# 1、数据架构

## 1.1 数据库ER模型

## 1.2 数据库逻辑模型

## 1.3 数据库物理模型

# 2、接口

## 2.1 订单完成发货

OrderFacadeService

Boolean informGoodsDeliveryFinishedEvent(Long orderId)

## 2.2 退货工单审核不通过

OrderFacadeService

Boolean informReturnGoodsWorksheetRejectedEvent(Long orderId)

## 2.3 退货工单审核通过

OrderFacadeService

Boolean informReturnGoodsWorsheetApprovedEvent(Long orderId)

## 2.4 确认收到退货商品

OrderFacadeService

Boolean informReturnGoodsReceivedEvent(Long orderId)

## 2.5 退货入库单审核通过

OrderFacadeService

Boolean informReturnGoodsInputOrderApprovedEvent(Long orderId)

## 2.6 财务中心完成退货打款

OrderFacadeService

Boolean informRefundFinishedEvent(Long orderId)

## 2.7 通知订单中心订单发表了评论

OrderFacadeService

Boolean informPublishCommentEvent(Long orderId)

## 2.8 获取确认收货超过7天而且还没发表评论的订单

OrderFacadeService

List<OrderDTO> listNotPublishedCommentOrders()

## 2.9 批量通知订单中心订单发表了评论

OrderFacadeService

Boolean informBatchPublishCommentEvent(List<Long> orderIds)

# 3、开发架构

## 3.1 实现类图

### 3.1.1 订单状态管理（状态模式）

订单每次执行一个业务操作，会需要进行状态的流转，每次流转到一个状态，都同样需要执行3个操作：

1. 更新订单自己的状态
2. 更新订单退货记录的状态（可能）
3. 新增一条订单的操作日志

定义一个state接口，里面定义好这么3个方法，定义一个context，对订单的状态流转，全部调用context的业务方法，context业务方法中决定，在执行这个业务操作的时候，订单的状态流转到哪个state，创建那个state的实例，然后执行那个state的3个方法，完成那个状态下的逻辑处理。

这不就是典型的状态模式吗？context决定数据的状态如何流转 -> 通过state来执行流转到那个状态之后要执行的一些操作。

实现类图，我们不画了，之前给大家已经画过了，这个大家可以自己去画一下，我们就直接写代码了。当然在公司里，肯定是像这种重要的实现类图，都是需要去画出来的。

### 3.1.2 价格计算（策略模式+抽象工厂模式）

大体上来说，在订单确认的页面：

1. 将每个商品的促销活动以及当前订单的优惠券给发送到后台去
2. 由后台，完成促销活动折扣金额的计算，完成赠品附加到订单中去，优惠券对整个订单的金额的折扣的计算，还要完成运费的计算
3. 在这里有这么一个思考，就是不同的促销活动如何计算，完全就是不同的策略
4. 如果不用策略模式的话，那么就会出现一大坨的if else，屎一样的代码，策略模式。
5. 运费的计算，实际上是找物流中心去现场计算的

策略模式+抽象工厂模式

每个商品都是有一个运费模板的，每个商品都绑定了促销活动，现在要做的事情，就是先遍历每个商品，对每个商品都计算出来对应的运费以及促销活动的折扣金额（可能不是计算折扣金额，而是附加赠品）

可以拿到一个总的，每个商品的促销活动折扣金额，同时赠品也处理完了，总的运费

也是对优惠券使用策略模式，获取优惠券类型对应的策略，然后计算出优惠券的抵扣金额

促销活动的金额、优惠券的金额、运费，商品总金额，本来就有

每个商品对应的赠品也处理好了

对每个商品的价格的计算，大致来说是需要几个策略的，总金额的计算；促销活动的处理；运费的计算 -> 一套策略的

总金额计算的策略 + 促销活动处理策略 + 运费计算策略 -> 根据不同的情况，可以组成不同的策略组合包，不同的策略之间，都是有关联关系的

促销活动如果是减免金额类型的：默认总金额计算策略 + 减免金额计算策略 + 默认运费计算策略

促销活动如果是赠品类型的：默认总金额计算策略 + 赠品处理策略 + 包含赠品运费计算策略

怎么创建出来一个一个的策略组合包，用抽象工厂模式，天然就是用于创建这种策略组合包的，比如说你现在要做个事儿，业务很复杂，里面需要同时使用多个策略，这不同的策略之间如何搭配都是有关联关系的

在满足某种情况下，要用一个策略组合包，用这套策略去执行整个业务逻辑

在另外一个情况下，要用另外一个策略组合包，要用另外一套策略去执行整个业务逻辑

### ~~3.1.3 构造器模式~~

~~构造这个订单数据~~

### ~~3.1.4 装饰器模式~~

~~装饰器模式，就是给某个目标类动态增加一些功能，可能会做一个装饰类包装那个目标类，然后在目标类的方法执行的前后，都完成一些增强的功能~~

~~咱们平时业务开发中，经典的用法，spring aop，动态代理包装目标类，然后可以织入我们想要的额外的增强功能：~~

1. ~~事务~~
2. ~~订单操作日志，记录一条订单操作日志，适合用装饰器模式来实现，spring aop，订单service织入一层增强，对我们指定的方法就可以自动插入一条订单操作日志~~

## 3.2 包设计

# 4、运行流程

## 4.1 提交订单

### 4.1.1 订单确认

从购物车页面，可以直接跳转到订单确认页，这个过程，走前端的，不是走我们后端的

1. 第一个计算接口，订单确认页一显示出来，直接前端都ajax请求找我们来算订单的各种价格，总金额、促销减免金额、运费
2. 第二个计算接口，在订单确认页上面，我们是要显示这个用户可以使用的优惠券的列表，然后用户可以选择一个优惠券，每次选择一个优惠券之后，前端再将优惠券的信息发送到我们的一个接口，计算出来，当前订单的应付金额之下，可以使用该优惠券抵扣掉多少钱

（3）订单确认页，别的就没什么了

在提交订单的确认页面中，需要显示一些数据 -> 购物车页面

1. 收货人列表，可以选择一个收货人+收货地址+联系电话（从会员中心去查询的）
2. 选择一个支付方式：仅仅支持支付宝/微信（写死的）
3. 商品列表：选择使用的促销活动、~~商品缩略图、~~sku编号、商品名称、销售属性、商品价格、商品毛重、商品长宽高、购买数量、库存数量、库存是否充足（从购物车点击结算的时候，将界面上的购物车数据，直接给带到了这个页面中来）
4. 赠品处理：还需要走订单系统的后台接口，去附加这个赠品信息，给订单补充一些赠品的数据，只不过将赠品的价格设置为0即可（走后台接口，去处理赠品信息）
5. 可以填写订单备注
6. 可以填写发票信息，填写~~发票的类型、~~抬头、纳税人识别号~~、内容~~
7. 可以选择是否使用优惠券（从促销中心加载这个会员领取的优惠券列表）
8. 显示价格明细：总价、运费、促销折扣、优惠券折扣 -> 最终算出一个总价格（商品总金额 - 促销折扣 - 优惠券折扣 + 运费）

点击提交订单按钮，就可以提交订单

### 4.1.2 订单价格计算

1、价格计算说明

购物车里选购的数据，显示在了订单确认页面里了之后，接着其实前台会走ajax请求，将价格计算需要的数据传递到后台，由订单中心来完成价格的计算：

这是一个接口

1、总金额：购买商品的金额 \* 数量

2、促销活动减免金额：每个商品，都可以选择某个促销活动，不选择，就会去计算，给这个商品的价格可以减免多少钱

3、赠品的处理：促销活动可能不是减免金额，可能是给你加一个赠品，此时就是在订单对应的商品条目中，把赠品给你加入进去

4、运费的计算：对每个商品条目，都要去计算一下对应的运费，而且如果有赠品的话，我们这个网站的策略，是对赠品也收取对应的运费的

这是另外一个接口

5、优惠券的减免金额：用户在订单确认页面里，可以动态的选择使用哪个优惠券，如果用户选择了一个优惠券之后，会再次动态的对整个订单的金额进行一个计算

这块，主要是在商品进入订单确认页面的时候，根据当前使用的优惠券，或者是促销活动，进行订单最终价格的一个计算

在订单确认页面，需要走一个价格计算的服务，动态的根据当前的促销信息，计算出每个商品实际要支付的价格

先判断订单里每个商品是否参加了促销，然后看下本地订单中是否满足了促销条件，最后根据促销规则计算出商品的最终价格

订单中心会将整个订单，发到商品中心的价格服务里，让它去完成订单以及其中各个商品的价格的计算。价格服务，会从促销中心中，将所有的促销活动的数据查询过来，然后进行统一的计算。

2、、促销活动和赠品处理

满减促销：满100减10、满300减50、满500减80

多买优惠：买2件9折，买3件8折，买4件7折

单品促销：促销期间商品售价直接打折，比如6折

~~套装促销：A商品50元，B商品80元，一起购买只要120元，便宜10元~~

满赠促销：满300元送自拍杆，满500送充电宝，满1000送高端耳机

赠品促销：购买某个商品可以赠送别的商品

一共买了3个商品：A商品是100元，B商品是300元，C商品是250元，650元

如果某个商品选择了单品促销

直接对这个商品的价格进行折扣，打指定的一个折扣，比如说某个商品，B商品，8折，240元

A商品是100元，B商品是240元，C商品是250元，590元

如果某个商品选择了套装促销

此时需要对订单中结成套装的商品，看一下，如果组成了一个套装的话，此时可以给一个总的优惠价格，A + B = 300元，比如A原来是100，B原来是240，套装是300 => 优惠分摊，40块钱，按照比例分摊优惠，扣减金额。100 - 100 / 440 \* 40 = 90.91，A商品；340 - 340 / 400 \* 40 = 309.09，B商品

如果某个商品选择的是多买优惠促销

会看一下，这个商品一共购买了多少件，如果达到了指定的件数，说可以打折，看一下具体按照规则，可以打几折

这个时候，每个商品的价格都降低了，有上面3种促销活动的商品价格，都降低了

满减促销，看一下选择满减促销的某个商品，这个商品的金额是否达到了一个满X元的条件，如果达到了，就对商品的价格减去Y元。比如C商品是180元，选择了满300件50的活动，C商品一共购买了2件，360元 - 50元 = 310元。

~~假设这个时候，A商品是90.91，B商品是309.09，C商品是180，加起来的总和 = 580元~~

~~满500减80，580 - 80 = 500元，省了80元钱，按照比例分摊~~

到此为止。。。。价格就算完了，总的优惠价，每个商品的优惠分摊价格，都出来了

（3）赠品的处理

然后就是处理促销

看一下，如果有某个商品选择的是买赠，此时会在订单中加入一个商品，只不过金额是0元，很多电商，多出来一个商品，只不过金额是0元

满赠，满多少元赠一个赠品，看一下此时单个商品的金额，如果达到一定的金额，可以赠送商品，在订单里再加入对应的赠品。比如说某个商品是100元，选择的是满180赠一个小耳机的活动，买了2件，200元，就给赠送一个商品。

3、优惠券计算

如果这个用户有一个有效的优惠券，在订单确认页面下面可以显示一个优惠券下拉框，你可以选择这个订单要使用一个优惠券

进来会直接去计算，在计算完促销活动的基础之上，会自动给你去应用那个券，折上折。现金券，直接抵扣；满减券，看整个订单满多少元，给减多少元。

~~还有一个，在购物车每个商品的下拉框中，单品折扣优惠券，跟其他的那些促销活动互斥，只能选择一种~~

### 4.1.3 创建订单

1. 接收到生成订单的请求之后，需要去在保存订单数据之前，需要去检查库存是否足够。这里是直接检查库存中心的可销售库存是否足够，因为库存中心、调度中心、WMS三层的库存理论上是一模一样的，所以此时就检查一个库存中心的库存即可。如果可销售库存充足，则进入下面的流程。

（2）如果库存校验ok，就执行创建订单的流程；如果库存校验不ok，那么就给前端返回一个异常的说明，告诉他订单中的某些商品库存不充足。

1. 保存订单的数据，生成订单号，订单状态：待付款
2. 执行锁定库存的操作：那么首先是订单中心要通知库存中心，商品的可销售库存减1，商品的锁定库存加1

~~如果将这一块做成同步的操作，那么会变成订单中心等待着库存中心，库存中心等待着wms中心。等到wms中心库存锁定之后，调度中心才会返回，然后订单中心的这个请求才算是结束。~~

~~库存锁定这一块，我们如果这样子给硬生生做成同步的话，可能会导致面向用户而言，他可能会觉得说等待的时间太长了。实际上用户创建订单，他是不care底层的库存锁定的逻辑的。他只要是订单创建完成就可以了。~~

~~是不是说可以做成异步的，做成异步的之后，肯定要考虑这块库存锁定以及别人在下单的时候，要去检查库存中心的库存，会不会出现一些数据不一致的情况。~~

~~但是刚开始的时候，内测，并发很少的，如果并发很小，那么出现库存数据不一致的情况，会比较少。所以我们前期就不去考虑这个问题了。但是，以后，一定会去考虑这个问题。~~

~~电商系统的库存中心，到了后面是重点中的重点，非常的重要，迭代演进这个库存中的价格。~~

~~我这边给大家一个思路，订单中心可以去调用库存中心，但是库存中心直接接收到请求之后，就给订单中心返回了；对于库存中心的库存锁定 -> 调度中心的库存锁定 -> wms中心的库存锁定，就走异步的这么一个策略了。~~

~~这边参照上面说的那套异步的思路，这块，我们直接将订单中心对于非订单核心逻辑的一些东西，全部做成订单中心跟其他中心是异步通信执行这个操作的。~~

~~订单中心会发送一条消息（包含了数据）到一个内存队列里去，由这个库存中心去消费，消费到之后，库存中心去完成后续的库存的锁定操作。~~

1. 保存一条订单操作日志，类型：生成订单

## 4.2 查询订单列表

直接查询用户的订单列表即可，支持分页查询，不支持查询条件

下单时间、订单号

订单中包含的商品列表：~~缩略图、~~sku编号、商品名称、销售属性、购买数量、商品价格、商品毛重、商品长宽高

收货人姓名

总金额、促销折扣金额、优惠券抵扣金额、运费、应付金额

支付方式

订单状态（未付款的时候，叫做待付款）

跟踪物流

查看订单详情

操作栏（取消订单、付款）

## 4.3 查询订单详情

（1）订单号、订单状态（等待付款）

（2）订单的操作日志记录：你对订单做的所有的操作，都会记录订单操作日志流水

（3）收货人信息（收货人、地址、手机号码）、配送信息（~~配送方式、~~运费）、付款信息（支付方式、总金额、促销活动抵扣、优惠券抵扣、应付金额）、发票信息（~~发票类型、~~发票抬头、纳税人识别号）

（4）商品列表：~~缩略图、~~sku编号、商品名称、销售属性、商品售价、购买数量、商品长宽高

（5）订单总金额、运费、促销活动折扣、优惠券折扣、应付总金额

## 4.4 取消订单

### 4.4.1 手动取消订单

用户提交一个订单之后，刚开始订单的状态不是待付款吗，等待用户对这个订单付款，如果用户在付款之前，是可以手动取消订单的

（1）点击取消订单的按钮，订单状态变为：已取消

（2）释放锁定库存，无论是哪种情况，进行了订单的取消之后，就需要将订单对应的之前的库存进行释放。订单中心需要去通知库存中心，让库存中心释放库存。

~~订单中心发送一条消息到内存队列中去，然后由这个库存中心去消费，消费到之后去异步的执行锁定库存的释放即可~~

（3）新增一条订单操作日志，类型：手动取消订单

### 4.4.2 自动取消订单

（1）系统后台线程，自动追踪每个订单，如果是超过24小时没支付的订单，系统直接自动取消

（2）更新订单状态，状态变为：已取消

（3）释放锁定库存，无论是哪种情况，进行了订单的取消之后，就需要将订单对应的之前的库存进行释放。订单中心需要去通知库存中心，让库存中心释放库存。

~~订单中心发送一条消息到内存队列中去，然后由这个库存中心去消费，消费到之后去异步的执行锁定库存的释放即可~~

（4）新增一条订单操作日志，类型：自动取消订单

## 4.5 支付订单

1. 完成支付本身的事情，用户需要支付订单金额到我们的账号里去。先显示出来第三方支付的页面（二维码），完成支付操作，支付中心会去记录支付交易流水。根据你这个订单之前选择的支付方式，因为我们这里简化为仅仅支持支付宝和微信两种，比如说你选择的是支付宝，然后我们就显示支付宝的二维码；如果你选择的是微信支付，那我们就显示微信的二维码

支付宝接入：其实就是调用一个接口，获取一个二维码，在前端展示出来；然后用户可以去扫描这个二维码进行支付，回调接口，我怎么知道这个用户完成支付？

微信支付接入：就是调用一个接口，获取一个二维码，在前端展示出来；然后用户扫描这个二维码进行付款，我怎么知道用户完成了支付？

我们需要让订单中心和支付中心进行交互，订单中心是不能去自己完成支付的，要将支付相关的事情交给支付中心去干，所以相当于是，订单中心，将这个订单的数据传递给支付中心，要求获取一个支付的二维码。

支付中心提供一个获取二维码的这么一个接口，根据订单中用户选择的支付方式，来调用支付宝或者是微信的接口，获取到一个二维码后返回给订单中心，然后订单中心将这个二维码返回给前端，前端需要自行跳转到支付的页面上去，显示这个二维码（简化）

然后用户扫码支付之后，支付宝还是微信，~~notify\_url，他们会走支付中心的一个支付结果回调通知接口，支付中心会拿到每一笔支付是否成功，如果支付成功之后，支付中心就需要在自己本地落地存储一行支付流水~~，我们走一个后台的线程，轮询所有的订单支付交易，去查询交易的状态，3秒轮询一次，根据返回的东西更新订单支付交易的状态，同时会通知订单中心，本次支付的这么一个状态。

也就是说订单中心得提供一个接口给支付中心，如果支付成功之后，需要走下面的流程

如果是微信支付，也是先跟微信拿到一个二维码来展示，然后如果用户支付之后，微信会回调支付中心的一个接口，来通知我们这个支付的结果，如果支付成功之后，支付中心一样是回调订单中心的一个接口，通知订单中心支付成功

支付中心，加一个接口是获取这个支付二维码；订单中心，加一个接口是获取支付结果的通知

1. 你可以自己用手机在电脑上去扫码然后去完成支付
2. 更新订单的状态：待发货
3. 更新订单中的商品相关的库存：通知库存中心，更新商品的库存

~~订单中心发送一条消息到内存队列，库存中心消费之后，异步去完成库存相关的更新操作。~~

1. 接着将订单同步到调度中心，由调度中心去调度订单中的商品的发货。

~~这块必须是得做成异步的操作，订单中心也是发送一条消息到内存队列，调度中心来消费这条消息，然后调度中心自己后台完成调度发货的逻辑~~

~~三种设计模式结合起来：中介者模式+观察者模式+备忘录模式~~

1. 将订单支付金额\*10%，累加到会员的成长值以及积分，同步调用会员中心的接口

（6）新增一条订单操作日志，类型：支付订单

## 4.6 更新订单状态

其实你写完代码之后，会发现一个事儿，就是在订单发生状态流转的时候，一定会添加一条订单操作日志。因为订单状态流转的时候，就代表着他发生了一个操作，那么肯定就要记录一条操作日志。

我们现在是将订单状态管理和操作日志是分离开来分别进行的，但是可能会有一个风险，就是说如果以后我们调用某个订单状态管理的方法，但是忘了添加订单操作日志的代码，或者是每次都要添加，比较麻烦。

我们完全可以使用装饰模式的思想，可以对现有的OrderStateManager进行装饰，装饰过后，对立面的每个方法，都加一个添加操作日志的这么一个行为，这样的话，就将订单状态的管理和添加操作日志这个事情和有机的结合到了一起

### 4.6.1 订单完成发货

WMS系统那边，确认订单完成发货之后，就会发送通知过来，通知订单中心：

1. 将订单状态修改为：待收货
2. 新增一条订单操作日志，类型：商品发货。

### 4.6.2 退货工单审核不通过

如果退货工单审核不太通过的话，会通知订单中心：

1. 将订单状态变为：交易关闭（退货审核不通过）
2. 新增一条订单操作日志，类型：退货审核不通过
3. 更新订单退货记录的状态：审核不通过。

### 4.6.3 退货工单审核通过

如果退货工单审核通过了，那么会通知订单中心：

1. 将退货记录的状态变为：审核通过
2. 新增一条订单操作日志：退货审核通过
3. 将订单状态更新为：交易中（待寄送退货商品）

### 4.6.4 确认收到退货商品

客服中心的客服收到退货商品之后，会做一个初步的检查，如果确认ok，会点击确认收到退货商品，此时会通知订单中心：

1. 新增一条订单操作日志：确认收到退货
2. 将订单的状态更新为：售后中（退货待入库）

### 4.6.5 退货入库单审核通过

wms中心完成退货入库，同时退货入库单审核通过之后：

1. 需要将订单的状态修改为：售后中（退货已入库）
2. 新增一条订单操作日志，类型：退货已入库。

### 4.6.6 财务中心完成退货打款

财务中心在接收到wms中心的退货入库的通知，同时完成退货打款之后，会来通知订单中心：

1. 新增一条订单操作日志：完成退款
2. 将订单的状态更新为：交易关闭（完成退款）

## 4.7 查看物流进度

订单中心去调用物流中心提供的接口，然后返回和显示物流的进度即可

## 4.8 确认收货

### 4.8.1 手动确认收货

用户收到货之后，拆包满意，那么可以在订单列表中自己手动点击确认收货，此时：

1. 就会将订单的状态改变为：已完成。
2. 更新一下确认收货的时间
3. 新增一条订单操作日志，类型：手动确认收货。

### 4.8.2 自动确认收货

系统对处于待收货状态的订单，会自动定期检查每个待收货订单是否签收，如果第三方物流商的反馈状态是已经签收了，然后超过7天用户没点击确认收货，系统自动确认收货。

1. 订单状态变为：已完成。
2. 更新一下确认收货的时间
3. 新增一条订单操作日志，类型：自动确认收货。

## 4.9 申请退货

### 4.9.1 执行申请退货

在订单列表页面，如果是处于已完成状态的订单，可以点击申请退货的按钮。然后会进入一个界面：

1. 要求你选择退货原因，输入退货的备注
2. 后台新增一条订单退货记录，状态为：待审核
3. 订单状态变为：售后中（退货申请待审核）

（4）新增一条订单操作日志，类型：申请退货

（5）通知客服中心创建一个售后工单

### 4.9.2 查看退货进度

用户肯定时不时过来看看的，此时如果状态为售后中，可以点击状态进入一个页面，在页面里，会显示：

1. 当时用户选择的那个退货原因，以及填写的退货备注
2. 同时给出一个审批结果
3. 如果是审批通过的状态，此时会显示出来需要用户将商品寄送到公司的某个地址（地址、收件人、联系方式）

### 4.9.3 填写退货物流单号

用户在寄出之后，需要在这个页面里填写对应的物流单号

1. 在退货记录中保存快递单号
2. 将快递单号同步到客服中心的退货工单中
3. 将订单的状态更新为：退货商品待收货
4. 新增一条订单操作日志：寄送退货商品

# 5、测试用例

# 6、日志设计