MODULE: 5 (Database)

Basics of Database

1. What do you understand By Database

   A database is an organized collection of data, stored and accessed electronically. Databases are used to store and manage large amounts of structured and unstructured data, and they can be used to support a wide range of activities, including data storage, data analysis, and data management.

2. What is Normalization?

   Normalization is a design of databases methodological method used in the to create a neat, structured, and structured table in which each table relates to just one subject or one-to-one correspondence. The objective is to extensively reduce data redundancy and dependency.

3. What is Difference between DBMS and RDBMS?

DBMS	RDBMS
Database management system.	Relation database management system.
Data stored is in file format .	Data Stored is in table format.
Individual access of data element.	Multiple data element is accessible together.
No connection between data.	Data in the form of a table are linked together.
No support for distributed database .	Support distributed database.
Data stored is a small quantity.	Data is Stored in large amount.
DBMS support a single user.	RDBMS supports multiple users.
The software and hardware requirements are low.	The software and hardware requirement are higher.
Example: - XML, Microsoft Assess.	Example: - Oracle, SQL, Server.

4. What is MF Cod Rule of RDBMS Systems?

   The MF Cod Rule of RDBMS Systems states that for a system to qualify as an RDBMS, it must be able to manage database entirely through the relational capabilities Rule 0 of the MF Cod Rules states that the system must qualify as relational, as a database, and as a management system. For a system to qualify as an RDBMS, that system must use its relational facilities exclusively to manage the database.

5. What do you understand By Data Redundancy?

   Data redundancy refers to the situation where the same pieces of data are stored in multiple places within a database or data storage system. This can happen intentionally or accidentally. Redundancy can be useful for data recovery in case of corruption or loss. In computer memory and storage, data redundancy allows for error correction.

6. What is DDL Interpreter?

DML Compiler: It processes the DML statements into low level instruction (machine language), so that they can be executed.

DDL Interpreter: It processes the DDL statements into a set of tables containing meta data (data about data).

7. What is DML Compiler in SQL?

The Data Manipulation Language, or DML for short, is the group of commands responsible for manipulating data in a database; this generally entails inserting, editing, or deleting rows in SQL tables**.**

8. What is SQL Key Constraints writing an Example of SQL Key Constraints.

Constraints are the rules that we can apply on the type of data in a table. That is, we can specify the limit on the type of data that can be stored in a particular column in a table using constraints.

The available constraints in SQL are:

- **NOT NULL:** This constraint tells that we cannot store a null value in a column. That is, if a column is specified as NOT NULL then we will not be able to store null in this particular column any more.

- **UNIQUE:** This constraint when specified with a column, tells that all the values in the column must be unique. That is, the values in any row of a column must not be repeated.

- **PRIMARY KEY:** A primary key is a field which can uniquely identify each row in a table. And this constraint is used to specify a field in a table as primary key.

- **FOREIGN KEY:** A Foreign key is a field which can uniquely identify each row in another table. And this constraint is used to specify a field as foreign key.

- **CHECK:** This constraint helps to validate the values of a column to meet a particular condition. That is, it helps to ensure that the value stored in a column meets a specific condition.

- **DEFAULT:** This constraint specifies a default value for the column when no value is specified by the user.

9. What is save Point? How to create a save Point write a Query?

A save point in SQL is a logical rollback point within a transaction. It allows you to specify a point in a transaction that you can roll back to without affecting the entire transaction.

To create a, save point, use the following syntax: 'SAVEPOINT savepoint_name'. You can then perform various SQL operations within the transaction.

To roll back to a specific save point, use 'ROLLBACK TO save_point_name'

10. What is trigger and how to create a Trigger in SQL?

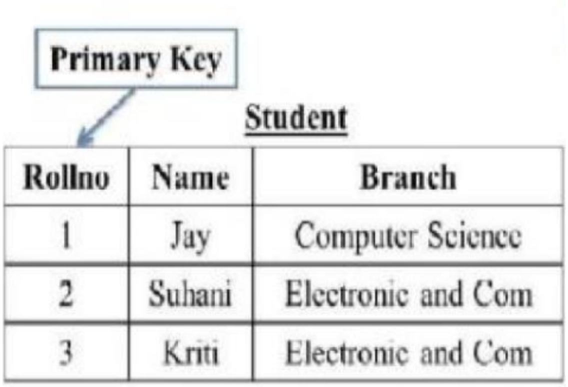
Trigger is a statement that a system executes automatically when there is any modification to the database. In a trigger, we first specify when the trigger is to be executed and then the action to be performed when the trigger executes. Triggers are used to specify certain integrity constraints and referential constraints that cannot be specified using the constraint mechanism of SQL.
Example –
Suppose, we are adding a tuple to the 'Donors' table that is some person has donated blood. So, we can design a trigger that will automatically add the value of donated blood to the 'Blood_record' table.

Types of Triggers –
We can define 6 types of triggers for each table:

1. **AFTER INSERT:** activated after data is inserted into the table.
2. **AFTER UPDATE:** activated after data in the table is modified.
3. **AFTER DELETE:** activated after data is deleted/removed from the table.
4. **BEFORE INSERT:** activated before data is inserted into the table.
5. **BEFORE UPDATE:** activated before data in the table is modified.
6. **BEFORE DELETE:** activated before data is deleted/removed from the table.

SQL Queries

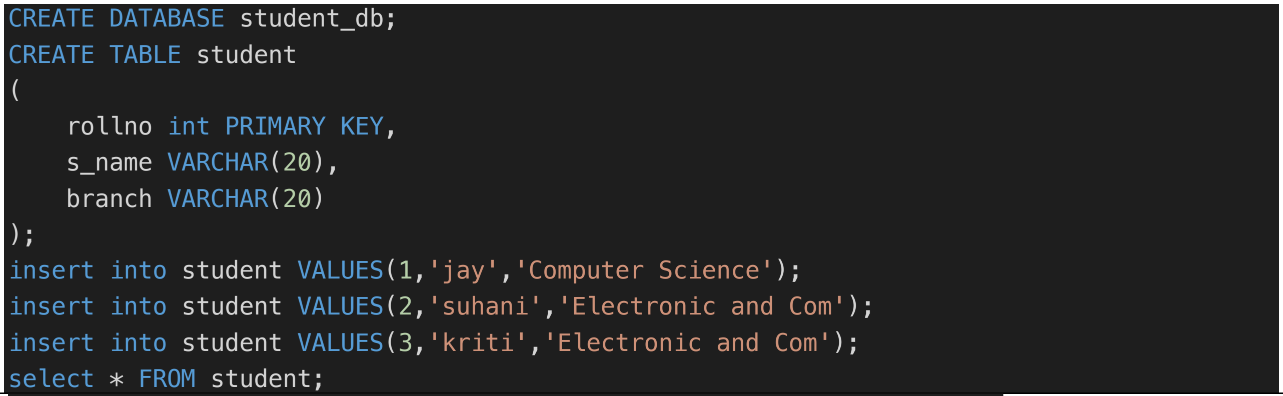
1. Create Table Name : Student and Exam



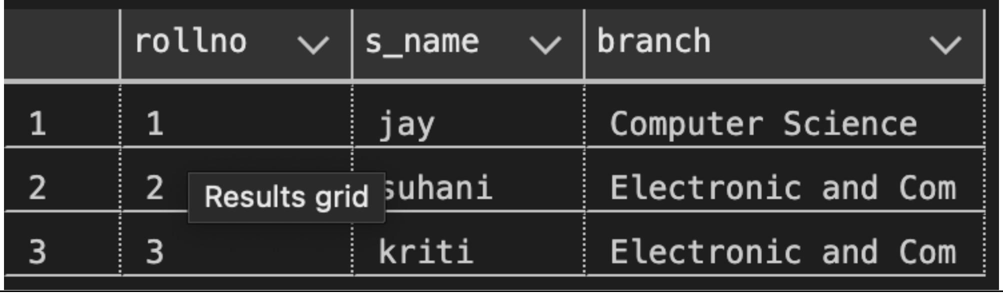
Primary Key

Student

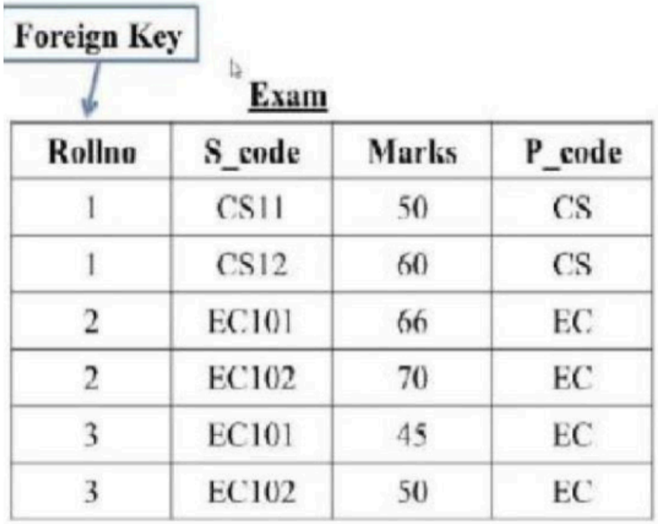
Rollno	Name	Branch
1	Jay	Computer Science
2	Suhani	Electronic and Com
3	Kriti	Electronic and Com



```
CREATE DATABASE student_db;
CREATE TABLE student
(
    rollno int PRIMARY KEY,
    s_name VARCHAR(20),
    branch VARCHAR(20)
);
insert into student VALUES(1,'jay','Computer Science');
insert into student VALUES(2,'suhani','Electronic and Com');
insert into student VALUES(3,'kriti','Electronic and Com');
select * FROM student;
```

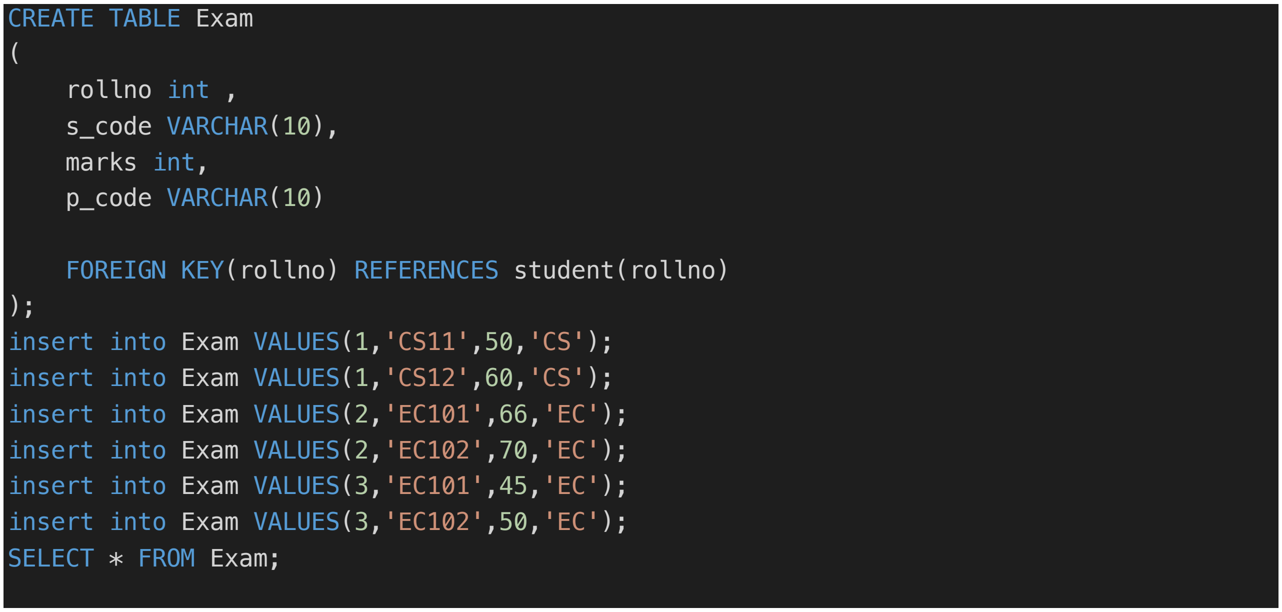


	rollno	s_name	branch
1	1	jay	Computer Science
2	2	suhani	Electronic and Com
3	3	kriti	Electronic and Com

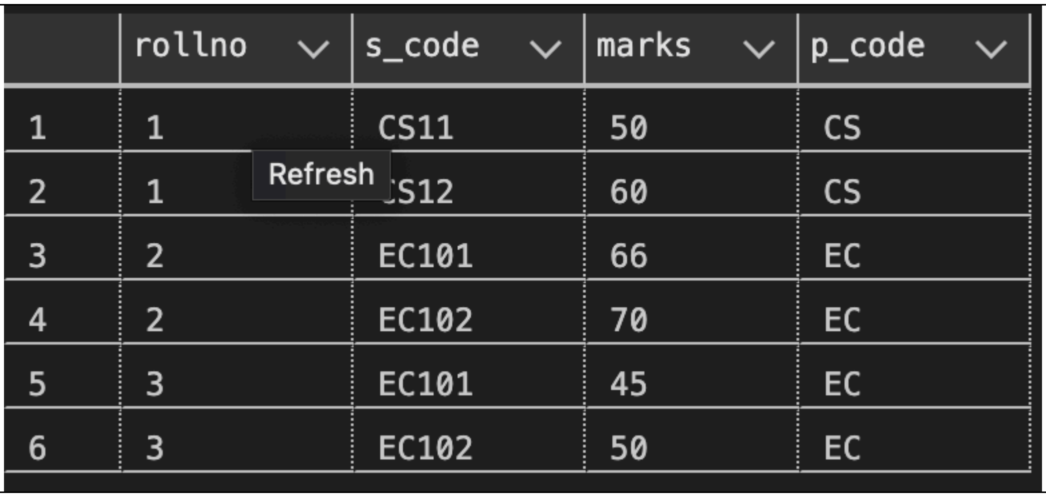

Foreign Key

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

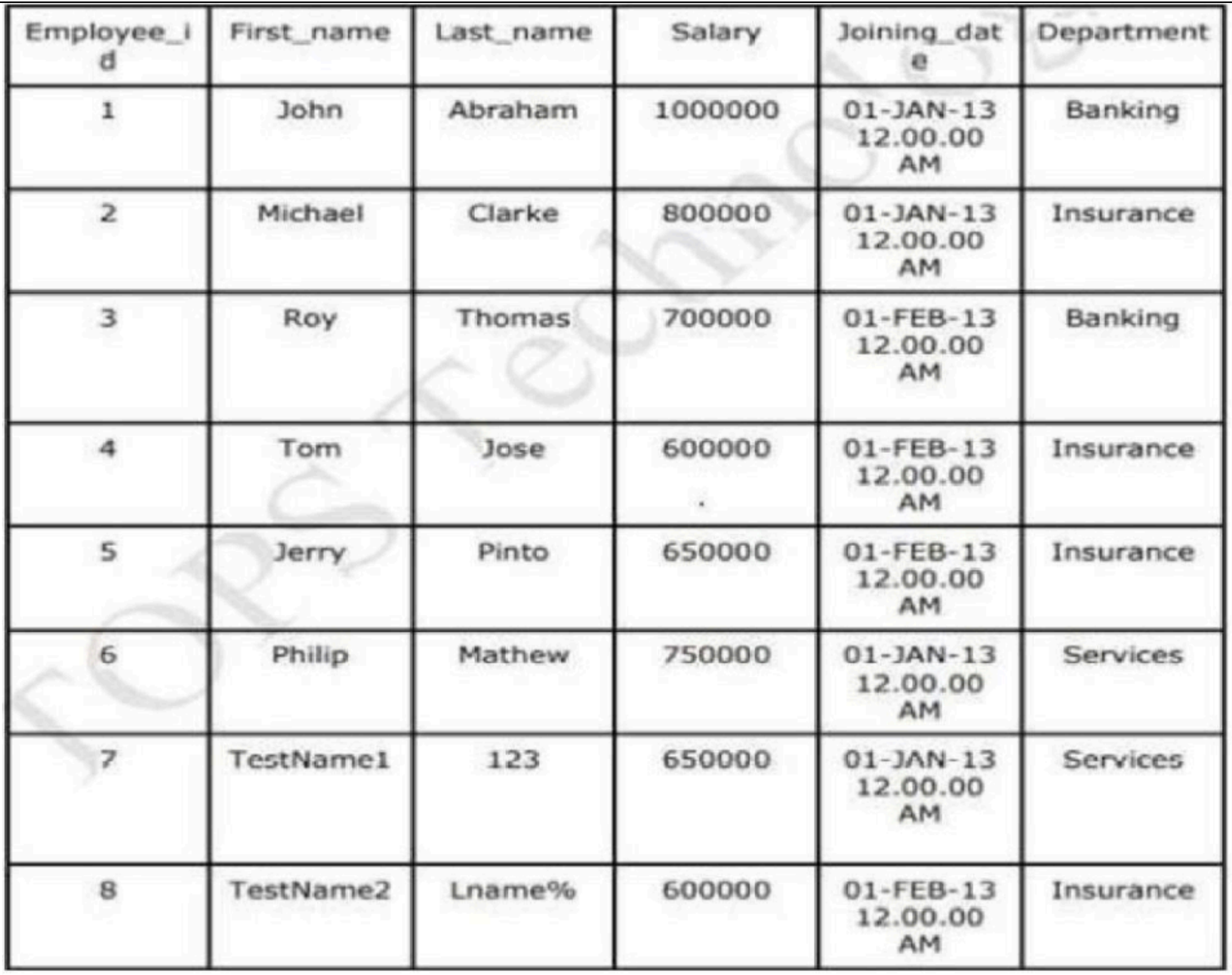


```
CREATE TABLE Exam
(
    rollno int ,
    s_code VARCHAR(10),
    marks int,
    p_code VARCHAR(10)

    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');
SELECT * FROM Exam;
```

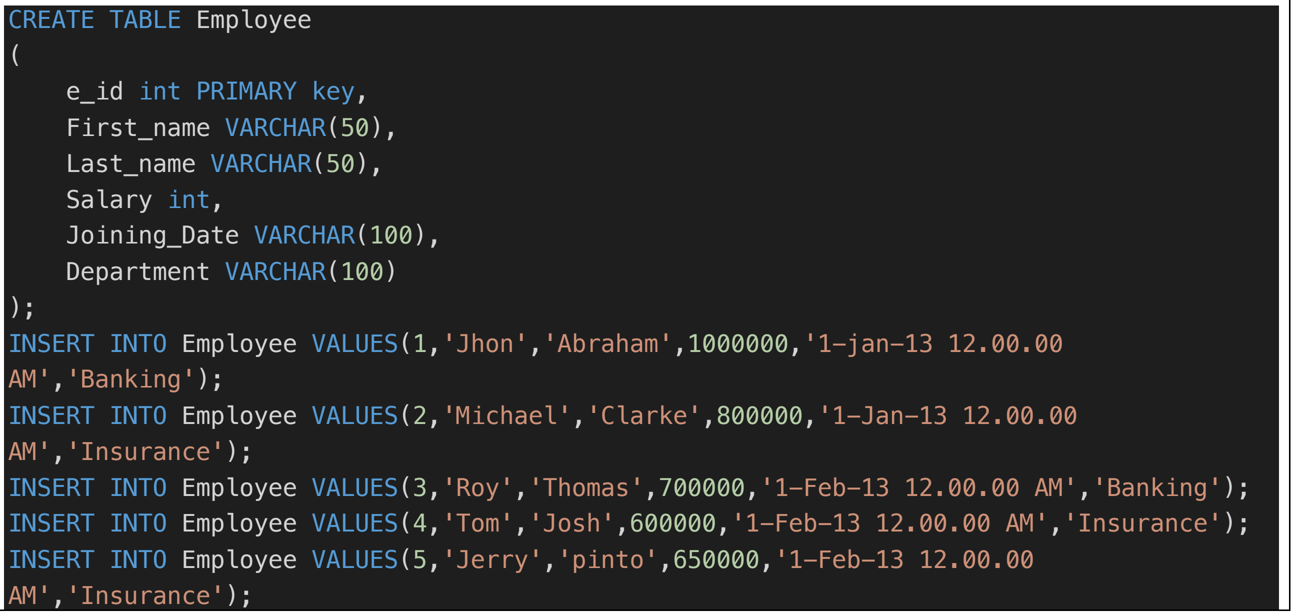


	rollno	s_code	marks	p_code
1	1	CS11	50	CS
2	1	CS12	60	CS
3	2	EC101	66	EC
4	2	EC102	70	EC
5	3	EC101	45	EC
6	3	EC102	50	EC

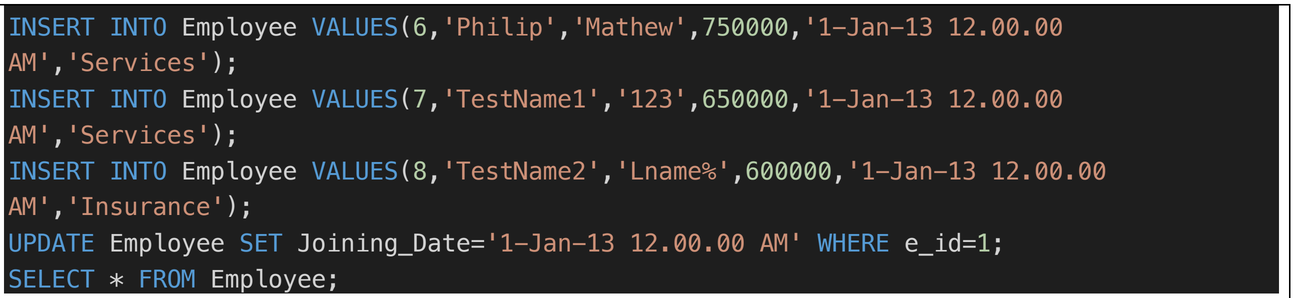
2. Create table given below: Employee and IncentiveTable



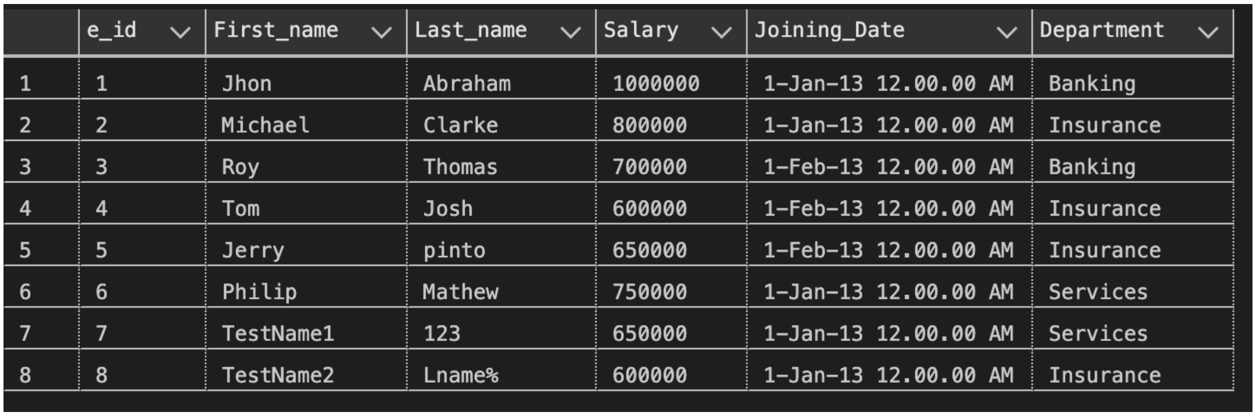
Employee_id	First_name	Last_name	Salary	Joining_date	Department
1	John	Abraham	1000000	01-JAN-13 12.00.00 AM	Banking
2	Michael	Clarke	800000	01-JAN-13 12.00.00 AM	Insurance
3	Roy	Thomas	700000	01-FEB-13 12.00.00 AM	Banking
4	Tom	Jose	600000	01-FEB-13 12.00.00 AM	Insurance
5	Jerry	Pinto	650000	01-FEB-13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



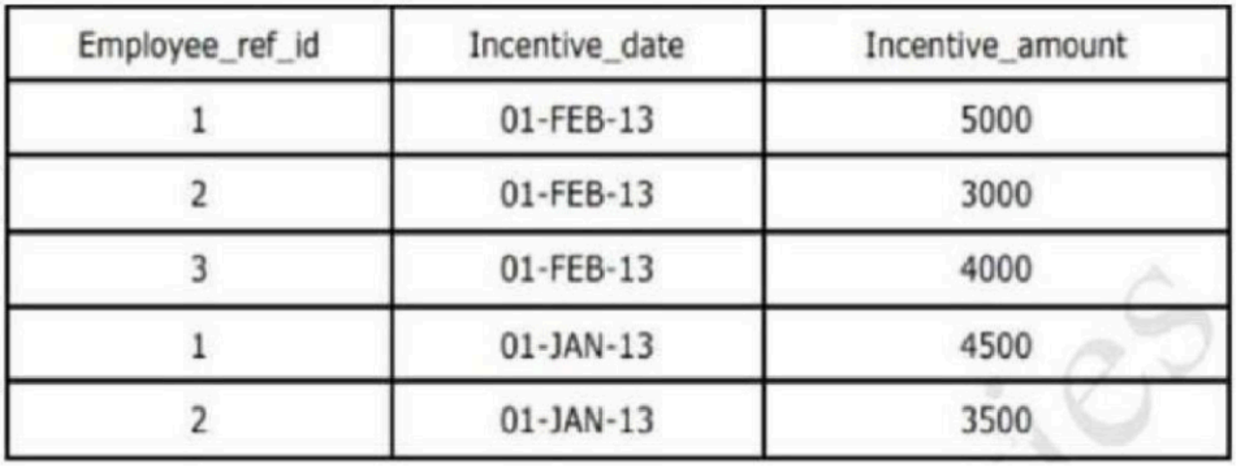
```
CREATE TABLE Employee
(
    e_id int PRIMARY key,
    First_name VARCHAR(50),
    Last_name VARCHAR(50),
    Salary int,
    Joining_Date VARCHAR(100),
    Department VARCHAR(100)
);
INSERT INTO Employee VALUES(1,'Jhon','Abraham',1000000,'1-jan-13 12.00.00
AM','Banking');
INSERT INTO Employee VALUES(2,'Michael','Clarke',800000,'1-Jan-13 12.00.00
AM','Insurance');
INSERT INTO Employee VALUES(3,'Roy','Thomas',700000,'1-Feb-13 12.00.00 AM','Banking');
INSERT INTO Employee VALUES(4,'Tom','Josh',600000,'1-Feb-13 12.00.00 AM','Insurance');
INSERT INTO Employee VALUES(5,'Jerry','pinto',650000,'1-Feb-13 12.00.00
AM','Insurance');
```



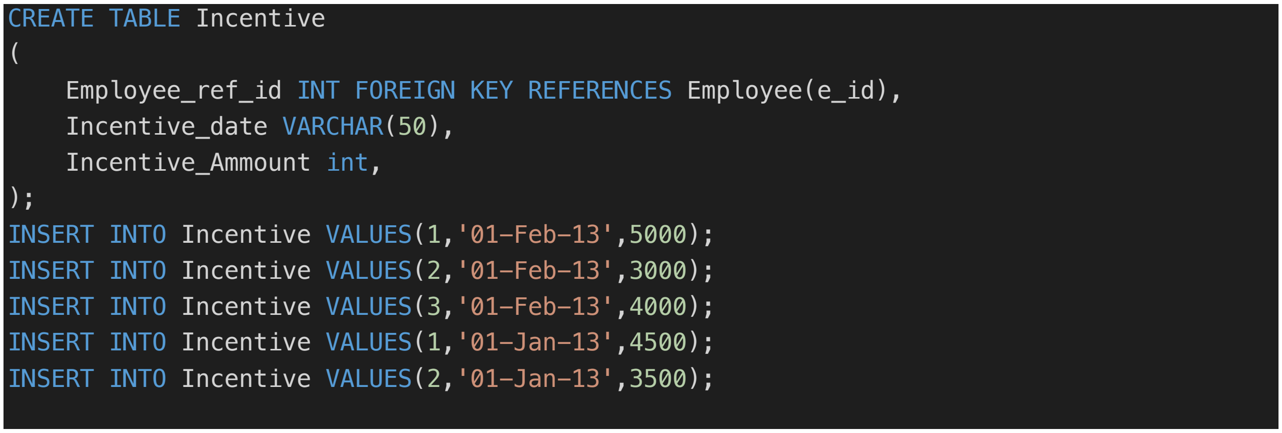
```
INSERT INTO Employee VALUES(6,'Philip','Mathew',750000,'1-Jan-13 12.00.00
AM','Services');
INSERT INTO Employee VALUES(7,'TestName1','123',650000,'1-Jan-13 12.00.00
AM','Services');
INSERT INTO Employee VALUES(8,'TestName2','Lname%',600000,'1-Jan-13 12.00.00
AM','Insurance');
UPDATE Employee SET Joining_Date='1-Jan-13 12.00.00 AM' WHERE e_id=1;
SELECT * FROM Employee;
```



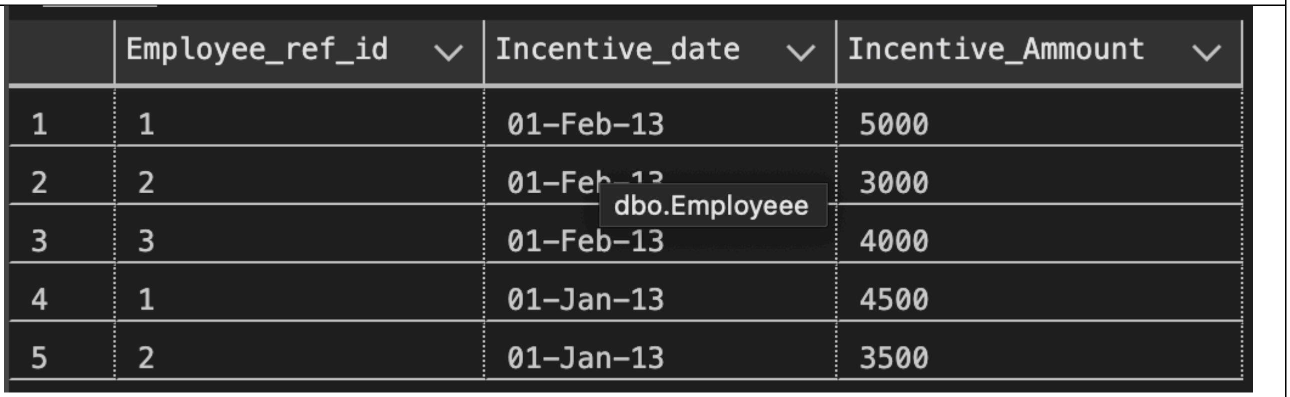
	e_id	First_name	Last_name	Salary	Joining_Date	Department
1	1	Jhon	Abraham	1000000	1-Jan-13 12.00.00 AM	Banking
2	2	Michael	Clarke	800000	1-Jan-13 12.00.00 AM	Insurance
3	3	Roy	Thomas	700000	1-Feb-13 12.00.00 AM	Banking
4	4	Tom	Josh	600000	1-Feb-13 12.00.00 AM	Insurance
5	5	Jerry	pinto	650000	1-Feb-13 12.00.00 AM	Insurance
6	6	Philip	Mathew	750000	1-Jan-13 12.00.00 AM	Services
7	7	TestName1	123	650000	1-Jan-13 12.00.00 AM	Services
8	8	TestName2	Lname%	600000	1-Jan-13 12.00.00 AM	Insurance



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

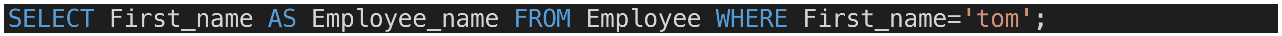


```sql
CREATE TABLE Incentive
(
    Employee_ref_id INT FOREIGN KEY REFERENCES Employee(e_id),
    Incentive_date VARCHAR(50),
    Incentive_Ammount int,
);
INSERT INTO Incentive VALUES(1,'01-Feb-13',5000);
INSERT INTO Incentive VALUES(2,'01-Feb-13',3000);
INSERT INTO Incentive VALUES(3,'01-Feb-13',4000);
INSERT INTO Incentive VALUES(1,'01-Jan-13',4500);
INSERT INTO Incentive VALUES(2,'01-Jan-13',3500);
```

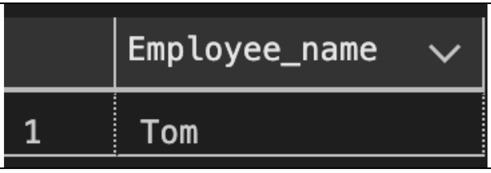


	Employee_ref_id	Incentive_date	Incentive_Ammount
1	1	01-Feb-13	5000
2	2	01-Feb-13	3000
3	3	01-Feb-13	4000
4	1	01-Jan-13	4500
5	2	01-Jan-13	3500

3. . Get First_Name from employee table using Tom name “Employee Name”

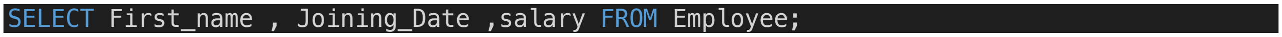


```sql
SELECT First_name AS Employee_name FROM Employee WHERE First_name='tom';
```

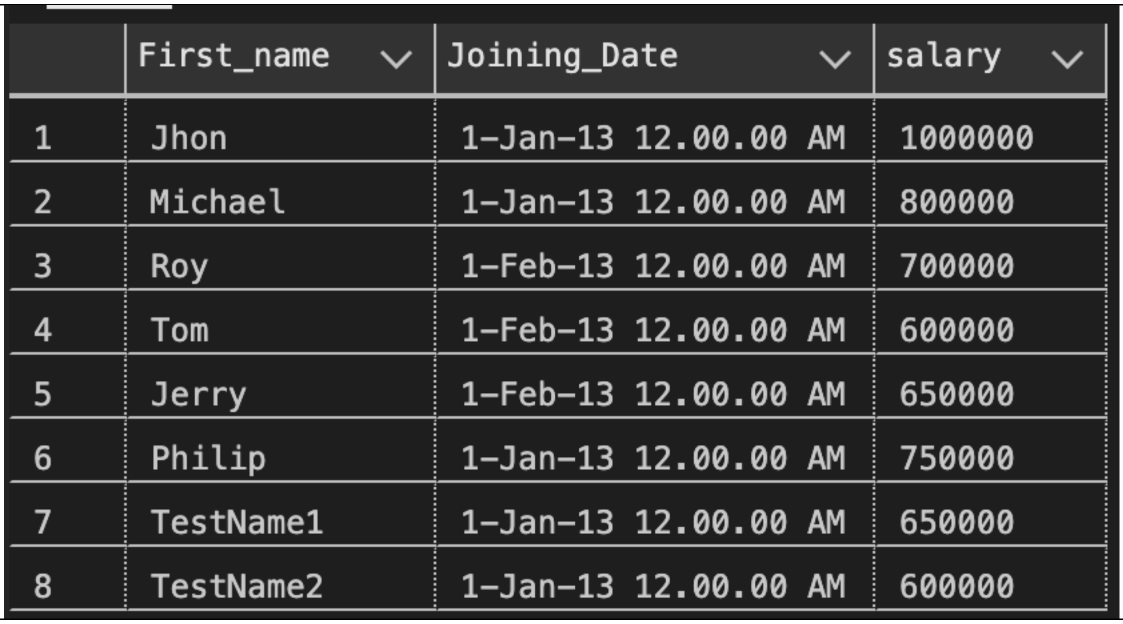


	Employee_name
1	Tom

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

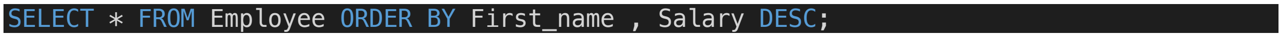


```sql
SELECT First_name , Joining_Date ,salary FROM Employee;
```

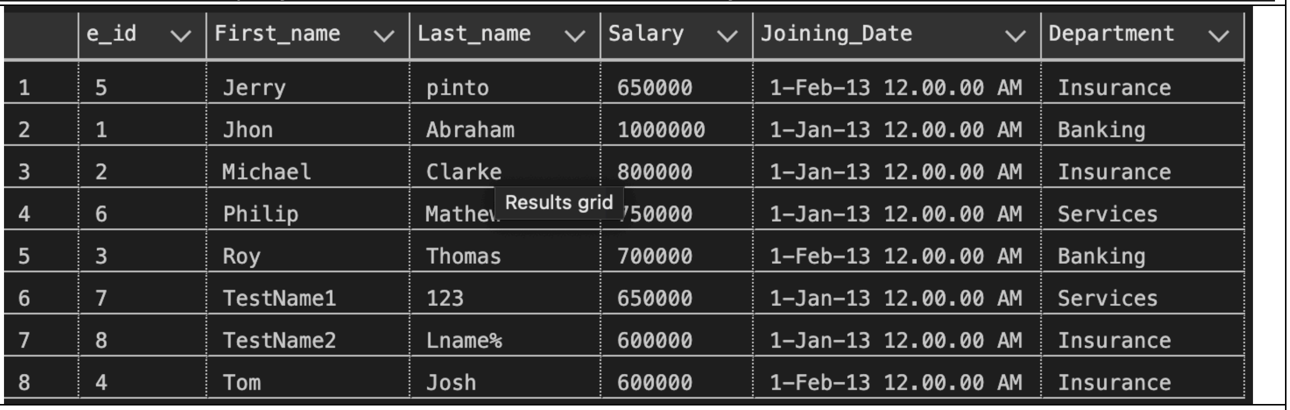


	First_name	Joining_Date	salary
1	Jhon	1-Jan-13 12.00.00 AM	1000000
2	Michael	1-Jan-13 12.00.00 AM	800000
3	Roy	1-Feb-13 12.00.00 AM	700000
4	Tom	1-Feb-13 12.00.00 AM	600000
5	Jerry	1-Feb-13 12.00.00 AM	650000
6	Philip	1-Jan-13 12.00.00 AM	750000
7	TestName1	1-Jan-13 12.00.00 AM	650000
8	TestName2	1-Jan-13 12.00.00 AM	600000

5. Get all employee details from the employee table order by First_Name

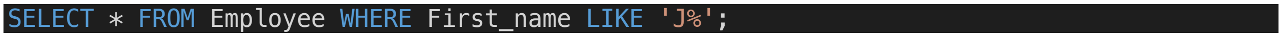


```sql
SELECT * FROM Employee ORDER BY First_name , Salary DESC;
```

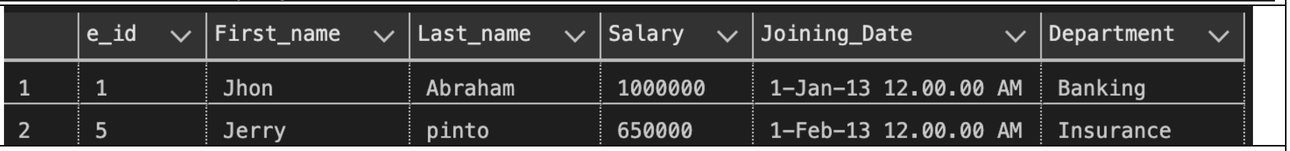


	e_id	First_name	Last_name	Salary	Joining_Date	Department
1	5	Jerry	pinto	650000	1-Feb-13 12.00.00 AM	Insurance
2	1	Jhon	Abraham	1000000	1-Jan-13 12.00.00 AM	Banking
3	2	Michael	Clarke	800000	1-Jan-13 12.00.00 AM	Insurance
4	6	Philip	Mathew	750000	1-Jan-13 12.00.00 AM	Services
5	3	Roy	Thomas	700000	1-Feb-13 12.00.00 AM	Banking
6	7	TestName1	123	650000	1-Jan-13 12.00.00 AM	Services
7	8	TestName2	Lname%	600000	1-Jan-13 12.00.00 AM	Insurance
8	4	Tom	Josh	600000	1-Feb-13 12.00.00 AM	Insurance

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

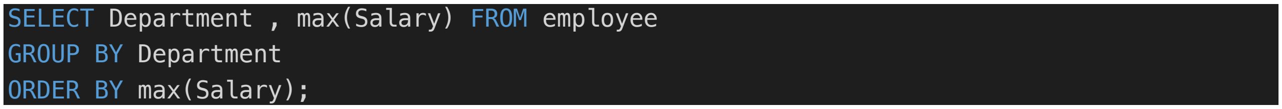


```sql
SELECT * FROM Employee WHERE First_name LIKE 'J%';
```

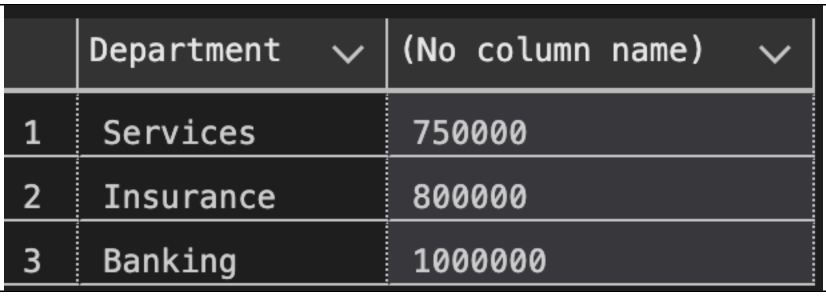


	e_id	First_name	Last_name	Salary	Joining_Date	Department
1	1	Jhon	Abraham	1000000	1-Jan-13 12.00.00 AM	Banking
2	5	Jerry	pinto	650000	1-Feb-13 12.00.00 AM	Insurance

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

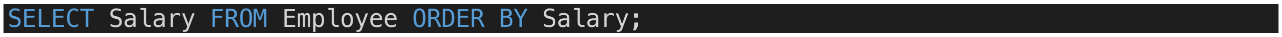


```sql
SELECT Department , max(Salary) FROM employee
GROUP BY Department
ORDER BY max(Salary);
```

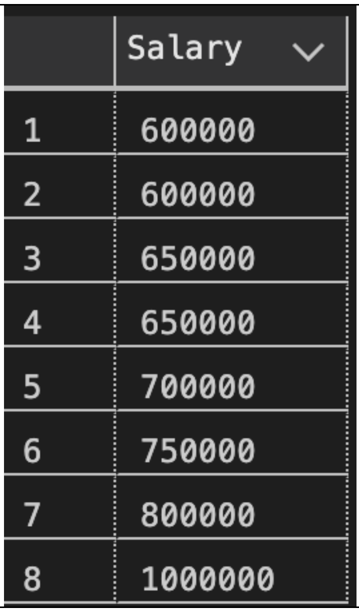


	Department	(No column name)
1	Services	750000
2	Insurance	800000
3	Banking	1000000

8. Salary ascending ?

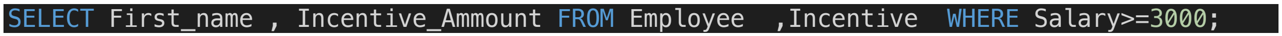


```
SELECT Salary FROM Employee ORDER BY Salary;
```

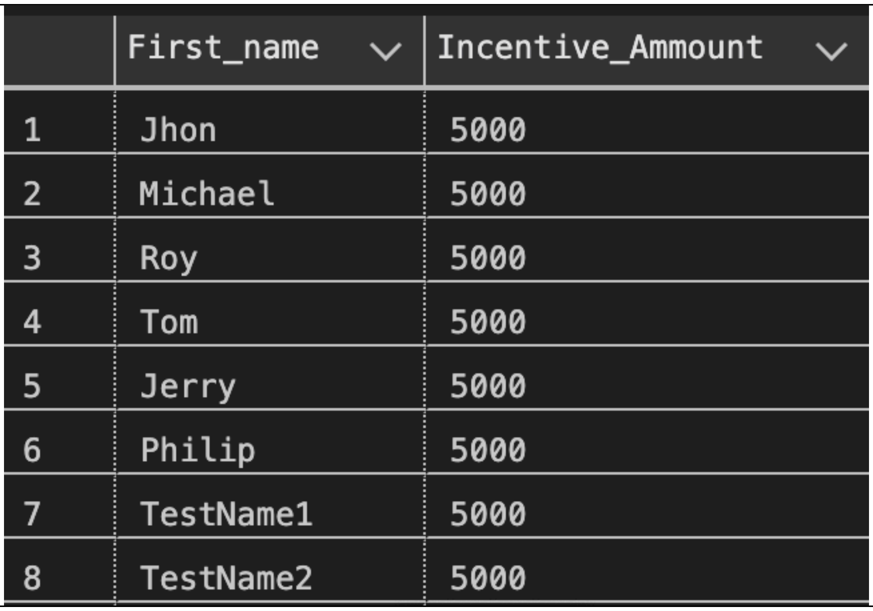


	Salary
1	600000
2	600000
3	650000
4	650000
5	700000
6	750000
7	800000
8	1000000

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

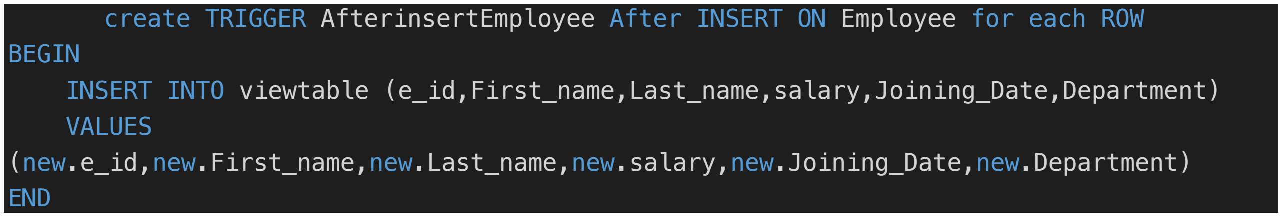


```
SELECT First_name , Incentive_Ammount FROM Employee  ,Incentive  WHERE Salary>=3000;
```



	First_name	Incentive_Ammount
1	Jhon	5000
2	Michael	5000
3	Roy	5000
4	Tom	5000
5	Jerry	5000
6	Philip	5000
7	TestName1	5000
8	TestName2	5000

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



```
       create TRIGGER AfterinsertEmployee After INSERT ON Employee for each ROW
BEGIN
    INSERT INTO viewtable (e_id,First_name,Last_name,salary,Joining_Date,Department)
    VALUES
(new.e_id,new.First_name,new.Last_name,new.salary,new.Joining_Date,new.Department)
END
```

11. Create table given below: Salesperson and Customer

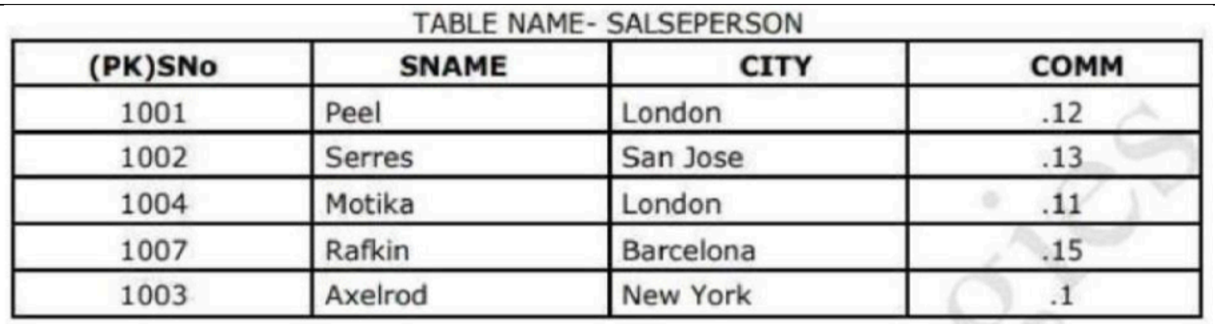
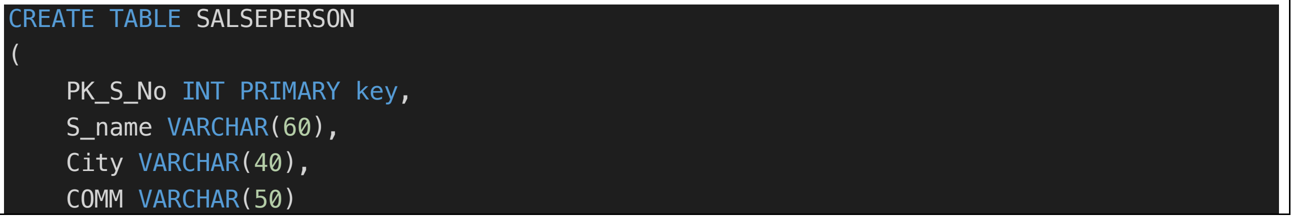
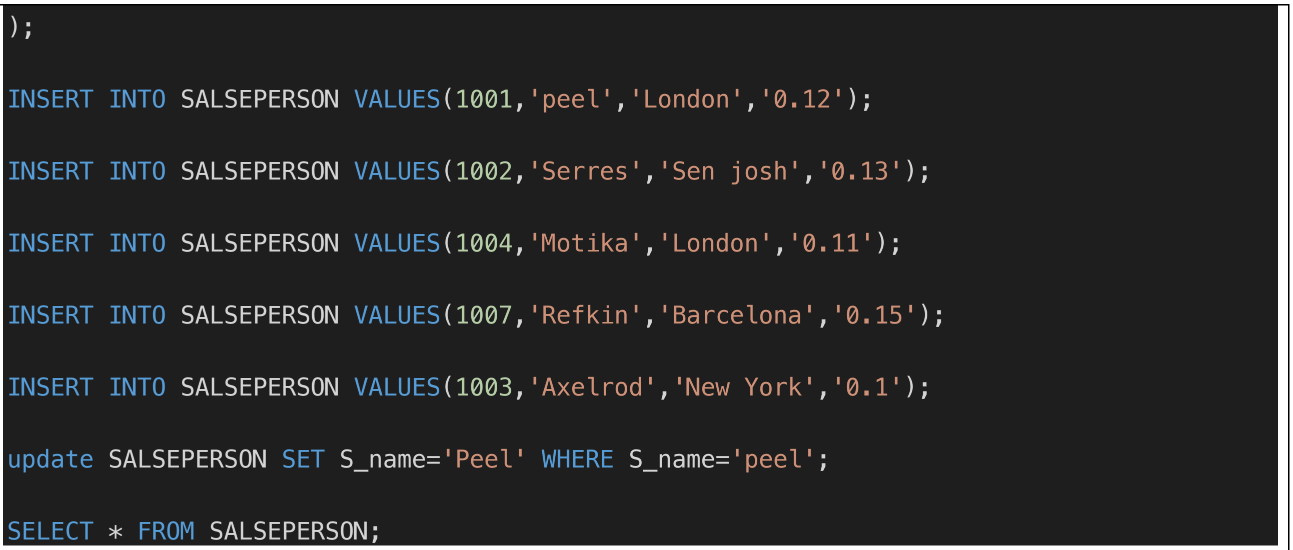


TABLE NAME- SALSEPERSON

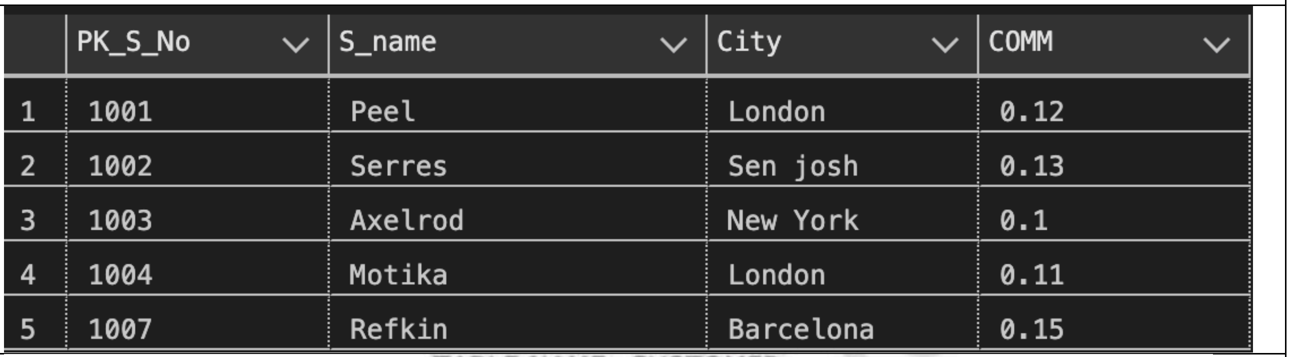
(PK)SNo	SNAME	CITY	COMM
1001	Peel	London	.12
1002	Serres	San Jose	.13
1004	Motika	London	.11
1007	Rafkin	Barcelona	.15
1003	Axelrod	New York	.1



```
CREATE TABLE SALSEPERSON
(
    PK_S_No INT PRIMARY key,
    S_name VARCHAR(60),
    City VARCHAR(40),
    COMM VARCHAR(50)
```



```
);

INSERT INTO SALSEPERSON VALUES(1001,'peel','London','0.12');

INSERT INTO SALSEPERSON VALUES(1002,'Serres','Sen josh','0.13');

INSERT INTO SALSEPERSON VALUES(1004,'Motika','London','0.11');

INSERT INTO SALSEPERSON VALUES(1007,'Refkin','Barcelona','0.15');

INSERT INTO SALSEPERSON VALUES(1003,'Axelrod','New York','0.1');

update SALSEPERSON SET S_name='Peel' WHERE S_name='peel';

SELECT * FROM SALSEPERSON;
```



	PK_S_No	S_name	City	COMM
1	1001	Peel	London	0.12
2	1002	Serres	Sen josh	0.13
3	1003	Axelrod	New York	0.1
4	1004	Motika	London	0.11
5	1007	Refkin	Barcelona	0.15

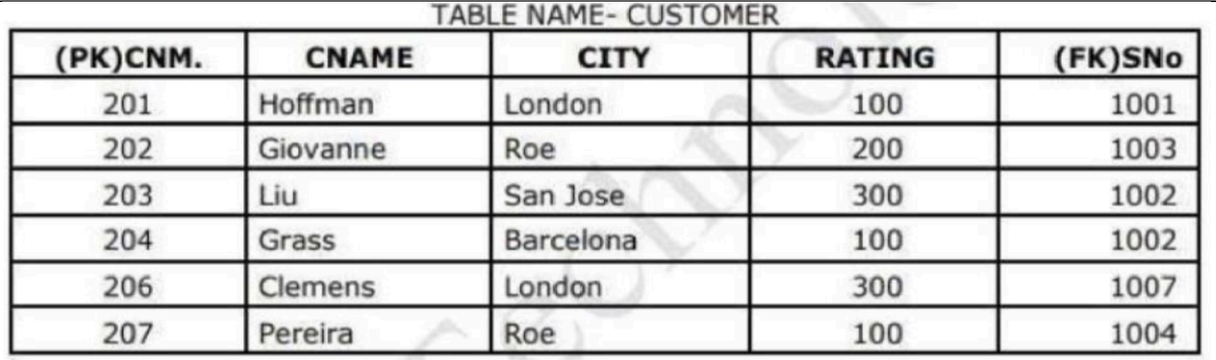
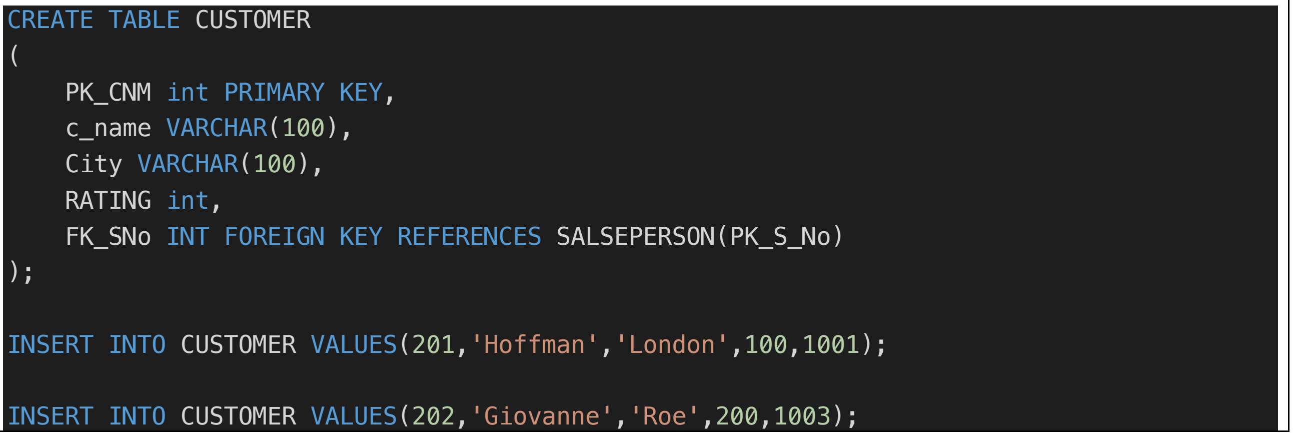
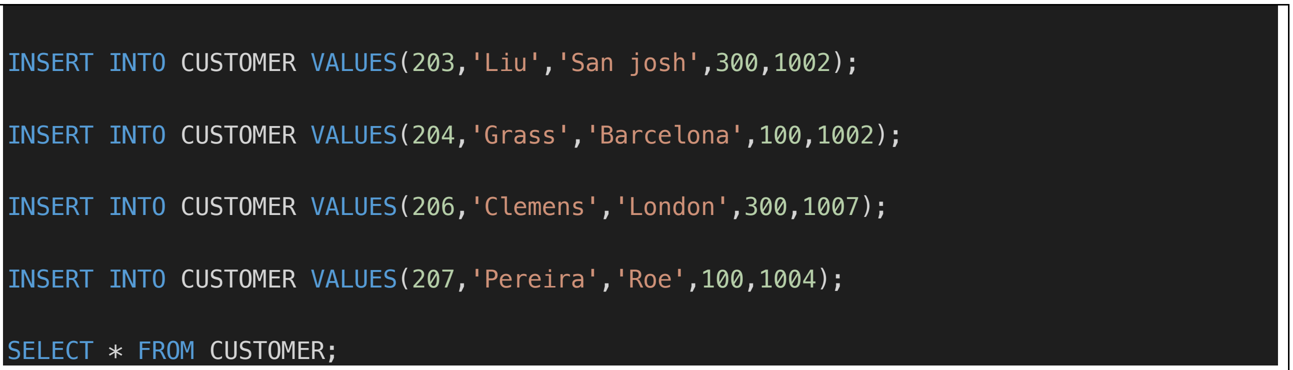


TABLE NAME- CUSTOMER

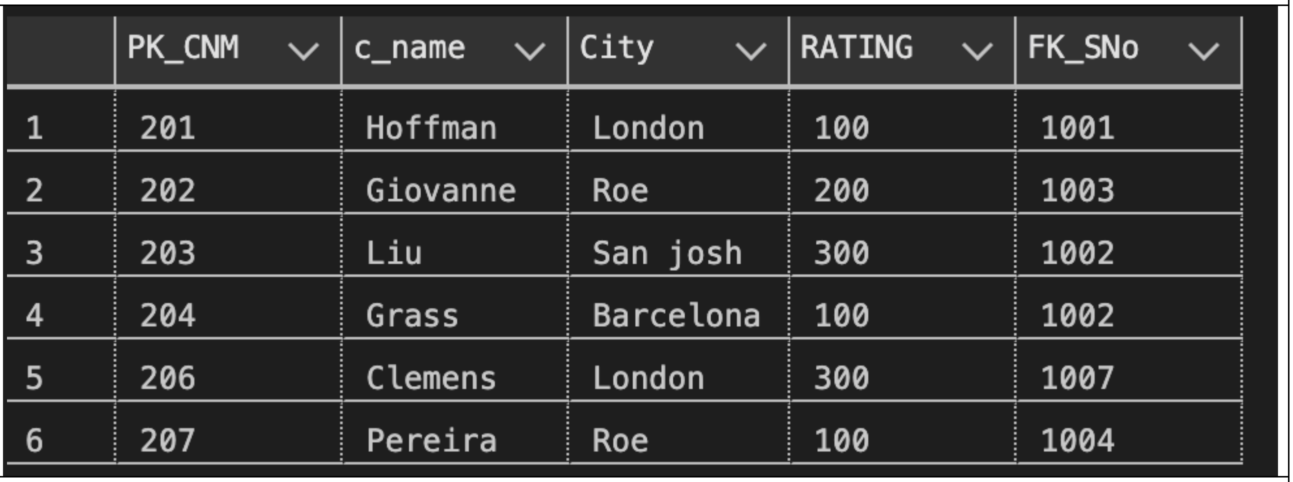
(PK)CNM.	CNAME	CITY	RATING	(FK)SNo
201	Hoffman	London	100	1001
202	Giovanne	Roe	200	1003
203	Liu	San Jose	300	1002
204	Grass	Barcelona	100	1002
206	Clemens	London	300	1007
207	Pereira	Roe	100	1004



```
CREATE TABLE CUSTOMER
(
    PK_CNM int PRIMARY KEY,
    c_name VARCHAR(100),
    City VARCHAR(100),
    RATING int,
    FK_SNo INT FOREIGN KEY REFERENCES SALSEPERSON(PK_S_No)
);

INSERT INTO CUSTOMER VALUES(201,'Hoffman','London',100,1001);

INSERT INTO CUSTOMER VALUES(202,'Giovanne','Roe',200,1003);
```



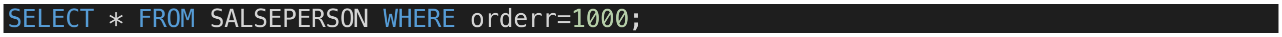
```
INSERT INTO CUSTOMER VALUES(203,'Liu','San josh',300,1002);

INSERT INTO CUSTOMER VALUES(204,'Grass','Barcelona',100,1002);

INSERT INTO CUSTOMER VALUES(206,'Clemens','London',300,1007);

INSERT INTO CUSTOMER VALUES(207,'Pereira','Roe',100,1004);

SELECT * FROM CUSTOMER;
```



	PK_CNM	c_name	City	RATING	FK_SNo
1	201	Hoffman	London	100	1001
2	202	Giovanne	Roe	200	1003
3	203	Liu	San josh	300	1002
4	204	Grass	Barcelona	100	1002
5	206	Clemens	London	300	1007
6	207	Pereira	Roe	100	1004

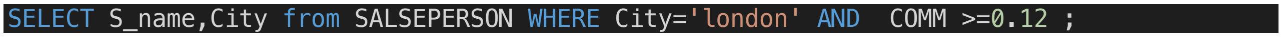
12. Retrieve the below data from above table

13 .All orders for more than $1000.

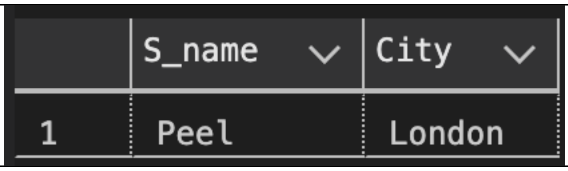


```
SELECT * FROM SALSEPERSON WHERE orderr=1000;
```

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

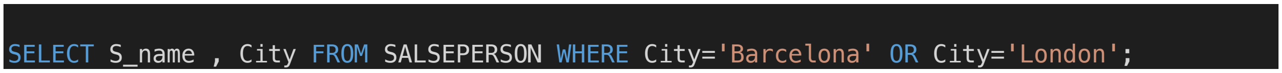


```
SELECT S_name,City from SALSEPERSON WHERE City='london' AND  COMM >=0.12 ;
```

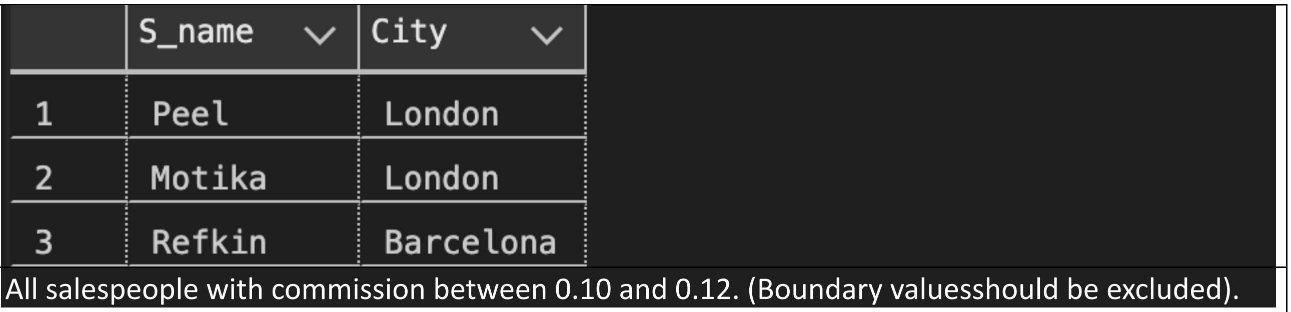


	S_name	City
1	Peel	London

15. All salespeople either in Barcelona or in London



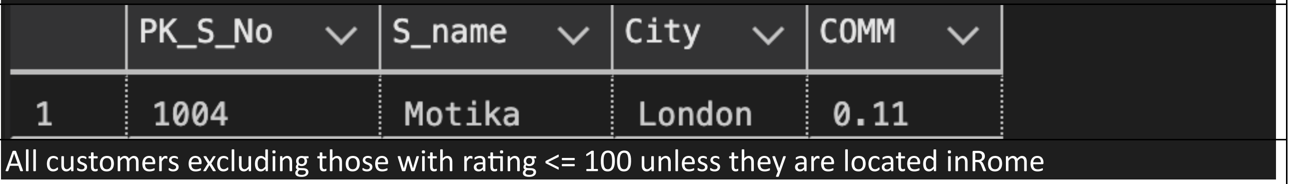
```
SELECT S_name , City FROM SALSEPERSON WHERE City='Barcelona' OR City='London';
```



	S_name	City
1	Peel	London
2	Motika	London
3	Refkin	Barcelona

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

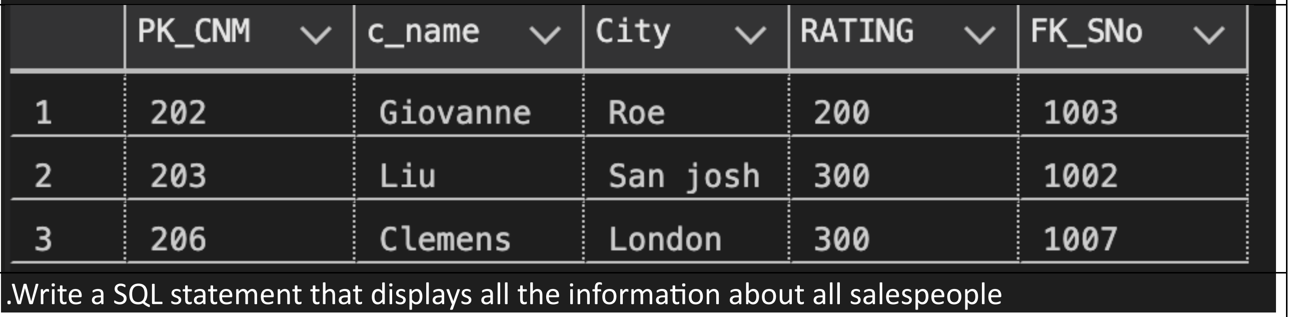
```
SELECT * FROM SALSEPERSON WHERE COMM >0.10 and COMM < 0.12;
```



	PK_S_No	S_name	City	COMM
1	1004	Motika	London	0.11

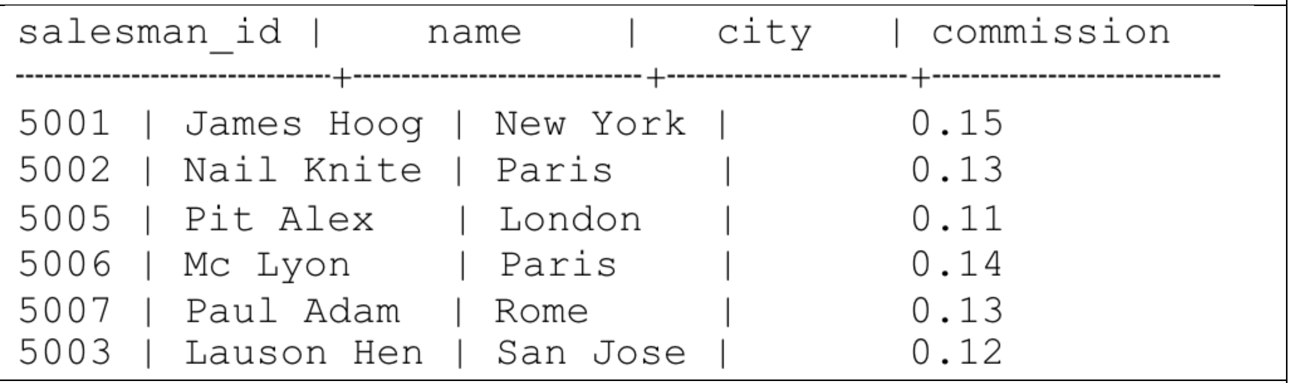
17. All customers excluding those with rating <= 100 unless they are located inRome

```
SELECT * FROM CUSTOMER WHERE RATING > 100 OR (RATING <= 100  AND City='ROME' );
```



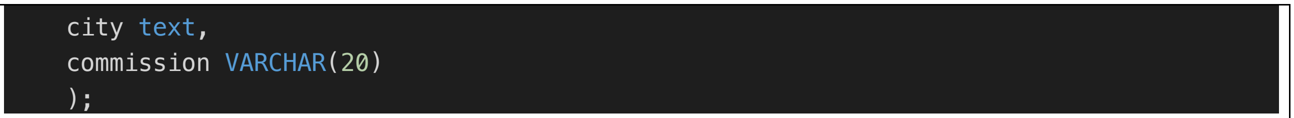
	PK_CNM	c_name	City	RATING	FK_SNo
1	202	Giovanne	Roe	200	1003
2	203	Liu	San josh	300	1002
3	206	Clemens	London	300	1007

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

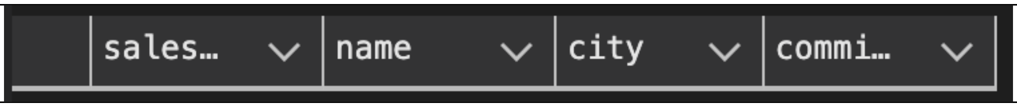


```
salesman_id |     name     |    city    | commission
------------+--------------+------------+------------
5001 | James Hoog | New York |         0.15
5002 | Nail Knite | Paris    |         0.13
5005 | Pit Alex   | London   |         0.11
5006 | Mc Lyon    | Paris    |         0.14
5007 | Paul Adam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

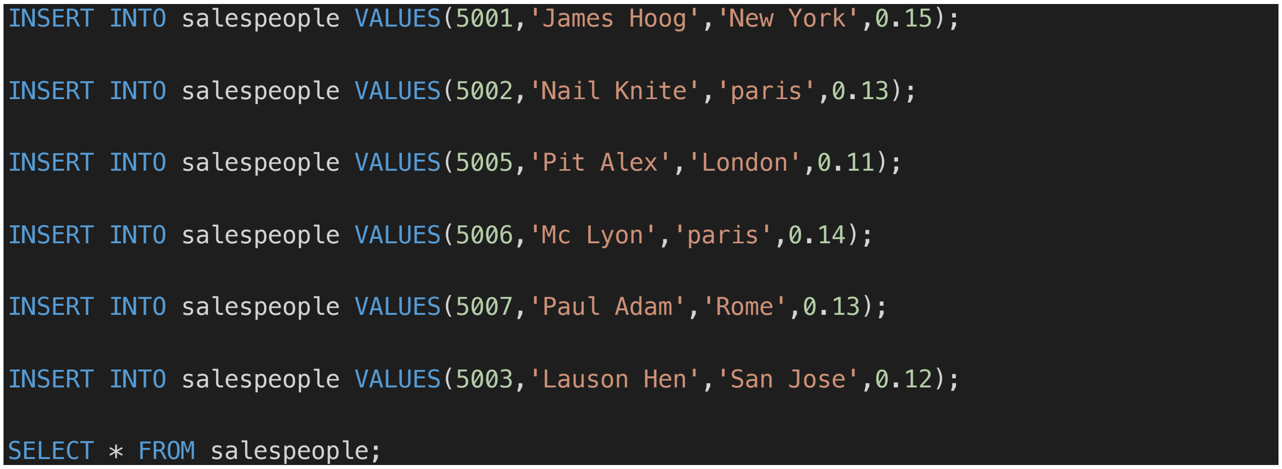
```
CREATE TABLE salespeople
(
    salesman_id int,
    name varchar(30),
```



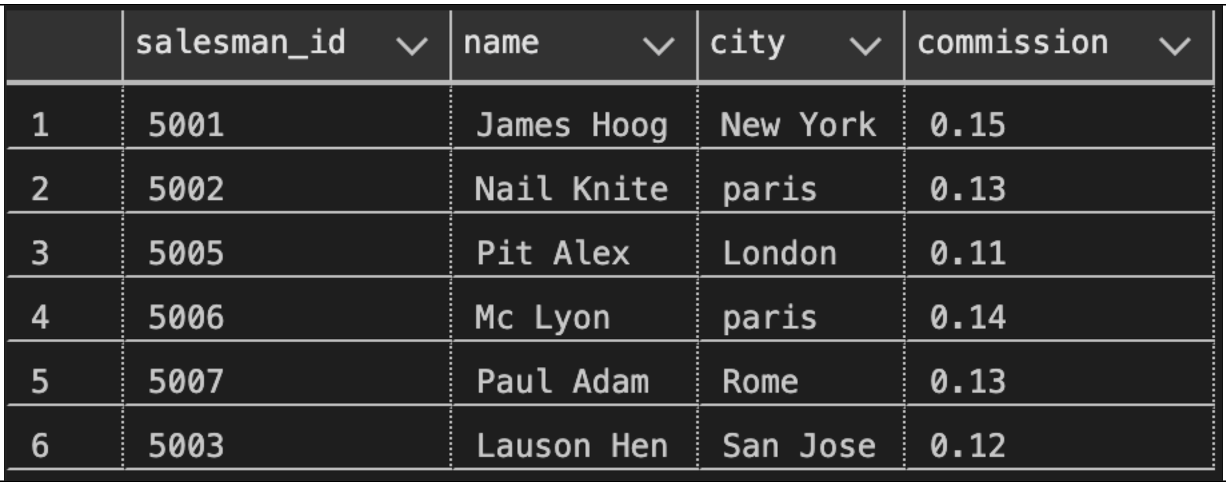
```
    city text,
    commission VARCHAR(20)
    );
```



	sales…	name	city	commi…



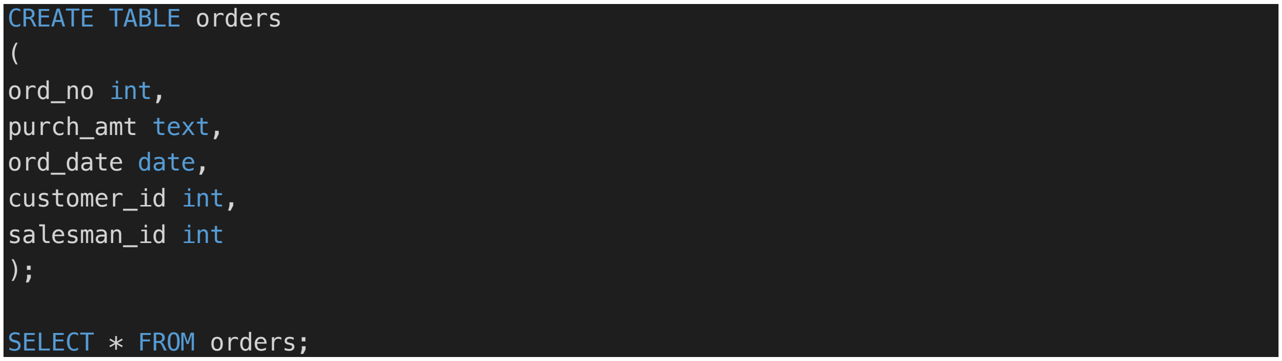
```
INSERT INTO salespeople VALUES(5001,'James Hoog','New York',0.15);

INSERT INTO salespeople VALUES(5002,'Nail Knite','paris',0.13);

INSERT INTO salespeople VALUES(5005,'Pit Alex','London',0.11);

INSERT INTO salespeople VALUES(5006,'Mc Lyon','paris',0.14);

INSERT INTO salespeople VALUES(5007,'Paul Adam','Rome',0.13);

INSERT INTO salespeople VALUES(5003,'Lauson Hen','San Jose',0.12);

SELECT * FROM salespeople;
```



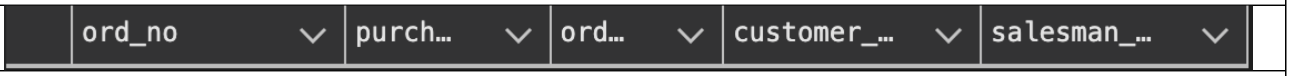
	salesman_id	name	city	commission
1	5001	James Hoog	New York	0.15
2	5002	Nail Knite	paris	0.13
3	5005	Pit Alex	London	0.11
4	5006	Mc Lyon	paris	0.14
5	5007	Paul Adam	Rome	0.13
6	5003	Lauson Hen	San Jose	0.12

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.

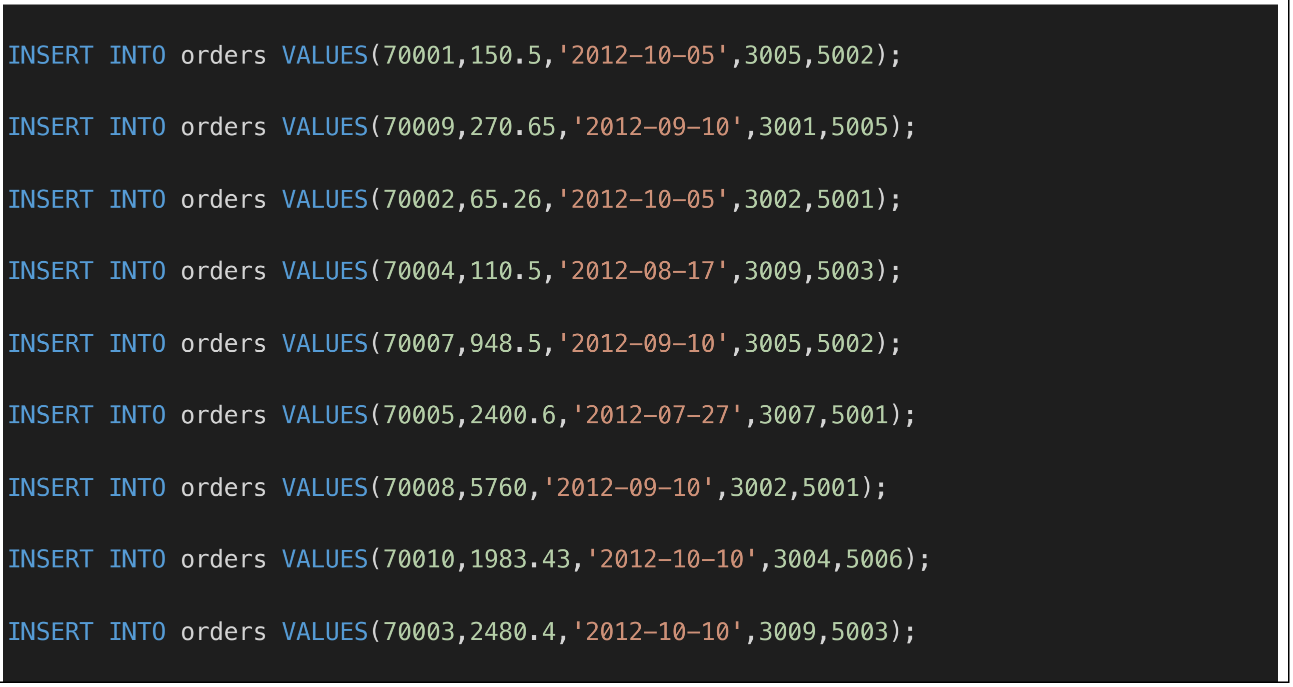
```
ord_no          purch_amt       ord_date        customer_id     salesman_id
--------------- --------------- --------------- --------------- ---------------
70001           150.5           2012-10-05      3005            5002
70009           270.65          2012-09-10      3001            5005
70002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



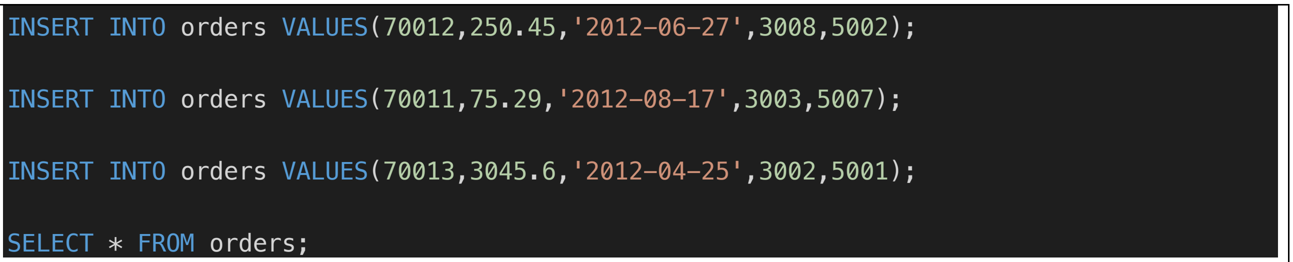
```sql
CREATE TABLE orders
(
ord_no int,
purch_amt text,
ord_date date,
customer_id int,
salesman_id int
);

SELECT * FROM orders;
```



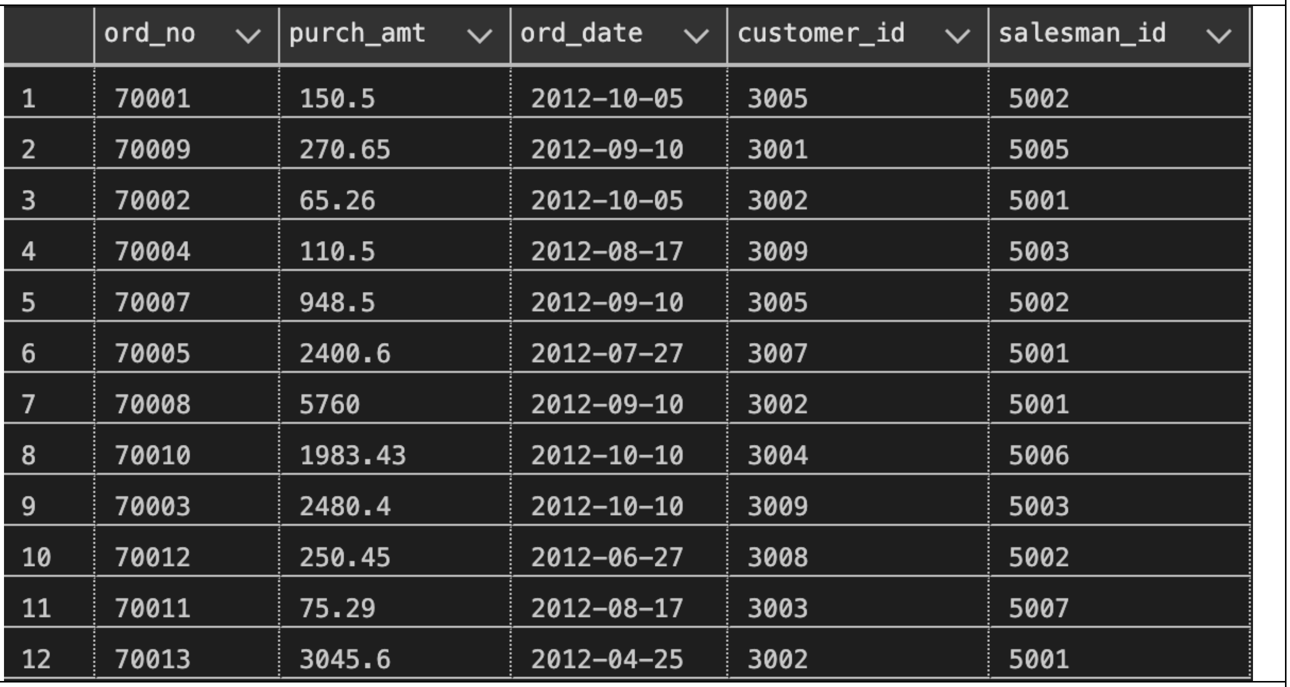
	ord_no	purch…	ord…	customer_…	salesman_…



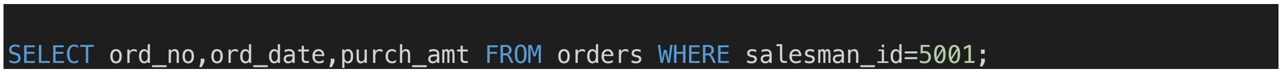
```sql
INSERT INTO orders VALUES(70001,150.5,'2012-10-05',3005,5002);

INSERT INTO orders VALUES(70009,270.65,'2012-09-10',3001,5005);

INSERT INTO orders VALUES(70002,65.26,'2012-10-05',3002,5001);

INSERT INTO orders VALUES(70004,110.5,'2012-08-17',3009,5003);

INSERT INTO orders VALUES(70007,948.5,'2012-09-10',3005,5002);

INSERT INTO orders VALUES(70005,2400.6,'2012-07-27',3007,5001);

INSERT INTO orders VALUES(70008,5760,'2012-09-10',3002,5001);

INSERT INTO orders VALUES(70010,1983.43,'2012-10-10',3004,5006);

INSERT INTO orders VALUES(70003,2480.4,'2012-10-10',3009,5003);
```



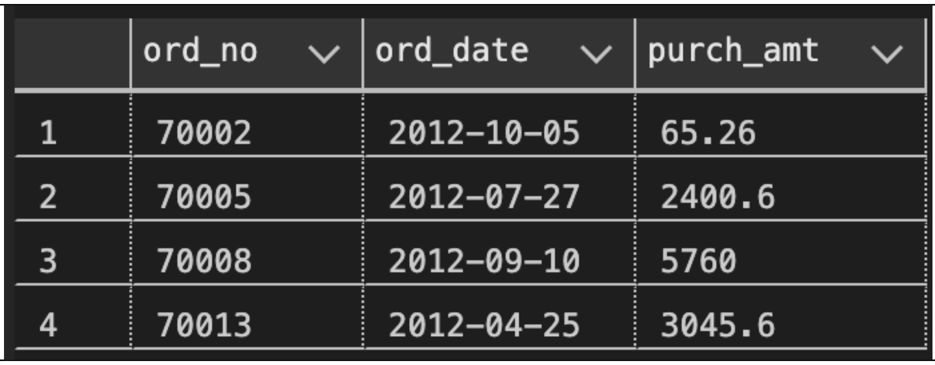
```
INSERT INTO orders VALUES(70012,250.45,'2012-06-27',3008,5002);

INSERT INTO orders VALUES(70011,75.29,'2012-08-17',3003,5007);

INSERT INTO orders VALUES(70013,3045.6,'2012-04-25',3002,5001);

SELECT * FROM orders;
```



	ord_no	purch_amt	ord_date	customer_id	salesman_id
1	70001	150.5	2012-10-05	3005	5002
2	70009	270.65	2012-09-10	3001	5005
3	70002	65.26	2012-10-05	3002	5001
4	70004	110.5	2012-08-17	3009	5003
5	70007	948.5	2012-09-10	3005	5002
6	70005	2400.6	2012-07-27	3007	5001
7	70008	5760	2012-09-10	3002	5001
8	70010	1983.43	2012-10-10	3004	5006
9	70003	2480.4	2012-10-10	3009	5003
10	70012	250.45	2012-06-27	3008	5002
11	70011	75.29	2012-08-17	3003	5007
12	70013	3045.6	2012-04-25	3002	5001



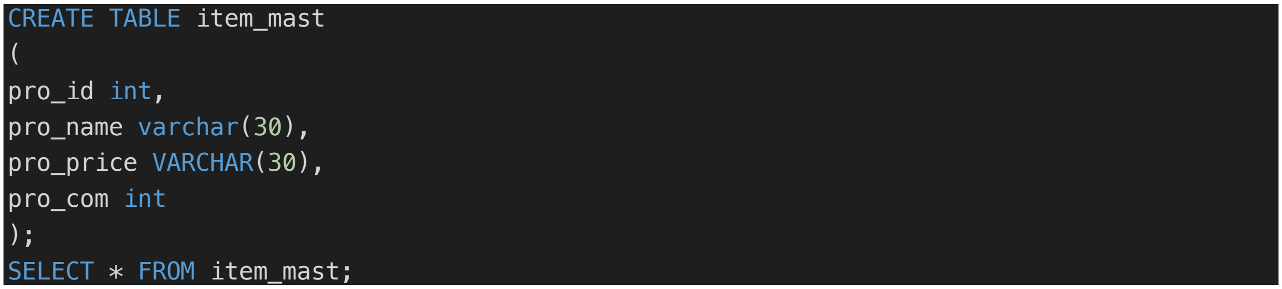
```
SELECT ord_no,ord_date,purch_amt FROM orders WHERE salesman_id=5001;
```



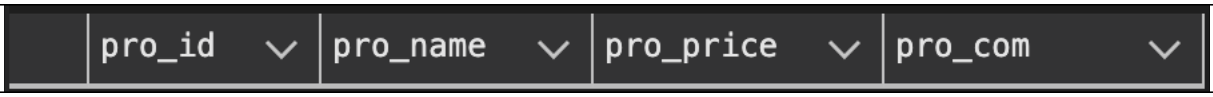
	ord_no	ord_date	purch_amt
1	70002	2012-10-05	65.26
2	70005	2012-07-27	2400.6
3	70008	2012-09-10	5760
4	70013	2012-04-25	3045.6

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.

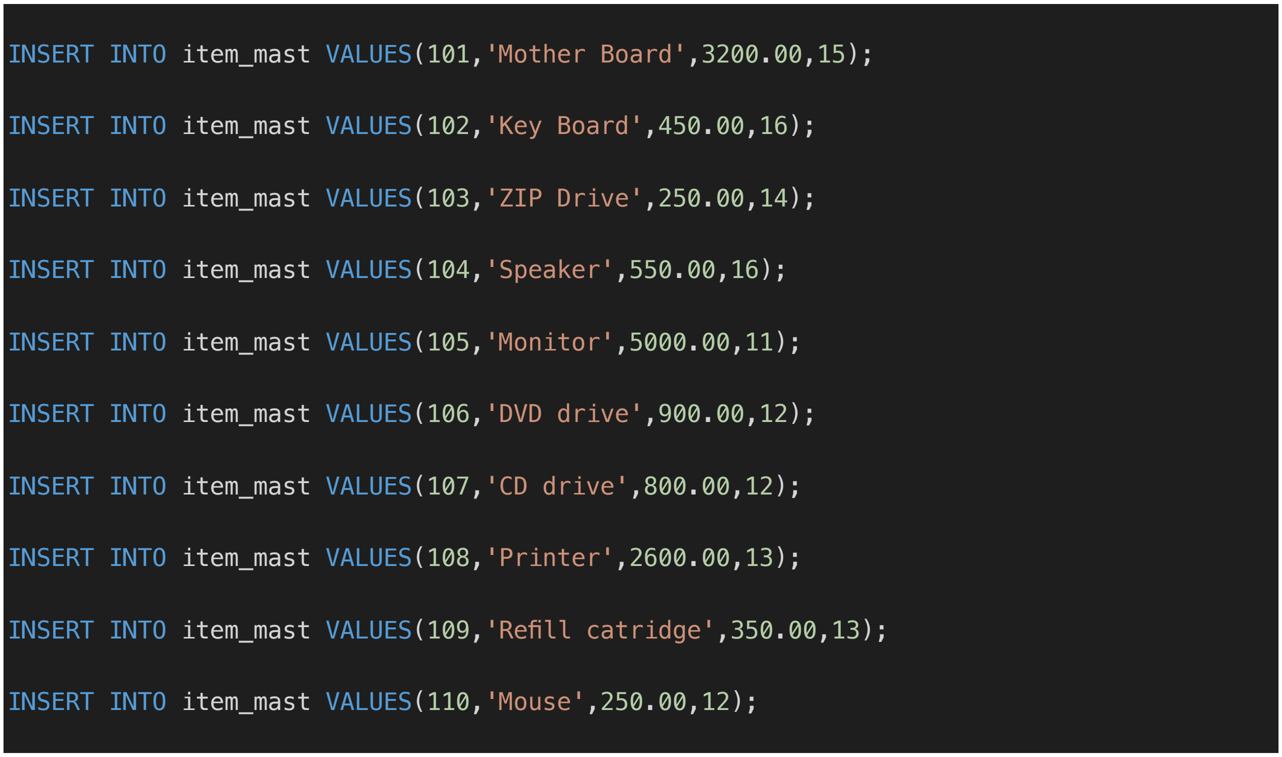
```
PRO_ID PRO_NAME                                   PRO_PRICE     PRO_COM
------------------------------------------------- ------------- ----------
101    Mother Board                               3200.00       15
102    Key Board                                   450.00       16
103    ZIP drive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



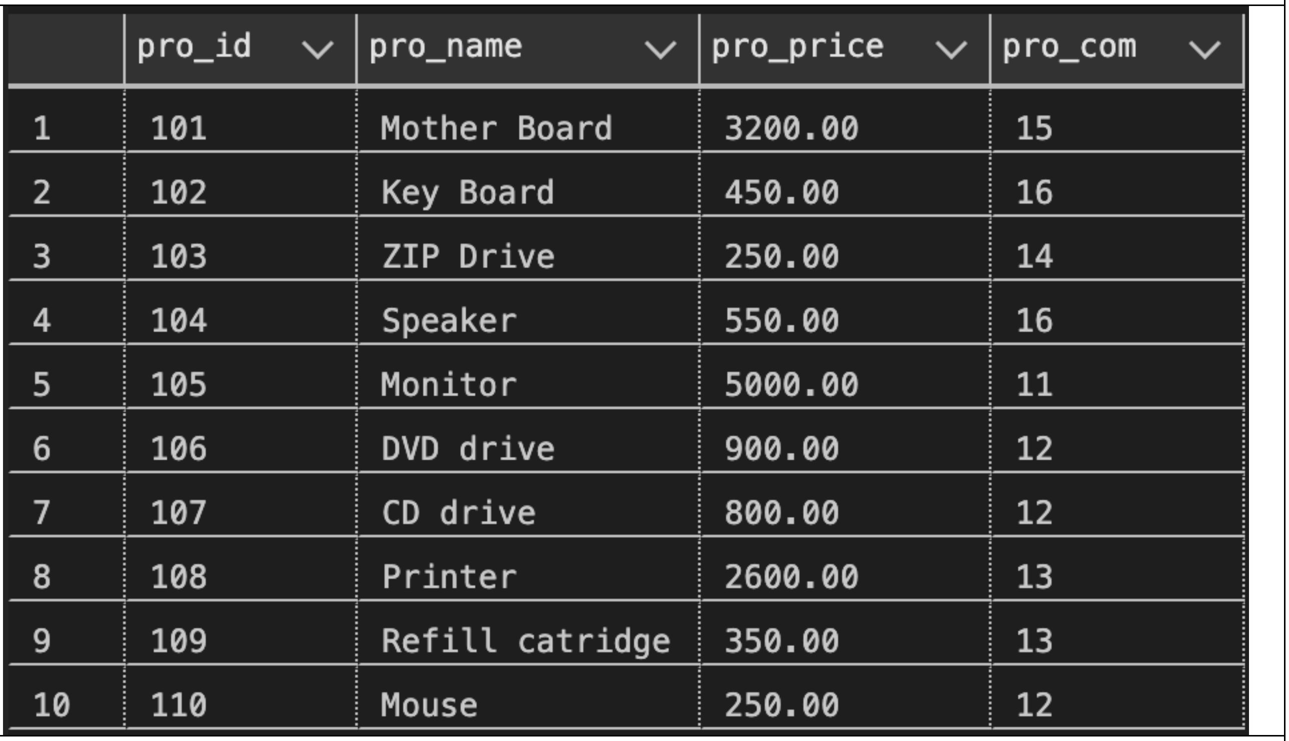
```sql
CREATE TABLE item_mast
(
pro_id int,
pro_name varchar(30),
pro_price VARCHAR(30),
pro_com int
);
SELECT * FROM item_mast;
```



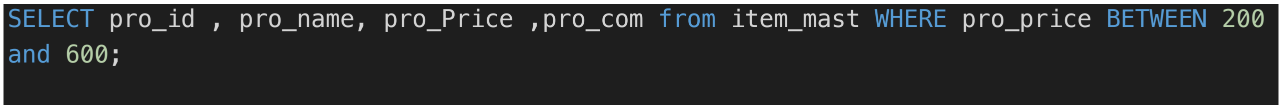
pro_id	pro_name	pro_price	pro_com



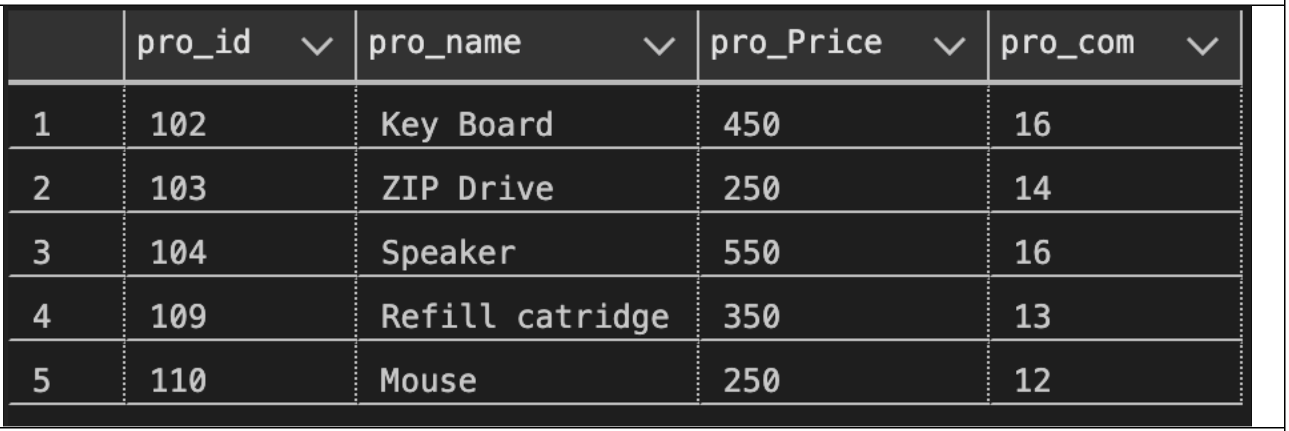
```sql
INSERT INTO item_mast VALUES(101,'Mother Board',3200.00,15);

INSERT INTO item_mast VALUES(102,'Key Board',450.00,16);

INSERT INTO item_mast VALUES(103,'ZIP Drive',250.00,14);

INSERT INTO item_mast VALUES(104,'Speaker',550.00,16);

INSERT INTO item_mast VALUES(105,'Monitor',5000.00,11);

INSERT INTO item_mast VALUES(106,'DVD drive',900.00,12);

INSERT INTO item_mast VALUES(107,'CD drive',800.00,12);

INSERT INTO item_mast VALUES(108,'Printer',2600.00,13);

INSERT INTO item_mast VALUES(109,'Refill catridge',350.00,13);

INSERT INTO item_mast VALUES(110,'Mouse',250.00,12);
```



	pro_id	pro_name	pro_price	pro_com
1	101	Mother Board	3200.00	15
2	102	Key Board	450.00	16
3	103	ZIP Drive	250.00	14
4	104	Speaker	550.00	16
5	105	Monitor	5000.00	11
6	106	DVD drive	900.00	12
7	107	CD drive	800.00	12
8	108	Printer	2600.00	13
9	109	Refill catridge	350.00	13
10	110	Mouse	250.00	12

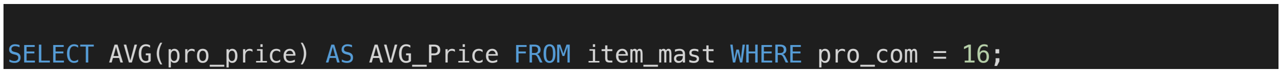


```
SELECT pro_id , pro_name, pro_Price ,pro_com from item_mast WHERE pro_price BETWEEN 200
and 600;
```

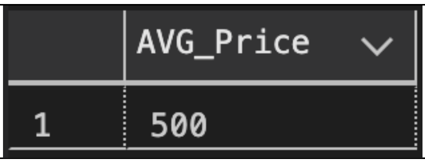


	pro_id	pro_name	pro_Price	pro_com
1	102	Key Board	450	16
2	103	ZIP Drive	250	14
3	104	Speaker	550	16
4	109	Refill catridge	350	13
5	110	Mouse	250	12

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



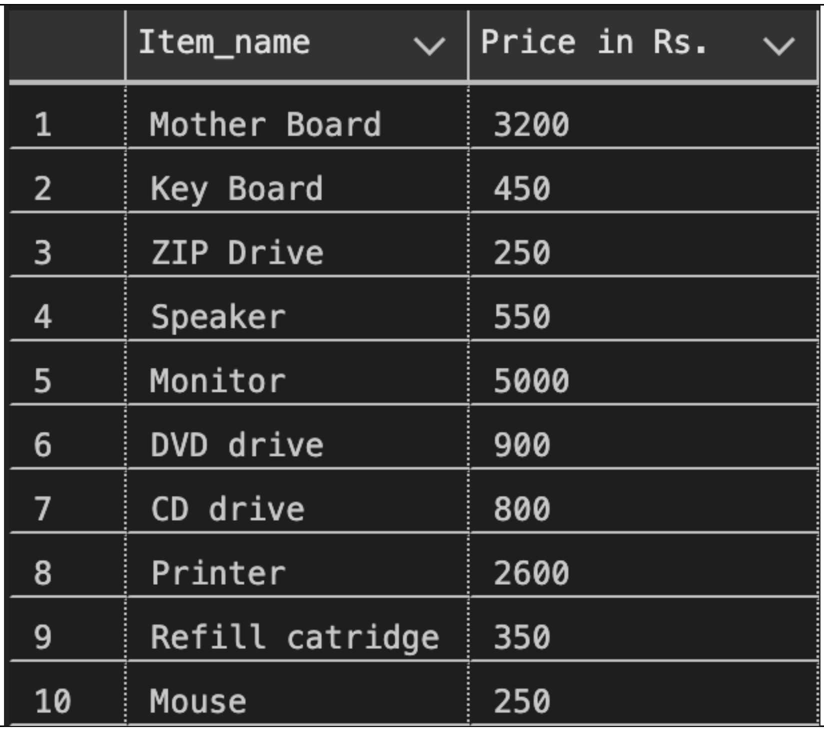
```
SELECT AVG(pro_price) AS AVG_Price FROM item_mast WHERE pro_com = 16;
```



	AVG_Price
1	500

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

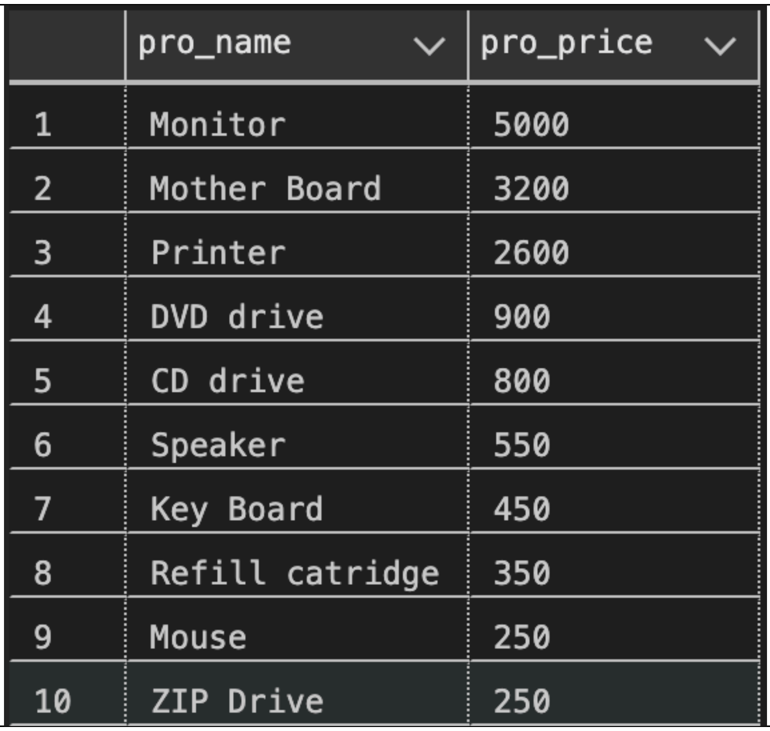
```
SELECT pro_name AS "Item_name",pro_Price AS "Price in Rs." FROM item_mast;
```



	Item_name	Price in Rs.
1	Mother Board	3200
2	Key Board	450
3	ZIP Drive	250
4	Speaker	550
5	Monitor	5000
6	DVD drive	900
7	CD drive	800
8	Printer	2600
9	Refill catridge	350
10	Mouse	250

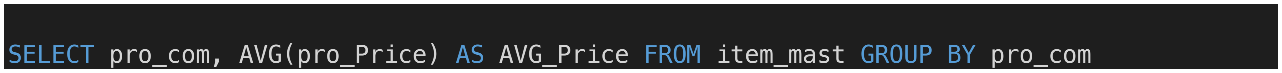
23. From the following table, write a SQL query to find the items whose prices are higher than or equal to $250. Order the result by product price in descending, then product name in ascending. Return pro_name and pro_price.

```
SELECT pro_name , pro_price FROM item_mast WHERE pro_price >= 250 ORDER BY pro_price
DESC,pro_name ASC;
```

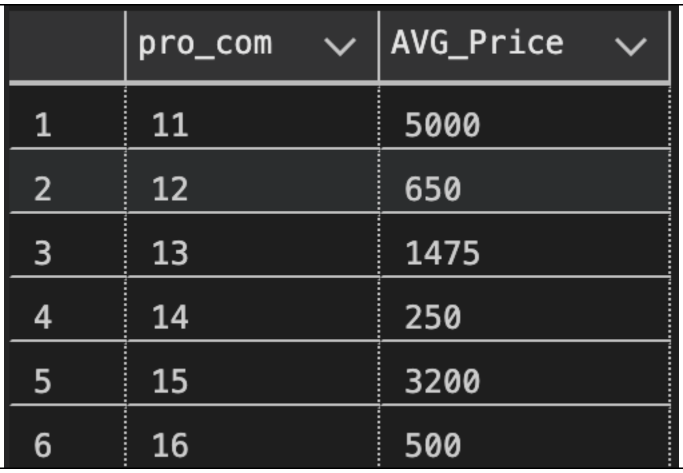


	pro_name	pro_price
1	Monitor	5000
2	Mother Board	3200
3	Printer	2600
4	DVD drive	900
5	CD drive	800
6	Speaker	550
7	Key Board	450
8	Refill catridge	350
9	Mouse	250
10	ZIP Drive	250

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



```
SELECT pro_com, AVG(pro_Price) AS AVG_Price FROM item_mast GROUP BY pro_com
```



	pro_com	AVG_Price
1	11	5000
2	12	650
3	13	1475
4	14	250
5	15	3200
6	16	500