SQL - Operators

What is SQL Operator?

An SQL operator is a reserved word or a character used primarily in an SQL statement's WHERE clause to perform operation(s), such as comparisons and arithmetic operations. These Operators are used to specify conditions in an SQL statement and to serve as conjunctions for multiple conditions in a statement.

An SQL operator can be either a **unary** or **binary operator**. A unary operator (example unary + or unary -) uses only one operand to perform the unary operation, whereas the binary operator (example + or - etc) uses two operands to perform the binary operation.

Types of Operator in SQL

SQL supports following types of operators:

- Arithmetic operators
- Comparison operators
- Logical operators
- Operators used to negate conditions

SQL Arithmetic Operators

SQL Arithmetic Operators are used to perform mathematical operations on the numerical values. SQL provides following operators to perform mathematical operations.

Here is a list of all the arithmetic operators available in SQL.

Operator	Description	Example
+	Addition	10 + 20 = 30
-	Subtraction	20 - 30 = -10
*	Multiplication	10 * 20 = 200
/	Division	20 / 10 = 2

[redacted]

SQL Comparison Operators test whether two given expressions are the same or not. These operators are used in SQL conditional statements while comparing one expression with another and they return a **Boolean** value which can be either TRUE or FALSE. The result of an SQL comparison operation can be UNKNOWN when one or another operand has it's value as NULL.

Here is a list of all the comparison operators available in SQL.

Operator	Description	Example
=	Equal to	5 = 5 returns TRUE
!=	Not equal	5 != 6 returns TRUE
<>	Not equal	5 <> 4 returns TRUE
>	Greater than	4 > 5 returns FALSE
<	Less than	4 < 5 returns TRUE
>=	Greater than or equal to	4 >= 5 returns FALSE
<=	Less than or equal to	4 <= 5 returns TRUE
!<	Not less than	4 !< 5 returns FALSE
!>	Not greater than	4 !> 5 returns TRUE

SQL Logical Operators

SQL Logical Operators are very similar to comparison operators and they test for the truth of some given condition. These operators return a **Boolean** value which can be either a TRUE or FALSE. The result of an SQL logical operation can be UNKNOWN when one or another operand has it's value as NULL.

Here is a list of all the logical operators available in SQL.

Operator	Description	Example
ALL	TRUE if all of a set of comparisons are TRUE.	Example
AND	TRUE if all the conditions separated by AND are TRUE.	**Example**
ANY	TRUE if any one of a set of comparisons are TRUE.	Example

EXISTS	TRUE if the subquery returns one or more records	**Example**
IN	TRUE if the operand is equal to one of a list of expressions.	**Example**
LIKE	TRUE if the operand matches a pattern specially with wildcard.	**Example**
NOT	Reverses the value of any other Boolean operator.	**Example**
OR	TRUE if any of the conditions separated by OR is TRUE	**Example**
IS NULL	TRUE if the expression value is NULL.	**Example**
SOME	TRUE if some of a set of comparisons are TRUE.	Example
UNIQUE	The UNIQUE operator searches every row of a specified table for uniqueness (no duplicates).	Example

Learn **SQL** in-depth with real-world projects through our **SQL certification course**. Enroll and become a certified expert to boost your career.

SQL Operator Precedence

The operator precedence in SQL is the sequence in which the SQL evaluates the different operators in a given expression. The operators with higher precedence get evaluated first.

Following table lists all SQL operators as per their precedence. The operators with the highest precedence are at the top and the operators with the lowest precedence are at the bottom.

Operator	**Operation**
+, -	identity, negation
*, /	multiplication, division
+, -	addition, subtraction
=, !=, <, >, <=, >=, IS NULL, LIKE, BETWEEN, IN	Comparison
NOT	logical negation

Example

Consider the following SQL statement:

Open Compiler

```
SELECT 20 - 3 * 5;
```

This will result in the following. Here multiplication operator gets evaluated first and then subtraction happens.

```
5
```