### 1.对微服务有何了解

微服务，又称微服务架构，是一种架构风格，它将应用程序构建以**业务领域**为模型的小型自治服务集合。

### 2.微服务架构有哪些优势

**独立开发---**所有微服务都可以根据各自的功能轻松的开发

**独立部署--**基于其服务，可以在任何应用程序中单独部署它们

**故障隔离--**即使应用程序中的一项服务不起作用，系统仍可运行

**混合技术堆栈-**-可以使用不同的语言和技术来构建同一应用程序的不同服务

**粒度缩放-**-单个组件可根据需要进行缩放，无需将所有组件缩放在一起

### 3.微服务的特点

解耦、组件化、业务能力、自治、持续交付、责任、分散治理、敏捷

### 4.springcloud解决了哪些问题

与分布式系统相关的复杂性

处理服务发现的能力

解决冗余问题

负载均衡

减少性能问题

### 5.springCloud和dubbo 有哪些区别？

Dubbo底层是使用Netty这样的NIO框架，是基于TCP协议传输的，配合以Hession序列化完成**RPC通信**。而SpringCloud是基于**Http协议+rest接口**调用远程过程的通信，相对来说，Http请求会有更大的报文，占的带宽也会更多。但是REST相比RPC更为灵活，服务提供方和调用方的依赖只依靠一纸契约，不存在代码级别的强依赖，这在强调快速演化的微服务环境下，显得更为合适，至于注重通信速度还是方便灵活性，具体情况具体考虑。

### 6.springboot和springcloud，请你谈谈对他们的理解

Springcloud是spring为微服务架构思想做的一个一站式实现。Springcloud是基于springboot的一套实现微服务的框架，它提供了微服务开发所需的配置管理、服务发现、断路器、智能路由、微代理、控制总线、全局锁、决策竞选、分布式会话和集群状态管理等组件。最重要的是，跟SpringBoot框架一起使用的话，会让你开发微服务架构的云服务非常方便。

### 7.什么是服务熔断？什么是服务降级？

熔断机制是对应雪崩效应的一种微服务链路保护机制。当某个微服务不可用或者响应时间太长时，会进行服务降级，进而熔断该节点微服务的调用，快速返回“错误”的响应信息。当检测到该节点微服务调用响应正常后恢复调用链路。在SpringCloud框架里熔断机制通过Hystrix实现，Hystrix会监控微服务间调用的状况，当失败的调用到一定阈值，缺省是5秒内调用20次，如果失败，就会启动熔断机制。

服务降级，一般是从整体负荷考虑。就是当某个服务熔断之后，服务器将不再被调用，此时客户端可以自己准备一个本地的fallback回调，返回一个缺省值。这样做，虽然水平下降，但好歹可用，比直接挂掉强。

### 8.Eureka和zookeeper都可以提供服务注册与发现的功能，请说说两个的区别？

Zookeeper保证了CP（C：一致性，P：分区容错性），Eureka保证了AP（A：高可用）

1.当向注册中心查询服务列表时，我们可以容忍注册中心返回的是几分钟以前的信息，但不能容忍直接down掉不可用。也就是说，服务注册功能对高可用性要求比较高，但zk会出现这样一种情况，当master节点因为网络故障与其他节点失去联系时，剩余节点会重新选leader。问题在于，选取leader时间过长，30 ~ 120s，且选取期间zk集群都不可用，这样就会导致选取期间注册服务瘫痪。在云部署的环境下，因网络问题使得zk集群失去master节点是较大概率会发生的事，虽然服务能够恢复，但是漫长的选取时间导致的注册长期不可用是不能容忍的。

2.Eureka保证了可用性，Eureka各个节点是平等的，几个节点挂掉不会影响正常节点的工作，剩余的节点仍然可以提供注册和查询服务。而Eureka的客户端向某个Eureka注册或发现时发生连接失败，则会自动切换到其他节点，只要有一台Eureka还在，就能保证注册服务可用，只是查到的信息可能不是最新的。除此之外，Eureka还有自我保护机制，如果在15分钟内超过85%的节点没有正常的心跳，那么Eureka就认为客户端与注册中心发生了网络故障，此时会出现以下几种情况：

①、Eureka不在从注册列表中移除因为长时间没有收到心跳而应该过期的服务。

②、Eureka仍然能够接受新服务的注册和查询请求，但是不会被同步到其他节点上（即保证当前节点仍然可用）

③、当网络稳定时，当前实例新的注册信息会被同步到其他节点。

因此，Eureka可以很好的应对因网络故障导致部分节点失去联系的情况，而不会像Zookeeper那样使整个微服务瘫痪。

### Zookeeper 面试相关

### 1.Zookeeper 是什么？

Zookeeper 是一个分布式，开源的分布式应用程序协调服务，是Google的Chubby一个开源的实现，它是集群的管理者，监视着集群各个节点的状态根据节点提交的反馈进行下一步合理操作。最终，将简单易用的接口和性能高效、功能稳定的系统提供给用户。

客户端的读请求可以被集群中的任意一台机器处理，如果读请求在节点上注册了监听器，这个监听器也是由所连接的Zookeeper机器来处理。对于写请求，这些请求会同时发给其他Zookeeper机器并且达成一致后，请求才会返回成功。因此，随着Zookeeper的集群机器增多，读请求的吞吐会提高但是写请求的吞吐会下降。

有序性是Zookeeper中非常重要的一个特性，所有的更新都是全局有序的，每个更新都有一个唯一的时间戳，这个时间戳称为zxid(Zookeeper Transaction Id)。而读请求只会相对于更新有序，也就是读请求的返回结果中会带有这个Zookeeper最新的zxid。

**2. Zookeeper提供了什么？**

1. 文件系统 2. 通知机制

**3. Zookeeper 文件系统**

Zookeeper提供一个多层级的节点命名空间（简称znode），与文件系统不同的是，这些节点都可以设置关联的数据，而文件系统中只有文件节点可以存放数据而目录节点不行。Zookeeper为了保证高吞吐量和低延迟，在内存中维护了这个树状的目录结构，这种特性使得Zookeeper不能用于存放大量的数据，每个节点的存放数据上限为1M.