|  |
| --- |
| 江苏移动终端销售运营支撑系统 |
| 设计规格说明书 |
| (V1.0) |
| **欣网视讯** |
| **2013/7/1** |

|  |
| --- |
| 内部资料，注意保密，严禁外传 |

**修改记录**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **日期** | **版次** | **描述** | **作者** | **审核** | **批准** |
| 2013.7.23 | V1.0 |  | 孙含新、方勇 |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

目录

[1. 引言 3](#_Toc363559075)

[1.1. 编写目的 3](#_Toc363559076)

[1.2. 读者对象 3](#_Toc363559077)

[1.3. 项目背景 3](#_Toc363559078)

[1.4. 定义 4](#_Toc363559079)

[1.4.1. 专门术语 4](#_Toc363559080)

[1.4.2. 缩写 4](#_Toc363559081)

[1.5. 参考资料 5](#_Toc363559082)

[2. 软件设计约束 5](#_Toc363559083)

[2.1. 设计目标和原则 5](#_Toc363559084)

[2.2. 设计约束 5](#_Toc363559085)

[3. 总体架构设计 5](#_Toc363559086)

[4. 软件架构 7](#_Toc363559087)

[5. 数据设计 8](#_Toc363559088)

[6. 详细设计 8](#_Toc363559089)

[6.1. 系统认证中心 8](#_Toc363559090)

[6.2. 运营管理中心 8](#_Toc363559091)

[6.2.1. 基础信息 8](#_Toc363559092)

[6.2.2. 合作伙伴管理 11](#_Toc363559093)

[6.2.3. 商品管理 11](#_Toc363559094)

[6.2.4. 合同管理 12](#_Toc363559095)

[7. 逻辑组件设计 12](#_Toc363559096)

[8. 接口设计 12](#_Toc363559097)

# 引言

## 编写目的

在概要设计的基础上,结合需求说明,对系统实现进行设计。

## 读者对象

系统开发人员、测试人员、设计人员

## 项目背景

在江苏移动终端统一运营的大背景下，对终端的售前、售中、售后进行全面管理，支撑江苏移动终端统一运营的业务需求。售前主要包括对终端选型，入围的过程进行管理，并提供有效数据支撑辅助运营人员对终端入围的决策。售中主要是对终端商品的生命周期，商品价格、促销，终端的供应商链、销售、库存、财务结算进行有效管理。售后主要是对终端商品的维修、退/换货、配件管理的整个管理过程进行支撑。

## 定义

### 专门术语

* 营销案：
* 裸机：
* 进销存：
* 库存周转率：
* 库存周转天数：

### 缩写

* **TSMS：Terminal Sell Manage System(终端销售运营支撑系统)**
* OMC：Order Manage Center(订单管理中心)
* IMS：Inventory Manage System(库存管理系统)
* FSC：Finance Settlement Center(财务结算中心)
* TSS：Terminal Selecting System（终端选型系统）
* BMS: BusinessManage Center(运营管理中心)
* DTC：Data Transform Center(数据交换中心)
* SAC: System[Authentication](http://www.iciba.com/authentication)Center(系统认证中心)
* OIP：Out Interface Platform（对外接口平台）
* CSS：Channel Sell System(渠道零售系统)
* AMS：After-Sale Manage System(售后管理系统)

## 参考资料

# 软件设计约束

## 设计目标和原则

## 设计约束

# 总体架构设计



子系统和对应数据库说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 系统 | 缩写 | 数据库 |
| 1 | 系统认证中心 | SAC | 接口管理库 |
| 2 | 对外接口平台 | OIP | 接口管理库 |
| 3 | 数据交换平台 | DTC | 接口管理库 |
| 4 | 终端选型系统 | TSS | 运营中心库 |
| 5 | 财务结算中心 | FSC | 运营中心库 |
| 6 | 运营管理中心 | BMC | 运营中心库 |
| 7 | 订单管理中心 | OMC | 订单库存库 |
| 8 | 库存管理系统 | IMS | 订单库存库 |
| 9 | 渠道零售系统 | CSS | 渠道销售库 |
| 10 | 售后管理系统 | AMS | 渠道销售库 |

* 系统认证中心(SAC)

各个子系统集成框架，负责子系统之间的统一认证，权限控制。主要模块保存整个系统层面的信息管理，大概包括用户角色、权限控制、统一登录、子系统维护等模块。

* 对外接口平台(OIP)

负责对外接口管理，包括调用第三方系统接口和提供第三方调用的接口，对外接口主要功能包括接口认证、交互协议管理、报文解析封装等。

* 数据交换平台(DTC)

负责各个子系统之间的数据交互，业务逻辑分享，统一运营指令的下发。主要功能包括，交互业务逻辑管理，交互方式控制，交互方管理，交互日志等。

* 终端选型系统(TSS)

负责售前终端入围决策支持和管理，提供入围模型和终端参数管理。主要包括终端参数管理、终端入围模型设计、终端选型入围支持。

* 财务结算中心(FSC)

负责整个系统对外部合作单位的结算流程管理，主要结算流程包括与供应商货款结算、物流商物流费用结算、分销商的合约补贴结算等。

* 运营管理中心(BMC)

主要包括基础档案的管理，商品管理，商品控制，价格管理，串号管理等运营控制模块。

* 订单管理中心(OMC)

负责整个终端订单流转管理，提供有效的供应链支持。订单管理中心主要包括销售订单、自销订单、分销订单、采购订单、要货计划、集采计划等流程管控。

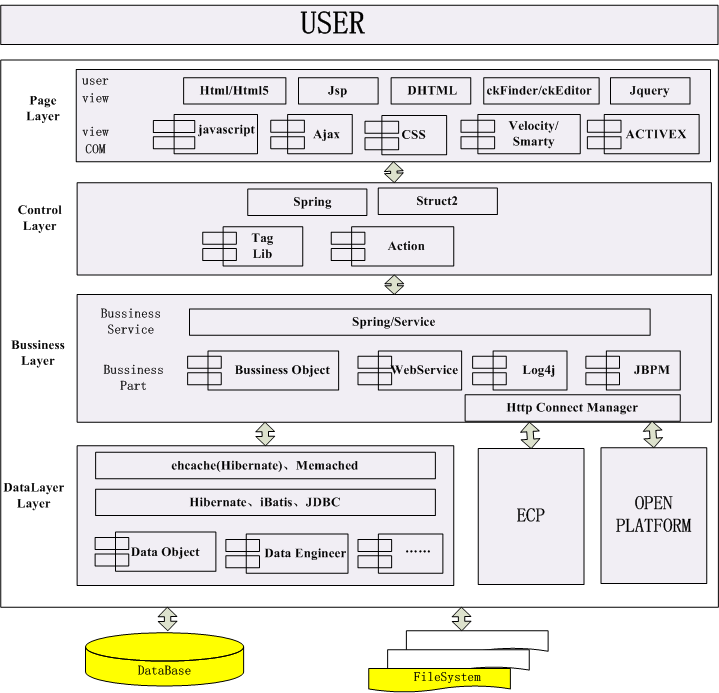
* 库存管理系统(IMS)

主要负责物流仓储的库存，通过入库、出库、库存、调拨、盘点等维度对物流仓储的实行管控，以及对订单的物流配送流程进行跟踪。通过实际数据支撑对物流商进行考核。

* 渠道零售系统(CSS)
* 售后管理系统(AMS)

# 软件架构





# 数据设计

见数据模型设计文件。

# 详细设计

## 库存管理中心

### 仓储信息同步

### 仓储送货单管理

#### 送货单手工发送

* 处理流程

无

* 功能简述

1. 点击功能菜单后，打开【送货单手工发送页面】（6.1.2.1 -1），用户点击【查询】按钮，根据查询条件将符合条件的单据信息查询出来并在列表中展现。

IDelivstockService.queryDelivNoSend

1. 查询条件说明：

* 送货单号、订单号后台模糊查询
* 供货商：（见订单管理组件IOrder.getUnityDistSupplier）
* 仓库：(见订单管理组件IOrder.getAllWarehouse)
* 开始时间、结束时间，提供日期选择控件。

开始时间默认：当前年月第一天+“00：00：00”；

结束时间默认当前时间。

* 需增加默认查询条件，【物流对接标志】(F\_IS\_TOWARE)：0 并且送货单状态是DRKJY(待入库校验)

1. 用户可多选，选择记录后，点击【手工同步】按钮，提示同步结果，并刷新列表。
2. 用户单击某一行，可选中该条数据。
3. 可多选。
4. 调用同步接口，同步数据。（暂时没有该接口，代码预留）
5. 每次调用接口前更新【物流对接次数】（F\_TOWARE\_COUNT），【物流最后对接时间】（F\_LAST\_TOWARE\_TIME），若调用接口成功曾更新【物流对接成功】（F\_IS\_TOWARE）标记为：1。
6. 用户选中某条记录，点击【查看详情】按钮，跳转到【送货单手工发送详情页面】（6.1.2.1 -2）。
7. 只能查看详细信息，不能修改。
8. 选中多条记录，默认展现选中记录的第一条。
9. 【送货单手工发送详情页面】（6.1.2.1 -2），上半步展现送货单基本信息，下半部展现商品明细及送货单串号的列表信息（两种信息tab页面切换）。

IDelivstockService.getOneDelivStockMain;

IDelivstockService.queryDelivstockSub;

IDelivstockService.queryDelivstockSubDetail;

* 数据库表结构

送货单主表（TO\_BILL\_DELIVSTOCK\_MAIN）

送货单子表（TO\_BILL\_DELIVSTOCK\_SUB）

送货单子表明细（TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL）

* 详细设计

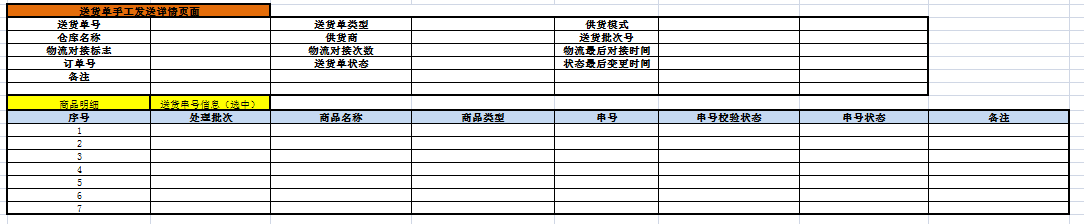
无

* 界面设计



6.1.2.1 -1





6.1.2.1 -2

#### 送货单跟踪

* 处理流程

无

* 功能简述

1. 点击功能菜单后，打开【送货单跟踪页面】（6.1.2.2 -1），用户点击【查询】按钮，根据查询条件将符合条件的单据信息查询出来并在列表中展现。

IDelivstockService.queryDelivstockMain

1. 查询条件说明：

送货单号、订单号后台模糊查询

供货商：（见订单管理组件IOrder.getUnityDistSupplier）

仓库：(见订单管理组件IOrder.getAllWarehouse)

开始时间、结束时间，提供日期选择控件

开始时间默认：当前年月第一天+“00：00：00”；

结束时间默认当前时间。

送货单状态:全部、（见TB\_BILL\_STATUS where F\_BILL\_TYPE\_NUM=’ BILL\_CCSHDGL’）;

1. 用户选择某条记录后，点击【查看详情】按钮，跳转到【送货单跟踪详情页面】（6.1.2.2 -2）。
2. 用户单击某一行，可选中该条数据。
3. 只能单选。
4. 【送货单跟踪详情页面】（6.1.2.2 -2），上半步展现送货单基本信息，下半部展现商品明细、送货串号、入库串号及单据历史信息。（四种信息tab页面切换）。(送货单详情做成公用组件页面)

IDelivstockService. getOneDelivStockMain(送货单主要信息)

IDelivstockService. queryDelivstockSub（送货单商品明细）

IDelivstockService. queryDelivstockSubDetail（送货串号）

IDelivstockService. queryDelivStockLog（操作日志）

IInstockService. queryInstockDetailByDelivNum（入库串号）

* 数据库表结构

送货单主表（TO\_BILL\_DELIVSTOCK\_MAIN）

送货单子表（TO\_BILL\_DELIVSTOCK\_SUB）

送货单子表明细（TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL）

入库单主表（TO\_BILL\_INSTOCK\_MAIN）

入库单子表明细（TO\_BILL\_INSTOCK\_SUB\_DETAIL）

* 详细设计

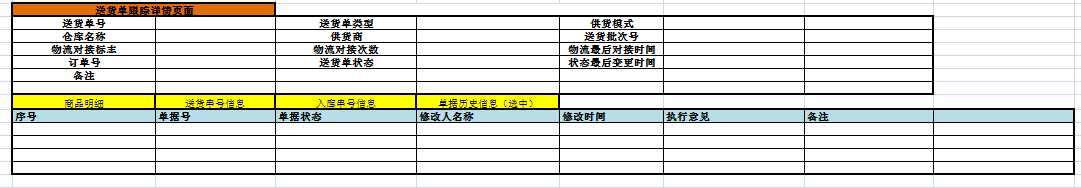
无

* 界面设计



6.1.2.2 -1





6.1.2.2 -2

#### 送货单统计

#### 送货单仓储校验

* 处理流程
* 功能简述

仓储物流操作人员对供货商送到仓储的手机串号进行校验.

根据选择的送货单号,导入手机终端串号,校验导入的串号是否在供应商生成送货单的时候有没有校验.

* 数据库表结构
* 详细设计

1. 查询待入库校验的送货单（见**送货单仓储校验查询页面）**

* 查询条件初始化

送货开始时间默认当前年月第一天+“00：00：00”

送货结束时间默认：当期时间

送货单号、订单号：支持模糊查询

仓储：下拉选择（默认全部）获取方法见：IOrder.getAllWarehouse

供应商：下拉（默认全部）方法见：IOrder.getAllWarehouse

* 查询

见方法：IDelivstockService.queryDelivWareCheck

主要查询表：TO\_BILL\_DELIVSTOCK\_MAIN

* 仓储串号校验

选择查询结果集中的一条送货单记录，进入送货单仓储校验页面。

1. 送货串号校验（见**送货单仓储校验页面**）

* 初始页面数据

根据送货单号获取送货单主表信息IDelivstockService. getOneDelivStockMain。

获取送货单商品信息：IDelivstockService.queryDelivstockSub;

* 串号校验

1. 选择导入的文件

文件为.XLS后缀的文件

文件内容格式是商品编码串号。

第一行是标题，第二行开始是具体的内容。

如

|  |  |
| --- | --- |
| 商品编码 | 串号 |
| JSYD-HTC-9308-01 | 989930203021 |
| JSYD-HTC-9308-01 | 989930203022 |

只要第一列或第二列为空，就忽略，遇到整行都是空直接忽略。

数据读入内存的时候需要去除左右空格。

1. 将文件中的数据读入内存中，如果读入的记录是0 提示用户，导入的数据为空或格式不正确，请重新导入文件。
2. 将内存中的数据写入到临时表中TEMP\_IMEI\_LOAD。

调用IOrder .saveDelivTempImei方法

导入批次号(F\_LOAD\_BATCH)：每次导入生成一个批次号（GUID），不是一个串号一个批次。F\_LOAD\_TYPE= CCCHJY

F\_ORDER\_NUM=送货单号。

1. 判断导入的商品串号数量是否大于供货商发货的数量。

IDelivstockService.checkWareCheckCount

参考SQL：

SELECT \* FROM TO\_BILL\_DELIVSTOCK\_SUB SUB, (

SELECT F\_GOODS\_NUM,COUNT(1) F\_COUNT FROM TEMP\_IMEI\_LOAD WHERE F\_BATCH\_NUM=? GROUP BY F\_GOODS\_NUM) TEMPIMEI where

SUB.F\_GOODS\_NUM = TEMP\_IMEI.f\_goods\_numandF\_BILL\_NUM=? AND

TEMPIMEI.F\_COUNT-SUB. F\_BILL\_AMOUNT>0。

如果查询记录集》0，表示存在导入的串号数量大于供应商发货的数量，提示用户”XXX商品导入串号数量XX,供货商发货数YYY，不能校验，请重新导入”

1. 保存串号校验结果（IDelivstockService.saveWareImeiCheck）

--判断不存在一级BOSS中的串号，并且更新串号状态=一级BOSS校验失败

UPDATE TEMP\_IMEI\_LOAD TEMPIMEI SET F\_IMEI\_CHECK\_STATUS =’BOSSFAIL’ WHERE F\_BATCH\_NUM=? AND F\_IMEI\_CHECK\_STATUS=’UNDO’ AND NOT EXISTS(SELECT 1 FROM TB\_BOSS\_IMEI BOSSIMEI WHERE F\_IMEI\_STATUS=1 AND TEMPIMEI.F\_GOODS\_IMEI=BOSSIMEI.F\_GOODS\_IMEI)

--判断存在一级BOSS中的串号，并且更新串号状态=一级BOSS校验成功

UPDATE TEMP\_IMEI\_LOAD TEMPIMEI SET F\_IMEI\_CHECK\_STATUS =’BOSSSUCC’ WHERE F\_BATCH\_NUM=? AND F\_IMEI\_CHECK\_STATUS=’UNDO’ AND EXISTS(SELECT 1 FROM TB\_BOSS\_IMEI BOSSIMEI WHERE F\_IMEI\_STATUS=1 AND TEMPIMEI.F\_GOODS\_IMEI=BOSSIMEI.F\_GOODS\_IMEI)

--判断省库状态，如果在省库不存在，并且更新串号状态= ‘PROVSUCC’;

UPDATE TEMP\_IMEI\_LOAD TEMPIMEI SET F\_IMEI\_CHECK\_STATUS =’ PROVSUCC’ WHERE F\_BATCH\_NUM=? AND F\_IMEI\_CHECK\_STATUS=’ BOSSSUCC’ AND NOT EXISTS(SELECT 1 FROM TB\_GOODS\_IMEI PROVIMEI WHERE TEMPIMEI.F\_GOODS\_IMEI=BOSSIMEI. F\_IMEI)

--判断省库状态，如果在省库存在但状态是退省库，更新串号状态= ‘PROVSUCC’;

UPDATE TEMP\_IMEI\_LOAD TEMPIMEI SET F\_IMEI\_CHECK\_STATUS =’ PROVSUCC’ WHERE F\_BATCH\_NUM=? AND F\_IMEI\_CHECK\_STATUS=’ BOSSSUCC’ AND F\_GOODS\_IMEI AND EXISTS(SELECT 1 FROM TB\_GOODS\_IMEI PROVIMEI WHERE TEMPIMEI.F\_GOODS\_IMEI=BOSSIMEI. F\_IMEI AND F\_IMEI\_STATUS=’ TSK’)

--其他改成省库校验失败

UPDATE TEMP\_IMEI\_LOAD TEMPIMEI SET F\_IMEI\_CHECK\_STATUS =’ PROVFAIL’ WHERE F\_BATCH\_NUM=? AND F\_IMEI\_CHECK\_STATUS=’ BOSSSUCC’

更新送货单子表明细中F\_IMEI\_STATUS状态。

参考SQL：

--如果子表中的串号在导入列表中，更新子表串号状态=核对正确。

UPDATE TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL subDetail set F\_IMEI\_STATUS =核对正确,F\_IS\_VALID=1 where f\_bill\_num=? exists(select 1 from temp\_imei\_load load where f\_batch\_num=? and subdetail.f\_goods\_num = load.f\_goods\_num and subdetail.f\_goods\_imei=load.f\_goods\_imei)。

--如果子表中的串号不在导入列表中，更新子表串号状态=导入未发货。

UPDATE TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL subDetail set F\_IMEI\_STATUS=导入未发货,F\_IS\_VALID=0 where f\_bill\_num=? not exists(select 1 from temp\_imei\_load load where f\_batch\_num=? and subdetail.f\_goods\_num = load.f\_goods\_num and subdetail.f\_goods\_imei=load.f\_goods\_imei)。

--如果串号在导入表中，不在送货单子表明细表中，子表串号状态=发货未导入。F\_IS\_VALID=0

INSERT INTO TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL(…发货未导入…)SELECT \* FROM temp\_imei\_load load WHERE F\_BATCH\_NUM=? NOT exists (SELECT 1 FROM TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL DETAIL WHERE DETAIL.F\_BILL\_NUM=? AND DETAIL.F\_GOODS\_NUM= LOAD.F\_GOODS\_NUM AND DETAIL.F\_GOODS\_IMEI=LOAD.F\_GOODS\_IMEI。

更新送货单子表明细表中F\_IMEI\_CHECK\_SATAUS状态

UPDATE TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL SUBDETAIL SET F\_IMEI\_CHECK\_STATUS=(SELECT F\_IMEI\_CHECK\_STATUS FROM TEMP\_IMEI\_LOAD TEMPIMEI TEMP WHERE SUBDETAIL.f\_GOODS\_IMEI =TEMP.F\_GOODS\_IMEI AND F\_BATCH\_NUM=?) AND SUBDETAIL.F\_BILL\_NUM=? AND EXISTS(SELECT 1 FROM TEMP\_IMEI\_LOAD TEMPIMEI TEMP2 WHERE SUBDETAIL.f\_GOODS\_IMEI =TEMP2.F\_GOODS\_IMEI AND F\_BATCH\_NUM=?)

更新商品状态和运货单状态

定义变量**送货单状态**=待入库

--获取导入的每个商品的IMEI的状态进行汇总

SELECT F\_GOODS\_NUM,COUNT(1) F\_COUNT F\_COUNT FROM TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL WHERE (F\_IMEI\_CHECK\_STATUS=’PROVFAIL’ or f\_imei\_check\_status=’BOSSFAIL’) and F\_BILL\_NUM=? GROUP BY F\_GOODS\_NUM

如果F\_COUNT >0 表示存在导入的串号校验失败的记录，记录**送货单状态=暂存**。并且更新TO\_BILL\_DELIVSTOCK\_SUB表中F\_GOODS\_STATUS=暂存，

更新送货单主表状态TO\_BILL\_DELIVSTOCK\_MAIN. F\_BILL\_STATUS\_NUM=**送货单状态**

**根据失败串号，生成串号调整单**

**新增串号调整单信息【串号调整表】（TO\_CHANGEIMEI），一个串号一条记录**

插入送货单操作日志表。

5、返回是否有校验结果IDelivstockService.getWareCheckResult。

XXX送货单，状态YYYY：合计导入串号数量为：XX个。

品牌，型号，商品SKU，状态，核对正确数量，导入未发货数量，发货未导入数量。

* 界面设计

**送货单仓储校验查询页面**



**送货单仓储校验页面**



#### 送货单暂存处理

* 处理流程



* 功能简述

1. 点击功能菜单后，打开【送货单暂存处理页面】（6.1.2.4 -1），用户点击【查询】按钮，根据查询条件将符合条件的单据信息查询出来并在列表中展现。
2. 查询条件说明：

* 送货单号、订单号、，后台模糊查询。
* 供货商、仓储（获取方法见上）
* 开始时间：默认当前年月第一天+”00:00:00”
* 结束时间，默认当前时间
* 需增加默认查询条件，【送货单状态】(F\_BILL\_STATUS\_NUM)：暂存。

调用方法IDelivstockService.queryDelivTempDo

1. 用户选择某条记录后，点击【暂存处理】按钮，跳转到【送货单暂处理详情页面】（6.1.2.4 -2）。
2. 用户单击某一行，可选中该条数据。
3. 只能单选。
4. 【送货单暂处理详情页面】（6.1.2.4-2），列出该送货单下所有商品，并标出状态.点击【查看待处理串号】跳转到【串号查看页面】（6.1.2.4 -3）。
5. 查询送货单暂存商品IDelivstockService.getTempGoodsInfo

SELECT F\_GOODS\_NUM, --商品编码

BOSSFAIL.F\_COUNT F\_BOSSFAIL,--BOSS校验失败数量

PROVFAIL.F\_COUNT F\_PROVFAIL,--省库校验失败数量

F\_BILL\_AMOUNT --发货数量

FROM TO\_BILL\_DELIVSTOCK\_SUB SUB

LEFT JOIN

(SELECT F\_GOODS\_NUM ,COUNT(1) F\_COUNT TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL WHERE F\_BILL\_NUM=?

AND F\_IMEI\_CHECK\_STATUS='BOSSFAIL'

GROUP BY F\_GOODS\_NUM) BOSSFAIL ON SUB.F\_GOODS\_NUM = BOSSFAIL.F\_SUB

LEFT JOIN

(SELECT F\_GOODS\_NUM ,COUNT(1) F\_COUNT TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL WHERE F\_BILL\_NUM=?

AND F\_IMEI\_CHECK\_STATUS='PROVFAIL'

GROUP BY F\_GOODS\_NUM)PROVFAIL ON SUB.F\_GOODS\_NUM = PROVFAIL.F\_SUB

WHERE SUB.F\_BILL\_NUM=?

【串号校验】IDelivstockService.doTempDelivStockBill

1. 判断是串号是否在一级BOSS中,

UPDATE TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL detail SET F\_CHECK\_IMEI\_STATUS= ‘BOSSSUCC’ where F\_CHECK\_IMEI\_STATUS=’BOSSFAIL’ and f\_bill\_num=’送货单号’ where exitis (select 1 from TB\_BOSS\_IMEI BOSSIMEI WHERE F\_IMEI\_STATUS=1 AND detail.F\_GOODS\_IMEI=BOSSIMEI.F\_GOODS\_IMEI)

1. 判断串号是否可入省库

UPDATE TO\_BILL\_DELIVSTOCK\_SUB\_DETAILdetail SET F\_IMEI\_CHECK\_STATUS =’ PROVSUCC’ WHERE f\_bill\_num=? AND (F\_IMEI\_CHECK\_STATUS=’ BOSSSUCC’ OR

F\_IMIE\_CHECK\_STATUS=’PROVFAIL’) AND NOT EXISTS(SELECT 1 FROM TB\_GOODS\_IMEI PROVIMEI WHERE detail.F\_GOODS\_IMEI=BOSSIMEI. F\_IMEI)

--判断省库状态，如果在省库存在但状态是退省库，更新串号状态= ‘PROVSUCC’;

UPDATE TO\_BILL\_DELIVSTOCK\_SUB\_DETAILdetail SET F\_IMEI\_CHECK\_STATUS =’ PROVSUCC’ WHERE f\_bill\_num =? AND (F\_IMEI\_CHECK\_STATUS=’ BOSSSUCC’ OR

F\_IMIE\_CHECK\_STATUS=’PROVFAIL’) AND EXISTS(SELECT 1 FROM TB\_GOODS\_IMEI PROVIMEI WHERE detail.F\_GOODS\_IMEI=BOSSIMEI. F\_IMEI AND F\_IMEI\_STATUS=’ TSK’)

1. 更新送货单子表商品状态

SELECT F\_GOODS\_NUM ,COUNT(1) F\_COUNT FROM TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL WHERE F\_BILL\_NUM=? AND F\_CHECK\_IMEI\_STATUS<>’PROVSUCC’ ,

如果 F\_COUNT<=0 更新TO\_BILL\_DELIVSTOCK\_SUB . F\_GOODS\_STATUS= 正常.

1. 更新送货单状态

SELECT COUNT(1) F\_COUNT FROMTO\_BILL\_DELIVSTOCK\_SUB WHERE F\_BILL\_NUM=? AND TO\_BILL\_DELIVSTOCK\_SUB . F\_GOODS\_STATUS<>2

如果F\_COUNT <=0 更新送货单单状态 = 待入库

1. 记录送货单更新日志

所有步骤需在同一个事务中完成。

执行完成后，刷新页面。

1. 【串号查看页面】（6.1.2.4 -3），将状态时BOSSFAIL 或 PROVFAIL的串号都列出来，根据【串号校验状态】，显示对应的状态提示。IDelivstockService.getTempDeilvGoodsImei
2. 串号校验状态：一级BOSS校验失败、省库校验失败。(见表TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL .F\_IMEI\_CHECK\_STATUS字段)
3. 当前状态：如果存在tb\_goods\_imei表中存在,现在是tb\_goods\_imei中的状态,如果不存在,显示为空.

导出：将查询结果导出成EXCEL。

* 数据库表结构

入库单主表（TO\_BILL\_DELIVSTOCK\_MAIN）

入库单子表（TO\_BILL\_DELIVSTOCK\_SUB）

入库单子表明细（TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL）

* 详细设计

无

* 界面设计
* 

6.1.2.4 -1



6.1.2.4 -2



6.1.2.4 -3

#### 送货单手工反馈

* 处理流程



* 功能简述

根据物流仓储实际入库的地串号信息,处理入库处理.

根据物流仓储提供的EXCEL文件,将文件导入到系统中,对导入的数据进行处理.

* 数据库设计
* 详细设计

1. **串号导入处理（见**串号导入页面**）**
2. 选择串号文件。

文件后缀格式为.XLS的EXCLE文件，

导入文件内容格式：商品编码、串号、仓库编码、送货单号

1. 确认提交

将EXCEL文件内容读入的内存中，如果读入记录数量为空，提示导入的文件的格式为空。所有字段都需要去空格处理，判断导入的送货单号是否重复，如果存在重复，提示用户“存在重复的送货单号”请分开导入。

1. 获取送货单主表信息（调用getOneDelivStockMain），如果送货单状态是待入

库校验，返回，提示用户，该入库单状态是待入库校验，不能上架。

1. 将内存导入数据保存到临时表，调用：IOrder.saveDelivTempImei。
2. **串号导入确认（见**串号导入确认页面**）**
3. 获取导入的串号信息，并且反馈到界面（**串号导入确认页面）。调用方法**

IDelivstockService.getInstoreLoadGoodsInfo。

SELECT

BRAND.F\_BRAND\_NAME, --品牌

CATE.F\_CATEGORY\_NAME, --品类

GOODS.F\_GOODS\_NAME, --商品名称

SUB.F\_BILL\_AMOUNT, --送货数量

SUB. F\_ACCEPT\_AMOUNT, --入库数量

TEMP.F\_COUNT --导入数量

FROM TO\_BILL\_DELIVSTOCK\_SUB SUB LEFT JOIN

(SELECT F\_GOODS\_NUM,COUNT(1) F\_COUNT FROM TEMP\_LOAD\_IME GROUP BY F\_GOODS\_NUM WHERE F\_BATCH\_NUM=?) TEMP ON SUB.F\_GOODS\_NUM = TEMP.F\_GOODS\_NUM

LEFT JOIN TB\_DA\_GOODS GOODS ON GOODS.F\_GOODS\_NUM = SUB.F\_GOODS\_NUM

LEFT JOIN TB\_DA\_CATEGORY CATE ON CATE.F\_CATEGORY\_NUM= GOODS. F\_CATEGORY\_NUM

LEFT JOIN TB\_DA\_BRAND BRAND ON BRAND. F\_BRAND\_NUM=CATE. F\_BRAND\_NUM

WHERE F\_BILL\_NUM=?

1. 获取送货主表信息（getOneDelivStockMain）
2. 获取导入明细信息（getInstoreLoadGoodsImei）

**SELECT temp.f\_goods\_imei,’已处理’ F\_IMEI\_STATUS TEMP\_LOAD\_IMEI TEMP**

**LEFT JOIN TB\_DA\_GOODS ON**

**LEFT JOIN TB\_DA\_CATEGORY**

**LEFT JOIN TB\_DA\_BRAND**

**where exitis (SELECT 1 FROM TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL subdetail WHERE F\_BILL\_NUM= ? and TEMP.F\_GOODS\_NUM = SUBDETAIL.F\_GOODS\_NUM AND TEMP.F\_GOODS\_IMEI=SUBDETAIL.F\_GOODS\_IMEI)**

**UNION**

**SELECT temp.f\_goods\_imei,’未处理’ F\_IMEI\_STATUS TEMP\_LOAD\_IMEI TEMP**

**LEFT JOIN TB\_DA\_GOODS ON**

**LEFT JOIN TB\_DA\_CATEGORY**

**LEFT JOIN TB\_DA\_BRAND**

**where NOT exitis (SELECT 1 FROM TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL subdetail WHERE F\_BILL\_NUM= ? and TEMP.F\_GOODS\_NUM = SUBDETAIL.F\_GOODS\_NUM AND TEMP.F\_GOODS\_IMEI=SUBDETAIL.F\_GOODS\_IMEI)**

1. **确认导入IDelivstockService .saveGoodsInstore**

* **生成入库单信息**

**操作表**

**TO\_BILL\_INSTOCK\_MAIN，TO\_BILL\_INSTOCK\_SUB,TO\_BILL\_INSTOCK\_SUB\_DETAIL**

**插入TO\_BILL\_INSTOCK\_SUB\_DETAIL考虑用**

**insert into TO\_BILL\_INSTOCK\_SUB\_DETAIL SELECT FROM TEMP\_IMEI\_LOAD 。**

* **更改TB\_GOODS\_IMEI状态（参考SQL）**

**MERG INTO TB\_GOODS\_IMEI IMEI**

**USING（SELECT \* FROM TEMP\_IMEI\_LOAD WHERE F\_BATCH\_NUM=?） TEMP**

**ON(IMEI.F\_GOODS\_IMEI=TEMP.F\_GOODS\_IMEI)**

**WHEN MATCHED THEN**

**UPDATE**

**SET IMEI.F\_GOODS\_NUM=TEMP.F\_GOODS\_NUM,**

**IMEI.F\_REGION\_NUM=99,**

**F\_SUPPLIER\_NUM=?**

**F\_SUPP\_TYPE=1,**

**F\_ORG\_NUM=统一运营对应的组织机构编码,**

**F\_IMEI\_STATUS=在库**

**F\_STOCK\_TYPE\_NUM=1**

**F\_STOCK\_MODE=?**

**F\_WARE\_NUM=?**

**F\_IS\_USE=1，**

**F\_ORDER\_NUM=采购订单，**

**F\_ORDER\_PRICE=？，**

**F\_PURCHPRICE\_ID=?，**

**F\_INWARE\_BILL\_NUM=入库单号**

**F\_INWARE\_TIME=？，**

**F\_IS\_FREEZE=0，**

**F\_LAST\_MANCODE=？，**

**F\_LAST\_MANNAME=？，**

**F\_LAST\_TIME=？**

**WHEN NOT MATCHED THEN**

**INSERT(F\_IMEI,F\_REGION\_NUM=99,F\_GOODS\_NUM,F\_SUPPLIER\_NUM,F\_SUPP\_TYPE=1,F\_ORG\_NUM(统一运营对应的组织机构编码),F\_IMEI\_STATUS（在库）,F\_PURCHPRICE\_ID,F\_STOCK\_TYPE\_NUM=1,F\_STOCK\_MODE,F\_WARE\_NUM,F\_IS\_USE=1,F\_ORDER\_NUM（采购订单号）,F\_ORDER\_PRICE,F\_INWARE\_BILL\_NUM（入库单号）,F\_INWARE\_TIME,F\_IS\_FREEZE（0）,F\_LAST\_MANCODE,F\_LAST\_MANNAME**,**F\_LAST\_TIME)**

**--插入IMEI日志**

**INSERT INTO TB\_GOODS\_IMEI\_LOG(……)**

**SELECT \* FROM TB\_GOODS\_IMEI IMEI WHERE EXISTS( SELECT 1 FROM TEMP\_IMEI\_LOAD TEMP WHERE IMEI.F\_IMEI = TEMP.F\_GOODS\_IMEI AND TEMP.F\_BATCH\_NUM =?)**

* **更新库存异动表**

**新增TO\_WARE\_STOCK\_TRACE记录**

* **更新分配库存信息（参考SQL）**

**如果是TO\_BILL\_INSTOCK\_MAIN.F\_BILL\_TYPE=’1-采购入库’执行以下**

**MERGE INTO TO\_STOCK\_REALTIME STOCK**

**USING (SELECT F\_GOODS\_NUM ,COUNT(1) F\_COUNT FROM TEMP\_IMEI\_LOAD WHERE F\_BATCH\_NUM=? GROUP BY F\_GOODS\_NUM) TEMP**

**ON STOCK.F\_GOODS\_NUM = TEMP.F\_GOODS\_NUM AND F\_WARE\_NUM=’仓库编码’ and F\_STOCK\_TYPE\_NUM=’ZPK’ and F\_ORG\_NUM=’统一运营对应的组织机构编码’ and F\_LOCATE\_NUM=’0000’**

**WHEN NOT MATCHED THEN**

**INSERT (F\_ORG\_NUM,F\_LOCATE\_NUM,F\_WARE\_NUM,F\_GOODS\_NUM,F\_STOCK\_TYPE\_NUM,F\_STOCK\_COUNT,F\_STOCK\_MONEY,F\_DIST\_COUNT)**

**(‘统一运营对应的组织机构编码’,’0000’,’仓库编码’,TEMP.F\_GOODS\_NUM,’ZPK’，temp.f\_count,0,temp.f\_count);**

**WHEN MATCHED THEN**

**UPDATE**

**SET STOCK.F\_DIST\_COUNT = F\_DIST\_COUNT+temp.f\_count,**

**STOCK.F\_STOCK\_COUNT= STOCK.F\_STOCK\_COUNT+temp.f\_count**

**增加可分配库存数量日志TO\_STOCKDIST\_ACTION插入记录**

**备注=”进货可分配库存量增加“**

* **更新仓库进销存表（参考SQL）**

**MERGE INTO TO\_WARE\_PSSM PSSM**

**USING**

**(SELECT F\_GOODS\_NUM ,COUNT(1) F\_COUNT FROM TEMP\_IMEI\_LOAD WHERE F\_BATCH\_NUM=? GROUP BY F\_GOODS\_NUM) TEMP**

**ON PSSM.F\_GOODS\_NUM =TEMP.F\_GOODS\_NUM AND F\_WARE\_NUM=’仓库编码’ and F\_LOCATE\_NUM=’0000’ and F\_SUPPLIER\_NUM=’供应商编码’**

**AND F\_PSSM\_DATE=to\_char(sysdate,’yyyyMM’)**

**When matched then**

**UPDATE**

**SET F\_INSTORE\_COUNT=F\_INSTORE\_COUNT +TEMP.F\_COUNT,**

**F\_INSTORE\_MONEY = F\_INSTORE\_MONEY +TEMP.F\_COUNT \* 采购进价(采购进价取送货单的采购进价),**

**F\_INSTORE\_COST = F\_INSTORE\_COST+‘如果是买断=TEMP.F\_COUNT \* 采购进价;否则为0’;**

**F\_INSTORE\_TAX = F\_INSTORE\_TAX+采购金额\*税率(0.17);**

**When not matched then**

**INSERT(F\_PSSM\_DATE,F\_WARE\_NUM,F\_LOCATE\_NUM,F\_GOODS\_NUM,F\_SUPPLIER\_NUM,F\_INSTORE\_COUNT,F\_INSTORE\_MONEY,F\_INSTORE\_COST,F\_INSTORE\_TAX)……。**

* **更改订单信息（参考SQL）**

**---更改履行数量**

**UPDATE TO\_ORDERPURCH\_SUB SUB SET F\_ACCEPT\_AMOUNT= F\_ACCEPT\_AMOUNT+(**

**SELECT F\_COUNT (F\_PSSM\_DATE,F\_WARE\_NUM,**

**SELECT F\_GOODS\_NUM ,COUNT(1) F\_COUNT FROM TEMP\_IMEI\_LOAD WHERE F\_BATCH\_NUM=? GROUP BY F\_GOODS\_NUM) TEMP**

**WHERE SUB.F\_GOODS\_NUM= TEMP.F\_GOODS\_NUM**

**)**

**WHERE F\_ORDER\_NUM=’订单号’and EXISTS(SELECT 1 FROM TEMP\_IMEI\_LOAD LOAD WHERE LOAD.F\_GOODS\_NUM =SUB.F\_GOODS\_NUM AND F\_BATCH\_NUM=?)**

**--更新采购订单主表**

**UPDATE TO\_ORDERPURCH\_MAIN SET F\_ACCEPT\_LASTBILL=’入库单号’ and F\_ACCEPT\_LASTDATE=to\_char(sysdate,’yyyyMMddhh24miss’) where f\_order\_num=’订单号’**

* **更新送货单信息（参考SQL）**

**--更新送货单入库数量**

**UPDATE TO\_BILL\_DELIVSTOCK\_SUB SUB SET F\_ACCEPT\_AMOUNT= F\_ACCEPT\_AMOUNT+(**

**SELECT F\_COUNT (**

**SELECT F\_GOODS\_NUM ,COUNT(1) F\_COUNT FROM TEMP\_IMEI\_LOAD WHERE F\_BATCH\_NUM=? GROUP BY F\_GOODS\_NUM) TEMP**

**WHERE SUB.F\_GOODS\_NUM= TEMP.F\_GOODS\_NUM**

**) WHERE F\_BILL\_NUM=’送货单号’ and**

**EXISTS(SELECT 1 FROM TEMP\_IMEI\_LOAD LOAD WHERE LOAD.F\_GOODS\_NUM =SUB.F\_GOODS\_NUM AND F\_BATCH\_NUM=?)**

**--更新送货单状态**

**UPDATE TO\_BILL\_DELIVSTOCK\_MAIN SET F\_BILL\_STATUS\_NUM= ‘已入库’ where f\_bill\_num=’送货单号’**

**--增加单据操作日志**

* 界面设计

**串号导入页面**



**串号导入确认页面**



### （重点）仓储配送管理

#### 配送单手工发送

* 处理流程

无

* 功能简述

1. 点击功能菜单后，打开【配送单手工发送页面】（图 -1），用户点击【查询】按钮，根据查询条件将符合条件的单据信息查询出来并在列表中展现。
2. 查询条件说明：

* 配送单号、要货单位，后台模糊查询。
* 开始时间、结束时间，提供日期选择控件。
* 需增加默认查询条件，【物流对接标记】(F\_LOGIS\_FLAG)：0。

1. 用户可多选，选择记录后，点击【手工同步】按钮，提示同步结果，并刷新列表。
2. 用户单击某一行，可选中该条数据。
3. 提供全选。
4. 可多选。
5. 调用同步接口，同步数据。（暂时没有该接口，代码预留）
6. 每次调用接口前更新【物流对接次数】（F\_LOGIS\_COUNT），【物流最后对接时间】（F\_LOGIS\_TIME），若调用接口成功曾更新【物流对接标记】（F\_LOGIS\_FLAG）标记为：1。
7. 用户选中某条记录，点击【查看详情】按钮，跳转到【配送单手工发送详情页面】（图-2）。
8. 只能查看详细信息，不能修改。
9. 选中多条记录，默认展现选中记录的第一条。

* 数据库表结构

配送单主表（TO\_BILL\_SENDSTOCK\_MAIN）

配送单子表（TO\_BILL\_SENDSTOCK\_SUB）

配送单子表明细（TO\_BILL\_SENDSTOCK\_SUB\_DETAIL）

* 详细设计

无

* 界面设计

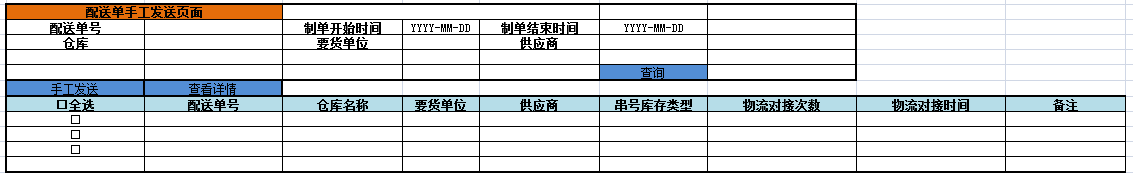


图 -1

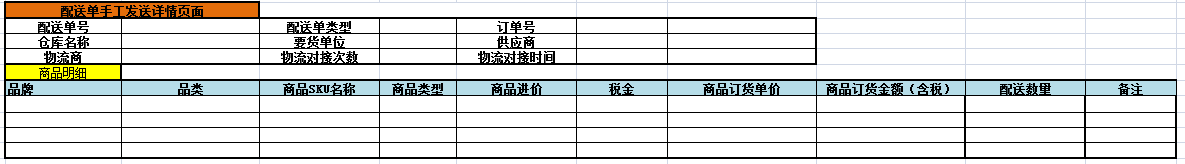


图 -2

#### 配送单出库反馈

* 处理流程
* 功能简述

1. 点击功能菜单后，打开【串号导入页面】（图 -1）
2. 选择串号文件。

文件后缀格式为.XLS的EXCLE文件，

导入文件内容格式：商品编码、串号、仓库编码、物流单号、配送单号

1. 确认提交

将EXCEL文件内容读入的内存中，如果读入记录数量为空，提示导入的文件的格式为空。所有字段都需要去空格处理，判断导入的配送单号是否一致，如果不一致，提示用户“存在多个的配送单号”请分开导入。

1. 获取配送单主表信息（调用：ISendstockService. getSendStockMain）如果配送单状态不是【待出库】，则返回，提示用户“该配送单已经完成配送”。
2. 将内存导入数据保存到临时表，（调用：ISendstockService.saveTempSendImeiLoad）。

--F\_IMEI\_DELIV\_STATUS默认记录1

1. 校验串号，修改串号状态

*--修改校验不通过的串号状态*

UPDATE TEMP\_SEND\_IMEI\_LOAD TEMP

SET TEMP.F\_IMEI\_DELIV\_STATUS=’2’ ,TEMP.F\_IMEI\_MEMO='串号状态校验失败'

WHEREEXISTS

(SELECT SEND.F\_GOODS\_IMEI FROM TEMP\_SEND\_IMEI\_LOAD SEND

LEFTJOIN TB\_GOODS\_IMEI IMEI ON SEND.F\_GOODS\_IMEI = IMEI.F\_IMEI

WHERE SEND.F\_LOAD\_BATCH = ?AND IMEI.F\_IMEI\_STATUS != 'ZK'

)

*--如果串号在档案表中查询不到，记录TEMP.F\_IMEI\_MEMO='串号不存在'*

1. 【串号导入页面】点击确定后，跳转到【串号导入确认页面】（图 -3）
2. 获取串号信息，并反馈到界面

参考SQL：

SELECT

BRAND.F\_BRAND\_NAME, *--品牌*

CATE.F\_CATEGORY\_NAME, *--品类*

GOODS.F\_GOODS\_NAME, *--商品名称*

SUB.F\_BILL\_AMOUNT, *--配送数量*

TEMP.F\_COUNT *--导入数量*

SUCC.F\_COUNT\_SUCC *--校验通过数量*

FROM TO\_BILL\_SENDSTOCK\_SUB SUB

LEFTJOIN (SELECT F\_GOODS\_NUM,COUNT(1) F\_COUNT FROM TEMP\_SEND\_IMEI\_LOAD GROUPBY F\_GOODS\_NUM WHERE F\_LOAD\_BATCH=? AND F\_SEND\_NUM =?) TEMP

ON SUB.F\_GOODS\_NUM = TEMP.F\_GOODS\_NUM AND SUB.F\_BILL\_NUM = TEMP.F\_SEND\_NUM

LEFTJOIN (SELECT F\_GOODS\_NUM,COUNT(1) F\_COUNT\_SUCC FROM TEMP\_SEND\_IMEI\_LOAD GROUPBY F\_GOODS\_NUM

WHERE F\_LOAD\_BATCH=? AND F\_SEND\_NUM =? AND F\_IMEI\_DELIV\_STATUS = 1) SUCC

ON SUB.F\_GOODS\_NUM = SUCC.F\_GOODS\_NUM AND SUB.F\_BILL\_NUM = SUCC.F\_SEND\_NUM

LEFTJOIN TB\_DA\_GOODS GOODS ON GOODS.F\_GOODS\_NUM = SUB.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_CATEGORY CATE ON CATE.F\_CATEGORY\_NUM= GOODS. F\_CATEGORY\_NUM

LEFTJOIN TB\_DA\_BRAND BRAND ON BRAND. F\_BRAND\_NUM=CATE. F\_BRAND\_NUM

WHERE SUB.F\_BILL\_NUM=?

1. 获取配送单主表信息（调用：ISendstockService. getSendStockMain）。
2. 获取导入明细信息

参考SQL：

SELECT

BRAND.F\_BRAND\_NAME, *--品牌*

CATE.F\_CATEGORY\_NAME, *--品类*

GOODS.F\_GOODS\_NAME, *--商品名称*

TEMP.F\_GOODS\_IMEI *--串号*

TEMP.F\_IMEI\_DELIV\_STATUS *--串号校验结果*

FROM TEMP\_SEND\_IMEI\_LOAD TEMP

LEFTJOIN TB\_DA\_GOODS GOODS ON GOODS.F\_GOODS\_NUM = TEMP.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_CATEGORY CATE ON CATE.F\_CATEGORY\_NUM= GOODS.F\_CATEGORY\_NUM

LEFTJOIN TB\_DA\_BRAND BRAND ON BRAND. F\_BRAND\_NUM=CATE. F\_BRAND\_NUM

WHERE TEMP.F\_LOAD\_BATCH=? AND TEMP.F\_SEND\_NUM=?

1. 【串号导入确认页面】（图 -3），点击【确认导入】。
2. 更改配送单信息

更新配送单子表信息，商品进价、税金

--商品进价=串号采购价之和/数量

1. 生成出库单信息。

操作表：

出库单主表（TO\_BILL\_OUTSTOCK\_MAIN）

出库单子表（TO\_BILL\_OUTSTOCK\_SUB）

出库单子明细表（TO\_BILL\_OUTSTOCK\_SUB\_DETAIL）

--更新子单明细需关联串号档案表，查出该串号的采购单号

1. **根据校验失败串号，生成串号调整单**

**新增串号调整单信息【串号调整表】（TO\_CHANGEIMEI），一个串号一条记录**

1. 更改TB\_GOODS\_IMEI状态

参考SQL ：

--更改串号归属、状态及库存类型

update TB\_GOODS\_IMEI imei set imei.f\_org\_num = ? and imei.f\_imei\_status ='PSZT'and imei. F\_STOCK\_TYPE\_NUM =?–-1-正品库、2-次品库、3-备机库

whereexists

(select temp.F\_GOODS\_IMEI form TEMP\_SEND\_IMEI\_LOAD temp wheretemp.F\_LOAD\_BATCH=?)

--增加串号操作日志

INSERTINTO TB\_GOODS\_IMEI\_LOG(……)

SELECT \* FROM TB\_GOODS\_IMEI IMEI WHEREEXISTS

( SELECT1FROM TEMP\_SEND\_IMEI\_LOAD TEMP WHERE IMEI.F\_IMEI = TEMP.F\_GOODS\_IMEI AND TEMP.F\_BATCH\_NUM =?)

1. 更新库存异动表

新增TO\_WARE\_STOCK\_TRACE记录

1. 更新仓库进销存表

参考SQL：

*--自办厅配送*

MERGEINTOTC\_ORG\_PSSM PSSM

USING

(SELECTF\_GOODS\_NUM ,COUNT(1) F\_COUNT FROM TEMP\_SEND\_IMEI\_LOAD WHERE F\_LOAD\_BATCH=? GROUPBY F\_GOODS\_NUM) TEMP

ON PSSM.F\_GOODS\_NUM =TEMP.F\_GOODS\_NUM AND F\_GOODS\_TYPE = '商品类型'

AND F\_OCCUR\_DATE=to\_char(sysdate,’yyyyMMdd’) AND F\_REGION\_NUM='99'AND F\_ORG\_NUM='仓库编码'AND F\_ORG\_TYPE='1'

Whenmatchedthen

UPDATE

SET

F\_OUT\_DIST\_COUNT=F\_OUT\_DIST\_COUNT +TEMP.F\_COUNT, *--配出数量*

F\_OUT\_DIST\_COST = F\_OUT\_DIST\_COST +商品订货金额（含税)(取配送单的商品订货金额),*--配出成本*

F\_OUT\_DIST\_MARGIN = 0,*--配出差价*

F\_OUT\_DIST\_TAX = 0;*--配出税金*

Whennotmatchedthen

INSERT(F\_REGION\_NUM,F\_ORG\_NUM,F\_ORG\_TYPE,F\_OCCUR\_DATE,F\_GOODS\_NUM)……。

*--合作厅配送*

MERGEINTO TO\_WARE\_PSSM PSSM

USING

(SELECTF\_GOODS\_NUM ,COUNT(1) F\_COUNT FROM TEMP\_SEND\_IMEI\_LOAD WHERE F\_LOAD\_BATCH=? GROUPBY F\_GOODS\_NUM) TEMP

ON PSSM.F\_GOODS\_NUM =TEMP.F\_GOODS\_NUM AND F\_GOODS\_TYPE = '商品类型'AND F\_SUPPLIER\_NUM=’供应商编码’

AND F\_OCCUR\_DATE=to\_char(sysdate,’yyyyMM’) AND F\_REGION\_NUM='地市编码'AND F\_ORG\_NUM='营业厅编码'AND F\_ORG\_TYPE='1'

Whenmatchedthen

UPDATE

SET

F\_INSALE\_COUNT=F\_INSALE\_COUNT +TEMP.F\_COUNT, *--内部销售数量*

F\_INSALE\_INCOME = F\_INSALE\_INCOME +商品订货金额（含税) \*（1-销项税率）,*----内部销售收入（无税），取配送单子表商品订货金额（含税)，销项税率取商品品类档案表销项税率*

F\_INSALE\_COST=商品进价,*--内部销售成本，去配送单子表商品进价*

F\_INSALE\_PROFIT =内部销售收入（无税）-商品进价,*--内部销售毛利*

F\_INSALE\_AGION= 0 ,*--内部销售折扣*

F\_INSALE\_TAX =商品订货金额（含税)\*销项税率;*--内部销售税金*

Whennotmatchedthen

INSERT(F\_REGION\_NUM,F\_ORG\_NUM,F\_ORG\_TYPE,F\_OCCUR\_DATE,F\_GOODS\_NUM)……

1. 更新配送单信息

参考SQL：

--更新配送单子表出库数量

UPDATE TO\_BILL\_SENDSTOCK\_SUB SUB SET F\_REAL\_AMOUNT= F\_REAL\_AMOUNT+(

SELECT F\_COUNT from (

SELECTF\_GOODS\_NUM ,COUNT(1) F\_COUNT FROM TEMP\_SEND\_IMEI\_LOAD WHERE F\_BATCH\_NUM=? GROUPBY F\_GOODS\_NUM) TEMP

WHERE SUB.F\_GOODS\_NUM= TEMP.F\_GOODS\_NUM

) WHERE F\_BILL\_NUM=’配送单号’and

EXISTS(SELECT1FROM TEMP\_SEND\_IMEI\_LOAD LOADWHERE LOAD.F\_GOODS\_NUM =SUB.F\_GOODS\_NUM AND F\_LOAD\_BATCH=?)

--更新子表明细表信息

--更新配送单状态

UPDATE TO\_BILL\_SENDSTOCK\_SUB SET F\_BILL\_STATUS\_NUM= ‘在途’ where f\_bill\_num=’配送单号’

--增加单据操作日志

1. 更新要货订单状态。

--订单状态

根据订单类型取对应的订单状态编码

update TO\_ORDERCOMMAND\_MAIN ordem

set ordem.F\_ORDER\_STATUSCODE='待签收'

where ordem.F\_ORDER\_NUM=?

--增加订单日志

* 数据库表结构
* 详细设计
* 界面设计



图 -1

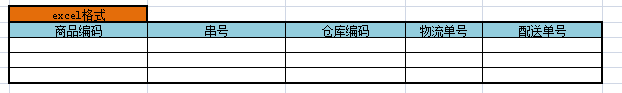


图 -2



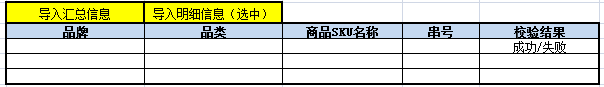


图 -3

#### 配送单签收反馈

* 处理流程
* 功能简述
* 数据库表结构
* 详细设计
* 界面设计

#### 配送单跟踪

* 处理流程

无

* 功能简述

1. 点击功能菜单后，打开【配送单跟踪页面】（图 -1），用户点击【查询】按钮，根据查询条件将符合条件的单据信息查询出来并在列表中展现。
2. 查询条件说明：

* 配送单号、要货单位，后台模糊查询。
* 仓库：(见订单管理组件IOrder.getAllWarehouse)
* 开始时间、结束时间，提供日期选择控件。
* 配送单状态：下拉列表
* 要货单位默认条件，省工号登陆，可以输入；营业厅工号登陆，要货单位默认登陆营业厅，不可修改。

1. 用户选择某条记录后，点击【查看详情】按钮，跳转到【配送单跟踪详情页面】（6.1.4.3 -2）。
2. 用户单击某一行，可选中该条数据。
3. 只能单选。

* 数据库表结构

配送单主表（TO\_BILL\_SENDSTOCK\_MAIN）

配送单子表（TO\_BILL\_SENDSTOCK\_SUB）

配送单子表明细（TO\_BILL\_SENDSTOCK\_SUB\_DETAIL）

* 详细设计

无

* 界面设计



图 -1

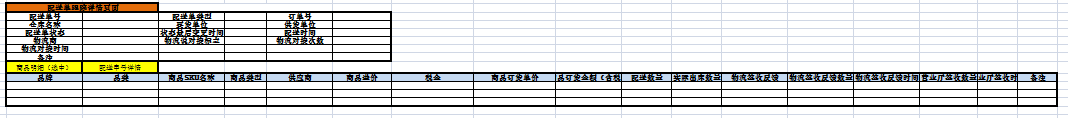




图 -2

### 仓储库存管理

#### 仓储出入库管理

##### 入库单查询

* 处理流程

无

* 功能简述

1. 点击功能菜单后，打开【入库单查询页面】（6.1.4.1.1 -1），用户点击【查询】按钮，根据查询条件将符合条件的单据信息查询出来并在列表中展现。
2. 查询条件说明：

入单号、订单号后台模糊查询

供货商：（见订单管理组件IOrder.getUnityDistSupplier）

仓库：(见订单管理组件IOrder.getAllWarehouse)

开始时间、结束时间，提供日期选择控件。

开始时间默认：当前年月第一天+“00：00：00”；

结束时间默认当前时间。

1. 用户选择某条记录后，点击【查看详情】按钮，跳转到【入库单查询详情页面】（6.1.4.1.1-2）。
2. 用户单击某一行，可选中该条数据。
3. 只能单选。
4. 【入库单查询详情页面】（6.1.4.1.1 -2），上半步展现送货单基本信息，下半部展现商品明细、串号信息

* 数据库表结构

入库单主表（TO\_BILL\_INSTOCK\_MAIN）

入库单子表（TO\_BILL\_INSTOCK\_SUB）

入库单子表明细（TO\_BILL\_INSTOCK\_SUB\_DETAIL）

仓库档案表（TB\_DA\_WAREHOUSE）

* 详细设计

无

* 界面设计



6.1.4.1.1 -1





6.1.4.1.1 -2

##### 入库单明细查询

* 处理流程

无

* 功能简述

查询入库单明细信息

* 数据库表结构
* 详细设计

入单号、订单号后台模糊查询

供货商：（见订单管理组件IOrder.getUnityDistSupplier）

仓库：(见订单管理组件IOrder.getAllWarehouse)

开始时间、结束时间，提供日期选择控件。

开始时间默认：当前年月第一天+“00：00：00”；

结束时间默认当前时间。

品牌，品类，商品SKU，三级联动处理。

查看串号。进入查看串号页面。

根据入库单号，商品编码查询入库单串号表

* 界面设计



C:\Users\sunky\AppData\Roaming\Tencent\Users\11350631\QQ\WinTemp\RichOle\BW)R`0U$9F8QF3N}L{R8AZ5.jpg

##### 出库单查询

* 处理流程

无

* 功能简述

1. 点击功能菜单后，打开【出库单查询页面】（6.1.4.1.2-1），用户点击【查询】按钮，根据查询条件将符合条件的单据信息查询出来并在列表中展现。
2. 查询条件说明：

* 出库单号、要货单位、商品名称、配送单号、订单号，后台模糊查询。
* 仓库(见订单管理组件IOrder.getAllWarehouse)
* 供货商（见订单管理组件IOrder.getUnityDistSupplier）
* 开始时间、结束时间，提供日期选择控件。

开始时间默认：当前年月第一天+“00：00：00”；

结束时间默认当前时间。

1. 用户选择某条记录后，点击【查看详情】按钮，跳转到【出库单查询详情页面】（6.1.4.1.2 -2）。
2. 用户单击某一行，可选中该条数据。
3. 只能单选。
4. 【出库单查询详情页面】（6.1.4.1.2 -2），上半步展现出库单基本信息，下半部展现商品明细、串号信息。

* 数据库表结构

出库单主表（TO\_BILL\_OUTSTOCK\_MAIN）

出库单子表（TO\_BILL\_OUTSTOCK\_SUB）

出库单子表明细（TO\_BILL\_OUTSTOCK\_SUB\_DETAIL）

仓库档案表（TB\_DA\_WAREHOUSE）

* 详细设计

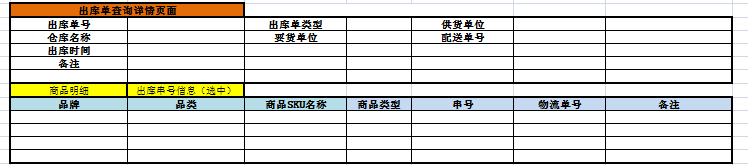
无

* 界面设计



6.1.4.1.2 -1





6.1.4.1.2 -2

##### 出库单明细查询

* 处理流程

无

* 功能简述

查询出库单明细信息

* 数据库表结构

出库单主表（TO\_BILL\_OUTSTOCK\_MAIN）

出库单子表（TO\_BILL\_OUTSTOCK\_SUB）

出库单子表明细（TO\_BILL\_OUTSTOCK\_SUB\_DETAIL）

仓库档案表（TB\_DA\_WAREHOUSE）

* 详细设计

出库单号、订单号后台模糊查询

供货商：（见订单管理组件IOrder.getUnityDistSupplier）

仓库：(见订单管理组件IOrder.getAllWarehouse)

开始时间、结束时间，提供日期选择控件。

开始时间默认：当前年月第一天+“00：00：00”；

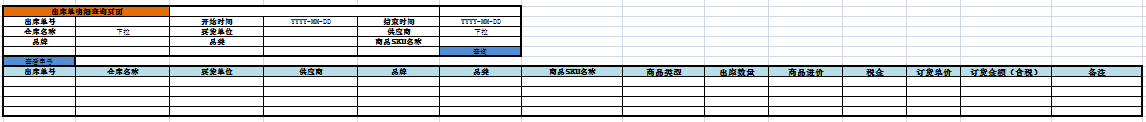
结束时间默认当前时间。

品牌，品类，商品SKU，三级联动处理。

查看串号。进入查看串号页面。

根据出库单号，商品编码查询出库单串号表

* 界面设计





#### 仓储库存统计

##### 仓储库存查询

* 处理流程

1. 根据查询条件查询满足条件的数据集
2. 展示查询结果，显示合计库存数量、库存金额、
3. 导出查询结果。

* 功能简述

查询当前仓库实际库存情况。

* 数据库表结构

TO\_STOCK\_REALTIME。

* 详细设计

1. 查询界面初始化
2. 仓储：下拉选择，默认（全部）IOrder.getAllWarehouse
3. 库存类型：下拉选择，默认(全部) IOrder.getStockType;
4. 品牌，品类、商品SKU，联动选择。
5. 查询结果集

（见表TO\_STOCK\_REALTIME）。

1. 导出查询结果集

将查询结果保存到EXCEL.

* 界面设计
* 

##### 仓储库存异动

* 处理流程

1. 根据查询条件查询满足条件的数据集

2、展示查询结果，分页显示。

3、导出查询结果。

* 功能简述

查询仓库库存的异动情况

* 数据库表结构

TO\_WARE\_STOCK\_TRACE。

* 详细设计

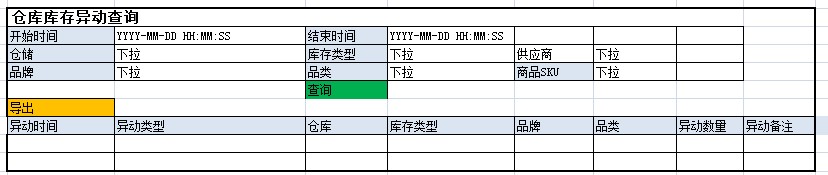
1. 查询界面初始化
2. 仓储：下拉选择，默认（全部）IOrder.getAllWarehouse
3. 库存类型：下拉选择，默认(全部) IOrder.getStockType;
4. 品牌，品类、商品SKU，联动选择。
5. 开始时间：默认当前年月一天+“00：00：”;
6. 结束时间：系统当前时间
7. 查询结果集

（见表TO\_WARE\_STOCK\_TRACE）。

1. 导出查询结果集

将查询结果保存到EXCEL.

* 界面设计



##### 仓储进销存明细

* 处理流程
* 功能简述

查询仓库进销存数据

* 数据库表结构

TC\_ORG\_PSSM。

* 详细设计

1. 查询界面初始化
2. 仓储：下拉选择，默认（全部）IOrder.getAllWarehouse
3. 库存类型：下拉选择(正常商品,备机) 默认正常商品;
4. 品牌，品类、商品SKU，联动选择。
5. 供应商：IOrder.getUnityDistSupplier
6. 开始日期：当前日期;
7. 结束结束：当前日期
8. 查询结果集

日期条件必须输入。开始日期不能大于结束日期

1. 导出查询结果集

将查询结果保存到EXCEL.

**[参考SQL]**

SELECT

F\_ORG\_NUM,

PSSM.F\_GOODS\_TYPE ,*--库存类型*

WARE.F\_WARE\_NAME F\_ORG\_NAME,

F\_OCCUR\_DATE,*--发生日期*

PSSM.F\_GOODS\_NUM, *--商品编码*

GOODSDA.F\_BRAND\_NAME,*--品牌名称*

GOODSDA.F\_GOODS\_MODEL,*--型号名称*

GOODSDA.F\_SKU\_ATTR\_NAME,*--商品SKU名称*

F\_FIRST\_COUNT,*--期初数量*

F\_FIRST\_COST, *--期初金额（无税）*

F\_PURCH\_COUNT + F\_PURCH\_COUNT\_PH , *--进货数量*

F\_PURCH\_COST + F\_PURCH\_COST\_PH,*--进货金额（无税）*

F\_OUT\_DIST\_COUNT+F\_INSALE\_COUNT,*--出库数量*

F\_OUT\_DIST\_COST+F\_INSALE\_COST, *--出库金额*

F\_COUNTPROFIT\_COUNT,*--盘升数量*

F\_COUNTPROFIT\_COST,*--盘升金额*

F\_COUNTLOSS\_COUNT,*--盘耗数量*

F\_COUNTLOSS\_COST,*--盘耗金额*

F\_DIALIN\_COUNT,*--拨入数量*

F\_DIALIN\_COST,*--拨入金额*

F\_DIALOUT\_COUNT,*--拨出数量*

F\_DIALOUT\_COST,*--拨出数量*

F\_ADJUST\_COUNT,*--调整数量*

F\_ADJUST\_COST,*--调整金额*

F\_FIRST\_COUNT+F\_PURCH\_COUNT + F\_PURCH\_COUNT\_PH +F\_COUNTPROFIT\_COUNT+F\_DIALIN\_COUNT+F\_ADJUST\_COUNT

-F\_OUT\_DIST\_COUNT+F\_INSALE\_COUNT-F\_COUNTLOSS\_COUNT-F\_DIALOUT\_COUNT,

*--期末数量 = 期初+进货+铺货进货+盘升+拨入+调整-配送出-内部销售-盘耗-拨出*

F\_FIRST\_COST+F\_PURCH\_COST + F\_PURCH\_COST\_PH +F\_COUNTPROFIT\_COST+F\_DIALIN\_COST+F\_ADJUST\_COST

-F\_OUT\_DIST\_COST+F\_INSALE\_COST-F\_COUNTLOSS\_COST-F\_DIALOUT\_COST

FROM TC\_ORG\_PSSM PSSM

LEFTJOIN VIEW\_GOODS\_BRAND\_MODEL GOODSDA ON PSSM.F\_GOODS\_NUM = GOODSDA.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_WAREHOUSE WARE ON WARE.F\_WARE\_NUM = PSSM.F\_ORG\_NUM

WHERE F\_REGION\_NUM='99'*--（固定）*

AND F\_ORG\_TYPE=1*--（固定）*

AND F\_GOODS\_TYPE=1*--（传入）*

*--AND 其他条件*

--LEFTJOIN VIEW\_SUPP\_GOODS\_UNIT GOODSSUPP ON GOODSSUPP.F\_GOODS\_NUM = --PSSM.F\_GOODS\_NUM

--WHERE GOODSSUPP.F\_SUPPLIER\_NUM=?

* 界面设计

**去掉供应商查询条件**



查询结果集合

|  |  |
| --- | --- |
| 日期 | YYYY-MM-DD |
| 仓库 | 仓库名称 |
| 库存类型 | 正常商品/备机 |
| 品牌 | 名称 |
| 品类 | 名称 |
| 商品SKU | 名称 |
| 期初数量 | 整数 |
| 期初金额 | 小数点后2位 |
| 期末数量 | 整数 |
| 期末金额 | 小数点后2位 |
| 出库数量 | 整数 |
| 出库金额 | 小数点后2位 |
| 入库数量 | 整数 |
| 入库金额 | 小数点后2位 |
| 拨入数量 | 整数 |
| 拨入金额 | 小数点后2位 |
| 拨出数量 | 整数 |
| 拨出金额 | 小数点后2位 |
| 调整数量 | 整数 |
| 调整金额 | 小数点后2位 |
| 盘耗数量 | 整数 |
| 盘耗金额 | 小数点后2位 |
| 盘升数量 | 整数 |
| 盘升金额 | 小数点后2位 |

##### 仓储进销存统计

* 处理流程
* 功能简述

统计当前进销存信息（根据日期/仓库/库存类型/供应商）。

* 数据库表结构

TC\_ORG\_PSSM。

* 详细设计

1. 查询界面初始化
2. 仓储：下拉选择，默认（全部）IOrder.getAllWarehouse
3. 库存类型:下拉选择(正常商品,备机) 默认正常商品：;
4. 品牌，品类、商品SKU，联动选择。
5. 供应商：IOrder.getUnityDistSupplier
6. 开始日期：当前日期;
7. 开始日期：当前日期
8. 统计方式：(日期/仓库）
9. 查询结果集

日期条件必须输入。开始日期必须小于结束日期

**[日期汇总参考SQL]**

SELECT

PSSM.F\_OCCUR\_DATE F\_OBJ,*--发生日期*

SUM(F\_FIRST\_COUNT),*--期初数量*

SUM(F\_FIRST\_COST), *--期初金额（无税）*

SUM(F\_PURCH\_COUNT + F\_PURCH\_COUNT\_PH) , *--进货数量*

SUM(F\_PURCH\_COST + F\_PURCH\_COST\_PH),*--进货金额（无税）*

SUM(F\_OUT\_DIST\_COUNT+F\_INSALE\_COUNT),*--出库数量*

SUM(F\_OUT\_DIST\_COST+F\_INSALE\_COST), *--出库金额*

SUM(F\_COUNTPROFIT\_COUNT),*--盘升数量*

SUM(F\_COUNTPROFIT\_COST),*--盘升金额*

SUM(F\_COUNTLOSS\_COUNT),*--盘耗数量*

SUM(F\_COUNTLOSS\_COST),*--盘耗金额*

SUM(F\_DIALIN\_COUNT),*--拨入数量*

SUM(F\_DIALIN\_COST),*--拨入金额*

SUM(F\_DIALOUT\_COUNT),*--拨出数量*

SUM(F\_DIALOUT\_COST),*--拨出数量*

SUM(F\_ADJUST\_COUNT),*--调整数量*

SUM(F\_ADJUST\_COST),*--调整金额*

SUM(F\_FIRST\_COUNT+F\_PURCH\_COUNT + F\_PURCH\_COUNT\_PH +F\_COUNTPROFIT\_COUNT+F\_DIALIN\_COUNT+F\_ADJUST\_COUNT

-F\_OUT\_DIST\_COUNT+F\_INSALE\_COUNT-F\_COUNTLOSS\_COUNT-F\_DIALOUT\_COUNT),

*--期末数量 = 期初+进货+铺货进货+盘升+拨入+调整-配送出-内部销售-盘耗-拨出*

SUM(F\_FIRST\_COST+F\_PURCH\_COST + F\_PURCH\_COST\_PH +F\_COUNTPROFIT\_COST+F\_DIALIN\_COST+F\_ADJUST\_COST

-F\_OUT\_DIST\_COST+F\_INSALE\_COST-F\_COUNTLOSS\_COST-F\_DIALOUT\_COST )

FROM TC\_ORG\_PSSM PSSM

LEFTJOIN VIEW\_GOODS\_BRAND\_MODEL GOODSDA ON PSSM.F\_GOODS\_NUM = GOODSDA.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_WAREHOUSE WARE ON WARE.F\_WARE\_NUM = PSSM.F\_ORG\_NUM

WHERE F\_REGION\_NUM='99'*--（固定）*

AND F\_ORG\_TYPE=1*--（固定）*

AND F\_GOODS\_TYPE=1*--（传入）*

GROUPBY PSSM.F\_OCCUR\_DATE

*--AND 其他条件*

--LEFTJOIN VIEW\_SUPP\_GOODS\_UNIT GOODSSUPP ON GOODSSUPP.F\_GOODS\_NUM = --PSSM.F\_GOODS\_NUM

--WHERE GOODSSUPP.F\_SUPPLIER\_NUM=?

GROUPBY PSSM.F\_OCCUR\_DATE

【按照仓库汇总参考SQL】

SELECT

FSPSSM.F\_WARE\_NAME F\_OBJ,

FSPSSM.F\_PURCH\_COUNT,

FSPSSM.F\_PURCH\_COST,

FSPSSM.F\_OUT\_COUNT,

FSPSSM.F\_OUT\_COST,

FSPSSM.F\_COUNTPROFIT\_COUNT,

FSPSSM.F\_COUNTPROFIT\_COST,

FSPSSM.F\_COUNTLOSS\_COUNT,

FSPSSM.F\_COUNTLOSS\_COST,

FSPSSM.F\_DIALIN\_COUNT,

FSPSSM.F\_DIALIN\_COST,

FSPSSM.F\_DIALOUT\_COUNT,

FSPSSM.F\_DIALOUT\_COST,

FSPSSM.F\_COUNTLOSS\_COST,

FSPSSM.F\_ADJUST\_COUNT,

FSPSSM.F\_ADJUST\_COST,

COALESCE(F\_FIRST\_COUNT,0) F\_FIRST\_COUNT,

COALESCE(F\_FIRST\_COST,0) F\_FIRST\_COST,

COALESCE(F\_FIRST\_COUNT,0) +COALESCE(F\_FS\_COUNT,0) F\_END\_COUNT,

COALESCE(F\_FIRST\_COST,0) +COALESCE(F\_FS\_COST,0) F\_END\_COST

FROM

(SELECT

WARE.F\_WARE\_NAME,

PSSM.F\_ORG\_NUM,

SUM(F\_PURCH\_COUNT + F\_PURCH\_COUNT\_PH) F\_PURCH\_COUNT , *--进货数量*

SUM(F\_PURCH\_COST + F\_PURCH\_COST\_PH) F\_PURCH\_COST,*--进货金额（无税）*

SUM(F\_OUT\_DIST\_COUNT+F\_INSALE\_COUNT)F\_OUT\_COUNT ,*--出库数量*

SUM(F\_OUT\_DIST\_COST+F\_INSALE\_COST) F\_OUT\_COST, *--出库金额*

SUM(F\_COUNTPROFIT\_COUNT) F\_COUNTPROFIT\_COUNT,*--盘升数量SUM(*

SUM(F\_COUNTPROFIT\_COST) F\_COUNTPROFIT\_COST,*--盘升金额*

SUM(F\_COUNTLOSS\_COUNT) F\_COUNTLOSS\_COUNT,*--盘耗数量*

SUM(F\_COUNTLOSS\_COST) F\_COUNTLOSS\_COST,*--盘耗金额*

SUM(F\_DIALIN\_COUNT) F\_DIALIN\_COUNT,*--拨入数量*

SUM(F\_DIALIN\_COST) F\_DIALIN\_COST,*--拨入金额*

SUM(F\_DIALOUT\_COUNT) F\_DIALOUT\_COUNT,*--拨出数量*

SUM(F\_DIALOUT\_COST) F\_DIALOUT\_COST,*--拨出数量*

SUM(F\_ADJUST\_COUNT) F\_ADJUST\_COUNT,*--调整数量*

SUM(F\_ADJUST\_COST) F\_ADJUST\_COST,*--调整金额*

SUM(F\_PURCH\_COUNT + F\_PURCH\_COUNT\_PH +F\_COUNTPROFIT\_COUNT+F\_DIALIN\_COUNT+F\_ADJUST\_COUNT

-F\_OUT\_DIST\_COUNT+F\_INSALE\_COUNT-F\_COUNTLOSS\_COUNT-F\_DIALOUT\_COUNT) F\_FS\_COUNT , *--发生数量*

SUM(F\_PURCH\_COST + F\_PURCH\_COST\_PH +F\_COUNTPROFIT\_COST+F\_DIALIN\_COST+F\_ADJUST\_COST

-F\_OUT\_DIST\_COST+F\_INSALE\_COST-F\_COUNTLOSS\_COST-F\_DIALOUT\_COST) F\_FS\_COST *--发生金额*

FROM TC\_ORG\_PSSM PSSM

LEFTJOIN VIEW\_GOODS\_BRAND\_MODEL GOODSDA ON PSSM.F\_GOODS\_NUM = GOODSDA.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_WAREHOUSE WARE ON WARE.F\_WARE\_NUM = PSSM.F\_ORG\_NUM

--LEFTJOIN VIEW\_SUPP\_GOODS\_UNIT GOODSSUPP ON GOODSSUPP.F\_GOODS\_NUM = --PSSM.F\_GOODS\_NUM

WHERE F\_REGION\_NUM='99'*--（固定）*

AND F\_ORG\_TYPE=1*--（固定）*

AND F\_GOODS\_TYPE=1*--（传入）*

AND PSSM.F\_OCCUR\_DATE BETWEEN'开始日期'AND'结束日期'

GROUPBY PSSM.F\_ORG\_NUM,F\_WARE\_NAME

) FSPSSM

LEFTJOIN

(SELECT

PSSM.F\_ORG\_NUM,

WARE.F\_WARE\_NAME,

SUM(F\_FIRST\_COUNT) F\_FIRST\_COUNT,*--期初数量*

SUM(F\_FIRST\_COST) F\_FIRST\_COST*--期初金额（无税）*

FROM TC\_ORG\_PSSM PSSM

LEFTJOIN VIEW\_GOODS\_BRAND\_MODEL GOODSDA ON PSSM.F\_GOODS\_NUM = GOODSDA.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_WAREHOUSE WARE ON WARE.F\_WARE\_NUM = PSSM.F\_ORG\_NUM

--LEFTJOIN VIEW\_SUPP\_GOODS\_UNIT GOODSSUPP ON GOODSSUPP.F\_GOODS\_NUM = --PSSM.F\_GOODS\_NUM

WHERE F\_REGION\_NUM='99'*--（固定）*

AND F\_ORG\_TYPE=1*--（固定）*

AND F\_GOODS\_TYPE=1*--（传入）*

AND PSSM.F\_OCCUR\_DATE='开始日期'

AND GOODSSUPP.F\_SUPPLIER\_NUM='供应商编码'

GROUPBY PSSM.F\_ORG\_NUM,F\_WARE\_NAME

)FIRSTPSSMON FSPSSM.F\_ORG\_NUM = FIRSTPSSM.F\_ORG\_NUM AND FSPSSM.F\_WARE\_NAME=FIRSTPSSM.F\_WARE\_NAME

【按照供应商汇总】

1. 导出查询结果集

将查询结果保存到EXCEL.

* 界面设计

**去掉供应商查询条件**



**查询主体**



##### 仓储商品进销存

* 处理流程
* 功能简述

查询当前仓库实际库存情况

* 数据库表结构

TC\_ORG\_PSSM

* 详细设计

1. 查询界面初始化
2. 仓储：下拉选择，默认（全部）IOrder.getAllWarehouse
3. 库存类型:下拉选择(正常商品,备机) 默认正常商品
4. 品牌，品类、商品SKU，联动选择。
5. 供应商：IOrder.getUnityDistSupplier
6. 开始日期：当前日期;
7. 开始日期：当前日期
8. 查询

开始日期和结束日期必须输入，且开始日期不能大于结束日期。

**统计方式=品牌参考SQL]**

SELECT

FSPSSM.F\_BRAND\_NAME F\_BRAND\_NAME,

'' F\_MODEL\_NAME,

'' F\_GOODS\_NAME,

FSPSSM.F\_PURCH\_COUNT,

FSPSSM.F\_PURCH\_COST,

FSPSSM.F\_OUT\_COUNT,

FSPSSM.F\_OUT\_COST,

FSPSSM.F\_COUNTPROFIT\_COUNT,

FSPSSM.F\_COUNTPROFIT\_COST,

FSPSSM.F\_COUNTLOSS\_COUNT,

FSPSSM.F\_COUNTLOSS\_COST,

FSPSSM.F\_DIALIN\_COUNT,

FSPSSM.F\_DIALIN\_COST,

FSPSSM.F\_DIALOUT\_COUNT,

FSPSSM.F\_DIALOUT\_COST,

FSPSSM.F\_COUNTLOSS\_COST,

FSPSSM.F\_ADJUST\_COUNT,

FSPSSM.F\_ADJUST\_COST,

COALESCE(F\_FIRST\_COUNT,0) F\_FIRST\_COUNT,

COALESCE(F\_FIRST\_COST,0) F\_FIRST\_COST,

COALESCE(F\_FIRST\_COUNT,0) +COALESCE(F\_FS\_COUNT,0) F\_END\_COUNT,

COALESCE(F\_FIRST\_COST,0) +COALESCE(F\_FS\_COST,0) F\_END\_COST

FROM

(SELECT

GOODSDA.F\_BRAND\_NAME,

SUM(F\_PURCH\_COUNT + F\_PURCH\_COUNT\_PH) F\_PURCH\_COUNT , *--进货数量*

SUM(F\_PURCH\_COST + F\_PURCH\_COST\_PH) F\_PURCH\_COST,*--进货金额（无税）*

SUM(F\_OUT\_DIST\_COUNT+F\_INSALE\_COUNT)F\_OUT\_COUNT ,*--出库数量*

SUM(F\_OUT\_DIST\_COST+F\_INSALE\_COST) F\_OUT\_COST, *--出库金额*

SUM(F\_COUNTPROFIT\_COUNT) F\_COUNTPROFIT\_COUNT,*--盘升数量SUM(*

SUM(F\_COUNTPROFIT\_COST) F\_COUNTPROFIT\_COST,*--盘升金额*

SUM(F\_COUNTLOSS\_COUNT) F\_COUNTLOSS\_COUNT,*--盘耗数量*

SUM(F\_COUNTLOSS\_COST) F\_COUNTLOSS\_COST,*--盘耗金额*

SUM(F\_DIALIN\_COUNT) F\_DIALIN\_COUNT,*--拨入数量*

SUM(F\_DIALIN\_COST) F\_DIALIN\_COST,*--拨入金额*

SUM(F\_DIALOUT\_COUNT) F\_DIALOUT\_COUNT,*--拨出数量*

SUM(F\_DIALOUT\_COST) F\_DIALOUT\_COST,*--拨出数量*

SUM(F\_ADJUST\_COUNT) F\_ADJUST\_COUNT,*--调整数量*

SUM(F\_ADJUST\_COST) F\_ADJUST\_COST,*--调整金额*

SUM(F\_PURCH\_COUNT + F\_PURCH\_COUNT\_PH +F\_COUNTPROFIT\_COUNT+F\_DIALIN\_COUNT+F\_ADJUST\_COUNT

-F\_OUT\_DIST\_COUNT+F\_INSALE\_COUNT-F\_COUNTLOSS\_COUNT-F\_DIALOUT\_COUNT) F\_FS\_COUNT , *--发生数量*

SUM(F\_PURCH\_COST + F\_PURCH\_COST\_PH +F\_COUNTPROFIT\_COST+F\_DIALIN\_COST+F\_ADJUST\_COST

-F\_OUT\_DIST\_COST+F\_INSALE\_COST-F\_COUNTLOSS\_COST-F\_DIALOUT\_COST) F\_FS\_COST *--发生金额*

FROM TC\_ORG\_PSSM PSSM

LEFTJOIN VIEW\_GOODS\_BRAND\_MODEL GOODSDA ON PSSM.F\_GOODS\_NUM = GOODSDA.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_WAREHOUSE WARE ON WARE.F\_WARE\_NUM = PSSM.F\_ORG\_NUM

LEFTJOIN VIEW\_SUPP\_GOODS\_UNIT GOODSSUPP ON GOODSSUPP.F\_GOODS\_NUM = PSSM.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_SUPPLIER SUPP ON SUPP.F\_SUPPLIER\_NUM = PSSM.F\_SUPPLIER\_NUM

WHERE F\_REGION\_NUM='99'*--（固定）*

AND F\_ORG\_TYPE=1*--（固定）*

AND F\_GOODS\_TYPE=1*--（传入）*

AND PSSM.F\_OCCUR\_DATE BETWEEN'开始日期'AND'结束日期'

GROUPBY GOODSDA.F\_BRAND\_NAME

) FSPSSM

LEFTJOIN

(SELECT

GOODSDA.F\_BRAND\_NAME,

SUM(F\_FIRST\_COUNT) F\_FIRST\_COUNT,*--期初数量*

SUM(F\_FIRST\_COST) F\_FIRST\_COST*--期初金额（无税）*

FROM TC\_ORG\_PSSM PSSM

LEFTJOIN VIEW\_GOODS\_BRAND\_MODEL GOODSDA ON PSSM.F\_GOODS\_NUM = GOODSDA.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_WAREHOUSE WARE ON WARE.F\_WARE\_NUM = PSSM.F\_ORG\_NUM

LEFTJOIN VIEW\_SUPP\_GOODS\_UNIT GOODSSUPP ON GOODSSUPP.F\_GOODS\_NUM = PSSM.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_SUPPLIER SUPP ON SUPP.F\_SUPPLIER\_NUM = PSSM.F\_SUPPLIER\_NUM

WHERE F\_REGION\_NUM='99'*--（固定）*

AND F\_ORG\_TYPE=1*--（固定）*

AND F\_GOODS\_TYPE=1*--（传入）*

AND PSSM.F\_OCCUR\_DATE='开始日期'

AND GOODSSUPP.F\_SUPPLIER\_NUM='供应商编码'

GROUPBY GOODSDA.F\_BRAND\_NAME

)FIRSTPSSMON FSPSSM.F\_BRAND\_NAME = FIRSTPSSM.F\_BRAND\_NAME

**统计方式=品类参考SQL]**

SELECT

FSPSSM.F\_BRAND\_NAME F\_BRAND\_NAME,

FSPSSM.F\_GOODS\_MODEL F\_MODEL\_NAME,

'' F\_GOODS\_NAME,

FSPSSM.F\_PURCH\_COUNT,

FSPSSM.F\_PURCH\_COST,

FSPSSM.F\_OUT\_COUNT,

FSPSSM.F\_OUT\_COST,

FSPSSM.F\_COUNTPROFIT\_COUNT,

FSPSSM.F\_COUNTPROFIT\_COST,

FSPSSM.F\_COUNTLOSS\_COUNT,

FSPSSM.F\_COUNTLOSS\_COST,

FSPSSM.F\_DIALIN\_COUNT,

FSPSSM.F\_DIALIN\_COST,

FSPSSM.F\_DIALOUT\_COUNT,

FSPSSM.F\_DIALOUT\_COST,

FSPSSM.F\_COUNTLOSS\_COST,

FSPSSM.F\_ADJUST\_COUNT,

FSPSSM.F\_ADJUST\_COST,

COALESCE(F\_FIRST\_COUNT,0) F\_FIRST\_COUNT,

COALESCE(F\_FIRST\_COST,0) F\_FIRST\_COST,

COALESCE(F\_FIRST\_COUNT,0) +COALESCE(F\_FS\_COUNT,0) F\_END\_COUNT,

COALESCE(F\_FIRST\_COST,0) +COALESCE(F\_FS\_COST,0) F\_END\_COST

FROM

(SELECT

GOODSDA.F\_BRAND\_NAME,

GOODSDA.F\_GOODS\_MODEL,

SUM(F\_PURCH\_COUNT + F\_PURCH\_COUNT\_PH) F\_PURCH\_COUNT , *--进货数量*

SUM(F\_PURCH\_COST + F\_PURCH\_COST\_PH) F\_PURCH\_COST,*--进货金额（无税）*

SUM(F\_OUT\_DIST\_COUNT+F\_INSALE\_COUNT)F\_OUT\_COUNT ,*--出库数量*

SUM(F\_OUT\_DIST\_COST+F\_INSALE\_COST) F\_OUT\_COST, *--出库金额*

SUM(F\_COUNTPROFIT\_COUNT) F\_COUNTPROFIT\_COUNT,*--盘升数量SUM(*

SUM(F\_COUNTPROFIT\_COST) F\_COUNTPROFIT\_COST,*--盘升金额*

SUM(F\_COUNTLOSS\_COUNT) F\_COUNTLOSS\_COUNT,*--盘耗数量*

SUM(F\_COUNTLOSS\_COST) F\_COUNTLOSS\_COST,*--盘耗金额*

SUM(F\_DIALIN\_COUNT) F\_DIALIN\_COUNT,*--拨入数量*

SUM(F\_DIALIN\_COST) F\_DIALIN\_COST,*--拨入金额*

SUM(F\_DIALOUT\_COUNT) F\_DIALOUT\_COUNT,*--拨出数量*

SUM(F\_DIALOUT\_COST) F\_DIALOUT\_COST,*--拨出数量*

SUM(F\_ADJUST\_COUNT) F\_ADJUST\_COUNT,*--调整数量*

SUM(F\_ADJUST\_COST) F\_ADJUST\_COST,*--调整金额*

SUM(F\_PURCH\_COUNT + F\_PURCH\_COUNT\_PH +F\_COUNTPROFIT\_COUNT+F\_DIALIN\_COUNT+F\_ADJUST\_COUNT

-F\_OUT\_DIST\_COUNT+F\_INSALE\_COUNT-F\_COUNTLOSS\_COUNT-F\_DIALOUT\_COUNT) F\_FS\_COUNT , *--发生数量*

SUM(F\_PURCH\_COST + F\_PURCH\_COST\_PH +F\_COUNTPROFIT\_COST+F\_DIALIN\_COST+F\_ADJUST\_COST

-F\_OUT\_DIST\_COST+F\_INSALE\_COST-F\_COUNTLOSS\_COST-F\_DIALOUT\_COST) F\_FS\_COST *--发生金额*

FROM TC\_ORG\_PSSM PSSM

LEFTJOIN VIEW\_GOODS\_BRAND\_MODEL GOODSDA ON PSSM.F\_GOODS\_NUM = GOODSDA.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_WAREHOUSE WARE ON WARE.F\_WARE\_NUM = PSSM.F\_ORG\_NUM

LEFTJOIN VIEW\_SUPP\_GOODS\_UNIT GOODSSUPP ON GOODSSUPP.F\_GOODS\_NUM = PSSM.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_SUPPLIER SUPP ON SUPP.F\_SUPPLIER\_NUM = PSSM.F\_SUPPLIER\_NUM

WHERE F\_REGION\_NUM='99'*--（固定）*

AND F\_ORG\_TYPE=1*--（固定）*

AND F\_GOODS\_TYPE=1*--（传入）*

AND PSSM.F\_OCCUR\_DATE BETWEEN'开始日期'AND'结束日期'

GROUPBY GOODSDA.F\_BRAND\_NAME,GOODSDA.F\_GOODS\_MODEL

) FSPSSM

LEFTJOIN

(SELECT

GOODSDA.F\_BRAND\_NAME,

GOODSDA.F\_GOODS\_MODEL,

SUM(F\_FIRST\_COUNT) F\_FIRST\_COUNT,*--期初数量*

SUM(F\_FIRST\_COST) F\_FIRST\_COST*--期初金额（无税）*

FROM TC\_ORG\_PSSM PSSM

LEFTJOIN VIEW\_GOODS\_BRAND\_MODEL GOODSDA ON PSSM.F\_GOODS\_NUM = GOODSDA.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_WAREHOUSE WARE ON WARE.F\_WARE\_NUM = PSSM.F\_ORG\_NUM

LEFTJOIN VIEW\_SUPP\_GOODS\_UNIT GOODSSUPP ON GOODSSUPP.F\_GOODS\_NUM = PSSM.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_SUPPLIER SUPP ON SUPP.F\_SUPPLIER\_NUM = PSSM.F\_SUPPLIER\_NUM

WHERE F\_REGION\_NUM='99'*--（固定）*

AND F\_ORG\_TYPE=1*--（固定）*

AND F\_GOODS\_TYPE=1*--（传入）*

AND PSSM.F\_OCCUR\_DATE='开始日期'

AND GOODSSUPP.F\_SUPPLIER\_NUM='供应商编码'

GROUPBY GOODSDA.F\_BRAND\_NAME,GOODSDA.F\_GOODS\_MODEL

)FIRSTPSSMON FSPSSM.F\_BRAND\_NAME = FIRSTPSSM.F\_BRAND\_NAME

AND FSPSSM.F\_MODEL\_NAME= FIRSTPSSM.F\_MODEL\_NAME

统计方式=商品SKU 参考SQL:

SELECT

FSPSSM.F\_BRAND\_NAME F\_BRAND\_NAME,

FSPSSM.F\_GOODS\_MODEL F\_MODEL\_NAME,

FSPSSM.F\_SKU\_ATTR\_NAME F\_GOODS\_NAME,

FSPSSM.F\_PURCH\_COUNT,

FSPSSM.F\_PURCH\_COST,

FSPSSM.F\_OUT\_COUNT,

FSPSSM.F\_OUT\_COST,

FSPSSM.F\_COUNTPROFIT\_COUNT,

FSPSSM.F\_COUNTPROFIT\_COST,

FSPSSM.F\_COUNTLOSS\_COUNT,

FSPSSM.F\_COUNTLOSS\_COST,

FSPSSM.F\_DIALIN\_COUNT,

FSPSSM.F\_DIALIN\_COST,

FSPSSM.F\_DIALOUT\_COUNT,

FSPSSM.F\_DIALOUT\_COST,

FSPSSM.F\_COUNTLOSS\_COST,

FSPSSM.F\_ADJUST\_COUNT,

FSPSSM.F\_ADJUST\_COST,

COALESCE(F\_FIRST\_COUNT,0) F\_FIRST\_COUNT,

COALESCE(F\_FIRST\_COST,0) F\_FIRST\_COST,

COALESCE(F\_FIRST\_COUNT,0) +COALESCE(F\_FS\_COUNT,0) F\_END\_COUNT,

COALESCE(F\_FIRST\_COST,0) +COALESCE(F\_FS\_COST,0) F\_END\_COST

FROM

(SELECT

GOODSDA.F\_BRAND\_NAME,

GOODSDA.F\_GOODS\_MODEL,

GOODSDA.F\_SKU\_ATTR\_NAME,

SUM(F\_PURCH\_COUNT + F\_PURCH\_COUNT\_PH) F\_PURCH\_COUNT , *--进货数量*

SUM(F\_PURCH\_COST + F\_PURCH\_COST\_PH) F\_PURCH\_COST,*--进货金额（无税）*

SUM(F\_OUT\_DIST\_COUNT+F\_INSALE\_COUNT)F\_OUT\_COUNT ,*--出库数量*

SUM(F\_OUT\_DIST\_COST+F\_INSALE\_COST) F\_OUT\_COST, *--出库金额*

SUM(F\_COUNTPROFIT\_COUNT) F\_COUNTPROFIT\_COUNT,*--盘升数量SUM(*

SUM(F\_COUNTPROFIT\_COST) F\_COUNTPROFIT\_COST,*--盘升金额*

SUM(F\_COUNTLOSS\_COUNT) F\_COUNTLOSS\_COUNT,*--盘耗数量*

SUM(F\_COUNTLOSS\_COST) F\_COUNTLOSS\_COST,*--盘耗金额*

SUM(F\_DIALIN\_COUNT) F\_DIALIN\_COUNT,*--拨入数量*

SUM(F\_DIALIN\_COST) F\_DIALIN\_COST,*--拨入金额*

SUM(F\_DIALOUT\_COUNT) F\_DIALOUT\_COUNT,*--拨出数量*

SUM(F\_DIALOUT\_COST) F\_DIALOUT\_COST,*--拨出数量*

SUM(F\_ADJUST\_COUNT) F\_ADJUST\_COUNT,*--调整数量*

SUM(F\_ADJUST\_COST) F\_ADJUST\_COST,*--调整金额*

SUM(F\_PURCH\_COUNT + F\_PURCH\_COUNT\_PH +F\_COUNTPROFIT\_COUNT+F\_DIALIN\_COUNT+F\_ADJUST\_COUNT

-F\_OUT\_DIST\_COUNT+F\_INSALE\_COUNT-F\_COUNTLOSS\_COUNT-F\_DIALOUT\_COUNT) F\_FS\_COUNT , *--发生数量*

SUM(F\_PURCH\_COST + F\_PURCH\_COST\_PH +F\_COUNTPROFIT\_COST+F\_DIALIN\_COST+F\_ADJUST\_COST

-F\_OUT\_DIST\_COST+F\_INSALE\_COST-F\_COUNTLOSS\_COST-F\_DIALOUT\_COST) F\_FS\_COST *--发生金额*

FROM TC\_ORG\_PSSM PSSM

LEFTJOIN VIEW\_GOODS\_BRAND\_MODEL GOODSDA ON PSSM.F\_GOODS\_NUM = GOODSDA.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_WAREHOUSE WARE ON WARE.F\_WARE\_NUM = PSSM.F\_ORG\_NUM

LEFTJOIN VIEW\_SUPP\_GOODS\_UNIT GOODSSUPP ON GOODSSUPP.F\_GOODS\_NUM = PSSM.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_SUPPLIER SUPP ON SUPP.F\_SUPPLIER\_NUM = PSSM.F\_SUPPLIER\_NUM

WHERE F\_REGION\_NUM='99'*--（固定）*

AND F\_ORG\_TYPE=1*--（固定）*

AND F\_GOODS\_TYPE=1*--（传入）*

AND PSSM.F\_OCCUR\_DATE BETWEEN'开始日期'AND'结束日期'

GROUPBY GOODSDA.F\_BRAND\_NAME,GOODSDA.F\_GOODS\_MODEL,GOODSDA.F\_SKU\_ATTR\_NAME

) FSPSSM

LEFTJOIN

(SELECT

GOODSDA.F\_BRAND\_NAME,

GOODSDA.F\_GOODS\_MODEL,

GOODSDA.F\_SKU\_ATTR\_NAME,

SUM(F\_FIRST\_COUNT) F\_FIRST\_COUNT,*--期初数量*

SUM(F\_FIRST\_COST) F\_FIRST\_COST*--期初金额（无税）*

FROM TC\_ORG\_PSSM PSSM

LEFTJOIN VIEW\_GOODS\_BRAND\_MODEL GOODSDA ON PSSM.F\_GOODS\_NUM = GOODSDA.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_WAREHOUSE WARE ON WARE.F\_WARE\_NUM = PSSM.F\_ORG\_NUM

LEFTJOIN VIEW\_SUPP\_GOODS\_UNIT GOODSSUPP ON GOODSSUPP.F\_GOODS\_NUM = PSSM.F\_GOODS\_NUM

LEFTJOIN TB\_DA\_SUPPLIER SUPP ON SUPP.F\_SUPPLIER\_NUM = PSSM.F\_SUPPLIER\_NUM

WHERE F\_REGION\_NUM='99'*--（固定）*

AND F\_ORG\_TYPE=1*--（固定）*

AND F\_GOODS\_TYPE=1*--（传入）*

AND PSSM.F\_OCCUR\_DATE='开始日期'

AND GOODSSUPP.F\_SUPPLIER\_NUM='供应商编码'

GROUPBY GOODSDA.F\_BRAND\_NAME,GOODSDA.F\_GOODS\_MODEL,GOODSDA.F\_SKU\_ATTR\_NAME

)FIRSTPSSMON FSPSSM.F\_BRAND\_NAME = FIRSTPSSM.F\_BRAND\_NAME AND

FSPSSM.F\_GOODS\_MODEL = FIRSTPSSM.F\_GOODS\_MODEL AND

FSPSSM.F\_SKU\_ATTR\_NAME = FIRSTPSSM.F\_SKU\_ATTR\_NAME

* 界面设计

去掉供应商查询条件



显示结果集合

|  |  |
| --- | --- |
| 品牌 |  |
| 品类 |  |
| 商品SKU |  |
| 期初数量 | 整数 |
| 期初金额 | 小数点后2位 |
| 期末数量 | 整数 |
| 期末金额 | 小数点后2位 |
| 出库数量 | 整数 |
| 出库金额 | 小数点后2位 |
| 入库数量 | 整数 |
| 入库金额 | 小数点后2位 |
| 拨入数量 | 整数 |
| 拨入金额 | 小数点后2位 |
| 拨出数量 | 整数 |
| 拨出金额 | 小数点后2位 |
| 调整数量 | 整数 |
| 调整金额 | 小数点后2位 |
| 盘耗数量 | 整数 |
| 盘耗金额 | 小数点后2位 |
| 盘升数量 | 整数 |
| 盘升金额 | 小数点后2位 |

#### 仓储盘点管理

##### 仓储盘点单维护

##### 仓储盘点单删除

##### 仓储盘点单审核

##### 仓储盘点单跟踪

##### 仓储盘点统计

#### 仓储调整管理

##### 仓储调整单维护

##### 仓储调整单删除

##### 仓储调整单审核

##### 仓储调整单跟踪

##### 仓储调整查询

#### 仓储移库管理

##### 仓储移库单查询

### 地市库存分配

地市库存分配总体功能图：



#### 地市库存分配

##### 地市分配库存阀值设置

* 处理流程

无

* 功能简述

1. 地市分配库存阀值设置功能只提供给省统一运营人员使用；
2. 左侧属性列表按照品牌-》型号-》商品进行属性展示，点击品牌或者型号可以进行叶子节点收缩，点击商品右侧展示地市分配库存阀值设置信息，如图：6.1.5.1.1-1；
3. 列表展示全省所有地市的信息，如后台无该地市商品的阀值设置信息，系统数据初始化：商品库存上限：0，商品库存预警-》数值：0；是否在用：是，修改人：空；修改时间：空；如有有无该地市商品的阀值设置信息，则以该数据数据为准
4. 商品库存上限为0时，表示库存上限为不限制，商品库存预警为0时，表示不做库存告警；
5. 根据当前商品的是否在用状态显示不同的启用或停用按钮：当在用时显示停用按钮，当不在用时显示启用按钮；点击“停用”或“启用”按钮，调用对应的stopRegionGoodsLimit或startRegionGoodsLimit方法，根据商品编码修改后台字段F\_IS\_USE对应的状态；同时修改操作人信息；
6. 点击“保存”按钮，将当前页面数据保存至后台数据库，并页面提示保存成功；同时修改操作人信息；
7. 当地市库存小于安全告警值时，需要通过短信和系统消息告知责任人，目前暂时不做，后续等系统通知模块建立后统一开发；

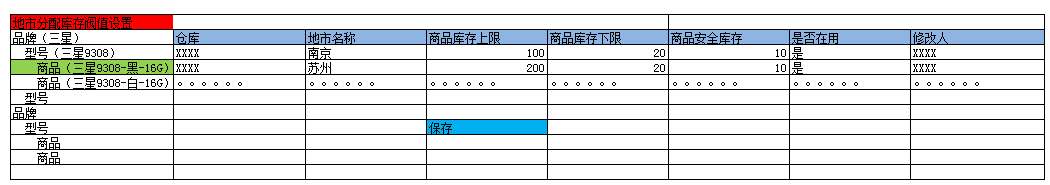
* 数据库表结构

见TO\_REGION\_GOODS\_LIMIT

* 详细设计

无

* 界面设计



(6.1.5.1.1-1)

##### 地市库存分配申请

* 处理流程

无

* 功能简述

1. 按照品牌名称、品类名称、商品名称、仓库、仓库可分配数量、最近一天入库（勾选）查询当前省仓商品的当前库存情况；如图：6.1.5.1.2-1
2. 选择其中一条展示记录，点击“地市库存分配”按钮，展示地市库存分配申请页面，如图：6.1.5.1.2-2；库存分配策略默认为手工分配，选择自动分配时，展示“自动分配策略”单选项：**A-库存待补货量加权平均B-库存消耗率加权平均C-按库存周转率加权平均**,输入本次分配数量，点击“计算”按钮，系统调用后台计算策略接口计算当前地市分配商品的本次分配量；否则提示需要输入本次分配量；不允许出现地市的本次分配量合计大于输入的分配量；
3. **库存待补货量加权平均**

策略条件：能取到当前各地市的库存，以及库存分配的预警阀值。

策略方案：按照该机型及分类当前的库存值进行补充，以与库存阀值的差值作为权重分配。



地市分配的算法描述说明：

先根据阀值计算需求最大量：总需求量 = SUM（地市阀值–当前库存），如果总需求量小于本次分配总量，则提醒分配有富余，可能会产生超阀值的分配，需要分配人员确认。

再计算分配系数：分配系数=1 /总需求量\* 本次分配总量

开始地市轮询，如果某地市分类或机型没有设定阀值，则可以以当前库存2倍，其中倍数设置为省系统参数。

如果当前地市库存阀值<= 当前库存，不分配，

否则该地市分配量=（地市阀值–当前库存）\* 分配系数。该地市分配量取整或者四舍五入后，如果大于当前地市库存阀值 - 当前库存的值，则根据分配人员确认的结果进行处理，如总量超确认分，则按照计算值分配，否则地市分配量 = 当前地市库存阀值 - 当前库存，即补充到阀值；如果不大于，则按计算值分配。

依次类推，最后一个地市的分配量=本次分配总量-其余地市已经分配量之和。

举例：假设本次分配总量（sendout）为：19

|  |  |  |  |
| --- | --- | --- | --- |
| 地市 | 当前库存 | 库存阀值 | 本次实际分配 |
| 1 | 3 | 7 | 2 |
| 2 | 0 | 10 | 5 |
| 3 | 5 | 9 | 2 |
| 4 | 8 | 10 | 1 |
| 5 | 10 | 15 | 2 |
| 6 | 13 | 20 | 3 |
| 7 | 7 | 10 | 1 |
| 8 | 4 | 8 | 3 |

第一步：计算总的需求量：

Sumqty = (7-3) + (10-0) + (9-5) + (10-8) + (15-10) + (20-13) + (10-7) + (8-4) = 41

第二步：计算分配系数：

Modulus=(1 / Sumqty) \* sendout = 1/41\*19 = 0.463

第三步：计算地市分配量QTY=（地市阀值 – 当前库存）\* 分配系数

地市1： QTY1=(7 -3) \* (19/41) = 1.85 == 2

地市2： QTY2=(10 -0) \* (19/41) = 4.60 == 5

地市3： QTY3=(9 -5) \* (19/41) = 1.85 == 2

地市4： QTY4=(10 -8) \* (19/41) = 0.92 == 1

地市5： QTY5=(15 -10) \* (19/41) = 2.32 == 2

地市6： QTY6=(20 -13) \* (19/41) = 3.24 == 3

地市7： QTY7=(10 -7) \* (19/41) = 1.39 == 1

地市8： QTY8=(8 -4) \* (19/41) = 1.85 == 2(此步不计算)。

第四步：计算最后一个地市分配量

地市8： QTY8=19-2-5-2-1-2-3-1 =3 前面计算数量为：2，其分配量为3，存误差。

策略优点：避免分配的库存过高，配置的库存阀值生效。

策略缺点：需要对库存阀值进行设定，假如某分类或者商品无阀值的话，本分配策略无法进行（除非系统默认一个阀值，这样可能比较复杂）；最后一个地市承担前面计算四舍五入的多余量，可能会分配多一个或者少一个。

1. **库存消耗率加权平均**

策略条件：能取到当前各地市的库存，以及（品牌或者分类）上周的销售量。

策略方案：根据库存被销售的时间（可以考虑以周为单位）加权平均。

地市分配的算法描述说明：

先计算各地市的库存消耗时间（单位：周），作为地市库存消耗系数： =上周销量/当前库存。

本次分配系数：本次分配量 /sum（地市库存消耗系数）;

地市分配量：分配系数\*地市库存消耗系数

依次类推，最后一个地市的分配量=本次分配总量-其余地市已经分配量之和。



举例：假设本次分配总量（sendout）为：19

|  |  |  |  |
| --- | --- | --- | --- |
| 地市 | 当前库存 | 上周销量 | 本次实际分配 |
| 1 | 3 | 6 | 3 |
| 2 | 0 | 9 | 7 |
| 3 | 5 | 5 | 2 |
| 4 | 8 | 6 | 1 |
| 5 | 10 | 7 | 1 |
| 6 | 13 | 8 | 1 |
| 7 | 7 | 6 | 1 |
| 8 | 4 | 5 | 3 |

第一步：计算地市库存消耗系数：

地市1：Modulus1 =6/3 =2

地市2：Modulus2 =9/0 == （当前库存为0，设定为销量的一半4.5）

地市3：Modulus3 =5/5 =1

地市4：Modulus4 =6/8 =0.75

地市5：Modulus5 =7/10 =0.7

地市6：Modulus6 =8/13 =0.616

地市7：Modulus7 =6/7 =0.857

地市9：Modulus8 =5/4 =1.25

第二步：计算本次分配系数：

Modulus=19 \* / Sum（Modulus1-8）

=19/(2+4.5+1+0.75+0.7+0.616+0.857+1.25) = 19/11.673 == 1.628

第三步：计算地市分配量QTY=地市库存消耗系数\* 本次分配系数

地市1： QTY1=1.628 \* 2 = 3.25 == 3

地市2： QTY2=1.628 \* 4.5 = 7.32 == 7

地市3： QTY2=1.628 \* 1 = 1.628 == 2

地市4： QTY3=1.628 \* 0.75 = 1.2 == 1

地市5： QTY4=1.628 \* 0.7 = 1.2 == 1

地市6： QTY5=1.628 \* 0.616 = 1.0 == 1

地市7： QTY6=1.628 \* 0.857 = 1.39 == 1

地市8： QTY7=1.628 \* 1.25 = 2.04 == 2(这步不用计算)

第四步：计算最后一个地市分配量

地市8： QTY7=19-3-7-2-1-1-1-1= 3 这个与计算值多1。

策略优点：考虑到了销售数据及库存消耗率，相对来说比较合理，分配的库存按照销售多库存少则分配多的原则进行。

策略缺点：销售依据只取上周的实际销售数量，无法全面涵盖该地市的真实情况（解决方案为尽量取多一段时间的销量来计算，比如半个月内，一个月内等）。库存为0的地市，具体消耗系数的计算方式（建议按一半计算）没有数据依据。

1. **按库存周转率加权平均**

策略条件：能取到各品牌或者分类上月期初库存，上月入库，上月销售量。

或者按照上月的数据作为计算值。

策略方案：根据库存周转率加权平均。

算法描述说明：

先计算商品或者分类的上月库存平均数：平均库存值=( 期初+期末)/2；

再计算库存周转率：售出数/ 平均库存值；

再根据库存周转率的权重值进行分配：分配数量=分配量 \* 库存周转率/sum(库存周转率)

依次类推，最后一个地市的分配量=本次分配总量-其余地市已经分配量之和。



举例：假设本次分配总量（sendout）为：19

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 地市 | 上月期初库存 | 上月入库 | 上月销量 | 上月末库存 | 本次实际分配 |
| 1 | 3 | 7 | 6 | 4 | 2 |
| 2 | 0 | 10 | 9 | 1 | 12 |
| 3 | 5 | 6 | 5 | 6 | 1 |
| 4 | 8 | 7 | 6 | 9 | 1 |
| 5 | 10 | 8 | 7 | 11 | 1 |
| 6 | 13 | 9 | 8 | 14 | 1 |
| 7 | 7 | 7 | 6 | 8 | 1 |
| 8 | 4 | 6 | 5 | 5 | 0 |

第一步：分地市计算地市上月末库存数：

见上表：上月期初库存+上月入库-上月销量。

第二步：分地市计算上月库存周转率

地市1：Modulus1 =6/(4/2) =3

地市2：Modulus2 =9/(1/2) =18

地市3：Modulus3 =5/(6/2) =1.67

地市4：Modulus4 =6/(9/2) =1.33

地市5：Modulus5 =7/(11/2) =1.25

地市6：Modulus6 =8/(14/2) =1.42

地市7：Modulus7 =6/(8/2) =1.5

地市8：Modulus8 =5/(5/2)=2

第三步：分地市计算本次分配量

Modulus = 3+18+1.67+1.33+1.25+1.42+1.5+2 = 30.17

地市1： QTY1 =19 \* 3/30.17 =1.89 == 2

地市2： QTY 2 =19 \* 18/30.17 =11.33 ==1 2

地市3： QTY 3 =19 \* 1.67/30.17 =1.05 == 1

地市4： QTY 4 =19 \* 1.33/30.17 =0.84 == 1

地市5： QTY 5 =19 \* 1.25/30.17 =0.78 == 1

地市6： QTY 6 =19 \* 1.42/30.17 =0.89 == 1

地市7： QTY 7 =19 \* 1.5/30.17 =0.94 == 1

地市8： QTY 8 =19 \* 2/30.17 =1.25 == 1(这步不用计算)

第四步：计算最后一个地市分配量

地市8： QTY7=19-2-12-1-1-1-1-1= 0 这个与计算值相差1，地市8本次分配数量为0。（地市2的极端情况造成的货不够分）

计算时，判断所需数据不存在，直接返回，提示用户不存在的数据；

1. 以下是上述3种分配策略涉及指标的取值方式，**目前涉及数据只能先做自动分配策略A，后续等依赖模块完成后继续策略B和C的开发，但前台页面先预留展示；**

|  |  |
| --- | --- |
|  |  |
| 指标名称 | 指标取值方式 |
| 当前地市库存 | 地市库存二级表TO\_STOCK\_REGION |
| 地市库存分配的预警阀值 | TO\_REGION\_GOODS\_LIMIT |
| 地市上周销量 | 数据目前依赖未设计的模块 |
| 地市上月期初库存 | 数据目前依赖未设计的模块 |
| 地市上月入库 | 地市库存分配单（单据主表）TO\_BILL\_STOCK\_DIST，状态为审核通过，且审核操作时间为在上个月 |
| 地市上月销量 | 数据目前依赖未设计的模块 |
| 地市上月末库存 | 数据目前依赖未设计的模块 |

1. 提货有效期默认当前月最后一天，可通过时间组件进行修改；根据当前选择的仓库需要管理查询仓库和地市对照表，默认一个地市只能对应一个仓库，按照当前的仓库查询当前地市信息，并进行列举；
2. 仓库库存和仓库可分配量根据仓库和商品进行查询而来；本次分配数量为必填且输入的值不能大于当前仓库可分配量；输入值大于时，界面就应该立即红色字体提示；分配策略只有在分配方式为自动时才显示；选择完数据后点击“计算”按钮，根据仓库和地市的对照关系列举出所以对应地市的数据；当自动分配策略时，需要调用后台策略计算方式计算当前库存分配值和查询其他对应的展示字段；
3. 修改本次分配量时动态计算分配量总计值，当总计值大于本次分配量时，红色显示，当修改本次分配量大于总计值时，正常颜色字体显示；输入的本次分配量均必须为数值，且不能小于0；地市本次分配量不能大于本月计划量减去本月已分配量；
4. 点击提交申请，插入地市库存分配申请单，提交后台保存，保存时需要校验当前分配量总计不能大于仓库的当前可分配量；保存后台数据为待审核状态，；
5. 当前提交申请的数据信息不能和现有待审核的数据有冲突，否则提示用户已有该商品的分配信息，如需调整请使用修改功能；判断冲突的依据是：仓库编码、商品编码、单据状态（须为待审核）；在点击“分配”时就需提示用户不允许操作；点击保存时也需在后台进行校验；
6. 后台保存数据时，对于分配量为0的数据不做处理；
7. 涉及查询页面的中指标取值信息如下：

|  |  |
| --- | --- |
| 指标 | 取值方式 |
| 当前地市库存 | 地市二级库存表：当前地市库存=累计要货数量-累计发货数量，根据地市、商品、仓库进行查询 |
| 当前地市可用要货量 | 地市二级库存表：可用要货数量=累计分配数量-累计要货数量，根据地市、商品、仓库进行查询 |
| 库存上限 | TO\_STOCK\_DIST\_LOG，中查询本月的分配和回收的总量 |
| 本月计划量 | 地市提货计划汇总表中根据地市编码、商品编码、月份查询地市计划量，没有数据显示无 |
| 本月已经分配量 | TO\_STOCK\_DIST\_LOG，中查询本月的分配和回收的总量 |

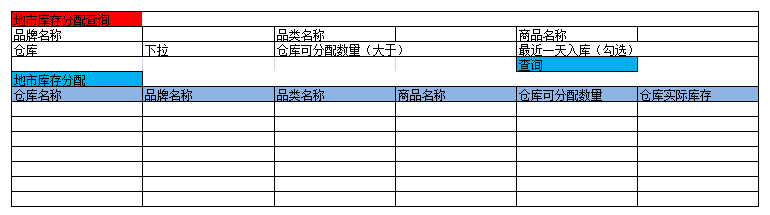
* 数据库表结构

见TO\_BILL\_STOCK\_DIST

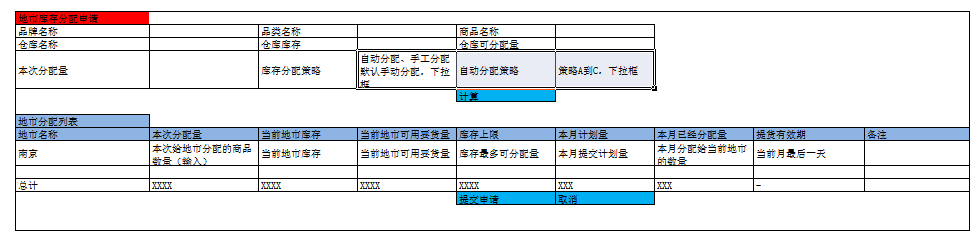
* 详细设计

无

* 界面设计



(6.1.5.1.2-1)



(6.1.5.1.2-2)

##### 地市库存分配修改

* 处理流程

无

* 功能简述

1. 当用户提交地市库存分配申请或审核不通过需要进行调整时，可对其进行数据调整；
2. 按照单据号、制单开始时间、制单结束时间、品牌名称、品类名称、商品名称和制单人查询单据状态为“待审核”和“审核不通过”的单据信息；如图：6.1.5.1.3-1
3. 选择其中一条数据，点击“删除”按钮，即可删除数据，提示操作成功，后台调用单据组件recordBillInfo记录单据信息，并修改单据状态为修改；
4. 目前不限制只能当前修改人只能修改自己申请的单据；
5. 选择其中一条数据，点击“修改”按钮，弹出修改页面，如图：6.1.5.1.3-2，其中当前地市库存、当前地市可用库存、库存上限、本月计划量、本月已分配量需要重新查询，展示字段和6.1.5.1.1一致；
6. 修改分配量或调整本次分配量，点击“提交申请”即可保存数据；调用单据组件记录单据信息，同时单据状态为待审核；
7. 如为自动分配策略或调整了本次分配量，需要根据自动策略计算本次分配量，需再次点击“计算”按钮；

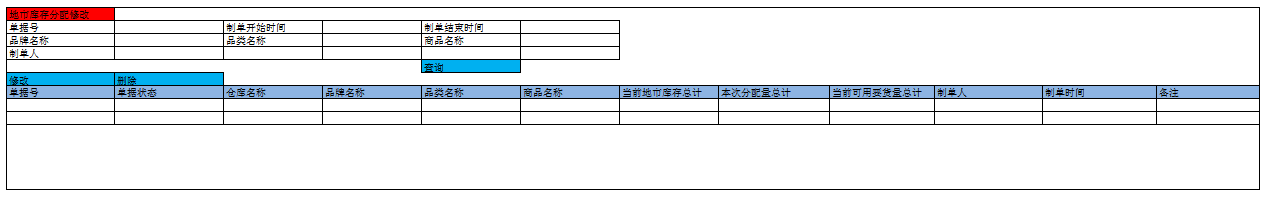
* 数据库表结构

见TO\_BILL\_STOCK\_DIST

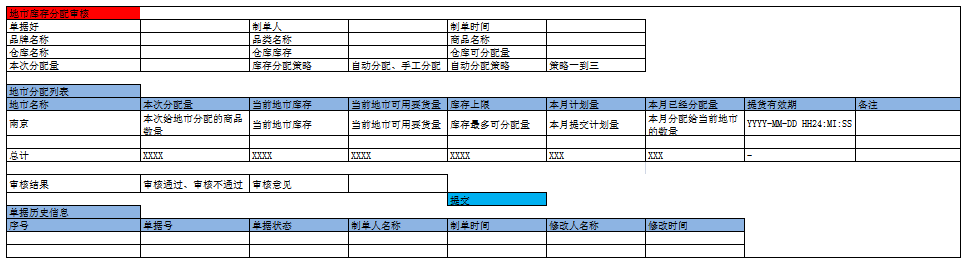
* 详细设计

无

* 界面设计



（6.1.5.1.3-1）



（6.1.5.1.3-2）

##### 地市库存分配审核

* 处理流程

无

* 功能简述

1. 地市库存分配审核功能只提供给省统一运营人员使用；
2. 按照单据号、制单开始时间、制单结束时间、品牌名称、品类名称和商品名称查询当前待审核的库存分配单信息；如图：6.1.5.1.4-1
3. 选择其中一条待审核数据，点击“审核”按钮，跳转到地市库存分配审核页面，如图：6.1.5.1.4-2
4. 选择审核结果：审核通过或审核不通过并输入审核意见，点击“提交”；调用对应单据组件接口recordBillInfo记录单据信息，并修改地市二级库存主表记录，增加累计分配数量，同时修改仓库实时库存表，减少可分配库存数量，修改单据表状态，同时记录地市二级库存分配日志（TO\_STOCK\_DIST\_LOG）；以上操作必须是在一个事物中进行操作；
5. 审核操作页面上展示的操作信息中，当前仓库库存与实际需要分配数量大于时，提示用户此单必须进行审核不通过处理；审核操作之前后台需要判断当前的仓库库存数量是否大于实际需要分配数量，如大于返回直接报错；提示不允许审核通过，只能进行审核不通过操作；

* 数据库表结构

见TO\_STOCK\_REGION、TO\_BILL\_STOCK\_DIST、TO\_STOCK\_REALTIME、TO\_STOCK\_DIST\_LOG

* 详细设计

审核通过

1、修改地市库存表

单据类型=’BJDSFP’ , 操作表TO\_STOCKBACKUP\_REGION

单据类型=’BILL\_DSKCFP’操作表TO\_STOCK\_REGION

增加累计分配数量，注意单据类型=’BILL\_DSKCFP’，需要将数据库当前年月加入到TO\_STOCK\_REGION的F\_STOCK\_MONTH中，渠道默认是=’STQD’；

1. 修改TO\_STOCK\_REALTIME实时库存表

单据类型=’BJDSFP’，减少表TO\_STOCK\_REALTIME中F\_STOCK\_TYPE\_NUM =‘BJK’的F\_DIST\_COUNT= F\_DIST\_COUNT-分配数量

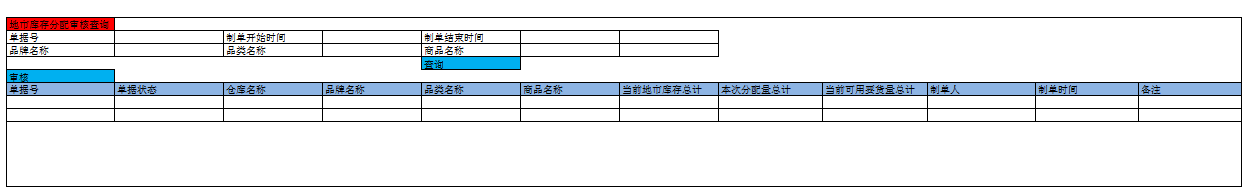
单据类型=‘BILL\_DSKCFP’减少表TO\_STOCK\_REALTIME中F\_STOCK\_TYPE\_NUM =‘ZPK’的F\_DIST\_COUNT= F\_DIST\_COUNT-分配数量

1. 修改单据状态=审核通过
2. 记录库存分配日志表TO\_STOCK\_DIST\_LOG

单据类型=’BJDSFP’ , F\_GOODS\_TYPE =2

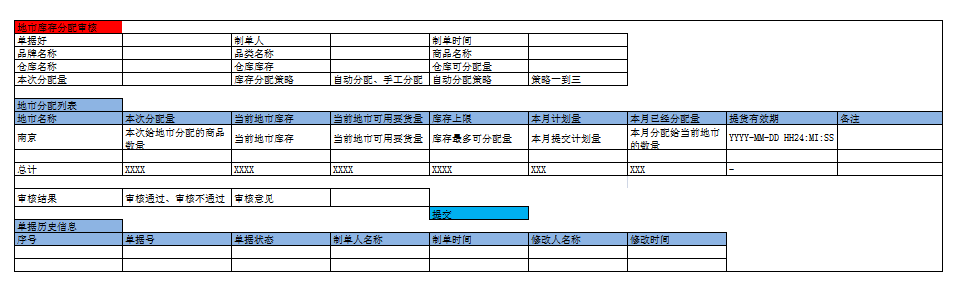
单据类型=’BILL\_DSKCFP’F\_GOODS\_TYPE=1

* 界面设计



(6.1.5.1.4-1)

界面增加单据类型(包括单据类型BJDSFP, BILL\_DSKCFP),查询条件增加BJDSFP, BILL\_DSKCFP



(6.1.5.1.4-2)

##### 地市库存分配回收

* 处理流程

无

* 功能简述

1. 根据品牌名称、品类名称、商品名称和地市（下拉框）查询地市库存分配信息；如图：6.1.5.1.5-1；点击“查询”按钮，可列表展示地市库存分配信息，初始化默认本次回收数量等于可回收库存数量；
2. 根据查询条件查询地市仓库分配信息，选择需要回收的数据，并录入本次分配回收的数量，点击“回收”按钮，即可选择的数据进行回收操作；提交时需要判断本次回收的数量不能为0，且必须小于可回收库存的数量；
3. 点击“地市一键回收”按钮，弹出页面展示如图：6.1.5.1.5-2，页面可通过全选按钮全选所有地市或只选址其中一个或多个地市；点击“回收”按钮即可进行回收操作；
4. 点击“商品一键回收”按钮，弹出页面展示如图：6.1.5.1.5-3，页面通过查询条件查询出当前需要回收的商品信息（一个商品可对应两个仓库），勾选需要回收的商品和仓库信息，点击“回收”按钮即可进行回收操作；
5. 回收操作：
6. 回收操作必须保证是在一个事物中操作；
7. 回收操作时必须判断当前回收量是否大于地市二级仓库的可要货量；如大于直接返回错误信息，不在进行回收操作；
8. 减少地市对应的累计分配数量值，如是回收操作按照页面传入的“本次回收数量”进行调整；
9. 如是一键回收，调整累计分配数量至累计要货数量，再将仓库实时库存表中的可分配数量值加上对应仓库下的所有地市回收值；
10. 地市一键回收：按照地市维度，将地市二级库存的中可要货数量大于0 的商品库存进行全力回收；
11. 商品一键回收：按照商品和仓库的维度，将对应仓库下所有地市该商品中可要货数量大于0的地市二级库存进行回收；
12. 回收操作后需记录地市二级库存分配记录表，分配类型为2-回收的数据，单据号不填；

* 数据库表结构

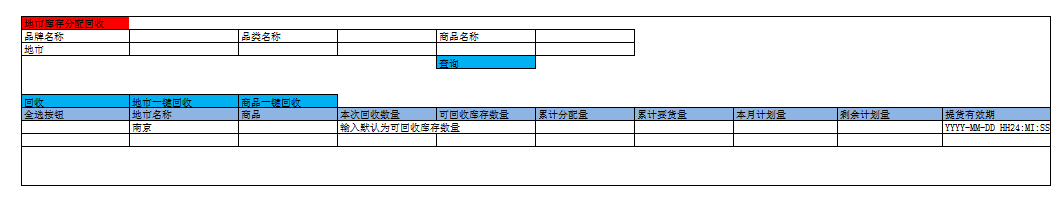
见TO\_STOCK\_REGION、TO\_STOCK\_REALTIME、TO\_STOCK\_DIST\_LOG

* 详细设计

只查询TO\_STOCK\_REALTIME中库存类型编码F\_STOCK\_TYPE\_NUM=‘ZPK’的数据

记录日志的时候TO\_STOCK\_DIST\_LOG表中 F\_GOODS\_TYPE=1(售机)

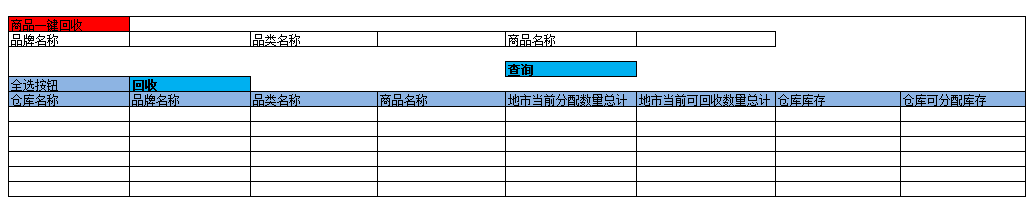
* 界面设计



(6.1.5.1.5-1)



(6.1.5.1.5-2)



(6.1.5.1.5-3)

##### 地市库存分配历史

* 处理流程

无

* 功能简述

1. 按照品牌名称、品类名称、商品名称、分配类型和地市名称查询地市库存分配历史；如图：6.1.5.1.6-1；点击查询即可展示对应地市库存分配历史；

* 数据库表结构

见TO\_STOCK\_DIST\_LOG

* 详细设计

无

* 界面设计

查询条件：增加商品类型（1-售机 2-备机），同时在显示的时候显示商品类型



（6.1.5.1.6-1）

##### 地市分配库存统计

* 处理流程

无

* 功能简述

1. 按照品牌名称、品类名称、商品名称、归属月份和地市名称统计地市库存分配；如图：6.1.5.1.7-1；点击查询即可展示对应条件的统计数据；
2. 查询示例：选择月份“2013-08”展示选择月份所有地市商品的统计信息，则按照月份、地市、品牌、品类和商品维度进行分组汇总；其中本月销售量目前缺少销售订单信息，暂无法获取数据；
3. 统计时间范围为当前月往前6个月；

* 数据库表结构

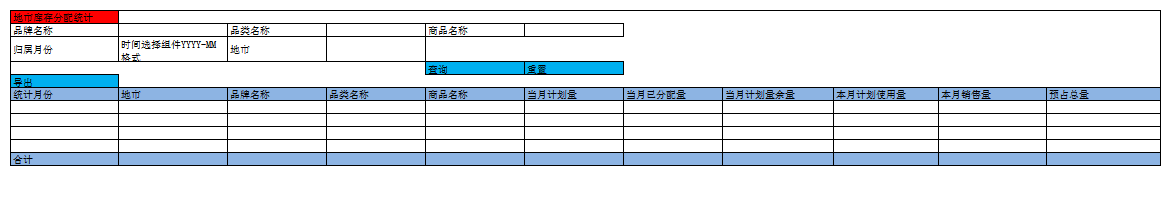
见TO\_STOCK\_REGION

* 详细设计

查询条件：增加商品类型（1-售机 2-备机），同时在显示的时候显示商品类型

本月销售,预占数量不要

* 界面设计



（6.1.5.1.7-1）

##### 地市库存查询

* 处理流程

无

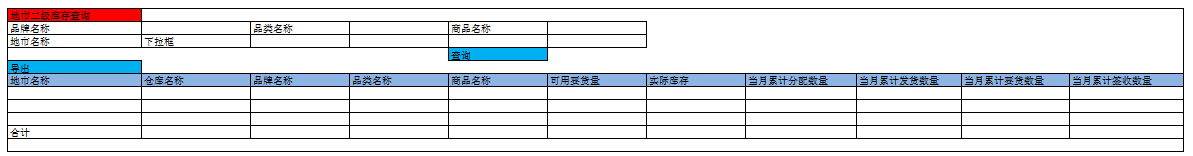
* 功能简述

1. 按照品牌名称、品类名称、商品名称、地市名称查询地市库存信息；如图：6.1.5.1.8-1；点击查询即可展示对应条件的统计数据；
2. 点击导出，可按照当前条件导出查询出的所有数据；
3. 地市员工登陆页面时通过地市名称下拉框限制只能查询当前地市的数据，不允许调整；统一运营员工登陆页面时，可查询所有地市的分配信息；功能页面地市管理员和统一运营管理员都可以查询；

* 数据库表结构

见TO\_STOCK\_REGION

* 详细设计
* 界面设计



（6.1.5.1.8-1）

##### 地市库存直接分配

* 处理流程

略

* 功能简述

通过快速定位 商品,直接分配地市可要货量，包括分配本月和下月要货量。

* 数据库表结构

TO\_STOCK\_REGION

TO\_STOCK\_DIST\_LOG

TO\_STOCK\_REALTIME

* 详细设计

1. 界面包括2部分，1）商品快速定位区域；2）分配操作区域，分配操作区域包括本月分配和下月预分配。
2. 商品快速定位区域，包括条件搜索区域和商品树形展示区域。

商品树形展示区域第一级是品牌；第二级是品类；第三级是SKU名称。

定位区域通过输入品牌名称或型号名称或SKU名称进行模糊匹配，联想出满足匹配的商品SKU，选择其中一条，在树形上定位该节点，并且触发选中事件 。

1. 在商品树形区域选中具体的一个商品，显示分配操作区域。品牌、型号、SKU名

名称根据选中的节点显示，仓储下拉选择对应的仓库档案。选择仓库档案获取当前仓库库存，可分配库存数量（TO\_STOCK\_REAL）,待审核分配量：（TO\_BILL\_STOCK\_DIST根据仓库汇总状态是待审核，单据类型=‘BILL\_DSKCFP’的商品的分配数量）。实际可分配库存数量=可分配库存数量-待审核分配量。

根据选择的仓库，从仓库地市对照表中（TB\_DZ\_WARE\_REGION）获取可分配的地市信息。

SELECT TB\_DZ\_WARE\_REGION A LEFT JOIN TB\_DA\_AREA B ON A.F\_REGION\_NUM=B.F\_REGION\_NUM

LEFT JOIN (SELECT \* FROM TO\_STOCK\_REGION WHERE F\_MONTH=’传入月份’ and f\_goods\_num=’传入’ ) c on c.f\_region\_num = a.f\_region\_num

Left join (select 1 from 计划量表) on ……。

如果是当月库存分配TAB页：传入月份=“数据库当前月份“

如果是下月预分配TAB页：传入月份=“数据库当前月份的下一个月“

1. 当月库存分配TAB页（根据数据库时间的当前月份），下月预分配（根据数据库

时间的当前月份+1）。

1. 当月库存分配TAB和下月预分配TAB 中的本次分配量默认为0，可填写数字。
2. 确认保存操作：首先提示“是否保存本次分配数量”。

1）、判断当月库存分配TAB页面所有的本次分配量+下月预分配TAB页的所有的本次分配量之和是否大于 实际可分配量，如果大于提示“本次合计分配量大约实际可分配量”; 。。。**实际可分配量在保存数据前重新获取一次**。

2)、在当月库存分配TAB页面，如果某个地市的本次分配量+当月累计分配量》本月计划量。提示用户“XXX地市本月累计分配量大于计划量，不能实行分配”；

3)、保存数据。

A、将当月库存分配TAB页和下月预分配TAB页数据保存到TO\_STOCK\_REGION。

**注意传入的月份。如果主键不存在，插入数据，如果主键存在**

**F\_DIST\_COUNT=F\_DIST\_COUNT+本次分配量。建议用MERGE INTO实现。**

B、记录库存分配记录TO\_STOCK\_DIST\_LOG：商品类型为=1，分配类型为分配

1. 扣减实际可分配库存数量；表TO\_STOCK\_REALTIME中字段

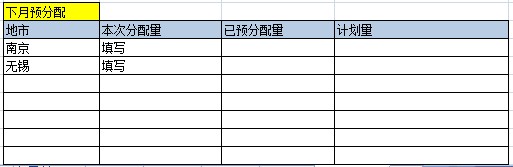
F\_DIST\_COUNT=F\_DIST\_COUNT **-** 当月库存分配TAB页中本次分配数量之和

-下月预分配TAB页中本次分配数量之和。

以上操作A\B\C保证在一次事务中执行。

* 界面设计





#### 地市库存调拨

##### 地市库存调拨申请

* 处理流程

无

* 功能简述

1. 按照品牌名称、品类名称、商品名称、仓库、地市名称和可用要货数量查询当前地市库存信息；展示页面如图：6.1.5.2.1-1；其中仓库和地市名称为下拉选择默认选项都为空；
2. 选择其中一条数据，点击“地市库存调拨”，展示地市库存调拨页面如图：6.1.5.2.1-2；其中展示信息：名称、品类名称、商品名称、仓库为选择数据时带入；并根据仓库信息列举对应所有地市信息；操作员可以根据调拨的类型1-调入2-调出，并调整各个地市的调拨数量，并输入备注信息，点击“提交申请”即可；
3. 调拨申请页面中，在点击提交时，需要前后台校验调出的合计数量必须严格等于调入合计数量；调出的数量必须小于等于地市可用要货量；
4. 调入是对地市库存做加法运算，调出是对地市库存做减法运算；
5. 修改地市调拨数量时，必须选择调拨方式，输入数据后页面动态的调整请页面左下角动态展示调入数量和调出数量、地市调拨后可用要货量，且调出数量不允许大于当前地市可用要货量；如大于页面立即提示调拨数量不允许操作当前地市的可用要货量；
6. 调拨方式默认为：不调拨，调拨数量默认为0；但如满足自动调拨的条件，页面初始化时默认展示计算后的调拨方式；
7. 自动调拨计算方式：默认有地市低于系统设置的阀值0（系统配置参数为：REGION\_WARE\_DIST\_IN\_NUM）时，自动计算调拨方式，将其余地市可用要货数量大于10（系统配置参数为：REGION\_WARE\_DIST\_OUT\_NUM）的地市可用要货数量乘于10%（系统配置参数为：REGION\_WARE\_DIST\_RATE）默认四舍五入计算方式，得出总共调出的数量，将总共调出的数量平均调拨给需调入的地市；

如：徐州和南京某机型可提货量为0时，宿、盐、淮各有133、55、8台，则淮安的提货量不变，宿迁和盐城各拿出10%（宿：13台；盐：6台；总计19台）给徐州和南京；徐州调入数量=19/2=9台；南京调入数量=19-9=10台；

1. 新增单据申请信息插入TO\_BILL\_STOCK\_DIAL，并调用单据组件recordBillInfo接口记录单据信息；渠道编码默认均为“STQD”；单据状态为待审核；
2. 保存时，后台需要校验当前是否存在相同的调拨单，依据仓库编码、商品编码、单据状态（待审核）；
3. 本功能只提供给省统一运营人员使用；调拨单据类型：BILL\_DSKCDB

* 数据库表结构

见TO\_BILL\_STOCK\_DIAL

* 详细设计

6.1.5.2.1-1界面查询参考SQL:

SELECT

BRAND.F\_BRAND\_NAME,

WARE.F\_WARE\_NAME,

AREA.F\_AREA\_NAME,

CATE.F\_CATEGORY\_NAME,

GOODS.F\_GOODS\_NAME,

STOCK. F\_DIST\_COUNT- F\_ORDER\_COUNT,--可要货量

LIMIT. F\_MIN\_VALUE --库存预警阀值

FROM TO\_STOCK\_REGION STOCK

LEFT JOIN TB\_DZ\_WARE\_REGION DZON STOCK.F\_REGION\_NUM = DZ. F\_REGION\_NUM

LEFT JON TB\_DA\_WARE WARE ON WARE.F\_WARE\_NUM= DZ.F\_WARE\_NUM

LEFT JOIN TO\_REGION\_GOODS\_LIMIT LIMIT ON LIMIT.F\_GOODS\_NUM = STOCK.F\_GOODS\_NUM AND STOCK.F\_REGION\_NUM = LIMIT.F\_REGION\_NUM

LEFT JOIN TO\_DA\_GOODS GOODS ON GOODS.F\_GOODS\_NUM = STOCK.F\_GOODS\_NUM

LEFT JOIN TB\_DA\_CATEGORY CATE ON CATE.F\_CATEGRY\_NUM = GOODS.F\_CATEGORY\_NUM

LEFT JION TB\_DA\_BRAND BRAND BRAND.F\_BRAND\_NUM = CATE.F\_BRAND\_NUM

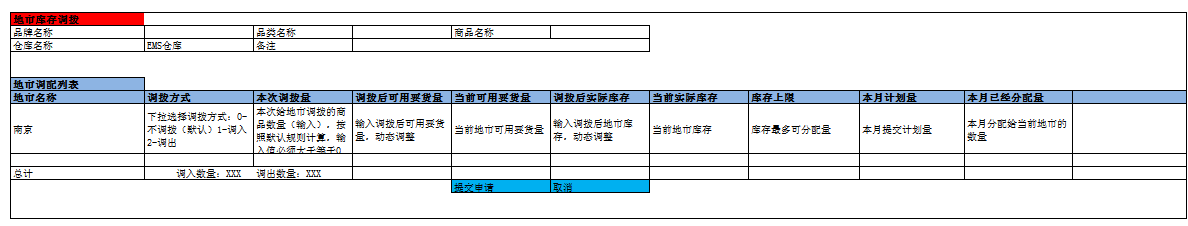
LEFT JOIN TB\_DA\_AREA AREA ON AREA.F\_AREA\_NUM = STOCK.F\_REGION\_NUM

* 界面设计



（6.1.5.2.1-1）

调拨后实际库存,当前实际库存界面上不处理,不显示



（6.1.5.2.1-2）

##### 地市库存调拨修改

* 处理流程

无

* 功能简述

1. 按照单据号、制单开始时间、制单结束时间、品牌名称、品类名称、商品名称、仓库名称查询当前地市库存调拨信息，单据状态为“待审核”或“审核不通过”；如图：6.1.5.2.2-1
2. 选择一条数据，点击“删除”按钮，提交后台，调用单据组件接口记录单据变更记录，单据状态变更为“已删除”；
3. 选择一条数据，点击“修改”按钮，展现地市库存调拨修改页面，如图：6.1.5.2.2-2；如单据状态为审核不通过，且后台有相同仓库和商品待审核的调拨单据，则直接提示用户相同的单据信息存在，请删除本条数据；
4. 修改数据后进行提交，数据约束关系见6.1.5.11，提交后，单据状态变更为“待审核”；
5. 本功能只提供给省统一运营人员使用；

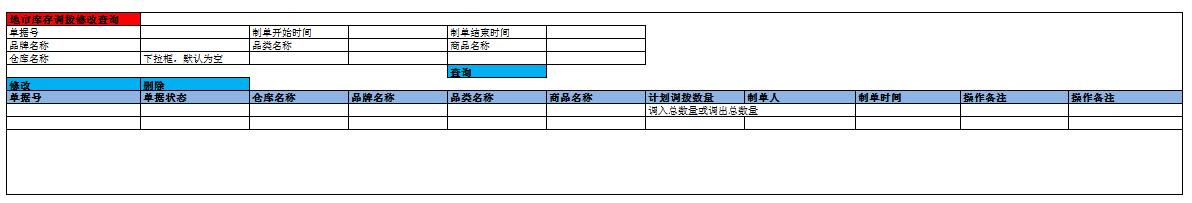
* 数据库表结构

见TO\_BILL\_STOCK\_DIAL

* 详细设计

无

* 界面设计



（6.1.5.2.2-1）

调拨后实际库存,当前实际库存界面上不处理,不显示



（6.1.5.2.2-2）

##### 地市库存调拨审核

* 处理流程

无

* 功能简述

1. 按照制单人、制单开始时间、制单结束时间、品牌名称、品类名称、商品名称、仓库名称查询地市库存调拨待审核信息；如图：6.1.5.2.3-1；点击查询即可展示对应条件的统计数据；
2. 选择其中一条数据，点击“审核”按钮，展现“地市库存调拨审核”页面，如图：6.1.5.2.3 -2；展示页面之前需要判断当前给各个地市调出的数量是否依然小于等于地市的可用要货量，和当前单据各个地市之间的调入量是否等于调出量；不满足上述条件将原因提示给用户，展示页面时只允许用户做审核不通过操作；
3. 审核不通过操作，记录审核原因，并变更单据接口记录单据状态为“审核不通过”；
4. 审核通过操作，变更单据状态为“审核通过”，根据调入和调出操作依次调整地市二级库存信息，同时记录地市二级库存变更日志；以上操作必须在同一个事物下操作。
5. 本功能只提供给省统一运营人员使用；

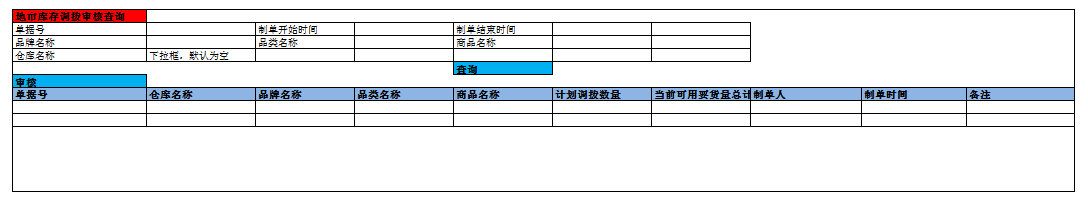
* 数据库表结构

见TO\_BILL\_STOCK\_DIAL、TO\_STOCK\_REGION、TO\_STOCK\_DIST\_LOG

* 详细设计

无

* 界面设计



（6.1.5.2.3 -1）

调拨后实际库存,当前实际库存界面上不处理,不显示



（6.1.5.2.3-2）

##### 地市库存调拨单据查询

* 处理流程

无

* 功能简述

1. 按照品牌名称、品类名称、商品名称、地市名称查询地市库存调拨单据信息；如图：6.1.5.2.4-1；点击查询即可展示对应条件的统计数据；查询时排序统一按照时间倒序排序；
2. 点击“导出”，可按照当前页面的查询条件导出Excel格式数据；
3. 本功能只提供给省统一运营人员使用；点击查看按钮，展示界面如图：6.1.5.2.4-2；

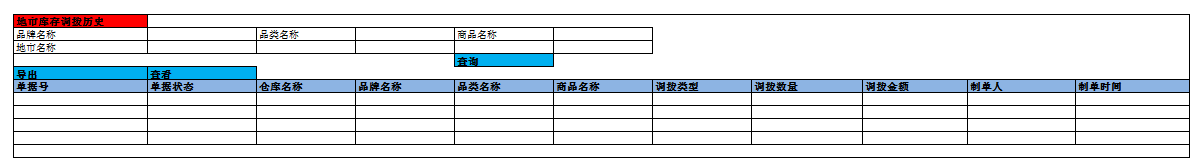
* 数据库表结构

见TO\_BILL\_STOCK\_DIAL, TB\_BILL\_LOG；

* 详细设计

无

* 界面设计



（6.1.5.2.4-1）



（6.1.5.2.4-2）

#### 地市库存预分配

##### 地市库存预分配

* 处理流程
* 功能简述
* 数据库表结构
* 详细设计
* 界面设计

##### 地市库存预分配生效

* 处理流程
* 功能简述
* 数据库表结构
* 详细设计
* 界面设计

#### 紧俏机型分配

* 处理流程



##### 省分配策略制定

* 处理流程

省分配紧俏商品，地市接收

* 功能简述

省管理人员选择紧俏商品，分配给各个地市，地市紧俏商品库存增加；

* 数据库表结构
  + 商品分货策略表 TO\_GOODS\_DIST\_RULE
  + 商品分货策略子表 TO\_GOODS\_DIST\_RULE\_SUB
  + 仓库实时库存表 TO\_STOCK\_REALTIME
  + 地市虚拟库存表 TO\_STOCK\_REGION
  + 地市系统参数表 T\_REGION\_SYSTEM,T\_REGION\_SYSTEM\_DZ
  + 仓库档案表 TB\_DA\_WAREHOUSE，TB\_DZ\_WARE\_REGION
* 详细设计
* 初始化：

界面如【图1】所示，展示省分配的历史记录；按照分配时间倒序排序；

调用【IGoodsDistRuleService. queryDistRule】方法，获取分配记录；

* 查询：

同“初始化”功能

* 制定策略：
* 初始化

1. 点击【图1】界面【制定】按钮，弹出【图2】省制定分货策略填写页面。
2. 查询TB\_DA\_WAREHOUSE，获取仓库列表
3. 选择仓库，关联TB\_DZ\_WARE\_REGION查询出该仓库对应的地市，并显示在表格中；
4. 通过商品组件查询商品，关联TO\_STOCK\_REALTIME表，查询该商品在该仓库对应的库存数量，并展示
5. 在表格中编辑，填入各地市的分配量

* 提交制定计划

1. 点击【图2】【提交】按钮
2. 调用【IGoodsDistRuleService. insertDistRule】插入主表信息
3. 按地市循环调用【IGoodsDistRuleService. insertDistRuleSub】插入子表信息

如果T\_REGION\_SYSTEM\_DZ中，F\_CSBM=‘DSSFFPJQJXYYT’的参数值=1，则TB\_GOODS\_DIST\_RULE\_SUB表，F\_STATUS=1未分配；F\_IS\_MJ=0

如果T\_REGION\_SYSTEM\_DZ中，F\_CSBM=‘DSSFFPJQJXYYT’的参数值=0，则TB\_GOODS\_DIST\_RULE\_SUB表，F\_STATUS=2已分配；F\_IS\_MJ=1

1. 修改TO\_STOCK\_REALTIME，条件：F\_ORG\_NUM = ‘99999999’， F\_LOCATE\_NUM=’0000’， F\_WARE\_NUM=‘仓库编码’， F\_GOODS\_NUM=’商品编码’， F\_STOCK\_TYPE\_NUM=’ZPK’

修改F\_DIST\_COUNT = F\_DIST\_COUNT–分配数量

1. 按地市修改TO\_STOCK\_REGION表，F\_DIST\_COUNT= F\_DIST\_COUNT + 分配数量，

F\_UPDATE\_TIME = TO\_CHAR(SYSTIMESTAMP,'YYYYMMDDHH24MISSFF3')

条件：F\_STOCK\_MONTH，F\_CHANNEL\_NUM，F\_REGION\_NUM，F\_GOODS\_NUM

1. 以上步骤需在同一个事务里进行

* 查看策略：

1. 【图1】界面，选择一条记录，点击【查看】按钮，弹出【图2】界面；
2. 调用【IGoodsDistRuleService. getDistRuleByRuleNum】方法，获取主表信息
3. 调用【IGoodsDistRuleService.getDistRuleSubByRuleNum】方法，获取子表信息列表展示

* 界面设计

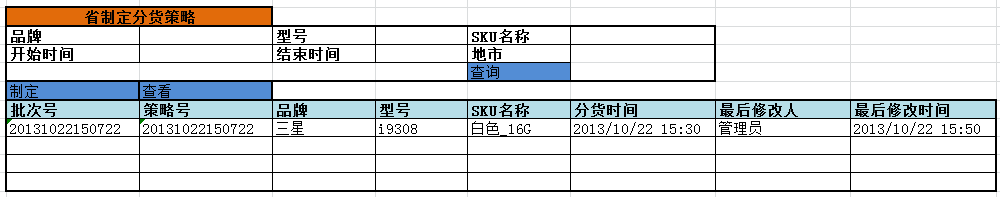


图1

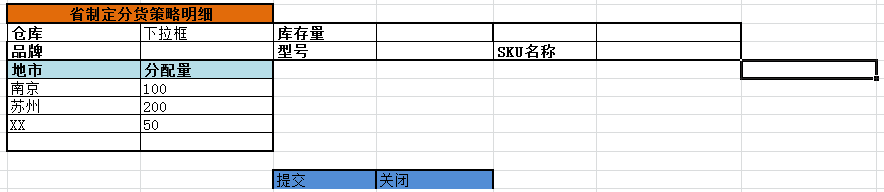


图2

##### 地市分配策略制定

* 处理流程

1. 地市分配到区县
2. 地市分配到营业厅

* 功能简述

1. 如果系统参数设置区县可以制定策略，则地市分配至区县
2. 如果系统参数设置区县不可以制定策略，则地市分配至营业厅

* 数据库表结构
  + 商品分货策略表 TO\_GOODS\_DIST\_RULE
  + 商品分货策略子表 TO\_GOODS\_DIST\_RULE\_SUB
  + 地市系统参数表 T\_REGION\_SYSTEM,T\_REGION\_SYSTEM\_DZ
* 详细设计
* 初始化：

界面如【图1】所示，展示省分配的历史记录；按照分配时间倒序排序；

调用【IGoodsDistRuleService.queryDistRuleSub】方法，获取分配记录；

默认条件：

F\_STATUS=1，F\_IS\_USE=1，F\_ORG\_TYPE=1，F\_ORG\_NUM=地市编码

* 查询：

连接UNION查询TB\_GOODS\_DIST\_RULE，TB\_GOODS\_DIST\_RULE\_SUB，主表均为已分配的策略，子表只查询未分配的策略；详见如下SQL

* 制定策略：

业务规则：只能在省分配的有效期时间段内进行分配；

分配的总数量<= 分配数量–已分配数量 + 回收数量

* 初始化

1. 点击【图1】界面【制定】按钮，

如果T\_REGION\_SYSTEM\_DZ中，F\_CSBM=‘QXFPJQJX’的参数值=1弹出【图2】地市分配区县制定分货策略填写页面。

如果T\_REGION\_SYSTEM\_DZ中，F\_CSBM=‘QXFPJQJX’的参数值=0弹出【图3】地市分配营业厅制定分货策略填写页面。

1. 【图2】页面，批次号显示对应的主表的批次号，商品信息均显示【图1】页面中选中的相应信息；

查询T\_REGION\_SYSTEM\_DZ，F\_CSBM =‘REGION\_AREA\_TYPE’参数，如果为区县，则查询该地市下所有区县，展示在表格中；

如果为营销单元，则查询该地市下所有营销单元，展示在表格中；

1. 地市选择的分配有效期范围必须小于等于省分配的有效期
2. 【图3】页面，批次号显示对应的主表的批次号，商品信息均显示【图1】页面中选中的相应信息；



通过该条件查询营业厅，显示在表格中（可以多选），点击【加入选择】按钮，将选择的营业厅加入到下方的表格中；在下方表格中填写营业厅对应的分货量；

* 提交策略-区县/营销单元【图2】

1. 点击【提交】按钮，传输数据至服务器端
2. 调用【IGoodsDistRuleService. insertDistRule】方法，将地市作为分配人存入主表，F\_WAVE\_NUM为省级分配的F\_WAVE\_NUM；
3. 循环调用【IGoodsDistRuleService. insertDistRuleSub】方法，将区县/营销单元作为被分配者存入子表；其中F\_PARENT\_RULE\_SUB\_NUM=地市策略的F\_RULE\_SUB\_NUM；F\_IS\_MJ=0；F\_STATUS=1；F\_ORG\_TYPE=2|3
4. 调用【IGoodsDistRuleService.updateDistRuleSub】方法，根据地市策略的F\_RULE\_SUB\_NUM修改F\_USED\_COUNT，F\_USED\_COUNT=F\_USED\_COUNT+分配给区县的总数量

* 提交策略-营业厅【图3】

1. 点击【提交】按钮，传输数据至服务器端
2. 调用【IGoodsDistRuleService. insertDistRule】方法，将地市作为分配人存入主表，F\_WAVE\_NUM为省级分配的F\_WAVE\_NUM；
3. 循环调用【IGoodsDistRuleService. insertDistRuleSub】方法，将营业厅作为被分配者存入子表；其中F\_PARENT\_RULE\_SUB\_NUM=地市策略的F\_RULE\_SUB\_NUM；F\_IS\_MJ=1；F\_STATUS=1；F\_ORG\_TYPE=4
4. 调用【IGoodsDistRuleService.updateDistRuleSub】方法，根据地市策略的F\_RULE\_NUM修改F\_USED\_COUNT，F\_USED\_COUNT=分配给营业厅的总数量

* 查看策略：
* 待分配策略，无法查看
* 已分配策略，根据策略编码查询策略子表，调用【IGoodsDistRuleService.getDistRuleSubByRuleNum】方法，如果策略子表为区县/营业厅，则如【图2】展示；如果策略子表为营业厅，则如【图4】展示；
* 界面设计

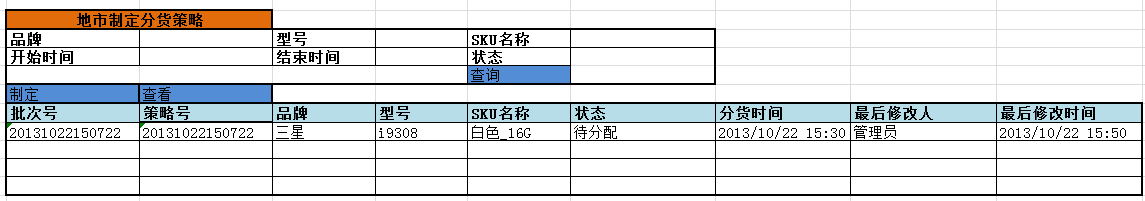


图1

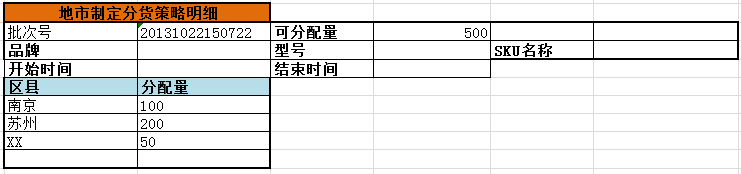


图2



图3



图4

##### 区县/营销单元分货策略制定

* 处理流程

区县/营销单元分货至营业厅

* 功能简述

区县/营销单元分货至营业厅

* 数据库表结构
  + 商品分货策略表 TO\_GOODS\_DIST\_RULE
  + 商品分货策略子表 TO\_GOODS\_DIST\_RULE\_SUB
* 详细设计
* 初始化：

界面如【图1】所示，展示地市分配的历史记录；按照分配时间倒序排序；

调用【IGoodsDistRuleService.queryDistRuleSub】方法，获取分配记录；

默认条件：

F\_STATUS=1，F\_IS\_USE=1，F\_ORG\_TYPE=1，F\_ORG\_NUM=区县/营销单元

* 查询：

连接UNION查询TB\_GOODS\_DIST\_RULE，TB\_GOODS\_DIST\_RULE\_SUB，主表均为已分配的策略，子表只查询未分配的策略；详见如下SQL

* 制定策略：

业务规则：只能在地市分配的有效期时间段内进行分配；

分配的总数量<= 分配数量–已分配数量 + 回收数量

* 初始化

1. 点击【图1】界面【制定】按钮，弹出【图2】区县分配营业厅制定分货策略填写页面。
2. 区县选择的分配有效期范围必须小于等于地市分配的有效期
3. 【图2】页面，批次号显示对应的主表的批次号，商品信息均显示【图1】页面中选中的相应信息；



通过该条件查询营业厅，显示在表格中（可以多选），点击【加入选择】按钮，将选择的营业厅加入到下方的表格中；在下方表格中填写营业厅对应的分货量；

* 提交策略【图2】

1. 点击【提交】按钮，传输数据至服务器端
2. 调用【IGoodsDistRuleService. insertDistRule】方法，将区县作为分配人存入主表，F\_WAVE\_NUM为上级分配的F\_WAVE\_NUM；
3. 循环调用【IGoodsDistRuleService. insertDistRuleSub】方法，将营业厅作为被分配者存入子表；其中F\_PARENT\_RULE=区县策略的F\_RULE\_NUM；F\_IS\_MJ=1；F\_STATUS=1；F\_ORG\_TYPE=4
4. 调用【IGoodsDistRuleService.updateDistRuleSub】方法，根据区县策略的F\_RULE\_NUM修改F\_USED\_COUNT，F\_USED\_COUNT=分配给营业厅的总数量

* 查看策略：
* 待分配策略，无法查看
* 已分配策略，根据策略编码查询策略子表，调用【IGoodsDistRuleService.getDistRuleSubByRuleNum】方法，如【图4】展示；
* 界面设计

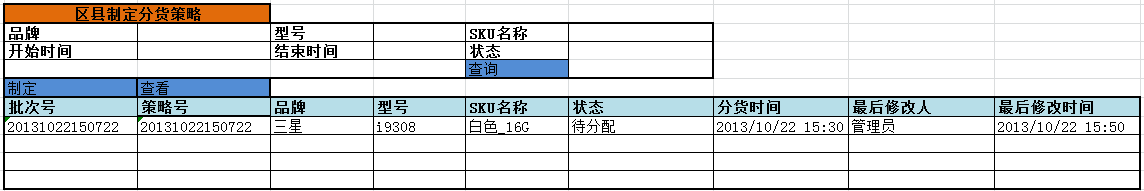


图1



图2



图3

##### 紧俏机型回收

* 处理流程

1.营业厅回收>>区县回收>>地市回收>>省

2.营业厅回收>>地市回收>>省

3.地市回收>>省

* 功能简述

系统通过数据库job自动回收过期的紧俏机型库存，分为以下几种方式：

1.营业厅回收>>区县回收>>地市回收>>省

2.营业厅回收>>地市回收>>省

3.地市回收>>省

* 数据库表结构
  + 商品分货策略表 TO\_GOODS\_DIST\_RULE
  + 商品分货策略子表 TO\_GOODS\_DIST\_RULE\_SUB
  + 回收记录表 TO\_GOODS\_RECOVERY\_LOG
  + 仓库实时库存表 TO\_STOCK\_REALTIME
  + 地市虚拟库存表 TO\_STOCK\_REGION
* 详细设计

如果过了有效期，则返回至上一级策略，直到返回省；

紧俏机型自动回收存储过程：

* 界面设计

##### 库存查询

* 处理流程
* 功能简述

查询机构（地市/区县/营业厅）当前库存

* 数据库表结构
  + 商品分货策略表 TB\_GOODS\_DIST\_RULE
  + 商品分货策略子表 TB\_GOODS\_DIST\_RULE\_SUB
* 详细设计

如【图1】所示，用户选择商品，选择机构类型，机构，查询当前库存；

查询SQL

|  |  |
| --- | --- |
| 查询机构当前库存 | SELECT T.F\_ORG\_TYPE,  T.F\_ORG\_NUM,  T.F\_GOODS\_NUM,  SUM(T.F\_GOODS\_COUNT) F\_GOODS\_COUNT,  SUM(T.F\_RECOVERY\_COUNT) F\_RECOVERY\_COUNT,  SUM(T.F\_USED\_COUNT) F\_USED\_COUNT,  SUM(T.F\_GOODS\_COUNT - T.F\_USED\_COUNT + T.F\_RECOVERY\_COUNT) AVAILABLE\_COUNT  FROM TO\_GOODS\_DIST\_RULE\_SUB T  WHERE  (T.F\_ORG\_TYPE =1AND T.F\_ORG\_NUM ='?'AND T.F\_IS\_MJ =1)  OR(T.F\_ORG\_TYPE =4AND T.F\_ORG\_NUM ='?')  AND T.F\_GOODS\_NUM ='?'  AND TO\_CHAR(SYSDATE,'YYYYMMDDHH24MISS')>= T.F\_START\_TIME  AND TO\_CHAR(SYSDATE,'YYYYMMDDHH24MISS')<= T.F\_END\_TIME  AND T.F\_IS\_USE =1  GROUPBY T.F\_ORG\_TYPE, T.F\_ORG\_NUM, T.F\_GOODS\_NUM ; |

* 查询条件规则：



以上3个查询条件，按照地市》区县》营业厅 3个级别进行展示；

1. 用户如果选择多个条件，则按最低级别的机构进行查询；如：选择了南京市，鼓楼区，则查询鼓楼区的库存；同样，选择营业厅，则按营业厅进行库存查询；
2. 如果用户选择营业厅级别，则必须选择营业厅；

* 界面设计

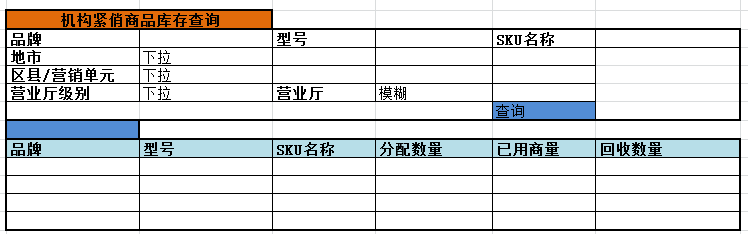


图1

##### 紧俏机型回收记录查询

* 处理流程
* 功能简述

查询紧俏机型回收记录

* 数据库表结构

紧俏机型回收记录表 TO\_GOODS\_RECOVERY\_LOG

* 详细设计

查询TO\_GOODS\_RECOVERY\_LOG表

* 界面设计



图1

### 串号管理

#### 串号退库管理

##### 串号退库申请

* 处理流程



* 功能简述

用户提交串号退库申请，生成串号退库申请单，单据状态为待审核。

* 数据库表结构

处理过程与其他单据类似。单据主表见TB\_BILL\_MAIN,业务单据表

TO\_BILL\_OUTIMEI。

* 详细设计

**串号退库申请**

1. 选择串号文件：文件格式只支持（.XLS）。文件内容格式如下：

|  |
| --- |
| 串号 |
| XXXX |

1. 点击上传，利用（共用的XLS）读取组件，将EXCEL文件中的数据读入到内存中。

在读取的时候，需要去除串号前后空字符串，如果遇见第一列数据为空，跳过处理。

读取完成后，如果读入到内存中的数据为空，提示用户“你导入的串号文件为空”；

将内存中的数据保存到串号临时表中IOrder.saveDelivTempImei。其中每次导入都产生一个导入批次号。

根据临时表中的串号信息，获取商品编码，串号状态信息。

UPDATE TEMP\_LAOD\_IMEI TEMPIMEI SET F\_GOODS\_NUM,F\_CHECK\_IMEI\_STATUS = (SELECT F\_GOODS\_NUM,F\_IMEI\_STATUS FROM TB\_GOODS\_IMEI IMEI WHERE TEMPIMEI.F\_GOODS\_IMEI = IMEI.F\_IMEI)

WHERE F\_BATCH\_NUM=’导入批次号’ and exists( select 1 from TB\_GOODS\_IMEI IMEI2 WHERE TEMPIMEI.F\_GOODS\_IMEI = IMEI2.F\_IMEI)

**退库申请提交页面**

1. 初始化

获取退库原因IOrder.getReasonByItem(原因类型编码) 原因类型编码=‘CHTKYYX’; 参考SQL：

SELECT \* from TB\_DA\_REASON where F\_REASON\_ITEM\_NUM=‘F\_REASON\_ITEM\_NUM’ AND F\_IS\_USE=1

获取导入串号信息

SELECT FROM TEMP\_LAOD\_IMEI LOAD

LEFT JOIN TO\_DA\_GOODS

LEFT JOIN TO\_DA\_CATEGORY

LEFT JOIN TO\_DA\_BRAND WHERE F\_BATCH\_Num=?

合计导入：记录总数

可退库记录数：串号状态不等于退库在途，配送在途，调拨在途，并且商品编码不能为空(循环记录结果,可获取)

不可退库记录：合计导入-可退库记录数

1. 提交申请

判断如果可退库记录数》0 提示。存在不能退库串号状态(退库在途，配送在途，调拨在途)，请重新导入。

保存数据到TO\_BILL\_OUTIMEI,可以通过 INSERT INTO TO\_BILL\_OUTIMEI SELECT FROM TEMP\_LAOD\_IMEI 实现.

操作相关表包括TB\_BILL\_MAIN,TB\_BILL\_LOG ( 参照可调用以前单据操作的方法)

其中将退库原因信息（名称）记录到TB\_BILL\_MAIN的备注信息里

退库原因编码　记录到　TO\_BILL\_OUTIMEI每条记录里面

* 界面设计

**串号退库申请页面**



**串号退库申请提交页面**



##### 串号退库审核

* 处理流程
* 功能简述

查询待审核串号退库申请,并且进行审核.

* 数据库表结构
* 详细设计

1. 串号退库审核查询页面

查询条件初始化

申请开始时间：默认当年年月第一天+“００：００：００”；

申请结束时间：当前时间

该页面只能查询单据状态是待审核的串号申请单。

1. 串号退库审核页面

进入页面

表头信息：查询单据主表，表头信息　除了审核备注可以修改，其他不可以修改。

商品信息：查询TO\_BILL\_OUTIMEI表

审核和审核不通过

修改to\_bill\_main单据状态，并且记录单据日志信息。

同时更改tb\_goods\_imei表里的ＩＭＥＩ状态。（批量修改）

并且记录tb\_goods\_imie\_log.IMEI变更记录（批量插入）

1. **串号退库单查看页面**

**做成通用页面。所有字段都不可以修改**

* 界面设计

**串号退库审核查询页面**



**串号退库审核页面**



**串号退库单查看页面**



##### 串号退库查询

* 处理流程

略

* 功能简述

查询串号退库单信息

* 数据库表结构

TO\_BILL\_OUTIMEI，TB\_BILL\_MAIN

* 详细设计

串号退库审核查询页面

查询条件初始化

申请开始时间：默认当年年月第一天+“００：００：００”；

申请结束时间：当前时间

　　　　单据状态：查询表tb\_bill\_status where f\_bill\_type=?。

查看：

　　进入**串号退库单查看页面**

* 界面设计

**串号退库单查询页面**



#### 串号查询统计

##### 串号查询

* 处理流程
* 功能简述

查询串号信息

* 数据库表结构

TB\_GOODS\_IMEI ,TB\_GOODS\_IMEI\_LOG

* 详细设计

1. 串号支持一次输入多个，多个串号以”,”分隔。

２、串号必须输入，并且在查询时通过全匹配的方式。

变更历史：

通过ＩＭＥＩ号查询tb\_goods\_imei\_log 表，根据F\_RECORD\_TIME倒序排序。

* 界面设计

串号查询页面



串号变更历史

C:\Users\sunky\AppData\Roaming\Tencent\Users\11350631\QQ\WinTemp\RichOle\(@~4@9LB)RPJHJL[E$RYADW.jpg

#### 备选串号管理

* 处理流程
* 功能简述

维护备选串号与主串号的对照关系，分为以下功能：

批量导入，修改（单个操作）

* 数据库表结构

TB\_BOSS\_IMEI，TB\_GOODS\_IMEI , TO\_DZ\_IMEI\_ALTERNATIVE

* 详细设计
* 初始化：如【图1】所示，默认为空
* 查询：如【图1】所示

1. 根据选择的商品关联TB\_BOSS\_IMEI，TB\_GOODS\_IMEI , TO\_DZ\_IMEI\_ALTERNATIVE查询
2. 按照串号查询，全匹配查询

* 导入串号：

1. 点击【图1】【导入串号】按钮，弹出【图2】界面；
2. 点击【选择文件】，选择需要的串号文件，点击【导入串号】按钮，系统提交串号列表至后台；
3. 校验串号，
   1. 如果串号存在TB\_BOSS\_IMEI表，则串号校验成功
   2. 如果串号不存在TB\_BOSS\_IMEI表，则串号校验失败

本步骤不做数据库插入操作，仅作查询校验，如【图2】所示，返回导入预期结果；

1. 串号存储

用户查看第3部的校验结果，如果串号校验成功，则用户点击【提交】按钮，将串号批量插入TO\_DZ\_IMEI\_ALTERNATIVE；

如果校验失败，则用户修改，修改成功后，插入TO\_DZ\_IMEI\_ALTERNATIVE；

插入TO\_DZ\_IMEI\_ALTERNATIVE表时，需where条件重新校验；

* 界面设计



图1



图2

#### 串号调整管理

##### 串号调整维护

##### 串号调整处理

* 处理流程

无

* 功能简述

1. 点击功能菜单后，打开【串号调整处理页面】（图 -1），用户点击【查询】按钮，根据查询条件将符合条件的单据信息查询出来并在列表中展现。
2. 初始化

* 单据状态：待处理和处理失败
* 制单时间 倒序排列

1. 查询条件说明：

* 原单据号，串号，商品，型号，商品SKU名称后台模糊查询。
* 原单据类型：下拉列表，【串号调整类型】（TO\_CHANGESOURCE\_TYPE），包括全选。
* 开始时间、结束时间，提供日期选择控件。
* 【处理状态】(F\_STATUS\_NUM)：1-待处理、3-处理失败及全部（全部只包括1和3）。

1. 点击【处理】按钮，打开【串号调整处理详情页面】（图 - 2）。
2. 点击【处理】按钮，保存数据。
3. 对串号进行校验。

* 【变更类型】=1-增加

校验【一级BOSS串号】（TB\_BOSS\_IMEI）是否存在

* 不存在【处理状态描述】：一级boss不存在，不能入库。
* 【变更类型】=2-调整

校验【串号档案表】（TB\_GOODS\_IMEI）是否存在

* 不存在【处理状态描述】：串号档案不存在，不能调整。

1. 保存数据。

* 调用（IChangeimeiService. updateChangeimei）方法，保存串号调整表信息。
* 校验通过

【处理状态】（F\_STATUS\_NUM）：2-处理成功

【处理状态描述】（F\_STATUS\_DESC）：处理完成。

* 校验不通过

【处理状态】（F\_STATUS\_NUM）：3-处理失败

【处理状态描述】（F\_STATUS\_DESC）：见1）。

* 校验通过，保存串号档案表信息【串号档案表】（TB\_GOODS\_IMEI）。
* 【变更类型】=1-增加，在串号档案表中增加该串号信息。状态记录ZK-在库。存在则修改，不存在则增加。
* 【变更类型】=2-调整，串号档案表中修改该串号状态。
* 【供货商编码】为空则取该商品对应供货商。
* 校验通过，增加串号操作日志【串号变更日志表】（TB\_GOODS\_IMEI\_LOG）。

所有操作在同一个事务中完成。

1. 【串号调整处理页面】多选后，点击【批量处理】按钮，执行批量审核操作。
2. 批量审核对单个串号进行处理，处理完成一个串号，保存一个。
3. 处理逻辑见【处理】功能。
4. 处理完成后弹出【批量处理结果页面】（图 - 3），告知用户处理结果。
5. 【串号调整处理页面】点击【导出】按钮，将列表中所有数据导出EXCEL文件。
6. 导出文件格式见【EXCEL格式】（图 - 4）

* 数据库表结构

串号调整表（TO\_CHANGEIMEI）

串号调整类型（TO\_CHANGEIMEI\_TYPE）

* 详细设计

无

* 界面设计

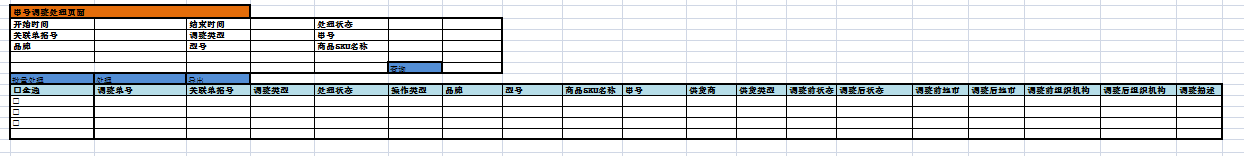


图 -1



图 -2



图 -3

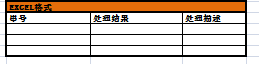


图 -4

##### 串号调整跟踪

* 处理流程

无

* 功能简述

1. 点击功能菜单后，打开【串号调整单跟踪页面】（图 -1），用户点击【查询】按钮，根据查询条件将符合条件的单据信息查询出来并在列表中展现。
2. 初始化

* 制单时间 倒序排列

1. 查询条件说明：

* 原单据号，串号，商品，型号，商品SKU名称后台模糊查询。
* 原单据类型：下拉列表，【串号调整类型】（TO\_CHANGESOURCE\_TYPE），包括全选。
* 开始时间、结束时间，提供日期选择控件。
* 变更单状态：下拉列表，包括全部。

1. 点击【查看详情】按钮，打开【串号调整单跟踪详情页面】（图 - 2）。
2. 点击【导出】按钮，将列表中所有数据导出EXCEL文件。

* 数据库表结构

串号调整表（TO\_CHANGEIMEI）

串号调整类型（TO\_CHANGEIMEI\_TYPE）

* 详细设计

无

* 界面设计

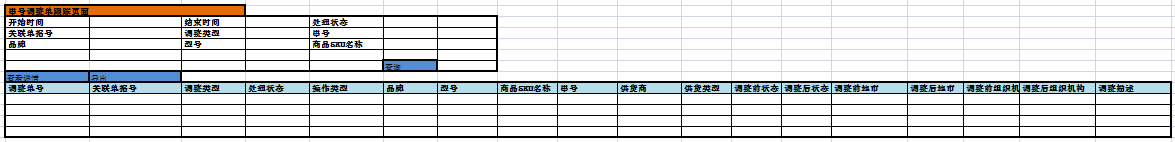


图 -1



图 -2

## 订单管理中心

### 采购订单管理

**状态变化图：**



#### 采购订单维护

* 处理流程

**1、订单维护主界面维护流程**

2、**采购订单新增和保存处理流程**



3 、**采购订单修改处理流程**

**4、采购订单作废操作流程**



* 功能简述

采购订单维护主要包括功能点：

1. 根据查询条件查询出满足条件的所有的采购订单，并且分页显示，订单状态必须是待审核和审核不通过的订单。
2. 针对查询的采购订单结果集，可进行作废和修改操作。
3. 可通过新增页面，新增采购订单。

界面包括：采购订单维护界面；采购订单新增界面；采购订单作废界面；采购订单修改界面；

* 数据库表结构

见TO\_ORDER\_HANDLE，TO\_ORDERPURCH\_MAIN，XW\_ORDERPURCH\_SUB

* 详细设计

1. 采购订单查询条件初始化

下单开始时间：默认当前年月第一天+” 00:00:00”；

下单结束事件：默认当前时间

订单状态：下拉选择包括（所有、待审核、审核不通过），默认显示全部

供货商：从TB\_DA\_SUPPLIER中配送方式=统一配送并且是否启用 =是的供货

商信息，下拉选择，默认所有供货商。

仓储：从TB\_DA\_WAREHOUSE档案中获取F\_IS\_USE=是的仓储，默认所有。

1. 采购订单查询结果集获取

根据TO\_ORDERPURCH\_MAIN表并且状态是=待审核或审核不通过的订单，已分页的形式查询和显示。

1. 采购订单新增页面初始化

订单类型包括【CGDD-采购订单；CGTD-采购退单】，默认为采购订单

订单属性：根据订单类型动态获取

申请人：显示登录人姓名

申请时间：默认数据库当前时间

有效时间：当前日期+10天

要求到货时间：当前日期+14天

供应商、品牌、型号、SKU 四级联动选择。

根据供货商获取可供货商品信息见视图。VIEW\_SUPP\_GOODS\_UNIT。

当选择最后商品SKU的是时候，获取供货模式和价保类型，只有供货模式=买断的情况下才有价保信息。

如果商品明细列表中存在记录后，仓储，供货商，不允许修改。

1. 采购订单新增页面操作

**增行操作：**

1. 增行前供应商编码，商品SKU，仓储，数量必须填写。
2. 如果在列表中已经存在（商品编码、仓库编码、供应商编码）三者相同，提示用户该行记录已经存在，不能新增。
3. 增行时，获取供货模式和价保类型，获取采购价格，如果不存在，不允许添加，查询采购价格见视图VIEW\_AVAILABLE\_PURCH\_PRICE传入商品编码，供货商编码，地市编码=99。

采购金额=采购进价×采购数量，采购税金 = 采购金额×商品税率

1. 将数量、供应商、商品SKU、仓储、保存到内存中，并显示在界面上。并且界面上供应商、仓储置为不允许修改。

**删除操作：**

1. 删除内存和界面上的对应的商品详情记录。如果删除后，没有商品订购记录，将供应商、仓储置为允许修改。

**保存操作：**

提示”是否确认需要提交采购订单”

判断是否存在商品列表数据,如果没有”提示用户录入需要采购的商品信息”

将界面上的商品列表，和表头信息传入后后台，生成采购订单，生成采购的订单逻辑有订单服务层实现。生成后的订单状态为待审核。

1. 采购订单修改

**采购订单修改页面初始化**

根据采购订单号获取采购订单主要信息,采购订单商品信息,采购订单主信息,除了

订单备注外,其余都不可修改.

**增行操作**

参考采购订单新增页面操作步骤增行操作。

**删除操作**

参考采购订单新增页面操作步骤删除操作。

**保存修改**

提示用户是否确认需要保存修改后的数据.

判断是否存在商品列表数据,如果没有”提示用户录入需要采购的商品信息”

保存前判断采购订单的状态是否=待审核或审核不通过,如果不是,提示用户”该采购订单状态产生变化,不能保存”;

保存业务处理: 先删除采购单号对应的采购订单子表中的数据,然后将界面上的数据传入保存,确保删除操作和保存操作在同一个事务中.

1. 采购订单作废

**界面初始化**

弹出采购订单作废页面,显示采购订单主表信息以及作废备注(可输入),以及采购订单商品数据.以及采购订单操作记录.如界面

**作废操作**

提示”是否确认要作废XXX采购订单”.

判断当前订单状态是否为待审核,或审核不通过状态,否则提示用户”订单状态发生变化,不能作废”;

更改订单状态为作废取消状态,增加订单操作记录

1. 订单详细信息[**做成组件页面**]

包括:

采购订单主信息

SELECT \* FROM TO\_ORDERPURCH\_MAIN

采购订单商品信息

SELECT \* FROM TO\_ORDERPURCH\_SUB

送货商品信息

TO\_BILL\_DELIVSTOCK\_MAIN MAIN, TO\_BILL\_DELIVSTOCK\_SUB SUBWHERE MAIN.F\_BILL\_NUM= MAIN.F\_BILL\_NUM

MAIN.F\_ORDER\_TYPE=? AndMAIN.F\_ORDER\_NUM=?

入库商品信息

TO\_BILL\_INSTOCK\_MAIN MAIN, TO\_BILL\_INSTOCK\_SUB SUB WHERE MAIN.F\_BILL\_NUM= MAIN.F\_BILL\_NUMF\_ORDER\_TYPE=? AndF\_ORDER\_NUM=?

* 界面设计
* 采购订单维护主界面



* 采购订单新增界面



* 采购订单修改界面



* 采购订单作废界面



* 采购订单详细数据



#### 采购订单审核发布

* 处理流程

**订单审核查询流程**



**订单审核流程**



* 功能简述

查询可以审核的采购订单，完成单个采购订单和批量的采购订单审核。

* 数据库表结构
* 详细设计

1. 订单审核查询页面

**界面初始化**

下单开始时间：默认当前年月第一天+” 00:00:00”；

下单结束事件：默认当前时间

供货商：从TB\_DA\_SUPPLIER中配送方式=统一配送并且是否启用 =是的供货

商信息，下拉选择，默认所有供货商。

仓储：从TB\_DA\_WAREHOUSE档案中获取F\_IS\_USE=是的仓储，默认所有。

**逻辑层**

只能查询待审核状态的采购订单

**查看**

进入采购订单详情页面

**批量审核通过**

勾选采购订单号

提示用户是否提交批量审核通过数据。循环调用采购订单审核逻辑。

1. 订单审核页面

**界面初始化**

弹出采购订单审核页面,显示采购订单主表信息以及审核备注(可输入),以及采购订单商品数据.以及采购订单操作记录.如界面

**审核通过操作**

提示”是否确认要提交XXX采购审核操作”.

判断当前订单状态是否为待审核 ,否则提示用户”订单状态发生变化,不能审核”;

更改订单状态为审核通过取消状态,增加订单操作记录。

* 界面设计

订单审核查询页面



订单审核页面



#### 采购订单取消作废

* 处理流程

**采购订单作废取消查询**



**采购订单作废取消操作**



* 功能简述
* 数据库表结构
* 详细设计

一、采购订单作废取消查询

下单开始时间：默认当前年月第一天+” 00:00:00”；

下单结束事件：默认当前时间

订单状态：下拉选择包括（所有、待审核、审核不通过），默认显示全部

供货商：从TB\_DA\_SUPPLIER中配送方式=统一配送并且是否启用 =是的供货

商信息，下拉选择，默认所有供货商。

仓储：从TB\_DA\_WAREHOUSE档案中获取F\_IS\_USE=是的仓储，默认所有。

1. 采购订单作废操作

**界面初始化**

弹出采购订单作废页面,显示采购订单主表信息以及作废备注(可输入),以及采购订单商品数据.以及采购订单操作记录.如界面

**作废操作**

提示”是否确认要作废XXX采购订单”.

判断当前订单状态是否为待审核,或审核不通过状态,否则提示用户”订单状态发生变化,不能作废”;

更改订单状态为作废取消状态,增加订单操作记录

* 界面设计

采购订单作废查询页面



采购订单作废页面



#### 采购订单延期处理

* 处理流程

**采购订单延期查询流程**

**采购订单延期处理流程**



* 功能简述

查询出可以延期的采购订单.(排除采购订单状态时取消和结束的采购订单)。

延长采购订单有效日期。

* 数据库表结构

采购订单主表

* 详细设计

一、采购订单延期查询

下单开始时间：默认当前年月第一天+” 00:00:00”；

下单结束事件：默认当前时间

供货商：从TB\_DA\_SUPPLIER中配送方式=统一配送并且是否启用 =是的供货

商信息，下拉选择，默认所有供货商。

仓储：从TB\_DA\_WAREHOUSE档案中获取F\_IS\_USE=是的仓储，默认所有。

1. 采购订单延期操作

**界面初始化**

弹出采购订单延期页面,显示采购订单主表信息以及延期备注(可输入),以及采购订单商品数据.以及采购订单操作记录.如界面

**作废操作**

提示”是否确认要延期XXX采购订单”.

判断当前订单状态如果是结束或取消 ,否则提示用户”订单状态发生变化,不能延期操作”;

修改订单有效日期，增加订单操作记录

* 界面设计

采购订单延期查询页面



采购订单延期处理页面



#### 采购订单发货

* 处理流程

**采购订单发货查询页面**

**采购订单发货详情流程**

**采购订单发货流程**



* 功能简述

查询可以发货的采购订单，选择采购订单进行发货。

发货根据采购订单，导入串号，进行发货。

* 数据库表结构
* 详细设计

1. 采购订单发货查询

下单开始时间：默认当前年月第一天+” 00:00:00”；

下单结束事件：默认当前时间

供货商：从TB\_DA\_SUPPLIER中配送方式=统一配送并且是否启用 =是的供货

商信息，下拉选择，默认所有供货商。

仓储：从TB\_DA\_WAREHOUSE档案中获取F\_IS\_USE=是的仓储，默认所有。

采购订单状态 = 待发货、部分发货状态的采购订单

有效时间》=当前时间

1. 采购订单发货详情页面

**界面初始化**

弹出采购订单发货详情页面,显示采购订单主表信息以及采购订单商品数据.以及送货记录、入库记录。如界面

**发货：**

进入发货页面。

1. 采购订单发货操作

选择文件：文件格式必须是.XLS后缀的文件。

内容格式：包括商品编码、串号。

如

|  |  |  |
| --- | --- | --- |
| 商品编码 | 串号 | 备选串号 |
| JSYD-HTC-9308-01 | 989930203021 | 989930203001 |
| JSYD-HTC-9308-01 | 989930203022 | 989930203002 |

从文件第二行开始读取。

只要第一列或第二列为空，就忽略。

遇到整行都是空直接忽略。

1. 将文件导入到临时表中TEMP\_IMEI\_LOAD

字段包括：

导入批次号(F\_LOAD\_BATCH)：每次导入生成一个批次号（GUID），不是一个串号一个批次。

导入类型(F\_LOAD\_TYPE)：CGDDFH

单据号(F\_ORDER\_NUM): 采购订单号

商品编码(F\_GOODS\_NUM)：

串号(F\_GOODS\_IMEI)：

1. 判断发货数量是否已经大于采购订单累计的发货数量。

SELECT F\_GOODS\_NUM,F\_ORDER\_NUM FROM TO\_ORDERPURCH\_SUB ORDERSUB LEFT JOIN (

SELECT F\_GOODS\_NUM,COUNT(1)F\_COUNT,F\_RDER\_NUM FROM TEMP\_IMEI\_LOAD WHERE F\_LAOD\_BATCH\_NUM=? GROUP BY F\_ORDER\_NUM,F\_GOODS\_NUM) TEMPSUB

ON ORDERSUB.F\_ORDER\_NUM = TEMPSUB.F\_ORDER\_NUM AND ORDERSUB.F\_GOODS\_NUM = TEMPSUB.F\_GOODS\_NUM

WHERE ORDERSUB.F\_GOODS\_AMOUNT-ORDERSUB.F\_DELIV\_AMOUNT-TEMPSUB.F\_COUNT<0

查询结果如果大于0,表示存在商品导入的数量大于实际可发货的数量，返回，不允许发货。提示用户，导入数量大于实际可送货数量，请重新导入。

1. 判断串号是否在一级BOSS中参考SQL

SELECT \* FROM TEMP\_IMEI\_LOAD A WHER F\_LOAD\_BATCH\_NUM =? AND F\_GOODS\_IMEI NOT EXISTS(SELECT 1 FROM TB\_BOSS\_IMEI B WHERE A.F\_GOODS\_IMEI= B.F\_GOODS\_IMEI AND F\_IMEI\_STATUS=1).

如果查询结果记录数量》0 表示，有串号不在一级BOSS中，提示用户存在串号不在一级BOSS中。如果返回记录数量=0，表示所有的串号都在一级BOSS中。

1. 验证通过后，将数据使用mergeinto 插入TO\_DZ\_IMEI\_ALTERNATIVE表，以主串号作为条件；
2. 调用采购订单发货接口。

A、获取每个商品的送货数量

SELECT F\_GOODS\_NUM,COUNT(1)F\_COUNT,F\_RDER\_NUM FROM TEMP\_IMEI\_LOAD WHERE F\_LAOD\_BATCH\_NUM=? GROUP BY F\_ORDER\_NUM,F\_GOODS\_NUM

B、更新采购订单累计发货数量

UPDATE TO\_ORDERPURCH\_SUB SET F\_DELIV\_AMOUNT= F\_DELIV\_AMOUNT+前面查询的数量 where F\_ORDER\_NUM=? AND F\_GOODS\_NUM =?

C、更新采购订单状态

SELECT 1 FROM TO\_ORDERPURCH\_SUB WHERE F\_ORDER\_NUM=? AND F\_ORDER\_TYPE=? WHERE F\_GOODS\_AMOUNT- F\_DELIV\_AMOUNT>0

如果查询记录数量》0，更行采购订单状态主表为部分发货

如果查询记录数量=0 更新采购订单状态为结束。

D、插入采购订单记录表

E、生产送货订单

插入送货单主表

插入送货单从表

插入送货单明细表（通过INSERT INTO TO\_BILL\_DELIVSTOCK\_SUB\_DETAIL SELECT \* FROM TEMP\_IMEI\_LOAD ）批量插入。

F\_IMEI\_CHECK\_STATUS= UNDO，F\_IMEI\_STATUS=未核对。

* 界面设计

发货主界面



发货详情页面



发货操作页面



#### 采购订单明细查询

* 处理流程
* 功能简述
* 数据库表结构
* 详细设计

1. 采购订单明细查询主界面

下单开始时间：默认当前年月第一天+” 00:00:00”；

下单结束事件：默认当前时间

供货商：从TB\_DA\_SUPPLIER中配送方式=统一配送并且是否启用 =是的供货

商信息，下拉选择，默认所有供货商。

仓储：从TB\_DA\_WAREHOUSE档案中获取F\_IS\_USE=是的仓储，默认所有。

订单类型：包括采购订单（默认显示）、采购退单

订单状态：根据订单类型获取订单状态。

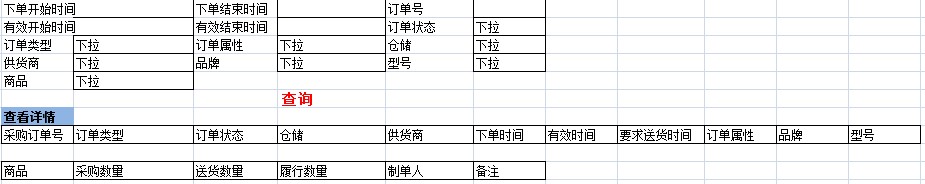
订单属性：根据订单类型获取订单属性

品牌、型号、商品实现三级联动。

点击查看详情：进入采购订单详细信息页面

* 界面设计

**采购订单明细查询主界面**



### 提货计划管理

* 处理流程



* 业务规则

提货计划页面，选择商品时，商品组件查询商品需关联“要货关系控制”进行查询，过滤不属于该地市的商品；

#### 区县提货计划维护

* 处理流程
  + 保存>>提交
  + 导入>>提交
* 功能简述

1. 新增提货计划
2. 修改“审核不通过”计划

* 数据库表结构
* （提货计划单据A）TO\_DELIV\_PLAN，（提货计划明细）TO\_DELIV\_PLAN\_DETAIL
* （业务单据主表）TB\_BILL\_MAIN
* 详细设计

1. 新增提货计划

业务规则：

* 初始化

界面如图：[6.3.1.1 – 1]，初始化相关提货计划，默认查询条件如下：

**状态：**“审核不通过”，“保存”

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

**区县/营销单元：**显示当前用户的组织区县/营销单元

**打开页面时，判断当前用户所在地市是否是“区县可以提交提货计划”，如果区县不能提交，则『新增』按钮disabled，无法做『新增』功能；**

**判断条件（T\_REGION\_SYSTEM表DELIV\_PLAN\_AREA\_TYPE字段）**

* 查询

同“初始化”

* 查看

在[6.3.1.1 – 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，弹出[6.3.1.1 – 2]界面，查看计划商品明细；点击【关闭】按钮关闭；

* 保存（临时操作）

**业务规则：**关键字-用户组织，计划周期

[6.3.1.1 – 1]界面加载时，需判断当前用户所在的组织，是否已经提交本计划周期的提货计划（包括提交完成的和正在审核过程中的），如本周期已经提交（如：该地市为月计划周期，如果本月计划已经提交，则不允许新增本月计划；旬、周计划同理），则【新增】按钮disabled掉；调用【IDelivPlanService.validAllowSubmitPlan】接口进行验证；

业务操作：

1. 点击[6.3.1.1 – 1]界面【新增】按钮，弹出[6.3.1.1 – 2]界面，此界面为商品粒度的提货计划明细；
2. 商品明细列表维护
   1. 点击[6.3.1.1 – 2]界面【新增】按钮，弹出[6.3.1.1 – 3]界面，使用“商品选择组件”进行商品选择，选择完成之后，带出“当前库存”，“上月销量”两个字段（不可修改）；调用【IDelivPlanService.getReferPlanCount】接口，带出“建议计划量”字段（不可修改）；

如果（分销+自销）>建议量，则红字显示计划量，并alert提醒用户；

填写完成之后，点击【保存】按钮，返回[6.3.1.1 – 2]界面

如果（分销+自销）>建议量，则红字显示当前商品所在行；

* 1. 选择需要修改的记录，点击[6.3.1.1 – 2]界面【修改】按钮，弹出[6.3.1.1 – 3]界面，修改相关可以修改的字段；

填写完成之后，点击【保存】按钮，返回[6.3.1.1 – 2]界面

* 1. 选择需要删除的记录，点击[6.3.1.1 – 2]界面【删除】按钮，弹出confirm提示框，让用户确认，并删除明细列表中的数据；

以上操作完成之后，点击[6.3.1.1 – 2]界面【保存】按钮，携带商品列表信息返回[6.3.1.1 – 1]界面；

1. 点击【保存】按钮，将相关信息传输至服务器端；

调用【IDelivPlanService. insertPlan】存入计划信息；

循环调用【IDelivPlanService .insertPlanDetail】方法，存入计划明细信息；调用【IBillService. getBillNum】方法，获取主单据号；

调用【IBillService. recordBillInfo】方法存入单据信息，单据状态为“保存”

* 提交

1. 如果用户不【保存】，直接【提交】，则所有操作同“保存”；提交之后，单据状态为“待审核”；
2. 如果用户之前有【保存】过，则选择需要提交的记录，点击[6.3.1.1 – 1]界面【修改】按钮，弹出[6.3.1.1 – 2]界面；

调用【IDelivPlanService .getPlanDetailByPlanNum】相关接口，显示之前已经保存的商品明细；

[6.3.1.1 – 2]界面，【新增】【修改】【删除】操作同“保存”操作；

点击【提交】按钮，调用【IDelivPlanService. deletePlan】【IDelivPlanService.deletePlanDetailByPlanNum】接口，删除保存的信息，重新提交提货计划相关信息(同保存操作)，状态变更为“待审核”

1. 修改提货计划

* 初始化

同“新增”初始化

* 查询

同“新增”查询

* 查看

同“新增”查看

不同点：

点击【查看】弹出[6.3.1.1 – 4]界面，调用【单据历史查询】接口，展示单据历史信息；

* 提交

同“新增”提交功能，不同点如下：

1. 点击【提交】按钮，调用【IDelivPlanService.updatePlan】修改计划；
2. 无“业务规则”限制；

* 界面设计

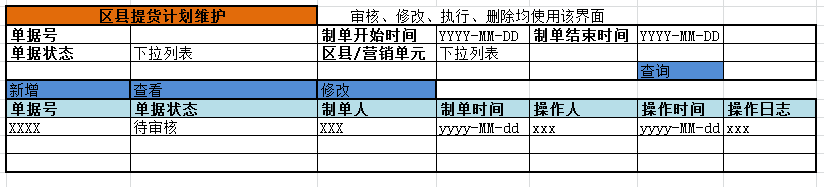


图6.3.1.1 – 1

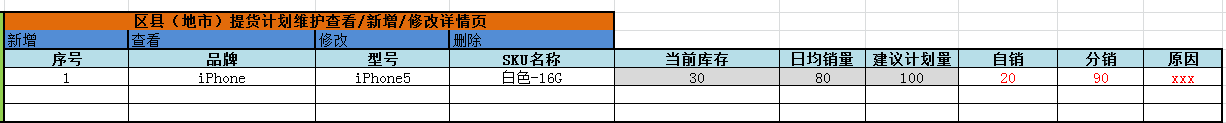


图6.3.1.1 – 2



图6.3.1.1 – 3

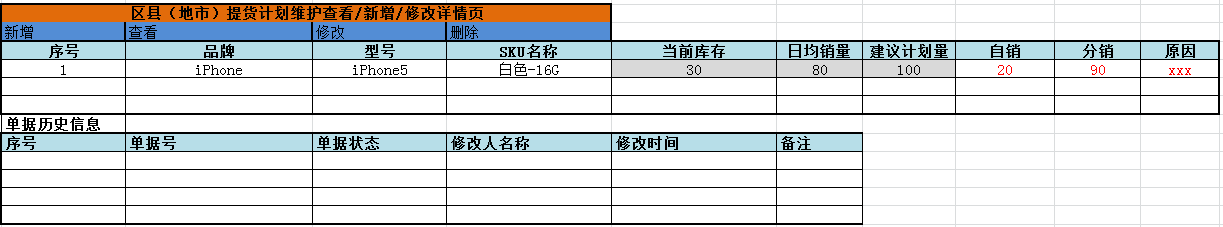


图6.3.1.1 – 4

#### 地市提货计划审核

* 处理流程
* 功能简述

地市审核提货计划

* 数据库表结构
* （提货计划单据A）TO\_DELIV\_PLAN，（提货计划明细）TO\_DELIV\_PLAN\_DETAIL
* （业务单据主表）TB\_BILL\_MAIN
* 详细设计
* 初始化

界面如图：[6.3.1.2 – 1]，初始化相关提货计划，默认查询条件如下：

**状态：**“待审核”

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

**地市：**显示当前用户的地市

* 查询

同“初始化”

* 查看

在[6.3.1.2 – 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，

弹出[6.3.1.2 – 2]界面，查看计划商品明细；如果某款商品的（自销+分销>建议计划量），则该商品所在行红色显示；点击【关闭】按钮关闭；

* 审核

在[6.3.1.2 – 1]界面中，选择需要审核的信息（单选），点击【审核】按钮，

弹出[6.3.1.2 – 2]界面，查看计划商品明细，调用【单据历史查询】接口，展示单据历史信息；点击【审核】按钮进行审核；

调用【IBillService. recordBillInfo】方法更新单据信息，单据状态为“待汇总”

* 界面设计

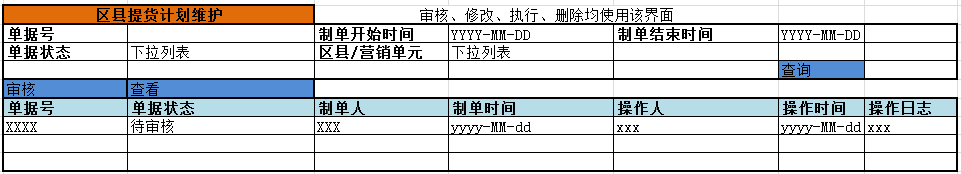


图6.3.1.2– 1



图6.3.1.2 – 2

#### 地市提货计划汇总

* 处理流程

1. 结束区县/营销单元/地市的流程（单据A），状态改为“已汇总”，单据A结束；
2. 生成地市提货计划汇总单（单据B），状态改为“待审核”

* 功能简述

地市汇总各区县/营销单元提货计划

* 数据库表结构
* （业务单据主表）TB\_BILL\_MAIN
* （提货计划单据A）TO\_DELIV\_PLAN，（提货计划明细）TO\_DELIV\_PLAN\_DETAIL
* （提货计划地市汇总表单据B）TO\_DELIV\_PLAN\_GATHER，

（提货计划汇总明细）TO\_DELIV\_PLAN\_GATHER\_DETAIL，

（提货计划汇总对照表）TO\_DZ\_DELIV\_PLAN\_GATHER

* 详细设计
* 初始化

界面如图[6.3.1.3 – 1]，初始化相关提货计划，默认查询条件如下：

**状态：**“待汇总”

**时间：**默认为当月

**地市：**显示当前用户的地市

**计划阶段：**调用地市参数查询接口，根据该地市配置的计划周期类型系统参数【DELIV\_PLAN\_CYCLE\_TYPE】，决定该字段的显示信息

页面显示加载：

1. 红色区域1，需显示目前地市已审核的该周期的计划单汇总数量；
2. 红色区域2，需显示目前地市，“未提交”、“待审核”、“审核未通过”的区县/营销单元；
3. 红色区域3，点击【筛选】按钮（区县/营销单元，可以多选），根据选择的条件进行汇总，逻辑同红色区域1，只是区县/营销单元范围不同

* 查询

点击【筛选】按钮，逻辑同“初始化”

* 查看

点击【查看】按钮，弹出[6.3.1.3 - 2]界面，表单部分展示商品相关信息，列表部分，展示计划里有该商品，并且计划审核通过的区县/营销单元列表；

点击【关闭】按钮，关闭页面；

* 汇总

**业务规则：**

1. 区域规则：

先判断该地市所有“区县/营销单元”是否都已经通过审核，如果通过审核，则直接提交；如果未通过审核，则confirm提醒用户『部分地市暂未提交或未通过审核』；

1. 时间规则：

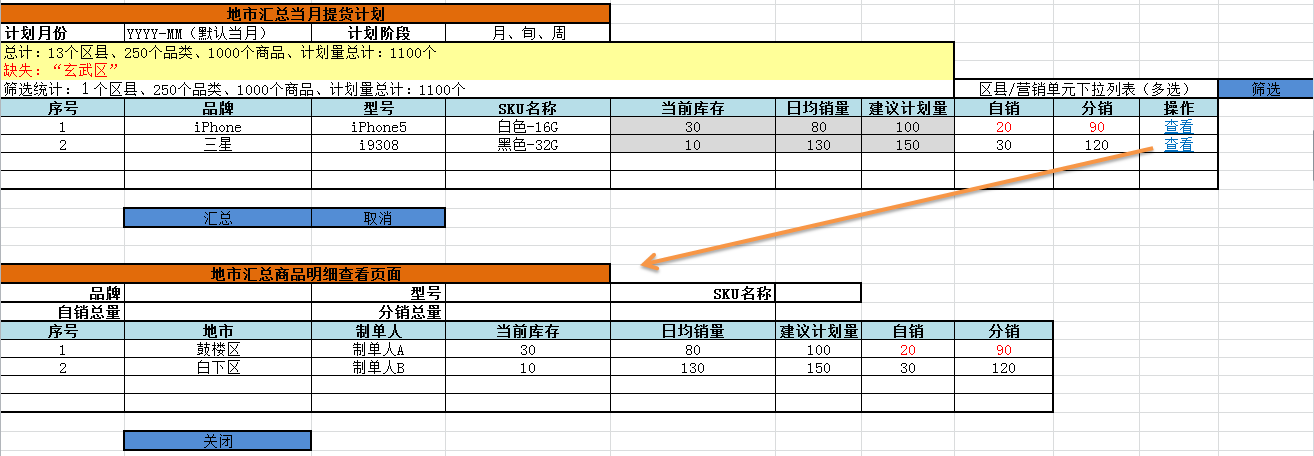
如果所有“区县/营销单元”都已审核通过，则在任意时间均可汇总提交；

如果部分“区县/营销单元”未审核通过，在当前周期最晚时间过后才可以提交；否则，继续等待未通过审核的“区县/营销单元”；

**业务操作：**

1. [6.3.1.3 - 1]界面点击【汇总】按钮，调用【IDelivPlanService.validAllowGather】方法，检验业务规则，判断用户是否可以汇总提交；
2. 如果业务规则通过，则提交前端数据至服务器端，并作以下处理：
   1. 调用【IBillService. recordBillInfo】方法更新（单据A）信息，单据状态为“已汇总”；
   2. 调用【IBillService. getBillNum】方法，获取（单据B）主单据号；
   3. 调用【IBillService. recordBillInfo】方法存入单据（单据B）信息，单据状态为“待审核”；
   4. 调用【IDelivPlanService.insertPlanGather】方法，新增汇总信息；
   5. 循环调用【IDelivPlanService.insertPlanGatherDetail】方法，新增汇总商品明细信息；
   6. 循环调用【IDelivPlanService.insertPlanGatherDz】方法，新增汇总-提货计划对照信息；

* 界面设计



上图6.3.1.3 – 1

下图6.3.1.3 - 2

#### 地市提货计划制定

* 处理流程

1. 保存>>提交；导入>>提交
2. 生成地市提货计划，状态直接改为“已汇总”，单据A结束；
3. 生成地市提货计划汇总单（单据B），状态改为“待审核”

* 功能简述

地市制定提货计划（只能保存、提交，新增提交后不能修改；修改功能见6.1.3.6功能）

* 数据库表结构
  + （业务单据主表）TB\_BILL\_MAIN
  + （提货计划单据A）TO\_DELIV\_PLAN，（提货计划明细）TO\_DELIV\_PLAN\_DETAIL
  + （提货计划地市汇总表单据B）TO\_DELIV\_PLAN\_GATHER，

（提货计划汇总明细）TO\_DELIV\_PLAN\_GATHER\_DETAIL，

（提货计划汇总对照表）TO\_DZ\_DELIV\_PLAN\_GATHER

* 详细设计

参考区县提货计划维护、地市汇总提货计划

**不同点：**

单据A的过程参考“区县提货计划维护”，逻辑一致；

单据B的过程参考“地市汇总提货计划”，逻辑一致；

**打开页面时，判断当前用户所在地市是否是“区县可以提交提货计划”，如果区县能提交，则地市不能提交；则『新增』按钮disabled，无法做『新增』功能；**

**判断条件：（T\_REGION\_SYSTEM表DELIV\_PLAN\_AREA\_TYPE字段）**

* 界面设计

参考区县提货计划维护

#### 省提货计划审核

* 处理流程

1. 审核通过，状态改为“已汇总”，单据B结束；该地市提货计划量合并到该地市的月度计划总量中
2. 审核不通过，状态改为“审核不通过”，地市管理员修改后重新提交

* 功能简述

省审核地市提货计划汇总单

* 数据库表结构
* （业务单据主表）TB\_BILL\_MAIN
* （提货计划地市汇总表单据B）TO\_DELIV\_PLAN\_GATHER，

（提货计划汇总明细）TO\_DELIV\_PLAN\_GATHER\_DETAIL，

（提货计划汇总对照表）TO\_DZ\_DELIV\_PLAN\_GATHER

* （地市提货计划量）TO\_REGIN\_DELIV\_PLAN
* 详细设计
* 初始化

界面如图：[6.3.1.5 – 1]，初始化相关提货计划汇总，默认查询条件如下：

**状态：**“待审核”

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

**地市：**默认全部地市

* 查询

同“初始化”

* 查看

在[6.3.1.5 – 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，

弹出[6.3.1.5 – 2]界面，查看计划商品明细；如果某款商品的（自销+分销>建议计划量），则该商品所在行红色显示；点击【关闭】按钮关闭；

* 审核

在[6.3.1.5 – 1]界面中，选择需要审核的信息（单选），点击【审核】按钮，

弹出[6.3.1.5 – 2]界面，查看计划商品明细，查询该单据处理日志信息；点击【审核】按钮进行审核；

1. 调用【IBillService. recordBillInfo】方法更新单据信息，单据状态为“已汇总”；
2. 调用【IBillService.mergeReginDelivPlanCount】方法，新增或修改该地市的提货月计划量；

* 界面设计



上图 6.3.1.5 – 1

下图 6.3.1.5 - 2

#### 地市汇总计划修改

* 处理流程
* 功能简述

省审核地市汇总计划不通过，则地市修改计划总量，重新提交；

* 数据库表结构
* （业务单据主表）TB\_BILL\_MAIN
* （提货计划地市汇总表单据B）TO\_DELIV\_PLAN\_GATHER，

（提货计划汇总明细）TO\_DELIV\_PLAN\_GATHER\_DETAIL

* 详细设计
* 初始化

界面如图：[6.3.1.6 – 1]，初始化相关提货计划，默认查询条件如下：

**状态：**“审核不通过”

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

* 查询

同“初始化”

* 查看

在[6.3.1.6 – 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，弹出[6.3.1.6 – 2]界面，查看计划商品明细；点击【关闭】按钮关闭；

* 修改

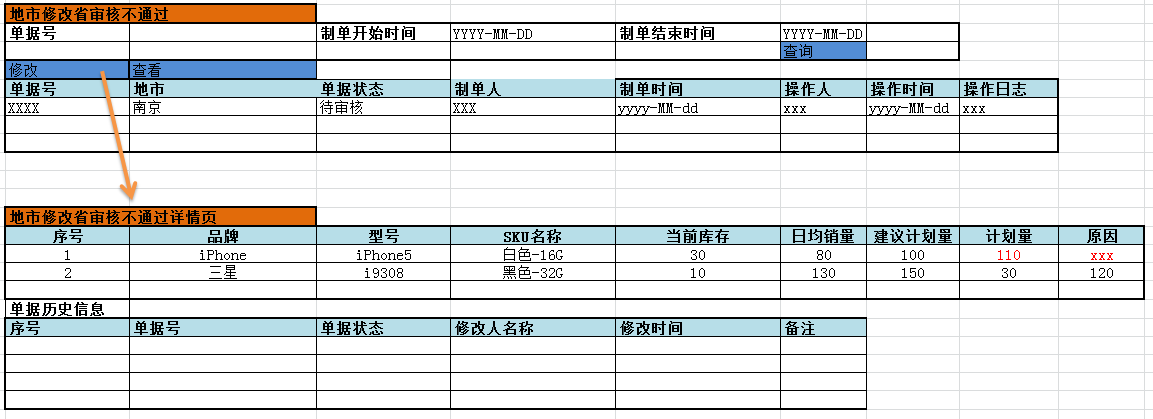
在[6.3.1.6 – 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，弹出[6.3.1.6 – 2]界面，查看计划商品明细；调用【单据历史查询】接口，展示单据历史信息；

双击[6.3.1.6 – 2]界面表格，修改该行数据，只允许修改“计划量”字段；

1. 调用【IBillService. recordBillInfo】方法存入单据信息，单据状态为“待审核”
2. 调用【IDelivPlanService.updatePlanGather】修改地市计划汇总单；

注意：只修改TO\_DELIV\_PLAN\_GATHER\_DETAIL，F\_REAL\_COUNT字段

* 界面设计



上图 6.3.1.6 – 1

下图 6.3.1.7 - 2

#### 提货计划历史查询

* 处理流程
* 功能简述

单据A的历史信息查询

查询提货计划历史信息；

* + 省可以查询所有地市提货计划历史
  + 地市可以查询该地市，以及该地市下属区县/营销单元计划历史
  + 区县/营销单元只能查询该机构的计划历史
* 数据库表结构
  + （提货计划单据A）TO\_DELIV\_PLAN，（提货计划明细）TO\_DELIV\_PLAN\_DETAIL
  + （业务单据主表）TB\_BILL\_MAIN
* 详细设计
  + 初始化

界面如图：[6.3.1.7 – 1]，初始化相关提货计划，默认查询条件如下：

**状态：**全部

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

**其余条件默认为空**

* + 查询

同“初始化”

* + 查看

在[6.3.1.7 – 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，弹出[6.3.1.7 – 2]界面，查看计划商品明细；点击【关闭】按钮关闭；

* 界面设计

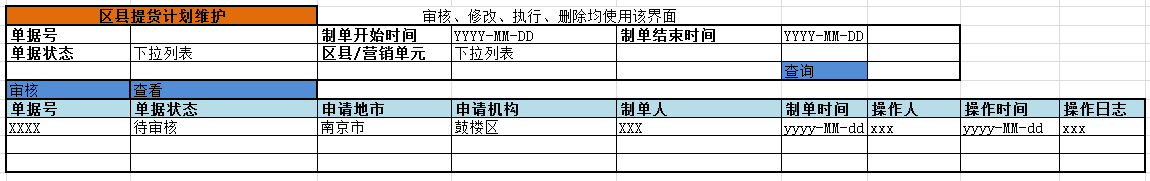


图 6.3.1.7 – 1

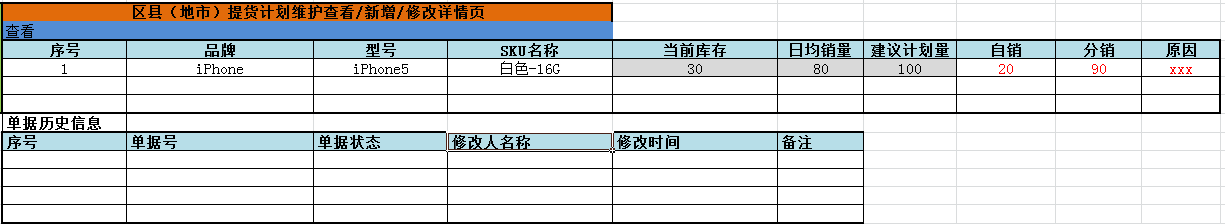


图 6.3.1.7 – 2

#### 提货计划汇总历史查询

* 处理流程
* 功能简述

单据B的历史信息查询

查询提货计划汇总历史信息；

* + 省可以查询所有地市提货计划汇总历史
  + 地市可以查询该地市提货计划汇总历史
* 数据库表结构
  + （提货计划单据B）TO\_BILL\_DELIV\_GATHER，（提货计划明细）TO\_BILL\_DELIV\_GATHER\_DETAIL
  + （业务单据主表）TB\_BILL\_MAIN
* 详细设计
  + 初始化

界面如图：[6.3.1.7 – 1]，初始化相关提货计划，默认查询条件如下：

**状态：**全部

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

**其余条件默认为空**

* + 查询

同“初始化”

* + 查看

在[6.3.1.7 – 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，弹出[6.3.1.7 – 2]界面，查看计划商品明细；调用【单据历史查询】接口，展示单据历史信息；点击【关闭】按钮关闭；

* 界面设计

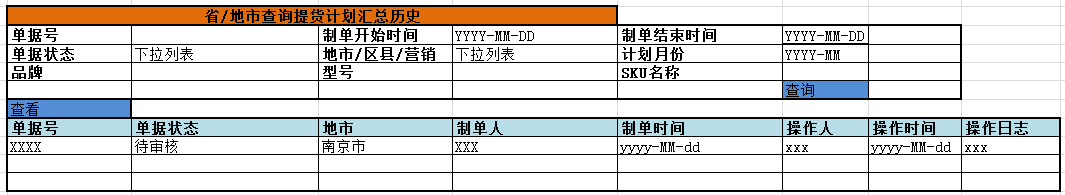


图 6.3.1.7 – 1

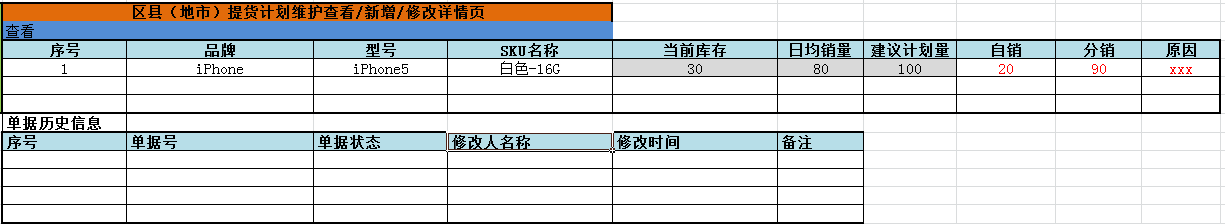


图 6.3.1.7 – 2

#### 提货（增补）计划汇总查询

* 处理流程
* 功能简述

查询“普通提货计划”或“增补提货计划”的统计数量；

* 数据库表结构

（地市提货计划量）TO\_REGIN\_DELIV\_PLAN

* 详细设计
  + 初始化

**地市：**全部地市

**商品相关信息：**空

**时间：**计划开始月份，当月；计划结束月份，当月；

**计划类型：**普通提货计划

* + 查询

同“初始化”

点击【查询】按钮，调用【IDelivPlanService.queryDelivPlanCount】方法，获取需要统计的地市的月计划量

* 界面设计

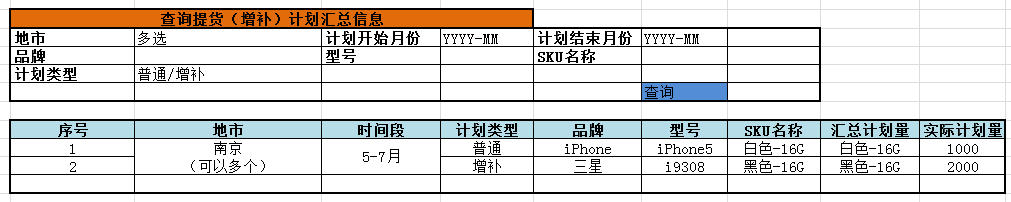


图 6.1.9.1 - 1

#### 提货计划参数设置

* 处理流程
* 功能简述

此部分信息均已完成初始化脚本，不用开发，详见初始化数据.sql；

* + 初始化：省统一计划周期参数
  + 初始化：省统一提货计划周期时间区间
  + 初始化：提货计划超过计划量原因
  + 初始化：各地市是否允许区县/营销单元提交提货计划
  + 初始化：是否启用一次提交多周期计划
* 数据库表结构
  + （系统参数表）T\_SYSTEM
  + （地市参数表）T\_REGION\_SYSTEM，（地市参数对照表）T\_REGION\_SYSTEM\_DZ
* 详细设计
  + 初始化：省统一计划周期参数

数据表：T \_SYSTEM，参数编码：DELIV\_PLAN\_CYCLE\_TYPE

* + 初始化：省统一提货计划周期时间区间

数据表：T \_SYSTEM，参数编码：待定

* + 初始化：提货计划超过计划量原因

数据表：TB\_DA\_REASON\_ITEM，参数编码：THJHCL

* + 初始化：各地市是否允许区县/营销单元提交提货计划

数据表：T\_REGION\_SYSTEM，参数编码：DELIV\_PLAN\_AREA\_TYPE

* + 初始化：是否启用一次提交多周期计划

数据表：T \_SYSTEM，参数编码：IS\_ALLOW\_MULTI\_CYCLE\_PLAN

* 界面设计

### 提货增补计划管理

* 处理流程



* 业务规则

提货增补计划页面，选择商品时，商品组件查询商品需关联“要货关系控制”进行查询，过滤不属于该地市的商品；

#### 增补计划维护

* 处理流程
  + 保存>>提交
  + 导入>>提交
* 功能简述

1. 新增增补计划
2. 修改“审核不通过”，“待审核”增补计划

* 数据库表结构
* （提货计划单据C）TO\_DELIV\_PLAN，（提货计划明细）TO\_DELIV\_PLAN\_DETAIL
* （业务单据主表）TB\_BILL\_MAIN
* 详细设计

1. 新增增补计划

* 初始化

界面如图：[6.3.2.1 – 1]，初始化相关提货计划，默认查询条件如下：

**状态：**“待审核”，“审核不通过”，“保存”

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

**地市：**当前用户所在地市

* 查询

同“初始化”

* 查看

在[6.3.2.1 – 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，弹出[6.3.2.1 – 2]界面，查看计划商品明细；点击【关闭】按钮关闭；

* 保存（临时操作）

业务操作：

1. 点击[6.3.2.1 – 1]界面【新增】按钮，弹出[6.3.2.1 – 2]界面，此界面为商品粒度的提货计划明细；
2. 商品明细列表维护
   1. 点击[6.3.1.1 – 2]界面【新增】按钮，弹出[6.3.1.1 – 3]界面，使用“商品选择组件”进行商品选择，选择完成之后，带出“当前库存”，“上月销量”两个字段（不可修改）；调用【IDelivPlanService.getReferPlanCount】接口，带出“建议计划量”字段（不可修改）；

填写完成之后，点击【保存】按钮，返回[6.3.1.1 – 2]界面

* 1. 选择需要修改的记录，点击[6.3.2.1 – 2]界面【修改】按钮，弹出[6.3.2.1 – 3]界面，修改相关可以修改的字段；

填写完成之后，点击【保存】按钮，返回[6.3.2.1 – 2]界面

* 1. 选择需要删除的记录，点击[6.3.2.1 – 2]界面【删除】按钮，弹出confirm提示框，让用户确认，并删除明细列表中的数据；

以上操作完成之后，点击[6.3.2.1 – 2]界面【保存】按钮，携带商品列表信息返回[6.3.2.1 – 1]界面；

1. 点击【保存】按钮，将相关信息传输至服务器端；

调用【IDelivPlanService. insertPlan】存入计划信息；

循环调用【IDelivPlanService .insertPlanDetail】方法，存入计划明细信息；调用【IBillService. getBillNum】方法，获取主单据号；

调用【IBillService. recordBillInfo】方法存入单据信息，单据状态为“保存”

* 提交

1. 如果用户不【保存】，直接【提交】，则所有操作同“保存”；提交之后，单据状态为“待审核”；
2. 如果用户之前有【保存】过，则选择需要提交的记录，点击[6.3.1.1 – 1]界面【修改】按钮，弹出[6.3.2.1 – 2]界面；

调用【IDelivPlanService .getPlanDetailByPlanNum】相关接口，显示之前已经保存的商品明细；

[6.3.2.1 – 2]界面，【新增】【修改】【删除】操作同“保存”操作；

点击【提交】按钮，调用【IDelivPlanService. deletePlan】【IDelivPlanService.deletePlanDetailByPlanNum】接口，删除保存的信息，重新提交增补计划相关信息(同保存操作)，状态变更为“待审核”

1. 修改增补计划

* 初始化

同“新增”初始化

* 查询

同“新增”查询

* 查看

同“新增”查看

不同点：

点击【查看】弹出[6.3.2.1 – 4]界面，调用【单据历史查询】接口，展示单据历史信息；

* 提交

同“新增”提交功能，不同点如下：

1. 点击【提交】按钮，调用【IDelivPlanService.updatePlan】修改计划；

* 界面设计



图 6.3.2.1 – 1



图 6.3.2.1 – 2



图 6.3.2.1 – 3



图 6.3.2.1 – 4

#### 增补计划审核

* 处理流程
  + 审核通过该地市增补计划量合并到该地市的月度增补计划总量中
  + 审核不通过，状态改为“审核不通过”，地市管理员修改后重新提交
* 功能简述

省审核地市增补计划申请

* 数据库表结构
  + （提货计划单据C）TO\_DELIV\_PLAN，（提货计划明细）TO\_DELIV\_PLAN\_DETAIL
  + （业务单据主表）TB\_BILL\_MAIN
  + （地市提货计划量）TO\_REGIN\_DELIV\_PLAN
* 详细设计
* 初始化

界面如图：[6.3.2.2 – 1]，初始化相关提货计划汇总，默认查询条件如下：

**状态：**“待审核”

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

**地市：**默认全部地市

* 查询

同“初始化”

* 查看

在[6.3.2.2– 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，

弹出[6.3.2.2 – 2]界面，查看计划商品明细；点击【关闭】按钮关闭；

* 审核

在[6.3.2.2 – 1]界面中，选择需要审核的信息（单选），点击【审核】按钮，

弹出[6.3.2.2 – 2]界面，查看计划商品明细，查询该单据处理日志信息；点击【审核】按钮进行审核；

1. 调用【IBillService. recordBillInfo】方法更新单据信息，单据状态为“已汇总”；
2. 调用【IDelivPlanService.mergeReginDelivPlanCount】方法，新增或修改该地市的增补月计划量；

* 界面设计



图 6.3.2.2 – 1



图 6.3.2.2 – 2

#### 增补计划历史查询

* 处理流程
* 功能简述

单据C的历史信息查询

查询增补计划历史信息；

* + 省可以查询所有地市增补计划历史
  + 地市可以查询该地市增补计划历史
* 数据库表结构
  + （提货计划单据C）TO\_DELIV\_PLAN，（提货计划明细）TO\_DELIV\_PLAN\_DETAIL
  + （业务单据主表）TB\_BILL\_MAIN
* 详细设计
  + 初始化

界面如图：[6.3.2.3 – 1]，初始化相关提货计划，默认查询条件如下：

**状态：**全部

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

**地市：**省-全部，地市-自己地市；

* + 查询

同“初始化”

* + 查看

在[6.3.2.3 – 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，弹出[6.3.2.3 – 2]界面，查看计划商品明细；点击【关闭】按钮关闭；

* 界面设计

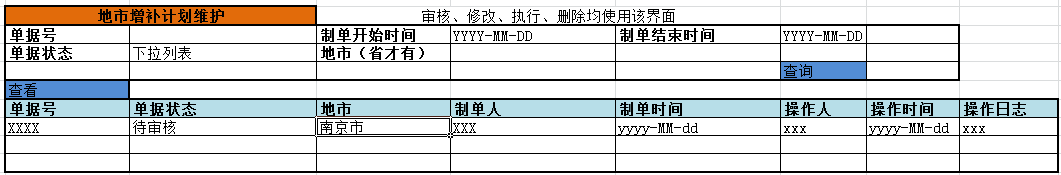


图 6.3.2.3 – 1



图 6.3.2.3 – 2

### 集采计划

* 处理流程
* 业务规则

集采计划页面，选择商品时，商品组件查询商品需关联“供货关系控制”进行查询；

#### 集采计划维护

* 处理流程

保存>>提交 ;

导入>>保存>>提交;

修改>>提交;

* 功能简述
  + 保存集采计划
  + 导入集采计划
  + 提交集采计划
  + 复制提交集采计划
  + （根据同周期提货计划）生成集采计划
  + 修改集采计划
* 数据库表结构
  + （集采计划单明细）TO\_BILL\_PURCH\_PLAN\_DETAIL
  + （业务单据主表）TB\_BILL\_MAIN
* 详细设计

1. 新增集采计划

* 初始化

界面如图：[6.3.3.1 – 1]，初始化集采计划，默认查询条件如下：

**状态：**“待审核”，“审核不通过”，“保存”

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

* 查询

同“初始化”

* 查看

在[6.3.3.1 – 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，弹出[6.3.3.1 – 2]界面，查看计划商品列表；点击【关闭】按钮关闭；

在[6.3.3.1 – 2]界面，选择需要查看的商品明细（单选），点击【查看】按钮，弹出[6.3.3.1 – 3]界面，查看计划商品明细；点击【关闭】按钮关闭；

* 保存（临时操作）

业务操作：以下3种方式只是录入方式不同，后续数据/逻辑操作一致

* 生成集采计划【生成】

1. 点击[6.3.3.1 – 1]界面【生成】按钮，弹出[6.3.2.1 – 2]界面，此界面为商品粒度的提货计划明细；
2. 弹出[6.3.2.1 – 2]界面同时，调用【IDelivPlanService. getDelivPlanCount】获取该周期的提货计划信息，展示在[6.3.2.1 – 2]界面的商品列表表格中；
3. 必须【修改】每一条商品数据，录入商品的集采计划量；点击【修改】按钮时，弹出[6.3.2.1 – 3]界面，根据商品编码调用【IPurchPlanService. getReferPlanCount】方法，获取界面中参考的相关字段；

填写完成之后，点击【保存】按钮，返回[6.3.1.1 – 2]界面

* 复制集采计划【复制】

1. 在[6.3.3.1 – 1]界面选择需要复制的记录，点击[6.3.3.1 – 1]界面【复制】按钮，弹出[6.3.2.1 – 2]界面，此界面为商品粒度的提货计划明细；
2. 弹出[6.3.2.1 – 2]界面同时，调用【IPurchPlanService.getPlanDetailByPlanNum】获取集采计划明细信息，展示在[6.3.2.1 – 2]界面的商品列表表格中；
3. 必须【修改】每一条商品数据，录入商品的集采计划量；点击【修改】按钮时，弹出[6.3.2.1 – 3]界面，根据商品编码调用【IPurchPlanService. getReferPlanCount】方法，获取界面中参考的相关字段；

填写完成之后，点击【保存】按钮，返回[6.3.1.1 – 2]界面

* 新增集采计划【新增】

1. 点击[6.3.3.1 – 1]界面【新增】按钮，弹出[6.3.3.1 – 2]界面，此界面为商品粒度的提货计划明细；
2. 商品明细列表维护
   1. 点击[6.3.1.1 – 2]界面【新增】按钮，弹出[6.3.1.1 – 3]界面，使用“商品选择组件”进行商品选择，选择完成之后，根据商品编码调用【IPurchPlanService. getReferPlanCount】方法，获取界面中参考的相关字段；

填写完成之后，点击【保存】按钮，返回[6.3.3.1 – 2]界面

* 1. 选择需要修改的记录，点击[6.3.2.1 – 2]界面【修改】按钮，弹出[6.3.2.1 – 3]界面，修改相关可以修改的字段；

如果该商品的“参考相关字段”已有，则直接展示，如果没有，在修改弹出页面时，需从后台获取该部分信息；

填写完成之后，点击【保存】按钮，返回[6.3.3.1 – 2]界面

* 1. 选择需要删除的记录，点击[6.3.2.1 – 2]界面【删除】按钮，弹出confirm提示框，让用户确认，并删除明细列表中的数据；

以上操作完成之后，点击[6.3.3.1 – 2]界面【保存】按钮，携带商品列表信息返回[6.3.3.1 – 1]界面；

1. 点击【保存】按钮，将相关信息传输至服务器端；

循环调用【IPurchPlanService.insertPlanDetail】方法，存入计划明细信息；调用【IBillService. getBillNum】方法，获取主单据号；

调用【IBillService. recordBillInfo】方法存入单据信息，单据状态为“保存”

* 提交

1. 如果用户不【保存】，直接【提交】，则所有操作同“保存”；提交之后，单据状态为“待审核”；
2. 如果用户之前有【保存】过，则选择需要提交的记录，点击[6.3.3.1 – 1]界面【修改】按钮，弹出[6.3.2.1 – 2]界面；

调用【IPurchPlanService.getPlanDetailByPlanNum】相关接口，显示之前已经保存的商品明细；

[6.3.2.1 – 2]界面，【新增】【修改】【删除】操作同“保存”操作；

点击【提交】按钮，调用【IPurchPlanService. deletePlan】【IPurchPlanService.deletePlanDetailByPlanNum】接口，删除保存的信息，重新提交增补计划相关信息(同保存操作)，状态变更为“待审核”

1. 修改集采计划

* 初始化

同“新增”初始化

* 查询

同“新增”查询

* 查看

同“新增”查看

不同点：

点击【查看】弹出[6.3.3.1 – 2]界面，调用【单据历史查询】接口，展示单据历史信息；

* 提交

同“新增”提交功能，不同点如下：

1. 点击【提交】按钮，不删除原单据信息；但是删除商品明细；

* 界面设计

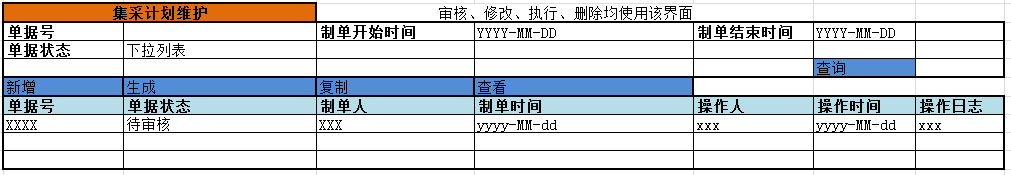


图6.3.3.1 – 1



图 6.3.3.1 – 2



图 6.3.3.1 - 3

#### 集采计划审核

* 处理流程
* 功能简述
  + 审核集采计划
* 数据库表结构
  + （集采计划单明细）TO\_BILL\_PURCH\_PLAN\_DETAIL
  + （业务单据主表）TB\_BILL\_MAIN
* 详细设计
* 初始化

界面如图：[6.3.3.2 – 1]，初始化待审核集采计划，默认查询条件如下：

**状态：**“待审核”

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

* 查询

同“初始化”

* 查看

在[6.3.3.2– 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，

弹出[6.3.3.2 – 2]界面，查看计划商品明细；点击【关闭】按钮关闭；

**选择商品记录，点击【查看】按钮，弹出[6.3.3.2 – 3]界面，其中以下字段需重新查询（由于该部分字段时刻变化）**

****

****

****

**剩余字段均从该条商品的单据历史信息获取；**

* 审核

在[6.3.3.2 – 1]界面中，选择需要审核的信息（单选），点击【审核】按钮，

弹出[6.3.3.2 – 2]界面，查看计划商品明细，查询该单据处理日志信息；点击【审核】按钮进行审核；

**选择商品记录，点击【查看】按钮，弹出[6.3.3.2 – 3]界面，其中以下字段需重新查询（由于该部分字段时刻变化）**

****

****

****

**剩余字段均从该条商品的单据历史信息获取；**

1. 调用【IBillService. recordBillInfo】方法更新单据信息，单据状态为“已汇总”；
2. 审核通过后，调用【IPurchPlanService.mergePurchPlanCount】方法，合并集采计划量；
3. 审核不通过，调用【IBillService. recordBillInfo】方法更新单据信息，单据状态为“审核不通过”；

* 界面设计

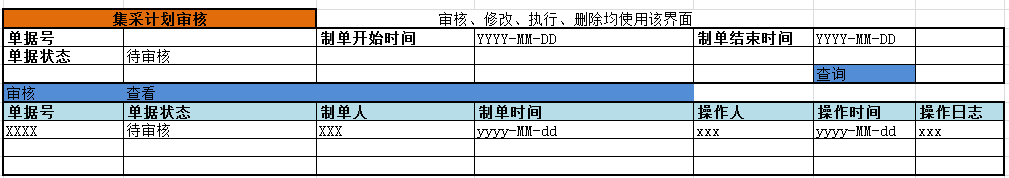


图6.3.3.2 – 1



图 6.3.3.2 – 2



图 6.3.3.2 - 3

#### 集采计划历史查询

* 处理流程
* 功能简述
  + 查询集采计划历史
* 数据库表结构
  + （集采计划单明细）TO\_BILL\_PURCH\_PLAN\_DETAIL
  + （业务单据主表）TB\_BILL\_MAIN
* 详细设计
  + 初始化

界面如图：[6.3.3.3 – 1]，初始化相关提货计划，默认查询条件如下：

**状态：**全部

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

**其余条件默认为空**

调用【IPurchPlanService . queryPurchPlanHistory】方法，进行查询；

* + 查询

同“初始化”

* + 查看

在[6.3.3.3 – 1]界面中，选择需要查看的信息（单选），点击【查看】按钮，弹出[6.3.3.3 – 2]界面，查看计划商品明细；点击【关闭】按钮关闭；

点击[6.3.3.3 – 2]界面【查看】按钮，弹出[6.3.3.3 – 3]界面，显示相关信息；

* 界面设计

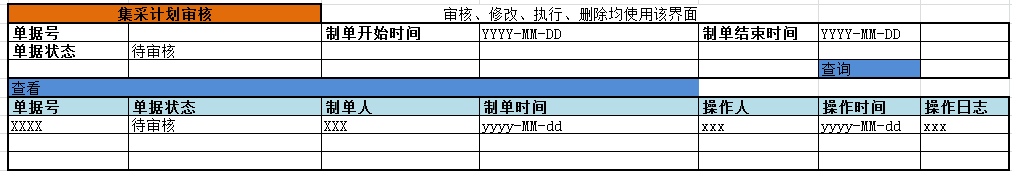


图6.3.3.3 – 1



图 6.3.3.3 – 2



图 6.3.3.3 - 3

#### 集采计划汇总查询

* 处理流程
* 功能简述
  + 查询集采计划汇总
* 数据库表结构
  + （集采计划汇总）TO\_PURCH\_PLAN\_GATHER
* 详细设计
  + 初始化

界面如图：[6.3.3.4 – 1]，初始化集采计划汇总，默认查询条件如下：

**状态：**全部

**时间：**默认为当月，制单开始时间=当前月初，制单结束时间=当前月末

**其余条件默认为空**

调用【IPurchPlanService .queryPurchPlanCount】方法，进行查询；

* + 查询

同“初始化”

* 界面设计

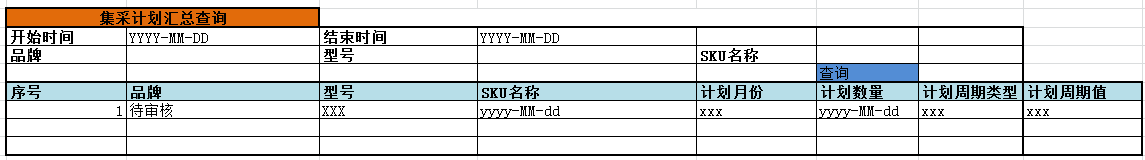


图 6.3.3.4 – 1

### 要货策略管理

#### 波次策略管理

##### 波次策略配置

* 处理流程

无

* 功能简述

1. 波次地市配置参数，配置每个地市是否启动波次。
2. 初始化地市参数。

地市参数表（T\_REGION\_SYSTEM）。

地市参数值对照表（T\_REGION\_SYSTEM\_DZ）。

1. 点击功能菜单后，打开【波次策略配置页面】（图 -1）。
2. 用户选择地市。

* 查询地市参数表，获取该地市是否启用地市参数。
* 查询地市参数获得该地市配置的波次数量。
* 查询【波次定义表】（TO\_DA\_WAVE），获得该地市实际的波次数量。
* 比较配置数量与实际数量是否相等。
* 相等则按实际的波次将信息显示在界面中。
* 不相等则按地市配置的数量显示，并将原配置中的信息删除（若实际数量=0则不用删除）。
* 波次名称生成规则：第一波次、第二波次递增；波次序号按数量从1开始递增，营业厅级别显示所有级别，其他字段均空，等待用户输入。
* 每个波次勾选一个低级营业厅级别，则将之前的所有比当前级别高的营业厅级别都够上。

1. 界面输入有效性验证。

* 每个波次的【开始时间】必须等于上一个波次的【结束时间】。
* 所有波次的【本次要货量】之和必须是100%。
* 累计要货量不可输入，系统自动计算。本波次【累计要货量】等于之前所有波次的【本次要货量】之和。
* 若实际波次已经配置过，则在点击保存的时候只能在该地市已经配置波次时间之外修改波次信息。
* 图 -1的波次配置信息如果有改动，必须点击保存以后才能点击【特殊商品】跳转到图 -2 进行特殊商品要货量的配置。
* 【本波次要货量】界面输入1-100整数，%系统自动添加。存入数据库记录0.00小数。

1. 【波次策略配置页面】（图 -1）点击【保存】按钮，将波次信息保存数据库。
2. 将该地市所有波次信息、营业厅级别信息删除。
3. 将界面上的波次信息记录数据库，循环记录多个波次。
4. 根据波次号，将对应波次的营业厅级别信息记录数据库，循环记录。

1),2),3)步骤在同一个事务中执行。

1. 用户点击【特殊商品】的超链接，跳转【波次策略商品配置页面】（图 -2）。
2. 【特殊商品】，根据地市查询出该地市是否有配置商品，没有配置显示“未配置”，已经配置显示具体的数量。
3. 点击【特殊商品】的超链接时，需校验波次信息是否有修改，有则提示“请先保存波次信息”。
4. 【波次策略商品配置页面】（图 -2），只有【本次要货量】可以修改，其他字段均灰化显示，不可修改。如果有特殊商品，则将特殊商品的信息显示在列表中。
5. 用户选择品牌、型号、商品SKU名称并输入对应波次的值后，点击【新增】，将该商品的信息添加到【特殊商品】列表中。

* 只有统一运营的商品才显示。（查询统一供货视图VIEW\_SUPP\_GOODS\_UNIT）
* 商品不能重复配置。
* 所有波次的【本次要货量】之和必须是100%。

1. 用户点击【删除】，则删除该商品对应的所有波次的信息。
2. 【波次策略商品配置页面】（图 -2）点击【保存】按钮，保存商品配置信息。
3. 将该地市所有商品配置信息删除。
4. 根据波次号，将商品信息记录数据库，循环记录。

1),2)步骤在同一个事务中执行。

* 数据库表结构

波次定义表（TO\_DA\_WAVE）

波次和营业厅对照表（TO\_DZ\_WAVE\_CHANNEL）

波次商品分配比例（TO\_GOODS\_WAVE）

* 详细设计

无

* 界面设计



图 -1



图 -2

##### 波次策略查询

* 处理流程

无

* 功能简述

1. 点击功能菜单后，打开【波次策略查询页面】（图 -1）。
2. 默认查询出所有商品的列表，操作时间倒序排序。
3. 省工号登陆显示全部，地市工号登陆显示本地市。
4. 用户点击【查询】按钮，根据查询条件将符合条件的单据信息查询出来并在列表中展现。
5. 查询条件说明：

* 地市：省工号显示（全省/13地市），地市工号显示本地市（不可修改）。
* 营业厅级别：下拉列表（全部/1级/2级/3级），选择全部，将所有级别的结果都查出来。

1. 查询结果时间倒序排序。
2. 【波次策略查询页面】（图 -1）点击【商品配置查询】按钮，跳转到【波次策略商品明细查询页面】（图-2）。
3. 调用【波次策略商品详情查询】，查询商品详情。

* 数据库表结构

波次定义表（TO\_DA\_WAVE）

波次和营业厅对照表（TO\_DZ\_WAVE\_CHANNEL）

波次商品分配比例（TO\_GOODS\_WAVE）

* 详细设计

无

* 界面设计



图 -1



图 -2

#### 审核策略管理

##### 审核策略配置

* 处理流程

无

* 功能简述

1. 点击功能菜单后，打开【审核策略配置页面】（图 -1）。
2. 地市默认登陆工号地市，省工号显示所有地市。
3. 用户选择二级菜单（策略类型名称），双击配置全地市策略，如：图 -1。
4. 用户选择三级菜单（营业厅级别），双击配置该营业厅级别策略，如：图 -2。
5. 地市名称、策略类型名称、生效范围灰化显示，不可修改。
6. 开始时间、结束时间提供日期控件，记录数据库精确到秒。结束时间可不填。
7. 选择营业厅级别时，生效范围显示选中的级别，如果勾选复制配置，则对选中的营业厅级别同样生效。
8. 审核级别提供下拉列表。
9. 是否生效不显示，默认生效。
10. 库存最大保有天数策略，商品列表不显示（不需要配置），阀值在主表字段中配置。
11. 品类列表默认显示所有品类。
12. 只显示统一运营的品类。（查询统一供货视图VIEW\_SUPP\_GOODS\_UNIT，根据视图中的商品将对应的品类查询出来）
13. 选中某个品牌，则只显示该品牌下的品类。
14. 是否设置，可根据特殊和未配置的品类显示。
15. 【审核策略配置页面】（图 -1）点击【保存】按钮，将用户输入的信息保存到数据库。
16. 调用【保存审核策略方法】，将策略信息插入数据库。
17. 保存时，先保存审核策略配置信息，然后在保存商品对照信息。
18. 将该地市所有审核策略信息、品类信息删除。
19. 将界面上的审核策略信息记录数据库，如果有复制信息则循环记录多个配置。
20. 根据审核策略号，将对应品类信息记录数据库，循环记录。

3),4)5)步骤在同一个事务中执行。

* 数据库表结构

审核策略配置表（TO\_REVIEW\_CONFIG）

策略商品对照表（TO\_DZ\_REVIEW\_GOODS）

* 详细设计

无

* 界面设计

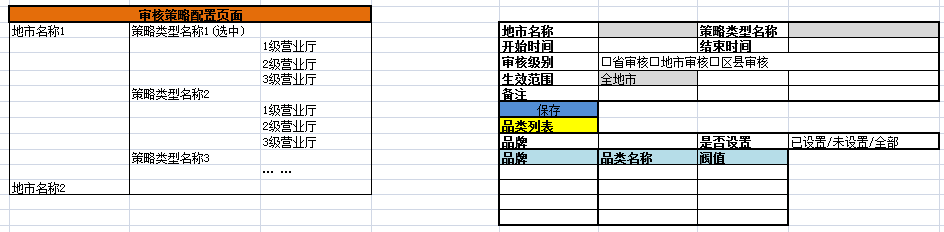


图 -1

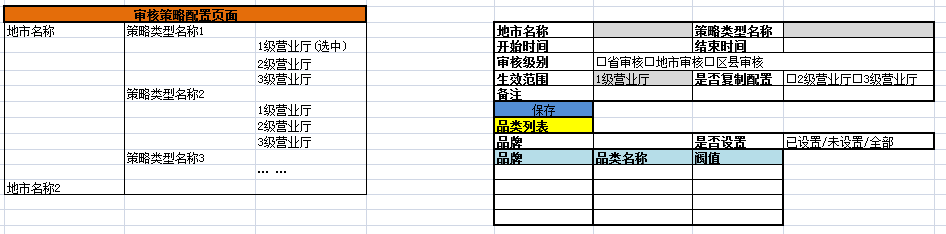


图 -2

##### 审核策略查询

* 处理流程

无

* 功能简述

1. 点击功能菜单后，打开【审核策略查询页面】（图 -1）。
2. 默认查询出所有商品的列表，操作时间倒序排序。
3. 省工号登陆显示全部，地市工号登陆显示本地市不可修改。
4. 用户点击【查询】按钮，根据查询条件将符合条件的单据信息查询出来并在列表中展现。
5. 查询条件说明：

* 品牌、品类，后台模糊查询
* 地市：省工号显示（全省/13地市），地市工号显示本地市（不可修改）。
* 营业厅级别：下拉列表（全部/1级/2级/3级），选择全部，将所有级别的结果都查出来。是否有效：下拉列表（全部/有效/无效）
* 策略名称：下拉列表

1. 查询结果时间倒序排序。
2. 调用【查询审核策略方法】，查询审核记录详情。
3. 用户点击【导出】按钮，将结果列表导出excel文件。

* 数据库表结构
* 审核策略配置表（TO\_REVIEW\_CONFIG）
* 策略商品对照表（TO\_DZ\_REVIEW\_GOODS）
* 详细设计

无

* 界面设计

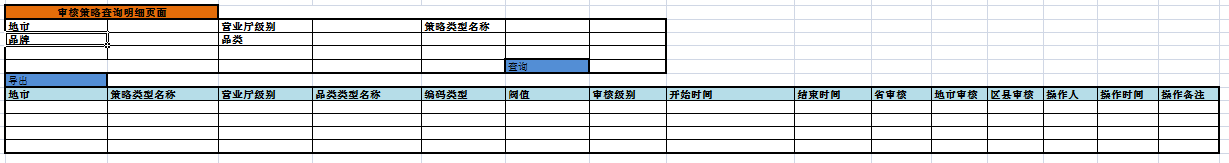


图 -1

#### 决策策略管理

复用【江苏终端销售运营支撑系统\_设计规格说明书】的【6.2.3.10 运营商品类型维护】功能。

### 要货订单管理

#### 要货订单批次发货

#### 要货订单审核

* 处理流程

无

* 功能简述

1. 点击功能菜单后，根据省审核、地市审核、区县审核打开对应的审核界面（图 -1），用户点击【查询】按钮，根据查询条件将符合条件的单据信息查询出来并在列表中展现。
2. 初始化数据

* 省审核，查询F\_ORDER\_STATUSCODE=’待省审核’状态订单
* 地市审核，查询F\_ORDER\_STATUSCODE=’待地市审核’状态订单
* 区县审核，查询F\_ORDER\_STATUSCODE=’待区县审核’状态订单

1. 查询条件说明：

* 单据号，后台模糊查询。
* 开始时间、结束时间，提供日期选择控件。
* 省审核界面中，地市、区县、入货部门都可选择（包括全选）
* 地市审核界面中，地市默认登录人地市，不可修改。
* 区县审核界面中，地市，区县默认登陆人地市、区县，不可修改。

1. 在【要货订单审核界面】，点击【审核】按钮后，跳转到【审核详情页面】。
2. 用户单选某条记录，可进行【审核】操作，提示用户“审核功能只能选中一条记录”。
3. 详情页面显示选中单据信息、要货订单信息、商品信息、单据历史信息及单据操作信息。
4. 在【要货订单审核界面】，点击【批量审核】按钮后，跳转到【审核详情页面】。
5. 用户选中多条记录，可进行【批量审核】操作。
6. 用户点击【批量审核】后，弹出图 -3，用户选择审核结果。
7. 【审核详情页面】（图-2）和（图- 3）选择审核结果，点击【保存】，后将审核结果保存数据库。
8. 审核通过。

* 修改要货订单状态。根据F\_ORDER\_STATUSCODE（订单状态编码）及F\_IS\_COUNTRY\_AUDIT（是否区县/营销单元审核）、F\_IS\_REGION\_AUDIT（是否地市审核）、F\_IS\_PROVINCE\_AUDIT（是否省审核），确定订单提交后状态。

|  |  |  |
| --- | --- | --- |
| **当前状态** | **审核策略** | **提交后状态** |
| 待区县审核 | 需要地市审核 | 待地市审核 |
| 待区县审核 | 不需地市审核，需省审核 | 待省审核 |
| 待区县审核 | 不需地市、省审核 | 待发货 |
| 待地市审核 | 需要省审核 | 待省审核 |
| 待地市审核 | 不需省审核 | 待发货 |
| 省审核 |  | 待发货 |

根据订单类型取对应的订单状态编码

* 增加订单操作日志
* 若提交后状态为“待发货”，则需生成配送单

操作表

下单主表（TO\_BILL\_SENDSTOCK\_MAIN）

配送单子表（TO\_BILL\_SENDSTOCK\_SUB）

--税金=商品订货单价\*进项税率

所有操作必须在同一事务中完成。

1. 审核不通过。

* 修改要货订单状态为“审核不通过”
* 增加订单操作日志
* 修改地市二级库存表【结转】TO\_STOCK\_REGION,累计要货数量（F\_ORDER\_COUNT）的值。累计要货数量=累计要货数量-要货订单商品数量。
* 地市要货预占记录TO\_STOCK\_OCCP\_LOG，新增一条记录，记录预占方向（F\_OCCP\_TYPE）=2。
* 波次可用要货量档案表TO\_WAVE\_GOODS\_COUNT
  1. 查询该地市当前时间是否有波次正在执行。
  2. 如果有，则修改TO\_WAVE\_GOODS\_COUNT，本波次的要货量释放F\_WAVE\_OCCP\_OUT。本波次的要货量释放=本波次的要货量释放+要货订单商品数量
* 营业厅实时库存表TC\_ORG\_STOCK，更新订单在途量和订单在途金额。（减少）。订单在途量=订单在途量 - 要货订单商品数量。订单在途金额=订单在途金额 - 要货订单商品金额。

所有操作必须在同一事务中完成。

1. 批量审核则循环以上操作。（所有批量操作数据放在一个事务中完成）

* 数据库表结构

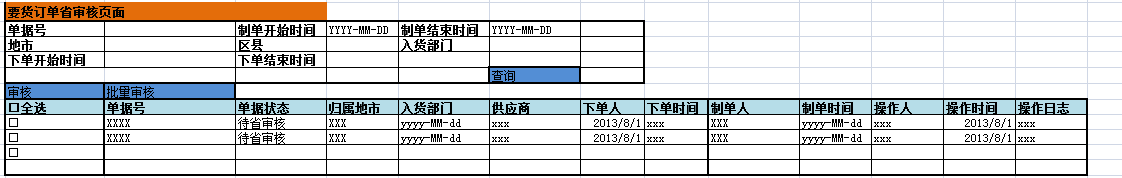
要货订单主表（TO\_ORDERCOMMAND\_MAIN）

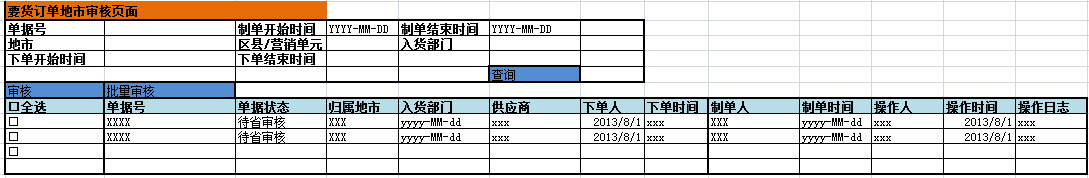
要货订单子表（TO\_ORDERCOMMAND\_SUB）

* 详细设计

无

* 界面设计





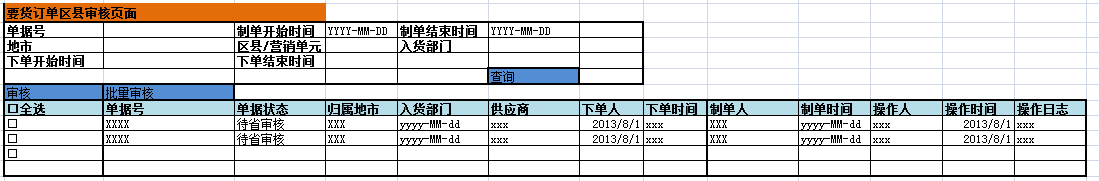


图 -1



图 -2

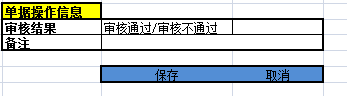


图 -3

#### 要货订单查询

#### 要货订单明细查询

### 调拨订单管理

* 处理流程



* 状态图：



#### 调拨订单申请

* 处理流程



* 功能简述
  + 查询调拨订单
  + 调拨订单　保存草稿
  + 修改调拨订单
  + 删除调拨订单
  + （批量）提交调拨订单
* 数据库表结构

订单处理日志:TO\_ORDER\_HANDLE

订单类型表:TO\_ORDER\_TYPE

订单状态表:TO\_ORDER\_STATUS

订单属性表:TO\_ORDER\_PROPERTY

调拨子订单明细：TO\_ORDERDIAL\_SUB\_DETAIL

调拨子订单：TO\_ORDERDIAL\_SUB

调拨订单：TO\_ORDERDIAL\_MAIN

##### 查询 调拨订单

* 处理流程
* 功能简述
  + 点击功能菜单后，打开【调拨订单申请】（图A）。用户点击【查询】按钮，根据查询条件，调用【IDialOrder.queryOrder】将符合条件的调拨订单查询出来。
* 详细设计

1. 查询条件业务规则如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 查询条件 | 是否有默认值 | 规则说明 | 数据来源 |
| 调拨订单号 | 否 | 模糊查询 | IPrimaryKey.getOrderMainNumber（YYYYMMDDhh24miss+6位SEQ） |
| 调拨类型 | 否 | 不选择时，不限制调拨类型 | 1-自到本自，3-自到本合,4-异地调拨,5-本厅移库，6-备机调拨（暂只支持1，3） |
| 开始时间 | 默认为当前年月第一天 YYYY-MM-DD HH:MM:SS。 | 时间控件 |  |
| 结束时间 | 默认当前时间。 | 时间控件（调拨结束时间>调拨开始时间） |  |
| 调入组织所属城市 | 营业厅人员登陆默认当前地市 |  |  |
| 调入组织机构 | 否 | 与调入组织所属城市联动下拉 |  |

1. 方法说明：

【IDialOrder.queryOrder】查询调拨订单（只可显示订单拨入组织机构与当前用户相同，订单状态为“起草”或“审核驳回”）：

SELECT \* FROM TO\_ORDERDIAL\_MAIN WHERE F\_ORDER\_NUM LIKE '%?%' AND F\_ORDER\_TYPE=? AND F\_ORDER\_STATUSCODE=? AND F\_DIAL\_IN\_ORG=? AND F\_DIAL\_OUT\_ORG=?

* 界面设计

图 A

##### 保存草稿 调拨订单

* 处理流程
* 功能简述
* 点击功能菜单后，打开【调拨订单申请】（图A）。用户点击【查询】按钮，跳转【调拨订单详情】（图B）。【调拨订单详情】分为三个区域：

区域A：涉及调拨主表（TO\_ORDERDIAL\_MAIN）相关信息；

区域B：涉及调拨子表（TO\_ORDERDIAL\_SUB）相关信息；

区域C：涉及调拨子表明细（TO\_ORDERDIAL\_SUB\_DETAIL）相关信息；

* 在【调拨订单详情】图B区域B填写信息，再点击【新增调拨商品】按钮，则在图B区域C中的GRID新增一行
* 在【调拨订单详情】图B区域C中的GRID勾选一行，再点击【删除商品】按钮，则在图B区域C中的GRID删除一行
* 详细设计

1. 业务规则如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | | 是否隐藏 | 验证规则 | 数据来源 |
| 调拨单主表  (区域A) | 调拨订单号 | 是 | 无 | IPrimaryKey.getOrderMainNumber |
| 订单年月 | 是 | 当前年月（分区用,格式YYYYMM） |  |
| 订单类型 | 是 | 固定值，=“DBDD” |  |
| 订单所属城市(调出组织机构所属地市) | 是 | 当前用户 营业厅所属地市 |  |
| 调出组织机构类型 | 是 | 为当前用户 所属营业厅类型 |  |
| 调出组织机构 | 是 | 当前用户 所属营业厅 |  |
| 调拨单类型 | 否 | 若当前用户 营业厅为自办厅，则默认自到本自；~~若当前用户 营业厅为合作厅，则默认为 合到本合。~~ | 自办厅：1-自到本自, 3-自到本合, 4-异地调拨, 5-本厅移库，6-备机调拨; （暂只支持1，2，3） ~~合作厅： 2-合到本合;~~ |
| 调入组织所属地市 | 否 | 调入组织机构所在地市（必填）,默认为用户当前地市。除异地调拨外不允许修改。 |  |
| 调入组织机构类型 | 是 | 调入组织机构类型：若调拨单类型为 自到本自 或 合到本合，则调入组织机构类型 必须与调出组织机构相同；若调拨单类型为自到本合，则调入组织机构类型 固定为 合作厅。 |  |
| 调入组织机构 | 否 | 1.若调拨单类型为自到本自/自到本合/异地调拨，则调入/调出组织机构不能为同一个。  2.若调拨单类型为本厅移库，则调入/调出组织机构必须都为操作人员所属营业厅，且不能修改。 | 统一控件,根据调入组织所属地市/调入组织机构类型 过滤 营业厅。 |
| 拨入库存类型 | 否 | 若调拨单类型为本厅移库，则显示拨入库存类型和拨出库存类型 | 1-正品,2-次品 |
| 拨出库存类型 | 否 | 1-正品,2-次品 |
| 备注 | 否 | 无 |  |
| 调拨子表  (区域B) | 品牌/品类/SQ | 否 | 无 | 统一控件 |
| 商品编码 | 是 | 根据 品牌/品类/SKU控件获得 |  |
| 商品品类 | 是 |  |
| 商品类型 | 是 | 根据 商品编码 查询 商品信息获得 | IGoodsService.getGoodsByGoodsNum |
| 调拨单价 | 是 | =SUM（所有串号采购价格）/调拨数量，串号变动后动态更新 | IGoodImei.queryGoodsImei |
| 调拨数量 | 是 | =导入串号数量 |  |
| 调拨金额 | 是 | = SUM（所有串号采购价格） |  |
| 调拨税额 | 否 | 调拨单营业厅 类型=‘合作营业厅’，则显示该输入框，只允许 number（8，2） |  |
| 备注 | 否 | 无 |  |
|  | 串号 | 否 | 串号之间已逗号分隔,串号必须归属 “调出组织机构”，且状态为“待售”。否则不许提交该调拨订单 |  |
|  | 供应商编码 | 是 | 根据 串号档案表 获得 | IGoodImei.queryGoodsImei |
|  | 串号采购价格 | 是 | 根据 串号档案表 获得 | IGoodImei.queryGoodsImei |
|  | 库存类型 | 是 | 根据 串号档案表 获得 | IGoodImei.queryGoodsImei |
|  | 采购模式 | 是 | 根据 串号档案表 获得1-铺货、2-买断、3-铺货不退货 | IGoodImei.queryGoodsImei |
|  | 结算标记 | 是 | 默认0 |  |

1. 串号导入规则：

串号信息从第二行第一列开始解析，要求串号信息非“#”字符开头。



1. 方法说明：

【IDialOrder.saveOrder】调拨订单暂存：

涉及四张表(TO\_ORDERDIAL\_MAIN,TO\_ORDERDIAL\_SUB,TO\_ORDERDIAL\_SUB\_DETAIL,

TO\_ORDER\_HANDLE)，参考流程如下：



【IDialOrder.deleteOrderSub】调拨订单删除商品：

删除当前调拨订单中的相关商品，不涉及数据库物理删除

* 界面设计

图A



图B

##### 修改 调拨订单

* 处理流程
* 功能简述
  + 1. 点击功能菜单后，打开【调拨订单申请】（图A）。选择一条记录后，用户点击【修改】按钮，调用【IDialOrder.editOrder】,跳转【调拨订单详情】（图B）。【调拨订单详情】分为三个区域：

区域A：涉及调拨主表（TO\_ORDERDIAL\_MAIN）相关信息；

区域B：涉及调拨子表（TO\_ORDERDIAL\_SUB）相关信息；

区域C：涉及调拨子表明细（TO\_ORDERDIAL\_SUB\_DETAIL）相关信息；

* + 1. 在【调拨订单详情】图B区域B填写子表信息，再点击【新增调拨商品】按钮，则在图B区域C中的GRID新增一行
    2. 在【调拨订单详情】图B区域C中的GRID勾选一行，再点击【删除商品】按钮，则在图B区域C中的GRID删除一行
* 详细设计

与“暂存/提交调拨订单“相同

* 界面设计

图A



图B

##### 删除 调拨订单

* 处理流程
* 功能简述

点击功能菜单后，打开【调拨订单申请】（图A）。选择一条记录后，用户点击【删除】按钮，弹出二次确认框，用户二次确认后，调用【IDialOrder.deleteOrder】,删除。

* 详细设计

只可删除状态为“起草”的调拨订单。

调用方法：IDialOrder.deleteOrder

1. 非物理删除，将订单状态置为“删除”
2. 需保存TO\_ORDER\_HANDLE

* 界面设计

图A

##### 提交（批量） 调拨订单

* 处理流程
* 功能简述

点击功能菜单后，打开【调拨订单申请】（图A）。选择一条记录后，用户点击【提交】按钮，弹出二次确认框，用户二次确认后，调用【IDialOrder.commitOrder】,删除。

* 详细设计

调用方法：IDialOrder.commitOrder

【IDialOrder.commitOrder】调拨订单提交：

涉及四张表(TO\_ORDERDIAL\_MAIN,TO\_ORDERDIAL\_SUB,TO\_ORDERDIAL\_SUB\_DETAIL,

TO\_ORDER\_HANDLE)，若T\_SYSTEM表中“isEnableDialStrategy”=1，则触发自动审核流程（为0则忽略自动审核，直接“待审核”）：（先调用CRM接口营业厅终端调参数操作类型=1（在途通知）。若接口返回成功。则执行以下操作（若接口返回失败，则提交失败））：

异常处理：

* 若调用CRM接口返回失败，则我方系统事务ROLLBACK，提示用户

[XX]接口返回失败，并记录ERROR级日志（[‘CRM营业厅终端调拨接口调用失败’,调拨单编码，调拨单状态，~~调用CRM接口报文内容~~，CRM接口返回结果]）

* 若接口返回成功但我方系统事务提交失败，则记录ERROR级日志（[‘CRM营业厅终端调拨接口数据同步失败’,调拨单编码，调拨单状态，CRM接口返回结果]）并更新状态为=‘数据同步异常’（新增TO\_ORDER\_HANDLE记录，更新F\_HANDLE\_MEMO信息为[调拨单编码，调拨单状态，CRM接口返回结果]）

参考流程如下：



* 界面设计

图A

#### 调拨订单审核

* 处理流程



* 功能简述

查询待审核调拨订单

审核待审核调拨订单

* 数据库表结构

订单处理日志:TO\_ORDER\_HANDLE

订单类型表:TO\_ORDER\_TYPE

订单状态表:TO\_ORDER\_STATUS

订单属性表:TO\_ORDER\_PROPERTY

调拨子订单明细：TO\_ORDERDIAL\_SUB\_DETAIL;

调拨子订单：TO\_ORDERDIAL\_SUB;

调拨订单：TO\_ORDERDIAL\_MAIN;

~~调拨付款子订单明细： TO\_ORDERDIAL\_PAY\_DETAIL;~~

~~调拨付款子订单： TO\_ORDERDIAL\_PAY\_SUB;~~

~~调拨付款单： TO\_ORDERDIAL\_PAY\_MAIN;~~

~~要货订单主表：TO\_ORDERCOMMAND\_MAIN~~

~~要货订单子表：TO\_ORDERCOMMAND\_SUB~~

调拨付款主单据:TO\_BILL\_DIAL\_PAY\_MAIN

调拨付款子单据:TO\_BILL\_DIAL\_PAY\_SUB

调拨付款子单据明细:TO\_BILL\_DIAL\_PAY\_SUB\_DETAIL

##### 查询 待审核调拨订单

* 处理流程
* 功能简述
  + 点击功能菜单后，打开【调拨订单审核】（图A）。用户点击【查询】按钮，根据查询条件，调用【IDialOrder.queryOrder】将符合条件的调拨订单查询出来。
* 详细设计

1. 查询条件业务规则如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 查询条件 | 是否有默认值 | 规则说明 | 数据来源 |
| 调拨订单号 | 否 | 模糊查询 | IPrimaryKey.getOrderPrimaryKey |
| 调拨类型 | 否 | 不选择时，不限制调拨类型 | 1-自到本自,~~2-合到本合~~，3-自到本合,4-异地调拨,5-本厅移库，6-备机调拨 |
| 调出组织所属地市 | 是 | 默认为当前用户所属地市（无法修改）；若当前用户为省属，则默认全部。 | 若用户修改调拨类型，可将此值清空 |
| 调出组织机构 | 是 | 与调拨类型/调出组织所属地市关联 | 与调拨类型/调出组织所属地市联动刷新，若用户修改调拨类型，可将此值清空 |
| 调入组织所属地市 | 是 | 默认为当前用户所属地市（无法修改）；若当前用户为省属，则默认全部。 | 若用户修改调拨类型，可将此值清空 |
| 调入组织机构 | 是 | 与调拨类型/调出组织所属地市关联 | 与调拨类型/调入组织所属地市联动刷新，若用户修改调拨类型，可将此值清空 |
| 调拨开始时间 | 是 | 时间控件，默认为当前年月第一天 YYYY-MM-DD HH:MM:SS。 |  |
| 调拨结束时间 | 是 | 时间控件（调拨结束时间>调拨开始时间），默认当前时间。 |  |

1. 方法说明：

【IDialOrder.queryOrder】查询调拨订单：（只可显示与当前登陆用户地市相同，且调拨订单状态为“待审核”）

SELECT \* FROM TO\_ORDERDIAL\_MAIN WHERE F\_ORDER\_NUM LIKE '%?%' AND F\_ORDER\_TYPE=? AND F\_ORDER\_STATUSCODE=? AND F\_DIAL\_IN\_ORG=? AND F\_DIAL\_OUT\_ORG=?

* 界面设计

图A

##### 审核 待审核调拨订单

* 处理流程
* 功能简述
  + 点击功能菜单后，打开【调拨订单审核】（图A）。选择一条记录后，用户点击【审核】按钮，跳转【调拨订单详情】（图B）。
  + 在【调拨订单详情】（图B）中TAB页，跳转到【调拨订单变更历史】（图C）
* 详细设计
  + 业务规则如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 是否隐藏 | 验证规则 | 数据来源 |
| 审核结果 | 否 | 必填 | 1-审核通过，2-审核拒绝，3-审核驳回 |
| 审核意见 | 否 | 非必填 |  |

图B中串号列需以逗号分隔显示在GRID中

* + 方法说明：

【IDialOrder.auditOrderManual】人工审核调拨订单：

1. 若审核拒绝/驳回：
   * + - 先调用CRM接口营业厅终端调参数操作类型=3（调拨取消）。
       - 数据库事务开始
       - 若接口返回成功。则修改调拨单相关串号状态（“调拨在途”->“待售”）。
       - 更新**审核结果到**TO\_ORDERDIAL\_MAIN业务表的F\_STATUS：
         * 将审核的相关信息更新到TO\_ORDERDIAL\_MAIN中的F\_ACCEPT\_TIME，F\_ACCEPT\_MANNAME，F\_ACCEPT\_MANCODE，F\_ACCEPT\_MEMO中
       - 将流程扭转记录存放在TO\_ORDER\_HANDLE表。
       - 数据库事务结束

异常处理：

* 若调用CRM接口返回失败，则我方系统事务ROLLBACK，提示用户

[XX]接口返回失败，并记录ERROR级日志（[‘CRM营业厅终端调拨接口调用失败’,调拨单编码，调拨单状态，~~调用CRM接口报文内容~~，CRM接口返回结果]）

* 若接口返回成功但我方系统事务提交失败，则记录ERROR级日志（[‘CRM营业厅终端调拨接口数据同步失败’,调拨单编码，调拨单状态，CRM接口返回结果]）并更新状态为=‘数据同步异常’（新增TO\_ORDER\_HANDLE记录，更新F\_HANDLE\_MEMO信息为[调拨单编码，调拨单状态，CRM接口返回结果]）

1. 若审核通过，更新（MERGE）营业厅实时库存表:
   * + - 先调用CRM接口营业厅终端调参数操作类型=1（在途通知）。若接口返回成功。则执行以下操作：
       - 数据库事务开始
       - 若调拨类型=“自到本自” OR （调拨类型=“自到本合” AND 付款方式=“货到付款“），营业厅实时库存表TC\_ORG\_STOCK更新：
         * 调入方更新（F\_IN\_ONWAY\_COUNT+调拨总数量，F\_IN\_ONWAY\_MONEY+调拨总金额）
         * 调出方更新（F\_GOODS\_COUNT-调拨总数量，F\_GOODS\_MONEY-调拨总金额，F\_OUT\_ONWAY\_COUNT+调拨总数量，F\_OUT\_ONWAY\_MONEY+调拨总金额）
       - 更新**审核结果到**TO\_ORDERDIAL\_MAIN业务表的F\_STATUS：

将审核的相关信息更新到TO\_ORDERDIAL\_MAIN中的F\_ACCEPT\_TIME，F\_ACCEPT\_MANNAME，F\_ACCEPT\_MANCODE，F\_ACCEPT\_MEMO中

* + - * 将流程扭转记录存放在TO\_ORDER\_HANDLE表。
      * 查询调入组织机构的协同关系:

select f\_csz from t\_region\_system\_dz where F\_CSBM='DBDD\_XTGX\_PAY\_TERMS' and f\_region\_num=调入组织地市;

select f\_csz from t\_region\_system\_dz where F\_CSBM='DBDD\_XTGX\_PAY\_CHANNEL' and f\_region\_num=调入组织地市;

* + - * if 协同关系.支付条件=’货到付款’ then TO\_ORDERDIAL\_MAIN.状态=“执行中”；
      * if协同关系.支付条件=’款到发货’ then TO\_ORDERDIAL\_MAIN.状态=“待付款”；

根据以下规则生成付款单据：

根据供应商和采购模式进行拆单：***调拨串号中，采购模式为“买断”（供应商为省终端运营中心）的合并为一个付款单；采购模式为“铺货”且供应商相同的，可合并为一个支付订单***：

TO\_BILL\_DIAL\_PAY\_MAIN数据保存规则如下：

TO\_BILL\_DIAL\_PAY\_SUB，TO\_BILL\_DIAL\_PAY\_SUB\_DETAIL保存规则略（与TO\_ORDERDIAL\_SUB，TO\_ORDERDIAL\_SUB\_DETAIL类似）

* + - * 数据库事务结束

异常处理：

* 若调用CRM接口返回失败，则我方系统事务ROLLBACK，提示用户

[XX]接口返回失败，并记录ERROR级日志（[‘CRM营业厅终端调拨接口调用失败’,调拨单编码，调拨单状态，~~调用CRM接口报文内容~~，CRM接口返回结果]）

* 若接口返回成功但我方系统事务提交失败，则记录ERROR级日志（[‘CRM营业厅终端调拨接口数据同步失败’,调拨单编码，调拨单状态，CRM接口返回结果]）并更新状态为=‘数据同步异常’（新增TO\_ORDER\_HANDLE记录，更新F\_HANDLE\_MEMO信息为[调拨单编码，调拨单状态，CRM接口返回结果]）
* 界面设计

图A

图B

#### 调拨收款确认

* 处理流程
* 功能简述

查询 待收款调拨单信息

确认 待收款调拨单 已收款

查看某条 待收款调拨单信息

* 数据库表结构

订单处理日志:TO\_ORDER\_HANDLE

订单类型表:TO\_ORDER\_TYPE

订单状态表:TO\_ORDER\_STATUS

订单属性表:TO\_ORDER\_PROPERTY

调拨子订单明细：TO\_ORDERDIAL\_SUB\_DETAIL;

调拨子订单：TO\_ORDERDIAL\_SUB;

调拨订单：TO\_ORDERDIAL\_MAIN;

要货订单主表：TO\_ORDERCOMMAND\_MAIN

要货订单子表：TO\_ORDERCOMMAND\_SUB

* 详细设计

1. 查询 待收款调拨单信息
   * 查询条件初始化

地市：默认为当前用户所属地市（无法修改）；若当前用户为省属，则默认全部。

是否启用：默认全部（可选择，启用/停用）。

* + 查询数据

根据查询条件，传入查询条件，调用逻辑层接口，查询数据（是否可用+最后更新时间 正序）返回后，分页显示查询结果。

1. 确认 待收款调拨单 已收款

单选某条 备待收款调拨单，提示是否确认。确认后直接提交该信息。

* 业务规则：
* 调出方 调拨单（TO\_ORDERDIAL\_MAIN）状态改为“调拨中”。
* 调入方 要货单（TO\_ORDERCOMMAND\_MAIN）状态改为“待发货”；支付完成时间F\_PAY\_TIME为当前时间；已支付金额F\_ORDER\_PAYMONEY为F\_ORDER\_MONEY。

1. 查看 待收款调拨单信息

单选某条 待收款调拨单信息，根据主键传递到【调拨单信息 查看】页面。

各字段无法修改。

* 界面设计
  + 查询 待收款调拨单信息页面



* + 查看 调拨单信息



#### 调拨订单验收

* 处理流程



* 功能简述
  + 查询带验收订单
  + 验收带验收订单
* 数据库表结构

订单处理日志:TO\_ORDER\_HANDLE

订单类型表:TO\_ORDER\_TYPE

订单状态表:TO\_ORDER\_STATUS

订单属性表:TO\_ORDER\_PROPERTY

调拨子订单明细：TO\_ORDERDIAL\_SUB\_DETAIL;

调拨子订单：TO\_ORDERDIAL\_SUB;

调拨订单：TO\_ORDERDIAL\_MAIN;

调拨单据: TO\_BILL\_DIAL\_MAIN

调拨子单据: TO\_BILL\_DIAL\_SUB

调拨子单据明细: TO\_BILL\_DIAL\_SUB\_DETAIL

调拨付款主单据:TO\_BILL\_DIAL\_PAY\_MAIN

调拨付款子单据:TO\_BILL\_DIAL\_PAY\_SUB

调拨付款子单据明细:TO\_BILL\_DIAL\_PAY\_SUB\_DETAIL

##### 查询 待验收订单

* 处理流程
* 功能简述
  + 点击功能菜单后，打开【调拨订单查询】（图A）。用户点击【查询】按钮，根据查询条件，调用【IDialOrder.queryOrder】将符合条件的调拨订单查询出来。
* 详细设计
  + 查询条件业务规则如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 查询条件 | 是否有默认值 | 规则说明 | 数据来源 |
| 调拨订单号 | 否 | 模糊查询 |  |
| 调拨类型 | 否 | 不选择时，不限制调拨类型 | 1-跨厅调拨,2-本厅移库 |
| 调拨开始时间 | 否 | 时间控件 |  |
| 调拨结束时间 | 否 | 时间控件（调拨结束时间>调拨开始时间） |  |
| 调出组织机构 | 否 | 点击按钮后，在新开窗口中选择组织机构，再带回到输入框 |  |

* + 方法说明：

【IDialOrder.queryOrder】查询调拨订单（只能查询调拨订单的拨入组织机构等于登陆用户所属组织机构，且调拨订单状态为“执行中”的所有调拨订单）

* 界面设计



图A

##### 验收 调拨订单

* 处理流程
* 功能简述
  + 点击功能菜单后，打开【验收调拨订单】（图A）。选择一条记录后，用户点击【验收】按钮，跳转【调拨订单详情】（图B）。
  + 在【调拨订单详情】（图B）勾选商品，再点击【验收串号】，跳转到【验收串号信息】（图C）
* 详细设计
  + 业务规则如下：

略

* + 方法说明：

【IDialOrder.acceptOrder】查询调拨订单（地市人员登录只能查看到本地市的调拨订单）：

业务规则：

* + - * 1. TO\_BILL\_DAIL\_MAIN主键生成规则参考IPrimaryKey.getPrimaryKey（YYYYMMDDhh24miss+6位SEQ）
        2. 验收后：
    1. 先调用CRM接口营业厅终端调参数操作类型=2（调拨确认）。若接口返回成功。则执行以下操作：
    2. 数据库事务开始。
    3. 修改所有串号状态（“调拨在途”->“待售”）和串号归属
    4. 调整拨入方和拨出方的商品库存。
    5. 更新调拨订单状态为“已验收”。
    6. 更新（MERGE）营业厅实时库存表（SQL仅供参考）：

针对拨入方：MERGE INTO TC\_ORG\_STOCK g

USING TO\_ORDERDIAL\_SUB ng

ON (g.F\_ORG\_NUM = '拨入组织机构编码' and g.F\_GOODS\_NUM = ng.F\_GOODS\_NUM and g.F\_STOCK\_TYPE\_NUM = ng.'正品/次品')

WHEN MATCHED THEN

UPDATE

SET g.F\_GOODS\_COUNT=G.F\_GOODS\_COUNT+'调拨总数量',g.F\_GOODS\_COUNT=G.F\_GOODS\_MONEY+'调拨总金额', F\_IN\_ONWAY\_COUNT =F\_IN\_ONWAY\_COUNT-‘调拨总数量’,F\_IN\_ONWAY\_MONEY=F\_IN\_ONWAY\_MONEY-‘调拨总金额’

WHERE (g.F\_ORG\_NUM = '拨入组织机构编码' and g.F\_GOODS\_NUM = ng.F\_GOODS\_NUM and g.F\_STOCK\_TYPE\_NUM = ng.'调拨商品类型')

WHEN NOT MATCHED THEN

INSERT INTO TC\_ORG\_STOCK (F\_REGION\_NUM, F\_ORG\_NUM, F\_GOODS\_NUM, F\_STOCK\_TYPE\_NUM, F\_GOODS\_COUNT, F\_GOODS\_MONEY,

F\_ONWAY\_COUNT, F\_ONWAY\_MONEY) VALUES (TO\_ORDERDIAL\_SUB.F\_ORDER\_REGION,'拨入组织机构编码' ,调拨商品编码 , ''正品/次品','调拨总数量','调拨总金额',0 ,0 );

针对拨出方：MERGE INTO TC\_ORG\_STOCK g

USING TO\_ORDERDIAL\_SUB ng

ON (g.F\_ORG\_NUM = '拨出组织机构编码' and g.F\_GOODS\_NUM = ng.F\_GOODS\_NUM and g.F\_STOCK\_TYPE\_NUM = ng.'正品/次品')

WHEN MATCHED THEN

UPDATE

SET F\_OUT\_ONWAY\_COUNT =F\_OUT\_ONWAY\_COUNT-‘调拨总数量’,F\_OUT \_ONWAY\_MONEY=F\_OUT \_ONWAY\_MONEY-‘调拨总金额’

WHERE (g.F\_ORG\_NUM = '拨出组织机构编码' and g.F\_GOODS\_NUM = ng.F\_GOODS\_NUM and g.F\_STOCK\_TYPE\_NUM = ng.'调拨商品类型')

WHEN NOT MATCHED THEN

INSERT INTO TC\_ORG\_STOCK (F\_REGION\_NUM, F\_ORG\_NUM, F\_GOODS\_NUM, F\_STOCK\_TYPE\_NUM, F\_GOODS\_COUNT, F\_GOODS\_MONEY,

F\_ONWAY\_COUNT, F\_ONWAY\_MONEY) VALUES (TO\_ORDERDIAL\_SUB.F\_ORDER\_REGION,'拨出组织机构编码' ,调拨商品编码 , '正品/次品',-'调拨总数量',-'调拨总金额',0 ,0 );

* + 1. 更新（MERGE）进销存表

若调拨订单类型为“自到本自”：

针对拨入方：MERGE INTO TC\_ORG\_PSSM g

USING TO\_ORDERDIAL\_SUB ng

ON (g.F\_ORG\_NUM = '拨入组织机构编码' and g.F\_OCCUR\_DATE='验收年月日' and g.F\_GOODS\_NUM=ng.F\_GOODS\_NUM ,g.F\_GOODS\_TYPE= ng.'正品/次品')

WHEN MATCHED THEN

UPDATE

SET g.F\_DIALIN\_COUNT=G.F\_DIALIN\_COUNT+'调拨总数量',g.F\_DIALIN\_COUNT=G.F\_DIALIN\_COUNT+'调拨总金额'

WHERE (g.F\_ORG\_NUM = '拨入组织机构编码' and g.F\_OCCUR\_DATE='验收年月日' and g.F\_GOODS\_NUM=ng.F\_GOODS\_NUM ,g.F\_GOODS\_TYPE= ng.'正品/次品')

WHEN NOT MATCHED THEN

INSERT INTO TC\_ORG\_PSSM (F\_REGION\_NUM,F\_ORG\_NUM,F\_OCCUR\_DATE,F\_GOODS\_NUM,F\_SUPPLIER\_NUM,F\_GOODS\_TYPE,F\_DIALIN\_COUNT,F\_DIALIN\_COST,F\_DIALIN\_MARGIN,F\_FIRST\_COUNT,F\_FIRST\_COST,F\_FIRST\_MARGIN)

VALUES (TO\_ORDERDIAL\_SUB.F\_ORDER\_REGION,'拨入组织机构编码','验收年月日',调拨商品编码 ,F\_SUPPLIER\_NUM, '正品/次品','调拨总数量','调拨总金额',0,0,0,0)

针对拨出方：MERGE INTO TC\_ORG\_PSSM g

USING TO\_ORDERDIAL\_SUB ng

ON (g.F\_ORG\_NUM = '拨出组织机构编码' and g.F\_OCCUR\_DATE='验收年月日' and g.F\_GOODS\_NUM=ng.F\_GOODS\_NUM ,g.F\_GOODS\_TYPE= ng.'正品/次品')

WHEN MATCHED THEN

UPDATE

SET g.F\_DIALIN\_COUNT=G.F\_DIALIN\_COUNT-'调拨总数量',g.F\_DIALIN\_COUNT=G.F\_DIALIN\_COUNT-'调拨总金额'

WHERE (g.F\_ORG\_NUM = '拨出组织机构编码' and g.F\_OCCUR\_DATE='验收年月日' and g.F\_GOODS\_NUM=ng.F\_GOODS\_NUM ,g.F\_GOODS\_TYPE= ng.'正品/次品')

WHEN NOT MATCHED THEN

INSERT INTO TC\_ORG\_PSSM (F\_REGION\_NUM,F\_ORG\_NUM,F\_OCCUR\_DATE,F\_GOODS\_NUM,F\_SUPPLIER\_NUM,F\_GOODS\_TYPE,F\_DIALIN\_COUNT,F\_DIALIN\_COST,F\_DIALIN\_MARGIN,,F\_FIRST\_COUNT,F\_FIRST\_COST,F\_FIRST\_MARGIN)

VALUES (TO\_ORDERDIAL\_SUB.F\_ORDER\_REGION,'拨出组织机构编码','验收年月日',调拨商品编码 ,F\_SUPPLIER\_NUM, '正品/次品','调拨总数量','调拨总金额',0,0,0,0)

若调拨订单类型为“自到本合”：

针对拨入方：MERGE INTO TC\_ORG\_PSSM g

USING TO\_ORDERDIAL\_SUB ng

ON (g.F\_ORG\_NUM = '拨入组织机构编码' and g.F\_OCCUR\_DATE='验收年月日' and g.F\_GOODS\_NUM=ng.F\_GOODS\_NUM ,g.F\_GOODS\_TYPE= ng.'正品/次品')

WHEN MATCHED THEN

UPDATE

SET g. F\_PURCH\_COUNT =G. F\_PURCH\_COUNT +'调拨总数量',g. F\_PURCH\_COST =G. F\_PURCH\_COST +'调拨总金额'

WHERE (g.F\_ORG\_NUM = '拨入组织机构编码' and g.F\_OCCUR\_DATE='验收年月日' and g.F\_GOODS\_NUM=ng.F\_GOODS\_NUM ,g.F\_GOODS\_TYPE= ng.'正品/次品')

WHEN NOT MATCHED THEN

INSERT INTO TC\_ORG\_PSSM (F\_REGION\_NUM,F\_ORG\_NUM,F\_OCCUR\_DATE,F\_GOODS\_NUM,F\_SUPPLIER\_NUM,F\_GOODS\_TYPE, F\_PURCH\_COUNT, F\_PURCH\_COST,F\_DIALIN\_MARGIN,F\_FIRST\_COUNT,F\_FIRST\_COST,F\_FIRST\_MARGIN)

VALUES (TO\_ORDERDIAL\_SUB.F\_ORDER\_REGION,'拨入组织机构编码','验收年月日',调拨商品编码 ,F\_SUPPLIER\_NUM, '正品/次品','调拨总数量','调拨总金额',0,0,0,0)

针对拨出方：MERGE INTO TC\_ORG\_PSSM g

USING TO\_ORDERDIAL\_SUB ng

ON (g.F\_ORG\_NUM = '拨出组织机构编码' and g.F\_OCCUR\_DATE='验收年月日' and g.F\_GOODS\_NUM=ng.F\_GOODS\_NUM ,g.F\_GOODS\_TYPE= ng.'正品/次品')

WHEN MATCHED THEN

UPDATE

SET g. F\_IN\_DIST\_COUNT =G. F\_IN\_DIST\_COUNT -'调拨总数量',g. F\_IN\_DIST\_COST =G. F\_IN\_DIST\_COST -'调拨总金额'

WHERE (g.F\_ORG\_NUM = '拨出组织机构编码' and g.F\_OCCUR\_DATE='验收年月日' and g.F\_GOODS\_NUM=ng.F\_GOODS\_NUM ,g.F\_GOODS\_TYPE= ng.'正品/次品')

WHEN NOT MATCHED THEN

INSERT INTO TC\_ORG\_PSSM (F\_REGION\_NUM,F\_ORG\_NUM,F\_OCCUR\_DATE,F\_GOODS\_NUM,F\_SUPPLIER\_NUM,F\_GOODS\_TYPE, F\_IN\_DIST\_COUNT, F\_IN\_DIST\_COST,F\_DIALIN\_MARGIN,,F\_FIRST\_COUNT,F\_FIRST\_COST,F\_FIRST\_MARGIN)

VALUES (TO\_ORDERDIAL\_SUB.F\_ORDER\_REGION,'拨出组织机构编码','验收年月日',调拨商品编码 ,F\_SUPPLIER\_NUM, '正品/次品','调拨总数量','调拨总金额',0,0,0,0)

查找地市对应的省库，更新该仓库的进销存：MERGE INTO TC\_ORG\_PSSM g

USING TO\_ORDERDIAL\_SUB ng

ON (g.F\_ORG\_NUM = '省库编码' and g.F\_OCCUR\_DATE='验收年月日' and g.F\_GOODS\_NUM=ng.F\_GOODS\_NUM ,g.F\_GOODS\_TYPE= ng.'正品/次品')

WHEN MATCHED THEN

UPDATE

SET g. F\_INSALE\_COUNT =G. F\_INSALE\_COUNT +'调拨总数量',g. F\_INSALE\_INCOME =G. F\_INSALE\_INCOME +'调拨总金额'

WHERE (g.F\_ORG\_NUM = '拨出组织机构编码' and g.F\_OCCUR\_DATE='验收年月日' and g.F\_GOODS\_NUM=ng.F\_GOODS\_NUM ,g.F\_GOODS\_TYPE= ng.'正品/次品')

WHEN NOT MATCHED THEN

INSERT INTO TC\_ORG\_PSSM (F\_REGION\_NUM,F\_ORG\_NUM,F\_OCCUR\_DATE,F\_GOODS\_NUM,F\_SUPPLIER\_NUM,F\_GOODS\_TYPE, F\_INSALE\_COUNT, F\_INSALE\_INCOME,F\_DIALIN\_MARGIN,,F\_FIRST\_COUNT,F\_FIRST\_COST,F\_FIRST\_MARGIN)

VALUES (TO\_ORDERDIAL\_SUB.F\_ORDER\_REGION,'省库编码'','验收年月日',调拨商品编码 ,F\_SUPPLIER\_NUM, '正品/次品','调拨总数量','调拨总金额',0,0,0,0)

MERGE INTO TC\_ORG\_PSSM g

USING TO\_ORDERDIAL\_SUB ng

ON (g.F\_ORG\_NUM = '省库编码' and g.F\_OCCUR\_DATE='验收年月日' and g.F\_GOODS\_NUM=ng.F\_GOODS\_NUM ,g.F\_GOODS\_TYPE= ng.'正品/次品')

WHEN MATCHED THEN

UPDATE

SET g. F\_OUT\_DIST\_COUNT =G. F\_OUT\_DIST\_COUNT -'调拨总数量',g. F\_OUT\_DIST\_COST =G. F\_OUT\_DIST\_COST -'调拨总金额'

WHERE (g.F\_ORG\_NUM = '拨出组织机构编码' and g.F\_OCCUR\_DATE='验收年月日' and g.F\_GOODS\_NUM=ng.F\_GOODS\_NUM ,g.F\_GOODS\_TYPE= ng.'正品/次品')

WHEN NOT MATCHED THEN

INSERT INTO TC\_ORG\_PSSM (F\_REGION\_NUM,F\_ORG\_NUM,F\_OCCUR\_DATE,F\_GOODS\_NUM,F\_SUPPLIER\_NUM,F\_GOODS\_TYPE, F\_OUT\_DIST\_COUNT, F\_OUT\_DIST\_COST,F\_DIALIN\_MARGIN,,F\_FIRST\_COUNT,F\_FIRST\_COST,F\_FIRST\_MARGIN)

VALUES (TO\_ORDERDIAL\_SUB.F\_ORDER\_REGION,'省库编码'','验收年月日',调拨商品编码 ,F\_SUPPLIER\_NUM, '正品/次品','调拨总数量','调拨总金额',0,0,0,0)

* + 1. 保存三张单据表（TO\_BILL\_DIAL\_MAIN，TO\_BILL\_DIAL\_SUB，TO\_BILL\_DIAL\_SUB\_DETAIL）
    2. 若调拨类型为“货到付款”：更新要货订单信息：TO\_BILL\_DIAL\_PAY\_MAIN表状态为“结束”；F\_PAY\_TIME为当前日期。
    3. 数据库事务结束。

异常处理：

* 若调用CRM接口返回失败，则我方系统事务ROLLBACK，提示用户

[XX]接口返回失败，并记录ERROR级日志（[‘CRM营业厅终端调拨接口调用失败’,调拨单编码，调拨单状态，~~调用CRM接口报文内容~~，CRM接口返回结果]）

* 若接口返回成功但我方系统事务提交失败，则记录ERROR级日志（[‘CRM营业厅终端调拨接口数据同步失败’,调拨单编码，调拨单状态，CRM接口返回结果]）并更新状态为=‘数据同步异常’（新增TO\_ORDER\_HANDLE记录，更新F\_HANDLE\_MEMO信息为[调拨单编码，调拨单状态，CRM接口返回结果]）
  + - * 1. 参考流程如下：



调用方法：IDialOrder.acceptOrder

* 界面设计



图A



图B

#### 调拨订单审核策略

* 处理流程
* 功能简述
  + 查询调拨订单自动审核配置
  + 新增/修改调拨订单自动审核配置
  + 删除调拨订单自动审核配置
* 数据库表结构
  + 业务表

自动审核调拨订单配置 运算符类型：TO\_DIAL\_STRATEGY\_OPER\_TYPE;

自动审核逻辑运算符：TO\_DIAL\_STRATEGY\_OPER;

调拨自动审核属性：TO\_DIAL\_STRATEGY\_FIELD;

调拨订单审核策略配置：TO\_DIAL\_STRATEGY;

##### 查询 调拨订单自动审核配置

* 处理流程
* 功能简述
* 点击功能菜单后，打开【查询调拨订单自动审核配置】（图A）。用户点击【查询】按钮，根据查询条件，调用【IDialAuditStrategy.queryStrategy】将符合条件的调拨订单查询出来。
* 详细设计
  + 查询条件业务规则如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 查询条件 | 是否有默认值 | 规则说明 | 数据来源 |
| 策略名称 | 否 | 模糊查询 |  |
| 策略生效时间晚于 | 否 | 时间控件 |  |
| 策略生效时间早于 | 否 | 时间控件 |  |

* + 方法说明：

【IDialAuditStrategy.queryStrategy】查询调拨订单：

SELECT \* FROM TO\_DIAL\_STRATEGY WHERE F\_STRATEGY\_NAME LIKE '%?%' AND F\_START\_DATE>='?' AND F\_END\_DATE <= '?'

* 界面设计



图Ａ

##### 新增/修改调拨订单自动审核配置

* 处理流程
* 功能简述
  + 点击功能菜单后，打开【查询调拨订单自动审核配置】（图A）。用户点击【新增】按钮，跳转【自动审核策略详情】（图B）。
  + 点击功能菜单后，打开【查询调拨订单自动审核配置】（图A）。用户选择一条记录后，点击【修改】按钮，跳转【自动审核策略详情】（图B）。
* 详细设计：
  + 业务规则如下：
    - 表TO\_DIAL\_STRATEGY\_FIELD，TO\_DIAL\_STRATEGY\_OPER，TO\_DIAL\_STRATEGY\_OPER\_TYPE

主要负责策略属性与运算符和值得对应关系控制页面联动



* + - 策略名称：必填（规则：字母　或　数字）
    - 策略所属地市：必填
    - 表达式配置业务规则：



如果属性选择“调拨类型”，运算符域为“等于，不等于”，值域变为下拉框（值改为“1-自到本自，3-自到本合,4-异地调拨,5-本厅移库，6-备机调拨”），生成的表达式值为（例：$dial\_type eq’自到本自'）



如果属性选择“调拨总数量，调拨总金额”，运算符域为“>,<,=,<>,<=,>=”,值域变为输入框（只能输入数字），生成的表达式值为（例：$total\_amount <=20）



如果属性选择“调拨发出方，调拨接收方”，运算符域为“等于，不等于”，值域变为两个下拉框（第一个下拉框选择营业厅等级），生成的表达式值为（例：$拨入组织机构 ne '二级营业厅'）



如果属性选择“商品品类，商品”，运算符域为“包含，不包含”，值域变为弹出窗口返回的输入框，生成的表达式值为（例：$goods\_category.in({'code31':'iphone5s','code32':'iphone4'}）



* + - 策略公式:



例：页面显示公式为：

$调拨类型 eq '跨厅调拨'

&& ($总数量<=20 ||$总金额<= 30000)

&& $拨出组织机构eq '一级营业厅'

|| $拨入组织机构 ne '二级营业厅'

&& $产品品类 .in {'code31':'iphone5s','code32':'iphone4'}

&& $产品 .notin {'code33':'iphone5s','code34':'iphone4'}

但实际使用(且保存入数据库时)公式为：

$dial\_type == '跨厅调拨'

&& ($total\_amount <=20 ||$total\_price <= 30000)

&& $dial\_out\_org eq '一级营业厅'

|| $dial\_in\_org ne '二级营业厅'

&& $goods\_category.in ({'code31':'iphone5s','code32':'iphone4'})

&& $goods.notin ({'code33':'iphone5s','code34':'iphone4'})

* + - 策略启用类型：分为“永久生效”和“一次生效”；只有“一次生效”才显示。



* + - 备注：非必选
  + 方法说明：

【IDialAuditStrategy.saveOrUpdateStrategy】保存/更新自动审核策略：

* + - 主键生成规则参考IPrimaryKey.getPrimaryKey（YYYYMMDDhh24miss+6位SEQ）
    - 将页面显示之公式转换成实际使用的公式格式：

//转换规则：

//$调拨类型 -> $dial\_type

//$总数量 -> $total\_amount

//$总金额 -> $total\_price

//$拨出组织机构 -> $dial\_out\_org

//$拨入组织机构 -> $dial\_in\_org

//$产品品类 -> $goods\_category

//$产品 -> $goods

//{ -> ({

//} -> })

自动审核策略的完整性验证：将测试变量值带入表达式，如果语句执行不报异常，且有正常返回，则验证通过

JexlContext context = **new**MapContext();

context.set("$dial\_type", "跨厅调拨");

context.set("$dial\_out\_org", category);

context.set("$dial\_in\_org", category);

context.set("$total\_amount", **new** Integer(11));

context.set("$total\_price", 40000);

context.set("$goods\_category", category);

* + - 自动审核策略的业务验证：所有的Map对象(类似：{'code31':'iphone5s','code32':'iphone4'})都要经过业务验证，”:”左侧的key和右侧的value必须与数据库中的数据完全匹配。
    - 采用APACHE 的JEXL组件，JAR和示例见附件



* 界面设计



图Ａ



图B

##### 删除调拨订单自动审核配置

* 处理流程
* 功能简述
  + 点击功能菜单后，打开【查询调拨订单自动审核配置】（图A）。用户选择一条记录后，点击【删除】按钮。弹出二次确认框，用户二次确认后，删除。
* 详细设计

调用方法：IDialAuditStrategy.deleteStrategy（物理删除）

* 界面设计



图A

#### 调拨订单查询

* 处理流程
* 功能简述
  + 查询调拨订单
  + 查看调拨订单
  + 导出调拨订单
* 数据库表结构

订单处理日志:TO\_ORDER\_HANDLE

订单类型表:TO\_ORDER\_TYPE

订单状态表:TO\_ORDER\_STATUS

订单属性表:TO\_ORDER\_PROPERTY

调拨子订单明细：TO\_ORDERDIAL\_SUB\_DETAIL;

调拨子订单：TO\_ORDERDIAL\_SUB;

调拨订单：TO\_ORDERDIAL\_MAIN;

##### 查询 调拨订单

* 处理流程
* 功能简述
  + 点击功能菜单后，打开【调拨订单查询】（图A）。用户点击【查询】按钮，根据查询条件，调用【IDialOrder.queryOrder】将符合条件的调拨订单查询出来。
* 详细设计
  + 查询条件业务规则如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 查询条件 | 是否有默认值 | 规则说明 | 数据来源 |
| 调拨订单号 | 否 | 模糊查询 |  |
| 调拨类型 | 否 | 不选择时，不限制调拨类型 | 见备注 |
| 调入组织所属城市 | 省运营人员登陆默认全部；地市/营业厅人员登陆默认当前地市 | 地市/营业厅人员登陆时，默认为当前地市，无法修改。 |  |
| 调入组织机构 | 默认全部 | 点击按钮后，在新开窗口中选择组织机构，再带回到输入框；营业厅人员登陆时，调出/调入组织机构 两者其一必须为当前营业厅 |  |
| 调出组织所属城市 | 省运营人员登陆默认全部；地市/营业厅人员登陆默认当前地市 | 地市/营业厅人员登陆时，默认为当前地市，无法修改。 |  |
| 调出组织机构 | 地市人员/省运营人员登陆默认全部； 营业厅人员登陆默认当前营业厅； | 点击按钮后，在新开窗口中选择组织机构，再带回到输入框；营业厅人员登陆时，调出/调入组织机构 两者其一必须为当前营业厅 |  |
| 调拨开始时间 | 默认为当前年月第一天 YYYY-MM-DD HH:MM:SS | 时间控件 |  |
| 调拨结束时间 | 默认当前时间 | 时间控件（调拨结束时间>调拨开始时间） |  |
| 状态 | 默认全部 |  | select F\_ORDER\_STATUSNAME from to\_order\_status where f\_order\_typecode='DBDD'and f\_serial<90 |

* + 方法说明：

【IDialOrder.queryOrder】查询调拨订单（状态不等于‘数据同步异常’）：

* 界面设计



图A

##### 查看 调拨订单

* 处理流程
* 功能简述
  + 点击功能菜单后，打开【调拨订单查询】（图A）。用户点击【查询】按钮，根据查询条件，调用【IDialOrder.queryOrder】将符合条件的调拨订单查询出来。
  + 打开【调拨订单查询】（图A）。选择一条记录后，用户点击【查看】按钮，跳转【调拨订单详情】（图B）。
  + 在【调拨订单详情】（图B）中TAB页，跳转到【调拨订单变更历史】（图C）
* 详细设计
  + 业务规则如下：

略（图B中串号列需以逗号分隔显示在GRID中）

* + 方法说明：

【IDialOrder.viewIMEI】查询调拨订单（地市人员登录只能查看到本地市的调拨订单）：

* 界面设计

图A

图B



图C

##### 导出 调拨订单

* 处理流程
* 功能简述
  + 点击功能菜单后，打开【调拨订单查询】（图A）。用户点击【查询】按钮，根据查询条件，调用【IDialOrder.queryOrder】将符合条件的调拨订单查询出来。
  + 打开【调拨订单查询】（图A）。用户点击【导出】按钮，调用【IDialOrder.exportOrder】将符合条件的调拨订单导出来。（图B）。
* 详细设计

导出功能参考架构组统一组件

* 界面设计

无

#### 协同关系维护

* 处理流程
* 功能简述

查询 协同关系信息

修改已经维护的 协同关系信息

查看某条 协同关系信息

新增新的 协同关系信息

* 数据库表结构

协同关系信息：TO\_DA\_ORG\_COOPERATE

* 详细设计

1. 查询 协同关系信息
   * 查询条件初始化

地市：默认为当前用户所属地市（无法修改）；若当前用户为省属，则默认全部。

是否启用：默认全部（可选择，启用/停用）。

* + 查询数据

根据查询条件，传入查询条件，调用逻辑层接口，查询数据（是否可用+最后更新时间 正序）返回后，分页显示查询结果。

1. 查看 协同关系信息

单选某条 协同关系信息，根据主键传递到【协同关系信息 查看】页面。

各字段无法修改。

1. 修改 协同关系信息

单选 某条协同关系信息，根据主键传递到【协同关系信息 新增/修改】页面。

编码不能修改，其他字段都可修改。保存数据到数据库，提示用户是否修改成功，如果成功关闭页面，刷新父页面。

* 页面初始化（新增/修改）： ：
* 所属地市：默认为当前用户所属地市（无法修改）；若当前用户为省属，则默认全部。
* 付款条件：（1-货到付款，2-款到发货），默认1。
* 付款方式：（1-线下交易，2-在线支付），默认1。
* 业务规则（新增/修改）：
* 同一地市 不可存在多条记录。
* 付款条件为“1-货到付款”时，付款方式只支持“1-线下交易”（暂）。
* 付款条件为“2-款到发货”时，付款方式只支持“2-在线支付”（暂）。

1. 新增 协同关系信息

点击新增按钮，传递到页面【协同关系信息 新增/修改】页面。

保存数据到数据库，提示用户是否修改成功，如果成功关闭页面，刷新父页面。

* 页面初始化（新增/修改）： ：
* 所属地市：默认为当前用户所属地市（无法修改）；若当前用户为省属，则默认全部。
* 付款条件：（1-货到付款，2-款到发货），默认1。
* 付款方式：（1-线下交易，2-在线支付），默认1。
* 业务规则（新增/修改）：
* 同一地市 不可存在多条记录。
* 付款条件为“1-货到付款”时，付款方式只支持“1-线下交易”（暂）。
* 付款条件为“2-款到发货”时，付款方式只支持“2-在线支付”（暂）。
* 界面设计
  + 协同关系信息 查询页面



* + 协同关系信息 新增/修改页面



* + 协同关系信息 查看页面



#### 调拨在线付款

* 处理流程





* 功能简述
  + 调用付款接口
  + 查询付款状态
  + 接受付款结果通知
* 数据库表结构

订单处理日志:TO\_ORDER\_HANDLE

订单类型表:TO\_ORDER\_TYPE

订单状态表:TO\_ORDER\_STATUS

订单属性表:TO\_ORDER\_PROPERTY

调拨子订单明细：TO\_ORDERDIAL\_SUB\_DETAIL;

调拨子订单：TO\_ORDERDIAL\_SUB;

调拨订单：TO\_ORDERDIAL\_MAIN;

调拨单据: TO\_BILL\_DIAL\_MAIN

调拨子单据: TO\_BILL\_DIAL\_SUB

调拨子单据明细: TO\_BILL\_DIAL\_SUB\_DETAIL

~~调拨付款子订单明细： TO\_ORDERDIAL\_PAY\_DETAIL;~~

~~调拨付款子订单： TO\_ORDERDIAL\_PAY\_SUB;~~

~~调拨付款单：TO\_ORDERDIAL\_PAY\_MAIN;~~

要货订单主表：TO\_ORDERCOMMAND\_MAIN

要货订单子表：TO\_ORDERCOMMAND\_SUB

##### （暂不使用）调用付款接口

1. 处理流程

运营支撑系统-〉B2B（或其他负责支付系统）

1. 功能简述

定时（每五分钟，可配置）扫描表 调拨要货单（select \* from TO\_ORDERCOMMAND\_MAIN where status=*创建*）;若存在记录，调用B2B（或其他负责支付系统）付款接口（每条记录一个事务：成功更新 调拨要货单status=*待付款*;失败则更新status=*付款通讯异常*）

1. 数据库表结构
2. 详细设计

* 请求内容content:
* 应答内容content:
* 业务规则：

##### （暂不使用）查询付款单状态

1. 处理流程

运营支撑系统-〉B2B（或其他负责支付系统）

1. 功能简述

查询B2B（或其他负责支付系统）系统中某 调拨要货单 的付款结果。

1. 数据库表结构
2. 详细设计

* 请求内容content:
* 应答内容content:
* 业务规则：

##### 接收付款结果通知

1. 处理流程

B2B（或其他负责支付系统）-〉运营支撑系统

1. 功能简述

营业厅在B2B中完成支付后，B2B需将付款结果通知我方。

1. 数据库表结构
2. 详细设计

* 请求内容content:

| 父元素名称 | 元素名称 | 约束 | 类型 | 宽度 | 参数描述 | 备注 |
| --- | --- | --- | --- | --- | --- | --- |
| operation\_in | content | 1 | string | — | 请求内容 | XML格式的字符串 |
| content | oper\_source | 1 | long | V8 | 请求来源 | B2B |
| content | bill\_num | 1 | string | V32 | 付款单据号 |  |
| content | pay\_serial\_num | 1 | string | V32 | 支付流水号 |  |
| content | bill\_amount | 1 | string | V20 | 单据金额 |  |
| content | pay\_amount | 1 | string | V20 | 付款金额 |  |
| content | pay\_time |  | string | V17 | 付款时间 | yyyymmddhhmmss |
| content | dial\_out\_org\_num | + | string | V32 | 调出组织编码 |  |
| content | dial\_in\_org\_num | 1 | string | V32 | 调入组织编码 |  |
| content | supplier\_num | 1 | string | V32 | 供应商编码 |  |

* 应答内容content:

| 父元素名称 | 元素名称 | 约束 | 类型 | 宽度 | 参数描述 | 备注 |
| --- | --- | --- | --- | --- | --- | --- |
| operation\_out | content | 1 | string | — | 应答内容 | XML格式的字符串 |
| content | ret\_code | 1 | struct | — | 错误码 |  |
| content | ret\_msg | 1 | struct | — | 错误描述 |  |

* 业务规则：
  + - * 数据库事务开始
      * 收到付款成功的通知后，运营支持系统将更新 调拨付款单状态为“*已付款”、*付款时间为当前时间.若调拨单关联的所有付款单状态都为“*已付款*”，将 调拨单状态 由”*待付款*”改为“*调拨中*”。
      * 营业厅实时库存表TC\_ORG\_STOCK更新：
      * 调入方更新（F\_IN\_ONWAY\_COUNT+调拨总数量，F\_IN\_ONWAY\_MONEY+调拨总金额）
      * 调出方更新（F\_GOODS\_COUNT-调拨总数量，F\_GOODS\_MONEY-调拨总金额，F\_OUT\_ONWAY\_COUNT+调拨总数量，F\_OUT\_ONWAY\_MONEY+调拨总金额）
      * 数据库事务结束
      * 通知 调拨发出方 安排线下调拨。

##### （暂不使用）付款异常补偿

1. 处理流程

运营支撑系统-〉B2B（或其他负责支付系统）

1. 功能简述

定时（每一小时，可配置）扫描表调拨付款单（select \* from TO\_ORDERDIAL\_PAY\_MAIN where status=*付款通讯异常*）;若存在记录，向B2B（或其他负责支付系统）系统查询付款状态：

若该订单不存在：调用B2B（或其他负责支付系统）付款接口，调用成功后更新调拨付款单status=2

若该订单已存在但未付款成功：更新调拨要货单status=*付款成功*；调拨单状态 由”*待付款*”改为“*执行中*”。并通知 调拨发出方 安排线下调拨。

1. 数据库表结构
2. 详细设计

* 请求内容content:
* 应答内容content:
* 业务规则：

##### （暂不使用）付款超时

1. 处理流程

运营支撑系统-〉B2B

1. 功能简述

定时（每天，可配置）扫描表调拨付款单（select \* from TO\_ORDERDIAL\_PAY\_MAIN where status=*待付款* and (当前日期-创建日期>1)）;若存在记录，向B2B系统查询付款状态：

若该订单已存在但未付款成功：更新调拨要货单status=*付款失败*；调拨单状态 由”*待付款*”改为“*付款失败*”。调拨终止。

1. 数据库表结构
2. 详细设计

* 请求内容content:
* 应答内容content:
* 业务规则：

# 逻辑组件设计

# 接口设计