概要设计说明书

Revision History:

|  |  |  |
| --- | --- | --- |
| Who | When | What |
| 0.1 | 2018/2/5 | Draft |
| 1.0 | 2019/1/31 | POS获得身份证 |
| 1.0 | 2019/2/2 | 部署备忘 |
| 1.0 | 2019/2/12 | 1.1.1 数据库存储过程（SP）1.1.2 http请求与响应 |
| 1.0 | 2019/2/18 | 1.1.1.1 商品类别的大类T\_CategoryParent |

目录

[概要设计说明书 1](#_Toc50496670)

[1 引言 7](#_Toc50496671)

[1.1 编写目的 7](#_Toc50496672)

[1.2 背景 7](#_Toc50496673)

[1.3 定义/名词解释 7](#_Toc50496674)

[1.4 参考资料 12](#_Toc50496675)

[1.5 文档迭代指引 12](#_Toc50496676)

[2 总体设计 13](#_Toc50496677)

[2.1 需求规定 13](#_Toc50496678)

[2.1.1 系统结构/模块 13](#_Toc50496679)

[2.1.2 系统功能 14](#_Toc50496680)

[2.1.3 系统性能 17](#_Toc50496681)

[2.1.4 输入输出要求 18](#_Toc50496682)

[2.1.5 数据管理能力要求 19](#_Toc50496683)

[2.1.6 故障处理要求 19](#_Toc50496684)

[2.1.7 其他专门要求 19](#_Toc50496685)

[2.2 运行环境 20](#_Toc50496686)

[2.2.1 设备 20](#_Toc50496687)

[2.2.2 软件 20](#_Toc50496688)

[2.2.3 接口 22](#_Toc50496689)

[2.2.4 控制 23](#_Toc50496690)

[2.2.5 限制描述 23](#_Toc50496691)

[2.3 基本设计概念和处理流程 25](#_Toc50496692)

[2.3.1 创建商品 25](#_Toc50496693)

[2.3.2 商品采购 26](#_Toc50496694)

[2.3.3 收银（售卖）/支付流程 27](#_Toc50496695)

[2.3.4 盘点 31](#_Toc50496696)

[2.3.5 生成采购建议 33](#_Toc50496697)

[2.3.6 入库（1次或N次入库） 34](#_Toc50496698)

[2.3.7 数据同步机制设计 34](#_Toc50496699)

[2.3.8 促销 53](#_Toc50496700)

[2.3.9 仓管退货 56](#_Toc50496701)

[2.3.10 购物退货和退款流程 57](#_Toc50496702)

[2.3.11 同步块缓存（别于普通对象缓存） 58](#_Toc50496703)

[2.3.12 普通对象缓存（别于同步块缓存） 59](#_Toc50496704)

[2.3.13 （Android）收银员登录POS 61](#_Toc50496705)

[2.3.14 （Android）收银员交班 64](#_Toc50496706)

[2.3.15 （Android）POS（不是门店人员）登录服务器 65](#_Toc50496707)

[2.3.16 Action 和 SyncAction的职责 66](#_Toc50496708)

[2.3.17 Android http请求之小票格式 67](#_Toc50496709)

[2.3.18 Android http请求之基础资料 75](#_Toc50496710)

[2.3.19 商品毛利率计算 76](#_Toc50496711)

[2.3.20 Android POS开机SIT测试设计 82](#_Toc50496712)

[2.3.21 系统交付流程 84](#_Toc50496713)

[2.3.22 微信消息生成和处理流程 86](#_Toc50496714)

[2.3.23 公共DB和私有DB设计 98](#_Toc50496715)

[2.3.24 上传商品图片 99](#_Toc50496716)

[2.3.25 上传公司营业执照 99](#_Toc50496717)

[2.3.26 上传微信退款证书和退款授权 100](#_Toc50496718)

[2.3.27 会话管理 100](#_Toc50496719)

[2.3.28 小程序上会员/顾客登录的流程 101](#_Toc50496720)

[2.3.29 wpos.exe的启动流程 103](#_Toc50496721)

[2.4 结构 104](#_Toc50496722)

[2.4.1 系统逻辑结构 104](#_Toc50496723)

[2.4.2 系统模块设计 104](#_Toc50496724)

[2.5 功能需求与程序的关系 116](#_Toc50496725)

[2.6 人工处理过程 117](#_Toc50496726)

[2.7 自动化测试设计 117](#_Toc50496727)

[2.7.1 销售开单价格计算 117](#_Toc50496728)

[2.7.2 数据同步 117](#_Toc50496729)

[2.7.3 xxxxxxxxx 123](#_Toc50496730)

[2.8 版本升级设计 124](#_Toc50496731)

[2.9 尚未解决的问题 124](#_Toc50496732)

[2.10 部署备忘 124](#_Toc50496733)

[2.10.1 DB部署 125](#_Toc50496734)

[2.10.2 POS机身份识别问题 125](#_Toc50496735)

[2.10.3 基础资料在POS中的初始化 126](#_Toc50496736)

[2.10.4 Mysql 126](#_Toc50496737)

[2.10.5 Tomcat配置 126](#_Toc50496738)

[2.10.6 Log4j配置 127](#_Toc50496739)

[2.10.7 Jenkins健康检查 128](#_Toc50496740)

[2.10.8 皇后服务 128](#_Toc50496741)

[2.10.9 微信公众号配置 129](#_Toc50496742)

[2.10.10 HouseKeepService 130](#_Toc50496743)

[3 接口设计 130](#_Toc50496744)

[3.1 用户接口 130](#_Toc50496745)

[3.1.1 界面设计 130](#_Toc50496746)

[3.1.2 微信小程序界面 130](#_Toc50496747)

[3.2 外部接口 131](#_Toc50496748)

[3.3 内部接口 131](#_Toc50496749)

[3.3.1 小票打印与小票设置 131](#_Toc50496750)

[3.3.2 网页跳转 132](#_Toc50496751)

[3.3.3 Android界面跳转 132](#_Toc50496752)

[3.3.4 数据库存储过程（SP） 133](#_Toc50496753)

[3.3.5 http请求与响应 133](#_Toc50496754)

[4 运行设计 133](#_Toc50496755)

[4.1 运行模块组合 133](#_Toc50496756)

[4.2 运行控制 133](#_Toc50496757)

[4.2.1 配置 133](#_Toc50496758)

[4.2.2 服务器运行过程 134](#_Toc50496759)

[4.2.3 POS机运行过程 136](#_Toc50496760)

[4.3 运行时间 139](#_Toc50496761)

[5 系统数据结构设计 140](#_Toc50496762)

[5.1 逻辑结构设计要点 140](#_Toc50496763)

[5.1.1 综述 141](#_Toc50496764)

[5.1.2 商品 141](#_Toc50496765)

[5.1.3 入库次表 161](#_Toc50496766)

[5.1.4 采购订单 164](#_Toc50496767)

[5.1.5 仓管退货单 168](#_Toc50496768)

[5.1.6 盘点单 170](#_Toc50496769)

[5.1.7 微信事件推送消息表T\_WxEventPushMessage 171](#_Toc50496770)

[5.1.8 零售和收银 172](#_Toc50496771)

[5.1.9 门店用户 201](#_Toc50496772)

[5.1.10 消息 207](#_Toc50496773)

[5.1.11 博昕客户（公司）信息及其门店 210](#_Toc50496774)

[5.1.12 （公共DB）博昕内部员工表 224](#_Toc50496775)

[5.1.13 会员 225](#_Toc50496776)

[5.1.14 供应商 296](#_Toc50496777)

[5.1.15 仓库 297](#_Toc50496778)

[5.1.16 询价单 298](#_Toc50496779)

[5.1.17 促销 298](#_Toc50496780)

[5.1.18 销售目标 303](#_Toc50496781)

[5.1.19 全局配置表（配置性数据） 305](#_Toc50496782)

[5.1.20 同步缓存表 307](#_Toc50496783)

[5.1.21 Nbr\_bx全局配置表 309](#_Toc50496784)

[5.1.22 Nbr\_bx.T\_CurrentRelease 311](#_Toc50496785)

[5.2 物理结构设计要点 312](#_Toc50496786)

[5.3 数据结构与程序的关系 312](#_Toc50496787)

[6 系统出错处理设计 313](#_Toc50496788)

[6.1 出错信息 313](#_Toc50496789)

[6.2 补救措施 313](#_Toc50496790)

[6.3 系统维护设计 313](#_Toc50496791)

# 引言

## 编写目的

## 背景

软件名称：博昕POS-ERP

开发者：博昕软件研发团队

用户：中小型零售门店全体员工

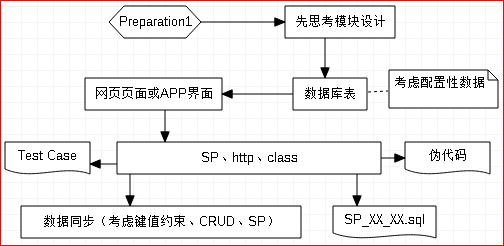
## 定义/名词解释

|  |  |
| --- | --- |
| 名词 | 定义/范围 |
| 瘦系统 | 只关注产品的主要部分，主要部分有数据和相应的程序程序。次要部分只有数据，几乎没有程序代码 |
| 肥系统 | 由瘦系统不断重构而来。重构会增减功能并不断完善整个产品 |
| CRUD | 即增删改查。C即Create，R即Retrieve，包含R1和RN（Retrieve1和RetrieveN），U即Update，D即Delete |
| 采购计划表 | 未选择供应商 |
| 采购订单 | 已经选择好供应商，已审核＋未审核 |
| 采购周期 | 1. 生成采购建议根据采购周期及周期内销量 2. 采购订单生成之后，到送货期限 |
| 商品类型 | 标准商品  非标商品（计重商品）  非标商品（计件商品） |
| 进货价 |  |
| 一码多品 | 商品种类惟一编码。举例：不同味的鱼 |
| 一品多码 | 一种商品有多个国际条形码。用于区域销售 |
| 一品多包装 | 散卖和整包卖，同价。整包卖和散卖价格不一致。会员价，不同店的价。支付方式不同价。每一种品类，都有一个最小单位，比如一罐加多宝。16罐装的时候，应该表明它与一罐加多宝的关系，16倍。主要用于计库存。散卖和整包卖利润不一样。DB存储的时候，当成多品处理，每品分别有个包装 |
| 备用金 |  |
| 供应商区域 | 供应商A只允许在广州销售，供应商B只允许在佛山销售，故须区别区域。这点对连锁店为必需 |
| 部门 | 举例，日班和夜班是两个部门，仓库1和仓库2属于两个不同部门 |
| 清空数据 | 除开商品资料的所有销售数据（包括出库、入库记录在内） |
| 中间结果表 | 设计中间结果表是为了打开网页时能够快速看到统计结果。  因为打开某些报表时，需要进行大量的计算，较为耗时，故预先在夜间任务中计算好结果，存储在中间结果表中。打开网页时，直接从中间结果表中读取数据进行计算，能大大简化计算过程，大大缩短网页响应的时间。 |
| 售前帐号 | 业务经理创建公司后售前帐号会被NBR自动创建。这个帐号可以为这个公司的POS机作初始化（比如获取自己的密码，下载基础资料等等）。它会在初始化完成后被禁用（否则可能对门店老板产生安全威胁）。 |
| 同步缓存 | NBR端存储同步对象的缓存。同步指同步到不同的POS机（多于1台POS机）。  它包含一个块，以及这个块被什么POS机同步过了。已经同步过的POS不再需要这个块，而其它POS需要。它可以是CUD块。  C块只会有一个  U块必须在C块之后，因为必须先有创建后有修改。  D块必须在C块之后。D块一旦出现，C块和U块都会被删除，因为既然此块即将被删除，那C和U出来的对象同步到POS机没意义 |
| 普通缓存 | 存储普通对象（不需要同步的对象）以外的对象的缓存 |
| 保持商品名称的历史原貌 | 部分表需要保存旧的商品名称，比如指定范围的促销、采购、入库。  比如，创建指定范围的促销时，将此时的商品名称插入到本字段中保留。此后不管T\_Commodity中的名称如何变化，此处不变，即“保持历史原貌” |
| 服务商 | 门店通过服务商向微信发送收款退款请求。比如我们博昕，就是一个服务商 |
| 子商户 | 子商户是使用服务商的服务的商业实体。比如使用我们收银系统的门店、商场等 |
|  |  |
| 非退货型零售单 | 普通零售单，卖出商品时生成 |
| 退货型零售单 | 对普通零售单进行退货生成的特殊零售单。它必须对应一张普通零售单，DB字段为T\_RetailTrade.F\_SourceID |
| ~~会员和会员单元~~ | ~~会员单元指单个人，会员是会员单元的组合。比如一个家庭是一个会员，家庭成员作为会员单元来消费。会员单元消费，全部是给会员增加积分而不是给自己增加积分~~ |
| PSB或博销宝 | 博销宝（Prosalesbox，不要漏掉s），博销宝整个产品线的统称，它是移动互联网时代的门店运营平台 |
| PSB-CMS | Client Management System（CMS），博销宝-博昕客户管理系统，是OP管理各个公司/门店的后台，创建、修改公司资料等 |
| PSB-CMP | Customer Mini-Program（CMP），博销宝-顾客/会员小程序 |
| PSB-MMP | Merchant Mini-Program（MMP），博销宝-商家小程序 |
| PSB-NBR | New Businees of Retail（NBR），博销宝管理后台，管理商家的资料、零售单，生成报表等，同时提供会员、积分、优惠券等服务 |
| PSB-APOS | Android POS（APOS），博销宝在商米收银机上的收银App |
| PSB-WPOS | Windows POS（WPOS），博销宝在Windows上的收银软件 |
| PSB-UPOS | Ubuntu POS(UPOS)，博销宝在Ubuntu上的收银软件 |
| PSB-POS | 博销宝收银前台，目前包含 PSB-WPOS和 PSB-UPOS |
|  |  |
|  |  |

## 参考资料

* 《功能需求说明书》。
* NTP协议。原文地址：<http://www.cnblogs.com/TianFang/archive/2011/12/20/2294603.html>。

## 文档迭代指引

文档迭代之时，必须遵循以下顺序修改文档：

此外，缓存也许需要考虑。

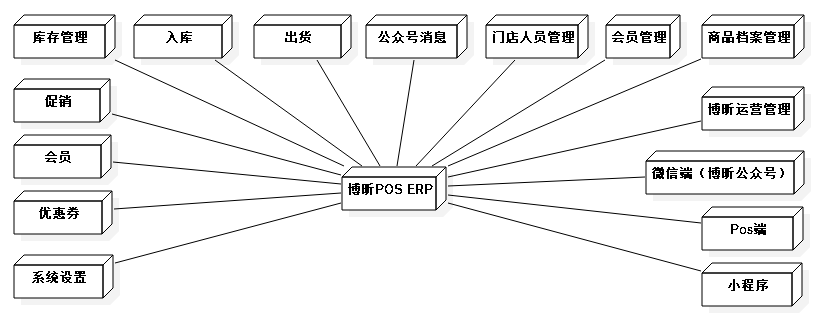
# 总体设计

## 需求规定

### 系统结构/模块

图示

描述已自动生成

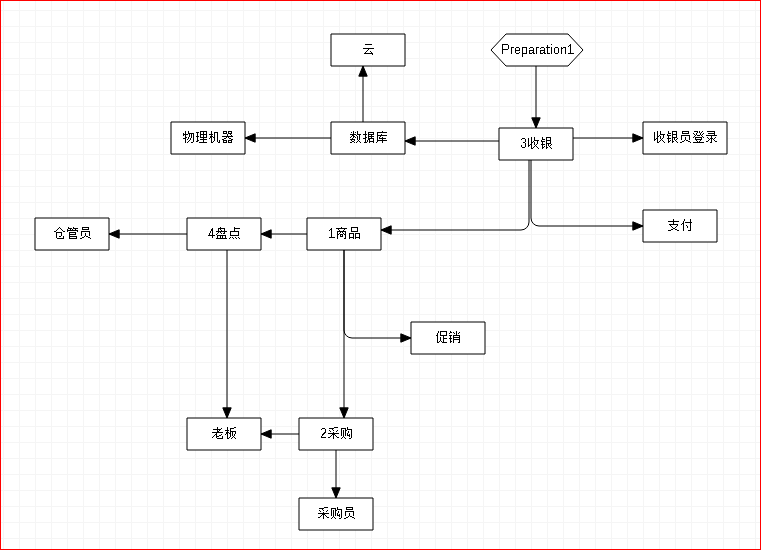


### 系统功能

系统开发分为二个阶段：

* 瘦系统阶段。
* 肥系统阶段。

瘦系统功能思维导图及开发优先级别如下图所示：



### 系统性能

#### 精度

对于每一笔交易，计算和存储精确到小数点后面6位，显示的时候只显示后面2位。

#### 时间特性要求

#### 安全可靠性

数据库用户的某些数据不可泄漏，比如数据库密码。

老板、门店和POS的钥匙即使拥有密码都不能看见明文。必须同时拥有源代码和钥匙加密后的串才能解密。

这里是否设计不同的DB用户？＞＞＞＞？？？？有待再设计。

？？？？？？？？

对服务器的请求不能太频繁，否则拉黑并触发警报。

商户API证书上传到的目录不能通过web访问和下载。此证书用于处理退款。参考[退款流程](#_退款流程)。

以下是微信的建议：

API证书安全：1.证书文件不能放在web服务器虚拟目录，应放在有访问权限控制的目录中，防止被他人下载；2.建议将证书文件名改为复杂且不容易猜测的文件名；3.商户服务器要做好病毒和木马防护工作，不被非法侵入者窃取证书文件。

#### 灵活性

### 输入输出要求

### 数据管理能力要求

### 故障处理要求

断电处理：？？

本地故障出现后，若无法解决，可清空本地数据，再次同步服务器数据。以达到统一的目的。

### 其他专门要求

本产品为BS架构。B限制为桌面电脑浏览器、手机微信小程序或商米T1访问。不允许使用其它设备上的B对系统进行访问。

## 运行环境

### 设备

瘦系统设备：

* 商米T1一体机：运行App。DB为SQLite。

肥系统设备：

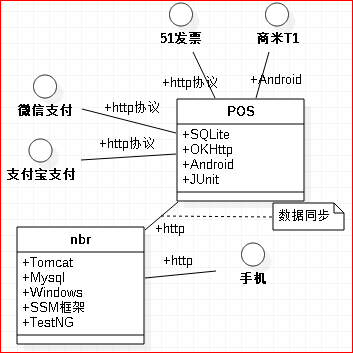
* 微信手机客户端：运行小程序版本。
* 商米T1一体机：运行App。DB为SQLite。
* 云端：云服务器，安装有Mysql及Tomcat。

### 软件

以下操作系统或软件运行本系统：

* 本地一体机操作系统：Android x.x.x
* 远程服务器操作系统：Windows。CPU？内存？
* JDK版本：1.8或以上
* Wamp版本：2.5或以上（内含Mysql）
* Tomcat版本：9.0或以上
* 微信版本：2018或以上

### 接口



最终演化到的肥系统如上图所示。

瘦系统只关注和实现本地系统部分。

### 控制

控制分为以下几部分：

* 本地系统：本地系统（Tomcat/Mysql）随商米T1开机而自启。在断电或关机时，系统随着关闭。
* 数据同步：按照用户设定的频率检查数据的更新以及将最新数据同步到云端。
* 云端控制指令：按照用户设定的频率检查云端指令并执行之。？？？
* 云端系统：由于云端系统托管在阿里云上，其完全通过远程桌面进行管理。
* 手机端：由手机使用者接入和断开云端。

### 限制描述

#### 收银限制

POS机收银的现金不能超过购买的商品的总价太多，比如错误地将收到的现金输入为1亿。应当有所限制。

示例：

总价＜100，输入＜＝200

总价＞＝100，输入＜＝5倍（即500）

#### 收银机上Android APP的限制

Android APP只做和零售相关的功能，不做全局配置性的功能。

此举可以避免多POS机一云服务器的情况下，处理复杂的数据同步。

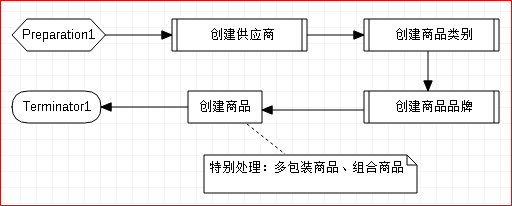
#### 普通电脑的限制

普通电脑用来登录后台，不能当APP用来做零售。

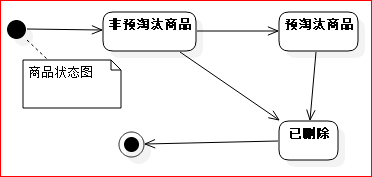
#### 微信小程序的限制

## 基本设计概念和处理流程

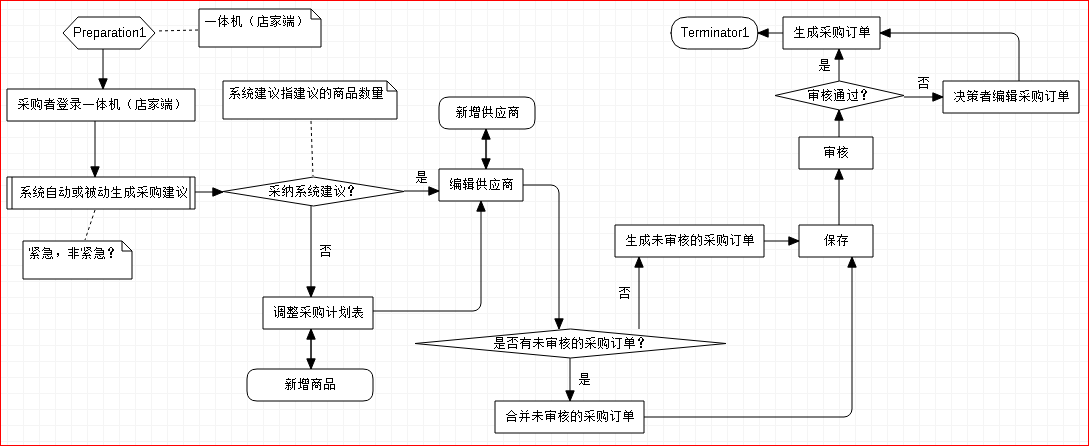
### 创建商品



商品状态图：

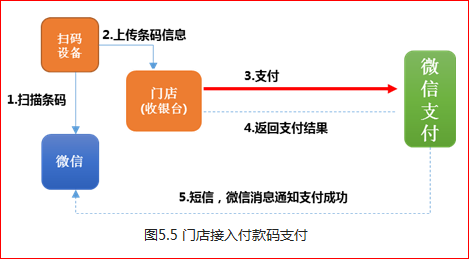


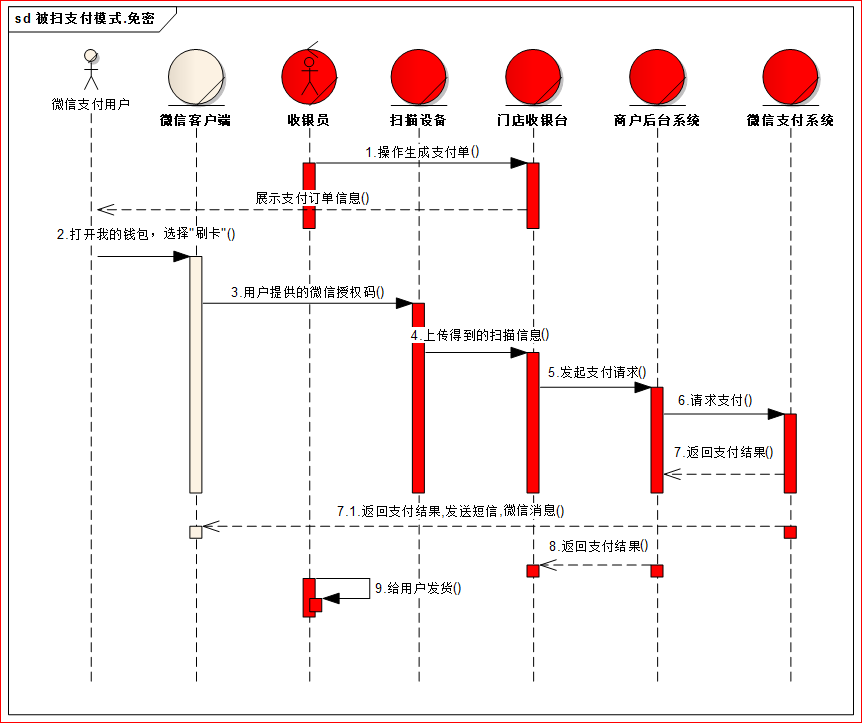
### 商品采购

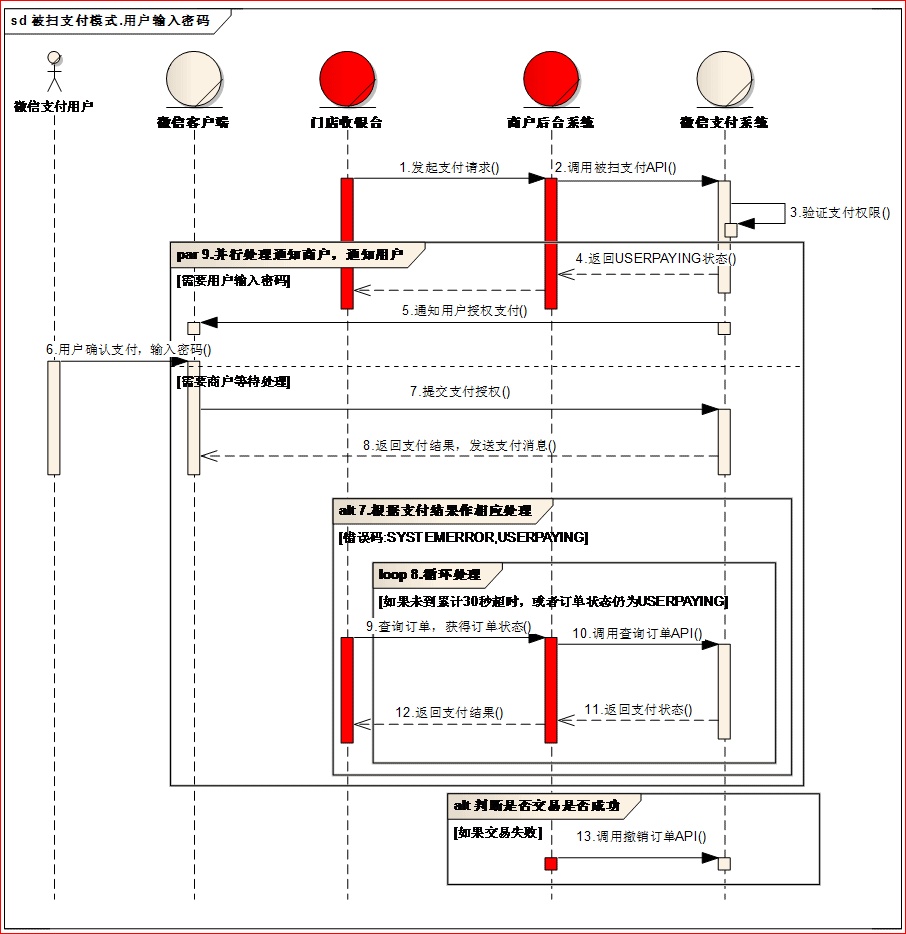


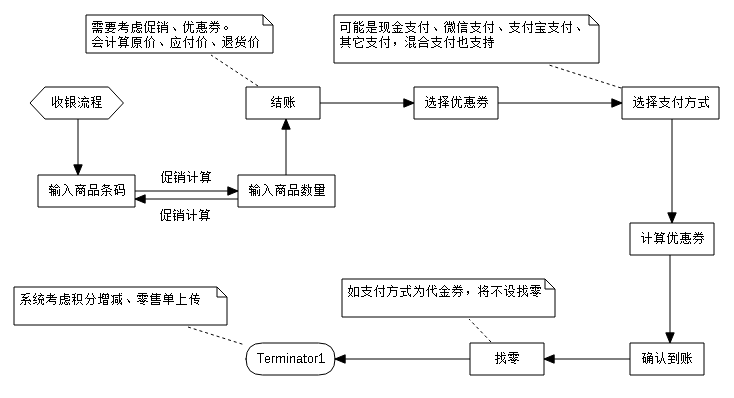
### 收银（售卖）/支付流程

参考：<https://pay.weixin.qq.com/wiki/doc/api/micropay_sl.php?chapter=5_4>

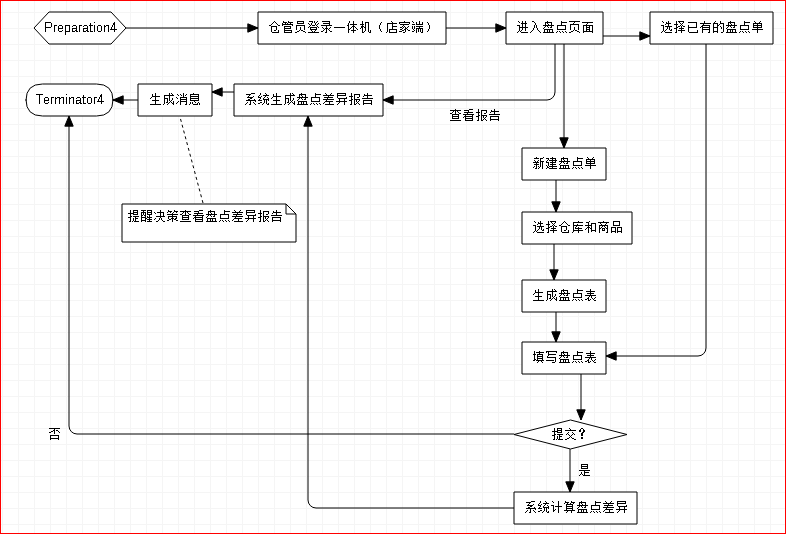




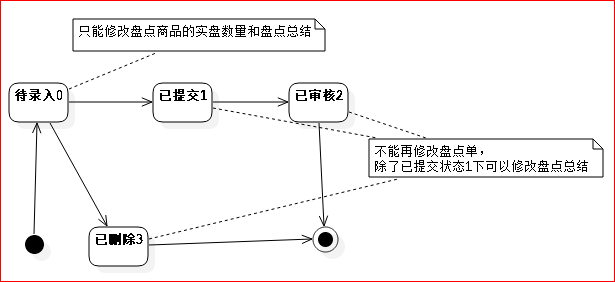




### 盘点

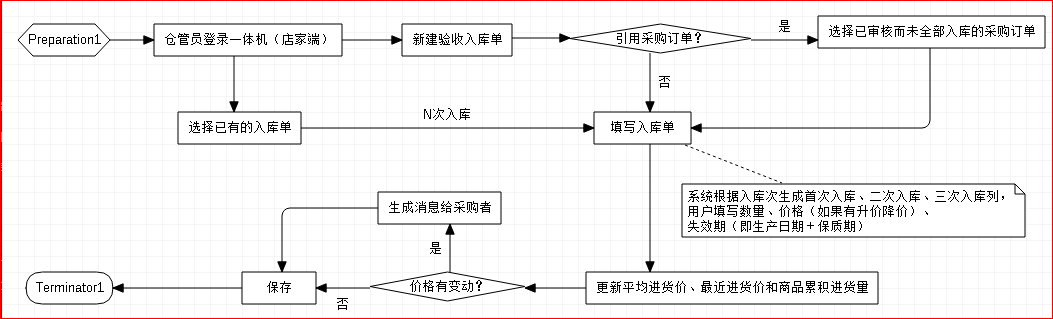
盘点流程：

盘点单状态图：



### 生成采购建议

### 入库（1次或N次入库）



### 数据同步机制设计

本节特指POS机本地数据库与远程数据库同步机制的设计。

有三种操作类型的数据需要同步： C型=添加型数据 U型=修改型数据 D型=删除型数据。

其中D型即特殊的U型，只设置一个字段用来标识该行已经被删除。故后面不再用D型的字眼。

#### 总体指导思想

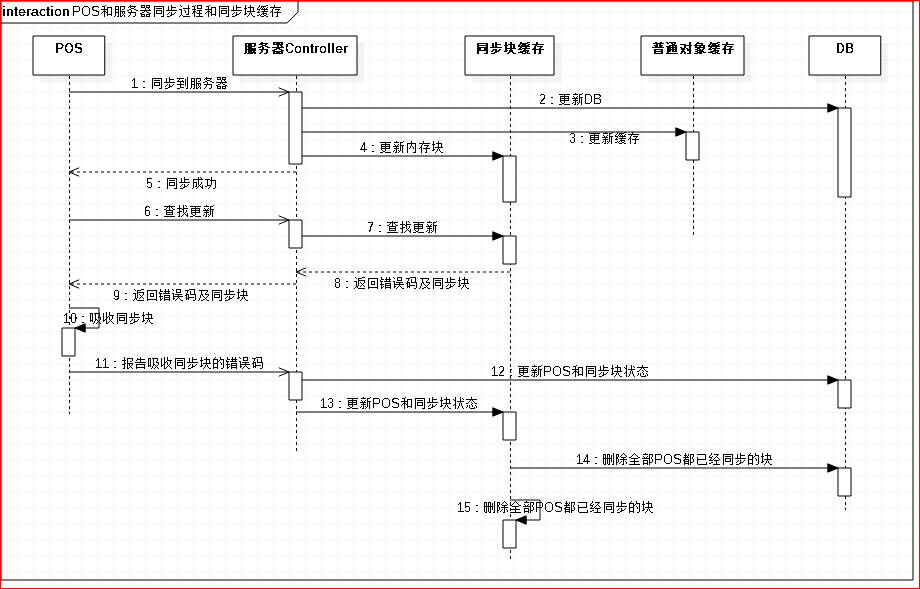
* 限制功能。以云服务器为中心，限制POS APP和电脑的功能，以便降低数据同步的复杂性。
* 数据操作类型。如上所述，被同步的数据分为三种：添加（C型）、修改（U型）。
* 写锁定。某些数据正在更新的时候，不允许写，只允许读。
* 防止重复同步。被同步的数据要考虑重复性问题。一行数据可能已经被上传过，一行数据的key在对方可能不是key，双方应当有统一key。
* 备份。执行同步前先备份数据库表以应付同步失败的情况。
* 部署失败预案。部署任何与DB相关的新版本时，都让系统自己执行部署过程，不能人工执行！执行期间，不允许有任何数据同步操作，确保无任何死角引起同步问题！
* 反部署。部署任何与DB相关的新版本时，既有部署过程，也有反部署过程，以恢复数据。
* DB更改影响同步。见[部署备忘](#_2.10_部署备忘)。为此需要建立一个测试环境。见[自动化测试设计](#_2.7_自动化测试设计)。
* **全部下载。极端同步。在极端的情况下，将POS上的所有数据（销售开单的数据必须提前同步到云）删除，重新向服务器下载对应的数据以达成同步**。
* （瘦系统不做）MD5。由于数据同步极端重要，在传输过程中，可以设置是否对传输的文件生成MD5，以便应对复杂网络。

#### 同步类型

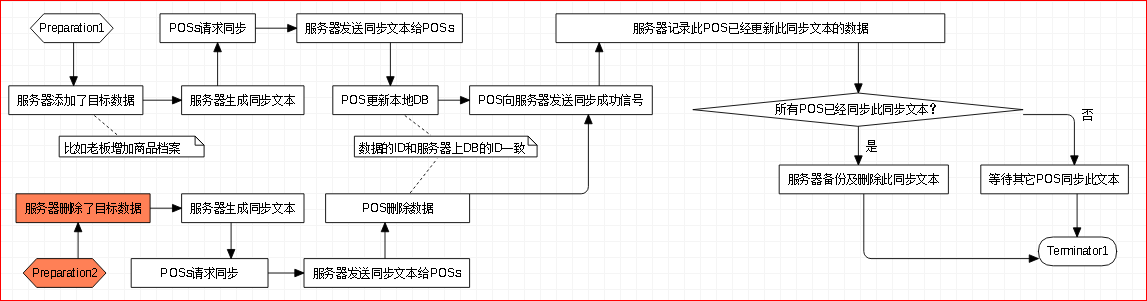
系统提供自动同步、手动同步。

#### 同步方法

##### UML视角下的同步方法

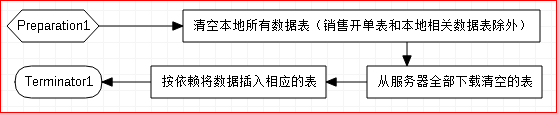


##### 部分下载

为节省流量减少服务器负担，一般只让POS下载受影响的数据。下图为其处理流程（其中C型和U型流程几乎一样）：

##### 全部下载

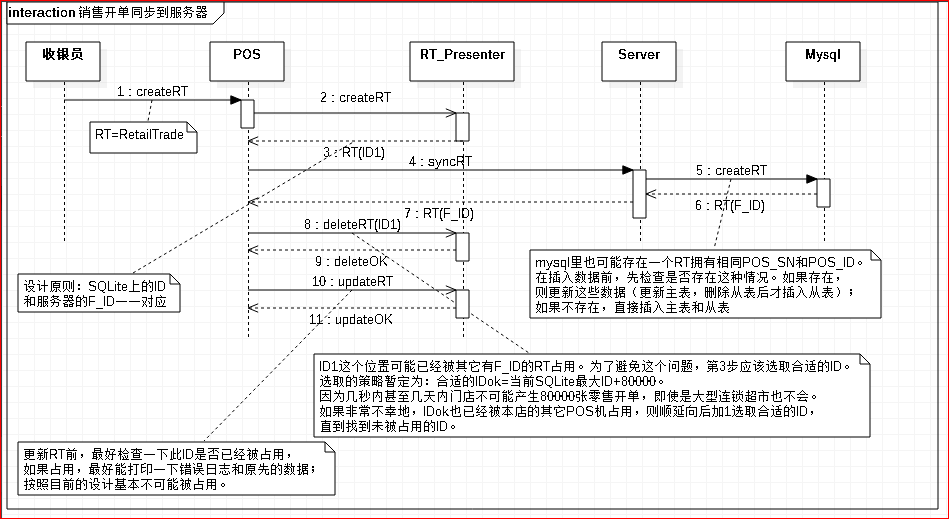
下载全部数据，通常发生在添加新的POS机之后。有时为了还原POS，也需要使用此功能。下图为其处理流程：



注意：部分表存在极多的约束，需要对每个表分别进行约束分析和设计。

##### 上传数据

###### 销售开单



参考[销售开单](#_销售开单)。

###### 会员

#### 系统版本更新预案设计（瘦系统暂不做）

如果系统版本有更新，这些更新可能对同步形成重大影响。这些更新可能导致的问题及其解决方案如下：

|  |  |  |
| --- | --- | --- |
| 序号 | 问题陈述 | 解决方案 |
| 1 | 销售开单的表结构发生变化 | 针对不同版本，有不同的数据格式和同步方法 |
| 2 |  |  |
| 3 |  |  |
|  |  |  |

时间同步：NTP

#### 需要同步的项目

|  |  |  |
| --- | --- | --- |
| 项目 | 同步时机 | 同步方法 |
| 时间 | 收银员每次登录POS APP | 服务器到POS机。服务器预先校正好时间 |
| **销售开单** | 收银员每次登录POS APP | POS机到服务器。每隔一个时间段同步一次。如果无销售开单，不会作无谓的同步。 |
| 收银员每次查询销售开单详情 | 服务器到POS机。销售开单详情以服务器为准。每次查询详情都会向服务器请求数据，防止其它机器对目标开单有过修改。如果本地POS机断网，应当提醒收银员修改开单的操作存在风险。恢复网络后，此开单会被上传到服务器。  　　同步时，应当更新POS机上的ID字段，以便知道服务器上的ID是多少。 |
| 会员 |  |  |
| 商品档案 | 收银员每次登录POS APP或服务器上的数据有变 | 服务器到POS机实时同步 |
| 促销信息 | 收银员每次登录POS APP或服务器上的数据有变 | 服务器到POS机实时同步 |
| 设置 | 收银员每次登录POS APP或服务器上的数据有变 | 服务器到POS机实时同步 |
| 门店人员信息 | 收银员每次登录POS APP或服务器上的数据有变 | 服务器到POS机实时同步 |
| 机构 | 收银员每次登录POS APP或服务器上的数据有变 | 服务器到POS机实时同步 |
| 商品调价单 | 收银员每次登录POS APP或服务器上的数据有变 | 服务器到POS机实时同步 |

#### 同步方法细节

##### 时间同步

###### 原理

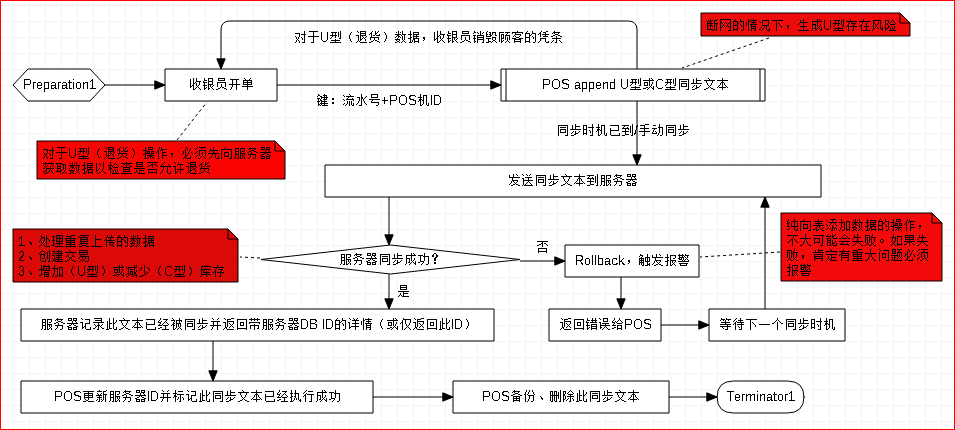
按照NTP协议（简化数据结构）实现POS APP到服务器的时间同步。见[参考资料](#_参考资料)。

###### 实现步骤

##### 

##### 销售开单

C型上传和U型上传：

U型是特殊的C型，其产生原因主要是顾客退货。顾客退货，并非对原先的销售开单产生修改，而是另外开一张新单，其中引用原先的销售开单。

##### 商品档案

###### 依赖分析

和[门店人员信息](#_门店人员信息)一样，数据表也有状态字段，其依赖处理遵从一样的原则。

###### 部分下载

同步文本法（互参：[同步文本数据结构](#_5.1.18_同步文本数据结构设计)）（互参：[部分下载同步流程](#_部分下载)）

###### 全部下载

即[全部下载同步法](#_全部下载)。

##### 促销信息

###### 依赖分析

###### 部分下载

###### 全部下载

##### 设置

###### 部分下载

参考[部分下载一般同步方法](#_部分下载)。

###### 全部下载

参考[全部下载一般同步方法](#_全部下载)。

##### 会员

###### 依赖分析

###### 部分下载

###### 全部下载

##### 门店人员信息

###### 依赖分析

C型数据：此依赖可忽略。

U型数据：和C型一样。

D型数据：参考[数据表T\_Staff](#_5.1.7_门店用户)。由于删除操作不实际进行物理删除，只将此员工状态改为离职，故此依赖可忽略。

###### 部分下载

参考[部分下载一般同步方法](#_部分下载)。

###### 全部下载

参考[全部下载一般同步方法](#_全部下载)。

##### 机构

###### 依赖分析

###### 部分下载

###### 全部下载

即[全部下载同步法](#_全部下载)。

##### 商品调价单

###### 依赖分析

无外部依赖。

###### 部分下载

参考[部分下载一般同步方法](#_部分下载)。

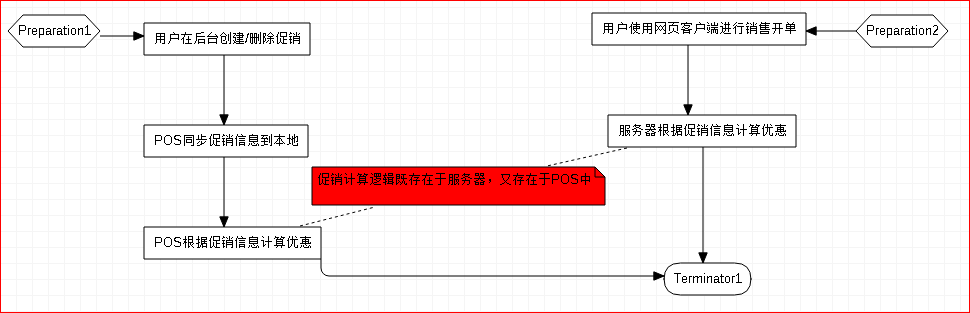
###### 全部下载

即[全部下载同步法](#_全部下载)。

### 促销

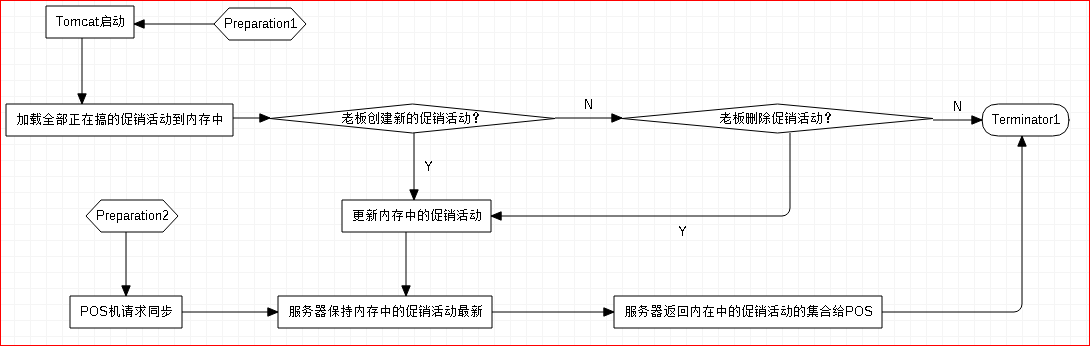
#### 流程

促销的主要业务流程为：



由于POS机和服务器都需要使用促销计算逻辑，所以此逻辑设计为一个独立的组件。它的输入是一个交易单，输出是一个交易单和计算优惠的过程以显示给用户看。

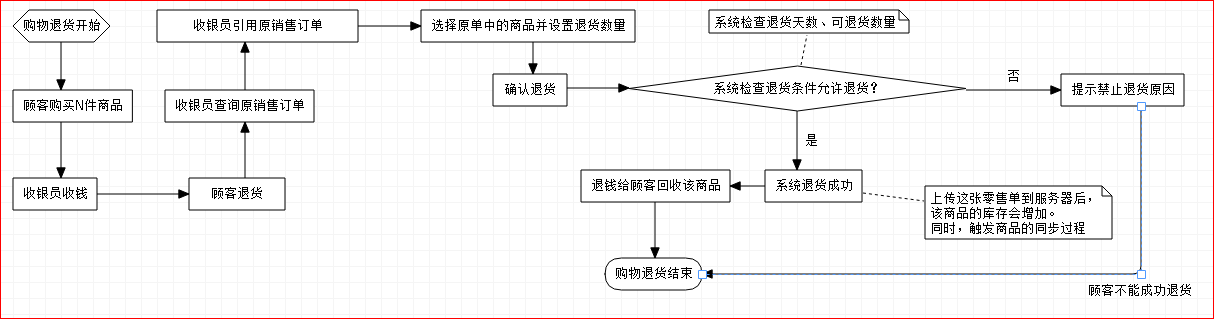
促销缓存和同步过程：

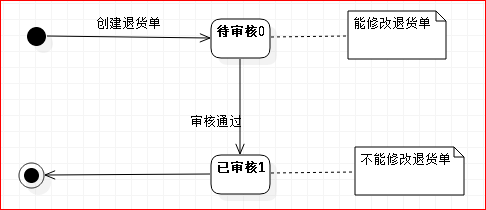


#### 促销计算方法

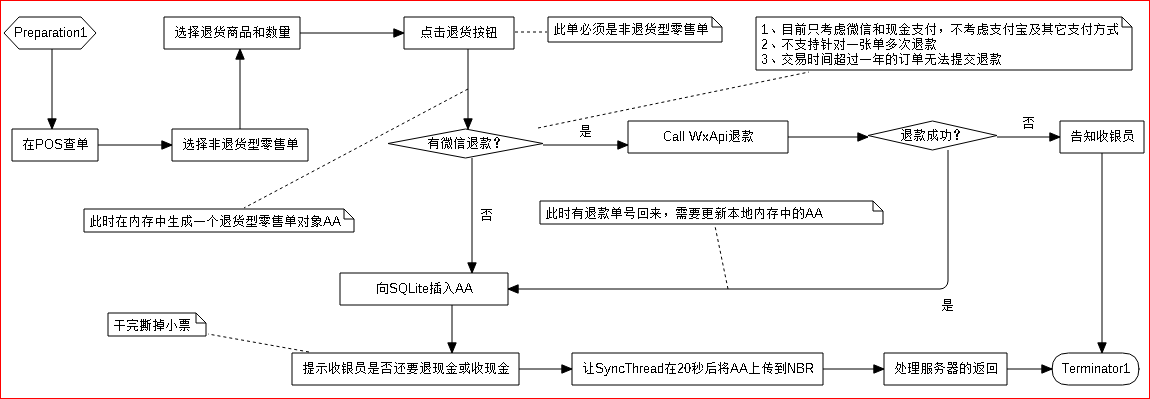
* 先算指定范围的促销，再算全场范围的促销。
* 如果指定范围的促销有多个，则找出优惠力度最大的。
* 如果全场范围的促销有多个，则找出优惠力度最大的。

### 仓管退货





### 购物退货和退款流程

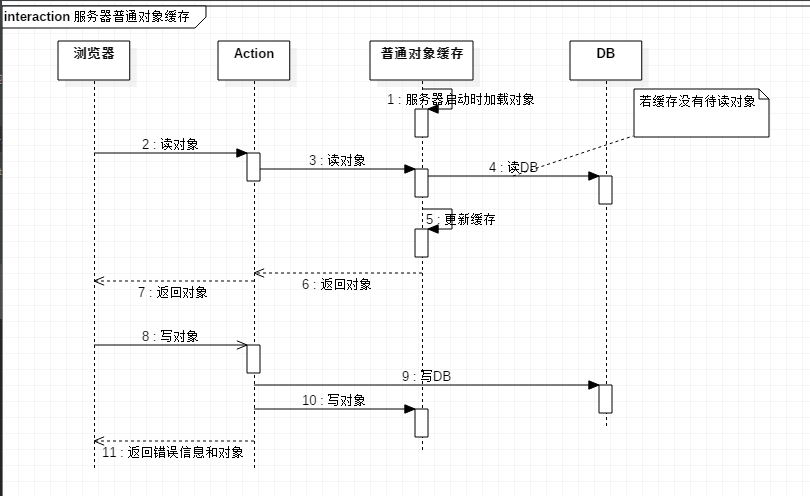


支持退款的商户API证书要确保安全有效。参考系统[安全可靠性](#_安全可靠性)。

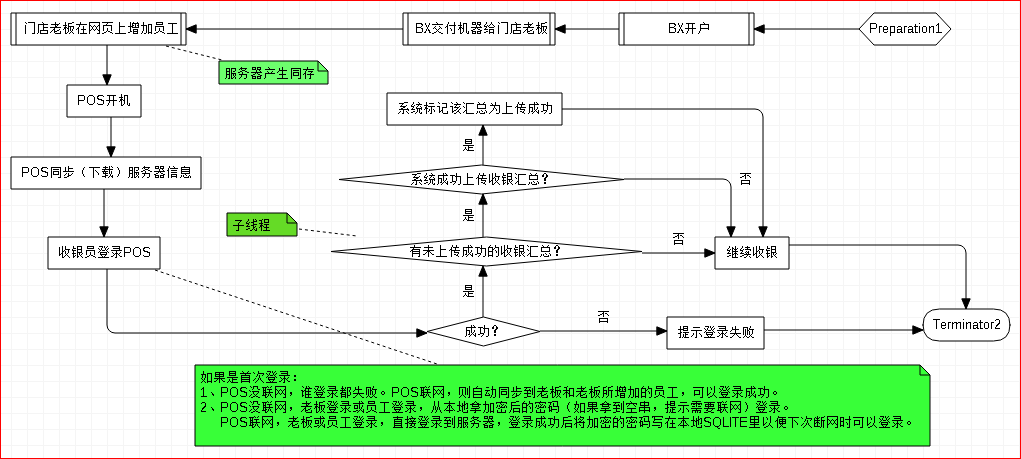
### 同步块缓存（别于普通对象缓存）

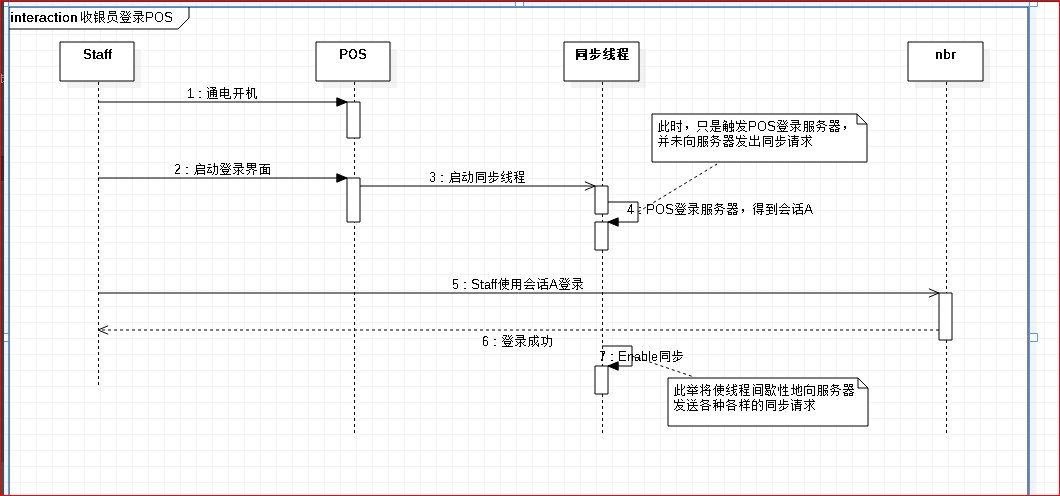
参考[UML视角下的同步方法](#_UML视角下的同步方法)。

### 普通对象缓存（别于同步块缓存）



### （Android）收银员登录POS





其它流程或规定：

1、开机启动就登录pos。

2、点击登录界面的登录按钮后，

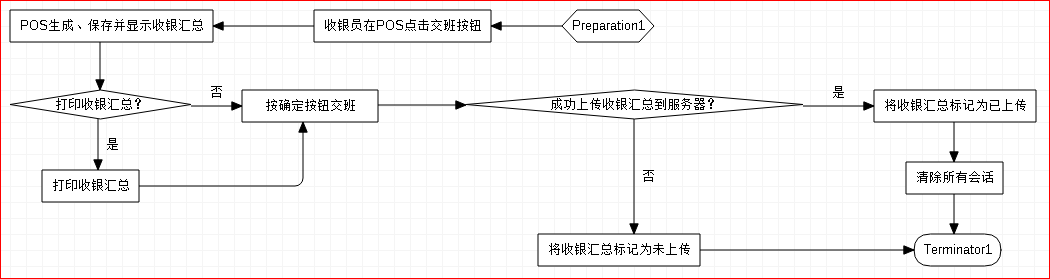
1）如果pos已经登录成功，则不再重复登录pos，直接登录staff。

2）如果pos没登录成功，则先登录pos，成功后再登录staff。

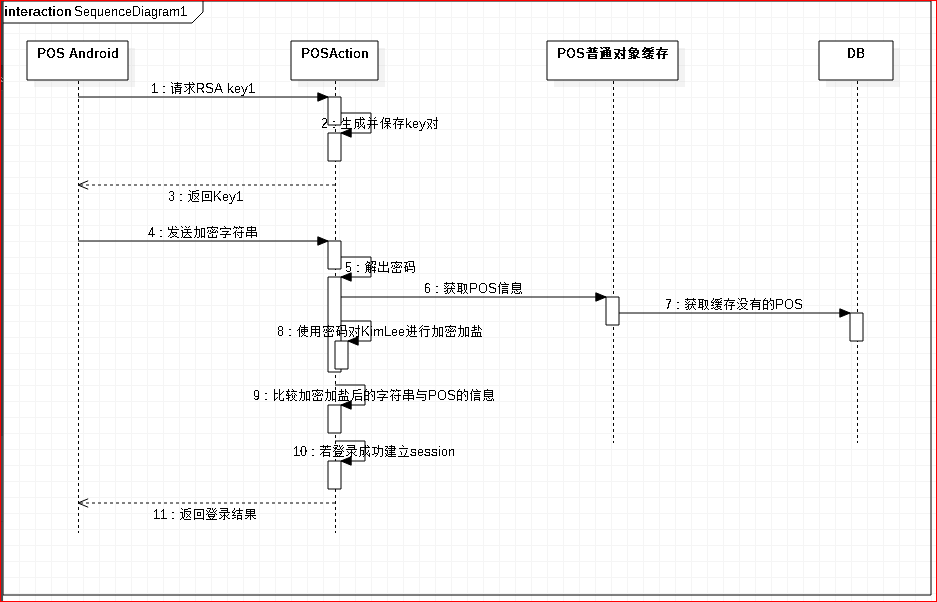
3、无论pos还是staff，只要其中一个登录失败，GlobalController中的sessionID都会设为null。下次登录，将会从pos登录开始，成功了再到staff的登录。这个约定非常重要，大家必须记住。

4、开机后，如果没有网络，不会进行网络登录，只会在本地SQLite验证用户名和密码。

### （Android）收银员交班



### （Android）POS（不是门店人员）登录服务器



### Action 和 SyncAction的职责

|  |  |  |
| --- | --- | --- |
|  | 何时使用Action | 何时使用SyncAction |
| 描述 | 普通控制器，处理业务对象的CRUD | 同步控制器，处理业务对象的CRUD之外，还要处理CUD操作引起的同步过程。 |
| 使用场合 | 业务对象不需要同步到不同的DB，只在中心服务器Mysql读写数据 | 业务对象需要同步到不同的DB，比如服务器的Mysql中的业务对象需要同步到不同POS机上的SQLite |
| C |  | 尽可能使用SyncAction。  1）考虑C VIP（无从表，支持小程序或POS、前端），应该全由SyncAction处理，不必重复写Action的C。网页C的话，可以同步到小程序或POS。小程序或POS C的话，网页请求必定是正确的数据，因为DB已经更新。  2）考虑C Smallsheet（有主从表，仅支持POS端）。必须用SyncAction。  3）考虑C 采购订单（有主从表，支持小程序、前端）。用SyncAction，不用Action。目前小程序未做，所以用Action。 |
| R1 | 返回此ID对应的对象 | 不支持的操作 |
| RN | 返回一页对象。页的size够大，则返回全部对象 | 返回同步块对应的对象列表。每个对象使用int2字段排序，使用string1字段标识为C型U型或D型。客户端按顺序同步更新这些对象到本地。如果这些对象出现同一个ID的N个对象，则可以考虑只同步其中一个。这些对象之间不应该有互相依赖的关系，尤其是外键依赖，否则在更新本地数据时很可能出错。 |
| U |  | 和C一样，尽可能使用SyncAction |
| D | 如果业务只支持网页端的对象，则不必用SyncAction。否则，必须使用SyncAction | |

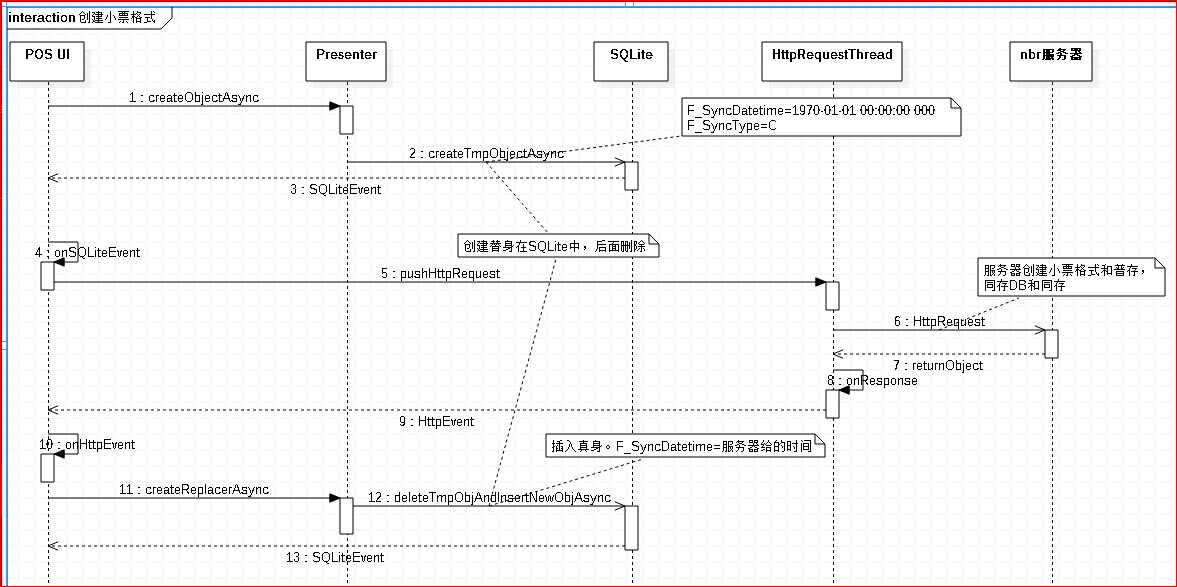
### Android http请求之小票格式

要点：所有服务器返回给POS的用于同步的MODEL，都会在服务器端设置同步时间字段，表明这条数据已经同步给了服务器。

#### 创建小票格式（SQLite->Http->SQLite）

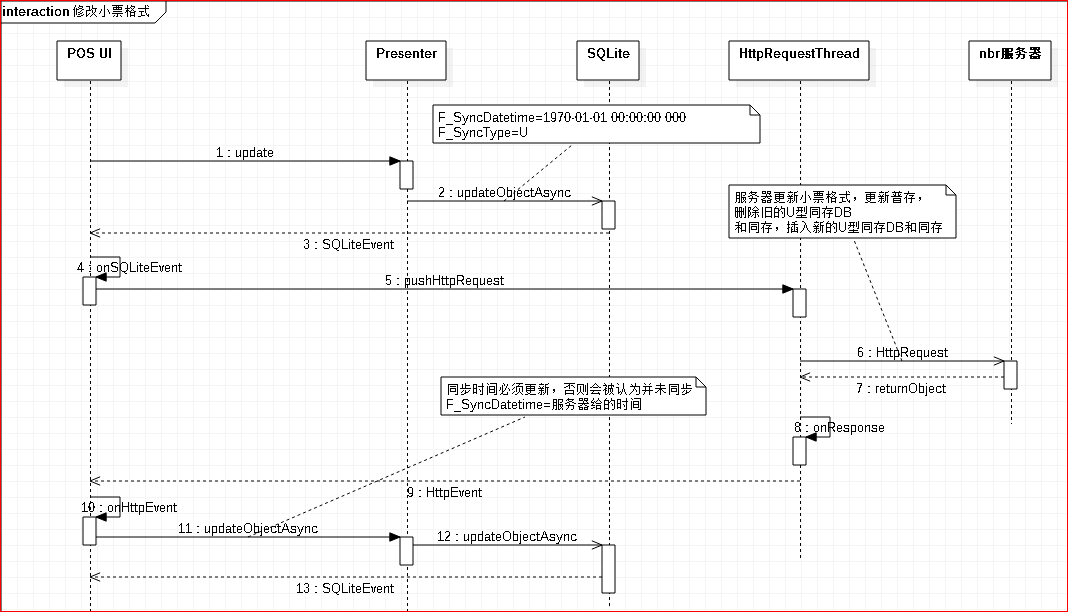
创建小票格式，先在POS的SQLite内插入一个临时小票格式对象（即替身，替身的**F\_SyncDatetime**为1970-01-01 00:00:00 000），然后上传它到服务器。服务器将此对象插入MYSQL，返回一个真正的对象（即真身，真身的**F\_SyncDatetime**不为1970-01-01 00:00:00 000，F\_ID为服务器Mysql生成的ID）给POS。POS收到此对象后，删除替身，并插真身到SQLite中。

其流程图为：



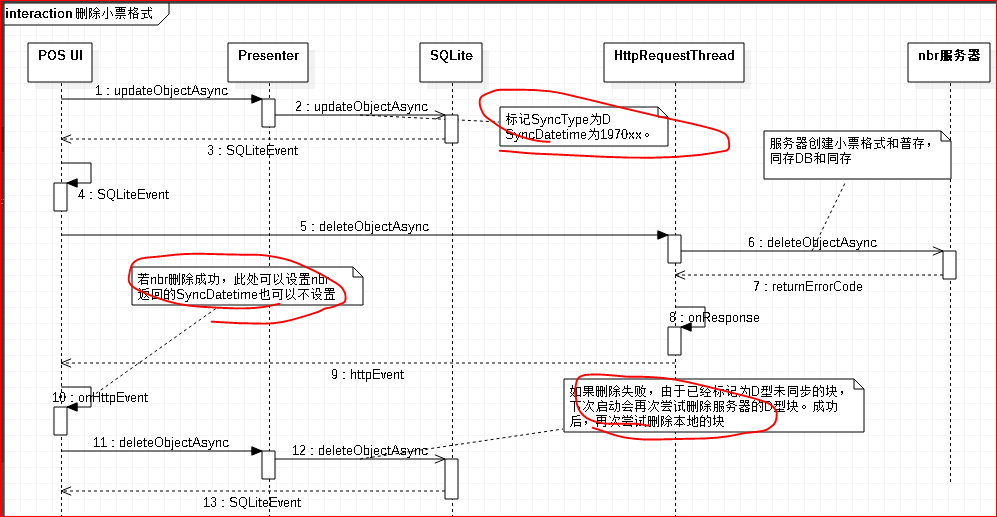
#### 修改小票格式（SQLite->Http->SQLite）

要点：修改对象时，要将字段**F\_SyncDatetime**设为1970-01-01 00:00:00 000，表明此数据未同步。服务器会在返回对象前将此字段设置为当前时间。

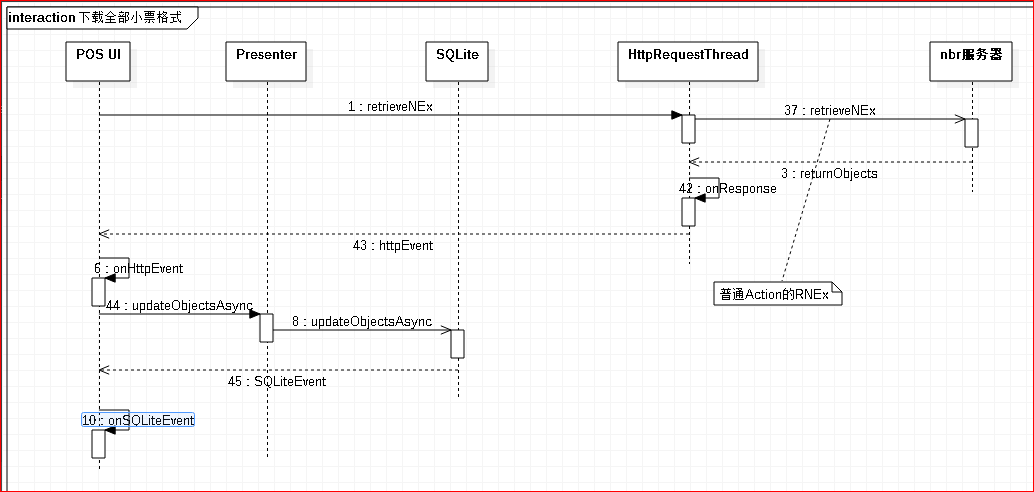


#### 删除小票格式

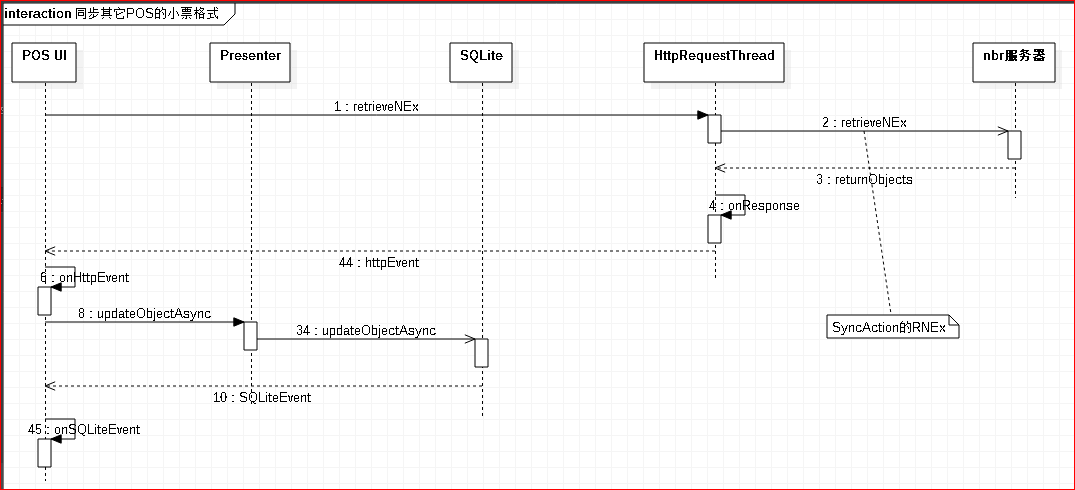
注意：这里是先删除本地SQLite数据再发出网络请求的。



#### 下载全部小票格式（普通Action的RN）



#### 同步其它POS的小票格式（SyncAction的RN）



### Android http请求之基础资料

### 商品毛利率计算

#### 问题与挑战

毛利率计算，是一个复杂的问题。如果一件商品A，进货价为1，零售价为2，库存3件，销售4天完毕，其毛利为：

（2-1）X3=3

毛利率为：

（2-1）/2X100%=50%，即（售价-此时的进货价）/售价

~~或~~

~~(2X3-1X3)/(1X3)X100%=100%~~

但现实世界发生的事并没有如此简单，表现在：

* 零售价变动。不同时间一个商品的售价可能变化。计算它的利润，不能以一个固定的价格计算。
* 进货价变动。不同时间进货价可能不同。计算它的利润，不能以一个固定的价格计算。
* 入库。商品的零售价和进货价都可能变化。这时，如果新入库的商品是有不同的零售价和进货价，其利润计算将变得复杂。
* 促销时，商品的利润计算方式与其它情况计算的不一样。利润甚至为负数。
* 购物退货时，利润怎么计算？算收银时的利润还是算退货时的利润？
  + 促销时利润计算方法不同，需要查看回促销时的退货规则。按促销退货规则计算。
  + 退货时，重算收银时的利润。
* 仓管退货。不论何时，对于入库，都可能有退货行为。

#### 解决方案论述

基本论述和设计：

* 对于每种商品，同一批次入库，其内的商品的进货价是一样的。
* 商品A入库B件，进货价是X，售价是Y，则其利润为B\*（Y-X），毛利率为B \*（Y-X）/ (B \* Y) \* 100%。
* 商品A入库B件，进货价是X，售价是Y，其售价在售出N件后变为W，则利润为（B-N）\*（Y-X）+ N \* （W-X），毛利率为（B-N）\*（Y-X）+ N \* （W-X）/ （（B-N）\* Y+ N \* W） \* 100%。
* 商品A入库B件，进货价是X，售价是Y。在促销发生时，假如有N件参与促销，促销价为Q，则其利润为（B-N）\*（Y-X）+ N\*（Q-X）。这可能令利润为负数。
* 商品A入库B件，进货价是X，售价是Y。购物退货A商品U件时，需要找到当时的零售单，对当时的对应的入库批次进行退货，以便知道当时的进货价M，当时的利润将会减少U \*（Y - M）。
* 商品A入库B件，进货价是X，售价是Y。进货发生退货，属于哪个入库批次的就在哪个批次进行退货，退货的数量不能大于入库的数量。因为是按原入库的商品的进货价进行退货，没有销售行为，所以没有利润的变化，也没有毛利率的变化。
* 毛利率的计算是因销售行为而起，故：
  + 零售单必须与时间及商品的进货价、售价、数量建立关系。而进货价与入库批次是一一对应的。故一张零售单，必须与时间及商品的进货价、售价、数量、入库批次建立对应关系。
  + 不论是何种促销在起作用，最终的售价都会按比例摊到每个商品头上。
  + 零售单在服务器创建后，需要令当值入库批次中的相应商品减少相应的数量。如果当值入库批次已经卖完，则剩下的数量应与下一入库批次发生关系。这会出现一张零售单对应N个入库批次的情况，计算利润时需要考虑不同的入库批次对应不同的进货价。故商品表应当增加当值入库批次字段F\_CurrentWarehousingID，表明当前是哪个入库批次的商品正在出售，可以出售多少件。零售单的商品表再增加从表零售单商品来源表T\_RetailTradeCommoditySource记录一个零售单商品中的N件分别对应哪几个入库批次及其数量，增加零售单商品退货去向表T\_ReturnRetailTradeDestination，记录退货的商品的数量和对应的入库ID，以便：
    - 退货时，可以退回哪个入库单中（退回到当值入库ID对应的入库中），当时卖出去时，是什么价格。退回了多少件且这几件对应哪个入库ID。
    - 计算利润时，可以知道利润是多少。

入库单的商品表中，增加可出售数量字段F\_NOSalable，表明当前的入库单商品表还能出售多少商品。

* + 对商品的退货，需要重建该商品的当值入库批次字段F\_CurrentWarehousingID，以便出售时，按正确的入库批次时的进货价计算利润。
  + 商品的平均进货价字段不再有作用。相关的业务代码需要删除。

解决方案示例一：



解决方案示例二：



利润和毛利率计算思路：

* 利润和毛利率计算需要从零售单出发。
* 年月日报表，需要大量的计算，系统需要创建一个零售日报汇总表T\_RetailTradeDailyReportSummary，每天都针对零售单中的商品生成利润信息插入到利润表中。每一天中的每张零售单都包含了必要的信息用于计算利润和毛利率，所以每天的利润表汇总起来可以推算到每月、每季度、每年的利润及其毛利率。
* 计算利润时，需要考虑一张零售单是否在后来有退货行为。如果有，需要减少相应的销售额。退货时也是生成一张零售单，只不过这张零售单的收入是负数。目前在DB中存储的是正数。

#### 典型场景

参考《RetailTradeSITTest3.xlsx》。这个文档实现了一种典型场景，其中有零售跨库、退货跨库。

* 所谓零售跨库，是指零售单中的商品数量较大，大到跨越了至少2次入库。同一商品，每次入库的进货价可能不同，跨库可能会导致进货价计算不是：

进货价X商品数量

而是：

进货价1X商品数量1+进货价2X商品数量2+…+进货价N X 数量N

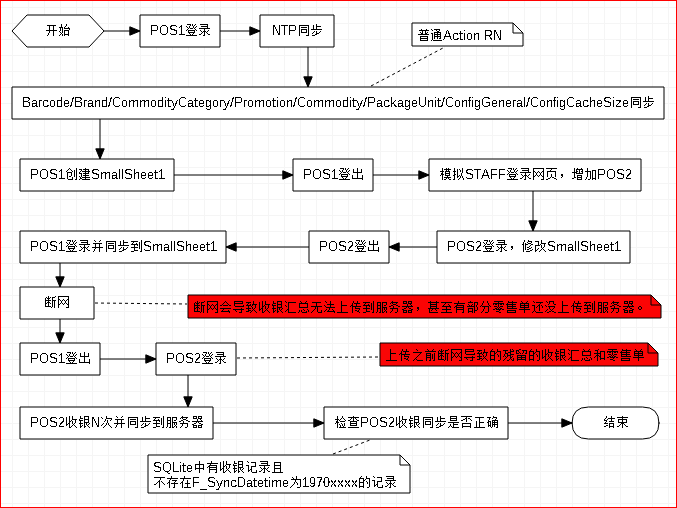
* 退货跨库，是零售跨库的逆过程，就是顾客退货的数量，跨越了至少2次入库，应该将正确的退货价返回给顾客，同时在商品表中恢复正确的当值入库ID。

#### 更多论述

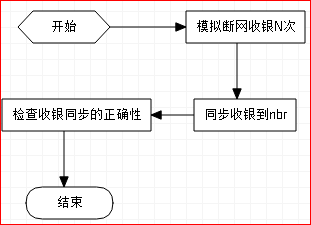
参看[报表](#_报表)部分。

### Android POS开机SIT测试设计

第1个测试的流程：

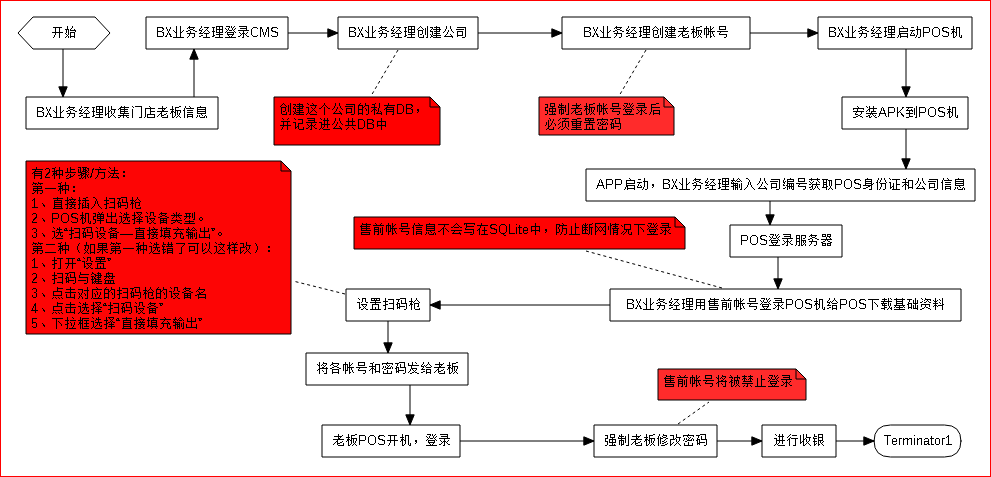


第1个运行完后，第2个测试运行的流程：



### 系统交付流程

#### 综述



#### POS机交付流程

POS机交付流程的具体流程是：

1. BX业务经理拆开POS机包装并开机。
2. BX业务经理查看POS SN（POS SN和公司SN不同）。
3. BX业务经理创建公司。默认会创建1个门店和1个售前帐号。
4. BX业务经理创建这个门店的POS。其中POS的SN是重要的一个字段。
5. BX业务经理在POS机上安装博销宝APP。
6. BX业务经理启动APP。它将读取配置文件，确定NBR地址。
7. APP检查SQLite里是否有公司SN。
   * 如果没有（一般是BX业务经理初次登录APP），将显示文本框给业务经理填写。BX业务经理在APP输入公司编号，点击获取POS身份证（关键是登录NBR的明文密码）和公司信息。向服务器成功获取到数据后，写进本地SQLite中。下次登录时检测到它们存在，就不会再向服务器请求。下次登录检测到SN存在，会直接进入LoginActivity，先触发POS登录。此时POS登录可以使用SQLite中的明文密码。POS登录成功后，得到SessionID。Staff登录将使用此SessionID来登录。POS和STAFF共用一个SessionID。SessionID为null时，表明没有网络。网络请求出错后，也会将SessionID设置为null。
   * 如果有（只能在交付后才有），只显示用户名和密码框给收银员填写。公司编号会显示在屏幕右下角。
8. 业务经理使用售前帐号登录POS机，进行基础资料下载。这个肯定**不会**让老板自己在家做。需要注意的是：售前帐号会到NBR端认证，但不会写到SQLite当中。这是为了防止售前帐号在断网的情况下，仍然能登录这台POS。
9. 设置好扫码枪。如果不设置，在支付界面会出现扫码后没有得到条形码的错误。
10. 封装POS机，发货。
11. 老板收到POS机后，登录，修改密码。修改密码时（包括以后再修改密码）如果检测到这个帐号的角色是老板（店长），则禁止售前帐号以后登录POS。

以后可能还设置其它数据。待续。

### 微信消息生成和处理流程

#### 关注公众号后绑定老板信息

公众号端及相应的后端流程：

* 老板在微信上关注公众号。
* 老板在公众号添加一个二级菜单做信息绑定，菜单名称为：？？？？。
* 老板填写公司编号、帐号和密码。
* 提交。后端：如果登录成功，表明此人是门店老板，将其openid更新到xxx.T\_Staff中。此后，此门店的微信消息直接发送到这个openid。更新T\_Staff缓存和视图缓存。
* 显示绑定成功消息。



#### 取消关注公众号

* 老板在微信上取消关注公众号。后端：搜索T\_Staff，将其openid设置为null。更新T\_Staff缓存和视图缓存。

#### 采购订单已生成、待审核消息

消息示例：

|  |  |
| --- | --- |
| 标题： | 采购订单提醒 |
|  |  |
| 内容示例： | 你好，有采购订单生成需要审核 |
|  | 采购单号：CG20190712XXXX |
|  | 创建时间：20190712 |
|  | 单击查看详情 |

附URL到页面审核采购订单

保存cookie信息，一周内不用重复登录，否则需要重新登录？？？（是否设置更长时间）

Jsp页面要求：

* 供应商信息；
* 单号
* 采购单详情列表
* 底部审核按钮

模板设置：



#### 采购超时消息

描述/规则：订单已经下但是供应产在规定时间内未送货到。超出送货期限后，一日提醒一次。如果后面不再送货怎么办？？？？

附URL，只显示采购超时的采购订单号，供应商等信息

示例：

|  |  |
| --- | --- |
| 标题 | 采购超时提醒 |
|  |  |
| 内容示例： | 你好，你有一个采购订单超过1天未入库 |
|  | 采购订单：CG201907120005 |
|  | 供应商：华南供应商 |
|  | 创建时间：2019年7月12日 |
|  | 请尽快前往NBR处理 |

10天内未入库，从第11天开始，每天提醒一次

模板设置：



#### 入库价格变动消息

描述：入库价格与采购订单上的价格不符，知会老板。

模板设置：



示例：

|  |  |
| --- | --- |
| 标题： | 入库价与采购价不符 |
|  |  |
| 内容示例： | 你好，该单据入库价与采购价不符 |
|  | 采购单号：CG201907120002 |
|  | 入库单号：RK201907120002 |
|  | 操作人：店员2号 |
|  | 请前往NBR查看具体信息· |

不附URL，单纯提醒。

#### 保质期商品提醒消息

暂时不做。

#### 盘点差异报告消息

描述：每次盘点结束后，发送消息知会老板盘点差异报告。

模板设置：



模板示例：

|  |  |
| --- | --- |
| 标题 | 盘点结果提醒 |
|  |  |
| 内容示例： | 你好，盘点详情如下： |
|  | 盘点单号：PD201907120001 |
|  | 偏差商品数量：0 |
|  | 请前往NBR查看具体信息 |

不附URL，单纯提醒。

#### 登录上班消息

描述：门店人员登录上班后发送消息知会老板。

模板设置：



不附URL，单纯提醒

老板，员工登录都要发送信息

#### 交班消息

描述：交班时，发送消息告诉老板今天的营业情况，具体包括店铝、销售额、客单数。

模板设置：



#### 商品滞销消息

描述：商品久未售出，发消息知会老板。滞销商品每月发送一次。模板设置：



示例：

|  |  |
| --- | --- |
| 标题 | 商品滞销提醒 |
|  |  |
| 内容示例： | 你好，最近一个月内有以下商品滞销 |
|  | 商品种类数：10 |
|  | 检查周期：30天 |
|  | 单击查看详情 |

附URL查看滞销商品信息

Jsp页面要求：滞销商品信息列表。

### 公共DB和私有DB设计

最终策略和架构演进方面：

1、功能代码方面：

决定从ACTION层决定采用哪个DB作为数据源（通过拦截器），BO层设AOP拦截以设置该BO要使用的数据源。

原因是所有的操作，都是从ACTION层发起的，除了夜间任务以外。

夜间任务方面，需要修改接口令线程知道当前执行的DB是哪个公司的。

2、测试代码方面：

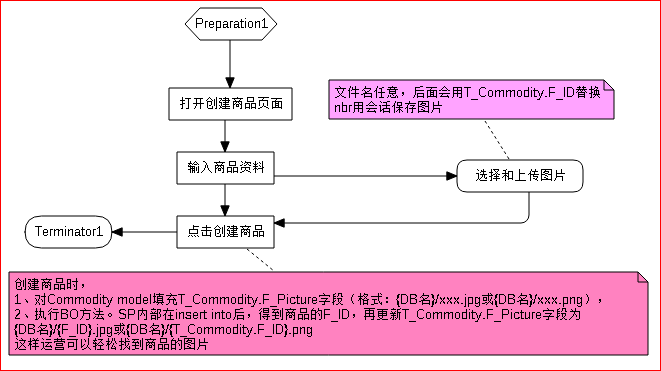
许多时候MapperTest需要指定公共或私有的数据源，这时，设计新的AOP专门针对测试代码。当然也可以在Mapper层设置AOP，靠传递的DB名决定用哪个数据源。

ACTIONTEST应该可以不用设置什么都能正常工作。

3、xxxCacheManager也需要传递DB名。

4、因ACTION和Mapper都有DB名，前面的AOP有可能会被后面的覆盖，所以不采用默认数据源。之前设计为默认采用公共DB，需要重新考虑。

### 上传商品图片



### 上传公司营业执照

主要步骤和[上传商品图片](#_上传商品图片)相似，但是营业执照的命名格式是{公司DB名称}.jpg或{公司DB名称}.png。T\_Commpany里保存的也是这个名称。

### 上传微信退款证书和退款授权

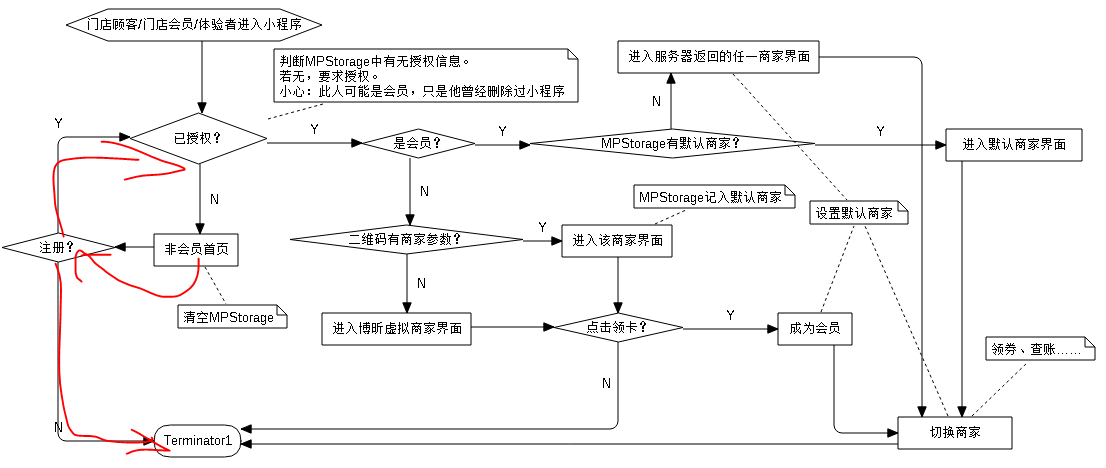
* 服务商，如博昕，必须上传微信退款证书到服务器，才能退款。
* 服务商为子商户（如门店）提交资料，申请帐号。服务器商发送授权邀请给子商户，子商户在微信平台确认授权后，服务商才能为子商户退款。
* 如果服务商没有得到子商户的授权，则必须让子商户上传退款证书到服务器才能为子商户正常退款。

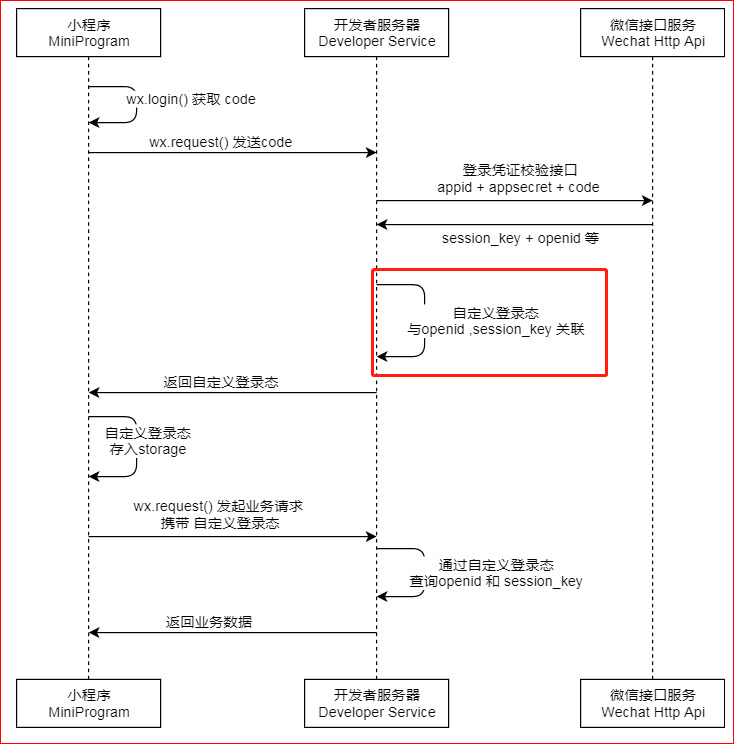
### 会话管理

#### 用户会话

* 限制OP帐号和门店帐号登录于一个浏览器中。OP登录成功后，清除门店帐号的会话。门店帐号登录成功后，清除OP会话。
* 同一帐号可以同时在POS和电脑网页登录，但不能同时在不同POS登录，也不能同时在不同的浏览器上登录。否则将踢出之前登录的帐号。在POS端，这将导致上传零售单和收银汇总无法成功，除非重新在本机登录。
* 会员只能在小程序登录，其会话是会员会话，不能做其它未授权的操作。

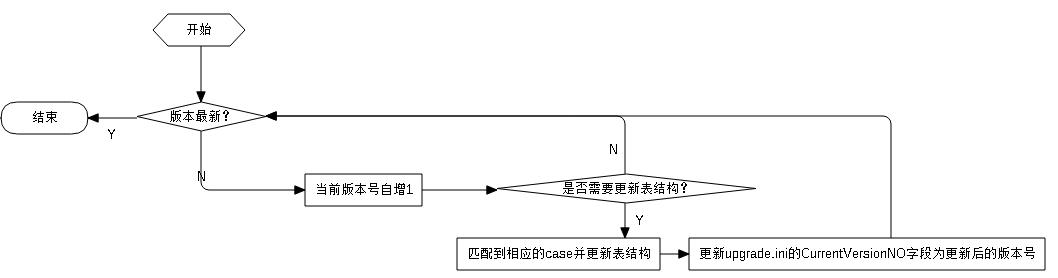
### 小程序上会员/顾客登录的流程





### wpos.exe的启动流程

WPOS在开发和部署后的运行过程不同，在部署到客户机器后，其流程如下：



其中，与wpos.exe同一个目录的upgrade.ini的内容初步设计为：

[CurrentVersion]

### 当前wpos的版本号

CurrentVersionSN = 2.0.9

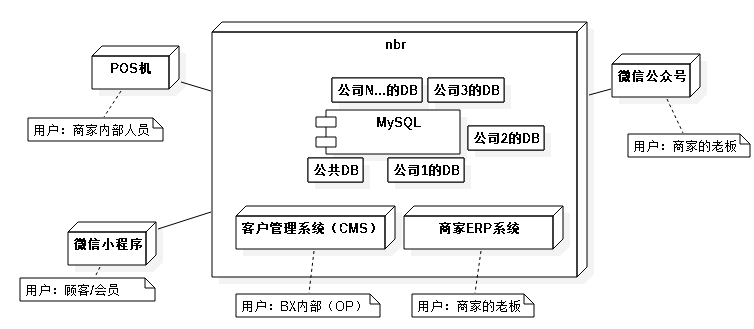
CurrentVersionNO = 9

注意：

wpos固定安装在D:/wpos/。

## 结构

### 系统逻辑结构



### 系统模块设计

#### 瘦系统模块设计

##### 博昕客户模块

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序 | 后端程序 |
| 身份识别 | * POS机身份识别 |  |  |
|  |  |  |  |
|  |  |  |  |

##### 用户首页模块

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序 | 后端程序 |
| 消息提醒 | * 用户登录成功后看到消息提醒 | Dashboard.jsp | Dashboard.java  Dashboard Action.java  Dashboard BO.java  Dashboard Mapper.java  Dashboard Mapper.XML  Dashboard Validation.java |
|  |  |  |  |

##### 消息模块

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序 | 后端程序 |
| 消息 | * 生成消息（由其它N个模块各自生成各自类型的消息并提供相应的处理界面） * 显示消息类别列表 * 获取消息列表（默认显示未读消息），分页，分类显示。显示未读消息总数 * 获取消息详情，标记为已读 * 将消息标记为未读 * 用户登录成功后看到消息提醒（见[用户首页模块](#_用户首页模块)） * 点击消息中的链接可以跳到处理消息的地方。比如商品采购建议类型的消息，会跳到相应模块处理 | Message.jsp | Message.java  MessageAction.java  MessageBO.java  MessageMapper.java  MessageMapper.XML  MessageValidation.java |

##### 商品模块

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序 | 后端程序 |
| 管理商品 | * 获取商品类别列表 * 获取供应商列表 * 所有商品列表及其分页 * 快速查询某种商品类别下的所有商品，分页 * 快速查询某个供应商下的所有商品，分页 * 根据商品的一个或多个属性查询商品档案，分页 * 新增商品档案（[点击参考](#_2.3.1_创建商品)）。处理一品多码、一码多品、一品多包装 * 编辑商品档案 * 删除商品档案 | Commodity.jsp | Commodity.java  CommodityAction.java  CommodityBO.java  CommodityMapper.java  CommodityMapper.xml  CommodityValidation.java |
| ~~采购建议~~ | * ~~系统扫描库存（商品库存减少触发扫描或系统定期扫描），生成采购建议的消息及采购计划表。采购计划表的商品的流程[如图](#_2.3.2_商品采购)所示。（PS：用户登录系统后，显示消息提醒。其中包括采购建议的消息。）参见[定时任务模块](#_定时任务模块)。~~   ~~另一种生成原因：POS机收银，服务器库存为0，触发生成该商品采购计划表及消息。~~   * ~~查询采购建议列表。~~ * ~~查询采购建议详情。（相应的消息默认被系统标为已读。）~~ * ~~用户从采购建议中编辑供应商。如果没有供应商，新增一个供应商。~~ * ~~用户从采购建议中增加或删除商品。~~ * ~~生成未审核订单或合并所有未审核的订单（后者暂不做）。这不会修改采购建议的数据。~~ | ~~PurchasingSuggestion.jsp~~ | ~~PurchasingSuggestion.java~~  ~~PurchasingSuggestionAction.java~~  ~~PurchasingSuggestionBO.java~~  ~~PurchasingSuggestionMapper.java~~  ~~PurchasingSuggestionMapper.xml~~  ~~PurchasingSuggestionValidation.java~~  ~~另考虑供应商相应的类等~~ |
| 采购订单 | * 采购订单列表。默认显示最近的未通过审核的采购订单详情 * 显示采购订单详情 * 编辑采购订单 * 审核通过采购订单 * 查询采购订单 * 删除采购订单 | PurchasingOrder.jsp | PurchasingOrder.java  PurchasingOrderAction.java  PurchasingOrderBO.java  PurchasingOrderMapper.java  PurchasingOrderMapper.xml  PurchasingOrderValidation.java |
| 入库 | 参见[入库模块](#_入库模块) |  |  |
| 商品缓存 |  |  |  |

##### 促销模块

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序(Android) | 后端程序 |
| 创建促销 | * 检查创建促销的业务规则。其规则包括：   + 规则一：创建促销时，禁止同时对商品A重复搞特价。   + 规则二：满减、满折扣二选一，不能同时进行。另外也不能满减后再满减，或满折扣后再满折扣 |  |  |
| 促销优惠计算 | * 计算逻辑实现。计算一个零售单经过满减、特价等促销活动后，顾客需要支付的费用。 * 计算日志呈现（开发人员部分）。用于查证、debug。 * 计算日志呈现（BA部分）。计算过程需要生成BA能看懂的文档以便考核开发质量。 |  |  |
| 促销同步信息 | * 参看[同步模块](#_同步模块_1)。 |  |  |
| 促销缓存 | * 服务器启动加载促销缓存 * 更新促销缓存 * 读取缓存的copy并返回给POS机 |  |  |
|  |  |  |  |

##### 收银模块

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序(Android) | 后端程序 |
| 收银 | * 登录收银机 * （前端功能）编辑待售商品列表 * （前端功能）增减待售商品。 * 结算并给出计算优惠值和优惠步骤。触发采购计划表及其消息。必须考虑改价、普通折扣、整单折扣、赠送、促销、会员。另外考虑备注和销售开单同步（见下面一点） * （前端功能）改价 * （前端功能）普通折扣 * （前端功能）整单折扣 * 列出所有商品，可分类。 * 赠送模式。若采用此模式，商品免费。 * （前端功能）挂单 * （前端功能）取消挂单？ * （前端功能）整单取消 * [打印小票和小票格式设置](#_小票打印与小票设置)。 * （前端功能）开单备注 * （前端功能）（后端检测）收银限制检测 * （前端功能）离线收银（和在线收银区别不大） * 销售开单同步。参见[同步模块](#_同步模块)。 | SaleActivity.java | Sale.java  SaleAction.java  SaleBO.java  SaleMapper.java  SaleMapper.xml  SaleValidation.java |
| 库存 | * 退货.。 * 换货。不作特别处理。 * 查询库存。 | WarehouseActivity.java | Warehouse.java  WarehouseAction.java  WarehouseBO.java  WarehouseMapper.java  WarehouseMapper.xml  WarehouseValidation.java |
| 会员 | * 获取会员基本信息 * 退订金给会员（不做） | VIPActivity.java | VIP.java  VIPAction.java  VIPBO.java  VIPMapper.java  VIPMapper.xml  VIPValidation.java |
| 门店人员 | * 显示收银员基本信息 * （前端功能）锁定帐户（班中离机） * 营业员查询 | StaffActivity.java | Staff.java  StaffAction.java  StaffBO.java  StaffMapper.java  StaffMapper.xml  StaffValidation.java |
|  |  |  |  |

##### 盘点模块

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序 | 后端程序 |
| 盘点 | * 新建盘点单 * 查询盘点单 * 填写盘点单（包括编辑盘点单信息和商品列表） * 提交盘点单 * 查询差异报告 * 审批盘点单 | Inventory.jsp | Inventory.java  InventoryAction.java  InventoryBO.java  InventoryMapper.java  InventoryMapper.xml  InventoryValidation.java |
|  |  |  |  |

##### 仓管模块

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序 | 后端程序 |
| 入库 | * 新建验收入库单 * 引用采购订单(已审核而未全部入库) * 修改入库单（入库单基本信息和商品信息） * 查询已有的入库单 | Warehousing.jsp | Warehousing.java  WarehousingAction.java  WarehousingBO.java  WarehousingMapper.java  WarehousingMapper.xml  WarehousingValidation.java |
| 采购退货 |  |  |  |

##### 同步模块

###### 上行同步

同步销售开单数据到远程服务器。

###### 下行同步

下载商品档案等其它数据同步到本地POS­­机。

互参：[数据同步机制设计](#_2.3.7_数据同步机制设计)。

###### 服务器端同步块的缓存

服务器上的同步信息是需要频繁访问的资源，必须长储于内存中，接受读写，最大限度地提高访问的效率。

当服务器关闭时，这些信息必须已经存储于DB中，以便启动时加载到内存中，再让其它客户端下载。

本模块负责管理同步信息的缓存。

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序 | 后端程序 |
| 同步缓存 | * 服务器启动。服务器启动时，加载要同步的信息到内存中。 * POS上传。服务器收到POS上传的同步信息时，   + 将其更新到DB中。   + 将所有POS机置为未同步。   + 刷新已经加载到内存的缓存。 * POS下载。POS请求下载->服务器将一个要同步的内存块发给POS->POS报告内存块已经同步成功->服务器标记此POS已经同步了该内存块->如果所有POS都同步了该内存块，则从DB及缓存中都删除这块内存块。 |  |  |
| 同步缓存代理（即Controller） | * 基于http协议和JSON格式，接收POS的上传同步请求和下载同步请求。 |  |  |
|  |  |  |  |

##### 缓存模块

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序 | 后端程序 |
| 缓存 | * 服务器启动。服务器启动时，加载要频繁访问的信息在内存中。 * 访问内存时，若没有击中，则从DB中加载到缓存中。如果击中，直接取出。 * 如果目标数据被其它请求更新，缓存应该被刷新 |  |  |
| 缓存管理器 | * 注册各种缓存信息，比如促销信息、商品信息 * 获取各种缓存信息 |  |  |

##### 定时任务模块

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序 | 后端程序 |
| 采购扫描 | * 定时（一般是在下半夜）扫描库存，生成采购建议的消息及采购计划表。互参：[商品模块](#_商品模块)；[消息表](#_5.1.8_消息)。 * 。 |  | NightJob.java |

##### 报表模块

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序 | 后端程序 |
|  |  |  |  |

##### 设置模块

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名 | 模块职责与系统功能需求 | 前端程序 | 后端程序 |
|  |  |  | NightJob.java |
|  |  |  |  |
| 上传销售开单间隔 |  |  |  |

DB设计参考：[5.1.17 全局配置表（配置性数据）](#_5.1.17_全局配置表（配置性数据）)。

#### 系统公共控件设计

搜索框

新增与编辑Tab

## 功能需求与程序的关系

财务对账功能在程序中没有实现。

## 人工处理过程

员工忘记密码

数据同步总是失败。

## 自动化测试设计

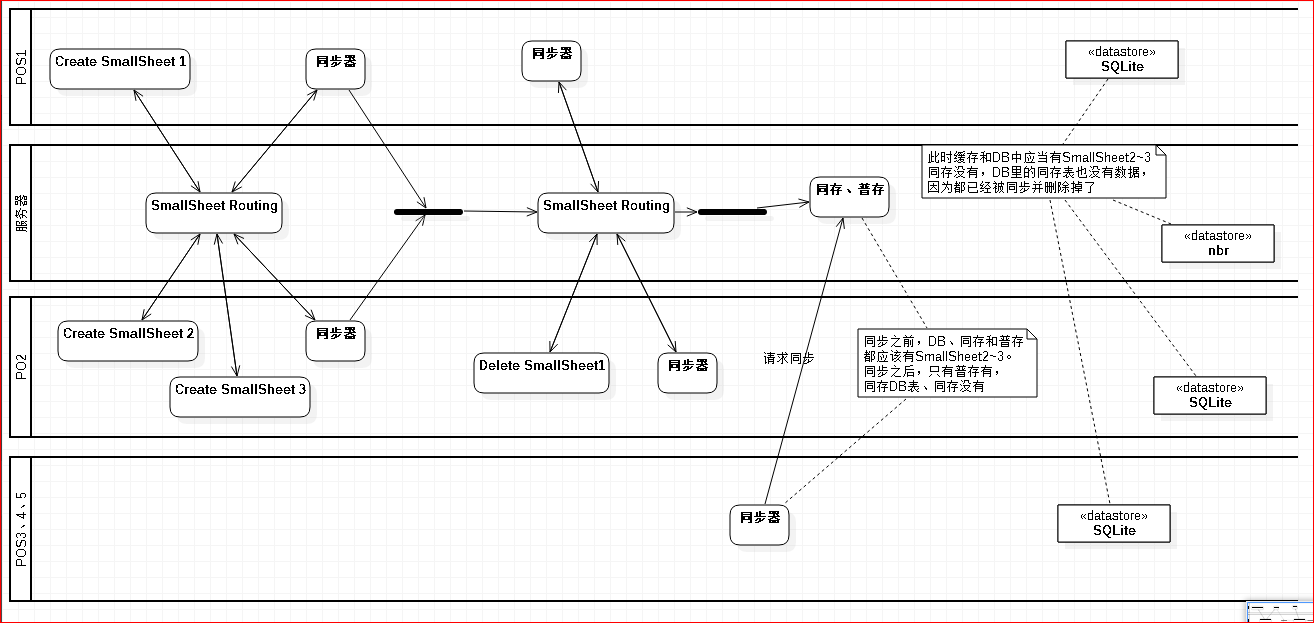
### 销售开单价格计算

### 数据同步

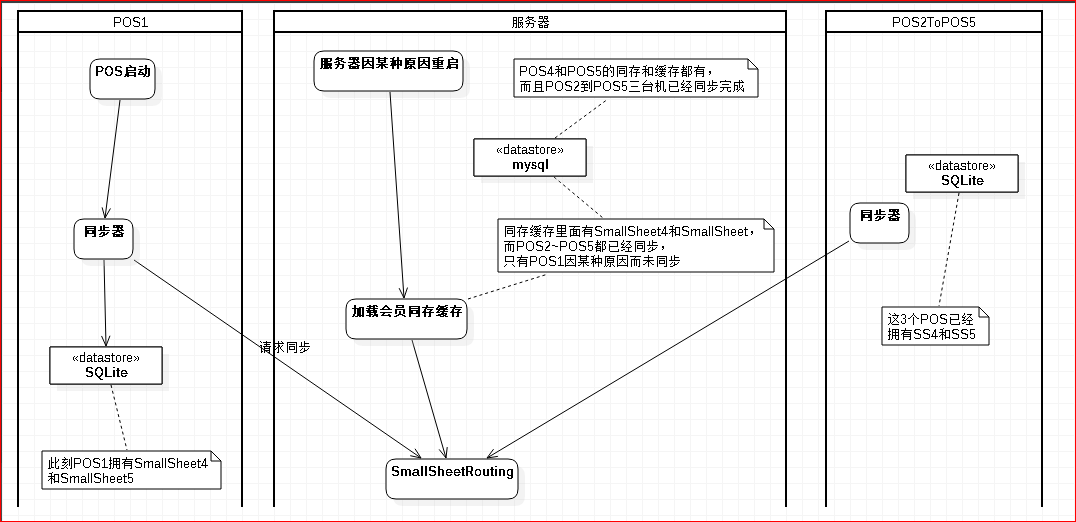
部分情况：需要结合版本更新合并测试。

#### 2.7.3.1 小票格式

小票格式同步SIT测试1

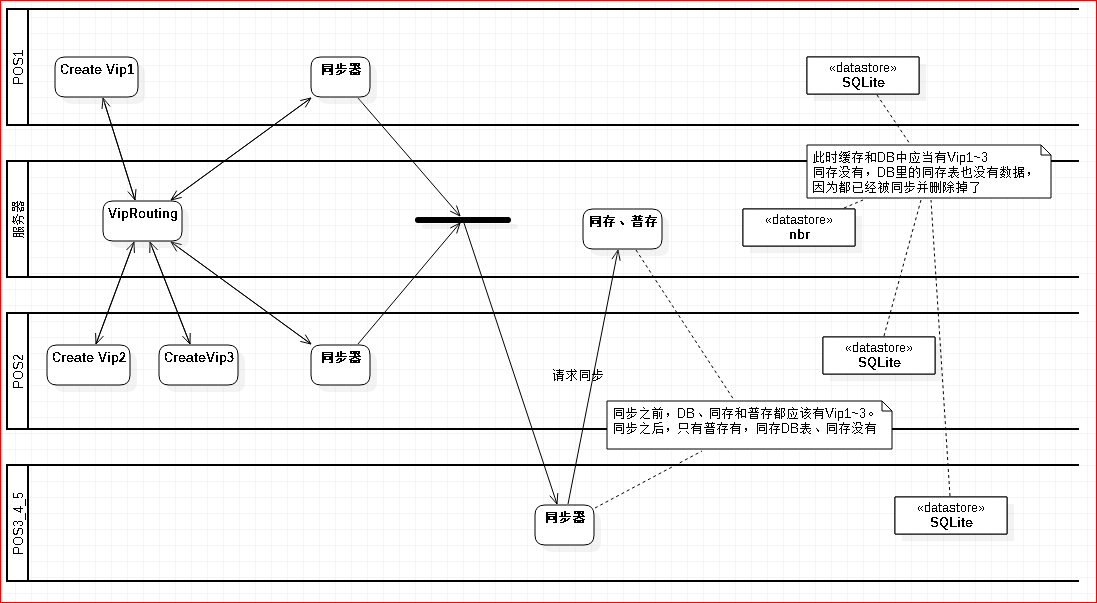


小票格式同步SIT测试2

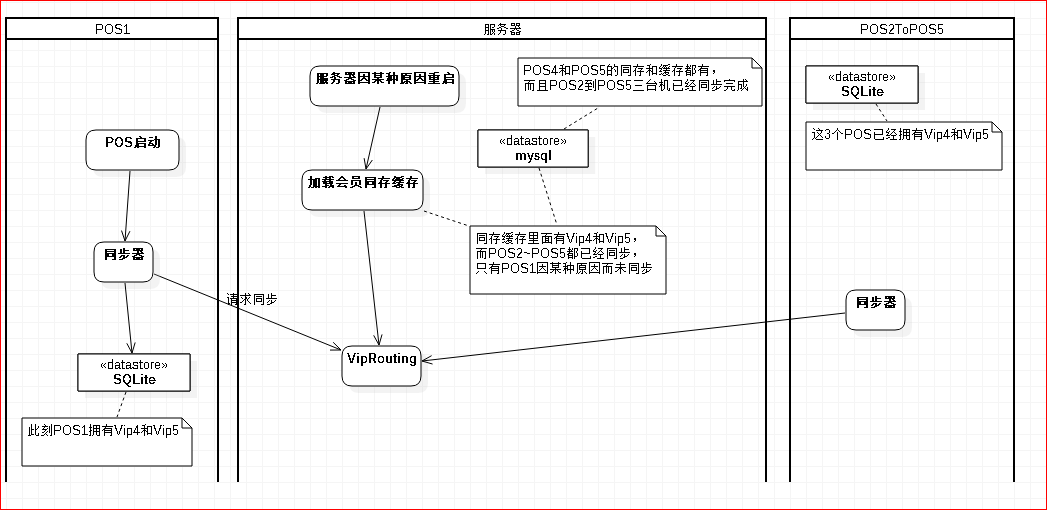


#### 2.7.3.2 VIP

SIT1：



SIT2：



### xxxxxxxxx

## 版本升级设计

见[部署备忘](#_2.10_部署备忘)。

## 尚未解决的问题

## 部署备忘

### DB部署

部署任何与DB相关的新版本时，要注意是否影响同步。具体要检查的点是：

* 数据表结构有无被更改。
* 字段有无被更改。
* 键值约束有无变化。
* 外键约束的值有无冲突。

### POS机身份识别问题

POS机身份识别问题。

### 基础资料在POS中的初始化

基础资料全部塞进POS。

### Mysql

* Mysql
  + 测试Mys ql的连接是否可以保持超过8小时，如不能，令其能。
  + 令MYSQL不能接收本机以外的IP的请求。DEV、SIT、UAT可以创建只能读的用户，并且打开远程访问的功能。

### Tomcat配置

* Tomcat配置**AES算法加密**。
  + 需要做(JDK8)：<ftp://119.23.160.192/Install/%CF%EE%C4%BF%B2%BF%CA%F0/jce_policy-8.zip>。下载后解压，可以看到local\_policy.jar和US\_export\_policy.jar以及readme.txt。 如果安装了JRE，将两个jar文件放到%JRE\_HOME%\lib\security下覆盖原来文件，记得先备份。 如果安装了JDK，将两个jar文件也放到%JDK\_HOME%\jre\lib\security下。
  + 原因：由于信息安全在军事等方面极其重要，如在第二次世界大战期间，使用了无线电，若是能够成功解密敌方的机密情报，往往预示着战争的胜利，因此美国对加密解密等软件进行了出口限制，JDK中默认加密的密钥长度较短，加密强度较低，而UnlimitedJCEPolicyJDK7中的文件则没有这样的限制，因此为了获得更好的加密强度，需要替换掉那两个文件。
  + JDK7的下载地址: http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.htmlJDK8的下载地址: http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html
* 商品图片映射路径设置。
* 公司营业执照映射路径设置。
* 商家支付证书路径设置。

### Log4j配置

* Log4j配置
  + 日志级别。
  + 报警邮件地址改为相应的环境的地址，比如SIT的不要用DEV的。
  + 打开邮件通知。

### Jenkins健康检查

* Jenkins健康检查

### 皇后服务

* Windows服务配置

在产品场中创建2个公司，1个用来复现客户出现的问题（通过DB复制迅速取得用户的数据），另1个用来给BX内部模拟使用。

* 配置运行小王子时，是否删除私有DB的数据。

在Production我们永远不会运行小王子Jenkins版，但是如果有人不小心运行了就可能出大事。

Helper.ini里设置以下项为1会删除所有私有DB的数据：

DeletePrivateDBs=1

在Production我们应该将其设置为0。

### 微信公众号配置

@R&D Manager\_Giggs 微信公众号更换测试场需要修改的内容：

1、服务器地址（URL）

2、Token(令牌)

3、AppSecret

4、IP白名单

5、消息加解密密钥(EncodingAESKey)

### HouseKeepService

安装配置此服务以令日志不会爆棚。

在UAT/Prod上，可能不需要自动清除这些日志，要注意。

# 接口设计

## 用户接口

### 界面设计

注意：以下所有页面的列表，只要有较多行数据，都提供分页功能。

### 微信小程序界面

## 外部接口

条码扫描仪

打印机

一体机

手机

## 内部接口

### 小票打印与小票设置

？？？？？？？？？

详情查看：

https://docs.sunmi.com/htmls/index.html?lang=zh##打印和客显

需要实地试验打印和打印的格式，然后进行格式设置这一功能的设计。

### 网页跳转

页面与struts的接口。

### Android界面跳转

模块间的接口，主要是调用接口。

类与类之间的接口

### 数据库存储过程（SP）

### http请求与响应

# 运行设计

## 运行模块组合

网页＋微信小程序界面

## 运行控制

### 配置

系统提供一系列默认设置，令系统运行。用户可以修改这些设置以控制系统的默认行为。

默认设置如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 子系统/模块 | 数据项 | 值 | 系统行为 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

### 服务器运行过程

#### 启动

加载不同版本的客户端的配置性数据和业务逻辑。

#### 运行中

##### 关键的会话

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 会话名称 | Key | 类型 | Value | 描述 |
| POS机登录的会话 | EnumSession.SESSION\_POS.getName() | String | POS对象 | 存储已经登录成功的POS的信息 |
| 登录的staff的会话 | EnumSession.SESSION\_Staff.getName() | String | Staff对象 | 存储已经登录成功的staff的信息。其特殊性在于，如果是在POS机上登录，POS机登录的会话和登录的STAFF的会话是一致的。POS机的使用流程一般为：   1. POS机登录NBR，得到会话A。 2. 收银员输入用户名和密码，登录NBR。这时，使用的会话是会话A而不是一个新的会话！ 3. 登录成功后开始收银过程… |
| 登录的BXSTAFF的会话 | EnumSession.SESSION\_BxStaff.getName() | String | BXStaff对象 | 存储已经登录成功的bxStaff的信息。他的权限和staff的权限不同，不能混淆。但特殊的地方在于：BXSTAFF（业务经理）可以创建公司，以及创建这个公司的默认门店（最多只能创建第1个）和售前帐号（最多只能创建1个）。 |
|  |  |  |  |  |

##### 权限控制

#### 关闭

### POS机运行过程

#### 售前准备

* 人
  + 创建博昕客户。
  + 定义权限、角色。
  + 创建员工，包括老板。
* 场
  + 门店。
  + POS机载入人和货的资料。
* 货
  + 添加供应商、品牌、商品类别、商品单位、商品档案。
* 相关的配置表初始化

系统初始化时，以下项需要默认添加：

仓库：默认仓库

采购：默认供应商

品牌：默认品牌

支付方式：现金

#### 启动

检查升级

读取配置

时间同步

其它档案同步

首次启动：初始状态，全部下载配置和档案。

账号每次登录需要填写准备金

#### 运行中

正常操作，

一天销售单大概180-230单

#### 关闭

## 运行时间

# 系统数据结构设计

## 逻辑结构设计要点

以下每个数据项都有相应的备份表，即在原表中做删除操作之前，会先回收在备份表中，然后才会真正删除原表中的数据。备份表中额外增加删除日期列和操作原因、操作页面等重要的相关信息。

此举系为所有操作提供恢复数据的机会和满足历史追溯的需要。

备份表参考格式：

表T\_BAK\_XXXX:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_BAK\_ID | INT NOT NULL |  | PRIMARY KEY，自增1 |
| ……其它各列被删除的数据 | | | | |
| 删除日期 | F\_DateDelete | DATETIME NOT NULL |  |  |
| 操作界面 | F\_Page | VARCHAR(30) NOT NULL |  |  |
| 操作原因 | F\_Cause | VARCHAR(30) NOT NULL |  |  |

### 综述

本系统的DB分为公共DB和私有DB。

前者存储BX内部员工的数据，私有DB的索引数据。后者是门店老板的门店数据。

### 商品

#### 参考商品库T\_RefCommodityHub

在新增商品界面，将以下参考库的所有字段的值搬到新增商品界面的相应字段。

如果已经有的商品当中不存在某些值，比如商品品牌，则新建一个这样的商品品牌。

其它情况，如果字段不能够匹配或不能照搬，则在网页上使用默认值，让用户自己选择。比如商品品牌，定为默认品牌，让用户自己选择。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 商品ID | F\_ID | INT NOT NULL |  | PRIMARY KEY，自增1。  针对某公司的某个商品的惟一标识。该公司可多门店 |
| 条形码 | F\_Barcode | 仅允许英文、数值形式出现，字符数暂不设上限  长度(0, 20] |  |  |
| 商品名称 | F\_Name | 允许以中英数值、空格形式出现，不允许仅使用空格（不同的商品，商品名字允许重复），长度(0, 32] |  |  |
| 商品简称 | F\_ShortName | 允许以中英数值、空格形式出现，不允许仅使用空格（不同的商品，商品名字允许重复）  长度(0, 32] |  |  |
| 规格 | F\_Specification | 允许以中英数值、空格形式出现，不允许仅使用空格  长度(0, 8] |  | 毫升之类 |
| 包装单位名称 | F\_PackageUnitName | 允许以中英数值、空格形式出现，不允许仅使用空格  长度(0, 8] |  |  |
| 采购单位 | F\_PurchasingUnit | 允许以中英数值、空格形式出现，不允许仅使用空格  长度(0, 16] | 无 | 采购N支N箱N包。采购时才设置 |
| 品牌名称 | F\_BrandName | (0, 20] 个中英文、数字，无空格 |  | 如果T\_Commodity中没有，则采用默认品牌 |
| 类别名称 | F\_CategoryName | (0, 10]个中英文、数字，无空格 |  | 如果T\_Commodity中没有，则采用默认类别 |
| 助记码 | F\_MnemonicCode | 允许以英、数值形式出现  长度[0, 32] |  | Y、YB＝怡宝 |
| 计价方式 | F\_PricingType | {0, 1} | 1 | 0=称重、1=计件 |
| ~~是否服务器类商品~~ | ~~F\_IsServiceType~~ | ~~{0, 1}~~ | ~~0~~ | ~~1是，0否。如收快递~~  ~~本字段已经删除，但Java层未需要修改~~ |
| 商品类型 | F\_Type | {0, 2} | 0 | 0=普通商品。1=组合型商品（一种组合型商品包括几种普通商品，打包出售）。2=多包装商品 |
| 进货价（采购价） | F\_PricePurchase | 非负浮点数 |  | 一直以来进货价加权平均得到的进货价，用于计算毛利等 |
| 最近进货价（最近采购价） | F\_LatestPricePurchase | 非负浮点数 | -1 | 最近进货的价格 |
| 零售价 | F\_PriceRetail | 非负浮点数 |  |  |
| 会员价 | F\_PriceVIP | 非负浮点数 |  |  |
| 批发价 | F\_PriceWholesale | 非负浮点数 |  |  |
| 毛利率 | F\_RatioGrossMargin | 非负浮点数 | 0.0 | 系统自动计算 |
| 保质期 | F\_ShelfLife | Int, 有效天数，自然数 |  |  |
| 退货天数 | F\_ReturnDays | 非负整数 | 0 | 0=不可退货。  其它正数N=N天内可以退货 |
| ~~是否赠送商品？？？~~ | ~~F\_IsGift~~ | ~~{0, 1}~~ |  | ~~本字段已经删除，但Java层未需要修改~~ |

#### 商品T\_Commodity

重要规则：

1. 默认多包装商品只有一个条形码。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 商品ID | F\_ID | INT NOT NULL |  | PRIMARY KEY，自增1。  针对某公司的某个商品的惟一标识。该公司可多门店 |
| 商品状态 | F\_Status | INT NOT NULL | 0 | 1=预淘汰商品，0=非预淘汰商品。生成采购建议时，如果此值为1，一定不考虑采购。  2=已删除。代表该商品已经删除。商品之所以有2这个状态，是因为它曾经参与零售、采购等行为，形成了历史记录。我们的产品需要随时查询得到这种历史记录 |
| 商品名称 | F\_Name | 允许以()（）-—\_、中英数值、空格形式出现，只允许中间使用空格（商品名字不允许重复），长度(0, 32] |  | ()（）-—\_分别是英文小括号、中文小括号、英文破折号、中文破折号、英文下划线。  商品名字不允许重复指的是不在删除状态的商品的名称不能重复。已经删除的商品的名称可以和没删除的商品的名称重复。 |
| 商品简称 | F\_ShortName | 允许以中英数值、空格形式出现，只允许中间使用空格（商品名字不允许重复）  长度(0, 32] |  |  |
| 规格 | F\_Specification | 允许以中英数值、空格形式出现，不允许仅使用空格  长度(0, 8] |  | 毫升之类 |
| 包装单位 | F\_PackageUnitID | 自然数，可NULL |  | 外键，对应T\_PackageUnit(F\_ID)。=NULL时，表明本商品已经在删除状态 |
| 采购单位 | F\_PurchasingUnit | 允许以中英数值、空格形式出现，不允许仅使用空格  长度(0, 16] | 空 | 采购N支N箱N包。采购时才设置 |
| ~~供应商ID~~ | ~~F\_ProviderID~~ | ~~数字，可NULL~~ |  | ~~FOREIGN KEY。指向供应商表的F\_ID。=NULL时，表明本商品已经在删除状态~~ |
| 品牌ID | F\_BrandID | 正数字，可NULL |  | FOREIGN KEY。指向商品品牌表的F\_ID。=NULL时，表明本商品已经在删除状态 |
| 类别ID | F\_CategortyID | 正整数，可NULL |  | FOREIGN KEY。指向商品类别表的F\_ID。=NULL时，表明本商品已经在删除状态 |
| 助记码 | F\_MnemonicCode | 允许以英、数值形式出现  长度[0, 32] |  | Y、YB＝怡宝 |
| 计价方式 | F\_PricingType | {0, 1} | 1 | 0=称重、1=计件 |
| ~~是否服务器类商品~~ | ~~F\_IsServiceType~~ | ~~{0, 1}~~ | ~~0~~ | ~~1是，0否。如收快递，它们进货价永远为0。库存是多少都无所谓，修改为多少都无所谓，将来可以考虑令其永远为0。~~ |
| 商品类型 | F\_Type | {0, 3} | 0 | 0=普通商品。1=组合型商品（一种组合型商品包括几种普通商品，打包出售）。2=多包装商品。3=服务型商品 |
| ~~进货价（采购价）~~ | ~~F\_PricePurchase~~ | ~~非负浮点数~~ |  | ~~一直以来进货价加权平均得到的进货价，用于计算毛利等~~  由于进货价是会变化的，所以应该保存到入库单中。不保存在本表中 |
| 最近进货价（最近采购价） | F\_LatestPricePurchase | 非负浮点数 | -1 | 最近进货的价格 |
| 零售价 | F\_PriceRetail | 非负浮点数，不能>2^31 |  | 由于它是会被用户不断修改的，所以交易后它会被保存到零售单中。但计算利润这些东西，不应该直接参考这个字段，而应该参考零售单。 |
| 会员价 | F\_PriceVIP | 非负浮点数，不能>2^31 |  |  |
| 批发价 | F\_PriceWholesale | 非负浮点数，不能>2^31 |  |  |
| ~~毛利率~~ | ~~F\_RatioGrossMargin~~ | ~~非负浮点数~~ | ~~0.0~~ | ~~系统自动计算~~ |
| 前台是否能改价 | F\_CanChangePrice | {0, 1} | 0 | 1是、0否。特殊商品，特殊人群 |
| 积分规则 | F\_RuleOfPoint | Int, 正整数 | 1 | 根据售价X倍数。  系统自动判断：根据售价决定 |
| 商品图片 | F\_Picture | Varchar(128)，长度[0,128] |  |  |
| 保质期 | F\_ShelfLife | Int, 有效天数，自然数 |  |  |
| 退货天数 | F\_ReturnDays | 非负整数 | 0 | 0=不可退货。  其它正数N=N天内可以退货 |
| 创建日期 | F\_CreateDate |  |  | 插入时，SP自动生成 |
| 采购阀值 | F\_PurchaseFlag | Int, 正整数 | 1 | 当商品的库存数目低于这个数目时，系统生成采购计划表 |
| 参照商品ID | F\_RefCommodityID | 非负整数 | 0 | 一品多包装。若无多包装，设置为0 |
| 参照商品倍数 | F\_RefCommodityMultiple | 非负整数 | 0 | 一品多包装。若无多包装，设置为0；否则为倍数。比如16支装，则为16。主要用于计算库存 |
| ~~是否赠送商品？？？~~ | ~~F\_IsGift~~ | ~~{0, 1}~~ |  | ~~暂时不考虑赠送商品~~ |
| Tag | F\_Tag | [0, 32] |  | 走葱/飞冰/XX/YY |
| 商品库存 | F\_NO | 整数 | 0 | 系统的数量，和盘点的实盘数量不同。人为原因可以令此数为负数。如果是负数，则仓库中本商品的库存总额也是负数。用户看到的本商品最大库存总额也是负数。  F\_RefCommodityID对应的商品叫做单品。单品存在且未删除的话，和组合商品一样，本商品（多包装商品）永远没有库存。真正用来计算库存的是单品。 |
| ~~商品累计入库数~~ | ~~F\_NOAccumulated~~ | ~~非负整数~~ | ~~0~~ | ~~每次商品入库都会将商品入库数量累加到这个字段。本字段用于每次入库后重新计算平均进货价，即F\_PricePurchase。~~  ~~只有入库，才会加到这里。其它任何情况都不加。~~ |
| 期初数量 | F\_NOStart | Int, 整数 | -1 | 新店内如果有存货，需要先将这些货的数量和采购价存入系统，以令后续报表计算正常。这2个字段禁止重复设置！ |
| 期初采购价 | F\_PurchasingPriceStart | 浮点数 | -1.000000 |
| 期初值备注 | F\_StartValueRemark | Varchar(50) null |  | 任何时候都可以修改 |
| 属性1值 | F\_ProeprtyValue1 | Varchar(50) |  |  |
| 属性2值 | F\_ProeprtyValue2 | Varchar(50) |  |  |
| 属性3值 | F\_ProeprtyValue3 | Varchar(50) |  |  |
| 属性4值 | F\_ProeprtyValue4 | Varchar(50) |  |  |
| 创建时间 | F\_CreateDatetime | Datetime, NOT NULL |  | 便于追踪 |
| 修改时间 | F\_UpdateDatetime | Datetime, NOT NULL |  | 最近修改时间。便于追踪 |
| 当值入库ID | F\_CurrentWarehousingID | Int NULL | NULL | 外键。正在出售的商品对应的入库批次。  要计算商品的毛利率，必须知道这批商品的进货价和出售价格。而入库批次当中有记录此商品的进货价 |
|  |  |  |  |  |

#### 商品属性表T\_CommodityProperty

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID | INT NOT NULL |  | PRIMARY KEY。只有一行数据 |
| 属性名称1 | F\_Name1 | Varchar(16), NULL。去除空格后不能为空串 |  |  |
| 属性名称2 | F\_Name2 | Varchar(16), NULL。去除空格后不能为空串 |  |  |
| 属性名称3 | F\_Name3 | Varchar(16), NULL。去除空格后不能为空串 |  |  |
| 属性名称4 | F\_Name4 | Varchar(16), NULL。去除空格后不能为空串 |  |  |
| 创建时间 | F\_CreateDatetime | Datetime, NOT NULL |  | 便于追踪 |
| 修改时间 | F\_UpdateDatetime | Datetime, NOT NULL |  | 最近修改时间。便于追踪 |
|  |  |  |  |  |

这个表只有一行数据。属于配置性数据，其值为：

|  |  |
| --- | --- |
| SQL field | 值 |
| F\_ID | 1 |
| F\_Name1 |  |
| F\_Name2 |  |
| F\_Name3 |  |
| F\_Name4 |  |
| F\_CreateDatetime | Now() |
| F\_UpdateDatetime | Now() |
|  |  |

#### 供应商商品表T\_ProviderCommodity

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID | INT NOT NULL |  | PRIMARY KEY，自增1 |
| 供应商ID | F\_ProviderID | 数字，NOT NULL |  | FOREIGN KEY。指向供应商表的F\_ID。与商品ID合成惟一键 |
| 商品ID | F\_CommodityID | 数字，NOT NULL |  | FOREIGN KEY。指向商品表的F\_ID。与供应商ID合成组合惟一键 |
| 创建时间 | F\_CreateDatetime | NOT NULL |  | 便于追踪 |
| 修改时间 | F\_UpdateDatetime | Datetime, NOT NULL |  | 最近修改时间。便于追踪 |
|  |  |  |  |  |

#### 商品修改历史T\_CommodityHistory

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID | INT NOT NULL |  | PRIMARY KEY，自增1。 |
| 商品ID | F\_CommodityID | INT NOT NULL |  | 外键 |
| 字段名称 | F\_FieldName | Varchar(20) not null |  | 只能是以下值之一：  商品条码  商品名称  规格  包装单位  商品类别  零售价  商品库存 |
| 字段旧值 | F\_OldValue | Varchar(32) not null |  |  |
| 字段新值 | F\_NewValue | Varchar(32) not null |  |  |
| 修改人 | F\_StaffID | Int null |  | 外键 |
| 是否系统自动修改 | F\_BySystem | int | 0 | 1是，0否 |
| 修改日期 | F\_Datetime | Datetime |  |  |

#### 商品包装单位T\_PackageUnit

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 包装单位名称 | F\_Name | 允许以中英数值、空格形式出现，不允许仅使用空格  长度(0, 8] |  | 支、条、个、包、箱等量词 |

#### 商品一品多码T\_Barcodes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID | 同上 |  |  |
| 商品ID | F\_CommodityID | 同上 |  | 组合唯一键 |
| 条码 | F\_Barcode | 仅允许英文、数值形式出现，字符数暂不设上限  长度[7, 64] |  |

#### 商品类别的大类T\_CategoryParent

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 大类名称 | F\_Name | (0, 10] 个中英文、数字，无空格 |  | 大类名称 |

#### 商品类别T\_Category

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 大类 | F\_ParentID | Int not NULL |  | 外键。对应T\_CategoryParent的F\_ID |
| 类别名称 | F\_Name | (0, 10]个中英文、数字，无空格 |  |  |
|  |  |  |  |  |

#### 商品品牌T\_Brand

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 品牌名称 | F\_Name | (0, 20] 个中英文、数字，无空格 |  |  |
|  |  |  |  |  |

#### 对供应商询价商品表T\_PriceSurvey

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 商品ID | F\_CommodityID |  |  |  |
| 国际条码 | F\_Barcode |  |  |  |
| 进货价 | F\_Price |  |  |  |
| 供应商ID | F\_ProviderID |  |  |  |
|  |  |  |  |  |

#### 进货价调整单T\_PriceAdjustment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 单号 | F\_ID |  |  |  |
| 开单日期 | F\_DateTime |  |  |  |
| 创建者ID | F\_StaffID |  |  |  |
|  |  |  |  |  |

#### 进货价调整单商品表T\_PriceAdjustmentCommodity

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 序号 | F\_ID |  |  |  |
| 单号 | F\_ParentID |  |  |  |
| 商品ID | F\_CommodityID |  |  |  |
| 国际条码 | F\_Barcode |  |  |  |
| 进货价 | F\_PriceOriginal |  |  |  |
| 调整价 | F\_PriceAdjusted |  |  |  |
|  |  |  |  |  |

#### （初版不做）促销计划进货价格单商品表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

#### 组合型商品表T\_SubCommodity

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
|  | F\_CommodityID |  |  | 组合型商品的ID。外键 |
| 子商品ID | F\_SubCommodityID |  |  | 被组合的商品的ID。  外键，对应商品表的F\_ID。与F\_CommodityID合成惟一键。  被组合的商品只能是单品，不能是服务型商品、多包装商品和组合商品。 |
| 子商品数量 | F\_SubCommodityNO | Int not null，>0 |  | 被组合的商品的数量 |
| 子商品的价格 | F\_Price | DECIMAL (20,6) NOT NULL，>0.000000d |  | 用户填写的价格。通常会低于这个商品的原价。  组合商品的零售价=各子商品乘以本价格之和 |
| 创建日期时间 | F\_CreateDatetime | Not null | Now() |  |
|  | F\_UpdateDatetime | Not null | Now() |  |

商品~~零售价调价历史表T\_RetailPriceHistory~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
|  | ~~F\_ID~~ |  |  |  |
|  | ~~F\_CommodityID~~ |  |  |  |
|  | ~~F\_PriceOld~~ | ~~Float~~ |  | ~~旧零售价~~ |
|  | ~~F\_PriceNew~~ | ~~Float~~ |  | ~~新零售价~~ |
|  | ~~F\_CreateDate~~ |  |  | ~~修改零售价的日期时间~~ |

### 入库次表

一个采购订单可多次入库，每次入库插一行数据进入库次表T\_Warehousing，同时插相关商品进入库次表商品表T\_WarehousingCommodity。

入库的商品，不一定是在采购订单中的商品，原因是用户看到批发商正好有自己另外想要的商品，会进行**计划外采购**。

#### 入库次表T\_Warehousing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 编号 | F\_SN | VARCHAR(20) NOT NULL |  | RK+XXXXXXXX(年月日)+0000(自增长的流水号) 例：RK201906100001s |
| 供应商ID | F\_ProviderID | Int not null |  | 外键。前端显示供应商名称时，拿最新的名称显示即可 |
| 收货仓库 | F\_WarehouseID |  |  |  |
| 业务员 | F\_StaffID |  |  |  |
| 入库日期 | F\_CreateDatetime |  |  | 可根据此字段确定是哪次入库 |
| 对应的采购订单ID | F\_PurchasingOrderID | Int null |  | 外键。可能不存在。如果存在，则对应的供应商、业务员、付款日期等相关信息全部可知 |
| 状态 | F\_Status | INT NOT NULL |  | -- 0:未审核,1,已审核 |

#### 入库次表商品表T\_WarehousingCommodity

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 入库次表ID | F\_WarehousingID |  |  |  |
| 商品ID | F\_CommodityID |  |  | 外键。入库的商品，不一定是在采购订单中的商品 |
| 商品名称 | F\_CommodityName |  |  | 根据“保持历史原貌”的设计原则，这个名称保存的是当时的商品名称。此F\_CommodityID对应的商品的名称可能在将来被修改 |
| 条形码ID | F\_BarcodesID |  |  | 外键 |
| 包装单位ID | F\_PackageUnitID |  |  | 外键 |
| **本次**收货数量 | F\_NO |  |  |  |
| 采购规格 | F\_PurchasingSpecification |  |  | 和商品本身的规格有何区别？一箱与一瓶的区别？ |
| **进货**价 | F\_Price |  |  |  |
| 金额 | F\_Amount |  |  | 进货时收的钱，不一定精确地= F\_NO \* F\_Price，所以需要此字段存储门店老板支付给批发商的真正金额 |
| 生产日期 | F\_ProductionDatetime |  |  |  |
| 保质期 | F\_ShelfLife |  |  |  |
| 失效期 | F\_ExpireDatetime |  |  | 失效期＝生产日期＋保质期 |
| 可售数量 | F\_NOSalable | Int not NULL |  | 1、最初入库时，可售数量为F\_NO。随着销售的进行，这个数量会不断减少，直至0。到达0的话，商品表的F\_CurrentWarehousingID将变成下一入库批次。  2、当发生仓管退货时，更新商品表的当值入库ID（F\_CurrentWarehousingID）为本次入库的ID，同时本字段的值要变。下次销售的是本次入库的商品。本字段的值如何变化要看原先的当值入库ID是不是本次入库ID：  1、如果是，则可售数量=仓管退货数量+原先的可售数量  2、如果不是，则可售数量=仓管退货数量 |

### 采购订单

#### 采购计划表表T\_PurchasingPlanTable

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ~~ID~~ | ~~F\_ID~~ |  |  |  |
| ~~创建日期时间~~ | ~~F\_CreateDatetime~~ |  |  |  |
| ~~审核日期时间~~ | ~~F\_ApproveDatetime~~ |  |  |  |
| ~~类型~~ | ~~F\_Type~~ |  | ~~0~~ | ~~1：紧急型、0：非紧急型~~ |
| ~~提醒日期时间~~ | ~~F\_ReminderDatetime~~ |  |  | ~~界面选项（待BA确认）：~~  ~~不再提醒~~  ~~10分钟后再提醒~~  ~~1天后再提醒~~  ~~2天后再提醒~~  ~~如果设置为不再提醒则此值为1900-1-1-1 00:00:00。否则按照提醒时间设置~~ |
|  |  |  |  |  |

#### 采购计划表商品表T\_PurchasingPlanTableCommodity

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ~~ID~~ | ~~F\_ID~~ |  |  |  |
| ~~采购计划表~~ | ~~F\_PurchasingPlanTableID~~ | ~~NULL~~ |  | ~~外键~~ |
| ~~商品ID~~ | ~~F\_CommodityID~~ |  |  | ~~以下情况的提醒：~~   * ~~预淘汰的商品（不要提醒）~~ * ~~和本表最近一条记录的商品列表完全相同。证明是重复，不再生成新的提醒。但是用户可以设置之前的旧提醒在多少时间消逝后再次提醒~~ * ~~已经在采购订单中的商品而该采购订单未全部入库~~ |
| ~~商品数量~~ | ~~F\_CommodityNO~~ |  | ~~最近一星期的交易量~~ | ~~如果最近一星期无交易，则再退一星期，直至有交易为止。用户可以设置此数值为＞0的任意自然数~~ |

#### 采购订单表T\_PurchasingOrder

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 编号 | F\_SN | VARCHAR(20) NOT NULL |  | CG+XXXXXXXX(年月日)+0000(自增长的流水号) 例：CG201906100001 |
| 操作者ID | F\_StaffID | Int not null |  | 外键 |
| 状态 | F\_Status |  | 0 | 0：未审核、1：已审核、2：部分入库、3：全部入库、4：已删除 |
| 供应商ID | F\_ProviderID | Int null |  | 删除采购订单时才设为NULL |
| 供应商名称 | F\_ProviderName | 参考供应商表名称字段, NULL |  | 删除采购订单时才设为NULL |
| 创建日期时间 | F\_CreateDatetime | Not null | Now() |  |
|  | F\_UpdateDatetime | Not null | Now() |  |
| 审核人ID | F\_ApproverID | NULL |  | 外键 |
| 审核日期时间 | F\_ApproveDatetime | NULL |  |  |
| 终止日期时间 | F\_EndDatetime | NULL |  |  |
| 采购总结 | F\_Remark | VARCHAR (128) NOT NULL |  |  |

#### 采购订单表商品表T\_PurchasingOrderCommodity

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 采购订单表ID | F\_PurchasingOrderID |  |  |  |
| 商品ID | F\_CommodityID |  |  |  |
| 商品名称 | F\_CommodityName | 参考商品表的名称字段 | NOT NULL | 记录采购此商品时的商品名称，保持历史上的一个快照，因为商品名称在采购之后，可能被用户改变。改变后，用户看回本采购，名字仍然是当初的名字，不是用户修改后的名字，保持了正确的历史 |
| 商品条形码ID | F\_BarcodeID |  | NOT NULL | 外键。 |
| 商品包装单位ID | F\_PackageUnitID |  | NOT NULL | 外键。 |
| 商品数量 | F\_CommodityNO |  | 1 | 用户可以设置此数值为＞0的任意自然数 |
| 建议采购单价 | F\_PriceSuggestion | float | 0.000 | 在审核采购订单前，此值都可以由门店老板或采购员更改。  它只是作为一个建议值，不一定是实际采购价格。  采购订单中的商品入库后，商品表中的采购价将会被系统重新计算。 |

### 仓管退货单

#### 仓管退货单表T\_ReturnCommoditySheet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 序号 | F\_ID |  |  |  |
| 编号 | F\_SN | VARCHAR(20) NOT NULL |  | TH+XXXXXXXX(年月日)+0000(自增长的流水号) 例：TH201906100001 |
| 操作人 | F\_StaffID | Int not null |  | 外键 |
| 审核状态 | F\_Status | int | 0 | 0，未审核。1，已审核 |
| 供应商 | F\_ProviderID | Int not null |  | 外键 |
| 退货日期 | F\_CreateDatetime | Datetime not null |  | 创建时间即为退货日期时间 |
| 修改日期 | F\_UpdateDatetime | Datetime not null |  | 修改和审批时要更新的字段 |
|  |  |  |  |  |

#### 仓管退货单商品表T\_ReturnCommoditySheetCommodity

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 序号 | F\_ID |  |  |  |
| 退货单ID | F\_ReturnCommoditySheetID | Int not null |  | 外键 |
| 商品ID | F\_CommodityID | Int not null |  | 外键 |
| 商品名称 | F\_CommodityName | Varchar(32) not null |  | 需要保持历史原貌 |
| 商品条形码 | F\_BarcodeID | Int not null |  | 外键 |
| 数量 | F\_NO | >0的整数 |  |  |
| 商品的规格 | F\_Specification | 允许以中英数值、空格形式出现，不允许仅使用空格  长度(0, 8] |  |  |
| 商品的采购价 | F\_PurchasingPrice | Float（在Java层为float，但在mysql中为Decimal(6,20) ）, 非负浮点数 |  |  |

### 盘点单

#### 盘点单T\_InventorySheet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 序号 | F\_ID |  |  |  |
| 编号 | F\_SN | VARCHAR(20) NOT NULL |  | PD+XXXXXXXX(年月日)+0000(自增长的流水号) 例：PD201906100001 |
| 盘点仓库 | F\_WarehouseID | 正整数 |  | 仓库名称；提供类别选择已创建仓库 |
| 盘点范围 | F\_Scope | 商品类别或特指商品 | 全类别 | ＝{0=全店盘点、1=指定类别、2=指定供应商、3=指定品牌} |
| 状态 | F\_Status |  | 待录入 | 种类：0=待录入、1=已提交、2=已审核、3=已删除 |
| 创建者 | F\_StaffID | 用户账号 |  | 系统登录账号 |
| 创建日期时间 | F\_CreateDatetime |  |  |  |
| 审核日期时间 | F\_ApproveDatetime |  |  |  |
| 盘点总结 | F\_Remark | [0, 128] |  | 备注 |
|  |  |  |  |  |

#### 盘点商品表T\_InventoryCommodity

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 序号 | F\_ID |  |  |  |
| 盘点单号 | F\_InventorySheetID |  |  |  |
| 商品ID | F\_CommodityID |  |  |  |
|  |  |  |  |  |
| 实盘商品数量 | F\_NOReal |  |  |  |
| 系统库存数量 | F\_NOSystem |  |  |  |
|  |  |  |  |  |

### 微信事件推送消息表T\_WxEventPushMessage

本表包含多种类型的消息，要根据里面的F\_MsgType判断是哪种消息推送。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 序号 | F\_ID | int |  |  |
|  | F\_ToUserName | Varchar(32) |  | 开发者微信号 |
|  | F\_FromUserName | Varchar(32) |  | 发送方帐号（一个OpenID） |
|  | F\_MsgType | Varchar(32) |  | 消息类型，event |
|  | F\_Event | Varchar(32) |  | 事件类型，user\_consume\_card(核销事件) |
|  | F\_CardId | Varchar(32) |  | 卡券ID |
|  | F\_UserCardCode | Varchar(32) |  | 卡券Code码 |
|  | F\_ConsumeSource | Varchar(32) |  | 核销来源。支持开发者统计API核销（FROM\_API）、公众平台核销（FROM\_MP）、卡券商户助手核销（FROM\_MOBILE\_HELPER）（核销员微信号） |
|  | F\_LocationName | Varchar(20),与t\_shop一致 |  | 门店名称，当前卡券核销的门店名称（只有通过自助核销和买单核销时才会出现该字段） |
|  | F\_StaffOpenId | Varchar(32) |  | 核销该卡券核销员的openid（只有通过卡券商户助手核销时才会出现） |
|  | F\_VerifyCode | Varchar(0) |  | 自助核销时，用户输入的验证码 |
|  | F\_RemarkAmount | Decimal(20,6) |  | 自助核销时 ，用户输入的备注金额 |
|  | F\_OuterStr | Varchar(0) |  | 开发者发起核销时传入的自定义参数，用于进行核销渠道统计 |
|  | F\_CreateTime | datetime |  | 消息创建时间 （整型） |
|  | F\_Deleted | int |  |  |

### 零售和收银

#### 零售单T\_RetailTrade

* 本表包含非退货型零售单和退货型零售单的数据。两者的区别在于，前者的F\_SourceID==-1，而后者的F\_SourceID>0。
* 使用优惠券消费，不算一种支付方式（因为支付方式及其金额的语义是：顾客必须有付钱，现金、微信、支付宝等等），只作为一种特殊的促销。在计算优惠时，先按照普通促销计算出价格，再使用优惠券，得出最终顾客需要付的钱。
* 退货的话优惠券不返回，积分不会减少。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 单据ID | F\_ID |  |  |  |
| 编号 | F\_SN | VARCHAR(26) NOT NULL |  | 对于非退货型的零售单，长度限制为24。LS+年月日+时分秒+4位pos\_id+四位随机数。查询RN可以用小写的LS，即ls。  退货类型的零售单的SN生成规则是源单SN后面加\_1(自增)，限制为26位。  模糊搜索RN时，必须至少输入LS+yyyyMMDD共10位 |
| 消费者（会员）ID | F\_VipID | Int NULL |  | 外键，指向T\_Vip.F\_ID |
| ~~小票格式ID~~ | ~~F\_SmallSheetID~~ | ~~Int not null~~ |  | ~~外键~~ |
| 源开单ID | F\_SourceID |  |  | 发生退货时，需要提供原单信息。退货数量不能大于购买的数量（如果发生此种情况，应如何处理？），所以每次退货不仅要找到源单据，还要检查退货数量是否符合此规则。退货完后，一般会销毁消费者的单据，所以退货行为一般只有一次。 |
| 收银机单号 | F\_LocalSN | Int not null |  | F\_POS\_ID为外键。  F\_POS\_ID+ F\_LocalSN+ F\_SaleDatetime为组合惟一键，惟一标识一个单。说明：  每台POS机上的每次消费都会形成本机上的流水号F\_LocalSN，是在这台收银机上的惟一单号。这个号加上收银机的ID和消费时间，形成一个独一无二的组合，惟一标识此次交易：   1. 一旦交易完成，这3个字段就会生成。 2. 一旦交易完成，这3个字段就永远不能再更改。因为别的POS机也好本机也好，都可能需要寻找出本次交易。只有这二个字段能够快速定位这次交易。当然，加上收银员ID可以更快定位。 3. 同步时，防止重复插入零售单流水到服务器，导致报表有低级错误！ |
| 收银机ID | F\_POS\_ID | Int not null |  |
| 消费时间 | F\_SaleDatetime | Datetime not null |  |
| 自定义logo | F\_Logo |  |  | 一般是门店logo |
| 收银员ID | F\_StaffID |  |  |  |
| 支付方式 | F\_PaymentType | Int |  | 共8种支付方式，用二进制方式记录支付方式，其中包含混合支付方式。  其它支付方式5/4/3/2/1，共5种，为预留支付方式。  目前频繁使用的支付方式是微信支付/支付宝支付/现金支付。  F\_PaymentType最高位7到最低位0依次为：其它支付方式5/4/3/2/1/微信支付/支付宝支付/现金支付，共8位有效，剩下的32-8=24位不作定义。  比如二进制00000001，是现金支付。  二进制00000101，是现金支付加微信支付。  二进制0000011，是现金加支付宝支付。  二进制00001000，是其它支付方式1支付。  二进制00010000，是其它支付方式2支付。  潜在的业务逻辑为：   1. 各种支付方式的总和必须=F\_Amount 2. 如果微信位或支付宝对应的位为1，则其对应的Amount必须>0.000000d |
| 帐号 | F\_PaymentAccount |  |  | 电子支付帐号，退钱，纠纷 |
| 状态 | F\_Status |  |  | 0挂起/1正常/2~~删除~~零售单正常但包含被删除的商品  为什么会出现状态为2的情况？主要有：   1. 后台已经删除了商品A，但是APP此时断网，没有同步到A已经被删除。这时售出A。连网后，上传到服务器，服务器发现A已经被删除，会标记此零售单的状态为2。 2. 后台已经删除了商品A，但是APP此时断网，没有同步到A已经被删除。这时售出A，又对A进行退货。连网后，上传到服务器，服务器发现A已经被删除，会标记这个零售单和退货单的状态为2。 3. 连网情况下，APP将商品A加入待售列表，但是此时老板删除了A。收银员照样能售出A。此零售单上传到服务器后，服务器发现A已经被删除，会标记这个零售单的状态为2。   以上3种情况下，APP查单时，如果零售单状态为2，则不显示商品列表给用户看，并告诉用户此单包含已经删除的商品。 |
| 备注 | F\_Remark |  |  |  |
| 同步时间 | F\_SyncDatetime |  |  | 服务器同步此交易的时间。在Action中设置值，其值为当前时间。目前多个表对应的MODEL都有SyncDatetime字段而对应的DB表没有此字段  ，只有本DB表有此字段 |
| 顾客支付的金额 | F\_Amount | Float |  | 此金额或许是已经通过促销计算后的金额 |
| 现金支付数目 | F\_AmountCash | Float，不能为负数 |  | 使用现金支付的数目。可以=0.000000d，商品免费。  纯现金支付才能为0.000000d，混合支付带有现金支付的话，现金支付也是要大于0.000000d |
| 支付宝支付数目 | F\_AmountAlipay | Float |  | 使用支付宝支付的数目 |
| 微信支付数目 | F\_AmountWeChat | Float |  | 使用微信支付的数目 |
| 其它支付的数目1 | F\_Amount1 | Float |  | 其它支付方式支付的数目1 |
| 其它支付的数目2 | F\_Amount2 | Float |  | 其它支付方式支付的数目2 |
| 其它支付的数目3 | F\_Amount3 | Float |  | 其它支付方式支付的数目3 |
| 其它支付的数目4 | F\_Amount4 | Float |  | 其它支付方式支付的数目4 |
| 其它支付的数目5 | F\_Amount5 | Float |  | 其它支付方式支付的数目5 |
| 支付宝订单号 | F\_AliPayOrderSN | Varchar(32) null |  | 支付宝支付后的订单号，惟一标识一次付款。等于支付宝返回的transaction\_id的值 |
| 微信订单号 | F\_WxOrderSN | Varchar(32) null |  | 微信支付后的订单号，惟一标识一次付款。等于微信返回的transaction\_id的值。  退款时也要使用这个字段 |
| 微信交易单号 | F\_WxTradeNO | Varchar(32) null |  | 微信支付时要传递给微信服务器的交易单号，惟一标识本次交易，格式为：CompanyID\_SHOPID\_POSID\_STAFFID\_yyyyMMDDhhmmssSSS。  等于微信返回的out\_trade\_no的值。  退款时也要使用这个字段 |
| 微信退款单号 | F\_WxRefundNO | Varchar(32) null |  | 微信退款单号，等于微信返回的out\_refund\_no的值。参看[购物退货和退款流程](#_购物退货和退款流程) |
| 微信退款描述 | F\_WxRefundDesc | Varchar(80) null |  | 微信退款描述，通常是描述原因，等于微信返回的refund\_desc的值。参看[购物退货和退款流程](#_购物退货和退款流程) |
| 微信退款子商户号 | F\_WxRefundSubMchID | Varchar(32) null |  | 微信退款要传递到微信服务器的的子商户ID：sub\_mch\_id。参看[购物退货和退款流程](#_购物退货和退款流程) |
| 优惠券抵扣金额 | F\_CouponAmount | Decimal(20, 6) default 0.000000d |  | 新增优惠券抵扣金额。使用优惠券时必填，不包括在Amount中，因为报表只记录真实金钱来往。在创建零售单时，是不知道值的。只有WX消息推送回来后，才知道F\_ConsumerOpenID用了什么券，用了多少张，F\_CouponAmount+=F\_CouponAmount，是累加起来的，不是直接赋值。处理后，标识对应的推送消息为已删除 |
| ~~积分抵扣的金额~~ | ~~F\_BonusAmout~~ | ~~Decimal(20,6) default 0.000000d~~ |  | ~~新增积分抵扣的金额，不包括在Amount中，因为报表只记录真实金钱来往~~ |
| 消费者的openId | F\_ConsumerOpenID | VARCHAR (100) |  | 标识谁用WX支付了本零售单，方便在接收WX消息推送时，识别是本零售单的消费者消费的，加券对应的钱到F\_CouponAmount上 |

微信退款注意事项： 1、交易时间超过一年的订单无法提交退款。2、微信支付退款支持单笔交易分多次退款，多次退款需要提交原支付订单的商户订单号和设置不同的退款单号。申请退款总金额不能超过订单金额。 一笔退款失败后重新提交，请不要更换退款单号，请使用原商户退款单号。在实现时，我们不支持针对一张单多次退款！3、请求频率限制：150qps，即每秒钟正常的申请退款请求次数不超过150次。    错误或无效请求频率限制：6qps，即每秒钟异常或错误的退款申请请求不超过6次。4、每个支付订单的部分退款次数不能超过50次。

参考：<https://pay.weixin.qq.com/wiki/doc/api/app/app.php?chapter=9_4&index=6>

#### 零售单使用的优惠券T\_RetailTradeCoupon

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
|  | F\_RetailTradeID |  |  | 外键 |
| ~~券ID~~ | ~~F\_CouponID~~ | ~~int not null~~ |  | ~~外键。券ID。使用优惠券时必填。字段名容易误会为卡ID，实际上是券ID~~ |
| 券code序列号 | ~~F\_ UserCouponCode~~  F\_ CouponCodeID | Varchar(100) not null,格式？ |  | ~~code序列号。使用优惠券时必填。不是外键，原因是发券后，才知道序列号。~~  外键，指向T\_CouponCode.F\_ID |
|  | F\_SyncDatetime | Datetime not null |  |  |

#### 零售单商品表T\_RetailTradeCommodity

对于非退货型零售单和退货型零售单，其从表均为T\_RetailTradeCommodity。

但是对于从表的从表，前者为T\_RetailTradeCommoditySource，后者为T\_ReturnRetailTradeCommodityDestination。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| F\_ID | F\_ID |  |  |  |
| 主表ID | F\_TradeID |  |  |  |
| 商品ID | F\_CommodityID |  | NULL | 外键。当本字段有效时，F\_GiftCommodityID字段无效 |
| 商品名称 | F\_CommodityName | 同T\_Commodity.F\_CommodityName |  | 记录售卖时此商品的名称。日后如果T\_Commodity中的商品名称改变了，此处名称仍然不变。 |
| 商品数量 | F\_NO |  |  |  |
| 条形码 | F\_BarcodeID |  |  | 外键。对应一个单品或一个多包装商品，其它业务逻辑处理条形码时，知道依赖性 |
| 可退货数量 | F\_NOCanReturn |  |  | 客户退货时，POS会从服务器重新下载原始交易并进行退货。可退货数量不能大于本字段的值，否则提醒收银员。退货后，同步到服务器时，服务器会将此值降低退货的数量。  意外情况1：零售开单创建后未同步到服务器，顾客就已经回来退货？--先触发同步，同步完成再退货  意外情况2：此字段值为负数，这表明出现了不正常的退货。在POS端获取本交易后，应当对收银员进行提醒。后台暂时不阻止这种情况的发生。 |
| 商品原会员价 | F\_PriceVIPOriginal | float | NULL | 记录在本零售进行时的商品会员价 |
| 商品原零售价 | F\_PriceOriginal | float | NULL | 记录在本零售进行时的商品单价 |
| 退货价 | F\_PriceReturn | float | NULL | 顾客进行商品退货的时候，需要归还金额。归还的金额和交易时商品的原价及促销后的价格相关。本字段记录的是退货时的价格。它可能等于F\_PriceOriginal，也可能等于1次或N次促销优惠后的价格（如F\_PriceSpecialOffer） |
| ~~商品特价~~ | ~~F\_PriceSpecialOffer~~ | ~~float~~ | ~~NULL~~ | ~~顾客进行交易时，本商品可能正在搞特价，本字段用于记录此时商品的特价。如果有多个特价（这种情况一般不会发生），则后面的特价会覆盖前面的特价~~ |
| ~~折扣~~ | ~~F\_Discount~~ | ~~float~~ | ~~1(代表不打折)~~ | ~~若本促销是折扣促销，则本字段记录的是折扣，如0.99和0.9，分别代表9.9折和9折~~  ~~POS和web界面上显示的是被折的价钱，其= F\_PriceOriginal- F\_PriceOriginal X F\_Discount，而不是本字段的值，因为显示本字段的值意义不大~~ |
| ~~赠送商品ID~~ | ~~F\_GiftCommodityID~~ |  | ~~NULL~~ | ~~外键。当本字段有效时，F\_CommodityID字段无效~~ |
|  |  |  |  |  |

#### 零售单商品来源表T\_RetailTradeCommoditySource

所谓来源，系标识该零售单中的商品分别属于哪几个入库，以便计算进货价和毛利等。绝大多数情况下，一张零售单内的商品只会对应一个入库ID。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 零售单商品ID | F\_RetailTradeCommodityID | Int not null |  | 组合惟一键。外键，指向T\_RetailTradeCommodity.F\_ID |
| 实际减少库存的商品ID | F\_ReducingCommodityID | INT NOT NULL |  | 组合惟一键。  实际被减少库存的商品ID。它可能是组合商品中的子商品，也可能是多包装商品对应的单品，也可能是服务型商品 |
| 零售单商品数量 | F\_NO | Int not null |  | 这个数量<=对应的零售单商品表中该商品的数量。它表明这F\_NO个商品，属于哪次入库，进而知道它的进货价—因为入库中有保存这F\_NO个商品的进货价是多少 |
| 入库ID | F\_WarehousingID | Int null |  | 组合惟一外键。  有的商品是没有来源的，比如新开店后，用户创建了没有期初值的商品，就直接售卖。此时，创建的零售单是找不到相应的入库记录的，于是本字段就可能为空。如果本字段为空，就找不到相应的进货价，进而影响到毛利和毛利率的计算。BA建议，一旦找不到入库记录，就将其进货价设置为0.000000d。售出多少利润就是多少，这会对毛利产生巨大影响，但实属无奈 |

组合惟一键：UNIQUE KEY UniqueWarehousing (F\_RetailTradeCommodityID, F\_WarehousingID, F\_ReducingCommodityID) -- 惟一标识N个同样的商品的来源（入库）

#### 零售单商品退货去向表T\_ReturnRetailTradeCommodityDestination

所谓去向，系标识该退货型零售单中的商品分别按照来源表退回到哪几个入库，以便计算员工业绩等。绝大多数情况下，一张零售单内的商品只会对应一个入库ID。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 零售单商品ID | F\_RetailTradeCommodityID | Int not null |  | 组合惟一键。外键，指向T\_RetailTradeCommodity.F\_ID |
| 实际增加库存的商品ID | F\_IncreasingCommodityID | INT NOT NULL |  | 组合惟一键。  实际被增加库存的商品ID。它可能是组合商品中的子商品，也可能是多包装商品对应的单品，也可能是服务型商品 |
| 零售单商品数量 | F\_NO | Int not null |  | 这个数量<=对应的零售单商品表中该商品的数量。它表明这F\_NO个商品，属于哪次入库，进而知道它的进货价—因为入库中有保存这F\_NO个商品的进货价是多少 |
| 入库ID | F\_WarehousingID | Int null |  | 组合惟一外键。  有的商品是没有来源因此没有去向的。  比如新开店后，用户创建了没有期初值的商品，就直接售卖。此时，创建的零售单是找不到相应的入库记录的，于是本字段就可能为空。如果本字段为空，就找不到相应的进货价，进而影响到毛利和毛利率的计算。BA建议，一旦找不到入库记录，就将其进货价设置为0.000000d。售出多少利润就是多少，这会对毛利产生巨大影响，但实属无奈。退货的时候，也无法确定这些商品应该归回哪次入库，所以本字段会变为NULL。作报表时，毛利为价格的负数（因进货价为0）。  如果有来源，那本字段不会为NULL。 |

组合惟一键：UNIQUE KEY UniqueWarehousing (F\_RetailTradeCommodityID, F\_WarehousingID, F\_IncreasingCommodityID) -- 退货时，惟一标识N个同样的商品退回到哪次入库。

#### 零售单促销表T\_RetailTradePromoting

用来记录一张零售单计算促销的过程

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 零售单ID | F\_TradeID |  |  | 外键，t\_retailtrade(F\_ID) |
|  | F\_SyncDatetime |  |  |  |
|  | F\_CreateDatetime DATETIME NOT NULL DEFAULT now() |  |  |  |
| ~~促销计算描述~~ | ~~F\_ProcessFlow~~ | ~~Varchar()~~ |  | ~~记录零售单在促销中的处理过程，至少包括以下要素：~~   1. ~~计算前的顾客需要支付的金额~~ 2. ~~计算过程，包含规则、规则约束下的算式及相关说明~~ 3. ~~计算后的顾客需要支付的金额~~   ~~生成的内容是html格式，方便BA通过浏览器审阅。~~ |

零售单促销过程表T\_RetailTradePromotingFlow

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 零售单促销ID | F\_RetailTradePromotingID INT NOT NULL |  |  | 外键 |
|  | F\_PromotionID int not null |  |  | 外键 |
|  | F\_ProcessFlow | VARCHAR(2048) NOT NULL。必须有内容 |  | 记录一个促销计算一张零售单优惠的全过程 |
|  | F\_SyncDatetime |  |  |  |
|  | F\_CreateDatetime DATETIME NOT NULL DEFAULT now() |  |  |  |

#### 收银汇总T\_RetailTradeAggregation

上班后创建一张空的收银汇总到SQLite，交班时由POS上传到服务器。

每次销售，都重新计算交易单数，营业额，现金收入、微信收入、支付宝收入。

每次退货，都重新计算营业额，现金收入、微信收入、支付宝收入。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 门店人员ID | F\_StaffID |  |  | 门店人员表的外键 |
| POS ID | F\_PosID | Int not null |  | 上传的POS的ID。外键 |
| 上班时间 | F\_WorkTimeStart |  |  |  |
| 下班时间 | F\_WorkTimeEnd |  |  | 和POSID、STAFFID合成一个惟一键。  本字段有值后不可再修改！！！ |
| 交易单数 | F\_TradeNO | Int |  | =销售的单数，不包含退货的单数 |
| 营业额 | F\_Amount | Double |  | =销售得到的钱-退货的钱 |
| 准备金 | F\_ ReserveAmount | Double |  |  |
| 现金收入 | F\_CashAmount | Double |  |  |
| 微信收入 | F\_WechatAmount | Double |  |  |
| 支付宝收入 | F\_AlipayAmount | Double |  |  |
| 其它支付的数目1 | F\_Amount1 | Double |  | 其它支付方式支付的数目1 |
| 其它支付的数目2 | F\_Amount2 | Double |  | 其它支付方式支付的数目2 |
| 其它支付的数目3 | F\_Amount3 | Double |  | 其它支付方式支付的数目3 |
| 其它支付的数目4 | F\_Amount4 | Double |  | 其它支付方式支付的数目4 |
| 其它支付的数目5 | F\_Amount5 | Double |  | 其它支付方式支付的数目5 |
| 上传到服务器的时间 | F\_UploadDateTime |  |  | 有时上传不成功，时间会滞后，要记录此时间，以观察客户的网络情况并提供收银汇总的证据。  此时间由服务器端设置。 |

#### 小票格式主表T\_SmallSheetFrame

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
|  | F\_Logo | TEXT |  |  |
| 分隔字符 | F\_DelimiterToRepeat | VARCHAR (1) NOT NULL DEFAULT ‘-‘ |  | 小票上面常用虚线-----隔开不同的区域。这里设置为可以让用户自定义一个字符做分隔线的单元，比如星号\*，&，￥等等 |
|  | F\_countOfBlankLineAtBottom | Int not null，[0，5] |  |  |
| 状态 | F\_Status | Int not null |  | 0，正常。1=已经删除 |

#### 小票格式从表，小票内容格式表T\_SmallSheetText

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
|  | F\_Content | VARCHAR(100) NOT NULL |  |  |
| F\_FrameID | F\_Size | FLOAT NOT NULL |  | x |
| 字体加粗 | F\_Bold | INT NOT NULL |  |  |
| 内容位置 | F\_Gravity | INT NOT NULL |  |  |
| 对应的小票的ID，主表ID | F\_FrameID | INT NOT NULL |  |  |
|  |  |  |  |  |

相关缓存表，查看[同步缓存表](#_5.1.18_同步缓存表)。

#### 报表

##### 综述

互参：[毛利率计算](#_商品毛利率计算)。

* 设计零售测试数据的流程：

1、根据数据库表结构和业务需求在excel表设计零售数据；

2、生成sql语句，插入insertTable表，运行小王子，搜索有无“测试错误”的字眼；

3、根据要测试的时间段运行相关test代码或用sql修改报表运行时间，生成需要的测试数据；

4、将生成的测试数据与自己手动计算的数据进行对比，查看有无错误；

PS：测试时不能直接将数据插入数据库，应放入insertTable表；

* 系统销售额的计算：

销售额=销售型零售单主表的支付金额总和-退货型零售单主表的支付金额总和；

系统销售毛利的计算：

销售毛利=销售型零售单主表的支付金额总和-退货型零售单主表的支付金额总和-零售来源表的实际减少库存\*对应商品关联的入库单的入库价；

PS：没有入库ID的相当于入库价为0，跨库的会根据关联的入库ID查找到对应的入库价；

* 商品来源：

如果该商品存在一部分有入库，一部分没入库的情况，其来源优先算有入库的，比如：

1、商品A创建时没有写期初值但是在现实中有数量，而后入库一批商品A，此时售卖或退货商品A优先算这批入库的商品A；

2、创建期初值为5的商品B（会默认创建一个入库单1），售卖3件，此时通过入库单2入库8件商品B，售出1件商品B，此商品B的来源优先取原先的入库单1；  
PS：先入库的先卖，卖完再算没入库的，售卖商品时对应入库单的可售卖数量会同时减少相应的数值；

* 关于零售来源表的设计：

普通商品：有来源表并且入库ID存在有和无两种情况；

多包装商品：有来源表但来源表记录的是其关联的单品并且入库ID存在有和无两种情况；

组合商品：有来源表但来源表记录的是其关联的子商品并且入库ID存在有和无两种情况；

服务商品：有来源表并且入库ID始终不存在；

* 关于退货的设计：

部分退货时，零售单从表的可退货数量字段NOCanReturn会相应减去对应的数值，售卖数量的字段NO则保持不变，零售单来源表的实际减少库存字段NO也会相应减去对应的数值；

全部退货时，零售单从表的可退货数量字段NOCanReturn为0，售卖数量的字段NO则保持不变，此时该零售单没有来源表；

超额退货......有无存在这种情况。。。

跨库退货时，与售卖时相反，优先退新入库的，并且退货的数量将加到最新入库的入库单的可售数量字段；

PS：每张零售单只能退一次，并且退货单的零售主表的SourceID字段会记录退的是哪张单，而销售型零售单的SourceID同一为-1；

* 退货和去向表设计：

##### 销售商品大类日报表T\_RetailTradeDailyReportByCategoryParent

本表是[中间结果表](#_定义)。

通常是明天凌晨计算今天的报表。由夜间任务自动计算。

如果某天出现了对今天的零售单的退货，则需要重新计算今天的报表。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 日期 | F\_Datetime | Datetime not null |  | 惟一组合键：日期+商品大类ID |
| 商品大类ID | F\_CategoryParentID | Int not null |  | 外键。 |
| 销售总额 | F\_TotalAmount | Decimal(20, 6) not null |  | 计算过程: 销售总额 = 售卖总额-退货总额。（当得出的是负数也是个正常的现象，这说明的是当天的退货总额大于当天的售卖总额）  售卖总额：通过零售单商品表的价格 \* 数量，由商品大类ID进行统计。  退货总额：通过零售单商品表的价格 \* 数量，由商品大类ID进行统计。 |
| 创建时间 | F\_CreateDatetime | Datetime, NOT NULL |  | 便于追踪 |
| 修改时间 | F\_UpdateDatetime | Datetime, NOT NULL |  | 最近修改时间。便于追踪。出现退货后，这个时间会被更新 |

惟一组合键：日期+商品大类ID

##### 商品零售日报表T\_RetailTradeDailyReportByCommodity

本表是[中间结果表](#_定义)。

通常是明天凌晨计算今天的报表。由夜间任务自动计算。

如果某天出现了对今天的零售单的退货，则需要重新计算今天的报表。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 日期 | F\_Datetime | Datetime not null |  | 惟一组合键：日期+商品ID |
| 商品ID | F\_CommodityID | Int not null |  | 外键。 |
| 销售数量 | F\_NO | Int not null |  | 销售数量: 当天所有销售总数量（只算卖出，不算退货）。 |
| 总进货价 | F\_TotalPurchasingAmount | Decimal(20, 6) not null |  | 总进货价： 总进货价 = 当天卖出商品的进货价 – 当天所退货商品的进货价。 |
| 销售总额 | F\_TotalAmount | Decimal(20, 6) not null |  | 销售总额：销售总额 = 当天销售商品的数量 \* 当天单个货品销售单价 – 当天退货的总额。（当得出的是负数也是个正常的现象，这说明的是当天的退货总额大于当天的售卖总额） |
| 销售毛利 | F\_GrossMargin | Decimal(20, 6) not null |  | 销售毛利：销售毛利 = 当天销售总额 – 当天总进货价。(销售毛利可能会出现负数的情况， 那就是当以上的销售总额的值为负数的时候) |
| 创建时间 | F\_CreateDatetime | Datetime, NOT NULL |  | 便于追踪 |
| 修改时间 | F\_UpdateDatetime | Datetime, NOT NULL |  | 最近修改时间。便于追踪。出现退货后，这个时间会被更新 |

惟一组合键：日期+商品ID

##### Staff零售日报表T\_RetailTradeDailyReportByStaff

本表是[中间结果表](#_定义)。

通常是明天凌晨计算今天的报表。由夜间任务自动计算。

如果某天出现了对今天的零售单的退货，则需要重新计算今天的报表。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 日期 | F\_Datetime | Datetime not null |  | 惟一组合键：日期+STAFF ID  时、分、秒均为0 |
|  | F\_StaffID | Int not null |  | 外键 |
| 销售总笔数 | F\_NO | Int not null |  | 员工业绩报表，统计零售单，不统计退货单。 |
| 销售总额 | F\_TotalAmount | Decimal(20, 6) not null |  | 计算过程：   1. 售卖总额 - 退货总额 = 销售总额（当退货金额大于售卖金额的时候销售总额就为负数）。 2. 售卖总额 = 每一张零售单（不包含退货）售卖金额之和 – 售卖商品的成本价之和 3. 退货总额 = 每一张退货零售单售卖金额之和 – 退货商品的成本价之和   退货价计算: 当统计退货价时，需要查看源单（零售单）的操作人此时有无上班。当在上班，则算源单的操作人头上；没有上班，则算退货单的操作人头上。  举例：  1.当A,B都上班，A销售100，B退货A的零售单50，此时A上班，退货的金额算在A头上,A的业绩是50。  2.当A没有上班，B上班，B对A的业绩退款50元，此时A未上班，退货的金额算在B头上,B的业绩是-50。 |
| 销售总毛利 | F\_GrossMargin | Decimal(20, 6) not null |  | 计算过程：销售总毛利 = 销售毛利 - 退货毛利 。（这里的销售毛利也是会出现负数的情况，那就是Staff的当天销售总额为负数的时候。）  销售毛利 = 销售总额 - 销售总成本。  退货毛利 = 退货总额 - 退货总成本。 |
| 创建时间 | F\_CreateDatetime | Datetime, NOT NULL |  | 便于追踪 |
| 修改时间 | F\_UpdateDatetime | Datetime, NOT NULL |  | 最近修改时间。便于追踪。出现退货后，这个时间会被更新 |

Staff的销售业绩=他的所有非退货型零售单(t\_retailtrade.F\_SourceID=-1)-他的所有退货型零售单(t\_retailtrade.F\_SourceID>0)。

比如，StaffA第1天有零售单1、2、3、4、5，StaffB将4退货，则StaffA在第一天的业绩要减去对4的退货。

##### 销售日报汇总表T\_RetailTradeDailyReportSummary

本表是[中间结果表](#_定义)。

通常是明天凌晨计算今天的报表。由夜间任务自动计算。

如果某天出现了对今天的零售单的退货，则需要重新计算今天的报表。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 统计日期 | F\_Datetime | Datetime not null |  | 惟一组合键：日期 |
| 销售笔数 | F\_TotalNO | INT NOT NULL |  | 销售笔数： 今天所销售的笔数总量。（不包含退货零售单） |
| 总进货价 | F\_PricePurchase | DECIMAL(20,6) NOT NULL |  | 总进货价： 总进货价 = 当天卖出商品的进货价 – 当天所退货商品的进货价 。 |
| 销售额 | F\_TotalAmount | DECIMAL(20, 6) NOT NULL |  | 销售额: 销售额 = （当天销售的商品 \* 当天零售单个商品的单价）之和 – 当天退货的总额。（销售额出现负数的正常的现象，这样说明的是当天所退货的总额比当天零售总额大） |
| 客单价 | F\_AverageAmountOfCustomer | DECIMAL(20, 6) NOT NULL | 0.000000 | 客单价： 客单价 = 销售额F\_TotalAmount / 零售单数（不包含退货）。  若本日无售出的零售单，则本字段为0.000000 |
| 销售毛利 | F\_TotalGross | DECIMAL(20, 6) NOT NULL | 0.000000 | 销售毛利： 销售毛利 = 当天销售额F\_TotalAmount – 当天总进货价F\_PricePurchase。（销售毛利为负数的时候，说明当天的销售额是为负数） |
| 销售毛利率 | F\_RatioGrossMargin | DECIMAL(20, 6) NOT NULL | 0.000000 | 销售毛利率：销售毛利率 = （当天销售额F\_TotalAmount – 当天总进货价F\_PricePurchase） / 当天销售额  若本日无销售额，则本字段为0.000000 |
| 销售数量最高的商品ID | F\_TopSaleCommodityID | INT NULL |  | 外键。当天不做生意的话，此项为NULL。  夜间任务先计算商品零售报表T\_RetailTradeReportByCommodity，从中得到本字段的值。 |
| 销售数量最高的商品的数量 | F\_TopSaleCommodityNO | INT NOT NULL | 0 | 夜间任务先计算商品零售报表T\_RetailTradeReportByCommodity，从中得到本字段的值。  某种商品的销售总额 = 该商品的当天销售总数 \* 当天该商品的销售单价 – 当天该商品所退货数量 \*当天该商品的销售单价。 |
| 购买额最高的客户 | F\_TopPurchaseCustomerName | VARCHAR(30) NOT NULL | ‘’ | 当天购买商品总额最高的顾客名称。  购买额最高的客户的总额：购买额最高的客户的总额 = 该顾客所购买商品的数量 \* 每一件商品的销售单价。包含退货？？？ |
| 创建时间 | F\_CreateDatetime | DATETIME NOT NULL | Now() |  |
| 修改时间 | F\_UpdateDatetime | DATETIME NOT NULL | Now() |  |

惟一组合键：日期

##### 销售月报汇总表T\_RetailTradeMonthlyReportSummary

由夜间任务插入数据到本表中。

每月1号，重新生成上月的数据。

每天重新生成本月的数据。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 统计日期 | F\_Datetime | Datetime not null |  | 此字段中只有年、月有效，因为本表是月报。惟一组合键：日期 |
| ~~销售笔数~~ | ~~F\_TotalNO~~ | ~~INT NOT NULL~~ |  | 销售笔数 |
| ~~总进货价~~ | ~~F\_PricePurchase~~ | ~~DECIMAL(20,6) NOT NULL~~ |  |  |
| 销售额 | F\_TotalAmount | DECIMAL(20, 6) NOT NULL |  | 销售额： 当月每天销售额之和 |
| ~~客单价~~ | ~~F\_AverageAmountOfCustomer~~ | ~~DECIMAL(20, 6) NOT NULL~~ | ~~0.000000~~ |  |
| 销售毛利 | F\_TotalGross | DECIMAL(20, 6) NOT NULL | 0.000000 | 销售毛利 ： 当月销售毛利= 当月销售额 – 当月总进货价。 |
| ~~销售毛利率~~ | ~~F\_RatioGrossMargin~~ | ~~DECIMAL(20, 6) NOT NULL~~ | ~~0.000000~~ |  |
| ~~销售额最高的商品ID~~ | ~~F\_TopSaleCommodityID~~ | ~~INT NULL~~ |  | ~~外键。当天不做生意的话，此项为NULL。~~  ~~夜间任务先计算商品零售报表T\_RetailTradeReportByCommodity，从中得到本字段的值~~ |
| ~~销售额最高的商品的数量~~ | ~~F\_TopSaleCommodityNO~~ | ~~INT NOT NULL~~ | ~~0~~ | ~~夜间任务先计算商品零售报表T\_RetailTradeReportByCommodity，从中得到本字段的值~~ |
| ~~购买额最高的客户~~ | ~~F\_TopPurchaseCustomerName~~ | ~~VARCHAR(30) NOT NULL~~ | ~~‘’~~ |  |
| 创建时间 | F\_CreateDatetime | DATETIME NOT NULL | Now() |  |
| 修改时间 | F\_UpdateDatetime | DATETIME NOT NULL | Now() |  |

惟一组合键：日期

##### 会员零售报表T\_RetailTradeReportByVIP

本表是[中间结果表](#_定义)。

通常是明天凌晨计算今天的报表。由夜间任务自动计算。

如果某天出现了对今天的零售单的退货，则需要重新计算今天的报表。

Xxxxxxxxxxxx

##### 单据零售报表T\_RetailTradeReportBySheet

本表是[中间结果表](#_定义)。

通常是明天凌晨计算今天的报表。由夜间任务自动计算。

如果某天出现了对今天的零售单的退货，则需要重新计算今天的报表。

Xxxxxxxxxxxx

### 门店用户

#### T\_Staff（老板和员工共用一个表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 姓名 | F\_Name | 中英文，(0, 12]字符。Varchar(12), NOT NULL |  |  |
| 手机号码 | F\_Phone | Varchar(32) not null, 11位数字组成，第1个数字必须为1 |  |  |
| ~~门店用户在POS机上的ID~~ | ~~F\_IDInPOS~~ |  |  | ~~门店用户在POS机上创建的话，会生成这个ID~~ |
| ~~POS机的SN码~~ | ~~F\_POS\_SN~~ |  |  | ~~门店用户在POS机上创建的话，会记录这个POS机的SN~~ |
| 身份证号码 | F\_ICID |  |  |  |
| 微信号 | F\_WeChat |  |  |  |
| 密码 | F\_Password | Varchar(32) not null，明文密码不能含有中文，6到16位，首尾不能为空。加密后的密码则固定是32位大写字母和数字的组合 |  | DB存储的是加密后的密码 |
| 密码有效期 | F\_PasswordExpireDate |  |  |  |
| 首次登录成功？ | F\_IsFirstTimeLogin |  | 1 | 若为首次使用密码登录成功，必须重置密码 |
| ~~问题1~~ | ~~F\_Q1~~ |  |  | ~~用于重置密码~~ |
| ~~回答1~~ | ~~F\_A1~~ |  |  | ~~用于重置密码~~ |
| ~~问题2~~ | ~~F\_Q2~~ |  |  | ~~用于重置密码~~ |
| ~~回答2~~ | ~~F\_A2~~ |  |  | ~~用于重置密码~~ |
| ~~问题3~~ | ~~F\_Q3~~ |  |  | ~~用于重置密码~~ |
| ~~回答3~~ | ~~F\_A3~~ |  |  | ~~用于重置密码~~ |
|  |  |  |  |  |
| 门店ID | F\_ShopID | Int not null |  |  |
| 部门ID | F\_DepartmentID | Int Not NULL |  |  |
| 状态 | F\_Status | Int Not null |  | 0，在职。1，离职。 |
| 创建时间 | F\_CreateDatetime | NOT NULL |  | 便于追踪 |
| 修改时间 | F\_UpdateDatetime | Datetime, NOT NULL |  | 最近修改时间。便于追踪 |
|  | F\_OpenID | VARCHAR (100) |  | 用于绑定公众号。老板绑定博昕公众号的流程：  1、门店老板关注公众号。  2、点击公众号中的绑定公司的菜单。  3、填写老板公司编号以及老板的手机号码。  4、绑定成功并更新相应的DB下Staff表的老板openid。 |
|  | F\_Unionid | VARCHAR (100) |  | 用于将公众号绑定到微信开放平台帐号 |

前端输入密码时，格式为：长度[6, 16]，首尾不能有空格，不能包含中文。

#### T\_Permission（操作权限表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 操作ID | F\_ID |  |  |  |
| 操作名称 | F\_Name |  |  | 显示给用户看的中文名称 |
| 领域名称 | F\_Domain | Varchar(16) |  | 比如采购就是一个领域名称，又如商品档案、商品类别也是 |
| SP | F\_SP | Varchar(32) |  | 相应的SP的名称。系统内的每一个操作都会对应一个SP。  漏洞：如果SPA有call到SPB，则只判断用户对SPA的权限，不会检查对SPB的权限。 |
| 备注 | F\_Remark | Varchar(32) not null | [1,32] |  |
| 创建时间 | F\_CreateDatetime | NOT NULL |  | 便于追踪 |
| 修改时间 | F\_UpdateDatetime | Datetime, NOT NULL |  | 最近修改时间。便于追踪 |

#### T\_Role（角色表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 角色ID | F\_ID |  |  |  |
| 角色名称 | F\_Name |  |  |  |
| 创建时间 | F\_CreateDatetime | NOT NULL |  | 便于追踪 |
| 修改时间 | F\_UpdateDatetime | Datetime, NOT NULL |  | 最近修改时间。便于追踪 |

#### T\_Role\_Permission（角色和操作的中间表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 角色ID | F\_RoleID |  |  | 外键 |
| 操作ID | F\_PermissionID |  |  | 外键 |
| 创建时间 | F\_CreateDatetime | NOT NULL |  | 便于追踪 |
| 修改时间 | F\_UpdateDatetime | Datetime, NOT NULL |  | 最近修改时间。便于追踪 |

#### T\_Department（用户组表，即部门表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 用户组ID | F\_ID |  |  |  |
| 用户组名称 | F\_DepartmentName |  |  | 采购组、收银员组、老板组 |
| 创建时间 | F\_CreateDatetime | NOT NULL |  | 便于追踪 |
| 修改时间 | F\_UpdateDatetime | Datetime, NOT NULL |  | 最近修改时间。便于追踪 |

#### T\_StaffRole（用户角色表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 用户ID | F\_StaffID |  |  | 外键 |
| 角色ID | F\_RoleID |  |  | 外键 |
| 创建时间 | F\_CreateDatetime | NOT NULL |  | 便于追踪 |
| 修改时间 | F\_UpdateDatetime | Datetime, NOT NULL |  | 最近修改时间。便于追踪 |

### 消息

#### 消息表T\_Message

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 类别 | F\_CategoryID | Int |  | 不同的业务生成不同的消息。不同的消息可能需要跳转到不同的模块当中去处理。外键，关联到T\_MessageType的F\_ID |
| 公司ID | F\_CompanyID | Int not null |  | 外键。标识这条消息属于哪个公司 |
| 消息状态 | F\_Status | Int not null default 0 | 0 | 0=未发送，1=已发送。发送指发送到公众号给老板们审阅 |
| 已读 | F\_IsRead |  | 0 | 1或0。一个消息可以标为已读或未读 |
| 参数 | F\_Parameter | Varchar(255),not null，去掉空格后不能为空 |  | JSON格式的字符串，描述输入的参数，以便填写到这种类别的消息模板（见消息处理配置表）当中 |
| 发送用户ID | F\_SenderID | Int | 0 | 0表示系统发送，否则为用户发送，值为用户ID |
| 接收用户ID | F\_ReceiverID | int |  | 真实用户ID。即便是群发，都要为每一人插入一条消息 |
| 创建时间 | F\_CreateDatetime | Datetime not null | Now() |  |
| 修改时间 | F\_CreateDatetime | Datetime not null | Now() |  |

#### 消息处理配置表T\_MessageHandlerSetting

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 类别ID | F\_CategoryID | Int |  | 不同的业务生成不同的消息。不同的消息可能需要跳转到不同的模块当中去处理  关联到T\_MessageType的F\_ID |
| 类别模板 | F\_Template | VARCHAR(128) NOT NULL  去掉空格后不能为空。 |  | 举例：采购订单%s已经生成，点击此处审批之。  然则通过修改本模板可以达到修改消息内容的目的。 |
| 链接 | F\_Link | VARCHAR(255) NOT NULL  去掉空格后不能为空。 |  | 举例：<http://www.bxit.vip/shop/order/ID/5> |
|  |  |  |  |  |

其中，类别模板项参见[消息生成和处理流程](#_消息生成和处理流程)。

#### 消息类别表T\_MessageCategory

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 类别名称 | F\_Name | VARCHAR(10) NOT NULL |  |  |
|  |  |  |  |  |

本表属配置性数据。

其数据如下：

|  |  |
| --- | --- |
| ID | 类别名称 |
| XXXXX | XXXXX |
| XXXXX | 待定 |
| XXXXX | XXXXX |

#### 消息数据项T\_MessageItem

有的消息，比如滞销消息，需要记录哪些商品滞销。本表用于存储这种数据。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 对应的消息ID | F\_MessageID | >0的整数，int not null |  | 外键，索引到T\_Message.F\_ID |
| 对应的消息类别 | F\_MessageCategoryID | >0的整数，int not null |  | 外键，索引到T\_MessageCategory.F\_ID。  插入滞销的数据项时，会先将之前的所有数据删除，即本表中最多只会存在一个消息的数据项，不会存在多个消息的数据项。 |
| 数据项 | F\_CommodityID | >0的整数，int not null |  | 外键，对滞销消息来说，是T\_Commodity.F\_ID，即商品ID。  插入时，如果检查到消息类别是滞销，则需要检查此ID对应的商品是否存在（状态是什么都无所谓） |
|  | F\_CreateDatetime |  |  |  |
|  | F\_UpdateDatetime |  |  |  |

对于滞销消息，

前端可能需要返回条形码、商品名称等，在SP中返回即可。

### 博昕客户（公司）信息及其门店

以下表在POS机上需要严格保密，否则黑客有可能利用它对云服务器的存储空间和CPU利用率形成重大威胁！

#### （公共DB）博昕客户信息表（公司表）T\_Company

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 公司名称 | F\_Name | Varchar(32) not null，中英文，(0, 12]字符unique key, |  |  |
| 公司编号 | F\_SN | Varchar(8)，Unique key，6~8位数字，从668866开始 |  | 实战时，不断加1 |
| 状态 | F\_Status | int |  | 0=正常营业的Open，1=停业的Closed |
|  | F\_BusinessScope | Varchar(24), not null，任何字符，长度>0 |  |  |
| 营业执照号 | F\_BusinessLicenseSN | Varchar(18) NOT NULL，unique key，15或18位，纯阿拉伯数字或纯大写字母或它们混合 |  |  |
| 营业执照图片 | F\_BusinessLicensePicture | Varchar(128) null unique key, |  |  |
| 公司招牌图片 | F\_Logo | Varchar(128) null unique key, |  |  |
| ~~商户名称~~ | ~~F\_MchName~~ | ~~varchar(36) null default null，不能为空且长度不超过12个汉字或24个英文字母。~~ |  | ~~在卡券页面必须填写。在创建公司时，不用填写~~ |
| ~~商户WX LOGO的URL~~ | ~~F\_LogoWXURL~~ | ~~varchar(128) null default null~~ |  | ~~在卡券页面必须填写。在创建公司时，不用填写~~ |
| 老板名字 | F\_BossName | Varchar(12) not null, 中英文，(0, 12]字符 |  |  |
| 老板手机 | F\_BossPhone | Varchar(32) not null,11位数字组成，第1个数字必须为1 |  | 一个老板可以有多个公司 |
| 老板登录密码 | F\_BossPassword | Varchar(16) not null，6到16位 |  | 明文存储。但插入T\_Staff表时是加密后的盐值。老板用手机和这个密码登录成功后将被强制修改密码 |
| 老板微信号 | F\_BossWechat | Varchar(20) not null, (0, 20] |  | 一个老板可以有多个公司 |
| DB名称 | F\_DBName | Varchar(20) not null, unique key,  数字、字母和下划线的组合，但首字符必须是字母，中间不能出现空格 |  | 此公司的数据库的名称。此名称可用于针对该公司的DB的jdbc连接。DB名称提前绝对不能以数字开头 |
| 钥匙 | F\_Key | unique key,varchar(32)，32位数字和大写字母组成 |  | 加盐存储。用于身份识别，形如：B1AFC07474C37C5AEC4199ED28E09705 |
| DB用户名 | F\_DBUserName | Varchar(20) not null,  数字、字母和下划线的组合，但首字符必须是字母，不能出现空格 |  |  |
| DB密码 | F\_DBUserPassword | Varchar(16) not null,最多16位字符，中间可以有空格 |  | 先用明文存储 |
| ~~公众号AppID~~ | ~~F\_AuthorizerAppid~~ | ~~VARCHAR(32) NULL~~ |  | ~~用户有公众号的情况下，授权给BX，BX会收到WX服务器的审核推送消息，如果审核成功，会设置本字段及F\_AuthorizerRefreshToken、F\_FuncInfo~~ |
| ~~接口调用凭据刷新令牌~~ | ~~F\_AuthorizerRefreshToken~~ | ~~VARCHAR(64) NULL~~ |  | ~~在授权的公众号具备API权限时，才有此返回值~~ |
| ~~公众号权限集列表~~ | ~~F\_FuncInfo~~ | ~~BIGINT NULL~~ |  | ~~1位代表一种权限。0到63位分别代表的权限是：xxxx~~ |
| 子商户号 | F\_Submchid | 0或10位数字 |  | 收钱的子商户号，和F\_MerchantID不同。后者是子商户的身份证！不要混淆！ |
| 失效时间 | F\_ExpireDatetime | Datetime not null |  | 失效后，其所有帐号将会被锁定 |
| 创建时间 | F\_CreateDatetime | Datetime not null |  |  |
| 修改时间 | F\_UpdateDatetime | Datetime not null |  |  |
|  | ~~F\_info~~  Model名：info | ~~JSON结构~~ |  |  |
|  | ~~F\_MerchantID~~ | ~~varchar(36) NULL~~ |  | ~~子商户的身份证。和F\_Submchid不同，本字段不是用来收钱的！~~  ~~商家无公众号的情况下，通过调用我们的接口WxSubMchAction.createEx()来授权给BX。当BX收到WX服务器的推送消息后，可以知道授权是否成功。如果成功，则设置本字段的值。当然，在商家有公众号的情况下，他也可以通过调用接口WxSubMchAction.createEx()来授权给BX，只是麻烦一点，需要填写授权函。~~  ~~如果微信的推送内容是审核不通过的话，可以设置本字段为NULL？？~~ |
| ~~是否启用会员系统~~ | ~~F\_EnableVipSystem~~ | ~~Int not null~~ | ~~0~~ | ~~1=启用，0=不启用~~  ~~并非每个商家都需要会员系统，因而不一定需要在NBR让他们看到会员系统~~ |
| 是否显示会员系统的提示给商家老板看 | F\_ShowVipSystemTip | Int not null | 0 | 1=要提示，0=不需要提示（老板已经点击过诸如“下次不再提示”的按钮）。 |
| 商家招牌 | F\_BrandName | varchar(20) not null，不能为空串 |  | 如“娃哈哈” |

判断一个公司是否已经授权给BX，可以这样判断

If (length(F\_AuthorizerAppid) > 0 || length(F\_MerchantID) > 0) {

已经授权

} else {

未授权

}

#### ~~子商户信息表~~~~T\_SubMch~~

~~本表为私有DB表。~~

~~Mch=merchant~~

~~认证的作用：使用更多代制功能，使用卡券功能。~~

1. ~~商家有公众号，分未认证、已认证两种情况。~~
   1. ~~已认证：扫码？填写APP ID，直接授权。~~
   2. ~~未认证：填写APP ID。强授权：三方平台、全网发布、扫码授权，后可进行卡券。商家帮忙做。~~
2. ~~商家无公众号，使用授权函授权。传递merchant\_id，代表代制，否则会被WX认为是自制。~~

~~互参：~~[~~代制授权流程~~](#_Xxxx)~~。~~

~~微信将认证结果发回给nbr，nbr需要更新此表，再回应给微信服务器。如果nbr关闭，没收到微信服务器的认证结果，怎么办？~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
|  | ~~F\_ID~~ |  |  |  |
|  | ~~F\_MerchantID~~  ~~Model名：merchant\_id~~ | ~~VARCHAR(36) NULL~~ |  | ~~创建时是空的，创建后WX才返回值以更新本表。子商户的身份证，不是用来收钱的~~ |
|  | ~~F\_CompanyID~~ |  |  | ~~外键~~ |
| ~~子商户的公众号app\_id~~ | ~~F\_AppID NULL~~  ~~Model名：app\_id~~ | ~~varchar(36)~~ |  | ~~子商户的公众号app\_id，配置后子商户卡券券面上的app\_id为该app\_id。注意：该app\_id须经过认证~~ |
| ~~子商户名称~~ | ~~F\_BrandName NOT NULL（12个汉字内）~~  ~~Model名：brand\_name~~ | ~~varchar(12)~~ |  | ~~子商户名称（12个汉字内），该名称将在制券时填入并显示在卡券页面上~~ |
| ~~子商户logo~~ | ~~F\_LogoUrl NOT NULL~~  ~~Model名：logo\_url~~ | ~~varchar(128)~~ |  | ~~子商户logo，可通过 上传图片接口 获取。该logo将在制券时填入并显示在卡券页面上~~ |
| ~~授权函ID~~ | ~~F\_Protocol NOT NULL~~ | ~~varchar(64)~~ |  | ~~授权函ID，即通过 上传临时素材接口 上传授权函后获得的meida\_id~~ |
| ~~授权函有效期截止时间~~ | ~~F\_EndTime NOT NULL~~  ~~Model名：end\_time~~ | ~~BIGINT unsigned~~ |  | ~~授权函有效期截止时间（东八区时间，单位为秒），需要与提交的扫描件一致~~ |
| ~~一级类目id,~~ | ~~F\_WxPrimaryCategoryID~~  ~~Model名：~~  ~~primary\_category\_id~~ | ~~外键~~ |  | ~~一级类目id~~ |
| ~~二级类目id~~ | ~~F\_WxSecondaryCategoryID~~  ~~Model名：~~  ~~secondary\_category\_id~~ | ~~外键~~ |  | ~~二级类目id~~ |
| ~~营业执照或个体工商户营业执照彩照或扫描件~~ | ~~F\_AgreementMediaID NULL~~  ~~Model名：~~  ~~agreement\_media\_id~~ | ~~varchar(64)~~ |  | ~~营业执照或个体工商户营业执照彩照或扫描件~~ |
| ~~营业执照内登记的经营者身份证彩照或扫描件~~ | ~~F\_OperatorMediaID NULL~~  ~~Model名：operator\_media\_id~~ | ~~varchar(64)~~ |  | ~~营业执照内登记的经营者身份证彩照或扫描件~~ |
| ~~子商户审核状态~~ | ~~F\_Status~~ | ~~Int~~ |  | ~~0=审核中, 1=已通过，2=被驳回，3=协议已过期~~ |
|  | ~~Model名：~~  ~~info~~ |  |  |  |
|  | ~~F\_ApproveDesc~~ | ~~varchar(50)~~ |  | ~~记录审核描述~~ |

#### ****~~子商户一级类目表 T\_WxPrimaryCategory~~****

~~本表为私有DB表。~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
|  | ~~F\_ID~~ |  |  |  |
| ~~一级目录id~~ | ~~F\_PrimaryCategoryId~~  ~~Model名：~~  ~~primary\_category\_id~~ | ~~int~~ |  | ~~一级目录id~~ |
| ~~一级目录名称~~ | ~~F\_CategoryName~~  ~~Model名：~~  ~~category\_name~~ | ~~varchar(30)~~ |  | ~~一级目录名称~~ |

#### ****~~子商户二级类目表 T\_WxSecondaryCategory~~****

~~本表为私有DB表。~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
|  | ~~F\_ID~~ |  |  |  |
|  | ~~F\_ PrimaryCategoryID~~ |  |  | ~~一级类目外键~~ |
| ~~二级目录id~~ | ~~F\_SecondaryCategoryId~~  ~~Model名：~~  ~~secondary\_category\_id~~ | ~~int~~ |  | ~~二级目录id~~ |
| ~~二级目录名称~~ | ~~F\_CategoryName~~  ~~Model名： category\_name~~ | ~~varchar(40)~~ |  | ~~二级目录名称~~ |

#### 门店信息表T\_Shop

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 名称 | F\_Name | Varchar(20) NOT NULL。长度[1,20]，unique key，首尾不能包含空格 |  |  |
| 博昕客户ID（公司ID） | F\_CompanyID | Int not null |  | 外键，对应公共DB表T\_Company的F\_ID |
| 博昕业务员ID | F\_BXStaffID | Int not null |  | 外键，对应公共DB表T\_BXStaff的F\_ID |
| 地址 | F\_Address | VARCHAR(30) NOT NULL，中间可以有空格，但首尾不能有空格 |  |  |
| 区域ID | F\_DistrictID | 正整数 |  | 外键 |
| 状态 | F\_Status | Int not null default 0 |  | 参看门店状态图  0=营业中，1=离线中，2=断网锁定，3=欠费锁定，4=财务待审核 |
| 经度 | F\_Longitude | Decimal(20,6) NOT NULL |  |  |
| 纬度 | F\_Latitude | Decimal(20,6) NOT NULL |  |  |
| 钥匙 | F\_Key | VARCHAR(32) NOT NULL |  | 加盐存储。用于身份识别 |
| 备注 | F\_Remark | Varchar(128) null，任意非空字符 |  |  |
| 创建时间 | F\_CreateDatetime | Datetime not null |  |  |
| 修改时间 | F\_UpdateDatetime | Datetime not null |  |  |

#### ****~~微信分店表 T\_WxBranchShop~~****

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
|  | ~~F\_ID~~ |  |  |  |
| ~~主体JSON~~ | ~~F\_Business~~  ~~Model名:business~~ | ~~JSON结构~~ |  |  |
| ~~详细JSON~~ | ~~F\_BaseInfo~~  ~~Model名:base\_info~~ | ~~JSON结构~~ |  |  |
|  | ~~F\_ShopID~~ |  |  | ~~外键，关联T\_Shop。~~ |
| ~~商户自己的id~~ | ~~F\_ Sid NULL~~ | ~~varchar（30）~~ |  | ~~商户自己的id，用于后续审核通过收到poi\_id 的通知时，做对应关系。请商户自己保证唯一识别性~~ |
| ~~门店名称~~ | ~~F\_BusinessName NOT NULL~~  ~~Model名:business\_name~~ | ~~varchar（30）~~ |  | ~~门店名称（仅为商户名，如：国美、麦当劳，不应包含地区、地址、分店名等信息，错误示例：北京国美） 不能为空，15个汉字或30个英文字符内~~ |
| ~~分店名称~~ | ~~F\_BranchName NOT NULL~~  ~~Model名:branch\_name~~ | ~~varchar（20）~~ |  | ~~分店名称（不应包含地区信息，不应与门店名有重复，错误示例：北京王府井店） 20 个字 以内~~ |
| ~~门店所在的省份~~ | ~~F\_Province NOT NULL~~ | ~~varchar（10）~~ |  | ~~门店所在的省份（直辖市填城市名,如：北京 市） 10个字 以内~~ |
| ~~门店所在的城市~~ | ~~F\_City NOT NULL~~ | ~~varchar（10）~~ |  | ~~门店所在的城市 10个字 以内~~ |
| ~~门店所在地区~~ | ~~F\_District NOT NULL~~ | ~~varchar（10）~~ |  | ~~门店所在地区 10个字 以内~~ |
| ~~门店所在的详细街道地址~~ | ~~F\_Address NOT NULL~~ | ~~varchar（30）~~ |  | ~~门店所在的详细街道地址（不要填写省市信息）（东莞等没有“区”行政区划的城市，该字段可不必填写。其余城市必填。）~~ |
| ~~门店的电话~~ | ~~F\_Telephone NOT NULL~~ | ~~varchar（16）~~ |  | ~~门店的电话（纯数字，区号、分机号均由“-”隔开）~~ |
| ~~门店的类型~~ | ~~F\_Categories NOT NULL~~ | ~~varchar（16）~~ |  | ~~门店的类型（不同级分类用“,”隔开，如：美食，川菜，火锅。详细分类参见附件：微信门店类目表）~~ |
| ~~坐标类型~~ | ~~F\_OffsetType NOT NULL~~  ~~Model名:offset\_type~~ | ~~varchar（2）~~ |  | ~~坐标类型： 1 为火星坐标 2 为sogou经纬度 3 为百度经纬度 4 为mapbar经纬度 5 为GPS坐标 6 为sogou墨卡托坐标 注：高德经纬度无需转换可直接使用~~ |
| ~~门店所在地理位置的经度~~ | ~~F\_Longitude NOT NULL~~ | ~~varchar（16）~~ |  | ~~门店所在地理位置的经度~~ |
| ~~门店所在地理位置的纬度~~ | ~~F\_Latitude NOT NULL~~ | ~~varchar（16）~~ |  | ~~门店所在地理位置的纬度（经纬度均为火星坐标，最好选用腾讯地图标记的坐标）~~ |
| ~~图片列表~~ | ~~F\_PhotoList NULL~~  ~~Model名:photo\_list~~ | ~~Array ???~~ |  | ~~图片列表，url 形式，可以有多张图片，尺寸为 640\*340px。必须为上一接口生成的url。 图片内容不允许与门店不相关，不允许为二维码、员工合照（或模特肖像）、营业执照、无门店正门的街景、地图截图、公交地铁站牌、菜单截图等~~ |
| ~~推荐品~~ | ~~F\_Recommend NULL~~ | ~~varchar(200)~~ |  | ~~推荐品，餐厅可为推荐菜；酒店为推荐套房；景点为推荐游玩景点等，针对自己行业的推荐内容 200字以内~~ |
| ~~特色服务~~ | ~~F\_Special NULL~~ | ~~varchar(60)~~ |  | ~~特色服务，如免费wifi，免费停车，送货上门等商户能提供的特色功能或服务~~ |
| ~~商户简介~~ | ~~F\_Introduction NULL~~ | ~~varchar(300)~~ |  | ~~商户简介，主要介绍商户信息等 300字以内~~ |
| ~~营业时间~~ | ~~F\_OpeTime NULL~~  ~~Model名:open\_time~~ | ~~varchar(30)~~ |  | ~~营业时间，24 小时制表示，用“-”连接，如 8:00-20:00~~ |
| ~~人均价格~~ | ~~F\_AvgPrice NULL~~  ~~Model名：avg\_price~~ | ~~Int~~ |  | ~~人均价格，大于0 的整数~~ |

#### 门店区域表T\_ShopDistrict

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 区域名字 | F\_Name | NOT NULL，长度[0, 20] |  |  |

#### POS机信息T\_POS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 状态 | F\_Status | Int not null | 0 | 0=可用，1=不可用（已删除或已坏掉不再工作）。  收银汇总表等，其它表对本表有外键依赖，所以必须增加本字段 |
| SN | F\_POS\_SN | Varchar(32),not null，目前看到的都是由英文字母和数字组成 |  | OP会填写真正的POS机的SN，不会乱填。 |
| 门店ID | F\_ShopID |  |  |  |
| 公钥加密后的用户密码 | F\_pwdEncrypted | Varchar(0) |  | 用公钥加密后的用户密码。这个字段只会在Model中使用，和数据库完全无关，所以定义为Varchar(0) |
| 加盐后的MD5值 | F\_Salt | Varchar(32) |  | 本字段在POS端是空串，在服务器端非空 |
| 在POS机上的密码 | F\_PasswordInPOS | Varchar(16) |  | POS将使用这个密码登录服务器。服务器上并不存储这个密码，则DB中此字段值为空串。  这个密码是部署的时候设置的。服务器将使用这个密码生成盐值并放在F\_Salt字段。在POS身份验证成功后，这个字段将置为NULL。这可以防止一些黑客行为：  1、黑客冒充已有的POS机，进行交付POS的流程。  2、黑客用一个不存在的POS，向服务器请求POS的身份证和公司信息。  以上2种情况下，在ACTION或SP中如果检查到F\_PasswordInPOS为NULL，认定为黑客行为，ACTION层return null，SP层返回黑客错误码。 |

### （公共DB）博昕内部员工表

#### 博昕内部员工表T\_BXStaff

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 手机号码 | F\_Mobile | VARCHAR(11) NOT NULL |  |  |
| 公钥加密后的用户密码 | F\_pwdEncrypted | VARCHAR(0) NULL |  |  |
| 密码 | F\_Salt | VARCHAR(32) NOT NULL |  | 解密后的密码6到16位，不能包含中文，首尾不能有空格 |
| 角色ID | F\_RoleID | Int not null |  |  |
| 用户组ID | F\_DepartmentID | Int not null |  |  |
| 姓名 | F\_Name | VARCHAR(12) NOT NULL, |  |  |
| 性别 | F\_Sex | Int not null |  |  |
| 身份证号码 | F\_ICID | VARCHAR(20) NOT NULL |  |  |
|  |  |  |  |  |

### 会员

#### 会员表T\_Vip

~~注意参看会员单元表：T\_VipUnit~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  | 会员在后台上创建的话，会生成这个ID |
| 会员编号 | F\_SN | Varchar(9) not null |  | 生成方式为VIP000001 ~ VIP999999 的流水号 |
| 会员卡 | F\_CardID | int not null |  | 外键，指向T\_VipCard，某种类型的会员卡的唯一标识 |
| ~~会员在POS机上的ID~~ | ~~F\_IDInPOS~~  ~~F\_LocalPosID~~ | ~~INT NULL~~ |  | ~~会员在POS机上创建的话，会生成这个ID~~ |
| POS机的SN码 | ~~F\_POS\_SN~~  F\_LocalPosSN | VARCHAR(32) NULL |  | 会员在POS机上创建的话，会记录这个POS机的SN |
| 性别 | F\_Sex | Int not null default 1 |  | 0=女，1=男，2=未知。不能为其它值 |
|  | F\_Logo |  |  |  |
| ~~证件类型~~ | ~~F\_CertType~~ | ~~Int not NULL~~ |  | ~~应该固定范围??~~ |
| ~~证件编号~~ | ~~F\_CertNO~~ | ~~Varchar(18)？~~ |  | ~~??~~ |
| ICID | F\_ICID |  |  | 字段惟一 |
| 手机 | F\_Mobile | Not NULL, Varchar(11), unique | NULL | 字段惟一 |
| ~~微信号~~ | ~~F\_Wechat~~ |  |  | ~~字段惟一~~ |
| ~~支付宝号~~ | ~~F\_Alipay~~ |  |  | ~~字段惟一~~ |
| ~~QQ~~ | ~~F\_QQ~~ |  |  | ~~字段惟一~~ |
| Name | F\_Name | 中英文，[2, 32]字符。Varchar(32), NOT NULL |  |  |
| Email | F\_Email | Varchar(30) null |  | 字段惟一 |
| ~~登录帐号~~ | ~~F\_Account~~ |  |  | ~~字段惟一~~ |
| ~~登录密码~~ | ~~F\_Password~~ | ~~F\_Password VARCHAR(32) NULL default 000000~~ |  | ~~明文密码不能含有中文，6到16位，首尾不能为空。加密后的密码则固定是32位大写字母和数字的组合。如果inserttable.sql中是明文，要改成密文~~ |
| 总消费次数 | F\_ConsumeTimes |  |  |  |
| 总消费金额 | F\_ConsumeAmount |  |  |  |
| 区域 | F\_District |  |  |  |
| 类别 | F\_Category | Int Default 0 |  | 先不做会员级别 |
| ~~状态~~ | ~~F\_Status~~ |  |  | ~~0=正常，1=挂失，2=已经删除。可能考虑会员卡过期的状态为2~~ |
| 生日 | F\_Birthday |  |  |  |
| ~~会员期限~~ | ~~F\_ExpireDatetime~~ |  |  | ~~微信会员卡有期限。支付宝会员也有期限？~~  ~~这个字段可能会删除，改为采用WX及Alipay各自的期限~~ |
| 当前积分 | F\_~~Point~~Bonus |  | 0 | ~~属于WX的积分，还是支付宝的积分，还是二者的和，还是和二者都一样？和二者都一样的情况下，每次用户消费WX5元，都会分别加积分到WX和支付宝~~ |
| 上次消费日期时间 | F\_LastConsumeDatetime |  |  |  |
| 创建时间 | F\_CreateDatetime | DATETIME NOT NULL DEFAULT now() |  | 即入会时间 |
| 修改时间 | F\_UpdateDatetime | DATETIME NOT NULL DEFAULT now() |  |  |
|  | F\_Remark | Varchar(50) null |  |  |

#### 会员来源表T\_VipSource

如果顾客既拿WX消费，又拿支付宝消费，则本表有2行数据的F\_VipID是一样的。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
|  | F\_VipID | Int not null |  | 外键，指向T\_Vip.F\_ID  与F\_SourceCode组合成惟一键 |
|  | F\_SourceCode | Int not null |  | {0=WX, 1=ALIPAY, 2=MEITUAN}  与F\_ VipID组合成惟一键 |
|  | F\_ID1 | VARCHAR(100) Not null |  | 若F\_SourceCode=WX，它代表微信公众号的openID  若F\_SourceCode= ALIPAY，它代表支付宝的openID |
|  | F\_ID2 | VARCHAR(100) Not null |  | 若F\_SourceCode=WX，它代表微信的unionID  若F\_SourceCode= ALIPAY，它代表支付宝的的unionID |
|  | F\_ID3 | VARCHAR(100) Not null |  | 若F\_SourceCode=WX，它代表微信的MiniProgramOpenID  若F\_SourceCode= ALIPAY，它代表支付宝的的MiniProgramOpenID |
|  |  |  |  |  |

#### 会员卡表

##### 一种会员卡T\_VipCard

创建公司时，令本表有且只有一行数据，给予默认值。在PrivateDBConfig.sql中插入数据。老板可以修改会员卡的以下字段：名称、背景图、积分清零规则。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| Wx会员卡ID | F\_ID | INT NOT NULL | 1 | PRIMARY KEY，自增1。  某张会员卡的唯一标识 |
| 名称 | F\_Title | varchar（9）NOT NULL，不能为空，[2,9]中英数 | 会员卡 | 字数上限为9个汉字。(建议涵盖卡券属性、服务及金额)。 |
| 会员卡背景颜色图 | F\_BackgroundColor | Varchar(23) not null  格式：2个RGB串以,连接，中间以;隔开 | 255,255,255; 255,255,255 | 商家自定义会员卡背景颜色（渐变）。 |
| ~~积分清零规则~~ | ~~F\_BonusCleared~~ | ~~Not NULL，varchar(128)，可空~~ | ~~每年元旦积分清零~~ | ~~积分清零规则文字描述：每年元旦积分清零~~ |
| 积分清零天数 | F\_ClearBonusDay | Int null |  | 领卡时间起计算，N天后系统通过夜间任务自动清零。F\_ClearDay和F\_ClearDatetime必须填写一项 |
| 积分清零日期 | F\_ClearBonusDatetime | Datetime null |  | 到了这个日期后，系统通过夜间任务自动清零。F\_ClearDay和F\_ClearDatetime必须填写一项 |
| WX开卡时间 | F\_CreateDatetime | DATETIME NOT NULL DEFAULT now() |  |  |

##### 一张会员卡T\_VipCardCode

哪个会员有哪张卡。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| F\_ID | F\_ID | INT NOT NULL |  |  |
| 会员ID | F\_VipID | INT NOT NULL |  | 外键，指向T\_Vip.F\_ID |
| 会员卡ID | F\_VipCardID | INT NOT NULL |  | 外键，指向T\_VipCard.F\_ID |
|  | F\_SN | Varchar(16) not null，16个数字合成 |  | 左边6位用公司编号填充，右面10位，随机生成。  公司范围内，不能重复 |
| 创建时间 | F\_CreateDatetime |  |  | 创建时间即领卡时间 |

#### 积分表

##### 积分规则表T\_BonusRule

描述一种会员卡，是如何积分的。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID | INT NOT NULL |  |  |
|  | F\_VipCardID | Int not null |  | 外键，指向T\_VipCardID.F\_ID |
| 消费金额 | F\_AmountUnit | Not NULL，int，>0 |  | 消费金额。以分为单位 |
| 增加的积分 | F\_IncreaseBonus | Not NULL，int，>0 |  | 对应增加的积分 |
| 单次可获取的积分上限 | F\_MaxIncreaseBonus | Not NULL，int,default 2^32-1，>0 |  | 用户单次可获取的积分上限 |
| 初始设置积分 | F\_InitIncreaseBonus | Not NULL，int, default 0，>=0 |  | 初始设置积分 |
| ~~每使用多少积分~~ | ~~F\_CostBonusUnit~~  ~~Model名:cost\_bonus\_unit~~ | ~~NULL，int~~ |  | ~~每使用5积分~~ |
| ~~抵扣xx元~~ | ~~F\_ReduceMoney~~  ~~Model名:reduce\_money~~ | ~~NULL，int~~ |  | ~~抵扣xx元，（这里以分为单位）~~  ~~Nbr需要判断业务逻辑，WX不管的~~ |
| ~~抵扣条件~~ | ~~F\_LeastMoneyToUseBonus~~  ~~Model名: least\_money\_to\_use\_bonus~~ | ~~NULL，int~~ |  | ~~抵扣条件，满xx元（这里以分为单位）可用~~ |
| ~~抵扣条件~~ | ~~F\_MaxReduceBonus~~  ~~Model名: max\_reduce\_bonus~~ | ~~NULL，int~~ |  | ~~抵扣条件，单笔最多使用xx积分。~~  ~~Nbr需要判断业务逻辑，WX不管的~~ |

##### 积分历史表T\_BonusConsumeHistory

积分支持手动修改。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| F\_ID | F\_ID | INT NOT NULL |  | PRIMARY KEY，自增1。  某个积分使用详情的唯一标识。 |
|  | F\_VipID |  |  | 外键，指向T\_Vip.F\_ID |
|  | F\_StaffID | Int null | Null | 谁动了积分 |
| 积分全量值 | F\_Bonus | NULL，int |  | 需要设置的积分全量值，传入的数值会直接显示。传入的数值应是积分变动后的值。 |
| 积分变动值 | F\_AddedBonus | NULL，int |  | 本次积分变动值，传负数代表减少。用于记录每次积分变动的情况。  退货时，已使用的积分不退回，但是产生的积分需要扣除 |
| 备注 | F\_Remark | NULL，varchar(48) |  | 商家说明积分变动的情况。如果是系统自动增减，则本字段为空。举例：因促销活动自相矛盾，手动帮会员增加10积分 |
|  | F\_CreateDatetime | Datetime not null |  |  |

#### 会员类别表~~T\_VIP\_Category~~T\_VipCategory暂时不做

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 类别名称 | F\_Name | VARCHAR(30) NOT NULL，中英文和数字的组合 |  |  |

#### 优惠券

##### ****一种优惠券表T\_Coupon****

一张零售单最多可以使用多少种券的多少张？2个做法：

1. 一张零售单最多允许使用一张券。但是这可能导致出现顾客买A和B，它们分别对应2张不同种类的优惠券，顾客必须分2次买单才可以同时得到A和B的优惠。
2. 一张零售单不限制使用券，规定有代金券则先用代金券，然后才到折扣券。而折扣券只能用一张。虽然这样规定，仍然有点混乱。
3. 一张零售单不限制使用券，但是只要有代金券，就不准再用折扣券。折扣券最多只能用一张。

目前采用1。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 序号 | F\_ID |  |  |  |
| 状态 | F\_Status | not NULL int |  | {0=正常，1=已终止}  已终止代表不能再领取。已终止是老板在web界面上终止了优惠券导致的。这时，即使未过有效期，该优惠券仍然无效。但是，顾客已经领取的券可以继续使用。 |
| 优惠券类型 | F\_Type | Int not null |  | 优惠券类型。  0=CASH，1=DISCOUNT |
| 需要多少积分兑换 | F\_Bonus | Int not null, >=0 |  | >0，需要积分兑换顾客才可以领取。=0，不需要积分领取 |
| 起用金额 | F\_LeastAmount | Decimal(20, 6) not null default 0 | 0 | 代金券专用，表示起用金额（单位为元）,如果无起用门槛则填0。 |
| 减免金额 | F\_ReduceAmount | Decimal(20, 6) not null | 0 | 代金券专用，表示减免金额。（单位为元），必须>0.00。减免金额必须小于起用金额 |
| 打折额度 | F\_Discount | Decimal(20, 6) not null  范围(0.00, 1.00) | 1.00 | 折扣券专用，表示打折额度。0.88就是指88折 |
| 名称 | F\_Title | varchar（9）NOT NULL，中英数，不能有空格或其它符号 |  | UI提示：建议涵盖卡券属性、服务及金额 |
| 颜色 | F\_Color | varchar（16）NOT NULL |  | 券颜色。按色彩规范标注填写Color010-Color100 |
| 使用说明 | F\_Description | varchar（1024）NOT NULL |  | 券使用说明，字数上限为1024个汉字。 |
| 每人可领券的数量限制 | F\_PersonalLimit | Int not NULL | 1 | 每人可领券的数量限制>=1。不填写默认为1。 |
| ~~指定可用的商品类目~~ | ~~F\_AcceptCategory~~ | ~~Int null~~ |  | ~~指定可用的商品类目，仅用于代金券类型 ，填入后将在券面拼写适用于xxx。~~ |
| ~~指定不可用的商品类目~~ | ~~F\_RejectCategory~~ | ~~Int null~~ |  | ~~指定不可用的商品类目，仅用于代金券类型 ，填入后将在券面拼写不适用于xxxx。~~ |
| 可用星期 | F\_WeekDayAvailable | Int not null | 0 | 例1：~~F\_ WeekDayAvailable =6=B110代表周二和周三可以使用。~~  F\_ WeekDayAvailable =6=B00011000代表周二和周三可以使用。  例2：~~F\_ WeekDayAvailable =0=B1111110，代表周二到周日可以使用。~~  F\_ WeekDayAvailable =0=B01111110，代表周二到周日可以使用。  F\_ WeekDayAvailable =0代表没有星期限制。 |
| 当天起始时间 | F\_BeginTime | Varchar(8) not null | 00:00:00 | 前端传递空串代表00:00:00 |
| 当天结束时间 | F\_EndTime | Varchar(8) not null | 23:59:59 | 前端传递空串代表23:59:59 |
| 起用日期时间 | F\_BeginDateTime | Datetime not null |  |  |
| 结束日期时间 | F\_EndDateTime | Datetime not null |  |  |
| 库存数量 | F\_Quantity | Int not null [1,10万]，所有券的库存总和必须<1亿 |  | 不可更新字段 |
| 当前剩余库存数量 | F\_ RemainingQuantity | Int not null >=0 |  |  |
| ~~失效日期~~ | ~~F\_ExpireDatetime~~ | ~~Datetime not null~~ |  |  |
| ~~可叠加其它优惠券~~ | ~~F\_Can~~ |  |  |  |
| 参与的商品的范围 | F\_Scope | Int not null | 0 | 0=全部商品，1=部分商品（这些商品记录在T\_CouponScope中） |

~~示例：~~

~~1、有优惠券A可叠加，B可叠加，C不可叠加，D不可叠加~~

~~一个零售单最多可以用1张A或1张B。CD随意。~~

~~不可以用2张A。CD随意。~~

~~不可以用2张B。CD随意。~~

~~2、有优惠券A可叠加，C不可叠加，D不可叠加~~

~~一个零售单最多可以用1张A。CD随意。~~

1. ~~有C不可叠加，D不可叠加~~

~~CD随意。~~

1. ~~有优惠券A可叠加，B可叠加，C可叠加，D不可叠加，E不可叠加~~

~~最多可以用1张A或1张B或1张C。DE随意。~~

##### 一张优惠券表T\_CouponCode

已经领取的，才会在此表中插入一行数据。

一个零售单可以使用同一种优惠券的多张。至于是否可以使用多种的一张或多种的多张，则要看优惠券的规则。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 序号 | F\_ID |  |  |  |
|  | F\_VipID | Int not null |  | 外键，指向T\_Vip.F\_ID |
|  | F\_CouponID | Int not null |  | 外键，指向T\_Coupon.F\_ID |
| 状态 | F\_Status | Int not null |  | 0=已领取、未核销，1=已核销 |
|  | F\_SN | Varchar(15) not null unique |  | 随机生成，但不能重复。SP中生成 |
| 创建时间 | F\_CreateDatetime | Datetime |  | 领取时间 |
| 核销时间 | F\_UsedDatetime | Datetime |  |  |

##### 优惠券作用范围表T\_CouponScope

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 优惠券ID | F\_CouponID |  |  | 外键，T\_Coupon.F\_ID |
| 商品ID | F\_CommodityID |  | NOT NULL | 限定为单品 |
| 商品名称 | F\_CommodityName | Varchar(32) not null |  | 创建指定范围的促销时，将此时的商品名称插入到本字段中保留。此后不管T\_Commodity中的名称如何变化，此处不变，即“保持历史原貌” |

##### ****~~优惠券结构表T\_WxCouponDetail~~****

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~序号~~ | ~~F\_ID~~ |  |  |  |
|  | ~~F\_~~ **~~CouponID~~** |  |  | ~~外键~~ |
| ~~优惠券类型~~ | ~~F\_CardType NOT NULL~~  ~~Model名:card\_type~~ | ~~varchar(24)~~ |  | ~~优惠券类型。暂时只有CASH，DISCOUNT~~ |
| ~~优惠券基础字段~~ | ~~F\_ BaseInfo NOT NULL~~  ~~Model名:base\_info~~ | ~~JSON结构~~ |  | ~~优惠券基础字段~~  ~~对应T\_WxCouponBaseInfo~~ |
| ~~优惠券高级字段~~ | ~~F\_AdvancedInfo NULL~~  ~~Model名:advanced\_info~~ | ~~JSON结构~~ |  | ~~优惠券高级字段~~  ~~T\_WxCardAdvancedInfo~~ |
| ~~起用金额~~ | ~~F\_LeastCost NULL~~  ~~Model名:least\_cost~~ | ~~Int~~ | ~~0~~ | ~~代金券专用，表示起用金额（单位为分）,如果无起用门槛则填0。~~ |
| ~~减免金额~~ | ~~F\_ReduceCost NULL~~  ~~Model名:reduce\_cost~~ | ~~Int~~ |  | ~~代金券专用，表示减免金额。（单位为分）~~ |
| ~~打折额度~~ | ~~F\_Discount NULL~~ | ~~Int~~ | ~~0~~ | ~~折扣券专用，表示打折额度（百分比）。填30就是七折。~~ |

##### 微~~信优惠券数据 T\_WxCouponBaseInfo~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~序号~~ | ~~F\_ID~~ |  |  |  |
|  | ~~F\_ CouponDetailID~~ |  |  | ~~外键~~ |
| ~~卡券的商户logo~~ | ~~F\_LogoUrl NOT NULL~~  ~~Model名:logo\_url~~ | ~~varchar(128)~~ |  | ~~卡券的商户logo，建议像素为300\*300。即公司表的从表的子商户logo~~ |
| ~~码型~~ | ~~F\_CodeType NOT NULL~~  ~~Model名:code\_type~~ | ~~varchar(30)~~ |  | ~~码型： "CODE\_TYPE\_TEXT"文 本 ； "CODE\_TYPE\_BARCODE"一维码~~  ~~"CODE\_TYPE\_QRCODE"二维码 "CODE\_TYPE\_ONLY\_QRCODE",二维码无code显示； "CODE\_TYPE\_ONLY\_BARCODE",一维码无code显示；CODE\_TYPE\_NONE， 不显示code和条形码类型~~ |
| ~~商户名字~~ | ~~F\_BrandName NOT NULL~~  ~~Model名:brand\_name~~ | ~~varchar（12）~~ |  | ~~商户名字,字数上限为12个汉字。~~~~即公司表的从表的子商户名称~~ |
| ~~卡券名~~ | ~~F\_Title NOT NULL~~ | ~~varchar（9）~~ |  | ~~卡券名，字数上限为9个汉字。(建议涵盖卡券属性、服务及金额)。~~ |
| ~~券颜色~~ | ~~F\_Color NOT NULL~~ | ~~varchar（16）~~ |  | ~~券颜色。按色彩规范标注填写Color010-Color100。~~ |
| ~~卡券使用提醒~~ | ~~F\_Notice NOT NULL~~ | ~~varchar（16）~~ |  | ~~卡券使用提醒，字数上限为16个汉字。~~ |
| ~~卡券使用说明~~ | ~~F\_Description NOT NULL~~ | ~~varchar（1024）~~ |  | ~~卡券使用说明，字数上限为1024个汉字。~~ |
| ~~商品信息。~~ | ~~F\_Sku NOT NULL~~  ~~Model名:sku~~ | ~~JSON结构~~ |  | ~~商品信息。~~ |
| ~~Sku表ID~~ | ~~F\_SkuID~~ | ~~Int null~~ |  | ~~外键~~ |
| ~~使用日期~~ | ~~F\_DateInfo NOT NULL~~  ~~Model名:date\_info~~ | ~~JSON结构~~ |  | ~~使用日期，有效期的信息。~~ |
| ~~DateInfo表ID~~ | ~~F\_ DateInfoID~~ | ~~Int null~~ |  | ~~外键~~ |
| ~~是否自定义Code码~~ | ~~F\_UseCustomCode NULL~~  ~~Model名:use\_custom\_code~~ | ~~Int~~ |  | ~~值范围{0,1}是否自定义Code码 。填写true或false，默认为false。 通常自有优惠码系统的开发者选择 自定义Code码，并在卡券投放时带入 Code码，详情见 是否自定义Code码 。~~ |
| ~~预存code模式~~ | ~~F\_GetCustomCodeMode NULL~~  ~~Model名：get\_custom\_code\_mode~~ | ~~varchar(32)~~ |  | ~~填入 GET\_CUSTOM\_CODE\_MODE\_DEPOSIT 表示该卡券为预存code模式卡券， 须导入超过库存数目的自定义code后方可投放， 填入该字段后，quantity字段须为0,须导入code 后再增加库存~~ |
| ~~是否指定用户领取~~ | ~~F\_BindOpenid NULL~~  ~~Model名：bind\_openid~~ | ~~Int~~ |  | ~~值范围{0,1}是否指定用户领取，填写true或false 。默认为false。通常指定特殊用户群体 投放卡券或防止刷券时选择指定用户领取。~~ |
| ~~客服电话~~ | ~~F\_ServicePhone NULL~~  ~~Model名：service\_phone~~ | ~~varchar（24）~~ |  | ~~客服电话。~~ |
| ~~门店位置poiid~~ | ~~F\_LocationIdList NULL~~  ~~Model名：location\_id\_list~~ | ~~Array ???~~ |  | ~~门店位置poiid。 调用 POI门店管理接 口 获取门店位置poiid。具备线下门店 的商户为必填。~~ |
| ~~设置本卡券支持全部门店~~ | ~~UseAllLocations NULL~~  ~~Model名：use\_all\_locations~~ | ~~int~~ |  | ~~值范围{0,1}设置本卡券支持全部门店，与location\_id\_list互斥~~ |
| ~~卡券顶部居中的按钮~~ | ~~F\_CenterTitle NULL~~  ~~Model名：center\_title~~ | ~~varchar（18）~~ |  | ~~卡券顶部居中的按钮，仅在卡券状 态正常(可以核销)时显示~~ |
| ~~显示在入口下方的提示语~~ | ~~F\_CenterSubTitle NULL~~  ~~Model名：center\_sub\_title~~ | ~~varchar（24）~~ |  | ~~显示在入口下方的提示语 ，仅在卡券状态正常(可以核销)时显示。~~ |
| ~~顶部居中的url~~ | ~~F\_CenterUrl NULL~~  ~~Model名：center\_url~~ | ~~varchar（128）~~ |  | ~~顶部居中的url ，仅在卡券状态正常(可以核销)时显示。~~ |
| ~~卡券跳转的小程序的user\_name~~ | ~~F\_CenterAppBrandUserName NULL~~  ~~Model名：center\_app\_brand\_user\_name~~ | ~~varchar（128）~~ |  | ~~卡券跳转的小程序的user\_name，仅可跳转该 公众号绑定的小程序~~ |
| ~~卡券跳转的小程序的path~~ | ~~F\_CenterAppBrandPass NULL~~  ~~Model名：center\_app\_brand\_pass~~ | ~~varchar（128）~~ |  | ~~卡券跳转的小程序的path~~ |
| ~~自定义跳转外链的入口名字。~~ | ~~F\_CustomUrlName NULL~~  ~~Model名：custom\_url\_name~~ | ~~varchar（15）~~ |  | ~~自定义跳转外链的入口名字。~~ |
| ~~自定义跳转的URL。~~ | ~~F\_CustomUrl NULL~~  ~~Model名：custom\_url~~ | ~~varchar（128）~~ |  | ~~自定义跳转的URL。~~ |
| ~~显示在入口右侧的提示语。~~ | ~~F\_CustomUrlSubTitle NULL~~  ~~Model名：custom\_url\_sub\_title~~ | ~~varchar（18）~~ |  | ~~显示在入口右侧的提示语。~~ |
| ~~卡券跳转的小程序的user\_name~~ | ~~F\_CustomAppBrandUserName NULL~~  ~~Model名：custom\_app\_brand\_user\_name~~ | ~~varchar（128）~~ |  | ~~卡券跳转的小程序的user\_name，仅可跳转该 公众号绑定的小程序 。~~ |
| ~~卡券跳转的小程序的path~~ | ~~F\_CustomAppBrandPass NULL~~  ~~Model名：custom\_app\_brand\_pass~~ | ~~varchar（128）~~ |  | ~~卡券跳转的小程序的path~~ |
| ~~营销场景的自定义入口名称。~~ | ~~F\_PromotionUrlName NULL~~  ~~Model名：promotion\_url\_name~~ | ~~varchar（15）~~ |  | ~~营销场景的自定义入口名称。~~ |
| ~~入口跳转外链的地址链接~~ | ~~F\_PromotionUrl NULL~~  ~~Model名：promotion\_url~~ | ~~varchar（128）~~ |  | ~~入口跳转外链的地址链接。~~ |
| ~~显示在营销入口右侧的提示语~~ | ~~F\_PromotionUrlSubTitle NULL~~  ~~Model名：promotion\_url\_sub\_title~~ | ~~varchar（18）~~ |  | ~~显示在营销入口右侧的提示语~~ |
| ~~卡券跳转的小程序的user\_name，仅可跳转该 公众号绑定的小程序~~ | ~~F\_PromotionAppBrandUserName NULL~~  ~~Model名：promotion\_app\_brand\_user\_name~~ | ~~varchar（128）~~ |  | ~~卡券跳转的小程序的user\_name，仅可跳转该 公众号绑定的小程序 。~~ |
| ~~卡券跳转的小程序的path~~ | ~~F\_PromotionAppBrandPass NULL~~  ~~Model名：promotion \_app\_brand\_pass~~ | ~~varchar（128）~~ |  | ~~卡券跳转的小程序的path~~ |
| ~~每人可领券的数量限制~~ | ~~F\_GetLimit NULL~~  ~~Model名：get\_limit~~ | ~~int~~ | ~~50~~ | ~~每人可领券的数量限制,不填写默认为50。~~ |
| ~~每人可核销的数量限制,~~ | ~~F\_UseLimit NULL~~  ~~Model名：use\_limit~~ | ~~int~~ | ~~50~~ | ~~每人可核销的数量限制,不填写默认为50~~ |
| ~~卡券领取页面是否可分享。~~ | ~~F\_CanShare NULL~~  ~~Model名：can\_share~~ | ~~int~~ |  | ~~值范围{0,1}卡券领取页面是否可分享。~~ |
| ~~卡券是否可转赠~~ | ~~F\_CanGgiveFriend NULL~~  ~~Model名：can\_give\_friend~~ | ~~int~~ |  | ~~值范围{0,1} 卡券是否可转赠。~~ |
|  | ~~Model名：~~  ~~merchant\_id~~ |  |  | ~~在第三方代制的情况下增加该字段指定哪个子商户创建卡券，非数据库字段；说明：创建卡券时需填入该id号，字段结构如下： sub\_merchant\_info:｛merchant\_id：｝~~ |

##### 微~~信优惠券高级信息表 T\_WxCouponAdvancedInfo~~

~~PS：不支持修改~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~F\_ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某个卡券高级信息的唯一标识。~~ |
| ~~主表ID~~ | ~~F\_CouponDetailID~~ | ~~INT NOT NULL~~ |  | ~~外键。~~  ~~该表的内容需以advanced\_info作为键值的json格式与主表的内容一起发给微信用以创建会员卡。~~ |
| ~~使用门槛（条件）~~ | ~~F\_UseCondition~~  ~~Model名:~~  ~~use\_condition~~ | ~~NULL，JSON~~ |  | ~~使用门槛（条件）字段，若不填写使用条件则在券面拼写：无最低消费限制，全场通用，不限品类；并在使用说明显示：可与其他优惠共享。~~  ~~具体数据见从表。~~ |
| ~~封面摘要结构体~~ | ~~F\_Abstract~~ | ~~NULL，JSON结构~~ |  | ~~封面摘要结构体名称。~~  ~~具体数据见从表。~~ |
| ~~图文列表~~ | ~~F\_Text\_image\_list~~  ~~Model名:~~  ~~text\_image\_list~~ | ~~NULL，JSON结构~~ |  | ~~图文列表，显示在详情内页 ，优惠券券开发者须至少传入 一组图文列表。~~  ~~具体数据见从表。~~ |
| ~~使用时段限制~~ | ~~F\_Time\_limit~~  ~~Model名:~~  ~~time\_limit~~ | ~~NULL，JSON结构~~ |  | ~~使用时段限制。~~  ~~具体数据见从表。~~ |
| ~~商家服务类型~~ | ~~F\_BusinessService~~  ~~Model名:~~  ~~business\_service~~ | ~~NULL，int。商家可提供什么服务，比如0101，代表商家提供1和3服务，即Z\_SERVICE\_WITH\_PET 可带宠物和ERVICE\_DELIVER 外卖服务~~ |  | ~~商家服务类型：~~  ~~1、~~~~ERVICE\_DELIVER 外卖服务；~~  ~~2、Z\_SERVICE\_FREE\_PARK 停车位；~~  ~~3、~~~~Z\_SERVICE\_WITH\_PET 可带宠物；~~  ~~4、Z\_SERVICE\_FREE\_WIFI 免费wifi。多选。~~ |
| ~~满减门槛字段~~ | ~~F\_LeastCost~~  ~~Model名:~~  ~~least\_cost~~ | ~~NULL，int~~ |  | ~~满减门槛字段，可用于兑换券和代金券，填入后将在全面拼写消费满xx元可用。~~ |
| ~~可用类型门槛~~ | ~~F\_ObjectUseFor~~  ~~Model名:~~  ~~object\_use\_for~~ | ~~NULL，varchar（512）~~ |  | ~~购买xx可用类型门槛，仅用于兑换 ，填入后自动拼写购买xxx可用。目前我们不做此功能，所以传递NULL~~ |

##### ~~微信优惠券使用条件表 T\_WxCouponUseCondition~~

~~商家可以在nbr创建代金券和折扣券。这二种券，分别可以针对不同商品类型创建不同的券。这些券要满足不同的条件，才能使用。~~

~~比如只针对水果的代金券，就不能用于蔬菜。只针对蔬菜的代金券就不能用于水果。~~

~~PS：不支持修改~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~F\_ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某个使用条件信息的唯一标识。~~ |
| ~~主表ID~~ | ~~F\_AdvancedInfoID~~ | ~~INT NOT NULL~~ |  | ~~该表的内容需以use\_condition作为键值的json格式。~~ |
| ~~指定可用的商品类目~~ | ~~F\_AcceptCategory~~  ~~Model名:~~  ~~accept\_category~~ | ~~NULL，varchar（512）~~ |  | ~~指定可用的商品类目，仅用于代金券类型 ，填入后将在券面拼写适用于xxx。~~ |
| ~~指定不可用的商品类目~~ | ~~F\_RejectCategory~~  ~~Model名:~~  ~~reject\_category~~ | ~~NULL，varchar（512）~~ |  | ~~指定不可用的商品类目，仅用于代金券类型 ，填入后将在券面拼写不适用于xxxx。~~ |
| ~~是否与其他类型共享门槛~~ | ~~F\_CanUseWithOtherDiscount~~  ~~Model名:~~  ~~can\_use\_with\_other\_discount~~ | ~~NULL，bool~~ |  | ~~不可以与其他类型共享门槛，填写false时系统将在使用须知里拼写“不可与其他优惠共享”，填写true时系统将在使用须知里拼写“可与其他优惠共享”，默认为true~~ |

##### 封~~面摘要结构表 T\_ WxCouponAbstract~~

~~~~

~~PS：不支持修改~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~F\_ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某个封面摘要结构体的唯一标识。~~ |
| ~~主表ID~~ | ~~F\_AdvancedInfoID~~ | ~~INT NOT NULL~~ |  | ~~该表的内容需以abstract作为键值的json格式。~~ |
| ~~封面摘要简介~~ | ~~F\_Abstract~~ | ~~NULL，varchar（24）~~ |  | ~~封面摘要简介。~~ |
| ~~封面图片列表~~ | ~~F\_IconUrlList~~  ~~Model名:~~  ~~icon\_url\_list~~ | ~~NULL，varchar（128）~~ |  | ~~封面图片列表，仅支持填入一个封面图片链接，上传图片接口上传获取图片获得链接，填写非CDN链接会报错，并在此填入。建议图片尺寸像素850\*350。~~ |

##### ~~图文列表 T\_WxCouponTextImage~~

~~~~

~~PS：不支持修改~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~F\_ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某个图文列表的唯一标识。~~ |
| ~~主表ID~~ | ~~F\_AdvancedInfoID~~ | ~~INT NOT NULL~~ |  | ~~该表的内容需以text\_image\_list作为键值的json数组格式。~~ |
| ~~图片链接~~ | ~~F\_ImageUrl~~  ~~Model名:~~  ~~image\_url~~ | ~~NULL，varchar（128）~~ |  | ~~图片链接，必须调用上传图片接口上传图片获得链接，并在此填入， 否则报错。~~ |
| ~~图文描述~~ | ~~F\_Text~~ | ~~NULL，varchar（512）~~ |  | ~~图文描述。~~ |

##### 使用~~时段限制表 T\_WxCouponTimeLimit~~

~~PS：不支持修改~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~F\_ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某个卡券的使用时段限制的唯一标识。~~ |
| ~~主表ID~~ | ~~F\_ AdvancedInfoID~~ | ~~INT NOT NULL~~ |  | ~~该表的内容需以time\_limit作为键值的json数组格式。~~ |
| ~~限制使用日期~~ | ~~F\_type~~ | ~~NULL，varchar（24）~~ |  | ~~限制类型枚举值：支持填入MONDAY周一，TUESDAY周二， WEDNESDAY周三，THURSDAY周四，FRIDAY周五，SATURDAY周六，SUNDAY周日，此处只控制显示，不控制实际使用逻辑，不填默认不显示。~~ |
| ~~起始时间（小时）~~ | ~~F\_Begin\_hour~~  ~~Model名:~~  ~~begin\_hour~~ | ~~NULL，int~~ |  | ~~当前type类型下的起始时间（小时），如当前结构体内填写了MONDAY，此处填写了10，则此处表示周一10:00可用。~~ |
| ~~结束时间（小时）~~ | ~~F\_End\_hour~~  ~~Model名:~~  ~~end\_hour~~ | ~~NULL，int~~ |  | ~~当前type类型下的结束时间（小时），如当前结构体内填写了MONDAY，此处填写了20，则此处表示周一10:00-20:00可用。~~ |
| ~~起始时间（分钟）~~ | ~~F\_Begin\_minute~~  ~~Model名:~~  ~~begin\_minute~~ | ~~NULL，int~~ |  | ~~当前type类型下的起始时间（分钟），如当前结构体内填写了MONDAY，begin\_hour填写10，此处填写了59，则此处表示周一10:59可用。~~ |
| ~~结束时间（分钟）~~ | ~~F\_End\_minute~~  ~~Model名:~~  ~~end\_minute~~ | ~~NULL，int~~ |  | ~~当前type类型下的结束时间（分钟），如当前结构体内填写了MONDAY，begin\_hour填写10，此处填写了59，则此处表示周一10:59-00:59可用。~~ |

#### 卡~~券领取事件推送表T\_WxCardCouponUserCode~~

~~微信在微信用户领取卡券后，向我们nbr服务器推送事件。我们用本表来接收这些事件的数据，并作扣减库存的处理。~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
|  | ~~F\_ID~~ |  |  |  |
| ~~领券者的OpenID~~ | ~~F\_VipOpenID~~ | ~~VARCHAR(32) NOT NULL~~ |  |  |
| ~~卡券ID~~ | ~~F\_CardCouponID~~ | ~~VARCHAR(32) not NULL~~ |  |  |
|  | ~~F\_IsGiveByFriend~~ | ~~INT NULL~~ |  | ~~是否为转赠领取，1代表是，0代表否~~ |
|  | ~~F\_FriendUserName~~ | ~~VARCHAR(32) NULL~~ |  | ~~当IsGiveByFriend为1时填入的字段，表示发起转赠用户的openid~~ |
|  | ~~F\_Code~~ | ~~VARCHAR(20) NOT NULL~~ |  | ~~code序列号~~ |
|  | ~~F\_OldUserCardCode~~ | ~~VARCHAR(128) NULL~~ |  | ~~为保证安全，微信会在转赠发生后变更该卡券的code号，该字段表示转赠前的code~~ |
|  | ~~F\_UnionId~~ | ~~VARCHAR(32) NULL~~ |  | ~~领券用户的UnionId~~ |
|  | ~~F\_Status~~ | ~~INT NOT NULL~~ |  | ~~会员卡状态：0=未激活；1=正常；2=已过期；3=转赠中；4=转赠成功；5=转赠超时；6=已删除；7=已失效~~  ~~优惠券状态：0=正常；1=已核销；2=已过期；3=转赠中；4=转赠超时；5=已删除；6=已失效~~ |
|  | ~~F\_Type~~ | ~~INT NOT NULL~~ |  | ~~卡券类型,0=会员卡,1=优惠券~~ |
|  |  |  |  |  |

#### ~~会员卡和券有效期表 T\_WxDateInfo~~

~~卡和券共用。~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~有效期ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某张会员卡的有效期的唯一标识。~~ |
| ~~使用时间的类型~~ | ~~F\_type~~ | ~~Not NULL，varchar~~ |  | ~~使用时间的类型，支持：~~  ~~1、固定时长有效类型()；~~  ~~2、固定日期有效类型()；~~  ~~3、永久有效类型( DATE\_TYPE\_PERMANENT)。~~  ~~支持调用修改会员卡信息的接口进行修改，仅支持更改type为1的时间戳，不支持填入2，此时数据格式为int。~~ |
| ~~起用时间~~ | ~~F\_BeginTimestamp~~  ~~Model名：begin\_timestamp~~ | ~~NULL，int~~ |  | ~~typ为DATE\_TYPE\_FIX\_TIME\_RANGE时专用，表示起用时间。从1970年1月1日00:00:00至起用时间的秒数 （东八区时间,UTC+8，单位为秒）。~~  ~~支持调用修改会员卡信息的接口进行修改，数据格式为unsigned int。~~ |
| ~~结束时间~~ | ~~F\_EndTimestamp~~  ~~Model名：end\_timestamp~~ | ~~NULL，int~~ |  | ~~type为DATE\_TYPE\_FIX\_TERM\_RANGE时专用，表示结束时间 （东八区时间,UTC+8，单位为秒）。~~  ~~支持调用修改会员卡信息的接口进行修改，数据格式为unsigned int。~~ |
| ~~领取后多少天内有效~~ | ~~F\_FixedTerm~~  ~~Model名：fixed\_term~~ | ~~NULL，int~~ |  | ~~type为DATE\_TYPE\_FIX\_TERM时专用，表示自领取后多少天内有效，领取后当天有效填写0（单位为天）。~~  ~~微信文档虽然没说该字段可以修改，但是以此表作为内容的键值（date\_info）微信是允许修改的。~~ |
| ~~领取后多少天开始生效~~ | ~~F\_FixedBeginTerm~~  ~~Model名：fixed\_begin \_term~~ | ~~NULL，int~~ |  | ~~type为DATE\_TYPE\_FIX\_TERM时专用，表示自领取后多少天开始生效。（单位为天）。~~  ~~微信文档虽然没说该字段可以修改，但是以此表作为内容的键值（date\_info）微信是允许修改的。~~ |

#### ~~商品（会员卡）信息 T\_WxSku~~

~~卡和券共用。~~

~~PS：不支持修改~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~F\_ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某张会员卡商品信息的唯一标识。~~ |
| ~~库存数量~~ | ~~F\_Quantity~~ | ~~Not NULL，int，上限为100000000~~ |  | ~~卡券库存的数量，不支持填写0，上限为100000000。~~ |

#### ~~微信会员表T\_WxVip~~

**~~字段来源参考的是查询会员信息接口：[https://developers.weixin.qq.com/doc/offiaccount/Cards\_and\_Offer/Membership\_Cards/Manage\_Member\_Card.html#1](https://developers.weixin.qq.com/doc/offiaccount/Cards_and_Offer/Membership_Cards/Manage_Member_Card.html" \l "1)~~**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
|  | ~~F\_ID~~ |  |  |  |
|  | ~~F\_VipID~~ |  |  | ~~外键~~ |
| ~~用户在本公众号内唯一识别码~~ | ~~F\_PublicAccountOpenID~~ | ~~VARCHAR(32)~~ |  | ~~用户在本公众号内唯一识别码。~~  ~~PS：该字段在会员与会员卡的关联表里也有记录，可在两个表中取一个表保留该字段，用于判断顾客是否为会员。~~ |
|  | ~~F\_MiniProgramOpenID~~ |  |  |  |
|  | ~~F\_UnionID~~ |  |  |  |
| ~~用户昵称~~ | ~~F\_Nickname~~ | ~~VARCHAR(32)~~ |  | ~~用户昵称。该字段在顾客修改自己信息时会发生改变，所以顾客每次在微信发起请求时，nbr是否都需要查询并更新该字段？还是只在界面提供一个刷新按钮，商家每次点击便批量更新会员表的该字段信息？~~ |
| ~~积分信息~~ | ~~F\_Bonus~~ | ~~int~~ |  | ~~积分信息。顾客每次使用微信支付增加积分或使用积分抵扣现金时该字段会发生改变，其改变有两种方式：~~  ~~1、nbr调用微信修改会员信息接口发起修改会员信息请求后，再调用微信查询会员信息接口进行会员信息更新；~~  ~~2、nbr调用微信修改会员信息接口发起修改会员信息请求时，也顺便自己根据发送内容自行更新会员信息；~~  ~~需giggs决定使用哪种，方式1的话一旦微信会员积分信息更新失败，nbr数据库的会员信息也会更新失败；方式2就可能出现微信和nbr只有一方更新会员信息成功的情况，造成数据不同步。~~ |
| ~~余额信息~~ | ~~F\_Balance~~ | ~~float~~ |  | ~~余额信息。目前会员卡不需要开通储值功能，所以该字段可以考虑暂不创建。~~ |
| ~~用户性别~~ | ~~F\_ Sex~~ | ~~int~~ |  | ~~值范围{0,1} 0代表男，1代表女 用户性别。跟上面用户昵称相同的疑问。该字段是需要的。~~ |
| ~~会员信息~~ | ~~F\_UserInfo~~  ~~Model名：user\_info~~ | ~~VARCHAR(100)~~ |  | ~~会员信息。暂不确定该字段具体会返回什么内容，有可能是一个数组数据，需测试得知，以便分析其可修改性。~~ |
| ~~会员卡会员信息类目~~ | ~~F\_~~~~CustomFieldList~~  ~~Model名：~~~~custom\_field\_list~~ |  |  | ~~开发者设置的会员卡会员信息类目，如等级。~~  ~~PS：这个可能会是一个json数据。~~  ~~该字段的内容在商家修改会员卡的相关内容时，会发生修改。~~ |
| ~~会员信息类目名称~~ | ~~F\_Name~~ | ~~VARCHAR(32)~~ |  | ~~会员信息类目名称。该字段内容可能会是F\_Custom\_field\_list字段里的内容。~~  ~~PS：已有的会员表已有该字段，需要改名。~~  ~~该字段的内容在商家修改会员卡的相关内容时，会发生修改。~~ |
| ~~会员卡信息类目值~~ | ~~F\_Value~~ | ~~VARCHAR(32)~~ |  | ~~会员卡信息类目值，比如等级值等。该字段内容可能会是F\_Custom\_field\_list字段里的内容。~~  ~~该字段的内容在商家修改会员卡的相关内容时，会发生修改。~~ |
| ~~用户会员卡状态~~ | ~~F\_UserCardStatus~~  ~~Model名：user\_card\_status~~ | ~~VARCHAR(16)~~ |  | ~~当前用户会员卡状态：~~  ~~1、NORMAL 正常；~~  ~~2、EXPIRE 已过期；~~  ~~3、GIFTING 转赠中；~~  ~~4、GIFT\_SUCC 转赠成功；~~  ~~5、GIFT\_TIMEOUT 转赠超时；~~  ~~6、DELETE 已删除；~~  ~~7、UNAVAILABLE 已失效。~~  ~~目前计划在收到领卡事件推送时创建会员表，此时会员卡应该是未激活状态，实现需要设置0作为未激活的状态？虽然目前是设置会员卡领取后自动激活，但是也有可能自动激活会员卡失败。~~  ~~用户删除会员卡时会有删除事件推送，可据此修改用户会员卡状态为已删除。~~ |
| ~~上次使用时间~~ | ~~F\_LastUsedDateTime~~ | ~~dateTime~~ |  | ~~上次使用积分时间，根据页面需要补充的字段。记录顾客最新一次使用会员卡的时间，用于商家根据此字段进行会员查询。~~ |
|  | ~~F\_CreateDatetime~~ |  |  | ~~用户在微信领会员卡的时间~~ |

~~再增加 某些字段，指向T\_WxVip和T\_AlipayVip。~~

#### 会~~员单元表T\_VipUnit此表暂时不需要~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~ID~~ | ~~F\_ID~~ |  |  | ~~会员在后台上创建的话，会生成这个ID~~ |
|  | ~~F\_VipID~~ | ~~Int not null~~ |  | ~~主表ID~~ |
| ~~ICID~~ | ~~F\_ICID~~ | ~~NULL, Varchar(18)~~ |  | ~~字段惟一~~ |
| ~~手机~~ | ~~F\_Mobile~~ | ~~NULL, Varchar(11)~~ | ~~NULL~~ | ~~字段惟一~~ |
| ~~微信号~~ | ~~F\_Wechat~~ | ~~NULL, Varchar(20)~~ |  | ~~字段惟一~~ |
| ~~QQ~~ | ~~F\_QQ~~ | ~~NULL, Varchar(12)~~ |  | ~~字段惟一~~ |
| ~~Name~~ | ~~F\_Name~~ | ~~中英文，(0, 12]字符。Varchar(12), NOT NULL~~ |  |  |
| ~~生日~~ | ~~F\_Birthday~~ |  |  |  |
| ~~创建时间~~ | ~~F\_CreateDatetime~~ | ~~DATETIME NOT NULL DEFAULT now()~~ |  |  |
| ~~修改时间~~ | ~~F\_UpdateDatetime~~ | ~~DATETIME NOT NULL DEFAULT now()~~ |  |  |

#### ~~会员类别变更表~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| 会员ID | F\_ID |  |  |  |
| 区域 |  |  |  |  |
| 变更日期时间 |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

#### ~~会员积分历史表T\_VIP\_PointHistoryT\_VipPointHistory~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~ID~~ | ~~F\_ID~~ |  |  |  |
| ~~会员ID~~ | ~~F\_VIP\_ID~~  ~~F\_VipID~~ |  |  |  |
| ~~增减积分值~~ | ~~F\_PointChanged~~ |  |  | ~~3，-3，等等，可正可负~~ |
| ~~增减原因~~ | ~~F\_PointChangeCause~~ |  |  | ~~？？？？？~~ |
| ~~对应的零售单ID~~ | ~~F\_RetailTradeID~~ |  |  | ~~代替增减原因字段~~ |
| ~~日期时间~~ | ~~F\_CreateDatetime~~ |  |  |  |
|  |  |  |  |  |

#### ~~会员消费历史表T\_VIP\_ConsumeHistory~~

不再需要这个表。其消费数据直接存储在T\_RetailTrade中。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~ID~~ | ~~F\_ID~~ |  |  |  |
| ~~会员ID~~ | ~~F\_VIP\_ID~~ |  |  |  |
| ~~交易ID~~ | ~~F\_TradeID~~ |  |  |  |
| ~~日期时间~~ | ~~F\_Datetime~~ |  |  |  |
|  |  |  |  |  |

#### ~~储值卡表~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~卡ID~~ | ~~F\_ID~~ |  |  |  |
| ~~会员ID~~ |  |  |  |  |
| ~~余额~~ |  | ~~实数~~ |  |  |

#### ~~储值卡消费历史表~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~卡ID~~ | ~~F\_ID~~ |  |  |  |
| ~~消费金额~~ |  | ~~实数~~ |  |  |

#### ~~计次卡：~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |

#### ~~Wx会员卡T\_WxCard~~

~~对应JSON中的member\_card节。在设计Model时，如果不采用member\_card这个名称，则需要考虑在toJSON后，如何将类名转换为member\_card。~~

~~PS：目前会员卡只提供创建和修改功能~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~Wx会员卡ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某张会员卡的唯一标识。目前一个公众号只支持创建一种多张会员卡，所以届时该表只会存在一条数据。目前会员卡设计是不支持删除，只支持修改。~~ |
| ~~会员卡类型~~ | ~~F\_CardType~~  ~~对应的Model字段名为：card\_type~~ | ~~Not NULL, varchar(24)~~ | ~~MEMBER\_CARD~~ | ~~目前一个公众号只支持创建一种多张会员卡，所以此表届时只会有一条数据。暂未找到支持修改该字段内容的接口。~~ |
| ~~会员卡结构体~~ | ~~F\_Member\_card~~  ~~Model对应的字段/类是member\_card~~ | ~~不可NULL，JSON接口~~ |  | ~~具体数据见从表T\_WxCardDetail。~~  ~~Model里需要增加此类，但是表不需要这个字段~~  ~~部分内容支持调用修改会员卡信息的接口进行修改。~~ |
| ~~状态~~ | ~~F\_Status~~ | ~~Varchar(30) not null~~ |  | ~~“CARD\_STATUS\_NOT\_VERIFY”, 待审核 ； “CARD\_STATUS\_VERIFY\_FAIL”, 审核失败； “CARD\_STATUS\_VERIFY\_OK”， 通过审核； “CARD\_STATUS\_DELETE”， 卡券被商户删除； “CARD\_STATUS\_DISPATCH”，在公众平台投放过的卡券；~~  ~~创建会员卡时，会提交给微信审核，审核成功会有审核事件推送，后端可据此修改会员卡的状态。~~ |
| ~~WX开卡时间~~ | ~~F\_WxCardCreateDatetime~~ | ~~DATETIME NOT NULL DEFAULT now()~~ |  |  |
| ~~WX开卡方式~~ | ~~F\_WxCardCreateWay~~ | ~~Int not null default 0~~ |  | ~~0=微信方式开通，1=手机方式开通~~ |

#### 微~~信会员卡结构体 T\_WxCardDetail~~

~~本表的某些字段，在下面的URL中有图解：~~

[~~https://developers.weixin.qq.com/doc/offiaccount/Cards\_and\_Offer/Membership\_Cards/Create\_a\_membership\_card.html~~](https://developers.weixin.qq.com/doc/offiaccount/Cards_and_Offer/Membership_Cards/Create_a_membership_card.html)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~会员卡结构ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某张会员卡结构体的唯一标识。创建会员卡时会相应生成数据用以记录会员卡的结构和相关信息，与会员卡是一对一的关系。~~ |
| ~~主表ID~~ | ~~F\_VipCardID~~ | ~~INT NOT NULL~~ |  | ~~记录关联的会员卡ID，该从表的内容需以member\_card作为键值的json格式与主表的内容一起发给微信用以创建会员卡，即Model可能采用字段名：member\_card~~ |
| ~~会员卡背景图~~ | ~~F\_BackgroundPicUrl~~  ~~对应的Model字段名为：background\_pic\_url~~ | ~~NULL，varchar(128)~~ |  | ~~商家自定义会员卡背景图，须先调用上传图片接口将背景图上传至WX服务器，否则报错，卡面设计请遵循微信会员卡自定义背景设计规范，像素大小控制在1000像素\*600像素以下。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~卡券数据~~ | ~~F\_Base\_info~~  ~~对应的Model字段名为：base\_info~~ | ~~不可NULL，JSON~~ |  | ~~基本的卡券数据，所有卡券类型通用。~~  ~~具体数据见从表。~~  ~~部分内容支持调用修改会员卡信息的接口进行修改。~~ |
| ~~卡券高级信息~~ | ~~F\_Advanced\_info~~  ~~对应的Model字段名为：~~  ~~advanced\_info，可以不用传递给WX服务器~~ | ~~可NULL，JSON结构~~ |  | ~~创建优惠券特有的高级字段。所以创建会员卡时该字段是否可以不传？如果可以不传，就能降低创建会员卡的字段复杂性。~~  ~~具体数据见从表。~~  ~~修改会员卡接口不支持修改该部分内容。~~ |
| ~~是否显示积分~~ | ~~F\_SupplyBonus~~  ~~对应的Model字段名为：supply\_bonus~~ | ~~Not NULL，int~~ |  | ~~值范围{0，1}。是否显示积分，填写true或false。~~  ~~如填写true，积分相关字段均为必填，若设置为true则后续不可以设置为false。~~  ~~支持调用修改会员卡信息的接口进行修改，且仅支持从false变为true，默认为false，修改该字段会重新提交审核。~~ |
| ~~是否支持储值~~ | ~~F\_SupplyBalance~~  ~~对应的Model字段名为：~~  ~~supply\_balance~~ | ~~Not NULL，int~~ |  | ~~值范围：{0, 1}。是否支持储值，填写true或false。如填写true，储值相关字段均为必填。若设置为true则后续不可以被关闭。该字段须开通储值功能后方可使用，详情见：获取特殊权限。目前不要求会员卡开通储值功能。~~  ~~支持调用修改会员卡信息的接口进行修改，且仅支持从false变为true，默认为false，修改该字段会重新提交审核。~~ |
| ~~特权说明~~ | ~~F\_Prerogative~~ | ~~Not NULL，varchar(3072)~~ |  | ~~会员卡特权说明,限制1024汉字。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~是否自动激活~~ | ~~F\_AutoActivate~~  ~~Model名：auto\_activate~~ | ~~可NULL，int~~ |  | ~~值范围{0,1}设置为true时用户领取会员卡后系统自动将其激活，无需调用激活接口，详情见自动激活。目前会员卡的激活方式暂定为自动激活。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~自定义会员信息类目1~~ | ~~F\_Custom\_field1~~  ~~对应的Model字段名为：~~  ~~custom\_field1~~  ~~有图解~~ | ~~NULL，JSON结构~~ |  | ~~自定义会员信息类目，会员卡激活后显示,包含name\_type (name) 和url字段。~~  ~~具体数据见从表。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~自定义会员信息类目2~~ | ~~F\_Custom\_field2~~  ~~对应的Model字段名为：~~  ~~custom\_field2~~  ~~有图解~~ | ~~NULL，JSON结构~~ |  | ~~自定义会员信息类目，会员卡激活后显示,包含name\_type (name) 和url字段。~~  ~~具体数据见从表。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~自定义会员信息类目3~~ | ~~F\_Custom\_field3~~  ~~对应的Model字段名为：~~  ~~custom\_field3~~  ~~有图解~~ | ~~NULL，JSON结构~~ |  | ~~自定义会员信息类目，会员卡激活后显示,包含name\_type (name) 和url字段。~~  ~~具体数据见从表。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~激活会员卡的url~~ | ~~F\_ActivateUrl~~  ~~对应的Model字段名为：~~  ~~activate\_url~~ | ~~NULL，varchar（128）~~ |  | ~~激活会员卡的url，是我们的URL。我们采用自动激活的方式，所以此URL永远为null。~~  ~~注意事项：~~  ~~1、填入了自动激活auto\_activate字段，激活链接activate\_url和一键开卡接口设置都会失效；~~  ~~2、若同时传入了activate\_url，则一键开卡接口设置会失效；~~  ~~3、建议开发者activate\_url、auto\_activate和wx\_activate只填写一项。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~跳转型一键激活跳转的地址链接~~ | ~~F\_WxActivateAfterSubmitUrl~~ | ~~varchar(128) null~~ |  | ~~当创建会员卡时设置wx\_activate\_after\_submit(即F\_F\_WxActivateAfterSubmit)设置为true时，需要该字段，代表跳转到哪个地址链接~~ |
| ~~自定义会员信息类目~~ | ~~F\_CustomCell1~~  ~~对应的Model字段/类名为：custom\_cell1~~  ~~有图解~~ | ~~NULL，JSON结构~~ |  | ~~自定义会员信息类目，会员卡激活后显示。~~  ~~具体数据见从表。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~自定义会员信息类目~~ | ~~F\_CustomCell2~~  ~~对应的Model字段/类名为：custom\_cell2~~  ~~有图解~~ | ~~NULL，JSON结构~~ |  | ~~自定义会员信息类目，会员卡激活后显示。~~  ~~具体数据见从表。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~自定义会员信息类目~~ | ~~F\_CustomCell3~~  ~~对应的Model字段/类名为：custom\_cell3~~  ~~有图解~~ | ~~NULL，JSON结构~~ |  | ~~自定义会员信息类目，会员卡激活后显示。~~  ~~具体数据见从表。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~积分规则 。~~ | ~~F\_BonusRule~~  ~~对应的Model字段/类名为：bonus\_rule~~ | ~~NULL，JSON结构~~ |  | ~~积分规则 。~~  ~~具体数据见从表。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~折扣~~ | ~~F\_Discount~~ | ~~NULL，int~~ |  | ~~折扣，该会员卡享受的折扣优惠，填10就是九折，填20就是八折。目前不要求会员卡可享受折扣。~~  ~~享受折扣的同时，也可以享受积分。二者必居其一，不能2个都没有。~~  ~~支持调用修改会员卡信息的接口进行修改，修改该字段会重新提交审核。~~ |
| ~~是否支持跳转型一键激活的字段~~ | ~~F\_WxActivateAfterSubmit~~ | ~~Int not null~~ |  | ~~值范围{0,1}~~  ~~当创建会员卡时设置wx\_activate(即F\_WxActivate)设置为true时，需要该字段，且也是true~~ |
| ~~是否支持一键开卡~~ | ~~F\_WxActivate~~  ~~对应的Model字段名为：~~  ~~wx\_activate~~ | ~~NULL，int~~ |  | ~~值范围{0,1}设置为true时会员卡支持一键开卡（自动激活），不允许同时传入activate\_url字段，否则设置wx\_activate失效。填入该字段后仍需调用接口设置开卡项方可生效，详情见一键开卡。~~  ~~支持调用修改会员卡信息的接口进行修改，该选项与activate\_url互斥。~~ |
| ~~设置跳转外链查看积分详情~~ | ~~F\_BonusUrl~~  ~~Model字段名bonus\_url~~ | ~~NULL，varchar(128)~~ |  | ~~设置跳转外链查看积分详情。仅适用于积分无法通过激活接口同步的情况下使用该字段。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~设置跳转外链查看余额详情~~ | ~~F\_BalanceUrl~~  ~~Model字段名balance\_url~~ | ~~NULL，varchar(128)~~ |  | ~~设置跳转外链查看余额详情。仅适用于余额无法通过激活接口同步的情况下使用该字段。目前会员卡不要求开通充值功能，该字段可以考虑舍去。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~积分清零规则~~ | ~~F\_BonusCleared~~  ~~Model字段名bonus\_cleared~~ | ~~NULL，varchar(128)~~ |  | ~~积分清零规则文字描述。~~  ~~支持调用修改会员卡信息的接口进行修改，且此时的数据格式是varchar（3072）。~~ |
| ~~积分规则~~ | ~~F\_BonusRules~~  ~~Model字段名bonus\_rules~~ | ~~NULL，varchar(128)~~ |  | ~~积分规则文字描述。~~  ~~支持调用修改会员卡信息的接口进行修改，且此时的数据格式是varchar（3072）。~~ |
| ~~储值说明~~ | ~~F\_BalanceRules~~  ~~Model字段名balance\_rules~~ | ~~NULL，varchar(128)~~ |  | ~~储值说明。目前会员卡不要求开通充值功能，该字段可以考虑舍去。~~  ~~支持调用修改会员卡信息的接口进行修改，且此时的数据格式是varchar（3072）~~ |
| ~~激活会员卡url对应的小程序user\_name~~ | ~~F\_ActivateAppBrandUserName~~  ~~Model字段名activate\_app\_brand\_user\_name~~ | ~~NULL，varchar(128)~~ |  | ~~激活会原卡url对应的小程序user\_name，仅可跳转该公众号绑定的小程序。根据目前的流程来看，暂未涉及小程序。~~  ~~不支持修改。~~ |
| ~~激活会原卡url对应的小程序path~~ | ~~F\_ActivateAppBrandPass~~  ~~Model字段名activate\_app\_brand\_pass~~ | ~~NULL，varchar(128)~~ |  | ~~激活会原卡url对应的小程序path。根据目前的流程来看，暂未涉及小程序。~~  ~~不支持修改。~~ |

#### ~~微信会员卡数据 T\_WxVipCardBaseInfo~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~卡数据ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某个基本卡券数据的唯一标识。属于会员卡结构体的一部分。~~ |
| ~~主表ID~~ | ~~F\_MemberCardID~~  ~~F\_WxCardDetailID~~ | ~~INT NOT NULL~~ |  | ~~外键，指向T\_WxCardDetail。~~  ~~该表的内容需以base\_info作为键值的json格式与主表的内容一起发给微信用以创建会员卡。~~ |
|  | ~~F\_CustomAppBrandUserName，Model名：center\_app\_brand\_user\_name~~ |  |  | ~~卡券跳转的小程序的user\_name~~ |
|  | ~~F\_CustomAppBrandPass~~  ~~Model名：custom\_app\_brand\_pass~~ |  |  | ~~卡券跳转的小程序的path~~ |
| ~~商户logo~~ | ~~F\_Logo\_url~~  ~~在Model中的命名为logo\_url~~ | ~~Not NULL，varchar(128)~~ |  | ~~公司表已经有这个值。~~  ~~卡券的商户logo，建议像素为300\*300。~~  ~~支持调用修改会员卡信息的接口进行修改。~~  ~~在Model中需要有本字段，对应T\_Company的F\_LogoWXURL~~ |
| ~~商户名字~~ | ~~F\_Brand\_name~~  ~~在Model中的命名为brand\_name~~ | ~~Not NULL，varchar，字数上限为12个汉字~~ |  | ~~商户名字，字数上限为12个汉字。~~  ~~不支持修改。~~  ~~在Model中需要有本字段，对应T\_Company的F\_MchName~~ |
| ~~Code展示类型~~ | ~~F\_CodeType~~  ~~Model名：code\_type~~ | ~~Not NULL，varchar(30)~~ |  | ~~Code展示类型，~~  ~~"CODE\_TYPE\_TEXT"：文本；~~  ~~"CODE\_TYPE\_BARCODE"：一维码；~~  ~~"CODE\_TYPE\_QRCODE"：二维码；~~  ~~"CODE\_TYPE\_ONLY\_QRCODE"：仅显示二维码；~~  ~~"CODE\_TYPE\_ONLY\_BARCODE"：仅显示一维码；~~  ~~"CODE\_TYPE\_NONE"：不显示任何码型。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~卡名~~ | ~~F\_Title~~ | ~~Not NULL，varchar，字数上限为9个汉字~~ |  | ~~卡名，字数上限为9个汉字(建议涵盖卡券属性、服务及金额，比如微信支付自动积分)。~~  ~~支持调用修改会员卡信息的接口进行修改，格式为varchar（27），修改该字段会重新提交审核。~~ |
| ~~卡颜色~~ | ~~F\_Color~~ | ~~Not NULL，varchar(16)~~ |  | ~~按色彩规范标注填写Color010-Color100。~~  ~~支持调用修改会员卡信息的接口进行修改，格式为varchar（3072）。~~ |
| ~~卡使用提醒~~ | ~~F\_Notice~~ | ~~Not NULL，varchar，字数上限为16个汉字~~ |  | ~~卡使用提醒，字数上限为16个汉字。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~客服电话~~ | ~~F\_ServicePhone~~  ~~Model名service\_phone~~ | ~~NULL，varchar（24）~~ |  | |  | | --- | | ~~客服电话。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ | |
| ~~卡使用说明~~ | ~~F\_Description~~ | ~~Not NULL，varchar，字数上限为1024个汉字~~ |  | ~~卡券使用说明，字数上限为1024个汉字。~~  ~~支持调用修改会员卡信息的接口进行修改，格式为varchar（3072），修改该字段会重新提交审核。~~ |
| ~~使用日期与有效期~~ | ~~F\_Date\_info~~  ~~Model名date\_info~~ | ~~Not NULL，JSON~~ |  | ~~使用日期，有效期的信息。~~  ~~具体数据见从表。~~  ~~支持调用修改会员卡信息的接口进行修改，有效期时间修改仅支持有效区间的扩大。~~ |
|  | ~~F\_DateInfoID~~ |  |  | ~~外键。指向T\_WxDateInfo的一行~~ |
| ~~商品信息~~ | ~~F\_Sku~~  ~~Model名sku~~ | ~~Not NULL，JSON~~ |  | ~~商品信息。~~  ~~具体数据见从表。~~  ~~不支持修改。~~ |
|  | ~~F\_SkuID~~ |  |  | ~~外键。指向T\_WxSku的一行~~ |
| ~~每人可领数量~~ | ~~F\_GetLimit~~  ~~Model名get\_limit~~ | ~~NULL，int， default 1~~ |  | ~~每人可领券的数量限制，建议会员卡每人限领一张。~~  ~~如果有人领多了，是需要nbr判断出来还是WX自己帮忙判断？WX判断~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~是否自定义Code码~~ | ~~F\_ UseCustomCode~~  ~~Model名：use\_custom\_code~~ | ~~NULL，int~~ | ~~false~~ | ~~值范围{0,1}是否自定义Code码。填写true或false，默认为false 通常自有优惠码系统的开发者选择自定义Code码，详情见 是否自定义code。~~  ~~Code码—标识这种卡的唯一一张，即独一无二的身份。~~  ~~不支持修改。~~ |
| ~~是否可转赠~~ | ~~F\_CanGiveFriend~~  ~~Model名：can\_give\_friend~~ | ~~NULL，int~~ | ~~true~~ | ~~值范围{0,1}卡券是否可转赠,默认为true。会员卡不支持转赠，该字段可以考虑删除。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~门店位置~~ | ~~F\_LocationIdList~~  ~~Model名：location\_id\_list~~ | ~~NULL，array。将来可能会使用这样的格式 ：aaa,bbb,ccc~~ |  | ~~门店位置ID。调用POI门店管理接口获取门店位置ID。~~  ~~跟微信门店相关。BA说后期要自己弄微信门店，目前未弄~~  ~~支持调用修改会员卡信息的接口进行修改，更新适用门店列表。~~ |
| ~~自定义外链入口~~ | ~~F\_CustomUrlName~~  ~~Model名：custom\_url\_name~~ | ~~NULL，varchar（15）~~ |  | ~~自定义跳转外链的入口名字。~~  ~~支持调用修改会员卡信息的接口进行修改，格式为varchar（16）。~~ |
| ~~自定义外链url~~ | ~~F\_CustomUrl~~  ~~Model名: custom\_url~~ | ~~NULL，varchar（128）~~ |  | ~~自定义跳转的URL。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~提示语~~ | ~~F\_CustomUrlSubTitle~~  ~~Model名: custom\_url\_sub\_title~~ | ~~NULL，varchar（18）~~ |  | ~~显示在入口右侧的提示语。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~自定义场景入口~~ | ~~F\_PromotionUrlName~~  ~~Model名:~~ ~~promotion\_url\_name~~ | ~~NULL，varchar（15）~~ |  | ~~营销场景的自定义入口名称。搞促销时要跳到的URL~~  ~~支持调用修改会员卡信息的接口进行修改，格式为varchar（16）。~~ |
| ~~自定义场景url~~ | ~~F\_PromotionUrl~~  ~~Model名: promotion\_url~~ | ~~NULL，varchar（128）~~ |  | ~~入口跳转外链的地址链接。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~营销入口提示语~~ | ~~F\_PromotionUrlSubTitle~~  ~~Model名: promotion\_url\_sub\_title~~ | ~~NULL，varchar（18）~~ |  | ~~显示在营销入口右侧的提示语。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~设置事件推送~~ | ~~F\_NeedPushOnView~~  ~~Model名: need\_push\_on\_view~~ | ~~NULL，int~~ | ~~false~~ | ~~值范围{0,1}填写true为用户点击进入会员卡时推送（到NBR的）事件，默认为false。详情见 进入会员卡事件推送。~~  ~~不支持修改。~~ |
| ~~支付功能结构体~~ | ~~F\_Pay\_info~~  ~~Model名: pay\_info~~ | ~~NULL，JSON~~ |  | ~~支付功能结构体，swipe\_card结构。该字段在参考示例中没有，但是微信文档有提供，需根据需求决定是否使用。若使用，应将下列相关字段建成它的从表。~~  ~~支持调用修改会员卡信息的接口进行修改。~~  ~~非必填字段，目前尚未明确相应字段以及使用用途，保留该字段~~ |
| ~~刷卡功能结构体~~ | ~~F\_Swipe\_card~~  ~~Model名: swipe\_card~~ | ~~NULL，JSON~~ |  | ~~刷卡功能结构体，包含is\_swipe\_card字段。该字段在参考示例中没有，但是微信文档有提供，需根据需求决定是否使用。若使用，应将下列相关字段建成它的从表。~~  ~~支持调用修改会员卡信息的接口进行修改。~~  ~~非必填字段，目前尚未明确相应字段以及使用用途，保留该字段~~ |
| ~~是否支持拉出微信支付刷卡界面~~ | ~~F\_IsSwipeCard~~  ~~Model名: is\_swipe\_card~~ | ~~NULL，int~~ |  | ~~值范围{0,1} 是否设置该会员卡支持拉出微信支付刷卡界面。该字段在参考示例中没有，但是微信文档有提供，需根据需求决定是否使用。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~是否同时支持微信支付刷卡和会员卡二维码~~ | ~~F\_IsPayAndQrcode~~  ~~Model名: is\_pay\_and\_qrcode~~ | ~~NULL，int~~ |  | ~~值范围{0,1} 是否设置该会员卡中部的按钮同时支持微信支付刷卡和会员卡二维码。该字段在参考示例中没有，但是微信文档有提供，需根据需求决定是否使用。~~  ~~提供2种方式，方便顾客给钱。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~是否指定用户领取~~ | ~~F\_BindOpenid~~  ~~Model名:bind\_openid~~ | ~~NULL，int~~ | ~~false~~ | ~~值范围{0,1} 是否指定用户领取，填写true或false。默认为false。用户关注公众号后，才可以指定。~~  ~~不支持修改。~~ |
| ~~是否支持全部门店~~ | ~~F\_UseAllLocations~~  ~~Model名: use\_all\_locations~~ | ~~NULL，int~~ |  | ~~值范围{0,1} 会员卡是否支持全部门店，填写后商户门店更新时会自动同步至卡券~~**~~。~~**~~该字段在参考示例中没有，但是微信文档有提供，需根据需求决定是否使用。~~  ~~对多门店而言。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~卡券中部居中的按钮~~ | ~~F\_CenterTitle~~  ~~Model名:center\_title~~ | ~~NULL，varchar（18）~~ |  | ~~卡券中部居中的按钮，仅在卡券激活后且可用状态时显示。该字段在参考示例中没有，但是微信文档有提供，需根据需求决定是否使用。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~入口下方提示语~~ | ~~F\_CenterSubTitle~~  ~~Model名:center\_sub\_title~~ | ~~NULL，varchar（24）~~ |  | ~~显示在入口下方的提示语 ， 仅在卡券激活后且可用状态时显示。该字段在参考示例中没有，但是微信文档有提供，需根据需求决定是否使用。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~顶部居中的url~~ | ~~F\_CenterUrl~~  ~~Model名:center\_url~~ | ~~NULL，varchar（128）~~ |  | ~~顶部居中的url ，仅在卡券激活后且可用状态时显示。该字段在参考示例中没有，但是微信文档有提供，需根据需求决定是否使用。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~卡券领取页面是否可分享~~ | ~~F\_CanShare~~  ~~Model名:can\_share~~ | ~~NULL，int~~ | ~~true~~ | ~~值范围{0,1}卡券领取页面是否可分享，默认为true。该字段在参考示例中没有，但是微信文档有提供，需根据需求决定是否使用。~~  ~~分享给朋友圈或朋友。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
|  | ~~Model名：~~  ~~merchant\_id~~ |  |  | ~~在第三方代制的情况下增加该字段指定哪个子商户创建卡券，非数据库字段；说明：创建卡券时需填入该id号，字段结构如下： sub\_merchant\_info:｛merchant\_id：｝~~ |

#### ~~自定义会员信息类目表 T\_WxCardCustomField~~

~~~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~F\_ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某个自定义会员信息类目的唯一标识。~~ |
| ~~主表ID~~ | ~~F\_MemberCardID~~ | ~~INT NOT NULL~~ |  | ~~外键，指向微信会员卡结构体T\_WxCardDetail的F\_ID。~~  ~~该表的内容需以custom\_field + num作为键值的json数组格式。~~ |
| ~~会员信息类目半自定义名称~~ | ~~F\_NameType~~  ~~Model名:name\_type~~ | ~~NULL，varchar（24）~~ |  | ~~会员信息类目半自定义名称，当开发者变更这类类目信息的value值时，可以选择触发系统模板消息通知用户。~~  ~~1、FIELD\_NAME\_TYPE\_LEVEL：等级；~~  ~~2、FIELD\_NAME\_TYPE\_COUPON：优惠券；~~  ~~3、FIELD\_NAME\_TYPE\_STAMP：印花；~~  ~~4、FIELD\_NAME\_TYPE\_DISCOUNT：折扣；~~  ~~5、FIELD\_NAME\_TYPE\_ACHIEVEMEN：成就；~~  ~~6、FIELD\_NAME\_TYPE\_MILEAGE：里程；~~  ~~7、FIELD\_NAME\_TYPE\_SET\_POINTS：集点；~~  ~~8、FIELD\_NAME\_TYPE\_TIMS：次数。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~会员信息类目自定义名称~~ | ~~F\_Name~~ | ~~NULL，varchar（24）~~ |  | ~~会员信息类目自定义名称，当开发者变更这类类目信息的value值时 不会触发系统模板消息通知用户。~~  ~~微信文档虽然没说该字段可以修改，但是以此表作为内容的键值（custom\_field1~ custom\_field3）微信是允许修改的，所以该字段应该是可以修改的。~~ |
| ~~点击类目跳转外链url~~ | ~~F\_Url~~ | ~~NULL，varchar（128）~~ |  | ~~点击类目跳转外链url。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |

#### ~~自定义会员信息类目T\_WxCardCustomCell~~

~~~~

~~PS：微信文档虽然没说下列字段可以修改，但是以此表作为内容的键值（custom\_cell1）微信是允许修改的，所以下列字段应该是可以修改的。~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~F\_ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某个自定义会员信息类目的唯一标识。~~ |
| ~~主表ID~~ | ~~F\_MemberCardID~~ | ~~INT NOT NULL~~ |  | ~~外键，指向微信会员卡结构体T\_WxCardDetail的F\_ID。~~  ~~该表的内容需以custom\_cell1作为键值的json数组格式。~~ |
| ~~入口名称~~ | ~~F\_Name~~ | ~~Not NULL，varchar（15）~~ |  | ~~入口名称。~~ |
| ~~入口右侧提示语~~ | ~~F\_Tips~~ | ~~Not NULL，varchar（18），6个汉字内~~ |  | ~~入口右侧提示语，6个汉字内。~~ |
| ~~入口跳转链接~~ | ~~F\_Url~~ | ~~Not NULL，varchar（128）~~ |  | ~~入口跳转链接。~~ |

#### 积~~分规则表 F\_WxBonusRule~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~F\_ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某个积分规则的唯一标识。~~ |
| ~~主表ID~~ | ~~F\_MemberCardID~~ | ~~INT NOT NULL~~ |  | ~~外键，指向T\_WxCardDetail.F\_ID~~  ~~该表的内容需以bonus\_rule作为键值的json数组格式。~~ |
| ~~消费金额~~ | ~~F\_CostMoneyUnit~~  ~~Model名: cost\_money\_unit~~ | ~~NULL，int~~ |  | ~~消费金额。以分为单位。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~增加的积分~~ | ~~F\_IncreaseBonus~~  ~~Model名:increase\_bonus~~ | ~~NULL，int~~ |  | ~~对应增加的积分。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~单次可获取的积分上限~~ | ~~F\_MaxIncreaseBonus~~  ~~Model名: max\_increase\_bonus~~ | ~~NULL，int~~ |  | ~~用户单次可获取的积分上限。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~初始设置积分~~ | ~~F\_InitIncreaseBonus~~  ~~Model名: init\_increase\_bonus~~ | ~~NULL，int~~ |  | ~~初始设置积分。~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~每使用多少积分~~ | ~~F\_CostBonusUnit~~  ~~Model名:cost\_bonus\_unit~~ | ~~NULL，int~~ |  | ~~每使用5积分。~~  ~~Nbr需要判断业务逻辑，WX不管的~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~抵扣xx元~~ | ~~F\_ReduceMoney~~  ~~Model名:reduce\_money~~ | ~~NULL，int~~ |  | ~~抵扣xx元，（这里以分为单位）~~  ~~Nbr需要判断业务逻辑，WX不管的~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~抵扣条件~~ | ~~F\_LeastMoneyToUseBonus~~  ~~Model名: least\_money\_to\_use\_bonus~~ | ~~NULL，int~~ |  | ~~抵扣条件，满xx元（这里以分为单位）可用。~~  ~~Nbr需要判断业务逻辑，WX不管的~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |
| ~~抵扣条件~~ | ~~F\_MaxReduceBonus~~  ~~Model名: max\_reduce\_bonus~~ | ~~NULL，int~~ |  | ~~抵扣条件，单笔最多使用xx积分。~~  ~~Nbr需要判断业务逻辑，WX不管的~~  ~~支持调用修改会员卡信息的接口进行修改。~~ |

#### ~~微信会员与微信会员卡关联表 T\_WxVipCard~~

~~T\_WxVipCard：表明一个会员有哪种卡。~~

~~T\_WxVipCardDetail：表明一个会员，有这种卡的哪几张。~~

~~PS：~~**~~字段来源参考的是卡券领取事件推送~~**

~~问题：该关联表和会员表创建的时机是收到领取事件推送还是收到激活会员卡事件推送？~~

~~答：由于激活会员卡事件推送返回的参数在领取事件推送返回的参数都可以找到，而领取事件推送返回的个别参数在激活会员卡事件推送中是没有的，所以应该是在收到领取事件推送的参数时创建该关联表和会员表。~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~F\_ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某个会员与会员卡关联的唯一标识。~~ |
| ~~会员ID~~ | ~~F\_WxVipID~~ | ~~INT NOT NULL~~ |  | ~~微信会员的ID。收到领卡事件推送时，需要调用查询会员的接口查询会员的相关信息用以创建会员，此时创建的会员所在表的ID会作为这里的会员ID用以与会员卡关联起来。~~ |
| ~~会员卡ID~~ | ~~F\_WxVipCardID~~ | ~~INT NOT NULL~~ |  | ~~微信会员卡的ID，来源于数据库的会员卡表自增ID。它是一个卡的种类，不是特定的一张卡。~~ |
| ~~创建时间~~ | ~~F\_CreateDatetime~~ |  |  | ~~消息创建时间（整型）。该字段用于记录会员的创建时间。激活会员卡事件推送的参数也有一个创建时间，不过我们目前设置的是领取后自动激活会员卡，所以该字段取哪个推送事件的参数作为创建时间应该差异不大，需Giggs确认。另外，会员表也会有一个创建时间字段，那么该关联表的这个字段是否需要保留？~~ |

#### ~~微信会员卡详情表T\_WxVipCardDetail~~

~~T\_WxVipCard：表明一个会员有哪种卡。~~

~~T\_WxVipCardDetail：表明一个会员，有这种卡的哪几张。~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~F\_ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~ |
|  | ~~F\_WxVipCardID~~ |  |  |  |
| ~~卡券Code码~~ | ~~F\_Code~~ | ~~Not NULL，varchar(20)~~ |  | ~~卡券Code码。用于查询该积分使用情况是属于哪个会员的。理论上该值应该是唯一的。~~ |
| ~~卡券ID~~ | ~~F\_CardID~~  ~~对应的model字段为Card\_id~~ | ~~Not NULL，varchar（32），Unique key~~ |  | ~~卡券ID~~ |
| ~~微信会员卡背景~~ | ~~F\_BackgroundPicUrl~~  ~~对应的model字段为background\_pic\_url~~ | ~~NULL，varchar（128）~~ |  | ~~支持商家激活时针对单个会员卡分配自定义的会员卡背景。目前不要求给单个会员卡分配自定义的会员卡背景，该字段可考虑舍去。~~ |
| ~~转赠前的code~~ | ~~F\_ OldUserCardCode~~ | ~~NULL，varchar（128）~~ |  | ~~为保证安全，微信会在转赠发生后变更该卡券的code号，该字段表示转赠前的code。目前会员卡是不允许转赠的，所以该字段应该不需要保留在数据库表中。~~  ~~我们目前不支持转赠。~~ |
| ~~code序列号~~ | ~~F\_UserCardCode~~ |  |  | ~~code序列号。转赠后，值会变化，但是目前的需求是会员卡不提供转赠功能。~~ |
| ~~是否为转赠领取~~ | ~~F\_ IsGiveByFriend~~ | ~~Int NULL~~ |  | ~~是否为转赠领取，1代表是，0代表否。目前会员卡是不允许转赠的，所以该字段应该不需要保留在数据库表中。~~ |
| ~~转赠用户的openid~~ | ~~F\_FriendUserName~~ | ~~NULL，varchar（32）~~ |  | ~~当IsGiveByFriend为1时填入的字段，表示发起转赠用户的openid。目前会员卡是不允许转赠的，所以该字段应该不需要保留在数据库表中。~~  ~~我们目前不支持转赠。~~ |
| ~~领取场景值~~ | ~~F\_OuterStr~~ | ~~NULL，varchar（8）~~ |  | ~~领取场景值，用于领取渠道数据统计。可在生成二维码接口及添加Addcard接口中自定义该字段的字符串值。目前不要求分开统计，只需要统计总数，即会员总数~~ |
| ~~是否删除后重新找回~~ | ~~F\_IsRestoreMemberCard~~ | ~~NULL，int~~ |  | ~~用户删除会员卡后可重新找回，当用户本次操作为找回时，该值为1，否则为0。所以该字段是可能会发生改变的。~~ |
| ~~领券用户的UnionId~~ | ~~F\_ UnionId~~ | ~~NULL，varchar（32）~~ |  | ~~领券用户的UnionId。一个微信开放平台下绑定的公众账号，每个用户在不同公众号下的openId是不同的，但是UnionID是唯一的。~~ |
| ~~领券方帐号~~ | ~~F\_FromUserName~~ |  |  | ~~领券方帐号（一个OpenID）。该字段可以考虑是否需要，届时会员表也会补充一个openID字段，可以在两个表中取一个表保留该字段即可。该字段主要用于判断该用户是否为会员。~~  ~~冗余字段。在表T\_WxVip中已经有。~~ |
| ~~消息类型~~ | ~~F\_MsgType~~ | ~~NULL，varchar（32）~~ |  | ~~微信推送消息类型，event。此字段一般也不用记录在数据库表中。~~ |
| ~~事件类型~~ | ~~F\_Event~~ | ~~NULL，varchar（32）~~ |  | ~~微信推送事件类型，user\_get\_card(用户领取卡券)。后端可用该值进行判断是什么事件推送，进而做相关的处理。此字段一般也不用记录在数据库表中。~~ |
| ~~开发者微信号~~ | ~~F\_ToUserName~~ | ~~NULL，varchar（32）~~ |  | ~~开发者微信号。一般情况下该字段不会变，除非公众号绑定的开发者账号发生修改。此字段一般也不用记录在数据库表中。~~ |

#### ~~微信积分使用明细表 T\_ WxBonusUsedDetails~~

~~PS：~~**~~字段来源参考的是更新会员信息接口，在更新会员信息的同时，会对应创建一条积分使用明细的数据，该表的数据一旦创建，理应不可修改和删除。~~**

**~~到时需要试验一下，会员信息更新或使用积分后，这里会怎样~~**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
| ~~F\_ID~~ | ~~F\_ID~~ | ~~INT NOT NULL~~ |  | ~~PRIMARY KEY，自增1。~~  ~~某个积分使用详情的唯一标识。~~ |
|  | ~~F\_WxVipCardDetailID~~ |  |  |  |
| ~~积分全量值~~ | ~~F\_Bonus~~ | ~~NULL，int~~ |  | ~~需要设置的积分全量值，传入的数值会直接显示。传入的数值应是积分变动后的值。~~ |
| ~~积分变动值~~ | ~~F\_AddBonus~~  ~~Model名:~~  ~~add\_bonus~~ | ~~NULL，int~~ |  | ~~本次积分变动值，传负数代表减少。用于记录每次积分变动的情况。~~  ~~退货时，已使用的积分不退回，但是产生的积分需要扣除~~ |
| ~~自定义积分消耗记录~~ | ~~F\_RecordBonus~~  ~~Model名:~~  ~~record\_bonus~~ | ~~NULL，varchar(42)~~ |  | ~~商家自定义积分消耗记录，不超过14个汉字。对积分变动情况的说明。~~ |
| ~~余额全量值~~ | ~~F\_Balance~~ | ~~NULL，int，default 0~~ |  | ~~需要设置的余额全量值，传入的数值会直接显示在卡面。目前会员卡不考虑储值功能，该字段可以考虑舍去。~~ |
| ~~余额变动值~~ | ~~F\_AddBalance~~  ~~Model名:~~  ~~add\_balance~~ | ~~NULL，int，default 0~~ |  | ~~本次余额变动值，传负数代表减少。目前会员卡不考虑储值功能，该字段可以考虑舍去。~~ |
| ~~自定义金额消耗记录~~ | ~~F\_RecordBalance~~  ~~Model名:record\_balance~~ | ~~NULL，varchar(42)~~ |  | ~~商家自定义金额消耗记录，不超过14个汉字。目前会员卡不考虑储值功能，该字段可以考虑舍去。~~ |
| ~~自定义类目名称1~~ | ~~F\_CustomFieldValue1~~  ~~Model名: custom\_field\_value1~~ | ~~NULL，varchar（12）~~ |  | ~~创建时字段custom\_field1定义类型的最新数值，限制为4个汉字，12字节。该字段每次获取的值可能不一样，取决于商家有无修改会员卡关于该字段的内容。该字段需要保留在积分明细表中。~~ |
| ~~自定义类目名称2~~ | ~~F\_CustomFieldValue2~~  ~~Model名: custom\_field\_value2~~ | ~~NULL，varchar（12）~~ |  | ~~创建时字段custom\_field2定义类型的最新数值，限制为4个汉字，12字节。该字段每次获取的值可能不一样，取决于商家有无修改会员卡关于该字段的内容。该字段需要保留在积分明细表中。~~ |
| ~~自定义类目名称3~~ | ~~F\_CustomFieldValue3~~  ~~Model名: custom\_field\_value3~~ | ~~NULL，varchar（12）~~ |  | ~~创建时字段custom\_field3定义类型的最新数值，限制为4个汉字，12字节。该字段每次获取的值可能不一样，取决于商家有无修改会员卡关于该字段的内容。该字段需要保留在积分明细表中。~~ |
| ~~原生消息结构体~~ | ~~F\_NotifyOptional~~  ~~Model名:~~  ~~notify\_optional~~ | ~~NULL，JSON~~ |  | ~~控制原生消息结构体，包含各字段的消息控制字段。~~  ~~该字段的内容应是包含下列的三个字段。~~ |
| ~~积分变动时是否触发系统模板消息~~ | ~~F\_IsNotifyBonus~~  ~~Model名:is\_notify\_bonus~~ | ~~NULL，int~~ | ~~true~~ | ~~值范围{0,1}积分变动时是否触发系统模板消息，默认为true。~~  ~~该字段需要记录在积分使用明细表中。~~ |
| ~~余额变动时是否触发系统模板消息~~ | ~~F\_IsNotifyBalance~~  ~~Model名: is\_notify\_balance~~ | ~~NULL，int~~ | ~~true~~ | ~~值范围{0,1}余额变动时是否触发系统模板消息，默认为true。~~  ~~该字段需要记录在积分使用明细表中。~~ |
| ~~自定义group1是否触发系统模板消息~~ | ~~F\_IsNotifyCustomField1~~  ~~Model名: is\_notify\_custom\_field1~~ | ~~NULL，int~~ | ~~false~~ | ~~值范围{0,1} 自定义group1变动时是否触发系统模板~~~~(~~~~有公众 号发公众号，没有发WX~~~~服务通知)消息，默认为false。（2、3同理）。~~  ~~前提是需要~~  ~~该字段需要记录在积分使用明细表中。~~ |
| ~~自定义group2是否触发系统模板消息~~ | ~~F\_IsNotifyCustomField2~~  ~~Model名: is\_notify\_custom\_field2~~ | ~~NULL，int~~ | ~~false~~ | ~~值范围{0,1} 自定义group2变动时是否触发系统模板(有公众 号发公众号，没有发WX服务通知)消息，默认为false。（2、3同理）。~~  ~~该字段需要记录在积分使用明细表中。~~ |
| ~~自定义group3是否触发系统模板消息~~ | ~~F\_IsNotifyCustomField3~~  ~~Model名: is\_notify\_custom\_field3~~ | ~~NULL，int~~ | ~~false~~ | ~~值范围{0,1} 自定义group3变动时是否触发系统模板(有公众 号发公众号，没有发WX服务通知)消息，默认为false。（2、3同理）。~~  ~~该字段需要记录在积分使用明细表中。~~ |

### 供应商

#### 供应商表T\_Provider

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 名称 | F\_Name | NOT NULL，中英文和数字，不能为空，不能有空格，长度(0, 32] |  |  |
| 区域ID | F\_DistrictID | 正整数 |  | 外键 |
| 联系地址 | F\_Address | NULL，长度[0, 50] ，中间可以有空格，但首尾不能有空格 |  |  |
| 联系人姓名 | F\_ContactName | NULL，长度[0, 20]，必须是中英文或数字，中间可以有空格，但首尾不能有空格 |  |  |
| 联系人电话 | F\_Mobile | NULL，Varchar(24)，[7,24]个字符，如020-38880846-118，020-38880846转118，020-38880846-118，020-38880846-18138 |  |  |

#### 供应商区域表T\_ProviderDistrict

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 区域名字 | F\_Name | NOT NULL，长度[0, 20] |  |  |

### 仓库

#### 仓库表T\_Warehouse

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 博昕客户ID | F\_BX\_CustomerID |  |  |  |
| 名称 | F\_Name | 中文、数字、英文长度为32位 |  |  |
| 地址 | F\_Address | 可选项，中文、数字、英文长度为32位 |  |  |
| 状态 | F\_Status | 0=正常使用，1=已删除不用 |  |  |
| 联系人ID | F\_StaffID |  |  | 外键 |
| 联系人电话 | F\_Phone |  |  |  |
| ~~备注~~ | ~~F\_Remark~~ | ~~Varchar(50) null~~ |  |  |

### 询价单

（瘦系统不做）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

### 促销

#### 商品促销类型表T\_PromotionType

属于~~配置性表，不能修改！~~

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ~~数据项~~ | ~~SQL field~~ | ~~Type~~ | ~~默认值~~ | ~~说明/限制/规则~~ |
|  | ~~F\_ID~~ |  |  | ~~主键~~ |
| ~~促销类型~~ | ~~F\_Type~~ | ~~Varchar(2)~~ |  | ~~特价、折扣、减现、赠送~~ |
| ~~促销规则~~ | ~~F\_Rule~~ | ~~Varchar(8)~~ |  | ~~组合特价、单笔满金额减现等等~~ |
| ~~促销规则描述~~ | ~~F\_RuleRemark~~ | ~~Varchar(100)~~ |  | ~~如：在某一特定时期，指定的单品做特价。如：XXX商品原价10元，1月1日到1月3日特价8元销售~~ |

~~其数据配置为：~~

|  |  |  |  |
| --- | --- | --- | --- |
| ~~ID~~ | ~~促销类型~~ | ~~促销规则~~ | ~~促销规则描述~~ |
| ~~1~~ | ~~特价~~ | ~~指定商品特价~~ | ~~在某一特定时期，指定的单品做特价。如：XXX商品原价10元，1月1日到1月3日特价8元销售~~ |
| ~~2~~ | ~~特价~~ | ~~满一定件数后特价~~ | ~~在某一特定时期，指定的单品购买一定数量时按特价销售。如：XXX商品原价10元，1月1日到1月3日购买3件及以上数量时按7元特价销售~~ |
| ~~3~~ | ~~特价~~ | ~~组合特价~~ | ~~在某一特定时期，同时购买两个以上的指定商品时，可以享受特价。如：A商品原价10元，B商品原价12元，如果同一单里购买了这两个商品，那么A商品将以8元，B商品将以9元的特价销售~~ |
| ~~4~~ | ~~特价~~ | ~~逢倍数特价~~ | ~~在某一特定时期，顾客购买指定商品达到指定倍数的那个商品可以享受特价。如：A商品原价10元，设置倍数为2倍，如果同一单里购买了两个A商品，那么第二个A商品将以8元的特价销售~~ |
| ~~５~~ | ~~折扣~~ | ~~指定商品折扣~~ | ~~在某一特定时期，指定的单品打折。如：1月1日到1月3日 XXX商品打7折销售~~ |
| ~~６~~ | ~~折扣~~ | ~~按品牌折扣~~ | ~~在某一特定时期，指定品牌下的所有商品打折。如：1月1日到1月3日XXX品牌下的所有商品打7折销售~~ |
| ~~７~~ | ~~折扣~~ | ~~按类别折扣~~ | ~~在某一特定时期，指定类别下的所有商品打折。如：1月1日到1月3日XXX类别下的所有商品打7折销售~~ |
| ~~８~~ | ~~折扣~~ | ~~全场折扣~~ | ~~在某一特定时期，全场打折。如：1月1日到1月3日 所有商品打7折销售~~ |
| ~~９~~ | ~~折扣~~ | ~~满一定件数后折扣~~ | ~~在某一特定时期，买指定的单品达到一定数量时打折。如：1月1日到1月3日XXX商品购买3件及以上打6折销售~~ |
| ~~１０~~ | ~~折扣~~ | ~~单笔满金额打折~~ | ~~在特定时期，消费满足了一定金额时，给本笔销售打折。如：1月1日到1月3日消费满500元及以上时，本单打8折销售~~ |
| ~~１１~~ | ~~减现~~ | ~~满一定件数后减现~~ | ~~在某一特定时期，单笔消费中，满足一定数量时，可以减现。如：消费满5件，减50元，付款时自动减现~~ |
| ~~１２~~ | ~~减现~~ | ~~单笔满金额减现~~ | ~~在某一特定时期，单笔消费中，满足一定金额时，可以减现。如：消费满500元，减50元，付款时自动减现~~ |
| ~~１３~~ | ~~赠送~~ | ~~满一定件数后赠送~~ | ~~在特定时期，买指定商品满足一定数量时，赠送礼品。如：1月1日至3日中XXX商品购买2个送a礼品一个或多个~~ |
| ~~１４~~ | ~~赠送~~ | ~~单笔满金额赠送~~ | ~~在某一特定时期，指定商品满足了一定金额时，赠送礼品。如：单笔消费满500元送a、b礼品一个或多个~~ |

#### 促销表T\_Promotion

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 编号 | F\_SN | VARCHAR(20) NOT NULL |  | CX+XXXXXXXX(年月日)+0000(自增长的流水号) 例：CX201906100001 |
| 活动名称 | F\_Name | Varchar(32) NOT NULL，不能为空串 |  |  |
| 状态 | F\_Status | Int |  | 0=有效，1=已经删除（无效）  已删除是老板在web界面上终止了促销导致的。这时，即使未过有效期，该促销仍然无效 |
| 活动类型 | F\_Type | int |  | 0=满减。1=满折 |
| 开始日期 | F\_DatetimeStart | Datetime not null |  | 必须至少提前一天 |
| 结束日期 | F\_DatetimeEnd | Datetime not null |  |  |
| 满减阀值 | F\_ExcecutionThreshold | Decimal(20, 6) not null |  | 最大阀值=10000，前端要限制输入最多2位小数，收到后端的数据时前端格式化为2位小数显示给用户 |
| 满减金额 | F\_ExcecutionAmount | Decimal(20, 6) null |  | 最大阀值=10000，前端要限制输入最多2位小数，收到后端的数据时前端格式化为2位小数显示给用户  F\_Type为0时有效 |
| 满减折扣 | F\_ExcecutionDiscount | Decimal(20, 6) null  范围(0.00, 1.00] |  | 0.88代表88折，0.80代表8折。  前端要限制输入最多2位小数，收到后端的数据时前端格式化为2位小数显示给用户。  F\_Type为1时有效 |
| 参与的商品的范围 | F\_Scope | Int not null | 0 | 0=全部商品，1=部分商品（这些商品记录在T\_PromotionScope中） |
| 创建者 | F\_Staff | Int not null |  | 外键 |
| 创建时间 | F\_CreateDatetime | datetime |  |  |
| 修改时间 | F\_UpdateDatetime | datetime |  |  |

#### 促销范围表T\_PromotionScope

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID |  |  |  |
| 促销ID | F\_PromotionID |  |  |  |
| 商品ID | F\_CommodityID |  | NOT NULL | 限定为单品 |
| 商品名称 | F\_CommodityName | Varchar(32) not null |  | 创建指定范围的促销时，将此时的商品名称插入到本字段中保留。此后不管T\_Commodity中的名称如何变化，此处不变，即“保持历史原貌” |

零售单的促销的计算过程参考[零售单促销表T\_RetailTradePromoting](#_零售单促销表T_RetailTradePromoting)。

### 销售目标

#### 年销售目标

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 机构编码 |  |  |  |  |
| 机构名称 |  |  |  |  |
| 年份 |  |  |  |  |
| 销售目标金额 |  |  |  |  |

#### 月销售目标

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 机构编码 |  |  |  |  |
| 机构名称 |  |  |  |  |
| 年月 |  |  |  |  |
| 销售目标金额 |  |  |  |  |

#### 业务员销售目标

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 业务员编码 |  |  |  |  |
| 业务员名称 |  |  |  |  |
| 机构编码 |  |  |  |  |
| 机构名称 |  |  |  |  |
| 年月 |  |  |  |  |
| 销售目标金额 |  |  |  |  |

### 全局配置表（配置性数据）

#### 普通配置表T\_ConfigGeneral

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 名称 | F\_Name | Varchar(60) not null |  | Unique值 |
| 值 | F\_Value | Varchar(128) not null |  |  |

其数据如下：

|  |  |  |  |
| --- | --- | --- | --- |
| F\_ID | F\_Name | F\_Value | 描述 |
|  | PageSize | 10 | 用来控制每个页面显示的记录条数。 |
|  |  |  |  |
|  |  |  |  |

#### 缓存对象个数配置表T\_ConfigCacheSize

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 名称 | F\_Name |  |  | Unique值 |
| 值 | F\_Value |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| F\_ID | F\_Name | F\_Value | 描述 |
|  | BrandCacheSize | 100000000 | 品牌缓存的对象个数 |
|  | CategoryCacheSize | 100000000 | 类别缓存的对象个数 |
|  | CommodityCacheSize | 100 | 商品缓存的对象个数。开发时使用10个来看看有无问题 |
|  |  |  | (其它的再加……) |

### 同步缓存表

#### 会员同步缓存表T\_VipSyncCache

存储要同步到各个POS的数据块：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 数据的ID | F\_SyncData\_ID |  |  | 对应T\_VIP的F\_ID。 |
| 数据类型 | F\_SyncType | Char(1) |  | C或U型或D型数据 |
| 同步次序 | F\_SyncSequence | int |  | 用于客户端收到N条同步块后，知道按照什么顺序去处理这些同步块。比如对一个对象A进行多次修改，生成了U1、U2、U3三个同步块，POS同步时，必须按顺序执行，否则会出现奇怪的值，比如拿到的是U1的值。 |
| 创建时间 | F\_CreateDatetime | Datetime, NOT NULL |  | 便于追踪 |

#### 会员同步缓存调度表T\_VipSyncCacheDispatcher

存储哪些POS已经同步了哪个数据块：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| ID | F\_ID |  |  |  |
| 同步缓存ID | F\_SyncCacheID |  |  |  |
|  | F\_POS\_ID |  |  | 同步了缓存F\_SyncCacheID的POS机的ID |
| 创建时间 | F\_CreateDatetime | Datetime, NOT NULL |  | 便于追踪 |

#### 其它缓存表

门店人员同步缓存表T\_StaffSyncCache：和会员一样。

门店人员同步缓存调度表T\_StaffSyncCacheDispatcher：和会员一样。

小票同步缓存表T\_SmallSheetSyncCache：和会员一样。

小票同步缓存调度表T\_SmallSheetSyncCacheDispatcher：和会员一样。

### Nbr\_bx全局配置表

T\_BXConfigGeneral：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
|  | F\_ID | NOT NULL AUTO\_INCREMENT |  |  |
|  | F\_Name | Varchar(60) NOT NULL |  |  |
|  | F\_Value | Varchar(128) NOT NULL |  |  |
|  | F\_CreateDatetime | DATETIME NOT NULL | now() |  |
|  | F\_UpdateDatetime | DATETIME NOT NULL | now() |  |

示例（其中第1、2列为固定值）：

|  |  |  |
| --- | --- | --- |
| F\_ID | F\_Name | F\_Value |
| 1 | CompanyBusinessLicensePictureDir | D:/nbr/pic/common\_db/license/ |
| 2 | CompanyBusinessLicensePictureVolumeMax | 102400 |
| 3 | CommodityNOStart | -1 |
| 4 | CommodityPurchasingPriceStart | -1 |
| 5 | CommodityLogoDir | D:/nbr/pic/private\_db/ |
| 6 | CompanyAPICertDir | D:/nbr/apicert/ |
| 7 | CompanyAPICertVolumeMax | 10240 |

### Nbr\_bx.T\_CurrentRelease

记录当前Release的一些信息，比如版本号。帮助Release时验证Release是否成功，帮助拿到正确的版本复现问题。

每次WAR Release都会更新本表。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | SQL field | Type | 默认值 | 说明/限制/规则 |
| F\_ID | F\_ID |  |  |  |
| 对应的WAR版本 | F\_WARReleaseNO | Varchar(20) |  | 当前DB对应的WAR版本 |
| 当前DB版本 | F\_DBReleaseNO | Varchar(20) |  | 当前DB的版本号 |
| 对应的APK版本 | F\_ApkReleaseNO | Varchar(20) |  | 当前DB对应的Apk版本 |
| 解决的JIRA | F\_Jira | Varchar(128) |  | 表明这次Release要解决的是哪些JIRA  示例值：PET-238,PET-92,PEF-77  如果JIRA十分多，不填写也可以 |
| 备注 | F\_Remark | Varchar(128) |  |  |
|  | F\_CreateDatetime | Datetime, not NULL |  |  |
|  | F\_UpdateDatetime | Datetime, not NULL |  |  |

## 物理结构设计要点

用户看到的在界面上显示的表数据

传输数据格式、协议

单据格式

## 数据结构与程序的关系

SQL语言创建数据表

# 系统出错处理设计

## 出错信息

页面不存在

黑客操作

## 补救措施

数据库备份与恢复

## 系统维护设计

日志

数据库备份

DB相关的更新对数据同步的影响及其预案。