SAP面经

对web3.0的认知，和1.0、2.0的区别

- **Web 1.0**：静态互联网，平台创造、平台所有、平台控制、平台受益

  由平台单方面向用户输送信息，平台输送什么信息用户就看什么信息

  代表：静态网页、传统门户网站

- **Web 2.0**：交互式互联网，用户创造、平台所有、平台控制、平台受益

  平台只提供基础设施及环境，由平台创造的内容较少，鼓励用户自我创作，但整个创作环境由平台掌握，例如微博可以删除和修改用户发布的任意内容，直播平台所创造的收益分配由平台说的算，用户在平台上产生的数据也由平台占有。

  代表：微博、facebook

- **Web 3.0**：协议互联网，用户创造、用户所有、用户控制、协议分配利益

  用户所创造的数字内容归属权和控制管理权明确为用户所有，其所创造的价值，根据用户与他人签订的协议进行分配。

  代表：区块链、元宇宙（数字资产权益由NFT确定与保障）

如何优化购物网站的性能（高峰时期并发订单较多）

提升系统高并发能力的方法：

- 提升单机处理能力：如增加CPU核数、升级更好的网卡、扩充硬盘或内存、使用cache、使用异步、使用无锁数据结构
- 增加服务器数量
- 系统集群化部署、负载均衡
- 读写分离（主库写，从库读）+数据库分库分表+分布式数据库
- 缓存：本地缓存、分布式缓存
- 消息中间件：可对不同业务的系统进行解耦
- 应用拆分（微服务）：按业务拆分，减少耦合，分级部署
- CDN（内容分发网络）：实时地根据网络流量、负载状况、到用户的距离、响应时间等综合信息将用户的请求重新导向离用户最近的服务节点上

数据库锁的几种分类方式

按锁住的范围

- 表级锁：一锁就锁住整张表，在并发写的情况下性能很差，是针对非索引字段加的锁（如果update、delete语句后的where条件中字段没有命中唯一索引或索引失效，那就会加表级锁），加锁快，不会出现死锁
- 行级锁：只对若干行进行上锁，是针对索引字段加的锁，加锁开销大，加锁慢、会出现死锁

按加锁模式（行级锁才有）

- 记录锁：对表中的一条条已存在的记录加锁，存在于唯一索引或主键中，如下面的句子，注意id必须是唯一索引列或主键列（主键列也是一种特殊的唯一索引列），否则会变成临键锁，并且查询语句必须为“=”这种精准匹配，不能为>< like，否则也会便为临键锁

```
SELECT * FROM `test` WHERE `id`=1 FOR UPDATE;
```

- 间隙锁：锁住一个范围，不包括记录本身（开区间），存在于非唯一索引中
- 临键锁：记录锁+间隙锁,锁定一个范围，包含记录本身（左开右闭，闭的就是一个记录），存在于非唯一索引中，可以解决幻读问题

InnoDB默认的隔离级别为可重复读，行锁默认用的是临键锁，但当操作索引是唯一索引或主键时，会降为记录锁

按兼容性（表级锁及行级锁都有）

- 共享锁（读锁，S锁）：事务在读取记录的时候获取共享锁，允许多个事务同时获取（锁兼容）。
- 排他锁（写锁，X锁）：事务在修改记录的时候获取排他锁，不允许多个事务同时获取。如果一个记录已经被加了排他锁，那其他事务不能再对这条事务加任何类型的锁（锁不兼容）。

	S 锁	X 锁
S 锁	不冲突	冲突
X 锁	冲突	冲突

意向锁（表级锁才有）

当要用到表级锁的时候，用来判断表中的记录是否有行级锁

- **意向共享锁（Intention Shared Lock，IS 锁）**：事务有意向对表中的某些记录加行级共享锁（S 锁），加共享锁前必须先取得该表的 IS 锁
- **意向排他锁（Intention Exclusive Lock，IX 锁）**：事务有意向对表中的某些记录加行级排他锁（X 锁），加排他锁之前必须先取得该表的 IX 锁

意向锁之间是互相兼容的。

	IS 锁	IX 锁
IS 锁	兼容	兼容
IX 锁	兼容	兼容

意向锁和共享锁和排它锁互斥（这里指的是表级别的共享锁和排他锁，意向锁不会与行级的共享锁和排他锁互斥）。

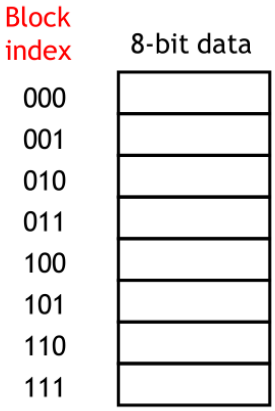
	IS 锁	IX 锁
S 锁	兼容	互斥
X 锁	互斥	互斥

操作系统中缓存的原理、作用

操作系统中最常用的缓存为高速缓存存储器cache，cache的访问速度很快，将一些频繁使用的数据存在cache中，当查询数据时先到cache里找，找不到再到内存或数据库找，从而加快访问速度，减轻服务器压力。

底层原理：

cache通常被分为多个行（通常为2的幂），每个行和主存中一个块大小相同



信息存在主存和cache中，确定它们的对应关系就需要借助地址映射（由硬件实现，将主存地址映射到对应的cache位置，让编程人员可以直接通过主存地址来得到cache里的数据，从而感觉不到cache的存在）

- 直接映射：每个主存块只能放到cache的一个特定行中，一般用`内存块号%cache总行数`的方式决定映射到哪一行

  (实现简单、成本低；灵活性差：每个主存块只能放到特定行，冲突概率高，如果cache容量小的话会发生频繁替换，适合容量大的cache)

- 全相联映射：主存中任意一个块可以映射到cache中任意一行。需要在cache中的一行增加标记部分，存放该行内容的主存块的块号。

  （灵活：只要cache有空行就可以存新进来的块；效率不高：需要额外存主存块号；速度慢：每次访问cache都需要一个一个遍历找目标行，适合容量小的cache）

- 组相联映射：全相联映射和直接映射的折中方案，将cache按相连行号划分为若干组，主存的每一块必须放到特定的组，但可以放到组内任意一行。

设计模式有什么

工厂、抽象工厂、单例、模板方法、观察者、适配器、装饰、代理.....

进程和线程，死锁，多线程编程的难点和解决

node.js和 js的区别

node.js 和 javascript 从本质上来说没有什么区别，语言是一样的，都是javascript语言编写。但是，node.js 主要从事后台操作，javascript主要操作HTML的元素 (前端)

node.js是一个基于Chrome JavaScript运行时建立的平台，它是对Google V8引擎进行了封装的运行环境；简单的说node.js就是 把浏览器的解释器封装起来作为服务器运行平台，用类似javascript的结构语法进行编程，在node.js上运行。

node.js怎么实现多线程

通过cluster、child_process API 创建子进程

方法重载和重写

- 重写（override）：子类重新编写父类的同名方法，参数列表必须同，子类方法返回值类型应比父类方法返回值类型更小或相等，抛出的异常范围小于等于父类，访问修饰符大于等于父类。发生在运行期。
- 重载（overload）：发生在同一个类中，方法名相同，参数列表必须不同，返回值和访问修饰符可以不同。发生在编译期。

自我介绍

大家好，我叫杨雨佳，本科和硕士均就读于南京大学软件学院，我成绩优秀，本科时位于年级前30%，研究生阶段位于年级前25%，我最擅长的编程语言是java，也使用过python、c#、mysql等其他语言，我做过很多项目，如web应用程序、游戏、微信小程序等。

Hello everyone, my name is Yang Yujia. I have been studying in Nanjing University Software Institute for both my undergraduate and master's degrees.

I have excellent grades, ranking in the top 30% of my undergraduate studies and in the top 25% of my graduate studies.

My best programming language is Java, and I also have used python, mysql and other languages. I have worked on many projects such as some web applications, games, and WeChat mini programs.