Database - Feature #1661

support case-sensitive fields in indexes

10/22/2012 12:21 AM - Eric Faulhaber

Status:	Closed	**Start date:**	07/19/2013
Priority:	Normal	**Due date:**	07/21/2013
Assignee:	Ovidiu Maxiniuc	**% Done:**	100%
Category:		**Estimated time:**	8.00 hours
Target version:	Cleanup and Stablization for Server Features		
billable:	No	**vendor_id:**	GCD

Description

Related issues:

Related to Database - Feature #1663: test/debug case-sensitive field support	**Closed**	**05/01/2013**	**05/02/2013**
Related to Database - Feature #1664: data import improvements	**New**		
Related to Database - Bug #2259: computed character columns are not created f...	**Closed**	**03/14/2014**	
Blocked by Database - Feature #2118: add runtime support for _index and _inde...	**Closed**	**07/11/2013**	**07/19/2013**

History

#1 - 10/22/2012 12:26 AM - Eric Faulhaber

- Estimated time changed from 8.00 to 24.00

This will require changes to:

- the DDL index generation code (don't emit UPPER function in CREATE INDEX statement);
- either the conversion or HQL preprocessing (probably both) of HQL where clauses and substitution parameters;
- either the conversion or runtime preprocessing (probably both) of HQL sort phrases (ORDER BY clauses);
- possibly to the index runtime classes themselves (not sure case-sensitivity is properly handled by these classes today).

#2 - 10/31/2012 04:02 PM - Greg Shah

- Target version set to Milestone 7

#3 - 12/21/2012 12:20 PM - Eric Faulhaber

- Assignee set to Ovidiu Maxiniuc

#4 - 05/02/2013 01:53 AM - Eric Faulhaber

- Estimated time changed from 24.00 to 8.00

- Start date set to 07/19/2013

- Due date set to 07/21/2013

- Assignee changed from Ovidiu Maxiniuc to Eric Faulhaber

I think everything here is working, except that P2JH2Dialect.isCaseInsensitiveColumn is implemented incorrectly. This should be addressed as part of implementing index metadata (#2118).

#5 - 01/07/2014 02:18 PM - Eric Faulhaber

- Status changed from New to WIP

- % Done changed from 0 to 70

- Target version changed from Milestone 7 to Milestone 11

The remaining portion of this task is not needed for M7; moving to M11.

#6 - 03/14/2014 12:26 AM - Eric Faulhaber

- Assignee changed from Eric Faulhaber to Ovidiu Maxiniuc

Ovidiu, please determine what is left to do with P2JH2Dialect.isCaseInsensitiveColumn. I think whatever index information I was waiting for from the metadata work should be available now via TableMapper instead, or directly via the DMO annotations.

#7 - 03/18/2014 04:47 PM - Ovidiu Maxiniuc

At this moment, we cannot tell if a field is case sensitive or not only by analyzing its name (as signature of isCaseInsensitiveColumn(String name) suggests). At conversion time, for H2 dialect, all character fields get a computed column mirror with either:

- __char_f_sens as rtrim(char_f_sens, ' \t\n\r')
- __char_f_inse as upper(rtrim(char_f_inse, ' \t\n\r'))
  Because the lack of other information, the CC signature is useless in this case.

In the case of permanent tables, the DDL is implemented by adding an alter table <table> alter column statement after definition of the table. In the case of static temp-tables, the new columns are injected by ORMHandler.

Other things I found that I think should be fixed:

- Looks like the dynamic temp-tables are missing the computed columns at all (especially the character columns that are case insensitive), attempting to define indexes on those fields will lead to Column "__FIELDn" not found errors and client is disconnected.
- Also, to allow dynamic temp-tables to be built using add-like-field method, the case-sensitive attribute must be saved within the P2JField.
- A annotation for character fields could be in handy here, although this could be mandatory as the dmo_index.xml keeps a case-sensitive tag with the list of fields with this attribute.
- For static temp-tables, the computed column are added only if there is at least an index that use them, no matter if case-sensitive or not. Yet, I am not sure if this is correct or not.

#8 - 03/20/2014 01:53 PM - Eric Faulhaber

Ovidiu Maxiniuc wrote:

> At this moment, we cannot tell if a field is case sensitive or not only by analyzing its name (as signature of isCaseInsensitiveColumn(String name) suggests). At conversion time, for H2 dialect, all character fields get a computed column mirror with either:
>
> - __char_f_sens as rtrim(char_f_sens, ' \t\n\r')
> - __char_f_inse as upper(rtrim(char_f_inse, ' \t\n\r'))
>   Because the lack of other information, the CC signature is useless in this case.

What do you propose to correct this? You can modify the dialect API as needed.

> In the case of permanent tables, the DDL is implemented by adding an alter table <table> alter column statement after definition of the table. In the case of static temp-tables, the new columns are injected by ORMHandler.

Other things I found that I think should be fixed:

- Looks like the dynamic temp-tables are missing the computed columns at all (especially the character columns that are case insensitive), attempting to define indexes on those fields will lead to Column "__FIELDn" not found errors and client is disconnected.

Stas, didn't you also determine this recently? I don't recall where I saw it, but I thought you had documented this. If so, please coordinate with each other so you are not duplicating effort on a fix.

- Also, to allow dynamic temp-tables to be built using add-like-field method, the case-sensitive attribute must be saved within the P2JField.

OK.

- A annotation for character fields could be in handy here, although this could be mandatory as the dmo_index.xml keeps a case-sensitive tag with the list of fields with this attribute.

I would like to reduce the information we store in dmo_index.xml and move more information into DMO annotations. This file originally was implemented as a list of DMOs to bootstrap at server startup, and to store legacy information we needed at runtime that didn't go into the database schema itself. Annotations are a better mechanism for the latter, as it reduces the artifacts to maintain for developers of new DMOs.

So, if you can cleanly remove the case-sensitive information from dmo-index.xml and store this in an annotation instead, please do so. If it seems like a lot of effort to retrofit this, please let me know the issues and we'll discuss further.

- For static temp-tables, the computed column are added only if there is at least an index that use them, no matter if case-sensitive or not. Yet, I am not sure if this is correct or not.

This is intentional and in fact goes to the heart of the reason we have computed columns in P2J. They only were introduced to be used in indexes, because in some databases (like H2 and SQL Server), expressions are not allowed in indexes.

#9 - 03/20/2014 02:07 PM - Stanislav Lomany

- Looks like the dynamic temp-tables are missing the computed columns at all (especially the character columns that are case insensitive), attempting to define indexes on those fields will lead to Column "__FIELDn" not found errors and client is disconnected.

> Stas, didn't you also determine this recently? I don't recall where I saw it, but I thought you had documented this. If so, please coordinate with each other so you are not duplicating effort on a fix.

Yes, there is the issue #2259.

#10 - 04/08/2014 03:58 PM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> What do you propose to correct this? You can modify the dialect API as needed.

To fix this issue, I am attempting to use two signatures for computed columns: one for case sensitive and one for case insensitive computed columns. The current CC_PREFIX will be replaced by other two constants with values __s and __i respectively. The only change in P2JDialect would be to add a boolean parameter to getComputedColumnPrefix() in order to return the appropriate prefix. The double underscore remains as marker for computed columns but the first 3 characters are analyzed when deciding extracting about some unknown column.
I believe that at this moment only the character fields need such handling in indexes as other Progress datatypes do not have such attributes (like case-sensitivity).

I did not started working on #2259 part, but I will contact Stanislav before I do.

LE: Just remember that as far as I can tell, the SQL Server dialect seems to suffer from the same defect, the two fields mentioned in note 7 being generated as:

```
alter table book2 add __char_f_sens as substring(char_f_sens, 0, len(replace(replace(replace(char_f_sens, CHAR
(9), ' '), CHAR(10), ' '), CHAR(13), ' ')));
alter table book2 add __char_f_inse as upper(substring(char_f_inse, 0, len(replace(replace(replace(char_f_inse
, CHAR(9), ' '), CHAR(10), ' '), CHAR(13), ' '))));
```

on SQL 2012.

#11 - 04/09/2014 04:42 PM - Ovidiu Maxiniuc

There is as chance that the proposed solution might not be implemented because at some points the indexes are dropped and recreated using only the information if the respective field is a character computed column. I will investigate if the extra case-sensitivity information can be added.

#12 - 04/11/2014 05:09 PM - Ovidiu Maxiniuc

- File om_upd20140411b.zip added

Intermediary update, not fully tested.
Changes:
Added case-sensitive attribute in P2JField.
Added caseSensitive field annotation so the attribute is available at runtime.

At this moment, it's not very clear to me how the of the changes in the name of character columns affects the import / installation of an existing database.

The update was not fully tested with dynamic temp-tables. That is probably some configurations must be adjusted as a result of merging the latest changes from bzr (most likely updates from #2235).

#13 - 04/17/2014 05:22 PM - Ovidiu Maxiniuc

- File om_upd20140418a.zip added

The update almost passed the full-regression testing, 3 tests failed:

- ctrlc_11_session1/3/4: because some semaphore timing issue (one client process was killed)
- gso_190: Expected '' (0x0000 at relative Y 21, relative X 0) and found 'R' in test 27 (I believe I saw this as a false but I may be wrong).
- tc_job_002: Unexpected EOF in actual at page # 1. (as expected)

I cleanup the sources as I discovered some extra traces in logs, I merged the today changes of Vadim from bzr and restarted the test.

#14 - 04/18/2014 01:26 PM - Ovidiu Maxiniuc

- File om_upd20140418b.zip added

I believe the current update passed the test. The only issues were:

- ctrlc_01_session5: Expected '' (0x0000 at relative Y 19, relative X 32) and found 'K' - client Killed
- ctrlc_01_session6: Timeout while waiting for event semaphore
- tc_job_002: Unexpected EOF in actual at page # 1. (expected)
- tc_job_clock_002: timeout before the specific screen buffer became available (known false positive)

Waiting for the review to continue with commit.

#15 - 04/18/2014 05:52 PM - Eric Faulhaber

Code review 20140416b:

Nice functional update (and a lot of code cleanup as well). Some issues/questions:

- Why was the change to schema/fixups.xml needed? Does the kw_case_sen token appear anywhere other than under a field_char? Was this just for code clarity?
- There is no header entry in hibernate.xml. What is the significance of the computed column name prefix changes (e.g., __oOo4%s, __oOo5%s, __oOo6%s)? Was this for debugging during development? Did you mean to leave it behind? Other than format changes, these seem to be the only changes in the file.
- generate_ddl.xml: only a minor format issue, but you added a 1-space indent in lines 211-225.
- DynamicQueryHelper: why have you added back the WorkArea.permSchemasLoaded flag? Perhaps the removal was missed while merging up to bzr?
- Multiple classes: please don't import individual classes unless you are resolving ambiguity.
- TempTableBuilder: in addNewField, you added the comment "Note: case-sensitivity (last parameter) cannot be specified using ADD-NEW-FIELD method (?)". Are you saying this is a limitation of the 4GL method or a flaw in our API?
- P2JDialect (and all subclasses): the parameter name to getComputedColumnPrefix is caseSensible. Shouldn't this be caseSensitive?
- Why don't the overrides of P2JDialect.isComputedColumn always return false for dialects that don't use computed columns? Or is this more of a test for an indexed, text column?
- DatabaseManager: please move mapComputedCharacterColumns with the other package private static methods, since you changed its access modifier.
- Regarding your comment in TempTableHelper ("OM: aren't temp-tables always H2 db-backed ?"). In practice, yes, but the dialect and other DB settings actually are driven from the directory. We used to use PostgreSQL to back temp-tables, but we changed it a long time ago to use H2 in practice. I don't think it would still work if one configured it otherwise in the directory, and we probably should disable this (if we haven't already - I can't remember).

#16 - 04/20/2014 12:21 PM - Eric Faulhaber

Eric Faulhaber wrote:

> - DynamicQueryHelper: why have you added back the WorkArea.permSchemasLoaded flag? Perhaps the removal was missed while merging up to bzr?

My mistake. I was comparing your update to the wrong baseline -- I have removed this flag in my working copy; it is not checked in yet.

#17 - 04/23/2014 01:00 PM - Ovidiu Maxiniuc

Eric Faulhaber wrote:

> - Why was the change to schema/fixups.xml needed? Does the kw_case_sen token appear anywhere other than under a field_char? Was this just for code clarity?

No, I don't remember how, but I managed to export a .df file that had the CASE-SENSITIVE attribute to other field than the normal character (namely an int). This is strange, indeed, I was not able to reproduce it again. However, I did not notice that until the generated code was using an inexistent two argument constructor (the second argument being true) and the compile failed. Going back I found the strange token and I added the constraint to respective rule just to be on the safe-zone.

- There is no header entry in hibernate.xml. What is the significance of the computed column name prefix changes (e.g., __oOo4%s, __oOo5%s, __oOo6%s)? Was this for debugging during development? Did you mean to leave it behind? Other than format changes, these seem to be the only changes in the file.

I forgot to mention about these issues:

- __oOo4%s and __oOo5%s are part of createList function, that is called at lines 812 (computed = false) and 820 (commented out).
- The __oOo6%s is part of createComputedColumn function which is not called by anywhere. The only two places where the name of the function is mentioned are the lines 836 and 890 and are commented-out.
  In conclusion, they are dead-code and I wonder if it should be dropped.

- generate_ddl.xml: only a minor format issue, but you added a 1-space indent in lines 211-225.

Sorry, the TAB characters at the start of the line were changed into 4 spaces instead of 3.

- Multiple classes: please don't import individual classes unless you are resolving ambiguity.

Fixed.

- TempTableBuilder: in addNewField, you added the comment "Note: case-sensitivity (last parameter) cannot be specified using ADD-NEW-FIELD method (?)". Are you saying this is a limitation of the 4GL method or a flaw in our API?

Yes, this looks like a 4GL limitation. The reference clearly states the syntax:

```
ADD-NEW-FIELD(field-name-exp, datatype-exp [, extent-exp [, format-exp [, initial-exp [, label-exp [, column-l
abel-exp]]]]])
```

I tried to force some extra logical parameter but the code won't compile. So there is no way to create a case-sensitive field dynamically using this method, only to copy an existing case-sensitive one using ADD-LIKE-FIELD.

- P2JDialect (and all subclasses): the parameter name to getComputedColumnPrefix is caseSensible. Shouldn't this be caseSensitive?

Fixed.

- Why don't the overrides of P2JDialect.isComputedColumn always return false for dialects that don't use computed columns? Or is this more of a test for an indexed, text column?

Yes, you're right. I made the change.
This method was added to replace the checking for computed columns when creating temp-tables in TempTableHelper, the only called implementation will be for P2JH2Dialect as this is the only dialect used now for temp-tables.

- DatabaseManager: please move mapComputedCharacterColumns with the other package private static methods, since you changed its access modifier.

Done.

- Regarding your comment in TempTableHelper ("OM: aren't temp-tables always H2 db-backed ?"). In practice, yes, but the dialect and other DB settings actually are driven from the directory. We used to use PostgreSQL to back temp-tables, but we changed it a long time ago to use H2 in practice. I don't think it would still work if one configured it otherwise in the directory, and we probably should disable this (if we haven't already - I can't remember).

I observed that some code is duplicated (TempTableHelper:516-527 and P2JH2Dialect:795-805). Maybe if we extract this a a new method in P2JDialect, it would be more elegant and not directly hard-coding the H2 syntax, letting the future option to use other dialects for temp-tables?

Regarding the code cleanup: I try not to be very aggressive about this, and do only changes localized to the part of the code I work with. I know that it's already difficult to track the other, more important changes.
Thanks for review.

#18 - 04/23/2014 01:21 PM - Eric Faulhaber

Ovidiu Maxiniuc wrote:

> I observed that some code is duplicated (TempTableHelper:516-527 and P2JH2Dialect:795-805). Maybe if we extract this a a new method in P2JDialect, it would be more elegant and not directly hard-coding the H2 syntax, letting the future option to use other dialects for temp-tables?

Makes sense. Please do this if it is not a lot of effort. If you think it will take more than an hour or so, put a TODO with the idea in the code instead; right now, we have no plans to move away from H2 as the backing dialect for temp-tables.

> Regarding the code cleanup: I try not to be very aggressive about this, and do only changes localized to the part of the code I work with. I know that it's already difficult to track the other, more important changes.

I think you're finding a good balance and I appreciate the cleanup. I use the same approach myself.

#19 - 04/25/2014 05:52 AM - Ovidiu Maxiniuc

- File om_upd20140424a.zip added

Testing this update based on bzr revision 10512.
The main passed flawlessly (except for the expected "Unexpected EOF" in tc_job_002).
The ctrl+c had many fails (the devsrv01 was visibly slow yesterday when the test started) so I am running this set again.

#20 - 04/25/2014 11:20 AM - Ovidiu Maxiniuc

In the second run of the ctrl+c the number of failed test went down by 10. It was not enough so I tested a 3rd time. The last time the gso_ctrlc_3way_tests finished without flaws. The gso_ctrlc_tests had only two:

- ctrlc_01_session5: failure in step 130: Expected ' ' (0x0020 at relative Y 0, relative X 0) and found 'K'
- ctrlc_01_session6: failure in step 57: 'Timeout while waiting for event semaphore to be posted.'

Looking back to previous tested, I can see that these tests PASSED in all previous runs. The harness results are saved as /home/om/shared/clients/timco/majic_test_results/10513_d2814f6_20140425_om.zip (the archive includes all 3 tests for CTRL+C).

While checking the results, the update mail from Constantin (ca_upd20140425a.zip) came. I checked the sources and they do not overlap so I think is not necessary to re-run the test.

#21 - 04/25/2014 12:10 PM - Eric Faulhaber

There is nothing in your update that affects network/session/context processing, so I would not worry about the Ctrl+C results too much for this update. This test set is very timing and load sensitive and the results are not very reliable. Since the main test set has passed and you had a pretty clean Ctrl+C run previously, I would check this in and distribute it.

#22 - 04/25/2014 12:44 PM - Ovidiu Maxiniuc

Update committed revision 10515 and distributed by mail.

#23 - 05/02/2014 02:54 PM - Eric Faulhaber

- % Done changed from 70 to 100

- Status changed from WIP to Closed

#24 - 05/07/2014 02:29 PM - Eric Faulhaber

- Status changed from Closed to WIP

I have re-opened this task, as it appears it creates a regression with projects that use Progress metadata tables. The following snippet is from the server.log of such an application:

```
[05/07/2014 13:07:01 EDT] (com.goldencode.p2j.persist.dialect.P2JH2Dialect:WARNING) Invalid column name: [__db
_name].Failed to extract the root name of the column.
[05/07/2014 13:07:01 EDT] (com.goldencode.p2j.persist.dialect.P2JH2Dialect:WARNING) Invalid column name: [__fi
le_name].Failed to extract the root name of the column.
[05/07/2014 13:07:01 EDT] (com.goldencode.p2j.persist.dialect.P2JH2Dialect:WARNING) Invalid column name: [__ow
ner].Failed to extract the root name of the column.
[05/07/2014 13:07:01 EDT] (com.goldencode.p2j.persist.dialect.P2JH2Dialect:WARNING) Invalid column name: [__du
mp_name].Failed to extract the root name of the column.
[05/07/2014 13:07:01 EDT] (com.goldencode.p2j.persist.dialect.P2JH2Dialect:WARNING) Invalid column name: [__ow
ner].Failed to extract the root name of the column.
[05/07/2014 13:07:01 EDT] (com.goldencode.p2j.persist.dialect.P2JH2Dialect:WARNING) Invalid column name: [__fi
le_name].Failed to extract the root name of the column.
com.goldencode.p2j.cfg.ConfigurationException:  Initialization failure
        at com.goldencode.p2j.main.StandardServer.hookInitialize(StandardServer.java:1565)
        at com.goldencode.p2j.main.StandardServer.bootstrap(StandardServer.java:766)
        at com.goldencode.p2j.main.ServerDriver.start(ServerDriver.java:428)
        at com.goldencode.p2j.main.CommonDriver.process(CommonDriver.java:423)
        at com.goldencode.p2j.main.ServerDriver.process(ServerDriver.java:155)
        at com.goldencode.p2j.main.ServerDriver.main(ServerDriver.java:772)
Caused by: java.lang.RuntimeException: com.goldencode.p2j.persist.PersistenceException: Unable to determine da
ta type for column 'meta_index:ndex_name'
        at com.goldencode.p2j.main.StandardServer$7.initialize(StandardServer.java:966)
        at com.goldencode.p2j.main.StandardServer.hookInitialize(StandardServer.java:1561)
```

```
        ... 5 more
Caused by: com.goldencode.p2j.persist.PersistenceException: Unable to determine data type for column 'meta_ind
ex:ndex_name'
        at com.goldencode.p2j.persist.Persistence.getColumnDataType(Persistence.java:1533)
        at com.goldencode.p2j.persist.Persistence.retrieveIndexData(Persistence.java:1343)
        at com.goldencode.p2j.persist.Persistence.queryIndexData(Persistence.java:1261)
        at com.goldencode.p2j.persist.DMOIndex.getIndexedCharacterColumns(DMOIndex.java:624)
        at com.goldencode.p2j.persist.ORMHandler.mapClass(ORMHandler.java:277)
        at com.goldencode.p2j.persist.DatabaseManager.registerDatabase(DatabaseManager.java:2312)
        at com.goldencode.p2j.persist.DatabaseManager.initialize(DatabaseManager.java:1295)
        at com.goldencode.p2j.persist.Persistence.initialize(Persistence.java:1081)
        at com.goldencode.p2j.main.StandardServer$7.initialize(StandardServer.java:962)
        ... 6 more
```

Please investigate.

#25 - 05/07/2014 02:43 PM - Ovidiu Maxiniuc

Where were those logs extracted from ? I double checked now the server logs from my regression test results and I could not find similar errors/warnings.
Is it possible that this is was a run with updated sources from bzr but the application was not clean converted ? The metadata schema might be obsolete.

#26 - 05/07/2014 02:47 PM - Eric Faulhaber

Ovidiu Maxiniuc wrote:

> Where were those logs extracted from ?

My local customer server project environment.

> I double checked now the server logs from my regression test results and I could not find similar errors/warnings.

The regression environment does not use metadata tables, so I wouldn't expect any such entries.

> Is it possible that this is was a run with updated sources from bzr but the application was not clean converted ? The metadata schema might be obsolete.

Yes, that's the case. I'm reconverting now, will close this task again if it resolves the issue. Thanks!

#27 - 05/13/2014 04:49 PM - Eric Faulhaber

- Status changed from WIP to Closed

#28 - 11/16/2016 12:07 PM - Greg Shah

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

Files

om_upd20140411b.zip	252 KB	04/11/2014	Ovidiu Maxiniuc
om_upd20140418a.zip	470 KB	04/17/2014	Ovidiu Maxiniuc
om_upd20140418b.zip	399 KB	04/18/2014	Ovidiu Maxiniuc
om_upd20140424a.zip	410 KB	04/25/2014	Ovidiu Maxiniuc