# Redis

# 1、常用的命令：

## 1、**字符串 - string**

**（value<=512MB）**

**Set key value -- 设置key**

**Get key -- 获取**

**Del key -- 删除**

**Type key -- 判断类型**

**Incr key -- 当存储的字符串是整数类型，让键值递增，并返回递增后的值（key不存在，则返回0；不是整型，则报错）**

**Incrby key num -- 以num递增**

**Decr key -- 递减**

**Decrby key num -- 以num递减**

**Append key value -- 向尾部增加**

**Strlen key -- 获取字符串的长度**

## 2、散列类型 - map

**Hset key field value -- 赋值**

**Hget key field -- 取值**

**Hmset key field value [field value …]**

**Hmget key field [field …]**

**Hgetall key -- 获取key的所有字段**

**Hexists key field -- 判断key的字段是否存在**

**Hsetnx key field value -- 存在，则不做处理；不存在字段，则添加字段并赋值**

**Hincrby key field increment -- 数值类型增加**

**Hdel key field [field ….] -- 删除字段**

**Hkeys key -- 获取key的所有字段**

**Hvals key -- 获取key的所有字段值**

**Hlen key -- 获取key的字段数**

## 3、列表类型 - list

**可以存储有序的字符串列表（向列表两端添加/删除元素），内部是使用双向链表实现的**

**Lpush key value [value ….] -- 列表左边添加**

**Rpush key value [value ….] -- 列表右边添加**

**Lpop key -- 左边弹出元素**

**Rpop key -- 右边弹出元素**

**Llen key -- 获取key列表中的元素个数**

**Lrange key start end -- 获取key列表中start到end之间的元素（下标从0开始的、包括两端）**

**start -- end < 0 , 则表示从右边开始**

**lrem key count value -- 删除列表key中的前count的值为value的元素**

**count > 0 ，则表示从左边开始**

**count < 0 ，则表示从右边开始**

**count = 0 ，则删除所有值为value的元素**

**lindex key index --获取列表key索引为index的值**

**lset key index value – 设置列表key中索引值为index的值为value**

**ltrim key start end – 截取列表key中start到end之间的值**

**linsert key before/after pivot value -- 向列表key中在值为pivot前/后插入value**

如：linsert sum before 7 3 , 表示向sum列表中在值为7之前插入3

Rpoplpush source dest -- 将一个列表移到另一个列表（当source=dest时，循环不断将队尾元素移到队首去）

## 4、集合类型 – set

**内容不重复**

**Sadd key member [member …..] -- 增加元素**

**Srem key member [member ….] -- 删除元素**

**Smembers key -- 获取集合key中所有元素**

**Sismember key member -- 判断集合key中是否包含member元素**

**Sdiff key [key …..] -- 差运算**

**Sinter key [key ….] -- 交集**

**Sunion key [key …..] -- 并集**

**（其中操作后，原集合内容不变）**

**Scard key -- 获取集合中的元素个数**

## 5、有序集合类型 – sorted set

### 5.1、添加

**Zadd key score member [score member ….] – 向集合key中添加分数为score,元素为member**

**Zincrby key increment member -- 向集合key中的member元素的值加上increment**

### 5.2、获取

**Zscore key member – 获取集合key中元素为member的分数**

**Zrange key start end [withscores] -- 获取集合key的start和end间的元素/携带分数（默认小到大顺序）**

**Zrevrange key start end [withscores] – ：大到小**

**Zrangebyscore key min max [withscores] -- 获取集合key中min到max之间的成绩(默认包括两端)**

**如果不需要两端，则可以在min/max前面 加 ‘(’**

**-inf +inf 分别表示负无穷/正无穷**

**Zcard key -- 获取集合中的元素个数**

**Zcount key min max -- 获取min-max分数范围的元素个数**

**Zrank key member – 获取排名(小到大 ，0开始) Zrevrank key member – (大到小，0开始)**

### 5.3、删除

**Zrem key member [member ….] -- 删除元素**

**Zremrangebyrank key start end -- 删除start-end下标范围的元素**

**Zremrangebyscore key min max -- 删除分数在min-max间的元素**

# 2、事务

一个事务中的命令，要么执行成功，要么就都不执行。

**原理：先将属于一个事务的命令发送给redis，然后再让redis依次执行这些命令。**

**>multi**

**>相关命令**

**>exec**

## 错误处理

**事务中发生的错误情况：**

1. **语法错误（都不会允许）**
2. **运行错误（正确的会允许）**

## 2、watch

**Watch命令可以监控一个/多个键值，如果其中有被修改，则之后的事务就不会被执行。**

**>Watch key [key ….]**

**>multi**

**>相关命令**

**>exec**

**Unwatch 来保证下一个事务不受影响**

## 3、过期时间

**Redis中使用expire命令来设置一个键的过期时间，到时间后redis自动删除**

**Expire key seconds (单位秒)**

**Ttl key 查看键的剩余时间（键已不存在，则返回-2；没有给 键设置失效时间，则返-1）**

**Persist key -- 取消过期时间（永久性）**

# 3、任务队列

**使用列表来实现的**

**Brpop key timeout -- 一旦key中有值，则输出**

**优先队列**

**Blpop key [key …..] timeout – 同时检测多个键，一旦其中一个键有值，则会被弹出。**

# 4、发布/订阅

**发布者： Publish channel message**

**订阅者： Subscribe channel [channel …..]**

**Unsubscribe [channel …..] – 取消订阅(如果没有加频道号，则取消所有)**

# 5、持久化

## 5.1、RDB方式

**RDB方式的持久化**是通过**快照**完成的。 当符 合**一定的条件**时Redis会自动将内存中的所有数据生成一份副本并保存在磁盘上，这个过程即为**快照**。(dump.rdb)

以下几种条件下会对数据进行快照：

1. 根据配置规则进行自动快照
2. 用户执行save或者bgsave命令
3. 执行flushall命令
4. 执行复制replication时

### 5.1.1、根据配置规则进行自动快照

在安装配置文件redis.conf中：

Save 900 1

Save 300 10

Save 60 1000

Save命令配置有俩个参数：save M N /M是时间，N是改动键的个数。

即：900秒内有一个或一个以上的键被更改了则进行快照

### 5.1.2、用户执行save或者bgsave命令

**1、save命令**

执行save命令时，redis同步的进行快照功能，会阻塞所有来自客户端的请求。尽量避免生产环境使用

**2、bgsave命令**

手动执行快照时，推荐使用bgsave命令。Bgsave命令可以在后台异步执行。可以通过lastsave命令来获取最近一次成功执行快照的时间，返回的结果是一个时间戳

### 5.1.3、执行flushall命令

执行该命令时，redis会清除数据库中的所有数据。只要自动快照条件不为空，redis都会执行一次快照操作。如果没有定义自动快照条件，则执行此命令是，不会进行快照。

### 5.1.4、执行复制时

执行复制操作时，无论有没有定义自动快照条件，或者无论有没有手动执行过快照操作，都会生成RDB快照文件

## 5.2、AOF方式

AOF（append only file）可以将redis执行的每一条写命令追加到硬盘文件中。(appendonly.aof)

### 5.2.1、开启AOF

Appendonly yes（在配置文件中可以修改）

AOF文件以纯文本的形式记录了Redis执行的写命令。

### 5.2.1、同步硬盘数据

Appendfsync always 表示每次写入都执行同步

Appendfsync everysec 表示每一秒执行一次同步

Appendfsync no 表示不主动执行同步(即每30秒执行一次)