

# 【Redis】相關指令

## 相關指令

### 連線

```
# redis-cli -h {$host} -p {$port} -a password
# ping 檢測服務正常
$redis-cli -h 127.0.0.1 -p 6379 -a "mypass"
redis 127.0.0.1:6379>
redis 127.0.0.1:6379> PING

PONG
```

### 使用redis-cli 進入redis

```
127.0.0.1:6379> select 1
OK
127.0.0.1:6379[1]>
```

### 進入db (具有16個

數據庫 (DB) 。這些數據庫編號從0到15，用於存儲不同的數據)

```
root@62952ea40eb3:/data# redis-cli
127.0.0.1:6379>
```

### 查詢key (線上環境勿用，請用scan)

```
#keys pattern
127.0.0.1:6379[2]> keys *
1) "v5.25.1_autocompleteKeywordV3_ASU"
2) "v5.25.1_autocompleteKeywordV3_AU"
3) "v5.25.1_autocompleteKeywordV3_ASUS"
4) "v5.25.1_autocompleteKeywordV3_AUS"
5) "v5.25.1_autocompleteKeywordV3_AS"
6) "v5.25.1_autocompleteKeywordV3_A"
7) "v5.25.1_autocompleteKeywordV3_AUU"
127.0.0.1:6379[2]>

#####
127.0.0.1:6379[2]> keys *autocompleteKeywordV3*
1) "v5.25.1_autocompleteKeywordV3_\xe9\x80\x81\xe5\xa5\xbd\xe7\xa6\xae"
2) "v5.25.1_autocompleteKeywordV3_\xe5\xb8\xb8\xe6\x98\xa5"
3) "v5.25.1_autocompleteKeywordV3_ASUS"
4) "v5.25.1_autocompleteKeywordV3_\xe5\xb8\xb8"
5) "v5.25.1_autocompleteKeywordV3_\xe7\x9b\xb4\xe6\x92\xad"
6) "v5.25.1_autocompleteKeywordV3_\xe7\x9b\xb4"
7) "v5.25.1_autocompleteKeywordV3_\xe9\x80\x81\xe5\xa5\xbd"
8) "v5.25.1_autocompleteKeywordV3_\xe7\x9b\xb4\xe6\x92\xad\xe6\xb8\x85\xe5\x96\xae"
9) "v5.25.1_autocompleteKeywordV3_\xe9\x80\x81"
```

### 查詢 scan

```
# scan {index} match {pattern}
#返回分2個部分如上面的代碼中， 1) 代表下一次迭代的游標，2) 代表本次迭代的結果集
#，注意如果返回游標0就代表全部匹配完成。
127.0.0.1:6379> scan 0 MATCH tony*
1) "42"
2) 1) "tony25"
   2) "tony2519"
   3) "tony2529"
   4) "tony2510"
   5) "tony2523"
   6) "tony255"
```

## 查詢資料

查詢該DB 資料總數

刪除該DB 所有資料

```
(integer) 0
```

## 刪除該db資料

```
# del key1 key2 ...
127.0.0.1:6379[2]> keys *
1) "v5.25.1_autocompleteKeywordV3_\xe9\x80\x81\xe5\xa5\xbd\xe7\xa6\xae"
2) "v5.25.1_autocompleteKeywordV3_\xe5\xb8\xb8\xe6\x98\xa5"
3) "v5.25.1_autocompleteKeywordV3_ASUS"
4) "v5.25.1_autocompleteKeywordV3_\xe5\xb8\xb8"
5) "v5.25.1_autocompleteKeywordV3_\xe7\x9b\xb4\xe6\x92\xad"
6) "v5.25.1_autocompleteKeywordV3_\xe7\x9b\xb4"
7) "v5.25.1_autocompleteKeywordV3_\xe9\x80\x81\xe5\xa5\xbd"
8) "v5.25.1_autocompleteKeywordV3_\xe7\x9b\xb4\xe6\x92\xad\xe6\xb8\x85\xe5\x96\xae"
9) "v5.25.1_autocompleteKeywordV3_\xe9\x80\x81"
127.0.0.1:6379[2]> del v5.25.1_autocompleteKeywordV3_ASUS
(integer) 1
127.0.0.1:6379[2]> keys *
1) "v5.25.1_autocompleteKeywordV3_\xe9\x80\x81\xe5\xa5\xbd\xe7\xa6\xae"
2) "v5.25.1_autocompleteKeywordV3_\xe5\xb8\xb8\xe6\x98\xa5"
3) "v5.25.1_autocompleteKeywordV3_\xe5\xb8\xb8"
4) "v5.25.1_autocompleteKeywordV3_\xe7\x9b\xb4\xe6\x92\xad"
5) "v5.25.1_autocompleteKeywordV3_\xe7\x9b\xb4"
6) "v5.25.1_autocompleteKeywordV3_\xe9\x80\x81\xe5\xa5\xbd"
7) "v5.25.1_autocompleteKeywordV3_\xe7\x9b\xb4\xe6\x92\xad\xe6\xb8\x85\xe5\x96\xae"
8) "v5.25.1_autocompleteKeywordV3_\xe9\x80\x81"
127.0.0.1:6379[2]>
```

## 刪除所有redis資料

```
# 清除整台redis 資料
127.0.0.1:6379[2]> flushall
OK
```

## 寫入資料(字串)

```
127.0.0.1:6379> set key value [EX seconds|PX milliseconds|KEEPTTL] [NX|XX]
```

set <key> <value>: 寫入資料

NX：當數據庫中的鍵不存在時，可以將鍵-值添加到數據庫

XX：當數據庫中的鍵存在時，可以將鍵-值添加到數據庫，與 NX 參數互斥

EX：鍵的超時秒數

PX：鍵的超時毫秒數，與 EX 互斥

get <key>：查詢對應鍵值

append <key> <value>：將給定的 <value> 追加到原值的末尾

strlen <key>：獲得值的長度

setnx <key> <value>：只有在鍵不存在時設置鍵的值

incr <key>：將鍵中儲存的數字值增加1

只能對數字值操作，如果為空，新增值為1

decr <key>：將鍵中儲存的數字值減少1

只能對數字值操作，如果為空，新增值為-1

incrby / decrby <key> <步長>：將鍵中儲存的數字值增減。自定義步長。

mset <key1> <value1> <key2> <value2> ...：同時設置一個或多個鍵-值對

mget <key1> <key2> <key3> ...：同時獲取一個或多個值

msetnx <key1> <value1> <key2> <value2> ...：同時設置一個或多個鍵-值對，當且僅當所有給定的鍵都不存在。原子性，有一個失敗則都失敗

getrange <key> <起始位置> <結束位置>：獲得值的範圍，類似於 Java 中的 substring，前包，後包

setrange <key> <起始位置> <value>：用 <value> 覆蓋 <key> 所儲存的字符串值，從 <起始位置> 開始（索引從0開始）。

setex <key> <過期時間> <value>：設置鍵值的同時，設置過期時間，單位秒。

getset <key> <value>：以新換舊，設置了新值同時獲得舊值。

## 寫入資料(List)

lpush/rpush <key> <value1> <value2> <value3> ...：從左邊/右邊插入一個或多個值。  
lpop/rpop <key>：從左邊/右邊吐出一個值。值在鍵在，值光鍵亡。

rpoplpush <key1> <key2>：從 <key1> 列表右邊吐出一個值，插到 <key2> 列表左邊。

lrange <key> <start> <stop>：按照索引下標獲得元素（從左到右）。  
lrange mylist 0 -1：0 左邊第一個，-1 右邊第一個，（0-1 表示獲取所有）。  
lindex <key> <index>：按照索引下標獲得元素（從左到右）。  
llen <key>：獲得列表長度。

linsert <key> before <value> <newvalue>：在 <value> 的後面插入 <newvalue>。  
lrem <key> <n> <value>：從左邊刪除 n 個 value（從左到右）。  
lset <key> <index> <value>：將列表 key 下標為 index 的值替換成 value。

## 寫入資料(Set)

sadd <key> <value1> <value2> ...：將一個或多個 member 元素加入到集合 key 中，已經存在的 member 元素將被忽略。  
smembers <key>：取出該集合的所有值。  
sismember <key> <value>：判斷集合 <key> 是否含有該 <value> 值，有1，沒有0。  
scard <key>：返回該集合的元素個數。  
srem <key> <value1> <value2> ...：刪除集合中的某個元素。  
spop <key>：隨機從該集合中吐出一個值。  
srandmember <key> <n>：隨機從該集合中取出 n 個值。不會從集合中刪除。  
smove <source> <destination> <value>：將集合中一個值從一個集合移動到另一個集合。  
sinter <key1> <key2>：返回兩個集合的交集元素。  
sunion <key1> <key2>：返回兩個集合的聯集元素。  
sdiff <key1> <key2>：返回兩個集合的差集元素（key1 中的，不包含 key2 中的）。

## 寫入資料(Zset)

zadd <key> <score1> <value1> <score2> <value2>...：將一個或多個 member 元素及其 score 值加入到有序集 key 當中。  
zrange <key> <start> <stop> [WITHSCORES]：返回有序集 key 中，下標在 <start> 到 <stop> 之間元素。帶有 WITHSCORES，可以讓分數一起和值返回到結果集。  
zrangebyscore <key> <minmax> [withscores] [limit offset count]：返回有序集 key 中，所有 score 值介於 min 和 max 之間（包括等於 min 或 max）的成員。有序集成員按 score 值遞增（從小到大）次序排列。  
zrevrangebyscore <key> <maxmin> [withscores] [limit offset count]：同上，改為從大到小排列。  
zincrby <key> <increment> <value>：為元素的 score 加上增量。  
zrem <key> <value>：刪除該集合下，指定值的元素。  
zcount <key> <min> <max>：統計該集合，分數區間內的元素個數。  
zrank <key> <value>：返回該值在集合中的排名，從0開始。

## 寫入資料(Hash)

hset <key> <field> <value>：給 <key> 集合中的 <field> 鍵賦值 <value>  
hget <key1> <field>：從 <key1> 集合 <field> 取出 value  
hmset <key1> <field1> <value1> <field2> <value2>...：批量設置 hash 的值  
hexists <key1> <field>：查看哈希表 key 中，給定域 field 是否存在。  
hkeys <key>：列出該 hash 集合的所有 field  
hvals <key>：列出該 hash 集合的所有 value  
hincrby <key> <field> <increment>：為哈希表 key 中的域 field 的值加上增量 1 或 -1  
hsetnx <key> <field> <value>：將哈希表 key 中的域 field 的值設置為 value，當且僅當域 field 不存在。

## 相關資源

- 在線練習 <https://try.redis.io/>
- 文檔總覽 <http://www.redis.cn/documentation.html>
- 命令文檔 <http://www.redis.cn/commands.html>
- docker中安装并配置redis <https://cloud.tencent.com/developer/article/1670205>
- 【Redis】用Docker-Compose搭建Redis Sentinel
- 【Redis】用Docker-Compose搭建Redis主从复制

⊙修訂版本 #12

★由 treeman 建立於 26 @ 2023 08:40:36

✎由 treeman 更新於 1 Q@ 2023 19:31:57