VegBank - Bug #1861

fix denorm SQL approach

01/10/2005 03:15 PM - Michael Lee

Status:	Resolved	**Start date:**	01/10/2005
Priority:	Normal	**Due date:**	
Assignee:	Chad Berkley	**% Done:**	0%
Category:	userLoad	**Estimated time:**	0.00 hour
Target version:	1.1.0	**Spent time:**	0.00 hour
Bugzilla-Id:	1861		

Description

denorm sql found here:

vegbank/src/sql/denorm/denorm-nullsonly.sql

safe to run at any time, will be needed after a load, as denorm fields NOT included in XML

Related issues:

Blocked by VegBank - Bug #2402: Strategy to update denormalized data, cache, ...	**New**	**04/06/2006**

History

#1 - 07/21/2005 06:27 PM - Michael Lee

denorm SQL should be added to SQLStore and referenced from there, using keys and where statements. UPDATE needs to be allowed, probably ok. This way, we can configure denorms to run, instead of doing everythign in the db

#2 - 08/16/2005 03:10 PM - Michael Lee

vegbank:denorm now exists, though a utility will be built that will steal its code and then it will use the utility. Useful to set all null values to denorm values after a load to db, and useful to force update of one plant/comm/observation by passing its accessioncode as wparam

#3 - 04/06/2006 06:05 PM - Michael Lee

We need to figure out how best to update denorm fields, and run that after XML loads.

A number of fields are "denorm" fields. These are indicated as attModel='denorm' in /vegbank/docs/xml/db_model_vegbank.xml.

Denorm fields are updated in two ways. The first approach was a .sql file (see /vegbank/src/sql/denorm). Then we added a java utility. I don't know what that's called. The utility uses SQL snippets from the SQLStore.properties file (/vegbank/src/java/org/vegbank/common/SQLStore.properties). We used AJaX to update the denorm fields with this utility or with custom jsp tags (vegbank:denorm). The AJaX approach failed because of http timeouts since the updates take a while sometimes.

#4 - 06/14/2006 01:43 PM - Michael Lee

the SQL that denormalizes fields is in:
vegbank/src/sql/denorm
2 files, one has SQL to update everything, and one just updates values that are null

Denormalization fields are highlighted in red in the data dictionary:
http://vegbank.org/vegdocs/dbdictionary/dd~table~plantconcept~type~tableview.html
has plantName and d_obsCount as denorm fields. Each denorm field looks up data in another field, or counts how much data meets some criteria. All the logic is in the denorm SQL files.

We will probably need 2 kinds of triggers. One will be a trigger that runs once per SQL transaction that will update records after someone inserts a bunch of new ones. The other would be a one-by-one trigger that updates rows whenever the data is updated in a row.

I read a fair bit about triggers at:
http://www.postgresql.org/docs/8.1/interactive/triggers.html

and nothing seemed out of the ordinary about what we want to do. Keywords may take a more clever way of figuring it out, since temporary tables are generated and it's just on a bigger scale.

#5 - 06/20/2006 01:38 PM - Chad Berkley

I've attempted to create a trigger system for updating the denorm fields when observation is updated. This seems to be creating a problem with the transaction system. postmaster aborts the entire transaction whenever the trigger starts to fire. I think I need to finda a different time to fire the trigger, probably once all of the updates to observation have been made during the xml load transaction. It doesn't seem to matter if i fire the trigger on row updates or statement updates, i still get the exceptions.

I think a solution to this would be to fire the trigger after the entire load transaction finishes, but I need to figure out exactly where to do that.

#6 - 06/21/2006 10:06 AM - Chad Berkley

I got a trigger working with the transaction system in vegbank. The problem now is that the subselect based updates in the denorm-forceupdate.sql file take forever to execute. If we implement this in a trigger, it's going to slow down the xml loading significantly. Each individual statement in the denorm-forceupdate.sql script takes between 20 seconds and 2 minutes to execute (on the command line). There are probably 100 or so of these statements in that file. So I don't think the approach of using a trigger to run these updates is going to work.

The other solutions that come to mind here are:
1) change the sql to make it faster. I'm not sure what this would entail since I'm still fuzzy on exactly what the denorm fields are doing. We might also consider removing some of the denorm fields alltogether if they are not being heavily used. Again, I can't make an educated decision on this without talking with michael.

2) run the denorm in a seperate thread after doing a load. This would allow the denorm to run in the background and take as much time as it needs. The problem with this is that if we get a bunch of people loading data at the same time, it could kill the database system with threads all trying to so subqueries and updates.

We need to have a chat on the next conf. call about what to do with this. Right now, I've noticed that the LoadTreeToDatabase.runDenorms() is commented out so the denorms are not running, probably because of this performance hit. I think we need to look at this closely and see if the performance increase we get from doing manual denormalizations is worth the additional processor time that it takes to compute them.

#7 - 06/21/2006 10:13 AM - Chad Berkley

Another note about my comment from a few minutes ago. I also tried the denorm-nullsonly.sql file which takes less time than the denorm-forceupdate.sql file, but still takes a while. I'm currently experimenting with it to see if its feasible to use it instead.

#8 - 09/07/2006 04:33 PM - Michael Lee

now uses the utility. Not entirely happy with that, but that's the way it goes.

#9 - 03/27/2013 02:18 PM - Redmine Admin

Original Bugzilla ID was 1861