# 一、简介

Redis是一个key-value存储系统。

和Memcached类似，它支持存储的value类型相对更多，

包括string(字符串)、list(链表)、set(集合)和zset(有序集合)。

这些数据类型都支持push/pop、add/remove及取交集并集和差集及更丰富的操作，而且这些操作都是原子性的。

Redis 是一个高性能的key-value数据库。

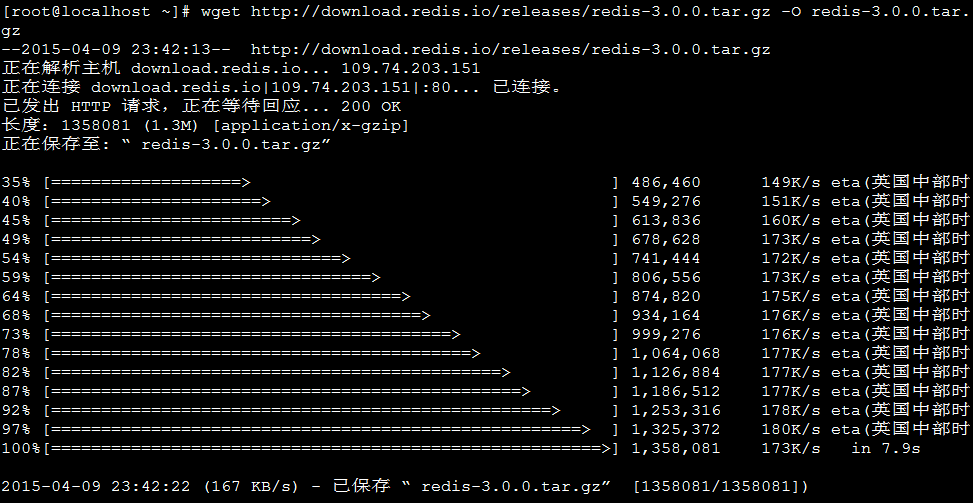
Redis的出现，很大程度补偿了memcached这类key/value存储的不足，在部 分场合可以对关系数据库起到很好的补充作用。

它提供了Python，Ruby，Erlang，PHP客户端，使用很方便。目前最新版本3.2.8

二、安装

1、默认安装位置

[root@localhost ~]# wget http://download.redis.io/releases/redis-3.2.8.tar.gz -O redis-3.2.8.tar.gz



下载过程

[root@localhost ~]# tar xzf redis-3.2.8.tar.gz

[root@localhost ~]# cd redis-3.2.8

[root@localhost ~]# make

报错解决：zmalloc.h:50:31: error: jemalloc/jemalloc.h: No such file or directory

zmalloc.h:55:2: error: #error "Newer version of jemalloc required"

报错则加选项MALLOC=libc

2、指定安装位置

[root@localhost ~]# tar xzf redis-3.2.8.tar.gz

[root@localhost ~]# cd redis-3.2.8

[root@localhost ~]# make

[root@localhost ~]# make install PREFIX=/usr/local/redis

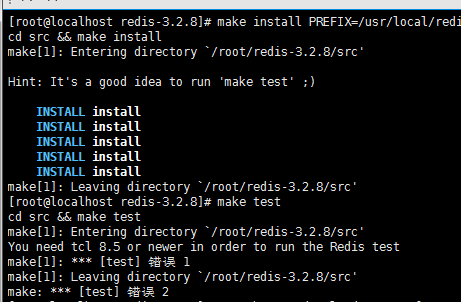
[root@localhost ~]# mkdir /usr/local/redis/etc

[root@localhost ~]# cp /root/redis-3.2.8/redis.conf /usr/local/redis/etc/

不指定安装位置，则会把redis的可执行文件安装到 redis-3.2.8/src/目录下

make install则会将常用脚本安装到指定位置，手动创建配置文件目录及拷贝配置文件

**在安装 让我们进行测试make test时候遇到问题,提示**



hadoop@stormspark:~/workspace/redis2.6.13/src$ make test

You need tcl 8.5 or newer in order to run the Redis test

make: \*\*\* [test] Error 1

**出现问题原因**

redis安装后，不需要安装tcl8.5也是可以使用的

但是要使用Redis的测试用例也就是tests目录下面用的是tcl脚本,就需要安装tcl8.5 or newer

**解决方式**

安装tcl

wget http://downloads.sourceforge.net/tcl/tcl8.6.1-src.tar.gz

sudo tar xzvf tcl8.6.1-src.tar.gz -C /usr/local/

cd /usr/local/tcl8.6.1/unix/

sudo ./configure

sudo make

sudo make install

3、安装的可执行文件的作用

redis-server 服务器端

redis-cli 客户端

redis-benchmark 调试

redis-check-dump 数据导出

redis-check-aof 数据导入

redis-setinel 监控功能（3.2.8新加）

三、启动与关闭

1、 启动

路径/redis-server 配置文件

/usr/local/redis/bin/redis-server /usr/local/redis/etc/redis.conf (这一行是一起打)

注意：需要修改配置文件

[root@localhost redis]# vi /usr/local/redis/etc/redis.conf

[

命令模式下 使用 /+后跟查找的字符串。vim会显示文本中第一个出现的字符串。

/daemonize

]

daemonize no 改为

daemonize yes #后台启动

重新启动redis 就在后台运行了.

/usr/local/redis/bin/redis-server /usr/local/redis/etc/redis.conf

netstat -ptlun |grep redis

端口 6379

[root@localhost redis]# /usr/local/redis/bin/redis-cli

#客户端连接

-h IP ： 连接指定的redis服务器

-p 6379： 指定redis服务器的端口

-a 密码： 使用密码登录

-n 数据库号： 指定连接哪个数据库

2、 关闭redis

[root@localhost ~]# /usr/local/redis/bin/redis-cli shutdown

或

[root@localhost ~]# pkill -9 redis

配置完成之后: 的开启 关闭

[root@localhost etc]# /usr/local/redis/bin/redis-server /usr/local/redis/etc/redis.conf

[root@localhost etc]# /usr/local/redis/bin/redis-cli

127.0.0.1:6379> quit

[root@localhost etc]# /usr/local/redis/bin/redis-cli shutdown

四、配置文件

详见《redis配置文件详解》

五、Redis的数据类型

1. 共计5种类型：string(字符串)、hash(哈希表) list(双向队列)、set(集合)和zset(有序集合)

2. String（子串类型）

String是最简单的类型，一个Key对应一个Value，string类型是二进制安全的。Redis的string可以包含任何数据，比如jpg图片或者序列化的对象。

--------------------------------------------

1）set 键 "值"

设置一个键和值，键存在则覆盖，返回ok

127.0.0.1:6379> set name liming

OK

2）get 键

获取一个键的值，返回值

127.0.0.1:6379> get name

"liming"

命令中的NX表示如果key不存在就添加，存在则直接返回

3） setnx 键 值

只有当该键不存在时设置一个键的值，若键已存在则返回0表示失败（防止覆盖），

127.0.0.1:6379> setnx age 18

(integer) 1

127.0.0.1:6379> setnx age 18

(integer) 0

3） setex 键 [有效时间] 值

设置一个指定有效期的键和值（单位秒）。不写有效时间则表示永久有效，等价于set

127.0.0.1:6379> setex movie 30 meigonghe (湄公河)

OK

127.0.0.1:6379> ttl movie //获取键的有效时间

(integer) 26

127.0.0.1:6379> ttl movie

(integer) 20

127.0.0.1:6379> get movie

"canglaoshi"

127.0.0.1:6379> ttl movie

(integer) -2

127.0.0.1:6379> get movie

(nil)

4）ttl 键

以秒为单位，返回给定 key 的剩余生存时间

当 key 不存在时，返回 -2 。

当 key 存在但没有设置剩余生存时间时，返回 -1 。

否则，以秒为单位，返回 key 的剩余生存时间。

5） setrange 键 位置 子字串

替换子字符串 (替换长度由子子串长度决定) abcdefg 从c开始替换,bad. Abbadfg

127.0.0.1:6379> set key1 "hello world"

OK

127.0.0.1:6379> get key1

"hello world"

127.0.0.1:6379> setrange key1 6 liming

(integer) 12

127.0.0.1:6379> get key1

"hello liming"

#将key1键对应值的第6个位置开始替换（字符串位置从0开始计算）

6） mset 键1 值1 键2 值2 键3 值3 ....

批量设置键和值,成功则返回ok

127.0.0.1:6379> mset name1 lm name2 sc name3 zjj

OK

7） mget 键1 键2 键3....

批量获取值

127.0.0.1:6379> mget name1 name2 name3

1) "lm"

2) "sc"

3) "zjj"

7） msetnx 键1 值1 键2 值2 键3 值3 ....

批量设置不存在的键和值(当键不存在时候才成功),成功则返回ok

127.0.0.1:6379> msetnx b1 1 b2 2 b3 3

(integer) 1

127.0.0.1:6379> mget b1 b2 b3

1) "1"

2) "2"

3) "3"

127.0.0.1:6379> msetnx b1 1 b2 2 b3 3

(integer) 0

9） getset 键 新值

获取原值，并设置新值

127.0.0.1:6379> set name "shen chao"

OK

127.0.0.1:6379> get name

"shen chao"

127.0.0.1:6379> getset name "liming"

"shen chao"

127.0.0.1:6379> get name

"liming"

10） getrange 键 0 4

获取指定范围的值（获取指定0到4位置上的值，字符串位置从0开始计算）

127.0.0.1:6379> getrange key1 0 4

"hello"

11） incr 键

指定键的值做加1操作，返回加后的结果（只能加数字）。

127.0.0.1:6379> set age 18

OK

127.0.0.1:6379> incr age

(integer) 19

127.0.0.1:6379> get age

"19"

12） incrby 键 m

//其中m可以是正整数或负整数

加指定值，键不存在时候会设置键

127.0.0.1:6379> incrby age 10

(integer) 29

127.0.0.1:6379> get age

"29"

127.0.0.1:6379> incrby age -5

(integer) 24

13） decr 键

指定键的值做减1操作，返回减后的结果。

127.0.0.1:6379> decr age

(integer) 23

127.0.0.1:6379> get age

"23"

14） decrby 键 n

//其中n可以是正整数或负整数

设置某个键减上指定值

127.0.0.1:6379> decrby age 5

(integer) 18

127.0.0.1:6379> decrby age -10

(integer) 28

127.0.0.1:6379> get age

"28"

15）append 键 追加字串

给指定key的字符串追加value，返回新字符串值的长度

127.0.0.1:6379> append name1 " have a new computer!"

(integer) 19

127.0.0.1:6379> get name1

"lm have a new computer!"

16） strlen 键名

strlen求一个键长度

127.0.0.1:6379> strlen name1

(integer) 19

17）del命令：删除一个键

127.0.0.1:6379> del name3

(integer) 1

127.0.0.1:6379> get name3

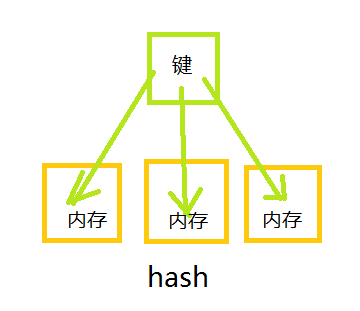
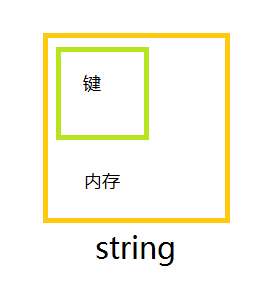
(nil)

3. hashs类型

#注意：redis中没有表概念，所有的数据都存入键中。

string键类型：所有的值（可以是任何数据类型）都保存在一个键当中，放在一个内存块中

hashs键类型：所有的值也都保存在一个键当中，只是放在不同的内存块中，每个块称作字段



1）hset key field value

设置一个键，在键中保存字段和值

hset 哈希集（键） 字段 值

127.0.0.1:6379> hset user1 name4 ysm

(integer) 1

127.0.0.1:6379> keys \* //查看库中所有的键

1) "aa"

2) "name"

3) "name1"

4) "user1"

5) "name2"

6) "age"

7) "key1"

2）>hsetnx 键 字段 值

设置一个键中，不存在的字段和值。如果字段存在则报错（成功返回1，失败返回0）

127.0.0.1:6379> hsetnx user1 name1 lm

(integer) 1

127.0.0.1:6379> hsetnx user1 name1 sc

(integer) 0 //报错

127.0.0.1:6379> hget user1 name1

"lm" //内容没有更新

3）hmset 键 字段1 值1 字段2 值2 …

在一个键中，批量设置字段

127.0.0.1:6379> hmset star name huanglei age 45 interest daughter

OK

4）hget 键 字段

获取键中的一个指定字段的值

127.0.0.1:6379> hget star name

"huanglei"

127.0.0.1:6379> hget star interest

"daughter"

5）hmget 键 字段1 [字段2…]

获取键中一个或多个字段的值

127.0.0.1:6379> hmget star name age interest

1) "huanglei"

2) "45"

3) "daughter"

6） hexists ：判断指定的字段是否存在于键中

例如：

127.0.0.1:6379> HEXISTS star age

(integer) 1 //存在

127.0.0.1:6379> HEXISTS user1 age

(integer) 0 //不存在

7） hlen ：获取键中的字段数量

127.0.0.1:6379> hlen star

(integer) 3 //user2键中有3个字段

8）hkeys ：获取键中的所有字段名

127.0.0.1:6379> hkeys star

1) "name"

2) "age"

3) "interest"

9）hvals：获取键中所有字段的值

127.0.0.1:6379> hvals star

1) "liming"

2) "36"

3) "AV-Girl"

10） hgetall ：获取键中的所有字段和值

127.0.0.1:6379> hgetall star

1) "name"

2) "huanglei"

3) "age"

4) "45"

5) "interest"

6) "daughter"

11）hincrby：将键中指定字段的值，增加指定的数字

127.0.0.1:6379> hincrby star age 5

(integer) 41

127.0.0.1:6379> hincrby star name 5

(error) ERR hash value is not an integer //值不是数字的字段，不能加数字

12）hdel 键 字段1 字段2

删除键中的一个或多个字段

127.0.0.1:6379> hdel star age interest

(integer) 2

127.0.0.1:6379> hkeys star

1) "name"

//删除一个键，还是要使用del命令

4. list类型（双向链表结构）

List是一个链表结构，主要功能是push、pop、获取一个范围的所有值等等，操作中key理解为链表的名字。Redis的list类型其实就是一个 每个子元素都是string类型的双向链表。列表允许有重复值

1）lpush 键 值1 [值2…]

从队列左边向队列写入一个或多个值（认为队列的左面为队列头，右边为队列尾）

127.0.0.1:6379> lpush list1 1

(integer) 1

127.0.0.1:6379> lpush list1 2

(integer) 2

127.0.0.1:6379> lpush list1 3

(integer) 3

127.0.0.1:6379> lpush list1 4

(integer) 4

127.0.0.1:6379> lpush list2 one two three four

(integer) 4

2）lrange 键 起始下标 终止下标

从队列中获取指定的返回值（从队列左边向右获取）

下标：0代表队列中第一个元素，1代表第二个元素，依次类推

-1代表队列中最后一个元素，-2代表倒数第二个元素

127.0.0.1:6379> LRANGE list1 0 -1

1) "4" //4是从左面写入队列的最后一个值，所以在队列的开头

2) "3"

3) "2"

4) "1" //1是从左面写入队列的第一个值，所以直接放到了队列尾。

127.0.0.1:6379> LRANGE list2 0 -1

1) "four"

2) "three"

3) "two"

4) "one"

127.0.0.1:6379> LRANGE list1 0 1

1) "4"

2) "3"

127.0.0.1:6379> LRANGE list2 -4 3

1) "four" //-4代表从队列右边数第四个元素

2) "three"

3) "two"

4) "one" //3代表从队列左边数第四个元素

3）rpush 键 值1 [值2…]

从队列右边向队列写入一个或多个值

127.0.0.1:6379> RPUSH list3 1 2 3 4

(integer) 4

127.0.0.1:6379> LRANGE list3 0 -1

1) "1" //从队列右边向队列写入值，第一个值就会写到队列的开头

2) "2"

3) "3"

4) "4"

4）linsert 键 before|after 原值 新值

在队列中指定元素之前或之后插入新值

127.0.0.1:6379> LINSERT list3 before 3 hello

(integer) 5

127.0.0.1:6379> LRANGE list3 0 -1

1) "1"

2) "2"

3) "hello"

4) "3"

5) "4"

5）lset 键 下标 新值

给队列中指定元素设定新值

127.0.0.1:6379> lset list3 2 "5"

OK

127.0.0.1:6379> LRANGE list3 0 -1

1) "1"

2) "2"

3) "5"

4) "3"

5) "4"

6）lrem 键 n 指定值

从队列中删除n个值为“指定值”的元素

n > 0 从队列头向尾删除n个元素

n < 0 从队列尾向头删除n个元素

n = 0 删除所有值为“指定值”的元素

127.0.0.1:6379> rpush list4 hello 1 hello 2 hello 3 hello

(integer) 7

127.0.0.1:6379> lrange list4 0 -1

1) "hello"

2) "1"

3) "hello"

4) "2"

5) "hello"

6) "3"

7) "hello"

127.0.0.1:6379> lrem list4 -2 hello //删除后两个hello

(integer) 2

127.0.0.1:6379> lrange list4 0 -1

1) "hello"

2) "1"

3) "hello"

4) "2"

5) "3"

127.0.0.1:6379> lrem list4 0 hello //删除所有hello

(integer) 2

127.0.0.1:6379> lrange list4 0 -1

1) "1"

2) "2"

3) "3"

下面希望同学们都能了解 起码有个印象

一定要自己敲2遍

7）ltrim 键 起始下标 结束下标

修剪队列，让队列只保留指定指定范围内的元素

127.0.0.1:6379> RPUSH list5 1 2 3 4

(integer) 4

127.0.0.1:6379> ltrim list5 1 2

OK

127.0.0.1:6379> lrange list5 0 -1

1) "2"

2) "3"

8）lpop 键

从指定的队列左面移除一个值

127.0.0.1:6379> lrange list1 0 -1

1) "4"

2) "3"

3) "2"

4) "1"

127.0.0.1:6379> lpop list1

"4"

127.0.0.1:6379> lrange list1 0 -1

1) "3"

2) "2"

3) "1"

9）rpop 键

从指定队列的右边移除一个值

127.0.0.1:6379> lrange list1 0 -1

1) "3"

2) "2"

3) "1"

127.0.0.1:6379> rpop list1

"1"

127.0.0.1:6379> lrange list1 0 -1

1) "3"

2) "2"

10）rpoplpush 源队列 目标队列

移除源队列的最后一个元素，并把该元素写入目标队列

127.0.0.1:6379> lrange list1 0 -1

1) "3"

2) "2"

127.0.0.1:6379> lrange list5 0 -1

1) "2"

2) "3"

127.0.0.1:6379> RPOPLPUSH list1 list5

"2"

127.0.0.1:6379> lrange list1 0 -1

1) "3"

127.0.0.1:6379> lrange list5 0 -1

1) "2"

2) "2"

3) "3"

11）lindex 键 下标

获取队列中指定下标元素的值

127.0.0.1:6379> lrange list2 0 -1

1) "four"

2) "three"

3) "two"

4) "one"

127.0.0.1:6379> lindex list2 1

"three"

127.0.0.1:6379> lindex list2 3

"one"

12）llen 键

获得队列的长度

127.0.0.1:6379> llen list2

(integer) 4

5.sets类型和操作

Set是集合，它是string类型的无序集合。对集合我们可以取并集、交集、差集。通过这些操作我们可以实现社交网站中的好友推荐和blog的tag功能。集合不允许有重复值。

1） sadd 键 值1[值2…]

添加一个或多个元素到集合中

127.0.0.1:6379> sadd mset1 1

(integer) 1

127.0.0.1:6379> sadd mset1 2 3 4

(integer) 3

2） smembers 键

获取集合里面所有的元素

127.0.0.1:6379> smembers mset1

1) "1"

2) "2"

3) "3"

4) "4"

3） srem 键 值1[值2…]

从集合中删除指定的一个或多个元素

127.0.0.1:6379> srem mset1 3 4

(integer) 2

127.0.0.1:6379> smembers mset1

1) "1"

2) "2"

（删除键，依然使用“del 键” 命令）

4） spop 键

随机从集合中删除一个元素，并返回

127.0.0.1:6379> sadd mset2 4 5 6 7 8

(integer) 5

127.0.0.1:6379> spop mset2

"4"

127.0.0.1:6379> spop mset2

"5"

127.0.0.1:6379> spop mset2

"8"

127.0.0.1:6379> smembers mset2

1) "6"

2) "7"

5） srandmember 键 值

随机返回集合中一个元素，但不删除

127.0.0.1:6379> sadd mset3 4 5 6 7 8

(integer) 5

127.0.0.1:6379> srandmember mset3

"5"

127.0.0.1:6379> srandmember mset3

"5"

127.0.0.1:6379> srandmember mset3

"4"

6） scard 键

获取集合里面元素个数

127.0.0.1:6379> scard mset1

(integer) 2

7） sismember 键 值

确定一个指定的值是否是集合中的元素

127.0.0.1:6379> smembers mset1

1) "1"

2) "2"

127.0.0.1:6379> sismember mset1 3

(integer) 0

127.0.0.1:6379> sismember mset1 1

(integer) 1

8） sdiff 集合1 集合2

返回集合1与集合2的差集。以集合1为主

127.0.0.1:6379> sadd mset4 1 2 3

(integer) 3

127.0.0.1:6379> sadd mset5 2 3 4

(integer) 3

127.0.0.1:6379> sdiff mset4 mset5

1) "1"

9） sdiffstore 新集合 集合1 集合2

返回集合1和集合2的差集，并把结果存入新集合

127.0.0.1:6379> sadd mset4 1 2 3

(integer) 3

127.0.0.1:6379> sadd mset5 2 3 4

(integer) 3

127.0.0.1:6379> sdiffstore mset6 mset5 mset4

(integer) 1 //返回值为1 ，证明成功

127.0.0.1:6379> smembers mset6

1) "4" //结果存入了mset6，值为4

10） sinter 集合1 集合2

获得两个集合的交集

127.0.0.1:6379> smembers mset4

1) "1"

2) "2"

3) "3"

127.0.0.1:6379> smembers mset5

1) "2"

2) "3"

3) "4"

127.0.0.1:6379> sinter mset4 mset5

1) "2"

2) "3"

11） sinterstore 新集合 集合1 集合2

获得集合1和集合2的交集，并把结果存入新集合

127.0.0.1:6379> sinterstore mset7 mset4 mset5

(integer) 2

127.0.0.1:6379> smembers mset7

1) "2"

2) "3"

12） sunion 集合1 集合2

获得指定集合的并集

127.0.0.1:6379> sunion mset4 mset5

1) "1"

2) "2"

3) "3"

4) "4"

13） sunionstore 新集合 集合1 集合2

获得指定集合的并集，并把结果保存如新集合

127.0.0.1:6379> sunionstore mset8 mset4 mset5

(integer) 4

127.0.0.1:6379> smembers mset8

1) "1"

2) "2"

3) "3"

4) "4"

14） smove 源集合 目标集合 值

将指定的值从源集合移动到目标集合

127.0.0.1:6379> smembers mset1

1) "1"

2) "2"

127.0.0.1:6379> smembers mset2

1) "6"

2) "7"

127.0.0.1:6379> smove mset1 mset2 1

(integer) 1

127.0.0.1:6379> smembers mset1

1) "2"

127.0.0.1:6379> smembers mset2

1) "1"

2) "6"

3) "7"

6. sorted sets类型和操作(zset)

sorted set是set的一个升级版本，它给集合中每个元素都定义一个分数，集合中的元素按照其分数排序。也不允许有重复值

1） zadd 键 分数1 值1 [分数2 值2…]

该命令添加指定的成员到key对应的有序集合中，每个成员都有一个分数。你可以指定多个分数/成员组合。如果一个指定的成员已经在对应的有序集合中了，那么其分数就会被更新成最新的，并且该成员会重新调整到正确的位置，以确保集合有序。分数的值必须是一个表示数字的字符串，并且可以是double类型的浮点数。

127.0.0.1:6379> zadd zset1 1 lm 2 sc 3 glf

(integer) 3

127.0.0.1:6379> zadd zset1 1 ymj

(integer) 1

2） zrange 集合 起始下标 截止下标 [withscores]

返回有序集合中，指定区间内的成员。其中成员按照score（分数）值从小到大排序。具有相同score值的成员按照字典顺序来排列。

起始下标与截止下标和list类型一致：

0代表队列中第一个元素，1代表第二个元素，依次类推

-1代表队列中最后一个元素，-2代表倒数第二个元素

withscores：返回集合中元素的同时，返回其分数（score）

127.0.0.1:6379> zrange zset1 0 -1 withscores

1) "lm"

2) "1"

3) "ymj"

4) "1"

5) "sc"

6) "2"

7) "glf"

8) "3"

3） zrevrange 集合 起始下标 截止下标 [withscores]

返回有序集合中，指定区间的成员。其成员按照score从大到小来排列。

127.0.0.1:6379> zrevrange zset1 0 -1 withscores

1) "glf" //下标为0

2) "3"

3) "sc" //下标为1

4) "2"

5) "ymj" //下标为2

6) "1"

7) "lm" //下标为3

8) "1"

127.0.0.1:6379> zrevrange zset1 1 2 withscores //查看集合中下标是1-2的值

1) "sc"

2) "2"

3) "ymj"

4) "1"

4） zrangebyscore 集合 起始分数 截止分数 withscores

返回有序集合中score（分数）在指定区间的值

127.0.0.1:6379> zadd zset2 1 one 2 two 3 three 4 four

(integer) 4

127.0.0.1:6379> zrange zset2 0 -1 withscores //按照下标区间返回值

1) "one"

2) "1"

3) "two"

4) "2"

5) "three"

6) "3"

7) "four"

8) "4"

127.0.0.1:6379> zrangebyscore zset2 2 3 withscores //按照分数区间返回值

1) "two"

2) "2"

3) "three"

4) "3"

5） zrem 集合 值1 [值2…]

删除有序集合中指定的值

127.0.0.1:6379> zrem zset1 lm

(integer) 1

127.0.0.1:6379> zrange zset1 0 -1 withscores

1) "ymj"

2) "1"

3) "sc"

4) "2"

5) "glf"

6) "3"

6） zincrby 集合 增量 值

给有序集合中指定值的成员的分数（score）值加上增量（increment）。如果集合中没有这个值，则给添加一个分数是increment的值。

127.0.0.1:6379> zincrby zset1 2 ymj //如果值存在，则在其分数上加增量

"3"

127.0.0.1:6379> zrange zset1 0 -1 withscores

1) "sc"

2) "2"

3) "glf"

4) "3"

5) "ymj"

6) "3"

127.0.0.1:6379> zincrby zset1 4 bro //如果值不存在，则加入值。并指定分数为增"4" 量

127.0.0.1:6379> zrange zset1 0 -1 withscores

1) "sc"

2) "2"

3) "glf"

4) "3"

5) "ymj"

6) "3"

7) "bro"

8) "4"

7） zrank 集合 值

返回有序集合中指定值的下标。值按照score从小到大排序

127.0.0.1:6379> zrank zset1 sc

(integer) 0

127.0.0.1:6379> zrank zset1 ymj

(integer) 2

8） zrevrank 集合 值

返回有序集合中指定值的下标，值按照score从大到小排序

127.0.0.1:6379> zrange zset1 0 -1 withscores

1) "sc"

2) "2"

3) "glf"

4) "3"

5) "ymj"

6) "3"

7) "bro"

8) "4"

127.0.0.1:6379> zrevrank zset1 ymj

(integer) 1

127.0.0.1:6379> zrevrank zset1 sc

(integer) 3

9） zcount 集合 起始分数 截止分数

返回有序集合中，score值在起始分数与截止分数之间的个数

127.0.0.1:6379> zrange zset2 0 -1 withscores

1) "one"

2) "1"

3) "two"

4) "2"

5) "three"

6) "3"

7) "four"

8) "4"

127.0.0.1:6379> zcount zset2 2 4

(integer) 3

10） zcard 集合

返回有序集合元素的个数

127.0.0.1:6379> zcard zset2

(integer) 4

11） zremrangebyrank 集合 起始下标 结束下标

删除有序集合中，下标在指定区间的元素

127.0.0.1:6379> zrange zset2 0 -1 withscores

1) "one"

2) "1"

3) "two"

4) "2"

5) "three"

6) "3"

7) "four"

8) "4"

127.0.0.1:6379> ZREMRANGEBYRANK zset2 0 1

(integer) 2

127.0.0.1:6379> zrange zset2 0 -1 withscores

1) "three"

2) "3"

3) "four"

4) "4"

12） zremrangebyscore 集合 起始分数 截止分数

删除有序集合中，分数在指定区间的元素

127.0.0.1:6379> zrange zset1 0 -1 withscores

1) "sc"

2) "2"

3) "glf"

4) "3"

5) "ymj"

6) "3"

7) "bro"

8) "4"

127.0.0.1:6379> ZREMRANGEBYSCORE zset1 2 3

(integer) 3

127.0.0.1:6379> zrange zset1 0 -1 withscores

1) "bro"

2) "4"

13）zinterstore 新集合 取交集的集合个数 集合1 集合2

取集合1和集合2的交集，并把结果保存到新集合中。在计算交集之前，需要指定计算交集的集合的个数。交集中，值的分数是多个集合中分数的和。

127.0.0.1:6379> zadd zset1 1 one 2 two 3 three 4 four

(integer) 4

127.0.0.1:6379> zadd zset2 2 two 3 three 4 four 5 five

(integer) 4

127.0.0.1:6379> ZINTERSTORE zset3 2 zset1 zset2

//有两个集合计算交集，所以集合个数是2

(integer) 3

127.0.0.1:6379> ZRANGE zset3 0 -1 withscores

1) "two"

2) "4" //分数是两个集合中two值的分数和

3) "three"

4) "6"

5) "four"

6) "8"

14） zunionstore 新集合 取并集的集合个数 集合1 集合2

取集合1和集合2的并集，并把结果保存到新集合中。在计算并集之前，需要指定计算并集的集合的个数。并集中，值的分数是多个集合中分数的和。

127.0.0.1:6379> zadd zset1 1 one 2 two 3 three 4 four

(integer) 4

127.0.0.1:6379> zadd zset2 2 two 3 three 4 four 5 five

(integer) 4

127.0.0.1:6379> ZUNIONSTORE zset4 2 zset1 zset2

(integer) 5

127.0.0.1:6379> ZRANGE zset4 0 -1 withscores

1) "one"

2) "1"

3) "two"

4) "4"

5) "five"

6) "5"

7) "three"

8) "6"

9) "four"

10) "8"

**个人推荐：**

**Hash类型：适合做数据缓存**

**List类型：适合做数组搜索**

**集合类型：适合做好友推送**

**有序集合类型：适合商品排序**

**问:**

**公司想要启用redis**

**首先,在服务器装redis. 记得更改配置**

**其次,安装php-redis**

**随后,你得知道基本的语句**