SQL for Citizen Data Scientists

SQL IN Operator - Learn By Example

SETScholars & WACAMLDS

The IN operator allows you to test whether a value falls within a set of values. A set can be a list of literal values, or the result of a subquery.

Syntax

The IN operator must be followed by a comma-delimited list of valid values enclosed within parentheses.

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

Sample Table

To help you better understand the examples, and enable you to follow along with the tutorial, we are going to use the following sample table.

This table is part of an ‘Employee Management System’ that contains basic information about employees.

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	60000
2	Kim	25	Chicago	Manager	55000
3	Eve	24	New York	Developer	32000
4	Joe	23	Chicago	Developer	30000
5	Max	26	New York	Janitor	9000
6	Sam	27	Chicago	Janitor	10000

The IN Operator

To use the IN operator specify a comma-delimited list of valid values in parentheses. For example, to select all employees who are either 28, 23 or 26 years old, you can issue:

```
SELECT *
FROM Employees
WHERE Age IN (28, 23, 26);
```

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	60000
3	Max	26	New York	Janitor	9000
5	Joe	23	Chicago	Developer	30000

You can use the IN operator with data types other than numbers, such as text. For example, the following query will return a list of all employees who are either ‘Managers’, ‘Developers’ or ‘Supervisors’

```
SELECT *
FROM Employees
WHERE Job IN ('Manager', 'Developer', 'Supervisor');
```

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	60000
2	Eve	24	New York	Developer	32000
4	Kim	25	Chicago	Manager	55000
5	Joe	23	Chicago	Developer	30000

As you can see there is no employee with a job title supervisor, so no record associated with it has been selected.

IN operator Vs OR operator

If you are thinking that the IN operator behaves as if many conditions were joined together with OR, then you are correct. The IN operator is a shorthand for multiple OR conditions.

The following SQL statement works exactly like the example above; it selects all employees who are either 28, 23 or 26 years old.

```
SELECT *
FROM Employees
WHERE Age = 28
   OR Age = 23
   OR Age = 26;
```

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	60000
3	Max	26	New York	Janitor	9000
5	Joe	23	Chicago	Developer	30000

Why use the IN operator if you can get the same result by OR'ing multiple conditions together? This is because:

1. The IN operator always executes more quickly than multiple OR operators (when the list is large).
2. When the list of valid values is long, the IN operator syntax is very clean and easy to read.
3. When IN is used in combination with other AND and OR operators, the order of evaluation is easier to maintain.
4. The biggest advantage of IN is that it can contain a subquery, enabling you to generate a set for you on the fly.

NOT IN Operator

By combining the IN operator with the NOT operator, you can select values that are not in the specified set. For example, if you wanted to select all employees who are neither 28, 23 nor 26 years old (the opposite of the previous example), you could run this query:

```
SELECT *
FROM Employees
WHERE Age NOT IN (28, 23, 26);
```

ID	Name	Age	City	Job	Salary
2	Eve	24	New York	Developer	32000
4	Kim	25	Chicago	Manager	55000
6	Sam	27	Chicago	Janitor	10000

Retrieving Values from One Table That Exist in Another

The biggest advantage of IN is that the IN operator can contain a subquery, enabling you to generate a set for you on the fly. You can use the following syntax:

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (subquery);
```

Suppose the ‘Client’ table looks like this:

ID	Name	City
1	Nick	Chicago
2	Anna	Seattle

And you wanted to select all employees who are from the same city as the clients, then you could write a query like:

```
SELECT *
FROM Employees
WHERE City IN (SELECT City
            FROM Client);
```

ID	Name	Age	City	Job	Salary
4	Kim	25	Chicago	Manager	55000
5	Joe	23	Chicago	Developer	30000
6	Sam	27	Chicago	Janitor	10000

The subquery returns a set of three values. The IN operator tests the cities from the ‘Employees’ table to see if they are in the list of clients’ cities returned by the subquery. If there is a match that row will be included in the result-set.