Tableau Spark SQL Setup Instructions

1. Prerequisites

There are a number of prerequisites required to be able to run Tableau with Spark SQL. The main requirements are:

Server Side:

- Spark V1.2 - please use the 1.2 branch at https://github.com/apache/spark/tree/branch-1.2
- Hadoop V2.4 or higher
- Hive V0.12 or V0.13

Client Side:

- Tableau 8.3.1+

- Simba Spark ODBC Driver V1.0.4:
    - http://databricks.com/spark-odbc-driver-download

2. Configuring Hive

- There are no special Hive configurations when using with Spark SQL
- If installing from scratch you can follow the Appendix 6B steps for our sample spark cluster configuration

3. Configuring Spark & Hive

- There are no special Spark configurations, the defaults will get you up and running
- See Appendix 6B for our sample cluster configuration

4. Starting the Spark Service and the Spark Thrift Server

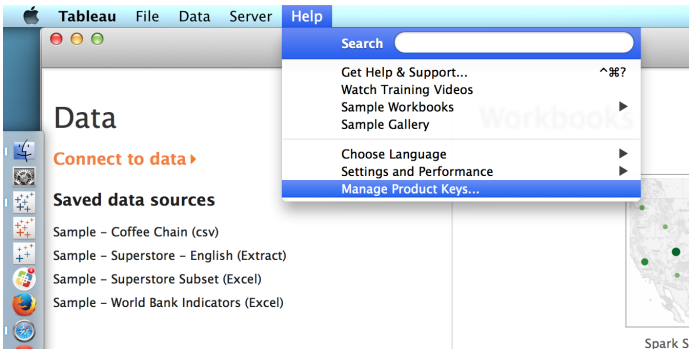
- **Verify that you have HiveServer2 running and you are using PostgreSQL or MySQL as a metastore then run the following**
- **$> $SPARK_HOME/sbin/start-master.sh**
- **$> $SPARK_HOME/sbin/start-slaves.sh**
- **$> $SPARK_HOME/sbin/start-thriftserver.sh --master spark://localhost:7077 --driver-class-path $CLASSPATH --hiveconf hive.server2.thrift.bind.host localhost --hiveconf hive.server2.thrift.port 10001**
    - **Note, we randomly choose port 10001**

5. Connecting Tableau to Spark SQL

5A. Install Tableau DevBuild 8.3.1+

The first thing you must do is install the latest version of Tableau - anything 8.3.1 or later should work. The Spark SQL connection will be hidden in the product unless you install a special license key. Please e-mail Jackie Clough if you do not have the special license key.

To install a new key you must go to Help -> Manage Product Keys



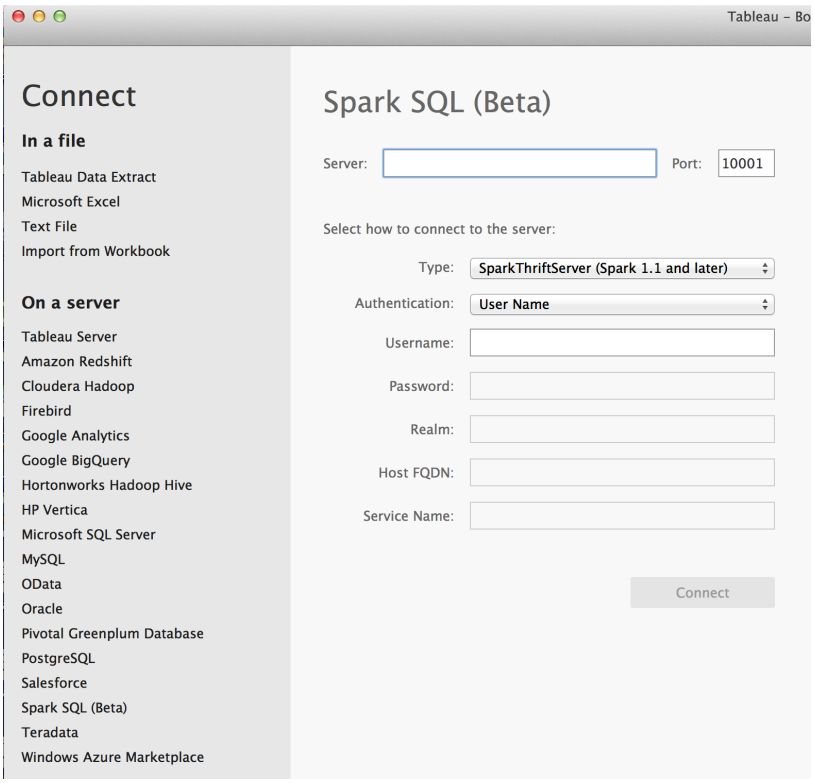
5B. Install the Spark SQL ODBC

To install the Spark SQL ODBC driver, simply open the appropriate version of the driver for your system and follow the instructions:

- Windows 64-bit: SimbaSparkODBC64.msi
- Windows 32-bit: SimbaSparkODBC32.msi
- Max OSX: SimbaSparkODBC.dmg
    - When installing the Mac driver, you may get a message that says "SimbaSparkODBC.dmg can't be opened because it is from an unidentified developer." To allow the driver to be installed, go to Applications -> System Preferences -> Security & Privacy -> General Tab and click Open Anyways

5C. Opening a Spark SQL ODBC Connection

If you have properly installed Tableau and the special license key, you should see *Spark SQL (Beta)* as one of the connection options after clicking *Connect to Data*. Select Spark SQL (Beta) and you will see a dialog box similar to below:



The parameters you need to enter include:

- **Server:** Server name or IP address of your Spark server
- **Port:** Default value of your Spark Thrift server port
- **Type:** Spark ThriftServer (Spark 1.1 and later)
- **Authentication:** User Name
- **User Name:** <blank>

6. Appendix: Spark SQL 1.1.x Installation Steps

6A. Pre-Requisites Sample:

- OS: CentOS 6.5
- CPU: 2 dual core
- RAM: 16GB

6B. Apache Hadoop Install:

Install Java:

- Copy/scp the java rpm file
  - jdk-7u25-linux-x64.rpm
- to /tmp and extract/install with rpm:
  - rpm -Uvh jdk-7u25-linux-x64.rpm
- Set environment variables:
  - export JAVA_HOME=/usr/java/jdk1.7.0_25/
- Verify java version:
  - java -version

Install Hadoop:

- ssh-keygen -t rsa -P ""
- cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
- ssh localhost
- ssh <actual server name>
- useradd hadoop
- cd /
- wget http://apache.mirror.uber.com.au/hadoop/common/current/hadoop-2.4.1.tar.gz
- tar xzvf hadoop-2.4.1.tar.gz
- chown -R hadoop:hadoop /hadoop-2.4.1

Edit Config Files

Edit the following config files located in /hadoop-2.4.1/etc/hadoop, These will vary depending on your environment but are provided here as a sample:

- core-site.xml

```
<configuration>
 <property>
  <name>fs.defaultFS</name>
    <value>hdfs://localhost:8020</value>
    <final>true</final>
 </property>

 <property>
  <name>hadoop.tmp.dir</name>
    <value>/data/hadoop_data</value>
    <description>A base for other temporary directories.</description>
 </property>
</configuration>
```

- mapred-site.xml

```
<configuration>
 <property>
  <name>mapreduce.framework.name</name>
    <value>yarn</value>
 </property>
</configuration>
```

- yarn-site.xml

```
<configuration>
 <property>
  <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
 </property>
 <property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
 </property>

 <!-- To increase number of apps that can run in YARN -->
 <property>
  <name>yarn.nodemanager.resource.cpu-vcores</name>
    <value>4</value>
 </property>
 <property>
  <name>yarn.nodemanager.resource.memory-mb</name>
    <value>8192</value>
 </property>
 <property>
  <name>yarn.scheduler.minimum-allocation-mb</name>
```

```
    <value>512</value>
  </property>
  <property>
   <name>yarn.nodemanager.pmem-check-enabled</name>
    <value>false</value>
  </property>
  <property>
   <name>yarn.nodemanager.vmem-check-enabled</name>
    <value>false</value>
  </property>
</configuration>
```

In addition, add the following environment variables to ~/.bashrc:

- export JAVA_HOME=/usr/java/jdk1.7.0_25
- export HADOOP_PREFIX=/hadoop-2.4.1
- export HADOOP_CONF_DIR=$HADOOP_PREFIX/etc/hadoop
- export YARN_CONF_DIR=$HADOOP_CONF_DIR
- export PATH=$PATH:$HADOOP_PREFIX/bin
- export HADOOP_INSTALL=/hadoop-2.4.1
- export HADOOP_HOME=/hadoop-2.4.1
- export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
- export HADOOP_OPTS="$HADOOP_OPTS -Djava.library.path=$HADOOP_HOME/lib/"
- export HIVE_HOME=/usr/local/hive-0.12.0/
- export PATH=$PATH:$HIVE_HOME/bin
- export SPARK_MASTER_PORT=7077

Start Hadoop and format namenode

- /hadoop-2.4.1/bin/hdfs namenode -format
- /hadoop-2.4.1/sbin/start-dfs.sh
- /hadoop-2.4.1/sbin/start-yarn.sh
- /hadoop-2.4.1/sbin/mr-jobhistory-daemon.sh start historyserver

Create HDFS directories

- hdfs dfs -mkdir -p /user/root
- hdfs dfs -mkdir -p /user/hive
- hdfs dfs -mkdir -p /user/hive/metastore
- hdfs dfs –mkdir /user/anonymous

Install PostgreSQL

- edit /etc/yum.repos.d/CentOS-Base.repo by adding "exclude=postgresql*" to the "[base]" and "[update]" sections
- $>wget -O http://yum.postgresql.org/9.3/redhat/rhel-6-x86_64/pgdg-centos93-9.3-1.noarch.rpm
- $>rpm -Uvh pgdg-centos93-9.3.1.noarch.rpm
- $>yum install postgresql93-server
- $>service postgresql-9.3 initdb
- $>chkconfig postgresql-9.3 on
- $>service postgresql-9.3 start

Install and configure Hive

- $>cd /usr/local
- $>wget http://apache.mirrors.hoobly.com/hive/hive-0.12.0/hive-0.12.0.tar.gz
- $>tar xvf hive-0.12.0.tar.gz
- Configure /usr/local/hive-0.12.0/conf/hive-site.xml to look something like this
    - *<configuration>*
    - *<property>*
    - *<name>javax.jdo.option.ConnectionURL</name>*
    - *<value>jdbc:postgresql://localhost/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>mypassword</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.warehouse.dir*</*name*>
- <*value*>*/user/hive/metastore*</*value*>
- </*property*>
- </*configuration*>

Configure PostgreSQL as Hive metastore

- Set "standard_conforming_strings" to off in /var/lib/pgsql/9.3/data/postgresql.conf
  - standard_conforming_strings = off listen_addresses = '*'
- Allow remote access by adding the following to /var/lib/pgsql/9.3/data/pg_hba.conf under the IPv6 section
  - host all all 0.0.0.0 0.0.0.0 password
- Restart service
  - service postgresql-9.3 restart
- Install PostgreSQL JDBC Driver
  - $>yum install postgresql-jdbc
  - $>ln -s /usr/share/java/postgresql-jdbc.jar /usr/local/hive/lib/postgresql-jdbc.jar
- Create metastore database and user account
  - $>su - postgres
  - $>psql
    - CREATE USER hiveuser WITH PASSWORD 'password';
    - CREATE DATABASE metastore;
    - \c metastore;
    - \i /usr/local/hive-0.12.0/scripts/metastore/upgrade/postgres/hive-schema-0.12.0.postgres.sql
    - \o /tmp/grant-privsSELECT 'GRANT SELECT,INSERT,UPDATE,DELETE ON '"|| schemaname ||'"."'|| tablename ||'" TO hiveuser;'FROM pg_tablesWHERE tableowner = CURRENT_USER and schemaname = 'public';\o
    - \i /tmp/grant-privs
- Verify connection with hive user
  - psql -h myhost -U hiveuser -d metastore
- Create softlink to hive-site.xml file
  - ln -s /hadoop-2.4.1/etc/hadoop/hive-site.xml /usr/local/hive-0.12.0/conf/hive-site.xml

To Shutdown Hadoop:

- $>/hadoop-2.4.1/sbin/mr-jobhistory-daemon.sh stop historyserver
- $>/hadoop-2.4.1/sbin/stop-yarn.sh
- $>/hadoop-2.4.1/sbin/stop-dfs.sh

To Start Hadoop:

- $>/hadoop-2.4.1/sbin/start-dfs.sh
- $>/hadoop-2.4.1/sbin/start-yarn.sh
- $>/hadoop-2.4.1/sbin/mr-jobhistory-daemon.sh start historyserver

Start metastore and hiveserver2 services

- $>mkdir /var/log/hive
- $>nohup hive --service metastore > /var/log/hive/metastore.log &
- $>nohup hive --service hiveserver2 > /var/log/hive/hiveserver2.log &

Spark SQL 1.1.x Install

- build spark from source with maven
    - $>cd /opt
    - $>wget http://mirrors.koehn.com/apache/maven/maven-3/3.2.3/binaries/apache-maven-3.2.3-bin.tar.gz
    - $>tar xvf apache-maven-3.2.3-bin.tar.gz
    - $>mv apache-maven-3.2.3 /opt/maven
    - $>ln -s /opt/maven/bin/mvn /usr/bin/mvn
- $>vim /etc/profile.d/maven.sh
- Add the following contents:
    - #!/bin/bash
    - MAVEN_HOME=/opt/maven
    - PATH=$MAVEN_HOME/bin:$PATH
    - export PATH MAVEN_HOME
    - export CLASSPATH=.

- Save and close the file. Make it executable using the following command.
    - $>chmod +x /etc/profile.d/maven.sh

- Then, set the environment variables permanently by running the following command:
    - $>source /etc/profile.d/maven.sh

- Get Spark source code
    - $>mkdir /usr/local/spark-1.1.x-bin-hadoop2.4

    - $>cd /usr/local/spark-1.1.x-bin-hadoop2.4
    - $>wget https://github.com/apache/spark/archive/master.zip
    - $>unzip master.zip
    - $>mv spark-master/* /usr/local/spark-1.1.x-bin-hadoop2.4/
    - $>cd /usr/local/spark-1.1.x-bin-hadoop2.4
    - $>export MAVEN_OPTS="-Xmx2g -XX:MaxPermSize=512M -XX:ReservedCodeCacheSize=512m"
    - $>mvn -Pyarn -Phadoop-2.4 -Dhadoop.version=2.4.0 -Phive -DskipTests clean package

- Wait for compiler to finish

Configure Spark:

- The following is optional as the default setting work just fine.
- edit /usr/local/spark-1.1.x-bin-hadoop2.4/conf/spark-env.sh add the following, which will vary depending on your environment.
- Add the following under the Yarn configurations
    - SPARK_EXECUTOR_CORES=4 #, Number of cores for the workers (Default: 1).
    - SPARK_EXECUTOR_MEMORY=4G #, Memory per Worker (e.g. 1000M, 2G) (Default: 1G)
    - SPARK_DRIVER_MEMORY=4G #, Memory for Master (e.g. 1000M, 2G) (Default: 512 Mb)

Starting Spark:

- To start spark master/worker and hive-thriftserver connector run the following
- $>/usr/local/spark-1.1.x-bin-hadoop2.4/sbin/start-master.sh
- $>/usr/local/spark-1.1.x-bin-hadoop2.4/sbin/start-slaves.sh
- $>/usr/local/spark-1.1.x-bin-hadoop2.4/sbin/start-thriftserver.sh --master spark://localhost:7077 --driver-class-path $CLASSPATH --hiveconf hive.server2.thrift.bind.host localhost --hiveconf hive.server2.thrift.port 10001