RoadShow Demo Cheat Sheet

Components Installed

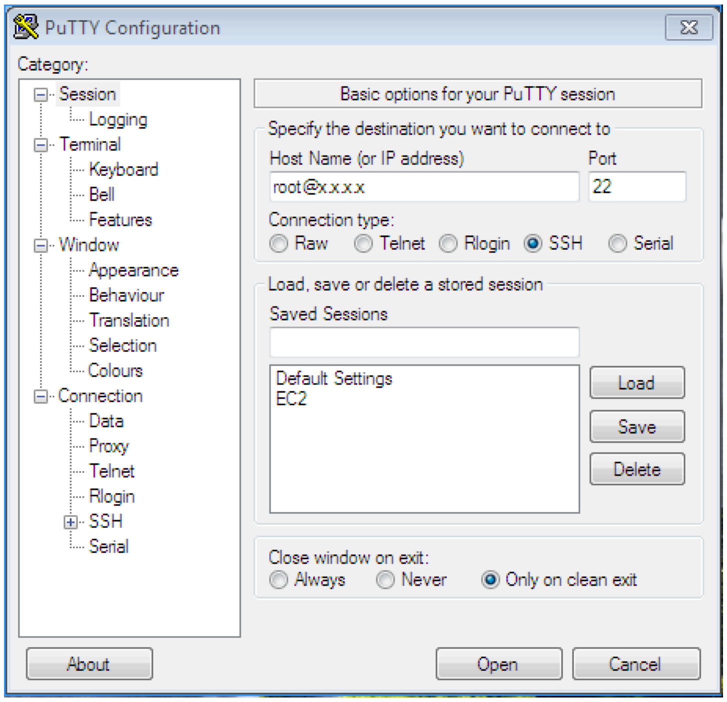
- Pivotal Greenplum Database (Analytical Distributed Database)
- Pivotal Gemfire (In-memory Database)
- Pivotal Hadoop (Pivotal Hadoop Distribution)
- Pivotal HAWQ (SQL Engine on HDFS)
- Pivotal SpringXD (Real-time stream/Ingestion tool)

NOTE: For OSX Users, please use open iTerm Or Terminal
Windows Users, use putty.exe here

1. Login to the virtual machine (Amazon Web Services)

```
ssh root@<assigned IP address>
```

On Windows:



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): root@x.x.x.x

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

Saved Sessions

Default Settings
EC2

Close window on exit: Always Never [✓ Only on clean exit]

NOTE: Use ‘changeme’ as password when prompted

2. Change login from root user to gpadmin user

```
[root@admin]# su - gpadmin
[gpadmin@admin]$
```

3. Start Greenplum Database & create model tables

```
[gpadmin@admin]$ source
/staging/FraudDectDemo/Server/scripts/sourcegpdb
[gpadmin@admin]$ gpstart –a
[gpadmin@admin]$ cd /staging/FraudDectDemo/Server/scripts
[gpadmin@admin]$ psql gemfire -f model1.sql
```

4. Change directory

```
[gpadmin@admin]$ cd /staging/FraudDectDemo/Server
```

5. Start Gemfire

```
[gpadmin@admin]$ ./startup.sh
```

6. Start Web Application

```
[gpadmin@admin]$ cd /staging/FraudDectDemo/WebConsole
[gpadmin@admin]$ ./gradlew bootRun
```

7. Verify the Web Interface using the following URL on a Web Browser, preferably Chrome OR Firefox

```
http://<assigned ip address>:8080/index.html
```

8. Open second Putty.exe/iTerm/Terminal session for the same IP address and change user as gpadmin

9. From second session, Start PoS Emulator to generate transaction data

```
[gpadmin@admin staging]$ cd /staging/FraudDectDemo/PoS_Emulator/
[gpadmin@admin PoS_Emulator]$ ./generate.sh
```

10. Wait for about 20 seconds and create the objects used for training using Regression Data Model in Pivotal Greenplum Database

```
[gpadmin@admin]$ source
/staging/FraudDectDemo/Server/scripts/sourcegpdb
[gpadmin@admin Server]$ cd /staging/FraudDectDemo/Server/scripts/
[gpadmin@admin scripts]$ psql gemfire -f model2.sql
```

11. Open third Putty.exe/iTerm/Terminal session for the same IP address and change user as gpadmin

12. Wait for 20 seconds and execute the following to train the data model in Greenplum Database

```
[gpadmin@admin]$ source
/staging/FraudDectDemo/Server/scripts/sourcegpdb
[gpadmin@admin Server]$ cd /staging/FraudDectDemo/Server/scripts/
[gpadmin@admin scripts]$ psql gemfire -f train.sql
```

13. Monitor the Web Interface for Fraud Detection Prediction based on newly trained data
14. Stop the demo for demonstration of use case for SpringXD and Hawq

```
 [gpadmin@admin Server]$ cd /staging/FraudDectDemo/Server/scripts/
[gpadmin@admin scripts]$ ./stop_demo1.sh
```

15. Exit and return as root user

```
 [gpadmin@admin Server]$ exit
[gpadmin@admin scripts]#
```

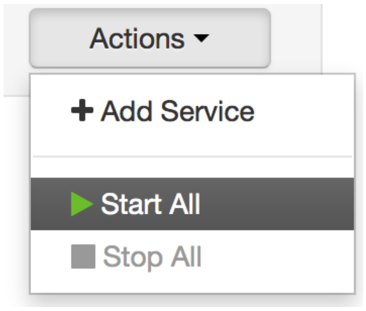
16. Close the first putty/iTerm session and Start Ambari Server in the third putty session

```
 [root@admin ~]# ambari-server start
```

17. Open a web browser and open the following URL

```
http://<assigned IP address>:8080
```

18. On the top left section of the browser, click on the dropdown tab named "Actions" and select "Start All"



19. Once all the services are up and green, shutdown the Ambari server process

```
[root@admin ~]# ambari-server stop
```

20. Start Spring-XD single node

```
[gpadmin@admin ~]$ xd-singlenode
```

21. Open a new putty.exe/iTerm session with the assigned IP address, login as root and change user to gpadmin

22. Start the xd-shell for creation of twitter stream and data ingestion process in the new putty.exe session/iTerm

```
[gpadmin@admin ~]$ xd-shell
```

Open the Spring-XD Web interface using following

```
http://<assigned-ip-address>:9393/admin-ui
```

23. From xd-shell interface, use the following code to deploy twitter stream

```
stream create tweets --definition "twitterstream --consumerSecret=xxx --
consumerKey=xx --accessToken=xx --accessTokenSecret=xx| log"

stream create tweetlang  --definition "tap:stream:tweets > field-value-counter --
fieldName=lang" --deploy

stream create tweetcount --definition "tap:stream:tweets > aggregate-counter" --deploy

stream create tagcount --definition "tap:stream:tweets > field-value-counter --
fieldName=entities.hashtags.text --name=hashtags" --deploy

stream deploy tweets
```

NOTE: Monitor the xd-singlenode interface to see the output generated from twitterstream. To avail twitter feed, you must have account on https://apps.twitter.com

24. Undeploy all the existing streams

```
stream undeploy tweets

stream undeploy tweetlang

stream undeploy tweetcount

stream undeploy tagcount
```

25. Create a new stream in xd-shell, using the following command

```
stream create tweets_hfs --definition "twitterstream --consumerSecret=xxx --
consumerKey=xxx --accessToken=xxx --accessTokenSecret=xxx| hdfs --rollover=20M --
idleTimeout=100" --deploy
```

26. Monitor Hadoop Filesystem to see the data being populated, run this command to list directory couple of times

```
[gpadmin@admin]$ sudo -u hdfs hdfs dfs -ls /xd/tweets_hfs
```

27. Review the table in Pivotal HDB a.k.a HAWQ, which reads the data from HDFS

```
[gpadmin@admin]$ psql gemfire -c “select count(*) from tweets_in_hawq”
[gpadmin@admin]$ psql gemfire -c “select * from tweets_in_hawq limit 10”
```

28. Review the external table definition of Pivotal Hadoop

```
[gpadmin@admin ~]$ psql gemfire
psql (8.2.15)
Type "help" for help.

gemfire=# \d tweets_in_hawq
```