1. 简单介绍下以前做过的项目

主要参与了广发证券企业客户信息管理系统的开发工作。

1. 项目的主要功能是以三要素为标准，将广发证券公司其他系统的账号信息归并成客户信息，其中三要素主要指的是客户姓名、客户证件类型、客户证件号码。
2. 涉及使用到的技术spring+springmvc+springBoot+mybatis
3. 数据库使用的是oracle,服务器使用的jetty
4. 主要的参与工作内容
5. 主要负责调度平台的数据处理工作。
6. 主要负责大数据平台的数据统计工作。
7. 主要负责厂商java代码和oracle存储工程的核查和测试工作。

主要参与了广州农商银行的绩效管理系统

1. 项目的主要功能是作为一个绩效考核的系统，是以手工数据和其他源头系统数据作为基础，以kpi作为一种载体，对机构、部门、柜员以及客户经理进行绩效考评和管理。
2. 涉及使用到的技术spring+springmvc+springBoot+mybatis
3. 数据主要是使用oracle，服务器主要使用的是tomcat、weblogic
4. 主要参与工作
5. 主要负责对公模块开发工作
   1. 绩效工资
   2. 营销费用
   3. 报表模块
6. 信用卡模块
   1. 主要做的是一个流程审批工作。

2、技术问题

1、定时器的使用

1、springboot @Scheduled注解

2、quartz,主要是通过和spring整合后使用，需要创建专门的配置文件spring-quartz.xml,需要在配置文件中定义job,定义触发器trigger

3、利用jdk提供的new Timer().schedule

2、微服务注册中心有哪些

1、如果是使用dubbo的话，一般是使用zookeeper

2、如果是使用springcloud的话，一般是使用eureka,

主要区别zookeeper主要协议使用的是rpc,主要使用的是主从复制，p(容错性) c(强一致性)，采用的master节点和slave节点，加入master节点挂掉了，需要重新选举主节点，有可能会因为网络问题造成系统崩溃。

eureka主要使用协议是http，主要使用的是对等复制，p(容错性)、a(高可用性)，每个节点都是平等的，每个节点都可以接收读写操作。

3、redis数据类型

1、是一个非关系型的内存数据库，数据存储在内存中，也是可以持久化到磁盘中，因为具有高效的读写效率，经常用来做缓存，主要是以键值对的方式存在，String(字符串)、hash(哈希)、list(列表)、set(集合)、Zset(sorted set 有序集合)

4、外部接口怎么管控

1、那我之前的项目中对于这类的接口有用过自定义注解标注在接口上，在注解中定义访问时间，该时间内允许访问的最高次数，然后通过相应的拦截器来处理解析这个注解，控制该注解标注的接口的访问频率。防止恶意刷接口。

2、设置黑名单。

3、通过定义接口方式签名。

4、像一些大厂的对外接口使用token接口验证的方式

接口调用需要先去他们那边拿到token值，然后token还有过期时间啊，接口每天的调 用次数等限制。

5、再或者有些情况可以让前端加上验证码什么的，ip限制，ip白名单什么的。

5、限流

在开发高并发系统时常常会考虑这些问题：缓存、降级和限流。

我理解的限流就是限制系统的输入和输出流量，设置一个需要限制的阈值，一旦系统的 吞吐量到达这个阈值，就需要限制流量并采取一些措施以完成限制流量的目的。比如： 延迟处理，拒绝处理，或者部分拒绝处理等等

我使用过Java 并发库 的Semaphore完成信号量控制进行并发数量限制。

还有像我上面提到的接口管控中用到的，自定义注解+拦截器+Redis实现限流。这种就 是适用于分布式系统的全局限流。原理类似于计数器算法。

还有一些限流控制速率的方案，像是使用限流算法：漏桶算法和令牌桶算法。

还了解过nginx 限流，通过在nginx里配置并发数和连接数来完成限流。

6、代码优化

1、代码优化是每天都需要做的工作

1、重复代码抽取，抽取成公共函数

2、代码性能优化，考虑到逻辑是否是最佳的性能、考虑到更好的实现方式

3、另外还需要有代码规范的优化，像单例的使用啊，静态变量的使用啊，java对象的创建和回收啊，IO操作流的关闭等等很多小的规范和细节需要平时经验的积累。

7、内存泄露怎么处理

1、内存泄漏主要是指的某些内存空间使用完毕未能及时回收的现象，java内存的释放，主要是依靠gc执行，在编写代码过程中如果有不需要再使用的对象可以赋值null,让gc回收，数据库连接、io、网络连接使用完后，要调用close()方法关闭资源。

2、分析方面：

1、可以使用linux的top命令观察cpu的使用状态

2、可以使用jdk自带的jconsole、jvisualvm等工具追踪内存使用、gc回收情况。

3、可以jstat和jmap去dump出JVM内存映像，然后使用内存分析工具分析dump文件

8、慢sql怎么处理。

1、生成慢sql日志文件方式

再my.ini中加上下面两句话

log-slow-queries = e:\mysql5.5\mysql\_slow\_query.log

long\_query\_time=10

前面一句是设置慢查询日志存放路径，第二句是指多少秒以上算慢查询，上面的语句，就是指10秒。然后show variables like '%slow%';就行了

2、show命令找出慢日志查询语句，然后使用explain进行语句分析

3、最后优化查询语句

9、消息队列

1、消息队列主要作用是解耦、异步、销峰。

2、主要是由生产者和消费者组成，2种方式的主题，第一中是订阅/发布模式（主题模式），这种模式的特点是一对多的情况，消费者在线的情况下都可以收到消息，生产者不保存消息，在时间上有一定的相关性。

第二种模式是点对点模式（队列），队列是一对一，发送和接受是异步的，一个消息只能被消费一次，时间上没有相关关系，消息被消费后，生产者不再保持消息。

3、高可用性：

1. 消息的持久化

队列模式服务器宕机，消息还是会存在的

1. 事务
2. 生产者事务：假如需要使用手动事务时，则需要手动关闭事务，建议使用手动事务，出现异常会回滚。
3. 消费者事务：如果使用手动事务，则需要手动关闭事务，否则会出现消息重复消费现象。

3、签收机制

策略：自动签收、手动签收、允许重复签收

非事务签收：

自动签收

手动签收

开启手动签收时，需要调用acknowledge()方法。

事务签收：

自动签收

手动签收：

开启手动签收时，有commit,不需要调用acknowledge()方法。

4、使用集群实现高可用

4、持久化机制有哪些？

Jdbc\kahaDB\levelDB\amq

KahaDB

KahaDB activemq5.4默认的存储机制，基于日志文件和索引文件存储它所有的地址

有五个文件

Da-1.log --数据被追加到data logs中，当不再需要log文件中的数据的时候，log文件会被丢弃

db.data 索引文件

Db.free

Db.redo 备份

Lock 锁

amq --基于文件形式存储

jdbc --性能慢，存储在数据库可靠性好一点

1、需要修改配置问及那

2、建立一个数据库

3、会产生三张表activemq\_msgs(生产者存放的信息)、activemq\_acks(存放订阅 者信息)、activemq\_locks

4、代码一定要开启持久化，数据库才会有数据

5、优化

activemq journal 使用高速缓存写入技术，大大提高了性能,当消费者的消费速度能够及时跟上生产者生产的消息，journal则文件能够大大减少写入数据库的数据

<persistenceAdapter>

<journalPersistenceAdapterFactory

journalLogFiles="4"

journalLogFileSize="32768"

useJournal="true"

useQuickJournal="true"

dataSource="#mysql-ds"

dataDirectory="activemq-data"/>

</persistenceAdapter>

levelDB --activemq5.8后会引入，也是基于文件形式存储，效率会比levelDB好，未来的趋势。

1. activemq端口
2. activemq默认的进程端口是61616
3. activemq前台端口是8161
4. activemq的boroker

activemq的broker,相当于一个activemq实例，是嵌在代码中的

1. activemq的传输协议

tcp、nio、udp、ssl、http(s)、vm,默认是tcp,常用的有tcp、nio、还有auto+nio（nio加强版），配置文件activemq.xml

1. activemq幂等性问题，保证消息不被重复消费

可以给每条消息加一个标识，先把消息存入redis,key当作主键，消费前先去redis查询，如果有则可以不消费。

9、acrivemq重复消费机制

1、为什么会出现重复消费？

1、client用了transactions且在session中调用了rollback()

2、Client用了transactions且在调用commit()之前关闭或者没有commit

3、Clinet在Client\_acknowLedge的传递模式下，在session中调用了recover()

2、请说说消息重发时间间隔和重发次数？

间隔： 1s

次数：6

3、有毒消息

投递失败和重复消费的消息会被标识为有毒消息，有毒消息会存放在死信队列（DLQ）

4、死信队列的消息要怎么处理。

1、会有过期时间，自动删除

2、队列满的情况可以把消息放置在死信队列

3、可以人工干预从新发送

10、activemq和kafka的区别

1、activemq是一种传统mq,要支持一些标准接口和一些标准协议。

2、kafka最初始的设计初衷是高吞吐+高性能。

11、设计模式

单例设计模式：

工厂设计模式：

1. 简单工厂设计模式
2. 工厂方法模式
   1. 抽象一个不同牌子的手机
3. 抽象工厂模式
   1. 抽象不同牌子的不同产品

策略模式：定义是定义一组算法，将每个算法封装起来，并使他们之间可以转换。

例如：线程Thread的设计是，定义了Runnable接口，具体的行为逻辑可以通过实现这个接口，重写run方法实现。再在Thread引用这个方法

1. 代理模式：spring，aop中使用到的代理模式
2. 观察者模式
3. 库存超卖模式

这种情况在什么秒杀啊，高并发场景经常会遇到，我之前有遇到过类似的情况，因为并发很高，如果直接操作数据库，就算利用数据量的各种锁机制控制或者版本号控制，这个读写量数据库性能压力还是很大的。所有我一般这种情况会考虑将存库从MySQL前移到Redis中，所有的写操作放到内存中，因为Redis的写性能和读性能都远高于MySQL，然后通过队列等异步手段，然后将所有写DB操作在单队列中排队，完全串行处理。当达到库存阀值的时候就不在消费队列，并关闭购买功能。这样来解决性能和超卖的问题。

1. mysql查询优化方式

1>. 当结果集只有一行数据时使用LIMIT 1

2>. 避免SELECT \*，始终指定你需要的列

3>. 使用连接（JOIN）来代替子查询(Sub-Queries)

4>. 使用合理的字段属性长度

5>. 尽可能的使用NOT NULL

6>. 固定长度的表会更快

7>. 拆分大的DELETE 或INSERT 语句

8>. 查询的列越小越快

有些where条件会导致索引无效：

Ø where子句的查询条件里有！=，MySQL将无法使用索引。

Ø where子句使用了Mysql函数的时候，索引将无效

Ø 使用LIKE进行搜索匹配的时候，模糊匹配不能放在前面，否则失效。如’%xxxx’

③配置优化

MySQL服务器全局参数优化

④分表（横、纵）

减少单表的数据量及将常用列和不常用列、大列和小列分到不同的表。

9、遵循最左匹配原则

1. spring、spring mvc 、springBoot 、springcloud的区别
2. spring是一个大家族框架，有很多衍生的产品boot、security、jpa、mvc等等

主要的核心思想是aop、ioc

1. springmvc ，主要是基于servlet的一个mvc框架，主要解决web的开发问题。
2. SpringBoot设计初衷是约定优于配置，简化了springmvc的开发，不过springboot最主要是专注于微服务接口的开发，和前端解耦。
3. Springcloud俗称是微服务全家桶，基于springboot提供了一套微服务解决方案，包括微服务注册与发现、配置中心、熔断等。