Database - Feature #6574

auto-detect native UDFs at server startup and install them if missing

07/04/2022 06:28 PM - Eric Faulhaber

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Eric Faulhaber	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**vendor_id:**	

Description

History

#2 - 07/04/2022 06:32 PM - Eric Faulhaber

We have had many users stumble with the installation of the native UDFs. In the event the directory does not specify to use Java UDFs, we should, upon startup of the FWD server, automatically detect whether the native UDFs are installed in the persistent databases for which the current FWD server is authoritative (and of course, for which native UDFs are supported). If they are not already installed, the native UDFs should be installed/configured.

#3 - 07/11/2022 11:18 AM - Eric Faulhaber

- Assignee set to Igor Skornyakov

#4 - 07/12/2022 07:48 AM - Igor Skornyakov

There is a problem with "on the fly" creation of native UDFs.
Many of them a declared as "leakproof" but such UDFs can only be defined by an admin while the application normally starts with a 'normal' user credentials.
Actually "leakproof" UDF attribute is not absolutely necessary. Here is an excerpt from the PG documentation:

```
LEAKPROOF indicates that the function has no side effects. It reveals no information about its arguments other
 than by its return value. For example, a function which throws an error message for some argument values but
not others, or which includes the argument values in any error message, is not leakproof. This affects how the
 system executes queries against views created with the security_barrier option or tables with row level secur
ity enabled. The system will enforce conditions from security policies and security barrier views before any u
ser-supplied conditions from the query itself that contain non-leakproof functions, in order to prevent the in
advertent exposure of data. Functions and operators marked as leakproof are assumed to be trustworthy, and may
 be executed before conditions from security policies and security barrier views. In addition, functions which
 do not take arguments or which are not passed any arguments from the security barrier view or table do not ha
ve to be marked as leakproof to be executed before security conditions. See CREATE VIEW and Section 41.5. This
 option can only be set by the superuser.
```

Should I remove "LEAKPROOF" attribute from the UDFs definitions?
Thank you.

#5 - 07/12/2022 08:14 AM - Igor Skornyakov

Another problem is with creation of the udf schema. The create schema if not exists udf statament also can be executed only by admin.
Any suggestions?
Thank you.

#6 - 07/12/2022 09:49 AM - Eric Faulhaber

Igor Skornyakov wrote:

> There is a problem with "on the fly" creation of native UDFs.
> Many of them a declared as "leakproof" but such UDFs can only be defined by an admin while the application normally starts with a 'normal' user credentials.
> Actually "leakproof" UDF attribute is not absolutely necessary. Here is an excerpt from the PG documentation:
> [...]
>
> Should I remove "LEAKPROOF" attribute from the UDFs definitions?
> Thank you.

It seems many of the UDFs should be leakproof by design, but others would definitely raise an error for certain arguments and not others. I have not carefully reviewed the UDFs with this constraint in mind. I think it is useful to retain this attribute where it is in fact true.

#7 - 07/12/2022 09:53 AM - Eric Faulhaber

Igor Skornyakov wrote:

> Another problem is with creation of the udf schema. The create schema if not exists udf statament also can be executed only by admin.
> Any suggestions?
> Thank you.

This makes sense. We could add an admin_username and admin_password in the .../database/<database_name>/orm/connection/ section of the directory, with the assumption that directory.xml is a secured document (in fact, we already have this assumption w.r.t. other information in the directory, including the non-superuser database role's username and password). The checking/installing of UDFs at server startup would use a dedicated connection; we would not set up a connection pool for admin connections. Eventually, we'll need to implement secure, passwordless authentication of a connection over JDBC, but not necessarily in the scope of this task.

#8 - 07/12/2022 09:54 AM - Igor Skornyakov

> This makes sense. We could add an admin_username and admin_password in the .../database/<database_name>/orm/connection/ section of the directory, with the assumption that directory.xml is a secured document (in fact, we already have this assumption w.r.t. other information in the directory, including the non-superuser database role's username and password). The checking/installing of UDFs at server startup would use a dedicated connection; we would not set up a connection pool for admin connections. Eventually, we'll need to implement secure, passwordless authentication of a connection over JDBC, but not necessarily in the scope of this task.

Got it. Thank you.

#9 - 07/12/2022 12:00 PM - Igor Skornyakov

- File missinng-udfs.diff added

Done. Please review.
Thank you.

#10 - 07/14/2022 10:43 AM - Eric Faulhaber

Code review missing-udf.diff:

It seems this should work.

Please note we will have to support this feature for every dialect which supports native UDFs. While I prefer not to have the technical debt of PostgreSQL-specific logic in ScriptRunner, I realize the proper division of common code vs. dialect-specific code may not be obvious without the benefit of looking at one or two other dialect implementations. For now, it is OK to leave PostgreSQL-specific code in ScriptRunner, but in the places where it looks likely that some refactoring will be necessary when native UDF support is added for other dialects, please put in more comments (including some TODO markers). In particular, please comment the COUNT instance variable, as this surely will need to differ between dialects.

#11 - 07/14/2022 11:01 AM - Igor Skornyakov

Eric Faulhaber wrote:

> Code review missing-udf.diff:
>
> It seems this should work.
>
> Please note we will have to support this feature for every dialect which supports native UDFs. While I prefer not to have the technical debt of PostgreSQL-specific logic in ScriptRunner, I realize the proper division of common code vs. dialect-specific code may not be obvious without the benefit of looking at one or two other dialect implementations. For now, it is OK to leave PostgreSQL-specific code in ScriptRunner, but in the places where it looks likely that some refactoring will be necessary when native UDF support is added for other dialects, please put in more comments (including some TODO markers). In particular, please comment the COUNT instance variable, as this surely will need to differ between dialects.

I was thinking about refactoring to extract dialect-specific logic to the Dialect class and its childs.
However, finally I came to the same conclusion you're mentioned - it is difficult to decide how it should look like for the dialects other than PostgreSQL and a premature abstraction can be wrong and even make adding support for the new dialects more complicated.

#12 - 07/14/2022 11:05 AM - Eric Faulhaber

- Status changed from New to Review

Right, so let's just go with the additional comments for now, to make that refactoring easier as we add UDF support in new dialects.

#13 - 07/14/2022 11:39 AM - Igor Skornyakov

- Assignee changed from Igor Skornyakov to Eric Faulhaber

- vendor_id deleted (GCD)

Eric Faulhaber wrote:

> Right, so let's just go with the additional comments for now, to make that refactoring easier as we add UDF support in new dialects.

Added TODO comments to the PostgrSQL-specific parts of code.
Committed to 3821c/14066.

#14 - 07/14/2022 11:53 AM - Igor Skornyakov

BTW: I think that right way to re-work ScriptRunner for more dialects is to create its dialect-specific versions instead of moving dialect-specific code to the corresponding Dialect. The common parts will remain in the base class. This approach seems to be more flexible and maintanable.

#15 - 07/15/2022 12:14 AM - Eric Faulhaber

- Status changed from Review to Closed

- % Done changed from 0 to 100

The changes in 3821c/14066 look good.

Note: to ensure missing native UDFs are installed, entries for admin_username and admin_password must be added to the .../database/<database_name>/orm/connection/ section of the directory. These credentials must be for a database account/role which has permission to create a schema (if PostgreSQL) and to create functions.

Files

missinng-udfs.diff	17.8 KB	07/12/2022	Igor Skornyakov