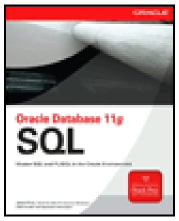

Chapter 2 - Retrieving Information from Database Tables
Oracle Database 11g SQL
by Jason Price
Oracle Press © 2008 *Citation*

Recommend? yes no

Previous Next

Using the SQL Operators

The SQL operators allow you to limit rows based on pattern matching of strings, lists of values, ranges of values, and null values. The SQL operators are listed in the following table:

Open table as spreadsheet

Operator	Description
`LIKE`	Matches patterns in strings
`IN`	Matches lists of values
`BETWEEN`	Matches a range of values
`IS NULL`	Matches null values
`IS NAN`	Matches the `NAN` special value, which means "not a number" (introduced in Oracle Database 10*g*)
`IS INFINITE`	Matches infinite `BINARY_FLOAT` and `BINARY_DOUBLE` values (introduced in Oracle Database 10*g*)

You can also use `NOT` to reverse the meaning of an operator:

- `NOT LIKE`
- `NOT IN`
- `NOT BETWEEN`
- `IS NOT NULL`
- `IS NOT NAN`
- `IS NOT INFINITE`

You'll learn about the `LIKE`, `IN`, and `BETWEEN` operators in the following sections.

Using the LIKE Operator

You use the `LIKE` operator in a `WHERE` clause to search a string for a pattern. You specify patterns using a combination of normal characters and the following two wildcard characters:

- **Underscore (_)** Matches one character in a specified position
- **Percent (`%`)** Matches any number of characters beginning at the specified position

For example, consider the following pattern:

```
'_o%'
```

The underscore (_) matches any one character in the first position, the `o` matches an *o* character in the second position, and the percent (%) matches any characters following the *o* character. The following query uses the

LIKE operator to search the first_name column of the customers table for the pattern '_o%':

```
SELECT *
FROM customers
WHERE first_name LIKE '_o%';

CUSTOMER_ID FIRST_NAME LAST_NAME  DOB       PHONE
----------- ---------- ---------- --------- ------------
          1 John       Brown      01-JAN-65 800-555-1211
          5 Doreen     Blue       20-MAY-70
```

As you can see from the results, two rows are returned, because the strings John and Doreen both have o as the second character.

The next query uses NOT LIKE to get the rows not retrieved by the previous query:

```
SELECT *
FROM customers
WHERE first_name NOT LIKE '_o%';

CUSTOMER_ID FIRST_NAME LAST_NAME  DOB       PHONE
----------- ---------- ---------- --------- ------------
          2 Cynthia    Green      05-FEB-68 800-555-1212
          3 Steve      White      16-MAR-71 800-555-1213
          4 Gail       Black                800-555-1214
```

If you need to search for actual underscore or percent characters in a string, you can use the ESCAPE option to identify those characters. For example, consider the following pattern:

```
'%\%%' ESCAPE '\'
```

The character after the ESCAPE tells the database how to differentiate between characters to search for and wildcards, and in the example the backslash character (\) is used. The first % is treated as a wildcard and matches any number of characters; the second % is treated as an actual character to search for; the third % is treated as a wildcard and matches any number of characters. The following query uses the promotions table, which contains the details for products being discounted by the store (you'll learn more about this table later in this book). The query uses the LIKE operator to search the name column of the promotions table for the pattern '%\%%' ESCAPE '\':

```
SELECT name
FROM promotions
WHERE name LIKE '%\%%' ESCAPE '\';

NAME
------------------------------
10% off Z Files
20% off Pop 3
30% off Modern Science
20% off Tank War
10% off Chemistry
20% off Creative Yell
15% off My Front Line
```

As you can see from the results, the query returns the rows whose names contain a percentage character.

Using the IN Operator

You may use the IN operator in a WHERE clause to retrieve the rows whose column value is in a list. The following query uses IN to retrieve rows from the customers table where the customer_id is 2, 3, or 5:

```
SELECT *
FROM customers
WHERE customer_id IN (2, 3, 5);

CUSTOMER_ID FIRST_NAME LAST_NAME  DOB       PHONE
----------- ---------- ---------- --------- ------------
          2 Cynthia    Green      05-FEB-68 800-555-1212
          3 Steve      White      16-MAR-71 800-555-1213
          5 Doreen     Blue       20-MAY-70
```

`NOT IN` retrieves the rows not retrieved by `IN`:

```
SELECT *
FROM customers
WHERE customer_id NOT IN (2, 3, 5);

CUSTOMER_ID FIRST_NAME LAST_NAME  DOB       PHONE
----------- ---------- ---------- --------- ------------
          1 John       Brown      01-JAN-65 800-555-1211
          4 Gail       Black                800-555-1214
```

One important point to be aware of is that `NOT IN` returns false if a value in the list is null. This is illustrated by the following query, which doesn't return any rows because null is included in the list:

```
SELECT *
FROM customers
WHERE customer_id NOT IN (2, 3, 5, NULL);

no rows selected
```

Caution `NOT IN` returns false if a value in the list is null. This is important because, since you can use any expression in the list and not just literal values, it can be difficult to spot when a null value occurs. Consider using `NVL()` with expressions that might return a null value.

Using the BETWEEN Operator

You may use the `BETWEEN` operator in a `WHERE` clause to retrieve the rows whose column value is in a specified range. The range is *inclusive*, which means the values at both ends of the range are included. The following query uses `BETWEEN` to retrieve the rows from the `customers` table where the `customer_id` is between 1 and 3:

```
SELECT *
FROM customers
WHERE customer_id BETWEEN 1 AND 3;

CUSTOMER_ID FIRST_NAME LAST_NAME  DOB       PHONE
----------- ---------- ---------- --------- ------------
          1 John       Brown      01-JAN-65 800-555-1211
          2 Cynthia    Green      05-FEB-68 800-555-1212
          3 Steve      White      16-MAR-71 800-555-1213
```

`NOT BETWEEN` retrieves the rows not retrieved by `BETWEEN`:

```
SELECT *
FROM customers
WHERE customer_id NOT BETWEEN 1 AND 3;

CUSTOMER_ID FIRST_NAME LAST_NAME  DOB       PHONE
----------- ---------- ---------- --------- ------------
```

```
        4 Gail       Black                   800-555-1214
        5 Doreen     Blue        20-MAY-70
```

Previous Next