SQLITE - LOGICAL OPERATORS

Here is a list of all the logical operators available in SQLite

Operator	Description
AND	The AND operator allows the existence of multiple conditions in an SQL statement's WHERE clause.
BETWEEN	The BETWEEN operator is used to search for values that are within a set of values, given the minimum value and the maximum value.
EXISTS	The EXISTS operator is used to search for the presence of a row in a specified table that meets certain criteria.
IN	The IN operator is used to compare a value to a list of literal values that have been specified.
NOT IN	The negation of IN operator which is used to compare a value to a list of literal values that have been specified.
LIKE	The LIKE operator is used to compare a value to similar values using wildcard operators.
GLOB	The GLOB operator is used to compare a value to similar values using wildcard operators. Also, GLOB is case sensitive, unlike LIKE.
NOT	The NOT operator reverses the meaning of the logical operator with which it is used. Eg. NOT EXISTS, NOT BETWEEN, NOT IN, etc. **This is negate operator.**
OR	The OR operator is used to combine multiple conditions in an SQL statement's WHERE clause.
IS NULL	The NULL operator is used to compare a value with a NULL value.
IS	The IS operator work like =
IS NOT	The IS operator work like !=
||	Adds two different strings and make new one.
UNIQUE	The UNIQUE operator searches every row of a specified table for uniqueness *noduplicates*.

Example

Consider COMPANY table has the following records:

```
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
```

Here are simple examples showing usage of SQLite Logical Operators. Following SELECT statement lists down all the records where AGE is greater than or equal to 25 and salary is greater

than or equal to 65000.00:

```
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
```

Following SELECT statement lists down all the records where AGE is greater than or equal to 25 **OR** salary is greater than or equal to 65000.00:

```
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
```

Following SELECT statement lists down all the records where AGE is not NULL which means all the records because none of the record is having AGE equal to 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
```

Following SELECT statement lists down all the records where NAME starts with 'Ki', does not matter what comes after 'Ki'.

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

Following SELECT statement lists down all the records where NAME starts with 'Ki', does not matter what comes after 'Ki':

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

Following SELECT statement lists down all the records where AGE value is either 25 or 27:

```
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
```

Following SELECT statement lists down all the records where AGE value is neither 25 nor 27:

```
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
```

Following SELECT statement lists down all the records where AGE value is in BETWEEN 25 AND 27:

```
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
```

Following SELECT statement makes use of SQL sub-query where sub-query finds all the records with AGE field having SALARY > 65000 and later WHERE clause is being used along with EXISTS operator to list down all the records where AGE from the outside query exists in the result returned by sub-query:

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

Following SELECT statement makes use of SQL sub-query where subquery finds all the records with AGE field having SALARY > 65000 and later WHERE clause is being used along with > operator to list down all the records where AGE from outside query is greater than the age in the result returned by sub-query:

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