Database - Bug #2097

FOR EACH with multiple, nested CAN-FINDs converts badly

03/18/2013 11:16 AM - Eric Faulhaber

Status:	Closed	**Start date:**	03/17/2013
Priority:	Normal	**Due date:**	
Assignee:	Eric Faulhaber	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	Conversion Support for Server Features		
billable:	No	**case_num:**	
vendor_id:	GCD		

Description

History

#1 - 03/18/2013 11:36 AM - Eric Faulhaber

This issue stems from note 127 in #2068.

The following code:

```
DEFINE TEMP-TABLE tt-pa1 LIKE pers-addr.
DEFINE TEMP-TABLE tt-pa2 LIKE pers-addr.

DEFINE BUFFER b-pers-addr FOR pers-addr.
DEFINE BUFFER b-tt-pa1    FOR tt-pa1.

CREATE tt-pa1.
FIND FIRST person NO-LOCK.
FIND FIRST pers-addr NO-LOCK.

FOR EACH b-pers-addr WHERE b-pers-addr.site-id = person.site-id
                       AND b-pers-addr.emp-num = pers-addr.emp-num,
    EACH b-tt-pa1    WHERE b-tt-pa1.site-id = person.site-id
                       AND (CAN-FIND(FIRST tt-pa2 WHERE tt-pa2.site-id = tt-pa1.site-id
                                                    AND tt-pa2.emp-num = b-pers-addr.emp-num
                                                    AND tt-pa2.active)
                            OR CAN-FIND(address   WHERE address.addr-id = b-tt-pa1.emp-num
                                                    AND address.addr-id = b-pers-addr.emp-num)):
   MESSAGE("whatever").
END.
```

...converts to this (portions relevant to FOR EACH only):

```
WhereExpression whereExpr0 = new WhereExpression()
{
   public Object[] getSubstitutions()
   {
      return new Object[]
      {
         new LogicalExpression()
         {
                           public logical execute()
            {
               return new RandomAccessQuery(ttPa2, "ttPa2.siteId = ? and ttPa2.empNum = bPersAddr.empNum and t
tPa2.active = true", null, (String) null, LockType.NONE).hasAny();
            }
         }
      };
   }
```

```
   public logical evaluate(final BaseDataType[] args)
   {
      return or((logical) args[0], new LogicalExpression()
      {
         public logical execute()
         {
            return new FindQuery(address, "address.addrId = ? and address.addrId = ?", null, (String) null, ne
w Object[]
            {
               new FieldReference(bTtPa1, "empNum"),
               new FieldReference(bPersAddr, "empNum")
            }, LockType.NONE).hasOne();
         }
      });
   }
};

...

RecordBuffer.openScope(bTtPa1, bPersAddr);
query0 = new CompoundQuery(false, false);
RecordBuffer.prepare(bPersAddr);
query0.addComponent(new AdaptiveQuery(bPersAddr, "bPersAddr.siteId = ? and bPersAddr.empNum = ?", null, "bPers
Addr.siteId asc, bPersAddr.empNum asc", new Object[]
{
   person.getSiteId(),
   persAddr.getEmpNum()
}));
RecordBuffer.prepare(bTtPa1);
query0.addComponent(new AdaptiveQuery(bTtPa1, "bTtPa1.siteId = ?exists(from TempRecord2 as ttPa2 where ttPa2.s
iteId = ? and ttPa2.empNum = bPersAddr.empNum and ttPa2.active = true)", whereExpr0, "bTtPa1.siteId asc, bTtPa
1.empNum asc", new Object[]
{
   person.getSiteId(),
   ttPa1.getSiteId()
}));
```

The first component of the CompoundQuery is OK, but the second is just all wrong.

I think the WhereExpression.execute method should return an or expression with 2 FindQuerys, and the RandomAccessQuery in the getSubstitutions method shouldn't be there.

#2 - 03/24/2013 05:10 PM - Eric Faulhaber

I'm working toward what I think the correct conversion would be:

```
WhereExpression whereExpr0 = new WhereExpression()
{
   public logical evaluate(final BaseDataType[] args)
   {
      return or(new FindQuery(ttPa2, "ttPa2.siteId = ? and ttPa2.empNum = ? and ttPa2.active = true", null, (S
tring) null, new Object[]
      {
         new FieldReference(ttPa1, "siteId"),
         new FieldReference(bPersAddr, "empNum")
      }, LockType.NONE).hasAny(), new LogicalExpression()
      {
         public logical execute()
         {
            return new FindQuery(address, "address.addrId = ? and address.addrId = ?", null, (String) null, ne
w Object[]
            {
               new FieldReference(bTtPa1, "empNum"),
               new FieldReference(bPersAddr, "empNum")
            }, LockType.NONE).hasOne();
         }
      });
   }
};

...

RecordBuffer.openScope(bTtPa1, bPersAddr);
query0 = new CompoundQuery(false, false);
RecordBuffer.prepare(bPersAddr);
query0.addComponent(new AdaptiveQuery(bPersAddr, "bPersAddr.siteId = ? and bPersAddr.empNum = ?", null, "bPers
Addr.siteId asc, bPersAddr.empNum asc", new Object[]
{
   person.getSiteId(),
   persAddr.getEmpNum()
}));
RecordBuffer.prepare(bTtPa1);
query0.addComponent(new AdaptiveQuery(bTtPa1, "bTtPa1.siteId = ?", whereExpr0, "bTtPa1.siteId asc, bTtPa1.empN
um asc", new Object[]
{
   person.getSiteId()
}));
```

A bit ugly, but I think it represents the logically correct conversion.

#3 - 03/25/2013 03:48 AM - Eric Faulhaber

- File ecf_upd20130325b.zip added

- File ecf_upd20130325a.zip added

A fix candidate and test cases are attached. I am currently putting the fix through conversion regression testing. It involves quite a lot of change in the very sensitive area of where clause processing, so I am quite nervous about this one.

#4 - 03/25/2013 03:58 AM - Eric Faulhaber

- File deleted (ecf_upd20130325a.zip)

#5 - 03/25/2013 03:59 AM - Eric Faulhaber

- File ecf_upd20130325a.zip added

Missed a bit of cleanup in one of the files.

#6 - 03/25/2013 04:53 AM - Eric Faulhaber

This update regresses many files in Majic; cannot check it in.

#7 - 03/25/2013 05:37 AM - Eric Faulhaber

- File ecf_upd20130325c.zip added

Fix candidate for regressions. Testing now.

#8 - 03/25/2013 09:49 AM - Eric Faulhaber

While there are a number of expected differences, there are also still some regressions, though far fewer than the last pass. This update cannot be checked in.

#9 - 03/25/2013 01:39 PM - Eric Faulhaber

- File ecf_upd20130325d.zip added

Another candidate, in conversion regression testing now.

#10 - 03/25/2013 03:56 PM - Eric Faulhaber

- File ecf_upd20130325e.zip added

Hopefully the final candidate; conversion regression testing it now.

#11 - 03/25/2013 04:31 PM - Eric Faulhaber

- Status changed from New to Review

- % Done changed from 0 to 70

The ecf_upd20130325e.zip update (finally) has passed conversion regression testing; there are some differences in the converted output, but they are expected. The fix is committed to bzr rev. 10331.

#12 - 03/26/2013 02:53 AM - Constantin Asofiei

I have isolated the #2068 note 272 regression like this:

```
def temp-table w-person like person.
```

```
/* this one works */
for each person,
    each w-person where w-person.emp-num = ? or w-person.emp-num <= person.emp-num:
end.

/* this one doesn't work, as client side where expression is emitted and CompareOps.isLessThanOrEqual(BDT, Wra
ppedResource) doesn't exist */
for each person,
    each w-person where person.emp-num = ? or w-person.emp-num <= person.emp-num:
end.
```

If we change the w-person.emp-num <= person.emp-num test with w-person.emp-num = person.emp-num no more compile errors, as there is a CompareOps.isEqual(BDT, WrappedResource). The short-term fix would be to add the CompareOps.isLessThanOrEqual(BDT, WrappedResource), but this would solve only the symptom, not the root cause, as I think there should be no client-side where expression emitted.

LE: how the Java code looks:

```
   WhereExpression whereExpr0 = new WhereExpression()
   {
      public logical evaluate(final BaseDataType[] args)
      {
         return or(isUnknown(person.getEmpNum()), new LogicalExpression()
         {
            public logical execute()
            {
               return isLessThanOrEqual(wPerson.getEmpNum(), new FieldReference(person, "empNum"));
            }
         });
      }
   };
...
            forEach("loopLabel0", new Block()
            {
               CompoundQuery query0 = null;

               public void init()
               {
                  RecordBuffer.openScope(person);
                  query0 = new CompoundQuery(false, false);
                  query0.addComponent(new AdaptiveQuery(person, (String) null, null, "person.siteId asc, perso
n.empNum asc"));
                  RecordBuffer.prepare(wPerson);
                  query0.addComponent(new AdaptiveQuery(wPerson, "wPerson.empNum is null or wPerson.empNum <=
?", null, "wPerson.siteId asc, wPerson.empNum asc", new Object[]
                  {
                     new FieldReference(person, "empNum")
                  }));
               }

               public void body()
               {
                  query0.iterate();
               }
            });

            forEach("loopLabel1", new Block()
            {
               CompoundQuery query1 = null;

               public void init()
               {
                  RecordBuffer.openScope(person);
                  query1 = new CompoundQuery(false, false);
                  query1.addComponent(new AdaptiveQuery(person, (String) null, null, "person.siteId asc, perso
n.empNum asc"));
                  RecordBuffer.prepare(wPerson);
                  query1.addComponent(new AdaptiveQuery(wPerson, (String) null, whereExpr0, "wPerson.siteId as
c, wPerson.empNum asc", new Object[]
                  {
                     new FieldReference(person, "empNum")
                  }));
               }
```

```
            public void body()
            {
               query1.iterate();
            }
         });
```

#13 - 03/26/2013 04:44 AM - Ovidiu Maxiniuc

The reason why the where clause filtered to client-side is probably the where_clause_prep2.rules. You can read in its header:

```
This rule set continues the work of where_clause_prep by further
analyzing WHERE clauses for the purpose of detecting client-side where
clauses and generating partial HQL statements downstream.  Partial HQL
will be derived from any components of a where clause which can be
isolated to reduce the result set which must be processed on the client.
This type of partial restriction of results is only possible if the
where clause is a conjunction expression (using AND) at the root of the
expression tree.  *Disjunction expressions (using OR) must execute
entirely on the client.*
```

This is the case, indeed the disjunction is the root of the expression but I also think (looking at the conditionals forms) that both where clauses could be executed on server-side. In fact, if both sub-trees can be executed server-side then I believe that we can execute both the conjuction and their disjunction on server-side.

#14 - 03/26/2013 09:31 AM - Eric Faulhaber

Ovidiu Maxiniuc wrote:

> The reason why the where clause filtered to client-side is probably the where_clause_prep2.rules. You can read in its header:
> [...]
>
> This is the case, indeed the disjunction is the root of the expression but I also think (looking at the conditionals forms) that both where clauses could be executed on server-side. In fact, if both sub-trees can be executed server-side then I believe that we can execute both the conjuction and their disjunction on server-side.

Sorry, my wording in the header was poor. The idea I meant to convey was that if any part of the expression was deemed to require client-side processing, the fact that that part existed inside an OR expression would pull in the rest of the OR expression to client-side processing as well. In the case of nested OR expressions, it pulls in the top-most OR expression. Even now, my wording on this is not great, but hopefully you get the idea.

#15 - 03/26/2013 09:57 AM - Ovidiu Maxiniuc

I understand. The algorithm is working in fact if you analyze the first clause that works.
And starting from here, I investigated the differences between the two forms.
IIRC, after parsing, the basic node looks something like this:

```
for
  each
    person
  each
    w-person
    where
      or
      ...
```

I see that the the difference is made by the fact that, in a multiple "each buffer" forms of the for statement, the order of the buffers declared is very important and only the inner-most one is treated as the current-buffer. At least for this particular example, nodes that contain the outer-buffer (person) are annotated with related_buffer instead of current_buffer and hql = false. As a consequence, the entire subtree will be marked as client_branch and will end up in client-side whereExprN.
I had not the time to understand exactly why all the above mentioned annotations are added, but I wonder if we shouldn't treat the first each person the same like each w-person and other each buffer from the same level?

#16 - 03/26/2013 03:23 PM - Eric Faulhaber

Ovidiu Maxiniuc wrote:

> ...but I wonder if we shouldn't treat the first each person the same like each w-person and other each buffer from the same level?

No, these buffers come from different databases (person from p2j_test and w-person from _temp, so we have to be very careful when composing HQL to not assume they are similar in that way. We use the current_buffer and related_buffer annotations (among other things) to help us make the determination of how to convert.

In this case, this construct will convert to a CompoundQuery, which joins records on the "client" (i.e., the P2J application server). The reference to person in the inner where clause really should be part of a query substitution parameter, but one that gets re-evaluated at each pass of the outer loop. But the comparison with the unknown literal (person.emp-num = ?) prevents this from converting that way, so it determines that this clause has to be refactored into a client where expression.

Another part of the problem is that person.emp-num is still being treated as a regular, query substitution parameter, so it is emitting as new FieldReference(person, "empNum") for the inner EACH, even though there is no where clause into which to substitute it (because it was moved to a client where expression).

#17 - 03/27/2013 02:20 PM - Eric Faulhaber

- File ecf_upd20130326a.zip added

A fix candidate is attached. Currently running it through conversion regression testing.

#18 - 03/27/2013 03:17 PM - Eric Faulhaber

The update has passed conversion regression testing and is committed to bzr rev. 10333.

#19 - 03/29/2013 09:43 AM - Eric Faulhaber

- % Done changed from 70 to 100

- Status changed from Review to Closed

The fix has addressed the original problem and does not appear to have regressed anything else in the customer's server project, so I'm closing out this issue.

#20 - 11/16/2016 11:06 AM - Greg Shah

- Target version changed from Milestone 4 to Conversion Support for Server Features

Files

ecf_upd20130325b.zip	2.72 KB	03/25/2013	Eric Faulhaber
ecf_upd20130325a.zip	39.5 KB	03/25/2013	Eric Faulhaber
ecf_upd20130325c.zip	39.5 KB	03/25/2013	Eric Faulhaber
ecf_upd20130325d.zip	39.5 KB	03/25/2013	Eric Faulhaber
ecf_upd20130325e.zip	51.2 KB	03/25/2013	Eric Faulhaber
ecf_upd20130326a.zip	13.5 KB	03/27/2013	Eric Faulhaber