Database - Bug #7165

Close multiplex scope using prepared statement

03/02/2023 08:22 AM - Alexandru Lungu

Status:	Test	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Alexandru Lungu	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

Related issues:

Related to Database - Feature #6830: find and fix all SQL SELECT statements w...	**Closed**

History

#1 - 03/02/2023 08:34 AM - Alexandru Lungu

TemporaryBuffer.removeRecords is widely used when closing a multiplex scope, but not only - e.g it is also used with copy-temp-table or empty-temp-table. The code looks quite complex, but I am sure it can be comprehended and refactored to use prepared statements:

```
delete from dtt1 where _multiplex = 1745
```

to

```
delete from dtt1 where _multiplex = ?
```

Note that we can use TemporaryBuffer.removeRecords with explicit where clause and args.

#2 - 03/03/2023 05:40 AM - Alexandru Lungu

We should also address the use of prepared statements when using FastCopyHelper. Most of the statements executed there are prepared, but there are some places in which the multiplex is hard-coded FastCopyHelper.getSrcTempColumns, FastCopyHelper.getSrcMappedColumns. These are the methods used to define the columns of the source table for a INSERT-SELECT operation. The multiplex should be explicit to ensure that the records reach the target table with a specified multiplex. However, the multiplex is more then explicit; it is hard-coded.

#3 - 03/07/2023 03:47 AM - Radu Apetrii

Alexandru Lungu wrote:

> TemporaryBuffer.removeRecords is widely used when closing a multiplex scope, but not only - e.g it is also used with copy-temp-table or empty-temp-table. The code looks quite complex, but I am sure it can be comprehended and refactored to use prepared statements:

I've refactored this. The arguments are stored separately from the ones received as a parameter to the method in order to ensure their correct order. Right before the statement is executed, they get merged.

> We should also address the use of prepared statements when using FastCopyHelper. Most of the statements executed there are prepared, but there are some places in which the multiplex is hard-coded FastCopyHelper.getSrcTempColumns, FastCopyHelper.getSrcMappedColumns.

Done. I've added the multiplexID to the list of arguments and removed the "hard-coding operation".

Additional note: In FastCopyHelper.safeTableCopy I believe there was a typo. In order to call persistence.executeSQL with arguments, I don't think the condition args.isEmpty() is the one intended, so I just reversed it. It is very likely that this was solved on another issue, I just couldn't find it.

The changes were committed to 7061a, rev.14487 and tested with various examples and a large customer application.

#4 - 03/14/2023 04:03 AM - Alexandru Lungu

- % Done changed from 0 to 100

- Status changed from WIP to Review

Radu, please move 7061a, rev. 14487 to 7026b.
I will review, test and profile it once there.

#5 - 03/14/2023 04:47 AM - Radu Apetrii

Alexandru Lungu wrote:

> Radu, please move 7061a, rev. 14487 to 7026b.
> I will review, test and profile it once there.

The changes have been committed to 7026b, rev. 14501.

#6 - 03/15/2023 05:19 AM - Alexandru Lungu

Review of 7026b/rev. 14501

- In TemporaryBuffer.java, add a proper default capacity for additionalArgs and ooArgs (maybe 2?). Otherwise, the array will be expended to 32 by default (too much for our needs).
- I am OK with the changes overall!

Tested 7026b/rev. 14503 and got -1.8% improvement overall.
There was no regression in the POCs I tested.

#7 - 03/15/2023 11:14 AM - Alexandru Lungu

- Related to Feature #6830: find and fix all SQL SELECT statements with inlined literal arguments, and rewrite it using arguments and prepared statement added

#8 - 03/16/2023 05:52 AM - Radu Apetrii

Alexandru Lungu wrote:

> - In TemporaryBuffer.java, add a proper default capacity for additionalArgs and ooArgs (maybe 2?). Otherwise, the array will be expended to 32 by default (too much for our needs).

I've added an initial capacity to those lists. The change is on 7026b, rev. 14506.

#9 - 03/23/2023 04:25 AM - Alexandru Lungu

- Status changed from Review to Test

#10 - 04/20/2023 07:13 AM - Alexandru Lungu

This was merged in trunk as rev. 14523 and can be closed.