# 5. Data Query Language:

DQL is used to retrieve or fetch data from the tables (Database).

**Statements in DQL:**

- Projection
- Selection
- Join

## 1. Projection:

- ➢ The process of fetching or retrieving the data from the table by choosing only columns is known as Projection.
- ➢ We cannot restrict Rows in Projection.
- ➢ The statement used for projection is “SELECT”.

**Syntax of Projection:**

```
Select Column_Name
From Table_Name;
```

**Example:**

1. Write a query to display all the details of the Employee.

```
Select *
From employee;
```

2. WAQTD Names of all the Employees.

```
Select EmpName
From employee;
```

3. WAQTD Employee number and Names of Employees.

```
Select EmpNo, EmpName
From employees;
```

## Order of Execution:

**Syntax:**

```
Select Column_Name
From Table_Name;
```

- The **“From”** clause starts the execution first, where it fetches the entire table for execution.
- The job of the **“From”** clause is to, go to the database search for the table, and put the table under execution.
- The **“Select’’** clause executes next to fetch the specified columns that are provided.

## Characteristics of “From” Clause

- In the “From” clause we specify the name of the table as an argument.
- The “From” Clause, points to the table that has to be executed.

## Characteristics of “Select” Clause

- The “Select” Clause is responsible to provide the output in the form of tables.
- In the “Select” Clause we can provide *, Column_Name, distinct, expression, alias name as arguments.
- The “Select” clause will execute row by row or record by record.

## Alias Name:

It is an alternative name or temporary name given to a column or expression in the output (result table).

- Whenever we give Alias name, it does not change the actual column name in the table.

**Syntax:**

```
Select Column_Name as Column
From Table_Name;
```

**Note:** "**as**" is a **keyword** to provide the **Alias name** but is optional that is we can also provide Alias name without using "**as**" keyword.

**Example:**

1. WAQTD Salary of the employee as "Earning" in the output.

```
Select Salary as Earning
From employee;
```

2. WAQTD Commission of all the employees where the commission has to be represented as "Incentives". Select Commission as Incentives

```
From employee;
```

## Distinct Clause:

When we want to fetch unique values from a particular column by removing duplicates we go for Distinct.

**Syntax:**

```
Select distinct Column_Name
From Table_Name;
```

- Distinct has to be the first argument to Select clause, that is we cannot provide a column name before Distinct keyword.
- If multiple columns are provided after distinct, it does not fetch unique values of the columns separately. Instead it combines the columns and then checks for the duplicate combinations.

## Order-by-Clause:

An Order By Clause is used to fetch output in ascending or descending order.

- By default order by clause fetches values in ascending order. So, the "**asc**" keyword is optional.
- If we want to sort the output in descending order, the "desc" keyword must be used.
- Order by Clause must be the last statement in the query.

**Syntax of Order By Clause:**

```
Select Column_Name
From Table_Name
Order by Column_Name/Expression/Alias Name desc;
```

**Example:**

1. WAQTD Employees Salary in Descending Order

```
Select Salary
From employee
Order by Salary desc;
```

2. WAQTD Salary of all the employees in descending order by removing duplicate salary.

```
Select distinct Salary
From employee
Order by Salary desc;
```

## Expressions:

- An expression is a statement which will generate result or output.
- Expression is a combination of **Operands** and **Operators.**

## What are Operands?

Operands is a value and classified into two types

### 1. Direct Value / Immediate values:

**Example:** 10 + 20 [ In this example 10 and 20 are Direct values

Operands and " + " is a Operator]

**2. Variable / Identifier:** In SQL variable / identifier are column name.

**Example:** Select Salary/2

from employee;

[ In this example Salary is Column name and "/" is operator.

## What are Operators:

- Whenever we want to write multiple conditions in a single where clause then we go for operators.
- We will execute only a single record at one shot.

### Classification of Operators:

Operators are classified into **four** types such as:

1. Arithematic Operators (+, -, *, /)
2. Relational Operators (< , >, <=, >=, !=)
3. Special Operators (IN, NOT IN, BETWEEN, NOT BETWEEN, LIKE, NOT LIKE, IS, IS NOT)
4. Logical Operator (AND, OR, NOT)

## Where Clause:

- ➢ Where Clause is used to filter the records.
- ➢ Where Clause executes row by row.
- ➢ Where Clause executes after the execution of From clause.

## Example:

1. WAQTD Salary who are earning more than 2000.

   ```
   Select Salary
   From employee
   Where Salary>2000;
   ```

2. WAQTD Commission who are earning 500.

   ```
   Select Commission
   From employee
   Where Commission=500;
   ```