

Types of SQL operators

Arithmetic

Arithmetic

Performs math operation on numerical data, we can find addition, subtraction, multiplication, division and modulus.

Comparison

Comparison

Compares two different data returning a boolean value (TRUE or FALSE), checking if equal, greater or lesser.

Logical

Logical

Creates conditional expressions that returns a boolean value (TRUE or FALSE). We can find ALL, AND, ANY, BETWEEN, EXISTS, IN, LIKE, NOT, OR, IS NULL.

Set

Set

Combines similar type of data from tables mixing the result of queries and returning a single result.

Its operators are UNION, UNION ALL, MINUS, INTERSECT.

Logical Operators

The most used logical operators are : AND, OR, NOT.
These can be found everywhere within Information Technology.

Logical Operators

AND

Checks if all the conditions of the clause are true, retrieving the row(s) in the result set.

.. WHERE CONDITION1 **AND** CONDITION2 ... **AND** CONDITIONN;

Logical Operators

AND EXAMPLE

```
SELECT * FROM employee  
WHERE 1 = 1 AND first_name = 'Manuel';
```

← Both conditions are
TRUE, so 1 record will
be shown

```
DELETE FROM employee  
WHERE last_name = 'Gentile' AND  
first_name = 'Joe';
```

← Only first condition is
TRUE, so no records will
be deleted

Logical Operators

OR

Checks if at least one of the conditions of the clause are true, retrieving the row(s) in the result set.

.. WHERE CONDITION1 **OR** CONDITION2 ... **OR** CONDITIONN;

Logical Operators

OR EXAMPLE

```
SELECT * FROM employee  
WHERE 1 = 2 OR first_name = 'Manuel';
```

← Only the second condition is TRUE, so 1 record will be shown

```
DELETE FROM employee  
WHERE last_name = 'Gentile' OR  
first_name = 'Joe';
```

← Both conditions are TRUE, so 2 records will be deleted

Logical Operators

NOT

It's used to negate a condition, so it returns false if the condition is true whilst it returns true if the condition is false

```
.. WHERE NOT CONDITION1;
```

Logical Operators

NOT EXAMPLE

```
SELECT * FROM employee  
WHERE NOT (first_name = 'Manuel');
```

← Only 1 record will be shown

```
UPDATE product SET label = 'useless'  
WHERE NOT (price > 0,01 OR  
category='music');
```

← All products with a price smaller than 0,01 AND a category different from music will be labelled as useless

Logical Operators

ALL

The ALL operator is used to compare a column or a value to a set of values returned by a subquery.

It must be preceded by a comparison operators (=, >, >=, <, <=, <>) and it checks if the condition is true for all the values in the result set of the mentioned subquery.

.. WHERE COLUMN1 COMPARISON_OPERATOR **ALL** (subquery);

Logical Operators

ALL EXAMPLE

```
SELECT * FROM product
WHERE price > ALL (
    SELECT price FROM product WHERE
    category='music' );
```

← This will extract all the products having a price greater than the biggest music product value

```
SELECT * FROM product
WHERE price < ALL (
    SELECT price FROM product WHERE
    category='music' );
```

← This will extract all the products having a price less than the smallest music product value

Logical Operators

ANY

The ANY operator is used to compare a column or a value to a set of values returned by a subquery.

It must be preceded by a comparison operators (=, >, >=, <, <=, <>) and it checks if the condition is true for any of the values in the result set of the mentioned subquery.

.. WHERE COLUMN1 COMPARISON_OPERATOR **ANY** (subquery);

Logical Operators

ANY EXAMPLE

```
SELECT * FROM product
WHERE price > ANY (
    SELECT price FROM product WHERE
    category='music' );
```

← This will extract all the products having a price greater than the smallest music product value

```
SELECT * FROM product
WHERE price < ANY (
    SELECT price FROM product WHERE
    category='music' );
```

← This will extract all the products having a price less than the greatest music product value

Logical Operators

BETWEEN

The BETWEEN operator is used to compare a column or a value to a range of values.

It can be used with numerics, chars and dates.

.. WHERE COLUMN1 **BETWEEN** VALUE1 **AND** VALUE2;

Logical Operators

BETWEEN EXAMPLE

```
SELECT * FROM product  
WHERE price BETWEEN 1 AND 10;
```

← This will extract all the products having a price greater than 1 AND smaller than 10

```
DELETE FROM employee  
WHERE first_name BETWEEN 'A' AND 'N';
```

← This will delete all the employees having a letter of name starting from A to N

Logical Operators

EXISTS

The EXISTS operator checks the existence or not of a column or a value to a result set of rows of a subquery.

.. WHERE COLUMN **EXISTS** (subquery);

Logical Operators

EXISTS EXAMPLE

```
SELECT * FROM employee  
WHERE EXISTS (  
    SELECT * FROM country);
```

← The country table is empty, so no records will be extracted from employee

```
DELETE FROM employee  
WHERE NOT EXISTS (  
    SELECT * FROM country);
```

← The country table is empty, so all records will be deleted from employee

Logical Operators

IN

The IN operator is used to compare a column or a value to a set of values returned by a subquery or a comma separated list. Retrieves the rows in case of matching the value in the set. It works like having different OR conditions.

.. WHERE COLUMN **IN** (subquery | list);

corresponds to ↓

.. WHERE COLUMN = value1 OR ... OR COLUMN = valueN;

Logical Operators

IN EXAMPLE

```
SELECT * FROM employee  
WHERE first_name IN ('Manuel','Joe');
```

← This will extract all the employees having first name equals to Manuel OR Joe

```
SELECT * FROM product  
WHERE price IN (1,5,10,100);
```

← This will extract all the products having a price equal to the values of the list

Logical Operators

LIKE

The LIKE operator is used to compare a CHAR or VARCHAR data type column to another one, a quoted string or a pattern. It allows to use wildcard characters as %, _, [], [^]

.. WHERE COLUMN **LIKE** pattern;

Logical Operators

LIKE WILDCARD

Wildcard	Description
%	It represent a sequence of 0 or more chars
_	It represent a single char
[charlist]	It represents any single char within a charlist
[^charlist] or [!charlist]	It represents any single char other than the charlist

Logical Operators

LIKE EXAMPLE

```
SELECT * FROM employee  
WHERE first_name LIKE 'M%';
```

← This will extract all the employees which first name starts with M

```
DELETE FROM employee  
WHERE first_name LIKE '%X%';
```

← This will delete all the employees which first name contains an X

Logical Operators

IS NULL

The IS NULL operator is used to compare whether a column has a null value (TRUE) or not (FALSE).

```
.. WHERE COLUMN IS NULL;
```

Logical Operators

IS NULL EXAMPLE

```
SELECT * FROM product  
WHERE price IS NOT NULL;
```

← This will extract all the products having a price set

```
DELETE FROM product  
WHERE price IS NULL;
```

← This will delete all the products having a price not set

Let's connect

If you want to learn more about the topic, connect or send me a DM.

