SQL

Q.1

Ans. select *

from CITY

WHERE COUNTRYCODE = 'USA' AND POPULATION > 100000

Q.2

Ans. select Name

from CITY

where CountryCode = 'USA' and Population >120000

Q.3

select * from city

Q.4

select * from city where id = 1661

Q.5

select * from City where countrycode = 'JPN'

Q.6

select name from city where countrycode = 'JPN'

Q.7

select city,state from station

Q.8

SELECT DISTINCT CITY FROM STATION WHERE ID%2=0

Q.9

select count(city)-count(distinct city) from station

Q.10

select city, length(city) from station order by 2 asc, city asc limit 1;

select city, length(city) from station order by 2 desc, city asc limit 1;

Q.11

select distinct city

from station

WHERE LEFT(CITY,1) IN ('A','E','I','O','U')

Q.12

select distinct city

from station where right(city,1) in ('A','E','I','O','U')

Q.13

select distinct city

from station

where not left(city,1) in ('a','e','i','o','u')

Q.14

select distinct city

from station

where not RIGHT(city,1) in ('a','e','i','o','u')

Q.15

select distinct city

from station

where left(city,1) in ('a','e','i','o','u') OR RIGHT(CITY,1) IN ('a','e','i','o','u')

Q.16

```
select distinct city
from station
where not left(city,1) in ('a','e','i','o','u') OR NOT RIGHT(CITY,1) IN ('a','e','i','o','u')
```

Q.17

```sql
with cte as
(
    Select *
    from Sales
    group by product_id
    having sum(if(sale_date between '2019-01-01' and '2019-03-31',1,0)) = sum(if(sale_date,1,0))
)
select s.product_id, product_name
from cte s
join Product p
on s.product_id = p.product_id
```

Q.18

```sql
select distinct author_id as id
from Views
where author_id = viewer_id
order by author_id
```

Q.19

```sql
select round(sum(case when order_date = customer_pref_delivery_date then 1 else 0 end)/count(delivery_id)*100,2) as immediate_percentage
from Delivery
```

Q.20

```sql
Select ad_id, round(if(clicks + views = 0, 0, clicks / (clicks + views) * 100),
2) as ctr
from (
 select ad_id, sum(if(action='Clicked', 1, 0)) as clicks,
sum(if(action='Viewed', 1, 0)) as views
 from Ads
 group by ad_id
) as a
order by ctr desc, ad_id asc;
```

q.21

```sql
Select employee_id, team_size
from Employee as e join (select team_id, count(*) as team_size from employee
group by team_id) as t
on e.team_id = t.team_id;
```

Q.22

```sql
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 as w
 left join
 Countries as c
 on c.country_id = w.country_id
where day between '2019-11-01' and '2019-11-30'
group by w.country_id
```

Q.23

```sql
select u.product_id, round(sum(units*price)/sum(units), 2) as average_price
from UnitsSold u inner join Prices p
```

```
on u.product_id = p.product_id
and u.purchase_date between p.start_date and p.end_date
group by u.product_id
```

Q.24

```
select player_id, min(event_date) as first_login
from Activity
group by player_id
order by player_id
```

Q.25

```
select player_id, device_id
from Activity
where (player_id, event_date) in (
 select player_id, min(event_date)
 from Activity
 group by player_id
)
```

Q.26

```
Select product_name, sum(unit) as unit
from Orders as o left join Products as p on o.product_id = p.product_id
where order_date between '2020-02-01' and '2020-02-29'
group by o.product_id
having sum(unit) >= 100;
```

Q.27

```
Select *
from Users
where mail REGEXP '^[A-Za-z][A-Za-z\_\.\-]*@leetcode.com$'
```

Q.28

```
select customer_id, name
from
(
    select o.customer_id, c.name,
        sum(case when left(o.order_date,7) = '2020-06' then p.price * o.quantity end) as JuneSpend,
        sum(case when left(o.order_date,7) = '2020-07' then p.price * o.quantity end) as JulySpend
    from Orders o
    left join Customers c on o.customer_id = c.customer_id
    left join Product p on o.product_id = p.product_id
    group by o.customer_id
    having JuneSpend >= 100 and JulySpend >= 100
) as temp
```

Q.29

```
select distinct title
from Content
join TVProgram using(content_id)
where kids_content = 'Y'
    and content_type = 'Movies'
    and (month(program_date), year(program_date)) = (6, 2020)
```

Q.30

```
select q.id, q.year, ifnull(n.npv,0) as npv
from queries as q
left join npv as n
on (q.id, q.year) = (n.id, n.year)
```

Q.32

```
select unique_id, name
from Employees emp left join EmployeeUNI en
on emp.id = en.id
```

Q.33

```
select name,sum(case when distance > 0 then distance else 0 end) as travelled_distance
from Users u
left join Rides r
on u.id = r.user_id
group by u.id
order by travelled_distance desc,name asc
```

Q.34

```
SELECT
    t2.product_name,
    t1.unit
FROM (
SELECT
    product_id,
    SUM(unit) as unit
FROM Orders
WHERE DATE_FORMAT(order_date, '%Y-%m') = '2020-02'
GROUP BY 1
HAVING SUM(unit) >= 100
    ) as t1 JOIN Products as t2
ON t1.product_id = t2.product_id
```

Q.35

```
create database test

use test

create TABLE Movies(
movie_id int PRIMARY KEY,
title varchar(30)
);
create TABLE Users(
user_id int PRIMARY KEY,
name varchar(40)
);

create TABLE MovieRating(
movie_id int,
user_id int,
rating int,
created_at date
);
INSERT INTO Movies VALUES(1,'Avengers'),
(2,'Frozen 2'),
(3,'Joker');
INSERT INTO Users VALUES(1,'Daniel'),
(2,'Monica'),
(3,'Maria'),
(4,'James');
INSERT INTO MovieRating VALUES(1, 1, 3 ,'2020-01-12'),
(1 ,2 ,4, '2020-02-11'),
(1 ,3 ,2, '2020-02-12'),
(1 ,4 ,1 ,'2020-01-01'),
(2 ,1 ,5, '2020-02-17'),
(2 ,2, 2,'2020-02-01'),
(2,3,2,'2020-03-01'),
(3, 1, 3,'2020-02-22'),
(3,2,4,'2020-02-25');

select name from(select u.name as name, count(*) as counts
from MovieRating m join Users u on m.user_id = u.user_id
GROUP BY m.user_id
order by counts desc, name asc limit 1) f
union
select movie_title from (
select m1.title as movie_title,avg(m.rating) as average_rating
```

```
from MovieRating m join Movies m1 on m.movie_id = m1.movie_id
where m.created_at BETWEEN '2020-02-01' and '2020-02-28'
group by m.movie_id
order by average_rating desc, movie_title asc limit 1) f1
```

Q.36

```
create database test

use test

CREATE TABLE Users(
id int PRIMARY KEY,
name varchar(30)
);
CREATE TABLE Rides(
id int PRIMARY KEY,
user_id int,
distance int
);
insert into Users VALUES(1,'Alice'),
(2,'Bob'),
(3,'Alex'),
(4,'Donald'),
(7,'Lee'),
(13,'Jonathan'),
(19,'Elvis');
insert into Rides VALUES
(1,1,120),
(2,2,317),
(3,3,222),
(4,7,100),
(5,13,312),
(6,19,50),
(7,7,120),
(8,19,400),
(9,7,230);

SELECT u.name as name,sum(ifnull(r.distance,0)) as travelled_distance
from Rides r right join Users u on r.user_id = u.id
group by name
order by travelled_distance desc,name asc
```

Q.37

```
select unique_id,name
from Employees e left join EmployeeUNI eu on e.id = eu.id
order by eu.id
```

Q.38

```
CREATE TABLE Departments(
id int PRIMARY KEY,
name varchar(30)
);
CREATE TABLE Students(
id int PRIMARY KEY,
name varchar(30),
department_id int
);
INSERT INTO Departments VALUES(1,'Electrical Engineering'),
(7,'Computer Engineering'),
(13,'Business Administration');
INSERT INTO Students VALUES(23,'Alice',1),
(1,'Bob',7),
(5,'Jennifer',13),
(2,'John',14),
(4,'Jasmine',77),
(3,'Steve',74),
```

```
(6,'Luis',1),
(8,'Jonathan',7),
(7,'Daiana',33),
(11,'Madelynn',1);
SELECT id, name
FROM Students
WHERE department_id not in (SELECT id from Departments);

select s.id, s.name
from Students s
left join Departments d
on s.department_id = d.id
where d.id is null
```

Q.39

```
CREATE TABLE Calls(
from_id int,
to_id int,
duration int
);
INSERT INTO Calls VALUES(1,2,59),
(2,1,11),
(1,3,20),
(3,4,100),
(3,4,200),
(3,4,200),
(4,3,499);

SELECT LEAST(from_id,to_id) as person1,GREATEST(From_id,to_id) as person2,count(*) as call_count, sum(duration) as total_duration
from Calls group by 1,2
```

Q.40

```
CREATE TABLE Pricess(
product_id int,
start_date date,
end_date date,
price int,
constraint pk PRIMARY key (product_id,start_date,end_date)
);
CREATE TABLE UnitsSold(
product_id int,
purchase_date date,
units int
);
insert into Pricess VALUES(1,'2019-02-17','2019-02-28',5),
(1,'2019-03-01','2019-03-22',20),
(2,'2019-02-01','2019-02-20',15),
(2,'2019-02-21','2019-03-31',30);
INSERT INTO UnitsSold VALUES(1,'2019-02-25',100),
(1,'2019-03-01',15),
(2,'2019-02-10',200),
(2,'2019-03-22',30);

select u.product_id, round(sum(units*price)/sum(units),2) as aveage_price
from Pricess p join UnitsSold u on p.product_id = u.product_id
where purchase_date between start_date and end_date
group by u.product_id
```

Q.41

```
CREATE TABLE Warehouse(
name varchar(128),
product_id int,
units int,
CONSTRAINT PK PRIMARY KEY(name,product_id)
);
CREATE TABLE Products(
product_id int PRIMARY KEY,
product_name varchar(128),
Width int,
Length int,
Height int
);
INSERT INTO Warehouse VALUES ('LCHouse1',1,1),
```

```
('LCHouse1',2,10),
('LCHouse1',3,5),
('LCHouse2', 1,2),
('LCHouse2',2,2),
('LCHouse3',4,1);
INSERT INTO Products VALUES(1 ,'LC-TV', 5 ,50, 40),
(2, 'LC-KeyChain', 5, 5, 5),
(3,'LC-Phone', 2, 10 ,10),
(4 ,'LC-T-Shirt',4,10 ,20);

with cte as(select product_id,(Width*Length*Height) as total_vol
from Products)
select name as warehouse_name,sum(total_vol*units) as volume
from Warehouse w join cte p on w.product_id = p.product_id
group by warehouse_name
```

Q.42

```
CREATE Table Sales(
sale_date date,
fruit enum("apples","oranges"),
sold_num int,
CONSTRAINT PK PRIMARY KEY(sale_date, fruit)
);
INSERT INTO Sales VALUES ('2020-05-01','apples', 10),
('2020-05-01', 'oranges',8),
('2020-05-02','apples', 15),
('2020-05-02','oranges',15),
('2020-05-03','apples',20),
('2020-05-03', 'oranges',0),
('2020-05-04','apples',15),
('2020-05-04','oranges',16);

select sale_date,sum(case when fruit = "apples" then sold_num
                         when fruit = "oranges" then -sold_num end) as diff
from Sales group by sale_date
```

Q.43

```
CREATE TABLE Activity(
player_id int,
device_id int,
event_date date,
games_played int,
CONSTRAINT PK PRIMARY KEY(player_id, event_date)
);
INSERT INTO Activity VALUES(1,2,'2016-03-01',5),
(1,2,'2016-03-02',6),
(2,3,'2017-06-25',1),
(3,1,'2016-03-02',0),
(3,4,'2018-07-03',5);



with cte as (
    select player_id,min(event_Date) as first_login from Activity group by player_id
)
select  round(sum(case when datediff(event_Date,first_login)=1 then 1 else 0 end)/count(DISTINCT c.player_id),2) as fraction
from Activity a join cte c on a.player_id = c.player_id
```

Q.44

```
Create TABLE Employee(
id int PRIMARY key,
name varchar(30),
department varchar(30),
managerId int
);
INSERT into Employee VALUES(101,'John','A',null),
(102,'Dan','A',101),
(103,'James','A',101),
(104,'Amy','A',101),
(105, 'Anne','A', 101),
(106,'Ron','B', 101);

select name from Employee where id in(select managerId from Employee GROUP BY managerId having count(*)>=5)
```

Q.45

```
create TABLE Student
(
    student_id int,
    student_name varchar(30),
    gender varchar(30),
    dept_id int,
    CONSTRAINT  PRIMARY KEY(student_id),
    CONSTRAINT  FOREIGN KEY(dept_id) REFERENCES Department(dept_id)
);

CREATE TABLE Department(
dept_id int PRIMARY KEY,
dept_name varchar(30)
);
INSERT INTO Student values(1,'Jack','M',1),
(2,'Jane','F',1),
(3,'Mark','M',2);
INSERT INTO Department VALUES(1,'Engineering'),
(2,'Science'),
(3,'Law');

with cte as (
    select dept_id,count(*) as student_number from Student GROUP BY  dept_id
)
select dept_name,ifnull(student_number,0) from Department d left join cte s on d.dept_id=s.dept_id
```

Q.46

```
Create table Customer(
customer_id int,
product_key int,
constraint Foreign Key(product_key) REFERENCES Product(product_key)
);
create table Product(
product_key int PRIMARY KEY
);
INSERT INTO Customer VALUES( 1,5),
(2,6),
(3,5),
(3,6),
(1,6);
INSERT INTO Product VALUES(5),(6);

select c.customer_id
from Customer c join Product p on c.product_key = p.product_key
group by c.customer_id HAVING count(*)=2
```

Q.47

```
CREATE TABLE Project(
project_id int,
employee_id int,
constraint pk PRIMARY KEY(project_id, employee_id),
constraint fk FOREIGN KEY(employee_id) REFERENCES Employees(employee_id)
);
CREATE TABLE Employees(
employee_id int PRIMARY KEY,
name varchar(40),
experience_years int
);
INSERT INTO Project VALUES(1,1),
(1,2),
(1,3),
(2,1),
(2,4);
INSERT INTO Employees VALUES(1,'Khaled',3),
(2,'Ali',2),
(3,'John',3),
(4,'Doe',2);



with cte as(select project_id,emp.employee_id,dense_rank() over(PARTITION BY project_id order by experience_years desc) as rank1
from Project p join Employees emp on p.employee_id = emp.employee_id
)
```

```sql
select project_id,employee_id
from cte
where rank1 =1;
```

Q.48

```sql
create table Books(
book_id int PRIMARY KEY,
name varchar(30),
available_from date
);
CREATE TABLE Orders(
order_id int PRIMARY KEY,
book_id int,
quantity int,
dispatch_date date,
CONSTRAINT FK FOREIGN KEY(book_id)REFERENCES Books(book_id)
);
INSERT INTO Books VALUES(1,"Kalila And Demna",'2010-01-01'),
(2 ,"28 Letters",'2012-05-12'),(3,"The Hobbit",'2019-06-10'),
(4 ,"13 Reasons Why",'2019-06-01'),(5,"The Hunger Games",'2008-09-21' );
INSERT INTO Orders VALUES(1,1,2,'2018-07-26'),(2,1,1,'2018-11-05'),(3,3,8,'2019-06-11'),
(4,4,6,'2019-06-05'),
(5,4,5,'2019-06-20'),
(6,5,9,'2009-02-02'),
(7,5,8,'2010-04-13');

select book_id,name
from Books
where book_id not in (select book_id from Orders
where dispatch_date >= '2018-06-23' and dispatch_date <= '2019-06-22'
group by book_id
having sum(quantity)>=10)
and available_from < '2019-05-23'
```

Q.49

```sql
CREATE TABLE Enrollments(
student_id int,
course_id int,
grade int,
CONSTRAINT PK PRIMARY KEY(student_id, course_id)
);
INSERT INTO Enrollments VALUES(2,2,95),
(2,3,95),
(1,1,90),
(1,2,99),
(3,1,80),
(3,2,75),
(3,3,82);

with cte as(SELECT student_id,course_id,grade,dense_rank() over(partition by student_id order by grade desc,course_id) as rank1
FROM Enrollments)

select student_id,course_id,grade
from cte where rank1=1
```

Q.50

```sql
CREATE TABLE Players(
player_id int PRIMARY KEY,
group_id varchar(30)
);
CREATE TABLE Matches(
match_id int primary KEY,
first_player int,
second_player int,
first_score int,
second_score int
);
insert into Players VALUES(15,1),
(25,1),
```

```
(30,1),
(45,1),
(10,2),
(35,2),
(50,2),
(20,3),
(40,3);
insert into Matches VALUES(1,15,45,3,0),
(2,30,25,1,2),(3,30,15,1,2),
(4,40,20,5,2),(5,35,50,1,1);




with players_score as
(select first_player as player_id,first_score as score from Matches
union all
select second_player as player_id,second_score as score from Matches),
final_score as(
select p.group_id,ps.player_id,sum(score) as max_score
from players_score ps inner join Players p on ps.player_id = p.player_id
group by p.group_id,ps.player_id),
final_ranking as(
select
*,
dense_rank() over(partition by group_id order by max_score desc,player_id) as rank1
from final_score)

select group_id,player_id
from final_ranking
where rank1=1
```

Q.51

```
Create TABLE world(
name varchar(30) PRIMARY KEY,
continent varchar(30),
area bigint,
population bigint,
gdp bigint
);

insert into world values('Afghanistan', 'Asia', 652230, 25500100, 20343000000),
('Albania', 'Europe', 28748, 2831741, 12960000000),
('Algeria', 'Africa', 2381741, 37100000, 188681000000),
('Andorra', 'Europe', 468, 78115, 3712000000),
('Angola', 'Africa', 1246700, 20609294, 100990000000);

SELECT name,population,area FROM world where
area >=3000000 or population >= 25000000;
```

Q.52

```
CREATE TABLE customer(
id int PRIMARY KEY,
name varchar(30),
referee_id int
);
insert into customer values(1, "Will", null);
insert into customer values(2, "Jane", null);
insert into customer values(3, "Alex", 2);
insert into customer values(4, "Bill", null);
insert into customer values(5, "Zack", 1);
insert into customer values(6, "Mark", 2);

select name from customer where referee_id != 2 or referee_id is null
```

Q.53

```
create TABLE Customers(
id int PRIMARY KEY,
name varchar(30)
);
insert into Customers values(1,"Joe"),(2, "Henry"),(3, "Sam"),(4,"Max");
create TABLE Orders(
id int PRIMARY KEY,
```

```
customerId INT
);
insert into Orders VALUES(1, 3),(2, 1)

select name as Customers
from Customers c left join Orders o
on c.id = o.customerId
where o.customerId is null
```

Q.54

```
Create table Employee(
employee_id int,
team_id int
);
INSERT INTO Employee values(1,8),
(2,8),(3,8),
(4,7),
(5,9),
(6,9);

select employee_id,count(*) over(partition by team_id) as team_size
from Employee order by employee_id
```

Q.55

```
CREATE TABLE person(
id int PRIMARY KEY,
name varchar(30),
phone_number varchar(30)
);
CREATE TABLE country(
name varchar(30),
country_code varchar(30) PRIMARY KEY
);
CREATE TABLE calls(
caller_id int,
callee_id int,
duration int
);
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");
insert into country values("Peru", '051'),("Israel", '972'),("Morocco",
'212'),("Germany", '049'),("Ethiopia", '251');
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);



select ct.name
from person p join calls c on p.id = c.caller_id or p.id=c.callee_id
join country ct on left(phone_number,3) = ct.country_code
group by ct.name
having avg(duration) > (select avg(duration) as world_wide_duration from calls)
```

Q.56

```
CREATE TABLE Activity(
player_id int,
device_id int,
event_date date,
games_played int
);
INSERT INTO Activity VALUES(1,2,'2016-03-01',5),(1,2,'2016-05-02',6),(2,3,'2017-06-25',1),
(3,1,'2016-03-02',0),(3,4,'2018-07-03',5);

select player_id,device_id
from Activity
where (player_id,event_date) in (select player_id,min(event_Date)
from Activity
group by player_id)
```

Q.57

```
create table Orders(
order_number int PRIMARY key,
```

```
customer_number int
);
INSERT into Orders VALUES(1,1),(2,2),(3,3),(4,3);

with cte AS(
    select customer_number,count(*) as count1
    from Orders
    group by customer_number
    order by count1 desc limit 1
)
select customer_number from cte
```

Q.58

```
CREATE TABLE Cinema(
seat_id int PRIMARY KEY AUTO_INCREMENT,
free bool
);
INSERT INTO Cinema VALUES(1,1),
(2,0),
(3,1),
(4,1),
(5,1);

SELECT distinct c1.seat_id
from Cinema c1 join Cinema c2 on abs(c1.seat_id - c2.seat_id)= 1 and c1.free = 1 and c2.free =1
order by c1.seat_id
```

Q.59

```
create TABLE SalesPerson(
sales_id int PRIMARY KEY,
name varchar(30),
salary int,
commission_rate int,
hire_date date
);
CREATE table Company(
com_id int PRIMARY KEY,
name varchar(30),
city varchar(30)
);
drop table Orders
create Table Orders(
order_id int PRIMARY KEY,
order_date date,
com_id int,
sales_id int,
amount int,
constraint fk FOREIGN KEY(com_id) REFERENCES Company(com_id)
);
insert into SalesPerson values(1, "John", 100000, 6,
STR_TO_DATE("4/1/2006","%m/%d/%Y")),(2, "Amy ", 12000,
5, STR_TO_DATE("5/1/2010","%m/%d/%Y")),(3, "Mark", 65000, 12,
STR_TO_DATE("12/25/2008","%m/%d/%Y")),
(4, "Pam ", 25000, 25, STR_TO_DATE("1/1/2005","%m/%d/%Y")),(5, "Alex", 5000,
10, STR_TO_DATE("2/3/2007" ,"%m/%d/%Y"));
insert into Company values(1, "RED", "Boston"),(2, "ORANGE", "New York"),(3,
"YELLOW", "Boston"),(4, "GREEN", "Austin");
insert into Orders values(1, STR_TO_DATE("1/1/2014","%m/%d/%Y"), 3, 4,
10000),(2, STR_TO_DATE("2/1/2014","%m/%d/%Y"), 4, 5, 5000),
(3,STR_TO_DATE("3/1/2014","%m/%d/%Y") , 1, 1, 50000),(4,
STR_TO_DATE("4/1/2014","%m/%d/%Y"), 1, 4, 25000);

select name from SalesPerson where sales_id not in
(select o.sales_id
from Orders o join Company c on o.com_id = c.com_id
join SalesPerson s on o.sales_id = s.sales_id
where c.name = "RED")
```

Q.60

```
create TABLE Triangle (
x int,
y int,
z int,
```

```
constraint pk PRIMARY KEY(x,y,z)
);
INSERT INTO Triangle values(13,15,30),(10,20,15);

select x,y,z,if(x+y>z and y+z>x and x+z>y,'YES','NO') as triangle
from Triangle
```

Q.61

```
create table Point(
x int
);
insert into Point values (-1),(0),(2);

select min(distance) as shortest from(select abs(p1.x - p2.x) as distance
from Point p1 cross join Point p2
where p1.x != p2.x) tmp
```

Q.62

```
CREATE TABLE ActorDirector(
actor_id int,
director_id int,
timestamp int primary key
);
insert into
ActorDirector values(1,1,0),(1,1,1),(1,1,2),(1,2,3),(1,2,4),(2,1,5),(2,1,6);

select actor_id,director_id
from ActorDirector
group by actor_id,director_id
having count(*)>=3
```

Q.63

```
create Table Sales(
sale_id int,
product_id int,
year int,
quantity int,
price int,
constraint pk primary key(sale_id,year),
constraint  foreign key(product_id) references Product(product_id)
);
create table Product(
product_id int primary key,
product_name varchar(30)
);
insert into Product values(100,"Nokia"),(200,"Apple"),(300,"Samsung");
insert into Sales values(1,100,2008,10,5000),
(2,100,2009,12,5000),
(7,200,2011,15,9000);



select product_name,year,price
from Sales s join Product p on s.product_id = p.product_id
```

Q.64

```
create Table Project(
project_id int,
employee_id int,
constraint pk primary key(project_id, employee_id)
);
INSERT INTO Project VALUES(1,1),(1,2),(1,3),(2,1),(2,4);
CREATE TABLE Employee(
employee_id int PRIMARY KEY,
name varchar(30),
experience_years int
);
insert into Employee
VALUES(1,'Khaled',3),(2,'Ali',2),(3,'John',1),(4,'Doe',2);



Select p.project_id,round(avg(experience_years),2)
from Project p inner join Employee e on p.employee_id = e.employee_id
group by p.project_id
```

Q.65

```
select seller_id from Sales group by seller_id
having sum(price)=(select max(price) FROM Sales );
```

Q.66

```
cREATE Table Product(
product_id int PRIMARY KEY,
product_name varchar(30),
unit_price int
);
drop table Sales
CREATE Table Sales(
seller_id int,
product_id int,
buyer_id int,
sale_date date,
quantity int,
price int,
constraint  FOREIGN KEY(product_id) REFERENCES Product(product_id)
);
insert into Product values (1,'S8',1000),(2,'G4',800),(3,'iPhone',1400);
insert into Sales values (1,1,1,'2019-01-21',2,2000),(1,2,2,'2019-02-17',1,800),(2,1,3,'2019-06-02',1,800),(3,3,3,'2019-05-13',2,2800);

with cte as(SELECT *
from Sales
where buyer_id not in(select product_id from Product where product_name = "iPhone"))

select buyer_id from cte c JOIN Product p on c.product_id = p.product_id where product_name = 'S8'
#another way
select s.buyer_id
from Product p
join Sales s
on p.product_id=s.product_id
group by buyer_id
having sum(p.product_name='S8') >=1 and sum(p.product_name = 'iPhone') =0 ;
```

Q.67

```
create Table Customer(
customer_id int,
name varchar(30),
visited_on date,
amount int,
constraint pk PRIMARY KEY(customer_id, visited_on)
);
INSERT INTO Customer VALUES(1,'Jhon','2019-01-01',100),(2,'Daniel','2019-01-02',110),
(3,'Jade','2019-01-03',120),(4,'Khaled','2019-01-04',130),(5,'Winston','2019-01-05',110),
(6,'Elvis','2019-01-06',140),(7,'Anna','2019-01-07',150),(8,'Maria','2019-01-08',80),
(9,'Jaze','2019-01-09',110),(1,'Jhon','2019-01-10',130),(3,'Jade','2019-01-10',150);



select c1.visited_on,sum(c2.amount) as amount,round(avg(c2.amount),2) as average_amount
from (select visited_on,sum(amount) as amount
from Customer group by visited_on) c1
join (select visited_on,sum(amount) as amount
from Customer group by visited_on) c2
on datediff(c1.visited_on,c2.visited_on) BETWEEN 0 and 6
group by c1.visited_on
having count(c2.amount) = 7
order by c1.visited_on
```

Q.68

```
cREATE TABLE Scores(
player_name varchar(30),
gender varchar(30),
day date,
score_points int,
constraint pk PRIMARY KEY(gender, day)
);
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);

select gender,day,
sum(score_points) over(partition by gender order by gender, day range between unbounded PRECEDING and current row) as total
from Scores
```

Q.69

```
create Table Logs(
log_id int
);
insert into Logs VALUES(1),(2),(3),(7),(8),(10)

with cte as(select l.log_id,l.log_id - row_num as diff
from (select log_id,row_number() over(order by log_id) as row_num
from Logs)l)

select min(log_id) as start_date, max(log_id) as end_date
from cte
group by diff order by start_date
```

Q.70

```
CREATE Table Students(
student_id int PRIMARY KEY,
student_name varchar(30)
);
CREATE Table Subjects(
subject_name varchar(30) PRIMARY KEY
);
CREATE Table Examinations(
student_id int,
subject_name varchar(30)
);
INSERT INTO Students VALUES(1,'Alice'),(2,'Bob'),(13,'John'),(6,'Alex');
insert into Subjects VALUES('Math'),('Physics'),('Programming');
INSERT INTO Examinations VALUES (1,'Math'),(1,'Physics'),(1,'Programming'),
(2,'Programming'),(1,'Physics'),(1,'Math'),(13,'Math'),(13,'Programming'),
(13,'Physics'),(2,'Math'),(1,'Math');


with cte as(select student_id,subject_name,count(subject_name) as attended_exam
from Examinations
GROUP BY student_id,subject_name
order by student_id)

select s.student_id,s.student_name,s.subject_name,ifnull(c.attended_exam,0)
from (select * from Students cross join Subjects) s left join cte c
on s.student_id = c.student_id and s.subject_name = c.subject_name
order by student_id,subject_name
```

Q.71

```
create TABLE Employees(
employee_id int ,
employee_name varchar(30),
manager_id int
);
INSERT INTO Employees VALUES(1,'Boss',1),(3,'Alice',3),
(2,'Bob',1),
(4,'Daniel',2),
(7,'Luis',4),
(8,'Jhon',3),
(9,'Angela',8),
(77,'Robert',1);

select a.employee_id
from Employees a
join Employees b
```

```
join Employees c
ON a.manager_id = b.employee_id and
    b.manager_id = c.employee_id
WHERE c.manager_id = 1 and
    a.employee_id != c.manager_id
```

Q.72

```
create Table Transactions(
id int PRIMARY KEY,
country varchar(30),
state enum("approved", "declined"),
amount int,
trans_date date
);
INSERT INTO Transactions VALUES (121,'US', 'approved',1000,'2018-12-18'),
(122,'US','declined',2000,'2018-12-19'),
(123,'US','approved',2000,'2019-01-01'),
(124,'DE','approved',2000,'2019-01-07');


select left(trans_date,7) as month,country,count(*) as
trans_count,sum(if(state='approved',1,0))as approved_count,sum(amount) as trans_total_amount,
sum(if(state = 'approved', amount, 0)) as approved_total_amount
from Transactions
GROUP BY LEFT(trans_date,7),country;
```

Q.73

```
create Table Actions(
user_id int,
post_id int,
action_date date,
action enum('view', 'like', 'reaction', 'comment', 'report', 'share'),
extra varchar(30)
);
create Table Removals(
post_id int,
remove_date date
);
insert into Actions values(1,1,'2019-07-01', 'view','null'),
(1,1,'2019-07-01','like','null'),
(1,1,'2019-07-01','share','null'),
(2,2,'2019-07-04','view','null'),
(2,2,'2019-07-04','report','spam'),
(3,4,'2019-07-04','view','null'),
(3,4,'2019-07-04','report','spam'),
(4,3,'2019-07-02','view','null'),
(4,3,'2019-07-02','report','spam'),
(5,2,'2019-07-03','view','null'),
(5,2,'2019-07-03','report','racism'),
(5,5,'2019-07-03','view','null'),
(5,5,'2019-07-03','report','racism');
insert into Removals values(2,'2019-07-20'),(3,'2019-07-18');


select round(avg(min_avg),2) as average_daily_percent
from (select avg(post_id in (select post_id from Removals))*100 as min_avg
from (select post_id,action_date,extra
from Actions) ct
where extra = 'spam'
group by action_date)temp
```

Q.74

```
CREATE TABLE Activity(
player_id int,
```

```
device_id int,
event_date date,
games_played int
);
INSERT INTO Activity VALUES(1,2,'2016-03-01',5),(1,2,'2016-05-02',6),(2,3,'2017-06-25',1),
(3,1,'2016-03-02',0),(3,4,'2018-07-03',5);



select round(count(cte.player_id)/(select count(distinct player_id) from
Activity) ,2)as fraction
from (SELECT player_id,min(event_date) as start_date from Activity GROUP BY
player_id) as cte inner join Activity a
on cte.player_id=a.player_id and datediff(cte.start_date,a.event_date)=-1;
```

Q.75

```
CREATE TABLE Activity(
player_id int,
device_id int,
event_date date,
games_played int
);
INSERT INTO Activity VALUES(1,2,'2016-03-01',5),(1,2,'2016-05-02',6),(2,3,'2017-06-25',1),
(3,1,'2016-03-02',0),(3,4,'2018-07-03',5);



select round(count(cte.player_id)/(select count(distinct player_id) from
Activity) ,2)as fraction
from (SELECT player_id,min(event_date) as start_date from Activity GROUP BY
player_id) as cte inner join Activity a
on cte.player_id=a.player_id and datediff(cte.start_date,a.event_date)=-1;
```

Q.76

```
create Table Salaries(
company_id int,
employee_id int,
employee_name varchar(30),
salary int,
constraint primary key(company_id, employee_id)
);
insert into Salaries values(1,1,'Tony',2000),
(1,2,'Pronub',21300),
(1,3,'Tyrrox',10800),
(2,1,'Pam',300),
(2,7,'Bassem',450),
(2,9,'Hermione',700),
(3,7,'Bocaben',100),
(3,2,'Ognjen',2200),
(3,13,'Nyan Cat',3300),
(3,15,'Morning Cat',7777);

with cte as(select *,(case when salary < 1000 then 0
              when salary between 1000 and 10000 then 24 else 49 end) as tax_slab from Salaries
where (company_id,salary)in (select company_id,max(salary) from Salaries group by company_id))

select s.company_id,s.employee_id,s.employee_name,round(s.salary -(s.salary*(t.tax_slab/100))) as salary
from Salaries s join cte t on s.company_id = t.company_id
```

Q.77

```
create Table Variables(
name varchar(30) primary key ,
value int
);
create Table Expressions(
```

```
left_operand varchar(30),
operator enum('<', '>', '='),
right_operand varchar(30),
constraint fk primary key(left_operand, operator, right_operand)
);
insert into Variables values('x',66),('y',77);

insert into Expressions values ('x', '>', 'y'),
('x', '<', 'y'),
('x', '=', 'y'),
('y', '>', 'x'),
('y','<', 'x'),
('x', '=', 'x');

select e.*,case when operator = '=' and v1.value=v2.value then 'true'
                when operator ='<' and v1.value < v2.value then 'true'
                when operator ='>' and v1.value > v2.value then 'true'
                else 'false' end as value
from Expressions e left join Variables v1 on e.left_operand = v1.name
left join Variables v2 on e.right_operand = v2.name;
```

Q.78

```
CREATE TABLE person(
id int PRIMARY KEY,
name varchar(30),
phone_number varchar(30)
);
CREATE TABLE country(
name varchar(30),
country_code varchar(30) PRIMARY KEY
);
CREATE TABLE calls(
caller_id int,
callee_id int,
duration int
);
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");
insert into country values("Peru", '051'),("Israel", '972'),("Morocco",
'212'),("Germany", '049'),("Ethiopia", '251');
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);



select ct.name
from person p inner join calls c on p.id = c.caller_id or p.id=c.callee_id
inner join country ct on ct.country_code = left(p.phone_number,3)
group by ct.name
having avg(c.duration) > (select avg(duration) from calls)
```

Q.79

```
CREATE TABLE EMPLOYEE(
employee_id INTEGER,
name varchar(30),
months INTEGER,
salary INTEGER
);



INSERT INTO EMPLOYEE VALUES(1228,'Rose',15,1968),(33645,'Angela',1,3443),(45692,'Frank',17,1608),
(56118,'Pratick',7,1345),(59725,'Lisa',11,2330),
(74197,'Kimberly',16,4372),(78454,'Bonnie',8,1771),
(83565,'Michele',6,2017),(98607,'Todd',5,3396),
```

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

select name from EMPLOYEE order by name
```

q.80

```
with cte as (

select extract(year from transaction_date) as year, product_id, spend as curr_year_spend,

lag(spend,1) over (PARTITION BY product_id) as prev_year_spend from user_transactions

)

select c.*, round(((curr_year_spend-prev_year_spend)/prev_year_spend)*100,2) as yoy_rate from cte c
```

Q.81

```
WITH CTE as

(

SELECT

    sum(CASE WHEN item_type = 'prime_eligible' then square_footage END) as prime_sum,

    count(CASE WHEN item_type = 'prime_eligible' then 1 END) as prime_count,

    sum(CASE WHEN item_type = 'not_prime' then square_footage END) as non_prime_sum,

    count(CASE WHEN item_type = 'not_prime' then square_footage END) as non_prime_count

FROM inventory

)


select 'prime_eligible' as item_type,

     floor(500000/prime_sum)*prime_count as item_count

from CTE

UNION

select 'not_prime' as item_type,

     non_prime_count*(floor((500000-(floor(500000/prime_sum)*prime_sum))/non_prime_sum)) as
item_count
```

```
from CTE
  ORDER BY item_count desc
```

Q.82

```
with cte as(
SELECT DISTINCT user_id, EXTRACT(month from event_date) as month
FROM user_actions)

select c1.month,count(*) as monthly_active_users
from cte c1,cte c2
where (c1.user_id = c2.user_id
    AND c1.month = c2.month+1
    AND c1.month=7)
group by c1.month
```

Q.83

```
WITH expanded AS(
SELECT searches
FROM search_frequency
GROUP BY searches, GENERATE_SERIES(1,num_users)
)
SELECT PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY searches) AS median
FROM expanded
```

Q.84

```
WITH joined_table AS (
```

```
SELECT a.user_id, a.status, d.paid
FROM advertiser a
LEFT JOIN daily_pay d
USING (user_id)
UNION ALL
SELECT d.user_id, a.status, d.paid
FROM advertiser a
RIGHT JOIN daily_pay d
USING (user_id)
WHERE a.user_id IS NULL)
SELECT user_id,
  CASE WHEN status IS NULL AND paid IS NOT NULL
  THEN 'NEW'
  WHEN status = 'NEW' AND paid IS NOT NULL
  THEN 'EXISTING'
  WHEN status = 'NEW' AND paid IS NULL
  THEN 'CHURN'
  WHEN status = 'EXISTING' AND paid IS NOT NULL
  THEN 'EXISTING'
  WHEN status = 'EXISTING' AND paid IS NULL
  THEN 'CHURN'
  WHEN status = 'CHURN' AND paid IS NOT NULL
  THEN 'RESURRECT'
  WHEN status = 'CHURN' AND paid IS NULL
  THEN 'CHURN'
```

```
  WHEN status = 'RESURRECT' AND paid IS NOT NULL
  THEN 'EXISTING'
  WHEN status = 'RESURRECT' AND paid IS NULL
  THEN 'CHURN'
END AS new_status
FROM joined_table
ORDER BY user_id
```

Q.85

```
WITH CTE1 AS(
        SELECT server_id,
              status_time AS start_time,
              session_status,
              LEAD(status_time,1) over(PARTITION BY server_id ORDER BY status_time) as end_time
        FROM server_utilization
)

SELECT EXTRACT(days FROM JUSTIFY_HOURS(SUM(end_time - start_time)))
FROM CTE1
WHERE session_status = 'start';
```

Q.86

```
with cte as(SELECT transaction_timestamp,
      transaction_timestamp -LAG(transaction_timestamp,1) over(PARTITION BY
credit_card_id,merchant_id,amount order by transaction_timestamp) as diff
```

FROM transactions)

select count(*) as payment_count

from cte

where EXTRACT(minutes from diff)<=10 and EXTRACT(hours from diff) = 0

Q.87

```
show databases;

create database test_db;

use test_db;
create table Scores
(
    player_name varchar(30),
    gender varchar(30),
    day date,
    score_points int
)

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)

select gender,day,sum(score_points) over(partition by gender order by day) as total
from Scores
```

Q.89

```
CREATE TABLE person(
id int PRIMARY KEY,
name varchar(30),
phone_number varchar(30)
);
CREATE TABLE country(
name varchar(30),
country_code varchar(30) PRIMARY KEY
```

```
);
CREATE TABLE calls(
caller_id int,
callee_id int,
duration int
);
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");
insert into country values("Peru", '051'),("Israel", '972'),("Morocco",
'212'),("Germany", '049'),("Ethiopia", '251');
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);

with cte as(SELECT c.name,avg(cl.duration) as avg_dur
FROM person p inner join country c on LEFT(p.phone_number,3) = c.country_code
inner join calls cl on p.id =cl.caller_id or p.id = cl.callee_id
group by c.name)

select name from cte where avg_dur > (select avg(duration) from calls)
```

Q.90

Q.91

```
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 a.pay_month,b.department_id,
       (case when dept_avg > company_avg then 'higher'
             when dept_avg < company_avg then 'lower'
             else 'same' end) as comparision
from  (SELECT DATE_FORMAT(pay_date,'%Y-%m') as pay_month,avg(amount) as company_avg
from salary
GROUP BY DATE_FORMAT(pay_date,'%Y-%m')) a
right join
(select DATE_FORMAT(pay_date,'%Y-%m') as pay_month,department_id,avg(amount) as dept_avg
from salary s join employee e on s.employee_id = e.employee_id
group by DATE_FORMAT(pay_date,'%Y-%m'),department_id)b
on a.pay_month = b.pay_month
```

Q.92

```
create  table if not EXISTS Activity(
    player_id int,
    device_id int,
    event_date date,
    games_played int
)

insert into Activity values(1, 2, '2016-03-01', 5),
(1, 2, '2016-03-02', 6),
(2, 3, '2017-06-25', 1),
(3, 1, '2016-03-01', 0),
(3, 4, '2016-07-03',5)

with cte as(select a.player_id,install_date,event_date
from (select player_id,min(event_date) as install_date
from Activity
```

```
group by player_id) a
LEFT JOIN Activity b on b.event_date = a.install_date+1 and b.player_id = a.player_id)

select install_date,count(player_id) as
installs,round(count(event_date)/count(player_id),2) as Day1_retention
from cte
group by install_date
```

Q.93

```
CREATE TABLE Players(
player_id int PRIMARY KEY,
group_id varchar(30)
);
CREATE TABLE Matches(
match_id int primary KEY,
first_player int,
second_player int,
first_score int,
second_score int
);
insert into Players VALUES(15,1),
(25,1),
(30,1),
(45,1),
(10,2),
(35,2),
(50,2),
(20,3),
(40,3);
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);
with cte as
(   select temp.group_id,
           temp.player_id,
           rank() over(PARTITION BY temp.group_id order by sum(temp.score)
desc,temp.player_id asc) as rank1

    from(select p.group_id as group_id,
                p.player_id as player_id,
                sum(first_score) as score
        from Players p inner join Matches m
                    on p.player_id = m.first_player
        group by p.group_id,p.player_id

        union all

        select p.group_id as group_id,
               p.player_id as player_id,
```

```
                sum(second_score) as score
        from Players p inner join Matches m
                    on p.player_id = m.second_player
        group by p.group_id,p.player_id) temp
    group by temp.group_id,temp.player_id
)
select group_id,
        player_id
from cte
where rank1=1
```

Q.94

```
CREATE TABLE Student
(student_id INT,
student_name VARCHAR(32));
INSERT INTO Student
VALUES
(1, 'Daniel'),
(2, 'Jade'),
(3, 'Stella'),
(4, 'Jonathan'),
(5, 'Will');
CREATE TABLE Exam
(exam_id INT,
student_id INT,
score INT);
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);

with CTE AS(
    select distinct(student_id)
    from (select *,
                rank() over(partition by exam_id order by score desc) as r_max,
                rank() over(partition by exam_id order by score asc) as r_min
         from Exam) t
         where r_max=1 or r_min=1
),
CTE1 AS (
    SELECT DISTINCT(student_id)
```

```
    FROM Exam where student_id not in (select student_id from CTE)
)

select a.student_id as student_id,b.student_name as student_name
from CTE1 a inner join Student b on a.student_id = b.student_id
```

Q.95

```
CREATE TABLE Student
(student_id INT,
student_name VARCHAR(32));
INSERT INTO Student
VALUES
(1, 'Daniel'),
(2, 'Jade'),
(3, 'Stella'),
(4, 'Jonathan'),
(5, 'Will');
CREATE TABLE Exam
(exam_id INT,
student_id INT,
score INT);
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);

with CTE AS(
    select distinct(student_id)
    from (select *,
                rank() over(partition by exam_id order by score desc) as r_max,
                rank() over(partition by exam_id order by score asc) as r_min
         from Exam) t
         where r_max=1 or r_min=1
),
CTE1 AS (
    SELECT DISTINCT(student_id)
    FROM Exam where student_id not in (select student_id from CTE)
)

select a.student_id as student_id,b.student_name as student_name
from CTE1 a inner join Student b on a.student_id = b.student_id
```

Q.96

```sql
create table songs_history(
    history_id int,
    user_id int,
    song_id int,
    song_plays int
)

insert into songs_history values(10011, 777, 1238, 11),
(12452, 695, 4520, 1)

create table songs_weekly
(
    user_id int,
    song_id int,
   listen_time datetime

)


insert into songs_weekly values(777, 1238, '2022-08-01 12:00:00'),
(695, 4520, '2022-08-04 08:00:00'),
(125, 9630, '2022-08-04 16:00:00'),
(695, 9852, '2022-08-07 12:00:00')



select t.user_id,
       t.song_id,
       sum(t.song_plays) as song_palys

from(  select user_id,
              song_id,
              song_plays
       from songs_history
       union all
       select user_id,
              song_id,
              1 as song_plays
       from songs_weekly
       where DATE_FORMAT(listen_time,'%y-%m-%d') <= date('2022-08-04')) t

group by t.user_id,t.song_id
```

Q.97

```sql

with confirmed_count as(
select email_id
```

```
from texts where signup_action = 'Confirmed'
),total_cnt as(
select email_id from emails)

SELECT round((1.0*count(b.email_id)/count(a.email_id)),2) as confirmation_rate
from total_cnt a left join confirmed_count b on a.email_id = b.email_id
```

Q.98

```
WITH user_tweets AS(
    SELECT tweet_date, user_id,
    COUNT(tweet_id) AS user_tweets_per_day
    FROM tweets
    GROUP BY(tweet_date, user_id)
    ORDER BY tweet_date ASC
)

SELECT user_id, tweet_date, ROUND(AVG(user_tweets_per_day)
OVER(
    PARTITION BY user_id
    ORDER BY tweet_date
      ROWS BETWEEN 2 PRECEDING AND CURRENT ROW
), 2) AS rolling_avg_3days
FROM user_tweets
```

Q.99

```
with cte as(SELECT a.user_id,
                   b.age_bucket as age_bucket,
                   sum(case when a.activity_type = 'open' then a.time_spent end) as open,
                   sum(case when a.activity_type = 'send' then a.time_spent end) as send

            FROM activities a join age_breakdown b on a.user_id = b.user_id
            group by a.user_id,b.age_bucket)

select age_bucket,
       round((100.0*send/(open+send)),2) as send_perc,
       round((100.0*open/(open+send)),2) as open_perc

from cte
order by age_bucket
```

Q.100

```
create table personal_profiles(
    profile_id int,
    name varchar(30),
    followers int
```

```
)

insert into personal_profiles values(1, 'Nick Singh', 92000),
(2, 'Zach Wilson',199000),
(3, 'Daliana Liu', 171000),
(4, 'Ravit Jain', 107000),
(5, 'Vin Vashishta', 139000),
(6, 'Susan Wojcicki', 39000)

create table employee_company(
    personal_profile_id int,
    company_id int
)


insert into employee_company values(1,4),
(1, 9),
(2, 2),
(3,1),
(4, 3),
(5, 6),
(6, 5)

create table company_pages(
    company_id int,
    name varchar(30),
    followers int
)


insert into company_pages values(1, 'The Data Science Podcast', 8000),
(2,'Airbnb', 700000),
(3, 'The Ravit Show', 6000),
(4, 'DataLemur', 200),
(5, 'YouTube', 16000000),
(6, 'DataScience.Vin', 4500),
(9, 'Ace The Data Science Interview ',4479
)

select distinct p.profile_id
from employee_company e join personal_profiles p on e.personal_profile_id = p.profile_id
join company_pages c on e.company_id = c.company_id
where p.followers > c.followers
order by p.profile_id
```

Q.101

```
create table UserActivity(
    username varchar(30),
    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')

with cte as (select *,
       rank() over(PARTITION BY username order by startdate) as rnk,
       count(activity) over(PARTITION BY username) as cnt
from UserActivity)

select username,activity,startDate,endDate
from cte
where (cnt<>1 and rnk=2) or (cnt=1 and rnk =1)
```

Q.102

```
create table UserActivity(
    username varchar(30),
    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')

with cte as (select *,
       rank() over(PARTITION BY username order by startdate) as rnk,
       count(activity) over(PARTITION BY username) as cnt
from UserActivity)

select username,activity,startDate,endDate
from cte
where (cnt<>1 and rnk=2) or (cnt=1 and rnk =1)
```

Q.103

```
select Name
from Students
where marks > 75
order by right(Name,3),id
```

Q.104

```
select name
from Employee
where months < 10 and salary > 2000
```

Q.105

```
select (case when (a=b and b=c and c=a) then 'Equilateral'
             when (a+b <=c or b+c<=a or c+a<=b) then 'Not A Triangle'
             when (a=b or b=c or c=a) then 'Isosceles'
             else 'Scalene' end) as Types
from Triangles
```

Q.106

```
SELECT CEILING(AVG(Salary) - AVG(REPLACE(Salary, '0', ''))) FROM EMPLOYEES;
```

Q.107

```
select max(months*salary),count(employee_id)
from Employee
where (months*salary) = (select max(months*salary) from Employee)
```

Q.108

```
SELECT CONCAT(Name,'(', LEFT(occupation, 1),')') FROM OCCUPATIONS ORDER BY Name;
SELECT CONCAT("There are a total of ", COUNT(Occupation)," ",LOWER(Occupation),"s.")
FROM OCCUPATIONS GROUP BY occupation ORDER BY COUNT(Occupation), Occupation ASC;
```

Q.109

```
select min(if(Occupation="DOCTOR",name,NULL))AS Doctor,
    min(if(Occupation="PROFESSOR",name,NULL)) as Professor,
    min(if(Occupation="SINGER",name,NULL)) as Singer,
    min(if(Occupation="ACTOR",name,NULL)) as Actor
from(select name,Occupation,row_number()over(partition by Occupation order by name) as rn
from Occupations)t
group by rn
```

Q.110

```
select n,
      (case when p is null then 'Root'
            when N in (select distinct p from BST) then 'Inner'
            else 'Leaf' end) as val
from BST
order by n
```