Source: The following hands-on tutorial is modified from http://holowczak.com/getting-started-with-apache-spark-on-aws/

Lab 3 – Apache Spark on Amazon EMR

In this lab, you will

- Create an AWS S3 storage bucket to hold data inputs, data outputs and logs
- Create, Configure and Launch an Amazon EMR Cluster
- Log in to the cluster master node
- Run Spark-SQL Command Line Interface and Issue commands to create tables and run queries
- Shut down cluster and remove any temporary resources

Part 1: Creating S3 Storage buckets to hold inputs, outputs and logs

1. Log in to Amazon Web Services. Pull down the Services menu and select S3 under the Storage & Content Delivery heading.
2. Click on the **Create Bucket** button. Select the region that matches your home region. You're your bucket, e.g., market-data-bucket. Click on the Create button to create the bucket.
3. Click on the Create button to create a new folder. Name the first folder "data" and click the check box to save it. Repeat the above step to create a folder named "output" and a third folder named "logs". When complete, the `market-data-bucket` will contain three folders data, output and logs.
4. Click the **Upload** button to upload a new file to the data folder. Click on the Green Plus sign next to Add Files. When the Open File dialog box appears, enter the following URL: `http://optionsdata.baruch.cuny.edu/data1/delivery/data/trades_sample.csv`
5. Click the **Open** button. The `trades_sample.csv` file will now be listed. Click on the **Start Upload** button to start uploading the file to S3 storage. When done the new `trades_sample.csv` file should appear in the `data` folder of the `market-data-bucket`.

Part 2: Creating and configuring an Amazon EMR Cluster

1. Pull down the **Services** menu and click on the **EMR** menu item. Click on the **Create Cluster** button.
2. Fill in the Cluster Name as `MySparkCluster`. Enable logging
3. Fill in the following parameters to configure the cluster:
    a. Choose Launch Mode: Cluster.
    b. Under the Software Configuration section select:
        i. Vendor: Amazon
        ii. Release: emr-5.4.0 (or whatever the latest version is)
        iii. Applications: Spark: Spark 2.1.0 with Ganglia 3.7.2 and Zeppelin 0.7.0 (or later versions)
    c. Under the Hardware Configuration select:
        i. Instance Type: m3.xlarge

ii. Number of Instances: 3

d. Under the Security and access section: Select your EC2 Key Pair.

e. Set the Permissions to **Default**
EMR Role: EMR_DefaultRole
EC2 Instance profile: EMR_EC2_DefaultRole

4. Once all of the settings are in place, click the **Create Cluster** button.

Part 3: Connecting to the Master Node using Secure Shell (SSH)

1. Open PUTTY, and Paste the Host Name (hadoop@..) of your cluster into the Putty client.
2. Navigate to the **Connection**, **SSH** and **Auth** item. Under the heading for Private Key file for authentication, click the Browse button and select the `.ppk` file prepared previously. Finally, click the **Open** button to connect to the master node (see below if the SSH connection times out).
3. At this point the EMR shell will appear.
4. Check to make sure the sample file is loaded and accessible on your S3 bucket storage by issuing an AWS command: `aws s3 ls s3://market-data-bucket/data/`
5. Note: Depending on the default security configuration of the EMR Cluster, it may be the case that remote SSH has been blocked or disabled. If the SSH connection times out, check the Security Group for the master node and if necessary, add an appropriate rule to enable remote SSH logins to the master node.

Part 4: Loading Additional Data to AWS S3 from the Master Node

1. The sample data file trades_sample.csv was previously loaded on the Amazon S3 storage in folder market-data-bucket/data. If more data needs to be loaded into Amazon S3 storage, this can be accomplished at this time from the EMR Master Node. For example, use the curl command to download a file from the web to the Master Node file system. Then use an aws command to copy or move the file to the S3 storage. For example to download the trades_sample.csv file again:
```
[hadoop@ip-172-31-4-0 ~]$ curl -O
http://optionsdata.baruch.cuny.edu/data1/delivery/data/trades_sample.csv
```
2. Then, to copy this file to AWS S3 storage use this aws s3 cp command (substitute the name of your bucket):
```
aws s3 cp trades_sample.csv  s3://market-data-bucket/data/
```

Part 5: Using the Spark-SQL Command Line Interface

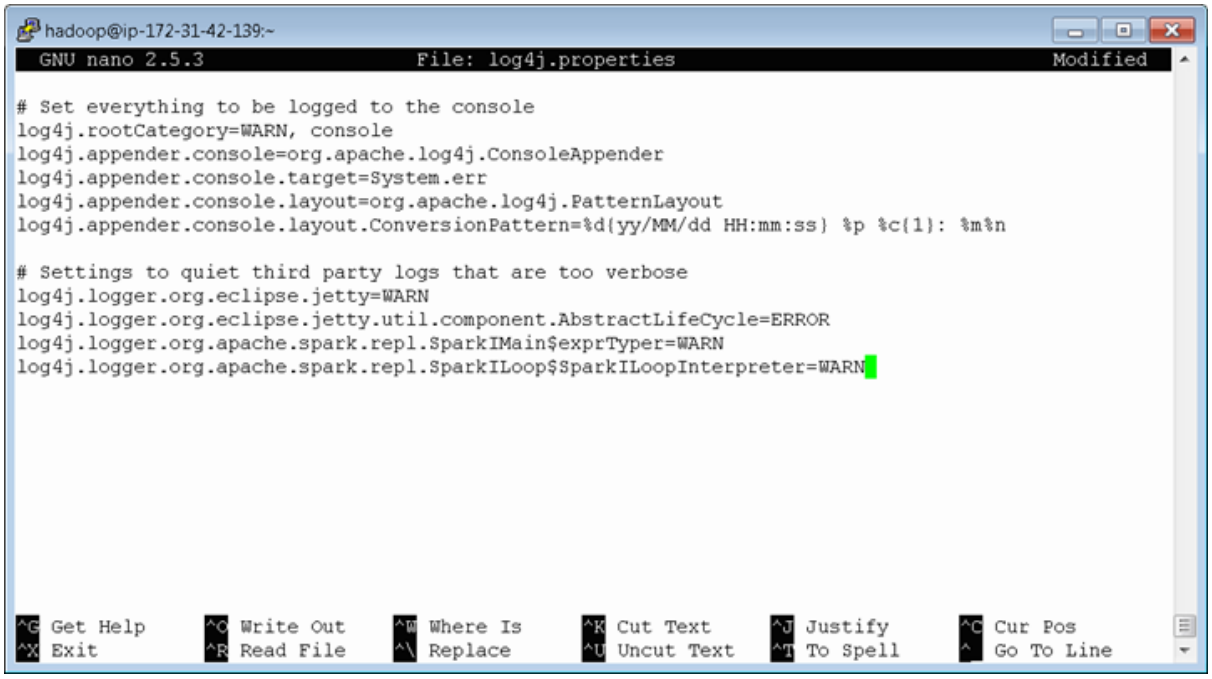
1. Make a configuration file named log4j.properties using the nano text editor:
```
[hadoop@ip-172-31-42-139 ~]$ nano log4j.properties
```
2. Paste in the following lines:

```
# Set everything to be logged to the console
log4j.rootCategory=WARN, console
```

```
log4j.appender.console=org.apache.log4j.ConsoleAppender
log4j.appender.console.target=System.err
log4j.appender.console.layout=org.apache.log4j.PatternLayout
log4j.appender.console.layout.ConversionPattern=%d{yy/MM/dd
HH:mm:ss} %p %c{1}: %m%n

# Settings to quiet third party logs that are too verbose
log4j.logger.org.eclipse.jetty=WARN
log4j.logger.org.eclipse.jetty.util.component.AbstractLifeCycl
e=ERROR
log4j.logger.org.apache.spark.repl.SparkIMain$exprTyper=WARN
log4j.logger.org.apache.spark.repl.SparkILoop$SparkILoopInterp
reter=WARN
```

3. At this point the file will look like the figure below:



4. Press ^X (hold down the Control key and press X) to exit the nano editor. The following prompt will appear:

```
Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES)?

 Y Yes
 N No           ^C Cancel
```

Press Y to save the file. A new prompt will appear for the file name:

```
File Name to Write: log4j.properties
^G Get Help        M-D DOS Format     M-A Append         M-B
Backup File
^C Cancel          M-M Mac Format     M-P Prepend        ^T To
Files
```

Press Enter and the file name given will be used.

Part 6. Launch Spark-SQL

1. Launch spark-sql and use the newly created configuration file to suppress INFO logging messages.

```
spark-sql --driver-java-options "-
Dlog4j.configuration=file:///home/hadoop/log4j.properties
```

2. Create an external table named trades_sample by pointing to the S3 bucket and folder.

```
CREATE EXTERNAL TABLE trades_sample
        (trading_date_time TIMESTAMP,
         network CHAR(1),
         message_category CHAR(1),
         message_type CHAR(1),
         message_sequence BIGINT,
         market_exchange CHAR(1),
         symbol VARCHAR(10),
         trade_price DOUBLE,
         trade_size BIGINT,
         trade_conditions VARCHAR(6),
         trade_conditions2 VARCHAR(6) )
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION 's3://market-data-bucket/data/';
```

3. Examine the structure of the table using the `DESCRIBE` command: `DESCRIBE trades_sample;`
4. Run an example query to see how many records there are: `SELECT COUNT(*) FROM trades_sample;`
5. Once done, be sure to shut down the EMR Cluster and remove any data files from S3 that are no longer needed.