SQLITE TUTORIAL

SQLite BETWEEN

Summary: in this tutorial, you will learn how to use the SQLite `BETWEEN` operator to test whether a value is in a range of values.

Introduction to SQLite BETWEEN Operator

The `BETWEEN` operator is a logical operator that tests whether a value is in range of values. If the value is in the specified range, the `BETWEEN` operator returns true. The `BETWEEN` operator can be used in the `WHERE` (https://www.sqlitetutorial.net/sqlite-where/) clause of the `SELECT` (https://www.sqlitetutorial.net/sqlite-select/) , `DELETE` (https://www.sqlitetutorial.net/sqlite-delete/) , `UPDATE` (https://www.sqlitetutorial.net/sqlite-update/) , and `REPLACE` (https://www.sqlitetutorial.net/sqlite-replace-statement/) statements.

The following illustrates the syntax of the SQLite `BETWEEN` operator:

```
test_expression BETWEEN low_expression AND high_expression
```

In this syntax:

- `test_expression` is an expression to test for in the range defined by `low_expression` and `high_expression` .
- `low_expression` and `high_expression` is any valid expression that specify the low and high values of the range. The `low_expression` should be less than or equal to `high_expression` , or the `BETWEEN` is always returns false.
- The `AND` keyword is a placeholder which indicates the `test_expression` should be within the range specified by `low_expression` and `high_expression` .

Note that the `BETWEEN` operator is inclusive. It returns true when the `test_expression` is less than or equal to `high_expression` and greater than or equal to the value of `low_expression` :

```
test_expression >= low_expression AND test_expression <= high_expressior
```

To specify an exclusive range, you use the greater than (>) and less than operators (<).

Note that if any input to the `BETWEEN` operator is NULL, the result is NULL, or unknown to be precise.

To negate the result of the `BETWEEN` operator, you use the `NOT BETWEEN` operator as follows:

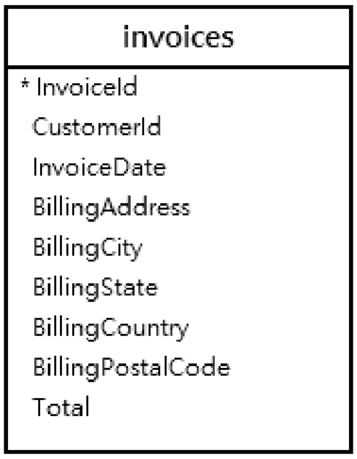
```
test_expression NOT BETWEEN low_expression AND high_expression
```

The NOT `BETWEEN` returns true if the value of `test_expression` is less than the value of `low_expression` or greater than the value of `high_expression` :

```
test_expression < low_expression OR test_expression > high_expression
```

SQLite BETWEEN operator examples

We will use the `invoices` table from the sample database (https://www.sqlitetutorial.net/sqlite-sample-database/) for the demonstration:



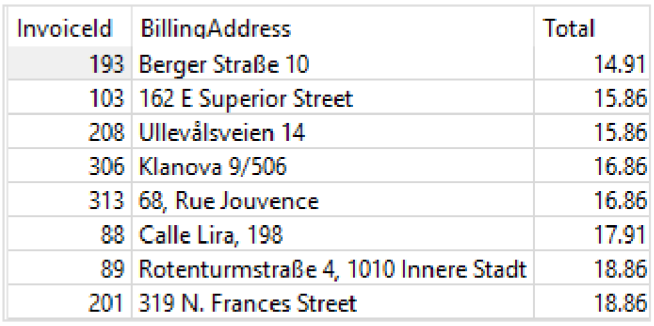
SQLite BETWEEN numeric values example

The following statement finds invoices whose total is `between` 14.96 and 18.86:

```
SELECT
    InvoiceId,
    BillingAddress,
    Total
FROM
    invoices
WHERE
```

```
    Total BETWEEN 14.91 and 18.86
ORDER BY
    Total;
```

Here is the output:



InvoiceId	BillingAddress	Total
193	Berger Straße 10	14.91
103	162 E Superior Street	15.86
208	Ullevålsveien 14	15.86
306	Klanova 9/506	16.86
313	68, Rue Jouvence	16.86
88	Calle Lira, 198	17.91
89	Rotenturmstraße 4, 1010 Innere Stadt	18.86
201	319 N. Frances Street	18.86

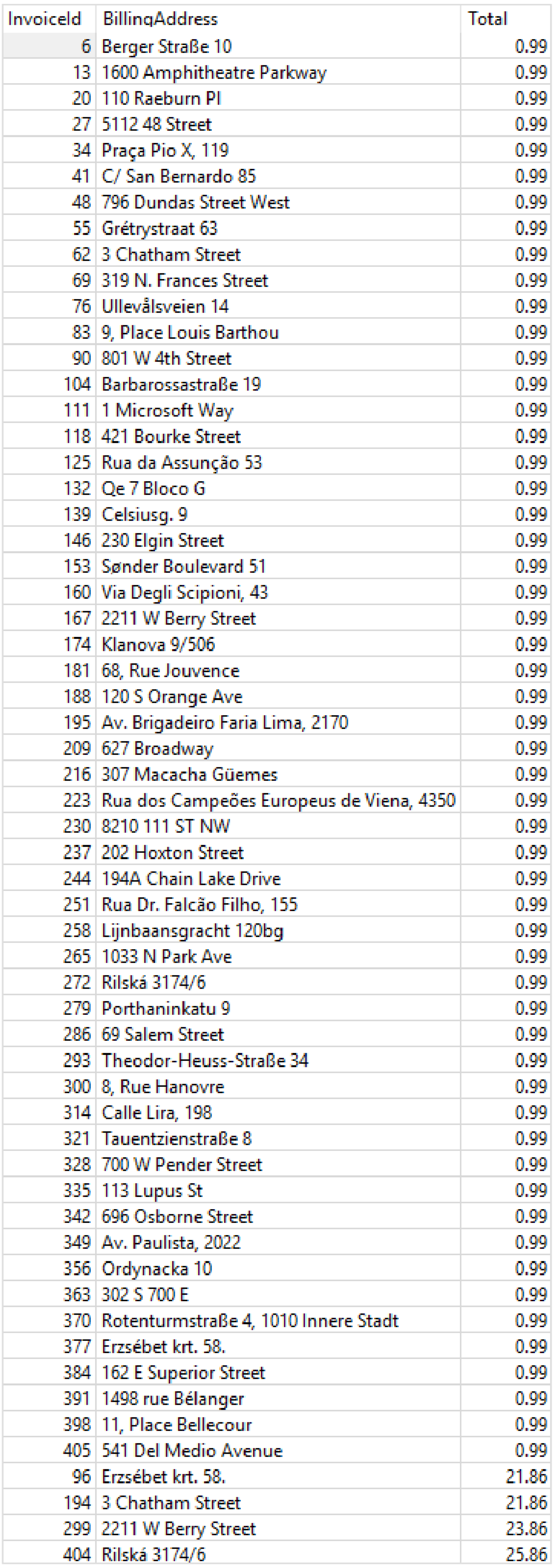
As you can see, the invoices whose total is 14.91 or 18.86 are included in the result set.

SQLite NOT BETWEEN numeric values example

To find the invoices whose total are not between 1 and 20, you use the `NOT BETWEEN` operator as shown in the following query:

```
SELECT
    InvoiceId,
    BillingAddress,
    Total
FROM
    invoices
WHERE
    Total NOT BETWEEN 1 and 20
ORDER BY
    Total;
```

The following picture shows the output:



InvoiceId	BillingAddress	Total
6	Berger Straße 10	0.99
13	1600 Amphitheatre Parkway	0.99
20	110 Raeburn Pl	0.99
27	5112 48 Street	0.99
34	Praça Pio X, 119	0.99
41	C/ San Bernardo 85	0.99
48	796 Dundas Street West	0.99
55	Grétrystraat 63	0.99
62	3 Chatham Street	0.99
69	319 N. Frances Street	0.99
76	Ullevålsveien 14	0.99
83	9, Place Louis Barthou	0.99
90	801 W 4th Street	0.99
104	Barbarossastraße 19	0.99
111	1 Microsoft Way	0.99
118	421 Bourke Street	0.99
125	Rua da Assunção 53	0.99
132	Qe 7 Bloco G	0.99
139	Celsiusg. 9	0.99
146	230 Elgin Street	0.99
153	Sønder Boulevard 51	0.99
160	Via Degli Scipioni, 43	0.99
167	2211 W Berry Street	0.99
174	Klanova 9/506	0.99
181	68, Rue Jouvence	0.99
188	120 S Orange Ave	0.99
195	Av. Brigadeiro Faria Lima, 2170	0.99
209	627 Broadway	0.99
216	307 Macacha Güemes	0.99
223	Rua dos Campeões Europeus de Viena, 4350	0.99
230	8210 111 ST NW	0.99
237	202 Hoxton Street	0.99
244	194A Chain Lake Drive	0.99
251	Rua Dr. Falcão Filho, 155	0.99
258	Lijnbaansgracht 120bg	0.99
265	1033 N Park Ave	0.99
272	Rilská 3174/6	0.99
279	Porthaninkatu 9	0.99
286	69 Salem Street	0.99
293	Theodor-Heuss-Straße 34	0.99
300	8, Rue Hanovre	0.99
314	Calle Lira, 198	0.99
321	Tauentzienstraße 8	0.99
328	700 W Pender Street	0.99
335	113 Lupus St	0.99
342	696 Osborne Street	0.99
349	Av. Paulista, 2022	0.99
356	Ordynacka 10	0.99
363	302 S 700 E	0.99
370	Rotenturmstraße 4, 1010 Innere Stadt	0.99
377	Erzsébet krt. 58.	0.99
384	162 E Superior Street	0.99
391	1498 rue Bélanger	0.99
398	11, Place Bellecour	0.99
405	541 Del Medio Avenue	0.99
96	Erzsébet krt. 58.	21.86
194	3 Chatham Street	21.86
299	2211 W Berry Street	23.86
404	Rilská 3174/6	25.86

As clearly shown in the output, the result includes the invoices whose total is less than 1 and greater than 20.

SQLite BETWEEN dates example

The following example finds invoices whose invoice dates are from `January 1 2010` and `January 31 2010` :

```
SELECT
    InvoiceId,
    BillingAddress,
    InvoiceDate,
    Total
FROM
    invoices
WHERE
    InvoiceDate BETWEEN '2010-01-01' AND '2010-01-31'
ORDER BY
    InvoiceDate;
```

Here is the output:

SQLite NOT BETWEEN dates example

The following statement finds invoices whose dates are not between January 03, 2009, and December 01, 2013:

```
SELECT
    InvoiceId,
    BillingAddress,
    date(InvoiceDate) InvoiceDate,
    Total
FROM
    invoices
WHERE
    InvoiceDate NOT BETWEEN '2009-01-03' AND '2013-12-01'
ORDER BY
    InvoiceDate;
```

The output is as follows:

In this tutorial, you have learned how to use the SQLite `BETWEEN` operator to test whether a value is in a range of values