**MODULE: 5**

**DATABASE**

1. **What do you understand By Database**

* A database is an **organized collection of data stored in a computer system** and usually controlled by a database management system (DBMS). The data in common databases is modeled in tables, making querying and processing efficient. Structured query language (SQL) is commonly used for data querying and writing.

1. **What is Normalization?**

* Normalization is the process of organizing a database to reduce redundancy and improve data integrity
* Normalization also simplifies the database design so that it achieves the optimal structure composed of atomic elements (i.e. elements that cannot be broken down into smaller parts).
* Also referred to as database normalization or data normalization, normalization is an important part of relational database design, as it helps with the speed, accuracy, and efficiency of the database.
* By normalizing a database, you arrange the data into [tables](https://database.guide/what-is-a-table/) and [columns](https://database.guide/what-is-a-column/).  You ensure that each table contains only related data. If data is not directly related, you create a new table for that data.

1. **What is Difference between DBMS and RDBMS?**

* **DBMS (Database management system)**
* Dbms is a software that is used to define, create, and maintain a database and provides controlled access to the data.
* A software program that not only stores databases but also maintains and manages them and controls access to them. Data is stored as files. Common examples of DBMS are Windows registry, Microsoft Access, MongoDB, and XML.
* A DBMS allows access, operation, and manipulation of databases. It manages the data in files through manipulation like sorting, deletion, and insertion to improve, refine, and control the database. It allows tasks like defining, updating, retrieving, and administration of data.
* **RDBMS (Relational database management system)**
* An advanced form of a DBMS. Data is organised in tables to interpret relations and differences. General examples of RDBMS are SQL, MySQL, Postgres, and Oracle.
* A RDBMS is a highly common and structured representation method. It focuses on multiple rows and columns in tables rather than files. Launched in the 1970s, it came with better productivity, innovation, efficiency, and speed than the DBMS.
* **SQL (structure query language)** is the standard language used to interact with RDBMS

1. **What is MF Cod Rule of RDBMS Systems?**

* Codd’s **twelve rules** provide a framework for defining what is required from a **Relational Database Management System (RDBMS)**. Here are some key rules:
* **Rule 0 (Foundation Rule)**: For a system to qualify as an RDBMS, it must manage databases entirely through its relational capabilities.
* **Rule 1 (Information Rule)**: All information in a relational database is explicitly represented at the logical level and in exactly one way—by values in tables.
* **Rule 2 (Guaranteed Access Rule)**: Each datum (atomic value) in a relational database is guaranteed to be logically accessible by combining table name, primary key value, and column name.
* **Rule 3 (Systematic Treatment of Null Values)**: Null values (distinct from empty strings or zeros) are supported systematically to represent missing or inapplicable information.
* **Rule 4 (Dynamic Online Catalog Based on the Relational Model)**: The database description (metadata) is represented at the logical level, allowing authorized users to query it using the same relational language.

1. **What do you understand By Data Redundancy?**

* **Data redundancy** in MySQL refers to storing the same data in multiple places within a database. While it has both advantages and disadvantages, let’s focus on the key points:
* **Disadvantages of Data Redundancy:**
  + - * **Inconsistency**: When you modify data (update/insert/delete), you need to do it in more than one place. This increases the risk of data inconsistencies across the database.
      * **Increased Size**: Redundant data unnecessarily inflates the size of the database.
      * **Efficiency Decrease**: It can decrease the efficiency of database operations.
      * **Data Corruption**: Excessive redundancy may lead to data corruption.
      * **Preventing Data Redundancy**: Database normalization techniques can help prevent redundancy.

1. **What is DDL Interpreter?**

* The **DDL Interpreter** (Data Definition Language Interpreter) processes statements related to database schema definitions. [It interprets commands like CREATE, ALTER, and DROP, which modify the structure of tables and other database objects](https://stackoverflow.com/questions/2578194/what-are-ddl-and-dml)
* The DDL interpreter interprets DDL statements and records the definition in the data dictionary. The DML compiler translates DML statements in a query language into an evaluation plan consisting of low-level instructions that the query evaluation engine understands. The DML compiler also performs query optimization, which is it picks the lowest cost evaluation plan from among the alternatives. Query evaluation engine executes low level instructions generated by the DML compiler.

1. **What is DML Compiler in SQL?**

* A data manipulation language (DML) is a family of computer languages including commands permitting users to manipulate data in a database. This manipulation involves inserting data into database tables, retrieving existing data, deleting data from existing tables and modifying existing data. DML is mostly incorporated in SQL databases.
* In SQL (Structured Query Language), a Data Manipulation Language (DML) compiler is a component responsible for processing and executing DML statements. DML statements are used to manipulate the data stored in the database, such as inserting, updating, deleting, and querying records.

1. **What is SQL Key Constraints writing an Example of SQL Key Constraints**

* SQL key constraints are used to enforce rules on columns in a database table, ensuring data integrity and consistency. There are several types of key constraints in SQL, including:

**NOT NULL Constraint**:

* Ensures that a column cannot store NULL values.

**UNIQUE Constraint**:

* Requires that the column values be unique.

**PRIMARY KEY Constraint**:

* Combines NOT NULL and UNIQUE constraints to uniquely identify rows.

**FOREIGN KEY Constraint**:

* References a record in another table.

1. **What is save point? How to create a save point write a query**

* Save point is a command in SQL that is used with the rollback command.
* It is a command in Transaction Control Language that is used to mark the transaction in a table.
* Consider you are making a very long table, and you want to roll back only to a certain position in a table then; this can be achieved using the save point.
* If you made a transaction in a table, you could mark the transaction as a certain name, and later on, if you want to roll back to that point, you can do it easily by using the transaction's name.
* Save point is helpful when we want to roll back only a small part of a table and not the whole table. In simple words, we can say save point is a bookmark in SQL.

**Example: -**

-- Start a transaction

START TRANSACTION;

-- Insert a record into the 'employees' table

INSERT INTO employees (employee\_id, first\_name, last\_name) VALUES (1, 'John', 'Doe');

-- Create a savepoint

SAVEPOINT savepoint1;

-- Insert another record into the 'employees' table

INSERT INTO employees (employee\_id, first\_name, last\_name) VALUES (2, 'Jane', 'Smith');

-- Rollback to the savepoint if needed

-- This will undo the insertion of Jane Smith but keep the insertion of John Doe

ROLLBACK TO SAVEPOINT savepoint1;

-- Insert a different record

INSERT INTO employees (employee\_id, first\_name, last\_name) VALUES (3, 'Alice', 'Brown');

-- Commit the transaction

COMMIT;

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

* A trigger in SQL is a set of commands that automatically executes or "fires" when certain events occur in the database. These events can be actions like INSERT, UPDATE, or DELETE on a specified table. Triggers are useful for maintaining data integrity, enforcing business rules, and automatically generating derived column values.

**Types of Triggers: -**

1. **BEFORE Trigger:** Executes before the specified operation (INSERT, UPDATE, DELETE) on the table.
2. **AFTER Trigger:** Executes after the specified operation on the table.

**Table structure: -**

CREATE TABLE employees (

employee\_id INT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

hire\_date DATE

);

CREATE TABLE audit\_log (

log\_id INT AUTO\_INCREMENT PRIMARY KEY,

employee\_id INT,

action VARCHAR(50),

action\_time TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

**Creating a Trigger: -**

DELIMITER $$

CREATE TRIGGER after\_employee\_insert

AFTER INSERT ON employees

FOR EACH ROW

BEGIN

INSERT INTO audit\_log (employee\_id, action)

VALUES (NEW. Employee\_id, 'INSERT');

END$$

DELIMITER;