Document Database Service

Service Overview

Issue	01
Date	2022-09-30

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


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

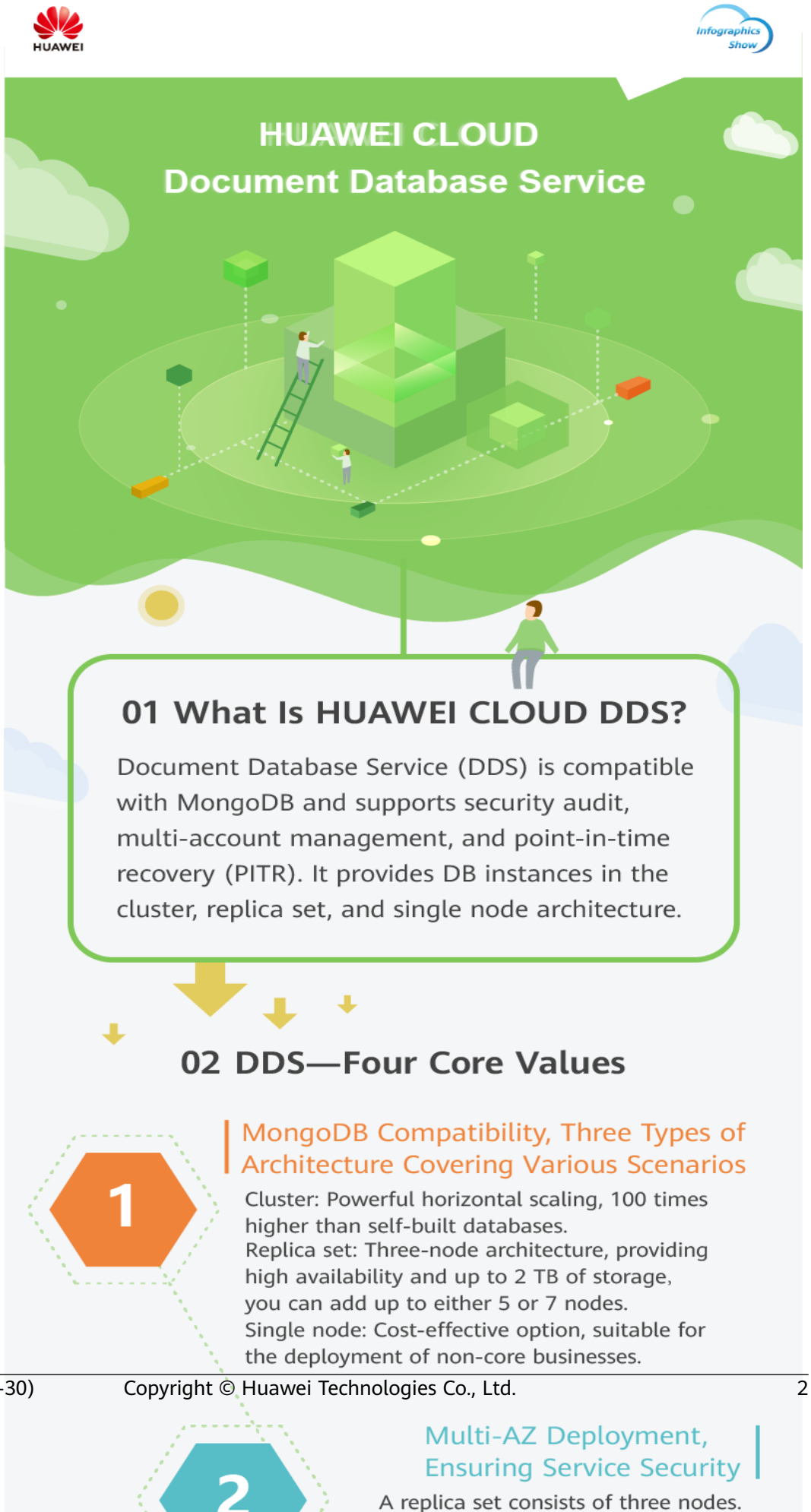
Notice

The purchased products, services and features are stipulated by the contract made between Huawei 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 Infographics



2 What Is DDS?

Document Database Service (DDS), compatible with MongoDB, is a secure, high availability (HA) database service that is reliable, scalable, and easy to use. It provides functions such as one-click deployment, elastic capacity expansion, disaster recovery, backup, restoration, monitoring, and alarm reporting.

Before using DDS, you need to be familiar with MongoDB and the MongoDB protocol. For details, see **official documents**.

Storage Structure

Figure 2-1 Storage structure

The basic management unit of DDS is an instance. A DDS instance consists of database, collection, and document.

DDS is fully compatible with the MongoDB protocol, so some terms, data structures, and basic syntax of DDS are the same as those of MongoDB. The following table lists the difference between common terms in MongoDB and relational databases to help you better understand and use DDS.

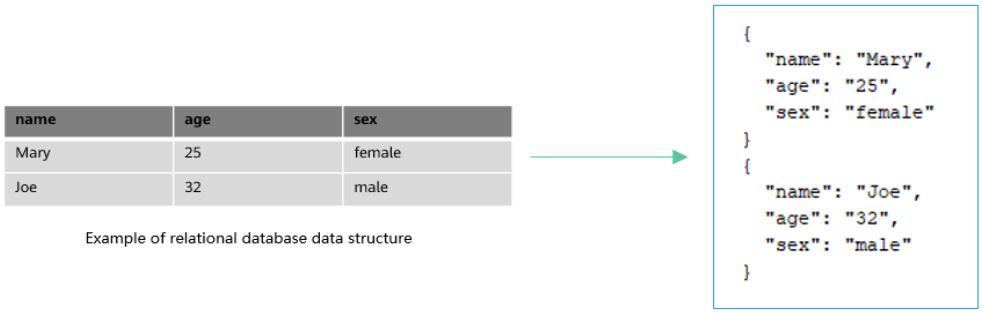
Table 2-1 Terms

MongoDB	Description	Relational Database
Database	A DDS instance contains multiple databases, and a database contains multiple collections.	Database
Collection	A collection is a group of MongoDB documents.	Table
Document	A document is a group of key-value pairs (BSONs) and is the most basic unit in MongoDB.	Row

Data Structure

MongoDB uses JSON-like documents when storing documents in collections. The following figure compares data structures between relational databases and MongoDB, helping you have a better understanding of MongoDB concepts.

Figure 2-2 Data structures



Example of relational database data structure

MongoDB data structure example

Why DDS?

For details, see **Advantages** and **Typical Application Scenarios**.

Product Pricing

For details, see **Pricing Details**.

Compatible DB Engines and Versions

For details about the compatible DB engines and versions, see **DB Engines and Versions**.

Deployment Suggestions

Before deploying DDS, consider the following factors:

- Region and Availability Zone: You should select a region and AZs based on the geographical locations of your users, product prices, DR capabilities, and network latency. Once DB instances are successfully created in specific AZs in a region, the region cannot be changed. For more information, see **Region and AZ**.
- Network planning: When creating a DDS DB instance, you are advised to use the Virtual Private Network (VPC) and subnet where the Elastic Cloud Server (ECS) is deployed.
- Data security: DDS provides comprehensive assurance for data security. You can deploy DB instances in multiple AZs and use audit logs, isolated networks, security groups, and data encryption to ensure data security.

3 Advantages

DDS provides several types of DB instances with high reliability and scalability. You can manage them using various O&M tools in a visual way.

Diverse Instance Types

There are many instance types to meet your requirements in different scenarios.

Table 3-1 Architecture

Instance Architecture	Description	Application Scenario
Cluster	A DDS cluster consists of three types of nodes: mongos, shard, and config. You can select the number of mongos and shard nodes and the cluster configuration you need based on your performance requirements.	Cluster instances are recommended for service systems that require both high availability and scalability.
Replica set	A DDS replica set consists of three nodes: primary, secondary, and hidden. Data is automatically synchronized between nodes to ensure high data reliability. If a primary node goes down or becomes faulty, the replica set elects a secondary node as a new primary node and continues normal operations.	Replica set instance suits small- and medium-sized service systems that require high availability.
Single node	While not as robust as replica sets, single node architecture can be a less expensive way to visualize O&M and provide elastic scaling.	Single node instances are useful for R&D, testing, and non-core data storage of enterprises.

Elastic Scaling

DDS allows you to change the storage space, node quantity, CPU, and memory of an instance.

Friendly UI

A web-based console provides comprehensive monitoring information, making your operations easy and visual.

Secure

- VPCs, subnets, security groups, storage encryption, SSL, and DDoS protection are provided to defend against various malicious attacks and ensure data security.
- DDS supports audit logs that record your operations on databases or collections, helping you audit database security and analyze root cause of faults.
- Fine-grained permission management is supported.

Abundant O&M Tools

- Backup and restoration

  DDS supports full backup, incremental backup, and point-in-time backup and restoration of replica sets. Backup data can be stored for up to 732 days.
- Monitoring and alarm reporting

  DDS monitors instance metrics. You can customize the objects you want to monitor and configure notification policies to learn about the database service status in a timely manner.

Other advantages

DDS also boasts outstanding service availability, data durability, system security, and maintenance cost. For details, see **Comparison Between DDS and On-Premises Databases**.

4 Comparison Between DDS and On-Premises Databases

DDS provides high availability. It is more reliable, secure, and cost-effective than self-built databases.

Table 4-1 Comparison between DDS and on-premises databases

Item	DDS	On-Premises Database
Service availability	99.95%	Requires setting up the primary/standby relationship and HA environment for ensuring the availability.
Data durability	99.99999999%	Requires self-guarantee, primary/standby relationship setup, and RAID setup.
System security	• Anti-Distributed denial of service (DDoS) protection is provided, and system vulnerabilities are automatically addressed in a timely manner. • Supports log auditing.	• Requires procurement of expensive devices and software, as well as manual detection and fix of security vulnerabilities. • Requires procurement of an audit system.

Item	DDS	On-Premises Database
Database backup	• Automated backup is supported. You can configure backup policies based on service requirements. • You can create manual or physical backups at any time. DDS helps increase backup efficiency by 3 times. • Backup files are automatically uploaded to Object Storage Service (OBS) for storage.	• Requires self-setup and maintenance. • The open-source version supports only logical backup, resulting in low backup efficiency.
Monitoring and alarm reporting	DDS interconnects with Cloud Eye, which provides a visualized platform for you to view monitoring metrics and set threshold alarms.	You have to purchase three servers and hosting fees can be expensive.
Hosting	There are no hosting fees.	Requires purchase of three servers and payment of high hosting fees.
Maintenance cost	There are no additional O&M costs, and second-by-second performance monitoring, threshold alarms, and event alarms can all be configured.	Requires large manpower investment and professional database administrator (DBA) for maintenance.
Deployment and scaling	Supports quick deployment, flexible application, elastic scaling, and one-click specification change.	Requires procurement, deployment, and coordination of hardware that matches original devices.
Log transfer	Stores slow query logs and error logs for one month.	Requires you to transfer, export, and query log information on your own.
High availability	Provides high availability capabilities, supporting second-level switchover and failover.	Requires self-setup of HA monitoring. Data may be lost after a manual primary/secondary switchover is performed using commands.

5 Typical Application Scenarios

Games

Player information generated, like their equipment and bonus points, are stored in DDS databases. During peak hours, DDS cluster instances can handle large amounts of concurrent requests. DDS cluster and replica set provide high availability to ensure the games are stable in high-concurrency scenarios.

In addition, DDS is compatible with MongoDB and provides a non-schema mode, which frees you from having to change table structures when the play modes change. DDS can meet the flexible gaming requirements. You can store structured data with fixed schemas in Relational Database Service (RDS), store services with flexible schemas in DDS, and store hot data in GaussDB(for Redis), improving data efficiency and reducing data storage costs.

Advantages:

- **Supports Embedded Documents**: Embedded documents eliminate the need for JOIN statements, which simplifies application development. Flexible schemas also facilitate rapid development and iteration.
- **Easy to Cope with Peak Pressure**: Sharded clusters provide enough capacity to store data into the TB range.

IoT

DDS is compatible with MongoDB and provides high-performance and asynchronous data write functionality. In certain scenarios, DDS can process data in the memory database. In addition, cluster instances can dynamically add mongos and shard nodes or upgrade specifications. The performance and storage space can be quickly expanded, making cluster instances suitable for IoT scenarios with high concurrent writes.

Intelligent IoT terminals need to collect various types of data, store device logs, and analyze various types of information. In recent years, IoT services have grown rapidly, generating huge volumes of data and increasing access traffic. IoT has created demand for horizontal storage scaling.

DDS provides a secondary index to meet dynamic query requirements and uses the MapReduce aggregation framework, which is compatible with MongoDB, to analyze data from multiple dimensions.

Advantages:

- **High Write Performance**: DDS sharded clusters provide the robust write performance needed to handle terabyte-scale databases.
- **High Performance and Scalability**: DDS supports applications with high QPS rates, and its sharding architecture can be scaled in or out to flexibly cope with application changes.

Internet

DDS replica sets use the three-node HA architecture. Three data nodes form an anti-affinity group and are deployed on different physical servers to automatically synchronize data. The primary and secondary nodes provide services. Each node has a private IP address and works with Driver to allocate read workloads.

Many organizations need to process and store data into the TB range, requiring data to be written to databases in real time and dynamic analysis capabilities in big data computing.

Advantages:

- **MapReduce:** With a complete data analysis utility, you can query statements or scripts, and distribute requests to DDS.
- **Excellent Scalability**: DDS DB instances can be scaled up to support growing services and data volumes in content management systems.

6 Functions and Features

This section describes the main functions and features of DDS.

Migrating Data

With Data Replication Service (DRS), you can migrate self-built MongoDB databases or other cloud MongoDB databases to DDS in real time. DRS simplifies data transmission between databases and reduces data transmission costs.

Scaling up Storage

If the storage you purchased cannot meet your requirements, you can **scale up the instance**.

Changing Instance Class

If the CPU or memory of an instance cannot meet your service requirements, you can **change the instance class** on the console.

Backup and Restoration

- You can **configure the automated backup policy** and **create backups**. The maximum retention period for an automated backup is 732 days. A manual backup can be retained until you delete it.
- Data can be **restored using backup files**. Replica set instances support point-in-time recovery at the instance level, database level and table level.

Monitoring Metrics

DDS **monitors key performance metrics** of instances and DB engines in real time, including the CPU usage, memory usage, storage space usage, command execution frequency, delete statement execution frequency, insert statement execution frequency, and number of active connections.

Managing Logs

DDS allows you to query and download database **error logs** and **slow query logs**.

Configuring Parameters

DDS allows you to manage parameter templates and **modify parameter template settings** on the console.

Cross-AZ Disaster Recovery

If the region where your instance is located supports three or more AZs, you can deploy the instance across three AZs to achieve cross-AZ disaster recovery.

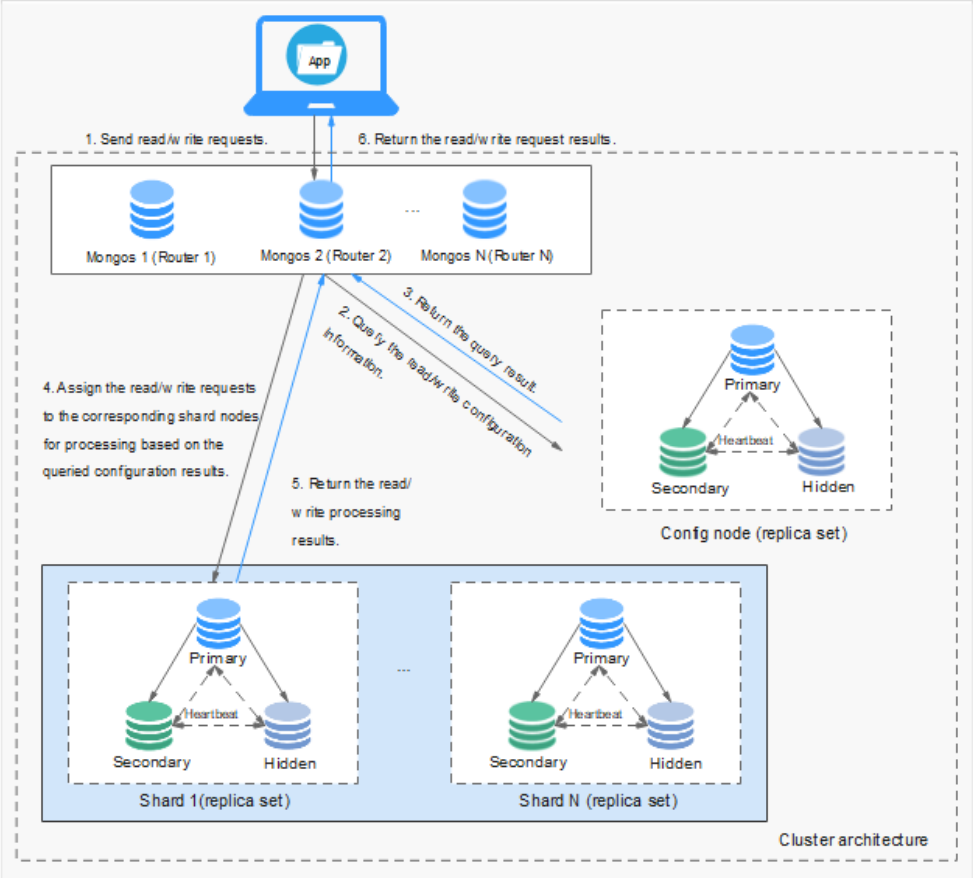
7 System Architecture

7.1 Cluster

A cluster consists of a config node, and multiple mongos and shard nodes.

Data read and write requests are forwarded by the mongos nodes, which read configuration settings from config, and then allocate the read and write requests to the shards, making it easy to cope with high concurrency scenarios. In addition, each config node, along with the shards in its cluster, is replicated in triplicate to ensure high availability.

Figure 7-1 Cluster architecture



- Each mongos is a single node, but you can provision multiple mongos nodes for load balancing and failovers. A single cluster can contain 2 to 16 mongos nodes.
- Each shard is a three-node replica set. A single cluster supports 2 to 16 shards.
- A config node is a necessary part of a cluster instance, and is also deployed as a replica set. The config node stores instance configuration data.
- Cluster is suitable for handling large amounts of data and can meet the requirements for high availability and scalability.

7.2 Replica Set

A replica set consists of a set of mongod processes. It is a collection of nodes that help ensure data redundancy and reliability.

NOTE

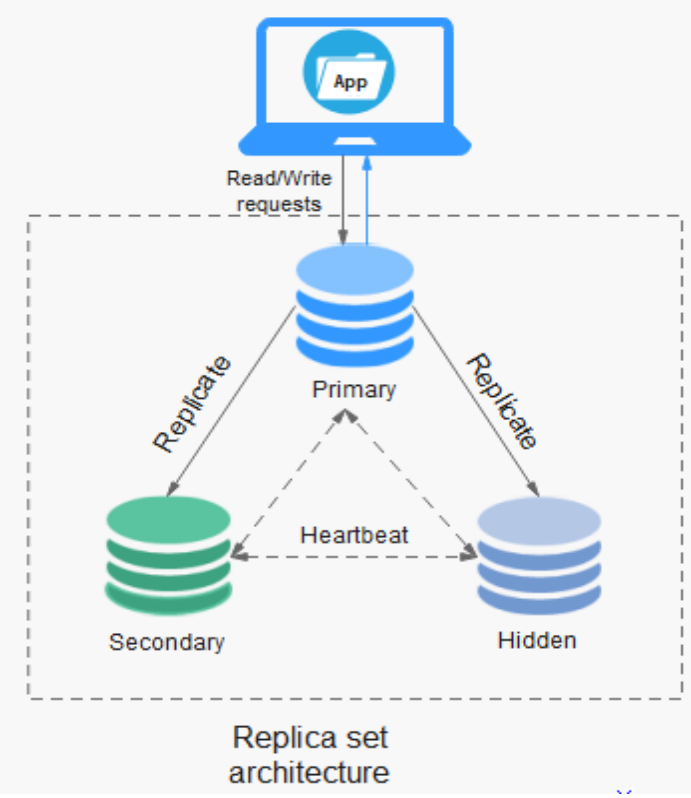
For details about the mongod process, see the MongoDB **official documentation**.

A replica set consists of three nodes: primary, secondary, and hidden. The three-node architecture is set up automatically, and the three nodes automatically synchronize data with each other to ensure data reliability. Replica sets are recommended for small- and medium-sized service systems that require high availability.

- Primary node: Primary nodes are used to process both read and write requests.
- Secondary node: Secondary nodes are used to process read requests only.
- Hidden node: Hidden nodes are used to back up data.

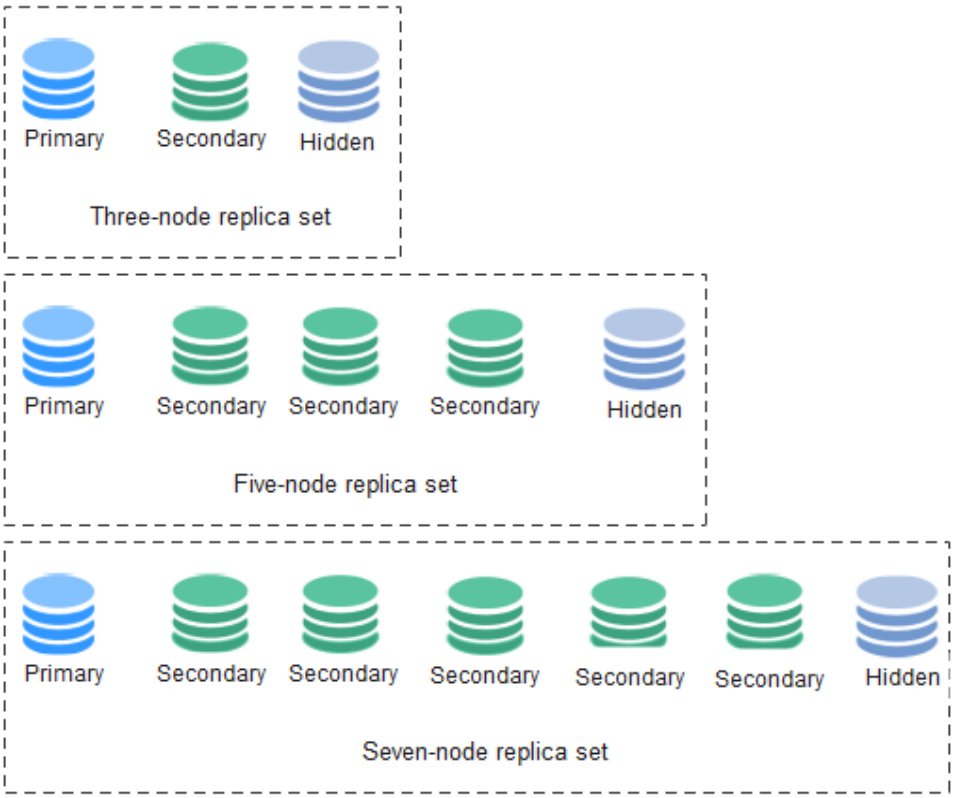
You can perform operations on the primary and secondary nodes. If the primary node is faulty, the system automatically selects a new primary node. The following figure shows the replica set architecture.

Figure 7-2 Three-node replica set architecture



After a replica set is created, you can add up to either 5 or 7 nodes. For details about how to add a replica set node, see **Adding Replica Set Instance Nodes**.

Figure 7-3 Replica set instance nodes

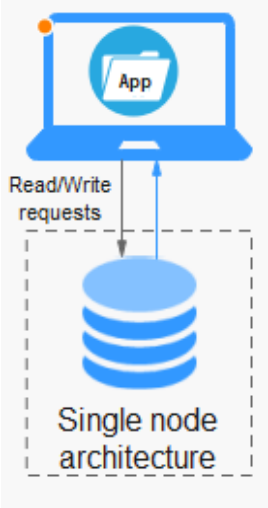


7.3 Single Node

A single node instance, as its name suggests, contains only one node and that node can be accessed directly.

As a supplement to the cluster and replica set architectures, the single-node architecture is useful for R&D, testing, and non-core data storage of enterprises. While not as robust as replica sets, single node architecture can be a less expensive way to visualize O&M and provide elastic scaling.

Figure 7-4 Single node architecture



8 Instances

8.1 DB Engines and Versions

DDS supports versions 3.4, 4.0, and 4.2, and you need to use a driver compatible with MongoDB 3.0 or later to access DDS. You can select the DB engine and version you need based on your service requirements.

Table 8-1 Supported DB engines and versions

Compatibility	CPU Type	DB Instance Type	Storage Engine
4.2	x86	• Cluster • Replica set • Single node	RocksDB
4.0	• x86 • Kunpeng	• Cluster • Replica set • Single node	WiredTiger
3.4	• x86 • Kunpeng	• Cluster • Replica set • Single node	WiredTiger

For details about the compatibility between DDS and MongoDB Community Editions, see **Versions**.

8.2 Instance Specifications

8.2.1 Cluster

Instances of the same type have different memory specifications. You can select instances of different specifications based on application scenarios.

This section describes cluster instance specifications. The instance specifications depend on the selected CPU model.

NOTE

The default maximum number of connections has been adjusted. Query the number of connections based on the instance creation time. For details about the number of connections for instances created before July 2021, see **Historical Node Specifications**.

Node Specifications

Table 8-2 config specifications

CPU Type	Specifications	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
x86	Enhanced II	2	4	dds.mongodb.c6.large.2.config	2000	200-2000
Kunpeng	-	2	4	dds.mongodb.large.arm2.config	2000	200-2000
		4	8	dds.mongodb.xlarge.arm2.config	4000	200-4000
		8	16	dds.mongodb.2xlarge.arm2.config	16000	200-16000

Table 8-3 mongos and shard specifications

CPU Type	Specifications	Component	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
x86	General-purpose	mongos	1	4	dds.mongodb.s6.medium.4.mongos	1000	200-1000
			2	4	dds.mongodb.s6.large.2.mongos	2000	200-2000

CPU Type	Specifications	Component	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
			2	8	dds.mongodb.s6.large.4.mongos	2000	200-2000
			4	8	dds.mongodb.s6.xlarge.2.mongos	4000	200-4000
			4	16	dds.mongodb.s6.xlarge.4.mongos	4000	200-4000
			8	16	dds.mongodb.s6.2xlarge.2.mongos	16000	200-16000
			8	32	dds.mongodb.s6.2xlarge.4.mongos	16000	200-16000
		shard	1	4	dds.mongodb.s6.medium.4.shard	1000	200-1000
			2	4	dds.mongodb.s6.large.2.shard	2000	200-2000
			2	8	dds.mongodb.s6.large.4.shard	2000	200-2000
			4	8	dds.mongodb.s6.xlarge.2.shard	4000	200-4000
			4	16	dds.mongodb.s6.xlarge.4.shard	4000	200-4000
			8	16	dds.mongodb.s6.2xlarge.2.shard	16000	200-16000
			8	32	dds.mongodb.s6.2xlarge.4.shard	16000	200-16000
	Enhanced II	mongos	2	8	dds.mongodb.c6.large.4.mongos	2000	200-2000
			4	16	dds.mongodb.c6.xlarge.4.mongos	4000	200-4000
			8	32	dds.mongodb.c6.2xlarge.4.mongos	16000	200-16000
			16	64	dds.mongodb.c6.4xlarge.4.mongos	16000	200-16000

CPU Type	Specifications	Component	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
			32	128	dds.mongodb.c6.8xlarge.4.mongos	16000	200-16000
			64	256	dds.mongodb.c6.16xlarge.4.mongos	16000	200-16000
		shard	2	8	dds.mongodb.c6.large.4.shard	2000	200-2000
			4	16	dds.mongodb.c6.xlarge.4.shard	4000	200-4000
			8	32	dds.mongodb.c6.2xlarge.4.shard	16000	200-16000
			16	64	dds.mongodb.c6.4xlarge.4.shard	16000	200-16000
			32	128	dds.mongodb.c6.8xlarge.4.shard	16000	200-16000
			64	256	dds.mongodb.c6.16xlarge.4.shard	16000	200-16000
Kunpeng	-	mongos	2	4	dds.mongodb.large.arm2.mongos	2000	200-2000
	-		2	8	dds.mongodb.large.arm4.mongos	2000	200-2000
	-		4	8	dds.mongodb.xlarge.arm2.mongos	4000	200-4000
	-		4	16	dds.mongodb.xlarge.arm4.mongos	4000	200-4000
	-		8	16	dds.mongodb.2xlarge.arm2.mongos	16000	200-16000
	-		8	32	dds.mongodb.2xlarge.arm4.mongos	16000	200-16000

CPU Type	Specifications	Component	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
	-		16	32	dds.mongodb.4xlarge.arm2.mongos	16000	200-16000
	-		16	64	dds.mongodb.4xlarge.arm4.mongos	16000	200-16000
	-	shard	2	4	dds.mongodb.large.arm2.shard	2000	200-2000
	-		2	8	dds.mongodb.large.arm4.shard	2000	200-2000
	-		4	8	dds.mongodb.xlarge.arm2.shard	4000	200-4000
	-		4	16	dds.mongodb.xlarge.arm4.shard	4000	200-4000
	-		8	16	dds.mongodb.2xlarge.arm2.shard	16000	200-16000
	-		8	32	dds.mongodb.2xlarge.arm4.shard	16000	200-16000
	-		16	32	dds.mongodb.4xlarge.arm2.shard	16000	200-16000
	-		16	64	dds.mongodb.4xlarge.arm4.shard	16000	200-16000

Historical Node Specifications

For details about the range of connections of instances created before July 2021, see the following table.

Table 8-4 config specifications

Instance Type	CPU Type	Specifications	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
Cluster	x86	General-purpose	2	4	dds.mongodb.s6.large.2.config	400	200-2000
		Enhanced	2	4	dds.mongodb.c3.large.2.config		
		Enhanced II	2	4	dds.mongodb.c6.large.2.config		
	Kunpeng	-	2	4	dds.mongodb.large.arm2.config		

Table 8-5 mongos and shard specifications

Instance Type	CPU Type	Specifications	Component	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
Cluster	x86	General-purpose	mongos	1	4	dds.mongodb.s6.medium.4.mongos	400	200-1000
				2	4	dds.mongodb.s6.large.2.mongos	400	200-2000
				2	8	dds.mongodb.s6.large.4.mongos	400	200-2000
				4	8	dds.mongodb.s6.xlarge.2.mongos	1000	200-4000

Instance Type	CPU Type	Specifications	Component	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
				4	16	dds.mongodb.s6.xlarge.4.mongos	1000	200-4000
				8	16	dds.mongodb.s6.2xlarge.2.mongos	4000	200-16000
				8	32	dds.mongodb.s6.2xlarge.4.mongos	4000	200-16000
			shard	1	4	dds.mongodb.s6.medium.4.shard	400	200-1000
				2	4	dds.mongodb.s6.large.2.shard	400	200-2000
				2	8	dds.mongodb.s6.large.4.shard	400	200-2000
				4	8	dds.mongodb.s6.xlarge.2.shard	1000	200-4000
				4	16	dds.mongodb.s6.xlarge.4.shard	1000	200-4000
				8	16	dds.mongodb.s6.2xlarge.2.shard	4000	200-16000
				8	32	dds.mongodb.s6.2xlarge.4.shard	4000	200-16000
		Enhanced	mongos	2	8	dds.mongodb.c3.large.4.mongos	400	200-2000

Instance Type	CPU Type	Specifications	Component	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
				4	16	dds.mongodb.c3.xlarge.4.mongos	1000	200-4000
				8	32	dds.mongodb.c3.2xlarge.4.mongos	4000	200-16000
				16	64	dds.mongodb.c3.4xlarge.4.mongos	8000	200-16000
				32	128	dds.mongodb.c3.8xlarge.4.mongos	8000	200-16000
				60	256	dds.mongodb.c3.15xlarge.4.mongos	8000	200-16000
			shard	2	8	dds.mongodb.c3.large.4.shard	400	200-2000
				4	16	dds.mongodb.c3.xlarge.4.shard	1000	200-4000
				8	32	dds.mongodb.c3.2xlarge.4.shard	4000	200-16000
				16	64	dds.mongodb.c3.4xlarge.4.shard	8000	200-16000
				32	128	dds.mongodb.c3.8xlarge.4.shard	8000	200-16000
				60	256	dds.mongodb.c3.15xlarge.4.shard	8000	200-16000

Instance Type	CPU Type	Specifications	Component	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
		Enhanced II	mongos	2	8	dds.mongodb.c6.large.4.mongos	400	200-2000
				4	16	dds.mongodb.c6.xlarge.4.mongos	1000	200-4000
				8	32	dds.mongodb.c6.2xlarge.4.mongos	4000	200-16000
				16	64	dds.mongodb.c6.4xlarge.4.mongos	8000	200-16000
				32	128	dds.mongodb.c6.8xlarge.4.mongos	8000	200-16000
				64	256	dds.mongodb.c6.16xlarge.4.mongos	8000	200-16000
			shard	2	8	dds.mongodb.c6.large.4.shard	400	200-2000
				4	16	dds.mongodb.c6.xlarge.4.shard	1000	200-4000
				8	32	dds.mongodb.c6.2xlarge.4.shard	4000	200-16000
				16	64	dds.mongodb.c6.4xlarge.4.shard	8000	200-16000
				32	128	dds.mongodb.c6.8xlarge.4.shard	8000	200-16000

Instance Type	CPU Type	Specifications	Component	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
				64	256	dds.mongodb.c6.16xlarge.4.shard	8000	200-16000
	Kunpeng	-	mongos	2	4	dds.mongodb.large.arm2.mongos	400	200-2000
		-		2	8	dds.mongodb.large.arm4.mongos	400	200-2000
		-		4	8	dds.mongodb.xlarge.arm2.mongos	1000	200-4000
		-		4	16	dds.mongodb.xlarge.arm4.mongos	1000	200-4000
		-		8	16	dds.mongodb.2xlarge.arm2.mongos	4000	200-16000
		-		8	32	dds.mongodb.2xlarge.arm4.mongos	4000	200-16000
		-		16	32	dds.mongodb.4xlarge.arm2.mongos	8000	200-16000
		-		16	64	dds.mongodb.4xlarge.arm4.mongos	8000	200-16000
		-	shard	2	4	dds.mongodb.large.arm2.shard	400	200-2000

Instance Type	CPU Type	Specifications	Component	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
		-		2	8	dds.mongodb.large.arm4.shard	400	200-2000
		-		4	8	dds.mongodb.xlarge.arm2.shard	1000	200-4000
		-		4	16	dds.mongodb.xlarge.arm4.shard	1000	200-4000
		-		8	16	dds.mongodb.2xlarge.arm2.shard	4000	200-16000
		-		8	32	dds.mongodb.2xlarge.arm4.shard	4000	200-16000
		-		16	32	dds.mongodb.4xlarge.arm2.shard	8000	200-16000
		-		16	64	dds.mongodb.4xlarge.arm4.shard	8000	200-16000

Helpful Links

1. How do I change the maximum number of connections of a cluster instance?

   You can change the maximum number of connections to a DB instance by modifying the **net.maxIncomingConnections** parameter. For details about how to change parameter values, see **Editing a Parameter Template**.

2. How do I purchase a cluster instance?

   For details, see **Buying a Cluster Instance**.

3. How do I change the CPU or memory of a cluster instance?

   You can change the CPU and memory of mongos and shards nodes. For details, see **Changing the CPU and Memory of a Cluster Instance**.

4. How do I scale up the storage space of a cluster instance?

   You can scale up the storage of shards nodes. For details, see **Scaling Up Storage**.

8.2.2 Replica Set

Instances of the same type have different memory specifications. You can select instances of different specifications based on application scenarios.

This section describes replica set instance specifications. The instance specifications depend on the selected CPU model.

NOTE

The default maximum number of connections has been adjusted. Query the number of connections based on the instance creation time. For details about the number of connections for instances created before July 2021, see **Historical Instance Specifications**.

Instance Specifications

Table 8-6 lists the specifications of replica set instances.

Table 8-6 Replica set instance specifications

CPU Type	Specifications	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
x86	General-purpose	2	4	dds.mongodb.s6.large.2.repset	1000	200-1000
		2	8	dds.mongodb.s6.large.4.repset	1000	200-1000
		4	8	dds.mongodb.s6.xlarge.2.repset	3000	200-3000
		4	16	dds.mongodb.s6.xlarge.4.repset	3000	200-3000
		8	16	dds.mongodb.s6.2xlarge.2.repset	8000	200-8000
		8	32	dds.mongodb.s6.2xlarge.4.repset	8000	200-8000

CPU Type	Specifications	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
	Enhanced II	2	8	dds.mongodb.c6.large.4.repset	1000	200-1000
		4	16	dds.mongodb.c6.xlarge.4.repset	3000	200-3000
		8	32	dds.mongodb.c6.2xlarge.4.repset	8000	200-8000
		16	64	dds.mongodb.c6.4xlarge.4.repset	16000	200-16000
		32	128	dds.mongodb.c6.8xlarge.4.repset	16000	200-16000
		64	256	dds.mongodb.c6.16xlarge.4.repset	16000	200-16000
Kunpeng	-	2	4	dds.mongodb.large.arm2.repset	1000	200-1000
	-	2	8	dds.mongodb.large.arm4.repset	1000	200-1000
	-	4	8	dds.mongodb.xlarge.arm2.repset	3000	200-3000
	-	4	16	dds.mongodb.xlarge.arm4.repset	3000	200-3000
	-	8	16	dds.mongodb.2xlarge.arm2.repset	8000	200-8000
	-	8	32	dds.mongodb.2xlarge.arm4.repset	8000	200-8000
	-	16	32	dds.mongodb.4xlarge.arm2.repset	16000	200-16000
	-	16	64	dds.mongodb.4xlarge.arm4.repset	16000	200-16000

Historical Instance Specifications

For details about the range of connections of instances created before July 2021, see the following table.

Table 8-7 Replica set instance specifications

CPU Type	Specifications	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
x86	General-purpose	2	4	dds.mongodb.s6.large.2.repset	400	200-1000
		2	8	dds.mongodb.s6.large.4.repset	400	200-1000
		4	8	dds.mongodb.s6.xlarge.2.repset	1000	200-3000
		4	16	dds.mongodb.s6.xlarge.4.repset	1000	200-3000
		8	16	dds.mongodb.s6.2xlarge.2.repset	4000	200-8000
		8	32	dds.mongodb.s6.2xlarge.4.repset	4000	200-8000
	Enhanced	2	8	dds.mongodb.c3.large.4.repset	400	200-1000
		4	16	dds.mongodb.c3.xlarge.4.repset	1000	200-3000
		8	32	dds.mongodb.c3.2xlarge.4.repset	4000	200-8000
		16	64	dds.mongodb.c3.4xlarge.4.repset	8000	200-16000
		32	128	dds.mongodb.c3.8xlarge.4.repset	8000	200-16000
		60	256	dds.mongodb.c3.15xlarge.4.repset	8000	200-16000
	Enhanced II	2	8	dds.mongodb.c6.large.4.repset	400	200-1000
		4	16	dds.mongodb.c6.xlarge.4.repset	1000	200-3000
		8	32	dds.mongodb.c6.2xlarge.4.repset	4000	200-8000

CPU Type	Specifications	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
		16	64	dds.mongodb.c6.4xlarge.4.repset	8000	200-16000
		32	128	dds.mongodb.c6.8xlarge.4.repset	8000	200-16000
		64	256	dds.mongodb.c6.16xlarge.4.repset	8000	200-16000
Kunpeng	-	2	4	dds.mongodb.large.arm2.repset	400	200-1000
	-	2	8	dds.mongodb.large.arm4.repset	400	200-1000
	-	4	8	dds.mongodb.xlarge.arm2.repset	1000	200-3000
	-	4	16	dds.mongodb.xlarge.arm4.repset	1000	200-3000
	-	8	16	dds.mongodb.2xlarge.arm2.repset	4000	200-8000
	-	8	32	dds.mongodb.2xlarge.arm4.repset	4000	200-8000
	-	16	32	dds.mongodb.4xlarge.arm2.repset	8000	200-16000
	-	16	64	dds.mongodb.4xlarge.arm4.repset	8000	200-16000

Helpful Links

1. How do I change the maximum number of connections of a replica set instance?

   You can change the maximum number of connections to a DB instance by modifying the **net.maxIncomingConnections** parameter. For details about how to change parameter values, see **Editing a Parameter Template**.

2. How do I buy a replica set instance?

   For details, see **Buying a Replica Set Instance**.

3. How do I change the CPU and memory of a replica set instance?

For details, see **Changing the CPU or Memory of a Replica Set Instance**.

4. How do I scale up the storage space of a replica set instance?

   For details, see **Scaling Up Storage**.

8.2.3 Single Node

Instances of the same type have different memory specifications. You can select instances of different specifications based on application scenarios.

This section describes single node instance specifications. The instance specifications depend on the selected CPU model.

NOTE

The default maximum number of connections has been adjusted. Query the number of connections based on the instance creation time. For details about the number of connections for instances created before July 2021, see **Historical Instance Specifications**.

Instance Specifications

Table 8-8 Single Node

CPU Type	Specifications	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
x86	General-purpose	1	4	dds.mongodb.s6.medium.4.single	500	200-500
		2	4	dds.mongodb.s6.large.2.single	1000	200-1000
		2	8	dds.mongodb.s6.large.4.single	1000	200-1000
		4	8	dds.mongodb.s6.xlarge.2.single	8000	200-8000
		4	16	dds.mongodb.s6.xlarge.4.single	8000	200-8000
		8	16	dds.mongodb.s6.2xlarge.2.single	10000	200-10000
		8	32	dds.mongodb.s6.2xlarge.4.single	10000	200-10000
	Enhanced II	2	8	dds.mongodb.c6.large.4.single	500	200-500

CPU Type	Specifications	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
		4	16	dds.mongodb.c6.xlarge.4.single	1000	200-1000
		8	32	dds.mongodb.c6.2xlarge.4.single	4000	200-4000
		16	64	dds.mongodb.c6.4xlarge.4.single	8000	200-8000
		32	128	dds.mongodb.c6.8xlarge.4.single	16000	200-16000
		64	256	dds.mongodb.c6.16xlarge.4.single	16000	200-16000
Kunpeng	-	2	4	dds.mongodb.large.arm2.single	1000	200-1000
	-	2	8	dds.mongodb.large.arm4.single	1000	200-1000
	-	4	8	dds.mongodb.xlarge.arm2.single	8000	200-8000
	-	4	16	dds.mongodb.xlarge.arm4.single	8000	200-8000
	-	8	16	dds.mongodb.2xlarge.arm2.single	10000	200-10000
	-	8	32	dds.mongodb.2xlarge.arm4.single	10000	200-10000
	-	16	32	dds.mongodb.4xlarge.arm2.single	16000	200-16000
	-	16	64	dds.mongodb.4xlarge.arm4.single	16000	200-16000

Historical Instance Specifications

For details about the range of connections of instances created before July 2021, see the following table.

Table 8-9 Single Node

CPU Type	Specifications	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
x86	General-purpose	1	4	dds.mongodb.s6.medium.4.single	400	200-500
		2	4	dds.mongodb.s6.large.2.single	400	200-1000
		2	8	dds.mongodb.s6.large.4.single	400	200-1000
		4	8	dds.mongodb.s6.xlarge.2.single	1000	200-8000
		4	16	dds.mongodb.s6.xlarge.4.single	1000	200-8000
		8	16	dds.mongodb.s6.2xlarge.2.single	4000	200-10000
		8	32	dds.mongodb.s6.2xlarge.4.single	4000	200-10000
	Enhanced	2	8	dds.mongodb.c3.large.4.single	400	200-1000
		4	16	dds.mongodb.c3.xlarge.4.single	1000	200-8000
		8	32	dds.mongodb.c3.2xlarge.4.single	4000	200-10000
		16	64	dds.mongodb.c3.4xlarge.4.single	8000	200-16000
		32	128	dds.mongodb.c3.8xlarge.4.single	8000	200-16000
		60	256	dds.mongodb.c3.15xlarge.4.single	8000	200-16000
	Enhanced II	2	8	dds.mongodb.c6.large.4.single	400	200-500
		4	16	dds.mongodb.c6.xlarge.4.single	1000	200-1000

CPU Type	Specifications	vCPUs	Memory (GB)	Flavor ID	Default Maximum Number of Connections	Range of Connections
		8	32	dds.mongodb.c6.2xlarge.4.single	4000	200-4000
		16	64	dds.mongodb.c6.4xlarge.4.single	8000	200-8000
		32	128	dds.mongodb.c6.8xlarge.4.single	8000	200-16000
		64	256	dds.mongodb.c6.16xlarge.4.single	8000	200-16000
Kunpeng	-	2	4	dds.mongodb.large.arm2.single	400	200-1000
	-	2	8	dds.mongodb.large.arm4.single	400	200-1000
	-	4	8	dds.mongodb.xlarge.arm2.single	1000	200-8000
	-	4	16	dds.mongodb.xlarge.arm4.single	1000	200-8000
	-	8	16	dds.mongodb.2xlarge.arm2.single	4000	200-10000
	-	8	32	dds.mongodb.2xlarge.arm4.single	4000	200-10000
	-	16	32	dds.mongodb.4xlarge.arm2.single	8000	200-16000
	-	16	64	dds.mongodb.4xlarge.arm4.single	8000	200-16000

Helpful Links

1. How do I change the maximum number of connections of a single node instance?

   You can change the maximum number of connections to a DB instance by modifying the **net.maxIncomingConnections** parameter. For details about how to change parameter values, see **Editing a Parameter Template**.

2. How do I buy a single node instance?
   For details, see **Buying a Single Node Instance**.
3. How do I change the CPU and memory of a single node instance?
   For details, see **Changing the CPU or Memory of a Single Node Instance**.
4. How do I scale up the storage of a single node instance?
   For details, see **Scaling Up Storage**.

8.3 Read Replica

To improve the read capability of the primary node, DDS provides read replicas that can be accessed independently, which is good for handling read-intensive workloads and reducing memory pressure on the primary node. You can also create one or more read replicas to process a large number of read requests and improve throughput.

Differences Between Read Replica and Secondary Node

Node	Description	Application Scenario
Secondary nodes	The secondary node of a replica set instance ensures high availability. If a primary node is faulty, the system automatically switches traffic to the secondary node. If the primary node is faulty, each secondary node may be elected as a new primary node to execute data write requests.	DDS provides an HA connection address to connect to both the primary and secondary nodes and separate read and write operations. In this way, a DDS instance can handle heavy read workloads with ease. In addition, the impact of node faults on services is wakened while the performance is improved.
Read replicas	A read replica in a replica set instance synchronizes data only from the primary node. It does not ensure high availability and will not be elected as the primary node. Read replicas can be connected independently and do not interfere with primary nodes.	Read replicas are useful when applications need to read large amounts of data from existing instances.

Constraints

- Only replica set instances can have read replicas.
- Instances that only have read replicas cannot be created.
- DDS instance version must be 3.4, 4.0, and 4.2.
- Read replicas are used only for processing read requests and do not participate in primary/secondary node election.

- A maximum of five read replicas can be added to a replica set instance. For details, see **Adding Read Replicas to a Replica Set Instance**.
- Data is asynchronously replicated between the primary node and the read replica, and this process will delay by a few seconds.

Functions

- You can change the number of read replicas to meet fluctuating service demands and reduce service costs.
- The specifications of a read replica are the same as those of the primary node. Data is automatically synchronized to the read replica.
- Resources on the primary node are not occupied. Adding or deleting read replicas does not affect services on the primary node.
- Read replicas can be connected independently and do not interfere with existing primary and secondary nodes. For details, see **Connecting to Read Replicas Using Mongo Shell**.
- You can monitor the performance of read replicas.

Billing

- After a read replica is added to the current instance, you will be billed for using the VM and disks of the read replica.
- The price of a read replica is equal to that of a single node in a replica set instance or a shard node in a cluster instance. For example, if the price of a three-node replica set instance is USD $3,000, the price of a read replica is USD $1,000.

9 Pricing Details

HUAWEI CLOUD Document Database Service (DDS) provides different DB instance types, including replica set and cluster. These instances are billed on a pay-per-use basis and there are no minimum usage requirements.

Pricing

Table 9-1 Description

Item	Description	Billing Standards
Instance specifications	The instance specifications that you select. For a cluster instance, the mongos, shard, and config nodes are billed based on their specifications.	You can use the DDS price calculator to quickly obtain an estimate price of a DB instance with the specifications you select.
Storage	Storage space is billed based on the capacity you select. For cluster instances, the storage fees include the storage fees of shard and config nodes.	
Backup storage (optional)	DDS backup files are stored in OBS and do not occupy the storage of DDS instances. DDS provides backup storage up to 100% of your provisioned database storage at no additional fees. If the backup storage usage exceeds 100% of your provisioned database storage, you will be charged for resource usage.	

Item	Description	Billing Standards
Audit log space (optional)	An audit log records operations performed on your databases and collections. The generated log files are stored in OBS.	Enabling audit log incurs certain fees.
(Optional) Public network traffic	• DDS instances can be accessed from the Internet, and traffic fees are generated accordingly. • You are not charged for the traffic generated in the cloud.	-

Billing Mode

DDS provides two billing modes: pay-per-use and yearly/monthly. Pay-per-use is recommended if you are unsure of your future service needs and want to avoid paying for unused resources. However, if you are sure of your needs, yearly/monthly will be less expensive.

- Yearly/Monthly is a prepaid billing mode, which provides a larger discount than pay-per-use billing and is recommended for long-term use of resources.
- Pay-per-use (hourly): This billing mode is flexible. You pay only for what you use. Pricing is listed on a per-hour basis, but bills are calculated based on the actual usage duration.
- You can change the billing mode from pay-per-use to yearly/monthly or vice versa.

  For operation details, see **Changing the Billing Mode from Pay-per-Use to Yearly/Monthly** and **Changing the Billing Mode from Yearly/Monthly to Pay-per-Use**.

Configuration Changes

You can change the specifications of an existing DB instance based on your service requirements. After the change, the instance is billed based on the new specifications.

- Adding instances or nodes: You can add instances to handle growing services. After the instances are added, you will be charged based on the new number of instances. The number of cluster instances can only be increased. The number of mongos nodes ranges from 2 to 32, the number of shard nodes ranges from 2 to 32, and the number of config nodes is 1 by default.
- Scaling up storage space: You can change the storage space based on service requirements. After scaling the storage space, you will be charged based on the new storage space. Note that storage space cannot be scaled down.

Renewal

DDS supports both yearly/monthly and pay-per-use billing modes.

- Pay-per-use: You are charged for a full hour regardless of whether you use the full or partial hour. You can use the service as long as your account balance is sufficient. If your account balance is insufficient, the service cannot be used normally. Top up your account in a timely manner.
- Yearly/monthly: An upfront payment is required when you purchase the service. No additional fees are incurred during your usage.

10 Basic Concepts

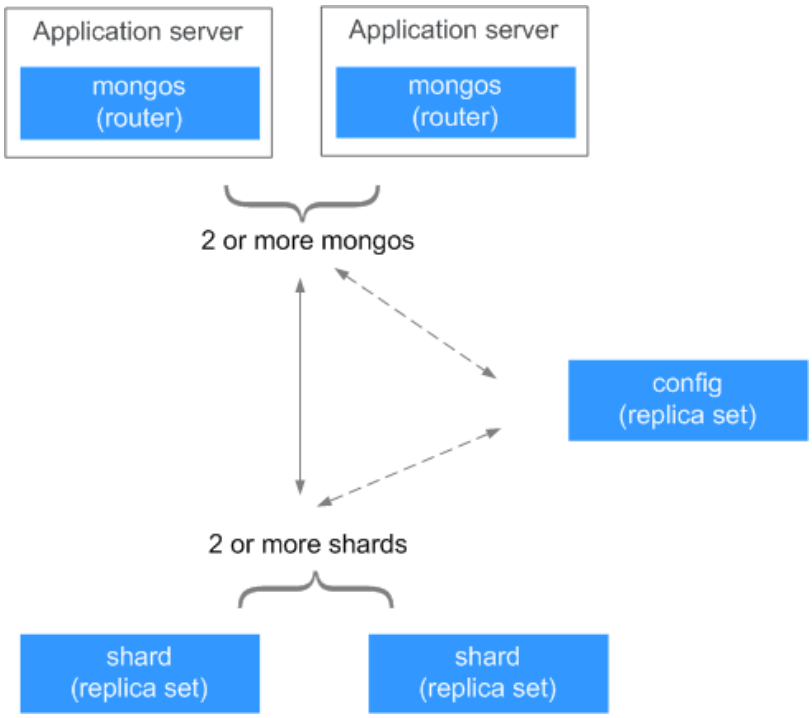
10.1 Instance

- An instance is a basic management unit of DDS. One or more databases can be created in a single DDS instance, and one or more collections can be created in each database.
- You can create and manage DB instances of various types and versions on the management console. For details about DB instance types, versions, and specifications, see **Instances**.
- Resources, such as the CPU, memory, and I/O, of each DDS DB instance are isolated from each other.

10.2 Cluster

Each DDS cluster consists of a config node, and multiple mongos and shard nodes. The following diagram shows the node relationships.

Figure 10-1 Diagram of node relationships



mongos

A mongos is a router for reading and writing data, providing a unified interface for accessing DB instances.

- Each DB instance has 2 to 16 mongos nodes. You can specify the quantity.
- A mongos reads configuration settings from configs and allocates read and write requests to shards. You can connect to a mongos directly.

config

A config stores configuration settings for DB instances and consists of one replica set.

You cannot connect to a config node directly.

shard

Shards are used to store user data.

- Each Community Edition instance has 2 to 16 shard nodes. You can specify the quantity.
- Each shard node is deployed as a replica set to ensure data redundancy and high reliability.
- You cannot connect to a shard node directly.

10.3 Database Parameter Template

A database parameter template is a collection of configuration parameters and values and can be applied to multiple DB instances.

10.4 Region and AZ

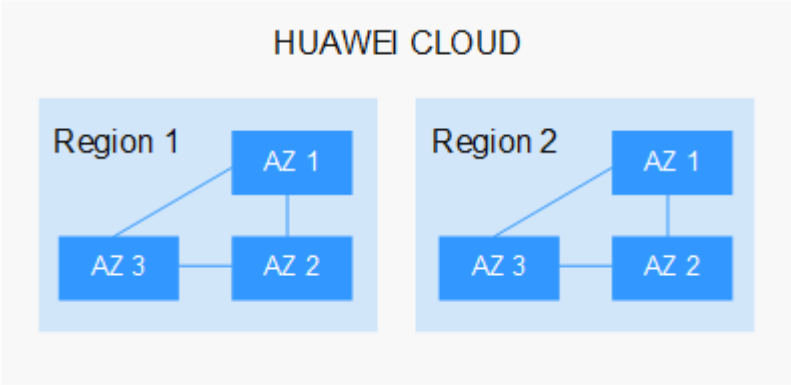
Concept

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

- Regions are defined by their geographical location and network latency. Public services, such as Elastic Cloud Server (ECS), Elastic Volume Service (EVS), Object Storage Service (OBS), Virtual Private Cloud (VPC), Elastic IP (EIP), and Image Management Service (IMS), are shared within the same region. There are universal regions and dedicated regions. A universal region provides universal cloud services for common tenants. A dedicated region only provides services of a specific type and only or for specific tenants.
- An AZ contains one or multiple physical data centers. Each AZ has independent cooling, fire extinguishing, moisture-proof, and electrical facilities. Within an AZ, computing, network, storage, and other resources are logically divided into multiple clusters. AZs within a region are interconnected using high-speed optical fibers to allow you to build cross-AZ high-availability systems.

Figure 10-2 shows the relationship between regions and AZs.

Figure 10-2 Region and AZ



HUAWEI CLOUD provides services in many regions around the world. You can select a region and AZ as needed.

Selecting a Region

When selecting a region, consider the following factors:

- Location

  You are advised to select a region close to you or your target users. This reduces network latency and improves the access rate. However, since all Chinese mainland regions provide basically the same infrastructure, BGP network quality, as well as operations and resource configurations, any of them will work well if your users are located there.

Selecting an AZ

When determining whether to deploy resources in the same AZ, consider your applications' requirements for disaster recovery (DR) and network latency.

- For better DR resilience, deploy resources in different AZs in the same region.
- For lower network latency, deploy resources in the same AZ.

Regions and Endpoints

Before using an API to call resources, specify its region and endpoint.

10.5 DB Connections

The number of connections is the number of databases that can be simultaneously connected to an application. The number of connections is irrelevant to the maximum number of users allowed by your applications or websites.

- For a cluster instance, the number of connections is the number of connections between the client and the mongos nodes.
- For a replica set instance, the number of connections is the number of connections between the client and the primary and secondary nodes.
- For a single-node instance, the number of connections is the number of connections between the client and the node.

11 Compatibility

11.1 Versions

DDS supports versions 3.4, 4.0, and 4.2. This section describes the compatibility between DDS and MongoDB Community Editions.

- DDS 3.4

  DDS 3.4 is fully compatible with MongoDB 3.4 Community Edition.
- DDS 4.0

  DDS 4.0 is fully compatible with MongoDB 4.0 Community Edition and partially compatible with MongoDB 4.4 Community Edition.
- DDS 4.2

  DDS 4.2 is fully compatible with MongoDB 4.2 Community Edition and partially compatible with MongoDB 4.4 Community Edition.

11.2 Details

This section describes the features and compatibility of DDS and MongoDB.

MongoDB Compatibility

Table 11-1 Compatible features

Category	Details	DDS 4.0	DDS 4.2	MongoDB 4.2 Community Edition	MongoDB 4.4 Community Edition
Database commands	The find command supports the **allowDiskUse** field, which allows MongoDB to use temporary files on disk to store data exceeding the system memory limit while processing a non-index sort operation.	×	×	×	√
Distributed transactions	Distributed transactions are supported. Distributed transactions refer to multi-document transactions on sharded clusters and replica sets. Multi-document transactions (whether on sharded clusters or replica sets) are also known as distributed transactions starting in MongoDB 4.2.	×	√	√	√
	The 16 MB total size limit for a transaction was removed.	×	×	√	√
	The MongoDB drivers must be updated for MongoDB 4.2.	×	√	√	√

Category	Details	DDS 4.0	DDS 4.2	MongoDB 4.2 Community Edition	MongoDB 4.4 Community Edition
Aggregation	• The aggregation capability is improved. Data can be updated in the aggregation pipeline. • Added $merge. • Added $accumulator and $function. • Supported union All. • Added $unionWith. • For details about aggregation operations, see Table 11-2.	√	√	√	√
Wildcard indexes	Wildcard indexes are supported.	√	√	√	√
Sharded cluster	A document's shard key field value can be changed.	×	×	√	√
	The fields associated with the shard key can be adjusted.	√	√	×	√
	Collections can be sharded using a composite shard key that consists of a single hash field.	×	×	×	√
	The 512-byte size limit on a shard key was removed.	×	×	×	√
	Multi-document transactions allow the creation of indexes and collections (non-distributed transactions).	×	×	×	√

Category	Details	DDS 4.0	DDS 4.2	MongoDB 4.2 Community Edition	MongoDB 4.4 Community Edition
Indexes	The size limit on indexes was removed.	√	√	√	√
	The length limit on index names was removed.	×	×	√	√
	Composite indexes can contain hash indexes.	×	×	×	√
	Hidden indexes are supported for commissioning.	√	√	×	√
	The size limit on collection namespaces is changed. The length of <database>.<collection> is increased from 120 bytes to 255 bytes.	×	×	×	√

NOTE

√ indicates that an item is supported, and × indicates that an item is not supported.

Aggregation Operations

Table 11-2 Aggregation operations

Operator	DDS 4.0	DDS 4.2	MongoDB 4.2 Community Edition	MongoDB 4.4 Community Edition
$sin	√	√	√	√
$cos	√	√	√	√
$tan	√	√	√	√
$asin	√	√	√	√
$acos	√	√	√	√
$atan	√	√	√	√

Operator	DDS 4.0	DDS 4.2	MongoDB 4.2 Community Edition	MongoDB 4.4 Community Edition
$atan2	√	√	√	√
$asinh	√	√	√	√
$acosh	√	√	√	√
$atanh	√	√	√	√
$degreesToRadians	√	√	√	√
$radiansToDegrees	√	√	√	√
$round	√	√	√	√
$trunc	√	√	√	√
$regexFind	√	√	√	√
$regexFindAll	√	√	√	√
$regexMatch	√	√	√	√
$merge	√	√	√	√
$planCacheStats	×	×	√	√
$replaceWith	√	√	√	√
$set	√	√	√	√
$unset	√	√	√	√
NOW	√	√	√	√
CLUSTER_TIME	√	√	√	√
findAndModify	√	√	×	√
update	√	√	√	√
$accumulator	√	√	×	√
$binarySize	√	√	×	√
$bsonSize	√	√	×	√
$first	√	√	×	√
$function	√	√	×	√
$last	√	√	×	√

Operator	DDS 4.0	DDS 4.2	MongoDB 4.2 Community Edition	MongoDB 4.4 Community Edition
$isNumber	√	√	×	√
$replaceOne	√	√	×	√
$unionWith	√	√	×	√

NOTE

√ indicates that an item is supported, and × indicates that an item is not supported.

12 Permissions Management

If you need to assign different permissions to employees in your enterprise to access your DDS resources, IAM is a good choice for fine-grained permissions management. IAM provides identity authentication, permissions management, and access control, helping you secure access to your Huawei Cloud resources.

With IAM, you can use your Huawei Cloud account to create IAM users for your employees, and assign permissions to the users to control their access to specific resource types. For example, some software developers in your enterprise need to use DDS resources but must not delete them or perform any high-risk operations. To achieve this result, you can create IAM users for the software developers and grant them only the permissions required for using DDS resources.

If your Huawei Cloud account does not need individual IAM users for permissions management, you may skip over this topic.

IAM can be used free of charge. You pay only for the resources in your account.

DDS Permissions

By default, new IAM users do not have permissions assigned. You need to add a user to one or more groups, and attach permissions policies or roles to these groups. Users inherit permissions from the groups to which they are added and can perform specified operations on cloud services based on the permissions.

DDS is a project-level service deployed and accessed in specific physical regions. To assign GaussDB NoSQL permissions to a user group, specify the scope as region-specific projects and select projects for the permissions to take effect. If **All projects** is selected, the permissions will take effect for the user group in all region-specific projects. When accessing DDS, users need to switch to a region where they have been authorized to use DDS.

You can grant users permissions by using roles and policies.

- Roles: A type of coarse-grained authorization mechanism that defines permissions related to user responsibilities. This mechanism provides only a limited number of service-level roles for authorization. When using roles to grant permissions, you need to also assign other roles on which the permissions depend to take effect. However, roles are not an ideal choice for fine-grained authorization and secure access control.

- Policies: A type of fine-grained authorization mechanism that defines permissions required to perform operations on specific cloud resources under certain conditions. This mechanism allows for more flexible policy-based authorization, meeting requirements for secure access control. For example, you can grant DDS users only the permissions for managing a certain type of resources.

Table 12-1 lists all the system-defined roles and policies supported by DDS.

Table 12-1 System-defined roles and policies supported by DDS

Policy Name/ System Role	Description	Type	Dependency
DDS FullAccess	All operations on DDS.	System-defined policy	None
DDS ReadOnlyAccess	Read-only permissions for DDS resources. Users granted these permissions can only view DDS data.	System-defined policy	None
DDS ManageAccess	DBA permissions on DDS except the DDS DB instance deletion operations.	System-defined policy	None
DDS Administrator	DDS administrator, who has all permissions of the service.	System-defined role	The Tenant Guest and Tenant Administrator roles need to be assigned in the same project.

Table 12-2 lists the common operations supported by each system-defined policy or role of DDS. Select the policies or roles as required.

Table 12-2 Common operations supported by the DDS system policy

Operation	DDS FullAccess	DDS ReadOnlyAccess	DDS ManageAccess	DDS Administrator
Creating an instance	√	x	√	√
Querying DB instances	√	√	√	√
Deleting a DB instance	√	x	x	√

Operation	DDS FullAccess	DDS ReadOnlyAccess	DDS ManageAccess	DDS Administrator
Restarting a DB instance	√	x	√	√
Performing a primary/ secondary switchover	√	x	√	√
Modifying the port	√	x	√	√
Resetting a password	√	x	√	√
Modified SSL	√	x	√	√
Modifying a security group	√	x	√	√
Binding and unbinding an EIP	√	x	√	√
Scaling up storage space	√	x	√	√
Changing DB instance classes	√	x	√	√
Adding nodes	√	x	√	√
Deleting the node that fails to be added	√	x	×	√
Modifying an automated backup policy	√	x	√	√
Renaming a DB instance	√	x	√	√
Changing a private IP address	√	x	√	√

Operation	DDS FullAccess	DDS ReadOnlyAccess	DDS ManageAccess	DDS Administrator
Changing the parameter template associated with the node in a DB instance	√	x	√	√
Showing original slow logs	√	x	√	√
Enabling or disabling local log auditing	√	x	√	√
Downloading audit logs	√	x	√	√
Deleting audit logs	√	x	×	√
Downloading a backup file	√	x	√	√
Changing the billing mode from pay-per-use to yearly/monthly	√	x	√	√
Creating a manual backup	√	x	√	√
Querying the backup list	√	√	√	√
Restoring data to a new DB instance	√	x	√	√
Restoring to an existing DB instance	√	x	√	√
Deleting a backup	√	x	×	√
Creating a parameter template	√	x	√	√

Operation	DDS FullAccess	DDS ReadOnlyAccess	DDS ManageAccess	DDS Administrator
Viewing parameter templates	√	√	√	√
Modifying a parameter template	√	x	√	√
Deleting a parameter template	√	x	×	√
Task Center list	√	x	√	√

Table 12-3 lists common DDS operations and corresponding actions. You can refer to this table to customize permission policies.

Table 12-3 Common Operations and actions

Operation	Actions	Authorization Scope	Description
Instance creation page	• vpc:vpcs:list • vpc:subnets:get • vpc:securityGroups:get	Supported: • IAM projects • Enterprise projects	The VPC, subnet, and security group are displayed on the instance creation page.
Creating an instance	• dds:instance:create • vpc:vpcs:list • vpc:vpcs:get • vpc:subnets:get • vpc:securityGroups:get • vpc:ports:get	Supported: • IAM projects • Enterprise projects	If the default VPC, subnet, and security group are used, the vpc:*:create permission must be configured. To create an encrypted instance, configure the KMS Administrator permission for the project.
Querying DB instances	dds:instance:list	Supported: • IAM projects • Enterprise projects	-

Operation	Actions	Authorization Scope	Description
Querying details of a DB instance	dds:instance:list	Supported: • IAM projects • Enterprise projects	If the VPC, subnet, and security group need to be displayed on the instance details page, add the vpc:*:get and vpc:*:list actions.
Exporting DB instance lists	dds:instance:list	Supported: • IAM projects • Enterprise projects	If the VPC, subnet, and security group are required, add the vpc:*:get and vpc:*:list actions.
Deleting a DB instance	dds:instance:deleteInstance	Supported: • IAM projects • Enterprise projects	When deleting a DB instance, delete the IP address on the data side.
Restarting a DB instance	dds:instance:reboot	Supported: • IAM projects • Enterprise projects	-
Performing a primary/ secondary switchover	dds:instance:switchover	Supported: • IAM projects • Enterprise projects	-
Changing a port	dds:instance:modifyPort	Supported: • IAM projects • Enterprise projects	-
Resetting a password	dds:instance:resetPasswd	Supported: • IAM projects • Enterprise projects	-
Modified SSL	dds:instance:modifySSL	Supported: • IAM projects • Enterprise projects	-
Modifying a security group	dds:instance:modifySecurityGroup	Supported: • IAM projects • Enterprise projects	-

Operation	Actions	Authorization Scope	Description
Binding an EIP	dds:instance:bindPublicIp	Supported: • IAM projects	When binding an EIP, you need to query the created EIP. • Enterprise projects are not supported. • Fine-grained authentication is not supported.
Unbinding an EIP	dds:instance:unbindPublicIp	Supported: • IAM projects	• Enterprise projects are not supported. • Fine-grained authentication is not supported.
Scaling up storage space	dds:instance:extendVolume	Supported: • IAM projects • Enterprise projects	-
Changing DB instance classes	dds:instance:modifySpec	Supported: • IAM projects • Enterprise projects	-
Adding nodes	• dds:instance:extendNode • vpc:vpcs:list • vpc:vpcs:get • vpc:subnets:get • vpc:securityGroups:get • vpc:ports:get	Supported: • IAM projects • Enterprise projects	-
Deleting the node that fails to be added	dds:instance:extendNode	Supported: • IAM projects • Enterprise projects	If the IP address has been created but the subsequent procedure fails, delete the IP address on the data side.
Modifying an automated backup policy	dds:instance:modifyBackupPolicy	Supported: • IAM projects • Enterprise projects	-

Operation	Actions	Authorization Scope	Description
Renaming a DB instance	dds:instance:modify	Supported: • IAM projects • Enterprise projects	-
Changing a private IP address	• dds:instance:modifyVIP • vpc:subnets:get • vpc:ports:get	Supported: • IAM projects • Enterprise projects	Before changing the private IP address, query available IP addresses.
Changing the parameter template associated with the node in a DB instance	dds:instance:modifyParameter	Supported: • IAM projects • Enterprise projects	-
Showing original slow logs	dds:instance:modifySlowLogPlaintextSwitch	Supported: • IAM projects • Enterprise projects	-
Enabling or disabling local log auditing	dds:instances:modifyAuditLogSwitch	Supported: • IAM projects • Enterprise projects	-
Downloading audit logs	dds:instances:downloadAuditLog	Supported: • IAM projects • Enterprise projects	-
Deleting audit logs	dds:instance:deleteAuditLog	Supported: • IAM projects • Enterprise projects	-
Downloading a backup file	dds:backup:download	Supported: • IAM projects • Enterprise projects	-

Operation	Actions	Authorization Scope	Description
Changing the billing mode from pay-per-use to yearly/ monthly	dds:instances:renew	Supported: • IAM projects • Enterprise projects	-
Creating a Manual Backup	dds:instance:createManualBackup	Supported: • IAM projects • Enterprise projects	-
Querying the backup list	dds:backup:list	Supported: • IAM projects • Enterprise projects	-
Restoring data to a new DB instance	• dds:backup:createInstanceFromBackup • vpc:vpcs:list • vpc:vpcs:get • vpc:subnets:get • vpc:securityGroups:get • vpc:ports:get	Supported: • IAM projects • Enterprise projects	The KMS Administrator permission needs to be configured for the encrypted instance in the project.
Restoring to an existing DB instance	dds:backup:refreshInstanceFromBackup	Supported: • IAM projects • Enterprise projects	-
Deleting a backup	dds:backup:delete	Supported: • IAM projects • Enterprise projects	-
Creating a parameter template	dds:param:create	Supported: • IAM projects • Enterprise projects	-

Operation	Actions	Authorization Scope	Description
Viewing parameter templates	dds:param:list	Supported: • IAM projects • Enterprise projects	-
Modifying a parameter template	dds:param:modify	Supported: • IAM projects • Enterprise projects	-
Deleting a parameter template	dds:param:delete	Supported: • IAM projects • Enterprise projects	-
Task Center list	dds:task:list	Supported: • IAM projects • Enterprise projects	-

13 Related Services

The following figure shows the relationship between DDS and other services.

Figure 13-1 Related services

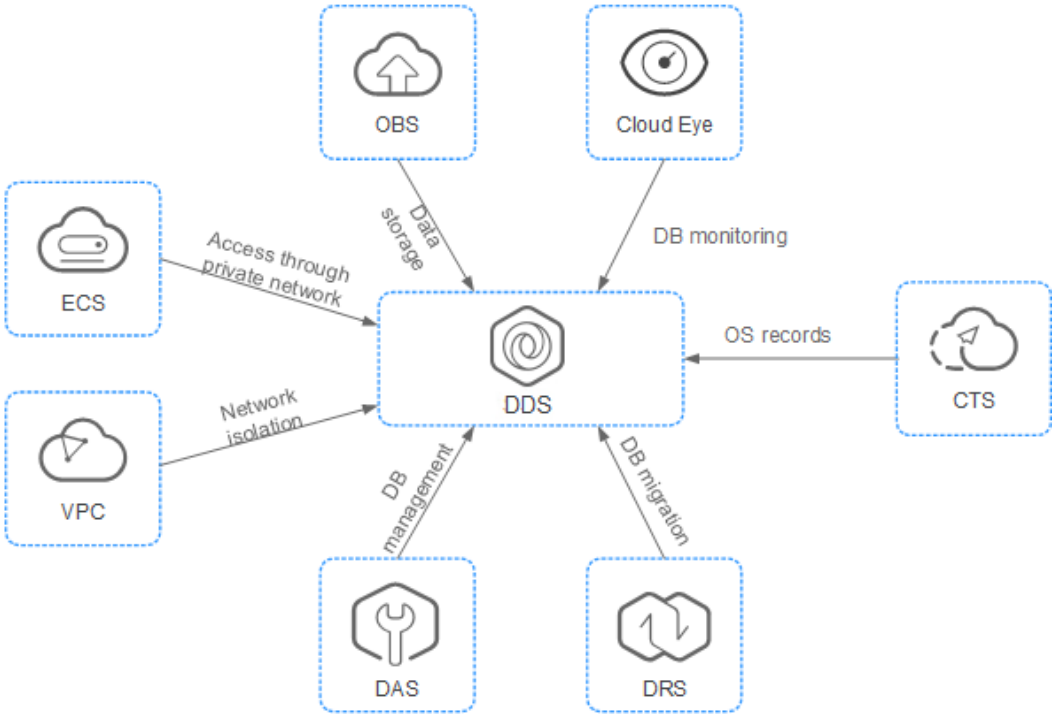


Table 13-1 Related services

Service Name	Function
Elastic Cloud Service (ECS)	ECS provides DDS with elastic computing resources and a running environment for DB instances.
Virtual Private Cloud (VPC)	VPC provides DDS with elastic network resources and implements network isolation and access control for your DB instances.

Service Name	Function
Object Storage Service (OBS)	OBS stores your DDS instance backup files.
Cloud Eye	Cloud Eye serves as a monitoring platform, monitoring DDS resources for you in real time. It reports alarms and issues warnings promptly to ensure that services are running properly.
Cloud Trace Service (CTS)	CTS records operations related to DDS, facilitating your further queries, audits, and retrievals.
Data Replication Service (DRS)	DRS smoothly migrates databases to the cloud. Source databases remain operational during migration, minimizing downtime and impact.

A Change History (Introduction) Europe Site

Released On	Description
2022-09-30	This issue is the first official release.

