**Exercise 6**

*More Apache Spark and Python*

**Prior Knowledge**

Unix Command Line Shell

Simple Python

**Learning Objectives**

Using Spark on EC2

Accessing S3 files on Spark

Reading CSV files in Spark

Seeing the differences between Spark and Hadoop by performing the Wind Analysis in Spark

Spark SQL

Spark statistics

**Software Requirements**

(see separate document for installation of these)

* Apache Spark 1.5.1
* Python 2.7.x
* Nano text editor or other text editor

**Part A. Starting Spark in EC2**

1. Do you remember the Access Key and Secret Key from Exercise 1? You need those now.
2. In a terminal window type:  
   export AWS\_ACCESS\_KEY\_ID=<your access key here>  
   export AWS\_SECRET\_ACCESS\_KEY=<your secret key here>
3. Now change into the Spark EC2 directory:  
   cd ~/spark-1.5.1/ec2
4. Now let’s launch a Spark cluster in EC2. Replace XX with your user details so these match the locations of your key files and so you can identify your own spark cluster  
     
   ./spark-ec2 --key-pair=oxclo*XX* \  
   --identity-file=/home/oxclo/oxclo*XX*.pem \  
   --region=eu-west-1 \  
   -s 1 \  
   launch oxcloXX-spark-cluster

The –s 1 indicates that there is just one slave (you could launch more but that might be expensive).

You should see output like:

Setting up security groups...

Searching for existing cluster my-spark-cluster in region eu-west-1...

Spark AMI: ami-1ae0166d

Launching instances...

Launched 1 slave in eu-west-1a, regid = r-52c4f5ff

Launched master in eu-west-1a, regid = r-c2c7f66f

Waiting for AWS to propagate instance metadata...

Waiting for cluster to enter 'ssh-ready' state..........

Warning: SSH connection error. (This could be temporary.)

Host: ec2-52-16-96-164.eu-west-1.compute.amazonaws.com

SSH return code: 255

SSH output: ssh: connect to host ec2-52-16-96-164.eu-west-1.compute.amazonaws.com port 22: Connection refused

.

1. Maybe go grab a coffee ☺ This takes a while
2. After a while the system will start logging a lot more as the setup on EC2 starts happening.  
   Eventually you will see:

Setting up ganglia

RSYNC'ing /etc/ganglia to slaves...

ec2-52-31-197-16.eu-west-1.compute.amazonaws.com

Shutting down GANGLIA gmond: [FAILED]

Starting GANGLIA gmond: [ OK ]

Shutting down GANGLIA gmond: [FAILED]

Starting GANGLIA gmond: [ OK ]

Connection to ec2-52-31-197-16.eu-west-1.compute.amazonaws.com closed.

Shutting down GANGLIA gmetad: [FAILED]

Starting GANGLIA gmetad: [ OK ]

Stopping httpd: [FAILED]

Starting httpd: httpd: Syntax error on line 154 of /etc/httpd/conf/httpd.conf: Cannot load /etc/httpd/modules/mod\_authz\_core.so into server: /etc/httpd/modules/mod\_authz\_core.so: cannot open shared object file: No such file or directory

[FAILED]

[timing] ganglia setup: 00h 00m 02s

Connection to ec2-52-16-96-164.eu-west-1.compute.amazonaws.com closed.

Spark standalone cluster started at http://ec2-52-16-96-164.eu-west-1.compute.amazonaws.com:8080

Ganglia started at http://ec2-52-16-96-164.eu-west-1.compute.amazonaws.com:5080/ganglia

Done!

1. It seems in this distribution Ganglia is not working, but let’s carry on regardless!
2. Find the Spark URL in the log above and go to that page in your browser. You might want to leave this open as we’ll need it later.
3. Let’s login to the master:  
     
   ./spark-ec2 -k oxcloXX -i /home/oxclo/oxcloXX.pem \  
   --region eu-west-1 \   
   login oxcloXX-spark-cluster   
     
   You see:

Last login: Wed Nov 18 09:27:01 2015 from ip-172-31-9-125.eu-west-1.compute.internal

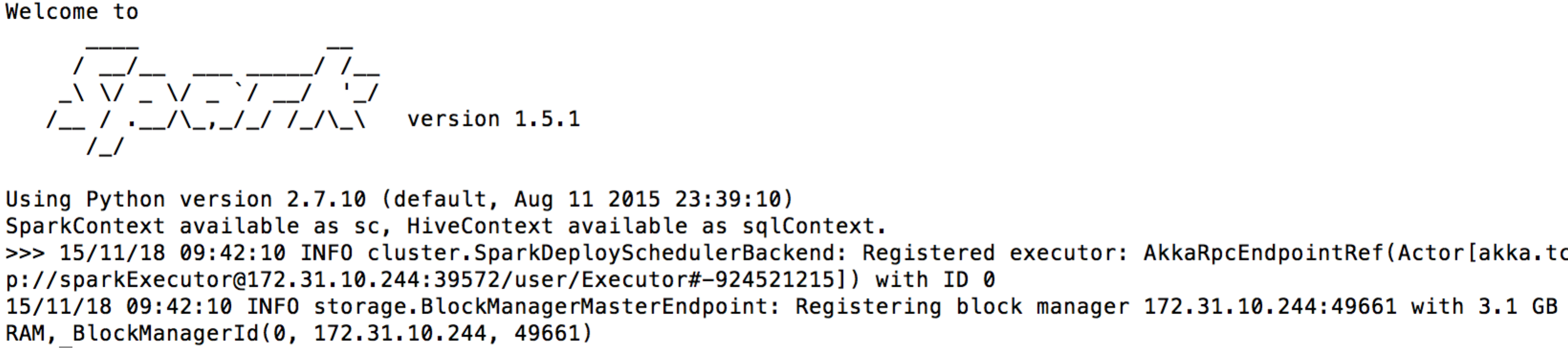
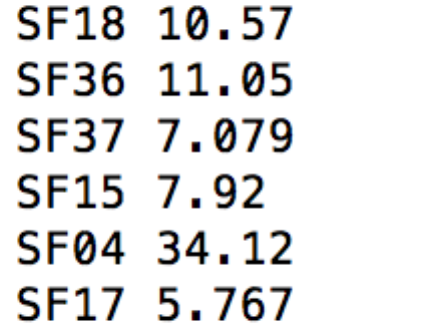
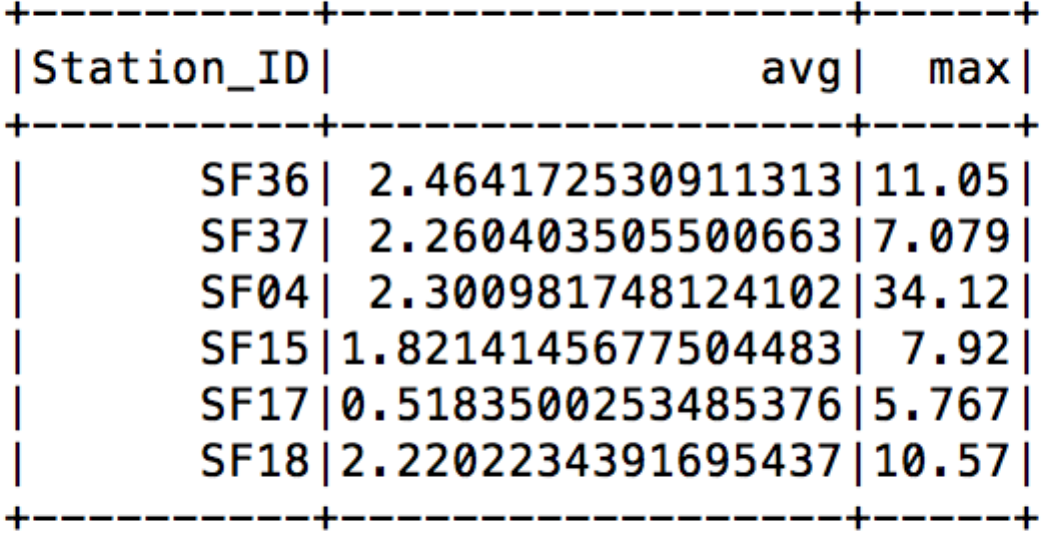
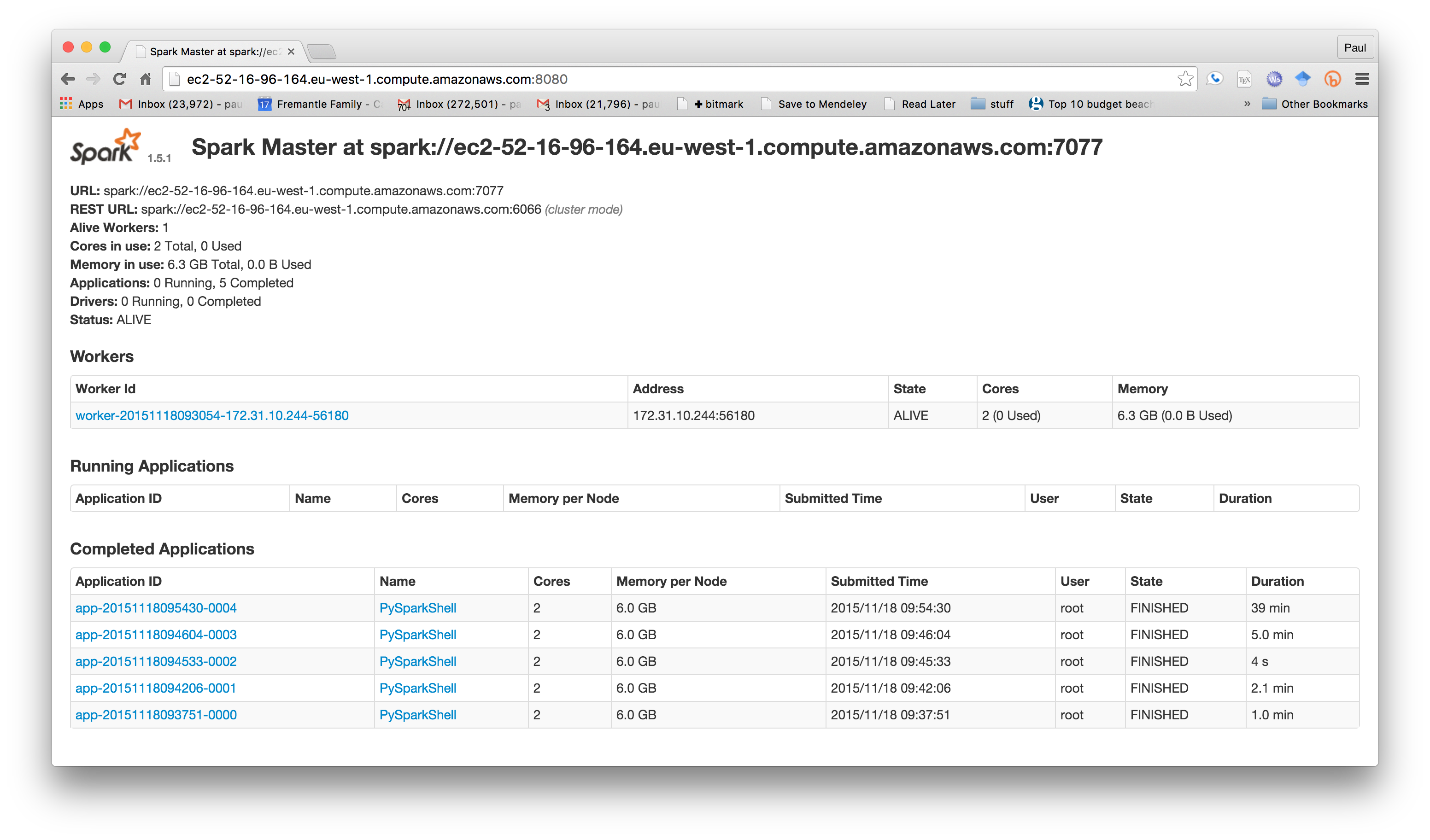
\_\_| \_\_|\_ )

\_| ( / Amazon Linux AMI

\_\_\_|\\_\_\_|\_\_\_|

https://aws.amazon.com/amazon-linux-ami/2013.03-release-notes/

Amazon Linux version 2015.09 is available.

1. This basically just SSH’s you into the master. You could do the same from the EC2 console as before.
2. We are going to need your AWS credentials in this session as well so that spark can access S3 resources. In this session type   
   export AWS\_ACCESS\_KEY\_ID=<your access key here>  
   export AWS\_SECRET\_ACCESS\_KEY=<your secret key here>
3. Type  
   cd spark
4. Now start pyspark once again. This time we are going to add in a Spark Package that supports easy reading of CSV files:  
     
   bin/pyspark \  
    --packages com.databricks:spark-csv\_2.11:1.2.0  
     
   You should see:  
   
5. We are going to use Spark’s SQL support which in turn uses Apache Hive. This combined with the CSV package we saw earlier makes it very easy to work with data.   
   First let’s tell spark we are using SQL:  
   from pyspark.sql import SQLContext  
   sqlContext = SQLContext(sc)
6. Now let’s load the data into a DataFrame.  
     
   df = sqlContext.\  
   read.format('com.databricks.spark.csv').\  
   options(header='true', inferschema='true').\  
   load('s3n://oxclo-wind/2015/\*')
7. You should see a lot of log go by.
8. The df object we have is not an RDD. Its basically a SQL construct. But we can easily convert it into an RDD.  
     
   winds = df.rdd
9. You can see the structure of each row by:  
     
   winds.first()
10. Let’s do the normal step of mapping the data into a simple <K,V> pair. Each column in the row can be accessed by the syntax e.g. row.Station\_ID  
      
    We can therefore map our RDD with the following:  
      
    mapped = \  
    winds.map(lambda s: (s.Station\_ID, \  
    s.Wind\_Velocity\_Mtr\_Sec))
11. We can simply calculate the maximum values with this reducer:  
      
    maxes = mapped.reduceByKey(lambda a, b: a if (a>b) else b)
12. And once again collect / print:  
      
    for (k,v) in maxes.collect(): print k,v
13. You will see a bunch of log before the following appears:  
      
    **PART B – Using SQL**
14. There is an easier way to do all this if you are willing to write some SQL.
15. First we need to give our DataFrame a table name:  
    df.registerTempTable('wind')
16. Now we can use a simple SQL statement against our data.   
    ALL ON ONE Line type:  
      
    sqlContext.sql("SELECT Station\_ID, avg(Wind\_Velocity\_Mtr\_Sec) as avg,max(Wind\_Velocity\_Mtr\_Sec) as max from wind group by Station\_ID").show()
17. Bingo you should see a lot of log followed by:  
    
18. Recap. We have:
    1. Started Spark in EC2
    2. Loaded data from S3
    3. Used SQL to read in CSV files
    4. Explored Map/Reduce on those CSV files
    5. Used SQL to query the data.
19. Go back to the browser view of the Spark console and you can take a look at the jobs that have been run:  
    
20. Quit the pyspark shell:   
    quit()
21. Exit the SSH session:  
    exit
22. We must remember to stop our cluster as well (its costing money!)  
    From Ubuntu terminal where you started the Spark cluster  
      
    ./ec2-spark –region eu-west-1 destroy oxcloXX-spark-cluster
23. Congratulations!