
回到首页

芋道源码 —— 知识星球

我是一段不羁的公告！
记得给芳芳这 3 个项目加油，添加一个 STAR 噢。
https://github.com/YunaiV/SpringBoot-Labs
https://github.com/YunaiV/onemall
https://github.com/YunaiV/ruoyi-vue-pro
2018-03-20
数据库实体设计

数据库实体设计 —— 交易（2.5）之退款维权

> 芳芳目前正在做一个开源的电商项目，胖友可以 star 下。
> https://gitee.com/zhijiantianya/onemall

1. 概述

本文主要分享交易模块的退款维权的数据库实体设计。

基于如下信息，逆向猜测数据库实体：

> 有赞云提供的商家退款API
> 有赞云提供的买家退款API
> 有赞微商城的退款维权
> 有赞云开发指南的交易场景【订单逆向调用】
>
> 【护脸旁白】
> 笔者非电商行业出身 && 非有赞工程师，所以有错误或不合理的地方，烦请斧正和探讨。
> 有赞是个各方面都很 NICE 的公司，推荐 。

2. 背景了解

2.1 买家如何申请退款/退款退货？

参见 《买家如何申请退款/退款退货？》 文档。

2.2 卖家如何处理维权订单

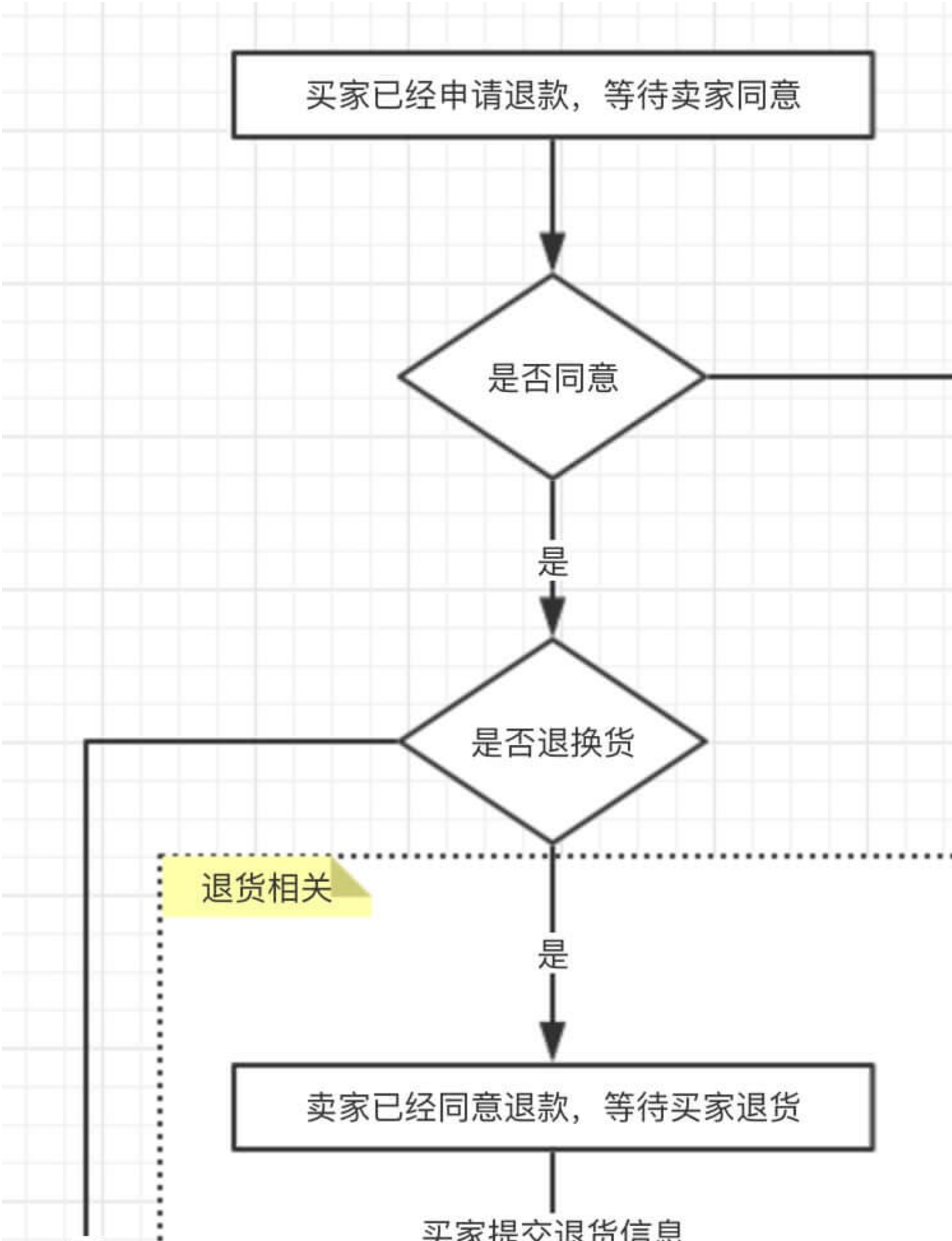
参见 《卖家如何处理维权订单？》 文档。

2.3 订单逆向调用

参见 《有赞云开发指南的交易场景【订单逆向调用】》 文档。

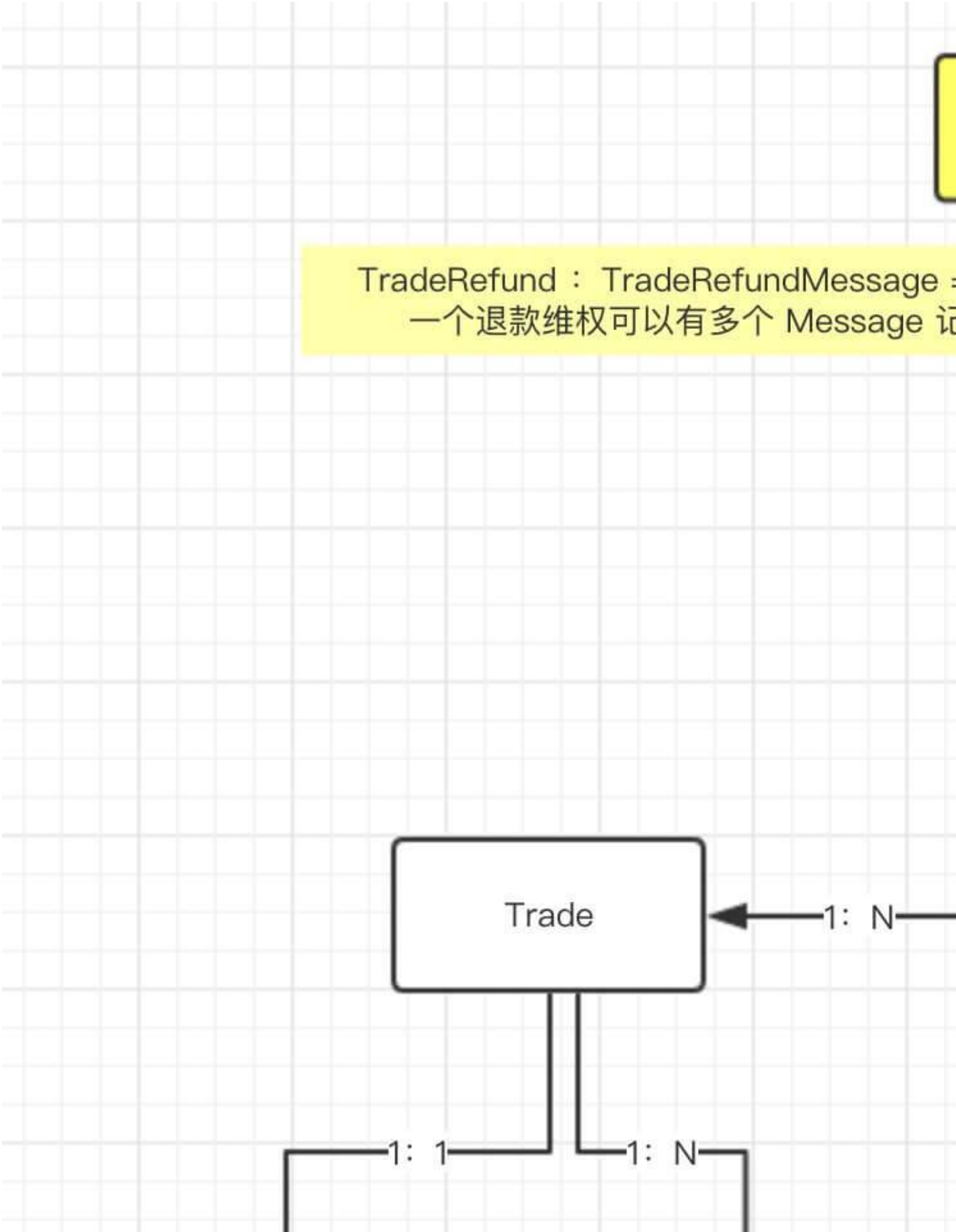
打开如上文档后，搜索 “订单逆向调用“ 关键字。

2.4 退款维权状态图



3. 数据库实体

整体实体类关系如下图：



全部实体在 Github 退款目录 下可见。

3.1 Trade

我们需要在 Trade 上，新增退款维权相关的字段。

```
/**
 * 交易维权状态。
 *
 * 0 无维权，
 *
 * 1 顾客发起维权，
 * 2 顾客拒绝商家的处理结果，
 * 3 顾客接受商家的处理结果，
 * 9 商家正在处理；
 *
 * 101 维权处理中，
 * 110 维权结束；
 *
 * 备注：1到10的状态码是微信维权状态码，100以上的状态码是有赞维权状态码
 */
private Integer feedback;
/**
 * 处于交易维权状态的订单数
 */
private Integer feedbackNum;
/**
 * 退款状态。
 *
 * 0 - NO_REFUND（无退款）
 * 1 - PARTIAL_REFUNDING（部分退款中）
 * 2 - PARTIAL_REFUNDED（已部分退款）
 * 3 - PARTIAL_REFUND_FAILED（部分退款失败）
 * 11 - FULL_REFUNDING（全额退款中）
 * 12 - FULL_REFUNDED（已全额退款）
 * 13 - FULL_REFUND_FAILED（全额退款失败）
 */
private Integer refundStatus;
```

`feedback` ，交易维权状态。

- 区间 `[1, 10]` ，微信退款维权状态码。 // TODO 6010，猜测是微信的交易记录的功能的投诉功能。目前暂时未测试出来。
- 区间 `[100, +∞)` ，有赞退款维权状态码。
  - `101` ，退款维权处理中。当交易中存在任意交易明细（订单）处于退款维权处理中时，交易则处于该状态。
  - `110` ，退款维权已结束。当交易中交易明细（订单）的退款维权全部处理完成时，交易则处于该状态。

`feedbackNum` ，处于退款维权状态的订单数。

- `feedback = 0 || 110` 时，`feedbackNum = 0` 。
- `feedback = 101` 时，`feedbackNum > 0` 。

`refundStatus` ，退款状态。分成了无退款、部分退款、全部退款的三种金额情况。

- 为什么存在退款中的中间状态？金额的退款不一定是实时到账，参见 《订单退款到账说明》 文档。
- 注意，系统会存在退款服务（不要和退款维权混淆在一起），这是一个通用的服务，对接不同的支付渠道（例如，支付宝、微信、银联等等）的退款功能，并且存储退款单数据与支付渠道对应。通过这样的系统拆分解耦，不同的业务功能有退款需求时，可以对

接该服务。例如，现在交易里提供退款维权功能，未来可以添加其他需要退款功能的功能。后面我们也会分享这块的设计。

refundedFee ，交易完成后退款的金额。单位：分。

3.2 TradeOrder

我们需要在 TradeOrder 上，新增退款维权相关的字段。

```
/**
 * 商品退款状态
 *
 * 0 - 无退款
 * 1 - 部分退款
 * 11 - 全部退款
 */
private Integer refundStatus;
/**
 * 退款id
 */
private String refundId;
```

refundStatus ，商品退款状态。不同于 Trade.refundStatus ，存在退款中的情况。所有退款中的情况，反馈在 Trade 记录上。为什么这么做呢？笔者猜测，所有支付单、退款单都仅仅跟交易进行关联。

refundId ，退款维权单编号，指向 TradeRefund.id 。目前有赞一个交易明细（订单）至多可以发起一次退款维权。

3.3 TradeRefund

TradeRefund ，交易退款维权。

买家可以对交易下的交易明细（订单）发起退款维权，一次只能针对一个。并且，一个交易明细只允许发起一次退款维权。
无论退款维权最终是成功还是失败，下一次都不能再次发起。

字段较多，我们进行简单的切块。

3.3.1 基础字段

```
/**
 * 编号
 *
 * 唯一
 */
private String id;
/**
 * 店铺编号 {@link cn.iocoder.doraemon.shopgroup.shop.entity.Shop#id}
 */
private Integer shopId;
/**
 * 交易编号 {@link cn.iocoder.doraemon.tradegroup.trade.entity.Trade#id}
 */
private String tid;
```

```
/**
 * 交易明细编号 {@link cn.iocoder.doraemon.tradegroup.trade.entity.TradeOrder#id}
 */
private String oid;
/**
 * 商品编号 {@link cn.iocoder.doraemon.tradegroup.trade.entity.TradeOrder#itemId}
 */
private Integer itemId;
/**
 * 退款申请时间
 */
private Date createTime;
/**
 * 退款申请修改时间
 */
private Date updateTime;
```

id ，退款维权编号，唯一。例如："20180206000803000010077" 。笔者猜测格式为：首字母 + 年月日时分秒 + 自增序列（六位，每秒归零） + 0077（可能是有什么业务含义）。

20180206000803000010077

20180205184403000010077

20180205170805000010077

20180205170554000020077

20180205170221000010077

20180205165847000010077

20180205165604000020077

20180205164505000030077

20180205164230000010077

```
 * 是否退货
 *
 * false - 仅退款
 * true - 退货退款
 */
private Boolean returnItem;
/**
 * 是否签收货物
 *
 * false - 否
 * true - 是
 */
private Boolean signed;
/**
 * 申请退款的金额，单位：分。
 */
private Integer refundFee;
/**
 * 申请退款人的手机号码
 */
private String mobile;
/**
 * 仅退款-未收到货申请原因
 *      11(质量问题)
 *      12(拍错/多拍/不喜欢)
 *      3(商品描述不符)
 *      14(假货), 15(商家发错货)
 *      16(商品破损/少件)
 *      17(其他)
 * 仅退款-已收到货申
 *      51(多买/买错/不想要)
 *      52(快递无记录)
 *      53(少货/空包裹)
 *      54(未按约定时间发货)
 *      55(快递一直未送达)
 *      56(其他)
 * 退货退款-申请原因
 *      101(商品破损/少件)
 *      102(商家发错货)
 *      103(商品描述不符)
 *      104(拍错/多拍/不喜欢)
 *      105(质量问题)
 *      107(其他)
 */
private Integer reasonType;
/**
 * 买家退款申请说明
 */
private String desc;
/**
 * 图片举证数组，以逗号分隔
 */
private String images;
```

returnItem ，是否退货。交易明细（订单）处于不同状态，可以选择的选项不同：

- 卖家未发货，可以选择 true ，不能选择退货(false)。
- 卖家已发货，可以选择 true 和 false 。

signed ，是否签收货物。

refundFee ，申请退款的金额，单位：分。交易明细（订单）处于不同状态，可以选择的金额不同：

- 卖家未发货，申请退款的金额只能全部。
- 卖家已发货，申请退款的金额可以 (0, 全部] 。
- 注意，无论交易本身处于什么状态，若退款成功的金额达到支付交易金额，交易将直接变成关闭状态，并标记关闭类型(Trade.closeType)为全额退款关闭。
- 交易明细（订单）可退款金额的上限计算，// TODO 6011 统一写。

mobile ，申请退款人的手机号码。

reasonType ，退款原因。根据 returnItem + signed 的组合，可选择的原因不同。并且，我们可以看到，即使原因的名字相同，对应的原因枚举编号不同。

desc ，退款备注信息。

images ，图片举证数组，以逗号分隔。

3.3.3 状态字段

```
/**
 * 退款状态
 *
 * 1 - WAIT_SELLER_AGREE - 买家已经申请退款，等待卖家同意
 * 2 - WAIT_BUYER_RETURN_GOODS - 卖家已经同意退款，等待买家退货
 * 3 - WAIT_SELLER_CONFIRM_GOODS - 买家已经退货，等待卖家确认收货
 * 4 - SELLER_REFUSE_BUYER - 卖家拒绝退款
 * 5 - SELLER_REFUSE_CONFIRM_GOODS - 卖家不同意，未收到货
 * 10 - CLOSED - 退款关闭
 * 20 - SUCCESS - 退款成功
 */
private Integer status;
/**
 * 客满介入状态
 *
 * 1 - 客满未介入
 * 2 - 客满介入中
 */
private Integer csStatus;
/**
 * 乐观锁版本号
 *
 * 解决并发修改，防止数据不一致。
 */
private Long version;
```

status ，退款状态。状态的变迁参见 「 2.4 退款维权状态图 」 。

csStatus ，客服介入状态。当买家和卖家对于维权退款无法达成一致时，可以申请有赞客服介入。

version ，乐观锁版本号，使用新增或修改记录时的时间戳。为什么会有该奇怪的字段呢？在对退款维权记录进行操作时，在执行操作前，我们需要对记录进行各种判断以判定是否满足操作的前置条件，若满足则进行更新。等.等等..等等等…下，当我们判断满足条件时，可能已经被另外一个并发的请求已经进行修改，也就是说，此时实际已经不满足条件。那么咋办呢？我们可以通过 SQL UPDATE trade_refund SET xxx = yyy , version = unix_timestamp(now()) WHERE id = ${id} AND version = ${version} 。即，判断是否满足操作前，我们查询退款维权记录的 version ，每次更新时，修改为当前时间戳，且判断数据库中的记录的 version 等于查询出来的 version 。另外，需要注意，更新时，我们需要判断更新的记录条数是否大于零。

3.4 TradeRefundDelivery

界面如下：

卖家设置退货地址

买家填写退货信息


10:16
1.26K/s
填写退货物流
物流方式
请选择物流公司
物流单号
请输入物流单号
手机号码
填写手机号便于卖家联
备注信息
最多可填写200个字
图片举证
可上传5张图片

字段如下：

TradeRefundDelivery ，交易退款维权的退货物流信息。

当买家选择退款并退货时，并且卖家同意退货后，买家需要提填写退货快递的物流信息。

```
/**
 * 退款编号 {@link TradeRefund#id}
 */
private String id;
/**
 * 店铺编号 {@link cn.iocoder.doraemon.shopgroup.shop.entity.Shop#id}
 */
private Integer shopId;
/**
 * 收件人（卖家）名
 */
private String receiverName;
/**
 * 收件人（卖家）手机号
 */
private String receiverMobile;
/**
 * 收件人（卖家）座机
 */
private String receiverTelephone;
/**
 * 收件人（卖家）收获地区编号
 */
private Integer receiverRegionId;
/**
 * 收件人（卖家）收获地址
 */
private String receiverAddress;
/**
 * 发件人（买家）手机号
 */
private String senderMobile;
/**
 * 发货备注
 */
private String desc;
/**
 * 图片举证数组，以逗号分隔
 */
private String images;
/**
 * 物流公司编号 {@link cn.iocoder.doraemon.tradegroup.delivery.entity.Express#id}
 */
private Integer expressId;
/**
 * 物流运单编号
 */
private String nu;
```

id ，退款编号，指向对应的 TradeRefund ，1 : 1 。

- 卖家在收到买家填写的退货物流信息可以拒绝，此时买家买家可以重填退货物流信息。
  如果想要记录每一次买家填写的退货物流信息，可以有如下几种方式：
  1. 拼接买家填写的物流信息成字符串，记录到 TradeRefundMessage 中的信息（

content ）字段。
2. 增加 TradeRefundDelivery 对应的 Log 记录表。
3. 【建议】相对来说，第二种比第一种更加结构化。

shopId ，店铺编号。

收件人（卖家）相关字段

- 名字 ： receiverName
- 联系方式 ： receiverMobile / receiverTelephone
- 地址 ： receiverRegionId / receiverAddress

发件人（买家）相关字段

- 联系方式 ： senderMobile
- 说明 ： desc / images
- 物流 ： expressId / nu 。通过这两个字段，可以查询到物流进度，在 《数据库实体设计 —— 交易（2.2）之物流信息（快递发货）》「 3.2 ExpressOrder 」 有分享。

3.5 TradeRefundMessage

界面：

字段如下：

TradeRefundMessage ， 交易退款维权的协商记录。

```
/**
 * 记录编号，唯一。
 */
private Integer id;
/**
 * 退款编号 {@link TradeRefund#id}
 */
private String refundId;
/**
 * 店铺编号 {@link cn.iocoder.doraemon.shopgroup.shop.entity.Shop#id}
 */
private Integer shopId;
/**
 * 发起角色
 *
 * 1 - 系统
 * 2 - 买家
 * 3 - 商家
 * 4 - 客服
 */
private Integer ownerRole;
/**
 * 具体操作
 *
 * 例如：
 *   250 - 同意退款给买家，本次维权结束
 *   206 - 已退货,等待商家确认收货
 *   205 - 已同意退款申请,等待买家退货
 *   201 - 发起了退款申请,等待商家处理
 */
private Integer op;
/**
 * 操作前状态
 */
private Integer beforeStatus;
/**
 * 操作后状态
 */
private String afterStatus;
/**
 * 明细数组字符串
 *
 * JSON 格式，数组，每个元素为 {@link Detail}
 */
private String details;
/**
 * 图片地址数组字符串
 */
private String picURLs;
/**
 * 创建时间
 */
private Date createTime;
```

id ， 记录编号，唯一。

refundId

，退款编号，指向 TradeRefund.id 。
shopId ，店铺编号。
ownerRole ，发起角色。其中，”系统“ 这个角色比较特殊，例如 《维权订单未处理会自动退款？》 。
op ，具体操作枚举序号。
beforeStatus ，操作前的退款维权状态。
afterStatus ，操作后的退款维权状态。

details ，明细数组字符串。JSON 格式化，数组，每个元素为 Detail 。代码如下：

```
public static class Detail {

    /**
     * 标题
     */
    private String title;
    /**
     * 内容
     */
    private String content;

}
```

- 上面界面对应的每一个项（例如，退款原因、处理方式等等），对应一个 Detail 对象。

picURLS ：图片地址数组字符串。

4. API

基于如下整理 API 类。

有赞云提供的商家退款API
有赞云提供的买家退款API

4.1 TradeRefundAPI

TradeRefundAPI ，退款维权 API 。

```
/**
 * 退款维权 API
 */
public interface TradeRefundAPI {

    // ========== 退款（买家） BEGIN ==========

    /**...*/
    YouzanTradeRefundConditionGetResult getCondition

    /**...*/
```

友情提示，实际场景下，API 会分拆成买家和卖家两个 API 类。

666. 彩蛋

其他和交易退款维权有关系的文档如下：

《如何操作批量退款？》
《微信支付－自有订单如何退款？》
《订单交易完成，还能退款吗？》

业务比较复杂，所以文档较多。

目前有赞提供 主动退款 功能，界面如下：

// TODO 6012 主动退款，未来补充

文章目录

回到首页