Database - Bug #7403

Copy-temp-table with replace-mode should replace already inserted records

06/01/2023 07:38 AM - Alexandru Lungu

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

Description

Related issues:

Related to Database - Feature #7404: Trasform replace-mode into append-mode w...	**Closed**

History

#1 - 06/01/2023 08:10 AM - Alexandru Lungu

This is quite a weird use-case:

```
def temp-table tt field f1 as INTEGER
                  field f2 as INTEGER
                  index idx1 f1 f2 asc.

def temp-table tt2 field f1 as INTEGER
                  field f2 as INTEGER
                  index idx1 is unique f1.

def temp-table tt3 field f1 as INTEGER
                  field f2 as INTEGER
                  index idx1 is unique f1.
do transaction:
    create tt.
    tt.f1 = 1.
    tt.f2 = 2.

    create tt.
    tt.f1 = 1.
    tt.f2 = 3.
end.

temp-table tt2:handle:copy-temp-table(temp-table tt:handle, true).

for each tt2:
    message tt2.f1 tt2.f2. // 1 2
end.

message "===".

temp-table tt3:handle:copy-temp-table(temp-table tt:handle, false, true).

for each tt3:
    message tt3.f1 tt3.f2. // 1 3
end.
```

Note that the results are different for append and replace:

- For append mode, the second record is not inserted as there is already one record satisfying the unique constraint. Therefore, the record is skipped.
- For replace mode, the second record overrides the first record, so that the final result-set has only one entry.

In FWD, this shows tt3 already exists with f1 1. (132) for replace-mode; append-mode works properly. I suspect this is because, even if in replace-mode, FWD code tries to create a new record 1 3 without actually replacing the 1 2.

#2 - 06/13/2023 03:29 AM - Alexandru Lungu

- Related to Feature #7404: Trasform replace-mode into append-mode when target table is empty added

#3 - 06/14/2023 04:42 AM - Alexandru Lungu

- Status changed from New to WIP

- Assignee set to Ștefan Roman

#4 - 06/26/2023 07:58 AM - Ștefan Roman

The error is occurring due to invalidation of the cache. Invalidation takes place after all records have been inserted, this does not affect append mode, but replace mode throws an error. After adding a force invalidation after each insert, the behavior was the same as 4GL. This is not a final solution, I did this to confirm that this is in fact the problem, so I will further investigate why this is happening and how to fix it in a better manner.

#5 - 06/26/2023 08:10 AM - Alexandru Lungu

Ștefan, please investigate why the invalidation doesn't occur when you are actually validating / saving the new record: RecordBuffer.validate or Persistence.save. For save, I really expected to have the cache invalidated under the hood.

#6 - 07/06/2023 09:45 AM - Ștefan Roman

I tried integrating the invalidation with other occurrences of invalidateFFCache but I couldn`t find a proper way, so the only solution was to invalidate manually after every insert operation, if the copyAllRows was in replaceMode.

I committed the change on 7403a, revision 14641.

#7 - 08/22/2023 05:01 AM - Alexandru Lungu

- % Done changed from 0 to 100

- Status changed from WIP to Review

- Assignee changed from Ștefan Roman to Radu Apetrii

I reviewed the changes in 7403a and they are OK.

Radu, please check if this fixes your issue in #7404-32. After, please extract the changes from 7403a and move them to 7404a as a new commit. I will then dead merge 7403a. I want the changes for fast-copy to get merged in one unitary and well-tested effort.

#8 - 08/23/2023 03:06 AM - Radu Apetrii

Alexandru Lungu wrote:

> Radu, please check if this fixes your issue in #7404-32. After, please extract the changes from 7403a and move them to 7404a as a new commit. I will then dead merge 7403a. I want the changes for fast-copy to get merged in one unitary and well-tested effort.

I extracted the changes from 7403a and I tried to apply them, but they were already present in 7404a. They have been added in rev. 14652. Still, I tried the test from #7404-32 with/without that piece of code and the result is the same.

The error (which might not help the investigation) that I'm getting is:

```
** Requested to persist DMO of type com.goldencode.dataset.dmo._temp. Tt2_2_1__Impl__ with recid 2. A differen
t DMO instance of type com.goldencode.dmo._temp.Tt2_2_1__Impl__ is already bound to this session with the same
 recid
** [00000002:0000000D:bogus-->local/_temp/primary] error persisting object
** java.lang.IllegalStateException: [00000002:0000000D:bogus-->local/_temp/primary] no current transaction ava
ilable to rollback
```

#9 - 08/23/2023 03:33 AM - Alexandru Lungu

Radu, please note that fast-copy with REPLACE happens **only if** the destination is empty. This way, append is used instead of replace.