Database - Bug #2402

ClassCastException thrown when accessing EXTENT fields

10/02/2014 05:42 AM - Ovidiu Maxiniuc

Status:	Closed	**Start date:**	10/02/2014
Priority:	Normal	**Due date:**	
Assignee:	Ovidiu Maxiniuc	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	Cleanup and Stablization for Server Features		
billable:	No	**case_num:**	
vendor_id:	GCD		

Description

History

#1 - 10/02/2014 05:53 AM - Ovidiu Maxiniuc

I found this while investigating queries in #2266. The following code was constructed based real code from a customer application:

```
DEFINE VARIABLE lma AS LOGICAL NO-UNDO INIT YES.
DEFINE VARIABLE lmid AS INTEGER NO-UNDO INIT 0.

DEF TEMP-TABLE t1 FIELD m AS LOGICAL EXTENT 10 INIT YES.
DEF TEMP-TABLE t2 FIELD m AS LOGICAL EXTENT 10 INIT YES.

CREATE t1.
CREATE t2.

IF lmid = 0 THEN
   ASSIGN lmid = ?.

ASSIGN lma = IF AVAILABLE t1 THEN (t1.m[lmid] AND t2.m[lmid]) ELSE FALSE NO-ERROR.

MESSAGE lma.
```

The whole idea is that the ASSIGN expression should evaluate to UNKNOWN because the lmid is UNKNOWN and used as index for an EXTENT field.
The original exception is lost and not printed in logs. It looks like this:

```
Caused by: java.lang.ClassCastException: com.goldencode.p2j.util.int64 cannot be cast to com.goldencode.p2j.ut
il.logical
    at com.goldencode.p2j.persist.$__Proxy2.isM(Unknown Source)
    at com.goldencode.p2j.testcases.AssignFail$1.body(AssignFail.java:2147)
    at com.goldencode.p2j.util.BlockManager.processBody(BlockManager.java:7053)
    at com.goldencode.p2j.util.BlockManager.topLevelBlock(BlockManager.java:6886)
    at com.goldencode.p2j.util.BlockManager.internalProcedure(BlockManager.java:252)
    at com.goldencode.p2j.util.BlockManager.internalProcedure(BlockManager.java:238)
        ...
```

#2 - 10/02/2014 06:16 AM - Ovidiu Maxiniuc

What is wring is that RecordBuffer.Handler.invoke(Object proxy, Method method, Object[] args), in the case of EXTENT fields that have UNKNOWN index, the returned value is the same 1 st argument, on the consideration that it was checked by previous code and it is UNKNOWN.
Indeed, it is UNKNOWN, but the class is wrong - int64 instead of the expected logical.

The solution is, that in the event of an UNKNOWN index, to return the UNKNOWN of the correct class, ie a new logical(), in this case. The UNKNOWN_VALUE branch of the switch should look like this:

```
    return (property != null) ? null : FieldReference.unknownValue(method);
```

#3 - 10/02/2014 11:34 AM - Eric Faulhaber

- Project changed from Bugs to Database

#4 - 10/02/2014 11:38 AM - Eric Faulhaber

- Target version set to Milestone 11

- Status changed from New to WIP

- Assignee set to Ovidiu Maxiniuc

Nice catch. If there is nothing more to do than the solution proposed above, please prepare an update and push it through the normal process.

#5 - 10/02/2014 11:40 AM - Eric Faulhaber

Ovidiu Maxiniuc wrote:

> The original exception is lost and not printed in logs.

Have you determined why it is lost? This seems like valuable information to discard, and suggests we need an update to the infrastructure that discarded it.

#6 - 10/02/2014 12:23 PM - Ovidiu Maxiniuc

The original exception is discarded because it is caused by an abnormalEnd() in the BlockManager so the whole unwind framework will drop it and only print on FINE log-level of the TransactionManager. P2J will proceed with rollback().

#7 - 10/02/2014 12:41 PM - Ovidiu Maxiniuc

- File om_upd20141002a.zip added

Eric Faulhaber wrote:

> Nice catch. If there is nothing more to do than the solution proposed above, please prepare an update and push it through the normal process.

Please review the attached update.
In both OUT_OF_BOUNDS and UNKNOWN_VALUE the returned value should be the same, except that for the latter, no error is generated, even if NO-ERROR is not specified. I understand that this is somewhat normal.

#8 - 10/02/2014 12:50 PM - Eric Faulhaber

Ovidiu Maxiniuc wrote:

> The original exception is discarded because it is caused by an abnormalEnd() in the BlockManager so the whole unwind framework will drop it and only print on FINE log-level of the TransactionManager. P2J will proceed with rollback().

OK, yes, I added that logging to TM recently, for this exact reason. Although it requires turning on FINE logging (which is very noisy), I think that's OK as a temporary requirement when you're diagnosing an abnormal end.

#9 - 10/02/2014 12:58 PM - Eric Faulhaber

- Status changed from WIP to Test

Code review 1002a:

The file is out of date; please merge your change up to bzr rev 10619. Other than that, the change looks good. After you've merged, please regression test and commit/distribute when it passes.

#10 - 10/08/2014 07:40 AM - Ovidiu Maxiniuc

- File om_upd20141002b.zip added

The attached update passed the regression test.
It was committed to bzr as rev. 10622 and distributed by mail.

#11 - 10/14/2014 11:22 AM - Eric Faulhaber

- % Done changed from 0 to 100

- Status changed from Test to Closed

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

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

Files

om_upd20141002a.zip	90.5 KB	10/02/2014	Ovidiu Maxiniuc
om_upd20141002b.zip	90.6 KB	10/08/2014	Ovidiu Maxiniuc