Subject: Sqlite3 int64 / wstring support patch
Posted by aroman on Tue, 25 Jul 2006 08:25:28 GMT
View Forum Message <> Reply to Message

Sqlite3 supports directly storing int64 values in the database. It also supports directly storing 16-bit strings. I would like to add support to UPP to handle that.

This involves patching the Value class to support int64's, along with adding the required methods in the sqlite3 value parsing. Also, it should be added to the schema.

Since Value already stores doubles, int64 does not increase size.

I also gently tweaked the storage format for time and date. The original stored time/date as:
'YYYY-MM-DD' or 'YYYY-MM-DD HH:MM:SS'
however the single quotes are not needed since the string is being bound and saved directly into the DB. (I think I messed up the order, though - please check that.)

While I was at it, I added support for "Limit(n)" to limit the number of rows returned by a query.

- Augusto

Subject: Re: Sqlite3 int64 / wstring support patch
Posted by mirek on Tue, 25 Jul 2006 18:52:12 GMT
View Forum Message <> Reply to Message

aroman wrote on Tue, 25 July 2006 04:25Sqlite3 supports directly storing int64 values in the database. It also supports directly storing 16-bit strings. I would like to add support to UPP to handle that.

This involves patching the Value class to support int64's, along with adding the required methods in the sqlite3 value parsing. Also, it should be added to the schema.

Since Value already stores doubles, int64 does not increase size.

I also gently tweaked the storage format for time and date. The original stored time/date as:
'YYYY-MM-DD' or 'YYYY-MM-DD HH:MM:SS'
however the single quotes are not needed since the string is being bound and saved directly into the DB. (I think I messed up the order, though - please check that.)

While I was at it, I added support for "Limit(n)" to limit the number of rows returned by a query.

- Augusto

Are you sure WString support is correct? I would expect sqlite3_bind_text16 to accept 16-bit parameters...

Mirek

Subject: Re: Sqlite3 int64 / wstring support patch
Posted by mirek on Tue, 25 Jul 2006 19:03:32 GMT
View Forum Message <> Reply to Message

Also, now that I had investigated the code more in depth...

The idea of "prepared statements cache" is not quite necessary.

The SqlConnection interface should keep just one prepared statement. It can easily detect which one and how long to keep it (parse flag indicates that statement should be parsed...).

Now if user wants to keep prepared statement, the simplest way is to make its variable 'static', leaving the control to the client.

Mirek

Subject: Re: Sqlite3 int64 / wstring support patch
Posted by mirek on Tue, 25 Jul 2006 19:26:53 GMT
View Forum Message <> Reply to Message

aroman wrote on Tue, 25 July 2006 04:25Sqlite3 supports directly storing int64 values in the database. It also supports directly storing 16-bit strings. I would like to add support to UPP to handle that.

This involves patching the Value class to support int64's, along with adding the required methods in the sqlite3 value parsing. Also, it should be added to the schema.

Since Value already stores doubles, int64 does not increase size.

I also gently tweaked the storage format for time and date. The original stored time/date as:
'YYYY-MM-DD' or 'YYYY-MM-DD HH:MM:SS'
however the single quotes are not needed since the string is being bound and saved directly into the DB. (I think I messed up the order, though - please check that.)

While I was at it, I added support for "Limit(n)" to limit the number of rows returned by a query.

- Augusto

Are you sure WString support is correct? I would expect sqlite3_bind_text16 to accept 16-bit parameters...

UPDATE: There is also missing fetch support for WString, worse, it seems impossible. Perhaps the rigth solution is to store everything using sqlite3_bind_text16 and fetch using sqlite3_column_text16....

I am also not very happy about DATE/TIME...

Well, there is one quite dirty, but effective way. What about to represent them as some high double numbers? Nobody would really miss numbers above 1E+300 and we could use that exponent to detect date/time number (number of seconds or days since set the choosen start of era). All date/time range comparisons would work just fine (with the help of dialect test in SqlFormat).

Mirek
