# Redis

## 为什么 Redis 单线程却能支撑高并发？

redis中使用I/O多路复用技术来防止I/O阻塞

在 I/O 多路复用模型中，最重要的函数调用就是 select，该方法的能够同时监控多个文件描述符的可读可写情况，当其中的某些文件描述符可读或者可写时， select方法就会返回可读以及可写的文件描述符个数。与此同时也有其它的 I/O 多路复用函数 epoll/kqueue/evport

## Redis为什么设计成单线程？

执行两个任务，在不涉及I/O操作的时候，单线程依次执行两个任务比双线程分别执行两个任务要快。Redis不涉及I/O操作，因此设计为单线程是效率最高的！

## Redis的性能瓶颈在哪？

一般在两个地方：

其一是机器内存大小，内存大小关系到Redis存储的数据量

其二是网络带宽，Redis客户端执行一条命令分为四个过程：发送命令、命令排队、命令执行、返回结果。而其中发送命令+返回结果这一过程被称为Round Trip Time（RTT，往返时间）

# 缓存问题

## 缓存穿透

正常情况下，我们去查询数据都是存在。那么请求去查询一条压根数据库中根本就不存在的数据，也就是缓存和数据库都查询不到这条数据，但是请求每次都会打到数据库上面去。这种查询不存在数据的现象我们称为缓存穿透。

解决办法：1.缓存空值；2.BloomFilter（BloomFilter 类似于一个hase set 用来判断某个元素（key）是否存在于某个集合中）

如何选择：针对这种key异常多，请求重复率比较低的数据，我们就没有必要进行缓存，使用第二种方案直接过滤掉。对于空数据的key有限的，重复率比较高的，我们则可以采用第一种方式进行缓存。

## 缓存击穿

在平常高并发的系统中，大量的请求同时查询一个 key 时，此时 这个key 正好失效了，就会导致大量的请求都打到数据库上面去。这种现象我们成为击穿。

如何解决：上面的现象是多个线程同时去查询数据库的这条数据，那么我们可以在第一个查询数据的请求上使用一个互斥锁来锁住它。其他的线程走到这一步拿不到锁就等着，等第一个线程查询到了数据，然后做缓存。后面的线程进来发现已经有缓存了，就直接走缓存。

## 缓存雪崩

缓存的情况是说，当某一时刻发生大规模的缓存失效的情况。比如你的缓存服务宕机了，会有大量的请求进来直接打到DB上面。结果就是DB 称不住，挂掉。

解决办法：事前：使用集群缓存，保证缓存服务的高可用（如果是使用 Redis，可以使用 主从+哨兵 ，Redis Cluster 来避免 Redis 全盘崩溃的情况）事中：使用 ehcache 本地缓存 + Hystrix 限流&降级 ,避免 MySQL 被打死的情况发生。事后：开启 Redis 持久化机制，尽快恢复缓存集群

## 热点数据集中失效问题怎么解决

我们在设置缓存的时候，一般会给缓存设置一个失效时间，过了这个时间，缓存就失效了。对于一些热点的数据来说，当缓存失效以后会存在大量的请求过来，然后打到数据库去，从而可能导致数据库崩溃的情况。

解决办法：设置不同的失效时间；互斥锁（在第一个请求去查询数据库的时候对他加一个互斥锁，其余的查询请求都会被阻塞住，直到锁被释放，从而保护数据库。）