

LIBREOFFICE BASE EMOJI TROUBLESHOOTING

STATEMENT OF THE PROBLEM

LibreOffice BASE fails Unicode compliance when BASIC SQL update statements contain multi-byte characters, such as emoji. For example 🤔 will appear as either a question mark, or as two different characters.

My hypothesis is that LibreOffice BASE is using SQLExecDirect when it should be using SQLExecDirectW. See far below as the result of the investigation.

Note well. Direct entry of an emoji into a table in LibreOffice via the Winkey period dialog, or by simply pasting it in, works.

But, trying to enter the data via the BASIC SQL Statement command fails. Example code below.

Similarly, reading a record from a SQLite table that contains an emoji into a recordset, and then writing from the recordset to another table fails.

Using an SQL substring statement to copy a portion of the record from one table to another works.

I therefore there is something in the LibreOffice functions to write to a database that is not Unicode compliant.

I'll supply system info and steps to reproduce below.

STEPS I TOOK TO INVESTIGATE

Changed the character set in LibreOffice to utf-8. No help.

I wrote a test routine.

I corresponded with the author of the ODBC driver, and sent trace results. This conclusively proved that the problem was not in the ODBC driver. Details are below.

Here is the test routine I wrote. (The data is from the project I'm working on for a linguist, and comes from YouTube comments. Linguists love that as a source of raw data.)

```

CREATE TABLE "TestEmoji" (
    "ID1" integer NOT NULL,
    "author"    wvarchar(100),
    "comment"   wvarchar(100),
    PRIMARY KEY("ID1")
)

sub TestEmojiHandling
    dim sql as string, emoji as string, result as string
    emoji = "❤️❤️SWIFTIE❤️❤️"
    result = join(split(emoji)) ' Result still shows the emoji

    ' Write via native createStatement command. This changes the
emoji into question marks, e.g. "??SWIFTIE??:
    REM make sure your connected to the database
    if IsNull(ThisComponent.CurrentController.ActiveConnection) then
        ThisComponent.CurrentController.connect
    endif

    REM execute!
    Dim oStatement As Object
    oStatement =
ThisComponent.CurrentController.ActiveConnection.createStatement()
    sql = " INSERT INTO TestEmoji (author, comment) VALUES('via native
BASIC statement command', '❤️❤️SWIFTIE❤️❤️') "
    result = oStatement.execute(sql)

    ' Write via string. This changes the emoji into question marks,
e.g. "??SWIFTIE??:
    ' Using Access2Base, which probably eventually does a
createStatement command, so we get the same result
    sql = " INSERT INTO TestEmoji (author, comment) VALUES('from a
string', '❤️❤️SWIFTIE❤️❤️') "
    DoCmd.RunSQL(sql)

    'Write via string variable. This changes the emoji into question
marks, e.g. "??SWIFTIE??:
    ' Using Access2Base, which probably eventually does a
createStatement command, so we get the same result
    sql = " INSERT INTO TestEmoji (author, comment) VALUES('from a
variable', " & delim(emoji) & ")"
    DoCmd.RunSQL(sql)

    ' Now via subselect. This WORKS! It keeps the emoji.
    ' In this case, the string containing the emoji is only handled
by the back-end database, in my case SQLite3
    sql = "insert into 'TestEmoji' (author, comment) select 'via
subSelect' AS author, comment from comments where comment_id = 'Ugwwz0Lx-
WeAdw7q_PR4AaABAg'"
    DoCmd.RunSQL(sql)
end sub

```

REM Record for the subselect is:

26	commentThread	UgwvzoLx-WeAdw7q_PR4AaABAg	Andrea Macias	I ♥ TS
----	---------------	----------------------------	---------------	--------

SYSTEM INFO

View basic information about your computer

Windows edition

Windows 10 Home

© 2019 Microsoft Corporation. All rights reserved.

System

Manufacturer:	ASUSTek Computer Inc.
Model:	UX305UAB Signature Edition
Processor:	Intel(R) Core(TM) i7-6500U CPU @ 2.50GHz 2.59 GHz
Installed memory (RAM):	8.00 GB (7.90 GB usable)
System type:	64-bit Operating System, x64-based processor
Pen and Touch:	No Pen or Touch Input is available for this Display

Database, SQLite3

ODBC Driver, Christian Werner's 64-bit SQLite odbc at <http://www.ch-werner.de/sqliteodbc/>

Confirmed the bug on LibreOffice 6.3.4.2 and LibreOffice 6.4

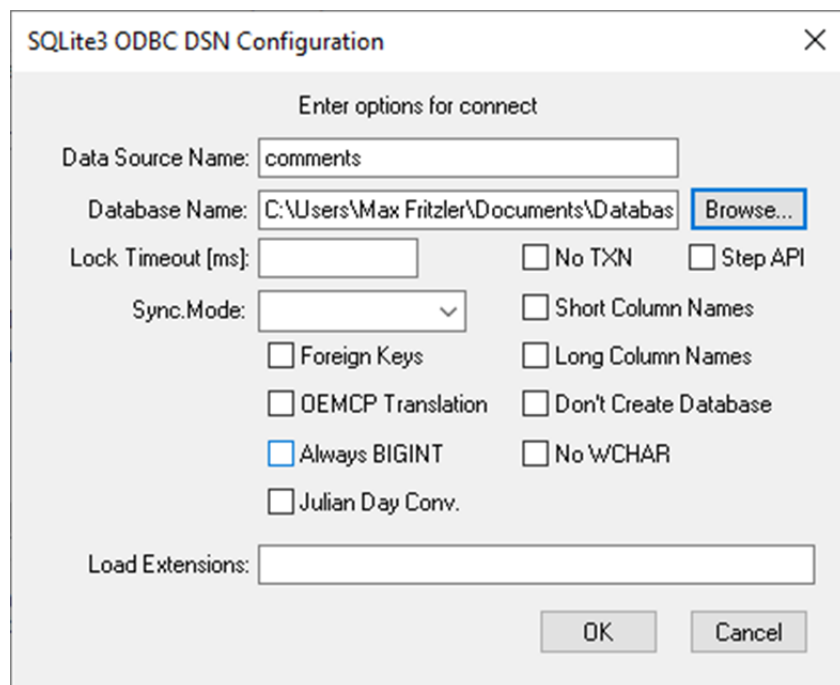
TROUBLESHOOTING THE ODBC DRIVER

Following Christian Werner's suggestions in an email from March 7, 2020.

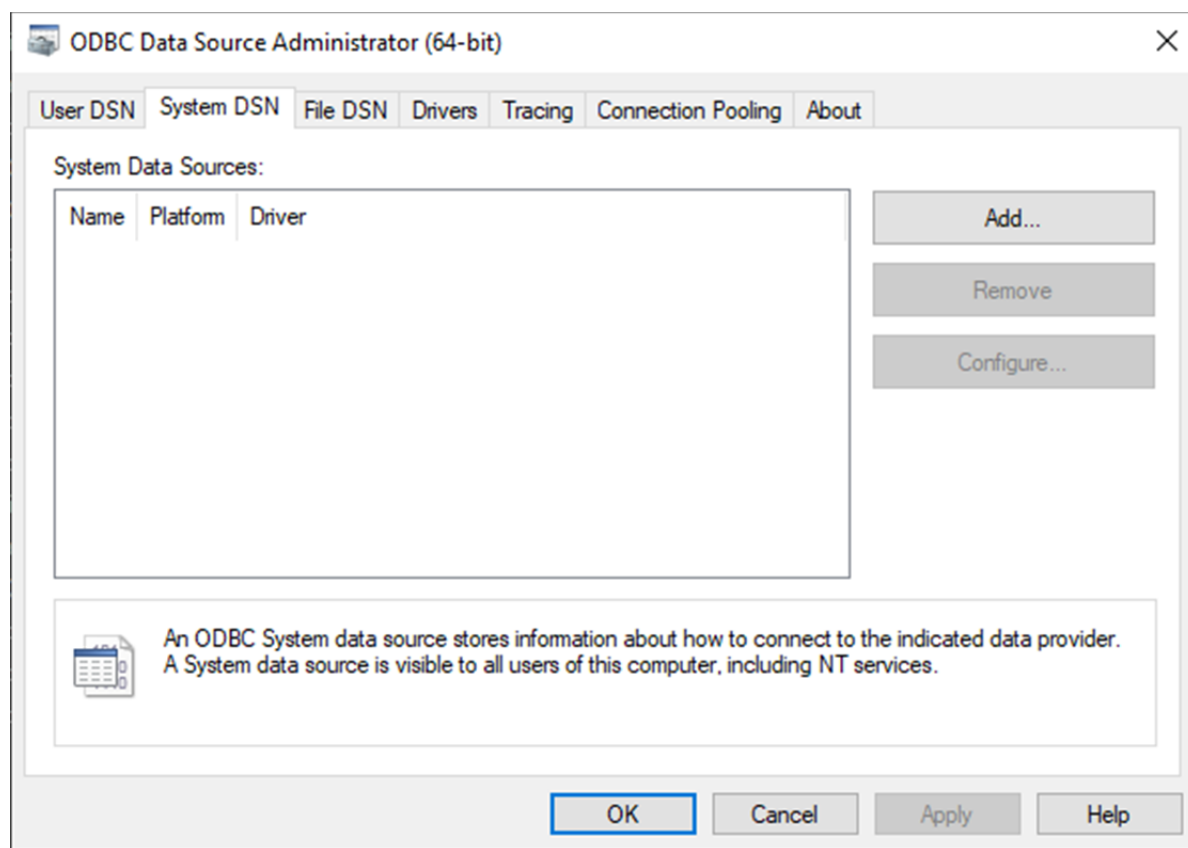
I installed SQLITE3 ODBC driver for Win64 from the downloaded installer.

I did not install SQLite2 drivers as I do not use them.

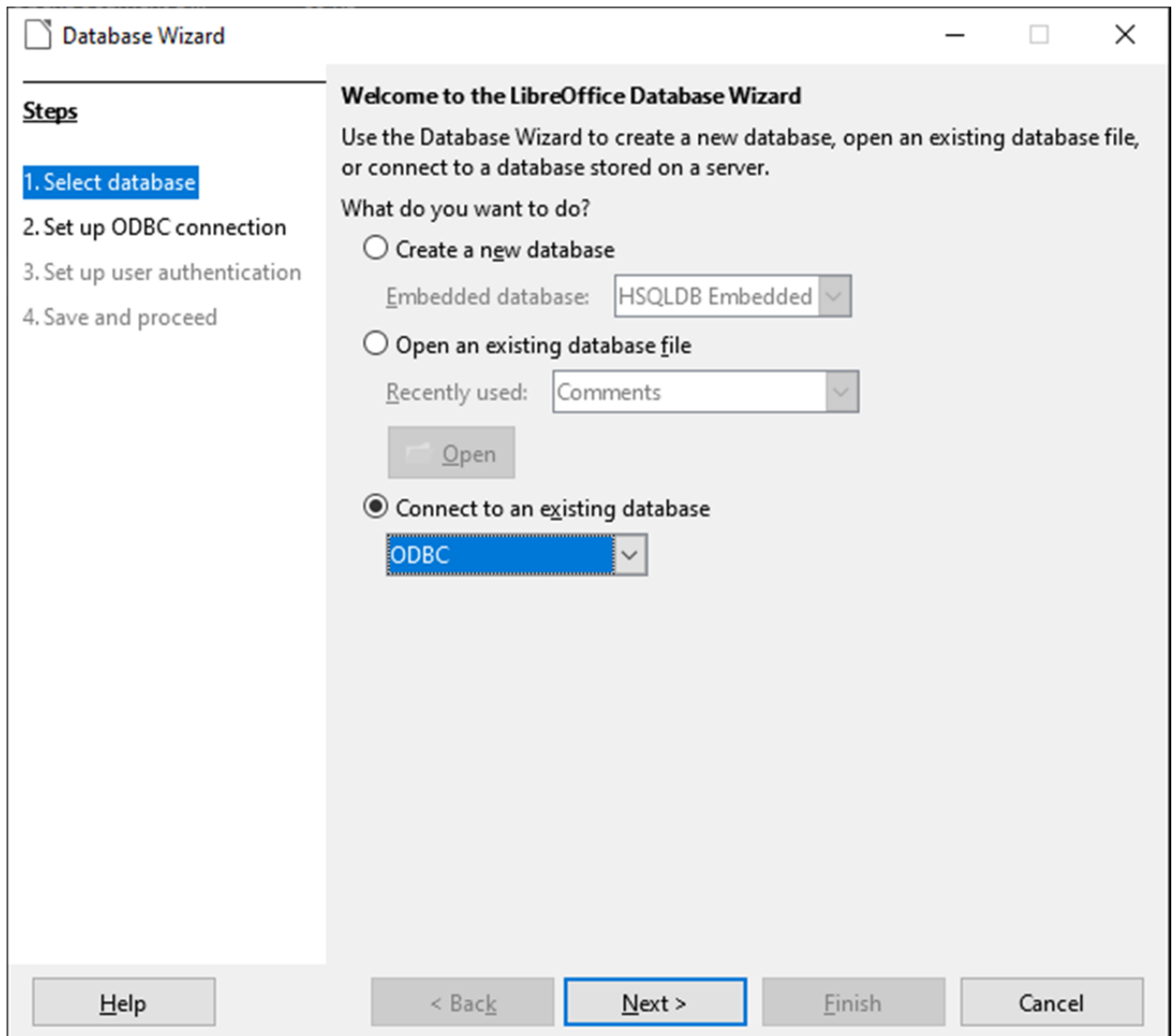
I created a 64 bit User DSN without OEMCPT translation:

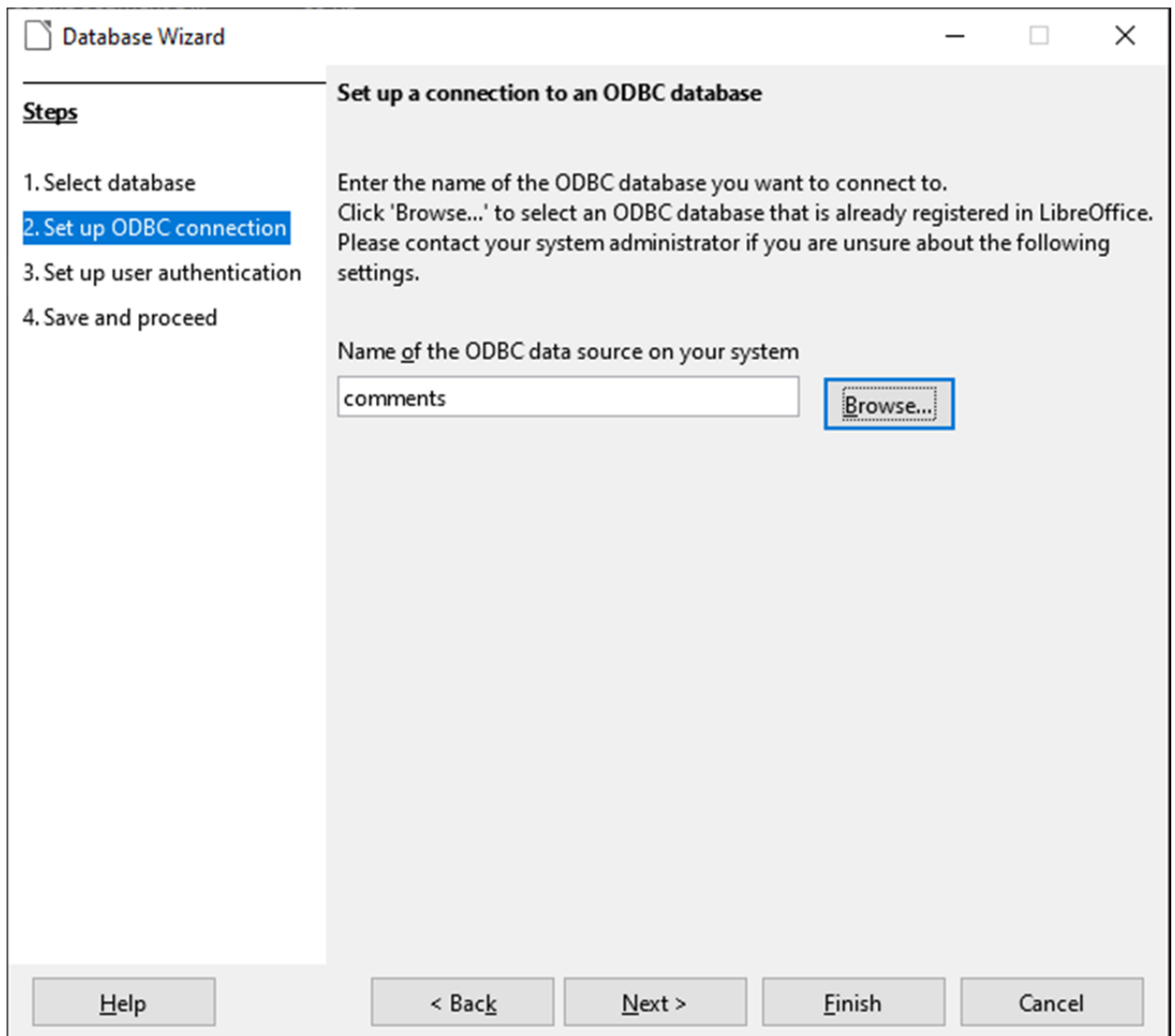


There is no System DSN



I connected to my SQLite database using ODBC






I confirmed I saw the tables in the SQLite backend.

I created a new table TestEmoji with the BASE table creation menu. (Create Table statement supplied with the code listing.)

Comments.odb : Table1 - LibreOffice Base: Table Design

File Edit View Tools Window Help




	Field Name	Field Type	
	id	Integer [integer]	
	col1	Text [wvarchar]	test column 1
	col2	Text [wvarchar]	test column 2

It demanded I create a primary key to enter data, so I allowed it to do so and wound up with this structure:

Comments.odb : TestEmoji - LibreOffice Base: Table Design

File Edit View Tools Window Help



	Field Name	Field Type	
	ID1	Integer [integer]	
	id	Integer [integer]	
	col1	Text [wvarchar]	test column 1
	col2	Text [wvarchar]	test column 2

I was APPARENTLY able to successfully paste in text containing emoji:

ID1	id	col1	col2
0	1	some text	some more text
0	2	❤️SWIFTIE❤️	❤️SWIFTIE❤️

But when I examined the table with DB Browser I saw

Table: TestEmoji

	ID1	id	col1	col2
	Filter	Filter	Filter	Filter
1	1	1	some text	some more text
2	2	2	??SWIFTIE??	??SWIFTIE??

When I did Data / Refresh in LibreOffice I saw the same thing:

	ID1	id	col1	col2
▶	1	1	some text	some more text
	2	2	??SWIFTIE??	??SWIFTIE??
+				

I tried to directly enter emoji using the keystroke combination Winkey+”.” This pops up an emoji grid from which you select. That appeared to work, but when I did Data / Refresh it failed, and appeared as above.

I changed the system encoding by:

Edit / Database / Connection Type:

Data Source Properties: ODBC

Select the type of database to which you want to establish a connection.

Database type:

On the following pages, you can make detailed settings for the connection.

The new settings you make will overwrite your existing settings.

Next

Database properties - Connection settings

General

Name of the ODBC data source on your system

User Authentication

User name:

☐ Password required

Next

It was:

Data Conversion

Character set:

Optional Settings

ODBC options:

☐ Use catalog for file-based databases

Database properties - Additional Settings

Data Conversion

Character set: **Unicode (UTF-8)**

Optional Settings

ODBC options:

☐ Use catalog for file-based databases

Now it ALL WORKED! (I show two different runs with two different tables, but same results in each case.)

ID1	id	col1	col2
1	1	some text	some more text
2	2	??SWIFTIE??	??SWIFTIE??
3	2	-??	
4	4	-👉😄😄😄❤️👤💡	-👉😄😄😄👤👉❤️✅
5	5	-✅👉👤😄👤	test -👉👉👉👉 test
6	6	Pasted 🍷SWIFTIE🍷	Pasted 🍷 edited in db browser

	line	author	comment
		Filter	Filter
1	1	via native BASIC statement command	ââSWIFTIEââ
2	2	from a string	ââSWIFTIEââ
3	3	from a variable	ââSWIFTIEââ
4	4	via subSelect	I â TS

line	author	comment
1	via native BASIC statement command	👉👉👉SWIFTIE👉👉👉
2	from a string	👉👉👉SWIFTIE👉👉👉
3	from a variable	👉👉👉SWIFTIE👉👉👉
4	via subSelect	I ♥ TS

ID1	author	comment
1	some text	some more text
2	??SWIFTIE??	??SWIFTIE??
3	-??	NULL
4	-👉😄😊😍💞👩♀️👩♂️👨♀️👨♂️🌿😏❤️	-👰😄😐👩♂️❤️✅
5	-✅👨♂️😊🙄	test -👰👰👰 test
6	Pasted 💕💕SWIFTIE💕💕	Pasted 💕💕 edited in db browser
7	via native BASIC statement command	ââ=SWIFTIEââ
8	from a string	ââ=SWIFTIEââ
9	from a variable	ââ=SWIFTIEââ
10	via subSelect	I ❤️ TS

Clearly, there is something in the LibreOffice functions to write to a database that is not Unicode compliant.

TRACELOGS

I turned on tracing in the odbc administrator, and ran just the first part of this. That generated a 320K plaintext log, which is attached. I've never looked at a SQL log before, but I searched for "via native BASIC statement command" which is one of the field values, and found it. That little chunk looks like this:

```

program"      2c70-3acc ENTER SQLExecDirect
               HSTMT          0x00000227F2DAD1B0
               UCHAR *        0x00000227F37D42F8 [ 98] " INSERT INTO exports (author,
comment) VALUES('via native BASIC statement command', '??SWIFTIE??')
               SDWORD          98

```

```
program"      2c70-3acc EXIT SQLExecDirect with return code 0 (SQL_SUCCESS)
              HSTMT      0x000000227F2DAD1B0
```

```

        UCHAR *      0x00000227F37D42F8 [ 98] " INSERT INTO exports (author,
comment) VALUES('via native BASIC statement command', '??SWIFTIE??')"
```

```

        SDWORD      98
```

I corresponded with Mr. Werner, the author of the ODBC driver. He was the one who discovered that direct entry of the emoji into the LibreOffice table grid worked. I sent the entire SQL trace log to him. He noted:

Regarding your Basic experiment: it is very fishy that SQLExecDirect is used (the driver has an SQLExecDirectW, too) so which encoding do we see in the ODBC log then? And why use SQLExecDirect at all? If the Basic runtime is modern enough it should provide you the wide char equivalents to SQLPrepare and friends which should be Unicode aware OOTB.

So, I turned on tracing and did a little direct entry of emojis into the table grid. The crucial portion seems to be:

```

program"      3188-1a14      ENTER SQLPrepare
        HSTMT      0x0000023ABE2D2AB0
        UCHAR *      0x0000023AD91B8F28 [ 67] "INSERT INTO
"TestEmoji" ( "author", "comment", "ID1") VALUES ( ?, ?, ?) "
        SDWORD      67
```

Note that no string containing the emoji is present, it is passed in as some sort of parameter.

I'll attach the two trace logs in my bug report.

Please advise.