Database - Feature #2421

Implement DELETE-RESULT-LIST-ENTRY for PresortCompoundQuery

10/17/2014 02:28 AM - Stanislav Lomany

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

History

#1 - 10/17/2014 02:34 AM - Stanislav Lomany

So far I've found two major problems for this type of query:
1. Cursor is not used for getting next/prev values - only sorted results.
2. Cursor contains unsorted data, while sorted results - sorted.

#2 - 03/29/2016 07:41 PM - Eric Faulhaber

- Target version set to Milestone 11

This probably is not needed for M11, but let's fix this just in case.

#3 - 04/16/2016 01:06 PM - Eric Faulhaber

- Assignee set to Stanislav Lomany

- Start date deleted (10/17/2014)

#4 - 04/27/2016 09:55 AM - Stanislav Lomany

- Status changed from New to WIP

Eric, first of all, why do we need PresortCompoundQuery.sorted? Isn't it better so store results in DynamicQuery.cursor?

#5 - 04/27/2016 10:08 AM - Eric Faulhaber

They are different APIs. I needed the API of a Results object. Why is it better to store results in DynamicQuery.cursor?

#6 - 04/27/2016 10:29 AM - Stanislav Lomany

Cursor holds common iteration/reposition/row deletion/etc code and, considering this, synchronization between these two set is tough and IMO excessive.

#7 - 04/27/2016 12:30 PM - Eric Faulhaber

What is your estimate of the effort to replace the implementation vs. making DELETE-RESULT-LIST-ENTRY work with the current implementation, considering it is critically important to regress neither functionality nor performance?

#8 - 04/28/2016 10:02 AM - Stanislav Lomany

PresortCompoundQuery is a compound query with 1. support for break groups / accumulation 2. presorted up front; so there is not actually much code in the PresortCompoundQuery so I would say that replacing implementation has low efforts and shouldn't bring any problems.

And the current implementation - I think it is only in place because we have very few PresortCompoundQueries and it is not used as a standalone query (only for FOR cycle). If we'll start to dig it, we'll find problems. DELETE-RESULT-LIST-ENTRY invloves updating break rows, and how and why we should sync sorted and unsorted sets - I don't know.

#9 - 04/28/2016 10:19 PM - Eric Faulhaber

Stanislav Lomany wrote:

> PresortCompoundQuery is a compound query with 1. support for break groups / accumulation 2. presorted up front; so there is not actually much code in the PresortCompoundQuery so I would say that replacing implementation has low efforts and shouldn't bring any problems.

That may be true (the parent class does most of the work), but I'm not clear on what you want to do, exactly. The cursor instance variable already is used to preselect the (unsorted) results. How do you intend to use it to store the sorted results, too? This sounds like it could potentially be expensive, requiring an additional copy to a temporary holding area (and changes to the parent).

> And the current implementation - I think it is only in place because we have very few PresortCompoundQueries and it is not used as a standalone query (only for FOR cycle). If we'll start to dig it, we'll find problems. DELETE-RESULT-LIST-ENTRY invloves updating break rows, and how and why we should sync sorted and unsorted sets - I don't know.

Sorry, I don't understand. Do you mean the current implementation of DELETE-RESULT-LIST-ENTRY or the current implementation of PresortCompoundQuery? And what do you mean it is not used as a standalone query? IIRC, don't we convert to it in the cases of:

- FOR [EACH|FIRST|LAST] ... [EACH|FIRST|LAST] ... BREAK BY ... and
- OPEN QUERY FOR [EACH|FIRST|LAST] ... [EACH|FIRST|LAST] ... BREAK BY ... (as the delegate of a QueryWrapper)?

Anyway, I don't think the unsorted results are important, only the sorted results are. Having the unsorted results is a detail of our implementation. You will have to write test cases to determine how DELETE-RESULT-LIST-ENTRY interacts with this.

#10 - 04/29/2016 11:23 AM - Stanislav Lomany

Eric, BREAK BY applies only to cycles. IIRC we can convert to presort compound queries some other rare types of queries (I don't remember exactly but it might be queries with some complex where clause). However I didn't see any queries converted to presort compound in any customers' applications. So I don't think DELETE-RESULT-LIST-ENTRY actually ever applied to this type of queries.

#11 - 04/29/2016 01:34 PM - Eric Faulhaber

There are actually a few dozen instances of PresortCompoundQuery in the statically converted code of the current project. However, DELETE-RESULT-LIST-ENTRY is not used with any of these. I'm more concerned about dynamically converted queries. This conversion outcome is very rare, but if it's possible to specify a dynamic query which will convert to PresortCompoundQuery (I don't know of any reason it would not be), we need to support this.

#12 - 04/29/2016 05:16 PM - Stanislav Lomany

OK, for testing we can use queries with complex sorting:

```
open query q for each tt, each book where book.book-id = tt.tt-id by book.book-id + book.cost.
```

So far I can say that reposition doesn't work at all for presort compound.

> That may be true (the parent class does most of the work), but I'm not clear on what you want to do, exactly. The cursor instance variable already is used to preselect the (unsorted) results. How do you intend to use it to store the sorted results, too? This sounds like it could potentially be expensive, requiring an additional copy to a temporary holding area (and changes to the parent).

My plan is:

1. Remove PresortCompoundQuery.sorted
2. Sorting code from SortedResults c'tor will create a sorted set for cursor instead. At the end of sorting, discard the unsorted set in cursor and switch to the sorted set.
3. For next/previous in PresortCompoundQuery use the parent versions (with addition of accumulation).

#13 - 04/29/2016 06:32 PM - Eric Faulhaber

OK, this plan sounds good.

#14 - 05/17/2016 03:55 PM - Stanislav Lomany

Created task branch 2421a from P2J trunk revision 11031.

#15 - 05/19/2016 01:36 PM - Stanislav Lomany

Eric, please consider the case when an inner record of a compound query is updated AND corresponding inner sub-query is preselect one AND the updated field is used in the join expression. E.g.:

```
def temp-table a field a1 as integer 
                 field a2 as integer index idxa a1.

do transaction:
 for each book: delete book. end.

 create book. assign book.book-id = 1   book.book-title = 'Progress Programming'     book.publisher = 'Atlanti
s'  book.isbn = '1-111111-11-1'  book.on-hand-qty = 5     book.cost = 5.00   book.pub-date = 07/31/1997 book.a
uthor-id = 1    book.sold-qty = 3     book.price = 29.95.
 message "created book: " + string(recid(book)).
 create book. assign book.book-id = 2   book.book-title = 'Java Programming'         book.publisher = 'Sun'
    book.isbn = '2-222222-22-2'  book.on-hand-qty = 35    book.cost = 3.00   book.pub-date = 01/01/2000 book.a
uthor-id = 1    book.sold-qty = 1200  book.price = 24.95.
 message "created book: " + string(recid(book)).
 create book. assign book.book-id = 3   book.book-title = 'Ruby Programming'         book.publisher = 'Gems'
    book.isbn = '3-333333-33-3'  book.on-hand-qty = 20    book.cost = 1.00   book.pub-date = 01/01/2004 book.a
uthor-id = 3    book.sold-qty = 350   book.price = 27.00.
 message "created book: " + string(recid(book)).
 create book. assign book.book-id = 4   book.book-title = 'Perl Programming'         book.publisher = 'Gems'
    book.isbn = '4-444444-44-4'  book.on-hand-qty = 1250  book.cost = 10.00  book.pub-date = 01/01/2003 book.a
```

```
uthor-id = 3    book.sold-qty = 650   book.price = 19.95.
 message "created book: " + string(recid(book)).
 create book. assign book.book-id = 5   book.book-title = 'Python Programming'       book.publisher = 'Animals
'   book.isbn = '5-555555-55-5'  book.on-hand-qty = 23    book.cost = 5.00   book.pub-date = 01/01/2004 book.a
uthor-id = 4    book.sold-qty = 220   book.price = 21.95.
 message "created book: " + string(recid(book)).
end.

def var i as integer.
repeat i = 1 to 5:
  create a. a.a1 = i. a.a2 = i.
end.

def query q for a, book scrolling.
open query q for each a, each book where book.book-id = a.a1 by a.a1 * 2.

find first book where book.book-id = 3.
book.book-id = 999.

reposition q to row 3.
get next q.

if avail(book) then message string(a.a1) + " " + string(a.a2) + " " + string(book.book-id) + " " + string(book
.book-title).
else message "N/A".
```

Because the query is scrolling, we have ids of all records and should load by them. However on practice it works in a bit different way.
Compound query calls Joinable.load for each sub-query. If a record cannot be found in the sub-query cache, sub-results are re-initialized. And since the record participating in the join is updated, sub-set is different and the target record again cannot be found.
AdaptiveQuery.load and PreselectQuery.load have different behaviour: adaptive query loads record by id regardless of sub-set state. I suggest to add the same feature to the preselect query:

```
try
{
   if (!repositionByID(ids, true))
   {
      if (retry)
      {
         // Query did not have the target record(s) cached in its current
         // result set, so reset the results and try again.
         resetResults();
         if (!repositionByID(ids, true))
         {
            throw new MissingRecordException();
         }
      }
      else
      {
         throw new MissingRecordException();
      }
   }

   fetch(offEnd == OffEnd.NONE, lockType, false);
}
catch (MissingRecordException exc)
{
++ try
++ {
++    // The target record doesn't present in the result set, just load by id.
++    coreFetch(data, lockType, false);
++ }
++ catch (MissingRecordException exc2)
++ {
      setEmptyBuffersUnknown();

++    throw exc2;
++ }
}
finally
{
   betweenRows = false;
}
```

Let me know if you foresee any problems with that.

#16 - 05/19/2016 02:34 PM - Eric Faulhaber

Stanislav Lomany wrote:

> Let me know if you foresee any problems with that.

PreselectQuery was not originally meant to adapt to change, which is probably why they differed in this way. I am wondering how we end up with a PreselectQuery instead of AdaptiveQuery for the inner query component in this test case. Does the complex sorting drive this?

#17 - 05/19/2016 02:43 PM - Stanislav Lomany

Yes, it is.

#18 - 05/19/2016 03:26 PM - Eric Faulhaber

I don't foresee an issue from the proposed change.

#19 - 05/23/2016 02:40 PM - Stanislav Lomany

Eric, for the case: am I right that DELETE-RESULT-LIST-ENTRY and {FIRST|LAST}[-OF] cannot be used on the same query (i.e. it is cycle with break groups OR a query without break groups)?

#20 - 05/23/2016 09:30 PM - Eric Faulhaber

Stanislav Lomany wrote:

> Eric, for the case: am I right that DELETE-RESULT-LIST-ENTRY and {FIRST|LAST}[-OF] cannot be used on the same query (i.e. it is cycle with break groups OR a query without break groups)?

Sorry, I don't know. Please confirm with a test case.

#21 - 05/24/2016 03:41 PM - Stanislav Lomany

Eric, there is the conversion issue for validation messages. Testcase:

```
for each book: delete book. end.
create book. assign book.book-id = 1   book.book-title = 'Progress Programming'     book.publisher = 'Atlantis
'  book.isbn = '1-111111-11-1'  book.on-hand-qty = 5     book.cost = 5.01   book.pub-date = 07/31/1997 book.au
thor-id = 1    book.sold-qty = 3     book.price = 29.95.

def buffer xbook for book.
```

```
release book.
find first xbook.

delete xbook.
```

4GL output: no output.
P2J output:

```
** No book record is available. (91)
<Validation message here>
```

The problem is that validator for delete xbook references book buffer instead of xbook in the validate expression.
Should I fix this as the part of this task?

#22 - 05/24/2016 03:53 PM - Eric Faulhaber

Stanislav Lomany wrote:

> Should I fix this as the part of this task?

Yes, please do.

#23 - 05/26/2016 02:07 PM - Stanislav Lomany

Rebased task branch 2421a from P2J trunk revision 11035.

#24 - 05/26/2016 02:22 PM - Stanislav Lomany

Please review task branch 2421a revision 11039.

#25 - 05/27/2016 12:15 AM - Eric Faulhaber

Code review 2421a/11039:

The code looks good to me. Please regression test.

#26 - 05/28/2016 04:51 AM - Stanislav Lomany

Rebased task branch 2421a from P2J trunk revision 11038.

#27 - 05/28/2016 06:21 PM - Stanislav Lomany

- Status changed from WIP to Review

Task branch 2421a was merged into the trunk as bzr revision 11039.

#28 - 06/01/2016 10:37 AM - Eric Faulhaber

- % Done changed from 0 to 100

- Status changed from Review to Closed

#29 - 11/16/2016 12:06 PM - Greg Shah

- Target version changed from Milestone 11 to Cleanup and Stablization for Server Features