TencentDB for Redis

Command Compatibility

Product Documentation

Contents

Command Compatibility Overview

Last updated：2023-01-30 10:43:08

Memory Edition (Cluster Architecture) stores data in a distributed manner, and its biggest difference from standard architecture lies in whether a single command supports multikey access. For the cluster architecture, commands can be categorized into partially supported, custom, and unsupported as detailed below:

Notes:

Currently, Redis 6.2 does not support the RESP3 protocol.

Command Type	Description
Unsupported commands	`(error) ERR unknown command 'keys'` will be returned for unsupported commands. For more information on the commands supported by different versions and architectures, see Overview.
Partially supported commands	Memory Edition (Cluster Architecture) is compatible with smart clients such as JedisCluster. For compatibility with JedisCluster, TencentDB for Redis modifies the IP list returned by the supported commands, and the IP address of each node in the returned information is the instance's VIP. For more information, see Use Case of Partially Supported Commands.
Supported cross-slot commands	Currently, cross-slot access commands supported by Memory Edition (Cluster Architecture) include MGET, MSET, and DEL but not other multikey commands.
Custom commands	Custom commands support the access of each node in a cluster. A new parameter **Node ID** is added to the right of the parameter list of the original command, including `INFO` , `MEMORY` , `SLOWLOG` , `FLUSHDB` , `PING` , and `KEYS` (with hashtag supported for preferred match). For more information, see Use Cases of Custom Commands.
Supported DMC commands	Database Management Center (DMC) allows you to log in to your TencentDB instances to access them, view their key metric information, and run Redis commands. For more information, see List of Supported DMC Commands.
Transactional commands	Memory Edition (Cluster Architecture) supports transactional commands provided that the transactions are started by the `WATCH` command. The keys of a transaction should be stored in the same slot, and the keys of `WATCH` and transaction-related keys should also be stored in the same slot. Hashtag is recommended for multikey transactions in cluster mode.
Multi-database commands	Memory Edition (Cluster Architecture) supports multiple databases (256 by default); therefore, it can support all commands related to database operations.

Commands Supported by Different Versions Overview

Last updated：2022-10-31 10:29:37

For supported commands by versions and architecture, see the following command groups. In the command group tables, ✓ indicates "supported", x indicates "unsupported", and - indicates that cross-slot access is not applicable to the command. For details about parameters and directions of the supported commands, see Redis commands.

Connection Group
Hash Group
Keys
list group
pub/sub group
sets group
sorted sets group
strings group
transactions group
hyperloglog group
scripting group
geo group
server group
stream group

Note:

For more information on custom commands, see Use Cases of Custom Commands.

For more information on command compatibility of Memory Edition (cluster architecture), see Overview.

Command table download address.

Connection Group

Last updated：2023-10-20 10:54:52

Redis 2.8 (standard architecture) and Redis 4.0/5.0/6.2 (standard and cluster architectures) all support the `auth`, `echo`, `ping`, `quit`, and `select` commands.
Redis 2.8 (standard architecture) and Redis 4.0/5.0/6.2 (standard and cluster architectures) don't support the `client caching`, `client getredir`, `client info`, `client tracking`, `client trackinginfo`, `client unpause`, `reset`, `client list`, and `client kill` commands.
Redis 2.8 (standard architecture) and Redis 4.0/5.0/6.2 (cluster architecture) don't support the `swapdb` command, while Redis 4.0/5.0/6.2 (standard architecture) support this command.
The `hello` command is a newly added command in Redis 6.2 and is supported by both the v6.2 standard andcluster architectures.
Cross-slot access is not applicable to the `auth`, `echo`, `ping`, `quit`, `select`, `swapdb`, and `hello` commands.
Here is the detailed version information for the "connection" family commands (auth, echo, ping, quit, select, swapdb, hello). In the following table, ✓ indicates "supported", x indicates "unsupported", and - indicates that cross-slot access is not applicable to the command.

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	6.2 I Edit (Sta Arcl
auth	✓	✓	✓	✓	✓	✓
auth name pwd	x	x	x	x	x	✓
echo	✓	✓	✓	✓	✓	✓
ping	✓	✓	✓	✓	✓	✓
quit	✓	✓	✓	✓	✓	✓
select	✓	✓	✓	✓	✓	✓
swapdb	x	✓	x	✓	x	✓
hello	x	x	x	x	x	✓
client	x	x	x	x	x	x

caching						
client getredir	x	x	x	x	x	x
client info	x	x	x	x	x	x
client tracking	x	x	x	x	x	x
client trackinginfo	x	x	x	x	x	x
client unpause	x	x	x	x	x	x
reset	x	x	x	x	x	x
client list	x	x	x	x	x	x
client kill	x	x	x	x	x	x

Hash Group Keys

Last updated：2022-10-31 10:29:37

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	Cross-Slot Support in Memory Edition (Cluster Architecture
hdel	✓	✓	✓	✓	✓	-
hexists	✓	✓	✓	✓	✓	-
hget	✓	✓	✓	✓	✓	-
hgetall	✓	✓	✓	✓	✓	-
hincrby	✓	✓	✓	✓	✓	-
hincrbyfloat	✓	✓	✓	✓	✓	-
hkeys	✓	✓	✓	✓	✓	-
hlen	✓	✓	✓	✓	✓	-
hmget	✓	✓	✓	✓	✓	-
hmset	✓	✓	✓	✓	✓	-
hset	✓	✓	✓	✓	✓	-
hsetnx	✓	✓	✓	✓	✓	-
hstrlen	✓	✓	✓	✓	✓	-
hvals	✓	✓	✓	✓	✓	-
hscan	✓	✓	✓	✓	✓	-

Keys

Last updated：2022-10-31 10:29:37

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	Cross-Slot Support in Memory Edition (Cluster Architecture
touch	x	✓	✓	✓	✓	-
restore	✓	✓	✓	✓	✓	-
object	✓	✓	✓	✓	✓	-
unlink	x	✓	✓	✓	✓	x
wait	x	✓	✓	✓	✓	-
migrate	x	x	x	x	x	-
dump	✓	✓	✓	✓	✓	-
del	✓	✓	✓	✓	✓	✓
scan	✓	✓	✓	✓	✓	-
exists	✓	✓	✓	✓	✓	✓
expire	✓	✓	✓	✓	✓	-
expireat	✓	✓	✓	✓	✓	-
keys	✓	✓	✓	✓	✓	-
type	✓	✓	✓	✓	✓	-
move	✓	✓	✓	✓	✓	-
ttl	✓	✓	✓	✓	✓	-
persist	✓	✓	✓	✓	✓	-
pexpire	✓	✓	✓	✓	C✓	-
pexpireat	✓	✓	✓	✓	✓	-

pttl	✓	✓	✓	✓	✓	-
randomkey	✓	✓	✓	✓	✓	-
rename	✓	✓	✓	✓	✓	x
renamenx	✓	✓	✓	✓	✓	x
sort	✓	✓	✓	✓	✓	-

List Group

Last updated：2022-11-08 17:18:57

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	Memory Edition (Cluster Architecture with Cross-Slot Suppor
lindex	✓	✓	✓	✓	✓	-
linsert	✓	✓	✓	✓	✓	-
llen	✓	✓	✓	✓	✓	-
lpop	✓	✓	✓	✓	✓	-
lpush	✓	✓	✓	✓	✓	-
lpushx	✓	✓	✓	✓	✓	-
lrange	✓	✓	✓	✓	✓	-
lrem	✓	✓	✓	✓	✓	-
lset	✓	✓	✓	✓	✓	-
ltrim	✓	✓	✓	✓	✓	-
rpop	✓	✓	✓	✓	✓	-
rpoplpush	✓	✓	✓	✓	✓	x
rpush	✓	✓	✓	✓	✓	-
rpushx	✓	✓	✓	✓	✓	-
blpop	✓	✓	✓	✓	✓	x
brpop	✓	✓	✓	✓	✓	x
brpoplpush	✓	✓	✓	✓	✓	x

Pub Group and Sub Group

Last updated：2024-07-24 10:23:01

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	Memory Edition (Cluster Architectu with Cross Slot Supp
psubscribe	✓	✓	✓	✓	✓	-
pubsub	✓	✓	✓	✓	✓	-
publish	✓	✓	✓	✓	✓	-
punsubscribe	✓	✓	✓	✓	✓	-
subscribe	✓	✓	✓	✓	✓	-
unsubscribe	✓	✓	✓	✓	&310003;	-

Sets Group

Last updated：2022-11-08 17:18:58

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	Memory Edition (Cluster Architectu with Cros Slot Supp
sadd	✓	✓	✓	✓	✓	-
scard	✓	✓	✓	✓	✓	-
sdiff	✓	✓	✓	✓	✓	x
sdiffstore	✓	✓	✓	✓	✓	x
sinter	✓	✓	✓		✓	x
sinterstore	✓	✓	✓	✓	✓	x
sismember	✓	✓	✓	✓	✓	-
smembers	✓	✓	✓	✓	✓	-
smove	✓	✓	✓	✓	✓	x
spop	✓	✓	✓	✓	✓	-
srandmember	✓	✓	✓	✓	✓	-
srem	✓	✓	✓	✓	✓	-
sscan	✓	✓	✓	✓	✓	-
sunion	✓	✓	✓	✓	✓	x
sunionstore	✓	✓	✓	✓	✓	x

Sorted Sets Group

Last updated：2022-11-08 17:18:58

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	Mem Editi (Clus Arch with Slot
zadd	✓	✓	✓	✓	✓	-
zcard	✓	✓	✓	✓	✓	-
zcount	✓	✓	✓	✓	✓	-
zincrby	✓	✓	✓	✓	✓	-
zinterstore	✓	✓	✓	✓	✓	x
zlexcount	✓	✓	✓	✓	✓	-
zrange	✓	✓	✓	✓	✓	-
zrangebylex	✓	✓	✓	✓	✓	-
zrangebyscore	✓	✓	✓	✓	✓	-
zrank	✓	✓	✓	✓	✓	-
zrem	✓	✓	✓	✓	✓	-
zremrangebylex	✓	✓	✓	✓	✓	-
zremrangebyrank	✓	✓	✓	✓	✓	-
zremrangebyscore	✓	✓	✓	✓	✓	-
zrevrange	✓	✓	✓	✓	✓	-
zrevrangebylex	✓	✓	✓	✓	✓	-
zrevrangebyscore	✓	✓	✓	✓	✓	-
zscore	✓	✓	✓	✓	✓	-
zrevrank	✓	✓	✓	✓	✓	-

zscan	✓	✓	✓	✓	✓	-
zunionstore	✓	✓	✓	✓	✓	-
zpopmax	x	x	x	✓	✓	-
zpopmin	x	x	x	✓	✓	-
bzpopmax	x	x	x	✓	✓	-
bzpopmin	x	x	x	✓	✓	-

Strings Group

Last updated：2023-10-20 10:53:09

Redis 2.8 (standard architecture) and Redis 4.0、5.0/6.2 (standard and cluster architectures) support the `append` , `bitcount` , `bitop` , `bitpos` , `decr` , `decrby` , `get` , `getbit` , `getrange` , `getset` , `incr` , `incrby` , `incrbyfloat` , `mget` , `mset` , `msetnx` , `psetex` , `setex` , `set` , `setbit` , `setnx` , `setrange` , `strlen` , and `bitfield` commands. Redis 4.0 and 5.0 don’t support the `getdel` and `getex` commands, and Redis 6.2 supports the `getex` command.

Redis 2.8 standard architecture doesn’t support the `bitfield` , `bitfield_ro` , and `stralgo` commands.

In a cluster architecture, cross-slot access is applicable to the `mget` and `mset` commands other than the `bitop` and `msetnx` commands.

As new command for Redis 6.2, the `bitfield_ro` command is supported for both architectures, but the `stralgo` command is only supported for the standard architecture.

In the following table, ✓ indicates "supported", x indicates "unsupported", and - indicates that cross-slot access is not applicable to the command:

Command	Memory Edition (Standard Architecture)	Memory Edition (Standard Architecture)	Memory Edition (Cluster Architecture)	Memory Edition (Standard Architecture)	Memory Edition (Cluster Architecture)	Mem Editi (Clu Arch
append	✓	✓	✓	✓	✓	✓
bitcount	✓	✓	✓	✓	✓	✓
bitop	✓	✓	✓	✓	✓	✓
bitpos	✓	✓	✓	✓	✓	✓
decr	✓	✓	✓	✓	✓	✓
decrby	✓	✓	✓	✓	✓	✓
get	✓	✓	✓	✓	✓	✓
getbit	✓	✓	✓	✓	✓	✓
getrange	✓	✓	✓	✓	✓	✓
getset	✓	✓	✓	✓	✓	✓

incr	✓	✓	✓	✓	✓	✓
incrby	✓	✓	✓	✓	✓	✓
incrbyfloat	✓	✓	✓	✓	✓	✓
mget	✓	✓	✓	✓	✓	✓
mset	✓	✓	✓	✓	✓	✓
msetnx	✓	✓	✓	✓	✓	✓
psetex	✓	✓	✓	✓	✓	✓
setex	✓	✓	✓	✓	✓	✓
set	✓	✓	✓	✓	✓	✓
setbit	✓	✓	✓	✓	✓	✓
setnx	✓	✓	✓	✓	✓	✓
setrange	✓	✓	✓	✓	✓	✓
strlen	✓	✓	✓	✓	✓	✓
bitfield	x	✓	✓	✓	✓	✓
bitfield_ro	x	x	x	x	x	✓
bitfield_ro	x	x	x	x	x	✓
getdel	x	x	x	x	x	x
getex	x	x	x	x	x	✓

Transactions

Last updated：2022-11-08 17:18:58

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	Memory Edition (Cluster Architecture) with Cross-Slot Support
discard	✓	✓	✓	✓	✓	-
exec	✓	✓	✓	✓	✓	-
multi	✓	✓	✓	✓	✓	-
unwatch	✓	✓	✓	✓	✓	-
watch	✓	✓	✓	✓	✓	-

Hyperloglog Group

Last updated：2022-11-08 17:18:58

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	Memory Edition (Cluster Architecture) with Cross-Slot Support
pfadd	✓	✓	✓	✓	✓	-
pfcount	✓	✓	✓	✓	✓	x
pfmerge	✓	✓	✓	✓	✓	x

Scripting Group

Last updated：2022-11-08 17:18:58

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	Memory Edition (Cluster Architecture) with Cross-Slot Support
eval	✓	✓	✓	✓	✓	x
evalsha	✓	✓	✓	✓	✓	x
script debug	x	x	x	x	x	-
script exists	✓	✓	✓	✓	✓	x
script flush	✓	✓	✓	✓	✓	-
script load	✓	✓	✓	✓	✓	-
script kill	✓	✓	✓	✓	✓	-

Geo Group

Last updated：2022-11-08 17:18:58

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	Me Ed (Cl Ar wit Sl
geoadd	x	✓	✓	✓	✓	-
geohash	x	✓	✓	✓	✓	-
geopos	x	✓	✓	✓	✓	-
geodist	x	✓	✓	✓	✓	-
georadius	x	✓	✓	✓	✓	-
georadiusbymember	x	✓	✓	✓	✓	-

Server Group

Last updated：2022-11-08 17:18:58

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	Memory Edition (Cluster Architec Cross-S Support
bgrewriteaof	x	x	x	x	x	-
bgsave	x	x	x	x	x	-
client kill	x	x	x	x	x	-
sync	x	x	x	x	x	-
psync	x	x	x	x	x	-
client list	✓	✓	✓	✓	✓	-
client getname	✓	✓	✓	✓	✓	-
client pause	x	x	x	x	x	-
client reply	x	x	x	x	x	-
client setname	✓	✓	✓	✓	✓	-
command count	✓	✓	✓	✓	✓	-
command getkeys	x	✓	✓	✓	✓	-
command info	✓	✓	✓	✓	✓	-
slaveof	x	x	x	x	x	-
config rewrite	✓	✓	✓	✓	✓	-
config set	x	x	x	x	x	-
config resetstat	✓	✓	✓	✓	✓	-
debug object	x	x	x	x	x	-

debug segfault	x	x	x	x	x	-
role	✓	✓	✓	✓	✓	-
save	x	x	x	x	x	-
lastsave	✓	✓	✓	✓	✓	-
shutdown	x	x	x	x	x	-
MEMORY	x	✓	x	✓	x	-
command	✓	✓	✓	✓	✓	-
dbsize	✓	✓	✓	✓	✓	-
info	✓	✓	✓	✓	✓	-
time	✓	✓	✓	✓	✓	-
config get	✓	✓	✓	✓	✓	-
monitor	✓	✓	✓	✓	✓	-
flushdb	✓	✓	✓	✓	✓	-
flushall	✓	✓	✓	✓	✓	-
slowlog	✓	✓	x	✓	x	-
cluster keyslot	x	x	✓	x	✓	-
cluster nodes	x	x	✓	x	✓	-
cluster getkeysinslot	x	x	✓	x	✓	-
cluster slots	x	x	✓	x	✓	-
cluster info	x	x	✓	x	✓	-
cluster countkeysinslot	x	x	✓	x	✓	-
cluster (others)	x	x	x	x	x	-
module	x	x	x	x	x	-
lolwut	x	x	x	✓	✓	-

Stream Group

Last updated：2023-10-20 10:54:08

Redis 5.0 and 6.2 (standard and cluster architectures) support the `xinfo` , `xadd` , `xtrim` , `xdel` , `xrange` , `xrevrange` , `xlen` , `xread` , `xgroup` , `xreadgroup` , `xack` , `xclaim` , `xpending` commands. But Redis 2.8 (standard and cluster architectures) don't support these commands. Redis 6.0 supports the `xautoclaim` command, which is not supported in Redis 5.0.

Cross-slot access is not applicable to the `xread` and `xreadgroup` commands.

In the following table, ✓ indicates "supported", x indicates "unsupported", and - indicates that cross-slot access is not applicable to the command.

Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Memory Edition (Cluster Architecture)	6.2 M Edit (Clu Arch
xinfo	x	x	x	✓	✓	✓
xadd	x	x	x	✓	✓	✓
xtrim	x	x	x	✓	✓	✓
xdel	x	x	x	✓	✓	✓
xrange	x	x	x	✓	✓	✓
xrevrange	x	x	x	✓	✓	✓
xlen	x	x	x	✓	✓	✓
xread	x	x	x	✓	✓	✓
xgroup	x	x	x	✓	✓	✓
xreadgroup	x	x	x	✓	✓	✓
xack	x	x	x	✓	✓	✓
xclaim	x	x	x	✓	✓	✓
xpending	x	x	x	✓	✓	✓
xautoclaim	x	x	x	x	x	✓

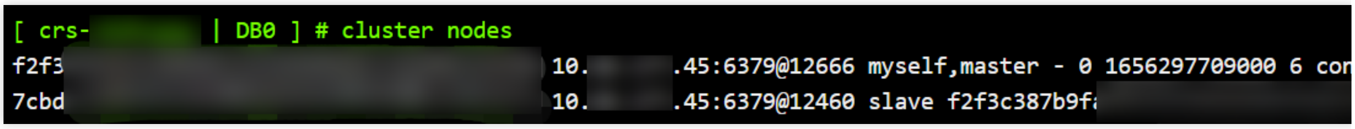
Use Case of Partially Supported Commands

Last updated：2023-05-23 10:45:36

Memory Edition (Cluster Architecture) is compatible with smart clients such as JedisCluster. For compatibility with JedisCluster, TencentDB for Redis modifies the IP list returned by the supported commands, and the IP address of each node in the returned information is the instance's private IPv4 address.

CLUSTER NODES

CLUSTER NODES is used to get the information of each node in a Redis cluster, where each output line represents a node. Node information includes node ID, private IPv4 address and port, node role (master or replica), attributes, and assigned slots.



```
[ crs-[redacted] | DB0 ] # cluster nodes
f2f3[redacted] 10.[redacted].45:6379@12666 myself,master - 0 1656297709000 6 con
7cbd[redacted] 10.[redacted].45:6379@12460 slave f2f3c387b9f[redacted]
```

CLUSTER SLOTS

CLUSTER SLOTS is used to get the mapping relationship between cluster slots and Redis instances. Each returned result contains:

Start slot range.

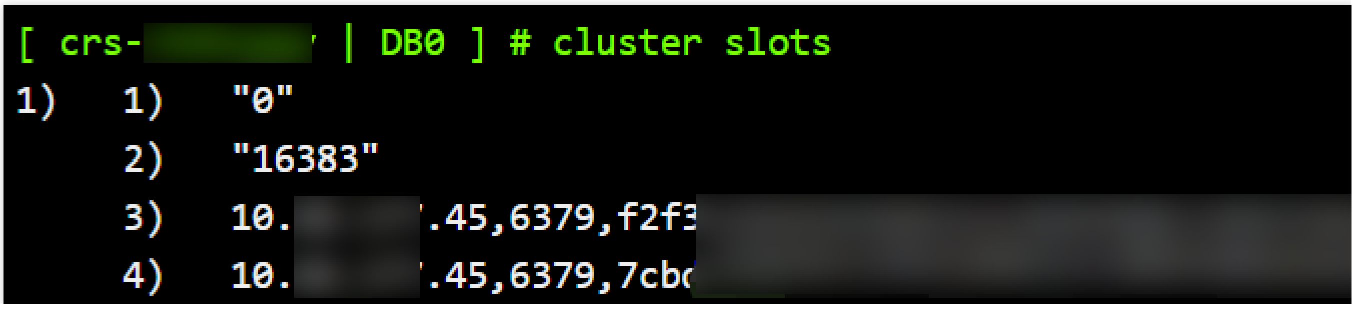
End slot range.

Information of the cluster's master node corresponding to the slot range, including the private IPv4 address, port, and node ID.

Information of the first replica of the cluster's master node corresponding to the slot range.

Information of the second replica.

And so on in a similar manner until the information of all replicas is returned.



```
[ crs-[redacted] | DB0 ] # cluster slots
1)  1)  "0"
    2)  "16383"
    3)  10.[redacted].45,6379,f2f3[redacted]
    4)  10.[redacted].45,6379,7cbd[redacted]
```

Use Cases of Custom Commands

Last updated：2023-05-23 10:32:13

Through VIP encapsulation, Memory Edition (Cluster Architecture) delivers a user experience in cluster mode comparable to the standalone edition, making it much easier for use in different scenarios. However, in Ops scenarios, each node in the cluster needs to be accessed frequently to locate exceptions. In this case, the custom command feature can add a **node ID** parameter to the parameter list of the original command in the format of `COMMAND arg1 arg2 ... [node ID]` in order to easily get the information of the specified node. The node ID can be obtained from the **Node Management** page in the TencentDB for Redis console or through the `cluster nodes` command.

Version Description

On proxy agent versions prior to v5.5.0, the node ID is required for the execution of custom commands, but it is unnecessary on v5.5.0 and later.

INFO

This command returns the information and statistics of a server.

Custom command format

```
info [section] [node ID]
```

Here, optional parameters can be used to select a specific part of the information:

`server` : The general information of a Redis server.

`clients` : The information of connected clients.

`memory` : The information of memory usage.

`persistence` : The information of RDB and AOF.

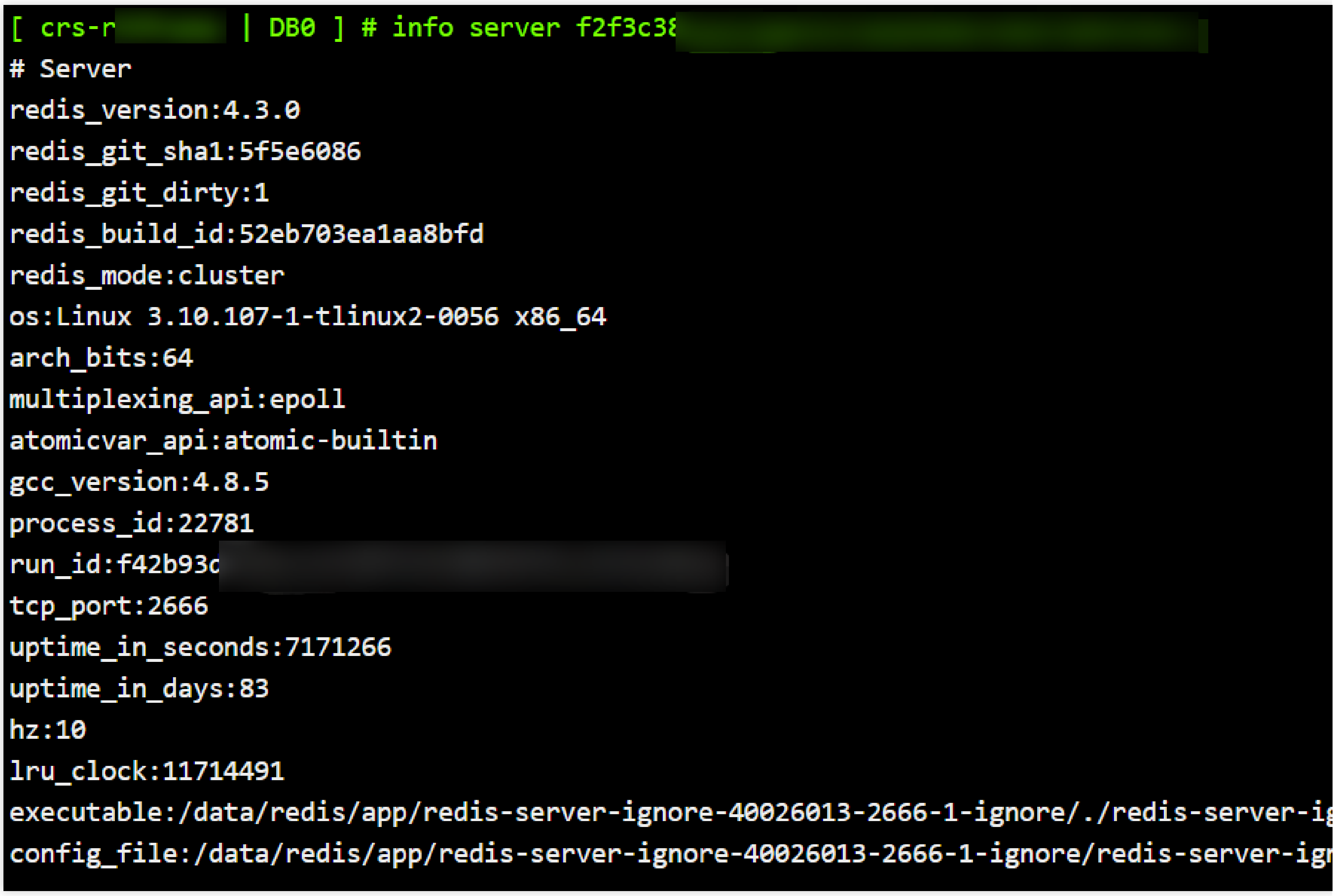
`stats` : The general statistics.

`replication` : The information of master/replica replication.

`cpu` : The information of CPU usage.
`commandstats` : The statistics of Redis commands.
`cluster` : The information of a Redis cluster.
`keyspace` : statistics of database
Optional parameters can also take the following values:
`all` : Returns all the information.
`default` : Returns the default information.

Sample code

The following example runs the `INFO` command with `section` being `server` :



```
[ crs-r[redacted] | DB0 ] # info server f2f3c3[redacted]
# Server
redis_version:4.3.0
redis_git_sha1:5f5e6086
redis_git_dirty:1
redis_build_id:52eb703ea1aa8bfd
redis_mode:cluster
os:Linux 3.10.107-1-tlinux2-0056 x86_64
arch_bits:64
multiplexing_api:epoll
atomicvar_api:atomic-builtin
gcc_version:4.8.5
process_id:22781
run_id:f42b93c[redacted]
tcp_port:2666
uptime_in_seconds:7171266
uptime_in_days:83
hz:10
lru_clock:11714491
executable:/data/redis/app/redis-server-ignore-40026013-2666-1-ignore/./redis-server-ig
config_file:/data/redis/app/redis-server-ignore-40026013-2666-1-ignore/redis-server-igr
```

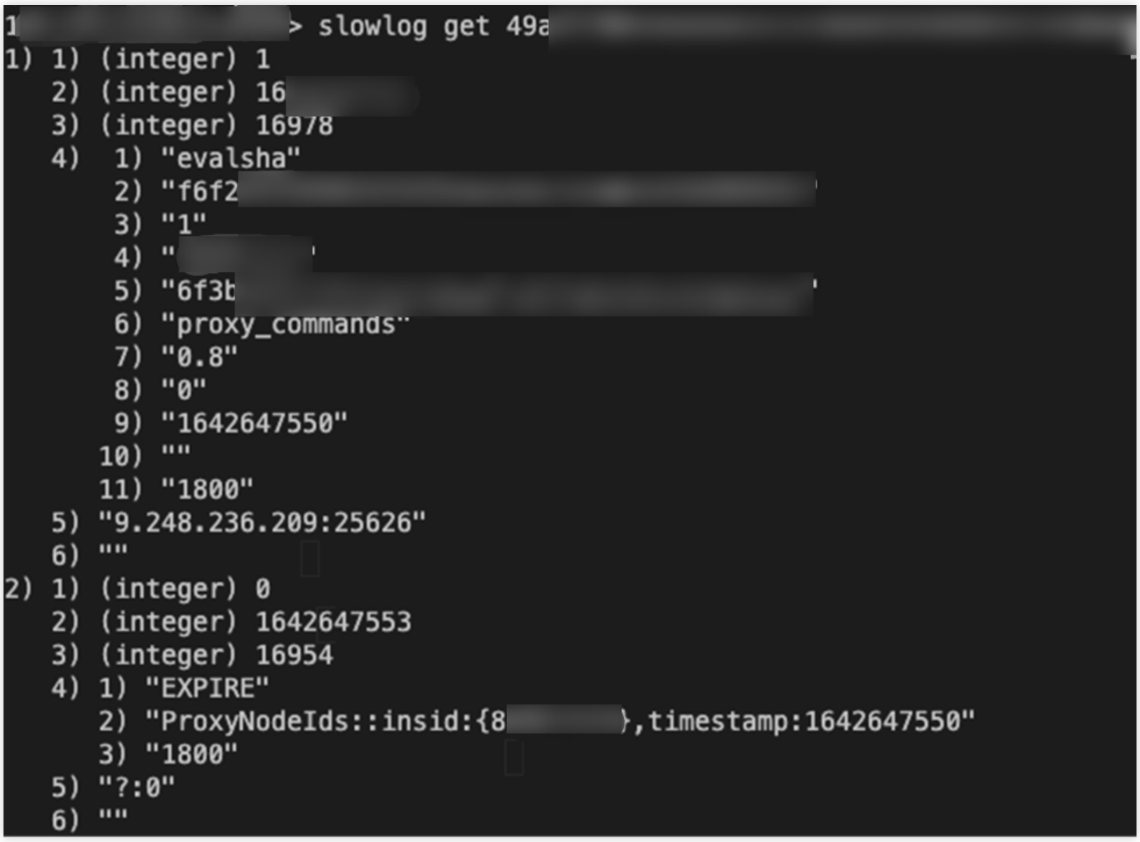
SLOWLOG

This command reads slow logs. It uses `SLOWLOG GET` to return the entries in slow logs. You can specify to return only the last N entries and pass other parameters to this command, such as `SLOWLOG GET 10` .

Custom command format

```
slowlog get [Redis node ID]
slowlog get [slow log quantity][Redis node ID]
```

Sample code



```
1[redacted]> slowlog get 49a[redacted]
1) 1) (integer) 1
   2) (integer) 16[redacted]
   3) (integer) 16978
   4)  1) "evalsha"
       2) "f6f2[redacted]"
       3) "1"
       4) "[redacted]"
       5) "6f3b[redacted]"
       6) "proxy_commands"
       7) "0.8"
       8) "0"
       9) "1642647550"
      10) ""
      11) "1800"
   5) "9.248.236.209:25626"
   6) ""
2) 1) (integer) 0
   2) (integer) 1642647553
   3) (integer) 16954
   4) 1) "EXPIRE"
      2) "ProxyNodeIds::insid:{8[redacted]},timestamp:1642647550"
      3) "1800"
   5) "?:0"
   6) ""
```

FLUSHDB

This command deletes all keys of the currently selected database. It will never fail.

Custom command format

```
flushdb [Redis node ID]
```

Sample code

```
cd-crs-rhxxxay.sql.tencentcdb.com:24894> flushdb f2f3c387b9fab0e67af02039845c60278b
OK
```

PING

This command is often used to test whether the connection still exists or to measure the latency.

Custom command format

```
ping [message] [node ID]
```

Sample code

```
[ crs-rh**** | DB0 ] # PING "PONG" f2f3c3************************
PONG
[ crs-rh**** | DB0 ] # PING "hello world"
hello world
```

KEYS

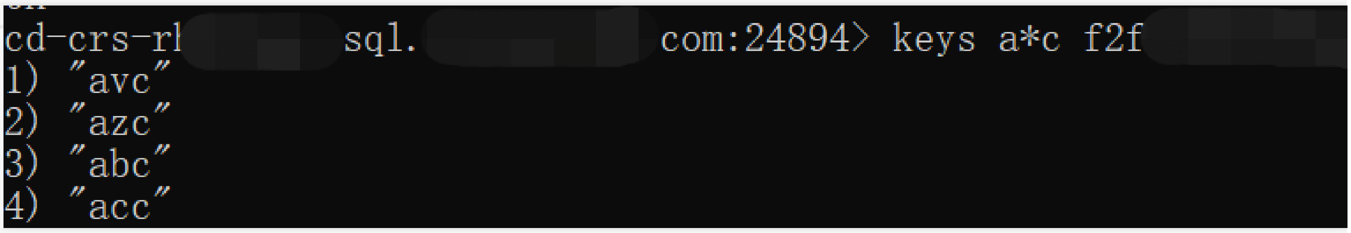
This command queries all the matched keys.

Custom command format

```
keys [pattern]  [Redis node ID]
```

Sample code

```
keys a* f2f3c3*************************
```



```
cd-crs-rl[redacted]sql.[redacted]com:24894> keys a*c f2f[redacted]
1) "avc"
2) "azc"
3) "abc"
4) "acc"
```

SCAN

Custom command format

```
scan cursor [MATCH pattern] [COUNT count] [Redis node ID]
```

Sample code

```
[ crs-******** | DB0 ] # scan 0 f2f3c3*************************
1)  "2"
```

IMonitor

The command needs to be executed on the proxy node, and the parameter is the ID of the Redis shard node.

```
imonitor [Redis node ID]
```

Use Cases

```
imonitor 3dba154c67925520ef1a1e2c41d8cc22d7f4****
+OK
+1680504260.729707 [0 127.0.0.1:6379] "auth" ******
+1680504260.730070 [0 127.0.0.1:6379] "info" "commandstats"
+1680504262.243004 [0 127.0.0.1:6379] "AUTH" ******
```

List of Supported DMC Commands

Last updated：2022-07-25 16:22:57

Database Management Center (DMC) allows you to log in to your TencentDB instances to access them, view their key metric information, and run Redis commands. The Redis commands currently supported by DMC are as listed below, where the supported custom commands include `INFO` , `SLOWLOG` , and `SCAN` .

In the following table, ✓ indicates "supported", x indicates "unsupported", and - indicates that cross-slot access is not applicable to the command:

Command Group	Command	2.8 Memory Edition (Standard Architecture)	4.0 Memory Edition (Standard Architecture)	4.0 Memory Edition (Cluster Architecture)	5.0 Memory Edition (Standard Architecture)	5.0 Me Editio (Clust Archit
connection group	echo	✓	✓	✓	✓	✓
	ping	✓	✓	Custom	✓	Custo
	select	✓	✓	✓	✓	✓
hash group	hdel	✓	✓	✓	✓	✓
	hexists	✓	✓	✓	✓	✓
	hget	✓	✓	✓	✓	✓
	hgetall	✓	✓	✓	✓	✓
	hincrby	✓	✓	✓	✓	✓
	hincrbyfloat	✓	✓	✓	✓	✓
	hkeys	✓	✓	✓	✓	✓
	hlen	✓	✓	✓	✓	✓
	hmget	✓	✓	✓	✓	✓
	hmset	✓	✓	✓	✓	✓
	hset	✓	✓	✓	✓	✓
	hsetnx	✓	✓	✓	✓	✓

	hstrlen	✓	✓	✓	✓	✓
	hvals	✓	✓	✓	✓	✓
	hscan	✓	✓	✓	✓	✓
keys group	del	✓	✓	✓	✓	✓
	scan	✓	✓	Custom	✓	Custo
	exists	✓	✓	✓	✓	✓
	expire	✓	✓	✓	✓	✓
	expireat	✓	✓	✓	✓	✓
	type	✓	✓	✓	✓	✓
	ttl	✓	✓	✓	✓	✓
	persist	✓	✓	✓	✓	✓
	pexpire	✓	✓	✓	✓	✓
	pexpireat	✓	✓	✓	✓	✓
	pttl	✓	✓	✓	✓	✓
	randomkey	✓	✓	✓	✓	✓
	rename	✓	✓	✓	✓	✓
	renamenx	✓	✓	✓	✓	✓
	touch	✓	✓	✓	✓	✓
	restore	✓	✓	✓	✓	✓
	unlink	x	✓	✓	✓	✓
	move	✓	✓	✓	✓	✓
	dump	✓	✓	✓	✓	✓
list group	lindex	✓	✓	✓	✓	✓
	linsert	✓	✓	✓	✓	✓
	llen	✓	✓	✓	✓	✓

	lpop	✓	✓	✓	✓	✓
	lpush	✓	✓	✓	✓	✓
	lpushx	✓	✓	✓	✓	✓
	lrange	✓	✓	✓	✓	✓
	lrem	✓	✓	✓	✓	✓
	lset	✓	✓	✓	✓	✓
	ltrim	✓	✓	✓	✓	✓
	rpop	✓	✓	✓	✓	✓
	rpoplpush	✓	✓	✓	✓	✓
	rpush	✓	✓	✓	✓	✓
	rpushx	✓	✓	✓	✓	✓
	blpop	✓	✓	✓	✓	✓
	brpop	✓	✓	✓	✓	✓
	brpoplpush	✓	✓	✓	✓	✓
sets group	sadd	✓	✓	✓	✓	✓
	scard	✓	✓	✓	✓	✓
	sdiff	✓	✓	✓	✓	✓
	sdiffstore	✓	✓	✓	✓	✓
	sinter	✓	✓	✓	✓	✓
	sinterstore	✓	✓	✓	✓	✓
	sismember	✓	✓	✓	✓	✓
	smembers	✓	✓	✓	✓	✓
	smove	✓	✓	✓	✓	✓
	spop	✓	✓	✓	✓	✓
	srandmember	✓	✓	✓	✓	✓

	srem	✓	✓	✓	✓	✓
	sscan	✓	✓	✓	✓	✓
	sunion	✓	✓	✓	✓	✓
	sunionstore	✓	✓	✓	✓	✓
sorted sets group	zadd	✓	✓	✓	✓	✓
	zcard	✓	✓	✓	✓	✓
	zcount	✓	✓	✓	✓	✓
	zincrby	✓	✓	✓	✓	✓
	zinterstore	✓	✓	✓	✓	✓
	zlexcount	✓	✓	✓	✓	✓
	zrange	✓	✓	✓	✓	✓
	zrangebylex	✓	✓	✓	✓	✓
	zrangebyscore	✓	✓	✓	✓	✓
	zrank	✓	✓	✓	✓	✓
	zrem	✓	✓	✓	✓	✓
	zremrangebylex	✓	✓	✓	✓	✓
	zremrangebyrank	✓	✓	✓	✓	✓
	zremrangebyscore	✓	✓	✓	✓	✓
	zrevrange	✓	✓	✓	✓	✓
	zrevrangebylex	✓	✓	✓	✓	✓
	zrevrangebyscore	✓	✓	✓	✓	✓
	zscore	✓	✓	✓	✓	✓
	zrevrank	✓	✓	✓	✓	✓
	zscan	✓	✓	✓	✓	✓
	zunionstore	✓	✓	✓	✓	✓

	zpopmax	x	x	x	✓	✓
	zpopmin	x	x	x	✓	✓
	bzpopmax	x	x	x	✓	✓
	bzpopmin	x	x	x	✓	✓
strings group	append	✓	✓	✓	✓	✓
	bitcount	✓	✓	✓	✓	✓
	bitop	✓	✓	✓	✓	✓
	bitpos	✓	✓	✓	✓	✓
	decr	✓	✓	✓	✓	✓
	decrby	✓	✓	✓	✓	✓
	get	✓	✓	✓	✓	✓
	getbit	✓	✓	✓	✓	✓
	getrange	✓	✓	✓	✓	✓
	getset	✓	✓	✓	✓	✓
	incr	✓	✓	✓	✓	✓
	incrby	✓	✓	✓	✓	✓
	incrbyfloat	✓	✓	✓	✓	✓
	mget	✓	✓	✓	✓	✓
	mset	✓	✓	✓	✓	✓
	msetnx	✓	✓	✓	✓	✓
	psetex	✓	✓	✓	✓	✓
	setex	✓	✓	✓	✓	✓
	set	✓	✓	✓	✓	✓
	setbit	✓	✓	✓	✓	✓
	setnx	✓	✓	✓	✓	✓

	setrange	✓	✓	✓	✓	✓
	strlen	✓	✓	✓	✓	✓
	bitfield	x	✓	✓	✓	✓
hyperloglog group	pfadd	✓	✓	✓	✓	✓
	pfcount	✓	✓	✓	✓	✓
	pfmerge	✓	✓	✓	✓	✓
server group	client list	✓	✓	✓	✓	✓
	client getname	✓	✓	✓	✓	✓
	client setname	✓	✓	✓	✓	✓
	dbsize	✓	✓	✓	✓	✓
	info	✓	✓	Custom	✓	Custo
	time	✓	✓	✓	✓	✓
	lastsave	✓	✓	✓	✓	✓
	slowlog	✓	✓	Custom	✓	Custo
	cluster keyslot	x	x	✓	x	✓
	cluster nodes	x	x	✓	x	✓
	cluster slots	x	x	✓	x	✓
	cluster info	x	x	✓	x	✓
	lolwut	x	x	x	✓	✓