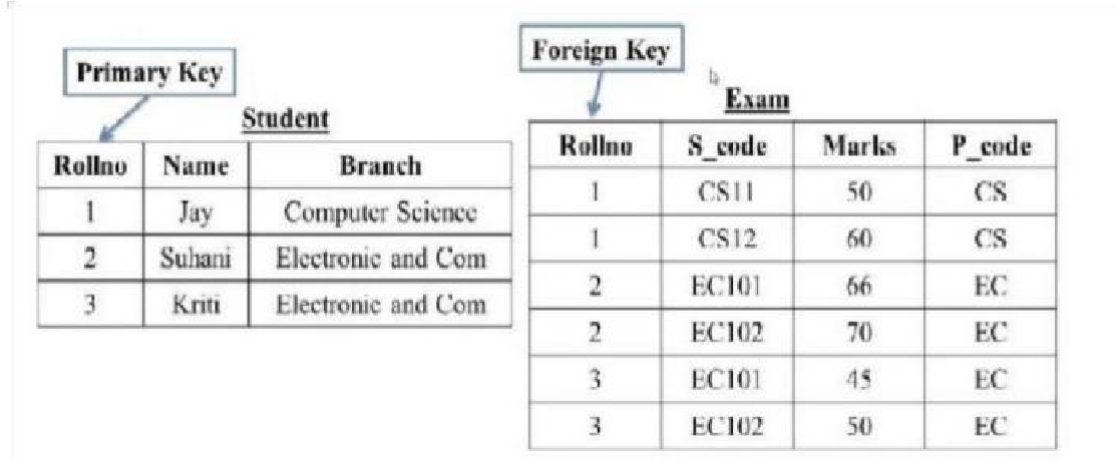
Module – 5.2

➢ SQL Queries

1. Create Table Name : Student and Exam



Student

Rollno	Name	Branch
1	Jay	Computer Science
2	Suhani	Electronic and Com
3	Kriti	Electronic and Com

Exam

Rollno	S_code	Marks	P_code
1	CS11	50	CS
1	CS12	60	CS
2	EC101	66	EC
2	EC102	70	EC
3	EC101	45	EC
3	EC102	50	EC

- **Answer:**

Student

Rollno	Name	Branch
1	Jay	Computer Science
2	Suhani	Electronic and Com
3	Kriti	Electronic and Com

Primary Key : Rollno

Exam

Rollno	S_code	Marks	P_Code
1	CS11	50	CS
1	CS12	60	CS
2	EC101	66	EC
2	EC102	70	EC
3	EC101	45	EC
3	EC102	50	EC

Foreign Key : Rollno From Student

2. Create table given below: Employee and Incentive Table

Employee_id	First_name	Last_name	Salary	Joining_date	Department
1	John	Abraham	1000000	01-JAN-13 12.00.00 AM	Banking
2	Michael	Clarke	800000	01-JAN-13 12.00.00 AM	Insurance
3	Roy	Thomas	700000	01-FEB-13 12.00.00 AM	Banking
4	Tom	Jose	600000	01-FEB-13 12.00.00 AM	Insurance
5	Jerry	Pinto	650000	01-FEB-13 12.00.00 AM	Insurance
6	Philip	Mathew	750000	01-JAN-13 12.00.00 AM	Services
7	TestName1	123	650000	01-JAN-13 12.00.00 AM	Services
8	TestName2	Lname%	600000	01-FEB-13 12.00.00 AM	Insurance

Name: Employee

Table Name: Incentive

Employee_ref_id	Incentive_date	Incentive_amount
1	01-FEB-13	5000
2	01-FEB-13	3000
3	01-FEB-13	4000
1	01-JAN-13	4500
2	01-JAN-13	3500

- **Answer:**

```
CREATE TABLE Employee(
Employee_id int NOT Null PRIMARY KEY,
    First_name varchar(40),
    Last_name varchar(40),
    Salary int,
    Joining_date Datetime,
    Department varchar(20)
);
```

DATABASE

```
CREATE TABLE Incentive(
Employee_ref_id int,
    Incentive_date Date,
    Insentive_amount int,
    FOREIGN KEY(Employee_ref_id) REFERENCES employee(Employee_id)
);
```

Employee

Employee_id	First_name	Last_name	Salary	Joining_date	Department
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	2013-02-01 12:00:00	Banking
4	Tom	Jose	600000	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

Incentive

Employee_ref_id	Incentive_date	Insentive_amount
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

3. **Get First_Name from employee table using Tom name "Employee Name".**

- **Answer:**

```
SELECT * FROM employee WHERE First_name = 'Tom';
```

Employee_id	First_name	Last_name	Salary	Joining_date	Department
4	Tom	Jose	600000	2013-02-01 12:00:00	Insurance

Second Way:

SELECT First_Name FROM employee WHERE First_name = 'Tom';

First_Name
Tom

4. **Get FIRST_NAME, Joining Date, and Salary from employee table.**

- **Solution:**

```
SELECT First_name,Joining_date,Salary FROM
employee;
```

First_name	Joining_date	Salary
John	2013-01-01 12:00:00	1000000
Michael	2013-01-01 12:00:00	800000
Roy	2013-02-01 12:00:00	700000
Tom	2013-02-01 12:00:00	600000
Jerry	2013-02-01 12:00:00	650000
Philip	2013-01-01 12:00:00	750000
TestName1	2013-01-01 12:00:00	650000
TestName2	2013-02-01 12:00:00	600000

5. **Get all employee details from the employee table order by First_Name Ascending and Salary descending?**

- **Answer:**

```
SELECT *
FROM employee
ORDER BY First_name ASC, Salary DESC;
```

- Here first_name is in ascending order so salary will not come in descending order so that's why there are 1 for first_name ascending order and when we click in salary column it give salary in descending order so 2 outputs

Employee_id	First_name ▲ 1	Last_name	Salary ▼ 2	Joining_date	Department
5	Jerry	Pinto	650000	2013-02-01 12:00:00	Insurance
1	John	Abraham	1000000	2013-01-01 12:00:00	Banking
2	Michael	Clarke	800000	2013-01-01 12:00:00	Insurance
6	Philip	Mathew	750000	2013-01-01 12:00:00	Services
3	Roy	Thomas	700000	2013-02-01 12:00:00	Banking
7	TestName1	123	650000	2013-01-01 12:00:00	Services
8	TestName2	Lname%	600000	2013-02-01 12:00:00	Insurance
4	Tom	Jose	600000	2013-02-01 12:00:00	Insurance

Employee_id	First_name	Last_name	Salary ▼ 1	Joining_date	Department
1	John	Abraham	1000000	2013-01-01 12:00:00	Banking
2	Michael	Clarke	800000	2013-01-01 12:00:00	Insurance
6	Philip	Mathew	750000	2013-01-01 12:00:00	Services
3	Roy	Thomas	700000	2013-02-01 12:00:00	Banking
5	Jerry	Pinto	650000	2013-02-01 12:00:00	Insurance
7	TestName1	123	650000	2013-01-01 12:00:00	Services
4	Tom	Jose	600000	2013-02-01 12:00:00	Insurance
8	TestName2	Lname%	600000	2013-02-01 12:00:00	Insurance

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

- **Answer:**

```
SELECT *
FROM employee
WHERE First_Name LIKE 'J%';
```

Employee_id	First_name	Last_name	Salary	Joining_date	Department
1	John	Abraham	1000000	2013-01-01 12:00:00	Banking
5	Jerry	Pinto	650000	2013-02-01 12:00:00	Insurance

7. **Get department wise maximum salary from employee table order by 8.salary ascending?**

- **Answer:**

```
SELECT Department, MAX(Salary) AS Max_Salary
FROM employee
GROUP BY Department
ORDER BY Max_Salary ASC;
```

Department	Max_Salary ▲ 1
Services	750000
Insurance	800000
Banking	1000000

9. Select first_name, incentive amount from employee and incentives table for those employees who have incentives and incentive amount greater than 3000

- **Answer:**

```
SELECT e.First_name, i.Insentive_amount
FROM employee e
JOIN incentive i ON e.Employee_id = i.Employee_ref_id
WHERE i.Insentive_amount > 3000;
```

First_name	Insentive_amount
John	5000
Roy	4000
John	4500
Michael	3500

10. Create After Insert trigger on Employee table which insert records in view table

- **Answer:**

Creating View Table :

```
CREATE TABLE View_Table(
View_id int NOT Null AUTO_INCREMENT PRIMARY KEY,
    Employee_id int,
    First_name varchar(40),
    Last_name varchar(40),
    Salary int,
    Joining_date Datetime,
    Department varchar(20)
);
```

Creating Trigger:

```
DELIMITER //

CREATE TRIGGER AfterEmployeeInsert
AFTER INSERT ON employee
FOR EACH ROW
BEGIN
    INSERT INTO view_table (Employee_id, First_name, Last_name, Salary, Joining_date, Department)
    VALUES (NEW.Employee_id, NEW.First_name, NEW.Last_name, NEW.Salary, NEW.Joining_date, NEW.Department);
END;
//

DELIMITER ;
```

Inserting Data into Employee Table:

```
INSERT INTO employee VALUES (9,'Abc','Xyz',400000,'2024-01-10 11:00:00','Computer');
INSERT INTO employee VALUES (10,'Def','Uvw',200000,'2024-01-06 11:00:00','Computer');
INSERT INTO employee VALUES (11,'Ghi','Rst',100000,'2024-01-01 11:00:00','Computer');
INSERT INTO employee VALUES (12,'Jkl','Opq',2500000,'2024-02-03 11:00:00','Computer');
```

DATABASE

Employee Table:

```
SELECT * FROM employee;
```

Employee_id	First_name	Last_name	Salary	Joining_date	Department
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	2013-02-01 12:00:00	Banking
4	Tom	Jose	600000	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
9	Abc	Xyz	400000	2024-01-10 11:00:00	Computer
10	Def	Uvw	200000	2024-01-06 11:00:00	Computer
11	Ghi	Rst	100000	2024-01-01 11:00:00	Computer
12	Jkl	Opq	2500000	2024-02-03 11:00:00	Computer

View Table:

```
SELECT * FROM view_table;
```

View_id	Employee_id	First_name	Last_name	Salary	Joining_date	Department
1	9	Abc	Xyz	400000	2024-01-10 11:00:00	Computer
2	10	Def	Uvw	200000	2024-01-06 11:00:00	Computer
3	11	Ghi	Rst	100000	2024-01-01 11:00:00	Computer
4	12	Jkl	Opq	2500000	2024-02-03 11:00:00	Computer

11. Create table given below: Salesperson and Customer

TABLE-1

TABLE NAME- SALSEPERSON

(PK)SNo	SNAME	CITY	COMM
1001	Peel	London	.12
1002	Serres	San Jose	.13
1004	Motika	London	.11
1007	Rafkin	Barcelona	.15
1003	Axelrod	New York	.1

TABLE-2

TABLE NAME- CUSTOMER

(PK)CNM.	CNAME	CITY	RATING	(FK)SNo
201	Hoffman	London	100	1001
202	Giovanne	Roe	200	1003
203	Liu	San Jose	300	1002
204	Grass	Barcelona	100	1002
206	Clemens	London	300	1007
207	Pereira	Roe	100	1004

- **Answer:**

```
CREATE TABLE Salseperson(
    PK_SNo int PRIMARY KEY,
    SNAME varchar(40),
    CITY varchar(40),
    COMM float
);
```

```
CREATE TABLE Customer(
PK_CNM int PRIMARY KEY,
    CNAME varchar(40),
    CITY varchar(40),
    RATING int,
    FK_SNo int,
    FOREIGN KEY(FK_SNo) REFERENCES salseperson(PK_SNo)
);
```

```
SELECT * FROM salseperson;
```

PK_SNo	SNAME	CITY	COMM
1001	Peel	London	0.12
1002	Serres	San Jose	0.13
1003	Axelrod	New York	0.1
1004	Motika	London	0.11
1007	Rafkin	Barcelona	0.15

```
SELECT * FROM customer;
```

PK_CNM	CNAME	CITY	RATING	FK_SNo
201	Hoffman	London	100	1001
202	Giovanne	Reo	200	1003
203	Liu	San Jose	300	1002
204	Grass	Barcelona	100	1002
206	Clemens	London	300	1007
207	Pereira	Roe	100	1004

12. Retrieve the below data from above table

14. Names and cities of all salespeople in London with commission above 0.12

- **Answer:**

```
SELECT SNAME, CITY
FROM salseperson
WHERE CITY = 'London' AND COMM > 0.12;
```

➢ Here there is no output because there are no salespersons in this table whose city name is London and whose commission is greater than 0.12

15. All salespeople either in Barcelona or in London

```
SELECT SNAME, CITY FROM salseperson WHERE CITY = 'London' AND
COMM > 0.12;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

SNAME	CITY

Query results operations

Create view

```
SELECT *
FROM salseperson
WHERE CITY IN ('Barcelona', 'London');
```

PK_SNo	SNAME	CITY	COMM
1001	Peel	London	0.12
1004	Motika	London	0.11
1007	Rafkin	Barcelona	0.15

16. All salespeople with commission between 0.10 and 0.12. (Boundary values should be excluded).

- **Answer:**

```
SELECT *
FROM salseperson
WHERE COMM > 0.10 AND COMM < 0.12;
```

PK_SNo	SNAME	CITY	COMM
1001	Peel	London	0.12
1003	Axelrod	New York	0.1
1004	Motika	London	0.11

17. All customers excluding those with rating <= 100 unless they are located in Rome

- **Answer:**

```
SELECT *
FROM customer
WHERE RATING > 100 OR (CITY = 'Rome' AND
RATING <= 100);
```

PK_CNM	CNAME	CITY	RATING	FK_SNo
202	Giovanne	Reo	200	1003
203	Liu	San Jose	300	1002
206	Clemens	London	300	1007

18. Write a SQL statement that displays all the information about all salespeople

```
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
```

- **Answer:**

```
CREATE TABLE Salespeople(
salesman_id int Not Null PRIMARY KEY,
    name varchar(40),
    city varchar(40),
    commisstion float
);
```

```
SELECT * FROM salespeople;
```

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

19. 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.

ord_no	purch_amt	ord_date	customer_id	salesman_id
70001	150.5	2012-10-05	3005	5002
70009	270.65	2012-09-10	3001	5005
70002	65.26	2012-10-05	3002	5001
70004	110.5	2012-08-17	3009	5003
70007	948.5	2012-09-10	3005	5002
70005	2400.6	2012-07-27	3007	5001
70008	5760	2012-09-10	3002	5001
70010	1983.43	2012-10-10	3004	5006
70003	2480.4	2012-10-10	3009	5003
70012	250.45	2012-06-27	3008	5002
70011	75.29	2012-08-17	3003	5007
70013	3045.6	2012-04-25	3002	5001

- **Answer:**

```
CREATE TABLE orders(
ord_no int Not Null PRIMARY KEY,
    purch_amt float,
    ord_date DATE,
    customer_id int,
    salesman_id int,
    FOREIGN KEY (salesman_id) REFERENCES salespeople(salesman_id)
);
```

```
SELECT * FROM orders;
```

ord_no	purch_amt	ord_date	customer_id	salesman_id
70001	150.5	2012-10-05	3005	5002
70002	65.26	2012-10-05	3002	5001
70003	2480.4	2012-10-10	3009	5003
70004	110.5	2012-08-17	3009	5003
70005	2400.6	2012-07-27	3007	5001
70007	948.5	2012-09-10	3005	5002
70008	5760	2012-09-10	3002	5001
70009	270.65	2012-09-10	3001	5005
70010	1983.43	2012-10-10	3004	5006
70011	75.29	2012-08-17	3003	5007
70012	250.45	2012-06-27	3008	5002
70013	3045.6	2012-04-25	3002	5001

```
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	2400.6
70008	2012-09-10	5760
70013	2012-04-25	3045.6

All orders for more than $1000.

```
SELECT *
FROM orders
WHERE purch_amt > 1000;
```

ord_no	purch_amt	ord_date	customer_id	salesman_id
70003	2480.4	2012-10-10	3009	5003
70005	2400.6	2012-07-27	3007	5001
70008	5760	2012-09-10	3002	5001
70010	1983.43	2012-10-10	3004	5006
70013	3045.6	2012-04-25	3002	5001

20. 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.

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	2600.00	13
109	Refill cartridge	350.00	13
110	Mouse	250.00	12

- **Answer:**

```
CREATE TABLE item_mast(
PRO_ID int NOT NULL PRIMARY KEY,
    PRO_NAME varchar(40),
    PRO_PRICE float,
    PRO_COM int
);
```

```
SELECT * FROM item_mast;
```

PRO_ID	PRO_NAME	PRO_PRICE	PRO_COM
102	Key Board	450	16
103	ZIP drive	250	14
104	Speaker	550	16
105	Monitor	5000	11
106	DVD drive	900	12
107	CD drive	800	12
108	Printer	2600	13
109	Refill cartridge	350	13
110	Mouse	250	12
101	Mother Board	3200	15

```
SELECT PRO_ID, PRO_NAME, PRO_NAME, PRO_COM
FROM item_mast
WHERE PRO_PRICE BETWEEN 200 AND 600;
```

PRO_ID	PRO_NAME	PRO_NAME	PRO_COM
102	Key Board	Key Board	16
103	ZIP drive	ZIP drive	14
104	Speaker	Speaker	16
109	Refill cartridge	Refill cartridge	13
110	Mouse	Mouse	12

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

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	2600.00	13
109	Refill cartridge	350.00	13
110	Mouse	250.00	12

```
SELECT AVG(PRO_PRICE) AS avg_price
FROM item_mast
WHERE PRO_COM = 16;
```

avg_price
500

22. From the following table, write a SQL query to display the pro_nameas 'Item Name' and pro_priceas 'Price in Rs.'

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	2600.00	13
109	Refill cartridge	350.00	13
110	Mouse	250.00	12

- **Answer:**

```
SELECT PRO_NAME AS 'Item Name', PRO_PRICE AS
'Price in Rs.' FROM item_mast;
```

Item Name	Price in Rs.
Key Board	450
ZIP drive	250
Speaker	550
Monitor	5000
DVD drive	900
CD drive	800
Printer	2600
Refill cartridge	350
Mouse	250
Mother Board	3200

23. 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.

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	2600.00	13
109	Refill cartridge	350.00	13
110	Mouse	250.00	12

- **Answer:**

```
SELECT PRO_NAME, PRO_PRICE
FROM item_mast
WHERE PRO_PRICE >= 250
ORDER BY PRO_PRICE DESC, PRO_NAME ASC;
```

PRO_NAME	PRO_PRICE ▲ 1
ZIP drive	250
Mouse	250
Refill cartridge	350
Key Board	450
Speaker	550
CD drive	800
DVD drive	900
Printer	2600
Mother Board	3200
Monitor	5000

PRO_NAME ▲ 2	PRO_PRICE ▼ 1
Monitor	5000
Mother Board	3200
Printer	2600
DVD drive	900
CD drive	800
Speaker	550
Key Board	450
Refill cartridge	350
Mouse	250
ZIP drive	250

- Here pro_name is in ascending order so pro_price will not come in descending order so that's why there are 1 for pro_name ascending order and when we click in salary column it give pro_price in descending order so 2 outputs

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

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	2600.00	13
109	Refill cartridge	350.00	13
110	Mouse	250.00	12

- **Answer:**

```
SELECT AVG(PRO_PRICE) AS avg_price, PRO_COM
FROM item_mast
GROUP BY PRO_COM;
```

avg_price	PRO_COM
5000	11
650	12
1475	13
250	14
3200	15
500	16