Redis是一个开源的使用ANSI C语言编写、支持网络、可基于内存亦可持久化的日志型、Key-Value数据库，并提供多种语言的API。数据结构是存储的基本单位，底层使用很多数据结构进行存储，同时所有的操作都保证了原子性，比如压缩链表对应C++ Vector，双向链表，HashTable，Set等等。将其数据库大部分保存在内存中（很久没用的数据会被交换到硬盘）（Memcached倒是完全基于内存），是一种内存非关系数据库，仅使用磁盘进行持久化。所以有极高的读写性能。

作为一个单线程运行的数据库，所有操作保持了原子性。之所以使用单线程模式估计是因为不想承受锁的代价，以及方便代码开发，同时也就尽量没有上下文切换的代价。虽然是单线程运行，但是如果两个客户端传送过去的命令串执行的时候（保证了单个命令执行的原子性），可能会混乱，所以还是要使用锁。Redis采用的是基于内存的采用的是单进程单线程模型的 KV 数据库，由C语言编写，官方提供的数据是可以达到100000+的QPS（每秒内查询次数）。这个数据不比采用单进程多线程的同样基于内存的 KV 数据库 Memcached 差！服务器需要不断监听文件事件的套接字才能得到待处理的文件事件，但是不能一直监听，否则时间事件无法在规定的时间内执行，因此监听时间应该根据距离现在最近的时间事件来决定。

除了作为缓存服务器，还支持消息队列以及消息的订阅与发布（不支持消息分组），多用于实时性较高的消息推送，并不保证可靠（断电就会清空）。使用redis作为消息队列，估计是刚好使用redis作为缓存，然后消息队列的场景比较简单，就不需要部署其他软件。

Redis可以将数据复制到任意数量的从机中。可以从mysql、Oracle等数据库同步。

**数据库的速度关键还是在IO方面，基于内存的时候减少了硬盘IO。**

**通常数据库使用nosql作为cache，后端使用关系数据库来存储数据。**

**优点：**

支持通过数据分片，将数据存储在多个实例中，使用多个实例还可以避免单击内存数量的限制

作为一个缓存数据库（不适合做持久化数据库），支持设置过期时间，一旦过期，会自动删除对象。

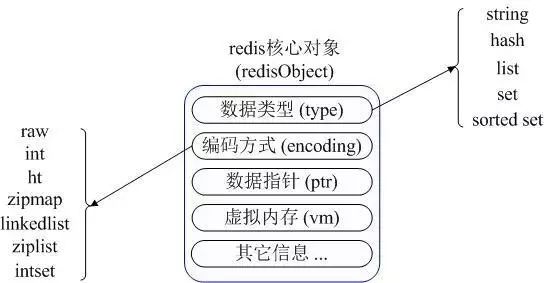
数据结构经过优化，保证了速度。相比较而言，Memcached（虽然作为一个单进程多线程nosql）在数据量增长的时候出现了低命中率的问题，导致数据访问经常穿透到后面的关系型数据库。

<https://blog.csdn.net/u011871037/article/details/51859065> Memcached 、redis 区别

Memcached 不支持分布式，只能通过在客户端使用一致性哈希来实现分布式存储，这种方式在存储和查询时都需要先在客户端计算一次数据所在的节点。

Redis Cluster 实现了分布式的支持。

首先Redis内部使用一个redisObject对象来表示所有的key和value，如下图所示：



虚拟内存的使用不一定是操作系统内存不够使用，真正是因为程序使用的内存数量不够。这一个现象在redis中常见。

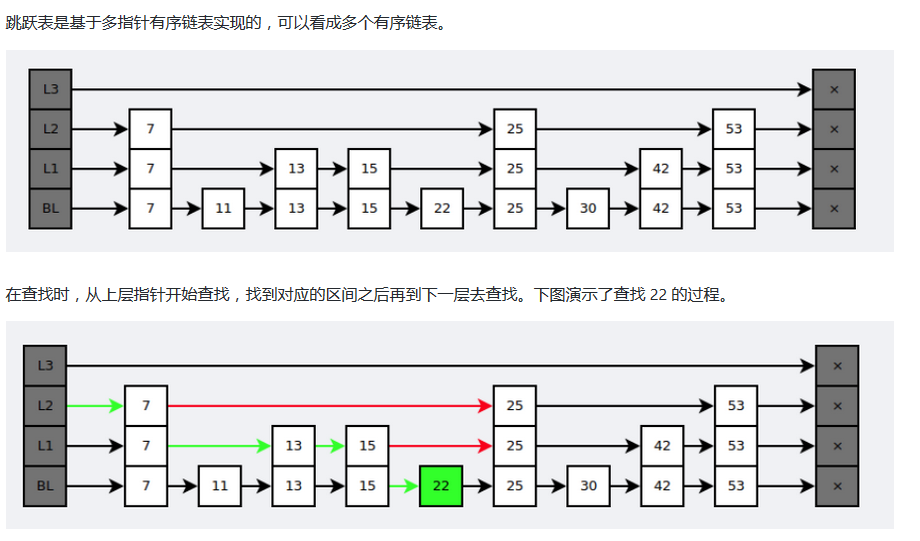
内存淘汰策略是指当内存数据集上升到一定高度之后，执行。比如挑选过期数据删除，向进程虚拟内存中写入不常用数据，也可能是随意选择几个数据淘汰，看具体的淘汰策略。

| **数据类型** | **可以存储的值** | **操作** |
| --- | --- | --- |
| STRING | 字符串、整数或者浮点数 | 对整个字符串或者字符串的其中一部分执行操作 对整数和浮点数执行自增或者自减操作 |
| LIST | 列表 | 从两端压入或者弹出元素 对单个或者多个元素 进行修剪，只保留一个范围内的元素 |
| SET | 无序集合 | 添加、获取、移除单个元素 检查一个元素是否存在于集合中 计算交集、并集、差集 从集合里面随机获取元素 |
| HASH | 包含键值对的无序散列表 | 添加、获取、移除单个键值对 获取所有键值对 检查某个键是否存在 |
| ZSET | 有序集合 | 添加、获取、删除元素 根据分值范围或者成员来获取元素 计算一个键的排名 |

集合、字典是通过哈希表实现的，所以添加，删除，查找的复杂度都是O(1)。存在rehash机制，保证希望哈希表的负载因子（load factor），维持在一个合理的范围之内，一旦出现过大的情况，需要创建一个更大的hash表，同时进行数据的转移操作（rehash），但是因为存储的数据量过大，一旦集中操作意味着很大的计算量（hash函数）可能会导致服务器因此死机。，所以只能是分次与渐进的 。

对于hash碰撞的处理，通过桶链接法。

有序集合只是内部有序。底层使用跳跃表实现：



与红黑树等平衡树相比，跳跃表具有以下优点：

* 插入速度非常快速，因为不需要进行旋转等操作来维护平衡性；
* 更容易实现；
* 支持无锁操作。

redis三种删除策略：

定时删除、惰性删除、定期删除

惰性删除是指，在使用这个值的时候，才进行过期时间验证，如果过期就会删除。

## 时间事件

服务器有一些操作需要在给定的时间点执行，时间事件是对这类定时操作的抽象。

时间事件又分为：

* 定时事件：是让一段程序在指定的时间之内执行一次；
* 周期性事件：是让一段程序每隔指定时间就执行一次。

Redis 将所有时间事件都放在一个无序链表中，通过遍历整个链表查找出已到达的时间事件，并调用相应的事件处理器。我觉得存放在按照时间顺序的有序链表中更好。

redis分区:

* 最简单的分区方式为**范围分区**，就是映射一定范围的对象到特定的Redis实例。比如，ID从0到10000的用户会保存到实例R0，ID从10001到 20000的用户会保存到R1，以此类推。
* 另外一种方式是hash一致算法实现分区，对key值进行hash一致性计算后得到结果，最终将数据保存到某一台redis实例中，具体的hash一致性算法可以自行百度一下。