

---

Subject: MSSQL error management

Posted by [Giorgio](#) on Fri, 16 Feb 2018 09:36:17 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

Hi there,  
to execute SQL queries I use code like this:

```
bool ImportData::InsertBody(myS_Table record)
{
    Sql sql(mssql);
    sql.ClearError();

    try { sql * Insert(My_Table)
        (My_Id, record.Id)
        (My_Row, record.Row)
        (My_Description, record.Description);
        return true;
    } catch(SqlExc) {
        ErrorOK(t_("Failed adding data to the database due to the following error: ") +
SQL.GetLastError());
        return false;
    }
    return true;
}
```

In this way, (1) if there are errors in executing the query the user is notified of the specific reason and (2) I can control the program flow using the true/false value returned by the method.

This is working as expected for MySql, PostgreSql and SQLite, but here it comes Microsoft and its @#!&\$ MS Sql Server.

When there is an error (e.g. a duplicate key) the user is NOT notified and the query returns always true. The only way to catch errors is looking at the log file activated using `.LogErrors()` and `SetTrace()`. This is not really user friendly. In case of an error what I got in the log file is the following:

```
ERROR [Microsoft][SQL Server Native Client 11.0][SQL Server]Violation of PRIMARY KEY
constraint 'Id'. Cannot insert duplicate key in object 'dbo.X_DORIG'. The duplicate key value is
(180107, 1, 'Art. number'          ).(0): insert into X_DORIG(Id, Row, Description,)
values (180107, 1, 'Art. number')
```

I tried using both `Driver={SQL Server Native Client 11.0};` and `Driver={SQL Server};` with the same result.

Is there a workaround for this issue?

Thanks,  
Gio

---

Subject: Re: MSSQL error management  
Posted by [mirek](#) on Sat, 24 Feb 2018 18:21:13 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Giorgio wrote on Fri, 16 February 2018 10:36Hi there,  
to execute SQL queries I use code like this:

```
bool ImportData::InsertBody(myS_Table record)
{
    Sql sql(mssql);
    sql.ClearError();

    try { sql * Insert(My_Table)
        (My_Id, record.Id)
        (My_Row, record.Row)
        (My_Description, record.Description);
        return true;
    } catch(SqlExc) {
        ErrorOK(t_("Failed adding data to the database due to the following error: ") +
SQL.GetLastError());
        return false;
    }
    return true;
}
```

In this way, (1) if there are errors in executing the query the user is notified of the specific reason and (2) I can control the program flow using the true/false value returned by the method.

This is working as expected for MySql, PostgreSql and SQLite, but here it comes Microsoft and its @#!&\$ MS Sql Server.

When there is an error (e.g. a duplicate key) the user is NOT notified and the query returns always true. The only way to catch errors is looking at the log file activated using .LogErrors() and SetTrace(). This is not really user friendly. In case of an error what I got in the log file is the following:

```
ERROR [Microsoft][SQL Server Native Client 11.0][SQL Server]Violation of PRIMARY KEY
constraint 'Id'. Cannot insert duplicate key in object 'dbo.X_DORIG'. The duplicate key value is
(180107, 1, 'Art. number'                               ).(0): insert into X_DORIG(Id, Row, Description,)
values (180107, 1, 'Art. number')
```

I tried using both Driver={SQL Server Native Client 11.0}; and Driver={SQL Server}; with the same

result.

Is there a workaround for this issue?

Thanks,  
Gio

You do have exception active (SqlConnection::ThrowOnError), right?

Mirek

---

Subject: Re: MSSQL error management  
Posted by [mirek](#) on Sat, 24 Feb 2018 18:39:46 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

I have just tested with this code, seems ok:

```
#include "app.h"

#include <Sql/sch_schema.h>
#include <Sql/sch_source.h>

using namespace Upp;

CONSOLE_APP_MAIN
{
    MSSQLSession mssql;
    for(;;) {
        String cs = "Driver={SQL Server Native Client 11.0}";
        cs << "Server=MAIN\\SQLEXPRESS;";
        cs << "Trusted_Connection=Yes;";
        if(!mssql.Connect(cs))
            Cout() << "Connect failed: " << mssql.GetLastError() << '\n';
        else
            break;
    }
    SQL = mssql;

#ifdef _DEBUG
    mssql.SetTrace();
#endif
}
```

```
SqlSchema sch(MSSQL);
StdStatementExecutor se(SQL.GetSession());
All_Tables(sch);
ODBCPerformScript(sch.Upgrade(), se);
ODBCPerformScript(sch.Attributes(), se);

mssql.ThrowOnError();

try {
    for(int i = 0; i < 10; i++)
        SQL * Insert(TEST)(ID, i)(TEXT, String('A' + i, 1));
}
catch(SqlExc) {
    DLOG("ERROR");
}
S_TEST tst;
Sql sql;
sql * Select(tst).From(TEST);
while(sql.Fetch(tst))
    DDUMP(tst.ID);
}
```

---

Subject: Re: MSSQL error management  
Posted by [Giorgio](#) on Mon, 26 Feb 2018 14:01:32 GMT  
[View Forum Message](#) <> [Reply to Message](#)

Uhm, I added explicitly `SqlSession::ThrowOnError`, but still I do not get an error, just a message in the log file; maybe the problem happen just with a GUI?

---

Subject: Re: MSSQL error management  
Posted by [mirek](#) on Mon, 26 Feb 2018 15:22:39 GMT  
[View Forum Message](#) <> [Reply to Message](#)

Giorgio wrote on Mon, 26 February 2018 15:01Uhm, I added explicitly `SqlSession::ThrowOnError`, but still I do not get an error, just a message in the log file; maybe the problem happen just with a GUI?

This is definitely NOT related to GUI (99%).

How does your connection code look like?

---

---

Subject: Re: MSSQL error management  
Posted by [Giorgio](#) on Fri, 09 Mar 2018 17:40:05 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

mirek wrote on Mon, 26 February 2018 16:22  
How does your connection code look like?

This is the method I use to connect (I tried both Driver={SQL Server Native Client 11.0} and {SQL Server} with the same result):

```
bool ImportData::OpenMsSql()
{
    String username = "sa";
    String pwd = "secret";
    String dbname = "ADB_TEST";
    String server = "servername\\sqlserver";
    String cs = "Driver={SQL Server};Server=" + server +
        ";UID=" + username + ";PWD=" + pwd + ";Database=" + dbname + ";";

    if(!mssql.Connect(cs))
    {
        Exclamation(t_("Unable to connect to database due to the following error: ") +
mssql.GetLastError());
        return false;
    }

    mssql.SqlSession::ThrowOnError();
    mssql.LogErrors();
    mssql.SetTrace();

    return true;
}
```

---

Subject: Re: MSSQL error management  
Posted by [mirek](#) on Wed, 14 Mar 2018 09:39:10 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

How is mssql defined? (I wonder why there is SqlSession:: before the ThrowOnError).

If it is MSSQLSession, please try these DUMPs:

```
bool Sql::Execute() {
    SqlSession &session = GetSession();
```

```

session.SetStatement(cn->statement);
session.SetStatus(SqlSession::BEFORE_EXECUTING);
cn->starttime = GetTickCount();
Stream *s = session.GetTrace();
if(s) {
#ifdef NOAPPSQL
    if(this == &AppCursor())
        *s << "SQL * ";
    else
        if(this == &AppCursorR())
            *s << "SQLR * ";
#endif
    String st = cn->statement;
    if(session.IsTraceCompression())
        st = CompressLog(st);
    int i = 0;
    for(const char *q = st; *q; q++)
        if(*q == '?' && i < param.GetCount()) {
            Value v = param[i++];
            if(IsString(v))
                *s << "\" << v << "\";
            else
                *s << v;
        }
        else
            s->Put(*q);
        *s << '\n';
    }
    if(!session.IsOpen())
    {
        session.SetStatus(SqlSession::CONNECTION_ERROR);
        return false;
    }
    session.SetStatus(SqlSession::START_EXECUTING);
    bool b = cn->Execute();
    session.SetTime(GetTickCount() - cn->starttime);
    session.SetStatus(SqlSession::END_EXECUTING);
    if(!b)
        session.SetStatus(SqlSession::EXECUTING_ERROR);
    for(int i = 0; i < cn->info.GetCount(); i++)
        cn->info[i].name = ToUpper(cn->info[i].name);

    session.SetStatus(SqlSession::AFTER_EXECUTING);
    DDUMP(b);
    DDUMP(session.throwonerror);
    if(!b && session.throwonerror)
        throw SqlExc(GetSession());

```

```
return b;  
}
```

---

Subject: Re: MSSQL error management  
Posted by [Giorgio](#) on Mon, 19 Mar 2018 11:40:23 GMT  
[View Forum Message](#) <> [Reply to Message](#)

Hi Mirek,  
mssql is defined as MSSQLSession: this application basically retrieves data from a sqlite db and put them into a MS SQL db, so I have two different sql sessions and I do not use the global SQL session.

The SqlSession:: before the ThrowOnError was a typo and I removed it.

I tried to put the DDUMP but I get a really weird error from the compiler: "error C2018: unknown character '0x40'". This error pops up on the lines "DDUMP(b);" and "DDUMP(session.throwonerror);", if I remove those lines the application compiles. I opened the .cpp file with an hex editor and did not find any 0x40 char. Also I updated the compiler to VS2017 (previously I used 2015), bu the error still appears.

Regards,  
gio

---

Subject: Re: MSSQL error management  
Posted by [mirek](#) on Mon, 19 Mar 2018 11:58:49 GMT  
[View Forum Message](#) <> [Reply to Message](#)

Giorgio wrote on Mon, 19 March 2018 12:40Hi Mirek,  
mssql is defined as MSSQLSession: this application basically retrieves data from a sqlite db and put them into a MS SQL db, so I have two different sql sessions and I do not use the global SQL session.

The SqlSession:: before the ThrowOnError was a typo and I removed it.

I tried to put the DDUMP but I get a really weird error from the compiler: "error C2018: unknown character '0x40'". This error pops up on the lines "DDUMP(b);" and "DDUMP(session.throwonerror);", if I remove those lines the application compiles. I opened the .cpp file with an hex editor and did not find any 0x40 char. Also I updated the compiler to VS2017 (previously I used 2015), bu the error still appears.

Regards,  
gio

This is because you are compiling in release mode. "DDUMP" is only supposed to be temporary debugging thing, in debug mode only.

Either compile as debug, or use RDUMP instead.

(Overview:

DDUMP, DLOG - logs in debug, prevents compilation in release mode - that is to force you to remove temporary dumps before the release  
LOG, DUMP - logs in debug, NOP in release  
RLOG, RDUMP - logs in both debug and release  
)

Mirek

---

Subject: Re: MSSQL error management  
Posted by [Giorgio](#) on Mon, 19 Mar 2018 13:50:19 GMT  
[View Forum Message](#) <> [Reply to Message](#)

Hi Mirek,  
this is the log:

```
b = false  
session.throwonerror = true
```

Regards,  
gio

---

Subject: Re: MSSQL error management  
Posted by [mirek](#) on Mon, 19 Mar 2018 15:35:12 GMT  
[View Forum Message](#) <> [Reply to Message](#)

Giorgio wrote on Mon, 19 March 2018 14:50Hi Mirek,  
this is the log:

```
b = false  
session.throwonerror = true
```

Regards,  
gio

Well, if you check the next line, 'throw' should have happened...

I would say something else must be going on here. Please try to add more logs to your code, e.g.:

```
SqlSchema sch(MSSQL);
DLOG("A");
StdStatementExecutor se(SQL.GetSession());
DLOG("B"); All_Tables(sch);
ODBCPerformScript(sch.Upgrade(), se);
DLOG("C");
ODBCPerformScript(sch.Attributes(), se);
DLOG("D");
mssql.ThrowOnError();
DLOG("E");
try {
DLOG("F");
for(int i = 0; i < 10; i++)
SQL * Insert(TEST)(ID, i)(TEXT, String('A' + i, 1));
DLOG("G"); }
catch(SqlExc) {
DLOG("ERROR");
}
DLOG("H");
S_TEST tst;
Sql sql;
sql * Select(tst).From(TEST);
while(sql.Fetch(tst))
DDUMP(tst.ID);
```

and send the whole log....

Mirek

---

---

Subject: Re: MSSQL error management  
Posted by [Giorgio](#) on Tue, 20 Mar 2018 10:15:36 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Hi mirek,  
this is the log:

```
b = true
session.throwonerror = false
A
```

B  
create table TEST ( ID integer primary key identity )  
ERROR [Microsoft][ODBC SQL Server Driver][SQL Server]There is already an object named 'TEST' in the database.(0): create table TEST ( ID integer primary key identity )  
b = false  
session.throwonerror = false  
alter table TEST add TEXT varchar(200)  
ERROR [Microsoft][ODBC SQL Server Driver][SQL Server]Column names in each table must be unique. Column name 'TEXT' in table 'TEST' is specified more than once.(0): alter table TEST add TEXT varchar(200)  
b = false  
session.throwonerror = false  
alter table TEST alter column TEXT varchar(200)  
b = true  
session.throwonerror = false  
C  
alter table TEST add constraint PK\_TEST\$ID primary key (ID) create index PKX\_TEST\$ID on TEST(ID)  
ERROR [Microsoft][ODBC SQL Server Driver][SQL Server]Table 'TEST' already has a primary key defined on it.  
[Microsoft][ODBC SQL Server Driver][SQL Server]Could not create constraint or index. See previous errors.(0): alter table TEST add constraint PK\_TEST\$ID primary key (ID) create index PKX\_TEST\$ID on TEST(ID)  
b = false  
session.throwonerror = false  
create index IDX\_TEST\$TEXT on TEST(TEXT)  
ERROR [Microsoft][ODBC SQL Server Driver][SQL Server]The operation failed because an index or statistics with name 'IDX\_TEST\$TEXT' already exists on table 'TEST'.(0): create index IDX\_TEST\$TEXT on TEST(TEXT)  
b = false  
session.throwonerror = false  
D  
E  
F  
SQL\* insert into TEST(ID, TEXT) values (0, 'A')  
ERROR [Microsoft][ODBC SQL Server Driver][SQL Server]Cannot insert explicit value for identity column in table 'TEST' when IDENTITY\_INSERT is set to OFF.(0): insert into TEST(ID, TEXT) values (0, 'A')  
b = false  
session.throwonerror = true  
ERROR  
H  
select ID, TEXT from TEST  
b = true  
session.throwonerror = true  
tst.ID = 1  
tst.ID = 2  
tst.ID = 3

```
tst.ID = 4
tst.ID = 5
tst.ID = 6
tst.ID = 7
tst.ID = 8
tst.ID = 9
tst.ID = 10
```

I did this using the SQL\_MSSQL application that comes as reference with upp. As far as I understand, there is no trace of the problem. The only difference I notice, is that in my application I do not have the following lines:

```
SqlSchema sch(MSSQL);
StdStatementExecutor se(SQL.GetSession());
All_Tables(sch);
ODBCPerformScript(sch.Upgrade(), se);
ODBCPerformScript(sch.Attributes(), se);
```

It may be the origin of the problem?  
Regards,  
gio

---

Subject: Re: MSSQL error management  
Posted by [mirek](#) on Tue, 20 Mar 2018 15:42:22 GMT  
[View Forum Message](#) <> [Reply to Message](#)

I am really sorry, I have missed that this is the reference code posted by me... I thought that it was code that is causing you problems.

Please try these:

```
bool ImportData::InsertBody(myS_Table record)
{
DLOG("InsertBody");
Sql sql(mssql);
sql.ClearError();
DLOG("A");
try {
DLOG("TRY1");
sql * Insert(My_Table)
(My_Id, record.Id)
(My_Row, record.Row)
(My_Description, record.Description);
```

```
DLOG("TRY2");
    return true;
} catch(SqlExc) {
DLOG("CATCH1");
    ErrorOK(t_("Failed adding data to the database due to the following error: ") +
SQL.GetLastError());
DLOG("CATCH2");
    return false;
}
DLOG("NEVER HERE");
return true;
}
```

---

---

Subject: Re: MSSQL error management  
Posted by [Giorgio](#) on Wed, 21 Mar 2018 12:14:12 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Hi Mirek,  
this is the log:

InsertBody

```
A
TRY1
insert into X_DORIG(Id, Row, Description) values (180107, 1, 'Art. number')
ERROR [Microsoft][ODBC SQL Server Driver][SQL Server]Violation of PRIMARY KEY constraint
'PK_X_DORIG'. Cannot insert duplicate key in object 'dbo.X_DORIG'. The duplicate key value is
(180107, 1).(0): insert into X_DORIG(Id, Row, Description) values (180107, 1, 'Art. number')
b = false
session.throwonerror = true
CATCH1
```

Regards,  
gio

---

---

Subject: Re: MSSQL error management  
Posted by [mirek](#) on Wed, 21 Mar 2018 13:41:49 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

CATCH1 is there, so obviously it works as expected.

The problem is likely in ErrorOK. Either it crashes or something. Does the program crash?

At this point, I suggest placing a breakpoint ErrorOK and step through it.

Is this running in the main thread?

Is any GUI working?

---

Subject: Re: MSSQL error management  
Posted by [mirek](#) on Wed, 21 Mar 2018 14:25:32 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Perhaps you can try this:

```
bool ImportData::InsertBody(myS_Table record)
{
    PromptOK("InsertBody");

    DLOG("InsertBody");
    Sql sql(mssql);
    sql.ClearError();
    DLOG("A");
    try {
        PromptOK("Try");
        DLOG("TRY1");
        sql * Insert(My_Table)
            (My_Id, record.Id)
            (My_Row, record.Row)
            (My_Description, record.Description);
        DLOG("TRY2");
        return true;
    } catch(SqlExc) {
        PromptOK("In catch");
        DLOG("CATCH1");
        ErrorOK(t_("Failed adding data to the database due to the following error: ") +
            SQL.GetLastError());
        DLOG("CATCH2");
        return false;
    }
    DLOG("NEVER HERE");
    return true;
}
```

Just to make sure prompts work as expected.

(Obviously report which ones of these display something...)

Mirek

---

---

Subject: Re: MSSQL error management  
Posted by [Giorgio](#) on Wed, 21 Mar 2018 14:58:19 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Neither the PropmptOK nor the ErrorOK are displayed.

---

---

Subject: Re: MSSQL error management  
Posted by [mirek](#) on Wed, 21 Mar 2018 15:14:01 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Giorgio wrote on Wed, 21 March 2018 15:58Neither the PropmptOK nor the ErrorOK are displayed.

Well, the there is a problem with GUI in general... Totally unrelated with SQL.

Start in GUI\_APP\_MAIN I guess. Place PromptOK there. If it does not display, you have GUI to fix.

Is multithreading involved? Like InsertBody running in non-main thread?

---

---

Subject: Re: MSSQL error management  
Posted by [Giorgio](#) on Wed, 21 Mar 2018 15:30:06 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Hi Mirek,  
the application has no GUI\_APP\_MAIN.

The application is not multithreaded and runs as follows:

1. It opens a text file, parses it and puts parsed data in a sqlite db;
2. Asks the users (using PromptYesNo that displays correctly) if they want to put the data in the MSSQL db;
3. If the user says yes, then the routine that insert data (the one I posted) is executed.

All the steps above are in the constructor.

If data are correct, the application runs smoothly.

Regards,  
gio

---

---

Subject: Re: MSSQL error management  
Posted by [mirek](#) on Wed, 21 Mar 2018 17:46:57 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Giorgio wrote on Wed, 21 March 2018 16:30Hi Mirek,  
the application has no GUI\_APP\_MAIN.

The application is not multithreaded and runs as follows:

1. It opens a text file, parses it and puts parsed data in a sqlite db;
2. Asks the users (using PromptYesNo that displays correctly) if they want to put the data in the MSSQL db;
3. If the user says yes, then the routine that insert data (the one I posted) is executed.

All the steps above are in the constructor.

Without GUI\_APP\_MAIN, GUI is broken.

GUI does not work before GUI\_APP\_MAIN.

I guess that is the problem...

Mirek

---

---

Subject: Re: MSSQL error management  
Posted by [Giorgio](#) on Thu, 22 Mar 2018 08:06:22 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Hi Mirek,  
I added the GUI\_APP\_MAIN and moved there the code that previously was in the constructor and now everything works.  
Thanks,  
gio

P.S. Could you have a look at this? I am sorry to push you, but it is really important to me.

---