Database - Bug #4926

"guest already exists with ? ?. (132)" error in Hotel GUI

09/28/2020 11:57 PM - Eric Faulhaber

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

Description

History

#2 - 09/29/2020 12:43 AM - Eric Faulhaber

Conversation via email:

> Ovidiu Maxiniuc wrote:
>
> > I am investigating the ** guest already exists with ? ?. (132) from hotel_gui project: apparently a fetched guest record is marked as NEW and an insert is attempted when buffer gets out of scope.
>
> Eric Faulhaber wrote:
>
> > Hm, those should never be marked NEW, but rather READ. Suspicious code that comes to mind is ChangeSet.rollback and any place that calls BaseRecord.initialize.

Ovidiu Maxiniuc wrote:

> I am sorry, I misinformed you. I went deeper with my investigation. The record is not READ, it is a record (based Java @ object id) that was previously saved to local/hotel/dirty database. Because the call comes from DefaultDirtyShareManager:1560 its dmo state is not updated.
>
> The problem is when the dialog is accepted (ON CHOOSE OF Btn_OK IN FRAME gDialog), the first thin is done is:
>
> ```
> FIND FIRST guest WHERE guest.stay-id = stay.stay-id NO-ERROR.
> ```
>
> Well, the guest buffer has the correct record already loaded before execution and the persistence.load() will also find it, but before RAQ.executeImpl() returns it will check the result of getDirtyInfo(). This last method will identify the copy I talked above still having (New Copy) state as in its info result so RAQ.executeImpl() will switch to / return the 'dirty' (meaning coming from dirty database) record. It will be loaded in record buffer.
>
> Fast forward, after the trigger code is executed, the OK button will trigger the close of the dialog/procedure and the guest buffer is discarded. Now it contains the 'dirty' record, still marked as NEW. Based on this, the Persister will attempt to insert it again which will trigger the message we see.
>
> I could not decide if:
>
> - the 'dirty' record is not the correct one to return in this case, or
> - the 'dirty' record NEW flag should be reset once the original record is persisted (from the same buffer).

Records marked with the COPY flag should never be processed as "regular" records with legacy validation/flushing logic, and they should never make it into the "primary" databases. These either originate from the dirty database or as snapshots taken for housekeeping. This limitation should be enforced at the legacy compatibility layer (generally in the persist package, rather than in the persist.orm package). The ORM classes like Session,

Persister, Loader, etc. to the degree possible should be unaware of state related to legacy compatibility or higher level housekeeping and should just do their ORM jobs. There are exceptions where the legacy behavior bleeds into the orm package out of necessity, but for the most part, it is up to the classes which hand records to these ORM classes to restrict what is validated, saved, etc.

It is not clear to me how a record is read from the dirty database with the NEW flag, though IIRC we might do some swapping of the result found in the dirty database with the associated "real" record with the same ID, if we have one in the session cache. But even if so, the combination of the NEW and COPY flags does not seem correct.

#3 - 10/01/2020 12:52 PM - Ovidiu Maxiniuc

The issue is fixed in 3821c/r11641.

#4 - 10/02/2020 07:22 AM - Greg Shah

- % Done changed from 0 to 100

- Status changed from WIP to Closed