MODULE – 5(DATABASE)

1.What do you understand By Database?

A database is a collection of data that is organized. It can be accessed or stored in a computer system. It can be managed through a [Database Management System](https://www.geeksforgeeks.org/introduction-of-dbms-database-management-system-set-1/)(DBMS), a software used to manage data.

2. What is Normalization?

Normalization is the process of organizing the data in the database.

Normalization divides the larger table into smaller and links them using relationships.

The normal form is used to reduce redundancy from the database table

Types of Normal Forms:

1. 1NF
2. 2NF
3. 3NF
4. BCNF
5. 4NF
6. 5NF

3. What is Difference between DBMS and RDBMS

1. RDBMS stores data in the form of tables, whereas DBMS stores data in the form of files.
2. In RDBMS, the tables have an identifier called primary key and the data values are stored in the form of tables. In DBMS, data is generally stored in either a hierarchical form or a navigational form.
3. **Normalization is** present in RDBMS. **Normalization is not** present in DBMS

4. What is MF Cod Rule of RDBMS Systems?

Some rules define a database to be the correct RDBMS. These rules were developed by Dr. Edgar F.Codd (E.F. Codd) in 1985, who has vast research knowledge on the Relational Model of database Systems. Codd presents his 13 rules for a database to test the concept of [DBMS](https://www.javatpoint.com/dbms-tutorial) against his relational model, and if a database follows the rule, it is called a true relational database (RDBMS). These 13 rules are popular in RDBMS, known as Codd's 12 rules.

1. The Foundation Rule
2. Information Rule
3. Guaranteed Access Rule
4. Systematic Treatment of Null Values
5. Active/Dynamic Online Catalog based on the relational model
6. Comprehensive Data SubLanguage Rule
7. View Updating Rule
8. Relational Level Operation (High-Level Insert, Update and delete) Rule
9. Physical Data Independence Rule
10. Logical Data Independence Rule
11. Integrity Independence Rule
12. Distribution Independence Rule
13. Non Subversion Rule

5. What do you understand By Data Redundancy?

Data redundancy occurs when the same piece of data exists in multiple places.

6. What is DDL Interpreter?

 DDL Interpreter DDL expands to [Data Definition Language](https://www.scaler.com/topics/ddl-in-dbms/). DDL Interpreter as the name suggests interprets the DDL statements such as schema definition statements like create, delete, etc.

7. What is DML Compiler in SQL?

A DML (data manipulation language) refers to a computer programming language that allows you to add (insert), delete (delete), and alter (update) data in a database.

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

SQL constraints are used to specify rules for the data in a table.

The following constraints are commonly used in SQL:

* [NOT NULL](https://www.w3schools.com/sql/sql_notnull.asp) - Ensures that a column cannot have a NULL value
* [UNIQUE](https://www.w3schools.com/sql/sql_unique.asp) - Ensures that all values in a column are different
* [PRIMARY KEY](https://www.w3schools.com/sql/sql_primarykey.asp) - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
* [FOREIGN KEY](https://www.w3schools.com/sql/sql_foreignkey.asp) - Prevents actions that would destroy links between tables
* [CHECK](https://www.w3schools.com/sql/sql_check.asp) - Ensures that the values in a column satisfies a specific condition
* [DEFAULT](https://www.w3schools.com/sql/sql_default.asp) - Sets a default value for a column if no value is specified

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

A SAVEPOINT is a point in a transaction in which you can roll the transaction back to a certain point without rolling back the entire transaction.

Syntax:

*SAVEPOINT SAVEPOINT\_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.

Example of trigger

CREATE TABLE ORDER\_TRIGGER(TID INT PRIMARY KEY,TNAME VARCHAR(39),TPRODUCT\_ID INT,TPRICE INT ,TDATE TIMESTAMP,ACTION VARCHAR(20));

DELIMITER $$

CREATE TRIGGER T01 BEFORE UPDATE ON order1 FOR EACH ROW

INSERT INTO order\_trigger (TID,TNAME,TPRODUCT\_ID,TPRICE,ACTION) VALUES (OLD.NAME,OLD.NAME,OLD.PRODUCT\_ID,OLD.PRICE,'DATA WILL UPDATE!');

UPDATE order1 SET NAME='JAIMIN' WHERE ID=7;

## **TOPICS - SQL QUERIES**

## Create Table Name : Student and Exam

* STUDENT:
* CREATE TABLE STUDENT

(ROLL\_NO INT PRIMARY KEY AUTO\_INCREMENT,NAME VARCHAR(30), BRANCH VARCHAR(30));

* INSERT INTO student (NAME,BRANCH) VALUES ("MAHIMA","INFORMATION TECHNOLOGY"),("DIMPLE","ELECTRICAL ENGINEERING"),

("NENCY","CHEMICAL ENGINEERING"),("NIKTA","CIVIL ENGINEERING"),("HEENA","COMPUTER ENGINEERING");

* SELECT \* FROM student;
* EXAM:
* CREATE TABLE EXAM

(ROLL\_NO INT ,S\_CODE VARCHAR(10) PRIMARY KEY,MARKS INT ,P\_CODE VARCHAR(10),FOREIGN KEY(ROLL\_NO) REFERENCES student(ROLL\_NO));

* INSERT INTO EXAM VALUES(1,"CS11",67,"CS"),(1,"CS12",78,"CS"),(3,"EC101",44,"EC"),(2,"EC102",97,"EC"),(5,"EC104",80,"EC"),(2,"EC202",62,"EC"),(3,"EC105",70,"EC");
* SELECT \* FROM EXAM LIMIT 4;

## 2.Create Table Name : PERSON

* CREATE TABLE PERSON

(FIRST\_NAME VARCHAR(20),LAST\_NAME VARCHAR(20),ADDRESS TEXT,CITY VARCHAR(30),AGE INT );

* INSERT INTO person VALUES

("RINKU","VAGHELA","123 KRISHNA SOCITY","KADI",21),

("JAIMINI","DABHI","002 RAMBAG SOCITY","DANGARVA",26),

("RIDHHIMA","SODHA","024 HOUSING BOARD","VIDAJ",31),

("MITTU","VAGHELA","98 HOUSING BOARD","MORBI",24),

("MALTI","ZALA","019 RADHE COMPLEX","AHEMDABAD",18),

("HEER","RAJ","005 SIVAM HOSPITAL","RAJKOT",29);

* SELECT FIRST\_NAME,CITY,AGE FROM `person` LIMIT 3;

### 3-CREATE TABLE: EMPLOYEE AND INCENTIVE

* TABLE-1: EMPLOYEE
* CREATE TABLE EMPLOYEE(EMPLOYEEE\_ID INT PRIMARY KEY AUTO\_INCREMENT,FIRST\_NAME VARCHAR(20),LAST\_NAME VARCHAR(20),SALARY BIGINT ,JOINING\_DATE DATETIME ,

DEPARTMENT VARCHAR(10));

* INSERT INTO employee(FIRST\_NAME,LAST\_NAME,SALARY,JOINING\_DATE,DEPARTMENT) VALUES("JOHN","ABRAHAM",1000000,'13-1-1',"BAMKING"); ("JOHN","ABRAHAM",1000000,'01-JAN-13 12:00',"BANKING");
* insert into employee (FIRST\_NAME,LAST\_NAME,SALARY,JOINING\_DATE,DEPARTMENT) VALUES

("MICHAEL","CLARKE",800000,'13-1-1',"INSURANCE"),("ROY","THOMAS",700000,'13-2-1',"BANKING"),

("TOM","JOSE",600000,'13-2-1',"INSURANCE"),("JERRY","PINTO",650000,'13-1-1',"INSURANCE"),

("PHILIP","MATHEW",750000,'13-1-1',"SERVICES"),("TESTNAME1","123",650000,'13-1-1',"SERVICES"),

("TESTNAME2","LNAME",600000,'13-2-1',"INSURANCE");

* SELECT EMPLOYEEE\_ID,FIRST\_NAME,LAST\_NAME,SALARY,DATE\_FORMAT(JOINING\_DATE,'%d-%b-%y %h.%i.%s %p'),DEPARTMENT FROM employee;
* TABLE-2: INCENTIVE
* [CREATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) INCENTIVE (EMPLOYEE\_REF\_ID INT ,INCENTIVE\_DATE DATE,INCENTIVE\_AMOUNT INT ,FOREIGN KEY (EMPLOYEE\_REF\_ID)REFERENCES EMPLOYEE(EMPLOYEEE\_ID));
* [INSERT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/insert.html) INTO incentive [VALUES](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/miscellaneous-functions.html%23function_values)(5,'13-2-1',5000),(2,'13-2-1',3000),(3,'13-2-1',4000),(5,'13-1-1',4500), (2,'13-2-1',3500);
* SELECT EMPLOYEE\_REF\_ID ,DATE\_format(INCENTIVE\_DATE,'%d-%b-%y'),INCENTIVE\_AMOUNT FROM `incentive` WHERE 1

RETRIEVE BELOW DATA FROM ABOVE TABLE

1. Get First\_Name from employee table using Tom name “Employee Name

SELECT FIRST\_NAME AS EMPLOYEE\_NAME FROM EMPLOYEE WHERE FIRST\_NAME="TOM";

1. Get FIRST\_NAME, Joining Date, and Salary from employee table.

SELECT FIRST\_NAME,DATE\_FORMAT(JOINING\_DATE,'%d-%b-%y %h.%i.%s %p'),SALARY FROM `employee`

c) Get all employee details from the employee table order by First\_Name

SELECT \* FROM employee ORDER BY FIRST\_NAME DESC;

D)Get employee details from employee table whose first name contains ‘J’.

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* FROM `employee` WHERE FIRST\_NAME [LIKE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-comparison-functions.html%23operator_like) '%J%';

E) Get department wise maximum salary from employee table order by salary ascending?

SELECT DEPARTMENT,MAX(SALARY) FROM employee GROUP BY DEPARTMENT ORDER BY SALARY ASC;

F) Select first\_name, incentive amount from employee and incentives table for those employees who have incentives and incentive amount greater than 3000

SELECT FIRST\_NAME ,INCENTIVE\_AMOUNT FROM employee AS E INNER JOIN incentive AS I ON E.EMPLOYEEE\_ID =I.EMPLOYEE\_REF\_ID

WHERE I.INCENTIVE\_AMOUNT>3000;

G) Create After Insert trigger on Employee table which insert records in viewtable

CREATE TABLE EMPLOYEE\_TRIGGER(TEMPLOYEEE\_ID INT PRIMARY KEY,TFIRST\_NAME VARCHAR(30),TLAST\_NAME VARCHAR(30),TSALARY INT,TJOINING\_DATE DATETIME,TDEPARTMENT VARCHAR(30),TDATE TIMESTAMP,ACTION\_PERFORMANCE VARCHAR(20));

DELIMITER $$

CREATE TRIGGER TEMPLOYEE AFTER INSERT ON employee FOR EACH ROW

INSERT INTO employee\_TRIGGER

(TEMPLOYEEE\_ID,TFIRST\_NAME,TLAST\_NAME,TSALARY,TJOINING\_DATE,TDEPARTMENT,ACTION\_PERFORMANCE) VALUES

(NEW.EMPLOYEEE\_ID,NEW.FIRST\_NAME,NEW.LAST\_NAME,NEW.SALARY,NEW.JOINING\_DATE,NEW.DEPARTMENT,'NEW DATA WILL INSERT');

INSERT INTO employee VALUES (10,'VAGHELA','RIN',30000,'12-3-21',"INFROMATION TECHNOLOGY");

### 4- CREATE TABLE : SALESPERSON AND CUSTOMER

* TABLE-1
* CREATE TABLE SALESPERSON

(SNO INT PRIMARY KEY AUTO\_INCREMENT,SNAME VARCHAR(20),CITY VARCHAR(20),COMM FLOAT(4));

CHANGE AUTO\_INCREAMENT STARTING VALUE

* [ALTER](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/alter-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/alter-table.html) salesperson AUTO\_INCREMENT=1001;
* INSERT INTO salesperson (SNAME,CITY,COMM) VALUES

("PEEL","LONDON",.12),

("SERRES","SAN JOSE",.13),

("MOTIKA","LONDON",.11),

("RAFKIN","BARCELONA",.15),

("AXEIROD","NEW YORK",.1);

* TABLE-2
* CREATE TABLE CUSTOMER(CNM INT PRIMARY KEY AUTO\_INCREMENT,CNAME VARCHAR(20),CITY VARCHAR(20), RATING INT ,SNO INT,FOREIGN KEY (SNO) REFERENCES salesperson(SNO));
* ALTER TABLE customer AUTO\_INCREMENT=201;
* INSERT INTO customer (CNAME,CITY,RATING,SNO) VALUES

("HOFFMAN","LONDON",100,1001),

("GLOVANNE","ROE",200,1003),

("LIU","SAN JOSE",300,1002),

("GRASS","BARCELONA",100,1002),

("CLEMENS","LONDON",300,1005),

("PEREIRA","ROE",100,1004);

RETRIEVE BELOW DATA FROM ABOVE TABLE

B) Names and cities of all salespeople in London with commission above 0.12

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) SNAME,CITY FROM `salesperson` WHERE CITY="LONDON" [AND](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_and) COMM>.12;

C) All salespeople either in Barcelona or in London

SELECT \* FROM `salesperson` WHERE CITY="BARCELONA" OR CITY="LONDON";

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

SELECT \* FROM salesperson WHERE COMM > 0.11 AND COMM <.12;

E ) All customers excluding those with rating <= 100 unless they are located in Rome

SELECT \* FROM customer WHERE NOT RATING<=100 AND NOT CITY="ROE";