1. **请简要描述类加载机制。jdk\_1.8较之前的jdk的虚拟机实现有什么不同，比较下。**

双亲委派,

加载、连接（验证、准备、解析）、初始化、使用、卸载

1.6 方法区通过非堆来实现，非堆。

1.7 方法区通过永久区来实现，且将永久区放入堆内存中。

1.8方法区通过元数据来实现，且存放在直接内存。常量池、静态变量放在堆中。

1. **现行的垃圾回收器有哪些？举例典型的使用场景。**

Serial适合客户端进程

Serial Old适合客户端进程

ParNew 适合作为服务端进程，现在服务端进程大多使用改来及会收器作为新生代垃圾会收器

Parallel Scavenge 适合做一些批量处理服务（新生代）

Parallel Old 适合做一些批量处理服务（老年代）

CMS 适合做一些高并发业务场景服务

G1 （CMS升级版）一些高并发业务场景。适合多核CPU，大内存，尽量保证垃圾回收期停顿时间最短。

1. **在SpringBoot演进得到的微服务框架SpringCloud有哪些组件。请描述各自的功能。**

Eureka 服务注册发现

Ribbon 客户端负载均衡

Hystrix 断路器（资源隔离、降级、限速）

Zull 服务网关（路由，鉴权、限速、负载均衡）

Config 分布式配置中心

1. **CAP定理是什么内容？是否接触过成熟的分布式事务方案？**

CAP

C一致性 A可用性 P分区容错性，所有的共享数据库服务只能满足其中两点。

分布式事务

* 2PC：需要一个事务协调器;第一步询问各个执行者，各执行这校验业务并准备相应资源且给资源上锁。第二部执行或回退。
* 3PC：需要一个事务协调器，执行者校验，准备资源加锁分两步，最后一步执行或回退。
* TCC： try（检查所有的业务，预留空间） confirm（执行业务） cancel（回滚业务）
* 基于可靠消息最终一致性
* 最大努力通知最终一致性
* 基于本地消息最终一致性

1. **熟悉的数据库有哪些？描述下Mysql的引擎及选择原则。描述下Mysql索引的原理和实现。**

Oracle db2 Mysql

InnerDB :支持事务，行锁，支持高并发

Myisam：不支持事务，表锁，查询速度快，支持全文检索。

Memory ：内存，类似redis

B+Tree ： 树的形式，叶子结点存放数据，叶子结点是以链表的形式相互连接，有序存储。分页存储。

Hash：通过计算hash值，hash碰撞则遍历链表。

1. **Java中锁的实现方式。请简单陈述锁的状态转换。synchronized修饰锁独占嘛？volatile关键字的作用。**

new、runnable、waiting、timed waiting、blocked、terminated

Synchronized 是独占锁

Volatile 可见性、禁止指令重排

1. **java中常见的容器，并挑出熟悉的简要描述其特征。**

HashMap：数组+链表+ 红黑树，key+value存储，key可以为空

HashSet：hash集合,数组+链表结构，无序。

TreeMap：有序的key，key不能为null

Hashtable：线程安全的hashmap，key不能为null

ConcurrentHashMap：支持高并发的hashMap，线程安全的，key不能为null

1. **描述下tcp三次握手，四次挥手；**

三次握手

* 第一次：A发送消息给B让B确认是否可以收到A的消息；
* 第二次：B在发送消息给A，确认能够收到A的消息+询问A是否可以收到B的消息
* 第三次：A回复B能够收到B的消息

四次挥手

* 第一次：A发送消息给B断开连接。
* 第二次：B回复A，OK没问题断开连接。（B继续发送数据）
* 第三次：B发送消息给A断开连接。
* 第四次：A回复B，OK没问题断开连接。

1. **请从协议和数据传输层面描述一次完整的HTTP请求过程；**
2. 建立TCP连接
3. Web浏览器向web服务器建立连接
4. 浏览器发送请求信息。
5. 服务器应答返回状态
6. 服务器应答头信息（响应服务信息、正文长度、编码格式…）
7. 服务器向浏览器发送数据
8. 服务器关闭TCP连接
9. **请用java伪代码实现“生产者-消费者-仓储”,请重点突出使用的数据结构和主要API。**

仓库声明一个LinkedBlockingQueue（10）

生产者线程put(元素)

消费者线程take()

1. **熟悉的大数据框架有哪些？请挑出最熟悉的加以阐述。回答一下问题。例如:**

**a、zookeeper的选主流程；**

**b、hdfs文件上传及下载的流程；**

**c、消息队列的作用和常见的选型。**

**C:消息队列可用于应用解耦，峰值降低等。**

熟悉的大数据框架：Zookeeper、kafka

Zk选主流程：

默认选自己，当收到别人的投票是，将该投票与自己比对。比对ZXID（事务ID）、SID（server id）。最终结果按照大多数来比对。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 特性 | RabbitMq | ActiveMQ | RocketMq | Kafka |
| 事务 | 支持 | 支持 | 支持 | 不支持 |
| 性能 | 一般 | 低 | 一般 | 高性能 |