Powershell, SQLite

Powershell: Working with a SQLite Database

April 28, 2021 | Michael Albert | 1 Comment

Hi,

SQLite Databases are widly used, even Windows uses for the organisation of its Appx Packages a SQLite database. This database is located in the file C:\ProgramData\Microsoft\Windows\AppRepository\StateRepository-Machine.srd.

SQLite provides a .NET assembly to open such a database file.

Download the latest release which also fits your .NET version. .NET4.5 is usually preinstalled and a good choice for all Windows 10 Versions. Extract the zip file.

```
PS D:\Temp > Invoke-WebRequest -Uri "http://system.data.sqlite.org/blobs/1.0.113.0/sqlit
PS D:\Temp > mkdir D:\Temp\sqlite.net
PS D:\Temp > Expand-Archive D:\temp\sqlite.zip -DestinationPath D:\Temp\sqlite.net
```

Load library

```
PS D:\Temp> [Reflection.Assembly]::LoadFile("D:\Temp\sqlite.net\System.Data.SQLite.dll")

GAC    Version        Location
---    -------        --------
False  v4.0.30319     C:\tmp\sqlite.net\System.Data.SQLite.dll
```

Open a database. For example the Windows Appx state database (make a copy as administrator).

```
copy-item C:\ProgramData\Microsoft\Windows\AppRepository\StateRepository-Machine.srd D:\
$sDatabasePath="D:\Temp\StateRepository-Machine.srd"
$sDatabaseConnectionString=[string]::Format("data source={0}",$sDatabasePath)
$oSQLiteDBConnection = New-Object System.Data.SQLite.SQLiteConnection
$oSQLiteDBConnection.ConnectionString = $sDatabaseConnectionString
$oSQLiteDBConnection.open()
```

Simple SELECT

```
$oSQLiteDBCommand=$oSQLiteDBConnection.CreateCommand()
```

```
$oSQLiteDBCommand.Commandtext="SELECT * from PACKAGE"
$oSQLiteDBCommand.CommandType = [System.Data.CommandTy
$oDBReader=$oSQLiteDBCommand.ExecuteReader()
```


Start reading, columns names can be queried by GetValues(), and all results can be enumerated by a while loop

```
$oDBReader.GetValues()
while($oDBReader.HasRows)
{
    if($oDBReader.Read())
    {
        $oDBReader["PackageFullName"]
    }
}
$oDBReader.Close()
```

Reading a trigger

```
$oSQLiteDBCommand.Commandtext="select * from sqlite_master where type = 'trigger' and n
$oSQLiteDBCommand.CommandType = [System.Data.CommandType]::Text
$oDBReader=$oSQLiteDBCommand.ExecuteReader()
while($oDBReader.HasRows)
{
    if($oDBReader.Read())
    {
        write-host "Trigger: " $oDBReader["name"] "SQL Statement:" $oDBReader["sql"]
    }
}
$oDBReader.Close()
```

Create and open your own database, CreateFile creates a zero length file and is initialized when writing to the database.

```
$sDatabasePath="D:\temp\MyDatabase.sqlite"
[System.Data.SQLite.SQLiteConnection]::CreateFile($sDatabasePath)
$sDatabaseConnectionString=[string]::Format("data source={0}",$sDatabasePath)
$oSQLiteDBConnection = New-Object System.Data.SQLite.SQLiteConnection
$oSQLiteDBConnection.ConnectionString = $sDatabaseConnectionString
$oSQLiteDBConnection.open()
```

Create a Table

```
$oSQLiteDBCommand=$oSQLiteDBConnection.CreateCommand()
$oSQLiteDBCommand.Commandtext="create table FavoriteMetalBands (name varchar(100), score
$oSQLiteDBCommand.CommandType = [System.Data.CommandType]::Text
$oSQLiteDBCommand.ExecuteNonQuery()
```

Insert to the new Dataset

```
$oSQLiteDBCommand.Commandtext="INSERT INTO FavoriteMetalBands (name , score) VALUES (@Ba
$oSQLiteDBCommand.Parameters.AddWithValue("BandName", "Kataklysm");
$oSQLiteDBCommand.Parameters.AddWithValue("MyScore", 10);
$oSQLiteDBCommand.ExecuteNonQuery()
```

At the end, close the connection

```
1 | $oSQLiteDBConnection.Close()
```


Michael

DATABASE EDIT OPEN POWERSHELL SQLITE

ONE THOUGHT ON "POWERSHELL: WORKING WITH A SQLITE DATABASE"

Kees

APRIL 17, 2022 AT 8:00 PM

Right, so I tried to implement this solution to write the output from my solar panels to a SQLite database. However, it's hard to grasp the complexities and then I found that you can install SQLite and just call sqlite3 from powershell.

So, now I'm down to 1 line of code for inserting a new record (only if power output is over 0 Watts obviously):
sqlite3.exe ./enphase.db "insert into production (wnow, whlifetime,time) VALUES ($wnow,$whlifetime,time());"

For the record, I'm not a professional programmer, though I do have some experience with powershell in a enterprise environment.