Database - Feature #7366

improve performance of buffer.fill

05/19/2023 02:36 PM - Constantin Asofiei

Status:	WIP	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Constantin Asofiei	**% Done:**	80%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**version:**	
vendor_id:	GCD		

Description

History

#1 - 05/19/2023 03:01 PM - Constantin Asofiei

Created branch 7366a from trunk rev 14573.

7366a/14574 contains a first attempt at improving performance of Buffer.fill. This gives ~1% improvement for a large app.

The main takeways/unknowns are:

- BEFORE-TABLE management - should direct record access be disabled in this case? If yes, check if BEFORE-TABLE can be handled explicitly.
- is the destination table in a FILL operation always a temp-table? If yes, then triggers can never be invoked. I'm not sure if other validation needs to be explicitly done.
- replace Runnable lambdas with a Runnable instance and check performance (especially in processDataSource).

#2 - 05/23/2023 10:15 AM - Constantin Asofiei

- % Done changed from 0 to 80

7366a rev 14575 contains the changes for FILL performance improvements. Extent is not added yet.

#3 - 05/23/2023 04:46 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Created branch 7366a from trunk rev 14573.
>
> 7366a/14574 contains a first attempt at improving performance of Buffer.fill. This gives ~1% improvement for a large app.
>
> The main takeways/unknowns are:
>
> - BEFORE-TABLE management - should direct record access be disabled in this case? If yes, check if BEFORE-TABLE can be handled explicitly.

Yes, I think this can use direct access. All BEFORE-TABLE-s have a single, non-unique index. And, as with the other temp-tables, they do not have triggers to fire.

> - is the destination table in a FILL operation always a temp-table? If yes, then triggers can never be invoked. I'm not sure if other validation

needs to be explicitly done.

Only temp-tables can be FILL-ed. In fact a temp-table must be first added to a dataset before this operation. The buffers of permanent tables cannot be components of a dataset.
I guess the only validation to be performed is null checking.

- replace Runnable lambdas with a Runnable instance and check performance (especially in processDataSource).

Yes, extracting the independent lambdas as Runnable instance makes sense, but in my tests the gain is only 10% when the code to be executed *is empty*. Adding code to be executed will cause the gain to be unnoticeable.

#4 - 05/24/2023 01:37 AM - Constantin Asofiei

Ovidiu, BEFORE-TABLE is not involved during FILL, as TRACKING-CHANGES is not allowed for FILL. So there is no need manage the BEFORE-TABLE records.

About null validation: how do you mark a temp-table field as non-null in 4GL? MANDATORY doesn't work.

#5 - 05/24/2023 10:28 AM - Ovidiu Maxiniuc

I was pretty sure if you use

```
DEFINE TEMP-TABLE <name> LIKE <permanent-table-with-mandatory-fields>
```

the mandatory attribute is preserved for the new fields.

However, this does not happens. The corresponding fields from the new temp-table will be able to store ? value. I googled it and there is the following kb article:
https://community.progress.com/s/article/Is-it-possible-to-make-a-temp-table-field-mandatory.

Actually, this is a good news. We can simplify the Validation.validateMaybeFlush() method for temp-tables.

#6 - 05/24/2023 12:12 PM - Constantin Asofiei

rebased branch 7366a from trunk rev 14584 - new rev 14586.

#7 - 05/26/2023 10:56 AM - Constantin Asofiei

Branch 7366a was merged to trunk rev 14589 and archived. What's left is a possible improvement for extent fields.

#8 - 03/21/2024 03:32 PM - Constantin Asofiei

Idea of improvements:

- the data-source query may or may not use all record fields, depending on the mapping. We can use the FIELDS option to force populating only those fields.
- try to create a FQL JOIN between the top and child buffer's data source queries, read all the records and cache them in a map on the 'join' condition.

#9 - 04/22/2024 08:03 AM - Constantin Asofiei

Changes related to this are in 8363g rev 15158: Allow the query to be reset to its initialization state, so the same instance can be used to execute child buffer FILL operations.

#11 - 04/22/2024 09:20 AM - Constantin Asofiei

Eric/Ovidiu: please review 8363g rev 15158.

#12 - 04/22/2024 12:36 PM - Ovidiu Maxiniuc

Review of 8363g, r15157/8.

I think I understand the idea: if the same query can be reused for a child buffer, it does not make sense to recreate it for each iteration of the parent buffer. So instead of close/open, we so a clearResult() instead. There one thing I am not sure I fully understand: the optimization is active only when the fill() is called for the parent database. Why not use this when an individual buffer is called? For example, a top-level-buffer with one or multiple children?

I see no other issues with the new code.

#13 - 04/23/2024 04:25 AM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> I think I understand the idea: if the same query can be reused for a child buffer, it does not make sense to recreate it for each iteration of the parent buffer. So instead of close/open, we so a clearResult() instead. There one thing I am not sure I fully understand: the optimization is active only when the fill() is called for the parent database. Why not use this when an individual buffer is called? For example, a top-level-buffer with one or multiple children?

You are right, I need to add a fillInt method to be called from the main Buffer.fill and for the child-buffer FILL in processDataSource. I'll see how I can change these.