Q1. Explain the fundamental differences between DDL, DML, and DQL commands in SQL. Provide one example for each type of command.

SQL commands are grouped based on what they do. **DDL (Data Definition Language)** commands define or modify the structure of database objects such as tables and schemas. They change the schema itself, not the data. Example:

```
CREATE TABLE Employees (id INT, name VARCHAR(50));
```

DML (Data Manipulation Language) commands handle inserting, updating, and deleting the actual data stored in tables. Example:

```
INSERT INTO Employees VALUES (1, 'Atul');
```

DQL (Data Query Language) is used only for fetching data from the database, usually through queries. Example:

```
SELECT * FROM Employees;
```

Q2.Question 2 : What is the purpose of SQL constraints? Name and describe three common types of constraints, providing a simple scenario where each would be useful.

SQL **constraints** ensure the accuracy, validity, and reliability of data inside a table. They protect the database from incorrect or inconsistent data.

1. **PRIMARY KEY** – Ensures each record is unique and not null.
   Scenario: An `employee_id` column must uniquely identify each employee.

2. **FOREIGN KEY** – Maintains referential integrity between tables by ensuring a value exists in another table.
   Scenario: An `order.customer_id` must match a valid `customer.id`.

3. **CHECK** – Enforces a rule on a column's values.
   Scenario: A `salary` column must be greater than zero: `CHECK (salary > 0)`.

These constraints help maintain structured, high-quality data across the database.

Question 3 : Explain the difference between LIMIT and OFFSET clauses in SQL. How would you use them together to retrieve the third page of results, assuming each page has 10 records?

The LIMIT clause in SQL is used to specify how many records you want to retrieve. It controls the number of rows returned.

The OFFSET clause is used to skip a certain number of rows before starting to return results. It controls where the result should start.

They are commonly used together when implementing pagination (showing data page-by-page).

Retrieving the 3rd page of results (10 records per page)

- **Each page has 10 records.**
- **Page 1 → skip 0 rows**
- **Page 2 → skip 10 rows**
- **Page 3 → skip 20 rows, then show the next 10.**

So the query becomes:

```
SELECT *
FROM your_table
LIMIT 10
OFFSET 20;
```

This skips the first 20 records and fetches the next 10, which corresponds to the third page of results.

Question 4 : What is a Common Table Expression (CTE) in SQL, and what are its main benefits? Provide a simple SQL example demonstrating its usage.
Answer :

A **Common Table Expression (CTE)** is a temporary named result set defined using the `WITH` keyword. It exists only for the duration of a single query. CTEs make complex queries easier to read, maintain, and debug.

Main Benefits:

1. **Improves readability** by breaking long queries into smaller logical blocks.
2. **Allows reusability** of the same result set within a query.

3. **Supports recursion**, useful for hierarchical data (e.g., employee–manager relationships).

Example:

```
WITH HighSalary AS (
    SELECT name, salary
    FROM Employees
    WHERE salary > 50000
)
SELECT * FROM HighSalary;
```

Question 5 : Describe the concept of SQL Normalization and its primary goals. Briefly explain the first three normal forms (1NF, 2NF, 3NF).

SQL Normalization & Normal Forms

Normalization is the process of organizing database tables to reduce redundancy and improve data integrity. Its primary goals are:

- Eliminate duplicate data
- Ensure data dependencies are logical
- Improve consistency and reduce anomalies (insert, update, delete)

First Three Normal Forms:

1. **1NF (First Normal Form):**
    - No repeating groups or multi-valued attributes.
    - All columns contain atomic (indivisible) values.
2. **2NF (Second Normal Form):**
    - Must satisfy 1NF.
    - No partial dependency: non-key attributes must depend on the whole primary key (applies to composite keys).
3. **3NF (Third Normal Form):**

- Must satisfy 2NF.
- No transitive dependency: non-key attributes should not depend on other non-key attributes.