QGIS Application - Bug report #17535
Postgresql: empty SAVEPOINTS

2017-11-23 11:01 AM - Hugo Mercier

Status:	Closed		
Priority:	High		
Assignee:	Vincent Mora		
Category:	Data Provider/PostGIS		
Affected QGIS version:	master	**Regression?:**	No
Operating System:		**Easy fix?:**	No
Pull Request or Patch supplied:	No	**Resolution:**	
Crashes QGIS or corrupts data:	Yes	**Copied to github as #:**	25432

Description

When using layers from a PostgreSQL in transaction mode, some strange things happen sometimes.

In particular, an error related to SAVEPOINT happens some times upon modifications on layers.

```
ERROR:  SAVEPOINT can only be used in transaction blocks
```

.. quickly followed by

```
[ERROR:  zero-length delimited identifier at or near """"
LIGNE 1 : SAVEPOINT ""
```

... and sometimes by a segmentation fault

The problem happens on https://github.com/qgis/QGIS/blob/master/src/core/qgsvectorlayerundopassthroughcommand.cpp#L77
when mSavePointId.isEmpty() AND savePointId.isEmpty()

I can't find yet a minimal reproducible scenario to reproduce, sorry.

Related issues:

Related to QGIS Application - Bug report # 17175: Relation reference widget t...	**Closed**	**2017-09-21**

Associated revisions

Revision af647340 - 2017-12-12 05:36 PM - Matthias Kuhn

Fix crash with error on autocreate savepoint

member variables are initialized in the order they are defined in the
header. We cannot use mError to initialize mSavePointId if the former is
declared after in the header.

References #17535

History

#1 - 2017-11-23 11:53 AM - Matthias Kuhn

- Assignee changed from Jürgen Fischer to Vincent Mora

I noticed the same in some tests.

Probably the transaction is not started, might be the begin transaction command is sent in a context where it's not expected.

#2 - 2017-11-23 12:09 PM - Vincent Mora

> *might be the begin transaction command is sent in a context where it's not expected*

I don't get it. How can an edit command be issued while the transaction has not begun. I'll try and see if I can understand what's going on, but I'd really, really appreciate some pointers to reproduce the bug.

#3 - 2017-11-23 07:08 PM - Giovanni Manghi

- Priority changed from Normal to High

#4 - 2017-11-24 07:36 AM - Vincent Mora

Hugo, is there by any chance a redo command issued (or an edit command) ? Can you describe, at least rougthly what you are (or your script is) trying to do ?

#5 - 2017-11-24 08:57 AM - Vincent Mora

Hugo, can you try checking the error msg from mTransaction->begin in QgsTransactionGroup::onEditingStarted() ?

Also, adding a Q_ASSERT(!mSavePointId.empty()) in QgsVectorLayerUndoPassthroughCommand::QgsVectorLayerUndoPassthroughCommand would help diagnose the problem at it's root.

For the record, the transaction should begin on signal beforeEditingStarted from vector layer, connected in QgsTransactionGroup:

QgsVectorLayer::startEditing emit beforeEditingStarted which is connected to QgsTransactionGroup::onEditingStarted which loops over layers and call QgsVectorLayer::startEditing which returns before emitting if editing is already underway.

#6 - 2017-12-12 05:31 PM - Matthias Kuhn

I just found a way to reproduce this behavior reliably.

Setup database and project

1. Install and enable the QgsProjectGenerator plugin
2. Database -> Project Generator -> Generate
3. Source: Interlis (Use PostGIS)
4. Models: KbS_LV95_V1_3
5. CRS: EPGS:2056
6. PostgreSQL: Choose any databaseserver near you
7. Save the project, so next time there's no reason to repeat these steps ;)

Reproduce problem:

1. Toggle editing
2. Add a feature on "belasteter standort"
3. Go to tab "egrid_"
4. Click child feature
5. Enter any number in value
6. Accept
7. Debug ;)

I have isolated the reason for the crash, but not for the missing transaction block

#7 - 2018-01-08 06:05 PM - Matthias Kuhn

It looks like what happens is the following

- A transaction is created
- At some point a command fails (in the example above it's a syntax error because it's trying to use a "default value definition" instead of an evaluatated default value)
- The transaction is aborted (silently)
- The user wants to commit something on the (aborted) transaction
- The system tries to create a snapshot (and fails because the transaction is aborted)

What could be done?

- Can we run commands in a sandbox (as in "if command fails, rollback to last savepoint instead of killing the whole transaction")?
- Can we run commands outside the transaction (i.e. can we run r/o select statements after a ROLLBACK TO SAVEPOINT or some other form of checkout in a different transaction)?
- Can we at least detect when a transaction fails and report this to the user and also repaint the map (because all the work in the current transaction is gone)?

#8 - 2018-01-09 09:02 AM - Vincent Mora

Matthias, thanks for the update, but there is something I don't get: why should a syntax error abort the transaction ? It's the primary cause of the issue apparently, and normally a failed instruction could be rolled back, or am I mistaken ?

I'm pretty sure the undo/redo code checks for errors, but it may not take the right course of action when error occur (like issuing a rollback).

#9 - 2018-01-09 10:24 AM - Regis Haubourg

related to #17175

#10 - 2018-01-09 10:30 AM - Regis Haubourg

transaction abort on syntax error in PostgreSQL seems to be the normal behavior. If we want to avoid that, the idea of a sandbox sounds good.
I found that strategy for psql client : https://www.endpoint.com/blog/2015/02/24/postgres-onerrorrollback-explained

It relies on creating a savepoint and rollback to it when using the psql specific '\set ON_ERROR_ROLLBACK' session setting.

#11 - 2018-01-12 08:39 AM - Jürgen Fischer

- Related to Bug report #17175: Relation reference widget triggers SQL syntax error with UUID fields in Postgres added

#12 - 2018-02-05 09:43 AM - Vincent Mora

Regis, as far as I understand the article, creating savepoint before each command is roughly what we do to implement the undo/redo.

Postgres doc on transactions (https://www.postgresql.org/docs/8.3/static/tutorial-transactions.html) states: "ROLLBACK TO is the only way to regain control of a transaction block that was put in aborted state by the system due to an error"

So we may not need a sandbox, but simply to "ROLLBACK TO" when an error occurs. The last example the the aforementioned doc points in this direction too.

#13 - 2018-02-05 10:00 AM - Regis Haubourg

You're right Vincent.
Anyway, the ROLLBACK TO will just get back to the previous SAVEPOINT without aborting the transaction, but the error will reoccur just when the user will try again the same action.
I think we should adress all the SQL syntax errors we have first.

The one that I isolated in #17175 is that when QGIS manipulates uuid fields and happen to emit a

```
WHERE my_uuid_field = 'NULL'
```

should be changed by a

```
WHERE my_uuid_field IS NULL
```

Matthias, Hugo, did you isolate other causes ?

#14 - 2018-02-05 04:42 PM - Vincent Mora

Matthias, I've tried your procedure (project generator) and couldn't reproduce the problem with today's master.

I've also tried to reproduce the syntax error using #17175, but again couldn't.

With the article Regis mentioned and according to the postgres doc, we should be able to "ROLLBACK TO transaction_savepoint" if a commands causes the transaction to abort, which is what is done in the postgres provider. In this case the last undo/redo savepoint (which is just before the rolled back transaction_savepoint) is not dirtied which is what we want... or maybe it should be released instead ?

I have also tried to introduce a syntax error in the "DELETE" of the postgres provider, the error is caught, user notified, roll back occurs and I can create features after that (the transaction is restored and undo/redo works as expected).

Matthias, I'm wondering if by "silently" aborted you mean the transaction is aborted with no PGException raised.

#15 - 2018-11-08 01:28 PM - Giovanni Manghi

- Status changed from Open to Feedback

Please try on QGIS 3.4.1, if the issue is still valid change the affected version, thanks.

#16 - 2018-11-15 09:26 AM - Vincent Mora

Régis, Hugo, have you got a test that reproduces the issue ?

#17 - 2019-01-29 02:00 PM - Hugo Mercier

- Status changed from Feedback to Closed

The last issues with a similar behavior that I've encountered were due to concurrency problems, which should be fixed now (with https://github.com/qgis/QGIS/pull/8850)

Closing