DBMS ASSIGNMENT -> 1

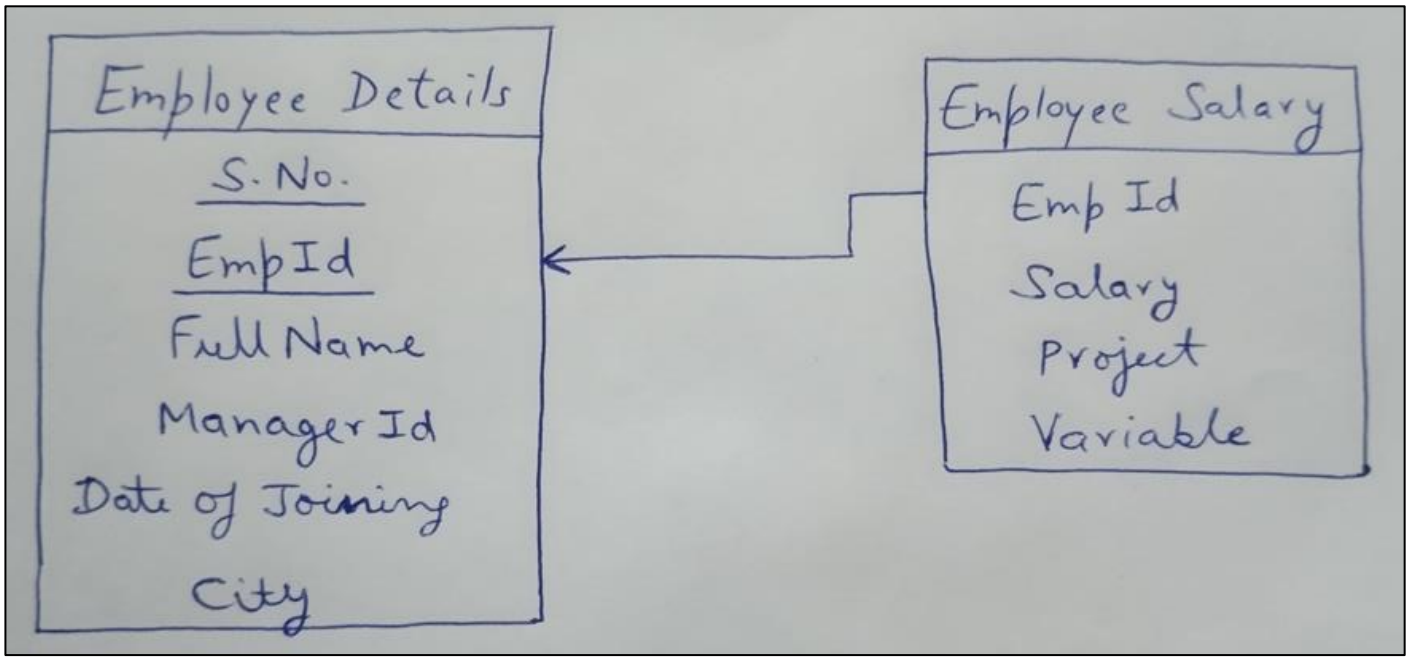
NAME -> RAUNAK GARHWAL

CLASS -> MCA 1st SEM

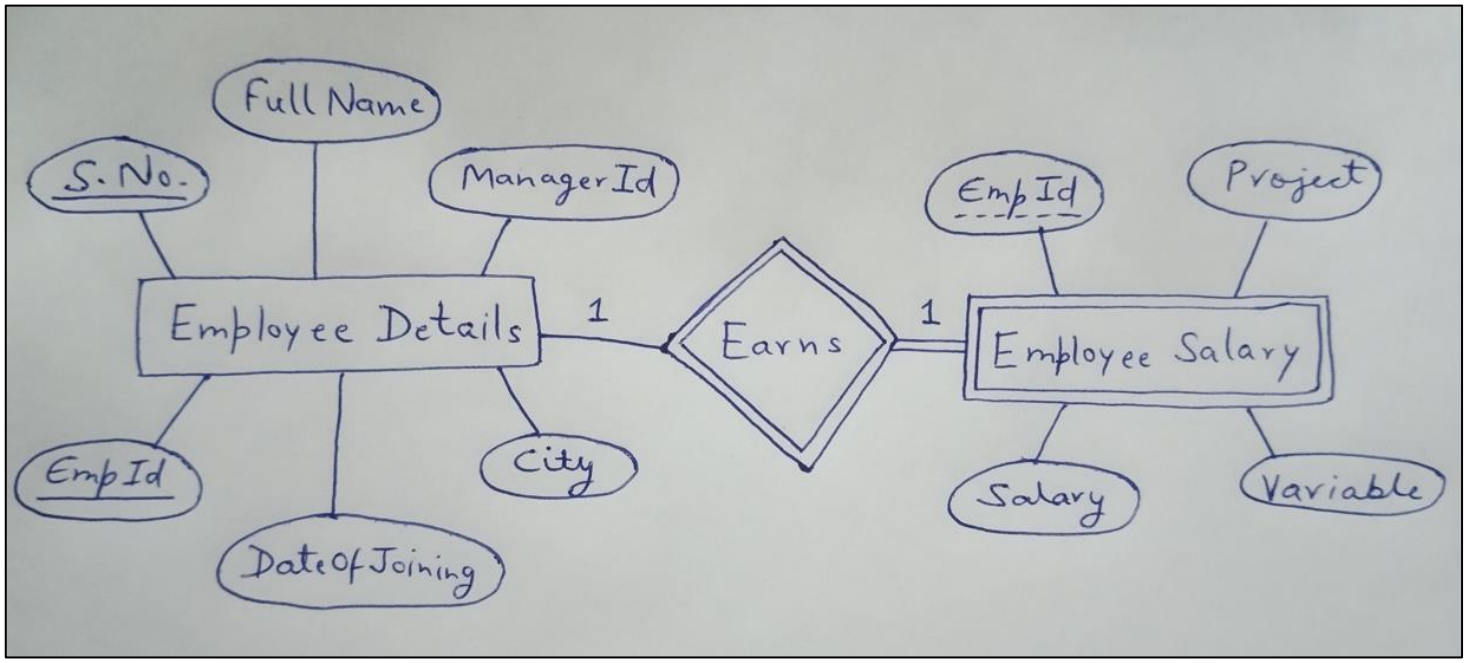
ROLL NO. -> 30

QUESTION – 1

RELATIONAL DIAGRAM



ER DIAGARM



SQL QUERIES

```
CREATE DATABASE Sql_Question_1;


CREATE TABLE EmployeeDetails(
    `S.no.` INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    EmpId INT NOT NULL UNIQUE,
    FullName VARCHAR(70),
    ManagerId INT,
    DateOfJoining DATE,
    City VARCHAR (50)
);


CREATE TABLE EmployeeSalary (
    EmpId INT NOT NULL,
    Project VARCHAR(10),
    Salary INT NOT NULL,
    Variable INT,
    FOREIGN KEY (EmpId) REFERENCES EmployeeDetails (EmpId) ON DELETE CASCADE ON UPDATE CASCADE
);


INSERT INTO EmployeeDetails(EmpId,FullName,ManagerId,DateOfJoining,City)
VALUES (111,'Amit Sharma', 231, '2014-01-31', 'Bangalore' ),
      (333,'Rajesh Vyas', 654, '2020-01-30' , 'Mumbai'),
      (555,'Kuldeep Tandon', 543, '2016-11-27' , 'New Delhi' ),
      (777,'Rohan Joshi', 854, '2012-10-15' , 'Chennai' ),
      (888,'Harry', 451, '2017-05-18' , 'Kolkata' );


INSERT INTO EmployeeSalary(EmpId,Project,Salary,Variable)
VALUES (555,'P1',12000,0),
      (111,'P1',10000,500),
      (888,'P2',8000,1000),
      (111,'P2',5000,700);
```

Employee Details

S.no.	EmpId	FullName	ManagerId	DateOfJoining	City
1	111	Amit Sharma	231	2014-01-31	Bangalore
2	333	Rajesh Vyas	654	2020-01-30	Mumbai
3	555	Kuldeep Tandon	543	2016-11-27	New Delhi
4	777	Rohan Joshi	854	2012-10-15	Chennai
5	888	Harry	451	2017-05-18	Kolkata

Employee Salary

EmpId	Project	Salary	Variable
555	P1	12000	0
111	P1	10000	500
888	P2	8000	1000
111	P2	5000	700

Q-1.1 -- SQL Query to fetch records that are present in one table but not in another table ?

SELECT employeedetails.* FROM employeedetails

LEFT JOIN employeesalary

USING (empid) WHERE employeesalary.empid is null;

S.no.	EmpId	FullName	ManagerId	DateOfJoining	City
2	333	Rajesh Vyas	654	2020-01-30	Mumbai
4	777	Rohan Joshi	854	2012-10-15	Chennai

Q-1.2 -- SQL query to fetch all the employees who are not working on any project ?

SELECT empId,FullName FROM employeedetails

WHERE EmpId NOT IN (SELECT EmpId FROM employeesalary);

empId	FullName
333	Rajesh Vyas
777	Rohan Joshi

Q-1.3 -- SQL query to fetch all the Employees from EmployeeDetails who joined in the Year 2020 ?

```
SELECT fullName,DateOfJoining FROM employeedetails

WHERE DateOfJoining BETWEEN '2020-01-01' AND '2020-12-31';


SELECT fullName,DateOfJoining FROM employeedetails

WHERE year(DateOfJoining) = 2020;


SELECT fullName,DateOfJoining FROM employeedetails

WHERE year(DateOfJoining) LIKE '2020%';
```

fullName	DateOfJoining
Rajesh Vyas	2020-01-30

Q-1.4 -- Write an SQL query to fetch records from EmployeeDetails where city ends with character 'i' ?

```
SELECT * FROM employeedetails

WHERE City LIKE '%i';
```

S.no.	EmpId	FullName	ManagerId	DateOfJoining	City
2	333	Rajesh Vyas	654	2020-01-30	Mumbai
3	555	Kuldeep Tandon	543	2016-11-27	New Delhi
4	777	Rohan Joshi	854	2012-10-15	Chennai

Q-1.5 -- Write an SQL query to fetch only odd rows from the table ?

-> <u>If there is a s.no. column:-(For Employee Details Table)</u>

```
SELECT * FROM employeedetails

WHERE `S.no.` % 2 != 0;
```

S.no.	EmpId	FullName	ManagerId	DateOfJoining	City
1	111	Amit Sharma	231	2014-01-31	Bangalore
3	555	Kuldeep Tandon	543	2016-11-27	New Delhi
5	888	Harry	451	2017-05-18	Kolkata

-> If there is no s.no. column:-(For Employee Salary Table)

```
WITH NumberedRows AS (
   SELECT e.*,ROW_NUMBER() OVER (ORDER BY EmpId,project) AS RowNum
   FROM Employeesalary e
)
SELECT EmpId,Salary,Project,variable FROM NumberedRows
WHERE RowNum % 2 != 0;
```

EmpId	Salary	Project	variable
111	10000	P1	500
555	12000	P1	0

Q-1.6 -- Sql Query to find 3rd highest salary from a table without using the TOP / LIMIT keyword ?

```
SELECT Salary
FROM EmployeeSalary Emp1
WHERE 2 = (
            SELECT COUNT( DISTINCT ( Emp2.Salary ) )
            FROM EmployeeSalary Emp2
            WHERE Emp2.Salary > Emp1.Salary
          );
```

Salary
8000

Q-1.7 -- Write an SQL query to fetch all those employees who work on Project other than P1 ?

```
SELECT FullName,Project FROM employeedetails
INNER JOIN employeesalary ON employeesalary.EmpId = employeedetails.EmpId
WHERE employeesalary.Project != 'P1';
```

FullName	Project
Harry	P2
Amit Sharma	P2

Q-1.8 -- Write an SQL query to fetch all the EmpIds which are present in either of the tables – 'EmployeeDetails' and 'EmployeeSalary' ?

```
SELECT EmpId FROM employeesalary

UNION

SELECT EmpId FROM employeedetails

ORDER BY EmpId ASC;
```

EmpId ▲ 1
111
333
555
777
888

Q-1.9 -- Write an SQL query to display both the EmpId and ManagerId together ?

```
SELECT EmpId , ManagerId

FROM employeedetails;
```

EmpId	ManagerId
111	231
333	654
555	543
777	854
888	451

```
SELECT concat(EmpId,"---",ManagerId) AS "EmpId & ManagerId Together"

FROM employeedetails;
```

EmpId --- ManagerId
111---231
333---654
555---543
777---854
888---451

Q-1.10 -- Write an SQL query to fetch project-wise count of employees sorted by project's count in descending order ?

```
SELECT PROJECT,COUNT(project) AS EMPLOYEES
FROM employeesalary
GROUP BY Project
ORDER BY COUNT(employeesalary.project) DESC;
```

PROJECT	EMPLOYEES
P1	2
P2	2

Q-1.11 -- Increase the salary of the Employee by 10% who is working on all the projects of the company ?

```
UPDATE EmployeeSalary
SET Salary = Salary * 1.1
WHERE EmpId IN (
   SELECT EmpId
   FROM EmployeeSalary
   GROUP BY EmpId
   HAVING COUNT(DISTINCT Project) = (SELECT COUNT(DISTINCT Project) FROM EmployeeSalary)
);
```

✔ 2 rows affected. (Query took 0.0039 seconds.)

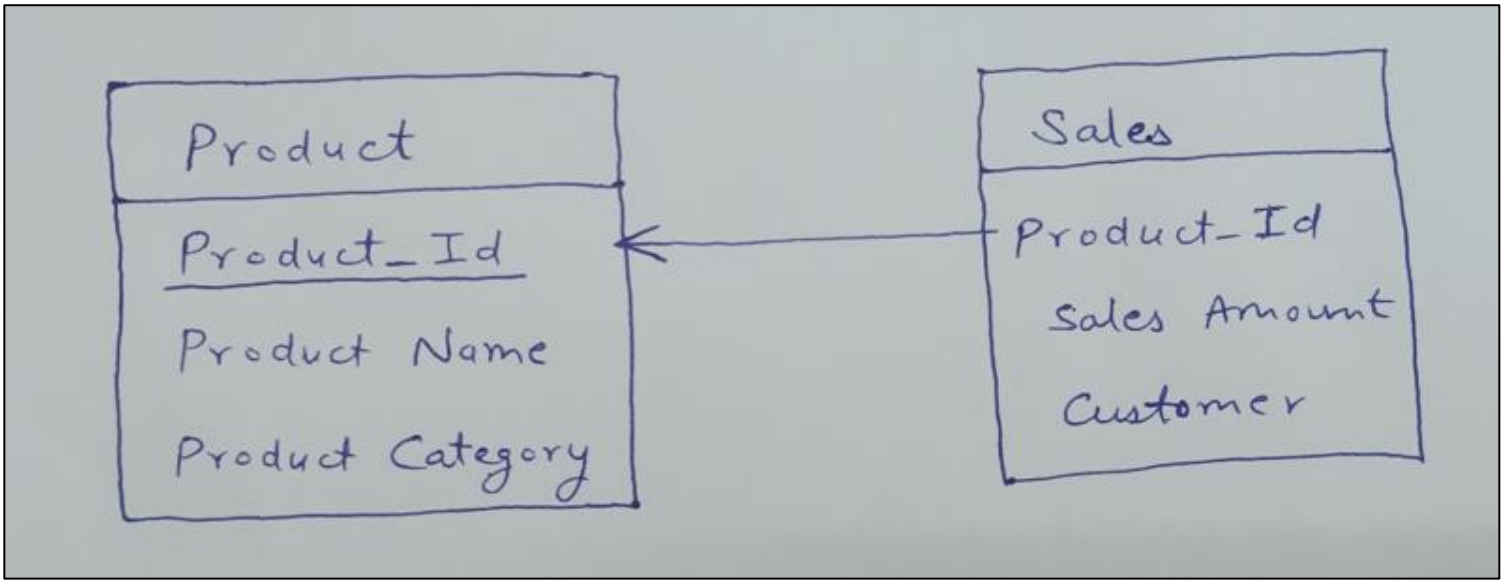
Employee Salary Table
(after Updating Salary)

EmpId	Project	Salary	Variable
555	P1	12000	0
111	P1	10000	500
888	P2	8000	1000
111	P2	5000	700

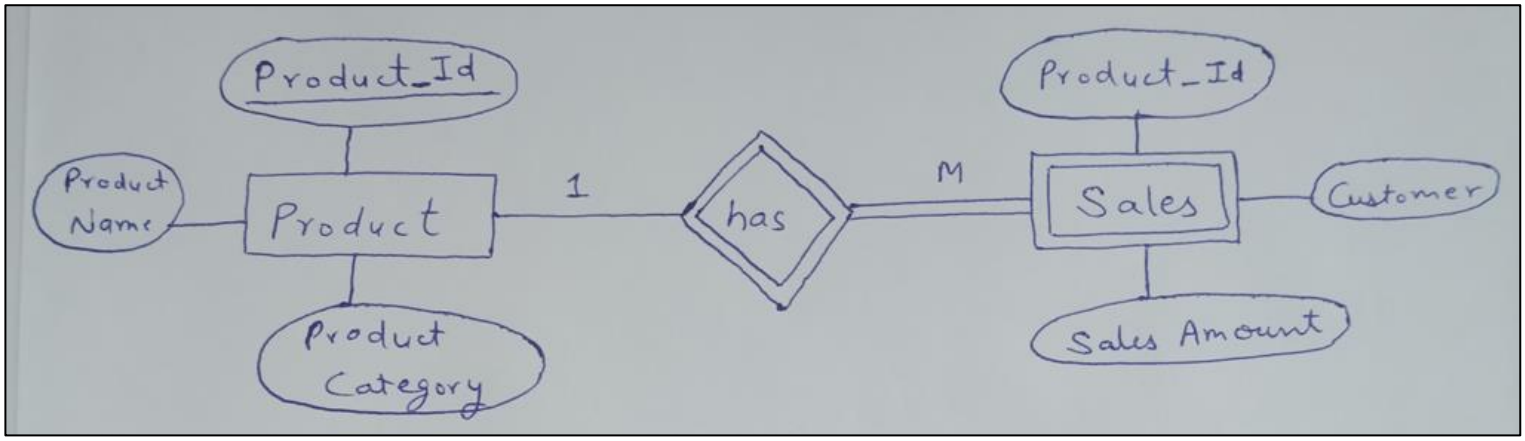
EmpId	Project	Salary	Variable
555	P1	12000	0
111	P1	11000	500
888	P2	8000	1000
111	P2	5500	700

QUESTION – 5

RELATIONAL DIAGRAM



ER DIAGRAM



SQL QUERIES

```
CREATE DATABASE Sql_question_5;



CREATE TABLE Product (
   `Product_Id` INT PRIMARY KEY,
   `Product_Name` VARCHAR (100),
   `Product_category` VARCHAR (50)
);



CREATE TABLE Sales (
   `Product_Id` INT NOT NULL,
   `Customer` VARCHAR (100),
   `Sales_Amount` INT,
   FOREIGN KEY (Product_Id) REFERENCES Product(Product_Id)
);



INSERT INTO Product
VALUES(12,'Bike ABC','Road Bike'),
        (13,'Bike DEF','Mountain Bike'),
        (14,'Bike GHI','Road Bike'),
        (15,'Bike JKL','Touring Bike');



INSERT INTO Sales
VALUES(12,'Joe',1000),
        (13,'Tom',2000),
        (14,'Joe',1500),
        (12,'Bill',1000);
```

Product

Product_Id	Product_Name	Product_category
12	Bike ABC	Road Bike
13	Bike DEF	Mountain Bike
14	Bike GHI	Road Bike
15	Bike JKL	Touring Bike

Sales

Product_Id	Customer	Sales_Amount
12	Joe	1000
13	Tom	2000
14	Joe	1500
12	Bill	1000

Q-5.1 -- Write a SQL statement that returns the distinct list of product categories from the Product table ?

```
SELECT DISTINCT product_category
FROM product;
```

product_category
Road Bike
Mountain Bike
Touring Bike

Q-5.2 -- Write a SQL statement that returns the Sum of Sales Amount grouped by Product Category having sales greater than 1500 ?

```
SELECT product_category,SUM(Sales_amount)
FROM product LEFT JOIN sales Using (Product_id)
GROUP BY Product_category
Having SUM(sales_amount) > 1500;
```

product_category	SUM(Sales_amount)
Mountain Bike	2000
Road Bike	3500

Q-5.3 -- Write a SQL statement that returns the total record count from the Sales table ?

```
SELECT Count(product_id) AS 'Total Records'
FROM sales;
```

Total Records
4

Q-5.4 -- Write a SQL Statement that returns a list of products that do not appear the Sales table ?

```
SELECT product.product_id FROM product
LEFT JOIN Sales USING (product_id)
WHERE sales.product_id IS NULL;
```

product_id
15

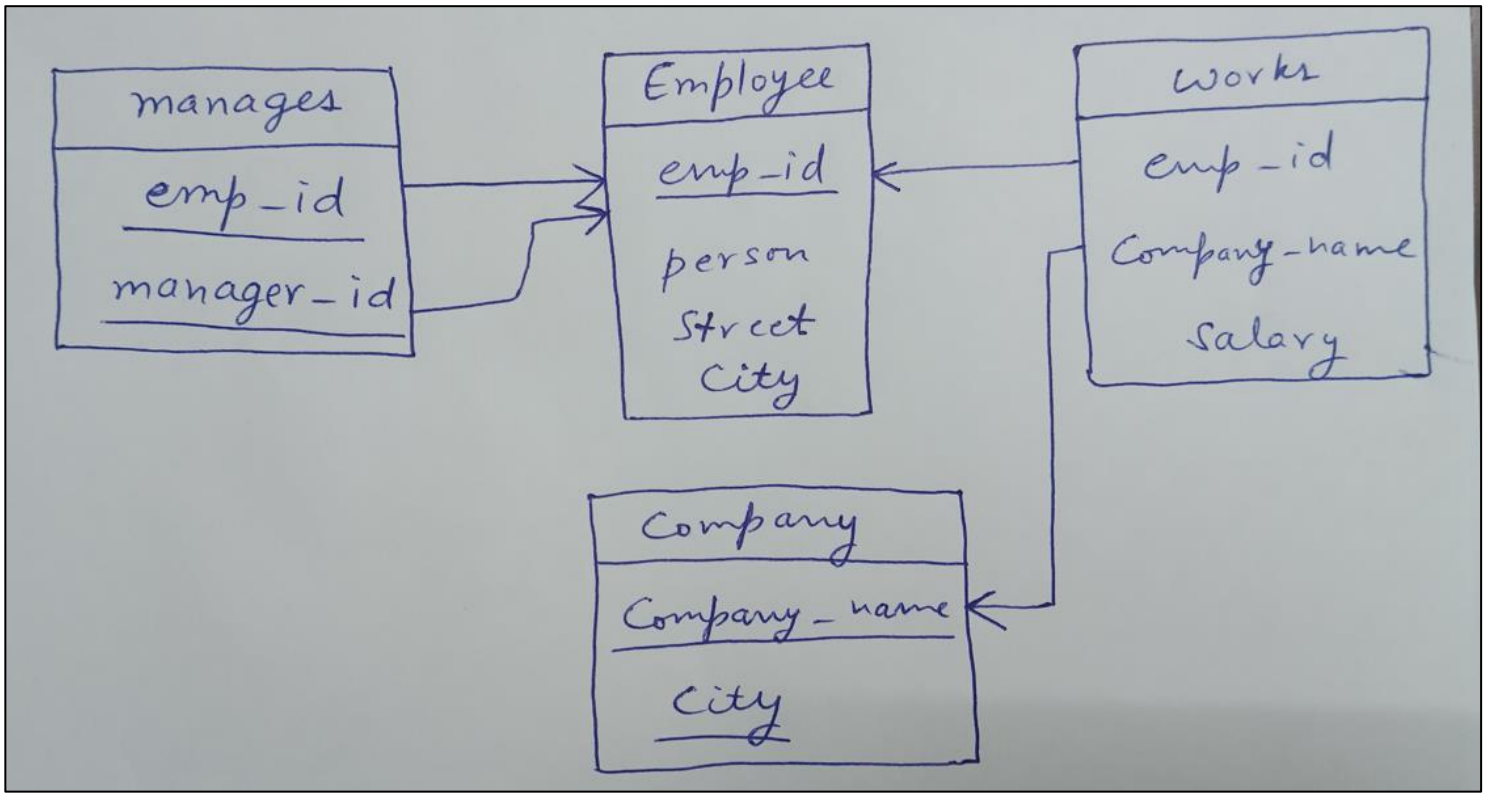
Q.5.5 -- Write a SQL statement that returns all the product categories(including that have 0 sales) having total sales less than 2100 ?

```
SELECT p.product_category, SUM(s.Sales_amount) AS Total_Sales_Amount
FROM product p
LEFT JOIN sales s USING (Product_id)
GROUP BY p.Product_Id
HAVING Total_Sales_Amount < 2100
UNION
SELECT p.product_category, 0 AS Total_Sales_Amount
FROM product p
WHERE p.Product_Id NOT IN (SELECT DISTINCT product_id FROM sales);
```

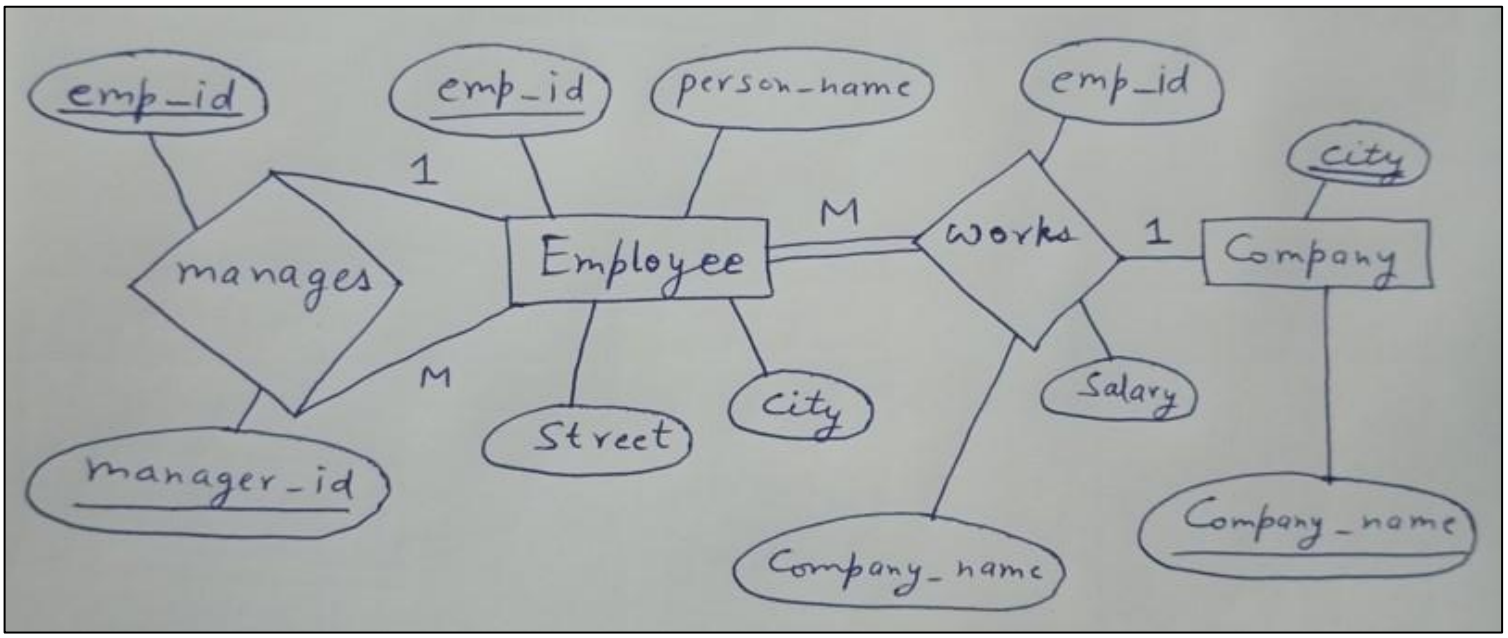
product_category	Total_Sales_Amount
Road Bike	2000
Mountain Bike	2000
Road Bike	1500
Touring Bike	0

QUESTION – 9

RELATIONAL DIAGRAM



ER DIAGRAM



SQL QUERIES

Create database **Employee_database**;

create table **employee**(emp_id int not null PRIMARY KEY,person_name varchar(30) not null,street varchar(30),city varchar(30));

create table **company**(company_name varchar(30),city varchar(30),PRIMARY KEY (company_name,city));

create table **works**(emp_id int not null,company_name varchar(30),salary int not null,FOREIGN KEY (emp_id) REFERENCES employee(emp_id),FOREIGN KEY (company_name) REFERENCES company(company_name));

create table **manages**(emp_id int not null,manager_id int,PRIMARY KEY (emp_id,manager_id), FOREIGN KEY (emp_id) REFERENCES employee(emp_id),FOREIGN KEY (manager_id) REFERENCES employee(emp_id));

INSERT INTO **employee**(emp_id, person_name, street, city)
VALUES ('101','Mohit','st.no.1','Udaipur'),
('102','Rohit','st.no.2','Udaipur'),
('103','Sumit','st.no.3','Kolkata'),
('104','Rahul','st.no.4','Kolkata'),
('105','Hardik','st.no.5','Delhi'),
('106','Ajay','st.no.6','Bihar');

INSERT INTO **company**(company_name, city)
VALUES ('First Bank Company','Udaipur'),
('First Bank Company','Delhi'),
('Big Bank Company','Kolkata'),
('Small Bank Company','Delhi'),
('Small Bank Company','Bihar'),
('Medium Bank Company','Bihar'),
('Medium Bank Company','Delhi');

INSERT INTO **works**(emp_id, company_name, salary)
VALUES (101,'First Bank Company',12000),
(102,'First Bank Company',9000),
(103,'Big Bank Company',10000),
(104,'Big Bank Company',10500),
(105,'Small Bank Company',10000),
(106,'Medium Bank Company',20000);

INSERT INTO **manages**(emp_id, manager_id)
VALUES (101,106),(102,106),(103,106),(104,106),(105,106);

Company

company_name	city
Big Bank Company	Kolkata
First Bank Company	Delhi
First Bank Company	Udaipur
Medium Bank Company	Bihar
Medium Bank Company	Delhi
Small Bank Company	Bihar
Small Bank Company	Delhi

Employee

emp_id	person_name	street	city
101	Mohit	st.no.1	Udaipur
102	Rohit	st.no.2	Udaipur
103	Sumit	st.no.3	Kolkata
104	Rahul	st.no.4	Kolkata
105	Hardik	st.no.5	Delhi
106	Ajay	st.no.6	Bihar

Works

emp_id	company_name	salary
101	First Bank Company	12000
102	First Bank Company	9000
103	Big Bank Company	10000
104	Big Bank Company	10500
105	Small Bank Company	10000
106	Medium Bank Company	20000
103	Medium Bank Company	800
102	Small Bank Company	12000

Manages

emp_id	manager_id
101	106
102	106
103	106
104	106
105	106

Q-9.1 -- Find the ID, name, and city of residence of each employee who works for “First Bank Company” ?

```
SELECT employee.emp_id,Person_name,city
FROM employee JOIN works ON employee.emp_id = works.emp_id
WHERE company_name = 'First bank Company';
```

emp_id	Person_name	city
101	Mohit	Udaipur
102	Rohit	Udaipur

Q-9.2 -- Find the ID, name, and city of residence of each employee who works for “First Bank Company” and earns more than $10000 ?

```
SELECT employee.emp_id,Person_name,city
FROM employee JOIN works ON employee.emp_id = works.emp_id
WHERE company_name = 'First bank company' AND salary > 10000;
```

emp_id	Person_name	city
101	Mohit	Udaipur

Q-9.3 -- Find the ID of each employee who does not work for “First Bank Company” ?

```
SELECT DISTINCT employee.emp_id
FROM employee JOIN works ON employee.emp_id = works.emp_id
WHERE NOT company_name = 'First bank company';
```

emp_id
103
104
105
106
102

Q-9.4 -- Find the ID of each employee who earns more than every employee of “Small Bank Company” ?

```
SELECT employee.emp_id
FROM employee JOIN works ON employee.emp_id = works.emp_id
WHERE Salary > (SELECT MAX(salary) FROM works WHERE company_name = 'Small Bank Company');
```

emp_id
106

Q-9.5 -- Assume that companies may be located in several cities. Find the name of each company that is located in every city in which “Small Bank Company” is located ?

```
SELECT DISTINCT company_name FROM company
WHERE city IN ( SELECT city FROM company WHERE company_name = 'Small Bank Company') AND
company_name != 'Small Bank Company';
```

company_name
First Bank Company
Medium Bank Company

Q-9.6 -- Find the name of the company that has the most employees (or companies, in the case where there is a tie for the most) ?

```
SELECT company_name, COUNT(emp_id) AS num_of_employees
FROM works
GROUP BY company_name
HAVING COUNT(emp_id) = (SELECT MAX(employee_count) FROM (SELECT COUNT(emp_id) AS
employee_count FROM works GROUP BY company_name) AS employee_counts);
```

company_name	num_of_employees
Big Bank Company	2
First Bank Company	2
Medium Bank Company	2
Small Bank Company	2

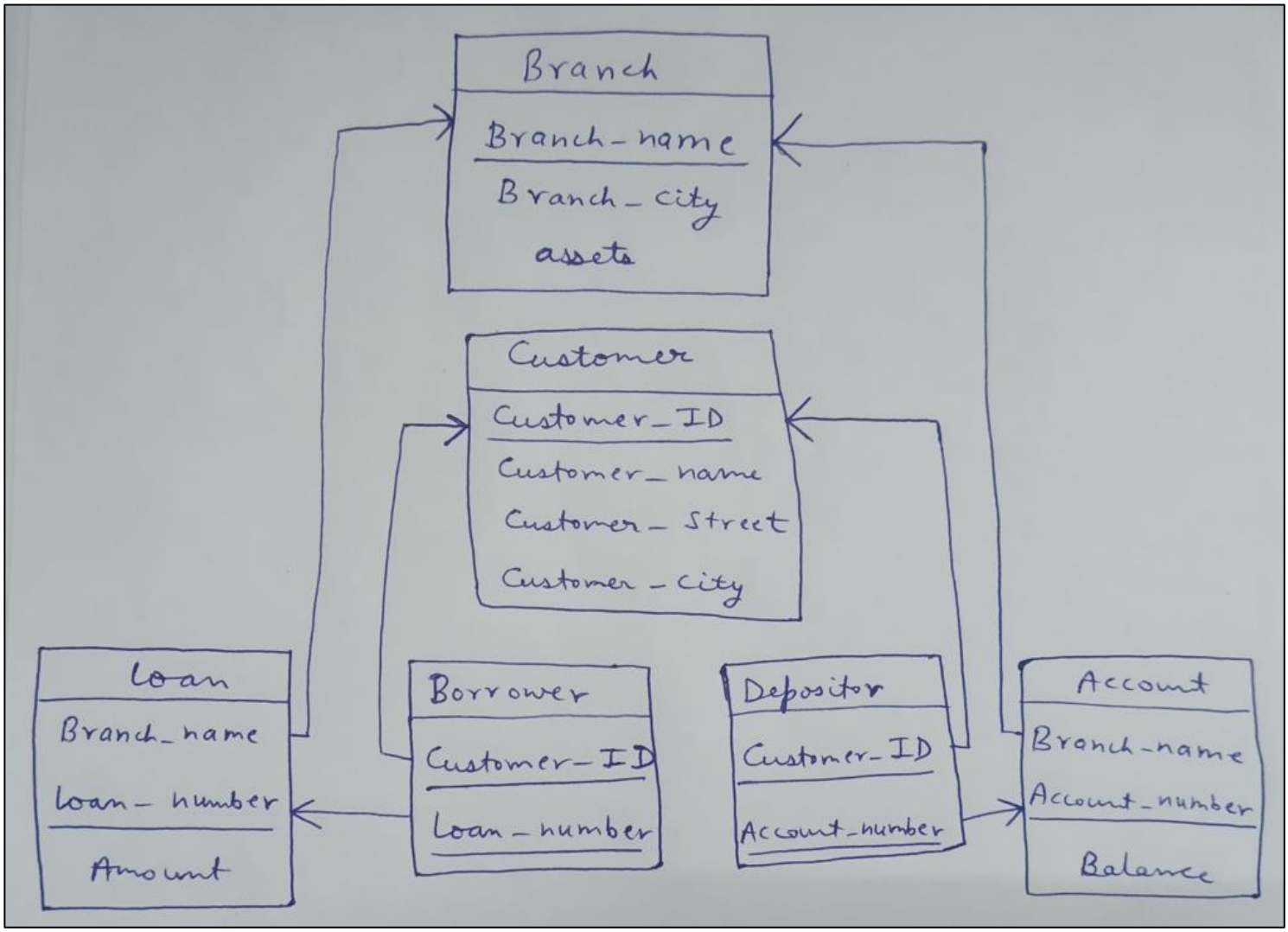
Q-9.7 -- Find the name of each company whose employees earn a higher salary, on average, than the average salary at “First Bank Company” ?

```
SELECT company_name FROM works GROUP BY company_name Having Avg(salary) > (SELECT avg(salary) FROM
works WHERE company_name = 'First Bank Company') AND company_name != 'First Bank Company';
```

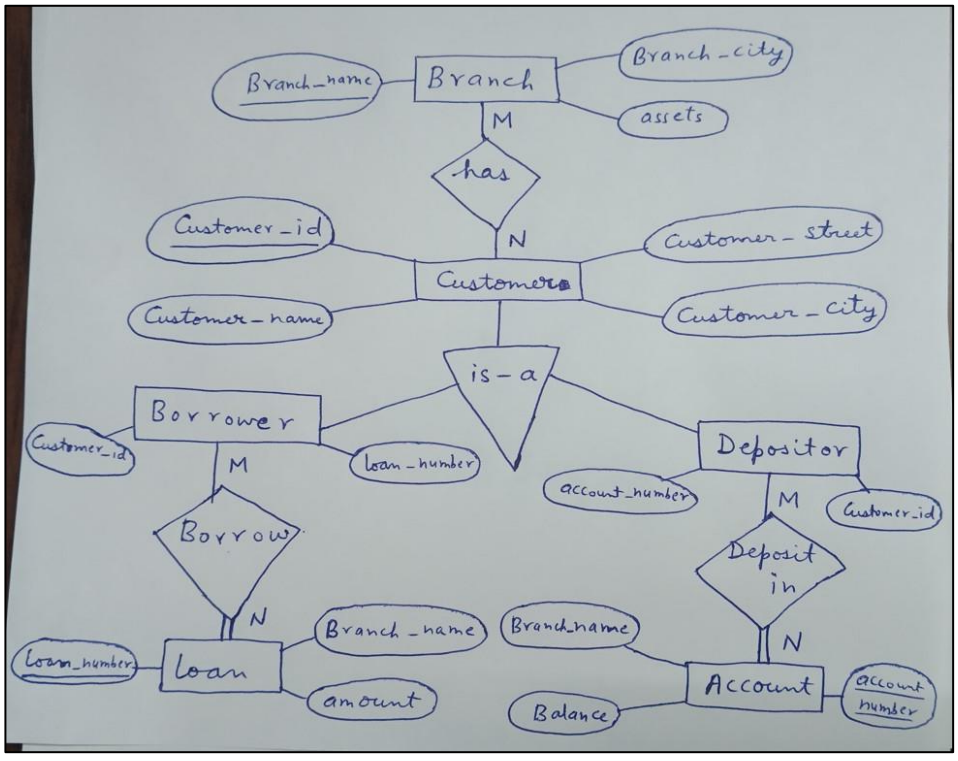
company_name
Small Bank Company

QUESTION – 3

RELATIONAL DIAGRAM



ER DIAGRAM



SQL QUERIES

```
Create database Banking_database;

Create table branch(branch_name varchar(30) PRIMARY KEY,branch_city varchar(30),assets int not null);

create table Customer(Customer_id int not null PRIMARY KEY,Customer_name varchar(30),customer_street
varchar(30),customer_city varchar(30));

create table Loan(loan_number VARCHAR(30) not null PRIMARY KEY ,branch_name varchar(30),amount int not
null,FOREIGN KEY (branch_name) REFERENCES branch(branch_name));

create table borrower(Customer_id int not null , loan_number VARCHAR(30) not null,
PRIMARY KEY (Customer_id, loan_number),FOREIGN KEY (loan_number) REFERENCES loan(loan_number),FOREIGN
KEY (customer_id) REFERENCES customer(customer_id));

create table account(account_number int not null PRIMARY KEY,branch_name varchar(30),balance int not null,
FOREIGN KEY (branch_name) REFERENCES branch(branch_name));

create table depositor(Customer_id int not null ,account_number int not null,
PRIMARY KEY ( Customer_id, account_number),FOREIGN KEY (account_number) REFERENCES
account(account_number),FOREIGN KEY (customer_id) REFERENCES customer(customer_id));



INSERT INTO branch(branch_name, branch_city, assets)
VALUES ('SBI Rajasthan','Udaipur',1000000),
('SBI Punjab','Chandigarh',2000000),
('SBI Haryana','Delhi',3000000),
('SBI Gujarat','Surat',4000000),
('SBI Maharastra','Mumbai',5000000),
('SBI Harrison','Harrison',5000000);

INSERT INTO customer(Customer_id, Customer_name, customer_street, customer_city)
VALUES (12345,'Rahul','st.no.2','Udaipur'),
(54321,'Mohan','st.no.2','Udaipur'),
(67452,'Sohan','st.no.3','Delhi'),
(34259,'Rohit','st.no.4','Mumbai'),
(42629,'Rohan','st.no.5','Chandigarh'),
(36537,'Harish','st.no.6','Harrison');

INSERT INTO loan(loan_number, branch_name, amount)
VALUES ('LN1','SBI Rajasthan', 100000),
('LN2','SBI Rajasthan',200000),
('LN3','SBI Punjab',300000),
('LN4','SBI Maharastra',500000);
```

```
INSERT INTO borrower(Customer_id, loan_number)
VALUES (54321,'LN1'),
(12345,'LN2'),
(42629,'LN3'),
(34259,'LN4');
```

```
INSERT INTO account(account_number, branch_name, balance)
VALUES (1234,'SBI Rajasthan',100000),
(4321,'SBI Punjab',200000),
(2164,'SBI Haryana',300000),
(4272,'SBI Harrison',400000),
(6254,'SBI Maharastra',500000),
(7264,'SBI Rajasthan',600000),
(7265,'SBI Rajasthan',800000);
```

```
INSERT INTO depositor(Customer_id, account_number)
VALUES (12345,1234),
(54321,7264),
(67452,2164),
(42629,4321),
(36537,4272),
(34259,6254),
(34259,7265);
```

Branch

branch_name	branch_city	assets
SBI Gujarat	Surat	4000000
SBI Harrison	Harrison	5000000
SBI Haryana	Delhi	3000000
SBI Maharastra	Mumbai	5000000
SBI Punjab	Chandigarh	2000000
SBI Rajasthan	Udaipur	1000000

Customer

Customer_id	Customer_name	customer_street	customer_city
12345	Rahul	st.no.2	Udaipur
34259	Rohit	st.no.4	Mumbai
36537	Harish	st.no.6	Harrison
42629	Rohan	st.no.5	Chandigarh
54321	Mohan	st.no.2	Udaipur
67452	Sohan	st.no.3	Delhi

Loan

loan_number	branch_name	amount
LN1	SBI Rajasthan	100000
LN2	SBI Rajasthan	200000
LN3	SBI Punjab	300000
LN4	SBI Maharastra	500000

Borrower

Customer_id	loan_number
12345	LN2
34259	LN4
42629	LN3
54321	LN1

Account

account_number	branch_name	balance
1234	SBI Rajasthan	100000
2164	SBI Haryana	300000
4272	SBI Harrison	400000
4321	SBI Punjab	200000
6254	SBI Maharastra	500000
7264	SBI Rajasthan	600000
7265	SBI Rajasthan	800000

Depositor

Customer_id	account_number
12345	1234
34259	6254
34259	7265
36537	4272
42629	4321
54321	7264
67452	2164

Q-3.1 -- Find the ID of each customer of the bank who has an account but not a loan ?

```
SELECT depositor.customer_id FROM depositor
LEFT JOIN borrower ON depositor.Customer_id = borrower.Customer_id
WHERE borrower.Customer_id is NULL;

SELECT customer_id FROM depositor
WHERE customer_id NOT IN ( SELECT Customer_id FROM borrower );
```

customer_id
36537
67452

Q-3.2 -- Find the ID of each customer who lives on the same street and in the same city as customer '12345' ?

```
SELECT customer_id
FROM customer
WHERE customer_street = (SELECT customer_street FROM customer WHERE customer_id = '12345')
  AND customer_city = (SELECT customer_city FROM customer WHERE customer_id = '12345')
  AND customer_id != '12345';
```

customer_id
54321

Q-3.3 -- Find the name of each branch that has at least one customer who has an account in the bank and who lives in “Harrison” ?

```
SELECT account.branch_name FROM
((depositor LEFT JOIN customer ON depositor.Customer_id = customer.Customer_id)
LEFT JOIN account ON depositor.account_number = account.account_number)
WHERE customer_city = 'Harrison';
```

branch_name
SBI Harrison

Q-3.4 -- Find the ID of each customer of the bank who has account and loan both ?

```
SELECT depositor.customer_id FROM depositor
INNER JOIN borrower ON depositor.Customer_id = borrower.Customer_id;

SELECT customer_id FROM depositor
WHERE customer_id IN ( SELECT Customer_id FROM borrower );
```

customer_id
12345
34259
34259
42629
54321

Q-3.5 -- Find the branch that has given most number of loans ?

```
SELECT branch_name AS Branch,count(loan_number) AS 'Loans Passed' FROM loan
GROUP BY branch_name
ORDER BY COUNT(loan_number) DESC
LIMIT 1;
```

Branch	Loans Passed
SBI Rajasthan	2

Q-3.6 -- Find the Customer record who has taken the biggest amount of loan from a branch ?

```
SELECT * FROM
((borrower JOIN loan USING (loan_number))
JOIN customer USING (customer_id))
WHERE amount = (SELECT MAX(amount) FROM loan);
```

Customer_id	loan_number	branch_name	amount	Customer_name	customer_street	customer_city
34259	LN4	SBI Maharastra	500000	Rohit	st.no.4	Mumbai

Q-3.7 -- Find the customer who has account in two branches ?

```
SELECT customer_name,COUNT(customer_id) AS No_of_Accounts FROM ((depositor JOIN account
USING(account_number)) JOIN customer USING(customer_id))
GROUP BY customer_id
HAVING COUNT(customer_id) = 2;
```

customer_name	No_of_Accounts
Rohit	2

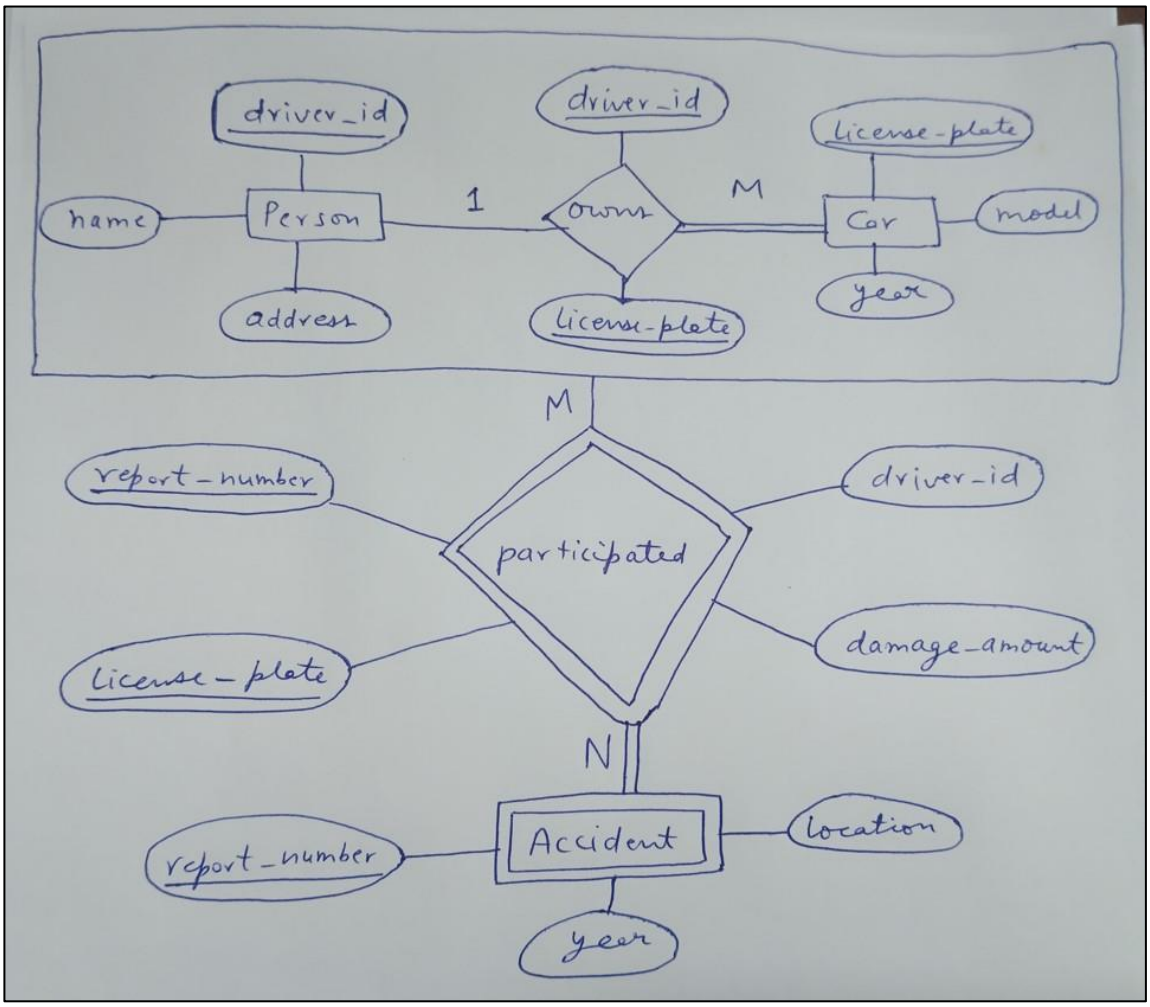
Q-3.8 -- Find the list of all customers along with their number of accounts in different branches ?

```
SELECT customer_name,COUNT(customer_id) AS No_of_Accounts FROM ((depositor JOIN account
USING(account_number)) JOIN customer USING(customer_id))
GROUP BY customer_id
ORDER BY COUNT(customer_id) DESC;
```

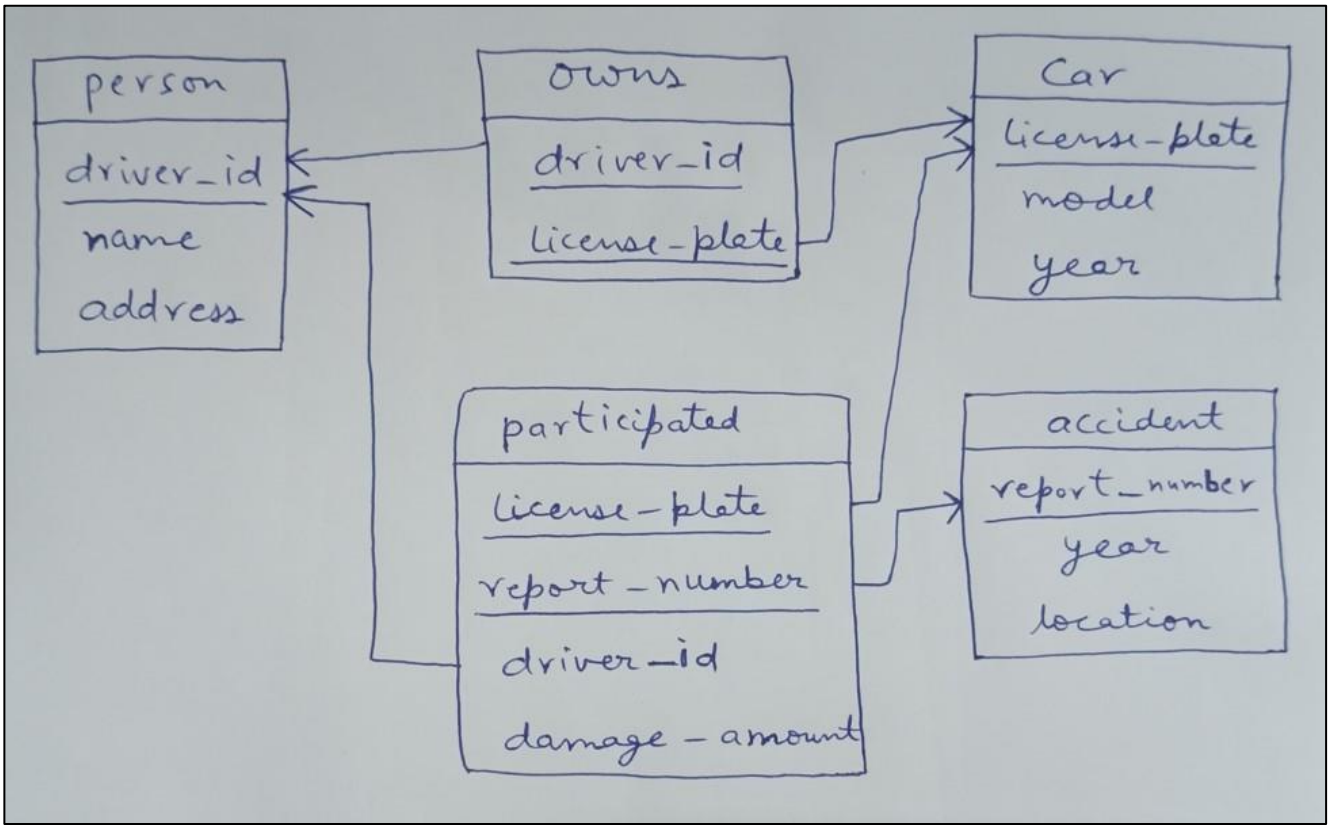
customer_name	No_of_Accounts
Rohit	2
Sohan	1
Rahul	1
Harish	1
Mohan	1
Rohan	1

QUESTION – 8

ER DIAGRAM



RELATIONAL DIAGRAM



SQL QUERIES

```
Create database Insurance_database;

Create table person(driver_id int PRIMARY KEY,name varchar(30),address varchar(30));

create table car(license_plate varchar(50) PRIMARY KEY,model varchar(30),`year` year);

create table accident(report_number int PRIMARY KEY,`year` year,location varchar(30));

create table owns(driver_id int,license_plate varchar(50),PRIMARY KEY(driver_id,license_plate),FOREIGN KEY
(driver_id) REFERENCES person(driver_id),FOREIGN KEY (license_plate) REFERENCES car(license_plate) ON DELETE
CASCADE);

create table participated(report_number int,license_plate varchar(50),driver_id int,damage_amount int,PRIMARY
KEY(report_number,license_plate),FOREIGN KEY (report_number) REFERENCES accident(report_number), FOREIGN
KEY (license_plate) REFERENCES car(license_plate) ON DELETE CASCADE,FOREIGN KEY (driver_id) REFERENCES
person(driver_id));


INSERT INTO `accident`(`report_number`, `year`, `location`)
VALUES (101,2010,'Gujarat'),
        (102,2010,'Pune'),
        (103,2012,'Bangalore'),
        (104,2014,'Mumbai'),
        (105,2017,'Chennai'),
        (106,2017,'Delhi'),
        (107,2017,'Kolkata');

INSERT INTO `car`(`license_plate`, `model`, `year`)
VALUES ('MH23','Omni',2010),
('GJ45','Verna',2014),
('WB35','Wagnar',2017),
('MH57','Creta',2010),
('DL07','Thar',2018),
('RJ27','Audi',2016);

INSERT INTO `person`(`driver_id`, `name`, `address`)
VALUES (12345,'Rahul','Chetak Circle'),
(15345,'Mohit','Shastri Circle'),
(14745,'Rohit','Fatehpura Circle'),
(18345,'Sumit','Delhi Gate'),
(16945,'Hardik','Mali Colony'),
(14521,'Manoj','M.G.Road');

INSERT INTO `owns`(`driver_id`, `license_plate`)
VALUES (12345,'WB35'),
(14745,'MH57'),
(16945,'WB35'),
(12345,'MH23'),
(16945,'GJ45'),
(12345,'DL07'),
(14521,'RJ27');
```

```
INSERT INTO `participated`(`report_number`, `license_plate`, `driver_id`, `damage_amount`)
VALUES (102,'DL07',15345,10000),
(103,'MH57',14745,62000),
(103,'MH23',12345,16000),
(106,'GJ45',16945,25000),
(104,'DL07',12345,15000),
(101,'WB35',16945,20000),
(107,'GJ45',14521,45000);
```

Accident

report_number	year	location
101	2010	Gujarat
102	2010	Pune
103	2012	Bangalore
104	2014	Mumbai
105	2017	Chennai
106	2017	Delhi
107	2019	Kolkata

Car

license_plate	model	year
DL07	Thar	2018
GJ45	Verna	2014
MH23	Omni	2010
MH57	Creta	2010
RJ27	Audi	2016
WB35	Wagnar	2017

Owns

driver_id	license_plate
12345	DL07
12345	MH23
12345	WB35
14521	RJ27
14745	MH57
16945	GJ45
16945	WB35

Participated

report_number	license_plate	driver_id	damage_amount
101	WB35	16945	20000
102	DL07	15345	10000
103	MH23	12345	16000
103	MH57	14745	62000
104	DL07	12345	15000
106	GJ45	16945	25000
107	GJ45	14521	45000

Person

driver_id	name	address
12345	Rahul	Chetak Circle
14521	Manoj	M.G.Road
14745	Rohit	Fatehpura Circle
15345	Mohit	Shastri Circle
16945	Hardik	Mali Colony
18345	Sumit	Delhi Gate

Q-8.1 -- Find the total number of people who owned cars that were involved in accidents in 2017 ?

```
SELECT COUNT(driver_id) AS 'Total Number of People'
FROM ((`participated` LEFT JOIN `owns` USING (`driver_id`)) LEFT JOIN `accident` USING(`report_number`))
WHERE `Year` = 2017;
```

Total Number of People
2

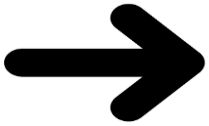
Q-8.2 -- Delete all year-2010 cars belonging to the person whose ID is '12345' ?

```
DELETE car FROM car
JOIN owns ON car.license_plate = owns.license_plate
WHERE `YEAR`= 2010 AND driver_id = 12345;
```

✔ 1 row deleted. (Query took 0.0038 seconds.)

license_plate	model	year
DL07	Thar	2018
GJ45	Verna	2014
MH23	Omni	2010
MH57	Creta	2010
RJ27	Audi	2016
WB35	Wagnar	2017

Car Table After Deletion



Omni Deleted

license_plate	model	year
DL07	Thar	2018
GJ45	Verna	2014
MH57	Creta	2010
RJ27	Audi	2016
WB35	Wagnar	2017

Q-8.3 -- Find the persons who doesn't owned any car and are involved in accidents in 2017 ?

```
SELECT driver_id
FROM `participated`
WHERE driver_id NOT IN (SELECT driver_id FROM `owns`);
```

driver_id
15345

Q-8.4 -- Write a Query to find the person who was involved in an accident with a car that is not owned by him ?

```
SELECT p.driver_id,p.name AS person_name, a.report_number, c.license_plate
FROM person p
JOIN participated pa ON p.driver_id = pa.driver_id
JOIN accident a ON pa.report_number = a.report_number
JOIN car c ON pa.license_plate = c.license_plate
LEFT JOIN owns o ON p.driver_id = o.driver_id AND c.license_plate = o.license_plate
WHERE o.driver_id IS NULL;
```

driver_id	person_name	report_number	license_plate
14521	Manoj	107	GJ45
15345	Mohit	102	DL07

QUESTION – 2 SQL QUERIES

```
CREATE TABLE classroom (
  Building varchar(50) NOT NULL,
  room_number int NOT NULL,
  capacity int NOT NULL,
  PRIMARY KEY(Building,room_number)
);

CREATE TABLE time_slot (
  time_slot_id int NOT NULL,
  day varchar(60) NOT NULL,
  start_time time NOT NULL,
  end_time time NOT NULL,
  PRIMARY KEY(time_slot_id,day,start_time)
);

CREATE TABLE department (
  dept_name varchar(60) NOT NULL,
  building varchar(60) NOT NULL,
  budget int NOT NULL,
  PRIMARY KEY (dept_name),
  FOREIGN KEY (building) REFERENCES classroom(building)
);

CREATE TABLE instructor (
  i_id int NOT NULL,
  Name varchar(60) NOT NULL,
  dept_name varchar(60) NOT NULL,
  salary int NOT NULL,
  PRIMARY KEY (i_id),
  FOREIGN KEY (dept_name) REFERENCES department(dept_name)
);

CREATE TABLE student (
  s_id int NOT NULL,
  Name varchar(60) NOT NULL,
  dept_name varchar(60) NOT NULL,
  tot_cred DECIMAL(4,2) NOT NULL,
  PRIMARY KEY (s_id),
  FOREIGN KEY (dept_name) REFERENCES department(dept_name)
);

CREATE TABLE course (
  course_id int NOT NULL,
  title VARCHAR(60) NOT NULL,
  dept_name varchar(60) NOT NULL,
  credits DECIMAL(4,2) NOT NULL,
  PRIMARY KEY (course_id),
  FOREIGN KEY (dept_name) REFERENCES department(dept_name)
);
```

```
CREATE TABLE advisors (
  s_id int NOT NULL,
  i_id int NOT NULL,
  FOREIGN KEY (s_id) REFERENCES student(s_id),
  FOREIGN KEY (i_id) REFERENCES instructor(i_id)
);


CREATE TABLE prereq (
  course_id int NOT NULL,
  prereq_id int NOT NULL,
  PRIMARY KEY (course_id,prereq_id),
  FOREIGN KEY (course_id) REFERENCES course(course_id)
);


CREATE TABLE section (
  course_id int NOT NULL,
  sec_id int NOT NULL,
  semester varchar(60) NOT NULL,
  `year` year NOT NULL,
  building varchar(60) NOT NULL,
  room_number int NOT NULL,
  time_slot_id int NOT NULL,
  PRIMARY KEY (course_id,sec_id,semester,`year`),
  FOREIGN KEY (building,room_number) REFERENCES classroom(building,room_number),
  FOREIGN KEY (time_slot_id) REFERENCES time_slot(time_slot_id),
  FOREIGN KEY (course_id) REFERENCES course(course_id)
);


CREATE TABLE takes (
  s_id int(60) NOT NULL,
  course_id int(20) NOT NULL,
  sec_id int(52) NOT NULL,
  semester varchar(100) NOT NULL,
  `year` year NOT NULL,
  grade varchar(50),
  PRIMARY KEY(s_id,course_id,sec_id,semester,`year`),
  FOREIGN KEY (course_id,sec_id,semester,`year`) REFERENCES section(course_id,sec_id,semester,`year`)
);


CREATE TABLE teaches (
  i_id int NOT NULL,
  course_id int NOT NULL,
  sec_id int NOT NULL,
  semester varchar(60) NOT NULL,
  `year` year NOT NULL,
  PRIMARY KEY (i_id,course_id,sec_id,semester,`year`),
  FOREIGN KEY (course_id,sec_id,semester,`year`) REFERENCES section(course_id,sec_id,semester,`year`)
);
```

```
CREATE TABLE grade_points (
  grade varchar(50) NOT NULL,
  points decimal(4,2) NOT NULL
);



/*       Inserting Data into the table :--     */


INSERT INTO `classroom`(`Building`, `room_number`, `capacity`)
VALUES ('Block-A',110,70),
('Block-B',105,60),
('Block-B',106,80),
('Block-C',117,100),
('Block-C',118,100),
('Block-D',205,50),
('Block-E',190,60);

INSERT INTO `department`(`dept_name`, `building`, `budget`)
VALUES ('Law','Block-A',100000),
('Computer Science','Block-B',200000),
('Science','Block-C',300000),
('Bio-Technology','Block-D',400000),
('Medical','Block-E',500000);

INSERT INTO `instructor`(`i_id`, `Name`, `dept_name`, `salary`)
VALUES ('24680','Mukesh','Law','100000'),
('29543','Einstein','Computer Science','4500'),
('59436','Sumit','Science','300000'),
('79624','Harsh','Bio-Technology','400000'),
('86347','Rahul','Medical','400000');

INSERT INTO `course`(`course_id`, `title`, `dept_name`, `credits`)
VALUES (111,'BCA','Computer Science',4),
(222,'MCA','Computer Science',4),
(333,'LLB','Law',2),
(444,'B-Tech','Science',3),
(55','M-Tech','Science',3),
(666,'MBBS','Medical',2),
(777,'BT','Bio-Technology',3),
(888,MD,'Medical',4);

INSERT INTO `time_slot`(`time_slot_id`, `day`, `start_time`, `end_time`)
VALUES ('100','Monday','08:00:00','09:00:00'),
('101','Monday','01:00:00','02:00:00'),
('102','Tuesday','10:00:00','11:00:00'),
('103','Wednesday','11:00:00','12:00:00'),
('104','Thursday','12:00:00','02:00:00'),
('105','Friday','01:00:00','02:00:00'),
('106','Saturday','10:00:00','12:00:00');
```

```
INSERT INTO `section`(`course_id`, `sec_id`, `semester`, `year`, `building`, `room_number`, `time_slot_id`)
VALUES ('111',50,'BCA-I','2017','Block-B','105','100'),
('222',60,'MCA-I','2017','Block-B','106','101'),
('333',70,'LLB-I','2017','Block-A','110','102'),
('444',80,'BT-I','2018','Block-C','118','103'),
('555',90,'MT-I','2019','Block-C','117','104'),
('666',100,'D-I','2020','Block-E','190','105'),
('777',110,'BioTech-I','2020','Block-D','205','106');


INSERT INTO `teaches`(`i_id`, `course_id`, `sec_id`, `semester`, `year`)
VALUES ('24680','333','70','LLB-I','2017'),
('29543','111','50','BCA-I','2017'),
('59436','444','80','BT-I','2018'),
('79624','777','110','BioTech-I','2020'),
('86347','555','90','MT-I','2019'),
('29543','222','60','MCA-I','2017');


INSERT INTO `takes`(`s_id`, `course_id`, `sec_id`, `semester`, `year`, `grade`)
VALUES ('34592','333','70','LLB-I','2017','A-'),
('12345','111','50','BCA-I','2017','A'),
('63495','444','80','BT-I','2018','B'),
('42697','777','110','BioTech-I','2020','A-'),
('74368','666','100','D-I','2020','B+'),
('54321','222','60','MCA-I','2017','B');

INSERT INTO `student`(`s_id`, `Name`, `dept_name`, `tot_cred`)
VALUES ('12345','Rohan','Computer Science','16'),
('54321','Mohan','Computer Science','12'),
('34592','Sohan','Law','7.4'),
('63495','Rohit','Science','9'),
('42697','Suresh','Bio-Technology','11.1'),
('74368','Ankit','Medical','6.6');

INSERT INTO `prereq`(`course_id`, `prereq_id`)
VALUES ('555','444'),
('222','111');

INSERT INTO `advisors`(`s_id`, `i_id`)
VALUES ('12345','29543'),
('54321','29543'),
('34592','24680'),
('63495','59436');


INSERT INTO `grade_points`(`grade`, `points`)
VALUES ('A+','4'),('A-','3.7'),('B+','3.3'),('B','3');
```

Advisors

s_id	i_id
12345	29543
54321	29543
34592	24680
63495	59436

Classroom

Building	room_number	capacity
Block-A	110	70
Block-B	105	60
Block-B	106	80
Block-C	117	100
Block-C	118	100
Block-D	205	50
Block-E	190	60

Course

course_id	title	dept_name	credits
111	BCA	Computer Science	4.00
222	MCA	Computer Science	4.00
333	LLB	Law	2.00
444	B-Tech	Science	3.00
555	M-Tech	Science	3.00
666	MBBS	Medical	2.00
777	BT	Bio-Technology	3.00
888	MD	Medical	4.00

Department

dept_name	building	budget
Bio-Technology	Block-D	400000
Computer Science	Block-B	200000
Law	Block-A	100000
Medical	Block-E	500000
Science	Block-C	300000

Grade points

grade	points
A+	4.00
A-	3.70
B+	3.30
B	3.00

Instructor

i_id	Name	dept_name	salary
24680	Mukesh	Law	100000
29543	Einstein	Computer Science	4500
59436	Sumit	Science	300000
79624	Harsh	Bio-Technology	400000
86347	Rahul	Medical	400000

Pre req

course_id	prereq_id
222	111
555	444

Section

course_id	sec_id	semester	year	building	room_number	time_slot_id
111	50	BCA-I	2017	Block-B	105	100
222	60	MCA-I	2017	Block-B	106	101
333	70	LLB-I	2017	Block-A	110	102
444	80	BT-I	2018	Block-C	118	103
555	90	MT-I	2019	Block-C	117	104
666	100	D-I	2020	Block-E	190	105
777	110	BioTech-I	2020	Block-D	205	106

Student

s_id	Name	dept_name	tot_cred
12345	Rohan	Computer Science	16.00
34592	Sohan	Law	7.40
42697	Suresh	Bio-Technology	11.10
54321	Mohan	Computer Science	12.00
63495	Rohit	Science	9.00
74368	Ankit	Medical	6.60

Takes

s_id	course_id	sec_id	semester	year	grade
12345	111	50	BCA-I	2017	A+
34592	333	70	LLB-I	2017	A-
42697	777	110	BioTech-I	2020	A-
54321	222	60	MCA-I	2017	B
63495	444	80	BT-I	2018	B
74368	666	100	D-I	2020	B+

Teaches

i_id	course_id	sec_id	semester	year
24680	333	70	LLB-I	2017
29543	111	50	BCA-I	2017
29543	222	60	MCA-I	2017
59436	444	80	BT-I	2018
79624	777	110	BioTech-I	2020
86347	555	90	MT-I	2019

Time Slot

time_slot_id	day	start_time	end_time
100	Monday	08:00:00	09:00:00
101	Monday	01:00:00	02:00:00
102	Tuesday	10:00:00	11:00:00
103	Wednesday	11:00:00	12:00:00
104	Thursday	12:00:00	02:00:00
105	Friday	01:00:00	02:00:00
106	Saturday	10:00:00	12:00:00

Q-2.1 -- Find the total grade points earned by the student with ID '12345', across all courses taken by the student ?

```
SELECT s_id,tot_cred FROM `student`
WHERE s_id = '12345';
```

s_id	tot_cred
12345	16.00

Q-2.2 -- Find the grade point average (GPA) for the above student, that is, the total grade points divided by the total credits for the associated courses ?

```
SELECT s.s_id,
    s.Name,
    COALESCE(SUM(g.points * c.credits) / NULLIF(SUM(c.credits), 0), 0) AS GPA
FROM student s
LEFT JOIN takes t ON s.s_id = t.s_id
LEFT JOIN grade_points g ON t.grade = g.grade
LEFT JOIN course c ON t.course_id = c.course_id
GROUP BY s.s_id, s.Name
HAVING s.s_id = 12345;
```

s_id	Name	GPA
12345	Rohan	4.00000000

Q-2.3 -- Find the ID and the grade-point average of each student ?

```
SELECT s_id, AVG(points) AS grade_point_average
FROM takes
JOIN grade_points ON takes.grade = grade_points.grade
GROUP BY s_id;
```

s_id	grade_point_average
12345	4.000000
34592	3.700000
42697	3.700000
54321	3.000000
63495	3.000000
74368	3.300000

Q-2.4 -- Find the sections that had the maximum enrollment in year 2017 ?

```
SELECT course_id, sec_id, semester, year, COUNT(s_id) AS enrollment
FROM takes
WHERE year = 2017
GROUP BY course_id, sec_id, semester, year
HAVING COUNT(s_id) = (SELECT MAX(enrollment_count)
            FROM (SELECT course_id, sec_id, semester, year, COUNT(s_id) AS enrollment_count
                FROM takes
                WHERE year = 2017
                GROUP BY course_id, sec_id, semester, year) AS max_enrollment);
```

course_id	sec_id	semester	year	enrollment
111	50	BCA-I	2017	1
222	60	MCA-I	2017	1
333	70	LLB-I	2017	1

Q-2.5 -- Find the maximum enrollment, across all sections, in year 2017 ?

```
SELECT MAX(enrollment_count) AS max_enrollment
FROM (
   SELECT COUNT(s_id) AS enrollment_count
   FROM takes
   WHERE year = 2017
   GROUP BY sec_id
) AS section_enrollments;
```

max_enrollment
1

Q-2.6 -- Find the highest salary of any instructor ?

```
SELECT MAX(salary) AS "Highest Salary" FROM instructor;
```

Highest Salary
400000

Q-2.7 -- Find all instructors earning the highest salary (there may be more than one with the same salary) ?

```
SELECT name,salary FROM instructor WHERE salary = (SELECT MAX(salary) FROM instructor);
```

name	salary
Harsh	400000
Rahul	400000

Q-2.8 -- Find the IDs of all students who were taught by an instructor named Einstein; make sure there are no duplicates in the result ?

```
SELECT s_id,student.Name AS Student_Name,instructor.Name AS Instructor_Name FROM student LEFT JOIN
instructor ON student.dept_name = instructor.dept_name
WHERE instructor.name = 'Einstein';
```

s_id	Student_Name	Instructor_Name
12345	Rohan	Einstein
54321	Mohan	Einstein

Q-2.9 -- Find the enrollment of each section that was offered in year 2017 ?

```
SELECT sec_id,`year`, COUNT(s_id) AS enrollment
FROM takes
WHERE year = 2017
GROUP BY sec_id;
```

sec_id	year	enrollment
50	2017	1
60	2017	1
70	2017	1

Q-2.10 -- Find the titles of courses in the Comp. Sci. department that have 4 credits ?

```
SELECT title,credits FROM course
WHERE dept_name = 'Computer Science' AND credits = 4;
```

title	credits
BCA	4.00
MCA	4.00

Q-2.11 -- Insert every student whose tot_cred attribute is greater than 10 as an instructor in the same department, with a salary of Rs.10,000 ?

```
INSERT INTO `instructor`(`i_id`, `Name`, `dept_name`, `salary`)
SELECT `s_id`, `Name`, `dept_name`,'10000'
FROM `student`
WHERE tot_cred > 10;
```

✔ 3 rows inserted. (Query took 0.0029 seconds.)

Instructor table after new data insertion

i_id	Name	dept_name	salary
24680	Mukesh	Law	100000
29543	Einstein	Computer Science	4500
59436	Sumit	Science	300000
79624	Harsh	Bio-Technology	400000
86347	Rahul	Medical	400000

i_id	Name	dept_name	salary
12345	Rohan	Computer Science	10000
24680	Mukesh	Law	100000
29543	Einstein	Computer Science	4500
42697	Suresh	Bio-Technology	10000
54321	Mohan	Computer Science	10000
59436	Sumit	Science	300000
79624	Harsh	Bio-Technology	400000
86347	Rahul	Medical	400000

Q-2.12 -- Increase the salary of each instructor in the Comp. Sci. department by 10% ?

```
UPDATE `instructor` SET `salary`= Salary * 1.1
WHERE dept_name = 'Computer Science';
```

✔ 1 row affected. (Query took 0.0048 seconds.)

Instructor table after Salary increment

i_id	Name	dept_name	salary
24680	Mukesh	Law	100000
29543	Einstein	Computer Science	4500
59436	Sumit	Science	300000
79624	Harsh	Bio-Technology	400000
86347	Rahul	Medical	400000

i_id	Name	dept_name	salary
24680	Mukesh	Law	100000
29543	Einstein	Computer Science	4950
59436	Sumit	Science	300000
79624	Harsh	Bio-Technology	400000
86347	Rahul	Medical	400000

Q-2.13 -- Delete all courses that have never been offered (i.e., do not occur in the section relation) ?

```
Delete FROM course
WHERE course.course_id IN (SELECT course.course_id FROM course LEFT JOIN section ON course.course_id =
section.course_id WHERE section.course_id is NULL);
```

✔ 1 row deleted. (Query took 0.0032 seconds.)

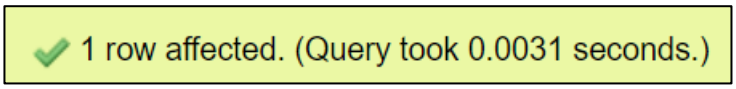
Course table after deletion

course_id	title	dept_name	credits
111	BCA	Computer Science	4.00
222	MCA	Computer Science	4.00
333	LLB	Law	2.00
444	B-Tech	Science	3.00
555	M-Tech	Science	3.00
666	MBBS	Medical	2.00
777	BT	Bio-Technology	3.00
888	MD	Medical	4.00

course_id	title	dept_name	credits
111	BCA	Computer Science	4.00
222	MCA	Computer Science	4.00
333	LLB	Law	2.00
444	B-Tech	Science	3.00
555	M-Tech	Science	3.00
666	MBBS	Medical	2.00
777	BT	Bio-Technology	3.00

Q-2.14 -- Increase the salary of instructor in the Comp. Sci. department where salary is less then 5000 rs. by 10% ?

```
UPDATE `instructor` SET `salary`= (salary+(salary*0.1))
WHERE dept_name = 'Computer Science' AND salary < 5000;
```



Instructor table after Salary updation

i_id	Name	dept_name	salary
24680	Mukesh	Law	100000
29543	Einstein	Computer Science	4500
59436	Sumit	Science	300000
79624	Harsh	Bio-Technology	400000
86347	Rahul	Medical	400000

i_id	Name	dept_name	salary
24680	Mukesh	Law	100000
29543	Einstein	Computer Science	4950
59436	Sumit	Science	300000
79624	Harsh	Bio-Technology	400000
86347	Rahul	Medical	400000