Comments and Tables :

Best Practices for SQL Comments

1. **Document the Purpose**:
   - Explain the purpose of the query or section of code.

```
-- Fetching active employees for the report
SELECT * FROM employees WHERE status = 'active';
```

2. **Disable Code Temporarily**:
   - Comment out code during debugging or testing.

```
-- DELETE FROM employees WHERE age < 25;
```

3. **Use Comments for Complex Logic**:
   - Provide explanations for complex queries or logic.

```
/*
Joining employees and departments
to get department names for each employee
*/
SELECT e.name, d.department_name
FROM employees e
INNER JOIN departments d ON e.department_id = d.id;
```

Multi-Line Comments

- Use `/* */` to write comments spanning multiple lines.
- Everything between `/*` and `/` is ignored by the SQL interpreter.

Syntax:

```
/*
This is a multi-line comment.
It can span across multiple lines.
Useful for detailed explanations.
*/
SELECT name, age FROM employees;
```

Tables:

CLASS CODE :

select database();

use prince_datas;

- - CREATE TABLE table_name (
  -- column1 datatype contriants,
  -- column2 datatype,
  -- column3 datatype,
  -- ....
  --);
- - CREATE TABLE STUDENTS1 (
  -- STUDENT_ID INT ,
  -- NAME VARCHAR(50),
  -- AGE INT
  --);

- - CREATE TABLE employees (
  -- emp_id INT PRIMARY KEY,
  -- name VARCHAR(50),
  -- department VARCHAR(50),
  -- salary DECIMAL(10, 2) DEFAULT 30000
  --);
- - DESCRIBE employees
  -- SHOW TABLES ;
  -- EXPLAIN employees;

- - RENAME THE TABLE
- - RENAME table employees TO STAFF;
- - SHOW TABLES;
  -- RENAME table STUDENTS1 TO MYSTUDENTS;
  -- SHOW TABLES;
- - DROP TABLE MYSTUDENTS ;
- - SHOW TABLES;
- - DROP TABLE STAFF;students

SHOW TABLES ;

- - CREATE TABLE employees (
  -- emp_id INT PRIMARY KEY,
  -- name VARCHAR(50),
  -- department VARCHAR(50),
  -- salary DECIMAL(10, 2) DEFAULT 30000
  --);

- - CREATE TABLE employees1 (
  -- emp_id INT PRIMARY KEY,
  -- name VARCHAR(50),
  -- department VARCHAR(50),
  -- salary DECIMAL(10, 2) DEFAULT 30000
  --);
- - DROP TABLE employees,employees1;

DROP TABLE IF EXISTS employees;

---

1. `SHOW TABLES`

- **Purpose**: Lists all the tables in the currently selected database.
- **Scope**: Shows only table names.
- **Use Case**: When you want to see which tables exist in a database.

Syntax:

```
sql
Copy code
SHOW TABLES;
```

Example:

If the current database contains two tables, `employees` and `departments`, the output will look like:

```
diff
Copy code
+--------------------+
| Tables_in_dbname   |
```

```
+--------------------+
| employees          |
| departments        |
+--------------------+
```

---

2. `DESC table`

- **Purpose**: Describes the structure of a specific table, including column names, data types, constraints, and other properties.
- **Scope**: Displays column details like type, nullability, primary key, default values, etc.
- **Alias**: `DESCRIBE table` is a shortcut for `SHOW COLUMNS FROM table`.
- **Use Case**: When you need to examine the schema (structure) of a table.

Syntax:

```
sql
Copy code
DESC table_name;
```

Example:

For the table `employees`:

```
sql
Copy code
DESC employees;
```

Output:

```
sql
Copy code
+----------+--------------+------+-----+---------+-----------
-----+
| Field    | Type         | Null | Key | Default | Extra
|
+----------+--------------+------+-----+---------+-----------
-----+
| emp_id   | INT          | NO   | PRI | NULL    | auto_incre
ment |
| name     | VARCHAR(50)  | YES  |     | NULL    |
|
| age      | INT          | YES  |     | NULL    |
|
+----------+--------------+------+-----+---------+-----------
-----+
```

---

3. `EXPLAIN table`

- **Purpose**: Used to analyze and understand how MySQL executes a query, particularly useful for optimizing SELECT statements.
- **Scope**: Provides execution details like access type, key usage, row estimates, etc.
- **Use Case**: When you need to analyze the performance of a query (like `SELECT`).

Syntax:

```
sql
Copy code
EXPLAIN SELECT * FROM table_name;
```

Example:

For the query:

```
sql
Copy code
EXPLAIN SELECT * FROM employees WHERE age > 30;
```

Output:

```
sql
Copy code
+----+-------------+-----------+-------+---------------+-----
----+---------+-------+------+-------------+
| id | select_type | table     | type  | possible_keys | key
| key_len | ref   | rows | Extra       |
+----+-------------+-----------+-------+---------------+-----
----+---------+-------+------+-------------+
|  1 | SIMPLE      | employees | ALL   | NULL          | NULL
| NULL    | NULL  |  100 | Using where |
+----+-------------+-----------+-------+---------------+-----
----+---------+-------+------+-------------+
```

Summary of Differences:

Command	Purpose	Scope	Output
`SHOW TABLES`	Lists all tables in the current database	Table names only	List of table names
`DESC table`	Describes the structure of a specific table	Column details (name, type, constraints, etc.)	Table schema details
`EXPLAIN table`	Analyzes how a query interacts with the table	Execution details for a query	Query execution plan

1. Basic Table Creation

A simple table with columns and their data types.

```
CREATE TABLE students (
    student_id INT,
    name VARCHAR(50),
    age INT
);
```

2. Table with Default Values

A table where a column has a default value.

```
CREATE TABLE employees (
    emp_id INT PRIMARY KEY,
    name VARCHAR(50),
    department VARCHAR(50),
    salary DECIMAL(10, 2) DEFAULT 30000
);
```

3. Table with Constraints

Adding constraints like `PRIMARY KEY`, `FOREIGN KEY`, `NOT NULL`, `UNIQUE`.

```
CREATE TABLE orders (
    order_id INT PRIMARY KEY,
    customer_id INT NOT NULL,
    order_date DATE NOT NULL,
```

```
    total_amount DECIMAL(10, 2) NOT NULL,
    UNIQUE (order_id, customer_id)
);
```

3. MySQL Rename Tables

Used to rename a table in the database.

Example 1: Rename a Single Table

```
RENAME TABLE employees TO staff;
```

. Basic Syntax for Dropping a Table

The `DROP TABLE` command is used to delete a table permanently from the database, including all its data and structure.

```
DROP TABLE table_name;
```

Example:

```
DROP TABLE employees;
```

This will delete the `employees` table from the database.

---

2. Drop Multiple Tables

You can drop multiple tables in a single command by separating their names with commas.

```
DROP TABLE table1, table2, table3;
```

Example:

```
DROP TABLE students, courses;
```

---

3. Drop Table If It Exists

To avoid errors when trying to drop a table that might not exist, use the `IF EXISTS` clause.

```
DROP TABLE IF EXISTS table_name;
```

Example:

```
DROP TABLE IF EXISTS temp_users;
```

This checks if the table `temp_users` exists before dropping it.

Tables and Database :

```
sql
Copy code
-- Create the database if it does not exist
CREATE DATABASE IF NOT EXISTS sales;

-- Use the sales database
USE sales;

-- Show all records from the customers table
SELECT * FROM customers;

-- Show all records from the customers table within the sales
database
SELECT * FROM sales.customers;
```