EASY：

595. Big Countries

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

613. Shortest Distance in a line

```
SELECT MIN(ABS(A.x-B.x)) AS shortest
FROM point A
JOIN point B
ON A.x != B.x
```

627. Swap Salary

```
UPDATE salary
SET sex=
CASE sex
    WHEN "f" THEN "m"
    ELSE "f"
END
```

584. Find Customer Referee

```
SELECT name
FROM customer
WHERE referee_id != 2 OR referee_id IS NULL
```

586. Customer Placing the Largest Number of Orders

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

620. Not Boring Movies

```
SELECT *
FROM cinema
WHERE MOD(id,2) = 1 AND description != 'boring'
ORDER BY rating DESC
```

610. Triangle Judgement

```
SELECT x,y,z,
CASE
WHEN ABS(x+y)>z AND ABS(x+z)>y AND ABS(z+y)>x AND ABS(x-y)<z AND ABS(x-
z)<y AND ABS(z-y)<x THEN "Yes"
ELSE "No"
END AS triangle
FROM triangle
```

603. Consecutive Available Seats

```
SELECT DISTINCT(A.seat_id) AS seat_id
FROM cinema A
JOIN cinema B
ON ABS(B.seat_id - A.seat_id) = 1 AND (A.free = TRUE AND B.free = TRUE)
ORDER BY seat_id ASC
```

577. Employee Bonus

```
SELECT A.name AS name, B.bonus AS bonus
FROM Employee A LEFT JOIN Bonus B
ON A.empId = B.empId
WHERE B.bonus < 1000 OR B.bonus IS NULL
```

607. Sales Person

```
SELECT S.name
FROM salesperson S
WHERE S.sales_id NOT IN(
  SELECT O.sales_id FROM orders O
    LEFT JOIN company C
    ON O.com_id = C.com_id
    WHERE C.name = 'RED'
)
```

182. Duplicate Emails

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

175. Combine Two Tables

```
SELECT P.FirstName, P.LastName, A.City, A.State
FROM Person P
LEFT JOIN Address A
ON P.PersonId = A.PersonId
```

181. Employees Earning More Than Their Managers

```
SELECT A.Name AS Employee
FROM Employee A
LEFT JOIN Employee B
ON B.Id = A.ManagerId
WHERE B.Salary < A.Salary
```

183. Customers Who Never Order

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

597. Friend Requests I: Overall Acceptance Rate

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

619. Biggest Single Number

```
SELECT MAX(num) AS num
FROM
(SELECT num
FROM number
GROUP BY num
HAVING COUNT(num) = 1
ORDER BY num DESC) B
```

596. Classes More Than 5 Students

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

197. Rising Temperature

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

OR

```
SELECT B.Id
FROM Weather A JOIN Weather B
ON A.RecordDate != B.RecordDate
WHERE TIMESTAMPDIFF(day,A.RecordDate,B.RecordDate) = 1 AND A.Temperature <
B.Temperature
```

196. Delete Duplicate Emails

```
DELETE P2
FROM Person P1 LEFT JOIN Person P2
ON P1.Email = P2.Email AND P1.Id < P2.Id
```

176. Second Highest Salary

```
SELECT MAX(Salary) AS SecondHighestSalary
FROM Employee E3
WHERE Salary NOT IN
(SELECT E1.Salary
FROM Employee E1 LEFT JOIN Employee E2
ON E1.Salary < E2.Salary
WHERE E2.Id IS NULL)
```

570. Managers with at Least 5 Direct Reports

```
SELECT Name
FROM Employee
```

```
WHERE ID IN(
   SELECT ManagerId
   FROM Employee
   GROUP BY ManagerId
   HAVING COUNT(ManagerId) >= 5)
```

608. Tree Node

```
SELECT DISTINCT C.Dad AS Id,
CASE WHEN C.Grandpa IS NULL THEN "Root"
WHEN C.Son IS NULL THEN "Leaf"
ELSE "Inner"
END AS Type
FROM(
   SELECT A.p_id AS Grandpa, A.id Dad, B.id Son
   FROM tree A LEFT JOIN tree B
   ON A.id = B.p_id
) C
```

612. Shortest Distance in a Plane

```
SELECT
   MIN(ROUND(SQRT(POW(p1.x - p2.x, 2) + POW(p1.y - p2.y, 2)),2)) AS shortest
FROM
   point_2d P1
      JOIN
   point_2d P2 ON P1.x != P2.x OR P1.y != P2.y;
```

626. Exchange Seats

? ? ?

585. Investments in 2016

```
SELECT SUM(TIV_2016) AS TIV_2016
FROM insurance
WHERE TIV_2015 IN(
   SELECT TIV_2015
   FROM insurance
   GROUP BY TIV_2015
   HAVING COUNT(TIV_2015) > 1
) AND CONCAT(LAT,LON) IN(
        SELECT CONCAT(LAT,LON)
        FROM insurance
        GROUP BY CONCAT(LAT,LON)
        HAVING COUNT(CONCAT(LAT,LON)) = 1)
```

602. Friend Requests II: Who Has the Most Friends

```
SELECT A.ID AS id, COUNT(A.ID) AS num
FROM(
    (SELECT requester_id AS ID
    FROM request_accepted)
    UNION ALL
    (SELECT accepter_id AS ID
     FROM request_accepted)
    ) A
GROUP BY A.ID
ORDER BY COUNT(A.ID) DESC
LIMIT 1;
```

580. Count Student Number in Departments

```
SELECT B.dept_name AS dept_name, SUM(B.count) AS student_number
FROM
   (SELECT A.D_dept_name AS dept_name,
    CASE
    WHEN A.S_stu IS NOT NULL THEN 1
    WHEN A.S_stu IS NULL THEN 0
    END AS count
    FROM(
        SELECT D.dept_id AS D_id, D.dept_name AS D_dept_name, S.student_id AS
S_stu,S.student_name,S.gender,S.dept_id AS S_id
        FROM department D LEFT JOIN student S
        ON D.dept_id = S.dept_id) A) B
GROUP BY dept_name
ORDER BY student_number DESC, dept_name ASC
```

574. Winning Candidate

```
SELECT Name
FROM Candidate
WHERE id = (
    SELECT CandidateId AS id
    FROM Vote
    GROUP BY CandidateId
    ORDER BY COUNT(CandidateId) DESC
    LIMIT 1
)
```

178. Rank Scores

```
SELECT S.Score AS Score,C.row_number AS Rank
FROM Scores S
LEFT JOIN
(
    SELECT A.Score, @curRow := @curRow + 1 AS row_number
    FROM(
        SELECT Score
        FROM Scores
        GROUP BY Score
        ORDER BY Score DESC
         ) A
        JOIN (SELECT @curRow := 0) B
) C
ON S.Score = C.Score
ORDER BY S.Score DESC
```

578. Get Highest Answer Rate Question

```
SELECT A.question_id AS survey_log
FROM(
    SELECT question_id,
    SUM(CASE WHEN action = "show" THEN 1 ELSE 0 END) AS show_num,
    SUM(CASE WHEN action = "answer" THEN 1 ELSE 0 END) AS answer_num
    FROM survey_log
    GROUP BY question_id
) A
ORDER BY A.answer_num/A.show_num DESC
LIMIT 1
```

180. Consecutive Numbers

```
SELECT DISTINCT(D.Num1) AS ConsecutiveNums
FROM(
    SELECT A.Id AS Id1,A.Num AS Num1, B.Id AS Id2,B.Num AS Num2,C.Id AS Id3,C.Num
AS Num3
    FROM Logs A
    LEFT JOIN Logs B
    ON B.Id = A.Id + 1
    LEFT JOIN Logs C
    ON C.Id = B.Id + 1
) D
WHERE D.Num1 = D.Num2 AND D.Num2 = D.Num3
```

184. Department Highest Salary

```
SELECT D.Name AS Department, E.Name AS Employee,E.Salary AS Salary
FROM Department D
LEFT JOIN Employee E
ON D.Id = E.DepartmentId
WHERE (E.Salary,E.DepartmentId) IN
(
   SELECT MAX(E2.Salary),E2.DepartmentId
   FROM Employee E2
   GROUP BY E2.DepartmentId
)
```

177. Nth Highest Salary

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

618. Students Report By Geography

```
SELECT
   America, Asia, Europe
FROM
   (SELECT @as:=0, @am:=0, @eu:=0) t,
   (SELECT
       @as:=@as + 1 AS asid, name AS Asia
   FROM
       student
   WHERE
       continent = 'Asia'
   ORDER BY Asia) AS t1
       RIGHT JOIN
   (SELECT
       @am:=@am + 1 AS amid, name AS America
   FROM
       student
   WHERE
       continent = 'America'
   ORDER BY America) AS t2 ON asid = amid
       LEFT JOIN
   (SELECT
       @eu:=@eu + 1 AS euid, name AS Europe
   FROM
       student
   WHERE
       continent = 'Europe'
   ORDER BY Europe) AS t3 ON amid = euid
;
```