QGIS Application - Bug report #6422
QGIS delete Postgis features after trying to merge them

2012-09-27 07:27 AM - Regis Haubourg

Status:	Closed		
Priority:	Severe/Regression		
Assignee:			
Category:	Data Provider		
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 #:**	15668

Description

Hi all, quite a rare bug, but a severe one here, since it destroys data with no way to recover without backup:

Data is in postgis 1.5, postgres 9.1. There is a primary key defined. See attached sql script to reproduce data, and postgres log file.

Editing data, I cut two multipolygon in half with advanced editing tools, in order to move one part to the another polygon. I use 'merge' tool to add reaffect the cut part to the other polygon. It goes well in qgis.
When saving (commiting in postgis), I get following error message, and loose my objects (row are deleted in postgres table!!):

Could not commit changes to layer bug_lost_feature_qgedit

Errors: ERROR: 2 feature(s) not added.
SUCCESS: 1 geometries were changed.
SUCCESS: 1 feature(s) deleted.
Provider errors:
PostGIS error while adding features: ERREUR: la valeur d'une clé dupliquée rompt la contrainte unique « pk_bv_masse_eau »
DETAIL: La clé « (eu_cd, version_dce)=(FRFT34, EDL_2013) » existe déjà.

- First problem: QGIS first insert a data, that is a double on key.. > postgres shouts against it. Is it possible to make insert only after delete?
- Second problem: RollBack instruction should be able to rollback ALL transactions QGIS does when saving, and not only subparts.

Related issues:

Related to QGIS Application - Bug report # 5475: Problem to insert splitted g...	**Closed**	**2012-04-23**
Related to QGIS Application - Bug report # 6164: postgis layer: duplicate pri...	**Closed**	**2012-08-07**
Related to QGIS Application - Bug report # 14247: can't merge features if res...	**Closed**	**2016-02-08**
Duplicates QGIS Application - Bug report # 5758: Primary key issue when using...	**Closed**	**2012-06-07**
Duplicated by QGIS Application - Bug report # 6872: "ERROR: duplicate key va...	**Closed**	**2012-12-13**

Associated revisions

Revision 94156ded - 2013-01-05 01:10 AM - Jürgen Fischer

QgsVectorLayer: delete before adding/changing features (fixes #6422)

History

#1 - 2012-09-27 07:31 AM - Giovanni Manghi

- Crashes QGIS or corrupts data changed from No to Yes

#2 - 2012-09-27 07:42 AM - Jürgen Fischer

- Category changed from Data Provider/PostGIS to Data Provider

addFeatures and deleteFeatures are independant operations on provider level. This would need a change in the provider interface. The duplicate key is also a problem on provider level - we currently don't have a way to expose which attributes are part of the primary key and need to be skipped, when copying attributes of existing features to new features. See also #5475.

#3 - 2012-09-27 08:00 AM - Regis Haubourg

OK, but saving does call AddFeature before deleteFeature if I read logs well. Any reason not to call DeleteFeature first?

#4 - 2012-12-14 02:56 AM - Regis Haubourg

Hi, #6872 points same bug. A little up on my last comment. Can we have provider delete features before trying to add them. It could be a way to prevent adding duplicates.
We should also embed this into a begin / rollback on error so that error message is raised and no commit to data is made.
Régis

#5 - 2012-12-14 04:42 AM - Jürgen Fischer

regis Haubourg wrote:

> *We should also embed this into a begin / rollback on error so that error message is raised and no commit to data is made.*

As said, that's not possible, unless we change the provider interface.

#6 - 2012-12-16 09:23 AM - Regis Haubourg

Thanks Jurgen,
I maybe should ask differently, sorry. This bug is a blocker since it destroys data. Is changing provider interface hard or complex thing? does it break anything?
Who is capable of doing it? Does it help funding some work?
Regards
Régis

#7 - 2012-12-16 09:57 AM - Jürgen Fischer

regis Haubourg wrote:

> *I maybe should ask differently, sorry. This bug is a blocker since it destroys data. Is changing provider interface hard or complex thing? does it break anything?*

I was only referring to the "embed this into begin/rollback" part - just changing the commit order of operation (delete first, insert after) in the vector layer would probably already fix this and not require a big change.

But adding transaction on a higher level would require that we add transactions to the vector layer provider interface and update all the vector data providers (delimitedtext, gpx, grass, memory, mssql, ogr, osm, postgres, spatialite & sqlanywhere) to support it.

Some providers (or their respective backend) might already support transactions (I suppose all database providers do) and some might not (gpx, delimitedtext, memory...) and some might optionally (OGR - depending on the respective datasource there, GRASS?). Some might already use transactions internally and need a change to make it available to the outside.

So "transactions" would probably need to be an optional provider capability.

> *Who is capable of doing it? Does it help funding some work?*

Um, I suppose most of our devs are and I guess funding always help - but the amount of work above is not easy to estimate...

#8 - 2012-12-16 12:07 PM - Regis Haubourg

Thanks Jurgen
could we try to estimate the benefits of having transaction implemented for providers?
I see at least:
- recover of any commit error (network disconnection, constraint error, database failure)

Any other ?

Switching delete/insert order is fine for PK constraints, but not for other constraints (geometrytype..). Do we have other issues raised for other constraints? If not, we should keep simple as suggested.
Regis

#9 - 2012-12-16 02:20 PM - Jürgen Fischer

regis Haubourg wrote:

> *Thanks Jurgen*
> *could we try to estimate the benefits of having transaction implemented for providers?*
> *I see at least:*
> *- recover of any commit error (network disconnection, constraint error, database failure)*

The changes are applied by operation, ie. add features, delete features, change attributes, change geometries.

For postgres each part of a operation is already done in a transaction. If one piece of the operation fails, the whole operation is rolled back, reported as unsuccessful and therefore kept in the editing session.

That means if you add multiple features in one session, you either have all features saved to the database or none. In the latter case the editing session will be kept open (as the addFeatures failed) and the changes are still there and pending - so for the cases above you could still commit the changes, when the database comes back or fix the changes that violate constraints.

If you mix operations in one session, it might lead into a limbo state. The successful operations are applied and removed from the editing session and the unsuccessful parts stay in the editing session.

For postgres, that brings up the question - sure if that worked, it would still be ugly - if trying to commit twice would help in the ticket case. In the first attempt the add fails, but the delete is successful, in the second attempt the add is also successful, because the conflicting feature was already deleted in

the first run. If so there at least isn't a data loss - unless you decide to discard the session.

For other providers it might be worse as the provider might not be able to rollback unsuccessful operations completely: the successful parts of the failing operation would be already applied to the data source, but still be kept in the editing session as if they weren't, because the operation wasn't completely successful. So if you decide to the discard the session later, you would be still have a partly changed data source.

#10 - 2012-12-27 12:53 PM - Giovanni Manghi

- Priority changed from High to Severe/Regression

#11 - 2013-01-04 04:10 PM - Jürgen Fischer

- Status changed from Open to Closed

Fixed in changeset commit:"94156deda42421745df16132a9722fe836975c82".

Files

data_sample_log.sql	74 KB	2012-09-27	Regis Haubourg