

条件计数：对同一个字段求和再除以计数，就是求平均：
avg (字段1) =sum (字段1) /count (字段1)

需要返回1和0的判断结果时可以使用表达式直接输出，对每一条记录都会进行判断
判断表达式：result= 'pass' 返回的结果就是1和0



案例分析

记录中行与行需要进行计算时 (如转化率, 占比等) , **先对记录排序, 用开窗函数返回上一行的值:**

将字段值变成数值: lag(count(distinct user_id),1) over(order by if(behavior_type='pv',1,if(behavior_type='cart',2,3))) as 上一行为用户数
返回上一行字段的值 指定字段排序顺序

count(distinct user_id)/lag(count(distinct user_id),1) over(order by if(behavior_type='pv',1,if(behavior_type='cart',2,3))) as 转化率
用户数 上一行为用户数

将第一行转化率为100%显示为不为空, 用ifnull(表达式, 1) 函数判断: 为空时, 返回1, 不为空时返回表达式的值

ifnull(count(distinct user_id)/lag(count(distinct user_id),1) over(order by if(behavior_type='pv',1,if(behavior_type='cart',2,3))),1) as 转化率

当需要统计的内容都在一个字段内时, 将需要统计的字段拆分成多个字段再进行统计

1.筛选出需要的字段;

```
select 日期,behavior_type, count(distinct user_id) as 用户数 from userbehavior_view where behavior_type in ('pv','cart','buy') group by 日期,behavior_type;
```

2.行转列 (一列变四列) :

```
select 日期, if(behavior_type='pv',用户数,0) as 浏览人数, if(behavior_type='cart',用户数,0) as 加购人数, if(behavior_type='buy',用户数,0) as 购买人数
```

```
from (select 日期,behavior_type, count(distinct user_id) as 用户数 from userbehavior_view where behavior_type in ('pv','cart','buy') group by 日期,behavior_type) as t;
```

3.按日期分组聚合 (多行变一行) :

```
select 日期, sum(if(behavior_type='pv',用户数,0)) as 浏览人数, sum(if(behavior_type='cart',用户数,0)) as 加购人数, sum(if(behavior_type='buy',用户数,0)) as 购买人数
```

```
from (select 日期,behavior_type, count(distinct user_id) as 用户数 from userbehavior_view where behavior_type in ('pv','cart','buy') group by 日期,behavior_type) as t group by 日期;
```

1.多个表进行连接: 直接用连接语句即可

```
select * from t_score left join t_lesson on t_score.lesson_id=t_lesson.lesson_id left join t_stu_profile on t_score.stu_id=t_stu_profile.stu_id;
```

2.按课程和分数进行排序: 使用开窗函数

```
select *,dense_rank() over(partition by lesson_name order by score desc) as 排名 from t_score left join t_lesson on t_score.lesson_id=t_lesson.lesson_id left join t_stu_profile on t_score.stu_id=t_stu_profile.stu_id;
```

3.筛选每门课程前三名:

where语句不可以使用聚合函数, 需要使用子查询方可引用别名

注意引用子查询时, 子查询select语句中必须指定要查询的字段

```
select * from (select lesson_name,stu_name,score,dense_rank() over(partition by lesson_name order by score desc) as 排名 from t_score left join t_lesson on t_score.lesson_id=t_lesson.lesson_id left join t_stu_profile on t_score.stu_id=t_stu_profile.stu_id) as t where 排名 <=3;
```

4.行转列: 新建字段

文本组合函数: concat

```
select lesson_name, if(排名=1,concat(stu_name,'+',score),null) as 第一名, if(排名=2,concat(stu_name,'+',score),null) as 第二名, if(排名=3,concat(stu_name,'+',score),null) as 第三名
```

```
from (select lesson_name,stu_name,score,dense_rank() over(partition by lesson_name order by score desc) as 排名 from t_score left join t_lesson on t_score.lesson_id=t_lesson.lesson_id left join t_stu_profile on t_score.stu_id=t_stu_profile.stu_id) as t where 排名 <=3;
```

5.分组聚合

分组合并函数: group_concat可忽略空值, 返回一个字符串结果

```
select lesson_name, group_concat(if(排名=1,concat(stu_name,'+',score),null)) as 第一名, group_concat(if(排名=2,concat(stu_name,'+',score),null)) as 第二名, group_concat(if(排名=3,concat(stu_name,'+',score),null)) as 第三名 from (select lesson_name,stu_name,score,dense_rank() over(partition by lesson_name order by score desc) as 排名 from t_score left join t_lesson on t_score.lesson_id=t_lesson.lesson_id left join t_stu_profile on t_score.stu_id=t_stu_profile.stu_id) as t where 排名 <=3 group by lesson_name;
```

1.计算销售额和累计销售额

对销售额进行开窗计算累计销售额

```
select item_category, sum(amount) as 销售额, sum(sum(amount)) over(order by sum(amount) desc) as 累计销售额 from userbehavior_view where behavior_type='buy' group by item_category;
```

2.计算销售百分比

当over中指定了排序, 但没有指定滑动窗口, 默认计算当前分区内第一行到当前行的值

over中不指定分区, 排序和指定滑动窗口, 默认计算当前分区内第一行到最后一行所有行的值

```
select item_category, sum(amount) as 销售额, sum(sum(amount)) over(order by sum(amount) desc) as 累计销售额, sum(sum(amount)) over() as 总销售额, sum(sum(amount)) over(order by sum(amount) desc)/sum(sum(amount)) over() as 销售百分比 from userbehavior_view where behavior_type='buy' group by item_category;
```

3.筛选20% (帕累托分析法)

having 语句中不能使用开窗函数, 只能使用子查询

```
select * from (select item_category, sum(amount) as 销售额, sum(sum(amount)) over(order by sum(amount) desc) as 累计销售额, sum(sum(amount)) over() as 总销售额, sum(sum(amount)) over(order by sum(amount) desc)/sum(sum(amount)) over() as 销售百分比 from userbehavior_view where behavior_type='buy' group by item_category) as t where 销售百分比 <=0.8;
```