Redmine - Defect #15306

No database update on git history rewrite

2013-11-08 11:22 - B P

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	SCM	**Estimated time:**	0.00 hour
Target version:			
Resolution:	Duplicate	**Affected version:**	2.3.3

Description

Versions

- redmine 2.3.3.stable
- ruby 2.0.0
- Rails 3.2.13

Bug summary:

I refer to issues as much as I can in my commit messages.
While migrating a repository from another ticket tracking
system into redmine, I linked redmine with an existing
repository. Then, obviously the tickets in redmine were
not in sync with the tickets referenced in the commits.

So I rewrote git history by doing a huge

```
git rebase -i HEAD~42
```

to change the referenced tickets into the new ones. And
then I did a

```
git push -f
```

to overwrite the repository's history. But then, I've been surprised to see no updates in the tickets. e.g.:

- ticket 1 should have become ticket #42 and
- new ticket #1 should have had been referenced by no commits...
  But after my change ticket #1 was still having references
  from commits.

I tried removing and re-adding the repository, but that
had no effects.

I finally found out that even after removing the repository,
the changesets table in the database was still having all
commits for the repository.

- Q&D solution:

To solve my problem I ended up issuing a:

```
delete from repositories where id=3;
delete from changesets where repository_id=3;
```

which finally removed the old commits from references of
the tickets. Though, now I'm afraid to have n-n tables
still having orphans.

Here is for the overall context of the bug, I did not dig further, as I'm not having a development instance of redmine and I need my production instance to work correctly. I neither checked whether that bug was still there at HEAD of redmine, and finally I'm not sure this is not a duplicate, though I've searched around the issues and found nothing.

- Reproduce

But I think that bug is definitely reproduceable:

1. make a new redmine instance
2. create a ticket #N
3. make a commit referring to that ticket (e.g. git ci -m "fixes #N")
4. git push the commit to the repository redmine is reading from
5. reload the ticket, the commit is now visible
6. rewrite the history: git rebase -i HEAD~42 or simply git commit --amend -m "fixes #M"
7. git push -f that commit to rewrite repository's history as well
8. reload the ticket, the commit is still visible whereas it shall not be visible anymore

HTH,

Guyzmo

Related issues:

Is duplicate of Redmine - Feature #12853: Removing git branch result of datab...	**New**

History

#1 - 2013-11-09 02:26 - Toshi MARUYAMA

- Status changed from New to Closed

- Resolution set to Duplicate

Duplicate with #12853.

#2 - 2013-11-09 02:26 - Toshi MARUYAMA

- Is duplicate of Feature #12853: Removing git branch result of database inconsistencies added