# 学生出勤看板

## 学习目标

了解学生出勤主题看板需求

能够编写学生出勤指标的DWD清洗转换SQL

能够编写学生出勤DWM中间层SQL

能够编写班级出勤DWM中间层SQL

能够编写班级请假DWM中间层SQL

能够编写班级旷课DWM中间层SQL

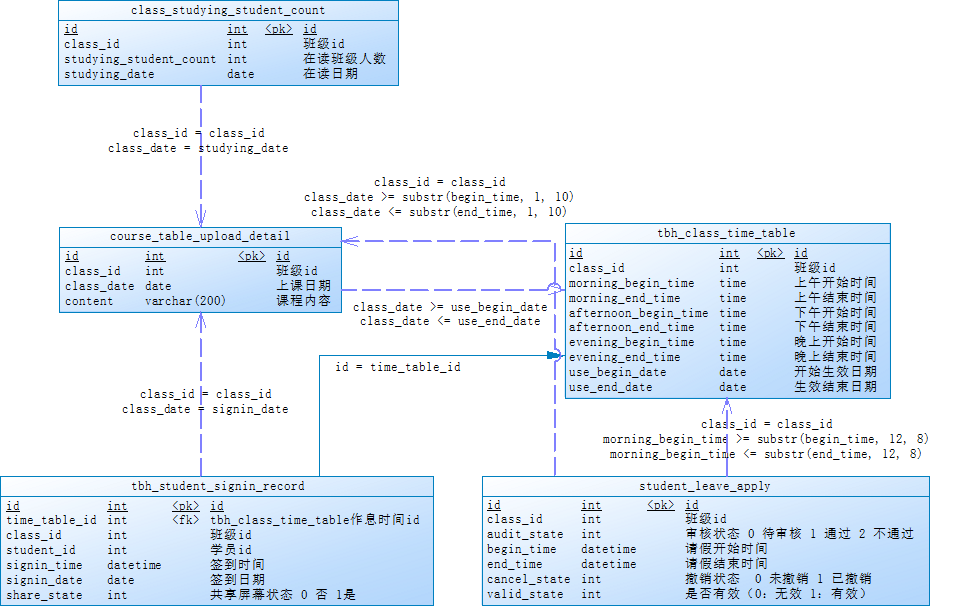
能够编写DWS出勤宽表SQL

能够编写学生出勤指标的APP应用层SQL

掌握如何导出学生出勤指标结果到Mysql

掌握增量数据分析的过程

## 主题需求



### 班级出勤人数

说明：统计指定时间段内，不同班级的出勤人数。打卡时间在上课前40分钟(否则认为无效)~上课时间点之内，且未早退，则为正常上课打卡。可以下钻到具体学生的出勤数据。跨天数据直接累加。

指标：出勤人数

维度：年、月、天

粒度：上午、下午、晚自习

条件：年、月

数据来源：教学实施与保障系统teach的course\_table\_upload\_detail班级课表、tbh\_student\_signin\_record学生打卡记录表、tbh\_class\_time\_table班级作息时间表。

### 班级出勤率

说明：统计指定时间段内，不同班级的学生出勤率。可以下钻到具体学生的出勤数据。出勤率=出勤人数/当日在读学员人数。

指标：出勤率

维度：年、月、天

粒度：上午、下午、晚自习

条件：年、月

数据来源：教学实施与保障系统的course\_table\_upload\_detail班级课表、tbh\_student\_signin\_record学生打卡记录表、tbh\_class\_time\_table班级作息时间表、class\_studying\_student\_count班级在读学生人数。

### 班级迟到人数

说明：统计指定时间段内，不同班级的迟到人数。上课10分钟后视为迟到。可以下钻到具体学生的迟到数据。跨天数据直接累加。

指标：迟到人数

维度：年、月、天

粒度：上午、下午、晚自习

条件：年、月

数据来源：教学实施与保障系统的course\_table\_upload\_detail班级课表、tbh\_student\_signin\_record学生打卡记录表、tbh\_class\_time\_table班级作息时间表。

### 班级迟到率

说明：统计指定时间段内，不同班级的学生迟到率。上课10分钟后视为迟到。可以下钻到具体学生的迟到数据。迟到率=迟到人数/当日在读学员人数。

指标：迟到率

维度：年、月、天

粒度：上午、下午、晚自习

条件：年、月

数据来源：教学实施与保障系统的course\_table\_upload\_detail班级课表、tbh\_student\_signin\_record学生打卡记录表、tbh\_class\_time\_table班级作息时间表、class\_studying\_student\_count班级在读学生人数。  
SQL：

|  |
| --- |
| select dt.every\_date,  ctud.class\_id,  tssr.student\_id,  *if*(  *#上午正常打卡为0，迟到10分钟以上为1，其他(请假+旷课)为2  sum*(  case  *#上午打卡时间是否在上课前40分钟~下课时间段之内* when *time*(tssr.signin\_time) between *TIMESTAMPADD*(minute, -40, tctt.morning\_begin\_time) and tctt.morning\_end\_time  then 1 *#上午来了* else 0 end *#上午没来* ) > 0, *#打卡多次，只要有一次正常打卡，就会>0，返回true；否则没来，返回false  if*(  *sum*(  case  *#上午打卡时间是否在上课前40分钟~上课后10分钟之内* when *time*(tssr.signin\_time) between *TIMESTAMPADD*(minute, -40, tctt.morning\_begin\_time) and *TIMESTAMPADD*(minute, 10, tctt.morning\_begin\_time)  then 1 *#正常出勤* else 0 end *#迟到* ) > 0, *#有一次打卡是正常出勤，就会>0，返回true；否则迟到，返回false* 0, *#正常出勤* 1 *#迟到* ),  2 *#上午没来* ) as morning\_signin,  *if*(  *#下午正常打卡为0，迟到10分钟以上为1，其他(请假+旷课)为2  sum*(case  when *time*(tssr.signin\_time) between *TIMESTAMPADD*(minute, -40, tctt.afternoon\_begin\_time) and tctt.afternoon\_end\_time  then 1  else 0 end) > 0,  *if*(*sum*(case  when *time*(tssr.signin\_time) between *TIMESTAMPADD*(minute, -40, tctt.afternoon\_begin\_time) and *TIMESTAMPADD*(minute, 10, tctt.afternoon\_begin\_time)  then 1  else 0 end) > 0, 0, 1), 2) as afternoon\_signin,  *if*(  *#晚自习正常打卡为0，迟到10分钟以上为1，其他(请假+旷课)为2  sum*(case  when *time*(tssr.signin\_time) between *TIMESTAMPADD*(minute, -20, tctt.evening\_begin\_time) and tctt.evening\_end\_time  then 1  else 0 end) > 0,  *if*(*sum*(case  when *time*(tssr.signin\_time) between *TIMESTAMPADD*(minute, -20, tctt.evening\_begin\_time) and *TIMESTAMPADD*(minute, 10, tctt.evening\_begin\_time)  then 1  else 0 end) > 0, 0, 1), 2) as evening\_signin from (  *#获取今天之前一周内的日期* select datelist as every\_date from calendar where datelist between '2019-09-01' and '2019-09-30'  ) dt  *#日期课表不为空且不是开班典礼* left join course\_table\_upload\_detail ctud  on ctud.class\_date = dt.every\_date and *ifnull*(ctud.content, '') != '' and  ctud.content != '开班典礼'  *#学生打卡记录日期和班级匹配，且开启共屏进入学习* left join tbh\_student\_signin\_record tssr  on tssr.class\_id = ctud.class\_id and tssr.signin\_date = dt.every\_date and  tssr.share\_state = 1  *#获取班级作息时间以判断是否按时出勤* left join tbh\_class\_time\_table tctt on tctt.id = tssr.time\_table\_id  *#按照日期、班级、学生分组统计* group by dt.every\_date, ctud.class\_id, tssr.student\_id; |

### 班级请假人数

说明：统计指定时间段内，不同班级的请假人数。跨天数据直接累加。

指标：请假人数

维度：年、月、天

粒度：上午、下午、晚自习

条件：年、月

数据来源：教学实施与保障系统的student\_leave\_apply学生请假申请表、tbh\_class\_time\_table班级作息时间表、course\_table\_upload\_detail班级课表。  
SQL：

|  |
| --- |
| select cud.class\_date as dateinfo,  cud.class\_id,  *count*(distinct sla.student\_id) as morning\_leave\_count from student\_leave\_apply sla,  tbh\_class\_time\_table ct,  course\_table\_upload\_detail cud *-- 表关联* where sla.class\_id = ct.class\_id = cud.class\_id *-- 课程表，当天有课程内容* AND cud.content IS NOT NULL  AND cud.content != '开班典礼' *-- 作息时间表，数据在生效期范围内* and cud.class\_date between ct.use\_begin\_date and ct.use\_end\_date *-- 请假表，请假状态已审核通过，且没有取消、数据有效* and sla.audit\_state = 1  and sla.cancel\_state = 0  and sla.valid\_state = 1 *-- 关联判断请假周期，请假时间周期要与课程和作息时间对比 -- cud.class\_date 课程表的上课日期 2020-09-16 -- ct.morning\_begin\_time 作息表的早上上课时间 09:00:00 -- 请假结束时间 >= 2020-09-16 09:00:00 >= 请假开始时间，认为上午请假了* and *concat*(cud.class\_date, ' ', ct.morning\_begin\_time) >= sla.begin\_time  and *concat*(cud.class\_date, ' ', ct.morning\_begin\_time) <= sla.end\_time group by cud.class\_date, cud.class\_id; |

### 班级请假率

说明：统计指定时间段内，不同班级的学生请假率。可以下钻到具体学生的请假数据。请假率=请假人数/当日在读学员人数。

指标：请假率

维度：年、月、天

粒度：上午、下午、晚自习

条件：年、月

数据来源：教学实施与保障系统的student\_leave\_apply学生请假申请表、class\_studying\_student\_count班级在读学生人数。

### 班级旷课人数

说明：统计指定时间段内，不同班级的旷课人数。跨天数据直接累加。旷课人数=当日在读学员人数-出勤人数-请假人数。

指标：旷课人数

维度：年、月、天

粒度：上午、下午、晚自习

条件：年、月

数据来源：教学实施与保障系统的course\_table\_upload\_detail班级课表、tbh\_student\_signin\_record学生打卡记录表、tbh\_class\_time\_table班级作息时间表、student\_leave\_apply学生请假申请表。

### 班级旷课率

说明：统计指定时间段内，不同班级的学生旷课率。旷课率=旷课人数/当日在读学员人数。

指标：旷课率

维度：年、月、天

粒度：上午、下午、晚自习

条件：年、月

数据来源：教学实施与保障系统的course\_table\_upload\_detail班级课表、tbh\_student\_signin\_record学生打卡记录表、tbh\_class\_time\_table班级作息时间表、student\_leave\_apply学生请假申请表、class\_studying\_student\_count班级在读学生人数。

SQL：

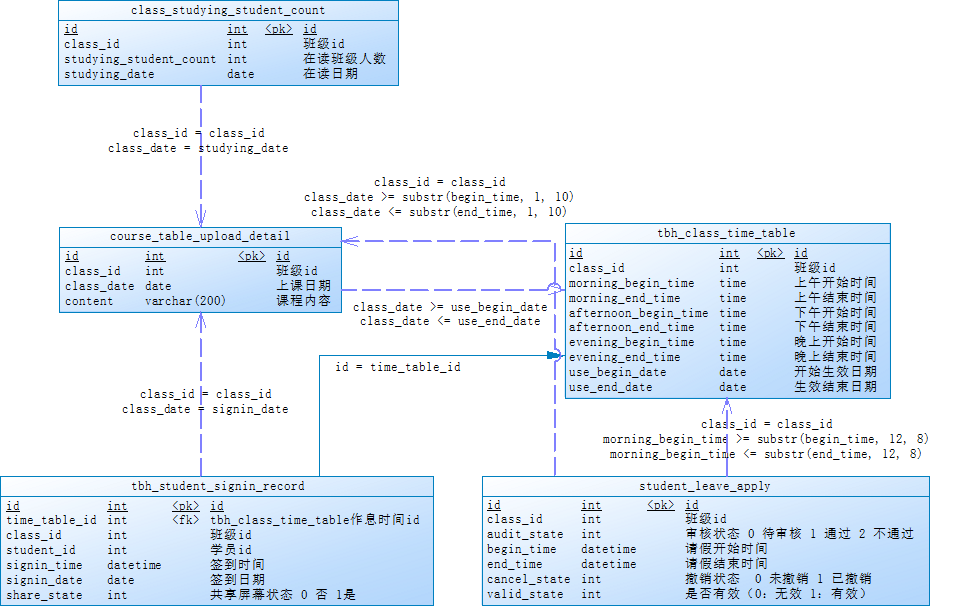
|  |
| --- |
| select *date\_format*(tmp3.every\_date, '%Y/%m/%d'),  tmp3.class\_count,  tmp3.student\_count,  tmp3.morning\_att\_count,  tmp3.morning\_late\_count,  tmp3.morning\_leave\_count,  *#减出旷课人数* (tmp3.student\_count - tmp3.morning\_att\_count - tmp3.morning\_leave\_count) as morning\_truant\_count,  *concat*(*cast*((tmp3.morning\_att\_count / tmp3.student\_count) \* 100 as decimal(8, 2)), '%') as '上午出勤率',  *concat*(*cast*((tmp3.morning\_late\_count / tmp3.student\_count) \* 100 as decimal(8, 2)), '%') as '上午迟到率',  *concat*(*cast*((tmp3.morning\_leave\_count / tmp3.student\_count) \* 100 as decimal(8, 2)), '%') as '上午请假率',  *concat*(*cast*(((tmp3.student\_count - tmp3.morning\_att\_count - tmp3.morning\_leave\_count) / tmp3.student\_count) \*  100 as decimal(8, 2)), '%') as '上午旷课率',  tmp3.afternoon\_att\_count,  tmp3.afternoon\_late\_count,  tmp3.afternoon\_leave\_count,  (tmp3.student\_count - tmp3.afternoon\_att\_count -  tmp3.afternoon\_leave\_count) as afternoon\_truant\_count,  *concat*(*cast*((tmp3.afternoon\_att\_count / tmp3.student\_count) \* 100 as decimal(8, 2)), '%') as '下午出勤率',  *concat*(*cast*((tmp3.afternoon\_late\_count / tmp3.student\_count) \* 100 as decimal(8, 2)), '%') as '下午迟到率',  *concat*(*cast*((tmp3.afternoon\_leave\_count / tmp3.student\_count) \* 100 as decimal(8, 2)), '%') as '下午请假率',  *concat*(*cast*(((tmp3.student\_count - tmp3.afternoon\_att\_count - tmp3.afternoon\_leave\_count) / tmp3.student\_count) \*  100 as decimal(8, 2)), '%') as '下午旷课率',  tmp3.evening\_att\_count,  tmp3.evening\_late\_count,  tmp3.evening\_leave\_count,  (tmp3.student\_count - tmp3.evening\_att\_count - tmp3.evening\_leave\_count) as evening\_truant\_count,  *concat*(*cast*((tmp3.evening\_att\_count / tmp3.student\_count) \* 100 as decimal(8, 2)), '%') as '晚上出勤率',  *concat*(*cast*((tmp3.evening\_late\_count / tmp3.student\_count) \* 100 as decimal(8, 2)), '%') as '晚上迟到率',  *concat*(*cast*((tmp3.evening\_leave\_count / tmp3.student\_count) \* 100 as decimal(8, 2)), '%') as '晚上请假率',  *concat*(*cast*(((tmp3.student\_count - tmp3.evening\_att\_count - tmp3.evening\_leave\_count) / tmp3.student\_count) \*  100 as decimal(8, 2)), '%') as '晚上旷课率' from (  select tmp2.every\_date,  *count*(tmp2.class\_id) as class\_count,  *sum*(tmp2.student\_count) as student\_count,  *sum*(tmp2.morning\_att\_count) as morning\_att\_count,  *sum*(tmp2.morning\_late\_count) as morning\_late\_count,  *sum*(tmp2.morning\_leave\_count) as morning\_leave\_count,  *sum*(tmp2.afternoon\_att\_count) as afternoon\_att\_count,  *sum*(tmp2.afternoon\_late\_count) as afternoon\_late\_count,  *sum*(tmp2.afternoon\_leave\_count) as afternoon\_leave\_count,  *sum*(tmp2.evening\_att\_count) as evening\_att\_count,  *sum*(tmp2.evening\_late\_count) as evening\_late\_count,  *sum*(tmp2.evening\_leave\_count) as evening\_leave\_count  from (  select tmp.every\_date,  tmp.class\_id,  *#班级人数* (select cssc.studying\_student\_count  from class\_studying\_student\_count cssc  where cssc.studying\_date = tmp.every\_date  and cssc.class\_id = tmp.**class\_id**) as student\_count,  *#上午出勤人数(包括迟到)  count*(distinct (  case  *#如果正常打卡，或者迟到，只要来了，就算到出勤人数中，返回student\_id后去重* when tmp.morning\_signin = 0 or tmp.morning\_signin = 1 then tmp.student\_id  *#如果没来，返回null，count不计算到出勤人数* else null end  )) as morning\_att\_count,  *#上午迟到人数  count*(distinct (  *#如果迟到，则返回student\_id后去重* case  when tmp.morning\_signin = 1 then tmp.student\_id  else null end  )) as morning\_late\_count,  *#下午出勤人数(包括迟到)  count*(distinct (case  when tmp.afternoon\_signin = 0 or tmp.afternoon\_signin = 1  then tmp.student\_id  else null end)) as afternoon\_att\_count,  *#下午迟到人数  count*(distinct  (case when tmp.afternoon\_signin = 1 then tmp.student\_id else null end)) as afternoon\_late\_count,  *#晚自习出勤人数(包括迟到)  count*(distinct (case  when tmp.evening\_signin = 0 or tmp.evening\_signin = 1 then tmp.student\_id  else null end)) as evening\_att\_count,  *#晚自习迟到人数  count*(distinct  (case when tmp.evening\_signin = 1 then tmp.student\_id else null end)) as evening\_late\_count,  *#上午请假学生人数，审批通过、未撤销、有效、班级匹配、请假时间在课表上课时间之内* (select *count*(distinct sla.student\_id)  from student\_leave\_apply sla  where sla.audit\_state = 1  and sla.cancel\_state = 0  and sla.valid\_state = 1  and sla.class\_id = tmp.**class\_id** and *concat*(tmp.every\_date, ' ', tctt2.**morning\_begin\_time**) >= sla.begin\_time  and *concat*(tmp.every\_date, ' ', tctt2.**morning\_begin\_time**) <=  sla.end\_time) as morning\_leave\_count,  *#下午请假学生人数，审批通过、未撤销、有效、班级匹配、请假时间在课表上课时间之内* (select *count*(distinct sla.student\_id)  from student\_leave\_apply sla  where sla.audit\_state = 1  and sla.cancel\_state = 0  and sla.valid\_state = 1  and sla.class\_id = tmp.**class\_id** and *concat*(tmp.every\_date, ' ', tctt2.**afternoon\_begin\_time**) >= sla.begin\_time  and *concat*(tmp.every\_date, ' ', tctt2.**afternoon\_begin\_time**) <=  sla.end\_time) as afternoon\_leave\_count,  *#晚自习请假学生人数，审批通过、未撤销、有效、班级匹配、请假时间在课表上课时间之内* (select *count*(distinct sla.student\_id)  from student\_leave\_apply sla  where sla.audit\_state = 1  and sla.cancel\_state = 0  and sla.valid\_state = 1  and sla.class\_id = tmp.**class\_id** and *concat*(tmp.every\_date, ' ', tctt2.**evening\_begin\_time**) >= sla.begin\_time  and *concat*(tmp.every\_date, ' ', tctt2.**evening\_begin\_time**) <=  sla.end\_time) as evening\_leave\_count  from (  *#上午、下午、晚自习：正常打卡为0，迟到10分钟以上为1，其他(请假+旷课)为2* select dt.every\_date,  ctud.class\_id,  tssr.student\_id,  *if*(  *#上午正常打卡为0，迟到10分钟以上为1，其他(请假+旷课)为2  sum*(  case  *#上午打卡时间是否在上课前40分钟~下课时间段之内* when *time*(tssr.signin\_time) between *TIMESTAMPADD*(minute, -40, tctt.morning\_begin\_time) and tctt.morning\_end\_time  then 1 *#上午来了* else 0 end *#上午没来* ) > 0, *#打卡多次，只要有一次正常打卡，就会>0，返回true；否则没来，返回false  if*(  *sum*(  case  *#上午打卡时间是否在上课前40分钟~上课后10分钟之内* when *time*(tssr.signin\_time) between *TIMESTAMPADD*(minute, -40, tctt.morning\_begin\_time) and *TIMESTAMPADD*(minute, 10, tctt.morning\_begin\_time)  then 1 *#正常出勤* else 0 end *#迟到* ) > 0, *#有一次打卡是正常出勤，就会>0，返回true；否则迟到，返回false* 0, *#正常出勤* 1 *#迟到* ),  2 *#上午没来* ) as morning\_signin,  *if*(  *#下午正常打卡为0，迟到10分钟以上为1，其他(请假+旷课)为2  sum*(case  when *time*(tssr.signin\_time) between *TIMESTAMPADD*(minute, -40, tctt.afternoon\_begin\_time) and tctt.afternoon\_end\_time  then 1  else 0 end) > 0,  *if*(*sum*(case  when *time*(tssr.signin\_time) between *TIMESTAMPADD*(minute, -40, tctt.afternoon\_begin\_time) and *TIMESTAMPADD*(minute, 10, tctt.afternoon\_begin\_time)  then 1  else 0 end) > 0, 0, 1), 2) as afternoon\_signin,  *if*(  *#晚自习正常打卡为0，迟到10分钟以上为1，其他(请假+旷课)为2  sum*(case  when *time*(tssr.signin\_time) between *TIMESTAMPADD*(minute, -20, tctt.evening\_begin\_time) and tctt.evening\_end\_time  then 1  else 0 end) > 0,  *if*(*sum*(case  when *time*(tssr.signin\_time) between *TIMESTAMPADD*(minute, -20, tctt.evening\_begin\_time) and *TIMESTAMPADD*(minute, 10, tctt.evening\_begin\_time)  then 1  else 0 end) > 0, 0, 1), 2) as evening\_signin  from (  *#获取今天之前一周内的日期* select datelist as every\_date  from calendar  where datelist between '2019-09-01' and '2019-09-30'  ) dt  *#日期课表不为空且不是开班典礼* left join course\_table\_upload\_detail ctud  on ctud.class\_date = dt.every\_date and *ifnull*(ctud.content, '') != '' and  ctud.content != '开班典礼'  *#学生打卡记录日期和班级匹配，且开启共屏进入学习* left join tbh\_student\_signin\_record tssr  on tssr.class\_id = ctud.class\_id and tssr.signin\_date = dt.every\_date and  tssr.share\_state = 1  *#获取班级作息时间以判断是否按时出勤* left join tbh\_class\_time\_table tctt on tctt.id = tssr.time\_table\_id  *#按照日期、班级、学生分组统计* group by dt.every\_date, ctud.class\_id, tssr.student\_id  ) as tmp  *#获取班级作息时间以判断是否按时出勤* left join tbh\_class\_time\_table tctt2  *#班级id相等，且作息数据当天正在生效中* on tctt2.class\_id = tmp.class\_id and tmp.every\_date >= tctt2.use\_begin\_date and  tmp.every\_date <= tctt2.use\_end\_date  *#按照日期和班级统计* group by tmp.every\_date, tmp.class\_id  ) as tmp2  *#按照日期统计* group by tmp2.every\_date  ) as tmp3; |

### 原始数据结构

**建库：**

|  |
| --- |
| create database teach default character set utf8mb4 collate utf8mb4\_unicode\_ci; |

测试数据：【Home\讲义\第8章 学生出勤主题看板\mysql测试数据】



#### 在读学员人数指标

##### class\_studying\_student\_count班级在读学生人数

根据班级和日期，查询出班级在读的学生人数。

|  |
| --- |
| create table class\_studying\_student\_count  (  id int auto\_increment  primary key,  school\_id int null comment '校区id',  subject\_id int null comment '学科id',  class\_id int null comment '**班级id**',  studying\_student\_count int null comment '**在读班级人数**',  studying\_date date null comment '**在读日期**'  )  comment '在读班级的每天在读学员人数';  create index idx\_classid\_in\_classstudyingstucount\_0  on class\_studying\_student\_count (class\_id);  create index idx\_schoolid\_in\_classstudyingstucount\_0  on class\_studying\_student\_count (school\_id);  create index idx\_studyingdate\_in\_classstudystudentcount\_4  on class\_studying\_student\_count (studying\_date);  create index idx\_subjectid\_in\_classstudyingstucount\_3  on class\_studying\_student\_count (subject\_id); |

#### 出勤人数指标

##### course\_table\_upload\_detail班级课表

用来判断班级是否有课，而且不是开班典礼。

|  |
| --- |
| create table course\_table\_upload\_detail (  id int auto\_increment comment 'id'  primary key,  base\_id int null comment '课程主表id',  class\_id int null comment '班级id',  class\_date date null comment '上课日期',  content varchar(200) null comment '课程内容',  teacher\_id int null comment '老师id',  teacher\_name varchar(20) null comment '老师名字',  job\_number varchar(45) null comment '工号',  classroom\_id int null comment '教室id',  classroom\_name varchar(50) null comment '教室名称',  is\_outline int null comment '是否大纲 0 否 1 是',  class\_mode int null comment '上课模式 0 传统全天 1 AB上午 2 AB下午 3 线上直播',  is\_stage\_exam int null comment '是否阶段考试（0：否 1：是）',  is\_pay int null comment '代课费（0：无 1：有）',  tutor\_teacher\_id int null comment '晚自习辅导老师id',  tutor\_teacher\_name varchar(20) null comment '辅导老师姓名',  tutor\_job\_number varchar(45) null comment '晚自习辅导老师工号',  is\_subsidy int null comment '晚自习补贴（0：无 1：有）',  answer\_teacher\_id int null comment '答疑老师id',  answer\_teacher\_name varchar(20) null comment '答疑老师姓名',  answer\_job\_number varchar(45) null comment '答疑老师工号',  remark varchar(2000) null comment '备注',  create\_time datetime null comment '创建时间' )  comment '班级课表明细表';  create index idx\_tutorueacheranddatesubsidy\_in\_ctud\_4  on course\_table\_upload\_detail (tutor\_teacher\_id, class\_date, is\_subsidy);  create index idx\_tutorueacheranddatesubsidy\_in\_ctud\_5  on course\_table\_upload\_detail (class\_date, tutor\_teacher\_id, is\_subsidy);  create index index\_coursetableupload\_classId\_2  on course\_table\_upload\_detail (class\_id);  create index index\_coursetableupload\_classId\_classDate\_6  on course\_table\_upload\_detail (class\_id, class\_date);  create index index\_coursetableupload\_jobNumber\_7  on course\_table\_upload\_detail (job\_number);  create index index\_coursetableuploaddetail\_answer\_teacher\_id\_7  on course\_table\_upload\_detail (answer\_teacher\_id);  create index index\_coursetableuploaddetail\_teacher\_id\_1  on course\_table\_upload\_detail (teacher\_id);  create index index\_coursetupload\_2\_baseid\_0  on course\_table\_upload\_detail (base\_id); |

##### tbh\_student\_signin\_record学生打卡记录表

用来判断学生是否开启共屏进入学习。

|  |
| --- |
| create table tbh\_student\_signin\_record (  id int auto\_increment comment '主键id'  primary key,  normal\_class\_flag int null comment '是否正课 1 正课 2 自习 3 休息',  time\_table\_id int null comment '作息时间id normal\_class\_flag=2 关联tbh\_school\_time\_table 或者 normal\_class\_flag=1 关联 tbh\_class\_time\_table',  class\_id int null comment '班级id',  student\_id int null comment '学员id',  signin\_time datetime null comment '签到时间',  signin\_date date null comment '签到日期',  inner\_flag int null comment '内外网标志 0 外网 1 内网',  signin\_type int null comment '签到类型 1 心跳打卡 2 老师补卡 3 直播打卡',  share\_state int default 0 null comment '共享屏幕状态 0 否 1是 在上午或下午段有共屏记录，则该段所有记录该字段为1，内网默认为1 外网默认为0 (暂不用)',  inner\_ip varchar(20) null comment '内网ip地址',  create\_time datetime null comment '创建时间' );  create index idx\_sharestateinnerflag\_in\_tbhstusigninrecord\_2  on tbh\_student\_signin\_record (share\_state, inner\_flag);  create index idx\_signindate\_in\_tbhstusigninrecord\_4  on tbh\_student\_signin\_record (signin\_date);  create index idx\_signintime\_in\_tbhstusigninrecord\_1  on tbh\_student\_signin\_record (signin\_time);  create index idx\_timetableid\_in\_tbhsigninrecord\_3  on tbh\_student\_signin\_record (time\_table\_id);  create index index\_student\_signin\_record\_class\_student\_id  on tbh\_student\_signin\_record (class\_id, student\_id, inner\_ip); |

##### tbh\_class\_time\_table班级作息时间表

用来获取班级作息时间以判断学生是否按时出勤。

|  |
| --- |
| create table tbh\_class\_time\_table (  id int auto\_increment comment '主键id'  primary key,  class\_id int null comment '班级id',  morning\_template\_id int null comment '上午出勤模板id',  morning\_begin\_time time null comment '上午开始时间',  morning\_end\_time time null comment '上午结束时间',  afternoon\_template\_id int null comment '下午出勤模板id',  afternoon\_begin\_time time null comment '下午开始时间',  afternoon\_end\_time time null comment '下午结束时间',  evening\_template\_id int null comment '晚上出勤模板id',  evening\_begin\_time time null comment '晚上开始时间',  evening\_end\_time time null comment '晚上结束时间',  use\_begin\_date date null comment '使用开始日期(开始生效时间)',  use\_end\_date date null comment '使用结束日期(生效结束时间)',  create\_time datetime null comment '创建时间',  create\_person int null comment '创建人',  remark varchar(500) null comment '备注' );  create index idx\_classid\_state\_in\_classtimetable\_0  on tbh\_class\_time\_table (class\_id); |

#### 迟到人数指标

同出勤人数指标。

#### 请假人数指标

##### student\_leave\_apply学生请假申请表

用来获取请假学生，条件：审批通过、未撤销、有效、班级匹配、请假时间包含了课表上课时间。

|  |
| --- |
| create table student\_leave\_apply  (  id int auto\_increment  primary key,  class\_id int null comment '**班级id**',  student\_id int null comment '学员id',  audit\_state int default 0 null comment '**审核状态** 0 待审核 1 通过 2 不通过',  audit\_person int null comment '审核人',  audit\_time datetime null comment '审核时间',  audit\_remark varchar(500) null comment '审核备注',  leave\_type int null comment '请假类型 1 请假 2 销假 （查询是否请假不用过滤此类型，通过有效状态来判断）',  leave\_reason int null comment '请假原因 1 事假 2 病假',  begin\_time datetime null comment '**请假开始时间**',  begin\_time\_type int null comment '1：上午 2：下午 3：晚自习',  end\_time datetime null comment '**请假结束时间**',  end\_time\_type int null comment '1：上午 2：下午 3：晚自习',  days float null comment '请假/已休天数',  cancel\_state int default 0 null comment '**撤销状态** 0 未撤销 1 已撤销',  cancel\_time datetime null comment '撤销时间',  old\_leave\_id int null comment '原请假id，只有leave\_type =2 销假的时候才有',  leave\_remark varchar(500) null comment '请假/销假说明',  valid\_state int default 1 null comment '**是否有效**（0：无效 1：有效）',  create\_time datetime null comment '创建时间'  );  create index index\_student\_leave\_apply\_begin\_time\_1  on student\_leave\_apply (begin\_time);  create index index\_student\_leave\_apply\_class\_student\_id  on student\_leave\_apply (class\_id, student\_id);  create index index\_student\_leave\_apply\_end\_time\_2  on student\_leave\_apply (end\_time); |

#### 旷课人数指标

旷课人数是通过之前的指标结果计算出来的。

旷课人数=班级在读学生人数 - 出勤人数（包括迟到）- 旷课人数。

因此只要前面的指标统计出来，旷课人数的结果就有了。

## 建模分析

### 指标和维度

从需求来看，这个主题下的指标和维度是非常清晰的。

指标：

* 出勤人数、出勤率
* 迟到人数、迟到率
* 请假人数、请假率
* 旷课人数、旷课率

1. 出勤人数指标和出勤率指标，其中出勤率=出勤人数/当日在读学员人数，分子出勤人数重叠，只需计算一次。只需要在此基础上统计分母-在读学员人数即可。共提炼出两个指标：出勤人数指标、在读学员人数指标。
2. 迟到人数指标和迟到率指标，迟到率=迟到人数/当日在读学员人数。和出勤数据一样，可提炼出两个指标：迟到人数指标、在读学员人数指标。
3. 请假人数指标和请假率指标，请假率=请假人数/当日在读学员人数。可提炼出两个指标：请假人数指标、在读学员人数指标。
4. 旷课人数指标和旷课率指标，旷课率=旷课人数/当日在读学员人数。可提炼出两个指标：旷课人数指标、旷课学员人数指标。

由此我们可以看出，任何一组数据都包含有在读学员人数指标，此指标是共用的。最终提炼出的指标为：

* 在读学员人数指标
* 出勤人数指标
* 迟到人数指标
* 请假人数指标
* 旷课人数指标

这些指标的维度也是一致的：年、月、天。

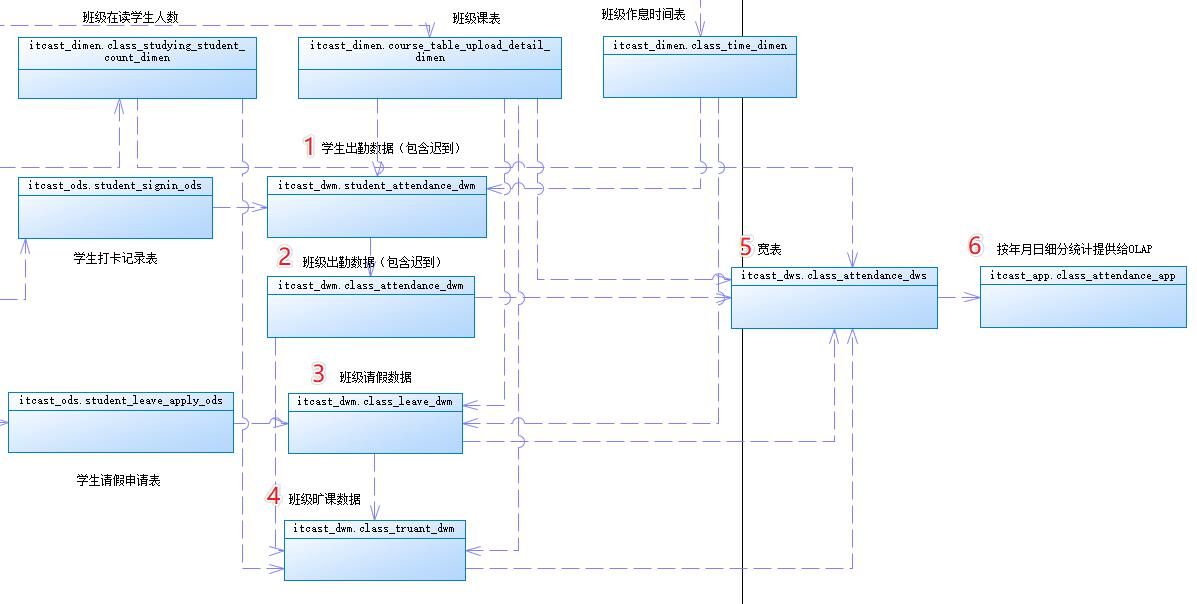
但需求要求这些指标的粒度为每天中的三个上课时间段：上午、下午、晚自习。因此维度需要增加：上午、下午、晚自习。

每个指标统计的都是班级数据，而且需求要求可以下钻到具体学生的出勤数据。因此维度需要增加：班级、学生。

最终提炼出的维度包括：

* 时间维度：年、月、天
* 时间段维度：上午、下午、晚自习
* 组织维度：班级、学生

### 分层设计



出勤人数和迟到人数的数据来源是一致的：course\_table\_upload\_detail班级课表、tbh\_student\_signin\_record学生打卡记录表、tbh\_class\_time\_table班级作息时间表；

请假人数的数据来源是：student\_leave\_apply学生请假申请表；

旷课人数的数据来源是：course\_table\_upload\_detail班级课表、tbh\_student\_signin\_record学生打卡记录表、tbh\_class\_time\_table班级作息时间表、student\_leave\_apply学生请假申请表。

通过观察我们可以看出，出勤人数和迟到人数指标的数据来源是相同的，也就是说这两个指标是有复用数据的；

请假人数的数据来源是一张单表：student\_leave\_apply学生请假申请表；

而旷课人数的数据来源则包含了前面三个指标的所有数据来源，因此旷课人数的数据也是可以复用前面三个指标的。

理清了指标之间的复用关系，可以更好的帮助我们来设计分层，尽量复用数据。

1. 因为此部分数据都已经过OLTP系统清洗过，所以跳过DWD直接进入DWM转换过程；
2. 各个指标的中间层是有依赖关系的，旷课数据是依赖于出勤（包含迟到）、请假数据；
3. 出勤和迟到人数指标DWM层，先根据学生打卡表、班级课表以及班级作息时间表，计算出学生的正常出勤和迟到数据；然后在学生出勤数据基础上，统计出班级的出勤和迟到数据。
4. 请假人数指标DWM层，根据学生请假申请表、班级课表以及班级作息时间表，计算出班级的请假数据。
5. 旷课人数指标DWM层，在班级出勤中间表、班级请假中间表的基础上，关联在读学员人数、班级课表，计算出旷课数据。
6. DWS层，将DWM层的班级出勤数据、班级请假数据、班级旷课数据进行汇总，并计算出其与在读学员人数的比值。
7. 最后在APP层，可以按照年月日维度进行二次汇总。

## 实现

### 建模

#### 指标和维度

指标：

* 在读学员人数指标
* 出勤人数指标
* 迟到人数指标
* 请假人数指标
* 旷课人数指标

维度：

* + 时间维度：年、月、天
  + 时间段维度：上午、下午、晚自习
  + 组织维度：班级、学生

因为学生数据较多，所以可以将学生与班级维度分为两个不同的模型。先统计学生数据作为中间数据，然后在此基础上统计班级数据作为最终数据。提升班级数据的查询效率。

#### 事实表和维度表

##### 在读学员人数指标

class\_studying\_student\_count班级在读学生人数表就是事实表。

##### 出勤人数指标

tbh\_student\_signin\_record学生打卡记录表是事实表。

course\_table\_upload\_detail班级课表、tbh\_class\_time\_table班级作息时间表是维度表。

##### 迟到人数指标

同出勤人数指标。

##### 请假人数指标

student\_leave\_apply学生请假申请表是事实表。

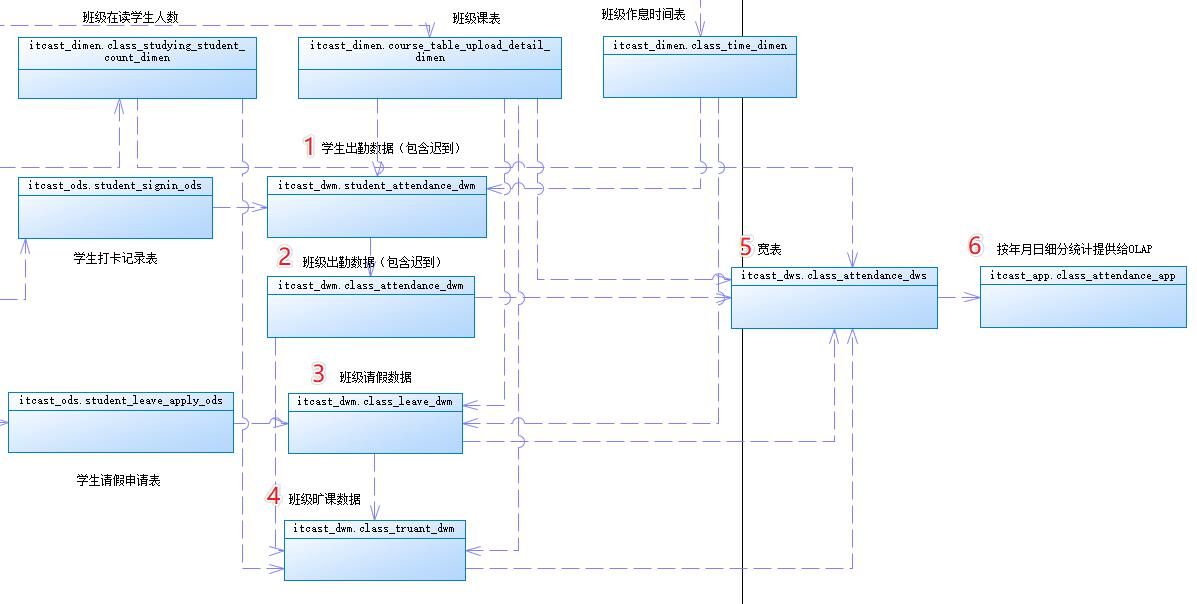
##### 旷课人数指标

通过其他指标计算出来。

#### 分层

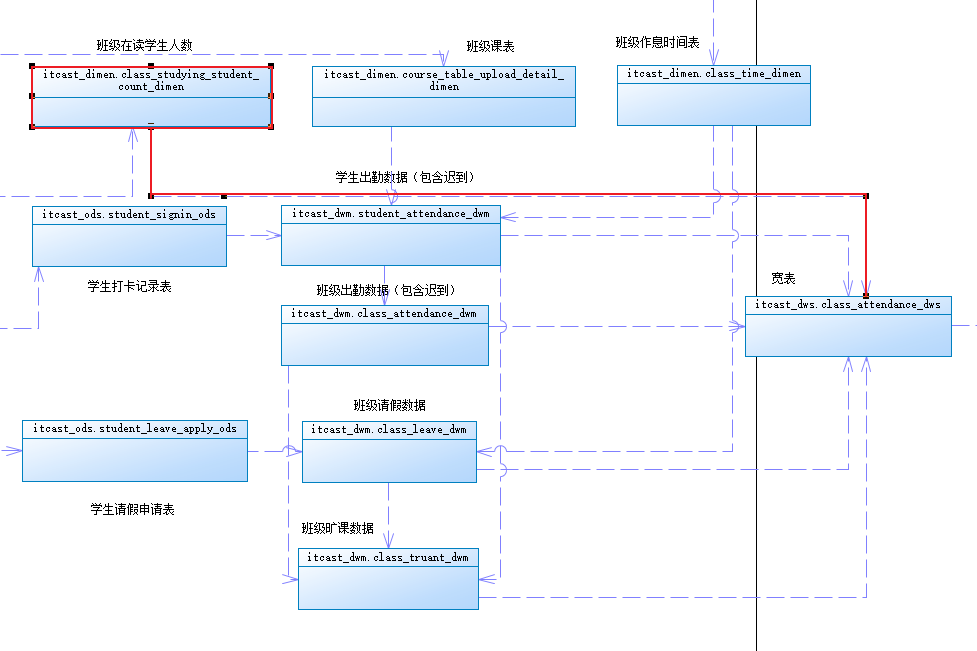
出勤人数指标和迟到人数指标，数据来源完全一致，因此可以合并计算。

旷课人数指标依赖于其他指标计算出的结果，因此在ODS和DWD层无需计算，只要在DWM和DWS层进行计算即可。



##### 在读学员人数指标

因为class\_studying\_student\_count班级在读学生人数表的数据，是经过OLTP应用计算好的，数据粒度就是按照班级来的，所以此处可以直接应用。因为要被各种出勤率指标使用，所以放在dimension层。



###### Dimen

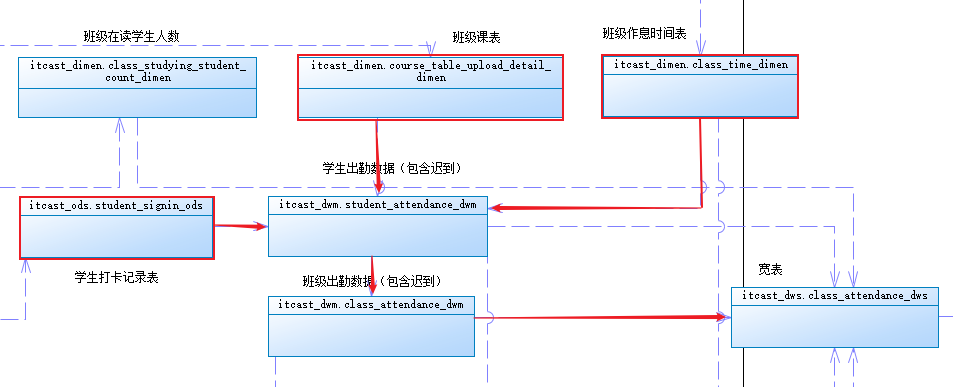
写入时压缩生效

|  |
| --- |
| set hive.exec.orc.compression.strategy=COMPRESSION; |

class\_studying\_student\_count\_dimen班级在读学生人数

|  |
| --- |
| drop table itcast\_dimen.class\_studying\_student\_count\_dimen;  CREATE TABLE IF NOT EXISTS itcast\_dimen.class\_studying\_student\_count\_dimen (  id int,  school\_id int comment '校区id',  subject\_id int comment '学科id',  class\_id int comment '班级id',  studying\_student\_count int comment '在读班级人数',  studying\_date STRING comment '在读日期') comment '在读班级的每天在读学员人数' PARTITIONED BY (dt STRING) ROW FORMAT DELIMITED  FIELDS TERMINATED BY '\t' stored as orcfile  TBLPROPERTIES ('orc.compress'='SNAPPY','orc.bloom.filter.columns'='studying\_student\_count,studying\_date'); |

##### 出勤人数和迟到人数指标



###### Dimension

course\_table\_upload\_detail\_dimen班级课表

|  |
| --- |
| drop table itcast\_dimen.course\_table\_upload\_detail\_dimen;  CREATE TABLE IF NOT EXISTS itcast\_dimen.course\_table\_upload\_detail\_dimen (  id int comment 'id',  base\_id int comment '课程主表id',  class\_id int comment '班级id',  class\_date STRING comment '上课日期',  content STRING comment '课程内容',  teacher\_id int comment '老师id',  teacher\_name STRING comment '老师名字',  job\_number STRING comment '工号',  classroom\_id int comment '教室id',  classroom\_name STRING comment '教室名称',  is\_outline int comment '是否大纲 0 否 1 是',  class\_mode int comment '上课模式 0 传统全天 1 AB上午 2 AB下午 3 线上直播',  is\_stage\_exam int comment '是否阶段考试（0：否 1：是）',  is\_pay int comment '代课费（0：无 1：有）',  tutor\_teacher\_id int comment '晚自习辅导老师id',  tutor\_teacher\_name STRING comment '辅导老师姓名',  tutor\_job\_number STRING comment '晚自习辅导老师工号',  is\_subsidy int comment '晚自习补贴（0：无 1：有）',  answer\_teacher\_id int comment '答疑老师id',  answer\_teacher\_name STRING comment '答疑老师姓名',  answer\_job\_number STRING comment '答疑老师工号',  remark STRING comment '备注',  create\_time STRING comment '创建时间') comment '班级课表明细表' PARTITIONED BY (dt STRING) ROW FORMAT DELIMITED  FIELDS TERMINATED BY '\t' stored as orcfile  TBLPROPERTIES ('orc.compress'='SNAPPY','orc.bloom.filter.columns'='class\_id,class\_date'); |

class\_time\_dimen班级作息时间表

|  |
| --- |
| drop table itcast\_dimen.class\_time\_dimen;  CREATE TABLE IF NOT EXISTS itcast\_dimen.class\_time\_dimen (  id int,  class\_id int comment '班级id',  morning\_template\_id int comment '上午出勤模板id',  morning\_begin\_time STRING comment '上午开始时间',  morning\_end\_time STRING comment '上午结束时间',  afternoon\_template\_id int comment '下午出勤模板id',  afternoon\_begin\_time STRING comment '下午开始时间',  afternoon\_end\_time STRING comment '下午结束时间',  evening\_template\_id int comment '晚上出勤模板id',  evening\_begin\_time STRING comment '晚上开始时间',  evening\_end\_time STRING comment '晚上结束时间',  use\_begin\_date STRING comment '使用开始日期',  use\_end\_date STRING comment '使用结束日期',  create\_time STRING comment '创建时间',  create\_person int comment '创建人',  remark STRING comment '备注') comment '班级作息时间表' PARTITIONED BY (dt STRING) ROW FORMAT DELIMITED  FIELDS TERMINATED BY '\t' stored as orcfile  TBLPROPERTIES ('orc.compress'='SNAPPY','orc.bloom.filter.columns'='id,class\_id'); |

###### ODS

student\_signin\_ods学生打卡记录表

|  |
| --- |
| drop table itcast\_ods.student\_signin\_ods;  CREATE TABLE IF NOT EXISTS itcast\_ods.student\_signin\_ods (  id int,  normal\_class\_flag int comment '是否正课 1 正课 2 自习 3 休息',  time\_table\_id int comment '作息时间id normal\_class\_flag=2 关联tbh\_school\_time\_table 或者 normal\_class\_flag=1 关联 tbh\_class\_time\_table',  class\_id int comment '班级id',  student\_id int comment '学员id',  signin\_time String comment '签到时间',  signin\_date String comment '签到日期',  inner\_flag int comment '内外网标志 0 外网 1 内网',  signin\_type int comment '签到类型 1 心跳打卡 2 老师补卡 3 直播打卡',  share\_state int comment '共享屏幕状态 0 否 1是 在上午或下午段有共屏记录，则该段所有记录该字段为1，内网默认为1 外网默认为0 (暂不用)',  inner\_ip String comment '内网ip地址',  create\_time String comment '创建时间') comment '学生打卡记录表' PARTITIONED BY (dt STRING) ROW FORMAT DELIMITED  FIELDS TERMINATED BY '\t' stored as orcfile  TBLPROPERTIES ('orc.compress'='SNAPPY','orc.bloom.filter.columns'='time\_table\_id,class\_id,signin\_date,share\_state'); |

###### DWM

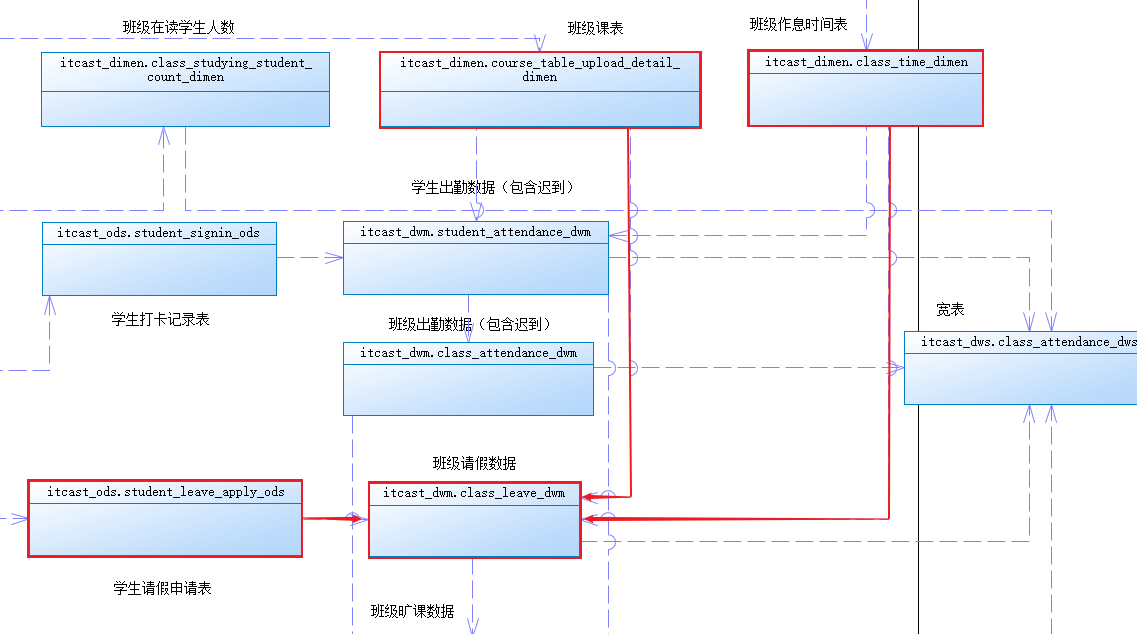
student\_attendance\_dwm 学生出勤(正常出勤和迟到)数据

|  |
| --- |
| drop table itcast\_dwm.student\_attendance\_dwm;  CREATE TABLE IF NOT EXISTS itcast\_dwm.student\_attendance\_dwm (  dateinfo String comment '日期',  class\_id int comment '班级id',  student\_id int comment '学员id',  morning\_att String comment '上午出勤情况：0.正常出勤、1.迟到、2.其他（请假+旷课）',  afternoon\_att String comment '下午出勤情况：0.正常出勤、1.迟到、2.其他（请假+旷课）',  evening\_att String comment '晚自习出勤情况：0.正常出勤、1.迟到、2.其他（请假+旷课）') comment '学生出勤(正常出勤和迟到)数据' PARTITIONED BY (yearinfo STRING, monthinfo STRING, dayinfo STRING) ROW FORMAT DELIMITED  FIELDS TERMINATED BY '\t' stored as orcfile  TBLPROPERTIES ('orc.compress'='SNAPPY'); |

class\_attendance\_dwm 班级出勤(正常出勤和迟到)数据

|  |
| --- |
| drop table itcast\_dwm.class\_attendance\_dwm; CREATE TABLE IF NOT EXISTS itcast\_dwm.class\_attendance\_dwm (  dateinfo String comment '日期',  class\_id int comment '班级id',  morning\_att\_count String comment '上午出勤人数',  afternoon\_att\_count String comment '下午出勤人数',  evening\_att\_count String comment '晚自习出勤人数',  morning\_late\_count String comment '上午迟到人数',  afternoon\_late\_count String comment '下午迟到人数',  evening\_late\_count String comment '晚自习迟到人数') comment '学生出勤(正常出勤和迟到)数据' PARTITIONED BY (yearinfo STRING, monthinfo STRING, dayinfo STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' stored as orcfile TBLPROPERTIES ('orc.compress'='SNAPPY'); |

##### 请假人数指标



###### ODS

student\_leave\_apply\_ods 学生请假申请表

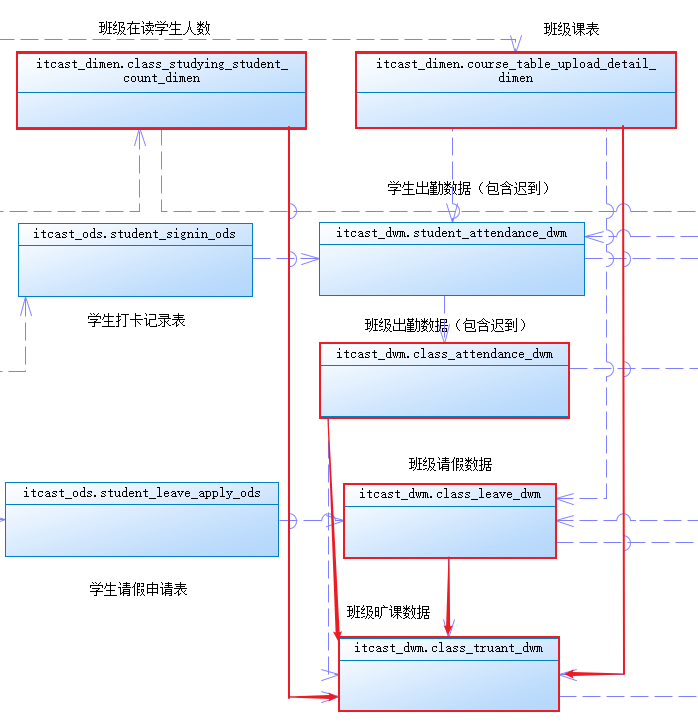
|  |
| --- |
| drop table itcast\_ods.student\_leave\_apply\_ods;  CREATE TABLE IF NOT EXISTS itcast\_ods.student\_leave\_apply\_ods (  id int,  class\_id int comment '班级id',  student\_id int comment '学员id',  audit\_state int comment '审核状态 0 待审核 1 通过 2 不通过',  audit\_person int comment '审核人',  audit\_time String comment '审核时间',  audit\_remark String comment '审核备注',  leave\_type int comment '请假类型 1 请假 2 销假 （查询是否请假不用过滤此类型，通过有效状态来判断）',  leave\_reason int comment '请假原因 1 事假 2 病假',  begin\_time String comment '请假开始时间',  begin\_time\_type int comment '1：上午 2：下午 3：晚自习',  end\_time String comment '请假结束时间',  end\_time\_type int comment '1：上午 2：下午 3：晚自习',  days float comment '请假/已休天数',  cancel\_state int comment '撤销状态 0 未撤销 1 已撤销',  cancel\_time String comment '撤销时间',  old\_leave\_id int comment '原请假id，只有leave\_type =2 销假的时候才有',  leave\_remark String comment '请假/销假说明',  valid\_state int comment '是否有效（0：无效 1：有效）',  create\_time String comment '创建时间') comment '学生请假申请表' PARTITIONED BY (dt STRING) ROW FORMAT DELIMITED  FIELDS TERMINATED BY '\t' stored as orcfile  TBLPROPERTIES ('orc.compress'='SNAPPY','orc.bloom.filter.columns'='class\_id,audit\_state,cancel\_state,valid\_state'); |

###### DWM

class\_leave\_dwm 班级请假数据

|  |
| --- |
| drop table itcast\_dwm.class\_leave\_dwm;  CREATE TABLE IF NOT EXISTS itcast\_dwm.class\_leave\_dwm (  dateinfo String comment '日期',  class\_id int comment '班级id',  morning\_leave\_count String comment '上午请假人数',  afternoon\_leave\_count String comment '下午请假人数',  evening\_leave\_count String comment '晚自习请假人数') comment '班级请假数据统计' PARTITIONED BY (yearinfo STRING, monthinfo STRING, dayinfo STRING) ROW FORMAT DELIMITED  FIELDS TERMINATED BY '\t' stored as orcfile  TBLPROPERTIES ('orc.compress'='SNAPPY'); |

##### 旷课人数指标



###### ODS

无。

###### DWM

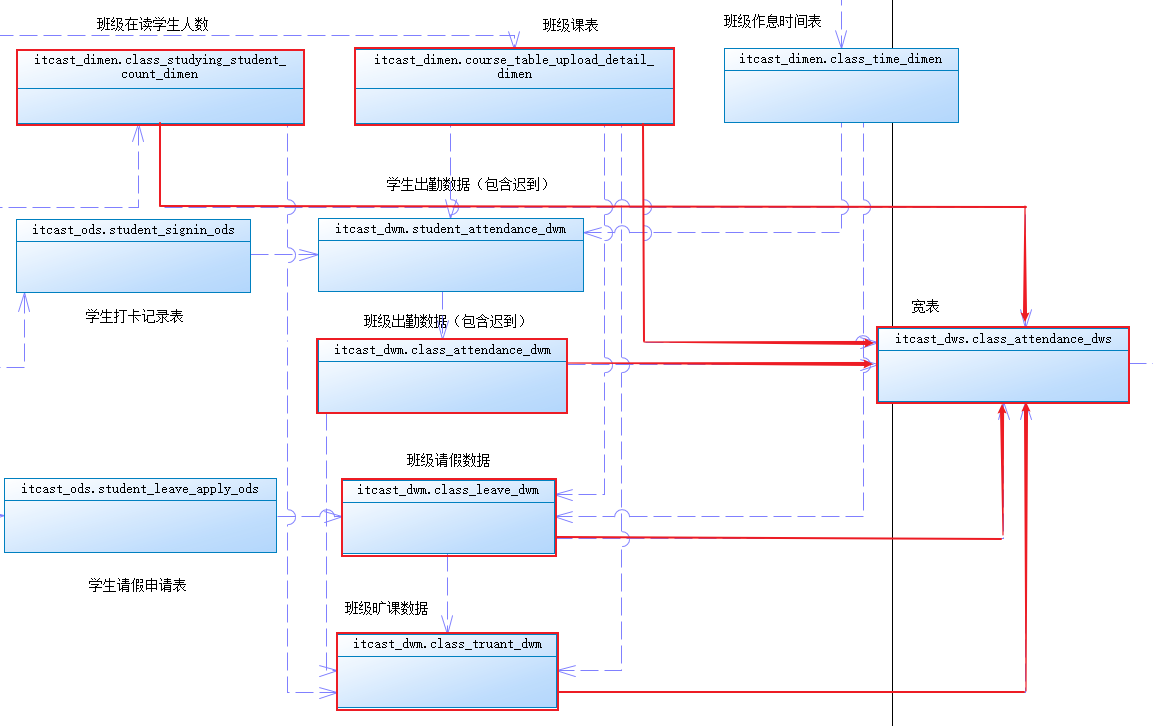
旷课人数无需下钻学生数据，因此只统计班级旷课数据即可。

class\_truant\_dwm 班级旷课数据

|  |
| --- |
| drop table itcast\_dwm.class\_truant\_dwm;  CREATE TABLE IF NOT EXISTS itcast\_dwm.class\_truant\_dwm (  dateinfo String comment '日期',  class\_id int comment '班级id',  morning\_truant\_count String comment '上午旷课人数',  afternoon\_truant\_count String comment '下午旷课人数',  evening\_truant\_count String comment '晚自习旷课人数') comment '班级请假数据统计' PARTITIONED BY (yearinfo STRING, monthinfo STRING, dayinfo STRING) ROW FORMAT DELIMITED  FIELDS TERMINATED BY '\t' stored as orcfile  TBLPROPERTIES ('orc.compress'='SNAPPY'); |

##### DWS数据集市

DWS数据集市将出勤和迟到数据、请假数据、旷课数据都整合在一起，便于APP使用。

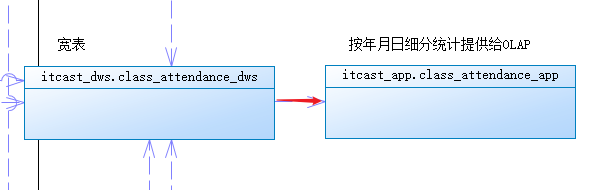


###### class\_attendance\_dws 班级出勤数据汇总

|  |
| --- |
| *--class\_attendance\_dws 班级出勤数据汇总* drop table itcast\_dws.class\_attendance\_dws;  CREATE TABLE IF NOT EXISTS itcast\_dws.class\_attendance\_dws (  dateinfo String comment '日期',  class\_id int comment '班级id',  studying\_student\_count int comment '在读班级人数',  morning\_att\_count String comment '上午出勤人数',  morning\_att\_ratio String comment '上午出勤率',  afternoon\_att\_count String comment '下午出勤人数',  afternoon\_att\_ratio String comment '下午出勤率',  evening\_att\_count String comment '晚自习出勤人数',  evening\_att\_ratio String comment '晚自习出勤率',  morning\_late\_count String comment '上午迟到人数',  morning\_late\_ratio String comment '上午迟到率',  afternoon\_late\_count String comment '下午迟到人数',  afternoon\_late\_ratio String comment '下午迟到率',  evening\_late\_count String comment '晚自习迟到人数',  evening\_late\_ratio String comment '晚自习迟到率',  morning\_leave\_count String comment '上午请假人数',  morning\_leave\_ratio String comment '上午请假率',  afternoon\_leave\_count String comment '下午请假人数',  afternoon\_leave\_ratio String comment '下午请假率',  evening\_leave\_count String comment '晚自习请假人数',  evening\_leave\_ratio String comment '晚自习请假率',  morning\_truant\_count String comment '上午旷课人数',  morning\_truant\_ratio String comment '上午旷课率',  afternoon\_truant\_count String comment '下午旷课人数',  afternoon\_truant\_ratio String comment '下午旷课率',  evening\_truant\_count String comment '晚自习旷课人数',  evening\_truant\_ratio String comment '晚自习旷课率') comment '班级请假数据统计' PARTITIONED BY (yearinfo STRING, monthinfo STRING, dayinfo STRING) ROW FORMAT DELIMITED  FIELDS TERMINATED BY '\t' stored as orcfile  TBLPROPERTIES ('orc.compress'='SNAPPY'); |

##### APP

APP层将需要的数据，按照年、月、天进行分组统计，以便于OLAP应用查询。



###### class\_attendance\_app

|  |
| --- |
| *--class\_attendance\_app OLAP应用数据* drop table itcast\_app.class\_attendance\_app;  CREATE TABLE IF NOT EXISTS itcast\_app.class\_attendance\_app (  dateinfo String comment '日期',  class\_id int comment '班级id',  studying\_student\_count int comment '在读班级人数',  morning\_att\_count String comment '上午出勤人数',  morning\_att\_ratio String comment '上午出勤率',  afternoon\_att\_count String comment '下午出勤人数',  afternoon\_att\_ratio String comment '下午出勤率',  evening\_att\_count String comment '晚自习出勤人数',  evening\_att\_ratio String comment '晚自习出勤率',  morning\_late\_count String comment '上午迟到人数',  morning\_late\_ratio String comment '上午迟到率',  afternoon\_late\_count String comment '下午迟到人数',  afternoon\_late\_ratio String comment '下午迟到率',  evening\_late\_count String comment '晚自习迟到人数',  evening\_late\_ratio String comment '晚自习迟到率',  morning\_leave\_count String comment '上午请假人数',  morning\_leave\_ratio String comment '上午请假率',  afternoon\_leave\_count String comment '下午请假人数',  afternoon\_leave\_ratio String comment '下午请假率',  evening\_leave\_count String comment '晚自习请假人数',  evening\_leave\_ratio String comment '晚自习请假率',  morning\_truant\_count String comment '上午旷课人数',  morning\_truant\_ratio String comment '上午旷课率',  afternoon\_truant\_count String comment '下午旷课人数',  afternoon\_truant\_ratio String comment '下午旷课率',  evening\_truant\_count String comment '晚自习旷课人数',  evening\_truant\_ratio String comment '晚自习旷课率',  time\_type STRING COMMENT '聚合时间类型：1、按小时聚合；2、按天聚合；3、按周聚合；4、按月聚合；5、按年聚合。') comment '班级请假数据统计' PARTITIONED BY (yearinfo STRING, monthinfo STRING, dayinfo STRING) ROW FORMAT DELIMITED  FIELDS TERMINATED BY '\t' stored as orcfile  TBLPROPERTIES ('orc.compress'='SNAPPY'); |

### 全量流程

#### 数据采集

##### Dimen

###### class\_studying\_student\_count\_dimen 在读班级的每天在读学员人数

|  |
| --- |
| sqoop import \  --connect jdbc:mysql://192.168.52.150:3306/teach \  --username root \  --password 123456 \  --query 'select id, school\_id, subject\_id, class\_id, ifnull(studying\_student\_count,0) studying\_student\_count, studying\_date, FROM\_UNIXTIME(unix\_timestamp(),"%Y-%m-%d") as dt from class\_studying\_student\_count where $CONDITIONS' \  --hcatalog-database itcast\_dimen \  --hcatalog-table class\_studying\_student\_count\_dimen \  -m 100 \  --split-by id |

###### course\_table\_upload\_detail\_dimen 班级课表

|  |
| --- |
| sqoop import \  --connect jdbc:mysql://192.168.52.150:3306/teach \  --username root \  --password 123456 \  --query 'select id,  base\_id,  class\_id,  class\_date,  content,  teacher\_id,  teacher\_name,  job\_number,  classroom\_id,  classroom\_name,  is\_outline,  class\_mode,  is\_stage\_exam,  is\_pay,  tutor\_teacher\_id,  tutor\_teacher\_name,  tutor\_job\_number,  is\_subsidy,  answer\_teacher\_id,  answer\_teacher\_name,  answer\_job\_number,  remark,  create\_time,  FROM\_UNIXTIME(unix\_timestamp(),"%Y-%m-%d") as dt from course\_table\_upload\_detail where $CONDITIONS' \  --hcatalog-database itcast\_dimen \  --hcatalog-table course\_table\_upload\_detail\_dimen \  -m 100 \  --split-by id |

###### class\_time\_dimen 班级作息时间表

|  |
| --- |
| sqoop import \  --connect jdbc:mysql://192.168.52.150:3306/teach \  --username root \  --password 123456 \  --query 'select \*, FROM\_UNIXTIME(unix\_timestamp(),"%Y-%m-%d") as dt from tbh\_class\_time\_table where $CONDITIONS' \  --hcatalog-database itcast\_dimen \  --hcatalog-table class\_time\_dimen \  -m 100 \  --split-by id |

##### ODS

###### student\_signin\_ods 学生打卡记录表

|  |
| --- |
| sqoop import \  --connect jdbc:mysql://192.168.52.150:3306/teach \  --username root \  --password 123456 \  --query 'select \*, FROM\_UNIXTIME(unix\_timestamp(),"%Y-%m-%d") as dt from tbh\_student\_signin\_record where $CONDITIONS' \  --hcatalog-database itcast\_ods \  --hcatalog-table student\_signin\_ods \  -m 100 \  --split-by id |

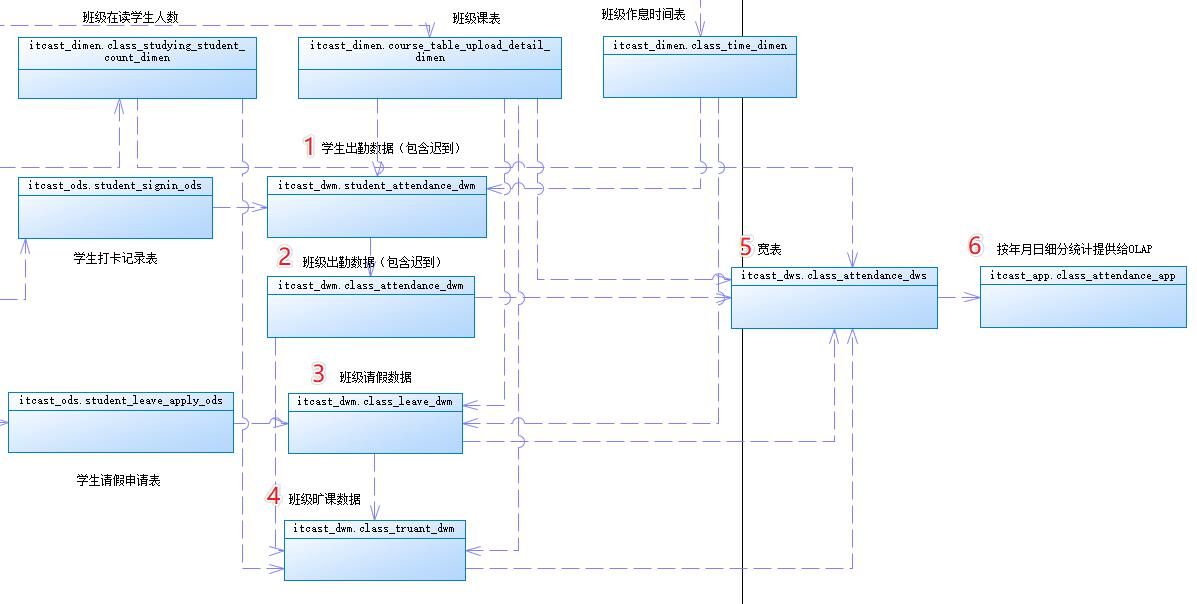
###### student\_leave\_apply\_ods 学生请假申请表

|  |
| --- |
| sqoop import \  --connect jdbc:mysql://192.168.52.150:3306/teach \  --username root \  --password 123456 \  --query 'select \*, FROM\_UNIXTIME(unix\_timestamp(),"%Y-%m-%d") as dt from student\_leave\_apply where $CONDITIONS' \  --hcatalog-database itcast\_ods \  --hcatalog-table student\_leave\_apply\_ods \  -m 100 \  --split-by id |

#### 数据清洗转换

因看板数据都已经过OLTP系统清洗过，所以跳过DWD直接进入DWM转换过程。

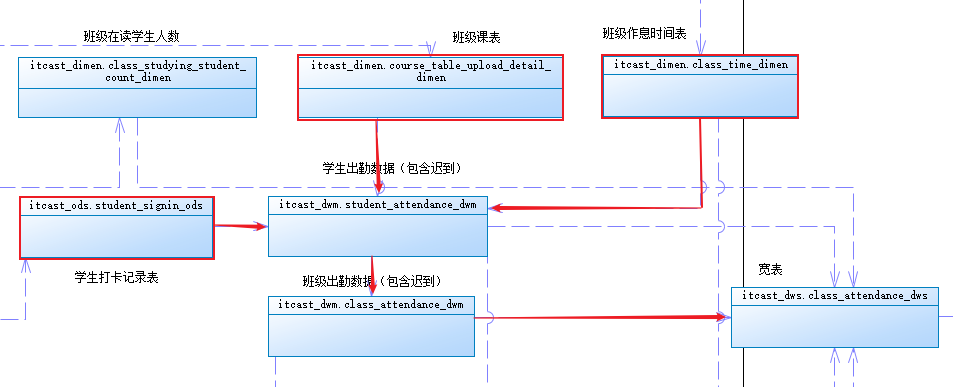
#### 统计分析



##### DWM

###### student\_attendance\_dwm 学生出勤(正常出勤和迟到)数据

分析



根据班级课表、学生打卡表以及班级作息时间表，计算出学生的正常出勤和迟到数据。

course\_table\_upload\_detail\_dimen班级课表当天的上课内容content不为空且不是开班典礼，即为正常上课，获取到上课班级的class\_id；

然后通过class\_id和student\_signin\_ods学生打卡记录表的class\_id关联，且打卡日期signin\_date要和班级课表的日期class\_date相同，且进入会议share\_state=1，即视为正常打卡，获取打卡时间signin\_time；

然后通过打卡表的time\_table\_id关联class\_time\_dimen班级作息时间表的id，以获取到规定的作息时间表;

（上午/下午/晚自习）打卡时间在上课前40分钟~下课时间段之内，则为正常出勤（未旷课）；打卡时间在上课前40分钟~上课后10分钟之内，则视为迟到；否则为其他（可能是请假或旷课）。

可先判断是否出勤（未旷课），然后再判断是否迟到，否则为其他（旷课或请假）。

因可能出现多次打卡，所以判断时，true返回1，false返回0，然后按照日期+学生id分组进行sum，只要有一次打卡满足条件，即为>0，视为满足条件。

出勤判断



上午打卡时间是否在上课前40分钟~下课时间段之内，原始SQL：

|  |
| --- |
| *time*(tssr.signin\_time) >= *TIMESTAMPADD*(minute, -40, tctt.morning\_begin\_time)  and  *time*(tssr.signin\_time) < tctt.morning\_end\_time |

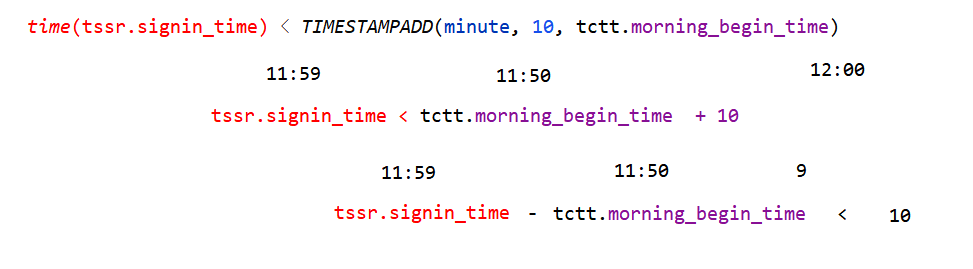
可以转换为

|  |
| --- |
| *cast*(  (*unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.morning\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS'))  / 60  as INT ) <= 40  and  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS') < *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.morning\_end\_time), 'yyyy-MM-dd HH:mm:ss') |

测试脚本：

|  |
| --- |
| *-- morning\_time - signup\_time <= 40* SELECT *cast*(  (*unix\_timestamp*(*concat*(signin\_date, ' ', morning\_time)) - *unix\_timestamp*(signup\_time))  / 60  as INT  ) <= 40 from (  SELECT '11:40:00' as morning\_time, '2020-08-20 11:01:00' as signup\_time, '2020-08-20' as signin\_date ) t; |

迟到判断



上午打卡时间是否在上课前40分钟~上课后10分钟之内，原始SQL：

|  |
| --- |
| *time*(tssr.signin\_time) >= *TIMESTAMPADD*(minute, -40, tctt.morning\_begin\_time)  and  *time*(tssr.signin\_time) < *TIMESTAMPADD*(minute, 10, tctt.morning\_begin\_time) |

可以转换为：

|  |
| --- |
| (  *cast*(  (  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.morning\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  )  / 60  as INT  ) <= **40** ) and (  *cast*(  (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  -  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.morning\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  )  / 60  as INT  ) <= **10** ) |

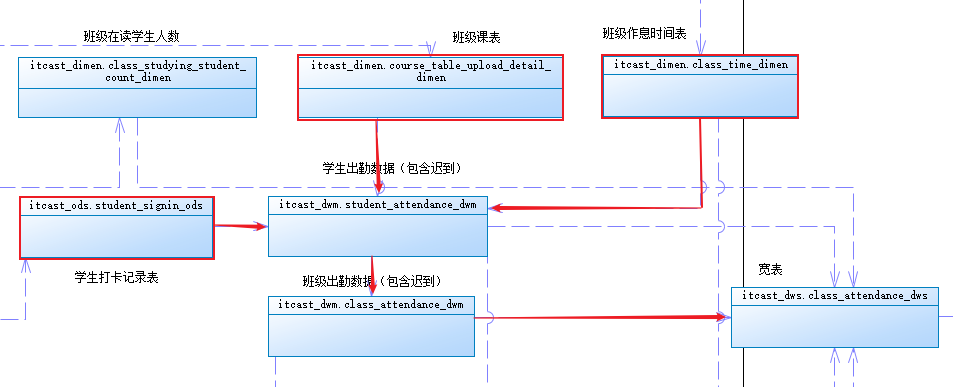
实现

注意：2019年9月3日有完善的测试数据。

|  |
| --- |
| *--分区* SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict; set hive.exec.max.dynamic.partitions.pernode=10000; set hive.exec.max.dynamic.partitions=100000; set hive.exec.max.created.files=150000; *--hive压缩* set hive.exec.compress.intermediate=true; set hive.exec.compress.output=true; *--写入时压缩生效* set hive.exec.orc.compression.strategy=COMPRESSION; *--分桶* set hive.enforce.bucketing=true; set hive.enforce.sorting=true; set hive.optimize.bucketmapjoin = true; set hive.auto.convert.sortmerge.join=true; set hive.auto.convert.sortmerge.join.noconditionaltask=true; *--并行执行* set hive.exec.parallel=true; set hive.exec.parallel.thread.number=8; *--小文件合并 -- set mapred.max.split.size=2147483648; -- set mapred.min.split.size.per.node=1000000000; -- set mapred.min.split.size.per.rack=1000000000; --矢量化查询* set hive.vectorized.execution.enabled=true; *--关联优化器* set hive.optimize.correlation=true; *--读取零拷贝* set hive.exec.orc.zerocopy=true; *--join数据倾斜* set hive.optimize.skewjoin=true; *-- set hive.skewjoin.key=100000;* set hive.optimize.skewjoin.compiletime=true; set hive.optimize.union.remove=true; *-- group倾斜* set hive.groupby.skewindata=true;   INSERT INTO itcast\_dwm.student\_attendance\_dwm PARTITION(yearinfo, monthinfo, dayinfo) SELECT   course.class\_date as dateinfo,  course.class\_id,  so.student\_id,  *-- 在上课前40分钟~上课时间点10分钟之内，为正常打卡0；在上课后10分钟~放学时间之内，为迟到1；否则为2.  if*(  *--判断是否正常出勤(包含迟到的)  sum*(  *if*(  (  *cast*(  (*unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.morning\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS'))  / 60  as INT  ) <= 40  )  and   (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')   <   *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.morning\_end\_time), 'yyyy-MM-dd HH:mm:ss')  ),  1, 0)  ) > 0,    *--正常出勤，但迟到了，判断并标记  if*(  *sum*(  *if*(  (  *cast*(  (  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.morning\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  )  / 60  as INT  ) <= 40  )  and   (  *cast*(  (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  -  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.morning\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  )  / 60  as INT  ) <= 10  ),  1, 0)  ) > 0,  0, 1  ),  2  ) as ***morning\_att***,   *-- 在上课前40分钟~上课时间点10分钟之内，为正常打卡0；在上课后10分钟~放学时间之内，为迟到1；否则为2.  if*(  *--判断是否正常出勤(包含迟到的)  sum*(  *if*(  (  *cast*(  (*unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.afternoon\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS'))  / 60  as INT  ) <= 40  )  and   (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')   <   *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.afternoon\_end\_time), 'yyyy-MM-dd HH:mm:ss')  ),  1, 0)  ) > 0,    *--正常出勤，但迟到了，判断并标记  if*(  *sum*(  *if*(  (  *cast*(  (  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.afternoon\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  )  / 60  as INT  ) <= 40  )  and   (  *cast*(  (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  -  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.afternoon\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  )  / 60  as INT  ) <= 10  ),  1, 0)  ) > 0,  0, 1  ),  2  ) as afternoon\_att,   *-- 在上课前40分钟~上课时间点10分钟之内，为正常打卡0；在上课后10分钟~放学时间之内，为迟到1；否则为2.  if*(  *--判断是否正常出勤(包含迟到的)  sum*(  *if*(  (  *cast*(  (*unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.evening\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS'))  / 60  as INT  ) <= 40  )  and   (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')   <   *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.evening\_end\_time), 'yyyy-MM-dd HH:mm:ss')  ),  1, 0)  ) > 0,    *--正常出勤，但迟到了，判断并标记  if*(  *sum*(  *if*(  (  *cast*(  (  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.evening\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  )  / 60  as INT  ) <= 40  )  and   (  *cast*(  (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  -  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.evening\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  )  / 60  as INT  ) <= 10  ),  1, 0)  ) > 0,  0, 1  ),  2  ) as evening\_att,  *substr*(course.class\_date, 1, 4) as yearinfo,   *substr*(course.class\_date, 6, 2) monthinfo,   *substr*(course.class\_date, 9, 2) dayinfo from itcast\_dimen.course\_table\_upload\_detail\_dimen course  LEFT JOIN itcast\_ods.student\_signin\_ods so on course.class\_id = so.class\_id and so.signin\_date=course.class\_date AND so.share\_state=1  LEFT JOIN itcast\_dimen.class\_time\_dimen ti on so.time\_table\_id=ti.id WHERE course.content IS NOT NULL AND course.content != '开班典礼' GROUP BY course.class\_date, course.class\_id, so.student\_id; |

###### class\_attendance\_dwm 班级出勤(正常出勤和迟到)数据

分析



在学生出勤数据基础上，统计出班级的出勤和迟到数据。此处涉及到count和distinct统计，可以使用hive优化器来进行查询。

所有聚合函数，比如sum、count、min、max、avg等均可针对包含空值的字段进行度量计算，其中sum、count(字段名)、min、max、avg会忽略空值，而count(1)或count(\*)在计数时会将空值包含在内。

上午出勤人数(包括迟到)，原始SQL：

|  |
| --- |
| *count*(distinct  (case when  tmp.morning\_signin = 0 or tmp.morning\_signin = 1  then tmp.student\_id  else null end))  as morning\_att\_count |

可以替换为：

|  |
| --- |
| *count*(DISTINCT  (  *if*(morning\_att='0' or morning\_att='1', stu.student\_id, null)  ) ) morning\_att\_count, |

迟到数据去掉morning\_att='0'的判断即可。

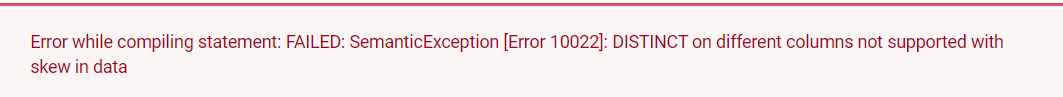
实现

注意：2019年9月3日有完善的测试数据。

|  |
| --- |
| *--分区* SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict; set hive.exec.max.dynamic.partitions.pernode=10000; set hive.exec.max.dynamic.partitions=100000; set hive.exec.max.created.files=150000; *--hive压缩* set hive.exec.compress.intermediate=true; set hive.exec.compress.output=true; *--写入时压缩生效* set hive.exec.orc.compression.strategy=COMPRESSION; *--分桶* set hive.enforce.bucketing=true; set hive.enforce.sorting=true; set hive.optimize.bucketmapjoin = true; set hive.auto.convert.sortmerge.join=true; set hive.auto.convert.sortmerge.join.noconditionaltask=true; *--并行执行* set hive.exec.parallel=true; set hive.exec.parallel.thread.number=8; *--小文件合并 -- set mapred.max.split.size=2147483648; -- set mapred.min.split.size.per.node=1000000000; -- set mapred.min.split.size.per.rack=1000000000; --矢量化查询* set hive.vectorized.execution.enabled=true; *--关联优化器* set hive.optimize.correlation=true; *--读取零拷贝* set hive.exec.orc.zerocopy=true; *--join数据倾斜* set hive.optimize.skewjoin=true; *-- set hive.skewjoin.key=100000;* set hive.optimize.skewjoin.compiletime=true; set hive.optimize.union.remove=true; *-- group倾斜* set hive.groupby.skewindata=false;   SELECT *\** from itcast\_dwm.class\_attendance\_dwm;  INSERT INTO itcast\_dwm.class\_attendance\_dwm PARTITION (yearinfo, monthinfo, dayinfo) SELECT  stu.dateinfo,  stu.class\_id,  *count*(DISTINCT  case WHEN (**stu.morning\_att='0' or stu.morning\_att='1'**) THEN stu.student\_id ELSE NULL END  ) morning\_att\_count,  *count*(DISTINCT  case WHEN (stu.afternoon\_att ='0' or stu.afternoon\_att='1') THEN stu.student\_id ELSE NULL END  ) afternoon\_att\_count,  *count*(DISTINCT  case WHEN (stu.evening\_att ='0' or stu.evening\_att='1') THEN stu.student\_id ELSE NULL END  ) evening\_att\_count,  *count*(DISTINCT  case WHEN (**stu.morning\_att='1'**) THEN stu.student\_id ELSE NULL END  ) morning\_late\_count,  *count*(DISTINCT  case WHEN (stu.afternoon\_att='1') THEN stu.student\_id ELSE NULL END  ) afternoon\_late\_count,  *count*(DISTINCT  case WHEN (stu.evening\_att='1') THEN stu.student\_id ELSE NULL END  ) evening\_late\_count,  stu.yearinfo,stu.monthinfo,stu.dayinfo  from itcast\_dwm.student\_attendance\_dwm stu GROUP BY stu.class\_id,stu.yearinfo,stu.monthinfo,stu.dayinfo,stu.dateinfo; |

group数据倾斜带来的问题

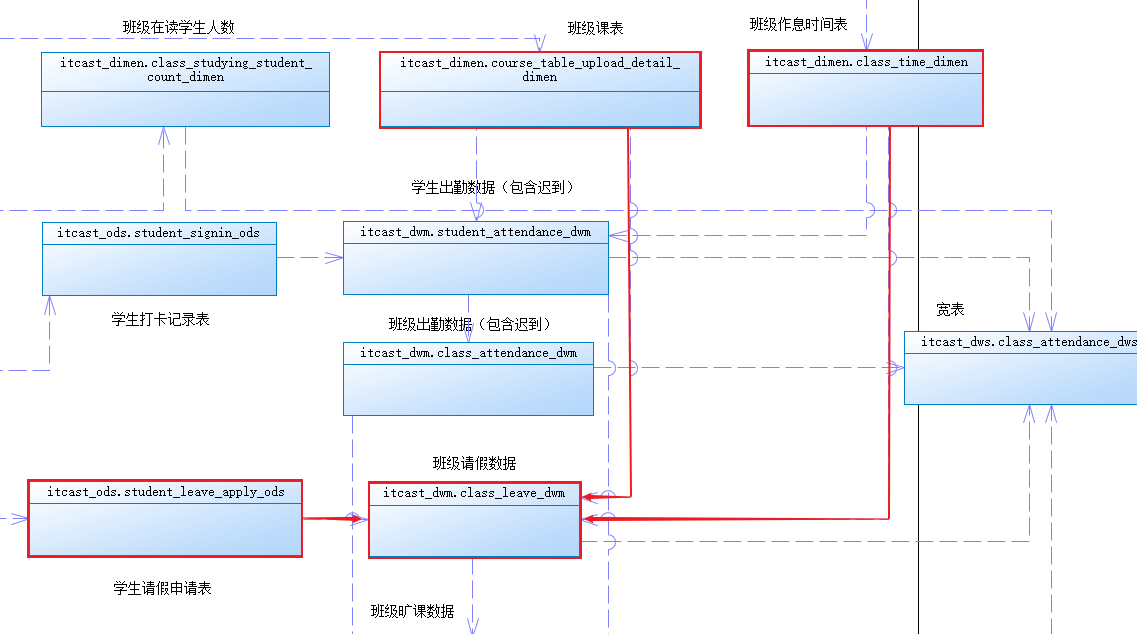
报错：DISTINCT on different columns not supported with skew in data。



hive.groupby.skewindata启用后虽然能够解决数据倾斜的问题，但是如果在查询语句中对多个字段进行去重distinct统计时就会报错，因此遇到这样的情况，必须临时关闭skewindata负载均衡：set hive.groupby.skewindata=false。

###### class\_leave\_dwm 班级请假数据

分析



根据学生请假申请表、班级课表以及班级作息时间表，计算出班级的请假数据。

请假合格的条件：审批通过、未撤销、有效、当天有课、班级匹配、请假时间包含课表上课开始时间。

student\_leave\_apply\_ods 学生请假申请表用来判断审批通过、未撤销、有效。

course\_table\_upload\_detail班级课表，主要用来获取上课日期（上课内容content不为空且不是开班典礼，即为正常上课），通过班级id进行匹配；

tbh\_class\_time\_table作息时间表, 用班级id和生效日期来匹配，主要用来获取规定的作息时间，用来和请假的时间做匹配：请假开始时间早于（小于等于）上课开始时间，且请假结束时间晚于（大于等于）上课开始时间。

不同的时间段（上午、下午、晚上）可以通过full join来进行连接。

实现

上午请假数据，原始SQL：

|  |
| --- |
| select cud.class\_date as dateinfo,   cud.class\_id,   *count*(distinct sla.student\_id) as morning\_leave\_count from student\_leave\_apply sla,  tbh\_class\_time\_table ct,  course\_table\_upload\_detail cud *-- 表关联* where sla.class\_id = ct.class\_id  and sla.class\_id = cud.class\_id *-- 课程生效，且当天有课程内容* and cud.class\_date between ct.use\_begin\_date and ct.use\_end\_date  AND cud.content IS NOT NULL  AND cud.content != '开班典礼' *-- 判断请假时间* and *concat*(cud.class\_date, ' ', ct.morning\_begin\_time) >= sla.begin\_time  and *concat*(cud.class\_date, ' ', ct.morning\_begin\_time) <= sla.end\_time *-- 请假状态已审核通过，且没有取消、数据有效* and sla.audit\_state = 1  and sla.cancel\_state = 0  and sla.valid\_state = 1 group by cud.class\_date, cud.class\_id; |

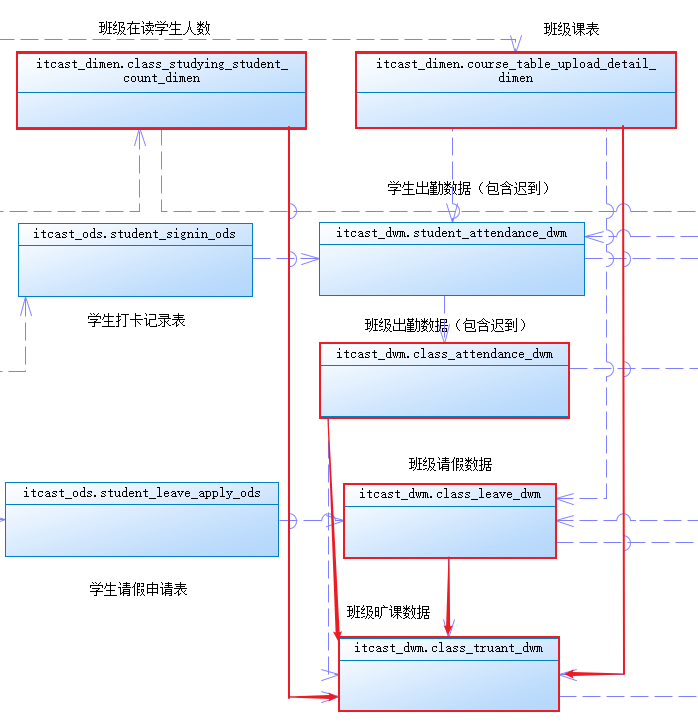
可以转换为：

|  |
| --- |
| SELECT cource.class\_date dateinfo,  cource.class\_id,  *count*(distinct lev.student\_id) as morning\_leave\_count from itcast\_ods.student\_leave\_apply\_ods lev,  itcast\_dimen.course\_table\_upload\_detail\_dimen cource,  itcast\_dimen.class\_time\_dimen ti where *-- 表关联* lev.class\_id = cource.class\_id  and lev.class\_id = ti.class\_id *-- 课程生效，且当天有课程内容* and cource.class\_date between ti.use\_begin\_date and ti.use\_end\_date  AND cource.content IS NOT NULL  AND cource.content != '开班典礼' *-- 判断请假时间* AND lev.begin\_time <= *concat\_ws*(' ', cource.class\_date, ti.morning\_begin\_time)  AND lev.end\_time >= *concat\_ws*(' ', cource.class\_date, ti.morning\_begin\_time) *-- 请假状态已审核通过，且没有取消、数据有效* AND lev.audit\_state = 1  AND lev.cancel\_state = 0  AND lev.valid\_state = 1 group by cource.class\_date, cource.class\_id; |

|  |
| --- |
| *--分区* SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict; set hive.exec.max.dynamic.partitions.pernode=10000; set hive.exec.max.dynamic.partitions=100000; set hive.exec.max.created.files=150000; *--hive压缩* set hive.exec.compress.intermediate=true; set hive.exec.compress.output=true; *--写入时压缩生效* set hive.exec.orc.compression.strategy=COMPRESSION; *--分桶* set hive.enforce.bucketing=true; set hive.enforce.sorting=true; set hive.optimize.bucketmapjoin = true; set hive.auto.convert.sortmerge.join=true; set hive.auto.convert.sortmerge.join.noconditionaltask=true; *--并行执行* set hive.exec.parallel=true; set hive.exec.parallel.thread.number=8; *--小文件合并 -- set mapred.max.split.size=2147483648; -- set mapred.min.split.size.per.node=1000000000; -- set mapred.min.split.size.per.rack=1000000000; --矢量化查询* set hive.vectorized.execution.enabled=true; *--关联优化器* set hive.optimize.correlation=true; *--读取零拷贝* set hive.exec.orc.zerocopy=true; *--join数据倾斜* set hive.optimize.skewjoin=true; *-- set hive.skewjoin.key=100000;* set hive.optimize.skewjoin.compiletime=true; set hive.optimize.union.remove=true; *-- group倾斜* set hive.groupby.skewindata=false;   INSERT INTO itcast\_ods.student\_leave\_apply\_ods partition (dt) values (125, 5032, 119142, 1, 3491, '2019-09-08 16:42:29', '狂犬疫苗最后一针', 1, 2, '2019-09-03 08:00:00', 2, '2019-09-03 23:30:00', 2, 1, 0, null, null, '医院打针，狂犬疫苗最后一针了', 1, '2019-09-02 08:56:54','2020-07-07');   INSERT INTO itcast\_dwm.class\_leave\_dwm PARTITION (yearinfo, monthinfo, dayinfo) SELECT  morning.dateinfo,  morning.class\_id,  morning.morning\_leave\_count,  afternoon.afternoon\_leave\_count,  evening.evening\_leave\_count,  *substr*(morning.dateinfo, 1, 4) yearinfo,  *substr*(morning.dateinfo, 6, 2) monthinfo,  *substr*(morning.dateinfo, 9, 2) dayinfo from (  SELECT  cource.class\_date as dateinfo,  cource.class\_id,  *count*(DISTINCT leave.student\_id) as morning\_leave\_count  FROM  itcast\_dimen.course\_table\_upload\_detail\_dimen cource,  itcast\_ods.student\_leave\_apply\_ods leave,  itcast\_dimen.class\_time\_dimen ti  WHERE cource.class\_id=leave.class\_id  and cource.class\_id=ti.class\_id  and cource.content is NOT NULL and cource.content != '开班典礼'  *--作息时间匹配* AND ti.use\_begin\_date <= cource.class\_date  AND ti.use\_end\_date >= cource.class\_date  *-- 请假开始时间 <= 上课开始时间 <= 请假结束时间* AND leave.begin\_time <= *concat*(cource.class\_date, ' ', ti.morning\_begin\_time)  AND *concat*(cource.class\_date, ' ', ti.morning\_begin\_time) <= leave.end\_time  *--请假是否有效* AND leave.audit\_state=1  AND leave.cancel\_state=0  AND leave.valid\_state=1  GROUP BY cource.class\_id, cource.class\_date ) morning FULL JOIN (  SELECT  cource.class\_date as dateinfo,  cource.class\_id,  *count*(DISTINCT leave.student\_id) as afternoon\_leave\_count  FROM  itcast\_dimen.course\_table\_upload\_detail\_dimen cource,  itcast\_ods.student\_leave\_apply\_ods leave,  itcast\_dimen.class\_time\_dimen ti  WHERE cource.class\_id=leave.class\_id  and cource.class\_id=ti.class\_id  and cource.content is NOT NULL and cource.content != '开班典礼'  *--作息时间匹配* AND ti.use\_begin\_date <= cource.class\_date  AND ti.use\_end\_date >= cource.class\_date  *-- 请假开始时间 <= 上课开始时间 <= 请假结束时间* AND leave.begin\_time <= *concat*(cource.class\_date, ' ', ti.afternoon\_begin\_time)  AND *concat*(cource.class\_date, ' ', ti.afternoon\_begin\_time) <= leave.end\_time  *--请假是否有效* AND leave.audit\_state=1  AND leave.cancel\_state=0  AND leave.valid\_state=1  GROUP BY cource.class\_id, cource.class\_date ) afternoon on morning.class\_id=afternoon.class\_id AND morning.dateinfo=afternoon.dateinfo FULL JOIN (  SELECT  cource.class\_date as dateinfo,  cource.class\_id,  *count*(DISTINCT leave.student\_id) as evening\_leave\_count  FROM  itcast\_dimen.course\_table\_upload\_detail\_dimen cource,  itcast\_ods.student\_leave\_apply\_ods leave,  itcast\_dimen.class\_time\_dimen ti  WHERE cource.class\_id=leave.class\_id  and cource.class\_id=ti.class\_id  and cource.content is NOT NULL and cource.content != '开班典礼'  *--作息时间匹配* AND ti.use\_begin\_date <= cource.class\_date  AND ti.use\_end\_date >= cource.class\_date  *-- 请假开始时间 <= 上课开始时间 <= 请假结束时间* AND leave.begin\_time <= *concat*(cource.class\_date, ' ', ti.evening\_begin\_time)  AND *concat*(cource.class\_date, ' ', ti.evening\_begin\_time) <= leave.end\_time  *--请假是否有效* AND leave.audit\_state=1  AND leave.cancel\_state=0  AND leave.valid\_state=1  GROUP BY cource.class\_id, cource.class\_date ) evening on morning.dateinfo=evening.dateinfo AND morning.class\_id=evening.class\_id; |

###### class\_truant\_dwm 班级旷课数据

分析



在班级课表的基础上，关联班级出勤中间表、班级请假中间表、在读学员人数，计算出旷课数据。

class\_studying\_student\_count\_dimen在读学员人数用来获取studying\_student\_count班级总人数，通过公式得到旷课人数=总人数-出勤人数（包含旷课人数）-请假人数。

course\_table\_upload\_detail班级课表，主要用来获取上课日期（上课内容content不为空且不是开班典礼，即为正常上课），通过班级id进行匹配。

在读学员人数以及课表内容不能为空，且课表内容不是开班典礼。

注意，数学运算时，如果有参数为null，则整体结果为null，要先进行null判断。

旷课人数计算，原始SQL：

|  |
| --- |
| (tmp3.student\_count - tmp3.morning\_att\_count - tmp3.morning\_leave\_count)  as morning\_truant\_count |

可以转换为：

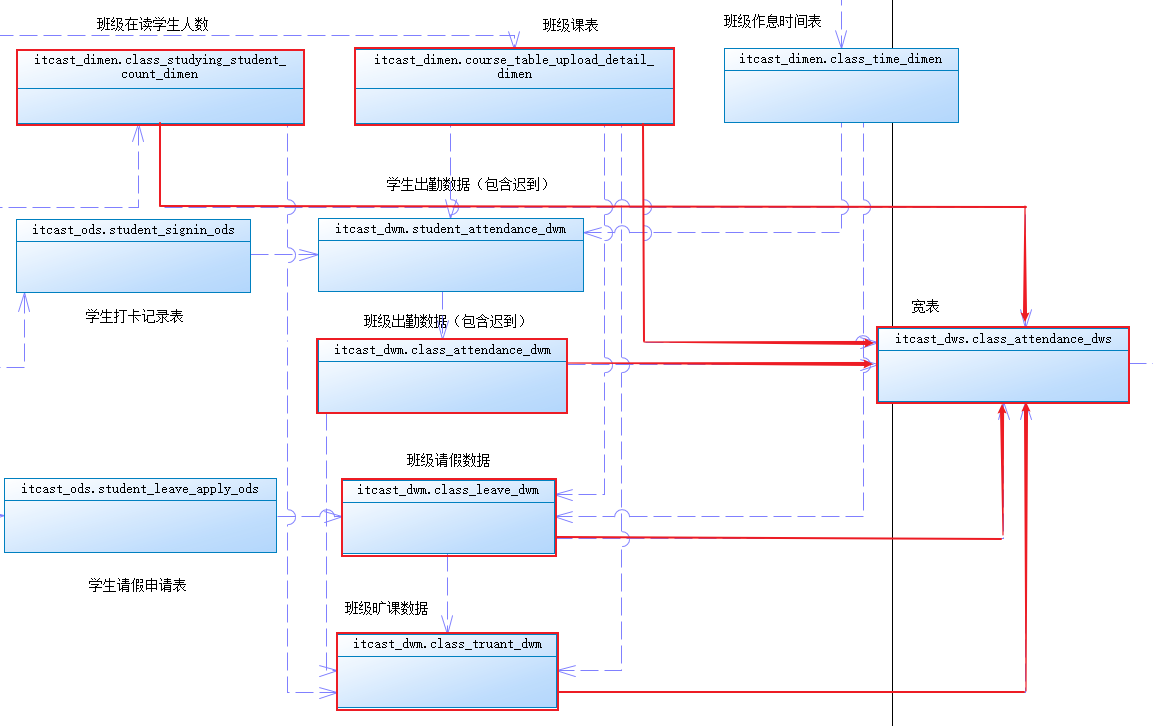
|  |
| --- |
| ***nvl***(ct.studying\_student\_count, **0**) - *nvl*(att.morning\_att\_count, 0) - *nvl*(leave.morning\_leave\_count, 0)  as morning\_truant\_count |

实现

|  |
| --- |
| *--分区* SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict; set hive.exec.max.dynamic.partitions.pernode=10000; set hive.exec.max.dynamic.partitions=100000; set hive.exec.max.created.files=150000; *--hive压缩* set hive.exec.compress.intermediate=true; set hive.exec.compress.output=true; *--写入时压缩生效* set hive.exec.orc.compression.strategy=COMPRESSION; *--分桶* set hive.enforce.bucketing=true; set hive.enforce.sorting=true; set hive.optimize.bucketmapjoin = true; set hive.auto.convert.sortmerge.join=true; set hive.auto.convert.sortmerge.join.noconditionaltask=true; *--并行执行* set hive.exec.parallel=true; set hive.exec.parallel.thread.number=8; *--小文件合并 -- set mapred.max.split.size=2147483648; -- set mapred.min.split.size.per.node=1000000000; -- set mapred.min.split.size.per.rack=1000000000; --矢量化查询* set hive.vectorized.execution.enabled=true; *--关联优化器* set hive.optimize.correlation=true; *--读取零拷贝* set hive.exec.orc.zerocopy=true; *--join数据倾斜* set hive.optimize.skewjoin=true; *-- set hive.skewjoin.key=100000;* set hive.optimize.skewjoin.compiletime=true; set hive.optimize.union.remove=true; *-- group倾斜* set hive.groupby.skewindata=false;   SELECT *\** from itcast\_dwm.class\_truant\_dwm WHERE dateinfo='2019-09-03';  INSERT INTO itcast\_dwm.class\_truant\_dwm PARTITION (yearinfo, monthinfo, dayinfo) SELECT  course.class\_date dateinfo,  course.class\_id,  *nvl*(ct.studying\_student\_count, 0) - *nvl*(att.morning\_att\_count, 0) - *nvl*(leave.morning\_leave\_count, 0) morning\_truant\_count,  *nvl*(ct.studying\_student\_count, 0) - *nvl*(att.afternoon\_att\_count, 0) - *nvl*(leave.afternoon\_leave\_count, 0) afternoon\_truant\_count,  *nvl*(ct.studying\_student\_count, 0) - *nvl*(att.evening\_att\_count, 0) - *nvl*(leave.evening\_leave\_count, 0) evening\_truant\_count,  *substr*(course.class\_date, 1,4) yearinfo,  *substr*(course.class\_date, 6,2) monthinfo,  *substr*(course.class\_date, 9,2) dayinfo from itcast\_dimen.course\_table\_upload\_detail\_dimen course  LEFT JOIN itcast\_dwm.class\_attendance\_dwm att on course.class\_id = att.class\_id and att.dateinfo = course.class\_date  LEFT JOIN itcast\_dwm.class\_leave\_dwm leave on course.class\_id = leave.class\_id AND course.class\_date = leave.dateinfo  LEFT JOIN itcast\_dimen.class\_studying\_student\_count\_dimen ct on ct.class\_id = course.class\_id and course.class\_date=ct.studying\_date WHERE ct.studying\_student\_count IS NOT NULL AND course.content IS NOT NULL AND course.content != '开班典礼'; |

##### DWS

分析



将DWM层的班级出勤数据、班级请假数据、班级旷课数据进行汇总，并计算出其与在读学员人数的比值。

除了DWM层的三张表之外，我们还要以course\_table\_upload\_detail班级课表为准，主要用来获取上课日期（上课内容content不为空且不是开班典礼，即为正常上课），然后通过班级id和上课日期匹配三张DWM表。

另外如果要计算比值，需要关联class\_studying\_student\_count\_dimen在读学员人数用来获取studying\_student\_count班级总人数。

在读学员人数以及课表内容不能为空，且课表内容不是开班典礼。

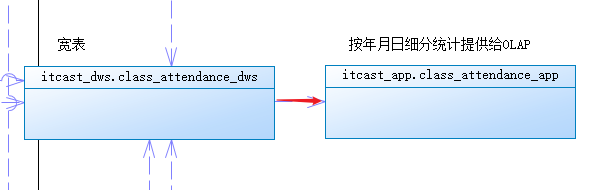
出勤率=出勤人数/班级总人数 \* 100，但是除法会出现小数点的情况，保留两位小数，比如86.55，hive可以通过decimal类型来限制小数位：

|  |
| --- |
| *cast*((att.morning\_att\_count / total.studying\_student\_count) \* 100 as **DECIMAL(8,2)**) |

实现

|  |
| --- |
| *--分区* SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict; set hive.exec.max.dynamic.partitions.pernode=10000; set hive.exec.max.dynamic.partitions=100000; set hive.exec.max.created.files=150000; *--hive压缩* set hive.exec.compress.intermediate=true; set hive.exec.compress.output=true; *--写入时压缩生效* set hive.exec.orc.compression.strategy=COMPRESSION; *--分桶* set hive.enforce.bucketing=true; set hive.enforce.sorting=true; set hive.optimize.bucketmapjoin = true; set hive.auto.convert.sortmerge.join=true; set hive.auto.convert.sortmerge.join.noconditionaltask=true; *--并行执行* set hive.exec.parallel=true; set hive.exec.parallel.thread.number=8; *--小文件合并 -- set mapred.max.split.size=2147483648; -- set mapred.min.split.size.per.node=1000000000; -- set mapred.min.split.size.per.rack=1000000000; --矢量化查询* set hive.vectorized.execution.enabled=true; *--关联优化器* set hive.optimize.correlation=true; *--读取零拷贝* set hive.exec.orc.zerocopy=true; *--join数据倾斜* set hive.optimize.skewjoin=true; *-- set hive.skewjoin.key=100000;* set hive.optimize.skewjoin.compiletime=true; set hive.optimize.union.remove=true; *-- group倾斜* set hive.groupby.skewindata=false;   INSERT INTO itcast\_dws.class\_attendance\_dws PARTITION (yearinfo, monthinfo, dayinfo) SELECT   course.class\_date dateinfo,   course.class\_id,  total.studying\_student\_count,  att.morning\_att\_count,  *cast*((att.morning\_att\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) morning\_att\_ratio,   att.afternoon\_att\_count,  *cast*((att.afternoon\_att\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) afternoon\_att\_ratio,   att.evening\_att\_count,  *cast*((att.evening\_att\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) evening\_att\_ratio,     att.morning\_late\_count,  *cast*((att.morning\_late\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) morning\_late\_ratio,   att.afternoon\_late\_count,  *cast*((att.afternoon\_late\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) afternoon\_late\_ratio,   att.evening\_late\_count,  *cast*((att.evening\_late\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) evening\_late\_ratio,    lev.morning\_leave\_count,  *cast*((lev.morning\_leave\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) morning\_leave\_ratio,   lev.afternoon\_leave\_count,  *cast*((lev.afternoon\_leave\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) afternoon\_leave\_ratio,   lev.evening\_leave\_count,  *cast*((lev.evening\_leave\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) evening\_leave\_ratio,     tru.morning\_truant\_count,  *cast*((tru.morning\_truant\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) morning\_truant\_ratio,   tru.afternoon\_truant\_count,  *cast*((tru.afternoon\_truant\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) afternoon\_truant\_ratio,   tru.evening\_truant\_count,  *cast*((tru.evening\_truant\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) evening\_truant\_ratio,     *substr*(course.class\_date, 1,4) yearinfo,  *substr*(course.class\_date, 6,2) monthinfo,  *substr*(course.class\_date, 9,2) dayinfo from itcast\_dimen.course\_table\_upload\_detail\_dimen course  LEFT JOIN itcast\_dimen.class\_studying\_student\_count\_dimen total on course.class\_date=total.studying\_date AND course.class\_id = total.class\_id  LEFT JOIN itcast\_dwm.class\_attendance\_dwm att on course.class\_date=att.dateinfo AND course.class\_id = att.class\_id  LEFT JOIN itcast\_dwm.class\_leave\_dwm lev on course.class\_id=lev.class\_id AND course.class\_date=lev.dateinfo  LEFT JOIN itcast\_dwm.class\_truant\_dwm tru on course.class\_id=tru.class\_id AND course.class\_date=tru.dateinfo WHERE course.content IS NOT NULL AND course.content != '开班典礼' AND total.studying\_student\_count IS NOT NULL; |

##### APP



按照年月日维度进行统计。

天的数据可以直接从DWS查询出来。

月和年的出勤人次数据，需要在DWS基础上，进行分组求和；

比值数据，需要求和后重新计算。

天：

|  |
| --- |
| *--分区* SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict; set hive.exec.max.dynamic.partitions.pernode=10000; set hive.exec.max.dynamic.partitions=100000; set hive.exec.max.created.files=150000; *--hive压缩* set hive.exec.compress.intermediate=true; set hive.exec.compress.output=true; *--写入时压缩生效* set hive.exec.orc.compression.strategy=COMPRESSION; *--分桶* set hive.enforce.bucketing=true; set hive.enforce.sorting=true; set hive.optimize.bucketmapjoin = true; set hive.auto.convert.sortmerge.join=true; set hive.auto.convert.sortmerge.join.noconditionaltask=true; *--并行执行* set hive.exec.parallel=true; set hive.exec.parallel.thread.number=8; *--小文件合并 -- set mapred.max.split.size=2147483648; -- set mapred.min.split.size.per.node=1000000000; -- set mapred.min.split.size.per.rack=1000000000; --矢量化查询* set hive.vectorized.execution.enabled=true; *--关联优化器* set hive.optimize.correlation=true; *--读取零拷贝* set hive.exec.orc.zerocopy=true; *--join数据倾斜* set hive.optimize.skewjoin=true; *-- set hive.skewjoin.key=100000;* set hive.optimize.skewjoin.compiletime=true; set hive.optimize.union.remove=true; *-- group倾斜* set hive.groupby.skewindata=false;  INSERT OVERWRITE TABLE itcast\_app.class\_attendance\_app partition (yearinfo, monthinfo, dayinfo) select  dateinfo,  class\_id,  studying\_student\_count,  morning\_att\_count,  morning\_att\_ratio,  afternoon\_att\_count,  afternoon\_att\_ratio,  evening\_att\_count,  evening\_att\_ratio,  morning\_late\_count,  morning\_late\_ratio,  afternoon\_late\_count,  afternoon\_late\_ratio,  evening\_late\_count,  evening\_late\_ratio,  morning\_leave\_count,  morning\_leave\_ratio,  afternoon\_leave\_count,  afternoon\_leave\_ratio,  evening\_leave\_count,  evening\_leave\_ratio,  morning\_truant\_count,  morning\_truant\_ratio,  afternoon\_truant\_count,  afternoon\_truant\_ratio,  evening\_truant\_count,  evening\_truant\_ratio,  '2',  yearinfo,  monthinfo,  dayinfo from itcast\_dws.class\_attendance\_dws; |

月：

|  |
| --- |
| INSERT OVERWRITE TABLE itcast\_app.class\_attendance\_app partition (yearinfo, monthinfo, dayinfo) select  *CONCAT\_WS*('-',yearinfo,monthinfo),  class\_id,  ***sum*(**studying\_student\_count),  ***sum*(**morning\_att\_count),  *cast*((***sum***(morning\_att\_count) / ***sum***(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_att\_ratio,  *sum*(afternoon\_att\_count),  *cast*((*sum*(afternoon\_att\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_att\_ratio,  *sum*(evening\_att\_count),  *cast*((*sum*(evening\_att\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_att\_ratio,  *sum*(morning\_late\_count),  *cast*((*sum*(morning\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_late\_ratio,  *sum*(afternoon\_late\_count),  *cast*((*sum*(afternoon\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_late\_ratio,  *sum*(evening\_late\_count),  *cast*((*sum*(evening\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_late\_ratio,  *sum*(morning\_leave\_count),  *cast*((*sum*(morning\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_leave\_ratio,  *sum*(afternoon\_leave\_count),  *cast*((*sum*(afternoon\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_leave\_ratio,  *sum*(evening\_leave\_count),  *cast*((*sum*(evening\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_leave\_ratio,  *sum*(morning\_truant\_count),  *cast*((*sum*(morning\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_truant\_ratio,  *sum*(afternoon\_truant\_count),  *cast*((*sum*(afternoon\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_truant\_ratio,  *sum*(evening\_truant\_count),  *cast*((*sum*(evening\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_truant\_ratio,  '4',  yearinfo,  monthinfo,  '-1' from itcast\_dws.class\_attendance\_dws group by yearinfo, monthinfo, class\_id; |

年：

|  |
| --- |
| INSERT OVERWRITE TABLE itcast\_app.class\_attendance\_app partition (yearinfo, monthinfo, dayinfo) select  yearinfo,  class\_id,  *sum*(studying\_student\_count),  *sum*(morning\_att\_count),  *cast*((*sum*(morning\_att\_count) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_att\_ratio,  *sum*(afternoon\_att\_count),  *cast*((*sum*(afternoon\_att\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_att\_ratio,  *sum*(evening\_att\_count),  *cast*((*sum*(evening\_att\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_att\_ratio,  *sum*(morning\_late\_count),  *cast*((*sum*(morning\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_late\_ratio,  *sum*(afternoon\_late\_count),  *cast*((*sum*(afternoon\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_late\_ratio,  *sum*(evening\_late\_count),  *cast*((*sum*(evening\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_late\_ratio,  *sum*(morning\_leave\_count),  *cast*((*sum*(morning\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_leave\_ratio,  *sum*(afternoon\_leave\_count),  *cast*((*sum*(afternoon\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_leave\_ratio,  *sum*(evening\_leave\_count),  *cast*((*sum*(evening\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_leave\_ratio,  *sum*(morning\_truant\_count),  *cast*((*sum*(morning\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_truant\_ratio,  *sum*(afternoon\_truant\_count),  *cast*((*sum*(afternoon\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_truant\_ratio,  *sum*(evening\_truant\_count),  *cast*((*sum*(evening\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_truant\_ratio,  '5',  yearinfo,  '-1',  '-1' from itcast\_dws.class\_attendance\_dws group by yearinfo, class\_id; |

#### 导出数据

##### 创建mysql表

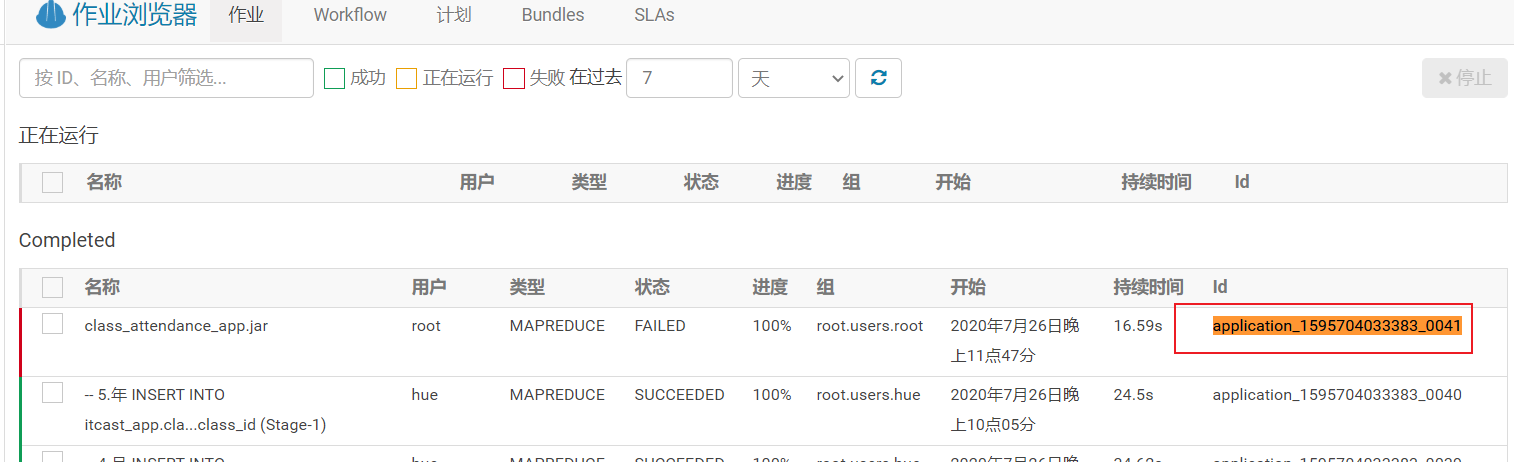
|  |
| --- |
| *--class\_attendance\_app OLAP应用数据* drop table class\_attendance\_app;  CREATE TABLE IF NOT EXISTS class\_attendance\_app (  class\_id int comment '班级id',  studying\_student\_count int comment '在读班级人数',  morning\_att\_count int comment '上午出勤人数',  morning\_att\_ratio float comment '上午出勤率',  afternoon\_att\_count int comment '下午出勤人数',  afternoon\_att\_ratio float comment '下午出勤率',  evening\_att\_count int comment '晚自习出勤人数',  evening\_att\_ratio float comment '晚自习出勤率',  morning\_late\_count int comment '上午迟到人数',  morning\_late\_ratio float comment '上午迟到率',  afternoon\_late\_count int comment '下午迟到人数',  afternoon\_late\_ratio float comment '下午迟到率',  evening\_late\_count int comment '晚自习迟到人数',  evening\_late\_ratio float comment '晚自习迟到率',  morning\_leave\_count int comment '上午请假人数',  morning\_leave\_ratio float comment '上午请假率',  afternoon\_leave\_count int comment '下午请假人数',  afternoon\_leave\_ratio float comment '下午请假率',  evening\_leave\_count int comment '晚自习请假人数',  evening\_leave\_ratio float comment '晚自习请假率',  morning\_truant\_count int comment '上午旷课人数',  morning\_truant\_ratio float comment '上午旷课率',  afternoon\_truant\_count int comment '下午旷课人数',  afternoon\_truant\_ratio float comment '下午旷课率',  evening\_truant\_count int comment '晚自习旷课人数',  evening\_truant\_ratio float comment '晚自习旷课率',  time\_type varchar(8) COMMENT '聚合时间类型：1、按小时聚合；2、按天聚合；3、按周聚合；4、按月聚合；5、按年聚合。',  yearinfo varchar(8) COMMENT '年',  monthinfo varchar(8) COMMENT '月',  dayinfo varchar(8) COMMENT '日' ) comment '班级请假数据统计'; |

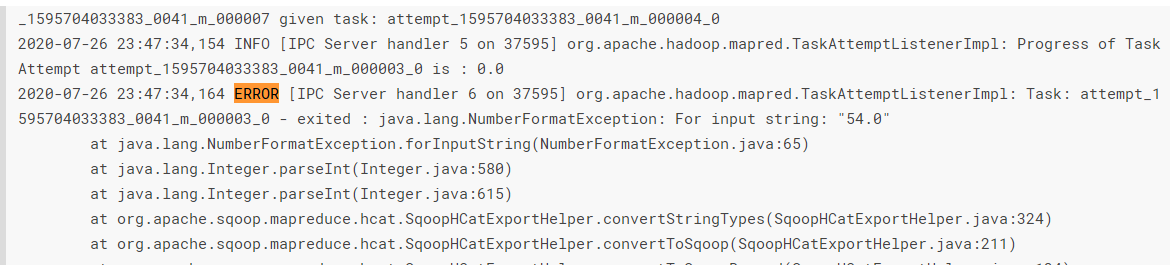
##### sqoop导出

|  |
| --- |
| sqoop export \  --connect "jdbc:mysql://192.168.52.150:3306/scrm\_bi?useUnicode=true&characterEncoding=utf-8" \  --username root \  --password 123456 \  --table class\_attendance\_app \  --hcatalog-database itcast\_app \  --hcatalog-table class\_attendance\_app \  -m 100 |

###### 报错

|  |
| --- |
| org.apache.hadoop.mapreduce.v2.app.job.impl.TaskAttemptImpl: Diagnostics report from attempt\_1592626945136\_0131\_m\_000001\_0: Error: java.lang.NumberFormatException: For input string: "0.0" |





解决：

将mysql表的int字段改为float，重新创建并导出数据。

|  |
| --- |
| *--class\_attendance\_app OLAP应用数据* drop table class\_attendance\_app;  CREATE TABLE IF NOT EXISTS class\_attendance\_app (  class\_id int comment '班级id',  studying\_student\_count float comment '在读班级人数',  morning\_att\_count float comment '上午出勤人数',  morning\_att\_ratio float comment '上午出勤率',  afternoon\_att\_count float comment '下午出勤人数',  afternoon\_att\_ratio float comment '下午出勤率',  evening\_att\_count float comment '晚自习出勤人数',  evening\_att\_ratio float comment '晚自习出勤率',  morning\_late\_count float comment '上午迟到人数',  morning\_late\_ratio float comment '上午迟到率',  afternoon\_late\_count float comment '下午迟到人数',  afternoon\_late\_ratio float comment '下午迟到率',  evening\_late\_count float comment '晚自习迟到人数',  evening\_late\_ratio float comment '晚自习迟到率',  morning\_leave\_count float comment '上午请假人数',  morning\_leave\_ratio float comment '上午请假率',  afternoon\_leave\_count float comment '下午请假人数',  afternoon\_leave\_ratio float comment '下午请假率',  evening\_leave\_count float comment '晚自习请假人数',  evening\_leave\_ratio float comment '晚自习请假率',  morning\_truant\_count float comment '上午旷课人数',  morning\_truant\_ratio float comment '上午旷课率',  afternoon\_truant\_count float comment '下午旷课人数',  afternoon\_truant\_ratio float comment '下午旷课率',  evening\_truant\_count float comment '晚自习旷课人数',  evening\_truant\_ratio float comment '晚自习旷课率',  time\_type varchar(8) COMMENT '聚合时间类型：1、按小时聚合；2、按天聚合；3、按周聚合；4、按月聚合；5、按年聚合。',  yearinfo varchar(8) COMMENT '年',  monthinfo varchar(8) COMMENT '月',  dayinfo varchar(8) COMMENT '日' ) comment '班级请假数据统计'; |

导出：

|  |
| --- |
| sqoop export \  --connect "jdbc:mysql://172.17.0.202:3306/scrm\_bi?useUnicode=true&characterEncoding=utf-8" \  --username root \  --password 123456 \  --table class\_attendance\_app \  --hcatalog-database itcast\_app \  --hcatalog-table class\_attendance\_app \  -m 100 |

在hive中int类型不要声明为string

直接在app层，将int类型的字段，声明为int，而不是string。

APP:

|  |
| --- |
| *--class\_attendance\_app OLAP应用数据* drop table itcast\_app.class\_attendance\_app; CREATE TABLE IF NOT EXISTS itcast\_app.class\_attendance\_app (  dateinfo String comment '日期',  class\_id int comment '班级id',  studying\_student\_count int comment '在读班级人数',  morning\_att\_count int comment '上午出勤人数',  morning\_att\_ratio String comment '上午出勤率',  afternoon\_att\_count int comment '下午出勤人数',  afternoon\_att\_ratio String comment '下午出勤率',  evening\_att\_count int comment '晚自习出勤人数',  evening\_att\_ratio String comment '晚自习出勤率',  morning\_late\_count int comment '上午迟到人数',  morning\_late\_ratio String comment '上午迟到率',  afternoon\_late\_count int comment '下午迟到人数',  afternoon\_late\_ratio String comment '下午迟到率',  evening\_late\_count int comment '晚自习迟到人数',  evening\_late\_ratio String comment '晚自习迟到率',  morning\_leave\_count int comment '上午请假人数',  morning\_leave\_ratio String comment '上午请假率',  afternoon\_leave\_count int comment '下午请假人数',  afternoon\_leave\_ratio String comment '下午请假率',  evening\_leave\_count int comment '晚自习请假人数',  evening\_leave\_ratio String comment '晚自习请假率',  morning\_truant\_count int comment '上午旷课人数',  morning\_truant\_ratio String comment '上午旷课率',  afternoon\_truant\_count int comment '下午旷课人数',  afternoon\_truant\_ratio String comment '下午旷课率',  evening\_truant\_count int comment '晚自习旷课人数',  evening\_truant\_ratio String comment '晚自习旷课率',  time\_type STRING COMMENT '聚合时间类型：1、按小时聚合；2、按天聚合；3、按周聚合；4、按月聚合；5、按年聚合。') comment '班级请假数据统计' PARTITIONED BY (yearinfo STRING, monthinfo STRING, dayinfo STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' stored as orcfile TBLPROPERTIES ('orc.compress'='SNAPPY'); |

按照年月日维度进行统计。

天：

|  |
| --- |
| INSERT OVERWRITE TABLE itcast\_app.class\_attendance\_app partition (yearinfo, monthinfo, dayinfo) select  dateinfo,  class\_id,  studying\_student\_count,  morning\_att\_count,  morning\_att\_ratio,  afternoon\_att\_count,  afternoon\_att\_ratio,  evening\_att\_count,  evening\_att\_ratio,  morning\_late\_count,  morning\_late\_ratio,  afternoon\_late\_count,  afternoon\_late\_ratio,  evening\_late\_count,  evening\_late\_ratio,  morning\_leave\_count,  morning\_leave\_ratio,  afternoon\_leave\_count,  afternoon\_leave\_ratio,  evening\_leave\_count,  evening\_leave\_ratio,  morning\_truant\_count,  morning\_truant\_ratio,  afternoon\_truant\_count,  afternoon\_truant\_ratio,  evening\_truant\_count,  evening\_truant\_ratio,  '2',  yearinfo,  monthinfo,  dayinfo from itcast\_dws.class\_attendance\_dws; |

月：

|  |
| --- |
| INSERT OVERWRITE TABLE itcast\_app.class\_attendance\_app partition (yearinfo, monthinfo, dayinfo) select  *CONCAT\_WS*('-',yearinfo,monthinfo),  class\_id,  *sum*(studying\_student\_count),  *sum*(morning\_att\_count),  *cast*((*sum*(morning\_att\_count) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_att\_ratio,  *sum*(afternoon\_att\_count),  *cast*((*sum*(afternoon\_att\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_att\_ratio,  *sum*(evening\_att\_count),  *cast*((*sum*(evening\_att\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_att\_ratio,  *sum*(morning\_late\_count),  *cast*((*sum*(morning\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_late\_ratio,  *sum*(afternoon\_late\_count),  *cast*((*sum*(afternoon\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_late\_ratio,  *sum*(evening\_late\_count),  *cast*((*sum*(evening\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_late\_ratio,  *sum*(morning\_leave\_count),  *cast*((*sum*(morning\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_leave\_ratio,  *sum*(afternoon\_leave\_count),  *cast*((*sum*(afternoon\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_leave\_ratio,  *sum*(evening\_leave\_count),  *cast*((*sum*(evening\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_leave\_ratio,  *sum*(morning\_truant\_count),  *cast*((*sum*(morning\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_truant\_ratio,  *sum*(afternoon\_truant\_count),  *cast*((*sum*(afternoon\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_truant\_ratio,  *sum*(evening\_truant\_count),  *cast*((*sum*(evening\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_truant\_ratio,  '4',  yearinfo,  monthinfo,  '-1' from itcast\_dws.class\_attendance\_dws group by yearinfo, monthinfo, class\_id; |

年：

|  |
| --- |
| INSERT OVERWRITE TABLE itcast\_app.class\_attendance\_app partition (yearinfo, monthinfo, dayinfo) select  yearinfo,  class\_id,  *sum*(studying\_student\_count),  *sum*(morning\_att\_count),  *cast*((*sum*(morning\_att\_count) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_att\_ratio,  *sum*(afternoon\_att\_count),  *cast*((*sum*(afternoon\_att\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_att\_ratio,  *sum*(evening\_att\_count),  *cast*((*sum*(evening\_att\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_att\_ratio,  *sum*(morning\_late\_count),  *cast*((*sum*(morning\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_late\_ratio,  *sum*(afternoon\_late\_count),  *cast*((*sum*(afternoon\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_late\_ratio,  *sum*(evening\_late\_count),  *cast*((*sum*(evening\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_late\_ratio,  *sum*(morning\_leave\_count),  *cast*((*sum*(morning\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_leave\_ratio,  *sum*(afternoon\_leave\_count),  *cast*((*sum*(afternoon\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_leave\_ratio,  *sum*(evening\_leave\_count),  *cast*((*sum*(evening\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_leave\_ratio,  *sum*(morning\_truant\_count),  *cast*((*sum*(morning\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_truant\_ratio,  *sum*(afternoon\_truant\_count),  *cast*((*sum*(afternoon\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_truant\_ratio,  *sum*(evening\_truant\_count),  *cast*((*sum*(evening\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_truant\_ratio,  '5',  yearinfo,  '-1',  '-1' from itcast\_dws.class\_attendance\_dws group by yearinfo, class\_id; |

导出：

|  |
| --- |
| sqoop export \  --connect "jdbc:mysql://172.17.0.202:3306/scrm\_bi?useUnicode=true&characterEncoding=utf-8" \  --username root \  --password 123456 \  --table class\_attendance\_app \  --hcatalog-database itcast\_app \  --hcatalog-table class\_attendance\_app \  -m 100 |

### 增量流程

#### 数据采集

##### Dimen

Dimen层采用全量覆盖的方式进行同步，全量的脚本可以复用。

##### ODS

###### student\_signin\_ods 学生打卡记录表

|  |
| --- |
| sqoop import \  --connect jdbc:mysql://172.17.0.202:3306/teach \  --username root \  --password 123456 \  --query 'select \*, FROM\_UNIXTIME(unix\_timestamp(),"%Y-%m-%d") as dt from tbh\_student\_signin\_record where signin\_date = "2019-12-02" and $CONDITIONS' \  --hcatalog-database itcast\_ods \  --hcatalog-table student\_signin\_ods \  -m 100 \  --split-by id |

###### student\_leave\_apply\_ods 学生请假申请表

|  |
| --- |
| sqoop import \  --connect jdbc:mysql://172.17.0.202:3306/teach \  --username root \  --password 123456 \  --query 'select \*, FROM\_UNIXTIME(unix\_timestamp(),"%Y-%m-%d") as dt from student\_leave\_apply  where  create\_time >= "2019-12-02 00:00:00"  and  create\_time < "2019-12-03 00:00:00"  and $CONDITIONS' \  --hive-partition-key dt \  --hive-partition-value 2019-12-03 \  --hcatalog-database itcast\_ods \  --hcatalog-table student\_leave\_apply\_ods \  -m 100 \  --split-by id |

#### 数据清洗转换

因此部分数据都已经过OLTP系统清洗过，所以跳过DWD直接进入DWM转换过程。

#### 统计分析

##### DWM

###### student\_attendance\_dwm 学生出勤(正常出勤和迟到)数据

|  |
| --- |
| *--分区* SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict; set hive.exec.max.dynamic.partitions.pernode=10000; set hive.exec.max.dynamic.partitions=100000; set hive.exec.max.created.files=150000; *--hive压缩* set hive.exec.compress.intermediate=true; set hive.exec.compress.output=true; *--写入时压缩生效* set hive.exec.orc.compression.strategy=COMPRESSION; *--分桶* set hive.enforce.bucketing=true; set hive.enforce.sorting=true; set hive.optimize.bucketmapjoin = true; set hive.auto.convert.sortmerge.join=true; set hive.auto.convert.sortmerge.join.noconditionaltask=true; *--并行执行* set hive.exec.parallel=true; set hive.exec.parallel.thread.number=8; *--小文件合并 -- set mapred.max.split.size=2147483648; -- set mapred.min.split.size.per.node=1000000000; -- set mapred.min.split.size.per.rack=1000000000; --矢量化查询* set hive.vectorized.execution.enabled=true; *--关联优化器* set hive.optimize.correlation=true; *--读取零拷贝* set hive.exec.orc.zerocopy=true; *--join数据倾斜* set hive.optimize.skewjoin=true; *-- set hive.skewjoin.key=100000;* set hive.optimize.skewjoin.compiletime=true; set hive.optimize.union.remove=true; *-- group倾斜* set hive.groupby.skewindata=true;   INSERT INTO itcast\_dwm.student\_attendance\_dwm PARTITION(yearinfo, monthinfo, dayinfo) SELECT   course.class\_date as dateinfo,  course.class\_id,  so.student\_id,  *-- 在上课前40分钟~上课时间点10分钟之内，为正常打卡0；在上课后10分钟~放学时间之内，为迟到1；否则为2.  if*(  *--判断是否正常出勤(包含迟到的)  sum*(  *if*(  (  *cast*(  (*unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.morning\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS'))  / 60  as INT  ) <= 40  )  and   (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')   <   *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.morning\_end\_time), 'yyyy-MM-dd HH:mm:ss')  ),  1, 0)  ) > 0,    *--正常出勤，但迟到了，判断并标记  if*(  *sum*(  *if*(  (  *cast*(  (  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.morning\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  )  / 60  as INT  ) <= 40  )  and   (  *cast*(  (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  -  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.morning\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  )  / 60  as INT  ) <= 10  ),  1, 0)  ) > 0,  0, 1  ),  2  ) as morning\_att,   *-- 在上课前40分钟~上课时间点10分钟之内，为正常打卡0；在上课后10分钟~放学时间之内，为迟到1；否则为2.  if*(  *--判断是否正常出勤(包含迟到的)  sum*(  *if*(  (  *cast*(  (*unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.afternoon\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS'))  / 60  as INT  ) <= 40  )  and   (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')   <   *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.afternoon\_end\_time), 'yyyy-MM-dd HH:mm:ss')  ),  1, 0)  ) > 0,    *--正常出勤，但迟到了，判断并标记  if*(  *sum*(  *if*(  (  *cast*(  (  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.afternoon\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  )  / 60  as INT  ) <= 40  )  and   (  *cast*(  (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  -  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.afternoon\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  )  / 60  as INT  ) <= 10  ),  1, 0)  ) > 0,  0, 1  ),  2  ) as afternoon\_att,   *-- 在上课前40分钟~上课时间点10分钟之内，为正常打卡0；在上课后10分钟~放学时间之内，为迟到1；否则为2.  if*(  *--判断是否正常出勤(包含迟到的)  sum*(  *if*(  (  *cast*(  (*unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.evening\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS'))  / 60  as INT  ) <= 40  )  and   (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')   <   *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.evening\_end\_time), 'yyyy-MM-dd HH:mm:ss')  ),  1, 0)  ) > 0,    *--正常出勤，但迟到了，判断并标记  if*(  *sum*(  *if*(  (  *cast*(  (  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.evening\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  -  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  )  / 60  as INT  ) <= 40  )  and   (  *cast*(  (  *unix\_timestamp*(so.signin\_time, 'yyyy-MM-dd HH:mm:ss.SSS')  -  *unix\_timestamp*(*concat\_ws*(' ', so.signin\_date, ti.evening\_begin\_time), 'yyyy-MM-dd HH:mm:ss')  )  / 60  as INT  ) <= 10  ),  1, 0)  ) > 0,  0, 1  ),  2  ) as evening\_att,  *substr*(course.class\_date, 1, 4) as yearinfo,   *substr*(course.class\_date, 6, 2) monthinfo,   *substr*(course.class\_date, 9, 2) dayinfo from itcast\_dimen.course\_table\_upload\_detail\_dimen course  LEFT JOIN itcast\_ods.student\_signin\_ods so on course.class\_id = so.class\_id and so.signin\_date=course.class\_date AND so.share\_state=1  LEFT JOIN itcast\_dimen.class\_time\_dimen ti on so.time\_table\_id=ti.id WHERE course.content IS NOT NULL AND course.content != '开班典礼' AND course.class\_date >='${TD\_DATE}' GROUP BY course.class\_date, course.class\_id, so.student\_id; |

###### class\_attendance\_dwm 班级出勤(正常出勤和迟到)数据

|  |
| --- |
| *--分区* SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict; set hive.exec.max.dynamic.partitions.pernode=10000; set hive.exec.max.dynamic.partitions=100000; set hive.exec.max.created.files=150000; *--hive压缩* set hive.exec.compress.intermediate=true; set hive.exec.compress.output=true; *--写入时压缩生效* set hive.exec.orc.compression.strategy=COMPRESSION; *--分桶* set hive.enforce.bucketing=true; set hive.enforce.sorting=true; set hive.optimize.bucketmapjoin = true; set hive.auto.convert.sortmerge.join=true; set hive.auto.convert.sortmerge.join.noconditionaltask=true; *--并行执行* set hive.exec.parallel=true; set hive.exec.parallel.thread.number=8; *--小文件合并 -- set mapred.max.split.size=2147483648; -- set mapred.min.split.size.per.node=1000000000; -- set mapred.min.split.size.per.rack=1000000000; --矢量化查询* set hive.vectorized.execution.enabled=true; *--关联优化器* set hive.optimize.correlation=true; *--读取零拷贝* set hive.exec.orc.zerocopy=true; *--join数据倾斜* set hive.optimize.skewjoin=true; *-- set hive.skewjoin.key=100000;* set hive.optimize.skewjoin.compiletime=true; set hive.optimize.union.remove=true; *-- group倾斜* set hive.groupby.skewindata=false;   SELECT *\** from itcast\_dwm.class\_attendance\_dwm;  INSERT INTO itcast\_dwm.class\_attendance\_dwm PARTITION (yearinfo, monthinfo, dayinfo) SELECT  stu.dateinfo,  stu.class\_id,  *count*(DISTINCT   (  *if*(morning\_att='0' or morning\_att='1', stu.student\_id, -1)  )  ) morning\_att\_count,   *count*(DISTINCT   (  *if*(afternoon\_att='0' or afternoon\_att='1', stu.student\_id, -1)  )  ) afternoon\_att\_count,   *count*(DISTINCT   (  *if*(evening\_att='0' or evening\_att='1', stu.student\_id, -1)  )  ) evening\_att\_count,   *count*(DISTINCT   (  *if*(morning\_att='1', stu.student\_id, -1)  )  ) morning\_late\_count,   *count*(DISTINCT   (  *if*(afternoon\_att='1', stu.student\_id, -1)  )  ) afternoon\_late\_count,   *count*(DISTINCT   (  *if*(evening\_att='1', stu.student\_id, -1)  )  ) evening\_late\_count,   stu.yearinfo, stu.monthinfo, stu.dayinfo from itcast\_dwm.student\_attendance\_dwm stu where stu.dateinfo >= '${TD\_DATE}' GROUP BY stu.class\_id, stu.yearinfo, stu.monthinfo, stu.dateinfo, stu.dayinfo; |

###### class\_leave\_dwm 班级请假数据

|  |
| --- |
| *--分区* SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict; set hive.exec.max.dynamic.partitions.pernode=10000; set hive.exec.max.dynamic.partitions=100000; set hive.exec.max.created.files=150000; *--hive压缩* set hive.exec.compress.intermediate=true; set hive.exec.compress.output=true; *--写入时压缩生效* set hive.exec.orc.compression.strategy=COMPRESSION; *--分桶* set hive.enforce.bucketing=true; set hive.enforce.sorting=true; set hive.optimize.bucketmapjoin = true; set hive.auto.convert.sortmerge.join=true; set hive.auto.convert.sortmerge.join.noconditionaltask=true; *--并行执行* set hive.exec.parallel=true; set hive.exec.parallel.thread.number=8; *--小文件合并 -- set mapred.max.split.size=2147483648; -- set mapred.min.split.size.per.node=1000000000; -- set mapred.min.split.size.per.rack=1000000000; --矢量化查询* set hive.vectorized.execution.enabled=true; *--关联优化器* set hive.optimize.correlation=true; *--读取零拷贝* set hive.exec.orc.zerocopy=true; *--join数据倾斜* set hive.optimize.skewjoin=true; *-- set hive.skewjoin.key=100000;* set hive.optimize.skewjoin.compiletime=true; set hive.optimize.union.remove=true; *-- group倾斜* set hive.groupby.skewindata=false;   INSERT INTO itcast\_ods.student\_leave\_apply\_ods partition (dt) values (125, 5032, 119142, 1, 3491, '2019-09-08 16:42:29', '狂犬疫苗最后一针', 1, 2, '2019-09-03 08:00:00', 2, '2019-09-03 23:30:00', 2, 1, 0, null, null, '医院打针，狂犬疫苗最后一针了', 1, '2019-09-02 08:56:54','2020-07-07');   insert into itcast\_dwm.class\_leave\_dwm PARTITION (yearinfo, monthinfo, dayinfo) SELECT  morning.dateinfo,  morning.class\_id,  morning.morning\_leave\_count,  afternoon.afternoon\_leave\_count,  evening.evening\_leave\_count,  *substr*(morning.dateinfo, 1, 4) yearinfo,  *substr*(morning.dateinfo, 6, 2) monthinfo,  *substr*(morning.dateinfo, 9, 2) dayinfo from (  SELECT  cource.class\_date dateinfo,  cource.class\_id,  *count*(distinct lev.student\_id) as morning\_leave\_count  from itcast\_ods.student\_leave\_apply\_ods lev, itcast\_dimen.course\_table\_upload\_detail\_dimen cource, itcast\_dimen.class\_time\_dimen ti  where lev.class\_id = cource.class\_id and lev.class\_id = ti.class\_id  and cource.class\_date between ti.use\_begin\_date and ti.use\_end\_date  AND cource.content IS NOT NULL AND cource.content != '开班典礼'  AND lev.begin\_time <= *concat\_ws*(' ', cource.class\_date, ti.morning\_begin\_time)  AND lev.end\_time >= *concat\_ws*(' ', cource.class\_date, ti.morning\_begin\_time)  AND lev.audit\_state = 1  AND lev.cancel\_state = 0  AND lev.valid\_state = 1  AND cud.class\_date >='${TD\_DATE}'  group by cource.class\_date, cource.class\_id ) morning full join (  SELECT  cource.class\_date dateinfo,  cource.class\_id,  *count*(distinct lev.student\_id) as afternoon\_leave\_count  from itcast\_ods.student\_leave\_apply\_ods lev, itcast\_dimen.course\_table\_upload\_detail\_dimen cource, itcast\_dimen.class\_time\_dimen ti  where lev.class\_id = cource.class\_id and lev.class\_id = ti.class\_id  and cource.class\_date between ti.use\_begin\_date and ti.use\_end\_date  AND cource.content IS NOT NULL AND cource.content != '开班典礼'  AND lev.begin\_time <= *concat\_ws*(' ', cource.class\_date, ti.afternoon\_begin\_time)  AND lev.end\_time >= *concat\_ws*(' ', cource.class\_date, ti.afternoon\_begin\_time)  AND lev.audit\_state = 1  AND lev.cancel\_state = 0  AND lev.valid\_state = 1  AND cud.class\_date >='${TD\_DATE}'  group by cource.class\_date, cource.class\_id ) afternoon on morning.dateinfo = afternoon.dateinfo and morning.class\_id = afternoon.class\_id full join (  SELECT  cource.class\_date dateinfo,  cource.class\_id,  *count*(distinct lev.student\_id) as evening\_leave\_count  from itcast\_ods.student\_leave\_apply\_ods lev, itcast\_dimen.course\_table\_upload\_detail\_dimen cource, itcast\_dimen.class\_time\_dimen ti  where lev.class\_id = cource.class\_id and lev.class\_id = ti.class\_id  and cource.class\_date between ti.use\_begin\_date and ti.use\_end\_date  AND cource.content IS NOT NULL AND cource.content != '开班典礼'  AND lev.begin\_time <= *concat\_ws*(' ', cource.class\_date, ti.evening\_begin\_time)  AND lev.end\_time >= *concat\_ws*(' ', cource.class\_date, ti.evening\_begin\_time)  AND lev.audit\_state = 1  AND lev.cancel\_state = 0  AND lev.valid\_state = 1  AND cud.class\_date >='${TD\_DATE}'  group by cource.class\_date, cource.class\_id ) evening on evening.dateinfo = afternoon.dateinfo and evening.class\_id = afternoon.class\_id; |

###### class\_truant\_dwm 班级旷课数据

|  |
| --- |
| *--分区* SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict; set hive.exec.max.dynamic.partitions.pernode=10000; set hive.exec.max.dynamic.partitions=100000; set hive.exec.max.created.files=150000; *--hive压缩* set hive.exec.compress.intermediate=true; set hive.exec.compress.output=true; *--写入时压缩生效* set hive.exec.orc.compression.strategy=COMPRESSION; *--分桶* set hive.enforce.bucketing=true; set hive.enforce.sorting=true; set hive.optimize.bucketmapjoin = true; set hive.auto.convert.sortmerge.join=true; set hive.auto.convert.sortmerge.join.noconditionaltask=true; *--并行执行* set hive.exec.parallel=true; set hive.exec.parallel.thread.number=8; *--小文件合并 -- set mapred.max.split.size=2147483648; -- set mapred.min.split.size.per.node=1000000000; -- set mapred.min.split.size.per.rack=1000000000; --矢量化查询* set hive.vectorized.execution.enabled=true; *--关联优化器* set hive.optimize.correlation=true; *--读取零拷贝* set hive.exec.orc.zerocopy=true; *--join数据倾斜* set hive.optimize.skewjoin=true; *-- set hive.skewjoin.key=100000;* set hive.optimize.skewjoin.compiletime=true; set hive.optimize.union.remove=true; *-- group倾斜* set hive.groupby.skewindata=false;   SELECT *\** from itcast\_dwm.class\_truant\_dwm WHERE dateinfo='2019-09-03';  INSERT INTO itcast\_dwm.class\_truant\_dwm PARTITION (yearinfo, monthinfo, dayinfo) SELECT  course.class\_date dateinfo,  course.class\_id,  *nvl*(ct.studying\_student\_count, 0) - *nvl*(att.morning\_att\_count, 0) - *nvl*(leave.morning\_leave\_count, 0) morning\_truant\_count,  *nvl*(ct.studying\_student\_count, 0) - *nvl*(att.afternoon\_att\_count, 0) - *nvl*(leave.afternoon\_leave\_count, 0) afternoon\_truant\_count,  *nvl*(ct.studying\_student\_count, 0) - *nvl*(att.evening\_att\_count, 0) - *nvl*(leave.evening\_leave\_count, 0) evening\_truant\_count,  *substr*(course.class\_date, 1,4) yearinfo,  *substr*(course.class\_date, 6,2) monthinfo,  *substr*(course.class\_date, 9,2) dayinfo from itcast\_dimen.course\_table\_upload\_detail\_dimen course  LEFT JOIN itcast\_dwm.class\_attendance\_dwm att on course.class\_id = att.class\_id and att.dateinfo = course.class\_date  LEFT JOIN itcast\_dwm.class\_leave\_dwm leave on course.class\_id = leave.class\_id AND course.class\_date = leave.dateinfo  LEFT JOIN itcast\_dimen.class\_studying\_student\_count\_dimen ct on ct.class\_id = course.class\_id and course.class\_date=ct.studying\_date WHERE ct.studying\_student\_count IS NOT NULL AND course.content IS NOT NULL AND course.content != '开班典礼' **and course.class\_date >='${TD\_DATE}'**; |

##### DWS

|  |
| --- |
| *--分区* SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict; set hive.exec.max.dynamic.partitions.pernode=10000; set hive.exec.max.dynamic.partitions=100000; set hive.exec.max.created.files=150000; *--hive压缩* set hive.exec.compress.intermediate=true; set hive.exec.compress.output=true; *--写入时压缩生效* set hive.exec.orc.compression.strategy=COMPRESSION; *--分桶* set hive.enforce.bucketing=true; set hive.enforce.sorting=true; set hive.optimize.bucketmapjoin = true; set hive.auto.convert.sortmerge.join=true; set hive.auto.convert.sortmerge.join.noconditionaltask=true; *--并行执行* set hive.exec.parallel=true; set hive.exec.parallel.thread.number=8; *--小文件合并 -- set mapred.max.split.size=2147483648; -- set mapred.min.split.size.per.node=1000000000; -- set mapred.min.split.size.per.rack=1000000000; --矢量化查询* set hive.vectorized.execution.enabled=true; *--关联优化器* set hive.optimize.correlation=true; *--读取零拷贝* set hive.exec.orc.zerocopy=true; *--join数据倾斜* set hive.optimize.skewjoin=true; *-- set hive.skewjoin.key=100000;* set hive.optimize.skewjoin.compiletime=true; set hive.optimize.union.remove=true; *-- group倾斜* set hive.groupby.skewindata=false;  INSERT INTO itcast\_dws.class\_attendance\_dws PARTITION (yearinfo, monthinfo, dayinfo) SELECT   course.class\_date dateinfo,   course.class\_id,  total.studying\_student\_count,  att.morning\_att\_count,  *cast*((att.morning\_att\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) morning\_att\_ratio,   att.afternoon\_att\_count,  *cast*((att.afternoon\_att\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) afternoon\_att\_ratio,   att.evening\_att\_count,  *cast*((att.evening\_att\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) evening\_att\_ratio,     att.morning\_late\_count,  *cast*((att.morning\_late\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) morning\_late\_ratio,   att.afternoon\_late\_count,  *cast*((att.afternoon\_late\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) afternoon\_late\_ratio,   att.evening\_late\_count,  *cast*((att.evening\_late\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) evening\_late\_ratio,    lev.morning\_leave\_count,  *cast*((lev.morning\_leave\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) morning\_leave\_ratio,   lev.afternoon\_leave\_count,  *cast*((lev.afternoon\_leave\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) afternoon\_leave\_ratio,   lev.evening\_leave\_count,  *cast*((lev.evening\_leave\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) evening\_leave\_ratio,     tru.morning\_truant\_count,  *cast*((tru.morning\_truant\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) morning\_truant\_ratio,   tru.afternoon\_truant\_count,  *cast*((tru.afternoon\_truant\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) afternoon\_truant\_ratio,   tru.evening\_truant\_count,  *cast*((tru.evening\_truant\_count / total.studying\_student\_count) \* 100 as DECIMAL(8,2)) evening\_truant\_ratio,     *substr*(course.class\_date, 1,4) yearinfo,  *substr*(course.class\_date, 6,2) monthinfo,  *substr*(course.class\_date, 9,2) dayinfo from itcast\_dimen.course\_table\_upload\_detail\_dimen course  LEFT JOIN itcast\_dimen.class\_studying\_student\_count\_dimen total on course.class\_date=total.studying\_date AND course.class\_id = total.class\_id  LEFT JOIN itcast\_dwm.class\_attendance\_dwm att on course.class\_date=att.dateinfo AND course.class\_id = att.class\_id  LEFT JOIN itcast\_dwm.class\_leave\_dwm lev on course.class\_id=lev.class\_id AND course.class\_date=lev.dateinfo  LEFT JOIN itcast\_dwm.class\_truant\_dwm tru on course.class\_id=tru.class\_id AND course.class\_date=tru.dateinfo WHERE course.content IS NOT NULL AND course.content != '开班典礼' AND total.studying\_student\_count IS NOT NULL AND course.class\_date >='${TD\_DATE}'; |

##### APP

天：

|  |
| --- |
| *--分区* SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict; set hive.exec.max.dynamic.partitions.pernode=10000; set hive.exec.max.dynamic.partitions=100000; set hive.exec.max.created.files=150000; *--hive压缩* set hive.exec.compress.intermediate=true; set hive.exec.compress.output=true; *--写入表时压缩生效* set hive.exec.orc.compression.strategy=COMPRESSION; *--分桶* set hive.enforce.bucketing=true; set hive.enforce.sorting=true; set hive.optimize.bucketmapjoin = true; set hive.auto.convert.sortmerge.join=true; set hive.auto.convert.sortmerge.join.noconditionaltask=true; *--并行执行* set hive.exec.parallel=true; set hive.exec.parallel.thread.number=16; *--数据倾斜* set hive.optimize.skewjoin=true; set hive.groupby.skewindata=true; *--小文件* set mapred.max.split.size=2147483648; set mapred.min.split.size.per.node=1000000000; set mapred.min.split.size.per.rack=1000000000; *-- 矢量化查询* set hive.vectorized.execution.enabled=true; *--关联shuffle优化器* set hive.optimize.correlation=true; *--零拷贝* set hive.exec.orc.zerocopy=true;  INSERT OVERWRITE TABLE itcast\_app.class\_attendance\_app partition (yearinfo, monthinfo, dayinfo) select  dateinfo,  class\_id,  studying\_student\_count,  morning\_att\_count,  morning\_att\_ratio,  afternoon\_att\_count,  afternoon\_att\_ratio,  evening\_att\_count,  evening\_att\_ratio,  morning\_late\_count,  morning\_late\_ratio,  afternoon\_late\_count,  afternoon\_late\_ratio,  evening\_late\_count,  evening\_late\_ratio,  morning\_leave\_count,  morning\_leave\_ratio,  afternoon\_leave\_count,  afternoon\_leave\_ratio,  evening\_leave\_count,  evening\_leave\_ratio,  morning\_truant\_count,  morning\_truant\_ratio,  afternoon\_truant\_count,  afternoon\_truant\_ratio,  evening\_truant\_count,  evening\_truant\_ratio,  '2',  yearinfo,  monthinfo,  dayinfo from itcast\_dws.class\_attendance\_dws where dateinfo >='${TD\_DATE}'; |

月：

|  |
| --- |
| INSERT OVERWRITE TABLE itcast\_app.class\_attendance\_app partition (yearinfo, monthinfo, dayinfo) select  dateinfo,  class\_id,  *sum*(studying\_student\_count),  *sum*(morning\_att\_count),  *cast*((*sum*(morning\_att\_count) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_att\_ratio,  *sum*(afternoon\_att\_count),  *cast*((*sum*(afternoon\_att\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_att\_ratio,  *sum*(evening\_att\_count),  *cast*((*sum*(evening\_att\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_att\_ratio,  *sum*(morning\_late\_count),  *cast*((*sum*(morning\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_late\_ratio,  *sum*(afternoon\_late\_count),  *cast*((*sum*(afternoon\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_late\_ratio,  *sum*(evening\_late\_count),  *cast*((*sum*(evening\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_late\_ratio,  *sum*(morning\_leave\_count),  *cast*((*sum*(morning\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_leave\_ratio,  *sum*(afternoon\_leave\_count),  *cast*((*sum*(afternoon\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_leave\_ratio,  *sum*(evening\_leave\_count),  *cast*((*sum*(evening\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_leave\_ratio,  *sum*(morning\_truant\_count),  *cast*((*sum*(morning\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_truant\_ratio,  *sum*(afternoon\_truant\_count),  *cast*((*sum*(afternoon\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_truant\_ratio,  *sum*(evening\_truant\_count),  *cast*((*sum*(evening\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_truant\_ratio,  '4',  yearinfo,  monthinfo,  '-1' from itcast\_dws.class\_attendance\_dws **where dateinfo >='${TD\_DATE}'** group by yearinfo, monthinfo, class\_id; |

年：

|  |
| --- |
| INSERT OVERWRITE TABLE itcast\_app.class\_attendance\_app partition (yearinfo, monthinfo, dayinfo) select  dateinfo,  class\_id,  *sum*(studying\_student\_count),  *sum*(morning\_att\_count),  *cast*((*sum*(morning\_att\_count) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_att\_ratio,  *sum*(afternoon\_att\_count),  *cast*((*sum*(afternoon\_att\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_att\_ratio,  *sum*(evening\_att\_count),  *cast*((*sum*(evening\_att\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_att\_ratio,  *sum*(morning\_late\_count),  *cast*((*sum*(morning\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_late\_ratio,  *sum*(afternoon\_late\_count),  *cast*((*sum*(afternoon\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_late\_ratio,  *sum*(evening\_late\_count),  *cast*((*sum*(evening\_late\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_late\_ratio,  *sum*(morning\_leave\_count),  *cast*((*sum*(morning\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_leave\_ratio,  *sum*(afternoon\_leave\_count),  *cast*((*sum*(afternoon\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_leave\_ratio,  *sum*(evening\_leave\_count),  *cast*((*sum*(evening\_leave\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_leave\_ratio,  *sum*(morning\_truant\_count),  *cast*((*sum*(morning\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) morning\_truant\_ratio,  *sum*(afternoon\_truant\_count),  *cast*((*sum*(afternoon\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) afternoon\_truant\_ratio,  *sum*(evening\_truant\_count),  *cast*((*sum*(evening\_truant\_ratio) / *sum*(studying\_student\_count)) \* 100 as decimal(8, 2)) evening\_truant\_ratio,  '5',  yearinfo,  '-1',  '-1' from itcast\_dws.class\_attendance\_dws where dateinfo >='${TD\_DATE}' group by yearinfo, class\_id; |

#### 导出数据

|  |
| --- |
| #! /bin/bash  SQOOP\_HOME=/usr/bin/sqoop  HOST=172.17.0.202  USERNAME="root"  PASSWORD="123456"  PORT=3306  DBNAME="scrm\_bi"  MYSQL=/usr/local/mysql\_5723/bin/mysql  if [[ $1 == "" ]];then  TD\_DATE=`date -d '1 days ago' "+%Y-%m-%d"`  else  TD\_DATE=$1  fi  ${MYSQL} -h${HOST} -P${PORT} -u${USERNAME} -p${PASSWORD} -D${DBNAME} -e "delete from class\_attendance\_app where yearinfo = '${TD\_DATE:0:4}'"  ${SQOOP\_HOME} export \  --connect "jdbc:mysql://${HOST}:${PORT}/${DBNAME}?useUnicode=true&characterEncoding=utf-8" \  --username ${USERNAME} \  --password ${PASSWORD} \  --table class\_attendance\_app \  --hcatalog-database itcast\_app \  --hcatalog-table class\_attendance\_app \  --hcatalog-partition-keys yearinfo \  --hcatalog-partition-values ${TD\_DATE:0:4} \  -m 100 |