1. DQL Command - SELECT

DQL (Data Query Language) commands are used for querying and retrieving data from the database. The most important DQL command is **SELECT**, which allows you to retrieve data from one or more tables.

Basic Syntax of SELECT:

```
sql
Copy
SELECT column1, column2, ...
FROM table_name;
```

- **SELECT** retrieves data from one or more columns in a table.
- You can use `*` to select all columns.

Examples:

1. **Selecting all columns:**

   ```
   sql
   Copy
   SELECT * FROM Employees;
   ```

2. **Selecting specific columns:**

   ```
   sql
   Copy
   SELECT Name, Age FROM Employees;
   ```

3. **Selecting with conditions (WHERE):**

   ```
   sql
   Copy
   SELECT * FROM Employees WHERE Department = 'Sales';
   ```

---

2. CRUD Operations

CRUD stands for **Create, Read, Update, Delete**, and these are the basic operations you can perform on data in a database.

CRUD and Corresponding SQL Commands:

- **Create**: `INSERT` statement.
- **Read**: `SELECT` statement.
- **Update**: `UPDATE` statement.
- **Delete**: `DELETE` statement.

Examples of CRUD Operations:

- **Create (INSERT):**

```
sql
Copy
INSERT INTO Employees (Name, Age, Department, Salary)
VALUES ('Alice Johnson', 29, 'HR', 75000);
```

- **Read (SELECT):**

```
sql
Copy
SELECT * FROM Employees WHERE Department = 'HR';
```

- **Update (UPDATE):**

```
sql
Copy
UPDATE Employees
SET Salary = 80000
WHERE EmployeeID = 1;
```

- **Delete (DELETE):**

```
sql
Copy
DELETE FROM Employees WHERE EmployeeID = 2;
```

These commands allow you to interact with the data in your database by creating, reading, updating, and deleting records.

---

3. Aggregate Functions

Aggregate functions are used to perform calculations on multiple rows of a table and return a single value. These functions are useful when you need to summarize or analyze data.

Common Aggregate Functions:

- **COUNT()**: Returns the number of rows.
- **SUM()**: Returns the sum of a numeric column.
- **AVG()**: Returns the average of a numeric column.
- **MIN()**: Returns the smallest value in a column.
- **MAX()**: Returns the largest value in a column.

Example 1: COUNT

To find the total number of employees in the `Employees` table:

```
sql
Copy
SELECT COUNT(*) FROM Employees;
```

Example 2: SUM

To find the total salary paid to all employees:

```
sql
Copy
SELECT SUM(Salary) FROM Employees;
```

Example 3: AVG

To find the average salary of all employees:

```
sql
Copy
SELECT AVG(Salary) FROM Employees;
```

Example 4: MIN and MAX

To find the minimum and maximum salaries in the company:

```
sql
Copy
SELECT MIN(Salary), MAX(Salary) FROM Employees;
```

---

4. Group By Clause

The **GROUP BY** clause is used to group rows that have the same values in specified columns into summary rows, such as finding the number of employees in each department.

Syntax:

```
sql
Copy
SELECT column1, aggregate_function(column2)
FROM table_name
GROUP BY column1;
```

- It is often used with aggregate functions like `COUNT()`, `SUM()`, `AVG()`, etc.

Example 1: Group By with COUNT

To count the number of employees in each department:

```
sql
Copy
SELECT Department, COUNT(*) AS EmployeeCount
FROM Employees
GROUP BY Department;
```

Example 2: Group By with SUM

To find the total salary paid to each department:

```
sql
```

```
Copy
SELECT Department, SUM(Salary) AS TotalSalary
FROM Employees
GROUP BY Department;
```

Example 3: Group By with AVG

To find the average salary of employees in each department:

```
sql
Copy
SELECT Department, AVG(Salary) AS AverageSalary
FROM Employees
GROUP BY Department;
```

---

5. Having Clause

The **HAVING** clause is used to filter the results of a **GROUP BY** query based on a condition applied to aggregate functions. While the **WHERE** clause filters rows before grouping, the **HAVING** clause filters the groups after grouping.

Syntax:

```
sql
Copy
SELECT column1, aggregate_function(column2)
FROM table_name
GROUP BY column1
HAVING condition;
```

- You can use **HAVING** to filter results based on aggregate values like `SUM()`, `AVG()`, `COUNT()`, etc.

Example 1: Using HAVING with COUNT

To find departments that have more than 5 employees:

```
sql
Copy
SELECT Department, COUNT(*) AS EmployeeCount
FROM Employees
GROUP BY Department
HAVING COUNT(*) > 5;
```

Example 2: Using HAVING with SUM

To find departments where the total salary is greater than $500,000:

```
sql
Copy
SELECT Department, SUM(Salary) AS TotalSalary
FROM Employees
GROUP BY Department
HAVING SUM(Salary) > 500000;
```

Example 3: Using HAVING with AVG

To find departments where the average salary is greater than $70,000:

```
sql
Copy
SELECT Department, AVG(Salary) AS AverageSalary
FROM Employees
GROUP BY Department
HAVING AVG(Salary) > 70000;
```

Summary of Key Concepts:

- **SELECT**: Retrieves data from the database.
- **CRUD Operations**: Basic operations performed on data: Create (`INSERT`), Read (`SELECT`), Update (`UPDATE`), and Delete (`DELETE`).
- **Aggregate Functions**: Perform calculations on groups of rows, e.g., `COUNT()`, `SUM()`, `AVG()`, `MIN()`, `MAX()`.
- **GROUP BY**: Groups rows that have the same values in specified columns and aggregates them with functions like `COUNT()`, `SUM()`, etc.
- **HAVING**: Filters the results of a `GROUP BY` query based on aggregate functions.
