

title	date	categories	tags
Redis 数据结构之 String	Sat May 21 2016 18:31:23 GMT+0800 (CST)	Redis	Redis

前言

Redis作为cache服务器，支持多种数据结构，String、List、Hash、Set、Zset。多种数据结构的存在，使得 Redis 适用于多种业务,Redis的适用也越发广泛，本文就介绍Redis中最简单的数据结构 String 的操作命令。

简介

String类似 memcache 的结构，即最简单的 key-value 结构，一个Key对应一个简单的value,简单明了.

命令简介

SET

语法： SET key value [NX] [XX] [EX] [PX [milliseconds]] 设置一对key value

必选参数说明

SET：命令

key：待设置的key

value：设置的key的value

可选参数说明

NX：表示key不存在才设置，如果存在则返回NULL

XX：表示key存在时才设置，如果不存在则返回NULL

EX seconds : 设置过期时间, 过期时间精确为秒 **PX millisecond** : 设置过期时间, 过期时间精确为毫秒

设置一对 key value 不带任何可选参数 SET key value

```
127.0.0.1:6379> SET key value
```

```
OK
```

成功

设置一对 key(已存在) value, 带可选参数 **NX**

```
127.0.0.1:6379> SET key value NX
```

```
(nil)
```

key 已经存在, 所以设置失败, 返回 nil

设置一对 key(不存在) value, 带可选参数 **NX**

```
127.0.0.1:6379> SET key1 value1 NX
```

```
OK
```

key1 不存在, 所以设置成功, 返回 OK

设置一对 key(不存在) value, 带可选参数 **XX**

```
127.0.0.1:6379> SET key3 value3 XX
```

```
(nil)
```

key3 不存在, 所以设置失败

设置一对 key(存在) value, 带可选参数 **XX**

```
127.0.0.1:6379> SET key value3 XX
```

```
OK
```

key 存在, 所以设置成功

设置一对 key(存在) value, 带可选参数 **EX**

```
127.0.0.1:6379> SET key value EX 10086
OK
127.0.0.1:6379> ttl key
(integer) 10084
```

设置一对 key(存在) value，带可选参数 **PX**

```
127.0.0.1:6379> SET key value PX 10086000
OK
127.0.0.1:6379> ttl key
(integer) 10085
```

EX 与 **PX** 参数都是设置key的过期时间，差异为一个单位为 **秒** 一个单位为 **毫秒**

SETNX

语法：SETNX key value

所有参数为 **必选** 参数,设置一对key value，如果key存在，则设置失败，等同于 SET key value NX

```
127.0.0.1:6379> SETNX key value
(integer) 0
key 已经存在，设置失败
```

SETEX

语法：SETEX key expire value

所有参数为必选参数，设置一对 key value，并设过期时间,单位为秒，等同于 SET key value **EX** expire

```
127.0.0.1:6379> SETEX key 10086 value
OK
127.0.0.1:6379> ttl key
(integer) 10084
```

PSETEX

语法： PSETEX key expire value

所有参数为必选参数，设置一对 key value，并设过期时间,单位为毫秒，
等同于 SET key value PX expire

```
127.0.0.1:6379> PSETEX key 10086000 value
OK
127.0.0.1:6379> ttl key
(integer) 10084
```

GET

语法： GET key

所有参数为必选参数，获取指定 key 的value,成功返回对应的 value，失败返回 NULL

```
127.0.0.1:6379> GET key
"value"
127.0.0.1:6379> GET Keykk
(nil)
```

GETSET

语法：GETSET key value

所有参数为必选参数，获取指定 key 的value，并设置 key 的值为新值 value

```
127.0.0.1:6379> GETSET key valuevalue
"value"
127.0.0.1:6379> GET key
"valuevalue"
```

SETRANGE

语法：SETRANGE key offset value

所有参数为必选参数，设置指定 key ， 偏移量 offset 后的值为 value ，影响范围为 value 的长度， offset 不能小于0

```
127.0.0.1:6379> GET key
"valuevalue"
127.0.0.1:6379> SETRANGE key 2 kk
(integer) 10
127.0.0.1:6379> GET key
"vakkevalue"
```

GETRANGE

语法：GETRANGE key start end

所有参数为必选参数，获取指定key指定区间的value值， start 、 end 可以为负数，如果为负数则反向取区间

```
127.0.0.1:6379> GET key
"vakkevalue"
127.0.0.1:6379> GETRANGE key 2 5
```

```
"kkey"
```

```
127.0.0.1:6379> GETRANGE key -5 -2
```

```
"valu"
```

MGET

语法：MGET key1 [key2 key3 ...]

所有参数为必选，key值至少为一个，获取多个key的value值，key值存的返回对应的value，不存在的返回NULL

```
127.0.0.1:6379> MGET key keykey key1 key1234
```

```
1) "vakkevalue"
```

```
2) (nil)
```

```
3) "value1"
```

```
4) (nil)
```

MSET

语法：MSET key1 value1 [key2 value2 key3 value3 ...]

所有参数为必选，key、value 对至少为一对。该命令功能是设置多对 key-value 值。

```
127.0.0.1:6379> MSET key1 v1 key2 v2 key3 v3
```

```
OK 127.0.0.1:6379> MGET key1 key2 key3
```

```
1) "v1"
```

```
2) "v2"
```

```
3) "v3"
```

MSETNX

语法：MSETNX key1 value1 [key2 value2 key3 value3 ...] 所有参数为必选，key、value 对至少为一对。该命令功能是设置多对 key-value 值，如果 key 存在，则不做任何操作。

```
127.0.0.1:6379> MGET key1 key2 key3
1) "v1"
2) (nil)
3) "v3" 127.0.0.1:6379> MSETNX key1 v11 key2 v12 key3 v13
(integer) 0
127.0.0.1:6379> MGET key1 key2 key3
1) "v1"
2) (nil)
3) "v3"
127.0.0.1:6379> MSETNX key11 v11 key12 v12 key13 v13
(integer) 1
127.0.0.1:6379> MGET key11 key12 key13
1) "v11"
2) "v12"
3) "v13"
```

INCR

语法：INCR key

所有参数为必选，指定 key 做加1操作。指定 key 对应的值必须为整型，否则返回错误,操作成功后返回操作后的值

```
127.0.0.1:6379> GET key
"vakkevalue"
127.0.0.1:6379> INCR key
(error) ERR value is not an integer or out of range 127.0.0.1:6379>
SET key 100 OK
```

```
127.0.0.1:6379> INCR key (integer) 101
```

DECR

语法： DECR key

所有参数为必选，指定 key 做减1操作。指定 key 对应的值必须为整型，否则返回错误,操作成功后返回操作后的值。为 DECR 的逆操作。

```
127.0.0.1:6379> DECR key  
(integer) 100
```

INCRBY

语法： INCRBY key data 所有参数为必选参数，指定 key 做加 data 操作,指定 key 对应的值和 data 必须为整型，否则返回错误,操作成功后返回操作后的值

```
27.0.0.1:6379> INCRBY key djfklaj  
(error) ERR value is not an integer or out of range  
127.0.0.1:6379> INCRBY key 20  
(integer) 120
```

DECRBY

语法： DECRBY key data 所有参数为必选参数，指定 key 做减 data 操作,指定 key 对应的值和 data 必须为整型，否则返回错误,操作成功后返回操作后的值

```
127.0.0.1:6379> DECRBY key 10  
(integer) 110
```

INCRBYFLOAT

语法：INCRBYFLOAT key data 所有参数为必选参数，对指定 key 做加 data 操作，data 为浮点型数据，key 对应的value也必须为数值类型，否则返回错误。操作成功后返回操作后的数值

```
127.0.0.1:6379> GET key
"22.22"
127.0.0.1:6379> INCRBYFLOAT key 1.1
"23.32"
127.0.0.1:6379> GET nonumber
"kdjfl"
127.0.0.1:6379> INCRBYFLOAT nonumber 1.1
(error) ERR value is not a valid float
```

APPEND

语法：APPEND key appendvalue 所有参数为必选参数，在指定 key 的 value 值后追加 appendvalue ,操作成功后返回新值得长度，如果 key 对应的 value 不存在，则以 appendvalue 创建一个新值

```
127.0.0.1:6379> GET key
"test"
127.0.0.1:6379> APPEND key append
(integer) 10
127.0.0.1:6379> GET key
"testappend"
127.0.0.1:6379> GET append
(nil)
127.0.0.1:6379> APPEND append append
```

```
(integer) 6
```

```
127.0.0.1:6379> GET append
```

```
"append"
```

总结

Redis String 数据结构看之简单，实则大有乾坤，有待探寻。String类型提供了丰富的操作命令，可以满足大部分针对String的操作要求。工具已提供，怎么用就看个人能力了。