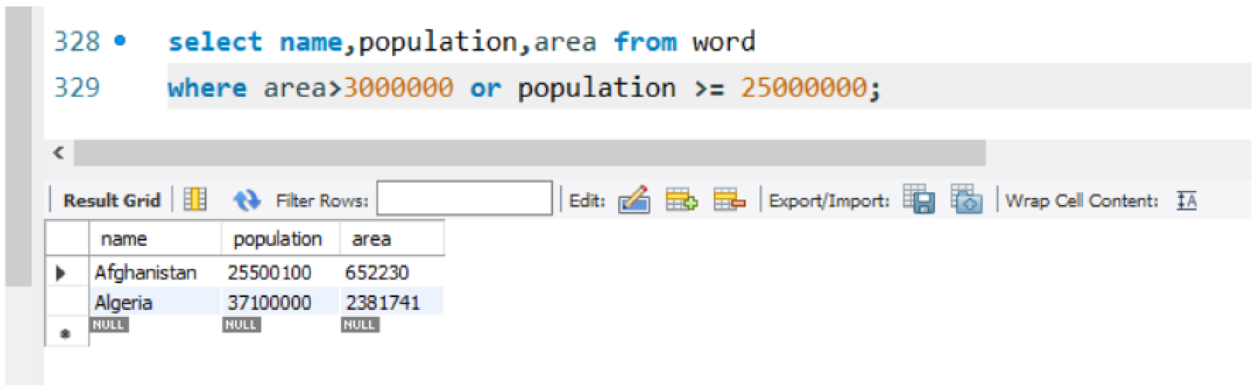
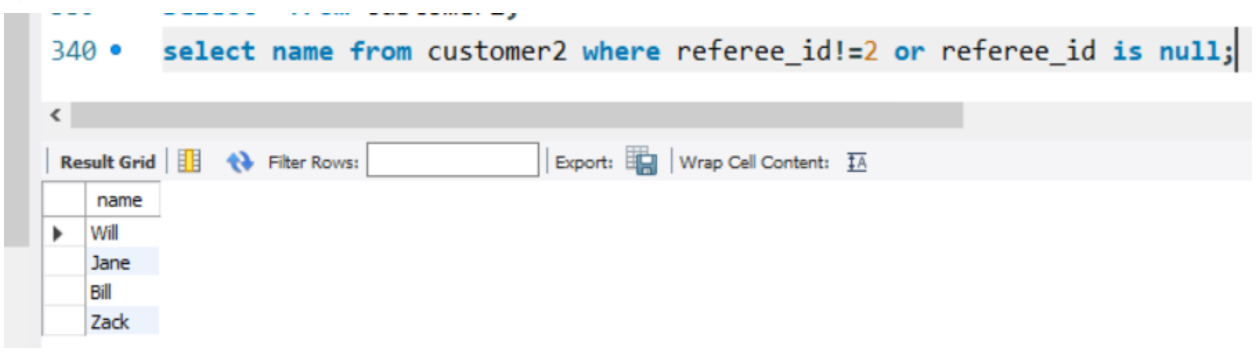
Q51.



```sql
select name,population,area from word
where area>3000000 or population >= 25000000;
```

```
   name         population   area
▶  Afghanistan  25500100     652230
   Algeria      37100000     2381741
*  NULL         NULL         NULL
```

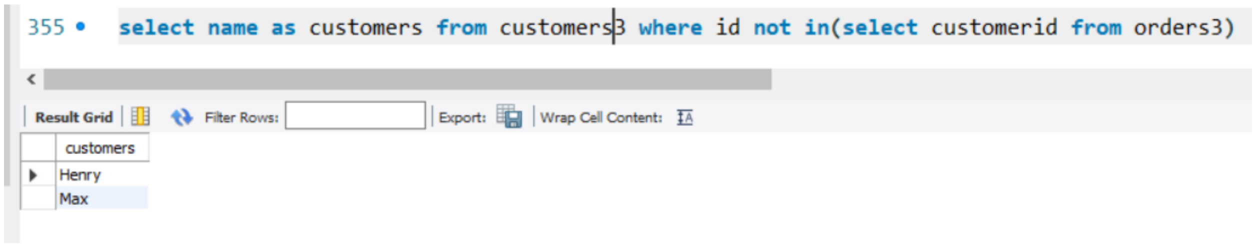
Q52.



```sql
select name from customer2 where referee_id!=2 or referee_id is null;
```

```
   name
▶  Will
   Jane
   Bill
   Zack
```

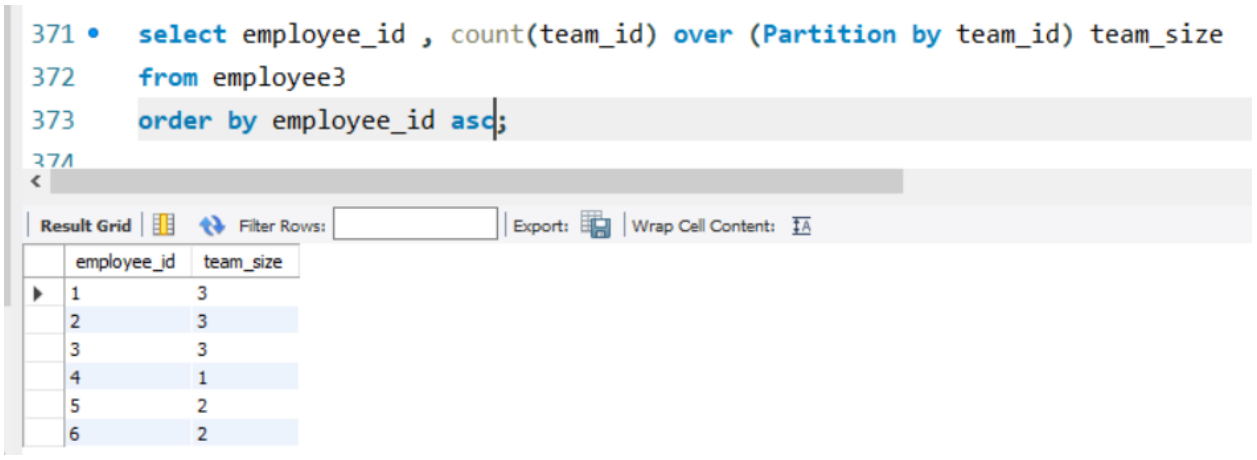
Q53.



```sql
select name as customers from customers3 where id not in(select customerid from orders3)
```

```
   customers
▶  Henry
   Max
```

Q54.

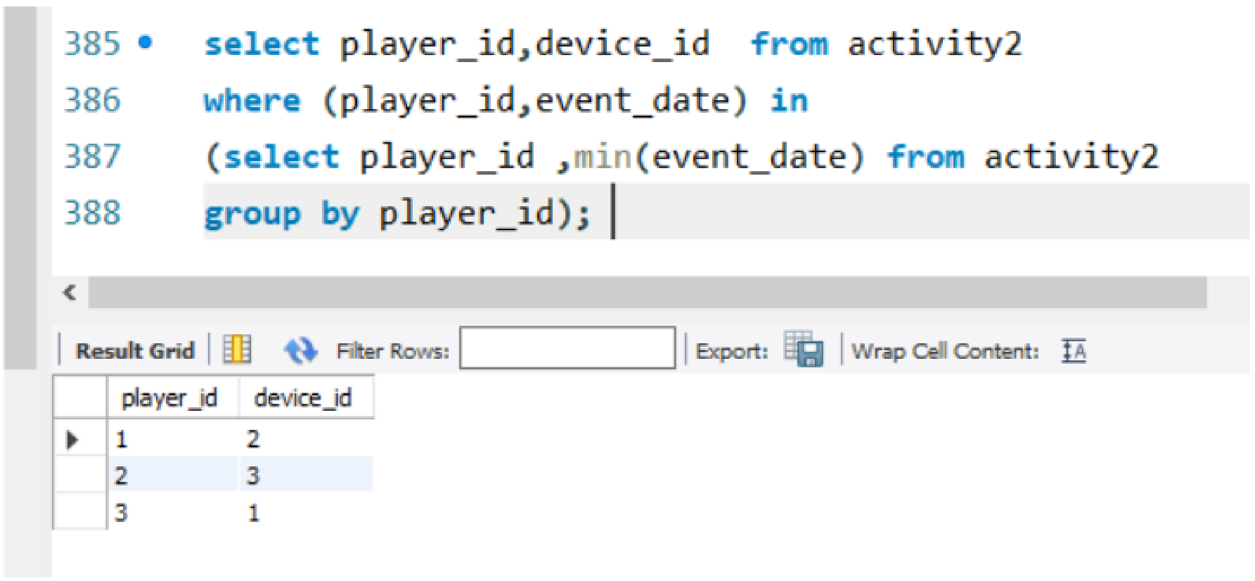


```sql
select employee_id , count(team_id) over (Partition by team_id) team_size
from employee3
order by employee_id asc;
```

```
   employee_id  team_size
▶  1            3
   2            3
   3            3
   4            1
   5            2
   6            2
```

Q55.

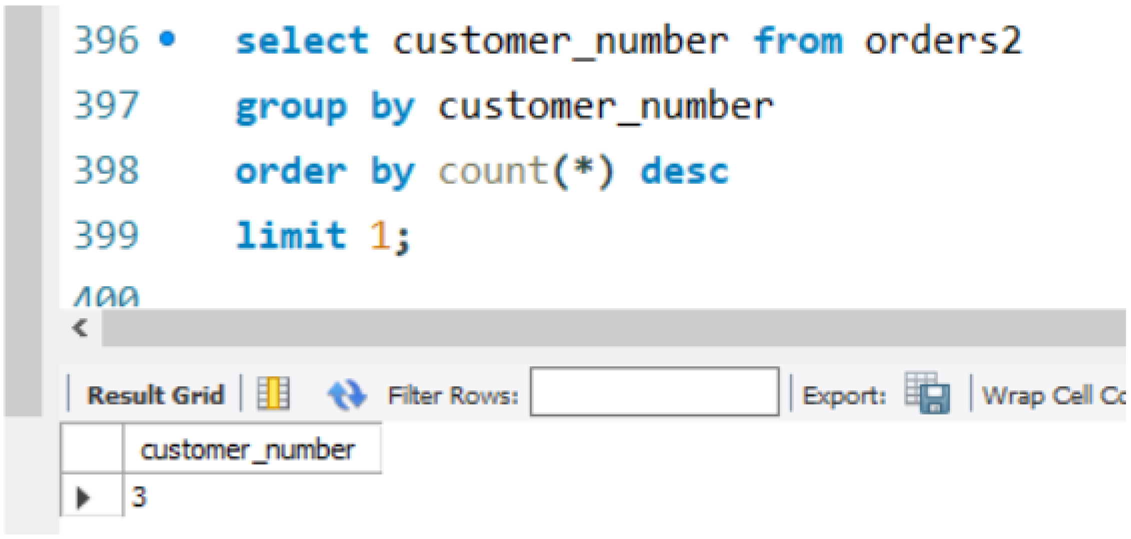
Q56.



```sql
select player_id,device_id  from activity2
where (player_id,event_date) in
(select player_id ,min(event_date) from activity2
group by player_id);
```

```
player_id  device_id
1          2
2          3
3          1
```

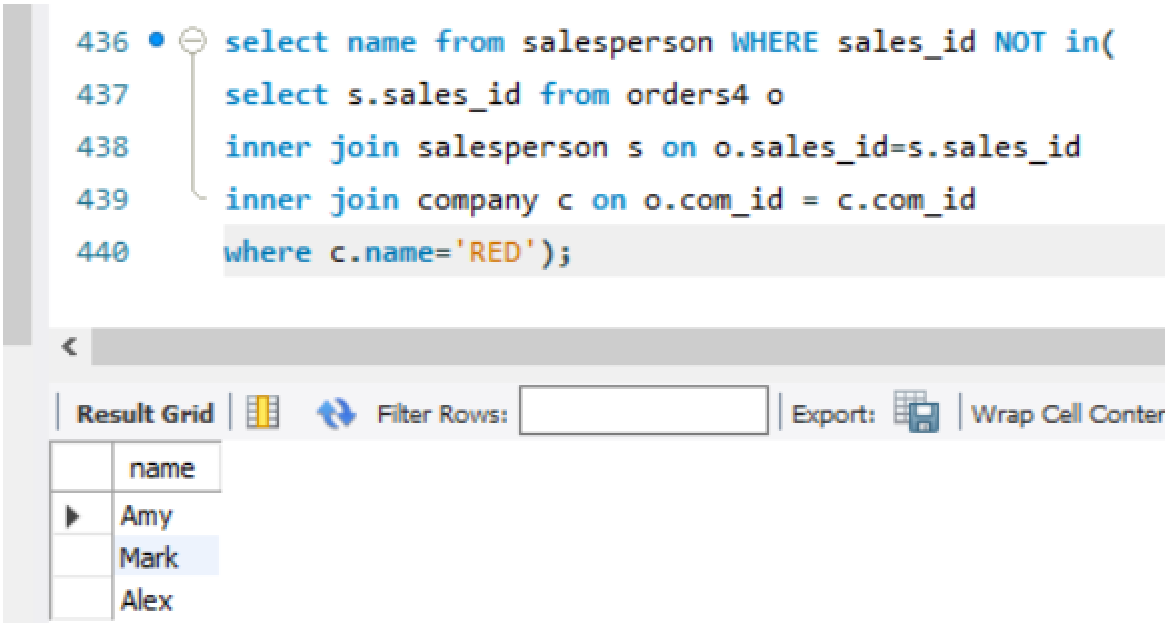
Q57.



```sql
select customer_number from orders2
group by customer_number
order by count(*) desc
limit 1;
```

```
customer_number
3
```

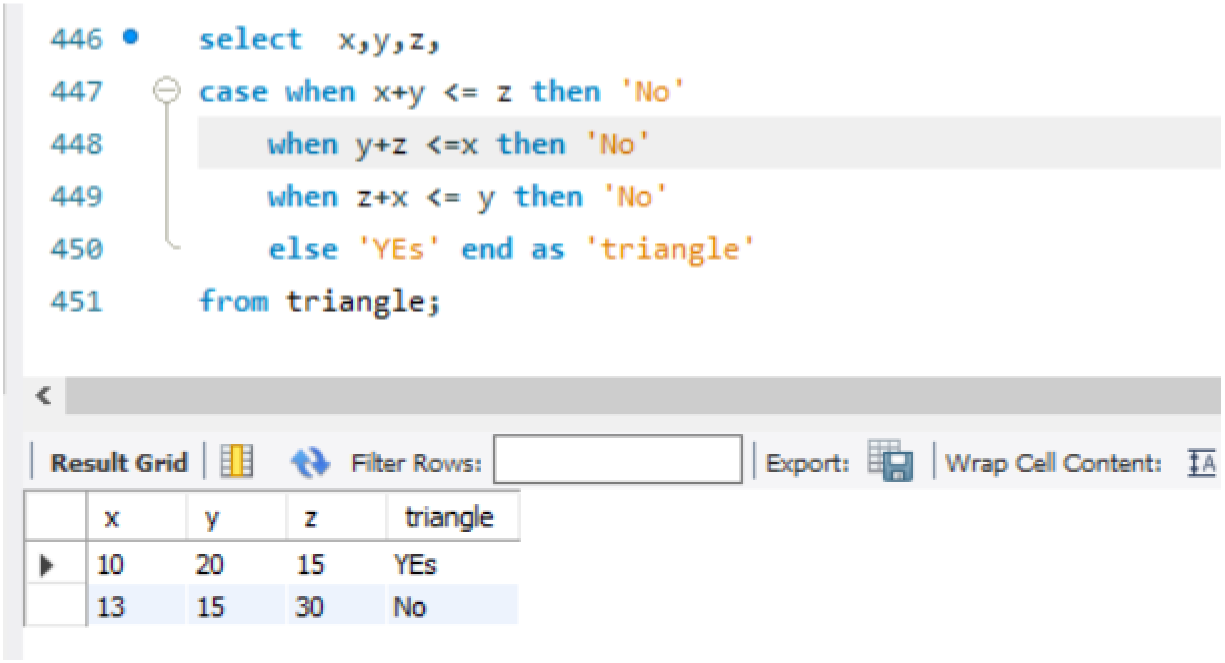
Q59.



```
select name from salesperson WHERE sales_id NOT in(
select s.sales_id from orders4 o
inner join salesperson s on o.sales_id=s.sales_id
inner join company c on o.com_id = c.com_id
where c.name='RED');
```

name
Amy
Mark
Alex

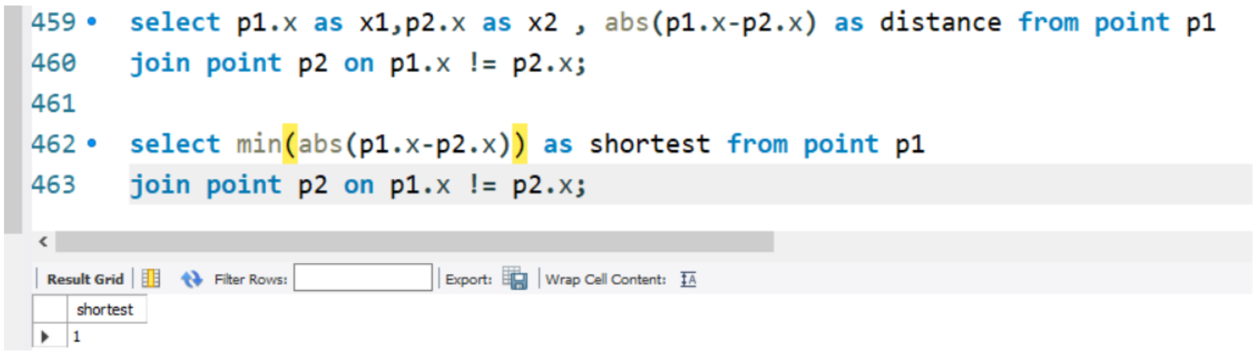
Q60.



```
select  x,y,z,
case when x+y <= z then 'No'
    when y+z <=x then 'No'
    when z+x <= y then 'No'
    else 'YEs' end as 'triangle'
from triangle;
```

x	y	z	triangle
10	20	15	YEs
13	15	30	No

Q61.



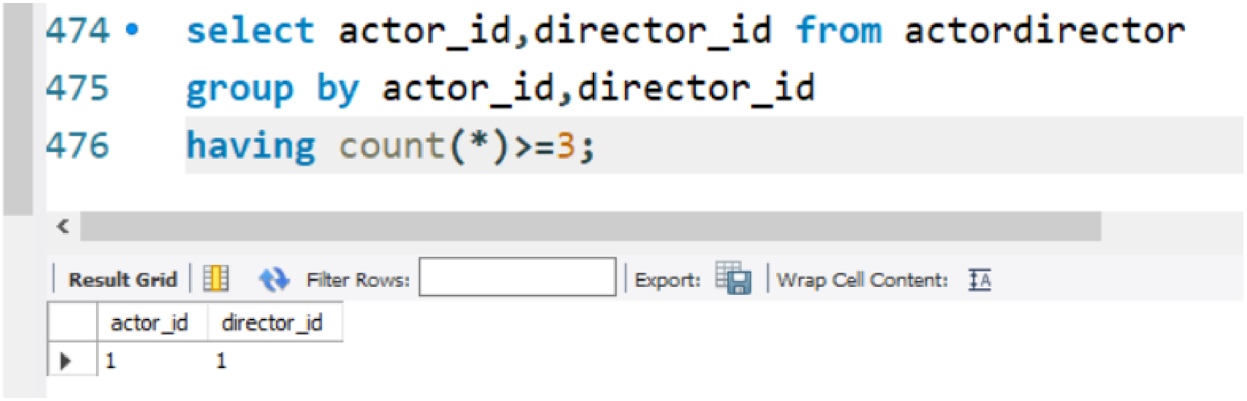
```
select p1.x as x1,p2.x as x2 , abs(p1.x-p2.x) as distance from point p1
join point p2 on p1.x != p2.x;

select min(abs(p1.x-p2.x)) as shortest from point p1
join point p2 on p1.x != p2.x;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

shortest
1

Q62.

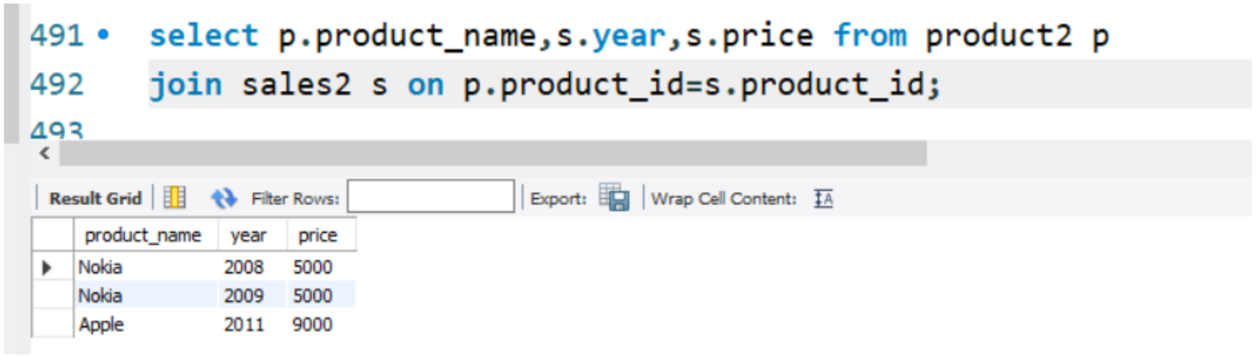


```
select actor_id,director_id from actordirector
group by actor_id,director_id
having count(*)>=3;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

actor_id	director_id
1	1

Q63.

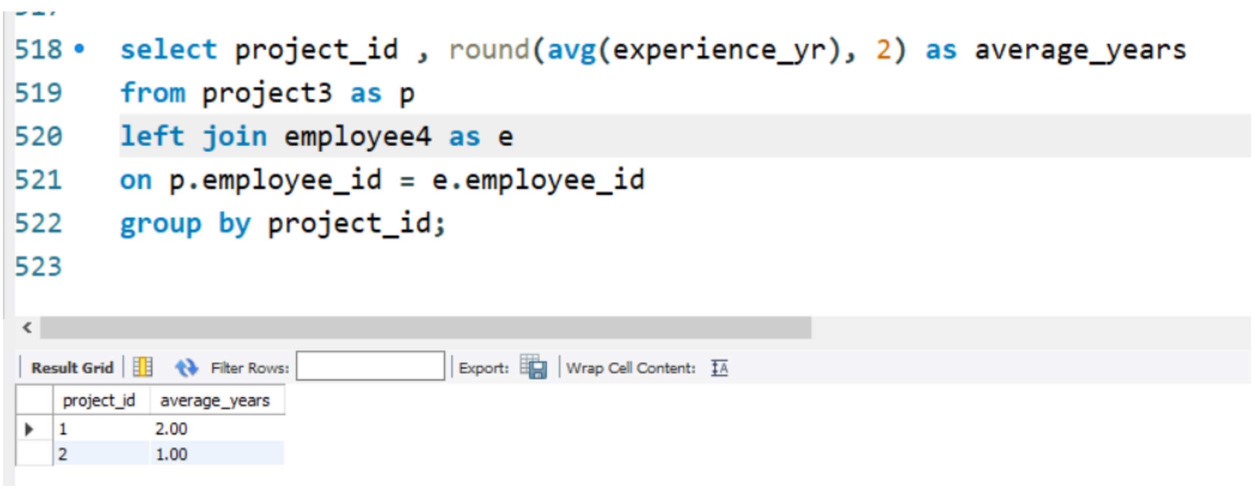


```
select p.product_name,s.year,s.price from product2 p
join sales2 s on p.product_id=s.product_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

product_name	year	price
Nokia	2008	5000
Nokia	2009	5000
Apple	2011	9000

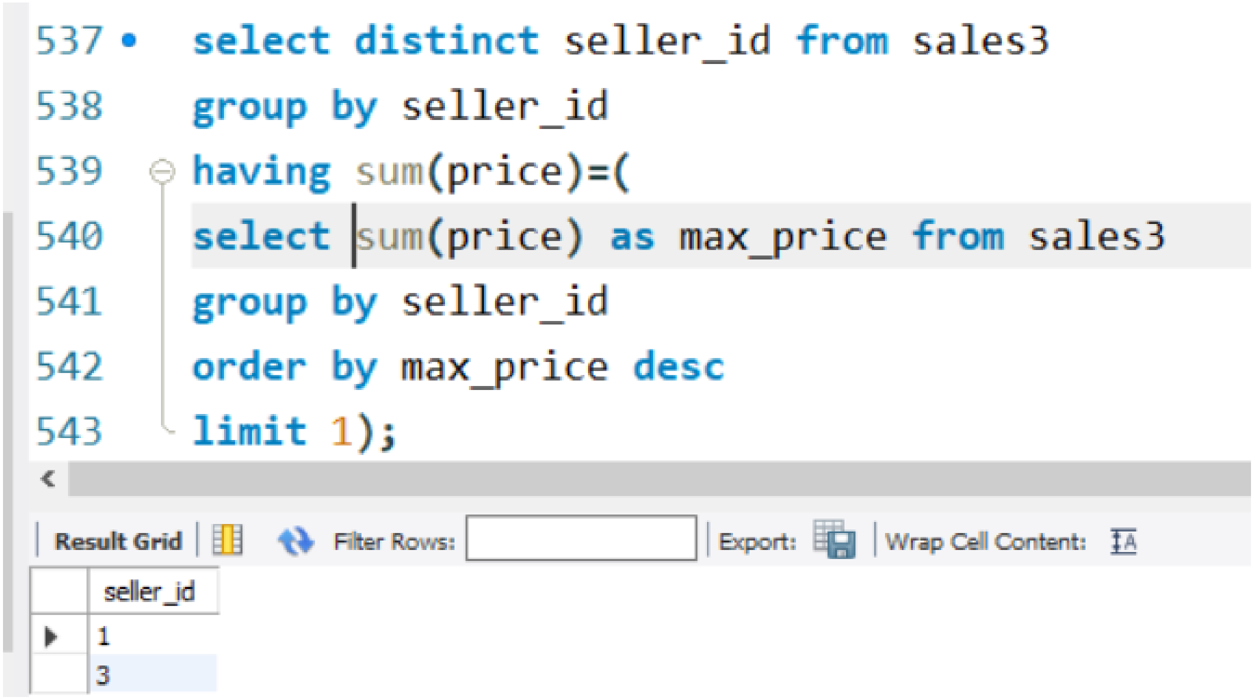
Q64.



```
select project_id , round(avg(experience_yr), 2) as average_years
from project3 as p
left join employee4 as e
on p.employee_id = e.employee_id
group by project_id;
```

```
project_id  average_years
1           2.00
2           1.00
```

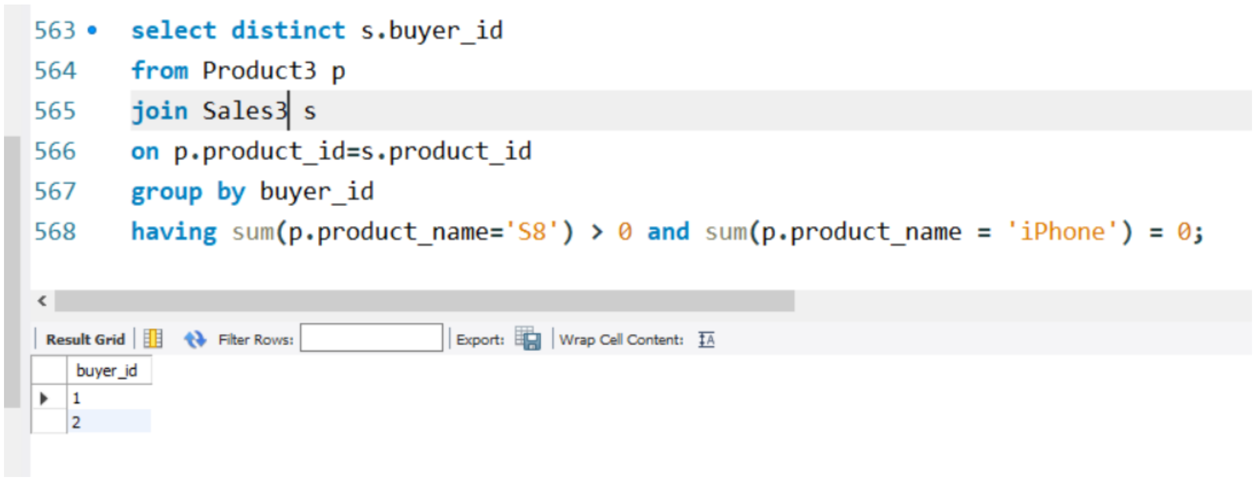
Q65.



```
select distinct seller_id from sales3
group by seller_id
having sum(price)=(
select sum(price) as max_price from sales3
group by seller_id
order by max_price desc
limit 1);
```

```
seller_id
1
3
```

Q66.

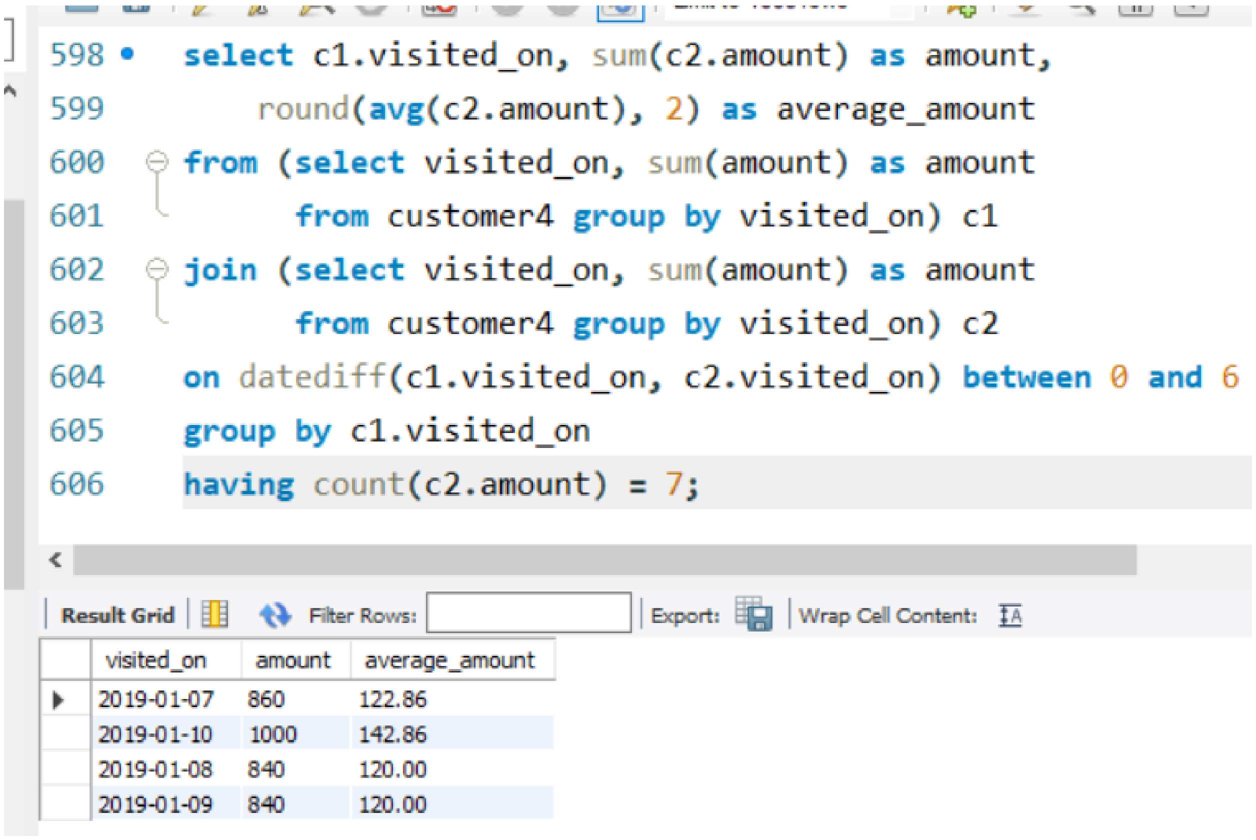


```sql
select distinct s.buyer_id
from Product3 p
join Sales3 s
on p.product_id=s.product_id
group by buyer_id
having sum(p.product_name='S8') > 0 and sum(p.product_name = 'iPhone') = 0;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

buyer_id
1
2

Q67.

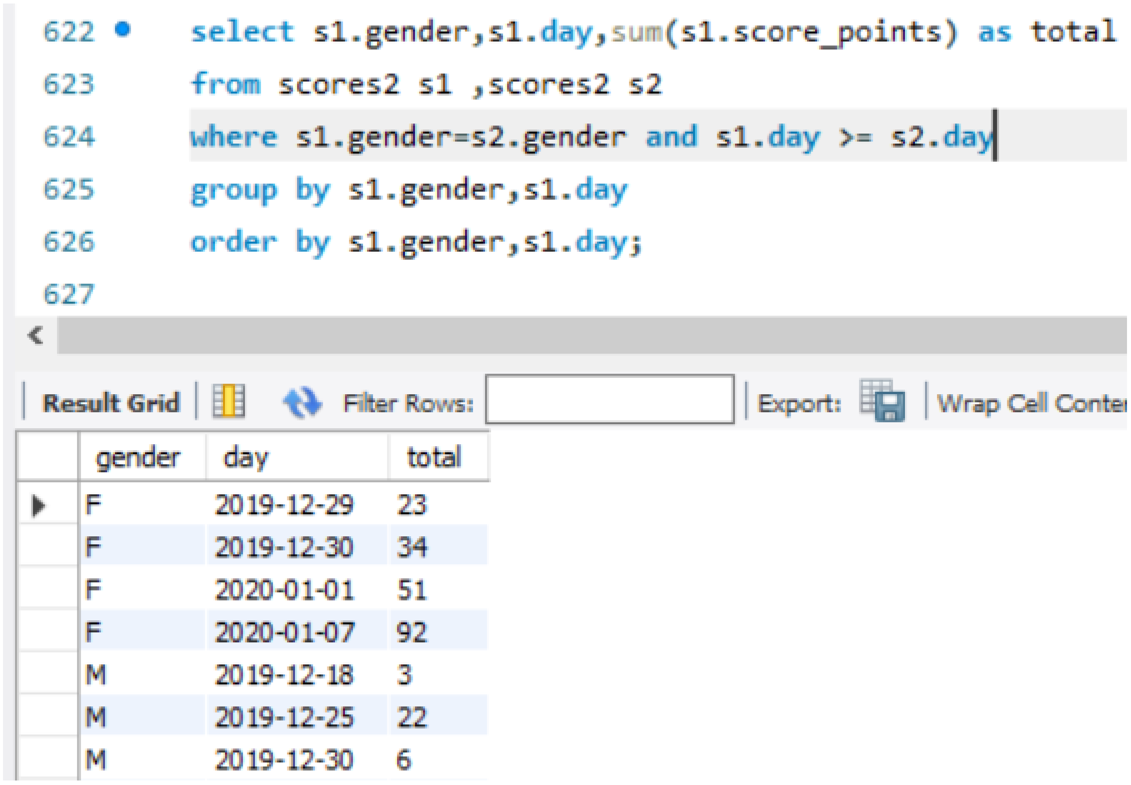


```sql
select c1.visited_on, sum(c2.amount) as amount,
       round(avg(c2.amount), 2) as average_amount
from (select visited_on, sum(amount) as amount
        from customer4 group by visited_on) c1
join (select visited_on, sum(amount) as amount
        from customer4 group by visited_on) c2
on datediff(c1.visited_on, c2.visited_on) between 0 and 6
group by c1.visited_on
having count(c2.amount) = 7;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

visited_on	amount	average_amount
2019-01-07	860	122.86
2019-01-10	1000	142.86
2019-01-08	840	120.00
2019-01-09	840	120.00

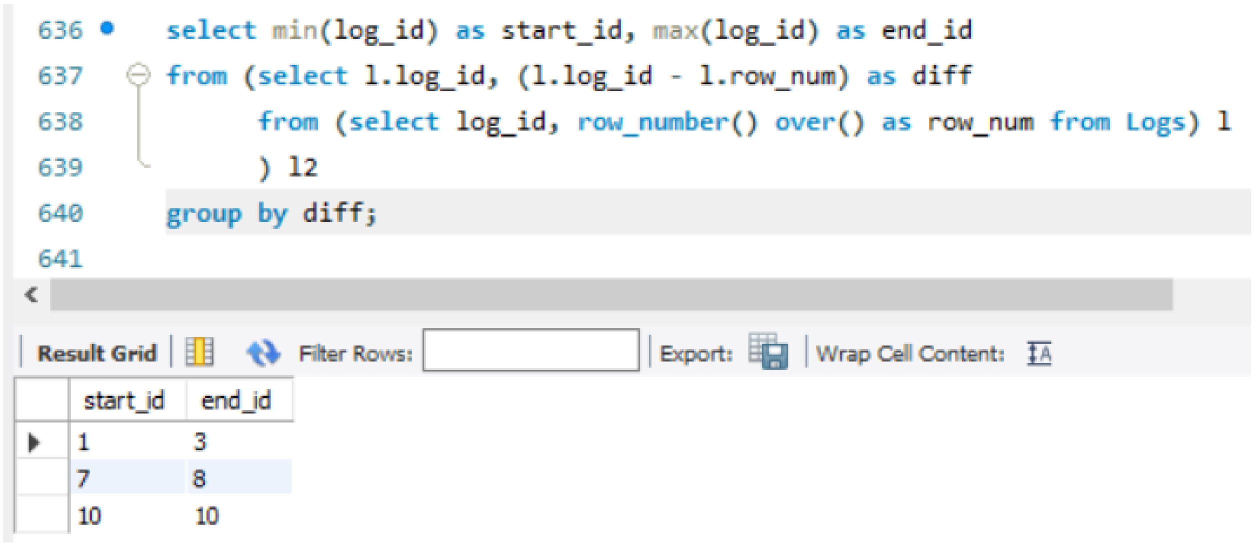
Q68.



```
select s1.gender,s1.day,sum(s1.score_points) as total
from scores2 s1 ,scores2 s2
where s1.gender=s2.gender and s1.day >= s2.day
group by s1.gender,s1.day
order by s1.gender,s1.day;
```

gender	day	total
F	2019-12-29	23
F	2019-12-30	34
F	2020-01-01	51
F	2020-01-07	92
M	2019-12-18	3
M	2019-12-25	22
M	2019-12-30	6

Q69.



```
select min(log_id) as start_id, max(log_id) as end_id
from (select l.log_id, (l.log_id - l.row_num) as diff
        from (select log_id, row_number() over() as row_num from Logs) l
        ) l2
group by diff;
```

start_id	end_id
1	3
7	8
10	10

Q70.

select a.student_id, a.student_name, b.subject_name, count(c.subject_name) as attended_exams
from Students as a
join Subjects as b
left join Examinations as c
on a.student_id = c.student_id and b.subject_name = c.subject_name
group by a.student_id, b.subject_name;

Q71.

```
select a.employee_id as EMPLOYEE_ID
from
   Employees as a
   left join
   Employees as b on a.manager_id = b.employee_id
   left join
   Employees as c on b.manager_id = c.employee_id
   left join
   Employees as d on c.manager_id = d.employee_id
where
   a.employee_id != 1
   and
   d.employee_id = 1;
```

Q72.

```
select date_format(trans_date,"%Y-%m") as month, country,
   count(id) as trans_count,
   sum(case when state='approved' then 1  else 0 end) as approved_count,
   sum(amount) as trans_total_amount,
   sum(case when state='approved' then amount  else 0 end) as approved_total_amount
from transactions
group by month, country
```

Q73.

```
select round(avg(daily_count), 2) as average_daily_percent
from (select count(distinct b.post_id)/count(distinct a.post_id)*100 as daily_count
   from actions a
   left join removals b
   on a.post_id = b.post_id
   where extra = 'spam'
   group by action_date
   ) b
```

Q74.

```
SELECT
round((count(distinct c.player_id) / (select count(distinct player_id) from activity)),2)as fraction
FROM
CTE c
JOIN Activity a
on c.player_id = a.player_id
and datediff(c.event_start_date, a.event_date) = -1
```

Q75.
```
SELECT
round((count(distinct c.player_id) / (select count(distinct player_id) from activity)),2)as fraction
FROM
CTE c
JOIN Activity a
on c.player_id = a.player_id
and datediff(c.event_start_date, a.event_date) = -1
```

Q76.
```
select company_id, employee_id, employee_name, round(salary - salary*tax, 0) as salary
from
(
  select *,
  case when max(salary) over(partition by company_id) < 1000 then 0
       when max(salary) over(partition by company_id) between 1000
       and 10000 then 0.24
       else 0.49 end as tax
  from Salaries
) x
```
Q77.
```
select e.left_operand, e.operator, e.right_operand,
    case
        when e.operator = '<' then if(l.value < r.value,'true','false')
        when e.operator = '>' then if(l.value > r.value,'true','false')
        else if(l.value = r.value,'true','false')
    end as value
from expressions e
left join variables l on e.left_operand = l.name
left join variables r on e.right_operand = r.name
```
Q78.
```
select c.name as country
from Person p
inner join Country c
on left (p.phone_number,3) = c.country_code
inner join (select caller_id as id, duration
            from Calls
            union all
            select callee_id as id, duration
            from Calls) phn
on p.id = phn.id
group by country
having avg(duration) > (select avg(duration) from Calls)
```

Q80.
```
WITH yearly_spend AS (
  SELECT
    product_id,
    EXTRACT(YEAR FROM transaction_date) AS year,
    SUM(spend) AS total_spend
  FROM user_transactions
  GROUP BY product_id, year
)
SELECT
  curr_year.year,
  curr_year.product_id,
  curr_year.total_spend AS curr_year_spend,
  prev_year.total_spend AS prev_year_spend,
  ROUND(((curr_year.total_spend - prev_year.total_spend) / prev_year.total_spend) * 100, 2)
AS yoy_rate
FROM yearly_spend curr_year
LEFT JOIN yearly_spend prev_year
  ON curr_year.product_id = prev_year.product_id
  AND curr_year.year = prev_year.year + 1
ORDER BY curr_year.product_id, curr_year.year;;
```

Q82.
```
SELECT
  EXTRACT(MONTH FROM curr_month.event_date) AS mth,
  COUNT(DISTINCT curr_month.user_id) AS monthly_active_users
FROM user_actions AS curr_month
WHERE EXISTS (
  SELECT last_month.user_id
  FROM user_actions AS last_month
  WHERE last_month.user_id = curr_month.user_id
    AND EXTRACT(MONTH FROM last_month.event_date) =
    EXTRACT(MONTH FROM curr_month.event_date - interval '1 month')
)
  AND EXTRACT(MONTH FROM curr_month.event_date) = 7
  AND EXTRACT(YEAR FROM curr_month.event_date) = 2022
GROUP BY EXTRACT(MONTH FROM curr_month.event_date)
```
Q83
```
WITH searches_expanded AS (
  SELECT searches
  FROM search_frequency
  GROUP BY
    searches,
    GENERATE_SERIES(1, num_users))
```

```
SELECT
  ROUND(PERCENTILE_CONT(0.50) WITHIN GROUP (
    ORDER BY searches)::DECIMAL, 1) AS  median
FROM searches_expanded;
```

Q84

```
SELECT
  advertiser.user_id,
  advertiser.status,
  payment.paid
FROM advertiser
LEFT JOIN daily_pay AS payment
  ON advertiser.user_id = payment.user_id
UNION
SELECT
  payment.user_id,
  advertiser.status,
  payment.paid
FROM daily_pay AS payment
LEFT JOIN advertiser
  ON advertiser.user_id = payment.user_id
```

Q85

```
WITH running_time
AS (
  SELECT
    server_id,
    session_status,
    status_time AS start_time,
    LEAD(status_time) OVER (
      PARTITION BY server_id
      ORDER BY status_time) AS stop_time
  FROM server_utilization
)

SELECT
  DATE_PART('days', JUSTIFY_HOURS(SUM(stop_time - start_time))) AS total_uptime_days
FROM running_time
WHERE session_status = 'start'
  AND stop_time IS NOT NULL;
```

Q86

```
WITH payments AS (
  SELECT
```

```
    merchant_id,
    EXTRACT(EPOCH FROM transaction_timestamp -
      LAG(transaction_timestamp) OVER(
        PARTITION BY merchant_id, credit_card_id, amount
        ORDER BY transaction_timestamp)
    )/60 AS minute_difference
  FROM transactions)

SELECT COUNT(merchant_id) AS payment_count
FROM payments
WHERE minute_difference <= 10;
```

Q87

```
with totorders as(
select o.order_id, o.customer_id, o.trip_id,o.status, o.order_timestamp,
t.estimated_delivery_timestamp as etimestamp, t.actual_delivery_timestamp as atimestamp,
c.signup_timestamp
from orders as o
join trips as t on t.trip_id = o.trip_id
join customers as c on c.customer_id = o.customer_id
where extract(month from c.signup_timestamp) = 06
AND extract(year from c.signup_timestamp) = 2022
and c.signup_timestamp+interval '14 days' > o.order_timestamp
),
```

Q88

```
select s1.gender, s1.day, sum(s2.score_points) as total from Scores s1, Scores s2
where s1.gender = s2.gender and s1.day >= s2.day
group by s1.gender, s1.day
order by s1.gender, s1.day
```

Q89

```
select c.name as country
from Person p
inner join Country c
on left (p.phone_number,3) = c.country_code
inner join (select caller_id as id, duration
            from Calls

            union all

            select callee_id as id, duration
            from Calls) phn
on p.id = phn.id
```

```
group by country
having avg(duration) > (select avg(duration) from Calls)
```

Q91

```
select department_salary.pay_month, department_id,
   case
      when department_avg>company_avg then 'higher'
      when department_avg<company_avg then 'lower'
      else 'same'
   end as comparison
from
   (
     select department_id, avg(amount) as department_avg, date_format(pay_date, '%Y-%m') as
pay_month
     from salary join employee on salary.employee_id = employee.employee_id
     group by department_id, pay_month
   ) as department_salary
join
   (
     select avg(amount) as company_avg,  date_format(pay_date, '%Y-%m') as pay_month
      from salary
      group by date_format(pay_date, '%Y-%m')
   ) as company_salary
on department_salary.pay_month = company_salary.pay_month
```

Q92

```
select t1.install_date as install_dt, count(t1.install_date) as installs,
   round(count(t2.event_date) / count(*), 2) as Day1_retention
from (
   select player_id, min(event_date) as install_date
   from Activity
   group by 1
) t1
left join Activity t2
on date_add(t1.install_date, interval 1 day) = t2.event_date
   and t1.player_id = t2.player_id
group by 1
order by 1
```

Q93

```
SELECT group_id,
       player_id
FROM   (SELECT p.group_id,
               ps.player_id,
               Sum(ps.score) AS score
        FROM   players p INNER JOIN
               (SELECT first_player AS player_id,
                       first_score  AS score
                FROM   matches
                UNION ALL
                SELECT second_player AS player_id,
                       second_score  AS score
                FROM   matches) ps
        ON  p.player_id = ps.player_id
        GROUP  BY ps.player_id
        ORDER  BY group_id,
                  score DESC,
                  player_id) top_scores
GROUP  BY group_id
```

Q97

```
SELECT
  ROUND(COUNT(texts.email_id)::DECIMAL
    /COUNT(DISTINCT emails.email_id),2) AS activation_rate
FROM emails
LEFT JOIN texts
  ON emails.email_id = texts.email_id
  AND texts.signup_action = 'Confirmed';
```

Q98

```
SELECT
  user_id,
  tweet_date,
  ROUND(AVG(tweet_count) OVER (
    PARTITION BY user_id
    ORDER BY tweet_date
    ROWS BETWEEN 2 PRECEDING AND CURRENT ROW)
  ,2) AS rolling_avg_3d
FROM tweets;
```

Q99

```
WITH snaps_statistics AS (
  SELECT
    age.age_bucket,
    SUM(CASE WHEN activities.activity_type = 'send'
      THEN activities.time_spent ELSE 0 END) AS send_timespent,
    SUM(CASE WHEN activities.activity_type = 'open'
      THEN activities.time_spent ELSE 0 END) AS open_timespent,
    SUM(activities.time_spent) AS total_timespent
  FROM activities
 INNER JOIN age_breakdown AS age
    ON activities.user_id = age.user_id
  WHERE activities.activity_type IN ('send', 'open')
  GROUP BY age.age_bucket)

SELECT
  age_bucket,
  ROUND(100.0 * send_timespent / total_timespent, 2) AS send_perc,
  ROUND(100.0 * open_timespent / total_timespent, 2) AS open_perc
FROM snaps_statistics;
```

Q100

```
SELECT p.person_id
FROM (
  SELECT person_id, MAX(follower_count) AS max_follower_count
  FROM (
    SELECT person_id, follower_count
    FROM company_followers
    WHERE person_id IS NOT NULL

    UNION ALL

    SELECT person_id, follower_count
    FROM person_followers
  ) AS followers
  GROUP BY person_id
) AS p
JOIN (
  SELECT person_id, company_id, MAX(follower_count) AS max_follower_count
  FROM company_followers
  WHERE company_id IS NOT NULL
  GROUP BY person_id, company_id
) AS c ON p.person_id = c.person_id AND p.max_follower_count > c.max_follower_count
ORDER BY p.person_id ASC;
```