链滴

redis 数据结构

作者：AshShawn

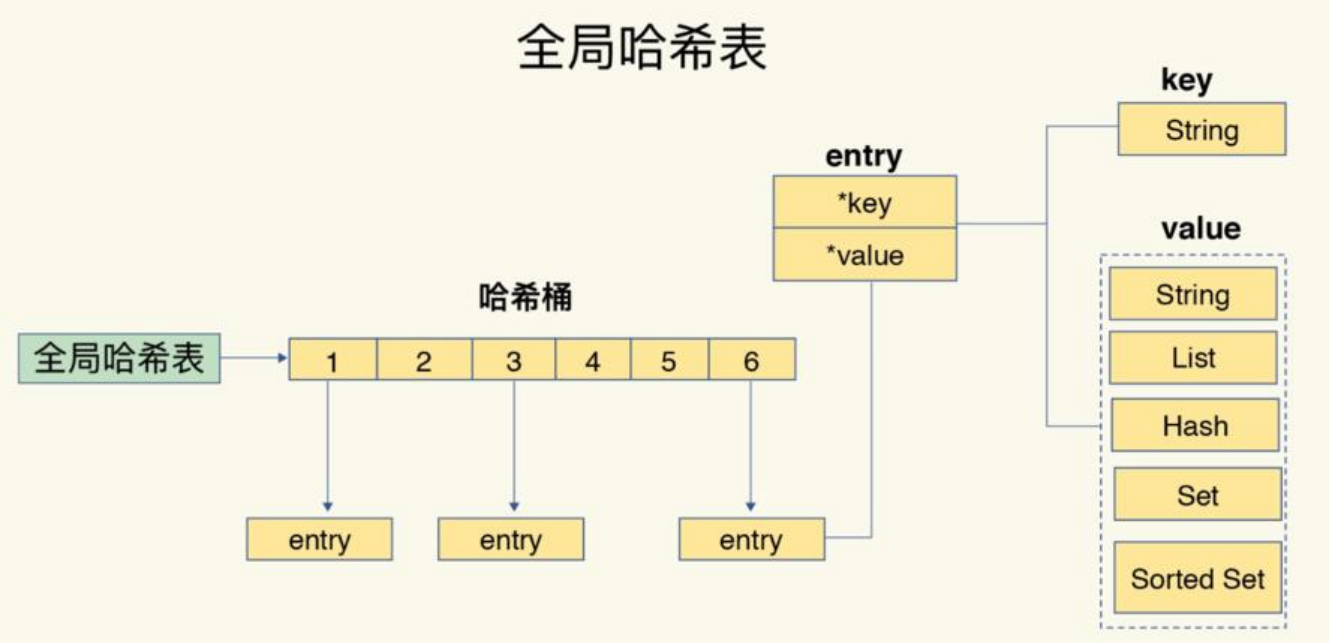
原文链接：https://ld246.com/article/1650381951675

来源网站：链滴

1. 全局hash表

字典

```
typedef struct dict {
    dictType *type; // 类型特定函数
    void *privdata; // 私有数据
    dictht ht[2]; // 哈希表
    // rehash索引
    long rehashidx; /* rehashing not in progress if rehashidx == -1 */
    unsigned long iterators; /* number of iterators currently running */
} dict;
```

1.1 hash冲突

开放寻址法: 当产生冲突时,向后寻址,直到找到第一个空位置

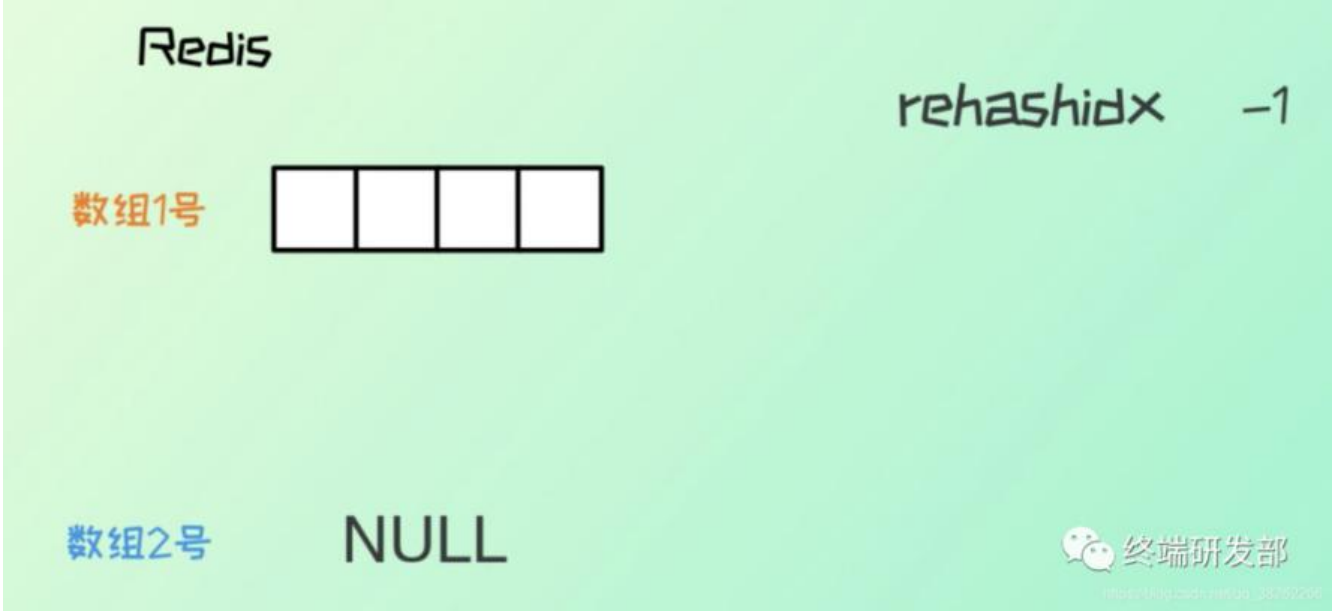
链表法(拉链法): 在桶中使用链表

1.2 rehash

当redis字典李数据越来越多的时候,就会发生扩容,即rehash

redis字典（hash表）底层有两个数组，还有一个rehashidx用来控制rehash

原文链接：redis 数据结构



渐进式rehash

渐进式rehash采用的是一种分而治之的方式，将rehash的操作分摊在每一个的访问中，避免集中式rehash而带来的庞大计算量。

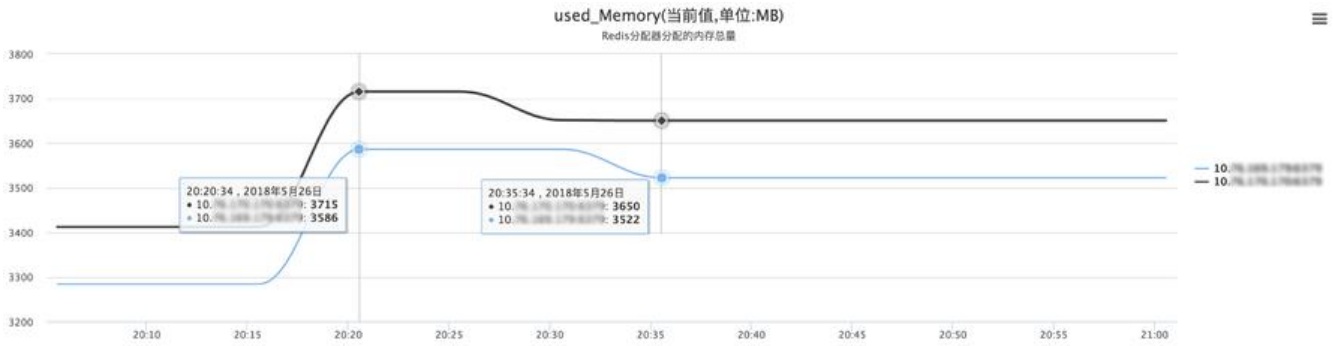
需要注意的是在渐进式rehash的过程，如果有增删改查操作时，如果 index 大于 rehashindex ，访问 ht[0] ，否则访问 ht[1]。

1. ht[1] 分配空间，让字典同时持有 ht[0] 和 ht[1] 两个哈希表

2. 将 rehashindex 的值设置为 0 ， 表示rehash工作正式开始

3. 在rehash期间，每次对字典执行增删改查操作是，程序除了执行指定的操作以外，还会顺带将 ht[0] 哈希表在 rehashindex 索引上的所有键值对rehash到 ht[1] ， 当rehash工作完成以后， rehashindex 的值 +1

4. 随着字典操作的不断执行，最终会在某一时间段上 ht[0] 的所有键值对都会被rehash到 ht[1] ， 这将 rehashindex 的值设置为 -1 ， 表示rehash操作结束



rehash会造成内存占用突升,当内存不足时会出发大量key被淘汰,可能影响业务,解决方案

1. 修改源码,内存不足不去rehash

2. 运维规避,监控内存

2.SDS

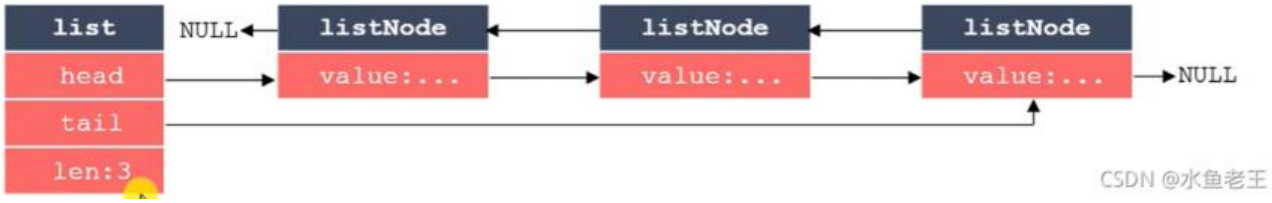
2.1 结构

```
struct sdshdr{
    int len;//字符串长度
    int free;//未使用字节数
    char buf[];//字节数组
}
```

2.2 为什么使用sds而不是普通c字符串

1. 自带length, 常数复杂度
2. 杜绝缓冲区溢出,每次修改sds时都会校验空间是否足够,如果不够,会扩容
3. 减少修改字符串时带来的内存重分配次数
    1. 空间预分配
    2. 惰性空间释放
4. 二进制安全,c字符串无法使用\0,sds通过len和free配合不需要使用\0为结束标志
5. 部分兼容string.h的函数

3.链表




```
//链表
typedef struct list {
    listNode *head;   //头结点
    listNode *tail;   //尾结点
    void *(*dup)(void *ptr);  //节点复制函数
    void (*free)(void *ptr);  //节点释放函数
    int (*match)(void *ptr, void *key); //节点对比函数
    unsigned long len;  //链表所包含的节点数量
} list;

//节点
typedef struct listNode {
    struct listNode *prev;
    struct listNode *next;
    void *value;
} listNode;
```

特点:

1. 双向,每个节点带prev和next指针
2. 无环,两头都指向null,链表访问到null即为终点

3. 双指针,头指针和尾指针
4. 带长度计数器,即自带length
5. 多态

4. 字典(map)

Redis（一）字典 - Jomini - 博客园 (cnblogs.com)

5. 跳表(skiplist)

(40条消息) redis 跳表_春天的早晨的博客-CSDN博客_redis跳表

6.整数集合(intset)

(40条消息) redis之intset_happytree001的博客-CSDN博客_redis的intset

7. 压缩链表(ziplist)

Redis源码笔记：压缩链表 - NotFoundException - 博客园 (cnblogs.com)

8.redisObject

Redis 学习笔记（篇五）：对象（RedisObject） - 风中抚雪 - 博客园 (cnblogs.com)