Database - Bug #2095

FIELDS/EXCEPT record phrase options not honored

03/15/2013 12:12 PM - Eric Faulhaber

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Stanislav Lomany	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	Conversion Support for Server Features		
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

Related issues:

Related to Database - Feature #2137: runtime support for FIELDS/EXCEPT record...	**New**
Related to Database - Feature #6459: query features	**New**

History

#1 - 03/15/2013 12:16 PM - Eric Faulhaber

Paraphrased from #2068:

Looks like P2J doesn't like the FIELDS clause, because with it, the fields get emitted as parameters to the *Query clause:

```
find first pers-addr.
for each person fields (emp-num first-name)
   where person.emp-num = pers-addr.emp-num:
end.
```

converted as:

```
query1 = new AdaptiveQuery(person, "person.empNum = ?", null, "person.siteId asc, person.empNum asc", person.g
etEmpNum(), person.getFirstName(), new Object[]
{
   persAddr.getEmpNum()
});
```

We need to convert this. Ideally, I would like to see these emitted to a separate method call after the query is constructed, something like:

```
query1 = new AdaptiveQuery(person, ...);
query1.include("empNum", "firstName");
```

or, if using the EXCEPT option instead of FIELDS:

```
query1 = new AdaptiveQuery(person, ...);
query1.exclude("empNum", "firstName");
```

For a multi-buffer query (e.g., CompoundQuery), these calls would need to come after each query component is added, or be qualified by the buffer

name.

#2 - 03/15/2013 12:41 PM - Eric Faulhaber

The use cases we must support:

```
FOR EACH table1 FIELDS (field-1, field-2, ..., field-n)...

FOR FIRST table1 FIELDS (field-1, field-2, ..., field-n)...

FOR FIRST table1 FIELDS (field-1, field-2, ..., field-n)...,
    FIRST table2 FIELDS (field-1, field-2, ..., field-n)...

FOR EACH table1 FIELDS (field-1, field-2, ..., field-n)...,
    FIRST table2 FIELDS (field-1, field-2, ..., field-n)...
```

#3 - 03/15/2013 01:20 PM - Eric Faulhaber

- Assignee set to Stanislav Lomany

The runtime API will look like:

```
public void P2JQuery.include(String... fields)
public void P2JQuery.exclude(String... fields)
```

where fields represents the converted field names. These are unqualified, since the FIELD|EXCEPT keyword is an option on the record phrase, which is associated with a specific buffer. So, for a CompoundQuery, each list will be associated with a specific Joinable added as a component to the query. Likewise, there is no ambiguity regarding field-to-buffer association in a single buffer query.

#4 - 03/15/2013 01:27 PM - Eric Faulhaber

BTW, since multi-table queries also implement P2JQuery indirectly, they should throw UnsupportedOperationException for the include/exclude methods with an error message like, "Field list must be defined for an individual query component".

Also, we don't need runtime support for this feature yet, other than possibly storing the list of fields in an instance variable.

#5 - 03/21/2013 11:21 AM - Stanislav Lomany

- Status changed from New to WIP

#6 - 03/21/2013 02:01 PM - Stanislav Lomany

It is possible to reference an element of an extent field. The progress documentation says that "If field is an array reference, the whole array is retrieved even if only one element is specified.", so it is safe to drop the index and leave only the extent field name. Documentation doesn't say anything on how it works in the EXCEPT case, but

```
DISPLAY tt EXCEPT f3[2] WITH FRAME f1.
```

drops all elements of f3, so I'll drop indexes, i.e. EXCEPT(f3[2]) will be converted to except("f3").
Do I need to ensure that there are no duplicates in the converted list (e.g. if we are converting EXCEPT(f3[1] f3[2]))?

#7 - 03/21/2013 02:20 PM - Eric Faulhaber

Stanislav Lomany wrote:

> Do I need to ensure that there are no duplicates in the converted list (e.g. if we are converting EXCEPT(f3[1] f3[2]))?

Yes, please do, it will keep the converted code neater in this case.

#8 - 03/22/2013 10:14 AM - Constantin Asofiei

Stanislav: if it doesn't take you long to build some rules which just drop/hide the FIELDS/EXCEPT nodes (so they will not be emitted), please make this change and send me an update (today would be best). I just want to put it through server folder conversion, to expose any other hidden compile errors.

#9 - 03/22/2013 10:38 AM - Stanislav Lomany

OK

#10 - 03/22/2013 11:36 AM - Stanislav Lomany

- File svl_test20130322a.zip added

Removes FIELDS/EXCEPT nodes.

#11 - 03/22/2013 11:42 AM - Constantin Asofiei

> Removes FIELDS/EXCEPT nodes.

Thanks, I'm starting server folder conversion, hope nothing new will appear.

#12 - 03/22/2013 03:07 PM - Stanislav Lomany

Unless someone has objections, for multi-component queries, I'll make include/exclude methods of the parent query:

```
query0 = new PresortQuery();
query0.enableBreakGroups();
query0.setNonScrolling();
query0.addComponent(tt, ((String) null));
query0.addComponent(tt2, ((String) null));
query0.addSortCriterion(byExpr0);

query0.include(tt, "f1", "f2");
```

#13 - 03/22/2013 03:32 PM - Eric Faulhaber

OK, but please emit this method immediately after the associated addComponent call.

```
    query0.include(tt, "f1", "f2");
```

Are you proposing to change the API to require a Buffer reference as the first parameter? Is this necessary?

#14 - 03/22/2013 04:28 PM - Stanislav Lomany

For a multi-table query the buffer should be specified, but as an option we can have qualified field names:

```
query0.include("tt.f1", "tt.f2");
```

#15 - 03/22/2013 04:34 PM - Eric Faulhaber

Stanislav Lomany wrote:

> For a multi-table query the buffer should be specified, but as an option we can have qualified field names:
> [...]

I don't want to parse qualified field names. If it's easier to specify the buffer, then so be it, but if the include/exclude method is specified immediately after the associated addComponent call, the context is enough for us to know with which buffer the fields are associated (i.e., the last one added). I don't think 2 versions of each include/exclude API are really necessary. If you want to specify the buffer, then do it consistently for single-buffer queries as well. Otherwise, I don't think it's necessary to specify the buffer. Am I missing a use case that this reasoning doesn't support?

#16 - 03/22/2013 05:04 PM - Stanislav Lomany

So if the full list of options is:
1. Add the list of fields to query c'tor / addComponent
2. include(String ...) - should be called immediately after addComponent
3. include(DMO, String ...)

then me personally prefer the 3rd option: for the 1st case we already have a lot of parameters specified in c'tor / addComponent; for the 2nd case I don't like very much the idea of making calls order-dependent.

#17 - 03/22/2013 05:31 PM - Eric Faulhaber

I guess there's one more option; chain the call, as in:

```
query0.addComponent(buffer, ...).include("f1", "f2", ...);
```

or

```
new AdaptiveQuery(buffer, ...).include("f1", "f2", ...);
```

In this case, addComponent will have to return Joinable or P2JQuery.

I was proposing option 2, but I can live with option 3. I still think the include/exclude call should directly follow the associated addComponent call. I think it is easier to understand this way.

Go with your favorite approach.

#18 - 03/22/2013 05:38 PM - Stanislav Lomany

> I still think the include/exclude call should directly follow the associated addComponent call. I think it is easier to understand this way.

For converted code - I agree, but I'm just afraid that someone who is writing code by hands may mess up with the order.

#19 - 03/24/2013 08:00 PM - Stanislav Lomany

- File svl_upd20130325a.zip added

Fixes the cases specified in note #2. Probably fixes all other cases too.

#20 - 03/24/2013 08:04 PM - Stanislav Lomany

- File fields-except.p added

#21 - 03/24/2013 10:43 PM - Eric Faulhaber

The update looks good to me. I am running conversion regression testing on it now.

#22 - 03/24/2013 11:42 PM - Eric Faulhaber

- Status changed from WIP to Review

- % Done changed from 0 to 70

The update has passed conversion regression testing and I have committed it to bzr rev. 10329.

#23 - 03/25/2013 01:01 PM - Stanislav Lomany

The current fix doesn't work for all cases, e.g.:
1. REPEAT PRESELECT.
2. DEFINE QUERY.
Note that FIELDS/EXCEPT is specified for DEFINE QUERY rather than OPEN QUERY, so converted code should probably look like this:

```
TempRecord1.Buf tt = TemporaryBuffer.define(TempRecord1.Buf.class, "tt", false);

final QueryWrapper query0 = new QueryWrapper(false);

public void execute()
{
   externalProcedure(new Block()
   {
      public void init()
      {
         query0.include(tt, "f1");
      }
```

#24 - 03/29/2013 10:26 AM - Eric Faulhaber

- % Done changed from 70 to 100

- Status changed from Review to Closed

#25 - 11/16/2016 11:06 AM - Greg Shah

- Target version changed from Milestone 4 to Conversion Support for Server Features

#26 - 07/28/2022 02:01 PM - Eric Faulhaber

- Related to Feature #6459: query features added

Files

svl_test20130322a.zip	818 Bytes	03/22/2013	Stanislav Lomany
svl_upd20130325a.zip	73.8 KB	03/25/2013	Stanislav Lomany
fields-except.p	2.28 KB	03/25/2013	Stanislav Lomany