ALIBABA CLOUD

Alibaba Cloud

ApsaraDB for MongoDB
Product Introduction

Document Version: 20201026

Alibaba Cloud

Legal disclaimer

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

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.

2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.

3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.

4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please directly contact Alibaba Cloud for any errors of this document.

Document conventions

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

Table of Contents

1.What is ApsaraDB for MongoDB?

ApsaraDB for MongoDB is developed based on the distributed system and high-reliability storage engine of Apsara, and is compatible with the MongoDB protocol. ApsaraDB for MongoDB uses a multi-node architecture to ensure high availability and supports elastic scaling, disaster recovery, backup and restoration, and performance optimization.

ApsaraDB, ApsaraDB for MongoDB

Why ApsaraDB for MongoDB?

For more information about the benefits of ApsaraDB for MongoDB, see Comparison between ApsaraDB for MongoDB and user-created databases and Scenarios.

Supported architectures

ApsaraDB for MongoDB supports multiple deployment architectures to meet the requirements of different business scenarios.

Architecture	Description
Architecture of standalone instances	You can use a standalone instance to perform development and testing or store enterprise data that is not critical. You can purchase a standalone instance at a lower price and enjoy the superior O&M support and kernel-level optimization that is provided by ApsaraDB for MongoDB. For more information about how to create a standalone instance, see Create a standalone instance.
Architecture of replica set instances	A replica set instance consists of a primary node that supports read/write operations, one or more high-availability secondary nodes, and a hidden node. For more information about the nodes and their differences, see Architecture of replica set instances. For more information about how to create a replica set instance, see Create a replica set instance. You can add or remove secondary nodes based on your business needs. For example, assume that you are running an informational website or an order query system that has more read operations than write operations. To improve the read performance, you can add secondary nodes. You can also add secondary nodes during business spikes and remove unnecessary nodes to save costs after the spikes. For more information, see Change the number of nodes for a replica set instance.
Architecture of sharded cluster instances	A sharded cluster instance is created based on multiple three-node replica sets. A sharded cluster instance consists of three components, which are mongos, shard, and config server. You can specify the number and configurations of mongos and shard nodes to create ApsaraDB for MongoDB clusters that have different service capabilities. For more information about how to create a sharded cluster instance, see Create a sharded cluster instance. For more information about different components, see Architecture of sharded cluster instances.

Pricing

For more information, see Billing items and pricing.

Deployment suggestions

Consider the following aspects when you create and use an ApsaraDB for MongoDB instance:

- Region and zone

  A region is an Alibaba Cloud data center. A zone is a physical area with independent power grids and networks in a region. The region and zone determine the physical location of an ApsaraDB for MongoDB instance. You cannot change the region of an ApsaraDB for MongoDB instance after the instance is created. For more information, see Regions and zones.

  Select a region and zone based on the geographical location, availability of Alibaba Cloud services, application availability requirements, and whether internal network communication is required. For example, if your application is deployed on an Elastic Compute Service (ECS) instance and requires an ApsaraDB for MongoDB instance to function as its database, you must select the same region and zone as the ECS instance when you create your ApsaraDB for MongoDB instance.

  > Note An ECS instance and an ApsaraDB for MongoDB instance in the same zone can be interconnected by using the internal network with minimal network latency.

- Network planning

  We recommend that you use Virtual Private Cloud (VPC) to connect to ApsaraDB for MongoDB instances. You can plan your own Classless Inter-Domain Routing (CIDR) blocks for VPCs. A VPC is an isolated network with higher security and performance than the classic network. You can use the default VPC or create your own VPC before you use ApsaraDB for MongoDB. For more information, see Configure a VPC for a new instance.

- Security solutions

  ApsaraDB for MongoDB eliminates your data security concerns by providing comprehensive security protection. You can ensure database security by using zone-disaster recovery, Resource Access Management (RAM) authorization, audit logs, network isolation, whitelists, password authentication, and Transparent Data Encryption (TDE). For more information, see Best practices for data security of ApsaraDB for MongoDB.

Manage ApsaraDB for MongoDB instances

You can use the following methods to manage ApsaraDB for MongoDB instances, such as creating instances, databases, and accounts, and specifying network-related settings:

- Console: The ApsaraDB for MongoDB console provides a graphical and easy-to-use web interface.
- API: All operations that are available in the console can be performed by calling API operations.

After you create an ApsaraDB for MongoDB instance, you can connect to the instance by using one of the following methods:

- Mongo shell: You can log on to the instance by using the official MongoDB CLI tool to manage databases. For more information, see Connect to a replica set instance by using the mongo shell.
- Client: ApsaraDB for MongoDB is compatible with the MongoDB protocol. You can use common database client tools, such as Robo 3T and Studio 3T to connect to ApsaraDB for MongoDB instances.

Related services

- ECS: The best performance is achieved when you connect to ApsaraDB for MongoDB instances

from ECS in the same region over an internal network. ECS and ApsaraDB for MongoDB instances compose a typical business architecture.

- Data Transmission Service (DTS): You can use DTS to migrate data from an on-premises MongoDB database to the cloud.
- Object Storage Service (OSS): OSS is a secure, cost-effective, and reliable cloud storage service that is provided by Alibaba Cloud. OSS allows you to store a large amount of data in the cloud.

2.System architecture

2.1. Architecture of ApsaraDB for MongoDB

This topic describes the architecture and components of ApsaraDB for MongoDB.

Architecture

Components

- Task control system

  ApsaraDB for MongoDB instances support various tasks, such as instance creation, configuration change, and instance backup. You can use the system to control tasks, track tasks, and manage errors.

- HA control system

  The system acts as a high-availability detection module to detect the running status of ApsaraDB for MongoDB instances. If the system determines that the primary node of an ApsaraDB for MongoDB instance is unavailable, the system fails over to a secondary node to maintain the availability of the instance. You can also manually switch over between the primary and secondary nodes. For more information, see Switch node roles.

- Log collection system

  The system collects the running logs of ApsaraDB for MongoDB, such as slow query and audit logs. For more information about the benefits of ApsaraDB for MongoDB, see Overview and Enable the new audit log feature.

- Monitoring system

  The system monitors the performance of ApsaraDB for MongoDB instances and collects information such as their basic metrics, disk capacities, access requests, and input/output operations per second (IOPS). For more information, see View monitoring information.

- Backup system

  The system backs up ApsaraDB for MongoDB instances and stores the generated backup files in Object Storage Service (OSS). It allows you to customize the backup policy to manually or automatically back up the instances. It retains the backup files for up to seven days. For more information about the benefits of ApsaraDB for MongoDB, see Configure automatic backup for an ApsaraDB for MongoDB instance and Manually back up an ApsaraDB for MongoDB instance.

- Online migration system

  If the physical server where the instance resides fails, the system creates another instance from the backup files in the backup system to prevent impacts on your business. For more information, see Overview.

2.2. Architecture of standalone instances

Standalone instances of ApsaraDB for MongoDB are developed for scenarios with high fault tolerance and apply to the storage of non-core data. This architecture is highly cost-effective and is ideal for environment testing, learning and training, and internal enterprise business.

It enables you to purchase ApsaraDB for MongoDB at a lower entry-level price to enjoy superiority in O&M support and kernel-level optimization. The standalone architecture can adapt ApsaraDB for MongoDB to various scenarios to help enterprises minimize their costs and expenses.

FAQ

Is high availability supported in the standalone instance architecture?

High availability is not supported in the standalone instance architecture because this architecture contains only one replica. When this architecture is faulty, business may be interrupted for more than 30 minutes in extreme cases. We recommend that you use the replica set instance architecture in your production environment.

Related information

- Architecture of replica set instances
- Architecture of sharded cluster instances

2.3. Architecture of replica set instances

ApsaraDB for MongoDB automatically configures replica sets for instances so that you can manage the primary and secondary nodes. Advanced functions such as disaster recovery and failover are available after instances are created. When you use the instances, these functions are triggered without you knowing about it.

Architecture

ApsaraDB for MongoDB uses a multi-node architecture to ensure high availability. A replica set instance consists of a primary node that supports read/write operations, one or more secondary nodes, and a hidden node. The following section describes the nodes of a replica set:

- Primary node: All read and write operations are performed on the primary node. Each replica set instance contains only one primary node.
- Secondary node: The data of a secondary node is synchronized with the primary node by using oplogs. If the primary node fails, a secondary node can be elected as the new primary node to ensure high availability.

  > **Note** When you connect to an instance by using the endpoint of a secondary node, you can only read data from the instance but cannot write data to it.

- Hidden node: The data of a hidden node is synchronized with the primary node by using oplogs. If a secondary node fails, the hidden node can be elected as the new secondary node to ensure high availability.

  > **Note** The hidden node is used only to ensure high availability and is not visible to users.

Scale out nodes in a replica set instance

ApsaraDB for MongoDB allows you to scale out the number of nodes in an instance. You can increase the number of secondary nodes based on your business needs. For more information, see Change the number of nodes for a replica set instance.

For example, assume that you are running an informational website or an order query system that has more read operations than write operations. To improve the read performance, you can add secondary nodes. You can also add secondary nodes during business spikes and remove unnecessary nodes to save costs after the spikes.

Related information

- Architecture of standalone instances
- Architecture of sharded cluster instances

2.4. Architecture of sharded cluster instances

A sharded cluster instance consists of three components: mongos, shard, and config server. You can choose the configuration and number of mongos and shards to create ApsaraDB for MongoDB sharded cluster instances that have different performance.

Architecture

Components

Component	Architecture	Description
Mongos	Standalone architecture	Routes queries and writes to the corresponding shards of sharded cluster instances. You can purchase multiple mongos in the console to achieve load balancing and failover. A single sharded cluster instance supports 2 to 32 mongos.
Shard	Replica set architecture (three nodes)	Stores database data. You can purchase multiple shards in the console to scale out the capacity of data storage and concurrent read/write operations. A single sharded cluster instance supports 2 to 32 shards.
Config server	Replica set architecture (three nodes)	Stores the metadata for clusters and shards. The metadata contains the data distribution information about each shard in a cluster. **Note** You cannot change the specifications of the config server (one core, 2 GB memory, and 20 GB disk storage capacity).

Related information

- Architecture of standalone instances
- Architecture of replica set instances

3.Features

ApsaraDB for MongoDB is developed based on the distributed system and high-reliability storage engine of Apsara, and is compatible with the MongoDB protocol. ApsaraDB for MongoDB uses a multi-node architecture to ensure high availability, and supports elastic scaling, disaster recovery, backup and restoration, and performance optimization. This topic describes the features of ApsaraDB for MongoDB.

Flexible architecture

ApsaraDB for MongoDB supports flexible deployment architectures, such as Architecture of standalone instances, Architecture of replica set instances, and Architecture of sharded cluster instances to meet requirements of different business scenarios.

Auto scaling

You can change the configuration of an ApsaraDB for MongoDB instance, including its specifications, storage space, and number of nodes, based on your business needs. You can also set the effective time for the configuration change. We recommend that you set the effective time to off-peak hours to avoid impacts on your business. For more information, see Configuration change overview.

Data security

Security technology	Description
Anti-DDoS	ApsaraDB for MongoDB monitors inbound traffic in real time, filters source IP addresses to scrub large amounts of malicious traffic, and triggers blackhole filtering when traffic scrubbing becomes ineffective.
IP address whitelist	ApsaraDB for MongoDB filters IP addresses for instance access to implement high-level security protection. You can configure up to 1,000 IP addresses in a whitelist. For more information, see 设置白名单及安全组.
VPC	A VPC is an isolated virtual network with higher security and better performance than the classic network. You must create a VPC in advance. For more information, see Create a VPC.
Disaster recovery	ApsaraDB for MongoDB provides zone-disaster recovery to further meet high-reliability and data security requirements. When you create an instance, you can deploy it in multiple zones. For more information, see Create a multi-zone replica set instance or Create a multi-zone sharded cluster instance. You can also migrate a replica set instance to multiple zones. For more information, see Migrate an ApsaraDB for MongoDB instance across zones in the same region.
SSL encryption	ApsaraDB for MongoDB encrypts network connections in compliance with SSL at the transport layer to improve data security and ensure data integrity during communication. For more information, see Configure SSL encryption for an ApsaraDB for MongoDB instance.

Security technology	Description
TDE	ApsaraDB for MongoDB performs real-time I/O encryption and decryption on data files. Data is encrypted before it is written to a disk, and decrypted when it is read from the disk and written into the memory. TDE does not increase the size of data files. You can use TDE without modifying your application that uses ApsaraDB for MongoDB. For more information, see Configure TDE for an ApsaraDB for MongoDB instance.
Automatic backup	You can set a backup policy to configure the start time for data backup during off-peak hours. For more information about how to set a backup policy, see Configure automatic backup for an ApsaraDB for MongoDB instance.
Temporary backup	You can manually back up ApsaraDB for MongoDB data. For more information, see Manually back up an ApsaraDB for MongoDB instance. Both physical backup and logical backup are supported.
Data restoration	You can create an instance based on a backup file, create an instance based on the backup file that is generated at a specific time point, and restore backup data to the current instance. For more information, see Create an instance from a backup, Restore data to a new ApsaraDB for MongoDB instance by point in time, and Restore data to the current ApsaraDB for MongoDB instance.
Backup file download	ApsaraDB for MongoDB retains your backup files free of charge for up to seven days. During this period, you can log on to the ApsaraDB for MongoDB console and download the backup files to a local device.

Comprehensive monitoring

ApsaraDB for MongoDB monitors up to 20 system performance metrics, such as disk capacity, IOPS, connections, CPU utilization, network traffic, transactions per second (TPS), queries per second (QPS), and cache hit ratio. For more information, see View monitoring information.

Professional tools

Data Management (DMS) allows you to manage relational databases such as MySQL, SQL Server, and PostgreSQL, non-relational databases such as MongoDB and Redis, and Linux servers. DMS offers an integrated solution to view BI charts and data trends, track data, optimize performance, implement access control, and manage data, schemas, and servers.

Data Transmission Service (DTS) is a data service that is provided by Alibaba Cloud to support data exchange between data sources such as relational database management system (RDBMS), NoSQL, and online analytical processing (OLAP). It provides data transmission capabilities such as data migration, real-time data subscription, and real-time data synchronization. DTS applies to scenarios such as data migration without server downtime, geo-disaster recovery, cross-border data synchronization, and cache refresh policies. It helps you build a secure, scalable, and highly available data architecture.

4.Comparison between ApsaraDB for MongoDB and user-created databases

Compared with user-created databases, ApsaraDB for MongoDB has outstanding advantages in terms of service availability, data reliability, security, and O&M cost. ApsaraDB for MongoDB helps you quickly start your business and reduce O&M costs.

Item	ApsaraDB for MongoDB	User-created database
Service availability	High availability.	You must ensure security and build primary/secondary replication and RAID.
	High availability disaster recovery in one or three zones in the same city.	Local disaster recovery requires self-deployment and maintenance. The dual-zone conditions are difficult to implement, and database-level availability is not guaranteed.
	Allows you to create geo-disaster recovery instances.	You must use a third-party tool to create disaster recovery instances.
Data reliability	• High reliability. • The recovery point objective (RPO) of single- and three-zone replica set instances is 0.	You must ensure security and build primary/secondary replication and RAID.
System security	Pre-protection: DDoS attack protection, automatic repair of various database security vulnerabilities, whitelist-based access control, and VPC isolation.	Pre-protection: requires additional security hardware or software to fix security vulnerabilities at high costs.
	In-process protection: SSL encryption and Transparent data encryption.	In-process protection: You must build SSL encryption and TDE systems.
	Event auditing: Database log auditing.	Event auditing: You must purchase an additional auditing system.
Backup and recovery	• Complete kernel enables both physical backup and logical backup to be performed in manual backup and increases backup efficiency three times. • Single-database recovery.	• The open-source version supports only logical backup and is inefficient. • Single-database recovery is unavailable and recovery efficiency is low. • You must ensure the accuracy of data recovery in the distributed architecture.
System hosting	No hosting fee.	You need to pay additional server hosting fees. The more complex the architecture, the more servers you need to host and the higher the hosting fees.

Item	ApsaraDB for MongoDB	User-created database
O&M cost	No additional O&M costs.	Requires dedicated maintenance personnel and high O&M costs.
	Performance monitoring in seconds and CloudDBA.	Conducts performance monitoring in minutes without any diagnosis. It is complex to troubleshoot slow queries.
Deployment and scaling	Supports real-time activation and elastic scaling.	Requires hardware procurement, hosting of data centers, and machine deployment, which takes rather a long period. You must maintain the node relationship when adding new nodes.
Kernel optimization	• Optimizes oplog synchronization performance and short-lived connection performance for general scenarios. • Ensures data consistency with technologies such as logical snapshots. This reduces costs by 50% in long-time data import scenarios.	The open-source version does not provide optimization and cannot be used in some scenarios.

5.Scenarios

ApsaraDB for MongoDB supports standalone, replica set and sharded cluster deployment architectures and provides enterprise-class capabilities such as security audit and point-in-time backup. It has been widely used in Internet, IoT, games, and finance fields.

Read/write splitting

ApsaraDB for MongoDB uses the architecture of three-node replica sets to guarantee high availability. Three data nodes are located on different physical servers and automatically synchronize data. The primary and secondary nodes are configured with different endpoints. MongoDB drivers allocate read/write requests to them. For more information, see Architecture of ApsaraDB for MongoDB.

Flexible business scenarios

ApsaraDB for MongoDB has no schema and is suitable for startup business needs. You do not need to worry about changing schemas. You can store structured data in ApsaraDB for Relational Database Service (RDS), flexible business data in ApsaraDB for MongoDB, and hot data in ApsaraDB for Redis or ApsaraDB for Memcache. This helps you write and read business data with high efficiency and reduce the cost of data storage.

Apps

ApsaraDB for MongoDB supports two-dimensional spatial indexes, so it can provide support for location-based apps. Its dynamic storage mode is also suitable for storing heterogeneous data from multiple systems. This satisfies the requirements of apps.

IoT scenarios

ApsaraDB for MongoDB features high performance and asynchronous data writing. It can achieve the processing capability of an in-memory database in specific scenarios. In a sharded cluster instance of ApsaraDB for MongoDB, you can adjust the configuration and quantity of mongos and shards to improve performance and expand storage space without limits. ApsaraDB for MongoDB is suitable for IoT scenarios with highly concurrent write operations. For more information, see Configuration change overview.

ApsaraDB for MongoDB provides a secondary index feature for dynamic queries. It can use the MapReduce aggregation framework of MongoDB to conduct multidimensional data analysis.

6.Instance types

This topic describes the instance types available in ApsaraDB for MongoDB.

Current instance types

The following new instance types are used for instances that were created on and after July 10, 2017 or whose configurations have been changed.

Standalone or replica set instance types

Architecture	Category	Specification	Instance type	Maximum connections	Maximum IOPS	Storage capacity
Replica set instance	General-purpose	1 vCPU, 2 GiB memory	dds.mongo.mid	500	8,000	10-2,000 GB
		2 vCPUs, 4 GiB memory	dds.mongo.standard	1,000	8,000	
		4 vCPUs, 8 GiB memory	dds.mongo.large	3,000	8,000	
		8 vCPUs, 16 GiB memory	dds.mongo.xlarge	5,000	8,000	
		8 vCPUs, 32 GiB memory	dds.mongo.2xlarge	8,000	14,000	
		16 vCPUs, 32 GiB memory	dds.mongo.4xlarge	16,000	16,000	
	Dedicated-	2 vCPUs, 16 GiB memory	mongo.x8.medium	2,500	4,500	250 GB
		4 vCPUs, 32 GiB memory	mongo.x8.large	5,000	9,000	500 GB
		8 vCPUs, 64 GiB memory	mongo.x8.xlarge	10,000	18,000	1,000 GB

Architecture	CPU Category	Specification	Instance type	Maximum connections	Maximum IOPS	Storage capacity
		16 vCPUs, 128 GiB memory	mongo.x8.2xlarge	20,000	36,000	• Subscription: 2,000-3,000 GB • Pay-as-you-go: 2,000 GB
		32 vCPUs, 256 GiB memory	mongo.x8.4xlarge	40,000	72,000	2,000-3,000 GB
	Dedicated-host	60 vCPUs, 440 GiB memory	dds.mongo.2xmonopolize	100,000	100,000	2,000-6,000 GB
Standalone instance	General-purpose	1 vCPU, 2 GiB memory	dds.n2.small.1	2,000	min{30 × Storage capacity, 20,000}	20-2,000 GB
		2 vCPUs, 4 GiB memory	dds.sn2.medium.1	4,000		
		2 vCPUs, 8 GiB memory	dds.sn4.large.1	6,000		
		4 vCPUs, 8 GiB memory	dds.sn2.large.1	6,000		
		4 vCPUs, 16 GiB memory	dds.sn4.xlarge.1	8,000		
		8 vCPUs, 16 GiB memory	dds.sn2.xlarge.1	8,000		

Sharded cluster instance types

Node type	Instance category	Specification	Instance type	Maximum connections	Maximum IOPS	Storage capacity

Node type	Instance category	Specification	Instance type	Maximum connections	Maximum IOPS	Storage capacity
Mongos	General-purpose	1 vCPU, 2 GiB memory	dds.mongos.mid	1,000	-	-
		2 vCPUs, 4 GiB memory	dds.mongos.standard	2,000		
		4 vCPUs, 8 GiB memory	dds.mongos.large	4,000		
		8 vCPUs, 16 GiB memory	dds.mongos.xlarge	8,000		
		8 vCPUs, 16 GiB memory	dds.mongos.2xlarge	16,000		
		16 vCPUs, 64 GiB memory	dds.mongos.4xlarge	16,000		
Shard	General-purpose	1 vCPU, 2 GiB memory	dds.shard.mid		1,000	10-2000 GB
		2 vCPUs, 4 GiB memory	dds.shard.standard		2,000	
		4 vCPUs, 8 GiB memory	dds.shard.large		4,000	
		8 vCPUs, 16 GiB memory	dds.shard.xlarge		8,000	
		8 vCPUs, 32 GiB memory	dds.shard.2xlarge		14,000	
		16 vCPUs, 64 GiB memory	dds.shard.4xlarge		16,000	

Node type	Instance category	Specification	Instance type	- Maximum connections	Maximum IOPS	Storage capacity
	Dedicated-CPU	2 vCPUs, 16 GiB memory	dds.shard.sn8.xlarge.3		4,500	10-250 GB
		4 vCPUs, 32 GiB memory	dds.shard.sn8.2xlarge.3		9,000	10-500 GB
		8 vCPUs, 64 GiB memory	dds.shard.sn8.4xlarge.3		18,000	10-1,000 GB
		16 vCPUs, 128 GiB memory	dds.shard.sn8.8xlarge.3		36,000	10-2,000 GB
		32 vCPUs, 256 GiB memory	dds.shard.sn8.16xlarge.3		72,000	10-3,000 GB
Configserver	General-purpose	1 vCPU, 2 GiB memory	dds.cs.mid		1,000	20 GB

Historical instance types

The following instance types will be used for instances that were created before July 10, 2017 and whose configurations have not been changed.

Three-node replica set instance types

Category	Specification	Instance type	Maximum connections	Maximum IOPS
General-purpose	1 vCPU, 2 GiB memory	dds.mongo.mid	200	800
	2 vCPUs, 4 GiB memory	dds.mongo.standard	400	1,600
	4 vCPUs, 8 GiB memory	dds.mongo.large	1,000	3,200
	8 vCPUs, 16 GiB memory	dds.mongo.xlarge	2,000	6,400
	8 vCPUs, 32 GiB memory	dds.mongo.2xlarge	4,000	12,800

Category	Specification	Instance type	Maximum connections	Maximum IOPS
	16 vCPUs, 64 GiB memory	dds.mongo.4xlarge	8,000	12,800
Dedicated-CPU	2 vCPUs, 16 GiB memory	mongo.x8.medium	2,000	4,500
	4 vCPUs, 32 GiB memory	mongo.x8.large	4,000	9,000
	8 vCPUs, 64 GiB memory	mongo.x8.xlarge	8,000	18,000
	16 vCPUs, 128 GiB memory	mongo.x8.2xlarge	16,000	36,000
	32 vCPUs, 256 GiB memory	mongo.x8.4xlarge	32,000	72,000
Dedicated-host	60 vCPUs, 440 GiB memory	dds.mongo.2xmonopolize	36,000	40,000

Sharded cluster instance types

Node type	Instance category	Specification	Instance type	Maximum connections	Maximum IOPS
Mongos	General-purpose	1 vCPU, 2 GiB memory	dds.mongos.mid	200	-
		2 vCPUs, 4 GiB memory	dds.mongos.standard	400	
		4 vCPUs, 8 GiB memory	dds.mongos.large	1,000	
		8 vCPUs, 16 GiB memory	dds.mongos.xlarge	2,000	
		8 vCPUs, 32 GiB memory	dds.mongos.2xlarge	4,000	
		16 vCPUs, 64 GiB memory	dds.mongos.4xlarge	8,000	
		1 vCPU, 2 GiB memory	dds.shard.mid		800
		2 vCPUs, 4 GiB memory	dds.shard.standard		1,600

Node type	Instance category	Specification	Instance type	Maximum connections	Maximum IOPS
Shard	General-purpose	4 vCPUs, 8 GiB memory	dds.shard.large	-	3,200
		8 vCPUs, 16 GiB memory	dds.shard.xlarge		6,400
		8 vCPUs, 32 GiB memory	dds.shard.2xlarge		12,800
		16 vCPUs, 64 GiB memory	dds.shard.4xlarge		12,800
Configserver	General-purpose	1 vCPU, 2 GiB memory	dds.cs.mid		800

7.MongoDB versions and storage engines

ApsaraDB for MongoDB supports MongoDB 3.4, 4.0, and 4.2, as well as the WiredTiger storage engine. You can choose a storage engine and a MongoDB version that best suits your business needs when you create an instance.

MongoDB versions and advantages

Compared with MongoDB 3.2, MongoDB 3.4 provides higher performance and security. MongoDB 4.0 is more suitable for finance and scenarios that are dependent on transactions and use NoSQL features. MongoDB 4.2 uses the two-stage commit method to ensure the atomicity, consistency, isolation, durability (ACID) feature of sharded cluster transactions. This greatly expands its business scenarios. The following table describes the advantages of MongoDB versions.

> **Note** MongoDB 3.2 has been phased out. For more information, see Notice: ApsaraDB for MongoDB has phased MongoDB 3.2 out and released MongoDB 4.2 since February 4.

MongoDB version	Advantage

MongoDB version	Advantage
MongoDB 3.4	• Faster primary-secondary synchronization All indexes are created when data is synchronized (only the _id index is created in earlier versions). During data synchronization, the secondary node continuously reads new oplog information to ensure that the local database of the secondary node has enough space to store temporary data. • More efficient load balancing In earlier versions, Mongos nodes are responsible for load balancing of sharded cluster instances. Multiple Mongos nodes contest a distributed lock. The node that obtains the lock performs load balancing tasks and migrates chunks between shard nodes. In MongoDB 3.4, the primary Configserver node is responsible for load balancing. This greatly improves the concurrency and efficiency of load balancing. • More aggregation operations Many aggregation operators are added in MongoDB 3.4 to provide more powerful data analysis capabilities. For example, `bucket` can conveniently classify data. `$grahpLookup` supports more complex relational operations than `$lookup` in MongoDB 3.2. `$addFields` enables richer document operations such as summing some fields and saving them as a new field. • Sharding zones supported The zone concept is introduced for sharded cluster instances to replace the current tag-aware sharding mechanism. It can allocate data to one or more specified shard nodes. This feature allows you to conveniently deploy sharded cluster instances across data centers. • Collation supported In earlier versions, strings stored in documents are always compared byte by byte regardless of Chinese, English, uppercase, or lowercase. After collation is introduced, the string content can be interpreted or compared based on the used locale. Case-insensitive comparison is also supported. • Read-only views MongoDB 3.4 supports read-only views. It virtualizes the data that meets a certain query condition into a special collection, on which you can perform further queries.

MongoDB version	Advantage
MongoDB 4.0	• Cross-document transactions As the first NoSQL database that supports cross-document transactions, MongoDB 4.0 combines the speed, flexibility, features, and ACID guarantee of document models. • Migration speed increase by 40% Concurrent read and write operations enable new shard nodes to migrate data fast and bear service load. • Read performance significantly improved With the transaction feature, secondary nodes no longer block read requests due to log synchronization. The multi-node scaling feature is supported in all versions to significantly improve reading capabilities.
MongoDB 4.2	• Distributed transactions The two-phase commit method helps the ACID feature of sharded cluster transactions, expands business scenarios, and achieves a leap from NoSQL to NewSQL. • Repeatable reads The repeatable read feature provides the automatic retry capability in a poor-quality network environment. This reduces logic complexity at the service side and ensures continuity of your business. • Wildcard indexes You can create a wildcard index for nondeterministic fields to overwrite multiple feature fields in a document for flexible management and usage. • Field-level encryption Field-level encryption is supported at the driver layer and can be used to separately encrypt specified sensitive information such as accounts, passwords, prices, and mobile phone numbers. You can use Field-level encryption to improve business flexibility and security without full-database encryption. • Materialized views Latest materialized views can cache computing results to avoid repeated computing for improved operational efficiency and reduce logical complexity.

Note You can manually upgrade the MongoDB version while an instance is running. However, you cannot downgrade the MongoDB version. For more information, see Upgrade MongoDB versions.

Storage engines

Storage engine	Description	Scenario
WiredTiger	Data is in the B-tree structure. Compared with the earlier MMAPv1 storage engine, WiredTiger provides significantly improved performance and reduces storage costs through data compression.	It is the default storage engine and applicable to most business scenarios.

Relationship between MongoDB versions and storage engines

The following tables describe the relationship between MongoDB versions and storage engines.

Storage engine	MongoDB 3.4	MongoDB 4.0	MongoDB 4.2
WiredTiger	Standalone instance (pay-as-you-go) Replica set instance Sharded cluster instance	Standalone instance (pay-as-you-go) Replica set instance Sharded cluster instance	Replica set instance Sharded cluster instance

8.Glossary

Term	Description
Region	• The geographical location of the server for an ApsaraDB for MongoDB instance that you have purchased. You need to specify the region when purchasing the instance. For now, you cannot change the region after purchasing the instance. • ApsaraDB for MongoDB only supports intranet access. When purchasing an ApsaraDB for MongoDB instance, ensure that you have purchased an ECS instance in the same region. For more information about how to connect to an ApsaraDB for MongoDB instance through an intranet, see Connect to an ApsaraDB for MongoDB instance through a cross-zone intranet.
Zone	• The physical area with its power supply and network isolated from other counterparts in the same region. • A zone is insulated from faults in other zones, and provides network connectivity to other zones in the same region through an intranet. The network latency within a zone is lower that across zones. • If an ApsaraDB for MongoDB replica set instance is a single-zone instance, all three nodes of the instance are located in the same zone. If a pair of ApsaraDB for MongoDB and ECS instances are deployed in the same zone, the network latency can be lower.
Instance	• The ApsaraDB for MongoDB instance, which is the basic unit of the ApsaraDB for MongoDB service that you can purchase. • The instance is the operating environment for ApsaraDB for MongoDB and exists as a separate process on the host. • Users can use the console to create, modify, and delete MongoDB instances. Instances are mutually independent and configured with isolated resources. They do not need to compete for CPU, memory, and I/O resources. • Each instance has its own characteristics such as the database type and version. ApsaraDB for MongoDB provides parameters to control the behavior of each instance.
Memory	The maximum memory that can be used by an ApsaraDB for MongoDB instance.
Disk capacity	• Disk capacity is the size of the disk which the user selects when purchasing the MongoDB instance. • The disk capacity occupied by the instance includes set data and the space required for normal instance operation, such as the system database, database rollback log, redo log, and indexing. • You need to ensure that an ApsaraDB for MongoDB instance has sufficient disk capacity to store data, otherwise the instance may be locked. If an instance is locked due to insufficient disk capacity, you can purchase more disk capacity to unlock the instance.
IOPS	The maximum number of read and write operations performed by a block device per second, measured in units of 4 KB.

Term	Description
CPU core	This is the instance's maximum computing power. A CPU core has the minimum computing power at 2.3 GHz (equivalent to an Intel Xeon processor which adopts Hyper-Threading technology).
Connections	TCP connections between clients and the MongoDB instances. If a client uses a connection pool, the client establishes persistent connections with ApsaraDB for MongoDB instances. Otherwise, it establishes transient connections.
ApsaraDB for MongoDB cluster	The cluster version of ApsaraDB for MongoDB. You can purchase multiple mongos nodes, multiple shards, and a config server to create an ApsaraDB for MongoDB cluster conveniently, which serves as a MongoDB distributed database system.
Mongos	• The entry to an ApsaraDB for MongoDB cluster for requests. Mongos nodes act as a request distribution center to coordinate all requests. They are responsible for forwarding data requests to the corresponding shards. • You can configure multiple mongos nodes as the entry for requests. In this case, if a mongos node is faulty, others can still process requests.
Shard	• Shards are the parts of MongoDB clusters. • Each shard is deployed as a three-node replica set to guarantee its high availability. Based on your application performance and storage requirements, you can purchase multiple shards to scale out the read and write performance and storage space of ApsaraDB for MongoDB and deploy a distributed database system.
Config server	• The configuration server that stores all database metadata for mongos nodes and shards in an ApsaraDB for MongoDB cluster. A mongos node does not store data, but caches the shard data and data routing information in its memory. The config server actually stores such data. • When a mongos node is started for the first time or is shut down and then restarted, it loads configuration information from the config server. If the configuration information changes, the config server notifies all mongos nodes, so that they can update their status to correctly route data. • The config server stores the metadata of mongos nodes and shards. Considering high requirements for service availability and data reliability, ApsaraDB for MongoDB deploys the config server as a three-node replica set to comprehensively ensure its service reliability.

9.Instance type families

Families

ApsaraDB for MongoDB provides general and dedicated instance type families.

Family	Description	Scenario
General	• Instances that have exclusive use of allocated memory and I/O resources and share CPU and storage resources with other general instances on the same physical server. • It is a highly cost-effective instance type family which allows you to minimize costs by reusing resources and enjoy the benefits brought by the scale. • Its storage capacity is independent of CPU and memory, which allows flexible configurations.	• Price-sensitive customers. • Fewer requirements for performance and stability.
Dedicated	Instances that have exclusive use of CPU, memory, storage, and I/O resources. Instances of this type family feature high performance and stability and are independent of other instances running on the same physical server. The top configuration of this type family is the Exclusive physical machine (also called the DDH), which exclusively occupies all resources owned by a physical server.	A database is a core component of certain businesses, such as finance, e-commerce, government, and large and medium-sized online businesses.

The following figure shows the differences between general and dedicated instance type families.

Instance types

For more information about instance types and specifications such as the number of CPU cores, memory, storage space, maximum number of allowed connections, and IOPS, see Instance types.

Pricing

For the price of each instance type, see Billing items and pricing.

Change instance types

You can change instance types as needed. For more information about specific operations, see Configuration change overview.