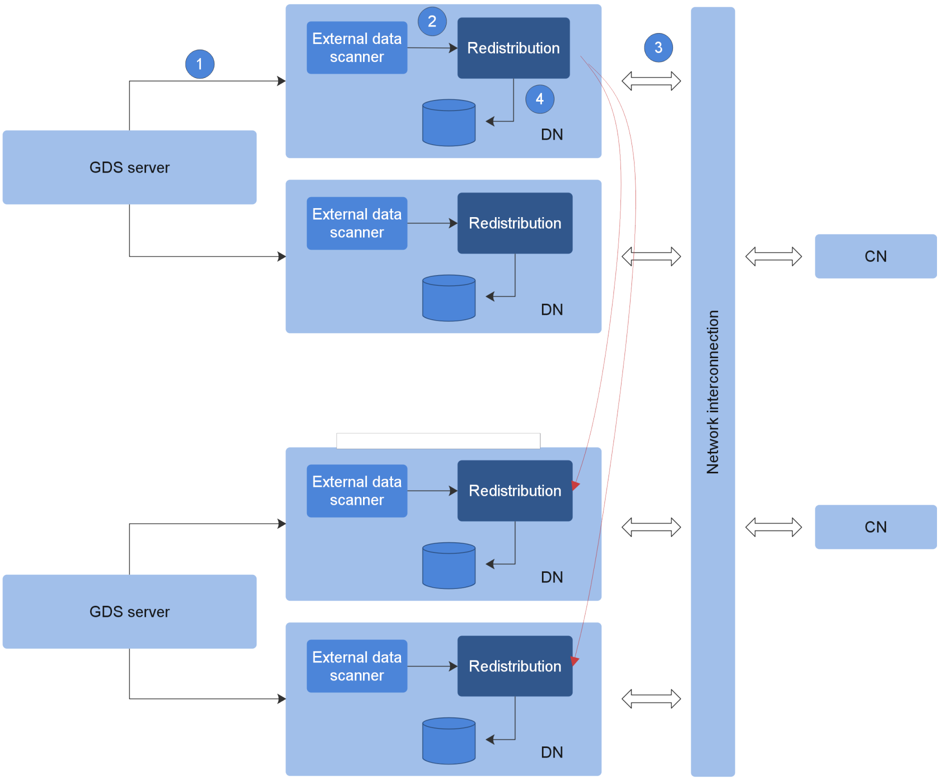
The key to importing data in parallel involves fully leveraging the computing and I/O capabilities of all nodes to maximize the import speed. Data is imported to GaussDB using a specific format (CSV, TEXT or FIXED).

Compared with the traditional method of inserting data one by one using the **INSERT** statement, the high-speed and parallel data import mode improves the data import performance. The principle is as follows:

- CNs only plan and distribute tasks, and transfer data import tasks to DNs. In this case, CNs are released to process external requests.
- The computing capability and network bandwidth of all the DNs are fully utilized, improving data import performance.

The following uses the Hash distribution policy as an example to describe how data is imported to GaussDB. **Figure 1-26** shows the parallel data importing process. Create an internal table and a foreign table in the database. Specify the Hash distribution key for the internal table and specify the data source for the foreign table. After data import starts, the GDS splits data into data blocks of a fixed size based on the data source specified in the foreign table and sends the data blocks to each DN. Each DN processes data blocks in parallel. Each time a data tuple is parsed, a physical location of storage is determined according to a Hash value calculated according to the distribution key of the data tuple. If the Hash is on another network node, the data needs to be redistributed to the target DN through the network. If the Hash is on the local node, the data is stored on the local DN. Data is written to the corresponding data file after reaching the node where the Hash is located. GDS interacts with CNs and DNs to implement high-speed parallel data loading.

Figure 1-26 Importing parallel data



SMP Parallel Technology

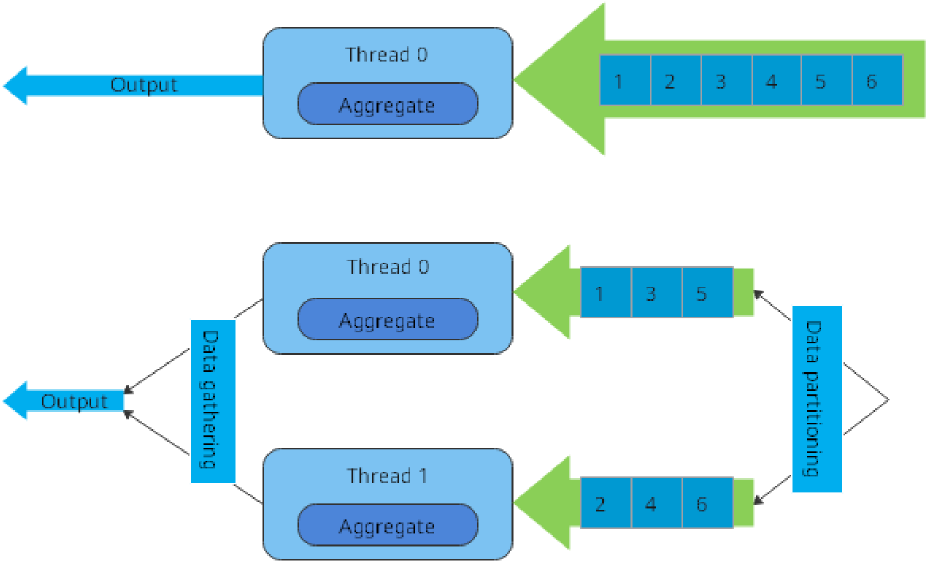
The Symmetric Multi-Processing (SMP) parallel technology of GaussDB uses the multi-core CPU architecture of a computer to implement multi-thread parallel computing, fully using CPU resources to improve query performance.

The modern computer architecture is developing towards multiple cores and large memory. The CPU of a machine can have dozens or even hundreds of cores. To better utilize CPU resources and obtain better performance in the big data analysis field, the parallel query execution technology has become a necessary part of modern database management systems. The database requires not only the Massively Parallel Processing (MPP) parallel technology to achieve good horizontal scale-out, but also the SMP parallel technology to achieve better scale-up on a single node. SMP can improve the database performance from two aspects. On the one hand, SMP can significantly reduce the execution time of a single query. On the other hand, SMP can improve the system throughput in the same time period, thereby effectively improving the utilization of system resources.

SMP parallel technology uses the mechanism of multi-thread and multi-subtask parallel execution to fully and efficiently use the system computing resources. Obviously, the SMP multi-thread lightweight execution mode can solve the problem of MPP architecture deployment.

1. First, SMP parallel execution refers to the parallel execution of tasks at the thread level. Theoretically, the number of subtasks that can be concurrently executed can reach the upper limit of the number of cores on the physical server.
2. Second, SMP parallel threads are in the same process, and data can be exchanged directly through the memory, without occupying network connection and bandwidth. This reduces the impact of network factors that restrict the performance improvement of the MPP system.
3. Finally, because the parallel subtasks do not need to be attached to another background worker thread after being started, the utilization rate of system computing resources can be increased effectively.

Figure 1-27 SMP parallel execution



LLVM Dynamic Compilation Technology

The Low Level Virtual Machine (LLVM) technology provides an intermediate layer of a complete compilation system, and extracts and optimizes intermediate representation (IR) from a compiler. The optimized IR is then converted and linked to an assembly language of a target platform. The LLVM can also generate relocatable code during compilation, linking, or even running.

Based on the query execution plan tree, with the library functions provided by LLVM, GaussDB moves the process of determining the actual execution path from the executor phase to the execution initialization phase. In this way, problems such as function calling, logic condition branch determination, and a large amount of data reading that are related to the original query execution are avoided, to improve the query performance.

Adaptive Compression

Currently, mainstream databases usually use the data compression technology. Various compression algorithms are used for different data types. If pieces of data of the same type have different characteristics, their compression algorithms and results will also be different. Adaptive compression chooses the suitable compression algorithm for data based on the data type and characteristics, achieving high performance in compression ratio, import, and query.

Currently, the database has implemented various compression algorithms, including RLE, DELTA, BYTEPACK/BITPACK, LZ4, ZLIB, and LOCAL DICTIONARY. **Table 1-11** lists data types and the compression algorithms suitable for them.

Table 1-11 Mapping between data types and compression algorithms

-	RLE	DELTA	BITPACK/BYTEPACK	LZ4	ZLIB	LOCAL DICTIONARY
Smallint/int/bigint/Oid Decimal/real/double Money/time/date/ timestamp	√	√	√	√	√	-
Tinterval/interval/Time with time zone/	-	-	-	-	√	-
Numeric/char/varchar/text/nvarchar2 and other supported data types	√	√	√	√	√	√

For example, large integer compression of mobile number-like character strings, large integer compression of the numeric type, and adjustment of the compression algorithm compression level are supported.

Partition

In the distributed GaussDB system, data is partitioned horizontally on a node using a specified policy. This operation splits a table into multiple partitions that are not overlapped.

In common scenarios, a partitioned table has the following advantages over an ordinary table:

- High query performance: The system queries only the relevant partitions rather than the entire table, improving the query efficiency.
- High availability: A faulty partition does not affect data availability in other partitions.
- Easy maintenance: If a certain partition in a table is faulty, only this partition rather than the entire table needs to be repaired.

Currently, the GaussDB database supports range partitioning. Data is mapped to each partition based on the range. The range is determined by the partition key specified when the partitioned table is created. This partitioning mode is most commonly used.

With the range partitioning function, the database divides a record, which is to be inserted into a table, into multiple ranges using one or multiple columns and creates a partition for each range to store data. Partition ranges do not overlap. If you configure the **PARTITION** parameter when running the **CREATE TABLE** statement, data in the table will be partitioned.

Users can modify partition keys as needed during table creation to make the query result stored in the same or least partitions (called partition pruning), to obtain consecutive I/O to improve the query performance.

In actual services, time is often used as a filter criterion for query objects. Therefore, users can select the time column as the partition key. The key value range can be adjusted based on the total data volume and the data volume queried at a time.

Advanced Analysis Functions

GaussDB provides window functions for advanced data analysis and processing. The window function groups the data in a table in advance. Each row belongs to a specific group. Then, a series of association analysis calculations are performed on the group. In this way, some attributes of each tuple in the set and association information with other tuples can be mined.

The following is an example of the window analysis function:

Analyze the comparison between the salary of each person in a department and the average salary of the department.

```
SELECT depname, empno, salary, avg(salary) OVER (PARTITION BY depname) FROM empsalary;
depname | empno | salary | avg
-----------+-------+--------+-----------------------
develop | 11 | 5200 | 5020.0000000000000000
develop | 7 | 4200 | 5020.0000000000000000
develop | 9 | 4500 | 5020.0000000000000000
develop | 8 | 6000 | 5020.0000000000000000
develop | 10 | 5200 | 5020.0000000000000000
personnel | 5 | 3500 | 3700.0000000000000000
personnel | 2 | 3900 | 3700.0000000000000000
sales | 3 | 4800 | 4866.6666666666666667
sales | 1 | 5000 | 4866.6666666666666667
sales | 4 | 4800 | 4866.6666666666666667
(10 rows)
```

The analysis function **avg(salary) OVER (PARTITION BY depname)** easily calculates each employee's salary and the average salary of the department.

Currently, the system supports the **row_number()**, **rank()**, **dense_rank()**, **percent_rank()**, **cume_dist()**, **ntile()**, **lag()**, **lead()**, **first_value()**, **last_value()**, and **nth_value()** analysis functions.

Data Skew Optimization Technology

Data skew breaks the balance among nodes in the distributed MPP architecture. If the amount of data stored or processed by a node is much greater than that by other nodes, the following problems may occur:

- Storage skew severely limits the system capacity. The skew on a single node hinders system storage utilization.
- Computing skew severely affects performance. The data to be processed on the skew node is much more than that on other nodes, deteriorating overall system performance.
- Data skew severely affects the scalability of the MPP architecture. During storage or computing, data with the same values is often placed on the same node. Therefore, even if we add nodes after a data skew occurs, the skew data (data with the same values) is still placed on the node and affects the system capacity or performance bottleneck.

GaussDB provides a complete solution for data skew, including storage and computing skew.

- To optimize the storage layer, GaussDB provides various views for viewing data skew.
- In terms of computing skew, GaussDB proposes Runtime Load Balance Technology (RLBT) to identify possible skew values based on statistics or hints, and then process skew data and non-skew data separately. For example, during join, non-skew data is redistributed based on hash, and skew data is redistributed based on round robin.

SQL Bypass

In a typical OLTP scenario, simple queries account for a large proportion. This type of queries involves only single tables and simple expressions. To accelerate such query, the SQL bypass framework is proposed. After simple mode judgment is performed on such query at the parse layer, the query enters a special execution path and skips the classic execution framework, including operator initialization and execution, expression, and projection. Instead, it directly rewrites a set of simple execution paths and directly invokes storage APIs, greatly accelerating the execution of simple queries.

HyperLogLog

HyperLoglog (HLL) is an approximation algorithm for efficiently counting the number of distinct values in a dataset. It features faster computing and lower space usage. You only need to store HLL data structures instead of datasets. When new data is added to a dataset, make hash calculation on the data and insert the result to an HLL. Then, you can obtain the final result based on the HLL.

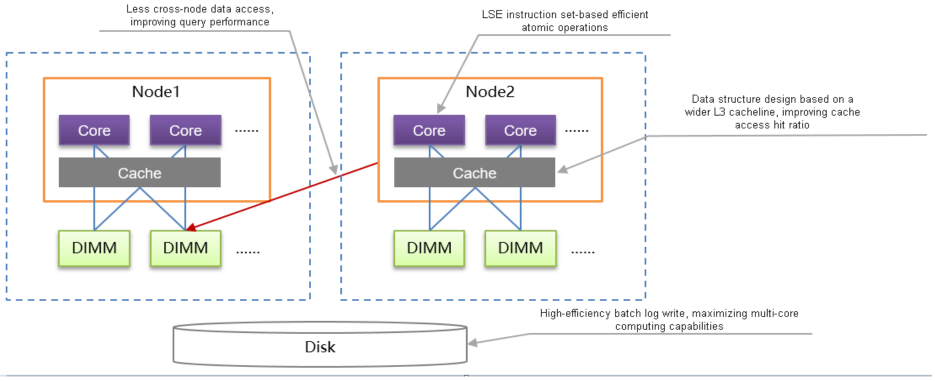
HLL has advantages over others in the computing speed and storage space requirement. In terms of time complexity, the Sort algorithm needs to sort at least O(n log n) time. Although the Hash algorithm can obtain the result by scanning the entire table O(n) time, the storage space is as follows: Both the Sort and Hash algorithms need to store the original data before collecting statistics, which consumes a large amount of storage space. For the HLL, the original data does not need to be stored, and only the HLL data structure needs to be maintained. Therefore, the occupied space is always at the 1280-byte constant level.

GaussDB uses the distributed HLL architecture. DNs calculate the HLL and summarize the results on CNs, avoiding CN calculation bottlenecks.

Kunpeng NUMA Architecture Optimization

1. Based on the multi-core NUMA architecture of the Kunpeng processor, GaussDB optimizes the NUMA architecture to reduce the cross-core memory access latency and maximize the multi-core Kunpeng computing capability. The key technologies include redo log batch insertion, NUMA distribution of hotspot data, and Clog partitions, greatly improving the processing performance of the TP system.
2. Based on the ARMv8.1 architecture used by the Kunpeng chip, GaussDB uses the LSE instruction set to implement efficient atomic operations, effectively improving the CPU usage, multi-thread synchronization performance, and XLog write performance.
3. Based on the wider L3 cache line provided by the Kunpeng chip, GaussDB optimizes hotspot data access, effectively improving the cache access hit ratio, reducing the cache consistency maintenance overhead, and greatly improving the overall data access performance of the system.

Figure 1-28 Kunpeng-based optimization


Less cross-node data access, improving query performance
LSE instruction set-based efficient atomic operations
Node1
Core
Core
......
Cache
DIMM
DIMM
......
Node2
Core
Core
......
Cache
DIMM
DIMM
......
Data structure design based on a wider L3 cacheline, improving cache access hit ratio
High-efficiency batch log write, maximizing multi-core computing capabilities
Disk

1.9.3 High Scalability

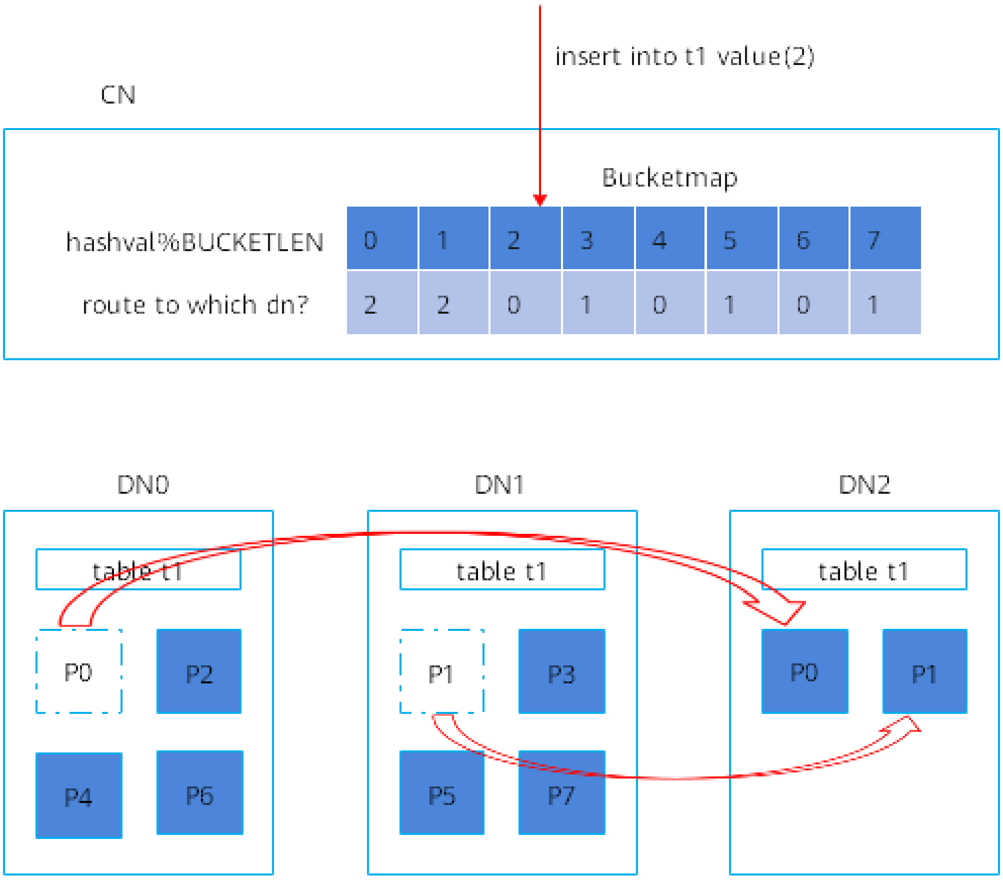
Online Scale-Out

In OLTP scenarios, database capacity expansion is required as the data volume increases. GaussDB provides the online scale-out capability to ensure that customer services are not interrupted and undergo smooth transition during scale-out and redistribution.

The principle and essence of online scale-out is to migrate data of multiple local tables in a cluster between different node groups. Each table that is redistributed after capacity expansion may undergo three steps: #1 baseline data redistribution after capacity expansion, #2 incremental data redistribution, and #3 table switchover. The second step is especially critical. In this step, the row-level incremental labeling technology is used to compensate for concurrent online insertion, deletion, and modification of customer services during baseline data redistribution in the first step.

On the basis of the preceding steps, the hash clustered storage is introduced to reduce the data scan and migration volume and the data reconstruction scale during data redistribution. In this way, the overall time of the redistribution process is shortened and the time window of impact on online services is reduced, implementing quick capacity expansion. The following figure shows the design roadmap.

Figure 1-29 Quick scale-out



hashval%BUCKETLEN	0	1	2	3	4	5	6	7
route to which dn?	2	2	0	1	0	1	0	1

Thread Pool High Concurrency

In the OLTP field, a database needs to process a large quantity of client connections. Therefore, the processing capability in high-concurrency scenarios is one of the important capabilities of the database.

The simplest processing mode for external connections is the per-thread-per-connection mode, in which a user connection generates a thread. This mode features simple processing thanks to its architecture. However, in high-concurrency scenarios, there are too many threads, causing heavy workload in thread switchover and large conflict between the lightweight lock areas of the database.

As a result, the performance (throughput) deteriorates sharply and the SLA of user performance cannot be met.

Therefore, a thread resource pooling and reuse technology needs to be used to resolve this problem. The overall design idea of the thread pool technology is to pool thread resources and reuse them among different connections. After the system is started, a fixed number of working threads are started based on the current number of cores or user configuration. A working thread serves one or more connection sessions. In this way, the session and thread are decoupled. The number of worker threads is fixed. Therefore, frequent thread switchover does not occur in case of high concurrency. The database layer schedules and manages sessions.

Distributed Data Storage

As a distributed database, GaussDB provides two-layer distributed storage capabilities.

- Distributed storage of data between shards: Hash distribution and replication distribution are supported. Hash distribution is mainly applied to user tables with a large amount of data. Hash calculation is performed on one or more distribution keys specified by a user, and data in a same user table is distributed and stored in different shards, thereby increasing the total data volume that can be supported by an entire database. In addition, the distributed parallel processing capability and pruning processing capability are provided. Replication distribution is mainly used for user tables with a small amount of data. Full data of the replication distribution table is stored in each shard, improving the performance of distributed multi-table associated query.
- Distributed storage of data in a shard: For each shard, quorum-based distributed multi-copy storage is supported to ensure high reliability and availability of the database. In addition, functions such as automatic primary/standby switchover, AZ switchover, and forcible promotion to the primary are provided, ensure that the RPO and RTO are stable and meet expectations.

Distributed Transaction Management

As a distributed database, GaussDB uses a central node solution. A unique GTM is deployed in a cluster, and the GTM assigns globally unique change sequence numbers (CSNs) to transactions as logical timestamps to ensure strong data read/write consistency. The processing capability of the central node is one of the important capabilities of the distributed database.

GaussDB provides the following features to optimize the GTM performance to the maximum extent:

- GTM thread pool. The thread resource pooling and reuse technology is used to decouple connections and sessions for a large number of concurrent connections, maximizing the GTM's capability of processing connection requests and improving performance and scalability in high-concurrency scenarios. In a 16-core system, the maximum concurrent processing capability of the GTM exceeds 20,000.
- GTM-lite. In the lightweight GTM, the GTM maintains only global CSNs. Distributed transactions are connected to the GTM only when they are committed, reducing the GTM communication and latency as well as the

requests processed by the GTM, optimizing the CPU usage of the GTM, and improving the performance and scalability in TP scenarios.

1.9.4 High Availability

Primary/Standby Deployment

To ensure that a fault can be rectified, data needs to be written into multiple copies. Multiple copies are configured for the primary and standby nodes, and logs are used for data synchronization. In this way, GaussDB has no data lost when a node is faulty or the system restarts after a stop. In the one primary and multiple standbys mode, all standby nodes need to redo logs and can be promoted to primary. The primary/standby deployment provides high disaster recovery and is more suitable for online transactional processing (OLTP) system that processes a large number of transactions.

If a primary instance is faulty, a primary/standby switchover or failover is performed to promote a standby node to the primary.

To ensure that the failover time is controllable, enable the log flow control function to control the rate at which logs are sent to the standby node. This ensures that the logs accumulated on the standby node will be replayed within the target time configured for flow control. After log flow control is enabled, the rate of sending logs to the standby node is dynamically adjusted. As a result, the overall transaction performance deteriorates.

In initial installation or backup and restoration, the data of standby instances needs to be rebuilt based on the primary instance. In this case, the build function is required to send the data and WAL logs of the primary instance to the standby instance. When the faulty primary instance is recovered and joins the cluster as a standby instance, the build function synchronizes its data and WALs with those of the new primary instance. In addition, in online capacity expansion, the build function is used to synchronize the metadata to the instance on the new node. Build includes full build and incremental build. Full build depends on node data for reconstruction. The amount of data to be copied is large, which takes a long time. Incremental build copies only differential files, which takes a short time. Generally, incremental build is preferred for fault recovery. If incremental build fails, full build is performed until the fault is rectified.

To implement HA DR for all instances, in addition to the preceding primary/standby multi-copy replication configured for DNs, GaussDB also provides other primary/standby DR capabilities, such as CN (primary/standby), GTM (one primary and multiple standbys), CM server (one primary and multiple standbys), and ETCD (one primary and multiple standbys). In this way, instances can be recovered as soon as possible without interrupting services, minimizing the impact of hardware, software, and human errors on services and ensuring service continuity.

Cross-AZ HA

AZ-level HA refers to the intra-city cross-AZ HA capability. Users usually set multiple AZs and distribute data across AZs. When an AZ is faulty due to power failure or network disconnection, other AZs can still provide services. Currently, GaussDB supports intra-city two-AZ and intra-city three-AZ deployment.

Cross-Region DR

DR refers to the DR capability of different regions. Users usually set multiple regions and distribute data across regions. When a region is faulty due to power failure or network disconnection, other regions can still provide services. Currently, GaussDB supports intra-city 1AZ + remote 1AZ DR, intra-city 2-AZ + remote 1-AZ DR, and intra-city 3-AZ + remote 1-AZ DR.

Logical Replication

GaussDB provides the logical decoding function to reversely parse physical logs into logical logs. Replication middleware such as DRS is used to convert logical logs into SQL statements and replay the SQL statements in the peer database to ensure data synchronization between clusters.

Reliable Transaction Processing

GaussDB provides high reliability for distributed transactions, supports strong consistency of distributed transactions, and meets the ACID attribute of transactions. In any scenario, reading only half of the data of distributed transactions does not occur. In synchronous commit scenarios, data is not lost and no error occurs. For failed two-phase commit (2PC) transactions, the database kernel provides the automatic gs_clean service for compensation and cleanup to ensure data reliability and availability. In any fault scenario, you do not need to manually clean data, and the database can correctly process data commit and rollback. In addition, services are not blocked.

Automatic Removal of CNs

Multiple CNs are deployed in the cluster to provide services for external systems and improve concurrent processing. CNs are peers to one another. When a DML statement is executed, the same result can be obtained by connecting to any CN. However, DDL statements must be executed on all CNs to ensure that database object definitions are consistent. If one or more CNs are faulty, the DDL statement cannot be executed in the entire cluster. To prevent CN faults from affecting DDL statement execution, the faulty CNs can be automatically removed from the cluster within a short period of time. In this way, DDL statements can be executed properly. After a CN is removed, you can replace the faulty node online and add it to the cluster again.

Online Node Replacement

If a node in a cluster is unavailable or the instance status is abnormal due to a hardware fault, and the cluster is not locked, you can replace the faulty node or rectify the faulty instance. During the process, DML operations are supported. DDL operations are supported in limited scenarios only.

Currently, enterprises need to add more nodes to process larger amounts of data and the probability of hardware damage increases accordingly. Replacing compute nodes has become a routine O&M work. The traditional offline node replacement cannot meet customers' requirements for service continuity. During routine O&M, frequent service interruption will bring great loss to customers. However, for most of databases in the industry, compute node replacement will interrupt services or affect some operations.

GaussDB supports online node replacement. During the node replacement window, DML operations are supported, and DDL operations are also supported in certain scenarios.

Users can use the gs_replace tool to replace the faulty node or instance. There are two scenarios for replacing a node:

- If the IP address of the new host is the same as that of the faulty host, change the name and IP address of the new host to be the same as those of the faulty host, and then replace the faulty host. This solution is applicable when the name and IP address of the host are the same as those of the host to be replaced. The solution requires little preinstallation preparation and quickly replaces the host.
- The IP address changes after the host replacement. In this case, the name and IP address of the new host can be replaced. This solution is applicable when the name and IP address of the new host are different from those of the original host. It is also applicable to the scenario where a high-performance host is used to replace a host with poor performance but the instances on the host are normal. The instance repair operation will repair all faulty instances on the faulty node. The instance repair may fail due to errors in some steps. If the instance repair fails, you need to run the same command to re-enter the repair process. After the instance repair is complete, the instance will be successfully started.

Logical Backup

GaussDB provides the logical backup capability to back up data in user tables in TEXT or CSV format and restore the data in homogeneous or heterogeneous databases. With the distributed parallel technology, logical backup extracts user table records to be backed up from each DN in streaming mode, providing high backup and restoration performance.

Physical Backup

GaussDB provides the physical backup capability to back up the data of the entire instance to OBS in the internal format of the database and restore the data of the entire cluster in the homogeneous database. Physical backup uses the distributed parallel technology to physically back up data files of each DN, providing high backup and restoration performance. Advanced functions such as compression, flow control, and resumable backup are provided.

Physical backup includes full backup and incremental backup. The difference is as follows: A full backup involves all data of a database at the backup time point. The time required for a full backup is long (in direct proportion to the total data volume of the database). You can use a full backup to restore data of a complete database. An incremental backup covers all files that have been changed since the last backup was made. It takes a short period of time (in direct proportion to the incremental data volume and irrelevant to the total data volume). However, incremental backups cannot restore all data of a database.

Typical physical backup policies and application scenarios are as follows:

- On Monday, perform a full backup of the database.
- On Tuesday, perform an incremental backup based on the full backup on Monday.

- On Wednesday, perform an incremental backup based on the incremental backup on Tuesday.
- ...
- On Sunday, perform an incremental backup based on the incremental backup on Saturday.

The preceding backup period is one week.

Automatic Retry upon a Job Failure

In common fault scenarios, such as network exception and deadlock, failed queries are retried to improve database usability. GaussDB provides two job retry mechanisms: gsql retry and CN retry. CN retry is more comprehensive. You are advised to use CN retry.

- The gsql retry mechanism uses a unique error code (SQL STATE) to identify an error that requires a retry. The function of the client tool gsql is enhanced. The error code configuration file **retry_errcodes.conf** is used to configure the list of errors that require a retry. The file is stored in the installation directory at the same level as gsql. gsql provides the **\set RETRY** [*number*] command to enable or disable the retry function. The number of retry times ranges from 5 to 10, and the default value is 5. When this function is enabled, **gsql** reads the preceding configuration file. The error retry controller records the error code list through the container. If an error in the configuration occurs, the controller resends cached queries to the servers to retry, until they are successfully executed or an error message indicating that the number of retry attempts has reached the maximum is displayed.
- CN retry: If an SQL statement from the JDBC or ODBC driver fails to be executed, the CN can automatically identify the error reported during statement execution and re-deliver the task for automatic retry. CN retry uses the GUC parameter **retry_ecode_list** to configure the error list to be retried. You are not advised to modify this parameter. Users can use the **max_query_retry_times** parameter to specify the retry times. If this parameter is set to **0**, CN retry is disabled. CN retry mainly supports communication and resource insufficiency faults. For details about the error types, see **Table 1-12**.

Table 1-12 Error types supported by CN retry

Error Type	Error Code	Remarks
CONNECTION_RESET_BY_PEER	YY001	TCP communication errors. Print information: **Connection reset by peer**. (reset between CN and DN)
STREAM_CONNECTION_RESET_BY_PEER	YY002	TCP communication error. Print information: "Stream connection reset by peer" (communication between DNs)

Error Type	Error Code	Remarks
LOCK_WAIT_TIMEOUT	YY003	Lock wait timeout. Print information: "Lock wait timeout"
CONNECTION_TIMED_OUT	YY004	TCP communication error. Print information: "Connection timed out"
SET_QUERY_ERROR	YY005	Failed to deliver the **SET** command. Print information: "Set query"
OUT_OF_LOGICAL_MEMORY	YY006	Failed to apply for memory. Print information: "Out of logical memory"
SCTP_MEMORY_ALLOC	YY007	SCTP communication error. Print information: "Memory allocate error"
SCTP_NO_DATA_IN_BUFFER	YY008	SCTP communication error. Print information: "SCTP no data in buffer"
SCTP_RELEASE_MEMORY_CLOSE	YY009	SCTP communication error. Print information: "Release memory close"
SCTP_TCP_DISCONNECT	YY010	SCTP communication error. Print information: "TCP disconnect"
SCTP_DISCONNECT	YY011	SCTP communication error. Print information: "SCTP disconnect"
SCTP_REMOTE_CLOSE	YY012	SCTP communication error. Print information: "Stream closed by remote"
SCTP_WAIT_POLL_UNKNOW	YY013	Waiting for an unknown poll. Print information: "SCTP wait poll unknow"
SNAPSHOT_INVALID	YY014	Invalid snapshot. Print information: "Snapshot invalid"
ERRCODE_CONNECTION_RECEIVE_WRONG	YY015	Failed to receive a connection. Print information: "Connection receive wrong"
OUT_OF_MEMORY	53200	Out of memory. Print information: "Out of memory"

Error Type	Error Code	Remarks
CONNECTION_FAILURE	08006	GTM error. Printed information: "Connection failure"
CONNECTION_EXCEPTION	08000	Failed to communicate with DNs due to connection errors. Print information: "Connection exception"
ADMIN_SHUTDOWN	57P01	System shutdown by the administrator. Print information: "Admin shutdown"
STREAM_REMOTE_CLOSE_SOCKET	XX003	Remote socket disabled. Print information: "Stream remote close socket"
ERRCODE_STREAM_DUPLICATE_QUERY_ID	XX009	Duplicate query. Print information: "Duplicate query id"
ERRCODE_STREAM_CONCURRENT_UPDATE	YY016	Concurrent stream query and update. Print information: "Stream concurrent update"

Load Balancing

Load balancing provides a unified portal for multiple CNs, evenly distributing client requests to each server where CNs reside.

As an important component of the cluster system, the load balancing has the following functions:

- Balance loads among CNs and make full use of the computing capabilities of multiple CNs.
- Isolate faults. When a CN is faulty, the load balancing function helps detect the fault and automatically stops forwarding requests to the faulty CN. The fault detection time depends on the load balancing mode. When the JDBC client is used for load balancing in intranet mode, the CN fault detection time is about 30 seconds. When the ELB is used for load balancing on the extranet mode, the fault detection time is 3 seconds (configurable) by default.

GaussDB supports two load balancing modes:

- JDBC driver

  Users only need to specify some CNs in the connection string. The JDBC driver obtains an available CN list (for HA) and performs load balancing among the available CNs. If a CN is faulty, it is removed from the list. The default interval for refreshing the available CN list is 10 seconds (configurable). Users are advised to use this mode when using the intranet to provide services. **Figure 1-30** shows the load balancing principle of the JDBC driver.

- ELB and other third-party load balancers

  GaussDB supports interconnection with third-party load balancing solutions, such as ELB. Users are advised to use this mode when using external networks to provide services. In this case, the load balancing function of the JDBC driver can be disabled. **Figure 1-31** shows the load balancing principle of third-party load balancers.

Figure 1-30 Load balancing principle of the JDBC driver

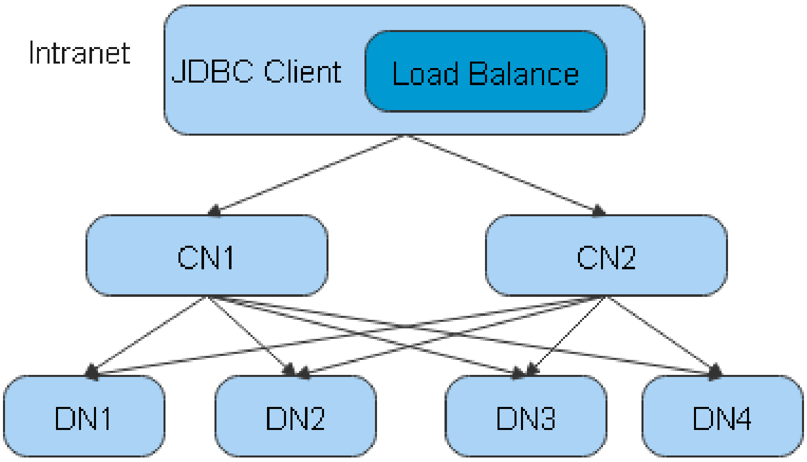
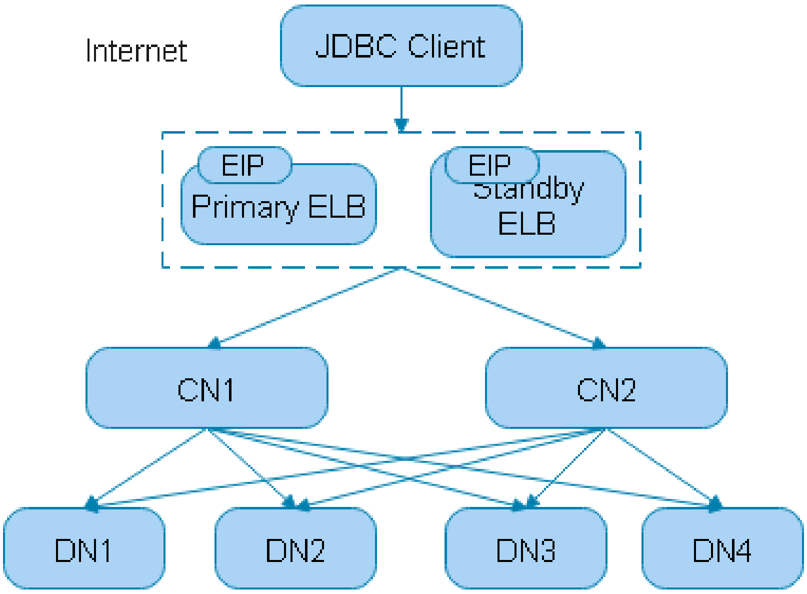


Figure 1-31 Third-party load balancing



1.9.5 Maintainability

Hot Patch Upgrade

A hot patch can be loaded without interrupting services and can be used to resolve some emergent database kernel problems online without affecting services. A hot patch generates a patch file based on a released code version, and then inserts the patch file into a running address space of a database kernel in a form of a module. By searching for the address of the hot patch target function and replacing the entry address dynamically and atomically, it redirects the function code segment to the patch file code segment to fix the online system defects.

- Hot patches are made by fixing specific defect functions, making modules, and dynamically loading the modules to the running kernel system.
- The hot patch finds the target function, and adds a jump instruction at the entrance of the target function. When the target function is invoked, the hot patch jumps to the patch area to execute the patch function.
- The replacement and restoration of the target function are atomic operations on the CPU register. Hot patches can be loaded and uninstalled anytime and anywhere. The online system can run the latest code at any time without interruption.

Benefits:

The biggest advantage of a hot patch is "zero" service interruption. A hot patch can solve some emergent database kernel problems without affecting services.

The benefits are as follows:

- The version release time is shortened, and the regression test for emergency problems is lightweight, changing from version regression test to patch regression test, which improves the response speed of online emergency problems.
- Hot patch installation and uninstallation are transparent to services, improving customer satisfaction.

Gray upgrade

Gray upgrade is an online upgrade mode that preferentially upgrades certain nodes.

The gray upgrade can be classified into:

1. Major version upgrade involving system table changes: During the upgrade, hard code the system table structures and system functions of different versions in binary mode to ensure that the binary files of the new and old versions can parse and use the system table tuples of the new and old versions.
2. Major version upgrade and binary upgrade involving replacement of old and new binary files: During the upgrade, replace the binary files on the specified node in gray mode first. After the system runs for a period of time, replace the binary files on the remaining nodes. If all gray upgrade nodes break down, the remaining nodes (old binary nodes) can still provide database services normally. That is, the remaining nodes must contain at least one copy (or multiple copies) of each shard.
3. Major version upgrade and binary upgrade involving replacement of old and new binary files as well as OS and hardware upgrade: If the upgrade cannot be performed in advance, switch all the primary instances on the gray upgrade nodes to non-gray upgrade nodes before the upgrade. If the upgrade

involves only the replacement of database binary files, to minimize the impact on services, two sets of binary files coexist on the same node. The process version is switched and upgraded through soft connection switchover (intermittent disconnection once within 10 seconds).

Benefits:

- Both major and minor database version upgrades are supported. (If the kernel version number remains unchanged, a minor version upgrade is used. Otherwise, a major version upgrade is used.)
- Online upgrade is supported. Some nodes can be upgraded first. The upgraded nodes are intermittently disconnected once within 10 seconds.

Rolling Upgrade

Rolling upgrade supports all service operations. The cluster is upgraded by rolling upgrade. For the rolling upgrade by AZ, users can upgrade certain AZs first and check whether the upgraded AZs are running properly. Then, upgrade the remaining AZs. Users can also upgrade all AZs at a time.

The rolling upgrade includes the following three aspects:

1. Major version upgrade involving system table changes: During the upgrade, hard code the system table structures and system functions of different versions in binary mode to ensure that the binary files of the new and old versions can parse and use the system table tuples of the new and old versions.
2. Major version upgrade and binary upgrade involving replacement of old and new binary files: During the upgrade, replace the binary files on the specified shard first. After the system runs for a period of time, replace the binary files on the remaining shards.
3. If only the binary files of the database need to be replaced during the upgrade, to minimize the impact on services, the two sets of binary files exist on the same node at the same time, and the soft connection switchover mode is used to switch the process version.

Benefits:

- Both major and minor database version upgrades are supported. (If the kernel version number remains unchanged, a minor version upgrade is used. Otherwise, a major version upgrade is used.)
- Supports online upgrade of data fragments. The standby DN is upgraded before the primary DN. Upgrading the standby DN does not interrupt services, but upgrading the primary DN interrupts services for less than 10s.

In-place Upgrade

In-place upgrade is an offline upgrade mode. Services are stopped during the in-place upgrade, and all nodes in the cluster are upgraded at a time.

Benefits:

- Both major and minor database version upgrades are supported. (If the kernel version number remains unchanged, a minor version upgrade is used. Otherwise, a major version upgrade is used.)

- It is a relatively stable and reliable upgrade mode.

Workload Diagnosis Report

The workload diagnosis report (WDR) generates a performance report between two different time points based on the system performance snapshot data at two different time points. The report is used to diagnose database kernel performance faults.

WDR depends on the following two components:

- SNAPSHOT: The performance snapshot can be configured to collect a certain amount of performance data from the kernel at a specified interval and store the data in the user tablespace. Any snapshot can be used as a performance baseline for comparison with other snapshots.
- WDR Reporter: This tool analyzes the overall system performance based on two snapshots, calculates the changes of more specific performance indicators between the two time periods, and generates summarized and detailed performance data. For details, see **Table 1-13** and **Table 1-14**.

Table 1-13 Summarized diagnosis report

Diagnosis Type	Description
Database Stat	Evaluates the load and I/O status of the current database. Load and I/O are the most important characteristics of a TP system. The statistics include the number of sessions connected to the database, number of committed and rolled back transactions, number of read disk blocks, number of disk blocks found in the cache, number of rows returned, captured, inserted, updated, and deleted through database query, number of conflicts and deadlocks, usage of temporary files, and I/O read/write time.
Load Profile	Evaluates the current system load from the time, I/O, transaction, and SQL dimensions. The statistics include the job running elapse time, CPU time, daily transaction volume, logical and physical read volume, read and write I/O times and size, login and logout times, SQL, transaction execution volume, and SQL P85 and P90 response time.
Instance Efficiency Percentages	Evaluates the cache efficiency of the current system. The statistics include the database cache hit ratio.
Events	Evaluates the performance of key system kernel resources and key events. Includes the number of times that the key events of the database kernel occur and the waiting time.

Diagnosis Type	Description
Wait Classes	Evaluates the performance of key events in the system. The statistics include the release of the data kernel in the main types of waiting events, such as STATUS, LWLOCK_EVENT, LOCK_EVENT, and IO_EVENT.
CPU	Includes time release of the CPU in user mode, kernel mode, wait I/O, or idle mode.
IO Profile	Includes the number of database I/O times, database I/O data volume, number of redo I/O times, and redo I/O volume.
Memory Statistics	Includes maximum process memory, used process memory, maximum shared memory, and used shared memory.

Table 1-14 Detailed diagnosis report

Diagnosis Type	Description
Time Model	Evaluates the performance of the current system in the time dimension. The statistics include time consumed by the system in each phase, including the kernel time, CPU time, execution time, parsing time, compilation time, query rewriting time, plan generation time, network time, and I/O time.
SQL Statistics	Diagnoses SQL statement performance problems. The statistics include normalized SQL performance indicators in multiple dimensions: elapsed time, CPU time, rows returned, tuples reads, executions, physical reads, and logical reads. The indicators can be classified into execution time, number of execution times, row activity, and cache I/O.
Wait Events	Diagnoses performance of key system resources and key time in detail. The statistics include the performance of all key events in a period of time, including the number of events and the time consumed.
Cache IO Stats	Diagnoses the performance of user tables and indexes. The statistics include read and write operations on all user tables and indexes, and the cache hit ratio.
Utility status	Diagnoses the performance of backend jobs. The statistics include the performance of backend operations such as page operation and replication.

Diagnosis Type	Description
Object stats	Diagnoses the performance of database objects. The statistics include user tables, tables on indexes, index scan activities, insert, update, and delete activities, number of valid rows, and table maintenance status.
Configuration settings	Determines whether the configuration is changed. It is a snapshot that contains all current configuration parameters.

Benefits:

- WDR is the main method for diagnosing long-term performance problems. Based on the performance baseline of a snapshot, performance analysis is performed from multiple dimensions, helping DBAs understand the system load, performance of each component, and performance bottlenecks.
- Snapshots are also an important data source for subsequent performance problem self-diagnosis and self-optimization suggestions.

Slow SQL Diagnosis

Slow SQL diagnosis can be classified into real-time slow SQL and historical slow SQL.

- Real-time slow SQL can output information about jobs that are being executed in the current system and whose execution time exceeds the threshold based on the execution time threshold provided by users.
- Historical slow SQL diagnosis records information about all jobs whose execution time exceeds the threshold.

Slow SQL provides table-based and file-based query interfaces. You can query the execution plan, start time, end time, query statement, row activity, kernel time, CPU time, execution time, parsing time, compilation time, query rewriting time, plan generation time, network time, and I/O time. All information is anonymized.

Benefits:

- Real-time slow SQL provides an interface for users to manage unfinished jobs. Users can manually stop abnormal jobs that consume too many resources.
- Historical slow SQL provides detailed information required for slow SQL diagnosis. Users can diagnose performance problems of specific slow SQL statements offline without reproducing the problem. The table-based and file-based interfaces help users collect statistics on slow SQL indicators and connect to third-party platforms.

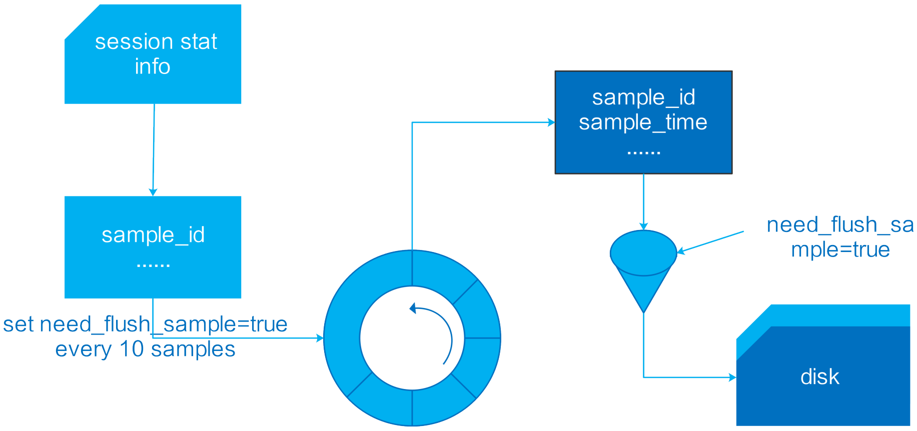
Session Diagnosis

The session diagnosis function diagnoses performance of all active sessions in the system. As real-time collection of indicators of all active sessions has a greater impact on user load, the session snapshot technology is used to sample indicators of active sessions, and collect statistics on active sessions from the sampling. The statistics reflect the basic information, status, and resources of active sessions from the dimensions of client information, execution start time, execution end time, SQL text, waiting events, and current database objects. The active session information collected based on the probability can help users diagnose which sessions consume more CPU and memory resources, which database objects are hot objects, and which SQL statements consume more key event resources in the system. In this way, users can locate faulty sessions, SQL statements, and database designs.

Session sampling data is classified into two levels, as shown in **Figure 1-32**.

1. The first level is real-time information stored in the memory. The active session information in the latest several minutes is displayed, which has the highest precision.
2. The second level is the persistent historical information stored in disk files. It displays the historical active session information in a long period of time and is sampled from the memory data. This level is suitable for long-run statistics and analysis.

Figure 1-32 Session performance diagnosis principle


session stat info
sample_id ......
set need_flush_sample=true every 10 samples
sample_id sample_time ......
need_flush_sample=true
disk

Benefits:

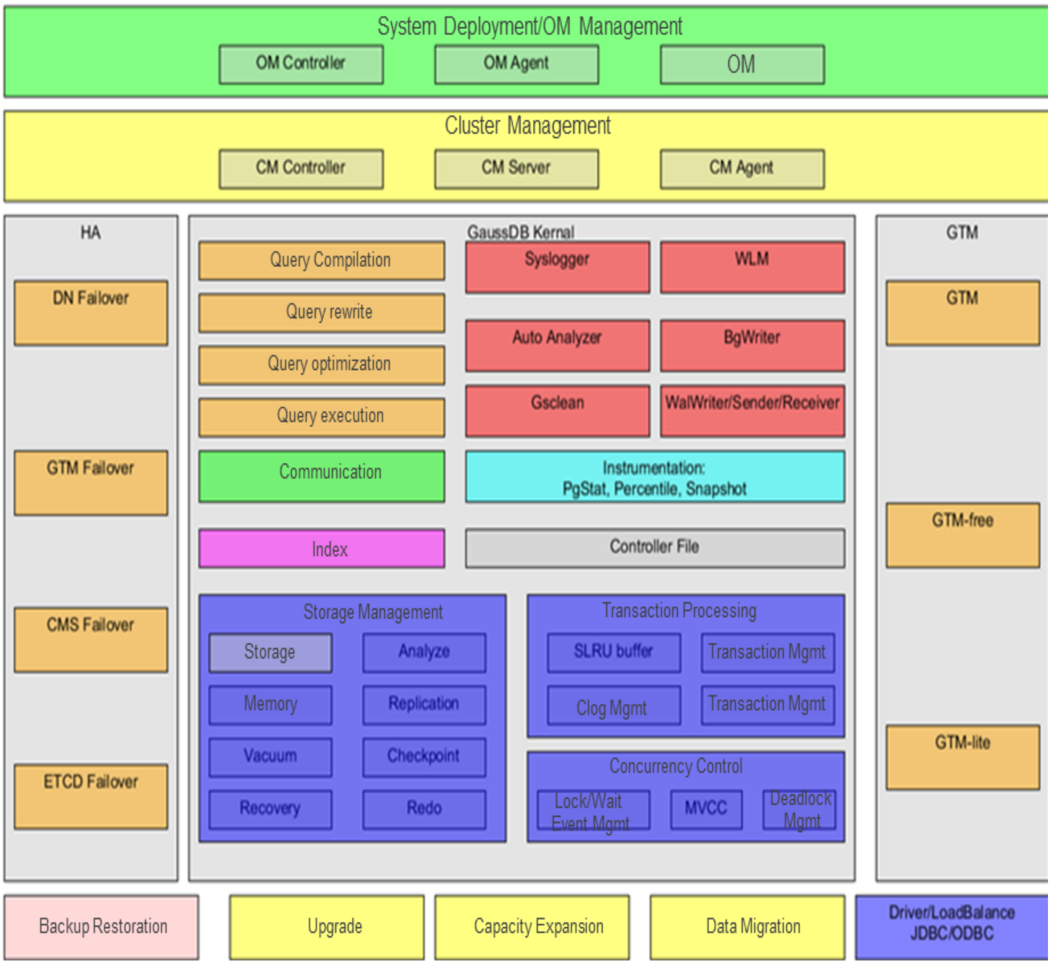
- Displays the latest events that consume the most resources of user sessions.
- Checks the waiting events that occupy the most resource-consuming SQL statements.
- Checks the waiting events that occupy the most resource-consuming sessions.
- Checks information about the most resource-consuming users.
- Checks the waiting relationship between blocked sessions.

System KPI-aided Diagnosis

GaussDB provides KPIs of 11 categories and 26 sub-categories, covering instances, files, objects, workload, communication, sessions, threads, cache I/O, locks, wait events, and clusters.

Figure 1-33 shows the distribution of kernel KPIs.

Figure 1-33 Distribution of kernel KPIs



Benefits:

- Summarized system load diagnosis
  - Precise alarms for system load exceptions (overload, stall, and service SLA) and precise system load profile
- Summarized system time model diagnosis
  - Instance-level and query-level time model segmentation, diagnosing the root causes of instance and query performance problems
- Query performance diagnosis
  - Cluster-level query summary, including top SQL, SQL CPU usage, I/O consumption, execution plan, and excessive hard parsing
- Diagnosis of disk I/O, index, and buffer performance problems
- Diagnosis of connection and thread pool problems
- Diagnosis of checkpoint and redo (RTO) performance problems
- Diagnosis of system I/O, LWLock, and wait performance problems
  - Diagnosis of over 60 modules and over 240 key operation performance problems
- Function-level performance monitoring and diagnosis (by GSTRACE)

- Tracing of over 50 functions at the storage and execution layers

One-Click Diagnosis Information Collection

Multiple suites are provided to capture, collect, and analyze diagnosis data, enabling fault diagnosis and accelerating the diagnosis process. Necessary database logs, cluster management logs, and stack information can be extracted from the production environment based on the requirements of development and fault locating personnel. Fault locating personnel demarcate and locate faults based on the obtained information.

The one-click collection tool obtains different information from the production environment depending on the actual faults, improving the fault locating and demarcation efficiency. Users can modify the configuration file to collect the required information:

- OS information by running Linux commands
- Database information by querying system catalogs or views
- Run logs of the database system and logs related to cluster management
- Database system configuration information
- Core files generated by database-related processes
- Stack information about database-related processes
- Trace information generated by the database process
- Redo log files (Xlogs) generated by the database
- Planned reproduction information

1.9.6 Database Security

Access Control

Access control is to manage users' database access control permissions, including database system permissions and object permissions.

Role-based access control is supported to associate roles and permissions. Permissions are assigned to roles and then roles are assigned to users, implementing user access control permission management. The login access control is implemented by using the user ID and authentication technology. The object access control is implemented by checking the object permission based on the user permission on the object. Users can assign the minimum permissions required for completing tasks to related database users to minimize database usage risks.

An access control model based on separation of permissions is supported. Database roles are classified into system administrators, security administrators, and audit administrators. Security administrators create and manage users, system administrators grant and revoke user permissions, and audit administrators audit all user behaviors.

By default, the role-based access control model is used. Users can set parameters to determine whether to enable the access control model based on separation of permissions.

Separation of Control and Access Permissions

For system administrators, the control and access permissions on table objects are separated to improve data security of common users and restrict the object access permissions of administrators.

This feature applies to the following scenarios: An enterprise has multiple business departments using different database users to perform service operations. Database maintenance departments at the same level use database administrators to perform O&M operations. The business departments require that administrators can only perform control operations (DROP, ALTER, and TRUNCATE) on data of each department and cannot perform access operations (INSERT, DELETE, UPDATE, SELECT, and COPY) without authorization. The control permissions of database administrators for tables need to be isolated from their access permissions to improve the data security of common users.

The system administrators can specify the **INDEPENDENT** attribute when creating a user, indicating that the user is a private user. Database administrators (including initial users and other administrators) can control (DROP, ALTER, and TRUNCATE) objects of private users but cannot access (INSERT, DELETE, UPDATE, SELECT, COPY, GRANT, REVOKE, and ALTER OWNER) the objects without authorization.

Database Encryption Authentication

GaussDB provides the RFC5802-based encryption and authentication mode.

The unidirectional, irreversible Hash encryption algorithm PBKDF2 is used for encryption and authentication, effectively defending against rainbow attacks.

The password of the created user is encrypted and stored in the system catalog. During the entire authentication process, passwords are encrypted for storage and transmission. The hash value is calculated and compared with the value stored on the server to verify the correctness.

The message processing flow in the unified encryption and authentication process effectively prevents attackers from cracking the username or password by capturing packets.

Data Encryption and Storage

Imported data is encrypted for storage.

This feature provides data encryption and decryption APIs for users and uses encryption functions to encrypt sensitive information columns identified by users, so that data can be stored in tables after being encrypted.

If a user needs to encrypt the entire table, the user can write an encryption function for each column. Different attribute columns can use different input parameters.

If a user with the required permission wants to view specific data, the user can decrypt required columns using the decryption function API.

Database Audit

Audit logs record user operations performed on database startup and stopping, connection, and DDL, DML, and DCL operations. The audit log mechanism enhances the database capability of tracing illegal operations and collecting evidence.

Users can set parameters to specify the statements or operations for which audit logs are recorded.

Audit logs record the event time, type, execution result, username, database, connection information, database object, database instance name, port number, and details. Users can query audit logs by start time and end time and filter audit logs by recorded field.

A database security administrator can use the audit logs to reproduce a series of events that cause faults in the database and identify unauthorized users, unauthorized operations, and the time when these operations are performed.

The unified audit mechanism is supported, which implements efficient security audit management by customizing audit policies. After an administrator defines the audit objects and audit behaviors, if the task executed by a user is associated with an audit policy, the corresponding audit behavior is generated and the audit log is recorded. Customized audit policies can cover common user management activities, as well as DDL and DML operations, meeting routine audit requirements.

Network Communication Security

SSL can be used to encrypt communication data between clients and servers.

The TLS 1.2 protocol and a highly secure encryption algorithm suite are adopted. **Table 1-15** lists the supported encryption algorithm suites.

Table 1-15 Encryption algorithm suites

OpenSSL Suite Name	IANA Suite Name	Security Level
DHE-RSA-AES128-GCM-SHA256	TLS_DHE_RSA_WITH_AES_128_GCM_SHA256	HIGH
DHE-RSA-AES256-GCM-SHA384	TLS_DHE_RSA_WITH_AES_256_GCM_SHA384	HIGH
DHE-RSA-AES128-CCM	TLS_DHE_RSA_WITH_AES_128_CCM	HIGH
DHE-RSA-AES256-CCM	TLS_DHE_RSA_WITH_AES_256_CCM	HIGH

Transparent Data Encryption

To prevent malicious users from reading data files without authentication, users can use the transparent data encryption function to encrypt data files in the database. This ensures that users can read decrypted data only after starting and connecting to the database properly.

Transparent data encryption refers to the encryption and decryption of user data in a cluster. The access program is unaware of the encryption and decryption and it can be directly applied to the encrypted database without modification or adaptation. After the transparent data encryption function is enabled, data is stored in the memory in plaintext. When data is written to disks, the data storage unit is encrypted. When data is read, the data storage unit is decrypted. The transparent encryption function supports row-store and column-store table encryption, and does not distinguish system tables from user data tables.

This feature needs to be configured during cluster installation to determine whether to enable this feature. Currently, only cluster-level transparent encryption is supported. That is, a working key is used for data encryption and decryption in the cluster. The encryption algorithms are AES128 and SM4, and the encryption mode is CTR.

This feature depends on the key management service provided by the external KMS.

Row-Level Security Policy

The row-level security (RLS) feature enables database access control to be accurate to each row of data tables. When different users perform the same SQL query operation, the read results may be different according to the RLS policy.

Users can create an RLS policy for a data table. The policy defines an expression that takes effect only for specific database users and SQL operations. When a database user accesses the data table, if a SQL statement meets the specified RLS policy of the data table, the expressions that meet the specified condition will be combined by using **AND** or **OR** based on the attribute type (**PERMISSIVE** | **RESTRICTIVE**) and applied to the execution plan in the query optimization phase.

RLS is used to control the visibility of row-level data in tables. By predefining filters for data tables, the expressions that meet the specified condition can be applied to execution plans in the query optimization phase, which will affect the final execution result. Currently, RLS supports the following SQL statements: SELECT, UPDATE, and DELETE.

Resource Labels

Resource labels classify database resources based on user-defined rules to implement resource classification and management. Administrators can configure resource labels to configure security policies, such as auditing or data masking, for a group of database resources.

Resource labels can be used to group database resources based on features and application scenarios. Users can manage all database resources with specified labels, greatly reducing policy configuration complexity and information redundancy and improving management efficiency.

Currently, resource labels support the following database resource types: schema, table, column, view, and function.

Dynamic Data Masking

To prevent unauthorized users from sniffing privacy data, the dynamic data masking feature can be used to protect user privacy data. When an unauthorized user accesses the data for which a dynamic data masking policy is configured, the database returns the anonymized data to protect privacy data.

Administrators can create dynamic data masking policies on data columns. The policies specify the data masking methods for specific user scenarios. After the dynamic data masking function is enabled, the system matches user identity information (such as the access IP address, client tool, and username) with the masking policy when a user accesses data in the sensitive column. After the matching is successful, the system masks the sensitive data in the query result of the column based on the masking policy.

The purpose of dynamic data masking is to flexibly protect privacy data by configuring the filter, and specifying sensitive column labels and corresponding masking functions in the masking policy without changing the source data.

Unified Auditing

Unified auditing allows administrators to configure audit policies for database resources or resource labels to simplify management, generate audit logs, reduce redundant audit logs, and improve management efficiency.

Administrators can customize audit policies for configuring operation behaviors or database resources. The policies are used to audit specific user scenarios, user behaviors, or database resources. After the unified auditing function is enabled, when a user accesses the database, the system matches the corresponding unified audit policy based on the user identity information, such as the access IP address, client tool, and username. Then, the system classifies the user behaviors based on the access resource label and user operation type (DML or DDL) in the policy to perform unified auditing.

The purpose of unified auditing is to change the existing traditional audit behavior into specific tracking audit behavior and exclude other behaviors from the audit, thereby simplifying management and improving the security of audit data generated by the database.

Password Strength Verification

To harden the security of customer accounts and data, do not set weak passwords. Users need to specify a password when initializing the database, creating a user, or modifying a user. The password must meet the strength requirements. Otherwise, the system prompts users to enter the password again.

The account password complexity policy restricts the minimum number of uppercase letters, lowercase letters, digits, and special characters in a password, the maximum and minimum length of a password, the password cannot be the same as the username or the reverse of the username, and the password cannot be a weak password. This policy enhances user account security.

Weak passwords are easy to crack. The definition of weak passwords may vary with users or user groups. Users can define their own weak passwords.

The **password_policy** parameter specifies whether to enable the password strength verification mechanism. The default value is **1**, indicating that the password strength verification mechanism is enabled.

1.10 Technical Specifications

This section describes the technical specifications of GaussDB, as shown in the following table.

Table 1-16 Technical specifications

Technical Specifications	Maximum Value
Number of DN shards	256
Size of a single table	32 TB x Number of nodes
Size of data in a single row	1,600 x 1 GB
Size of a single field in each record	1 GB
Number of records in a single table	2^{32} x (8 KB/Row width). At the code level, a single table can contain a maximum of 2^{32} pages, and the size of each page is 8 KB. Assume that the current data row width is 1 KB. The number of records in a single table is 2^{32} x 8 = 2^{35}. The current page size is 8 KB, and each page contains eight rows of data.
Maximum number of columns in a table	1,600
Maximum number of indexes in a table	2^{32}
Maximum number of columns in a single table index	32
Number of constraints in each table	2^{32}
Object name length	63 bytes
Number of concurrent connections	100,000
Intra-AZ RTO	< 10s
Cross-AZ RTO	< 60s
Cross-region RTO	< 10 min (stream DR: The write speed of xlogs in a single shard cannot be greater than 10 MB/s.)
Intra-AZ RPO	0
Cross-AZ RPO	0

Technical Specifications	Maximum Value
Cross-region RPO	< 10s (Stream DR:The write speed of xlogs in a single shard cannot be greater than 10 MB/s.)
PITR	Logs in a single shard are backed up to OBS at a speed of 40 MB/s, and the RPO is less than 5 min.

Note:

- Note: In the manual startup scenario, RTO indicates the software execution time.
- Cross-region DR (OBS solution) requires that the traffic of a single shard does not exceed 4 M/s (about 1,000 TPS). You can determine whether to use this solution based on your workloads.

1.11 Constraints

To ensure the stability and security of GaussDB, certain constraints are put in place for access or permissions control. **Table 1-17** describes such constraints.

After GaussDB is installed or upgraded, you need to load license. Otherwise, new resources may fail to be provisioned or added.

You need to search for business model in the LLD template of base installation project. If the value is **BusinessModelOne/BusinessModelTwo**, you need to apply for a cloud service license.

You need to apply for a product license for **BusinessModelThree**.

NOTICE

- If the business model cannot be found in the template, contact the frontline delivery manager to confirm the business model in the customer contract.
- If the current site is used for testing, the frontline manager can apply for a temporary license or use the default resources, but the new license is required for commercial use.
- If no license resource certificates are imported into the environment, you can use resources (288 vCPUs) for 60 days by default. When the service resource usage exceeds the total resources authorized by the license or the license is expired, new resources cannot be added.
- If a license resource certificate is imported into the environment, new resources are controlled based on the time when the license was imported and the total number of resources authorized by the license.
- For details about cloud service license control items, see "Other Information" > "Cloud Service License Control Items" in the *Huawei Cloud Stack License Guide*.

Table 1-17 Function constraints

Function	Constraints
Database access	● Security group rules must be added to allow the ECSs to access the GaussDB instances. By default, a GaussDB instance cannot be accessed by an ECS in a different security group. To allow it, you must add an inbound rule to the GaussDB security group. ● The default port is **8000**. You can only change it when creating a DB instance.
Deployment	ECSs where DB instances are deployed are not directly visible to you. You can only access the DB instances through IP addresses and database ports.
Database root permissions	Only the **root** user permissions are available on the instance creation page.
DB instance reboot	GaussDB DB instances cannot be rebooted through commands. They must be rebooted on the management console.
Backup files	GaussDB backup files are stored in OBS buckets and are not visible to you.

1.12 Related Services

Table 1-18 shows the relationship between GaussDB and other services.

Table 1-18 Related services

Service	Description
Elastic Cloud Server (ECS)	Enables you to access DB instances through an ECS to reduce application response time.
Virtual Private Cloud (VPC)	Isolates your network and controls access to your DB instances.
Object Storage Service (OBS)	Stores automated and manual backups of your DB instances.
Data Admin Service (DAS)	Provides a visualized GUI interface for you to connect and manage cloud databases.