Base Language - Feature #2184

create and populate the temp-table on the remote side, based on the received metadata and result set

09/18/2013 09:29 AM - Greg Shah

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Stanislav Lomany	**% Done:**	100%
Category:		**Estimated time:**	32.00 hours
Target version:	Cleanup and Stablization for Server Features		
billable:	No	**vendor_id:**	GCD

Description

Related issues:

Related to Base Language - Feature #1608: implement full appserver support (f...	**Closed**	**02/08/2013**	**05/24/2013**
Blocked by Database - Feature #2122: add runtime support for dynamically prep...	**Closed**	**04/11/2013**	**06/18/2013**

History

#1 - 11/26/2013 11:14 AM - Eric Faulhaber

- Assignee changed from Constantin Asofiei to Stanislav Lomany

#2 - 01/06/2014 03:36 PM - Greg Shah

- Target version changed from Milestone 7 to Milestone 12

This comes into play to dynamically create and/or populate an input temp-table (on the appserver side) or an output temp-table (on the requester 4GL side).

#3 - 01/10/2014 02:59 PM - Eric Faulhaber

- Assignee changed from Stanislav Lomany to Constantin Asofiei

- Target version changed from Milestone 12 to Milestone 5

- Start date deleted (10/07/2013)

#4 - 02/20/2014 01:33 PM - Eric Faulhaber

- Target version changed from Milestone 5 to Milestone 11

- Assignee changed from Constantin Asofiei to Stanislav Lomany

#5 - 04/05/2014 05:03 AM - Stanislav Lomany

Guys, there is an issue with scoping on app server. Consider a simple testcase:

```
def temp-table tt
     field field-one as integer
     field field-two as char.

create tt.
tt.field-one = 3. tt.field-two = "field 3".
create tt.
tt.field-one = 4. tt.field-two = "field 4".

message "appserver external program 2 ran".
for each tt:
   message string(tt.field-one) + "    " + string(tt.field-two).
end.
```

It does the job, but finishes with two warnings and exception:

```
[04/05/2014 12:52:07 TMT] (com.goldencode.p2j.persist.trigger.DatabaseTriggerManager:WARNING) Invalid scope fi
nished.
[04/05/2014 12:52:07 TMT] (com.goldencode.p2j.persist.trigger.DatabaseTriggerManager:WARNING) Invalid scope fi
nished.
[04/05/2014 12:52:07 TMT] (com.goldencode.p2j.util.Agent:WARNING) Agent encountered an error while executing a
 command for appserver app_server
java.lang.NullPointerException
        at com.goldencode.p2j.persist.BufferManager.scopeDeleted(BufferManager.java:808)
        at com.goldencode.p2j.util.ProcedureManager.delete(ProcedureManager.java:1087)
        at com.goldencode.p2j.util.ExternalProgramWrapper.delete(ExternalProgramWrapper.java:735)
        at com.goldencode.p2j.util.Agent$9.execute(Agent.java:1131)
        at com.goldencode.p2j.util.Agent.listen(Agent.java:349)
        at com.goldencode.p2j.util.AgentPool.start(AgentPool.java:409)
        at com.goldencode.p2j.util.AppServerManager.startAppServer(AppServerManager.java:766)
        at com.goldencode.p2j.main.StandardServer.standardEntry(StandardServer.java:265)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:57)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:601)
        at com.goldencode.p2j.util.MethodInvoker.invoke(MethodInvoker.java:76)
        at com.goldencode.p2j.net.Dispatcher.processInbound(Dispatcher.java:693)
        at com.goldencode.p2j.net.Conversation.block(Conversation.java:319)
        at com.goldencode.p2j.net.Conversation.run(Conversation.java:163)
        at java.lang.Thread.run(Thread.java:722)
```

The issue is about the global scope which is popped by scopeFinished, and when scopeDeleted comes into play loadedBuffers has no scopes.

```
public void scopeDeleted()
   {
      Object referent = ProcedureManager.getProcessedProcedure();
      PersistentProcScope scope = persistProcScopes.get(referent);
      // this needs to be executed only when the procedure gets deleted.
      if (referent != null && scope != null && !ProcedureManager._isPersistent(referent))
      {
         // remove all these from the global scope
         loadedBuffers.getDictionaryAtScope(loadedBuffers.size() - 1, true)
                      .keySet().removeAll(scope.loadedBuffers.keySet());
```

Simple loadedBuffers.size() > 0 check works as workaround for me, but warnings remain.
I'm not sure how this code is supposed to work.

#6 - 04/07/2014 03:30 AM - Constantin Asofiei

- File ca_upd20140407a.zip added

Stanislav, you are correct: the loadedBuffers.size() > 0 check is needed in scopeDeleted(). The problem here is that an appserver persistent procedure can be deleted when the connection terminates, and at this time the appserver agent has no active scopes (as the procedure has finished executing). The code wants to remove from the global scope all the buffers associated with the persistent procedure, but in this case, as the global scope is no more, it needs to be a no-op.

The warnings are related to something else: the DatabaseTriggerManager's context-local var needs to be loaded each time the agent initiates or resets its context.

See attached for both fixes. You can integrate them and regression test them with your update.

#7 - 04/10/2014 08:39 AM - Stanislav Lomany

Guys, consider an external persistent procedure and a buffer scoped to that procedure. When we call an internal procedure from that external procedure which references that buffer then we do not have a proper record for that buffer in allBuffers. I see that that contents of allBuffers is saved into PersistentProcScope.allBuffers on external procedure call, but I cannot find where they are restored (should they?) when internal one is called. Note that topmost scope for the external procedure is also deleted.

#8 - 04/10/2014 09:44 AM - Constantin Asofiei

Stanislav Lomany wrote:

> Note that topmost scope for the external procedure is also deleted.

I think the problem is here. When an appserver procedure is ran persistent and its topmost scope is finished, I think BufferManager needs to be aware of this and keep the global (and current?) scope for the ScopedDictionary variables.

When a persistent proc is deleted and BufferManager.scopeDeleted is called, this needs to remove all the data saved in the PersistentProcScope instance from the BufferManager - see the scopeDeleted implementation.

#9 - 04/10/2014 12:49 PM - Stanislav Lomany

> I think the problem is here. When an appserver procedure is ran persistent and its topmost scope is finished, I think BufferManager needs to be aware of this and keep the global (and current?) scope for the ScopedDictionary variables.

The scope which is referenced as "global" in the comments

```
// add all these to the global scope, too
allBuffers.getDictionaryAtScope(allBuffers.size() - 1, true).putAll(scope.allBuffers);
```

is actually the "startup" scope:

```
// add the topmost scope to the TransactionManager
TransactionManager.pushScope("startup", TransactionManager.NO_TRANSACTION,...);
...
// remove the topmost scope from the TransactionManager
TransactionManager.popScope();
```

Where should we "keep" the values - in the global scopes (should we add them?) of scoped dictionaries inside BufferManager or separately in PersistentProcScope?

#10 - 04/10/2014 03:18 PM - Constantin Asofiei

Stanislav Lomany wrote:

> Where should we "keep" the values - in the global scopes (should we add them?) of scoped dictionaries inside BufferManager or separately in PersistentProcScope?

I think we need a real global scope, for the appserver agents, created just below the "startup" scope. The PersistentProcScope instances just keep data about to the buffers which need to be deleted when the procedure gets deleted, I don't want to use it for anything else than that, as will complicate things in BufferManager. Instead, following might work:

1. at the end of Agent.prepare, use TM.pushScope to push an appserver-agent scope (similar to the "startup" scope).
2. in Agent$ResetContextCommand.execute, before calling resetContext, call TM.popScope to pop the appserver-agent scope.

This will allow a global, per-agent, scope, in which the surviving buffers can be "leaked".

#11 - 04/14/2014 04:59 AM - Stanislav Lomany

Constantin, is there a simple way do determine that the specific context is run using an agent?

#12 - 04/14/2014 05:12 AM - Constantin Asofiei

Stanislav Lomany wrote:

> Constantin, is there a simple way do determine that the specific context is run using an agent?

Yes, use AppServerManager.isRemote - it will return true if this context is for an agent.

#13 - 04/15/2014 09:36 AM - Stanislav Lomany

- File svl_upd20140415a.zip added

Update for review.

#14 - 05/07/2014 10:01 PM - Eric Faulhaber

Stas, I apologize it has taken so long for me to review this. Because of this delay, it is no longer in sync with the latest bzr revision; please sync it up.

Constantin: The update looks OK to me, but I am out of my depth with the appserver implications. Once Stas has sync'd to the latest revision, please review also.

#15 - 05/08/2014 07:09 AM - Stanislav Lomany

- File svl_upd20140508a.zip added

Merged with the latest revision.

#16 - 05/08/2014 07:24 AM - Constantin Asofiei

Stanislav Lomany wrote:

> Merged with the latest revision.

Stanislav, the logic looks good to me.

#17 - 05/19/2014 07:33 AM - Stanislav Lomany

- Status changed from New to WIP

- File svl_upd20140519a.zip added

Merged update.

#18 - 05/19/2014 07:34 AM - Stanislav Lomany

- Status changed from WIP to Review

#19 - 05/19/2014 04:31 PM - Eric Faulhaber

- % Done changed from 0 to 100

- Status changed from Review to Closed

#20 - 11/16/2016 12:07 PM - Greg Shah

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

Files

ca_upd20140407a.zip	33.1 KB	04/07/2014	Constantin Asofiei
svl_upd20140415a.zip	137 KB	04/15/2014	Stanislav Lomany
svl_upd20140508a.zip	137 KB	05/08/2014	Stanislav Lomany

svl_upd20140519a.zip	137 KB	05/19/2014	Stanislav Lomany