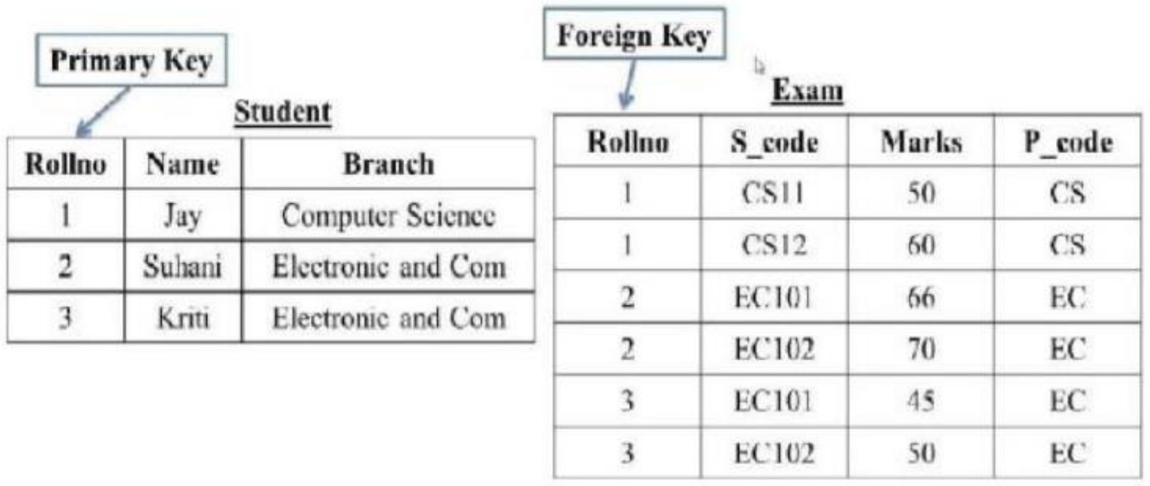
SQL-Queries

[1]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

Foreign Key

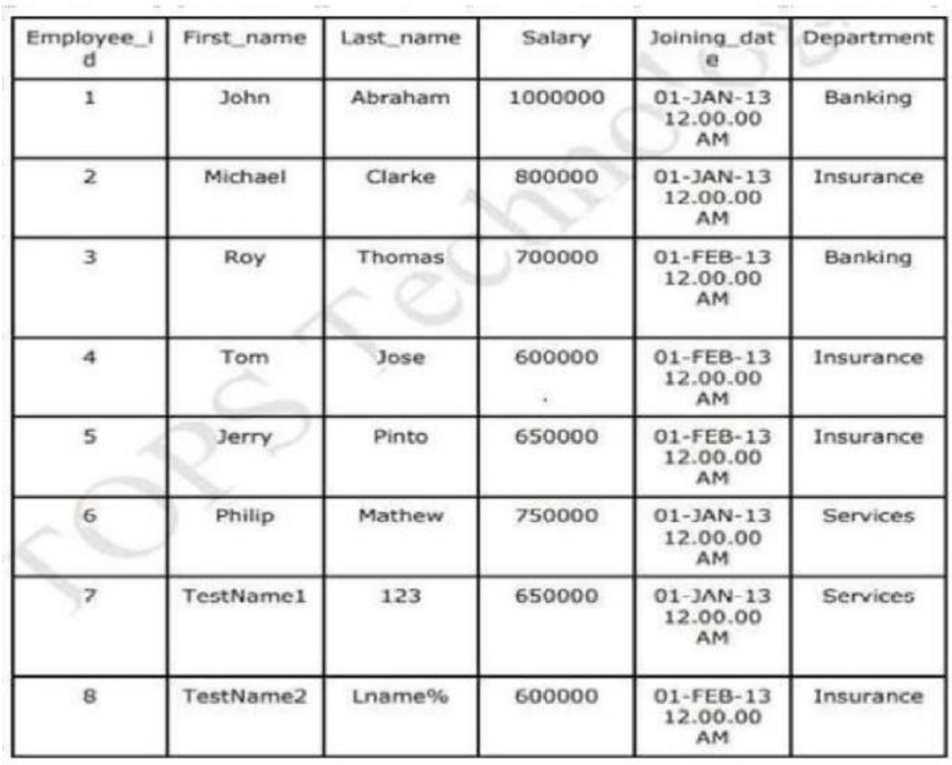
Exam

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

Student table	CREATE TABLE Student (
	   Roll_no INT AUTO_INCREMENT PRIMARY KEY,
	   name VARCHAR(255) NOT NULL,
	   branch VARCHAR(255) NOT NULL
	);

Exam table	CREATE TABLE Exam (
	   Roll_no INT AUTO_INCREMENT,
	   s_code int NOT NULL,
	   mark int ,
	   p_code VARCHAR(255) NOT NULL,
	Foreign KEY (Roll_no) REFERENCES student(Roll_no)
	);

2. Create table given below: Employee and IncentiveTable

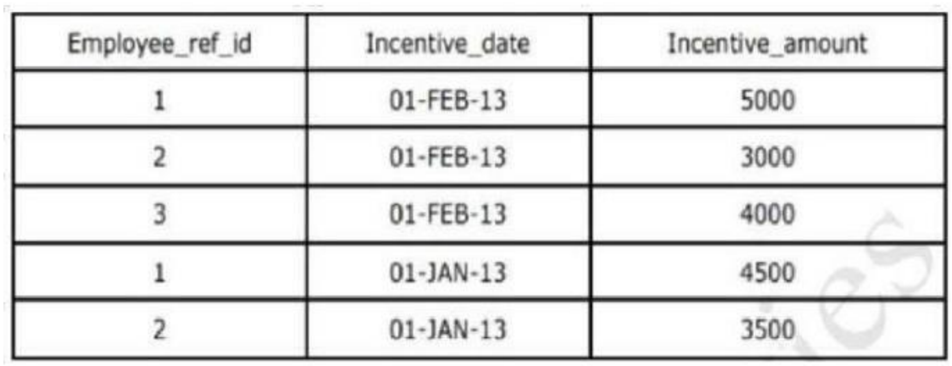


Employee_id	First_name	Last_name	Salary	Joining_date	Department
1	John	Abraham	1000000	01-JAN-13 12.00.00 AM	Banking
2	Michael	Clarke	800000	01-JAN-13 12.00.00 AM	Insurance
3	Roy	Thomas	700000	01-FEB-13 12.00.00 AM	Banking
4	Tom	Jose	600000	01-FEB-13 12.00.00 AM	Insurance
5	Jerry	Pinto	650000	01-FEB-13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

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

Employee table 2	CREATE TABLE Employee (Employee_id INT NOT NULL UNIQUE AUTO_INCREMENT, first_name VARCHAR(30), last_name VARCHAR(30), salary INT, joining_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL, department VARCHAR(50), PRIMARY KEY (Employee_id));
Incentent table	CREATE TABLE incentive (Employee_ref_id INT, incentive_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL, incentent_amount INT NOT NULL, FOREIGN KEY (Employee_ref_id) REFERENCES Employee(Employee_id));

3. Get First_Name from employee table using Tom name “Employee Name”.	SELECT * FROM `employee` WHERE first_name='tom';
4. Get FIRST_NAME, Joining Date, and Salary from employee table.	SELECT first_name,joining_date,salary FROM employee;
5. Get all employee details from the employee table order by First_Name ascending and Salary descending?	SELECT * FROM `employee`ORDER BY first_name ASC,salary DESC;
6. Get employee details from employee table whose first name contains ‘J’.	SELECT * FROM `employee` WHERE first_name LIKE 'j%';
7. Get department wise maximum salary from employee table order by 8. salaryascending?	SELECT department, MAX(salary) AS max_salary FROM employee GROUP BY department ORDER BY max_salary ASC;
9. Select first_name, incentive amount from employee and incentives table forthose employees who have incentives and incentive amount greater than 3000	SELECT e.first_name, i.amount AS incentive_amount FROM employee e INNER JOIN incentive i ON e.Employee_id = i.Employee_id WHERE i.amount > 3000;
10. Create After Insert trigger on Employee table which insert records in viewtable	create table viewtable(Employee_id int auto_increment, First_name varchar(30), Last_name varchar(30), Salary int, Joining_date varchar(50), Department varchar(30), primary key(Employee_id));

11. Create table given below: Salesperson and Customer

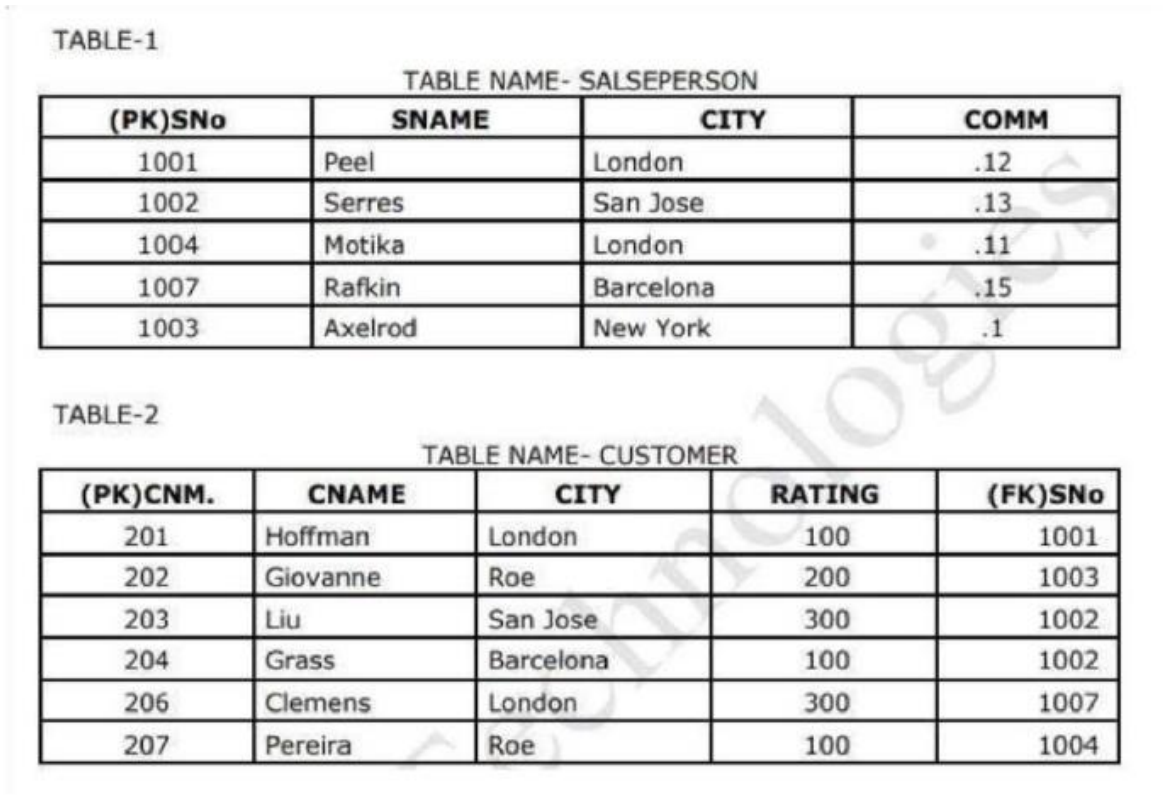


TABLE-1

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

Create salesperson table	CREATE TABLE salesperson
	(
	    sno int UNION NOT null,
	    sname varchar(30),
	    city varchar(35),
	    comm float,
	    PRIMARY KEY (sno)
	    );

Create customer Table	CREATE TABLE customer
	(
	   cno int UNION NOT null,
	   cname varchar(30),
	   city varchar(35),
	   rating int,
	   sno int
	
	   PRIMARY KEY (cno)
	   FOREIGN KEY (sno) REFERENCES salesperson(sno)
	   );

3

12. Retrieve the below data from above table
13.All orders for more than $1000.
	SELECT *
	FROM customer
	WHERE ORDER_value > 1000;

14.Names and cities of all salespeople in London with commission above 0.12	SELECT sname , city from salesperson
	WHERE comm >0.12;

15.All salespeople either in Barcelona or in London	SELECT * FROM salespersom
	WHERE city = 'barcelona' or city='london';

16. All salespeople with commission between 0.10 and 0.12. (Boundary valuesshould be excluded).	SELECT * FROM salesperson WHERE COMM BETWEEN 0.10 AND 0.12;

17. All customers excluding those with rating <= 100 unless they are located inRome	SELECT * from customer
	WHERE rating <= 100 or city = 'rome';

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
```

```
SELECT *  FROM salsepeople
```

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

```
SELECT ord_no, ord_date , purch_amt
FROM orders
```

```
[4]WHERE salesman_id = 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.
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

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

21. 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

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

22. 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

```
SELECT pro_name as 'item name',
concat('price in rs .',fromat(pro_price,2))
AS 'price in rs.'
from item mast;
```

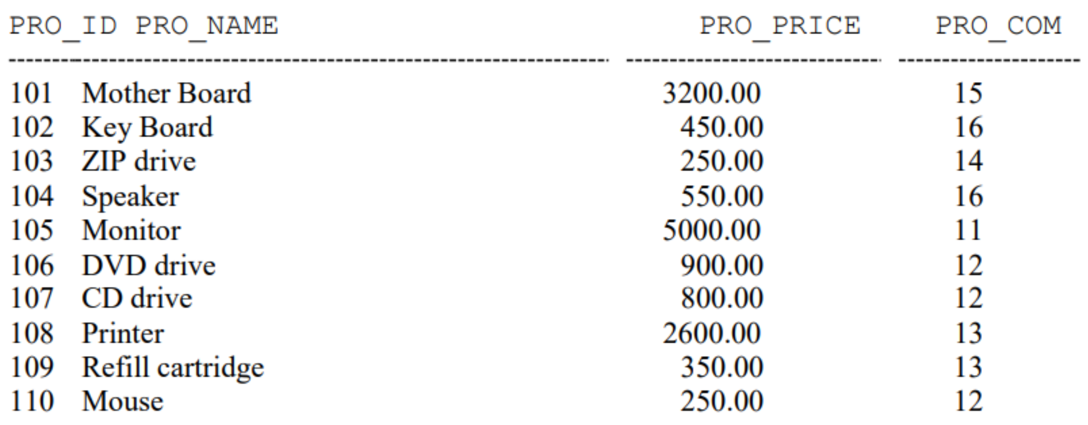
23. From the following table, write a SQL query to find the items whose prices are higher than or equal to $250. Order the result by product price in descending, then product name in ascending. Return pro_name and pro_price.

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
```

```
SELECT pro_name , pro_price
FROM item_mast
WHERE pro_price >=250.00
ORDER BY pro_price DESC , pro_name ASC
```

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

```
select pro_com, avg(pro_price) as average_price
from item_mast
group by pro_com;
```