<https://unetway.com/tutorial/sqlite>

**SQLite - Операторы**

Оператором является зарезервированное слово или символ, используемый в предложении WHERE оператора SQLite для выполнения операций, таких как сравнения и арифметические операции.

Операторы используются для указания условий в выражении SQLite и служат конъюнкциями для нескольких условий в инструкции.

* Арифметические операторы
* Операторы сравнения
* Логические операторы
* Побитовые операторы

**Арифметические операторы SQLite**

Предположим, что переменная **a** имеет значение 10, а переменная **b -** 20, то SQLite-арифметические операторы будут использоваться следующим образом:

|  |  |  |
| --- | --- | --- |
| **Оператор** | **Описание** | **Пример** |
| + (Сложение) | Добавляет значения по обе стороны оператора | a + b даст 30 |
| - (Вычитание) | Вычитает правый операнд из левого операнда | a - b даст -10 |
| \* (Умножение) | Умножает значения по обе стороны оператора | a \* b даст 200 |
| / (Деление) | Делит левый операнд на правый операнд | b / a даст 2 |
| % (Модуль) | Делит левый операнд на правый операнд и возвращает остаток | b% a даст 0 |

**Операторы сравнения SQLite**

Предположим, что переменная **a** имеет значение 10, а переменная **b -** 20, то операторы сравнения SQLite будут использоваться следующим образом:

|  |  |  |
| --- | --- | --- |
| **Оператор** | **Описание** | **Пример** |
| == | Проверяет, равны ли значения двух операндов или нет, если да, то условие становится истинным. | (a == b) не соответствует действительности. |
| знак равно | Проверяет, равны ли значения двух операндов или нет, если да, то условие становится истинным. | (a = b) неверно. |
| знак равно | Проверяет, равны ли значения двух операндов или нет, если значения не равны, тогда условие становится истинным. | (a! = b) истинно. |
| <> | Проверяет, равны ли значения двух операндов или нет, если значения не равны, тогда условие становится истинным. | (a <> b) верно. |
| > | Проверяет, превышает ли значения левого операнда значение правого операнда, если да, то условие становится истинным. | (a> b) неверно. |
| < | Проверяет, являются ли значения левого операнда меньше значения правильного операнда, если да, то условие становится истинным. | (a <b) истинно. |
| > = | Проверяет, является ли значение левого операнда больше или равно значению правильного операнда, если да, тогда условие становится истинным. | (a> = b) неверно. |
| <= | Проверяет, является ли значение левого операнда меньше или равно значению правильного операнда, если да, тогда условие становится истинным. | (a <= b) истинно. |
| <! | Проверяет, не превышает ли значение левого операнда значение правого операнда, если да, то условие становится истинным. | (a! <b) является ложным. |
| !> | Проверяет, не превышает ли значение левого операнда значение правого операнда, если да, то условие становится истинным. | (a!> b) истинно. |

**Логические операторы SQLite**

Вот список всех логических операторов, доступных в SQLite.

|  |  |
| --- | --- |
| **Оператор** | **Описание** |
| **AND** | Оператор AND допускает существование множества условий в предложении WHERE оператора SQL. |
| **BETWEEN** | Оператор BETWEEN используется для поиска значений, находящихся в пределах набора значений, с учетом минимального значения и максимального значения. |
| **EXISTS** | Оператор EXISTS используется для поиска наличия строки в указанной таблице, соответствующей определенным критериям. |
| **IN** | Оператор IN используется для сравнения значения со списком литеральных значений, которые были указаны. |
| **NOT IN** | Отрицание оператора IN, которое используется для сравнения значения со списком значений буквального значения, которые были указаны. |
| **LIKE** | Оператор LIKE используется для сравнения значения с аналогичными значениями с помощью подстановочных операторов. |
| **GLOB** | Оператор GLOB используется для сравнения значения с аналогичными значениями с помощью подстановочных операторов. Кроме того, GLOB чувствителен к регистру, в отличие от LIKE. |
| **NOT** | Оператор NOT меняет смысл логического оператора, с которым он используется. Например. НЕ СУЩЕСТВУЕТ, НЕ МЕЖДУ, НЕ ВХОДИТ и т. Д. Это оператор отрицания. |
| **OR** | Оператор OR используется для объединения нескольких условий в предложение WHERE оператора SQL. |
| **IS NULL** | Оператор NULL используется для сравнения значения со значением NULL. |
| **IS** | Оператор IS работает как = |
| **IS NOT** | Оператор IS работает как! = |
| **||** | Добавляет две разные строки и создает новую. |
| **UNIQUE** | Оператор UNIQUE выполняет поиск каждой строки указанной таблицы для уникальности (без дубликатов). |

**SQLite Побитовые операторы**

Побитовый оператор работает с битами и выполняет побитовые операции. Ниже приведена таблица истинности для **&** и **| :**

|  |  |  |  |
| --- | --- | --- | --- |
| **p** | **q** | **p & q** | **p | q** |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |

Предположим, что **A** = 60; и **B** = 13, то в двоичном формате они будут выглядеть следующим образом:

A = 0011 1100

B = 0000 1101

**A** & B = 0000 1100

A | B = 0011 1101

~ A = 1100 0011

Побитовые операторы, поддерживаемые языком SQLite, перечислены в следующей таблице. Предположим, что переменная **A** имеет значение 60, а переменная **B** имеет значение 13, тогда

Побитовые операторы, поддерживаемые языком SQLite, перечислены в следующей таблице.

|  |  |  |
| --- | --- | --- |
| **Оператор** | **Описание** | **Пример** |
| & | Двоичный оператор AND копирует бит в результат, если он существует в обоих операндах. | (A & B) даст 12, что составляет 0000 1100 |
| | | Двоичный оператор OR копирует бит, если он существует в любом из операндов. | (A | B) даст 61, который равен 0011 1101 |
| ~ | Binary Ones Оператор дополнения является унарным и имеет эффект «flipping» бит. | (~ A) даст -61, что составляет 1100 0011 в форме дополнения 2 из-за подписанного двоичного числа |
| << | Двойной левый оператор сдвига.Значение левых операндов перемещается влево на количество бит, заданных правым операндом. | A << 2 даст 240, что составляет 1111 0000 |
| >> | Двоичный оператор правого сдвига. Значение левых операндов перемещается вправо на количество бит, заданных правым операндом. | A >> 2 даст 15, что составляет 0000 1111 |