

# 廈門大學



## 面向对象分析与设计课程

### 顾客订单模块

### 对象模型、API、状态和数据库设计

组 别 3-4 组

组 员 黄子安 崔方博 范周喆 侯好欣 徐森彬

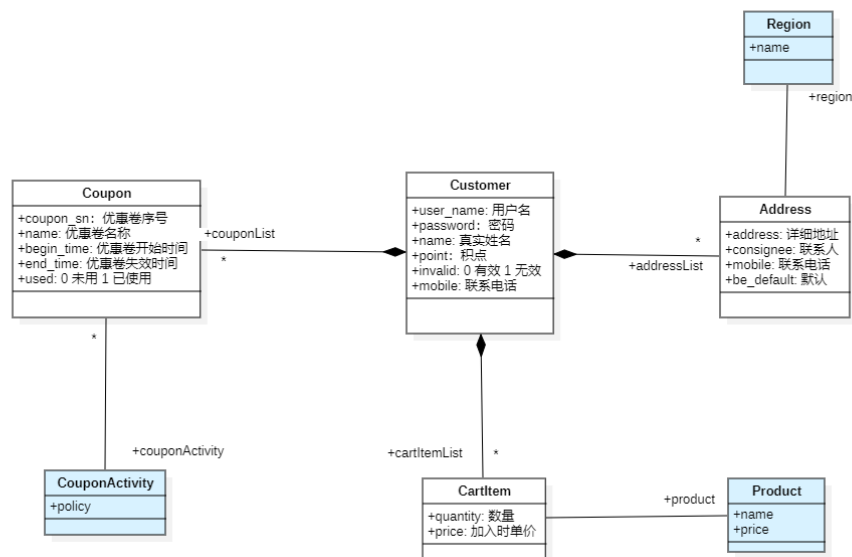
2023 年 11 月 11 日

## 目录

一、顾客模块 .....	3
1. 对象模型 .....	3
2. 状态机 .....	4
3. API .....	6
4. 数据库设计 .....	7
二、订单模块 .....	8
1. 对象模型 .....	8
2. 状态机 .....	9
3. API .....	12
4. 数据库设计 .....	15

# 一、 顾客模块

## 1. 对象模型



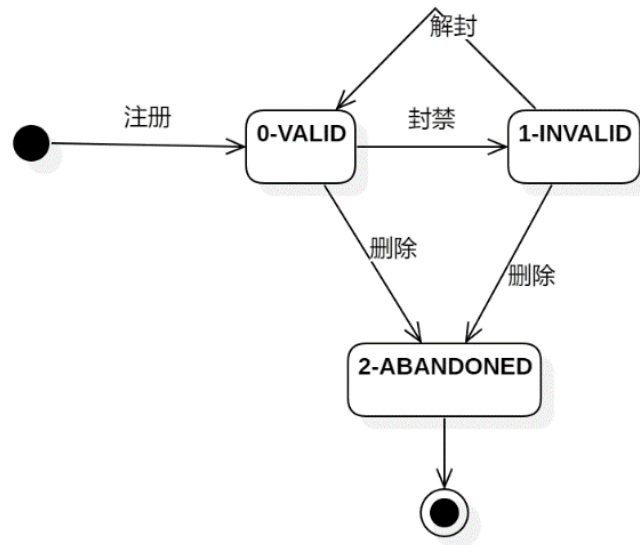
说明：

- 1、API 以**顾客为中心**，在调用的时候通过顾客类获得其余类的信息，因此对象模型中箭头的方向指向顾客
- 2、顾客和地址是一对多的关系，因为一个顾客可以同时管理多个收货地址
- 3、优惠券指的是被**顾客领取的优惠券**，如果指活动发放的优惠券，会导致出现大量的存储浪费，所以将其设计成只有优惠券被领取了才会成为实体类，同理优惠券的状态机是从**待使用**开始而不是从待领取开始
- 4、顾客和优惠券、地址、购物车明细都是**组合关系**，它们符合组合关系的要求，是构成了整体和局部的关系，局部对象（优惠券、购物车明细、地址）必须属于且只属于一个母体对象（顾客），母体对象负责创建和销毁局部对象

## 2. 状态机

说明：状态机上状态转移的 API 链接合并在了下一个部分中

### 2.1 顾客状态机

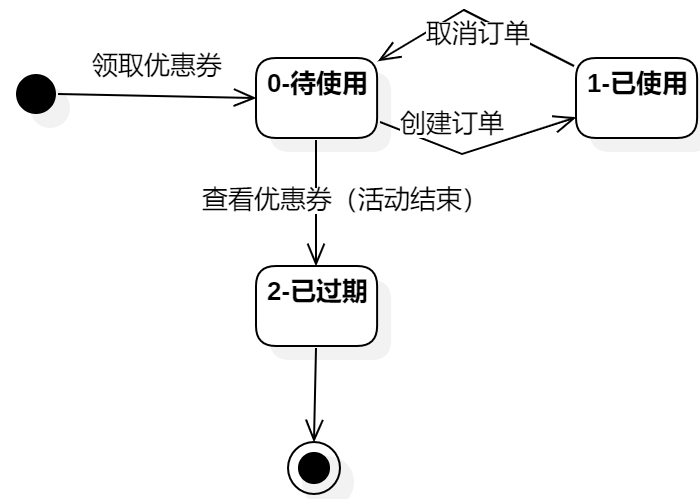


顾客有三种状态，有效（VALID）、已封禁（INVALID）、已删除（ABANDONED）

顾客状态类型的设定主要是考虑到这三个状态将会影响顾客的登录操作：

- **顾客的状态决定了顾客是否可以进行登录操作**，处于有效状态（VALID）的顾客可以正常登录并且进行登录后的操作。
- 处于已封禁的顾客（INVALID）**无法进行登陆操作**，只有管理员将顾客解封后才能回到有效状态进行登录。
- 处于已删除状态的顾客将被永久删除，无法登录。

## 2.2 优惠券状态机



优惠券有待使用、已使用、已过期三种状态。

- 优惠券的起始状态是待使用状态

原因：如果从优惠券创建（即活动的创建）开始，系统将要保存巨量的优惠券数据。而事实上，真正被用户领取的优惠券数据远小于此，将初始状态设成用户已经领取的、待使用的状态，可以减小系统负载。

- 取消订单有两个 API，因为顾客和商户都能进行取消订单的操作。

- 不论优惠券是否被使用，其最终状态均是已过期。

因为已经使用的优惠券有可能因为取消订单而回到待使用状态，于是，已使用不能作为最终状态。

- 优惠券是否过期的相关状态转移 API 是查看优惠券，因为在平时不会去关心优惠券是否还有效，只有在使用或者想使用的时候会关心是否过期，因此在查看优惠券 API 中检验是否过期即可，可以减小系统的负担

### 3. API

#### 3.1 顾客状态机相关 API

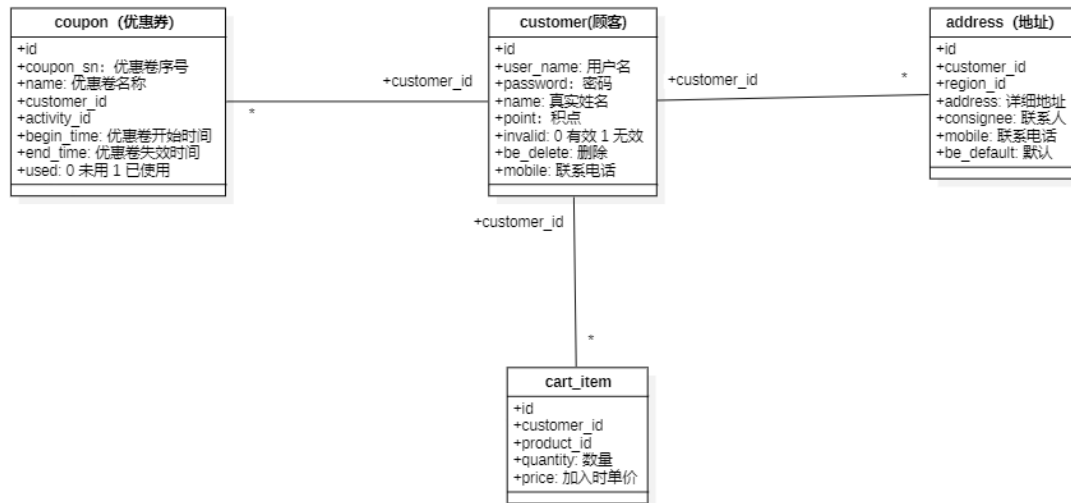
事件	API
注册	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/customer/registerUser">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/customer/registerUser</a>
封禁	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/customer/banUser">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/customer/banUser</a>
解禁	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/customer/releaseUser">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/customer/releaseUser</a>
删除 *	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/customer/deleteUser">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/customer/deleteUser</a>

#### 3.2 优惠券状态机相关 API

事件	API
领取优惠券	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/customer/post couponactivities id coupons">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/customer/post couponactivities id coupons</a>
创建订单*	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/customerPostNewNormalOrder">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/customerPostNewNormalOrder</a>
取消订单	<div>✧ <a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/delete shops shopId orders id">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/delete shops shopId orders id</a></div> <div>✧ <a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/delete orders id">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/delete orders id</a></div>
查看优惠券列表	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/customer/showCoupons">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/customer/showCoupons</a>

注：创建订单 API 有进行修改，将在后续订单模块进行说明。

## 4. 数据库设计

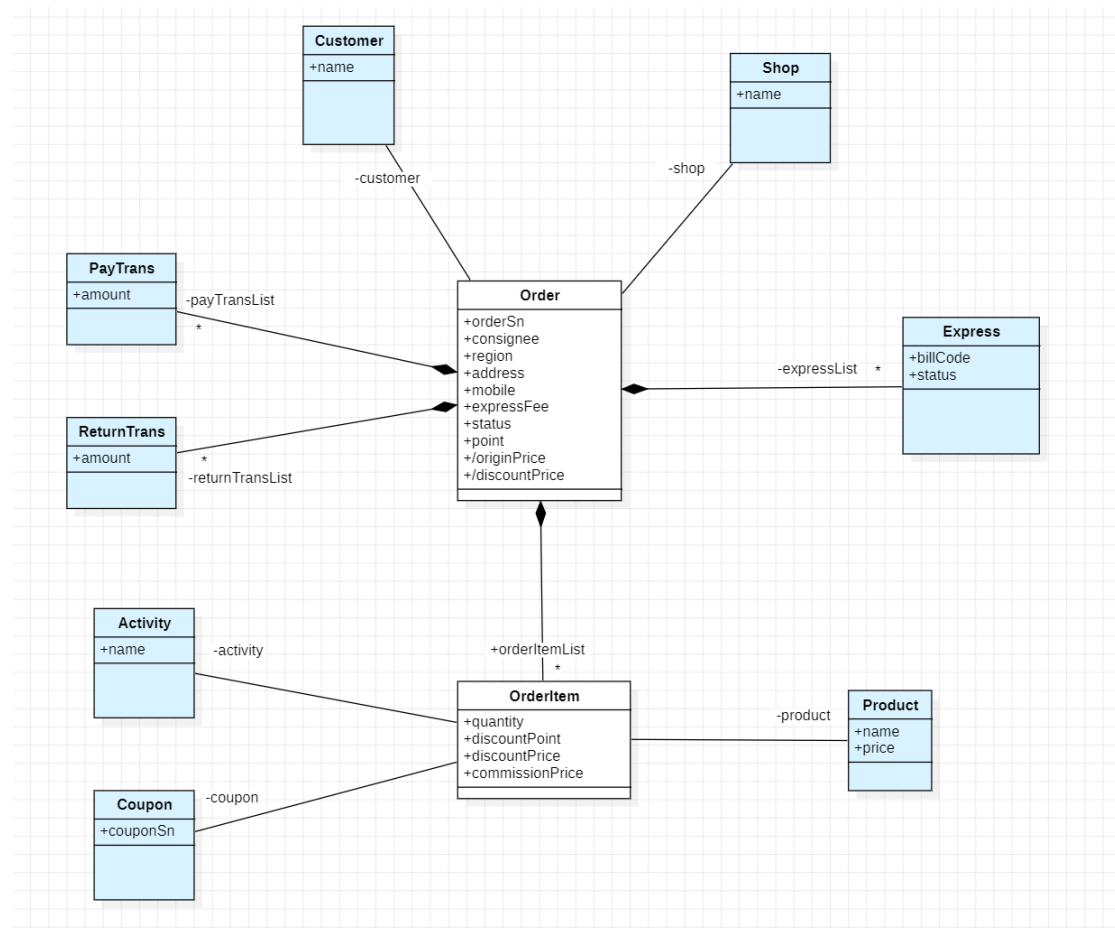


说明:

数据库模型和对象模型中顾客和其余部分方向出现了相反, 没有从 customer 指向边上的三个类, 因为如果 customer 对于其他对象是一对多的关系, 在数据库中需要**第三张表**进行存储连接, 为了节约存储的开销, 将它们转换为**多对一**的关系, 将 **customer\_id** 作为字段放在边上的三个数据表里, 从而不用额外增加一张表, 数据库模型和对象模型中这里出现的反向问题需要在代码中进行处理

## 二、 订单模块

### 1. 对象模型

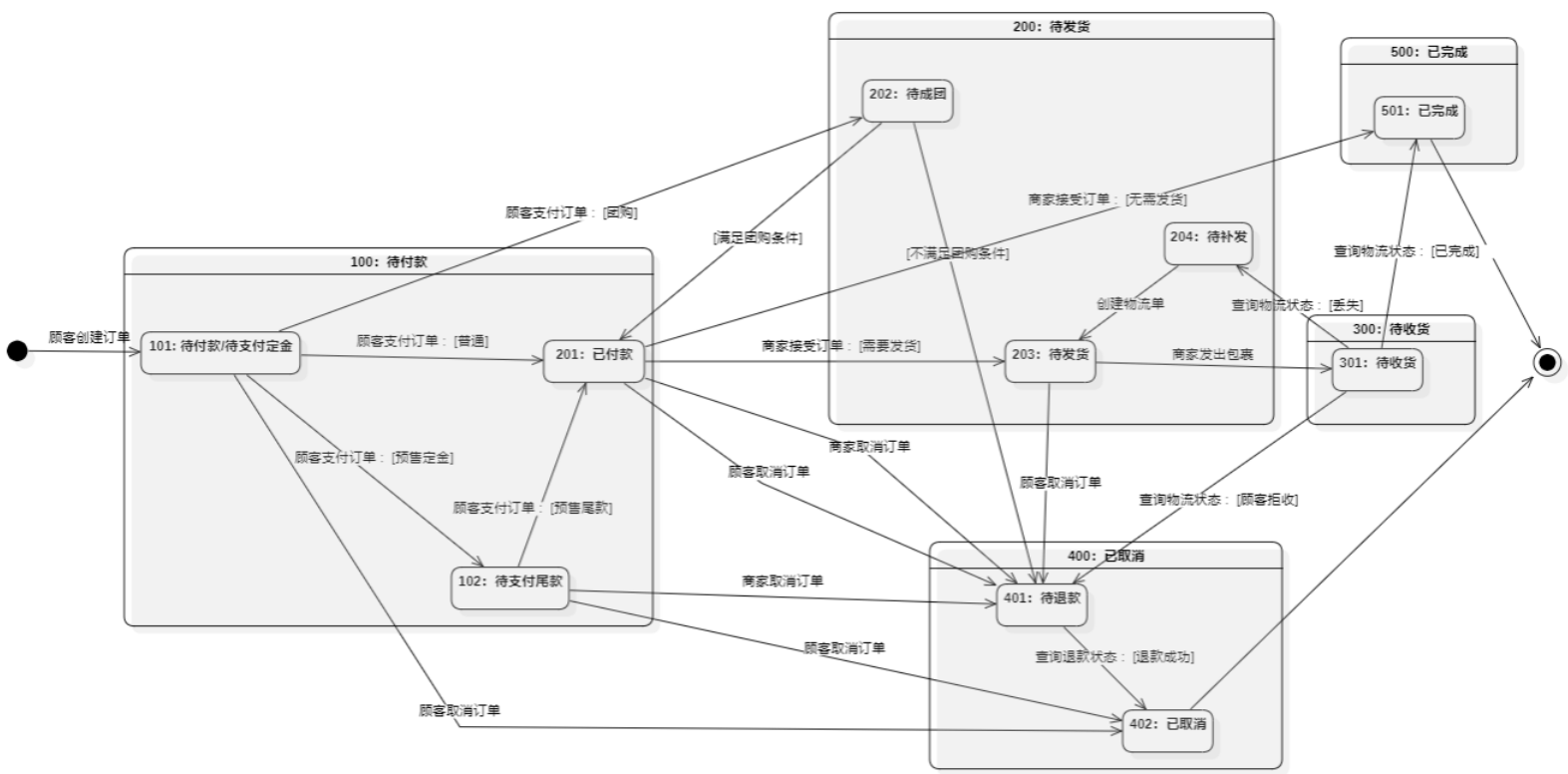


注：图中考虑到从订单对象模型角度来看，一个订单只对应一个顾客，一个商家，故而把领域模型中顾客、商家与订单一对多的关系，简化为了一对一的关系。类似的简化，还有 OrderItem 与 Activity 的多对一关系，OrderItem 与 Coupon 的多对一关系。



2. 状态机

2.1 订单状态机图

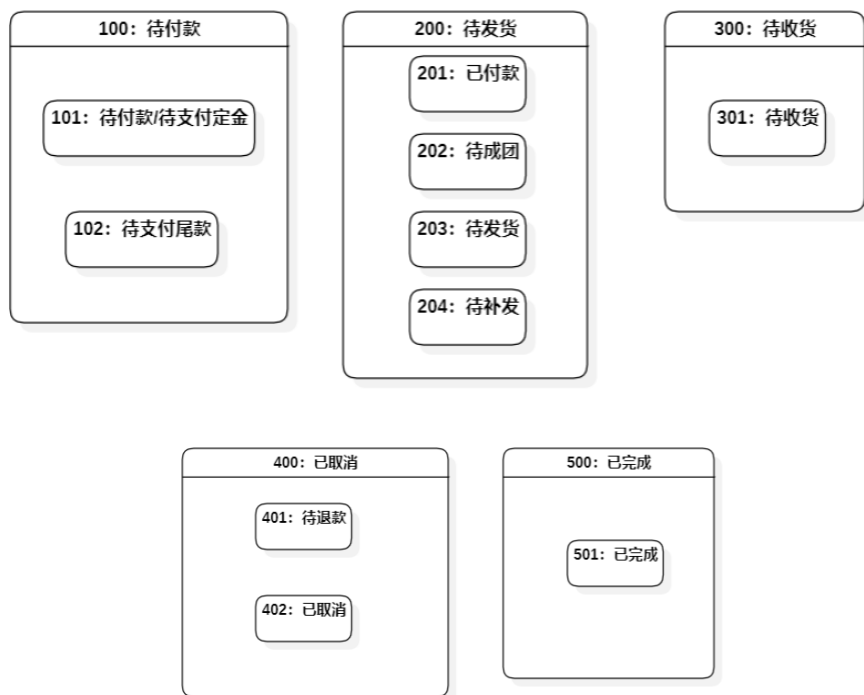


2.2 状态说明

订单的状态我们区分为**顾客视角**下的状态和**商户视角**下的状态，其中顾客视角下的状态属于大的订单状态（即图中大的框），是虚化的概念，方便顾客的理解和查看，这五个状态包括：

100 待付款, 200 待发货, 300 待收货, 400 已取消, 500 已完成

商户视角的订单信息可以用大状态下包含的若干小状态表示，是实际的订单状态，其中虽然 300 待收货和 500 已完成只包括一个状态，但为了方便区分顾客和商户的视角差异，仍保留了各自的小状态，具体包含信息如下：



## 2.3 订单状态描述

订单状态的变化可以描述如下：

- 首先订单由顾客创建订单产生，此时为 101 待付款/待支付定金状态；
- 对于不同种类的待付款状态的订单，可以分为：

### ■ 团购订单：

首先顾客要全款支付团购订单，订单转为 202 待成团状态；

定时器每天会自动调用查询团购结果的接口来更新订单的状态，如果团购时间结束，且满足团购条件（团购达到一定的人数），则订单会转为 201 已付款状态，并自动产生退款退还给顾客；如果不满足团购条件（团购没有达到一定的人数），则会取消订单，进入 401 待退款状态，将顾客支付的全部费用退还给顾客，团购失败。

### ■ 预售订单：

顾客首先要支付定金，订单进入 102 待支付尾款状态；

如果在 102 待支付尾款状态下，商家取消订单，则需要将顾客所付定金退还给顾客，进入 401 待退款状态；如果是顾客取消订单，则定金不会退还，订单会进入 402 已取消状态；

■ 普通订单：

顾客可以直接支付订单，进入 201 已付款状态；

- 在已付款状态下，需要商家先进行接受订单，如果商品是不需要发货的电子商品，则可以直接完成，进入 501 已完成状态；如果是需要发货的订单，商家接受订单时系统会自动同步创建物流单，如果是需要拆单发货的商品，系统会自动将一个订单拆分成多个物流单并记录到数据库中，订单进入 203 待发货状态。
- 待发货状态下的订单，商户发货后订单状态更新为 301 待收货状态；
- 在待收货下的订单，是处于物流运输状态下的订单，在此过程中可以通过物流模块的接口查询物流状态来更新订单状态，查询物流状态又会出现以下几种情况：
  - 物流状态已收货：说明订单已完成，进入 501 已完成状态
  - 物流状态丢失：说明在途包裹丢失，需要商户进行补发，订单状态转到 204 待补发状态，调用创建物流单进行重新进入 203 待发货状态。
  - 物流状态顾客拒接签收：需要进行退款，进入 401 待退款状态
- 在 100 待付款和 200 待发货这两个大状态下的订单用户可以直接取消使订单进入 400 已取消大状态，其他状态则不可以直接取消；

3. API

3.1 状态机相关 API


事件	API
顾客创建订单*	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/customerPostNewNormalOrder">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/customerPostNewNormalOrder</a>
顾客支付订单	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/payOrder">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/payOrder</a>
顾客取消订单	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/delete_orders_id">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/delete_orders_id</a>
商家取消订单	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/delete_shops_shopId_orders_id">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/delete_shops_shopId_orders_id</a>
商家接受订单*	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/postFreights">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/postFreights</a>
商家发出包裹*	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/putFreights">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/order/putFreights</a>

3.2 API 修改说明

- 修改顾客创建订单 API 的参数：  
  
添加了优惠券 ID(couponId)作为参数传入

POST

/orders 买家申请建立订单（普通，团购，预售）\*

接口名称	接口地址
买家申请建立订单	/orders

● 修改商家接受订单 API 的描述与命名：

商家在顾客下单付款后决定是否接受此订单，如果是无需物流发货的电子商品订单状态可以直接从已付款修改到已完成；

如果是需要物流的商品，则将订单状态从已付款改为待发货，创建物流单时系统会自动根据所买的商品的信息决定是否进行拆单处理，并将订单对应的物流单信息记录到数据库中。

PUT

/shops/{shopId}/orders/{id}/confirm 店家接受订单\*

接口名称	接口地址
店家接受订单	/shops/{shopId}/orders/{id}/confirm




● 新增店家发出包裹 API：

对于一个待发货的订单，根据传入的物流单 id 进行单个包裹的发货，在完成单个包裹的发货后会根据订单 id 检查所有的物流单是否已经发完，如果已经发完(所有物流单的状态是已发货)，则更改订单的状态为待收货。

因为从负责发货的人的视角来看是**以包裹为单位而不是以订单为单位**，如果一个订单被拆分成了多个包裹，这些包裹不一定是被同时完成拣货的，为了便于仓库拣货操作，将其设置为以包裹为单位，拣货人员拣完一个包裹就向系统提交这个包裹信息就行，不用关心整个订单。

PUT

/shops/{shopId}/orders/{id}/deliver  
/{billcode} 店家发出包裹\*

接口名称	接口地址
店家发出包裹	/shops/{shopId}/orders/{id}/deliver/{billcode}

- 新增更新团购订单状态的 API

定时器**每天**调用此接口来自动更新团购订单的状态，从而可以减小系统的负担。

如果团购时间结束，且参与人数达到团购要求，那么就会修改对应订单的状态从待成团到已付款，并退还部分费用；如果参与人数没有达到团购要求，则将订单状态转移到待退款状态，团购订单取消。

PUT

/groupons 更新所有团购订单状态\*

接口名称	接口地址
更新所有团购订单状态	<a href="#">/groupons</a>

- 修改 models 里 OrderInfo 模块的属性：

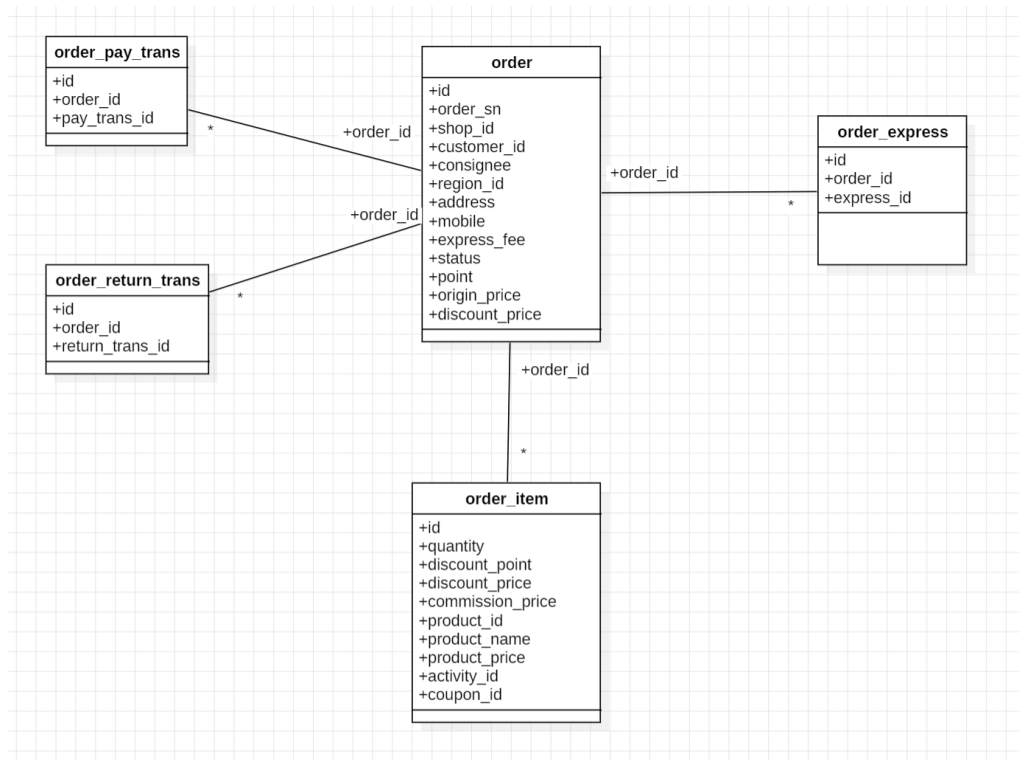
将参数 pack 修改为 expressList，因为一个订单对应多个物流单

```
OrderInfo {
  description: 订单的完整信息*

  id > [...]
  orderSn > [...]
  customer IDName > [...]
  shop IDName > [...]
  pid > [...]
  status > [...]
  gatCreate > [...]
  gatModified > [...]
  originPrice > [...]
  discountPrice > [...]
  expressFee > [...]
  message > [...]
  consignee Consignee > [...]
  expressList > [...]
  orderItems > [...]
}
```

名称	链接
OrderInfo	<a href="https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/OrderInfo">https://app.swaggerhub.com/apis/hyxHome/oomall/1.3.1#/OrderInfo</a>

## 4. 数据库设计



注：

- ① 为什么要有 order\_pay\_trans, order\_return\_trans, order\_express 三个数据库外表？

以 order 与 payTrans 的关系为例，因为订单与支付单的关系为一对多的关系，按照数据库表的设计原理，本该将一对多的关系，转化为多对一的关系，即把 order\_id 的属性存储在 payTrans 的数据库表中，但这里存在一个问题，payTrans 的数据库表并不是属于订单模块的数据库表，考虑到模块间的独立性，最好不在 payTrans 添加一个属性 order\_id。更好的解决方案就是如上图所示，创建一个外表 order\_pay\_trans，来存储 order 与 payTrans 之间的联系。

② order\_item 表中为什么有 product\_name 和 product\_price?

因为一个产品的名称和价格可能随时间发生改变，而对于订单来说，产品的名称和价格固定是创建订单那一时刻的名称和价格，不能随时间改变，所以要把订单产生时刻的产品名称和价格记录下来。