Database - Feature #3809

ProDataSet support

11/25/2018 07:17 AM - Greg Shah

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

Description

Related issues:

Related to Database - Feature #3815: more temp-table options	**Closed**
Related to Database - Feature #4514: run ProDataSet test suite in FWD and res...	**WIP**
Related to Database - Feature #4397: add database attrs, methods and options	**Closed**
Related to Database - Feature #3574: finish implementation of temp-table XML ...	**Closed**
Related to Database - Feature #6444: dataset improvements	**Closed**

History

#1 - 11/25/2018 07:25 AM - Greg Shah

This task is intended to implement compatible and comprehensive support for the ProDataSet features of the 4GL. At this time, we are creating testcases to explore the following:

- Base Features
  - Creating/defining a dataset
    - Static and dynamic syntax (including CREATE-LIKE)
  - Deleting a dynamic dataset
  - Relationships between temp-tables within a dataset
    - Static and dynamic syntax
  - Data sources
    - Static and dynamic syntax
    - Persistent tables
    - Temp-tables
    - Non-database (custom FILL logic)
    - Defining with QUERY, source-buffer-phrase, or both.
    - Attaching/detaching at runtime (BUFFER-HANDLE:{ATTACH|DETACH}-DATA-SOURCE).
  - CRUD operations
    - Intra-client
    - AppServer
    - FILL operations
      - Specifying a custom query.
      - Allowing the default query to drive the FILL.
      - Using the various FILL-MODEs.
      - Deactivating Data-Relations to limit the scope of the FILL.
      - Full vs. partial FILL. FILL a specific buffer and its related children (vs. the entire dataset).
    - Updates to datasets which must be applied back to the original data sources
      - Simple (single session) case.
      - Concurrent (multi-session) case. Should be deterministically ordered, not timing-sensitive. For example, one session updates table, another session updates data from same table in a dataset.
    - Manually changing data in a temp-table involved in a dataset.
- Passing a dataset as a parameter, using all valid combinations of options:
  - By value, by reference (with and without REFERENCE-ONLY in dataset definition), bind
  - Append or not
  - INPUT, OUTPUT, INPUT-OUTPUT
  - As DATASET or DATASET-HANDLE
  - Local and remote sessions
  - Controlling how much schema information to marshal
- Events
  - Defining event procedures
  - Registering event procedures

    - Retrieving callback information (procedure context and name by event)
    - FILL events – before/after at:
        - dataset level
        - temp-table level
        - at row level
    - FILL related methods
        - CURRENT-ITERATION
        - GET-ITERATION
        - NUM-ITERATIONS
    - Non-FILL events
- Attributes and methods
    - All attributes and methods applicable to a dataset or its members, which are not already explicitly referenced above.
    - Where an attribute or method is applicable to more than one handle type (e.g., buffer, temp-table), we are only interested in use cases specific to datasets.
    - Enhanced query support for dataset buffers in QUERY-PREPARE (being able to reference related buffers directly in a predicate)

#2 - 11/25/2018 09:29 AM - Greg Shah

There are also features in non-database parts of the system.

- SESSION:FIRST-DATASET

#3 - 11/25/2018 12:29 PM - Eric Faulhaber

The following list of features is not exhaustive, but we know we need to implement at least...

Language Statements

- DEFINE DATASET
- CREATE DATASET
- CREATE DATA-SOURCE

Functions

- ROW-STATE

Attributes

- BUFFER:AFTER-BUFFER
- BUFFER:AFTER-ROWID
- BUFFER:BEFORE-ROWID
- BUFFER:ORIGIN-ROWID
- BUFFER:BEFORE-BUFFER
- BUFFER:FILL-MODE
- BUFFER:ROW-STATE
- BUFFER:DATA-SOURCE
- BUFFER:DATA-SOURCE-COMPLETE-MAP
- BUFFER:DATA-SOURCE-ROWID
- BUFFER:DATASET
- BUFFER:NUM-CHILD-RELATIONS
- BUFFER:PARENT-RELATION
- BUFFER:KEYS
- BUFFER:BATCH-SIZE

- DATA-RELATION:ACTIVE
- DATA-RELATION:CHILD-BUFFER
- DATA-RELATION:NESTED
- DATA-RELATION:PARENT-BUFFER

- DATA-RELATION:RELATION-FIELDS
- DATA-RELATION:WHERE-STRING

- DATA-SOURCE:MERGE-BY-FIELD
- DATA-SOURCE:NEXT-ROWID
- DATA-SOURCE:NUM-SOURCE-BUFFERS
- DATA-SOURCE:PREFER-DATASET
- DATA-SOURCE:RESTART-ROWID
- DATA-SOURCE:SAVE-WHERE-STRING

- DATASET:NUM-BUFFERS
- DATASET:NUM-RELATIONS
- DATASET:NUM-TOP-BUFFERS
- DATASET:RELATIONS-ACTIVE
- DATASET:SERIALIZE-HIDDEN
- DATASET:SERIALIZE-NAME
- DATASET:TOP-NAV-QUERY

- TEMP-TABLE:TRACKING-CHANGES

- SESSION:FIRST-DATASET

Methods

- BUFFER:APPLY-CALLBACK
- BUFFER:ATTACH-DATA-SOURCE
- BUFFER:DETACH-DATA-SOURCE
- BUFFER:FILL
- BUFFER:GET-CHILD-RELATION
- BUFFER:MARK-NEW
- BUFFER:ACCEPT-ROW-CHANGES
- BUFFER:REJECT-ROW-CHANGES
- BUFFER:SAVE-ROW-CHANGES
- BUFFER:SET-CALLBACK

- DATA-SOURCE:ADD-SOURCE-BUFFER
- DATA-SOURCE:GET-SOURCE-BUFFER

- DATASET:ADD-RELATION
- DATASET:COPY-DATASET
- DATASET:EMPTY-DATASET
- DATASET:CREATE-LIKE
- DATASET:FILL
- DATASET:ACCEPT-CHANGES
- DATASET:REJECT-CHANGES
- DATASET:GET-BUFFER-HANDLE
- DATASET:GET-CHANGES
- DATASET:GET-RELATION
- DATASET:GET-TOP-BUFFER
- DATASET:SET-CALLBACK
- DATASET:READ-XML
- DATASET:READ-JSON
- DATASET:WRITE-JSON
- DATASET:WRITE-XML
- DATASET:WRITE-XMLSCHEMA

#4 - 11/25/2018 01:59 PM - Eric Faulhaber

- Related to Feature #3815: more temp-table options added

#5 - 11/25/2018 02:37 PM - Eric Faulhaber

Dataset parameter options:

- BY-REFERENCE
- APPEND
- BIND

#6 - 03/20/2019 12:40 PM - Eric Faulhaber

- Status changed from New to WIP

- Assignee set to Ovidiu Maxiniuc

#7 - 03/20/2019 02:38 PM - Eric Faulhaber

I have created task branch 3809a for this work. It is based on trunk 11301, but I think that's ok. I don't know of any dependencies on 3750b and we'll pick up those changes shortly anyway, when that branch is merged to trunk.

#8 - 03/20/2019 06:20 PM - Ovidiu Maxiniuc

I tried to convert the provided testcases an I encountered some unknown token: assert-true, assert-err, assert-not-err.

How am I supposed to handle these?

#9 - 03/20/2019 06:29 PM - Eric Faulhaber

These are procedures defined in dataset/log_hlp.i or functions defined in dataset/help/log_hlp.i. I'm not sure which include file your test case uses.

#10 - 03/21/2019 03:08 PM - Eric Faulhaber

As of 3750b/11512, the parser is missing token-types for the following dataset-related features:

```
current-query
foreign-key-hidden
maximum-level
parent-fields-after
parent-fields-before
restart-row
```

#11 - 03/22/2019 08:22 AM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> As of 3750b/11512, the parser is missing token-types for the following dataset-related features:

I fixed the Progress syntax for these. Some of them are new, some did not have the type properly declared. I will commit the update immediately.
I encountered Dataset.Events.Handler.EventsHandler class. I cannot find it anywhere, so I am adding the minimum interface to class skeletons.

#12 - 03/22/2019 08:25 AM - Eric Faulhaber

Ovidiu, how safe are these changes?

Greg, Constantin, is it ok to commit this to 3750b, or are we only allowing performance and regression fixes in that branch? We can't use 3809a for parsing test cases at this time, as it is missing the OO functionality we need in 3750b.

#13 - 03/22/2019 08:29 AM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> Ovidiu, how safe are these changes?

They are only token declarations. However, since the token IDs change, a full reconversion is required since some tokens are shifted.

BTW, I have just found EventsHandler. Is in the PROPATH, but I do not understand why it is not loaded. Investigating.

#14 - 03/22/2019 08:35 AM - Constantin Asofiei

Eric Faulhaber wrote:

> Ovidiu, how safe are these changes?
>
> Greg, Constantin, is it ok to commit this to 3750b, or are we only allowing performance and regression fixes in that branch? We can't use 3809a for parsing test cases at this time, as it is missing the OO functionality we need in 3750b.

As Ovidiu mentioned, 3750b will be unusable until reconversion, if you place these changes there. I'm not against placing changes there, but lets do this in large batches, and not for each token (and commit on EOD, so that we can reconvert over night...).

#15 - 03/22/2019 03:16 PM - Constantin Asofiei

Greg/Eric, can we postpone committing Ovidiu's changes until Monday?

#16 - 03/22/2019 03:21 PM - Eric Faulhaber

Constantin Asofiei wrote:

> Greg/Eric, can we postpone committing Ovidiu's changes until Monday?

Yes.

Ovidiu, please post the patch here so I will be able to use your changes in the meantime.

#17 - 03/22/2019 03:29 PM - Ovidiu Maxiniuc

- File progress_g&SymbolResolver_java.diff added

Here it is, attached.

I will commit my changes for ProDataSet conversion to 3809a.

#18 - 03/22/2019 04:54 PM - Ovidiu Maxiniuc

Ovidiu Maxiniuc wrote:

> I will commit my changes for ProDataSet conversion to 3809a.

I added conversion support and signature in interfaces for following methods and attributes: kw_rejected, kw_att_dsrc, kw_get_chg, kw_merge_ch, kw_empty_ds, kw_trac_chg, kw_b4_table, kw_aft_tbl , kw_fill , kw_xml_ntyp, kw_serialzh, kw_num_ref, kw_data_sm, kw_get_rel, kw_top_navq, kw_get_topb, kw_rels_act, kw_num_rel , kw_active , kw_xml_nnam, kw_error , kw_num_topb.

On a second thought, I am not sure whether I should commit to this branch. The truth is that I worked over 3750b. I think I will create a 3809b, based on 3750b which should be quite stable for now and I think I can keep the pace with it.

#19 - 03/22/2019 04:57 PM - Eric Faulhaber

My intention was to rebase 3809a when 3750b was merged to trunk, because I didn't think we had dependencies on 3750b for our initial work (other than parsing the test suite). But if you found you do have dependencies on 3750b, you can create a new branch.

#20 - 03/22/2019 05:01 PM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> My intention was to rebase 3809a when 3750b was merged to trunk, because I didn't think we had dependencies on 3750b for our initial work (other than parsing the test suite). But if you found you do have dependencies on 3750b, you can create a new branch.

I understand. The problem was related to some classes. The trunk lacks some fixed that were done in 3750b. I will delay the commit until Monday. I don't think the 3750b will be merged into the trunk by then but we will have a better view of the situation.

#21 - 03/28/2019 02:04 AM - Eric Faulhaber

I've previously created a class called DataSource for use with XML and JSON serialization. However, this name already will have a different meaning to 4GL developers in the context of ProDataSets. We could rename the existing class to SourceData maybe (and the corresponding DataTarget class

to TargetData for consistency). Conversion rules which emit the current names will have to change, of course.

#22 - 03/29/2019 07:17 AM - Ovidiu Maxiniuc

I did the refactoring as you suggested. I think it's a good solution.

I am working now on the conversion of static definitions of datasets, data sources and data relations. I encountered some issues with the syntax, but I fixed this in progress.g. I also need to create my own (isolated) testcases. The ones provided do not cover (by far) all forms of syntax for these statements. I think they are more runtime-oriented that for syntax/conversion.

The dynamic creation for these objects seems to be fine, from POV of conversion.

A few words on the future implementation of these new items. The datasets can be shared. This is very similar to temp-tables. As you remember, the resolution and matching of shared temp-tables with their prototype was a bit difficult. The temp-tables can be also passed as arguments and both sides of the call need to match, although we notice some really strange behaviour for edge test cases. I am not yet aware how much similar the datasets are in this regard. I decided to generate these definitions in their Java sources and a manager will be responsible for matching the shared (and possible parameter passing) items. I do not see much advantage of defining independent classes / interfaces for each defined dataset. The construction is very similar with dynamic creation and will share the same code.

The latest revision of branch 3809b is 11565.

#23 - 04/01/2019 02:51 PM - Ovidiu Maxiniuc

Committed r11576 of branch 3809b.

At this moment all the sources provided convert. However, the code is not 100% compilable. I could spot a parent-id-relation option I missed to fully implement and a case when a dataset was defined both as static and dynamic (same identifier). I do not know if this is possible, I guess yes, because they are in different name-spaces.

I encountered a problem. My test virtual machine stopped working after latest OP update. I updated it to very latest version but with no luck.

#24 - 04/02/2019 03:16 PM - Ovidiu Maxiniuc

Revision 11577.

Added missing parent-id-relation option and resolved conflicts with other variables.
Compiling the project I discovered the new class DataSet lacks the parameter support. I started working on this - most likely this is similar to TABLE PARAMETERS - but not yet functional.

#25 - 04/03/2019 03:44 PM - Ovidiu Maxiniuc

Revision 11583.

The project converts and generated sources compiles without errors.
I am going tomorrow to check missing methods/attributes, update gaps, document the stubs.

#26 - 04/04/2019 04:59 PM - Ovidiu Maxiniuc

Revision 11584.

Added missing method declarations. Added attributes and methods from existing interfaces. Fixed sound issues. Started adding javadocs.
There is only one major issue: the matching signatures for some heavy overloaded methods. I guess most of them if not all are related to XML persistence. Beside the classic String (literal) / character (expression) issue, the provided testcases also uses ? for some parameters. This is converted in many cases into new unknown(), which makes all defined signature invalid.

#27 - 04/04/2019 05:02 PM - Eric Faulhaber

[Originally sent via email, but I wanted to document here...]:

Please consider the recent refactoring I did with BUFFER-COPY and BUFFER-COMPARE. I would much rather follow this chaining model where it makes sense, than to have exponentially growing API signature variants. It can make the converted code a bit more verbose, but I think it makes that code much easier to understand, compared to viewing a long list of parameters passed to a method, where you have to review the javadoc to know what is what. It is especially useful for one-time operations which can be encapsulated into a class, or to define an object which has a large variety of configuration options.

#28 - 04/04/2019 11:16 PM - Eric Faulhaber

Eric Faulhaber wrote:

> Please consider the recent refactoring I did with BUFFER-COPY and BUFFER-COMPARE.

BTW, I'm referring to the language statement forms, not the handle method forms.

Conversion support primarily is in:

- rules/annotations/database_general.rules
- rules/convert/buffer_copy_compare.rules

Public runtime API is in:

- src/com/goldencode/p2j/persist/BufferCompare.java
- src/com/goldencode/p2j/persist/BufferCopy.java

RecordBuffer does the grunt work behind the public API.

#29 - 04/05/2019 01:37 PM - Ovidiu Maxiniuc

Revision 11587.

All provided files in Test Suite converted and compiled successfully. I updated the project and wiki with minimal information to get the project

Because the lack of information at conversion time a special signature was added for the overloaded method.
The methods that have more then 3 parameters were forced to wrap the literal parameters so the overload forms are kept to a minimum.

I haven't used the idiom from BUFFER-COPY and BUFFER-COMPARE. However, the DataSet and DataSource static definition do use a similar construct so that they are defined by a builder that chains the calls in order to accommodate the full set of (optional) parameters.

#30 - 04/05/2019 04:26 PM - Ovidiu Maxiniuc

Revision 11591.

Two important changes in converted code: * the static definition of DataSet was using buffer field values. We are not interested in such things, in fact, at that moment the buffers are not even open. Switched to emit the legacy names of the fields instead which can be used later by the relation. * the PARENT-FIELDS-BEFORE and PARENT-FIELDS-AFTER were not optional and the absence of either one generated invalid code.

#31 - 04/08/2019 06:03 AM - Constantin Asofiei

Ovidiu, please post a small snapshot of how DATASET and DATASET-HANDLE parameters will convert.

Also, I assume that for a DATASET-HANDLE parameter, the caller can pass any handle instance, right?

#32 - 04/08/2019 10:31 AM - Ovidiu Maxiniuc

Consider the following ABL code:

```
PROCEDURE test-ds-handle:
    DEFINE INPUT PARAMETER dshi AS HANDLE NO-UNDO.
    DEFINE OUTPUT PARAMETER dsho AS HANDLE NO-UNDO.
    DEFINE INPUT-OUTPUT PARAMETER dshu AS HANDLE NO-UNDO.

    DEFINE INPUT PARAMETER DATASET FOR dsmaster.
    DEFINE OUTPUT PARAMETER DATASET FOR dsrefmaster.
    DEFINE INPUT-OUTPUT PARAMETER DATASET FOR dsorder.

   MESSAGE dshi:NUM-TOP-BUFFERS
           dsho:SERIALIZE-NAME
           dshu:NUM-REFERENCES.

   dshi:CLEAR().
   dsho:FILL().
   dshu:EMPTY-DATASET().

   MESSAGE DATASET dsmaster:NUM-TOP-BUFFERS
           DATASET dsrefmaster:SERIALIZE-NAME
           DATASET dsorder:NUM-REFERENCES.

   DATASET dsmaster:CLEAR().
   DATASET dsrefmaster:FILL().
   DATASET dsorder:EMPTY-DATASET().
END PROCEDURE.
```

With current revision on 3890b, the procedure will be converted in Java as:

```
public void testDsHandle(final handle _dshi,
                         final handle dsho,
                         final handle dshu,
                         final DataSetParameter dsMaster_1,
                         final DataSetParameter dsRefMaster_1,
                         final DataSetParameter dsOrder_1)
{
   handle dshi = TypeFactory.initInput(_dshi);

   internalProcedure(new Block((Init) () ->
   {
      TypeFactory.initOutput(dsho);
   }, (Body) () ->
   {
      DataSet.associate(dsMaster_1, dsMaster, true, false, false, false, false);
      DataSet.associate(dsRefMaster_1, dsRefMaster, false, true, false, false, false);
      DataSet.associate(dsOrder_1, dsOrder, true, true, false, false, false);
      message(new Object[]
      {
         dshi.unwrapDataSet().getNumTopBuffers(),
         dsho.unwrapNamedSerializable().getSerializeName(),
         dshu.unwrapDataSet().getNumReferences()
      });
      dshi.unwrapClearable().clear();
      dsho.unwrapDataSet().fill();
      dshu.unwrapDataSet().emptyDataset();
      message(new Object[]
      {
         dsMaster.getNumTopBuffers(),
         dsRefMaster.getSerializeName(),
         dsOrder.getNumReferences()
      });
      dsMaster.clear();
      dsRefMaster.fill();
      dsOrder.emptyDataset();
   }));
}
```

For the moment the DATASET-HANDLE is not recognized as special data type. I am adding now support for it. In this case, the conversion does not check the handle type, so you can pass any handle instance.

#33 - 04/08/2019 11:04 AM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> For the moment the DATASET-HANDLE is not recognized as special data type. I am adding now support for it. In this case, the conversion does not check the handle type, so you can pass any handle instance.

I need exactly this; I assume this is similar to TABLE-HANDLE. Am I correct to assume that, in 4GL, if you have a parameter like def input parameter table-handle for tt1., the caller can call proc0(h). - where h is a def var h as handle.? If this is correct, I expect dataset-handle to behave the same in 4GL.

Why I need this: is important to know these rules for method overloading in OO. I need to know if the parameter's data-type for dataset-handle and table-handle can be set as handle, as the caller can pass any handle reference here (with runtime taking care of checking if the handle argument is compatible with the parameter).

#34 - 04/08/2019 11:33 AM - Greg Shah

Please see #3751-492 for some interesting details about TABLE and TABLE-HANDLE parameters for methods. It is my understanding that methods just map to internal procedures (if there is a VOID return type) or user-defined functions (if return type is not VOID). This probably means that the testing results for these types is also meaningful for the other types of top level block. I would expect there to be some similarities for datasets too.

#35 - 04/08/2019 11:41 AM - Constantin Asofiei

Greg, yes, these makes sense, but I am trying to understand if a dataset-handle parameter can receive a generic handle as argument. Looks like 4GL allows a test like this to compile:

```
def temp-table tt1 field f1 as int.
def var h as handle.

procedure proc0.
 def input parameter table-handle htt1.
end.

run proc0(h).
```

but it will not match the parameters. I've tried h = temp-table tt1:handle or h = buffer tt1:handle, but the result is the same, code will not work at runtime (parameter mismatch). So looks like there is no runtime usage to pass a generic handle as an argument for a table-handle parameter.

Ovidiu, can you give me a similar test with a dataset-handle?

#36 - 04/08/2019 12:24 PM - Ovidiu Maxiniuc

```
PROCEDURE test-ds-handle-2:
    DEFINE INPUT PARAMETER DATASET-HANDLE dsphi.
    DEFINE OUTPUT PARAMETER DATASET-HANDLE dspho.
    DEFINE INPUT-OUTPUT PARAMETER DATASET-HANDLE dsphu.

   MESSAGE dsphi:NUM-TOP-BUFFERS
           dspho:SERIALIZE-NAME
           dsphu:NUM-REFERENCES.

   dsphi:CLEAR().
   dspho:FILL().
   dsphu:EMPTY-DATASET().
END PROCEDURE.

/*
DEF VAR h AS HANDLE.
h = DATASET dsmaster:HANDLE.
RUN test-ds-handle-2(INPUT h , OUTPUT h, INPUT-OUTPUT h).
*/

RUN test-ds-handle-2(
    INPUT DATASET dsmaster,
    OUTPUT DATASET dsmaster,
    INPUT-OUTPUT DATASET dsmaster).
```

The commented code does not work, the error 3230 is raised at runtime: Mismatched parameter types passed to procedure <procedure>. But the active RUN statement is OK.

#37 - 04/09/2019 06:42 PM - Eric Faulhaber

Ovidiu, antlr reports a lot of errors when I build 3809b/11596. Is that expected?

#38 - 04/10/2019 02:43 AM - Ovidiu Maxiniuc

Yes, there are two ambiguity issues.
I haven't added options { generateAmbigWarnings = false; } to hide them yet. I would like Greg to review the changes before doing so.

#39 - 04/11/2019 10:31 AM - Ovidiu Maxiniuc

Branch 3809b was updated. Current revision is (LE) 11605 . It contains fixes for regression introduces the last two days and increased runtime support for data-relation objects.
I also committed the fix for build.xml. It fixes the inexisting resource issue that stopped the compile task. Now DataSet Test Suite should convert and compile without errors.

#40 - 04/11/2019 11:26 AM - Greg Shah

Code Review Task Branch 3809b Revision 11605

1. annotations/datasets.rules implements linking of dataset/datasource references to their static definitions. This kind of processing is normally handled at parse time using the symbol resolver, using tempidx and leaving behind refid annotations. We should probably move to that approach to be consistent AND to enable certain forms of analysis or reporting that can be done before annotations.

2. In annotations/datasets.rules, there is code at the end that tries to determine wrapper types for unknown value parms. Isn't this normally handled by SignatureHelper?

3. In progress.g, the DATA_SOURCE_QUERY and DATA_SOURCE_BUFFER tokens should not be defined inside of the unreserved keywords section.

4. In progress.g, the for_record_spec no longer needs the 2nd parameter and the datasrc_keys_clause references can no longer be reached. Those should be removed.

5. In progress.g, I think the source_buffer_phrase should not reference record_phrase but instead it should use record directly. Correct me if I am wrong here but the docs suggest this is just a buffer reference and it has no WHERE clause, options etc... That will reduce at least some of the ambiguity.

6. In progress.g, the code in for_parent_relation_spec now drops the KW_FOR. That is OK, but now when we drop tokens we must add an action to hide them like this: f:KW_FOR! { hide(f); }. Otherwise we lose these tokens, which is a problem for usage in ProgressTransformationDriver because we then cannot reconstruct the original source code from the AST.

7. In progress.g, why did you remove the root node creation in nested_clause?

8. In variable_definitions.rules the comment identify all non-var backed widget definitions that are also frame fields is no longer valid since the dataset/datasource resources cannot be widgets.

9. Methods added to the end of TempTable.java and Temporary.java are missing javadoc.

10. method_defs.rules, method_definitions.rules, gaps/database.rules, gaps/expressions.rules, ChangeBroker.java, LogicalTerminal, LegacyResource are missing a history entry.

#41 - 04/11/2019 02:34 PM - Ovidiu Maxiniuc

Greg Shah wrote:

> Code Review Task Branch 3809b Revision 11605

Thank you for the review. It helps me fix some issues earlier, before building new code and 'burying' the issues.

> 1. annotations/datasets.rules implements linking of dataset/datasource references to their static definitions. This kind of processing is normally handled at parse time using the symbol resolver, using tempidx and leaving behind refid annotations. We should probably move to that approach to be consistent AND to enable certain forms of analysis or reporting that can be done before annotations.

Understood. I will adjust the code.

> 2. In annotations/datasets.rules, there is code at the end that tries to determine wrapper types for unknown value parms. Isn't this normally handled by SignatureHelper?

I have reviewed those tokens but the affected ones were already there. A bit of a puzzle. I added a couple of methods that were missing.

> 3. In progress.g, the DATA_SOURCE_QUERY and DATA_SOURCE_BUFFER tokens should not be defined inside of the unreserved keywords section.

These are just some markers I needed for grouping parameters from a plain tree. Is it OK to add move within RESERVED boundaries?

> 4. In progress.g, the for_record_spec no longer needs the 2nd parameter and the datasrc_keys_clause references can no longer be reached. Those should be removed.

I removed the second parameter of for_record_spec but datasrc_keys_clause cannot be dropped as it is called from source_buffer_phrase / def_datasrc_stmt.

> 5. In progress.g, I think the source_buffer_phrase should not reference record_phrase but instead it should use record directly. Correct me if I am wrong here but the docs suggest this is just a buffer reference and it has no WHERE clause, options etc... That will reduce at least some of the ambiguity.

You are correct. Some ambiguities still remain.

> 6. In progress.g, the code in for_parent_relation_spec now drops the KW_FOR. That is OK, but now when we drop tokens we must add an action to hide them like this: f:KW_FOR! { hide(f); }. Otherwise we lose these tokens, which is a problem for usage in ProgressTransformationDriver because we then cannot reconstruct the original source code from the AST.

Done.

> 7. In progress.g, why did you remove the root node creation in nested_clause?

In Java call, they are both boolean parameters at same level.

> 8. In variable_definitions.rules the comment identify all non-var backed widget definitions that are also frame fields is no longer valid since the dataset/datasource resources cannot be widgets.

I will fix this.

> 9. Methods added to the end of TempTable.java and Temporary.java are missing javadoc.

Added.

> 10. method_defs.rules, method_definitions.rules, gaps/database.rules, gaps/expressions.rules, ChangeBroker.java, LogicalTerminal, LegacyResource are missing a history entry.

Done.

Committed revision 11606.

#42 - 04/11/2019 03:00 PM - Greg Shah

> Is it OK to add move within RESERVED boundaries?

That is still for keywords. Instead, please put them inside the BEGIN_MULTIWORD/END_MULTIWORD section.

> datasrc_keys_clause cannot be dropped as it is called from source_buffer_phrase / def_datasrc_stmt.

I just want the references to datasrc_keys_clause removed from inside for_record_spec. As you note, the rule itself must survive.

#43 - 04/12/2019 12:17 PM - Ovidiu Maxiniuc

Greg Shah wrote:

> Code Review Task Branch 3809b Revision 11605

1. annotations/datasets.rules implements linking of dataset/datasource references to their static definitions. This kind of processing is normally handled at parse time using the symbol resolver, using tempidx and leaving behind refid annotations. We should probably move to that approach to be consistent AND to enable certain forms of analysis or reporting that can be done before annotations.

I tried to implement this the first time. I re-analysed the conversion and I am inclined to keep the current solution here.
When I started adding conversion support I inspected how these data types are used. It turned that they are very similar to the queries than to simple variable. So, the static dataset s and data-source s variables are handled the same way as static queries. There are not variables of this kind. Their usages are associated with the define statements exactly the same way.
To do the association at parsing time, we need to add a variable types VAR_DATASET and VAR_DATA_SOURCE. The SYMBOL node type used now needs to be replaced with these and annotated. This will complicate the detection of BDTs, as these are not.

#44 - 04/12/2019 12:31 PM - Greg Shah

I'm not suggesting to use variable types. The idea is to use the tempIdx value in SymbolResolver (see addWrappedWorker() to mark a definition of a resource (dataset, data-source or even a data relation if it makes sense) with a tempidx annotation. This would be stored in the ScopedSymbolDictionary for that resource using a simple implementation of the TokenDataWrapper. When a dataset (or other resource) is looked up in that dictionary, we would annotate the reference using the same tempidx value. Then it is easy at fixups to replace these with the real node id. We already do this work in addWrappedWorker() and Variable.annotateOptions(), but we don't need to re-use that implementation. We can do something simple here.

#45 - 04/12/2019 12:59 PM - Constantin Asofiei

Ovidiu, do your testcases cover DATASET and DATASET-HANDLE as OO method arguments?

#46 - 04/12/2019 01:03 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, do your testcases cover DATASET and DATASET-HANDLE as OO method arguments?

I do not think I remember seeing much OO code in that project. In fact beside the EventsHandler.cls I do not think there is anything related to OO.

#47 - 04/12/2019 01:15 PM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> Constantin Asofiei wrote:
>
> > Ovidiu, do your testcases cover DATASET and DATASET-HANDLE as OO method arguments?

> I do not think I remember seeing much OO code in that project. In fact beside the EventsHandler.cls I do not think there is anything related to OO.

Hmm... related to OO, we need tests at least for:

- DATASET and DATASET-HANDLE as method arguments
- DATASET as class member (instance or static), different access modes

Also, do you have tests for shared datasets? I see no changes in SharedVariableManager related to this.

#48 - 04/12/2019 01:20 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Hmm... related to OO, we need tests at least for:
>
> - DATASET and DATASET-HANDLE as method arguments
> - DATASET as class member (instance or static), different access modes

Eric, can our collaborator handle this?

> Also, do you have tests for shared datasets? I see no changes in SharedVariableManager related to this.

My intention is to call SharedVariableManager methods from DataSetManager.register(). I have not reach point where I add support code in SVM yet.

#49 - 04/12/2019 01:22 PM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> > Also, do you have tests for shared datasets? I see no changes in SharedVariableManager related to this.
>
> My intention is to call SharedVariableManager methods from DataSetManager.register(). I have not reach point where I add support code in SVM yet.

OK, looks good.

Same for functions, do you have tests for DATASET and DATASET-HANDLE as parameter?

#50 - 04/12/2019 01:25 PM - Constantin Asofiei

Ovidiu, please add conversion and stub support for BUFFER:DATA-SOURCE attribute.

#51 - 04/12/2019 01:29 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu Maxiniuc wrote:
>
> > Also, do you have tests for shared datasets? I see no changes in SharedVariableManager related to this.
>
> My intention is to call SharedVariableManager methods from DataSetManager.register(). I have not reach point where I add support code in SVM yet.
>
> OK, looks good.
>
> Same for functions, do you have tests for DATASET and DATASET-HANDLE as parameter?

There are a few procedures that use that. I had to implement them to make the code convert. I am not sure whether (in fact am strongly inclined to think that they don't) the tests covers all syntax forms and runtime behaviour.

#52 - 04/12/2019 06:07 PM - Ovidiu Maxiniuc

Greg Shah wrote:

> I'm not suggesting to use variable types. The idea is to use the tempIdx value in SymbolResolver (see addWrappedWorker() to mark a definition of a resource (dataset, data-source or even a data relation if it makes sense) with a tempidx annotation. This would be stored in the ScopedSymbolDictionary for that resource using a simple implementation of the TokenDataWrapper. When a dataset (or other resource) is looked up in that dictionary, we would annotate the reference using the same tempidx value. Then it is easy at fixups to replace these with the real node id. We already do this work in addWrappedWorker() and Variable.annotateOptions(), but we don't need to re-use that implementation. We can do something simple here.

Committed revision 11608.
However, at this moment it is highly untested and possible unstable.

#53 - 04/13/2019 06:11 PM - Eric Faulhaber

Ovidiu, now that branch 3750b has been merged to trunk 11302, I have rebased branch 3809a (currently at revision 11303).

Please update/rebase 3809b to the last revision (11588) of 3750b (I have not yet archived it), then merge your changes from branch 3809b into 3809a. We will continue the ProDataSet work in 3809a once this is done. After you have moved your changes into 3809a, please mark 3809b as dead.

#54 - 04/15/2019 03:23 AM - Constantin Asofiei

Ovidiu, once you have the dataset conversion/parsing changes in 3809a stable enough, I think we need to merge this to trunk. I've been working on top of 3809b to fix parsing/conversion issues.

Also, I have some dataset conversion changes which I will commit to 3809a, once it's ready.

#55 - 04/16/2019 05:40 AM - Ovidiu Maxiniuc

Branch 3750b was relased and merged into 3750a, then archived as 'dead'.
Branch 3750a was updated to r11304. DataSet Test Suite project is converting and compiling without errors.

#56 - 04/16/2019 07:00 AM - Greg Shah

Ovidiu: Do you think it is stable/safe enough to merge to trunk or does it need additional work?

If this has no real runtime implications, we can regression test conversion only. I do expect 3811a to merge to trunk in the next few hours, so a rebase will be needed.

#57 - 04/16/2019 07:19 AM - Ovidiu Maxiniuc

Greg Shah wrote:

> Ovidiu: Do you think it is stable/safe enough to merge to trunk or does it need additional work?
> If this has no real runtime implications, we can regression test conversion only. I do expect 3811a to merge to trunk in the next few hours, so a rebase will be needed.

I think the branch is stable. There is minimal runtime support but it is practically inaccessible by the code of current projects.
More important are some changes in interface hierarchy. I had to refactor (move/extract) some methods/attributes because they are used for different objects (ex: Buffers and DataSets) so they needed a common interface to be able to be unwrapped from handles.

To make the branch ready for merging to trunk I will check again the javadocs, I know some of them are missing. If you have the time, please do a final review, mostly because of the temp-id/refid changes. And one more thing, I was not able to eliminate the ambiguity warning from progress grammar.

#58 - 04/16/2019 10:59 AM - Greg Shah

Code Review Task Branch 3809b Revision 11327

It was easier to review the old branch than the new one since my last review revision was more obvious. Since this is the basis for the head of 3809a, this should still work.

I'm good with the changes. Here are some minor notes:

1. SymbolResolver.setDatasetOptions() needs javadoc finished.

2. annotations.xml can have its history entry removed since you reversed the change.

I will resolve the parser ambiguity warnings in 3809a.

#59 - 04/16/2019 11:40 AM - Constantin Asofiei

I have some untracked changes, some related to dataset conversion, some to other, but safe enough to commit here - please review 3809a rev 11306.

#60 - 04/16/2019 12:07 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> I have some untracked changes, some related to dataset conversion, some to other, but safe enough to commit here - please review 3809a rev 11306.

The support for BUFFER:DATA-SOURCE was already in 11304. Can I remove duplicate code?

#61 - 04/16/2019 12:09 PM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> Constantin Asofiei wrote:
>
> > I have some untracked changes, some related to dataset conversion, some to other, but safe enough to commit here - please review 3809a rev 11306.
>
> The support for BUFFER:DATA-SOURCE was already in 11304. Can I remove duplicate code?

Yes, please do; sorry about that, not sure when you added it, but I somehow missed it and add it my own.

#62 - 04/16/2019 12:12 PM - Constantin Asofiei

Constantin Asofiei wrote:

> Ovidiu Maxiniuc wrote:
>
> > Constantin Asofiei wrote:
> >
> > > I have some untracked changes, some related to dataset conversion, some to other, but safe enough to commit here - please review 3809a rev 11306.
> >
> > The support for BUFFER:DATA-SOURCE was already in 11304. Can I remove duplicate code?
>
> Yes, please do; sorry about that, not sure when you added it, but I somehow missed it and add it my own.

Something else - wasn't at some point the idea to avoid the generic get/set method name convention in Buffer interface, as at some point we will have collisions with DMO get/set methods?

#63 - 04/16/2019 12:16 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Something else - wasn't at some point the idea to avoid the generic get/set method name convention in Buffer interface, as at some point we will have collisions with DMO get/set methods?

I really do not recall the discussion, probably my memory has a black hole. Nevertheless, it clearly makes sense. I am dropping the prefixes.

#64 - 04/16/2019 12:44 PM - Constantin Asofiei

3809a 11308 contains some required legacy classes .

#65 - 04/16/2019 01:00 PM - Constantin Asofiei

Constantin Asofiei wrote:

> 3809a 11308 contains some required legacy classes .

and another legacy class in 11309 (last one).

#66 - 04/17/2019 03:04 PM - Ovidiu Maxiniuc

Fixed hierarchy and some annotation. Added runtime support. refid: #3809

Committed revision 11311.

#67 - 04/18/2019 12:21 PM - Greg Shah

> And one more thing, I was not able to eliminate the ambiguity warning from progress grammar.

Resolved in revision 11312.

#68 - 04/18/2019 01:44 PM - Ovidiu Maxiniuc

The r11310 passed the conversion test on devsrv01, with no changes in generated code compared to latest trunk. I do not expect any changes in any of the current projects. Are there any other test to run before merging this to trunk?

#69 - 04/18/2019 02:30 PM - Greg Shah

Is there enough runtime change to make runtime regression testing useful?

#70 - 04/18/2019 02:36 PM - Constantin Asofiei

Ovidiu, I need support for dataset references passed to i.e. function calls (and from this I will adapt for OO method calls). The test looks like:

```
def temp-table tt1 field f1 as int.
define dataset MtlDs serialize-name "MtlDs"  for tt1.
function func0 returns int(output dataset for MtlDs bind).
end.

func0(output dataset mtlds append by-reference).
```

this needs to emit something like func0(new DataSetParameter(...)).

Is the same issue as for TABLE parameters at function calls emitted as direct Java calls.

#71 - 04/18/2019 02:56 PM - Ovidiu Maxiniuc

Greg Shah wrote:

> Is there enough runtime change to make runtime regression testing useful?

I think not. I summary: for the moment only classes were added and some old classes gained new stubs or even implemented methods which cannot be invoked from the code in the tests. The DataSource and DataTarget were renamed to SourceData and TargetData. A few legacy attributes/methods were moved from an interface to another.

#72 - 04/18/2019 05:10 PM - Greg Shah

I'm inclined to get this merged to trunk.

Constantin: Is it OK to deal with #3809-70 in another branch? Or will that cause us other issues?

#73 - 04/18/2019 05:15 PM - Constantin Asofiei

Greg Shah wrote:

> Constantin: Is it OK to deal with #3809-70 in another branch? Or will that cause us other issues?

I think is OK. Unless next 3809 branch will be contain conversion-only changes and expected to be merged fast, I'll cherry-pick any conversion changes I need from it and place them in 3751a. I don't want to work on top of a branch which is expected to take long to merge.

#74 - 04/18/2019 05:19 PM - Greg Shah

Ovidiu: How long do you expect the parameter support changes to take? It may make sense to hold 3809a for those changes and then do one more round of conversion testing. That will make it easier to work on the conversion of the customer applications in process right now.

#75 - 04/19/2019 09:13 AM - Ovidiu Maxiniuc

A bit later than expected I have the solution for append and/or by-reference datasets.
I tried to use the same conversion path as for tables but I encountered an issue: calling a function with a table by-reference will crash the conversion. This is because of the wrapping implementation. The KW_BY_REF node is created as child of the function call (at same level as the other parameters), but the wrapping uses the index of the nodes to lookup the class. Since KW_BY_REF nodes are not real parameters they will cause the following IOOBE exception:

```
wrapper = typelist.get(copy.indexPos)
                   ^  { Index: 1, Size: 1 }
ERROR:
com.goldencode.p2j.pattern.TreeWalkException: ERROR!  Active Rule:
-----------------------
      RULE REPORT
-----------------------
Rule Type :   WALK
```

```
Source AST:  [ by-reference ] BLOCK/ASSIGNMENT/EXPRESSION/FUNC_INT/KW_BY_REF/ @74:35 {30064771656}
Copy AST  :  [ by-reference ] BLOCK/ASSIGNMENT/EXPRESSION/FUNC_INT/KW_BY_REF/ @74:35 {30064771656}
Condition :  wrapper = typelist.get(copy.indexPos)
Loop      :  false
--- END RULE REPORT ---
```

To avoid such issues, for dataset s the KW_BY_REF node is parented to KW_DATASET instead, as the KW_APPEND is for both dataset s and table s. I tried to move it at same position for table s, but I encountered TRPL rules that expects the KW_BY_REF at the current location.
Committed as r11313.

#76 - 04/19/2019 01:26 PM - Constantin Asofiei

Ovidiu, I assume the conversion rules for dataset as function call argument are still WIP?

#77 - 04/19/2019 01:29 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, I assume the conversion rules for dataset as function call argument are still WIP?

No, not in WIP, the r11313 of 3809a already supports the conversion of dataset as function call argument. You can test/use it.
It does not support TABLE as function call argument passed BY-REFERENCE.

#78 - 04/19/2019 01:32 PM - Ovidiu Maxiniuc

Sorry,
After my first reply I double checked the branch. I forgot to merge in the TRPL changes :(. Please update to r11314.

#79 - 04/19/2019 02:38 PM - Constantin Asofiei

Thanks, Ovidiu - 3809a rev 11315 fixes my case for the OO method calls.

#80 - 04/19/2019 03:30 PM - Greg Shah

Code Review Task Branch 3809a Revision 11315

Generally, I'm good with this code. Some small items are below.

1. Constantin: In regard to the Preprocessor change:

- Does this resolve #3846 (I assume so)?
- Doesn't the change also need to be added to the handle CR escapes section at the top too?

2. I think the gap marking needs to be updated with some additional items. For example, FIRST-DATASET is supported (maybe full in both?).

#81 - 04/19/2019 03:34 PM - Constantin Asofiei

Greg Shah wrote:

1. Constantin: In regard to the Preprocessor change:

   - Does this resolve #3846 (I assume so)?

Yes, that task describes the fixed problem.

   - Doesn't the change also need to be added to the handle CR escapes section at the top too?

Yes, it needs to be there, too. I'll add it.

#82 - 04/19/2019 03:38 PM - Ovidiu Maxiniuc

Greg Shah wrote:

Code Review Task Branch 3809a Revision 11315
2. I think the gap marking needs to be updated with some additional items. For example, FIRST-DATASET is supported (maybe full in both?).

At the moment I updated the gap marking, the runtime support was not yet present. It (and a few other methods and attributes) was gained when I moved to proper interfaces and inheritance of already existing abstract classes. Also, I am actively working on adding new runtime support for other methods and attributes but they are not fully tested right now. In consequence, I would like to keep them as they are for the moment and update them in batches, as the runtime support is improved.

#84 - 04/19/2019 04:44 PM - Ovidiu Maxiniuc

The revision 11316 adds some new runtime support. It passed the DataSet Test Suite conversion test.

#85 - 04/19/2019 04:55 PM - Greg Shah

Code Review Task Branch 3809a Revision 11316

I'm good with the changes. Constantin will include the Preprocessor change in a different branch.

To be safe, please put this back through conversion regression testing. If it passes for the ChUI regression application and at least one of the GUI application conversions, then you can merge to trunk.

#86 - 04/22/2019 02:46 AM - Constantin Asofiei

Ovidiu, please rebase 3809a.

#87 - 04/23/2019 02:43 AM - Constantin Asofiei

Ovidiu, there is a regression in 3809a: you've added the kw_cpy_ttbl to the top load_descriptors function, and this code on line 4573:

```
                     <rule>descriptors.containsKey(ftype)
                        <action>unwrap = false</action>
                     </rule>
```

prevents for a buffer b:copy-temp-table(...) to emit properly. I've fixed it like:

```
                     <rule>descriptors.containsKey(ftype) and ftype != prog.kw_cpy_ttbl
                        <!-- WARNING: for same cases, wrapping is still needed -->
                        <action>unwrap = false</action>
                     </rule>
```

but I'm not sure why we would avoid wrapping if the attr/meth is in load_descriptors...

#88 - 04/23/2019 09:40 AM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, there is a regression in 3809a: you've added the kw_cpy_ttbl to the top load_descriptors function, and this code on line 4573:
> [...]
> prevents for a buffer b:copy-temp-table(...) to emit properly. I've fixed it like:
> [...]
> but I'm not sure why we would avoid wrapping if the attr/meth is in load_descriptors...

I think emitting

```
TempTableBuilder.asTempTable(b).copyTempTable(new handle(TempTableBuilder.asHandle(tt1)));
```

is a bit to verbose. Why don't we emit instead:

```
b.copyTempTable(new handle(TempTableBuilder.asHandle(tt1)));
```

This will require:

- Buffer interface to extend the new TempTableDuplicator;
- implement logical copyTempTable(handle other) and overloaded in BufferImpl as:

```
   @Override
   public logical copyTempTable(handle ref)
   {
```

```
      return TempTableBuilder.asHandle((Temporary) this).unwrapTempTable().copyTempTable(ref);
   }
```

Eric, what do you say?

BTW, Constantin, where did you encounter this testcase?

#89 - 04/23/2019 09:53 AM - Eric Faulhaber

Ovidiu Maxiniuc wrote:

> Eric, what do you say?

Yes, I like this better.

#90 - 04/23/2019 11:47 AM - Ovidiu Maxiniuc

Greg Shah wrote:

> To be safe, please put this back through conversion regression testing. If it passes for the ChUI regression application and at least one of the GUI application conversions, then you can merge to trunk.

The generated code for the GUI application I tested showed the following changes:

trunk	3809a	cause
hQuery.unwrapQuery().bufferHandle(n)	hQuery.unwrapBufferCollection().bufferHandle(n)	The bufferHandle was extracted to BufferCollection interface
hQuery.unwrapQuery().numBuffers()	hQuery.unwrapBufferCollection().numBuffers()	The numBuffers was extracted to BufferCollection interface
hQuery.unwrapQuery().addBuffer(hBuffer)	hQuery.unwrapBufferCollection().addBuffer(hBuffer)	The addBuffer was extracted to BufferCollection interface
hQuery.unwrapQuery().setBuffers(...)	hQuery.unwrapBufferCollection().setBuffers(...)	The setBuffers was extracted to BufferCollection interface
hTable.unwrapTempTable().createTableLike(hBuffer)	hTable.unwrapTempTableDuplicator().createLike(hBuffer)	The createTableLike was renamed to createLike and extracted to TempTableDuplicator interface
new DataTarget(...)	new TargetData(...)	The class DataTarget was renamed to TargetData

All these changes were expected.

The ChUI test passed. There were a few of fails but they are in testcases known to generate false errors.

I am going to merge 3809a to trunk and open 3809c for following updates, including the one from note-88.

#91 - 04/23/2019 01:17 PM - Ovidiu Maxiniuc

3809a was merged to trunk as r11304. Fresh new 3809c was created for following changes on this task.

#92 - 05/02/2019 10:24 AM - Constantin Asofiei

Ovidiu, dataset-handle parameters need to convert as DatasetParameter (same as table-handle does). Currently they emit as handle, which is not correct. Please fix this.

#93 - 05/08/2019 06:13 PM - Constantin Asofiei

Constantin Asofiei wrote:

> Ovidiu, dataset-handle parameters need to convert as DatasetParameter (same as table-handle does). Currently they emit as handle, which is not correct. Please fix this.

Please port the changes related to this (and only this) from your branch to 3751a.

#94 - 05/09/2019 04:39 AM - Ovidiu Maxiniuc

Please check out the 3751a/11308.

#95 - 05/09/2019 08:10 AM - Ovidiu Maxiniuc

Constantin, I have the following definitions in a object:

```
    method public void onFill (dataset-handle dsHdl): [...]

    method public void onFill (table-handle ttHdl): [...]
```

They are converted in Java as:

```
   @LegacySignature(type = Type.METHOD, name = "onFill", parameters =
   {
      @LegacyParameter(name = "dsHdl", type = "HANDLE", mode = "INPUT")
   })
   public void onFill(final DataSetParameter _dsHdl)
   {...}

   @LegacySignature(type = Type.METHOD, name = "onFill", parameters =
   {
      @LegacyParameter(name = "ttHdl", type = "HANDLE", mode = "INPUT")
   })
   public void onFill(final TableParameter _ttHdl)
```

```
   {...}
```

I think this is not correct. The CFO will try to use HANDLE type to match the actual parameter (a DataSetParameter or TableParameter) and will fail (as they are not handle s).

#96 - 05/09/2019 08:11 AM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> Constantin, I have the following definitions in a object:
> [...]
> They are converted in Java as:
> [...]
>
> I think this is not correct. The CFO will try to use HANDLE type to match the actual parameter (a DataSetParameter or TableParameter) and will fail (as they are not handle s).

You mean the annotation is wrong ? What does the name_map.xml say for a internal procedure with the same arguments?

#97 - 05/09/2019 09:19 AM - Ovidiu Maxiniuc

Yes, I though the annotation is incorrect. The type attribute is "HANDLE", too, in name-map. I understand they have the same role.
Please, let me do a bit of debug to understand why the signature is not a match.

#98 - 05/09/2019 10:46 AM - Greg Shah

I do agree that using "HANDLE" seems incorrect for these cases. Aren't these specific types which cannot be intermixed with any regular handle?

#99 - 05/10/2019 12:01 PM - Ovidiu Maxiniuc

Related to previous discussion about BEFORE-TABLE in #3815, entries 7 to 10.
We had a problem with the attribute conversion.

```
hbuff = BUFFER tt2:HANDLE.
MESSAGE hbuff:BEFORE-TABLE
        BUFFER tt2:BEFORE-TABLE
        TEMP-TABLE tt2:BEFORE-TABLE
        TEMP-TABLE tt2:BUFFER-COMPARE(BUFFER tt1:HANDLE).
```

This was converted as

```
hbuff.assign(tt2);
message(new Object[]
{
```

```
   hbuff.unwrapTempTable().getBeforeTable(),
   tt2.getBeforeTable(),
   tt2.getBeforeTable(),
   TempTableBuilder.asTempTable(tt2).bufferCompare(new handle(tt1))
});
```

This was wrong. The BEFORE-TABLE attribute applies to a TEMP-TABLE object handle:

- the first generated, when run, will properly display "** BEFORE-TABLE is not a queryable attribute for BUFFER widget. (4052)". This seems fine;
- BUFFER tt2:BEFORE-TABLE is converted to tt2.getBeforeTable() as it was a before-table field of the tt2 table. Of course, this will display ** No tt2 record is available. (91) and quit. Normally, it should have displayed the same message as above, accessing (or better said trying to access and faling) the Buffer(Impl) interface;
- the 3rd (TEMP-TABLE tt2:BEFORE-TABLE) is the proper way to access the attribute in ABL. We were loosing the TEMP-TABLE namespace during the conversion. Evidently there is a big difference between the last two expressions, but they get to be generated the same. I think this is a problem for all TEMP-TABLE attributes accessed this way;
- the last one is even worse: it's not even compilable. ABL/OE compiler accepts it, will issue the same 4052 runtime error.

I fixed these issues in 11310/3809c. The new generated code is more explicit now:

```
hbuff.assign(tt2);
message(new Object[]
{
   hbuff.unwrapTempTable().getBeforeTable(),
   buffer(tt2).unwrapTempTable().getBeforeTable(),
   tempTable(tt2).getBeforeTable(),
   tempTable(tt2).unwrapBuffer().bufferCompare(new handle(tt1))
});
```

Maybe the general aspect of code would have looked better if the static methods buffer and tempTable were named Buffer and TempTable, but I kept the Java standard naming for methods.

#100 - 05/13/2019 11:11 AM - Constantin Asofiei

Ovidiu, I need some help; this is valid code (both runtime and compile) in OpenEdge 11.7:

```
def temp-table tt1 field f1 as int.

temp-table tt1:empty-temp-table().
```

We ened empty-temp-table method at both the BUFFER and TEMP-TABLE resources. Please fix this in 3751a.

#101 - 05/13/2019 11:13 AM - Ovidiu Maxiniuc

I will add this in a couple of hours.

#102 - 05/13/2019 12:42 PM - Ovidiu Maxiniuc

Please update 3751a to r11316.
I could not test on Dataset project because of some other issue.

#103 - 05/14/2019 01:53 PM - Constantin Asofiei

I'm posting this here, but we may need to work it in a different branch; when calling a TABLE argument with the BY-REFERENCE option, the called procedure's 'locally defined' reference has different resource IDs than the source one. See this test:

- runner program p0.p:

```
def temp-table tt1 field f1 as int.

create tt1.
tt1.f1 = 10.
def var h as handle.
run p1.p persistent set h.
run proc0 in h (output table tt1).

message temp-table tt1:handle buffer tt1:handle.
run proc0 in h (output table tt1 by-reference).
```

- second program @p1.p2:

```
def temp-table tt1 field f1 as int .
message temp-table tt1:handle buffer tt1:handle.

procedure proc0.
   def output parameter table for tt1.
   message temp-table tt1:handle buffer tt1:handle.
end.
```

This emits 4 sets of resource IDs:

- the first two sets will be the same (for the def temp-table tt1 field f1 as int resource defined at p1.p)
- the last two sets will be the same, but different, as they are for the def temp-table tt1 field f1 as int resource defined at p0.p.

In FWD, in p1.p we link the references in proc0 to the def temp-table statement, not to the proc0's parameter. I don't think this is OK. The same applies for DATASET, too.

#104 - 05/14/2019 04:02 PM - Constantin Asofiei

I need compiler constant support for the ROW-STATE possible values... planing to add them now to 3751a.

#105 - 05/14/2019 04:07 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> I need compiler constant support for the ROW-STATE possible values... planing to add them now to 3751a.

I've already added them in r11318/3809a. LE: 3809c, sorry.

#106 - 05/14/2019 04:19 PM - Ovidiu Maxiniuc

I committed the conversion support for you as r11321 in 3751a.

#107 - 05/17/2019 03:19 PM - Constantin Asofiei

There are issues with by-reference, append and bind options for caller's DATASET-HANDLE, DATASET, TABLE-HANDLE, TABLE arguments - I'm not going to touch these now. The problem is that the rules expected the i.e. KW_BY_REF to be a child of KW_TAB_HAND, but this is a sibling, not a child... also, the parameters for these calls are not parented via a PARAMETER node, so is not correct to check the parent, as everybody is a sibling - you will not be able to distinguish between two arguments, one with and the other without BY-REFERENCE.

#108 - 05/18/2019 12:49 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> There are issues with by-reference, append and bind options for caller's DATASET-HANDLE, DATASET, TABLE-HANDLE, TABLE arguments - I'm not going to touch these now. The problem is that the rules expected the i.e. KW_BY_REF to be a child of KW_TAB_HAND, but this is a sibling, not a child... also, the parameters for these calls are not parented via a PARAMETER node, so is not correct to check the parent, as everybody is a sibling - you will not be able to distinguish between two arguments, one with and the other without BY-REFERENCE.

I will investigate these on Monday.

Meanwhile I added new implementation for runtime. Latest revision of 3809c is 11320.

#109 - 05/21/2019 03:49 PM - Ovidiu Maxiniuc

The problem with parameter options is that FWD uses a mixed tree locations. Here is an example:

```
RUN test-all-flags(
   OUTPUT TABLE tt1               APPEND /*BY-VALUE BY-REFERENCE BIND*/,
   OUTPUT DATASET ds-1            APPEND /*BY-VALUE*/ BY-REFERENCE /*BIND*/,
   OUTPUT TABLE-HANDLE hbuff      APPEND /*BY-VALUE BY-REFERENCE*/ BIND,
```

```
   OUTPUT DATASET-HANDLE hDataSet APPEND BY-VALUE /*BY-REFERENCE BIND*/
).
```

There is one of each time of parameter for tables/datasets. The simplified Ast tree looks like this:

```
   statement [STATEMENT]
      RUN [KW_RUN]
         test-all-flags [FILENAME]
         "OOOO" [STRING]
         ( [LPARENS]
            parameter [PARAMETER]
               OUTPUT [KW_OUTPUT]
               TABLE [KW_TABLE]
                  tt1 [TEMP_TABLE]
                  APPEND [KW_APPEND]
            parameter [PARAMETER]
               OUTPUT [KW_OUTPUT]
               expression [EXPRESSION]
                  DATASET [KW_DATASET]
                     ds-1 [DATA_SET]
                     APPEND [KW_APPEND]
                     BY-REFERENCE [KW_BY_REF]
            parameter [PARAMETER]
               OUTPUT [KW_OUTPUT]
               expression [EXPRESSION]
                  TABLE-HANDLE [KW_TAB_HAND]
                     hbuff [VAR_HANDLE]
               APPEND [KW_APPEND]
               BIND [KW_BIND]
            parameter [PARAMETER]
               OUTPUT [KW_OUTPUT]
               expression [EXPRESSION]
                  DATASET-HANDLE [KW_DSET_HND]
                     hDataSet [VAR_HANDLE]
               APPEND [KW_APPEND]
               BY-VALUE [KW_BY_VALUE]
```

Notice the location of KW_APPEND and passing type options (KW_BY_REF, KW_BY_VALUE and KW_BIND). For the first two parameters they are parented to the passed object. In the last two cases, we find them parented to PARAMETER node. Semantically, I think the latter ones use the better location. The problem is that the nodes are constructed this way since the beginning (parsing).

#110 - 05/21/2019 03:55 PM - Constantin Asofiei

The problem is not just RUN statement: for function/method calls, there is no PARAMETER node - so every option (APPEND, BY-REF, etc) is a sibling, as a child of the func call node.

See func_call_parameters in progress.g; you can experiment with 4GL function calls to better see this.

#111 - 05/22/2019 10:22 AM - Greg Shah

> For the first two parameters they are parented to the passed object. In the last two cases, we find them parented to PARAMETER node. Semantically, I think the latter ones use the better location. The problem is that the nodes are constructed this way since the beginning (parsing).

I agree that these should be handled consistently. I'm fine with parenting these options to the PARAMETER node (for procedure parameters) for table and dataset cases.

The function cases are already structured this way so it should be consistent enough. Adding a parameter parent node may be too much change right now, though I agree it would be a better structure.

#112 - 05/24/2019 06:24 PM - Ovidiu Maxiniuc

I am struggling to make before-buffer conversion work correctly.
Apparently FWD converts correct these tables/buffers if there is at least one reference (strong/weak or even free). I added recently support for navigating from before- to after-table and back.
The problem is when the before-buffer is not explicitly used. It is not generated at all. The datasets need access to them to store data there and access is done usually dynamically.
I have listed a small code to see the before-table buffer's scope. A bit strange, they are not listed unless the buffer is explicitly used.

The before-table is duplicated in SymbolResolver and data asts (dict/schema/p2o) but not in procedure.ast. This is done very early at parsing time.
I tried to duplicate the parent table tree later, in record-scoping-prep so that the buffer to be declared. The define_table seems enough, as TRPL will assume the table is already processed. I reached the buffer-definitions where the buffer is generated but sometimes using the permanent-table buffer format. However, when a buffer reference occurs in the code, my buffer is ignored and another duplicate is created.

As an intermediary conclusion, I believe they are some kind of dynamic buffers. Not sure how to verify this piece of information.

#113 - 05/24/2019 06:27 PM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> As an intermediary conclusion, I believe they are some kind of dynamic buffers. Not sure how to verify this piece of information.

If they are dynamic, you might see them in the session:first-buffer chain. Otherwise, there is the dynamic attribute you can check at the buffer.

#114 - 05/30/2019 04:06 PM - Constantin Asofiei

Ovidiu, are you planning on implementing this method: buffer-handle:SYNCHRONIZE().? Looks like there is an implicit behaviour when the dataset is used in a BROWSE. Unless you have a reason to implement it (if you need this behavior for something else), the runtime for it is low priority.

I plan to add a stub for it in 3751b.

#115 - 05/30/2019 04:08 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, are you planning on implementing this method: buffer-handle:SYNCHRONIZE().? Looks like there is an implicit behaviour when the dataset is used in a BROWSE. Unless you have a reason to implement it (if you need this behavior for something else), the runtime for it is low priority.
>
> I plan to add a stub for it in 3751b.

No, for the moment this method is not on my work plan as I did not encounter any direct relation to datasets. Please go ahead and stub it.

#116 - 05/30/2019 04:36 PM - Constantin Asofiei

There is a case where in a define dataset there is a line like data-relation rxx for t1, t2 relation-fields (f1,f1,f2,f2), but (before the define dataset), there is a def var f1 as char. FWD gets confused and emits the var reference instead of the literal.

#117 - 05/30/2019 07:26 PM - Constantin Asofiei

Ovidiu, the testcase is like this:

```
def var f1 as char.

def temp-table stt1 field f1 as int.
def temp-table stt2 field f1 as int.

define dataset dss  for stt1, stt2
data-relation r1 for stt1, stt2 relation-fields (f1, f1) nested.
```

Please fix fix this and commit it to 3751b.

#118 - 05/31/2019 08:58 AM - Ovidiu Maxiniuc

This was caused because in Progress parser, resolveLvalueCoreType *variables by the same name as a field, will hide that field*.
Since at this moment we cannot see the location/parent of the processed token, I added a bracket that will force the fields to be prefer over variables when processing the field_mapping_clause.
If there are other places where the syntax allows only field names and not variables, the same bracket will be used.

Committed revision 11314 of 3751b.

#119 - 06/04/2019 04:07 AM - Constantin Asofiei

WRITE-JSON(...,JsonObject|JsonArray) is needed at least for buffer.

#120 - 06/04/2019 07:51 AM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> WRITE-JSON(...,JsonObject|JsonArray) is needed at least for buffer.

Is this urgent?
If so, add the new method to XmlData in your branch. If it can wait for 3809c to be merged into trunk, I will add it.

#121 - 06/04/2019 07:59 AM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> Constantin Asofiei wrote:
>
> > WRITE-JSON(...,JsonObject|JsonArray) is needed at least for buffer.
>
> Is this urgent?
> If so, add the new method to XmlData in your branch. If it can wait for 3809c to be merged into trunk, I will add it.

I added the c'tor to TargetData, so I can compile the code - see 3751b rev 11316. For the runtime, just put it on your list.

#122 - 06/12/2019 08:38 AM - Ovidiu Maxiniuc

I have the following issue.
In a previous commit to 3809c I enriched the DMO with two new fields: _rowState and _peerRowid. They hold the values for ROW-STATE and exclusively, one of BEFORE-ROWID or AFTER-ROWID, respectively. They are generated only for temp-table DMOs. They are some kind of surrogate fields, just like the id, but they are writeable. The ABL programmer cannot directly access them (so not annotated with @LegacyField), only by specialized methods.
Their declaration looks like this:

```
   private Integer _rowState = null;
   private Long _peerRowid = null;
   public Integer _rowState() { return _rowState; }
   public void _rowState(Integer _rowState) { this._rowState = _rowState; }
   public Long _peerRowid() { return _peerRowid; }
   public void _peerRowid(Long _peerRowid) { this._peerRowid = _peerRowid;}
```

and are declared in .hbm.xml as:

```
    <!-- ROW-STATE, BEFORE-ROWID, and AFTER-ROWID support properties -->
    <property access="field" column="_peerRowid" name="_peerRowid" type="long"/>
    <property access="field" column="_rowState" name="_rowState" type="integer"/>
```

Everything works fine, except one thing: in rare cases (only on DELETE operations, see TemporaryBuffer.delete(), line ~4372) when these fields are changes by FWD the records are not updated in database. It seems to me that the transient records are saved correctly although, in some cases I had the impression that they remain null (TemporaryBuffer.copyChanges(), line ~2721).

Do you have any idea?

#123 - 06/12/2019 09:48 AM - Eric Faulhaber

Ovidiu Maxiniuc wrote:

> Everything works fine, except one thing: in rare cases (only on DELETE operations, see TemporaryBuffer.delete(), line ~4372) when these fields are changes by FWD the records are not updated in database.

What do you mean here by "not updated in database"? The records still exist after we attempt to delete them? Or something else?

#124 - 06/12/2019 10:00 AM - Ovidiu Maxiniuc

I've leant something about the TEMP-TABLE that is not documented. At least for GET-CHANGES, the source and destination tables must "match". This is because the data is copied from the source to target. In the description they say that, I quote: *the number of columns, data types, and so on*. What does "and so on" mean? The name maybe? Well, not.

After some tests I discovered that the name of the field is irrelevant. The only thing that matters is the **order** and the **type** of the fields. You can name the field anyway you want and the compatibility is assured as long the types are matching. This pose a bit of a problem with the current implementation where FWD iterates each field and forcing the copy using the getter and the setter for a field named the same. If the name of some fields are swapped, the data mismatch error will probably occur, except when the type actually matches, in which case the method succeeds but the content is incorrect.

What I think P4GL does is either:

- working with low-level memory areas: the records use a contiguous memory and are accessed using a C/C++ pointer and record-copy process is actually a blind memcpy()-ing memory bytes;
- using serialization: the record is serialized with field type prefix token (probably as noted in #3810) but no field-name, then read back to destination.

#125 - 06/12/2019 10:11 AM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> What do you mean here by "not updated in database"? The records still exist after we attempt to delete them? Or something else?

Sorry. I was not too explicit here: there are two records involved. Each time a record is worked on on a tracked temp-table, a record is created or updated in the linked before-table. When the main record is first modified, a before-record is created with original data, in M state and holding a pointer (rowid) to main table. If the same record is then deleted, the before-record is kept, but only row-state is changed to D and the pointer to main record is deleted (as the original record does not exist anymore). The actual user-data remains unchanged in the before-record. I even tried to expressly save() the before-record, but it is not persisted.

#126 - 06/12/2019 10:49 AM - Greg Shah

> I quote: the number of columns, data types, and so on. What does "and so on" mean? The name maybe? Well, not.

Please see my findings from #3751-492. Look at the Exact Matches section, for TABLE parameters. The most important part is this:

- This allows passing a direct reference to a table.

- It is NOT hard coded to the specific table being referenced, instead it can be passed to or recieved by any table that is sufficiently compatible in structure.
- The structural comparison rules check the number of fields, type of each field, extent of each field and the order these fields appear. There are the only cases are detected as differences in structure.
- The following things are are ignored: table names, table options, field names, field options (other than EXTENT), anything to do with indexes.
- These comparison rules are also used to detect when two overloaded methods have equivalent signatures such that such a definition will generate a compile error.

Although this is for parameter passing in methods, I suspect it also is the same behavior that would be found in parameter passing for procedures/functions, for BUFFER COPY/COMPARE and for your ProDataSet case.

#127 - 06/12/2019 11:05 AM - Eric Faulhaber

Ovidiu Maxiniuc wrote:

> Eric Faulhaber wrote:
>
> > What do you mean here by "not updated in database"? The records still exist after we attempt to delete them? Or something else?
>
> Sorry. I was not too explicit here: there are two records involved. Each time a record is worked on on a tracked temp-table, a record is created or updated in the linked before-table. When the main record is first modified, a before-record is created with original data, in M state and holding a pointer (rowid) to main table. If the same record is then deleted, the before-record is kept, but only row-state is changed to D and the pointer to main record is deleted (as the original record does not exist anymore). The actual user-data remains unchanged in the before-record. I even tried to expressly save() the before-record, but it is not persisted.

I wonder if our interceptor within ChangeBroker is preventing Hibernate from realizing that the data has changed. We track changes to legacy data and we get callbacks from Hibernate (see ChangeBroker$SessionInterceptor.findDirty) to "short-circuit" the expensive dirty-checking Hibernate would otherwise have to do. However, since you've added special-purpose, non-legacy data to the DMOs, these will not be tracked. If only that special-purpose data (and no legacy data) has changed, findDirty probably is telling Hibernate that no data is dirty, such that Hibernate never issues SQL update statements for those changes.

#128 - 06/12/2019 03:42 PM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> I wonder if our interceptor within ChangeBroker is preventing Hibernate from realizing that the data has changed. We track changes to legacy data and we get callbacks from Hibernate (see ChangeBroker$SessionInterceptor.findDirty) to "short-circuit" the expensive dirty-checking

> Hibernate would otherwise have to do. However, since you've added special-purpose, non-legacy data to the DMOs, these will not be tracked. If only that special-purpose data (and no legacy data) has changed, findDirty probably is telling Hibernate that no data is dirty, such that Hibernate never issues SQL update statements for those changes.

You was close. Starting from this point I traced back to the point where the changes were not correctly reported to the ChangeBroker. After calling changeBroker.stateChanged(RecordChangeEvent.Type.UPDATE...) with right parameters, the modified records are successfully flushed to database.

#129 - 06/12/2019 04:03 PM - Ovidiu Maxiniuc

Greg Shah wrote:

> [...]
> Although this is for parameter passing in methods, I suspect it also is the same behavior that would be found in parameter passing for procedures/functions, for BUFFER COPY/COMPARE and for your ProDataSet case.

I changed the old DMO-copy method which was based on property names for locating pair of fields to a new solution that takes the fields in their declaration order and uses reflection to retrieve and set the value to destination DMO. It can be found in r11340 of 3809c.

BTW, the devsrv01 conversion test was successful, but the overnight runtime test crashed. In fact I found it hanging at CTRL+C phase. Restarted.

#130 - 06/20/2019 09:43 AM - Ovidiu Maxiniuc

I have the following issue while working on save/merging changes back to DATA-SOURCE: how to identify the records/rows where the changes should be merged.

When the Buffer s /DataSet s are FILL ed, the data is copied from the DataSource attached to each Buffer into the respective buffer. There is no required UNIQUE index to make the rows uniquely identifiable, more than that, if the FILL-MODE is set to APPEND, the same row can be store in the dataset buffer. After the data is processed, the changes can be eventually extracted in a parallel DataSet / Buffer using the GET-CHANGES method.

At the end of the DATA-SET typical processing, the changed data is usually MERGE/SAVE-d back to the DATA-SOURCE. I haven't find yet the proper way to identify correct records to be updated. An initial thought was to keep a hidden reverse-link (ROWID) to original row in the BEFORE/AFTER-IMAGES. This means adding a new ROWID field in the TempTableRecord like I've done with ROW-STATE and BEFORE/AFTER-ROWID. This would clearly identify the target source/destination record. However, from my tests, this is NOT how P4GL does it. Also, if a DataSet record comes from a complex DataSet (like a multi-table QUERY), each processed field should have a separate ROWID, for the table where it comes from.

ABL offers a DATA-SOURCE-ROWID attribute that *locates the data-source row corresponding to either an after-table buffer or a before-table buffer*. The ABL reference manual says that this attribute shares some piece of code with SAVE-ROW-CHANGES method. In fact this is my issue. The attribute is indexed. It allows to optionally specify either the join-level or the buffer-name for the complex DataSource cases.

#131 - 06/24/2019 12:46 AM - Eric Faulhaber

Ovidiu, is 3809c in a state that is read for code review?

I'm sorry I do not have an idea for your previous post ATM.

#132 - 06/24/2019 05:56 AM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> Ovidiu, is 3809c in a state that is read for code review?

I added some unfinished code at the end of last week. I will finalize the methods and add necessary javadoc, comments and H-entries to Java/TRPL sources today. If you have time, please do a review for future 11352+.

> I'm sorry I do not have an idea for your previous post ATM.

I am a bit blocked on this, but hope to figure it out soon, maybe I get some insights while implementing the other missing methods. Otherwise I will try the googling the OE's KB.

#133 - 06/27/2019 09:24 AM - Ovidiu Maxiniuc

I have dumped the schema for some TEMP-TABLE and the list of fields looks like this:

```
   <xsd:element name="f11" type="xsd:string" nillable="true"/>
   <xsd:element name="f12" type="xsd:int" nillable="true"/>
   <xsd:element name="f13" type="xsd:date" nillable="true"/>
   <xsd:element name="f14" type="xsd:decimal" nillable="true"/>
   <xsd:element name="__error-flag__" type="xsd:int" nillable="true" prodata:initial="prodata:unknown"/>
   <xsd:element name="__origin-rowid__" type="xsd:base64Binary" nillable="true" prodata:dataType="
prodata:rowid"/>
   <xsd:element name="__error-string__" type="xsd:string" nillable="true" prodata:initial="prodata:unknown"/>
   <xsd:element name="__after-rowid__" type="xsd:base64Binary" nillable="true" prodata:dataType="prodata:rowid
"/>
   <xsd:element name="__row-state__" type="xsd:int" nillable="true" prodata:initial="prodata:unknown"/>
```

This gives some insight of the (datatset) temp-tables. Some of them I was able to observe/guess while working with DATASETs, but some of them are new. You can see that, beside the f1, f2, f3 and f4 fields that were declared in the DEFINE TABLE statement, there are other 5 "hidden" fields. Of those:

- I have already added conversion and runtime support for __after-rowid__ and __row-state__ in current branch;
- __origin-rowid__ seems to be exactly what I meant in note 130, for locating the original record in the source table;
- the __error-flag__ and __error-string__ are (relatively) new for me, I was not aware of such fine error level in ABL.

For the moment I will not change the conversion to support these.m

#134 - 06/27/2019 06:19 PM - Ovidiu Maxiniuc

The branch 3809c was updated to r11355 (XML serialization for DATASETs). Note that it is not yet rebased to latest trunk (I am going to do this tomorrow).

#135 - 06/28/2019 07:12 PM - Ovidiu Maxiniuc

The branch 3809c was rebased to latest trunk and updated to r11358 (includes XMLSCHEMA serialization for DATASETs).

#136 - 06/30/2019 05:07 PM - Constantin Asofiei

For JSON serialization, I need the legacy table and field names - but these are lowercased in FWD, I can't find a way to get the original one. Ovidiu, do you recall discussing this before? I have a vague memory that I inquired before about this.

#137 - 06/30/2019 05:09 PM - Hynek Cihlar

3809c build fails with:

```
[ant:javac] /home/hc/gcd/p2j_repo/3809c/src/com/goldencode/p2j/persist/RecordBuffer.java:3658: error: cannot f
ind symbol
[ant:javac]       List<TableMapper.LegacyFieldInfo> srcFlds = TableMapper.getLegacyPositionalList(srcBuffer);
[ant:javac]                                                                 ^
[ant:javac]   symbol:   method getLegacyPositionalList(BufferImpl)
[ant:javac]   location: class TableMapper
```

#138 - 07/01/2019 06:27 AM - Ovidiu Maxiniuc

Hynek Cihlar wrote:

> 3809c build fails with:
>
> [...]

You are correct. On Friday I commit my changes then I rebased the branch. On the other side, in the target trunk revision the getLegacyPositionalList method was renamed to getLegacyOrderedList. My bad, I did not build FWD before pushing the rebased revision. I did it now and made sure the Dataset Test Suite converts and compiles successfully. You can access it as r11360.

#139 - 07/01/2019 06:34 AM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> For JSON serialization, I need the legacy table and field names - but these are lowercased in FWD, I can't find a way to get the original one. Ovidiu, do you recall discussing this before? I have a vague memory that I inquired before about this.

I am not sure what you mean. The JSON serialization for DATASETs is already in 3809c. The TEMP-TABLEs is already in trunk. The DATASETs

serialize the name case-sensitive. The TEMP-TABLEs use TableMapper.getLegacyName(buf). I believe this is also case-sensitive. Please add mode details of what part of JSON serialization you mean.

#140 - 07/01/2019 06:43 AM - Ovidiu Maxiniuc

Apparently the revision 11359 added some changes, but the JsonBackend file is missing. Hynek, please add it.
LE: The ValueConverter class is also missing.

#141 - 07/01/2019 10:42 AM - Ovidiu Maxiniuc

Ovidiu Maxiniuc wrote:

> Apparently the revision 11359 added some changes, but the JsonBackend file is missing. Hynek, please add it.
> LE: The ValueConverter class is also missing.

Please disregard my previous message. I found the files.

#142 - 07/04/2019 12:29 PM - Hynek Cihlar

Ovidiu please check in the class ThrowingConsumer. Do you have other upcoming changes for JsonExport?

#143 - 07/04/2019 01:27 PM - Ovidiu Maxiniuc

Hynek Cihlar wrote:

> Ovidiu please check in the class ThrowingConsumer. Do you have other upcoming changes for JsonExport?

That is not a public class. It is a private @FunctionalInterface you can find at the end of JsonExport.java. You need to refresh the source cache of your IDE, probably.
I have a couple of changes, but it's very small/localized.

#144 - 07/04/2019 01:33 PM - Ovidiu Maxiniuc

Btw, I would like to merge JSON & XML export classes to use a single output code because both these classes write the same information, ultimately. The difference is in presentation/language, but the actual data is the same. However, even this would increase code-maintenance, I do not have time right for the moment for this kind of changes.

#145 - 07/05/2019 05:04 AM - Hynek Cihlar

Ovidiu, I have some code changes to JsonExport I'd like to merge with your changes and deliver to the branch. If you have nothing else in your pipeline for this file, I will do it by the EOD.

#146 - 07/05/2019 05:05 AM - Hynek Cihlar

Ovidiu Maxiniuc wrote:

> Hynek Cihlar wrote:
>
> > Ovidiu please check in the class ThrowingConsumer. Do you have other upcoming changes for JsonExport?
>
> That is not a public class. It is a private @FunctionalInterface you can find at the end of JsonExport.java. You need to refresh the source cache of your IDE, probably.

Thanks, I found it. The problem was that the class was not parsed due to conflicts in the sources.

#147 - 07/05/2019 09:08 AM - Ovidiu Maxiniuc

Hynek Cihlar wrote:

> Ovidiu, I have some code changes to JsonExport I'd like to merge with your changes and deliver to the branch. If you have nothing else in your pipeline for this file, I will do it by the EOD.

I was thinking of rebasing to latest trunk, but I will do this after your commit. Just let me know when it's done.

#148 - 07/05/2019 05:37 PM - Ovidiu Maxiniuc

I updated the branch to r11365. Note that the temporary DMO will change a bit. (JSON not affected)

#149 - 07/07/2019 06:13 PM - Hynek Cihlar

Ovidiu Maxiniuc wrote:

> Hynek Cihlar wrote:
>
> > Ovidiu, I have some code changes to JsonExport I'd like to merge with your changes and deliver to the branch. If you have nothing else in your pipeline for this file, I will do it by the EOD.
>
> I was thinking of rebasing to latest trunk, but I will do this after your commit. Just let me know when it's done.

Checked in.

#150 - 07/08/2019 06:13 AM - Ovidiu Maxiniuc

The trunk revision number has increased by 2. I am starting the rebase op now.

#151 - 07/08/2019 07:47 AM - Constantin Asofiei

Ovidiu, can 3809c be tested and merged to trunk in the near future? I need both Hynek's JSON changes and dataset runtime support.

#152 - 07/08/2019 10:45 AM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, can 3809c be tested and merged to trunk in the near future? I need both Hynek's JSON changes and dataset runtime support.

I would like to do this by Thursday, even if there are some unfinished functional issues, if I have the branch tested and reviewed. In fact 3809c passed devsrv01 test a couple of weeks ago, but I want to test at least one more test suite.

The update is quite big, and started to be difficult to keep it updated; it took me a lot of time to rebase it to r11323 of trunk. Most likely there will still be things to fix after the merge, in next branch iteration, but I am positive they will not have the same impact on code aspect as this branch.

BTW: Hynek, please review your changes in r11370. I encountered several conflicts with JsonConstruct, JsonObject and JsonArray code while rebasing and I am not sure if the process was finalized with success.

#153 - 07/08/2019 04:32 PM - Ovidiu Maxiniuc

Task branch 3809c was updated with ORIGIN-ROWID & DATA-SOURCE-ROWID implementations. Latest revision is: 11371.

#154 - 07/09/2019 04:23 AM - Hynek Cihlar

Ovidiu Maxiniuc wrote:

> BTW: Hynek, please review your changes in r11370. I encountered several conflicts with JsonConstruct, JsonObject and JsonArray code while rebasing and I am not sure if the process was finalized with success.

I checked in a few fixes after rebase in revision 11372.

#155 - 07/09/2019 04:25 AM - Ovidiu Maxiniuc

I've started the rebase process to latest trunk. Thanks for notifying me. I was about to overwrite your revision.

#156 - 07/09/2019 07:22 AM - Constantin Asofiei

Ovidiu, please work with high priority on DATASET support for appserver - follow how TableResultSet and TableWrapper are used and add something similar.

#157 - 07/09/2019 07:23 AM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, please work with high priority on DATASET support for appserver - follow how TableResultSet and TableWrapper are used and add something similar.

OK

#158 - 07/10/2019 12:20 AM - Eric Faulhaber

Code review task branch 3809c, rev 11326-11375 (part 1):

Ovidiu, this is quite an impressive update! I am still reviewing the runtime changes, but here are a few questions/comments on the conversion changes so far...

input_output.rules: There are a number of copy.setHidden(true) calls commented out, starting at line 169. What is the status of these?

Please update the comment at line 189 to reflect that the nodes are actually removed, not hidden.

record_scoping_post.rules: Please update the year in the copyright notice and correct the year in the header entry.

database_method.rules needs a header entry.

literals.rules: Is the intention to leave the commented, dead code starting at line 478 there as historical documentation, or will this be removed?

brew_dmo_asm.xml: Could you please show an example of the Java source equivalent associated with the change to the rule at line 301? I want to be sure I understand what you are mapping to ASM, and I can't simply convert to see it, since the source code is not emitted. I know it is related to new surrogate fields of before/after tables, but I'd like to see an example.

hibernate.xml: There is the comment, "NOTE: the ERROR-FLAG and ERROR-STRING are not added at this time". What is the plan for implementing these? The comment also notes BEFORE and AFTER ROWID, but there are no properties grafted for those. Is _peerRowid an implementation-specific property?

java_templates.tpl: Just curious: why the change to instance_var_init to specify final is false? Were we emitting something like, final Type var = null;?

Note: while I scanned over the classes in com.goldencode.p2j.oo.* in this branch, I don't really know what I'm looking at. It would be best if someone involved in the OO implementation reviewed these.

#159 - 07/10/2019 07:42 AM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> Code review task branch 3809c, rev 11326-11375 (part 1):
>
> Ovidiu, this is quite an impressive update! I am still reviewing the runtime changes, but here are a few questions/comments on the conversion changes so far...

This branch accumulated my work from +50 commits in about two months.

> input_output.rules: There are a number of copy.setHidden(true) calls commented out, starting at line 169. What is the status of these?

They were not effective. The copy.remove() does the trick. The setHidden() lines will be removed.

> Please update the comment at line 189 to reflect that the nodes are actually removed, not hidden.

OK.

> record_scoping_post.rules: Please update the year in the copyright notice and correct the year in the header entry.

OK.

> database_method.rules needs a header entry.

Done.

> literals.rules: Is the intention to leave the commented, dead code starting at line 478 there as historical documentation, or will this be removed?

I would like to drop them but I wonder whether the relative paths belong to next rule. They are about triggers. Perhaps Greg or Constantin can help decide their future?

> brew_dmo_asm.xml: Could you please show an example of the Java source equivalent associated with the change to the rule at line 301? I want to be sure I understand what you are mapping to ASM, and I can't simply convert to see it, since the source code is not emitted. I know it is related to new surrogate fields of before/after tables, but I'd like to see an example.

The JAST code looks like this:

```
<ast col="0" id="395136991259" line="0" text="" type="CS_INSTANCE_VARS">
  <ast col="0" id="395136991272" line="0" text="=" type="ASSIGN">
    <ast col="0" id="395136991273" line="0" text="java.lang.Integer" type="REFERENCE_DEF">
      <annotation datatype="java.lang.Boolean" key="final" value="false"/>
      <annotation datatype="java.lang.String" key="name" value="_rowState"/>
      <annotation datatype="java.lang.String" key="access" value="private"/>
    </ast>
    <ast col="0" id="395136991274" line="0" text="" type="NULL_LITERAL"/>
  </ast>
```

and will be converted to following field definition in DMO class:

```
   private java.lang.Integer _rowState = null;
```

> hibernate.xml: There is the comment, "NOTE: the ERROR-FLAG and ERROR-STRING are not added at this time". What is the plan for implementing these? The comment also notes BEFORE and AFTER ROWID, but there are no properties grafted for those. Is _peerRowid an implementation-specific property?

At this time I have no testcase that generates non-null values for ERROR-FLAG and ERROR-STRING. In fact, I noticed these fields only recently, when I serialized the XML schema of the temp-tables. They are listed as __error-flag__ and __error-string__.

OTOH, BEFORE and AFTER ROWID are exclusively using the _peerRowid field. I added it because of the necessities. I did not find any documentation to mention these fields so I added the surrogate. Last week when I implemented WRITE-XMLSCHEMA methods and inspected the output from 4GL I noticed a __after-rowid__ field declared. I think this is the equivalent of my _peerRowid. There is no such ting as __before-rowid__ and I believe they are using it the same way (exclusive to link the before and after images). However, this naming may be confusing so I would like to keep the 'peer' name. Also, in reference serialized files, the __after-rowid__ is not used, at least with this naming, not even for a before/after pair. Instead the link between the two records is done using the prods:id attribute which is a concatenation the name of the after-temp-table and the decimal representation of the long value of the rowid of the after-image.

> java_templates.tpl: Just curious: why the change to instance_var_init to specify final is false? Were we emitting something like, final Type var = null;?

Yes we are now, see above the declaration for _rowState. All these 'hidden' fields default to null / nill / unknown ?.

> Note: while I scanned over the classes in com.goldencode.p2j.oo.* in this branch, I don't really know what I'm looking at. It would be best if someone involved in the OO implementation reviewed these.

Constantin, could you review Hynek's changes?

#160 - 07/10/2019 09:00 AM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> Constantin, could you review Hynek's changes?

I'm OK with the OO changes.

Hynek, btw, I don't think the BlockManager.setFuncReturn is enough, the object was still being deleted - I've changed ObjectOps.scopeFinished to check if the pendingAssigned is part of the current function's return, and if so, it postpones the delete.

#161 - 07/10/2019 10:26 AM - Hynek Cihlar

Constantin Asofiei wrote:

> Ovidiu Maxiniuc wrote:
>
> > Constantin, could you review Hynek's changes?
>
> I'm OK with the OO changes.
>
> Hynek, btw, I don't think the BlockManager.setFuncReturn is enough, the object was still being deleted - I've changed ObjectOps.scopeFinished to check if the pendingAssigned is part of the current function's return, and if so, it postpones the delete.

Yes, I saw this, too, for some cases, but didn't have time to look at this closer.

#162 - 07/10/2019 02:09 PM - Eric Faulhaber

Code review task branch 3809c, rev 11326-11375 (part 2 - I am not done with the runtime review yet, but I wanted to get you some feedback while I continue to go through the update...):

Note: I will address performance concerns separately, as the current focus is correctness and completeness, and I don't want to distract from this. We can address performance issues in a separate pass.

AbstractTempTable: Why do we override/implement the readXml methods in this abstract class only to throw UnimplementedException? Is this just to satisfy the compiler for subclasses which do not implement them differently? Is there any use case which causes this version of the readXml methods to be invoked? In other words, are we likely at some point to be raising this exception unexpectedly in production?

Buffer: Should FILL-related constants be implemented as enum?

BufferImpl:

- java.lang.* import unnecessary.
- Is it necessary for buffer() to be public? I went through great pains to keep it protected until now, so as not to expose the backing RecordBuffer to outside code. If yes, then it's ok, but if there is a simple way to keep it protected, I'd prefer this.
- Many ErrorManager calls just use the Java name of the buffer (or nothing at all) as the error message, though the error message is often mentioned in comments. Are the actual messages going to be used? I think I've seen this idiom in other classes as well.
- What is the plan for writeXmlSchema (currently uses UnimplementedFeature)?
- We changed all the writeXml variants to pass null to the "worker" method to represent default values (was passing BDTs before). This is fine, but I noticed writeJson variants have been added to pass the default BDTs instead of null. Plus, they don't call the next more specific variant of the method, so the default values are repeated in every variant instead of each being encoded in only one place. More efficient, but less maintainable. Anyway, these shouldn't change over time, so the choice of efficiency is probably best. Nevertheless, the inconsistent approach between writeXml and writeJson is slightly confusing. Different authors at different times, I guess?
- Can we use an import statement rather than the fully qualified com.fasterxml.jackson.core.JsonEncoding?
- All variants of saveRowChanges are unimplemented. What is the plan for these?
- Methods which return handle for various resources which may be null (e.g., afterBuffer, dataSource, etc.) will return new handle(null). Is this correct, or should it be unknown value in that case? I don't know the right answer, it just struck me as a possible problem.
- Several variants of the dataSourceRowid setter use UnimplementedFeature at the end, but there is a partial implementation. What is the idea here? Please document in the code. The remaining variant has a TODO at the end. What is the plan for finishing these implementations?
- Same questions for mergeRowChanges as previous bullet. Looks like the setup work is done, but not the actual merge.
- setDataset uses System.err.println for an error. Is it recoverable? What does 4GL do in this case?
- Please log a warning or error at line 8135, so we know if we ever got into this situation, which seems from the comments to be rare or unexpected.
- mergeChangesImpl seems to look only for rowid conflicts. Is this the only type of conflict resolved in a merge operation? Nothing for unique indexes?

#163 - 07/10/2019 02:37 PM - Hynek Cihlar

Eric Faulhaber wrote:

> - Is it necessary for buffer() to be public? I went through great pains to keep it protected until now, so as not to expose the backing RecordBuffer to outside code. If yes, then it's ok, but if there is a simple way to keep it protected, I'd prefer this.

FYI, I just used buffer() to get to the RecordBuffer instance from a buffer handle.

#164 - 07/10/2019 03:28 PM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> AbstractTempTable: Why do we override/implement the readXml methods in this abstract class only to throw UnimplementedException? Is this just to satisfy the compiler for subclasses which do not implement them differently? Is there any use case which causes this version of the readXml methods to be invoked? In other words, are we likely at some point to be raising this exception unexpectedly in production?

The readXml methods are declared as unimplemented because they are not yet. I planned to do this this week (after in previous I've done the write), but at the moment I am busy with appserver support. They will be implemented afterwards.

> Buffer: Should FILL-related constants be implemented as enum?

These constants are used to compare the values passed in by the ABL programmer. I extracted them from initial code. I suppose they can be gathered in a String enum.

> BufferImpl:
>
> - java.lang.* import unnecessary.

Removed.
LE: this is funny. I am positive I did not add the line so the IDE must have done it in order to automatically fix something. Most likely, related to the InstantiationException from the next line.

> - Is it necessary for buffer() to be public? I went through great pains to keep it protected until now, so as not to expose the backing RecordBuffer to outside code. If yes, then it's ok, but if there is a simple way to keep it protected, I'd prefer this.

The method is accessed multiple times from com.goldencode.p2j.persist.serial package (XmlExport & JsonExport). I will think of a solution to keep it protected.

> - Many ErrorManager calls just use the Java name of the buffer (or nothing at all) as the error message, though the error message is often mentioned in comments. Are the actual messages going to be used? I think I've seen this idiom in other classes as well.

I added a new API for ErrorManager. I believe this reflect more closely the way P4GL handles errors. See ErrorManager.recordOrShowError(int id, String... params) and ErrorManager.recordOrThrowError(int id, String... params). They both call ErrorManager.replaceTokens(). The comment is just to let the reader what the user will see on screen.

> - What is the plan for writeXmlSchema (currently uses UnimplementedFeature)?

I believe you mean readXmlSchema. It will be implemented together with readXml.

> - We changed all the writeXml variants to pass null to the "worker" method to represent default values (was passing BDTs before). This is fine, but I noticed writeJson variants have been added to pass the default BDTs instead of null. Plus, they don't call the next more specific variant of the method, so the default values are repeated in every variant instead of each being encoded in only one place. More efficient, but less maintainable. Anyway, these shouldn't change over time, so the choice of efficiency is probably best. Nevertheless, the inconsistent approach between writeXml and writeJson is slightly confusing. Different authors at different times, I guess?

See note #3809-144. I would like to unify them but time is short and I wanted to keep things stable while we both (I and Hynek) were updating the branch. This is a next-iteration issue.

> - Can we use an import statement rather than the fully qualified com.fasterxml.jackson.core.JsonEncoding?

I also not agree with the code. I would like to move the detection of the encoding to serial package. Can we ignore this for the moment and fix it with previous issue?

- All variants of saveRowChanges are unimplemented. What is the plan for these?

Same like read methods. They were planned for this week, but delayed because of appserver support.

- Methods which return handle for various resources which may be null (e.g., afterBuffer, dataSource, etc.) will return new handle(null). Is this correct, or should it be unknown value in that case? I don't know the right answer, it just struck me as a possible problem.

At first I used to return new handle() for these cases. However, in some cases I encountered issues because new handle() and new handle(null) are not semantically equal. The latter suit the local needs.

- Several variants of the dataSourceRowid setter use UnimplementedFeature at the end, but there is a partial implementation. What is the idea here? Please document in the code. The remaining variant has a TODO at the end. What is the plan for finishing these implementations?

The problem is explained in note #3809-130, above. I do not have an answer yet.

- Same questions for mergeRowChanges as previous bullet. Looks like the setup work is done, but not the actual merge.

Yes, same reason: the incomplete origin-rowid attribute. It was added recently. For cases where it is known, the implementation is straightforward. When not I do not have the answer ...yet.

- setDataset uses System.err.println for an error. Is it recoverable? What does 4GL do in this case?

As noted in javadoc, this should already be checked the other way around. I do not know if a 4GL code can be written to make a similar error occur in Progress code. Probably the best thing to do here is to log it, instead. At any rate, if such a case will occur, the new DataSet reference will overwrite the possible remnant.

- Please log a warning or error at line 8135, so we know if we ever got into this situation, which seems from the comments to be rare or unexpected.

OK. It is very likely I did not write the right code to trigger that condition. Hoping to occur in next-phase.

- mergeChangesImpl seems to look only for rowid conflicts. Is this the only type of conflict resolved in a merge operation? Nothing for unique indexes?

Yes, in allCopyMode. This case needs to be added. In this mode, records with unmatching rowids from the changes are documented to be forcefully added in the target (origin) dataset/table. I have not created a testcase for this yet.

#165 - 07/11/2019 12:07 AM - Eric Faulhaber

Code review task branch 3809c, rev 11326-11375 (part 3):

CallbackData: Should geTarget be getTarget? It is called from Dataset and BufferImpl.

DataRelation:

- dataSet and strPairs parameters in c'tor need javadoc.
- getCurrentQuery method variants use UnimplementedFeature. What is the plan to implement these?
- createQuery has the comment, "this is just heuristic: must be investigated". To what does this refer?

DataSet:

- associate w/ mode parameter is unimplemented. There is a note that suggests it may be implemented in DataSetManager. I guess the other variant would move, too, if you decide to move it?
- createDynamicDataSet to be implemented in next round of work?
- createLike(character) looks problematic if dsName is unknown value. Similar issues in other variants of this method that take BDT parameters (including the prefix parameter).

DataSetManager: scopeDeleted has a TODO to be implemented. What needs to be happen when this is called?

DataSource:

- How much effort to implement attributes?
  - NEXT-ROWID
  - RESTART-ROWID
- FieldReference.toString needs javadoc.

Can you please (briefly) explain the concept of the substitution buffers which have been added to the dynamic queries and elsewhere? I read the javadoc, but maybe a 4GL query expression example would make it clearer to me what these represent. Thanks.

#166 - 07/11/2019 01:15 AM - Eric Faulhaber

Code review task branch 3809c, rev 11326-11375 (part 4):

RecordBuffer:

- Please use proper logging instead of e.printStackTrace in copyDMO.
- Why did you remove the 5365 error from copy at line 3614 (rev 11326) / 3730 (rev 11375)? Did you find conclusively it cannot be raised in the method form of BUFFER-COPY?

TemporaryBuffer: Multiple cases of printStackTrace need to be replaced with proper logging or error handling.

XmlData needs a header entry.

JsonExport:

- Does the note at line 469 refer to legacy behavior or to a FWD deviation?
- Please update the comment at line 612, which is no longer accurate, since a different method is now in use.

TempTableSchema needs a header entry.

XmlExport:

- Needs a header entry.
- The width of INDENT has changed from 3 to 2, so either that was a mistake or the javadoc is now wrong.
- The javadoc at line 452 will almost certainly become a mess. Maybe that comment needs to move into the code itself.

Has there been any regression testing of "classic" TEMP-TABLE JSON/XML export/import, now that the implementations have changed significantly to accommodate datasets?

Greg, please review the non-persist changes. I read over them, but you should have a look.

Ovidiu, the dataset implementation is an impressive bit of work. I honestly can't say I fully understood all of it; it is just too complex to fully digest from a code review. It is clear that you have done a lot of work and put a lot of thought into this implementation. Well done! I do have some performance concerns, but we will need to do some testing and profiling to see if these are warranted. I will document them separately.

#167 - 07/11/2019 01:25 AM - Eric Faulhaber

Performance/Resource Concerns:

As noted earlier, these are things that stood out to me, but I obviously haven't done any profiling at this point. If they need to be addressed, it should be in a different branch, after the 3809c has been merged to trunk.

AbstractTempTable: we instantiate the changeTables HashSet for all cases, not just dataset use. Similar concern in deleteAll. Ideally, the dataset implementation should have zero overhead in the non-dataset use cases. This means not creating/using data structures that are not needed in the non-dataset cases.

BufferImpl:

- How often are acceptRowChanges and rejectRowChanges called? Are these only called by business logic? They use the findByRowid method, which looks expensive.
- Please don't duplicate calls to buffer() within a method. This goes through the proxy's invocation handler, and while it returns early, there is no reason to incur this overhead multiple times when we can just store the result of the first call in a local variable.
- Anything inside the Consumer lambdas in mergeChangesImpl should be as efficient as possible. We should use something more efficient than BufferImpl.findByRowid (maybe a prepared statement that can be re-used for each row?). Where possible, we should avoid using business logic-facing APIs for internal work, and prefer internal workers which don't do the same setup that might be needed to interface with legacy logic.

DataRelation:

- createQuery: although we cache at various levels in the dynamic query conversion infrastructure, perhaps it is worth a quick check to determine whether query is not null and avoid generating it if this.whereString already equals the passed in where value?

RecordBuffer:

- copyCommonFields seems to do work that is redundant with PropertyHelper in terms of identifying methods. Please use PropertyHelper to obtain the methods that need to be used, creating a new mapping if an existing one does not fit the exact need of this copy method. This will take advantage of caching the method collection across sessions, so this method identification work does not need to be done for every invocation.

#168 - 07/11/2019 10:39 AM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> Code review task branch 3809c, rev 11326-11375 (part 3):
>
> CallbackData: Should geTarget be getTarget? It is called from Dataset and BufferImpl.

Indeed. Fixed.

> DataRelation:
>
> - dataSet and strPairs parameters in c'tor need javadoc.

Done.

- getCurrentQuery method variants use UnimplementedFeature. What is the plan to implement these?

Ooops, I forgot about them. There is a QUERY attribute which was already implemented. In ABL reference I see that these is related to bound .NET grid. I added them for completeness, not sure at this moment if they are to be implemented.

- createQuery has the comment, "this is just heuristic: must be investigated". To what does this refer?

It means the method (and its logic) was created as a result of my observations using the black box principle. Basically, this is what I've done with all implementation, but the DataRelation was the first class I added runtime support. Most likely, after this time, the comment is no longer valid and with the addition of setSubstitutionBuffers proved the implementation is very close to what OE does.

> DataSet:
>
> - associate w/ mode parameter is unimplemented. There is a note that suggests it may be implemented in DataSetManager. I guess the other variant would move, too, if you decide to move it?

This is true. It feel logical to move them to manager, but the similar methods for tables/buffer are located in TemporaryBuffer.

- createDynamicDataSet to be implemented in next round of work?

I am adding now appserver support for DATASETs. associate and createDynamicDataSet will certainly change by my final commit today. If the test results will show, these methods will be improved in next rounds of work.

- createLike(character) looks problematic if dsName is unknown value. Similar issues in other variants of this method that take BDT parameters (including the prefix parameter).

That is correct. I will add parameter validations here and investigate the errors OE shows in these cases.

> DataSetManager: scopeDeleted has a TODO to be implemented. What needs to be happen when this is called?

For the moment I haven't tested the persistent procedures. The current implementation is probably not supporting static DATASETs correctly. Their management should be fixed in scopeFinished() and scopeDeleted(), but I haven't the time to write and test this scenario.

> DataSource:

- How much effort to implement attributes?
  - NEXT-ROWID
  - RESTART-ROWID

I believe these are some kind of getter and setter for our TemporaryBuffer$Context.openTables values. They will affect the way we compute the nextPrimaryKey(Class<?> dmoIface). If this is correct, the implementation is straightforward, just get the current value without incrementation, and respectively, put the new value to the map. I do not know if the values are somehow validated.

- FieldReference.toString needs javadoc.

Done.

> Can you please (briefly) explain the concept of the substitution buffers which have been added to the dynamic queries and elsewhere? I read the javadoc, but maybe a 4GL query expression example would make it clearer to me what these represent. Thanks.

A pure 4GL I do not think it is possible to create because their queries do not support such additional buffers. In dynamic buffers only the configured buffers can be used and they will be iterated. A dataset source require for filling that, having the parent fixed on a record, to iterate all children records. For example, if we have the following predicate "WHERE tt-2src.h12=tt-1.f12" with tt-2src buffer being the source and tt-1 a dataset parent buffer that was already (partially) FILL ed and loaded with a current record. We want to find all tt-2src records that match the predicate and save to tt-2. We cannot setBuffers(tt-2src, tt-1) to our query because the query will iterate both buffers. OTOH, setting only setBuffers(tt-2src), the query will fail to convert because tt-1 is unknown. I added the setSubstitutionBuffers method which allows the converter to accept these buffers as substitutions. They are not iterated, but the current record can be accessed in SUBST nodes.

#169 - 07/11/2019 01:10 PM - Eric Faulhaber

Ovidiu, FYI, I have committed in rev 11380 some very minor format and typo cleanups that I made inline during my code review.

Greg, you may want to have a look at the progress.g changes since rev 11326, at minimum.

#170 - 07/11/2019 01:57 PM - Greg Shah

Code Review Task Branch 3809c Revisions 11325 through 11380

I only reviewed the changes in util/ and uast/. I'm generally OK with the changes. Some items:

1. I wonder about the safety of the handle.possibleChained changes. I think it is OK, but I worry that we use handles in so many different ways that there may be some usage which is broken by this. I'm most worried about cases where we may use handle internally.

2. With this much improvement in ProDataSet support, I'm surprised that the rules/gaps/* don't contain more updates. Surely there is quite a bit more that is full or partial or basic.

3. I **really like** the new replaceTokens() approach in ErrorManager. It is a big improvement!

4. I checked in rev 11381 with some typos fixed.

#171 - 07/11/2019 01:57 PM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> Code review task branch 3809c, rev 11326-11375 (part 4):
> RecordBuffer:
>
> - Please use proper logging instead of e.printStackTrace in copyDMO.

The error should be ignored. I logged it at FINE level.

- Why did you remove the 5365 error from copy at line 3614 (rev 11326) / 3730 (rev 11375)? Did you find conclusively it cannot be raised in the method form of BUFFER-COPY?

Error 5365 (*Source element of a BUFFER-COPY statement has no record*) is generated only from statements, not methods. The error is still thrown in static void copy() (line 3456) which is the BUFFER-COPY statement implementation, but I removed from the pair method.

> TemporaryBuffer: Multiple cases of printStackTrace need to be replaced with proper logging or error handling.

Done.

> XmlData needs a header entry.

Done.

> JsonExport:
>
> - Does the note at line 469 refer to legacy behavior or to a FWD deviation?

Legacy. Am FWD mimics this.

- Please update the comment at line 612, which is no longer accurate, since a different method is now in use.

Done.

TempTableSchema needs a header entry.

Done.

XmlExport:

- Needs a header entry.

Done.

- The width of INDENT has changed from 3 to 2, so either that was a mistake or the javadoc is now wrong.

The 4GL output files use a 2-space indentation. I updated javadoc.

- The javadoc at line 452 will almost certainly become a mess. Maybe that comment needs to move into the code itself.

Moved.

Has there been any regression testing of "classic" TEMP-TABLE JSON/XML export/import, now that the implementations have changed significantly to accommodate datasets?

I tried to keep the original code as much as possible, but get the output to match the files generated by 4GL. Unfortunately Jackson does not offer customization for all elements, for example, the xml header always use ', but 4GL write them with ". How can I test for regressions? Are-there any testcases to run?

#172 - 07/11/2019 02:19 PM - Ovidiu Maxiniuc

Greg Shah wrote:

> Code Review Task Branch 3809c Revisions 11325 through 11380
> I only reviewed the changes in util/ and uast/. I'm generally OK with the changes. Some items:
>
> 1. I wonder about the safety of the handle.possibleChained changes. I think it is OK, but I worry that we use handles in so many different ways that there may be some usage which is broken by this. I'm most worried about cases where we may use handle internally.

Evidently, I tried to detect chaining in generated code and use the alternate error message for these cases. This is the single difference. I do not think there are cases in FWD when somebody expects a 4052 for an invalid error and instead with this change will get a 10068 but, indeed, this is a change that may cause subtle effects that I am not aware yet.

> 2. With this much improvement in ProDataSet support, I'm surprised that the rules/gaps/* don't contain more updates. Surely there is quite a bit more that is full or partial or basic.

Certainly, there are lots of changes to be added there. The problem is time and the fact that the mapping are to PPTT constants, which started to be very cryptic. I will probably need a couple of hours to search them and update their states.

> 3. I **really like** the new replaceTokens() approach in ErrorManager. It is a big improvement!

Thanks. There is a formatting issue for the moment, because some are rather long and probably need to be splitted. The same for the comments I added for any reader/maintainer.

#173 - 07/11/2019 02:26 PM - Greg Shah

> The problem is time and the fact that the mapping are to PPTT constants, which started to be very cryptic. I will probably need a couple of hours to search them and update their states.

I'm OK if you want to make the gap edits in 4069a. I plan to rebase to trunk as soon as you have merged to trunk, so it would be fine to edit the gaps

there tomorrow. This way the gap marking doesn't block your getting this merged to trunk tonight.

Has this passed regression testing recently enough?

#174 - 07/11/2019 02:44 PM - Eric Faulhaber

If the branch passes ChUI regression testing, I am good with it being merged to trunk.

#175 - 07/11/2019 02:51 PM - Ovidiu Maxiniuc

Greg Shah wrote:

> The problem is time and the fact that the mapping are to PPTT constants, which started to be very cryptic. I will probably need a couple of hours to search them and update their states.

> I'm OK if you want to make the gap edits in 4069a. I plan to rebase to trunk as soon as you have merged to trunk, so it would be fine to edit the gaps there tomorrow. This way the gap marking doesn't block your getting this merged to trunk tonight.
>
> Has this passed regression testing recently enough?

The last successful test was (3809c_11349) on 06/20/2019 13:14:28. However, the overnight (3809c_11376) run failed with a NPE. I have fixed it in 11382 and I am re-starting now the tests.

#176 - 07/12/2019 02:02 PM - Constantin Asofiei

Runtime testing (main part) passed with 3809c rev 11382.

#177 - 07/12/2019 02:21 PM - Greg Shah

Please merge 3809c to trunk.

#178 - 07/12/2019 02:42 PM - Constantin Asofiei

Branch 3809c passed MAJIC testing and was merged to trunk rev 11325, and archived. It contains changes:

- OM: many changes related to DATA-SET (conversion and runtime). This includes BEFORE/AFTER-TABLE/BUFFER/ROWID, COPY-DATASET, JSON/XML/XMLSCHEMA serialization, MERGE-CHANGES, ORIGIN-ROWID and many more. Refs #3809
- AVK: #4078 issue with Hotel demo login in Swing mode
- HC: adds JSON support via runtime implementation of legacy (previously skeleton) classes. Refs #3751

#179 - 07/12/2019 04:42 PM - Constantin Asofiei

3809d was merged to 4124a rev 11329 and dead-archived.

#180 - 07/13/2019 06:27 AM - Constantin Asofiei

4124a rev 11333 fixes a naming issue with DATASET or TABLE parameters, when the source is in a super-class: for DATASET, now the full reference (for static or instance field) is emitted, instead of renaming the parameter. For buffers, I load the super names in buffer_conflicts.rules (but a similar solution is OK, as the second parameter for .associate is always an instance or static field).

Ovidiu, there is also a DataSetParameter c'tor conversion issue, at the callers, when DATASET-HANDLE is involved: this is emitted twice. I've fixed this in FWD at runtime, by adding a DataSetParameter version of the c'tor, where a DataSet exists. But this needs to be fixed at conversion (the runtime is semantically the same).

#181 - 07/14/2019 07:44 PM - Constantin Asofiei

Ovidiu, please look at 4124a rev 11336 - it fixes the DatasetContainer so that it works with appserver calls.

#182 - 07/19/2019 09:05 AM - Constantin Asofiei

Ovidiu, as we discussed, I need NEXT-ROWID, BATCH-SIZE and RESTART-ROWID to work.

#183 - 07/19/2019 09:53 AM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, as we discussed, I need NEXT-ROWID, BATCH-SIZE and RESTART-ROWID to work.

I switched my focus to them. I hope to get them done this evening, but I will update you in a few hours.

#184 - 07/19/2019 02:46 PM - Constantin Asofiei

The other part which I need is dynamic construction of a DATA-SET from a DataSetContainer received from a remote side, when the 4GL argument is a DATASET-HANDLE.

In the javadoc for DataSetSDOHelper you can find what JavaOpenClient uses for the dataset's table metadata. This info is available in the metadata, but it is not in the DataSetContainer, DsTableDefinition or PropertyDefinition - it needs to be added there, so that it can be transported from a remote, non-FWD side, to FWD.

#185 - 07/19/2019 05:58 PM - Ovidiu Maxiniuc

Ovidiu Maxiniuc wrote:

> Constantin Asofiei wrote:
>
> > Ovidiu, as we discussed, I need NEXT-ROWID, BATCH-SIZE and RESTART-ROWID to work.

The branch 4124a was updated to r11355. The update contains the runtime support for all attributes above-mentioned.

#186 - 07/19/2019 08:38 PM - Constantin Asofiei

I've placed some notes to #3816-34, #3816-35, #3816-36, as that task handles BY-REFERENCE (only for tables). But this is needed for DATASET, too.

#187 - 07/20/2019 08:44 AM - Constantin Asofiei

After fixing #3860-63, to allow runtime-specification of default databases per a project, I get a validation exception during dataset fill: I think the loop which does the record create should be in its own transaction (so records can be flushed properly on iteration). I'll make these changes, but if it works, this needs to be actually proven in 4GL.

#188 - 07/21/2019 07:26 AM - Constantin Asofiei

Ovidiu, why is this code in BufferImpl.fill being called for each record you add in the parent buffer? Because the child buffer gets fully filled with any record, I don't see any limitation to limit the child buffer so that it gets only records related to the current record in the parent. I have a table with 10s of thousands or rows, and it is pretty heavy on performance. I would assume that the child buffers will get filled after the parent does? Or at least limit the child's query to the parent's current record?

```
               // call nested fill on child relation buffers
                for (DataRelation rel : childRelations)
                {
                   rel.getChildBuffer().unwrapBuffer().fill();
                }

                dataSet.invokeCallback(callbacks.get(AFTER_ROW_FILL_EVENT));

                fillQuery.getNext();
```

#189 - 07/21/2019 10:23 AM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, why is this code in BufferImpl.fill being called for each record you add in the parent buffer? Because the child buffer gets fully filled with any record, I don't see any limitation to limit the child buffer so that it gets only records related to the current record in the parent. I have a table with 10s of thousands or rows, and it is pretty heavy on performance. I would assume that the child buffers will get filled after the parent does? Or at least limit the child's query to the parent's current record?
> [...]

The FILL operation is executed recursively, based on the relations defined for the dataset. Once a record is filled for a parent-table, we advance to its child-tables and fill all records in child relation to the parent-record. For each of these the grand-child tables are then filled. Note that the parent record remains loaded in the buffer and during the child-fill operation these parent are used in queries as "substitutionBuffer" (in fact there can be only one, but I let the code be more generic) so that the current value can be read from the parent buffer as a kind of constant for that iteration.
More, for each record filled there are the callbacks that are called, if they are configured. This probably explains the low performance.

#190 - 07/21/2019 01:13 PM - Constantin Asofiei

The structure of the indexes for a table in the JavaOpenClient is described here:
https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/dvjav%2FConstructor_2.html%23wwID0EUVDQ

#191 - 07/21/2019 07:31 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> The structure of the indexes for a table in the JavaOpenClient is described here:
> https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/dvjav%2FConstructor_2.html%23wwID0EUVDQ

I added the numIndexes, indexes, xmlns, and xmlPrefix and created indexes on caller side. The content of the dataset temp-tables is also inserted in the newly created INPUT tables.

Committed in r11360 of 4124a branch.

#192 - 07/22/2019 11:45 AM - Constantin Asofiei

When serializing a DATASET or TABLE to JSON/XML, we need to keep the legacy field names (and table names) in the same case as they appear at the definition (be it a .df table, a static temporary table or a dynamic temp-table). Why: JSON for example is case-sensitive; if we send field names all lowercase (as we do now), there is a chance that the client using this JSON will not be able to read the data. More, for the DataGrid in JavaOpenClient, a record property can be accessed by its name - and AFAIK Tuscany is case-sensitive, too.

Ovidiu, I know we talked about this in the past, but I think now it's time to implement it.

#193 - 07/23/2019 04:50 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> When serializing a DATASET or TABLE to JSON/XML, we need to keep the legacy field names (and table names) in the same case as they appear at the definition (be it a .df table, a static temporary table or a dynamic temp-table). Why: JSON for example is case-sensitive; if we send field names all lowercase (as we do now), there is a chance that the client using this JSON will not be able to read the data. More, for the DataGrid in JavaOpenClient, a record property can be accessed by its name - and AFAIK Tuscany is case-sensitive, too.
>
> Ovidiu, I know we talked about this in the past, but I think now it's time to implement it.

Please see r11369 from branch 4124a. All annotation for historical names of fields, tables and indexes in generated DMO-s are in camel-casing (not lowercase).

#194 - 07/24/2019 05:43 AM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> Please see r11369 from branch 4124a. All annotation for historical names of fields, tables and indexes in generated DMO-s are in camel-casing (not lowercase).

11372 fixes a conversion issue; but I have other rutime issue to this, were we don't compare the names case-insensitively.

#195 - 07/24/2019 11:31 AM - Constantin Asofiei

Ovidiu, there is other issue for DATA-RELATION and DATA-SOURCE: when attaching a DATA-SOURCE, this may or may not have the fields qualified with the table name. Same for DATA-RELATION: the fields may or may not be qualified with the table name.

When DataSource.attach is called, FWD can't map the child relation field with the attach mapping, as it is qualified... please fix this.

#196 - 07/24/2019 11:34 AM - Constantin Asofiei

Related to the legacy field names changes: I have some cases where a PreselectQuery gets converted to RandomAccessQuery, I think because of a wrong index calculated. So there are other cases where the legacy names were assumed to be lowercased.

#197 - 07/24/2019 11:35 AM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, there is other issue for DATA-RELATION and DATA-SOURCE: when attaching a DATA-SOURCE, this may or may not have the fields qualified with the table name. Same for DATA-RELATION: the fields may or may not be qualified with the table name.
>
> When DataSource.attach is called, FWD can't map the child relation field with the attach mapping, as it is qualified... please fix this.

FWD should already support this. Unless there is a bug somewhere.
Please add more details: the mapping strings you found to be faulty. (And the names of the the buffers involved, of course.)

#198 - 07/24/2019 06:13 PM - Ovidiu Maxiniuc

I fixed support for FILL operations with a DATA-SOURCE having a provided query.
Also improved camel-casing support for DATASET related classes.

Committed in 4124a as r11380.

#199 - 07/26/2019 03:22 PM - Ovidiu Maxiniuc

Branch 4124a was updated with following changes:

- BufferImpl has SET-CALLBACK methods implemented;
- DataSet parameters use BY-REFERENCE passing mode;
- a NPE fix in debugging AST trees.

Latest revision is 11386.

#200 - 07/31/2019 04:41 PM - Ovidiu Maxiniuc

Support for REFERENCE-ONLY option and NUM-REFERENCES attribute of DATASET was added in the branch 4124a.
Committed revision 11407.

#201 - 07/31/2019 05:03 PM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> Support for REFERENCE-ONLY option and NUM-REFERENCES attribute of DATASET was added in the branch 4124a.

Does this fix the BY-REFERENCE, too? I mean, any DATASET can be replaced by an argument which is BY-REFERENCE.

#202 - 07/31/2019 05:07 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Does this fix the BY-REFERENCE, too? I mean, any DATASET can be replaced by an argument which is BY-REFERENCE.

Oops, it does not.
I am going to add support right now, it's straightforward now, but I need to do the usual tests to verify it is working fine.

#203 - 07/31/2019 06:30 PM - Ovidiu Maxiniuc

Constantin, please see revision 11409.

#204 - 08/07/2019 04:22 PM - Constantin Asofiei

I think I'm on to something: if the NO-ERROR mode is ON while DATASET:FILL is executed, this gets deactivated if there are any fill callbacks registered (thus executed).

Considering that this 'fill must have a data-source or before-fill callback' doesn't raise an ERROR condition, it means that the caller will never know that something has happened, when we are in batch/appserver mode.

#205 - 08/07/2019 05:23 PM - Constantin Asofiei

Constantin Asofiei wrote:

> Considering that this 'fill must have a data-source or before-fill callback' doesn't raise an ERROR condition, it means that the caller will never know that something has happened, when we are in batch/appserver mode.

This SILENT error mode being disabled by a callback was only a part; the root cause was pretty obvious, but I didn't notice it until now. The ERROR-STATUS:ERROR flag is not set for this kind of error!

#206 - 08/09/2019 11:05 AM - Constantin Asofiei

Ovidiu, some notes about the DATASET import/export; the current WRITE implementation doesn't follow the 'relation tree', where both the XSD portion and the data portion have the tables defined in a parent-child mode, and not a list mode. Same for the READ methods.

Please work on fixing this.

#207 - 08/09/2019 12:21 PM - Constantin Asofiei

Constantin Asofiei wrote:

> Ovidiu, some notes about the DATASET import/export; the current WRITE implementation doesn't follow the 'relation tree', where both the XSD portion and the data portion have the tables defined in a parent-child mode, and not a list mode. Same for the READ methods.
>
> Please work on fixing this.

I'm working on this.

#208 - 08/11/2019 04:10 PM - Constantin Asofiei

There's a problem with a dynamic query:

```
PRESELECT EACH btResultCollectionRow WHERE ROW-STATE(btResultCollectionRow) = ROW-DELETED
```

the ROW-DELETED constant is not resolved.

#209 - 08/11/2019 09:18 PM - Constantin Asofiei

Ovidiu, please work on SAVE-ROW-CHANGES with priority.

#210 - 08/12/2019 04:43 AM - Constantin Asofiei

Ovidiu, another question; for a data-source query like:

```
for each tb1 where ...,
    each tb2 where ...,
    last tb3 no-lock
```

the 'next-rowid' for tb3 gets set, even after the query is 'off-end' - but for last, first, unique, I don't think we need to set the next-rowid. Does this make sense?

#211 - 08/16/2019 08:32 AM - Constantin Asofiei

Ovidiu, after the save-row-changes and the 'next-rowid' in #3809-210, please work on the read/write-xml/json methods. I have some TODOs marked as questions in the related Java files, please look through those.

#212 - 08/19/2019 05:55 PM - Constantin Asofiei

Ovidiu, when serializing a dataset to JSON, what is the order of the records on a child table (associated with a parent record in a relation)? Currently, we just get the query from the relation, but I don't see any sorting applied to it. I would assume the primary index is used.

#213 - 08/20/2019 05:58 AM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, when serializing a dataset to JSON, what is the order of the records on a child table (associated with a parent record in a relation)? Currently, we just get the query from the relation, but I don't see any sorting applied to it. I would assume the primary index is used.

That is a good question. I did not run extended testcases to observe this, up to this moment I noticed only the incorrect order of BEFORE records (apparently these tables have a default index on the surrogate ROW-STATE field), but the AFTER records seem to be in right order.

#214 - 08/21/2019 04:15 AM - Constantin Asofiei

Ovidiu, please stabilize and commit your READ-XML changes; as I mentioned before, the urgent part is reading a single-table dataset (no relations), into a dynamic dataset.

You can work on the more complex cases after this.

#215 - 08/21/2019 12:03 PM - Ovidiu Maxiniuc

Constantin, please see revision 11477. I stabilized it so it should handle the simple xml as the one you sent to me.
I'm going on with missing details.

#216 - 08/21/2019 06:35 PM - Ovidiu Maxiniuc

Constantin, please see revision 11479. Further stabilized and with support for indexes and relations.

#217 - 08/22/2019 09:01 PM - Ovidiu Maxiniuc

Constantin,
Please see revision 11483. Further stabilized. Now the 8.9MB xml you sent me is read successfully and, when written back, it grows to 9.5MB. The round-trip takes about 10 seconds but probably can be halved.

#218 - 08/23/2019 03:14 PM - Constantin Asofiei

Found another issue: bind can be bi-directional, from the destination to source, if the source is BY-REFERENCE. I'm working on fixing it. The effects of this:

- a static dataset (and its associated temp-tables) can survive the defining external program if it was bound at the caller's source; I assume the delete relies on NUM-REFERENCES, even for static datasets, and it will prevent deleting it, even if the external program gets deleted, if this is greater than zero.
- I assume when a by-reference dataset gets unbound (because it was switched to another dataset or if its defining external program gets deleted), it will automatically see that the currently bound dataset has an unknown instantiating procedure, NUM-REFERENCES is zero, and it will delete it

#219 - 08/25/2019 05:09 PM - Constantin Asofiei

Ovidiu, another issue: DATASET:ACCEPT-CHANGES is not working for static datasets, as you are emulating this via the EMPTY-TEMP-TABLE 4GL method on the before-table buffer, which is not allowed for before-tables.

#220 - 08/28/2019 06:09 PM - Ovidiu Maxiniuc

As noted in a previous email, __error-flag__, __origin-rowid__, __error-string__, __after-rowid__ and __row-state__ are true fields for before-buffers and CAN be accessed directly, loke this:

```
buffer tt0::__row-state__
tt0.__row-state__
buffer tt0:buffer-field("__row-state__")
```

However, buffer tt0:buffer-field(n) for any value of integer n will not point to these hidden fields.
They are read-only, meaning that trying to write-access them will generate errors 12375 and 142 (You may not update BEFORE-TABLE %1 record. and ** Unable to update %1 Field.. This is good because the attribute/function remain consistent.

Of course, they return the same result as ROW-STATE function. On the other hand, the after-buffers do not have these fields, but the ROW-STATE function (and attribute) can be applied to them, too. The ROW-STATE function will convert naturally once it is converted into __row-state__ field (like the rowid is also converted to id field). This works for before buffers, but will fail with the after-buffers.

Note that the before-buffers have the __after-rowid__ field, which is used to locate the after image (the current row (with the updates)). There is also an AFTER-ROWID attribute which also does the same thing. The problem is that the symmetrical BEFORE-ROWID works for after-buffers, but there is no __before-rowid__ field. At least I was unable to find it. It is unclear to me how this link is implemented. At this moment, I created true fields in before-buffer and surrogate properties in after-buffer and I am trying to make them work together.

The current solution does the ROW-STATE management so that both after and before image are synchronized. It looks to me that 4GL keeps this information in a single place (the __row-state__ field in before buffer) but again, it's unknown to me how this value is obtained for after buffer as long as the __before-rowid__ (or equivalent) does not exist.

There is one more thing to take care of: when the fields are expanded (in a DISPLAY or UPDATE stmt for example)/copied (although I do not think this is possible because of errors 12375 and 142), these special fields need to be skipped.

#221 - 09/02/2019 06:15 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, another question; for a data-source query like:
> [...]
>
> the 'next-rowid' for tb3 gets set, even after the query is 'off-end' - but for last, first, unique, I don't think we need to set the next-rowid. Does this make sense?

I investigated the issue. I found other issues (related to reopening a provided query, and configuring it to data-source - these are fixed now) but nothing related to next-rowid. It seems to me that FWD works the same as P4GL.

#222 - 09/04/2019 05:51 PM - Constantin Asofiei

Ovidiu, you've added this to dmo_common.rules:

```
<!-- DataSet Buffers support: before-rowid, after-rowid, and row-status. They
     are not normally directly accessible to ABL programmer from SIMPLE and
     AFTER-TABLES, but from BEFORE-TABLES. They are read-only
     and normally accessible using buffer attribute / ABL built-in functions. -->
<action>
   breakpoint("DMO0-FIELDS", false, this, this.dumpTree(true))
</action>
```

This produces lots of noise... what was the reason behind it?

#223 - 09/04/2019 05:53 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, you've added this to dmo_common.rules:
> [...]
> This produces lots of noise... what was the reason behind it?

You can drop it (all the lines). It was just for debug. I saw it, but you have already started the rebase process so I could not drop them myself. Sorry.

#224 - 09/04/2019 05:54 PM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> You can drop it (all the lines). It was just for debug. I saw it, but you have already started the rebase process so I could not drop them myself. Sorry.

The rebase is finished - please clean this up.

#225 - 10/13/2019 01:59 PM - Constantin Asofiei

Ovidiu, please add history entries to the files you modified in 3809e. Thanks!

#226 - 11/08/2019 04:34 AM - Hynek Cihlar

I see 53 failed tests in my last ChUI regression run of 3809e. I'm looking at them.

#227 - 11/11/2019 03:58 AM - Hynek Cihlar

With 3809e revision 11373 all the ChUI regressions should be fixed. I started another run to make sure the changes don't cause new regressions.

#228 - 01/20/2020 04:15 AM - Hynek Cihlar

3809e passed ChUI and GUI regression tests. It was merged to trunk as revision 11340 and archived.

#229 - 01/22/2020 05:42 PM - Greg Shah

- Related to Feature #4514: run ProDataSet test suite in FWD and resolve any issues such that it works the same way as in the 4GL added

#230 - 01/22/2020 05:44 PM - Greg Shah

- Related to Feature #4397: add database attrs, methods and options added

#231 - 01/22/2020 05:44 PM - Greg Shah

- Related to Feature #3574: finish implementation of temp-table XML support added

#232 - 01/22/2020 05:49 PM - Greg Shah

All remaining work (of which I'm aware) related to ProDataSets is documented in these tasks:

- #4514 run ProDataSet test suite in FWD and resolve any issues such that it works the same way as in the 4GL
- #3574 finish implementation of temp-table XML support
- #4397 add database attrs, methods and options

If there is anything else I'm missing, please document it here and I will update this running list. I'm closing this task as we will work the remainders in those other tasks.

Ovidiu: Your work on this task was very impressive! It is a really tricky and large feature set to implement. Well done.

#233 - 01/22/2020 05:49 PM - Greg Shah

- % Done changed from 0 to 100

- Status changed from WIP to Closed

#234 - 11/28/2022 04:02 PM - Ovidiu Maxiniuc

- Related to Feature #6444: dataset improvements added

Files

progress_g&SymbolResolver_java.diff	13.1 KB	03/22/2019	Ovidiu Maxiniuc