**SQL – SET3**

**Create database class\_3;**

**Use class\_3;**

**Q101.**

create table UserActivity(

username varchar(6), activity varchar(10),

startDate date, endDate date);

**IN:**

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');

**Query:**

with tmp as (select username,activity, startDate, endDate,

dense\_rank() over(partition by username order by startDate, endDate) as activity\_num ,

count(\*) over( partition by username) as activity\_count

from UserActivity)

select username, activity, startDate, endDate from tmp

where case when activity\_count = 1 then 1

when activity\_num = 2 then 1

end = 1;



**Q102.**

**101 Question is Reapeated**

**Q103.**

create table students(

id int, name varchar(10),

marks int) ;

**IN:**

insert into students values(1, 'Ashley', 81),(2, 'Samantha', 75),(4, 'Julia', 76),(3, 'Belvet', 84);

**Query:**

select name from students

where marks > 75

order by right(name,3), id;



**Q104.**

create table employee(

employee\_id int,name varchar(15),

months int, salary int);

**IN:**

INSERT INTO employee 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),

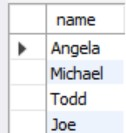
(78454, 'Bonnie', 8, 1771),(83565, 'Michael', 6, 2017),(98607, 'Todd', 5, 3396),(99989, 'Joe', 9, 3573);

**Query:**

select name from employee

where months < 10

and salary > 2000;



**Q105.**

create table triangles(

a int, b int, c int);

**IN:**

insert into triangles values(20,20,23),(20,20,20),(20,21,21),(13,14,30);

**Query:**

select CASE

When a + b < c or a + c < b or b + c < a then 'Not A Triangle'

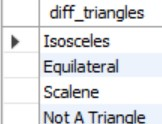
When a = b and b = c then 'Equilateral'

When a = b then 'Isosceles'

else 'Scalene'

end as diff\_triangles

from triangles;



**Q106.**

create table employees(

id int, name varchar(10),

salary int );

**IN:**

insert into employees values(1,'Kristeen', 1420),(2, 'Ashley', 2006),(3, 'Julia', 2210),(4, 'Maria', 3000);

**Query:**

SELECT CEIL(AVG(salary) - AVG(REPLACE(CAST(salary AS char), '0', ''))) as error\_amount

FROM EMPLOYEES;



**Q107.**

**Same Q 104. Data**

**Query:**

select concat(salary, ' ', count(salary) ) as output from

(select max(salary \* months) as salary

from employee) as tmp

group by salary;



**Q108.**

create table occupations(

name varchar(15), occupation varchar(10) );

**IN:**

insert into occupations values('Samantha', 'Doctor'),('Julia', 'Actor'),('Maria', 'Actor'),('Meera', 'Singer'),

('Ashely', 'Professor'),('Ketty', 'Professor'),('Christeen', 'Professor'),('Jane', 'Actor'),('Jenny', 'Doctor'),

('Priya', 'Singer');

**Query:**

WITH occupations\_cte AS (

SELECT concat(name, '(', left(occupation, 1), ')') AS Designation

FROM occupations

ORDER BY name ASC ),

val AS (

SELECT occupation, count(occupation) AS occ\_count

FROM occupations

GROUP BY occupation )

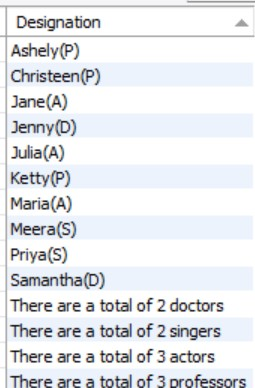
SELECT Designation FROM occupations\_cte

UNION ALL

SELECT concat('There are a total of ', occ\_count, ' ', lower(occupation), 's')

FROM val

ORDER BY Designation ASC;



**Q109.**

**Same data from Q108.**

**Query:**

WITH serialized\_ocp AS(

SELECT name, occupation,

row\_number() over(partition by occupation order by name) as serial

FROM occupations )

SELECT

MAX(CASE WHEN occupation = 'Doctor' THEN name END) AS Doctor,

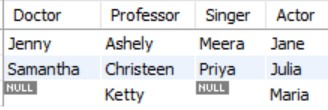
MAX(CASE WHEN occupation = 'Professor' THEN name END) AS Professor,

MAX(CASE WHEN occupation = 'Singer' THEN name END) AS Singer,

MAX(CASE WHEN occupation = 'Actor' THEN name END) AS Actor

FROM serialized\_ocp

GROUP BY serial;



**Q110.**

create table BST(

N int, P int);

**IN:**

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

**Query:**

select N as Node,

case

when P is null then 'root'

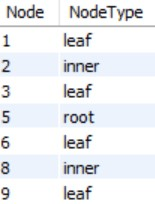
when N in (select P from BST) then 'inner'

else 'leaf'

end as NodeType

from BST

order by N;



**Q111.**

**-- Table - company**

CREATE TABLE company(

company\_code VARCHAR(25),

founder VARCHAR(25) );

**IN:** INSERT INTO company VALUES('C1', 'Monika'),('C2', 'Samantha');

**-- Table - lead\_manager**

CREATE TABLE lead\_manager(

lead\_manager\_code VARCHAR(25),

company\_code VARCHAR(25) );

**IN:** INSERT INTO lead\_manager VALUES('LM1', 'C1'),('LM2', 'C2');

**-- Table - senior\_manager**

CREATE TABLE senior\_manager(

senior\_manager\_code VARCHAR(25),

lead\_manager\_code VARCHAR(25),

company\_code VARCHAR(25) );

**IN:** INSERT INTO senior\_manager VALUES('SM1', 'LM1', 'C1'),('SM2', 'LM1', 'C1'),('SM3', 'LM2', 'C2');

**-- Table - manager**

CREATE TABLE manager(

manager\_code VARCHAR(25),

senior\_manager\_code VARCHAR(25),

lead\_manager\_code VARCHAR(25),

company\_code VARCHAR(25) );

**IN:** INSERT INTO manager VALUES('M1', 'SM1', 'LM1', 'C1'),('M2', 'SM3', 'LM2', 'C2'),('M3', 'SM3', 'LM2', 'C2');

**-- Table - employee**

CREATE TABLE employ(

employee\_code VARCHAR(25),

manager\_code VARCHAR(25),

senior\_manager\_code VARCHAR(25),

lead\_manager\_code VARCHAR(25),

company\_code VARCHAR(25) );

**IN:** INSERT INTO employ VALUES('E1', 'M1', 'SM1', 'LM1', 'C1'),('E2', 'M1', 'SM1', 'LM1', 'C1'),('E3', 'M2', 'SM3', 'LM2', 'C2'),('E4', 'M3', 'SM3', 'LM2', 'C2');

**Query:**

SELECT c.company\_code, c.founder,

count(distinct lm.lead\_manager\_code) AS lead\_manager\_count,

count(distinct sm.senior\_manager\_code) AS senior\_manager\_count,

count(distinct m.manager\_code) AS manager\_count,

count(distinct e.employee\_code) AS employeee\_count

FROM lead\_manager lm

LEFT JOIN senior\_manager sm ON lm.lead\_manager\_code = sm.lead\_manager\_code

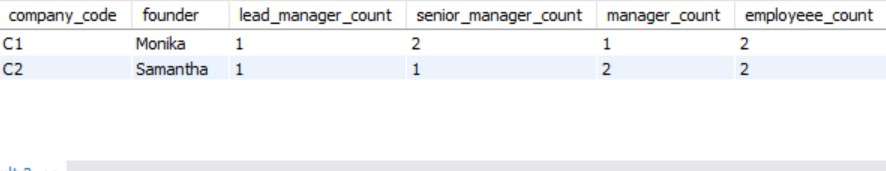
LEFT JOIN manager m ON m.senior\_manager\_code = sm.senior\_manager\_code

LEFT JOIN employ e ON e.manager\_code = m.manager\_code

LEFT JOIN company c ON c.company\_code = lm.company\_code

GROUP BY c.company\_code, c.founder

ORDER BY c.company\_code;



**Q112.**

**Query:**

WITH RECURSIVE numbers AS(

SELECT 1 AS n

UNION

SELECT n+1 FROM numbers WHERE n < 1000),

prime\_numbers AS( SELECT n1.n

FROM numbers n1

JOIN numbers n2 ON n1.n >= n2.n\*2 AND n2.n <> 1

WHERE n1.n > 3

GROUP BY n1.n

HAVING MIN(MOD(n1.n,n2.n)) <> 0

ORDER BY n1.n),

all\_prime AS(

SELECT 1 AS n UNION

SELECT 2 AS n UNION

SELECT 3 AS n UNION

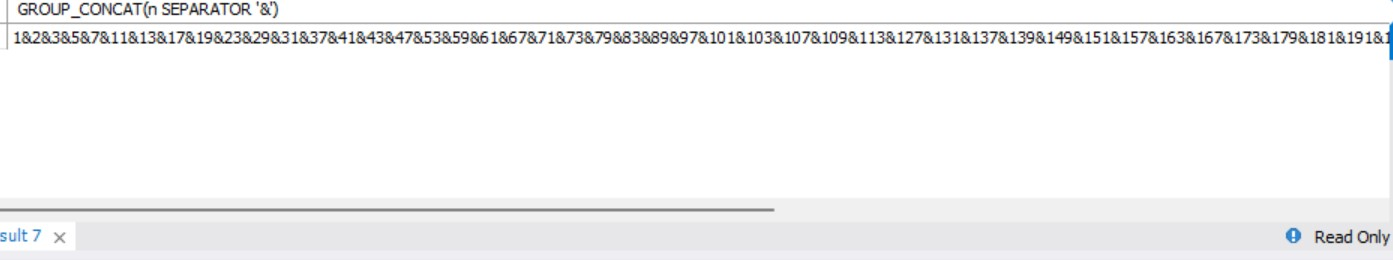
SELECT n

FROM prime\_numbers

ORDER BY n)

SELECT GROUP\_CONCAT(n SEPARATOR '&')

FROM all\_prime;



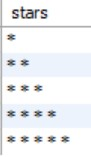
**Q113.**

set @number = 0;

select repeat('\* ', @number := @number + 1) as stars

from information\_schema.tables

where @number < 5



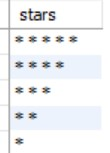
**Q114.**

set @number = 6;

select repeat('\* ', @number := @number - 1) as stars

from information\_schema.tables

where @number > 0



**Q115.**

Q103**. Is repeated.**

**Q116.**

CREATE TABLE functions(x INT, y INT);

**IN:** INSERT INTO functions VALUES(20, 20),(20, 20),(20, 21),(22, 23),(23,22),(21, 20);

**Query:**

select f1.x, f1.y from functions f1 join functions f2

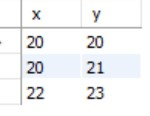
on f1.x = f2.y and f2.x = f1.y

group by f1.x,f1.y

having count(f1.x) > 1

or f1.x < f1.y

order by f1.x asc;



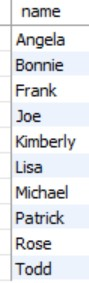
**Q117.**

**104 question is repeated in part b and same data for part a question**

**Query:**

select name from employee

order by name;



**Q118.**

**105 question is repeated**

**Q119.**

create table user\_transactions(

transaction\_id int, product\_id int,

spend double, transaction\_date datetime);

**IN:**

insert into user\_transactions values(1341, 123424, 1500.60, STR\_TO\_DATE('12/31/2019 12:00:00', '%m/%d/%Y %H:%i:%s')),

(1423, 123424, 1000.20, STR\_TO\_DATE('12/31/2020 12:00:00', '%m/%d/%Y %H:%i:%s')),

(1623, 123424, 1246.44, STR\_TO\_DATE('12/31/2021 12:00:00', '%m/%d/%Y %H:%i:%s')),

(1322, 123424, 2145.32, STR\_TO\_DATE('12/31/2022 12:00:00', '%m/%d/%Y %H:%i:%s'));

**Query:**

select year(transaction\_date) as year, product\_id,

spend as curr\_year\_spend,

lag(spend) over(order by transaction\_date) as prev\_year\_spend,

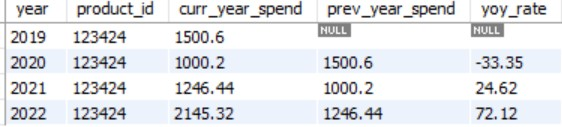
round(

(spend - lag(spend) over(order by transaction\_date))\*100/lag(spend) over(order by transaction\_date),

2) as yoy\_rate

from user\_transactions

order by year;



**Q120.**

create table amazon\_inventory(

item\_id int, item\_type varchar(25),

item\_category varchar(25), square\_footage double);

**IN:**

insert into amazon\_inventory values(1374,'prime\_eligible','minirefrigerator',68.00),(4245,'not\_prime','standing lamp',26.40),(2452,'prime\_eligible','television',85.00),

(3255,'not\_prime','side table',22.60),(1672,'prime\_eligible','laptop',8.50);

**Query:**

with prime\_tmp as (select floor((500000/sum(square\_footage))) as std,sum(square\_footage) as val

from amazon\_inventory

where item\_type = 'prime\_eligible')

select item\_type,

Case

when item\_type = 'prime\_eligible' then p.std\*count(item\_type)

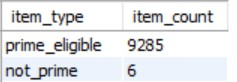
when item\_type = 'not\_prime' then

floor((500000 - (p.std\*p.val))\*count(item\_type)/(sum(square\_footage)))

end as item\_count

from amazon\_inventory,prime\_tmp p

group by item\_type,p.std,p.val;



**Q121.**

create table user\_actions(

user\_id int, event\_id int,

event\_type varchar(10), event\_date datetime);

**IN:** INSERT INTO user\_actions VALUES

(445,7765,'sign-in',STR\_TO\_DATE('05/31/2022 12:00:00', '%m/%d/%Y %H:%i:%s')),

(742,6458,'sign-in',STR\_TO\_DATE('06/03/2022 12:00:00', '%m/%d/%Y %H:%i:%s')),

(445,3634,'like',STR\_TO\_DATE('06/05/2022 12:00:00', '%m/%d/%Y %H:%i:%s')),

(742,1374,'comment',STR\_TO\_DATE('06/05/2022 12:00:00', '%m/%d/%Y %H:%i:%s')),

(648,3124,'like',STR\_TO\_DATE('06/18/2022 12:00:00', '%m/%d/%Y %H:%i:%s'));

**Query:**

with dir as (select user\_id , extract(month from event\_date) as mon from user\_actions

order by user\_id)

select a.mon , count(a.user\_id) as monthly\_active\_users from dir a join dir b on a.user\_id = b.user\_id

where a.mon - b.mon = 1

group by a.mon;



**Q122.**

create table search\_frequency\_table(

searches int, num\_users int);

**IN:**

insert into search\_frequency\_table values(1,2),(2,2),(3,3),(4,1);

**Query:**

with recursive CTE\_median as (

select searches,num\_users , 1 as temp from search\_frequency\_table

union all

select searches , num\_users , temp+1 from CTE\_median where temp+1 <= num\_users)

select round(avg(t.searches),1) as median

from (

select searches,

row\_number() over(order by searches,temp) as r1,

row\_number() over(order by searches desc,temp desc ) as r2

from CTE\_median

order by searches) as t

where t.r1 in (t.r2,t.r2-1,t.r2 +1);



**Q123.**

**-- Table - advertiser**

create table advertiser(

user\_id varchar(15),status varchar(15) );

**IN:** insert into advertiser values('bing','NEW'),('yahoo','NEW'),('alibaba','EXISTING');

**-- Table - daily\_pay**

create table daily\_pay(

user\_id varchar(10),paid decimal);

**IN:** insert into daily\_pay values('yahoo',45.00),('alibaba',100.00),('target',13.00);

**Query:**

select a.user\_id,

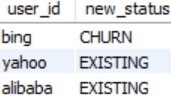
case

when a.status in ('NEW','EXISTING','RESURRECT') and paid is not null then 'EXISTING'

else 'CHURN'

end as new\_status

from advertiser a left join daily\_pay d on a.user\_id = d.user\_id;



**Q124.**

create table server\_utilization(

server\_id int, status\_time timestamp,

session\_status varchar(5));

**IN:**

insert into server\_utilization values(1,STR\_TO\_DATE('08/02/2022 10:00:00', '%m/%d/%Y %H:%i:%s'),'start'),

(1,STR\_TO\_DATE('08/04/2022 10:00:00', '%m/%d/%Y %H:%i:%s'),'stop'),

(2,STR\_TO\_DATE('08/17/2022 10:00:00', '%m/%d/%Y %H:%i:%s'),'start'),

(2,STR\_TO\_DATE('08/24/2022 10:00:00', '%m/%d/%Y %H:%i:%s'),'stop');

**Query:**

SELECT FLOOR(SUM(TIME\_TO\_SEC(TIMEDIFF(stop\_time, start\_time))) / (60\*60\*24)) AS total\_uptime\_days

FROM (

SELECT server\_id, MIN(status\_time) AS start\_time, MAX(status\_time) AS stop\_time

FROM server\_utilization

GROUP BY server\_id

HAVING COUNT(\*) % 2 = 0

) AS server\_uptime;



**Q125.**

create table transactions\_new (

transaction\_id int, merchant\_id int,

credit\_card\_id int, amount int,

transaction\_timestamp datetime);

**IN:**

insert into transactions\_new values (1,101,1,100,str\_to\_date('09/25/2022 12:00:00','%m/%d/%Y %H:%i:%s')),

(2,101,1,100,str\_to\_date('09/25/2022 12:08:00','%m/%d/%Y %H:%i:%s')),

(3,101,1,100,str\_to\_date('09/25/2022 12:28:00','%m/%d/%Y %H:%i:%s')),

(4,102,2,300,str\_to\_date('09/25/2022 12:00:00','%m/%d/%Y %H:%i:%s')),

(6,102,2,400,str\_to\_date('09/25/2022 14:00:00','%m/%d/%Y %H:%i:%s'));

**Query:**

SELECT COUNT(\*) as payment\_count

FROM ( SELECT \*,

lag(merchant\_id) over(partition by credit\_card) as prev\_merchant,

lag(transaction\_timestamp) OVER (partition by credit\_card ORDER BY transaction\_timestamp) AS prev\_datetime

FROM transactions\_new ) t

WHERE TIMESTAMPDIFF(MINUTE, t.prev\_datetime, transaction\_timestamp) between 1 and 10

and prev\_merchant - merchant\_id = 0;



**Q126.**

**-- Table - door\_orders**

create table door\_orders(

order\_id int, customer\_id int,

trip\_id int, status varchar(30),

order\_timestamp timestamp );

**IN:**

insert into door\_orders values(727424,8472,100463,'completed successfully',str\_to\_date('06/05/2022 09:12:00','%m/%d/%Y %H:%i:%s')),

(242513,2341,100482,'completed incorrectly',str\_to\_date('06/05/2022 14:40:00','%m/%d/%Y %H:%i:%s')),

(141367,1314,100362,'completed incorrectly',str\_to\_date('06/07/2022 15:03:00','%m/%d/%Y %H:%i:%s')),

(582193,5421,100657,'never\_received',str\_to\_date('07/07/2022 15:22:00','%m/%d/%Y %H:%i:%s')),

(253613,1314,100213,'completed successfully',str\_to\_date('06/12/2022 13:43:00','%m/%d/%Y %H:%i:%s'));

**-- Table - trips**

create table trips(

dasher\_id int, trip\_id int,

estimated\_delivery\_timestamp timestamp, actual\_delivery\_timestamp timestamp);

**IN:**

insert into trips values(101,100463,str\_to\_date('06/05/2022 09:42:00','%m/%d/%Y %H:%i:%s'),str\_to\_date('06/05/2022 09:38:00','%m/%d/%Y %H:%i:%s')),

(102,100482,str\_to\_date('06/05/2022 15:10:00','%m/%d/%Y %H:%i:%s'),str\_to\_date('06/05/2022 15:46:00','%m/%d/%Y %H:%i:%s')),

(101,100362,str\_to\_date('06/07/2022 15:33:00','%m/%d/%Y %H:%i:%s'),str\_to\_date('06/07/2022 16:45:00','%m/%d/%Y %H:%i:%s')),

(102,100657,str\_to\_date('07/07/2022 15:52:00','%m/%d/%Y %H:%i:%s'),null),

(103,100213,str\_to\_date('06/12/2022 14:13:00','%m/%d/%Y %H:%i:%s'),str\_to\_date('06/12/2022 14:10:00','%m/%d/%Y %H:%i:%s'));

**-- Table - door\_customers**

create table door\_customers(

customer\_id int, signup\_timestamp timestamp);

**IN:**

insert into door\_customers values(8472,str\_to\_date('05/30/2022 00:00:00','%m/%d/%Y %H:%i:%s')),

(2341,str\_to\_date('06/01/2022 00:00:00','%m/%d/%Y %H:%i:%s')),

(1314,str\_to\_date('06/03/2022 00:00:00','%m/%d/%Y %H:%i:%s')),

(1435,str\_to\_date('06/05/2022 00:00:00','%m/%d/%Y %H:%i:%s')),

(5421,str\_to\_date('06/07/2022 00:00:00','%m/%d/%Y %H:%i:%s'));

**Query:**

SELECT round((COUNT(CASE WHEN status <> 'completed successfully' THEN 1 ELSE NULL END)/count(\*))\*100,2) as tot

FROM door\_orders o join door\_customers c on o.customer\_id = c.customer\_id

where month(c.signup\_timestamp) = 6;



**Q127.**

create table if not exists Scores(

player\_name varchar(12), gender varchar(2),

day date, score\_points int,

primary key(gender,day) );

**IN:**

insert into Scores values('Aron','F','2020-01-01',17),('Alice','F','2020-01-07',23),

('Bajrang','M','2020-01-07',7),('Khali','M','2019-12-25',11),('Slaman','M','2019-12-30',13),

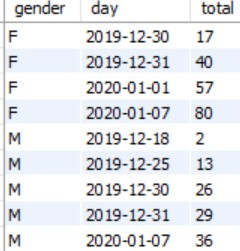
('Joe','M','2019-12-31',3),('Jose','M','2019-12-18',2),('Priya','F','2019-12-31',23),

('Priyanka','F','2019-12-30',17);

**Query:**

select gender,day,

sum(score\_points) over(partition by gender order by day) as total from Scores;



**Q128.**

**-- Table - Person**

create table if not exists Person(

id int, name varchar(10), phone\_number varchar(11),

primary key(id) );

**IN:**

insert into Person values(3,'Jonathan','051-1234567'),(12,'Elvis','051-7654321'),(1,'Moncef','212-1234567'),

(2,'Maroua','212-6523651'),(7,'Meir','972-1234567'),(9,'Rachel','972-0011100');

**-- Table - Country**

create table if not exists Country(

name varchar(8), country\_code varchar(3),

primary key(country\_code) );

**IN:**

insert into Country values('Peru','051'),('Israel','972'),('Morocco','212'),('Germany','049'),('Ethiopia','251');

**--Table - Calls**

create table if not exists Calls(

caller\_id int, callee\_id int, duration int);

**IN:**

insert into Calls values(1,9,33),(2,9,4),(1,2,59),(3,12,102),(3,12,330),(12,3,5),(7,9,13),(7,1,3),

(9,7,1),(1,7,7);

**Query:**

SELECT co.name AS country FROM person p JOIN country co ON left(phone\_number,3) = country\_code

JOIN calls c ON p.id IN (c.caller\_id, c.callee\_id)

GROUP BY co.name

HAVING AVG(duration) > (SELECT AVG(duration) FROM calls);



**Q129.**

create table numbers(

num int, frequency int,

primary key (num) );

**IN:**

insert into numbers values(0,7),(1,1),(2,3),(3,1);

**Query:**

with recursive tmp as(

select num,frequency,1 as tag from numbers

union

select num,frequency,tag+1 from tmp where tag < frequency

)

select round(sum(num)/2,0) as median from (select num,row\_number() over(order by num) as r\_num from tmp) as test

where r\_num = (select count(\*) from tmp)/2

or r\_num = (select count(\*) from tmp)/2 +1;



**Q130.**

**Table -salary**

create table salary(

id int, employee\_id int,

amount int, pay\_date date,

primary key(id),

foreign key (employee\_id) references new\_employee(employee\_id) );

**IN:**

insert into salary values(8,1,9000,date\_format('2017/03/31','%Y-%m-%d')),

(2,2,6000,'2017/03/31'),

(3,3,1000,'2017/03/31'),

(4,1,7000,'2017/02/28'),

(5,2,6000,'2017/02/28'),

(6,3,8000,'2017/02/28');

**Table – new\_employee**

create table new\_employee(

employee\_id int, department\_id int,

primary key(employee\_id) );

**IN:**

insert into new\_employee values(1,1),(2,2),(3,2);

**Query:**

with tmp as (select date\_format(pay\_date,"%Y-%m") as pay\_month, department\_id,

avg(amount) over(partition by month(pay\_date) ) as average\_1,

avg(amount) over(partition by department\_id order by month(pay\_date) ) as average\_2

from salary s join new\_employee e on s.employee\_id = e.employee\_id)

select pay\_month,department\_id,

case

when average\_2 > average\_1 then 'higher'

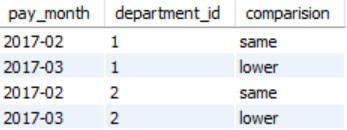
when average\_2 < average\_1 then 'lower'

else 'same'

end as comparision

from tmp

group by pay\_month,department\_id,comparision;



**Q131.**

create table if not exists Activity(

player\_id int, device\_id int,

event\_date date, games\_played int,

primary key(player\_id,event\_date) );

**IN:**

insert into Activity values(1,2,'2016-03-01',5),(1,2,'2016-05-02',6),(2,3,'2017-06-25',1),

(3,1,'2016-03-01',0),(3,4,'2018-07-03',5);

**Query:**

select install\_date as install\_dt, count(install\_date) as installs,

round( count(case when activity\_date = DATE\_ADD(install\_date, interval 1 day) then player\_id end) / COUNT(install\_date),

1) as Day1\_retention

from (select player\_id, MIN(event\_date) as install\_date,

case

when max(event\_date) = DATE\_ADD(MIN(event\_date), interval 1 day) then max(event\_date)

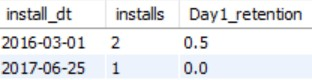
else 0

end AS activity\_date

from Activity

group by player\_id) as subquery

group by install\_date;



**Q132.**

**-- Table - Players**

create table players(

player\_id int primary key,

group\_id int) ;

**IN:**

insert into players(player\_id, group\_id) values(15,1),(25,1),(30,1),(45,1),(10,2),(35,2),(50,2),(20,3),(40,3);

**-- Table - Matches**

create table Matches(

match\_id int primary key,

first\_player int, second\_player int,

first\_score int, second\_score int);

**IN:**

insert into Matches values(1,15,45,3,0),(2,30,25,1,2),(3,30,15,2,0),(4,40,20,5,2),(5,35,50,1,1);

**Query:**

select group\_id, player\_id from

(SELECT group\_id, player as player\_id, points,

ROW\_NUMBER() OVER (PARTITION BY group\_id ORDER BY points DESC, player ASC) AS row\_num

from

(select p.group\_id, m.first\_player as player, sum(first\_score) as points from Matches m

join players p on p.player\_id = m.first\_player

group by p.group\_id, m.first\_player

union

select p.group\_id, m.second\_player as player, sum(second\_score) as points from Matches m

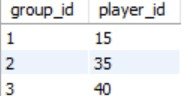
join players p on p.player\_id = m.second\_player

group by p.group\_id, m.second\_player) as tmp

) as result

where row\_num = 1

order by group\_id asc;



**Q133.**

-- Table - Student

create table student(

student\_id int primary key,

student\_name varchar(10) );

insert into student values(1,'Daniel'),(2,'Jade'),(3,'Stella'),(4,'Jonathan'),(5,'Will');

-- Table - Exam

create table exam(

exam\_id int, student\_id int,

score int,

primary key(exam\_id, student\_id) );

insert into exam values(10,1,70),(10,2,80),(10,3,90),(20,1,80),

(30,1,70),(30,3,80),(30,4,90),(40,1,60),(40,2,70),(40,4,80);

SELECT s.student\_id, s.student\_name

FROM Student s

WHERE s.student\_id IN (

SELECT DISTINCT e.student\_id

FROM Exam e

WHERE e.student\_id NOT IN (

SELECT DISTINCT student\_id

FROM Exam

WHERE (exam\_id, score) IN (

SELECT exam\_id, MIN(score)

FROM Exam

GROUP BY exam\_id

UNION

SELECT exam\_id, MAX(score)

FROM Exam

GROUP BY exam\_id

)

)

)

ORDER BY s.student\_id;



**Q134.**

**133 Question repeated**

**Q135.**

Create Table userActivity(

username varchar(6),activity varchar(10),

startDate date,endDate date);

**IN:**

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');

**Query:**

with tmp as (select \*,

row\_number() over(partition by username order by startDate) as row\_num,

count(username) over(partition by username) as tot

from userActivity)

select username, activity, startDate, endDate

from tmp

where row\_num = case

when tot > 1 then 2

when tot = 1 then 1

end



**Q136.**

**135 Question is repeated**

**Q137.**

create table employees(

id int, name varchar(10),

salary int );

**IN:**

insert into employees values(1,'Kristeen', 1420),(2, 'Ashley', 2006),(3, 'Julia', 2210),(4, 'Maria', 3000);

**Query:**

SELECT CEIL(AVG(salary) - AVG(REPLACE(CAST(salary AS char), '0', ''))) as error\_amount

FROM EMPLOYEES;



**Q138.**

create table employee(

employee\_id int,name varchar(15),

months int, salary int);

**IN:**

INSERT INTO employee 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);

**Query:**

select concat(salary, ' ', count(salary) ) as output from

(select max(salary \* months) as salary

from employee) as tmp

group by salary;



**Q139.**

**108 Question is repeated**

**Q140.**

**109 Question is repeated.**

**Q141.**

**110 Question is repeated.**

**Q142.**

**111 Question is repeated.**

**Q143.**

**116 Question is repeated.**

**Q144.**

**-- Table - student**

create table student(

id int, name varchar(8) );

**IN:**

insert into student values(1,'Ashley'),(2,'Samantha'),(3,'Julia'),(4,'Scarlet');

**-- Table - friends**

create table friends(

id int, friend\_Id int);

**IN:**

insert into friends values(1,2),(2,3),(3,4),(4,1);

**-- Table - packages**

create table packages(

id int, salary float);

**IN:**

insert into packages values(1,15.20),(2,10.06),(3,11.55),(4,12.12);

**Query:**

with b as (select f.id, f.friend\_id, p.salary from friends f

join packages p on f.friend\_Id = p.id)

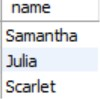
select name

from student s join b on s.id = b.id

join packages p on p.id = s.id

where p.salary < b.salary

order by b.salary



**Q145.**

**-- Table - Hackers**

create table Hackers(

hacker\_id int, name varchar(10) );

**IN:**

INSERT INTO Hackers VALUES(5580, 'Rose'),(8439, 'Angela'),(27205, 'Frank'),(52243, 'Patrick'),(52348, 'Lisa'),

(57645, 'Kimberly'),(77726, 'Bonnie'),(83082, 'Michael'),(86870, 'Todd'),(90411, 'Joe');

**-- Table - Difficulty**

create table Difficulty(

difficulty\_level int, score int );

**IN:**

INSERT INTO Difficulty VALUES(1, 20),(2, 30),(3, 40),(4, 60),(5, 80),(6, 100),(7, 120);

**-- Table - Challenges**

create table Challenges(

challenge\_id int, hacker\_id int, difficulty\_level int );

**IN:**

INSERT INTO challenges VALUES(4810, 77726, 4),(21089, 27205, 1),(66730, 52243, 6),(71055, 52243, 2);

**-- Table - Submissions**

create table Submissions(

submission\_id int, hacker\_id int,

challenge\_id int, score int );

**IN:**

INSERT INTO submissions VALUES(68628, 77726, 36566, 30),(65300, 77726, 21089, 10),(40326, 52243, 36566, 77),(8941, 27205, 4810, 4),

(83554, 77726, 66730, 30),(43353, 52243, 66730, 0),(55385, 52348, 71055, 20),(39784, 27205, 71055, 23),

(94613, 86870, 71055, 30),(45788, 52348, 36566, 0),(93058, 86870, 36566, 30),(7344, 8439, 66730, 92),

(2721, 8439, 4810, 36),(523, 5580, 71055, 4),(49105, 52348, 66730, 0),(55877, 57645, 66730, 80),

(38355, 27205, 66730, 35),(3924, 8439, 36566, 80),(97397, 90411, 66730, 100),(84162, 83082, 4810, 40),

(97431, 90411, 71055, 30);

**Query:**

with output as (select s.hacker\_id, h.name,count(\*) as tot

from submissions s left join hackers h

on s.hacker\_id = h.hacker\_id

join challenges c

on s.challenge\_id = c.challenge\_id

join difficulty d

on c.difficulty\_level = d.difficulty\_level

where s.score = d.score

group by s.hacker\_id, h.name)

select hacker\_id, name

from output

where tot > 1

order by tot desc, hacker\_id;



**Q146.**

create table projects(

Task\_ID int, Start\_Date date,

End\_Date date);

**IN:**

INSERT INTO projects VALUES(1, '2015-10-01', '2015-10-02'),(2, '2015-10-02', '2015-10-03'),

(3, '2015-10-3', '2015-10-4'),(4, '2015-10-13', '2015-10-14'),(5, '2015-10-14', '2015-10-15'),

(6, '2015-10-28', '2015-10-29'),(7, '2015-10-30', '2015-10-31');

**Query:**

SELECT Start\_Date, min(End\_Date)

FROM

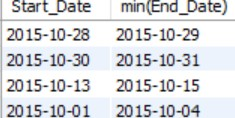
(SELECT Start\_Date FROM Projects WHERE Start\_Date NOT IN (SELECT End\_Date FROM Projects)) a ,

(SELECT End\_Date FROM Projects WHERE End\_Date NOT IN (SELECT Start\_Date FROM Projects)) b

WHERE Start\_Date < End\_Date

GROUP BY Start\_Date

ORDER BY DATEDIFF(min(End\_Date), Start\_Date) ASC, Start\_Date ASC;



**Q147.**

create table transactions(

user\_id int, amount float,

transaction\_date timestamp);

**IN:**

INSERT INTO transactions VALUES(1, '9.99', '2022-08-01 10:00:00'),(1, '55', '2022-08-17 10:00:00'),

(2, '149.5', '2022-08-05 10:00:00'),(2, '4.89', '2022-08-06 10:00:00'),(2, '34', '2022-08-07 10:00:00');

**Query:**

SELECT distinct user\_id

FROM

(SELECT \*,

DATEDIFF(LEAD(transaction\_date) OVER(PARTITION BY user\_id ORDER BY transaction\_date), transaction\_date) AS diff1,

DATEDIFF(transaction\_date, LAG(transaction\_date) OVER(PARTITION BY user\_id ORDER BY transaction\_date)) AS diff2

FROM transactions) AS trx\_date\_diff

WHERE trx\_date\_diff.diff1 = 1 AND trx\_date\_diff.diff2 = 1;



**Q148.**

create table payments(

payer\_id int, recipient\_id int,

amount int);

**IN:**

INSERT INTO payments VALUES(101, 201, 30),(201, 101, 10),(101, 301, 20),(301, 101, 80),(201, 301, 70);

**Query:**

WITH T1 AS (SELECT PAYER\_ID, RECIPIENT\_ID FROM PAYMENTS

INTERSECT

SELECT RECIPIENT\_ID, PAYER\_ID FROM PAYMENTS)

SELECT round(COUNT(PAYER\_ID)/2,0) AS UNIQUE\_RELATIONSHIPS

FROM T1;



**Q149.**

create table user\_transaction(

transaction\_id int, user\_id int,

spend float, transaction\_date timestamp);

**IN:**

INSERT INTO user\_transaction VALUES(759274, 111, 49.50, '2022-02-03 00:00:00'),

(850371, 111, 51.00, '2022-03-15 00:00:00'),(615348, 145, 36.30, '2022-03-22 00:00:00'),

(137424, 156, 151.00, '2022-04-04 00:00:00'),(248475, 156, 87.00, '2022-02-16 00:00:00');

**Query:**

with tmp as (select user\_id,spend,

dense\_rank() over(partition by user\_id order by transaction\_date) as transaction\_count

from user\_transaction)

select count(user\_id) as users from tmp

where spend >= 50.00

and transaction\_count = 1;



**Q150.**

create table measurements(

measurement\_id int, measurement\_value float,

measurement\_time datetime);

**IN:**

INSERT INTO measurements VALUES(131233, 1109.51, '2022-07-10 09:00:00'),

(135211, 1662.74, '2022-07-10 11:00:00'),(523542, 1246.24, '2022-07-10 13:15:00'),

(143562, 1124.50, '2022-07-11 15:00:00'),(346462, 1234.14, '2022-07-11 16:45:00');

**Query:**

with cte as (select measurement\_value, date(measurement\_time) as measurement\_time,

row\_number() over(partition by day(measurement\_time) order by measurement\_time) as num\_type

from measurements)

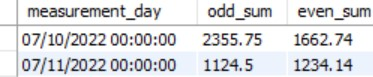
select DATE\_FORMAT(STR\_TO\_DATE(measurement\_time, '%Y-%m-%d'), '%m/%d/%Y 00:00:00') as measurement\_day,

round(sum(case when num\_type % 2 <> 0 then measurement\_value end), 2) as odd\_sum,

round(sum(case when num\_type % 2 = 0 then measurement\_value end), 2) as even\_sum

from cte

group by measurement\_time;



**Q151.**

**147 Qusetion is repeated**

**Q152.**

CREATE TABLE rental\_amenities(

rental\_id INT,amenity VARCHAR(10));

**IN:**

INSERT INTO rental\_amenities VALUES(123, 'pool'),(123, 'kitchen'),(234, 'hot tub'),(234, 'fireplace'),

(345, 'kitchen'),(345, 'pool'),(456, 'pool');

**Query:**

WITH rental\_amenities AS(SELECT rental\_id,

GROUP\_CONCAT(amenity ORDER BY amenity) AS amenities

FROM rental\_amenities

GROUP BY rental\_id)

SELECT count(distinct amenities) AS matching\_airbnb

FROM rental\_amenities

GROUP BY amenities

HAVING COUNT(DISTINCT rental\_id) > 1;



**Q153.**

create table ad\_campaigns(

campaign\_id int, spend int,

revenue float, advertiser\_id int);

**IN:**

INSERT INTO ad\_campaigns VALUES(1, 5000, 7500, 3),(2, 1000, 900, 1),(3, 3000, 12000, 2),

(4, 500, 2000, 4),(5, 100, 400, 4);

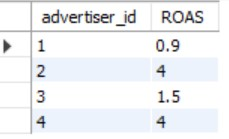
**Query:**

select advertiser\_id, round(sum(revenue)/sum(spend), 2) as ROAS

from ad\_campaigns

group by advertiser\_id

order by advertiser\_id;



**Q154.**

create table employee\_pay(

employee\_id int, salary int,

title varchar(20));

**IN:**

INSERT INTO employee\_pay VALUES(101, 80000, 'Data Analyst'),(102, 90000, 'Data Analyst'),

(103, 100000, 'Data Analyst'),(104, 30000, 'Data Analyst'),(105, 120000, 'Data Scientist'),

(106, 100000, 'Data Scientist'),(107, 80000, 'Data Scientist'),(108, 310000, 'Data Scientist');

**Query:**

with cte as (select employee\_id, salary,

avg(salary) over(partition by title) as avg\_salary

from employee\_pay)

select employee\_id, salary,

case

when salary > avg\_salary \* 2 then 'Overpaid'

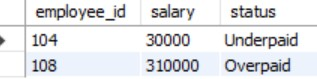
when salary < avg\_salary / 2 then 'Underpaid'

end as status

from cte

where salary > avg\_salary \* 2

or salary < avg\_salary / 2



**Q155.**

**148 Question is repeated**

**Q156.**

create table purchases(

user\_id int, product\_id int,

quantity int, purchase\_date datetime);

**IN:**

INSERT INTO purchases VALUES(536, 3223, 6, '2022-01-11 12:33:44'),(827, 3585, 35, '2022-02-20 14:05:26'),

(536, 3223, 5, '2022-03-02 09:33:28'),(536, 1435, 10, '2022-03-2 08:40:00'),

(827, 2452, 45, '2022-04-09 00:00:00');

**Query:**

SELECT COUNT(DISTINCT user\_id) AS repeat\_purchasers

FROM( SELECT user\_id FROM purchases

GROUP BY user\_id, product\_id

HAVING COUNT(DISTINCT DATE(purchase\_date)) > 1

) repeat\_purchase\_product ;



**Q157.**

create table transactions\_new(

transaction\_id int, type enum('deposit', 'withdrawal'),

amount decimal(10,2), transaction\_date timestamp);

**IN:**

INSERT INTO transactions\_new VALUES(19153, 'deposit', 65.90, '2022-07-10 10:00:00'),

(53151, 'deposit', 178.55, '2022-07-08 10:00:00'),(29776, 'withdrawal', 25.90, '2022-07-08 10:00:00'),

(16461, 'withdrawal', 45.99, '2022-07-08 10:00:00'),(77134, 'deposit', 32.60, '2022-07-10 10:00:00');

**Query:**

WITH daily\_balance AS (SELECT

DATE\_FORMAT(transaction\_date, '%Y-%m') AS transaction\_month,

DATE\_FORMAT(transaction\_date, '%m/%d/%Y') AS transaction\_date,

SUM(CASE

WHEN type = 'deposit' THEN amount

WHEN type = 'withdrawal' THEN amount \* (-1)

END) AS balance

FROM transactions\_new

GROUP BY

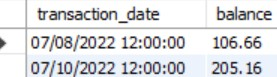
DATE\_FORMAT(transaction\_date, '%m/%d/%Y'),

DATE\_FORMAT(transaction\_date, '%Y-%m'))

SELECT CONCAT(transaction\_date , ' 12:00:00') AS transaction\_date,

SUM(balance) OVER( PARTITION BY transaction\_month ORDER BY transaction\_date) AS balance

FROM daily\_balance;



**Q158.**

create table product\_spend(

category varchar(15), product varchar(20),

user\_id int, spend decimal(10,2),

transaction\_date timestamp);

**IN:**

INSERT INTO product\_spend VALUES('appliance', 'refrigerator', 165, 246.00, '2021-12-26 12:00:00'),

('appliance', 'refrigerator', 123, 299.99, '2022-03-02 12:00:00'),

('appliance', 'washing machine', 123, 219.80, '2022-03-02 12:00:00'),

('electronics', 'vacuum', 178, 152.00, '2022-04-05 12:00:00'),

('electronics', 'wireless headset', 156, 249.90, '2022-07-08 12:00:00'),

('electronics', 'vacuum', 145, 189.00, '2022-07-15 12:00:00');

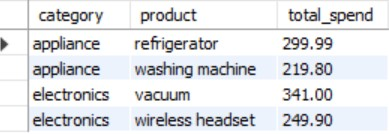
**Query:**

select category, product, sum(spend) as total\_spend

from product\_spend

where year(transaction\_date) = 2022

group by category, product;



**Q159.**

create table users(

user\_id int, signup\_date datetime,

last\_login datetime);

**IN:**

INSERT INTO users VALUES(1001, '2022-06-01 12:00:00', '2022-07-05 12:00:00'),

(1002, '2022-06-03 12:00:00', '2022-06-15 12:00:00'),

(1004, '2022-06-02 12:00:00', '2022-06-15 12:00:00'),

(1006, '2022-06-15 12:00:00', '2022-06-27 12:00:00'),

(1012, '2022-06-16 12:00:00', '2022-07-22 12:00:00');

**Query:**

WITH users\_churn\_weekly AS (

SELECT \*,

CASE

WHEN DATEDIFF(last\_login, signup\_date) < 28 THEN 'yes'

ELSE 'no'

END AS churn,

DATEDIFF(last\_login, signup\_date) as diff,

WEEK(signup\_date) - WEEK('2022-05-28') AS week\_no

FROM users

WHERE DATE\_FORMAT(signup\_date, '%Y-%m') = '2022-06')

SELECT week\_no,

ROUND(COUNT(CASE WHEN churn = 'yes' THEN user\_id END) \* 100.00 /COUNT(\*), 2) AS churn\_rate

FROM users\_churn\_weekly

GROUP BY week\_no;

