

**51.)** name is the primary key column for this table.

Each row of this table gives information about the name of a country, the continent to which it

belongs, its area, the population, and its GDP value.

A country is big if:

● it has an area of at least three million (i.e., 3000000 km2), or

● it has a population of at least twenty-five million (i.e., 25000000).

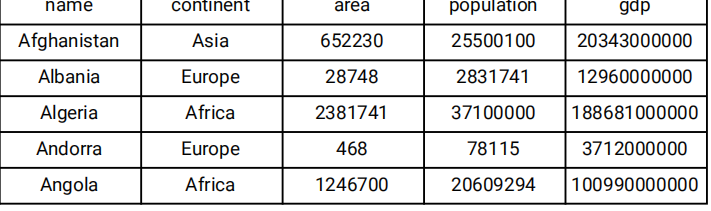
Write an SQL query to report the name, population, and area of the big countries.

Return the result table in any order.

The query result format is in the following example.

Input:

World table:



**Solution:**

create DATABASE Ineuron;

use Ineuron;

Create TABLE World (

    Name varchar(255),

    continent varchar(255),

    area int,

    Population int,

    gdp INT,

    PRIMARY KEY (Name)

);

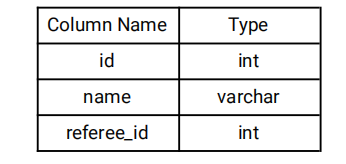
ALTER table World MODIFY COLUMN  gdp BIGINT;

INSERT INTO World VALUES ('Afghanistan','Asia',652230,25500100,20343000000);

INSERT INTO World VALUES ('Albania','Europe',28748,2831741,1296000000),('Algeria','Africa',2381741,37100000,188681000000),('Andorra','Europe',468,78115,3712000000),('Angola','Africa',1246700,20609294,100990000);

select \* from World;

select name,population,area from World where area >=3000000 or population >= 25000000;



**52.)** id is the primary key column for this table.

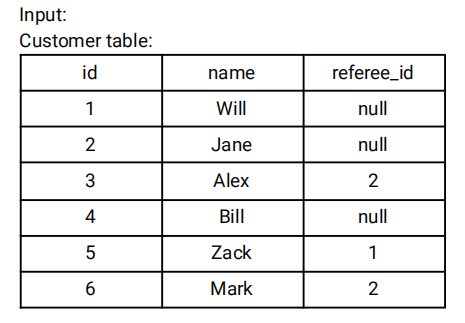
Each row of this table indicates the id of a customer, their name, and the id of the customer who

referred them.

Write an SQL query to report the names of the customer that are not referred by the customer with id = 2.

Return the result table in any order.

The query result format is in the following example.



**Solution:**

create table Customer (

    id int,

    name varchar(50),

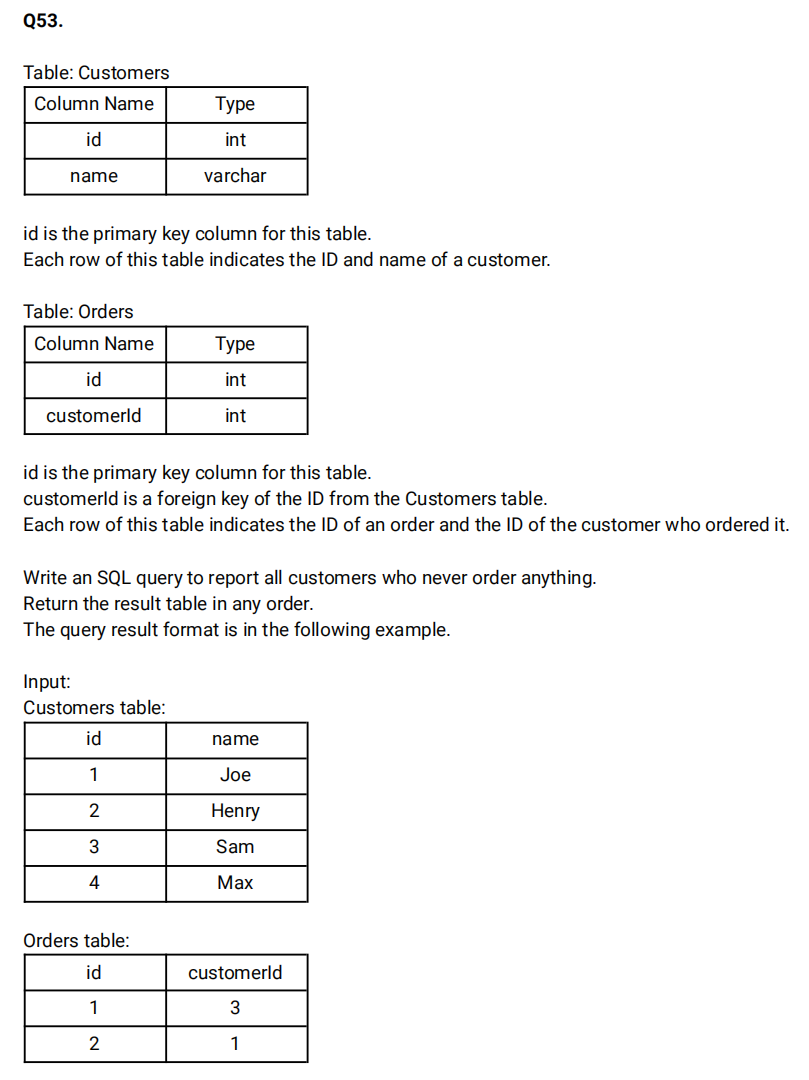
    refree\_id INT

);

Insert INTO Customer VALUES (1,'Will',null),(2,'Jane',null),(3,'Alex',2),(4,'Bill',null),(5,'Zack',1),(6,'Mark',2);

select \* from Customer;

select name from Customer where id<>2 OR id IS NULL;



**Solution:**

create Table Customers (

    id int,

    name VARCHAR(100),

    PRIMARY KEY(id)

);

create table orders (

    id int PRIMARY KEY,

    customer\_id INT,

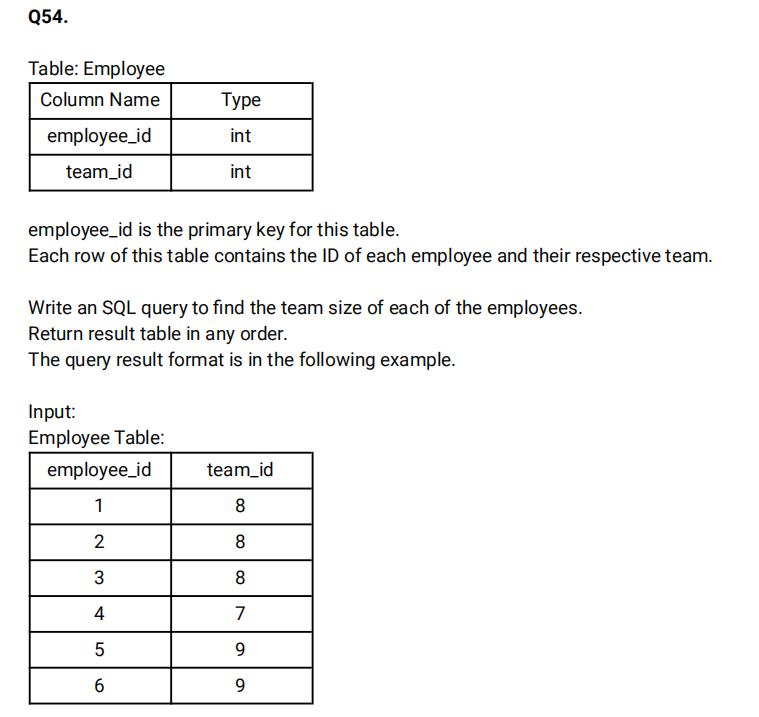
    FOREIGN KEY(customer\_id) REFERENCES Customers(id)

);

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

INSERT INTO orders VALUES(1,3),(2,1);

 select distinct name from Customers C join orders o where C.id NOT IN(Select customer\_id from orders) ;



create table employee (

    employee\_id INT PRIMARY KEY,

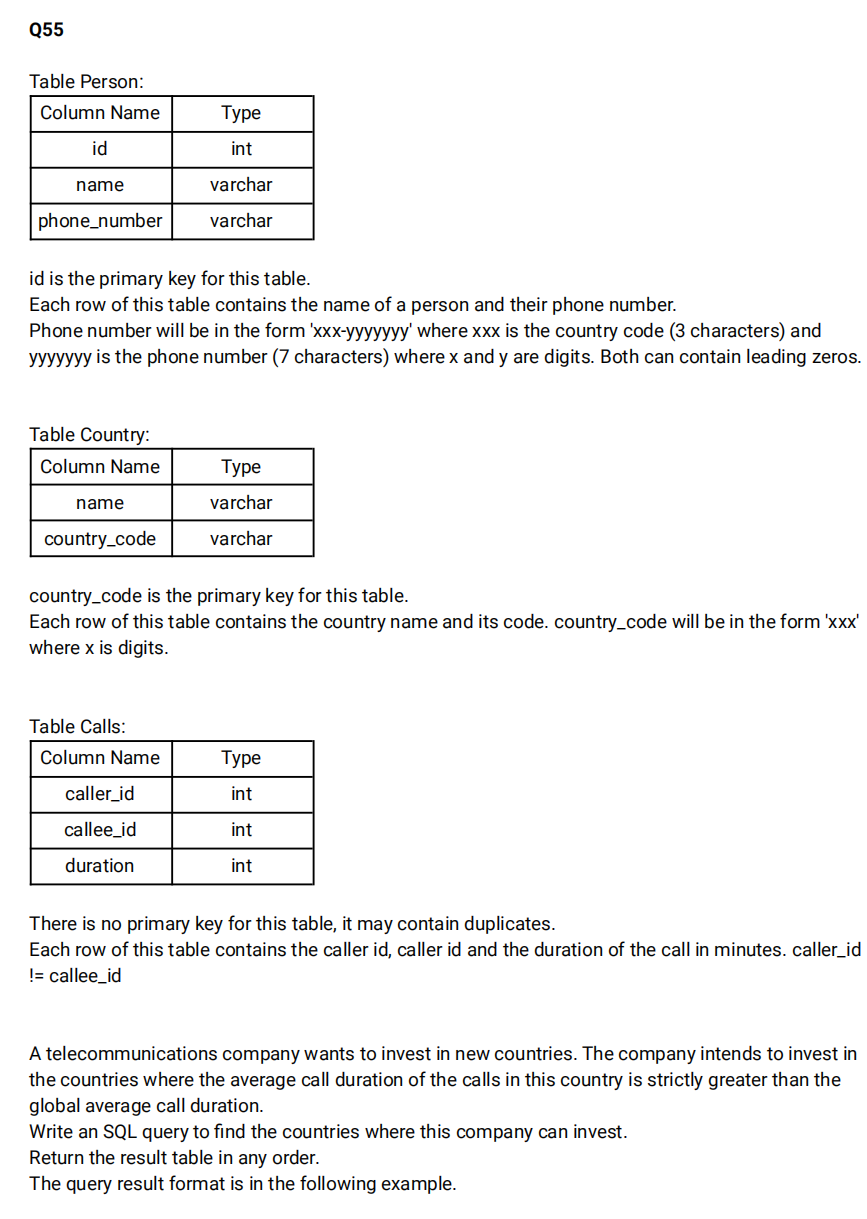
    team\_id INT

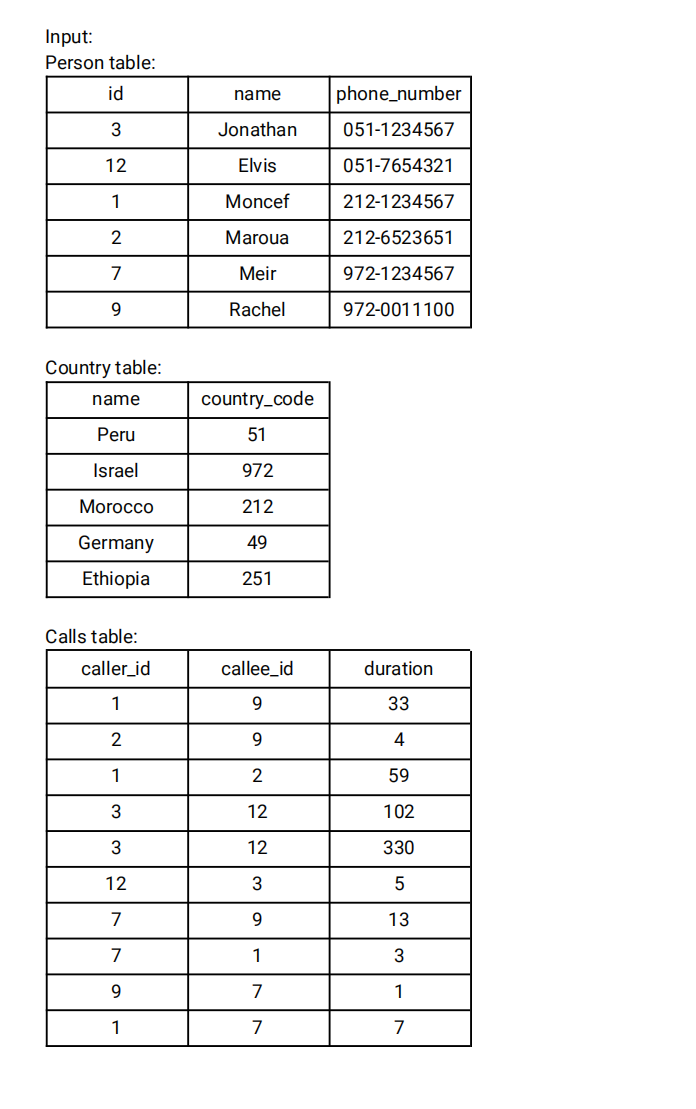
 );

 INSERT INTO employee VALUES (1,8),(2,8),(3,8),(4,7),(5,9),(6,9);

 select \* from employee;

 select employee\_id, COUNT(team\_id) OVER(PARTITION BY team\_id) as team\_size from employee order by employee\_id;





create table person(

    id INT,

    name VARCHAR(255),

    phone\_number VARCHAR(255)

 );

 create table country(

    name VARCHAR(50),

    country\_code VARCHAR(10) PRIMARY KEY

 );

 ALTER Table person ADD constraint PRIMARY KEY(id);

 create table calls(

    caller\_id INT,

    callee\_id INT,

    duration INT

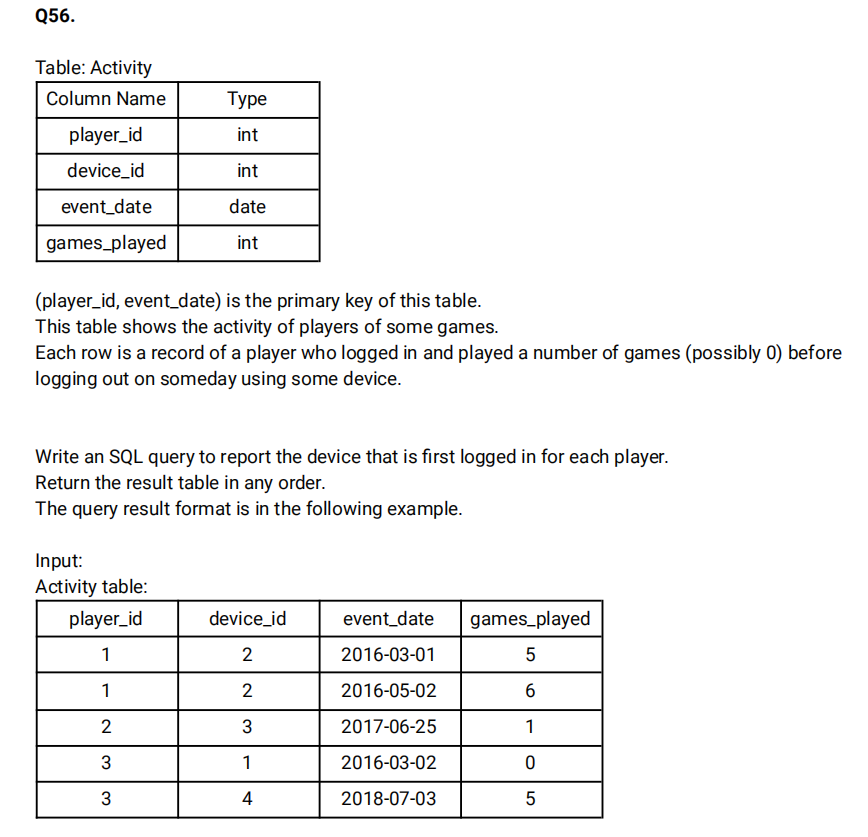
 );

 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','49'),('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 c.name as country from person p join country c on SUBSTRING(phone\_number,1,3)=c.country\_code join calls ca on p.id IN (ca.caller\_id,ca.callee\_id) group by c.name HAVING AVG(duration)>(select AVG(duration) from calls);



create table Activity (

    player\_id INT,

    device\_id INT,

    event\_date DATE,

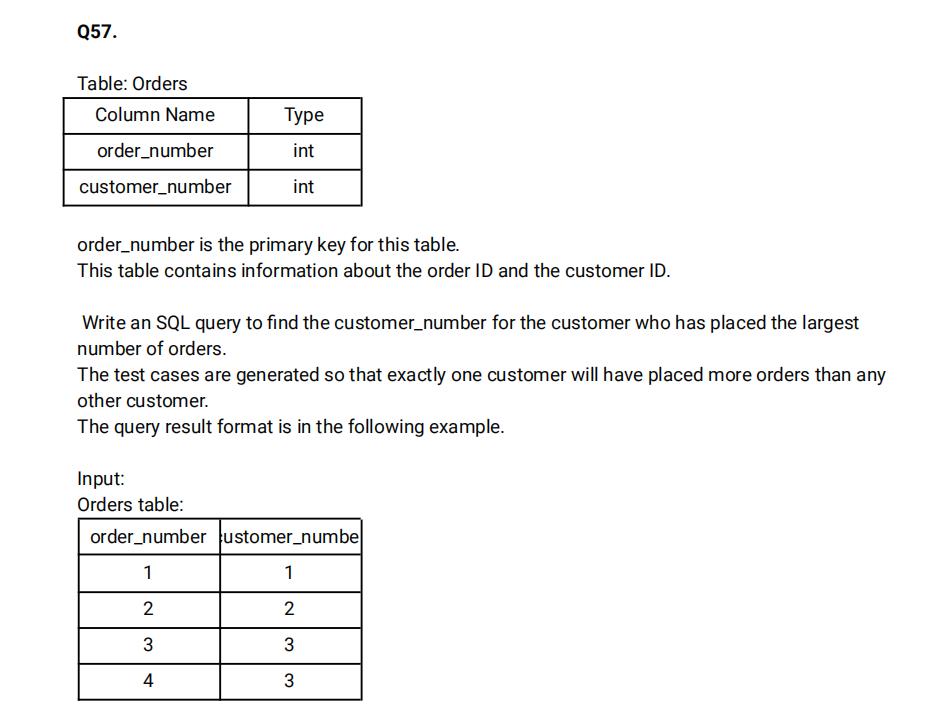
    games\_played INT,

    PRIMARY KEY(player\_id,event\_date)

 );

 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 (select player\_id,device\_id, DENSE\_RANK() OVER (partition by player\_id order by event\_date ASC) as cnt from Activity)temp WHERE cnt=1;



create table Orders(

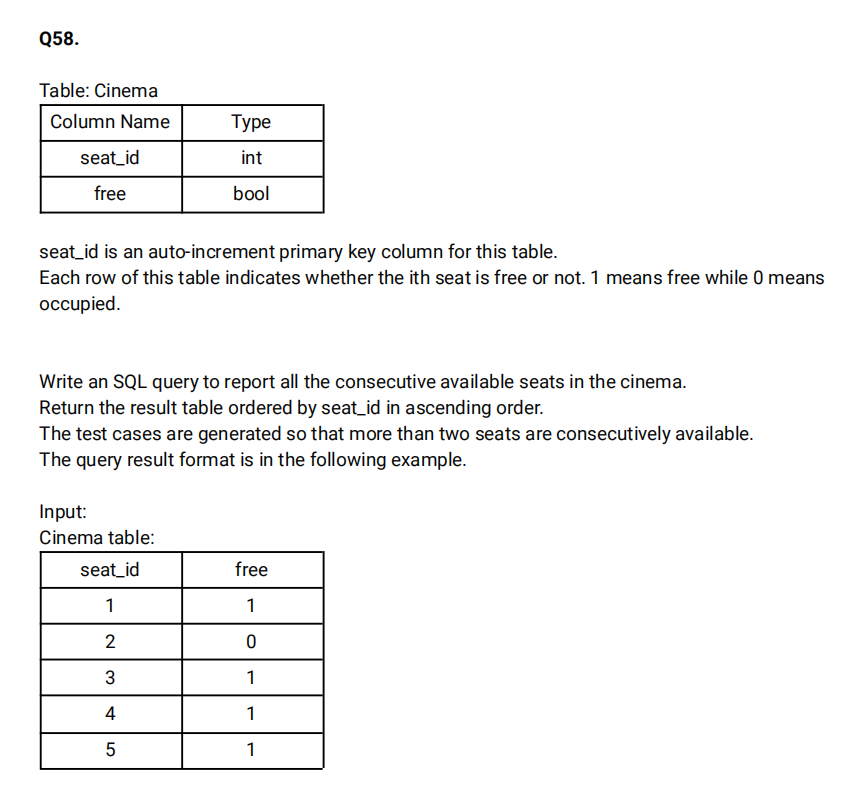
    order\_number INT PRIMARY KEY,

    customer\_number INT

 );

 INSERT INTO Orders VALUES (1,1),(2,2),(3,3),(4,3);

 select customer\_number from (select customer\_number,count(order\_number) as cnt from Orders group by customer\_number order by cnt desc limit 1)t;



create table cinema (

   seat\_id INT PRIMARY KEY AUTO\_INCREMENT,

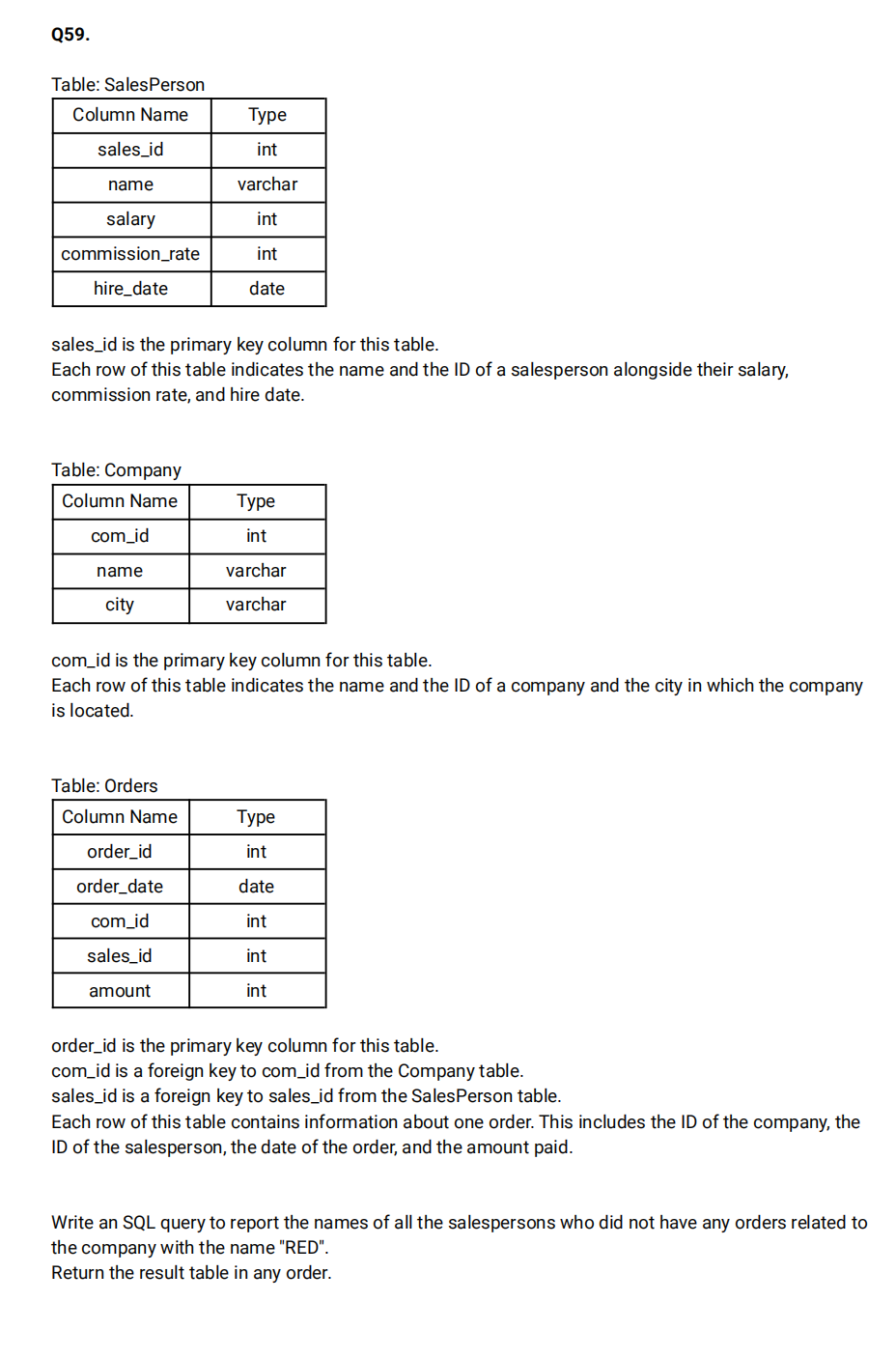
   free BOOLEAN

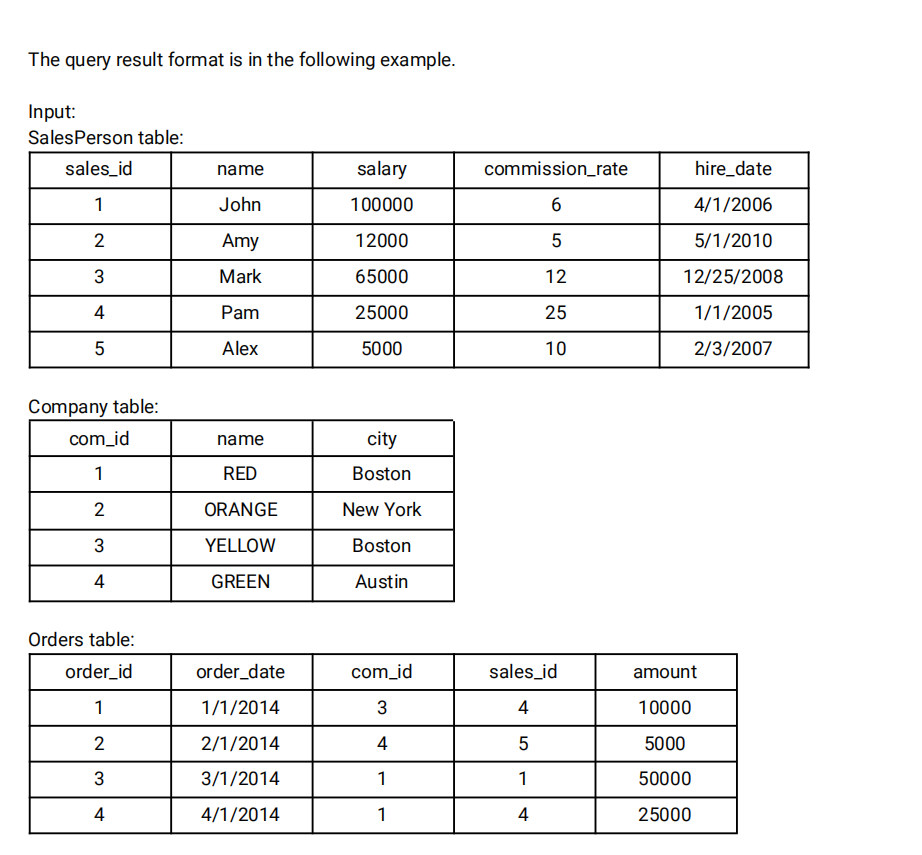
 );

 INSERT INTO cinema (free) VALUES (1),(0),(1),(1),(1);

 select \* from cinema;

 select  s1.seat\_id from cinema s1 join cinema s2 on s1.seat\_id=s2.seat\_id+1 and s1.free <> 0;





create table salesperson (

   sales\_id INT PRIMARY KEY,

   name VARCHAR(50),

   salary INT,

   commission\_rate INT,

   hire\_date DATE

 );

 create Table company (

   com\_id INT PRIMARY KEY,

   name VARCHAR(50),

   city VARCHAR(50)

 );

 create table Orders (

   order\_id INT PRIMARY KEY,

   order\_date DATE,

   com\_id INT,

   sales\_id INT,

   amount INT,

   FOREIGN KEY(com\_id) REFERENCES company(com\_id),

   FOREIGN KEY(sales\_id) REFERENCES salesperson(sales\_id)

 );

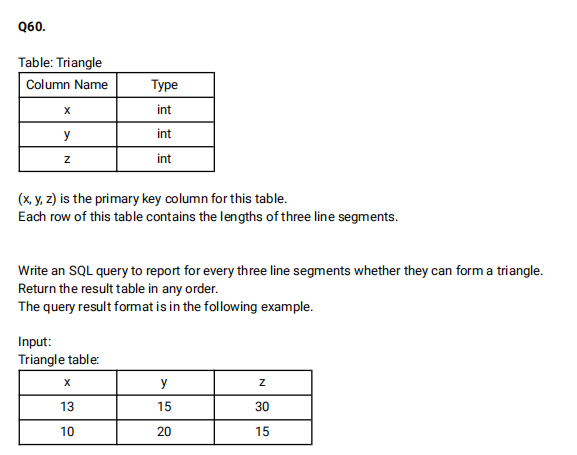
 insert into salesperson VALUES (1,'John',100000,6,STR\_TO\_DATE('04/01/2006','%m/%d/%Y'));

 insert into salesperson VALUES(2,'Amy',12000,5,'2010-05-01'),(3,'Mark',65000,12,'2008-12-25'),(4,'Pam',25000,25,'2005-01-01'),(5,'Alex',5000,10,'2007-02-03');

 INSERT INTO company VALUES (1,'Red','Boston'),(2,'Orange','New York'),(3,'Yellow','Boston'),(4,'Green','Austin');

 INSERT INTO Orders VALUES (1,'2014-01-01',3,4,10000),(2,'2014-01-02',4,5,5000),(3,'2014-01-03',1,1,50000),(4,'2014-01-04',1,4,25000);

 select s.name from salesperson s where s.sales\_id NOT IN (select o.sales\_id from Orders o join company c on o.com\_id=c.com\_id where c.name='Red');



create table Triangle (

  x int,

  y int,

  z int,

  PRIMARY KEY (x,y,z)

 );

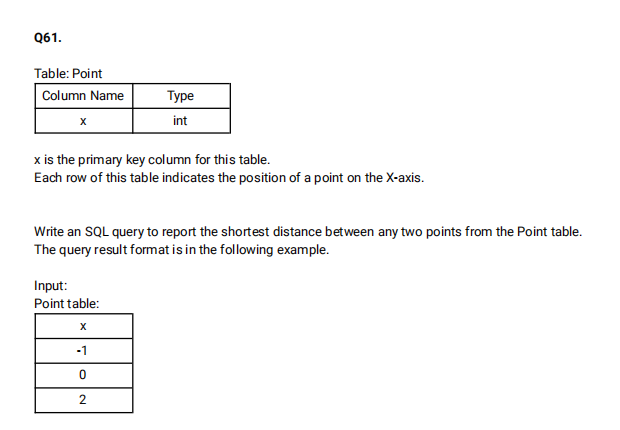
 INSERT INTO Triangle VALUES(13,15,30),(10,20,15);

 SELECT x,y,z,CASE

 when (x+y>z AND x+z>y AND y+z>x) then 'Yes'

 else 'No'

 end as triangle FROM Triangle;



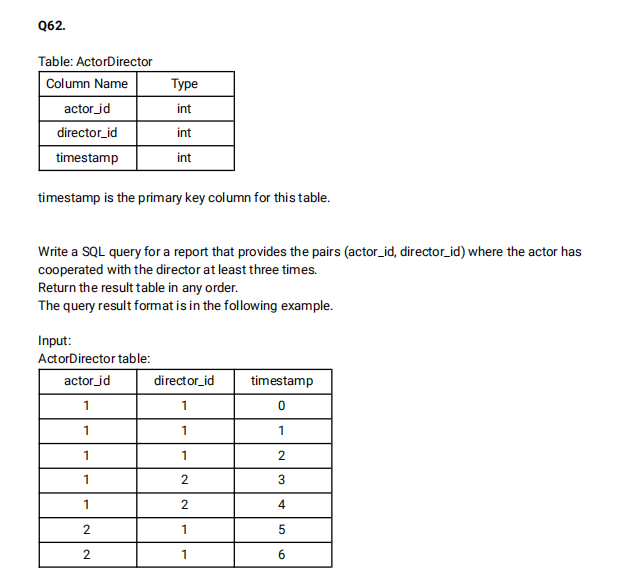
create table point (

  x INT PRIMARY KEY

 );

 insert INTO point VALUES(-1),(0),(2);

  select MIN(shortest) as shortest from (select  LEAD(a.x) OVER(order by a.x) -b.x as shortest from point a join point b on a.x=b.x)c;



create Table Actordirector(

    actor\_id int,

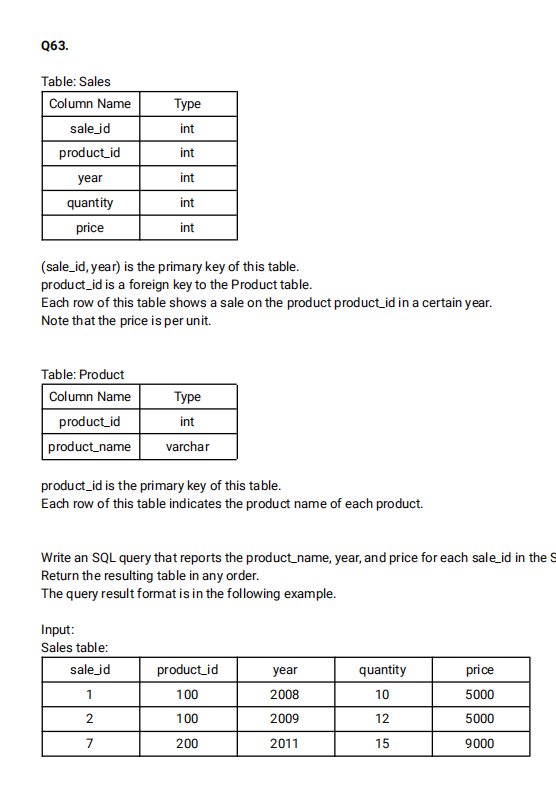
    director\_id int,

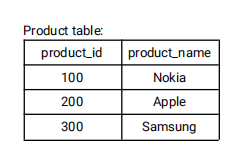
    timestamp int PRIMARY KEY

  );

  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 (select actor\_id,director\_id,COUNT(timestamp) as cnt from Actordirector group by actor\_id,director\_id) a where cnt >=3;





create table Sales (

    sale\_id int,

    product\_id int,

    year int,

    quantity int,

    price int,

    PRIMARY KEY(sale\_id,year)

  );

  create Table product(

    product\_id int PRIMARY KEY,

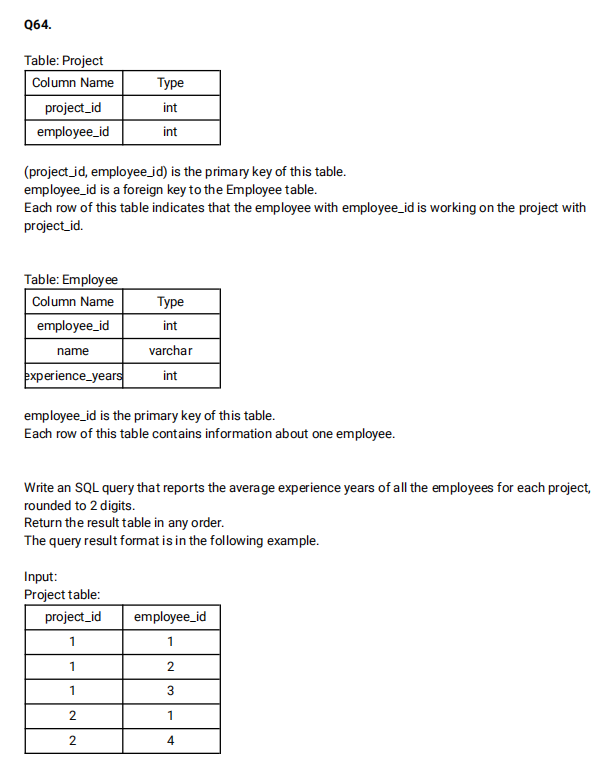
    product\_name VARCHAR(50)

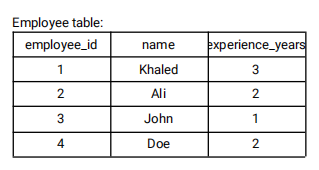
  );

  INSERT INTO Sales VALUES(1,100,2008,10,5000),(2,100,2009,12,5000),(7,200,2011,15,9000);

  INSERT INTO product VALUES(100,'Nokia'),(200,'Apple'),(300,'Samsung');

  select product\_name,year,price from Sales s join product p on s.product\_id=p.product\_id;





create table project(

    project\_id INT,

    employee\_id INT,

    PRIMARY KEY (project\_id,employee\_id)

  );

  create table Employee(

    employee\_id int PRIMARY KEY,

    employee\_name varchar(50),

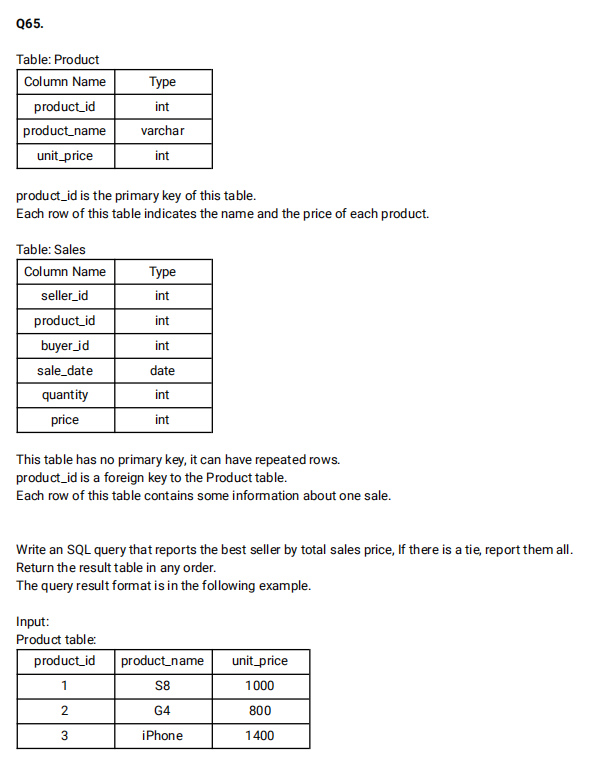
    experience\_years INT

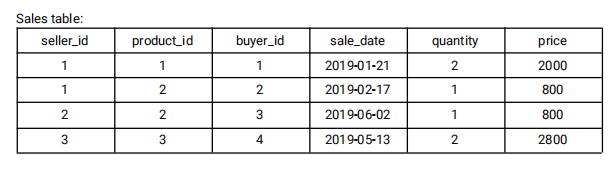
  );

  INSERT into project VALUES (1,1),(1,2),(1,3),(2,1),(2,4);

  INSERT INTO Employee VALUES (1,'Khaled',3),(2,'Ali',2),(3,'John',1),(4,'Doe',2);

  select project\_id,Round(AVG(experience\_years),2) as average\_years from Employee e join project p on e.employee\_id=p.employee\_id group by project\_id;





create table product (

    product\_id INT,

    product\_name VARCHAR(50),

    unit\_price INT

  );

 ALTER table product  ADD CONSTRAINT  PRIMARY KEY(product\_id);

  create table sales(

    seller\_id INT,

    product\_id INT,

    buyer\_id INT,

    sale\_date DATE,

    quantity INT,

    price INT,

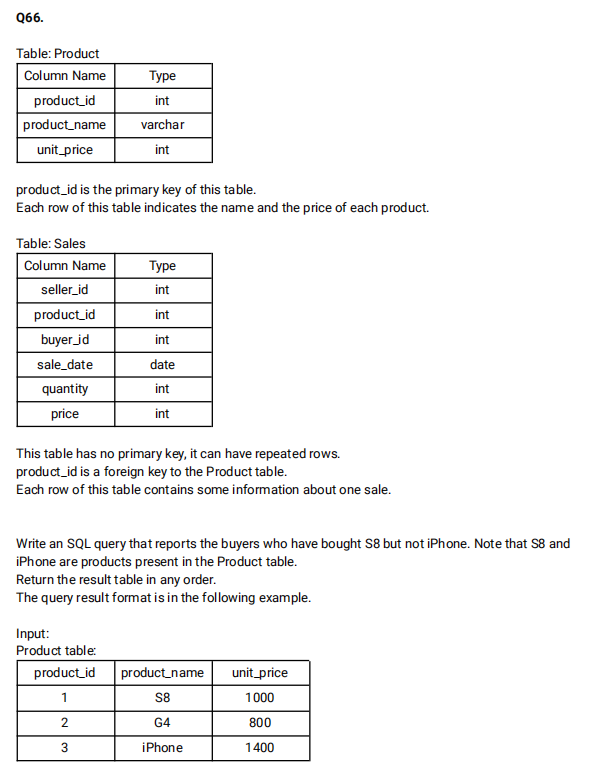
    Foreign Key (product\_id) REFERENCES product(product\_id)

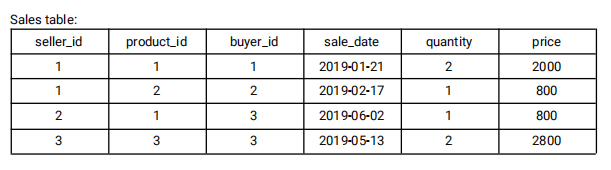
  );

  INSERT INTO product VALUES(1,'S8',1000),(2,'G4',800),(3,'iphone',1400);

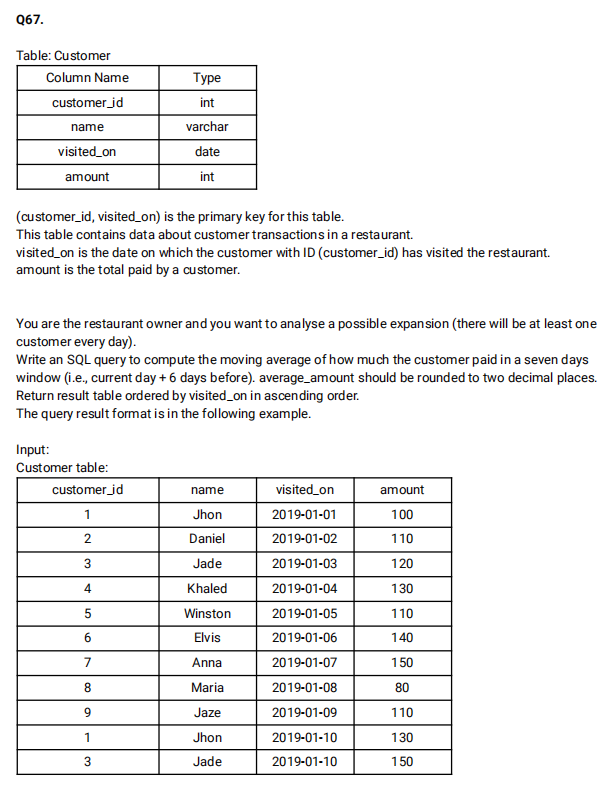
  INSERT INTO sales VALUES(1,1,1,'2019-01-21',2,2000),(1,2,2,'2019-02-17',1,800),(2,2,3,'2019-06-02',1,800),(3,3,4,'2019-05-13',2,2800);

  select seller\_id from (select seller\_id,sum,RANK() over( order by sum desc) as rnk from (select seller\_id, SUM(price) as sum from sales s join product p on s.product\_id=p.product\_id group by seller\_id)a)b where rnk=1 ;





select buyer\_id from sales s join product p on s.product\_id=p.product\_id where p.product\_name ='S8' AND p.product\_name <> 'iphone';



create table customer (

    customer\_id INT,

    name varchar(50),

    visited\_on DATE,

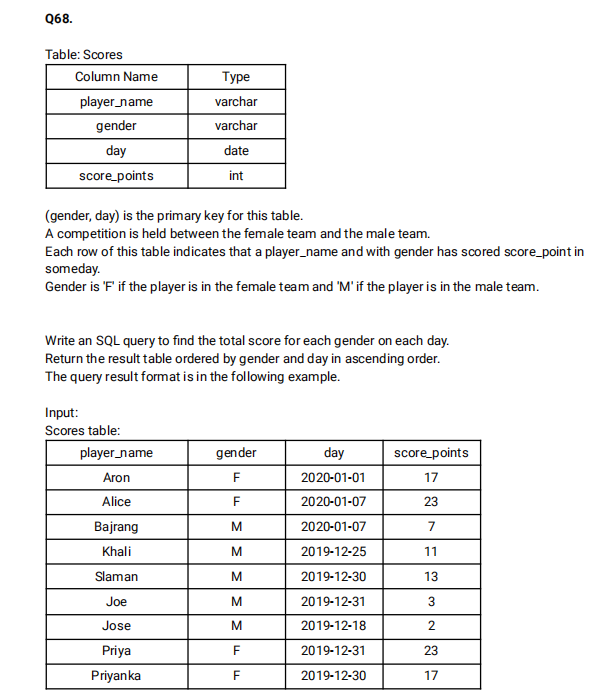
    amount INT

  );

  ALTER table customer ADD constraint PRIMARY KEY(customer\_id,visited\_on);

  insert into customer VALUES (1,'John','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,'John','2019-01-10',130),(3,'Jade','2019-01-10',150);

 select c1.visited\_on, SUM(c1.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;



create table Scores (

  player\_name varchar(100),

  gender varchar(10),

  day date,

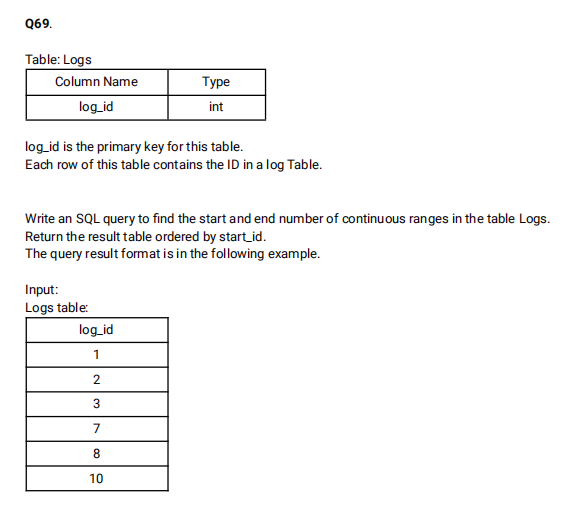
  score\_points int,

  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) as total from Scores;



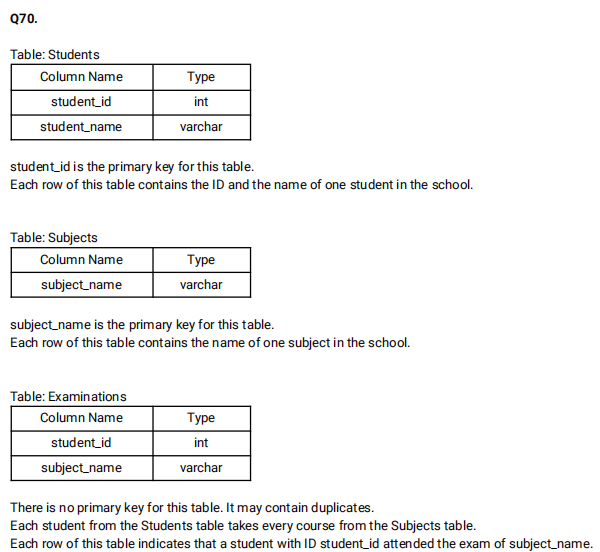
create table Logs (

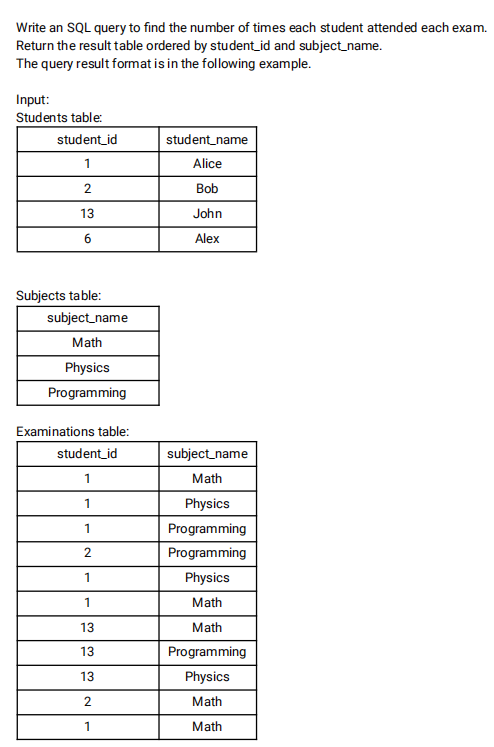
  log\_id int PRIMARY KEY

 );

 INSERT INTO Logs VALUES (1),(2),(3),(7),(8),(10);

 select min(log\_id) as start\_id,max(log\_id) as end\_id from (select log\_id,rn,log\_id-rn as diff from (select log\_id, ROW\_NUMBER() OVER() as rn from Logs)l1)l2 GROUP BY diff;





create table examinations(

  student\_id INT,

  subject\_name varchar(20)

 );

 create table subjects(

  subject\_name varchar(20) PRIMARY KEY

 );

 ALTER Table examinations RENAME COLUMN subject\_name to student\_name;

 ALTER Table examinations RENAME COLUMN student\_name to subject\_name;

 create table students(

  student\_id INT primary KEY,

  student\_name varchar(25)

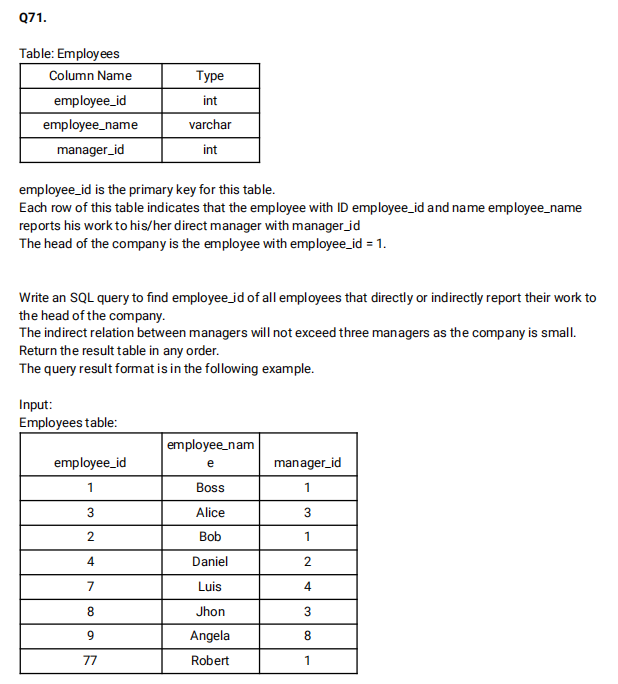
 );

 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');

 select s.student\_id,student\_name,s1.subject\_name,count(e.student\_id) as attended\_exams from students s join subjects s1 left join examinations e on e.student\_id=s.student\_id and s1.subject\_name=e.subject\_name GROUP BY s.student\_id,student\_name,subject\_name order by s.student\_id,subject\_name;



create table Employees(

  employee\_id INT PRIMARY KEY,

  employee\_name VARCHAR(50),

  manager\_id INT

 );

 INSERT INTO Employees VALUES (1,'Boss',1),(3,'Alice',3),(2,'Bob',1),(4,'Daniel',2),(7,'Luis',4),(8,'John',3),(9,'Angela',8),(77,'Robert',1);

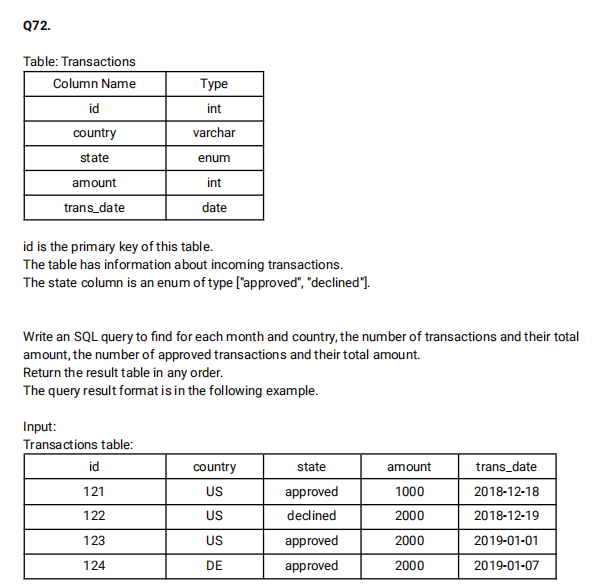
 select e4.emp\_id  FROM

((select e2.employee\_id as e\_id,e1.manager\_id as m\_id from Employees e1 join Employees e2 where e2.manager\_id=e1.employee\_id and e2.employee\_id <>1)e3

  RIGHT join

(select e2.employee\_id as emp\_id,e1.manager\_id as mng\_id from Employees e1 join Employees e2 where e2.manager\_id=e1.employee\_id and e2.employee\_id <>1)e4

 ON e4.mng\_id=e3.e\_id) where m\_id is null or m\_id=1;



create table Transactions(

  id INT PRIMARY KEY,

  country VARCHAR(20),

  state ENUM('approved','declined'),

  amount INT,

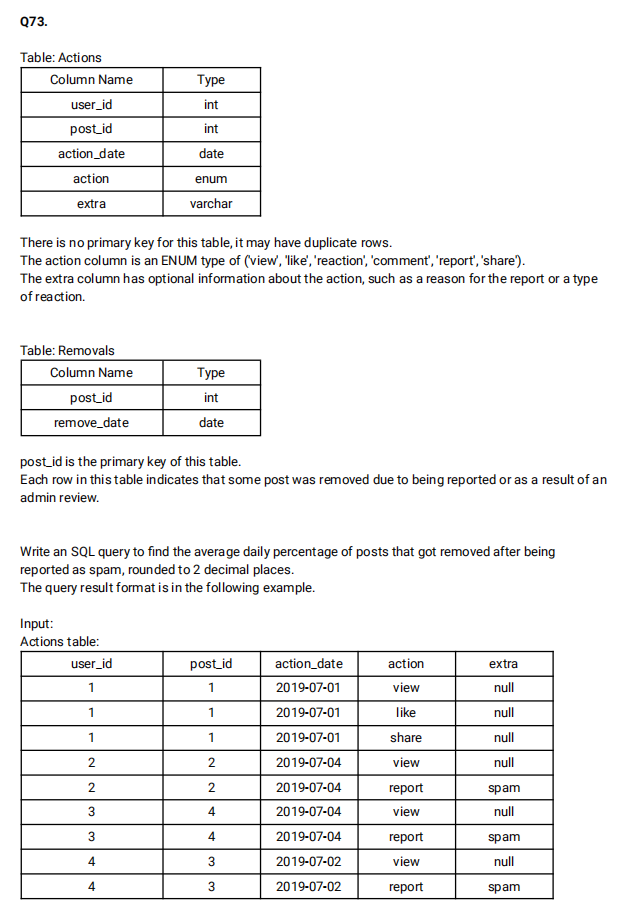
  trans\_date date

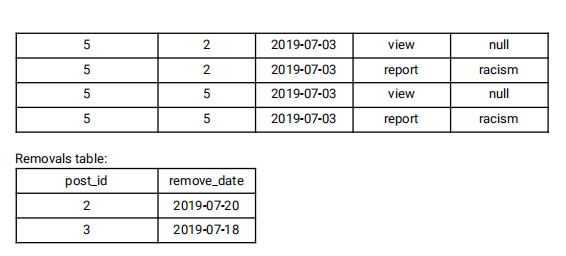
);

INSERT INTO Transactions VALUES(121,"US",'approved',1000,'2018-12-18');

INSERT INTO Transactions VALUES(122,"US",'declined',2000,'2018-12-19'),(123,"US",'approved',2000,'2019-01-01'),(124,"DE",'approved',2000,'2019-01-07');

select EXTRACT(YEAR\_MONTH from trans\_date) as month,country,count(amount) as trans\_count,sum(case when state='approved' then 1 else 0 end) as approved\_count,sum(amount) as trans\_total\_amount, sum(case when state='approved' then amount else 0 end)as approved\_amount from Transactions GROUP BY month,country





create table Actions(

  user\_id int,

  post\_id int,

  action\_date date,

  action enum('view','like','reaction','comment','report','share'),

  extra varchar(10)

  );

  create table removals(

    post\_id INT PRIMARY KEY,

    remove\_date date

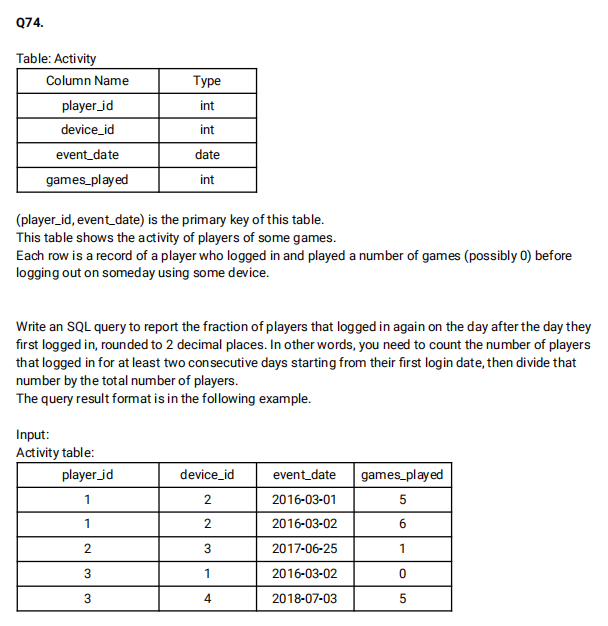
  );

  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');

  insert into Actions VALUES(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(cnt1),2) as avg\_daily\_percent from (select (count(b.post\_id)\*100)/(count(a.post\_id)) as cnt1 from Actions a left join removals b on a.post\_id=b.post\_id where extra='spam' group by action\_date)c;



create table Activity(

  player\_id INT,

  device\_id INT,

  event\_date date,

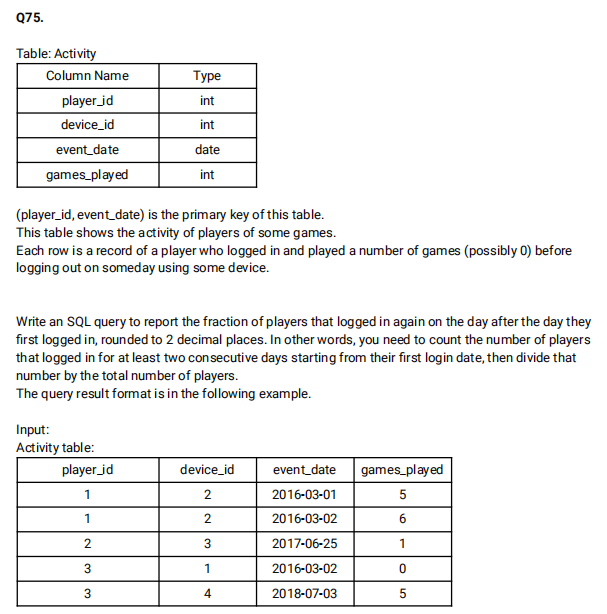
  games\_played INT,

  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);

select ROUND(count(distinct b.player\_id)/count(distinct a.player\_id),2) as fraction from Activity a left join Activity b on b.player\_id=a.player\_id and a.device\_id=b.device\_id and DATEDIFF(a.event\_date,b.event\_date)<>0



create table Activity(

  player\_id INT,

  device\_id INT,

  event\_date date,

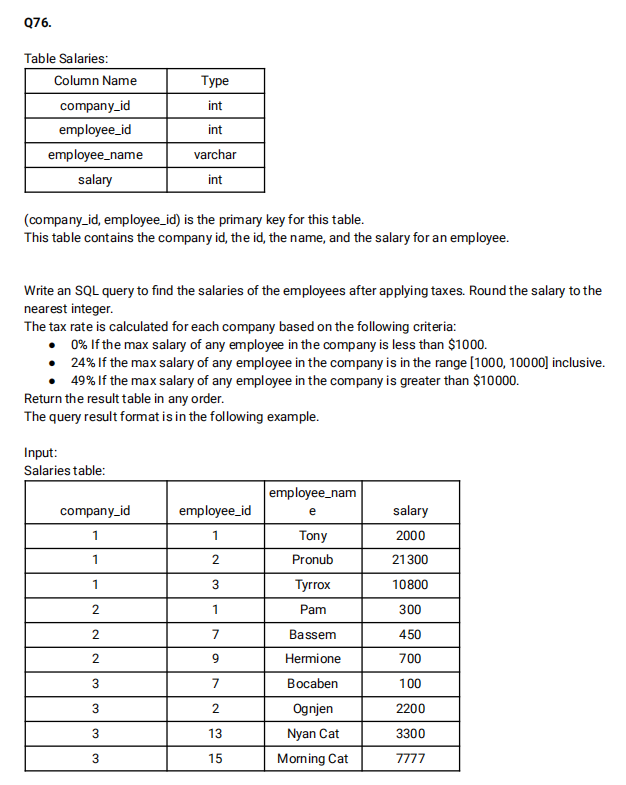
  games\_played INT,

  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);

select ROUND(count(distinct b.player\_id)/count(distinct a.player\_id),2) as fraction from Activity a left join Activity b on b.player\_id=a.player\_id and a.device\_id=b.device\_id and DATEDIFF(a.event\_date,b.event\_date)<>0;



create table salaries(

  company\_id INT,

  employee\_id INT,

  employee\_name varchar(25),

  salary INT,

  PRIMARY KEY (company\_id,employee\_id)

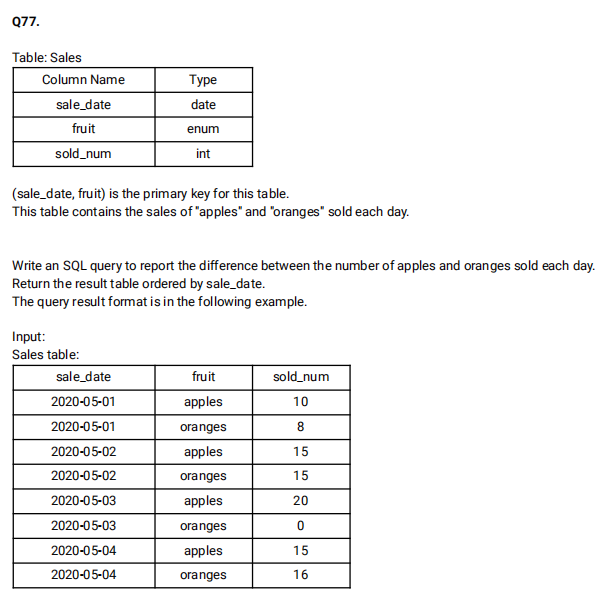
);

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);

select s1.company\_id,s1.employee\_id,s1.employee\_name, ROUND(CASE when s2.max\_salary BETWEEN 1000 AND 10000 then 0.76\*salary

                                                  when s2.max\_salary>10000 then 0.51\*salary

                                                  else salary end) as salary from salaries s1 join (select company\_id,MAX(salary) as max\_salary from salaries s2 group by company\_id)s2 on s1.company\_id=s2.company\_id order by company\_id;



create table Sales(

  sale\_date date,

  fruit ENUM('apples','oranges'),

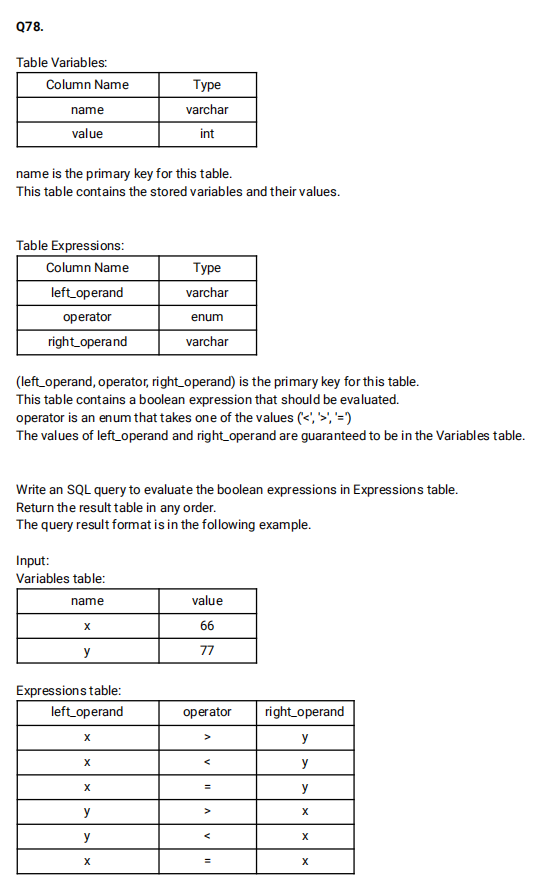
  sold\_num int,

  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 else 0 end) - sum(case when fruit='oranges' then sold\_num else 0 end)) as diff from Sales GROUP BY sale\_date;



create table Variables (

  name varchar(50),

  value INT,

  PRIMARY KEY(name)

);

create table Expressions(

  left\_operand varchar(2),

  operator ENUM('<','>','='),

  right\_operand VARCHAR(2),

  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');

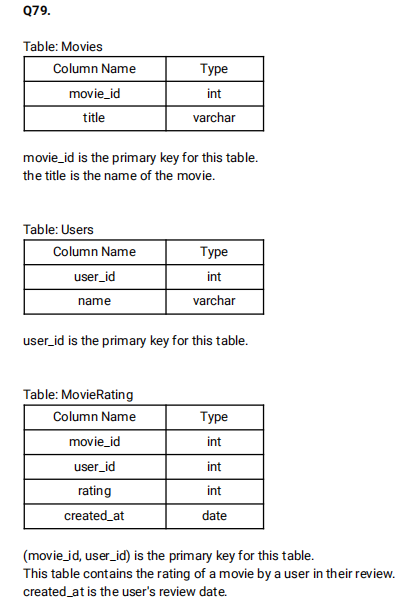
select left\_operand,operator,right\_operand,

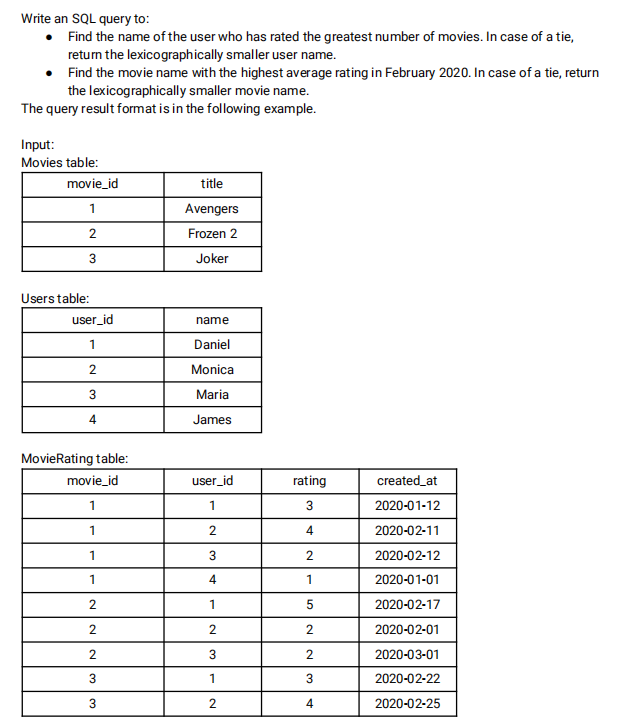
(case when operator='<' then if(v1.value<v2.value,'true','false')

when operator='>' then if(v1.value>v2.value,'true','false')

else if(v1.value=v2.value,'true','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;





create table movies (

  movie\_id INT PRIMARY KEY,

  title VARCHAR(20)

);

create table Users(

  user\_id INT PRIMARY KEY,

  name VARCHAR(50)

);

create table movierating(

  movie\_id INT,

  user\_id INT,

  rating INT,

  created\_at DATE,

  PRIMARY KEY(movie\_id,user\_id)

);

INSERT INTO movies VALUES(1,'Avengers'),(2,'Frozen2'),(3,'Joker');

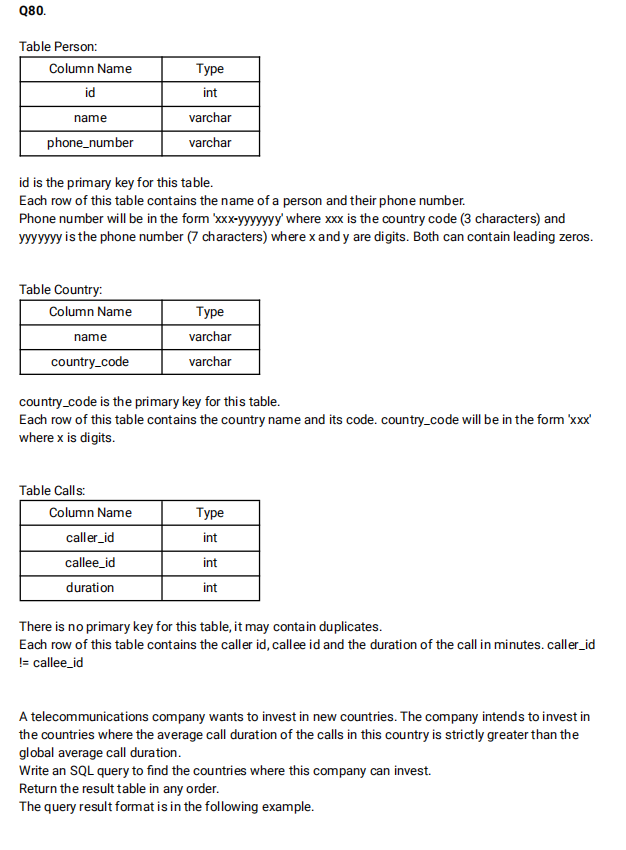
INSERT INTO Users VALUES(1,'Daniel'),(2,'Monica'),(3,'Maria'),(4,'James');

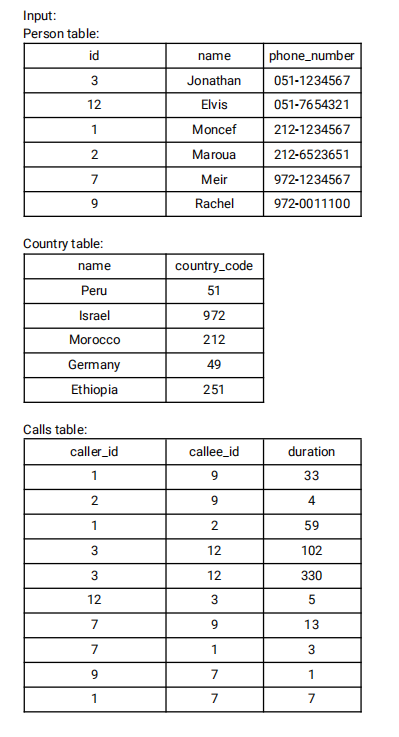
INSERT INTO movierating VALUES(1,1,3,'2020-01-12'),(1,2,4,'2020-02-11'),(1,3,2,'2020-02-12'),(1,4,1,'2020-01-01'),(2,1,5,'2020-02-17'),(2,2,2,'2020-02-01'),(2,3,2,'2020-03-01'),(3,1,3,'2020-02-22'),(3,2,4,'2020-02-25');

select a.name as results from (select u.name,count(mr.rating) as cnt from Users u join movierating mr on mr.user\_id=u.user\_id group by u.name order by cnt desc,u.name limit 1)a

union ALL

select b.title as results from (select title,AVG(mr.rating) as avg from movies m join movierating mr on mr.movie\_id=m.movie\_id where Extract(YEAR\_MONTH from created\_at)='202002' group by title order by avg desc,title limit 1)b;





create table person(

    id INT,

    name VARCHAR(255),

    phone\_number VARCHAR(255)

 );

 create table country(

    name VARCHAR(50),

    country\_code VARCHAR(10) PRIMARY KEY

 );

 ALTER Table person ADD constraint PRIMARY KEY(id);

 create table calls(

    caller\_id INT,

    callee\_id INT,

    duration INT

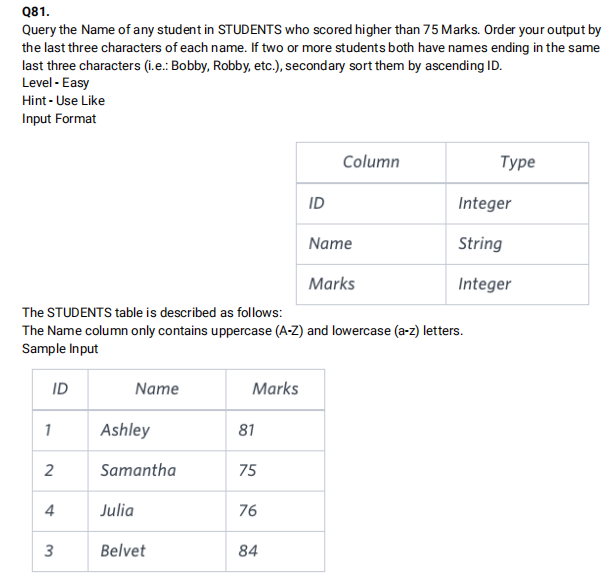
 );

 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','49'),('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 c.name as country from person p join country c on SUBSTRING(phone\_number,1,3)=c.country\_code join calls ca on p.id IN (ca.caller\_id,ca.callee\_id) group by c.name HAVING AVG(duration)>(select AVG(duration) from calls);



create Table Students(

  ID INT,

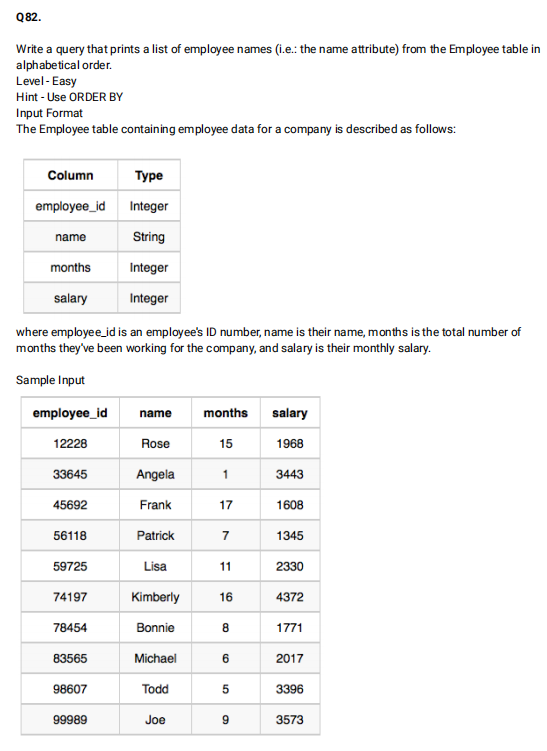
  Name VARCHAR(30),

  Marks INT

);

INSERT INTO Students VALUES(1,'Ashley',81),(2,'Samantha',75),(4,'Julia',76),(3,'Belvet',84);

select name from Students where marks>75 order by right(name,3),id;



create table Employees(

  employee\_id INT,

  name VARCHAR(30),

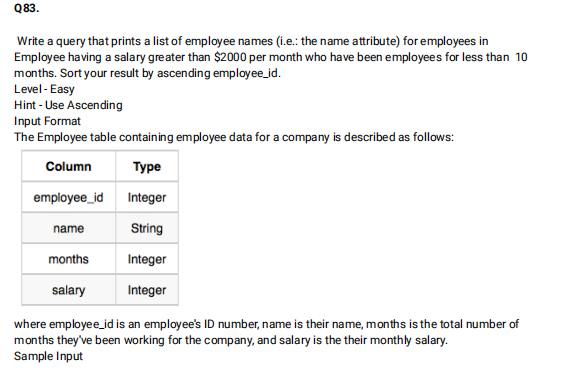
  months INT,

  salary INT

);

INSERT INTO Employees VALUES(12228,'Rose',15,1968),(33645,'Angela',1,3443),(45692,'Frank',17,1608),(56118,'Pattrick',7,1345),(59725,'Lisa',11,2330),(74197,'Kimberly',16,4372),(78454,'Bonnie',8,1771),(83565,'Michael',6,2017),(98607,'Todd',5,3396),(99989,'Joe',9,3573);

select name from Employees order by name;





create table Employees(

  employee\_id INT,

  name VARCHAR(30),

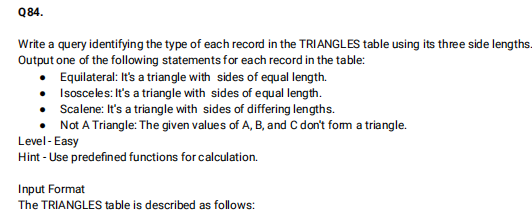
  months INT,

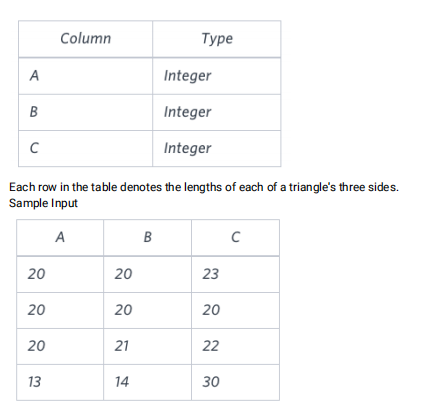
  salary INT

);

INSERT INTO Employees VALUES(12228,'Rose',15,1968),(33645,'Angela',1,3443),(45692,'Frank',17,1608),(56118,'Pattrick',7,1345),(59725,'Lisa',11,2330),(74197,'Kimberly',16,4372),(78454,'Bonnie',8,1771),(83565,'Michael',6,2017),(98607,'Todd',5,3396),(99989,'Joe',9,3573);

select name from Employees where months<10 and salary >2000;





create table Triangle(

  A int,

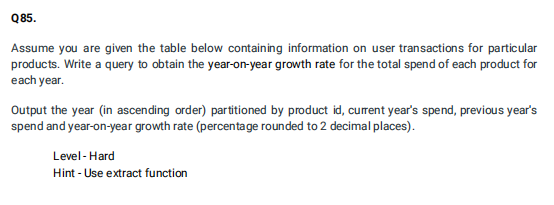
  B INT,

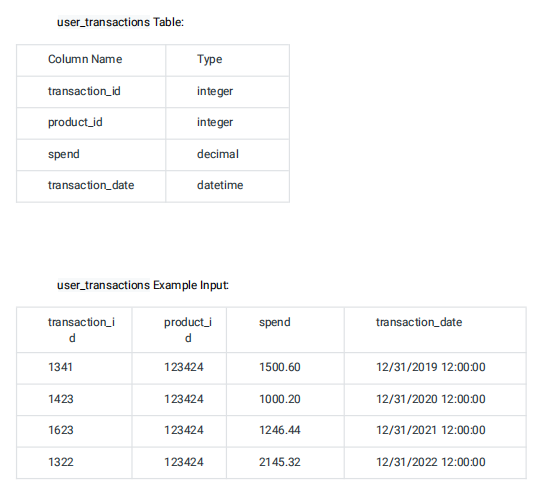
  C INT

);

INSERT INTO Triangle VALUES(20,20,23),(20,20,20),(20,21,22),(13,14,30);

select IF((A+B<C or B+C<A or A+C<B),'Not a Triangle',(case when A=B=C then 'Equilateral' when A<>B and B<>C then 'Scalene' else 'Isoceles' end)) as Type from Triangle;





create table user\_transactions(

  transaction\_id INT,

  product\_id INT,

  spend DECIMAL(6,2),

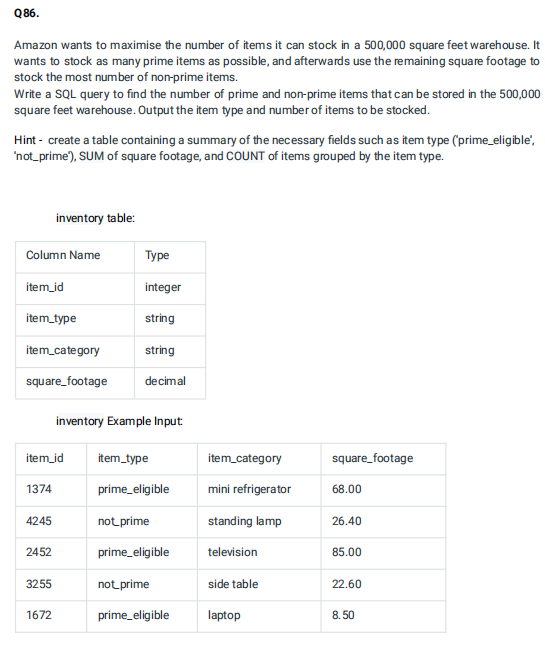
  transaction\_date datetime

);

INSERT INTO user\_transactions VALUES(1341,123424,1500.60,'2019-12-31 12:00:00');

INSERT INTO user\_transactions VALUES(1423,123424,1000.20,'2020-12-31 12:00:00'),(1623,123424,1246.44,'2021-12-31 12:00:00'),(1322,123424,2145.32,'2022-12-31 12:00:00');

select year,curr\_year\_spend,prev\_year\_spend,ROUND((((curr\_year\_spend-prev\_year\_spend)\*100)/prev\_year\_spend),2) as yoy\_rate from (select Extract(year from transaction\_date) as year,spend as curr\_year\_spend,lag(spend) over(order by EXTRACT(year from transaction\_date)) as prev\_year\_spend from user\_transactions order by year)a;



create table Inventory(

  item\_id INT,

  item\_type VARCHAR(20),

  item\_category VARCHAR(20),

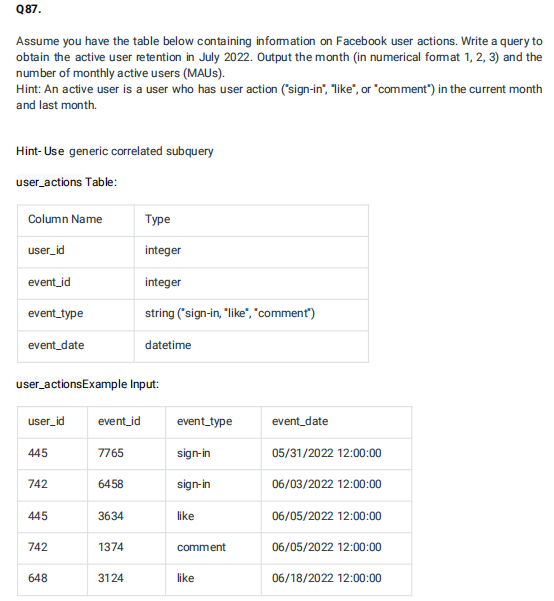
  square\_footage DECIMAL(6,2)

);

INSERT INTO Inventory VALUES(1374,'prime eligible','mini refrigerator',68.00);

INSERT into Inventory VALUES(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);

select item\_type,count(item\_id) as item\_count from Inventory where square\_footage<500000 GROUP BY item\_type;



create table user\_actions(

  user\_id INT,

  event\_id INT,

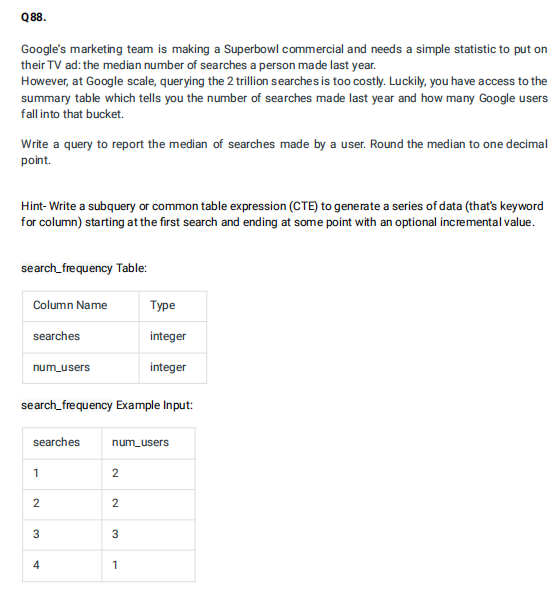
  event\_type ENUM('like','comment','sign-in'),

  event\_date DATETIME

);

INSERT INTO user\_actions VALUES(445,7765,'sign-in','2022-05-31 12:00:00'),(742,6458,'sign-in','2022-06-03 12:00:00'),(445,3634,'like','2022-06-05 12:00:00'),(742,1374,'comment','2022-06-05 12:00:00'),(648,3124,'like','2022-06-18 12:00:00');

select EXTRACT(MONTH from event\_date) as month,sum(case when event\_type='like' or event\_type='comment' or event\_type='sign-in' then 1 else 0 end) as monthly\_active\_users from user\_actions group by month;



create table search\_frequency (

  searches INT,

  num\_users INT

);

DELETE from search\_frequency where searches=1 or searches=2 or searches=3 or searches=4;

INSERT INTO search\_frequency VALUES (1,2),(2,2),(3,3),(4,1);

alter table search\_frequency modify column searches INTEGER;

select AVG(searches) from (select searches, ROW\_NUMBER() OVER(order by searches) as rn\_asc,ROW\_NUMBER() OVER(order by searches desc) rn\_desc from search\_frequency order by searches)a where ABS(rn\_asc-rn\_desc)<=1;





create table advertiser (

  user\_id VARCHAR(10),

  status VARCHAR(15)

);

create table daily\_pay(

  user\_id VARCHAR(10),

  paid DECIMAL(6,2)

);

INSERT INTO advertiser VALUES('bing','New'),('yahoo','New'),('alibaba','existing');

INSERT INTO daily\_pay VALUES('yahoo',45.00),('alibaba',100.00),('target',13.00);

WITH pay\_status as

 ((select a.user\_id, a.status ,paid from advertiser a left join daily\_pay d on d.user\_id=a.user\_id)

UNION

(select d.user\_id,a.status,paid from daily\_pay d left join advertiser a on d.user\_id=a.user\_id))

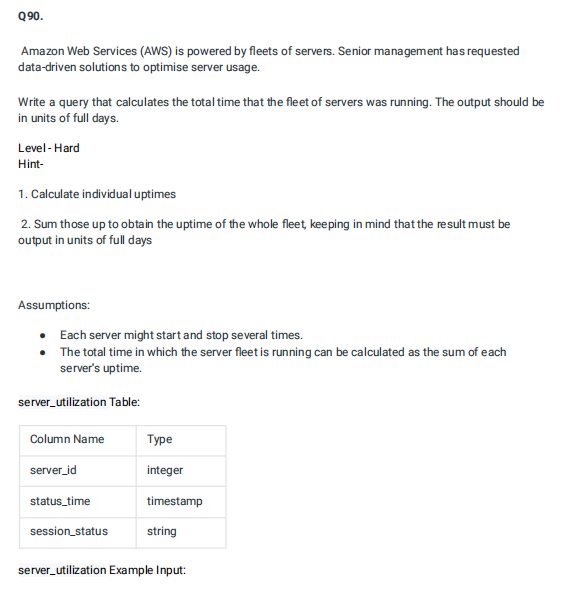
select user\_id, (case when paid is NULL then 'Churn'

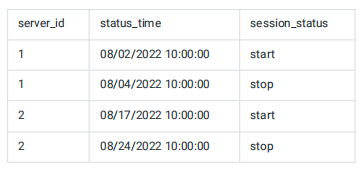
when STATUS ='Churn' and paid is not NULL then 'Resurrect'

when STATUS <> 'Churn' and paid is NOT NULL then 'Existing'

when STATUS is NULL then 'NEW' end) as new\_status

FROM pay\_status;





create table server\_utilization(

  server\_id INT,

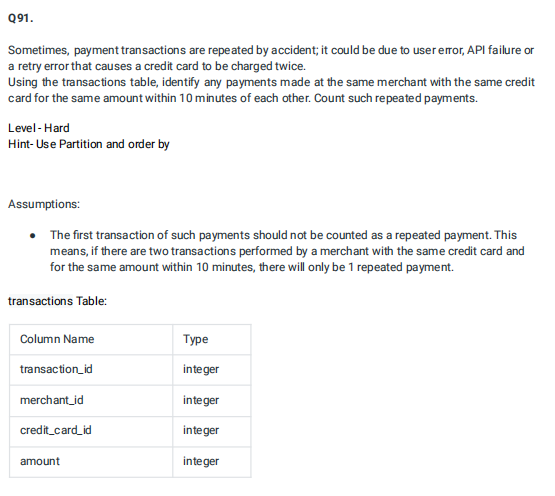
  status\_time TIMESTAMP,

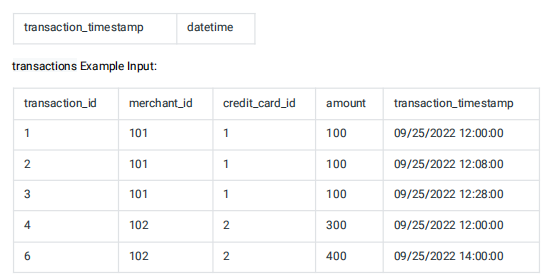
  session\_status VARCHAR(20)

);

INSERT INTO server\_utilization VALUES(1,'2022-08-02 10:00:00','start'),(1,'2022-08-04 10:00:00','stop'),(2,'2022-08-17 10:00:00','start'),(2,'2022-08-24','stop');

select sum(diff) as total\_uptime\_days from (select  s2.server\_id,DATEDIFF(s2.status\_time,s1.status\_time) as diff from server\_utilization s1 join server\_utilization s2 on s2.server\_id=s1.server\_id where s2.session\_status='stop')a;





create table transactions (

  transaction\_id INT,

  merchant\_id INT,

  credit\_card\_id INT,

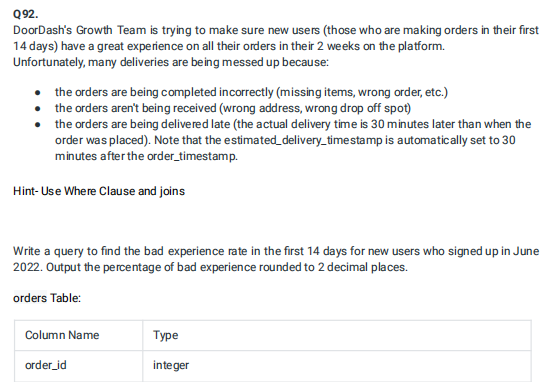
  amount INT,

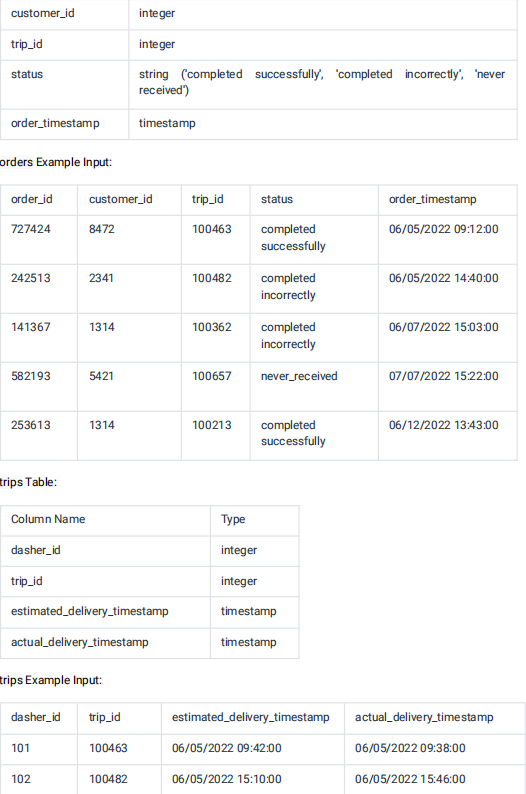
  transaction\_timestamp DATETIME

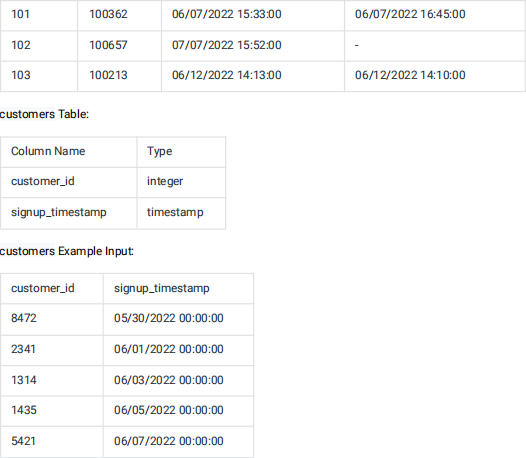
);

INSERT INTO transactions VALUES(1,101,1,100,'2022-09-25 12:00:00'),(2,101,1,100,'2022-09-25 12:08:00'),(3,101,1,100,'2022-09-25 12:28:00'),(4,101,1,300,'2022-09-25 12:00:00'),(6,102,2,400,'2022-09-25 14:00:00');

select count(distinct amount) as payment\_count from (select  t2.transaction\_id,t2.amount,TIMESTAMPDIFF(MINUTE,t2.transaction\_timestamp,t1.transaction\_timestamp) as t from transactions t1 join transactions t2 WHERE t1.transaction\_id <> t2.transaction\_id AND t1.merchant\_id=t2.merchant\_id AND t1.credit\_card\_id=t2.credit\_card\_id AND t1.amount=t2.amount)a where abs(t)<=10;







create table orders (

  order\_id INT,

  customer\_id INT,

  trip\_id INT,

  status ENUM('completed successfully','completed incorrectly','never recieved'),

  order\_timestamp timestamp

);

create table trips (

  dasher\_id INT,

  trip\_id INT,

  estimated\_delivery\_timestamp TIMESTAMP,

  actual\_delivery\_timestamp TIMESTAMP

);

create table customers (

  customer\_id INT,

  signup\_timestamp TIMESTAMP

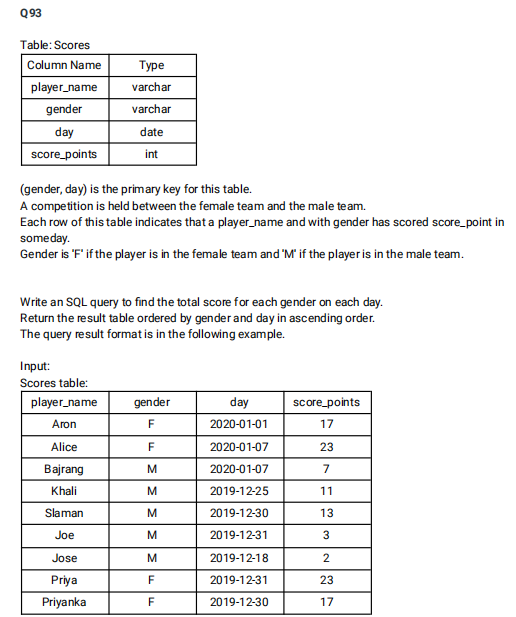
);

INSERT INTO orders VALUES(727424,8472,100463,'completed successfully','2022-06-05 09:12:00'),(242513,2341,100482,'completed incorrectly','2022-06-05 14:40:00'),(141367,1314,100362,'completed incorrectly','2022-06-07 15:03:00'),(582193,5421,100657,'never recieved','2022-07-07 15:22:00'),(253613,1314,100213,'completed successfully','2022-06-12 13:43:00');

INSERT INTO trips VALUES(101,100463,'2022-06-05 09:42:00','2022-06-05 15:46:00'),(102,100482,'2022-06-05 15:10:00','2022-06-05 15:46:00'),(101,100362,'2022-06-07 15:33:00','2022-06-07 16:45:00'),(102,100657,'2022-07-07 15:52:00',NULL),(103,100213,'2022-06-12 14:13:00','2022-06-12 14:10:00');

INSERT INTO customers VALUES(8472,'2022-05-30 00:00:00'),(2341,'2022-06-01 00:00:00'),(1314,'2022-06-03 00:00:00'),(1435,'2022-06-05 00:00:00'),(5421,'2022-06-07 00:00:00');

select (((cnt-good\_delivery)\*100)/cnt) as bad\_pct from (select COUNT(a.trip\_id) as cnt,sum(case when a.status='completed successfully' AND TIMESTAMPDIFF(MINUTE,estimated\_delivery\_timestamp,actual\_delivery\_timestamp)>=0 then 1 else 0 end) as good\_delivery from orders a left join trips b on b.trip\_id=a.trip\_id)a;



create table Scores (

  player\_name varchar(100),

  gender varchar(10),

  day date,

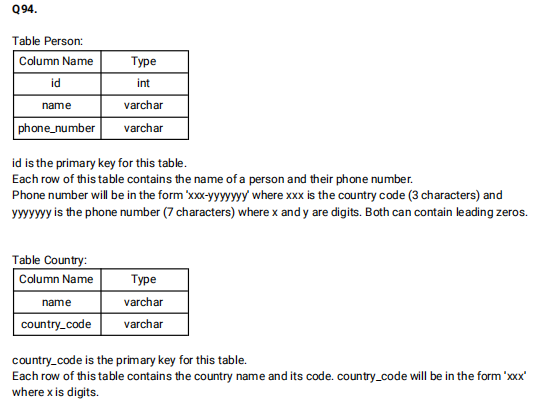
  score\_points int,

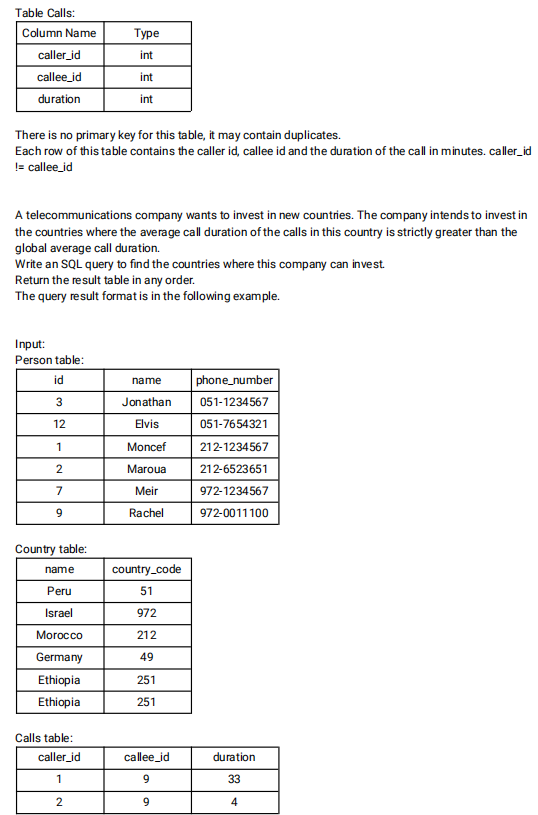
  PRIMARY KEY(gender,day)

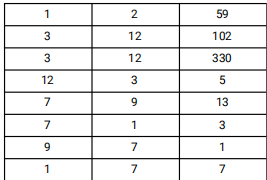
 );

 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) as total from Scores;







create table person(

    id INT,

    name VARCHAR(255),

    phone\_number VARCHAR(255)

 );

 create table country(

    name VARCHAR(50),

    country\_code VARCHAR(10) PRIMARY KEY

 );

 ALTER Table person ADD constraint PRIMARY KEY(id);

 create table calls(

    caller\_id INT,

    callee\_id INT,

    duration INT

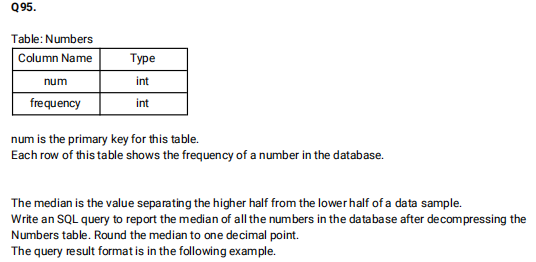
 );

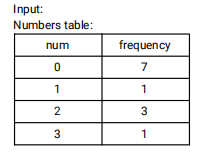
 INSERT INTO person VALUES(3,'Jonathan','051-1234567'),(12,'Elvis','051-7654321'),(1,'Moncef','212-1234567'),(2,'Maroua','212-6523651'),(7,'Meir','972-1234567'),(9,'Rachel','972-0011100');

 INSERT INTO country VALUES ('Peru','051'),('Israel','972'),('Morocco','212'),('Germany','49'),('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 c.name as country from person p join country c on SUBSTRING(phone\_number,1,3)=c.country\_code join calls ca on p.id IN (ca.caller\_id,ca.callee\_id) group by c.name HAVING AVG(duration)>(select AVG(duration) from calls);





create table numbers(

  num INT,

  freq INT

);

INSERT INTO numbers VALUES(0,7),(1,1),(2,3),(3,1);

with cte as(

  select \* from numbers

  union all

  select num, freq - 1

  from cte where freq > 1

), cte2 as(

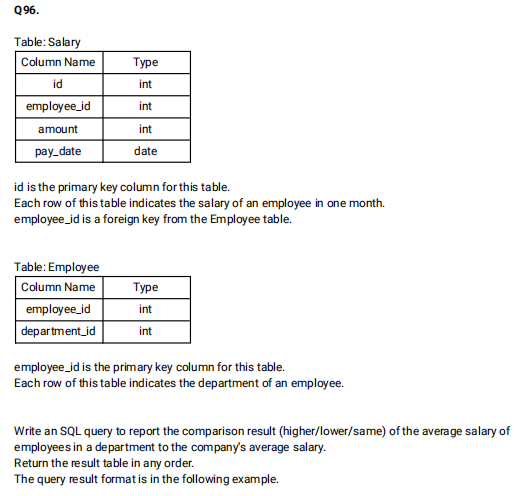
  select \*, rn = row\_number() over(order by num), cn = count(1) over()

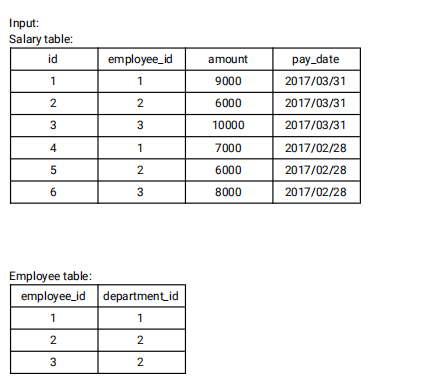
  from cte

)

select avg(num) from cte2

where rn between cn\*1.0/2 and cn\*1.0/2 + 1





create Table Salary(

  id int,

  employee\_id INT,

  amount INT,

  pay\_date DATE

);

create table Employee(

  employee\_id INT,

  department\_id INT

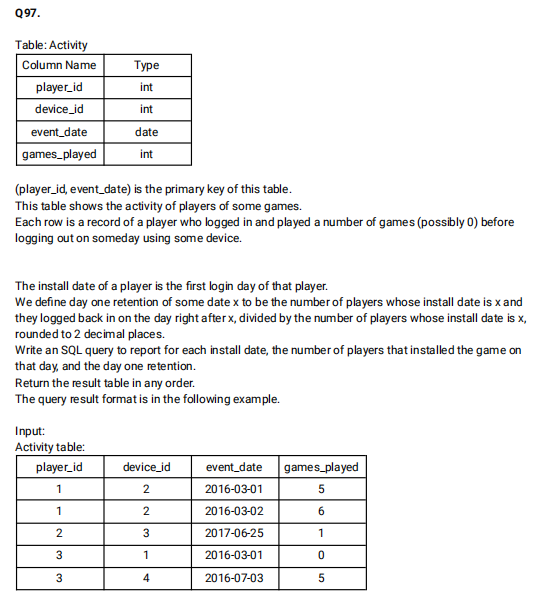
);

INSERT INTO Salary VALUES(1,1,9000,'2017-03-31'),(2,2,6000,'2017-03-31'),(3,3,10000,'2017-03-31'),(4,1,7000,'2017-02-28'),(5,2,6000,'2017-02-28'),(6,3,8000,'2017-02-28');

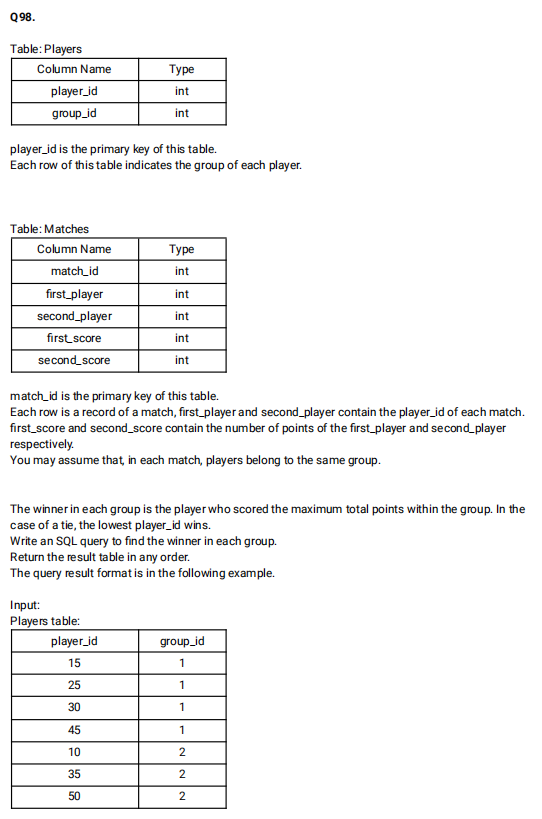
INSERT INTO Employee VALUES(1,1),(2,2),(3,2);

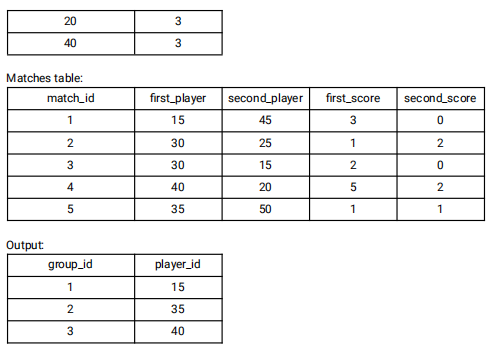
select dept\_salary.pay\_month,dept\_salary.department\_id,(case when dept\_avg>comp\_avg then 'Higher' when dept\_avg<comp\_avg then 'Lower' else 'Same' end) as comparison  from ((select e.department\_id,EXTRACT(YEAR\_MONTH from pay\_date) as pay\_month,AVG(amount) as dept\_avg from Salary s join Employee e on e.employee\_id=s.employee\_id group by e.department\_id,pay\_month) as dept\_salary join

(select AVG(amount) as comp\_avg,EXTRACT(year\_month from pay\_date) as pay\_month from Salary GROUP BY pay\_month) as comp\_salary on dept\_salary.pay\_month=comp\_salary.pay\_month) order by dept\_salary.department\_id;



select a.install\_date,count(a.install\_date) as installs,ROUND(count(b.event\_date)/count(\*),2) from (select player\_id,min(event\_date) as install\_date from Activity GROUP BY player\_id)a left join Activity b on DATE\_ADD(a.install\_date, INTERVAL 1 DAY)=b.event\_date AND a.player\_id=b.player\_id group by a.install\_date order by a.install\_date;





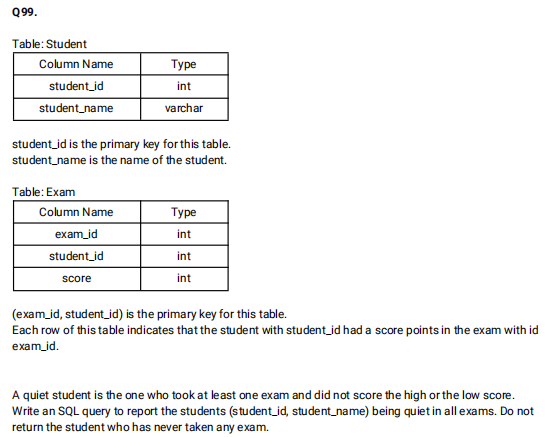
SELECT group\_id,player\_id

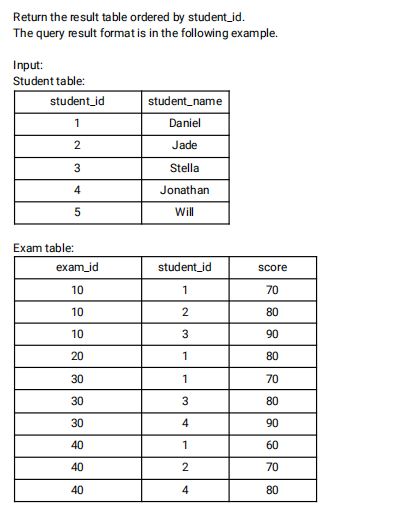
FROM   (SELECT p.group\_id,ps.player\_id,  Sum(ps.score) AS score   FROM players p INNER JOIN (SELECT first\_player AS player\_id,  first\_score  AS score  FROM   matches

  UNION ALL

  SELECT second\_player AS player\_id, second\_score  AS score FROM   matches) ps  ON  p.player\_id = ps.player\_id   GROUP  BY ps.player\_id  ORDER  BY group\_id, score DESC, player\_id) top\_scores

GROUP  BY group\_id;



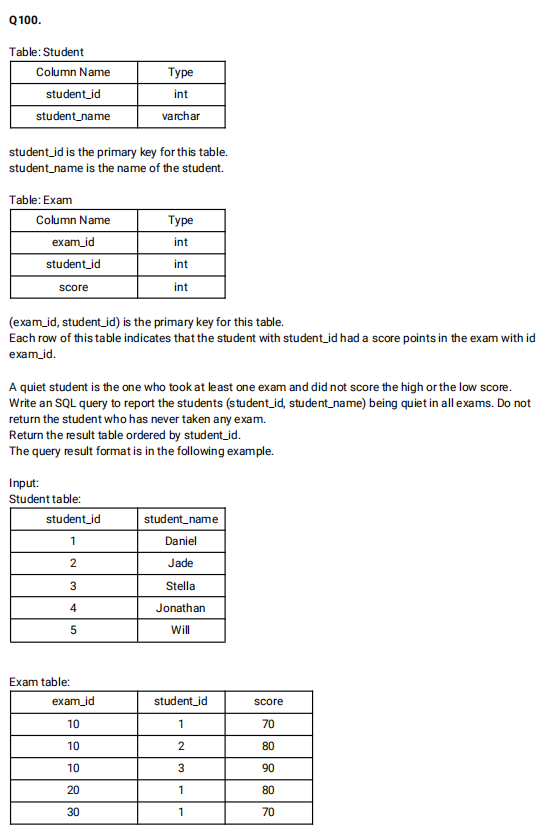


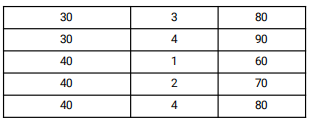
select distinct s.student\_id,s.student\_name

from Student s inner join Exam e

on s.student\_id = e.student\_id

where s.student\_id not in (select e1.student\_id from Exam as e1 inner join (select exam\_id, min(score) as min\_score, max(score) as max\_score from Exam group by exam\_id) as e2 on e1.exam\_id = e2.exam\_id where e1.score = e2.min\_score or e1.score = e2.max\_score) order by student\_id;





select distinct s.student\_id,s.student\_name

from Student s inner join Exam e

on s.student\_id = e.student\_id

where s.student\_id not in (select e1.student\_id from Exam as e1 inner join (select exam\_id, min(score) as min\_score, max(score) as max\_score from Exam group by exam\_id) as e2 on e1.exam\_id = e2.exam\_id where e1.score = e2.min\_score or e1.score = e2.max\_score) order by student\_id;