# =============数据库===============

## mycat原理

## 独占锁（排它锁：其他线程不能加锁，单表数据使用）与共享锁（其他线程可以加共享锁，主从表使用，锁主表数据）：结合MYSQL

## mysql实现死锁

## mysql悲观锁、乐观锁

悲观锁：行锁、表锁

乐观锁：mysql并没有实现乐观锁，可以通过版本号列，实现乐观锁

## binlog如何实现数据备份，相关

过程：

1、从数据库创建一个线程请求主数据库的binlog日志

2、主数据库接收到请求后，判断请求是否包含日志位置信息，如果有，从位置读取，如果没有，从第一个日志文件开始读取

3、从数据库接收到日志文件先保存在中继日志中，并记录位置

4、从数据库创建线程，从中继日志中读取

## 大表join小表优化，如何处理数据倾斜

1、MySQL的话，尽量加where提前缩小范围

## 聚簇索引和非聚簇索引，如何用可重复度的隔离级别避免幻读问题？

使用间隙锁

Innodb的主键索引为聚簇索引（索引指向数据行的索引时聚簇索引）

其他列的索引就是非聚簇索引，非聚簇索引叶子节点指向主键ID

间隙锁：解决幻读问题（可重复读会出现幻读，增加操作），间隙锁，锁定区间

# =============spring===============

## lockback原理，为什么不能打印控制台

## spring原理、异步事件

## spring事物传播特性，实现原理、异步事件

## spring事物传播机制

## 动态代理，区别

## spring 中的 bean 是线程安全的吗

## spring IOC 加载过程

## 为什么要使用spring？

1、解耦、方便开发

2、IOC、AOP

3、注解，方便实现

## spring循环引用问题，如何解决

1、构造函数引用循环无法解决，因为构造函数引用，得保证属性实例完全加载完毕才能引用

2、set注入，解决方案：三级缓存

## 多数据源如何实现

# ========分布式&微服务===========

## 分布式事务

## 分布式任务处理

## 啊阿萨斯大所大所大所大所大所大所多

## netty、IO

## HttpClient、OKhttp、Netty

## 微服务、分布式的好处，为什么要用？

## Condition原理，

## 分布式锁，实现方式

## Config原理，消息总线是什么

## 分布式事务、分布式锁

## zuul和gateway

## springboot加载流程

## springboot常见题

## SpringBoot加载过程

## Feign原理，SpringCloud没有优化HttpConnection创建么？

## Fegin、Ribbon原理

## CAP理论，分区容错性的意义

## HttpClient为什么要单例

## OKhttpClient和HttpClient区别

## 分布式会遇到什么问题？

1、多服务治理、通信问题

2、调用链路增长，性能减缓

3、事物管理

## **hystrix熔断之后，再次调用怎么处理？**

超过服务健康状态设置值时，熔断器开启，一段时间后，熔断器自动转换为半开状态，允许一个请求发送，如果这个请求发送成功，熔断器更新为关闭状态，否则，继续等下下次半开状态

## **eureka客户端和服务端如何连接**

客户端启动后会去找服务端，然后注册，服务端添加该客户端的服务信息，双层map存储，第一层村服务名，第二层存服务实例，如IP，端口号等，客户端如果没有缓存的服务列表，会去读取服务端服务列表，并缓存到本地。

客户端更新服务列表的机制：第一次注册到服务端时，没有服务列表，会从服务端拉取列表，之后客户端维护，创建新的线程，定期去服务端刷新服务列表

## **springcloud微服务之间传递数据，是否有加密、序列化措施？**

## **Auth、JWT**

# ========多线程&高并发=======

## 线程安全的3要素：原子性、有序性、可见性

## 为什么使用计数器栅栏、和future.get什么区别？

## forkThreadPool：

https://www.cnblogs.com/gaopeng527/p/4910508.html

## 什么是AQS

## 线程池原理，主要参数，参数意义

1、核心线程数

2、最大线程数

3、线程存活时间

4、存活时间时间单位

5、阻塞队列，队列大小

6、线程工厂

7、拒绝策略

## CountDownLatch原理，cycy\sampla原理

## Lock如何唤醒线程

## Executors创建线程池的缺点

## 如何保证线程安全

1、synchronized

2、lock

3、final不可变

4、cas

5、ThreadLocal

## java的SPI

## java高并发常见题

## 阻塞队列和非阻塞队列

## 常见的阻塞队列

## ArrayBlockingQueue

## LinkedBlockingQueue

## arrayqueue和linkedqueue区别

## 线程间通信

1、Synchronized-同步

2、volatile

3、pidInputStream、pidOutputStream

## semaphore信号量

1、读取速度快

2、

## countdownlatch（重写AQS，实现Sync）、

## cycliclebrray（基于reentranlock）

## AQS原理，使用场景

## lock和Synchronized区别

1、Synchronized基于JVM层面指令实现，lock是jdk类

2、Synchronized无法判断锁的状态，会阻塞其他线程，lock可以判断锁的状态，选择是否阻塞

3、Synchronized自动释放锁，lock手动释放

4、Synchronized是可重入、非公平、不可中断锁，lock是可重入、公平或非公平、可中断锁

5、Lock功能更多

coutDownLatch 底层实现了AQS

lock中AQS原理

ReenTrantLock实现公平锁、非公平锁原理，写代码样例

aqs公平锁和非公平锁，哪个吞吐量高，为什么

非公平锁更快，因为减少了线程等待过程。

如何解决高并发问题

1、服务分布式处理

2、热点数据缓存

3、服务降级，避免用户涌入系统

4、MQ异步处理，解耦系统

3、线程池超过核心线程数的线程，为什么要放在队列中，而不直接放在核心线程数中执行？

1、等待核心线程执行完，提高线程复用性

2、线程池创建线程，需要获取mainlock全局锁，影响并发效率，阻塞队列很好的起到缓冲

线程池阻塞队列https://blog.csdn.net/nanruitao10/article/details/85114552

jdk8中新增的LongAdder类和AtomicLong的区别

共享锁

=============数据结构&算法===============

9、介绍快速排序，是不是稳定排序，介绍所知道的稳定排序

快速排序不稳定

稳定排序：相邻两个相等元素，在排序后，顺序不变，如a[1] = 2; a[2] = 2，排序完成后两者顺序不变（冒泡排序就是稳定排序）

稳定排序使用场景：要求先来后到的次序时（相当于排队，谁排在前，就一直在前）

11、算法题：（1）最大子序列和；（2）反转链表；（3）数组中查找两数之和为K的序列对

二叉树层次遍历：https://www.cnblogs.com/bigsai/p/11393609.html

数组存储二叉树

树：B树，B+树、二叉树、二叉平衡树、红黑树

5、什么是时间复杂度

代码处理的时间长度

6、介绍B树、B+树、红黑树

找链表的倒数第K个数

方案一：

1、循环得出链表的长度n

2、循环读取第n-k个数，就是倒数k个数

方案二：

1、定义两个指针，一个指针先走，走到K位置时，第二个指针开始

2、当第一个指针为空时，此时第二个指针指向的值就是倒数第K个数

一本字典找出出现频率最高的十个单词

很多数，找前10个（分治，堆排序）

1、堆排序：升序-大顶堆，降序-小顶堆

2、创建长度为10的链表，顺序存储前十个，后面循环比较

堆排序算法

怎么实现链表反转

1、从头部循环，将数据写入结尾

2、双向链表，直接反转

怎么判断链表是否有环

1、定义个map，每次遍历的时候比较是否存在，存在的话就说明有环

2、定义两个指针，一个一次走一步，另一个一次走两步，如果走一步的追上了走两步的，说明存在环

求数组的最大子序列和

堆和栈的定义

堆先进先出

栈先进后出（递归处理会造成StackOverFlowError）

快排，口述

以第一个数为基准，从前找比它大的，从后找比它小的，调换位置，直到前后循环碰头，将第一个基准数放到碰头的位置，然后用同样的方法，操作碰头位置两端的数组，递归，直到排序完成

二分查找

中序遍历

使用过哪些算法？为什么使用

# ========网络协议==========

10、UDP、TCP、HTTP

UDP：不可靠连接，会发生丢包

TCP：可靠连接，连接三次握手，断开四次挥手

HTTP：基于TCP可靠连接建立

osi七层模型

在浏览器输入网址后，执行的全部过程

1、DNS域名解析，找到服务器对应的IP

2、TCP三次握手和服务器建立连接

3、发送http请求到服务器

4、服务器接收并处理请求

5、服务器返回请求结果

6、客户端接收到请求结果数据

7、通过四次挥手断开连接

8、客户端渲染页面

# =============IO===============

Netty

## =============javva===============

hashmap为什么使用红黑树，好处是啥

sortset获取不存在的value的score时 报错，为什么？看源码

java自动装箱-拆箱，Double（null）-->double报空指针，因为拆箱过程是通过Double.doubleValue()调用发放实现；

redis什么时间发生数据类型转换？

string使用final修饰的好处

1、不可变，保证了线程安全

2、不可变字符串放入常量池，提高性能

3、不可变，防止继承，破坏结构，因为jdk源码中大部分类都使用了string

http-jave基础文档中，连接池的相关概念

equals和hashcode的联系

7、jdk1.7和1.8的区别

2、组合和聚合的区别

人的身体由各种组合而成，但是可以将房子、车子通过聚合，聚合到人

3、反射应用场景

1、jdbc加载驱动

2、序列化，通过反射创建对象

3、通过类名，加载类对象

## ===============JVM & GC=================

6、JVM优化

熟悉的垃圾回收器，工作原理

垃圾回收器和、回收算法，什么时间stop the world

回收算法：

1、标记清理

2、复制算法

3、标记整理清除

4、分代收集

GC Root节点都有哪些？

1、方法区中静态变量引用的对象

2、发放去中常量引用的对象

3、栈中引用的对象

4、native（本地方法栈）引用的对象

fullgc是否可以回收方法区？可以

什么情况下引发fullgc？

1、system.gc();只是建议，并不一定执行

2、老年代内存不足

3、永久代内存不足

4、年轻代往老年代发送的数据平均大小大于老年代剩余内存大小

5、超大对象，老年代剩余内存不足以存放

fullgc接口导致接口响应慢，如何排插

垃圾回收器

如何定位代码性能问题

## =============redis===============

redis支持的数据类型，跳跃表，如何自己实现lru

redis数据统计如何实现-bitmap

（1）redis是多线程还是单线程？（回答单线程的请回吧，为什么请回，请往下看）

（2）redis的持久化开启了RDB和AOF下重启服务是如何加载的？(10个人9个回答错误)

（3）redis如果做集群该如何规划？AKF/CAP如何实现和设计？

（4）10万用户一年365天的登录情况如何用redis存储，并快速检索任意时间窗内的活跃用户？

（5）redis的5种Value类型你用过几种，能举例吗？

（6）100万并发4G数据，10万并发400G数据，如何设计Redis存储方式？

redis常见性能问题和解决方案

MySQL里有2000w数据, redis中只存20w的数据,如何保证redis中的数据都是热点数据

Memcache与Redis的区别都有哪些

redis最适合的场景

1、会话缓存

2、排行榜

3、热点数据缓存

Redis的同步机制了解么、redis集群如何保证一致性（CAP理论，只能保证两者）

是否使用过Redis集群,集群的原理是什么

1、Cluster；2、主从；3、哨兵模式

同步机制：

1、client节点启动后，发送sync命令到master

2、master库fork一个线程，执行数据快照保存（RDB），并缓存期间执行的命令

3、快照完成后，发送快照和缓存命令到从数据库

4、从数据库接收到快照和缓存命令后，载入

C:一致性

A:可用性

P:分区容错性（遇到任何一个分区出错时，任然能够保证服务的可用性，最基本要求）

## ===============MQ消息队列=================

MQ消费失败如何处理？消费端是拉数据，还是推数据

2、RabbitMQ为什么这样设计？和其他相关MQ的区别

1、RabbitMQ异步消息，发布订阅模式

2、RabbitMQ延迟消息

## ===============设计模式=================

简单工厂和抽象工厂有什么区别

https://www.jianshu.com/p/d27d698802fd

CPU，gpu

用户请求在服务器是一个请求一个线程么

tps：每秒系统能处理的请求量

剑指offer，letcode