Configuring PostgreSQL on UCB W205 AMI

In order to persist Hive and SparkSQL metadata between reboots on the W205 EC2 AMI, we need a database which is more robust than the simple Derby database configured by default. Additionally, as we explore RDBMS concepts and more advanced SQL concepts, it is beneficial to have a complete RDBMS solution in place. We will use PostgreSQL to meet both of these requirements.

In order to install and configure PostgreSQL, we will need to make some additions to our /data directory. Follow the steps below to configure your /data directory and environment.

Installing Postgres

First mount your data directory:

```
mount -t ext3 /dev/<attached volume location> /data
```

Next, make sure the top-level data directory is open to all users:

```
chmod a+rwx /data
```

Ensure that postgres is installed:

```
yum install postgresql postgresql-server postgresql-jdbc
```

If posgresql was NOT installed, you should make a new AMI at the end of this worksheet.

Create a directory and database files for postgres:

```
mkdir /data/pgsql
mkdir /data/pgsql/data
mkdir /data/pgsql/logs
chown -R postgres /data/pgsql
su postgres
initdb -D /data/pgsql/data
```

Edit the /data/pgsql/postgresql.conf file as follows:

Change

```
#listen_addresses = 'localhost'
```

to

```
listen_addresses = ‘*’
```

Change

```
#standard_conforming_strings = off
```

to

```
standard_conforming_strings = off
```

Edit the /data/pgsql/pg_hba.conf file. At the end of the file add the following line:

```
host    all         all         0.0.0.0         0.0.0.0
md5
```

This allows the database to listen and authenticate users on all network interfaces.

Starting postgres

To start the database, we need to use the pg_ctl command:

```
cd /data
sudo -u postgres pg_ctl -D /data/pgsql/data -l
/data/pgsql/logs/pgsql.log start
```

In a file called /data/start_postgres.sh, place the following:

```
#! /bin/bash
sudo -u postgres pg_ctl -D /data/pgsql/data -l
/data/pgsql/logs/pgsql.log start
```

Close the file. Make it executable by typing:

```
chmod +x /data/start_postgres.sh
```

In a file called /data/stop_postgres.sh, place the following:

```
#! /bin/bash
sudo -u postgres pg_ctl -D /data/pgsql/data -l
/data/pgsql/logs/pgsql.log stop
```

Close the file. Make it executable by typing:

```
chmod +x /data/stop_postgres.sh
```

Creating a database for yourself

We will use PostgreSQL later in the course, so it's useful to have a database all to ourselves. We'll do this with the psql command line. Type:

```
sudo -u postgres psql
```

You should see:

```
psql (8.4.20)
Type "help" for help.

postgres=#
```

Type the following at the prompt, hitting ENTER at the end of each line:

```
CREATE USER <your user> WITH PASSWORD 'postgres';
CREATE DATABASE w205;
```

```
ALTER DATABASE w205 OWNER TO w205;
GRANT ALL ON DATABASE w205 TO w205;
\q
```

You can now connect to your personal database by calling:

```
psql –U <your user> -d w205
```

Creating the Hive Metastore

We need a database for Hive and SparkSQL to share that is separate from our personal database.
First, we need to make sure the PostgreSQL JDBC driver is in a place Hive can find it.

As root, run:

```
ln -s /usr/share/java/postgresql-jdbc.jar
/usr/lib/hive/lib/postgresql-jdbc.jar
```

Now, we need to create a database for Hive and SparkSQL to use. We'll do this in the psql tool:

```
sudo -u postgres psql
```

You should see:

```
psql (8.4.20)
Type "help" for help.

postgres=#
```

Type the following at the prompt, hitting ENTER at the end of each line:

```
CREATE USER hiveuser WITH PASSWORD 'hive';
CREATE DATABASE metastore;
\c metastore
\i /usr/lib/hive/scripts/metastore/upgrade/postgres/hive-
schema-1.1.0.postgres.sql
\i /usr/lib/hive/scripts/metastore/upgrade/postgres/hive-
txn-schema-0.13.0.postgres.sql
\c metastore
\pset tuples_only on
\o /tmp/grant-privs
SELECT 'GRANT SELECT,INSERT,UPDATE,DELETE ON "'  ||
schemaname || '". "' ||tablename ||'" TO hiveuser ;'
FROM pg_tables
WHERE tableowner = CURRENT_USER and schemaname = 'public';
\o
\pset tuples_only off
\i /tmp/grant-privs
\q
```

Test that the hiveuser works:

```
psql -U hiveuser -d metastore
```

```
\q
```

Creating your personal Hive configuration

We need to create a personal configuration for hive, so that our changes our preserved between reboots.

First, start hadoop using your start-hadoop.sh script from the previous worksheet.

Next, add a directory to /data/hive:

```
sudo -u hadoop mkdir /data/hadoop/hive/conf
```

Now copy your existing hive-site.xml to the configuration directory:

```
sudo -u hadoop cp /etc/hive/conf/* /data/hadoop/hive/conf/
```

Next, we need to edit our hive-site.xml to use postgresql as the metastore:

Remove the following:

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>

<value>jdbc:derby:;databaseName=/data/${user.name}/hive/meta
store/metastore_db;create=true</value>
  <description>JDBC connect string for a JDBC
metastore</description>
</property>

<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>org.apache.derby.jdbc.EmbeddedDriver</value>
  <description>Driver class name for a JDBC
metastore</description>
</property>
```

And add the following:

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:postgresql://localhost:5432/metastore</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>org.postgresql.Driver</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>hiveuser</value>
</property>
```

```
<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>hive</value>
</property>

<property>
  <name>datanucleus.autoCreateSchema</name>
  <value>false</value>
</property>

<!-- <property>
  <name>hive.metastore.uris</name>
  <value>thrift://localhost:9083</value>
  <description>IP address (or fully-qualified domain name)
and port of the metastore host</description>
</property>
-->

<property>
<name>hive.metastore.schema.verification</name>
<value>true</value>
</property>
```

Test the hive installation (start as root):

```
su - <your user>
export HIVE_CONF_DIR=/data/hadoop/hive/conf
hive -e ‘show tables;’
exit
```

Tip

If you are going to make a new AMI, edit /etc/profile to include the following line at the end of the file:

```
export HIVE_CONF_DIR=/data/hadoop/hive/conf
```