零声学院 Mark老师 QQ: 2548898954

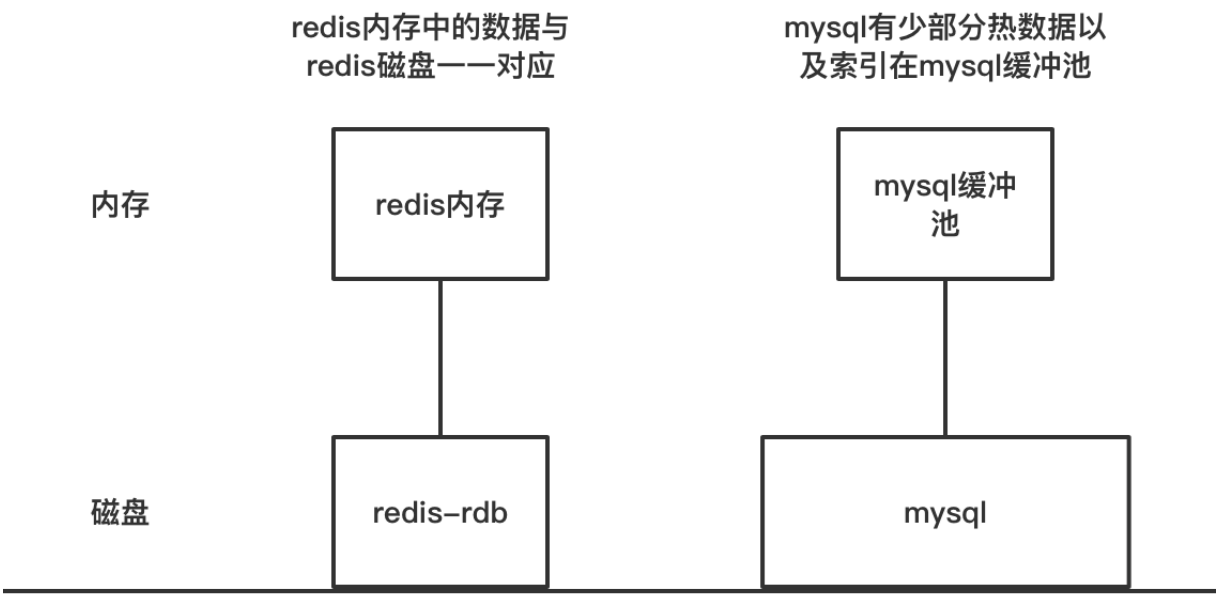
主题

- redis数据存储概述
- string结构以及应用
- list结构以及应用
- hash结构以及应用
- set结构以及应用
- zset结构以及应用

redis数据存储概述

什么是redis?

> redis是内存数据库、kv数据库、以及数据结构数据库



如何理解redis?

> 操作 redis 就相当于操作 unordered_map

```
template class<T>;
unordered_map<string, T>

unordered_map 和 map?
map 红黑树
unordered_map hashtable
```

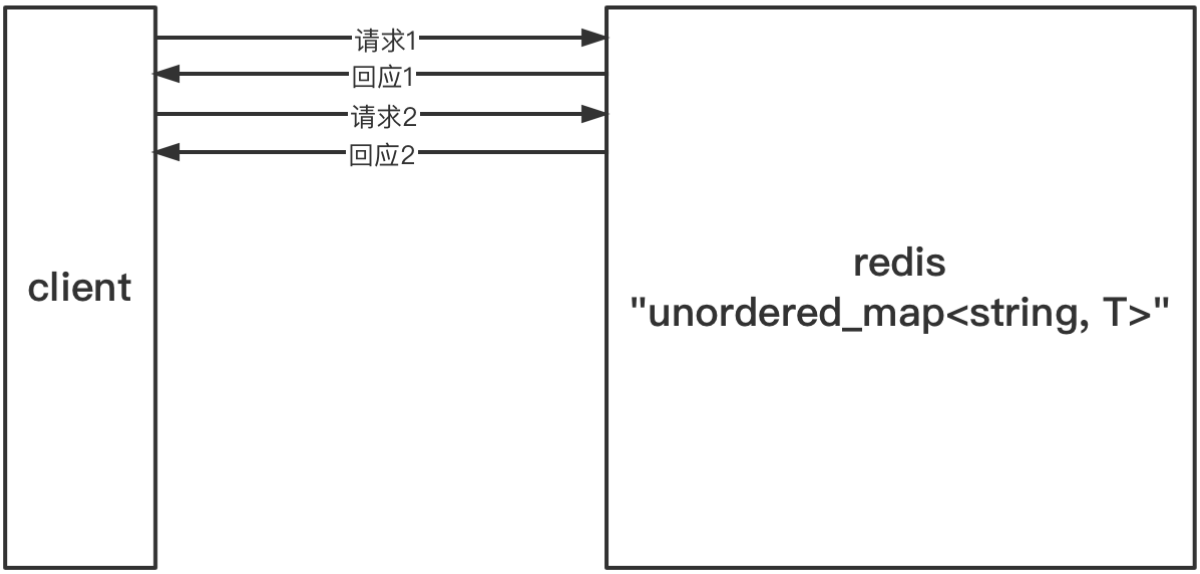
> T 支持 string，list， set， zset，hash

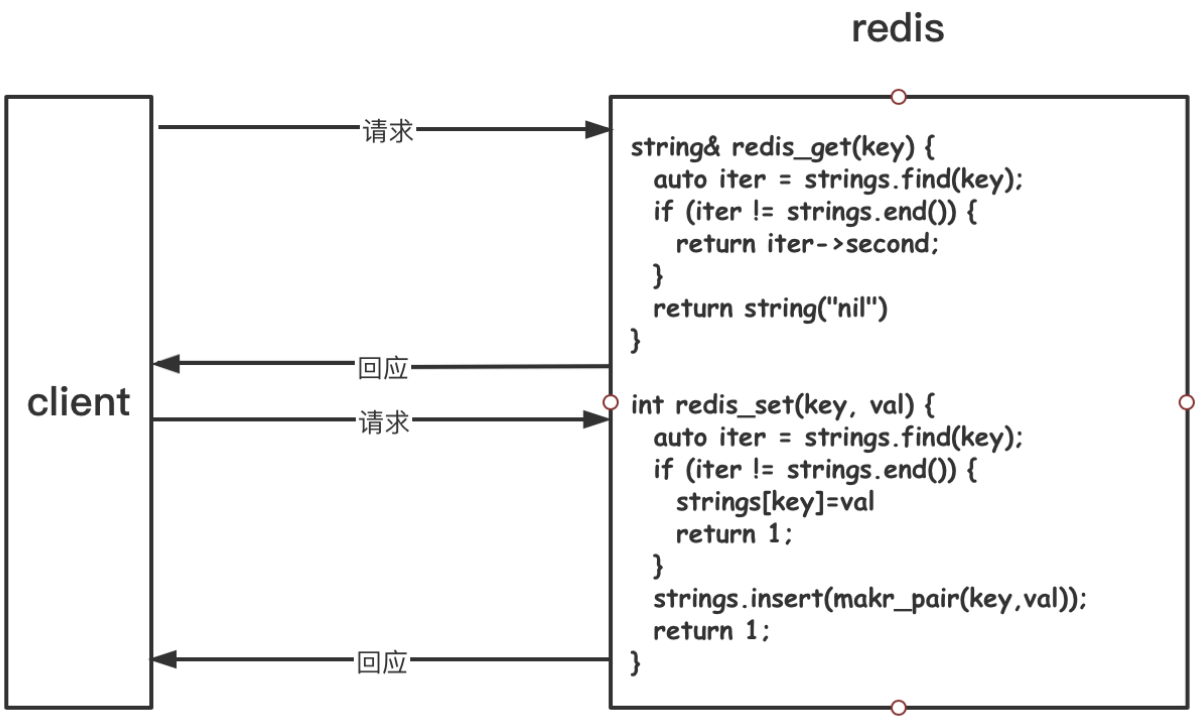
```
#include <unordered_map>
#include <unordered_set>
#include <set>
#include <string>
using namespace std;

// string
unordered_map<string, string> strings;

// list
unordered_map<string, list<string>> lists;

// set
unordered_map<string, unordered_set<string>> sets;

// zset
unordered_map<string, skiplist<string, string>> zsets;

// hash
unordered_map<string, unordered_map<string, string>> hashes;
```

> 要点：
>
> 1. redis中的字符串并非c结构中的字符串，它是一个二进制安全的字符串（不会被特殊字符（\0）隔断）;所以string中包含长度信息来标识字符串的长度；
> 2. map 采用红黑树（平衡二叉搜索树）实现，而 unordered_map 采用哈希表实现；更接近 redis dict 的实现；redis 中的 dict 也是采用哈希表实现的；
> 3. 红黑树是有序的树结构，查找需要比较 key，时间复杂度为 o(logn)；而哈希表是无序的，查找不需要比较 key，因为它先将 key 通过 hash 函数生成整数，然后映射到数组当中，它的时间复杂度为o(1)；
> 4. set 一般采用有序的结构来实现，因为集合中可能涉及到交、并、差集的运算（思考为什么这些运算要求结构有序？）；
> 5. skiplist 是一个多层级的有序链表，并且方便进行范围查询；与B+树实现的功能类似，但比B+效率要高；

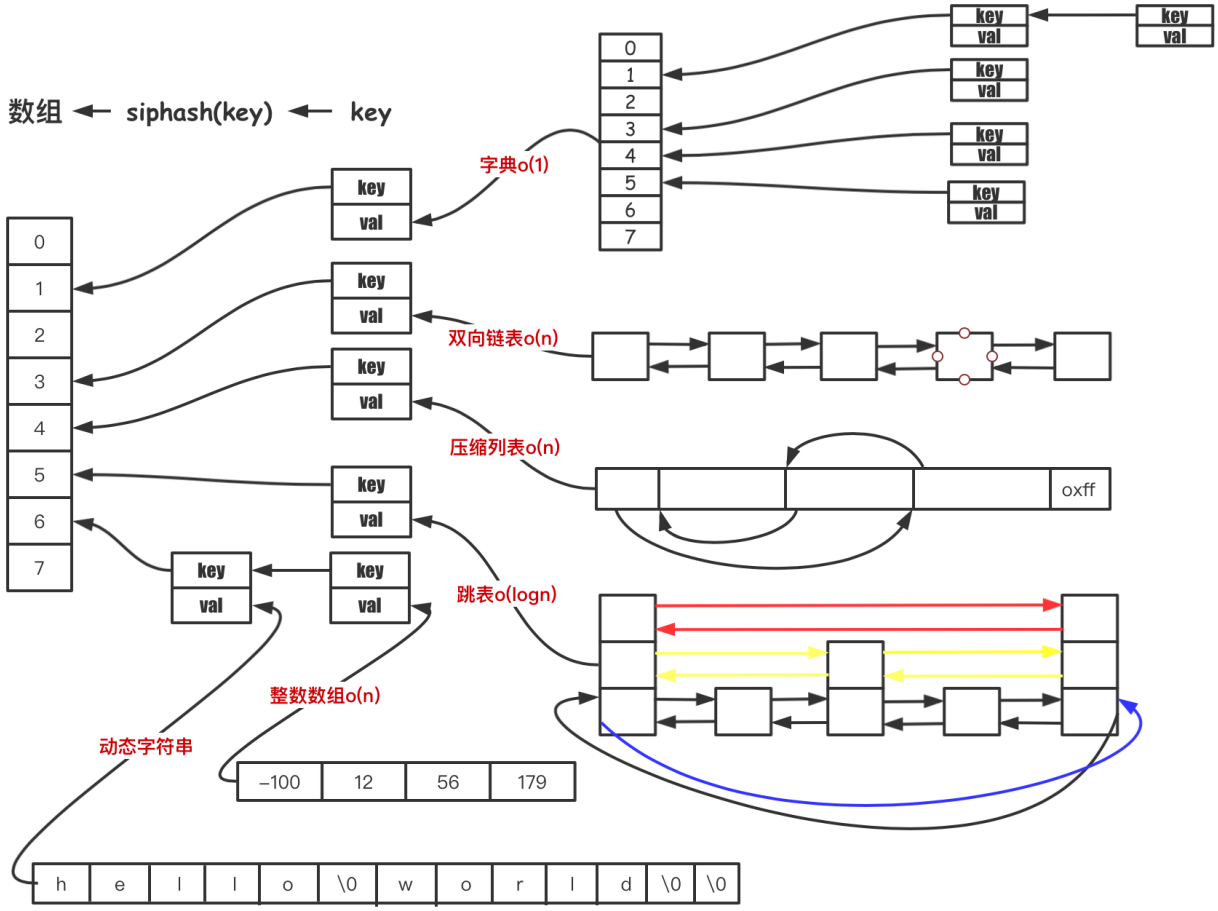
如何操作redis?

redis采用RESP协议，客户通过请求回应的方式来操作redis

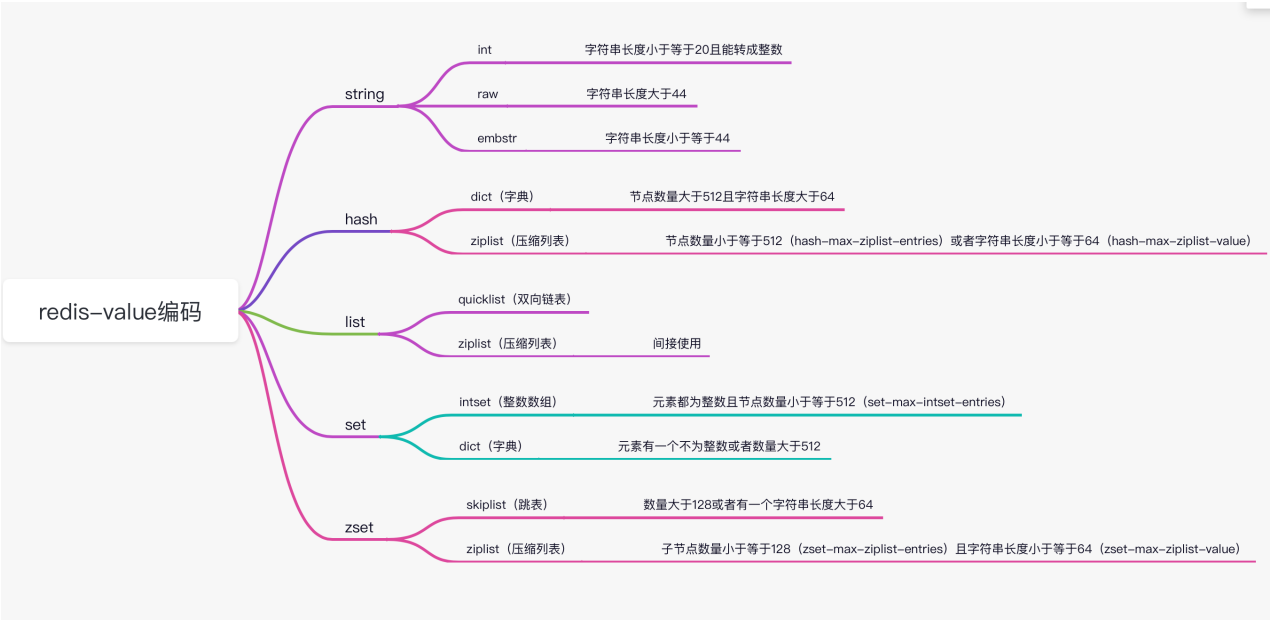




redis数据存储



redis数据存储规则



运行时间和存储效率进行平衡

string结构以及应用

结构以及细节


```
相当于操作：unordered_map<string, string>
unordered_map<key, value>
```

主要命令以及应用

```
SET key val              // 设置 key 的 value 值
GET key                  // 获取 key 的 value

INCR key                 // 执行原子加一的操作
INCRBY key increment     // 执行原子加一个整数的操作

DECR key                 // 执行原子减一的操作
DECRBY key decrement     // 执行原子减一个整数的操作

SETNX key value          // 如果key不存在，这种情况下等同SET命令。 当key存在时，什么也不做
DEL key                  // 删除 key val 键值对
```

- 设置单值

```
set hello world
```

- 存储对象

```
set role:10001 '{["name"]:"mark",["sex"]:"male",["age"]:30}'
这里面的字段不会改变的时候
```

- 累加器

```
// 统计阅读数 公众号
incr reads
```

- 分布式锁

```
// 获取锁
setnx lock 客户唯一标识
操作临界资源
// 释放锁
del lock
```

list结构以及应用

结构以及细节

quicklist 当中的节点 存储的就是 压缩列表

```
相当于操作 unordered_map<string, list<string>>

unordered_map<key, list<value>>
```

主要命令以及应用

```
LPUSH key value [value ...]     // 从队列的左侧入队一个或多个元素
LPOP key                        // 从队列的左侧弹出一个元素

RPUSH key value [value ...]     // 从队列的右侧入队一个或多个元素
RPOP key                        // 从队列的右侧弹出一个元素

LRANGE key start end            // 返回从队列的 start 和 end 之间的元素  0, 1
2
LREM key count value            // 从存于 key 的列表里移除前 count 次出现的值为
value 的元素

BRPOP key timeout               // 它是 RPOP 的阻塞版本，因为这个命令会在给定
list无法弹出任何元素的时候阻塞连接
```

- 栈（先进后出 FILO）

```
LPUSH + LPOP
或者
RPUSH + RPOP
```

- 队列（先进先出 FIFO）

```
LPUSH + RPOP
或者
RPUSH + LPOP
```

- 阻塞队列（blocking queue）

```
LPUSH + BRPOP
或者
RPUSH + BLPOP
阻塞? 阻塞在当前连接上  命令执行上
```

- 朋友圈消息推送


```
消息存储为 json 格式
将用户id作为 key
添加一条朋友圈信息:
    lpush role:10001 '{["name"]:"零声学院【Mark老师】", ["text"]:"祝大家新年快乐!", ["picture"]:["url://image-20210203172741434.jpg", "url://image-20210203172741435.jpg"], timestamp = 123123123}'
刷新出前10条朋友圈:
    lrange role:10001 0 9
```

hash结构以及应用

结构以及细节


```
相当于操作 unordered_map<string, unordered_map<string, string>>

unordered_map<key, unordered_map<field, value>>
```

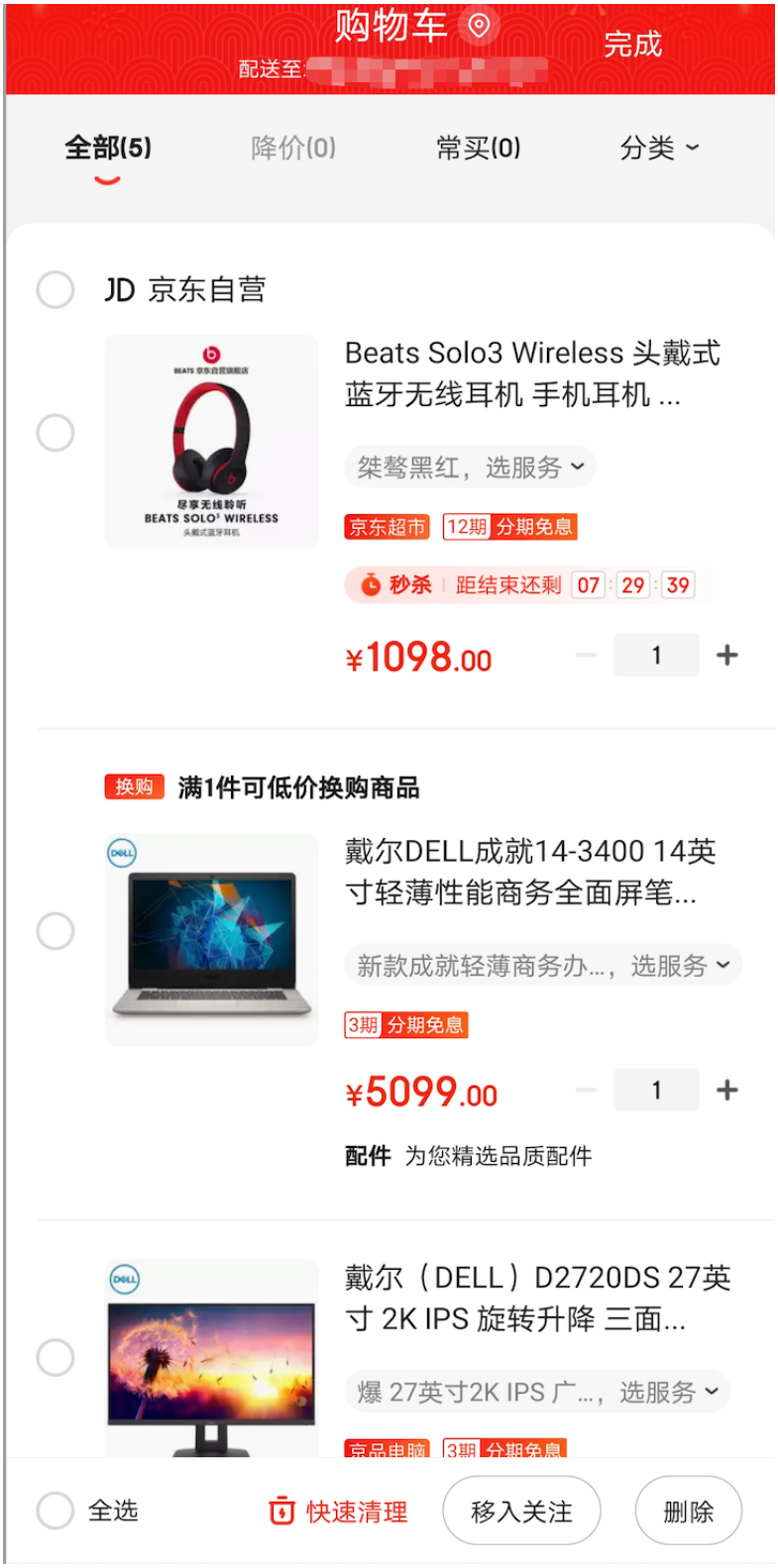
主要命令以及应用

```
HGET key field              // 获取 key 对应 hash 中的 field 对应的值
HSET key field value        // 设置 key 对应 hash 中的 field 对应的值

HMSET key field1 value1 field2 value2 ... fieldn valuen // 设置多个hash键值对
HMGET key field1 field2 ... fieldn

HINCRBY key field increment // 给 key 对应 hash 中的 field 对应的值加一个整数值
HLEN key                    // 获取 key 对应的 hash 有多少个键值对
HDEL key field              // 删除 key 对应的 hash 的键值对，该键为field
```

- 存储对象

```
hmset role:10001 name mark age 30 sex male

与 string 比较
set role:10001 '{["name"]:"mark",["sex"]:"male",["age"]:30}'

假设现在修改 mark的年龄为31岁

hash:
    hset hash:10001 age 31
string:
    get role:10001
    将得到的字符串调用json解密，取出字段，修改 age 值
    然后调用json加密
    set role:10001 '{["name"]:"mark",["sex"]:"male",["age"]:31}'
```

- 购物车

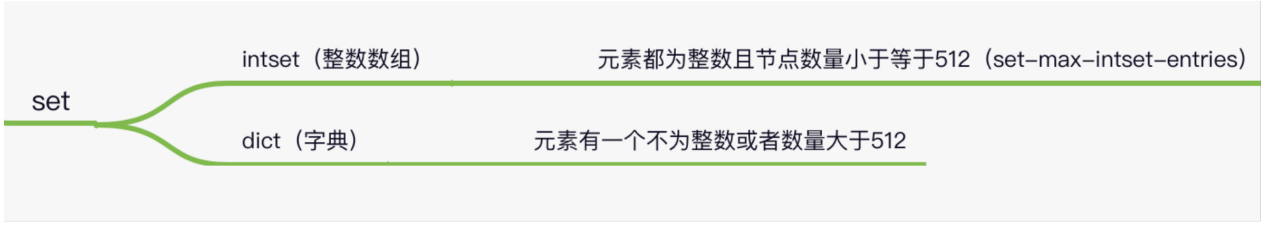



```
将用户id作为 key
商品id作为 field
商品数量作为 value

添加商品:
    hset MyCart:10001 40001 1
    lpush MyItem:10001 40001
增加数量:
    hincrby MyCart:10001 40001 1
```

```
    hincrby MyCart:10001 40001 -1  // 减少数量1
显示所有物品数量:
    hlen MyCart:10001
删除商品:
    hdel MyCart:10001 40001
    lrem MyItem:10001 1 40001
获取所有物品:
    lrange MyItem:10001
    40001 40002 40003
    hget MyCart:10001 40001
    hget MyCart:10001 40002
    hget MyCart:10001 40003

显示顺序 是 添加顺序?
思考:这里显示的按照我们添加顺序进行排列的,那么我们应该怎么做?
redis 经常是用过结构一起使用     不需要知道添加的时间    只需要知道添加先后顺序

```

set结构以及应用

结构以及细节




```
相当于操作 unordered_map<string, unordered_set<string>>

unordered_map<key, unordered_set<member>>
```

主要命令以及应用

```
SADD key member [member ...]    // 添加一个或多个指定的member元素到集合的 key中
SCARD key                       // 计算集合元素个数

SMEMBERS key                    // SMEMBERS key
SISMEMBER key member            // 返回成员 member 是否是存储的集合 key的成员

SRANDMEMBER key [count]         // 随机返回key集合中的一个或者多个元素，不删除这些元素
SPOP key [count]                // 从存储在key的集合中移除并返回一个或多个随机元素

SDIFF key [key ...]             // 返回一个集合与给定集合的差集的元素
SINTER key [key ...]            // 返回指定所有的集合的成员的交集
SUNION key [key ...]            // 返回给定的多个集合的并集中的所有成员
```

- 抽奖

```
添加抽奖用户
    sadd Award:1 10001 10002 10003 10004 10005 10006
    sadd Award:1 10009
查看所有抽奖用户
    smembers Award:1
抽取多名幸运用户
    srandmember Award:1 10
如果抽取一等奖1名，二等奖2名，三等奖3名，该如何操作?

```

- 共同关注

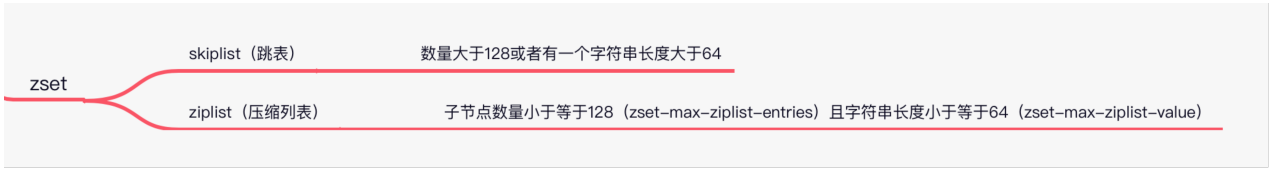
```
sadd follow:A mark king darren mole vico
sadd follow:C mark king darren
sinter follow:A follow:C
```

- 可能认识的人

```
sadd follow:A mark king darren mole vico
sadd follow:C mark king darren
C可能认识的人:
    sdiff follow:A follow:C
```

zset结构以及应用

结构以及细节




```
相当于操作 unordered_map<string, skiplist<string, string>>

unordered_map<key, skiplist<member, score>>
```

主要命令以及应用

```
ZADD key [NX|XX] [CH] [INCR] score member [score member ...] //添加到键为key有序集合（sorted set）里面
ZREM key member [member ...] // 从键为key有序集合中删除 member 的键值对
ZSCORE key member // 返回有序集key中，成员member的score值
ZINCRBY key increment member // 为有序集key的成员member的score值加上增量increment
ZCARD key // 返回key的有序集元素个数
ZRANK key member // 返回有序集key中成员member的排名
ZRANGE key start stop [WITHSCORES] // 返回存储在有序集合key中的指定范围的元素
ZREVRANGE key start stop [WITHSCORES] // 返回有序集key中，指定区间内的成员(逆序)
```

- 百度热榜


```

10001 新闻为 '31省区市新增确诊25例 本土15例'
```

```
10002 新闻为 '多地调整返乡政策'
点击新闻:
    zincrby hot:20210203 1 10001
    zincrby hot:20210203 1 10002
    zincrby hot:20210203 1 10003
    zincrby hot:20210203 1 10004
    zincrby hot:20210203 1 10005
    zincrby hot:20210203 1 10006
    zincrby hot:20210203 1 10007
    zincrby hot:20210203 1 10008
    zincrby hot:20210203 1 10009
    zincrby hot:20210203 1 10010
获取排行榜:
    zrevrange hot:20210203 0 9 withscores
```