

# How To Use The SQLite Dump Command

**Summary:** in this tutorial, you will learn how to use the SQLite dump command to back up and restore a database.

SQLite project delivers the `sqlite3` tool that allows you to interact with the SQLite database using a command-line program.

By using the `sqlite3` tool, you can use the SQL statements to query or update data in the database. Also, you can use special commands, which are known as dot commands to perform various useful database operations.

One of these dot-commands is the `.dump` command that gives you the ability to dump the entire database or tables into a text file.

## Dump the entire database into a file using the SQLite dump command

The following command opens a new SQLite database connection to the `chinook.db` file.

```
C:\sqlite>sqlite3 c:/sqlite/chinook.db
SQLite version 3.13.0 2016-05-18 10:57:30
Enter ".help" for usage hints.
sqlite>
```

Code language: JavaScript (javascript)

To dump a database into a file, you use the `.dump` command. The `.dump` command converts the entire structure and data of an SQLite database into a single text file.

By default, the `.dump` command outputs the SQL statements on screen. To issue the output to a file, you use the `.output FILENAME` command.

The following commands specify the output of the dump file to `chinook.sql` and dump the `chinook` database into the `chinook.sql` file.

```
sqlite> .output c:/sqlite/chinook.sql
sqlite> .dump
sqlite> .exit
```

Code language: JavaScript (javascript)

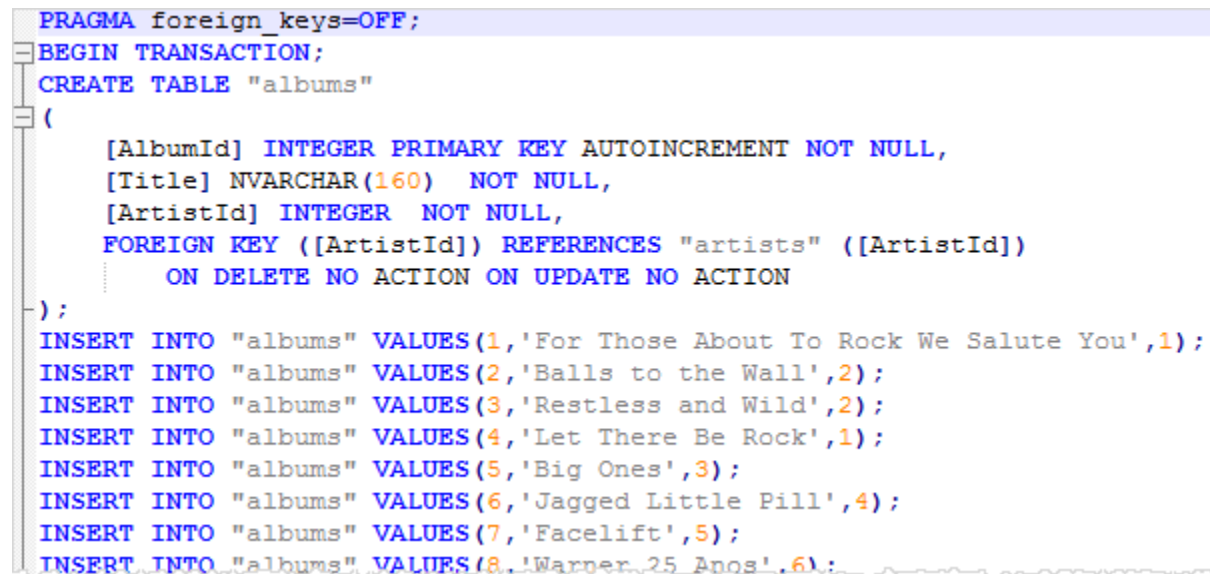
## Dump a specific table using the SQLite dump command

To dump a specific table, you specify the table name after the .dump command. For example, the following command saves the albums table to the albums.sql file.

```
sqlite> .output c:/sqlite/albums.sql
sqlite> .dump albums
sqlite> .quit
```

Code language: JavaScript (javascript)

The following picture shows the contents of the albums.sql file.



```
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE "albums"
(
    [AlbumId] INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
    [Title] NVARCHAR(160) NOT NULL,
    [ArtistId] INTEGER NOT NULL,
    FOREIGN KEY ([ArtistId]) REFERENCES "artists" ([ArtistId])
        ON DELETE NO ACTION ON UPDATE NO ACTION
);
INSERT INTO "albums" VALUES(1,'For Those About To Rock We Salute You',1);
INSERT INTO "albums" VALUES(2,'Balls to the Wall',2);
INSERT INTO "albums" VALUES(3,'Restless and Wild',2);
INSERT INTO "albums" VALUES(4,'Let There Be Rock',1);
INSERT INTO "albums" VALUES(5,'Big Ones',3);
INSERT INTO "albums" VALUES(6,'Jagged Little Pill',4);
INSERT INTO "albums" VALUES(7,'Facelift',5);
INSERT INTO "albums" VALUES(8,'Warner 25 Anos',6);
```

## Dump tables structure only using schema command

To dump the table structures in a database, you use the .schema command.

The following commands set the output file to chinook\_structure.sql file and save the table structures into the chinook\_structure.sql file:

```
sqlite> .output c:/sqlite/chinook_structure.sql
sqlite> .schema
sqlite> .quit
```

Code language: JavaScript (javascript)

The following picture shows the content of the chinook\_structure.sql file.

```

CREATE TABLE "albums"
(
    [AlbumId] INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
    [Title] NVARCHAR(160) NOT NULL,
    [ArtistId] INTEGER NOT NULL,
    FOREIGN KEY ([ArtistId]) REFERENCES "artists" ([ArtistId])
        ON DELETE NO ACTION ON UPDATE NO ACTION
);
CREATE TABLE "artists"
(
    [ArtistId] INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
    [Name] NVARCHAR(120)
);
CREATE TABLE "customers"
(

```

## Dump data of one or more tables into a file

To dump the data of a table into a text file, you use these steps:

First, set the mode to insert using the `.mode` command as follows:

```
sqlite> .mode insert
```

Code language: CSS (css)

From now on, every [SELECT statement](#) will issue the result as the [INSERT statements](#) instead of pure text data.

Second, set the output to a text file instead of the default standard output. The following command sets the output file to the `data.sql` file.

```
sqlite> .output data.sql
```

Code language: CSS (css)

Third, issue the [SELECT](#) statements to query data from a table that you want to dump. The following command returns data from the `artists` table.

```
sqlite> select * from artists;
```

Code language: SQL (Structured Query Language) (sql)

Check the content of the `data.sql` file, if everything is fine, you will see the following output:

```
INSERT INTO table VALUES(1,'AC/DC');  
INSERT INTO table VALUES(2,'Accept');  
INSERT INTO table VALUES(3,'Aerosmith');  
INSERT INTO table VALUES(4,'Alanis Morissette');  
INSERT INTO table VALUES(5,'Alice In Chains');  
INSERT INTO table VALUES(6,'Antônio Carlos Jobim');  
INSERT INTO table VALUES(7,'Apocalyptica');  
INSERT INTO table VALUES(8,'Audioslave');  
INSERT INTO table VALUES(9,'BackBeat');  
INSERT INTO table VALUES(10,'Billy Cobham');
```

To dump data from other tables, you need to issue the SELECT statements to query data from those tables.

# Import a CSV File Into an SQLite Table

**Summary:** in this tutorial, you will learn various ways to import CSV data into an SQLite table using sqlite3 and SQLite Studio tools.

## Importing a CSV file into a table using sqlite3 tool

In the first scenario, you want to import data from CSV file into a table that does not exist in the SQLite database.

1. First, the sqlite3 tool creates the table. The sqlite3 tool uses the first row of the CSV file as the names of the columns of the table.
2. Second, the sqlite3 tool import data from the second row of the CSV file into the table.

We will import a CSV file named `city.csv` with two columns: name and population. You can download it here for practicing.

[Download the city.csv file](#)

To import the `c:\sqlite\city.csv` file into the cities table:

First, set the mode to CSV to instruct the command-line shell program to interpret the input file as a CSV file. To do this, you use the `.mode` command as follows:

```
sqlite> .mode csv
```

Second, use the command `.import FILE TABLE` to import the data from the `city.csv` file into the cities table.

```
sqlite>.import c:/sqlite/city.csv cities
```

To verify the import, you use the command `.schema` to display the structure of the cities table.

```
sqlite> .schema cities
CREATE TABLE cities(
  "name" TEXT,
  "population" TEXT
);
```

Code language: SQL (Structured Query Language) (sql)

To view the data of the cities table, you use the following SELECT statement.

```
SELECT
```

```
name,  
population  
FROM  
cities;
```

Code language: SQL (Structured Query Language) (sql)

In the second scenario, the table is already available in the database and you just need to import the data.

First, [drop](#) the cities table that you have created.

```
DROP TABLE IF EXISTS cities;
```

Code language: SQL (Structured Query Language) (sql)

Second, use the following [CREATE TABLE](#) statement to create the table cities.

```
CREATE TABLE cities(  
    name TEXT NOT NULL,  
    population INTEGER NOT NULL  
);
```

Code language: SQL (Structured Query Language) (sql)

If the table already exists, the sqlite3 tool uses all the rows, including the first row, in the CSV file as the actual data to import. Therefore, you should delete the first row of the CSV file.

The following commands import the city\_without\_header.csv file into the cities table.

```
sqlite> .mode csv  
sqlite> .import c:/sqlite/city no header.csv cities
```

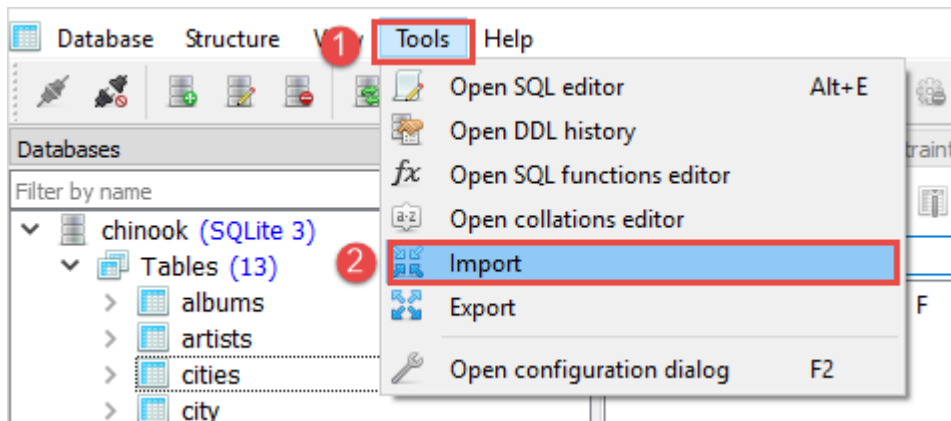
Code language: SQL (Structured Query Language) (sql)

## Import a CSV file into a table using SQLite Studio

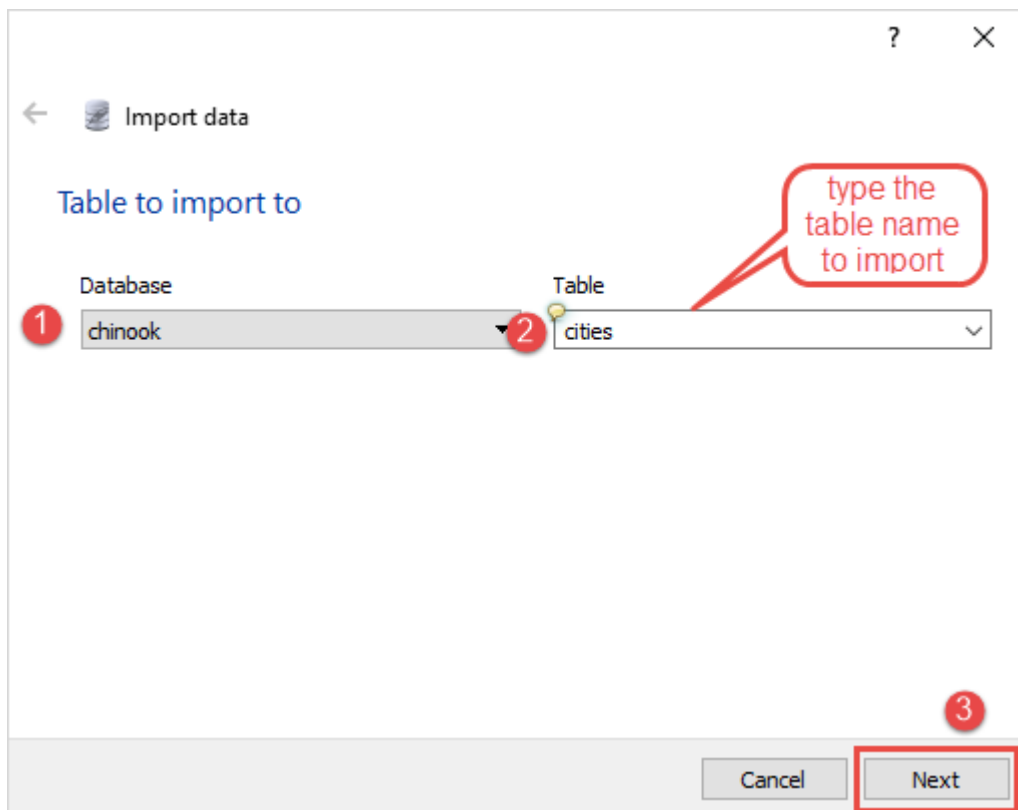
Most SQLite GUI tools provide the import function that allows you to import data from a file in CSV format, tab-delimited format, etc., into a table.

We will use the SQLite Studio to show you how to import a CSV file into a table with the assumption that the target table already exists in the database.

First, from the menu choose tool menu item.



Second, choose the database and table that you want to import data then click the **Next** button.



Third, choose CSV as the data source type, choose the CSV file in the **Input file** field, and choose the **,(comma)** option as the **Field separator** as shown in the picture below. Then click the **Finish** button to import the data.

The screenshot shows the 'Import data' dialog box with the following elements and annotations:

- 1**: A red circle next to the 'Data source type' dropdown menu, which is set to 'CSV'.
- 2**: A red circle next to the 'Input file' text box, which contains the path 'C:/sqlite/city\_no\_header.csv'.
- 3**: A red circle next to the 'Field separator' dropdown menu, which is set to ', (comma)'.
- 4**: A red circle next to the 'Finish' button.

Other visible elements include the 'Options' section with 'Text encoding' set to 'windows-1258' and an 'Ignore errors' checkbox, and the 'Data source options' section with a checkbox for 'First line represents CSV column names' and a 'NULL values' field.

## Export SQLite Database To a CSV File

**Summary:** in this tutorial, you will learn how to export SQLite database to a CSV file.

There are several ways to dump data from an SQLite database to a CSV file.

### Export SQLite Database to a CSV file using sqlite3 tool

SQLite project provides you with a command-line program called `sqlite3` or `sqlite3.exe` on Windows. By using the `sqlite3` tool, you can use the SQL statements and dot-commands to interact with the SQLite database.

To export data from the SQLite database to a CSV file, you use these steps:

1. Turn on the header of the result set using the `.header` on command.
2. Set the output mode to CSV to instruct the `sqlite3` tool to issue the result in the CSV mode.
3. Send the output to a CSV file.
4. Issue the query to select data from the table to which you want to export.



The following commands select data from the customers table and export it to the data.csv file.

```
>sqlite3 c:/sqlite/chinook.db
sqlite> .headers on
sqlite> .mode csv
sqlite> .output data.csv
sqlite> SELECT customerid,
...>         firstname,
...>         lastname,
...>         company
...> FROM customers;
sqlite> .quit
```

If you check the data.csv file, you will see the following output.

```
CustomerId,FirstName,LastName,Company
1,"Luis","Gonçalves","Embraer - Empresa Brasileira de Aeronáutica S.A."
2,Leonie,"Köhler",
3,"François",Tremblay,
4,"Bjørn",Hansen,
5,"František","Wichterlová","JetBrains s.r.o."
6,Helena,"Holý",
7,Astrid,Gruber,
8,Daan,Peeters,
9,Kara,Nielsen,
10,Eduardo,Martins,"Woodstock Discos"
11,Alexandre,Rocha,"Banco do Brasil S.A."
```

Besides using the dot-commands, you can use the options of the sqlite3 tool to export data from the SQLite database to a CSV file.

For example, the following command exports the data from the tracks table to a CSV file named tracks.csv.

```
>sqlite3 -header -csv c:/sqlite/chinook.db "select * from
tracks;" > tracks.csv
Code language: SQL (Structured Query Language) (sql)
TrackId,Name,AlbumId,MediaTypeId,GenreId,Composer,Milliseconds,Bytes,UnitPrice
1,"For Those About To Rock (We Salute You)",1,1,1,"Angus Young, Malcolm Young, Br
2,"Balls to the Wall",2,2,1,,342562,5510424,0.99
3,"Fast As a Shark",3,2,1,"F. Baltes, S. Kaufman, U. Dirkschneider & W. Hoffman",2
4,"Restless and Wild",3,2,1,"F. Baltes, R.A. Smith-Diesel, S. Kaufman, U. Dirksch
5,"Princess of the Dawn",3,2,1,"Deaffy & R.A. Smith-Diesel",375418,6290521,0.99
6,"Put The Finger On You",1,1,1,"Angus Young, Malcolm Young, Brian Johnson",20566
7,"Let's Get It Up",1,1,1,"Angus Young, Malcolm Young, Brian Johnson",233926,7636
8,"Inject The Venom",1,1,1,"Angus Young, Malcolm Young, Brian Johnson",210834,685
9,Snowballed,1,1,1,"Angus Young, Malcolm Young, Brian Johnson",203102,6599424,0.9
10,"Evil Walks",1,1,1,"Angus Young, Malcolm Young, Brian Johnson",263497,8611245,
11,C.O.D.,1,1,1,"Angus Young, Malcolm Young, Brian Johnson",199836,6566314,0.99
12,"Breaking The Rules",1,1,1,"Angus Young, Malcolm Young, Brian Johnson",263288,
13,"Night Of The Long Knives",1,1,1,"Angus Young, Malcolm Young, Brian Johnson",2
14,Spellbound,1,1,1,"Angus Young, Malcolm Young, Brian Johnson",270863,8817038,0.
15,"Go Down",4,1,1,AC/DC,331180,10847611,0.99
```

If you have a file named `query.sql` that contains the script to query data, you can execute the statements in the file and export data to a CSV file.

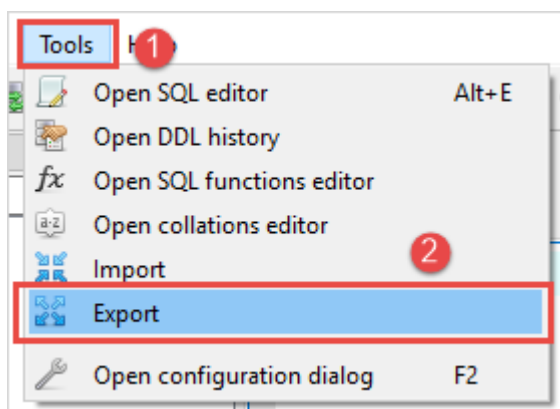
```
>sqlite3 -header -csv c:/sqlite/chinook.db < query.sql > data.csv
```

## Export SQLite database to a CSV file using SQLiteStudio

The SQLiteStudio provides the export function that allows you to export data in a table or the result of a query to a CSV file.

The following steps show you how to export data from a table to a CSV file.

First, click the **Tools > Export** menu item



Next, choose the database and table that you want to export data; check the Export table data.

The screenshot shows the 'Export' dialog box with the title 'Table to export'. It features two dropdown menus: 'Database' (labeled with a red circle 1) set to 'chinook' and 'Table' (labeled with a red circle 2) set to 'customers'. Below these is an 'Options' section with three checkboxes: 'Export table data' (checked, labeled with a red circle 3), 'Export table indexes', and 'Export table triggers'. A note at the bottom of the options section states: 'Note, that exporting table indexes and triggers may be unsupported by some output formats.' At the bottom right are 'Cancel' and 'Next' buttons.

Then, choose a single table to export the data.

The screenshot shows the 'Export' dialog box with the title 'What do you want to export?'. It contains three radio button options: 'A database', 'A single table' (selected and highlighted with a red box, labeled with a red circle 1), and 'Query results'. At the bottom right, the 'Next' button is highlighted with a red box and labeled with a red circle 2. 'Cancel' and 'Next' buttons are also present at the bottom.

After that, (1) choose the CSV as the export format, (2) specify the CSV file name, (3) check the column names in the first row, (4) choose comma (,) as the column separator, (5) treat the NULL value as empty string, (6) click Finish button to complete exporting.


← Export

### Export format and options

Export format

1 CSV

Output

2 ☒ File  

☐ Clipboard

Exported text encoding: windows-1258

Export format options

3 ☒ Column names in first row

Column separator: 4 , (comma)

Export NULL values as: 5 Empty string

6

Cancel Finish

Finally, check the customer.csv file, you will see the following content:

```
CustomerId,FirstName,LastName,Company,Address,City,State,Country,PostalCode,Phone,Fax,Email,SupportRepId
1,Luis,Gonçalves,Embraer - Empresa Brasileira de Aeronáutica S.A., "Av. Brigadeiro Faria Lima, 2170", São Paulo, Brazil, 05508-900, +55 11 3745-1000, +55 11 3745-1001, leonekohler@surfeu.de
2,Leonie,Köhler,,Theodor-Heuss-Straße 34, Stuttgart,,Germany,70174,,+49 0711 2842222,,leonekohler@surfeu.de
3,François,Tremblay,,1498 rue Bélanger, Montréal, QC, Canada, H2G 1A7, +1 (514) 721-4711,,ftremblay@gmail.com,4
4,Bjørn,Hansen,,Ullevålsveien 14, Oslo,,Norway,0171,,+47 22 44 22 22,,bjorn.hansen@yahoo.no,4
5,František,Wichterlová,JetBrains s.r.o.,Klanova 9/506, Prague,,Czech Republic,14700,,+420 2 4172 5555,,+420 2 4172 5555,,frantisek.wichterlov@jetbrains.com,4
6,Helena,Holý,,Rilská 3174/6, Prague,,Czech Republic,14300,,+420 2 4177 0449,,hholy@gmail.com,5
7,Astrid,Gruber,, "Rotenturmstraße 4, 1010 Innere Stadt", Vienne,,Austria,1010,,+43 01 5134505,,astrid.gruber@rotenturmstrasse.at,4
8,Daan,Peeters,,Grétrystraat 63, Brussels,,Belgium,1000,,+32 02 219 03 03,,daan_peeters@apple.be,4
9,Kara,Nielsen,,Sønder Boulevard 51, Copenhagen,,Denmark,1720,,+453 3331 9991,,kara.nielsen@jubii.dk,4
```