**Module 4 – Introduction to DBMS**

* **Introduction to SQL:-**

**1. What is SQL, and why is it essential in database management?**

SQL is a declarative programming language used to communicate with and manage relational databases. It allows users to perform various operations like querying, creating, modifying, and deleting data, as well as managing access permissions. SQL is essential because it provides a standardized way to interact with relational databases, regardless of the specific DBMS being used.

**2. Explain the difference between DBMS and RDBMS.**

A DBMS is a software system that allows users to create, define, manipulate, and manage databases. It provides tools for managing data storage, retrieval, and security. An RDBMS is a specific type of DBMS that uses a relational database model. This means that data is organized into tables with rows and columns, and relationships between tables are defined through keys.

**3. Describe the role of SQL in managing relational databases.**

**SQL plays a crucial role in managing relational databases by providing the language for interacting with and manipulating the data stored within them. It allows users to perform tasks like:**

* **Querying data:** Retrieving specific data based on defined criteria.
* **Creating and modifying database structures:** Defining tables, columns, and relationships.
* **Updating and deleting data:** Modifying and removing data within the database. **Managing access permissions:** Controlling who can access and modify specific data

**4. What are the key features of SQL?**

* **Data Definition Language (DDL):** Used for creating, modifying, and deleting database structures (tables, columns, constraints).
* **Data Manipulation Language (DML):** Used for inserting, updating, and deleting data within the database
* **Data Query Language (DQL):** Used for retrieving data from the database
* **Data Control Language (DCL):** Used for managing access permissions and security within the database
* **Transactions:** Allowing for a sequence of SQL statements to be executed as a single unit of work, ensuring data consistency and integrity
* **Functions and Procedures:** Allowing for the creation of reusable code blocks for performing specific tasks.
* **Views:** Providing virtual tables that can be used to simplify complex queries or provide different perspectives of the data.
* **SQL Syntax :-**

**1. What are the basic components of SQL syntax?**

**Basic Components of SQL Syntax:**

* **Keywords:** These are reserved words that define the action or type of SQL statement (e.g., SELECT, FROM, WHERE, INSERT, UPDATE, DELETE).
* **Clauses:** These are blocks of code within a SQL statement that add specific functionality, such as selecting columns, filtering data, or grouping results. Common clauses include SELECT, FROM, WHERE, ORDER BY, GROUP BY, and HAVING.
* **Expressions:** These are used to evaluate values, perform calculations, and compare data. They can include literals, variables, and operators.
* **Table Names:** These identify the specific table within the database that the SQL statement is operating on.
* **Column Names:** These specify the individual columns (attributes or fields) within a table to be retrieved or modified.

**2. General Structure of an SQL SELECT Statement:**

The basic structure of a SELECT statement is:

|  |
| --- |
| SELECT column\_name(s) FROM table\_name WHERE condition\_to\_filter\_data ORDER BY column\_name(s) [ASC | DESC] GROUP BY column\_name(s) HAVING aggregate\_filter |

**Here's a breakdown:**

* SELECT: Specifies which columns to retrieve from the table.
* FROM: Indicates the table from which data will be retrieved.
* WHERE: Filters the rows based on a specified condition, only returning rows that meet the criteria.
* ORDER BY: Sorts the results in ascending or descending order based on one or more columns.
* GROUP BY: Groups rows with the same values in a specific column or columns, often used with aggregate functions.
* HAVING: Filters the groups created by GROUP BY based on an aggregate function.

**3. Role of Clauses in SQL Statements:**

**Clauses are essential for controlling and refining the results of SQL queries. They allow you to:**

* **Filter Data:** The WHERE and HAVING clauses enable you to select only the data that meets specific criteria.
* **Sort Data:** The ORDER BY clause allows you to organize the retrieved data in a meaningful order.
* **Group Data:** The GROUP BY clause facilitates the aggregation of data based on shared values in specific columns.
* **Aggregate Data:** Aggregate functions (e.g., SUM, AVG, COUNT, MAX, MIN) are often used with GROUP BY to calculate statistics for groups of rows.
* **Specify Columns:** The SELECT clause determines which columns from the table will be included in the result set.
* **Define the Table:** The FROM clause specifies the source table or tables for the query.
* **SQL Constraints:-**

**1. What are constraints in SQL? List and explain the different types of constraints?**

In SQL, constraints are rules that limit the type of data that can be inserted into a table, ensuring data integrity and consistency. Types of constraints include Primary Key, Foreign Key, NOT NULL, UNIQUE, CHECK, DEFAULT, and Index. Primary Key and Foreign Key relate to uniquely identifying rows and establishing relationships between tables, while NOT NULL and UNIQUE control the values within specific columns.

Types of Constraints:

* **Primary Key:** A column (or set of columns) that uniquely identifies each row in a table. It cannot contain NULL values and must be unique.
* **Foreign Key:** A column (or set of columns) in one table that references the Primary Key of another table, establishing a relationship between the tables.
* **NOT NULL:** Ensures that a column cannot contain NULL values.
* **UNIQUE:** Ensures that all values in a column are distinct, allowing NULL values, unlike Primary Key.
* **CHECK:** Specifies a condition that a column must satisfy.
* **DEFAULT:** Sets a default value for a column if no value is provided during insertion.
* **INDEX:** Used to create and retrieve data from the database more quickly.

**2. How do PRIMARY KEY and FOREIGN KEY constraints differ?**

* **Purpose:**

Primary Keys uniquely identify rows within their table, while Foreign Keys establish relationships between tables by referencing the Primary Key of another table.

* **Scope:**

Primary Keys are defined within a single table, while Foreign Keys connect columns across multiple tables.

* **Relationship:**

Primary Keys define the identity of records within a table, while Foreign Keys enforce referential integrity by ensuring that values in one table correspond to values in another.

**3. What is the role of NOT NULL and UNIQUE constraints?**

* **NOT NULL:**

Prevents blank or NULL values from being entered into a column. This is useful for ensuring that essential information is always present.

* **UNIQUE:**

Ensures that each value in a column is unique, preventing duplicate entries. This is important for maintaining data integrity and consistency.