Database - Bug #7999

FWD does not honor FIELDS/EXCEPT at dynamic queries

11/02/2023 03:44 PM - Constantin Asofiei

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

Description

Related issues:

Related to Database - Feature #2137: runtime support for FIELDS/EXCEPT record...	**New**

History

#1 - 11/02/2023 03:48 PM - Constantin Asofiei

Consider this test:

```
def var hb as handle.
def var hq as handle.

create buffer hb for table "book".
create query hq.
hq:add-buffer(hb).
hq:query-prepare("for each book fields (isbn) no-lock").
hq:query-open().
hq:get-first().
message hb::book-title. // error
```

For static queries, FIELDS and EXCEPT can appear only at DEFINE QUERY. In dynamic query case, FWD appends them to the OPEN QUERY statement, from where they get ignored.

More, FWD doesn't convert properly the DEFINE QUERY ... FIELDS case, either:

```
define query q for book fields (isbn).
open query q for each book.
get first q.
message book.book-title.
```

This was found in standalone tests.

#2 - 11/02/2023 03:54 PM - Constantin Asofiei

- Related to Feature #2137: runtime support for FIELDS/EXCEPT record phrase options added

#3 - 11/15/2023 03:35 AM - Alexandru Lungu

- Status changed from New to WIP

- Assignee set to Eduard Soltan

#4 - 11/17/2023 10:05 AM - Eduard Soltan

Committed on 7999a, revision 14833.
- DynamicQueryHelper, added methods to traverse fields and except phrases of record phrase of the query string, and set include and exclude variables of AbstractQuery.
- in CompundQuery, added methods to set include and exclude of child components.

#5 - 11/17/2023 12:43 PM - Alexandru Lungu

- Status changed from WIP to Review

- % Done changed from 0 to 100

#6 - 11/17/2023 12:49 PM - Greg Shah

Ovidiu: Please review.

#7 - 11/17/2023 06:31 PM - Ovidiu Maxiniuc

Review of 7999a, r14833.

Good job. The goal of the task was reached. But I have some things to note, mainly because of performance. They may not seem as much, but they add to global application speed:

- CompoundQuery.java:
  - history header: try keeping the information compact. The two entries are highly redundant;
  - there is a common typo in the new method names setIncludedCompoents() / setExcludedCompoents;
  - use an local variable to cache included.get(buffers[j].getDMOProxy()) value used in 2 places;
  - the code in these methods are pretty much identical. I think we can create a parametrised method to merge both of them;
- DynamicQueryHelper.java:
  - history header: see above. The closing tag */ is broken. Luckily it is closed by the copyright comment;
  - the import list contains new unneeded and unwanted entries;
  - line 650 and 663: the cast to AbstractQuery is not needed, the include() and exclude() methods are declared in P2JQuery interface;
  - the for loops at lines 644 and 657 can be replaces with a bit faster iterations over the Map's entry set. For example:

```
Set<Map.Entry<Buffer, String[]>> kvPairs = include.entrySet();
for (Map.Entry<Buffer, String[]> pair : kvPairs)
{
   String[] fields = pair.getValue();

   if (fields != null)
   {
      BufferImpl buffer = (BufferImpl) pair.getKey();
      query0.include(buffer.buffer().getDMOProxy(), fields);
   }
}
```

    The creation and iteration of KeySet and EntrySet are very similar but in the latter case the Map.get() call is replaced by the simple getValue() getter. Also, note the include.get(buffer) was invoked twice in initial code;
  - in methods collectFieldsList() and collectExcludeList():
    - as above they share pretty much of their code so they are candidate for merging into a single one, to avoid code duplication.
    - line 1452: I think the correct code is propList[j] = field.name; (the property name), not fieldAast.getText() (the legacy field name);
    - does collectFieldsList() throw any Exception?
    - the errMsg. Please be sure the message is the same as in OE, including inner spaces and final dot. Please split it on left aligned substrings which do not pass the standard line limit (110);
    - the int nrChildren = chAst.getNumImmediateChildren(); / for (int i = 1; i < nrChildren; i++) / Aast bufParamAast = chAst.getChildAt(i); is not recommended from PoV of performance. getNumImmediateChildren() iterate children to count them, the getChildAt(i) iterates them again. Use instead a iteration starting with getFirstChild() and advance with getNextSibling. This is probably the fastest way. Alternatively, there is getImmediateChild(int type, Aast start) method in AnnotatedAst but it's slower. The same stands for the inner

loop;
- AFAIK, there can be only one KW_FIELD / KW_EXCEPT sub-node of RECORD_PHRASE. In this case it's normal to leave the outer loop as soon as the (first) node was encountered and processed;
- when computing the bufname. Instead of bufname.contains("."), please compute bufname.lastIndexOf(".") first (returns -1 if the needle is not present in the stack), cache it and reuse in substring.

#8 - 01/26/2024 10:06 AM - Alexandru Lungu

Eduard, please address the review to get this done.

#9 - 01/30/2024 04:19 AM - Eduard Soltan

Committed on 7999a, revision 14837.

Addressed review in #7999-7. Also for second case in #7999-1, with DEFINE QUERY ... FIELDS, added support at conversion phase to add include/exclude method call for query variable.

#10 - 01/30/2024 04:37 AM - Alexandru Lungu

Eduard, I rebased 7999a to latest trunk. It is now at rev. 14958. Please do the testing of it with POC and a large customer application regression tests.
I see changes in database_access.rules. Are these mandatory and require reconversion?

Ovidiu, please review latest 7999a.

#11 - 01/30/2024 04:43 AM - Eduard Soltan

Alexandru Lungu wrote:

> I see changes in database_access.rules. Are these mandatory and require reconversion?

It is required for support of DEFINE QUERY ... FIELDS, because for now a fields/except clause inside a DEFINE QUERY is not taken into consideration at conversion phase. And it requires reconversion.
I tested conversion of Hotel_Gui with this change.

#12 - 02/08/2024 10:12 AM - Ovidiu Maxiniuc

Review of 7999a/14954-14958.
The code looks promising, most of the issues below are low priority. but there are a couple of things in DynamicQueryHelper which seem unfinished. Has the code been incorrectly merged when rebased?

- please update the copyright year in file header to all affected files, regardless of the date in H section;
- database_access.rules:
  - line 420: routines exposed in CommonAstSupport are automatically available to TRPL code, no need to create a special worker. Actually, I have not seen it in use.
  - line 1739: variable name, typo bufnameJavanmae instead of bufnameJavaname ?
- CompoundQuery.java:
  - typo: setIncludedCompoents, setFieldsCompoents, nrCompoents
  - line 3487-3492: I guess a more expressive (and compact) initialization for fieldsCompMap would be:
    Map<DataModelObject, Property[]> fieldsCompMap = isInclude ? included : excluded;
  - setFieldsCompoents() can be private I think;
- DynamicQueryHelper.java:

    - line 92: invalid multi-line comment terminator. Does this compile?
    - lines 153, 520, 521, 1309, 1316: rogue empty lines
    - line 1324: @param for token parameter is missing from method javadoc
    - line 1351: we have getImmediateChild(tokenType) but your code is locally optimized to seek a specific token type and I agree with it;
    - line 1368: byLegacyName() requires that the argument to be normalized (lowercase). You may test with an exclude/field name with not matching casing;
    - line 1377: in case of an invalid field name does OE attempt to recover by looking for another list of fields? is it possible to have multiple fields/exclude options?
    - in parse() method, I think something is missing. The include and exclude maps are defined (line 347) and at lines 637/653 they are checked for null. Where is the private method exclude() called?
- PreselectQuery.java: missing H entry

#13 - 02/09/2024 03:59 AM - Eduard Soltan

Ovidiu Maxiniuc wrote:

> - line 1368: byLegacyName() requires that the argument to be normalized (lowercase). You may test with an exclude/field name with not matching casing;

I tested with not matching cases, and it works fine with byLegacyName.

> - line 1377: in case of an invalid field name does OE attempt to recover by looking for another list of fields? is it possible to have multiple fields/exclude options?

I checked in OE, it is not possible to have multiple fields/except options.

> - in parse() method, I think something is missing. The include and exclude maps are defined (line 347) and at lines 637/653 they are checked for null. Where is the private method exclude() called?

It is missing, I do not why it was deleted at some point.

```
include = collectFields(pAst, buffers, ProgressParserTokenTypes.KW_FIELD);
exclude = collectFields(pAst, buffers, ProgressParserTokenTypes.KW_EXCEPT);
```

I tested this changed on POC, and it was causing some regressions:
1) in query 2 consecutive buffers, where referenced to the same persistence table. for each Customer1 fields (name), each Customer2 fields (name), like in this case Customer1 and Customer2 reference the same table Customer.

This issue is solved by changes in FqlToSqlConverter.

2) a issue related to #8259.

Commited on 7999a, revision 14959.