Subject: Database portability: Sqlite3
Posted by fabio on Fri, 08 Dec 2006 22:02:49 GMT
View Forum Message <> Reply to Message

I like database portability, yet complete portability is never possible.....

In uppsrc/plugin/sqlite3/Sqlite3upp.cpp .....

```
Vector<String> Sqlite3Session::EnumDatabases() {
 // In theory, sqlite3 can "ATTACH" multiple databases (up to 10).
 // However, I don't know how to list them.
 Vector<String> out;
 out.Add(current_dbname);
 return out;
}

Vector<String> Sqlite3Session::EnumTables(String database) {
 // Ignores database
 Vector<String> out;
 Sql sql(*this);
 sql*Select(SqlId("tbl_name")).From(SqlId("sqlite_master")).Where(SqlId("type")=="table");
 while (sql.Fetch())
  out.Add(sql[0]);
 return out;
}
```

The solution (from file uppsrc/plugin/sqlite3/lib/shell.c) is:

for get list of databases:
PRAGMA database_list;

for get the list of all tables and views:
SELECT name FROM sqlite_master WHERE type IN ('table','view') AND name NOT LIKE
'sqlite_%'
ORDER BY 1;

for get list of column:
PRAGMA table_info(table);

and for date emulation problem .....

the functions:
const void *sqlite3_column_decltype16(sqlite3_stmt*,int); //UTF-16
or
const char *sqlite3_column_decltype(sqlite3_stmt *, int i); // UTF-8

return a c-string with type of column or null pointer if computed column.
For declared 'date'-column return the pseudo-type 'date' and not the real type 'text'.

Real type SQLITE_TEXT (3) is returned from function:
int sqlite3_column_type(sqlite3_stmt*, int iCol);
(1,2,3,4,5) or (SQLITE_INTEGER,SQLITE_FLOAT,SQLITE_TEXT,SQLITE_BLOB,SQLITE_NULL)

one solution is test the two value SQLITE_TEXT && 'date' for detect colum of pseudo-type date

consequently ....

My first little (useless?) contribution is:
Sqlite3Session::EnumDatabases(), modified for database list
Sqlite3Session::EnumTables(), modified for database name support
Sqlite3Session::EnumViews(), created
Sqlite3Session::EnumColumns(), created

```
Vector<String> Sqlite3Session::EnumDatabases() {
	Vector<String> out;
 Sql sql(*this);
 sql.Execute("PRAGMA database_list;");
 while (sql.Fetch())
		out.Add(sql[1]);  // sql[1] is database name, sql[2] is filename
 return out;
}

Vector<String> Sqlite3Session::EnumTables(String database) {
 Vector<String> out;
	String dbn=database;
	if(dbn.IsEmpty()) dbn=current_dbname; // for backward compatibility
 Sql sql(*this);
	sql.Execute("SELECT name FROM "+dbn+".sqlite_master WHERE type='table' AND name NOT LIKE 'sqlite_%' ORDER BY 1;");
 while (sql.Fetch())
  out.Add(sql[0]);
 return out;
}

Vector<String> Sqlite3Session::EnumViews(String database) {
 Vector<String> out;
	String dbn=database;
	if(dbn.IsEmpty()) dbn=current_dbname;
 Sql sql(*this);
	sql.Execute("SELECT name FROM "+dbn+".sqlite_master WHERE type='view' AND name NOT LIKE 'sqlite_%' ORDER BY 1;");
 while (sql.Fetch())
```

```
  out.Add(sql[0]);
 return out;
}
```

and a few changes of
void Sqlite3Connection::BindParam(int i, const Value& r)
bool Sqlite3Connection::Execute()
void Sqlite3Connection::GetColumn(int i, Ref f)

for date emulation.....

the result work with EditDate, SqlCtrl, SqlArray.

Remain the problem of aggregate function like max and min, but Sqlite3 permit creation of user function ..... and so .....

In the attached file (zip):
new uppsrc/plugin/sqlite3/sqlite3.h
new uppsrc/plugin/sqlite3/sqlite3upp.cpp
Sqlite3upp_changes.txt

Mirek can you modify the source of distribution from the attached file ? Thanks.

Fabio

File Attachments

1) Sqlite3upp_Upgrade.zip, downloaded 1446 times

Subject: Re: Database portability: Sqlite3
Posted by mirek on Mon, 11 Dec 2006 13:37:48 GMT
View Forum Message <> Reply to Message

I am really sorry, but it seems your attachment was lost during forum migration.

Can you resend it please?

Mirek

Subject: Re: Database portability: Sqlite3
Posted by fabio on Tue, 12 Dec 2006 13:39:53 GMT
View Forum Message <> Reply to Message

ok

Fabio

File Attachments

1) Sqlite3upp_Upgrade.zip, downloaded 349 times

Subject: Re: Database portability: Sqlite3
Posted by mirek on Thu, 14 Dec 2006 19:57:29 GMT
View Forum Message <> Reply to Message

Well, there seems to be problem with this date emulation - no expressions are supported, only Date columns selected.

My suggestion was to encode dates as some very high doubles.

E.g.

```
Date d;
//...
double x = (d - Date(1, 1, 1)) * 1.0e300;
```

Loosing a double range between 1.0e300 - 1.0e307 does not seem a big problem to me. Moreover, at least some expressions work as expected.

Mirek

Subject: Re: Database portability: Sqlite3
Posted by mirek on Sat, 23 Dec 2006 11:38:48 GMT
View Forum Message <> Reply to Message

luzr wrote on Thu, 14 December 2006 14:57Well, there seems to be problem with this date emulation - no expressions are supported, only Date columns selected.

My suggestion was to encode dates as some very high doubles.

After further investigation, I came to conclusion that your solution based on column type is better and the best we can acutally get with Sqlite....

Applied all of yours patches. Welcome to the U++ contributors

Mirek

Subject: Re: Database portability: Sqlite3
Posted by fabio on Sat, 23 Dec 2006 12:14:24 GMT
View Forum Message <> Reply to Message

Thanks Mirek.

Sqlite user function can't get declared type, and is not actually possible to solve problem of expressions in columns.

I try to contact sqlite team for implement 'date' column type.
No response from team ....

For date stored as text 'yyyy-mm-dd', is very simple to implement get from table (and put to table) with littles changes in sqlite text function. The only difference is the returned type: SQLITE_DATE and not SQLITE_TEXT.

User function can execute date-conversion and date-expression, but the core of sqlite must support the 'date' type.

I wait..... and .... retry.

Fabio