Binxiao's Leetcode SQL (updated to Q1294)

Q175

```
SELECT FirstName, LastName, City, State
FROM Person
LEFT JOIN Address
USING(PersonId)
```

Q176

```
SELECT IFNULL((
    SELECT DISTINCT Salary
    FROM Employee
    ORDER BY Salary DESC
    LIMIT 1
    OFFSET 1
),NULL) AS SecondHighestSalary
```

Q177

```
CREATE FUNCTION getNthHighestSalary(N ) RETURNS
BEGIN
  SET N = N-1;
  RETURN (
      # Write your MySQL query statement below.
      SELECT (SELECT DISTINCT Salary
            FROM Employee
            ORDER BY Salary DESC
            LIMIT 1
            OFFSET N)

  );
END
```

Q178

```
SELECT s0.Score,
       (SELECT COUNT(DISTINCT s1.Score) + 1
       FROM Scores s1
       WHERE s1.Score > s0.Score) AS Rank
FROM Scores s0
ORDER BY Rank
```

Q180

```
SELECT DISTINCT l1.Num AS ConsecutiveNums
FROM Logs l1
INNER JOIN Logs l2
ON (l1.Id + 1 = L2.Id) AND (l1.Num = l2.Num)
INNER JOIN Logs l3
ON (l2.Id + 1 = l3.Id) AND (l1.Num = l3.Num)
```

Q181

```
SELECT e.Name AS Employee
FROM Employee e
LEFT JOIN Employee m
ON e.ManagerId = m.Id
WHERE e.Salary > m.Salary
```

Q182

```
SELECT Email
FROM Person
GROUP BY Email
HAVING COUNT(*) > 1
```

Q183

```
SELECT Name AS Customers
FROM Customers c
WHERE c.Id NOT IN (SELECT CustomerId FROM Orders)
```

Q184

```
SELECT d.Name AS Department, e.Name AS Employee, cte.Salary
FROM
(SELECT DepartmentId, MAX(Salary) AS Salary
FROM Employee
GROUP BY DepartmentId
) cte
INNER JOIN Employee e
USING(DepartmentId, Salary)
INNER JOIN Department d
ON d.Id = cte.DepartmentId
```

Q185

```
```

```
SELECT d.Name AS Department, e.Name AS Employee, e.Salary
FROM Employee e
INNER JOIN Department d
ON e.DepartmentId = d.Id
WHERE (SELECT COUNT(DISTINCT e1.Salary) FROM Employee e1 WHERE e1.Departm
entId = e.DepartmentId AND e1.Salary >= e.Salary) <=3
```

Q196

```
DELETE p1
FROM Person p1
INNER JOIN Person p2
USING(Email)
WHERE p1.Id > p2.Id
```

Q197

```
SELECT w1.Id AS Id
FROM Weather w1, Weather w2
WHERE DATEDIFF(w1.RecordDate,w2.RecordDate) = 1
      AND w1.Temperature > w2.Temperature
```

Q262

```
SELECT Request_at AS Day, ROUND(AVG(Status <> 'completed'),2) AS 'Cancell
ation Rate'
FROM Trips t
INNER JOIN Users c
ON t.Client_Id = c.Users_Id
INNER JOIN Users d
ON t.Driver_Id = d.Users_Id
WHERE c.Banned = 'No' AND d.Banned = 'No' AND Request_at BETWEEN '2013-10
-01' AND '2013-10-03'
GROUP BY Request_at
ORDER BY Day
```

Q511

```
SELECT player_id, MIN(event_date) AS first_login
FROM Activity
GROUP BY player_id
```

Q512

```
SELECT player_id, device_id
FROM Activity
```

```
INNER JOIN
(SELECT player_id, MIN(event_date) AS event_date
FROM Activity
GROUP BY player_id) cte
USING(player_id,event_date)
```

Q534

```
SELECT a1.player_id, a1.event_date, SUM(a2.games_played) AS games_played_
so_far
FROM Activity a1
INNER JOIN Activity a2
USING(player_id)
WHERE a2.event_date <= a1.event_date
GROUP BY a1.player_id, a1.event_date
```

Q550

```
SELECT
    ROUND((SELECT COUNT(*)
    FROM Activity
    INNER JOIN
        (SELECT player_id, MIN(event_date) AS init
        FROM Activity
        GROUP BY player_id) cte
    USING(player_id)
    WHERE event_date = init + 1) / (SELECT COUNT(DISTINCT player_id) FROM
 Activity),2) AS fraction
```

Q569

```
SELECT e.*
FROM
(SELECT  e1.Id, SUM(CASE
        WHEN e1.Salary > e.Salary OR (e1.Salary = e.Salary AND e1.Id > e.
Id) THEN 1
        WHEN e1.Salary < e.Salary OR (e1.Salary = e.Salary AND e1.Id < e.
Id) THEN -1
        ELSE 0 END) dif
FROM Employee e
INNER JOIN Employee e1
ON e.Company = e1.Company
GROUP BY e1.Id) cte
INNER JOIN Employee e
USING(Id)
WHERE dif BETWEEN -1 AND 1
ORDER BY Company,Salary
```

Q570

```
SELECT m.Name
FROM Employee m
LEFT JOIN Employee e
ON m.Id = e.ManagerId
GROUP BY m.Id
HAVING COUNT(*) >= 5
```

Q571

```
SELECT IF(
    (SELECT SUM(Frequency) FROM Numbers) % 2 = 1,

(
SELECT Number
FROM
    (SELECT n.Number,
            SUM(CASE WHEN n1.Number < n.Number THEN n1.Frequency ELSE 0 E
ND) AS lower,
            SUM(CASE WHEN n1.Number = n.Number THEN n1.Frequency ELSE 0 E
ND) AS same,
            SUM(CASE WHEN n1.Number > n.Number THEN n1.Frequency ELSE 0 E
ND) AS higher
    FROM Numbers n
    JOIN Numbers n1
    GROUP BY n.Number) cte
WHERE lower + same > higher AND same + higher > lower
),


(
SELECT AVG(Number)
FROM
    (SELECT n.Number,
            SUM(CASE WHEN n1.Number < n.Number THEN n1.Frequency ELSE 0 E
ND) AS lower,
            SUM(CASE WHEN n1.Number = n.Number THEN n1.Frequency ELSE 0 E
ND) AS same,
            SUM(CASE WHEN n1.Number > n.Number THEN n1.Frequency ELSE 0 E
ND) AS higher
    FROM Numbers n
    JOIN Numbers n1
    GROUP BY n.Number) cte
WHERE lower + same >= higher  AND same + higher >= lower
)

) as median
```

Q574

```
SELECT Name
FROM
    (SELECT CandidateId, COUNT(*) AS votes
    FROM Vote
    GROUP BY CandidateId
    ORDER BY votes DESC
    LIMIT 1) cte
INNER JOIN Candidate c
ON cte.CandidateId = c.id
```

Q577

```
SELECT name,bonus
FROM Employee
LEFT JOIN Bonus
USING(empId)
WHERE bonus < 1000 OR bonus IS NULL
```

Q578

```
SELECT question_id AS survey_log
FROM
    (SELECT question_id, 2*COUNT(answer_id)/COUNT(*) AS answer_rate
    FROM survey_log
    GROUP BY question_id
    ORDER BY answer_rate DESC
    LIMIT 1) cte
```

Q579

```
SELECT e1.Id AS id, e1.Month AS month, SUM(e2.Salary) AS Salary
FROM Employee e1
INNER JOIN Employee e2
WHERE (e1.Id = e2.Id) AND (e2.Month BETWEEN e1.Month-2 AND e1.Month) AND
(e1.Month < (SELECT MAX(e3.Month) FROM Employee e3 WHERE e1.Id = e3.Id))
GROUP BY e1.Id, e1.Month
ORDER BY e1.Id, e1.Month DESC
```

Q580

```
SELECT dept_name,
    (SELECT COUNT(*) FROM student s WHERE s.dept_id = d.dept_id) AS stude
nt_number
FROM department d
ORDER BY student_number DESC,dept_name
```

Q584

```sql
SELECT name
FROM customer
WHERE (referee_id IS NULL) OR (referee_id <> '2')
```

Q585

```sql
SELECT SUM(TIV_2016) AS TIV_2016
FROM insurance i
INNER JOIN
(SELECT DISTINCT i3.PID
FROM insurance i3
INNER JOIN insurance i4
USING(TIV_2015)
WHERE i3.PID <> i4.PID
    AND i3.PID NOT IN
    (SELECT DISTINCT i1.PID
    FROM insurance i1
    INNER JOIN insurance i2
    USING(LAT,LON)
    WHERE i1.PID <> i2.PID)) cte
USING(PID)
```

Q586

```sql
SELECT customer_number
FROM orders
GROUP BY customer_number
ORDER BY COUNT(*) DESC
LIMIT 1
```

Q595

```sql
SELECT name,population,area
FROM World
WHERE area > 3000000
    OR population > 25000000
```

Q596

```sql
SELECT class
FROM courses
GROUP BY class
HAVING COUNT(DISTINCT student) >= 5
```

Q597

```
SELECT ROUND(IFNULL((
(SELECT COUNT(DISTINCT requester_id, accepter_id) FROM request_accepted)
/
(SELECT COUNT(DISTINCT sender_id, send_to_id) FROM friend_request)
),0),2) AS accept_rate
```

Q601

```
SELECT * FROM
(
SELECT s1.*
FROM stadium s1
INNER JOIN stadium s2
ON s1.id + 1 = s2.id
INNER JOIN stadium s3
ON s1.id + 2 = s3.id
WHERE s1.people >= 100 AND s2.people >= 100 AND s3.people >= 100
UNION
SELECT s2.*
FROM stadium s1
INNER JOIN stadium s2
ON s1.id + 1 = s2.id
INNER JOIN stadium s3
ON s1.id + 2 = s3.id
WHERE s1.people >= 100 AND s2.people >= 100 AND s3.people >= 100
UNION
SELECT s3.*
FROM stadium s1
INNER JOIN stadium s2
ON s1.id + 1 = s2.id
INNER JOIN stadium s3
ON s1.id + 2 = s3.id
WHERE s1.people >= 100 AND s2.people >= 100 AND s3.people >= 100
) cte
ORDER BY id
```

Q602

```
SELECT id, COUNT(*) AS num
FROM
    (SELECT requester_id AS id
    FROM request_accepted
    UNION ALL
    SELECT accepter_id AS id
    FROM request_accepted) cte
GROUP BY id
ORDER BY COUNT(*) DESC
LIMIT 1
```

Q603

```
SELECT seat_id
FROM
    (SELECT c1.seat_id
    FROM cinema c1
    INNER JOIN cinema c2
    ON c1.seat_id + 1 = c2.seat_id AND c1.free = '1' AND c2.free = '1'

    UNION

    SELECT c2.seat_id
    FROM cinema c1
    INNER JOIN cinema c2
    ON c1.seat_id + 1 = c2.seat_id AND c1.free = '1' AND c2.free = '1') c
te
ORDER BY seat_id
```

Q607

```
SELECT name
FROM salesperson
LEFT JOIN
(SELECT DISTINCT sales_id
FROM orders
INNER JOIN company
USING(com_id)
WHERE name = 'RED') cte
USING(sales_id)
WHERE cte.sales_id IS NULL
```

Q608

```
SELECT t.id,
        CASE WHEN COUNT(p.id) = 0 THEN 'Root'
            WHEN COUNT(k.id) = 0 THEN 'Leaf'
            ELSE 'Inner' END AS Type
FROM tree t
LEFT JOIN tree p
ON t.p_id = p.id
LEFT JOIN tree k
ON k.p_id = t.id
GROUP BY t.id
```

Q610

```
SELECT *,
        CASE WHEN x + y <= z THEN 'No'
            WHEN x + z <= y THEN 'No'
```

```
            WHEN y + z <= x THEN 'No'
            ELSE 'Yes' END AS triangle
FROM triangle
```

Q612

```
SELECT ROUND(SQRT(POWER(p1.x-p2.x,2) + POWER(p1.y-p2.y,2)),2) AS shortest
FROM po_2d p1, po_2d p2
WHERE p1.x <> p2.x OR p1.y <> p2.y
ORDER BY shortest
LIMIT 1
```

Q613

```
SELECT MIN(ABS(p1.x - p2.x)) AS shortest
FROM po p1, po p2
WHERE p1.x <> p2.x
```

Q614

```
SELECT f1.follower, COUNT(DISTINCT f2.follower) AS num
FROM follow f1
INNER JOIN follow f2
ON f1.follower = f2.followee
GROUP BY f1.follower
ORDER BY f1.follower
```

Q615

```
SELECT dep.pay_month, department_id,
    CASE WHEN dep_avg > comp_avg THEN 'higher'
        WHEN dep_avg = comp_avg THEN 'same'
        ELSE 'lower' END AS comparison
FROM
(SELECT SUBSTRING(pay_date,1,7) AS pay_month, department_id, AVG(amount)
AS dep_avg
FROM salary
INNER JOIN employee
USING(employee_id)
GROUP BY pay_month, department_id) dep
INNER JOIN
(SELECT SUBSTRING(pay_date,1,7) AS pay_month, AVG(amount) AS comp_avg
FROM salary
GROUP BY pay_month) comp
USING(pay_month)
```

Q618

```sql
SELECT America, Asia, Europe
FROM
    (SELECT @row := @row + 1 AS id, name AS America
    FROM (SELECT @row := 0) s1, student
    WHERE continent = 'America'
    ORDER BY America) america
LEFT JOIN
    (SELECT @row1 := @row1 + 1 AS id, name AS Asia
    FROM (SELECT @row1 := 0) s1, student
    WHERE continent = 'Asia'
    ORDER BY Asia) asia
USING(id)
LEFT JOIN
    (SELECT @row2 := @row2 + 1 AS id, name AS Europe
    FROM (SELECT @row2 := 0) s1, student
    WHERE continent = 'Europe'
    ORDER BY Europe) europe
USING(id)
```

Q619

```sql
SELECT IFNULL(
    (SELECT num
    FROM my_numbers
    GROUP BY num
    HAVING COUNT(num) = 1
    ORDER BY num DESC
    LIMIT 1),
NULL) AS num
```

Q620

```sql
SELECT *
FROM cinema
WHERE description != 'boring' AND id % 2 = 1
ORDER BY rating DESC
```

Q626

```sql
SELECT s1.id,
    CASE WHEN s1.id % 2 = 0 THEN s3.student
        WHEN s1.id != (SELECT MAX(id) FROM seat) THEN s2.student
        ELSE s1.student END AS student
FROM seat s1
LEFT JOIN seat s2
ON s1.id + 1 = s2.id
LEFT JOIN seat s3
ON s1.id - 1 = s3.id
ORDER BY s1.id
```

```

```

Q627

```

```

Q1045

```
SELECT customer_id
FROM Customer
INNER JOIN Product
USING(product_key)
GROUP BY customer_id
HAVING COUNT(DISTINCT product_key) = (SELECT COUNT(DISTINCT product_key)
FROM Product)
```

Q1050

```
SELECT actor_id, director_id
FROM ActorDirector
GROUP BY actor_id, director_id
HAVING COUNT(timestamp) >= 3
```

Q1068

```
SELECT product_name, year, price
FROM Sales
LEFT JOIN Product
USING(product_id)
```

Q1069

```
SELECT product_id, SUM(quantity) as total_quantity
FROM Sales
GROUP BY product_id
```

Q1070

```
SELECT s.product_id, first_year, quantity, price
FROM Sales s
INNER JOIN
    (SELECT product_id, MIN(year) AS first_year
    FROM Sales
    GROUP BY product_id) cte
WHERE s.product_id = cte.product_id AND s.year = cte.first_year
```

Q1075

```sql
SELECT project_id, ROUND(AVG(experience_years),2) AS average_years
FROM Project
LEFT JOIN Employee
USING(employee_id)
GROUP BY project_id
```

Q1076

```sql
SELECT project_id
FROM Project
GROUP BY project_id
HAVING COUNT(DISTINCT employee_id) = (SELECT COUNT(DISTINCT employee_id)
AS count
                                     FROM Project
                                     GROUP BY project_id
                                     ORDER BY count DESC
                                     LIMIT 1)
```

Q1077

```sql
SELECT p.project_id, p.employee_id
FROM Project p
LEFT JOIN Employee e
USING(employee_id)
LEFT JOIN
    (SELECT project_id, MAX(experience_years) AS max_years
    FROM Project
    LEFT JOIN Employee
    USING(employee_id)
    GROUP BY project_id) cte
USING(project_id)
WHERE e.experience_years = cte.max_years
```

Q1082

```sql
SELECT seller_id
FROM Sales
GROUP BY seller_id
HAVING SUM(price) =
    (SELECT SUM(price) AS total_price
    FROM Sales
    GROUP BY seller_id
    ORDER BY total_price DESC
    LIMIT 1)
```

Q1083

```
SELECT DISTINCT cte1.buyer_id
FROM
    (SELECT buyer_id
    FROM Sales
    INNER JOIN Product
    USING(product_id)
    WHERE product_name = 'S8') cte1
LEFT JOIN
    (SELECT buyer_id
    FROM Sales
    INNER JOIN Product
    USING(product_id)
    WHERE product_name = 'iPhone') cte2
USING(buyer_id)
WHERE cte2.buyer_id IS NULL
```

Q1084

```
SELECT cte1.product_id, product_name
FROM
    (SELECT DISTINCT product_id
    FROM Sales
    WHERE sale_date BETWEEN '2019-01-01' AND '2019-03-31') cte1
LEFT JOIN
    (SELECT DISTINCT product_id
    FROM Sales
    WHERE NOT (sale_date BETWEEN '2019-01-01' AND '2019-03-31')) cte2
USING(product_id)
LEFT JOIN Product
USING(product_id)
WHERE cte2.product_id IS NULL
```

Q1097

```
SELECT install_dt, COUNT(ist.player_id) AS installs,
        ROUND(COUNT(a2.player_id)/COUNT(a1.player_id),2) as Day1_retentio
n
FROM Activity a1
INNER JOIN
    (SELECT player_id, MIN(event_date) AS install_dt
    FROM Activity
    GROUP BY player_id) ist
ON a1.player_id = ist.player_id AND a1.event_date = ist.install_dt
LEFT JOIN Activity a2
ON DATEDIFF(a2.event_date,a1.event_date) = 1 AND a2.player_id = a1.player
_id
GROUP BY install_dt
```

Q1098

```
SELECT b.book_id,name
FROM Books b
LEFT JOIN
    (SELECT book_id, SUM(quantity) AS total_sale
    FROM Orders
    WHERE dispatch_date > '2018-06-23'
    GROUP BY book_id) cte
USING(book_id)
WHERE available_from < '2019-05-23' AND (total_sale < 10 OR total_sale IS
 NULL)
```

Q1107

```
SELECT login_date, COUNT(DISTINCT t.user_id) AS user_count
FROM Traffic t
INNER JOIN
    (SELECT user_id, MIN(activity_date) AS login_date
    FROM Traffic
    WHERE activity = 'login'
    GROUP BY user_id) cte
ON t.user_id = cte.user_id AND t.activity_date = cte.login_date
WHERE t.activity = 'login' AND DATEDIFF('2019-06-30',activity_date) <= 90
GROUP BY login_date
```

Q1112

```
SELECT student_id,
        (SELECT MIN(course_id) FROM Enrollments e
        WHERE e.student_id = cte.student_id AND e.grade = cte.grade) AS c
ourse_id,
        grade
FROM
    (SELECT student_id, MAX(grade) AS grade
    FROM Enrollments
    GROUP BY student_id) cte
```

Q1113

```
SELECT extra AS report_reason, COUNT(DISTINCT post_id) AS report_count
FROM Actions
WHERE action_date = '2019-07-04' AND action = 'report'
GROUP BY extra
```

Q1126

```
SELECT business_id
```

```
FROM Events
LEFT JOIN
    (SELECT event_type, AVG(occurences) AS avg_oc
    FROM Events
    GROUP BY event_type) cte
USING(event_type)
WHERE occurences > avg_oc
GROUP BY business_id
HAVING COUNT(*) > 1
```

Q1127

```
SELECT  base.spend_date, base.platform,
        IFNULL(total_amount,0) AS total_amount,
        IFNULL(total_users,0) AS total_users
FROM
        (SELECT DISTINCT(spend_date), 'desktop' platform FROM Spending
        UNION
        SELECT DISTINCT(spend_date), 'mobile' platform FROM Spending
        UNION
        SELECT DISTINCT(spend_date), 'both' platform FROM Spending) base
LEFT JOIN
        (SELECT  spend_date, platform,
                SUM(amount) AS total_amount,
                COUNT(DISTINCT user_id) AS total_users
        FROM
            (SELECT s.user_id,
                    s.spend_date,
                    IF(bt.user_id IS NULL, s.platform, 'both') AS platfor
m,
                    s.amount
            FROM
                Spending s
            LEFT JOIN
                (SELECT user_id,spend_date
                FROM Spending
                GROUP BY user_id, spend_date
                HAVING COUNT(DISTINCT platform) = 2) bt
            USING(user_id, spend_date)
            ) cte
        GROUP BY spend_date, platform) cte
USING(spend_date,platform)
```

Q1132

```
SELECT ROUND(AVG(rate),2) AS average_daily_percent
FROM
    (SELECT action_date,
    ROUND(100* COUNT(DISTINCT r.post_id) / COUNT(DISTINCT a.post_id), 2)
AS rate
```

```
    FROM Actions a
    LEFT JOIN Removals r
    USING(post_id)
    WHERE action = 'report' AND extra = 'spam'
    GROUP BY action_date) cte
```

Q1141

```
SELECT activity_date AS day, COUNT(DISTINCT user_id) AS active_users
FROM Activity
WHERE DATEDIFF('2019-07-27',activity_date) BETWEEN 0 AND 29
GROUP BY activity_date
```

Q1142

```
SELECT IFNULL((SELECT ROUND(COUNT(DISTINCT session_id)/COUNT(DISTINCT use
r_id),2)
FROM Activity
WHERE DATEDIFF('2019-07-27',activity_date) BETWEEN 0 AND 29),0) AS averag
e_sessions_per_user
```

Q1148

```
SELECT DISTINCT author_id AS id
FROM Views
WHERE author_id = viewer_id
ORDER BY author_id
```

Q1149

```
SELECT DISTINCT viewer_id AS id
FROM Views
GROUP BY viewer_id, view_date
HAVING COUNT(DISTINCT article_id) > 1
```

Q1158

```
SELECT user_id AS buyer_id, join_date, COUNT(DISTINCT order_id) AS orders
_in_2019
FROM Users u
LEFT JOIN Orders o
ON u.user_id = o.buyer_id AND SUBSTRING(order_date,1,4) = '2019'
GROUP BY user_id
```

Q1159

```
SELECT u.user_id AS seller_id,
        CASE WHEN cte.id IS NULL THEN 'no'
            ELSE 'yes' END AS '2nd_item_fav_brand'
FROM Users u
LEFT JOIN
    (SELECT seller_id AS id
    FROM Orders
    LEFT JOIN
        (SELECT cur.order_id, TRUE as 2nd
        FROM Orders cur
        INNER JOIN Orders prv
        ON cur.seller_id = prv.seller_id AND cur.order_date > prv.order_d
ate
        GROUP BY cur.seller_id, cur.order_date
        HAVING COUNT(DISTINCT prv.order_id) = 1) second
    USING(order_id)
    LEFT JOIN Users u0
    ON u0.user_id = Orders.seller_id
    LEFT JOIN Items
    USING(item_id)
    WHERE 2nd = TRUE AND favorite_brand = item_brand) cte
ON u.user_id = cte.id
ORDER BY seller_id
```

Q1164

```
SELECT cte2.product_id,
    CASE WHEN cte1.new_price IS NULL THEN 10
        ELSE cte1.new_price END AS price
FROM
    (SELECT DISTINCT product_id
    FROM Products) cte2
LEFT JOIN
    (SELECT cte.product_id, new_price
    FROM Products
    INNER JOIN
        (SELECT product_id, MAX(change_date) AS change_date
        FROM Products
        WHERE change_date <= '2019-08-16'
        GROUP BY product_id) cte
    USING(product_id,change_date))cte1
USING(product_id)
```

Q1173

```
SELECT ROUND(100*AVG(order_date = customer_pref_delivery_date),2) AS imme
diate_percentage
FROM Delivery
```

Q1174

```
SELECT ROUND(100*AVG(d.order_date = customer_pref_delivery_date),2) AS im
mediate_percentage
FROM Delivery d
INNER JOIN
    (SELECT customer_id, MIN(order_date) AS order_date
    FROM Delivery
    GROUP BY customer_id) first_order
USING(customer_id,order_date)
```

Q1179

```
SELECT
    id,
    SUM(IF(month = 'Jan', revenue, null)) AS Jan_Revenue,
    SUM(IF(month = 'Feb', revenue, null)) AS Feb_Revenue,
    SUM(IF(month = 'Mar', revenue, null)) AS Mar_Revenue,
    SUM(IF(month = 'Apr', revenue, null)) AS Apr_Revenue,
    SUM(IF(month = 'May', revenue, null)) AS May_Revenue,
    SUM(IF(month = 'Jun', revenue, null)) AS Jun_Revenue,
    SUM(IF(month = 'Jul', revenue, null)) AS Jul_Revenue,
    SUM(IF(month = 'Aug', revenue, null)) AS Aug_Revenue,
    SUM(IF(month = 'Sep', revenue, null)) AS Sep_Revenue,
    SUM(IF(month = 'Oct', revenue, null)) AS Oct_Revenue,
    SUM(IF(month = 'Nov', revenue, null)) AS Nov_Revenue,
    SUM(IF(month = 'Dec', revenue, null)) AS Dec_Revenue
FROM Department
GROUP BY id
```

Q1193

```
SELECT SUBSTRING(trans_date,1,7) AS month,
        country,
        COUNT(DISTINCT id) AS trans_count,
        SUM(state = 'approved') AS approved_count,
        SUM(amount) AS trans_total_amount,
        SUM((state = 'approved')*amount) AS approved_total_amount
FROM Transactions
GROUP BY month, country
```

Q1194

```
SELECT GROUP_ID, MIN(PLAYER_ID) AS PLAYER_ID
FROM Players
LEFT JOIN
    (SELECT cte1.player_id, COUNT(DISTINCT cte2.player_id) +1 AS rank
     FROM
        (SELECT player_id, group_id,
```

```
            IFNULL(
                (SELECT SUM(first_score)
                    FROM Matches
                    WHERE first_player = player_id),
                    0)
            +
            IFNULL(
                (SELECT SUM(second_score)
                    FROM Matches
                    WHERE second_player = player_id),
                    0) AS total_score
        FROM Players) cte1
    LEFT JOIN
        (SELECT player_id, group_id,
            IFNULL(
                (SELECT SUM(first_score)
                    FROM Matches
                    WHERE first_player = player_id),
                    0)
            +
            IFNULL(
                (SELECT SUM(second_score)
                    FROM Matches
                    WHERE second_player = player_id),
                    0) AS total_score
        FROM Players) cte2
    ON (cte2.group_id = cte1.group_id AND cte1.total_score < cte2.total_s
core)
    GROUP BY cte1.player_id) cte
USING(player_id)
WHERE rank = 1
GROUP BY group_id
```

Q1204

```
SELECT q1.person_name
FROM Queue q1
LEFT JOIN Queue q2
ON q2.turn <= q1.turn
GROUP BY q1.turn
HAVING SUM(q2.weight) <= 1000
ORDER BY q1.turn DESC
LIMIT 1
```

Q1205

```
SELECT base.month, base.country,
        IFNULL(approved_count,0) AS approved_count,
        IFNULL(approved_amount,0) AS approved_amount,
        IFNULL(chargeback_count,0) AS chargeback_count,
```

```
        IFNULL(chargeback_amount,0) AS chargeback_amount
FROM
    (SELECT SUBSTRING(trans_date,1,7) AS month, country
    FROM Transactions
    GROUP BY month, country
    UNION
    SELECT SUBSTRING(c.trans_date,1,7) AS month,country
    FROM Chargebacks c
    INNER JOIN Transactions t
    ON c.trans_id = t.id
    GROUP BY month, country) base
LEFT JOIN
    (SELECT  SUBSTRING(trans_date,1,7) AS month,
            country,
            SUM(state = 'approved') AS approved_count,
            SUM((state = 'approved')*amount) AS approved_amount
    FROM Transactions
    GROUP BY month, country) cte1
USING(month,country)
LEFT JOIN
    (SELECT  SUBSTRING(c.trans_date,1,7) AS month,
            country,
            COUNT(trans_id) AS chargeback_count,
            SUM(IF(c.trans_date IS NULL,0,amount)) AS chargeback_amount
    FROM Transactions t
    LEFT JOIN Chargebacks c
    ON t.id = c.trans_id
    GROUP BY month,country) cte2
USING(month,country)
WHERE approved_count != 0 OR approved_amount != 0 OR chargeback_count !=
0 OR chargeback_amount != 0
```

Q1211

```
SELECT query_name,
        ROUND(AVG(rating/position),2) AS quality,
        ROUND(100*AVG(rating < 3),2) AS poor_query_percentage
FROM Queries
GROUP BY query_name
```

Q1212

```
SELECT t.team_id, team_name, IFNULL(home_po,0) + IFNULL(away_po,0) AS num
_pos
FROM Teams t
LEFT JOIN
    (SELECT host_team AS team_id,
            SUM(CASE WHEN host_goals > guest_goals THEN 3
                WHEN host_goals = guest_goals THEN 1
                ELSE 0 END) AS home_po
```

```
    FROM Matches
    GROUP BY host_team) home
USING(team_id)
LEFT JOIN
    (SELECT guest_team AS team_id,
            SUM(CASE WHEN host_goals > guest_goals THEN 0
                WHEN host_goals = guest_goals THEN 1
                ELSE 3 END) AS away_po
    FROM Matches
    GROUP BY guest_team) away
USING(team_id)
ORDER BY num_pos DESC, team_id
```

Q1225

```
SELECT period_state,
        IF(start_date > '2019-01-01', start_date, '2019-01-01') AS start_
date,               IF(end_date < '2019-12-31', end_date, '2019-12-31') A
S end_date
FROM
    (SELECT  'succeeded' AS period_state,
            success_start.success_date AS start_date,
            MIN(success_end.success_date) AS end_date
    FROM
        (SELECT success_date
        FROM Succeeded
        WHERE ADDDATE(success_date,-1) NOT IN (SELECT * FROM Succeeded))
 success_start,
        (SELECT success_date
        FROM Succeeded
        WHERE ADDDATE(success_date,1) NOT IN (SELECT * FROM Succeeded)) s
uccess_end
    WHERE success_end.success_date >= success_start.success_date
    GROUP BY start_date
    UNION
    SELECT  'failed' AS period_state,
            fail_start.fail_date AS start_date,
            MIN(fail_end.fail_date) AS end_date
    FROM
        (SELECT fail_date
        FROM Failed
        WHERE ADDDATE(fail_date,-1) NOT IN (SELECT * FROM Failed)) fail_s
tart,
        (SELECT fail_date
        FROM Failed
        WHERE ADDDATE(fail_date,1) NOT IN (SELECT * FROM Failed)) fail_en
d
    WHERE fail_end.fail_date >= fail_start.fail_date
    GROUP BY start_date) cte
WHERE end_date >= '2019-01-01' AND start_date <= '2019-12-31'
ORDER BY start_date
```

Q1241

```
SELECT post.sub_id AS post_id, COUNT(DISTINCT c.sub_id) AS number_of_comm
ents
FROM
    (SELECT DISTINCT sub_id
    FROM Submissions
    WHERE parent_id IS NULL) post
LEFT JOIN
    Submissions c
ON post.sub_id = c.parent_id
GROUP BY post.sub_id
```

Q1251

```
SELECT u.product_id, ROUND(SUM(units*price)/SUM(units),2) AS average_pric
e
FROM UnitsSold u
LEFT JOIN Prices p
ON  u.product_id = p.product_id AND
    u.purchase_date BETWEEN p.start_date AND p.end_date
GROUP BY product_id
```

Q1264

```
SELECT DISTINCT l.page_id AS recommended_page
FROM
    (SELECT user2_id AS user_id
    FROM Friendship
    WHERE user1_id = 1
    UNION
    SELECT user1_id AS user_id
    FROM Friendship
    WHERE user2_id = 1) cte
INNER JOIN Likes l
USING(user_id)
LEFT JOIN
    (SELECT DISTINCT page_id
    FROM Likes
    WHERE user_id = 1) cte2
USING(page_id)
WHERE cte2.page_id IS NULL
```

Q1270

```
SELECT *
FROM
```

```
    (SELECT employee_id
    FROM Employees
    WHERE manager_id = 1

    UNION

    SELECT e.employee_id
    FROM Employees e
    INNER JOIN
        (SELECT employee_id
        FROM Employees
        WHERE manager_id = 1) level_1
    ON e.manager_id = level_1.employee_id

    UNION

    SELECT e.employee_id
    FROM Employees e
    INNER JOIN
        (SELECT e.employee_id
        FROM Employees e
        INNER JOIN
            (SELECT employee_id
            FROM Employees
            WHERE manager_id = 1) level_1
        ON e.manager_id = level_1.employee_id) level_2
    ON e.manager_id = level_2.employee_id) cte
WHERE employee_id != 1
```

Q1280

```
SELECT s.student_id,student_name,s1.subject_name, COUNT(e.subject_name) A
S attended_exams
FROM Students s
JOIN Subjects s1
LEFT JOIN Examinations e
USING(student_id,subject_name)
GROUP BY s.student_id, s1.subject_name
```

Q1285

```
SELECT start_id, MIN(end_id) AS end_id
FROM
    (SELECT log_id AS start_id
    FROM Logs
    WHERE log_id - 1 NOT IN (SELECT * FROM Logs)) start
LEFT JOIN
    (SELECT log_id AS end_id
    FROM Logs
    WHERE log_id + 1 NOT IN (SELECT * FROM Logs)) end
```

```
ON end.end_id >= start.start_id
GROUP BY start_id
```

Q1294

```
SELECT  country_name,
        CASE WHEN AVG(weather_state) <= 15 THEN 'Cold'
            WHEN AVG(weather_state) >= 25 THEN 'Hot'
            ELSE 'Warm' END AS weather_type
FROM Weather
INNER JOIN Countries
USING(country_id)
WHERE SUBSTRING(day,1,7) = '2019-11'
GROUP BY country_id
```