Technical Note 044

Gestione di un database in SQLite (file db) con System Platform

Rev 1 - 12/04/2023



Introduzione

Questa TN descrive come gestire un database in SQLite con Wonderware System Platform, ovvero costruisce un esempio su come leggere da un file con estensione db.

Versioni

Quanto descritto è stato verificato con la versione 2020 R2 SP1 (20.1.100) installata su Windows Server 2019 e 2023 P01 (23.0.001) installata su Windows Server 2022.

Cosa sono i file .db

I dispositivi mobili, come i telefoni cellulari Android e iOS, memorizzano una serie di informazioni in file di database DB. Ad esempio, un file DB può contenere i contatti, le informazioni sugli SMS o altri dati del dispositivo o dell'applicazione. I file DB sono generalmente salvati in un formato di database SQLite, ma sono anche generalmente bloccati o crittografati, in modo da non poter accedere o modificare i dati.

Un esempio di file db dell'ambiente Wonderware è il file LicenseManagerData.db che contiene informazioni sulle licenze XML importate sul License Manager.

Procedura

Di seguito la procedura per costruire il nostro esempio:

1. Per prima cosa scarichiamo il pacchetto evidenziato in foto dal link https://system.data.sqlite.org/index.html/doc/trunk/www/downloads.wiki

(c. 90 mb)	(sha1: e2c2x3946949c51559432d5650f6b9f1577fe613)
Precompiled Binaries for 32-bit Windows (.NET Framework 3.5 SP1)	
sqlite-netFx35-binary-bundle-Win32-2008-1.0.117.0.zip (2.58 MB)	This binary package features the mixed-mode assembly and contains all the binaries for the x86 version of the System Data. SQLite 1.0. 117.0 (3.40.0) package. The Visual C++ 2008 SP1 numlime for x88 and the .NET Framework 3.5 SP1 are required. (ba1: 05084c4: 1582c926097056075119f9090334136c0)
sqilte-netFx35-binary-Win32-2008-1-0-117-0-zip. (2.57 MiB)	This binary package contains all the binaries for the x88 version of the System Data SQLite 1.0 117.0 (3.40.0) package. The Visual C++ 2008 SP1 runtime for x86 and the .NET Framework 0.5 SP1 are required. (sha1: 615cdd0uect3014402829a72049db5dcectd5984)

- 2. Sblocchiamo il file zip subito dopo il download (tasto destro -> proprietà -> annulla blocco).
- 3. A questo punto importiamo il file System.Data.SQLite.dll nella galaxy entrando nell'IDE e poi cliccando su Galaxy > Import > Script Function Library.
- 4. Copiamo i file System.Data.SQLite.dll e System.Data.SQLite.dll.config nei seguenti percorsi:
 - a. C:\Program Files (x86)\Common Files\ArchestrA

- b. C:\Program Files (x86)\Common Files\ArchestrA\Services
- c. C:\Program Files (x86)\ArchestrA\Framework\Bin
- d. C:\Windows\Microsoft.NET\assembly\GAC_MSIL\System.Data.SQLite\v4.0_1.0.117 .0__db937bc2d44ff139 (da System.Data.SQLite dobbiamo creare le cartelle).

A questo punto possiamo leggere il nostro file db con uno script simile a questo di seguito

dim connection as System.Data.SQLite.SQLiteConnection; dim connString as string; dim query as string; dim command as System.Data.SQLite.SQLiteCommand;

dim reader as System.Data.SQLite.SQLiteDataReader;

try

connString = "Data Source=c:\myDbFile.db;Version=3"; connection = new System.Data.SQLite.SQLiteConnection(connString); connection.Open(); LogWarning("connessione aperta");

'SELECT

try

query = "Select * from myDbTable"; command = new System.Data.SQLite.SQLiteCommand(query,connection); reader = command.ExecuteReader(); LogWarning("Inizio Lettura"); while reader.Read()

LogMessage(reader("myDbField"));

endwhile;

```
reader.Close();
```

LogWarning("Fine Lettura");

catch

LogError(error);

endtry;

connection.Close();

LogWarning("connessione chiusa");

catch

LogError(error);

endtry;

Me.trig = false;

Lo script è inserito in un oggetto UserDefined con un attributo trig che viene valorizzato a false a fine script. Lo script viene eseguito sull'OnTrue dell'espressione me.trig

COLite *		C 2 B 1
SQUICE Scripte Object To Supervise		
Attributes 30 pts Object Information		
Collabor		Configure execution order
Name V St On Ex Of	Aliases:	L'
ReadFromDBFile x	Declarations:	dî di
	Scripts:	Execution type: Execute 🗸 🚽 🗟
	Basics @	
	Expression: me.trig	
	Trigger type: OnTrue	✓ Quality changes 🗳
	Trigger period: 00:00:00.0000000 🖆	Runs asynchronously
	Deadband: 0.0	Timeout limit: 0 ms 🖨
	Historize script state	Report alarm on execution error
		Priority:
	<pre>1 dim connection as System.Data.SQLite.SQLiteConnection;</pre>	
	2 dim construing as string;	
	4 dim command as System.Data.SQLite.SQLiteCommand;	
	s dim reader as System.Data.SQLite.SQLiteDataReader;	
	7 try	
	<pre>@ connString = "Data Source=c:\wyDbFile.db;Version=3"; </pre>	
	<pre>9 connection = new system.lata.sulite.sulite.sulite.connection(connastring); 10 connection.open();</pre>	
	<pre>LogWarning("connessione aperta");</pre>	
	12 SELECT	
	14 try	
	10 query = Search - Lion myohanke, 16 command = new System Data SQLiteCommand(query,connection);	
	<pre>17 reader = command.ExecuteReader(); 1 reader = command.ExecuteReader();</pre>	
	is while reader. Read() 15	
	20 LogMessage(reader("myDbField"));	
	2 reader.Close();	
Inherited scripts:	<pre>23 LogWarning("Fine Lettura");</pre>	
Name St On Ex Of	Sh 24 catch 25 LogError(error);	
	26 endtry;	
	27 28 connection.Close():	
	<pre>29 LogWarning("connessione chiusa");</pre>	
	30 31 catch	
	12 LogError(error);	
	33 endtry;	
	35 Me.trig = false;	
	Line: 11 Col: 38	

Nel nostro caso il risultato della query viene scritto nel log.

Autore: F. Pastore

Disclaimer

Il presente documento è fornito a scopo di esempio e non sostituisce la documentazione AVEVA. L'applicazione di quanto contenuto, in un preciso ambito applicativo, deve essere sempre validata da un tecnico Wonderware. La documentazione rilasciata da AVEVA resta il riferimento tecnico ufficiale da seguire: <u>softwaresupport.aveva.com</u>. Wonderware Italia non si assume la responsabilità di un'applicazione scorretta di questo documento.