链滴

01-Redis 核心数据结构与高性能原理

作者：hu244785010

原文链接：https://ld246.com/article/1682607473949

来源网站：链滴

Redis安装

下载地址：http://redis.io/download
安装步骤：

```
# 安装gcc
yum install gcc

# 把下载好的redis-5.0.3.tar.gz放在/usr/local文件夹下，并解压
wget http://download.redis.io/releases/redis-5.0.3.tar.gz
tar -zxvf redis-5.0.3.tar.gz
cd redis-5.0.3

# 进入到解压好的redis-5.0.3目录下，进行编译与安装
make
#
 修改配置
daemonize yes  #后台启动
protected-mode no  #关闭保护模式，开启的话，只有本机才可以访问redis
# 需要注释掉bind
#bind 127.0.0.1（bind绑定的是自己机器网卡的ip，如果有多块网卡可以配多个ip，代表允许客户端过机器的哪些网卡ip去访问，内网一般可以不配置bind，注释掉即可）

# 启动服务
src/redis-server redis.conf

# 验证启动是否成功
ps -ef | grep redis

# 进入redis客户端
src/redis-cli
```

```
# 退出客户端
quit

# 退出redis服务:
  (1) pkill redis-server
  (2) kill 进程号
  (3) src/redis-cli shutdown
```

Redis基本类型

string

Redis 字符串数据类型的相关命令用于管理 redis 字符串值

hash

Redis hash 是一个 string 类型的 field（字段） 和 value（值） 的映射表，hash 特别适合用于存储象。 Redis 中每个 hash 可以存储 232 - 1 键值对（40多亿)

list

Redis列表是简单的字符串列表，按照插入顺序排序。你可以添加一个元素到列表的头部（左边）或尾部（右边）一个列表最多可以包含 232 - 1 个元素 (4294967295, 每个列表超过40亿个元素)。

set

Redis 的 Set 是 String 类型的无序集合。集合成员是唯一的，这就意味着集合中不能出现重复的数。集合对象的编码可以是 intset 或者 hashtable。Redis 中集合是通过哈希表实现的，所以添加，删，查找的复杂度都是 O(1)。集合中最大的成员数为 232 - 1 (4294967295, 每个集合可存储40多亿个员)

zset

Redis 有序集合和集合一样也是 string 类型元素的集合,且不允许重复的成员。不同的是每个元素都关联一个 double 类型的分数。redis 正是通过分数来为集合中的成员进行从小到大的排序。有序集的成员是唯一的,但分数(score)却可以重复。集合是通过哈希表实现的，所以添加，删除，查找的复度都是 O(1)。 集合中最大的成员数为 232 - 1 (4294967295, 每个集合可存储40多亿个成员)。

geo

Redis GEO 主要用于存储地理位置信息，并对存储的信息进行操作

Redis GEO 操作方法有：(经度，维度)

* geoadd：添加地理位置的坐标。
* geopos：获取地理位置的坐标。
* geodist：计算两个位置之间的距离。
* georadius：根据用户给定的经纬度坐标来获取指定范围内的地理位置集合。

* georadiusbymember：根据储存在位置集合里面的某个地点获取指定范围内的地理位置集合。

* geohash：返回一个或多个位置对象的 geohash 值。

hyperloglog计数

1. HyperLogLog是一种算法，并非redis独有。

2. 目的是做基数统计，故不是集合，不会保存元数据，只记录数量而不是数值。

3. 耗空间极小，支持输入非常体积的数据量。

4. 核心是基数估算算法，主要表现为计算时内存的使用和数据合并的处理。最终数值存在一定误差。

5. redis中每个hyperloglog key占用了12K

6. 的内存用于标记基数（官方文档）。

7. pfadd命令并不会一次性分配12k内存，而是随着基数的增加而逐渐增加内存分配；而pfmerge操则会将sourcekey合并后存储在12k大小的key中，这由hyperloglog合并操作的原理（两个hyperlogl g合并时需要单独比较每个桶的值）可以很容易理解。

8. 误差说明：基数估计的结果是一个带有

9. 0.81% 标准错误（standard error）的近似值。是可接受的范围。

10. Redis 对 HyperLogLog 的存储进行了优化，在计数比较小时，它的存储空间采用稀疏矩阵存储空间占用很小，仅仅在计数慢慢变大，稀疏矩阵占用空间渐渐超过了阈值时才会一次性转变成稠密矩，才会占用 12k 的空间。

bitSet

Redis的单线程和高性能

Redis是单线程吗?

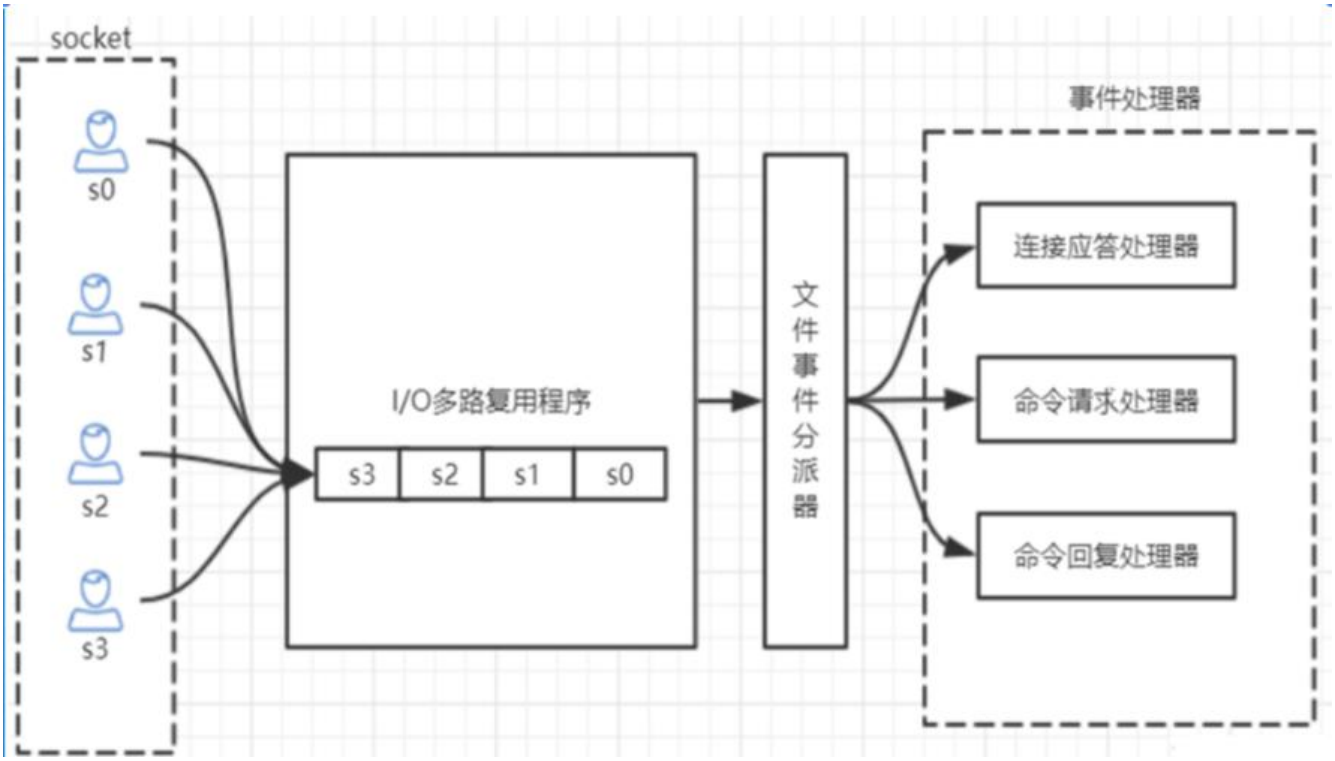
Redis 的单线程主要是指 Redis 的网络 IO 和键值对读写是由一个线程来完成的，**这也是 Redis 对提供键值存储服务的主要流程。但 Redis 的其他功能，比如持久化、异步删除、集群数据同步等，其是由额外的线程执行的。

Redis 单线程为什么还能这么快?

因为它所有的数据都在内存中，所有的运算都是内存级别的运算，而且单线程避免了多线程的切换性损耗问题。正因为 Redis 是单线程，所以要小心使用 Redis 指令，对于那些耗时的指令(比如keys)，定要谨慎使用，一不小心就可能会导致 Redis 卡顿。

Redis 单线程如何处理那么多的并发客户端连接?

Redis的IO多路复用：redis利用epoll来实现IO多路复用，将连接信息和事件放到队列中，依次放到件事件分派器，事件分派器将事件分发给事件处理器。




```
# 查看redis支持的最大连接数，在redis.conf文件中可修改，# maxclients 10000
127.0.0.1:6379> CONFIG GET maxclients
   ##1) "maxclients"
   ##2) "10000"
```

其他高级命令

keys： 全量遍历键

用来列出所有满足特定正则字符串规则的key，当redis数据量比较大时，性能比较差，要避免使用

scan：渐进式遍历键

scan 参数提供了三个参数，第一个是 cursor 整数值(hash桶的索引值)，第二个是 key 的正则模式，三个是一次遍历的key的数量(参考值，底层遍历的数量不一定)，并不是符合条件的结果数量。第一次历时，cursor 值为 0，然后将返回结果中第一个整数值作为下一次遍历的 cursor。一直遍历到返回的ursor 值为 0 时结束。

注意：但是scan并非完美无瑕， 如果在scan的过程中如果有键的变化（增加、 删除、 修改） ，那遍历效果可能会碰到如下问题： 新增的键可能没有遍历到， 遍历出了重复的键等情况， 也就是说sca并不能保证完整的遍历出来所有的键， 这些是我们在开发时需要考虑的。

Info：查看redis服务运行信息

Server 服务器运行的环境参数

Clients 客户端相关信息

Memory 服务器运行内存统计数据

Persistence 持久化信息

Stats 通用统计数据

Replication 主从复制相关信息

CPU CPU 使用情况

Cluster 集群信息

KeySpace 键值对统计数量信息

```
connected_clients:2                 # 正在连接的客户端数量

instantaneous_ops_per_sec:789       # 每秒执行多少次指令

used_memory:929864                  # Redis分配的内存总量(byte)，包含redis进程内部的开销和数
占用的内存
used_memory_human:908.07K           # Redis分配的内存总量(Kb，human会展示出单位)
used_memory_rss_human:2.28M         # 向操作系统申请的内存大小(Mb)（这个值一般是大于use
_memory的，因为Redis的内存分配策略会产生内存碎片）
used_memory_peak:929864             # redis的内存消耗峰值(byte)
used_memory_peak_human:908.07K      # redis的内存消耗峰值(KB)

maxmemory:0                         # 配置中设置的最大可使用内存值(byte),默认0,不限制，一般配置为
器物理内存的百分之七八十，需要留一部分给操作系统
maxmemory_human:0B                  # 配置中设置的最大可使用内存值
maxmemory_policy:noeviction         # 当达到maxmemory时的淘汰策略
```

中文官网:http://www.redis.cn/