Database - Feature #2207

Remaining issues for dynamic buffers and dynamic tables

12/09/2013 12:25 PM - Stanislav Lomany

Status:	WIP	**Start date:**	12/09/2013
Priority:	Normal	**Due date:**	
Assignee:	Stanislav Lomany	**% Done:**	30%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**vendor_id:**	GCD

Description

History

#1 - 12/10/2013 09:28 AM - Stanislav Lomany

Remaining issues:
1. Bug which causes abend if there is constraints violation error in two buffers for a dynamic table. Testcase:

```
def var th as handle.
def var bh1 as handle.
def var bh2 as handle.

create temp-table th.
th:add-new-field("field-one", "integer").
th:add-new-index("idx1", true).
th:add-index-field("idx1", "field-one").
th:temp-table-prepare("some-table").

bh1 = th:default-buffer-handle.

create buffer bh2 for table th buffer-name "some-buffer".

bh1:buffer-create.
bh1:buffer-field("field-one"):buffer-value = 1.

do transaction on error undo, leave:
   bh1:buffer-create.
   bh1:buffer-field("field-one"):buffer-value = 1.
end.

do transaction on error undo, leave:
   bh2:buffer-create.
   bh2:buffer-field("field-one"):buffer-value = 1.
end.
```

Stacktrace:

```
[12/10/2013 17:28:24 TMT] (TransactionManager.deferError:SEVERE) {00000001:00000006:syman} <depth = 5; trans_l
evel = 4; trans_label = blockLabel1; rollback_scope = -1; rollback_label = null; rollback_pending = false; in_
quit = false; retry_scope = -1; retry_label = null; ignore_err = false> [label = blockLabel1; type = DO; full
= true; trans_level = TRANSACTION; external = false; top_level = false; loop = false; loop_protection = true;
had_pause = false; endkey_retry = false; next_or_leave = leave; is_retry = false; needs_retry = false; ilp_cou
nt = -1; pending_break = false; properties = 'ERROR'] Error processing master rollback; deferring throw until
block/iteration completes
org.hibernate.AssertionFailure: possible non-threadsafe access to session
        at org.hibernate.action.internal.EntityInsertAction.execute(EntityInsertAction.java:92)
        at org.hibernate.engine.spi.ActionQueue.execute(ActionQueue.java:362)
        at org.hibernate.engine.spi.ActionQueue.executeActions(ActionQueue.java:354)
        at org.hibernate.engine.spi.ActionQueue.executeActions(ActionQueue.java:275)
        at org.hibernate.event.internal.AbstractFlushingEventListener.performExecutions(AbstractFlushingEventL
istener.java:326)
        at org.hibernate.event.internal.DefaultFlushEventListener.onFlush(DefaultFlushEventListener.java:52)
```

```
        at org.hibernate.internal.SessionImpl.flush(SessionImpl.java:1213)
        at com.goldencode.p2j.persist.Persistence.flush(Persistence.java:4225)
        at com.goldencode.p2j.persist.Persistence.flush(Persistence.java:4195)
        at com.goldencode.p2j.persist.Persistence.flush(Persistence.java:3595)
        at com.goldencode.p2j.persist.DMOValidator.checkUniqueAndShare(DMOValidator.java:792)
        at com.goldencode.p2j.persist.DMOValidator.validate(DMOValidator.java:508)
        at com.goldencode.p2j.persist.DMOValidator.validate(DMOValidator.java:413)
        at com.goldencode.p2j.persist.RecordBuffer$ValidationHelper.validateFull(RecordBuffer.java:8795)
        at com.goldencode.p2j.persist.RecordBuffer$ReversibleDelete.rollback(RecordBuffer.java:10150)
        at com.goldencode.p2j.persist.BufferManager$DBTxWrapper.executeReversibleRollbacks(BufferManager.java:
2165)
        at com.goldencode.p2j.persist.BufferManager$DBTxWrapper.rollback(BufferManager.java:2038)
        at com.goldencode.p2j.util.TransactionManager$WorkArea.notifyMasterCommit(TransactionManager.java:5411
)
        at com.goldencode.p2j.util.TransactionManager$WorkArea.access$3000(TransactionManager.java:5134)
        at com.goldencode.p2j.util.TransactionManager.processRollback(TransactionManager.java:4586)
        at com.goldencode.p2j.util.TransactionManager.rollbackWorker(TransactionManager.java:1690)
        at com.goldencode.p2j.util.TransactionManager.rollback(TransactionManager.java:1577)
        at com.goldencode.p2j.util.BlockManager.processCondition(BlockManager.java:8796)
        at com.goldencode.p2j.util.BlockManager.doBlockWorker(BlockManager.java:7783)
        at com.goldencode.p2j.util.BlockManager.doBlock(BlockManager.java:492)
        at com.goldencode.testcases.Test$1.body(Test.java:56)
```

2. Integration of dynamic tables with TableMapper should be finished (see TODOs in DynamicTablesHelper).
3. Implement StaticTempTable - I'm close to finish it, so I'll make it at the first opportunity.
4. Implement copyTempTable for dynamic and static cases.

#2 - 01/26/2014 09:59 AM - Eric Faulhaber

- Assignee set to Stanislav Lomany

#3 - 01/28/2014 10:59 AM - Stanislav Lomany

The first issue is not a dynamic tables issue, it is about undo processing for NO-UNDO temp tables. The same result can be achieved using static tables:

```
def temp-table some-table no-undo
    field field-one as integer
        index idx1 is unique field-one.

define buffer some-buffer for some-table.

create some-table.
```

```
some-table.field-one = 1.

do transaction on error undo, leave:
   create some-table.
   some-table.field-one = 1.
end.

do transaction on error undo, leave:
   create some-buffer.
   some-buffer.field-one = 1.
end.
```

Should I fix it?

#4 - 01/31/2014 07:47 AM - Stanislav Lomany

FYI, there is the problem with COPY-TEMP-TABLE. Consider the tables

```
def temp-table src
      field field-one as integer
      field field-two as integer
      index idx1 is unique field-two.

def temp-table dest
      field field-one as integer
      field field-two as integer
      index idx2 is unique field-one.
```

And records

```
create src.
src.field-one = 33.
src.field-two = 2.

create src.
src.field-one = 33.
src.field-two = 1.
```

In this case COPY-TEMP-TABLE will copy only one record because of unique index idx2(field-one) violation. Moreover, in loose-copy mode some fields may turn to their default values which increases the probability of an unique index violation. So we have two problems:
1. Each record should be validated against the records in the destination table AND already copied records from the sources table. I have to look into validation code to see why it doesn't always validate against already saved records.
2. Which record will be saved and which will be dropped depends on the sorting order in the source table. We solve sorting problem at conversion time, but this time we need a runtime solution. Tests has proved that the sorting order follows the general sorting rules. In our case we have no where clause and sorting follows primary index if it presents or the index with the lexicographically first name. In the above case the record (33, 1) will be saved because the records are ordered by idx1(field-two).

#5 - 01/31/2014 09:24 AM - Eric Faulhaber

Stas, please put your work on the COPY-TEMP-TABLE handle method on hold. As we discussed previously, this method is not used by the current project, so it is out of scope. We only need the underlying copying to work (without the additional modes introduced by the 4GL method), and AFAIK that feature set already is implemented.

Regarding note 3 above, yes, please fix it.

#6 - 02/03/2014 04:03 PM - Stanislav Lomany

About the issue in note 3. What P2J does in this case:

```
do transaction on error undo, leave:
   create some-table.
   some-table.field-one = 1.
end.
```

1. A reversible delete is created for no-undo buffer.
2. A reversible update is not created for no-undo buffer because of constraints violation.
3. After the block the reversible delete is applied and field-one will have default value 0.
4. If we have two blocks of this kind we have constraints violation for field-one (which is 0).

What 4GL does: after the block field-one will have assigned value 1. I.e., unlike P2J, 4GL allows existence of records cannot be committed to database (so any database flush raises error). Overall, it seems that potential fix will heavily affect core persistent processing. Eric, any thoughts?

#7 - 03/12/2014 06:38 AM - Stanislav Lomany

- File svl_upd20140311a.zip added

Fix for review for issue 1 in note 1. The fix is limited to constraints violation case for NO-UNDO buffers.

#8 - 03/12/2014 08:52 PM - Eric Faulhaber

Update 20140311a looks OK to me, but please hold off regression testing for now. I have a large update to RecordBuffer going through testing now. I will want you to merge your changes into that version when it passes.

What cases remain to be handled?

#9 - 03/13/2014 06:46 AM - Stanislav Lomany

> What cases remain to be handled?

1. In the original testcase (violation error for the dynamic table) error is displayed twice (the fix reproduces this behavior), while in the same case for the static table error is displayed eight times, so flush points are not 100% correct in P2J.
2. If we change a field in an invalid record and the value for that field is valid, that shouldn't trigger an error message. Specifically, I suppose that DMOValidator.checkUniqueAndShare should only check that dirty unique indexes which contain the modified property.

#10 - 03/14/2014 11:05 AM - Eric Faulhaber

> 1. In the original testcase (violation error for the dynamic table) error is displayed twice (the fix reproduces this behavior), while in the same case for the static table error is displayed eight times, so flush points are not 100% correct in P2J.

Stas, please have a look at Ovidiu's om_upd20140213a.zip update in #2222. It is not necessarily current with the latest bzr revision, but I believe he has made changes to the flushing to support the proper behavior of WRITE triggers. Let me know if this looks like it will affect (for better or worse) the issue you describe.

#11 - 03/19/2014 10:48 PM - Eric Faulhaber

Stas, the changes you were waiting on based on my note 8 above have been committed to bzr rev. 10493, and there is a newer revision (10494), but that shouldn't conflict with your update. Please merge, regression test, and commit your update if it passes.

#12 - 03/22/2014 02:34 PM - Stanislav Lomany

> Stas, please have a look at Ovidiu's om_upd20140213a.zip update in #2222. It is not necessarily current with the latest bzr revision, but I believe he has made changes to the flushing to support the proper behavior of WRITE triggers. Let me know if this looks like it will affect (for better or worse) the issue you describe.

Ovidiu's code doesn't affect my testcases.

#13 - 03/22/2014 02:42 PM - Stanislav Lomany

- File deleted (svl_upd20140311a.zip)

#14 - 03/22/2014 02:53 PM - Stanislav Lomany

- File svl_upd20140320c.zip added

Committed to bzr revision 10495.

#15 - 03/30/2014 02:56 PM - Eric Faulhaber

- Status changed from New to WIP

- % Done changed from 0 to 30

#16 - 07/31/2014 11:01 AM - Eric Faulhaber

- Target version deleted (Milestone 11)

Since none of these use cases are known to be in the current project, I am unassigning the milestone target.

Files

svl_upd20140320c.zip	86.7 KB	03/22/2014	Stanislav Lomany