

Re: Stored Procedure error is not caught

Source:

<http://www.tech-archive.net/Archive/SQL-Server/microsoft.public.sqlserver.jdbcdriver/2005-08/msg00018.html>

- *From:* Joe Weinstein <joeNOSPAM@xxxxxxx>
 - *Date:* Thu, 25 Aug 2005 17:30:46 -0700
-

Rizwan wrote:

"Joe Weinstein" <joeNOSPAM@xxxxxxx> wrote in message
<news:430DF786.3010002@xxxxxxxxxxx>

Rizwan wrote:

I am using MS SQL Server JDBC Driver. I call a stored procedure from my java code. The Stored Procedure does some inserts. One of the insert failed but in my java code the SQLException is not thrown. Can anybody tell me how to fix this bug?

Thanks

Hi. The problem is that the SQLException (or really the error message from the DBMS, that will be turned into a SQLException) is still out on the network for the statement, until it is read by the Statement. This isn't happening immediately because your procedure is doing multiple inserts, and some of the first ones succeed, so the first things on the line are successful update counts.

You need to add processing after the execute() to get all the returns, and then you will get your exception. Here is an example which contains the ideal code for processing all the inline returns from any stored procedure. With this, you do get your expected exception.

Joe Weinstein at BEA Systems

Re: Stored Procedure error is not caught

```
c = d.connect("jdbc:bea:sqlserver://joe", props);

System.out.println("Driver version is " +
c.getMetaData().getDriverVersion() );

DatabaseMetaData dd = c.getMetaData();
System.out.println("Driver version is " +
dd.getDriverVersion() );
System.out.println("Database Major version is " +
dd.getDatabaseMajorVersion() );
System.out.println("Database Minor version is " +
dd.getDatabaseMinorVersion() );

Statement s = c.createStatement();

try { s.executeUpdate("drop procedure joeproc"); }
catch (Exception ignore){}
try { s.executeUpdate("drop table joetable"); }
catch (Exception ignore){}

s.executeUpdate("create table joetable (bar
varchar(30) not null)");

s.executeUpdate("create procedure joeproc as "
+ " begin "
+ " insert into joetable values('1') "
+ " insert into joetable values(NULL) " // second
insert will fail
+ " insert into joetable values('2') "
+ " end ");

PreparedStatement ps = c.prepareStatement("{ call joeproc() }");
boolean getResultSet = ps.execute();
int updateCount = -1;

while (true) { // handle all in-line results from
any procedure
if (getResultSet) {
ResultSet r = ps.getResultSet();
while (r.next()) {
// process result set
}
r.close();
} else {
updateCount = ps.getUpdateCount();
```

Re: Stored Procedure error is not caught

```
if (updateCount != -1) {
  // process update count
}
}
if ((!getResultSet) && (updateCount == -1)) break;
// done with loop
getResultSet = ps.getMoreResults();
}
```

I get:

```
Driver version is 3.40.57 (012747.007227.008728)
Driver version is 3.40.57 (012747.007227.008728)
Database Major version is 8
Database Minor version is 0
java.sql.SQLException: [BEA][SQLServer JDBC
Driver][SQLServer]Cannot insert the value NULL into
column 'bar', table 'CCTEST.dbo.joetable'; column
does not allow nulls. INSERT fails.
```

The exception arises during the getMoreResults() call.

My java code is pretty simple:

```
Connection conn = null;
CallableStatement stmt = null;
try {
  conn = ConnectionHelper.getConnection();
  stmt = conn.prepareCall("{call ta_ProcessPayroll()}");
  stmt.execute();
} catch (SQLException sqle) {
  sqle.printStackTrace();
} finally {
  ...
}
```

I tried the same code with Type 1 Driver (JDBC-ODBC Bridge) and error was caught by the SQLException. So I think this is bug with Microsoft SQL Server JDBC Driver. What do you think?

Right. Your code is too simple. Sorry to be a pain, but never rely on the

Re: Stored Procedure error is not caught

old buggy unsupported JDBC-ODBC bridge as a validity test. Your procedure will return an update count for every separate update (update/insert/delete) statement it does. If the first one fails you should get your exception from the execute() call, but later ones won't. In order for the jdbc-odbc bridge to act the way it does, it would have to buffer up all the incoming data, including possibly multiple result sets that could come before the failure, and would blow up memory. I know the issue, and the code example I showed for processing the inline returns from any stored procedure will always work. Your code should be:

```
Connection conn = null;
CallableStatement stmt = null;
try {
    conn = ConnectionHelper.getConnection();
    stmt = conn.prepareCall("{call ta_ProcessPayroll()}");
    boolean getResultSet = stmt.execute();
    int updateCount = -1;

    while (true) { // handle all in-line results from any complex procedure
        if (getResultSet) {
            ResultSet r = ps.getResultSet();
            while (r.next()) {
                // process result set if expected/wanted
            }
            r.close();
        } else {
            updateCount = ps.getUpdateCount();
            if (updateCount != -1) {
                // process update count if expected/wanted
            }
        }
        if ((!getResultSet) && (updateCount == -1)) break; // done with loop
        getResultSet = ps.getMoreResults();
    }

} catch (SQLException sqle) {
    sqle.printStackTrace();
} finally {
    ...
}
```