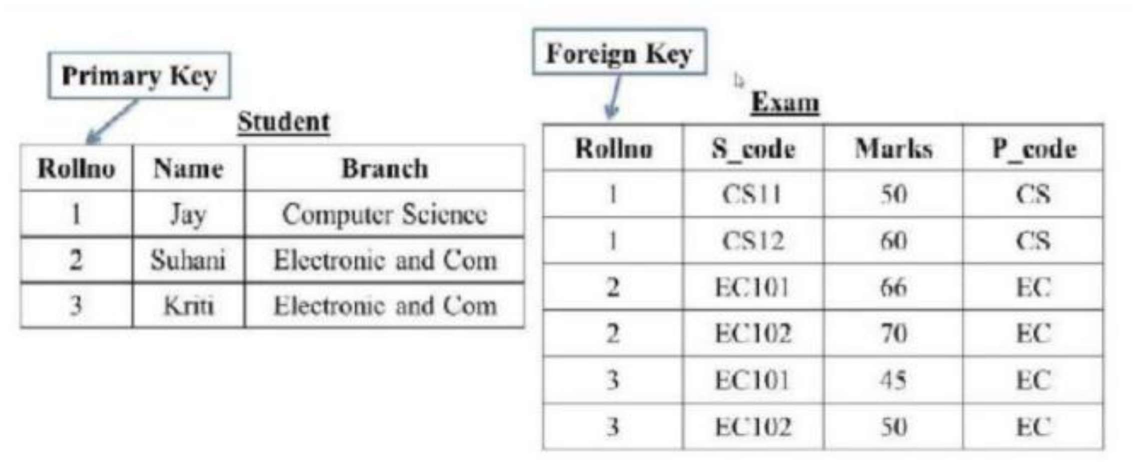
MODULE 5.2 : SQL Queries

1. Create Table Name : Student and Exam



Primary Key
Foreign Key

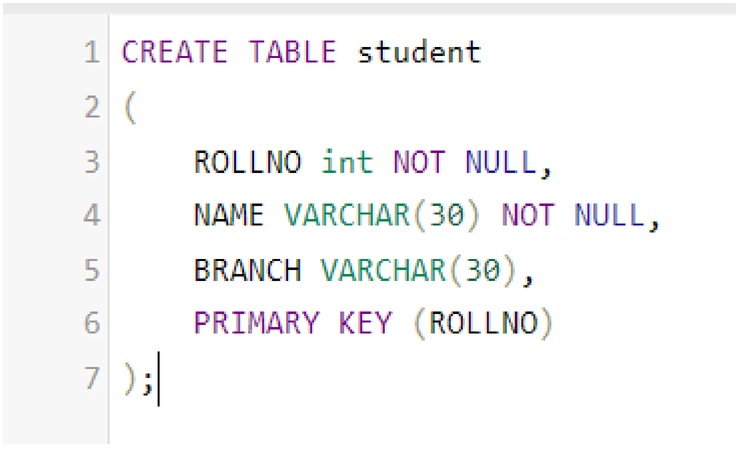
Student

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

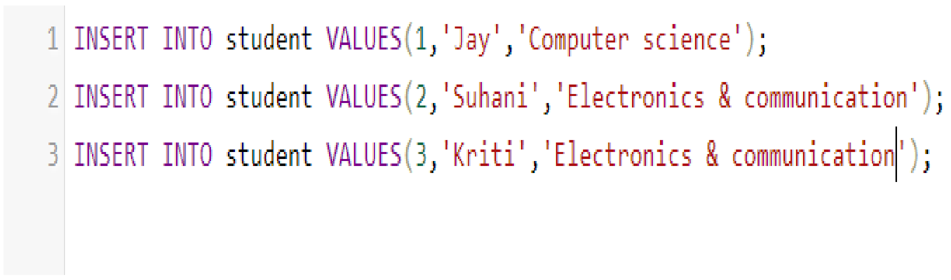
Exam

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

➢ Create Student Table:

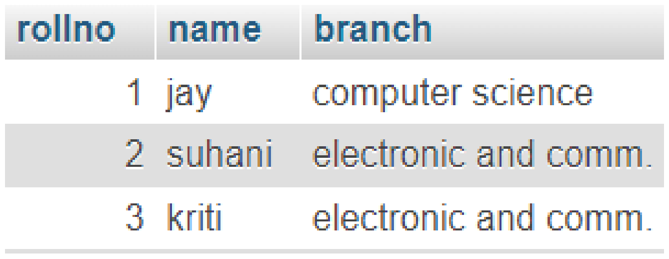


```
CREATE TABLE student
(
    ROLLNO int NOT NULL,
    NAME VARCHAR(30) NOT NULL,
    BRANCH VARCHAR(30),
    PRIMARY KEY (ROLLNO)
);
```



```
INSERT INTO student VALUES(1,'Jay','Computer science');
INSERT INTO student VALUES(2,'Suhani','Electronics & communication');
INSERT INTO student VALUES(3,'Kriti','Electronics & communication');
```

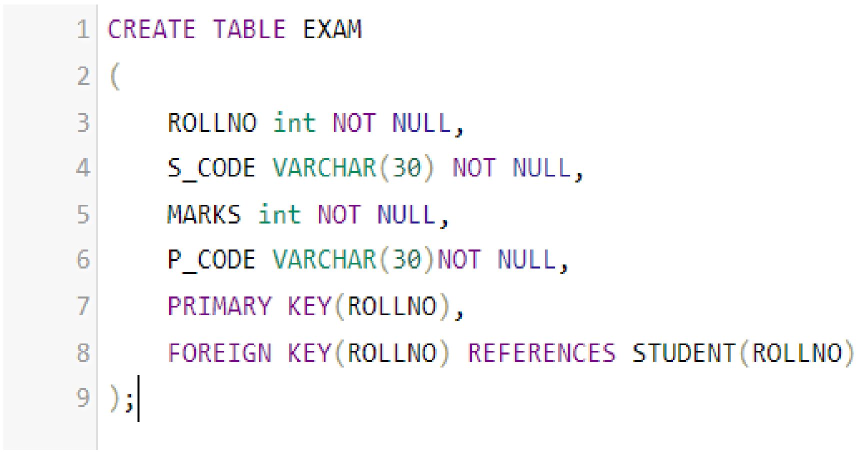
➢ Result:



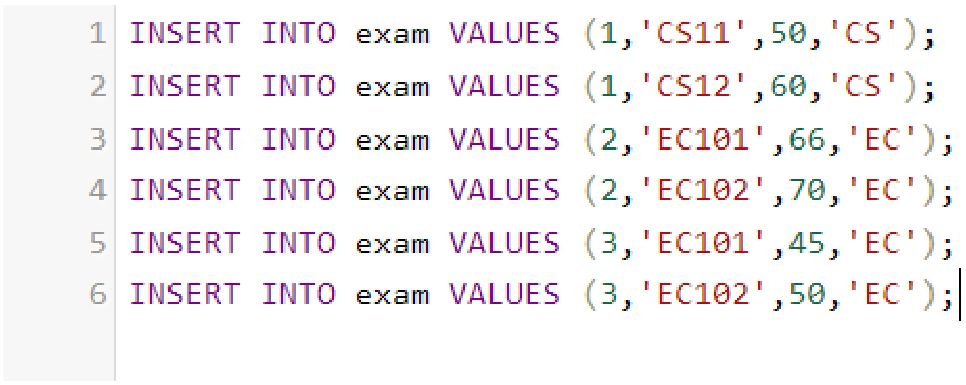
rollno	name	branch
1	jay	computer science
2	suhani	electronic and comm.
3	kriti	electronic and comm.

MODULE 5.2 : SQL Queries

➢ Exam Table:

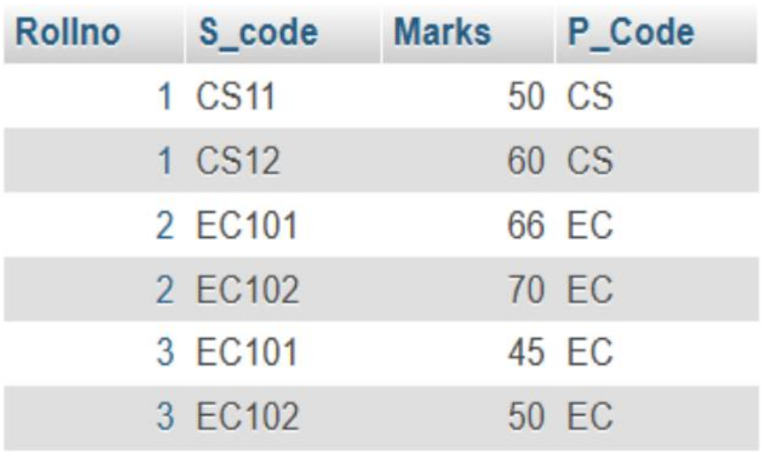


```
CREATE TABLE EXAM
(
    ROLLNO int NOT NULL,
    S_CODE VARCHAR(30) NOT NULL,
    MARKS int NOT NULL,
    P_CODE VARCHAR(30)NOT NULL,
    PRIMARY KEY(ROLLNO),
    FOREIGN KEY(ROLLNO) REFERENCES STUDENT(ROLLNO)
);
```



```
INSERT INTO exam VALUES (1,'CS11',50,'CS');
INSERT INTO exam VALUES (1,'CS12',60,'CS');
INSERT INTO exam VALUES (2,'EC101',66,'EC');
INSERT INTO exam VALUES (2,'EC102',70,'EC');
INSERT INTO exam VALUES (3,'EC101',45,'EC');
INSERT INTO exam VALUES (3,'EC102',50,'EC');
```

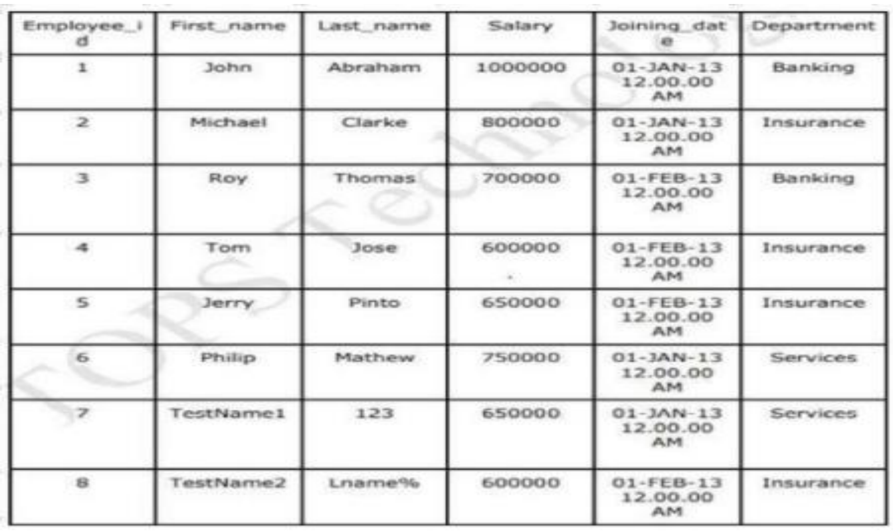
➢ Result:



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

2. **Create table given below: Employee and Incentive Table**

MODULE 5.2 : SQL Queries

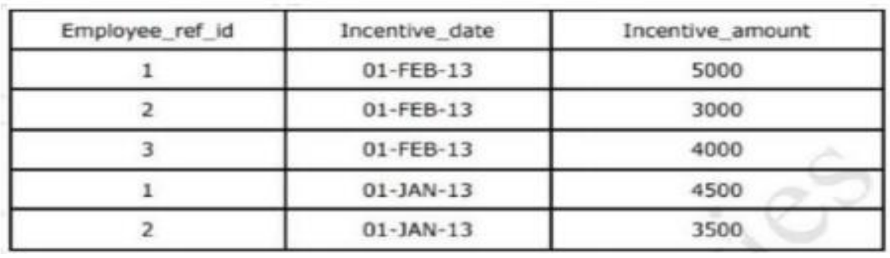


Employee_id	First_name	Last_name	Salary	Joining_date	Department
1	John	Abraham	1000000	01-JAN-13 12.00.00 AM	Banking
2	Michael	Clarke	800000	01-JAN-13 12.00.00 AM	Insurance
3	Roy	Thomas	700000	01-FEB-13 12.00.00 AM	Banking
4	Tom	Jose	600000	01-FEB-13 12.00.00 AM	Insurance
5	Jerry	Pinto	650000	01-FEB-13 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:

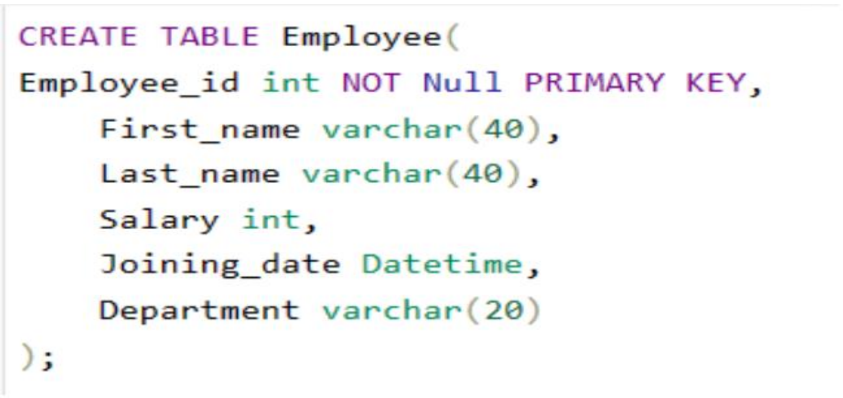
Incentive



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

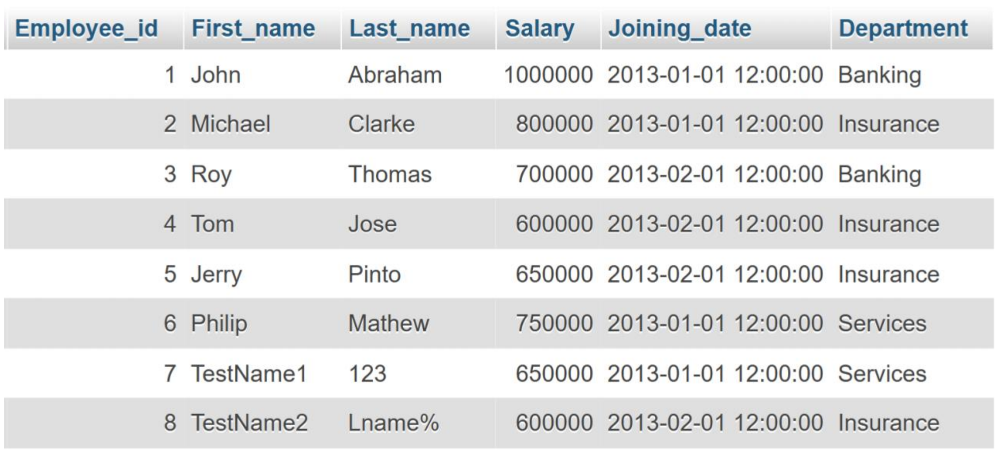
- Solution:

  Employee:



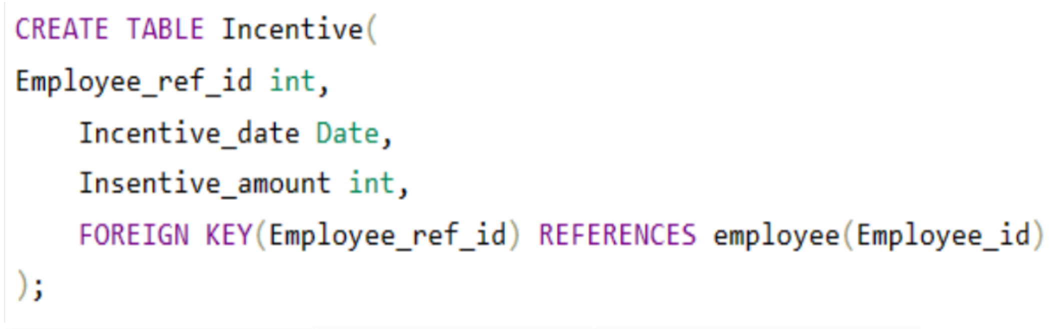
```
CREATE TABLE Employee(
Employee_id int NOT Null PRIMARY KEY,
    First_name varchar(40),
    Last_name varchar(40),
    Salary int,
    Joining_date Datetime,
    Department varchar(20)
);
```

MODULE 5.2 : SQL Queries

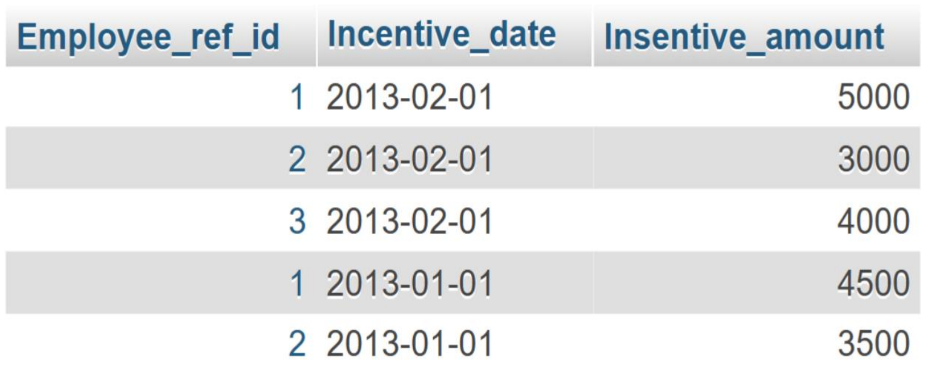


Employee_id	First_name	Last_name	Salary	Joining_date	Department
1	John	Abraham	1000000	2013-01-01 12:00:00	Banking
2	Michael	Clarke	800000	2013-01-01 12:00:00	Insurance
3	Roy	Thomas	700000	2013-02-01 12:00:00	Banking
4	Tom	Jose	600000	2013-02-01 12:00:00	Insurance
5	Jerry	Pinto	650000	2013-02-01 12:00:00	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:



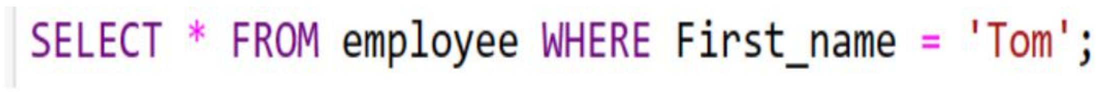
```
CREATE TABLE Incentive(
Employee_ref_id int,
    Incentive_date Date,
    Insentive_amount int,
    FOREIGN KEY(Employee_ref_id) REFERENCES employee(Employee_id)
);
```



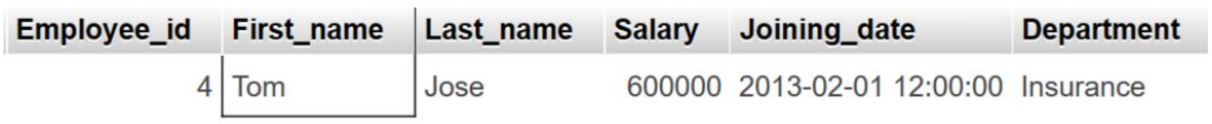
Employee_ref_id	Incentive_date	Insentive_amount
1	2013-02-01	5000
2	2013-02-01	3000
3	2013-02-01	4000
1	2013-01-01	4500
2	2013-01-01	3500

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

➢ Solution:



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



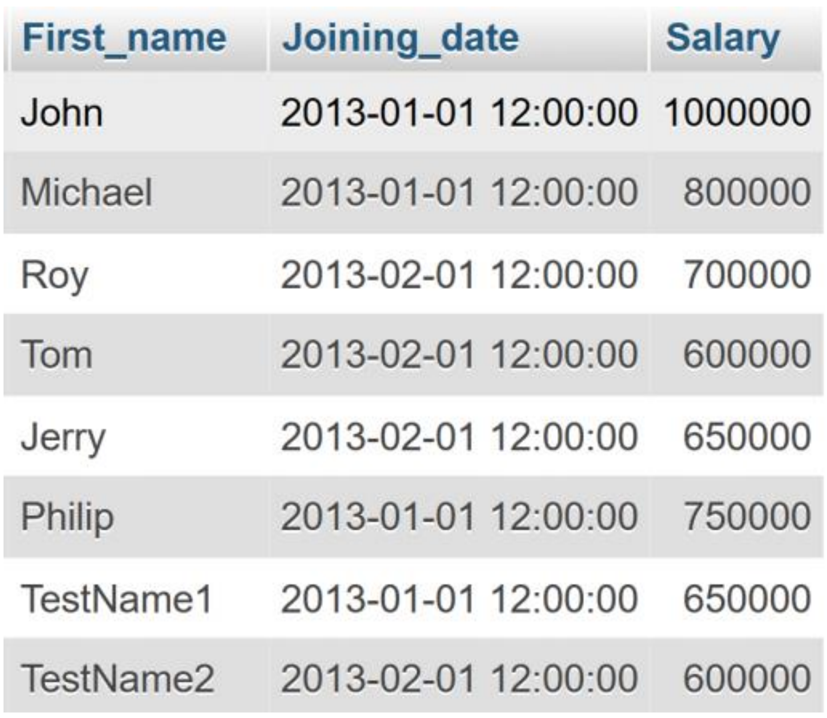
Employee_id	First_name	Last_name	Salary	Joining_date	Department
4	Tom	Jose	600000	2013-02-01 12:00:00	Insurance

MODULE 5.2 : SQL Queries

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

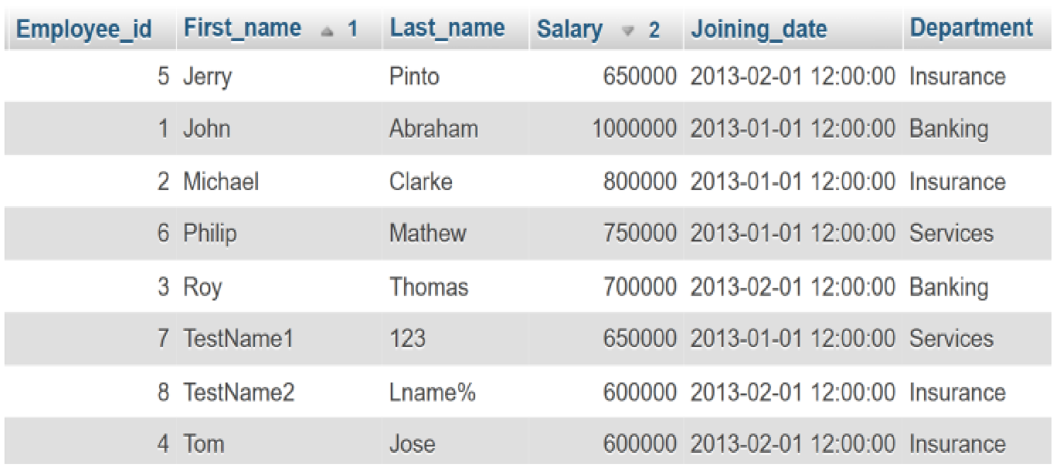
5. **Get all employee details from the employee table order by First_Name Ascending and Salary descending?**

➢ Solution:

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

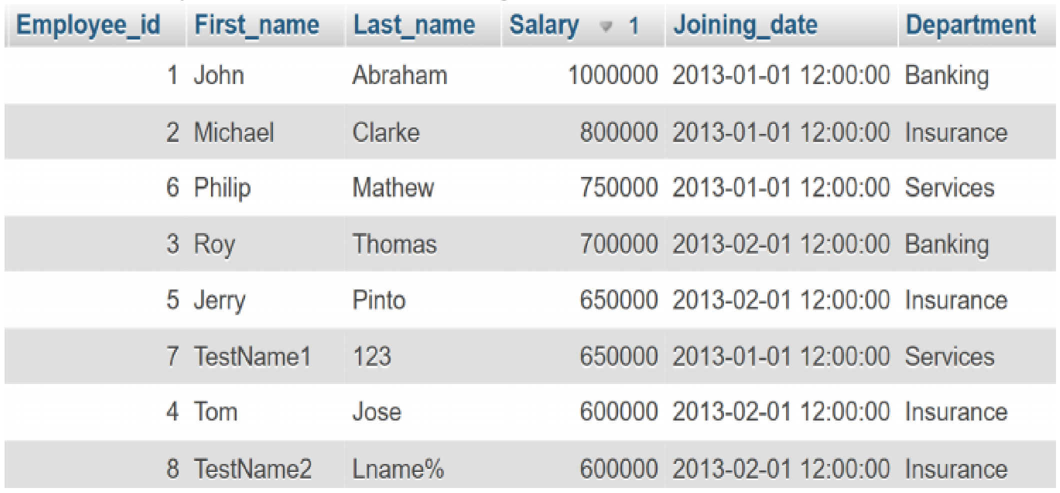
- First_name is in ascending order:

MODULE 5.2 : SQL Queries



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

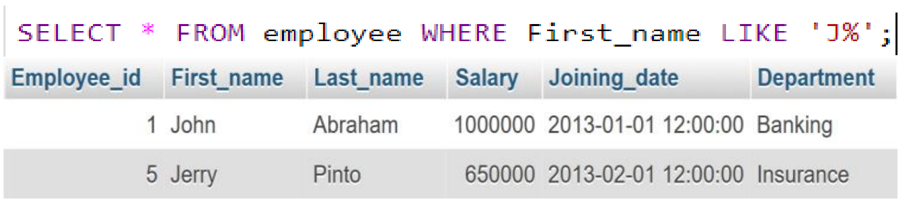
- Salary is in descending order:



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’.**

➢ solution:

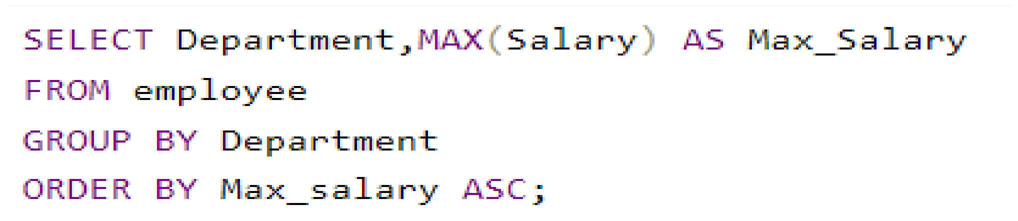


```
SELECT * FROM employee WHERE First_name LIKE 'J%';
```

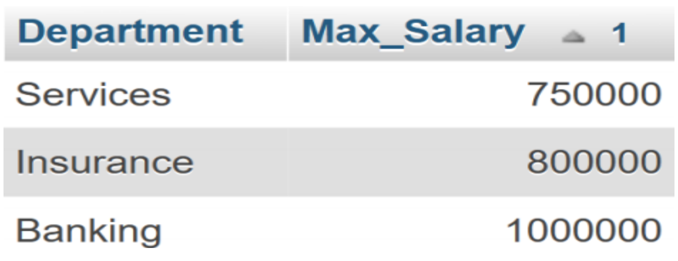
Employee_id	First_name	Last_name	Salary	Joining_date	Department
1	John	Abraham	1000000	2013-01-01 12:00:00	Banking
5	Jerry	Pinto	650000	2013-02-01 12:00:00	Insurance

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

➢ Solution:



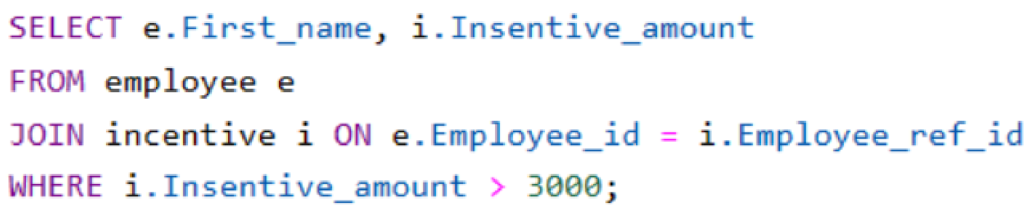
```
SELECT Department,MAX(Salary) AS Max_Salary
FROM employee
GROUP BY Department
ORDER BY Max_salary ASC;
```



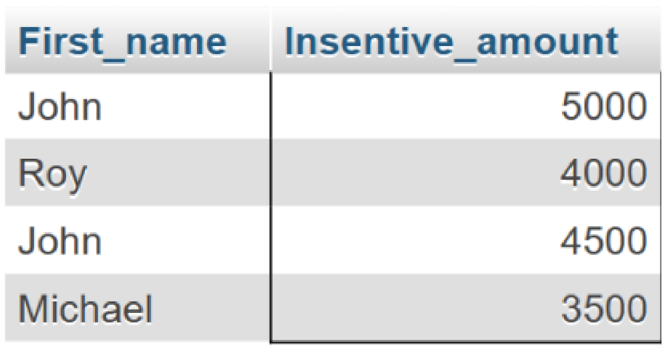
Department	Max_Salary ▲ 1
Services	750000
Insurance	800000
Banking	1000000

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

➢ Solution:



```
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;
```



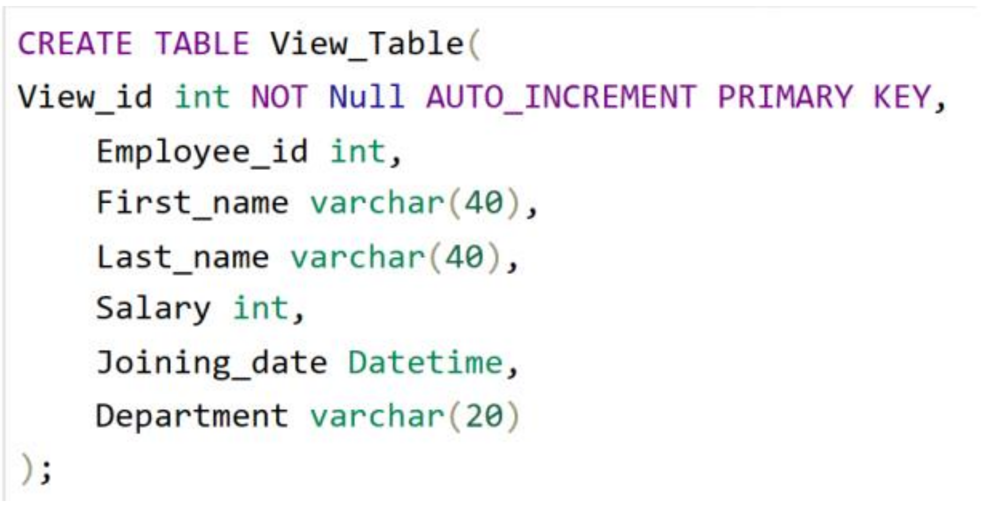
First_name	Insentive_amount
John	5000
Roy	4000
John	4500
Michael	3500

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

➢ Solution:

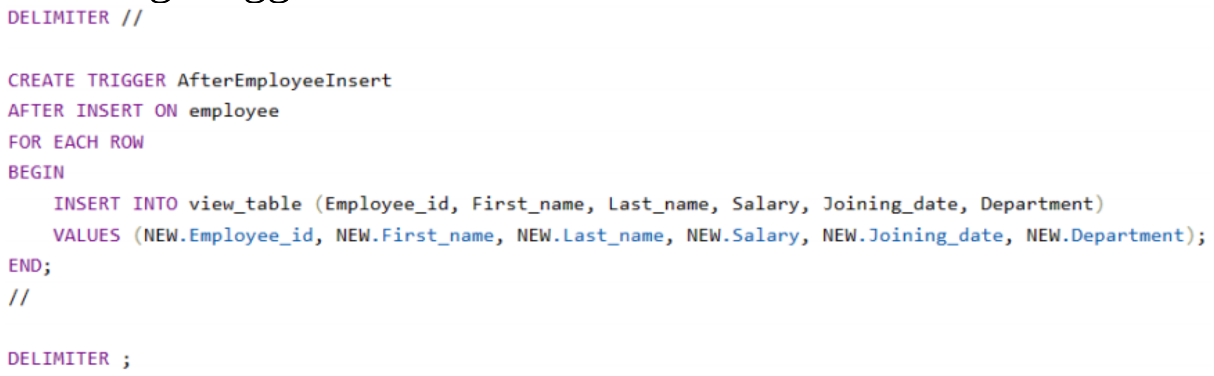
- Creating View Table:

MODULE 5.2 : SQL Queries



```
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:

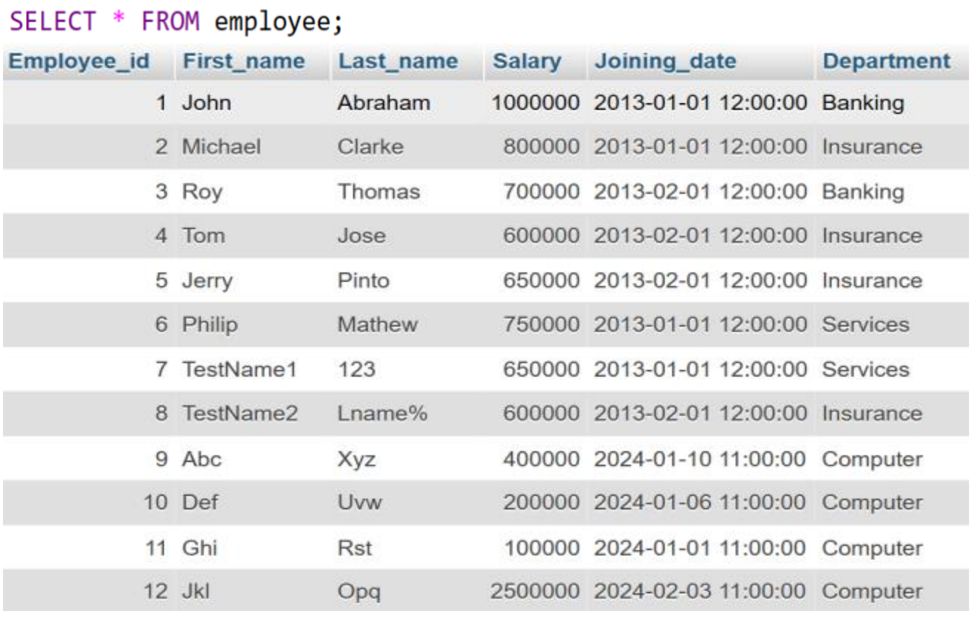


```
DELIMITER //

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 ;
```

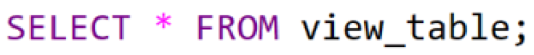
- Employee Table:



```
SELECT * FROM employee;
```

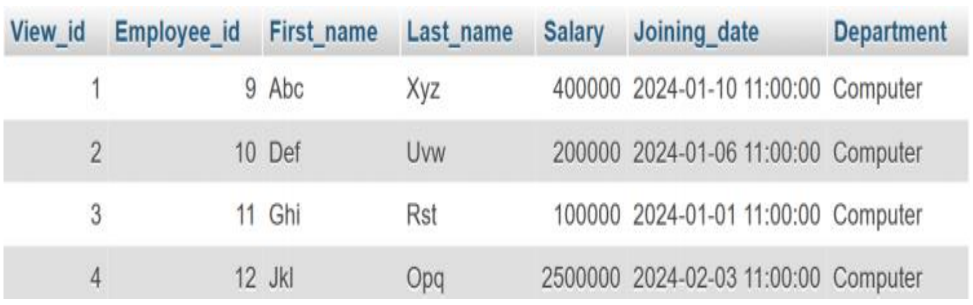
Employee_id	First_name	Last_name	Salary	Joining_date	Department
1	John	Abraham	1000000	2013-01-01 12:00:00	Banking
2	Michael	Clarke	800000	2013-01-01 12:00:00	Insurance
3	Roy	Thomas	700000	2013-02-01 12:00:00	Banking
4	Tom	Jose	600000	2013-02-01 12:00:00	Insurance
5	Jerry	Pinto	650000	2013-02-01 12:00:00	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;
```

MODULE 5.2 : SQL Queries



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

10. Create table given below: Salesperson and Customer.

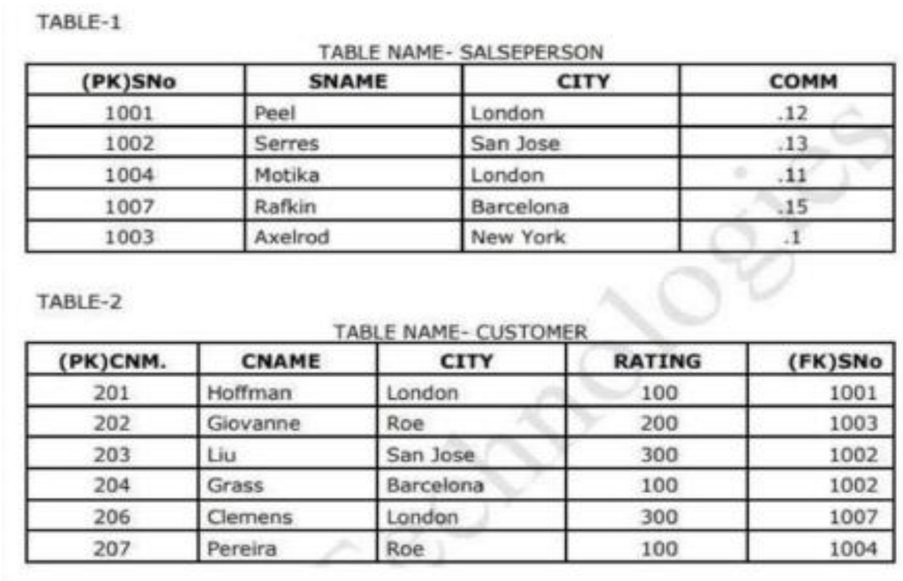


TABLE-1

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

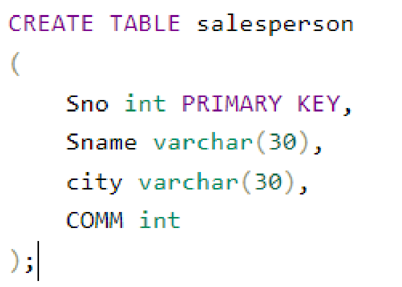
TABLE-2

TABLE NAME- CUSTOMER

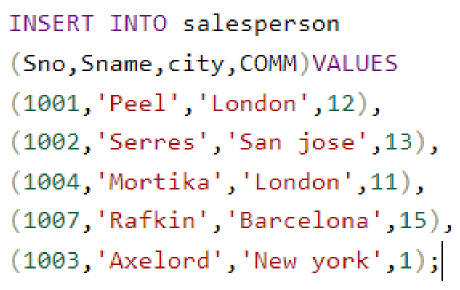
(PK)CNM.	CNAME	CITY	RATING	(FK)SNo
201	Hoffman	London	100	1001
202	Giovanne	Roe	200	1003
203	Liu	San Jose	300	1002
204	Grass	Barcelona	100	1002
206	Clemens	London	300	1007
207	Pereira	Roe	100	1004

➢ Solution:

- Creating salesperson table:

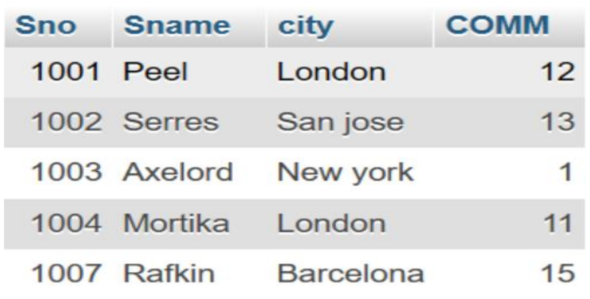


```
CREATE TABLE salesperson
(
    Sno int PRIMARY KEY,
    Sname varchar(30),
    city varchar(30),
    COMM int
);
```



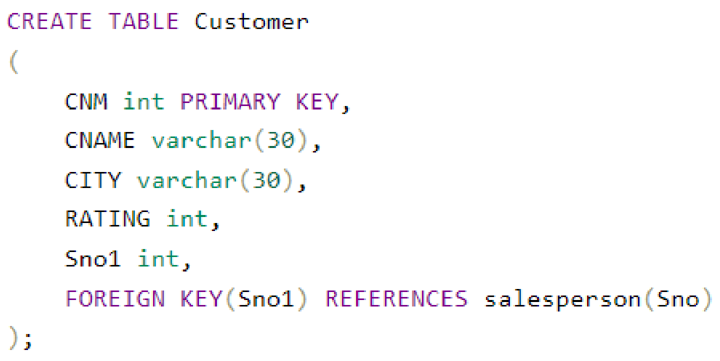
```
INSERT INTO salesperson
(Sno,Sname,city,COMM)VALUES
(1001,'Peel','London',12),
(1002,'Serres','San jose',13),
(1004,'Mortika','London',11),
(1007,'Rafkin','Barcelona',15),
(1003,'Axelord','New york',1);
```

MODULE 5.2 : SQL Queries

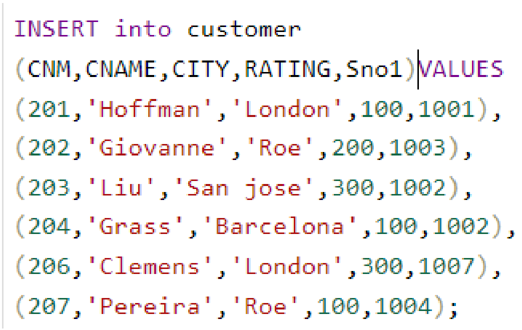


Sno	Sname	city	COMM
1001	Peel	London	12
1002	Serres	San jose	13
1003	Axelord	New york	1
1004	Mortika	London	11
1007	Rafkin	Barcelona	15

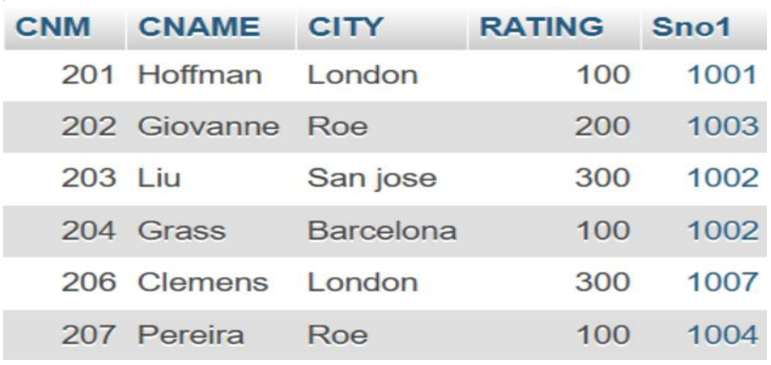
- Creating customer table:



```
CREATE TABLE Customer
(
    CNM int PRIMARY KEY,
    CNAME varchar(30),
    CITY varchar(30),
    RATING int,
    Sno1 int,
    FOREIGN KEY(Sno1) REFERENCES salesperson(Sno)
);
```



```
INSERT into customer
(CNM,CNAME,CITY,RATING,Sno1)VALUES
(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);
```

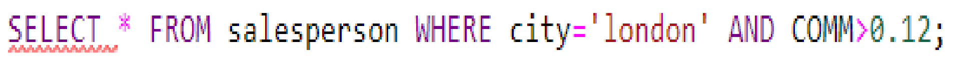


CNM	CNAME	CITY	RATING	Sno1
201	Hoffman	London	100	1001
202	Giovanne	Roe	200	1003
203	Liu	San jose	300	1002
204	Grass	Barcelona	100	1002
206	Clemens	London	300	1007
207	Pereira	Roe	100	1004

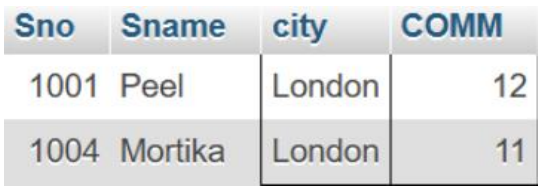
11. Retrieve the below data from above table
12. All orders for more than $1000.
13. Names and cities of all salespeople in London with commission above 0.12

MODULE 5.2 : SQL Queries

➢ Solution:



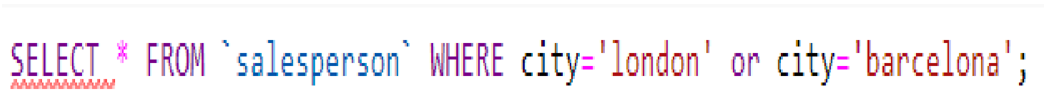
```
SELECT * FROM salesperson WHERE city='london' AND COMM>0.12;
```



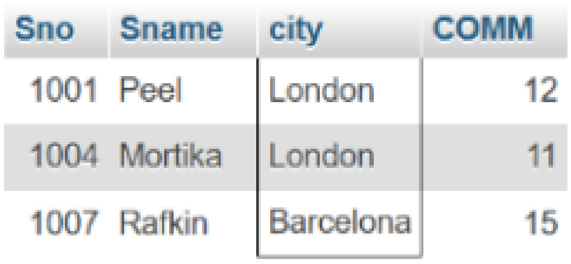
Sno	Sname	city	COMM
1001	Peel	London	12
1004	Mortika	London	11

14. All salesperson either in barcelona or in London.

➢ Solution:



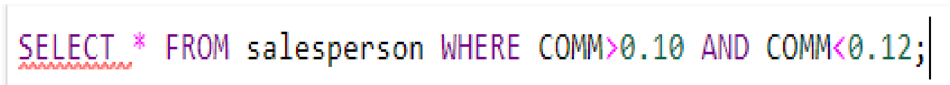
```
SELECT * FROM `salesperson` WHERE city='london' or city='barcelona';
```



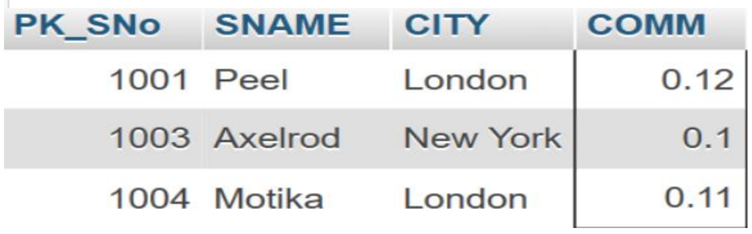
Sno	Sname	city	COMM
1001	Peel	London	12
1004	Mortika	London	11
1007	Rafkin	Barcelona	15

15. All salespeople with commission between 0.10 and 0.12. (Boundaryvaluesshould be excluded).

➢ Solution:



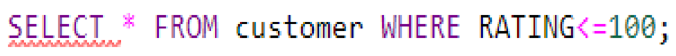
```
SELECT * FROM salesperson WHERE COMM>0.10 AND COMM<0.12;
```



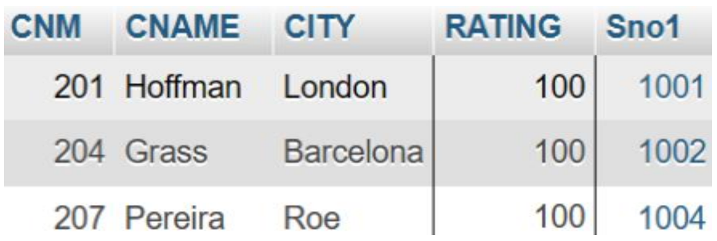
PK_SNo	SNAME	CITY	COMM
1001	Peel	London	0.12
1003	Axelrod	New York	0.1
1004	Motika	London	0.11

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

➢ Solution:

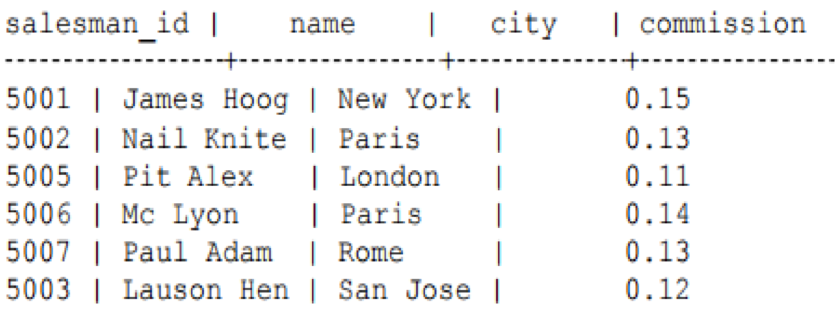


```
SELECT * FROM customer WHERE RATING<=100;
```



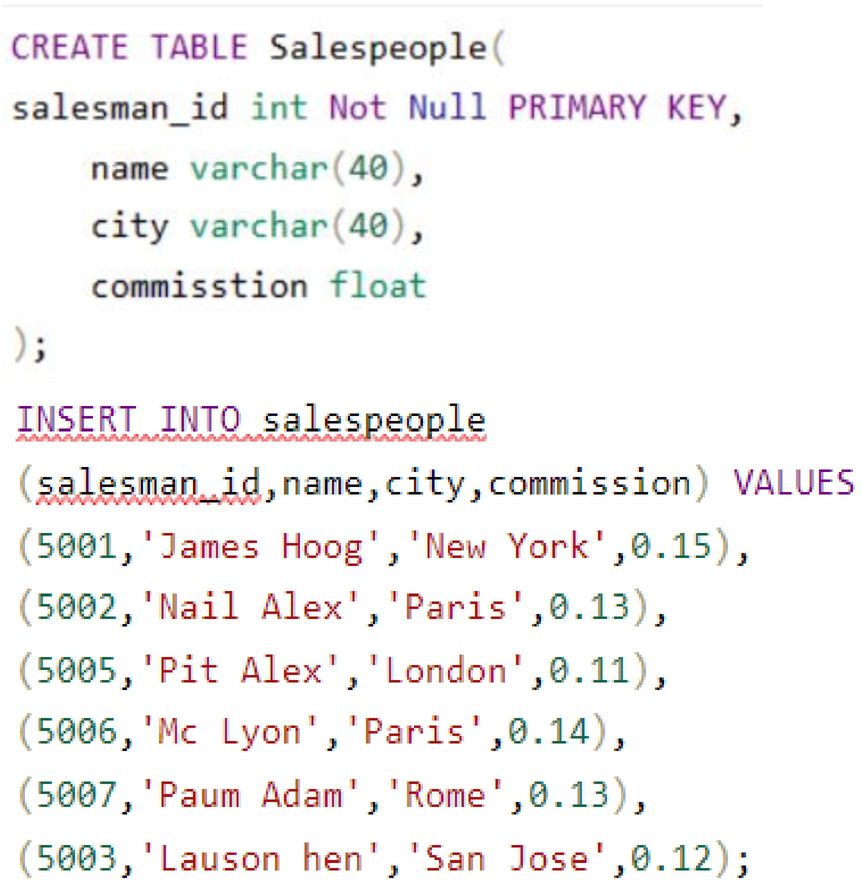
CNM	CNAME	CITY	RATING	Sno1
201	Hoffman	London	100	1001
204	Grass	Barcelona	100	1002
207	Pereira	Roe	100	1004

17. 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
```

➢ Solution:



```
CREATE TABLE Salespeople(
salesman_id int Not Null PRIMARY KEY,
    name varchar(40),
    city varchar(40),
    commisstion float
);
INSERT INTO salespeople
(salesman_id,name,city,commission) VALUES
(5001,'James Hoog','New York',0.15),
(5002,'Nail Alex','Paris',0.13),
(5005,'Pit Alex','London',0.11),
(5006,'Mc Lyon','Paris',0.14),
(5007,'Paum Adam','Rome',0.13),
(5003,'Lauson hen','San Jose',0.12);
```

MODULE 5.2 : SQL Queries

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

18. From the following table, write a SQL query to find orders that are delivered by salesperson with id.5001.Return ord_no, ord_name, purch_amt.

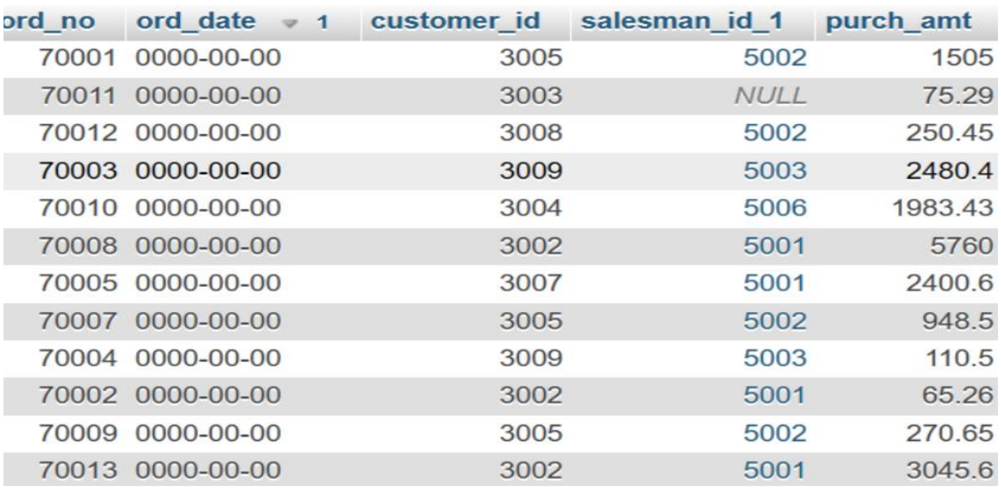
```
ord_no          purch_amt       ord_date        customer_id     salesman_id
--------------- --------------- --------------- --------------- ---------------
70001           150.5           2012-10-05      3005            5002
70009           270.65          2012-09-10      3001            5005
70002           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
```

- Solution:

```
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)
);
```

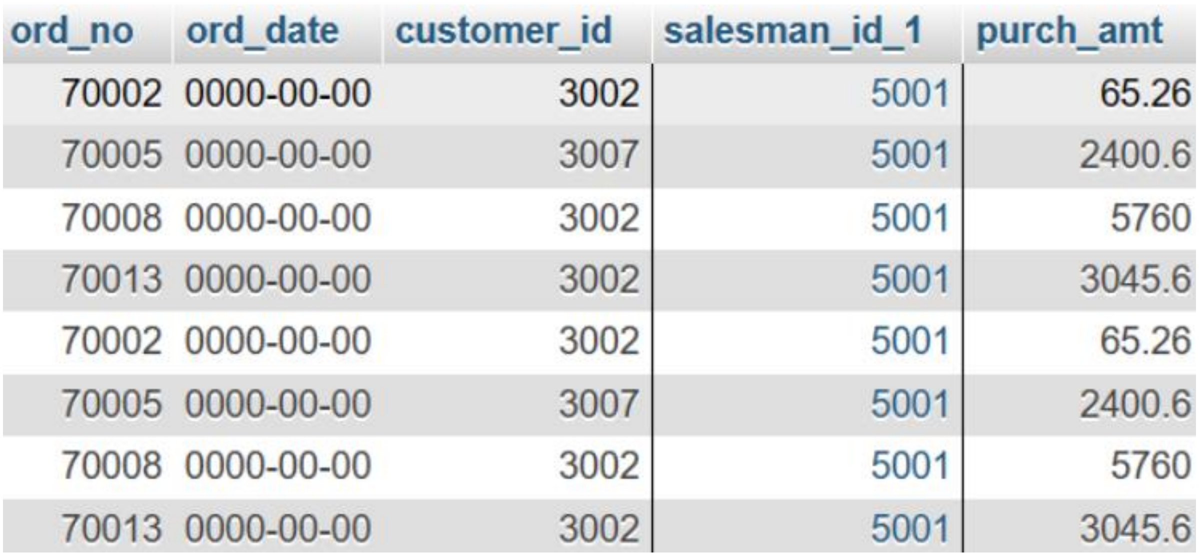
MODULE 5.2 : SQL Queries

```
INSERT INTO orders
(ord_no,ord_date,customer_id,salesman_id_1,purch_amt) VALUES
(70001,2012-10-05,3005,5002,1505),
(70009,2012-09-10,3005,5002,270.65),
(70002,2012-10-05,3002,5001,65.26),
(70004,2012-08-17,3009,5003,110.5),
(70007,2012-09-17,3005,5002,948.5),
(70005,2012-07-27,3007,5001,2400.6),
(70008,2012-09-10,3002,5001,5760),
(70010,2012-10-10,3004,5006,1983.43),
(70003,2012-10-10,3009,5003,2480.4),
(70012,2012-06-27,3008,5002,250.45),
(70011,2012-08-17,3003,null,75.29),
(70013,2012-04-25,3002,5001,3045.6);
```



ord_no	ord_date ▼ 1	customer_id	salesman_id_1	purch_amt
70001	0000-00-00	3005	5002	1505
70011	0000-00-00	3003	NULL	75.29
70012	0000-00-00	3008	5002	250.45
70003	0000-00-00	3009	5003	2480.4
70010	0000-00-00	3004	5006	1983.43
70008	0000-00-00	3002	5001	5760
70005	0000-00-00	3007	5001	2400.6
70007	0000-00-00	3005	5002	948.5
70004	0000-00-00	3009	5003	110.5
70002	0000-00-00	3002	5001	65.26
70009	0000-00-00	3005	5002	270.65
70013	0000-00-00	3002	5001	3045.6

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

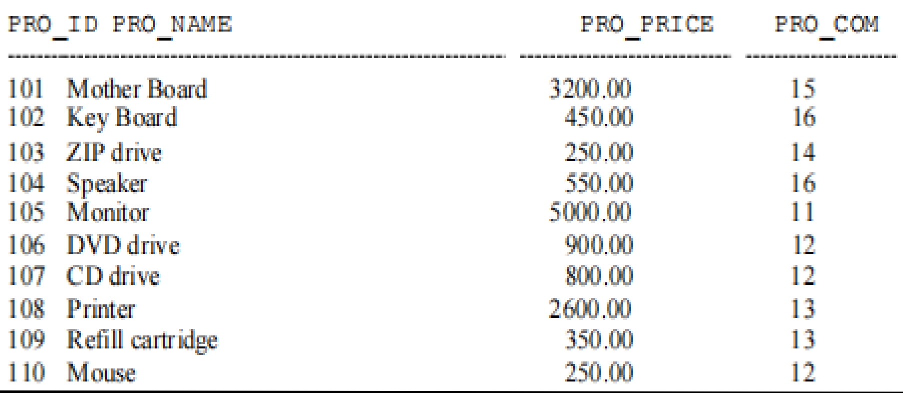


ord_no	ord_date	customer_id	salesman_id_1	purch_amt
70002	0000-00-00	3002	5001	65.26
70005	0000-00-00	3007	5001	2400.6
70008	0000-00-00	3002	5001	5760
70013	0000-00-00	3002	5001	3045.6
70002	0000-00-00	3002	5001	65.26
70005	0000-00-00	3007	5001	2400.6
70008	0000-00-00	3002	5001	5760
70013	0000-00-00	3002	5001	3045.6

MODULE 5.2 : SQL Queries

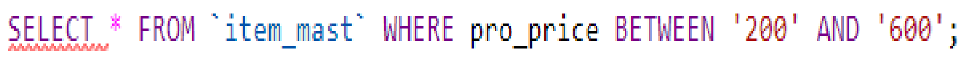
19. From the following table, write a SQL query to select a range of products whose price is in the range Rs.200 to Rs.600. Begin and end values are included. Return pro_id, pro_name, pro_price, and pro_com.

Sample table: item_mast

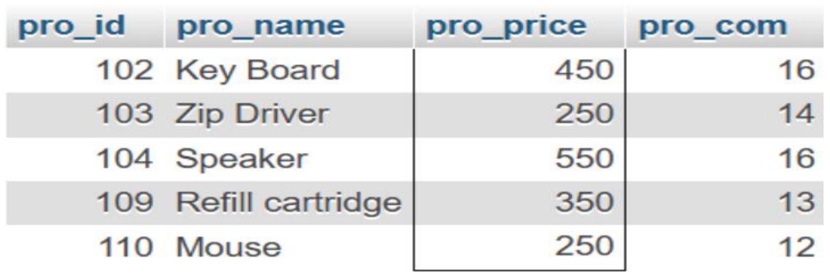


PRO_ID	PRO_NAME	PRO_PRICE	PRO_COM
101	Mother Board	3200.00	15
102	Key Board	450.00	16
103	ZIP drive	250.00	14
104	Speaker	550.00	16
105	Monitor	5000.00	11
106	DVD drive	900.00	12
107	CD drive	800.00	12
108	Printer	2600.00	13
109	Refill cartridge	350.00	13
110	Mouse	250.00	12

➢ Solution:



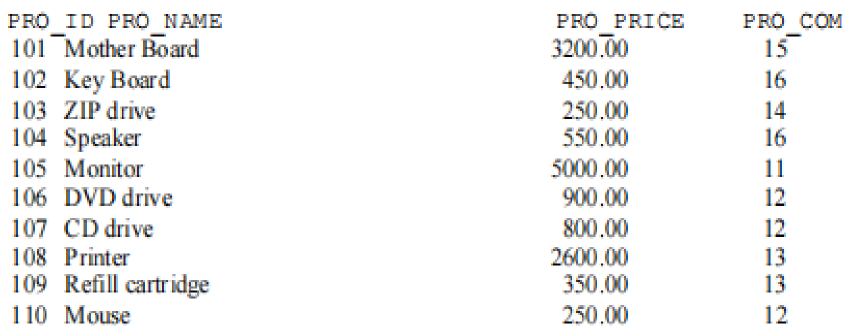
```
SELECT * FROM `item_mast` WHERE pro_price BETWEEN '200' AND '600';
```



pro_id	pro_name	pro_price	pro_com
102	Key Board	450	16
103	Zip Driver	250	14
104	Speaker	550	16
109	Refill cartridge	350	13
110	Mouse	250	12

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

Sample table: item_mast

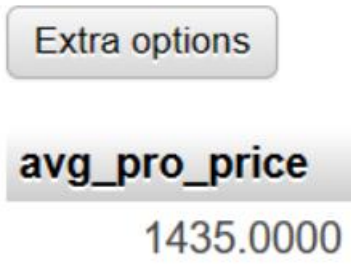


PRO_ID	PRO_NAME	PRO_PRICE	PRO_COM
101	Mother Board	3200.00	15
102	Key Board	450.00	16
103	ZIP drive	250.00	14
104	Speaker	550.00	16
105	Monitor	5000.00	11
106	DVD drive	900.00	12
107	CD drive	800.00	12
108	Printer	2600.00	13
109	Refill cartridge	350.00	13
110	Mouse	250.00	12

MODULE 5.2 : SQL Queries

➢ Solution:

```
SELECT AVG (pro_price)avg_pro_price FROM item_mast;
```

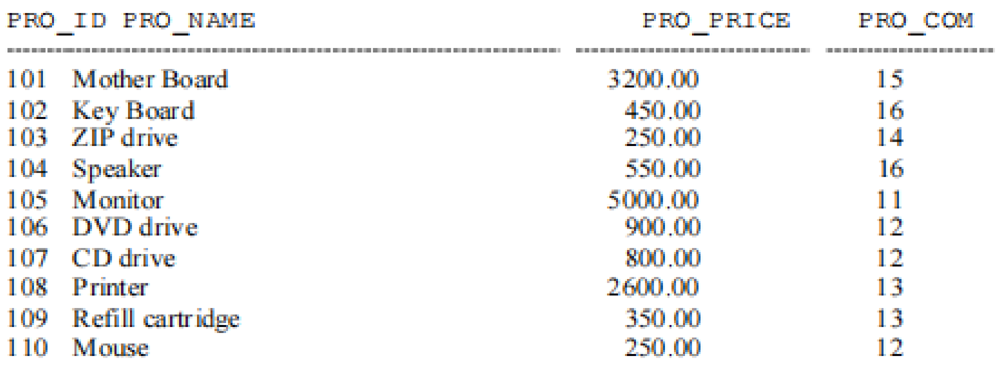


Extra options

avg_pro_price
1435.0000

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

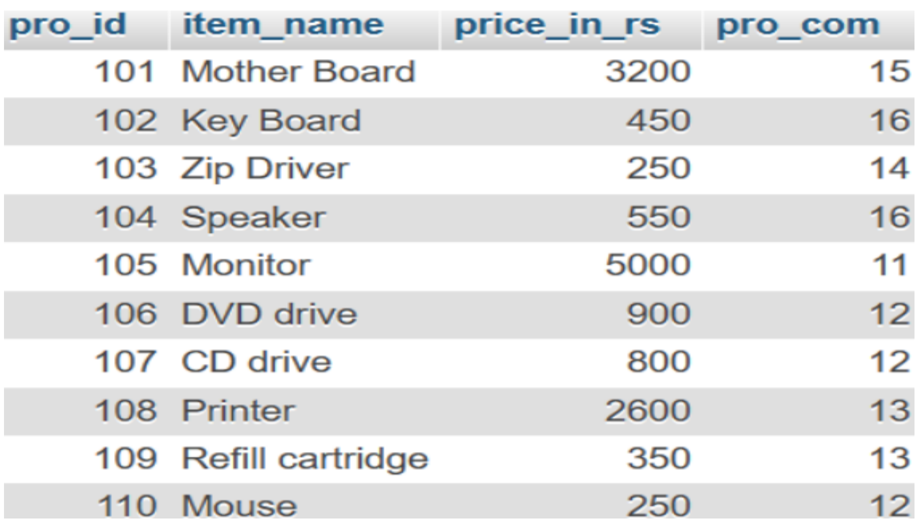
Sample table: item_mast



```
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
```

➢ Solution:

```
ALTER TABLE item_mast CHANGE pro_name item_name varchar(30);
ALTER TABLE item_mast CHANGE pro_price price_in_rs int;
```

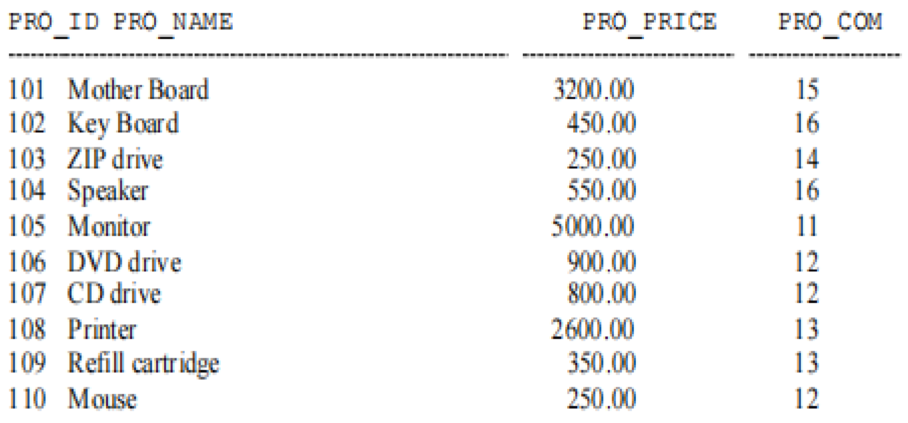


pro_id	item_name	price_in_rs	pro_com
101	Mother Board	3200	15
102	Key Board	450	16
103	Zip Driver	250	14
104	Speaker	550	16
105	Monitor	5000	11
106	DVD drive	900	12
107	CD drive	800	12
108	Printer	2600	13
109	Refill cartridge	350	13
110	Mouse	250	12

MODULE 5.2 : SQL Queries

22. 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.

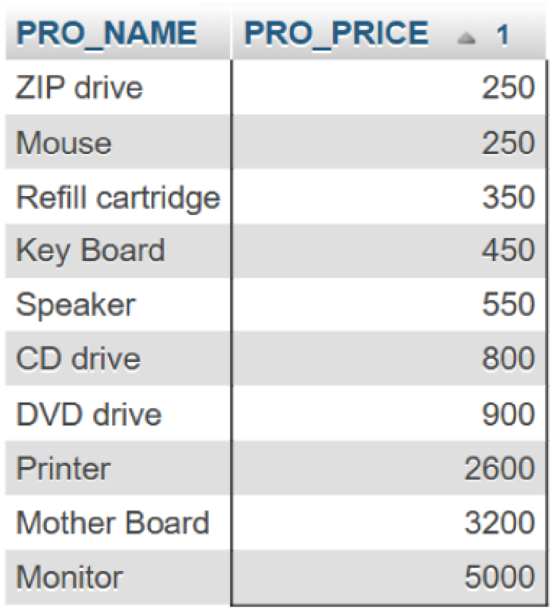
Sample table: item_mast



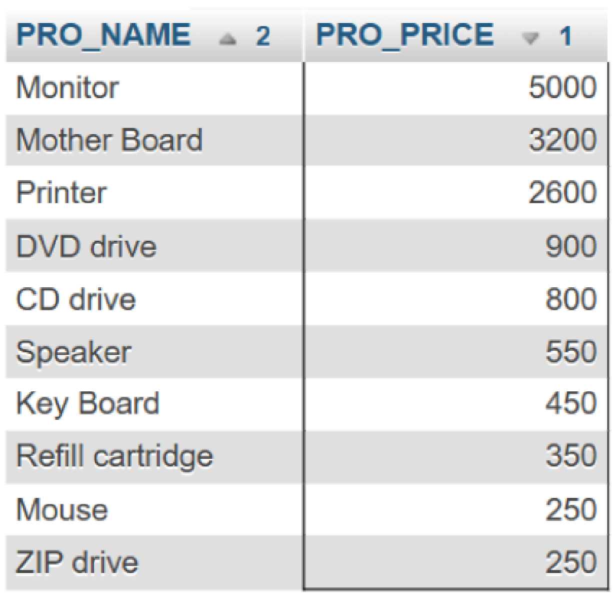
```
PRO_ID PRO_NAME                                   PRO_PRICE    PRO_COM
------------------------------------------------  -----------  --------
101    Mother Board                                 3200.00       15
102    Key Board                                     450.00       16
103    ZIP drive                                     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
```

➢ Solution:

```sql
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

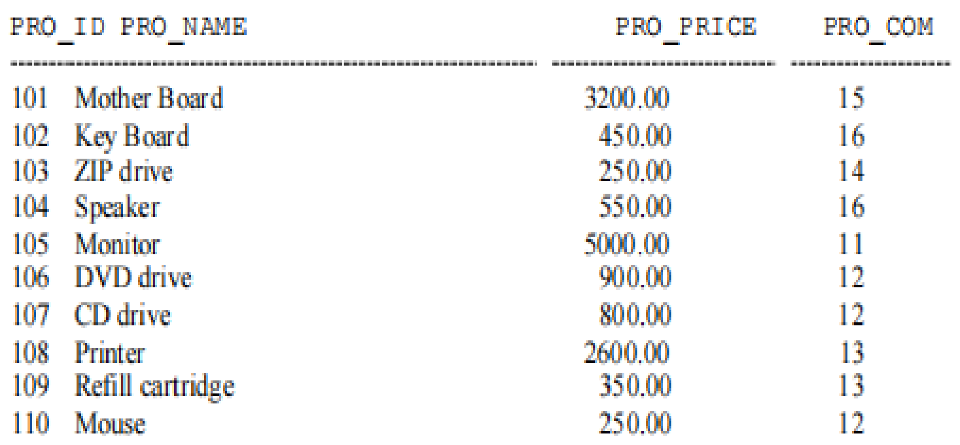


PRO_NAME 2	PRO_PRICE 1
Monitor	5000
Mother Board	3200
Printer	2600
DVD drive	900
CD drive	800
Speaker	550
Key Board	450
Refill cartridge	350
Mouse	250
ZIP drive	250

MODULE 5.2 : SQL Queries

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

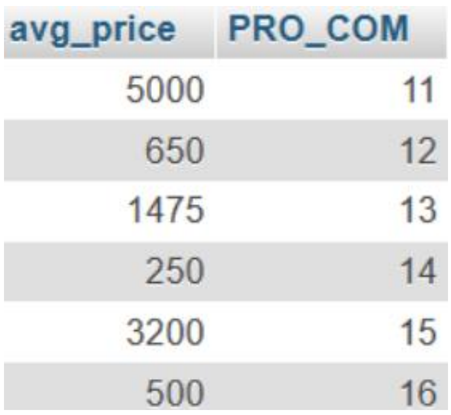
Sample table: item_mast



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

➢ Solution:

```
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