Database - Bug #2083

client-side where-clause leak in FindQuery

03/06/2013 01:25 PM - Ovidiu Maxiniuc

Status:	Closed	**Start date:**	03/06/2013
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

History

#1 - 03/06/2013 01:50 PM - Ovidiu Maxiniuc

I tried to simplify the sample. Suppose we have a simple function:

```
FUNCTION f1 RETURNS LOGICAL (d AS DATE).
   RETURN YES.
END FUNCTION.
```

A statement like:

```
MESSAGE CAN-FIND(FIRST Customer
      WHERE Customer.Address <> "COM"
        AND f1(Customer.description)
).
```

when converted, will attempt to call the constructor for FindQuery with following signature:

```
FindQuery(Customer.Buf, String, WhereExpression, String, logical, LockType)
```

I tried to understand how the 5th parameter should be integrated into existing forms of the constructor, to fix the issue by calling the appropriate version of it but later I realized that the actual argument (the result of the evaluation of the function) is not useful at all.
The parameter is in fact an echo of the content of the client-side WhereExpression: f1(customer.getDescription()) and no additional constructors should be implemented but the extra parameter should be removed. I cannot tell at this moment but probably during the client-side where-clause construction the ast remains linked to Query ast as a child.

#2 - 03/06/2013 01:59 PM - Ovidiu Maxiniuc

Here are the converted code for two cases (which attempt to call two new constructors), with and without another joined table:

```
FIND FIRST Book.
MESSAGE CAN-FIND(FIRST Customer 
      WHERE Customer.Address <> Book.book-title 
        AND Customer.Balance <> 0 
        AND (f1(Customer.description) AND f2(Customer.uniq))
).

MESSAGE CAN-FIND(FIRST Customer 
      WHERE Customer.Address <> "COM" 
        AND f1(Customer.description)
).
```

Converted code:

```
message(new FindQuery(customer, "upper(customer.address) != ? and customer.balance != 0", whereExpr0, "custome
r.id asc", and(f1(customer.getDescription()), new LogicalExpression()
{
   public logical execute()
   {
      return f2(customer.isUniq());
   }
}), new Object[]
{
   toUpperCase((character) new FieldReference(book, "bookTitle", true).getValue())
}, LockType.NONE).hasAny());

message(new FindQuery(customer, "upper(customer.address) != 'COM'", whereExpr1, "customer.id asc", f1(customer
.getDescription()), LockType.NONE).hasAny());
```

with the fiollowing where delaratrions:

```
   WhereExpression whereExpr0 = new WhereExpression()
   {
      public logical evaluate(final BaseDataType[] args)
      {
         return and(f1(customer.getDescription()), new LogicalExpression()
         {
            public logical execute()
            {
               return f2(customer.isUniq());
            }
         });
      }
   };

   WhereExpression whereExpr1 = new WhereExpression()
   {
      public logical evaluate(final BaseDataType[] args)
      {
         return f1(customer.getDescription());
      }
   };
```

#3 - 03/06/2013 02:02 PM - Eric Faulhaber

Ovidiu Maxiniuc wrote:

> The parameter is in fact an echo of the content of the client-side WhereExpression: f1(customer.getDescription()) and no additional constructors should be implemented but the extra parameter should be removed.

I think your analysis is exactly correct. The user defined function has been moved correctly to the client-side WhereExpression. The parameter should not be emitted into the FindQuery constructor. It must be that we are making a duplicate of the expression which calls the function, attaching it to the body of the WhereExpression.evaluate method, but then not hiding the original expression we copied. Actually, hiding may not be enough, we may need to remove it.

I currently am working in the same area, so I am curious to know what you find.

#4 - 03/07/2013 05:33 AM - Constantin Asofiei

Another can-find leak:

```
message CAN-FIND (FIRST book WHERE book.book-id > 0 and CAN-DO("A,H",TRIM(book.book-title))). /* this fails */
message CAN-FIND (FIRST book WHERE CAN-DO("A,H",TRIM(book.book-title))). /* this works */
```

#5 - 03/07/2013 09:57 AM - Ovidiu Maxiniuc

I did some patch (removing the extra expression from where node after generating the anonymous inner class needed by client-side expression) which will fix the issue in some cases. Then I start testing with more complex queries. I found another strange conversion issue (may be the same, cannot tell yet).
Here is the source code:

```
FIND FIRST Book.
MESSAGE CAN-FIND(FIRST Customer
      WHERE f1(Customer.description) AND f2(Book.book-title <> "")
```

and the converted code:

```
WhereExpression whereExpr0 = new WhereExpression()
{
   public logical evaluate(final BaseDataType[] args)
   {
      return f1(customer.getDescription());
   }
};

new FindQuery(book, (String) null, null, "book.bookId asc").first();
message(new FindQuery(customer, (String) null, whereExpr0, "customer.id asc", f1(customer.getDescription()), n
ew Object[]
{
   and(f1(customer.getDescription()), new LogicalExpression()
   {
      public logical execute()
      {
         return f2(isNotEqual((character) new FieldReference(book, "bookTitle").getValue(), ""));
      }
   })
}, LockType.NONE).hasAny());
```

You can see that the whereExpr0 is broken, the correct form of it can be found into the 6th parameter, Object[], which in turn should have been empty.

#6 - 03/08/2013 09:09 AM - Ovidiu Maxiniuc

In my previous note there the and operator from where clause contains a reference to a buffer other than the one that the current Query applies.
In the phase that separates server-side from client-side branched, the f2(Book.book-title <> "") is not marked as client_branch or client_tint as the function that do this is only called for nodes marked as "current_buffer".

This is the change in where_clause_prep2.rules, line ~190:

```
...
<rule>
   evalLib("fields") and
   ancestor(prog.kw_where, -1) and
   (!getNoteBoolean("hql") or clientCF)

   <rule>
      parent.type == prog.func_poly and
      #(long) parent.getAnnotation("oldtype") == prog.kw_input
      <action on="false">execLib("mark_client_branch", copy, 1)</action>
   </rule>

   <rule>!getNoteBoolean("current_buffer")
      <!-- related buffer references that are the root of a
           sub-expression are not nested refs BUT a more deeply
           nested related buffer reference IS a nested ref -->
      <rule>
         !clientCF and
         getNoteBoolean("related_buffer") and
         (!isNote("sub_expression") or
          !getNoteBoolean("sub_expression"))
         <action>execLib("mark_client_branch", copy, 2)</action>
      </rule>
      <rule>
         ...
```

Using this substitution object table is clean and all client-side constraints from where clause are moved to respective whereExprN.
At this moment I am not sure how deep is this change because there are a lot of conditionals in the rule and if this modification has any negative side-effects.

#7 - 03/08/2013 12:09 PM - Ovidiu Maxiniuc

I'm getting closer. Here what I found:

The extra node (f1 [FUNC_LOGICAL]) should be made hidden so it will not be emitted in java source. This should happen in where_clause_post.rules: ~715:

```
<rule> parent.type == prog.kw_where and
       (type == prog.expression or evalLib("type_pair", parent.parent.parent, prog.func_logical, prog.kw_can_f
ind))
   <action>copy.setHidden(true)</action>
</rule>
```

However, it seems to me that the node is created some time later, after this rule set is already processed.

#8 - 03/08/2013 12:19 PM - Ovidiu Maxiniuc

At this moment, I use the following simplified testcase:

```
MESSAGE CAN-FIND(FIRST Customer WHERE f1(Customer.description)).

MESSAGE CAN-FIND(FIRST Customer WHERE 
         Customer.Address <> "COM" AND 
         Customer.Name = "gigi").

MESSAGE CAN-FIND(FIRST Customer WHERE 
         Customer.Address <> "COM" AND 
         f1(Customer.description)).
```

The first 2 of them are converted successfully, but not the 3rd (some kind of mix between server and client side).

#9 - 03/09/2013 09:11 AM - Constantin Asofiei

I couldn't get my mind of client_taint and I took a quick look where is used. In where_clause_post.rules, at line 413, there is this rule:

```
      <!-- prune server side only child node of an AND where the other child
           is a client branch;  promote client branch to replace parent AND
           node -->
      <rule>
         parent.type == prog.kw_and                      and
         parent.isAnnotation("client_taint")             and
         #(boolean) parent.getAnnotation("client_taint") and
         ancestor(prog.kw_where, -1)                     and
         evalLib("is_server_branch")
```

This moves a client branch node (like a function call) to be the child of the KW_WHERE node. If you disable this rule, the code converts with no compile errors, but the server-side condition is duplicated at the client-side WhereExpression.

I think this was intended to ensure that the WhereExpression doesn't contain tests which can be done at the server-side.

#10 - 03/11/2013 06:07 AM - Ovidiu Maxiniuc

- File om_upd20130311a.zip added

With help from Constantin, the missing setHidden is now in place.

#11 - 03/11/2013 11:24 AM - Ovidiu Maxiniuc

After some more tests, it seems like this update does not fix the issue, in fact it is too restrictive, it will cut more branches including from the evaluate() of the whereExprN anonymous declarations. Here are some examples:

```
public logical evaluate(final BaseDataType[] args)
{
   return and(or(f2(customer.isUniq()), (logical) args[0]));
}
...
public logical evaluate(final BaseDataType[] args)
{
   return ;
}
```

The code that was converted looks like this:

```
MESSAGE "2" + CAN-FIND(FIRST Customer WHERE
         f1(Customer.description)                                      AND
         Customer.Address <> Book.book-title AND Customer.Balance <> 0 AND
         (f2(Customer.uniq) OR f2(Book.book-title <> "3"))).
```

and

```
FOR EACH Customer WHERE Customer.Address <> "0" AND f1(Customer.description):
END.
```

The f1 and f2 functions are defined like before.

#12 - 03/11/2013 03:32 PM - Ovidiu Maxiniuc

The <action>sibref.setHidden(true)</action> from where_clause_post.rules at line 438 from my previous update seems correct, however, the clause is broken somewhere in the next step in where_clause_post2.rules.
There is an <action>copy.remove()</action> at line 235 (or so) that leads somewhere.
However, there are some extra nodes that are mixed between [CLIENT_WHERE] and [CLIENT_WHERE_SUBST], I mean , args [VAR_LOGICAL] looks wrong placed in [CLIENT_WHERE].

#13 - 03/12/2013 12:19 PM - Eric Faulhaber

Ovidiu, I was having many similar issues with client-side query substitution parameters and CAN-FIND while fixing #2059. Please see if my fix candidate (ecf_upd20130312a.zip, attached to #2059) improves things here. I would recommend trying it with no other changes.

#14 - 03/12/2013 01:36 PM - Ovidiu Maxiniuc

- File om_upd20130312a.zip added

I see you found another solution.

I also found one simple and straightforward solution by refining the condition for hiding the involved node. Please see the attached where_clause_post.rules:438.
It did passed the regression test, just a few minutes ago. Please have a look at it. I hope it does not interfere with your changes.

#15 - 03/12/2013 03:37 PM - Eric Faulhaber

- File om_upd20130312b.zip added

This fix looks good to me. I have merged it with the latest bzr revision, re-run conversion regression testing with the merged version (it passed again), and committed this to bzr rev. 10283. The merged version is attached.

#16 - 04/09/2013 10:58 AM - Eric Faulhaber

Ovidiu, I would like to close out this issue. Please confirm that the test case(s) you were using work correctly with the latest revision of P2J in bzr; upload your test cases here and check them into the testcases project.

#17 - 04/16/2013 08:08 AM - Ovidiu Maxiniuc

- File om_upd20130416a.zip added

Yes, my testcases converts fine. They are attached.
You can close the issue.

#18 - 04/17/2013 10:22 PM - Eric Faulhaber

- Status changed from New to Closed

- % Done changed from 0 to 100

Files

om_upd20130311a.zip	11.2 KB	03/11/2013	Ovidiu Maxiniuc
om_upd20130312a.zip	11.2 KB	03/12/2013	Ovidiu Maxiniuc
om_upd20130312b.zip	11.1 KB	03/12/2013	Eric Faulhaber
om_upd20130416a.zip	1.58 KB	04/16/2013	Ovidiu Maxiniuc