Module 5 (Database)

1. What do you understand By Database

- Database – It is a collection of data in a format that can be easily accessed
- We can't access database directly
- We have to use DBMS (Database management system) to access data it works on backend and gives or provide required information.

2. What is Normalization?

- Normalization refers to a process that makes something clearer and understanding (Normal or regular).
- It optimizes storage space.
- Also helps to organize the data in the database.
- It is a multi-step process that sets the data into tabular form and removes the duplicated data from the relational tables.

3. What is Difference between DBMS and RDBMS?

DBMS	***RDBMS***
Data stored in file format	Data stored in table format
Supports a single user	Supports a multiple user

We can store data in small quantity	We can store data in large quantity
No connection between data	Data are linked to each other
The requirement of software and hardware are low	The requirement of software and hardware are low

4.What is MF Cod Rule of RDBMS Systems?

- Null values must be uniformly treated as “missing information,” not as empty strings, blanks, or zeros.
- Null values can also be interpreted as ‘inapplicable data’ or we can say ‘unknown information’.
- It should be handled consistently
- Expression on NULL must give null, primary key must not be null, ever.

5. What do you understand By Data Redundancy?

- It occurs when identical copies of the same data are stored in multiple locations, leading to a range of problems
- it can occur either intentionally or accidentally (Due to complex processes or inefficient coding).
- It is concern because it can lead to inconsistencies, update anomalies, database performance etc.

6. What is DDL Interpreter?

- Data definition language – Describe the portion of SQL that creates, alters, and deletes database objects.
- It includes schemas, table, catalogs, variables and many more.
- We can create table (using link also).

7. What is DML Compiler in SQL?

- DML- Data manipulation language.
- It is the component of the SQL statements.
- List of DML commands - Insert (For inserting data into table), Update (For updating existing data within data), Delete (For delete records), Lock (Table control concurrency)

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

- Primary key (Uniquely identifies)
- Foreign key (Referencing key)
- A primary key is a combination of fields that uniquely identify a record in a table, so that an individual record can be located without confusion.
- A foreign key sometimes called a referencing key used to link two tables together.
- Example: create table student (
  Roll no int,
  Name varchar,
  Number int,
  Primary key (Roll no),
  Foreign key (Number) REFERENCES students (Roll no);

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

- Rolling back any transaction.
- Also known as nested transactions.
- A special mark inside a transaction that allows all commands that are executed after it was established to be rolled back.

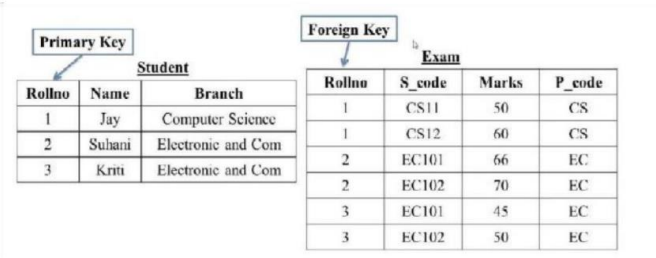
- Syntax for save point command:
  SAVEPOINT_NAME;(Creation of save point among all the transaction) ROLLBACK to the SAVEPOINT are ROLLBACK to SAVEPOINT_NAME.

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

- Trigger allows you to specify SQL actions that should be executed automatically when a specific event occurs in the database.
- SQL triggers are generally associated with a particular table, this means that when the table is deleted, all its associated triggers are deleted accordingly.
- Trigger creation in SQL:
  CREATE TRIGGER
  'games'. AFTER
  INSERT
  ON
  'games'. 'USERS' FOR EACH row BEGIN
  INSERT INTO
  another Table ()

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

Student:

```
CREATE TABLE Student(
   Rollno int NOT null PRIMARY KEY,
   Name varchar(10),
   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');
```

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

Exam:

```
CREATE TABLE Exam (
        Rollno int,
        S_code varchar(5),
        Marks int,
        P_code varchar(3),

    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');
```

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 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:

```
CREATE TABLE employee(
  employee_id int not null PRIMARY KEY,
   First_name varchar(10),
   Last_name varchar(10),
   Salary int(50),
   joining_date datetime,
   Department varchar(30)
);
INSERT INTO employee VALUES(7,'John','Abraham',1000000'2013-01-01 12:00:00 AM','Banking');
INSERT INTO employee VALUES(2,'Michael','Clarke',800000,'2013-01-01 12:00:00 AM','Banking');
INSERT INTO employee VALUES(3,'Roy','Thomas',700000,'2013-01-02 12:00:00 AM','Banking');
INSERT INTO employee VALUES(4,'Tom','Jose',600000,'2013-01-02 12:00:00 AM','Banking');
INSERT INTO employee VALUES(5,'Jerry','Pinto',650000,'2013-02-01 12:00:00 AM','Banking');
INSERT INTO employee VALUES(6,'Philip','Mathew',750000,'2013-01-01 12:00:00 AM','Banking');
INSERT INTO employee VALUES(7,'TestName1','123',650000,'2013-01-01 12:00:00 AM','Banking');
INSERT INTO employee VALUES(1,'TestName2','Lname%',600000,'2013-02-01 12:00:00 AM','Banking');
```

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	Banking
3	Roy	Thomas	700000	2013-01-02 12:00:00	Banking
4	Tom	Jose	600000	2013-01-02 12:00:00	Banking
5	Jerry	Pinto	650000	2013-02-01 12:00:00	Banking
6	Philip	Mathew	750000	2013-01-01 12:00:00	Banking
7	TestName1	123	650000	2013-01-01 12:00:00	Banking
8	TestName2	Lname%	600000	2013-02-01 12:00:00	Banking

Incentive Table:

```
CREATE TABLE Incentive (
  Employee_ref_id int,
  Incentive_date date,
  Incentive_amount int(10),

    FOREIGN KEY Employee_ref_id REFERENCES employee(employee_id)

);
```

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

Employee_ref_id	Incentive_date	Incentive_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".

```
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;
```

First_name	joining_date	Salary
John	2013-01-01 12:00:00	1000000
Michael	2013-01-01 12:00:00	800000
Roy	2013-01-02 12:00:00	700000
Tom	2013-01-02 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?

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

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

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

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

```
1 SELECT Department, MAX(Salary) as Maximum_salary FROM employee GROUP BY Department ORDER BY
  Salary asc;
```

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

```sql
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 view Table.

```sql
DELIMITER $$
CREATE TRIGGER INSERT_into_viewtable AFTER INSERT ON employee
FOR EACH ROW BEGIN
INSERT INTO Viewtable(e_id,name,Department,STATUS)
VALUES(new.e_id, new.e_name,new.department,'Insert Record');
END;
```

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

TABLE-1

```
CREATE TABLE salesperson(
    Sno int PRIMARY KEY not null,
    Sname varchar(10),
    City varchar(15),
    COMM float
);
```

```
INSERT INTO salesperson VALUES(1001,'Peel','London',.12);
INSERT INTO salesperson VALUES(1002,'Serres','San jose',.13);
INSERT INTO salesperson VALUES(1004,'Motika','London',.11);
INSERT INTO salesperson VALUES(1007,'Rafkin','Barcelona',.15);
INSERT INTO salesperson VALUES(1003,'Axeriod','New York',.1);
```

Sno	Sname	City	COMM
1001	Peel	London	0.12
1002	Serres	San jose	0.13
1003	Axeriod	New York	0.1
1004	Motika	London	0.11
1007	Rafkin	Barcelona	0.15

TABLE-2

```
CREATE TABLE customer(
    CNM int PRIMARY key not null,
    Cname varchar(20),
    City varchar(10),
    Rating int(10),
    sno int,

    FOREIGN KEY (sno) REFERENCES salesperson(sno)
);
```

```
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 jose',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,'Perira','Reo',100,1004);
```

CNM	Cname	City	Rating	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	Perira	Reo	100	1004

12.Retrieve the below data from above table
13.All orders for more than $1000.

```
1 SELECT * FROM customer WHERE order_value > 1000;
```

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

```
1 SELECT Sname, City from salesperson WHERE COMM>0.12;
```

Sname	City
Serres	San jose
Rafkin	Barcelona

15.All salespeople either in Barcelona or in London

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

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).

```
1 SELECT * FROM salesperson WHERE COMM BETWEEN 0.10 AND 0.12;
```

Sno	Sname	City	COMM
1001	Peel	London	0.12
1003	Axeriod	New York	0.1
1004	Motika	London	0.11

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

```
1 SELECT * FROM customer WHERE Rating <=100 or City='Rome';
```

CNM	Cname	City	Rating	sno
201	Hoffman	London	100	1001
204	Grass	Barcelona	100	1002
207	Perira	Reo	100	1004

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
```

```
1 SELECT * FROM salesperson;
```

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.

Sample table: orders

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

```
1 SELECT ord_no,ord_date,purch_amt FROM orders
2 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

```sql
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

```sql
SELECT AVG(pro_price) AS Average_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

```
1 SELECT POR_NAME AS 'Item Name',
2 CONCAT('Price in Rs. ', FORMAT(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

```
1 SELECT PRO_NAME,PRO_PRICE FROM item_mast WHERE PRO_PRICE >=250.00
2 ORDER BY PRO_PRICE DESC, PRO_NAME ASC;
```