

【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>

```
1  -- 基本思路
2  select
3  video_id
4  ,round(sum(comp) / count(comp), 3) avg_comp_play_rate
5  from
6  (
7      select
8      info.video_id video_id
9      ,if(timestampdiff(second, start_time, end_time) >= duration, 1, 0) comp
10     from tb_user_video_log log
11     left join tb_video_info info
12     on log.video_id = info.video_id
13     where year(start_time) = '2021'
14 ) a
15 group by 1
16 order by 2 desc;
17
18 -- 优化思路
19 select
20 log.video_id video_id
21 ,round(avg(if(timestampdiff(second, start_time, end_time) >= duration, 1, 0)), 3) avg_comp_play_rate
22 from tb_user_video_log log
23 left join tb_video_info info
24 on log.video_id = info.video_id
25 where left(start_time, 4) = '2021'
26 group by 1
27 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>

```
1  -- 子查询思路
2  select
3  tag
4  ,concat(round(pro * 100, 2), '%') as avg_play_progress
5  from
6  (
7      select
8      tag
9      -- 不要在这里套round, 会影响你的筛选
10     ,avg(if(timestampdiff(second, start_time, end_time) > duration, 1, timestampdiff(second, start_time, end_time) / duration)) pro
11     from tb_user_video_log log
12     left join tb_video_info info
13     on log.video_id = info.video_id
```

```

14     group by 1
15 )a
16 where pro > 0.6
17 order by 2 desc;
18
19 -- 优化思路
20 select
21 tag
22 ,concat(round(avg(if(timestampdiff(second, start_time, end_time) > duration, 1, timestampdiff(second,
    start_time, end_time) / duration))) * 100, 2), '%') pro
23 from tb_user_video_log log
24 left join tb_video_info info
25 on log.video_id = info.video_id
26 group by 1
27 having avg(if(timestampdiff(second, start_time, end_time) > duration, 1, timestampdiff(second,
    start_time, end_time) / duration)) > 0.6
28 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>

```

1 select
2 b.tag
3 ,sum(if_retweet) retweet_cnt
4 ,ROUND(sum(if_retweet) / count(*), 3) retweet_rate
5 from tb_user_video_log log
6 left join tb_video_info info
7 on log.video_id = info.video_id
8 where datediff((select max(start_time) from tb_user_video_log), date(a.start_time)) <= 29
9 group by 1
10 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>

```

1 -- 套两层子查询
2 select
3 author
4 ,month
5 ,round(fans_growth_rate, 3) fans_growth_rate
6 ,sum(inc_fo)over(partition by author order by month) total_fans
7 from
8 (
9     select
10     author
11     ,left(start_time, 7) month
12     ,sum(inc_fo) / count(1) as fans_growth_rate
13     ,sum(inc_fo) inc_fo
14     from
15     (
16         select
17         log.video_id video_id
18         ,start_time
19         ,case when if_follow = 1 then 1
20             when if_follow = 2 then -1
21             else 0 end inc_fo

```

```

22         ,author
23         from tb_user_video_log log
24         left join tb_video_info info
25         on log.video_id = info.video_id
26     ) a
27     group by 1, 2
28 ) b
29 where left(month, 4) = '2021'
30 order by 1, 4
31
32 -- 窗口函数嵌套聚合函数
33 select
34 author
35 ,left(start_time, 7) month
36 ,round(sum(inc_fo) / count(1), 3) as fans_growth_rate
37 ,sum(sum(inc_fo)) over(partition by author order by left(start_time, 7)) total_fans
38 from
39 (
40     select
41         log.video_id video_id
42         ,start_time
43         ,case when if_follow = 1 then 1
44             when if_follow = 2 then -1
45         else 0 end inc_fo
46         ,author
47         from tb_user_video_log log
48         left join tb_video_info info
49         on log.video_id = info.video_id
50     ) a
51 where left(start_time, 4) = '2021'
52 group by 1, 2
53 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>

```

1 select
2 *
3 from
4 (
5     select
6         tag
7         ,dt
8         ,sum(like_cnt)over(partition by tag rows between 6 preceding and current row) sum_like_cnt_7d
9         ,max(retweet_cnt)over(partition by tag rows between 6 preceding and current row) max_retweet_cnt_7d
10    from
11    (
12        select
13            tag
14            ,left(start_time, 10) dt
15            ,sum(if_like) like_cnt
16            ,sum(if_retweet) retweet_cnt
17        from tb_user_video_log log
18        left join tb_video_info info
19        on log.video_id = info.video_id
20        where left(start_time, 4) = '2021'
21        group by 1, 2
22    ) a
23 ) b

```

```
24 where dt in ('2021-10-01', '2021-10-02', '2021-10-03')
25 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>

```
1 select
2 video_id
3 ,round((avg(if_comp) * 100 + 5 * sum(if_like) + 3 * sum(is_comment) + 2 * sum(if_retweet))
4         * (1 / (1 + min(diff_time)))) hot_index
5 from (
6     select
7         log.video_id video_id
8         ,if(timestampdiff(second,start_time,end_time) >= duration, 1, 0) as if_comp
9         ,if_like
10        ,if(comment_id is null, 0 ,1) as is_comment
11        ,if_retweet
12        ,abs(datediff(end_time, (select max(end_time) from tb_user_video_log))) diff_time
13    from tb_user_video_log log
14    left join tb_video_info info
15    on log.video_id = info.video_id
16    where abs(datediff(release_time, (select max(end_time) from tb_user_video_log))) < 30
17 ) a
18 group by 1
19 order by 2 desc
20 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>

```
1 -- 思路一：子查询
2 select
3 dt
4 ,round(sum(sum_second) / count(distinct uid), 1) avg_vview_len_sec
5 from
6 (
7     select
8         uid
9         ,date(in_time) dt
10        ,sum(timestampdiff(second, in_time, out_time)) sum_second
11    from tb_user_log
12    where artical_id != '0' and artical_id != 0
13    group by 1, 2
14 ) t
15 where dt like '2021-11%'
16 group by 1
17 order by 2;
18
19 -- 思路二：拆解公式直接算
20 select
21 left(in_time, 10) dt,
22 round(sum(timestampdiff(second, in_time, out_time)) / count(distinct uid), 1) avg_vview_len_sec
23 from tb_user_log
```

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

2.每篇文章同一时刻最大在看人数

```
1 with log
2 as(
3     select
4         uid
5         ,artical_id
6         ,in_time as dt
7         ,1 is_in
8     from tb_user_log
9     where artical_id != '0' and artical_id != 0
10
11     union
12     select
13         uid
14         ,artical_id
15         ,out_time as dt
16         ,-1 is_in
17     from tb_user_log
18     where artical_id != '0' and artical_id != 0
19 )
20 select
21     artical_id
22     ,max(uv) max_uv
23 from
24 (
25     select
26         artical_id
27         ,dt
28         ,sum(is_in)over(partition by artical_id order by dt, is_in desc) uv
29     from log
30 ) t
31 group by 1
32 order by 2 desc;
```

3.2021年11月每天新用户的次日留存率

```
1 with reg_user
2 as(
3     select
4         uid
5         ,date(min(in_time)) first_date
6     from tb_user_log
7     group by 1
8 ), -- 用户注册表
9
10 user_log
11 as(
12     select
13         uid
14         ,date(in_time) dt
15     from tb_user_log
```

```

16     union
17     select
18     uid
19     ,date(out_time) dt
20     from tb_user_log
21 ) -- 用户活跃表
22
23 select
24 reg_user.first_date dt
25 ,round(count(user_log.dt) / count(reg_user.first_date), 2) uv_left_rate
26 from reg_user
27 left join user_log
28 on reg_user.uid = user_log.uid
29 and reg_user.first_date = date_sub(user_log.dt, interval 1 day)
30 where reg_user.first_date like '%2021-11%'
31 group by 1
32 order by 1

```

4.统计活跃间隔对用户分级结果

```

1  with user_tb
2  as(
3      select
4      uid
5      ,date(min(in_time)) first_time
6      ,date(max(out_time)) last_time
7      ,(select date(max(out_time)) from tb_user_log) today
8      from tb_user_log
9      group by 1
10 )
11 select
12 grade
13 ,round(count(distinct uid) / (select count(1) from user_tb), 2) ratio
14 from
15 (
16     select
17     uid
18     ,case when datediff(today, first_time) <= 6 then '新晋用户'
19           when datediff(today, first_time) > 6 and datediff(today, last_time) <= 6 then '忠实用户'
20           when datediff(today, first_time) > 6 and datediff(today, last_time) > 29 then '流失用户'
21           when datediff(today, first_time) > 6 and datediff(today, last_time) > 6 then '沉睡用户'
22     else '其他' end grade
23     from user_tb
24 ) t1
25 group by 1
26 order by 2 desc

```

5.每天的日活数及新用户占比

```

1  with user_reg
2  as(
3      select
4      uid
5      ,date(min(in_time)) dt
6      from tb_user_log
7      group by 1
8  ), -- 用户注册表

```

```

9
10 user_log
11 as(
12     select
13         uid
14         ,date(in_time) dt
15     from tb_user_log
16
17     union
18
19     select
20         uid
21         ,date(out_time) dt
22     from tb_user_log
23 ) -- 用户活跃表
24
25 select
26 log.dt dt
27 ,count(distinct log.uid) dau
28 ,round(count(distinct reg.uid) / count(distinct log.uid), 2) uv_new_ratio
29 from user_log log
30 left join user_reg reg
31 on log.dt = reg.dt
32 and log.dt = reg.dt
33 group by 1
34 order by 1

```

6.连续签到领金币

```

1 with t1
2 as(
3     select distinct
4         uid
5         ,date(in_time) dt
6     from tb_user_log
7     where artical_id = 0
8     and date(in_time) between '2021-07-07' and '2021-10-31'
9     and sign_in = 1
10 ),
11 t2 as
12 (
13     select
14         uid
15         ,dt
16         ,row_number()over(partition by uid order by dt) rk
17     from t1
18 )
19
20 select
21 uid
22 ,date_format(dt, '%Y%m') month
23 ,sum(coin) coin
24 from
25 (
26     select
27         uid
28         ,dt
29         ,case when con_sign = 3 then 3
30             when con_sign = 0 then 7

```

```

31     else 1 end coin
32 from
33 (
34     select
35     uid
36     ,dt
37     ,date_sub(dt, interval rk day) dt_tmp
38     ,row_number()over(partition by uid, date_sub(dt, interval rk day) order by dt) % 7 con_sign
39     from t2
40 ) a
41 ) b
42 group by 1, 2
43 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>

```

1 select
2 left(event_time, 7) month
3 ,round(sum(total_amount)) GMV
4 from tb_order_overall
5 where status in (0, 1)
6 and year(event_time) = 2021
7 group by 1
8 having sum(total_amount) > 1000000
9 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>

```

1 select
2 product_id
3 ,round(ifnull(sum(if_click) / count(1), 0), 3) ctr
4 ,round(ifnull(sum(if_cart) / sum(if_click), 0), 3) cart_rate
5 ,round(ifnull(sum(if_payment) / sum(if_cart), 0), 3) payment_rate
6 ,round(ifnull(sum(if_refund) / sum(if_payment), 0), 3) refund_rate
7 from tb_user_event
8 where left(event_time, 7) = '2021-10'
9 group by 1
10 having sum(if_refund) / sum(if_payment) <= 0.5
11 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>

```

1 select
2 product_id
3 ,concat(round(profit_rate * 100, 1), '%') profit_rate
4 from
5 (

```



```

6     select
7     tb_product_info.product_id as product_id
8     ,(1- sum(in_price) / (sum(price))) as profit_rate
9     from tb_order_detail
10    left join tb_order_overall
11    on tb_order_detail.order_id = tb_order_overall.order_id
12    left join tb_product_info
13    on tb_order_detail.product_id = tb_product_info.product_id
14    where left(event_time, 7) >= '2021-10'
15    and shop_id = '901'
16    and status != 2 and status != '2'
17    group by 1
18    having (1- sum(in_price) / sum(price)) > 0.249
19
20    union all
21
22    select
23    '店铺汇总' as product_id
24    ,(1- sum(in_price * cnt) / sum(price * cnt)) as profit_rate
25    from tb_order_detail
26    left join tb_order_overall
27    on tb_order_detail.order_id = tb_order_overall.order_id
28    left join tb_product_info
29    on tb_order_detail.product_id = tb_product_info.product_id
30    where left(event_time, 7) >= '2021-10'
31    and shop_id = '901'
32    and status != 2 and status != '2'
33    group by 1
34
35 ) t
36 order by product_id in ('8001', '8002', '8003')

```

4.零食类商品中复购率top3高的商品

```

1  with tmp
2  as (
3      select
4      tb_order_overall.uid uid
5      ,tb_order_detail.product_id product_id
6      ,count(1) cnt
7      from tb_order_detail
8      left join tb_product_info
9      on tb_order_detail.product_id = tb_product_info.product_id
10     left join tb_order_overall
11     on tb_order_detail.order_id = tb_order_overall.order_id
12     where tag = '零食'
13     and abs(datediff(cast(event_time as date), (select max(event_time) from tb_order_overall))) <= 89
14     and status = 1
15     group by 1, 2
16 )
17
18 select
19 product_id
20 ,round(sum(if(cnt >= 2, 1, 0)) / count(uid), 3)
21 from tmp
22 group by 1
23 order by 2 desc, 1
24 limit 3

```

5.10月的新户客单价和获客成本

```
1 with reg
2 as(
3     select
4         uid
5         ,min(event_time) first_time
6     from tb_order_overall
7     group by 1
8 )
9
10 select round(avg(total_amount) ,1) avg_amount,
11         round(avg(cost), 1) avg_cost
12 from
13 (
14     select
15         reg.uid as uid
16         ,avg(total_amount) as total_amount
17         ,(sum(price * cnt) - avg(total_amount)) as cost
18     from tb_order_detail a
19     left join tb_order_overall b
20     on a.order_id = b.order_id
21     left join reg
22     on b.uid = reg.uid
23     and b.event_time = reg.first_time
24     where left(event_time, 7) = '2021-10'
25     and reg.first_time is not null
26     group by 1
27 ) t
```

6.店铺901国庆期间的7日动销率和滞销率

```
1 with tb
2 as (
3     select
4         tpi.product_id
5         ,date(tpi.release_time) as release_dt
6         ,date(too.event_time) as sale_dt
7         ,shop_id
8     from tb_product_info tpi
9     left join tb_order_detail tod
10    on tod.product_id = tpi.product_id
11    left join tb_order_overall too
12    on tod.order_id = too.order_id
13    where too.status = 1
14    and tpi.release_time <= '2021-10-03'
15    and date(too.event_time) between '2021-09-25' and '2021-10-03'
16 ),
17 dt_tb
18 as(
19     select date(event_time) as dt
20     from tb_order_overall
21     where date(event_time) between '2021-10-01' and '2021-10-03'
22     group by 1
23 )
24
25 select
26 dt
```

```

27 ,round(sum(is_sale) / sum(is_shelf), 3) as sale_rate
28 ,round(1 - sum(is_sale) / sum(is_shelf), 3) as unsale_rate
29 from
30 (
31     select
32     dt
33     ,product_id
34     ,max(if(datediff(dt, sale_dt) between 0 and 6, 1, 0)) as is_sale
35     ,max(if(release_dt <= dt, 1, 0)) as is_shelf
36     from tb as t1
37     join dt_tb as t2
38     where shop_id = 901
39     group by 1, 2
40 ) as t
41 group by 1
42 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

```

1  -- 场景说明:
2  -- 用户申请打车 ---->司机接单 ----> 上车 ----> 到达目的地 ----> 订单完成
3  --          ----->取消打车
4  --          ----> 取消打车 (司机、乘客)
5
6  -- 用户申请打车 记录表生成记录, order_id set null
7  -- 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
8  -- 取消打车
9
10 select
11 city
12 ,round(avg(cnt), 3) avg_order_num
13 ,round(avg(tol_fare), 3) avg_income
14 from
15 (
16     select
17     city
18     ,driver_id
19     ,count(1) cnt
20     ,sum(fare) tol_fare
21     from tb_get_car_order
22     left join tb_get_car_record
23     on tb_get_car_order.order_id = tb_get_car_record.order_id
24     where city = '北京'
25     and date(tb_get_car_order.order_time) between '2021-10-01' and '2021-10-07'
26     group by 1, 2
27     having count(1) >=3
28 ) t
29 group by 1

```

2.有取消订单记录的司机平均评分

```

1 with driver_cancel_tb
2 as(
3     select

```

```

4     driver_id
5     from tb_get_car_order
6     where start_time is null
7     and left(order_time, 7) = '2021-10'
8     group by 1
9 )
10
11 select
12 driver_id
13 ,avg_grade
14 from
15 (
16     select
17     driver_id
18     ,round(avg(grade), 1) avg_grade
19     from tb_get_car_order
20     where driver_id in (select driver_id from driver_cancel_tb)
21     group by 1
22
23     union all
24     select
25     '总体'
26     ,round(avg(grade), 1) avg_grade
27     from tb_get_car_order
28     where driver_id in (select driver_id from driver_cancel_tb)
29     group by 1
30 ) t
31 order by 1

```

3.每个城市中评分最高的司机信息

```

1 with driver_info
2 as (
3     select
4     city
5     ,driver_id
6     ,avg(grade) avg_grade
7     ,count(1) / count(distinct left(order_time, 10)) avg_order_num
8     ,sum(mileage) / count(distinct left(order_time, 10)) avg_mileage
9     from tb_get_car_order tbco
10    left join tb_get_car_record tgcr
11    on tbco.order_id = tgcr.order_id
12    group by city, driver_id
13 )
14
15 select
16 city
17 ,driver_id
18 ,round(avg_grade, 1) avg_grade
19 ,round(avg_order_num, 1) avg_order_num
20 ,round(avg_mileage, 3) avg_mileage
21 from
22 (
23     select
24     *
25     ,dense_rank()over(partition by city order by avg_grade desc) rk
26     from driver_info
27 ) a
28 where rk = 1

```

4. 国庆期间近7日日均取消订单量

```

1 select
2 *
3 from
4 (
5     select
6     dt
7     ,round(sum(finish_cnt) over(order by dt rows 6 preceding) /7,2) finish_num_7d
8     ,round(sum(cancel_cnt) over(order by dt rows 6 preceding) /7,2) cancel_num_7d
9     from
10    (
11        select
12        date(event_time) as dt
13        ,sum(if(mileage is not null, 1, 0)) finish_cnt
14        ,sum(if(mileage is null, 1, 0)) cancel_cnt
15        from tb_get_car_record tbgc
16        inner join tb_get_car_order tgco
17        on tbgc.order_id = tgco.order_id
18        where date(event_time) between '2021-09-25' AND '2021-10-03'
19        group by 1
20    ) a
21 ) b
22 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

```

1 with tmp
2 as(
3     select
4     case when hour(event_time) >= 9 and hour(event_time) < 17 then '工作时间'
5         when hour(event_time) >= 17 and hour(event_time) < 20 then '晚高峰'
6         when hour(event_time) >= 7 and hour(event_time) < 9 then '早高峰'
7     else '休息时间' end period
8     ,count(tgcr.id) get_car_num
9     ,avg(timestampdiff(second, event_time, end_time)) avg_wait_time_second
10    ,avg(timestampdiff(second, order_time, start_time)) avg_dispatch_time_second
11    from tb_get_car_record tgcr
12    left join
13    (
14        select * from tb_get_car_order tgco
15        where mileage is not null
16    ) a
17    on tgcr.order_id = a.order_id
18    where dayofweek(event_time) - 1 between 1 and 5
19    group by 1
20 )
21
22 select
23 *
24 from
25 (
26     select
27     period
28     ,get_car_num

```

```

29     ,round(avg_wait_time_second / 60, 1) avg_wait_time
30     ,round(avg_dispatch_time_second / 60, 1) avg_dispatch_time
31   from tmp
32   where period = '工作时间'
33
34   union all
35
36   select
37   period
38   ,get_car_num
39   ,round(avg_wait_time_second / 60, 1) avg_wait_time
40   ,round(avg_dispatch_time_second / 60, 1) avg_dispatch_time
41   from tmp
42   where period = '休息时间'
43
44   union all
45
46   select
47   period
48   ,get_car_num
49   ,round(avg_wait_time_second / 60, 1) avg_wait_time
50   ,round(avg_dispatch_time_second / 60, 1) avg_dispatch_time
51   from tmp
52   where period = '晚高峰'
53
54   union all
55
56   select
57   period
58   ,get_car_num
59   ,round(avg_wait_time_second / 60, 1) avg_wait_time
60   ,round(avg_dispatch_time_second / 60, 1) avg_dispatch_time
61   from tmp
62   where period = '早高峰'
63 ) b
64 order by 2

```

6.各城市最大同时等车人数

```

1  -- 打车 event_time 1
2  -- 打车取消 order_id is null -1
3  -- 等车取消 start_time is null -1
4  -- 上车 -1
5
6  with tmp
7  as(
8    select
9    city
10   ,event_time as uv_time
11   ,1 uv
12   from tb_get_car_record
13
14   union all
15   select
16   city
17   ,end_time as uv_time
18   ,-1 uv
19   from tb_get_car_record
20   where order_id is null

```

```

21
22     union all
23     select
24     city
25     ,ifnull(start_time, finish_time) as uv_time
26     ,-1 uv
27     from tb_get_car_order tgco
28     left join tb_get_car_record tgcr
29     on tgco.order_id = tgcr.order_id
30 )
31 select
32 city
33 ,max(uv_cnt)
34 from
35 (
36     select
37     city
38     ,sum(uv)over(partition by city order by uv_time, uv desc) uv_cnt
39     from tmp
40     where left(uv_time, 7) = '2021-10'
41 ) a
42 group by 1
43 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

```

1 select
2 style_id
3 ,count(1) SUP_num
4 from product_tb
5 group by 1
6 order by 2 desc

```

2.某宝店铺的实际销售额与客单价

```

1 select
2 sum(sales_price)
3 ,round(sum(sales_price) / count(distinct user_id), 2)
4 from sales_tb

```

3.某宝店铺折扣率

```

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

```

4.某宝店铺动销率与售罄率

```
1 SELECT
2 t1.style_id
3 ,round(sales_cnt / (inventory_cnt - sales_cnt) * 100 ,2) `pin_rate(%)`
4 ,round(gmv / cost * 100 ,2) `sell-through_rate(%)`
5 from
6 (
7     SELECT
8     style_id
9     ,sum(inventory) inventory_cnt
10    ,sum(tag_price * inventory) cost
11    from product_tb
12    group by style_id
13 ) t1
14 LEFT JOIN
15 (
16     SELECT
17     style_id
18     ,sum(sales_num) sales_cnt
19     ,sum(sales_price) gmv
20    from sales_tb
21    left join product_tb
22    on sales_tb.item_id = product_tb.item_id
23    group by style_id
24 )t2
25 on t1.style_id = t2.style_id
26 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

```
1 with tmp
2 as(
3     select
4     *
5     ,row_number()over(order by dt) rk
6     from
7     (
8         select
9         user_id
10        ,sales_date dt
11        from sales_tb
12        group by 1, 2
13    ) t1
14 )
15
16 select
17 user_id
18 ,count(1) days_count
19 from
20 (
21     select
22     dt
23     ,user_id
24     ,date_sub(dt, interval 1 day) tmp_dt
25    from tmp
26 ) a
```



```
27 group by 1
28 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

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

2.牛客直播开始时各直播间在线人数

```
1
```