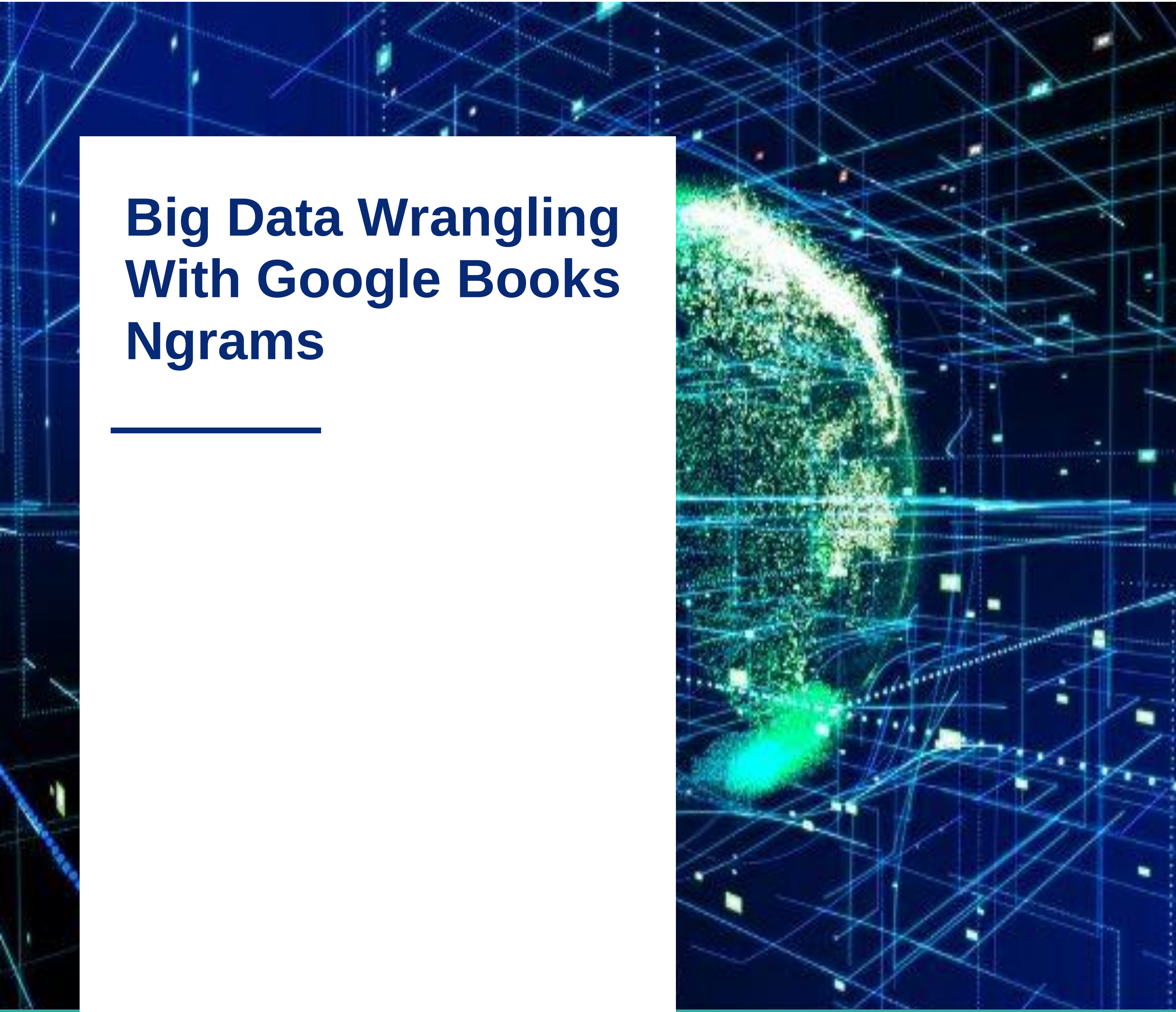


Big Data Wrangling With Google Books Ngrams

AUGUST 1, 2022

BY:

Tolulope Oludemi

Table of Contents

Introduction

This report summarizes the findings of loading, filtering, and visualizing a real-world dataset in a cloud-based distributed computing environment using Hadoop, Spark, Hive, and the S3 filesystem. The Appendix in the last few pages showcases screenshots of the steps completed for specific questions identified in the assignment.

The dataset used is the Google Ngrams dataset created by Google's research team. It entails digitized texts from Google Books from the 1800s to the 2000s.

Question 1

Spin up a new EMR cluster using the AWS Console. Be sure to include Hadoop, Spark, Hive, and Jupyterhub for your cluster. For the release version, make sure to use EMR 5.29.0.

1. The first thing I did was to go to the AWS Management Console at https://aws.amazon.com/console/. I then logged into my AWS account.
2. I clicked on EMR to take me to the EMR service. I then clicked on "Create Cluster".
3. I clicked "Advanced Options" at the top of the page which took me to "Step 1: Software and Steps". In the 'Release' dropdown, I selected EMR-5.29.0 as stated in the question.
4. I made sure to check boxes for Hadoop, Spark, Hive, and Jupyterhub. I clicked 'Next' until I got to 'Step 3: General Cluster Settings', where I named my cluster 'BigDataDeliverable'. I clicked 'Next' and in 'Step 4: Security', I chose my already created key pair from a previous class, 'tolubrainstation'.
5. I clicked 'Create Cluster', and the cluster creation process started with "Starting" displayed at the top of the page.
6. After waiting, "Starting" turned into "Running", meaning that my cluster was ready!

The screenshots of the process from Question 1 can be found in the Appendix.

Question 2

Connect to the head node of the cluster using SSH.

1. To connect to the head node of the cluster using SSH, I clicked on SSH under 'Master public DNS'
2. A pop-up appeared displaying instructions on how to connect to the master node using SSH.
3. I followed the instructions and went into the directory/filename that had my .pem file in the terminal.
4. I copied the command in the pop-up displaying instructions. I ran the command in my terminal and was greeted by the EMR ASCII banner, meaning that the connection was successful.

The screenshots of this process can be found in the Appendix.

Question 3

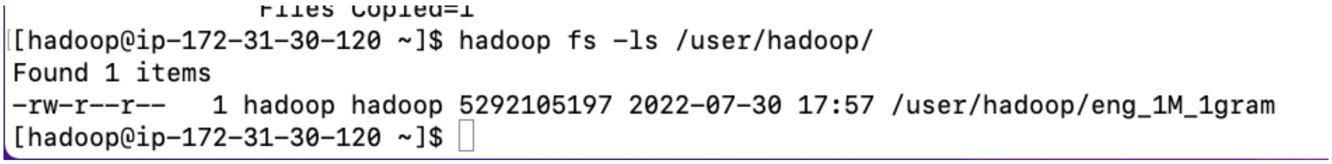
Copy the data folder from the S3 bucket directly into a directory on the Hadoop File System (HDFS) named */user/Hadoop/eng_1M_1gram*.

To copy the data folder from the S3 bucket, I used the command: 'hadoop distcp' to directly copy the data from the source to the data nodes. I put the following command in the terminal to copy the data folder from the S3 bucket directly into a directory on the HDFS:

```
hadoop distcp s3://brainstation-dsft/eng_1M_1gram.csv
/user/hadoop/eng_1M_1gram
```

It should be noted that there is a space between '.csv' and '/user/'. After I ran the command above, I used "*hadoop fs -ls /user/hadoop/*" to make sure that the data folder from the S3 bucket was copied to a directory on the HDFS named "*eng_1M_1gram*".

The screenshot showing the data folder in the HDFS is provided below:



```
            Files Copied=1
[hadoop@ip-172-31-30-120 ~]$ hadoop fs -ls /user/hadoop/
Found 1 items
-rw-r--r--   1 hadoop hadoop 5292105197 2022-07-30 17:57 /user/hadoop/eng_1M_1gram
[hadoop@ip-172-31-30-120 ~]$ 
```

Question 4

Using pyspark, read the data you copied into HDFS in Step 3.

1. Under 'EMR on EC2', I clicked 'Notebooks', then I created a Notebook, using the existing cluster created in Question 1. For the AWS service role, I used 'EMR_Notebooks_DefaultRole' since I am a returning user.
2. I clicked 'Create Notebook', and after the status changed from 'Starting' to 'Running', I clicked 'Open in Jupyter' to view the Notebook.
3. I clicked on the created Notebook – BigDataDeliverable-Notebook and changed the Kernel to 'PySpark'. I made sure it worked by running "spark" and after getting results stating: "SparkSession available as "spark"." I moved onto reading the data by using the following code:

```
df = spark.read.csv('s3://brainstation-dsft/eng_1M_1gram.csv', header=True)
```

PART A

Describe the dataset (examples include size, shape, schema) in pyspark.

The dataset has 261,823,225 rows and 5 columns. The datatype for the dataset were all strings. The columns are token, year, frequency, pages, and books.

PART B

Create a new DataFrame from a query using Spark SQL, filtering to include only the rows where the token is "data" and describe the new dataset.

1. I used the following code to register the data as a view in the Spark session prior to filtering:

```
df.createOrReplaceTempView(“google_books”)
```

2. I then filtered out rows that only had “data” for the “token” column, using the following code for the SQL query:

```
SELECT *
FROM google_books
WHERE token=”data”
```

The complete code used to create a new dataset with this filter was as follows:

```
df_data = spark.sql(“SELECT * FROM google_books WHERE token=’data’”)
```

PART C

Write the filtered data back to a directory in the HDFS from Spark using “df.write.csv()”. Be sure to pass the “header=True” parameter and examine the contents of what you’ve written.

I used the ‘*df.write.csv()*’ to write the filtered data back to a directory in the HDFS from Spark. I created a new directory called “*filtered_data*” to store the filtered data. The code I used to do this is as follows:

```
df_data.write.csv("/user/hadoop/filtered_data", header=True)
```

The notebook, named “BigDataDeliverable-Notebook” with answers for Question 4 are included in the submission.

Question 5

Collect the contents of the directory into a single file on the local drive of the head node using “getmerge” and move this file into a S3 bucket in your account.

I used the following command in my terminal to collect the contents of the “*filtered_data*” directory into a single file on the local drive of the head node:

```
hadoop fs -getmerge /user/hadoop/filtered_data filtered_data.csv
```

As stated earlier, I created a new directory where the csv was saved, named “*filtered*_data”. The “*getmerge*” command was used to merge all the files in the source directory into one file which was saved as “*filtered_data.csv*” in the local drive of the head node.
I went into my S3 bucket on AWS to find the path link, and I used the “aws s3 cp” command to move the “filtered_data.csv” into my S3 bucket. The exact command used in the terminal is as follows:

```
aws s3 cp filtered_data.csv s3://tolubucket1
```

The csv file is now saved in my “tolubucket1”.

Question 6

On your local machine (or on AWS outside of Spark) in python, read the CSV data from the S3 folder into a pandas DataFrame.

I first used the command: “aws configure” to input my AWS Access ID and Secret Access Key obtained from my security credentials on AWS Management Console.
I opened Jupyter Notebook through Anaconda using Python 3 kernel, and I read the CSV data from the “tolubucket1” folder by following these steps:

1. Imported libraries – pandas and boto3.
    a. I had to first install boto3 in my terminal using the following command:

```
conda install boto3 s3fs
```

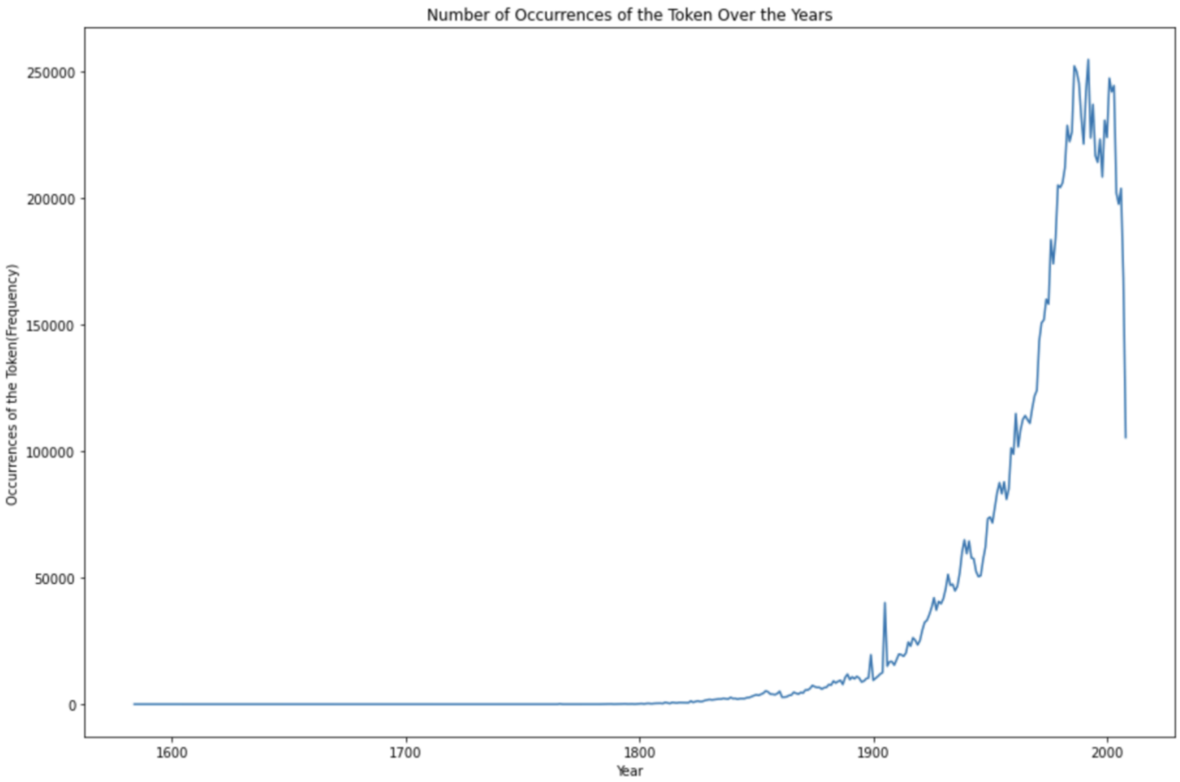
2. I defined the variables needed to read the csv file (the bucket name and file name).
3. Created the low-level functional client since I'll only be accessing one folder.
4. Created the S3 object using the bucket name and file name.
5. Read the csv using "pd.read_csv".

The notebook, named Big-Data-Question-6 with steps on how I answered this question is included in the submission.

Question 7

Plot the number of occurrences of the token (the *frequency* column) of the "data" over the years using matplotlib.

After reading the csv data from the S3 folder into a pandas DataFrame, I imported matplotlib and plotted the "Year" column against the "frequency" column to plot the number of occurrences of the token over the years. The plot is provided below.



The notebook with detailed answers on Question 6 and 7 are provided in the submission.

Question 8

Compare Hadoop and Spark as distributed file systems.

PART A

What are the advantages/differences between Hadoop and Spark? List two advantages for each.

<u>Differences:</u>

- Hadoop is an open-source software framework designed for storage and processing, while Spark is a lightning-fast in-memory cluster computing technology – allowing more type of computations over Hadoop.
- Hadoop MapReduce just reads and writes from a disk, but Spark immediately stores each read/write iteration resulting in faster processing.

Advantages of Hadoop:

- Because Hadoop is open source, it is free to use, and uses commodity hardware, so it is economical.
- Hadoop is scalable, it reliably stores and processes petabytes of data.

Advantages of Spark:

- Because Spark is a lightning-fast in-memory cluster computing technology, the processing speed is very fast, faster than Hadoop.
- Spark supports MapReduce like Hadoop, but in addition to that, it also supports Machine Learning, SQL queries, graph algorithms, etc.

PART B

Explain how the HDFS stores the data.

The HDFS stores the data by splitting the files into blocks (no more than 128 MB in size) and distributes the separate blocks into different nodes in a cluster. The NameNode which is the master server that manages the file system, keeps track of the blocks that make up a file, and where they are stored.

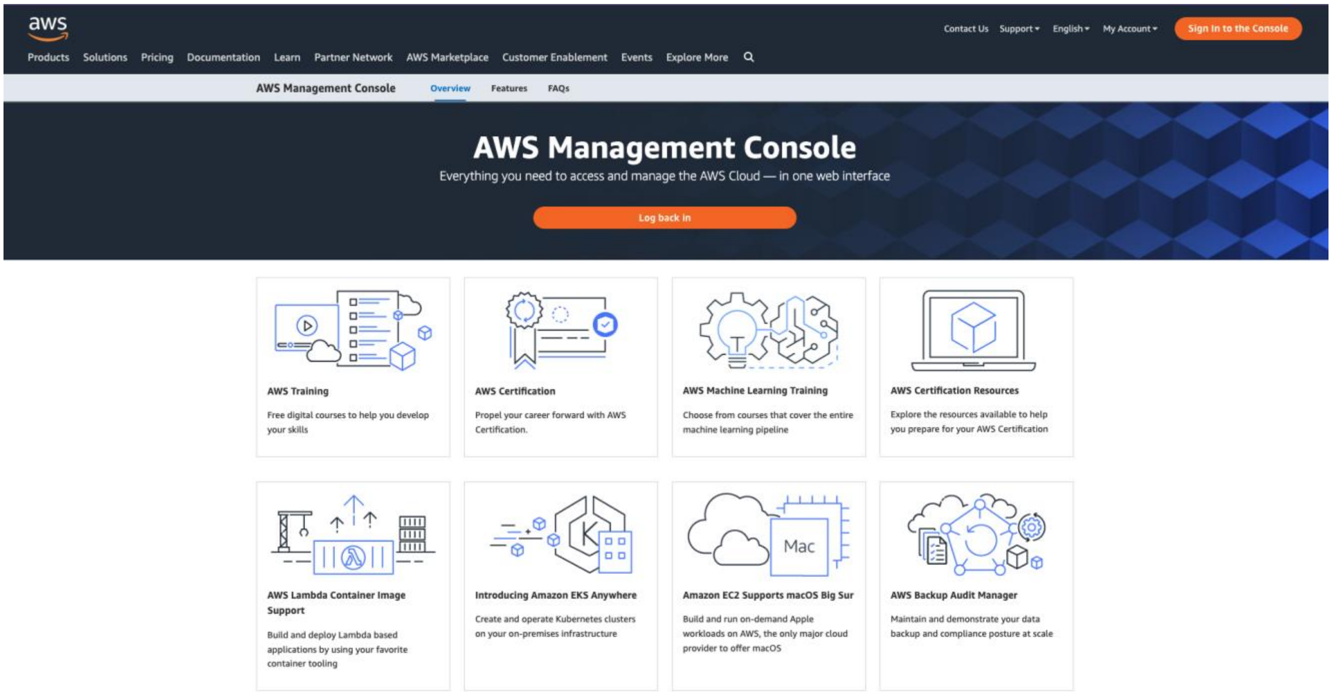
Conclusion

This report outlined the steps taken to load, filter, and visualize the Google Ngrams dataset in a cloud-based computing distributed environment. In addition, Hadoop and Spark were differentiated and compared, explaining how the Hadoop File System works. The Appendix showing screenshots of how Question 1,2, and 4 were answered are included in the next page.

Appendix

QUESTION 1 SCREENSHOTS

AWS Management Console Log In Page



AWS Management Console

Everything you need to access and manage the AWS Cloud — in one web interface

AWS Training

Free digital courses to help you develop your skills

AWS Certification

Propel your career forward with AWS Certification.

AWS Machine Learning Training

Choose from courses that cover the entire machine learning pipeline

AWS Certification Resources

Explore the resources available to help you prepare for your AWS Certification

AWS Lambda Container Image Support

Build and deploy Lambda based applications by using your favorite container tooling

Introducing Amazon EKS Anywhere

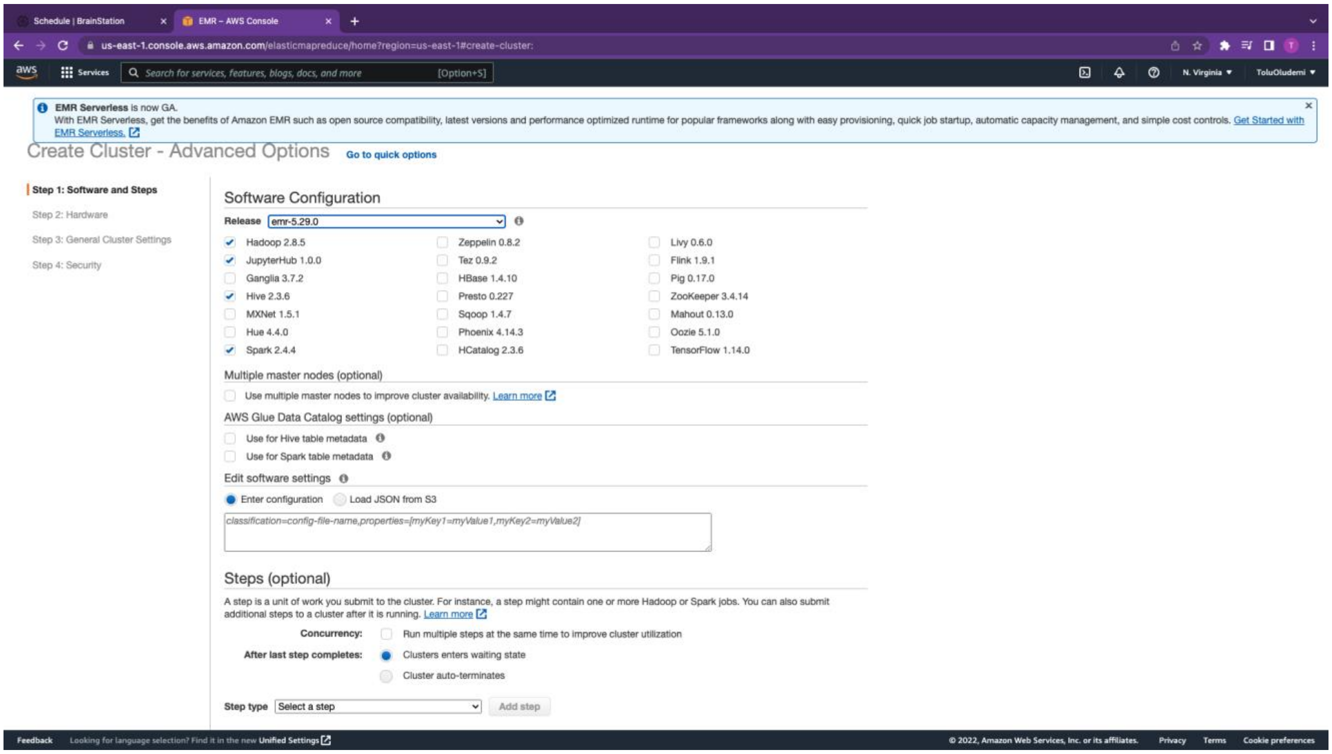
Create and operate Kubernetes clusters on your on-premises infrastructure

Amazon EC2 Supports macOS Big Sur

Build and run on-demand Apple workloads on AWS, the only major cloud provider to offer macOS

AWS Backup Audit Manager

Maintain and demonstrate your data backup and compliance posture at scale



us-east-1.console.aws.amazon.com/elasticmapreduce/home?region=us-east-1#create-cluster:

EMR Serverless is now GA.
With EMR Serverless, get the benefits of Amazon EMR such as open source compatibility, latest versions and performance optimized runtime for popular frameworks along with easy provisioning, quick job startup, automatic capacity management, and simple cost controls. Get Started with EMR Serverless.

Create Cluster - Advanced Options   Go to quick options

Step 1: Software and Steps

Step 2: Hardware

Step 3: General Cluster Settings

Step 4: Security

Software Configuration

Release emr-5.29.0

☑ Hadoop 2.8.5
☑ JupyterHub 1.0.0
□ Ganglia 3.7.2
☑ Hive 2.3.6
□ MXNet 1.5.1
□ Hue 4.4.0
☑ Spark 2.4.4
□ Zeppelin 0.8.2
□ Tez 0.9.2
□ HBase 1.4.10
□ Presto 0.227
□ Sqoop 1.4.7
□ Phoenix 4.14.3
□ HCatalog 2.3.6
□ Livy 0.6.0
□ Flink 1.9.1
□ Pig 0.17.0
□ ZooKeeper 3.4.14
□ Mahout 0.13.0
□ Oozie 5.1.0
□ TensorFlow 1.14.0

Multiple master nodes (optional)

□ Use multiple master nodes to improve cluster availability. Learn more

AWS Glue Data Catalog settings (optional)

□ Use for Hive table metadata
□ Use for Spark table metadata

Edit software settings

● Enter configuration ○ Load JSON from S3

classification=config-file-name,properties=[myKey1=myValue1,myKey2=myValue2]

Steps (optional)

A step is a unit of work you submit to the cluster. For instance, a step might contain one or more Hadoop or Spark jobs. You can also submit additional steps to a cluster after it is running. Learn more

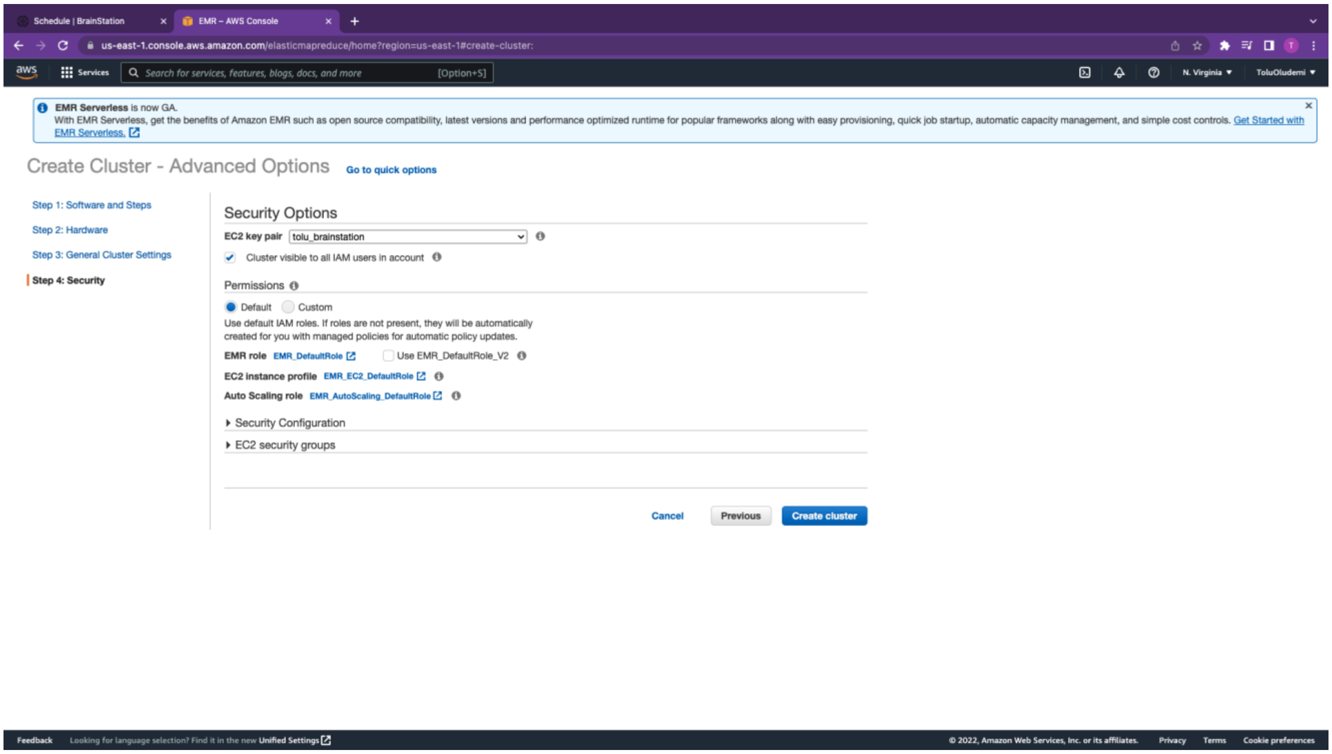
Concurrency: □ Run multiple steps at the same time to improve cluster utilization

After last step completes: ● Clusters enters waiting state
○ Cluster auto-terminates

Step type Select a step   Add step

Looking for language selection? Find it in the new **Unified Settings**

© 2022, Amazon Web Services, Inc. or its affiliates.



Schedule | BrainStation
EMR – AWS Console
us-east-1.console.aws.amazon.com/elasticmapreduce/home?region=us-east-1#create-cluster:
aws
Services
Search for services, features, blogs, docs, and more
[Option+S]
N. Virginia ▾
ToluOludemi ▾

EMR Serverless is now GA.
With EMR Serverless, get the benefits of Amazon EMR such as open source compatibility, latest versions and performance optimized runtime for popular frameworks along with easy provisioning, quick job startup, automatic capacity management, and simple cost controls. Get Started with EMR Serverless.

Create Cluster - Advanced Options   Go to quick options

- Step 1: Software and Steps
- Step 2: Hardware
- Step 3: General Cluster Settings
- **Step 4: Security**

Security Options

EC2 key pair tolu_brainstation

☑ Cluster visible to all IAM users in account

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   □ Use EMR_DefaultRole_V2

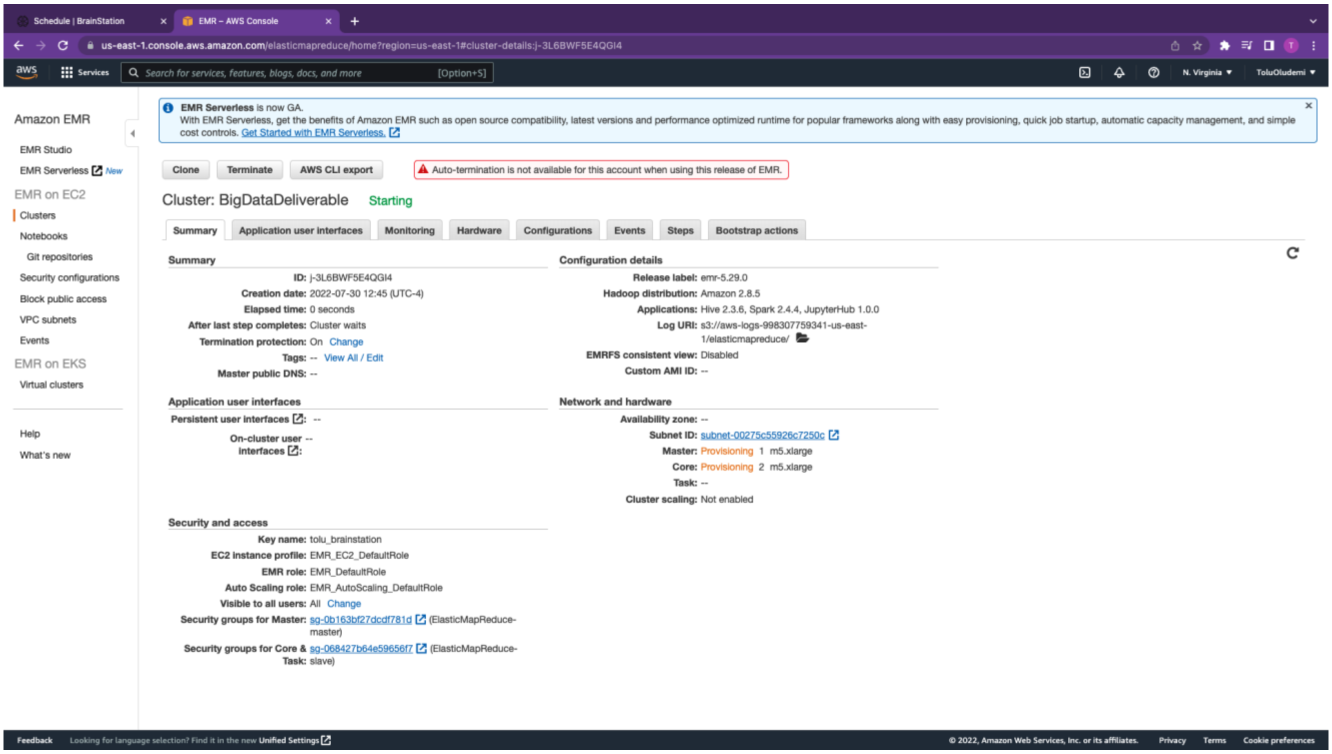
EC2 instance profile EMR_EC2_DefaultRole

Auto Scaling role EMR_AutoScaling_DefaultRole

▸ Security Configuration

▸ EC2 security groups

Cancel
Previous
Create cluster
Feedback
Looking for language selection? Find it in the new Unified Settings
© 2022, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences



EMR Serverless is now GA.
With EMR Serverless, get the benefits of Amazon EMR such as open source compatibility, latest versions and performance optimized runtime for popular frameworks along with easy provisioning, quick job startup, automatic capacity management, and simple cost controls. Get Started with EMR Serverless.

Auto-termination is not available for this account when using this release of EMR.

Cluster: BigDataDeliverable   Starting

Summary

ID: j-3L6BWF5E4QGI4
Creation date: 2022-07-30 12:45 (UTC-4)
Elapsed time: 0 seconds
After last step completes: Cluster waits
Termination protection: On Change
Tags: -- View All / Edit
Master public DNS: --

Configuration details

Release label: emr-5.29.0
Hadoop distribution: Amazon 2.8.5
Applications: Hive 2.3.6, Spark 2.4.4, JupyterHub 1.0.0
Log URI: s3://aws-logs-998307759341-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled
Custom AMI ID: --

Application user interfaces

Persistent user interfaces: --
On-cluster user interfaces: --

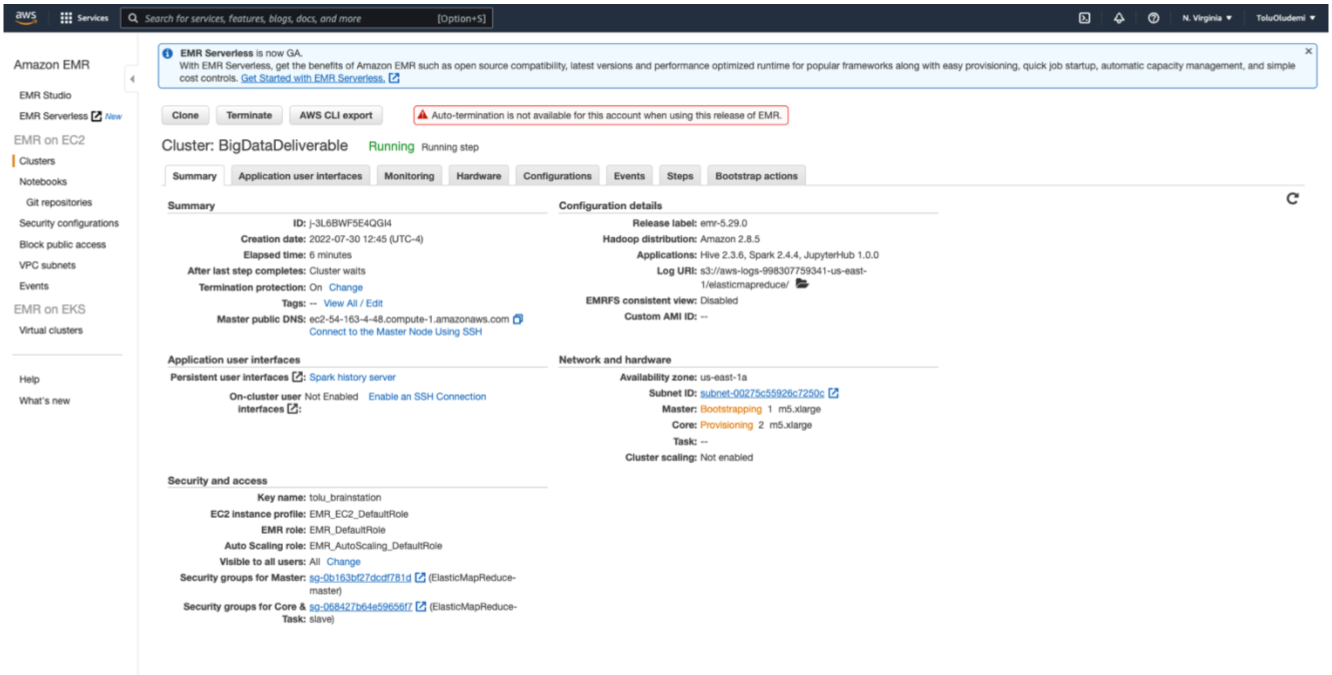
Network and hardware

Availability zone: --
Subnet ID: subnet-00275c55926c7250c
Master: Provisioning 1 m5.xlarge
Core: Provisioning 2 m5.xlarge
Task: --
Cluster scaling: Not enabled

Security and access

Key name: tolu_brainstation
EC2 instance profile: EMR_EC2_DefaultRole
EMR role: EMR_DefaultRole
Auto Scaling role: EMR_AutoScaling_DefaultRole
Visible to all users: All Change
Security groups for Master: sg-0b163bf27dcdf781d (ElasticMapReduce-master)
Security groups for Core & Task: sg-068427b64e59656f7 (ElasticMapReduce-slave)

Feedback
Looking for language selection? Find it in the new Unified Settings
© 2022, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences



EMR Serverless is now GA.
With EMR Serverless, get the benefits of Amazon EMR such as open source compatibility, latest versions and performance optimized runtime for popular frameworks along with easy provisioning, quick job startup, automatic capacity management, and simple cost controls. Get Started with EMR Serverless.

Auto-termination is not available for this account when using this release of EMR.

Cluster: BigDataDeliverable Running Running step

Summary

ID: j-3L6BWF5E4QGI4
Creation date: 2022-07-30 12:45 (UTC-4)
Elapsed time: 6 minutes
After last step completes: Cluster waits
Termination protection: On Change
Tags: -- View All / Edit
Master public DNS: ec2-54-163-4-48.compute-1.amazonaws.com
Connect to the Master Node Using SSH

Configuration details

Release label: emr-5.29.0
Hadoop distribution: Amazon 2.8.5
Applications: Hive 2.3.6, Spark 2.4.4, JupyterHub 1.0.0
Log URI: s3://aws-logs-998307759341-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled
Custom AMI ID: --

Application user interfaces

Persistent user interfaces: Spark history server
On-cluster user interfaces: Not Enabled Enable an SSH Connection

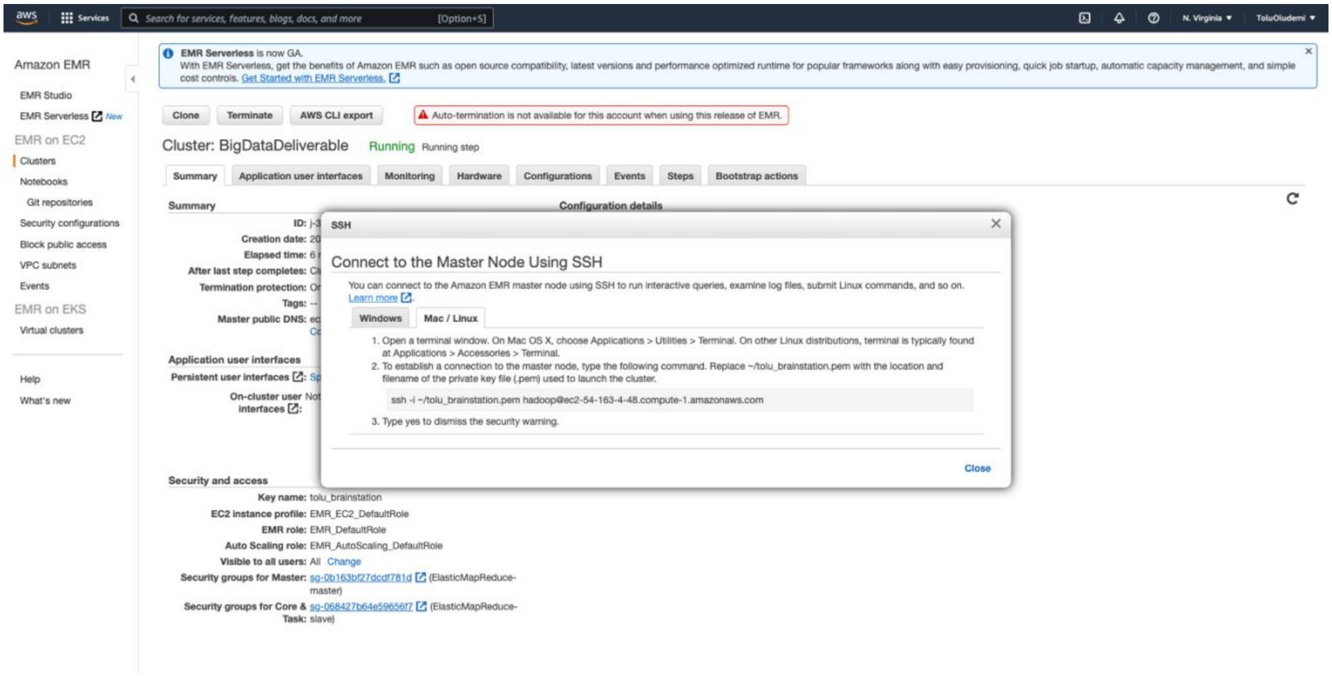
Network and hardware

Availability zone: us-east-1a
Subnet ID: subnet-00275c55926c7250c
Master: Bootstrapping 1 m5.xlarge
Core: Provisioning 2 m5.xlarge
Task: --
Cluster scaling: Not enabled

Security and access

Key name: tolu_brainstation
EC2 instance profile: EMR_EC2_DefaultRole
EMR role: EMR_DefaultRole
Auto Scaling role: EMR_AutoScaling_DefaultRole
Visible to all users: All Change
Security groups for Master: sg-0b163bf27dcdf781d (ElasticMapReduce-master)
Security groups for Core & Task: sg-068427b64e59656f7 (ElasticMapReduce-slave)

QUESTION 2 SCREENSHOTS



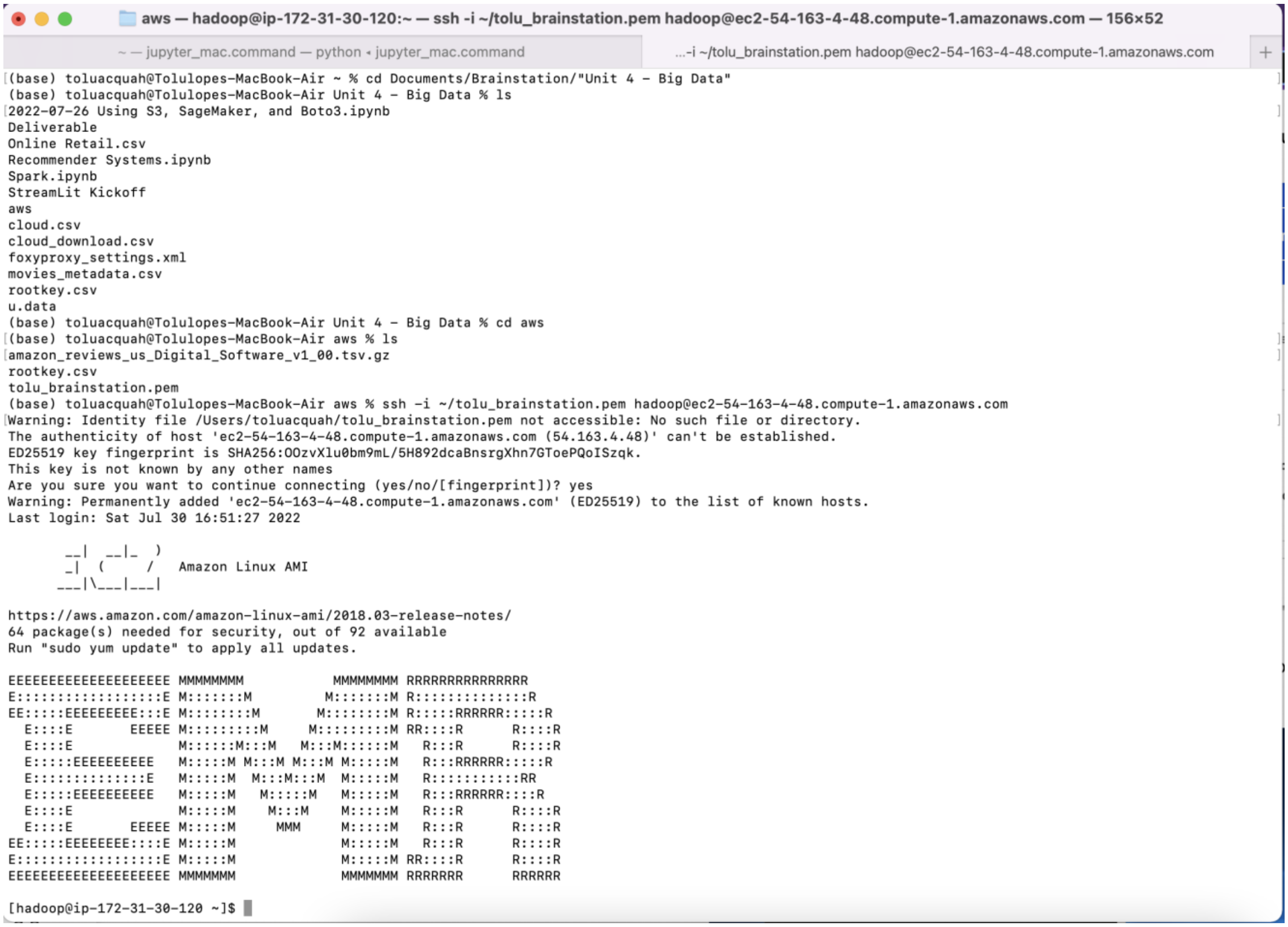
Connect to the Master Node Using SSH

You can connect to the Amazon EMR master node using SSH to run interactive queries, examine log files, submit Linux commands, and so on. Learn more.

1. Open a terminal window. On Mac OS X, choose Applications > Utilities > Terminal. On other Linux distributions, terminal is typically found at Applications > Accessories > Terminal.
2. To establish a connection to the master node, type the following command. Replace ~/tolu_brainstation.pem with the location and filename of the private key file (.pem) used to launch the cluster.

```
ssh -i ~/tolu_brainstation.pem hadoop@ec2-54-163-4-48.compute-1.amazonaws.com
```

3. Type yes to dismiss the security warning.



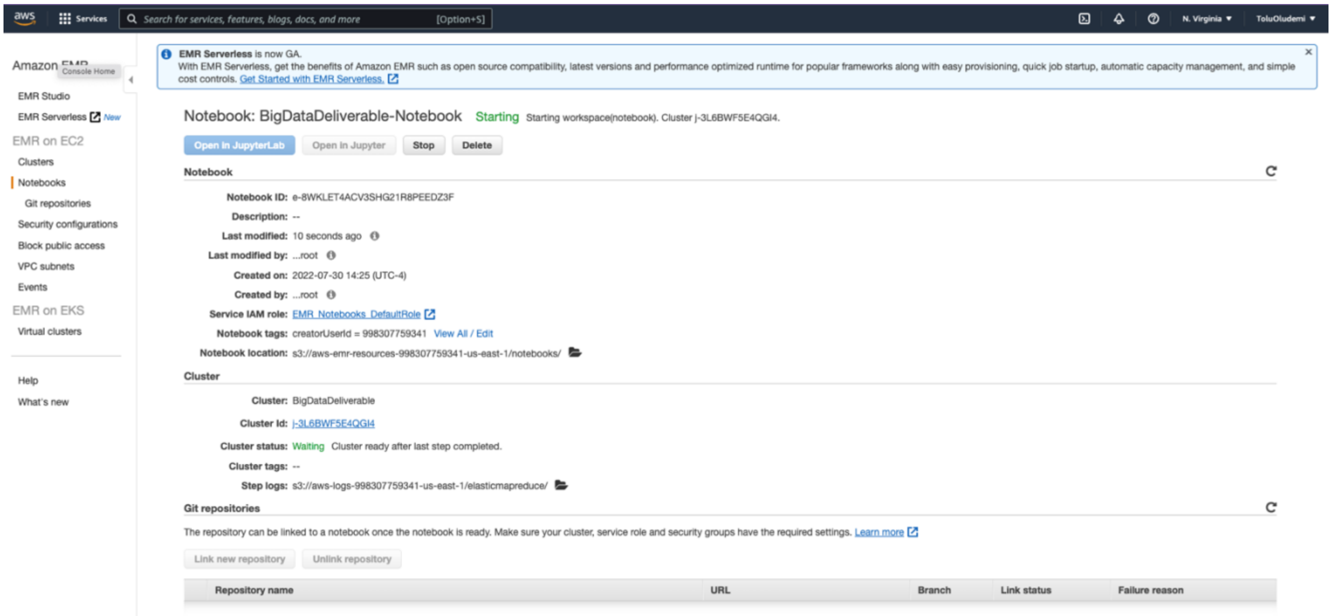
aws — hadoop@ip-172-31-30-120:~ — ssh -i ~/tolu_brainstation.pem hadoop@ec2-54-163-4-48.compute-1.amazonaws.com — 156×52

~ — jupyter_mac.command — python ◂ jupyter_mac.command

...-i ~/tolu_brainstation.pem hadoop@ec2-54-163-4-48.compute-1.amazonaws.com

```
(base) toluacquah@Tolulopes-MacBook-Air ~ % cd Documents/Brainstation/"Unit 4 - Big Data"
(base) toluacquah@Tolulopes-MacBook-Air Unit 4 - Big Data % ls
2022-07-26 Using S3, SageMaker, and Boto3.ipynb
Deliverable
Online Retail.csv
Recommender Systems.ipynb
Spark.ipynb
StreamLit Kickoff
aws
cloud.csv
cloud_download.csv
foxyproxy_settings.xml
movies_metadata.csv
rootkey.csv
u.data
(base) toluacquah@Tolulopes-MacBook-Air Unit 4 - Big Data % cd aws
(base) toluacquah@Tolulopes-MacBook-Air aws % ls
amazon_reviews_us_Digital_Software_v1_00.tsv.gz
rootkey.csv
tolu_brainstation.pem
(base) toluacquah@Tolulopes-MacBook-Air aws % ssh -i ~/tolu_brainstation.pem hadoop@ec2-54-163-4-48.compute-1.amazonaws.com
Warning: Identity file /Users/toluacquah/tolu_brainstation.pem not accessible: No such file or directory.
The authenticity of host 'ec2-54-163-4-48.compute-1.amazonaws.com (54.163.4.48)' can't be established.
ED25519 key fingerprint is SHA256:OOzvXlu0bm9mL/5H892dcaBnsrgXhn7GToePQoISzqk.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-163-4-48.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Last login: Sat Jul 30 16:51:27 2022

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
64 package(s) needed for security, out of 92 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEEE MMMMMMMM           MMMMMMMM RRRRRRRRRRRRRRR
E::::::::::::::::::E M:::::::M         M:::::::M R::::::::::::::R
EE:::::EEEEEEEEE:::E M::::::::M       M::::::::M R:::::RRRRRR:::::R
  E::::E       EEEEE M:::::::::M     M:::::::::M RR::::R      R::::R
  E::::E             M::::::M:::M   M:::M::::::M   R:::R      R::::R
  E:::::EEEEEEEEEE   M:::::M M:::M M:::M M:::::M   R:::RRRRRR:::::R
  E::::::::::::::E   M:::::M  M:::M:::M  M:::::M   R:::::::::::RR
  E:::::EEEEEEEEEE   M:::::M   M:::::M   M:::::M   R:::RRRRRR::::R
  E::::E             M:::::M    M:::M    M:::::M   R:::R      R::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R:::R      R::::R
EE:::::EEEEEEEE::::E M:::::M             M:::::M   R:::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-30-120 ~]$
```

QUESTION 4 SCREENSHOT



EMR Serverless is now GA.
With EMR Serverless, get the benefits of Amazon EMR such as open source compatibility, latest versions and performance optimized runtime for popular frameworks along with easy provisioning, quick job startup, automatic capacity management, and simple cost controls. Get Started with EMR Serverless.

Notebook: BigDataDeliverable-Notebook Starting Starting workspace(notebook). Cluster j-3L6BWF5E4QGI4.

Notebook

Notebook ID: e-8WKLET4ACV3SHG21R8PEEDZ3F
Description: --
Last modified: 10 seconds ago
Last modified by: ...root
Created on: 2022-07-30 14:25 (UTC-4)
Created by: ...root
Service IAM role: EMR_Notebooks_DefaultRole
Notebook tags: creatorUserId = 998307759341 View All / Edit
Notebook location: s3://aws-emr-resources-998307759341-us-east-1/notebooks/

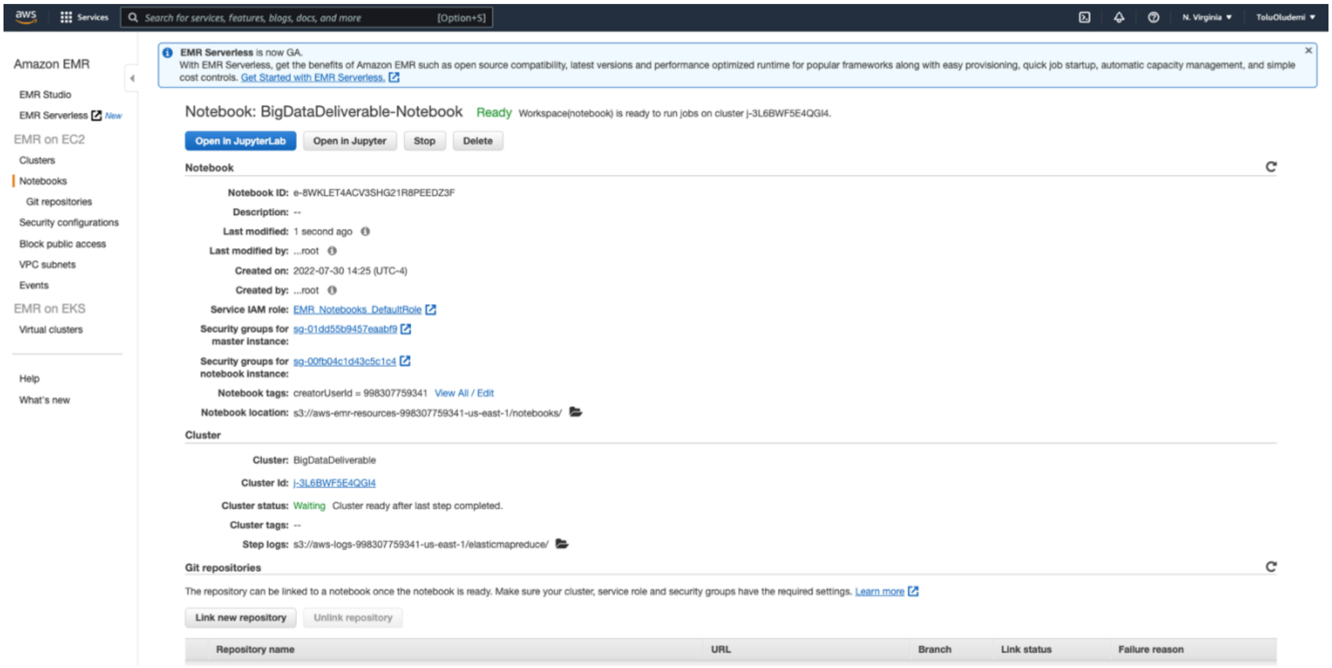
Cluster

Cluster: BigDataDeliverable
Cluster Id: j-3L6BWF5E4QGI4
Cluster status: Waiting Cluster ready after last step completed.
Cluster tags: --
Step logs: s3://aws-logs-998307759341-us-east-1/elasticmapreduce/

Git repositories

The repository can be linked to a notebook once the notebook is ready. Make sure your cluster, service role and security groups have the required settings. Learn more

Repository name	URL	Branch	Link status	Failure reason



EMR Serverless is now GA.
With EMR Serverless, get the benefits of Amazon EMR such as open source compatibility, latest versions and performance optimized runtime for popular frameworks along with easy provisioning, quick job startup, automatic capacity management, and simple cost controls. Get Started with EMR Serverless.

Notebook: BigDataDeliverable-Notebook Ready Workspace(notebook) is ready to run jobs on cluster j-3L6BWF5E4QGI4.

Notebook

Notebook ID: e-8WKLET4ACV3SHG21R8PEEDZ3F
Description: --
Last modified: 1 second ago
Last modified by: ...root
Created on: 2022-07-30 14:25 (UTC-4)
Created by: ...root
Service IAM role: EMR_Notebooks_DefaultRole
Security groups for master instance: sg-01dd55b9457eaabf9
Security groups for notebook instance: sg-00fb04c1d43c5c1c4
Notebook tags: creatorUserId = 998307759341 View All / Edit
Notebook location: s3://aws-emr-resources-998307759341-us-east-1/notebooks/

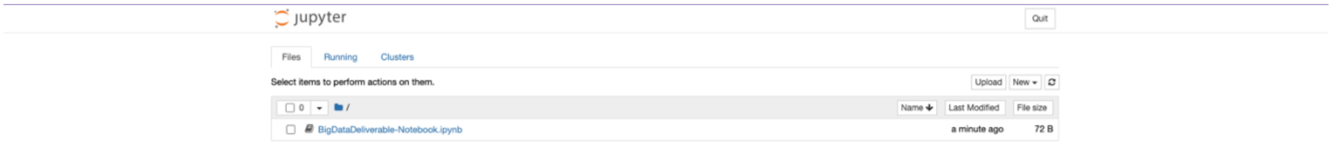
Cluster

Cluster: BigDataDeliverable
Cluster Id: j-3L6BWF5E4QGI4
Cluster status: Waiting Cluster ready after last step completed.
Cluster tags: --
Step logs: s3://aws-logs-998307759341-us-east-1/elasticmapreduce/

Git repositories

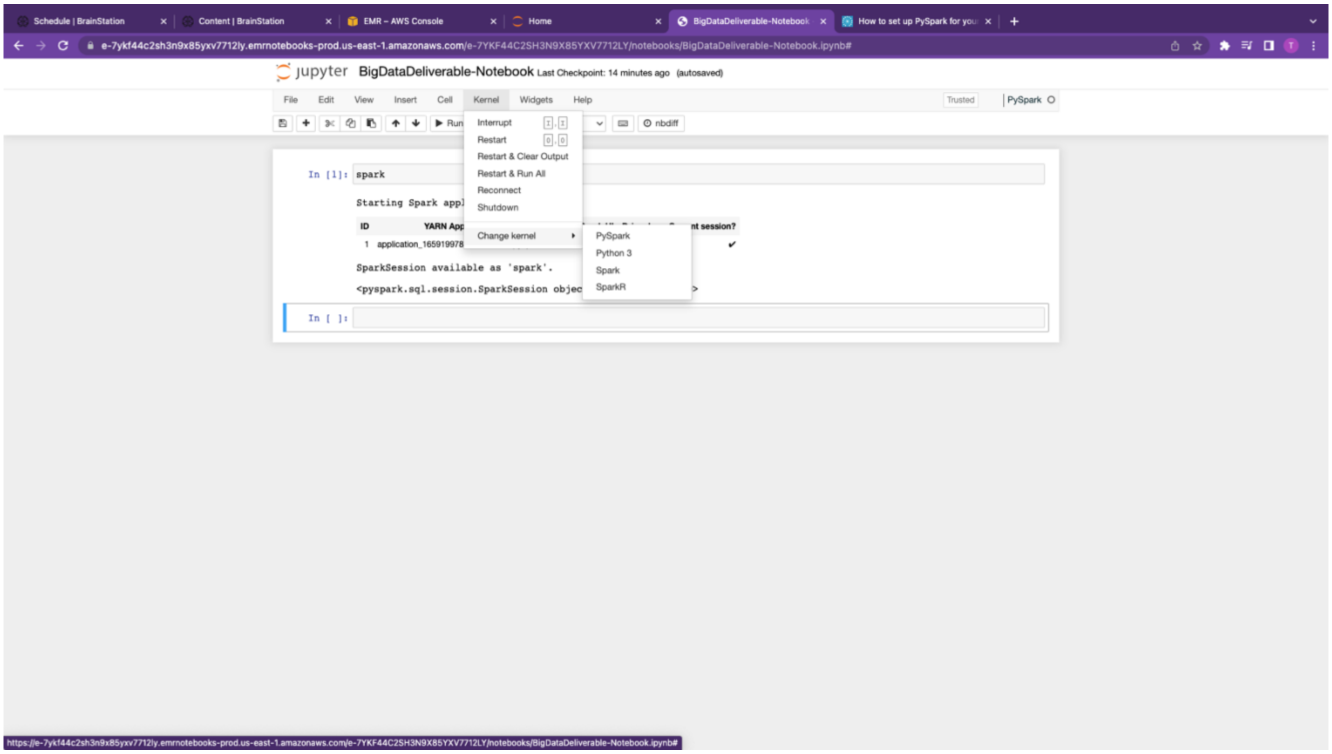
The repository can be linked to a notebook once the notebook is ready. Make sure your cluster, service role and security groups have the required settings. Learn more

Repository name	URL	Branch	Link status	Failure reason

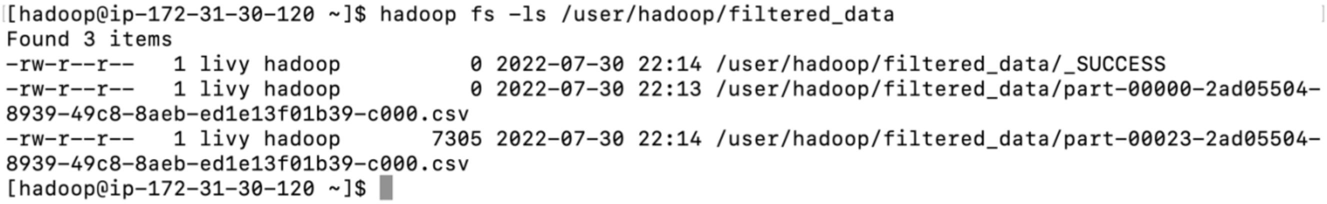


Select items to perform actions on them.

☐ 0 /		
☐ BigDataDeliverable-Notebook.ipynb	a minute ago	72 B



```
In [1]: spark

Starting Spark appl
ID    YARN App                                   nt session?
 1   application_165919978                          ✔

SparkSession available as 'spark'.

<pyspark.sql.session.SparkSession objec                  >

In [ ]:
```



```
[hadoop@ip-172-31-30-120 ~]$ hadoop fs -ls /user/hadoop/filtered_data
Found 3 items
-rw-r--r--   1 livy hadoop          0 2022-07-30 22:14 /user/hadoop/filtered_data/_SUCCESS
-rw-r--r--   1 livy hadoop          0 2022-07-30 22:13 /user/hadoop/filtered_data/part-00000-2ad05504-
8939-49c8-8aeb-ed1e13f01b39-c000.csv
-rw-r--r--   1 livy hadoop       7305 2022-07-30 22:14 /user/hadoop/filtered_data/part-00023-2ad05504-
8939-49c8-8aeb-ed1e13f01b39-c000.csv
[hadoop@ip-172-31-30-120 ~]$ 
```