



SQLiteManager

User's Manual

Table of Contents

Introduction	3
Overview	3
Register	3
Encodings	3
Special Notes for Windows Users	4
Startup Wizard	5
Design Panel	6
Overview	6
Creating/Altering Tables	7
Creating Views	8
Creating Indexes	9
Creating Triggers	10
Creating Notes	11
Panels	12
Data Panel	12
SQL Panel	15
Verify Panel	16
Analyze Panel	17
Chart Panel	18
Vacuum, Log and Settings	19
Vacuum	19
Log Panel	19
Settings	20

Import and Export	21
Importing Data	21
Exporting Data	22
 Others	 23
cubeSQL	23
Printing	24
Inline Help	25
Scripting Language	26
 Appendix	 27
Contact Information	27
Copyright	27
Legal Stuff	27

Introduction

Overview

SQLiteManager is a powerful database management system for sqlite databases which combines an easy to use interface with advanced features and blazing speed. SQLiteManager allows you to work with a wide range of sqlite 3 databases (e.g. plain databases, in memory databases, AES 128/256/RC4 encrypted databases and also with cubeSQL server databases).

You can perform basic operations such as creating and browsing tables, views, triggers and indexes in a very easy to use and powerful GUI. SQLiteManager's built-in Lua scripting language engine is flexible enough to let you generate reports or interact with sqlite databases in just about any way you can imagine.

Each database that you open with SQLiteManager is presented in a main window with a toolbar which has nine different panels: Design, Data, SQL, Verify, Analyze, Chart, Vacuum, Log and Settings. This user's manual will cover each panel in detail.

In addition to the main window, SQLiteManager provides a number of functions that you access through menus and buttons. This user's manual will cover each of these functions in detail, but you might want to take a minute to browse all of the menus within SQLiteManager's to familiarize yourself.

Register

Unregistered, SQLiteManager operates with certain limitations. Queries are limited to return no more than 20 rows. Exporting and importing are also limited to the first 20 lines of data into a database. The serial number for SQLiteManager can be entered into the registration dialog, which you can reach through the Help menu.

To become a registered user and receive updates and technical support, you must register SQLiteManager via the web at: <http://www.sqlabs.com/store/>

Encodings

By default, SQLiteManager displays all data as UTF-8 strings. To change SQLiteManager's text encoding, choose a new encoding from the Database menu (under the Encoding sub-menu) The new encoding only affects the database displayed within that window. Databases in other windows are unaffected.

When you change the text encoding with the encodings menu, the new text encoding takes effect immediately for records in the Data and SQL panels. If you add or edit records in the Data pane, your data will be inserted into the database with the encoding in effect at that time.

Special Notes for Windows Users

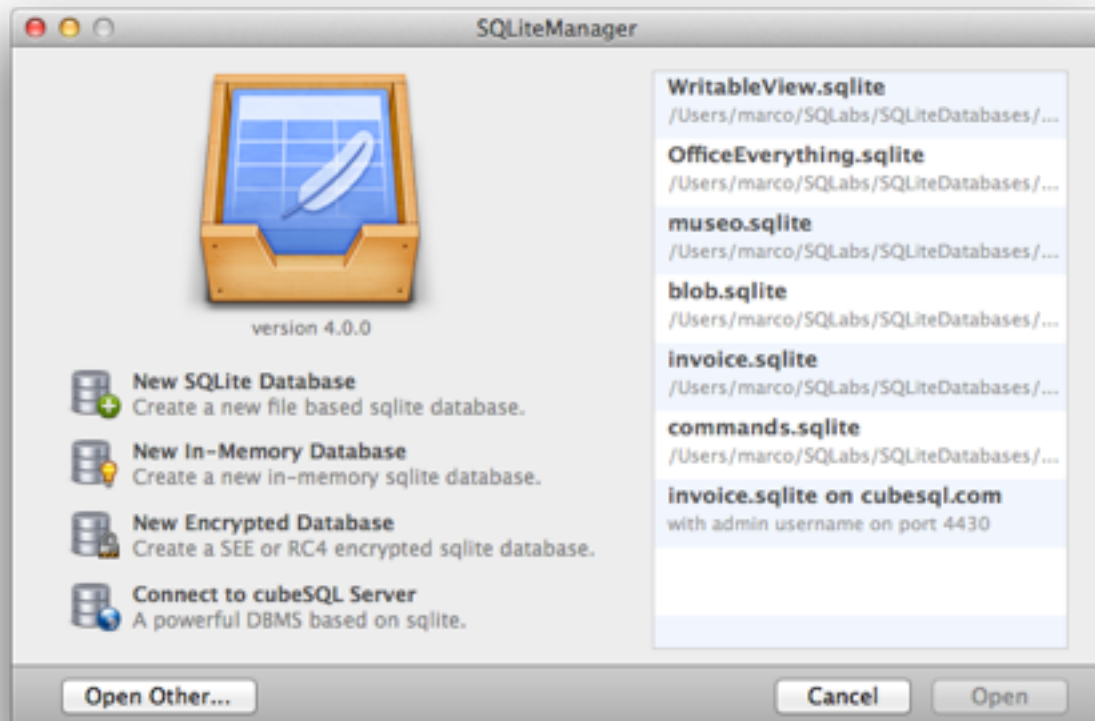
In order to be able to correctly display Asian characters under Windows XP you need to install a special Asian Language Pack.

1. Open the Windows Control Panel and (If the Control Panel is in "Category view") Select "Date, Time, Language and Regional options."
2. Open the "Regional and Language Options" icon.
3. Choose the "Languages" tab, and ensure the "Install files for East Asian languages" is checked.
4. Click the "Details" button to open the "Text services and input languages" dialog.
5. If "Chinese" is not listed in the "Installed Services" box, click "Add".
6. In the "Input Language" list, choose "Chinese (PRC)". The "Keyboard layout" should default to "Chinese (Simplified) - Microsoft Pinyin IME 3.0". Click "Ok" on both dialogs to return to the "Regional and Language Options".
7. Click "Ok", you will probably need to insert your Windows CD for the files to be installed.
8. The language bar should now have appeared in the bottom right of the taskbar. It should default to English - "EN".
9. Click on the "EN" button to show the available languages.
10. By changing the language to "CH", you can now type in Pinyin. You can change the current language by pressing "ALT" + "SHIFT" on your keyboard.

More information can be found at: <http://www.li-ming.org/Form/XPSetup.pdf>

Startup Wizard

At startup SQLiteManager allows you to perform frequently used operations:



Within the Startup Wizard you can:

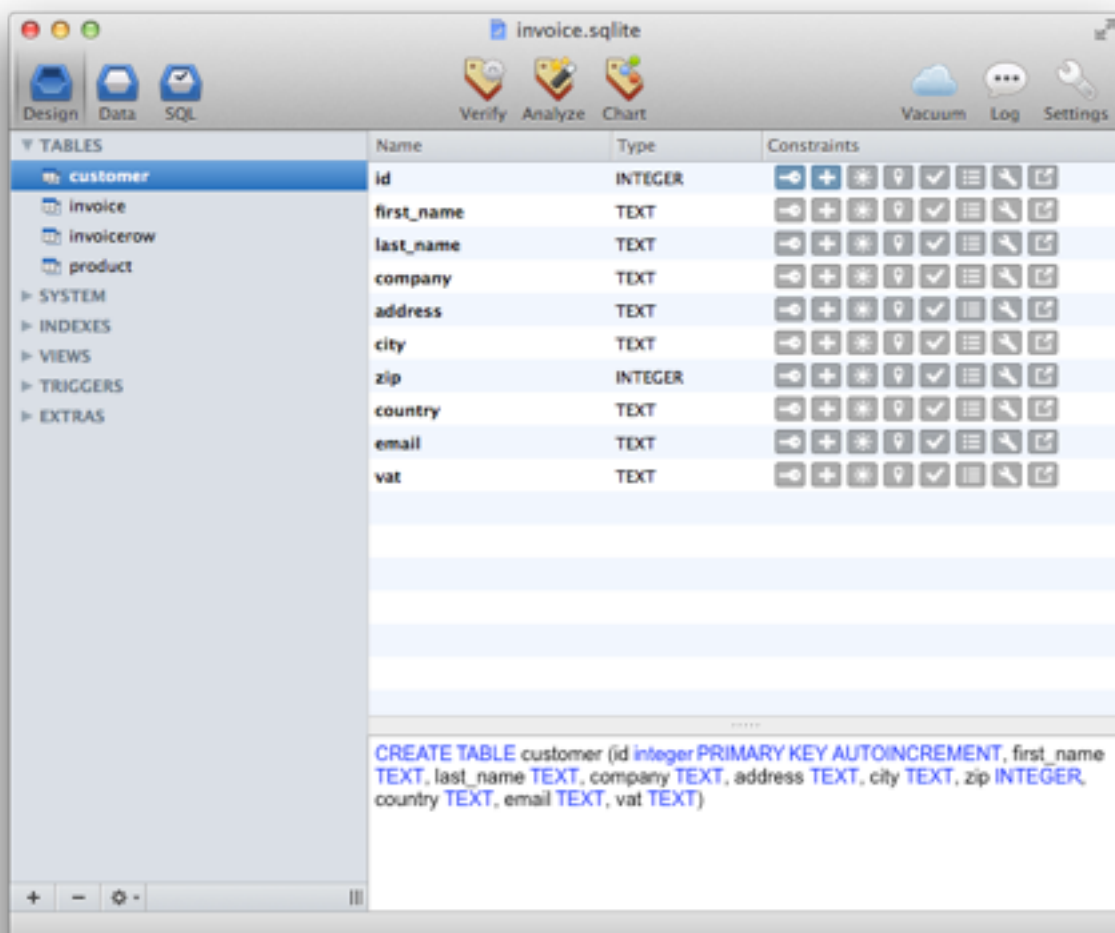
- Open an existing database
- Open a recently accessed database
- Create a new database (sqlite3, encrypted database or in-memory)
- Connect to a REAL Server or cubeSQL Server

Under MacOS X you can disable the Wizard using the Preferences window.

Design Panel

Overview

The Design panel is where you can create, inspect and update tables, views, indexes, triggers, SQL and Notes within a database. The panel is divided into two basic areas: an object browser on the left, information about selected objects is displayed on the right. The object browser is presented as a hierarchical list divided into Tables, System Tables, Views, Indexes, Triggers and Extras. Opening any one of those categories displays the objects of that type which can be found in the database. If a category doesn't have any objects, then no objects will be displayed. Selecting an item in a category reveals details about that item in the space to the right of the object browser.



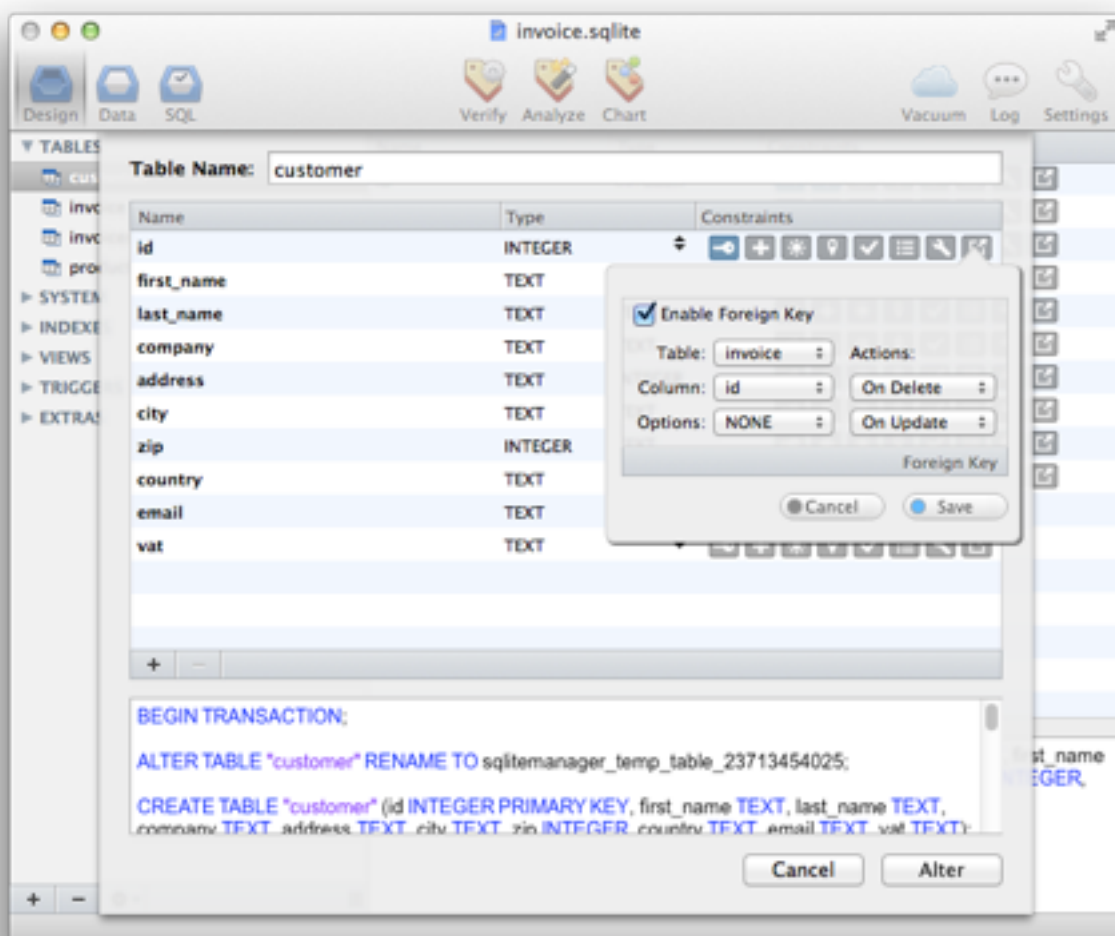
For example, selecting a table displays the schema for that table in a read-only edit field as well as a table containing more detail about the table's columns and the properties of those columns. Column properties

include the name, type and a series of icons which represent all the constraints set for the column. Selecting a view displays information about that view which is very much like the information which is displayed for a table. Note that the schema field for tables, views, and indexes is read-only, you can copy the text in the field and paste it elsewhere.

If you happen to find a table named "sqlitemanager_extras", we strongly recommend that you do not delete it, as you may lose functionality by doing so. On the other hand, deleting that table should not interfere with the rest of your SQLite database objects in any way. While in the Design tab, you may wish to drop tables, indexes, or views. To do so, select the object you wish to drop from the object browser and then choose "Drop" from the Edit menu or just push the "Delete" button at the bottom of the Object browser table.

Creating/Altering Tables

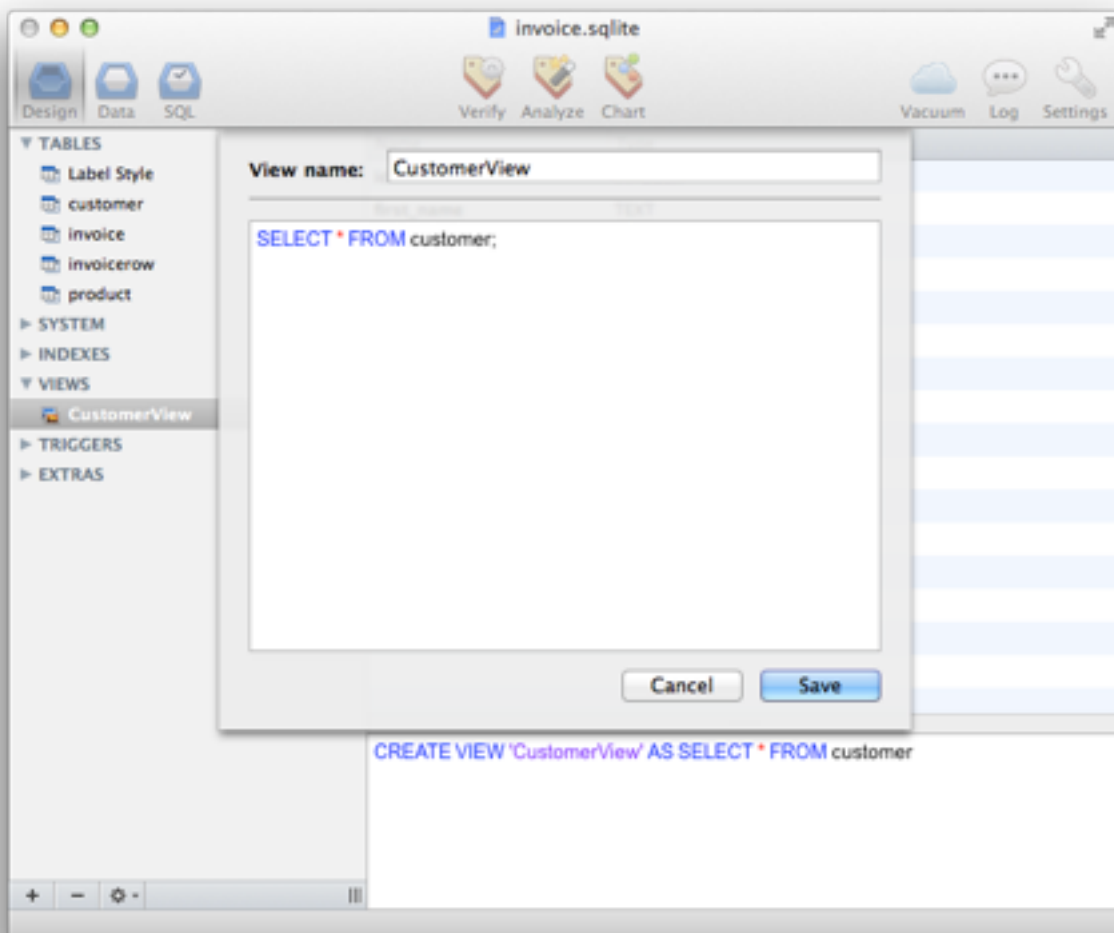
To create a new table in SQLiteManager, press the "Add Table" shortcut button at the bottom of the object browser. You will be presented with a new table dialog, pictured below. Within the new table dialog, you can name your table, and then add columns to it. You may also set the constraints for each column as displayed below.



Column types may either be typed in or set with the popup. The popup has the column types that SQLite and REALbasic understand because SQLite only supports four native datatype TEXT, INTEGER, REAL and BLOB. However, you are free to type in just about any type you'd like and sqlite will interpret it as TEXT. In case of ALTERING an existing table, SQLiteManager is smart enough to understand whether the ALTER TABLE operation is natively supported by sqlite or whether a special transaction is required. Live SQL preview offers you detailed information about the sql as it is generated.

Creating Views

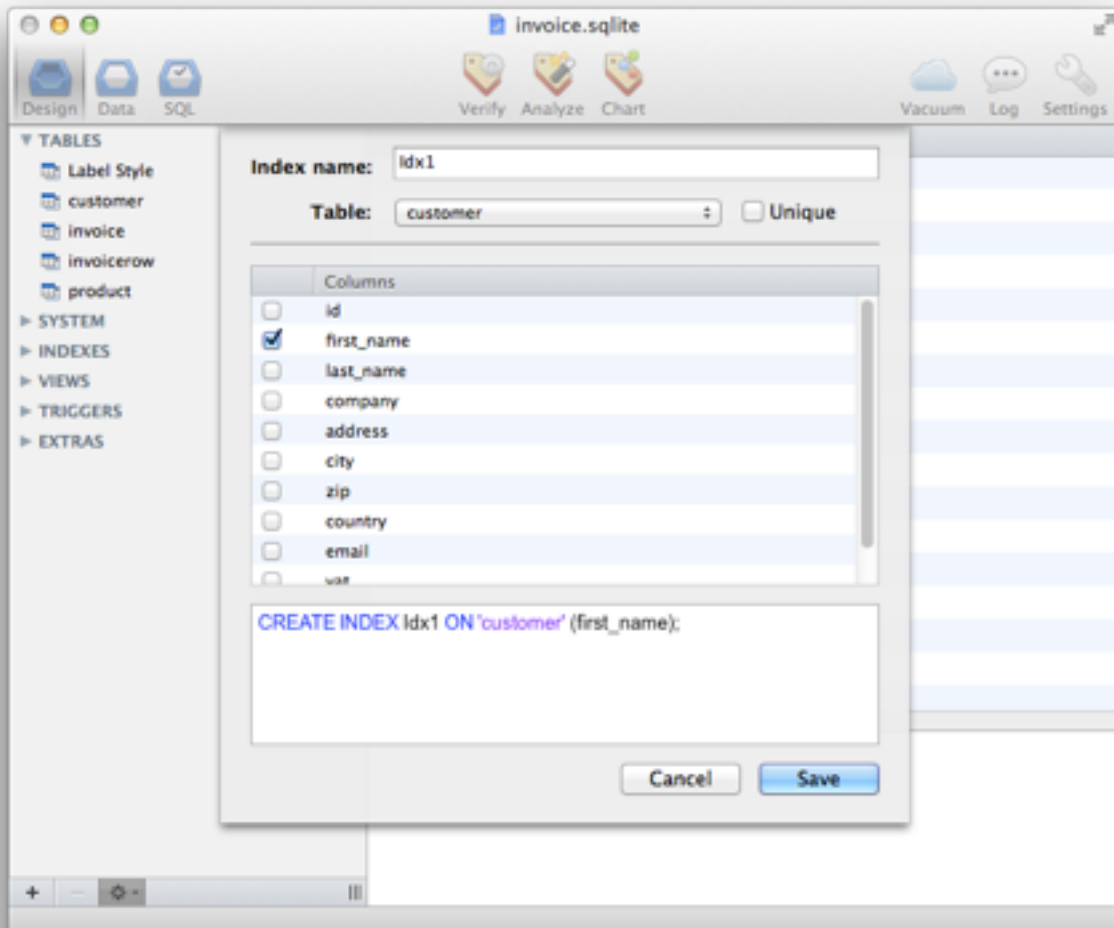
To create a new view in SQLiteManager, choose Create View from the Action button (at the bottom of the object browser). You will be presented with a new view dialog, pictured below. A view is basically a saved query (SELECT SQL statement, in other words).



SQLite treats views very much like read-only tables. You can query them as you would a table, and you can include them in other queries as well. You are free to name your view anything you would like. You may then type any arbitrary SELECT statement into the query field and that SELECT statement will become the query for the view.

Creating Indexes

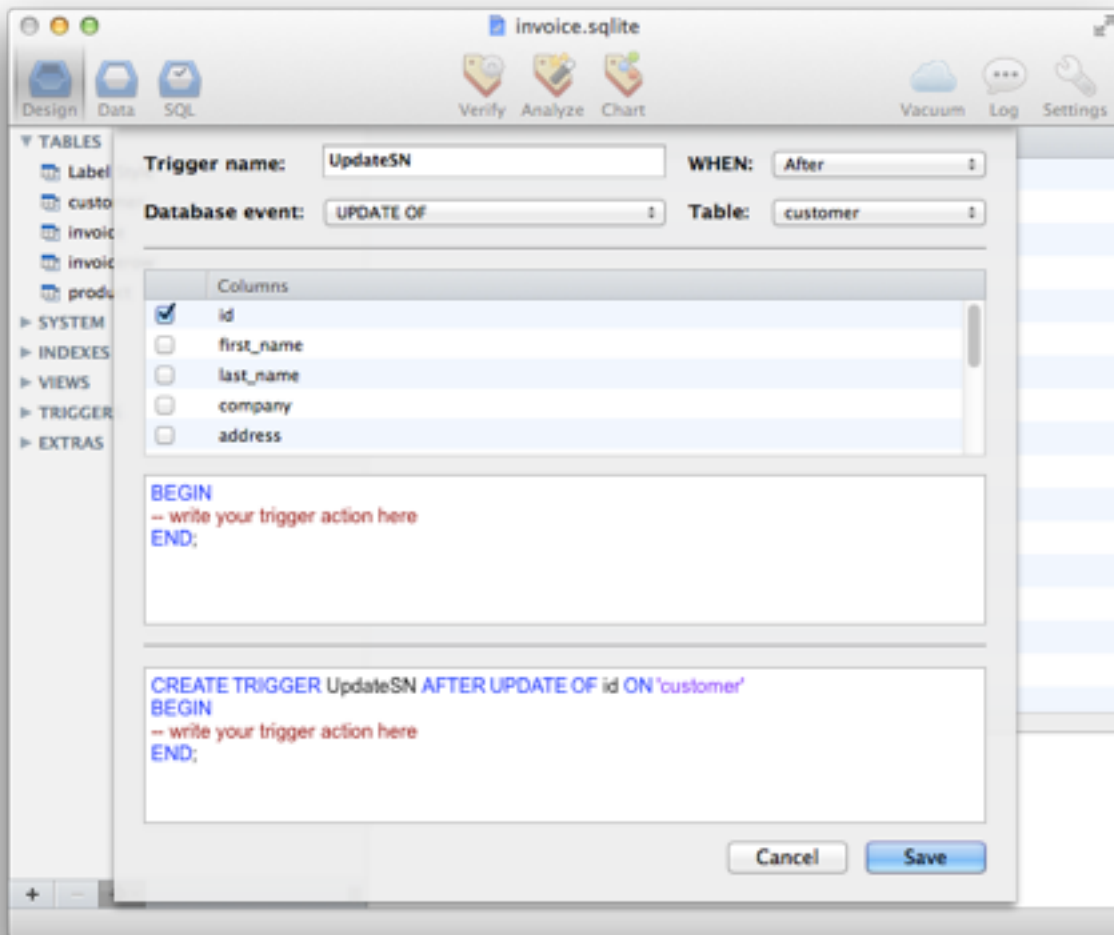
SQLite supports multiple indexes on a table and multiple columns in an index. To create a new index, choose Create Index from the Action button (at the bottom of the object browser). You will be presented with a new index dialog, pictured below.



The dialog allows you to name the index and choose the table for which the index will be created. Once you have chosen a table from the table popup menu, the ListBox will display the columns for that table. Check all columns that you wish to include in the index. You may also indicate whether or not the index is unique by checking the Unique checkbox.

Creating Triggers

SQLite fully supports triggers. To create a new trigger, choose Create Trigger from the Action button (at the bottom of the object browser) and you will be presented with a new trigger dialog, pictured below.

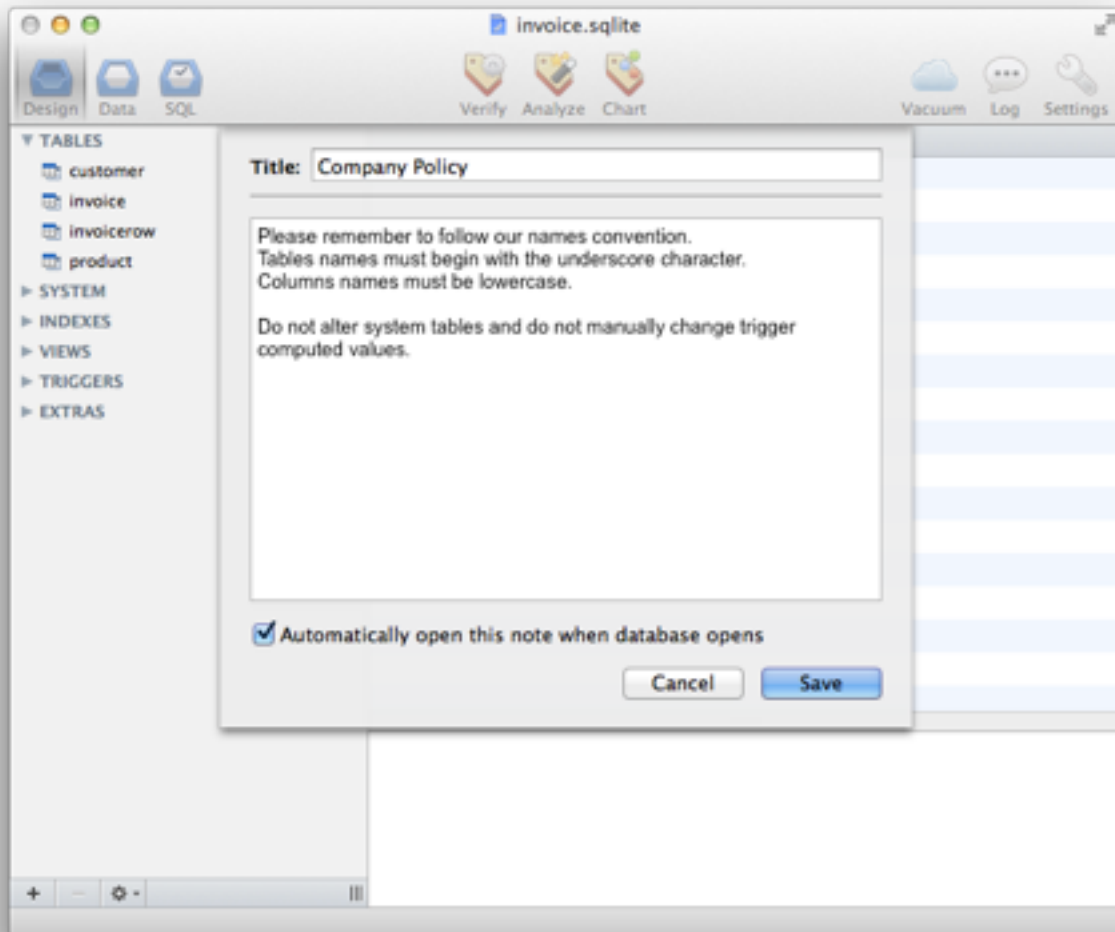


Triggers are database operations (the trigger-action) that are automatically performed when a specified database event (the database-event) occurs. A trigger may be specified to fire whenever a DELETE, INSERT or UPDATE of a particular database table occurs, or whenever an UPDATE of one or more specified columns of a table are updated. At this time SQLite supports only FOR EACH ROW triggers, not FOR EACH STATEMENT triggers.

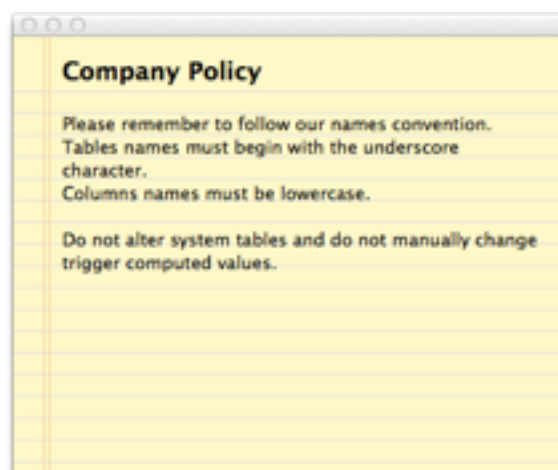
Hence explicitly specifying FOR EACH ROW is optional. FOR EACH ROW implies that the SQL statements specified as a trigger step may be executed (depending on the WHEN clause) for each database row being inserted, updated or deleted by the statement causing the trigger to fire.

Creating Notes

Choose Create Note from the Action button. You will be presented with a new view dialog, pictured below.



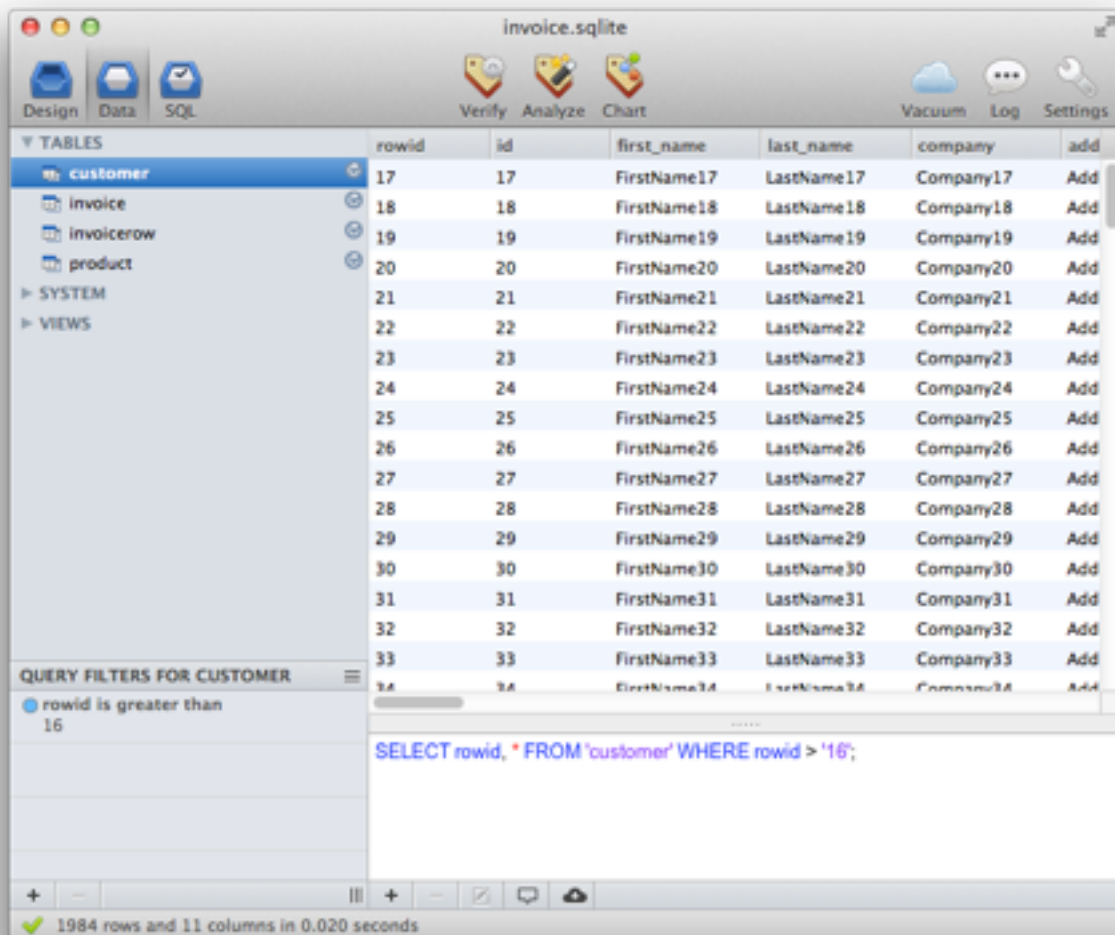
A note is basically a way to add comments and information to a database. You can optionally decide to automatically display the new note each time the database is opened by SQLiteManager. Note will be displayed with a sticky like style:



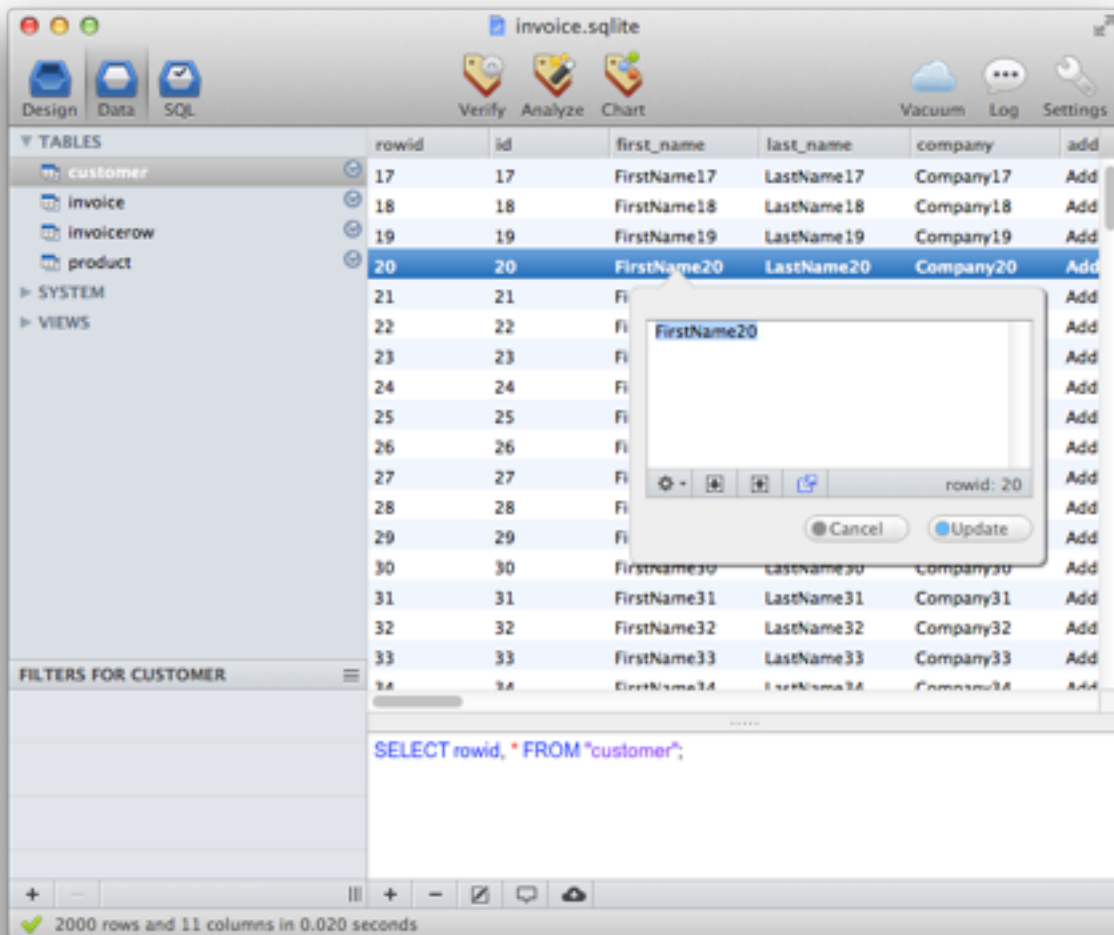
Panels

Data Panel

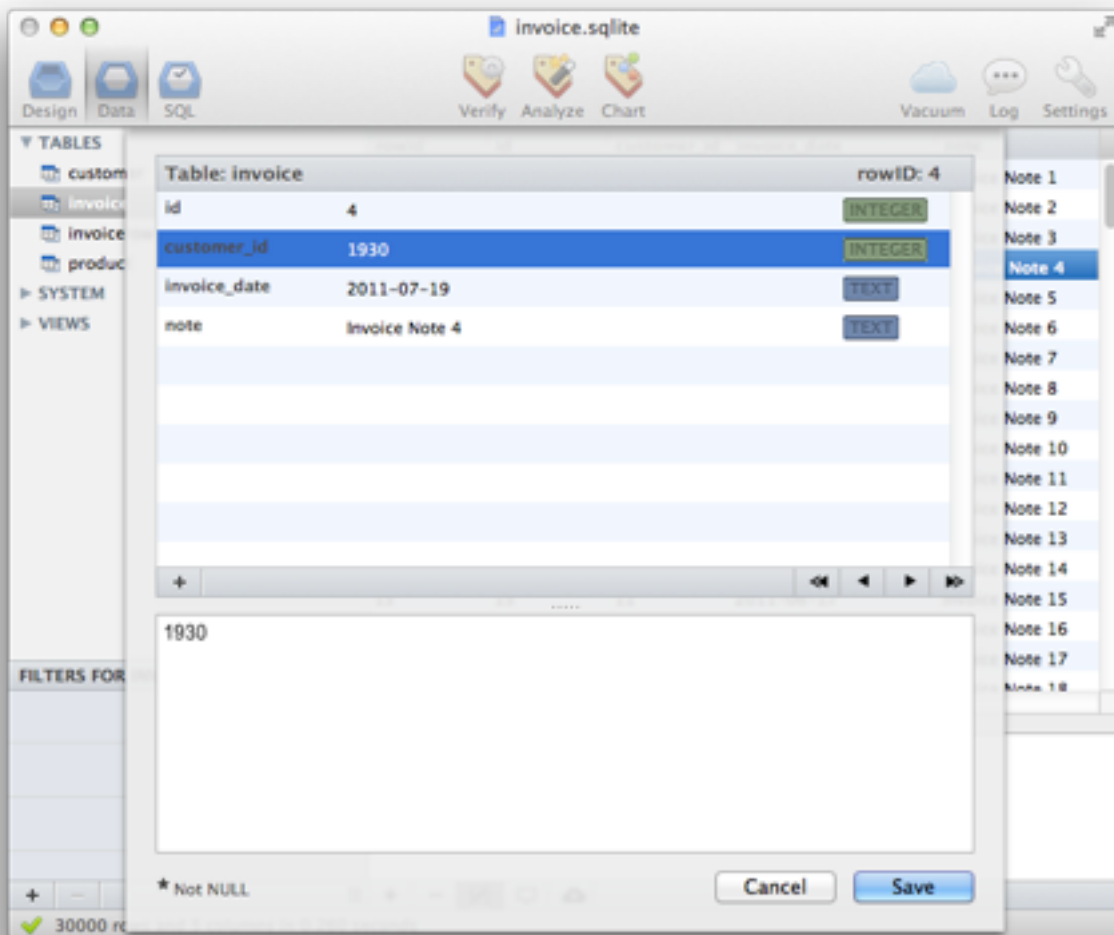
The Data panel is used when you want to insert, remove, or edit records in a table. The tab is divided into three main areas. At the top of the tab are the popup menus and edit fields for building a query. Below that is a read-only edit field where the SQL for the query is displayed. You are free to copy this SQL and paste it elsewhere, but you cannot type arbitrary SQL into that field.



Below the edit field is a list box containing the results of a query. To build a query, simply construct a SELECT statement out of the popups and edit fields and click the Query button. The query results will be displayed in the list box. To remove a record from the queried table, select it and click the Remove button. To edit a record, double-click it in the list box and the Record Editor dialog will appear (if inline editing has been disabled in the Preferences).



When you open a record to edit it (just double click on it if inline editing is disabled or select the record and then press the modify button), the fields of the record will be presented in a list box. Values of the column can be changed either using inline editing on the Value column or using the expanded panel and the text box on the bottom. Please note that you can also use some powerful contextual menu options when you are in the Name/Values listbox. Upon completing the changes you would like, click the Save button. The changes will be saved in the database immediately. Alternatively, click Cancel and the changes will be discarded. This powerful dialog gives you the option to display current fields as TEXT, various graphical formats like JPEG, TIFF, BMP and so on (it depends on the platform) and to show it as a raw BinHex image.

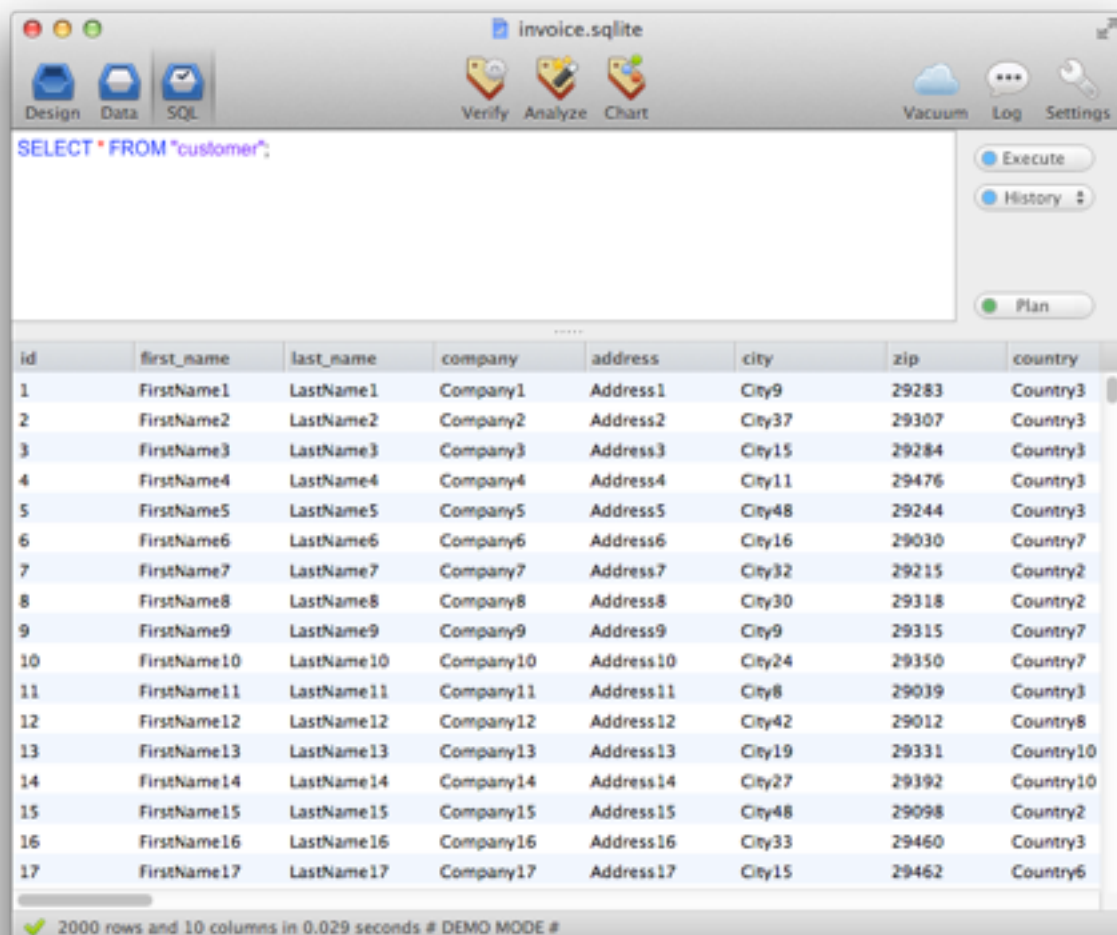


To insert a new record into a table in the database, click the Insert button (represented by a + icon) in the Manage tab of SQLiteManager's main window. The dialog used for inserting records is exactly the same as that used for editing records.

SQL Panel

The SQL panel is used to enter an arbitrary SQL allowing you to perform complex queries and commands on an SQLite database. The tab is divided into two main sections: a large edit field for typing SQL at the top of the tab and a list box for displaying results at the bottom of the tab. To use the SQL tab, just type any SQL into the edit field and click either the Execute/Query button.

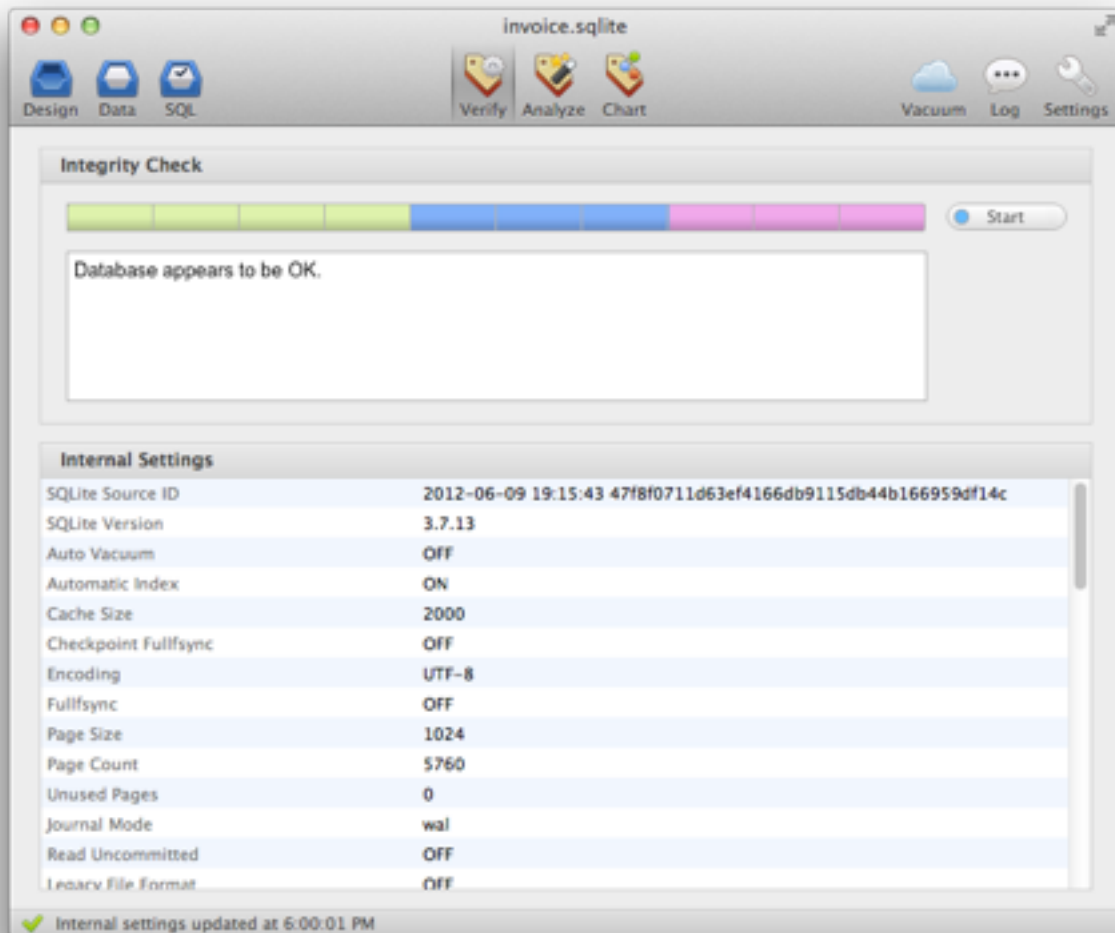
If you want to display the results from the query, click the Select button. If you type an SQL statement that you'd like to save for use at another time, just click the left mouse button and a popup menu will appear allowing you to save the SQL in the database or retrieving a previously saved one.



If you want your custom queries to be editable then you have to make sure that the rowid column or the primary key column is returned in your result set. The Plan button allows you to obtain a high-level description of the strategy or plan that sqlite uses to implement a specific SQL query. Most significantly, it reports on the way in which the query uses database indices.

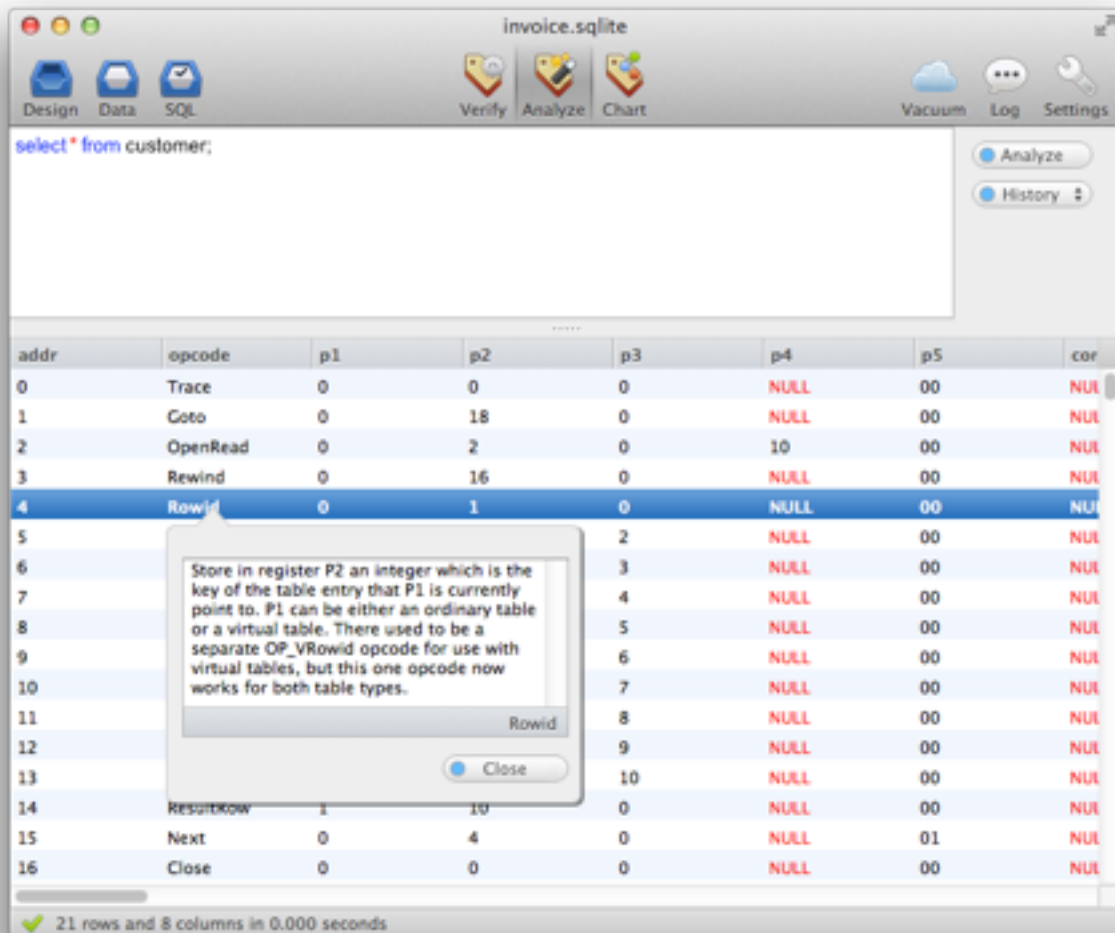
Verify Panel

You will visit the Verify panel when you want to perform a sanity check over your entire database or when you want to examine all the most important internal database settings in a very easy way. The Integrity Checker checks for out of orders records, for missing pages, for malformed records and for corrupted indexes. A detailed reports is shown if some problems are found inside the database.



Analyze Panel

You will visit the Analyze panel when you want to examine and disassemble SQL commands into low level virtual machine instructions. This is the most powerful way to check if a query or a SQL statement is efficient or if it can be rewritten in a better way. Each virtual machine opcode has a very detailed description in a convenient popover window.



The screenshot shows the SQLite Manager interface with the 'Analyze' panel selected. The query 'select * from customer;' is entered in the top text area. Below the query, a table displays the virtual machine instructions (opcodes) for the query. The table has columns: addr, opcode, p1, p2, p3, p4, p5, and cor. The current row selected is address 4, opcode 'Rowid', with p1=0 and p2=1. A popover window is open over this row, providing a detailed description of the 'Rowid' opcode.

addr	opcode	p1	p2	p3	p4	p5	cor
0	Trace	0	0	0	NULL	00	NULL
1	Goto	0	18	0	NULL	00	NULL
2	OpenRead	0	2	0	10	00	NULL
3	Rewind	0	16	0	NULL	00	NULL
4	Rowid	0	1	0	NULL	00	NULL
5				2	NULL	00	NULL
6				3	NULL	00	NULL
7				4	NULL	00	NULL
8				5	NULL	00	NULL
9				6	NULL	00	NULL
10				7	NULL	00	NULL
11				8	NULL	00	NULL
12				9	NULL	00	NULL
13				10	NULL	00	NULL
14	ResultRow	1	10	0	NULL	00	NULL
15	Next	0	4	0	NULL	01	NULL
16	Close	0	0	0	NULL	00	NULL

Store in register P2 an integer which is the key of the table entry that P1 is currently point to. P1 can be either an ordinary table or a virtual table. There used to be a separate OP_VRowid opcode for use with virtual tables, but this one opcode now works for both table types.

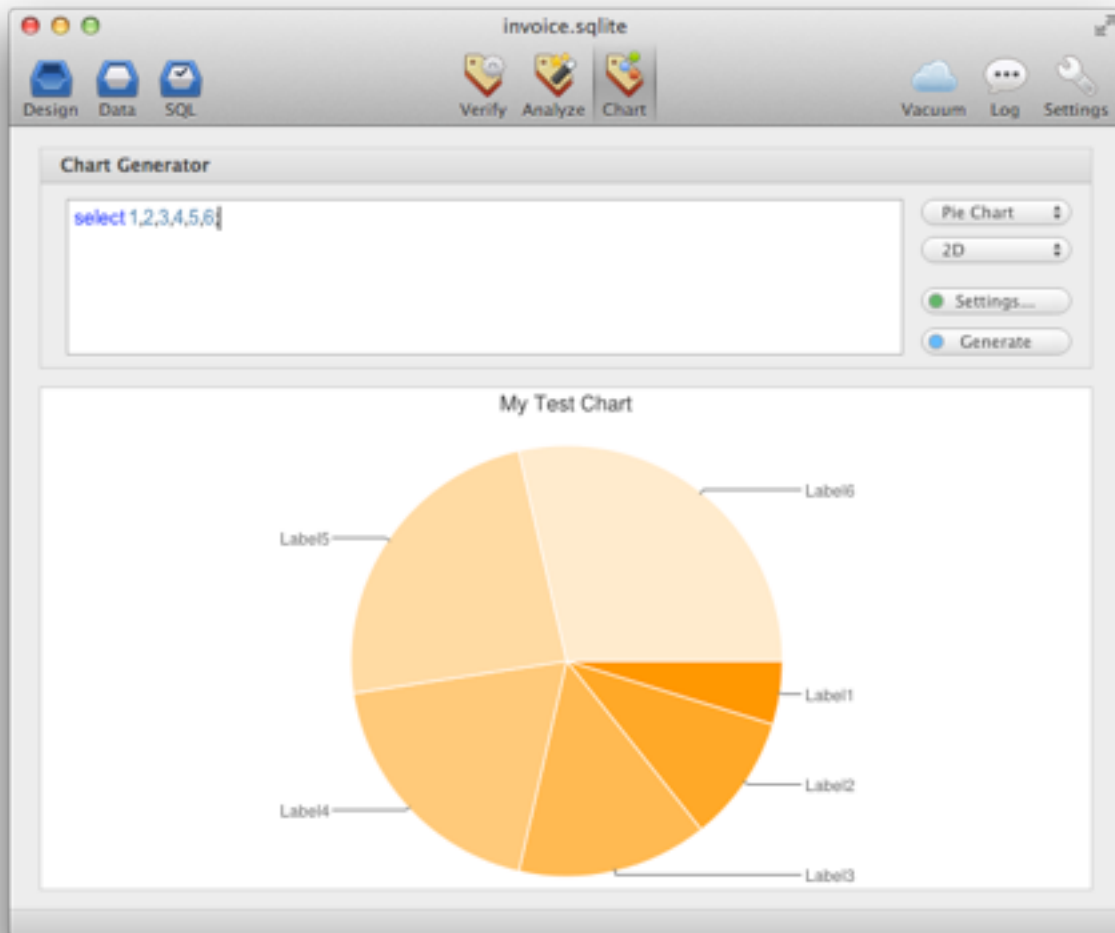
Rowid

Close

21 rows and 8 columns in 0.000 seconds

Chart Panel

The Chart Panel allows you to easily visualize your queries in fully exportable PNG charts. Many charts are supported like Line Chart, Bar Chart, Pie Chart, Venn Chart, Scatter, Radar and even QR Code. Create an sql query which returns numbers and you can plot your data in 2D and 3D. The Settings button allows you to customize many aspects of the chart.



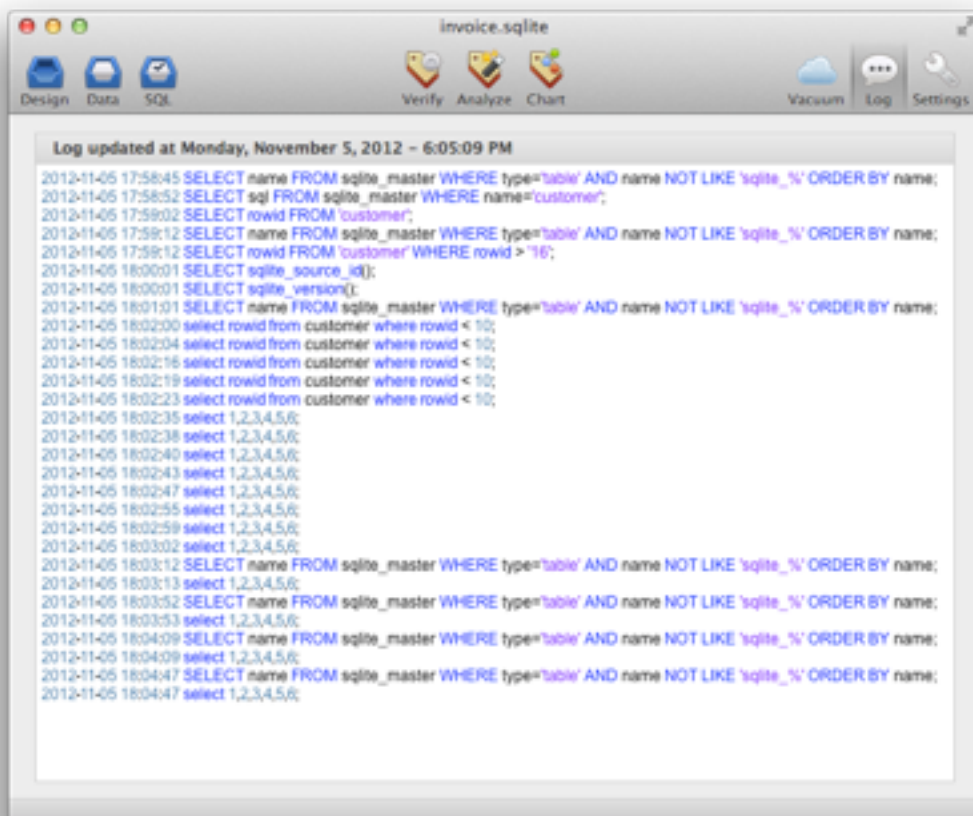
Vacuum, Log and Settings

Vacuum

When an object is dropped from the database, it leaves behind empty space. This makes the database file larger than it needs to be, but can speed up inserts. In time inserts and deletes can leave the database file structure fragmented, which slows down disk access to the database contents. The VACUUM command cleans the main database by copying its contents to a temporary database file and reloading the original database file from the copy. This eliminates free pages, aligns table data to be contiguous, and otherwise cleans up the database file structure.

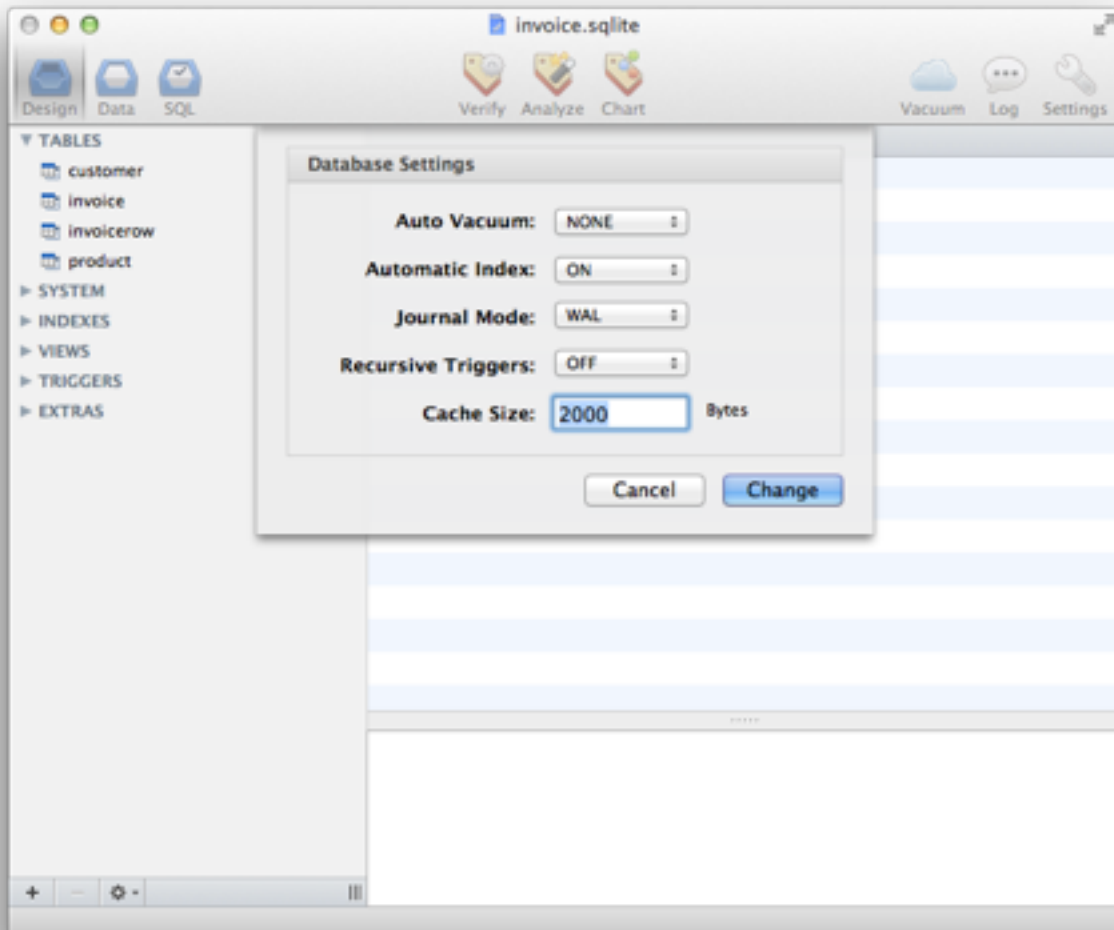
Log Panel

Every time an sql statement is executed, SQLiteManager automatically records the statement within a separate log panel, allowing you to take a look and understand precisely what has happened to the data within your database.



Settings

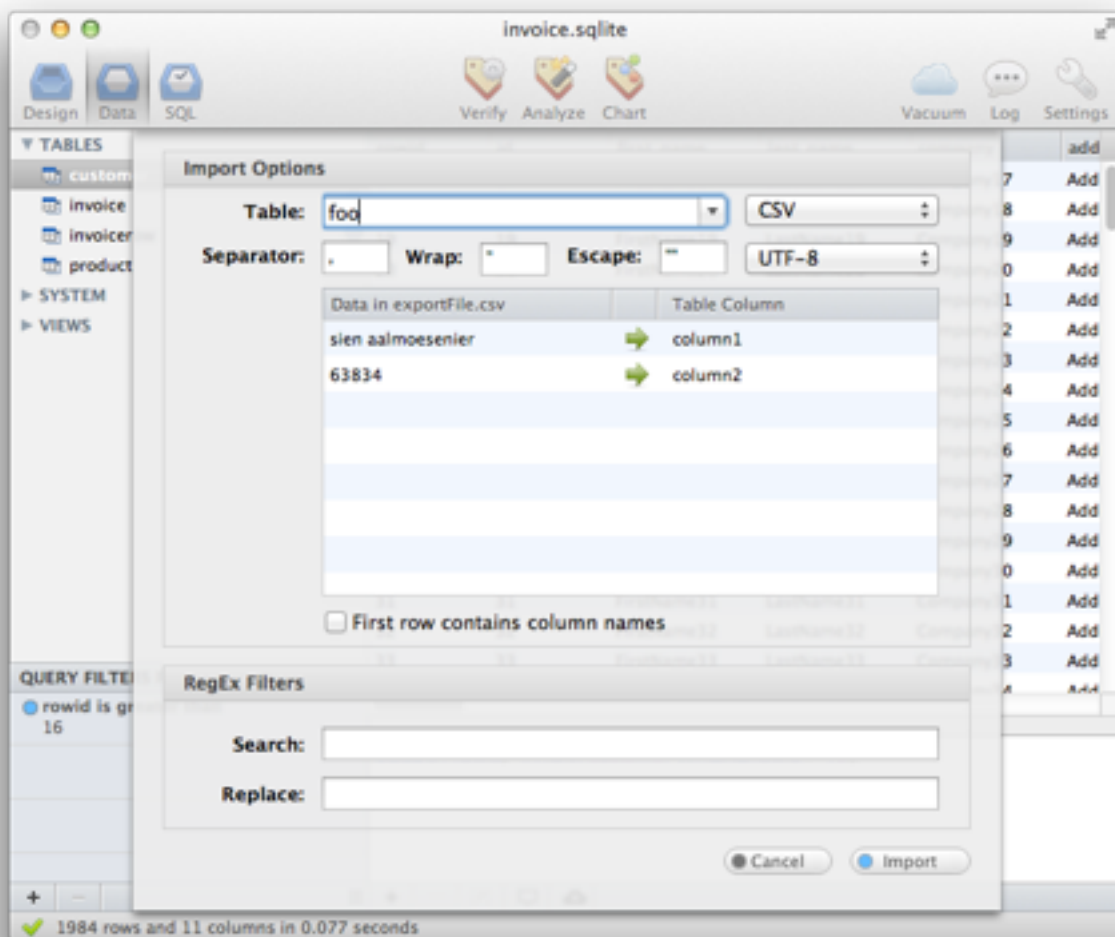
The Setting panel is where you inspect or change the most important internal database settings. Please make sure you completely understand what are you trying to change because these settings can really affect your database performance.



Import and Export

Importing Data

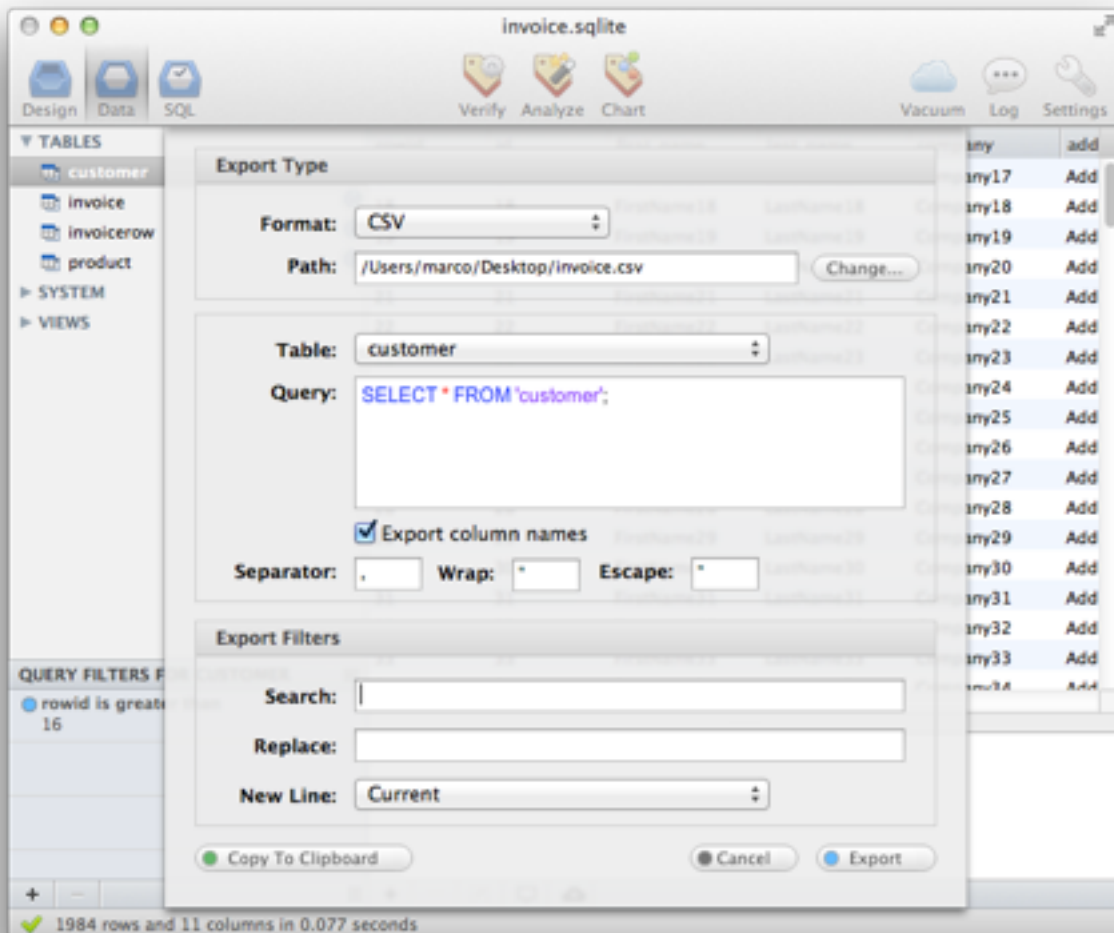
To import data from an open database, choose Import from the File menu. Importing capabilities are a very important aspect for any database management system. We have developed this feature to be flexible enough to satisfy all of your needs while maintaining an easy to use interface which can be setup within a matter of seconds.



If you choose to import SQL, then you will not be prompted for a table in which to import the data, as SQL statements that insert data into tables refer to those tables directly. SQL statements can be in the file you import, including statements to create tables and indexes etc. If you wanted to recreate an entire database, you could create a new, empty database in SQLiteManager, and then import the SQL that would create and populate all of the tables within that database.

Exporting Data

SQLiteManager can export data in several common formats. The currently supported formats are CSV, SQL, tab-delimited and custom character delimited. To export data from an open database, choose Export from the File menu.



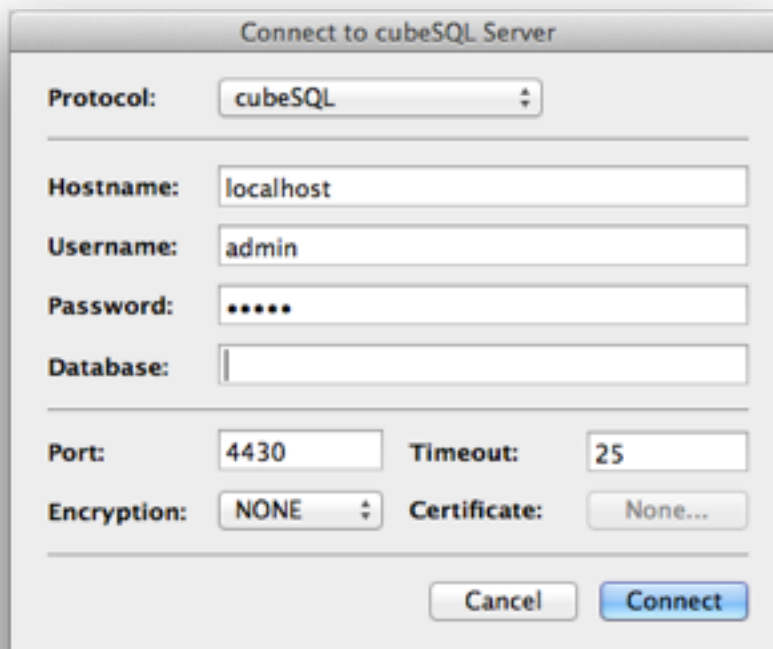
Upon exporting, a dialog appears, asking which table in the database you wish to export, which format to use, which encodings and so on. Each type of export includes the table header as the first row of the exported data. Choosing an SQL export will display the dialog pictured below. Select which tables, views, and indexes you wish to export. You can choose to export only the CREATE statements by unchecking the “Export data” checkbox.

You can also select the new line type (MacOS, Windows, Linux or Current) and apply a special RegEx search/replace pattern to “transform” your data before exporting. Although SQLiteManager can export in CSV, SQL, and tab-delimited formats automatically for you, you have virtually unlimited exporting options when you use the built-in Lua scripting language.

Others

cubeSQL

cubeSQL Server is a powerful and fully featured SQL server build upon SQLite 3 database technology. SQLiteManager can easily connect to a remote server and use its database in the same manner as a normal local database file. For a better experience, an admin account is recommended when you try to manage a remote database.

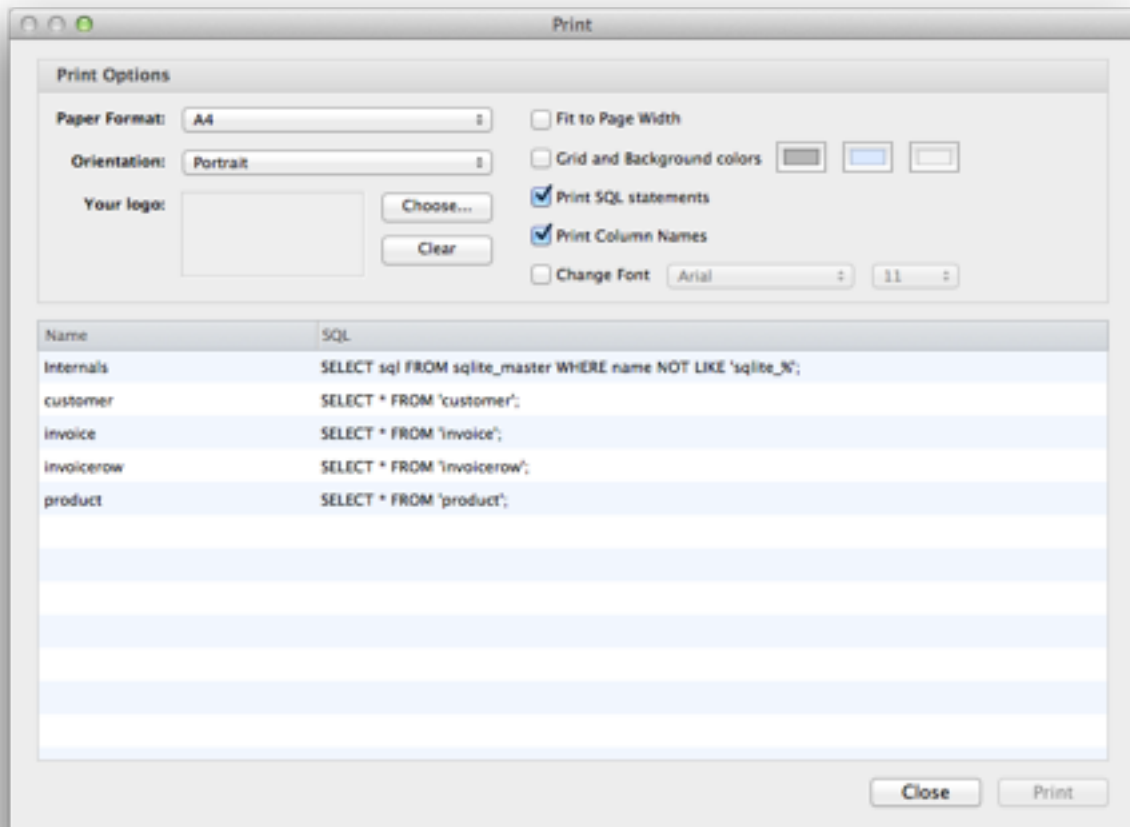


The image shows a dialog box titled "Connect to cubeSQL Server". It contains several input fields and buttons. The "Protocol" field is a dropdown menu set to "cubeSQL". The "Hostname" field is a text box containing "localhost". The "Username" field is a text box containing "admin". The "Password" field is a text box with masked characters "*****". The "Database" field is an empty text box. Below these fields, there are two rows of fields: "Port" (text box with "4430") and "Timeout" (text box with "25"); "Encryption" (dropdown menu set to "NONE") and "Certificate" (button labeled "None..."). At the bottom right are two buttons: "Cancel" and "Connect".

If you leave the Database field blank, SQLiteManager will automatically retrieve all the databases available on the server, simply double click the one you want to connect to.

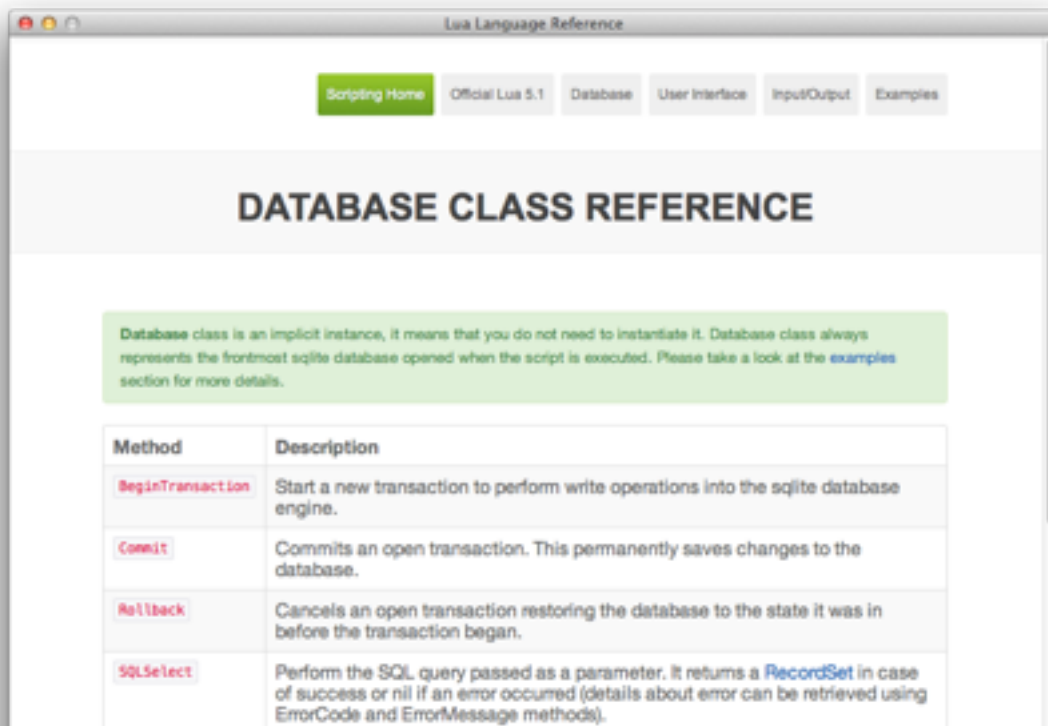
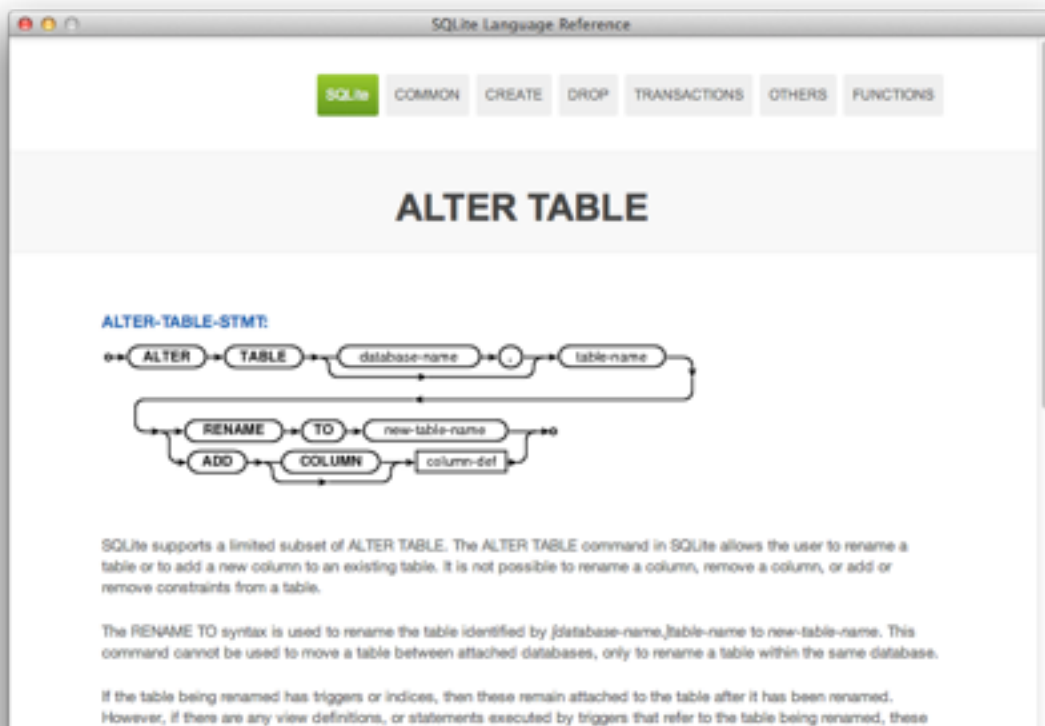
Printing

Thanks to a powerful print engine you can customize and preview all your prints. BLOB images are automatically recognized and printed and the print engine is driven by simple SQL queries which you can customize at any time (built-in PDF exporting capabilities is also available on both MacOS X and Windows).



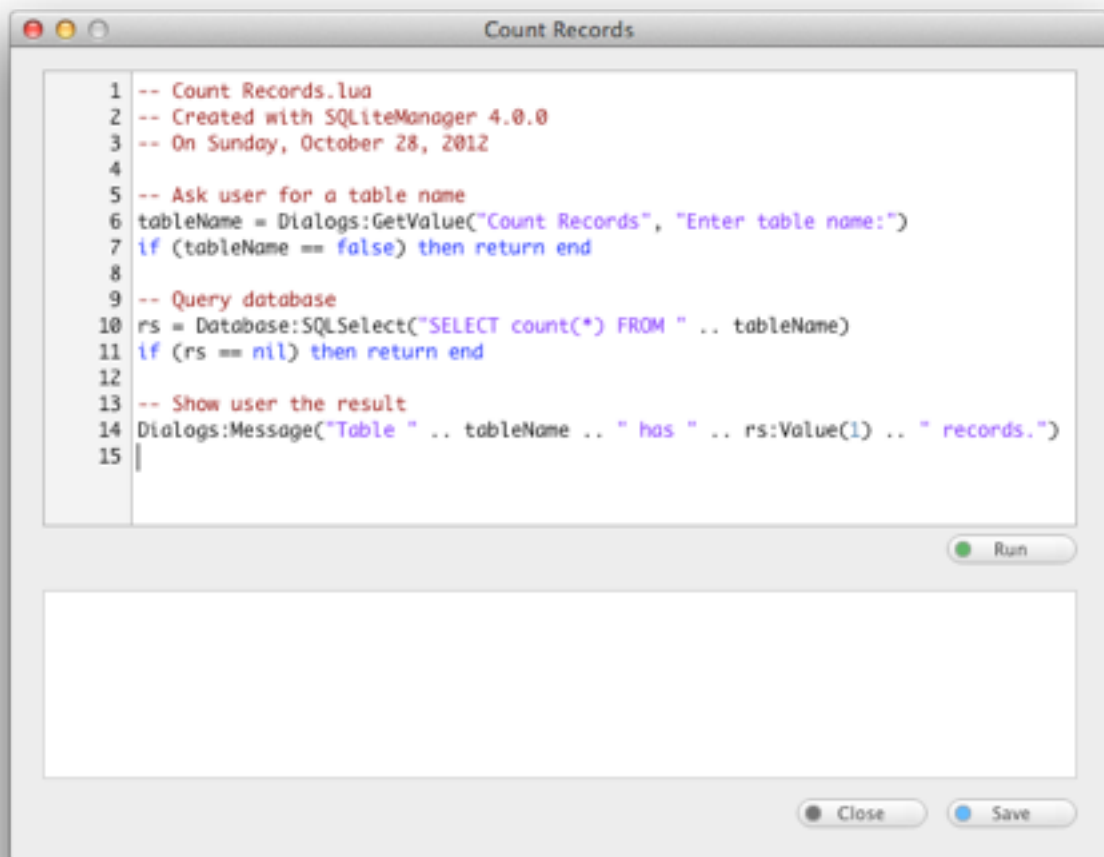
Inline Help

SQLiteManager has a new built-in SQLite and Lua scripting language inline help accessible from the Help menu.



Scripting Language

A built-in scripting language based on Lua enables you to write plugins and to automate repetitive tasks inside SQLiteManager. You can insert records in loops, download data from the Internet, interact with the user etc.



Appendix

Contact Information

SQLabs srl

Via Piave n.21

46019 Viadana (MN) - ITALY

<http://www.sqlabs.com>

info@sqlabs.com

Copyright

All materials are copyright 2003-2013 by SQLabs.

All Rights Reserved. SQLiteManager may be freely distributed, so long as it is not sold for profit, and registration serial numbers are not distributed. Express permission is granted to online services and other shareware/public domain distribution avenues to carry SQLiteManager.

Express permission is further granted to include SQLiteManager on CD-ROMs or floppy disks accompanying books, or on shareware collections, provided that no more than a nominal compilation and/or media fee is charged for these collections.

Legal Stuff

Unregistered copies may be used and evaluated in demonstration mode. Copies are registered per individual developer and may be used by that developer, royalty-free, in an unlimited number of applications, commercial or otherwise. Once obtained, licenses may not be transferred to other individuals or organizations.

SQLabs reserves the right to revoke the license of anyone who ignores or violates these restrictions. See our web site at <http://www.sqlabs.com/> for registration information. Company licenses are available.

SQLiteManager is distributed AS IS. There is no warranty of any kind as to its fitness for any purpose. The risks associated with the use of this product are borne by the user in their entirety. In other words, the SQLiteManager, although it is in no way designed to do so, could be capable of ruining your software, crashing your computer and erasing your hard drive. Marco Bambini and SQLabs take no responsibility for these or other consequences.

REALbasic® it is a registered trademark of REAL Software, Inc. See their web site at

<http://www.realsoftware.com> for more details. SQLabs is not way affiliated with REAL Software. All questions regarding REALbasic should be directed to REAL Software, Inc.