Database - Feature #1588

add conversion and runtime support for embedded SQL

10/11/2012 11:46 PM - Eric Faulhaber

Status:	Closed	**Start date:**	02/14/2013
Priority:	Normal	**Due date:**	05/31/2013
Assignee:		**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	Runtime Support for Server Features		
billable:	No	**vendor_id:**	GCD

Description

Subtasks:

Feature # 2042: add conversion support for some embedded SQL features	**Closed**
Feature # 2043: add runtime support for some embedded SQL features	**Closed**

Related issues:

Related to Database - Feature #2126: add support for table-level locking (sha...	**Closed**	**06/27/2013**	**07/01/2013**
Blocked by Database - Feature #1580: upgrade Hibernate to latest level	**Closed**	**10/23/2012**	**05/03/2013**

History

#1 - 10/11/2012 11:48 PM - Eric Faulhaber

The following use cases need to be supported:

- SELECT COUNT(*) INTO var FROM table WHERE ...
- SELECT SUM(field) INTO var FROM table WHERE ...
- DELETE FROM table

#2 - 10/22/2012 02:13 PM - Eric Faulhaber

- Start date deleted (10/15/2012)

#3 - 10/31/2012 04:09 PM - Greg Shah

- Target version set to Milestone 7

#4 - 02/25/2013 02:34 AM - Eric Faulhaber

- File ecf_upd20130224b.zip added

- File ecf_upd20130224a.zip added

Added support for the simple, single table (and column) constructs listed above. These convert to methods in Buffer.

Code has passed conversion regression testing (not runtime) and is committed to bzr as revision 10199. Update a is code; b is test cases.

#5 - 02/28/2013 12:40 AM - Eric Faulhaber

- File ecf_upd20130227a.zip added

Attached update fixes a defect during annotations processing.

#6 - 02/28/2013 01:48 AM - Eric Faulhaber

20130227a update has passed regression testing and is committed to bzr rev. 10225.

#7 - 04/23/2013 02:16 PM - Eric Faulhaber

In terms of adding runtime support, there are 2 main feature categories to support: aggregate functions (COUNT, SUM, etc.) and bulk delete (DELETE FROM...).

Aggregate Functions

The relevant part of the 4GL example:

```
DEF VAR i AS INT.
DEF VAR t AS INT INIT 10.
COUNT(*) INTO i FROM my-table WHERE my-table.some-field = t.
```

converts to:

```
myTable.count("*", i, "someField = ?", t);
```

It should be pretty straightforward to implement the count method to compose an HQL query...

```
select count(*) from MyTable myTable where myTable.someField = ?
```

and pass it to Persistence.list with t as a substitution parameter. There is nothing to lock.

The use of the other aggregate functions (e.g., SUM, AVG, MAX, etc.) will be very similar, except that instead of "*" as the first parameter, we will have a field name string. This name *already will be the converted form of the field name*, so it can simply be assembled into the HQL. For example, the 4GL:

```
def var i as int.
def var s as int init 3.
select sum(sold-qty) into i from book where book.book-id > s and book.book-title <> ?.
```

...converts to (relevant portion only):

```
book.sum("soldQty", i, "bookId > ? and upper(bookTitle) is not null", s);
```

So, the HQL string would be as follows:

```
select sum(soldQty) from Book book where bookId > ? and upper(bookTitle) is not null
```

This work should be delegated by BufferImpl to RecordBuffer, since the latter contains the DMO interface name and buffer name.

I don't think there are any special buffer scope or application-level [sub]transaction considerations for these queries.

Bulk Delete

The 4GL DELETE FROM table phrase converts to Buffer.deleteAll(). This ultimately should resolve to RecordBuffer.deleteAll(). Currently, this is unimplemented for permanent tables and mostly implemented for temp-tables (via the polymorphic method TemporaryBuffer.deleteAll). The part still missing from the latter is to implement error handling properly (must also be done for the former, when implemented).

For permanent tables, a prerequisite for the bulk delete may be table-level locking (#2126). We need some testing to determine what happens in the 4GL with a DELETE FOR statement when another user session holds a share or exclusive lock on one of the records in the target table. Does it fail entirely? Does it fall back to index-driven behavior, such that it walks a table index (probably the primary index in this case, since there is no WHERE clause), deleting records until it comes upon the locked record?

Assuming that is needed and once that is available, I expect the implementation of RecordBuffer.deleteAll() will look something like this (pseudocode):

```
begin [sub]transaction block
  store current lock type for target table
  acquire exclusive lock on target table
  bulk delete using Persistence.executeSQL with HQL like "delete from MyDMO"
  reset table lock type to previous level
end block
```

...plus error handling code, as appropriate (we may need to change the signature of the deleteAll method to return a logical). Based on the testing specified above, we may also need an alternative code path if we can't acquire the exclusive table lock, such that we fall back to a looping, index-driven approach and leave the table in the same state as the 4GL would.

I added #1580 as a related task because the Persistence APIs used by these features may change with the Hibernate upgrade, but the bulk of this issue's work can be done using the existing API and then retrofitted later.

#8 - 06/12/2013 02:52 PM - Vadim Nebogatov

What about support of COUNT(DISTINCT column)? 4GL supports such expressions as part of obsolete SQL-89, with the following results.

Test example

```
DEF VAR i AS INT.
DEF VAR t AS CHAR INIT "111223333".

SELECT COUNT(*) INTO i FROM person WHERE person.ssn = t.
message "result = " i.
SELECT COUNT(DISTINCT ssn) INTO i FROM person WHERE person.ssn = t.
message "DISTINCT result = " i.
```

returns

```
result =  0
DISTINCT result =  ?
```

or

```
result =  1
```

```
DISTINCT result =  1
```

depending on presence record in table

#9 - 06/12/2013 03:36 PM - Eric Faulhaber

No, we only need runtime support for the forms listed in note 1 above.

Please see the following in the testcases project (under subdirectory uast), which are the programs I used to add the conversion support for this feature:

- sql_count.p
- sql_count2.p
- sql_count3.p
- sql_count4.p
- sql_count5.p
- sql_count6.p
- sql_delete.p
- sql_delete2.p
- sql_delete3.p
- sql_delete4.p
- sql_delete5.p
- sql_sum1.p

#10 - 06/14/2013 06:09 PM - Vadim Nebogatov

My tests for COUNT(*) runtime work for permanent tables, but for temporary tables I receive exception in common-progress.rules:

```
Caused by: com.goldencode.p2j.pattern.CommonAstSupport$UserGeneratedException: Unsupported form of embedded SQ
L [EMBEDDED_SQL id <12884901918> 0:0]

    at com.goldencode.p2j.pattern.CommonAstSupport$Library.throwException(CommonAstSupport.java:2110)
    at com.goldencode.p2j.pattern.CommonAstSupport$Library.throwException(CommonAstSupport.java:2095)
    at com.goldencode.expr.CE3157.execute(Unknown Source)
    at com.goldencode.expr.Expression.execute(Expression.java:341)
```

Are any additional changes in common-progress.rules required for temporary tables?

#11 - 06/14/2013 06:29 PM - Eric Faulhaber

Please upload the test case that is causing this error.

#12 - 06/15/2013 04:49 PM - Vadim Nebogatov

sql_count.p conversion failed with error in #10

```
def var i as int.
def temp-table tt field num as int.
select count(*) into i from tt.
message i.
```

#13 - 06/15/2013 04:50 PM - Vadim Nebogatov

failed in latest revision in bazaar

#14 - 06/18/2013 05:05 PM - Eric Faulhaber

- File ecf_upd20130618a.zip added

The temp-table problem was further back than common-progress.rules. We were only adding the "simple_sql_conversion" annotation for permanent tables, and we needed to add this for temp-table and workfile table types as well. The attached update fixes this. Please merge this update with your own changes for this issue.

#15 - 06/18/2013 06:01 PM - Vadim Nebogatov

Thanks, conversion works now for count(*)

#16 - 06/19/2013 04:13 PM - Vadim Nebogatov

Problem fixed with COUNT(*) runtime in case when WHERE clause is missed because of bug in BufferImpl:

```
public void count(String column, NumberType num, String where)
{
   count(column, num, where, null, (Object[]) null);
}
```

Problem happened inside

```
public void count(String column, NumberType num, String where, Object...args)
{
      buffer().count(column, num, where, args);
}
```

where args was not null as expected but Object[2] {null, null}.

Correct implementation is

```
public void count(String column, NumberType num, String where)
{
   count(column, num, where, (Object[]) null);
}
```

#17 - 06/22/2013 05:20 PM - Vadim Nebogatov

Testing, code commenting aggregate functions runtime in embedded SQL.
cnt, max, min, sum work properly, but there is NPE with avg function inside hibernate.

```
Caused by: java.lang.NullPointerException
        at org.hibernate.dialect.function.StandardAnsiSqlAggregationFunctions$AvgFunction.determineJdbcTypeCod
e(StandardAnsiSqlAggregationFunctions.java:98)
        at org.hibernate.dialect.function.StandardAnsiSqlAggregationFunctions$AvgFunction.render(StandardAnsiS
qlAggregationFunctions.java:92)
        at org.hibernate.hql.internal.ast.SqlGenerator$FunctionArguments.render(SqlGenerator.java:304)
        at org.hibernate.hql.internal.ast.SqlGenerator.endFunctionTemplate(SqlGenerator.java:224)
        at org.hibernate.hql.internal.antlr.SqlGeneratorBase.aggregate(SqlGeneratorBase.java:2456)
        at org.hibernate.hql.internal.antlr.SqlGeneratorBase.selectExpr(SqlGeneratorBase.java:2021)
        at org.hibernate.hql.internal.antlr.SqlGeneratorBase.selectColumn(SqlGeneratorBase.java:1847)
        at org.hibernate.hql.internal.antlr.SqlGeneratorBase.selectClause(SqlGeneratorBase.java:500)
        at org.hibernate.hql.internal.antlr.SqlGeneratorBase.selectStatement(SqlGeneratorBase.java:162)
        at org.hibernate.hql.internal.antlr.SqlGeneratorBase.statement(SqlGeneratorBase.java:111)
        at org.hibernate.hql.internal.ast.QueryTranslatorImpl.generate(QueryTranslatorImpl.java:232)
        at org.hibernate.hql.internal.ast.QueryTranslatorImpl.doCompile(QueryTranslatorImpl.java:203)
        at org.hibernate.hql.internal.ast.QueryTranslatorImpl.compile(QueryTranslatorImpl.java:136)
        at org.hibernate.engine.query.spi.HQLQueryPlan.<init>(HQLQueryPlan.java:105)
        at org.hibernate.engine.query.spi.HQLQueryPlan.<init>(HQLQueryPlan.java:80)
        at org.hibernate.engine.query.spi.QueryPlanCache.getHQLQueryPlan(QueryPlanCache.java:168)
        at org.hibernate.internal.AbstractSessionImpl.getHQLQueryPlan(AbstractSessionImpl.java:221)
        at org.hibernate.internal.AbstractSessionImpl.createQuery(AbstractSessionImpl.java:199)
        at org.hibernate.internal.SessionImpl.createQuery(SessionImpl.java:1735)
        at com.goldencode.p2j.persist.Persistence$Context.getQuery(Persistence.java:3974)
        at com.goldencode.p2j.persist.Persistence.getQuery(Persistence.java:3850)
        at com.goldencode.p2j.persist.Persistence.list(Persistence.java:1504)
        at com.goldencode.p2j.persist.RecordBuffer.aggregate(RecordBuffer.java:6951)
        at com.goldencode.p2j.persist.RecordBuffer.average(RecordBuffer.java:5000)
        at com.goldencode.p2j.persist.BufferImpl.average(BufferImpl.java:569)
        at com.goldencode.p2j.persist.BufferImpl.average(BufferImpl.java:548)
```

4GL example I used is

```
select avg(book.book-id) into i from book where book.book-id > 3 and book.book-title <> ?.
```

After some googling I have found similar stack trace in

```
http://stackoverflow.com/questions/9496571/agregate-functions-max-and-avg-are-not-working-with-datediff-in-hib
ernate-hql
```

(with datediff, but I think it is no so important)

Maybe it is driver bug?

#18 - 06/25/2013 12:20 PM - Eric Faulhaber

Have you turned on logging in Hibernate to see what SQL is being generated from this? Have you debugged into Hibernate to see what exactly the JDBC type code failure is?

Ideally, we would fix this in Hibernate or in P2J. However, if this is not feasible, we probably can work around it by using select sum(...), count(...), ... and calculating the average in the runtime, though this is a bit of a hack.

#19 - 06/25/2013 06:08 PM - Vadim Nebogatov

Eric Faulhaber wrote:

> Have you turned on logging in Hibernate to see what SQL is being generated from this?

Not yet, will do.

> Have you debugged into Hibernate to see what exactly the JDBC type code failure is?

Do you mean debug using Hibernate sources? It fails in method

```
protected final int determineJdbcTypeCode(Type firstArgumentType, SessionFactoryImplementor factory) throws Qu
eryException {
try {
final int[] jdbcTypeCodes = firstArgumentType.sqlTypes( factory );
if ( jdbcTypeCodes.length != 1 ) {
throw new QueryException( "multiple-column type in avg()" );
}
return jdbcTypeCodes[0];
}
catch ( MappingException me ) {
throw new QueryException( me );
}
}
```

firstArgumentType equals to null. It seems problem with internal Ast for HQL statement

> Ideally, we would fix this in Hibernate or in P2J. However, if this is not feasible, we probably can work around it by using select sum(...), count(...), ... and calculating the average in the runtime, though this is a bit of a hack.

Agreed. Will analyse more deeply

#20 - 06/26/2013 05:31 AM - Vadim Nebogatov

For change hibernate logging I need to create corresponding cfg/log4j.properties file. As I see in note 111 for issue 1580, it is already discussed but it seems not committed. Should I use some path or create cfg/log4j.properties myself?

#21 - 06/26/2013 10:46 AM - Eric Faulhaber

Vadim Nebogatov wrote:

> For change hibernate logging I need to create corresponding cfg/log4j.properties file. As I see in note 111 for issue 1580, it is already discussed but it seems not committed. Should I use some path or create cfg/log4j.properties myself?

We don't want to use a log4j.properties file at runtime. We want to use the logging settings in the directory (server/directory.xml). Search for the "loggers" section. Keep reading #1580 notes 111-115, and consult with Stanislav if you have problems.

#22 - 06/27/2013 05:48 PM - Eric Faulhaber

Vadim wrote (in email):

> select avg(...) is not logged because exception happens on statement creation time

You will need to debug into Hibernate to find out what is going wrong here:

```
final int[] jdbcTypeCodes = firstArgumentType.sqlTypes( factory );
```

#23 - 06/27/2013 05:50 PM - Vadim Nebogatov

Yes, I am just doing this now

#24 - 07/05/2013 04:32 PM - Vadim Nebogatov

Some intermediate ideas about avg() runtime bugs related with Hibernate.

The reason is not clear so far. I debugged with 5.1.8 sources from hibernate site, they are mismatched, but the reason is for sure related with creating/parsing internal Ast based on SQL statement. First node of average function is expected but missed (NPE is as a consequence).

I did not complete debugging with updated sources so far.

#25 - 07/08/2013 06:11 AM - Vadim Nebogatov

I have found the reason of NPE in avg runtime support.
Problem is in patching org.hibernate.hql.internal.ast.SqlGenerator.java.

Original hibernate-core-4.1.8.Final-sources.jar and hibernate-core-4.1.9.Final-sources.jar use the same code

```
    protected void endFunctionTemplate(AST node) {
        FunctionNode functionNode = ( FunctionNode ) node;
        SQLFunction sqlFunction = functionNode.getSQLFunction();
        if ( sqlFunction == null ) {
            super.endFunctionTemplate( node );
        }
        else {
            final Type functionType = functionNode.getFirstArgumentType();
            // this function has a registered SQLFunction -> redirect output and catch the arguments
            FunctionArguments functionArguments = ( FunctionArguments ) writer;
            writer = outputStack.removeFirst();
            out( sqlFunction.render( functionType, functionArguments.getArgs(), sessionFactory ) );
        }
    }
```

but patched hibernate-4.1.8.Final-with-gcd-mods.zip uses changed code

```
    protected void endFunctionTemplate(AST node) {
        FunctionNode functionNode = ( FunctionNode ) node;
        SQLFunction sqlFunction = functionNode.getSQLFunction();
        if ( sqlFunction == null ) {
            super.endFunctionTemplate( node );
        }
        // save the current writer
        FunctionWriter func = (FunctionWriter) writer;
        // restore the original writer
        writer = (SqlWriter) outputStack.pop();
        // generate the correct output to the original writer
        out( func.render() );
    }
```

and does not pass functionType to render(...).

I don't know the reason why these changes were made, but I debugged avg runtime with hibernate-core-4.1.8.Final-sources.jar and functionType was calculated and passed properly (com.goldencode.p2j.persist.type.DatetimeUserType in my example).

#26 - 07/08/2013 06:25 AM - Stanislav Lomany

I just ported the GCD changes for Hibernate 3.0.5 to Hibernate 4.1.8.
3.0.5 didn't have functionType parameter so I missed to add it while porting.

#27 - 07/08/2013 06:29 AM - Stanislav Lomany

Vadim, can you provide the failing code so I can try to fix Hibernate?

#28 - 07/08/2013 06:59 AM - Vadim Nebogatov

- File vmn_upd20130707a.zip added

Stas,

I have attached update vmn_upd20130707a.zip merged to latest revision 10362.

I used example

```
def var d as decimal.
select avg(price) into d from book where book-id > 3.
```

#29 - 07/08/2013 02:16 PM - Vadim Nebogatov

Testing DELETE FROM in multi-user environment.

The behavior does not depend on SHARE-LOCK | EXCLUSIVE-LOCK | NO-LOCK settings: error message is shown if record is locked.

_sqlbufN in use by on pts/1. Wait or press CTRL-C to stop. (121)

(N is some increasing number, possible buffer id)

If select Wait, all records are deleted after records are unlocked, if CTRL-C is selected – no records removed at all. I was not able to reproduce partial records removing.

Test scenario: start

```
FOR EACH book SHARE-LOCK:
DISPLAY book.price book.book-title book.isbn.
END.

message "before".

delete from book.

create book.
book.isbn = "11111".
book.book-id = 11111.

create book.
book.isbn = "2222".
book.book-id = 22222.

create book.
book.isbn = "3333".
book.book-id = 33333.

FOR EACH book SHARE-LOCK:
message "after".
message "after".
DISPLAY book.price book.book-title book.isbn.
END.
```

stop on one of <message "after"> and execute

```
delete from book
```

for another user

#30 - 07/09/2013 06:09 AM - Stanislav Lomany

- File hibernate-core-4.1.8.jar added

Fixes the problem with avg.

#31 - 07/09/2013 06:13 AM - Stanislav Lomany

- File SqlGenerator.java added

Update SqlGenerator.

#32 - 07/09/2013 09:45 AM - Vadim Nebogatov

avg runtime works properly now

#33 - 07/09/2013 04:44 PM - Vadim Nebogatov

Additionally to note 29.
I have tested DELETE FOR with WHERE clause. I understand it is not require to support, but behaviour is the same. As result I have noticed that even such statement also waits for whole table even if only one record is locked independently whether this record satisfies WHERE clause or not . So, I think idea of table-level locking is correct.
The question is how to continue DELETE FOR support without implemented #2126?

#34 - 07/11/2013 03:09 PM - Eric Faulhaber

Vadim Nebogatov wrote:

> Additionally to note 29.
> I have tested DELETE FOR with WHERE clause. I understand it is not require to support, but behaviour is the same. As result I have noticed that even such statement also waits for whole table even if only one record is locked independently whether this record satisfies WHERE clause or not . So, I think idea of table-level locking is correct.

Good information. Please note this in the JavaDoc for the appropriate methods so it is not forgotten.

> The question is how to continue DELETE FOR support without implemented #2126?

For now, please implement all you can without this feature, and put TODOs in the code at the points where the following actions will need to be taken:

- current table lock level is queried/stored;
- exclusive table lock is acquired;
- original table lock level is restored.

#35 - 07/11/2013 04:52 PM - Vadim Nebogatov

Are there sense of reversible bulk delete for persistent tables?

#36 - 07/11/2013 04:58 PM - Eric Faulhaber

Good question. I'm sure a DELETE FROM style bulk delete would be rolled back at a full transaction scope, but we need to know if it can be undone at a subtransaction scope. If so, we will have a problem with this approach, since the plan was not to track the actual records deleted at each subtransaction scope. Please come up with a test case for this.

#37 - 07/12/2013 01:49 PM - Vadim Nebogatov

Example undo for DELETE FROM for subtransaction. It works: record exists after bulk delete and undo

```
create book.
book.book-id = 100.
book.book-title = "init1".
book.isbn = "qwerty".

run tran.p.
FIND LAST book EXCLUSIVE-LOCK.
message "record found" book-title.

/**** tran.p *****

delete from book.
undo.
```

#38 - 07/12/2013 06:25 PM - Eric Faulhaber

OK, this complicates the implementation.

Based on your findings so far, I think the way we need to handle this is to determine whether we are in a full transaction scope or a sub-transaction scope, and differ the delete mechanism accordingly. If in a full transaction, we do the bulk delete, but in a sub-transaction, we do the equivalent of a converted:

```
for each my-buf exclusive-lock:
  delete my-buf.
end.
```

...including all of the enclosing BlockManager block scaffolding. Although this will actually create a sub-transaction nested one level deeper than the current one, it should ensure that all of the proper undo processing is done. All ReversibleDelete objects should roll up to the enclosing sub-transaction block when this sub-transaction is committed, such that they will be reversed if the enclosing block is undone.

A future, alternative is to see if we can leverage the Savepoint capability in ~~Hibernate~~ JDBC, but that is more complicated than is justified currently.

A question I still have with the bulk delete approach: how do we deal with any buffers which currently hold records from the table being deleted, both in the current user context and loaded with no-lock in other user contexts? We probably need to force a merge with the Hibernate session and make sure these are empty. I suspect we need to update the DirtyShareManager in this case, too. This has to be reversible as well. This needs some testing...

#39 - 07/15/2013 05:24 PM - Vadim Nebogatov

Reversible operations for temporary tables are based on multiplex ID value and records are not actually deleted from table until transaction is not commited.

For persistent tables we have no such unique key, so it seems only way for rollback bulk delete support (without where clause) is cloning table. Or faster, creating empty table clone (with copy of all indexes) and rename both tables on commit/rollback like switch partition works for partitioned tables.

#40 - 07/15/2013 06:33 PM - Eric Faulhaber

Interesting idea.

Please detail your proposed design for bulk delete "undo"-ability (to use the Progress term) for both temp-tables and persistent tables. For large, persistent tables which must be completely emptied, I like the empty clone table swapping idea. The other approach of cloning all the records into a secondary table (like my subtransaction bulk delete loop idea above) is impractical for very large tables. It may be that we have to go with one of these approaches for partial bulk deletes, but that is not required right now.

Another consideration: how do we ensure that the Hibernate second level cache is appropriately cleared for the table being swapped?

#41 - 07/16/2013 10:24 AM - Eric Faulhaber

The other option is to look at savepoints sooner rather than later, since this is probably the most correct/supported way to do this kind of thing. Thoughts on this option?

#42 - 07/17/2013 06:11 PM - Vadim Nebogatov

- File vmn_upd20130717b.zip added

- File vmn_upd20130717a.zip added

Savepoint seems more promising and common approach. I have attached very draft deleteAll support based on Savepoint (merged with latest revision 10364).

I tested it with attached examples with commented and uncommented undo in external procedure tran.p – it works exactly like in 4GL (error if undo is commented).

But as far as I understood Savepoint could give some problems with Hibernate cache and of course should be tested separately on H2 and PostgreSQL.

BTW, on Hibernate logs I see creation of temporary empty table with book structure with the same indexes. I could mistaken but possible it is elementary undo operation prepared for delete all based on th same idea like I wrote.

#43 - 07/18/2013 06:17 PM - Vadim Nebogatov

I was mistaken, temporary table was created by DefaultDirtyShareManager for track changes

#44 - 07/19/2013 03:01 PM - Vadim Nebogatov

- File vmn_upd20130719a.zip added

Savepoint based reversible actually could work for other operations, not only for deleteAll. I have attached update vmn_upd20130719a.zip with common SavepointReversibleImpl. Of course it is first draft. Tested with the same tests vmn_upd20130717b.zip.

It is interesting to execute regressions for all these changes including for sum/count/avg. What do you think, are there any sense to make this now?

#45 - 07/21/2013 06:02 PM - Eric Faulhaber

Code review 20130719a:

- This version of database_general.rules is the same as the one currently checked into bzr. Was the correct file included in this update?
- Please update the javadoc in RecordBuffer.deleteAll. This method is no longer an unsupported operation and no longer throws UnsupportedOperationException.
- Please put the Hibernate jar in a separate update and put it under the lib directory instead of build/lib. The latter directory is for external jar files copied from lib in the prepare target of the Ant build.xml script, and for jars built at compile time.
- Please move RecordBuffer$SavepointReversible to a separate, top-level class (pass the fields it needs from RecordBuffer into its c'tor. I don't think it needs to invoke any methods of the enclosing class, so this should be fairly straightforward. Add the missing javadoc.

I talked over the savepoint approach with Greg. Since we don't currently have simple answers for the Hibernate second-level cache or for dealing with any records currently held in buffers, we'd like to first use a simple, "converted-style" FOR-EACH loop as the basis for the RecordBuffer.deleteAll implementation. This approach leverages the code we already have to deal with these issues. If we find this causes memory problems, due to the bulk delete of very large tables, we can revisit the savepoint implementation and figure out these remaining issues. Don't delete the savepoint-related code in deleteAll, just comment it out -- we may need this in the near future.

#46 - 07/22/2013 09:34 AM - Vadim Nebogatov

Eric Faulhaber wrote:

- This version of database_general.rules is the same as the one currently checked into bzr. Was the correct file included in this update?

This is file you sent me separately. Bzr version does not contain 023 ECF 20130618

#47 - 07/22/2013 10:05 AM - Vadim Nebogatov

Eric Faulhaber wrote:

- Please put the Hibernate jar in a separate update and put it under the lib directory instead of build/lib. The latter directory is for external jar files copied from lib in the prepare target of the Ant build.xml script, and for jars built at compile time.

Good. But hibernate-core-4.1.8.jar is file I received from Stas, I am not sure if he will commit (or already committed it) himself?

#48 - 07/22/2013 10:18 AM - Stanislav Lomany

> Good. But hibernate-core-4.1.8.jar is file I received from Stas, I am not sure if he will commit (or already committed it) himself?

Please commit it with your update.

#49 - 07/23/2013 02:48 PM - Eric Faulhaber

Vadim Nebogatov wrote:

> This is file you sent me separately. Bzr version does not contain 023 ECF 20130618

Yes, of course. My mistake.

#50 - 07/23/2013 03:05 PM - Vadim Nebogatov

- File vmn_upd20130723a.zip added

- File vmn_upd20130723b.zip added

I have attached update vmn_upd20130723a.zip/vmn_upd20130723b.zip with deleteAll based on "converted-style" FOR-EACH loop, and top-level SavepointReversible class.
Merged with last revision 10366.

#51 - 07/23/2013 03:25 PM - Eric Faulhaber

Code review 20130723a/b:

Code looks good. Please regression test and check it in if successful.

#52 - 07/24/2013 02:48 PM - Vadim Nebogatov

Regression 20130724_051719 completed. There are 8 FAILED and FAILED_DEPENDENCY tests in ts_tests:

1. tc_job_002 FAILED failure in step 40: 'Unexpected EOF in actual at page # 1.'
2. tc_job_clock_001 FAILED failure in step 9: 'timeout before the specific screen buffer became available (Mismatched data at line 5, column 18. Expected '' (0x0000 at relative Y 5, relative X 18) and found '┌' (0x250C at relative Y 5, relative X 18).)'
3. tc_job_clock_002 FAILED failure in step 20: 'timeout before the specific screen buffer became available (Mismatched data at line 5, column 18. Expected '' (0x0000 at relative Y 5, relative X 18) and found '┌' (0x250C at relative Y 5, relative X 18).)'
4. tc_job_clock_003 FAILED_DEPENDENCY dependency chain: tc_job_clock_001
5. tc_job_clock_004 FAILED failure in step 10: 'timeout before the specific screen buffer became available (Mismatched data at line 5, column 18. Expected '' (0x0000 at relative Y 5, relative X 18) and found '┌' (0x250C at relative Y 5, relative X 18).)'
6. tc_job_clock_005 FAILED failure in step 23: 'timeout before the specific screen buffer became available (Mismatched data at line 5, column 18. Expected '' (0x0000 at relative Y 5, relative X 18) and found '┌' (0x250C at relative Y 5, relative X 18).)'
7. tc_job_clock_006 FAILED_DEPENDENCY dependency chain: tc_job_clock_001 --> tc_job_clock_003
8. tc_dc_slot_023 FAILED_DEPENDENCY dependency chain: tc_job_clock_001 --> tc_job_clock_003 --> tc_job_clock_006

Or should I repeat regression one more time?

#53 - 07/24/2013 03:01 PM - Eric Faulhaber

IIRC, some of the "job clock" tests are sensitive to the time of day they are run, so these may be false negatives (other than tc_job_002, which is known to fail, but it is OK). Please try running the harness again, between 14:00 and 22:00 (devsrv01 time).

#54 - 07/26/2013 05:38 AM - Vadim Nebogatov

Regression 20130725_172005 completed in time you pointed. There are less errors: 3 FAILED tests in ts_tests.

1. tc_job_002 FAILED failure in step 40: 'Unexpected EOF in actual at page # 1.'
2. tc_job_matlcron_002 FAILED timeout before the specific screen buffer became available (Mismatched data at line 4, column 8. Expected ' ┌' (0x250C at relative Y 4, relative X 8) and found '' (0x0000 at relative Y 4, relative X 8).)
3. tc_job_clock_005 FAILED failure in step 8: 'timeout before the specific screen buffer became available (Mismatched data at line 4, column 9. Expected '' (0x0000 at relative Y 4, relative X 9) and found '1' (0x0031 at relative Y 4, relative X 9).)'

#55 - 07/29/2013 11:12 AM - Vadim Nebogatov

Eric, may I commit this update?

#56 - 07/29/2013 11:59 AM - Eric Faulhaber

Yes, please commit and distribute.

#57 - 08/12/2013 02:12 PM - Vadim Nebogatov

Is field name to java name mapping available in runtime now? I need it to retrieve correct sort clause for deleteAll operation implementation. I don't know the order of delete for operation, so I intend to take first, not first primary, index.

#58 - 08/12/2013 02:43 PM - Eric Faulhaber

Still working on field name mapping.

Why do you plan to take the first index instead of the primary index? Please review annotations/index_selection.rules to see which index is chosen during conversion in the case of a FOR EACH loop with no where clause. IIRC, it is the primary index, but you should double-check this.

#59 - 08/12/2013 04:46 PM - Vadim Nebogatov

I checked

```
        <!-- if there was no order by clause, we are using a temp/work-table
             with no index;  use primary key ascending as a default →
```

So, it seems I have to take primary index. From previous issues, I was not sure that primary flag is set correctly for permanent table indexes. Maybe I was mistaken.

#60 - 08/12/2013 10:09 PM - Eric Faulhaber

This is not exactly what I was looking for; a missing WHERE clause does not mean you will have a missing order by clause, and "primary key ascending" is not the same as the primary index, in Progress terms.

However, it does appear that in most cases of a missing WHERE clause, you would use the primary index. If you look at IndexSelectionWorker$IndexHelper.calcSelectedIndex, it defaults to the primary index (if there is one), if none of the other, higher precedence selections are possible. This would be the case in the absence of a WHERE clause. In the case a table has no index defined at all (only possible for temp-tables, which is not what this issue is about), you would use the recid ascending.

See the full description of rules we have determined for index selection at dist/docs/api/com/goldencode/p2j/schema/package-summary.html#Determining_Query_Sorting_.

#61 - 08/13/2013 02:35 PM - Vadim Nebogatov

Is it impossible to use IndexSelectionWorker (and P2OLookup) in runtime?

#62 - 08/13/2013 03:21 PM - Eric Faulhaber

For the moment, these resources are available at conversion time only, but we are adding support for index selection at runtime to support dynamic queries. I am making a new class, MetaSchema which will offer runtime services similar to those provided by P2OLookup at conversion. This also will support many of the remaining 4GL methods and attributes.

Please provide the specific types of services you require (both from IndexSelectionWorker and P2OLookup) for this issue; this will help shape the API we are creating.

#63 - 08/13/2013 03:42 PM - Vadim Nebogatov

I think for delete all only needed:

1) method repeating the logic of calcIndexSelection() including all private methods used in calcIndexSelection(). Maybe even move calcIndexSelection() from IndexSelectionWorker, make static and pass metadata as parameter. Yes, it does not work with IndexSelectionWorker in runtime just because of absence of metadata.
2) method converting field name to java name

#64 - 08/13/2013 04:00 PM - Vadim Nebogatov

In principle, calcIndexSelection() does not work in runtime only because of problem with method indexes(), requiring metadata. But indexes could be retrieved in runtime with

```
      Iterator<P2JIndex> indexes = DMOIndex.databaseIndexes(getSchema(),
                                                            getDatabase(),
                                                            getDMOInterface());
```

#65 - 08/13/2013 04:21 PM - Eric Faulhaber

IndexSelectionWorker works within the context of the pattern engine and relies upon the set up work done by index_selection.rules to populate its data structures before doing the calculation. So, there is a bit more work to be done to get this going at runtime. Stas has done a lot of the foundation work in #1652.

However, since you already know you want the primary index only for the no-WHERE clause variant of delete from, you don't need the full index selection analysis. You could get away with using DMOIndex as you suggest, then iterating through the P2JIndex objects that come back, and testing each with isPrimary until you find the one you're looking for. Do you have enough information available at that point to make the DMOIndex.databaseIndexes call?

#66 - 08/13/2013 04:23 PM - Vadim Nebogatov

Yes. What about field name to java name map? It seems it should be simple, or I do not see some problems?

#67 - 08/13/2013 04:31 PM - Vadim Nebogatov

I even think such or similar method exists in Hibernate

#68 - 08/13/2013 04:43 PM - Eric Faulhaber

Do you mean a facility where you specify a Progress legacy table and field name, and it provides the Java DMO's property name for the associated column? This is not available yet, but will be part of the MetaSchema API. I assume this is for the purpose of assembling the HQL order by clause?

#69 - 08/13/2013 04:44 PM - Vadim Nebogatov

Yes.

#70 - 08/13/2013 04:46 PM - Eric Faulhaber

Hibernate does not know anything about the Progress legacy names. It only maps the Java object (DMO/property) names to the SQL (table/column) names.

#71 - 08/13/2013 04:49 PM - Vadim Nebogatov

I understood. I will implement and test with simple fields having names equal to java names and insert TODO in corresponding place. Is it OK?

#72 - 08/13/2013 04:50 PM - Eric Faulhaber

Yes, good approach.

#73 - 08/14/2013 06:02 AM - Vadim Nebogatov

There is only problem with primary recognition. I tested and just remembered why I did not trust primary flag in runtime:

method DMOIndex.databaseIndexes() uses Persistence.retrieveIndexData() and retrieves only JDBC properties from ResultSet (from JDBC metadata) and sets always false for all indexes

```
index = new P2JIndex(table, indexName, !nonUniq, false, false);
```

#74 - 11/25/2013 02:01 PM - Eric Faulhaber

Please use the new primary index annotation you have added for #2126 to ensure that the no-WHERE-clause variant of delete from works properly.

#75 - 11/26/2013 03:36 PM - Vadim Nebogatov

Actually I tested deleteAll() with DELETE FOR for permanent tables. Rest issue, – fix in P2JIndex.getOrderByClause(), used dmoAlias instead of table name, - made in #2126, note 44.

#76 - 12/05/2013 03:35 PM - Eric Faulhaber

See my review of 20131129a/b in #2126.

#77 - 01/07/2014 12:45 PM - Eric Faulhaber

- Status changed from New to Closed

#78 - 11/16/2016 11:42 AM - Greg Shah

- Target version changed from Milestone 7 to Runtime Support for Server Features

Files

ecf_upd20130224a.zip	457 KB	02/25/2013	Eric Faulhaber
ecf_upd20130224b.zip	2.31 KB	02/25/2013	Eric Faulhaber
ecf_upd20130227a.zip	19.2 KB	02/28/2013	Eric Faulhaber
ecf_upd20130618a.zip	7.05 KB	06/18/2013	Eric Faulhaber
vmn_upd20130707a.zip	91.6 KB	07/08/2013	Vadim Nebogatov
hibernate-core-4.1.8.jar	4.29 MB	07/09/2013	Stanislav Lomany
SqlGenerator.java	11.9 KB	07/09/2013	Stanislav Lomany
vmn_upd20130717a.zip	3.98 MB	07/17/2013	Vadim Nebogatov
vmn_upd20130717b.zip	647 Bytes	07/17/2013	Vadim Nebogatov
vmn_upd20130719a.zip	3.98 MB	07/19/2013	Vadim Nebogatov
vmn_upd20130723a.zip	3.98 MB	07/23/2013	Vadim Nebogatov
vmn_upd20130723b.zip	679 Bytes	07/23/2013	Vadim Nebogatov