Database - Feature #6444

dataset improvements

05/27/2022 02:21 AM - Eric Faulhaber

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Igor Skornyakov	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**vendor_id:**	GCD

Description

Related issues:

Related to Database - Feature #3809: ProDataSet support	**Closed**
Related to User Interface - Feature #7032: implement the EDITOR:EMPTY attribute	**New**
Related to Base Language - Bug #7193: READ-JSON is not supported for the "LON...	**Closed**
Related to Base Language - Bug #7247: Dataset:READ-XML does nothing	**Closed**
Related to Database - Bug #6475: double-check and fix any read/write-xml/json...	**New**

History

#1 - 05/27/2022 02:24 AM - Eric Faulhaber

The following areas need to be implemented or improved:

- DATA-RELATION attributes RECURSIVE, FOREIGN-KEY-HIDDEN are marked runtime stubs but really should have been marked runtime partial; what is left?
- DATA-SOURCE RESTART-ROW attribute is marked runtime stubs
- ATTACHED-PAIRLIST is marked conversion none, runtime full?
- ~~EMPTY attribute~~ (moved to #7032 as this is only for the EDITOR widget)
- DATA-SOURCE-COMPLETE-MAP attribute (dataset buffer)
- FOREIGN-KEY-HIDDEN attribute marked runtime stubs
- FILL-WHERE-STRING attribute - runtime is marked partial, what is needed?
- WHERE-STRING attribute is marked runtime stubs
- SYNCHRONIZE() method (dataset buffer) - marked runtime stubs

#2 - 06/17/2022 10:25 AM - Eric Faulhaber

Ovidiu, could you please review the list above and indicate what the actual support is currently (and update gap marking accordingly)?

#3 - 09/09/2022 12:07 AM - Eric Faulhaber

- Assignee set to Ovidiu Maxiniuc

#4 - 11/18/2022 01:50 PM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> Ovidiu, could you please review the list above and indicate what the actual support is currently (and update gap marking accordingly)?

Attribute/	Object type	Keyword ID	Conversion support	Runtime support

Method				
RECURSIVE	DATA-RELATION	KW_RECURSE	Complete	At least partial: getters and setters done. The semantics is probably provided by implementation, but this needs to be proved by testcases.
FOREIGN-KEY-HIDDEN	DATA-RELATION	KW_F_KEY_H	Complete	Partial: getters and setters done. Not implemented in JSON / XML serialization. Most likely it's just a set of if conditionals to skip the rows if attribute is active.
DATA-SOURCE	Buffer object	KW_DATA_SRC	Complete	Complete
RESTART-ROW	DATA-SOURCE	KW_REST_ROW	Complete	Stub only. To be implemented. Note that there is a very similar attribute RESTART-ROWID which is fully functional.
ATTACHED-PAIRLIST	Buffer object	KW_ATT_PLST	Complete	Complete
EMPTY	EDITOR widget	KW_EMPTY	Marked as none in GAP	Marked as none in GAP and this seems correct. Not related to datasets. OTOH, the EMPTY-DATASET (kw_empty_ds) and EMPTY-TEMP-TABLE (kw_empty_tt) should be fully supported. The methods are emptyDataset() and deleteAll() respectively.
DATA-SOURCE-COMPLETE-MAP	Buffer object	KW_DATA_SCM	Complete	Complete
FILL-WHERE-STRING	DATA-SOURCE	KW_FILL_WST	Complete	Complete
WHERE-STRING	DATA-RELATION	KW_WHERE_ST	Complete	Complete
SYNCHRONIZE	Buffer object	KW_SYNCHRON	Complete	Stubbed in Buffer interface but not implemented at all. LE: This refers to 3821c/trunk branch. The 6129b have a (partial) implementation of this method.

I updated the gaps so that it reflects the current state also visible in this table. Revision is 14372/3821c. I took the liberty of highlighting the remaining work in red.

#5 - 11/28/2022 01:25 PM - Eric Faulhaber

- Assignee changed from Ovidiu Maxiniuc to Igor Skornyakov

Ovidiu, thank you for the summary of status. I am reassigning to Igor for implementation. Please assist him if he has questions. All these sub-features have equal priority from my point of view, but if you think it makes sense to implement them in a certain order, please post your suggestions.

Greg, Constantin, 3821c or 6129b for these changes? They most likely will be needed in 6129b first, but we should do this in the way that makes the most sense from the standpoint of effort to merge and risk.

#6 - 11/28/2022 01:29 PM - Igor Skornyakov

- Status changed from New to WIP

#7 - 11/28/2022 03:52 PM - Greg Shah

> Greg, Constantin, 3821c or 6129b for these changes? They most likely will be needed in 6129b first, but we should do this in the way that makes the most sense from the standpoint of effort to merge and risk.

We are actively implementing changes to dataset processing in 3821c. I suspect it is best to put the changes there.

#8 - 11/28/2022 04:01 PM - Igor Skornyakov

Greg Shah wrote:

> > Greg, Constantin, 3821c or 6129b for these changes? They most likely will be needed in 6129b first, but we should do this in the way that makes the most sense from the standpoint of effort to merge and risk.
>
> We are actively implementing changes to dataset processing in 3821c. I suspect it is best to put the changes there.

OK. Please note however, that I've made a number of changes to dataset processing in 6129b in the scope of #6453. Do we plan to merge 6129b with 3821c in the near future?
Thank you.

#9 - 11/28/2022 04:02 PM - Ovidiu Maxiniuc

- Related to Feature #3809: ProDataSet support added

#10 - 11/28/2022 04:15 PM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> Ovidiu, thank you for the summary of status. I am reassigning to Igor for implementation. Please assist him if he has questions. All these sub-features have equal priority from my point of view, but if you think it makes sense to implement them in a certain order, please post your suggestions.

Probably in the exact order from the table. The first two just need to tested or small changes added. This will help getting in touch with datasets. The last two need in depth research of how they work. These attributes/methods were not in focus of the task (#3809) which added initial support for datasets because they were not present in the report of the used features of the customers, so I know less of them. The SYNCHRONIZE method seems to me to be easier so I recommend working on it before RESTART-ROW. For this last attribute I am not sure how the implementation should work. It seems to require row iterations in the background for counting and positioning on requested row.

#11 - 11/29/2022 10:16 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Eric Faulhaber wrote:
>
> > Ovidiu, thank you for the summary of status. I am reassigning to Igor for implementation. Please assist him if he has questions. All these sub-features have equal priority from my point of view, but if you think it makes sense to implement them in a certain order, please post your suggestions.
>
> Probably in the exact order from the table. The first two just need to tested or small changes added. This will help getting in touch with datasets. The last two need in depth research of how they work. These attributes/methods were not in focus of the task (#3809) which added initial support for datasets because they were not present in the report of the used features of the customers, so I know less of them. The SYNCHRONIZE method seems to me to be easier so I recommend working on it before RESTART-ROW. For this last attribute I am not sure how the implementation should work. It seems to require row iterations in the background for counting and positioning on requested row.

Some parts of the functionality to be implemented is obviously related to what was done in 6129b in the scope of #6453.
Maybe it makes sense to manually port these changes to 3821c?
Thank you.

#12 - 11/29/2022 12:37 PM - Ovidiu Maxiniuc

I assume you talk about the recent serialization updates for temp-tables and FOREIGN-KEY-HIDDEN attribute. I do not think these are related. Yes, they share the same source code and, indeed, bzr will struggle and probably will require human intervention when 6129b will be rebased. I do not know how this attribute will change the output xml, if the tree-like structure will be kept or only the presence of the keys will be changed. At least, they expect significant smaller XMLs, in fact this is the reason for this attribute. The other items are not really related and will alter different source code.

If we cherry-pick some changes I guess will confuse even more bzr. I've seen cases when it marked similar changes as conflicts even if they were really the same code backported as you suggested. So I see no advantage in doing this.

#13 - 11/29/2022 01:06 PM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> I assume you talk about the recent serialization updates for temp-tables and FOREIGN-KEY-HIDDEN attribute. I do not think these are related. Yes, they share the same source code and, indeed, bzr will struggle and probably will require human intervention when 6129b will be rebased. I do not know how this attribute will change the output xml, if the tree-like structure will be kept or only the presence of the keys will be changed. At least, they expect significant smaller XMLs, in fact this is the reason for this attribute. The other items are not really related and will alter different source code.

Yes, I have FOREIGN-KEY-HIDDEN support in mind in the first place. But, as you know, the #6453 changes are significant and affect many places, not only XMLImport/XmlExport. For example DataSet in 6129c has multiple changes made by different people which are not in 3821c yet. Maybe it makes sense to rebase 6129b now and merge it to 3821c? I rememeber that Constantin planned to do at least a 6129b rebase some time ago. The longer we postpone this the more problems we will potentially have. It can happen that after rebase it will take substantial time to re-test them and (possibly) fix the regressions. I just want to avoid doing the same job twice or at least to make it easier.

As I can see we have a lot differences between 6129b and 3821c now. Sooner or later we'll have to deal with this.

#14 - 11/30/2022 11:20 AM - Igor Skornyakov

The OE documentation about RECURSIVE Data-Relation object I see the following statement about such realtions:

```
Support is provided for a recursive Data-Relation during a FILL of a ProDataSet or temp-table buffer. Navigati
on is not supported, but can be done normally through .NET, Java, or a recursive ABL procedure.
```

This looks cryptic for me.
I've found some documentation about FILL and I'm trying to understand it now.
However I have not found any explanation what recursive ABL procedure means in this context.
Can anybody give me a clue?
Thank you.

#15 - 11/30/2022 12:23 PM - Eric Faulhaber

Marian, can you shed any light on the RECURSIVE feature?

#16 - 11/30/2022 12:28 PM - Igor Skornyakov

- File ds-test.p added

Another problem with RECURSIVE.
I'm trying to write a test for FILL (see the attachment).

When I run it I get the following warning:

```
*** 11877 : FILL on a dataset buffer not allowed when query in data-source is for the same table tt. (11877)
```

What I'm doing wrong?
Thank you.

#17 - 11/30/2022 06:31 PM - Ovidiu Maxiniuc

Igor,

You cannot have the same table to be both the source and destination for a fill operation (we do not support this 11877 error, so please add it to ErrorManager).

The ds dataset definition looks right to me. But for dataset, please define a new table independent of the dataset (let's name it srctt). To make things simple, use the same structure from tt. Create the records in this new table. Then create a data-source for it. Attach this data-source to your tt and now fill it. Something like this:

```
DEFINE TEMP-TABLE srctt NO-UNDO
  FIELD cE  AS CHARACTER
  FIELD cM  AS CHARACTER
  FIELD val AS CHARACTER.

CREATE srctt. srctt.cE = 'm0'. srctt.val = '-m0'.
CREATE srctt. srctt.cE = 'm1'. srctt.cM = 'm0'. srctt.val = 'm0-m1'.
CREATE srctt. srctt.cE = 'm2'. srctt.cM = 'm0'. srctt.val = 'm0-m2'.
CREATE srctt. srctt.cE = 'e1'. srctt.cM = 'm1'. srctt.val = 'm1-e1'.
CREATE srctt. srctt.cE = 'e2'. srctt.cM = 'm2'. srctt.val = 'm2-22'.

DEFINE TEMP-TABLE tt NO-UNDO
  FIELD cE  AS CHARACTER
  FIELD cM  AS CHARACTER
  FIELD val AS CHARACTER
  INDEX idxE AS UNIQUE cE.

DEFINE DATASET ds FOR tt
  DATA-RELATION r1 FOR tt, tt
    RELATION-FIELDS (cE, cM) RECURSIVE.

DEFINE DATA-SOURCE srctt1 FOR srctt.
BUFFER tt:ATTACH-DATA-SOURCE(DATA-SOURCE srctt1:HANDLE).

...
```

#18 - 12/01/2022 01:00 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Igor,
>
> You cannot have the same table to be both the source and destination for a fill operation (we do not support this 11877 error, so please add it to ErrorManager).
>
> The ds dataset definition looks right to me. But for dataset, please define a new table independent of the dataset (let's name it srctt). To make things simple, use the same structure from tt. Create the records in this new table. Then create a data-source for it. Attach this data-source to your tt and now fill it. Something like this:
> [...]

Oh, I see. Thank you, Ovidiu!

#19 - 12/01/2022 02:54 AM - Igor Skornyakov

Found a number of incompatibilities in (READ|WRITE)-XML(SCHEMA)? support for RECURSIVE DATASET.
Fixing them in 6129b.

#20 - 12/01/2022 03:34 AM - Marian Edu

Eric Faulhaber wrote:

> Marian, can you shed any light on the RECURSIVE feature?

This is mostly used by the 'tree-like' .NET UI components, as far as 4GL is concerned the effect on the FILL operation is that the corresponding parent records of recursive relations are loaded automatically, like in this example the record for Dave's manager is also loaded in the dataset. If you wonder, the other way around it does not work - aka load all child records for a parent.

```
define temp-table ttEmployee
    field cEmployee as character
    field cManager  as character
    field iAge      as integer
    index idxEmployee as unique cEmployee.

define temp-table srcEmployee like ttEmployee.

create srcEmployee.
assign
    srcEmployee.cEmployee = 'Dave'
    srcEmployee.cManager  = 'Mustaine'.

create srcEmployee.
assign
    srcEmployee.cEmployee = 'Mustaine'.

create srcEmployee.
assign
    srcEmployee.cEmployee = 'Toto'
    srcEmployee.cManager  = 'Mustain'.
```

```
define dataset myRecurs for ttEmployee
    data-relation r1 for ttEmployee, ttEmployee
    relation-fields (cManager, cEmployee) recursive.

define query qEmp for srcEmployee.

define data-source dsEmployee for query qEmp srcEmployee.

buffer ttEmployee:attach-data-source(data-source dsEmployee:HANDLE).

query qEmp:query-prepare("for each srcEmployee where cEmployee = 'Dave'").
//query qEmp:query-prepare("for each srcEmployee where cManager = 'Mustain'").

dataset myRecurs:fill().

for each  ttEmployee:
    display ttEmployee with side-labels.
end.
```

#21 - 12/01/2022 04:09 AM - Igor Skornyakov

Marian Edu wrote:

> Eric Faulhaber wrote:
>
> > Marian, can you shed any light on the RECURSIVE feature?
>
> This is mostly used by the 'tree-like' .NET UI components, as far as 4GL is concerned the effect on the FILL operation is that the corresponding parent records of recursive relations are loaded automatically, like in this example the record for Dave's manager is also loaded in the dataset. If you wonder, the other way around it does not work - aka load all child records for a parent.
>
> [...]

Thanks a lot, Marian!
What about navigation& I mean the following statement:

```
Navigation is not supported, but can be done normally through .NET, Java, or a recursive ABL procedure.
```

How a recursive ABL procedure can be used for navigation?
Thank you.

#22 - 12/01/2022 04:27 AM - Igor Skornyakov

- File ds-test.p added

Marian Edu wrote:

> This is mostly used by the 'tree-like' .NET UI components, as far as 4GL is concerned the effect on the FILL operation is that the corresponding parent records of recursive relations are loaded automatically, like in this example the record for Dave's manager is also loaded in the dataset. If you wonder, the other way around it does not work - aka load all child records for a parent.

Marian,
In my test (attached) only one record is loaded, w/o its parent(s).
What I'm doing wrong?
Thank you.

#23 - 12/01/2022 04:34 AM - Marian Edu

Igor Skornyakov wrote:

> What I'm doing wrong?

By looking at the records created the 'cE' is the PK and 'cM' is the FK so the relation-fields should be `(cM, cE)` - aka, parent field first.

#24 - 12/01/2022 04:35 AM - Igor Skornyakov

Marian Edu wrote:

> Igor Skornyakov wrote:
>
> > What I'm doing wrong?
>
> By looking at the records created the 'cE' is the PK and 'cM' is the FK so the relation-fields should be `(cM, cE)` - aka, parent field first.

Yes! Now it works.
Thank you.

#25 - 12/01/2022 04:44 AM - Marian Edu

Igor Skornyakov wrote:

> What about navigation& I mean the following statement:
> [...]
> How a recursive ABL procedure can be used for navigation?

Since there is only one buffer for the recursive data relation there can't be any navigation - the buffer can only be on one particular record ar a time. .NET UI components can with with those recursive relations to show a tree-like (BOM) structure, if it must be done in 4GL a second buffer must be used and navigation must be implemented manually - as in when the query on dataset buffer changes the second buffer (FK) will lookup the corresponding record, in anyway this has to be done manually so there is nothing that needs to be implemented in this regard.

#26 - 12/01/2022 04:46 AM - Igor Skornyakov

Marian Edu wrote:

> Igor Skornyakov wrote:
>
> > What about navigation& I mean the following statement:
> > [...]
> > How a recursive ABL procedure can be used for navigation?
>
> Since there is only one buffer for the recursive data relation there can't be any navigation - the buffer can only be on one particular record ar a time. .NET UI components can with with those recursive relations to show a tree-like (BOM) structure, if it must be done in 4GL a second buffer must be used and navigation must be implemented manually - as in when the query on dataset buffer changes the second buffer (FK) will lookup the corresponding record, in anyway this has to be done manually so there is nothing that needs to be implemented in this regard.

Got it. Thanks a lot!

#27 - 12/01/2022 05:29 AM - Igor Skornyakov

- File ds.xml added

- File ttd.xml added

I've noticed an intersting thing. In the absense of the nsPrefix 4GL WRITE-XML creates "html-style" output. See attachment.

#28 - 12/01/2022 10:30 AM - Igor Skornyakov

Igor Skornyakov wrote:

> I've noticed an intersting thing. In the absense of the nsPrefix 4GL WRITE-XML creates "html-style" output. See attachment.

Sorry. Actually it is not that unusual. I was confused with the table name "ttd").
However, FWD in this case generates no row data at all.

#29 - 12/01/2022 10:46 AM - Igor Skornyakov

When PK and a FK are specified in an incorrect order in the relation-fields clause (see #6444-23), 4GL does not complain but produces a strange output for WRITE-XML(SCHEMA)? and generates an error on READ-XML.
Should I reproduce this quirk with FWD?
Thank you.

#30 - 12/02/2022 08:44 AM - Igor Skornyakov

Igor Skornyakov wrote:

> However, FWD in this case generates no row data at all.

It appears that this is the problem not with WRTIE-XML but with FILL.
Both with 3821c and 6129b after DATASET:FILL() the target table is empty in my test.
Working on this.

#31 - 12/05/2022 03:01 PM - Igor Skornyakov

It appears that DATASET:FILL is not working at all for RECURSIVE DATA-RELATION (the DataSet.getTopBufferList() returns empty list in this case).
If this is fixed the DATASET:FILL() still does not add parent records if they are missed in the DATA-SOURCE QUERY reuslt set.

Working on this.

#32 - 12/07/2022 10:45 AM - Igor Skornyakov

The BufferImpl.fill() method is pretty long (400 lines of code) and complicated.
I'm thinking about adding support for a RECURSIVE DATA-RELATION by re-writing the DATA-SOURCE query using H2 CTE and leaving the rest of the logic intact.

This looks possible if the query is simple enough - corresponds to a single SQL statement which can be used for a CTE definition.
Is it always the fact for a RECURSIVE DATA-RELATION?
Thank you.

#33 - 12/07/2022 11:56 AM - Igor Skornyakov

- File ds-test.p added

Igor Skornyakov wrote:

> I'm thinking about adding support for a RECURSIVE DATA-RELATION by re-writing the DATA-SOURCE query using H2 CTE and leaving the rest of the logic intact.
> This looks possible if the query is simple enough - corresponds to a single SQL statement which can be used for a CTE definition.
> Is it always the fact for a RECURSIVE DATA-RELATION?
> Thank you.

As an example of the what I mean consider the attached test. The re-written query for it is:

```
with recursive r(cE, cM, val) as (
   select cE, cM, val
   from tt where cE = 'e1'
   union all
   select rtt.cE, rtt.cM, rtt.val
   from tt rtt, r
   where rtt.cE = r.cM
)
select * from r
```

Please note that the original query is just a base part of the recusive CTE (before union all).

#34 - 12/07/2022 04:42 PM - Ovidiu Maxiniuc

Igor Skornyakov wrote:

> It appears that DATASET:FILL is not working at all for RECURSIVE DATA-RELATION (the DataSet.getTopBufferList() returns empty list in this case).

Since the relation is cyclic there are no top-level buffers so, logically this is correct. But the true question is: does 4GL return the same?

> If this is fixed the DATASET:FILL() still does not add parent records if they are missed in the DATA-SOURCE QUERY reuslt set.

1: What do you mean by fixed?
2: Assuming you forced the buffers to be top-level, does FILL does nothing because the query fails to iterate the data-source?

> The BufferImpl.fill() method is pretty long (400 lines of code) and complicated.

I know that the trend is to have smaller, very specialized methods. Unfortunately, this one contains a lot of logic. We could separate the validations (a lot of them) and notifications (they take about 1/2 of the lines) by keeping them in main method and extract the query iteration (the try starting with finalFillQuery.queryOpen();) into a new private method, fillImpl(...), but it will still be long: there are various FILL options (FILL-MODE, BATCH-SIZE, REPOSITION) to take into account.

> I'm thinking about adding support for a RECURSIVE DATA-RELATION by re-writing the DATA-SOURCE query using H2 CTE and leaving the rest of the logic intact.
> This looks possible if the query is simple enough - corresponds to a single SQL statement which can be used for a CTE definition.

I do not think that will be generically possible. We are not in control of the query. If the ABL programmer constructed the data-sources from buffers directly, this might be possible, but if a query is supplied, it could be anything. Note that, even if use TEMP-TABLEs for simplicity in our testcases, the source buffers are not mandatory to be from TEMP-TABLES. Additionally, they could have substitutions and we need to do this one-step at a time to handle child-buffer relations and notifications/triggers.

> Is it always the fact for a RECURSIVE DATA-RELATION?

Not sure what you mean. Maybe I give you a partial answer above.

#35 - 12/08/2022 01:31 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

Igor Skornyakov wrote:

> It appears that DATASET:FILL is not working at all for RECURSIVE DATA-RELATION (the DataSet.getTopBufferList() returns empty list in this case).

Since the relation is cyclic there are no top-level buffers so, logically this is correct. But the true question is: does 4GL return the same?

> If this is fixed the DATASET:FILL() still does not add parent records if they are missed in the DATA-SOURCE QUERY reuslt set.

1: What do you mean by fixed?

I mean the following:

```
=== modified file 'src/com/goldencode/p2j/persist/DataSet.java'
--- src/com/goldencode/p2j/persist/DataSet.java 2022-12-07 04:10:10 +0000
+++ src/com/goldencode/p2j/persist/DataSet.java 2022-12-07 08:25:53 +0000
@@ -6155,7 +6155,8 @@

       for (DataRelation relation : relations)
       {
-         if (relation._isActive() && !relation._isReposition())
+         if (relation._isActive() && !relation._isReposition() &&
+             !relation.isRecursive().booleanValue())
          {
             topBuffers.remove(relation.getChildBuffer().unwrapBuffer());
          }
```

2: Assuming you forced the buffers to be top-level, does FILL does nothing because the query fails to iterate the data-source?

The problem with RECURSIVE dataset is that the FILL method must add the parent records for all records in the query result se even if these parents are not in the result set. See #6444-33.

> I'm thinking about adding support for a RECURSIVE DATA-RELATION by re-writing the DATA-SOURCE query using H2 CTE and leaving the rest of the logic intact.
> This looks possible if the query is simple enough - corresponds to a single SQL statement which can be used for a CTE definition.

> I do not think that will be generically possible. We are not in control of the query. If the ABL programmer constructed the data-sources from buffers directly, this might be possible, but if a query is supplied, it could be anything. Note that, even if use TEMP-TABLEs for simplicity in our testcases, the source buffers are not mandatory to be from TEMP-TABLES. Additionally, they could have substitutions and we need to do this one-step at a time to handle child-buffer relations and notifications/triggers.

Oh, I see. Indeed, in this case we cannot be sure which version of the CTE syntax to use.
Thank you.

#36 - 12/08/2022 05:59 AM - Igor Skornyakov

Igor Skornyakov wrote:

> Oh, I see. Indeed, in this case we cannot be sure which version of the CTE syntax to use.

Actually the syntax for recursive CTEs is almost identical for H2, PostgreSQL and MariaDB.
Please note also that to implement the FILL logic for RECURSIVE DATE-RELATION (as I understand it) we need to make some assumptions about the corresponding DATA-SOURCE anyway.

Marian,
Is it correct that the DATA-SOURCE query should select a subset of a single table and this only table should be specified as a buffer?
Thank you.

#37 - 12/08/2022 03:02 PM - Ovidiu Maxiniuc

Igor Skornyakov wrote:

> Oh, I see. Indeed, in this case we cannot be sure which version of the CTE syntax to use.

> Actually the syntax for recursive CTEs is almost identical for H2, PostgreSQL and MariaDB.
> Please note also that to implement the FILL logic for RECURSIVE DATE-RELATION (as I understand it) we need to make some assumptions about the corresponding DATA-SOURCE anyway.

I do not think the dialect for a single database is the problem here, but the fact that the query may join tables from different databases. More than that, I understand that you aim for a 'fast-copy' mechanism where the rows are copied with a single SQL statement. If this is the case we will probably lose the ability to trigger the notification events for each record. How will MERGE/REPLACE be handled?

> Marian,
> Is it correct that the DATA-SOURCE query should select a subset of a single table and this only table should be specified as a buffer?

This is not the case. Here is an example:

```
define temp-table ttsrc field f1 as char field f2 as int.
create ttsrc. ttsrc.f1 = "isbn". ttsrc.f2 = 999.

define variable qh as handle.
create query qh.
qh:set-buffers(buffer book:handle, buffer ttsrc:handle).
qh:query-prepare("for each book, first ttsrc where ttsrc.f1 eq book.isbn").

define temp-table dstt field f2 as int field book-id as int.
define dataset ds1 for dstt.

define data-source dsrc for book, ttsrc.
data-source dsrc:query = qh.
buffer dstt:attach-data-source(data-source dsrc:handle).

dataset ds1:fill().
```

The qh query is defined over permanent table book and temporary ttsrc. After fill() execution you can see that the dstt is correctly populated.

#38 - 12/08/2022 03:27 PM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Igor Skornyakov wrote:
>
> > Oh, I see. Indeed, in this case we cannot be sure which version of the CTE syntax to use.
>
> Actually the syntax for recursive CTEs is almost identical for H2, PostgreSQL and MariaDB.
> Please note also that to implement the FILL logic for RECURSIVE DATE-RELATION (as I understand it) we need to make some assumptions about the corresponding DATA-SOURCE anyway.

I do not think the dialect for a single database is the problem here, but the fact that the query may join tables from different databases. More than that, I understand that you aim for a 'fast-copy' mechanism where the rows are copied with a single SQL statement. If this is the case we will probably lose the ability to trigger the notification events for each record. How will MERGE/REPLACE be handled?

Sorry, I do not understand. What I suggest is a fast processing of the **source** data which should be different from the target one. I have not found any events for the table which is only read (except OFF-END which we do not process anyway)

> Marian,
> Is it correct that the DATA-SOURCE query should select a subset of a single table and this only table should be specified as a buffer?

This is not the case. Here is an example:
[...]

The qh query is defined over permanent table book and temporary ttsrc. After fill() execution you can see that the dstt is correctly populated.

Please note that I'm asking only about **RECURSIVE** DATA-RELATION. I understand that it is a very special situation. As I can see your example is not about RECURSIVE data-relation.

#39 - 12/08/2022 03:45 PM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> More than that, I understand that you aim for a 'fast-copy' mechanism where the rows are copied with a single SQL statement. If this is the case we will probably lose the ability to trigger the notification events for each record. How will MERGE/REPLACE be handled?

It looks that I was not clear enough, sorry. What I suggest is to re-write the DATA-SOURCE QUERY using RECURSIVE CTE like described in #6444-33 . The rest of the BufferImpl.fill() is not affected (including iteration over the query result set and callbacks invocation). And this will be done only if the DATA-RELATION is RECURSIVE.

#40 - 12/09/2022 04:46 AM - Marian Edu

Ovidiu Maxiniuc wrote:

> > Marian,
> > Is it correct that the DATA-SOURCE query should select a subset of a single table and this only table should be specified as a buffer?
>
> This is not the case. Here is an example:

As Ovidiu said the query of any temp-table's data source can be on multiple tables (and it often is for de-normalization), I've slightly extended the test for this in data_relation/recursive_relation_fill.p inside testcases project.

Please not that while the recursive relation does load the FK parent in the parent temp-table (same table as it's recursive) the child records for those records are not loaded - aka, in the example the 'manager' records is loaded in employee but his benefits aren't.

#41 - 12/09/2022 04:51 AM - Igor Skornyakov

Marian Edu wrote:

> Ovidiu Maxiniuc wrote:
>
> > > Marian,
> > > Is it correct that the DATA-SOURCE query should select a subset of a single table and this only table should be specified as a buffer?

> This is not the case. Here is an example:

> As Ovidiu said the query of any temp-table's data source can be on multiple tables (and it often is for de-normalization), I've slightly extended the test for this in data_relation/recursive_relation_fill.p inside testcases project.
>
> Please not that while the recursive relation does load the FK parent in the parent temp-table (same table as it's recursive) the child records for those records are not loaded - aka, in the example the 'manager' records is loaded in employee but his benefits aren't.

Got it. Thank you, Marian!

#42 - 12/09/2022 05:32 AM - Igor Skornyakov

The data_relation/recursive_relation_fill.p test fails with FWD with Cannot run GET methods on query qEmp until it is opened. (7313).
Working in this.

#43 - 12/09/2022 07:45 AM - Igor Skornyakov

Marian Edu wrote:

> Please not that while the recursive relation does load the FK parent in the parent temp-table (same table as it's recursive) the child records for those records are not loaded - aka, in the example the 'manager' records is loaded in employee but his benefits aren't.

What puzzles me is the logic for 'relaxing' the qEmp conditions to find the parent record for 'Dave'.

Marian,
Do you have any ideas?
Thank you.

#44 - 12/09/2022 08:19 AM - Marian Edu

Igor Skornyakov wrote:

> What puzzles me is the logic for 'relaxing' the qEmp conditions to find the parent record for 'Dave'.

The 'relaxation' of filter conditions on the query doesn't apply only to the FK field(s), this will actually returns the parent record albeit the 'hide' flag is

set to true on this one:

```
define temp-table ttEmployee
    field cEmployee as character
    field cManager  as character
    field cAddr     as character
    index idxEmployee as unique cEmployee.

define temp-table ttBenefits
    field cEmployee as character
    field cBenefits as character
    index idxEmp cEmployee.

define temp-table ttAddr
    field iAddr as int
    field cAddr as character
    index idxAddr as unique iAddr.

define temp-table srcEmployee
    field cEmployee as character
    field cManager  as character
    field iAddr     as integer
    field lHide     as logical.

define temp-table srcBenefits like ttBenefits.

create ttAddr.
assign
    ttAddr.iAddr = 1
    ttAddr.cAddr = 'Home'.

create ttAddr.
assign
    ttAddr.iAddr = 2
    ttAddr.cAddr = 'Outerspace'.

create srcEmployee.
assign
    srcEmployee.cEmployee = 'Dave'
    srcEmployee.cManager  = 'Mustaine'
    srcEmployee.iAddr     = 1.

create srcEmployee.
assign
    srcEmployee.cEmployee = 'Mustaine'
    srcEmployee.iAddr     = 2
    srcEmployee.lHide     = true.

create srcEmployee.
assign
    srcEmployee.cEmployee = 'Toto'
    srcEmployee.cManager  = 'Mustain'
    srcEmployee.iAddr     = 1.

create srcBenefits.
assign
    srcBenefits.cEmployee = 'Dave'
    srcBenefits.cBenefits = 'Voice training'.

create srcBenefits.
assign
    srcBenefits.cEmployee = 'Dave'
    srcBenefits.cBenefits = 'Groupies'.

create srcBenefits.
assign
    srcBenefits.cEmployee = 'Mustain'
    srcBenefits.cBenefits = 'Free beer'.

define dataset myRecurs for ttEmployee, ttBenefits
    data-relation r1 for ttEmployee, ttEmployee
    relation-fields (cManager, cEmployee) recursive
    data-relation r2 for ttEmployee, ttBenefits
    relation-fields(cEmployee, cEmployee).
```

```
define query qEmp for srcEmployee, ttAddr.
define query qBen for srcBenefits.

define data-source dsEmployee for query qEmp srcEmployee, ttAddr.
define data-source dsBen for query qBen srcBenefits.

buffer ttEmployee:attach-data-source(data-source dsEmployee:HANDLE).
buffer ttBenefits:attach-data-source(data-source dsBen:HANDLE).

query qEmp:query-prepare("for each srcEmployee where cEmployee = 'Dave' and lHide = false, each ttAddr where t
tAddr.iAddr = srcEmployee.iAddr").
//query qEmp:query-prepare("for each srcEmployee where cManager = 'Mustain'").

query qBen:query-prepare("for each srcBenefits").

dataset myRecurs:fill().

for each  ttEmployee:
    display ttEmployee with side-labels.
    for each ttBenefits where ttBenefits.cEmployee = ttEmployee.cEmployee:
        display ttBenefits with side-labels.
    end.
end.
```

IMHO the same query is used for retrieving the recursive records, aka the join on ttAddr remains and address name is correctly fetched, but the whole where filter on the first buffer is replaced with only the filter on relation-fields. There is something I've meant to test out which is pagination but I expect the 'additional' records to be added on top of the requested batch size... remains to see if this is confirmed or not.

#45 - 12/09/2022 08:39 AM - Igor Skornyakov

Marian Edu wrote:

> Igor Skornyakov wrote:
>
> > What puzzles me is the logic for 'relaxing' the qEmp conditions to find the parent record for 'Dave'.
>
> The 'relaxation' of filter conditions on the query doesn't apply only to the FK field(s), this will actually returns the parent record albeit the 'hide' flag is set to true on this one:
>
> [...]

> IMHO the same query is used for retrieving the recursive records, aka the join on ttAddr remains and address name is correctly fetched, but the whole where filter on the first buffer is replaced with only the filter on relation-fields. There is something I've meant to test out which is pagination but I expect the 'additional' records to be added on top of the requested batch size... remains to see if this is confirmed or not.

Thank you, Marian.
I'm thinking now on the tests to figure out what is the exact 'relaxation' logic. For example in case when relation-fields of the source for recursive relation are taken form are taken from different source tables.

BTW: it looks that FILL for recursive dataset can be very expensive since it can require multiple passes. Another option is to use an auxiliary temp table holding the result set of the 'relaxed' query and a recursive CTE against it (trading space for speed). In this case multiple passes will be done indirectly by H2 engine.

#46 - 12/12/2022 02:29 PM - Igor Skornyakov

Based on my tests the FILL for recursive DATA-RELATION add records which are not in the DATA-SOURCE QUERY only if both RELATION-FIELDS come from the same table.
In this case the parent records are taken from the result set of the query which differs from the original one only with respect to the table containing RELATION-FIELDS - all WHERE predicates for this table are removed. Other components of the 'relaxed' query are the same.

Working on the implementation of this logic.

Please let me know what you think about my suggestion regarding optimization in #6444-45.
Thank you.

#47 - 12/13/2022 06:48 PM - Ovidiu Maxiniuc

Igor Skornyakov wrote:

> BTW: it looks that FILL for recursive dataset can be very expensive since it can require multiple passes. Another option is to use an auxiliary temp table holding the result set of the 'relaxed' query and a recursive CTE against it (trading space for speed). In this case multiple passes will be done indirectly by H2 engine.

The FILL operation is QUERY driven. And that happens for each buffer affected (that is, after a record is inserted in the parent buffer of the relation, a FILL will be performed on all its children). In recursion, for all buffers in the subtree of each top-level buffer. In other words, for each row written in a parent buffer, the queries of all its children are reopened and repositioned for next batch and a new set of records are fetched in the FILL method of each child. At least this is in case of non recursive data-sets.

When a full dataset is populated bu FILL, it starts with the top-level buffers and the procedure propagate to all buffers following the relation trees.

There is also something else which crossed my mind and has to be tested: how does this work with indirect recursion (circular but not direct/self-reference in parent-child relations)?

What you say looks like a record caching of the query backed by the new _auxiliary temp_table within the H2 SQL server. Even if it will not work for all cases, it it is able to detect when the specific conditions, it might be a local boost of performance and we crave for it. Note however, that the usual FILL scenario will take the source data from a permanent table. We built examples with temp-table (H2) data-sources for sake of convenience, but in real application, the source data is extracted from a persistent database (PSQL/MariaDb/MSSQL and sometimes on developer's machine - H2).

Maybe I did not understand it correctly. I would like to see some more details on this issue, maybe an example of a best-case scenario where your suggestion has maximum impact on performance. I assume you have a testcase you (intent to) work on.

#48 - 12/14/2022 02:31 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> The FILL operation is QUERY driven. And that happens for each buffer affected (that is, after a record is inserted in the parent buffer of the relation, a FILL will be performed on all its children). In recursion, for all buffers in the subtree of each top-level buffer. In other words, for each row written in a parent buffer, the queries of all its children are reopened and repositioned for next batch and a new set of records are fetched in the FILL method of each child. At least this is in case of non recursive data-sets.
>
> When a full dataset is populated bu FILL, it starts with the top-level buffers and the procedure propagate to all buffers following the relation trees.
>
> There is also something else which crossed my mind and has to be tested: how does this work with indirect recursion (circular but not direct/self-reference in parent-child relations)?

The problem with recursive DATE-RELATION is that we need to add not only records from the QUERY result set but also their parents. In Marian's test (see #6444-44) the record with srcEmployee.cEmployee = 'Mustaine' will be also added.

> What you say looks like a record caching of the query backed by the new _auxiliary temp_table within the H2 SQL server. Even if it will not work for all cases, it it is able to detect when the specific conditions, it might be a local boost of performance and we crave for it. Note however, that the usual FILL scenario will take the source data from a permanent table. We built examples with temp-table (H2) data-sources for sake of convenience, but in real application, the source data is extracted from a persistent database (PSQL/MariaDb/MSSQL and sometimes on developer's machine - H2).
>
> Maybe I did not understand it correctly. I would like to see some more details on this issue, maybe an example of a best-case scenario where your suggestion has maximum impact on performance. I assume you have a testcase you (intent to) work on.

Well, imagine that in Marian's test the srcEmployee table contains not just a two-level relation cManager -> cEmployee but a deep tree (or multiple trees) and the DATA-SOURCE query selects only some leaves. In this case, we will have to change the BufferImpl.fill() logic to recursively add the parents of already selected nodes. This cannot be done in one pass unless we cache the result set of the 'relaxed' QUERY (see #6444-46) and use a recursive CTE (#6444-37). We will also need to make sure that we do not select the same record twice which is guaranteed by UNION ALL clause of the recursive CTE if we use a cached result set. The deeper is the tree the more iterations we will need to make.

#49 - 12/14/2022 02:07 PM - Igor Skornyakov

I've realized that we do not really need an auxiliary temporary table to use recursive CTE instead of iteration (see previous note).
Indeed, as described in #6444-46 the additional records are added on FILL only if both RELATION-FIELDS come from the same table. This means that the 'relaxed' query can be constructed by removing all WHERE predicates for this table and replacing this table with a SELECT from the recursive CTE. As mentioned in #6444-36 the syntax for such a CTE is almost identical for H2, PostgreSQL, and MariaDB. In fact, it is also the same for MS SQL Server.

#50 - 12/16/2022 03:48 AM - Igor Skornyakov

I need to iterate over a DATA-SOURCE twice - first with the original QUERY and then with the 'relaxed' one. What is the right way to reset the DATA-SOURCE to the 'initial' state before the second iteration?
Thank you.

#51 - 12/16/2022 02:59 PM - Igor Skornyakov

Something is wrong in the BufferImpl.fill() logic regarding child relations. In the data_relation/recursive_relation_fill.p the r1 relation appears to be its own child which causes error 7131 (see #6444-42).

#52 - 12/19/2022 10:12 AM - Ovidiu Maxiniuc

Igor Skornyakov wrote:

> I need to iterate over a DATA-SOURCE twice - first with the original QUERY and then with the 'relaxed' one. What is the right way to reset the DATA-SOURCE to the 'initial' state before the second iteration?

I will start with the question first: I do not know for sure. I guess you need to close and reopen the query which drives the fill operation or at least calling first/get-first on it.
I am not very happy with this double iteration. It implies a performance penalty. Ultimately, if this is really needed, we must identify the minimum set of cases when we cannot live without it and implement this query reset only for those cases.

> Something is wrong in the BufferImpl.fill() logic regarding child relations. In the data_relation/recursive_relation_fill.p the r1 relation appears to be its own child which causes error 7131 (see #6444-42).

Generally speaking, in direct recursive relations, yes, the parent an the child are the same. That makes the relation (direct) recursive. It is not clear to me yet if indirect recursive relation graphs are possible - when multiple table are used to form the recursion loop. It's normal that current implementation to detect this situation and report is as abnormal since now FWD supports only non-recursive relations, so it identifies this case as an unwanted exception.
The error 7313 is caused because fill-query failed to be constructed or opened. I assume it is generated on finalFillQuery.first() from BufferImpl.fill() because the result of finalFillQuery.queryOpen(); from previous line is not checked.

#53 - 12/19/2022 10:28 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> I am not very happy with this double iteration. It implies a performance penalty. Ultimately, if this is really needed, we must identify the minimum set of cases when we cannot live without it and implement this query reset only for those cases.

Unfortunately, we cannot avoid the second iteration for RECURSIVE DATA-RELATION because of the semantics of FILL for such relations. In #6444-46 I describe the situations when the second pass is not required. I think that we also can avoid it if the DATA-SOURCE QUERY uses a single table but at this moment some details are still to be designed.

> Generally speaking, in direct recursive relations, yes, the parent an the child are the same. That makes the relation (direct) recursive. It is not clear to me yet if indirect recursive relation graphs are possible - when multiple table are used to form the recursion loop. It's normal that current implementation to detect this situation and report is as abnormal since now FWD supports only non-recursive relations, so it identifies this case as an unwanted exception.
> The error 7313 is caused because fill-query failed to be constructed or opened. I assume it is generated on finalFillQuery.first() from BufferImpl.fill() because the result of finalFillQuery.queryOpen(); from previous line is not checked.

Thank you, Ovidiu. I think that I'm close to a fix at the moment.

#54 - 12/23/2022 09:43 AM - Igor Skornyakov

In the relation model, the physical order or the records in the table is irrelevant and the order of records in the result set of the SELECT statement w/o ORDER BY clause is unspecified. I'm unsure if this is the fact for the OE tables/queries.
Should we care about the order in which records are added to the target tables during FILL?
Thank you.

#55 - 12/25/2022 04:20 AM - Igor Skornyakov

Igor Skornyakov wrote:

> I think that we also can avoid it if the DATA-SOURCE QUERY uses a single table but at this moment some details are still to be designed.

Actually, we can avoid the second pass only if the DATA-SOURCE QUERY uses a single table **and** there is only one DATA-SOURCE. This is because e.g. in #6444-44 the ttBenefits records are not added for the records from the relaxed qEmp query result set.

#56 - 12/28/2022 02:56 AM - Igor Skornyakov

Working on the second pass query for the recursive data-relation FILL support I have to deal with the situation when the query should be converted to SQL in two different ways.
This appears to be tricky because we need to propagate the conversion context through auxiliary data structures such as QueryWrapper -> CompoundQuery -> CompoundComponent -> PreselectQuery -> Query.
An additional problem is the Persistence.staticQueryCache which should be invalidated after conversion with the non-default conversion context.
This would be much easier if a 'holder' class was used instead of just the query string. This holder can be easily changed to hold the conversion context.
I've faced a similar (but more simple) problem with mutable SESSION attributes as query parameters.
I understand that refactoring from using a 'plain' query string (hql) to a query string holder is a massive (albeit straightforward) change, but maybe it makes sense to consider it.
What do you think?
Thank you.

#57 - 12/28/2022 05:30 AM - Igor Skornyakov

At this moment it looks that FqlToSqlConverter.generateUniqueSqlColumnNames is always true.
Can I rely on this?
Thank you.

#58 - 12/28/2022 01:37 PM - Eric Faulhaber

Igor Skornyakov wrote:

> At this moment it looks that FqlToSqlConverter.generateUniqueSqlColumnNames is always true.
> Can I rely on this?

I don't know what the intent of this variable was, maybe something that is only changed for debugging? Ovidiu can answer definitively when he returns from PTO in early January, but it seems we would always want unique SQL column names, so I would think it would be safe to rely upon this.

#59 - 12/28/2022 01:38 PM - Igor Skornyakov

Eric Faulhaber wrote:

> I don't know what the intent of this variable was, maybe something that is only changed for debugging? Ovidiu can answer definitively when he returns from PTO in early January, but it seems we would always want unique SQL column names, so I would think it would be safe to rely upon this.

Thank you!

#60 - 12/28/2022 01:40 PM - Eric Faulhaber

Igor Skornyakov wrote:

> [...]
> I understand that refactoring from using a 'plain' query string (hql) to a query string holder is a massive (albeit straightforward) change, but maybe it makes sense to consider it.
> What do you think?

I think we should wait for Ovidiu's input when he returns before undertaking any massive changes to the implementation.

#61 - 12/28/2022 01:43 PM - Eric Faulhaber

Eric Faulhaber wrote:

> Igor Skornyakov wrote:
>
> > [...]
> > I understand that refactoring from using a 'plain' query string (hql) to a query string holder is a massive (albeit straightforward) change, but maybe it makes sense to consider it.
> > What do you think?
>
> I think we should wait for Ovidiu's input when he returns before undertaking any massive changes to the implementation.

If this blocks you from making headway on the recursive FILL operation, please work on another item that needs attention from #6444-4 until we can get his feedback on this point. Thanks.

#62 - 12/28/2022 01:51 PM - Igor Skornyakov

Eric Faulhaber wrote:

> I think we should wait for Ovidiu's input when he returns before undertaking any massive changes to the implementation.

I've already resolved the problem for this particular task and do not suggest making these changes right now. I just decided to mention the problem

which we can experience in the future since it has a well-defined and general nature.

#63 - 12/28/2022 01:59 PM - Igor Skornyakov

> If this blocks you from making headway on the recursive FILL operation, please work on another item that needs attention from #6444-4 until we can get his feedback on this point. No, No, as mentioned in #6444-62 I have a solution for this task.

I planned to finish the task today but realized that the FILL query can use CONTAINS operator converted to the word tables queries and generation of corresponding CTEs. This complicates the query re-writing. I think that I have a solution now, but it requires more testing.

#64 - 12/30/2022 01:34 PM - Ovidiu Maxiniuc

Igor Skornyakov wrote:

> I understand that refactoring from using a 'plain' query string (hql) to a query string holder is a massive (albeit straightforward) change, but maybe it makes sense to consider it. What do you think?

Basically I am not against of such change. Carrying additional (meta) information with the FQL should be beneficial. Grouping the FQL with these 'annotation' and propagate them from FqlPreprocessor down to FqlToSqlConverter may give the latter some hints with its work. I am interesting in how these changes looks like, regardless how apparently massive is the refactoring (I assume the changes of signatures of some methods from .., String fql, ... to using the new wrapper like .., Fql fql, ... The Fql class will have a String fql member which would be current information).

Igor Skornyakov wrote:

> Should we care about the order in which records are added to the target tables during FILL?

I think we should. They are added in the order specified by the query which drives the FILL operation. Even if the BY clause is absent, 4GL will use the default index of the tables, or in absence of that the rowid s to order the rows. We attempt to replicate this by enforcing all indices to be unique in order to have a predictible fetching order for the records. More than that, the ABL programmer expects the BEFORE-ROW-FILL and AFTER-ROW-FILL events to be fired in a very specific order, not only within a specific table, but the order of these are imposed by the data-relations between the tables.

Igor Skornyakov wrote:

> At this moment it looks that FqlToSqlConverter.generateUniqueSqlColumnNames is always true. Can I rely on this?

The idea behind this flag was that, if in a complex query two subselects are not directly related (maybe on same table), they can use the overlapped column names without ambiguity. This would reduce the size of the statement overall. However, in practice, this is difficult to detect and we force all columns to have a unique alias even if no name collision will occur for a specific query.
The answer is positive, all columns will have a unique alias.

#65 - 12/30/2022 01:59 PM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> I understand that refactoring from using a 'plain' query string (hql) to a query string holder is a massive (albeit straightforward) change, but maybe it makes sense to consider it. What do you think?

Basically I am not against of such change. Carrying additional (meta) information with the FQL should be beneficial. Grouping the FQL with these 'annotation' and propagate them from FqlPreprocessor down to FqlToSqlConverter may give the latter some hints with its work. I am interesting in how these changes looks like, regardless how apparently massive is the refactoring (I assume the changes of signatures of some methods from .., String fql, ... to using the new wrapper like .., Fql fql, ... The Fql class will have a String fql member which would be current information).

Yes, I see the refactoring exactly like this. It can be pretty massive but straightforward. And the IDE will help to see the places to be changed. However, as I wrote before I do not think that we have to do it right now.

> Should we care about the order in which records are added to the target tables during FILL?

I think we should. They are added in the order specified by the query which drives the FILL operation. Even if the BY clause is absent, 4GL will use the default index of the tables, or in absence of that the rowid s to order the rows. We attempt to replicate this by enforcing all indices to be unique in order to have a predictible fetching order for the records. More than that, the ABL programmer expects the BEFORE-ROW-FILL and AFTER-ROW-FILL events to be fired in a very specific order, not only within a specific table, but the order of these are imposed by the data-relations between the tables.

Well, this will make FILL **very** expensive for recursive data relation. As I can see from my tests for such relations 4GL adds missed parent records right after adding the child record from the data source query result set. So we'll have to look up the source table after processing every record in the data source (and do it recursively).
Do we really want to implement the same order of adding records as 4GL? As I understand this can be visible only if the target table(s) have no indexes at all and we iterate over this table without explicit ordering. I can be wrong but it seems to me that one should be an extremely brave person to rely on a specific order of records in the FILL target table in this situation.

> Igor Skornyakov wrote:
>
> > At this moment it looks that FqlToSqlConverter.generateUniqueSqlColumnNames is always true. Can I rely on this?
>
> The idea behind this flag was that, if in a complex query two subselects are not directly related (maybe on same table), they can use the overlapped column names without ambiguity. This would reduce the size of the statement overall. However, in practice, this is difficult to detect and we force all columns to have a unique alias even if no name collision will occur for a specific query.
> The answer is positive, all columns will have a unique alias.

Thank you!

#66 - 12/30/2022 02:11 PM - Ovidiu Maxiniuc

Igor Skornyakov wrote:

> > Should we care about the order in which records are added to the target tables during FILL?
>
> I think we should. [...]

> Well, this will make FILL **very** expensive for recursive data relation. As I can see from my tests for such relations 4GL adds missed parent records right after adding the child record from the data source query result set. So we'll have to look up the source table after processing every record in the data source (and do it recursively).

I understand that. Can we isolate this damage to recursive relations only? These cases are not too frequent so the overall impact might not be really visible, but the performance in normal cases should not be affected. We may improve this local issue later, if we have new ideas.

> Do we really want to implement the same order of adding records as 4GL? As I understand this can be visible only if the target table(s) have no indexes at all and we iterate over this table without explicit ordering. I can be wrong but it seems to me that one should be an extremely brave person to rely on a specific order of records in the FILL target table in this situation.

Unfortunately, this is what we struggle to achieve, full binary compatibility with 4GL so that it would be transparent for end user if the application runs on OE or Java.
Greg, please correct me if I am overly aggressive on this.

#67 - 12/30/2022 02:35 PM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> I understand that. Can we isolate this damage to recursive relations only? These cases are not too frequent so the overall impact might not be really visible, but the performance in normal cases should not be affected. We may improve this local issue later, if we have new ideas.

Sure. I suggest not caring about the order of added records for recursive relations only. My current implementation does not affect non-recursive relations at all - it is just an additional step for recursive ones.

> Do we really want to implement the same order of adding records as 4GL? As I understand this can be visible only if the target table(s) have no indexes at all and we iterate over this table without explicit ordering. I can be wrong but it seems to me that one should be an extremely brave person to rely on a specific order of records in the FILL target table in this situation.

> Unfortunately, this is what we struggle to achieve, full binary compatibility with 4GL so that it would be transparent for end user if the application runs on OE or Java.

The question is what *binary compatibility* actually means. My point regarding the ordering in a situation in questions is that the incompatibility is not *observable* in a quantum mechanics parlance.

#68 - 12/30/2022 03:19 PM - Ovidiu Maxiniuc

Well, the *binary* might not have been the best attribute here to describe the compatibility.

I understand that, ultimately, the content of the databases (ABL and FWD) are identically at the end of the process, but this process is not atomic. There are events for each record created/inserted in destination table and my concern is that they might not match 4GL's order. I must (re-)read the ref-manual for the FILL and related events (https://docs.progress.com/bundle/openedge-prodatasets-117/page/Advanced-Events-and-Attributes.html).

Nevertheless, I like your analogy with Schrödinger's □□. □□

#69 - 12/30/2022 05:07 PM - Greg Shah

> Greg, please correct me if I am overly aggressive on this.

If the application code can be implemented to rely upon a consistent behavior in the 4GL, then we should implement it the same way. The only exception is when the behavior cannot be legitimately considered useful. For example, there are some cases in browse where the behavior as implemented in the 4GL appears to be a bug and from a user perspective it is actually bad behavior. In such cases, we have created special "quirk" tasks to document the "bug" or "feature" and then we defer the implementation.

In this case, I suspect we should implement the application-visible behavior.

#70 - 12/31/2022 02:15 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> I understand that, ultimately, the content of the databases (ABL and FWD) are identically at the end of the process, but this process is not atomic. There are events for each record created/inserted in destination table and my concern is that they might not match 4GL's order. I must (re-)read the ref-manual for the FILL and related events (https://docs.progress.com/bundle/openedge-prodatasets-117/page/Advanced-Events-and-Attributes.html).

The problem is that in a relational database the "natural" order of the record in a table is not defined (an order of records in the SELECT w/o ORDER BY clause is undefined). I have an impression that in 4GL the situation is the same if there the table has no indexes, but I can be wrong. The order of events on the FILL is a different story. This can be a problem if the event handlers are stateful, but I cannot imagine a scenario when it can make sense.

#71 - 12/31/2022 02:17 AM - Igor Skornyakov

Greg Shah wrote:

> In this case, I suspect we should implement the application-visible behavior.

I see. In this case, I have to completely change my approach since my main concern was performance.

#72 - 12/31/2022 02:56 AM - Eric Faulhaber

Greg Shah wrote:

> Greg, please correct me if I am overly aggressive on this.
>
> If the application code can be implemented to rely upon a consistent behavior in the 4GL, then we should implement it the same way. The only exception is when the behavior cannot be legitimately considered useful. For example, there are some cases in browse where the behavior as implemented in the 4GL appears to be a bug and from a user perspective it is actually bad behavior. In such cases, we have created special "quirk" tasks to document the "bug" or "feature" and then we defer the implementation.
>
> In this case, I suspect we should implement the application-visible behavior.

If I understand the issue correctly, I respectfully disagree. In the event there is no explicit sort order defined for a 4GL query using a BY clause, NOR is there an implicit sort order which can be relied upon, based on the index selected for a query, we already have the issue in non-dataset query handling that the sort order for that query is undefined; it very well may not match the legacy order, which is determined by internal implementation details (generally the ROWID, I believe, which often is sequentially in sync with record creation order, but not always). The best we can do in this case is to sort by primary key, but this may not always match the legacy behavior exactly.

If we are talking about the same type of issue here, I don't think we can guarantee the order of the records, if that order was never defined properly by

the legacy logic. And I don't think we should go to great lengths (and take unreasonable performance hits) to try. We **should**, however, always match the legacy order, if we have enough information (i.e., explicit BY clauses or implicit order based on selected query index) to do so.

I think most customers, given the choice of not matching undefined sort behavior which was never specified properly in legacy code, and having abysmal performance in an attempt to match this undefined behavior, will gladly change their legacy code to define some explicit sorting which can be relied upon consistently. If I have misunderstood the issue in this case, please let me know.

#73 - 12/31/2022 03:29 AM - Igor Skornyakov

Eric Faulhaber wrote:

> If I understand the issue correctly, I respectfully disagree. In the event there is no explicit sort order defined for a 4GL query using a BY clause, NOR is there an implicit sort order which can be relied upon, based on the index selected for a query, we already have the issue in non-dataset query handling that the sort order for that query is undefined; it very well may not match the legacy order, which is determined by internal implementation details (generally the ROWID, I believe, which often is sequentially in sync with record creation order, but not always). The best we can do in this case is to sort by primary key, but this may not always match the legacy behavior exactly.
>
> If we are talking about the same type of issue here, I don't think we can guarantee the order of the records, if that order was never defined properly by the legacy logic. And I don't think we should go to great lengths (and take unreasonable performance hits) to try. We **should**, however, always match the legacy order, if we have enough information (i.e., explicit BY clauses or implicit order based on selected query index) to do so.
>
> I think most customers, given the choice of not matching undefined sort behavior which was never specified properly in legacy code, and having abysmal performance in an attempt to match this undefined behavior, will gladly change their legacy code to define some explicit sorting which can be relied upon consistently. If I have misunderstood the issue in this case, please let me know.

I think that the situation is a little more delicate. Putting aside the problem with stateful event handlers for the target table the order of adding records (even if there is an implicit or explicit order for the data source query) is not 'observable' to the application since it should use query to retrieve the data when the order is defined either by BY clause of a target table indexes. This is why I think that we can ignore the order of records in the data source query.
I'm not sure how common is the practice of using stateful event handlers in 4GL applications. I can hardly imagine what such handlers can be used for.

#74 - 01/03/2023 11:09 AM - Igor Skornyakov

I've changed my test so that data sources used for FILL get data from the permanent tables.
However, at runtime, I see that temp tables are still used. The corresponding DMO classes do not exist in the project (I guess that they were generated dynamically). However, I do not see where these temp tables were created and populated.

This looks strange.

Is my understanding correct and what is the reason for using temp tables here?

Thank you.

#75 - 01/03/2023 11:33 AM - Igor Skornyakov

Igor Skornyakov wrote:

> I've changed my test so that data sources used for FILL get data from the permanent tables.
> However, at runtime, I see that temp tables are still used. The corresponding DMO classes do not exist in the project (I guess that they were generated dynamically). However, I do not see where these temp tables were created and populated.
>
> This looks strange.
>
> Is my understanding correct and what is the reason for using temp tables here?
>
> Thank you.

Sorry,
It was my fault. Please disregard.
Thank you.

#76 - 01/04/2023 10:19 AM - Igor Skornyakov

I have a strange situation.
The second pass query provides a correct result set for PostgreSQL and MariaDB, but returns nothing with H2 (both for temporary and permanent tables). The same query executed against the same H2 database with H2 Console with the same JDBC driver (fwd-h2-1.4.200-6.jar) returns a correct result (for the permanent table).
The query is the following:

```
with recursive r2(id0_, c1_0_, c2_0_, i3_0_, i4_0_, l5_0_) as (
with base3 as (

select
    srcemploye0_.recid as id0_, srcemploye0_.c_employee as c1_0_, srcemploye0_.c_manager as c2_0_, srcemploye0
_.i_manager as i3_0_, srcemploye0_.i_addr as i4_0_, srcemploye0_.l_hide as l5_0_
from
    src_employee srcemploye0_
where
    upper(rtrim(srcemploye0_.c_employee)) = 'DAVE' and srcemploye0_.l_hide = false
)
select * from base3
union all

select
    srcemploye0_.recid as id0_, srcemploye0_.c_employee as c1_0_, srcemploye0_.c_manager as c2_0_, srcemploye0
_.i_manager as i3_0_, srcemploye0_.i_addr as i4_0_, srcemploye0_.l_hide as l5_0_
from
    src_employee srcemploye0_
join r2 on (
srcemploye0_.c_employee = r2.c2_0_
```

```
)
)
select * from r2
where id0_ not in (
select recid

from
    src_employee srcemploye0_
where
    upper(rtrim(srcemploye0_.c_employee)) = 'DAVE' and srcemploye0_.l_hide = false
)
```

In fact, it is the same for all 3 dialects. For temp tables, the only difference is that 'special' fields (_multiplex, _errorFlag, ...) are added to the fields' lists, and a condition for the _multiplex is added to the where clause
Any suggestions?
Thank you.

#77 - 01/04/2023 02:35 PM - Ovidiu Maxiniuc

Igor Skornyakov wrote:

> I have a strange situation.
> In fact, it is the same for all 3 dialects. For temp tables, the only difference is that 'special' fields (_multiplex, _errorFlag, ...) are added to the fields' lists, and a condition for the _multiplex is added to the where clause
> Any suggestions?

From the PoV of the FILL operation, these 'special' fields could be ignored, with two exceptions:

- the _errorFlag, or other similar field is actually used in the query predicate or, more exotically, in source field pairs. This is a thing I come up after reading your note and it is highly improbably that a ABL programmer will use that to fill a dataset table;
- the _multiplex must always be present in all temp-table selects.

The temp-table - friendly query above should look like this:

```
with recursive r2(id0_, c1_0_, c2_0_, i3_0_, i4_0_, l5_0_) as (
   with base3 as (
      select srcemploye0_.recid as id0_, srcemploye0_.c_employee as c1_0_, srcemploye0_.c_manager as c2_0_,
             srcemploye0_.i_manager as i3_0_, srcemploye0_.i_addr as i4_0_, srcemploye0_.l_hide as l5_0_
      from   src_employee srcemploye0_
      where  srcemploye0_._multiplex = ? and
             (upper(rtrim(srcemploye0_.c_employee)) = 'DAVE' and srcemploye0_.l_hide = false)
   )
   select * from base3
   union all
      select  srcemploye1_.recid as id0_, srcemploye1_.c_employee as c1_0_, srcemploye1_.c_manager as c2_0_,
              srcemploye1_.i_manager as i3_0_, srcemploye1_.i_addr as i4_0_, srcemploye1_.l_hide as l5_0_
      from    src_employee srcemploye1_
      where   srcemploye1_._multiplex = ?
      join r2 on (
         srcemploye1_.c_employee = r2.c2_0_
      )
)
select * from r2
where id0_ not in (
   select recid
   from   src_employee srcemploye2_
```

```
   where srcemploye2_._multiplex = ? and
         (upper(rtrim(srcemploye2_.c_employee)) = 'DAVE' and srcemploye2_.l_hide = false)
)
```

Note that, although not strictly necessary, I also refactored the queries so that the selects on src_employee use unique table aliases.

To test these queries during in real application using the data currently in temp-tables, you can put breakpoint the right place (for example in BufferImpl.fill()) and execute the following in your debugger expression evaluator:

```
buffer().getPersistence().getSession().getConnection().prepareStatement("<the-query-above>").executeQuery()
```

and investigate the ResultSet value returned.

#78 - 01/04/2023 03:00 PM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Igor Skornyakov wrote:
>
> > I have a strange situation.
> > In fact, it is the same for all 3 dialects. For temp tables, the only difference is that 'special' fields (_multiplex, _errorFlag, ...) are added to the fields' lists, and a condition for the _multiplex is added to the where clause
> > Any suggestions?
>
> From the PoV of the FILL operation, these 'special' fields could be ignored, with two exceptions:
>
> - the _errorFlag, or other similar field is actually used in the query predicate or, more exotically, in source field pairs. This is a thing I come up after reading your note and it is highly improbably that a ABL programmer will use that to fill a dataset table;
> - the _multiplex must always be present in all temp-table selects.
>
> The temp-table - friendly query above should look like this:
> [...]
>
> Note that, although not strictly necessary, I also refactored the queries so that the selects on src_employee use unique table aliases.
>
> To test these queries during in real application using the data currently in temp-tables, you can put breakpoint the right place (for example in BufferImpl.fill()) and execute the following in your debugger expression evaluator:
> [...]and investigate the ResultSet value returned.

Thank you, Ovidiu.
Please note, however, that I have the same problem with non-temp tables with H2 where there are no special fields at all.

#79 - 01/04/2023 03:38 PM - Ovidiu Maxiniuc

You said *"The same query executed against the same H2 database with H2 Console with the same JDBC driver (fwd-h2-1.4.200-6.jar) returns a correct result (for the permanent table)."* There must be some difference in the actual data, then. Use

```
buffer().sqlTableContent(0);
```

to "see" what is the content of a specific temp-table. Note that this will pretty-print the content of the table with all records from all multiplexes. In this case you will have to manually drop the extras. The multiplex of a temp-table buffer is stored in multiplexID field.

#80 - 01/04/2023 03:56 PM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> You said *"The same query executed against the same H2 database with H2 Console with the same JDBC driver (fwd-h2-1.4.200-6.jar) returns a correct result (for the permanent table)."* There must be some difference in the actual data, then. Use[...]to "see" what is the content of a specific temp-table. Note that this will pretty-print the content of the table with all records from all multiplexes. In this case you will have to manually drop the extras. The multiplex of a temp-table buffer is stored in multiplexID field.

Thank you, Ovidiu.
I will continue the investigation tomorrow.

#81 - 01/06/2023 03:55 AM - Igor Skornyakov

Created task branch 6444a from the trunk

#82 - 01/06/2023 09:51 AM - Igor Skornyakov

Committed the current state of the RECURSIVE DATA-RELATION support to 6444a/14473.

#83 - 01/09/2023 06:31 AM - Igor Skornyakov

Here is my understanding of the current task status

DATA-RELATION:RECURSIVE attribute

- The FILL method support - the approach based on the second pass query is almost finished (see the discussion above)
- I've noticed some problems with (READ|WRITE)-XML(SCHEMA)? support, see #6444-19.
- I believe that it makes sense to check the (de)serialization of recursive data relation for the remote calls

The problem with (READ|WRITE)-XML(SCHEMA)? and remote calls is that significant changes for the corresponding functionality was made in the 6129b branch.

FOREIGN-KEY-HIDDEN attribute

I understand that we have to add support for the (READ|WRITE)-XMLSCHEMA. This should be easy. However, I think it also makes sense to check the support of this attribute in the remote calls. However, this also depends on the changes made in 6129b.

RESTART-ROW

According to Ovidiu, this should be similar to the RESTART-ROWID support. However, as far as I can see we support only setting/getting of the RESTART-ROWID. I was unable to find how it affects any functionality.
Fortunately, Marian's team created tests for RESTART-ROW/RESTART-ROWID.

SYNCHRONIZE

So far I cannot say that I completely understand the semantics of this method, but it looks pretty complicated. I was also unable to find any tests for it.

EDITOR:EMPTY attribute

As Ovidio mentioned this is not related to DATA-SET at all. EMPTY-DATASET and EMPTY-TEMP-TABLE methods look to be completely supported. At least the corresponding deleteAll() methods work fine in my tests for FILL.

I understand that the remaining attributes/methods are already completely supported.

#84 - 01/10/2023 09:01 AM - Greg Shah

I'm not sure how the EDITOR:EMPTY attribute was included here. We will move that work to #7032.

#85 - 01/10/2023 09:01 AM - Greg Shah

- Related to Feature #7032: implement the EDITOR:EMPTY attribute added

#86 - 01/11/2023 10:47 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> To test these queries during in real application using the data currently in temp-tables, you can put breakpoint the right place (for example in BufferImpl.fill()) and execute the following in your debugger expression evaluator:
> [...]and investigate the ResultSet value returned.

I've tested with permanent H2 tables.

If I stop with the debugger in the SQLQuery.scroll(Session, int, boolean, List<RowStructure>) after

```
Connection session.getConnection();
```

and execute conn.prepareStatement(<sql>).executeQuery().next() with <sql> is taken from the H2 trace with manually replaced ? with the query parameters (taken from the same trace), I get true.
However, if (in another run) I set a breakpoint in the com.mchange.v2.c3p0.impl.NewProxyResultSet.next() I see that delegated call inner.next() returns false.

The buffer().sqlTableContent(0); (#6444-79) also returns an expected result set.

This looks really weird... Please note that the same test with the same data works as expected with PostgreSQL and MariaDB permanent tables.

#87 - 01/11/2023 11:29 AM - Igor Skornyakov

In addition to #6444-86.
If I set a breakpoint at the last statement of the FqlToSqlConverter.toSQL() (return sql) and replace SQL string with ? placeholders replaced with the query parameters and set paramCount to zero, the test works as expected.
So it looks like the problem is related to setting PreparedStatement parameters.

Has anybody experienced something like this?
Thank you.

#88 - 01/11/2023 02:16 PM - Igor Skornyakov

Igor Skornyakov wrote:

> In addition to #6444-86.
> If I set a breakpoint at the last statement of the FqlToSqlConverter.toSQL() (return sql) and replace SQL string with ? placeholders replaced with the query parameters and set paramCount to zero, the test works as expected.
> So it looks like the problem is related to setting PreparedStatement parameters.

It seems to be a bug in H2 CTE support. I think that what confuses H2 is that the rewritten query contains a query parameter placeholder in the CTE definition.
I have some thoughts about the possible workaround. Working on it.

#89 - 01/11/2023 03:22 PM - Ovidiu Maxiniuc

Igor Skornyakov wrote:

> I've tested with permanent H2 tables.
> If I stop with the debugger in the SQLQuery.scroll(Session, int, boolean, List<RowStructure>) after
> [...]
> and execute conn.prepareStatement(<sql>).executeQuery().next() with <sql> is taken from the H2 trace with manually replaced ? with the query parameters (taken from the same trace), I get true.

Executing .next() will cause the result to advance. You lose the object of interest. You need to investigate the result set obtained from executeQuery() in debugger's watch.
HINT: Try to replace the <sql> with other statement meant to give you a better image on the issue. Use limit with a small value to get a small set of rows.

> However, if (in another run) I set a breakpoint in the com.mchange.v2.c3p0.impl.NewProxyResultSet.next() I see that delegated call inner.next() returns false.

Probably there are no rows to advance to?
LE: IMHO, the query are different. I do not think you can stop with a breakpoint a piece of code already executing with expression evaluator while the main application is also suspended at a breakpoint.

> The buffer().sqlTableContent(0); (#6444-79) also returns an expected result set.
> This looks really weird... Please note that the same test with the same data works as expected with PostgreSQL and MariaDB permanent tables.

OK, after reading your full note, I understand now your problem with next() returning true. You expected it to return false because buffer is empty, as sqlTableContent() returned. This is indeed, strange. Please inspect the rows returned by executeQuery() as noted above, to see what is that query returning and maybe you will understand what goes unexpected.

#90 - 01/12/2023 01:43 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Executing .next() will cause the result to advance. You lose the object of interest. You need to investigate the result set obtained from executeQuery() in debugger's watch.
> HINT: Try to replace the <sql> with other statement meant to give you a better image on the issue. Use limit with a small value to get a small set of rows.
> I understand this. But I call next() for the result set of a different query. Moreover I compare the result of manual query with a 'real' one in different runs.
>
> > However, if (in another run) I set a breakpoint in the com.mchange.v2.c3p0.impl.NewProxyResultSet.next() I see that delegated call inner.next() returns false.
>
> Probably there are no rows to advance to?
> LE: IMHO, the query are different. I do not think you can stop with a breakpoint a piece of code already executing with expression evaluator while the main application is also suspended at a breakpoint.

I disable the com.mchange.v2.c3p0.impl.NewProxyResultSet.next() when I execute a manual query after stopping at the breakpoint in the SQLQuery.scroll(Session, int, boolean, List<RowStructure>) after getting a connection.

> The buffer().sqlTableContent(0); (#6444-79) also returns an expected result set.
> This looks really weird... Please note that the same test with the same data works as expected with PostgreSQL and MariaDB permanent tables.
>
> > OK, after reading your full note, I understand now your problem with next() returning true. You expected it to return false because buffer is empty, as sqlTableContent() returned. This is indeed, strange. Please inspect the rows returned by executeQuery() as noted above, to see what is that query returning and maybe you will understand what goes unexpected.

It returns nothing. Moreover, as I mentioned the problem is gone if the query has no parameters.
It is known that H2 has problems with CTE support, and I'm sure that we have an H2 bug, and looking for a workaround.
Do you think that it makes sense to debug H2 code?
Thank you.

#91 - 01/12/2023 01:51 AM - Igor Skornyakov

> It returns nothing. Moreover, as I mentioned the problem is gone if the query has no parameters.
> It is known that H2 has problems with CTE support, and I'm sure that we have an H2 bug, and looking for a workaround.

In fact, it is a known issue: https://groups.google.com/g/h2-database/c/OJfqNF_Iqyo/m/Z748UP7W3NAJ

#92 - 01/12/2023 10:54 AM - Igor Skornyakov

Added fix for H2 bug with bind parameters in recursive CTEs.
Now the test works fine both temp tables and permanent H2 tables.

Committed to 6444a/14474.

#93 - 01/15/2023 10:48 AM - Igor Skornyakov

Fixed second-pass query generation for queries with CONTAINS operator.

Committed to 6444a/14475.

#94 - 01/16/2023 04:48 AM - Igor Skornyakov

What is the reason to use maxResults and startOffset in the QueryKey for the Persistence.staticQueryCache?
I understand that it results in the SQL generation for different values of maxResults and startOffset for the same query.
This seems to be neither logical nor efficient.
Thank you.

#95 - 01/16/2023 12:00 PM - Eric Faulhaber

Igor Skornyakov wrote:

> What is the reason to use maxResults and startOffset in the QueryKey for the Persistence.staticQueryCache?
> I understand that it results in the SQL generation for different values of maxResults and startOffset for the same query.
> This seems to be neither logical nor efficient.
> Thank you.

The actual values for maxResult and startOffset are not stored in the query key, they are just used to store boolean flags as to whether the query has a starting offset and/or a row limit. I presume this is to differentiate paging queries from queries that request the entire available result set, which as you note will produce different SQL (the latter will have no substitution parameters). This differentiation is needed because the offset and max are stored in the Query object, and the FQL is otherwise the same between paged and full queries, even though the generated SQL will differ. A Query object returned from the cache will not be used by more than one thread, so these values may safely be changed for each use. BTW, the synchronization is therefore not needed around the static query cache checks, since the cache is context-local. This synchronization is being removed in another branch.

Ovidiu, please review my answer above and correct me if I've misstated anything.

#96 - 01/16/2023 12:04 PM - Ovidiu Maxiniuc

I was writing a pretty equivalent response. That is correct. The values for paging are not really used per-se, only to flag the type of paging of the SQL query.

#97 - 01/16/2023 12:06 PM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> I was writing a pretty equivalent response. That is correct. The values for paging are not really used per-se, only to flag the type of paging of the SQL query.

Got it. Thanks a lot!

#98 - 01/17/2023 04:57 AM - Igor Skornyakov

I've noticed a problem with DDL generation.
If the list of dialects in p2j.cfg.xml contains mariadb then in the schema_word_tables_<database>_postgresql.sql I see the following:

```
drop index if exists <idxName>  on <tableName>
```

This form is incorrect for PostgreSQL.
It looks that on conversion the getDropIndexString was called from WordTable.dropIndex() for incorrect dialect MariaDB.

This seems to be a regression.

#99 - 01/17/2023 05:17 AM - Igor Skornyakov

I've got a strange exception.
A very simple test

```
DEFINE TEMP-TABLE ttd NO-UNDO
  NAMESPACE-URI "http://goldencode.com/testNamespace"
  NAMESPACE-PREFIX "fwdPrefix"
  FIELD cE  AS CHARACTER
  FIELD cM  AS CHARACTER
  FIELD val AS CHARACTER
  INDEX idxE AS UNIQUE cE.

DEFINE DATASET ds
  NAMESPACE-URI "http://goldencode.com/testNamespace"
/*  NAMESPACE-PREFIX "fwdPrefix"*/
  FOR ttd
  DATA-RELATION r1 FOR ttd, ttd
    RELATION-FIELDS (cM, cE) RECURSIVE.

DEFINE VARIABLE bh AS HANDLE  NO-UNDO.

CREATE BUFFER bh FOR TABLE "ttsrc".
```

```
DEFINE QUERY qtt FOR ttsrc.

DEFINE DATA-SOURCE srctt FOR QUERY qtt ttsrc .
BUFFER ttd:ATTACH-DATA-SOURCE(DATA-SOURCE srctt:HANDLE).
QUERY qtt:QUERY-PREPARE("FOR EACH ttsrc where ttsrc.cE = 'e1'").

DATASET ds:FILL() NO-ERROR.
RUN show-error("after ds:FILL()").

BUFFER ttsrc:DETACH-DATA-SOURCE().

FOR EACH  ttd:
    MESSAGE ttd.cE ttd.cM ttd.val.
END.

PROCEDURE show-error:
    DEF INPUT PARAM action AS CHAR.

    DEF VAR nmsg AS INTEGER NO-UNDO.
    MESSAGE "After" action ": error =" ERROR-STATUS:ERROR
            "num-messages =" ERROR-STATUS:NUM-MESSAGES
            "type =" ERROR-STATUS:TYPE
            .
    IF ERROR-STATUS:NUM-MESSAGES > 0 THEN DO:
        DO nmsg = 1 TO ERROR-STATUS:NUM-MESSAGES:
          MESSAGE "***" ERROR-STATUS:GET-NUMBER(nmsg) ':' ERROR-STATUS:GET-MESSAGE(nmsg).
        END.
    END.
END.
```

Where ttsrc is a permanent table passes with H2 and MariaDB but with PostgreSQL I see the following:

```
org.postgresql.util.PSQLException: Operation requires a scrollable ResultSet, but this ResultSet is FORWARD_ON
LY.
        at org.postgresql.jdbc.PgResultSet.checkScrollable(PgResultSet.java:280)
        at org.postgresql.jdbc.PgResultSet.first(PgResultSet.java:355)
        at com.mchange.v2.c3p0.impl.NewProxyResultSet.first(NewProxyResultSet.java:815)
        at com.goldencode.p2j.persist.orm.ScrollableResults.execute(ScrollableResults.java:434)
        at com.goldencode.p2j.persist.orm.ScrollableResults.first(ScrollableResults.java:139)
        at com.goldencode.p2j.persist.ScrollingResults.first(ScrollingResults.java:134)
        at com.goldencode.p2j.persist.ResultsAdapter.first(ResultsAdapter.java:131)
        at com.goldencode.p2j.persist.PreselectQuery.first(PreselectQuery.java:2456)
        at com.goldencode.p2j.persist.AdaptiveQuery.first(AdaptiveQuery.java:1228)
        at com.goldencode.p2j.persist.PreselectQuery.first(PreselectQuery.java:2423)
        at com.goldencode.p2j.persist.QueryWrapper.lambda$first$0(QueryWrapper.java:1892)
        at com.goldencode.p2j.persist.QueryWrapper.handleQueryOffEnd(QueryWrapper.java:6884)
        at com.goldencode.p2j.persist.QueryWrapper.first(QueryWrapper.java:1892)
```

This happens on the first pass of the fill() operation with an initial query.

Investigating.

#100 - 01/17/2023 09:30 AM - Igor Skornyakov

I believe that FILL support for recursive DATA-RELATIONs is done.
Committed to 6444a/14476.

Working on #6444-99 (seems to be not related to my changes).

#101 - 01/17/2023 11:24 AM - Igor Skornyakov

Regarding #6444-99:

The situation is a little confusing.
The AdaptiveQuery.executeQuery() instead of creating ProgressiveResults which seems to be most appropriate here, calls PreselectQuery.executeQuery creating ScrollingResults because probablyRequiresResort() returns true.
In addition ScrollingResults.first() calls ResultSet.first() even if it is the first call after executeQuery. This is OK for H2 and MariaDB, but not for PostgreSQL.
Can anybody clarify this logic?
Thank you.

#102 - 01/17/2023 01:18 PM - Igor Skornyakov

Igor Skornyakov wrote:

> Regarding #6444-99:
>
> The situation is a little confusing.
> The AdaptiveQuery.executeQuery() instead of creating ProgressiveResults which seems to be most appropriate here, calls PreselectQuery.executeQuery creating ScrollingResults because probablyRequiresResort() returns true.
> In addition ScrollingResults.first() calls ResultSet.first() even if it is the first call after executeQuery. This is OK for H2 and MariaDB, but not for PostgreSQL.
> Can anybody clarify this logic?
> Thank you.

If I add an explicit ordering to the query in the #6444-99 test, the test passes. So the problem looks to be with using ScrollableResults in the AdaptiveQuery.
Any suggestions?
Thank you.

#103 - 01/17/2023 02:13 PM - Igor Skornyakov

Igor Skornyakov wrote:

> If I add an explicit ordering to the query in the #6444-99 test, the test passes. So the problem looks to be with using ScrollableResults in the AdaptiveQuery.

The ttsrc table in the initial version of #6444-99 had no indexes. If I define one then the test passes with PostgreSQL w/o an explicit ordering on the query.

So I suggest creating a separate task for #6444-99 and switching to the remaining items in #6444-83.
Is it OK?
Thank you.

#104 - 01/17/2023 03:24 PM - Eric Faulhaber

Igor Skornyakov wrote:

> So I suggest creating a separate task for #6444-99 and switching to the remaining items in #6444-83.
> Is it OK?

Yes, that's ok. This doesn't seem specific to the dataset implementation; it seems you have discovered a bug with AdaptiveQuery, so please open a bug issue in the database project. If you can easily simplify the test case to remove the dataset dependency, that would be good, but don't spend a lot of time on it. Thanks.

#105 - 01/18/2023 04:41 AM - Igor Skornyakov

Rebased the task branch 6444a to the trunk/14480.
Pushed up to revision 14484.

#106 - 01/18/2023 05:25 AM - Igor Skornyakov

As mentioned in #6444-83 DATA-SET serialization features should better be tested/fixed on top of changes made for #6453 in 6129b.
Do we have plans to merge 6129b to the trunk in the near future?
Thank you.

#107 - 01/18/2023 10:06 AM - Greg Shah

We would like to do so, but it will not happen for at least 3 weeks.

#108 - 01/18/2023 10:08 AM - Igor Skornyakov

Greg Shah wrote:

> We would like to do so, but it will not happen for at least 3 weeks.

I see. In this case, maybe it makes sense to test/fix the corresponding functionality in 6129b?

#109 - 01/18/2023 10:21 AM - Igor Skornyakov

Igor Skornyakov wrote:

> Greg Shah wrote:
>
> > We would like to do so, but it will not happen for at least 3 weeks.
>
> I see. In this case, maybe it makes sense to test/fix the corresponding functionality in 6129b?

Another option is to manually retrofit #6453 changes from 6129b to 6444a. These changes mostly affect 2-3 classes.
What do you think?
Thank you.

#110 - 01/18/2023 11:13 AM - Eric Faulhaber

Igor, which approach do you think requires the least amount of effort, both for you now, and for merging 6129b to trunk later? Constantin, do you have a preference?

#111 - 01/18/2023 11:20 AM - Igor Skornyakov

Eric Faulhaber wrote:

> Igor, which approach do you think requires the least amount of effort, both for you now, and for merging 6129b to trunk later? Constantin, do you have a preference?

It depends on how many changes were made to the same classes in 6129b compared with the trunk. I feel gutted that manual retrofit will take less effort, especially if we consider re-testing (which is necessary anyway), but I can be wrong.

#112 - 01/18/2023 11:23 AM - Eric Faulhaber

OK, then if Constantin has no objection, let's go with the manual retrofit approach.

#113 - 01/24/2023 10:53 AM - Eric Faulhaber

Igor, have you ported the changes needed from 6129b into 6444a? Where do you feel we now stand on % Done for this issue overall, based on your summary in #6444-83?

#114 - 01/24/2023 11:10 AM - Igor Skornyakov

Eric Faulhaber wrote:

> Igor, have you ported the changes needed from 6129b into 6444a? Where do you feel we now stand on % Done for this issue overall, based on your summary in #6444-83?

Eric,
At this moment I', working on the SYNCHRONIZE method support and hope to finish it in 1-2 days.
After that I will port my changes made for #6453 from 6129b to 6444a. I think that it will take two days (with re-testing).

#115 - 01/26/2023 04:35 AM - Igor Skornyakov

I've encountered a strange problem.
Consider the following program:

```
DEFINE TEMP-TABLE tt1 NO-UNDO
  FIELD id  AS INTEGER
  FIELD val AS CHARACTER
  INDEX idx AS UNIQUE id
.

DEFINE TEMP-TABLE tt2 NO-UNDO
  FIELD id  AS INTEGER
  FIELD ref AS CHARACTER
  FIELD val AS CHARACTER
  INDEX idx AS UNIQUE id
.

DEFINE TEMP-TABLE tt3 NO-UNDO
  FIELD id  AS INTEGER
  FIELD ref AS CHARACTER
  FIELD val AS CHARACTER
  INDEX idx AS UNIQUE id
.

// populate tt1, tt2, tt3

DEFINE DATASET ds FOR tt1, tt2, tt3
    DATA-RELATION r12 FOR tt1, tt2 RELATION-FIELDS(val, ref) REPOSITION
    DATA-RELATION r23 FOR tt2, tt3 RELATION-FIELDS(val, ref) REPOSITION.

DEFINE VARIABLE hDS        AS HANDLE  NO-UNDO.
DEFINE VARIABLE hQuery     AS HANDLE  NO-UNDO.
DEFINE VARIABLE hQ         AS HANDLE  NO-UNDO.
DEFINE VARIABLE hBuffer    AS HANDLE  NO-UNDO.
DEFINE VARIABLE hB         AS HANDLE  NO-UNDO.
DEFINE VARIABLE iBuffer    AS INTEGER NO-UNDO.
DEFINE VARIABLE nR         AS INTEGER NO-UNDO.
DEFINE VARIABLE lc         AS LOGICAL NO-UNDO.

hDS = DATASET ds:HANDLE.

CREATE QUERY hQuery.
```

```
hBuffer = hDS:GET-BUFFER-HANDLE(1).
hQuery:ADD-BUFFER(hBuffer).
hQuery:QUERY-PREPARE("FOR EACH " + hBuffer:NAME).
  hQuery:QUERY-OPEN().
  hQuery:GET-FIRST().
  nR = 0.
  MESSAGE 'Start iterate' iBuffer.
  DO WHILE NOT hQuery:QUERY-OFF-END:
      MESSAGE nR.
      nR = nR + 1.
      CREATE QUERY hQ.
      hB = hDS:GET-BUFFER-HANDLE(2).
      hQ:ADD-BUFFER(hB).
      hQ:QUERY-PREPARE("FOR EACH " + hB:NAME + " WHERE " +  hB:NAME + ".ref = " + hBuffer:NAME + ".val").
      hQ:QUERY-OPEN().
      hQ:GET-LAST().

      MESSAGE '0' hDS:GET-BUFFER-HANDLE(1):BUFFER-FIELD('val'):BUFFER-VALUE()
              hDS:GET-BUFFER-HANDLE(2):BUFFER-FIELD('val'):BUFFER-VALUE()
              hDS:GET-BUFFER-HANDLE(3):BUFFER-FIELD('val'):BUFFER-VALUE()
      .
      hQ:QUERY-CLOSE().
      DELETE OBJECT hQ.

      lc = hDS:GET-BUFFER-HANDLE(1):SYNCHRONIZE().
      MESSAGE '1' lc hDS:GET-BUFFER-HANDLE(1):BUFFER-FIELD('val'):BUFFER-VALUE()
              hDS:GET-BUFFER-HANDLE(2):BUFFER-FIELD('val'):BUFFER-VALUE()
              hDS:GET-BUFFER-HANDLE(3):BUFFER-FIELD('val'):BUFFER-VALUE()
      .
  END.

  hQuery:QUERY-CLOSE().

  DELETE OBJECT hQuery.
  OUTPUT CLOSE.
```

It works fine with 4GL but with FWD I see the following error on hQ:QUERY-PREPARE("FOR EACH " + hB:NAME + " WHERE " + hB:NAME + ".ref = " + hBuffer:NAME + ".val"):

```
[01/26/2023 12:23:35 GMT+03:00] (com.goldencode.p2j.persist.DynamicQueryHelper:WARNING) com.goldencode.testcas
es.ias.DsSync.lambda$execute$0(DsSync.java:190) (ias/ds-sync.p): failed to dynamically convert P4GL code: 'OPE
N QUERY DynGenQuery FOR EACH tt2 WHERE tt2.ref = tt1.val'
tt1.val must be a quoted constant or an unabbreviated, unambiguous buffer/field reference for buffers known to
 query
```

Please note that I cannot add hBuffer (containing tt1.val definition) to hQ since 4GL reports the following error in this case

```
QUERY-PREPARE text must have 1 FOR EACH/PRESELECT for each query buffer. (7325)
```

Any suggestions?
Thank you.

#116 - 01/26/2023 05:40 AM - Igor Skornyakov

I've found a workarount for #6444-115, but at the cost of more complicated (and expensive) query:

```
      crid = STRING(hBuffer:ROWID).
      qstr = "FOR EACH " + hBuffer:NAME + " WHERE ROWID(" + hBuffer:NAME + ") = TO-ROWID('" + crid + "'), EACH
" + hB:NAME + " WHERE " +  hB:NAME + ".ref = " + hBuffer:NAME + ".val".
      CREATE QUERY hQ.
      hQ:ADD-BUFFER(hBuffer).
      hQ:ADD-BUFFER(hB).
      hQ:QUERY-PREPARE(qstr).
      hQ:QUERY-OPEN().
      hQ:GET-LAST().
```

I will use this until a better solution will be found.

#117 - 01/26/2023 07:23 AM - Igor Skornyakov

The HQLPrepocessor.getFindByRowid method looks strange

1. It checks only the first parameter. In my test the rowid is the second parameter.
2. The params instanceof NumberType is false for an integer parameter so the method returns zero in this case and reports Incompatible query parameters warning.

Any suggestions?
Thank you.

#118 - 01/26/2023 10:02 AM - Igor Skornyakov

What is the right way to retrieve the value of the primary key from the BufferImpl instance.
Thank you.

#119 - 01/26/2023 11:53 AM - Igor Skornyakov

Igor Skornyakov wrote:

> What is the right way to retrieve the value of the primary key from the BufferImpl instance.
> Thank you.

Well, it appears that we cannot explicitly use recid in query string which is used in the QUERY-PREPARE.
Please disregard.
Thank you.

#120 - 01/26/2023 01:45 PM - Igor Skornyakov

Finished support for the SYNCHRONIZE method, albeit not in a very efficient way (see #6444-115, #6444-116).

Committed to 6444a/14485.

Please note that I'm still not sure that HQLPrepocessor.getFindByRowid works correctly (see #6444-117)

#121 - 01/26/2023 06:20 PM - Ovidiu Maxiniuc

Igor Skornyakov wrote:

> I've encountered a strange problem. Consider the following program:
> [...]
> It works fine with 4GL but with FWD I see the following error on hQ:QUERY-PREPARE("FOR EACH " + hB:NAME + " WHERE " + hB:NAME + ".ref = " + hBuffer:NAME + ".val"):
> [...]
> Please note that I cannot add hBuffer (containing tt1.val definition) to hQ since 4GL reports the following error in this case
> [...]
> Any suggestions? Thank you.

The program works correctly for me. Almost. In fact it is a neverending loop because you do not call hQuery:GET-NEXT(). before closing the main DO WHILE block. However, if this code does not work for you, the branch have a regression you need to fix.

> I've found a workarount for #6444-115, but at the cost of more complicated (and expensive) query:
> [...]
> I will use this until a better solution will be found.

Sorry, this does not make sense.

I did a quick review of your commits up to 14485/6444a since 14480:

- AbstractQuery, CompoundComponent, P2JQuery, QueryWrapper: method clearDataRelation is equivalent to setDataRelation(null). No need to have a more complicated API then required;
- AdaptiveQuery: lacks the H entry;
- BufferImpl:
  - synchronize(): I think the query's text can be built using a single StringBuilder. Please move this private method in private-area of the the source file.
  - querySynchronize: the childRelations are iterated even after one childBuffer returns false. Is this the case in 4GL? I understand that the method should "reopen each data-relation query", but your code creates new queries. Regardless, after synchronizing the child buffers, the local objects must be deleted.
  - fill(): changes are extensive here, you refactored the old fill method so it is difficult to fully understand the changes in 500+ lines of code
- FqlToSqlConverter:
  - line 2499: I like you coalesced the case s for SUBST and POSITIONAL. Just please put a 'fall through' comment to let the reader know the intention of executing the block of SUBST after the POSITIONAL;
  - generateFrom(): has a return value. Please add the @return javadoc and explain the returned value.
- Session: in createQuery's javadoc there is a new unpaired parameter declared which can be reverted.

So I think you are on the right track. The changes in BufferImpl and FqlToSqlConverter are localized but extensive. I did not expect such expansion of the changeset. I cannot say I fully understand them. The code needs to be thoroughly tested for regression before merging into trunk. How are the new features behave in your tests so far?

#122 - 01/27/2023 02:24 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Igor Skornyakov wrote:
>
> > I've encountered a strange problem. Consider the following program:
> > [...]
> > It works fine with 4GL but with FWD I see the following error on hQ:QUERY-PREPARE("FOR EACH " + hB:NAME + " WHERE " + hB:NAME + ".ref = " + hBuffer:NAME + ".val"):
> > [...]
> > Please note that I cannot add hBuffer (containing tt1.val definition) to hQ since 4GL reports the following error in this case
> > [...]
> > Any suggestions? Thank you.
>
> The program works correctly for me. Almost. In fact it is a neverending loop because you do not call hQuery:GET-NEXT(). before closing the main DO WHILE block. However, if this code does not work for you, the branch have a regression you need to fix.

I've provided only a fragment of the code that demonstrates the problem. With what branch have you tested it?

> I've found a workarount for #6444-115, but at the cost of more complicated (and expensive) query:
> [...]
> I will use this until a better solution will be found.

> Sorry, this does not make sense.

I see. Thank you.

> I did a quick review of your commits up to 14485/6444a since 14480:
>
> - AbstractQuery, CompoundComponent, P2JQuery, QueryWrapper: method clearDataRelation is equivalent to setDataRelation(null). No need to have a more complicated API then required;

Initially I used setDataRelation(null) but for some reasons I had to add clearDataRelation. I will double check.

> - AdaptiveQuery: lacks the H entry;
> - BufferImpl:
>   - synchronize(): I think the query's text can be built using a single StringBuilder. Please move this private method in private-area of the the source file.
>   - querySynchronize: the childRelations are iterated even after one childBuffer returns false. Is this the case in 4GL? I understand that the method should "reopen each data-relation query", but your code creates new queries. Regardless, after synchronizing the child buffers, the local objects must be deleted.
>   - fill(): changes are extensive here, you refactored the old fill method so it is difficult to fully understand the changes in 500+ lines of code
> - FqlToSqlConverter:
>   - line 2499: I like you coalesced the case s for SUBST and POSITIONAL. Just please put a 'fall through' comment to let the reader know the intention of executing the block of SUBST after the POSITIONAL;
>   - generateFrom(): has a return value. Please add the @return javadoc and explain the returned value.
> - Session: in createQuery's javadoc there is a new unpaired parameter declared which can be reverted.

I will address these points. Thank you.

> So I think you are on the right track. The changes in BufferImpl and FqlToSqlConverter are localized but extensive. I did not expect such expansion of the changeset. I cannot say I fully understand them. The code needs to be thoroughly tested for regression before merging into trunk. How are the new features behave in your tests so far?

So far all my tests passed. Of course I will re-test after the problems you've mentioned will be addressed.

#123 - 01/27/2023 07:30 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> I did a quick review of your commits up to 14485/6444a since 14480:
> AbstractQuery, CompoundComponent, P2JQuery, QueryWrapper: method clearDataRelation is equivalent to setDataRelation(null). No need to have a more complicated API then required;

Fixed.

> AdaptiveQuery: lacks the H entry;

Fixed

> BufferImpl:
> synchronize(): I think the query's text can be built using a single StringBuilder. Please move this private method in private-area of the the source file.

Done.

> querySynchronize: the childRelations are iterated even after one childBuffer returns false. Is this the case in 4GL? I understand that the method should "reopen each data-relation query", but your code creates new queries. Regardless, after synchronizing the child buffers, the local objects must be deleted.

Actually the documentations I've seen does not specify when SYNCHRONIZE returns no. I was also unable neither to get this in my tests no to see in the OE examples that the returned value is checked.
Added cleanup to synchronized()

> fill(): changes are extensive here, you refactored the old fill method so it is difficult to fully understand the changes in 500+ lines of code

I see. Unfortunatelly, it was difficult to work with the original version of fill(), sorry

> FqlToSqlConverter:
> line 2499: I like you coalesced the case s for SUBST and POSITIONAL. Just please put a 'fall through' comment to let the reader know the intention of executing the block of SUBST after the POSITIONAL;

Done.

> generateFrom(): has a return value. Please add the @return javadoc and explain the returned value.

Done.

> Session: in createQuery's javadoc there is a new unpaired parameter declared which can be reverted.

Done.

Committed to 6444a/14486.

Working on #6444-115

#124 - 01/27/2023 09:57 AM - Igor Skornyakov

Indeed, #6444-115 does not manifest itself in 6129b.
However, there are a lot of differences between this branch and the trunk.
Are you sure that it makes sense try to fix 6444a? It will take me significant time just to understand what was done in 6129b and retrofit relavant changes to 6444a.
I still suggest to leave my implementation of SYNCHRONIZE in 6444a as-is and wait when 6129b will be merged to the trunk.
What do you think?

One more questions. I see that there is an implementation of the BufferImpl.querySynchronize() method in 6129b, but in my test it effectively does nothing.
Ovidiu,
Can you please provide a test you've used for testing your implementation? I want to test my version with it.

Thank you.

#125 - 01/27/2023 12:08 PM - Igor Skornyakov

Retrofitted Ovidio's changes for the DynamicQueryHelper from 6129b to 6444a.
This resolves #6444-115. Reworked BufferImpl.synchronize() using more simple and efficient query/
Committed to 6444a/14487.

#126 - 01/27/2023 08:22 PM - Ovidiu Maxiniuc

I looked in the log of FWD repository and, indeed it shows I added some implementation for synchronize method (and some other methods), in May, last year. I really forgot about this one. I was originally committed to 6129a and it has not yet merged into 3821c (now trunk). My review in note-4 (dated November) was using that latter branch. Note that while initial changes in 6129a were addressing missing features required by a new project, the latest changes in 6129b are primarily targetted at performance improvements. So 6129b contains a ton of changes and the missing buffers issue was only one of them.

The implementation use a de-recursive solution to iterate the buffers and re-open the relation query. The code is not fully functional: it lacks the error handling (if any) and reposition of the query on the current row. Also, I do not know how the recursive relations are supported.

I am glad you spot the duplicated code. Hopefully this will reduce a bit the nightmare of merging 6129b into trunk, too.

Please see the DynamicQueryHelper's header, it probably make the file not compilable.
And generally, please place the { on a new line (FWD coding standard). I forgot to add this in my previous review.

#127 - 01/28/2023 02:56 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> I looked in the log of FWD repository and, indeed it shows I added some implementation for synchronize method (and some other methods), in May, last year. I really forgot about this one. I was originally committed to 6129a and it has not yet merged into 3821c (now trunk). My review in note-4 (dated November) was using that latter branch. Note that while initial changes in 6129a were addressing missing features required by a new project, the latest changes in 6129b are primarily targetted at performance improvements. So 6129b contains a ton of changes and the missing buffers issue was only one of them.
>
> The implementation use a de-recursive solution to iterate the buffers and re-open the relation query. The code is not fully functional: it lacks the error handling (if any) and reposition of the query on the current row. Also, I do not know how the recursive relations are supported.
>
> I am glad you spot the duplicated code. Hopefully this will reduce a bit the nightmare of merging 6129b into trunk, too.
>
> Please see the DynamicQueryHelper's header, it probably make the file not compilable.

Thank you, Ovidiu.I've managed to extract relevant changes from 6129b (see #6444-125).

> And generally, please place the { on a new line (FWD coding standard). I forgot to add this in my previous review.

I remember about the coding standards and trying to obey them. I will review my changes, but, unfortunately, I often just do not see formatting problems. Sorry.

#128 - 01/28/2023 03:35 AM - Igor Skornyakov

> > And generally, please place the { on a new line (FWD coding standard). I forgot to add this in my previous review.
>
> I remember about the coding standards and trying to obey them. I will review my changes, but, unfortunately, I often just do not see formatting problems. Sorry.

I've scanned *.java files for the ^[^/\s]+.*\{ regexp but found it only in comments, generated files and StringOutputStream.java (created 2020/09/26).
Have aI overlooked something?
Thank you.

#129 - 01/28/2023 03:54 AM - Igor Skornyakov

- File ds-sync.p added

Marian,
The description of the SYNCHRONIZE method semantics in the OE documentation looks not very clear for me.
I've tested my implementation with the attached test. This test now provides the identical results with 4GL and FWD.
Can you suggest any additional test cases to ensure that this implementation is correct and complete?
Thank you.

#130 - 01/30/2023 03:12 AM - Marian Edu

Igor Skornyakov wrote:

> Marian,
> The description of the SYNCHRONIZE method semantics in the OE documentation looks not very clear for me.
> I've tested my implementation with the attached test. This test now provides the identical results with 4GL and FWD.
> Can you suggest any additional test cases to ensure that this implementation is correct and complete?
> Thank you.

Hi Igor, will look at what tests we have for this one and maybe I can come up with something more - I think there is also a 'SYNCHRONIZE' event you might want to look out for. If there is an event handler for that, one can stop synchronization to cascade down the child tables with return no-apply. Other than that this is often used with a browser and behaviour is somehow different if the query is attached to a browse widget or not.

#131 - 01/30/2023 03:17 AM - Igor Skornyakov

Marian Edu wrote:

> Igor Skornyakov wrote:
>
> > Marian,
> > The description of the SYNCHRONIZE method semantics in the OE documentation looks not very clear for me.
> > I've tested my implementation with the attached test. This test now provides the identical results with 4GL and FWD.
> > Can you suggest any additional test cases to ensure that this implementation is correct and complete?
> > Thank you.
>
> Hi Igor, will look at what tests we have for this one and maybe I can come up with something more - I think there is also a 'SYNCHRONIZE' event you might want to look out for. If there is an event handler for that, one can stop synchronization to cascade down the child tables with return no-apply. Other than that this is often used with a browser and behaviour is somehow different if the query is attached to a browse widget or not.

Thank you, Marian.
I will greatly appreciate any additional tests and/or references. The ones I've found look absolutely cryptic for me...

#132 - 02/01/2023 10:11 AM - Igor Skornyakov

Rebased task branch 6444a to the trunk rev.14483.
Suspended working on this task while the trunk and 6129b are frozen.

#133 - 02/15/2023 10:52 AM - Igor Skornyakov

Branch 6444a was rebased to the trunk rev. 14484.
Pushed up to revision 14493.

#134 - 02/23/2023 12:35 PM - Igor Skornyakov

I'm re-testing temp-table ans data-set changes from #6453 and this task with 6444a rebased to the trunk.
After fixing READ-XMLSCHEMA for the recursive DATA-SET I've noticed the problem with the dynamic DATA-SET for the new test case I've not before.

If the dynamic DATA-SET is created from the XML schema containing an index with a mixed case name the name of the corresponding index after READ-XMLSCHEMA is converted to lower case.
Strangely enough I cannot find where it happens. When I trace READ-XMLSCHEMA with a debugger I always see a correct value of the legacyName, but at the end I see it in a lower case.
Can anybody give me a clue where should I look? I've trying to set a breakpoint on the access to legacyName field, but it mysteriously get its value out of a sudden))
Thank you.

#135 - 02/25/2023 07:16 AM - Igor Skornyakov

Finished re-testing fixes for this task and #6453 after rebasing 6444a branch to the new trunk is finished.
Committed to 6444a/14498.

The only differences with 4GL are:

1. The names of the dynamic table index is converted to lowercase in FWD but not with 4GL (#6444-134).
2. The order of the DATA-SET indexes in the WRITE-XMLSCHEMA output is different in FWG and in 4GL. In 4GL they look to be ordered by the index name while in FWD we use an "order of their buffers" which looks a little strange for me. How can we be 100% sure that the internal order of the DATA-SET buffers is always the same in FWD and 4GL?

Anyway, both differences does not seem important and can hardly affect applicatio's behavior, so I suggest leaving them as-is.

Please note that 6444a now contains a lot of changes.
May be it makes sense to consider merging them into the trunk?
Thank you.

#136 - 02/27/2023 04:58 AM - Igor Skornyakov

Tested the (de)serialization of recursive data relation for the remote calls.

Works OK.

#137 - 03/01/2023 01:31 PM - Igor Skornyakov

Added support for the RESTART-ROW attribute.

Committed to 6444a/14499.

#138 - 03/02/2023 02:38 AM - Igor Skornyakov

Rebased 6444a to the trunk/14486.
Pushed up to revision 14501.

#139 - 03/02/2023 02:11 PM - Igor Skornyakov

I'm working now on the only remaining subtask - the DATA-RELATION:FOREIGN-KEY-HIDDEN attribute support which can be used only with NESTED attribute.
It appears however, that the NESTED support in broken in 6444a, so I have to fix it first.

#140 - 03/07/2023 07:23 AM - Igor Skornyakov

I've created a more advanced test with five tables, multiple namespaces and multiple nested data-reltions.
It appears that 4GL logic for WRITE-XMLSCHEMA is much more complicated than I expected before based on a basic test with only two tables and one relation.
It involves logical ordering of tables based on parent/child relation in the nested data-reltions and grouping tables with the same namespace in external schemas.
Trying to decipher it.

#141 - 03/09/2023 08:39 AM - Igor Skornyakov

- File hds_tt5.xsd added

- File nested-test.p added

- File ds.xsd added

- File ds_tt1.xsd added

- File ds_tt3.xsd added

- File hds.xsd added

- File hds_tt3.xsd added

I've noticed a strange thing. Consider the attached test nested-test.p.

It defines a DATA-SET with 5 tables (tt1, tt2, tt3, tt4, and tt5), 4 nested relations and 3 namespaces. The WRITE-XMLSCHEMA for this dataset creates 3 .xsd files (one per namespace) ds.xsd, ds_tt1.xsd and ds.tt3.xsd.

After importing ds.xsd using READ-XMLSCHEMA we get a dynamic DATA-SET hds with 5 tables (tt5, tt1, tt2, tt3, and tt4), 4 nested relations and 3 namespaces. However, if we export it using WRITE-XMLSCHEMA we again get 3 .xsd files hds.xsd, ds_tt3.xsd and ds.tt5.xsd.
The problems is that these .xsd files do not contain definitions for tt1 and tt2, albeit one can see definitions of data-relations and indexes referring to these tables.
Moreover, hds.xsd cannot be imported with READ-XMLSCHEMA, the import fail with

```
*** 3135 : Invalid handle.  Not initialized or points to a deleted object. (3135)
*** 3140 : Cannot access the READ-XMLSCHEMA attribute because the widget does not exist. (3140)
```

This looks incorrect. As far as I understand which tables are missed depends on the namespaces. For example if all 5 tables have different names spaces there are no missing tables in the export of the imported DATA-SET.

Should I try to reproduce this quirk in FWD?
Thank you.

#142 - 03/09/2023 01:20 PM - Igor Skornyakov

More about the situation described in #6444-141.

Although in the Progress Developer Studio debugger hds after import looks identically to dsh, it looks that internally the hds DATA-SET is somehow corrupted.
In addition to the strange output of the hds:WRITE-XMLSCHEMA the hds:READ-XML from the exported ds data fails with

```
*** 13035 : Error reading XML file 'ds-nested-test/ds.xml'. (13035)
*** 13064 : READ-XML encountered an error while parsing the XML Document: 'no declaration found for element 'f
wdPrefix:tt2''. (13064)
```

#143 - 03/10/2023 02:09 PM - Eric Faulhaber

Igor Skornyakov wrote:

> [...]
> This looks incorrect. As far as I understand which tables are missed depends on the namespaces. For example if all 5 tables have different names spaces there are no missing tables in the export of the imported DATA-SET.
>
> Should I try to reproduce this quirk in FWD?
> Thank you.

I see no reason to recreate this; it seems like a bug upon which it would be extremely difficult to rely for the correct behavior of an application.

Ovidiu?

#144 - 03/12/2023 09:26 AM - Igor Skornyakov

Fixed WRITE-XML for multiple namespaces and NESTED relations.
Committed to 6444a/14508.

#145 - 03/12/2023 10:13 AM - Igor Skornyakov

Rebased 6444a to the trunk/14500.

Pushed up to revision 14517.

#146 - 03/12/2023 10:28 AM - Igor Skornyakov

After rebasing I see a regression in the FILL test for temporary source tables and recursive DATA-RELATION.

#147 - 03/12/2023 12:40 PM - Igor Skornyakov

- File AdaptiveQuery.diff added

Igor Skornyakov wrote:

> After rebasing I see a regression in the FILL test for temporary source tables and recursive DATA-RELATION.

The problem is with 'lazyMode'. After applying the attached path the regression has gone.

Should I try to fix this?
Thank you.

#148 - 03/13/2023 02:14 AM - Eric Faulhaber

Igor Skornyakov wrote:

> Igor Skornyakov wrote:
>
> > After rebasing I see a regression in the FILL test for temporary source tables and recursive DATA-RELATION.
>
> The problem is with 'lazyMode'. After applying the attached path the regression has gone.
>
> Should I try to fix this?
> Thank you.

Please post the test case, explain the regression to Radu, and let him fix it.

#149 - 03/13/2023 02:51 AM - Igor Skornyakov

- File recursive_relation_fill_tmp.txt added

- File recursive_relation_fill_temp.p added

- File recursive_relation_fill_tmp.txt added

Eric Faulhaber wrote:

> Igor Skornyakov wrote:
>
> > Igor Skornyakov wrote:
> >
> > > After rebasing I see a regression in the FILL test for temporary source tables and recursive DATA-RELATION.
> >
> > The problem is with 'lazyMode'. After applying the attached path the regression has gone.
> >
> > Should I try to fix this?
> > Thank you.
>
> Please post the test case, explain the regression to Radu, and let him fix it.

The testcase is attached.
It tests the FILL operation for a recursive data relation. This operation is performed in two steps:

1. First a 'normal' FILL is done
2. After that the second step is executed at which the SQL query is re-created using a recursive CTE (see #6444-75).

The main logic is implemented in the BufferImpl.FillWorker and FqlToSqlConverter. However, the logic is pretty complicated and fragile since we have to propagate information about a 'special' conversion mode to FqlToSqlConverter (see #6444-56). It looks that this propagation is somehow broken with lazyMode.

#150 - 03/13/2023 03:08 AM - Igor Skornyakov

- File deleted (recursive_relation_fill_tmp.txt)

#151 - 03/13/2023 03:09 AM - Igor Skornyakov

- File recursive_relation_fill_tmp.txt added

Sorry, I've attached a wrong file.

#152 - 03/13/2023 03:59 AM - Radu Apetrii

Eric Faulhaber wrote:

> Please post the test case, explain the regression to Radu, and let him fix it.

I'm working on it. I'll let you know as soon as I find the problem.

#153 - 03/13/2023 05:28 AM - Igor Skornyakov

Fixed minor issues with WRITE-XML(SCHEMA)? for tables.
Committed to 6444a/14519.

#154 - 03/13/2023 09:32 AM - Radu Apetrii

Igor Skornyakov wrote:

> The problem is with 'lazyMode'. After applying the attached path the regression has gone.

Igor, I applied the patch that you have attached, but the output of the test is still incorrect (I only get an empty file). More than that, I removed most of the "lazy" changes and I still couldn't get the expected result.

I've even tried running the test on an older version of FWD and the result is still the same (incorrect). Note: that version did not include any "lazy" changes.

Are you sure that the problem is with 'lazyMode'? Or is there something that I did wrong?

#155 - 03/13/2023 09:55 AM - Igor Skornyakov

Radu Apetrii wrote:

> Igor Skornyakov wrote:
>
> > The problem is with 'lazyMode'. After applying the attached path the regression has gone.
>
> Igor, I applied the patch that you have attached, but the output of the test is still incorrect (I only get an empty file). More than that, I removed most of the "lazy" changes and I still couldn't get the expected result.
>
> I've even tried running the test on an older version of FWD and the result is still the same (incorrect). Note: that version did not include any "lazy" changes.
>
> Are you sure that the problem is with 'lazyMode'? Or is there something that I did wrong?

Yes, I'm pretty sure about this.
Please note that this happens in 6444a branch. Please double check that you've tested with this branch.
Thank you.

#156 - 03/13/2023 03:21 PM - Ovidiu Maxiniuc

Igor Skornyakov wrote:

> I've noticed a strange thing. Consider the attached test nested-test.p.
> It defines a DATA-SET with 5 tables (tt1, tt2, tt3, tt4, and tt5), 4 nested relations and 3 namespaces. The WRITE-XMLSCHEMA for this dataset creates 3 .xsd files (one per namespace) ds.xsd, ds_tt1.xsd and ds.tt3.xsd.
>
> After importing ds.xsd using READ-XMLSCHEMA we get a dynamic DATA-SET hds with 5 tables (tt5, tt1, tt2, tt3, and tt4), 4 nested relations and 3 namespaces. However, if we export it using WRITE-XMLSCHEMA we again get 3 .xsd files hds.xsd, ds_tt3.xsd and ds.tt5.xsd.
> The problems is that these .xsd files do not contain definitions for tt1 and tt2, albeit one can see definitions of data-relations and indexes referring to these tables.
> Moreover, hds.xsd cannot be imported with READ-XMLSCHEMA, the import fail with
> [...]
> This looks incorrect. As far as I understand which tables are missed depends on the namespaces. For example if all 5 tables have different names spaces there are no missing tables in the export of the imported DATA-SET.
>
> Should I try to reproduce this quirk in FWD?

Eric Faulhaber wrote:

> I see no reason to recreate this; it seems like a bug upon which it would be extremely difficult to rely for the correct behavior of an application.
>
> Ovidiu?

The main difference between ds.xsd and the deserialied hds.xsd is that the namespace, the former is using xpath=".//tns1:tt1" and the latter xpath=".//fwdPrefix:tt1". I do not understand what tns1 stands for.

The big issue is with ds_tt1.xsd and especially its counterpart hds_tt5.xsd. I understand that there was a bit on non-determinism in picking the name but the big problem is hat the former has a double definition of tt1 and tt2, while the latter has none of them.

It's, evidently, a bug reveal by the complexity of the testcase. I think we should not dirty the already complex code for de-/serializing xml schema in FWD. If a customer really needs this (I really do not expect anybody to develop a part of an application based on this strange behaviour) we will address it at that time.

#157 - 03/13/2023 03:44 PM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> The main difference between ds.xsd and the deserialied hds.xsd is that the namespace, the former is using xpath=".//tns1:tt1" and the latter xpath=".//fwdPrefix:tt1". I do not understand what tns1 stands for.

tnsN is a synthetic prefix for table namspaces where the first table in a namespace has no prefix.

> The big issue is with ds_tt1.xsd and especially its counterpart hds_tt5.xsd. I understand that there was a bit on non-determinism in picking the name but the big problem is hat the former has a double definition of tt1 and tt2, while the latter has none of them.

Actually the re-ordering is understandable. It happens when the a table in a namespace is a (possibly indirect) parent of the table from the same namespace defined earlier in the the original DATA-SET. The order of the tables in the imported DATA-SET is a transitive closure of the partial ordering defined by the parent/child in the NESTED relations in the DATA-SET and an intital ordering if the parent/child order is undefined.

> It's, evidently, a bug reveal by the complexity of the testcase. I think we should not dirty the already complex code for de-/serializing xml schema in FWD. If a customer really needs this (I really do not expect anybody to develop a part of an application based on this strange behaviour) we will address it at that time.

I agree. In fact I've finished WRITE-XML(SCHEMA)? support for DATA-SETs with NESTED DATA-RELATIONs and multiple namespaces which is fully compatible with 4GL and almost finished a **correct** READ-XML(SCHEMA)? implementation for such DATA-SETs.

#158 - 03/14/2023 04:51 AM - Igor Skornyakov

Fixed READ-XML(SCHEMA)? support for DATA-SETs with NESTED DATA-RELATIONs and multiple namespaces.
It works correctly even in situations when 4GL support is broken.
Committed to 6444a/14520.

#159 - 03/14/2023 04:59 AM - Igor Skornyakov

Rebased 6444a to the trunk/14503.

Pushed up to revision 14523.

#160 - 03/14/2023 07:23 AM - Igor Skornyakov

- Related to Bug #7193: READ-JSON is not supported for the "LONGCHAR" source type added

#161 - 03/14/2023 02:11 PM - Igor Skornyakov

Added FOREIGN-KEY-HIDDEN support for WRITE-XML,(READ|WRITE)-XMLSCHEMA.
Committed to 6444a/14524

The remaining things are FOREIGN-KEY-HIDDEN support for READ-XML and (READ|WRITE)-JSON.

READ-XML and WRITE-JSON are simple and will be finished tomorrow.
However, considering the current state of the READ-JSON support (see #7193) I cannot say how much time will take FOREIGN-KEY-HIDDEN support for it.
Please note also that we cannot merge 6444a to the trunk until #6444-149 will be fixed.

#162 - 03/15/2023 12:45 PM - Igor Skornyakov

- File ds-nested-test-FWD.zip added

- File nested-fk.p added

- File ds-nested-test-4GL.zip added

Fixed the the NESTED/FOREIGN-KEY-HIDDEN support for WRITE-JSON.
6444a/14525.
The current status of the NESTED/FOREIGN-KEY-HIDDEN support (the last subtask of this task is following).

- For XML (de)serializtion everything is done. I've noticed another 4GL bug for READ-XML in addition to problem with READ-XML-SCHEMA described in #6444-141. In my test if I serialize static populated dataset ds to an XML file, empty it and try to reload data from the XML 4GL fails with

```
*** 13035 : Error reading XML file 'ds-nested-test/ds.xml'. (13035)
*** 13064 : READ-XML encountered an error while parsing the XML Document: 'element 'int64-field' is not al
lowed for content model '(char-node,f2charcs,f2decimal,f2int,int64-field,logical)''. (13064)
```

  FWD handles both this test case and #6444-141 correctly.
- As mentioned above WRITE-JSON support for the FOREIGN-KEY-HIDDEN in FWD is now compatible with 4GL albeit 4GL and FWD format output JSON in a slightly different way.

As mentiond in #7193, READ-JON support in FWD is incomplete. However, for the NESTED/FOREIGN-KEY-HIDDEN support is doesn't matter since if we use READ-JSON in unitinialized DATA-SET handle 4GL knows nothing about the presence of this attribute.
So I've tested the scanario when populated DATA-SET is serialized to JSON file, emptied and then reloaded from this file.
In this situation 4GL support is broken - it does't populate relation fields in the child table, moreover these fields sometime get a random(?) value.
FWD support for READ-JSON in this testcase is also broken, but in a different way.

The attached files contain my test and its results for 4GL and FWD. To simplify comparision I've post-processed 4GL output using find -type f -name "*.x*" -exec sed -i "1s/\"/'/g" {} +
To replace " with ' in the first line of .xml/.xsd files (XML header).

At the botton line.
Apart from READ-JSON support and #6444-149 all is done and 6444a is ready for the code review.
I cannot say how long will take fixing the READ-JSON support but #6444-149 should be simple.
Should I work on one of these subtasks (or both)?
Thank you.

#163 - 03/15/2023 11:48 PM - Eric Faulhaber

Igor Skornyakov wrote:

> Apart from READ-JSON support and #6444-149 all is done and 6444a is ready for the code review.

Ovidiu, please review, but with lower priority than the urgent bugs you are working on right now.

> I cannot say how long will take fixing the READ-JSON support but #6444-149 should be simple.

Radu, have you been able to recreate the problem with branch #6444a?

> Should I work on one of these subtasks (or both)?

Rather than just commenting out the lazy mode code in AdaptiveQuery, as in #6444-147, I still want Radu to make a proper fix for the regression. So, please do not work on that.

Please fix the READ-JSON support. If it is not dependent on changes in 6444a, please do this work in a 7193a task branch, so we don't complicate the review of 6444a.

#164 - 03/16/2023 02:14 AM - Igor Skornyakov

Eric Faulhaber wrote:

> Rather than just commenting out the lazy mode code in AdaptiveQuery, as in #6444-147, I still want Radu to make a proper fix for the regression. So, please do not work on that.
>
> Please fix the READ-JSON support. If it is not dependent on changes in 6444a, please do this work in a 7193a task branch, so we don't complicate the review of 6444a.

Got it. Thank you.
Working on 7139.

#165 - 03/16/2023 05:12 AM - Radu Apetrii

Eric Faulhaber wrote:

> Radu, have you been able to recreate the problem with branch #6444a?

Yes, I have.

In AdaptiveQuery.executeQuery, due to the introduction of lazyMode, all the ProgressiveResults that were generated for these cases (for temporary tables) have been replaced by ScrollingResults. Although this shouldn't be a problem, when executing the fill method, it seems that ScrollingResults is not capable of retrieving all the records necessary.
I'm currently looking further into this matter and hopefully I'll solve it ASAP.

#166 - 03/16/2023 05:41 AM - Radu Apetrii

Radu Apetrii wrote:

> In AdaptiveQuery.executeQuery, due to the introduction of lazyMode, all the ProgressiveResults that were generated for these cases (for temporary tables) have been replaced by ScrollingResults. Although this shouldn't be a problem, when executing the fill method, it seems that ScrollingResults is not capable of retrieving all the records necessary.

Regarding my previous comment, I would like to point out one thing. If you were to remove that code with lazyMode, the program would work, but not because lazy is broken. The problem was the replacement of ProgressiveResults with ScrollingResults.
If you were to simply execute the queries in the fill method with non-lazy ScrollingResults instead of ProgressiveResults, the output would still be incorrect.
I recall seeing this problem with ScrollingResults vs ProgressiveResults in some other tests I did.

#167 - 03/16/2023 09:54 AM - Radu Apetrii

- File recursive_relation_fill_temp.patch added

Igor, I've attached a patch that should solve the problem.

There was a parameter in AdaptiveQuery.executeScroll at persistence.scroll which should've been the dataRelation instead of null. There were some missing queries when executing fill and it looks like they needed the dataRelation.

Also, I found two spots in the code that were missing the lazy logic. This meant that no queries were executed lazy because the SQL didn't have that keyword.
Side note: In 7026b, rev. 14504 there were some conditions added in order to handle more effective the whole lazy/non-lazy situation, as well as the registration as a listener process. They are listed in #7061-28.

If you encounter any more problems, please let me know.

#168 - 03/16/2023 10:06 AM - Igor Skornyakov

Radu Apetrii wrote:

> Igor, I've attached a patch that should solve the problem.
>
> There was a parameter in AdaptiveQuery.executeScroll at persistence.scroll which should've been the dataRelation instead of null. There were some missing queries when executing fill and it looks like they needed the dataRelation.
>
> Also, I found two spots in the code that were missing the lazy logic. This meant that no queries were executed lazy because the SQL didn't have that keyword.
> Side note: In 7026b, rev. 14504 there were some conditions added in order to handle more effective the whole lazy/non-lazy situation, as well as the registration as a listener process. They are listed in #7061-28.
>
> If you encounter any more problems, please let me know.

Thank you. Radu.
I will check your patch shortly.

#169 - 03/16/2023 10:31 AM - Igor Skornyakov

- Assignee changed from Igor Skornyakov to Ovidiu Maxiniuc

- Status changed from WIP to Review

Radu Apetrii wrote:

> Igor, I've attached a patch that should solve the problem.
>
> There was a parameter in AdaptiveQuery.executeScroll at persistence.scroll which should've been the dataRelation instead of null. There were some missing queries when executing fill and it looks like they needed the dataRelation.
>
> Also, I found two spots in the code that were missing the lazy logic. This meant that no queries were executed lazy because the SQL didn't have that keyword.
> Side note: In 7026b, rev. 14504 there were some conditions added in order to handle more effective the whole lazy/non-lazy situation, as well as the registration as a listener process. They are listed in #7061-28.
>
> If you encounter any more problems, please let me know.

The patch was tested and committed to 6444a/14526.
6444a is ready for the code review.

#170 - 03/16/2023 12:02 PM - Greg Shah

Are all of your testcases integrated into the dataset tests in Testcases? If not, they should be.

Marian's team is reworking them to use ABLUnit in a fully automated mode. We would want your testcases to match those changes.

#171 - 03/16/2023 12:06 PM - Igor Skornyakov

Greg Shah wrote:

> Are all of your testcases integrated into the dataset tests in Testcases? If not, they should be.
>
> Marian's team is reworking them to use ABLUnit in a fully automated mode. We would want your testcases to match those changes.

This not done yet. Will be added soon.

#172 - 03/16/2023 12:08 PM - Greg Shah

Marian: Do you want Igor to commit his tests into the current Testcases or do you want some other process to make it easier to merge your pending ABLUnit changes in?

#173 - 03/16/2023 12:25 PM - Igor Skornyakov

Please note that my approach for testing is based on visual comparision of multiple output files generated by the test running with 4GL and FWD.
It can be tricky to create a fully automated version of them.
I also think that it is desirable to retain these output files will for analisys if the tests fails. Just "testcase failed" will be not enough.

#174 - 03/16/2023 12:40 PM - Greg Shah

I agree. It is good to be able to look deeply at any issue. I just want the comparison/detection of a problem to be automated.

The Harness has good Java based facilities for text file comparisons but we would need to use these tools from 4GL code which is not hard in FWD but is harder in OE.

Marian: Has your team had to do this with the existing tests?

#175 - 03/16/2023 12:46 PM - Igor Skornyakov

Greg Shah wrote:

> I agree. It is good to be able to look deeply at any issue. I just want the comparison/detection of a problem to be automated.
>
> The Harness has good Java based facilities for text file comparisons but we would need to use these tools from 4GL code which is not hard in FWD but is harder in OE.

My understand that we will run the tests with 4GL first and then re-run them with FWD, so there is not need to compare files in OE.
The problem is to automate the OE output files' placement. I understand that this is indeed very similar to what we have in Harness.

#176 - 03/16/2023 12:53 PM - Igor Skornyakov

Please note also that some tests generate files not only in the client home but also in the app. server home. This makes gathering the output data more complicated.

#177 - 03/16/2023 12:55 PM - Greg Shah

We may want this to be configurable.

Why does it generate output on the server side? That doesn't sound expected.

#178 - 03/16/2023 01:08 PM - Igor Skornyakov

Greg Shah wrote:

> We may want this to be configurable.
>
> Why does it generate output on the server side? That doesn't sound expected.

This is for testing TEMP-TABLEs DATA-SETs as arguments in the remote calls.

#179 - 03/17/2023 03:09 AM - Marian Edu

Greg Shah wrote:

> Marian: Has your team had to do this with the existing tests?

We've followed a similar approach in the beginning, it might still be used at time but normally we've added some helper methods to compare temp-table/dataset so we often use one as a baseline and then dump/load (write/read) in another one and we compare them in the end.

#180 - 03/17/2023 03:12 AM - Marian Edu

Greg Shah wrote:

> Marian: Do you want Igor to commit his tests into the current Testcases or do you want some other process to make it easier to merge your pending ABLUnit changes in?

He can commit it to testcases and will merge them when we push the dataset rewrite, we're moving out old tests from the project root into a `tests` folder because there are many other things in there like database definition/dump, application server stuff, support files we're using in tests.

#181 - 03/17/2023 03:16 AM - Marian Edu

Igor Skornyakov wrote:

> This is for testing TEMP-TABLEs DATA-SETs as arguments in the remote calls.

Igor, I think the easiest way to do that is to simply use a 'pipe through' routine - I think we have that already for when Constantin implemented the appsrv stuff, it's just passing a temp-table/dataset around either using input-output or a pair of input/output parameters. So you can send something out and expect to receive the same thing back, then I would assume the `serialisation` is working.

#182 - 03/17/2023 04:14 AM - Igor Skornyakov

Marian Edu wrote:

> Igor, I think the easiest way to do that is to simply use a 'pipe through' routine - I think we have that already for when Constantin implemented the appsrv stuff, it's just passing a temp-table/dataset around either using input-output or a pair of input/output parameters. So you can send something out and expect to receive the same thing back, then I would assume the `serialisation` is working.

Marian,
I was thinking about this, but then decided that for me it will take less time to use my approach (my knowledge of 4GL is much less advanced than yours). In addition it simplifies the testing since I see a result of a single (de)serialization instead of the double ones which can be more difficult to analyze if something is going wrong. It also excludes false positive results due to compensating errors.

#183 - 03/17/2023 05:49 AM - Igor Skornyakov

Tests for #6453 and #6444 has been added to 6453-6444 subfolder of the xfer.goldencode.com/opt/testcases/ rev. 1335.

#184 - 03/17/2023 06:11 AM - Greg Shah

Marian/Constantin: Can you recommend a better place for these tests than 6453-6444/?

Igor: We don't want to store these in such a random location. The tests need to be grouped with other tests based on functionality. Either these are related to parameter passing or they are related to datasets. The problem with the naming is that no one will ever realize "I need to run the tests from 6453-6444/ because I am making changes to parameter processing or datasets".

#185 - 03/17/2023 06:15 AM - Igor Skornyakov

Greg Shah wrote:

> Marian/Constantin: Can you recommend a better place for these tests than 6453-6444/?
>
> Igor: We don't want to store these in such a random location. The tests need to be grouped with other tests based on functionality. Either these

are related to parameter passing or they are related to datasets. The problem with the naming is that no one will ever realize "I need to run the tests from 6453-6444/ because I am making changes to parameter processing or datasets".

Greg,
I understand that these tests will be re-worked by Marian's team so I decided to put them in a separate folder because they are implemented in a very different style than existing tests for the same or related functunality.

#186 - 03/17/2023 06:30 AM - Greg Shah

Please don't. They will get forgotten. Marian has already confirmed he will handle the merge.

#187 - 03/27/2023 11:34 PM - Ovidiu Maxiniuc

Review of 6444a/14538:

- AbstractQuery.java, CompoundComponent.java, Dialect.java, etc: GCD coding standard: please move the method's opening curly braces ({) on the next line for g/setDataRelation() methods.
- BufferImpl.java:
  - please define PK as Session.PK (or maybe better use this final public field. It is computed at runtime as opposed to SchemaConfig.schemaConfig which is used at conversion time;
  - you reverted Constantin's work on lazy callback (20230110);
  - querySynchronize() and synchronize(DataRelation rel):
    - will throw NPE if the buffer is not a DataSet component;
    - local variable relation is not used;
    - please use the new, faster rel.getChildBufferNative() instead of rel.getChildBuffer().unwrapBuffer(). Generally, if you **know** the expected resource type, please use getResource() instead of unwrap...().
    - according to https://docs.progress.com/bundle/openedge-prodatasets-117/page/Using-the-SYNCHRONIZE-method.html, method should position the query OFF-END and the GET-FIRST should be called by the ABL programmer. More than that I am no sure creating a new dynamic query is the proper way to implement this. What if the relation is a PARENT-ID-RELATION or if the query is was supplied by the user?
  - fill() / FillWorker - I cannot pretend I have a clear understanding of all changes here. There are some refactoring of the old code and new code added to support the newly added features. I like the fact that the big method was split in more manageable pieces and appreciate the effort that was required. Some additional testing should be perform to confirm there are no regressions for this important method.
    - self member is used with dataset callbacks. We are trying not to use more BDTs as needed. Please create the handle locally, only when needed (when the CallbackData is not null);
    - line 11084: a rogue ;
    - missing javadoc in fillQuery()
    - when using ErrorManager.recordOrShowError(11885); syntax, please let the text of the error in a comment to let the reader know the text which will be presented to the user;
- CompoundQuery.java: why set the data relations a second time in retrieveImpl()? They should be already set with setDataRelation(), if any.
- DataSet.java:
  - please rename relations() to getRelations(). We use this approach only for buffers because we need to avoid collisions between the field accessors and buffer's method/attribute.
  - getRelation(): please move the method's opening curly braces ({) on the next line. Replace unwrapBuffer() with getResource().
  - getTopBufferList(): if condition: chop each operand of && on a new line and left align them if they do not fit in 110 char limit;
- Persistence.java:
  - line 4063, 4435: methods are commented.
  - QueryKey c'tor: toString is used primarily for debugging and not the best data for a key component. Maybe use the DataRelation's whereString instead?
- P2JH2Dialect.java: "currval" is dropped from reserved keywords list.
- SQLQuery.java:
  - setSessionVar: (at the end) shouldn't be setter.setParameter(pstmt, i, val, false); and pstmt.setObject(i, null); ?
  - setSessionVars: unused variable nbnd;
- Session.java: has only a H entry, no actual code changes
- JsonExport.java: childFieldsByChild is a bit misleading, they are the dropped fields so they are used to skip pieces from output;
- XmlExport.java / XmlImport.java: it seems like the namespace fixes took more then the implementation of the new things. There are too many changes to be able to fully analyse them in meld. What stood up was the "Auto-generated catch block" at line 2071, a shift in indentation between lines 1633 and 1669, but there are not big issues. I guess testing the update against big project which intensively use xml serialization and deserialization will surface any regression.

- general:
  - I think DataRelation is now too aggressively passed around as parameter to queries, their components and results. I understand that some information need to reach FqlToSqlConverter in order to apply some transformations, but I am not sure this is the right approach. I imagine

it would be cleaner if we generate the full fql with recursive syntax directly from the fill methods and let FqlToSqlConverter identify the case from the context and do a normal conversion?
- please move and group your H entries at the end of the list in a single entry and add it a new number. The branch is based on trunk and not a common branch as it was 3821c;
- some lines are chopped to 90 char even if the original code were longer. We have a 110 char per line limit. More than that, the parameters are not aligned. Here is one example:

```
                List<DataRelation> relations = dataSet.getRelations(selfBuf, true, false,
                        true);
```

What a big collection of changes. Good job!

#188 - 03/28/2023 02:12 AM - Igor Skornyakov

- Assignee changed from Ovidiu Maxiniuc to Igor Skornyakov

- Status changed from Review to WIP

#189 - 03/28/2023 02:30 AM - Igor Skornyakov

Branch 6444a was rebase to the trunk rev. 14518.

Pushed up to revision 14541.

#190 - 03/28/2023 07:11 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Review of 6444a/14538:
>
> - AbstractQuery.java, CompoundComponent.java, Dialect.java, etc: GCD coding standard: please move the method's opening curly braces ({) on the next line for g/setDataRelation() methods.

fixed

- BufferImpl.java:
  - please define PK as Session.PK (or maybe better use this final public field. It is computed at runtime as opposed to SchemaConfig.schemaConfig which is used at conversion time;

done

- you reverted Constantin's work on lazy callback (20230110);

restored

- querySynchronize() and synchronize(DataRelation rel):
  - will throw NPE if the buffer is not a DataSet component;

fixed

- local variable relation is not used;

fixed

- please use the new, faster rel.getChildBufferNative() instead of rel.getChildBuffer().unwrapBuffer(). Generally, if you **know** the expected resource type, please use getResource() instead of unwrap...().

fixed

- according to https://docs.progress.com/bundle/openedge-prodatasets-117/page/Using-the-SYNCHRONIZE-method.html, method should position the query OFF-END and the GET-FIRST should be called by the ABL programmer. More than that I am no sure creating a new dynamic query is the proper way to implement this. What if the relation is a PARENT-ID-RELATION or if the query is was supplied by the user?

Indeed, I've tested SYNCHRONIZE support only with one simple test. The problem is that the OE documentation describes the functionality in a very unclear (for me) way and I was unable to find ant examples. May be you or Marian's team can provide more tests?

- fill() / FillWorker - I cannot pretend I have a clear understanding of all changes here. There are some refactoring of the old code and new code added to support the newly added features. I like the fact that the big method was split in more manageable pieces and appreciate the effort that was required. Some additional testing should be perform to confirm there are no regressions for this important method.

I've tested it using tests which are committed to the xfer.goldencode.com/opt/testcases/ (see #6444-183). May be somebody with better OE knowledge can suggest more tests?

- self member is used with dataset callbacks. We are trying not to use more BDTs as needed. Please create the handle locally, only when needed (when the CallbackData is not null);

Sorry, I do not understand. self is used even when callbacks is null. See e.g. (from the trunk)

```
                if (!dataSet.invokeCallback(self, callbacks == null ? null : callbacks.get(BEFORE_ROW_FILL_EVE
NT)))
                {
                   fillOk.assign(false);
                   BlockManager.leave();
                }

                this.create();
```

Is it really bad to created a BDT instance which may not be used?

- line 11084: a rogue ;

fixed

- missing javadoc in fillQuery()

fixed

- when using ErrorManager.recordOrShowError(11885); syntax, please let the text of the error in a comment to let the reader know the text which will be presented to the user;

fixed

- CompoundQuery.java: why set the data relations a second time in retrieveImpl()? They should be already set with setDataRelation(), if any.

In setDataRelation we call comp.setDataRelation only if relation is null. This is a tricky place indeed and I've tested it thoroughly.

- DataSet.java:
  - please rename relations() to getRelations(). We use this approach only for buffers because we need to avoid collisions between the field accessors and buffer's method/attribute.
  - getRelation(): please move the method's opening curly braces ({) on the next line. Replace unwrapBuffer() with getResource().
  - getTopBufferList(): if condition: chop each operand of && on a new line and left align them if they do not fit in 110 char limit;
- Persistence.java:
  - line 4063, 4435: methods are commented.
  - QueryKey c'tor: toString is used primarily for debugging and not the best data for a key component. Maybe use the DataRelation's whereString instead?

In setDataRelation we call comp.setDataRelation only if relation is null. This is a tricky place indeed and I've tested it thoroughly.

- P2JH2Dialect.java: "currval" is dropped from reserved keywords list.

In setDataRelation we call comp.setDataRelation only if relation is null. This is a tricky place indeed and I've tested it thoroughly.

- SQLQuery.java:
  - setSessionVar: (at the end) shouldn't be setter.setParameter(pstmt, i, val, false); and pstmt.setObject(i, null); ?

the code looks correct and it works correct. In fact setter != null only if val != null and has supported data type. Added a comment.

- setSessionVars: unused variable nbnd;

fixed

- Session.java: has only a H entry, no actual code changes

fixed

- JsonExport.java: childFieldsByChild is a bit misleading, they are the dropped fields so they are used to skip pieces from output;

I think that it is a matter of taste. My name desribes what the variable holds instead of how it is used. I've augemented the Javadoc for this field.

- XmlExport.java / XmlImport.java: it seems like the namespace fixes took more then the implementation of the new things. There are too many changes to be able to fully analyse them in meld. What stood up was the "Auto-generated catch block" at line 2071, a shift in indentation between lines 1633 and 1669, but there are not big issues. I guess testing the update against big project which intensively use xml serialization and deserialization will surface any regression.

I've tested this pretty extensively.

- general:
  - I think DataRelation is now too aggressively passed around as parameter to queries, their components and results. I understand that some information need to reach FqlToSqlConverter in order to apply some transformations, but I am not sure this is the right approach. I imagine it would be cleaner if we generate the full fql with recursive syntax directly from the fill methods and let FqlToSqlConverter identify the case from the context and do a normal conversion?

I agree that it doesn't look good. Sorry, but I do not understand your suggestion. We actually discussed this in #6444-56 and #6444-64. However, I understand that wee agreed that the required changes will be massive albeit pretty strightforward, so I decided to use the implemented approach. Please note that it was extensively tested and any significant re-work will take some time.

- please move and group your H entries at the end of the list in a single entry and add it a new number. The branch is based on trunk and not a common branch as it was 3821c;

done

- some lines are chopped to 90 char even if the original code were longer. We have a 110 char per line limit. More than that, the parameters are not aligned. Here is one example:
  [...]

I've fixed this in the places I've noticed. Hopefully it is not very critical.

> What a big collection of changes. Good job!

Thank you!

Committed to 6444a/14542.

What I have to do now before merging the changes to the trunk?
Thank you.

#191 - 03/28/2023 07:37 PM - Ovidiu Maxiniuc

Igor Skornyakov wrote:

- according to https://docs.progress.com/bundle/openedge-prodatasets-117/page/Using-the-SYNCHRONIZE-method.html, method should position the query OFF-END and the GET-FIRST should be called by the ABL programmer. More than that I am no sure creating a new dynamic query is the proper way to implement this. What if the relation is a PARENT-ID-RELATION or if the query is was supplied by the user?

Indeed, I've tested SYNCHRONIZE support only with one simple test. The problem is that the OE documentation describes the functionality in a very unclear (for me) way and I was unable to find ant examples. May be you or Marian's team can provide more tests?

True, the information on this is scarce. I will do a minimal set of tests, but the final tests for xfer suite will probably need to be developed by Marian.

- fill() / FillWorker - [...]

I've tested it using tests which are committed to the xfer.goldencode.com/opt/testcases/ (see #6444-183). May be somebody with better OE knowledge can suggest more tests?

I will test with some large customer applications and see how the new code works. This will also cover some part of XML de-/serialization, too.

- self member is used with dataset callbacks. We are trying not to use more BDTs as needed. Please create the handle locally, only when needed (when the CallbackData is not null);

Sorry, I do not understand. self is used even when callbacks is null. See e.g. (from the trunk)
[...]
Is it really bad to created a BDT instance which may not be used?

Form the instrumentations Constantin has done recently this seems true. So we are trying to avoid them, if possible, using Java native data types.

- JsonExport.java: childFieldsByChild is a bit misleading, they are the dropped fields so they are used to skip pieces from output;

I think that it is a matter of taste. My name desribes what the variable holds instead of how it is used. I've augemented the Javadoc for this field.

That is better, thank you.

- general:
  - please move and group your H entries at the end of the list in a single entry and add it a new number. The branch is based on trunk and not a common branch as it was 3821c;

done

Actually, there are more (AbstractQuery.java, AdaptiveQuery.java, BufferImpl.java, CompoundQuery.java, etc). If you use bzr diff -r14519 --using meld on 6444a branch you may spot these easier and fix on the spot. The H entries must be in proper order when they are merged into trunk.

> What I have to do now before merging the changes to the trunk?

Please update the H entries first. I will do some tests regarding the synchronize, fill and serialization, including the customer projects. If everything goes smooth, the branch will be pushed up on the commit waiting list.

#192 - 03/29/2023 12:23 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Please update the H entries first. I will do some tests regarding the synchronize, fill and serialization, including the customer projects. If everything goes smooth, the branch will be pushed up on the commit waiting list.

Done.
Committed to 6444a/14544.

#193 - 03/29/2023 04:09 PM - Eric Faulhaber

Ovidiu, please post when you have some test results, so we can figure out where this falls in the merge order. Thanks.

#194 - 03/30/2023 12:15 AM - Ovidiu Maxiniuc

Igor Skornyakov wrote:

> Ovidiu Maxiniuc wrote:
>
> > Please update the H entries first. I will do some tests regarding the synchronize, fill and serialization, including the customer projects. If everything goes smooth, the branch will be pushed up on the commit waiting list.
>
> Done.
> Committed to 6444a/14544.

Actually, not what I meant by updating the history entries.
Your entry must be the last one, with a new H number assigned; all changes, even if the dates are different must be grouped. Do not re-number the H entries from previous commits (unless the rare case when the author increment it wrongly). Do not insert the individual notes in the date order among other entries. The full branch will be merged in trunk as a single VCS-node and the H entries should reflect that. In other words: one H entry per merge operation in trunk.

Eric Faulhaber wrote:

> Ovidiu, please post when you have some test results, so we can figure out where this falls in the merge order. Thanks.

I tested the branch manually, with several VM scenarios and I had only one failure. Which I do not have a confirmation whether it was caused by this branch or other regression in trunk.
For the other large customer POC project I created a changeset by applying branch 6733a over 6444a. There were no regression with this branch.

At the same time, I used isolated testcases to test for regressions the fill (without recursion), and serializations. The fill was OK. There were some minor differences (additional attributes) in xml-schema. The JSON decimal numbers serialization is regressed, they are always saved rounded to the first decimal (Ex: 11.2 instead of 11.16).

#195 - 03/30/2023 01:35 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Actually, not what I meant by updating the history entries.
> Your entry must be the last one, with a new H number assigned; all changes, even if the dates are different must be grouped. Do not re-number the H entries from previous commits (unless the rare case when the author increment it wrongly). Do not insert the individual notes in the date order among other entries. The full branch will be merged in trunk as a single VCS-node and the H entries should reflect that. In other words: one H entry per merge operation in trunk.

Do you mean that I have to move my H entries to the end of the list even if they where done before the other changes (e.g. a months ago)?

> Eric Faulhaber wrote:
>
> > Ovidiu, please post when you have some test results, so we can figure out where this falls in the merge order. Thanks.
>
> I tested the branch manually, with several VM scenarios and I had only one failure. Which I do not have a confirmation whether it was caused by this branch or other regression in trunk.
>
> For the other large customer POC project I created a changeset by applying branch 6733a over 6444a. There were no regression with this branch.
> At the same time, I used isolated testcases to test for regressions the fill (without recursion), and serializations. The fill was OK. There were some minor differences (additional attributes) in xml-schema. The JSON decimal numbers serialization is regressed, they are always saved rounded to the first decimal (Ex: 11.2 instead of 11.16).

I've intentionally changed decimal serialization to be compatible with OE since it was not. So it is not a regression.

#196 - 03/30/2023 12:22 PM - Igor Skornyakov

Branch 6444a was rebased to the trunk rev. 14523.
Pushed up to revision 14548.

#197 - 03/30/2023 12:23 PM - Constantin Asofiei

Igor, please specify a testcase I can use for the decimal serialization - where you saw that the rounding is necessary.

#198 - 03/30/2023 12:27 PM - Igor Skornyakov

Constantin Asofiei wrote:

> Igor, please specify a testcase I can use for the decimal serialization - where you saw that the rounding is necessary.

See e.g. tt-marshal.p in xfer.goldencode.com/opt/testcases/6453-6444.

#199 - 03/30/2023 12:39 PM - Constantin Asofiei

This test uses appserver calls - the data may be affected by the transfer over the appserver. Do you have a test which just uses WRITE-XML for a temp-table with some records?

#200 - 03/30/2023 12:39 PM - Constantin Asofiei

Constantin Asofiei wrote:

> Do you have a test which just uses WRITE-XML for a temp-table with some records?

I mean WRITE-JSON

#201 - 03/30/2023 12:40 PM - Constantin Asofiei

For example, this test:

```
def temp-table tt1 field f1 as dec decimals 4.

create tt1.
tt1.f1 = 1234.5678.
temp-table tt1:write-json("file", "a.json").
```

creates this:

```
{"tt1":[{"f1":1234.5678}]}
```

#202 - 03/30/2023 12:54 PM - Igor Skornyakov

- File json.p added

Well, in fact I've used the attached test for WRITE-JSON.

#203 - 03/30/2023 01:08 PM - Constantin Asofiei

If DECIMALS is not specified, it defaults to 10 in OE. In FWD, there may be a problem as we assume '0' in persist.orm.Property.scale (and based on this sometimes we default to 10 and sometimes not).

I agree that JSON looks like it honors the DECIMALS option, but current changes don't look correct.

#204 - 03/30/2023 01:12 PM - Igor Skornyakov

Constantin Asofiei wrote:

> If DECIMALS is not specified, it defaults to 10 in OE. In FWD, there may be a problem as we assume '0' in persist.orm.Property.scale (and based on this sometimes we default to 10 and sometimes not).
>
> I agree that JSON looks like it honors the DECIMALS option, but current changes don't look correct.

Sorry, I do understand what is wrong with this change. W/o it FWD WRITE-JSON output was different from OE and with it it is the same.
Of course it is very easy to revert this change. Are you sure that it makes sense?
Thank you.

#205 - 03/30/2023 01:13 PM - Constantin Asofiei

Igor Skornyakov wrote:

> Constantin Asofiei wrote:
>
> > If DECIMALS is not specified, it defaults to 10 in OE. In FWD, there may be a problem as we assume '0' in persist.orm.Property.scale (and based on this sometimes we default to 10 and sometimes not).
> >
> > I agree that JSON looks like it honors the DECIMALS option, but current changes don't look correct.
>
> Sorry, I do understand what is wrong with this change. W/o it FWD WRITE-JSON output was different from OE and with it it is the same.

We need Ovidiu's test first. Also, try the test in #6444-201 with and without DECIMALS and post here the output.

#206 - 03/30/2023 02:20 PM - Ovidiu Maxiniuc

Here is the isolated testcase:

```
DEFINE TEMP-TABLE tt-1src /* the source for tt-1 */
   FIELD g11 AS CHAR
   FIELD g12 AS INT
   FIELD g13 AS DATE
   FIELD f14 AS DECIMAL
   /*INDEX gx IS UNIQUE g12 f14*/.

CREATE tt-1src. g11 = "A". g12 = 11. g13 = 11/16/66. tt-1src.f14 = 11.16.
CREATE tt-1src. g11 = "B". g12 = 15. g13 = 06/15/55. tt-1src.f14 =  6.15.
CREATE tt-1src. g11 = "C". g12 = 17. g13 = 02/17/99. tt-1src.f14 = 17.02.
CREATE tt-1src. g11 = "D". g12 = 23. g13 = 03/14/15. tt-1src.f14 = 3.1415926.
CREATE tt-1src. g11 = "E". g12 = 31. g13 = 03/18/06. tt-1src.f14 = 5.
CREATE tt-1src. g11 = "C". g12 = 17. g13 = 02/17/99. tt-1src.f14 = 17.02.
RELEASE tt-1src.

BUFFER tt-1src:WRITE-JSON("file", "\tmp\tt-1src.json", TRUE).
```

And the results are as follows:

Reference (4GL)	FWD (6444a)

{"tt-1src": [
  {
  "g11": "A",
  "g12": 11,
  "g13": "1966-11-16",
  "f14": 11.16
  },
  {
  "g11": "B",
  "g12": 15,
  "g13": "1955-06-15",
  "f14": 6.15
  },
  {
  "g11": "C",
  "g12": 17,
  "g13": "1999-02-17",
  "f14": 17.02
  },
  {
  "g11": "D",
  "g12": 23,
  "g13": "2015-03-14",
  "f14": 3.1415926
  },
  {
  "g11": "E",
  "g12": 31,
  "g13": "2006-03-18",
  "f14": 5.0
  },
  {
  "g11": "C",
  "g12": 17,
  "g13": "1999-02-17",
  "f14": 17.02
  }
]}
	{
	  "tt-1src": [{
	  "g11": "A",
	  "g12": 11,
	  "g13": "1966-11-16",
	  "f14": 11.2
	  }, {
	  "g11": "B",
	  "g12": 15,
	  "g13": "1955-06-15",
	  "f14": 6.2
	  }, {
	  "g11": "C",
	  "g12": 17,
	  "g13": "1999-02-17",
	  "f14": 17.0
	  }, {
	  "g11": "D",
	  "g12": 23,
	  "g13": "2015-03-14",
	  "f14": 3.1
	  }, {
	  "g11": "E",
	  "g12": 31,
	  "g13": "2006-03-18",
	  "f14": 5.0
	  }, {
	  "g11": "C",
	  "g12": 17,
	  "g13": "1999-02-17",
	  "f14": 17.0
	  }]
	}

#207 - 03/30/2023 02:22 PM - Igor Skornyakov

Constantin Asofiei wrote:

> We need Ovidiu's test first. Also, try the test in #6444-201 with and without DECIMALS and post here the output.

I've testedv this and found that we see only 2 digits after dot while with 4GL we get 4. However, the problem is not with my fix but somewhere else. The column value **1234.5678** comes to writeDatum as **1234.57**

#208 - 03/30/2023 02:28 PM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Here is the isolated testcase:
> [...]
> And the results are as follows:

Reference (4GL)	FWD (6444a)
[...]	[...]

Please see #6444-207. The problem seems to be **not** in writeDatum.

Sorry, I'm already 12+ hours at my desk. Lets continue tomorrow.

#209 - 03/30/2023 03:04 PM - Constantin Asofiei

The 'scale' computed in int colDecs = Math.max(1, column.getDecimals()); results in '0' so colDecs becomes 1, in Ovidiu's test. The datum at the row has the correct scale - 10 in Ovidiu's case.

If you want to force the decimals, use the decimal.scale at the lambda's d argument - the datum will always have the scale properly set, you don't need to interpret it from the column.

#210 - 03/30/2023 03:15 PM - Igor Skornyakov

Constantin Asofiei wrote:

> The 'scale' computed in int colDecs = Math.max(1, column.getDecimals()); results in '0' so colDecs becomes 1, in Ovidiu's test. The datum at the row has the correct scale - 10 in Ovidiu's case.
>
> If you want to force the decimals, use the decimal.scale at the lambda's d argument - the datum will always have the scale properly set, you don't need to interpret it from the column.

Indeed, after using

```
            fn = (d) -> json.numberValue(
                    ((decimal) d).toBigDecimal().setScale(((decimal) d).getPrecision(), RoundingMode
.HALF_UP));
```

I see the same output as with OE.
Thank you!

Committed to 6444a/14549.

#211 - 03/30/2023 04:32 PM - Igor Skornyakov

Igor Skornyakov wrote:

> Ovidiu Maxiniuc wrote:
>
> > Actually, not what I meant by updating the history entries.
> > Your entry must be the last one, with a new H number assigned; all changes, even if the dates are different must be grouped. Do not re-number the H entries from previous commits (unless the rare case when the author increment it wrongly). Do not insert the individual notes in the date order among other entries. The full branch will be merged in trunk as a single VCS-node and the H entries should reflect that. In other words: one H entry per merge operation in trunk.
>
> Do you mean that I have to move my H entries to the end of the list even if they where done before the other changes (e.g. a months ago)?

Ovidiu,
Please confirm if my understanding is correct (and I have to change the H entry date if I have to move it to the end of the list).
Thank you.

#212 - 03/30/2023 07:29 PM - Ovidiu Maxiniuc

Yes, That is correct. The time-stamps remain, but they need to be grouped together at the end of history list, under a new H-id.

Although it makes sense to have them sorted by date, at the moment the branch is merged, all your changes, regardless of the moment they were done, will share a single revision number in bzr. There is a relation 1-to-1 between the merges in trunk and H entries for each file.

Note that since you are the last one adding content to a file, you are responsible of fixing any conflicts which may occur with other changes in a parallel branch already merged in. For that, your original description messages may alter or even become obsolete (for example, if you added a fix for a feature which was dropped in a parallel branch).

More briefly: it does not really matter the date when a change was added to your branch but the moment it was merged to trunk.

#213 - 03/31/2023 02:03 AM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Yes, That is correct. The time-stamps remain, but they need to be grouped together at the end of history list, under a new H-id.
>
> Although it makes sense to have them sorted by date, at the moment the branch is merged, all your changes, regardless of the moment they were done, will share a single revision number in bzr. There is a relation 1-to-1 between the merges in trunk and H entries for each file.
>
> Note that since you are the last one adding content to a file, you are responsible of fixing any conflicts which may occur with other changes in a parallel branch already merged in. For that, your original description messages may alter or even become obsolete (for example, if you added a fix for a feature which was dropped in a parallel branch).
>
> More briefly: it does not really matter the date when a change was added to your branch but the moment it was merged to trunk.

Got it, thank you!
I will update history entries right before the merge since it is unclear when it will happen and how the history of the files will look at this moment.

#214 - 04/05/2023 02:00 AM - Igor Skornyakov

- Related to Bug #7247: Dataset:READ-XML does nothing added

#215 - 04/05/2023 02:25 AM - Igor Skornyakov

Branch 6444a was rebased to the trunk rev. 14525.
Pushed up to revision 14551.

#216 - 04/05/2023 02:11 PM - Igor Skornyakov

Updated history entries.
Committed to 6444a/14552.

Changes to the rules/gaps/expressions.rules are also committed.

#217 - 04/05/2023 03:29 PM - Ovidiu Maxiniuc

Review of 6444a/14552.

expressions.rules: please update the gap status for KW_SYNCHRON ("synchronize"), KW_F_KEY_H ("foreign-key-hidden"), and KW_REST_ROW

("restart-row"). Set the number (55) for the H entry.

I see in some files where the history entry were grouped together that you replaced their timestamps with current date. That is OK, but you could have kept the original timestamps. They reflect the date of the change inside the branch/merge, not globally.

Please merge after updating the expressions.rules with the rest of implemented attributes.

#218 - 04/05/2023 03:45 PM - Igor Skornyakov

Ovidiu Maxiniuc wrote:

> Review of 6444a/14552.
>
> expressions.rules: please update the gap status for KW_SYNCHRON ("synchronize"), KW_F_KEY_H ("foreign-key-hidden"), and KW_REST_ROW ("restart-row"). Set the number (55) for the H entry.

Done.

> I see in some files where the history entry were grouped together that you replaced their timestamps with current date. That is OK, but you could have kept the original timestamps. They reflect the date of the change inside the branch/merge, not globally.

Sorry, I've misunderstood you.

> Please merge after updating the expressions.rules with the rest of implemented attributes.

Merged to the trunk rev. 14526.

#219 - 04/05/2023 04:00 PM - Eugenie Lyzenko

Igor Skornyakov wrote:

> Ovidiu Maxiniuc wrote:
>
> > Review of 6444a/14552.

> expressions.rules: please update the gap status for KW_SYNCHRON ("synchronize"), KW_F_KEY_H ("foreign-key-hidden"), and KW_REST_ROW ("restart-row"). Set the number (55) for the H entry.

Done.

> I see in some files where the history entry were grouped together that you replaced their timestamps with current date. That is OK, but you could have kept the original timestamps. They reflect the date of the change inside the branch/merge, not globally.

Sorry, I've misunderstood you.

> Please merge after updating the expressions.rules with the rest of implemented attributes.

Does it mean the conversion is required/desired.

#220 - 04/05/2023 04:02 PM - Ovidiu Maxiniuc

The changes in expressions.rules do not affect the code generated by the conversion.

#221 - 04/10/2023 07:02 AM - Greg Shah

From Sergey, in regard to the merged changes from 6444a:

> It seems <customer_application_with_hand_coded_persistence_usage> is not compatible with these API changes because scroll/list methods were changed. DataRelation parameter was added. Please explain how to replace old Persistence.scroll and list usages with new ones.
>
> If this parameter can be replaced with null value, then it makes sense to preserve these old methods for compatibilities with the existing code, otherwise it needs to change accordingly.

#222 - 04/10/2023 07:03 AM - Greg Shah

From Igor:

> An additional DataRelation parameter is used only for the second pass of the FILL operation. In all other situations it should be null. It really affects only the SQL generation.

#223 - 04/10/2023 07:05 AM - Greg Shah

The application Sergey is talking about has no datasets, no data relations. I would expect that usage of those APIs would indeed be more common for the non-dataset cases. Can we please restore the original signatures and implement a safe null parameter in this case? It would avoid changing all the hand-written application code and it seems like it would be better for the simple non-dataset use cases.

#224 - 04/10/2023 07:09 AM - Igor Skornyakov

Greg Shah wrote:

> The application Sergey is talking about has no datasets, no data relations. I would expect that usage of those APIs would indeed be more common for the non-dataset cases. Can we please restore the original signatures and implement a safe null parameter in this case? It would avoid changing all the hand-written application code and it seems like it would be better for the simple non-dataset use cases.

OK. I will do it right now in 6444b (just created).

#225 - 04/10/2023 08:31 AM - Igor Skornyakov

- Status changed from WIP to Review

Greg Shah wrote:

> The application Sergey is talking about has no datasets, no data relations. I would expect that usage of those APIs would indeed be more common for the non-dataset cases. Can we please restore the original signatures and implement a safe null parameter in this case? It would avoid changing all the hand-written application code and it seems like it would be better for the simple non-dataset use cases.

Added 'old' methods w/o DataRelation argument in 6444b/14528.

#226 - 04/11/2023 05:53 AM - Eric Faulhaber

Code review 6444b/14528:

Only the "old" Persistence.scroll and Persistence.list methods with the public access modifier need to be restored. Private and package private methods cannot be accessed by hand-written, application, Java code, and they are not used internally. So, the "old" versions of those methods (e.g., Persistence.getQuery, Persistence$Context.getQuery) are not needed.

If a method signature fits within 110 characters, please format it all on the same line. For example:

```
public <T> ScrollableResults<T> scroll(String fql, Object[] values, int maxResults, int startOffset)
...
```

...instead of:

```
   public <T> ScrollableResults<T> scroll(String fql, Object[] values,
             int maxResults, int startOffset)
   ...
```

With these changes, 6444b will be ready to merge to trunk.

#227 - 04/11/2023 07:17 AM - Igor Skornyakov

Eric Faulhaber wrote:

> Code review 6444b/14528:
>
> Only the "old" Persistence.scroll and Persistence.list methods with the public access modifier need to be restored. Private and package private methods cannot be accessed by hand-written, application, Java code, and they are not used internally. So, the "old" versions of those methods (e.g., Persistence.getQuery, Persistence$Context.getQuery) are not needed.
>
> If a method signature fits within 110 characters, please format it all on the same line. For example:
>
> [...]
>
> ...instead of:
>
> [...]
>
> With these changes, 6444b will be ready to merge to trunk.

Branch 6444b was merged to the trunk/14531 and archived.

#228 - 05/26/2023 01:39 PM - Eric Faulhaber

- % Done changed from 0 to 100

- Status changed from Review to Closed

#229 - 05/26/2023 02:09 PM - Eric Faulhaber

- Related to Bug #6475: double-check and fix any read/write-xml/json issues at table and dataset added

Files

ds-test.p	1.34 KB	11/30/2022	Igor Skornyakov
ds-test.p	2.55 KB	12/01/2022	Igor Skornyakov
ds.xml	564 Bytes	12/01/2022	Igor Skornyakov
ttd.xml	344 Bytes	12/01/2022	Igor Skornyakov
ds-test.p	2.85 KB	12/07/2022	Igor Skornyakov
ds-sync.p	5.5 KB	01/28/2023	Igor Skornyakov
nested-test.p	8.7 KB	03/09/2023	Igor Skornyakov
ds.xsd	4.28 KB	03/09/2023	Igor Skornyakov
ds_tt1.xsd	9.89 KB	03/09/2023	Igor Skornyakov
ds_tt3.xsd	2.08 KB	03/09/2023	Igor Skornyakov
hds_tt3.xsd	2.09 KB	03/09/2023	Igor Skornyakov
hds.xsd	4.35 KB	03/09/2023	Igor Skornyakov
hds_tt5.xsd	3.45 KB	03/09/2023	Igor Skornyakov
AdaptiveQuery.diff	1.2 KB	03/12/2023	Igor Skornyakov
recursive_relation_fill_temp.p	3.75 KB	03/13/2023	Igor Skornyakov
recursive_relation_fill_tmp.txt	106 Bytes	03/13/2023	Igor Skornyakov
recursive_relation_fill_tmp.txt	71 Bytes	03/13/2023	Igor Skornyakov
nested-fk.p	13.1 KB	03/15/2023	Igor Skornyakov
ds-nested-test-FWD.zip	28.2 KB	03/15/2023	Igor Skornyakov
ds-nested-test-4GL.zip	25.3 KB	03/15/2023	Igor Skornyakov
recursive_relation_fill_temp.patch	1.4 KB	03/16/2023	Radu Apetrii
json.p	13 KB	03/30/2023	Igor Skornyakov