# Redis笔记

# Redis能干吗?

1. **内存存储,持久化**
2. **效率高,可以用于高速缓存**
3. **发布订阅**
4. **地图信息分析**
5. **计时器,计数器(浏览量)**

# Redis在liunx环境下的安装

## 把我们下载好的Redis压缩包，上传到Linux的 /mnt/ 文件目录下

## 安装gcc环境

yum install gcc-c++

## 进入到  /usr/local/redis-x.x.x/ 目录下，进行编译与安装

//进入到/usr/local/redis-5.0.0/ 文件目录下

[root@localhost mnt]# cd /usr/local/redis-5.0.0/

[root@localhost redis-5.0.0]# make //对解压后的文件进行编译

[root@localhost redis-5.0.0]# cd ./src //进入到 redis-5.0.0/src 文件目录下

[root@localhost src]# make install //进行redis安装

## 进入到/usr/local/bin目录下

mkdir config

cd /usr/local/bin/config

cp /usr/local/redis-5.0.7/redis.conf redis.conf

vi redis.conf

修改daemonize为yes

## 启动Redis

cd /usr/local/bin

redis-server config/redis.conf

## 连接Redis(本机)

redis-cli -p 6379

## 关闭Redis服务

redis-cli连接好后直接执行shutdown

# Redis的基础命令

## 选择数据库(默认16个数据库)

select 0

## 清空当前数据库

flushdb

## 清空所有数据库

FLUSHALL

## 查看当前库所有的Key

keys \*

## 设置key多久时间过期

expire key 时间(秒)

expire name 10

## 查看key还要多久过期

ttl key

ttl name

## 判断当前是否存在key

exists key

exists name

## 移除当前的key

move key 1(代表当前库)

move key name 1

## 查看key的类型

type key

# 4.Redis的数据类型

## 1.String(字符串)

#####################################################################

127.0.0.1:6379> set name shibc #设置值

OK

127.0.0.1:6379> get name #获得值

"shibc"

127.0.0.1:6379> keys \* #获得所有的key

1) "name"

127.0.0.1:6379> append name hello #追加字符串,如果当前key不存在,就相当于setkey

(integer) 10

127.0.0.1:6379> get name

"shibchello"

127.0.0.1:6379> strlen name #获取字符串的长度

(integer) 10

127.0.0.1:6379>

#####################################################################

127.0.0.1:6379> set views 0 #初始浏览量

OK

127.0.0.1:6379> get views

"0"

127.0.0.1:6379> incr views #自增! 浏览量+1 如果当前key不存在,就相当于setkey

(integer) 1

127.0.0.1:6379> get views

"1"

127.0.0.1:6379> decr views #自减1 浏览量-1 (integer) 0

127.0.0.1:6379> get views

"0"

127.0.0.1:6379> incrby views 10 #可以设置步长 自定增量 +10

(integer) 10

127.0.0.1:6379> get views

"10"

127.0.0.1:6379> decrby views 5 #可以设置步长 自定减量 -5

(integer) 5

127.0.0.1:6379> get views

"5"

#####################################################################

#字符串范围 range

127.0.0.1:6379> set key hello,shibc

OK

127.0.0.1:6379> get key

"hello,shibc"

127.0.0.1:6379> getrange key 0 3 #截取字符串[0,3]

"hell"

127.0.0.1:6379> getrange key 0 -1 #获取全部的字符串

"hello,shibc"

#替换

127.0.0.1:6379> set key1 abcdefg

OK

127.0.0.1:6379> get key1

"abcdefg"

127.0.0.1:6379> setrange key1 1 xx #替换字符串

(integer) 7

127.0.0.1:6379> get key1

"axxdefg"

#####################################################################

#setex #设置过期时间

#setnx #不存在再设置(在分布式锁中会常常使用)

127.0.0.1:6379> setex key2 60 yaoguifeng #设置key2的值为yaoguifeng,60秒后过期

OK

127.0.0.1:6379> ttl key2

(integer) 55

127.0.0.1:6379> get key2

"yaoguifeng"

127.0.0.1:6379> setnx mykey redis #如果mykey不存在,创建mykey

(integer) 1

127.0.0.1:6379> get mykey

"redis"

127.0.0.1:6379> setnx mykey mysql #如果mykey存在,创建mykey失败

(integer) 0

127.0.0.1:6379> get mykey

"redis"

#####################################################################

#####################################################################

mset

mget

127.0.0.1:6379> mset key1 v1 key2 v2 key3 v3 #同时设置多个key

OK

127.0.0.1:6379> keys \*

1) "key2"

2) "key3"

3) "key1"

127.0.0.1:6379> mget key1 key2 key3 #同时获取多个key

1) "v1"

2) "v2"

3) "v3"

127.0.0.1:6379> msetnx key1 v1 key4 v4 #msetnx是一个原子性操作,要么一起成功,要么一起失败

(integer) 0

#####################################################################

#对象

set user:1 {name:zhangsan,age:3}

#这里的key是一个巧妙的设计: user:{id}:{filed} 如此设计在Redis中晚期ok!

127.0.0.1:6379> mset user:!:name zhangsan user:1:age 8

OK

127.0.0.1:6379> mget user:!:name user:1:age

1) "zhangsan"

2) "8"

#####################################################################

getset

127.0.0.1:6379> getset db redis #如果不存在值,则返回(nil)

(nil)

127.0.0.1:6379> getset db mongodb #如果存在值,获取原来的值.并设置新的值

"redis"

#####################################################################

## List(列表)

#####################################################################

127.0.0.1:6379> lpush list one two three #将一个值或者多个值插入列表头部(左)

(integer) 3

127.0.0.1:6379> lrange list 0 -1 #获取list中的值

1) "three"

2) "two"

3) "one"

127.0.0.1:6379> lrange list 0 1 #通过区间获取list中的值

127.0.0.1:6379> lpush list four

(integer) 4

127.0.0.1:6379> rpush list five #将一个值或者多个值插入列表尾部(右)

(integer) 5

127.0.0.1:6379> lrange list 0 -1

1) "four"

2) "three"

3) "two"

4) "one"

5) "five"

#####################################################################

lpop

rpop

127.0.0.1:6379> lrange list 0 -1

1) "four"

2) "three"

3) "two"

4) "one"

5) "five"

127.0.0.1:6379> lpop list #移除list的第一个元素

"four"

127.0.0.1:6379> rpop list #移除list的最后一个元素

"five"

127.0.0.1:6379> lrange list 0 -1

1) "three"

2) "two"

3) "one"

#####################################################################

lindex

127.0.0.1:6379> lrange list 0 -1

1) "three"

2) "two"

3) "one"

127.0.0.1:6379> lindex list 0 #通过下标获取list的某个值

"three"

127.0.0.1:6379> lindex list 1

"two"

127.0.0.1:6379>

#####################################################################

Llen

127.0.0.1:6379> llen list #获取list的长度

(integer) 3

#####################################################################

移除指定的值

Lrem

127.0.0.1:6379> lrange list 0 -1

1) "three"

2) "two"

3) "one"

127.0.0.1:6379> llen list

(integer) 3

127.0.0.1:6379> lrem list 1 one #移除list集合中指定个数的值,精确匹配

(integer) 1

127.0.0.1:6379> lrange list 0 -1

1) "three"

2) "two"

#####################################################################

trim 修剪

127.0.0.1:6379> lpush mylist hello1 hello 2 hello3 hello4

(integer) 5

127.0.0.1:6379> lrange mylist 0 -1

1) "hello4"

2) "hello3"

3) "2"

4) "hello"

5) "hello1"

127.0.0.1:6379> ltrim mylist 0 2 #通过下标截取指定的长度,这个list已经改变,只剩下截取的

OK

127.0.0.1:6379> lrange mylist 0 -1

1) "hello4"

2) "hello3"

3) "2"

#####################################################################

rpoplpush 移除列表的最后一个元素,将它移动到新的列表中

127.0.0.1:6379> lpush mylist hello1 hello2 hello3

(integer) 3

127.0.0.1:6379> lrange mylist 0 -1

1) "hello3"

2) "hello2"

3) "hello1"

127.0.0.1:6379> rpoplpush mylist otherlist #移除列表的最后一个元素,将它移动到新的列表中

"hello1"

127.0.0.1:6379> lrange mylist 0 -1

1) "hello3"

2) "hello2"

127.0.0.1:6379> lrange otherlist 0 -1

1) "hello1"

#####################################################################

lset 将列表中指定下标的值替换为另外一个值,更新操作

127.0.0.1:6379> lpush list hello

(integer) 1

127.0.0.1:6379> lrange list 0 -1

1) "hello"

127.0.0.1:6379> lset list 0 item #如果下标存在,则会更新当前下标的值

OK

127.0.0.1:6379> lrange list 0 -1

1) "item"

127.0.0.1:6379> lset list 1 item #如果下标不存在,更新会报错

(error) ERR index out of range

#####################################################################

linsert #将某个具体的值插入到列表中某个元素的前面或者后面

127.0.0.1:6379> lpush list hello word

(integer) 2

127.0.0.1:6379> lrange list 0 -1

1) "word"

2) "hello"

127.0.0.1:6379> linsert list before word other

(integer) 3

127.0.0.1:6379> lrange list 0 -1

1) "other"

2) "word"

3) "hello"

#####################################################################

## Set(集合)

#####################################################################

127.0.0.1:6379> sadd myset hello shibc yaogf #set集合中添加元素

(integer) 3

127.0.0.1:6379> sadd myset loveygf #set集合中添加元素

(integer) 1

127.0.0.1:6379> smembers myset #查看指定set的所有值

1) "loveygf"

2) "yaogf"

3) "hello"

4) "shibc"

127.0.0.1:6379> sismember myset hello #判断某一个值是不是在set集合中

(integer) 1

127.0.0.1:6379> sismember myset word

(integer) 0

#####################################################################

127.0.0.1:6379> scard myset #获取set集合中内容元素的个数

(integer) 4

#####################################################################

rem

127.0.0.1:6379> srem myset hello #移除set集合中的指定元素

(integer) 1

127.0.0.1:6379> smembers myset

1) "loveygf"

2) "yaogf"

3) "shibc"

#####################################################################

127.0.0.1:6379> SRANDMEMBER myset

"loveygf"

127.0.0.1:6379> SRANDMEMBER myset #随机抽取某个元素

"shibc"

#####################################################################

127.0.0.1:6379> spop myset #随机删除某个元素

"shibc"

127.0.0.1:6379> spop myset

"loveygf"

#####################################################################

将一个指定的值,移动到另外一个set集合

127.0.0.1:6379> sadd myset hello word shibc

(integer) 3

127.0.0.1:6379> smembers myset

1) "loveygf"

2) "word"

3) "hello"

4) "shibc"

127.0.0.1:6379> sadd myset1 loveshibc

(integer) 1

127.0.0.1:6379> smembers myset1

1) "loveshibc"

127.0.0.1:6379> smove myset myset1 loveygf #将一个指定的值,移动到另外一个set集合

(integer) 1

127.0.0.1:6379> smembers myset1

1) "loveygf"

2) "loveshibc"

127.0.0.1:6379> smembers myset

1) "word"

2) "hello"

3) "shibc"

#####################################################################

数字集合

1. 差集
2. 交集
3. 并集

127.0.0.1:6379> sadd key1 a b c

(integer) 3

127.0.0.1:6379> sadd key2 c d e

(integer) 3

127.0.0.1:6379> sdiff key1 key2 #差集

1) "a"

2) "b"

127.0.0.1:6379> sinter key1 key2 #交集

1) "c"

127.0.0.1:6379> sunion key1 key2 #并集

1) "a"

2) "b"

3) "c"

4) "d"

5) "e"

#####################################################################

## Hash(哈希)

#####################################################################

127.0.0.1:6379> hset myhash field1 hello field2 word #set 具体的key-value

(integer) 2

127.0.0.1:6379> hget myhash field1 #get 具体的key-value

"hello"

127.0.0.1:6379> hgetall myhash #get所有的key-value

1) "field1"

2) "hello"

3) "field2"

4) "word"

127.0.0.1:6379> hdel myhash field1 #删除某个key-value

(integer) 1

127.0.0.1:6379> hgetall myhash

1) "field2"

2) "word"

127.0.0.1:6379> hlen myhash #获取hash的字段数量

(integer) 1

127.0.0.1:6379> hexists myhash field2 #判断hash中指定字段是否存在

(integer) 1

127.0.0.1:6379> hexists myhash field3

(integer) 0

127.0.0.1:6379> hkeys myhash #获取所有的key

1) "field2"

127.0.0.1:6379> hvals myhash #获取所有的value

1) "word"

#####################################################################

127.0.0.1:6379> hset myhash field5 5

127.0.0.1:6379> hincrby myhash field5 1 #可以设置步长 自定增量 +1

127.0.0.1:6379> hsetnx myhash filed6 sss #如果不存在则set成功

(integer) 1

127.0.0.1:6379> hsetnx myhash filed6 bb #如果存在则set失败

(integer) 0

#####################################################################

## 5.Zset(有序集合)

127.0.0.1:6379> zadd myset 1 one #添加一个值

(integer) 1

127.0.0.1:6379> zadd myset 2 two 3 three #添加多个值

(integer) 2

127.0.0.1:6379> zrange myset 0 -1 #查看所有的值

1) "one"

2) "two"

3) "three"

#####################################################################

127.0.0.1:6379> zadd salary 500 shibicheng 1000 yaoguifeng 200000 zhangsan

(integer) 3

127.0.0.1:6379> zrangebyscore salary -inf +inf #显示全部的用户,从小到大

1) "shibicheng"

2) "yaoguifeng"

3) "zhangsan"

127.0.0.1:6379> zrevrange salary 0 -1 #显示全部的用户,从大到小

1) "yaoguifeng"

2) "shibicheng"

127.0.0.1:6379> zrangebyscore salary -inf +inf withscores #显示全部的用户且附带成绩

1) "shibicheng"

2) "500"

3) "yaoguifeng"

4) "1000"

5) "zhangsan"

6) "200000"

127.0.0.1:6379> zrangebyscore salary -inf 2500 withscores #显示工资小于2500的用户且附带成绩

1) "shibicheng"

2) "500"

3) "yaoguifeng"

4) "1000"

#####################################################################

#移除元素zrem

127.0.0.1:6379> zrem salary zhangsan #移除元素

(integer) 1

127.0.0.1:6379> zrange salary 0 -1

1) "shibicheng"

2) "yaoguifeng"

#####################################################################

127.0.0.1:6379> zcard salary #获取有序集合的个数

(integer) 2

127.0.0.1:6379> zadd myset 1 shibc 2 yaoguifeng 3 zhangsan

(integer) 3

127.0.0.1:6379> zcount myset 1 2 #获取指定区间的成员个数

(integer) 2

127.0.0.1:6379> zcount myset 1 3

(integer) 3