QGIS Application - Bug report #20592
MSSQL: Inserting a feature in a table with an after insert trigger attached failed!

2018-11-22 01:24 PM - Alexander Zidek

Status:	Closed		
Priority:	High		
Assignee:			
Category:	Data Provider/MSSQL		
Affected QGIS version:	3.4.1	**Regression?:**	Yes
Operating System:	Windows 10	**Easy fix?:**	No
Pull Request or Patch supplied:	Yes	**Resolution:**	
Crashes QGIS or corrupts data:	No	**Copied to github as #:**	28412

Description

Inserting a feature via the editor in a table with an after insert trigger fails with the following error message:

[Microsoft][ODBC SQL Server Driver][SQL Server]The target table '#tablename#' of the DML statement cannot have any enabled triggers if the statement contains an OUTPUT clause without INTO clause. [Microsoft][ODBC SQL Server Driver][SQL Server]Statement(s) could not be prepared.

Works under 3.2.3 without any problems.

Associated revisions

Revision 69f6ea52 - 2019-01-07 02:48 AM - Alex Hay

[mssql] Fix inserting features into tables with an after insert trigger attached

Fixes #20592

Revision c710e0b8 - 2019-01-07 04:19 AM - Alex Hay

[mssql] Fix inserting features into tables with an after insert trigger attached

Fixes #20592

(cherry picked from commit 69f6ea521b8ee3a78bda0dfd431a9cf6668ad111)

History

#1 - 2018-11-23 09:34 AM - Andy Gio

I'm using Qgis client with a layer linked to a Qgis Server (as WFS-T).
Qgis Server is linked to an MSSQL server.
In Qgis Server v3.2 the insert action works fine.
Now Qgis Server v3.4 return error when I try to insert a feature.

This is the RPC:Starting line from Sql Server Profiler:

--from QGIS SERVER 3.2
declare @p1 int
set @p1=-1

```
exec sp_prepexec @p1 output,N'@P1 int,@P2 int,@P3 nvarchar(4000),@P4 int,@P5 ntext',N'INSERT INTO [dbo].[vTabPolygon]
([Field1],[MI_PRINX],[Field3],[Field4],[SP_GEOMETRY]) VALUES
(@P1,@P2,@P3,@P4,geometry::STGeomFromText(@P5,3004))',NULL,NULL,NULL,0,N'Polygon (( ... ))'
select @p1

--from QGIS SERVER 3.4
declare @p1 int
set @p1=-1
exec sp_prepexec @p1 output,N'@P1 int,@P2 int,@P3 nvarchar(4000),@P4 int,@P5  ntext',N'INSERT INTO [dbo].[vTabPolygon]
([Field1],[MI_PRINX],[Field3],[Field4],[SP_GEOMETRY])  *OUTPUT inserted.MI_PRINX* VALUES
(@P1,@P2,@P3,@P4,geometry::STGeomFromText(@P5,3004))',NULL,NULL,NULL,0,N'Polygon (( ... ))'
select @p1
```

When SQL execute the second block, this is the result:
Message 334, level 16, state 1, row 21
The target table 'dbo.vTabPolygon' of the DML statement cannot have any enabled triggers if the statement contains an OUTPUT clause without INTO clause.
Statement(s) could not be prepared.

'dbo.vTabPolygon' is a view with a 'TRIGGER INSTEAD of INSERT,DELETE,UPDATE'

#2 - 2018-11-23 12:47 PM - Giovanni Manghi

- Regression? changed from No to Yes

#3 - 2018-11-24 10:25 AM - Saber Razmjooei

- Status changed from Open to Feedback

Please update to 3.4.2 and test there. There have been some improvements on the MSSQL driver.

#4 - 2018-11-26 09:15 AM - Andy Gio

Installed version 3.4.2
The problem is the same.
The insert query (from QGis to MSSql) is not correct:
the 'OUTPUT inserted.' clause generates the problem.

#5 - 2018-11-26 09:22 AM - Nyall Dawson

Andy- can you suggest an alternative approach to returning the newly created feature IDs which works with your triggers?

#6 - 2018-11-26 12:15 PM - Andy Gio

As reported in
https://blogs.msdn.microsoft.com/sqlprogrammability/2008/07/11/update-with-output-clause-triggers-and-sqlmoreresults/
the OUTPUT clause wants INTO statement,
but don't return the real id.
The triggers are execute after the sp_prepexec.
So this solution avoid the error, but it don't return the correct id.

```
declare p1 int
set @p1=-1
exec sp_prepexec @p1 output,N'@P1 int,@P2 int,@P3 nvarchar(4000),@P4 int,@P5  ntext',N'DECLARE @px TABLE (id INT);INSERT INTO [dbo].[vTabPolygon] ([Field1],[MI_PRINX],[Field3],[Field4],[SP_GEOMETRY]) @OUTPUT inserted.MI_PRINX INTO @px VALUES (@P1,@P2,@P3,@P4,geometry::STGeomFromText(@P5,3004));select * from @px',NULL,NULL,NULL,0,N'Polygon (( ... ));select id from @px'
select @p1
```

#7 - 2018-11-26 12:38 PM - Nyall Dawson

The issue is that we need a statement which reliably returns the new feature's id -- if this isn't done, then other assumptions throughout qgis are violated and there'll be a range of subtle bugs.

#8 - 2018-11-27 12:05 PM - Andy Gio

This can be a solution:

```
DECLARE @Pid int
DECLARE @P1 int,@P2 int,@P3 nvarchar(4000),@P4 int,@P5 nvarchar(max)

SET @P1 = NULL
SET @P2 = NULL
SET @P3 = NULL
SET @P4 = 0
SET @P5 = N'Polygon (( ... ))'

exec sp_executesql N'INSERT INTO [dbo].[vTabPolygon] ([Field1],[MI_PRINX],[Field3],[Field4],[SP_GEOMETRY]) VALUES ( P1,@P2,@P3,@P4,geometry::STGeomFromText(@P5,3004)); SELECT @PidOUT=@IDENTITY', N'@P1 int,@P2 int,@P3 nvarchar(4000),@P4 int,@P5 nvarchar(max),@PidOUT int output',@P1,@P2,@P3,@P4,@P5,@Pid output

select @Pid
```

#9 - 2018-12-03 12:06 PM - Andy Gio

Do you have any news to solve this problem?

#10 - 2018-12-04 01:56 AM - Nyall Dawson

Andy --

what we need is a reliable replacement for the SQL generated in https://github.com/qgis/QGIS/blob/master/src/providers/mssql/qgsmssqlprovider.cpp#L875 which works safely with your trigger.

If you can suggest revisions to that function which work for you then I'll happily review, but revising the existing sql query is out of my mssql knowledge depth.

#11 - 2018-12-04 08:55 AM - Andy Gio

The SQL function you generate does not work under any circumstances with tables or views with triggers.

In version 3.2.1 the INSERT INTO worked well.
In many cases it is necessary to use tables or views with triggers.
So different people will have the same problem and the trigger can not be replaced with other MSSQL functions. The INSERT INTO in https://github.com/qgis/QGIS/blob/master/src/providers/mssql/qgsmssqlprovider.cpp#L875 is transformed from QGIS into sp_prepexec with an OUTPUT clause and this generates the error.
You could replace the INSERT INTO of the 875 line with the stored procedure that I suggested to you and which returns the record id.

#12 - 2018-12-05 05:56 PM - Giovanni Manghi

- Status changed from Feedback to Open

#13 - 2018-12-05 08:47 PM - Nyall Dawson

- Status changed from Open to Feedback

Can you wrap that change up into a patch and submit as a pull request for wider review? I'm honestly not familiar enough with stored procedures to be able to do this myself.

#14 - 2018-12-06 05:51 PM - Andy Gio

Unfortunately, I do not program in Python.
I can try to give you a solution that I tried in C#.
The final SQL should be:

```
INSERT INTO [dbo].[Table] (
  [field1]
  ,[field2]
  ,[field3]
) VALUES (
   @p1,@p2,@p3
);
SELECT @p0 = @@IDENTITY; --this is the new line in SQL string
```

'@p0' is an output parameter for 'query' object.

In the source at Line 948

```
    //
    //EXCLUDE - START
    //if ( !( flags & QgsFeatureSink::FastInsert ) )
    //{
    //  statement += QStringLiteral( " OUTPUT inserted." ) + mFidColName;
    //}
    //EXCLUDE - END

    statement += QStringLiteral( " VALUES (" ) + values + ')';

    //
    //NEW - START
    // @@IDENTITY get the inserted id
```

```
        // @p0 is the output parameter
        if ( !( flags & QgsFeatureSink::FastInsert ) )
        {
          statement += QStringLiteral( "; SELECT @p0 = @@IDENTITY;" );
        }
        //NEW - END
```

Then at Line 968 (after the loop)

```
   for ( int i = 0; i < attrs.count(); ++i )    {
...
   }
```

you'd add the code lines for declaring the output parameter '@p0'.

At Line 1085 you'd add the code lines
to get the output value of parameter '@p0'

```
      //
      //MODIFY FOR RETURN THE VALUE OF PARAMETER @p0
      //it->setId( query.value( 0 ).toLongLong() );
      ...
      //
```

I hope this help you.

#15 - 2018-12-07 12:34 AM - Nyall Dawson

Ah - that's the approach previously used, but:

1. It doesn't work for non-identity primary keys
2. There's many issues in using @@identity - see some discussion at
https://stackoverflow.com/questions/481395/identity-scope-identity-output-and-other-methods-of-retrieving-last-identi (and many other places via google)

I think what we may need is a variation of this answer:
https://stackoverflow.com/a/13198551/1861260

The issues pointed out in that answer shouldn't (hopefully?) apply here, because we're not retrieving the timestamp, just the newly created ID.

Are you able to test a variant of the SQL using OUTPUT with a temporary table and see if that works with your trigger?

#16 - 2018-12-07 04:53 PM - Andy Gio

I read this article:
https://msdn.microsoft.com/it-it/communitydocs/server-enterprise/sql/sql-output-clause-e-triggers

I understand that the trigger must return a result-set and
in this result-set there must be the id of the inserted record.

So I modified my trigger:

```
CREATE TRIGGER [dbo].[TR_TestTB]
  ON  [dbo].[TestTB]
  INSTEAD of INSERT
AS
BEGIN
   SET NOCOUNT ON;

   INSERT INTO dbo.TestTB (
      [field1]
      ,[SP_GEOMETRY]
   )

   output inserted.MI_PRINX -- ROW ADDED TO OUTPUT A RESUL-SET WITH THE ID

   select
      [field1]
       ,[SP_GEOMETRY]
   from inserted
END
```

Now your code can send the query:

```
DECLARE @px TABLE (id INT);
INSERT INTO [dbo].[TestTB]
      ([field1]
      ,[SP_GEOMETRY])
OUTPUT inserted.MI_PRINX INTO @px
   VALUES
      (@p1,@p2);
SELECT id FROM @px;
```

This way you will get the id from a table/view with or without trigger.

You could change your code this way:

```
   QString statement;
   QString values;
   //ROW 875
   //
   //MODIFY - START
   //statement = QStringLiteral( "INSERT INTO [%1].[%2] (" ).arg( mSchemaName, mTableName );
   statement = '';
   if ( !( flags & QgsFeatureSink::FastInsert ) )
   {
     statement += QStringLiteral( "DECLARE @px TABLE (id INT); " );
   }
   statement += QStringLiteral( "INSERT INTO [%1].[%2] (" ).arg( mSchemaName, mTableName );
```

```
//MODIFY - END

...

statement += QStringLiteral( ") " );
if ( !( flags & QgsFeatureSink::FastInsert ) )
{
  //ROW 950
  //MODIFY - START
  //statement += QStringLiteral( " OUTPUT inserted." ) + mFidColName;
  statement += QStringLiteral( " OUTPUT inserted." ) + mFidColName + QStringLiteral( "  INTO @px " );
  //MODIFY - END
}
statement += QStringLiteral( " VALUES (" ) + values + ')';

//ROW 953
//ADD- START
if ( !( flags & QgsFeatureSink::FastInsert ) )
{
    statement += '; SELECT id FROM @px;';
}
//ADD - END

// use prepared statement to prevent from sql injection
if ( !query.prepare( statement ) )
{
```

#17 - 2018-12-11 01:07 AM - Nyall Dawson

- Pull Request or Patch supplied changed from No to Yes

- Status changed from Feedback to Open

Thanks Andy!

#18 - 2018-12-11 11:16 PM - Alexis Roy-L

Judging from the error message and https://docs.microsoft.com/en-us/sql/t-sql/queries/output-clause-transact-sql?view=sql-server-2017

I would assume the fix would be to change line 950 of
https://github.com/qgis/QGIS/blame/4e38193bf382e3bae7764a2daf4a8a74342b5c71/src/providers/mssql/qgsmssqlprovider.cpp#L950

Might be causing the issue as OUTPUT and VALUES are called but OUTPUT is not followed by an INTO statement that specifies the output table of OUTPUT.

This is what the error message is indicating and if we look in the examples provided by microsoft OUTPUT is followed by INTO just before VALUES.

This might fix the problem at hand.

#19 - 2018-12-12 10:39 PM - Alexis Roy-L

I have done some test on simple tables.

It is possible to get the function to trigger BUT OUTPUT must not point to the same table as INSERT INTO if INSERT INTO is specified.

I got the following to work:

```
INSERT INTO dbo.Table_1 (test3,test1,test2,pk)
OUTPUT inserted.* INTO dbo.Table_3 (test3,test1,test2,updatec,pk)
VALUES (2,'21','22',3)
```

And :

```
INSERT  dbo.Table_1 (test3,test1,test2,pk)
OUTPUT inserted.* INTO dbo.Table_3 (test3,test1,test2,updatec,pk)
VALUES (2,'21','22',4)
```

In both cases the same results are outputed to both Table_1 and Table_2.

I'm not sure what the intended behavior of OUTPUT is in the INSERT query when adding a new feature but output seem to simply return the inserted data. If this is the intended case to have the data added in the table and catch the data emmited by the OUTPUT clause, it might be best to link the OUTPUT to a temporary database in order to retrieve the info and have INSERT triggers perform properly.

Edit:The last solution proposed by Andy would be the best way to adapt the source code and still maintain the behavior associated with the OUTPUT clause without the INTO.

#20 - 2019-01-07 02:47 AM - Alex Hay

- Status changed from Open to Closed

- % Done changed from 0 to 100

Applied in changeset commit:qgis|69f6ea521b8ee3a78bda0dfd431a9cf6668ad111.