**排程流程分析**

## **一、总流程图**

营销系统提供的视图V\_SALE\_ORDER\_HEAD、V\_SALE\_ORDER\_DETAIL

接口方式：营销系统给WMS系统一个账号，只能访问营销系统数据库的中间表。营销系统把订单信息更新到中间表， WMS系统定时去获取数据

先将信息存入中间主、从表

T\_SALE\_ORDER\_HEAD

T\_SALE\_ORDER\_DETAIL

客户表T\_SALE\_CUSTMER

商品表T\_SALE\_ITEM

信息**自动同步**进订单主、从表

T\_WMS\_SHIPORDER

T\_WMS\_SHIPORDERLINE

客户表T\_WMS\_CUSTMER

商品表T\_WMS\_ITEM

调用存储过程

P\_PRODUCE\_OROERRECEIVE

**检测品牌**

**新增品牌**

**通道表**

有新增

无新增

**检测件码**

无维护

**增加件码**

**提示维护**

有维护

无批次

**检测批次**

**创建批次**

有批次

**生成分拣主从表**

调用存储过程

P\_PRODUCE\_SCHEDULE EIVE

**数据效验**

数据检测异常

数据效验通过

**结束流程**

**检测包装机**

无包装机

有可用包装机

无主皮带

**检测主皮带**

有可用主皮带

**取主皮带号和包装机号**

**生成正常烟任务主、从表**

调用存储过程

p\_produce\_poke

**生成拨烟计划表**

**给合流分配出口号**

**异型烟排程**

调用存储过程

P\_UN\_SCHEDULE

调用存储过程

P\_PRODUCE\_updatesortnum

调用存储过程

P\_PRODUCE\_updateexportnum

**给分拣分配出口号**

**处理纯异型烟分拣**

给PLC电控DB块写任务

**开始分拣、合流**

**开始打件码**

**开始包装**

**分拣完成**

调用存储过程

P\_PRODUCE\_REMOV

和P\_UN\_REMOVE

**将数据存入历史表，并删除现有表**

**完成**

## **二、排程流程详细分析**

### **第一步，调用存储过程P\_PRODUCE\_OROERRECEIVE；**

#### 一、品牌检测；

关键字段说明：troughtype:分拣通道:10 重力式货架:20 皮带机:30 分拣出口(包装机):40 cigarettetype: 20:标准 88个分拣道 30:异形 60 \*2+6 40:异形烟混合道

groupno: 正常烟柜:1-8组 ,异型烟1-2组 六个异型烟烟柜 3 重力式货架 1-4

注释：异形烟混合道在每条异型烟分拣道都有。

select count(DISTINCT line.item\_id) into v\_itemcount

from t\_wms\_shiporder rr,

t\_wms\_shiporderline line

where

rr.orderno=line.orderno

AND rr.orderdate=TO\_date(p\_time,'yyyy-mm-dd') AND rr.routecode

-- To\_date 订单日期=今日 routecode 车组 p\_routestr 存储过程自动传过来的车组。

IN(p\_routestr)

and line.item\_id not in (select r.cigarettecode FROM

--item\_id 品牌ID

t\_produce\_sorttrough r

where r.cigarettecode is not null and r.state='10' and troughtype=10 and cigarettetype=20

--正常烟分拣通道的卷烟编码

union select r.cigarettecode FROM t\_produce\_sorttrough r

where r.cigarettecode is not null and r.state='10' and troughtype=10 and cigarettetype!=20

and groupno!=2);

--异型烟分拣通道的卷烟编码，groupno!=2表示的是第一组异型烟分拣通道，同理groupno!=1表示的是第二组异型烟分拣通道。

--总结：以车组对单位，查询该车组所有品牌是否有和通道表品牌ID不一样的，不一样说明要新增，一样没有新增。

if v\_itemcount>0 then

--如果v\_itemcount=0，则没有需要维护的品牌。

--取具体品牌和数量

open List\_itemtype for –-打开游标

select line.itemname,sum(line.qty)

from t\_wms\_shiporder rr,

t\_wms\_shiporderline line

where

rr.orderno=line.orderno

AND rr.orderdate=TO\_date(p\_time,'yyyy-mm-dd') AND rr.routecode IN(p\_routestr)

and line.item\_id not in (select r.cigarettecode FROM t\_produce\_sorttrough r

where r.cigarettecode is not null and r.state='10' and troughtype=10 and cigarettetype=20 --正常道

union select r.cigarettecode FROM t\_produce\_sorttrough r

where r.cigarettecode is not null and r.state='10' and troughtype=10 and cigarettetype!=20

and groupno!=2) group by line.ITEMNAME; --六个道+异形烟组一

loop

fetch List\_itemtype into tmp\_itemname,tmp\_qty;

exit when List\_itemtype%notfound;

var\_iteminfo:=var\_iteminfo||tmp\_itemname||'('||tmp\_qty||')';

--将品牌信息累计起来，按照这个格式

end loop;

close List\_itemtype;

p\_ErrCode:='0';

p\_ErrMsg:=p\_time||'有新增品牌'||var\_iteminfo||',请先更新烟道数据';

--输出这些信息。

总结：同理可以查询异形烟组二需要新增的品牌。

#### 二、件码检测；

select count(DISTINCT line.item\_id) into v\_itemcount

from t\_wms\_shiporder rr,

t\_wms\_shiporderline line

where

rr.orderno=line.orderno

AND rr.orderdate=TO\_date(p\_time,'yyyy-mm-dd') AND rr.routecode IN(p\_routestr)

and line.item\_id in (select t.itemno FROM t\_wms\_item t);

--itemno 商品代码

if v\_itemcount>0 then

--如果v\_itemcount=0，则没有需要维护的件码

open List\_itemtype for

select DISTINCT line.itemname

from t\_wms\_shiporder rr,

t\_wms\_shiporderline line

where

rr.orderno=line.orderno

AND rr.orderdate=TO\_date(p\_time,'yyyy-mm-dd') AND rr.routecode IN(p\_routestr)

and line.item\_id in (select t.itemno FROM t\_wms\_item t);

loop

fetch List\_itemtype into tmp\_itemname;

exit when List\_itemtype%notfound;

var\_iteminfo:=var\_iteminfo||tmp\_itemname||' ';

end loop;

close List\_itemtype;

p\_ErrCode:='0';

p\_ErrMsg:=p\_time||' '||var\_iteminfo||' 没有维护6位件烟码,请先更新件烟码数据';

#### 三、批次检测；

SELECT MAX(batchcode) INTO Var\_batchcode FROM t\_produce\_batch WHERE state=10 and batchtype=10;

if Var\_batchcode='' or Var\_batchcode is null then

p\_ErrCode:='0';

p\_ErrMsg:='查询不到批次号，接收订单前请先创建批次！';

rollback;

return;

end if;

总结：直接在批次表中进行查询最大批次号是否可用即可。

#### 四、生成分拣主表和从表。

##### 1、生成分拣临时表

insert into T\_PRODUCE\_ORDERID (orderid)

(

select rr.orderno

from t\_wms\_shiporder rr

where rr.orderdate=TO\_date(p\_time,'yyyy-mm-dd') AND rr.routecode IN(p\_routestr)

and rr.schedulestatus='新增' and rr.orderno not in (select billcode from t\_produce\_order)

);

-- billcode 订单号 schedulestatus 订单状态：**新增**为刚接收的订单，**排程**为已经生成了分拣表，这样区分是为了避免反复分拣同一订单。

总结：将新增的订单号按照原始车组顺序排列，插入分拣临时表。

##### 2、生成分拣主表

insert INTO t\_produce\_order

(

ID,BILLCODE,COMPANYCODE,COMPANYNAME,BATCHCODE,SYNSEQ,ORDERQUANTITY,

ORDERMONEY,CUSTOMERCODE,CUSTOMERNAME,ADDRESS,PRIORITY,REGIONCODE,

TASKNUMBERS,TASKBOXIES,STATE,UNSTATE,DEVSEQ,createtime,orderdate

)

--主要字段说明：ID，从1开始顺序生成 、BILLCODE 订单编号、BATCHCODE 排程批次、SYNSEQ 第几次接收订单、ORDERQUANTITY 订单条数、PRIORITY 按原始送货顺序，从1开始排，目前和原始送货顺序一样、REGIONCODE 车组、STATE 状态 UNSTATE 异型烟排程状态、DEVSEQ 原始送货顺序、

Createtime 当前时间、orderdate订单日期。

(select s\_produce\_order.nextval,t.\* from(

select rr.orderno,rr.shipper\_id,'' as companyname,Var\_batchcode,p\_synseq,totalqty,

0,rr.customer\_id,rr.customername,rr.realshipaddress,rr.deliveryseq,rr.routecode,

'' as tasknumbers,'' as taskboxies,'新增' as state,'新增' as unstate,rr.deliveryseq as devseq,sysdate,rr.orderdate

-- sysdate 取当前时间

from

t\_wms\_shiporder rr,

t\_produce\_orderid d

where rr.orderdate=TO\_date(p\_time,'yyyy-mm-dd') AND rr.orderno=d.orderid

--按送货顺序排序 deliveryseq 配送顺序 totalqty条数

order by rr.deliveryseq,rr.totalqty

)

);

总结：这样就按照原始发货顺序组成了分拣主表。

##### 3、生成分拣从表

INSERT INTO t\_produce\_orderline

(

billcode,linenum,cigarettecode,cigarettename,quantity,price,

unit,moeny,multiple,allowsort

)

--字段说明：订单编号、ID号、卷烟编号、卷烟名称、条数、0、单位：条、0、2、非标

注释：allowsort=非标 和 multiple=2，是先假设全部是异型烟。下面会进行正常烟和异型烟区分。

(

select line.orderno,line.id,line.item\_id,line.itemname,line.qty,0,

'条',0,2,'非标'

from t\_wms\_shiporderline line,

t\_produce\_orderid d

where line.orderno = d.orderid

);

--以分拣临时表中ID来匹配。分拣临时表中ID，插入时，必须是订单主表中有而分拣主表中没有的ID，即确定是新增的。

##### 4、区分正常烟和异型烟

open List\_itemtype for

SELECT DISTINCT cigarettecode FROM t\_produce\_sorttrough

WHERE cigarettetype＝20 and state='10' and troughtype=10;--88个烟柜中的烟，查出其中唯一值，即查出66个品牌对应的卷烟编号

Loop

fetch List\_itemtype into Var\_itemno;

exit when List\_itemtype%notfound;

--将值全部存入Var\_itemno，然后通过loop循环，一条一条通过修改语句进行修改。

update t\_produce\_orderline line

set line.multiple=1,line.allowsort='分拣'

where line.cigarettecode=Var\_itemno

AND line.billcode IN (SELECT orderid FROM T\_PRODUCE\_ORDERID);

-- multiple=1表示正常烟，2表示异型烟，插入时，默认=2，现在修改为1. allowsort='分拣'表示正常烟。

end loop;

close List\_itemtype;

##### 5、更新订单表已接收订单状态为排程，然后删除分拣临时表

UPDATE t\_wms\_shiporder rr SET rr.schedulestatus='排程'

WHERE rr.orderdate=to\_date(p\_time,'yyyy-mm-dd') AND rr.routecode IN (p\_routestr);

--清空临时表数据

delete from T\_PRODUCE\_ORDERID;

##### 6、更新送货顺序

--更新送货顺序，priority是从1开始递增排序，DEVSEQ还是原来的原始顺序

open List\_seqinfo for

select rownum as seq,billcode from(

select a.devseq,billcode from t\_produce\_order a where regioncode=p\_routestr order by a.id

);

-- devseq 订单中原有的送货顺序 rownum 是一个函数，从1开始记录行数

loop

fetch List\_seqinfo into tmp\_seq,tmp\_billcode;

exit when List\_seqinfo%notfound;

--更新送货顺序

update t\_produce\_order r set r.priority=tmp\_seq where r.billcode=tmp\_billcode;

end loop;

close List\_seqinfo;

注释：priority在生成分拣主表时，已原始送货顺序进行插入，现在使用rownum函数，从1开始依次排列。这样就能保证新的送货顺序与原始的一致。

### **第二步，调用存储过程P\_PRODUCE\_SCHEDULE**

#### 第一、数据效验；

1. 检测异型烟是否占多通道

异形烟品牌是否占完多个通道

open List\_ErrInfo for

select trim(cigarettename),count(\*) ct

--trim 去除空格函数，count(\*) ct 统计该品牌有几条通道

from t\_produce\_sorttrough h

where h.state='10' and h.cigarettetype!=20 and h.troughtype=10 and groupno!=2

--这是检测第一组异形烟+6烟柜。同理，groupno!=1，检测第二组异形烟+6烟柜

group by trim(h.cigarettename) having count(\*)>1;

loop

fetch List\_ErrInfo into tmp\_itemname,tmp\_count;

exit when List\_ErrInfo%notfound;

temp\_ErrInfo:=temp\_ErrInfo||tmp\_itemname||'('||tmp\_count||')';

p\_ErrCode:='0';

p\_ErrMsg:='品牌'||temp\_ErrInfo||'在第一组异形烟中占用了多个通道!';

rollback;

return;

end loop;

总结：如果检测有多通道，必须终止分拣，修改异常通道对应的品牌。

#### 第二、检测卷烟是否同时在正常烟道和异型烟道

open List\_ErrInfo for

--正常烟和异形烟设置成都有 cigarettename 卷烟名称

select cigarettename from t\_produce\_sorttrough s

where s.troughtype=10 and s.cigarettetype=20 and s.state='10'

and trim(s.cigarettename) in (select trim(cigarettename) from t\_produce\_sorttrough st

where st.troughtype=10 and st.cigarettetype!=20 and st.state='10' );

--字段说明：troughtype:分拣通道:10 重力式货架:20 皮带机:30 分拣出口(包装机):40

cigarettetype: 20:标准 88个道 30:异形 96 \*2+6 -1 40:异形烟混合道

groupno: 正常烟柜:1-8组 ,异型烟1-2组 六个异型烟烟柜 3 重力式货架 1-4

loop

fetch List\_ErrInfo into tmp\_itemname;

exit when List\_ErrInfo%notfound;

temp\_ErrInfo:=temp\_ErrInfo||tmp\_itemname;

p\_ErrCode:='0';

p\_ErrMsg:='品牌'||temp\_ErrInfo||'同时存在于正常道与异型烟道!';

rollback;

return;

end loop;

close List\_ErrInfo;

总结：如果检测一个品牌对应有多通道，必须终止分拣，修改异常通道对应的品牌。

#### 第三、检测包装机；

select count(\*) into tmp\_count

from t\_produce\_sorttrough h

where h.state='10' and h.troughtype=40;

if tmp\_count=0 then

p\_ErrCode:='0';

p\_ErrMsg:='无可用的包装机';

总结：如果无可用包装机，终止分拣。

#### 第四、检测主皮带；

首先，检测主皮带是否可用

select count(\*) into tmp\_count

from t\_produce\_sorttrough h

where h.state='10' and h.troughtype=30;

if tmp\_count=0 then

p\_ErrCode:='0';

p\_ErrMsg:='无可用的主皮带';

总结：如果无可用主皮带，终止分拣。

然后，检查是否有主皮带对应的两个包装机禁用了，但主皮带没有禁用，目前不允许这种设置

open List\_ErrInfo for

select groupno from t\_produce\_sorttrough s where s.state='10' and s.troughtype=30;

-- groupno 组次（正常烟柜:1-8组 ,异型烟1-2组 六个异型烟烟柜 3 重力式货架 1-4）

loop

fetch List\_ErrInfo into temp\_groupno;

exit when List\_ErrInfo%notfound;

select count(\*) into tmp\_count

from t\_produce\_sorttrough s

where s.troughtype=40 and state='10' and s.replenishline=temp\_groupno;

--在通道表中，用可用主皮带的组次来查该组次对应的包装机是否可用。

注释：通道表中数据可用通过页面或者数据库进行修改。

if tmp\_count =0 then

temp\_ErrInfo:=temp\_ErrInfo||temp\_groupno||',';

p\_ErrCode:='0';

end if;

end loop;

close List\_ErrInfo;

--这种情况是不允许的，如果有报错，必须终止分拣。

#### 第五、取主皮带和包装机号；

##### 1、取主皮带号；

--上一个车组已经用了的主皮带linenum为1，选出本车组的主皮带（顺着来）。目前暂不考虑两个包装机坏了主皮带也就不能用了

select count(\*) into tmp\_count from t\_produce\_sorttrough where

troughtype=30 and state='10' and linenum='1';

if tmp\_count=0 then—说明没有使用中的皮带。初始化的情况下，取最小可用皮带

select min(machineseq) into Var\_mainbelt from t\_produce\_sorttrough where

troughtype=30 and state='10' and linenum='0';

else

select machineseq into Var\_mainbelt from t\_produce\_sorttrough where

troughtype=30 and state='10' and linenum='1';

--这里是直接赋值Var\_mainbelt。

tmp\_count:=0; --从新赋值=0

--看有没有下一组皮带，没有就需要从头开始，有就下一个，如1皮带的下一组皮带是2，4皮带的下一组皮带是1

select count(\*) into tmp\_count from t\_produce\_sorttrough where

troughtype=30 and state='10' and machineseq>Var\_mainbelt;

--查看是否有比上面所查Var\_mainbelt还大的主皮带，如果Var\_mainbelt=4，则没有。

if tmp\_count=0 then

select min(machineseq) into Var\_mainbelt from t\_produce\_sorttrough where

troughtype=30 and state='10' ;

else –-如果Var\_mainbelt=3，则新的Var\_mainbelt=4，依次类推。

select min(machineseq) into Var\_mainbelt from t\_produce\_sorttrough where

troughtype=30 and state='10' and machineseq>Var\_mainbelt;

end if;

end if;

##### 2、取包装机号。

主皮带已经分配，看本主皮带有几台包装机可用,分配了主皮带至少有一个包装机能用,linenum,replenishline,groupno都可对应主皮带号

--包装机号赋值给temp\_spackageno，temp\_lpackageno，s-small对应左边的包装机，l-large对应右边包装机

select count(1) into tmp\_count from t\_produce\_sorttrough where TROUGHTYPE=40 and state=10 and linenum=Var\_mainbelt;

--注释：Var\_mainbelt为上面查到的主皮带号，linenum

if tmp\_count >1 then --两个包装机都可用

select machineseq into temp\_spackageno

from (select machineseq,rownum as nums from t\_produce\_sorttrough where TROUGHTYPE=40 and state=10 and linenum=Var\_mainbelt order by machineseq) where nums=1;

select machineseq into temp\_lpackageno

from (select machineseq,rownum as numl from t\_produce\_sorttrough where TROUGHTYPE=40 and state=10 and linenum=Var\_mainbelt order by machineseq ) where numl=2;

end if;

if tmp\_count =1 then --只有一个包装机，赋值相同

select machineseq into temp\_spackageno

from (select machineseq,rownum as num from t\_produce\_sorttrough where TROUGHTYPE=40 and state=10 and linenum=Var\_mainbelt order by machineseq) where num=1;

temp\_lpackageno := temp\_spackageno;

end if;

#### 第六、生成正常烟任务主、从表

##### 1、过滤只有异型烟的订单

1. 取最新批次号

SELECT MAX(batchcode) INTO Var\_batchcode FROM t\_produce\_batch WHERE state=10 and batchtype=10

--总结：定义值Var\_batchcode 为生成任务表做准备。

1. 取最大分拣号

select decode(max(priority),'',0,max(priority)) into temp\_seq from t\_produce\_task

temp\_seq:=temp\_seq+1;

--temp\_seq是按车组顺序跨车组一直排下来，+1就是取上一批次最后一个分拣顺序号的下一个。

--总结：定义值temp\_seq 为生成任务表做准备。

1. 过滤异型烟订单

select distinct a.billcode,a.orderquantity,a.id

from t\_produce\_order a,t\_produce\_orderline b

where a.billcode=b.billcode

and a.regioncode=p\_code and a.state='新增' and b.allowsort='分拣' order by a.id;

分析：.allowsort='分拣'表示正常烟，.allowsort=’非标'表示异型烟

--总结：所查的值是为生成任务表做准备，因为从任务表开始分正常烟任务表和异型烟任务表。

##### 2、生成任务主表

1. 生成任务主表；

1，将过滤异型烟的值传入；

loop

fetch List\_billcode into Var\_billcode,Var\_quantity,Var\_seqnum;

exit when List\_billcode%notfound;

2，自定义任务号，从1开始；

SELECT s\_produce\_task.nextval INTO Var\_tasknum FROM dual;

3，根据序号判断出口，序号为单数,则为编号小的包装机,序号为双数,则为编号大的包装机；

temp\_priority :=0;

temp\_priority :=temp\_priority+1;

if mod(temp\_priority,2)=1 then

temp\_packageno:= temp\_spackageno;

else

temp\_packageno:= temp\_lpackageno;

end if;

--分析：这里是按照序号来判断的，因为一条主皮带分2个订单。

4，生成任务主表

--此时priority是一个超过一个车组的顺序号了，sortseq为车组内的顺序号

--此时状态分配为新增0，等sortnum顺序号定好后置为10

--exportnum为虚拟出口号，用于和电控交互，暂时未0，在给sortnum赋值时一起赋值

Insert INTO t\_produce\_task tasknum,linenum,exportnum,regioncode,regiondesc,billcode,companycode,companyname,

batchcode,synseq,customercode,orderquantity,customername,ordermoney,taskquantity,capacity,

priority,taskbox,sortseq,lablenum,plantime,state,labelbatch,mainbelt,packagemachine

)

(select Var\_tasknum,'BS01','0',regioncode,regioncode,a.billcode,a.companycode,a.companyname,

Var\_batchcode,a.synseq,a.customercode,a.orderquantity,a.customername,a.ordermoney,a.orderquantity,

temp\_seq,temp\_seq,a.taskboxies,a.priority,'1-1',sysdate,'0',1,Var\_mainbelt,temp\_packageno）

字段说明：Var\_tasknum 从1开始到N、billcode订单编号、Var\_batchcode 最大批次号、synseq 第几次接收订单。temp\_seq 分拣序号：上面查了最大分拣号，然后+1=现在的分拣号，意思是取下次分拣的序号。priority在task表中不一定连续，有只有异型烟的订单，Var\_mainbelt 主皮带号，temp\_packageno 包装机号。

from

t\_produce\_order a

WHERE state='新增' AND a.billcode=Var\_billcode

-- Var\_billcode 这里面是只有正常烟的订单号。

);

##### 3、生成任务从表。

INSERT INTO t\_produce\_taskline

(

tasknum,cigarettecode,cigarettename,quantity,unit,allowsort

)

--Tasknum 任务号、卷烟编号、卷烟名称、条数、单位：条、非标和分拣

(

select Var\_tasknum,cigarettecode,cigarettename,quantity,unit,allowsort

ROM t\_produce\_orderline line

where billcode=Var\_billcode AND allowsort='分拣'

--排除了异形烟

);

备注：任务从表中，所有的订单已经有了一个跨车组的顺序号，从1开始排序。

--更新任务单中除掉异形烟后的数量即，只有正常烟数量

UPDATE t\_produce\_task SET taskquantity= --只有正常烟的条数

(SELECT SUM(quantity) FROM t\_produce\_taskline WHERE tasknum=Var\_tasknum)

WHERE tasknum=Var\_tasknum;

--设置当前主皮带为1,下个车组则采用当前主皮带的下一个主皮带

update t\_produce\_sorttrough set linenum='0' where troughtype=30;

update t\_produce\_sorttrough set linenum='1' where troughtype=30 and machineseq=Var\_mainbelt;

分析：上面插入的表中，皮带号都是一个值，在这里先全部清0，然后在根据最皮带号，来定义linenum='1'。

总结，这些数据插入、数据修改是在一个loop循环里面，直到全部数据插入完成，结束LOOP。

### **第三步，调用存储过程p\_produce\_poke**

#### 第一、生成拨烟计划表

##### 1、初始化pokeid

SELECT decode(MAX(pokeid),NULL,0,MAX(pokeid)) INTO var\_pokeid FROM t\_produce\_poke

--分析：查出最大的POKEID是为了排序，如果第一次分拣，var\_pokeid=0.

##### 2、取任务中卷烟明细

var\_frw :='1430204';--芙蓉王编号

var\_jbs :='1430105';--精白沙编号

open lst\_taskinfo FOR

*--未排序前，先按tasknum大小排序，*

select task.tasknum,task.taskquantity from t\_produce\_task task where task.state='0' order by task.tasknum;

--字段说明：tasknum任务号、taskquantity正常烟条数

--查看任务顺序和条数，这里规范了下面所有的poke顺序，按照发货顺序排列，

loop

fetch lst\_taskinfo into temp\_tasknum,temp\_quantity; exit when lst\_taskinfo%notfound;

*--任务号/任务数量*

*--任务中芙蓉王和精白沙的量*

*--***重点**，该游标最后面才关闭，说明以下所有查询、修改、插入都以一组（任务号、条数）来进行，然后循环将所有数据全部插入。

var\_frwqty := 0;--定义芙蓉王条数=0

select count(\*) into temp\_count from t\_produce\_taskline where cigarettecode=var\_frw and tasknum=temp\_tasknum;

if temp\_count>0 then

select quantity into var\_frwqty from t\_produce\_taskline where cigarettecode=var\_frw and tasknum=temp\_tasknum;

--注释：在任务表中，一个任务号只有一条数据，但是在任务从表中，1个任务号对应该任务需分拣的所有品牌。所有1个任务号，在从表中有多条数据。

end if;

select count(\*) into temp\_count from t\_produce\_taskline where cigarettecode=var\_jbs and tasknum=temp\_tasknum;

var\_jbsqty :=0;

if temp\_count>0 then

select quantity into var\_jbsqty from t\_produce\_taskline where cigarettecode=var\_jbs and tasknum=temp\_tasknum;

end if;

*--取任务发送所需的信息*

select task.exportnum,task.packagemachine,task.mainbelt,task.billcode

into temp\_exportnum,temp\_packagemachine,temp\_mainbelt,temp\_billcode

from t\_produce\_task task where task.tasknum=temp\_tasknum;

字段说明：虚拟出口号、包装机出口号、主皮带号、订单号

##### 3、取可用的分拣通道

temp\_i :=0;

*--找出可用的分拣通道组*

open lst\_tasklineinfo for

select 0,0,groupno

from t\_produce\_sorttrough st

where st.troughtype=10 and st.cigarettetype=20 and st.state=10

group by st.groupno;--组次

-- troughtype=10表示分拣通道，目前有8组、cigarettetype=20 表示标准烟通道

loop

fetch lst\_tasklineinfo into var\_otherqty,var\_othercount,var\_groupno;--前面2个值都是0

exit when lst\_tasklineinfo%notfound;

--该组大品牌的通道号是否可用

var\_iffrw:=0;var\_ifjbs :=0;*--*初始化

var\_frwtroughnum:=0;var\_jbstroughnum :=0;

select count(\*) into temp\_count from t\_produce\_sorttrough st where st.groupno=var\_groupno and st.cigarettecode=var\_frw and st.state=10 and st.troughtype=10 and st.cigarettetype=20;

if temp\_count>0 then—-如果<0 说明没有通道可用。

select st.state,st.troughnum into var\_iffrw,var\_frwtroughnum

from t\_produce\_sorttrough st where st.groupno=var\_groupno and st.cigarettecode=var\_frw and st.state=10 and st.troughtype=10 and st.cigarettetype=20;

end if;

---通道编号 troughnum。

select count(\*) into temp\_count from t\_produce\_sorttrough st where st.groupno=var\_groupno and st.cigarettecode=var\_jbs and st.state=10 and st.troughtype=10 and st.cigarettetype=20;

if temp\_count>0 then

select st.state,st.troughnum into var\_ifjbs,var\_jbstroughnum

from t\_produce\_sorttrough st where st.groupno=var\_groupno and st.cigarettecode=var\_jbs and st.state=10 and st.troughtype=10 and st.cigarettetype=20;

end if;

--这里是假设其他品牌每个都只抓一下，实际上有可能超过10条，就需要抓两下，

select decode(sum(line.quantity),null,0,sum(line.quantity)),decode(sum(ceil(line.quantity/10)),null,0,sum(ceil(line.quantity/10))) into var\_otherqty,var\_othercount--其他品牌数量/其他品牌抓烟次数

from t\_produce\_taskline line ,t\_produce\_sorttrough st

where line.cigarettecode=st.cigarettecode and line.cigarettecode not in (var\_frw,var\_jbs)

and st.troughtype=10 and st.cigarettetype=20 and st.state=10 and line.tasknum=temp\_tasknum

and st.groupno = var\_groupno;

if var\_othercount = 0 then *--*该组没有品牌有可能订单只有两个大品牌。

var\_otherqty :=0 ;

end if;

--分析：上面语句是为了查其他品牌卷烟的数据和抓烟次数，为合理分配芙蓉王和精白沙抓烟做准备。

##### 4、相关信息插入临时表

temp\_i :=temp\_i +1;--上面已经定义temp\_i :=0

*--相关信息插入临时表中*

insert into t\_produce\_lineinfo(ID,TASKNUM,TASKQTY,FRWQTY,JBSQTY,OTHERQTY,OTHERCOUNT,GROUPNO,

IFFRW,IFJBS,FRWDISTRUBUTEQTY,JBSDISTRUBUTEQTY,FRWSTATE,JBSSTATE,FRWDISTRUBUTECOUNT,JBSDISTRUBUTECOUNT,

FRWTROUGHNUM,JBSTROUGHNUM)

--关键字段说明：ID、任务号、总条数、芙蓉王数、白沙数、其他品牌数、抓烟数、分拣通道组次、芙蓉王通道状态、白沙通道状态、。。。。芙蓉王通道编号、白沙通道编号

values

(temp\_i,temp\_tasknum,temp\_quantity,var\_frwqty,var\_jbsqty,var\_otherqty,var\_othercount,var\_groupno,

var\_iffrw,var\_ifjbs,0,0,0,0,0,0,

var\_frwtroughnum,var\_jbstroughnum);

end loop;

close lst\_tasklineinfo;

插入临时表是为了

##### 5、分配芙蓉王分拣通道

*--*计算芙蓉王有几个十条,剩下几条。

select ceil(var\_frwqty/10),mod(var\_frwqty,10) into var\_frwcount,var\_frwmod from dual;

--函数说明：ceil 取整 mod 取余

--整除数、余数

*--*看芙蓉王能分配到几个道上

select count(\*) into temp\_count from t\_produce\_lineinfo where iffrw=10;--能用的芙蓉王道

--开始给每个通道分配

--看是芙蓉王通道多还是芙蓉抓取的次数多,循环比较小的值,最后的一个芙蓉王通道分配剩下的烟,前面的可能是整十条倍数的烟

--如芙蓉王只有抓一下的量，那就只能分配到一个通道上，循环一次。没必要每道分几条，如果芙蓉王够抓十下，八组能分，那就循环八次

whilecount := var\_frwcount;

if temp\_count<var\_frwcount then

whilecount :=temp\_count; *--取抓取次数和可用通道的最小值*

end if;

j:=0;

temp\_bigbrandqty :=var\_frwqty;*--*芙蓉王的量赋给一个临时变量，便于后面的计算

temp\_bigbrandcount := var\_frwcount; *--*芙蓉王抓取的次数赋给一个临时变量，便于后面的计算

temp\_whilecount := whilecount;*--*循环次数赋给一个临时变量，便于后面的计算

while j<whilecount loop

j:=j+1;第一次循环

*--*分配到每组上的次数，每次向上取整，要考虑最后一组是尾数

temp\_distributecount := ceil(temp\_bigbrandcount/temp\_whilecount);--需要分配的抓烟次数----用抓取次数除以最大通道数。

temp\_distributeqty := temp\_distributecount \* 10; --分配的都是抓一次满十条的,对于最后的一组可能不是整数，最后再处理

--循环一次，计算剩余量，芙蓉王的抓取次数/数量和通道数量都要减少.

temp\_bigbrandcount :=

--循环一次后，剩余的

temp\_bigbrandcount-temp\_distributecount;

temp\_bigbrandqty := temp\_b igbrandqty- temp\_distributeqty;

--剩余的总条数=以前总条数-循环抓取的

temp\_whilecount := temp\_whilecount -1;

if temp\_bigbrandqty < 0 then*--说明已经是最后一组了，按十条一次算下来已经是负数了，分配的应该是最后的散条了*

*--就是最后剩下的*

temp\_distributeqty := temp\_bigbrandqty + temp\_distributeqty;

end if;

*--每次取抓烟次数最少的组*

select groupno into temp\_groupno from

(select groupno,rownum from t\_produce\_lineinfo

where iffrw=10 and frwstate=0 order by othercount,otherqty,groupno)

where rownum=1;

分析：计算每组有芙蓉王品牌的分拣通道的其他品牌抓烟次数，并已这个次数进行排序，取抓烟次数最小的组，赋值给temp\_groupno。

*--该组芙蓉王数量计算完毕，frwstate改变*

update t\_produce\_lineinfo o set frwstate=10,o.frwdistrubutecount=temp\_distributecount,o.frwdistrubuteqty=temp\_distributeqty

temp\_distributecount 每组抓几次烟，temp\_distributeqty 每组抓烟总量

where groupno= temp\_groupno;

分析：将上面查到的第一组芙蓉王抓取值，修改进临时表中，依据最小抓烟次数对应的分拣组自修改。修改完后，因为是loop循环，所有要给下以一个组分配芙蓉王抓取量，直到8个组全部分配完毕。

end loop;

##### 6、分配精白沙分拣通道

和分配芙蓉王相似，唯一不同的是选最少吸烟次数的组的时候要加上分配的芙蓉王的次数和量。

select groupno into temp\_groupno from

(select groupno,rownum from t\_produce\_lineinfo

where ifjbs=10 and jbsstate=0 order by othercount+frwdistrubutecount,otherqty+frwdistrubuteqty,groupno)

where rownum=1;

分析，这里加上了每组芙蓉王抓烟量和抓烟次数，统计后，来计算最小抓烟分拣通道。

*--*该组精白沙数量计算完毕，jbsstate改变

update t\_produce\_lineinfo o set jbsstate=10,o.jbsdistrubutecount=temp\_distributecount,o.jbsdistrubuteqty=temp\_distributeqty

where groupno= temp\_groupno;

end loop;

##### 7、将不包含芙蓉王和精白沙的品牌信息插入POKE表

open lst\_tasklineinfo2 FORM

select line.tasknum,st.troughnum,st.machineseq,line.quantity, st.groupno

from t\_produce\_taskline line ,t\_produce\_sorttrough st,t\_produce\_task task

where line.cigarettecode=st.cigarettecode and task.tasknum=line.tasknum

and line.cigarettecode not in (var\_frw,var\_jbs)

and st.troughtype=10 and st.cigarettetype=20 and st.state=10 and task.tasknum = temp\_tasknum

ORDER BY st.troughnum;

--按通道编号排列

loop

fetch lst\_tasklineinfo2 into temp\_tasknum,temp\_troughnum,temp\_machineseq,temp\_pokenum,temp\_groupno;

exit when lst\_tasklineinfo2%notfound;

var\_pokeid:=var\_pokeid+1;

--注释：如果是第一批次分拣，var\_pokeid=0+1

INSERT INTO t\_produce\_poke(pokeid,troughnum,pokenum,machineseq,tasknum,taskqty,groupno,

sortstate,unionstate,machinestate,mainbelt,packagemachine,pokeplace,meragenum,billcode)VALUES

(var\_pokeid,temp\_troughnum,temp\_pokenum,temp\_machineseq,temp\_tasknum,temp\_quantity,temp\_groupno,

0,0,0,temp\_mainbelt,temp\_packagemachine,1,temp\_pokenum,temp\_billcode);

字段说明：顺序号、通道编号、条数、通道物理编号、任务号、任务数量、分拣通道编号、0/0/0、

主皮带号、包装机号、1、条数、订单编号

**重点**：任务号、任务数量是最上面查询到的一组数据，将这组数据全部分配完成后，在开始分配第二组任务号、任务数量。

end loop;

close lst\_tasklineinfo2;

##### 8、将芙蓉王插入POKE表

delete from t\_produce\_lineinfo o where o.frwdistrubuteqty=0 and o.jbsdistrubuteqty=0;--因已经将非芙蓉王和精白沙的插入临时表了，所有现在插入芙蓉王和精白沙要先删除。

*--*将芙蓉王插入拨烟表中

open lst\_tasklineinfofrw FOR

select o.frwdistrubuteqty,o.frwtroughnum,o.groupno

from t\_produce\_lineinfo o

where o.frwdistrubuteqty>0;

loop

fetch lst\_tasklineinfofrw into temp\_pokenum,temp\_troughnum,temp\_groupno;

--分析：吸烟数量、分拣通道编号（1-11）、分拣组次（1-8组）

exit when lst\_tasklineinfofrw%notfound;

var\_pokeid:=var\_pokeid+1;

*--取芙蓉王对应的物理通道号*

select machineseq into temp\_machineseq from t\_produce\_sorttrough h where h.troughnum = temp\_troughnum and h.cigarettetype=20 and h.troughtype=10 and h.state='10';

INSERT INTO t\_produce\_poke(pokeid,troughnum,pokenum,machineseq,tasknum,taskqty,groupno,

sortstate,unionstate,machinestate,mainbelt,packagemachine,pokeplace,meragenum,billcode)VALUES

(var\_pokeid,temp\_troughnum,temp\_pokenum,temp\_machineseq,temp\_tasknum,temp\_quantity,temp\_groupno,

0,0,0,temp\_mainbelt,temp\_packagemachine,1,temp\_pokenum,temp\_billcode);

--关键字段说明：拨烟号、通道编号、条数、通道物理号、任务号、任务数量、分拣通道编号、0/0/0、主皮带号、包装机号、1、机械手抓烟数、订单号

end loop;

close lst\_tasklineinfofrw;

为更明确该表数据关系，举列：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| POKEID | 通道编号 | 条数 | 机械手抓烟数 | 任务号 | 任务数量 | 分拣通道 | 主皮带 | 包装机 | 订单号 |
| 1 | 1 | 3 | 3 | 1 | 25 | 1 | 2 | 1 | C001 |
| 2 | 6 | 7 | 7 | 1 | 25 | 2 | 2 | 1 | C001 |
| 3 | 23 | 5 | 5 | 1 | 25 | 5 | 2 | 1 | C001 |
| 4 | 75 | 6 | 6 | 1 | 25 | 6 | 2 | 1 | C001 |
| 5 | 88 | 4 | 4 | 1 | 25 | 8 | 2 | 1 | C001 |
| 6 | 6 | 2 | 2 | 2 | 8 | 3 | 2 | 1 | C001 |

##### 9、将精白沙插入POKE表

精白沙和芙蓉王相似，这里不在列举。

*--*删除临时表

delete from t\_produce\_lineinfo;

*--*标示已经分配好了，修改任务表状态，为合流分配出口号做准备。

UPDATE t\_produce\_task SET state='10' where state='0' and tasknum=temp\_tasknum;

### **第四步，调用存储过程P\_UN\_SCHEDULE**

异型烟排程

#### 第一，前期准备，基础数据查询

SELECT decode(MAX(pokeid),NULL,1,MAX(pokeid)) INTO var\_pokeid FROM t\_un\_poke;

--查出最大pokeid，为下面插入POKE表做准备。

SELECT MAX(batchcode) INTO Var\_batchcode FROM t\_produce\_batch WHERE state=10 and batchtype=20;*--需要创建批次*

--为插入批次号做准备。

select decode(max(priority),'',0,max(priority)) into temp\_seq from t\_un\_task;

-- priority是从1开始排序，按原始顺序排的顺序数。

var\_linenum:=1;

*--var\_linenum*指分配到哪条异形烟分拣线上去分拣,一二主皮带对第一组异型烟分拣线，三四主皮带对第二组异型烟分拣线

select h.replenishline into var\_linenum from t\_produce\_sorttrough h where h.troughtype=30 and h.machineseq=p\_mainbelt;

-- p\_mainbelt 存储过程传过来的主皮带号，是按照车组顺序排列的，与正常烟分拣对应的主皮带一样。h.replenishline 主皮带对应的异型烟线，要么是1，要么是2。

分析：如果主皮带对应1/2/3/4，那么var\_linenum=1,1,，2,2

open List\_billcode for

select a.billcode,sum(b.quantity),a.id,a.orderdate

from t\_produce\_order a,t\_produce\_orderline b where a.billcode=b.billcode *--*必须该车组正常烟排完程后，才能排异形烟，不然不知道分到哪一组及包装机号

and a.regioncode=p\_code and b.allowsort='非标' and a.state='排程' and a.unstate='新增'

group by a.billcode,a.id,a.orderdate order by a.id;

loop

fetch List\_billcode into Var\_billcode,Var\_quantity,Var\_id,var\_ordate;

exit when List\_billcode%notfound;

--字段说明：异型烟订单编号、订单对应的总条数、id、订单日期

从分拣表里面，把所有的异型烟订单全部获取过来。包含了有正常烟的和只有异型烟的。

BEGIN

Var\_existrcd:=1;

*--*是否在正常烟排程中存在对应的订单

SELECT count(\*)into tmp\_count FROM t\_produce\_task t where t.billcode=Var\_billcode; --查询异型的订单数

if tmp\_count=0 then *--*在正常烟排程中找不到对应的订单（即该订单全部是异形烟）

SELECT s\_produce\_task.nextval INTO Var\_tasknum FROM dual;*--*取新的任务号，每次取值，自动加1

*--包装机根据主皮带号任意选取一个*

select machineseq into Var\_packageNo from (select machineseq,rownum as num —从1开始记录，

from t\_produce\_sorttrough where TROUGHTYPE=40 and state=10 and linenum=p\_mainbelt –-存储自带的主皮带号

order by machineseq)

--上面语句查询出的是主皮带对应的包装机物理编号 1-8号

where num=1;

--加上num=1后，即查询一条数据

Var\_existrcd:=0;

else

SELECT tasknum,t.packagemachine,t.palletnum

INTO Var\_tasknum,Var\_packageNo,var\_palletnum

FROM t\_produce\_task t where t.billcode=Var\_billcode;

end if;

--任务号、包装机出口号、托盘编号，其中包装机出口号已经在前面分配好。

temp\_seq:=temp\_seq+1;

--如果任务表中有数据，那么temp\_seq就有最大的顺序号，这样一来，下面的插入数据，就排在后面即可，反正，如果没有数据，那么temp\_seq=1

#### 第二、生成异型烟任务表和从表

*--***数据插入异形烟任务表中**

insert INTO t\_un\_task

(

tasknum,linenum,exportnum,regioncode,regiondesc,billcode,companycode,companyname,

batchcode,synseq,customercode,orderquantity,customername,ordermoney,taskquantity,orderdate,existrcd,

priority,taskbox,sortseq,lablenum,plantime,state,labelbatch,mainbelt,packagemachine,sortnum,secsortnum

)

--关键字段说明：tasknum 顺序号、linenum 对应分拣线号、exportnum虚拟出口号、regioncode车组号、billcode订单号、batchcode批次号、synseq 接第几次、orderquantity 正异型烟总条数、taskquantity异常烟条数、orderdate订单日期、existrcd 有无正常烟对应的任务、priority 跨车组顺序号、sortseq组内顺序、state状态、mainbelt主皮带号、packagemachine包装机号、sortnum分拣顺序号、secsortnum只有异型烟分拣顺序号

(select Var\_tasknum,var\_linenum,Var\_exportnum,regioncode,regioncode,a.billcode,a.companycode,a.companyname,

Var\_batchcode,a.synseq,a.customercode,a.orderquantity,a.customername,a.ordermoney,Var\_quantity,var\_ordate,Var\_existrcd,

temp\_seq,a.taskboxies,a.priority,'1-1',sysdate,'10',1,Var\_mainbelt,Var\_packageNo,0,a.priority

from

t\_produce\_order a

WHERE a.billcode=Var\_billcode

--异型烟订单号，temp\_seq是连续的顺序号，任务号不一定连续。

--**注意**：sortnum=0

);

select customercode into Var\_customercode from t\_produce\_order WHERE billcode=Var\_billcode;

**备注：生成异型烟任务从表**

INSERT INTO t\_un\_taskline

(

tasknum,cigarettecode,cigarettename,quantity,unit,allowsort

)

字段说明：任务号、卷烟编号、名称、条数、单位：条、非标

(

**select** Var\_tasknum,cigarettecode,cigarettename,quantity,unit,allowsort

FROM t\_produce\_orderline line

where billcode=Var\_billcode AND allowsort='非标'

);

--分析，这里是将所有异型烟的订单数据插入到异型烟任务主、从表中

#### 第三，生成异型烟拨烟计划表

*--***拆分任务**，**每条烟是一条记录，到时发送的时候取25条记录发送**

if var\_linenum =1 then var\_tempgroupno:=2;

else

var\_tempgroupno:=1;

end if;

分析：定义这个变量，是为了下面区分是哪条异型烟分拣组

open List\_bill\_linecode for

select line.cigarettecode,line.cigarettename,line.quantity,sort.troughnum,ord.priority –-字段说明：卷烟编号、卷烟名称、条数、通道编号、原始送货顺序

FROM t\_produce\_orderline line ,t\_produce\_sorttrough sort,t\_produce\_order ord

where line.billcode=Var\_billcode and ord.billcode=line.billcode AND allowsort='非标' and line.cigarettecode=sort.cigarettecode and sort.troughtype=10

and cigarettetype!=20 and sort.groupno!=var\_tempgroupno;--!=20是所有异形烟道，!=2是指第一组，!=1指第二组。因为有六个烟柜组为3，它即是第一组也是第二组，可以不用区分。

loop

fetch List\_bill\_linecode into Var\_cigarettecode,Var\_cigarettename,Var\_quantityline,temp\_troughnum,var\_secsortnum;

exit when List\_bill\_linecode%notfound;

while Var\_quantityline!= 0 loop

var\_pokeid:=var\_pokeid+1;

备注：上面已经查询过最大值，如果前一批次有数据在里面，则var\_pokeid:=var\_pokeid+1，否则，var\_pokeid:=0+1

INSERT INTO t\_un\_poke(pokeid,cigarettecode,customercode,troughnum,pokenum,tasknum,

billcode,taskqty,PACKAGEMACHINE,linenum,sortnum,secsortnum)

VALUES

字段说明：ID号、卷烟编号、卷烟名称、分拣通道编号（60\*2+6个）、 拨烟数、任务号、订单号、任务数量、包装机号、1或2分拣线、分拣顺序、异型烟分拣顺序（根据原始的送货顺序来）

分析：由于异型烟是已1条烟分拣来进行分配任务的，所有拨烟数、任务数量都等于1. (var\_pokeid,Var\_cigarettecode,Var\_customercode,temp\_troughnum,1,Var\_tasknum,

Var\_billcode,1,Var\_packageNo,var\_linenum,0,var\_secsortnum);

Var\_quantityline:=Var\_quantityline-1;

--分析：已总条数-1的方式来循环插入到POKE表中。

--备注：异型烟生成poke表，已分拣一条烟为一条任务。Var\_quantityline表示一个订单中一个品牌对应的分拣条数。

end loop;

end loop;

close List\_bill\_linecode;

*--更改状态位，不知t\_un\_task是否需要更改状态位，默认是10*

UPDATE t\_produce\_order SET unstate='异形烟排程' WHERE billcode=Var\_billcode;

UPDATE t\_un\_task SET taskquantity=

(SELECT SUM(quantity) FROM t\_un\_taskline WHERE tasknum=Var\_tasknum)

字段说明：taskquantity 条数

WHERE tasknum=Var\_tasknum;

END;

end loop;

close List\_billcode;

为明确拨烟计划表中数据关系，举例：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| POKEID | 卷烟名称 | 分拣通道号 | 拨烟数 | 任务号 | 订单号 | 主皮带 | 分拣线号 | 分拣顺序 | 包装机号 |
| 1 | A | 22 | 1 | 1 | C001 | 1 | 2 | 1 | 2 |
| 2 | A | 22 | 1 | 1 | C001 | 1 | 2 | 2 | 2 |
| 3 | A | 22 | 1 | 1 | C001 | 1 | 2 | 3 | 2 |
| 4 | B | 23 | 1 | 1 | C001 | 1 | 2 | 4 | 2 |
| 5 | B | 23 | 1 | 1 | C001 | 1 | 2 | 5 | 2 |
| 6 | B | 23 | 1 | 1 | C001 | 1 | 2 | 6 | 2 |

### **第五步，调用存储过程P\_PRODUCE\_updatesortnum**

给合流分配出口号

#### 第一、查看任务表状态

select count(\*) into temp\_count from t\_produce\_task where state='10';

-- state='10'表示已经生成POKE表。

if temp\_count =0 then

p\_ErrCode:='0';

p\_ErrMsg:='没有可排序的数据!';--说明POKE表中任务已经排好顺序。

return;

end if ;

--取sortnum初始值

SELECT decode(MAX(sortnum),NULL,0,MAX(sortnum)) INTO var\_sortnum FROM t\_produce\_task where to\_number(state)>=15;

--已经排程好,有可能已经处于分拣中，如果有分拣的，取最大的，即最后一组分拣的，如果没有分拣的，sortnum =0，因为初始值=0，只有在分配出口号时才会分配数据。

#### 第二、取虚拟出口号

SELECT decode(MAX(exportnum),NULL,0,MAX(exportnum)) INTO var\_exportnum FROM t\_produce\_task where to\_number(state)>=15;

-- state=15表示已经排程好，没有分拣好的是没有出口号的。Exportnum 虚拟出口号

如果今天是第一次分拣，那么exportnum=0

#### 第三、给订单关联主皮带号

前面已经根据车组分配好主皮带号了，现在是根据主皮带号，来关联2个订单。

temp\_numperroute :=2;

--定义每个车组取几个订单，为了减少合流前的摇摆次数，此值可超过1

计算有几根主皮带，并排好顺序,使用一个临时表t\_produce\_taskmainbelt

--取第一根主皮带

temp\_whilecount :=1;

while temp\_whilecount<=4 loop--最多选排在前面的四个车组看分配了那些主皮带

--先判断有无需要排序的任务

select count(\*) into temp\_count from t\_produce\_task where state='10';--**注意**：前面生成POKE表时，已经修改了这个状态=10。

if temp\_count>0 then

--如果有，取最小订单号的车组号和主皮带号，找出接下来的车组的主皮带分配顺序，1，2，3，4或2，3，4，1等等

select task.mainbelt,task.regioncode into temp\_mainbelt,temp\_regioncode from t\_produce\_task task where task.tasknum =(select min(tasknum) from t\_produce\_task where state='10');

--分析：因为下面查最小的任务号，所有这条语句只能查询出一组主皮带信息和车组信息，

select count(\*) into temp\_count from t\_produce\_taskmainbelt where mainbelt=temp\_mainbelt ;

if temp\_count=0 then --没有该主皮带号则加入临时表中

insert into t\_produce\_taskmainbelt(mainbelt,seq,regioncode) values (temp\_mainbelt,temp\_whilecount,temp\_regioncode);

--临时表中主皮带号、顺序号、车组

update t\_produce\_task set state='12' where mainbelt=temp\_mainbelt and regioncode=temp\_regioncode ;--状态为12是个临时状态，为的是上面查询最小任务号时区分开。

end if;

end if;

temp\_whilecount :=temp\_whilecount+1;

--这样循环4次，如果第一个最小的是1，就会插入以1 2 3 4主皮带信息和车组信息。

end loop;

--将状态临时置为12的订单恢复为10 state10—15表示从生成poke表到指定完成任务

update t\_produce\_task task set task.state='10' where task.state='12' and task.regioncode in (select regioncode from t\_produce\_taskmainbelt);

--此时t\_produce\_taskmainbelt应该有数据，有已4组数据

#### 第四，分配合流出口号

--循环排程中的主皮带，这里是认为排完程后不会再排一次并修改部分主皮带不可用

select count(\*) into temp\_count from t\_produce\_task where state=10;

while temp\_count >0 loop

--说明还有任务没有分配出口号

这个循环，把所有订单都进行一次如下分配出口号。

open lst\_mainbelt for

select mainbelt from t\_produce\_taskmainbelt order by seq;--按顺序，这个顺序是1-4.和主皮带号不一定一致。

loop

fetch lst\_mainbelt into temp\_mainbelt;

exit when lst\_mainbelt%notfound;

temp\_whilecount :=1;

while temp\_whilecount<= temp\_numperroute loop

--temp\_numperrout=2 一条主皮带分2个订单

select count(\*) into temp\_tt from t\_produce\_task where state='10' and mainbelt= temp\_mainbelt;

if temp\_tt > 0 then

select min(tk.tasknum) into var\_tasknum

from t\_produce\_task tk where tk.state='10' and tk.mainbelt= temp\_mainbelt;

var\_sortnum :=var\_sortnum+1;

var\_exportnum := mod(var\_sortnum,80);--合流需要八十个出口号,一个任务一个合流出口号，可以直接和sortnum绑定

if var\_exportnum =0 then

var\_exportnum :=80;

end if;

--更新任务和明细表中的sortnum，给最小的订单号分配出口号

update t\_produce\_task task set task.state=15,task.exportnum=var\_exportnum,task.sortnum = var\_sortnum where task.tasknum=var\_tasknum;

--分析：var\_exportnum为上面查询出的最大虚拟出口号，如果以前没有分拣任务，=0,。--var\_sortnum位上面查询的最大分拣排序号，以前没有分拣任务var\_sortnum =0

update t\_produce\_poke set sortnum= var\_sortnum ,sortstate=8,unionstate=10,machinestate=10,unionexportnum=var\_exportnum --排序后可以向电控发送任务了，sortnum 是按照1开始按顺序排列的

--字段说明：machinestate=0 表示机械手抓烟状态已经排好序、unionexportnum合流出口号、unionstate=10说明合流状态已经排好、**预分拣赋值sortstate =8（过渡值，为预分拣分配出口号）**

where tasknum=var\_tasknum;

--更新异形烟的顺序,对于只有异形烟的订单会更新不到

update t\_un\_task ut set ut.state=20,ut.sortnum = var\_sortnum where ut.tasknum=var\_tasknum;

update t\_un\_poke up set up.status=10,up.sortnum = var\_sortnum where up.tasknum=var\_tasknum;--已相同任务号为条件进行修改。

--给异型烟任务分配相同的任务号，和正常烟的分拣顺序号一致。

end if;

temp\_whilecount :=temp\_whilecount+1;

--分析：给第一组皮带号对应的最小任务分配一个订单后，在循环一次，在给其分配第二个订单。然后给第二组皮带号对应的最小任务分配一个订单后，在循环一次，在给其分配第二个订单，依次循环，直到全部分配完毕。

end loop;

--分析，这个循环完成后，说明，给第一组主皮带号分配2个订单已经完成。

end loop;

--这个循环完成，说明已经全部分配完成。

close lst\_mainbelt;

--看是否还有未分配sortnum的任务

select count(\*) into temp\_count from t\_produce\_task where state='10';--预留的效验判断，确保合流全部分配好出口号。

end loop;

--删除临时表

delete from t\_produce\_taskmainbelt ;

### **第六步，调用存储过程P\_PRODUCE\_updateexportnum**

#### 第一、给分拣分配出口号

##### 1、查询状态

select count(\*) into temp\_count from t\_produce\_poke where sortstate=8;

---在生成合流出口号时，将这个值赋值=8.

if temp\_count =0 then

p\_ErrCode:='0';

p\_ErrMsg:='没有可设置预分拣和合流出口号的数据!';

return;

end if ;

*--对export进行设置*

temp\_whilecount :=1;

while temp\_whilecount<=8 loop*--*共八组烟柜

SELECT decode(MAX(sortnum),NULL,0,MAX(sortnum)) INTO var\_sortnum *--找各组的上一个任务*

FROM t\_produce\_poke where sortstate>8 and groupno=temp\_whilecount;*--状态8是一个临时状态* sortnum 分拣顺序

if var\_sortnum = 0 then

var\_export :=0;-- exportnum 预分拣出口号

else

select exportnum into var\_export

from t\_produce\_poke where sortnum=var\_sortnum;*--*取各组出口号初始值，每天第一次排程，该值=0。第二次排程，才会查到这个值。

end if;

if temp\_whilecount =1 then

var\_export1 :=var\_export;

elsif temp\_whilecount =2 then

var\_export2 :=var\_export;

elsif temp\_whilecount =3 then

var\_export3 :=var\_export;

elsif temp\_whilecount =4 then

var\_export4 :=var\_export;

elsif temp\_whilecount =5 then

var\_export5 :=var\_export;

elsif temp\_whilecount =6 then

var\_export6 :=var\_export;

elsif temp\_whilecount =7 then

var\_export7 :=var\_export;

elsif temp\_whilecount =8 then

var\_export8 :=var\_export;

end if;

temp\_whilecount :=temp\_whilecount+1;

end loop;

分析：如果是第一次分拣，那么上面的所有值均=0.如果是第二次分拣，那么才会有最大值。

##### 2、分配出口号

open lst\_pokeinfo for

*--找出未排出口号的poke任务组*

select distinct sortnum,groupno from t\_produce\_poke t where sortstate=8 order by sortnum,groupno;

--字段说明：sortnum 分拣顺序号、groupno 八组分拣通道编号，这里的sortnum取唯一值，

是因为在POKE表中1个任务号对应1-8条数据。例如，

任务表数据

|  |  |  |
| --- | --- | --- |
| 任务号 | 订单号 | 数量 |
| 1 | C001 | 86 |

POKE表数据

|  |  |  |
| --- | --- | --- |
| 任务号 | 数量 | 分拣通道编号 |
| 1 | 26 | 1 |
| 1 | 15 | 3 |
| 1 | 25 | 5 |
| 1 | 12 | 7 |
| 1 | 8 | 8 |

loop

fetch lst\_pokeinfo into temp\_sortnum,temp\_groupno;

exit when lst\_pokeinfo%notfound;

*-- 出口号接着进行顺延*

if temp\_groupno = 1 then

var\_export1 :=var\_export1+1;

var\_export :=var\_export1;

elsif temp\_groupno = 2 then

var\_export2 :=var\_export2+1;

var\_export :=var\_export2;

elsif temp\_groupno = 3 then

var\_export3 :=var\_export3+1;

var\_export :=var\_export3;

elsif temp\_groupno = 4 then

var\_export4 :=var\_export4+1;

var\_export :=var\_export4;

elsif temp\_groupno = 5 then

var\_export5 :=var\_export5+1;

var\_export :=var\_export5;

elsif temp\_groupno = 6 then

var\_export6 :=var\_export6+1;

var\_export :=var\_export6;

elsif temp\_groupno = 7 then

var\_export7 :=var\_export7+1;

var\_export :=var\_export7;

elsif temp\_groupno = 8 then

var\_export8 :=var\_export8+1;

var\_export :=var\_export8;

end if;

*--*计算出口号

var\_export := mod(var\_export,40);*--*预分拣需要四十个出口号

if var\_export =0 then

var\_export :=40;

end if;

update t\_produce\_poke set sortstate=10,exportnum=var\_export *--*排序后可以向电控发送任务了

where sortnum= temp\_sortnum and groupno=temp\_groupno;

-- sortstate=10表示拍好序了。

end loop;

close lst\_pokeinfo;

##### 3、给纯异型烟分配任务号

open lst\_untaskinfo for

select untask.regioncode,tasknum from t\_un\_task untask where untask.existrcd=0 and untask.sortnum=0 order by sortnum;

--字段说明：车组、任务号

分析：sortnum=0，是因为在生成任务表时，初始值=0，然后在合流分配出口号时，只修改了订单中既有正常烟也有异型烟的值，只有异型烟的值依然=0.

loop

fetch lst\_untaskinfo into temp\_regioncode,temp\_untasknum;

exit when lst\_untaskinfo%notfound;

--车组编号、任务号

*--*向本**车组**前面找最近的sortnum号，没有则向后找本车组最近的sortnum号，都没有所有本车组只有这么一条记录，置为1

**重点：是在本车组内查找。**

select decode(MIN(tasknum),null,0,MIN(tasknum)) into temp\_nextuntask

from t\_un\_task untask where untask.regioncode=temp\_regioncode and untask.existrcd=1 and tasknum>temp\_untasknum;

--注释：existrcd=1在异型烟排程中，先确定变量existrcd=1，如果查到是纯异型烟，修改变量existrcd=0。即existrcd=1的意义是该订单有正常烟和异型烟。

select decode(MAX(tasknum),null,0,MAX(tasknum)) into temp\_lastuntask

from t\_un\_task untask where untask.regioncode=temp\_regioncode and untask.existrcd=1 and tasknum<temp\_untasknum;

if temp\_lastuntask >0 then

select sortnum into temp\_sortnum from t\_un\_task where tasknum=temp\_lastuntask;

elsif temp\_nextuntask >0 then

select sortnum into temp\_sortnum from t\_un\_task where tasknum=temp\_nextuntask;

else

temp\_sortnum :=1;

end if;

update t\_un\_task set sortnum=temp\_sortnum where tasknum= temp\_untasknum;

update t\_un\_poke set sortnum=temp\_sortnum where tasknum= temp\_untasknum; *--*排序后可以向电控发送任务了

end loop;

close lst\_untaskinfo;

分析：异型烟分拣比正常烟简单。

一条分拣任务分拣一条烟，一次执行25条分拣任务。

### **第七步，打条码和包装**

**第一、打条码**：由WMS系统将每条主皮带对应的订单写入TXT文件，压缩后，通过socket发送给打码工控机，工控机通过中续器和激光打码机进行信息交互，当条烟经过打码区时，对条烟进行打码，不区分是否是异型烟。

**第二、包装**：由WMS系统，将该包装机对应的订单信息，通过视图发送给包装机自带电脑上的数据库，包装机可以对一个订单中的正常烟和异型烟统一包装。

当正常烟线或者异型烟线挂起，可以单独包装正常烟或者异型烟。

### **第八步，调用存储过程P\_PRODUCE\_REMOVE和P\_UN\_REMOVE**

当天分拣任务完成后，先将订单主、从表，分拣主、从表，任务主、从表中数据存入历史表，然后删除订单主、从表，分拣主、从表，任务主、从表，拨烟表。这样第二天从新分拣时，就不会有任务冲突了。

#### 第一、处理正常烟各表数据

INSERT INTO t\_produce\_order\_h

(SELECT ord.\*,sysdate FROM t\_produce\_order ord);

delete from t\_produce\_order;

INSERT INTO t\_produce\_orderline\_h

(SELECT orderline.\*,sysdate FROM t\_produce\_orderline orderline);

delete from t\_produce\_orderline;

INSERT INTO t\_produce\_task\_h(SELECT task.\*,sysdate FROM t\_produce\_task task);

delete from t\_produce\_task;

INSERT INTO t\_produce\_taskline\_h (SELECT taskline.\*,sysdate FROM t\_produce\_taskline taskline);

delete from t\_produce\_taskline;

DELETE t\_produce\_poke;

*--*将分拣主皮带设置为初始状态，为第二天分拣做准备。

update t\_produce\_sorttrough set linenum='0' where troughtype=30;

#### 第二、处理异型烟各表数据

INSERT INTO t\_un\_task\_h(SELECT task.\*,sysdate FROM t\_un\_task task);

delete from t\_un\_task;

INSERT INTO t\_un\_taskline\_h(SELECT taskline.\*,sysdate FROM t\_un\_taskline taskline);

delete from t\_un\_taskline;

insert into t\_un\_order\_h(select \* from t\_un\_order);

delete from t\_un\_order;

insert into t\_un\_orderline\_h(select \* from t\_un\_orderline);

delete from t\_un\_orderline;

delete from t\_un\_poke;

**上位与电控信息交互流程简介**

**备注**：现在上位于电控信息交互方式还没完全确定下来，以下只供参考。

**机械手与电控的信息交互流程**

第一，信息写入区DB1

|  |  |  |  |
| --- | --- | --- | --- |
| 写入标志 | 已完成任务号 | 任务号 | 抓烟数量 |
| DBW0 | DBW2 | DBW6 | DBW10 |
| 0或1 | 1-65535 | 1-65535 | 1-10条 |

在给机械手下达任务时，会同时给一组22台机械手写任务。

例如：以下已5台机械手为例。

任务下发列表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1号机械手 | 1 |  | 1 | 2 |
| 2号机械手 | 1 |  | 2 | 5 |
| 3号机械手 | 1 |  | 3 | 8 |
| 4号机械手 | 1 |  | 4 | 7 |
| 5号机械手 | 1 |  | 5 | 10 |

任务接收后列表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1号机械手 | 0 |  | 1 | 2 |
| 2号机械手 | 0 |  | 2 | 5 |
| 3号机械手 | 0 |  | 3 | 8 |
| 4号机械手 | 0 |  | 4 | 7 |
| 5号机械手 | 0 |  | 5 | 10 |

任务完成列表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1号机械手 | 0 | 1 | 1 | 2 |
| 2号机械手 | 0 | 2 | 2 | 5 |
| 3号机械手 | 0 | 3 | 3 | 8 |
| 4号机械手 | 0 | 4 | 4 | 7 |
| 5号机械手 | 0 | 5 | 5 | 10 |

**说明**：上位写入任务时，在写入标志位写1，PLC读取任务后，将标志位1改成0.同时将任务送给机械手执行，执行完PLC在任务完成号这一列，写入任务完成号。上位读到后，将对应的任务状态更改为已完成，同时，将任务列表中的信息清0.

**分拣与电控的信息交互流程**

1. 信息写入区DB1

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 写入标志 | 任务号 | 出口号 | 主皮带号 | 总条数 | 1 | | 2 | | 。。。 | 11 | |
| DBW0 | DBW2 | DBW4 | DBW6 | DBW8 | DBW10 | DBW12 | DBW14 | DBW16 | 。。。 | DBW50 | DBW52 |
| 0或1 | 1-65535 | 1-40 | 1-4 | 1-11之和 | 数量 | 位置 | 数量 | 位置 | 。。。 | 数量 | 位置 |

例如：

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 写入标志 | 任务号 | 出口号 | 主皮带号 | 总条数 | 1 | | 2 | | 。。。 | 11 | |
| 1 | 135 | 1 | 2 | 32 | 9 | 2 | 8 | 1 | 0 | 15 | 5 |

**说明**:位置是指机械手的放烟位置。2表示从机械手第二条烟对应的位置开始放烟。

1. 任务完成反馈区DB101

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| DBW100 | DBW102 | DBW104 | DBW106 | DBW108 | DBW110 | DBW112 | DBW114 | DBW116 | DBW118 |
| **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** |
| DBW120 | DBW122 | DBW124 | DBW126 | DBW128 | DBW130 | DBW132 | DBW134 | DBW136 | DBW138 |
| **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** |
| DBW140 | DBW142 | DBW144 | DBW146 | DBW148 | DBW150 | DBW152 | DBW154 | DBW156 | DBW158 |
| **31** | **32** | **33** | **34** | **35** | **36** | **37** | **38** | **39** | **40** |
| DBW160 | DBW162 | DBW164 | DBW166 | DBW168 | DBW170 | DBW172 | DBW174 | DBW176 | DBW178 |

**说明**：当分拣电控给机械手下发的放烟指令完成后，这里反馈完成任务信息。

例如： DBw100对应的任务已经完成，就会在该位置写1，上位读到1后，将POKE表任务状态修改为已完成，同时修改DBw100位置为0。

1. 已完成出口号反馈区DB101

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| DBW200 | DBW202 | DBW204 | DBW206 | DBW208 | DBW210 | DBW212 | DBW214 | DBW216 | DBW218 |
| **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** |
| DBW220 | DBW222 | DBW224 | DBW226 | DBW228 | DBW230 | DBW232 | DBW234 | DBW236 | DBW238 |
| **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** |
| DBW240 | DBW242 | DBW244 | DBW246 | DBW248 | DBW250 | DBW252 | DBW254 | DBW256 | DBW258 |
| **31** | **32** | **33** | **34** | **35** | **36** | **37** | **38** | **39** | **40** |
| DBW260 | DBW262 | DBW264 | DBW266 | DBW268 | DBW270 | DBW272 | DBW274 | DBW276 | DBW278 |

**说明**：这里反馈已完成出口号信息，提供给上位以便于上位判断哪一个出口号可以下达任务。

例如：当上位查询到DBW200位置=1时，就知道该位置任务已经完成，然后可以继续给出口号DBW200下任务了。

**合流与电控的信息交互流程**

1. 信息写入区DB1

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 写入标志 | 任务号 | 出口号 | 包装机号 | 总条数 | 1 | 2 | 3 | 4 | 。。。 | 7 | 8 |
| DBW0 | DBW2 | DBW4 | DBW6 | DBW8 | DBW10 | DBW12 | DBW14 | DBW16 | 。。。 | DBW22 | DBW24 |
| 0或1 | 1-65535 | 1-80 | 1-8 | 1-8之和 | 数量 | 数量 | 数量 | 数量 | 。。。 | 数量 | 数量 |

例如：

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 写入标志 | 任务号 | 出口号 | 包装机号 | 总条数 | 1 | 2 | 3 | 4 | 。。。 | 7 | 8 |
| 1 | 121 | 2 | 2 | 65 | 12 | 15 | 25 | 3 | 0 | 4 | 6 |

**说明**：当任务下达给分拣的电控后，由分拣电控和对应的8台机械手交互信息，完成抓烟任务，这里上位没有和机械手进行任务信息交互。

1. 任务完成反馈区DB30

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| DBW0 | DBW2 | DBW4 | DBW6 | DBW8 | DBW10 | DBW12 | DBW14 | DBW16 | DBW18 |
| **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** |
| DBW20 | DBW22 | DBW24 | DBW26 | DBW28 | DBW30 | DBW32 | DBW34 | DBW36 | DBW38 |
| **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** |
| DBW40 | DBW42 | DBW44 | DBW46 | DBW48 | DBW50 | DBW52 | DBW54 | DBW56 | DBW58 |
| **31** | **32** | **33** | **34** | **35** | **36** | **37** | **38** | **39** | **40** |
| DBW60 | DBW62 | DBW64 | DBW66 | DBW68 | DBW70 | DBW72 | DBW74 | DBW76 | DBW78 |
| **41** | **42** | **43** | **44** | **45** | **46** | **47** | **48** | **49** | **50** |
| DBW80 | DBW82 | DBW84 | DBW86 | DBW88 | DBW90 | DBW92 | DBW94 | DBW96 | DBW98 |
| **51** | **52** | **53** | **54** | **55** | **56** | **57** | **58** | **59** | **60** |
| DBW100 | DBW102 | DBW104 | DBW106 | DBW108 | DBW110 | DBW112 | DBW114 | DBW116 | DBW118 |
| **61** | **62** | **63** | **64** | **65** | **66** | **67** | **68** | **69** | **70** |
| DBW120 | DBW122 | DBW124 | DBW126 | DBW128 | DBW130 | DBW132 | DBW134 | DBW136 | DBW138 |
| **71** | **72** | **73** | **74** | **75** | **76** | **77** | **78** | **79** | **80** |
| DBW140 | DBW142 | DBW144 | DBW146 | DBW148 | DBW150 | DBW152 | DBW154 | DBW156 | DBW158 |

**说明**：这里反馈已完成出口号信息，提供给上位以便于上位判断哪一个出口号可以下达任务。

例如：当上位查询到DBW0位置=1时，就知道该位置任务已经完成，然后可以继续给出口号DBW0下任务了。

1. 已完成出口号反馈区DB30

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| DBW200 | DBW202 | DBW204 | DBW206 | DBW208 | DBW210 | DBW212 | DBW214 | DBW216 | DBW218 |
| **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** |
| DBW220 | DBW222 | DBW224 | DBW226 | DBW228 | DBW230 | DBW232 | DBW234 | DBW236 | DBW238 |
| **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** |
| DBW240 | DBW242 | DBW244 | DBW246 | DBW248 | DBW250 | DBW252 | DBW254 | DBW256 | DBW258 |
| **31** | **32** | **33** | **34** | **35** | **36** | **37** | **38** | **39** | **40** |
| DBW260 | DBW262 | DBW264 | DBW266 | DBW268 | DBW270 | DBW272 | DBW274 | DBW276 | DBW278 |
| **41** | **42** | **43** | **44** | **45** | **46** | **47** | **48** | **49** | **50** |
| DBW280 | DBW282 | DBW284 | DBW286 | DBW288 | DBW290 | DBW292 | DBW294 | DBW296 | DBW298 |
| **51** | **52** | **53** | **54** | **55** | **56** | **57** | **58** | **59** | **60** |
| DBW300 | DBW302 | DBW304 | DBW306 | DBW308 | DBW310 | DBW312 | DBW314 | DBW316 | DBW318 |
| **61** | **62** | **63** | **64** | **65** | **66** | **67** | **68** | **69** | **70** |
| DBW320 | DBW322 | DBW324 | DBW326 | DBW328 | DBW330 | DBW332 | DBW334 | DBW336 | DBW338 |
| **71** | **72** | **73** | **74** | **75** | **76** | **77** | **78** | **79** | **80** |
| DBW340 | DBW342 | DBW344 | DBW346 | DBW348 | DBW350 | DBW352 | DBW354 | DBW356 | DBW358 |

**说明**：这里反馈已完成出口号信息，提供给上位以便于上位判断哪一个出口号可以下达任务。

例如：当上位查询到DBW200位置=1时，就知道该位置任务已经完成，然后可以继续给出口号DBW200下任务了。

**上位与异型烟分拣电控信息交互**

定义： A线DB30

B线DB31

分拣订单信息：分拣流水号、分拣烟道地址、尾数标志、客户号、包装机号、条烟条码、备用三个写入位、成功标志。

异型烟分拣一次下发25条任务。

**数据库字典**

说明：现在上位系统在实施过程中，可能还会修改数据库，附件是我收集的，仅供参考。

详细请见附件：分拣类数据库字典