SQL Operators and Single-Row Functions in MySQL

1. SQL Operators

1.1 LIKE Operator

The LIKE operator is used to search for a specified pattern in a column. It is typically used with the WHERE clause for filtering data.

Syntax:

```
SELECT * FROM table_name WHERE column_name LIKE 'pattern';
```

Wildcards:

- %: Represents zero, one, or multiple characters.
- _: Represents a single character.

Examples:

```
-- Find names starting with 'A'
SELECT * FROM students WHERE name LIKE 'A%';

-- Find names containing 'son'
SELECT * FROM employees WHERE last_name LIKE '%son%';

-- Find names with exactly five characters
SELECT * FROM students WHERE name LIKE '_____';
```

1.2 IS NULL Operator

The IS NULL operator is used to check for NULL values in a column. NULL means missing or undefined data.

Syntax:

```
SELECT * FROM table_name WHERE column_name IS NULL;
```

Example:

```
-- Find employees with missing phone numbers
SELECT * FROM employees WHERE phone_number IS NULL;
```

1.3 IN Operator

The IN operator allows you to specify multiple values in a WHERE clause. It checks if a value is in a list of values.

Syntax:

```
SELECT * FROM table_name WHERE column_name IN (value1, value2, ...);
```

Example:

```
-- Find students from specific countries
SELECT * FROM students WHERE country IN ('USA', 'Canada', 'UK');
```

---

2. Single-Row Functions

2.1 DATE_FORMAT() and CAST()

DATE_FORMAT() Function:

The DATE_FORMAT() function is used to format date values into a specified format. It converts date or datetime values into a string in various formats.

Syntax:

```
DATE_FORMAT(date, format)
```

Example:

```
SELECT DATE_FORMAT(NOW(), '%Y-%m-%d') AS formatted_date;
```

CAST() Function:

The CAST() function is used to convert one data type to another. It is often used to convert numbers or dates to strings.

Syntax:

```
CAST(expression AS data_type)
```

Example:

```
-- Convert a number to a string
SELECT CAST(1234 AS CHAR) AS number_as_string;
```

2.2 CONCAT() Function

The CONCAT() function is used to join two or more strings together.

Syntax:

```
CONCAT(string1, string2, ...)
```

Example:

```
SELECT CONCAT(first_name, last_name) AS full_name FROM employees;
```

2.3 SUBSTR() Function

The SUBSTR() function extracts a part of a string. It's useful for extracting characters from a string starting at a specific position.

Syntax:

```
SUBSTR(string, start_position, length)
```

Example:

```
-- Extract first three characters from the name
SELECT SUBSTR(name, 1, 3) AS short_name FROM students;

-- Extract characters starting from position 7 to the end
SELECT SUBSTR('Hello World', 7) AS remaining_string;

-- Start extracting from the 3rd character from the end
SELECT SUBSTR('Hello World', -3) AS last_three_chars;
```

2.4 INSTR() Function

The INSTR() function returns the position of the first occurrence of a substring in a string.

Syntax:

```
INSTR(string, substring)
```

Example:

```
-- Find the position of 'son' in last_name
SELECT INSTR(last_name, 'son') AS position FROM employees;
```

2.5. Conditional Functions IFNULL(), IF(), and NULLIF()

IFNULL() Function:

The IFNULL() function returns the first argument if it is not NULL, otherwise, it returns the second argument.

Syntax:

```
IFNULL(expression, replacement_value)
```

Example:

```
SELECT IFNULL(phone_number, 'N/A') AS contact_info FROM employees;
```

IF() Function:

The IF() function is used to perform conditional logic, returning one value if a condition is true and another value if it’s false.

Syntax:

```
IF(condition, true_value, false_value)
```

Example:

```
SELECT IF(age >= 18, 'Adult', 'Minor') AS age_category FROM users;
```

NULLIF() Function:

The NULLIF() function returns NULL if the two arguments are equal, otherwise, it returns the first argument.

Syntax:

```
NULLIF(expression1, expression2)
```

Example:

```
SELECT NULLIF(salary, 0) AS adjusted_salary FROM employees;
```

Overview:

Category	Function/Operator	Example	Output
SQL Operators	**LIKE**	`SELECT * FROM books WHERE title LIKE '%adventure%';`	Finds titles containing "adventure"
SQL Operators	**IS NULL**	`SELECT * FROM customers WHERE address IS NULL;`	Finds customers with missing addresses
SQL Operators	**IN**	`SELECT * FROM products WHERE category IN ('Electronics', 'Toys');`	Finds products in Electronics or Toys
Single-Row Functions	**DATE_FORMAT()**	`SELECT DATE_FORMAT(birthdate, '%M %d, %Y') FROM employees;`	Formats birthdate as "Month Day, Year"
Single-Row Functions	**CAST()**	`SELECT CAST(price AS CHAR) FROM items;`	Converts `price` to a string
Single-Row Functions	**CONCAT()**	`SELECT CONCAT(city, ', ', state) FROM locations;`	Concatenates city and state with a comma
Single-Row Functions	**SUBSTR()**	`SELECT SUBSTR(description, 5, 6) FROM articles;`	Extracts 6 characters starting from position 5
Single-Row Functions	**INSTR()**	`SELECT INSTR(email, '@') FROM users;`	Finds the position of '@' in email addresses
Conditional Functions	**IFNULL()**	`SELECT IFNULL(manager, 'No Manager') FROM employees;`	Replaces NULL manager values with 'No Manager'
Conditional Functions	**IF()**	`SELECT IF(salary > 5000, 'High', 'Low') FROM staff;`	Categorizes salary as 'High' or 'Low'
Conditional Functions	**NULLIF()**	`SELECT NULLIF(commission, 0) FROM sales;`	Returns NULL if commission is 0