**EXPERIMENT-1**

**DBMS (Database Management System)** is a software system designed to manage databases. It allows users to store, retrieve, and manipulate data in an organized way. A DBMS provides tools for creating, reading, updating, and deleting data (CRUD operations), as well as managing security, access control, and backups.

**Key features of DBMS:**

* Data storage and management.
* Data retrieval.
* Data integrity and security.
* Data concurrency control (handling multiple users).

Examples: Microsoft Access, Oracle, MySQL, SQLite (when not using relational features).

**RDBMS (Relational Database Management System)** is a type of DBMS that stores data in a structured format using rows and columns (tables). It follows the relational model, where data is stored in tables, and relationships can be established between these tables using keys (primary and foreign keys).

**Key features of RDBMS:**

* Tables (relations) with rows (tuples) and columns (attributes).
* Data is stored in a tabular form.
* Supports SQL (Structured Query Language) for querying and managing data.
* Allows data integrity and relationships using primary and foreign keys.
* Ensures ACID properties (Atomicity, Consistency, Isolation, Durability).

Examples: MySQL, PostgreSQL, Oracle Database, Microsoft SQL Server.

| **Feature** | **DBMS** | **RDBMS** |
| --- | --- | --- |
| **Data Storage** | Data is stored in files (not necessarily structured). | Data is stored in tables (structured format). |
| **Data Model** | Supports various models (hierarchical, network, etc.). | Follows the relational model (tables). |
| **Normalization** | Not necessarily supported. | Data normalization is supported to reduce redundancy. |
| **Relationships** | Does not support relationships between tables. | Supports relationships using primary and foreign keys. |
| **Query Language** | Supports simple query languages (e.g., File-based queries). | Uses SQL to query and manage data. |
| **Data Integrity** | Basic checks (optional). | Stronger integrity constraints (primary keys, foreign keys, etc.). |
| **ACID Properties** | Not strictly implemented. | Supports ACID properties for transaction management. |
| **Multi-user Support** | May or may not support multiple users. | Supports multiple users with concurrent access. |

In an **RDBMS (Relational Database Management System)**, a **table** is a collection of related data organized in a structured format using **rows** and **columns**. Each table represents a specific entity or concept within the database, such as customers, orders, products, etc. The table is the fundamental building block where all the data is stored.

**Key Characteristics of a Table in RDBMS:**

1. **Rows (Records/tuples)**:
   * Each row in a table represents a single record or instance of the entity.
   * For example, in a table of "Employees," each row could represent a single employee with all their details (ID, name, salary, etc.).
2. **Columns (Attributes/fields)**:
   * Columns represent the characteristics or attributes of the entity.
   * For example, in the "Employees" table, columns could include **EmployeeID**, **Name**, **Department**, **Salary**, etc.
   * Each column has a defined **data type** (e.g., INTEGER, VARCHAR, DATE) that determines what kind of data it can store.
3. **Primary Key**:
   * A primary key is a unique identifier for each row in the table.
   * For instance, in the "Employees" table, **EmployeeID** could be the primary key because each employee will have a unique ID.
4. **Foreign Key**:
   * A foreign key is a column that creates a relationship between two tables. It links to the primary key of another table.
   * For example, in an "Orders" table, a **CustomerID** column may act as a foreign key linking back to the **CustomerID** primary key in the "Customers" table.
5. **Constraints**:
   * **Not Null**: Ensures that a column cannot have a NULL value.
   * **Unique**: Ensures that all values in a column are distinct.
   * **Check**: Enforces domain integrity by limiting the values a column can hold (e.g., age cannot be negative).
6. **Indexes**:
   * Indexes can be created on columns to speed up data retrieval and query performance. For example, an index on the **EmployeeID** column can make searches faster.

**Introduction to SQL (Structured Query Language)**

SQL (Structured Query Language) is a standard programming language used to manage and manipulate relational databases. It provides a powerful way to communicate with a database, allowing users to create, retrieve, update, and delete data. SQL is widely used by developers, database administrators, and data analysts for tasks ranging from simple data retrieval to complex reporting and data analysis.

**Key Features of SQL:**

1. **Data Querying**: SQL is primarily used for querying and retrieving data from databases using the SELECT statement.
2. **Data Manipulation**: SQL allows for inserting new data (INSERT), updating existing data (UPDATE), and removing data (DELETE).
3. **Database Management**: SQL can be used to define and manage database schemas using CREATE, ALTER, and DROP commands.
4. **Data Integrity**: It supports enforcing constraints like PRIMARY KEY, FOREIGN KEY, UNIQUE, and NOT NULL to maintain data consistency and validity.
5. **Data Security**: SQL allows for controlling access to data using GRANT and REVOKE permissions, ensuring data security.

**Types of SQL Languages**

SQL is divided into five categories based on functionality:

1**. Data Query Language (DQL)** – Used for retrieving data.

* Command: SELECT

2. **Data Definition Language (DDL)** – Defines the database structure.

* Commands: CREATE, ALTER, DROP, TRUNCATE

3. **Data Manipulation Language (DML)** – Modifies existing data.

* Commands: INSERT, UPDATE, DELETE

4. **Data Control Language (DCL)** – Controls user access to data.

* Commands: GRANT, REVOKE

5. **Transaction Control Language (TCL)** – Manages database transactions.

* Commands: COMMIT, ROLLBACK, SAVEPOINT

**SQL Process (Query Execution Process)**

When an SQL query is executed, it follows these steps:

1 .Query Parsing – SQL syntax is checked.

2 .Query Optimization – The best execution plan is selected.

3 .Query Execution – The database processes the query.

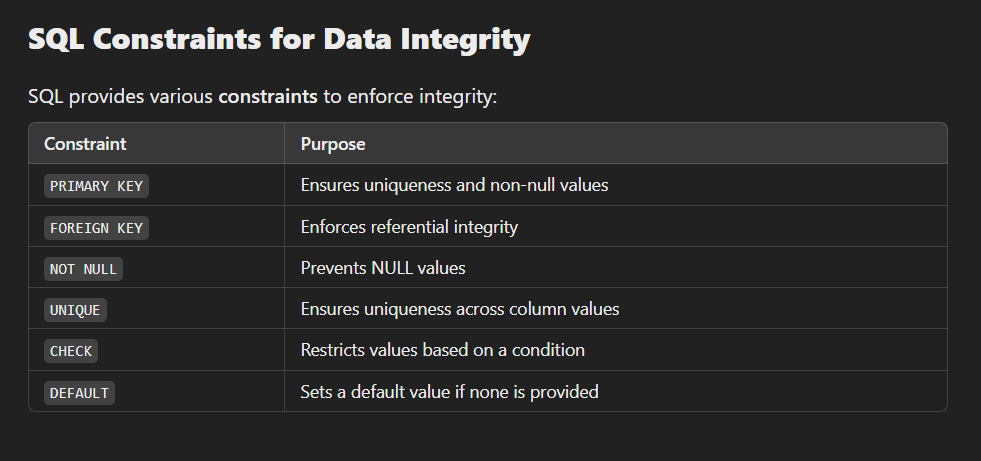
4 .Result Fetching – The final output is displayed to the user.

**Data Integrity in SQL (Summary)**

Data integrity ensures accuracy, consistency, and reliability in a database. It is enforced using **constraints, transactions, and normalization**.

**Types of Data Integrity:**

1. **Entity Integrity** → Uses **PRIMARY KEY** to ensure uniqueness.
2. **Referential Integrity** → Uses **FOREIGN KEY** to maintain relationships.
3. **Domain Integrity** → Uses **NOT NULL, CHECK, UNIQUE** to enforce valid values.
4. **User-Defined Integrity** → Enforces business rules via **triggers & procedures**.

**Transactions for Consistency:**

* Use **COMMIT** to save changes.
* Use **ROLLBACK** to undo errors.