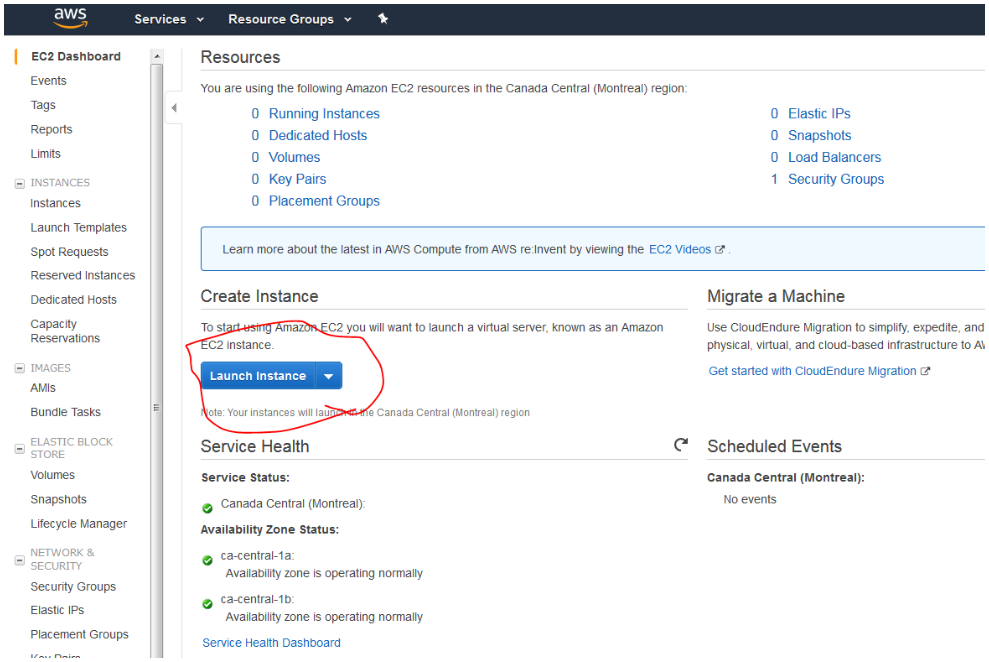
1) Creating an account on AWS Console

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:



Resources

You are using the following Amazon EC2 resources in the Canada Central (Montreal) region:

0 Running Instances	0 Elastic IPs
0 Dedicated Hosts	0 Snapshots
0 Volumes	0 Load Balancers
0 Key Pairs	1 Security Groups
0 Placement Groups	

Learn more about the latest in AWS Compute from AWS re:Invent by viewing the EC2 Videos.

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Launch Instance

Note: Your instances will launch in the Canada Central (Montreal) region

Migrate a Machine

Use CloudEndure Migration to simplify, expedite, and physical, virtual, and cloud-based infrastructure to AV

Get started with CloudEndure Migration

Service Health

Service Status:

Canada Central (Montreal):

Availability Zone Status:

ca-central-1a:
Availability zone is operating normally

ca-central-1b:
Availability zone is operating normally

Service Health Dashboard

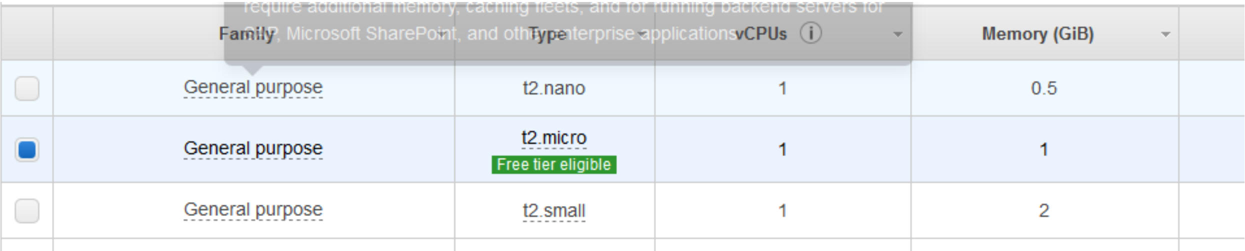
Scheduled Events

Canada Central (Montreal):

No events

2) Creating an instance on AWS EC2

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):



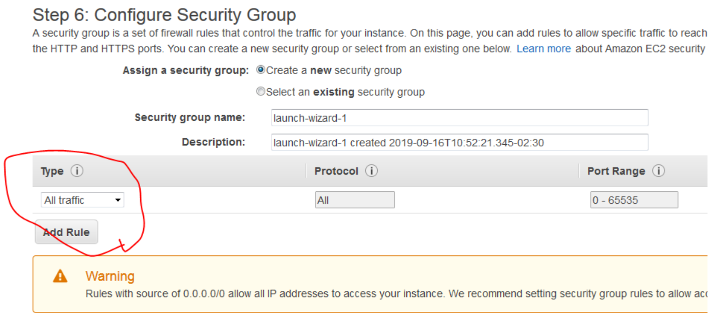
require additional memory, caching fleets, and for running backend servers for SAP, Microsoft SharePoint, and other enterprise applications

	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:



Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security

Assign a security group: ● Create a **new** security group

○ Select an **existing** security group

Security group name: launch-wizard-1

Description: launch-wizard-1 created 2019-09-16T10:52:21.345-02:30

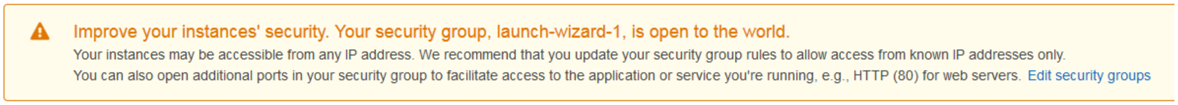
Type	Protocol	Port Range
All traffic	All	0 - 65535

Warning

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow acc

Then click on Review and Launch.

Here it gives you a warning:



Improve your instances' security. Your security group, launch-wizard-1, is open to the world.

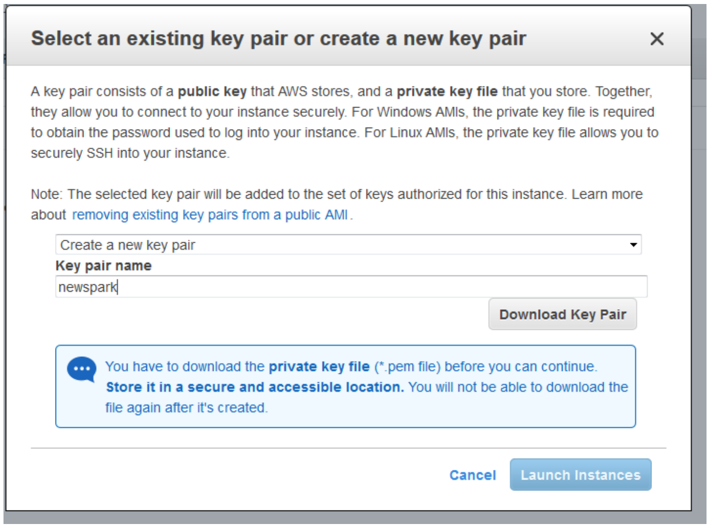
Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow access from known IP addresses only.

You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. Edit security groups

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:



Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI.

Create a new key pair

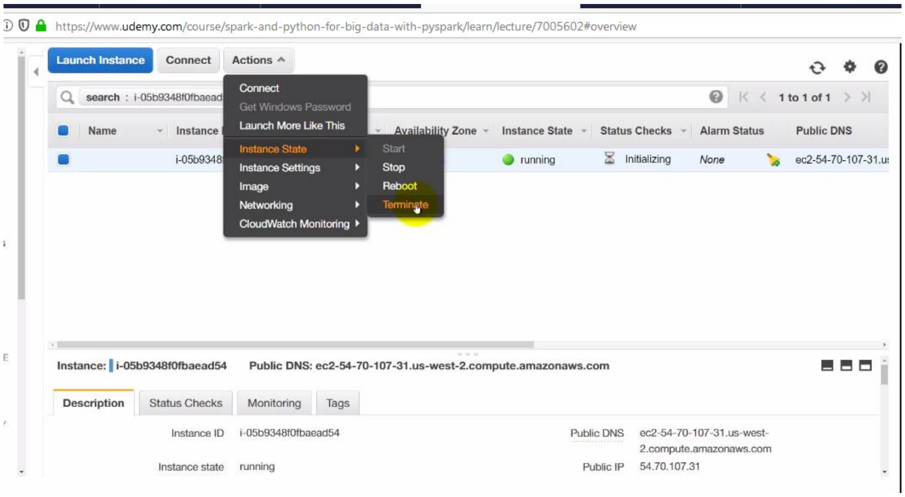
Key pair name

newspark

You have to download the **private key file** (*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

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:



Name	Instance	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
	i-05b9348		running	Initializing	None	ec2-54-70-107-31.us

Instance: i-05b9348f0fbaead54 Public DNS: ec2-54-70-107-31.us-west-2.compute.amazonaws.com

Instance ID	i-05b9348f0fbaead54	Public DNS	ec2-54-70-107-31.us-west-2.compute.amazonaws.com
Instance state	running	Public IP	54.70.107.31

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

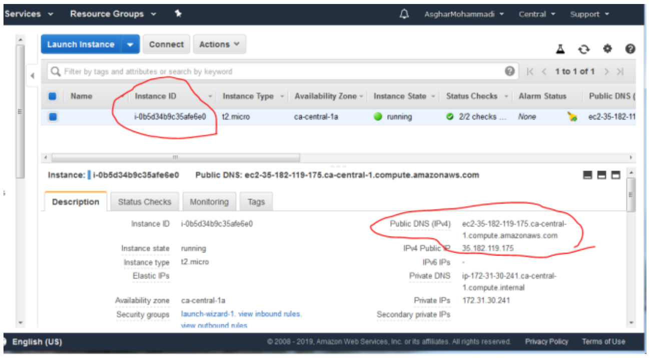
How to access your instance from a Windows Machine:

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) Installing PuTTY and setting up the security using Pair Key

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.



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (
	i-0b5d34b9c35afe6e0	t2.micro	ca-central-1a	running	2/2 checks ...	None	ec2-35-182-11

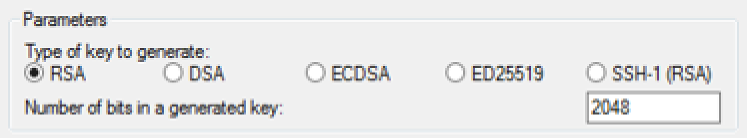
Instance: i-0b5d34b9c35afe6e0 Public DNS: ec2-35-182-119-175.ca-central-1.compute.amazonaws.com

Instance ID	i-0b5d34b9c35afe6e0	Public DNS (IPv4)	ec2-35-182-119-175.ca-central-1.compute.amazonaws.com
Instance state	running	IPv4 Public IP	35.182.119.175
Instance type	t2.micro	IPv6 IPs	-
Elastic IPs		Private DNS	ip-172-31-30-241.ca-central-1.compute.internal
Availability zone	ca-central-1a	Private IPs	172.31.30.241
Security groups	launch-wizard-1. view inbound rules. view outbound rules	Secondary private IPs	

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**.



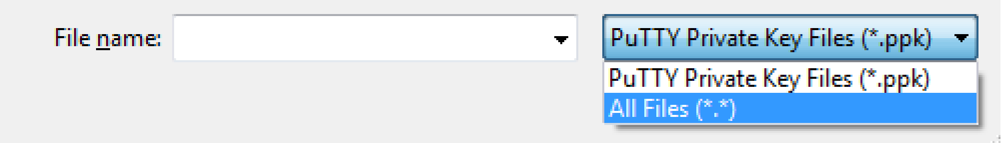
Parameters

Type of key to generate:
● RSA ○ DSA ○ ECDSA ○ ED25519 ○ SSH-1 (RSA)

Number of bits in a generated key: 2048

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.


File name:
PuTTY Private Key Files (*.ppk)
PuTTY Private Key Files (*.ppk)
All Files (*.*)

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)

4) Connecting to your AWS EC2 instance using PuTTY and setting up SSH Tunneling

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.

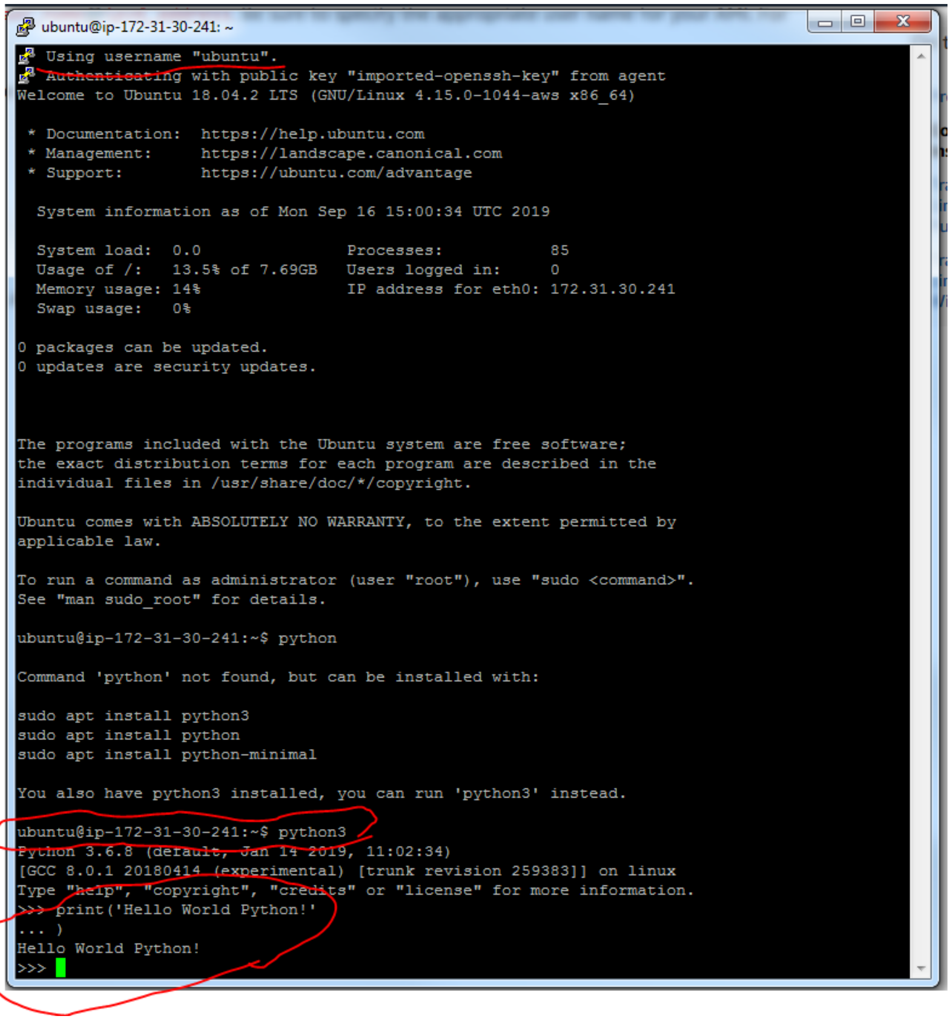
In the **Category** pane, expand **Connection**, expand **SSH**, and then choose **Auth**. Complete the following:

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:

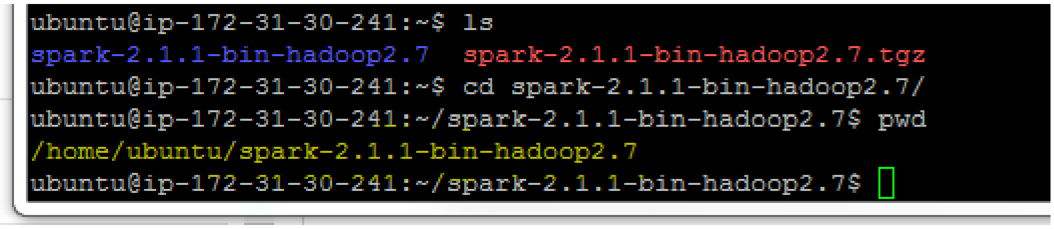


```
 Using username "ubuntu".
 Authenticating with public key "imported-openssh-key" from agent
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-1044-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Mon Sep 16 15:00:34 UTC 2019

  System load:  0.0               Processes:           85
  Usage of /:   13.5% of 7.69GB   Users logged in:     0
  Memory usage: 14%               IP address for eth0: 172.31.30.241
  Swap usage:   0%

0 packages can be updated.
0 updates are security updates.



The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-30-241:~$ python

Command 'python' not found, but can be installed with:

sudo apt install python3
sudo apt install python
sudo apt install python-minimal

You also have python3 installed, you can run 'python3' instead.

ubuntu@ip-172-31-30-241:~$ python3
Python 3.6.8 (default, Jan 14 2019, 11:02:34)
[GCC 8.0.1 20180414 (experimental) [trunk revision 259383]] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Hello World Python!'
... )
Hello World Python!
>>>
```

5) Installing Spark and Jupyter on your AWS EC2 Instance

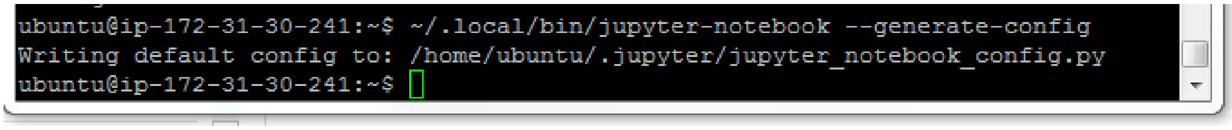
- **On this instance of EC2, we will**
- **Download Spark and then install it**
- **Install Jupyter Notebook**
- **Setting up PySpark and connecting to it**
- **SSH Tunneling from our local machine to AWS**
- **Running Jupyter Notebook on our local browser**

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.

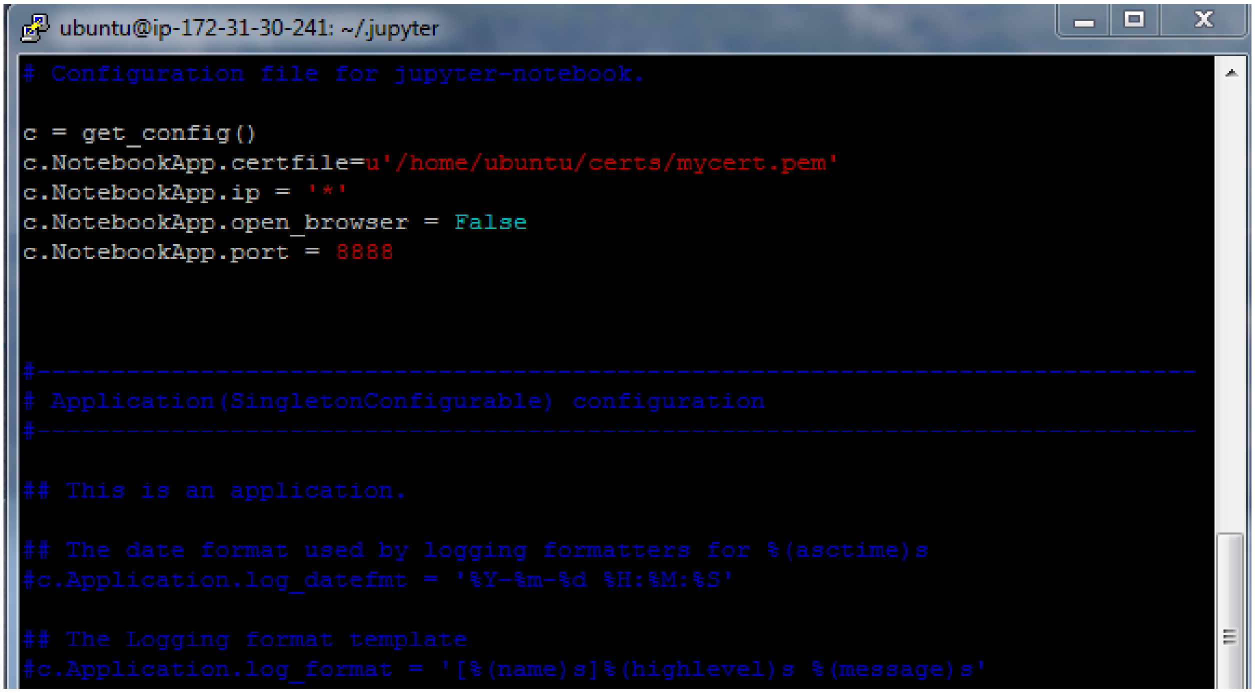


```
ubuntu@ip-172-31-30-241:~$ ~/.local/bin/jupyter-notebook --generate-config
Writing default config to: /home/ubuntu/.jupyter/jupyter_notebook_config.py
ubuntu@ip-172-31-30-241:~$
```

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:

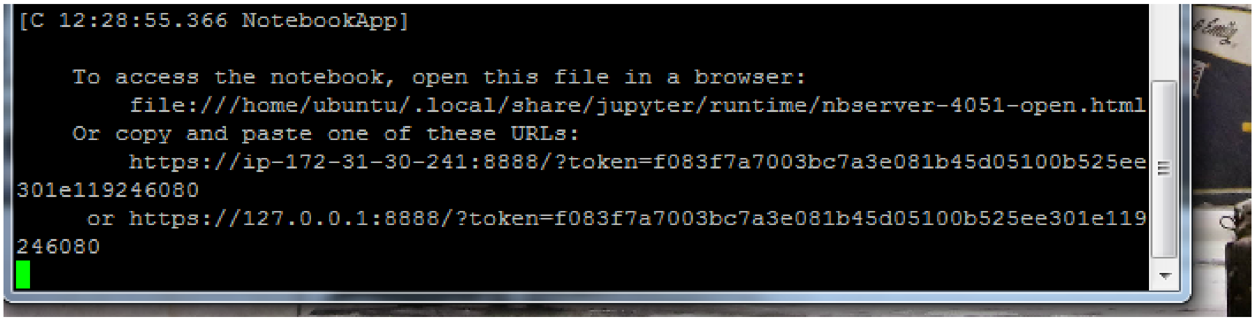


```
# Configuration file for jupyter-notebook.

c = get_config()
c.NotebookApp.certfile=u'/home/ubuntu/certs/mycert.pem'
c.NotebookApp.ip = '*'
c.NotebookApp.open_browser = False
c.NotebookApp.port = 8888



#------------------------------------------------------------------------------
# Application(SingletonConfigurable) configuration
#------------------------------------------------------------------------------

## This is an application.

## The date format used by logging formatters for %(asctime)s
#c.Application.log_datefmt = '%Y-%m-%d %H:%M:%S'

## The Logging format template
#c.Application.log_format = '[%(name)s]%(highlevel)s %(message)s'
```

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)

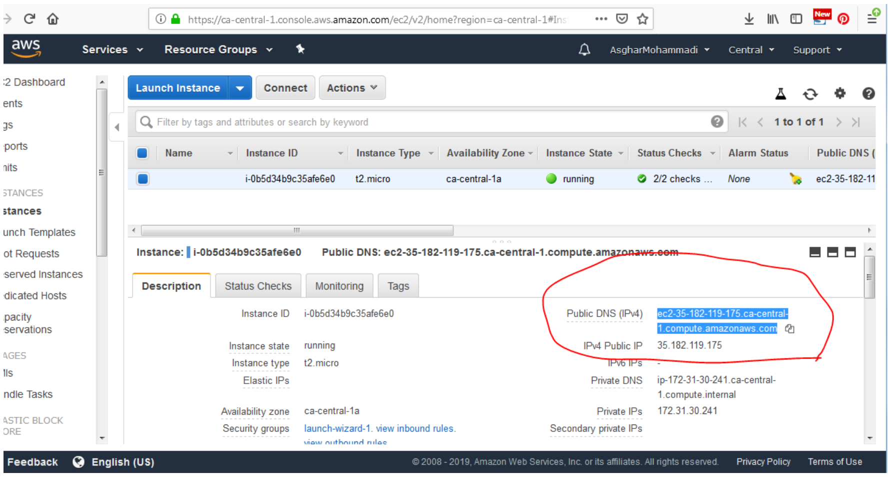
25) $ jupyter notebook (then it gives you a URL like: https://127.0.0.1:8888/?token=f083f7a7003bc7a3e081b45d05100b525ee301e119246080



```
[C 12:28:55.366 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///home/ubuntu/.local/share/jupyter/runtime/nbserver-4051-open.html
    Or copy and paste one of these URLs:
        https://ip-172-31-30-241:8888/?token=f083f7a7003bc7a3e081b45d05100b525ee
301e119246080
     or https://127.0.0.1:8888/?token=f083f7a7003bc7a3e081b45d05100b525ee301e119
246080
```

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)

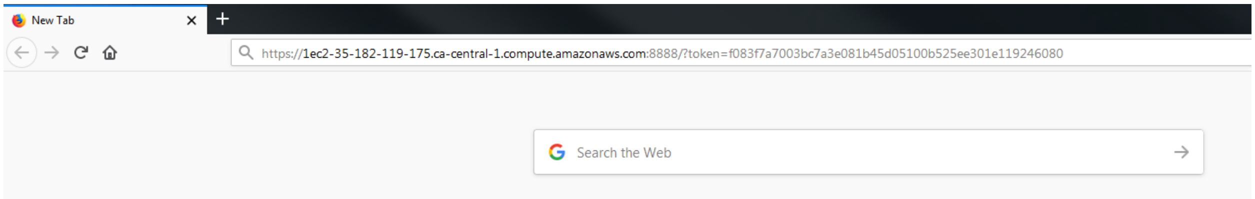


Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (
	i-0b5d34b9c35afe6e0	t2.micro	ca-central-1a	running	2/2 checks ...	None	ec2-35-182-11

Instance: i-0b5d34b9c35afe6e0 Public DNS: ec2-35-182-119-175.ca-central-1.compute.amazonaws.com

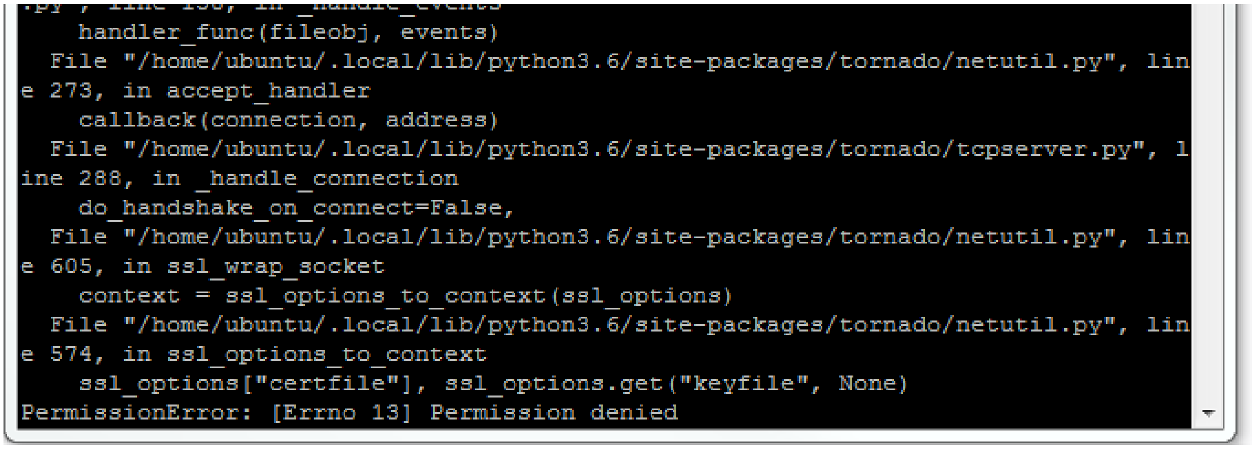
Instance ID	i-0b5d34b9c35afe6e0	Public DNS (IPv4)	ec2-35-182-119-175.ca-central-1.compute.amazonaws.com
Instance state	running	IPv4 Public IP	35.182.119.175
Instance type	t2.micro	IPv6 IPs	-
Elastic IPs		Private DNS	ip-172-31-30-241.ca-central-1.compute.internal
Availability zone	ca-central-1a	Private IPs	172.31.30.241
Security groups	launch-wizard-1. view inbound rules. view outbound rules	Secondary private IPs	

Now the URL should look like:



https://1ec2-35-182-119-175.ca-central-1.compute.amazonaws.com:8888/?token=f083f7a7003bc7a3e081b45d05100b525ee301e119246080

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):

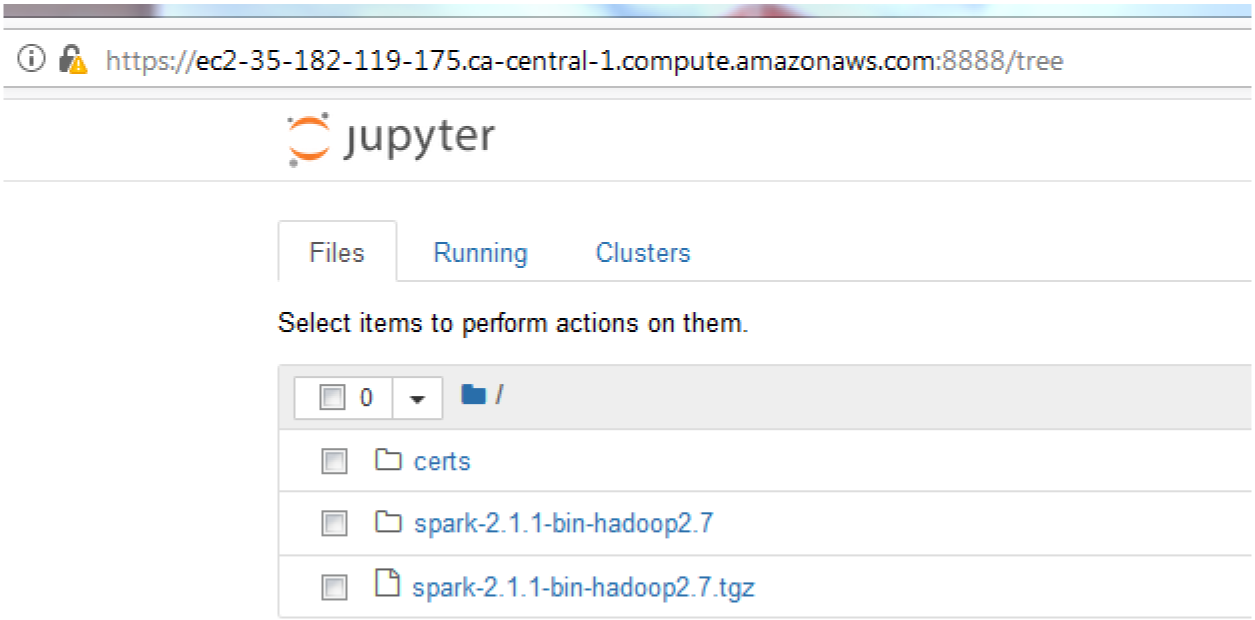


```
.py", line 159, in _handle_events
    handler_func(fileobj, events)
  File "/home/ubuntu/.local/lib/python3.6/site-packages/tornado/netutil.py", lin
e 273, in accept_handler
    callback(connection, address)
  File "/home/ubuntu/.local/lib/python3.6/site-packages/tornado/tcpserver.py", l
ine 288, 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

Select items to perform actions on them.

□ 0 /

□ certs

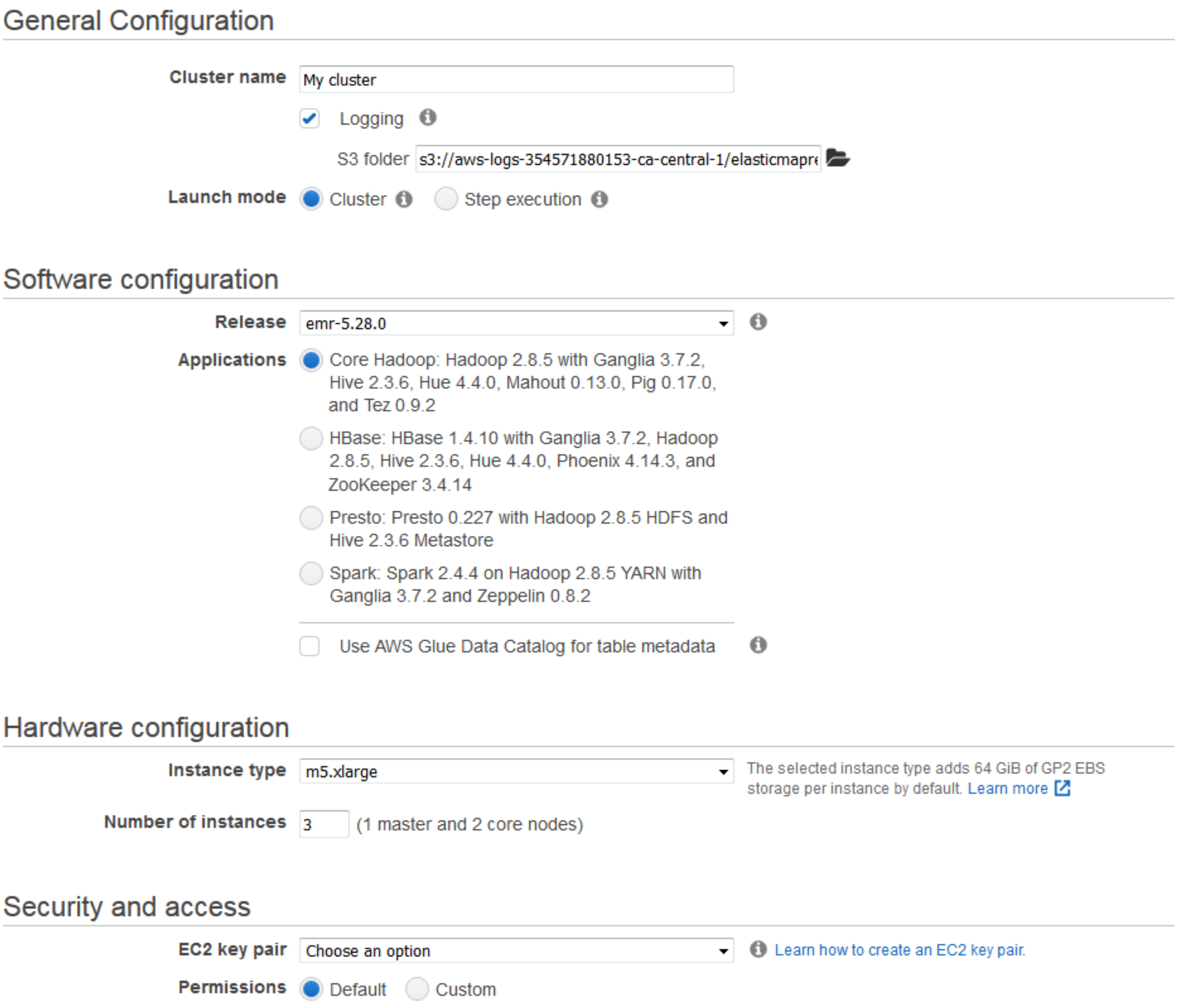
□ spark-2.1.1-bin-hadoop2.7

□ 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



General Configuration

Cluster name My cluster

☑ Logging

S3 folder s3://aws-logs-354571880153-ca-central-1/elasticmapre

Launch mode ● Cluster ○ Step execution

Software configuration

Release emr-5.28.0

Applications

● Core Hadoop: Hadoop 2.8.5 with Ganglia 3.7.2, Hive 2.3.6, Hue 4.4.0, Mahout 0.13.0, Pig 0.17.0, and Tez 0.9.2

○ HBase: HBase 1.4.10 with Ganglia 3.7.2, Hadoop 2.8.5, Hive 2.3.6, Hue 4.4.0, Phoenix 4.14.3, and ZooKeeper 3.4.14

○ Presto: Presto 0.227 with Hadoop 2.8.5 HDFS and Hive 2.3.6 Metastore

○ Spark: Spark 2.4.4 on Hadoop 2.8.5 YARN with Ganglia 3.7.2 and Zeppelin 0.8.2

□ Use AWS Glue Data Catalog for table metadata

Hardware configuration

Instance type m5.xlarge

The selected instance type adds 64 GiB of GP2 EBS storage per instance by default. Learn more

Number of instances 3 (1 master and 2 core nodes)

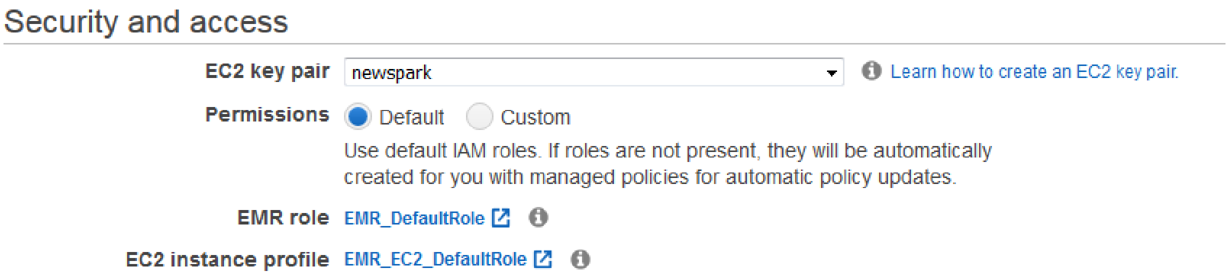
Security and access

EC2 key pair Choose an option

Learn how to create an EC2 key pair.

Permissions ● Default ○ Custom

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.



Security and access

EC2 key pair: newspark

Permissions: ● Default ○ Custom

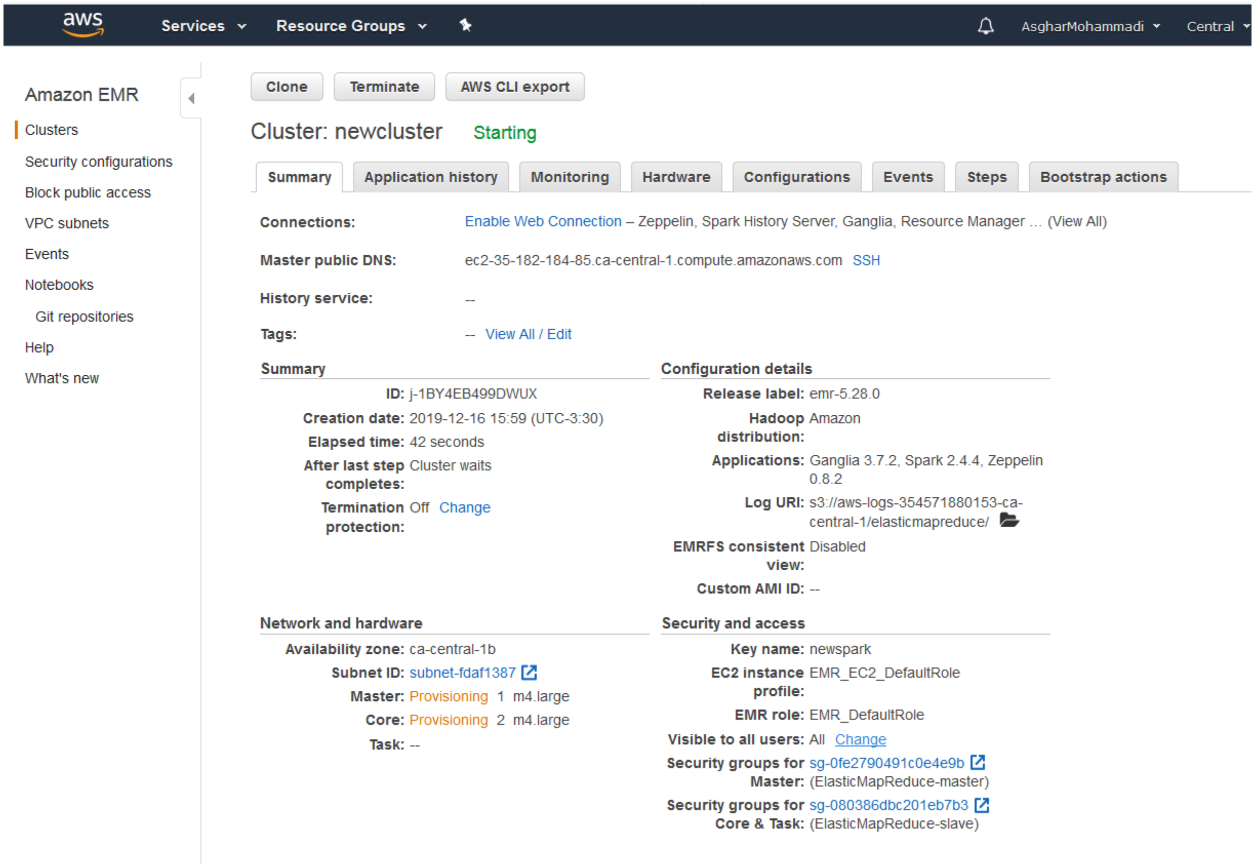
Use default IAM roles. If roles are not present, they will be automatically created for you with managed policies for automatic policy updates.

EMR role: EMR_DefaultRole

EC2 instance profile: EMR_EC2_DefaultRole

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:



Cluster: newcluster Starting

Connections: Enable Web Connection – Zeppelin, Spark History Server, Ganglia, Resource Manager ... (View All)

Master public DNS: ec2-35-182-184-85.ca-central-1.compute.amazonaws.com SSH

History service: --

Tags: -- View All / Edit

Summary

ID: j-1BY4EB499DWUX
Creation date: 2019-12-16 15:59 (UTC-3:30)
Elapsed time: 42 seconds
After last step completes: Cluster waits
Termination protection: Off Change

Configuration details

Release label: emr-5.28.0
Hadoop distribution: Amazon
Applications: Ganglia 3.7.2, Spark 2.4.4, Zeppelin 0.8.2
Log URI: s3://aws-logs-354571880153-ca-central-1/elasticmapreduce/
EMRFS consistent view: Disabled
Custom AMI ID: --

Network and hardware

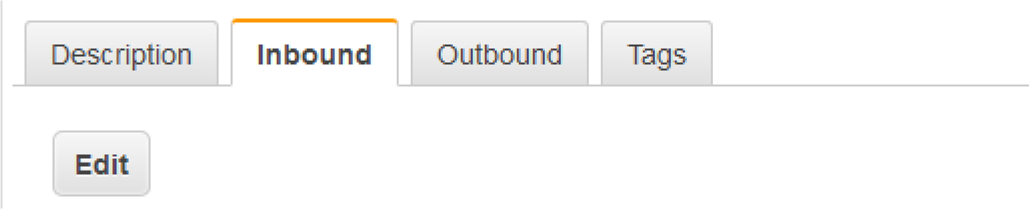
Availability zone: ca-central-1b
Subnet ID: subnet-fdaf1387
Master: Provisioning 1 m4.large
Core: Provisioning 2 m4.large
Task: --

Security and access

Key name: newspark
EC2 instance profile: EMR_EC2_DefaultRole
EMR role: EMR_DefaultRole
Visible to all users: All Change
Security groups for Master: sg-0fe2790491c0e4e9b (ElasticMapReduce-master)
Security groups for Core & Task: sg-080386dbc201eb7b3 (ElasticMapReduce-slave)

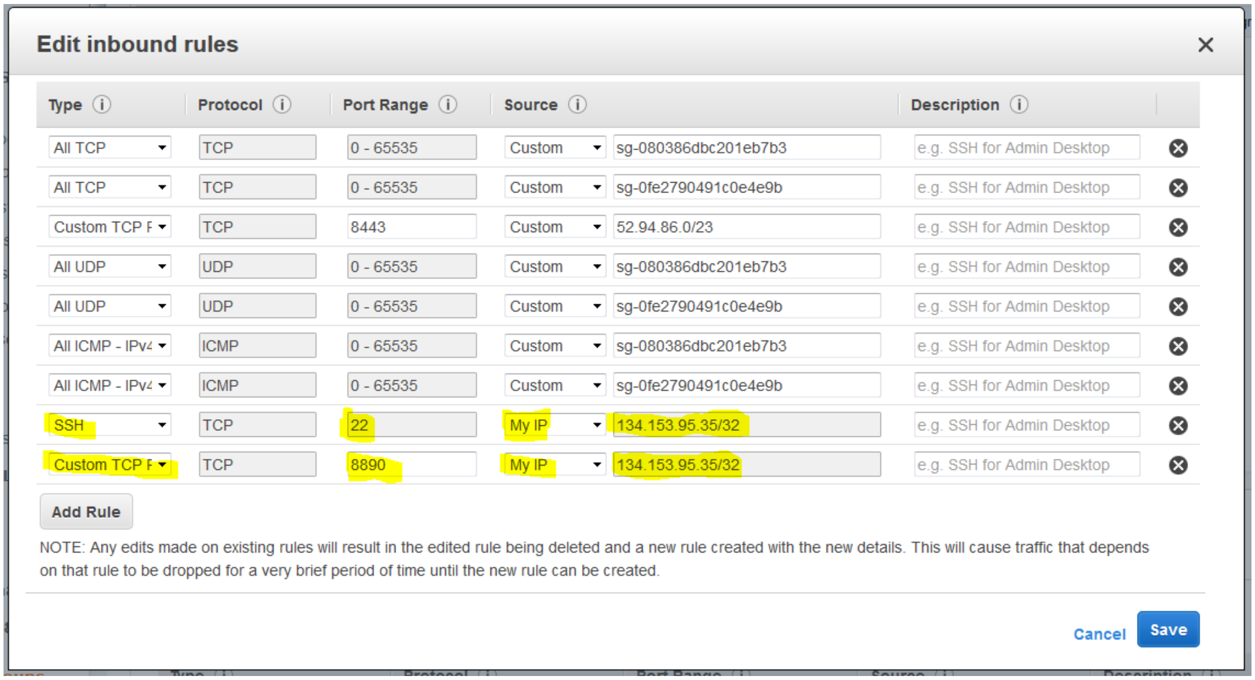
7) Now we should make sure we can SSH to Zeppelin notebook. But you may have to change the security settings first.

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



Edit inbound rules ✕

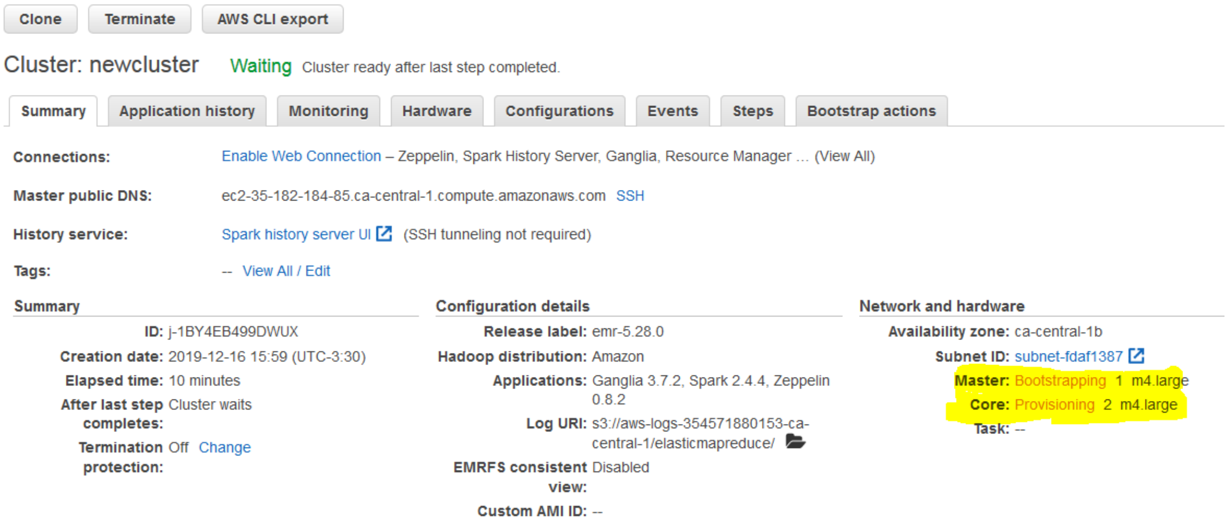
Type	Protocol	Port Range	Source		Description	
All TCP	TCP	0 - 65535	Custom	sg-080386dbc201eb7b3	e.g. SSH for Admin Desktop	⊗
All TCP	TCP	0 - 65535	Custom	sg-0fe2790491c0e4e9b	e.g. SSH for Admin Desktop	⊗
Custom TCP F	TCP	8443	Custom	52.94.86.0/23	e.g. SSH for Admin Desktop	⊗
All UDP	UDP	0 - 65535	Custom	sg-080386dbc201eb7b3	e.g. SSH for Admin Desktop	⊗
All UDP	UDP	0 - 65535	Custom	sg-0fe2790491c0e4e9b	e.g. SSH for Admin Desktop	⊗
All ICMP - IPv4	ICMP	0 - 65535	Custom	sg-080386dbc201eb7b3	e.g. SSH for Admin Desktop	⊗
All ICMP - IPv4	ICMP	0 - 65535	Custom	sg-0fe2790491c0e4e9b	e.g. SSH for Admin Desktop	⊗
SSH	TCP	22	My IP	134.153.95.35/32	e.g. SSH for Admin Desktop	⊗
Custom TCP F	TCP	8890	My IP	134.153.95.35/32	e.g. SSH for Admin Desktop	⊗

Add Rule

NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic that depends on that rule to be dropped for a very brief period of time until the new rule can be created.

Cancel Save

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



Cluster: newcluster Waiting Cluster ready after last step completed.

Connections: Enable Web Connection – Zeppelin, Spark History Server, Ganglia, Resource Manager ... (View All)

Master public DNS: ec2-35-182-184-85.ca-central-1.compute.amazonaws.com SSH

History service: Spark history server UI (SSH tunneling not required)

Tags: -- View All / Edit

Summary

ID: j-1BY4EB499DWUX
Creation date: 2019-12-16 15:59 (UTC-3:30)
Elapsed time: 10 minutes
After last step completes: Cluster waits
Termination protection: Off Change

Configuration details

Release label: emr-5.28.0
Hadoop distribution: Amazon
Applications: Ganglia 3.7.2, Spark 2.4.4, Zeppelin 0.8.2
Log URI: s3://aws-logs-354571880153-ca-central-1/elasticmapreduce/
EMRFS consistent view: Disabled
Custom AMI ID: --

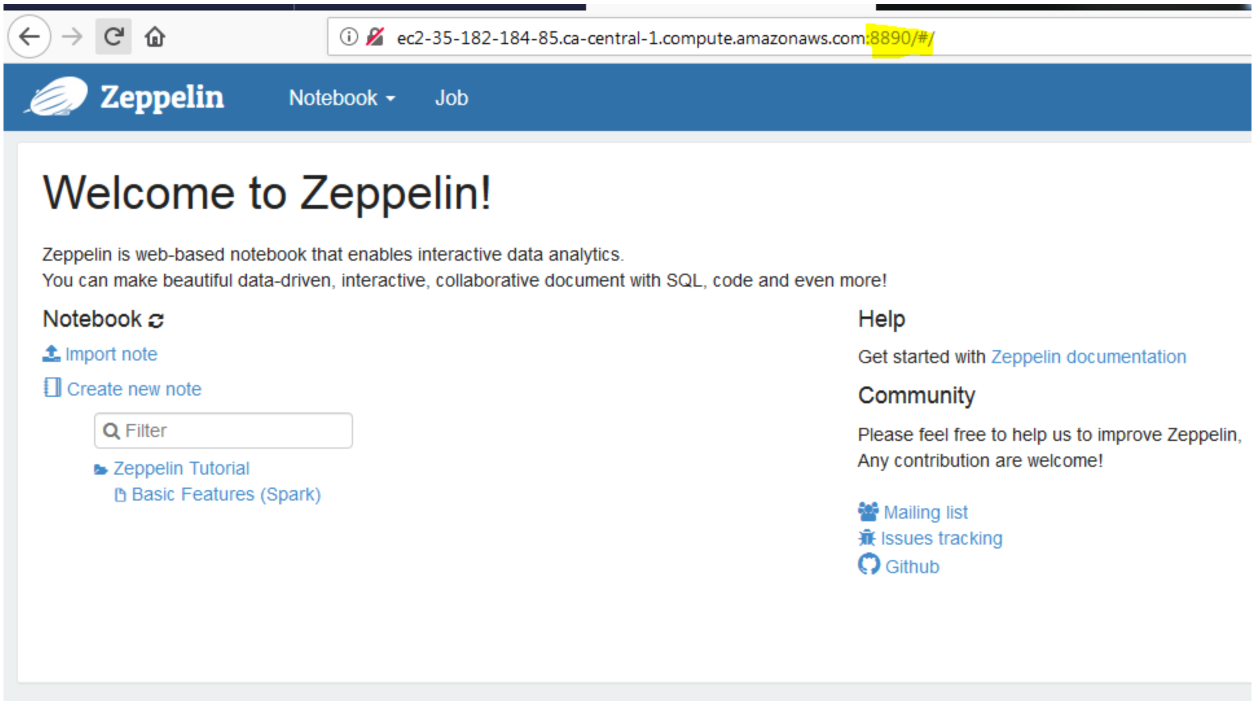
Network and hardware

Availability zone: ca-central-1b
Subnet ID: subnet-fdaf1387
Master: Bootstrapping 1 m4.large
Core: Provisioning 2 m4.large
Task: --

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)



ec2-35-182-184-85.ca-central-1.compute.amazonaws.com:8890/#/

Welcome to Zeppelin!

Zeppelin is web-based notebook that enables interactive data analytics.
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

Notebook

Import note
Create new note
Filter
Zeppelin Tutorial
Basic Features (Spark)

Help

Get started with Zeppelin documentation

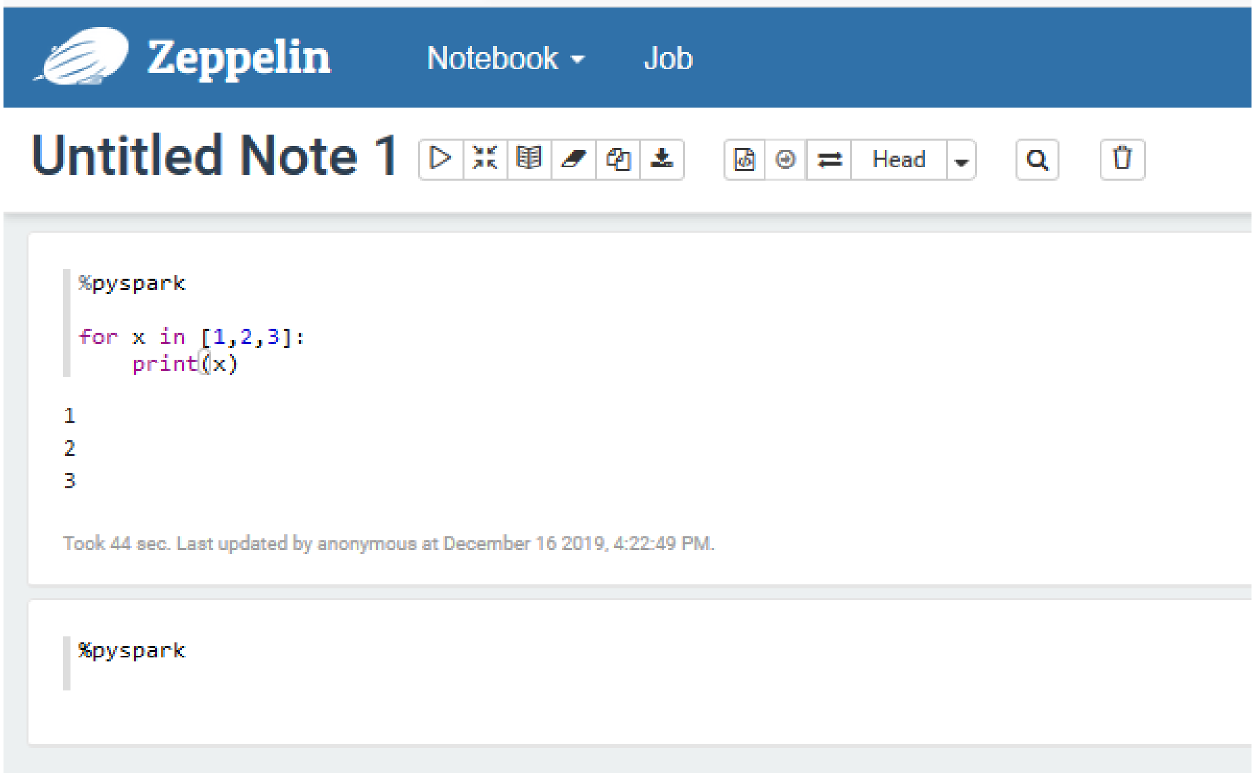
Community

Please feel free to help us to improve Zeppelin, Any contribution are welcome!

Mailing list
Issues tracking
Github

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.

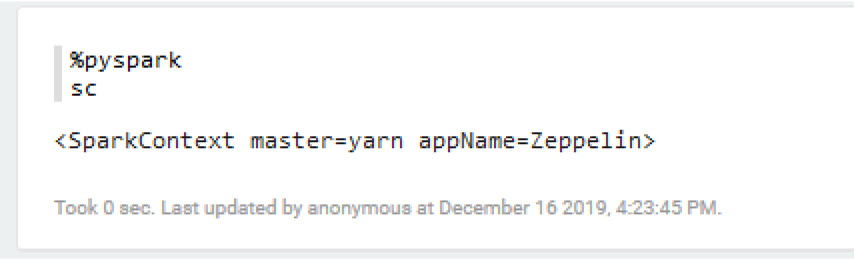


```
%pyspark

for x in [1,2,3]:
    print(x)

1
2
3
```

Took 44 sec. Last updated by anonymous at December 16 2019, 4:22:49 PM.

```
%pyspark
```

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

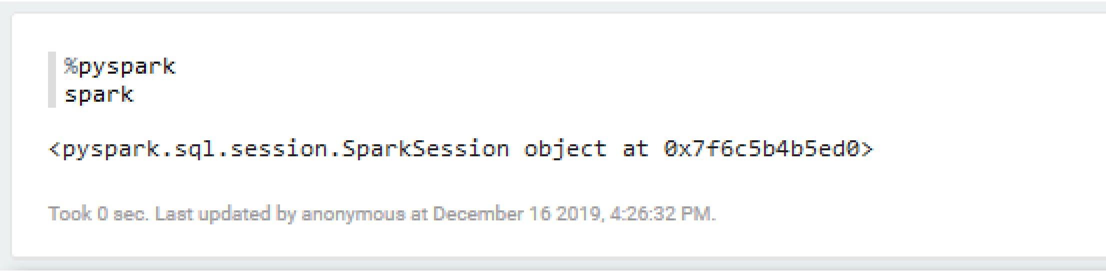


```
%pyspark
sc

<SparkContext master=yarn appName=Zeppelin>
```

Took 0 sec. Last updated by anonymous at December 16 2019, 4:23:45 PM.

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

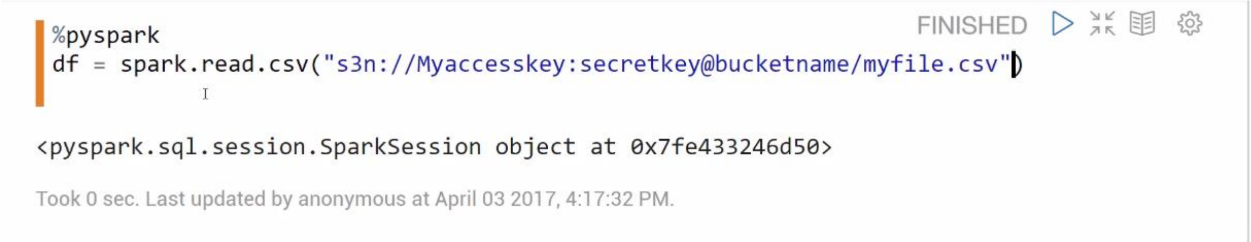
18) Type spark in a new cell to see you can use the newer data frame that spark is using.



```
%pyspark
spark

<pyspark.sql.session.SparkSession object at 0x7f6c5b4b5ed0>
```

Took 0 sec. Last updated by anonymous at December 16 2019, 4:26:32 PM.

19) Here you can read a csv file:



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

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

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

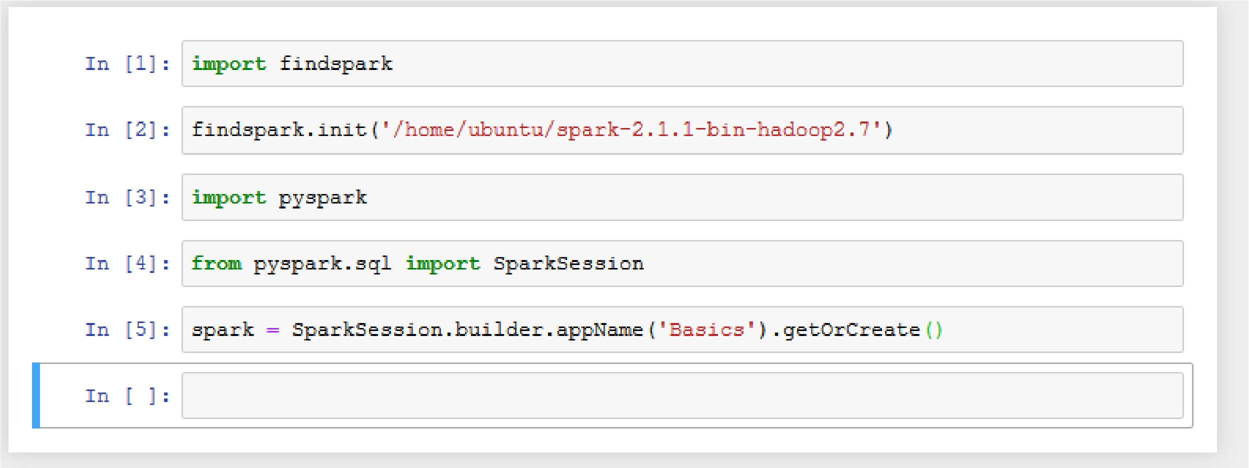
After finishing your work, don’t forget to terminate the cluster otherwise you will keep getting charged.

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

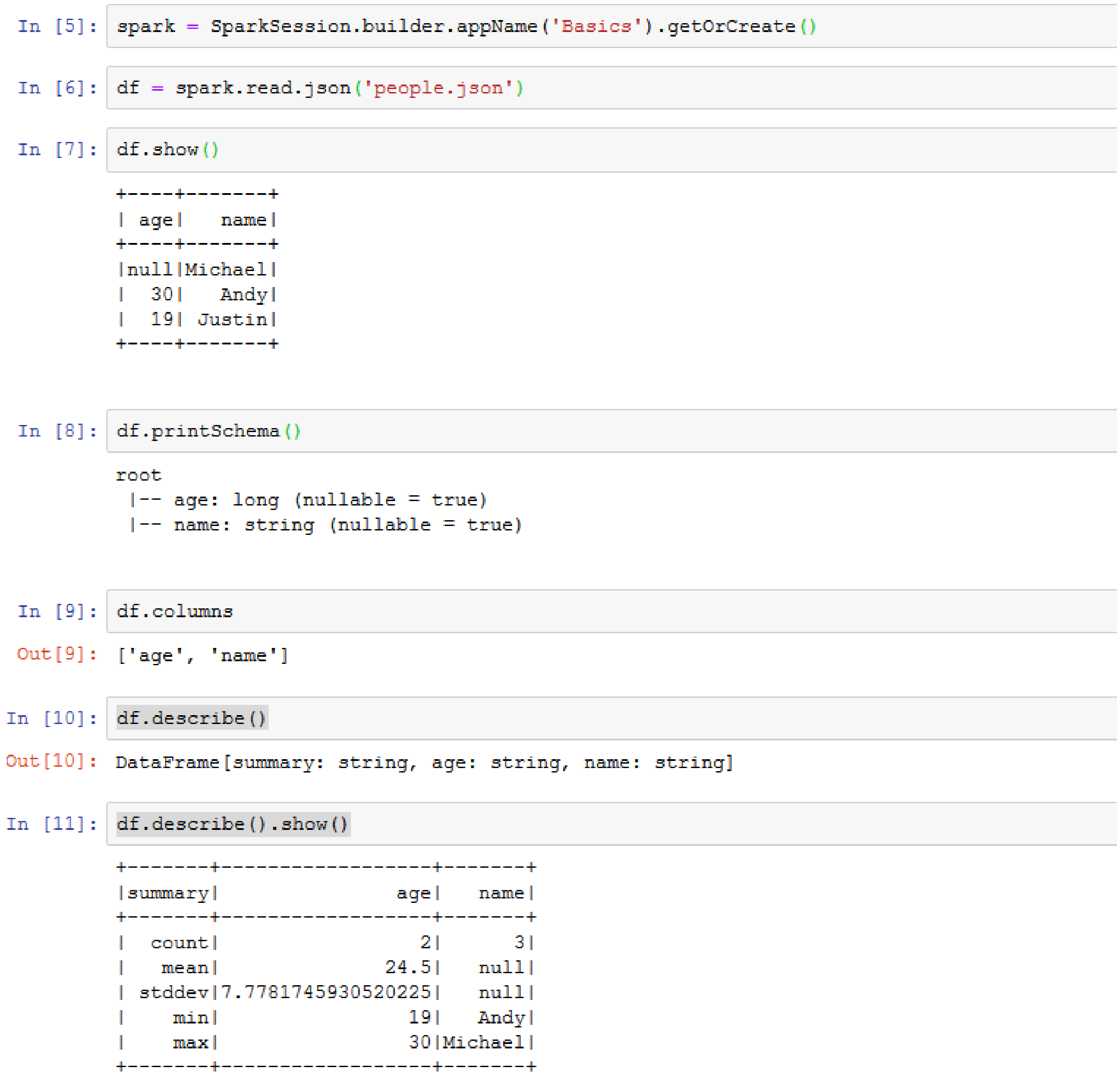
Working with Spark DataFrames:

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