totn PostgreSQL

PostgreSQL: IN Condition

This PostgreSQL tutorial explains how to use the PostgreSQL **IN condition** with syntax and examples.

Description

The PostgreSQL IN condition is used to help reduce the need to use multiple OR conditions in a SELECT, INSERT, UPDATE, or DELETE statement.

Syntax

The syntax for the IN condition in PostgreSQL is:

```
expression IN (value1, value2, .... value_n);
```

OR

```
expression IN (subquery);
```

Parameters or Arguments

expression
A value to test.

value1, value2..., or value_n
The values to test against *expression*.

subquery
This is a SELECT statement whose result set will be tested against *expression*. If any of these values matches *expression*, then the IN condition will evaluate to true.

Note

- The PostgreSQL IN condition will return the records where *expression* is *value1*, *value2*..., or *value_n*.
- The PostgreSQL IN condition is also called the PostgreSQL IN operator.

Example - With Character

Let's look at a PostgreSQL IN condition example using character values.

The following is a PostgreSQL SELECT statement that uses the IN condition to compare character values:

```
SELECT *
FROM suppliers
WHERE supplier_name IN ('Apple', 'Samsung', 'RIM');
```

This PostgreSQL IN condition example would return all rows from the *suppliers* table where the *supplier_name* is either 'Apple', 'Samsung' or 'Asus'. Because the * is used in the SELECT, all fields from the *suppliers* table would appear in the result set.

The above IN example is equivalent to the following SELECT statement:

```
SELECT *
FROM suppliers
WHERE supplier_name = 'Apple'
OR supplier_name = 'Samsung'
OR supplier_name = 'RIM';
```

As you can see, using the PostgreSQL IN condition makes the statement easier to read and more efficient.

Example - With Numeric

Next, let's look at a PostgreSQL IN condition example using numeric values.

For example:

```
SELECT *
FROM employees
WHERE employee_id IN (300, 301, 500, 501);
```

This PostgreSQL IN condition example would return all employees where the *employee_id* is either 300, 301, 500, or 501.

The above IN example is equivalent to the following SELECT statement:

```
SELECT *
FROM employees
WHERE employee_id = 300
OR employee_id = 301
OR employee_id = 500
OR employee_id = 501;
```

Example - Using NOT operator

Finally, let's look at an IN condition example using the NOT operator.

For example:

```
SELECT *
FROM suppliers
WHERE supplier_name NOT IN ('Apple', 'Samsung', 'RIM');
```

This PostgreSQL IN condition example would return all rows from the *suppliers* table where the *supplier_name* is **not** 'Apple', 'Samsung', or 'RIM'. Sometimes, it is more efficient to list the values that you do **not** want, as opposed to the values that you do want.