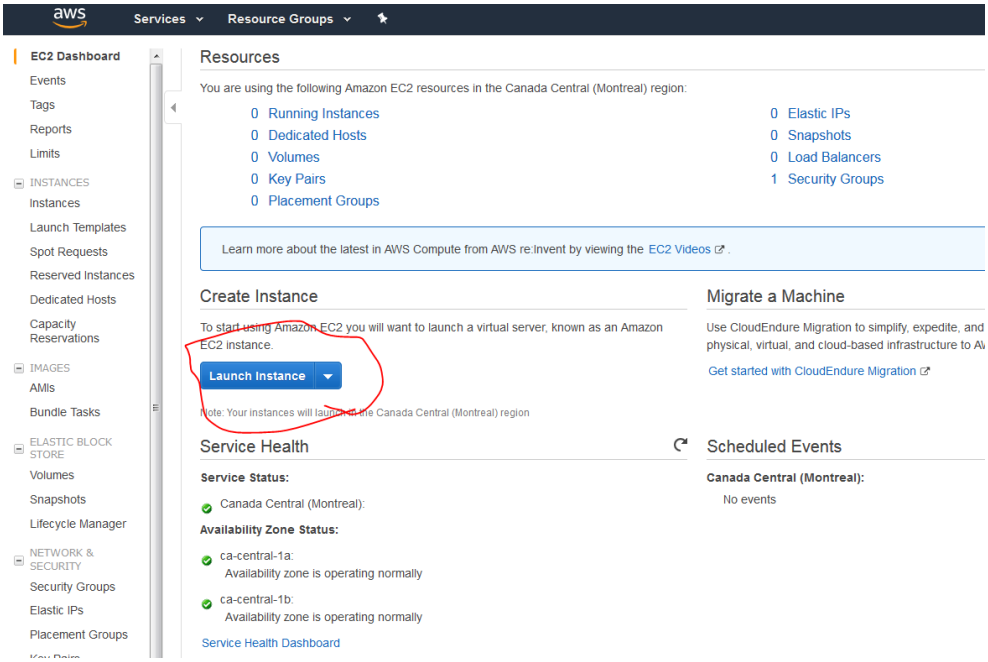
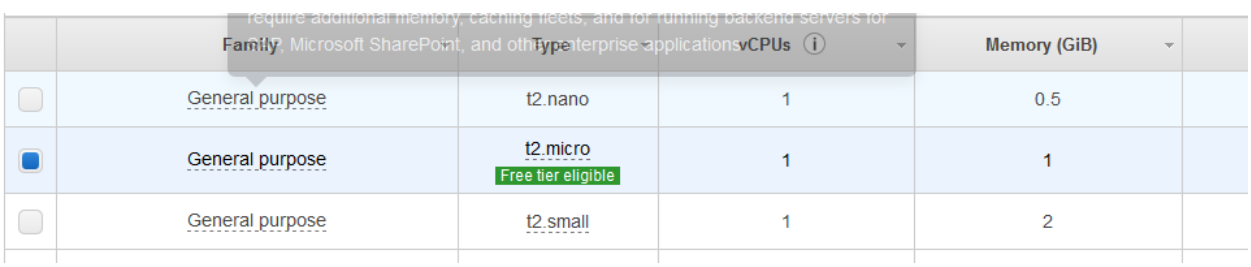
First you will need to create an account on AWS Console, please visit https://aws.amazon.com/console/ and create your own account. Then login to your account, and choose EC2 service, you can also search EC2 on the search box. After selecting the EC2 service, please select Launch Instance button:



Then choose Ubuntu (eligible for free tier). Then choose the general purpose micro that is eligible for free tier (You can choose instances with multiple CPUs but they are not free):

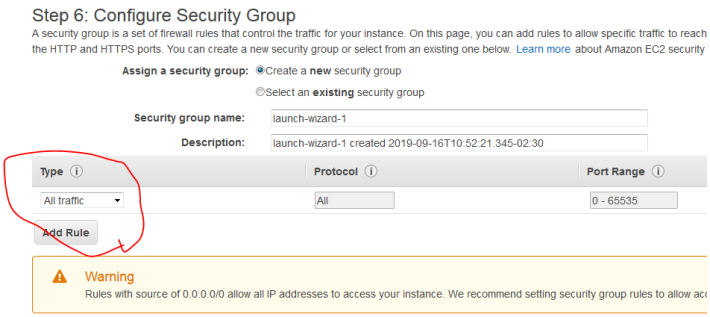


	Family	Type	vCPUs	Memory (GiB)
	General purpose	t2.nano	1	0.5
	General purpose	t2.micro Free tier eligible	1	1
	General purpose	t2.small	1	2

Press Next: Configure Instance Details button. Here you can change the number of cores (instances) but then if you have more than 1 core, you will be charged, so we stay with one core and press Next: Add Storage. Here you can change the storage but we are going with the default 8 GB.

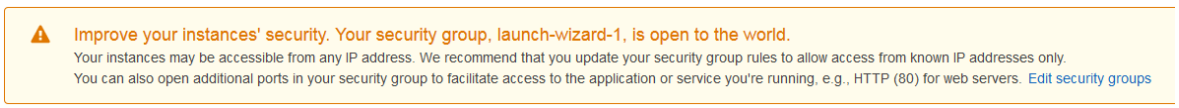
Press Next: Add Tags. Then choose a name and a value for your instance, for example : myspark and mymachine.

Press Next: Configure Security Group and from Type you can select All traffic:



Then click on Review and Launch.

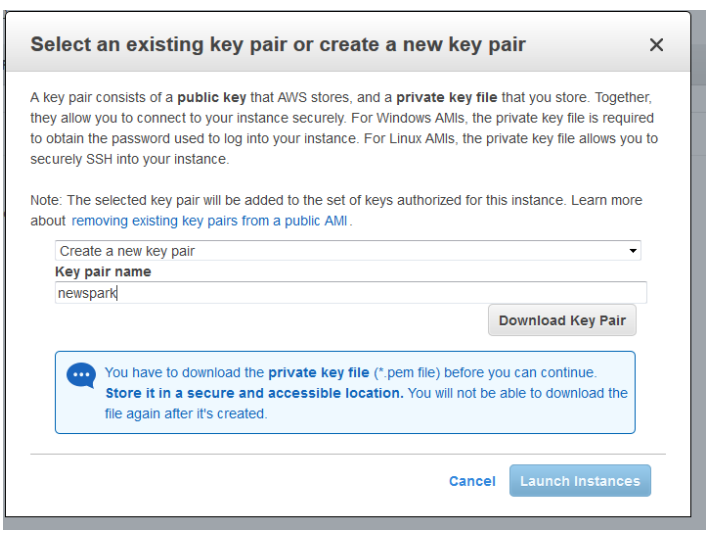
Here it gives you a warning:



That means that your instance is not secured but because this is for just a course that would be fine but if you have sensitive data, you should make it more secure.

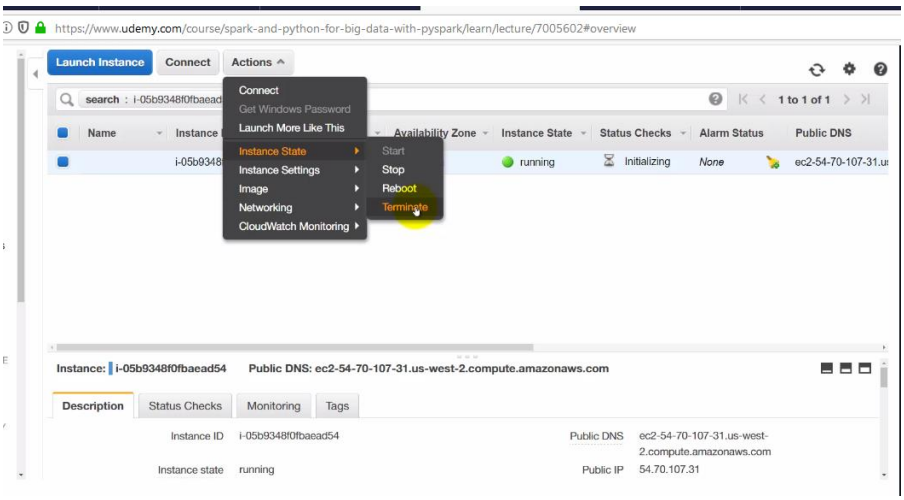
Review and if everything is fine then click Launch.

Then a window appears asking you for a key pair, choose “Create a new key pair” then choose a name for it and click on Download Key Pair:



Then save the downloaded .pem file, once you have downloaded it you are ready to use your instance. Please make sure you don’t miss this part (downloading the keys) otherwise you’ll have to start from scratch again.

Press Launch Instance. Then press Instance Then you will see your instance. You should always keep in mind to terminate the service after you are finished:



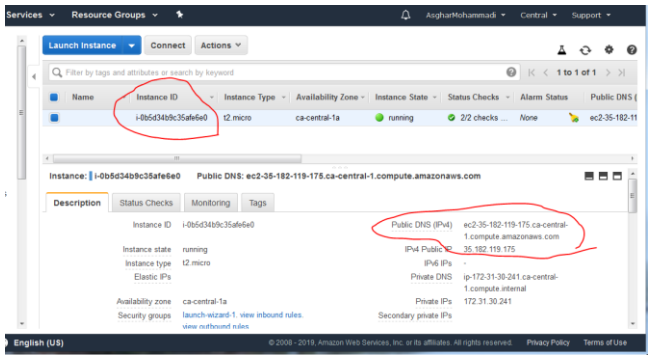
With the free tier you have 750 hours/ month for 12 month.

How to access your instance from a Windows Machine:

Google this terms “ssh windows ec2” then choose the first link from AWS that has instruction about “Connecting to Your Linux Instance from Windows Using PuTTY“.

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/putty.html

1) Install the latest versions of both PuTTY and PuTTYgen
2) You will need these information from AWS:
   The ID of the instance, Public DNS, and the .pem file that you downloaded.

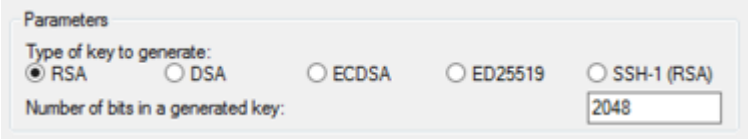


3) Convert Your Private Key Using PuTTYgen

To convert your private key

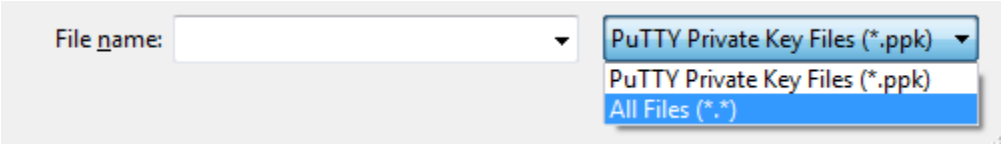
1. From the **Start** menu, choose **All Programs**, PuTTY, PuTTYgen.

2. Under **Type of key to generate**, choose **RSA**.



   If you're using an older version of PuTTYgen, choose **SSH-2 RSA**.

3. Choose **Load**. By default, PuTTYgen displays only files with the extension `.ppk`. To locate your `.pem` file, select the option to display files of all types.



4. Select your `.pem` file for the key pair that you specified when you launched your instance and choose **Open**. Choose **OK**.
5. To save the key in the format that PuTTY can use, choose **Save private key**. PuTTYgen displays a warning about saving the key without a passphrase. Choose **Yes**.

   Note

   A passphrase on a private key is an extra layer of protection. Even if your private key is discovered, it can't be used without the passphrase. The downside to using a passphrase is that it makes automation harder because human intervention is needed to log on to an instance, or to copy files to an instance.

6. Specify the same name for the key that you used for the key pair (for example, `my-key-pair`). PuTTY automatically adds the `.ppk` file extension.

Your private key is now in the correct format for use with PuTTY. You can now connect to your instance using PuTTY's SSH client.

- (I saved the .ppk file in the directory C:\Asghar\Other\Udemy\Spark and Python for Big Data\AWS Key Gen and the name of the file is AWS_Key_Pair.ppk)

Then start PuTTY and in the host name field put the public DNS that you found in your instance properties in AWS and you should add username@ in front of the host name. For the Ubuntu instances, you should put **ubuntu@**yourhostname.

Leave the port as default 22, and the Connection type as SSH.

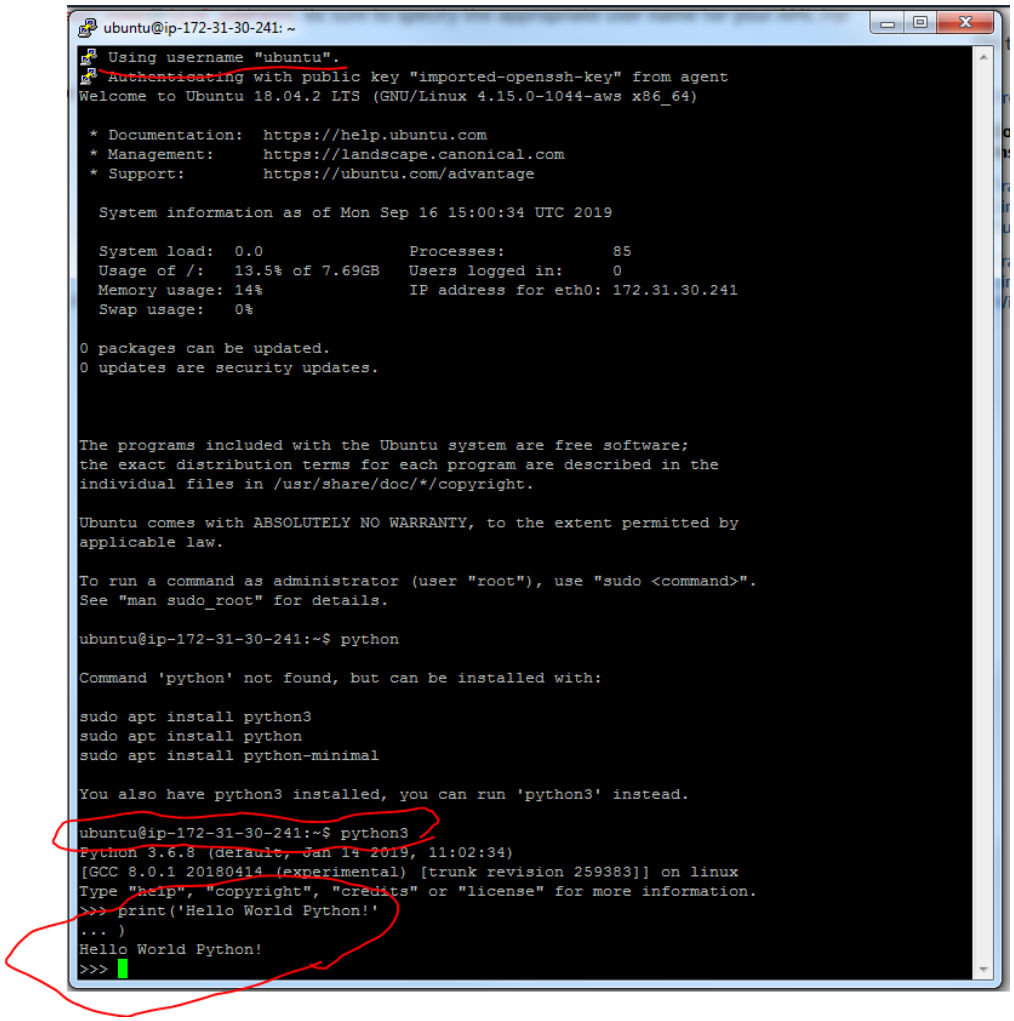
In the **Category** pane, expand **Connection**, expand **SSH**, and then choose **Auth**. Complete the following:

Choose **Browse**. spar

Select the `.ppk` file that you generated for your key pair and choose **Open**.

Choose **Open**.

When you click open, a command line that you can run commands. If you type Python3, you will have python running:



On this instance of EC2, we will

- Download Spark and then install it
- Install Jupyter Notebook

- Setting up PySpark and connecting to it
- SSH Tunneling from our local machine to AWS
- Running Jupyter Notebook on our local browser

1) Try $ **sudo apt-get update**
2) $ **sudo apt install python3-pip** will install pip for python3 so that installing Python libraries will be easier.
3) $ **pip3 install jupyter** for installing Jupyter Notebook
4) $ **sudo apt-get install default-jre** for installing Java in case you don't have it already. (Spark needs Scala and Scala needs Java).
5) $ **java –version** to confirm java is installed
6) $ **sudo apt-get install scala**
7) **$ scala –version**
8) **$ pip3 install py4j**
9) **$ wget** http://archive.apache.org/dist/spark/spark-2.1.1/spark-2.1.1-bin-hadoop2.7.tgz (this will download the .tgz file for installation)
10) $ **sudo tar -zxvf spark-2.1.1-bin-hadoop2.7.tgz** for unzipping the file and installing Spark and Hadoop folder is added to the directory.
11) $ **cd spark-2.1.1-bin-hadoop2.7/** to go into the directory
12) $ **pwd** to print working directory. you should see:

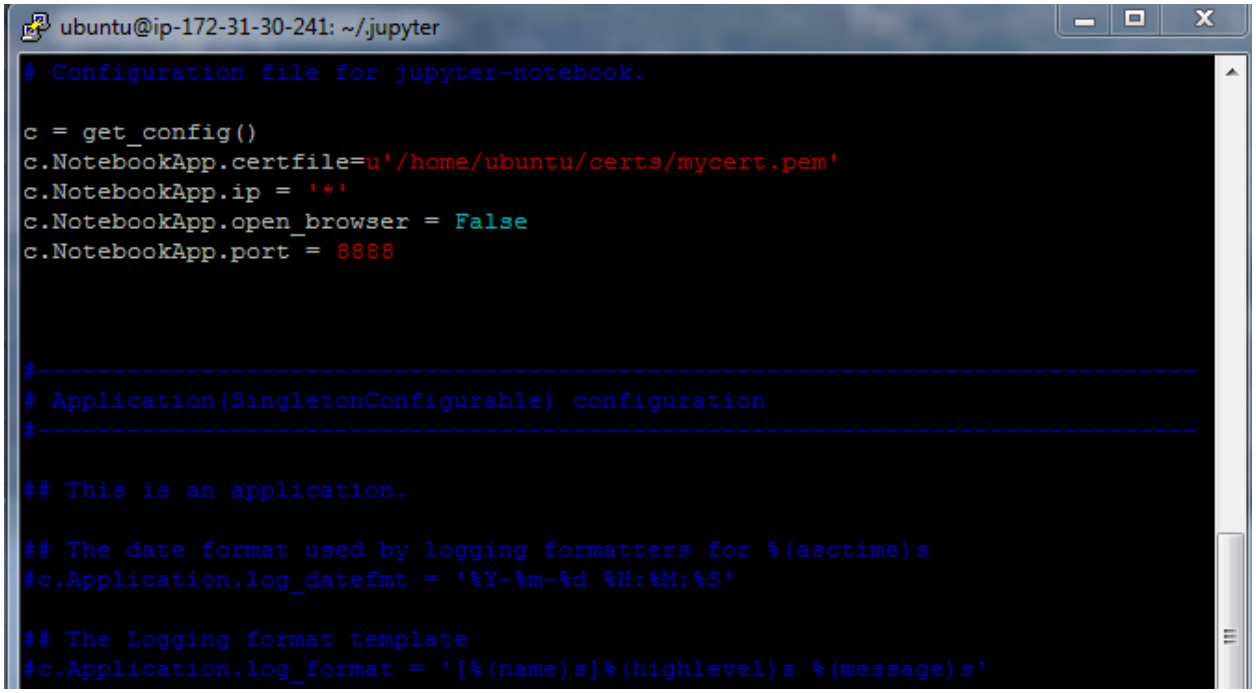
```
ubuntu@ip-172-31-30-241:~$ ls
spark-2.1.1-bin-hadoop2.7  spark-2.1.1-bin-hadoop2.7.tgz
ubuntu@ip-172-31-30-241:~$ cd spark-2.1.1-bin-hadoop2.7/
ubuntu@ip-172-31-30-241:~/spark-2.1.1-bin-hadoop2.7$ pwd
/home/ubuntu/spark-2.1.1-bin-hadoop2.7
ubuntu@ip-172-31-30-241:~/spark-2.1.1-bin-hadoop2.7$
```

13) $ **cd** to go back to your home directory (it works like cd..)
14) $ **pip3 install findspark** findspark will help us connecting Python with Spark easily
15) **$ jupyter notebook --generate-config** (I faced an error after using this command saying that Jupyter is not installed and can be installed using **sudo snap install jupyter command**, so I used it.
It didn't work so I tried this command: **$ ~/.local/bin/jupyter-notebook --generate-config** and it worked.

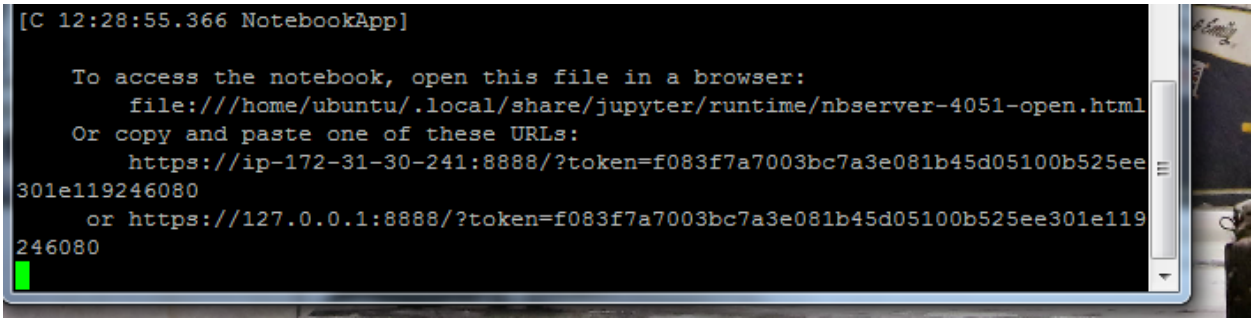
```
ubuntu@ip-172-31-30-241:~$ ~/.local/bin/jupyter-notebook --generate-config
Writing default config to: /home/ubuntu/.jupyter/jupyter_notebook_config.py
ubuntu@ip-172-31-30-241:~$
```

16) **$ cd**
17) **$ mkdir certs** (creating a directory for certificates)
18) **$ cd certs**

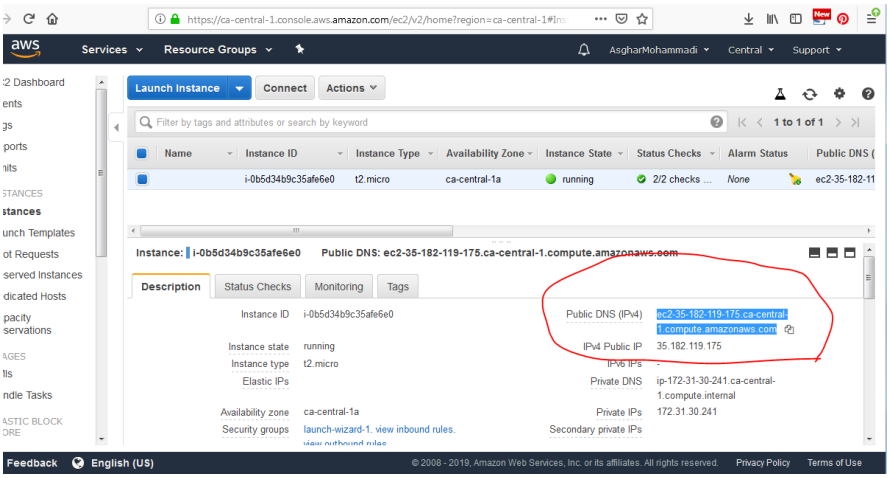
19) $ **sudo openssl req -x509 -nodes -days 365 -newkey rsa:1024 -keyout mycert.pem -out mycert.pem** (This will write our jupyter configuration in a .pem file). Then it asks for your country that you can put "CA" and you can leave the rest with blank.
20) $ **cd ~/.jupyter/** (this will take us to the jupyter hidden folder)
21) $ **vi jupyter_notebook_config.py** to see the inside of the file. Then we want to edit this file.
22) Press I in your keyboard to be able to edit the file. And then add these lines into the file:



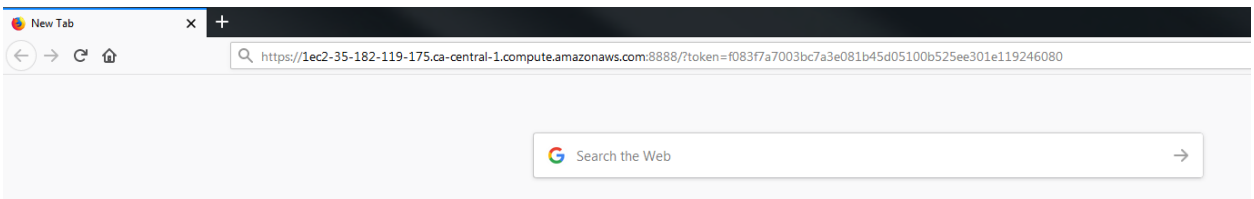
23) Then press esc to close from insert mode, then type ":" and then type "wq!" to save and quit the file.
24) $ cd (to come back to your home directory)
25) $ jupyter notebook (then it gives you a URL like: https://127.0.0.1:8888/?token=f083f7a7003bc7a3e081b45d05100b525ee301e119246080



26) Now copy this URL to your browser and change the "127.0.0.1" (or "localhost") with the public DNS of your ECT instant from AWS:
27)



Now the URL should look like:



28) Now hit Enter and you may see a warning page: (but in my case it didn't work and got an error of "This site can't be reached, then I checked the terminal to see the error. The error was Permission Denied for the certfile):

```
    handler_func(fileobj, events)
  File "/home/ubuntu/.local/lib/python3.6/site-packages/tornado/netutil.py", lin
e 273, in accept_handler
    callback(connection, address)
  File "/home/ubuntu/.local/lib/python3.6/site-packages/tornado/tcpserver.py", l
ine 288, in _handle_connection
    do_handshake_on_connect=False,
  File "/home/ubuntu/.local/lib/python3.6/site-packages/tornado/netutil.py", lin
e 605, in ssl_wrap_socket
    context = ssl_options_to_context(ssl_options)
  File "/home/ubuntu/.local/lib/python3.6/site-packages/tornado/netutil.py", lin
e 574, in ssl_options_to_context
    ssl_options["certfile"], ssl_options.get("keyfile", None)
PermissionError: [Errno 13] Permission denied
```

For fixing this issue we should terminate the Jupyter Notebook on the terminal by pressing ctrl+c, then write the command **$ cd** to go back to our home directory, then **$ cd certs**, then **$ sudo chmod 777 mycert.pem** this will change the permission so that the file can be reached by jupyter.

Then you can try the URL once again.

29) You may see a warning page, if so, you should click on "Advance" then accept the risk and continue. After that you should see your Jupyter Notebook.

https://ec2-35-182-119-175.ca-central-1.compute.amazonaws.com:8888/tree

jupyter

Files Running Clusters

Select items to perform actions on them.

0 /

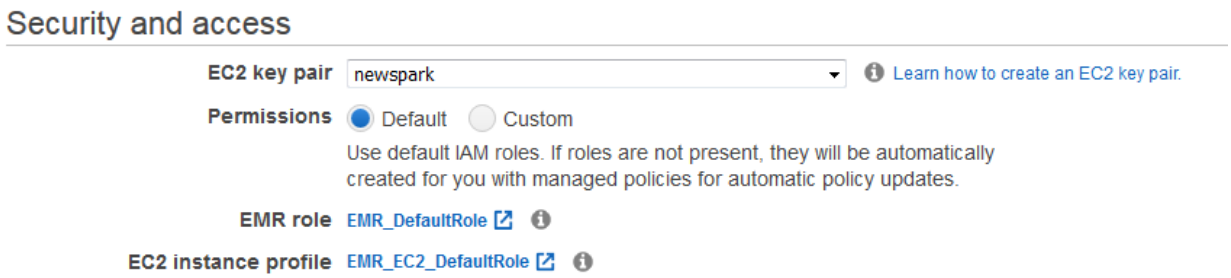
- certs
- spark-2.1.1-bin-hadoop2.7
- spark-2.1.1-bin-hadoop2.7.tgz

Amazon EMR (Elastic Map Reduce)

Setting up a cluster (is not free)
We will be using Zeppelin notebook, because it has a lot of Big Data tools built in it.

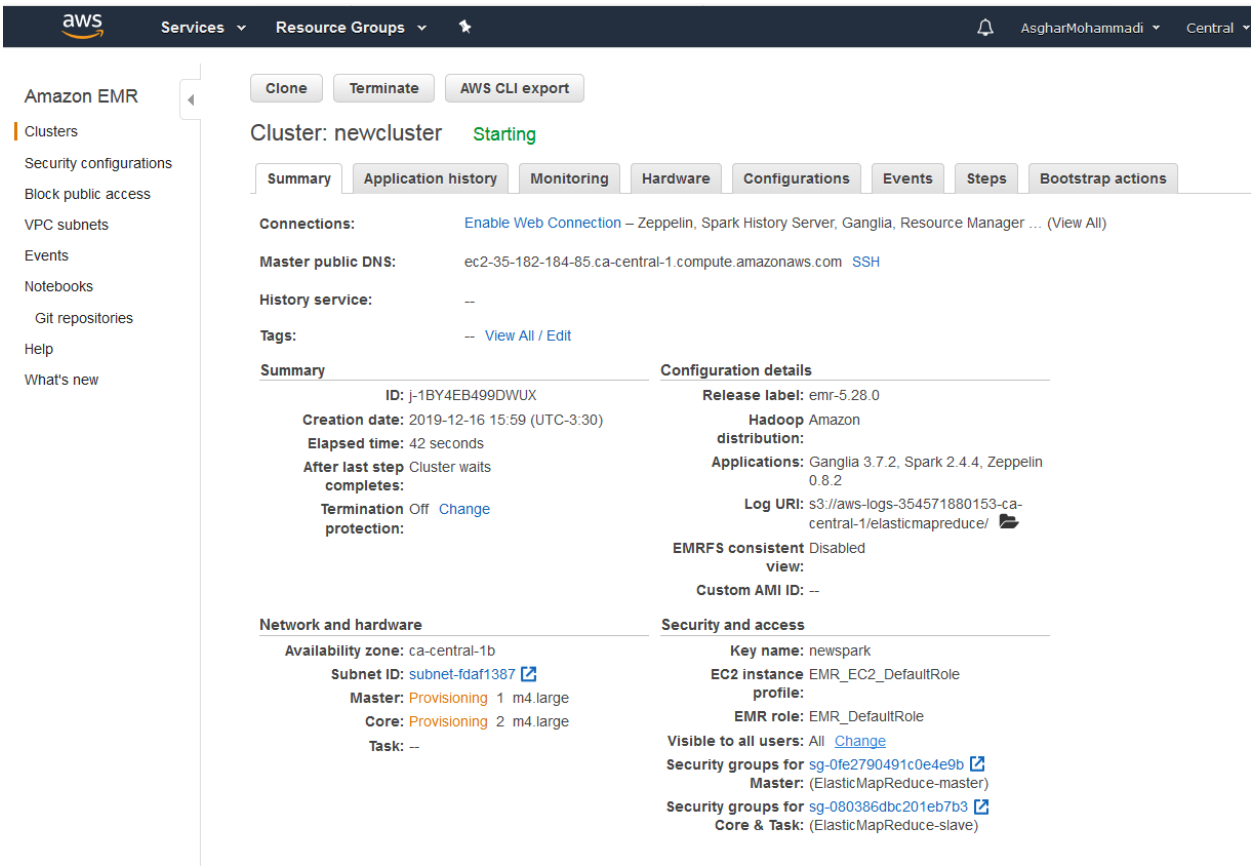
1) Log into your AWS Console and search for EMR

2) Select a name for your cluster
3) In software configuration part select the option that has both Spark and Zeppelin
4) For hardware configuration you can choose the number of servers and worker nodes that you want. We left it on default.
5) Security and Access: You can choose to create an Amazon EC2 Key Pair like what we did for the previous steps and use that for security purposes. Or you can choose the key pair that you already have in AWS. So we had the key pair “newspark” and we chose it.

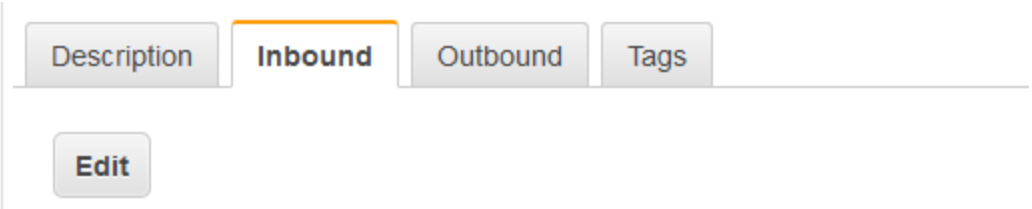


6) You are all set now and you can press Create Cluster button. And once you click on this button, you will be charged.

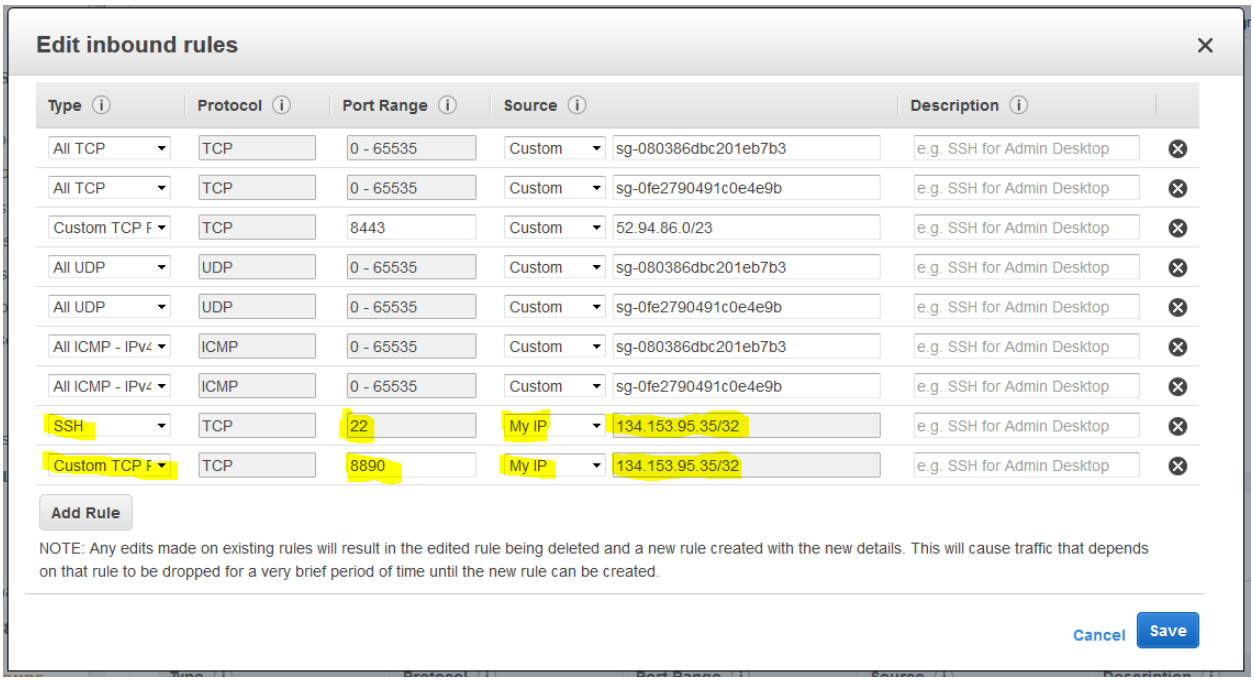
Now your cluster is created:



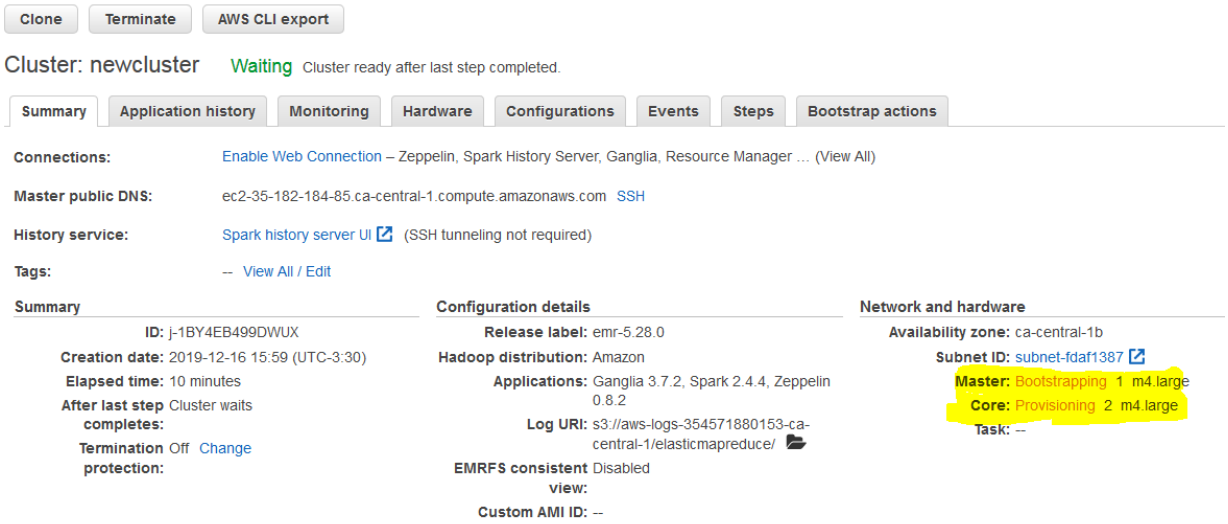
7) Now we should make sure we can SSH to Zeppelin notebook. But you may have to change the security settings first.
8) Click on the Security groups for Master, then click the Map Reduce Master, then select Inbound tab at the bottom of the page, then press Edit.



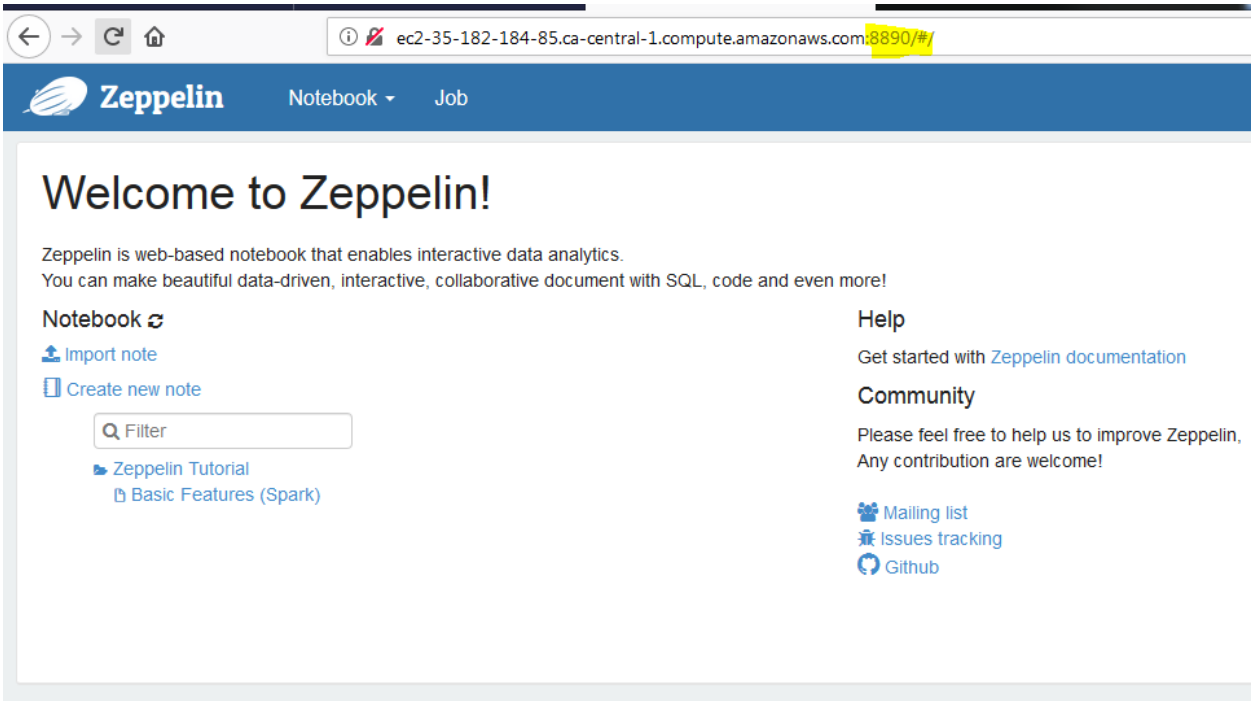
9) Then click Add Rule button. Then on Type column, select SSH and make sure the port is 22. On the Source column, you can either choose anywhere or my IP. The My YP lets you only connect with your physical IP address which is the safest way, but if you select anywhere it means anyone with any IP address can connect.
10) Please add another rule, and choose the port number 8890. And again choose My YP. Then click Save



11) Now you can get back to your AWS Console to see if the cluster is started. It usually takes a few minutes.

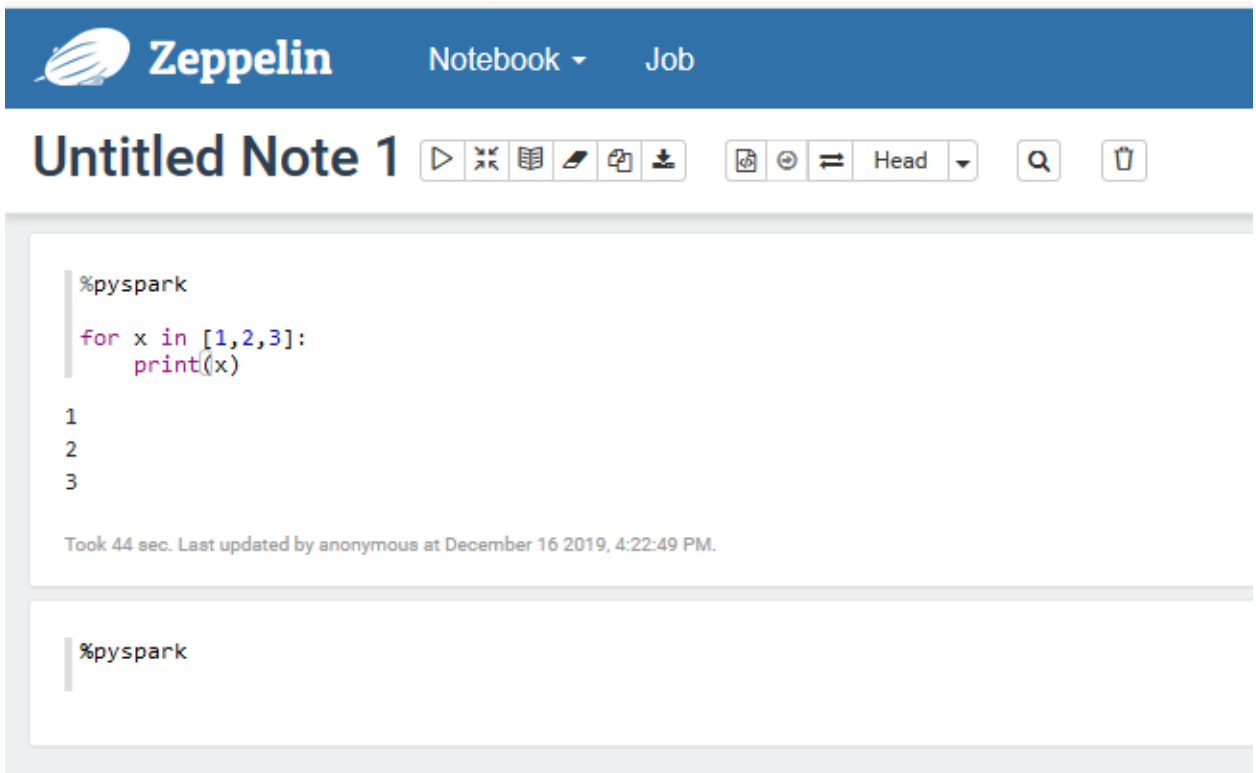


12) This means that the cluster is still working on it. As soon as the Master and Core are changed to Running (in green), then it means that it is up and running. (you may have to refresh the page to see if they are running)
13) Now you have two options to connect to the cluster. One is much more restrict for the security, the most secure one is to SSH to the cluster with the key pairs that have been prepared for you, but you can also copy and paste the Master public DNS to a browser and add ":8890" to the end of it and that will bring us to the Zeppelin notebook.
14)

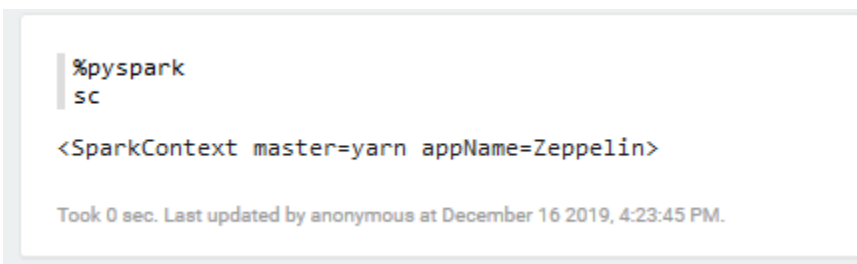


15) Now you can create a Zeppelin notebook by clicking on Create new note. And you can choose the interpreter. You can choose Spark for now.

16) Now if you want to use pySpark, in the first cell you can type %pyspark, and that tells Zeppelin that you are going to use pySpark.

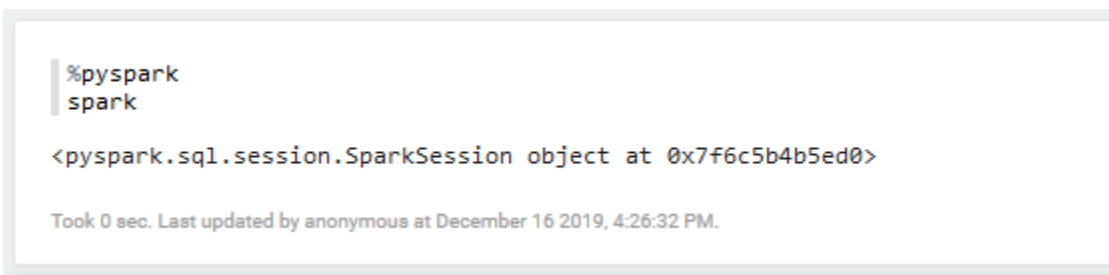


17) By typing sc in a new cell you can make sure you have access to Spark Context:



** But now it is better you use spark session because it's the newer data frame that spark is running on it.

18) Type spark in a new cell to see you can use the newer data frame that spark is using.



19) Here you can read a csv file:

```
%pyspark
df = spark.read.csv("s3n://Myaccesskey:secretkey@bucketname/myfile.csv")
```

FINISHED

```
<pyspark.sql.session.SparkSession object at 0x7fe433246d50>
```

Took 0 sec. Last updated by anonymous at April 03 2017, 4:17:32 PM.

After finishing your work, don't forget to terminate the cluster otherwise you will keep getting charged.

Spark DataFrames

- Spark began with something known as the RDD syntax which was a little ugly and tricky to learn
- Now Spark 2.0 and higher has shifted towards a DataFrame syntax.
- MLib is the Machine Learning tool for Spark

Working with Spark DataFrames:

1) Open Jupyter Notebook
2) Import SparkSession
3) Start the SparkSession

```
In [1]: import findspark

In [2]: findspark.init('/home/ubuntu/spark-2.1.1-bin-hadoop2.7')

In [3]: import pyspark

In [4]: from pyspark.sql import SparkSession

In [5]: spark = SparkSession.builder.appName('Basics').getOrCreate()

In [ ]:
```

4) Read a data set

```
In [5]: spark = SparkSession.builder.appName('Basics').getOrCreate()
```

```
In [6]: df = spark.read.json('people.json')
```

```
In [7]: df.show()
```

```
+----+-------+
| age|   name|
+----+-------+
|null|Michael|
|  30|   Andy|
|  19| Justin|
+----+-------+
```

```
In [8]: df.printSchema()
```

```
root
 |-- age: long (nullable = true)
 |-- name: string (nullable = true)
```

```
In [9]: df.columns
Out[9]: ['age', 'name']
```

```
In [10]: df.describe()
Out[10]: DataFrame[summary: string, age: string, name: string]
```

```
In [11]: df.describe().show()
```

```
+-------+------------------+-------+
|summary|               age|   name|
+-------+------------------+-------+
|  count|                 2|      3|
|   mean|              24.5|   null|
| stddev|7.7781745930520225|   null|
|    min|                19|   Andy|
|    max|                30|Michael|
+-------+------------------+-------+
```

DataFrame Schema:

Sometimes you need to clearify what the schema is. They has to know what columns are string, what columns are integer, etc.
For the rest, you can see the notebook on AWS