

TencentDB for Redis

Best Practices

Product Documentation



Copyright Notice

©2013-2024 Tencent Cloud. All rights reserved.

Copyright in this document is exclusively owned by Tencent Cloud. You must not reproduce, modify, copy or distribute in any way, in whole or in part, the contents of this document without Tencent Cloud's the prior written consent.

Trademark Notice



All trademarks associated with Tencent Cloud and its services are owned by Tencent Cloud Computing (Beijing) Company Limited and its affiliated companies. Trademarks of third parties referred to in this document are owned by their respective proprietors.

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

Contents

Best Practices

AZ Migration Scheme

Hot Key and Big Key

Best Practices

AZ Migration Scheme

Last updated : 2023-10-20 11:01:13

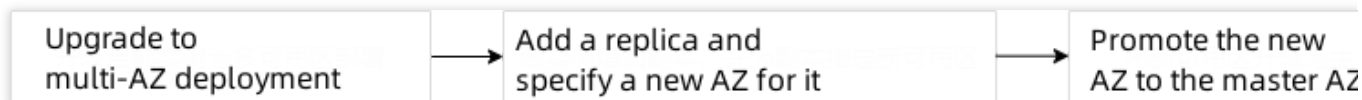
Overview

When adjusting database resources at the AZ level, you may consider deactivating all AZs with a low resource utilization. In this case, you need to migrate the data in these AZs to other AZs without affecting the normal operations of the current business.

Solution

For convenience considerations, we recommend that you migrate data with the following solutions.

If the current instance is deployed in one single AZ, and all resources in the AZ are to be deactivated, you can use the following data migration scheme:



If the current instance is deployed in multiple AZs, and its master or replica AZ is to be deactivated:

If the master AZ is to be deactivated, manually promote the replica AZ to master AZ, add a new replica for the instance and specify another AZ for it, and delete the replica node in the original master AZ.

If the replica AZ is to be deactivated, change the AZ of the replica as instructed in [Adding Replicas to Multi-AZ Deployed Instance](#).


Directions

Single-AZ deployment


If the AZ of the current instance is to be changed, data needs to be migrated as detailed below. We recommend that you perform the migration during the maintenance time. For more information, see [Setting Maintenance Time](#).

1. Upgrade the current single-AZ deployed instance to a multi-AZ deployed instance as instructed in [Upgrading to Multi-AZ Deployment](#).

After upgrading to multi-AZ deployment, you can see the

 icon next to the **AZ** of the instance in the **AZ** column in the instance list or in the **Basic Info** section on the **Instance Details** page.

Hover over

 and you can see that the AZ information of the master and replica nodes of the current instance has not changed.

2. Add a replica to the multi-AZ deployed instance and specify another AZ for it as instructed in [Adding Replicas to Multi-AZ Deployed Instance](#).

Wait for the instance configuration change to complete. Then, you can view the instance's AZ information, and you can see that two replicas have been added to the instance in Guangzhou Zones 4 and 3 respectively.

3. Promote the AZ of the new replica to master AZ.

3.1 Log in to the [TencentDB for Redis console](#).

3.2 In the instance list, find the target instance, and click **Instance ID**.

3.3 Enter the **Instance Details** page, select **Node Management** tab, and click **Promote Replica to Master** to set the AZ of replica node as the master AZ. For more information, see [Manually Promoting to Master Node \(Group\)](#).

4. Delete all nodes in the original master AZ to clear resources.

On the **Node Management** tab in the console, you can see that the original master AZ has been automatically switched to the replica AZ, and all nodes in it have been changed to replica nodes.

4.1 On the instance management page, click **More > Delete Replica**.

4.2 On the **TencentDB for Redis Configuration Changes** page, select replicas to be deleted for standard architecture and the replica group for cluster architecture, and click **OK**.

Multi-AZ deployment

If the master AZ is to be deactivated, data needs to be migrated as detailed below. We recommend that you perform the migration during the maintenance time. For more information, see [Setting Maintenance Time](#).

1. Manually promote a replica AZ to master AZ. For more information, see step3 in single-AZ deployment in [AZ Migration Scheme > Directions](#).

2. Add a new replica for the instance as instructed in [Adding Replicas to Multi-AZ Deployed Instance](#).

3. Delete the replica node in the original master AZ. For more information, see step4 in single-AZ deployment in [AZ Migration Scheme > Directions](#).

Hot Key and Big Key

Last updated : 2023-05-23 10:37:27

During daily business operations, your TencentDB for Redis instance may experience performance degradation, access timeouts, and poor user experience when you fail to handle big key or hot key issues in time. This may even cause a large-scale failure of instances. This document describes the causes of big keys and hot keys as well as relevant troubleshooting and optimization solutions.

Definition

Big key

A big key is one that has large value and takes up a large space. Essentially, this is a big value issue. The following are some common examples of data structure types in TencentDB for Redis.

For a value of string type, the value exceeding 10 MB is considered too large.

For a value of set type, the member number of 10,000 is considered too large.

For a value of list type, the member number of 10,000 is considered too large.

For a value in hash format, there are 1000 members yet the total value of all member variables is 1000 MB, then the total size of members is considered too large.

Hot key

A hot key is one that gets more accesses than other keys over a period of time and has high QPS in a specific TencentDB for Redis instance. It also refers to a key with high CPU or bandwidth utilization. Common examples are shown below.

When the total QPS (command executions per second) of the TencentDB for Redis instance is 10,000 and one of the keys has 7,000 accesses per second, it could be a hot key.

When a hash-formatted key containing 2000 fields sends a large number of **hgetall** operation requests per second, it could be a hot key.

When a key containing 10,000 fields sends a large number of **zrange** operation requests per second, it could be a hot key.

Symptoms and Impacts

Big key

Memory usage is uneven

In the TencentDB for Redis cluster architecture, the memory utilization of a certain data shard is far higher than that of other data shards, and the memory resources cannot be balanced. In addition, Redis memory may reach the upper limit defined by the `maxmemory` parameter, causing important keys to be evicted and even the memory to overflow.

Timeout blocking occurs when request response time rises

As Redis adopts a single-threaded architecture, it takes a long time to operate a big Key, which may cause request blocking.

Data sync has interrupted or master-replica switch is being performed.

When the memory is insufficient, the master database will be blocked for a long time if you evict a big key or **rename** it, which may cause sync interruption or master-replica switch.

Network is congested

A big key occupies 1 MB, and 1000 accesses per second will result in 1000 MB of traffic, which may cause the bandwidth of the instance or LAN to be fully occupied. This slows down its own services while also affecting other services.

Hot key

The CPU utilization of the instance stays high, compromising the overall service performance.

Due to the uneven distribution of requests under the cluster architecture and the increased access pressure on nodes with hot keys, the data shard may experience an exhaustion of connections or even go down. Even if expansion is performed in this case, there will be a great waste of resources.

The highly concentrated hotspot cache traffic surpasses the capacity of TencentDB for Redis, making it easy to cause the cache and database breakdown and thus triggering an avalanche of the system.

Cause Analysis

Big key

The key-value pair in TencentDB for Redis is improperly set, such as using a key of string type to store large-volume binary file-type data. This results in a particularly large key value.

For list and set structures, invalid data is not cleaned up in time, which causes the number of members in the key to continuously increase.

Before business launch, the business analysis is inaccurate, and the members in the key are not reasonably split, resulting in too many members in an individual key.

Hot key

Unexpected sudden increase in the number of accesses, such as the sudden appearance of hot products, hot news with soaring visits, a large number of screen likes brought by a certain anchor in the live broadcast room, and conflicts between multiple unions involving a large number of players in a specific area of the game.

Troubleshooting

TencentDB for Redis is connected to DBbrain's performance optimization feature, helping you quickly find big keys and hot keys in the database.

For directions on big key troubleshooting, see [Memory Analysis](#).

For directions on hot key troubleshooting, see [Hot Key Analysis](#).

Solutions

Big key

1. Clear invalid data

This mainly applies to the data of list and set types. Since such data will accumulate over time with the previous stored ones becoming invalid, you need to clear them regularly.

2. Compress the value of the corresponding big key

You can compress the value by serialization or compression to make it smaller, but if the corresponding value is still very large after compression, you need to use the split method to solve it.

3. Split big key

By splitting the big key into the key-value pairs of multiple small keys, and the corresponding value size and the number of split members are more reasonable after splitting, and then store it. You can use `get` or `mget` to obtain stored key-value pairs in batches.

4. Monitor Redis memory, bandwidth and key growth trends in real time

You can monitor the memory usage and network bandwidth usage in TencentDB for Redis through the monitoring system as well as the growth rate of memory usage within a fixed period of time. When the set threshold is exceeded, an alarm notification is triggered for troubleshooting. For specific information on monitoring metrics, see [Monitoring at Five-Second Granularity](#). For directions on setting alarm thresholds, see [Configuring Alarms](#).

Hot key

You can use the read/write separation architecture. If the generation of hot keys comes from read requests, then read/write separation is a good solution. When using the read/write separation architecture, the read request pressure in each TencentDB for Redis instance can be reduced by continuously increasing the number of replica nodes. For more information, see [Enabling/Disabling Read/Write Separation](#).

