SQL Queries

Q-1) create table students and exam

Query:

1. *Creating 'student' table:*

```
create table students (Rollno int primary key, name varchar(100), branch varchar(100));
```

2. *Creating 'exam' table:*

```
create table exam (Rollno int, S_code varchar(10), Marks int, P_code varchar(10), foreign key (Rollno) REFERENCES student (Rollno));
```

3. *Inserting data into 'student' table:*

```
INSERT INTO `students`(`Rollno`, `name`, `branch`) VALUES
('1', 'Jay', 'Computer Science'),
('2', 'Suhani', 'Electronic and com'),
('3', 'Kriti', 'Electronic and com');
```

4. *Inserting data into 'exam' table:*

```
INSERT INTO `exam`(`Rollno`, `S_code`, `Marks`, `P_code`)
VALUES
('1','CS111','50','CS'),
('1','CS112','60','CS'),
('2','EC101','66','EC'),
('2','EC102','70','EC'),
('3','EC101','45','EC'),
('3','EC102','50','EC');
```

Rollno	name	branch
1	Jay	Computer Science
2	Suhani	Electronic and com
3	Kriti	Electronic and com

Rollno	S_code	Marks	P_code
1	CS111	50	CS
1	CS112	60	CS
2	EC101	66	EC
2	EC102	70	EC
3	EC101	45	EC
3	EC102	50	EC

Q-2) Create table employee and incentive:

Queries:-

1. *Creating 'employee' table:*

```
create table Employee (Employee_id int primary key, first_name varchar(100), last_name varchar(100), salary decimal (10,2), joining_date datetime, department varchar(100));
```

2. *Creating 'incentive' table:*

```
create table Incentive (Employee_ref_id int, incentive_date date, incentive_amount decimal(10,2), foreign key (Employee_ref_id) references employee (Employee_id));
```

3. *Inserting data into 'employee' table:*

```
INSERT INTO `employee`(`Employee_id`, `first_name`, `last_name`, `salary`, `joining_date`, `department`) VALUES
('1', 'John ', 'Abraham', '1000000', '2013-01-01 12:00:00', 'Banking'),
('2', 'Michael', 'Clarke', '800000', '2013-01-01 12:00:00', 'Insurance'),
('3', 'Roy', 'Thomas', '700000', '2013-01-01 12:00:00', 'Banking'),
('4', 'Tom', 'Jose', '800000', '2013-02-01 12:00:00', 'Insurance'),
('5', 'Jerry', 'Pinto', '650000', '2013-02-01 12:00:00', 'Insurance'),
('6', 'Philip', 'Mathew', '750000', '2013-01-01 12:00:00', 'Services'),
('7', 'TestName1', '123', '650000', '2013-01-01 12:00:00', 'Services'),
('8', 'TestName2', 'Lname', '600000', '2013-02-01 12:00:00', 'Insurance');
```

Employee_id	first_name	last_name	salary	joining_date	department
1	John	Abraham	1000000.0(	2013-01-01 12:00:00	Banking
2	Michael	Clarke	800000.00	2013-01-01 12:00:00	Insurance
3	Roy	Thomas	700000.00	2013-01-01 12:00:00	Banking
4	Tom	Jose	800000.00	2013-02-01 12:00:00	Insurance
5	Jerry	Pinto	650000.00	2013-02-01 00:00:00	Insurance
6	Philip	Mathew	750000.00	2013-01-01 12:00:00	Services
7	TestName1	123	650000.00	2013-01-01 12:00:00	Services
8	TestName2	Lname	600000.00	2013-02-01 12:00:00	Insurance

4. *Inserting data into 'incentive' table:*

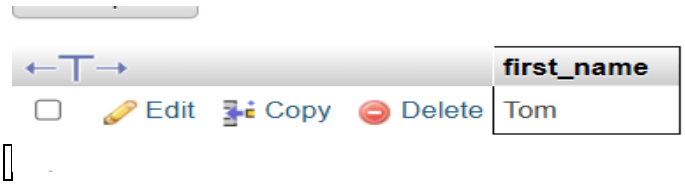
```
INSERT INTO `incentive`(`Employee_ref_id`, `incentive_date`, `incentive_amount`)
VALUES
('1','2013-02-01','5000'),
('2','2013-02-01','3000'),
('3','2013-02-01','4000'),
('1','2013-01-01','4500'),
('2','2013-01-01','3500');
```

Employee_ref_id	incentive_date	incentive_amount
1	2013-02-01	5000.00
2	2013-02-01	3000.00
3	2013-02-01	4000.00
1	2013-01-01	4500.00
2	2013-01-01	3500.00

5. *Get First_Name from 'employee' table using '**Tom**' name "Employee Name":*

Query:-

```
SELECT first_name FROM `employee` WHERE first_name = 'Tom';
```



6. *Get FIRST_NAME, Joining Date, and Salary from 'employee' table:*

Query:-

```
SELECT first_name, joining_date, salary FROM `employee`;
```

first_name	joining_date	salary
John	2013-01-01 12:00:00	1000000.00
Michael	2013-01-01 12:00:00	800000.00
Roy	2013-01-01 12:00:00	700000.00
Tom	2013-02-01 12:00:00	800000.00
Jerry	2013-02-01 00:00:00	650000.00
Philip	2013-01-01 12:00:00	750000.00
TestName1	2013-01-01 12:00:00	650000.00
TestName2	2013-02-01 12:00:00	600000.00

7. *Get all employee details from 'employee' table ordered by first_name ascending and salary descending:*

Query:-

```
SELECT * FROM `employee` order by first_name asc, salary desc;
```

Employee_id	first_name ▲ 1	last_name	salary ▼ 2	joining_date	department
5	Jerry	Pinto	650000.00	2013-02-01 00:00:00	Insurance
1	John	Abraham	1000000.00	2013-01-01 12:00:00	Banking
2	Michael	Clarke	800000.00	2013-01-01 12:00:00	Insurance
6	Philip	Mathew	750000.00	2013-01-01 12:00:00	Services
3	Roy	Thomas	700000.00	2013-01-01 12:00:00	Banking
7	TestName1	123	650000.00	2013-01-01 12:00:00	Services
8	TestName2	Lname	600000.00	2013-02-01 12:00:00	Insurance
4	Tom	Jose	800000.00	2013-02-01 12:00:00	Insurance

8. *Get employee details from 'employee' table whose first name contains **'J'**.*

Query:-

```
SELECT * FROM `employee` WHERE first_name like '% J %';
```

Employee_id	first_name	last_name	salary	joining_date	department
1	John	Abraham	1000000.00	2013-01-01 12:00:00	Banking
5	Jerry	Pinto	650000.00	2013-02-01 00:00:00	Insurance

9. *Get department wise maximum salary from the 'employee' table and order by salary ascending:*

Query:-

```
SELECT `department`, MAX(salary) as Max_Salary FROM `employee`
group by department
order by Max_Salary asc;
```

department	Max_Salary ▲ 1
Services	750000.00
Insurance	800000.00
Banking	1000000.00

10. *Select first_name, incentive amount from 'employee' and 'incentives' table for those employees who have incentives and incentive amount greater than 3000:*

Query:-

```
SELECT e.first_name, i.incentive_amount from employee e join incentive i on
e.Employee_id = i.Employee_ref_id where i.incentive_amount > 3000;
```

first_name	incentive_amount
John	5000.00
Roy	4000.00
John	4500.00
Michael	3500.00

11. Create After Insert trigger on 'Employee' table which insert records in view table:

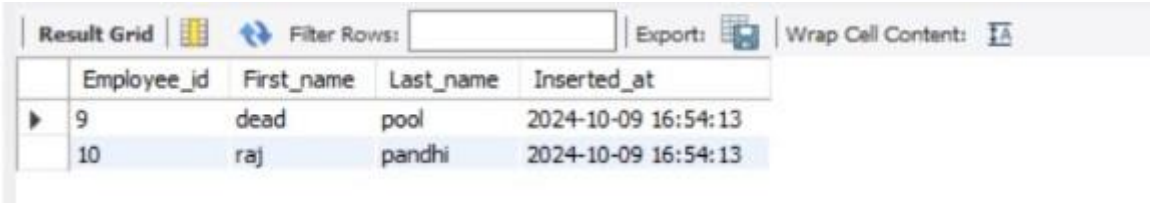
Query:-

```
CREATE TABLE ViewTable (Employee_id INT, First_name VARCHAR(100), Last_name VARCHAR(100), Inserted_at DATETIME);
```

```
DELIMITER $$

CREATE TRIGGER after_employee_insert AFTER INSERT ON Employee

FOR EACH ROW

BEGIN

        INSERT INTO ViewTable (Employee_id, First_name, Last_name, Inserted_at)

        VALUES (NEW.Employee_id, NEW.First_name, NEW.Last_name, NOW());

END $$
```

```
INSERT INTO Employee (Employee_id, First_name, Last_name, Salary, Joining_date, Department) VALUES

(9, 'dead', 'pool', 50000.00, '2001-02-01', 'HR'),

(10, 'raj', 'pandhi', 60000.00, '2002-02-01', 'sales');
```

```
select * from ViewTable;
```



Employee_id	First_name	Last_name	Inserted_at
9	dead	pool	2024-10-09 16:54:13
10	raj	pandhi	2024-10-09 16:54:13

Q-3) Create table below : salesperson and customer:

Query:-

1. *Creating table salesperson:*

```
create table salesperson (s_no int primary key, s_name varchar(50), city varchar(50),
commision decimal(10,2));
```

2. *Inserting data into table salesperson*:

```
INSERT INTO `salesperson`(`s_no`, `s_name`, `city`, `commision`)
VALUES
('1001', 'Peel', 'London', '0.12'),
('1002', 'Serres', 'San Jose', '0.13'),
('1004', 'Motika', 'London', '0.11'),
('1007', 'Rafkin', 'Barcelona', '0.15'),
('1003', 'Axelrod', 'New York', '0.10');
```

s_no	s_name	city	commision
1001	Peel	London	0.12
1002	Serres	San Jose	0.13
1003	Axelrod	New York	0.10
1004	Motika	London	0.11
1007	Rafkin	Barcelona	0.15

3. *Creating table customer*:

```
create table customer (c_nm int primary key, c_name varchar(50), city varchar(50),
rating int, s_no int, foreign key(s_no) REFERENCES salesperson(s_no));
```

4. *Inserting data into table customer*:

```
INSERT INTO `customer`(`c_nm`, `c_name`, `city`, `rating`, `s_no`)
VALUES
('201', 'Hoffman', 'London', '100', '1001'),
('202', 'Giovanne', 'Rome', '200', '1003'),
('203', 'Liu', 'San Jose', '300', '1002'),
('204', 'Grass', 'Barcelona', '100', '1002'),
('206', 'Clemens', 'London', '300', '1007'),
('207', 'Pereira', 'Rome', '100', '1004');
```

c_nm	c_name	city	rating	s_no
201	Hoffman	London	100	1001
202	Giovanne	Rome	200	1003
203	Liu	San Jose	300	1002
204	Grass	Barcelona	100	1002
206	Clemens	London	300	1007
207	Pereira	Rome	100	1004

5. *Names and cities of all salespeople in London with commission above 0.12*:

Query:-

```
Select s_name, city from salesperson where city = 'London' and commission > 0.12;
```

--no output because there is no data where city is London and commission is above 0.12

6. *All salespeople either in Barcelona or in London*:

Query:-

```
SELECT * FROM `salesperson` WHERE city = 'Barcelona' or city = 'London';
```

s_no	s_name	city	commision
1001	Peel	London	0.12
1004	Motika	London	0.11
1007	Rafkin	Barcelona	0.15

7. *All salespeople with commission between 0.10 and 0.12*:

Query:-

```
SELECT * FROM `salesperson` WHERE commission  > 0.10 and commision < 0.12;
```

s_no	s_name	city	commision
1004	Motika	London	0.11

8. *All customers whose rating <= 100 unless they are located in Rome*:

Query:-

```
SELECT * FROM `customer` WHERE rating <= 100 or city = 'Rome';
```

c_nm	c_name	city	rating	s_no
201	Hoffman	London	100	1001
202	Giovanne	Rome	200	1003
204	Grass	Barcelona	100	1002
207	Pereira	Rome	100	1004

Q-4) Write a SQL statement that displays all the information about all salespeople:

Query:-

1. *Creating the table salesman*:

```
create table salesman (salesman_id int, name varchar(50), city varchar(50),
commission decimal(4,2));
```

2. *Inserting values in table salesman*:

```
INSERT INTO salesman (salesman_id, name, city, commission)
VALUES
(5001, 'James Hoog', 'New York', 0.15),
(5002, 'Nail Knite', 'Paris', 0.13),
(5005, 'Pit Alex', 'London', 0.11),
(5006, 'Mc Lyon', 'Paris', 0.14),
(5007, 'Paul Adam', 'Rome', 0.13),
(5003, 'Lauson Hen', 'San Jose', 0.12);
```

salesman_id	name	city	commission
5001	James Hoog	New York	0.15
5002	Nail Knite	Paris	0.13
5005	Pit Alex	London	0.11
5006	Mc Lyon	Paris	0.14
5007	Paul Adam	Rome	0.13
5003	Lauson Hen	San Jose	0.12

3. *Displaying all the information about all sales people*:

```
Select * from salesman;
```

salesman_id	name	city	commission
5001	James Hoog	New York	0.15
5002	Nail Knite	Paris	0.13
5005	Pit Alex	London	0.11
5006	Mc Lyon	Paris	0.14
5007	Paul Adam	Rome	0.13
5003	Lauson Hen	San Jose	0.12

Q-5) From the following table, write a SQL query to find orders that are delivered by a salesperson with ID. 5001. Return ord_no, ord_date, purch_amt:

Query:-

1. *Creating the sample table order*:

```
CREATE TABLE orders (ord_no INT, purch_amt DECIMAL(10, 2), ord_date DATE,
customer_id INT, salesman_id INT);
```

2. *Inserting values into table orders*:

```
INSERT INTO orders (ord_no, purch_amt, ord_date, customer_id, salesman_id)
VALUES
(70001, 150.50, '2012-10-05', 3005, 5002),
(70009, 270.65, '2012-09-10', 3001, 5005),
(70002, 65.26, '2012-10-05', 3002, 5001),
(70004, 110.50, '2012-08-17', 3009, 5003),
(70007, 948.50, '2012-09-10', 3005, 5002),
(70005, 2400.60, '2012-07-27', 3007, 5001),
(70008, 5760.00, '2012-09-10', 3002, 5001),
(70010, 1983.43, '2012-10-10', 3004, 5006),
(70003, 2480.40, '2012-10-10', 3009, 5003),
(70012, 250.45, '2012-06-27', 3008, 5002),
(70011, 75.29, '2012-08-17', 3003, 5007),
(70013, 3045.60, '2012-04-25', 3002, 5001);
```

ord_no	purch_amt	ord_date	customer_id	salesman_id
70001	150.50	2012-10-05	3005	5002
70009	270.65	2012-09-10	3001	5005
70002	65.26	2012-10-05	3002	5001
70004	110.50	2012-08-17	3009	5003
70007	948.50	2012-09-10	3005	5002
70005	2400.60	2012-07-27	3007	5001
70008	5760.00	2012-09-10	3002	5001
70010	1983.43	2012-10-10	3004	5006
70003	2480.40	2012-10-10	3009	5003
70012	250.45	2012-06-27	3008	5002
70011	75.29	2012-08-17	3003	5007
70013	3045.60	2012-04-25	3002	5001

3. *Finding orders that are delivered by the salesman with id 5001, returning only ord_date, ord_no, purch_amt*:

```
SELECT `ord_no`, `ord_date`, `purch_amt` FROM `orders` WHERE
salesman_id = 5001;
```

ord_no	ord_date	purch_amt
70002	2012-10-05	65.26
70005	2012-07-27	2400.60
70008	2012-09-10	5760.00
70013	2012-04-25	3045.60

Q-6) From the following table, write a SQL query to select a range of products whose price is in the range Rs.200 to Rs.600. Begin and end values are included. Return pro_id, pro_name, pro_price, and pro_com.

Query:-

1. *Creating table item_mast*;

```
CREATE TABLE item_mast (pro_id INT, pro_name VARCHAR(50), pro_price
DECIMAL(10, 2), pro_com INT);
```

2. *Inserting data into table item_mast*:

```
INSERT INTO item_mast (pro_id, pro_name, pro_price, pro_com)
VALUES
(101, 'Mother Board', 3200.00, 15),
(102, 'Key Board', 450.00, 16),
(103, 'ZIP drive', 250.00, 14),
(104, 'Speaker', 550.00, 16),
(105, 'Monitor', 5000.00, 11),
(106, 'DVD drive', 900.00, 12),
(107, 'CD drive', 800.00, 12),
(108, 'Printer', 250.00, 13),
(109, 'Refill cartridge', 350.00, 13),
(110, 'Mouse', 250.00, 12);
```

pro_id	pro_name	pro_price	pro_com
101	Mother Board	3200.00	15
102	Key Board	450.00	16
103	ZIP drive	250.00	14
104	Speaker	550.00	16
105	Monitor	5000.00	11
106	DVD drive	900.00	12
107	CD drive	800.00	12
108	Printer	250.00	13
109	Refill cartridge	350.00	13
110	Mouse	250.00	12

3. *select a range of products whose price is in the range Rs.200 to Rs.600. Begin and end values are included. Return pro_id, pro_name, pro_price, and pro_com*:

```
SELECT pro_id, pro_name, pro_price, pro_com FROM item_mast WHERE
pro_price BETWEEN 200 AND 600;
```

pro_id	pro_name	pro_price	pro_com
102	Key Board	450.00	16
103	ZIP drive	250.00	14
104	Speaker	550.00	16
108	Printer	250.00	13
109	Refill cartridge	350.00	13
110	Mouse	250.00	12

4. *From the following table, write a SQL query to calculate the average price for a manufacturer code of 16. Return avg*:

```
SELECT avg(pro_price) as avg_price FROM `item_mast` WHERE pro_com = 16;
```

avg_price
500.000000

5. *From the following table, write a SQL query to display the pro_name as 'Item Name' and pro_priceas 'Price in Rs.'*:

```
SELECT pro_name as item_name, pro_price as price_in_rs from item_mast;
```

item_name	price_in_rs
Mother Board	3200.00
Key Board	450.00
ZIP drive	250.00
Speaker	550.00
Monitor	5000.00
DVD drive	900.00
CD drive	800.00
Printer	250.00
Refill cartridge	350.00
Mouse	250.00

6. *From the following table, write a SQL query to find the items whose prices are higher than or equal to $250. Order the result by product price in descending, then product name in ascending. Return pro_name and pro_price*:

```
SELECT pro_name, pro_price FROM `item_mast` WHERE pro_price >= 250 order by pro_price desc, pro_name asc;
```

pro_name ▲ 2	pro_price ▼ 1
Monitor	5000.00
Mother Board	3200.00
DVD drive	900.00
CD drive	800.00
Speaker	550.00
Key Board	450.00
Refill cartridge	350.00
Mouse	250.00
Printer	250.00
ZIP drive	250.00

7. *From the following table, write a SQL query to calculate average price of the items for each company. Return average price and company code*:

```
SELECT pro_com, avg(pro_price) as avg_price FROM `item_mast` group by pro_com;
```

pro_com	avg_price
11	5000.000000
12	650.000000
13	300.000000
14	250.000000
15	3200.000000
16	500.000000