**--- get into database**

**use SQL\_Assignment;**

--- Q101

**Write an SQL query to show the second most recent activity of each user. If the user only has one activity, return that one. A user cannot perform more than one activity at the same time.**

create table UserActivity

(

username varchar(50),

activity varchar(30),

startDate Date,

endDate Date

);

insert into UserActivity values ('Alice', 'Travel', '2020-02-12', '2020-02-20'),

('Alice', 'Dancing', '2020-02-21', '2020-02-23'),

('Alice', 'Travel', '2020-02-24', '2020-02-28'),

('Bob', 'Travel', '2020-02-11', '2020-02-18');

select \* from UserActivity where (username, startDate) in (

select u1.username, max(u1.startDate) from UserActivity u1

where (u1.username, u1.startDate) not in (

select u2.username, max(u2.startDate) from UserActivity u2

group by u2.username

having count(u2.username) > 1

)

group by u1.username

);

--- Q102

--- Same as Q101

--- Q103

**Query the Name of any student in STUDENTS who scored higher than 75 Marks. Order your output by the last three characters of each name. If two or more students both have names ending in the same last three characters (i.e.: Bobby, Robby, etc.), secondary sort them by ascending ID.**

create table students\_2

(

id int,

name varchar(40),

marks int

);

insert into students\_2 values (1, 'Ashley', 81),

(2, 'Samantha', 75),

(3, 'Julia', 76),

(4, 'Belvet', 84);

select name

from students\_2

where marks > 75

order by substr(name, -3);

--- Q104

**Write a query that prints a list of employee names (i.e.: the name attribute) for employees in Employee having a salary greater than $2000 per month who have been employees for less than 10 months. Sort your result by ascending employee\_id.**

create table employee\_6

(

employee\_id int,

name varchar(50),

months int,

salary int

);

insert into employee\_6 values (12228, 'Rose', 15, 1968),

(33645, 'Angela', 1, 3443),

(45692, 'Frank', 17, 1608),

(56118, 'Patrick', 7, 1345),

(59725, 'Lisa', 11, 2330),

(74197, 'Kimberly', 16, 4372),

(78454, 'Bonnie', 8, 1771),

(83565, 'Michael', 6, 2017),

(98607, 'todd', 5, 3396),

(99989, 'Joe', 9, 3573);

select name

from employee\_6

where salary > 2000 and months < 10

order by employee\_id;

--- Q105

**Write a query identifying the type of each record in the TRIANGLES table using its three side lengths. Output one of the following statements for each record in the table: ● Equilateral: It's a triangle with sides of equal length. ● Isosceles: It's a triangle with sides of equal length. ● Scalene: It's a triangle with sides of differing lengths. ● Not A Triangle: The given values of A, B, and C don't form a triangle.**

create table triangles

(

A int,

B int,

C int

);

insert into triangles values (20, 20, 23),

(20, 20, 20),

(20, 21, 22),

(13, 14, 30);

select

case

when C > A + B or A > B + C or B > C + A then 'Not a Triangle'

when A = B and B != C then 'Isosceles Triangle'

when A = B and B = C then 'Equilateral Triangle'

when A != B and B != C then 'Scalene Triangle'

end as Type

from triangles;

--- Q106

**Samantha was tasked with calculating the average monthly salaries for all employees in the EMPLOYEES table, but did not realise her keyboard's 0 key was broken until after completing the calculation. She wants your help finding the difference between her miscalculation (using salaries with any zeros removed), and the actual average salary. Write a query calculating the amount of error (i.e.: actual - miscalculated average monthly salaries), and round it up to the next integer.**

create table employees\_2

(

id int,

name varchar(50),

salary int

);

insert into employees\_2 values (1, 'Kristeen', 1420),

(2, 'Ashley', 2006),

(3, 'Julia', 2210),

(4, 'Maria', 3000);

select (avg(salary) - avg(replace(salary, '0', ''))) as avg\_salary\_diff

from employees\_2;

--- Q107

**We define an employee's total earnings to be their monthly salary \* months worked, and the maximum total earnings to be the maximum total earnings for any employee in the Employee table. Write a query to find the maximum total earnings for all employees as well as the total number of employees who have maximum total earnings. Then print these values as 2 space-separated integers.**

create table employee\_6

(

employee\_id int,

name varchar(50),

months int,

salary int

);

insert into employee\_6 values (12228, 'Rose', 15, 1968),

(33645, 'Angela', 1, 3443),

(45692, 'Frank', 17, 1608),

(56118, 'Patrick', 7, 1345),

(59725, 'Lisa', 11, 2330),

(74197, 'Kimberly', 16, 4372),

(78454, 'Bonnie', 8, 1771),

(83565, 'Michael', 6, 2017),

(98607, 'todd', 5, 3396),

(99989, 'Joe', 9, 3573);

select concat(earnings, space(2), count(earnings)) as earning\_\_count

from

(select max(salary \* months) as earnings

from employee\_6) as temp;

--- Q108

**Generate the following two result sets:**

**1. Query an alphabetically ordered list of all names in OCCUPATIONS, immediately followed by the first letter of each profession as a parenthetical (i.e.: enclosed in parentheses). For example: AnActorName(A), ADoctorName(D), AProfessorName(P), and ASingerName(S). Query the number of occurrences of each occupation in OCCUPATIONS. Sort the occurrences in ascending order, and output them in the following format: Level - Medium There are a total of [occupation\_count] [occupation]s.**

**2. where [occupation\_count] is the number of occurrences of an occupation in OCCUPATIONS and [occupation] is the lowercase occupation name. If more than one Occupation has the same [occupation\_count], they should be ordered alphabetically.**

create table occupations

(

name varchar(50),

occupation varchar(30)

);

insert into occupations values ('Samantha', 'Doctor'),

('Julia', 'Actor'),

('Maria', 'Actor'),

('Meera', 'Singer'),

('Ashely', 'Professor'),

('Ketty', 'Professor'),

('Christeen', 'Professor'),

('Jane', 'Actor'),

('Jenny', 'Doctor'),

('Priya', 'Singer');

select concat(name, '(', left(substr(occupation,1),1), ')')as output

from occupations

order by name;

--- Q109

**Pivot the Occupation column in OCCUPATIONS so that each Name is sorted alphabetically and displayed underneath its corresponding Occupation. The output column headers should be Doctor, Professor, Singer, and Actor, respectively.**

select

Doctor,

Professor,

Singer,

Actor

from (

select

NameOrder,

max(case Occupation when 'Doctor' then Name end) as Doctor,

max(case Occupation when 'Professor' then Name end) as Professor,

max(case Occupation when 'Singer' then Name end) as Singer,

max(case Occupation when 'Actor' then Name end) as Actor

from (

select

Occupation,

Name,

row\_number() over(partition by Occupation order by Name ASC) as NameOrder

from Occupations

) as NameLists

group by NameOrder

) as Names;

--- Q110

**Write a query to find the node type of Binary Tree ordered by the value of the node. Output one of the following for each node: ● Root: If node is root node. ● Leaf: If node is leaf node. ● Inner: If node is neither root nor leaf node.**

create table BST

(

N int,

P int

);

insert into BST values (1,2), (3,2), (6,8), (9,8), (2,5), (8,5), (5,null);

select N,

case

when P is null then 'Root'

when N not in (select P from BST where P is not null) then 'Leaf'

else 'Inner'

end as temp

from BST

order by N;