SQL WHERE Clause

The WHERE clause in MySQL serves to filter the results of your database queries. It specifies conditions that the data in your table must meet for it to be included in the output. This powerful tool enhances the precision and efficiency of data retrieval.

It works in conjunction with commands like SELECT, UPDATE, and DELETE to specify which rows you want to work with. Here's how it works:

- **After FROM:** The WHERE clause comes after you declare which table you're working with (FROM employees).
- **Conditions:** Inside the WHERE clause, you provide conditions. These are expressions that evaluate to True, False, or Unknown.
- **Filtering:** MySQL examines each row in your table. Only rows where the WHERE condition is True are affected by your query.

A. WHERE with SELECT

- **Purpose:** Filters which rows are returned from your table.
- **Placement:** Comes after the FROM table_name part of your query.

Example 01:

```
SELECT * FROM employees
WHERE department = 'HR';
```

This retrieves all employees in the 'HR' department.

B. WHERE with UPDATE

- **Purpose:** Pinpoints which rows should be modified.
- **Placement:** After the SET column = value portion of the query.

Example 02:

```
UPDATE employees
SET department = 'Finance'
WHERE job_title = 'Accountant';
```

This would change the department to 'Finance' only for employees with the job title 'Accountant'. Without a WHERE clause, this update would affect all employees.

C. WHERE with DELETE

- **Purpose:** Targets specific rows for removal.
- **Placement:** Follows the FROM table_name part of the query.

Example 03:

```
DELETE FROM employees
WHERE start_date < '2020-01-01';
```

This would delete employee records whose start date is before January 1st, 2020. Omitting the WHERE clause would delete all records in the table.

MySQL WHERE Clause Examples with Comparison Operators:

1. Equal (=):

```
SELECT * FROM employees
WHERE department = 'Engineering';
```

This retrieves all employees in the 'Engineering' department.

2. Greater than (>):

```
SELECT * FROM employees
WHERE salary > 50000;
```

This lists employees who earn more than 50,000

3. Less than (<):

```
SELECT * FROM employees
WHERE experience_years < 5;
```

(Assuming an 'experience_years' column exists)

This retrieves employees with less than 5 years of experience.

4. Greater than or equal to (>=):

```
SELECT * FROM employees
WHERE bonus >= 1000;
```

(Assuming a 'bonus' column exists)

This shows employees receiving a bonus of 1,000 or more.

5. Less than or equal to (<=):

```
SELECT * FROM employees
WHERE age <= 30;
```

(Assuming an 'age' column exists)

This lists employees 30 years old or younger.

6. Not equal to (!=):

```
SELECT * FROM employees
WHERE job_title != 'Manager';
```

This retrieves employees who are not Managers.

7. Between (BETWEEN):

```
SELECT * FROM employees
WHERE hire_date BETWEEN '2023-01-01' AND '2023-12-31';
```

(Assuming a 'hire_date' column exists)

This shows employees hired between January 1st, 2023, and December 31st, 2023 (inclusive).

8. Combining Conditions (AND):

```
SELECT * FROM employees
WHERE department = 'Sales' AND salary >= 50000;
```

Retrieves employees in 'Sales' earning 50000 or above.

9. Using OR

```
SELECT * FROM employees
WHERE job_title = 'Developer' OR job_title = 'Analyst';
```

Selects employees who are either 'Developer' or 'Analyst'.