Database - Feature #4022

upgrade to a newer PostgreSQL release

04/01/2019 05:48 PM - Eric Faulhaber

Status:	WIP	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Stanislav Lomany	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**vendor_id:**	GCD

Description

History

#1 - 04/01/2019 06:10 PM - Eric Faulhaber

At the time of this writing, the most recent production PostgreSQL release is 11.2. We are currently working primarily with version 9.5, with a regression testing environment that uses 9.2.

Newer PostgreSQL releases advertise better performance, primarily through parallel processing and improvements to the query planner. The most common use case we see with FWD is many small, fast queries, where parallel processing would not be possible and the query plans already are quite efficient.

However, there are also many cases of more complex queries whose performance may benefit from an upgrade. Also, as we continue to push to optimize the common Progress idioms of accessing data with FIND statements in a loop or other looping (or nested looping) access, we will be composing multi-table joins which may benefit from these improvements.

The main consideration from a development perspective in moving to newer versions is integrating PL/Java. The most recent release (1.5.2) purports to support PostgreSQL 11, but it remains to be seen whether this involves any changes to the build or integration process for PL/Java.

Migrating data across major versions involves a dump and restore cycle.

#2 - 07/17/2020 04:16 PM - Eric Faulhaber

- Status changed from New to WIP

- Assignee set to Stanislav Lomany

#3 - 07/17/2020 04:22 PM - Roger Borrello

Just a note that the last VM I created used PL/java 1.5.5 and PostgreSQL 10.

#4 - 07/17/2020 04:42 PM - Greg Shah

At this point there is now a PostgreSQL 13.0. I don't think we should waste time on the intermediate versions.

#5 - 07/21/2020 06:24 PM - Stanislav Lomany

I'm not able to build the latest pljava 1.5.5 with postgresql 13 beta 2. There're compilation errors caused by incompatible .h files. Pljava 1.5.5 states that it is compatible at build and run time only with postgresql 12.
As well, I cannot use pre-built package for postgresql 12 with postgresql 13:

```
ERROR:  could not load library "/usr/local/pgsql/lib/libpljava-so-1.5.5.so": /usr/local/pgsql/lib/libpljava-so
-1.5.5.so: undefined symbol: elog_finish
```

What should I do next?

#6 - 07/21/2020 06:46 PM - Greg Shah

If 13 is beta, don't use it. Use the latest stable released build.

#7 - 07/22/2020 12:48 PM - Eric Faulhaber

What is the status with v12.x?

#8 - 07/22/2020 03:40 PM - Stanislav Lomany

Import of the customer's data is running.

#9 - 07/22/2020 03:55 PM - Eric Faulhaber

Which version of PostgreSQL are you using?

Were you able to leverage the pre-built PL/Java binaries?

Any issues getting the environment going?

Is the existing FWD documentation still applicable/sufficient?

#10 - 07/22/2020 08:12 PM - Stanislav Lomany

> Which version of PostgreSQL are you using?

12.3

> Were you able to leverage the pre-built PL/Java binaries?

I installed PL/Java package using apt, along with postgresql packages.

> Any issues getting the environment going?
> Is the existing FWD documentation still applicable/sufficient?

It definitively took some time to set up the environment. Especially considering that it's a new installation of FWD. But I believe that the customer's app conversion documentation is good and detailed enough.

#11 - 07/23/2020 08:29 PM - Stanislav Lomany

I've compared times for postgres 12.3 vs 9.5 and they're absolutely the same (in average). I've tested times for:

1. server startup,
2. user login,
3. loading times for three screens.

#12 - 07/24/2020 05:59 AM - Stanislav Lomany

- File not-set.png added

#13 - 07/24/2020 06:02 AM - Stanislav Lomany

- File deleted (not-set.png)

#14 - 07/24/2020 01:06 PM - Stanislav Lomany

The only question I have regarding installation is about this code from the customer's build_db.xml:

```
  <condition property="p2jpl.jar.path" value="${deploy.home.abs}/lib/p2jpl.jar" else="/usr/share/java/p2jpl.ja
r">
      <isset property="isWindows"  />
   </condition>
```

Why we assume p2jpl.jar to be under /usr/share/java/ in Linux? ${deploy.home.abs}/lib/p2jpl.jar seem to me as a more reliable path for any OS.

#15 - 07/24/2020 01:07 PM - Constantin Asofiei

Stanislav Lomany wrote:

> The only question I have regarding installation is about this code from the customer's build_db.xml:
> [...]
>
> Why we assume p2jpl.jar to be under /usr/share/java/ in Linux? ${deploy.home.abs}/lib/p2jpl.jar seem to me as a more reliable path for any OS.

The p2jpl.jar will be installed by the postgres user - so it needs to be a path readable by this user.

#16 - 07/24/2020 01:19 PM - Stanislav Lomany

"Installation of PL/Java Base Support" documentation states that this file should be put *somewhere* in a directory readable by postgres user.

"Importing the database" documentation of the customer's app says nothing about p2jpl.jar expected to be, and to be in this directory. So I support we should add a step about this before the ant import.db step.

#17 - 07/24/2020 01:22 PM - Constantin Asofiei

Stanislav Lomany wrote:

> "Installation of PL/Java Base Support" documentation states that this file should be put *somewhere* in a directory readable by postgres user. "Importing the database" documentation of the customer's app says nothing about p2jpl.jar expected to be, and to be in this directory. So I support we should add a step about this before the ant import.db step.

Yes, edit the customer's wiki documentation, to emphasize that p2jpl.jar is expected to be in /usr/share/java/p2jpl.jar in case of linux.

#18 - 07/24/2020 01:50 PM - Greg Shah

Also, please update the FWD docs to make it more clear where we might put it by default.

#19 - 02/18/2021 01:55 PM - Greg Shah

> > Were you able to leverage the pre-built PL/Java binaries?
>
> I installed PL/Java package using apt, along with postgresql packages.

Stanislav: Please provide more details here.

- Did the package come out of the standard Ubuntu repos? Or did you have to add a PPA to the system?
- What was the PL/Java version?
- What were the specific commands used to install the pre-built PL/Java package?
- Was there one pre-built package that supported multiple PostgreSQL versions, or version-specific pre-built packages or just a pre-built package that only supported one version of PostgreSQL?

#20 - 02/18/2021 03:13 PM - Stanislav Lomany

- Did the package come out of the standard Ubuntu repos? Or did you have to add a PPA to the system?

Ubuntu.

- What was the PL/Java version?

1.5.5

- What were the specific commands used to install the pre-built PL/Java package?

IIRC using apt-get install

- Was there one pre-built package that supported multiple PostgreSQL versions, or version-specific pre-built packages or just a pre-built package that only supported one version of PostgreSQL?

Version-specific: postgresql-9.5-pljava and postgresql-12-pljava, although it seems that only location of the files differ for these two packages.

#21 - 07/01/2021 08:25 PM - Roger Borrello

Greg suggested I create a new task in database, but this task looks well suited for the issues related to installing PostgreSQL and PLJava on a new installation. I have Linux Mint 20.1, which is really using 20.04 of Ubuntu.

```
Linux rfb 5.11.0-22-generic #23~20.04.1-Ubuntu SMP Thu Jun 17 12:51:00 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux
```

I tried to follow the main steps in https://proj.goldencode.com/projects/p2j/wiki/Database_Server_Setup_for_PostgreSQL_on_Linux but utilize software levels that were more current, without going to extremes. The ~/bin/psql_install.sh was the script I was trying to follow.

I installed PG 10, and when I installed pljava, 1.6.2 came:

```
postgresql-10-pljava/focal-pgdg,now 1.6.2-2.pgdg20.04+1 amd64 [installed]
postgresql-pljava-common/focal-pgdg,now 1.6.2-2.pgdg20.04+1 all [installed,automatic]
```

I have issues when I get to UDF Libraries step. I created the cluster:

```
                                                 List of databases
    Name    |   Owner   | Encoding |          Collate           |           Ctype            |   Access privileg
es   
-----------+-----------+----------+----------------------------+----------------------------+------------------
-----
  menu      | fwd_admin | LATIN1   | en_GB@ISO8859-1@basic@FWD | en_GB@ISO8859-1@basic@FWD | 
  postgres  | postgres  | LATIN1   | en_GB@ISO8859-1@basic@FWD | en_GB@ISO8859-1@basic@FWD | 
  template0 | postgres  | LATIN1   | en_GB@ISO8859-1@basic@FWD | en_GB@ISO8859-1@basic@FWD | =c/postgres      
    +
            |           |          |                            |                            | postgres=CTc/post
gres
  template1 | postgres  | LATIN1   | en_GB@ISO8859-1@basic@FWD | en_GB@ISO8859-1@basic@FWD | =c/postgres      
    +
            |           |          |                            |                            | postgres=CTc/post
gres
 (4 rows)
```

But installing P2J yields:

```
psql -U fwd_admin -h localhost -p 5432 -f install_p2j.sql menu

psql:install_p2j.sql:1: ERROR:  schema "sqlj" does not exist
LINE 1: select sqlj.remove_jar('p2j', true);
               ^
psql:install_p2j.sql:2: ERROR:  schema "sqlj" does not exist
LINE 1: select sqlj.install_jar('file:///usr/share/java/p2jpl.jar', ...
               ^
psql:install_p2j.sql:3: ERROR:  schema "sqlj" does not exist
LINE 1: select sqlj.set_classpath('public', 'p2j');
```

When I run "create extension pljava":

```
rfb@rfb:~/project$ psql -U fwd_admin -h localhost -p 5432 -c "create extension pljava" menu
Password for user fwd_admin:
WARNING:  unrecognized configuration parameter "pljava.classpath" 
WARNING:  failed to create Java virtual machine
DETAIL:  JNI_CreateJavaVM returned an error code: -3
ERROR:  cannot use PL/Java before successfully completing its setup
HINT:  Check the log for messages closely preceding this one, detailing what step of setup failed and what wil
l be needed, probably setting one of the "pljava." configuration variables, to complete the setup. If there is
 not enough help in the log, try again with different settings for "log_min_messages" or "log_error_verbosity"
.
```

Eric and I went over quite a few things, but it doesn't appear that the Java 11 I have is really in the way.

What I see from the ~/bin/psql_install.sh script is where I think something is wrong.

```
# build pl/java
echo "** Building pl/java..." 
cd ~/tmp
gzip -d < $pljava_srcs | tar xf -
cd pljava-snapshot.20120525.0
unzip -o $pljava_patch
JAVA_HOME=/usr/lib/jvm/default-java/
export JAVA_HOME
make all
```

```
# install pl/java
echo "** Installing pl/java..."
rm -frv /opt/pljava
mkdir -v /opt/pljava
# the files are owned by root and can't be changed by group
cp -v ~/tmp/pljava-snapshot.20120525.0/build/pljava.jar /opt/pljava/
cp -v ~/tmp/pljava-snapshot.20120525.0/build/objs/pljava.so /opt/pljava/
cp -v ~/tmp/pljava-snapshot.20120525.0/src/sql/*.sql /opt/pljava/

dir="/opt/pljava/$u"
rm -fr $dir
install -d $dir
cp $p2jpl_jar $dir

dir="/opt/pljava"
chown -R $u:$u /opt/pljava
chmod -R gou+rX /opt/pljava
chmod -R u+w /opt/pljava
chmod go-w /opt/pljava/pljava.* /opt/pljava/*.sql
```

There is /opt/pljava/install.sql that seems to create the sqlj schema, but it didn't get execute. I tried, and get the below:

```
rfb@rfb:/opt/pljava$ psql -U fwd_admin -h localhost -p 5432 -f install.sql menu
Password for user fwd_admin: 
CREATE SCHEMA
GRANT
psql:/opt/pljava/install.sql:6: ERROR:  could not access file "pljava": No such file or directory
psql:/opt/pljava/install.sql:8: ERROR:  function sqlj.java_call_handler() does not exist
psql:/opt/pljava/install.sql:12: ERROR:  could not access file "pljava": No such file or directory
psql:/opt/pljava/install.sql:14: ERROR:  function sqlj.javau_call_handler() does not exist
CREATE TABLE
GRANT
CREATE TABLE
GRANT
ALTER TABLE
CREATE TABLE
GRANT
CREATE TABLE
GRANT
psql:/opt/pljava/install.sql:55: ERROR:  language "java" does not exist
psql:/opt/pljava/install.sql:59: ERROR:  language "java" does not exist
psql:/opt/pljava/install.sql:63: ERROR:  language "java" does not exist
psql:/opt/pljava/install.sql:67: ERROR:  language "java" does not exist
psql:/opt/pljava/install.sql:71: ERROR:  language "java" does not exist
psql:/opt/pljava/install.sql:75: ERROR:  language "java" does not exist
psql:/opt/pljava/install.sql:79: ERROR:  language "java" does not exist
psql:/opt/pljava/install.sql:83: ERROR:  language "java" does not exist
psql:/opt/pljava/install.sql:87: ERROR:  language "java" does not exist
```

Any help would be appreciated.

#22 - 07/01/2021 10:07 PM - Eric Faulhaber

Roger Borrello wrote:

> ...
> Eric and I went over quite a few things, but it doesn't appear that the Java 11 I have is really in the way.

This warning during create extension pljava, is what led me to the conclusion that there is some low-level clash between Java 11 and Java 8:

```
WARNING:  failed to create Java virtual machine
DETAIL:  JNI_CreateJavaVM returned an error code: -3
```

From jni.h (see https://github.com/openjdk/jdk/blob/master/src/java.base/share/native/include/jni.h):

```
/*
 * possible return values for JNI functions.
 */

#define JNI_OK           0                 /* success */
#define JNI_ERR          (-1)              /* unknown error */
#define JNI_EDETACHED    (-2)              /* thread detached from the VM */
#define JNI_EVERSION     (-3)              /* JNI version error */
#define JNI_ENOMEM       (-4)              /* not enough memory */
#define JNI_EEXIST       (-5)              /* VM already created */
#define JNI_EINVAL       (-6)              /* invalid arguments */
```

The -3 return code is JNI_EVERSION, indicating a "JNI version error". I am interpreting this as a possible conflict with Java 11, but it could mean something else.

I know we reviewed the following:

- update-java-alternatives --verbose --list and everything looked ok at the executable level;
- /usr/lib/jvm/default-java pointed to the correct, Java 8 location;
- in postgresql.conf, you had pljava.libjvm_location = '/usr/lib/jvm/default-java/jre/lib/amd64/server/libjvm.so' set.

But I still suspect we missed something in this regard. Have you tried uninstalling Java 11 altogether?

#23 - 07/02/2021 12:08 AM - Roger Borrello

- File install.sql added

What are your thoughts on the /opt/pljava/install.sql script (attached) in terms of when/how to run it? I don't see mention of creating that schema in our documentation.

#24 - 07/02/2021 02:12 AM - Eric Faulhaber

Roger Borrello wrote:

> What are your thoughts on the /opt/pljava/install.sql script (attached) in terms of when/how to run it? I don't see mention of creating that schema in our documentation.

This script was used in pre-1.5.x PL/Java; it is deprecated now. create extension pljava replaced it.

Eric Faulhaber wrote:

> Have you tried uninstalling Java 11 altogether?

The warning message is telling us there is a JNI version issue when running create extension pljava. The only other version mismatch we came across was the psql client being version 13 and the database server being version 10, but that is supposed to work, and it shouldn't have any impact on JNI, AFAIK.

#25 - 07/02/2021 01:05 PM - Roger Borrello

I started reading the fine manual at PL/Java: https://tada.github.io/pljava/install/install.html

When I ran pg_config --version, it indicated I was using 13. How, since I haven't done anything to configure 13, just 10, I thought it might be good to eradicate that first.

```
rfb@rfb:~$ pg_config --version
PostgreSQL 13.3 (Ubuntu 13.3-1.pgdg20.04+1)
```

sudo apt remove postgresql-13 and sudo apt purge postgresql-13 with a reboot weren't good enough, as pg_config --version still showed 13. I had to reinstall 10. Finally, I am at:

```
PostgreSQL 10.17 (Ubuntu 10.17-1.pgdg20.04+1)
```

Another thing I found in looking at the PL/Java installation guide, it may be that the pljava.classpath in the postgresql.conf should be pljava_module_path.

So getting to the Java 11 uninstall... it also uninstall pljava. So when this is done, I'll reinstall and see if it's in a better place.

#26 - 07/02/2021 01:09 PM - Roger Borrello

Roger Borrello wrote:

> So getting to the Java 11 uninstall... it also uninstall pljava. So when this is done, I'll reinstall and see if it's in a better place.

When I reinstalled, it mentioned:

```
The following additional packages will be installed:
  default-jre-headless openjdk-11-jre-headless
```

So it looks like Java 11 will be back.

#27 - 07/02/2021 01:45 PM - Eric Faulhaber

OK, don't install PL/Java through apt. This is actually untested, so you are breaking new ground here.

Does PL/Java 1.6.2 *require* Java 11, or is this possibly just a dependency added by the Ubuntu package maintainers?

Please follow the documented instructions for installing PL/Java instead of using apt, but apply 1.6.2 instead of 1.5.2. If this doesn't work, and PL/Java really requires Java 11 (which I doubt), then we don't support that, since Java 8 is the current requirement for FWD. In that case, you will need to drop back to 1.5.2, which I believe already supported PostgreSQL 10 (but confirm that first).

#28 - 07/02/2021 01:56 PM - Constantin Asofiei

I went through setting up PL/Java 1.6.2 last week (with psql 12.7). 1.6.2 does require Java 11 - it is compiled as such, you can't run it with Java 8.

My install process was this:

```
# pl/java
sudo apt-get install openjdk-11-jre-headless
wget https://apt.postgresql.org/pub/repos/apt/pool/main/p/postgresql-pljava/postgresql-pljava-common_1.6.2-2.p
gdg20.04%2B1_all.deb
sudo dpkg -i postgresql-pljava-common_1.6.2-2.pgdg20.04+1_all.deb
wget https://apt.postgresql.org/pub/repos/apt/pool/main/p/postgresql-pljava/postgresql-12-pljava_1.6.2-2.pgdg2
0.04%2B1_amd64.deb
```

```
sudo dpkg -i postgresql-12-pljava_1.6.2-2.pgdg20.04+1_amd64.deb
```

The above downloads the package for ubuntu 20.04. If you have another version, then you need to choose that. Also, I don't recall if pljava-common was really needed in the end or not.

After this, to configure it for postgres I had to add these lines:

```
sudo vi /etc/postgresql/12/main/postgresql.conf
# add at end of file
pljava.libjvm_location = '/usr/lib/jvm/java-11-openjdk-amd64/lib/server/libjvm.so'
pljava.module_path = '/usr/share/postgresql/12/pljava/pljava-1.6.2.jar:/usr/share/postgresql/12/pljava/pljava-
api-1.6.2.jar'
pljava.vmoptions = '-Xmx16m -Djava.awt.headless=true'
```

Note the module_path instead of classpath.

I also had to add these for some WORD query issues, in the pljava.policy file specified in the pg_config --sysconfdir folder:

```
grant {
    permission java.util.PropertyPermission
        "WORD_DELIMITERS", "read";
    permission java.lang.RuntimePermission
        "getenv.WORD_DELIMITERS", "read";
};
```

Please read the above as 'unofficial steps'. There may be a better way to do this, but I didn't bother once I got it working.

#29 - 07/02/2021 01:57 PM - Roger Borrello

The release notes for PL/Java https://tada.github.io/pljava/releasenotes.html indicate the moved to the Java 9 module system, where pljava.classpath -> pljava.module_path

> Because PL/Java itself is now modular code conforming to the module system introduced with Java 9, one configuration variable has changed: pljava.classpath is now pljava.module_path.
>
> As before, its default value will be correct when PL/Java is installed to the usual locations. It should be rare for any installation to have needed to think about the old one, or to need to think about the new one. For a rare installation that does, the details are in the documentation.
>
> In this release, user code is not treated as modular; the SQLJ.INSTALL_JAR routine still treats its jars as unnamed-module code on a class path, as before.

So I went back to PL/Java 1.5.6. The current documentation is mostly accurate. Here's where I grabbed the pre-built:

```
wget https://apt.postgresql.org/pub/repos/apt/pool/main/p/postgresql-pljava/postgresql-10-pljava_1.5.6-1.pgdg9
0%2B1_amd64.deb
sudo dpkg -i postgresql-10-pljava_1.5.6-1.pgdg90+1_amd64.deb
```

#30 - 07/02/2021 02:01 PM - Roger Borrello

Constantin Asofiei wrote:

> Please read the above as 'unofficial steps'. There may be a better way to do this, but I didn't bother once I got it working.

Thanks so much, Constantin. This is a great starting point. However, I was really trying to polish up the wiki for a customer, and wanted to make sure they didn't have to run through as many hoops as we are typically willing to do. When we can move off Java 8, it should be much easier to configure PL/Java... if we still require it ;-)

#31 - 07/05/2021 06:51 PM - Roger Borrello

I noticed that postgresql-10-pljava was on the auto-update list, and pljava was upgraded at 5pm to 1.6.2-2.pgdg20.04+1 against my wishes. I had to blacklist it, so that I wouldn't have to get to the Java 11 level.

Just a reminder for anyone documenting.

#32 - 07/15/2021 06:30 PM - Roger Borrello

I have been trying to get my test system going using PostgreSQL 13 for updated client code, and when I get to the PL/Java steps of ant import.db which is:

```
psql -U fwd_admin -h localhost -p 5432 -c "select sqlj.install_jar('file:///usr/share/java/p2jpl.jar', 'p2j', 
true)" db
```

It results in:

```
ERROR:  java.lang.OutOfMemoryError: Java heap space
```

I went through Constantin's steps to get the Java 11 to work. I've rebuilt the cluster and still continue to get the error. Anyone have a clue what would lead to that? I am guessing that the wrong jvm is being launched but the postgresql.conf points to the Java 11 version.

#33 - 07/21/2021 10:02 AM - Roger Borrello

I was able to alleviate the error by going back to PL/Java 1.5.6, and abandoning the 1.6.2 version. Good riddance (for now).

#34 - 10/08/2021 11:56 AM - Eric Faulhaber

The primary maintainer of PL/Java, Chapman Flack, noticed this discussion and proactively sent us the following information per email. Thank you, Chapman!

Hi Eric and Greg,

Chapman Flack, PL/Java maintainer here, again.

I was asking Google who had PL/Java questions recently, and it seems
that there is a current issue in your ticketing system:

#4022

that was opened a couple years ago and has had recent activity in
the last couple of months.

I don't have a registration to just answer on the ticket, but maybe
I can help with some of the questions I have seen come up there.

The biggest news (as you have kind of discovered) is that PL/Java
now has two parallel lines of releases: the legacy 1.5.x, and the
current recommended 1.6.x.

1.5.x is still (barely) maintained for supporting legacy workloads.
It will work with any PostgreSQL back to 8.2 and any Java back to 6.
That's some seriously old stuff. It will also work with the latest new
stuff (PostgreSQL 14 and Java 17), but only with PL/Java 1.5 features.

1.6.x is where the improvements happen. It's recommended if you're not
supporting crazy-old stuff. It will work with Java back to 9 and PostgreSQL
back to 9.5. 1.6.2 supports PostgreSQL up to 13; when I release 1.6.3
later this week, it will support PG 14.

In your ticket #4022, comment #4022-22 assumed there was some JNI version
clash, but I suspect something else was going on. As a rule, PL/Java
does not care what Java version you are running it on. There is the
version you used to build it (which has to be 8 to 14 for 1.5.x, or
9 to current for 1.6.x), but once it is built, you just set
pljava.libjvm_location in the database to the libjvm.so of any Java
version you would like it to use, and it uses that. For 1.6, it has
to be a Java 9 or later.

A nice thing there is that while PL/Java itself keeps its compatibility
with a broad range of Java versions, if you point libjvm_location at
a shiny recent-version JVM, your own PL/Java functions can use
all the shiny recent-version cool Java stuff.

Comment #4022-25 had it right that going from PL/Java 1.5.x to 1.6.x,
pljava.classpath changes to pljava.module_path. BUT ... I think there
is something in your procedures that is being made harder than
necessary, because you should hardly ever need to supply either of
those settings. If you just leave them unset, they default to the
standard places, exactly where the standard PL/Java installation will
have put the files.

Comment #4022-27 - your FWD product will run in Java 8, but it won't run
in later versions? I haven't run into that very much. Have you seen
specific things go wrong when you try it in a later version? I can't
guarantee it's anything I can help with, but if you mention the issues,
maybe I'd have some ideas.

Note that while PL/Java 1.6 has moved to the Java 9 module system for
itself, that doesn't mean it requires any change in user code. The same
ol' jars with the same ol' stuff ought to run the same ol' way.

1.6 gives more security control and some editing of the policy file
might be needed at first. Constantin got a start on that in #28.

#4022-31 ... yes, I wish the apt packaging treated the 1.5.x and 1.6.x
as different series, but it doesn't, so it can surprise auto-update
you from a 1.5.x to a 1.6.x. Sorry about that.

#4022-32 OutOfMemoryError ... there was an email I sent you guys back on
7/29/2019 describing memory settings in pljava.vmoptions that could
be of help there.

Overall, if you find that you have a ticket where time is being spent
on some puzzle with PL/Java, don't be shy about letting me know, or
opening a GitHub issue or putting the pljava tag on a Stack Overflow
question.

Regards,
-Chap

#35 - 10/08/2021 12:36 PM - Roger Borrello

Eric Faulhaber wrote:

> The primary maintainer of PL/Java, Chapman Flack, noticed this discussion and proactively sent us the following information per email. Thank you, Chapman!

What a great help! A very nice chap is Chap!

> Comment #4022-27 - your FWD product will run in Java 8, but it won't run
> in later versions? I haven't run into that very much. Have you seen
> specific things go wrong when you try it in a later version? I can't
> guarantee it's anything I can help with, but if you mention the issues,
> maybe I'd have some ideas.
>
> Note that while PL/Java 1.6 has moved to the Java 9 module system for
> itself, that doesn't mean it requires any change in user code. The same
> ol' jars with the same ol' stuff ought to run the same ol' way.
>
> 1.6 gives more security control and some editing of the policy file
> might be needed at first. Constantin got a start on that in #28.

All-in-all it looks to me like FWD limitations to moving off Java 8 will lead to us sticking with PL/Java 1.5.x. Is that what you see?

> #4022-31 ... yes, I wish the apt packaging treated the 1.5.x and 1.6.x
> as different series, but it doesn't, so it can surprise auto-update
> you from a 1.5.x to a 1.6.x. Sorry about that.

Moot, but I wonder if he had named it 2.1.x it would have allowed apt to determine a different series?

> #4022-32 OutOfMemoryError ... there was an email I sent you guys back on
> 7/29/2019 describing memory settings in pljava.vmoptions that could
> be of help there.

This could be handy to post here, but as Chap indicated, we shouldn't have an issue with PL/Java inhibiting us to support newer PostGreSQL, which is what this task is for. But we may have an issue moving from PL/Java 1.5.x to 1.6.x vis-a-vis Java 8.

#36 - 10/08/2021 01:17 PM - Eric Faulhaber

Roger Borrello wrote:

> All-in-all it looks to me like FWD limitations to moving off Java 8 will lead to us sticking with PL/Java 1.5.x. Is that what you see?

Here's some follow-up clarification provided by Chapman:

> As long as the code implementing in-db functions will run in Java 9 or above, it's possible to have more than one JRE installed on the box, and you can have 8 set as the default for your external code, and in the db set pljava.libjvm_location to the libjvm.so for the later one.

I don't know of any limitation in the FWD code called within the PL/Java UDFs that would require Java 8, so we should be good to use PL/Java 1.6.x with newer PostgreSQL and Java. Allowing Ubuntu to manage the PL/Java installation is another matter.

Files

install.sql	2.82 KB	07/02/2021	Roger Borrello