SQLite Select 语句 SQLite 表达式

SQLite 运算符

SQLite 运算符是什么？

运算符是一个保留字或字符，主要用于 SQLite 语句的 WHERE 子句中执行操作，如比较和算术运算。

运算符用于指定 SQLite 语句中的条件，并在语句中连接多个条件。

- 算术运算符
- 比较运算符
- 逻辑运算符
- 位运算符

SQLite 算术运算符

假设变量 a=10，变量 b=20，则：

运算符	描述	实例
+	加法 - 把运算符两边的值相加	a + b 将得到 30
-	减法 - 左操作数减去右操作数	a - b 将得到 -10
*	乘法 - 把运算符两边的值相乘	a * b 将得到 200
/	除法 - 左操作数除以右操作数	b / a 将得到 2
%	取模 - 左操作数除以右操作数后得到的余数	b % a will give 0

实例

下面是 SQLite 算术运算符的简单实例：

```
sqlite> .mode line
sqlite> select 10 + 20;
10 + 20 = 30


sqlite> select 10 - 20;
10 - 20 = -10


sqlite> select 10 * 20;
```

```
10 * 20 = 200


sqlite> select 10 / 5;
10 / 5 = 2


sqlite> select 12 %  5;
12 %  5 = 2
```

SQLite 比较运算符

假设变量 a=10，变量 b=20，则：

运算符	描述	实例
==	检查两个操作数的值是否相等，如果相等则条件为真。	(a == b) 不为真。
=	检查两个操作数的值是否相等，如果相等则条件为真。	(a = b) 不为真。
!=	检查两个操作数的值是否相等，如果不相等则条件为真。	(a != b) 为真。
<>	检查两个操作数的值是否相等，如果不相等则条件为真。	(a <> b) 为真。
>	检查左操作数的值是否大于右操作数的值，如果是则条件为真。	(a > b) 不为真。
<	检查左操作数的值是否小于右操作数的值，如果是则条件为真。	(a < b) 为真。
>=	检查左操作数的值是否大于等于右操作数的值，如果是则条件为真。	(a >= b) 不为真。
<=	检查左操作数的值是否小于等于右操作数的值，如果是则条件为真。	(a <= b) 为真。
!<	检查左操作数的值是否不小于右操作数的值，如果是则条件为真。	(a !< b) 为假。
!>	检查左操作数的值是否不大于右操作数的值，如果是则条件为真。	(a !> b) 为真。

实例

假设 COMPANY 表有以下记录：

```
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
1           Paul        32          California  20000.0
2           Allen       25          Texas       15000.0
3           Teddy       23          Norway      20000.0
4           Mark        25          Rich-Mond   65000.0
5           David       27          Texas       85000.0
6           Kim         22          South-Hall  45000.0
7           James       24          Houston     10000.0
```

下面的实例演示了各种 SQLite 比较运算符的用法。

> 在这里，我们使用 **WHERE** 子句，这将会在后边单独的一个章节中讲解，但现在您需要明白，WHERE 子句是用来设置 SELECT 语句的条件语句。

下面的 SELECT 语句列出了 SALARY 大于 50,000.00 的所有记录：

```
sqlite> SELECT * FROM COMPANY WHERE SALARY > 50000;
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
4           Mark        25          Rich-Mond   65000.0
5           David       27          Texas       85000.0
```

下面的 SELECT 语句列出了 SALARY 等于 20,000.00 的所有记录：

```
sqlite>  SELECT * FROM COMPANY WHERE SALARY = 20000;
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
1           Paul        32          California  20000.0
3           Teddy       23          Norway      20000.0
```

下面的 SELECT 语句列出了 SALARY 不等于 20,000.00 的所有记录：

```
sqlite>  SELECT * FROM COMPANY WHERE SALARY != 20000;
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
2           Allen       25          Texas       15000.0
4           Mark        25          Rich-Mond   65000.0
5           David       27          Texas       85000.0
6           Kim         22          South-Hall  45000.0
7           James       24          Houston     10000.0
```

下面的 SELECT 语句列出了 SALARY 不等于 20,000.00 的所有记录：

```
sqlite> SELECT * FROM COMPANY WHERE SALARY <> 20000;
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
2           Allen       25          Texas       15000.0
4           Mark        25          Rich-Mond   65000.0
5           David       27          Texas       85000.0
6           Kim         22          South-Hall  45000.0
7           James       24          Houston     10000.0
```

下面的 SELECT 语句列出了 SALARY 大于等于 65,000.00 的所有记录：

```
sqlite> SELECT * FROM COMPANY WHERE SALARY >= 65000;
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
4           Mark        25          Rich-Mond   65000.0
5           David       27          Texas       85000.0
```

SQLite 逻辑运算符

下面是 SQLite 中所有的逻辑运算符列表。

运算符	描述
AND	AND 运算符允许在一个 SQL 语句的 WHERE 子句中的多个条件的存在。
BETWEEN	BETWEEN 运算符用于在给定最小值和最大值范围内的一系列值中搜索值。
EXISTS	EXISTS 运算符用于在满足一定条件的指定表中搜索行的存在。
IN	IN 运算符用于把某个值与一系列指定列表的值进行比较。
NOT IN	IN 运算符的对立面，用于把某个值与不在一系列指定列表的值进行比较。
LIKE	LIKE 运算符用于把某个值与使用通配符运算符的相似值进行比较。
GLOB	GLOB 运算符用于把某个值与使用通配符运算符的相似值进行比较。GLOB 与 LIKE 不同之处在于，它是大小写敏感的。
NOT	NOT 运算符是所用的逻辑运算符的对立面。比如 NOT EXISTS、NOT BETWEEN、NOT IN，等等。**它是否定运算符。**
OR	OR 运算符用于结合一个 SQL 语句的 WHERE 子句中的多个条件。
IS NULL	NULL 运算符用于把某个值与 NULL 值进行比较。

IS	IS 运算符与 = 相似。
IS NOT	IS NOT 运算符与 != 相似。
||	连接两个不同的字符串，得到一个新的字符串。
UNIQUE	UNIQUE 运算符搜索指定表中的每一行，确保唯一性（无重复）。

实例

假设 COMPANY 表有以下记录：

```
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
1           Paul        32          California  20000.0
2           Allen       25          Texas       15000.0
3           Teddy       23          Norway      20000.0
4           Mark        25          Rich-Mond   65000.0
5           David       27          Texas       85000.0
6           Kim         22          South-Hall  45000.0
7           James       24          Houston     10000.0
```

下面的实例演示了 SQLite 逻辑运算符的用法。

下面的 SELECT 语句列出了 AGE 大于等于 25 **且**工资大于等于 65000.00 的所有记录：

```
sqlite> SELECT * FROM COMPANY WHERE AGE >= 25 AND SALARY >= 65000;
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
4           Mark        25          Rich-Mond   65000.0
5           David       27          Texas       85000.0
```

下面的 SELECT 语句列出了 AGE 大于等于 25 **或**工资大于等于 65000.00 的所有记录：

```
sqlite> SELECT * FROM COMPANY WHERE AGE >= 25 OR SALARY >= 65000;
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
1           Paul        32          California  20000.0
2           Allen       25          Texas       15000.0
4           Mark        25          Rich-Mond   65000.0
5           David       27          Texas       85000.0
```

下面的 SELECT 语句列出了 AGE 不为 NULL 的所有记录，结果显示所有的记录，意味着没有一个记录的 AGE 等于 NULL：

```
sqlite>  SELECT * FROM COMPANY WHERE AGE IS NOT NULL;
ID          NAME        AGE         ADDRESS     SALARY
```

```
----------  ----------  ----------  ----------  ----------
1           Paul        32          California  20000.0
2           Allen       25          Texas       15000.0
3           Teddy       23          Norway      20000.0
4           Mark        25          Rich-Mond   65000.0
5           David       27          Texas       85000.0
6           Kim         22          South-Hall  45000.0
7           James       24          Houston     10000.0
```

下面的 SELECT 语句列出了 NAME 以 'Ki' 开始的所有记录，'Ki' 之后的字符不做限制：

```
sqlite> SELECT * FROM COMPANY WHERE NAME LIKE 'Ki%';
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
6           Kim         22          South-Hall  45000.0
```

下面的 SELECT 语句列出了 NAME 以 'Ki' 开始的所有记录，'Ki' 之后的字符不做限制：

```
sqlite> SELECT * FROM COMPANY WHERE NAME GLOB 'Ki*';
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
6           Kim         22          South-Hall  45000.0
```

下面的 SELECT 语句列出了 AGE 的值为 25 或 27 的所有记录：

```
sqlite> SELECT * FROM COMPANY WHERE AGE IN ( 25, 27 );
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
2           Allen       25          Texas       15000.0
4           Mark        25          Rich-Mond   65000.0
5           David       27          Texas       85000.0
```

下面的 SELECT 语句列出了 AGE 的值既不是 25 也不是 27 的所有记录：

```
sqlite> SELECT * FROM COMPANY WHERE AGE NOT IN ( 25, 27 );
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
1           Paul        32          California  20000.0
3           Teddy       23          Norway      20000.0
6           Kim         22          South-Hall  45000.0
7           James       24          Houston     10000.0
```

下面的 SELECT 语句列出了 AGE 的值在 25 与 27 之间的所有记录：

```
sqlite> SELECT * FROM COMPANY WHERE AGE BETWEEN 25 AND 27;
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
2           Allen       25          Texas       15000.0
4           Mark        25          Rich-Mond   65000.0
5           David       27          Texas       85000.0
```

下面的 SELECT 语句使用 SQL 子查询，子查询查找 SALARY > 65000 的带有 AGE 字段的所有记录，后边的 WHERE 子句与 EXISTS 运算符一起使用，列出了外查询中的 AGE 存在于子查询返回的结果中的所有记录：

```
sqlite> SELECT AGE FROM COMPANY
        WHERE EXISTS (SELECT AGE FROM COMPANY WHERE SALARY > 65000);
AGE
----------
32
25
23
25
27
22
24
```

下面的 SELECT 语句使用 SQL 子查询，子查询查找 SALARY > 65000 的带有 AGE 字段的所有记录，后边的 WHERE 子句与 > 运算符一起使用，列出了外查询中的 AGE 大于子查询返回的结果中的年龄的所有记录：

```
sqlite> SELECT * FROM COMPANY
        WHERE AGE > (SELECT AGE FROM COMPANY WHERE SALARY > 65000);
ID          NAME        AGE         ADDRESS     SALARY
----------  ----------  ----------  ----------  ----------
1           Paul        32          California  20000.0
```

SQLite 位运算符

位运算符作用于位，并逐位执行操作。真值表 & 和 | 如下：

p	q	p & q	p | q
0	0	0	0
0	1	0	1
1	1	1	1
1	0	0	1

假设如果 A = 60，且 B = 13，现在以二进制格式，它们如下所示：

A = 0011 1100

B = 0000 1101

A&B = 0000 1100

A|B = 0011 1101

~A = 1100 0011

下表中列出了 SQLite 语言支持的位运算符。假设变量 A=60，变量 B=13，则：

运算符	描述	实例
&	如果同时存在于两个操作数中，二进制 AND 运算符复制一位到结果中。	(A & B) 将得到 12，即为 0000 1100
|	如果存在于任一操作数中，二进制 OR 运算符复制一位到结果中。	(A | B) 将得到 61，即为 0011 1101
~	二进制补码运算符是一元运算符，具有"翻转"位效应，即0变成1，1变成0。	(~A) 将得到 -61，即为 1100 0011，一个有符号二进制数的补码形式。
<<	二进制左移运算符。左操作数的值向左移动右操作数指定的位数。	A << 2 将得到 240，即为 1111 0000
>>	二进制右移运算符。左操作数的值向右移动右操作数指定的位数。	A >> 2 将得到 15，即为 0000 1111

实例

下面的实例演示了 SQLite 位运算符的用法：

```
sqlite> .mode line
sqlite> select 60 | 13;
60 | 13 = 61

sqlite> select 60 & 13;
60 & 13 = 12

sqlite>  select  (~60);
(~60) = -61

sqlite>  select  (60 << 2);
(60 << 2) = 240

sqlite>  select  (60 >> 2);
(60 >> 2) = 15
```

SQLite Select 语句 SQLite 表达式

点我分享笔记