面试真题答案及代码解析

目录

一、Soul 用户活跃、留存和粘性分析

本场景使用用户登录记录表，表结构如下

td_load_rcd(用户登录记录表)		
usr_id(用户id)	load_dt(登录日期)	load_tm(登录时间)
C10000	2020/6/9	16:16:13

记录了2020年6月1日-2020年7月11日的用户登录状态； 用户每成功登录一次，就会记录一次。

1、2020 年 6 月的活跃用户数为？

```
select
    substr(load_dt, 1, 7) load_month --题干要求 6 月份，但是实际场景中，都是取更长周期，比如近 6 个月、近 1 年等
    , count(distinct usr_id) cst_cnt --活跃用户数是去重用户数，初学者容易写成 count(1)
from
    td_load_rcd
group by
    substr(load_dt, 1, 7)
;
```

2、7月份工作日期间，各时间段的月活分布，通勤（7：00-9：00、18：00-20：00），午休（11：00-13：00），临睡（22：00-1：00），哪段时间的活跃用户数最高？

```
select
    case when hour(load_tm) between 7 and 8 or hour(load_tm) between 18 and 19 then 'commute'
         when hour(load_tm) between 11 and 12  then 'lunch'
         when hour(load_tm) in (22, 23, 0) then 'before_sleep'
        end as time_prd
    ,count(distinct usr_id) as cst_dt
from
    td_load_rcd where load_dt in ('2020-07-01', '2020-07-02',
        '2020-07-03', '2020-07-06', '2020-07-07', --题干要求工作日，这里通过枚举
        '2020-07-08', '2020-07-09', '2020-07-10') --大家想想有没有其他方式呢？提示(mysql 提取星期函数，大家可自行百度)
group by
    case when hour(load_tm) between 7 and 8 or hour(load_tm) between 18 and 19 then 'commute'
         when hour(load_tm) between 11 and 12  then 'lunch'
         when hour(load_tm) in (22, 23, 0) then 'before_sleep'
        end --注意两处 case when 的区别，此处没有 as time_prd
```

```
;
```

3、单日登录次数大于等于5次的用户数？

```
select count(distinct usr_id) as usr_cnt
from
    (
        select usr_id, load_dt, count(1) as load_times --本题的思路是子查询，保存每个用户在每天的登录次数
        from td_load_rcd
        group by usr_id, load_dt
        having count(1) >=5 --把count(1) 改成 load_times 行不行？大家可以试一下
    )t
;
```

4、6月12日的T+1日留存、6月15日的T+3日留存、6月20日的T+7日留存分别为

```
--经典题，T+N留存率查询，同学们需要反复琢磨
--首先明确留存率的定义：T日新增用户中，在第n日（即T+n日）再次活跃的用户，占T日新增用户的比例。
--谷歌的官方说法更简洁，叫：Percentage of new users who return each day
```

```
--总之，一定得是新用户
create view td_distinct_load_rcd_min as
    select usr_id,min(load_dt) load_dt
    from td_load_rcd
    group by usr_id --既然是新用户，首先得找到每天新增用户
;
create view td_distinct_load_rcd as
    select load_dt, usr_id
    from td_load_rcd
    group by load_dt, usr_id --这是sql代码优化的一种思路。建中间表，减少代码量，提升查询速度。中间表保存了用户在每天的去重登录情况
;
select
    t0.load_dt
    , count(t0.usr_id) as cst_dt_0
    , count(t1.usr_id) as cst_dt_1
    , count(t1.usr_id)/count(t0.usr_id) as cst_dt_pct_1
```

```
    , count(t2.usr_id) as cst_dt_2
    , count(t2.usr_id)/count(t0.usr_id) as cst_dt_pct_2
    , count(t3.usr_id) as cst_dt_3
    , count(t3.usr_id)/count(t0.usr_id) as cst_dt_pct_7
from
 td_distinct_load_rcd_min t0
    left join
    td_distinct_load_rcd t1
    on t0.usr_id=t1.usr_id and t0.load_dt=date_sub(t1.load_dt, interval 1 day) --修改本处的1, 3, 7即可得到任意一天的
任意N日留存率
    left join
    td_distinct_load_rcd t2
    on t0.usr_id=t2.usr_id and t0.load_dt=date_sub(t2.load_dt, interval 3 day)
    left join
    td_distinct_load_rcd t3
    on t0.usr_id=t3.usr_id and                    b(t3.load_dt, interval 7 day)
group by
```

```
    t0.load_dt
order by
    t0.load_dt
;
```

写法２

```
select a.*, b.v_d as vd2, datediff(b.v_d, a.v_d) d_diff
from
(select usr_id, min(load_dt) v_d from td_load_rcd group by usr_id)a
left join
 (select usr_id, load_dt v_d from td_load_rcd group by usr_id, load_dt)b
 on a.usr_id = b.usr_id
```

5、6月份连续7天登录的用户数为

```
--本题的复杂程度，已经超过文字所能描述的范围。
select count(distinct usr_id)
from
```

```
(
    select usr_id, load_dt2, count(1) load_days
    from
        (
            select usr_id, load_dt, rnk, date_sub(load_dt, interval rnk day) as load_dt2
            from
            (
                select
                        a.usr_id
                    , a.load_dt
                    , row_number()
                    over(partition by a.usr_id order by a.load_dt) rnk
                    from
                        (
                            select
                                usr_id
                                , load_dt
```

```
                        from td_load_rcd where substr(load_dt, 1, 7)='2020-06'
                        group by
                            usr_id, load_dt
                    )a
            )b
        )c
    group by usr_id,load_dt2
    having load_days >= 7
)t
;
```

二、微信-情人节红包流向探索分析

本场景主要考察多表连接，凡是涉及到多表关联，建议用画图的方式理解

本场景共使用3张表，表结构如下:

tx_cty_map(城市省份等级映射表)		
cty(城市名称)	prov(所属省份)	cty_cls(城市等级)
重庆市	重庆市	新一线
郑州市	河南省	新一线
长沙市	湖南省	新一线
长春市	吉林省	二线

tx_red_pkt_rcd(红包发送记录简表)				
snd_usr_id(发送方用户id)	rcv_usr_id(接收方用户id)	pkt_amt(红包金额)	snd_datetime(发送时间)	rcv_datetime(接收时间)
T01234	T01235	66.00	2021-02-13 13:00:34	2021-02-13 13:00:40
T01235	T01345	520.00	2021-02-13 22:12:12	1900-01-01 00:00:00

tx_usr_bas_inf(用户基本信息简表)			
usr_id(用户id)	gdr(性别)	bth_dt(出生日期)	cty(所在城市)
T01234	F	1996-02-03	北京
T01235	M	1992-03-04	上海

用户基本信息简表模拟真实的数据治理场景，含有部分脏数据，因此第5题没有标准答案

具体可参考答案及解析

1、红包发送方用户的基本信息缺失率有多高？（即有多少红包发送方用户无法在用户基本信息表中匹配？）

```
select count(1), count(b.usr_id), 1-count(b.usr_id)/ count(1)
```

```
from
    (select distinct snd_usr_id from tx_red_pkt_rcd) a --以有红包记录的用户为左表
left join
    (select distinct usr_id from tx_usr_bas_inf) b --以用户记录表为右表
on a.snd_usr_id = b.usr_id
;
```

2、哪一组红包金额的拒收率最高？

```
select
    case when pkt_amt between 0 and 50 then 'bin1'
         when pkt_amt between 50 and 200 then 'bin2'
    else 'bin3' end as pkt_amt_bin, sum(if_ref)
    , count(1)
    ,sum(if_ref)/count(1) c1
from
    (select *, case when date(rcv_datetime)='1900-01-01' --关键考点，如何识别未接收红包。通常数据开发人员在设计物理模型时，会以特殊日期标注
```

```
    then 1 else 0 end as if_ref from tx_red_pkt_rcd)t
group by
    case when pkt_amt between 0 and 50 then 'bin1'
        when pkt_amt between 50 and 200 then 'bin2'
        else 'bin3' end
order by c1 desc
;
```

3、最受二线城市欢迎的红包金额为？（即发出次数最多）

本题没有标准答案，只是表达了一种数据处理的思路，4、5 题同理。原因是用户基本信息表模拟了错误的场景，一个用户对应了多个信息，需要想办法强制唯一匹配

```
select pkt_amt, count(1) c1
from
    tx_red_pkt_rcd a
inner join
    (select usr_id, max(cty) cty --从第 3 题开始，涉及到 tx_usr_bas_inf 表，这张表是有问题的（为什么会有问题？），一个用户对应了多条用户信息。需要强制对应唯一信息
```

```
    from tx_usr_bas_inf group by usr_id) b
on a.snd_usr_id = b.usr_id
inner join
    tx_cty_map c on
b.cty = c.cty
and c.cty_cls like '%二线%'
group by pkt_amt order by c1 desc;
```

4、北上广深 4 大城市中，哪座城市的男性用户发出的 520 红包比例最低？

```
--本题没有标准答案，只是表达了一种数据处理的思路
select cty, count(1), sum(if_520), sum(if_520)/count(1)
from
    (
    select *,
    case when a.pkt_amt=520 then 1 else 0 end as if_520 --给符合条件的红包打上 1 的标签
    from
        tx_red_pkt_rcd a
```

```
    inner join --不要用 in 子查询来圈定北上广男性用户，in 查询的效率比不上 inner join，大家切记!
        (select usr_id
        , max(cty) cty
        , max(gdr) gdr --如上，仍需要唯一对应，也可以是 min
        from tx_usr_bas_inf group by usr_id)b
    on a.snd_usr_id = b.usr_id
    where b.cty in ('北京市', '上海市', '广州市')
    and b.gdr='M')a
group by cty
```

5、将用户划分为两大群体，都市丽人（年龄 25-35 岁，性别女，一线城市）和时尚大妈（年龄 45-55 岁，性别女，三四线城市）收到的红包平均金额分别是？

```
select cls, avg(pkt_amt)
from
    (
    select a.*, case when c.cty_cls='一线' and b.gdr='F' and age between 25 and 35 then 'dslr'
    when c.cty_cls in ('三线', '四线') and b.gdr='F' and age between 45 and 55 then 'ssdm' end as cls
```

```
    from
        (select * from tx_red_pkt_rcd
            where year(rcv_datetime)<>1900  --细节，不要把这个条件漏了，得是接收成功的红包
            ) a
    inner join
        (select usr_id, max(cty) cty, max(gdr) gdr, datediff(now(), max(bth_dt))/365.25 --细节，求年龄除以365.25
            as age from tx_usr_bas_inf group by usr_id) b
    on a.snd_usr_id = b.usr_id
    inner join
    tx_cty_map c
    on b.cty = c.cty
    ) t
group by cls
;
```

三、京东电商购物漏斗

本场景共使用3张表，表结构如下:

tb_clk_rcd(用户点击行为记录简表)					
cust_uid(用户id)	if_snd(是否展示)	if_vw(是否浏览商品详情)	if_cart(是否加入购物车)	if_buy(是否购买)	prd_id
20003	1	1	0	0	A
20006	1	1	1	1	C

tb_cst_bas_inf(用户信息简表)		
cust_uid(用户id)	gdr(性别)	age(年龄)
20003	F	34
20006	M	23

tb_prd_map(产品基本信息映射简表)		
prd_id(产品编号)	prd_nm(产品名称)	price(价格)
A	新疆哈密瓜10斤	9.8
B	散养土鸡蛋40枚约10斤	29.9

1、从展示到浏览、浏览到加购、加购到购买的转化率分别为？(按照用户数而非点击量算)(1 分)

经典场景，漏斗转化率的求法,左连接

```
select count(a.cust_uid)
    , count(b.cust_uid)
    , count(c.cust_uid)
    , count(d.cust_uid)
from
```

```
(select distinct cust_uid, prd_id from tb_clk_rcd where if_snd=1)a --step1:触达
left join
(select distinct cust_uid, prd_id from tb_clk_rcd where if_vw=1)b --step2:浏览
on a.cust_uid=b.cust_uid and a.prd_id=b.prd_id --细节，关联条件必须为cust_uid & prd_id，两个都要写。很容易漏掉prd_id
left join
(select distinct cust_uid, prd_id from tb_clk_rcd where if_cart=1)c --step3:加购
on b.cust_uid=c.cust_uid and b.prd_id=c.prd_id
left join
(select distinct cust_uid, prd_id from tb_clk_rcd where if_buy=1)d --step4:购买(付款)，实际电商漏斗比这个长，之后至少还有两步，付款成功、签收成功
on c.cust_uid = d.cust_uid and c.prd_id=d.prd_id
;
```

2、以下哪个商品的加购率最高？

```
select t2.prd_nm, t1.*
from
  (select a.prd_id, count(b.cust_uid), count(a.cust_uid),
  count(b.cust_uid)/count(a.cust_uid) pct --求出每个商品的加购率
```

```
  from
  (select distinct cust_uid, prd_id from tb_clk_rcd where if_vw=1)a--从浏览
  left join
  (select distinct cust_uid,prd_id from tb_clk_rcd where if_cart=1)b--到加购，是加购率
  on a.cust_uid=b.cust_uid and a.prd_id = b.prd_id
  group by prd_id)t1
inner join
  tb_prd_map t2
on t1.prd_id = t2.prd_id
order by pct desc;
```

3、购买哪个商品的用户的平均年龄最高？

```
select c.prd_nm, avg(age)
from
(select cust_uid,age from tb_cst_bas_inf)a
inner join
(select cust_uid, prd_id from tb_clk_rcd where if_buy=1 group by cust_uid, prd_id)b
```

```
on a.cust_uid = b.cust_uid
inner join tb_prd_map c
on b.prd_id = c.prd_id
group by c.prd_nm order by 2 desc
```

4、以下哪组价格区间的购买人数最多？

```
select case when price<= 100 then 'bin1'
                    when price >100 and price <=500 then 'bin2'
            else 'bin3'
       end as price_bin, count(distinct cust_uid)
from
(select a.*, b.price
from
   tb_clk_rcd a
inner join
   tb_prd_map b on a.prd_id=b.prd_id
where a.if_buy=1)t
```

```
group by
  case when price<= 100 then 'bin1'
       when price >100 and price <=500 then 'bin2'
       else 'bin3'
  end
```

5.以下说法正确的是?

```
select prd_nm,count(distinct cust_uid)
from
(select a.*, b.age,b.gdr,c.prd_nm from tb_clk_rcd a
inner join tb_cst_bas_inf b
on a.cust_uid =b.cust_uid
inner join tb_prd_map c
 on a.prd_id = c.prd_id
where b.gdr='M' and a.if_vw=1 and b.age between 20 and 35) t --A组用户每个产品的浏览量
group by prd_nm
;
select prd_nm,count(distinct cust_uid)
```

```
from
(select a.*, b.age,b.gdr,c.prd_nm from tb_clk_rcd a
inner join tb_cst_bas_inf b
on a.cust_uid =b.cust_uid
inner join tb_prd_map c
 on a.prd_id = c.prd_id
where b.gdr='F' and a.if_vw=1 and b.age between 45 and 55) t --B组用户每个产品的浏览量
group by prd_nm
;
select count(distinct a.cust_uid) from tb_cst_bas_inf a
inner join
(select distinct cust_uid from tb_clk_rcd where if_vw=1) b --A组用户总浏览量
on a.cust_uid = b.cust_uid
where a.age between 20 and 35 and a.gdr='M'
group by gdr
;
```

```
select count(distinct a.cust_uid) from tb_cst_bas_inf a
inner join
(select distinct cust_uid from tb_clk_rcd where if_vw=1) b --B组用户总浏览量
on a.cust_uid = b.cust_uid
where a.age between 45 and 55 and a.gdr='F'
group by gdr
```

四、滴滴面试真题-订单呼叫完答率分析

https://blog.csdn.net/SeizeeveryDay/article/details/112914590

本场景共使用1张表，表结构如下:

didi_order_rcd					
order_id	cust_uid	call_time	grab_time	cancel_time	finish_time
1	asdf213	2021/5/2 12:23	2021/5/2 12:23	1970/1/1 0:00	2021/5/2 12:45
2	asdasfe3	2021/5/2 13:20	2021/5/2 13:23	1970/1/1 0:00	2021/5/2 13:56
3	asd2rg	2021/5/2 14:20	2021/5/2 14:24	1970/1/1 0:00	2021/5/2 14:58
4	asdf4234	2021/5/2 15:24	2021/5/2 15:24	1970/1/1 0:00	2021/5/2 15:30
5	kjhd24	2021/5/2 16:23	2021/5/2 16:25	1970/1/1 0:00	2021/5/2 18:01
6	kjhd25	2021/5/2 17:23	2021/5/2 17:25	2021/5/2 17:25	1970/1/1 0:00
7	kjhd26	2021/5/2 18:22	2021/5/2 18:25	2021/5/2 18:26	1970/1/1 0:00
8	kjhd27	2021/5/2 19:22	2021/5/2 19:26	2021/5/2 19:28	1970/1/1 0:00
9	kjhd28	2021/5/2 20:21	2021/5/2 20:26	2021/5/2 20:29	1970/1/1 0:00

1、订单应答率为

```
select  sum(if_grab)/count(1)
from
```

```
(select *, case when year(grab_time)=1970  --类似与微信红包场景，通过特殊日期识别取消用户
   then 0 else 1 end as if_grab
from didi_order_rcd)t
;
```

2、订单完单率为

```
select sum(if_finish)/count(1)
from
(select *, case when year(finish_time)<>1970 then 1 else 0 end as if_finish
from didi_order_rcd)t
;
```

3、呼叫量最高的是哪一个小时？

```
select hour(call_time), count(1) c1
from didi_order_rcd
group by hour(call_time)
order by 2 desc;  -- 规范的写法是 order by c1,甚至 c1 也是不规范的，群主图快，你可不要学哦
```

4、第二天继续呼叫的比例为？

```
select  count(b.cust_uid)/count(a.cust_uid)
from
(select distinct cust_uid from
    didi_order_rcd where
    substr(call_time, 1, 10) = '2021-05-02')a --类似与留存率的写法，这里又偷懒了，因为只有两天数据
left join
(select distinct cust_uid from
    didi_order_rcd where substr(call_time, 1, 10) = '2021-05-03')b
on a.cust_uid = b.cust_uid
;
```

5、哪个小时的呼叫应答时间最短？

```
select hour(call_time),
sum(TIMESTAMPDIFF(second,call_time, grab_time))/count(1)/60 --日期，时间相减函数需要掌握
from didi_order_rcd
```

```
where year(grab_time)<>1970
group by hour(call_time)
order by 2
;
```

五、货拉拉面试题

本场景共使用2张表，表结构如下:

hll_t1					
order_id	usr_id	driver_id	cty	status	order_dt
1	1	d16	北京	cancel_by_driver	2020/1/23
2	6	d12	上海	completed	2020/1/24
3	3	d15	深圳	canle_by_usr	2020/1/25
4	5	d14	广州	cancel_by_driver	2020/1/26

hll_t2		
usr_id	banned	role
1	0	usr
d11	1	driver
d12	0	driver
5	1	usr

1、用车方和司机被禁止(banned=1)的比率分别为？(保留两位小数)

```
select role, count(1), sum(banned),
round(sum(banned)/count(1), 4) --百分比保留两位小数，就是小数保留 4 位小数。同学们也可以研究直接生成百分比形式
from hll_t2 group by role
;
```

2、2020 年 1 月 25 日的订单完成率为？

```
Select
order_dt, --顺带求出每天的完成率
sum(if(status='completed',1,0))/count(1)  '完成率'
from hll_t1
group by order_dt
;
```

3、用车至少两次，且主动取消过至少 1 次的用车方有多少名？

```
--错误写法 13 名
select count(a.usr_id)  from
(select usr_id, count(1) from hll_t1 group by usr_id having count(1) >=2)  a
```

```
inner join
    (select usr_id, count(1) from hll_t1 where status= 'cancel' group by usr_id having count(1)>=1)b
on a.usr_id =b.usr_id
--正确写法9名
select count(a.usr_id)  from
(select usr_id, count(1) from hll_t1 group by usr_id having count(1) >=2)  a --限制用车至少两次
inner join
(select usr_id, count(1) from hll_t1
where status= 'cancel_by_usr' --限制主动取消至少1次，不要写成canel，必须是cancel_by_usr
group by usr_id having count(1)>=1)b
on a.usr_id =b.usr_id
```

4、北京、上海的非禁止用户的用车取消率分别为？（要求输出结果保留两位小数）

```
select cty
    , count(1),
    sum(if(status<>'completed',1, 0)) --if函数的用法，可节省代码量，请大家比较case when的写法
    ,sum(if(status<>'completed',1, 0))/count(1)  from
```

```
(
select a.* from hll_t1 a
inner join
hll_t2 b
on a.usr_id = b.usr_id --1.先找到被禁止的用车方
where b.banned=0
union --3.二者联合，union all 和union 的区别，请大家自行百度
select a.* from hll_t1 a
inner join
hll_t2 b
on a.driver_id = b.usr_id --2.再找到被禁止的司机方
where b.banned=0)t
group by cty
```

5、长沙、北京被用车方取消率排第一的司机编号为？

```
select * from
(select cty,driver_id,cancel_rate,
```

dense_rank() --有3种rank，请大家思考这里应该用什么rank？百度排序窗口函数

```
   over(partition by cty order by cancel_rate desc) rnk
from
(
select cty, driver_id, sum(if(status='cancel_by_usr', 1, 0))/count(1) as cancel_rate
from hll_t1
group by cty, driver_id)t
 )t
 where rnk=1
```

六、哔哩哔哩面试真题-观看偏好分析

本场景共使用3张表，bilibili_t2 记录了某天用户的观察记录。表结构如下:

bilibili_t1(用户访问日志表)		
usr_id	v_date	m_flg(是否会员)
A01	2020/1/3	1
A02	2020/1/3	0
A03	2020/1/4	1

bilibili_t2(用户观看记录表)		
usr_id	v_time	v_id
A01	18:01:07	V01

bilibili_t3(视频名称类型映射表)		
v_id	v_nm	v_typ
A01	消失的爱人-惊悚片-本·阿弗莱克	放映厅
A13	小木乃伊到我家	番剧
A14	影视经典-霸王别姬	放映厅

1、2020 年 1 月 4 日、1 月 6 日的新增会员分别为？

```
 select v_date, count(1) from
(
select usr_id,
min(v_date) v_date --新增会员的写法，就是看这个用户最早登录的那天。看答案才能想到的话，你已经落后了
from bilibili_t1 where m_flg=1 group by usr_id ) t
group by v_date
;
```

2、用户观看高峰期为？

```
select hour(v_time), count(1) from bilibili_t2 group by hour(v_time)
order by 2 desc
;
```

3、鬼畜区用户里，有多少用户看过汽车，番剧区用户里，有多少用户看过放映厅？

```
create view bilibili_view_2 as
select a.*, b.v_typ v_typ2 from (select usr_id, v_typ --这题很难，难到我不想说话。
```

```
from (select  a.*, b.v_typ --啤酒尿布4个字看起来简单，背后需要很长的代码实现
from bilibili_t2 a --我之后会开专题讲这类题的写法及应用
inner join bilibili_t3 b
on a.v_id = b.v_id)t
group by usr_id, v_typ) a
left join
(select usr_id, v_typ from (select  a.*, b.v_typ
from bilibili_t2 a
inner join  bilibili_t3 b
on a.v_id = b.v_id)t group by usr_id, v_typ) b
on a.usr_id = b.usr_id
order by usr_id
;

select a.*,b.c2,c1/c2
from
(select v_typ, v_typ2, count(distinct usr_id) c1
```

```
from bilibili_view_2
group by v_typ, v_typ2
order by v_typ, v_typ2) a
inner join
(select v_typ, count(distinct usr_id) c2
from bilibili_view_2
group by v_typ) b
on a.v_typ= b.v_typ
;
```

4、哪一类用户的观看视频个数最多？（以每个用户观看的视频个数平均数衡量）

```
select
v_typ,avg(c1)
from
(select b.v_typ, a.usr_id, count(1) c1
from bilibili_t2 a
inner join
```

```
bilibili_t3 b
on a.v_id = b.v_id
group by b.v_typ, a.usr_id)t
group by v_typ
order by 2 desc
;
```

5、当天最受欢迎的放映厅、番剧分别是？

```
select * from
(
select t.*, dense_rank() over(partition by v_typ order by c1 desc) as rnk --不要偷懒，使用窗口函数找出每一类最受欢迎的视频
from
(select v_typ, v_nm, count(1) c1
from bilibili_t2 a
inner join
bilibili_t3 b
on a.v_id = b.v_id
```

```
group by v_typ, v_nm) t
order by v_typ, rnk
) t
having rnk=1
;
```

七、滴滴热门目的地

本场景共使用2张表，记录了2021年5月某天的用户用车记录。表结构如下:

didi_sht_rcd(用户点击行为记录简表)				
cust_uid(用户id)	start_loc(出发地址)	end_loc(目的地址)	start_tm(出发时间)	car_cls(乘车等级)
C234234	亚朵酒店	壹方城	18:00:34	A
C234234	壹方城	红浪漫休闲会所	23:34:56	C

loc_nm_ctg	
loc_nm	loc_ctg
凤凰里二期	住宅
科兴科技园	写字楼
凑凑火锅大悦城店	餐饮

1、以下哪个地址的用车人数最多？

```
select
    start_loc,count(distinct cust_uid)
from
    didi_sht_rcd
group by
    start_loc
order by 2 desc
;
```

2、以前海湾休闲会所为目的地的订单高峰期是几点？

```
select
    hour(start_tm),
    count(1)
from didi_sht_rcd
where end_loc like '前海湾休闲%'
group by hour(start_tm) order by 2 desc
;
```

3、用车人次最高的住宅、用车人次第一的酒吧分别是？

```
select b.loc_ctg, a.start_loc, c1, dense_rank()over(partition by loc_ctg order by c1 desc)rnk
from
(select start_loc, count(1) c1
from didi_sht_rcd group by start_loc) a
inner join
( select loc_nm, loc_ctg from loc_nm_ctg group by loc_nm, loc_ctg)b
on a.start_loc = b.loc_nm
where loc_ctg in ('住宅', '酒吧')
;
```

4、从机场到酒店，单量最高的车型为？

```
select
    r.car_cls,
    count(distinct cust_uid) 'cnt'
from didi_sht_rcd r
inner join loc_nm_ctg s on r.start_loc=s.loc_nm
inner join loc_nm_ctg e on r.end_loc=e.loc_nm
```

```
where s.loc_ctg = '机场'
and e.loc_ctg = '酒店'
group by r.car_cls
order by cnt desc
;
```

5、以下哪种说法错误？

```
select * from
    (
    select b.loc_ctg as start_ctg,
    a.start_loc,c.loc_ctg as end_ctg,
    a.end_loc, count(1) c1,
    dense_rank()over(partition by start_ctg, end_ctg order by c1 desc ) rnk
    from didi_sht_rcd a
    inner join loc_nm_ctg b
    on a.start_loc = b.loc_nm
    inner join loc_nm_ctg c
    on a.end_loc = c.loc_nm
```

```
    group by b.loc_ctg , a.start_loc,c.loc_ctg, a.end_loc
    order by b.loc_ctg)t
where start_ctg in ('酒店', '住宅', '写字楼') and rnk=1
;
```

八、小红书面试真题-用户行为分析

本场景共使用3张表，表结构如下:

gd_inf(商品信息表)		
gd_id	gd_nm	gd_typ
asfasasg1214	耐克 Nike Air Monarch 4 White Navy	潮鞋
asfijpoupoj345	阿迪达斯 女鞋 Adidas AQUA 女子三道杠	潮鞋

xhs_fav_rcd(用户收藏商品表)			
fav_trq	cust_uid	mch_id	fav_tm
1	12314	asfasasg1214	2021-06-23 12:00:23
2	12345	asfasasg1214	2021-06-23 12:00:26

xhs_pchs_rcd(用户订单表)			
pchs_trq	cust_uid	mch_id	pchs_tm
1	12314	asfasasg1214	2021-06-23 12:00:23
2	12345	asfasasg1214	2021-06-23 12:00:26

1、被收藏次数最多的商品为？

```
select
    i.gd_nm,
    count(1) 'cnt'
from xhs_fav_rcd r
join gd_inf i on r.mch_id = i.gd_id
group by i.gd_nm
order by cnt desc
;
```

2、购买人数最多的商品类目为？

```
select
    i.gd_typ,
    count(distinct r.cust_uid) 'cnt'
from xhs_pchs_rcd r
join gd_inf i on r.mch_id = i.gd_id
group by i.gd_typ
order by cnt desc
```

;

3、以下哪个商品只被收藏，却未被购买？

```
select
    i.gd_nm
from  (select distinct mch_id  from xhs_fav_rcd )f
left join (select distinct mch_id from xhs_pchs_rcd )p
on f.mch_id = p.mch_id
join gd_inf i on f.mch_id = i.gd_id
where p.mch_id is null
;
```

4、以下哪个商品只被购买，却从未被收藏？

```
select
    i.gd_nm
from  (select distinct mch_id  from xhs_fav_rcd )f
right join (select distinct mch_id from xhs_pchs_rcd )p
on f.mch_id = p.mch_id
```

```
join gd_inf i on p.mch_id = i.gd_id
where f.mch_id is null
;
```

5、以下哪个商品，既被同一个用户购买，又被同一个用户收藏，且购买人数最多？

```
select
    i.gd_nm,
    count(1) 'cnt'
from xhs_fav_rcd f
join xhs_pchs_rcd p on f.cust_uid =p.cust_uid  and f.mch_id=p.mch_id
join gd_inf i on f.mch_id = i.gd_id
group by i.gd_nm
order by cnt desc
```

九、快手直播-直播间观看人数峰值分析

本题由快手实习生贡献，是一线业务的实战真题！

本场景共使用2张表，ks_live_t1, ks_live_t2

ks_live_t1			
usr_id	live_id	enter_time	leave_time
KS1000	KSL100034	2021-09-12 12:00:23	2021-09-12 12:00:49

ks_live_t2		
live_id	live_nm	live_type
KS100000	广东靓仔峰少	购物
KS100003	文刀大美人	娱乐

1、进入直播间的高峰期为？（以进入用户数衡量）

```
select hour(enter_time), count(distinct usr_id)
from ks_live_t1
```

```
group by hour(enter_time)
order by 2 desc
;
```

2、晚上 11 点，哪个直播间的进入人数最多？

```
select b.live_nm, usr_cnt
from
    (select live_id, count(distinct usr_id) usr_cnt
    from ks_live_t1
    where hour(enter_time)=23
    group by live_id)a
inner join
    ks_live_t2 b
on a.live_id = b.live_id
order by usr_cnt desc
;
```

3、20：00-23：00，娱乐类、搞笑类，进入人数最多直播间分别是？

```
select * from
(
select live_type, live_nm, dense_rank()over(partition by live_type order by cst_cnt desc) rnk
from
(select live_type, live_nm, count(distinct usr_id) cst_cnt
from
    (
    select a.*, b.live_nm, b.live_type
    from ks_live_t1 a
    inner join
    ks_live_t2 b
    on a.live_id = b.live_id)t
where hour(enter_time) in (20, 21, 22)
group by live_type, live_nm)t
)t
where rnk=1
```

;

4、娱乐类、搞笑类，人均在线时长（退出时间-进入时间）最长的直播间分别是？

```
select * from
(
select *,dense_rank()over(partition by live_type order by retain_time desc) rnk
from
(select live_type, live_nm, avg(retain_time) retain_time
from
    (
    select live_type, live_nm, usr_id, (leave_time - enter_time) as retain_time
    from ks_live_t1 a
    inner join
    ks_live_t2 b
    on a.live_id = b.live_id)t
group by live_type, live_nm)t
)t
where rnk=1
```

;

5、关于同时在线人数，以下哪个说法错误？

```
select b.live_nm, a.*
from
(select live_id, max(num) max_num
from(
select live_id,tms, sum(tag)over(partition by live_id order by tms) as num
from(
select usr_id, live_id,enter_time tms, 1 as tag from ks_live_t1
union all
select usr_id, live_id,leave_time tms, -1 as tag from ks_live_t1)t
)t
group by live_id)a
inner join ks_live_t2 b
on a.live_id = b.live_id
;
```

十、哔哩哔哩面试真题-大会员收入均摊折算

本场景共使用2张表，bilibili_m1, bilibili_m2，对应下图题干中的table_A 和table_B

编程问答题 | 10.0分

1、 请根据题目要求写出完整的SQL代码：
目前有一张全量的用户购买大会员的明细表，需要将每笔大会员的收入摊销，即按用户购买的时间均匀的记到每一天中（例如用户购买了一个15元的7月26日-8月25日的月度大会员，则在7月26日-8月25日期间，每天计入15/31≈0.48元的收入），现在想要统计2021年1月至6月每个月的大会员摊销收入。
Table A，字段和数据样例

user_id（用户id）	begin_date（大会员生效开始日期）	end_date（大会员生效结束日期）	days（生效持续天数）	pay_amount（支付金额）
123	2020-12-01	2021-12-01	366	148
124	2021-07-26	2021-08-25	31	15
...	...	...	...	...

可能会用到的表 Table B（日期维表），字段和数据样例

log_date（日期）	month（日期对应月份）	year（日期对应年份）
2001-01-01	2001-01	2001
2001-01-02	2001-01	2001
...	...	...

```
select y_m, sum(avg_day_amt) from
(
select *, pay_amount/datediff(end_date, begin_date) as avg_day_amt
from (select * from bilibili_m2 where m_date between '2021-01-01' and '2021-05-31' ) a
left join
(select * from bilibili_m1) b
on 1 -无实意，本质是笛卡尔连接
where user_id like '%1014%' -只是为了验证，实际查数时需要去掉
    and m_date>= begin_date and m_date <=end_date
  )t
group by y_m
```

十一、连续登录专题

1、美团连续登录

```
select count(distinct usr_id)
from
(
```

```
select usr_id, load_dt2, count(1) load_days
from
    (
        select usr_id, load_dt, rnk, date_sub(load_dt, interval rnk day) as load_dt2
        from
        (
            select
                    a.usr_id
                , a.load_dt
                , row_number()
                over(partition by a.usr_id order by a.load_dt) rnk
                from
                    (
                        select
                            usr_id
                            , load_date load_dt
                        from mt_t1
```

```
                    )a
            )b
        )c
    group by usr_id,load_dt2
    having load_days >= 2 --大于等于2
)t
;
```

2、小鹏汽车连续快充

```
select usr_id,max(times)
from
(
select usr_id, rnk3, count(1) times
from
(select
  *
  , row_number() over(partition by usr_id order by charge_time) rnk1
  , sum(charge_type) over(partition by usr_id order by charge_time) rnk2
```

```
  ,row_number() over(partition by usr_id order by charge_time)- sum(charge_type) over(partition by usr_id order by
charge_time) rnk3
from xp_t1)t
group by usr_id, rnk3)t
group by usr_id
having max(times)>=13
```

3、微保连续点击

```
select distinct usr_id
from
( select *, rank_1- rank_2  as diff
   from
  (select *,
      row_number() over(order by click_time) as  rank_1,
      row_number() over(partition by usr_id order by click_time) as rank_2
      from wb_t1
   ) b
) c
```

```
group by diff,usr_id
having count(diff) >=2
```