Lab Session - 1

1. What is DML?

- **Definition:** DML (Data Manipulation Language) refers to a subset of SQL used to manage and manipulate data stored in a database.
- **Key Operations:** INSERT, UPDATE, DELETE, SELECT.
- **SQL Example:**

  ```
  INSERT INTO students (id, name, age) VALUES (1, 'John Doe', 20);
  SELECT * FROM students;
  ```

---

2. What are the commands in DML?

- **List:**
  - INSERT: Adds new records to a table.
  - UPDATE: Modifies existing records.
  - DELETE: Removes records.
  - SELECT: Retrieves data from one or more tables.
- **Examples:**
  - INSERT:

    ```
    INSERT INTO employees (id, name, position) VALUES (1, 'Alice', 'Manager');
    ```

  - UPDATE:

    ```
    UPDATE employees SET position = 'Senior Manager' WHERE id = 1;
    ```

  - DELETE:

    ```
    DELETE FROM employees WHERE id = 1;
    ```

  - SELECT:

    ```
    SELECT name FROM employees WHERE position = 'Manager';
    ```

---

3. What is a Primary Key?

- **Definition:** A Primary Key is a unique identifier for a record in a table. It ensures no duplicate or null values.
- **Rules:**
  - Each table must have one primary key.
  - A primary key column cannot contain NULL values.
- **SQL Example:**

  ```
  CREATE TABLE users (
      user_id INT PRIMARY KEY,
      username VARCHAR(50)
  );
  ```

---

4. What are the types of databases?

- **Types and Examples:**
  1. Relational Databases (e.g., MySQL, PostgreSQL).
  2. NoSQL Databases (e.g., MongoDB, Cassandra).
  3. Hierarchical Databases (e.g., IBM Information Management System).

---

5. What are the types of relationships in databases?

- **Explanation:**

  - **One-to-One:** A single record in one table corresponds to one record in another. *(Example: Passport and Citizen)*
  - **One-to-Many:** One record in a table is related to multiple records in another. *(Example: Teacher and Students)*
  - **Many-to-Many:** Multiple records in one table relate to multiple records in another. *(Example: Students and Courses)*
- **ER Diagram:** *(A simple drawing tool would be needed for this.)*

---

6. What is an Entity?

- **Definition:** An entity is an object in a database that stores data about a real-world concept.
- **Entity vs. Attribute:**
  - Entity: A thing (e.g., Student).
  - Attribute: Properties of the entity (e.g., Name, Age).
- **Example:**
  - Entity: Book.
  - Attributes: Title, Author, ISBN.

---

7. What are the types of Inner and Outer Joins?

- **Types:**
  - **Inner Join:** Returns matching rows from both tables.
  - **Left Outer Join:** Returns all rows from the left table and matching rows from the right.
  - **Right Outer Join:** Returns all rows from the right table and matching rows from the left.
  - **Full Outer Join:** Returns rows when there’s a match in either table.
- **Example:**

```
SELECT employees.name, departments.department_name
FROM employees
INNER JOIN departments ON employees.department_id = departments.id;
```

8. Difference between Primary Key and Unique Key?

- **Comparison:**
    1. Primary Key does not allow NULL; Unique Key allows one NULL value.
    2. Only one Primary Key per table; multiple Unique Keys allowed.
    3. Primary Key is used as a table's main identifier; Unique Key ensures unique values.
- **SQL Example:**

```
CREATE TABLE test (
    id INT PRIMARY KEY,
    email VARCHAR(100) UNIQUE
);
```

9. What is a Relationship in a Database Model?

- **Definition:** A relationship connects two or more tables logically.
- **Role of Foreign Key:** Ensures data integrity by linking a column in one table to a column in another.
- **Example:**

```
CREATE TABLE orders (
    order_id INT PRIMARY KEY,
    customer_id INT,
    FOREIGN KEY (customer_id) REFERENCES customers(id)
);
```

10. What is the Foreign Key?

- **Definition:** A Foreign Key is a column in one table that refers to a Primary Key in another table.
- **Purpose:** To enforce referential integrity between tables.
- **SQL Example:**

```
CREATE TABLE payments (
    payment_id INT PRIMARY KEY,
    order_id INT,
    FOREIGN KEY (order_id) REFERENCES orders(order_id)
);
```