QGIS Application - Bug report #17990
Values reset to default when cutting line

2018-01-29 08:36 AM - Daan Goedkoop

Status:	Closed			
Priority:	Normal			
Assignee:				
Category:	Digitising			
Affected QGIS version:	master	**Regression?:**	Yes	
Operating System:		**Easy fix?:**	No	
Pull Request or Patch supplied:	No	**Resolution:**	fixed/implemented	
Crashes QGIS or corrupts data:	No	**Copied to github as #:**	25886	

Description

When I cut a line in a PostGIS line layer, the newly created features have their values changed to the DEFAULT value of the database (wherever applicable).

History

#1 - 2018-02-05 11:13 PM - Nyall Dawson

- Status changed from Open to Feedback

This is by design - it's done to allow default values which are based on sequences and other logic - e.g. populating a primary key field with a new unique id.

#2 - 2018-02-06 07:46 AM - Daan Goedkoop

I understand this argument.

However, that example (a primary key with an auto-generated unique id) worked already without problems in 2.18 and before.

But what about columns without a unique constraint? E.g. fields like "digitisation_date" (default now), "one_way_road" (default false) or "already_inspected" (default false). When cutting a feature, the distinction between the "original" feature and the cut-off "new" features is merely a technical one, and not visible to the user. So that some of the resulting parts have their values set to default and one part not, is not very intuitive, I think.

#3 - 2018-02-08 06:23 AM - Nyall Dawson

> *However, that example (a primary key with an auto-generated unique id) worked already without problems in 2.18 and before.*

Actually that's not the case - there's numerous bug reports about broken behavior of the split tool in 2.18 when a layer has a field with unique constraints (such as sqlite/geopackage layers).

I think a possible solution here would be to discard only columns with default value clauses (as opposed to default value literals). So if the default value is an expression, it is recalculated. But if it's a literal (e.g. false) then we would keep the original value. I'm just not sure if it's even possible to differentiate these two default types from a postgres database...

#4 - 2018-02-17 11:25 AM - Daan Goedkoop

Apparently QGIS 2.18 has the following priorities:

1. For primary key columns, use the DEFAULT value or else NULL

2. For any other column, use the supplied value

And QGIS 2.99 has the following priorities - apparently since November 2016 already:

1. Client-side default value

2. Provider-side default clause

3. Provider-side default literal value

4. Supplied value

5. An automatic value for UNIQUE constraint columns without a default clause

The PostGIS provider returns any DEFAULT expression for 2. and will try to perform a SELECT on the DEFAULT expression in order to obtain 3. Theoretically one could compare both; an identical result would indicate a literal value. But that doesn't seem very robust.

If the problems were with UNIQUE columns, then why not give the default clauses only precedence for those columns? Hmm... I know this is a bit short before the release date to discuss something like this.

Otherwise, could I at least suggest to mention this change in behaviour in the changelog?

#5 - 2018-02-20 01:10 PM - Daan Goedkoop

This also seems to affect the copy/paste functionality. Could there be an overview whether there are any other affected functions?

#6 - 2018-02-25 11:20 PM - Giovanni Manghi

- Operating System deleted (Windows)

Where do we stand with this ticket?

#7 - 2018-02-27 09:29 AM - Daan Goedkoop

The Visual Changelog for 3.0 mentions "Feature: Improved handling of defaults".

I have added a sentence to this section about how it affects "certain editing operations (e.g. copy-paste, split features etc."

For me, the resetting to default for non-unique columns still feels like an unexpected side effect. However, apparently I am alone in this. Also, this is part of a change developed with sponsor money. I don't have any contacts to the sponsors, so I don't know what behaviour they requested exactly.

#8 - 2018-02-28 07:33 AM - Nyall Dawson

Daan,

Please be aware that the current logic isn't set in stone - it's open for review and amendment!

The current logic was carefully designed to fix numerous problems in earlier versions with editing operations failing due to constraint violations in providers (this was affecting Postgres and sqlite based formats (geopackage, spatialite)... likely other database providers were also affected).

I'm coming round to the conclusion that default priorities 3 & 4 SHOULD be swapped - i.e. we prioritise existing values over provider side literal defaults. But that doesn't help (yet) with postgres, because the postgres provider can't currently differentiate between literal defaults and clause based defaults. So perhaps the ultimate fix here is a combination of tweaking priorities 3 & 4 and implementing literal default detection for postgres provider (assuming it's

possible to do)

#9 - 2018-12-28 12:18 PM - Paolo Cavallini

What is the current situation? Could you please check and report back?

#10 - 2018-12-28 09:15 PM - Nyall Dawson

- Status changed from Feedback to Closed

This is fixed in 3.4

#11 - 2018-12-29 12:04 PM - Giovanni Manghi

- Resolution set to fixed/implemented