**---CREATING TABLE FOR CITY---**

create table CITY(

ID int,

`NAME` VARCHAR(17),

COUNTRY\_CODE VARCHAR(3),

DISTRICT VARCHAR(20),

POPULATION INT

);

**---INSERTING VALUES INTO CITY---**

insert into city values

(6,'Rotterdam', 'NLD','Zuid-Holland',1345678 ),

(19,'Zaanstad', 'NLD','Noord-Holland',135621 ),

(214,'Porto Alegre','BRA','Rio Grande do Sul',1314032 ),

(397,'Lauro de Freitas','BRA','Bahia',109236 );

#ONLY few are selected here

**---QUESTIONS---**

**Q1.** Query all columns for all American cities in the CITY table with populations larger than 100000.

The CountryCode for America is USA.

Ans. SELECT \* FROM CITY

WHERE COUNTRY\_CODE='USA'

AND POPULATION>100000;

**Q2**. Query the NAME field for all American cities in the CITY table with populations larger than 120000.

The CountryCode for America is USA.

Ans. SELECT NAME FROM CITY

WHERE COUNTRY\_CODE='USA'

AND POPULATION>100000;

**Q3.** Query all columns (attributes) for every row in the CITY table.

Ans. SELECT \* FROM CITY;

**Q4.** Query all columns for a city in CITY with the ID 1661.

Ans. select \* from CITY

where id=1661;

**Q5**. Query all attributes of every Japanese city in the CITY table. The COUNTRYCODE for Japan is

JPN.

Ans. SELECT \* FROM CITY

WHERE COUNTRY\_CODE='JPN';

**Q6**. Query the names of all the Japanese cities in the CITY table. The COUNTRYCODE for Japan is

JPN.

Ans. SELECT NAME FROM CITY

WHERE COUNTRY\_CODE='JPN';

**---CREATING TABLE FOR STATION---**

CREATE TABLE STATION(

ID INT,

CITY VARCHAR(21),

STATE VARCHAR(2),

LAT\_N INT,

LONG\_W INT

);

**---INSERTING VALUES INTO TABLE---**

INSERT INTO STATION VALUES

(794, 'Kissee Mills','MO', 139, 73 ),

(824, 'Loma Mar', 'CA', 48 , 130 ),

(603, 'Sandy Hook', 'CT', 72 , 148 ),

(478, 'Tipton', 'IN', 33 , 97 );

#Only few are selected here.

**---QUESTIONS----**

**Q7.** Query a list of CITY and STATE from the STATION table.

Ans. SELECT CITY,STATE FROM STATION;

**Q8**. Query a list of CITY names from STATION for cities that have an even ID number. Print the results

in any order, but exclude duplicates from the answer.

Ans. SELECT DISTINCT CITY FROM STATION

WHERE ID%2=0;

**Q9**. Find the difference between the total number of CITY entries in the table and the number of

distinct CITY entries in the table.

Ans. SELECT count(CITY) - count(DISTINCT city) FROM STATION;(#gives 13)

**Q10.** Query the two cities in STATION with the shortest and longest CITY names, as well as their

respective lengths (i.e.: number of characters in the name). If there is more than one smallest or largest city, choose the one that comes first when ordered alphabetically.

Ans. SELECT CITY,LENGTH(CITY) FROM STATION

WHERE CITY=(SELECT CITY FROM STATION ORDER BY length(CITY),CITY LIMIT 1)

OR CITY=(SELECT CITY FROM STATION ORDER BY length(CITY) DESC,CITY LIMIT 1);

**Q11.** Query the list of CITY names starting with vowels (i.e., a, e, i, o, or u) from STATION. Your result

cannot contain duplicates.

Ans. select distinct city from station

where substr(city,1,1) in ('a','e','i','o','u');

**Q14**. Query the list of CITY names from STATION that do not end with vowels. Your result cannot

contain duplicates.

Ans. select distinct city from station

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

**Q15**. Query the list of CITY names from STATION that either do not start with vowels or do not end

with vowels. Your result cannot contain duplicates.

Ans. SELECT DISTINCT CITY FROM STATION

WHERE lower(LEFT(CITY, 1)) not IN ('a', 'e', 'i', 'o', 'u')

OR lower(RIGHT(CITY, 1)) not IN ('a', 'e', 'i', 'o', 'u');

**Q16.** Query the list of CITY names from STATION that do not start with vowels and do not end with

vowels. Your result cannot contain duplicates

Ans. SELECT DISTINCT CITY FROM STATION

WHERE lower(LEFT(CITY, 1)) not IN ('a', 'e', 'i', 'o', 'u')

AND lower(RIGHT(CITY, 1)) not IN ('a', 'e', 'i', 'o', 'u');

**----PRODUCT TABLE CREATION---**

CREATE TABLE PRODUCT(

product\_id INT,

product\_name varchar(30),

unit\_price int,

constraint pk primary key(product\_id)

);

Insert into product values(1,'S8',1000),(2,'G4',800),(3,'iPhone',1400);

**----SALES TABLE CREATION----**

CREATE TABLE SALES(

seller\_id int,

product\_id int,

buyer\_id int,

sale\_date date,

quantity int,

price int,

constraint fk foreign key(product\_id) references product(product\_id) );

**-----Inserting values----**

Insert into sales values(1,1,1,'2019-01-21',2,2000),(1,2,2,'2019-02-17',1,800),(2,2,3,'2019-06-02',1,800), (3,3,4,'2019-05-13',2,2800);

**----Problems-----**

**Q17.** Write an SQL query that reports the products that were only sold in the first quarter of 2019.

That is, between 2019-01-01 and 2019-03-31 inclusive.

Ans. SELECT product\_id, product\_name FROM product

WHERE product\_id NOT IN (SELECT product\_id FROM sales

WHERE sale\_date NOT BETWEEN '2019-01-01'

AND '2019-03-31');

**---TABLE CREATION----**

create table views(

article\_id int,

author\_id int,

viewer\_id int,

view\_date date);

**-----INSERTING----**

insert into views values(1,3,5,'2019-08-01'),(1,3,6,'2019-08-02'),(2,7,7,'2019-08-01'),(2,7,6,'2019-08-02'),(4,7,1,'2019-07-22'),(3,4,4,'2019-07-21'),(3,4,4,'2019-07-21');

**----PROBLEMS-----**

**Q18.** Write an SQL query to find all the authors that viewed at least one of their own articles. Return the result table sorted by id in ascending order.

**Ans.** select distinct author\_id as id from views

where author\_id=viewer\_id

order by id;

**Q19.** create table Delivery(

delivery\_id int,

customer\_id int,

order\_date date,

customer\_pref\_delivery\_date date);

insert into Delivery values(1,1,'2019-08-01','2019-08-02'),(2,5,'2019-08-02','2019-08-02'),(3,1,'2019-08-11','2019-08-11'), (4,3,'2019-08-24','2019-08-26'),(5,4,'2019-08-21','2019-08-22'),

(6,2,'2019-08-11','2019-08-13');

**Ans.** select round(avg(order\_date=customer\_pref\_delivery\_date)\*100,2) as immediate\_percentage

from Delivery;

**Q20**. create table Ads(

ad\_id int,

user\_id int,

`action` enum('Clicked','Viewed','Ignored'),

constraint pk primary key(ad\_id,user\_id)

);

**In:** insert into Ads values

(1,1,'Clicked'),(2,2,'Clicked'),(3,3,'Viewed'),(5,5,'Ignored'),(1,7,'Ignored'),(2,7,'Viewed'),

(3,5,'Clicked'),(1,4,'Viewed'),(2,11,'Viewed'),(1,2,'Clicked');

**Query.** select ad\_id, ifnull(

Round(sum(action='Clicked')/(sum(action='Clicked')+sum(action='Viewed'))\*100,2),0)

as ctr from Ads

group by ad\_id

order by ctr desc, ad\_id asc;

**Q21.** create table if not exists Employee(

employee\_id int,

team\_id int,

constraint pk primary key(employee\_id)

);

**IN:** insert into Employee values(1,8),(2,8),(3,8),(4,7),(5,6),(6,9);

**Query:** select employee\_id,count(team\_id)over(partition by team\_id) from Employee

order by employee\_id;

**Q22.** -- Table-Countries

create table if not exists Countries(

country\_id int,

country\_name varchar(10),

constraint pk primary key(country\_id) );

**In:** insert into Countries values(2,'USA'),(3,'Australia'),(7,'Peru'),(5,'China'),(8,'Morocco'),(9,'Spain');

-- Table-Weather

create table if not exists Weather(

country\_id int,

weather\_state int,

`day` date,

constraint pk primary key(country\_id,day));

**In:** insert into Weather values(2, 15, '2019-11-01'),

(2, 12, '2019-10-28'),(2, 12, '2019-10-27'),(3, -2, '2019-11-10'),(3, 00, '2019-11-11'),

(3, 03, '2019-11-12'),(5, 16, '2019-11-07'),(5, 18, '2019-11-09'),(5, 21, '2019-11-23'),

(7, 25, '2019-11-28'),(7, 22, '2019-12-01'),(7, 20, '2019-12-02'),(8, 25, '2019-11-05'),

(8, 27, '2019-11-15'),(8, 31, '2019-11-25'),(9, 07, '2019-10-23'),(9, 03, '2019-12-23');

**Query:** select distinct c.country\_name,

case

when avg(w.weather\_state)<=15 then 'Cold'

when avg(w.weather\_state)>=25 then 'Hot'

else 'Warm'

end as weather\_type

from Countries c inner join Weather w on c.country\_id=w.country\_id

where w.`day` between '2019-11-01' and '2019-11-30'

group by c.country\_name;

**Q23**. -- Table-Prices

create table if not exists Prices(

product\_id int,

start\_date date,

end\_date date,

price int,

constraint pk primary key(product\_id,start\_date,end\_date));

**In:** insert into Prices 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);

-- Table-UnitsSold

create table if not exists UnitsSold(

product\_id int,

purchase date date,

units int) ;

**In:**  insert into UnitsSold values(1,'2019-02-25',100),(1,'2019-03-01',15),

(2,'2019-02-10',200),(2,'2019-03-22',30);

**Query:** with tmp as (select p.product\_id as product\_id, units, price\*units as tot\_price

from Prices p left join UnitsSold u on p.product\_id =u.product\_id

and purchase\_date between start\_date and end\_date)

select product\_id,round(sum(tot\_price)/sum(units),2) as average\_price from tmp

group by product\_id ;

**Q24.** create table if not exists Activity(

player\_id int,

device\_id int,

event\_date date, games\_played int,

constraint pk 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,'2016-03-02',0),(3,4,'2018-07-03',5);

**Query:** Select distinct player\_id, min(event\_date) over(partition by player\_id order by event\_date)

From Activity;

**(OR)**

select player\_id, min(event\_date) from Activity group by player\_id;

**Q25.** **Same Table Actitvity**

**Query:** select player\_id,device\_id from Activity

where event\_date in (select min(event\_date) from Activity

group by player\_id);

**Q26.** -- Table-Products

create table if not exists Products(

product\_id int,

product\_name varchar(30),

product\_category varchar(10),

constraint pk primary key (product\_id) );

**In:** insert into Products values (1,'Leetcode Solutions','Book'), (2,'Jewels of Stringology ','Book'), (3,'HP','Laptop'), (4,'Lenovo','Laptop'), (5,'Leetcode Kit','T-shirt');

-- Table-Orders

create table if not exists Orders(

product\_id int,

order\_date date,

unit int,

foreign key(product\_id) references Products(product\_id));

**In:** insert into Orders values (1, '2020-02-05', 60),(1, '2020-02-10', 70),(2, '2020-01-18', 30),

(2, '2020-02-11', 80),(3, '2020-02-17', 2 ),(3, '2020-02-24', 3 ),(4, '2020-03-01', 20),

(4, '2020-03-04', 30),(4, '2020-03-04', 60),(5, '2020-02-25', 50),(5, '2020-02-27', 50),

(5, '2020-03-01', 50);

**Query:** select product\_name, unit from( select distinct p.product\_id, product\_name,

sum(unit) as unit from Products p inner join Orders o

on p.product\_id=o.product\_id and

order\_date between '2020-02-01' and '2020-02-29'

group by p.product\_id,product\_name

having sum(unit)>=100) as tmp ;

**Q27.** create table if not exists Users(

user\_id int,

name varchar(15),

mail varchar(25),

primary key (user\_id)

);

**IN:** insert into Users values (1,'Winston','winston@leetcode.com'),(2,'Jonathan','jonathanisgreat'),

(3,'Annabelle','bella-@leetcode.com'),(4,'Sally','sally.come@leetcode.com'),

(5,'Marwan','quarz#2020@leetcode.com'),(6,'David','david69@gmail.com'),

(7,'Shapiro','.shapo@leetcode.com');

**Query:**

select user\_id,`name`,mail from Users

where REGEXP\_LIKE(mail, '^[a-zA-Z][a-zA-Z0-9\\_\.\-][\*@leetcode.com](mailto:*@leetcode.com)');

**Q28.** -- Table-Customers

create table if not exists Customers(

customer\_id int,

name varchar(15),

country varchar(10),

primary key (customer\_id) );

**IN:**

insert into Customers values(1,'Winston','USA'),(2,'Jonathan','Peru'),(3,'Moustafa','Egypt');

-- Table-Product

create table if not exists Product(

product\_id int,

`description` varchar(15),

price int,

primary key(product\_id) );

**IN:**

insert into Product values(10,'LC Phone',300),(20,'LC T-Shirt',10),(30,'LC Book',45),(40,'LC Keychain',2);

-- Table-Orders

create table if not exists `Orders`(

order\_id int,

customer\_id int,

product\_id int,

order\_date date,

quantity int,

primary key (order\_id) );

**IN:**

insert into `Orders` values(1,1,10,'2020-06-10',1),

(2,1,20,'2020-07-01',1),(3,1,30,'2020-07-08',2),(4,2,10,'2020-06-15',2),

(5,2,40,'2020-07-01',10),(6,3,20,'2020-06-24',2),(7,3,30,'2020-06-25',2),(9,3,30,'2020-05-08',3);

**Query:**

select o.customer\_id, c.name

from Customers c, Product p, `Orders` o

where c.customer\_id = o.customer\_id and p.product\_id = o.product\_id

group by o.customer\_id

having

(

sum(case when o.order\_date like '2020-06%' then o.quantity\*p.price else 0 end) >= 100

and

sum(case when o.order\_date like '2020-07%' then o.quantity\*p.price else 0 end) >= 100 ) ;

**Q29. -- Table-TVProgram**

create table if not exists TVProgram(

program\_date date,

content\_id int,

channel varchar(15),

primary key (program\_date,content\_id) );

**IN:**

insert into TVProgram values('2020-06-10',1,'LC-Channel'),('2020-05-11',2,'LC-Channel'),

('2020-05-12',3,'LC-Channel'), ('2020-05-13',4,'Disney Ch'),('2020-06-18',4,'Disney Ch'),

('2020-07-15',5,'Disney Ch');

**-- Table-Content**

create table if not exists Content(

content\_id int,

title varchar(20),

kids\_content enum('Y','N'),

content\_type varchar(10),

primary key(content\_id) );

**IN:**

insert into Content values(1,'Leetcode Movie','N','Movies'),(2,'Alg. for kids','Y','Series'),

(3,'Database Sols','N','Series'),(4,'Aladdin','Y','Movies'),(5,'Cinderella','Y','Movies');

**QUERY:**

select distinct title from TVProgram t join Content c on t.content\_id=c.content\_id

and t.program\_date like '2020-06-%' and c.kids\_content='Y';

**Q30. -- Table-NPV**

create table if not exists NPV(

id int, `year` int, npv int,

primary key(id,`year`) );

**IN:** insert into NPV values(1,2018,100),(7,2020,30),(13,2019,40),(1,2019,113),

(2,2008,121),(3,2009,12),(11,2020,99), (7,2019,0);

**-- Table-Queries**

create table if not exists Queries(

id int, `year` int,

primary key(id,`year`) );

**IN:** insert into Queries values(1,2019),(2,2008),(3,2009),(7,2018),(7,2019),(7,2020),(13,2019);

**Query:**

select q.\*,ifnull(n.npv,0) from NPV n right join Queries q on n.id=q.id and n.year=q.year;

**Q31.**  **30** **Question repeated ……….**

**Q32**. **-- Table - Employees**

create table if not exists Employees(

id int, `name` varchar(10),

primary key (id) );

**IN:**

insert into Employees values(1,'Alice'),(2,'Bob'),(11,'Meir'),(90,'Winston'),(3,'Jonathan');

**-- Table -- EmployeeUNI**

create table if not exists EmployeeUNI(

id int, unique\_id int,

primary key(id,unique\_id) );

**IN:**

insert into EmployeeUNI values(3,1),(11,2),(90,3);

**Query:**

select u.unique\_id,e.name from Employees e left join EmployeeUNI u on e.id=u.id;

**Q33**. **-- Table - Users**

create table if not exists Users(

id int, name varchar(10),

primary key (id) );

**IN:**

insert into `User` values(1,'Alice'),(2,'Bob'),(3,'Alex'),(4,'Donald'),(7,'Lee'),(13,'Jonathan'),(19,'Elvis');

**-- Table - Rides**

create table if not exists Rides(

id int, user\_id int, distance int,

primary key(id) );

**IN:**

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);

**Query:**

select distinct u.name, ifnull (sum (r.distance) over(partition by user\_id) , 0) as travelled\_distance from Users u left join Rides r on u.id = r.user\_id

order by travelled\_distance desc , u.name;

**Q34. 26 Question Repeated…**

**Q35**. **-- Table - Movies**

create table if not exists Movies(

movie\_id int, title varchar(10),

primary key(movie\_id) );

**IN:**

insert into Movies values(1,'Avengers'),(2,'Frozen 2'),(3,'Joker');

**-- Table - Users**

create table if not exists Users(

user\_id int, name varchar(10),

primary key(user\_id) );

**IN:**

insert into Users values(1,'Daniel'),(2,'Monica'),(3,'Maria'),(4,'James');

**-- Table - MovieRating**

create table if not exists MovieRating(

movie\_id int, user\_id int,

rating int, created\_at date,

primary key (movie\_id,user\_id) );

**IN:**

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');

**Query:**

select user\_name as results from

(

select u.name as user\_name, COUNT(mr.user\_id) from MovieRating as mr

join Userss as u

on u.user\_id = mr.user\_id

group by mr.user\_id

order by count(mr.user\_id) desc, user\_name asc limit 1

) as first\_one

Union

select title as results from(

select m.title,sum(mr.rating) as rate from MovieRating mr join Movies m

on mr.movie\_id = m.movie\_id

where created\_at like '2020-02-%'

group by mr.movie\_id

order by rate desc, m.title asc limit 1

) snd\_one;

**Q36. 33 Question is repeated……**

**Q37. 32 Question is repeated……**

**Q38**. **-- Table - Departments**

create table if not exists Departments(

id int, name varchar(30),

primary key(id) );

**IN:**

insert into Departments values(1,'Electrical Engineering'),(7,'Computer Engineering'),

(13,'Business Administration');

**-- Table - Students**

create table if not exists Students(

id int, name varchar(15), department\_id int,

primary key (id) );

**IN:**

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);

**Query:**

select s.id,s.name from Students s

where department\_id not in (select id from Departments);

**Q39**. create table if not exists Calls(

from\_id int, to\_id int, duration int);

**IN:**

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);

**Query:**

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 person1, person2;

**Q40. 23 Question is repeated……**

**Q41. -- Table Warehouse**

create table if not exists Warehouse(

name varchar(10), product\_id int, units int,

primary key (name,product\_id) );

**IN:**

insert into Warehouse values('LCHouse1', 1,1),('LCHouse1', 2,10),('LCHouse1', 3,5),

('LCHouse2', 1,2),('LCHouse2', 2,2), ('LCHouse3', 4,1);

**-- Table Products**

create table if not exists Products(

product\_id int, product\_name varchar(15),

width int, length int, height int,

primary key (product\_id) );

**IN:**

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);

**Query:**

select w.name,sum(w.units\*p.width\*p.length\*p.height) as volume

from Warehouse w join Products p on p.product\_id = w.product\_id

group by w.name;

**Q42.** create table if not exists Sales(

sale\_date date,

fruit enum('apples','oranges'),

sold\_num int,

primary key(sale\_date,fruit) );

**IN:** 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);

**Query:**

select a.sale\_date, (a.sold\_num - b.sold\_num) as diff

from Sales a left join Sales b

on a.sale\_date = b.sale\_date

where a.fruit = 'apples' and b.fruit = 'oranges';

**Q43.** create table if not exists Activityy(

player\_id int, devide\_id int,

event\_date date, games\_played int,

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

**IN:** insert into Activityy 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);

**Query:**

with tmp as (select a.player\_id from Activityy a left join Activityy b on a.player\_id=b.player\_id

where datediff(a.event\_date,b.event\_date)=-1)

select round **(**

sum(t.player\_id) / ( select count (distinct player\_id) from Activity), 2 **)** as fraction

from tmp t;

**Q44.** create table if not exists Employee(

id int, name varchar(8),

department varchar(1), managerId int,

primary key (id) );

**IN:** 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);

**Query:** select Name from Employee as t1

join (select ManagerId from Employee

group by ManagerId

having count(ManagerId) >= 5) as t2

on t1.Id = t2.ManagerId;

**Q45. -- Table - Student**

create table if not exists Student(

student\_id int, student\_name varchar(4),

gender varchar(1),dept\_id int,

primary key(student\_id),

foreign key(dept\_id) references Department(dept\_id) );

**IN:**

insert into Student values(1,'Jack','M',1),(2,'Jane','F',1),(3,'Mark','M',2);

**-- Table - Department**

create table if not exists Department(

dept\_id int, dept\_name varchar(15),

primary key(dept\_id) );

**IN:**

insert into Department values(1,'Engineering'),(2,'Science'),(3,'Law');

**Query:**

select d.dept\_name, count(s.dept\_id) as student\_number

from Department d left join Student s

on d.dept\_id=s.dept\_id

group by d.dept\_id;

**Q46. -- Table - Customer**

create table if not exists Customer(

customer\_id int, product\_key int);

**IN:**

insert into Customer values(1,5),(2,6),(3,5),(3,6),(1,6);

**-- Table - Product**

create table if not exists Product(

product\_key int,

primary key (product\_key) );

**IN:**

insert into Product values (5),(6);

**Query:**

select customer\_id from Customer

group by customer\_id

having count(customer\_id) = (select count(\*) from Product);

**Q47**. **-- Table Project**

create table if not exists Project(

project\_id int, employee\_id int,

primary key(project\_id,employee\_id),

foreign key(employee\_id) references Employyee(employee\_id) );

**IN:**

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

**-- Table Employee**

create table if not exists Employyee(

employee\_id int, name varchar(8),

experience\_years int,

primary key (employee\_id) );

**IN:**

insert into Employyee values(1,'Khaled',3),(2,'Ali',2),(3,'John',3),(4,'Doe',2);

**Query:**

select project\_id,employee\_id from

(

select

p.project\_id,

e.employee\_id,e.experience\_years,

Rank() over (partition by project\_id order by experience\_years desc ) as rank\_experience

from Project p join employee e

on p.employee\_id = e.employee\_id

) as tmp

where rank\_experience=1;

**Q48. -- Table - Books**

create table if not exists Books(

book\_id int, name varchar(30), available\_from date,

primary key(book\_id) );

**IN:**

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');

**-- Table - Orders**

create table if not exists Orderss(

order\_id int, book\_id int,

quantity int, dispatch\_date date,

primary key(order\_id),

foreign key(book\_id) references Books(book\_id) );

**IN:**

insert into Orderss 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');

**Query:**

select book\_id, name

from books

where book\_id not in (

select book\_id

from Orderss

where (dispatch\_date between date\_sub('2019-06-23',interval 1 year) and '2019-06-23')

group by (book\_id)

having sum(quantity) >= 10)

and

available\_from < date\_sub('2019-06-23', interval 1 month);

**Q49. -- Table - Enrollments**

create table if not exists Enrollments(

student\_id int, course\_id int, grade int,

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

**IN:**

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);

**Query:**

select student\_id,course\_id,grade from (select \*,

dense\_rank() over(partition by student\_id order by grade desc,course\_id asc) as ranks

from Enrollments) as tmp

where ranks=1;

**Q50. Two questions given with half data…..**