# 大件底盘逻辑梳理

## 一、接驳相关

-- 460984 -- 每小时调度接驳车辆 在途+已到

$1 $[time(yyyyMMdd,-3d)]

$2 $[time(yyyyMMdd)]

$3 $[time(yyyy-MM-dd HH:mm:ss)]

1、//取最近3天去重。

输入：bdp.dm\_freight.op\_tp\_wl\_handover\_detail\_info 【车辆运输详情信息 kafka to hive】

逻辑：1、inc\_day 过滤近3天（T-3d,T）

2、partition by eventtype,tripid（行程id）,packageno order by eventtime 去重

3、取所有字段

输出：detailData

2、//获取已到达的汇总数据

输入：detail\_data

逻辑：1、过滤eventtype = 'ARRIVE\_END'

2、分组eventtype,deptcode,tripid

3、取min(eventtime) as event\_time, sum(packageweight) as weight, count(packageno) as ticket

输出：rs1

3、//获取在途的汇总数据

输入：detail\_data ， rs1

逻辑：1、detail\_data 过滤eventtype = 'TRANSPORT' , 关联 rs1,且 rs1.trip\_id is null，

2、分组eventtype,deptcode,tripid

3、取min(eventtime) as event\_time, sum(packageweight) as weight, count(packageno) as ticket

输出：rs2

rs1拼接rs2 ，输出rs

rs添加1列uuid。

4、//在途的事件时间+3h

输入：取 rs，eventtime 生成“plan\_arrive\_time”，“arrTm”（event\_type='TRANSPORT',则时间+3h，） ，当前日期生成“count\_date”“count\_time”。

输出：resultNoBatch

5、//读取中转到达班次信息

输入：ky.dm\_heavy\_cargo.dm\_arrive\_batch\_info\_dtl\_di【中转到达班次信息】

输出：batch\_info

逻辑：过滤T-4d~ T+5d

6、// 获取批次编码，批次日期

输入：resultNoBatch 关联 batch\_info

输出：resultAll

逻辑：根据网点关联，到达日期介于“上个班次最晚到达日期”与“最晚到达日期”之间，取batch\_code，batch\_date 。

根据 event\_type，生成，depart\_time( 在途取)

arrive\_time（到达时间）

pre\_arrive\_time（），

status = 3 = event\_type（ARRIVE\_END）

else status = 2

7、//

输入：resultAll

输出：ky.dm\_heavy\_cargo.dm\_cargo\_quantity\_handover\_dtl\_rt 【货量预测\_接驳车辆数据】

逻辑：取数写入

写入inc\_day,inc\_hour（当前天&小时）

-- 473852 每天调度\_接驳车辆 已到达 --每日计算已到达用于校验预测

输入：（ky.dm\_freight.dm\_wl\_handover\_forecast\_dtl\_df【（物联接驳货量预测流向明细表）】t1 + ky.dim\_freight.dim\_heavy\_transi\_info【顺丰实时车辆的场地列表】 b ）

left join （ky.dm\_heavy\_cargo.dm\_arrive\_batch\_info\_dtl\_di t2 + + ky.dim\_freight.dim\_heavy\_transi\_info【顺丰实时车辆的场地列表】 b）

union ky.dm\_heavy\_cargo.dm\_handover\_has\_arrive\_dtl\_di【货量预测\_接驳到达货量按到达时间分区】 t3

输出：ky.dm\_heavy\_cargo.dm\_handover\_has\_arrive\_dtl\_di

逻辑：

1-、t1 关联 b ,过滤T-1d，网点是中转场，按照trip\_id（需求id）,zone\_code 分组，取 min(depart\_time)，max(arrive\_time)，weight,计数=ticket，min(inc\_day).

时间format，生成plan\_arrive\_time，plan\_arr\_tm（+3h），arr\_tm。

2-、t2 关联b,过滤T-2D,T-D ，操作网点是中转场，取班次编码，上个班次最晚到达日期”与“最晚到达日期“，

3-、1,2关联，根据网点，根据（plan\_arr\_tm，arr\_tm 介于“上个班次最晚到达日期”与“最晚到达日期”之间）取plan\_batch\_code，plan\_batch\_date，batch\_code，batch\_date，反之取null。

4-、3根据dept\_code,trip\_id开窗，取 max(plan\_batch\_code)，max(plan\_batch\_date)， max(batch\_code)，max(batch\_date)

5-、t3过滤inc\_day T-2D,T-D.

6-、4,5union， pre\_arrive\_tim 过滤 T-2D,T-D.

-- 459747 班次基表-货量预测\_中转已到达班次

-- 459748 - 货量预测\_ 中转到达班次信息

输入： bdp.dm\_pass\_rss.sbsa\_tm\_transfer\_batch\_pro【中转班次表】 t1 left Join bdp.dm\_freight.dim\_calendar t2

输出：ky.dm\_heavy\_cargo.dm\_arrive\_batch\_info\_dtl\_di

逻辑 :

1-、t1 过滤 T-D，取有效数据 ，根据 （集散类型：1集货，2散货，3集散货）, = 2 取disperse\_last\_arrive\_tm（T2散货最晚到达时间:HH:MM） ，

反之取collect\_last\_arrive\_tm（T1集货最晚到达时间:HH:MM） 作为 last\_arrive\_tmm （T1集货最晚到达时间）、

2-、关联日期表，过滤有效时间在T-7D,T+10D 之间的数据，workday在1-7之间。

3-、生成一列last\_last\_arrive\_tm（lag(last\_arrive\_tm,1) --向前找1条 。根据操作网点分区，last\_arrive\_tm 正序） ， 兜底处理：班次生效第一天,上个班次的最晚到达时间是本班次的开始时间

-- 475426 货量预测\_班次

输入：ky.dm\_heavy\_cargo.dm\_arrive\_batch\_info\_dtl\_di

输出：ky.dm\_heavy\_cargo.dm\_transfer\_batch\_info\_dtl\_di

逻辑: 过滤 T-1D ，T+10D, 根据 。根据操作网点分区，last\_arrive\_tm 正序 ， lead(plan\_begin\_tm,1) , 当天，取下1条，拼接batch\_date，plan\_begin\_tm，，次天，则拼接batch\_date+1，plan\_begin\_tm

生成next\_begin\_tm， 同理去掉lead函数拼接 begin\_tm

再过滤 T-1D ，T+9D,

## 二、顺丰相关

-- 451091 -- 顺丰车辆实时已到达同步离线 (tidb --> hive )

-、ETL :

vt\_has\_arrive\_cars --> dm\_heavy\_cargo.ky\_has\_arrive\_cargo [$[time(yyyyMMdd)]]

-、车标去重 ：

输入：ky.dm\_heavy\_cargo.ky\_has\_arrive\_cargo

输出：ky.dm\_heavy\_cargo.ky\_has\_arrive\_cargo

逻辑： 过滤 当天 ，(partition by car\_no order by arrive\_time)

-- 247099 -- sf车辆 未发+在途+已到 【HBase】

参数1: $[time(yyyyMMdd,-60d)]

参数2: $[time(yyyyMMdd)]

参数3: $[time(yyyyMMdd,-3d)]

1、获取中转场信息

select dept\_code as deptCode from dim\_heavy\_transit\_info;

[tidb.cloud2.jdbc.url]{

jdbc.url="jdbc:mysql://fmsrms-tilb.db.sfcloud.local:3306/fmsrms?rewriteBatchedStatements=true"

user="fmsrms"

passwd="CGQMh\_Nf^BXS"

}

2、获取车辆信息前48小时车辆任务

输入：bdp.dm\_heavy\_cargo.rt\_vehicle\_task\_monitor\_for\_not\_send\_detail4 【车辆详情 kafka to hive】

输出：requireTaskDf

逻辑：过滤 参数1,T-60D ， 数据"null"处理为null，取字段。

3.取车辆任务的最新状态

1-、未发车辆计算

-、requireTaskDf 根据 需求id , partitionBy($"requireId").orderBy($"lastUpdateTm".desc) ， 开窗取最新的数据 --> new\_require\_task\_info

-、过滤 carStatus in (1,2,3,4,5) ， srcactualdeparttm（始发-实际发车时间）is null

-、 途径网点1 ，途径网点2 ，目的网点 分别关联 中转场信息， union 三者，过滤关联字段不为null， 根据 "requireId","srcZoneCode","preArriveZoneCode" 去重 （需求，始发网点，途径1,2&目的网点） -- > lastResult1

-、过滤 carStatus in (1,2,3,4,5) ， srcactualdeparttm（始发-实际发车时间）>0 , secondActualDepartTm(途径1-实际发车时间)is null

-、途径网点2，目的网点 ， 分别关联 中转场信息， union 二者，过滤关联字段，以及途径网点1不为null，根据 "requireId","srcZoneCode","preArriveZoneCode" 去重 （需求，途径网点1，途径2&目的网点） -- > lastResult2

-、过滤 carStatus in (1,2,3,4,5) ，and srcActualDepartTm >0 and secondActualDepartTm >0 and thirdActualDepartTm is null

-、目的网点，关联中转场信息，过滤关联字段，以及途径网点2不为null ，根据 "requireId","srcZoneCode","preArriveZoneCode" 去重 （需求，途径网点2，目的网点） -- > lastResult3、

-、lastResult1，2，3 union 且，srcZoneCode != preArriveZoneCode --> lr --> count1

-、mysql更新

-、更新mysql 状态（t\_monitor\_detail\_data\_process） 更新status = 1 tableName = t\_monitor\_in\_road\_cargo\_new

update t\_monitor\_detail\_data\_process set status = 1,start\_time = '"+updateTime+"',insertTime = current\_timestamp() where table\_name = '"+tableName+"'

-、lr 覆盖写入mysql

t\_monitor\_in\_road\_cargo\_new

tidb {

jdbc.url="jdbc:mysql://dmpdsp-tilb.db.sfcloud.local:3306/dmpdsp?rewriteBatchedStatements=true"

user="dmpdsp"

passwd="VjFX\_HXpA^1y"

}

2-、开始计算在途

-、requireTaskDf ， 过滤T-3D数据，需求id , partitionBy($"requireId").orderBy($"lastUpdateTm".desc) ， 开窗取最新的数据 --> new\_require\_task\_info2

2.1-、过滤 carStatus in (1,2,3,4,5) 【车辆状态(1待指派,2已指派,3待执行,4异常,5执行中,6已完成,7取消)】， lastUpdateTm（最近更新时间）> T-7D --> in\_road\_cars

2.1.1-、srcActualDepartTm（始发-实际发车时间）>0, --> in\_road\_cars\_first

secondZoneCode(途径网点1) is not null ，

secondActualArriveTm(途径网点1-实际到达时间) is null

thirdActualArriveTm(途径网点2-实际到达时间) is null

destActualArriveTm(目的网点-实际到达时间) is null

secondJobType <> '1'

2.1.1.1-、in\_road\_cars\_first ， 列传行， secondArriveContnrNos 逗号分割 --> carNo 等其他字段 -- > inFirstCarNos

同理2.1.1 , 条件不一样 分别生成，inFirstCarNos，inSecondCarNos，inThirdCarNos，inForthCarNos ， inFivethCarNos ， union 5个df， --> inAllCarNos

2.2、

2.2.1-、inAllCarNos 关联 中转场信息，过滤carNo is not null / ''，取所有 --> kyAllCarNos

-、carNo 作为rowkey查询hbase表 rt\_container\_waybill\_relation

-、匹配上

根据运单，packageMeterageWeightQty字段，获取weight（weight > 1000000D -> 1000000D,） , ticket=1L

取出： listRow += Row(requireId,carNo,translevel,carStatus,srcZoneCode,preArriveTm,preArriveZoneCode,waybillNo,ticket,weight)

-、未配上

waybillNo,ticket,weight 分别复制 null, 0 ,0

-、输出为 f1Df

-、读取mysql -- 获取快管修改车辆预计到达时间

select require\_id,zone\_code,next\_zone\_code,prologis\_in\_tm from t\_transportct\_info where prologis\_in\_tm is not null

mysql.kg {

jdbc.url="jdbc:mysql://sdmti-tilb.dbdr.sfcloud.local:3306/sdmti?useUnicode=true&characterEncoding=UTF8&autoReconnect=true&zeroDateTimeBehavior=convertToNull"

user="readonly"

passwd="sdmti#65readonly"

}

输出--> kg\_data

-、f1Df， 开窗聚合tickets，weight取第一条 --> in\_road\_total

-、关联 kg\_data ， a.requireId = b.require\_id and a.srcZoneCode = b.zone\_code and a.preArriveZoneCode = b.next\_zone\_code

生成preArriveTm(优先取 prologis\_in\_tm然后取 preArriveTm) ,status = 2

输出-- > inTotal

2.2.2-、 inAllCarNos 关联 中转场信息 ， 过滤 carNo is null 数据 --> inNotCarNos

-、in\_not\_carno\_cars 关联 kg\_data 生成preArriveTm(优先取 prologis\_in\_tm然后取 preArriveTm) ,status = 2 ， weight = 0 ,ticket=0

输出-- > inTota2

2.2.3 -、 inTotal , 2 union , srcZoneCode != preArriveZoneCode --> inTotalAll --> count2

2.2.4-、inTotalAll --> append mysql （t\_monitor\_in\_road\_cargo\_new）

3-、开始计算已到达车辆

-、读取mysql

select \* from vt\_has\_arrive\_cars -- > has\_arrive\_cars\_pre

tidb.jdbc.url=url

逻辑 ： 24),1 过滤 分区 ， T-1， 24h

-、(partition by carNo order by actualTime） 取第一条 --> has\_arrive\_cars

-、聚合tickets,weight,开窗取第一套 --> has\_arrive\_total

-、取 status = 3 ,countTime，countDate, -- > hasArriveTotal

-、补充已到达车辆没有车标的情况

-、requireTaskDf=requireTaskDf1 开窗 --> last\_day\_car\_info

-、过滤carNo is null or carNo = '' ， 根据 起始网点，途径网点1，2，到达网点的实际到达时间不同，生成hasArrive1，2，3,4,5 ， union --> has\_arrive\_total\_pre2

-、has\_arrive\_total\_pre2 关联 中转场信息 inner join ， --》 hasArriveTotal2

-、hasArriveTotal union hasArriveTotal2 --> has\_arrive\_all

-、partition by requireId,preArriveZoneCode order by weight desc 开窗取第一条，srcZoneCode != preArriveZoneCode --> hasArriveResult --> count3

-、hasArriveResult --> append mysql t\_monitor\_in\_road\_cargo\_new

-、count = count1 + count2 + count3

更新 t\_monitor\_detail\_data\_process ，status = 2

count写入 t\_monitor\_detail\_data\_row [tidb.jdbc.url]

-- 449074 -- 每30分钟刷新车标货量信息 【Hbase】

参数1:$[time(yyyy-MM-dd HH:mm:ss)]

参数2：$[time(yyyyMMdd)]

-、读取mysql

select \* from vt\_has\_arrive\_cars

tidb.jdbc.url=url

逻辑 ： 24),1 过滤 分区 ， T-1， 24h

-、读取idKey，requireId，carNo，，transLevel，carStatus，srcZoneCode，destZoneCode，actualTime

-、carNo 作为rowkey查询hbase表 rt\_container\_waybill\_relation

-、匹配上

根据运单，packageMeterageWeightQty 字段，获取weight（weight > 1000000D -> 1000000D,） , ticket=1L

listRow += Row(idKey,requireId, carNo, translevel, carStatus, srcZoneCode,destZoneCode, actualTime, waybillNo, ticket, weight)

-、未匹配上

waybillNo,ticket,weight 分别复制 null, 0 ,0

-、输出--> f1Df

-、fdDf 根据group by idKey,requireId,carNo,transLevel,carStatus,srcZoneCode,destZoneCode,actualTime 分组，求和 ticket,weight ,添加插入日期，取所有除了waybillNo

-、输出 --> result

-、写入mysql

replace into vt\_has\_arrive\_cars

-- 248838 -- 快运离线已发出货量\_场地发出货量 【Hbase】

参数1：$[time(yyyyMMdd,-24d)]

参数2：$[time(yyyyMMdd)]

参数3：$[time(yyyyMMdd)]

1-、读取mysql 获取中转场信息

select dept\_code as deptCode from dim\_heavy\_transit\_info

2-、输入:dm\_heavy\_cargo.rt\_vehicle\_task\_monitor\_for\_not\_send\_detail4

输出： require\_detail

逻辑 ： -、过滤 T-24D ~ T-D

-、始发网点，途径网点1,2 -实际发车时间 > 0

-、partition by requireId order by lastUpdateTm desc

-、carStatus in (3,4,5,6)

3-、require\_detail 内关联 中转场信息

根据 始发网点，途径网点1 ，2 ，关联（始发，途径1关联2次，过滤条件不一样） ，

取car\_nos 列转行生成 car\_no .

输出：hasArrive1 ~ 5

4-、union hasArrive1 ~ 5 --> allArrive

-、过滤 car\_no 不为空

-、遍历数据，取require\_id，translevel（运输等级），car\_no，src\_zone\_code，dest\_zone\_code，arrive\_time，send\_time

-、carNo 作为rowkey查询hbase表 rt\_container\_waybill\_relation

-、匹配上

根据运单，packageMeterageWeightQty 字段，获取weight， ticket=1L

取 listRow += Row(requireId,carNo,srcZoneCode,destZoneCode,sendTime,arriveTime,ticket,weight,waybillNo,translevel.toString)

-、未匹配上

为null

-、输出为： requireCarNoDetail ， 根据 ("waybill\_no","dest\_zone\_code")去重

-、分组聚合 ticket，weight （groupBy("require\_id","car\_no","translevel","src\_zone\_code","dest\_zone\_code","send\_time","arrive\_time")）

-、写入hive表 -->

输出： dm\_heavy\_cargo.ky\_has\_send\_cargo

逻辑 ：src\_zone\_code" =!= $"dest\_zone\_code

写入 当天分区 。

-- 470818 -- 每小时调度\_sf车辆 未发+在途+已到 加班次

参数1： $[time(yyyyMMdd,-60d)]

参数2： $[time(yyyy-MM-dd HH:mm:ss)]

参数3: $[time(yyyyMMdd,-3d)]

1-、读取mysql数据 tidb

select \* from t\_monitor\_in\_road\_cargo\_new

逻辑 ： -3D - 3D (T-6D) , preArriveTm 在 8 ~ 24点，

输出： in\_road\_datil

-、读取 in\_road\_datil 字段format ，输出 in\_road\_datil\_noBatch

2-、读取班次信息

输入： ky.dm\_heavy\_cargo.dm\_arrive\_batch\_info\_dtl\_di

输出： batch\_info

逻辑：过滤 T-3D-4D ,T-3D+5D，（T-7D，T+2D）

3-、

输入： 1,2

输出： resultAll

逻辑：1 left join 2 ,1.preArriveZoneCode=2.operate\_zone\_code ，

1的arrTm，介于“上个班次最晚到达日期”与“最晚到达日期”之间，取batch\_code，batch\_date,

partition by requireId,srcZoneCode,preArriveZoneCode order by preArriveTm 开窗取1条。

4-、输入：resultAll

输出：ky.dm\_heavy\_cargo.dm\_cargo\_quantity\_prediction\_dtl\_rt

逻辑：字段写入指定分区 ，T-3D，hour

-- 469910 -- 每天调度\_sf车辆 已到达(班次信息)--每日计算已到达用于校验预测

输入：ky.dm\_heavy\_cargo.ky\_has\_arrive\_cargo t1 left join （ ky.dm\_heavy\_cargo.dm\_arrive\_batch\_info\_dtl\_di a + ky.dim\_freight.dim\_heavy\_transi\_info b） t2

输出：ky.dm\_heavy\_cargo.dm\_has\_arrive\_cargo\_dtl\_di

逻辑： -、t1 inc\_day 过滤当天， arrive\_time 过滤 T-15D~ T+7D

-、a inc\_day 过滤 T-17D~T+9D a.operate\_zone\_code in b --> t2

-、t1 left join t2 根据 t1.dest\_zone\_code = t2.operate\_zone\_code

-、判断t1.arr\_tm 介于“上个班次最晚到达日期”与“最晚到达日期”之间）取plan\_batch\_code，plan\_batch\_date，batch\_code，batch\_date

-、partition by require\_id,car\_no,src\_zone\_code,dest\_zone\_code 开窗取第1条

-、arrive\_time 作为 inc\_day 写入 ky.dm\_heavy\_cargo.dm\_has\_arrive\_cargo\_dtl\_di

-- 473276 -- 每天调度\_sf车辆 已发出归班次 --每日计算sf已发出用于校验sf预测未发

输入：( bdp.dm\_heavy\_cargo.ky\_has\_send\_cargo a left ( ky.dm\_heavy\_cargo.dm\_transfer\_batch\_info\_dtl\_di b + ky.dim\_freight.dim\_heavy\_transi\_info c ) ) t1 union ky.dm\_heavy\_cargo.dm\_has\_send\_cargo\_dtl\_di t2

输出： ky.dm\_heavy\_cargo.dm\_has\_send\_cargo\_dtl\_di

逻辑： -、a 过滤inc\_day 当天，send\_time 在T-15D~ t+1D

-、b 过滤inc\_day T-16D，T+1D b.operate\_zone\_code in C

-、a left b on a.src\_zone\_code= b.operate\_zone\_code

-、a.send\_tm 介于 “本班次规定开始时间” 及 “下个班次规定开始时间” ，取batch\_code，batch\_date

-、partition by require\_id,car\_no,src\_zone\_code,dest\_zone\_code 开窗 ， max(batch\_code),max(batch\_date) --> t1

-、t1 union t2

-、artition by require\_id,car\_no,src\_zone\_code,dest\_zone\_code order by weight desc) 开窗，按照重量倒叙取一条， 写入 ky.dm\_heavy\_cargo.dm\_has\_send\_cargo\_dtl\_di

-- 553293

-- 555314 -- 货物预测-回刷已到达 【Hbase】

参数1：$[time(yyyyMMdd,-2d)]

参数2: $[time(yyyyMMdd)]

1-、读取mysql ，读取中转场信息

select dept\_code as deptCode from dim\_heavy\_transit\_info ，去重

输出： transit\_info\_tmp

2-、 输入： dm\_heavy\_cargo.rt\_vehicle\_task\_monitor\_for\_not\_send\_detail4

输出：require\_detail

逻辑： 过滤 T-2D，T-D ，

(secondActualArriveTm >0 or thirdActualArriveTm>0 or destActualArriveTm >0 ） 途径网点1 ，2 ，目的网点-实际到达时间 > 0

carStatus in (3,4,5,6)

partition by requireId order by lastUpdateTm desc 开窗取1条

3-、1,2关联inner join ,

途径网点1,2 分别关联，列转行生成 car\_no ,输出 hasArrive1，hasArrive2

目的网点分别关联 ， 列转行生成 car\_no ， 根据途径网点1为null ， 不为null但途径网点2为null ，途径网点2不为null，生成 hasArrive3 ~ 5

union hasArrive1~ 5 输出 --> allArrive, 过滤 car\_no 空值

4-、-、carNo 作为rowkey查询hbase表 rt\_container\_waybill\_relation

-、匹配上

根据运单，packageMeterageWeightQty字段，获取weight（weight > 1000000D -> 1000000D,） , ticket=1L

取 listRow += Row(requireId, carNo, srcZoneCode, destZoneCode, sendTime, arriveTime, ticket, weight, waybillNo, translevel.toString, carStatus.toString)

-、未匹配上

ticket, weight, waybillNo 置为 0 ，0 ， null

-、去重 "waybill\_no","dest\_zone\_code"

求和 tickets,weight (分组 "require\_id","car\_no","translevel","src\_zone\_code","dest\_zone\_code","send\_time","arrive\_time","car\_status" )

输出：result\_pre --> 添加创建日期 写入 result

5-、输入： result

输出: dm\_heavy\_cargo.ky\_has\_arrive\_cargo\_rb

逻辑： 覆盖写入，分区inc\_day = 当天日期

-- 553366 -- hive -- 已到达补刷idkey

输入： bdp.dm\_heavy\_cargo.ky\_has\_arrive\_cargo\_rb

输出： bdp.dm\_heavy\_cargo.ky\_has\_arrive\_cargo\_rb

逻辑： 重新生成id，取当天数据覆盖写入inc\_day='$[time(yyyyMMdd)]'

输出： bdp.tmp\_dm\_predict.ky\_has\_arrive\_cargo\_huishua\_tmp\_20221025

写入临时表，为etl使用

-- 553379 补数etl任务

输入： bdp.tmp\_dm\_predict.ky\_has\_arrive\_cargo\_huishua\_tmp\_20221025

输出： vt\_has\_arrive\_cars

-- 405031 etl (顺丰实时车辆的场地列表同步hive)

-、输入: dim\_heavy\_transit\_info （tidb ~ mysql ）

输出：ky.dim\_freight.dim\_heavy\_transi\_info

逻辑： 按天，

dept\_code 中转场编码

dept\_name 中转场名称

hq\_code 分拨区编码

hq\_name 分拨区名称

is\_mix\_ground 是否融合场站

max\_process\_weight 最大处理货量

## 三、顺心相关

-- 416809 -- sx\_已到达数据(旧版) --每日计算已到达用于校验预测 （顺心白名单场站到货量）

输入： ( ky.dm\_heavy\_cargo.dm\_sx\_has\_arrive\_cargo\_sum\_di a + ky.dim\_freight.dim\_heavy\_transi\_info b) t1 left ky.dm\_heavy\_cargo.ky\_has\_arrive\_cargo t2

输出： ky.dm\_heavy\_cargo.dm\_sx\_has\_arrive\_cargo\_distinct\_sum\_di

逻辑： -、a 过滤inc\_day T-60D~ T-D, a.destzonecode in b -> t1 ， 根据 partition by shiftno,ewbslistno,srczonecode,destzonecode order by createtime desc 开窗 创建时间倒叙取1条

-、t2 过滤 inc\_day T-60D~ T-D ， 去重取require\_id

-、t1.shiftno = t2.require\_id ， 取 t2.require\_id is null and t1.shiftno is not null , 写入结果表

-- 430625 -- 顺心大件场地 - 近实时计算顺心车辆基础数据\_需求id去重\_聚合

-- 431213 -- zip 大件ky场地顺心货量聚合

参数1：$[time(yyyy-MM-dd HH:mm:ss)]

参数2：dws\_sx\_vehicle\_sum

1、读取SX车辆实时进港明细

-、输入： ky.dm\_freight.dwd\_sx\_vehicle\_dtl\_di

输出： cargoInfoDf

逻辑： 过滤 T-14D ~ T-D , 按照 shift\_no(班次号) "site\_code"（当前网点代码）, "next\_site\_code"（下一个网点编码） 分组

聚合 total\_out\_weight 总重量

min 开始 卸车时间

max 标准计划到达时间

.. 其余都取最大。

in\_confirm\_time 点到时间

transport\_level 运输等级

ewblist\_send\_time 交接单发车时间

vehicle\_status 车辆状态

total\_out\_ewbcount 交接票数

total\_out\_piece 交接件数

total\_out\_vol 交接体积

2、场地过滤 快运/小件场地

读取mysql select dept\_code as deptCode from dim\_heavy\_transit\_info ， dept\_code去重

3、顺丰车辆数据过滤

读取mysql

select \* from vt\_has\_arrive\_cars

逻辑 ：过滤最近8\*24小时， actualTime 在8~ 24 h ，取 requireId 去重

输出： hasArriveTaskDf

4、 1 left join 2 2.deptCode = 1.next\_site\_code -- ,过滤场地，

left join 3 , 1.shift\_no = 3.requireId , requireId is null -- 过滤顺丰数据

取1字段。

输出： resultBatch

5、 写入tidb - dws\_sx\_vehicle\_sum

- 读取版本配置表 ： mysql pub\_table\_sync\_version （tidb） , where table\_name = dws\_sx\_vehicle\_sum , 取versionId

- 如果当前版本是1, 删除版本为2的数据 则写入数据为version=2

DELETE FROM dmpdsp.dws\_sx\_vehicle\_sum where version\_id =

-、resultBatch 按照"shift\_no", "site\_code", "next\_site\_code","version\_id" 去重 ，写入表 dws\_sx\_vehicle\_sum

- 更新版本

-- 432978 小件场地顺心车辆数据into同步hive (etl)

输入： dws\_sx\_vehicle\_sum

输出： ky.dm\_freight.dm\_sx\_vehicle\_his\_sum\_di inc\_day = $[time(yyyyMMdd)]

-- 472969 -- 近实时计算顺心车辆基础数据

-- 472962 聚合顺心车辆数据\_dm\_dwd\_sx\_vehicle\_mid

输入： ky.dm\_freight.dm\_sx\_in\_vehicle\_detail\_info\_dtl\_di

输出： ky.dm\_heavy\_cargo.dm\_dwd\_sx\_vehicle\_mid

逻辑： -、过滤 inc\_day T-7D，T-D

-、按照 group by shift\_no, site\_code, next\_site\_code 分组

聚合

sum(out\_weight) as total\_out\_weight, --总交接重量

min(first\_unlaod\_time) as first\_unlaod\_time, --开始卸车时间

max(standard\_plan\_arrival\_time) standard\_plan\_arrival\_time, --标准预计到达时间

max(in\_confirm\_time) in\_confirm\_time, --交接单点到时间

max(ewblist\_send\_time) ewblist\_send\_time, --交接单发车时间

max(vehicle\_status) vehicle\_status, --车辆状态0未发 1在途 2待卸 3卸车中 4卸车完成

max(updated\_time) updated\_time,

max(inc\_day) inc\_day1

-- 472963 聚合顺心交接单数体积dm\_sx\_vehicle\_rt\_jd\_mid

输入: ky.dm\_heavy\_cargo.sx\_rt\_jd\_sink

输出： ky.dm\_heavy\_cargo.dm\_sx\_vehicle\_rt\_jd\_mid

逻辑： -、 inc\_day 过滤 T-7D，T-D and ewbliststatus>='1'

-、 partition by ewbslistno,shiftno order by dataversion desc 开窗取1条

-、按照shiftno 分组， 聚合

sum(cast (outewbcount as bigint)) total\_out\_ewbcount, --交接票数

sum(cast (outpiece as bigint)) total\_out\_piece,--交接件数

sum(cast (outvol as decimal(32,2))) total\_out\_vol,--交接体积

shiftno

逻辑：

-、 b过滤 inc\_day T-7D，T-D and requirevendor in ('SX','SF') and transoportlevel in('1','2','3') ， requireid,transoportlevel 去重取数

-、a left b on a.shift\_no = b.requireid ,过滤 shift\_no is not null

取所有字段，case when (b.transoportlevel is null or b.transoportlevel =999) then 3 else b.transoportlevel end as transport\_level -> t1

-、t1 left t2 on t2.shiftno=t1.shift\_no , 过滤 t2.shiftno is not null and shiftno <>'';

取inc\_day及其他字段写入分区inc\_day结果表 。

-- 472976 -- 每小时调度sx车辆 在途+已到

参数1：$[time(yyyy-MM-dd HH:mm:ss)]

1、读取SX车辆实时进港明细

-、输入： ky.dm\_freight.dwd\_sx\_vehicle\_dtl\_di

输出： cargoInfoDf

逻辑： 过滤 T-14D ~ T-D , 按照 shift\_no(班次号) "site\_code"（当前网点代码）, "next\_site\_code"（下一个网点编码） 分组

聚合 total\_out\_weight 总重量

min 开始 卸车时间

max 标准计划到达时间

.. 其余都取最大。

in\_confirm\_time 点到时间

transport\_level 运输等级

ewblist\_send\_time 交接单发车时间

vehicle\_status 车辆状态

total\_out\_ewbcount 交接票数

total\_out\_piece 交接件数

total\_out\_vol 交接体积

2、场地过滤 快运/小件场地

读取mysql select dept\_code as deptCode from dim\_heavy\_transit\_info ， dept\_code去重

3、顺丰车辆数据过滤

读取mysql

select \* from vt\_has\_arrive\_cars\_batch

逻辑 ：过滤最近T-30D， actualTime ，取 requireId 去重

输出： hasArriveTaskDf

4、 1 left join 2 2.deptCode = 1.next\_site\_code -- ,过滤场地，

left join 3 , 1.shift\_no = 3.requireId , requireId is null -- 过滤顺丰数据

取1字段，格式化 pre\_arrive\_time

过滤pre\_arrive\_time between T-3D~ T+5D, 日期处理：

if( datediff(first\_unlaod\_time,FROM\_UNIXTIME(UNIX\_TIMESTAMP()))>365,if(datediff(in\_confirm\_time,FROM\_UNIXTIME(UNIX\_TIMESTAMP()))<-365,standard\_plan\_arrival\_time,in\_confirm\_time),first\_unlaod\_time ) as pre\_arrive\_time",

date\_format(if( datediff(first\_unlaod\_time,FROM\_UNIXTIME(UNIX\_TIMESTAMP()))>365,if(datediff(in\_confirm\_time,FROM\_UNIXTIME(UNIX\_TIMESTAMP()))<-365,standard\_plan\_arrival\_time,in\_confirm\_time),first\_unlaod\_time ),'yyyy-MM-dd HH:mm') as arrTm

根据 卸车时间（与当前日期对比<365 取），（点到时间（与当前日期对比，>-365 ,取），标准计划到达时间 ） ， 作为 pre\_arrive\_time

输出： sxNoBatch

5-、读取班次信息

输入： ky.dm\_heavy\_cargo.dm\_arrive\_batch\_info\_dtl\_di

输出： batch\_info

逻辑： 过滤inc\_day ， T-4D~ T+5D

6-、sxNoBatch left join batch\_info

- sxNoBatch的 arrTm ，介于“上个班次最晚到达日期”与“最晚到达日期”之间，取batch\_code，batch\_date,

- 开窗 partition by shift\_no,site\_code,next\_site\_code order by pre\_arrive\_time

vehicle\_status （0->1 , 1->2 ,else 3 ） --> status

按照 "shift\_no", "site\_code", "next\_site\_code" 去重

写入： ky.dm\_heavy\_cargo.dm\_cargo\_quantity\_sx\_vehicle\_dtl\_rt 【货量预测\_顺心车辆数据】

inc\_day = incDay ，inc\_hour = inc\_hour ，写入当天日期，当天小时

-- 473836 -- 每天调度\_sx车辆 已到达 --每日计算已到达用于校验预测

输入：（ ky.dm\_freight.dm\_sx\_vehicle\_his\_sum\_di b + (ky.dm\_heavy\_cargo.dm\_arrive\_batch\_info\_dtl\_di a1 + ky.dim\_freight.dim\_heavy\_transi\_info a2) c ） t1 union ky.dm\_heavy\_cargo.dm\_sx\_has\_arrive\_cargo\_dtl\_di t2

输出 ky.dm\_heavy\_cargo.dm\_sx\_has\_arrive\_cargo\_dtl\_di

逻辑：

-、b 过滤inc\_day当天，modify\_time当天8:20,8:00 ,vehicle\_status in('2','3','4') ,

-、根据partition by shift\_no,site\_code,next\_site\_code order by total\_out\_weight desc 开窗取1条

-、生成 arrive\_time ，arrTm ，

根据 （first\_unlaod\_time）卸车时间（与当前日期对比<365 取），反之（点到时间[in\_confirm\_time]（与当前日期对比，>-365 ,取），标准计划到达时间[standard\_plan\_arrival\_time] ） ， 作为 arrive\_time

format 作为 arrTm

-、过滤 arrive\_time in T-15D，T-D

-、输出 b

-、 a1 过滤 inc\_day ，T-16D,T+2D , operate\_zone\_code in a2 --> c

-、 b left c on b.next\_site\_code=c.operate\_zone\_code

-、 b.arrTm 介于“c.上个班次最晚到达日期”与“c.最晚到达日期”之间，取batch\_code，batch\_date

partition by shift\_no,site\_code,next\_site\_code order by total\_out\_weight desc 开窗

-、输出t1

-、 t1 union t2 (过滤 inc\_day ，T-15D~T-D)

-、 row\_number() over(partition by shift\_no,src\_zone\_code,dest\_zone\_code order by weight desc 开窗

-、输出结果表 arrive\_time = inc\_day 写入 。 ky.dm\_heavy\_cargo.dm\_sx\_has\_arrive\_cargo\_dtl\_di

## 模型输入