Easy

175

```
use leetcode;

drop table if exists person;
drop table if exists address;

Create table Person (
    PersonId int,
    FirstName varchar(255),
    LastName varchar(255));
Create table Address (
    AddressId int,
    PersonId int,
    City varchar(255),
    State varchar(255));

Truncate table Person;
insert into Person (PersonId, LastName, FirstName)
values ('1', 'Wang', 'Allen');

Truncate table Address;
insert into Address (AddressId, PersonId, City, State)
values ('1', '2', 'New York City', 'New York');


SELECT FirstName, LastName, City, State

FROM Person p

LEFT JOIN Address a

ON p.`PersonId` = a.`PersonId`
```

176

```
use leetcode;

drop table if exists Employee;
CREATE TABLE IF NOT EXISTS Employee (Id INT, Salary INT);
TRUNCATE TABLE Employee;
INSERT INTO Employee (Id, Salary) VALUES ('1', '100');
INSERT INTO Employee (Id, Salary) VALUES ('2', '200');
INSERT INTO Employee (Id, Salary) VALUES ('3', '300');

SELECT MAX(salary)
FROM Employee
WHERE salary < (SELECT MAX(salary)
               FROM Employee
               )
;


SELECT
(SELECT DISTINCT salary
FROM employee
ORDER BY salary DESC
LIMIT 1,1
) AS SecondHighestSalary
```

603

```
use leetcode;

drop table if exists cinema;
Create table If Not Exists cinema (
    seat_id int primary key auto_increment,
    free bool);
Truncate table cinema;

insert into cinema (seat_id, free) values ('1', '1');
insert into cinema (seat_id, free) values ('2', '0');
insert into cinema (seat_id, free) values ('3', '1');
insert into cinema (seat_id, free) values ('4', '1');
insert into cinema (seat_id, free) values ('5', '1');


SELECT DISTINCT a.seat_id

FROM cinema a

JOIN cinema b

ON ABS(a.`seat_id` - b.`seat_id`) = 1

AND a.`free` = 1

AND b.`free` = 1

;
```

595

```
Create table If Not Exists World (name varchar(255), continent varchar(255), area int, population int, gdp int);

Truncate table World;

insert into World (name, continent, area, population, gdp) values ('Afghanistan', 'Asia', '652230', '25500100', '20343000000');

insert into World (name, continent, area, population, gdp) values ('Albania', 'Europe', '28748', '2831741', '12960000000');

insert into World (name, continent, area, population, gdp) values ('Algeria', 'Africa', '2381741', '37100000', '188681000000');

insert into World (name, continent, area, population, gdp) values ('Andorra', 'Europe', '468', '78115', '3712000000');

insert into World (name, continent, area, population, gdp) values ('Angola', 'Africa', '1246700', '20609294', '100990000000');
```

```
SELECT NAME, population, AREA

FROM world

WHERE AREA > 3000000 OR population > 25000000
```

196

```
USE leetcode;

DROP TABLE IF EXISTS person;
CREATE TABLE person (id INT, email TEXT);
TRUNCATE TABLE person;

INSERT INTO person VALUES(1, 'john@example.com');
INSERT INTO person VALUES(2, 'bob@example.com');
INSERT INTO person VALUES(3, 'john@example.com');

DELETE a
FROM Person a
JOIN Person b
ON a.`email` = b.`email`
AND a.`id` > b.`id`
```

597

```
USE leetcode;

DROP TABLE IF EXISTS friend_request;
DROP TABLE IF EXISTS request_accepted;

CREATE TABLE IF NOT EXISTS friend_request (
 sender_id INT NOT NULL,
 send_to_id INT NULL,
 request_date DATE NULL);
CREATE TABLE IF NOT EXISTS request_accepted (
 requester_id INT NOT NULL,
 accepter_id INT NULL,
 accept_date DATE NULL);

TRUNCATE TABLE friend_request;
INSERT INTO friend_request (sender_id, send_to_id, request_date)
 VALUES ('1', '2', '2016/06/01');
INSERT INTO friend_request (sender_id, send_to_id, request_date)
 VALUES ('1', '3', '2016/06/01');
INSERT INTO friend_request (sender_id, send_to_id, request_date)
 VALUES ('1', '4', '2016/06/01');
INSERT INTO friend_request (sender_id, send_to_id, request_date)
 VALUES ('2', '3', '2016/06/02');
INSERT INTO friend_request (sender_id, send_to_id, request_date)
 VALUES ('3', '4', '2016/06/09');

TRUNCATE TABLE request_accepted;
INSERT INTO request_accepted (requester_id, accepter_id, accept_date)
 VALUES ('1', '2', '2016/06/03');
INSERT INTO request_accepted (requester_id, accepter_id, accept_date)
 VALUES ('1', '3', '2016/06/08');
```

```
INSERT INTO request_accepted (requester_id, accepter_id, accept_date)

 VALUES ('2', '3', '2016/06/08');

INSERT INTO request_accepted (requester_id, accepter_id, accept_date)

 VALUES ('3', '4', '2016/06/09');

INSERT INTO request_accepted (requester_id, accepter_id, accept_date)

 VALUES ('3', '4', '2016/06/10');


SELECT IFNULL(ROUND(COUNT(DISTINCT requester_id, accepter_id) / COUNT(DISTINCT
sender_id, send_to_id), 2),0)

            AS accept_rate

FROM friend_request, request_accepted
```

610

```
USE leetcode;


DROP TABLE IF EXISTS triangle;

CREATE TABLE IF NOT EXISTS triangle (X INT, Y INT, z INT);

TRUNCATE TABLE triangle;


INSERT INTO triangle (X, Y, z) VALUES ('13', '15', '30');

INSERT INTO triangle (X, Y, z) VALUES ('10', '20', '15');


SELECT X,Y,z,

CASE WHEN (X + Y > z) AND (X + z > Y) AND (Y + z > X) THEN 'Yes'

        ELSE 'No'


END AS 'triangle'

FROM triangle
```

181

```
use leetcode;

drop table if exists employee;
Create table If Not Exists Employee (
Id int,
Name varchar(255),
Salary int,
ManagerId int);

Truncate table Employee;

insert into Employee (Id, Name, Salary, ManagerId)
 values ('1', 'Joe', '70000', '3');
insert into Employee (Id, Name, Salary, ManagerId)
 values ('2', 'Henry', '80000', '4');
insert into Employee (Id, Name, Salary, ManagerId)
 values ('3', 'Sam', '60000', null);
insert into Employee (Id, Name, Salary, ManagerId)
 values ('4', 'Max', '90000', null);


SELECT a.Name AS Employee

FROM Employee a

JOIN Employee b

ON a.`ManagerId` = b.`Id`

AND a.`Salary` > b.`Salary`
```

182

```
use leetcode;

drop table if exists person;
Create table If Not Exists Person (Id int, Email varchar(255));
Truncate table Person;

insert into Person (Id, Email) values ('1', 'a@b.com');
insert into Person (Id, Email) values ('2', 'c@d.com');
insert into Person (Id, Email) values ('3', 'a@b.com');


SELECT Email

FROM Person

GROUP BY Email

HAVING COUNT(*) > 1
```

183

```
USE leetcode;

DROP TABLE IF EXISTS customers;
DROP TABLE IF EXISTS orders;

CREATE TABLE IF NOT EXISTS Customers (Id INT, NAME VARCHAR(255));
CREATE TABLE IF NOT EXISTS Orders (Id INT, CustomerId INT);

TRUNCATE TABLE Customers;
INSERT INTO Customers (Id, NAME) VALUES ('1', 'Joe');
INSERT INTO Customers (Id, NAME) VALUES ('2', 'Henry');
INSERT INTO Customers (Id, NAME) VALUES ('3', 'Sam');
INSERT INTO Customers (Id, NAME) VALUES ('4', 'Max');

TRUNCATE TABLE Orders;
INSERT INTO Orders (Id, CustomerId) VALUES ('1', '3');
INSERT INTO Orders (Id, CustomerId) VALUES ('2', '1');

SELECT NAME
FROM Customers c
LEFT JOIN Orders o
ON c.Id = o.`CustomerId`
WHERE o.Id IS NULL
```

197

```
use leetcode;

drop table if exists weather;
Create table If Not Exists Weather (
 Id int,
 RecordDate date,
 Temperature int);

Truncate table Weather;
insert into Weather (Id, RecordDate, Temperature)
 values ('1', '2015-01-01', '10');
insert into Weather (Id, RecordDate, Temperature)
 values ('2', '2015-01-02', '25');
insert into Weather (Id, RecordDate, Temperature)
 values ('3', '2015-01-03', '20');
insert into Weather (Id, RecordDate, Temperature)
 values ('4', '2015-01-04', '30');



SELECT b.Id

FROM Weather a, Weather b

WHERE b.`Temperature` > a.`Temperature`

AND DATEDIFF(b.`RecordDate`, a.`RecordDate`) = 1
```

577

```
use leetcode;

drop table if exists employee;
drop table if exists bonus;

Create table If Not Exists Employee (
    EmpId int,
    Name varchar(255),
    Supervisor int,
    Salary int);
Create table If Not Exists Bonus (
    EmpId int,
    Bonus int);

Truncate table Employee;
insert into Employee (EmpId, Name, Supervisor, Salary)
    values ('3', 'Brad', null, '4000');
insert into Employee (EmpId, Name, Supervisor, Salary)
    values ('1', 'John', '3', '1000');
insert into Employee (EmpId, Name, Supervisor, Salary)
    values ('2', 'Dan', '3', '2000');
insert into Employee (EmpId, Name, Supervisor, Salary)
    values ('4', 'Thomas', '3', '4000');

Truncate table Bonus;
insert into Bonus (EmpId, Bonus)
    values ('2', '500');
insert into Bonus (EmpId, Bonus)
    values ('4', '2000');


SELECT NAME, bonus

FROM Employee e

LEFT JOIN Bonus b

ON e.`EmpId` = b.`EmpId`

WHERE Bonus < 1000 OR Bonus IS NULL
```

584

```
use leetcode;

drop table if exists customer;
CREATE TABLE IF NOT EXISTS customer (
   id INT,
   name VARCHAR(25),
   referee_id INT);

Truncate table customer;
insert into customer (id, name, referee_id)
    values ('1', 'Will', null);
insert into customer (id, name, referee_id)
    values ('2', 'Jane', null);
insert into customer (id, name, referee_id)
    values ('3', 'Alex', '2');
insert into customer (id, name, referee_id)
    values ('4', 'Bill', null);
insert into customer (id, name, referee_id)
    values ('5', 'Zack', '1');
insert into customer (id, name, referee_id)
    values ('6', 'Mark', '2');


SELECT NAME

FROM customer

WHERE referee_id <> 2 OR referee_id IS NULL
```

586

```
drop table if exists orders;
Create table If Not Exists orders (
 order_number int,
 customer_number int,
 order_date date,
 required_date date,
 shipped_date date,
 status char(15),
 comment char(200),
 key(order_number));

Truncate table orders;
insert into orders (order_number, customer_number)
 values ('1', '1');
insert into orders (order_number, customer_number)
 values ('2', '2');
insert into orders (order_number, customer_number)
 values ('3', '3');
insert into orders (order_number, customer_number)
 values ('4', '3');

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

596

```
use leetcode;

drop table if exists courses;
Create table If Not Exists courses (
    student varchar(255),
    class varchar(255));
Truncate table courses;

insert into courses (student, class) values ('A', 'Math');
insert into courses (student, class) values ('B', 'English');
insert into courses (student, class) values ('C', 'Math');
insert into courses (student, class) values ('D', 'Biology');
insert into courses (student, class) values ('E', 'Math');
insert into courses (student, class) values ('F', 'Computer');
insert into courses (student, class) values ('G', 'Math');
insert into courses (student, class) values ('H', 'Math');
insert into courses (student, class) values ('I', 'Math');

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

607

```sql
DROP TABLE IF EXISTS salesperson;

DROP TABLE IF EXISTS company;

DROP TABLE IF EXISTS orders;


CREATE TABLE IF NOT EXISTS salesperson (sales_id INT, NAME VARCHAR(255), salary
INT,commission_rate INT, hire_date VARCHAR(255));

CREATE TABLE IF NOT EXISTS company (com_id INT, NAME VARCHAR(255), city
VARCHAR(255));

CREATE TABLE IF NOT EXISTS orders (order_id INT, DATE VARCHAR(255), com_id INT, sales_id
INT, amount INT);


TRUNCATE TABLE company; INSERT INTO company (com_id, NAME, city) VALUES ('1', 'RED',
'Boston'); INSERT INTO company (com_id, NAME, city) VALUES ('2', 'ORANGE', 'New York');
INSERT INTO company (com_id, NAME, city) VALUES ('3', 'YELLOW', 'Boston'); INSERT INTO
company (com_id, NAME, city) VALUES ('4', 'GREEN', 'Austin');

TRUNCATE TABLE salesperson; INSERT INTO salesperson (sales_id, NAME, salary, commission_rate,
hire_date) VALUES ('1', 'John', '100000', '6', '4/1/2006'); INSERT INTO salesperson (sales_id, NAME,
salary, commission_rate, hire_date) VALUES ('2', 'Amy', '12000', '5', '5/1/2010'); INSERT INTO
salesperson (sales_id, NAME, salary, commission_rate, hire_date) VALUES ('3', 'Mark', '65000', '12',
'12/25/2008'); INSERT INTO salesperson (sales_id, NAME, salary, commission_rate, hire_date) VALUES
('4', 'Pam', '25000', '25', '1/1/2005'); INSERT INTO salesperson (sales_id, NAME, salary, commission_rate,
hire_date) VALUES ('5', 'Alex', '5000', '10', '2/3/2007');

TRUNCATE TABLE orders; INSERT INTO orders (order_id, DATE, com_id, sales_id, amount)
VALUES ('1', '1/1/2014', '3', '4', '10000'); INSERT INTO orders (order_id, DATE, com_id, sales_id,
amount) VALUES ('2', '2/1/2014', '4', '5', '5000'); INSERT INTO orders (order_id, DATE, com_id,
sales_id, amount) VALUES ('3', '3/1/2014', '1', '1', '50000'); INSERT INTO orders (order_id, DATE,
com_id, sales_id, amount) VALUES ('4', '4/1/2014', '1', '4', '25000');


SELECT NAME

FROM salesperson

WHERE sales_id NOT IN (SELECT sales_id

                       FROM orders

                       WHERE com_id = (SELECT com_id

                                       FROM company

                                       WHERE NAME='RED'

                                      )
```

)

613

```
DROP TABLE IF EXISTS POINT;

CREATE TABLE IF NOT EXISTS POINT (X INT);

TRUNCATE TABLE POINT;

INSERT INTO POINT (X) VALUES (-1);

INSERT INTO POINT (X) VALUES (0);

INSERT INTO POINT (X) VALUES (2);


SELECT MIN(ABS(a.x - b.x)) shortest

FROM POINT a, POINT b

WHERE a.x <> b.x
```

619

```
DROP TABLE IF EXISTS number;

CREATE TABLE IF NOT EXISTS number (num INT);

TRUNCATE TABLE number;

INSERT INTO number (num) VALUES (8);

INSERT INTO number (num) VALUES (8);

INSERT INTO number (num) VALUES (3);

INSERT INTO number (num) VALUES (3);

INSERT INTO number (num) VALUES (1);

INSERT INTO number (num) VALUES (4);

INSERT INTO number (num) VALUES (5);

INSERT INTO number (num) VALUES (6);


SELECT (SELECT num

        FROM number

        GROUP BY num

        HAVING COUNT(*) = 1

        ORDER BY num DESC

        LIMIT 1

        ) num
```

620

```
use leetcode;

drop table if exists cinema;
Create table If Not Exists cinema (
    id int, movie varchar(255),
    description varchar(255),
    rating float(2, 1));

Truncate table cinema;

insert into cinema (id, movie, description, rating)
    values ('1', 'War', 'great 3D', '8.9');
insert into cinema (id, movie, description, rating)
    values ('2', 'Science', 'fiction', '8.5');
insert into cinema (id, movie, description, rating)
    values ('3', 'irish', 'boring', '6.2');
insert into cinema (id, movie, description, rating)
    values ('4', 'Ice song', 'Fantacy', '8.6');
insert into cinema (id, movie, description, rating)
    values ('5', 'House card', 'Interesting', '9.1');


SELECT *

FROM cinema

WHERE id%2 = 1

AND description <> 'boring'

ORDER BY rating DESC
```

627

```
use leetcode;

drop table if exists salary;
create table if not exists salary(id int, name varchar(100), sex char(1), salary int);
Truncate table salary;

insert into salary (id, name, sex, salary) values ('1', 'A', 'm', '2500');
insert into salary (id, name, sex, salary) values ('2', 'B', 'f', '1500');
insert into salary (id, name, sex, salary) values ('3', 'C', 'm', '5500');
insert into salary (id, name, sex, salary) values ('4', 'D', 'f', '500');



UPDATE salary

SET sex = (

        CASE sex

        WHEN 'm' THEN 'f'

        WHEN 'f' THEN 'm'

        END

        )
```

Medium

177

```
DROP TABLE IF EXISTS Employee;
CREATE TABLE IF NOT EXISTS Employee (Id INT, Salary INT);
TRUNCATE TABLE Employee;

INSERT INTO Employee VALUES (1,100);
INSERT INTO Employee VALUES (2,200);
INSERT INTO Employee VALUES (3,300);

CREATE FUNCTION getNthHighestSalary(N INT) RETURNS INT
BEGIN
    DECLARE M int;
    set M = N-1;
  RETURN (
      # Write your MySQL query statement below.
    select distinct e1.salary
      from employee e1
      where N-1 = (select count(distinct e2.salary)
                  from employee e2
                  where e1.salary < e2.salary)

  );
END
```

180

```
DROP TABLE IF EXISTS LOGS;
CREATE TABLE IF NOT EXISTS LOGS (Id INT, Num INT);
TRUNCATE TABLE LOGS;

INSERT INTO LOGS VALUES (1,1);
INSERT INTO LOGS VALUES (2,1);
INSERT INTO LOGS VALUES (3,1);
INSERT INTO LOGS VALUES (4,2);
INSERT INTO LOGS VALUES (5,1);
INSERT INTO LOGS VALUES (6,2);
INSERT INTO LOGS VALUES (7,2);

SELECT DISTINCT a.Num AS ConsecutiveNums
FROM LOGS a, LOGS b, LOGS c
WHERE a.Id = b.Id - 1
AND b.Id = c.Id - 1
AND a.Num = b. Num
AND b.Num = c.Num
```

184

```
DROP TABLE IF EXISTS Employee;
DROP TABLE IF EXISTS Department;
CREATE TABLE IF NOT EXISTS Employee (Id INT, NAME VARCHAR(255), Salary INT, DepartmentId INT);
CREATE TABLE IF NOT EXISTS Department (Id INT, NAME VARCHAR(255));
TRUNCATE TABLE Employee;
INSERT INTO Employee (Id, NAME, Salary, DepartmentId) VALUES ('1', 'Joe', '70000', '1');
INSERT INTO Employee (Id, NAME, Salary, DepartmentId) VALUES ('2', 'Jim', '90000', '1');
INSERT INTO Employee (Id, NAME, Salary, DepartmentId) VALUES ('3', 'Henry', '80000', '2');
INSERT INTO Employee (Id, NAME, Salary, DepartmentId) VALUES ('4', 'Sam', '60000', '2');
INSERT INTO Employee (Id, NAME, Salary, DepartmentId) VALUES ('5', 'Max', '90000', '1');
TRUNCATE TABLE Department;
INSERT INTO Department (Id, NAME) VALUES ('1', 'IT');
INSERT INTO Department (Id, NAME) VALUES ('2', 'Sales');

SELECT d.name Department, e.name Employee, e.Salary
FROM Employee e
JOIN Department d
ON e.DepartmentId = d.Id
WHERE (e.salary,e.DepartmentId) IN (SELECT MAX(salary),DepartmentId FROM employee GROUP BY DepartmentId)
```

626

```
DROP TABLE IF EXISTS seat;
CREATE TABLE IF NOT EXISTS seat(id INT, student VARCHAR(255));
TRUNCATE TABLE seat;

INSERT INTO seat (id, student) VALUES ('1', 'Abbot');
INSERT INTO seat (id, student) VALUES ('2', 'Doris');
INSERT INTO seat (id, student) VALUES ('3', 'Emerson');
INSERT INTO seat (id, student) VALUES ('4', 'Green');
INSERT INTO seat (id, student) VALUES ('5', 'Jeames');

SELECT
(CASE
WHEN id%2 = 1 AND id != c.counts THEN id+1
WHEN id%2 = 1 AND id = c.counts THEN id
ELSE id-1
END) AS id, student
FROM seat,
   (SELECT COUNT(*) counts FROM seat ) c
ORDER BY id
```

570

```
drop table if exists employee;
Create table If Not Exists Employee (
    Id int,
    Name varchar(255),
    Department varchar(255),
    ManagerId int);
Truncate table Employee;
insert into Employee (Id, Name, Department, ManagerId)
    values ('101', 'John', 'A', null);
insert into Employee (Id, Name, Department, ManagerId)
    values ('102', 'Dan', 'A', '101');
insert into Employee (Id, Name, Department, ManagerId)
    values ('103', 'James', 'A', '101');
insert into Employee (Id, Name, Department, ManagerId)
    values ('104', 'Amy', 'A', '101');
insert into Employee (Id, Name, Department, ManagerId)
    values ('105', 'Anne', 'A', '101');
insert into Employee (Id, Name, Department, ManagerId)
    values ('106', 'Ron', 'B', '101');

SELECT NAME
FROM Employee a
WHERE Id IN (SELECT ManagerId
                  FROM Employee
                  GROUP BY ManagerId
                  HAVING COUNT(*) >= 5)
```

574

```
drop table if exists candidate;
drop table if exists vote;

Create table If Not Exists Candidate (id int, Name varchar(255));
Create table If Not Exists Vote (id int, CandidateId int);

Truncate table Candidate;
insert into Candidate (id, Name) values ('1', 'A');
insert into Candidate (id, Name) values ('2', 'B');
insert into Candidate (id, Name) values ('3', 'C');
insert into Candidate (id, Name) values ('4', 'D');
insert into Candidate (id, Name) values ('5', 'E');
Truncate table Vote;
insert into Vote (id, CandidateId) values ('1', '2');
insert into Vote (id, CandidateId) values ('2', '4');
insert into Vote (id, CandidateId) values ('3', '3');
insert into Vote (id, CandidateId) values ('4', '2');
insert into Vote (id, CandidateId) values ('5', '5');

SELECT NAME
FROM Candidate
WHERE id = (SELECT candidateId
            FROM vote
            GROUP BY CandidateId
            ORDER BY COUNT(*) DESC
            LIMIT 1)
```

178

```
DROP TABLE IF EXISTS Scores;

CREATE TABLE IF NOT EXISTS Scores (Id INT, Score DECIMAL(3,2));

TRUNCATE TABLE Scores;

INSERT INTO Scores (Id, Score) VALUES (1, 3.50);

INSERT INTO Scores (Id, Score) VALUES (2, 3.65);

INSERT INTO Scores (Id, Score) VALUES (3, 4.00);

INSERT INTO Scores (Id, Score) VALUES (4, 3.85);

INSERT INTO Scores (Id, Score) VALUES (5, 4.00);

INSERT INTO Scores (Id, Score) VALUES (6, 3.65);


SELECT a.Score, (SELECT COUNT(DISTINCT Score)

                      FROM  Scores

                      WHERE Score >= a.Score

                      ) AS Rank

FROM Scores a

ORDER BY Rank
```

608

```
DROP TABLE IF EXISTS tree;
CREATE TABLE IF NOT EXISTS tree (id INT, p_id INT);
TRUNCATE TABLE tree;
INSERT INTO tree (id, p_id) VALUES ('1', NULL);
INSERT INTO tree (id, p_id) VALUES ('2', '1');
INSERT INTO tree (id, p_id) VALUES ('3', '1');
INSERT INTO tree (id, p_id) VALUES ('4', '2');
INSERT INTO tree (id, p_id) VALUES ('5', '2');

SELECT
    id,
    (CASE
     WHEN p_id IS NULL THEN "Root"
     WHEN id IN (SELECT p_id FROM tree) THEN "Inner"
     ELSE "Leaf"
     END) TYPE
FROM tree
```

578

```
drop table if exists survey_log;
Create table If Not Exists survey_log (
    uid int,
    action varchar(255),
    question_id int,
    answer_id int,
    q_num int,
    timestamp int);
Truncate table survey_log;
insert into survey_log values ('5', 'show', '285', null, '1', '123');
insert into survey_log values ('5', 'answer', '285', '124124', '1', '124');
insert into survey_log values ('5', 'show', '369', null, '2', '125');
insert into survey_log values ('5', 'skip', '369', null, '2', '126');



SELECT question_id AS survey_log
FROM (SELECT question_id,
                SUM(CASE WHEN ACTION = "show" THEN 1 ELSE 0 END) AS show_count,
                SUM(CASE WHEN ACTION = "answer" THEN 1 ELSE 0 END) AS answer_count
        FROM survey_log
        GROUP BY question_id
        ) a
ORDER BY (answer_count/show_count) DESC
LIMIT 1
```

580

```
drop table if exists student;

drop table if exists department;

CREATE TABLE IF NOT EXISTS student (

    student_id INT,

    student_name VARCHAR(45),

    gender VARCHAR(6),

    dept_id INT);

CREATE TABLE IF NOT EXISTS department (

    dept_id INT,

    dept_name VARCHAR(255));

Truncate table student;

insert into student values ('1', 'Jack', 'M', '1');

insert into student values ('2', 'Jane', 'F', '1');

insert into student values ('3', 'Mark', 'M', '2');

Truncate table department;

insert into department values ('1', 'Engineering');

insert into department values ('2', 'Science');

insert into department values ('3', 'Law');


SELECT dept_name, COUNT(student_id) AS student_number

FROM student s

RIGHT JOIN department d

ON s.`dept_id` = d.`dept_id`

GROUP BY dept_name

ORDER BY student_number DESC, dept_name ASC
```

585

```
drop table if exists insurance;

create table if not exists insurance (pid integer(11), tiv_2015 numeric(15,2), tiv_2016 numeric(15,2), lat numeric(5,2), lon numeric(5,2) );

Truncate table insurance;

insert into insurance (PID, TIV_2015, TIV_2016, LAT, LON) values ('1', '10', '5', '10', '10');

insert into insurance (PID, TIV_2015, TIV_2016, LAT, LON) values ('2', '20', '20', '20', '20');

insert into insurance (PID, TIV_2015, TIV_2016, LAT, LON) values ('3', '10', '30', '20', '20');

insert into insurance (PID, TIV_2015, TIV_2016, LAT, LON) values ('4', '10', '40', '40', '40');
```

```
SELECT SUM(TIV_2016)
FROM insurance a
WHERE (SELECT COUNT(*)
        FROM insurance b
        WHERE a.TIV_2015 = b.TIV_2015) > 1
AND (SELECT COUNT(*)
        FROM insurance c
        WHERE a.`LAT` = c.LAT AND a.`LON` = c.LON) = 1
```

602

```
drop table if exists request_accepted;
Create table If Not Exists request_accepted (
    requester_id INT NOT NULL,
    accepter_id INT NULL,
    accept_date DATE NULL);
Truncate table request_accepted;
insert into request_accepted (requester_id, accepter_id, accept_date)
    values ('1', '2', '2016/06/03');
insert into request_accepted (requester_id, accepter_id, accept_date)
    values ('1', '3', '2016/06/08');
insert into request_accepted (requester_id, accepter_id, accept_date)
    values ('2', '3', '2016/06/08');
insert into request_accepted (requester_id, accepter_id, accept_date)
    values ('3', '4', '2016/06/09');

SELECT id1 id, COUNT(id2) num
FROM (SELECT requester_id id1, accepter_id id2
          FROM request_accepted

          UNION

          SELECT accepter_id id1, requester_id id2
          FROM request_accepted
          ) tbl
GROUP BY id1
ORDER BY num DESC
LIMIT 1
```

612

```
DROP TABLE IF EXISTS point_2d;

CREATE TABLE IF NOT EXISTS point_2d (X INT NOT NULL, Y INT NOT NULL);

TRUNCATE TABLE point_2d;

INSERT INTO point_2d (X, Y) VALUES ('-1', '-1');

INSERT INTO point_2d (X, Y) VALUES ('0', '0');

INSERT INTO point_2d (X, Y) VALUES ('-1', '-2');


SELECT ROUND(SQRT(MIN(POW(a.x-b.x, 2)+ POW(a.y-b.y, 2))),2) shortest

FROM point_2d a, point_2d b

WHERE (a.x, a.y) <> (b.x, b.y)
```

614

```
drop table if exists follow;
Create table If Not Exists follow (followee varchar(255), follower varchar(255));
Truncate table follow;
insert into follow (followee, follower) values ('A', 'B');
insert into follow (followee, follower) values ('B', 'C');
insert into follow (followee, follower) values ('B', 'D');
insert into follow (followee, follower) values ('D', 'E');

SELECT a.follower, COUNT(b.follower) num
FROM follow a
JOIN follow b
ON a.follower = b. followee
GROUP BY a.`follower`
ORDER BY a.`follower`
```

Hard

569

```
drop table if exists employee;
Create table If Not Exists Employee (
    Id int,
    Company varchar(255),
    Salary int);

Truncate table Employee;
insert into Employee (Id, Company, Salary)
    values ('1', 'A', '2341');
insert into Employee (Id, Company, Salary)
    values ('2', 'A', '341');
insert into Employee (Id, Company, Salary) values
    ('3', 'A', '15');
insert into Employee (Id, Company, Salary) values
    ('4', 'A', '15314');
insert into Employee (Id, Company, Salary) values
    ('5', 'A', '451');
insert into Employee (Id, Company, Salary) values
    ('6', 'A', '513');
insert into Employee (Id, Company, Salary) values
    ('7', 'B', '15');
insert into Employee (Id, Company, Salary) values
    ('8', 'B', '13');
insert into Employee (Id, Company, Salary) values
    ('9', 'B', '1154');
insert into Employee (Id, Company, Salary) values
    ('10', 'B', '1345');
insert into Employee (Id, Company, Salary) values
    ('11', 'B', '1221');
insert into Employee (Id, Company, Salary) values
```

```
    ('12', 'B', '234');
insert into Employee (Id, Company, Salary) values
    ('13', 'C', '2345');
insert into Employee (Id, Company, Salary) values
    ('14', 'C', '2645');
insert into Employee (Id, Company, Salary) values
    ('15', 'C', '2645');
insert into Employee (Id, Company, Salary) values
    ('16', 'C', '2652');
insert into Employee (Id, Company, Salary) values
    ('17', 'C', '65');


SELECT *
FROM Employee e
WHERE ABS( (SELECT COUNT(*) FROM Employee e1 WHERE e.`Company`=e1.`Company` AND e.`Salary` >= e1.`Salary` )
                    -
                    (SELECT COUNT(*) FROM Employee e2 WHERE e.`Company`=e2.`Company` AND e.`Salary` <= e2.`Salary`) )

                    <= (SELECT COUNT(*) FROM Employee e3 WHERE e.`Company`=e3.`Company` AND e.`Salary` = e3.`Salary`)
GROUP BY Company, Salary
```

Leetcode answer:

```
SELECT
    Employee.Id, Employee.Company, Employee.Salary
FROM
    Employee,
    Employee alias
```

```
WHERE
    Employee.Company = alias.Company
GROUP BY Employee.Company , Employee.Salary
HAVING SUM(CASE
    WHEN Employee.Salary = alias.Salary THEN 1
    ELSE 0
END) >= ABS(SUM(SIGN(Employee.Salary - alias.Salary)))
```

185

```
DROP TABLE IF EXISTS employee;
DROP TABLE IF EXISTS department;

CREATE TABLE IF NOT EXISTS Employee (
 Id INT,
 NAME VARCHAR(255),
 Salary INT,
 DepartmentId INT);
CREATE TABLE IF NOT EXISTS Department (
 Id INT,
NAME VARCHAR(255));

TRUNCATE TABLE Employee;
INSERT INTO Employee (Id, NAME, Salary, DepartmentId)
 VALUES ('1', 'Joe', '70000', '1');
INSERT INTO Employee (Id, NAME, Salary, DepartmentId)
 VALUES ('2', 'Henry', '80000', '2');
INSERT INTO Employee (Id, NAME, Salary, DepartmentId)
 VALUES ('3', 'Sam', '60000', '2');
INSERT INTO Employee (Id, NAME, Salary, DepartmentId)
 VALUES ('4', 'Max', '90000', '1');
INSERT INTO Employee (Id, NAME, Salary, DepartmentId)
 VALUES ('5', 'Janet', '69000', '1');
 INSERT INTO Employee (Id, NAME, Salary, DepartmentId)
 VALUES ('6', 'Randy', '85000', '1');



TRUNCATE TABLE Department;
INSERT INTO Department (Id, NAME)
 VALUES ('1', 'IT');
```

```
INSERT INTO Department (Id, NAME)
 VALUES ('2', 'Sales');
```

```
SELECT d.name Department, e.name Employee, e.salary Salary
FROM employee e
JOIN department d
ON (e.departmentid = d.id)
WHERE (SELECT COUNT(DISTINCT b.Salary) FROM Employee b
      WHERE e.DepartmentId = b.DepartmentId
      AND e.Salary <= b.Salary) <= 3
ORDER BY Department, salary DESC
```

615

```
drop table if exists salary;
drop table if exists employee;

Create table If Not Exists salary (
    id int,
    employee_id int,
    amount int,
    pay_date date);
Create table If Not Exists employee (
    employee_id int,
    department_id int);

Truncate table salary;
insert into salary
    (id, employee_id, amount, pay_date)
values
    ('1', '1', '9000', '2017/03/31');
insert into salary
    (id, employee_id, amount, pay_date)
values
    ('2', '2', '6000', '2017/03/31');
insert into salary
    (id, employee_id, amount, pay_date)
values
    ('3', '3', '10000', '2017/03/31');
insert into salary
    (id, employee_id, amount, pay_date)
values
    ('4', '1', '7000', '2017/02/28');
insert into salary
    (id, employee_id, amount, pay_date)
```

```
values
    ('5', '2', '6000', '2017/02/28');
insert into salary
    (id, employee_id, amount, pay_date)
values
    ('6', '3', '8000', '2017/02/28');

Truncate table employee;
insert into employee
    (employee_id, department_id)
values
    ('1', '1');
insert into employee
    (employee_id, department_id)
values
    ('2', '2');
insert into employee
    (employee_id, department_id)
values
    ('3', '2');



    SELECT DATE_FORMAT(dpt.pay_date, '%Y-%m') pay_month, dpt.department_id,
              CASE
              WHEN dpt_avg > com_avg THEN "higher"
              WHEN dpt_avg < com_avg THEN "lower"
              ELSE "same"
              END AS comparison
    FROM (SELECT AVG(amount) com_avg, pay_date
              FROM salary
              GROUP BY pay_date) com
              JOIN

              (SELECT AVG(amount) dpt_avg, pay_date, department_id
              FROM salary s
              JOIN employee e
              ON s.employee_id = e.employee_id
              GROUP BY department_id, pay_date) dpt

              ON com.pay_date = dpt.pay_date
    ORDER BY pay_month DESC, department_id ASC
```

579

```
drop table if exists employee;
Create table If Not Exists Employee (Id int, Month int, Salary int);
Truncate table Employee;

insert into Employee (Id, Month, Salary) values ('1', '1', '20');
insert into Employee (Id, Month, Salary) values ('2', '1', '20');
insert into Employee (Id, Month, Salary) values ('1', '2', '30');
insert into Employee (Id, Month, Salary) values ('2', '2', '30');
insert into Employee (Id, Month, Salary) values ('3', '2', '40');
insert into Employee (Id, Month, Salary) values ('1', '3', '40');
insert into Employee (Id, Month, Salary) values ('3', '3', '60');
insert into Employee (Id, Month, Salary) values ('1', '4', '60');
insert into Employee (Id, Month, Salary) values ('3', '4', '70');

### add a.month to make it clear
SELECT a.id, a.`Month`,MAX(b.`Month`) MONTH, SUM(b.`Salary`) Salary
FROM Employee a
JOIN Employee b
ON a.`Id` = b.`Id`
AND b.`Month`BETWEEN a.`Month` - 3 AND a.`Month` - 1
GROUP BY a.`Id`, a.`Month`
ORDER BY a.`Id` ASC, MONTH DESC
```

262

```sql
drop table if exists trips;
drop table if exists users;

Create table If Not Exists Trips (Id int, Client_Id int, Driver_Id int, City_Id int, Status ENUM('completed', 'cancelled_by_driver', 'cancelled_by_client'), Request_at varchar(50));
Create table If Not Exists Users (Users_Id int, Banned varchar(50), Role ENUM('client', 'driver', 'partner'));

Truncate table Trips;
insert into Trips (Id, Client_Id, Driver_Id, City_Id, Status, Request_at) values ('1', '1', '10', '1', 'completed', '2013-10-01');
insert into Trips (Id, Client_Id, Driver_Id, City_Id, Status, Request_at) values ('2', '2', '11', '1', 'cancelled_by_driver', '2013-10-01');
insert into Trips (Id, Client_Id, Driver_Id, City_Id, Status, Request_at) values ('3', '3', '12', '6', 'completed', '2013-10-01');
insert into Trips (Id, Client_Id, Driver_Id, City_Id, Status, Request_at) values ('4', '4', '13', '6', 'cancelled_by_client', '2013-10-01');
insert into Trips (Id, Client_Id, Driver_Id, City_Id, Status, Request_at) values ('5', '1', '10', '1', 'completed', '2013-10-02');
insert into Trips (Id, Client_Id, Driver_Id, City_Id, Status, Request_at) values ('6', '2', '11', '6', 'completed', '2013-10-02');
insert into Trips (Id, Client_Id, Driver_Id, City_Id, Status, Request_at) values ('7', '3', '12', '6', 'completed', '2013-10-02');
insert into Trips (Id, Client_Id, Driver_Id, City_Id, Status, Request_at) values ('8', '2', '12', '12', 'completed', '2013-10-03');
insert into Trips (Id, Client_Id, Driver_Id, City_Id, Status, Request_at) values ('9', '3', '10', '12', 'completed', '2013-10-03');
insert into Trips (Id, Client_Id, Driver_Id, City_Id, Status, Request_at) values ('10', '4', '13', '12', 'cancelled_by_driver', '2013-10-03');

Truncate table Users;
insert into Users (Users_Id, Banned, Role) values ('1', 'No', 'client');
insert into Users (Users_Id, Banned, Role) values ('2', 'Yes', 'client');
insert into Users (Users_Id, Banned, Role) values ('3', 'No', 'client');
insert into Users (Users_Id, Banned, Role) values ('4', 'No', 'client');
insert into Users (Users_Id, Banned, Role) values ('10', 'No', 'driver');
insert into Users (Users_Id, Banned, Role) values ('11', 'No', 'driver');
insert into Users (Users_Id, Banned, Role) values ('12', 'No', 'driver');
insert into Users (Users_Id, Banned, Role) values ('13', 'No', 'driver');


SELECT Request_at AS DAY,
        ROUND( SUM(CASE WHEN STATUS LIKE "cancelled%" THEN 1 ELSE 0 END)/ COUNT(*) ,2) AS "Cancelled Rate"
FROM (SELECT *
        FROM Trips
        WHERE Client_Id IN (SELECT Users_id FROM Users WHERE Banned = "No" AND Role = "client")
        AND   Driver_Id IN (SELECT Users_id FROM Users WHERE Banned = "No" AND Role = "driver")
        AND   Request_at BETWEEN "2013-10-01" AND "2013-10-03") newtrips
GROUP BY DAY
```

571

```
drop table if exists numbers;
Create table If Not Exists Numbers (
    Number int,
    Frequency int);

Truncate table Numbers;
insert into Numbers (Number, Frequency) values ('0', '7');
insert into Numbers (Number, Frequency) values ('1', '1');
insert into Numbers (Number, Frequency) values ('2', '3');
insert into Numbers (Number, Frequency) values ('3', '1');

SELECT AVG(number) median
FROM Numbers a
WHERE ABS((SELECT SUM(Frequency)
                FROM Numbers b
                WHERE a.Frequency >= b.Frequency)
        -
        (SELECT SUM(Frequency)
                FROM Numbers b
                WHERE a.Frequency <= b.Frequency)
        ) <= a.Frequency
```

601

```
DROP TABLE IF EXISTS stadium;
CREATE TABLE IF NOT EXISTS stadium (id INT, DATE DATE NULL, people INT);
TRUNCATE TABLE stadium;

INSERT INTO stadium (id, DATE, people) VALUES ('1', '2017-01-01', '10');
INSERT INTO stadium (id, DATE, people) VALUES ('2', '2017-01-02', '109');
INSERT INTO stadium (id, DATE, people) VALUES ('3', '2017-01-03', '150');
INSERT INTO stadium (id, DATE, people) VALUES ('4', '2017-01-04', '99');
INSERT INTO stadium (id, DATE, people) VALUES ('5', '2017-01-05', '145');
INSERT INTO stadium (id, DATE, people) VALUES ('6', '2017-01-06', '1455');
INSERT INTO stadium (id, DATE, people) VALUES ('7', '2017-01-07', '199');
INSERT INTO stadium (id, DATE, people) VALUES ('8', '2017-01-08', '188');

SELECT DISTINCT a.*
FROM stadium a, stadium b, stadium c
WHERE ((a.id + 1 = b.id AND b.id + 1 = c.id )
        OR
        (b.id + 1 = a.id AND a.id + 1 = c.id )
        OR
        (b.id + 1 = c.id AND c.id + 1 = a.id ))
        AND a.people >= 100
        AND b.people >= 100
        AND c.people >= 100
ORDER BY id
```

618

```
drop table if exists student;
Create table If Not Exists student (name varchar(50), continent varchar(7));
Truncate table student;

insert into student (name, continent) values ('Jane', 'America');
insert into student (name, continent) values ('Pascal', 'Europe');
insert into student (name, continent) values ('Xi', 'Asia');
insert into student (name, continent) values ('Jack', 'America');

SELECT America.name AS America, Asia.name AS Asia, Europe.name AS Europe FROM
(SELECT NAME, @a :=@a +  1 AS id FROM student, (SELECT @a := 0) i
WHERE continent = 'America' ORDER BY NAME) AS America LEFT JOIN

(SELECT NAME, @b :=@b +  1 AS id FROM student, (SELECT @b := 0) i
WHERE continent = 'Asia' ORDER BY NAME) AS Asia ON (America.id = Asia.ID) LEFT JOIN

(SELECT NAME, @c :=@c +  1 AS id FROM student, (SELECT @c := 0) i
WHERE continent = 'Europe' ORDER BY NAME) AS Europe ON (America.id = Europe.id)
```