Database - Bug #6502

DATASET:CREATE-LIKE copies temp-table options (like XML-NODE-NAME), while TEMP-TABLE:CREATE-LIKE does not

06/07/2022 04:26 PM - Constantin Asofiei

Status:	WIP	**Start date:**	
Priority:	High	**Due date:**	
Assignee:	Ovidiu Maxiniuc	**% Done:**	50%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	
Description			

History

#2 - 06/07/2022 04:27 PM - Constantin Asofiei

See this test:

```
def temp-table tt1 xml-node-name "table1" field f1 as int.
def dataset ds1 for tt1.

def var ht as handle.
def var hds as handle.

create temp-table ht.
ht:create-like(temp-table tt1:handle).
ht:temp-table-prepare("tab1").
message ht:xml-node-name. // tab1

create dataset hds.
hds:create-like(dataset ds1:handle).
message hds::tt1:table-handle:xml-node-name. // table1

message hds::tt1:xml-node-name. // fails in FWD, must show table1 (BUFFER:XML-NODE-NAME is not supported)
```

For some reason, DATASET:CREATE-LIKE copies the temp-table metadata (like XML-NODE-NAME), too, while the TEMP-TABLE version seems to not do this.

#3 - 06/07/2022 04:29 PM - Constantin Asofiei

- Status changed from New to WIP

- Assignee set to Constantin Asofiei

#4 - 06/07/2022 04:44 PM - Constantin Asofiei

- Assignee changed from Constantin Asofiei to Ovidiu Maxiniuc

Ovidiu, please work on this, I thought it will be just as easy to get/set the options, but the getter (like getXmlNodeName()) returns the 'resolved', not the real value. I can't tell how correct this is.

#5 - 06/07/2022 04:45 PM - Ovidiu Maxiniuc

I will do it tomorrow, first thing.

#6 - 06/08/2022 02:01 AM - Constantin Asofiei

A fix is in 6129a/13933. Ovidiu, please do more tests and confirm this is OK (see the TODOs in the fix).

#7 - 06/08/2022 02:12 AM - Constantin Asofiei

6128a/13934 adds unknown protection to the temp-table options copy.

#8 - 06/08/2022 09:16 PM - Ovidiu Maxiniuc

From my tests, so far, the namespace-URI and namespace-prefix attributes are incorrectly copied in 6128a/13934. (lines 6210-6218 in copyTempTableOptions() must be removed).

There is also a copy/paste bug in AbstractTempTable.setMinSchemaMarshal(). At line 928 in setMinSchemaMarshal(boolean on) the line should be:

```
      schemaMarshal = on ? SM_MIN : SM_DEFAULT;
```

I have other changes (including TRPL) but they are risky - I think - now and I will commit them when not so stressed.

#9 - 06/09/2022 08:20 PM - Ovidiu Maxiniuc

I encountered an incorrect optimization issue. Assume we create a copy of a dataset. It will use the DynamicTablesHelper.createDynamicDMO() to create the DMOInfo structure and a CacheData to avoid all TPL processing for creating a new DMO. The CacheData is mapped using CacheKey which is composed on the fields and indexes of the table. However, there is no information on the table.

Next time a new table is created using a very similar structure the cache in DynamicTablesHelper will hit and the old CacheData is returned. It will be used with all old data of the old table, including the table attributes (like the legacy table name which is mandatory in @Table annotation).

I am thinking of adding the table attributes in the CacheKey but I think we will rarely have a cache hit in this case.

A second issue: is it OK to commit a large change set to 6129a (~25 files)?

#10 - 06/10/2022 02:29 AM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> Next time a new table is created using a very similar structure the cache in DynamicTablesHelper will hit and the old CacheData is returned. It will be used with all old data of the old table, including the table attributes (like the legacy table name which is mandatory in @Table annotation).

Why not force setting the legacy table name after the dynamic table is created, like we do for XML-NODE-NAME and others?

> A second issue: is it OK to commit a large change set to 6129a (~25 files)?

Please wait a little to have some downtime with #6277 first, so I can test it.

#11 - 06/10/2022 12:41 PM - Ovidiu Maxiniuc

- % Done changed from 0 to 50

Constantin Asofiei wrote:

> Ovidiu Maxiniuc wrote:
>
> > Next time a new table is created using a very similar structure the cache in DynamicTablesHelper will hit and the old CacheData is returned. It will be used with all old data of the old table, including the table attributes (like the legacy table name which is mandatory in @Table annotation).
>
> Why not force setting the legacy table name after the dynamic table is created, like we do for XML-NODE-NAME and others?

The DmoMeta objects are immutable and populated directly from the annotations of the DMO interface. The problem is the cached values are reused even though they do not fully match the newly constructed dynamic table. I will try to find a solution here.

> > A second issue: is it OK to commit a large change set to 6129a (~25 files)?
>
> Please wait a little to have some downtime with #6277 first, so I can test it.

OK, I will do a partial commit, only the most risk-less files. (like the changes from #6502-8).

#12 - 06/10/2022 12:48 PM - Constantin Asofiei

Another idea for CREATE-LIKE: does this actually build a dynamic DMO? Why not reuse the source DMO, as multiplexing will work, regardless if the source DMO is dynamic or static? Or is it because the legacy name and other annotations?

#13 - 06/10/2022 03:20 PM - Ovidiu Maxiniuc

Yes, it is possible that some attributes can be changed between create-like and temp-table-prepare.

#14 - 06/10/2022 08:34 PM - Ovidiu Maxiniuc

Constantin, please let me know when it is a good time to commit my changes (note that a full reconversion is required after that).

#15 - 07/05/2022 02:35 PM - Ovidiu Maxiniuc

- File #6502+#6450.patch.zip added

Please find attached the current patch for #6502 and #6450.

#16 - 07/06/2022 06:39 AM - Constantin Asofiei

Review for the #6502-15 patch:

- rules/convert/methods_attributes.rules: why was this code removed?

```
               <!-- COMPILER -->
               <rule>methodText.length() == 0 and
                     (htype == prog.kw_compiler or ref.type == prog.var_handle)

                  <!-- ERROR  attribute of the COMPILER handle -->             <!--- this was removed
                  <rule>ftype == prog.kw_error
                     <rule>isAssign
                        <action>methodText = "error"</action>
                        <action on="false">methodText = "error"</action>
                     </rule>
                  </rule>
```

  This prevents COMPILER:ERROR to convert properly.
- rules/convert/expressions.rules, BufferField, BufferFieldImpl, DynamicTablesHelper, TempTableBuilder, XmlNode, DmoMeta, Property, ErrorManager - missing history entry
- AbstractTempTable.numReferences - missing javadoc
- P2jField.computeHash - missing javadoc, method should be in the private section
- P2JIndex has a import org.aspectj.org.eclipse.jdt.core.dom.*; import
- BufferImpl.getDbName has this return new character("PROGRESST"); instead of "PROGRESS" ... is this a typo?
- handle$WorkArea - what is the reason for this change?

```
      private long nextId = 7753 - 1000 + 1000;
```

  The IDs start from 1000 in OE...

I have not tested functional issues in 6129a, as I need to reconvert.

#17 - 07/06/2022 03:08 PM - Ovidiu Maxiniuc

- File #6502+#6450_2.patch.zip added

Constantin Asofiei wrote:

> Review for the #6502-15 patch:
>
> - rules/convert/methods_attributes.rules: why was this code removed?
>   [...]
>   This prevents COMPILER:ERROR to convert properly.

I understand partial support for some COMPILER system handle were added recently and not thoroughly tested. The problem is this new code breaks the normal conversion of error attribute, attempting to use the compiler specific method instead of buffer's error attribute.
My solution (not only for COMPILER, but for other system handles which share common methods and attributes with other objects) is to create a class of this kind and a context/static singleton. We wrap it in a handle access them using the interfaces which needs to be implemented. Here is a small construct to test conversion:

```
bh = buffer tt01:handle.
h1 = COMPILER:handle.
h2 = ERROR-STATUS:handle.
MESSAGE bh:ERROR h1:error
        h1:NUM-MESSAGES h2:NUM-MESSAGES.
```

For the moment I let the error conversion to:

- use direct access if the left-side of : is the COMPILER handle;
- use the unwrap/interface if the left-side of : is a undetermined-type handle (like a buffer).

> - rules/convert/expressions.rules, BufferField, BufferFieldImpl, DynamicTablesHelper, TempTableBuilder, XmlNode, DmoMeta, Property, ErrorManager - missing history entry

Added.

> - AbstractTempTable.numReferences - missing javadoc

Added.

> - P2jField.computeHash - missing javadoc, method should be in the private section

Done.

> - P2JIndex has a import org.aspectj.org.eclipse.jdt.core.dom.*; import

IDE added this automatically. I removed it.

> - BufferImpl.getDbName has this return new character("PROGRESST"); instead of "PROGRESS" ... is this a typo?

This is the value I get when printing DBNAME attribute on OE 11.6 I have installed on test machine. You can check this using:
message buffer tt01:dbname.

> - handle$WorkArea - what is the reason for this change?
>   [...]
>   The IDs start from 1000 in OE...

I will drop this. Just a debugging trick to have a match with the handles from the capture form OE.

> I have not tested functional issues in 6129a, as I need to reconvert.

Thank you for the review. I have attached an updated patch based on your notes.

#18 - 07/07/2022 11:23 AM - Constantin Asofiei

Ovidiu, good catch with COMPILER:ERROR, please commit the patch to 6129a.

#19 - 07/07/2022 12:45 PM - Ovidiu Maxiniuc

#6502+#6450_2.patch.zip was committed in revision 6129a/13955.

Files

#6502+#6450.patch.zip	16.6 KB	07/05/2022	Ovidiu Maxiniuc
#6502+#6450_2.patch.zip	18 KB	07/06/2022	Ovidiu Maxiniuc