Here's a comprehensive overview of various operators in MySQL, including how to use them for retrieving records in a database.

1. Comparison Operators

Comparison operators are used to compare two values. The result of a comparison can be either true or false.

Operator	Description	Example
`=`	Equal to	`SELECT * FROM emp WHERE salary = 5000;`
`!=` or `<>`	Not equal to	`SELECT * FROM emp WHERE salary != 5000;`
`>`	Greater than	`SELECT * FROM emp WHERE salary > 5000;`
`<`	Less than	`SELECT * FROM emp WHERE salary < 5000;`
`>=`	Greater than or equal to	`SELECT * FROM emp WHERE salary >= 5000;`
`<=`	Less than or equal to	`SELECT * FROM emp WHERE salary <= 5000;`

2. Logical Operators

Logical operators are used to combine multiple conditions in a query.

Operator	Description	Example
`AND`	True if both conditions are true	`SELECT * FROM emp WHERE salary > 5000 AND hire_date > '2023-01-01';`
`OR`	True if at least one condition is true	`SELECT * FROM emp WHERE salary < 3000 OR department = 'HR';`
`NOT`	True if the condition is false	`SELECT * FROM emp WHERE NOT department = 'HR';`

3. LIKE and NOT LIKE Operators

The `LIKE` operator is used for pattern matching with wildcard characters.

Operator	Description	Example
`LIKE`	Searches for a specified pattern	`SELECT * FROM emp WHERE name LIKE 'A%';` (Names starting with A)
`NOT LIKE`	Searches for a pattern not matching	`SELECT * FROM emp WHERE name NOT LIKE 'A%';` (Names not starting with A)

Wildcards:

- `%` - Represents zero or more characters.
- `_` - Represents a single character.

4. IN and NOT IN Operators

The `IN` operator is used to filter records based on a list of values.

Operator	Description	Example
`IN`	Matches any value in a list	`SELECT * FROM emp WHERE department IN ('HR', 'IT');`
`NOT IN`	Excludes values in a list	`SELECT * FROM emp WHERE department NOT IN ('HR', 'IT');`

5. BETWEEN and NOT BETWEEN Operators

The `BETWEEN` operator is used to filter records within a specific range.

Operator	Description	Example
`BETWEEN`	Includes both boundary values	`SELECT * FROM emp WHERE salary BETWEEN 3000 AND 5000;`
`NOT BETWEEN`	Excludes both boundary values	`SELECT * FROM emp WHERE salary NOT BETWEEN 3000 AND 5000;`

6. ORDER BY Operator

The `ORDER BY` clause is used to sort the result set of a query.

Operator	Description	Example
`ORDER BY`	Sorts the result set	`SELECT * FROM emp ORDER BY salary DESC;` (Descending)
		`SELECT * FROM emp ORDER BY name ASC;` (Ascending)

7. IS NULL and IS NOT NULL Operators

These operators are used to test for null values in a column.

Operator	Description	Example
`IS NULL`	True if the value is null	`SELECT * FROM emp WHERE hire_date IS NULL;`
`IS NOT NULL`	True if the value is not null	`SELECT * FROM emp WHERE hire_date IS NOT NULL;`

8. LIMIT Operator

The `LIMIT` clause is used to specify the maximum number of records to return.

Operator	Description	Example
`LIMIT`	Limits the number of records returned	`SELECT * FROM emp LIMIT 5;` (Returns the first 5 records)

Summary

These operators are essential for retrieving and manipulating data in MySQL. By understanding how to effectively use comparison, logical, and other operators, you can

write powerful SQL queries that meet your data retrieval needs.

Certainly! The `LIKE` and `NOT LIKE` operators in MySQL are useful for performing pattern matching in string data. Here's a more detailed explanation along with various examples to demonstrate their usage.

LIKE Operator

The `LIKE` operator is used to search for a specified pattern in a column. It allows the use of wildcard characters:

- `%` - Represents zero or more characters.
- `_` - Represents a single character.

Examples of LIKE Operator

1. **Basic Pattern Matching**
   - **Example**: Find employees whose names start with the letter 'J'.
     ```sql
     SELECT * FROM emp WHERE name LIKE 'J%';
     ```
2. **Pattern Matching with Wildcard at the End**
   - **Example**: Find employees whose names end with 'son'.
     ```sql
     SELECT * FROM emp WHERE name LIKE '%son';
     ```
3. **Pattern Matching with Wildcard in the Middle**
   - **Example**: Find employees whose names contain 'an' anywhere in the name.
     ```sql
     SELECT * FROM emp WHERE name LIKE '%an%';
     ```
4. **Using Underscore for Single Character Match**
   - **Example**: Find employees whose names have 'a' as the second character.
     ```sql
     SELECT * FROM emp WHERE name LIKE '_a%';
     ```
5. **Case Sensitivity**
   - By default, `LIKE` is case-insensitive in MySQL.
   - **Example**: Find employees whose names start with either 'A' or 'a'.
     ```sql
     SELECT * FROM emp WHERE name LIKE 'A%' OR name LIKE 'a%';
     ```
6. **Multiple Wildcards**
   - **Example**: Find employees whose names start with any character followed by 'an' and end with any character.
     ```sql
     SELECT * FROM emp WHERE name LIKE '_an_';
     ```

NOT LIKE Operator

The `NOT LIKE` operator is used to exclude records that match a specified pattern.

Examples of NOT LIKE Operator

1. **Basic Exclusion Pattern Matching**

- **Example**: Find employees whose names do not start with the letter 'A'.

```
SELECT * FROM emp WHERE name NOT LIKE 'A%';
```

2. **Excluding Names Ending with a Specific Suffix**

- **Example**: Find employees whose names do not end with 'y'.

```
SELECT * FROM emp WHERE name NOT LIKE '%y';
```

3. **Excluding Names Containing a Specific Substring**

- **Example**: Find employees whose names do not contain 'smith'.

```
SELECT * FROM emp WHERE name NOT LIKE '%smith%';
```

4. **Using Underscore for Exclusion of Single Character Match**

- **Example**: Find employees whose names do not have 'a' as the second character.

```
SELECT * FROM emp WHERE name NOT LIKE '_a%';
```

5. **Excluding Multiple Patterns**

- **Example**: Find employees whose names do not start with 'A' or 'B'.

```
SELECT * FROM emp WHERE name NOT LIKE 'A%' AND name NOT
LIKE 'B%';
```

Summary

- **LIKE** is useful for finding records that match a specific pattern, with wildcards `%` and `_` allowing flexible search criteria.
- **NOT LIKE** is used to filter out records that match a pattern, providing a way to exclude specific data.

By utilizing `LIKE` and `NOT LIKE`, you can perform complex queries that enhance your data retrieval capabilities in MySQL.

In [ ]: