

# Exercise 4

*Spark SQL on AWS*

## Prior Knowledge

Unix Command Line Shell

Simple Python

## Learning Objectives

Understanding how to run Spark on AWS using Flintrock

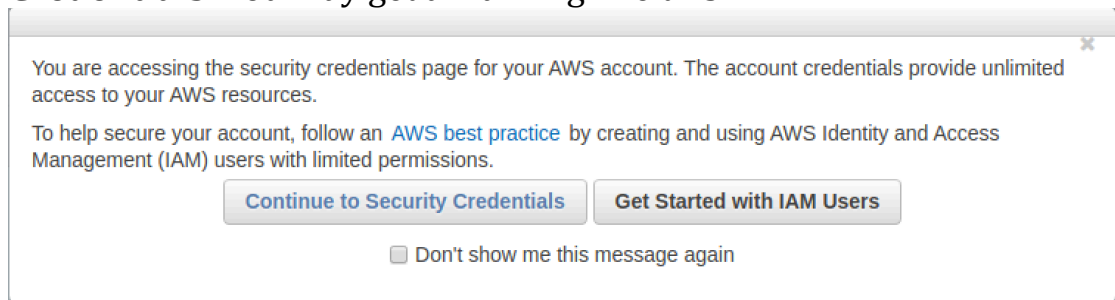
## Software Requirements

(see separate document for installation of these)

- EC2 credentials
- Flintrock

## Part A. Starting Spark in EC2

1. Log into your AWS account on the browser, and go to **Security Credentials**. You may get a warning like this:



- Expand the section labeled **Access Keys**  
(I have blanked out key details in the screen shot)

**Your Security Credentials**

Use this page to manage the credentials for your AWS account. To manage credentials for AWS Identity and Access Management (IAM) users, use the [IAM Console](#).  
To learn more about the types of AWS credentials and how they're used, see [AWS Security Credentials](#) in AWS General Reference.

- + Password
- + Multi-factor authentication (MFA)
- Access keys (access key ID and secret access key)

You use access keys to sign programmatic requests to AWS services. To learn how to sign requests using your access keys, see the [signing documentation](#). For your protection, store your access keys securely and do not share them. In addition, AWS recommends that you rotate your access keys every 90 days.  
Note: You can have a maximum of two access keys (active or inactive) at a time.

Created	Deleted	Access Key ID	Last Used	Last Used Region	Last Used Service	Status	Actions
							Make Inactive   Delete

[Create New Access Key](#)

**Important Change - Managing Your AWS Secret Access Keys**  
As described in a [previous announcement](#), you cannot retrieve the existing secret access keys for your AWS root account, though you can still create a new root access key at any time. As a [best practice](#), we recommend [creating an IAM user](#) that has access keys rather than relying on root access keys.

- Click **Create New Access Key**

**Create Access Key**

✓ **Your access key (access key ID and secret access key) has been created successfully.**  
Download your key file now, which contains your new access key ID and secret access key. If you do not download the key file now, you will not be able to retrieve your secret access key again.  
To help protect your security, store your secret access key securely and do not share it.  
[► Show Access Key](#)

[Download Key File](#) [Close](#)

- Download the key file**
- Display the keyfile (e.g. edit it with Atom)
- On a command line type:  
`aws configure`  
  
You should see:  
AWS Access Key ID [\*\*\*\*\*J3EA]:
- Copy the Access Key ID from the keyfile, and then hit Enter
- Do the same for the Secret Access Key
- Set the default region to **eu-west-1**
- Set the output to **json**

11. It should like this (but with your keys):

```
AWS Access Key ID [*****J3EA]: AKIASF22343434UNM33UVIA
Secret Access Key [*****JXb7]: 8z1rtTbU3Ur/llksdafkjhd398u34msndHnGaDY
Default region name [eu-west-1]: eu-west-1
Default output format [json]: json
```

12. These keys allow applications to interact with Amazon AWS on your behalf.

13. In addition to these “Access Keys”, we also need an SSH key to continue. If you successfully completed the pre-course Amazon lab, you should have a file `~/keys/bigkp.pem`. If not, grab one of the instructors to help you create one.

14. There is a project from the creators of Spark to run it in EC2, but it is not very good! Instead we will use a tool called **flintrock**, which can configure Spark clusters in AWS for you.

15. Before we can use flintrock, you need to modify the config file for flintrock so that it uses your own keys. Edit the flintrock config file:

```
atom ~/.config/flintrock/config.yaml
```

It will look something like:

```
1 services:
2   spark:
3     version: 2.2.0
4     # git-commit: latest # if not 'latest', provide a full commit SHA; e.g. d6dc12ef0146ae409834c78737d
5     # git-repository: # optional; defaults to https://github.com/apache/spark
6     # optional; defaults to download from from the official Spark S3 bucket
7     # - must contain a {v} template corresponding to the version
8     # - Spark must be pre-built
9     # - must be a tar.gz file
10    # download-source: "https://www.example.com/files/spark/{v}/spark-{v}.tar.gz"
11    # executor-instances: 1
12  hdfs:
13    version: 2.7.3
14    # optional; defaults to download from a dynamically selected Apache mirror
15    # - must contain a {v} template corresponding to the version
16    # - must be a .tar.gz file
17    # download-source: "https://www.example.com/files/hadoop/{v}/hadoop-{v}.tar.gz"
18    # download-source: "http://www-us.apache.org/dist/hadoop/common/hadoop-{v}/hadoop-{v}.tar.gz"
19
20  provider: ec2
21
22  providers:
23    ec2:
24      key-name: key_name
25      identity-file: /path/to/key.pem
26      instance-type: m3.medium
27      region: us-east-1
```

We need to replace the contents of this file, with one that will work for us.

The source for this is here: <https://freo.me/big-flintrock>

It should look like:

This is modified in a couple of ways. Firstly, it gives the Ireland region and AMI files. Secondly, there is an “instance-profile-name”. This is a AWS feature that gives the running VM access to other APIs - in this case S3.

Finally, I’ve changed the key name and identity file to match your key name and identity file.

16. Copy and paste from the web version to your local version, replacing the existing text.

17. Save the file

18. You should now be able to launch a cluster in Amazon. From a new terminal window

```
flintrock launch big
```

19. Now you should see something like (except with more lines):

```
Launching 3 instances...
[54.154.17.100] SSH online.
[54.154.17.100] Configuring ephemeral storage...
[54.154.17.100] Installing Java 1.8...
[34.253.201.139] SSH online.
[34.253.201.139] Configuring ephemeral storage...
[34.253.201.139] Installing Java 1.8...
[54.154.17.100] Installing Spark...
[34.253.201.139] Installing Spark...
[34.253.201.139] Configuring Spark master...
Spark Health Report:
  * Master: ALIVE
  * Workers: 1
  * Cores: 1
  * Memory: 2.7 GB
launch finished in 0:03:49.
```

20. In the meantime, you could start yet another terminal window and prepare your code to run on AWS.

```
cd sql
cp wind.py wind-s3.py
```

21. Change the URL so that instead of loading the data from the local filesystem, it reaches out to S3 to do it:

```
atom wind-s3.py
```

Instead of reading from `‘/home/big/sql/*.csv’` change it to read from:

```
's3a://oxclo-wind/2015/*'
```

22. Delete the first two lines (`import findspark` and `findspark.init()`).

23. Save the file

24. Once the launch of your cluster has finished, we need to copy the code into the cluster:

```
flintrock copy-file big wind-s3.py /home/ec2-user/wind-s3.py
```

25. Let's login to the master (all one line):

```
flintrock login big
```

You see something like:

```
Warning: Permanently added '34.253.201.139' (ECDSA) to the list
of known hosts.
Last login: Mon Jul 10 18:55:35 2017 from host109-156-251-
208.range109-156.btcentralplus.com
```

```
  _ | _ | _ )
 _ | ( _ | /  Amazon Linux AMI
 _ | \ _ | _ |
```

```
https://aws.amazon.com/amazon-linux-ami/2017.03-release-notes/
1 package(s) needed for security, out of 1 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-6-32 ~]$
```

26. This basically just SSH's you into the master.

27. Now launch your code:

```
~/spark/bin/spark-submit wind-s3.py
```

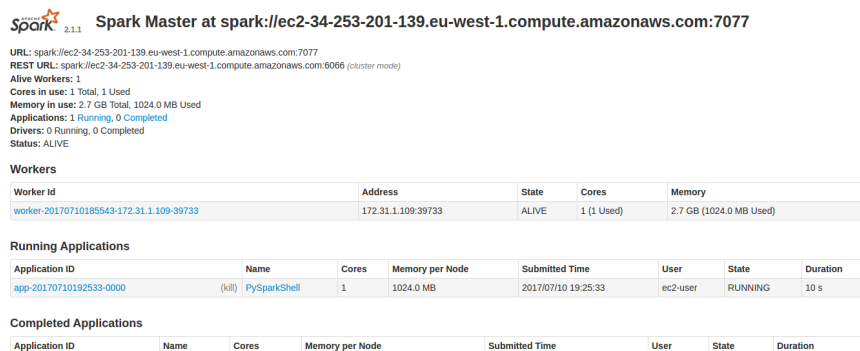
You should see a lot of logging, eventually ending with:

```
17/12/08 13:59:18 INFO TaskSetManager: Finished task 32.0 in stage 10.0 (TID 201) in 19 ms on local
host (executor driver) (75/75)
17/12/08 13:59:18 INFO TaskSchedulerImpl: Removed TaskSet 10.0, whose tasks have all completed, fro
n pool
17/12/08 13:59:18 INFO DAGScheduler: ResultStage 10 (showString at NativeMethodAccessorImpl.java:0)
finished in 2.497 s
17/12/08 13:59:18 INFO DAGScheduler: Job 5 finished: showString at NativeMethodAccessorImpl.java:0,
took 2.598755 s
17/12/08 13:59:18 INFO CodeGenerator: Code generated in 98.36736 ms
+-----+
|Station_ID|          avg|      max|
+-----+
|      SF37| 2.260403505500663| 7.079|
|      SF15| 1.8214145677504483| 7.92|
|      SF04| 2.300981748124102| 9.92|
|      SF17| 0.5183500253485376| 5.767|
|      SF18| 2.2202234391695437| 9.85|
|      SF36| 2.464172530911313| 9.71|
+-----+
17/12/08 13:59:18 INFO SparkContext: Invoking stop() from shutdown hook
17/12/08 13:59:18 INFO SparkUI: Stopped Spark web UI at http://ec2-54-171-162-131.eu-west-1.compute
.amazonaws.com:4040
17/12/08 13:59:18 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
17/12/08 13:59:18 INFO MemoryStore: MemoryStore cleared
17/12/08 13:59:18 INFO BlockManager: BlockManager stopped
17/12/08 13:59:18 INFO BlockManagerMaster: BlockManagerMaster stopped
17/12/08 13:59:18 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordin
ator stopped!
17/12/08 13:59:18 INFO SparkContext: Successfully stopped SparkContext
17/12/08 13:59:18 INFO ShutdownHookManager: Shutdown hook called
17/12/08 13:59:18 INFO ShutdownHookManager: Deleting directory /media/ephemeral0/spark/spark-64f957
99-d0df-44ce-a2b7-553595f72c89/pyspark-87d51d42-afda-4a62-99d5-9615d9c308eb
17/12/08 13:59:18 INFO ShutdownHookManager: Deleting directory /media/ephemeral0/spark/spark-64f957
99-d0df-44ce-a2b7-553595f72c89
[ec2-user@ip-172-31-4-254 ~]$ $
```

28. In this case, it actually took us longer to run on the cluster than on our local machines. However, you will note that this would speed up for bigger problems where the parallelization would add benefits.
29. It is perfectly possible to get Jupyter to talk to Spark on our cluster, but it is slightly complex, so we will just use the normal Python command-line for the moment.
30. Find the IP address of the Spark Master. There are two ways. Firstly, it showed up in the console when you first launched the flintrock cluster: [34.253.201.139] Configuring Spark master...

Alternatively, you can find it by typing  
`flintrock describe big`

31. Go to <http://xx.xx.xx.xx:8080> using the master's IP address. You should see something like:



The screenshot shows the Spark Master web interface for a cluster named 'big'. The URL is `spark://ec2-34-253-201-139.eu-west-1.compute.amazonaws.com:7077`. The interface displays the following information:

- URL:** `spark://ec2-34-253-201-139.eu-west-1.compute.amazonaws.com:7077`
- REST URL:** `spark://ec2-34-253-201-139.eu-west-1.compute.amazonaws.com:6066 (cluster mode)`
- Alive Workers:** 1
- Cores in use:** 1 Total, 1 Used
- Memory in use:** 2.7 GB Total, 1024.0 MB Used
- Applications:** 1 Running, 0 Completed
- Drivers:** 0 Running, 0 Completed
- Status:** ALIVE

**Workers**

Worker id	Address	State	Cores	Memory
worker-20170710185543-172.31.1.109-39733	172.31.1.109:39733	ALIVE	1 (1 Used)	2.7 GB (1024.0 MB Used)

**Running Applications**

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
app-20170710192533-0000	(kill) PySparkShell	1	1024.0 MB	2017/07/10 19:25:33	ec2-user	RUNNING	10 s

**Completed Applications**

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
----------------	------	-------	-----------------	----------------	------	-------	----------

32. You can explore your Spark cluster here.
33. Exit the SSH session:  
`exit`
34. We must remember to stop our cluster as well (its costing money...)  
 From Ubuntu terminal where you started the Spark cluster

`flintrock destroy big`  
 Type y when prompted.

35. Congratulations, this lab is complete.