【牛客网】SQL大厂真题答案

一、某音短视频

1.各个视频的平均完播率

原题链接：https://www.nowcoder.com/practice/96263162f69a48df9d84a93c71045753

```sql
-- 基本思路
select
video_id
,round(sum(comp) / count(comp), 3) avg_comp_play_rate
from
(
    select
    info.video_id video_id
    ,if(timestampdiff(second, start_time, end_time) >= duration, 1, 0) comp
    from tb_user_video_log log
    left join tb_video_info info
    on log.video_id = info.video_id
    where year(start_time) = '2021'
) a
group by 1
order by 2 desc;

-- 优化思路
select
log.video_id video_id
,round(avg(if(timestampdiff(second, start_time, end_time) >= duration, 1, 0)), 3) avg_comp_play_rate
from tb_user_video_log log
left join tb_video_info info
on log.video_id = info.video_id
where left(start_time, 4) = '2021'
group by 1
order by 2 desc;
```

2.平均播放进度大于60%的视频类别

原题链接：https://www.nowcoder.com/practice/c60242566ad94bc29959de0cdc6d95ef

```sql
-- 子查询思路
select
tag
,concat(round(pro * 100, 2), '%') as avg_play_progress
from
(
    select
    tag
    -- 不要在这里套round，会影响你的筛选
    ,avg(if(timestampdiff(second, start_time, end_time) > duration, 1, timestampdiff(second, start_time, end_time) / duration)) pro
    from tb_user_video_log log
    left join tb_video_info info
    on log.video_id = info.video_id
    group by 1
)a
where pro > 0.6
order by 2 desc;

-- 优化思路
select
tag
,concat(round(avg(if(timestampdiff(second, start_time, end_time) > duration, 1, timestampdiff(second,
start_time, end_time) / duration)) * 100, 2), '%') pro
from tb_user_video_log log
left join tb_video_info info
on log.video_id = info.video_id
group by 1
having avg(if(timestampdiff(second, start_time, end_time) > duration, 1, timestampdiff(second,
start_time, end_time) / duration)) > 0.6
order by 2 desc;
```

3.每类视频近一个月的转发量/率

原题链接：https://www.nowcoder.com/practice/a78cf92c11e0421abf93762d25c3bfad

```sql
select
b.tag
,sum(if_retweet) retweet_cnt
,ROUND(sum(if_retweet) / count(*), 3) retweet_rate
from tb_user_video_log log
left join tb_video_info info
onlog.video_id = info.video_id
where datediff((select max(start_time) from tb_user_video_log), date(a.start_time)) <= 29
group by 1
order by 3 desc;
```

4.每个创作者每月的涨粉率及截止当前的总粉丝量

原题链接：https://www.nowcoder.com/practice/d337c95650f640cca29c85201aecff84

```sql
-- 套两层子查询
select
author
,month
,round(fans_growth_rate, 3) fans_growth_rate
,sum(inc_fo)over(partition by author order by month) total_fans
from
(
    select
    author
    ,left(start_time, 7) month
    ,sum(inc_fo) / count(1) as fans_growth_rate
    ,sum(inc_fo) inc_fo
    from
    (
        select
        log.video_id video_id
        ,start_time
        ,case when if_follow = 1 then 1
            when if_follow = 2 then -1
        else 0 end inc_fo
        ,author
        from tb_user_video_log log
        left join tb_video_info info
        on log.video_id = info.video_id
    ) a
    group by 1, 2
) b
where left(month, 4) = '2021'
order by 1, 4

-- 窗口函数嵌套聚合函数
select
author
,left(start_time, 7) month
,round(sum(inc_fo) / count(1), 3) as fans_growth_rate
,sum(sum(inc_fo)) over(partition by author order by left(start_time, 7)) total_fans
from
(
    select
    log.video_id video_id
    ,start_time
    ,case when if_follow = 1 then 1
        when if_follow = 2 then -1
    else 0 end inc_fo
    ,author
    from tb_user_video_log log
    left join tb_video_info info
    on log.video_id = info.video_id
) a
where left(start_time, 4) = '2021'
group by 1, 2
order by 1, 4
```

5.国庆期间每类视频点赞量和转发量

原题链接：https://www.nowcoder.com/practice/f90ce4ee521f400db741486209914a11

```sql
select
*
from
(
    select
    tag
    ,dt
    ,sum(like_cnt)over(partition by tag rows between 6 preceding and current row) sum_like_cnt_7d
    ,max(retweet_cnt)over(partition by tag rows between 6 preceding and current row) max_retweet_cnt_7d
    from
    (
        select
        tag
        ,left(start_time, 10) dt
        ,sum(if_like) like_cnt
        ,sum(if_retweet) retweet_cnt
        from tb_user_video_log log
        left join tb_video_info info
        on log.video_id = info.video_id
        where left(start_time, 4) = '2021'
        group by 1, 2
    ) a
) b
```

```sql
where dt in ('2021-10-01', '2021-10-02', '2021-10-03')
order by 1 desc, 2;
```

6.近一个月发布的视频中热度最高的top3视频

原题链接：https://www.nowcoder.com/practice/0226c7b2541c41e59c3b8aec588b09ff

```sql
select
video_id
,round((avg(if_comp) * 100 + 5 * sum(if_like) + 3 * sum(is_comment) + 2 * sum(if_retweet))
     * (1 / (1 + min(diff_time))))  hot_index
from (
    select
    log.video_id video_id
    ,if(timestampdiff(second,start_time,end_time) >= duration, 1, 0) as if_comp
    ,if_like
    ,if(comment_id is null, 0 ,1) as is_comment
    ,if_retweet
    ,abs(datediff(end_time, (select max(end_time) from tb_user_video_log))) diff_time
    from tb_user_video_log log
    left join tb_video_info info
    on log.video_id = info.video_id
    where abs(datediff(release_time, (select max(end_time) from tb_user_video_log))) < 30
) a
group by 1
order by 2 desc
limit 3;
```

二、用户增长场景（某度信息流）

1.2021年11月每天的人均浏览文章时长

原题链接：https://www.nowcoder.com/practice/8e33da493a704d3da15432e4a0b61bb3

```sql
-- 思路一：子查询
select
dt
,round(sum(sum_second) / count(distinct uid), 1) avg_viiew_len_sec
from
(
    select
    uid
    ,date(in_time) dt
    ,sum(timestampdiff(second, in_time, out_time)) sum_second
    from tb_user_log
    where artical_id != '0' and artical_id != 0
    group by 1, 2
) t
where dt like '2021-11%'
group by 1
order by 2;

-- 思路二：拆解公式直接算
select
left(in_time, 10) dt,
round(sum(timestampdiff(second, in_time, out_time)) / count(distinct uid), 1) avg_viiew_len_sec
from tb_user_log
```

```sql
where left(in_time,7) = '2021-11'
and artical_id != 0
group by 1
order by 2
```

2.每篇文章同一时刻最大在看人数

```sql
with log
as(
    select
    uid
    ,artical_id
    ,in_time as dt
    ,1 is_in
    from tb_user_log
    where artical_id != '0' and artical_id != 0

    union
    select
    uid
    ,artical_id
    ,out_time as dt
    ,-1 is_in
    from tb_user_log
    where artical_id != '0' and artical_id != 0
)
select
artical_id
,max(uv) max_uv
from
(
    select
    artical_id
    ,dt
    ,sum(is_in)over(partition by artical_id order by dt, is_in desc) uv
    from log
) t
group by 1
order by 2 desc;
```

3.2021年11月每天新用户的次日留存率

```sql
with reg_user
as(
    select
    uid
    ,date(min(in_time)) first_date
    from tb_user_log
    group by 1
), -- 用户注册表

user_log
as(
    select
    uid
    ,date(in_time) dt
    from tb_user_log
    union
    select
    uid
    ,date(out_time) dt
    from tb_user_log
) -- 用户活跃表

select
reg_user.first_date dt
,round(count(user_log.dt) / count(reg_user.first_date), 2) uv_left_rate
from reg_user
left join user_log
on reg_user.uid = user_log.uid
and reg_user.first_date = date_sub(user_log.dt, interval 1 day)
where reg_user.first_date like '%2021-11%'
group by 1
order by 1
```

4.统计活跃间隔对用户分级结果

```sql
with user_tb
as(
    select
    uid
    ,date(min(in_time)) first_time
    ,date(max(out_time)) last_time
    ,(select date(max(out_time)) from tb_user_log) today
    from tb_user_log
    group by 1
)
select
grade
,round(count(distinct uid) / (select count(1) from user_tb), 2) ratio
from
(
    select
    uid
    ,case when datediff(today, first_time) <= 6 then '新晋用户'
          when datediff(today, first_time) > 6 and datediff(today, last_time) <= 6 then '忠实用户'
          when datediff(today, first_time) > 6 and datediff(today, last_time) > 29 then '流失用户'
          when datediff(today, first_time) > 6 and datediff(today, last_time) > 6 then '沉睡用户'
    else '其他' end grade
    from user_tb
) t1
group by 1
order by 2 desc
```

5.每天的日活数及新用户占比

```sql
with user_reg
as(
    select
    uid
    ,date(min(in_time)) dt
    from tb_user_log
    group by 1
), -- 用户注册表
```

```

user_log
as(
    select
    uid
    ,date(in_time) dt
    from tb_user_log

    union

    select
    uid
    ,date(out_time) dt
    from tb_user_log
) -- 用户活跃表

select
log.dt dt
,count(distinct log.uid) dau
,round(count(distinct reg.uid) / count(distinct log.uid), 2) uv_new_ratio
from user_log log
left join user_reg reg
on log.dt = reg.dt
and log.dt = reg.dt
group by 1
order by 1
```

6.连续签到领金币

```
with t1
as(
    select distinct
    uid
    ,date(in_time) dt
    from tb_user_log
    where artical_id = 0
    and date(in_time) between '2021-07-07' and '2021-10-31'
    and sign_in = 1
),
t2 as
(
    select
    uid
    ,dt
    ,row_number()over(partition by uid order by dt) rk
    from t1
)

select
uid
,date_format(dt, '%Y%m') month
,sum(coin) coin
from
(
    select
    uid
    ,dt
    ,case when con_sign = 3 then 3
          when con_sign = 0 then 7
```

```
         else 1 end coin
        from
        (
            select
            uid
            ,dt
            ,date_sub(dt, interval rk day) dt_tmp
            ,row_number()over(partition by uid, date_sub(dt, interval rk day) order by dt) % 7 con_sign
            from t2
        ) a
) b
group by 1, 2
order by 2, 1;
```

三、电商场景（某东商城）

1.计算商城中2021年每月的GMV

原题链接：https://www.nowcoder.com/practice/5005cbf5308249eda1fbf666311753bf

```
select
left(event_time, 7) month
,round(sum(total_amount)) GMV
from tb_order_overall
where status in (0, 1)
and year(event_time) = 2021
group by 1
having sum(total_amount) > 100000
order by 2
```

2.统计2021年10月每个退货率不大于0.5的商品各项指标

原题链接：https://www.nowcoder.com/practice/cbf582d28b794722becfc680847327be

```
select
product_id
,round(ifnull(sum(if_click) / count(1), 0), 3) ctr
,round(ifnull(sum(if_cart) / sum(if_click), 0), 3) cart_rate
,round(ifnull(sum(if_payment) / sum(if_cart), 0), 3) payment_rate
,round(ifnull(sum(if_refund) / sum(if_payment), 0), 3) refund_rate
from tb_user_event
where left(event_time, 7) = '2021-10'
group by 1
having sum(if_refund) / sum(if_payment) <= 0.5
order by 1
```

3.某店铺的各商品毛利率及店铺整体毛利率

原题链接：https://www.nowcoder.com/practice/65de67f666414c0e8f9a34c08d4a8ba6

```
select
product_id
,concat(round(profit_rate * 100, 1), '%') profit_rate
from
(
```

```sql
    select
    tb_product_info.product_id as product_id
    ,(1- sum(in_price) / (sum(price))) as profit_rate
    from tb_order_detail
    left join tb_order_overall
    on tb_order_detail.order_id = tb_order_overall.order_id
    left join tb_product_info
    on tb_order_detail.product_id = tb_product_info.product_id
    where left(event_time, 7) >= '2021-10'
    and shop_id = '901'
    and status != 2 and status != '2'
    group by 1
    having (1- sum(in_price) / sum(price)) > 0.249

    union all

    select
    '店铺汇总' as product_id
    ,(1- sum(in_price * cnt) / sum(price * cnt)) as profit_rate
    from tb_order_detail
    left join tb_order_overall
    on tb_order_detail.order_id = tb_order_overall.order_id
    left join tb_product_info
    on tb_order_detail.product_id = tb_product_info.product_id
    where left(event_time, 7) >= '2021-10'
    and shop_id = '901'
    and status != 2 and status != '2'
    group by 1

) t
order by product_id in ('8001', '8002', '8003')
```

4.零食类商品中复购率top3高的商品

```sql
with tmp
as (
    select
    tb_order_overall.uid uid
    ,tb_order_detail.product_id product_id
    ,count(1) cnt
    from tb_order_detail
    left join tb_product_info
    on tb_order_detail.product_id = tb_product_info.product_id
    left join tb_order_overall
    on tb_order_detail.order_id = tb_order_overall.order_id
    where tag = '零食'
    and abs(datediff(cast(event_time as date), (select max(event_time) from tb_order_overall))) <= 89
    and status = 1
    group by 1, 2
)

select
product_id
,round(sum(if(cnt >= 2, 1, 0)) / count(uid), 3)
from tmp
group by 1
order by 2 desc, 1
limit 3
```

5.10月的新户客单价和获客成本

```sql
with reg
as(
    select
    uid
    ,min(event_time) first_time
    from tb_order_overall
    group by 1
)

select round(avg(total_amount) ,1) avg_amount,
       round(avg(cost), 1) avg_cost
from
(
    select
    reg.uid as uid
    ,avg(total_amount) as total_amount
    ,(sum(price * cnt) - avg(total_amount)) as cost
    from tb_order_detail a
    left join tb_order_overall b
    on a.order_id = b.order_id
    left join reg
    on b.uid = reg.uid
    and b.event_time = reg.first_time
    where left(event_time, 7) = '2021-10'
    and reg.first_time is not null
    group by 1
) t
```

6.店铺901国庆期间的7日动销率和滞销率

```sql
with tb
as (
    select
    tpi.product_id
    ,date(tpi.release_time) as release_dt
    ,date(too.event_time) as sale_dt
    ,shop_id
    from tb_product_info tpi
    left join tb_order_detail tod
    on tod.product_id = tpi.product_id
    left join tb_order_overall too
    on tod.order_id = too.order_id
    where too.status = 1
    and tpi.release_time <= '2021-10-03'
    and date(too.event_time) between '2021-09-25' and '2021-10-03'
),
dt_tb
as(
    select date(event_time) as dt
    from tb_order_overall
    where date(event_time) between '2021-10-01' and '2021-10-03'
    group by 1
)

select
dt
```

```sql
,round(sum(is_sale) / sum(is_shelf), 3) as sale_rate
,round(1 - sum(is_sale) / sum(is_shelf), 3) as unsale_rate
from
(
    select
    dt
    ,product_id
    ,max(if(datediff(dt, sale_dt) between 0 and 6, 1, 0)) as is_sale
    ,max(if(release_dt <= dt, 1, 0)) as is_shelf
    from tb as t1
    join dt_tb as t2
    where shop_id = 901
    group by 1, 2
) as t
group by 1
order by 1;
```

四、出行场景（某滴打车）

1. 2021年国庆在北京接单3次及以上的司机统计信息

```sql
-- 场景说明：
-- 用户申请打车 --->司机接单 ---> 上车 ---> 到达目的地 ---> 订单完成
--             ---->取消打车
--                          --> 取消打车（司机、乘客）

-- 用户申请打车 记录表生成记录，order_id set null
-- 司机接单 订单表生成记录，开始时间、完成时间、里程、费用全部为空，同时填充记录表的申请时间和完成时间
-- 取消打车

select
city
,round(avg(cnt), 3) avg_order_num
,round(avg(tol_fare), 3) avg_income
from
(
    select
    city
    ,driver_id
    ,count(1) cnt
    ,sum(fare) tol_fare
    from tb_get_car_order
    left join tb_get_car_record
    on tb_get_car_order.order_id = tb_get_car_record.order_id
    where city = '北京'
    and date(tb_get_car_order.order_time) between '2021-10-01' and '2021-10-07'
    group by 1, 2
    having count(1) >=3
) t
group by 1
```

2.有取消订单记录的司机平均评分

```sql
with driver_cancel_tb
as(
    select
    driver_id
    from tb_get_car_order
    where start_time is null
    and left(order_time, 7) = '2021-10'
    group by 1
)

select
driver_id
,avg_grade
from
(
    select
    driver_id
    ,round(avg(grade), 1) avg_grade
    from tb_get_car_order
    where driver_id in (select driver_id from driver_cancel_tb)
    group by 1

    union all
    select
    '总体'
    ,round(avg(grade), 1) avg_grade
    from tb_get_car_order
    where driver_id in (select driver_id from driver_cancel_tb)
    group by 1
) t
order by 1
```

3.每个城市中评分最高的司机信息

```sql
with driver_info
as (
    select
    city
    ,driver_id
    ,avg(grade) avg_grade
    ,count(1) / count(distinct left(order_time, 10)) avg_order_num
    ,sum(mileage) / count(distinct left(order_time, 10)) avg_mileage
    from tb_get_car_order tbco
    left join tb_get_car_record tgcr
    on tbco.order_id = tgcr.order_id
    group by city, driver_id
)

select
city
,driver_id
,round(avg_grade, 1) avg_grade
,round(avg_order_num, 1) avg_order_num
,round(avg_mileage, 3) avg_mileage
from
(
    select
    *
    ,dense_rank()over(partition by city order by avg_grade desc) rk
    from driver_info
) a
where rk = 1
```

```
order by 3
```

4. 国庆期间近7日日均取消订单量

```
select
*
from
(
    select
    dt
    ,round(sum(finish_cnt) over(order by dt rows 6 preceding) /7,2) finish_num_7d
    ,round(sum(cancel_cnt) over(order by dt rows 6 preceding) /7,2) cancel_num_7d
    from
    (
        select
        date(event_time) as dt
        ,sum(if(mileage is not null, 1, 0)) finish_cnt
        ,sum(if(mileage is null, 1, 0)) cancel_cnt
        from tb_get_car_record tbgc
        inner join tb_get_car_order tgco
        on tbgc.order_id = tgco.order_id
        where date(event_time) between '2021-09-25' AND '2021-10-03'
        group by 1
    ) a
) b
where dt between '2021-10-01' AND '2021-10-03'
```

5.工作日各时段叫车量、等待接单时间和调度时间

```
with tmp
as(
    select
    case when hour(event_time) >= 9 and hour(event_time) < 17 then '工作时间'
         when hour(event_time) >= 17 and hour(event_time) < 20 then '晚高峰'
         when hour(event_time) >= 7 and hour(event_time) < 9 then '早高峰'
    else '休息时间' end period
    ,count(tgcr.id) get_car_num
    ,avg(timestampdiff(second, event_time, end_time)) avg_wait_time_second
    ,avg(timestampdiff(second, order_time, start_time)) avg_dispatch_time_second
    from tb_get_car_record tgcr
    left join
    (
        select * from tb_get_car_order tgco
        where mileage is not null
    ) a
    on tgcr.order_id = a.order_id
    where dayofweek(event_time) - 1 between 1 and 5
    group by 1
)

select
*
from
(
    select
    period
    ,get_car_num
    ,round(avg_wait_time_second / 60, 1) avg_wait_time
    ,round(avg_dispatch_time_second / 60, 1) avg_dispatch_time
    from tmp
    where period = '工作时间'

    union all

    select
    period
    ,get_car_num
    ,round(avg_wait_time_second / 60, 1) avg_wait_time
    ,round(avg_dispatch_time_second / 60, 1) avg_dispatch_time
    from tmp
    where period = '休息时间'

    union all

    select
    period
    ,get_car_num
    ,round(avg_wait_time_second / 60, 1) avg_wait_time
    ,round(avg_dispatch_time_second / 60, 1) avg_dispatch_time
    from tmp
    where period = '晚高峰'

    union all

    select
    period
    ,get_car_num
    ,round(avg_wait_time_second / 60, 1) avg_wait_time
    ,round(avg_dispatch_time_second / 60, 1) avg_dispatch_time
    from tmp
    where period = '早高峰'
) b
order by 2
```

6.各城市最大同时等车人数

```
-- 打车 event_time 1
-- 打车取消 order_id is null -1
-- 等车取消 start_time is null -1
-- 上车 -1

with tmp
as(
    select
    city
    ,event_time as uv_time
    ,1 uv
    from tb_get_car_record

    union all
    select
    city
    ,end_time as uv_time
    ,-1 uv
    from tb_get_car_record
    where order_id is null
```

```sql

    union all
    select
    city
    ,ifnull(start_time, finish_time) as uv_time
    ,-1 uv
    from tb_get_car_order tgco
    left join tb_get_car_record tgcr
    on tgco.order_id = tgcr.order_id
)
select
city
,max(uv_cnt)
from
(
    select
    city
    ,sum(uv)over(partition by city order by uv_time, uv desc) uv_cnt
    from tmp
    where left(uv_time, 7) = '2021-10'
) a
group by 1
order by 2, 1
```

五、某宝店铺分析（电商模式）

出的啥题，比前面4个板块都要简单的多，适合初学者练习。

1.某宝店铺的SPU数量

```sql
select
style_id
,count(1) SUP_num
from product_tb
group by 1
order by 2 desc
```

2.某宝店铺的实际销售额与客单价

```sql
select
sum(sales_price)
,round(sum(sales_price) / count(distinct user_id), 2)
from sales_tb
```

3.某宝店铺折扣率

```sql
select
round(sum(sales_price) / sum(tag_price * sales_num) * 100, 2) as `discount_rate(%)`
from sales_tb
left join product_tb
on sales_tb.item_id = product_tb.item_id
```

4.某宝店铺动销率与售罄率

```sql
SELECT
t1.style_id
,round(sales_cnt / (inventory_cnt - sales_cnt) * 100 ,2) `pin_rate(%)`
,round(gmv / cost * 100 ,2) `sell-through_rate(%)`
from
(
    SELECT
    style_id
    ,sum(inventory) inventory_cnt
    ,sum(tag_price * inventory) cost
    from product_tb
    group by style_id
) t1
LEFT JOIN
(
    SELECT
    style_id
    ,sum(sales_num) sales_cnt
    ,sum(sales_price) gmv
    from sales_tb
    left join product_tb
    on sales_tb.item_id = product_tb.item_id
    group by style_id
)t2
on t1.style_id = t2.style_id
order by style_id;
```

5.某宝店铺连续2天及以上购物的用户及其对应的天数

```sql
with tmp
as(
    select
    *
    ,row_number()over(order by dt) rk
    from
    (
        select
        user_id
        ,sales_date dt
        from sales_tb
        group by 1, 2
    ) t1
)

select
user_id
,count(1) days_count
from
(
    select
    dt
    ,user_id
    ,date_sub(dt, interval 1 day) tmp_dt
    from tmp
) a
```

```
group by 1
having count(1) >= 2
```

六、牛客直播课分析（在线教育行业）

1.牛客直播转换率

```
select
btb.course_id course_id
,course_name
,round(sum(if_sign) / sum(if_vw) * 100, 2) `sign_rate(%)`
from behavior_tb btb
left join course_tb ctb
on btb.course_id = ctb.course_id
group by 1, 2
order by 1
```

2.牛客直播开始时各直播间在线人数

```

```