Redis

讲师：王洪利

Redis

1. Redis是REmote DIctionary Server的缩写,是一个key- value存储系统.
2. Redis提供了一些丰富的数据结构,包括Strings,Lists, Hashes,Sets和Ordered Sets以及Hashes.包括对这些数据结 构的操作支持.
3. Redis可以替代Memcached,并且解决了断电后数据完全丢 失的问题.
4. Redis官方网站: http://redis.io

Redis作者Blog: http://antirez.com

Redis优点

1.性能极高,Redis能支持10万每秒的读写频率.

2.丰富的数据类型及对应的操作.

3.Redis的所有操作都是原子性的,同时Redis还支持对几个操作全并后的原子性执行,也即支持事 务.

4.丰富的特性,Redis还支持publish/subscribe, key过期等特性.

Redis性能

以下摘自官方测试描述:

在50个并发的情况下请求10W次,写的速度是11W次/s,读的速度是8.1w次/s.

数据类型

作为Key-value型数据库，Redis也提供了键（key）和值（value）的映射关系。但是，除了常规的数值或字符串，Redis的键值还可以是以下形式之一：

lists（列表）

sets （集合）

sorted sets（有序集合）

hashes（哈希表）

键值的数据类型决定了该键值支持的操作。Redis支持如列表、集合或有序集合的交集、并集、差集等高级原子操作；同时，如果键值的类型是普通数字，Redis则提供自增等原子操作；

持久化

通常，Redis将数据存储于内存中，或被配置为使用虚拟内存。通过两种方式可以实现数据持久化使用截图的方式；将内存中的数据不断写入磁盘；或使用类似MySql的日志方式，记录每次更新的日志。前者性能较高，但是可能引起一定程度的数据丢失；后者相反。

适用场合

1.取最新N个数据的操作

2.排行榜应用，取TOP N操作

3.需要精准设定过期时间的应用

4.计数器应用

5.uniq操作，获取某段时间内所有数据排重

6.实时系统，所垃圾系统

7.pub/sub构建实时消息系统

8.构建队列系统

9.缓存

安装

Redis的官方下载站是http://redis.io/download

步骤一：下载Redis

步骤二：编译源程序

步骤三：启动Redis服务

步骤四：将Redis作为Linux服务随机启动

步骤五：客户端连接验证

步骤六：停止Redis实例

Redis数据类型及操作-string类型及操作

常用命令: set,get,decr,incr,mget 等.

应用场景: String是最常用的一种数据类型,普通的key/value存储.

实现方式:String在redis内部存储默认就是一个字符串,被redisObject所引用,当遇到incr, decr等操作时会转成数值型进行计算,此时redisObject的encoding字段为int.

Redis数据类型及操作-string类型及操作

1.set 设置key对应的值为string类型的value

set name wanghongli

2.setnx 如果key已经存在，返回0，nx是not exist的意思

set name wanghongli

3.setex 设置key对应的值为string类型的value，并指定此键值对应的有效期

set color 10 red

4.setrange设定key的value值进行替换

setrange name 8 gmail.com 从下标为8（包含8）的字符开始替换

Redis数据类型及操作-string类型及操作

5.mset 一次设置多个key的值，成功返回ok

mest key1 wanghongli1 key2 wanghongli2

6.msetnx 一次设置多个key的值，不会覆盖已经存在的key

msetnx key2 waghongli2_new key3 waghongli3

7.get 获取key对应的string值，如果不存在返回nil

get name

8.getset 设置key的值，并返回key的旧值，如果key不存在返回nil

getset name wanghonglinew

Redis数据类型及操作-string类型及操作

9.getrange 获取指定key的value值的子字符串

getrange name 0 6 字符串是从下标0开始的

getrange name -7 -1 字符串右面下标是从-1开始的

10.mget 一次获取多个key的值，如果对应key不存在，则对应返回nil

mget key1 key2 key3

Redis数据类型及操作-string类型及操作

11.incr 对key的值做加加操作，并返回新值。vlaue的类型为int

set age 20

incr age

get age

12.incrby 同incr类似，加指定值，key不存在的时候会设置key

incrby age 5

Redis数据类型及操作-string类型及操作

13.decr 对key的值做减减操作，decr一个不存在的key，则设置为 -1

set age 25

decr age

14.decrby 减指定的值

decrby age 5

Redis数据类型及操作-string类型及操作

15.append 给指定key的字符串值追加value，返回新字符串值的长度

set name wanghongli

append name @126.com

16.strlen 取指定key的value值长度

strlen name

Redis数据类型及操作-hashes类型及操作

常用命令: hget,hset,hgetall 等.

应用场景:比如,我们存储供应商酒店价格的时候可以采取此结构,用酒店编码作为Key, RatePlan+RoomType作为Filed,价格信息作为Value.

Redis数据类型及操作-hashes类型及操作

1.hset 设置 hash field 为指定值，如果key不存在，则先创建

hset myhash field1 hello

2.hsetnx 设置hash field 为指定值，如果key不存在，则先创建，如果已存在，返回0，nx是not exist的ujln

hsetnx myhash field “hello”

3.hmset 同时设置hash的多个field

hmset myhash field1 hello field2 world

Redis数据类型及操作-hashes类型及操作

4.hget 获取指定的hash field

```
hget myhash field1
```

5.hmget 获取全部指定的hash field

```
hmget myhash field1 field2 field3
```

6.hincrby 指定的hash field 加上给定的值

```
hset myhash field3 20
hincrby myhash field3 -8
hget myhash field3
```

Redis数据类型及操作-hashes类型及操作

7.hexists 测试指定的filed是否存在

hexists myhash field1

8. hlen 返回指定hash的field的数量

hlen myhash

9.hdel 删除指定hash的field

hdel myhash field1

Redis数据类型及操作-hashes类型及操作

10.hkeys 返回hash所有field

hkeys myhash

11.hvals 返回hash的所有value

hvals myhash

12.hgetall 获取某个hash中全部的filed及value

hgetall myhash

Redis数据类型及操作-list类型及操作

常用命令: lpush,rpush,lpop,rpop,lrang等.

应用场景:Redis list应用场景非常多,也是Redis最重要的数据结构之一,比如twitter的关注 列表,粉丝列表等都可以用Redis的list结构来实现.

Redis数据类型及操作-list类型及操作

1.lpush 在key对应list的头部添加字符串元素

```
lpush mylist world
lpush mylist hello
lrange mylist 0 -1
```

2. rpush 在key对应list的尾部添加字符串元素

```
rpush mylist2 hello
rpush mylist2 world
lrange mylist 0 -1
```

Redis数据类型及操作-list类型及操作

3.linsert 在key对应list的特定位置之前之添加字符串元素

redis 127.0.0.1:6379> rpush mylist3 "hello" (integer) 1
redis 127.0.0.1:6379> rpush mylist3 "world" (integer) 2
redis 127.0.0.1:6379> **linsert mylist3 before "world" "there"** (integer) 3
redis 127.0.0.1:6379> lrange mylist3 0 -1
1) "hello"
2) "there"
3) "world"
redis 127.0.0.1:6379>

Redis数据类型及操作-list类型及操作

4.lset 设置list中指定下标的元素值（下标从0开始）

```
redis 127.0.0.1:6379> rpush mylist4 "one" (integer) 1
redis 127.0.0.1:6379> rpush mylist4 "two"
(integer) 2
redis 127.0.0.1:6379> rpush mylist4 "three" (integer) 3
redis 127.0.0.1:6379> lset mylist4 0 "four" OK
redis 127.0.0.1:6379> lset mylist4 -2 "five" OK
redis 127.0.0.1:6379> lrange mylist4 0 -1
1) "four"
2) "five"
3) "three"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-list类型及操作

5.lrem

从key对应list中删除count个和value相同的元素

count>0时，按从头到尾的顺序删除

redis 127.0.0.1:6379> (integer) 1
redis 127.0.0.1:6379> (integer) 2
redis 127.0.0.1:6379> (integer) 3
redis 127.0.0.1:6379> (integer) 4
redis 127.0.0.1:6379> (integer) 2
redis 127.0.0.1:6379>
1) "foo"
2) "hello"

Redis数据类型及操作-list类型及操作

count<0时，按从尾到头的顺序删除

```
redis 127.0.0.1:6379> rpush mylist6 "hello" (integer) 1
redis 127.0.0.1:6379> rpush mylist6 "hello" (integer) 2
redis 127.0.0.1:6379> rpush mylist6 "foo" (integer) 3
redis 127.0.0.1:6379> rpush mylist6 "hello" (integer) 4
redis 127.0.0.1:6379> lrem mylist6 -2 "hello" (integer) 2
redis 127.0.0.1:6379> lrange mylist6 0 -1
1) "hello"
2) "foo"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-list类型及操作

6.ltrim保留指定key的值范围内的数据

redis 127.0.0.1:6379> rpush mylist8 "one" (integer) 1
redis 127.0.0.1:6379> rpush mylist8 "two" (integer) 2

redis 127.0.0.1:6379> rpush mylist8 "three" (integer) 3
redis 127.0.0.1:6379> rpush mylist8 "four" (integer) 4

redis 127.0.0.1:6379> **ltrim mylist8 1 -1** OK

redis 127.0.0.1:6379> lrange mylist8 0 -1

1) "two"
2) "three"
3) "four"
redis 127.0.0.1:6379>

Redis数据类型及操作-list类型及操作

7.lpop 从list的头部删除元素，并返回删除元素

```
redis 127.0.0.1:6379> lrange mylist 0 -1
1) "hello"
2) "world"
redis 127.0.0.1:6379> lpop mylist "hello"
redis 127.0.0.1:6379> lrange mylist 0 -1
1) "world"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-list类型及操作

8.rpop 从list的尾部删除元素，并返回删除元素

```
redis 127.0.0.1:6379> lrange mylist2 0 -1
1) "hello"
2) "world"
redis 127.0.0.1:6379> rpop mylist2 "world"
redis 127.0.0.1:6379> lrange mylist2 0 -1
1) "hello"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-list类型及操作

9.rpoplpush 从第一个list的尾部移除元素并添加到第二个list的头部，最后移除的元素值，整个操作是原子的。如果第一个list是空或者不存在返回nil

```
redis 127.0.0.1:6379> lrange mylist5 0 -1 1) "three"
2) "foo"
3) "hello"
redis 127.0.0.1:6379> lrange mylist6 0 -1 1) "hello"
2) "foo"
redis 127.0.0.1:6379> rpoplpush mylist5 mylist6 "hello"
redis 127.0.0.1:6379> lrange mylist5 0 -1
1) "three"
2) "foo"
redis 127.0.0.1:6379> lrange mylist6 0 -1
1) "hello"
2) "hello"
3) "foo"
```

Redis数据类型及操作-list类型及操作

10 lindex 返回名称为key的list中index位置的元素

```
redis 127.0.0.1:6379> lrange mylist5 0 -1
1) "three"
2) "foo"
redis 127.0.0.1:6379> lindex mylist5 0 "three"
redis 127.0.0.1:6379> lindex mylist5 1 "foo"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-list类型及操作

11 llen 返回key对应list的长度
redis 127.0.0.1:6379> llen mylist5 (integer) 2
redis 127.0.0.1:6379>

Redis数据类型及操作-set类型及操作

常用命令: sadd,spop,smembers,sunion 等.

应用场景:Set对外提供的功能与list类似,当你需要存储一个列表数据,又不希望出现重复 数据时,set 是一个很好的选择,并且set提供了判断某个成员是否在一个set集合 内的接口,这个也是list所不能提供的.

Redis数据类型及操作-sets类型及操作

1. sadd 向名称为key的set中添加元素

```
redis 127.0.0.1:6379> sadd myset "hello"
(integer) 1
redis 127.0.0.1:6379> sadd myset "world"
(integer) 1
redis 127.0.0.1:6379> sadd myset "world"
(integer) 0
redis 127.0.0.1:6379> smembers myset
1) "world"
2) "hello"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-sets类型及操作

2. srem 删除名称为key的set中的元素memeber

```
redis 127.0.0.1:6379> sadd myset2 “one”
(integer) 1
redis 127.0.0.1:6379> sadd myset2 “two”
(integer) 1
redis 127.0.0.1:6379> sadd myset2 “three”
(integer) 1
redis 127.0.0.1:6379> srem myset2 “one”
(integer) 1
redis 127.0.0.1:6379> 1) smembers myset2
1)”three”
2) "two"
```

Redis数据类型及操作-sets类型及操作

3. spop 随机返回并删除名称为key的set中一个元素

```
redis 127.0.0.1:6379> sadd myset3 "one"
(integer) 1
redis 127.0.0.1:6379> sadd myset3 "two"
(integer) 1
redis 127.0.0.1:6379> sadd myset3 "three"
(integer) 1
redis 127.0.0.1:6379> spop myset3
"three"
redis 127.0.0.1:6379> smembers myset3
1) "two"
2) "one"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-sets类型及操作

4.sdiff 返回所有给定key与第一个key的差集

```
redis 127.0.0.1:6379> smembers myset2
1) "three"
2) "two"
redis 127.0.0.1:6379> smembers myset3
1) "two"
2) "one"
redis 127.0.0.1:6379> sdiff myset2 myset3
1) "three"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-sets类型及操作

5.sdiffstore 返回所有给定key与第一个key的差集，并将结果为一个key

```
redis 127.0.0.1:6379> smembers myset2
1) "three"
2) "two"
redis 127.0.0.1:6379> smembers myset3
1) "two"
2) "one"
redis 127.0.0.1:6379> sdiffstore myset4 myset2 myset3
(integer) 1
redis 127.0.0.1:6379> smembers myset4
1) "three"
```

Redis数据类型及操作-sets类型及操作

6. sinter 返回所有给定key的交集

```
redis 127.0.0.1:6379> smembers myset2
1) "three"
2) "two"
redis 127.0.0.1:6379> smembers myset3
1) "two"
2) "one"
redis 127.0.0.1:6379> sinter myset2 myset3
1) "two"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-sets类型及操作

7.sinterstore 返回所有给定key的交集，并将结果存为另一个key

```
redis 127.0.0.1:6379> smembers myset2
1) "three"
2) "two"
redis 127.0.0.1:6379> smembers myset3
1) "two"
2) "one"
redis 127.0.0.1:6379> sinterstore myset5 myset2 myset3
(integer) 1
redis 127.0.0.1:6379> smembers myset5
1) "two"
```

Redis数据类型及操作-sets类型及操作

8.sunion 返回所有给定key的并集

```
redis 127.0.0.1:6379> smembers myset2
1) "three"
2) "two"
redis 127.0.0.1:6379> smembers myset3
1) "two"
2) "one"
redis 127.0.0.1:6379> sunion myset2 myset3
1) "three"
2) "one"
3) "two"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-sets类型及操作

9. 返回所有给定key的并集，并将结果存为另一个key

```
redis 127.0.0.1:6379> smembers myset2
1) "three"
2) "two"
redis 127.0.0.1:6379> smembers myset3
1) "two"
2) "one"
redis 127.0.0.1:6379> sunionstore myset6 myset2 myset3
(integer) 3
redis 127.0.0.1:6379> smembers myset6
1) "three"
2) "one"
3) "two"
```

Redis数据类型及操作-sets类型及操作

10. smove 从第一个key对应的set中移除member并添加到第二个对应set中

```
redis 127.0.0.1:6379> smembers myset2
1) "three"
2) "two"
redis 127.0.0.1:6379> smembers myset3
1) "two"
2) "one"
redis 127.0.0.1:6379> smove myset2 myset7 three
(integer) 1
redis 127.0.0.1:6379> smembers myset7
1) "three"
```

Redis数据类型及操作-sets类型及操作

11. scard 返回名称为key的set的元素个数

```
redis 127.0.0.1:6379> scard myset2
(integer) 1
redis 127.0.0.1:6379>
```

Redis数据类型及操作-sets类型及操作

13.srandmember 随机返回名称为key的set的一个元素，但是不删除元素

```
redis 127.0.0.1:6379> smembers myset3
1) "two"
2) "one"
redis 127.0.0.1:6379> srandmember myset3
"two"
redis 127.0.0.1:6379> srandmember myset3
"one"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-Sorted set类型及操作

常用命令: zadd,zrange,zrem,zcard等.

使用场景:Sorted set的使用场景与set类似,区别是set不是自动有序的,而sorted set可以 通过用户额外提供一个优先级(score)的参数来为成员排序,并且是插入有序的, 即自动排序.当你需要一个有序的并且不重复的集合列表,那么可以选择sorted set数据结构.

Redis数据类型及操作-Sorted set类型及操作

1.zadd 向名称为key的zset中添加元素member，score用于排序。如果该元素已经存在，则根据score更新该元素的顺序

```
redis 127.0.0.1:6379> zadd myzset 1 "one"
(integer) 1
redis 127.0.0.1:6379> zadd myzset 2 "two"
(integer) 1
redis 127.0.0.1:6379> zadd myzset 3 "two"
(integer) 0
redis 127.0.0.1:6379> zrange myzset 0 -1 withscores
1) "one"
2) "1"
3) "two"
```

Redis数据类型及操作-Sorted set类型及操作

2.zrem 删除名称为key的zset中的元素memeber

```
redis 127.0.0.1:6379> zrange myzset 0 -1 withscores
1) "one"
2) "1"
3) "two"
4) "3"
redis 127.0.0.1:6379> zrem myzset two
(integer) 1
redis 127.0.0.1:6379> zrange myzset 0 -1 withscores
1) "one"
2) "1"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-Sorted set类型及操作

3.zincreby 如果在名称为key的zset中已经存在元素member，则该元素的score增加increment；否则向集合中添加元素，其score的值为increment

```
redis 127.0.0.1:6379> zadd myzset2 1 "one" (integer) 1
redis 127.0.0.1:6379> zadd myzset2 2 "two" (integer) 1
redis 127.0.0.1:6379> zincrby myzset2 2 "one"
"3"
redis 127.0.0.1:6379> zrange myzset2 0 -1 withscores
1) "two"
2) "2"
3) "one"
4) "3"
```

Redis数据类型及操作-Sorted set类型及操作

4.zrank 返回名称为key的zset中member元素的排名（按score）从小到大排序即下标

```
redis 127.0.0.1:6379> zrange myzset3 0 -1 withscores
1) "one"
2) "1"
3) "two"
4) "2"
5) "three"
6) "3"
7) "five"
8) "5"
redis 127.0.0.1:6379> zrank myzset3 two
(integer) 1
```

Redis数据类型及操作-Sorted set类型及操作

5.zrevrank 返回名称为key的zset中member元素的排名（按score从大到小排序）即下标

```
redis 127.0.0.1:6379> zrange myzset3 0 -1 withscores
1) "one"
2) "1"
3) "two"
4) "2"
5) "three"
6) "3"
7) "five"
8) "5"
redis 127.0.0.1:6379> zrevrank myzset3 two
(integer) 2
```

Redis数据类型及操作-Sorted set类型及操作

6.zrevrange 返回名称为key的zset（按score从大到小排序）中的index从start到end的所有元素

```
redis 127.0.0.1:6379> zrevrange myzset3 0 -1 withscores
1) "five"
2) "5"
3) "three"
4) "3"
5) "two"
6) "2"
7) "one"
8) "1"
redis 127.0.0.1:6379>
```

Redis数据类型及操作-Sorted set类型及操作

7.zrangebyscore 返回集合score在给定区间的元素

```
redis 127.0.0.1:6379> zrange myzset3 0 -1 withscores
1) "one"
2) "1"
3) "two"
4) "2"
5) "three"
6) "3"
7) "five"
8) "5"
redis 127.0.0.1:6379> zrangebyscore myzset3 2 3 withscores
1) "two"
2) "2"
3) “three"
4) "3"
```

Redis数据类型及操作-Sorted set类型及操作

8.zcount 返回集合中score在给定区间的数量

```
redis 127.0.0.1:6379> zrange myzset3 0 -1 withscores
1) "one"
2) "1"
3) "two"
4) "2"
5) "three"
6) "3"
7) "five"
8) "5"
redis 127.0.0.1:6379> zcount myzset3 2 3
(integer) 2
```

Redis数据类型及操作-Sorted set类型及操作

9.zcard 返回集合中元素个数

```
redis 127.0.0.1:6379> zrange myzset3 0 -1 withscores
1) "one"
2) "1"
3) "two"
4) "2"
5) "three"
6) "3"
7) "five"
8) "5"
redis 127.0.0.1:6379> zcard myzset3
(integer) 4
```

Redis数据类型及操作-Sorted set类型及操作

10.zscore 返回给定元素对应的score

```
redis 127.0.0.1:6379> zrange myzset3 0 -1 withscores
1) "one"
2) "1"
3) "two"
4) "2"
5) "three"
6) "3"
7) "five"
8) "5"
redis 127.0.0.1:6379> zscore myzset3 two
"2"
```

Redis数据类型及操作-Sorted set类型及操作

11.zremrangebyrank 删除集合中排名在给定区间的元素

```
redis 127.0.0.1:6379> zrange myzset3 0 -1 withscores
1) "one"
2) "1"
3) "two"
4) "2"
5) "three"
6) "3"
7) "five"
8) "5"
```

Redis数据类型及操作-Sorted set类型及操作

```
redis 127.0.0.1:6379> zremrangebyrank myzset3 3 3
(integer) 1
redis 127.0.0.1:6379> zrange myzset3 0 -1 withscores
1) "one"
2) "1"
3) "two"
4) "2"
5) “three"
6）6
```

Redis数据类型及操作-Sorted set类型及操作

12.zremrangebyscore 删除集合中score在给定区间的元素

```
redis 127.0.0.1:6379> zrange myzset3 0 -1 withscores
1) "one"
2) "1"
3) "two"
4) "2"
5) "three"
6) "3"
redis 127.0.0.1:6379> zremrangebyscore myzset3 1 2
(integer) 2
redis 127.0.0.1:6379> zrange myzset3 0 -1 withscores
1) "three"
2) "3"
```

Redis常用命令

Redis 提供了丰富的命令(command)对数据库和各种数据类型进行操作,这些 command 可以在 Linux 终端使用。在编程时,比如各类语言包,这些命令都有对应的方法。下面将 Redis 提供的命令做一总结。

键值相关命令

1.keys 返回满足给定pattern 的所有key
redis 127.0.0.1:6379> keys *

用表达式 mylist*,代表取出所以
redis 127.0.0.1:6379> **keys mylist***，代表取出所有以**mylist**开头的**key**

键值相关命令

2.exists 确认一个key是否存在

```
redis 127.0.0.1:6379> exists HongWan
(integer) 0
redis 127.0.0.1:6379> exists age
 (integer) 1
redis 127.0.0.1:6379>
```

键值相关命令

3. del 删除一个吸

```
redis 127.0.0.1:6379> del age
 (integer) 1
redis 127.0.0.1:6379> exists age
(integer) 0
redis 127.0.0.1:6379>
```

键值相关命令

4. expire 设置一个key的过期时间（单位：秒）

redis 127.0.0.1:6379> **expire addr 10**
(integer) 1
redis 127.0.0.1:6379> ttl addr
(integer) 8
redis 127.0.0.1:6379> ttl addr
(integer) 1
redis 127.0.0.1:6379> ttl addr
(integer) -1
redis 127.0.0.1:6379>

ttl 来获取这个 key 的有效时长

键值相关命令

5.move 将当前数据库中的key转移到其它数据库中

```
redis 127.0.0.1:6379> select 0
OK
redis 127.0.0.1:6379> set age 30
OK
redis 127.0.0.1:6379> get age
"30"
redis 127.0.0.1:6379> move age 1
(integer) 1
redis 127.0.0.1:6379> get age (nil)
redis 127.0.0.1:6379> select 1
 OK
redis 127.0.0.1:6379[1]> get age
"30"
```

我先显式的选择了数据库 0,然后在这个库中设置一个 key,接下来我们将这个 key 从数据库 0 移到数据库 1,之后我们确认在数据库 0 中无此 key 了, 但在数据库 1 中存在 这个 key,说明我们转移成功了

键值相关命令

6.persist 移除给定key的过期时间

```
redis 127.0.0.1:6379[1]> expire age 300
(integer) 1
redis 127.0.0.1:6379[1]> ttl age
(integer) 294
redis 127.0.0.1:6379[1]> persist age
(integer) 1
redis 127.0.0.1:6379[1]> ttl age
(integer) -1
redis 127.0.0.1:6379[1]>
```

键值相关命令

7.randomkey 随机返回key空间的一个key
redis 127.0.0.1:6379> **randomkey**
"mylist7"
redis 127.0.0.1:6379> **randomkey**
"mylist5"
redis 127.0.0.1:6379>

键值相关命令

8.rename 重命名key

```
redis 127.0.0.1:6379[1]> keys *
1) "age"
redis 127.0.0.1:6379[1]> rename age age_new
OK
redis 127.0.0.1:6379[1]> keys *
1) "age_new"
redis 127.0.0.1:6379[1]>
```

键值相关命令

9. type 返回值的类型

```
redis 127.0.0.1:6379> type addr
string
redis 127.0.0.1:6379> type myzset2
zset
redis 127.0.0.1:6379> type mylist
 list
redis 127.0.0.1:6379>
```

扫描上面的二维码

关注兄弟连官方微信账号

兄弟连官方网址：www.lampbrother.net