Redis:为什么使用redis而不是map做缓存。

Redis 可以用几十G内存来做缓存，Map 不行，一般 JVM 也就分几个 G 数据就够大了

Redis 的缓存可以持久化，Map 是内存对象，程序一重启数据就没了

Redis 可以实现分布式的缓存，Map 只能存在创建它的程序里

Redis 可以处理每秒百万级的并发，是专业的缓存服务，Map 只是一个普通的对象

Redis 缓存有过期机制，Map 本身无此功能

Redis 有丰富的 API，Map 就简单太多了

我的理解是从数据结构和方法层面来理解，从数据结构来说，都是k-v键值对，但是redis根据各种需要加了各种的功能。本质上来说是实现了各种api的一个模块，只是其是用c写的而已。比如其持久化aof,rdb,并发度高，过期策略，丰富的api。

Redis为什么那么快？

1. 基于内存2.HashMap结构查找快而传统的关系型数据库需要索引，比如b+树，其比hash查找要慢3.采用单线程4.使用了多路复用的io机制5.底层模型不用，通信的应用协议不同

哨兵模式：哨兵的作用：1.监控redis的节点是否正常运行，2如果非正常运行将会通知管理员同时3进行故障转移，选举出新的master.并且通知客户端新的master地址。

集群模式：

redis cluster(采用分槽的方法，每个redis占用一定的分片，请求会发送到任意redis服务上再计算对应的key在哪个redis上) 运维比较复杂，数据迁移时需要人工干预。路由分区

redis sharding(基于客户端做的路由，通过hash计算当前key存放在哪个redis上) 不支持动态增删节点，客户端需要进行相应的调整。客户端分区

中间代理的方式：请求发送到代理服务器，由代理进行请求的转发。 Proxy的存储和逻辑是分开的，但是多了一层proxy的性能消耗。代理分区

事务：隔离性，非原子性

过期策略：定时过期，惰性过期，定时过期

内存淘汰策略：随机淘汰，最近最少使用，写入报错。

线程模型：多路复用。

最大节点数：16384.

缓存雪崩：缓存同时间大面积失效。后面的请求会大量落在数据库中。1.缓存过期时间设置随机；2.访问量小的情况下加锁访问；

缓存击穿：缓存中没有的数据库中有的数据，大量服务器同时读取缓存但是读不到又同时到数据库中读取造成服务器压力过大。1.设置热点数据永不过期2.加锁

缓存穿透：缓存和数据库中都没有数据，也是导致请求都落在数据库中。

1. 从缓存取不到的数据，在数据库中也没有取到，这时也可以将key-value对写为key-null
2. 采用布隆过滤器，将所有可能存在的数据哈希到一个足够大的 bitmap 中，一个一定不存在的数据会被这个 bitmap 拦截掉，从而避免了对底层存储系统的查询压力

消息中间件：解耦，异步，流量削峰

消息的顺序性：可以拆分成多个队列，使用一个队列对应一个消费者的模式。

消息的可靠性：采用确认机制，rabbit收到消息后

消息分发：多个订阅者循环分发。

消息路由：广播，路由键直接匹配和topic可以使用通用字符串匹配

消费者不被重复消费：每个消息带有一个唯一标识，在被消费一次后可以写入一个已消费日志中啦判断该消息是否已经被消费。

五种工作模式：发布订阅，路由，topic模式

Netty 的特点是什么？

高并发：Netty 是一款基于 NIO（Nonblocking IO，非阻塞IO）开发的网络通信框架，对比于 BIO（Blocking I/O，阻塞IO），他的并发性能得到了很大提高。

//传输快：Netty 的传输依赖于零拷贝特性，尽量减少不必要的内存拷贝，实现了更高效率的传输。

封装好：Netty 封装了 NIO 操作的很多细节，提供了易于使用调用接口。

3.Netty 的优势有哪些？

**使用简单**：封装了 NIO 的很多细节，使用更简单。

功能强大：预置了多种编解码功能，支持多种主流协议。

**定制能力强**：可以通过 ChannelHandler 对通信框架进行灵活地扩展。

**性能高**：通过与其他业界主流的 NIO 框架对比，Netty 的综合性能最优。

稳定：Netty 修复了已经发现的所有 NIO 的 bug，让开发人员可以专注于业务本身

阿里分布式服务框架 Dubbo

单线程模型：所有I/O操作都由一个线程完成，即多路复用、事件分发和处理都是在一个Reactor线程上完成的;

多线程模型: 有一个NIO 线程（Acceptor） 只负责监听服务端，接收客户端的TCP 连接请求；NIO 线程池负责网络IO 的操作

主从多线程模型:

接受和发送主要用了byteBuffer,directBuffer,消息在channel中移动进行处理，责任链模式。

序列化协议：xml,json等多种序列化协议

和tomcat很像，tomcat也是有一组线程监听端口，数据处理交给后续的service来处理，并且也是使用的责任链模式来进行数据的处理。但是其只支持http协议，是特有的web容器。