[http://www.SQLite.org/omitted.html](http://www.sqlite.org/omitted.html)

**SQL Features That SQLite Does Not Implement**

# 关于SQLite对SQL语言的支持情况

## SQLite不支持ANSI SQL92的部分特性

**（1）右外连接与全外连接**

支持左外连接LEFT OUTER JOIN，但是还不支持右外连接RIGHT OUTER JOIN和全外连接FULL OUTER JOIN。

**（2）大部分的修改表操作**

只有RENAME TABLE和ADD COLUMN这两种ALTER TABLE命令被支持。其它ALTER TABLE操作比如DROP COLUMN、ALTER COLUMN、ADD CONSTRAINT未被支持。

对于COUNT（DISTINCT column），SQLite在执行如下语句的时候会报错：

SELECT COUNT(DISTINCT watchid) FROM [watch\_item] WHERE watch\_item.watchid = 1;

其原因是SQLite的所有内置函数都不支持DISTINCT限定，所以如果要统计不重复的记录数的时候会出现一些麻烦。比较可行的做法是先建立一个不重复的记录表的视图，然后再对该视图进行计数。

**（3）触发器**

支持FOR EACH ROW触发器，不支持FOR EACH STATEMENT触发器。

**（4）视图的创建与改写**

SQLite在创建多表视图的时候有一个BUG，问题如下：

CREATE VIEW watch\_single AS SELECT DISTINCT watch\_item.[watchid],watch\_item.[itemid] FROM watch\_item;

上面这条SQL语句执行后会显示成功，但是实际上除了

SELECT COUNT(\*) FROM [watch\_single ] WHERE watch\_ single.watchid = 1;

能执行之外是无法执行其他任何语句的。其原因在于建立视图的时候指定了字段所在的表名，而SQLite并不能正确地识别它。所以上面的创建语句要改为：

CREATE VIEW watch\_single AS SELECT DISTINCT [watchid],[itemid] FROM watch\_item;

但是随之而来的问题是如果是多表的视图，且表间有重名字段的时候就会出现问题。

SQLite中的视图是只读的，对视图不能执行DELETE、INSERT或UPDATE命令。但是可以创建一个触发器试图DELETE、INSERT或UPDATE一个视图。

**（5）授权与撤销**

因为SQLite读取和写入一个普通磁盘文件，唯一的访问许可对于底层操作系统的一般文件有效。GRANT和REVOKE命令（通常见于CS RDBMSes） 不被支持，它们对于嵌入式数据库引擎无意义。

## 附录一SQL的四种连接

**1、内连接**

典型的连接运算，使用像 =  或 <> 之类的比较运算符。内连接包括相等连接和自然连接。内连接使用比较运算符根据每个表共有的列的值匹配两个表中的行。例如，检索 students和courses表中学生标识号相同的所有行。

**2、外连接**

外连接可以是左向外连接、右向外连接或完整外部连接。在FROM子句中指定外连接时，可以由下列几组关键字中的一组指定：

1）LEFT JOIN或LEFT OUTER JOIN

左向外连接的结果集包括LEFT OUTER子句中指定的左表的所有行，而不仅仅是连接列所匹配的行。如果左表的某行在右表中没有匹配行，则在相关联的结果集行中右表的所有选择列表列均为空值。

2）RIGHT JOIN或RIGHT OUTER JOIN

右向外连接是左向外连接的反向连接。将返回右表的所有行。如果右表的某行在左表中没有匹配行，则将为左表返回空值。

3）FULL JOIN或FULL OUTER JOIN

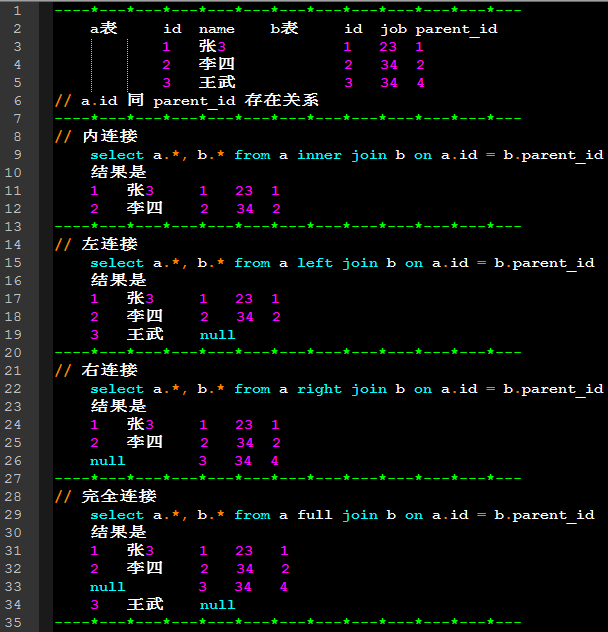
完整外部连接返回左表和右表中的所有行。当某行在另一个表中没有匹配行时，则另一个表的选择列表列包含空值。如果表之间有匹配行，则整个结果集行包含基表的数据值。

**3、交叉连接**

交叉连接返回左表中的所有行，左表中的每一行与右表中的所有行组合。交叉连接也称作笛卡尔积。

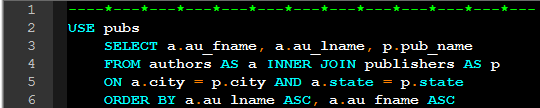
FROM 子句中的表或视图可通过内连接或完整外部连接按任意顺序指定；但是用左或右向外连接指定表或视图时，表或视图的顺序很重要。

**4、例子**



交叉连接。

交叉连接返回左表中的所有行，左表中的每一行与右表中的所有行组合。交叉连接也称作笛卡尔积。例如，下面的内连接检索与某个出版商居住在相同州和城市的作者：



## 附录二SQLite Wiki关于SQLite对SQL的支持历史记录

[http://www.SQLite.org/cvstrac/wiki?p=UnsupportedSql](http://www.sqlite.org/cvstrac/wiki?p=UnsupportedSql)

SQLite - Unsupported Sql

2011-05-06: SQLite doesn't support name subqueries (WITH):

WITH

a AS (SELECT 1 a\_id, 2 a\_ref),

b AS (SELECT 2 b\_id, 3 b\_val)

SELECT \*

FROM a, b WHERE a\_ref=b\_id;

2001-05-07: But it does support: SELECT \* FROM (SELECT 1 a\_id, 2 a\_ref) as a, (SELECT 2 b\_id, 3 b\_val) as b WHERE a\_ref=b\_id;

2009-10-19: SQLite doesn't support functions and subqueries in constraints clauses:

DEFAULT SELECT AVG(\*) FROM table

CHECK (field > SELECT MIN(field) FROM table)

...

2009-08-04: Table and column comments - I have scoured the doco and can't find anything about applying comments to tables or their columns. Easy workaround:

CREATE TABLE SQLite\_tab\_comments (

table\_name text,

comment text

);

CREATE TABLE SQLite\_col\_comments (

table\_name text,

col\_name text,

comment text

);

May I also suggest an even easier workaround:

CREATE TABLE User

-- A table comment

(

uid INTEGER, -- A field comment

flags INTEGER, -- Another field comment

);

.schema User --> will print out the above exactly

2009-08-03: Can we please have support for creating pivot tables/cross tables? That would be really sweet -> bug 1424. See http://en.wikipedia.org/wiki/Pivot\_table for more information.

2011-04-18: vector expressions: (see here for more detail)

SELECT First, Last, Score

FROM mytable

WHERE

('John', 'Jordan', 5) <= (First, Last, Score )

AND (First, Last, Score) <= ('Mike', 'Taylor', 50)

ORDER BY First, Last, Score

LIMIT 1

2009-04-01: positioned updates and deletes

DELETE table WHERE CURRENT OF cursor\_name;

2007-12-30: EXTRACT for TIMESTAMP types

SELECT EXTRACT(YEAR FROM TIMESTAMP '2013-07-02');

These are also useful in the WHERE clause

2007-12-03 : Multi-row INSERT a.k.a. compound INSERT not supported.

INSERT INTO table (col1, col2) VALUES ('row1col1', 'row1col2'), ('row2col1', 'row2col2'), ...

Actually, according to the SQL92 standard, a VALUES expression should be able to stand on itself. For example, the following should return a one-column table with three rows:

VALUES 'john', 'mary', 'paul';

2007-11-13 : STARTING WITH clause not supported

WHERE <Column> STARTING WITH "Text"

returns an error but works on MSSQL and is in my SQL book.

2007.08.23 : DELETE from multiple tables:

DELETE t1, t2 FROM t1, t2, t3

WHERE t1.id=t2.id AND t2.id=t3.id;

...or...

DELETE FROM t1, t2 USING t1, t2, t3 WHERE

t1.id=t2.id AND t2.id=t3.id;

2006.09.15 : CREATE DOMAIN

CREATE DOMAIN code\_postal\_us AS TEXT;

2006.04.28 : DROP CONSTRAINT statement:

ALTER TABLE myTable DROP CONSTRAINT defPK

2006.04.07 : ANSI 99 windowing functions:

SELECT department\_id, last\_name, salary, commission\_pct,

RANK() OVER (PARTITION BY department\_id

ORDER BY salary DESC, commission\_pct) comp\_rank

FROM employees

WHERE department\_id = 80

2006.03.06 : select without left join with the \*= operator

SELECT t1.code, t2.code

FROM table1 t1, table2 t2

WHERE t1.t2\_ref\_id \*= t2.id

This is Sybase ASE syntax. Use LEFT JOIN or RIGHT JOIN instead.

(This is related to "Oracle's join syntax" mentioned below.)

2005.12.27 : Oracle's Named Parameter output syntax. In Oracle, one can declare parameters and select into them as such

Select A1, A2, A3 into (:p1, :p2, :p3) from TableA

2005.10.06 : This appears to be unsupported: updating multiple columns with subselect

update T1 set (theUpdatedValue, theOtherValue) =

(select theTop, theValue from T2 where T2.theKey = T1.theID)

2005.09.25 : free text search capabilities in select statements: Mysql does free text search Match(field\_list) Against(keyword) - Now somewhat supported with 3.3.8, and being enhanced.

2005.08.25 : The ALL and ANY quantifiers for comparisons with subquery results aren't supported.

2005.06.01 : Named parts of natural joins. For example: SELECT a.c1 FROM T1 a NATURAL JOIN T1 b. Because SQLite reduces the number of columns kept, the name is lost.

2004.11.17 : INSERTing one record with all VALUES to DEFAULT:

INSERT INTO example ()

VALUES ();

To get the desired behavior you can use:

INSERT INTO example (rowid) values (null);

2004.08.13 : Oracle's join syntax using (+) and (-):

SELECT a1.a, a1.b, a2.a, a2.b

FROM a1 LEFT JOIN a2 ON a2.b = a1.a

...can be written in Oracle as:

SELECT a1.a, a1.b, a2.a, a2.b

FROM a1, a2

WHERE a1.a = a2.b(+);

2004.07.29 : UPDATE t1, t2 SET t1.f1 = value WHERE t1.f2 = t2.fa

2004.04.25 : a password('') function to mask some values (as used in MySQL) would be fine, I need it, if I give the db out of the house, or is there something I didn't find? Or a simple MD5 function to obscure data using a one way hash. See the MySQL function MD5 or Password for examples.

2004.03.26 : Hierarchical Queries.

START WITH <conditions> CONNECT BY [PRIOR]<conditions> (ORACLE)

2004.03.17 : FLOOR and CEILING functions, e.g. "SELECT FLOOR(salary) FROM personnel;"

2004.02.25 : name columns in views (i.e. CREATE VIEW (foo, bar) AS SELECT qux, quo FROM baz;)

You can use CREATE VIEW v1 AS SELECT qux AS foo, quo AS bar FROM baz;

2004.01.08 : MEDIAN and standard deviation... are they standard? Essential for SQLite standalone executable for shell script users.

\_MEDIAN is difficult because it cannot be done "on-line," i.e., on a stream of data. Following is a solution to MEDIAN credited to David Rozensh tein, Anatoly Abramovich, and Eugene Birger; it is explained here: http://www.oreilly.com/catalog/transqlcook/chapter/ch08.html

SELECT x.Hours median

FROM BulbLife x, BulbLife y

GROUP BY x.Hours

HAVING

SUM(CASE WHEN y.Hours <= x.Hours

THEN 1 ELSE 0 END)>=(COUNT(\*)+1)/2 AND

SUM(CASE WHEN y.Hours >= x.Hours

THEN 1 ELSE 0 END)>=(COUNT(\*)/2)+1

2004.01.01 : DISTINCT ON (expr,...) - this is from Postgres, where expr,... must be the leftmost expressions from the ORDER BY clause

2003.10.30 : INSERTing fewer values than columns does not fill the missing columns with the default values; if fewer values than columns in the table are supplied, all columns filled have to be named before the keyword values

2003.09.04 : Multi-column IN clause (ie. SELECT \* FROM tab WHERE (key1, key2) IN (SELECT...)

2003.08.29 : UPDATE with a FROM clause (not sure if this is standard, Sybase and Microsoft have it). -> UnsupportedSqlUpdateWithFrom

2003.08.06 : ALTER VIEW, ALTER TRIGGER, ALTER TABLE

2003.08.06 : CREATE DATABASE, DROP DATABASE - Does not seem meaningful for an embedded database engine like SQLite. To create a new database, just do SQLite\_open(). To drop a database, delete the file.

2003.08.06 : Schemas - See: http://www.postgresql.org/docs/8.1/static/ddl-schemas.html -> UnsupportedSqlSchemas

2003.08.06 : TRUNCATE (MySQL, Postgresql and Oracle have it... but I don't know if this is a standard command) - SQLite does this automatically when you do a DELETE without a WHERE clause. You can use also VACUUM command. The SQLite DELETE without a WHERE clause is significantly slower though due to the logging of transactions while TRUNCATE doesn't support rollback.

2003.08.06 : ORDER BY myfield ASC NULLS LAST (Oracle)

You can use ORDER BY CASE WHEN myfield ISNULL THEN <somethingBig> ELSE myfield END

2003.08.06 : CREATE TRIGGER [BEFORE | AFTER | INSTEAD OF] (Oracle)

2003.07.28 : Stored Procedures

2003.07.28 : Rollup and Cube -> UnsupportedSqlRollupAndCube

More than one primary key per table, I can specify this with MySQL for example and SQLite returns me an error: more than one primary key specified...

"More than one primary key" is an oxymoron when you're talking about the relational data model. By definition, a primary key uniquely identfies a row. What's the real problem you're trying to solve?

A combined primary key is possible in SQLite, for example:

CREATE TABLE strings (

string\_id INTEGER NOT NULL,

language\_id INTEGER NOT NULL,

string TEXT,

PRIMARY KEY (string\_id, language\_id)

);

SHOW TABLES and DESCRIBE [tablename] would be nice - not sure if they're standard, but they are a rather nice feature of MySQL... -------- No, it's not standard. The standard says it should be a special database called INFORMATION\_SCHEMA, wich contains info about all databases, tables, columns, index, views, stored procedures, etc.

There is a way to simulte that functions, See: http://www.SQLite.org/faq.html#q7 -> FAQ for more informations.

Can someone tell me how to fake describe until something like this is implemented? Sorry, I'm too dependent on Oracle apparently :(

Use the "dot" commands:

.tables

.schema

-------------------------

Or via SQL:

select sql from SQLite\_master where name = 'tablename';

SELECT ... FOR UPDATE OF ... is not supported. This is understandable considering the mechanics of SQLite in that row locking is redundant as the entire database is locked when updating any bit of it. However, it would be good if a future version of SQLite supports it for SQL interchageability reasons if nothing else. The only functionality required is to ensure a "RESERVED" lock is placed on the database if not already there.

create table wg ( cpf numeric not null, id numeric not null, nome varchar(25), primary key (cpf) foreign key (id) ); the foreign key (id) generate an error, a chance to be supported in the future? foreign key dont supported in SQLite? or generater automaticaly or ?

That's not a legal FOREIGN KEY clause; you have to specify what the foreign key references. SQLite parses, but does not enforce, syntactically-legal FOREIGN KEY specifications; there's a PRAGMA that will retrieve foreign-key information from table definitions, allowing you to enforce such constraints with application code.

Analytical functions -> UnsupportedSqlAnalyticalFunctions

2003.07.02 : SQL92 Character sets, collations, coercibility.

FEATURES ADDED IN RECENT VERSIONS

ORDER BY and LIMIT on UPDATE/DELETE, e.g. "UPDATE TABLE SET col = 'value' LIMIT 1" (only when compiled with SQLITE\_ENABLE\_UPDATE\_DELETE\_LIMIT)

Added in 3.6.4

IF EXISTS function, e.g. "DROP TABLE IF EXISTS temp;"

Added in 3.3

Extended POSIX regular expressions (should be easy, man 3 regcomp, or http:// mirbsd.bsdadvocacy.org/man3/regcomp.htm for reference) SELECT \* FROM table WHERE name REGEX '[a-zA-Z]+\_{0,3}';

The infrastructure for this syntax now exists, but you have to create a user-defined regex matching function.

The EXISTS keyword is not supported (IN is, but IN is only a special case of EXISTS). And what about corelated subqueries ?

Both supported as of 3.1.

Inserting blob using X'AABBCCDD' syntax. (note: supported in SQLite3)

CURRENT-Functions like CURRENT\_DATE, CURRENT\_TIME are missing Try "SELECT date('now');" or "SELECT datetime('now','localtime');"

Added as of 3.1

ESCAPE clause for LIKE

Added as of 3.1

AUTO\_INCREMENT field type. SQLite supports auto\_incrementing fields but only if that field is set as "INTEGER PRIMARY KEY".

Oh god no! Stop the evil from spreading! AUTO\_INCREMENT is possibly the worst way of doing unique ids for tables. It requires cached per-connection-handle last\_insert\_id() values. And you're probably already familiar with how much of a hack THAT is.

A much better solution would be to give SQLite proper SEQUENCE support. You already have a private table namespace, so using SQLite\_sequences to store these wouldn't be such a big deal. This is created when the database is created, and looks something like this, taken from a perl MySQL sequence emulation module.

create table mysql\_sequences (

sequence\_name char(32) not null primary key,

sequence\_start bigint not null default 1,

sequence\_increment bigint not null default 1,

sequence\_value bigint not null default 1

)

In fact, why don't you just take a look at the original module (DBIx::MySQLSequence): http://search.cpan.org/~adamk/DBIx-MySQLSequence-1.00/lib/DBIx/MySQLSequence.pm. In fact, why don't you just copy that module, and rewrite using code inside the database.

The main reason for doing this, is that if you want to insert multiple records which reference each other, and these references are not null, you cannot insert one record until you have inserted the one to which it refers, then fetched the last\_insert\_id(), added it to the other record, then insert that, and so in. In trivial cases this isn't too bad, but imagine the cases where you have circular references, or don't know the structure of the data in advance at all.

With sequence support and access to ids before inserting, there are algorithms to resolve these cases. Without it, you are left with things like just outright suspending contraints checking, inserting everything incorrectly, then hoping you can find all the cases of broken values, and fixing them. Which sucks if you don't know the structure beforehand.

To resolve compatibility issues, just do what you do now with the INTEGER PRIMARY\_KEY fields with no default, but allow a DEFAULT SEQUENCENAME.NEXTVAL() or something...

For better or worse, the requested feature was added in 3.1

SELECT t1.ID, (SELECT COUNT(\*) FROM t2 WHERE t2.ID=t1.ID) FROM t1

In other words, in a subselect backreferencing to a field in its parent select.

Now supported as of 3.1

2004.04.07 : Creating a table in a database db1 based on a table in database db2:

create table db1.table1 as select \* from db2.table1;

This is supported by ATTACH DATABASE.