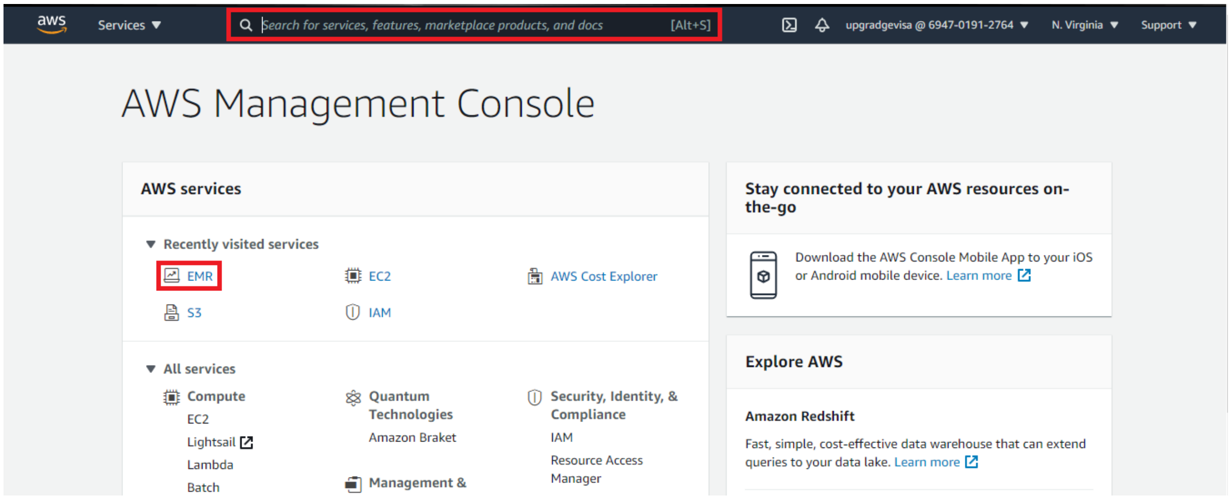
Case Study Report

CLUSTER CREATION:

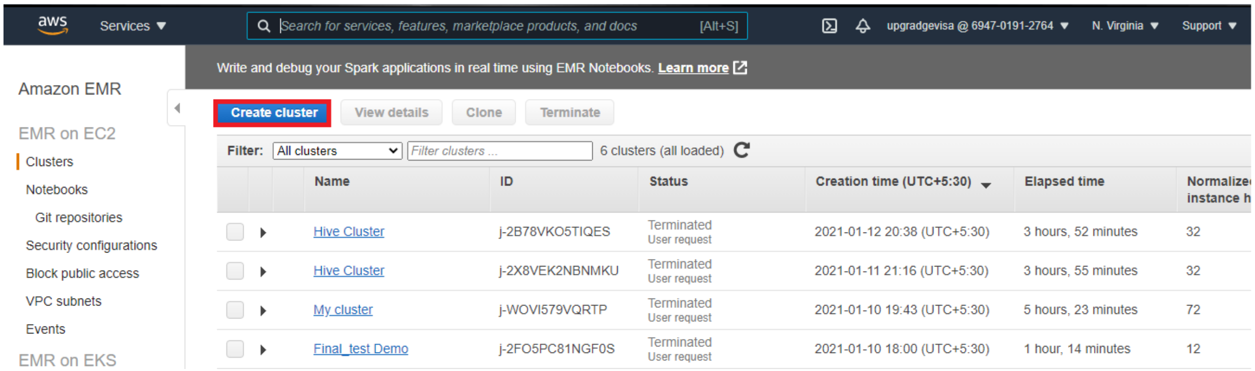
Step 1: Open the AWS console and either search for EMR in the search tab or click on the EMR button highlighted which is in the recently used tab.



Download the AWS Console Mobile App to your iOS or Android mobile device. Learn more

Fast, simple, cost-effective data warehouse that can extend queries to your data lake. Learn more

Step 2: In Amazon EMR home page click on create cluster button.

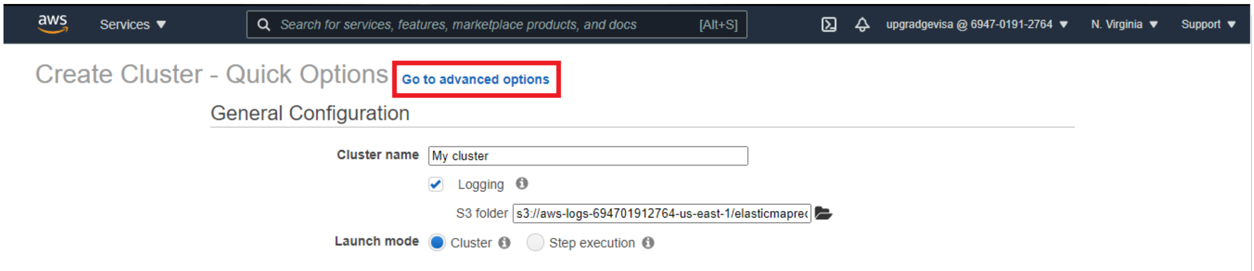


Write and debug your Spark applications in real time using EMR Notebooks. Learn more

Filter: All clusters ▾ Filter clusters ... 6 clusters (all loaded)

		Name	ID	Status	Creation time (UTC+5:30) ▼	Elapsed time	Normalized instance h
□	▸	Hive Cluster	j-2B78VKO5TIQES	Terminated User request	2021-01-12 20:38 (UTC+5:30)	3 hours, 52 minutes	32
□	▸	Hive Cluster	j-2X8VEK2NBNMKU	Terminated User request	2021-01-11 21:16 (UTC+5:30)	3 hours, 55 minutes	32
□	▸	My cluster	j-WOVI579VQRTP	Terminated User request	2021-01-10 19:43 (UTC+5:30)	5 hours, 23 minutes	72
□	▸	Final_test Demo	j-2FO5PC81NGF0S	Terminated User request	2021-01-10 18:00 (UTC+5:30)	1 hour, 14 minutes	12

Step 3: Then click on ‘Go to advanced options’ in the next page


aws
Services ▼
Search for services, features, marketplace products, and docs [Alt+S]
upgradgevisa @ 6947-0191-2764 ▼
N. Virginia ▼
Support ▼
Create Cluster - Quick Options
Go to advanced options
General Configuration

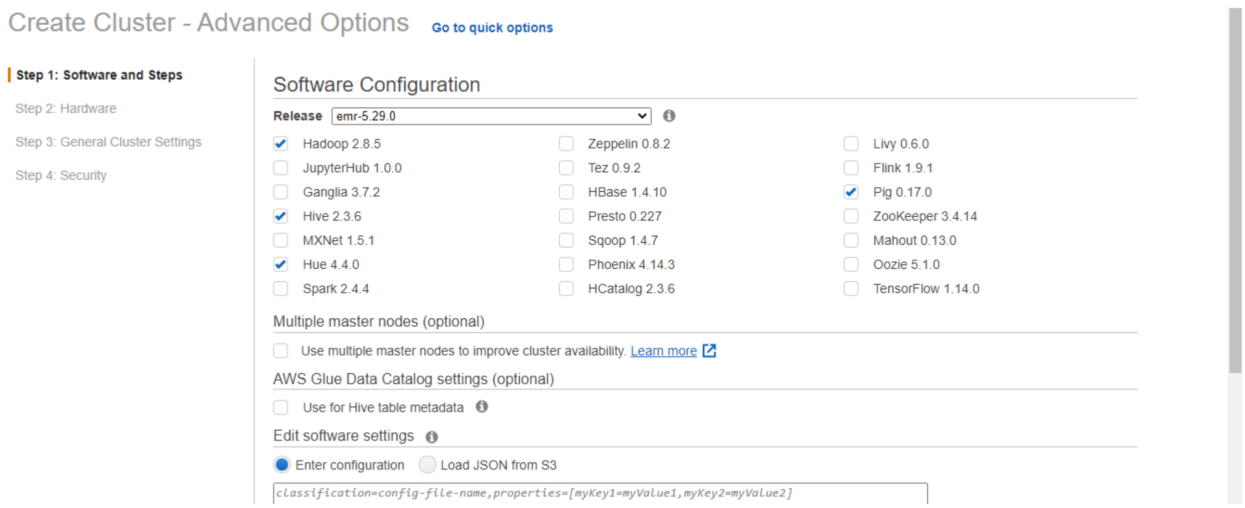
Cluster name: My cluster

☑ Logging

S3 folder: s3://aws-logs-694701912764-us-east-1/elasticmaprec

Launch mode: ● Cluster ○ Step execution

Step 4: In software configuration page we have to select all the required services needed to complete this analysis. We are selecting emr-5.29.0 release as per the instruction with required services.



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)

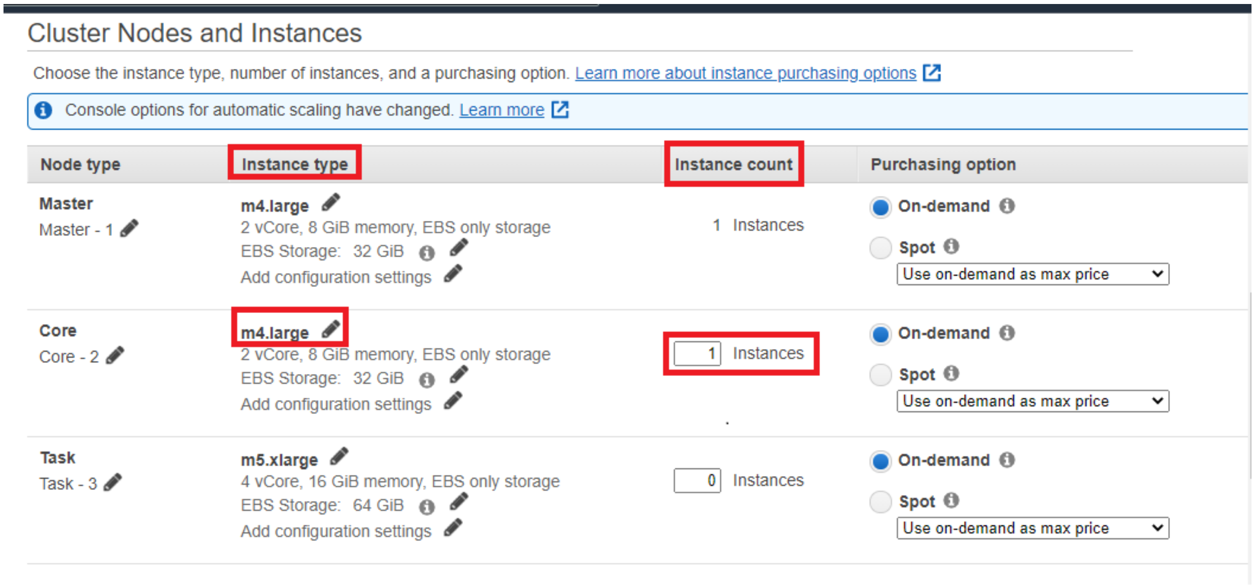
□ Use for Hive table metadata

Edit software settings

● Enter configuration ○ Load JSON from S3

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

Step 5: Then we will proceed to next page in this we have to specify the required configuration needed for the cluster. Here we have to select the instance type and instance count for master and core node. We have selected instance type as m4.large and 1 instance count for both master and core node as we need a 2-node cluster for the analysis



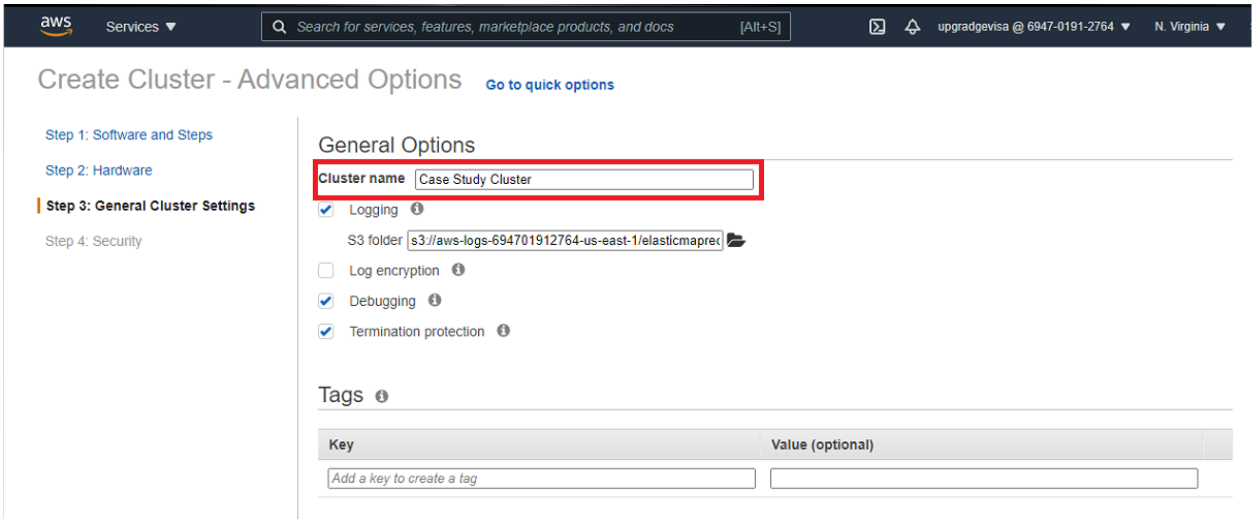
Cluster Nodes and Instances

Choose the instance type, number of instances, and a purchasing option. Learn more about instance purchasing options

Console options for automatic scaling have changed. Learn more

Node type	Instance type	Instance count	Purchasing option
Master Master - 1	m4.large 2 vCore, 8 GiB memory, EBS only storage EBS Storage: 32 GiB Add configuration settings	1 Instances	● On-demand ○ Spot Use on-demand as max price
Core Core - 2	m4.large 2 vCore, 8 GiB memory, EBS only storage EBS Storage: 32 GiB Add configuration settings	1 Instances	● On-demand ○ Spot Use on-demand as max price
Task Task - 3	m5.xlarge 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 64 GiB Add configuration settings	0 Instances	● On-demand ○ Spot Use on-demand as max price

Step 6: Here we have given the name of the cluster.


aws
Services ▼
Search for services, features, marketplace products, and docs [Alt+S]
upgradgevisa @ 6947-0191-2764 ▼
N. Virginia ▼

Create Cluster - Advanced Options Go to quick options

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

General Options

Cluster name Case Study Cluster

☑ Logging

S3 folder s3://aws-logs-694701912764-us-east-1/elasticmaprec

□ Log encryption

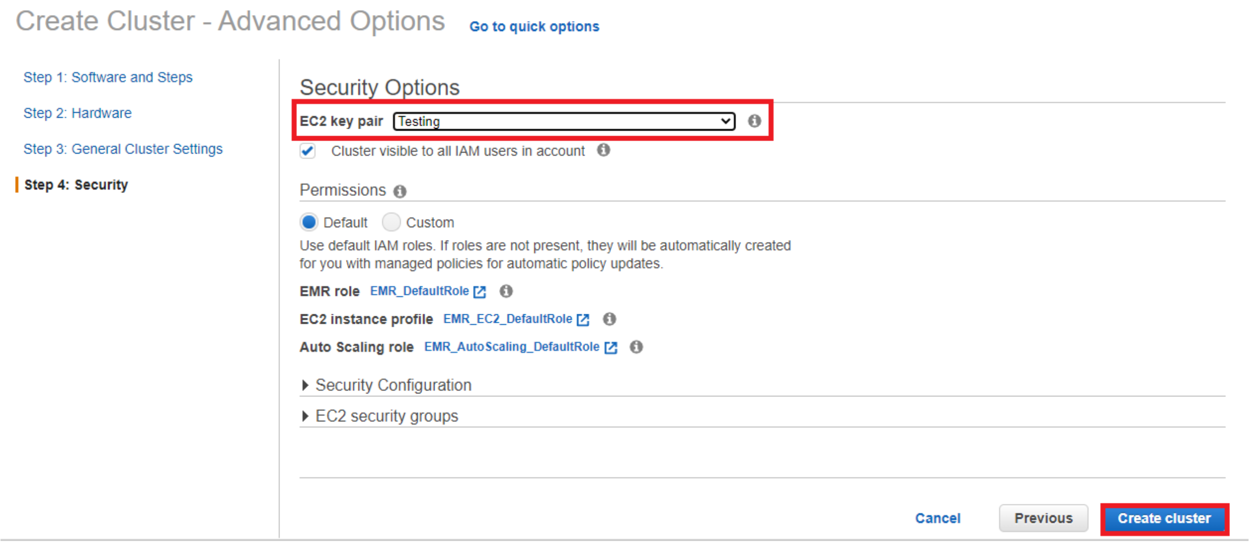
☑ Debugging

☑ Termination protection

Tags

Key	Value (optional)
Add a key to create a tag	

Step 7: We have selected the created EC2 key pair and then click on the create cluster option.



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 Testing

☑ 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

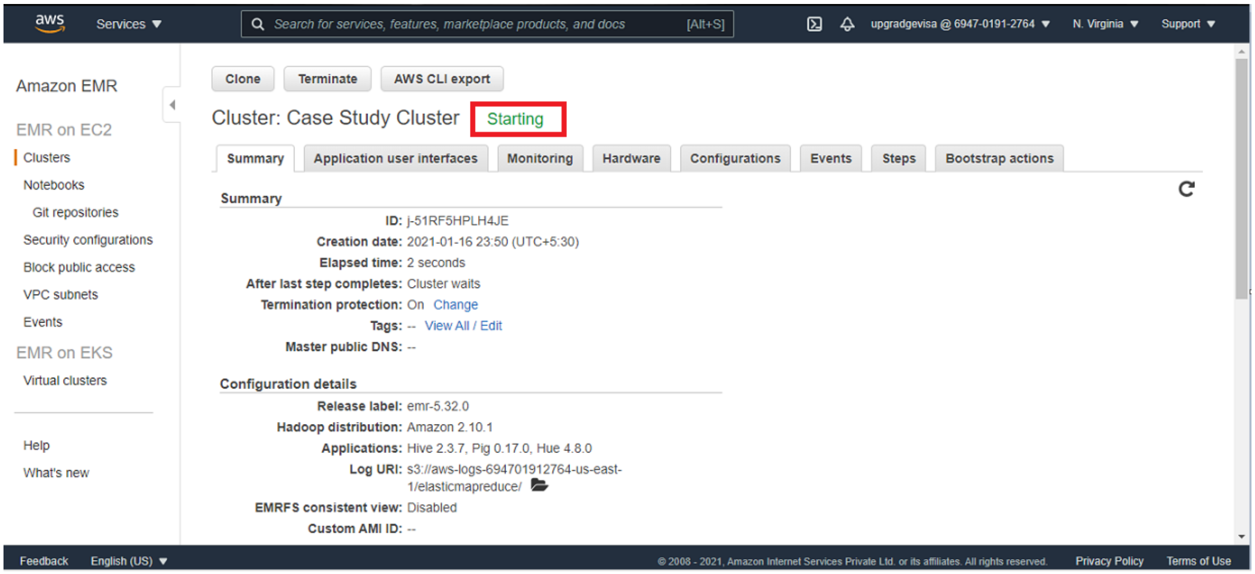
EC2 instance profile EMR_EC2_DefaultRole

Auto Scaling role EMR_AutoScaling_DefaultRole

▸ Security Configuration

▸ EC2 security groups

Step 8: Then cluster has been created and will be in the “Starting” state and will change to “Running “ state after sometime.



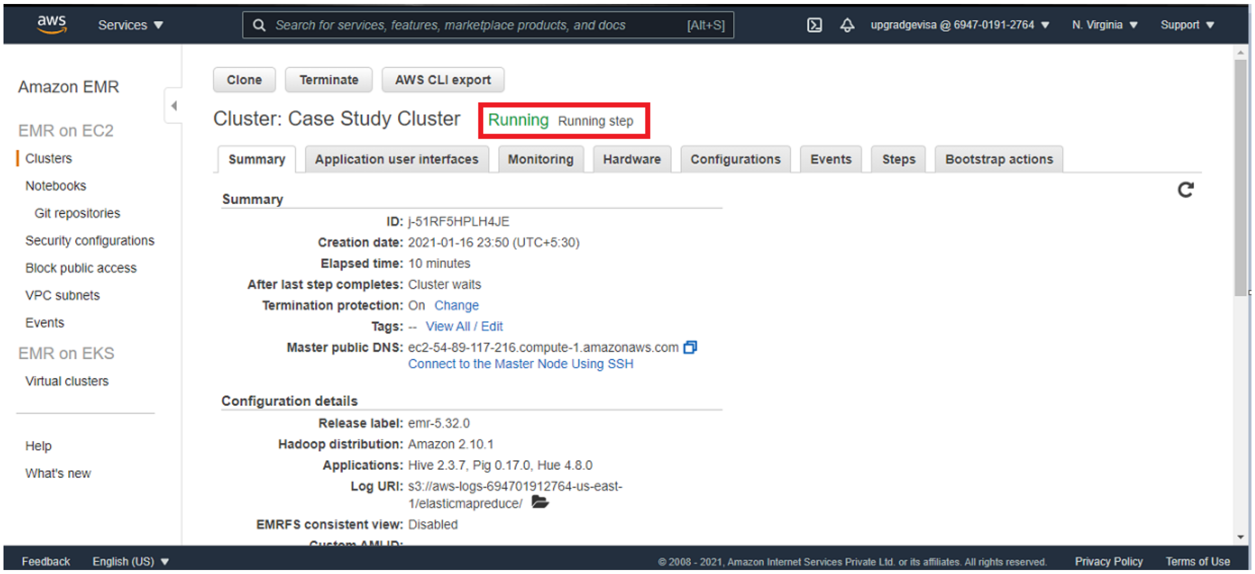
Cluster: Case Study Cluster Starting

Summary

ID: j-51RF5HPLH4JE
Creation date: 2021-01-16 23:50 (UTC+5:30)
Elapsed time: 2 seconds
After last step completes: Cluster waits
Termination protection: On Change
Tags: -- View All / Edit
Master public DNS: --

Configuration details

Release label: emr-5.32.0
Hadoop distribution: Amazon 2.10.1
Applications: Hive 2.3.7, Pig 0.17.0, Hue 4.8.0
Log URI: s3://aws-logs-694701912764-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled
Custom AMI ID: --



Cluster: Case Study Cluster Running Running step

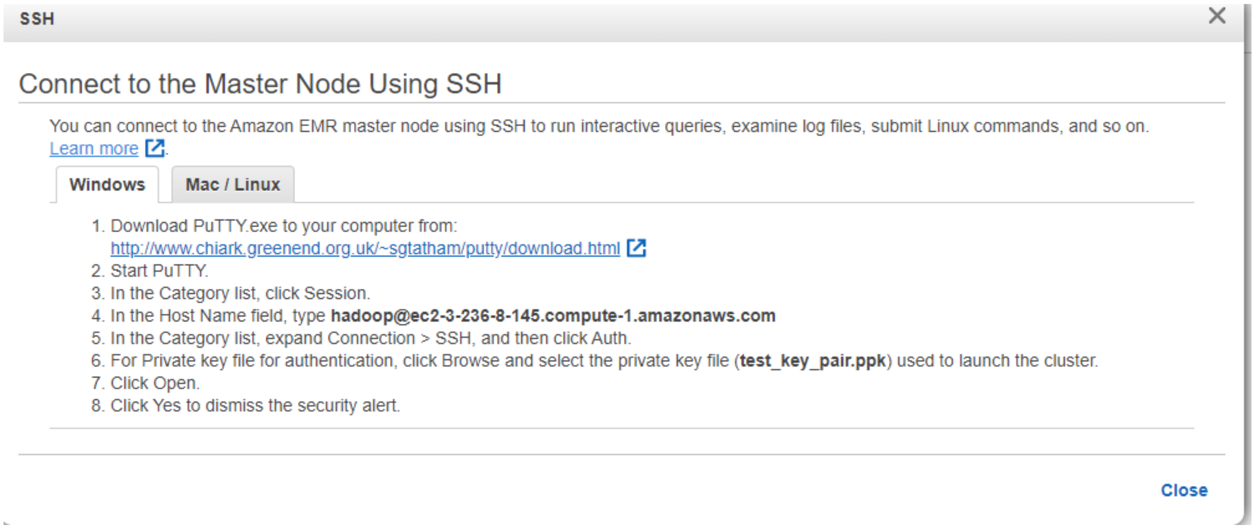
Summary

ID: j-51RF5HPLH4JE
Creation date: 2021-01-16 23:50 (UTC+5:30)
Elapsed time: 10 minutes
After last step completes: Cluster waits
Termination protection: On Change
Tags: -- View All / Edit
Master public DNS: ec2-54-89-117-216.compute-1.amazonaws.com
Connect to the Master Node Using SSH

Configuration details

Release label: emr-5.32.0
Hadoop distribution: Amazon 2.10.1
Applications: Hive 2.3.7, Pig 0.17.0, Hue 4.8.0
Log URI: s3://aws-logs-694701912764-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled

Step 9: Once cluster in running state we have to click on Master public DNS.



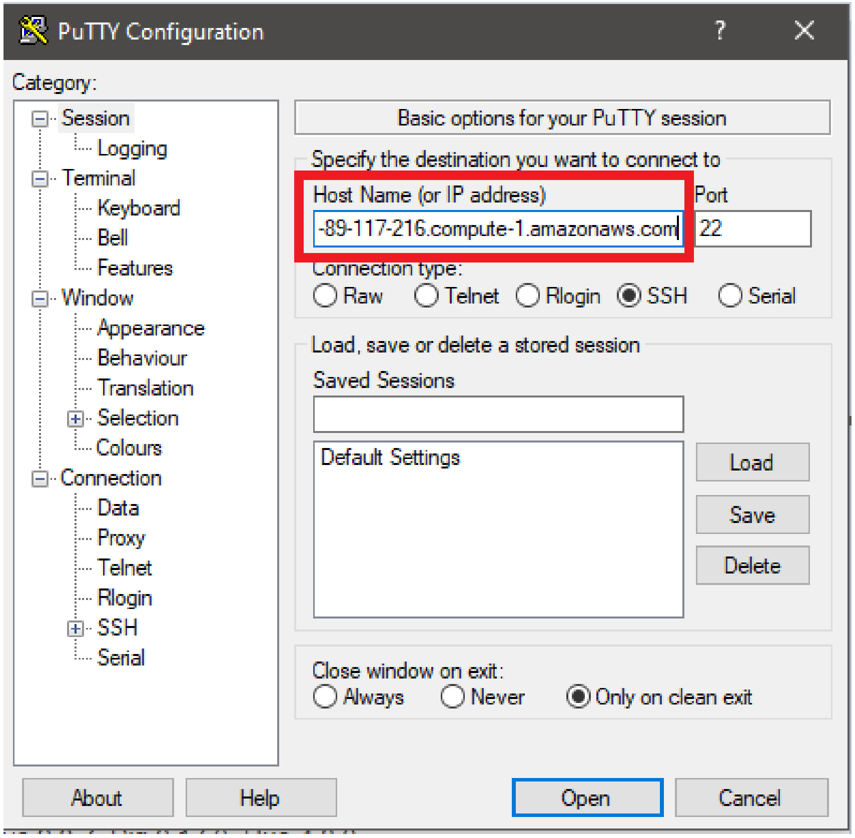
SSH

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. Download PuTTY.exe to your computer from: http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html
2. Start PuTTY.
3. In the Category list, click Session.
4. In the Host Name field, type **hadoop@ec2-3-236-8-145.compute-1.amazonaws.com**
5. In the Category list, expand Connection > SSH, and then click Auth.
6. For Private key file for authentication, click Browse and select the private key file (**test_key_pair.ppk**) used to launch the cluster.
7. Click Open.
8. Click Yes to dismiss the security alert.

Step 10: We have to open the putty configuration and then give the host name(master node DNS) and then browse to the private key file location by clicking on Connection → SSH → Auth.



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): -89-117-216.compute-1.amazonaws.com

Port: 22

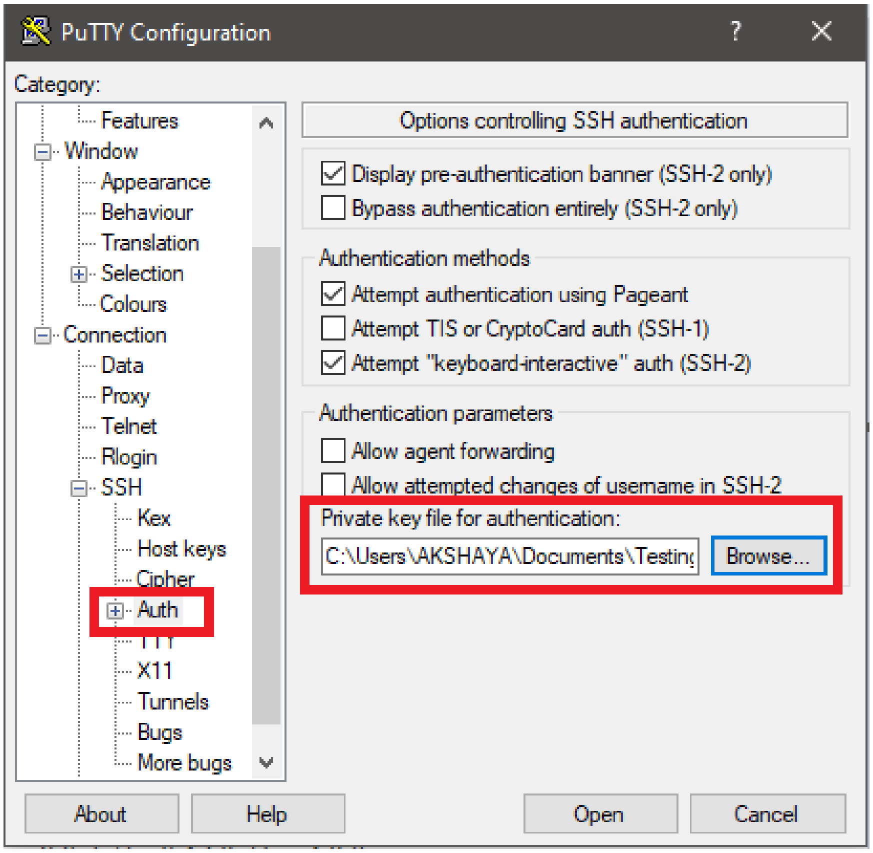
Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit



Options controlling SSH authentication

☑ Display pre-authentication banner (SSH-2 only)
☐ Bypass authentication entirely (SSH-2 only)

Authentication methods

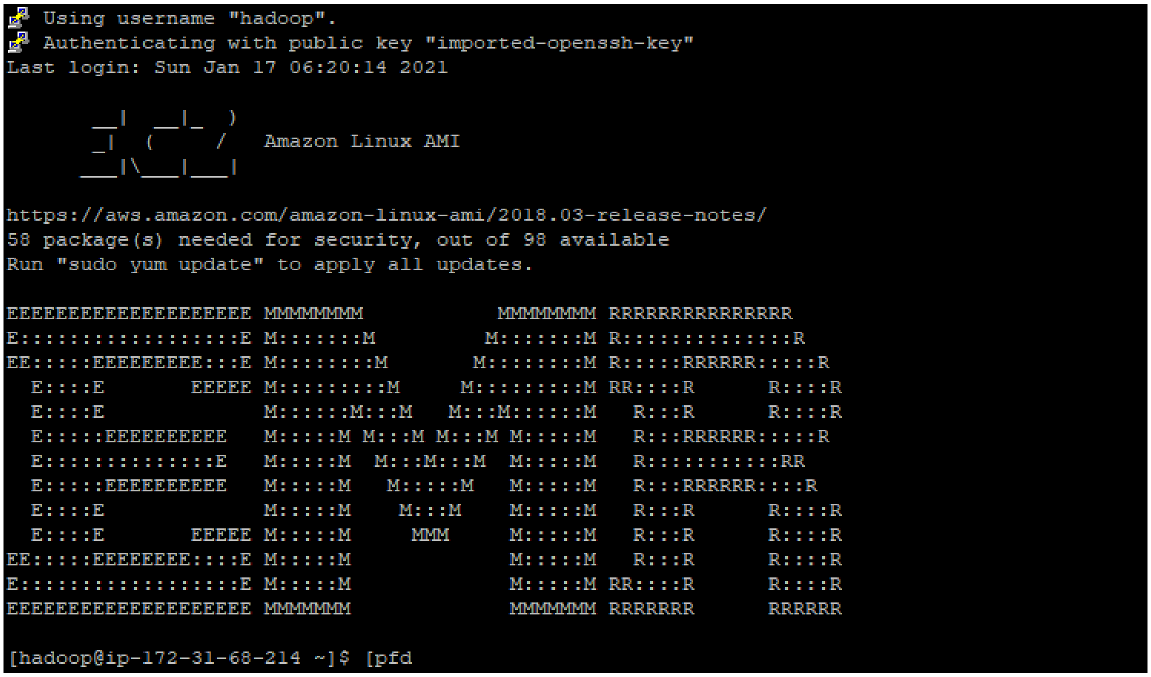
☑ Attempt authentication using Pageant
☐ Attempt TIS or CryptoCard auth (SSH-1)
☑ Attempt "keyboard-interactive" auth (SSH-2)

Authentication parameters

☐ Allow agent forwarding
☐ Allow attempted changes of username in SSH-2

Private key file for authentication:
C:\Users\AKSHAYA\Documents\Testin Browse...

Step 11: Then EMR command line interface will be open and login as Hadoop .

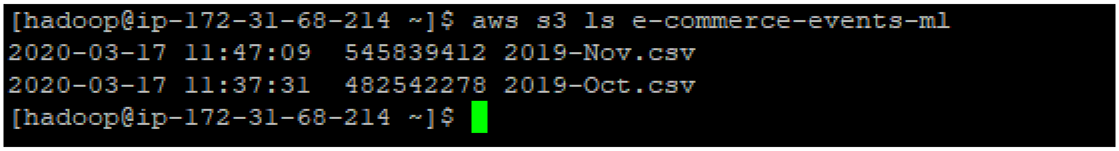


```
 Using username "hadoop".
 Authenticating with public key "imported-openssh-key"
Last login: Sun Jan 17 06:20:14 2021

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

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
58 package(s) needed for security, out of 98 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-172-31-68-214 ~]$ [pfd
```

Step 12: Copying the data into HDFS

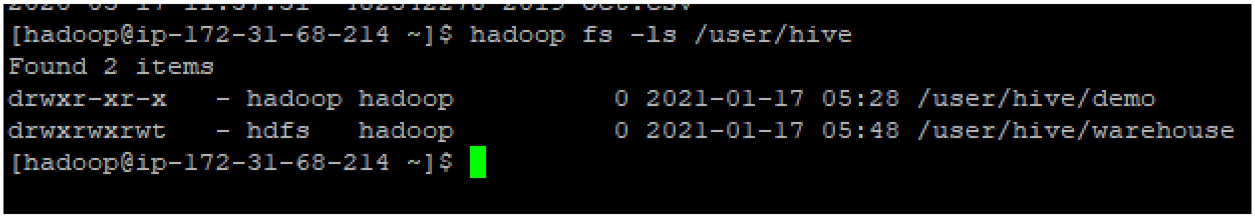
a. Creating a directory in HDFS.
Command: hadoop fs -mkdir /user/hive/demo

b. To access the public s3 bucket.
Command: aws s3 ls e-commerce-events-ml



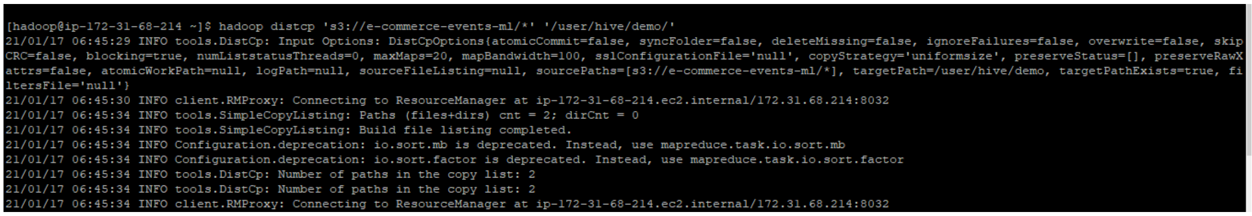
```
[hadoop@ip-172-31-68-214 ~]$ aws s3 ls e-commerce-events-ml
2020-03-17 11:47:09  545839412 2019-Nov.csv
2020-03-17 11:37:31  482542278 2019-Oct.csv
[hadoop@ip-172-31-68-214 ~]$
```

c. Checking the available directory.
Command: hadoop fs -ls /user/hive

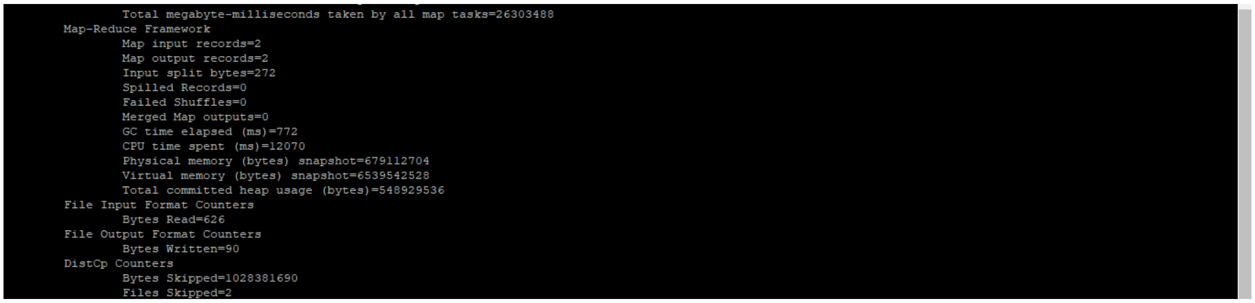


```
2020-03-17 11:37:31  482542278 2019-Oct.csv
[hadoop@ip-172-31-68-214 ~]$ hadoop fs -ls /user/hive
Found 2 items
drwxr-xr-x   - hadoop hadoop          0 2021-01-17 05:28 /user/hive/demo
drwxrwxrwt   - hdfs   hadoop          0 2021-01-17 05:48 /user/hive/warehouse
[hadoop@ip-172-31-68-214 ~]$
```

d. Loading the s3 public data set to created directory “Demo” in hadoop .
Command: hadoop distcp 's3://e-commerce-events-ml/*' '/user/hive/demo/'

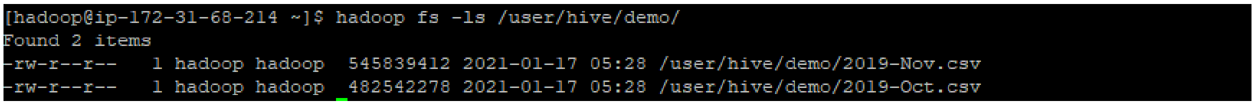


```
[hadoop@ip-172-31-68-214 ~]$ hadoop distcp 's3://e-commerce-events-ml/*' '/user/hive/demo/'
21/01/17 06:45:29 INFO tools.DistCp: Input Options: DistCpOptions{atomicCommit=false, syncFolder=false, deleteMissing=false, ignoreFailures=false, overwrite=false, skip
CRC=false, blocking=true, numListstatusThreads=0, maxMaps=20, mapBandwidth=100, sslConfigurationFile='null', copyStrategy='uniformsize', preserveStatus=[], preserveRawX
attrs=false, atomicWorkPath=null, logPath=null, sourceFileListing=null, sourcePaths=[s3://e-commerce-events-ml/*], targetPath=/user/hive/demo, targetPathExists=true, fi
ltersFile='null'}
21/01/17 06:45:30 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-68-214.ec2.internal/172.31.68.214:8032
21/01/17 06:45:34 INFO tools.SimpleCopyListing: Paths (files+dirs) cnt = 2; dirCnt = 0
21/01/17 06:45:34 INFO tools.SimpleCopyListing: Build file listing completed.
21/01/17 06:45:34 INFO Configuration.deprecation: io.sort.mb is deprecated. Instead, use mapreduce.task.io.sort.mb
21/01/17 06:45:34 INFO Configuration.deprecation: io.sort.factor is deprecated. Instead, use mapreduce.task.io.sort.factor
21/01/17 06:45:34 INFO tools.DistCp: Number of paths in the copy list: 2
21/01/17 06:45:34 INFO tools.DistCp: Number of paths in the copy list: 2
21/01/17 06:45:34 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-68-214.ec2.internal/172.31.68.214:8032
```



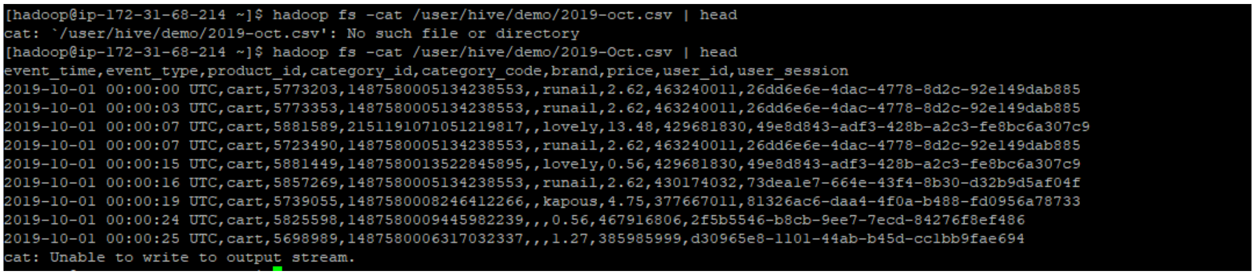
```
                Total megabyte-milliseconds taken by all map tasks=26303488
        Map-Reduce Framework
                Map input records=2
                Map output records=2
                Input split bytes=272
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=772
                CPU time spent (ms)=12070
                Physical memory (bytes) snapshot=679112704
                Virtual memory (bytes) snapshot=6539542528
                Total committed heap usage (bytes)=548929536
        File Input Format Counters
                Bytes Read=626
        File Output Format Counters
                Bytes Written=90
        DistCp Counters
                Bytes Skipped=1028381690
                Files Skipped=2
```

e. After loading the data set we have used below command to check the data set file in the hadoop directory.
   Command: hadoop fs -ls /user/hive/demo/



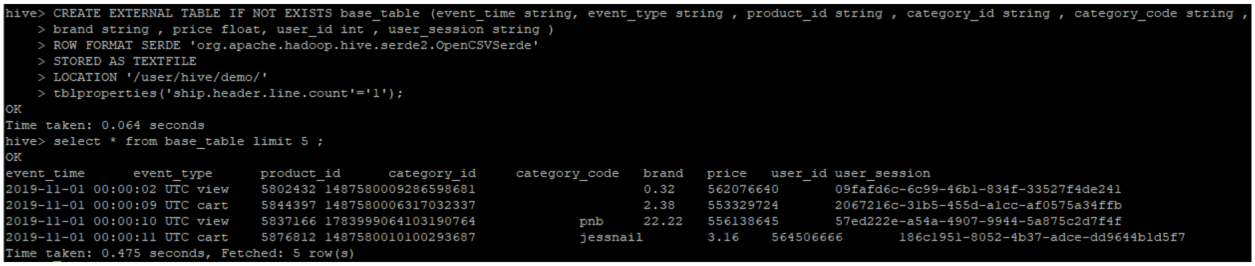
```
[hadoop@ip-172-31-68-214 ~]$ hadoop fs -ls /user/hive/demo/
Found 2 items
-rw-r--r--   1 hadoop hadoop  545839412 2021-01-17 05:28 /user/hive/demo/2019-Nov.csv
-rw-r--r--   1 hadoop hadoop  482542278 2021-01-17 05:28 /user/hive/demo/2019-Oct.csv
```

f. We have used below command to check the saved data set in the hadoop directory.
   Command: hadoop fs -cat /user/hive/demo/2019-oct.csv | head



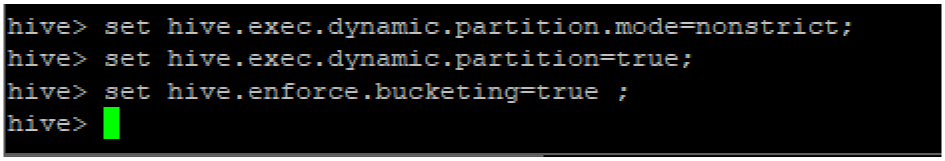
```
[hadoop@ip-172-31-68-214 ~]$ hadoop fs -cat /user/hive/demo/2019-oct.csv | head
cat: `/user/hive/demo/2019-oct.csv': No such file or directory
[hadoop@ip-172-31-68-214 ~]$ hadoop fs -cat /user/hive/demo/2019-Oct.csv | head
event_time,event_type,product_id,category_id,category_code,brand,price,user_id,user_session
2019-10-01 00:00:00 UTC,cart,5773203,1487580005134238553,,runail,2.62,463240011,26dd6e6e-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:03 UTC,cart,5773353,1487580005134238553,,runail,2.62,463240011,26dd6e6e-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:07 UTC,cart,5881589,2151191071051219817,,lovely,13.48,429681830,49e8d843-adf3-428b-a2c3-fe8bc6a307c9
2019-10-01 00:00:07 UTC,cart,5723490,1487580005134238553,,runail,2.62,463240011,26dd6e6e-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:15 UTC,cart,5881449,1487580013522845895,,lovely,0.56,429681830,49e8d843-adf3-428b-a2c3-fe8bc6a307c9
2019-10-01 00:00:16 UTC,cart,5857269,1487580005134238553,,runail,2.62,430174032,73dea1e7-664e-43f4-8b30-d32b9d5af04f
2019-10-01 00:00:19 UTC,cart,5739055,1487580008246412266,,kapous,4.75,377667011,81326ac6-daa4-4f0a-b488-fd0956a78733
2019-10-01 00:00:24 UTC,cart,5825598,1487580009445982239,,,0.56,467916806,2f5b5546-b8cb-9ee7-7ecd-84276f8ef486
2019-10-01 00:00:25 UTC,cart,5698989,1487580006317032337,,,1.27,385985999,d30965e8-1101-44ab-b45d-cc1bb9fae694
cat: Unable to write to output stream.
```

g. After moving the data to the directory we create the base table and check for the data in the table.
   Command:
   CREATE EXTERNAL TABLE IF NOT EXISTS base_table (event_time string, event_type string ,
   product_id string , category_id string , category_code string ,
   brand string , price float, user_id int , user_session string)
   ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
   STORED AS TEXTFILE
   LOCATION '/user/hive/demo/'
   tblproperties('ship.header.line.count'='1');



```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS base_table (event_time string, event_type string , product_id string , category_id string , category_code string ,
    > brand string , price float, user_id int , user_session string )
    > ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
    > STORED AS TEXTFILE
    > LOCATION '/user/hive/demo/'
    > tblproperties('ship.header.line.count'='1');
OK
Time taken: 0.064 seconds
hive> select * from base_table limit 5 ;
OK
event_time      event_type      product_id      category_id     category_code   brand   price   user_id user_session
2019-11-01 00:00:02 UTC view    5802432 1487580009286598681                     0.32    562076640       09fafd6c-6c99-46b1-834f-33527f4de241
2019-11-01 00:00:09 UTC cart    5844397 1487580006317032337                     2.38    553329724       2067216c-31b5-455d-a1cc-af0575a34ffb
2019-11-01 00:00:10 UTC view    5837166 1783999064103190764             pnb     22.22   556138645       57ed222e-a54a-4907-9944-5a875c2d7f4f
2019-11-01 00:00:11 UTC cart    5876812 1487580010100293687             jessnail        3.16    564506666       186c1951-8052-4b37-adce-dd9644b1d5f7
Time taken: 0.475 seconds, Fetched: 5 row(s)
```

h. Once the base table is created, we need to optimize the table for quick query result through partitioning and bucketing.



```
hive> set hive.exec.dynamic.partition.mode=nonstrict;
hive> set hive.exec.dynamic.partition=true;
hive> set hive.enforce.bucketing=true ;
hive> 
```

Command:

create table if not exists base_bucket (event_time string, product_id string , category_id string , category_code string ,

brand string, price float, user_id bigint , user_session string)

partitioned by (event_type string)

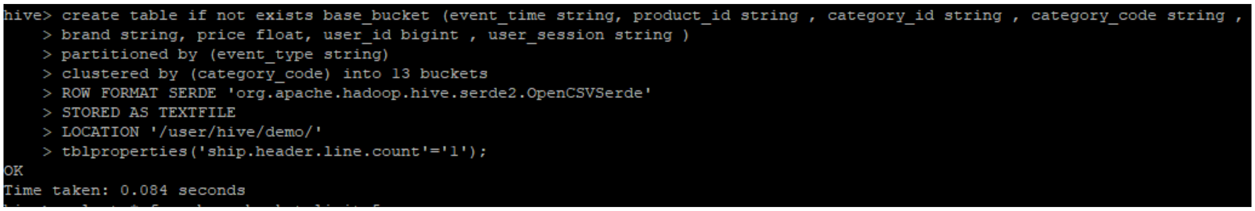
clustered by (category_code) into 13 buckets

ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'

STORED AS TEXTFILE

LOCATION '/user/hive/demo/'

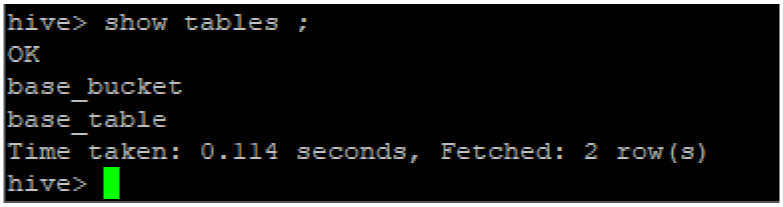
tblproperties('ship.header.line.count'='1');



```
hive> create table if not exists base_bucket (event_time string, product_id string , category_id string , category_code string ,
    > brand string, price float, user_id bigint , user_session string )
    > partitioned by (event_type string)
    > clustered by (category_code) into 13 buckets
    > ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
    > STORED AS TEXTFILE
    > LOCATION '/user/hive/demo/'
    > tblproperties('ship.header.line.count'='1');
OK
Time taken: 0.084 seconds
```

i. Once the table is created check for the created tables.

Command: show tables;



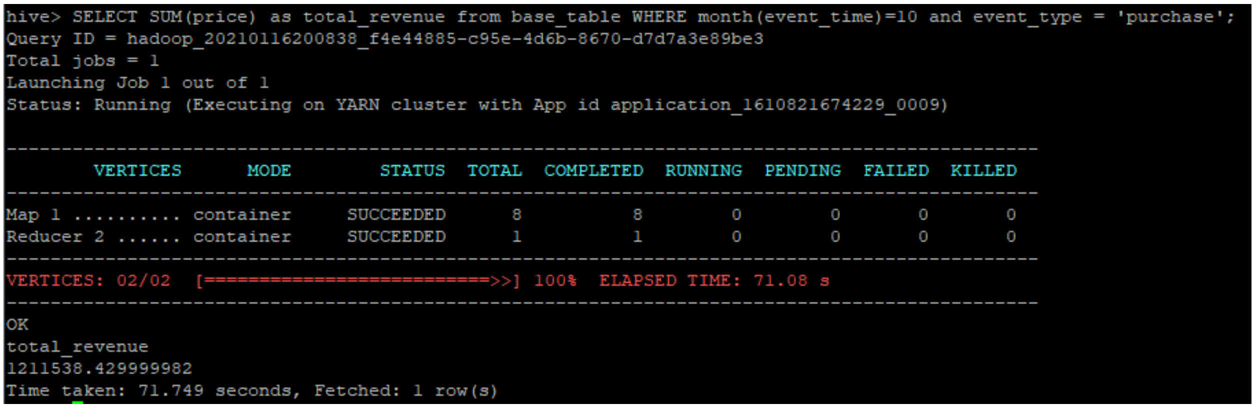
```
hive> show tables ;
OK
base_bucket
base_table
Time taken: 0.114 seconds, Fetched: 2 row(s)
hive> 
```

Step 13: Query Analysis:

1.Find the revenue generated due to purchases made in October

Code:

SELECT SUM(price) as total_revenue from base_table WHERE month(event_time)=10 and event_type = 'purchase';



```
hive> SELECT SUM(price) as total_revenue from base_table WHERE month(event_time)=10 and event_type = 'purchase';
Query ID = hadoop_20210116200838_f4e44885-c95e-4d6b-8670-d7d7a3e89be3
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1610821674229_0009)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      8          8        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 71.08 s
----------------------------------------------------------------------------------------------
OK
total_revenue
1211538.429999982
Time taken: 71.749 seconds, Fetched: 1 row(s)
```

Inference: From the query we get to know that the total revenue of the October month is found to be 1211538.429999982

==

2. Write a query to yield the total sum of the purchases per month in a single output

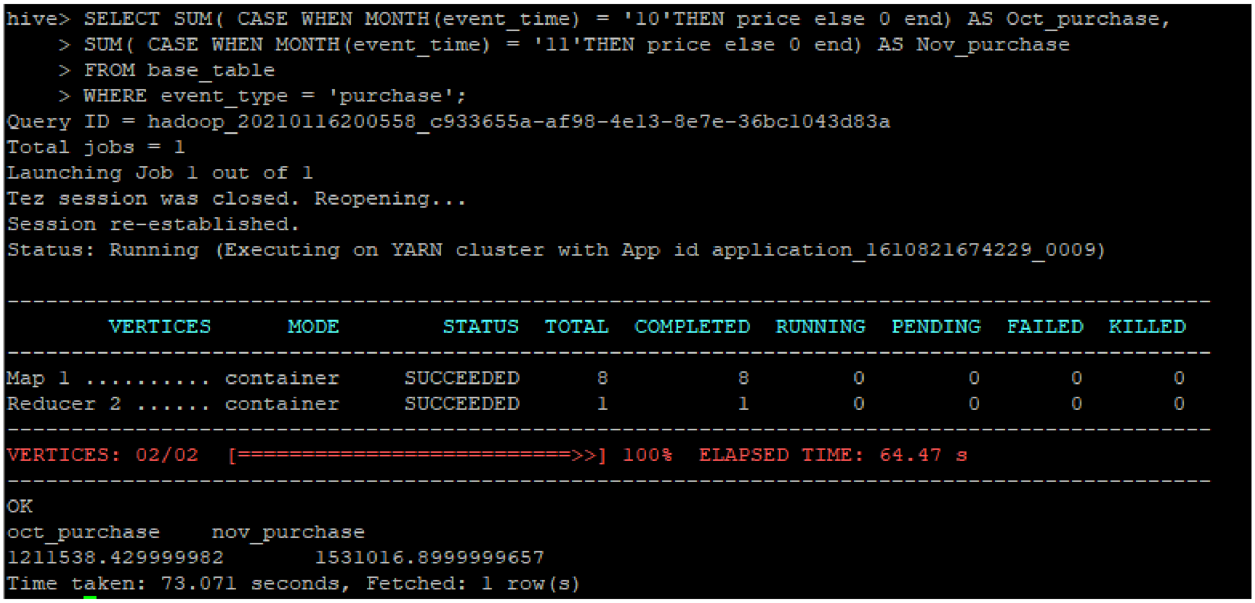
Code :

SELECT SUM(CASE WHEN MONTH(event_time) = '10'THEN price else 0 end) AS Oct_purchase,

SUM(CASE WHEN MONTH(event_time) = '11'THEN price else 0 end) AS Nov_purchase

FROM base_table

WHERE event_type = 'purchase';



```
hive> SELECT SUM( CASE WHEN MONTH(event_time) = '10'THEN price else 0 end) AS Oct_purchase,
    > SUM( CASE WHEN MONTH(event_time) = '11'THEN price else 0 end) AS Nov_purchase
    > FROM base_table
    > WHERE event_type = 'purchase';
Query ID = hadoop_20210116200558_c933655a-af98-4e13-8e7e-36bc1043d83a
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1610821674229_0009)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      8          8        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 64.47 s
----------------------------------------------------------------------------------------------
OK
oct_purchase    nov_purchase
1211538.429999982       1531016.8999999657
Time taken: 73.071 seconds, Fetched: 1 row(s)
```

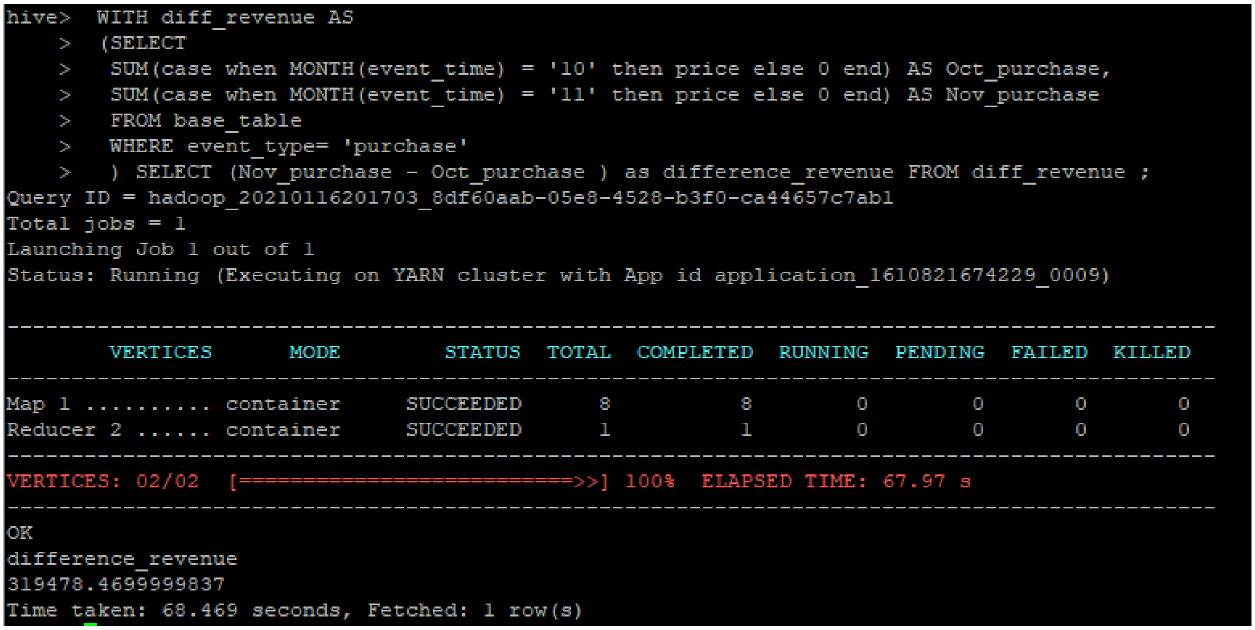
Inference: From the query we get to know the revenue generated at the end of October and November month. It is seen that the revenue of November month is higher than that of the October month.

==

3. Write a query to find the change in revenue generated due to purchases from October to November.

Code:

```
WITH diff_revenue AS
 (SELECT
  SUM(case when MONTH(event_time) = '10' then price else 0 end) AS Oct_purchase,
  SUM(case when MONTH(event_time) = '11' then price else 0 end) AS Nov_purchase
  FROM base_table
  WHERE event_type= 'purchase'
  ) SELECT (Nov_purchase - Oct_purchase) as difference_revenue FROM diff_revenue ;
```



```
hive>  WITH diff_revenue AS
    >  (SELECT
    >   SUM(case when MONTH(event_time) = '10' then price else 0 end) AS Oct_purchase,
    >   SUM(case when MONTH(event_time) = '11' then price else 0 end) AS Nov_purchase
    >   FROM base_table
    >   WHERE event_type= 'purchase'
    >   ) SELECT (Nov_purchase - Oct_purchase ) as difference_revenue FROM diff_revenue ;
Query ID = hadoop_20210116201703_8df60aab-05e8-4528-b3f0-ca44657c7ab1
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1610821674229_0009)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      8          8        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 67.97 s
----------------------------------------------------------------------------------------------
OK
difference_revenue
319478.4699999837
Time taken: 68.469 seconds, Fetched: 1 row(s)
```

Inference: From the query it is seen that the difference in revenue between October and November month is 319478.4999 which means the revenue of November month is higher than October.

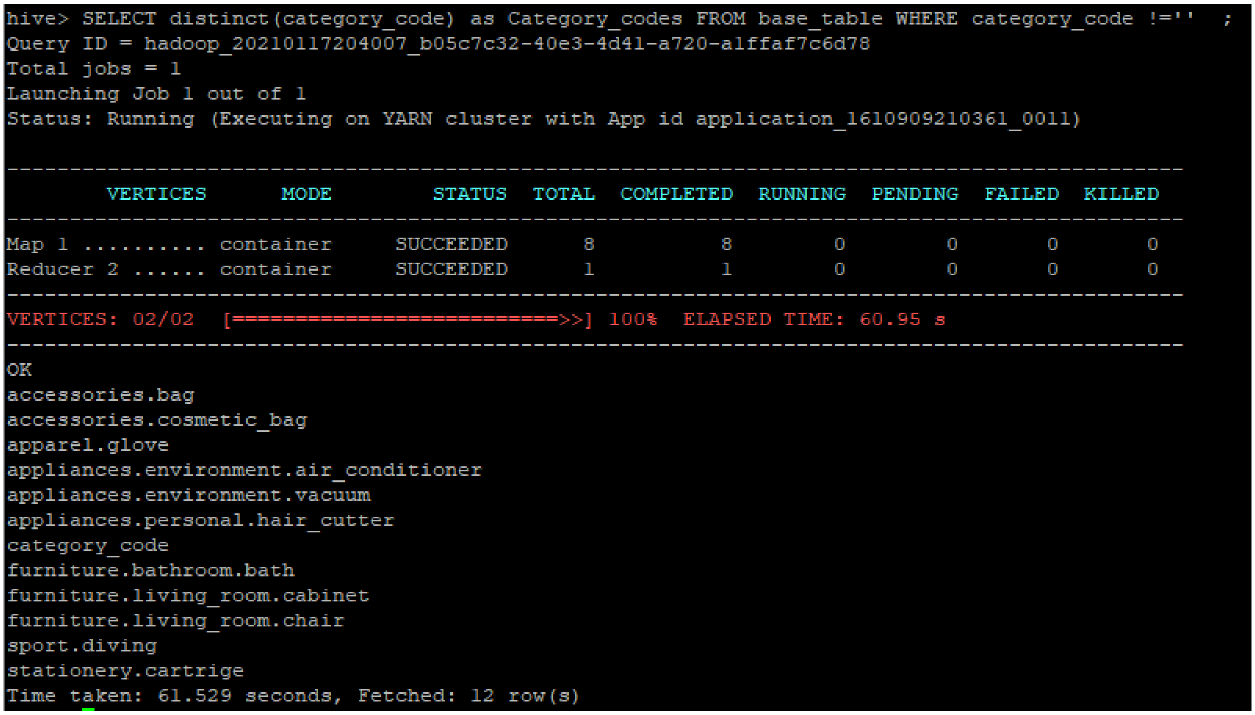
===

4. Find the distinct categories of products. Categories with null value can be ignored.

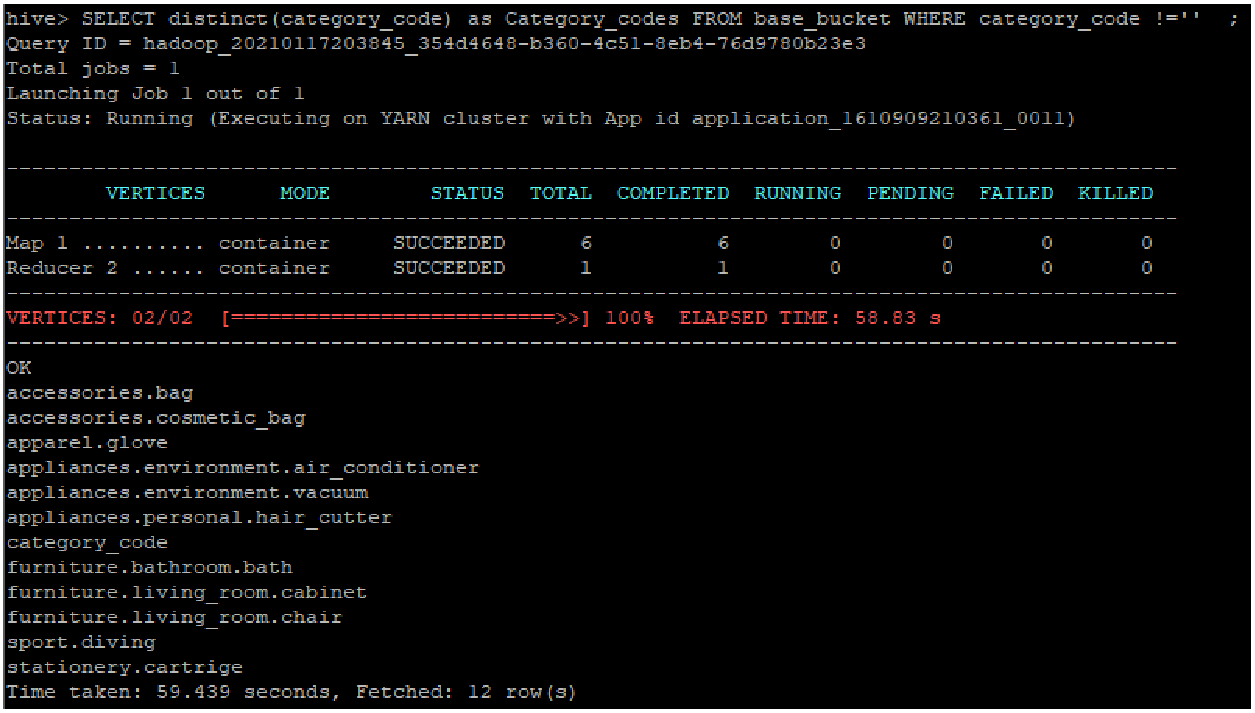
Code:

SELECT distinct(category_code) as Category_codes FROM base_table WHERE category_code !='' ;

Note: There are two screenshots of the same query from both the base table and the bucketed table. When compared the bucketed table takes less time to query the result than the base table. This is the use of partitioning and bucketing the data. **Bucketed table reduces the query time when compared to the base table.**



```
hive> SELECT distinct(category_code) as Category_codes FROM base_table WHERE category_code !=''  ;
Query ID = hadoop_20210117204007_b05c7c32-40e3-4d41-a720-a1ffaf7c6d78
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1610909210361_0011)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      8          8        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 60.95 s
----------------------------------------------------------------------------------------------
OK
accessories.bag
accessories.cosmetic_bag
apparel.glove
appliances.environment.air_conditioner
appliances.environment.vacuum
appliances.personal.hair_cutter
category_code
furniture.bathroom.bath
furniture.living_room.cabinet
furniture.living_room.chair
sport.diving
stationery.cartrige
Time taken: 61.529 seconds, Fetched: 12 row(s)
```



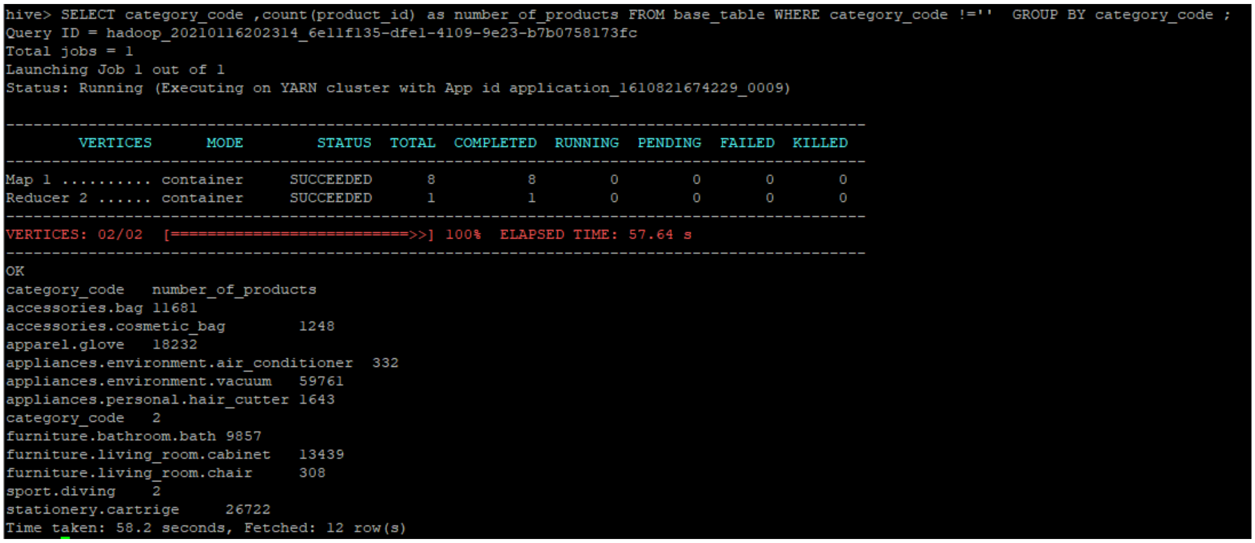
```
hive> SELECT distinct(category_code) as Category_codes FROM base_bucket WHERE category_code !=''  ;
Query ID = hadoop_20210117203845_354d4648-b360-4c51-8eb4-76d9780b23e3
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1610909210361_0011)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      6          6        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 58.83 s
----------------------------------------------------------------------------------------------
OK
accessories.bag
accessories.cosmetic_bag
apparel.glove
appliances.environment.air_conditioner
appliances.environment.vacuum
appliances.personal.hair_cutter
category_code
furniture.bathroom.bath
furniture.living_room.cabinet
furniture.living_room.chair
sport.diving
stationery.cartrige
Time taken: 59.439 seconds, Fetched: 12 row(s)
```

Inference: From the query it is seen that there are 12 different categories available in the dataset which includes different categories as accessories, apparels, appliances, furniture, sports and stationery.

5. Find the total number of products available in each category.

Code:

SELECT category_code, count(product_id) as number_of_products FROM base_table WHERE category_code !='' GROUP BY category_code ;



```
hive> SELECT category_code ,count(product_id) as number_of_products FROM base_table WHERE category_code !=''  GROUP BY category_code ;
Query ID = hadoop_20210116202314_6e11f135-dfe1-4109-9e23-b7b0758173fc
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1610821674229_0009)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      8          8        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 57.64 s
----------------------------------------------------------------------------------------------
OK
category_code   number_of_products
accessories.bag 11681
accessories.cosmetic_bag        1248
apparel.glove   18232
appliances.environment.air_conditioner  332
appliances.environment.vacuum   59761
appliances.personal.hair_cutter 1643
category_code   2
furniture.bathroom.bath 9857
furniture.living_room.cabinet   13439
furniture.living_room.chair     308
sport.diving    2
stationery.cartrige     26722
Time taken: 58.2 seconds, Fetched: 12 row(s)
```

Inference: From the query it is seen that a lot of products are available in the category environmental appliance vacuum. And the least number of products are available in the category sports diving.

==

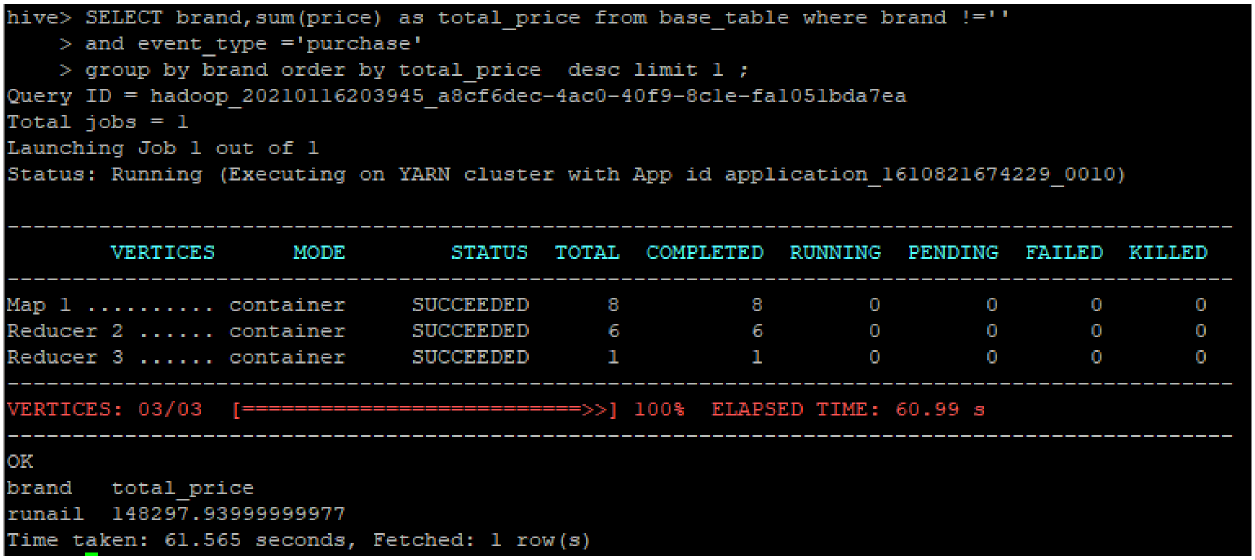
6.Which brand had the maximum sales in October and November combined?

Code:

SELECT brand,sum(price) as total_price from base_table where brand !=''

and event_type ='purchase'

group by brand order by total_price desc limit 1;



```
hive> SELECT brand,sum(price) as total_price from base_table where brand !=''
    > and event_type ='purchase'
    > group by brand order by total_price  desc limit 1 ;
Query ID = hadoop_20210116203945_a8cf6dec-4ac0-40f9-8c1e-fa1051bda7ea
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1610821674229_0010)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      8          8        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      6          6        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 60.99 s
----------------------------------------------------------------------------------------------
OK
brand   total_price
runail  148297.93999999977
Time taken: 61.565 seconds, Fetched: 1 row(s)
```

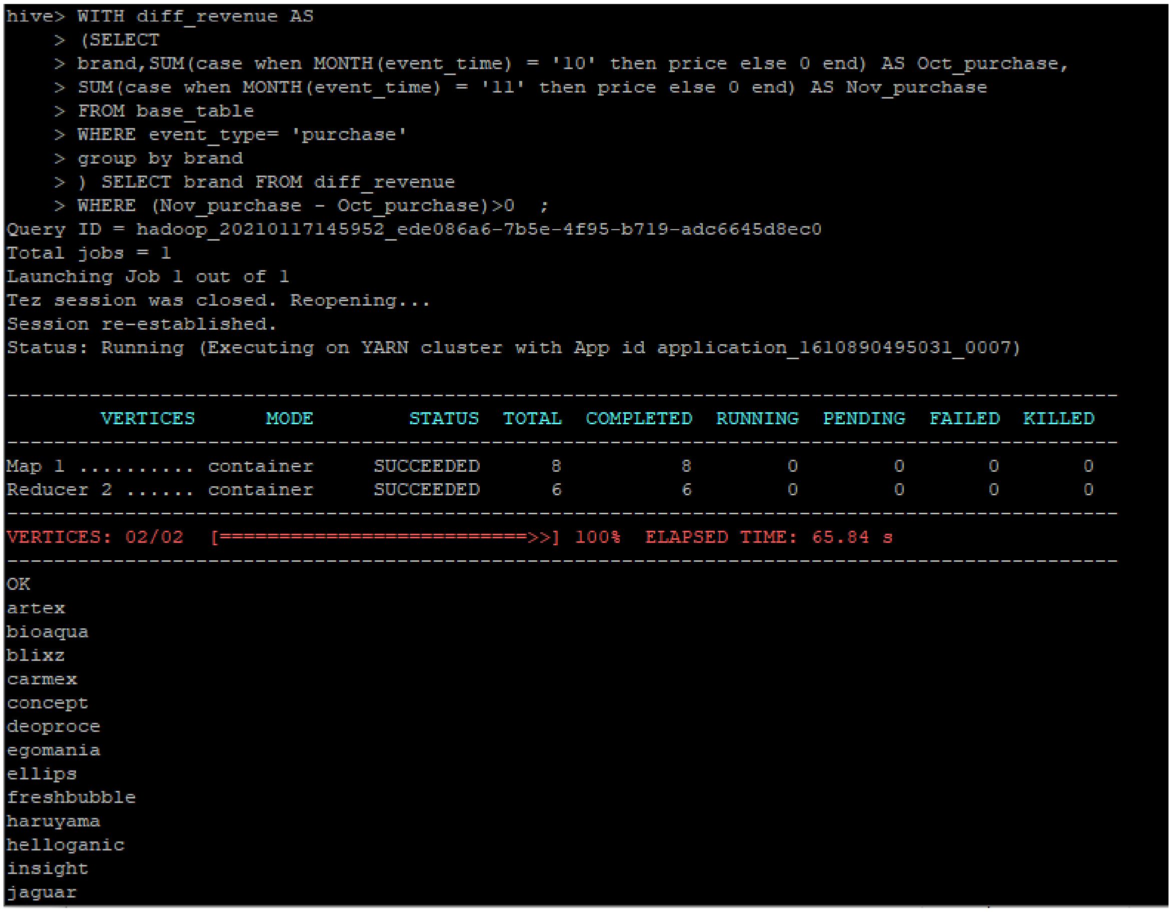
Inference: From the query it is seen that the brand having highest sales with 148297.93999 is Runail.

===

7.Which brands increased their sales from October to November?

Code:

```
WITH diff_revenue AS
(SELECT brand,SUM(case when MONTH(event_time) = '10' then price else 0 end) AS Oct_purchase,
 SUM(case when MONTH(event_time) = '11' then price else 0 end) AS Nov_purchase
 FROM base_bucket WHERE event_type= 'purchase'
  group by brand)
  SELECT brand FROM diff_revenue  WHERE (Nov_purchase - Oct_purchase) > 0  ;
```



```
hive> WITH diff_revenue AS
    > (SELECT
    > brand,SUM(case when MONTH(event_time) = '10' then price else 0 end) AS Oct_purchase,
    > SUM(case when MONTH(event_time) = '11' then price else 0 end) AS Nov_purchase
    > FROM base_table
    > WHERE event_type= 'purchase'
    > group by brand
    > ) SELECT brand FROM diff_revenue
    > WHERE (Nov_purchase - Oct_purchase)>0  ;
Query ID = hadoop_20210117145952_ede086a6-7b5e-4f95-b719-adc6645d8ec0
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1610890495031_0007)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      8          8        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      6          6        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 65.84 s
----------------------------------------------------------------------------------------------
OK
artex
bioaqua
blixz
carmex
concept
deoproce
egomania
ellips
freshbubble
haruyama
helloganic
insight
jaguar
```

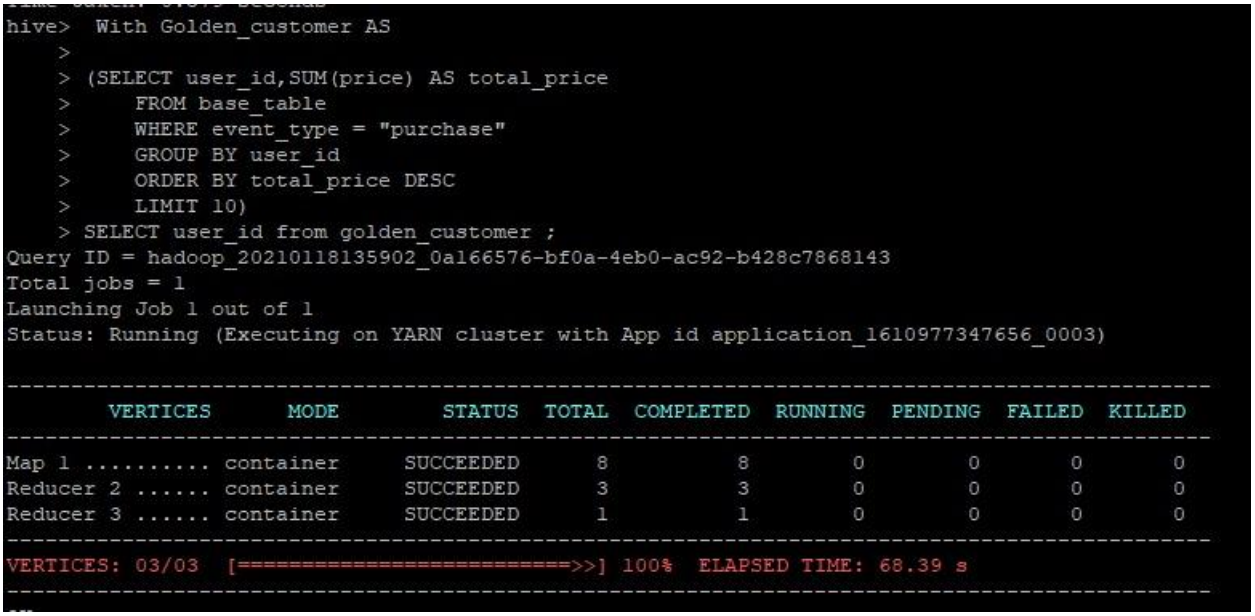
Inference: From the query we get nearly 161 out of 234 brands whose sales got increased from October to November month.

==

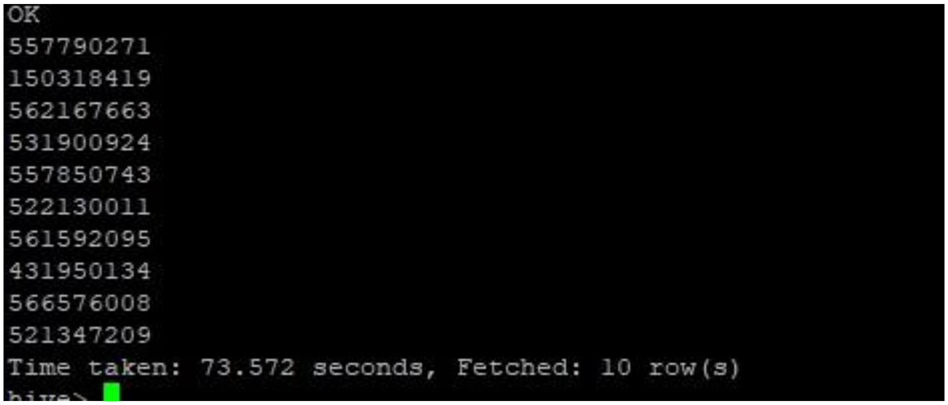
8.Your Company wants to reward the top 10 users in its website with a Golden Customer plan. Write a query to generate a list of top 10 users who spent the most.

Code:

```
With golden_customer AS
(SELECT user_id,SUM(price) AS total_price
   FROM base_table
   WHERE event_type = "purchase"
   GROUP BY user_id
   ORDER BY total_price DESC
   LIMIT 10) SELECT user_id from golden_customer ;
```



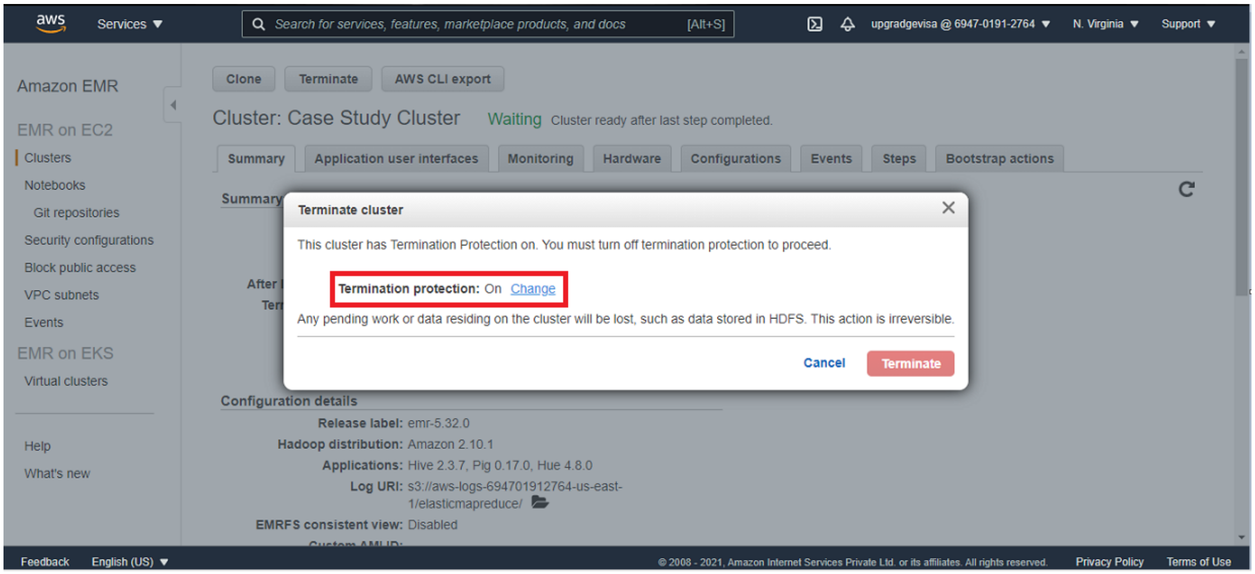
```
hive>  With Golden_customer AS
    >
    > (SELECT user_id,SUM(price) AS total_price
    >     FROM base_table
    >     WHERE event_type = "purchase"
    >     GROUP BY user_id
    >     ORDER BY total_price DESC
    >     LIMIT 10)
    > SELECT user_id from golden_customer ;
Query ID = hadoop_20210118135902_0a166576-bf0a-4eb0-ac92-b428c7868143
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1610977347656_0003)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      8          8        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      3          3        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 68.39 s
----------------------------------------------------------------------------------------------
```



```
OK
557790271
150318419
562167663
531900924
557850743
522130011
561592095
431950134
566576008
521347209
Time taken: 73.572 seconds, Fetched: 10 row(s)
hive>
```

Inference: From the query we are able to find the top 10 users who have spent the most on purchasing the goods in the e-commerce website.

Step 14: Once the analysis is done, we can terminate the cluster by changing the Termination protection from **ON** to **OFF** and then click on the terminate button.



Cluster: Case Study Cluster Waiting Cluster ready after last step completed.

Terminate cluster

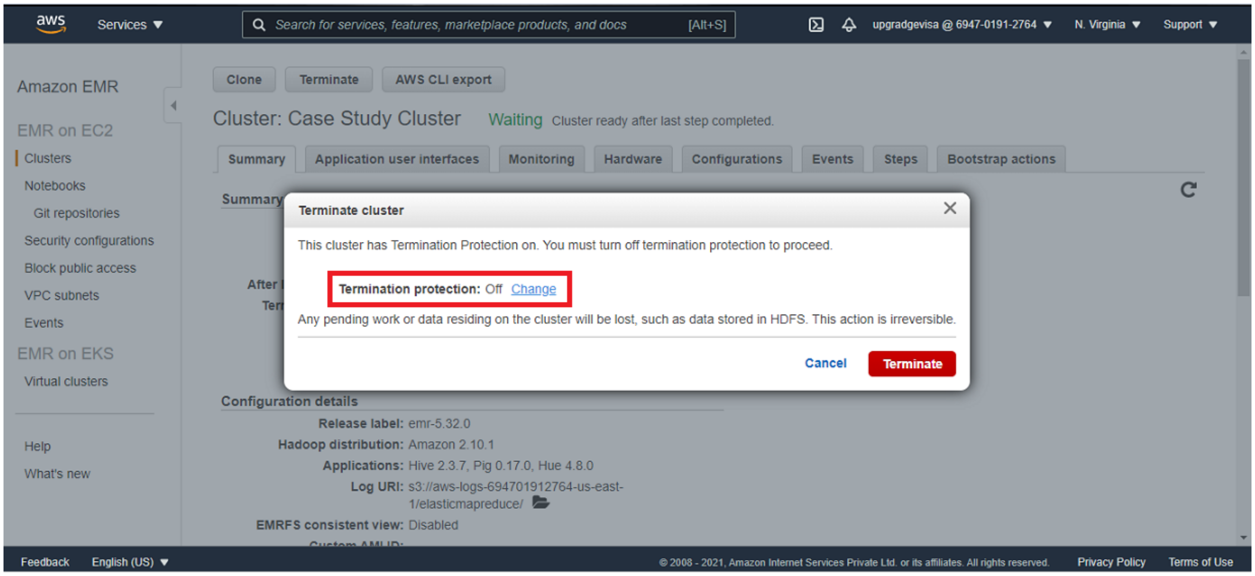
This cluster has Termination Protection on. You must turn off termination protection to proceed.

Termination protection: On Change

Any pending work or data residing on the cluster will be lost, such as data stored in HDFS. This action is irreversible.

Configuration details

Release label: emr-5.32.0
Hadoop distribution: Amazon 2.10.1
Applications: Hive 2.3.7, Pig 0.17.0, Hue 4.8.0
Log URI: s3://aws-logs-694701912764-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled



Cluster: Case Study Cluster Waiting Cluster ready after last step completed.

Terminate cluster

This cluster has Termination Protection on. You must turn off termination protection to proceed.

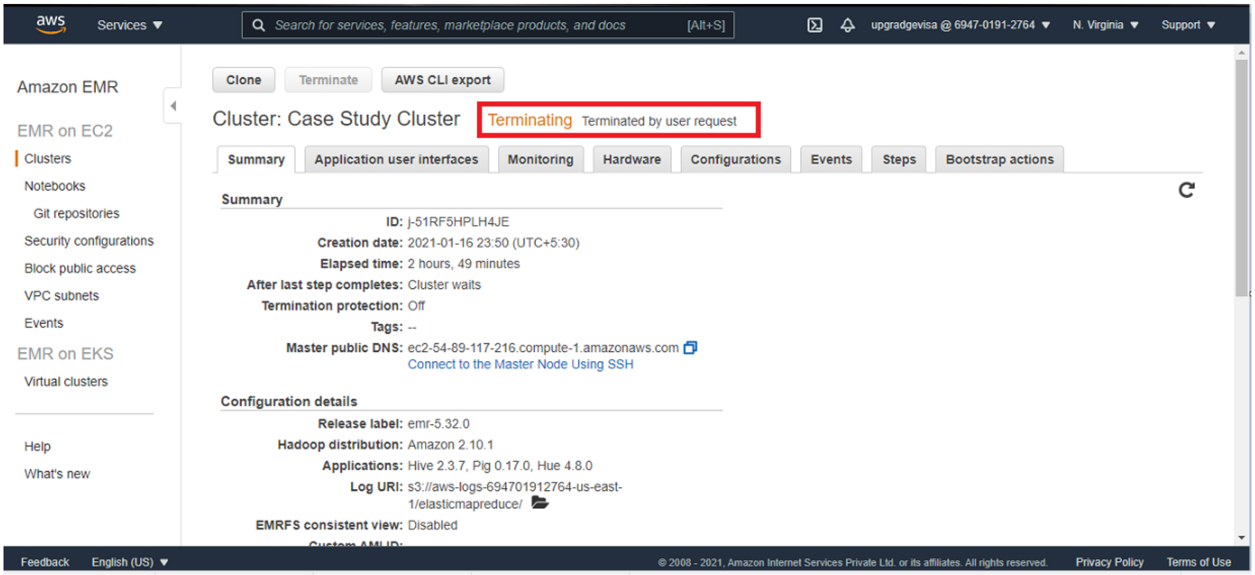
Termination protection: Off Change

Any pending work or data residing on the cluster will be lost, such as data stored in HDFS. This action is irreversible.

Configuration details

Release label: emr-5.32.0
Hadoop distribution: Amazon 2.10.1
Applications: Hive 2.3.7, Pig 0.17.0, Hue 4.8.0
Log URI: s3://aws-logs-694701912764-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled

Step 15: Once we click on the terminate button the status of the cluster changes to terminating and then the cluster is terminated.



Cluster: Case Study Cluster Terminating Terminated by user request

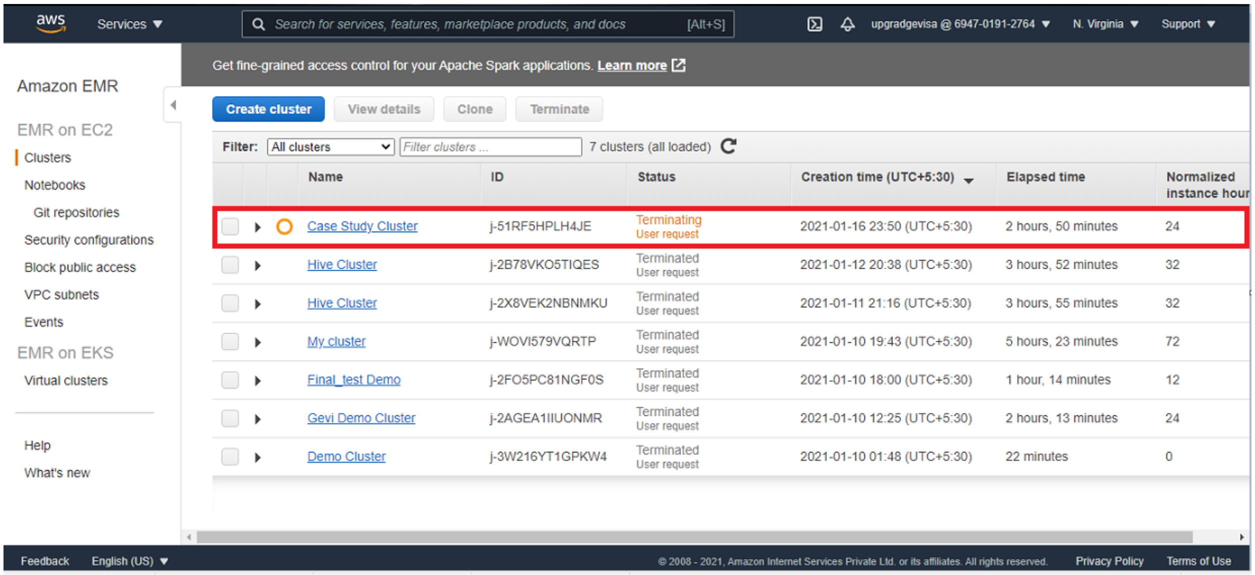
Summary

ID: j-51RF5HPLH4JE
Creation date: 2021-01-16 23:50 (UTC+5:30)
Elapsed time: 2 hours, 49 minutes
After last step completes: Cluster waits
Termination protection: Off
Tags: --
Master public DNS: ec2-54-89-117-216.compute-1.amazonaws.com
Connect to the Master Node Using SSH

Configuration details

Release label: emr-5.32.0
Hadoop distribution: Amazon 2.10.1
Applications: Hive 2.3.7, Pig 0.17.0, Hue 4.8.0
Log URI: s3://aws-logs-694701912764-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled

Step 16: Once the cluster is terminated, we can see the status of the cluster in the EMR Cluster Home.



Get fine-grained access control for your Apache Spark applications. Learn more

Filter: All clusters ▼ Filter clusters ... 7 clusters (all loaded)

Name	ID	Status	Creation time (UTC+5:30)	Elapsed time	Normalized instance hour
Case Study Cluster	j-51RF5HPLH4JE	Terminating User request	2021-01-16 23:50 (UTC+5:30)	2 hours, 50 minutes	24
Hive Cluster	j-2B78VKO5TIQES	Terminated User request	2021-01-12 20:38 (UTC+5:30)	3 hours, 52 minutes	32
Hive Cluster	j-2X8VEK2NBNMKU	Terminated User request	2021-01-11 21:16 (UTC+5:30)	3 hours, 55 minutes	32
My cluster	j-WOVI579VQRTP	Terminated User request	2021-01-10 19:43 (UTC+5:30)	5 hours, 23 minutes	72
Final_test Demo	j-2FO5PC81NGF0S	Terminated User request	2021-01-10 18:00 (UTC+5:30)	1 hour, 14 minutes	12
Gevi Demo Cluster	j-2AGEA1IIUONMR	Terminated User request	2021-01-10 12:25 (UTC+5:30)	2 hours, 13 minutes	24
Demo Cluster	j-3W216YT1GPKW4	Terminated User request	2021-01-10 01:48 (UTC+5:30)	22 minutes	0