# 缓存

## Redis

### 基本性质

单线程，内存数据库

### 基本数据类型

String hash set zset list

### 如何保证缓存一致性

延迟双删：修改操作时，先删除缓存，再修改数据库，修改完成后等1s再删除缓存。第二次删除目的是删掉第一次删除和修改数据库之间的写入缓存的脏数据。

### 缓存击穿

热点数据突然失效导致大量请求打在数据库。解决方法：设置热点数据永不过期；缓存失效时查数据库加互斥锁，暂时降低性能但是可以防止数据库击穿。

### 缓存雪崩

热点时间放入缓存的大量key在同一时间过期，导致后续热点请求打在数据库上宕机。解决方法：设置失效时间时加上一个随机时间，不让热点key在同一时间过期。使用流量限制手段，对超出限制的流量进行熔断，可选流量控制算法或者sentinel流控。

### 缓存穿透

大量请求查不存在的key，全部打在数据库上导致宕机。解决方法：布隆过滤器，把缓存中所有数据加入布隆过滤器，所有的请求都经过布隆过滤器。如果布隆过滤器验证存在，则缓存中有可能存在，如果布隆过滤器验证不存在，则缓存中一定不存在。缓存中增加元素时需要同时加入布隆过滤器，删除缓存时不用动布隆过滤器。

### Redis比memcached的优势

Redis支持多种数据结构，更灵活。Memcached只能存键值对，值只能时字符串，且memcached没有持久化手段，只能在内存。

### Redis持久化机制

Rdb和aof两种方式。rdb是一个单独的dump.rdb文件，后台持久化，性能强。Aof是append only file，会记录每一条redis命令到aof文件中，恢复的时候就是把每条命令再执行一遍。Aof恢复速度慢，文件占用大，但是更能有效恢复文件。

### Redis事务中pipeline和multi的区别

Pipeline是连续执行一系列的命令，并不是事务。一般使用会在pipeline中再开启multi事务。Pipeline非原子性，但是由于redis是单线程，看上去比较像原子性。

### Redis事务几个命令multi、exec、watch、discard作用

Multi：开启一个事务

Exec：提交一个事务

Watch：监视一个key，如果在下一个事务提交之前（exec）watch的key有变化，则事务会回滚。

Discard：取消一个事务，和回滚一个意思。因为redis是单线程的，所以在提交之前不会有别的线程来更改。

### Redis过期键删除策略

Redis同时存在两个过期策略，惰性删除、定期删除。

惰性删除：过期的键不会直接删除，要下一次访问到该key，调用函数判断该key是否过期，过期了才会删除。

定期删除：redis.conf中设置了hz 10参数，意思是每秒进行的过期扫描次数，默认10。扫描是随机在redis服务器的过期字典中抽取20个键，判断是否过期，过期则删除。如果抽取过期的比例超过25%会追加一次扫描。

### Redis分布式锁

应用使用setnx命令获取锁，再用expire设置过期时间释放锁，使用完了再缓存中删除锁。如果遇到没来得及expire，redis服务器宕机等情况，则可以使用setnx的过期参数。

### Redis乐观锁、悲观锁

乐观锁基于watch命令实现，先watch一个key，然后执行事务，如果事务提交前有更改，则回滚。主要用于抢票应用，watch库存，如果提交订单时库存没变，则提交成功，如果变了则回滚。乐观锁不影响读库存。

悲观锁基于setnx命令实现，有人获取了锁其他人则无法做任何操作。

# 消息队列

## Rabbitmq

## Kafka

# 数据库

## Mysql

### 索引的数据结构

B+tree，阶数很高