|  |
| --- |
| 技术问题(至少50个) |
| 事务篇：  问题1：事务的  问题1：对Spring事务的理解  回答：  问题：对分布式事务了解吗？  回答：简单的说，就是一次大的操作由不同的小操作组成，这些小的操作分布在不同的服务器上，而且属于不同的应用，分布式事务需要保证这些小的操作要么全部成功，要么全部失败，分布式事务就是为了保证不同的数据库一致性。  问题：你是如何解决分布式事务的？  回答：  1.两阶段提交方案/XA方案： 举个例子：公司每隔一段时间都会搞一次团建，团建就相当于是一次事件，组织者也就是负责人，第一个阶段：他回询问各自的意见（比如果去哪，都有啥活动），如果都ok就去做这件事，如果其中一个人不ok就取消。第二个阶段：  前提是第二个都ok，大家就可以一起完成了。 这里的负责人就是个事务管理器，主要负责协调多个数据库的事务，保证了数据的一致性，首先去执行各地的本地事务，然后就可以正式提交了整个事务了，如果有一个数据库执行失败了，就需要回滚。可以使用spring+JTA来实现。  缺点：这种方案适用于单体应用，（一个应用连接多个数据源，会跨库操作，严重依赖于数据库操作事务，操作效率低，不适合用于高并发场景，现在都用微服务了，主张的是一个应用连接一个数据库，不允许跨库操作）   1. TCC方案（强一致性）补偿性事务:属于两阶段提交方式的一个变种。T（Try）：对各个服务的资源做确认，及对资源锁定和保留。C（Confirm）：各个服务实际的操作。C（Cancel）：只要有其中一个服务的业务方法出现错误，就需要补偿（即已经执行成功的操作进行回滚）   举个例子：跨行转账：T：先将两个银行账户的资金冻结；C：实际进行转账，A减钱，B加钱；C：如果有任何银行的操作失败了，就需要回滚补偿。比如：A已经扣款成功，B在增加资金的时候失败了，此时就需要把A的钱加回去。  缺点：回滚操作是严重依赖于开发人员的代码进行回滚或者补偿的，代价比较大，所以适应场景是有限的。一般是要求数据一致性要求特别高的时候才用到TCC方案，比如说银行关于钱的一些业务。  3.本地消息表：1）A系统在自己的本地事务操作的同时，会插入一条数据到消息表  2）接着A系统会把找个数据发送到MQ，A系统就是消息的生产者  3）B系统接收到消息（B系统是个消息的消费者），在自己的本地事务中，往自己的本地消息插入一个数据，同时执行自己的业务，如果消息被处理过了，那么这个事务就需要回滚。  4）B系统执行成功后，会更新自己的本地消息表及A系统的本地消息表的状态  5）如果B系统处理失败了，首先不会更新自己的本地消息表，A系统会自己定时扫描自己的消息表，如果发现数据没有改变成成功的状态，会往B继续发送消息。  6）这个方案保证的是消息的最终一致性，即使B失败无数次，A会一直发送消息，直到B成功。A和B并不是严格一致的，而是最终一致的，我们在这里牺牲了一致性，换来了性能的大幅度提升。当然，这种玩法也是有风险的，如果B一直执行不成功，那么一致性会被破坏。  缺点：这个方案严重依赖数据库的消息表来管理事务，不适合于高并发的场景，实际使用少。  4.可靠消息的最终一致性：如果没有本地消息表，直接使用MQ来完成整个事务的操作，RocketMQ  1）A系统先发送一条prepared消息到MQ，如果消息发送失败，就取消操作别执行  2）如果发送成功，MQ会给A发送回调消息，那么A系统就可以处理自己的本地事务，如果失败了，告诉MQ进行回滚。  3）A系统执行成功给MQ发送确认消息，此时B系统就会接收到这个消息然后执行本地事务  注：一般一个项目用到分布式事务也就两三个场景，全用分布式事务的话成本太大，会导致项目特别复杂。  问题：分布式事务如何保证事务的一致性：  回答：  多线程篇：  问题2：进程和线程的区别  回答：进程是正在运行的应用程序，进程是线程的集合。  线程就是一条执行路径，一个独立的执行单元。  多线程就是为了提高程序的效率，多线程之间每个线程互不影响。  问题3：对线程的应用场景：多线程下载，QQ，爬虫，前端开发ajax（异步上传），分布式job（需要同时一个执行多个任务调度）、使用多线程体现程序的效率。  问题4：线程的5种状态：   1. 创建（new Thread()）、 2. 就绪（调用线程start()方法）、 3. 运行（当线程获得CPU事件后，才进入运行状态，真正开始执行run()方法）、 4. 阻塞（由于各种原因进入阻塞状态：1>线程通过调用sleep()方法进入睡眠状态；   2>线程调用一个在I/O上被阻塞的操作，即该操作在输入输出操作完成之前不会返回到它的调用者；3>线程试图得到一个锁，而该锁正被其他线程持有；4>线程在等待某个触发条件。）   1. 死亡（两个原因会导致线程死亡：1）run方法正常退出而自然死亡，2）出现异常而未被捕获，从而终止了run方法而使线程猝死。   为了确定线程在当前是否存活（要么是可运行的，要么是被阻塞了），需要使用isAlive()方法。如果是可运行或被阻塞，这个方法返回true；如果线程仍旧是new状态且不是是可运行的，或者线程死亡了，则返回false；  问题4：  问题：拦截器和过滤器有哪些区别？  回答：  1.拦截器是基于java反射机制的，过滤器是基于函数回调  2.拦截器只对action请求起作用，而过滤器可以对几乎所有请求起作用  3.拦截器不依赖于servlet容器，过滤器依赖于servlet容器  4.在action生命周期中，拦截器可以多次被调用，而过滤器只能在容器初始化的时候被调用一次。  ....... |
| **人资问题(至少30个)** |

|  |
| --- |
| 您好，我叫刘佳鑫，  辽宁铁岭人，今年28岁，  2014年6月份 毕业于沈阳工程学院，毕业后去外地工作，工资不高，听朋友介绍it行业前景好，工资高。  培训经历：2014.10--2015.10，it培训1年  第一家公司：2015.10---2016.11 畅捷通信息技术股份有限公司  第二家公司（外包）：2016.12---2019.04，北京华路时代信息技术股份有限公司 项目结项了  北京市海淀区上地三街9号B座4层B407室。（做完了结项了）  （大概3年半工作经历）  上一个项目做的是金融，  做的是p2p的一个信用借贷的项目，这个项目用到了项目管理：项目使用maven来管理前后台两个子系统系统项目 数据库：MySQL+Redis 持久化层：MyBatis 业务层：spring Web：SpringMVC 前端：jQUery+Bootstrap3  项目一共分为前台后台两个系统，前台用户系统，后台管理员系统，我参与的项目有前后台的个人信息绑定，用户实名制认证，前后台登陆注册，借款还款等等。  问题1：你的离职原因是啥  回答：项目结项了。  问题2：之前是做什么的？  回答：电厂集控值班员。  问题3：哪个大学毕业的？  回答：毕业于沈阳工程学院  问题4：学的什么专业？  回答：热能与动力工程  问题5：之前在哪儿工作？  回答：重庆綦江区  问题6：学历学信网可查吗？  回答：可查  问题7：说说你的经历  回答：2014年6月毕业，之后就到重庆工作了，干了三个月，那儿的生活不是我想要的，工资低，危险。正好有朋友搞it而且已经  问题8  回答：  问题9  回答：  问题10  回答：  问题11  回答：  问题12  回答：  问题13  回答：  问题14  回答：  问题15  回答：  问题16  回答：  问题17  回答：  问题18  回答：  问题19  回答：  问题20  回答：  问题21  回答：  问题22  回答：  问题23  回答：  问题24  回答：  问题25  回答：  问题26  回答：  问题27  回答：  问题28  回答：  问题29  回答：  问题30  回答：  回答3：  ........ |