WHERE Clause uses

In this reading, you'll explore the usage of the WHERE clause for filtering data. You've learned about the purpose and the syntax of the WHERE clause. You've also learned how it behaves with different types of operands (namely text-based or numeric) based on the data type of the table column. You explored the types of operators that can be used in the WHERE clause. The main objective of this reading is to present some more examples and scenarios in which the WHERE clause is used to filter data in a table.

The WHERE clause

The WHERE clause is useful when you want to filter data in a table based on a given condition in the SQL statement.The WHERE clause in SQL is there for the purpose of filtering records and fetching only the necessary records. This can be used in SQL SELECT, UPDATE and DELETE statements.
The filtering happens based on a condition. The condition can be written using any of the following comparison or logical operators.

Comparison operators

Operator	Description
=	Checks if the values of two operands are equal or not. If yes, then condition becomes true.
!=	Checks if the values of two operands are equal or not. If values are not equal, then condition becomes true.
<>	Checks if the values of two operands are equal or not. If values are not equal, then condition becomes true.
>	Checks if the value of the left operand is greater than the value of the right operand. If yes, then condition becomes true.
<	Checks if the value of left operand is less than the value of right operand. If yes, then condition becomes true.
>=	Checks if the value of the left operand is greater than or equal to the value of right operand. If yes, then condition becomes true.
<=	Check if the value of the left operand is less than or equal to the value of the right operand. If yes then condition becomes true.
!<	Checks if the value of the left operand is not less than the value of the right operand. If yes, then condition becomes true.
!>	Checks if the value of the left operand is not greater than the value of the right operand. If yes, then condition becomes true.

Logical operators

Operator	Description

ALL	Used to compare a single value to all the values in another value set.
AND	Allows for the existence of multiple conditions in an SQL statement's WHERE clause.
ANY	Used to compare a value to any applicable value in the list as per the condition.
BETWEEN	Used to search for values that are within a set of values, given the minimum value and the maximum value.
EXISTS	Used to search for the presence of a row in a specified table that meets a certain criterion.
IN	Used to compare a value to a list of literal values that have been specified.
LIKE	Used to compare a value to similar values using wildcard operators.
NOT	Reverses the meaning of the logical operator with which it is used. For example: NOT EXISTS, NOT BETWEEN, NOT IN, etc. **This is a negate operator.**
OR	Used to combine multiple conditions in an SQL statement's WHERE clause.
IS NULL	Used to compare a value with a NULL value.
UNIQUE	Searches every row of a specified table for uniqueness (no duplicates).

Using the sample database, let’s review an example that uses the comparison operator > (greater than) to formulate the WHERE clause condition to filter criteria. If you want to fetch the invoices that have a total value of more than $2, you will need to filter out the records in the invoicetable by using the WHERE clause in the SELECT statement. To perform this action, you can run the following query:

```
1
2
3
4
5
```

```
SELECT *

FROM invoices

WHERE Total > 2;
```

Run

Reset

You'll notice that this query filters out the records based on the condition given in the WHERE clause Total > 2. It brings in only the records that have a totalfield value of more than $2. But what if you want to combine multiple conditions in the WHERE clause? Multiple conditions in the WHERE clause can be combined using the AND / OR logical operators. Therefore, these two operators are also known as conjunctive operators. The syntax required to use the AND operator in the WHERE clause of a SELECT statement is as follows:

```
SELECT column1, column2, columnN
```

```
FROM table_name

WHERE [condition1] AND [condition2]...AND [conditionN];
```

N can be any number. Here, for the entire condition to be TRUE, all conditions separated by the AND must be TRUE.
Let's review an example. You need a list of invoices for which the total is over $2 and the BillingCountry is the USA. Here's an example of how the WHERE clause condition can be given in the SELECT statement:

```
1
2
3
4
5
```

```
SELECT *

FROM invoices

WHERE Total > 2 AND BillingCountry = 'USA';
```

Run

Reset

```
+-----------+------------+--------------------
+--------------------------+----------------+--------------
+----------------+-------------------+-------+
| InvoiceId | CustomerId | InvoiceDate         | BillingAddress
| BillingCity    | BillingState | BillingCountry | BillingPostalCode |
Total |
+-----------+------------+--------------------
+--------------------------+----------------+--------------
+----------------+-------------------+-------+
|         5 |         23 | 2009-01-11 00:00:00 | 69 Salem Street
| Boston         | MA           | USA            | 2113              |
13.86 |
|        16 |         21 | 2009-03-05 00:00:00 | 801 W 4th Street
| Reno           | NV           | USA            | 89503             |
3.96 |
|        17 |         25 | 2009-03-06 00:00:00 | 319 N. Frances Street
| Madison        | WI           | USA            | 53703             |
5.94 |
|        26 |         19 | 2009-04-14 00:00:00 | 1 Infinite Loop
| Cupertino      | CA           | USA            | 95014             |
13.86 |
|        37 |         17 | 2009-06-06 00:00:00 | 1 Microsoft Way
| Redmond        | WA           | USA            | 98052-8300        |
3.96 |
|        38 |         21 | 2009-06-07 00:00:00 | 801 W 4th Street
| Reno           | NV           | USA            | 89503             |
5.94 |
|        39 |         27 | 2009-06-10 00:00:00 | 1033 N Park Ave
| Tucson         | AZ           | USA            | 85719             |
8.91 |
|        59 |         17 | 2009-09-08 00:00:00 | 1 Microsoft Way
| Redmond        | WA           | USA            | 98052-8300        |
5.94 |
|        60 |         23 | 2009-09-11 00:00:00 | 69 Salem Street
| Boston         | MA           | USA            | 2113              |
8.91 |
|        81 |         19 | 2009-12-13 00:00:00 | 1 Infinite Loop
```

Cupertino	CA	USA	95014	8.91
82	28	2009-12-18 00:00:00	302 S 700 E	Salt Lake City	UT	USA	84102	13.86
93	26	2010-02-09 00:00:00	2211 W Berry Street	Fort Worth	TX	USA	76110	3.96
103	24	2010-03-21 00:00:00	162 E Superior Street	Chicago	IL	USA	60611	15.86
114	22	2010-05-13 00:00:00	120 S Orange Ave	Orlando	FL	USA	32801	3.96
115	26	2010-05-14 00:00:00	2211 W Berry Street	Fort Worth	TX	USA	76110	5.94
124	20	2010-06-22 00:00:00	541 Del Medio Avenue	Mountain View	CA	USA	94040-111	13.86
135	18	2010-08-14 00:00:00	627 Broadway	New York	NY	USA	10012-2612	3.96
136	22	2010-08-15 00:00:00	120 S Orange Ave	Orlando	FL	USA	32801	5.94
137	28	2010-08-18 00:00:00	302 S 700 E	Salt Lake City	UT	USA	84102	8.91
145	16	2010-09-23 00:00:00	1600 Amphitheatre Parkway	Mountain View	CA	USA	94043-1351	13.86
157	18	2010-11-16 00:00:00	627 Broadway	New York	NY	USA	10012-2612	5.94
158	24	2010-11-19 00:00:00	162 E Superior Street	Chicago	IL	USA	60611	8.91
179	20	2011-02-20 00:00:00	541 Del Medio Avenue	Mountain View	CA	USA	94040-111	8.91
191	27	2011-04-19 00:00:00	1033 N Park Ave	Tucson	AZ	USA	85719	3.96
200	16	2011-05-24 00:00:00	1600 Amphitheatre Parkway	Mountain View	CA	USA	94043-1351	8.91

```
+----------+-----------+--------------------
+-------------------------+---------------+-------------
+---------------+------------------+-------+
(Output limit exceeded, 25 of 54 total rows shown)
```

Here, the AND operator is used as a conjunctive operator to combine the two conditions Total > 2 AND BillingCountrywhich is the USA. You'll receive the invoice records with a total bill value of more than $2 with the USA as billing country. This means that for a record to be included in the result, both the conditions should be true. Similarly, the OR operator can also be used to combine multiple conditions in the WHERE clause.
The syntax is as follows:

```
SELECT column1, column2, columnN

FROM table_name

WHERE [condition1] OR [condition2]...OR [conditionN]
```

Let’s continue to use the same invoicestable for the next example. If you

want to get a list of invoices for which the BillingCountryis the USA or France, how would you use the OR operator to combine the two conditions?
You can write the following SQL syntax:

```
SELECT *

FROM invoices

WHERE BillingCountry = 'USA' OR BillingCountry='France';
```

Run

Reset

```
+-----------+------------+--------------------
+--------------------------+----------------+--------------
+----------------+-------------------+-------+
| InvoiceId | CustomerId | InvoiceDate         | BillingAddress
| BillingCity    | BillingState | BillingCountry | BillingPostalCode |
Total |
+-----------+------------+--------------------
+--------------------------+----------------+--------------
+----------------+-------------------+-------+
```

```
|         5 |         23 | 2009-01-11 00:00:00 | 69 Salem Street
| Boston         | MA           | USA            | 2113              |
13.86 |
|         8 |         40 | 2009-02-01 00:00:00 | 8, Rue Hanovre
| Paris          | None         | France         | 75002             |
1.98 |
|         9 |         42 | 2009-02-02 00:00:00 | 9, Place Louis Barthou
| Bordeaux       | None         | France         | 33000             |
3.96 |
|        13 |         16 | 2009-02-19 00:00:00 | 1600 Amphitheatre
Parkway | Mountain View  | CA           | USA            | 94043-1351
|  0.99 |
|        14 |         17 | 2009-03-04 00:00:00 | 1 Microsoft Way
| Redmond        | WA           | USA            | 98052-8300        |
1.98 |
|        15 |         19 | 2009-03-04 00:00:00 | 1 Infinite Loop
| Cupertino      | CA           | USA            | 95014             |
1.98 |
|        16 |         21 | 2009-03-05 00:00:00 | 801 W 4th Street
| Reno           | NV           | USA            | 89503             |
3.96 |
|        17 |         25 | 2009-03-06 00:00:00 | 319 N. Frances Street
| Madison        | WI           | USA            | 53703             |
5.94 |
|        19 |         40 | 2009-03-14 00:00:00 | 8, Rue Hanovre
| Paris          | None         | France         | 75002             |
13.86 |
|        26 |         19 | 2009-04-14 00:00:00 | 1 Infinite Loop
| Cupertino      | CA           | USA            | 95014             |
13.86 |
|        31 |         42 | 2009-05-07 00:00:00 | 9, Place Louis Barthou
| Bordeaux       | None         | France         | 33000             |
5.94 |
|        37 |         17 | 2009-06-06 00:00:00 | 1 Microsoft Way
| Redmond        | WA           | USA            | 98052-8300        |
3.96 |
|        38 |         21 | 2009-06-07 00:00:00 | 801 W 4th Street
| Reno           | NV           | USA            | 89503             |
5.94 |
|        39 |         27 | 2009-06-10 00:00:00 | 1033 N Park Ave
| Tucson         | AZ           | USA            | 85719             |
8.91 |
|        59 |         17 | 2009-09-08 00:00:00 | 1 Microsoft Way
| Redmond        | WA           | USA            | 98052-8300        |
5.94 |
|        60 |         23 | 2009-09-11 00:00:00 | 69 Salem Street
| Boston         | MA           | USA            | 2113              |
```

```
8.91 |
|        69 |         25 | 2009-10-25 00:00:00 | 319 N. Frances Street
| Madison        | WI           | USA            | 53703             |
0.99 |
|        70 |         26 | 2009-11-07 00:00:00 | 2211 W Berry Street
| Fort Worth     | TX           | USA            | 76110             |
1.98 |
|        71 |         28 | 2009-11-07 00:00:00 | 302 S 700 E
| Salt Lake City | UT           | USA            | 84102             |
1.98 |
|        74 |         40 | 2009-11-12 00:00:00 | 8, Rue Hanovre
| Paris          | None         | France         | 75002             |
8.91 |
|        81 |         19 | 2009-12-13 00:00:00 | 1 Infinite Loop
| Cupertino      | CA           | USA            | 95014             |
8.91 |
|        82 |         28 | 2009-12-18 00:00:00 | 302 S 700 E
| Salt Lake City | UT           | USA            | 84102             |
13.86 |
|        83 |         42 | 2009-12-26 00:00:00 | 9, Place Louis Barthou
| Bordeaux       | None         | France         | 33000             |
0.99 |
|        84 |         43 | 2010-01-08 00:00:00 | 68, Rue Jouvence
| Dijon          | None         | France         | 21000             |
1.98 |
|        90 |         21 | 2010-01-26 00:00:00 | 801 W 4th Street
| Reno           | NV           | USA            | 89503             |
0.99 |
+-----------+------------+---------------------
+--------------------------+----------------+-------------
+---------------+------------------+-------+
(Output limit exceeded, 25 of 126 total rows shown)
```

You'll notice that the result consists of records where the billing country is the USA or France. This means that for a record to be included in the result, either condition should be true.
Let's consider another scenario. If you want to get a list of invoices where the total value is over $2 and the BillingCountry is USA or France, here's the syntax for the SELECT query using both AND / OR conjunctive operators together in the WHERE clause:

```sql
SELECT *

FROM invoices

WHERE Total > 2 AND (BillingCountry = 'USA' OR BillingCountry
= 'France');
```

Run

Reset

```
+-----------+------------+--------------------
+-----------------------+---------------+-------------
+----------------+-------------------+-------+
| InvoiceId | CustomerId | InvoiceDate         | BillingAddress
| BillingCity    | BillingState | BillingCountry | BillingPostalCode |
Total |
+-----------+------------+--------------------
+-----------------------+---------------+-------------
+----------------+-------------------+-------+
|         5 |         23 | 2009-01-11 00:00:00 | 69 Salem Street
| Boston         | MA           | USA            | 2113              |
13.86 |
|         9 |         42 | 2009-02-02 00:00:00 | 9, Place Louis Barthou
| Bordeaux       | None         | France         | 33000             |
3.96 |
|        16 |         21 | 2009-03-05 00:00:00 | 801 W 4th Street
| Reno           | NV           | USA            | 89503             |
3.96 |
|        17 |         25 | 2009-03-06 00:00:00 | 319 N. Frances Street
```

| Madison | WI | USA | 53703 |
5.94 |
| 19 | 40 | 2009-03-14 00:00:00 | 8, Rue Hanovre
| Paris | None | France | 75002 |
13.86 |
| 26 | 19 | 2009-04-14 00:00:00 | 1 Infinite Loop
| Cupertino | CA | USA | 95014 |
13.86 |
| 31 | 42 | 2009-05-07 00:00:00 | 9, Place Louis Barthou
| Bordeaux | None | France | 33000 |
5.94 |
| 37 | 17 | 2009-06-06 00:00:00 | 1 Microsoft Way
| Redmond | WA | USA | 98052-8300 |
3.96 |
| 38 | 21 | 2009-06-07 00:00:00 | 801 W 4th Street
| Reno | NV | USA | 89503 |
5.94 |
| 39 | 27 | 2009-06-10 00:00:00 | 1033 N Park Ave
| Tucson | AZ | USA | 85719 |
8.91 |
| 59 | 17 | 2009-09-08 00:00:00 | 1 Microsoft Way
| Redmond | WA | USA | 98052-8300 |
5.94 |
| 60 | 23 | 2009-09-11 00:00:00 | 69 Salem Street
| Boston | MA | USA | 2113 |
8.91 |
| 74 | 40 | 2009-11-12 00:00:00 | 8, Rue Hanovre
| Paris | None | France | 75002 |
8.91 |
| 81 | 19 | 2009-12-13 00:00:00 | 1 Infinite Loop
| Cupertino | CA | USA | 95014 |
8.91 |
| 82 | 28 | 2009-12-18 00:00:00 | 302 S 700 E
| Salt Lake City | UT | USA | 84102 |
13.86 |
| 93 | 26 | 2010-02-09 00:00:00 | 2211 W Berry Street
| Fort Worth | TX | USA | 76110 |
3.96 |
| 103 | 24 | 2010-03-21 00:00:00 | 162 E Superior Street
| Chicago | IL | USA | 60611 |
15.86 |
| 107 | 43 | 2010-04-12 00:00:00 | 68, Rue Jouvence
| Dijon | None | France | 21000 |
3.96 |
| 114 | 22 | 2010-05-13 00:00:00 | 120 S Orange Ave
| Orlando | FL | USA | 32801 |
3.96 |

```
|       115 |         26 | 2010-05-14 00:00:00 | 2211 W Berry Street
| Fort Worth     | TX           | USA            | 76110             |
5.94 |
|       117 |         41 | 2010-05-22 00:00:00 | 11, Place Bellecour
| Lyon           | None         | France         | 69002             |
13.86 |
|       124 |         20 | 2010-06-22 00:00:00 | 541 Del Medio Avenue
| Mountain View  | CA           | USA            | 94040-111         |
13.86 |
|       128 |         39 | 2010-07-14 00:00:00 | 4, Rue Milton
| Paris          | None         | France         | 75009             |
3.96 |
|       129 |         43 | 2010-07-15 00:00:00 | 68, Rue Jouvence
| Dijon          | None         | France         | 21000             |
5.94 |
|       135 |         18 | 2010-08-14 00:00:00 | 627 Broadway
| New York       | NY           | USA            | 10012-2612        |
3.96 |
+-----------+------------+--------------------
+-----------------------+----------------+-------------
+---------------+-------------------+-------+
(Output limit exceeded, 25 of 76 total rows shown)
```

You'll notice that it has filtered out the invoice records that have a total value of more than $2.From that result, it has also filtered out the records that have a country value of either the USA or France. In the query, the two conditions combined with the OR operator are surrounded by a pair of parentheses to ensure that they are evaluated as one single expression.

The other SQL logical and comparison operators which were not demonstrated in this reading can also be used in the WHERE clause. In addition, the WHERE clause can also be used with UPDATE and DELETE statements. To learn more, consult the additional resources reading of this lesson.