HIVE CASE STUDY

SUBMITTED BY:

SUSHANT YADAV
NURSHAFIZAH MOHD KAMIL

PROBLEM STATEMENT:

On the clickstream cosmetic data, we are required to gain insights and analyse our customer browsing patter which will help to increase profit of the company. This can be done through analysing customer browsing pattern on which products, purchases and views when they are browsing the tracking the clicks and pattern.

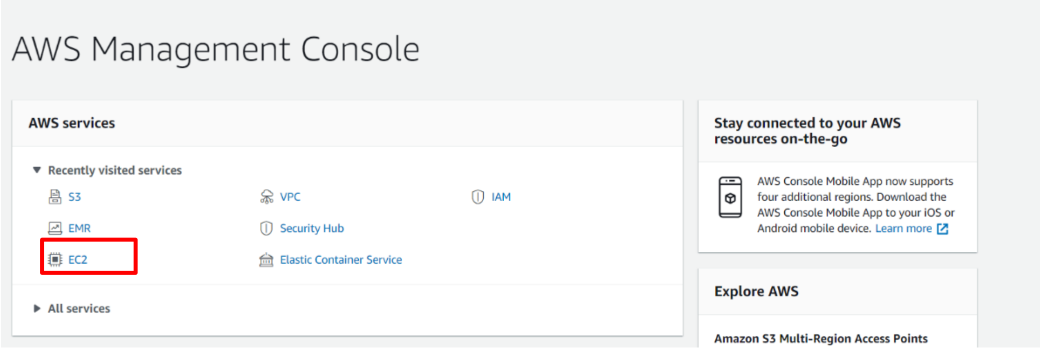
IMPLEMENTATION PHASE:

The implementation phase can be divided into the following parts:

- Copying the data set into the HDFS:
- Launch an EMR cluster that utilizes the Hive services, and
- Move the data from the S3 bucket into the HDFS
- Creating the database and launching Hive queries on your EMR cluster:
- Create the structure of your database,
- Use optimized techniques to run your queries as efficiently as possible
- Show the improvement of the performance after using optimization on any single query.
- Run Hive queries to answer the questions given below.
- Cleaning up
- Drop your database, and
- Terminate your cluster

STEPS: CREATING KEY-PAIR

On the AWS Services, click on EC2 then ‘Create key pair’.



AWS Management Console

AWS services

▼ Recently visited services

- S3
- VPC
- IAM
- EMR
- Security Hub
- EC2
- Elastic Container Service

▶ All services

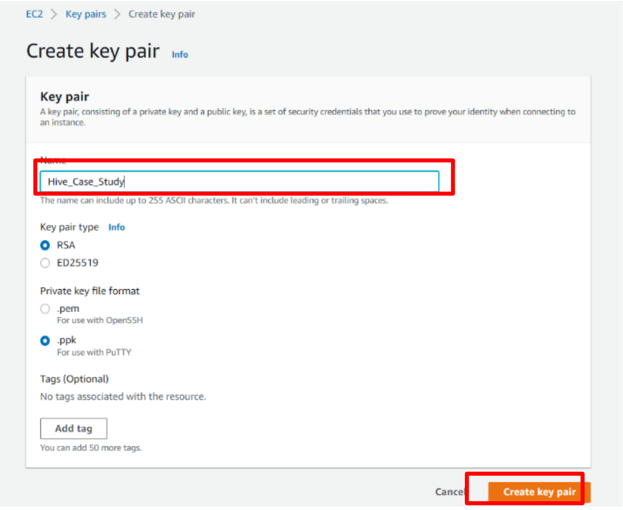
Stay connected to your AWS resources on-the-go

AWS Console Mobile App now supports four additional regions. Download the AWS Console Mobile App to your iOS or Android mobile device. Learn more

Explore AWS

Amazon S3 Multi-Region Access Points

Fill the Name section, as in our case, our key pair name is ‘Hive_Case_Study’. It is directly created as .ppk file. Next click on ‘Create key pair’.



EC2 > Key pairs > Create key pair

Create key pair Info

Key pair

A key pair, consisting of a private key and a public key, is a set of security credentials that you use to prove your identity when connecting to an instance.

Name

Hive_Case_Study

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type Info

● RSA

○ ED25519

Private key file format

○ .pem
For use with OpenSSH

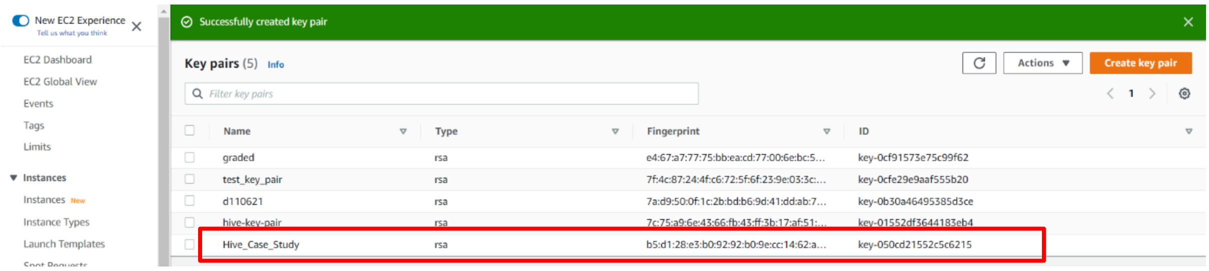
● .ppk
For use with PuTTY

Tags (Optional)

No tags associated with the resource.

Add tag

You can add 50 more tags.



Successfully created key pair

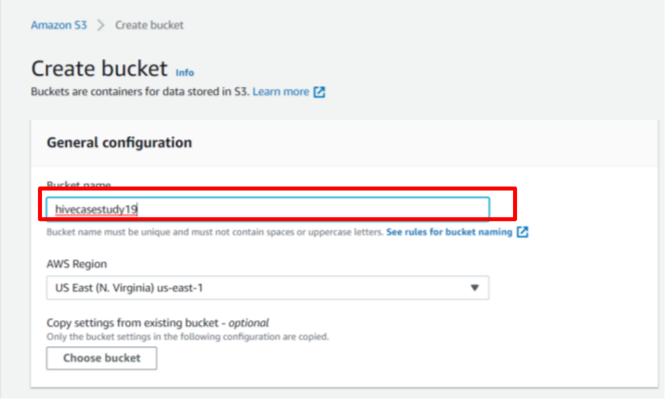
Key pairs (5) Info

□	Name	Type	Fingerprint	ID
□	graded	rsa	e4:67:a7:77:75:bb:ea:cd:77:00:6e:bc:5...	key-0cf91573e75c99f62
□	test_key_pair	rsa	7f:4c:87:24:4f:c6:72:5f:6f:23:9e:03:3c:...	key-0cfe29e9aaf555b20
□	d110621	rsa	7a:d9:50:0f:1c:2b:bd:b6:9d:41:dd:ab:7...	key-0b30a46495385d3ce
□	hive-key-pair	rsa	7c:75:a9:6e:43:66:fb:43:ff:3b:17:af:51:...	key-01552df3644183eb4
□	Hive_Case_Study	rsa	b5:d1:28:e3:b0:92:92:b0:9e:cc:14:62:a...	key-050cd21552c5c6215

Our key pair ‘Hive_Case_Study’ has been created and downloaded.

STEPS: CREATING BUCKET FOR THE CASE STUDY

We need to store our files in S3, thus the first step to serve the purpose is to ‘Create Bucket’. On ‘Amazon S3’ go to ‘Create bucket’.



Amazon S3 > Create bucket

Create bucket Info

Buckets are containers for data stored in S3. Learn more

General configuration

Bucket name

hivecasestudy19

Bucket name must be unique and must not contain spaces or uppercase letters. See rules for bucket naming

AWS Region

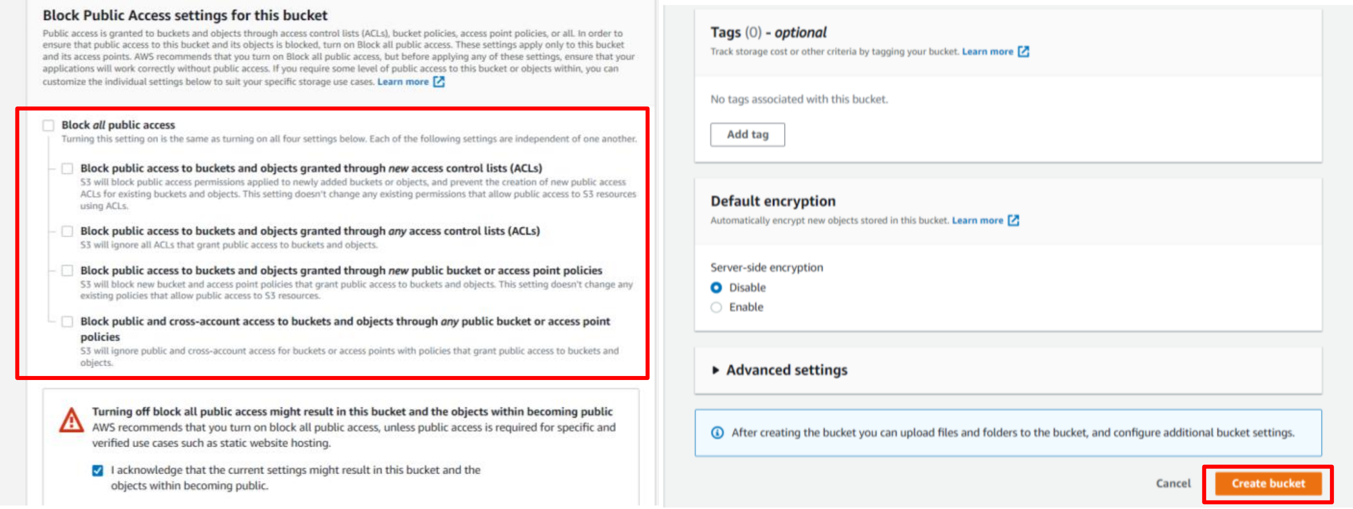
US East (N. Virginia) us-east-1

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

Field our Bucket name, do note that the bucket name should be unique.

When we are creating our Bucket, we need to enable our Public Access settings for the Bucket. Untick the following option > Create Bucket.



Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

□ Block *all* public access
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- □ Block public access to buckets and objects granted through *new* access control lists (ACLs)
  S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- □ Block public access to buckets and objects granted through *any* access control lists (ACLs)
  S3 will ignore all ACLs that grant public access to buckets and objects.
- □ Block public access to buckets and objects granted through *new* public bucket or access point policies
  S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- □ Block public and cross-account access to buckets and objects through *any* public bucket or access point policies
  S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Turning off block all public access might result in this bucket and the objects within becoming public
AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.

☑ I acknowledge that the current settings might result in this bucket and the objects within becoming public.

Tags (0) - *optional*

Track storage cost or other criteria by tagging your bucket. Learn more

No tags associated with this bucket.

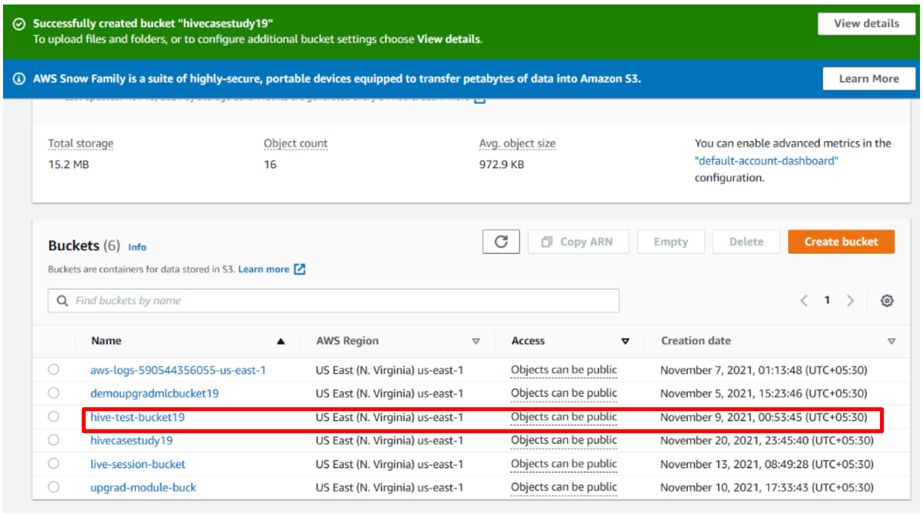
Default encryption

Automatically encrypt new objects stored in this bucket. Learn more

Server-side encryption
● Disable
○ Enable

After creating the bucket you can upload files and folders to the bucket, and configure additional bucket settings.

As we can see, our bucket ‘hivecasestudy19’ has successfully created. We should now be ready to upload our 2019-Oct.csv and 2019-Nov.csv files into the Bucket.



Successfully created bucket "hivecasestudy19"
To upload files and folders, or to configure additional bucket settings choose **View details**.

AWS Snow Family is a suite of highly-secure, portable devices equipped to transfer petabytes of data into Amazon S3.

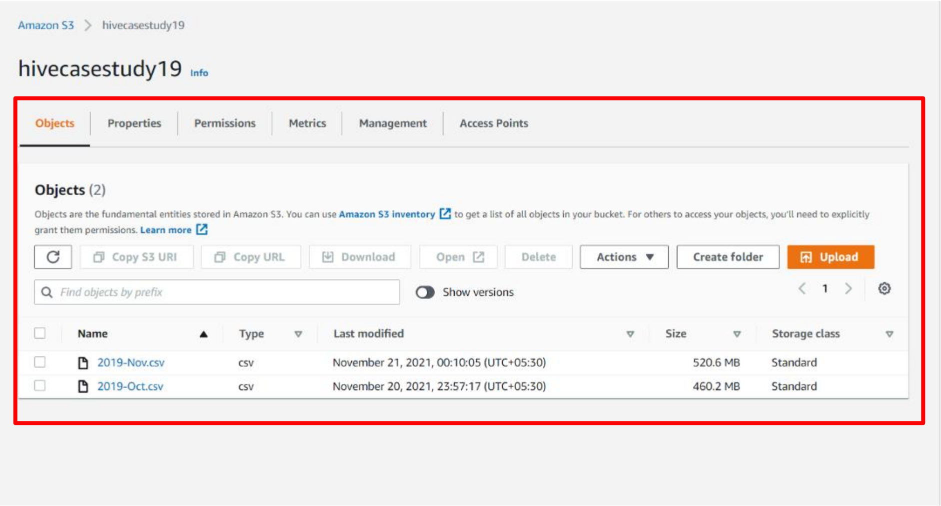
Total storage	Object count	Avg. object size	You can enable advanced metrics in the "default-account-dashboard" configuration.
15.2 MB	16	972.9 KB	

Buckets (6) Info

Buckets are containers for data stored in S3. Learn more

	Name	AWS Region	Access	Creation date
○	aws-logs-590544356055-us-east-1	US East (N. Virginia) us-east-1	Objects can be public	November 7, 2021, 01:13:48 (UTC+05:30)
○	demoupgradmlcbucket19	US East (N. Virginia) us-east-1	Objects can be public	November 5, 2021, 15:23:46 (UTC+05:30)
○	hive-test-bucket19	US East (N. Virginia) us-east-1	Objects can be public	November 9, 2021, 00:53:45 (UTC+05:30)
○	hivecasestudy19	US East (N. Virginia) us-east-1	Objects can be public	November 20, 2021, 23:45:40 (UTC+05:30)
○	live-session-bucket	US East (N. Virginia) us-east-1	Objects can be public	November 13, 2021, 08:49:28 (UTC+05:30)
○	upgrad-module-buck	US East (N. Virginia) us-east-1	Objects can be public	November 10, 2021, 17:33:43 (UTC+05:30)

In our Bucket, we uploaded both of our files successfully. We will proceed now with our cluster creation.



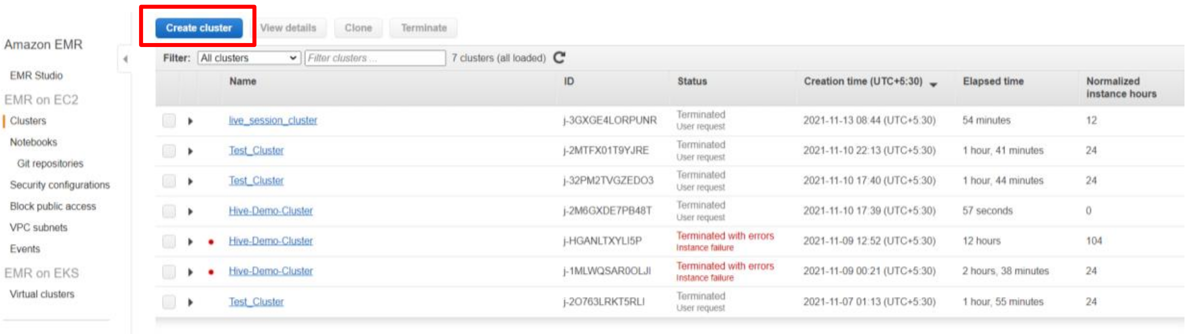
Objects (2)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

□	Name	Type	Last modified	Size	Storage class
□	2019-Nov.csv	csv	November 21, 2021, 00:10:05 (UTC+05:30)	520.6 MB	Standard
□	2019-Oct.csv	csv	November 20, 2021, 23:57:17 (UTC+05:30)	460.2 MB	Standard

STEPS: EMR (ELASTIC MAP REDUCE) – CREATING CLUSTER

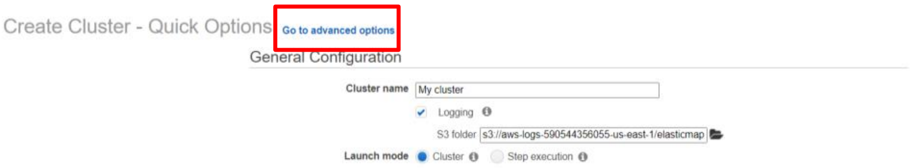
When we type EMR in the AWS Services, the below screen will pop up. As you can see, we are trying to create a cluster for this purpose of case study. Hence, below are the next steps.



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

Name	ID	Status	Creation time (UTC+5:30)	Elapsed time	Normalized instance hours
live_session_cluster	j-3GXGE4LORPUNR	Terminated User request	2021-11-13 08:44 (UTC+5:30)	54 minutes	12
Test_Cluster	j-2MTFX01T9YJRE	Terminated User request	2021-11-10 22:13 (UTC+5:30)	1 hour, 41 minutes	24
Test_Cluster	j-32PM2TVGZEDO3	Terminated User request	2021-11-10 17:40 (UTC+5:30)	1 hour, 44 minutes	24
Hive-Demo-Cluster	j-2M6GXDE7PB48T	Terminated User request	2021-11-10 17:39 (UTC+5:30)	57 seconds	0
Hive-Demo-Cluster	j-HGANLTXYLI5P	Terminated with errors Instance failure	2021-11-09 12:52 (UTC+5:30)	12 hours	104
Hive-Demo-Cluster	j-1MLWQSAR0OLJI	Terminated with errors Instance failure	2021-11-09 00:21 (UTC+5:30)	2 hours, 38 minutes	24
Test_Cluster	j-2O763LRKT5RLI	Terminated User request	2021-11-07 01:13 (UTC+5:30)	1 hour, 55 minutes	24

Click on the 'Create Cluster' then 'Go to advanced options'.



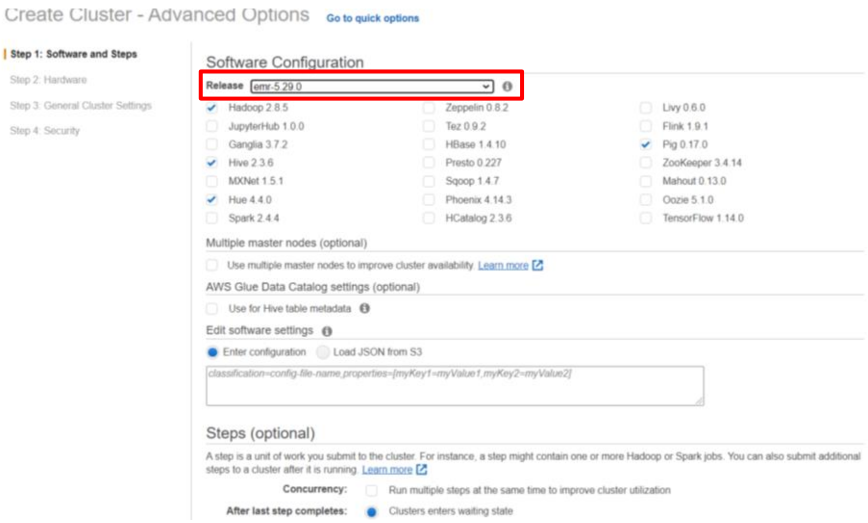
Cluster name: My cluster

☑ Logging

S3 folder: s3://aws-logs-590544356055-us-east-1/elasticmap

Launch mode: ● Cluster ○ Step execution

On the Advance Options, 'Step 1: Software and Steps' we will choose the ***EMR RELEASED 5.29.0*** for this case study. And we clicked 'Next'.


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

Steps (optional)

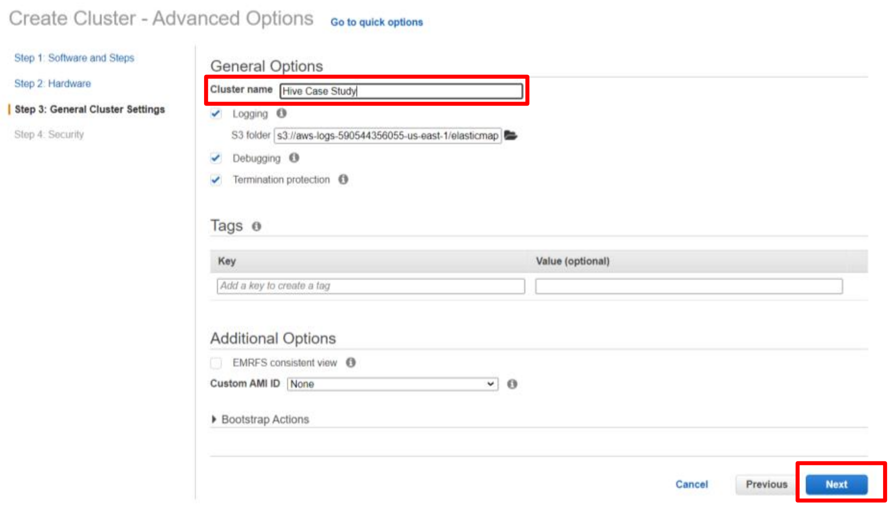
A step is a unit of work you submit to the cluster. For instance, a step might contain one or more Hadoop or Spark jobs. You can also submit additional steps to a cluster after it is running. Learn more

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

After last step completes: ● Clusters enters waiting state

Note that for the purpose of case study, we have selected EMR Released 5.29.0 as some queries might run longer in the latest released.

Our cluster name in the case study is ‘Hive Case Study’. And then, we clicked ‘Next’



Cluster name: Hive Case Study

☑ Logging

S3 folder: s3://aws-logs-590544356055-us-east-1/elasticmap

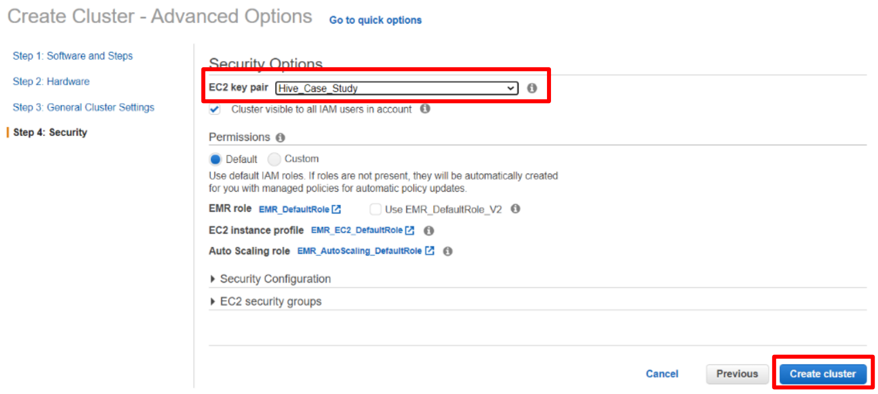
☑ Debugging

☑ Termination protection

□ EMRFS consistent view

Custom AMI ID: None

In the Security Options, we will be using our EC2 Key-Pair ‘Hive_Case_Study’ which had been downloaded earlier, and we clicked on ‘Create Cluster’.



EC2 key pair: Hive_Case_Study

☑ Cluster visible to all IAM users in account

● Default ○ Custom

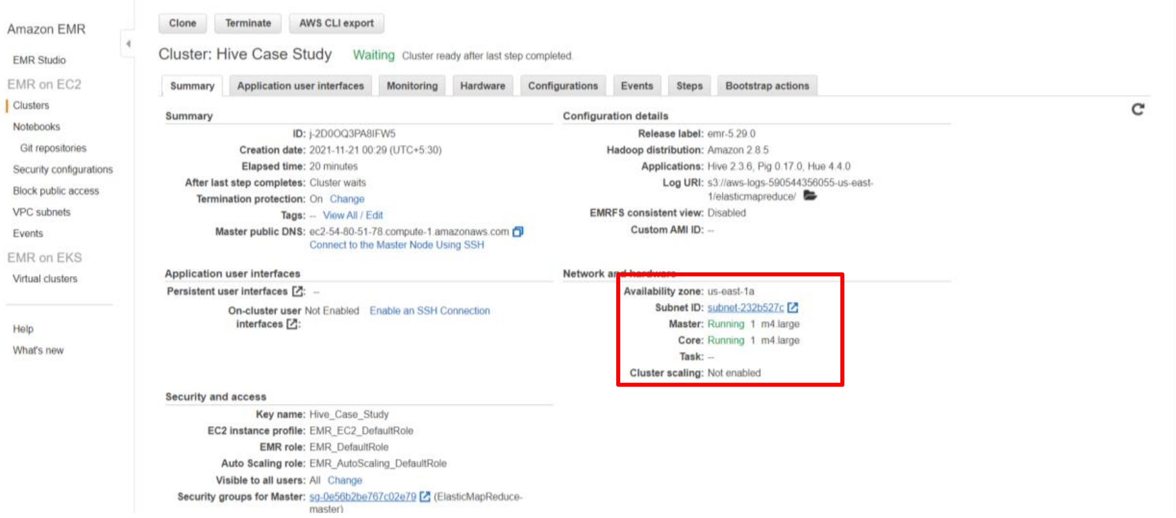
Use default IAM roles. If roles are not present, they will be automatically created for you with managed policies for automatic policy updates.

EMR role: EMR_DefaultRole □ Use EMR_DefaultRole_V2

EC2 instance profile: EMR_EC2_DefaultRole

Auto Scaling role: EMR_AutoScaling_DefaultRole

As you can see, our cluster is ready. For this case study, we will be using ***2 node cluster*** which consist of **1 Master Node (m4.large)** and **1 Core Node (m4.large)**. And both of these nodes are ready and running.



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

Summary

ID: j-2D0OQ3PA8IFW5
Creation date: 2021-11-21 00:29 (UTC+5:30)
Elapsed time: 20 minutes
After last step completes: Cluster waits
Termination protection: On Change
Tags: -- View All / Edit
Master public DNS: ec2-54-80-51-78.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, Pig 0.17.0, Hue 4.4.0
Log URI: s3://aws-logs-590544356055-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled
Custom AMI ID: --

Application user interfaces

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

Network and hardware

Availability zone: us-east-1a
Subnet ID: subnet-232b527c
Master: Running 1 m4.large
Core: Running 1 m4.large
Task: --
Cluster scaling: Not enabled

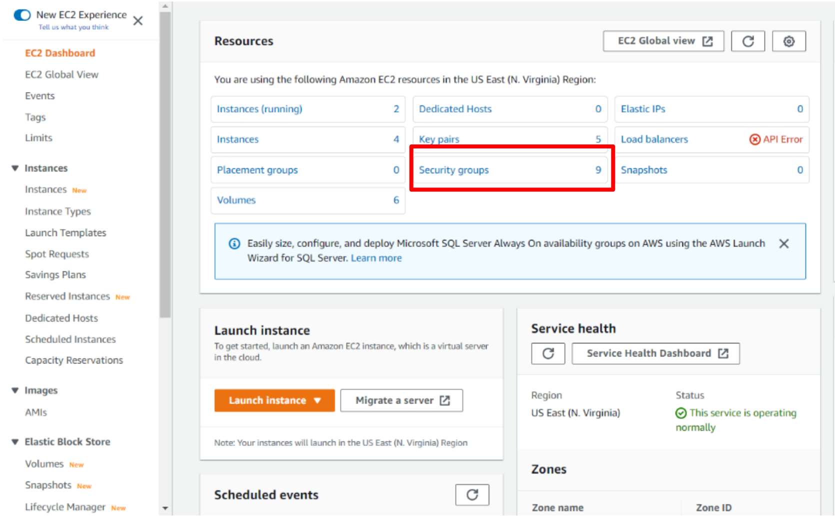
Security and access

Key name: Hive_Case_Study
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-0e56b2be767c02e79 (ElasticMapReduce-master)

STEPS: SECURITY GROUPS

Once the cluster has successfully created, we have to edit the ‘inbound rule’.

We will go to the EC2 and click on the ‘Security Groups’.



Resources

You are using the following Amazon EC2 resources in the US East (N. Virginia) Region:

Instances (running)	2	Dedicated Hosts	0	Elastic IPs	0
Instances	4	Key pairs	5	Load balancers	⊗ API Error
Placement groups	0	Security groups	9	Snapshots	0
Volumes	6				

Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. Learn more

Launch instance

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Note: Your instances will launch in the US East (N. Virginia) Region

Service health

Region: US East (N. Virginia)

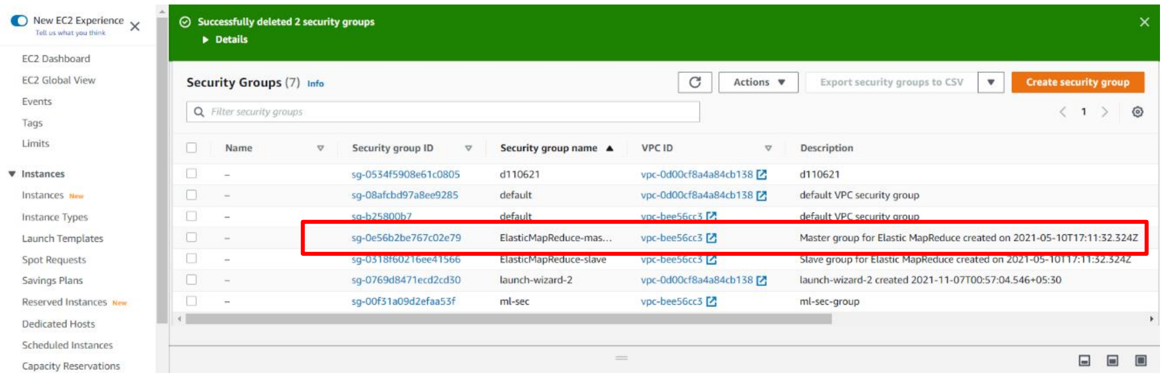
Status: This service is operating normally

Zones

Zone name | Zone ID

Scheduled events

Click on the Master Group.



Successfully deleted 2 security groups

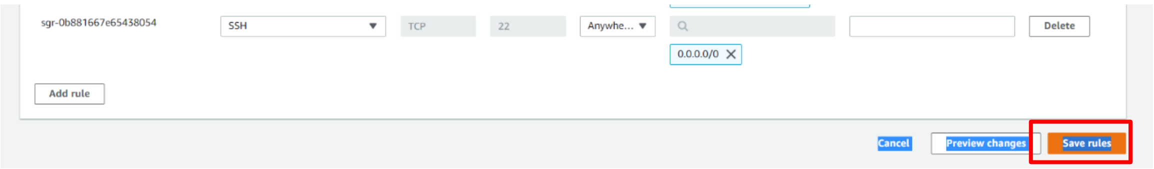
▶ Details

Security Groups (7)

Filter security groups

□	Name	Security group ID	Security group name ▲	VPC ID	Description
□	–	sg-0534f5908e61c0805	d110621	vpc-0d00cf8a4a84cb138	d110621
□	–	sg-08afcbd97a8ee9285	default	vpc-0d00cf8a4a84cb138	default VPC security group
□	–	sg-b25800b7	default	vpc-bee56cc3	default VPC security group
□	–	sg-0e56b2be767c02e79	ElasticMapReduce-mas...	vpc-bee56cc3	Master group for Elastic MapReduce created on 2021-05-10T17:11:32.324Z
□	–	sg-0318f60216ee41566	ElasticMapReduce-slave	vpc-bee56cc3	Slave group for Elastic MapReduce created on 2021-05-10T17:11:32.324Z
□	–	sg-0769d8471ecd2cd30	launch-wizard-2	vpc-0d00cf8a4a84cb138	launch-wizard-2 created 2021-11-07T00:57:04.546+05:30
□	–	sg-00f31a09d2efaa53f	ml-sec	vpc-bee56cc3	ml-sec-group

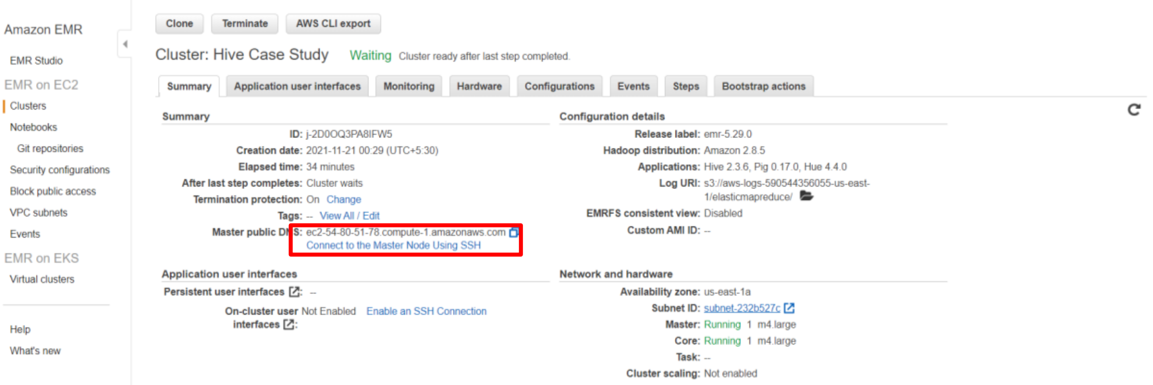
Save the ‘SSH’ inbound rules to ‘Anywhere’



sgr-0b881667e65438054 | SSH ▼ | TCP | 22 | Anywhe... ▼ | 0.0.0.0/0 ✕

STEPS: CONNECTING THE MASTER NODE WITH PUTTY

Go to our ready cluster and click on the ‘Connect to the Master Node Using SSH’, another screen will pop up:



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

Summary

ID: j-2D0OQ3PA8IFW5
Creation date: 2021-11-21 00:29 (UTC+5:30)
Elapsed time: 34 minutes
After last step completes: Cluster waits
Termination protection: On Change
Tags: -- View All / Edit
Master public DNS: ec2-54-80-51-78.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, Pig 0.17.0, Hue 4.4.0
Log URI: s3://aws-logs-590544356055-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled
Custom AMI ID: --

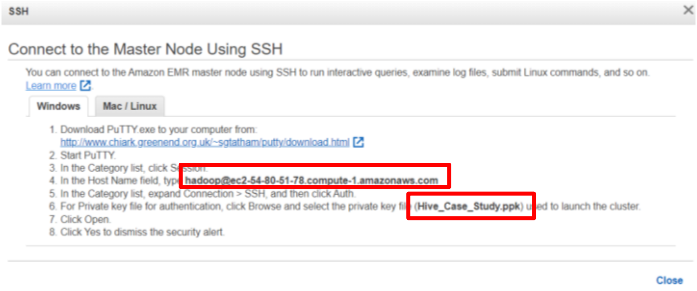
Application user interfaces

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

Network and hardware

Availability zone: us-east-1a
Subnet ID: subnet-232b527c
Master: Running 1 m4.large
Core: Running 1 m4.large
Task: --
Cluster scaling: Not enabled

Important notes to be remembered here is our ‘Host Name field’ as well as our PPK file name saved earlier. Copied the ‘Host Name field’.



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

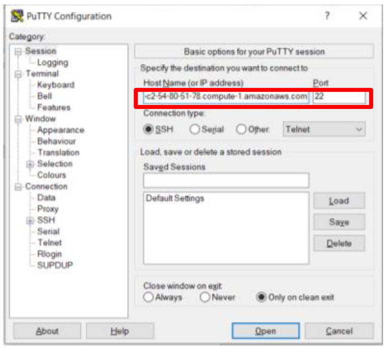
1. Download PuTTY.exe to your computer from: http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html
2. Start PuTTY.
3. In the Category list, click Session.
4. In the Host Name field, type hadoop@ec2-54-80-51-78.compute-1.amazonaws.com
5. In the Category list, expand Connection > SSH, and then click Auth.
6. For Private key file for authentication, click Browse and select the private key file (Hive_Case_Study.ppk) used to launch the cluster.
7. Click Open.
8. Click Yes to dismiss the security alert.

Host Name:

hadoop@ec2-54-80-51-78.compute-1.amazonaws.com

PPK file: Hive_Case_Study.ppk

Launch ‘Putty’, paste the ‘Host Name’ address which had been copied earlier in the ‘Host Name’ field. On the left side, click on ‘SSH’ > ‘Auth’ then browse the PPK file, in our case ‘Hive_Case_Study.ppk’, click ‘Open’ the ‘Accept’.



Basic options for your PuTTY session

Specify the destination you want to connect to

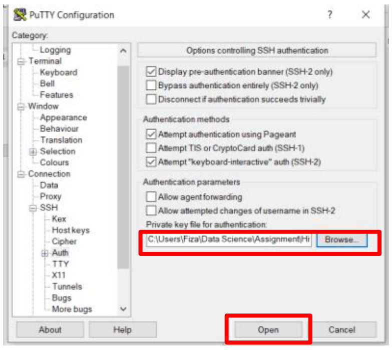
Host Name (or IP address): c2-54-80-51-78.compute-1.amazonaws.com Port: 22

Connection type: ● SSH ○ Serial ○ Other: Telnet

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)
☐ Disconnect if authentication succeeds trivially

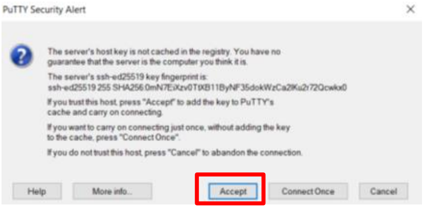
Authentication methods

☑ 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\Fiza\Data Science\Assignment\Hi



The server's host key is not cached in the registry. You have no guarantee that the server is the computer you think it is.

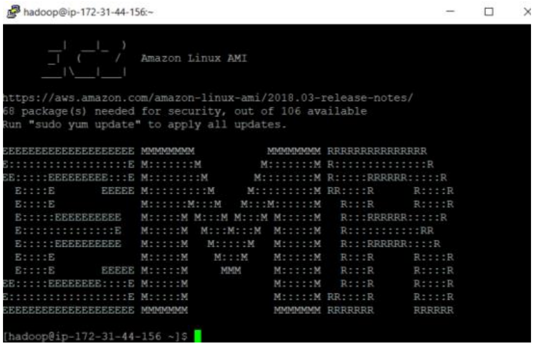
The server's ssh-ed25519 key fingerprint is:
ssh-ed25519 255 SHA256:0mN7EiXzv0TtXB11ByNF35dokWzCa2lKu2r72Qcwkx0

If you trust this host, press "Accept" to add the key to PuTTY's cache and carry on connecting.

If you want to carry on connecting just once, without adding the key to the cache, press "Connect Once".

If you do not trust this host, press "Cancel" to abandon the connection.

EMR (Elastic Map Reduce) CLI has successfully launched.

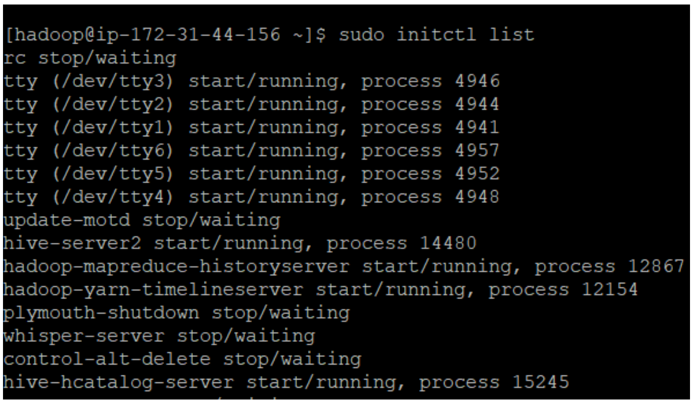


```
       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
68 package(s) needed for security, out of 106 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-172-31-44-156 ~]$
```

STEPS: QUERYING IN HADOOP AND HIVE

Check the services running in Hadoop cluster

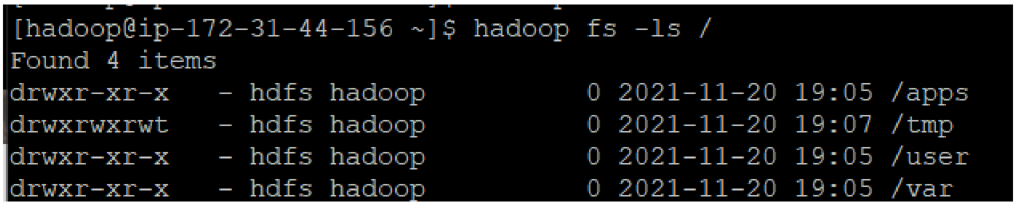
Command : ***sudo initctl list***



```
[hadoop@ip-172-31-44-156 ~]$ sudo initctl list
rc stop/waiting
tty (/dev/tty3) start/running, process 4946
tty (/dev/tty2) start/running, process 4944
tty (/dev/tty1) start/running, process 4941
tty (/dev/tty6) start/running, process 4957
tty (/dev/tty5) start/running, process 4952
tty (/dev/tty4) start/running, process 4948
update-motd stop/waiting
hive-server2 start/running, process 14480
hadoop-mapreduce-historyserver start/running, process 12867
hadoop-yarn-timelineserver start/running, process 12154
plymouth-shutdown stop/waiting
whisper-server stop/waiting
control-alt-delete stop/waiting
hive-hcatalog-server start/running, process 15245
```

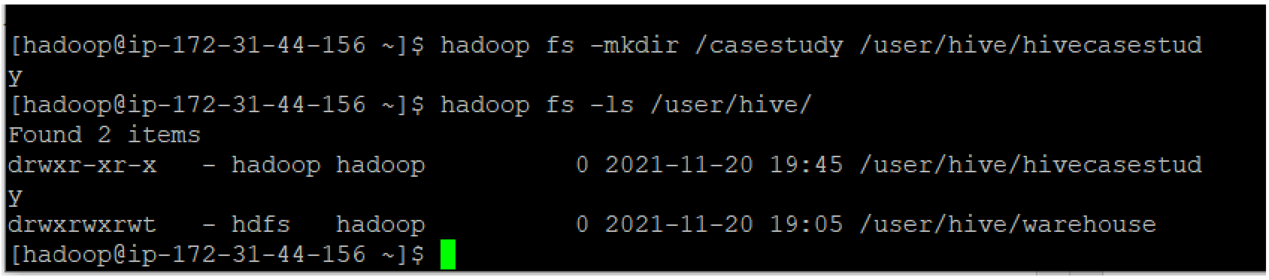
Load the data sets into HDFS from S3

1. Verifying the inbuilt file directories in HDFS.
   Command: ***hadoop fs -ls /***



```
[hadoop@ip-172-31-44-156 ~]$ hadoop fs -ls /
Found 4 items
drwxr-xr-x   - hdfs hadoop          0 2021-11-20 19:05 /apps
drwxrwxrwt   - hdfs hadoop          0 2021-11-20 19:07 /tmp
drwxr-xr-x   - hdfs hadoop          0 2021-11-20 19:05 /user
drwxr-xr-x   - hdfs hadoop          0 2021-11-20 19:05 /var
```

2. Creating a directory for our case study
   Create directory command: ***hadoop fs -mkdir /user/hive/hivecasestudy***
   Verifying the directory command: ***hadoop fs -ls /user/hive/***

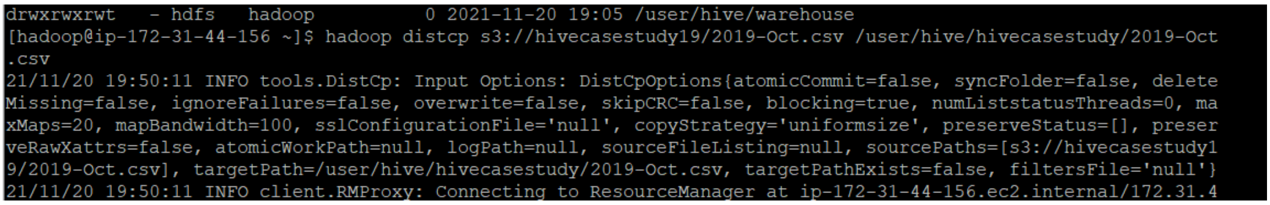


```
[hadoop@ip-172-31-44-156 ~]$ hadoop fs -mkdir /casestudy /user/hive/hivecasestud
y
[hadoop@ip-172-31-44-156 ~]$ hadoop fs -ls /user/hive/
Found 2 items
drwxr-xr-x   - hadoop hadoop          0 2021-11-20 19:45 /user/hive/hivecasestud
y
drwxrwxrwt   - hdfs   hadoop          0 2021-11-20 19:05 /user/hive/warehouse
[hadoop@ip-172-31-44-156 ~]$
```

From the above screenshot, we can see that the new directory is successfully created.

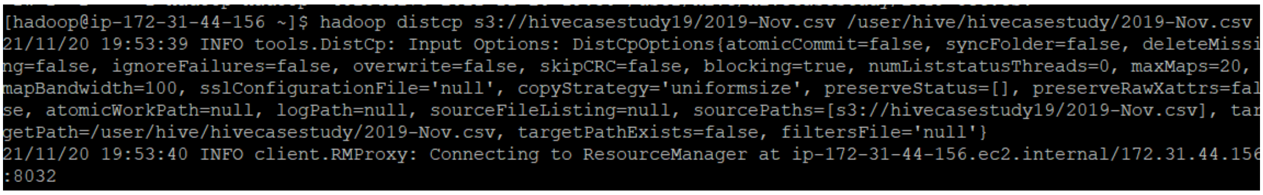
3. Load the data from S3 bucket to the HDFS

   A. For our clickstream data, Month of Oct-2019.
   Command:
   hadoop distcp s3://hivecasestudy19/2019-Oct.csv User/hive/hivecasestudy/2019-Oct.csv



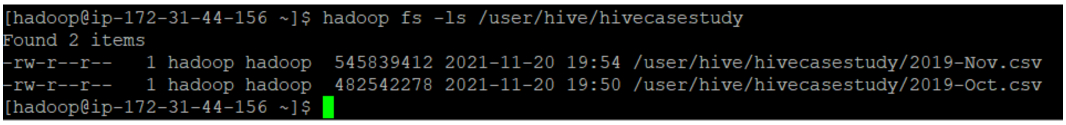
```
drwxrwxrwt   - hdfs   hadoop          0 2021-11-20 19:05 /user/hive/warehouse
[hadoop@ip-172-31-44-156 ~]$ hadoop distcp s3://hivecasestudy19/2019-Oct.csv /user/hive/hivecasestudy/2019-Oct
.csv
21/11/20 19:50:11 INFO tools.DistCp: Input Options: DistCpOptions{atomicCommit=false, syncFolder=false, delete
Missing=false, ignoreFailures=false, overwrite=false, skipCRC=false, blocking=true, numListstatusThreads=0, ma
xMaps=20, mapBandwidth=100, sslConfigurationFile='null', copyStrategy='uniformsize', preserveStatus=[], preser
veRawXattrs=false, atomicWorkPath=null, logPath=null, sourceFileListing=null, sourcePaths=[s3://hivecasestudy1
9/2019-Oct.csv], targetPath=/user/hive/hivecasestudy/2019-Oct.csv, targetPathExists=false, filtersFile='null'}
21/11/20 19:50:11 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-44-156.ec2.internal/172.31.4
```

B. Clickstream data, Month of Nov-2019.
Command:
hadoop distcp s3://hivecasestudy19/2019-Nov.csv /user/hive/hivecasestudy/2019-Nov.csv



```
[hadoop@ip-172-31-44-156 ~]$ hadoop distcp s3://hivecasestudy19/2019-Nov.csv /user/hive/hivecasestudy/2019-Nov.csv
21/11/20 19:53:39 INFO tools.DistCp: Input Options: DistCpOptions{atomicCommit=false, syncFolder=false, deleteMissi
ng=false, ignoreFailures=false, overwrite=false, skipCRC=false, blocking=true, numListstatusThreads=0, maxMaps=20,
mapBandwidth=100, sslConfigurationFile='null', copyStrategy='uniformsize', preserveStatus=[], preserveRawXattrs=fal
se, atomicWorkPath=null, logPath=null, sourceFileListing=null, sourcePaths=[s3://hivecasestudy19/2019-Nov.csv], tar
getPath=/user/hive/hivecasestudy/2019-Nov.csv, targetPathExists=false, filtersFile='null'}
21/11/20 19:53:40 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-44-156.ec2.internal/172.31.44.156
:8032
```

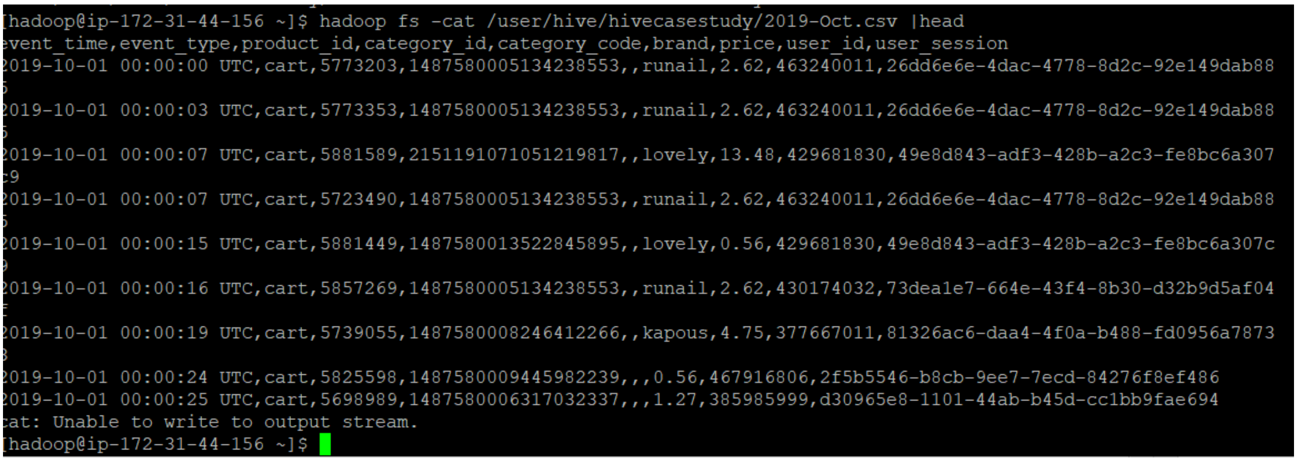
Verifying the loaded data in HDFS. Command: ***hadoop fs -ls /user/hive/hivecasestudy***



```
[hadoop@ip-172-31-44-156 ~]$ hadoop fs -ls /user/hive/hivecasestudy
Found 2 items
-rw-r--r--   1 hadoop hadoop  545839412 2021-11-20 19:54 /user/hive/hivecasestudy/2019-Nov.csv
-rw-r--r--   1 hadoop hadoop  482542278 2021-11-20 19:50 /user/hive/hivecasestudy/2019-Oct.csv
[hadoop@ip-172-31-44-156 ~]$
```

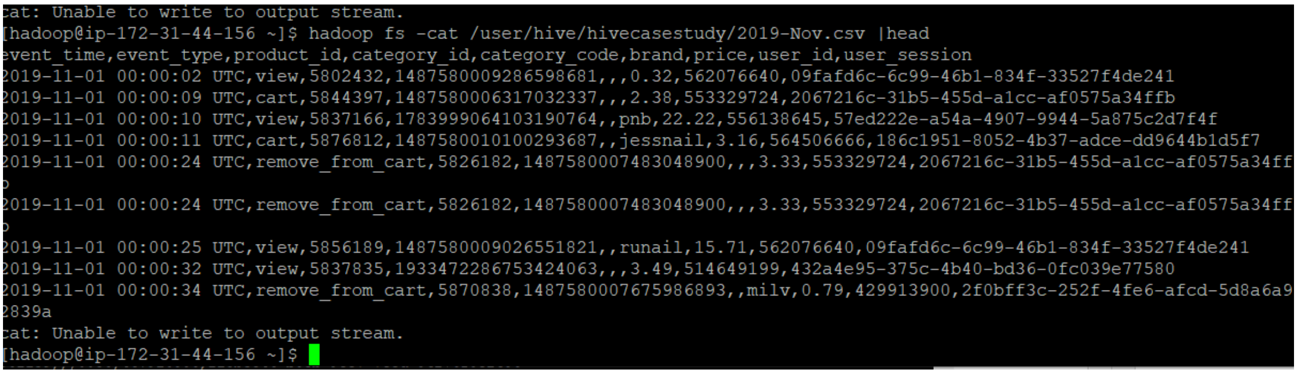
From the above screenshots, we can see that both of our files has been successfully loaded.

4. View the loaded data
   Command: ***hadoop fs -cat /user/hive/hivecasestudy/2019-Oct.csv |head***



```
[hadoop@ip-172-31-44-156 ~]$ hadoop fs -cat /user/hive/hivecasestudy/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-92e149dab88
5
2019-10-01 00:00:03 UTC,cart,5773353,1487580005134238553,,runail,2.62,463240011,26dd6e6e-4dac-4778-8d2c-92e149dab88
5
2019-10-01 00:00:07 UTC,cart,5881589,2151191071051219817,,lovely,13.48,429681830,49e8d843-adf3-428b-a2c3-fe8bc6a307
c9
2019-10-01 00:00:07 UTC,cart,5723490,1487580005134238553,,runail,2.62,463240011,26dd6e6e-4dac-4778-8d2c-92e149dab88
5
2019-10-01 00:00:15 UTC,cart,5881449,1487580013522845895,,lovely,0.56,429681830,49e8d843-adf3-428b-a2c3-fe8bc6a307c
9
2019-10-01 00:00:16 UTC,cart,5857269,1487580005134238553,,runail,2.62,430174032,73dea1e7-664e-43f4-8b30-d32b9d5af04
f
2019-10-01 00:00:19 UTC,cart,5739055,1487580008246412266,,kapous,4.75,377667011,81326ac6-daa4-4f0a-b488-fd0956a7873
8
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.
[hadoop@ip-172-31-44-156 ~]$
```

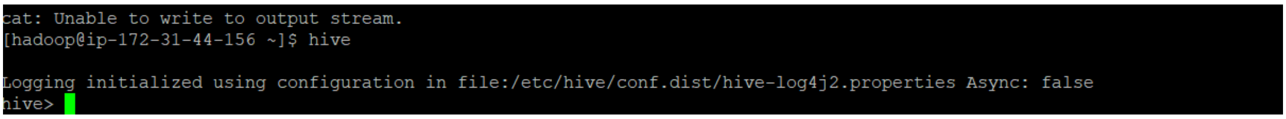
Command: ***hadoop fs -cat /user/hive/hivecasestudy/2019-Nov.csv |head***



```
cat: Unable to write to output stream.
[hadoop@ip-172-31-44-156 ~]$ hadoop fs -cat /user/hive/hivecasestudy/2019-Nov.csv |head
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
2019-11-01 00:00:24 UTC,remove_from_cart,5826182,1487580007483048900,,,3.33,553329724,2067216c-31b5-455d-a1cc-af0575a34ff
b
2019-11-01 00:00:24 UTC,remove_from_cart,5826182,1487580007483048900,,,3.33,553329724,2067216c-31b5-455d-a1cc-af0575a34ff
b
2019-11-01 00:00:25 UTC,view,5856189,1487580009026551821,,runail,15.71,562076640,09fafd6c-6c99-46b1-834f-33527f4de241
2019-11-01 00:00:32 UTC,view,5837835,1933472286753424063,,,3.49,514649199,432a4e95-375c-4b40-bd36-0fc039e77580
2019-11-01 00:00:34 UTC,remove_from_cart,5870838,1487580007675986893,,milv,0.79,429913900,2f0bff3c-252f-4fe6-afcd-5d8a6a9
2839a
cat: Unable to write to output stream.
[hadoop@ip-172-31-44-156 ~]$
```

<u>Create database</u>

Launch Hive: Hadoop > Hive.



```
cat: Unable to write to output stream.
[hadoop@ip-172-31-44-156 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive>
```

1. Creating a database

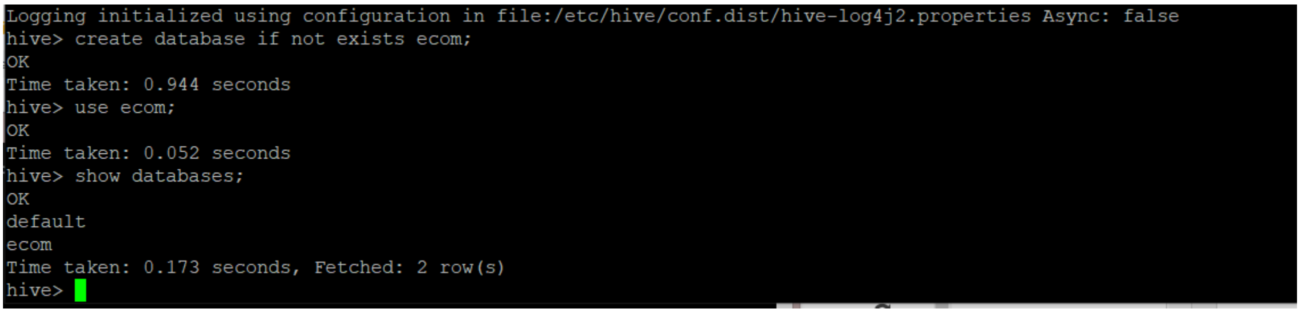
   Command: ***Create database if not exists ecom;***

2. Use database created

   Command : ***use ecom;***

3. Verifying the database created

   Command: ***show databases;***



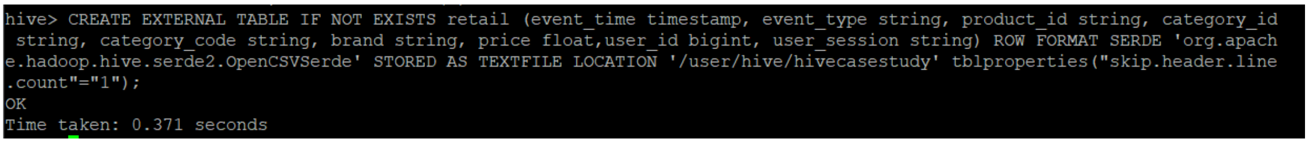
```
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive> create database if not exists ecom;
OK
Time taken: 0.944 seconds
hive> use ecom;
OK
Time taken: 0.052 seconds
hive> show databases;
OK
default
ecom
Time taken: 0.173 seconds, Fetched: 2 row(s)
hive>
```

<u>Create a table</u>

1. Create a table using CSVSerde

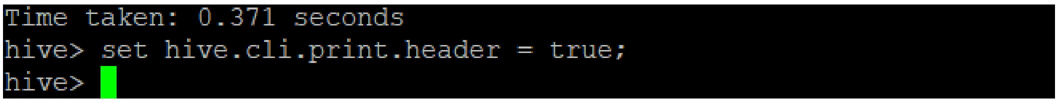
   Command:

CREATE EXTERNAL TABLE IF NOT EXISTS retail (event_time timestamp, event_type string, product_id string, category_id string, category_code string, brand string, price float,user_id bigint, user_session string) ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS TEXTFILE LOCATION '/user/hive/hivecasestudy' tblproperties("skip.header.line .count"="1");



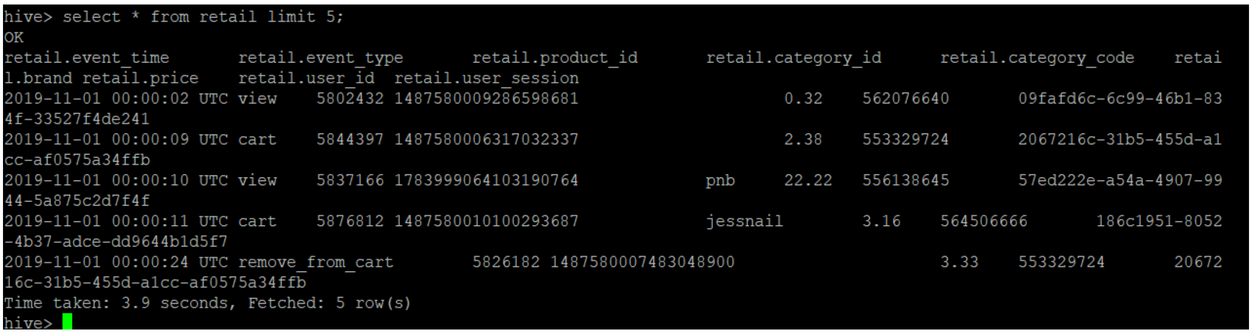
```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS retail (event_time timestamp, event_type string, product_id string, category_id
 string, category_code string, brand string, price float,user_id bigint, user_session string) ROW FORMAT SERDE 'org.apach
e.hadoop.hive.serde2.OpenCSVSerde' STORED AS TEXTFILE LOCATION '/user/hive/hivecasestudy' tblproperties("skip.header.line
.count"="1");
OK
Time taken: 0.371 seconds
```

2. Set the display for the header column
   Command: ***set hive.cli.print.header = true;***



```
Time taken: 0.371 seconds
hive> set hive.cli.print.header = true;
hive> 
```

3. Verifying the table creation by checking the top 5 rows in the table.
   Command: ***select * from retail limit 5;***



```
hive> select * from retail limit 5;
OK
retail.event_time       retail.event_type       retail.product_id       retail.category_id      retail.category_code    retai
l.brand retail.price    retail.user_id  retail.user_session
2019-11-01 00:00:02 UTC view    5802432 1487580009286598681                     0.32    562076640       09fafd6c-6c99-46b1-83
4f-33527f4de241
2019-11-01 00:00:09 UTC cart    5844397 1487580006317032337                     2.38    553329724       2067216c-31b5-455d-a1
cc-af0575a34ffb
2019-11-01 00:00:10 UTC view    5837166 1783999064103190764             pnb     22.22   556138645       57ed222e-a54a-4907-99
44-5a875c2d7f4f
2019-11-01 00:00:11 UTC cart    5876812 1487580010100293687             jessnail        3.16    564506666       186c1951-8052
-4b37-adce-dd9644b1d5f7
2019-11-01 00:00:24 UTC remove_from_cart        5826182 1487580007483048900                     3.33    553329724       20672
16c-31b5-455d-a1cc-af0575a34ffb
Time taken: 3.9 seconds, Fetched: 5 row(s)
hive> 
```

STEPS: APPLYING OPTIMIZATION TECHNIQUES (PARTITIONING & BUCKETING)

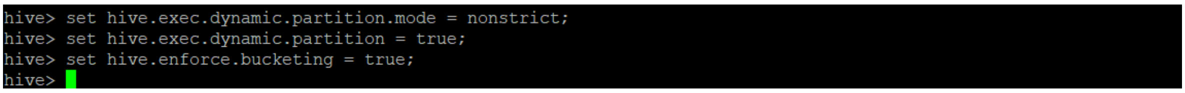
Enable Dynamic Partitioning

Command:

set hive.exec.dynamic.partition.mode = nonstrict;
set hive.exec.dynamic.partition = true;

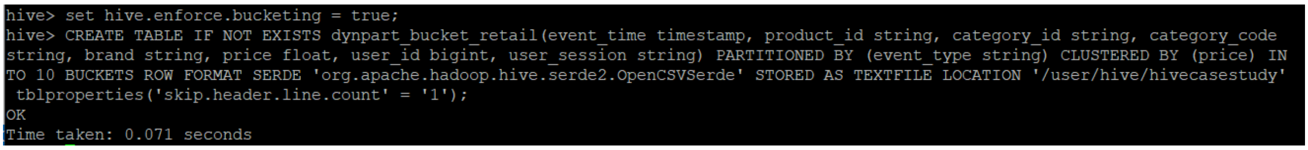
Enable Bucketing

Command: ***set hive.enforce.bucketing = true;***



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

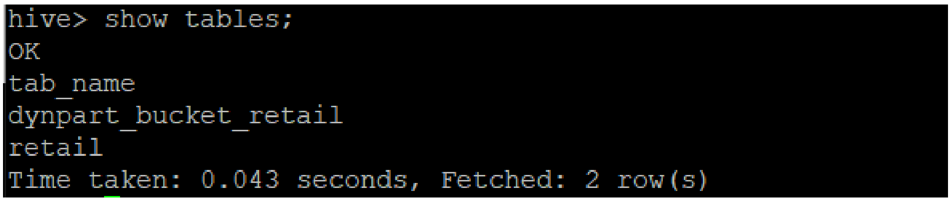
1. Create an optimized table
   Command:
   CREATE TABLE IF NOT EXISTS dynpart_bucket_retail(event_time timestamp, product_id string, category_id string, category_code string, brand string, price float, user_id bigint, user_session string) PARTITIONED BY (event_type string) CLUSTERED BY (price) INTO 10 BUCKETS ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS TEXTFILE LOCATION '/user/hive/hivecasestudy' tblproperties('skip.header.line.count' = '1');



```
hive> set hive.enforce.bucketing = true;
hive> CREATE TABLE IF NOT EXISTS dynpart_bucket_retail(event_time timestamp, 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 (price) IN
TO 10 BUCKETS ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS TEXTFILE LOCATION '/user/hive/hivecasestudy'
 tblproperties('skip.header.line.count' = '1');
OK
Time taken: 0.071 seconds
```

For this table optimization based on the partitioning and bucketing, we have decided to create a partition on ‘event_type) into 10 buckets and clustered by ‘price’.

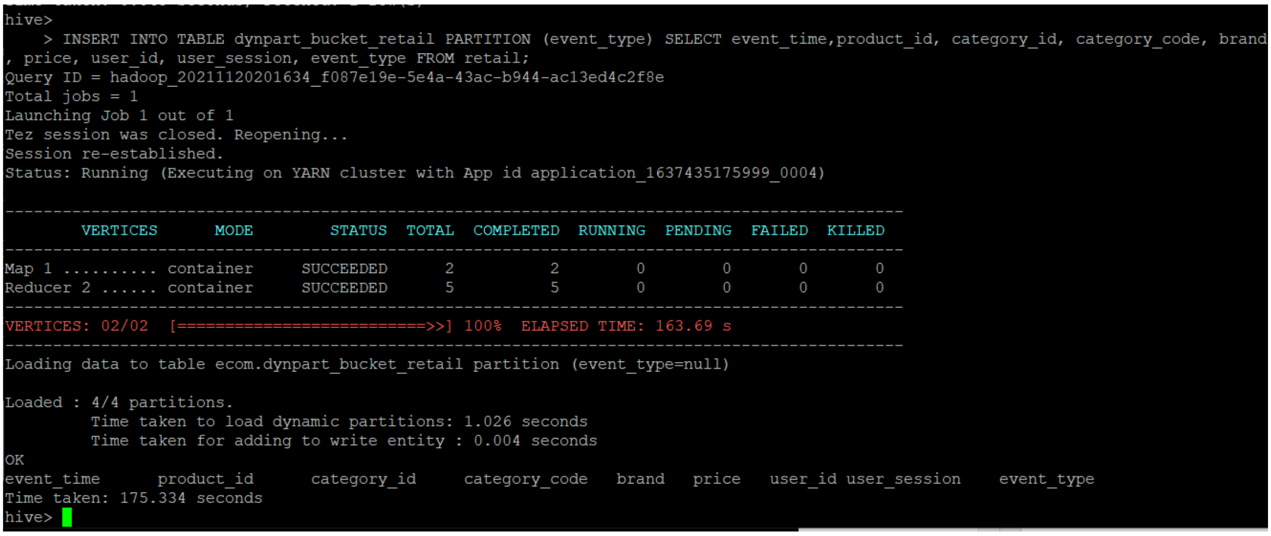
2. Verifying the table
   Command: ***show tables;***



```
hive> show tables;
OK
tab_name
dynpart_bucket_retail
retail
Time taken: 0.043 seconds, Fetched: 2 row(s)
```

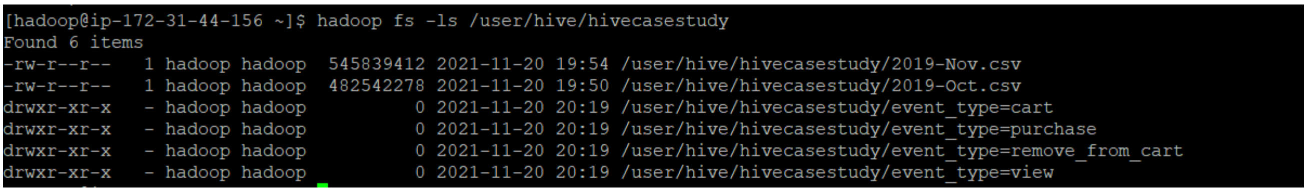
As you can see, the table with partitions and bucketing named ‘dynpart_bucket_retail’ has been successfully created.

3. Insert the data into the optimized table
   Command:
   INSERT INTO TABLE dynpart_bucket_retail PARTITION (event_type) SELECT event_time,product_id, category_id, category_code, brand , price, user_id, user_session, event_type FROM retail;



```
hive>
    > INSERT INTO TABLE dynpart_bucket_retail PARTITION (event_type) SELECT event_time,product_id, category_id, category_code, brand
, price, user_id, user_session, event_type FROM retail;
Query ID = hadoop_20211120201634_f087e19e-5e4a-43ac-b944-ac13ed4c2f8e
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_1637435175999_0004)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      2          2        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      5          5        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 163.69 s
----------------------------------------------------------------------------------------------
Loading data to table ecom.dynpart_bucket_retail partition (event_type=null)

Loaded : 4/4 partitions.
         Time taken to load dynamic partitions: 1.026 seconds
         Time taken for adding to write entity : 0.004 seconds
OK
event_time      product_id      category_id     category_code   brand   price   user_id user_session    event_type
Time taken: 175.334 seconds
hive>
```

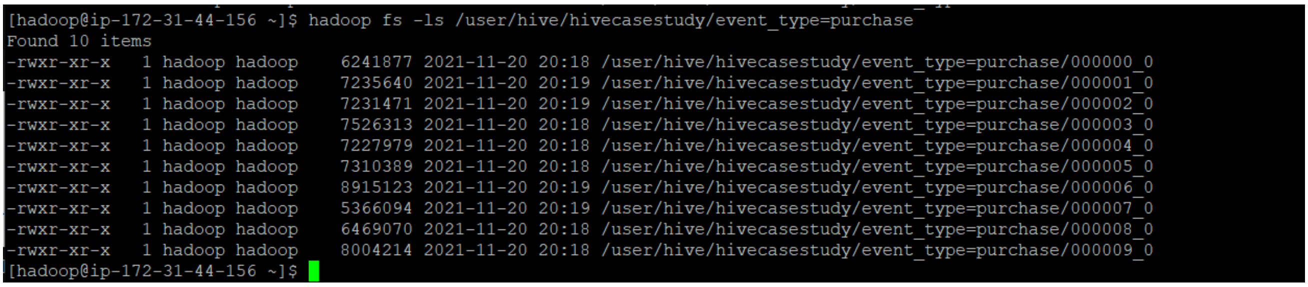
4. Verifying the table created in Hadoop
   Command: ***hadoop fs -ls /user/hive/hivecasestudy***



```
[hadoop@ip-172-31-44-156 ~]$ hadoop fs -ls /user/hive/hivecasestudy
Found 6 items
-rw-r--r--   1 hadoop hadoop  545839412 2021-11-20 19:54 /user/hive/hivecasestudy/2019-Nov.csv
-rw-r--r--   1 hadoop hadoop  482542278 2021-11-20 19:50 /user/hive/hivecasestudy/2019-Oct.csv
drwxr-xr-x   - hadoop hadoop          0 2021-11-20 20:19 /user/hive/hivecasestudy/event_type=cart
drwxr-xr-x   - hadoop hadoop          0 2021-11-20 20:19 /user/hive/hivecasestudy/event_type=purchase
drwxr-xr-x   - hadoop hadoop          0 2021-11-20 20:19 /user/hive/hivecasestudy/event_type=remove_from_cart
drwxr-xr-x   - hadoop hadoop          0 2021-11-20 20:19 /user/hive/hivecasestudy/event_type=view
```

As you can see the partitioned files has been created and partitioned by ‘event_type’ which consists of cart, purchase, remove_from_cart and view. Let’s explore further on the partitioning.

Command: ***hadoop fs -ls /user/hive/hivecasestudy/event_type=purchase***



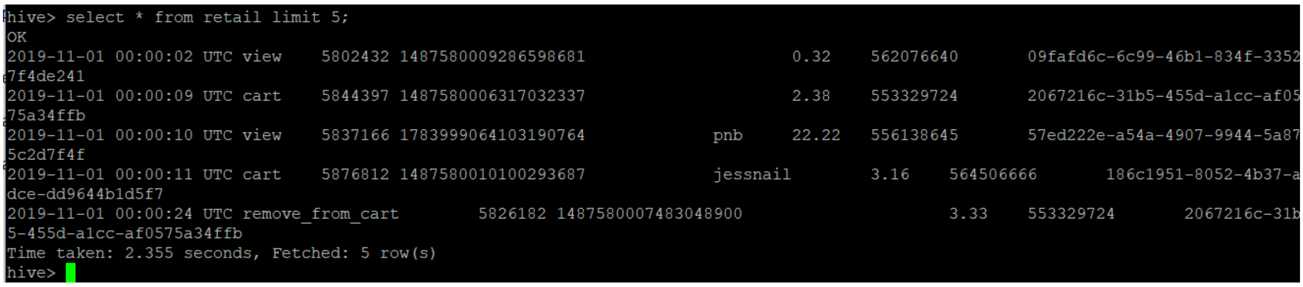
```
[hadoop@ip-172-31-44-156 ~]$ hadoop fs -ls /user/hive/hivecasestudy/event_type=purchase
Found 10 items
-rwxr-xr-x   1 hadoop hadoop    6241877 2021-11-20 20:18 /user/hive/hivecasestudy/event_type=purchase/000000_0
-rwxr-xr-x   1 hadoop hadoop    7235640 2021-11-20 20:19 /user/hive/hivecasestudy/event_type=purchase/000001_0
-rwxr-xr-x   1 hadoop hadoop    7231471 2021-11-20 20:19 /user/hive/hivecasestudy/event_type=purchase/000002_0
-rwxr-xr-x   1 hadoop hadoop    7526313 2021-11-20 20:18 /user/hive/hivecasestudy/event_type=purchase/000003_0
-rwxr-xr-x   1 hadoop hadoop    7227979 2021-11-20 20:18 /user/hive/hivecasestudy/event_type=purchase/000004_0
-rwxr-xr-x   1 hadoop hadoop    7310389 2021-11-20 20:18 /user/hive/hivecasestudy/event_type=purchase/000005_0
-rwxr-xr-x   1 hadoop hadoop    8915123 2021-11-20 20:19 /user/hive/hivecasestudy/event_type=purchase/000006_0
-rwxr-xr-x   1 hadoop hadoop    5366094 2021-11-20 20:19 /user/hive/hivecasestudy/event_type=purchase/000007_0
-rwxr-xr-x   1 hadoop hadoop    6469070 2021-11-20 20:18 /user/hive/hivecasestudy/event_type=purchase/000008_0
-rwxr-xr-x   1 hadoop hadoop    8004214 2021-11-20 20:18 /user/hive/hivecasestudy/event_type=purchase/000009_0
[hadoop@ip-172-31-44-156 ~]$
```

For the ‘event_type=purchase’, there are exactly total of 10 buckets has been created.

Verifying the performance for the tables before and after optimization.

A. Retail table without optimization

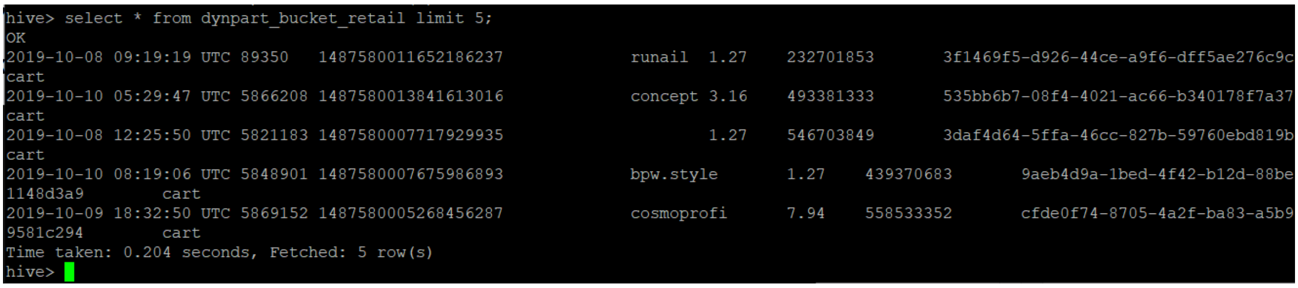
Command: ***select * from retail limit 5;***



```
hive> select * from retail limit 5;
OK
2019-11-01 00:00:02 UTC view    5802432 1487580009286598681                     0.32    562076640       09fafd6c-6c99-46b1-834f-3352
7f4de241
2019-11-01 00:00:09 UTC cart    5844397 1487580006317032337                     2.38    553329724       2067216c-31b5-455d-a1cc-af05
75a34ffb
2019-11-01 00:00:10 UTC view    5837166 1783999064103190764             pnb     22.22   556138645       57ed222e-a54a-4907-9944-5a87
5c2d7f4f
2019-11-01 00:00:11 UTC cart    5876812 1487580010100293687             jessnail        3.16    564506666       186c1951-8052-4b37-a
dce-dd9644b1d5f7
2019-11-01 00:00:24 UTC remove_from_cart        5826182 1487580007483048900                     3.33    553329724       2067216c-31b
5-455d-a1cc-af0575a34ffb
Time taken: 2.355 seconds, Fetched: 5 row(s)
hive>
```

B. Dynpart_bucket_retail table with optimization

Command: ***select * from dynpart_bucket_retail limit 5;***



```
hive> select * from dynpart_bucket_retail limit 5;
OK
2019-10-08 09:19:19 UTC 89350   1487580011652186237             runail  1.27    232701853       3f1469f5-d926-44ce-a9f6-dff5ae276c9c
cart
2019-10-10 05:29:47 UTC 5866208 1487580013841613016             concept 3.16    493381333       535bb6b7-08f4-4021-ac66-b340178f7a37
cart
2019-10-08 12:25:50 UTC 5821183 1487580007717929935                     1.27    546703849       3daf4d64-5ffa-46cc-827b-59760ebd819b
cart
2019-10-10 08:19:06 UTC 5848901 1487580007675986893             bpw.style       1.27    439370683       9aeb4d9a-1bed-4f42-b12d-88be
1148d3a9        cart
2019-10-09 18:32:50 UTC 5869152 1487580005268456287             cosmoprofi      7.94    558533352       cfde0f74-8705-4a2f-ba83-a5b9
9581c294        cart
Time taken: 0.204 seconds, Fetched: 5 row(s)
hive>
```

Insights:

1. Before optimization for the ‘retail table’ it tooks 2.355 seconds to retrieve the data from the query, however with the dynpart_bucket_table (after optimization), it tooks only 0.204 seconds to read the data. Difference of 2.151 seconds, it’s too early to make any assumptions, lets look into further queries.

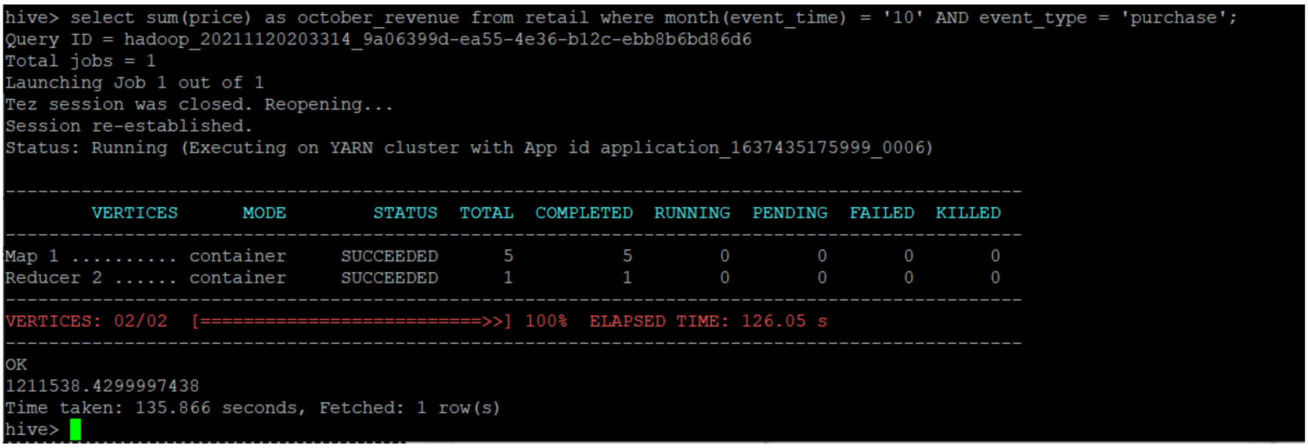
QUESTIONS

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

Without Optimization (Retail Table)

Command:

select sum(price) as october_revenue from retail where month(event_time) = '10' AND event_type = 'purchase';

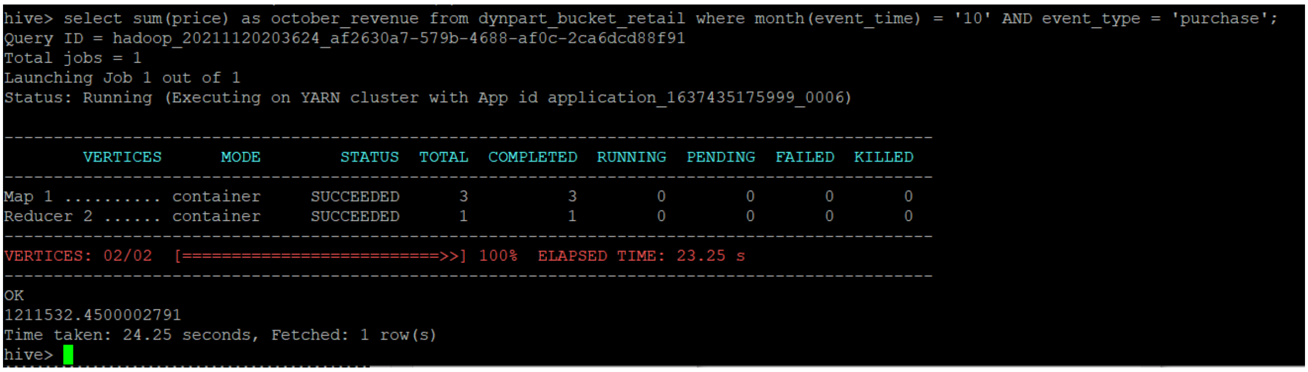


```
hive> select sum(price) as october_revenue from retail where month(event_time) = '10' AND event_type = 'purchase';
Query ID = hadoop_20211120203314_9a06399d-ea55-4e36-b12c-ebb8b6bd86d6
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1637435175999_0006)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      5          5        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 126.05 s
----------------------------------------------------------------------------------------------
OK
1211538.4299997438
Time taken: 135.866 seconds, Fetched: 1 row(s)
hive>
```

With Optimization (Dynpart_bucket_retail Table)

Command:

select sum(price) as october_revenue from dynpart_bucket_retail where month(event_time) = '10' AND event_type = 'purchase';



```
hive> select sum(price) as october_revenue from dynpart_bucket_retail where month(event_time) = '10' AND event_type = 'purchase';
Query ID = hadoop_20211120203624_af2630a7-579b-4688-af0c-2ca6dcd88f91
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1637435175999_0006)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 23.25 s
----------------------------------------------------------------------------------------------
OK
1211532.4500002791
Time taken: 24.25 seconds, Fetched: 1 row(s)
hive>
```

Insights:

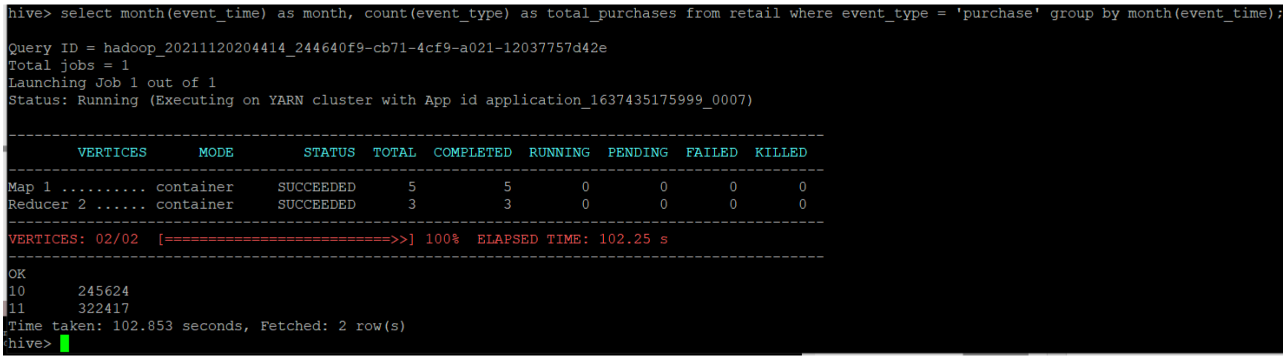
1. In this query, we are required to find the total revenue generated on October 2019 based on the Purchases.
2. The time taken to read the data for the ‘retail table (without optimization)’ is 135.866 seconds, while for the ‘dynpart_bucket_retail table (with optimization)’ is 24.25 seconds, which is 5x lesser than the retail table.
3. To answer the above questions, there are total of 1211538.43 +/- revenue generate on October on Purchases.
4. There are huge differences in the time taken to retrieve the data from both tables, in this query, the optimization table performed faster than ‘retail table’ which is without optimization.

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

Without Optimization (Retail Table)

Command:

select month(event_time) as month, count(event_type) as total_purchases from retail where event_type = 'purchase' group by month(event_time);

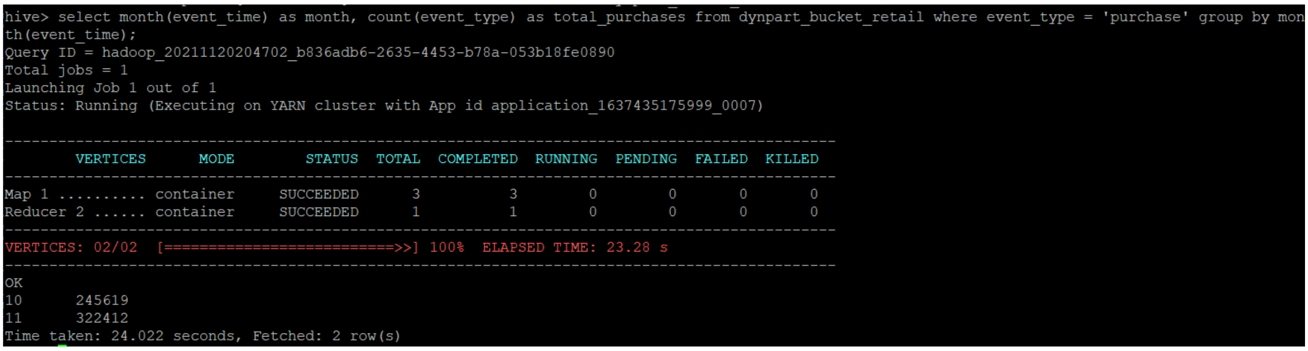


```
hive> select month(event_time) as month, count(event_type) as total_purchases from retail where event_type = 'purchase' group by month(event_time);

Query ID = hadoop_20211120204414_244640f9-cb71-4cf9-a021-12037757d42e
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1637435175999_0007)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      5          5        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      3          3        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 102.25 s
----------------------------------------------------------------------------------------------
OK
10      245624
11      322417
Time taken: 102.853 seconds, Fetched: 2 row(s)
hive>
```

With Optimization (Dynpart_bucket_retail Table)

Command:

select month(event_time) as month, count(event_type) as total_purchases from dynpart_bucket_retail where event_type = 'purchase' group by month(event_time);



```
hive> select month(event_time) as month, count(event_type) as total_purchases from dynpart_bucket_retail where event_type = 'purchase' group by mon
th(event_time);
Query ID = hadoop_20211120204702_b836adb6-2635-4453-b78a-053b18fe0890
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1637435175999_0007)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 23.28 s
----------------------------------------------------------------------------------------------
OK
10      245619
11      322412
Time taken: 24.022 seconds, Fetched: 2 row(s)
```

Insights:

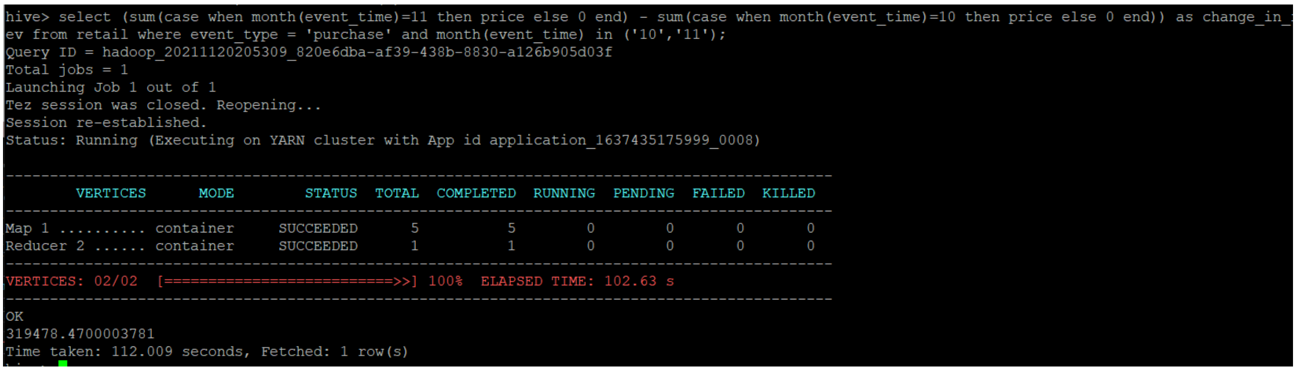
1. In this next question, we are required to find the total sum for the purchases for the month of October and November.
2. The Optimized table again prove the faster query with only 24.02 seconds for the time taken and the ratil table took 102.85 seconds to retrieve the data.
3. For the month of October, the total purchases is 245624 and 322417 for the month of November. There are a significant improvement on the purchase value which is around 30%.

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

<u>Without Optimization (Retail Table)</u>

Command:

select (sum(case when month(event_time)=11 then price else 0 end) - sum(case when month(event_time)=10 then price else 0 end)) as change_in_rev from retail where event_type = 'purchase' and month(event_time) in ('10','11');

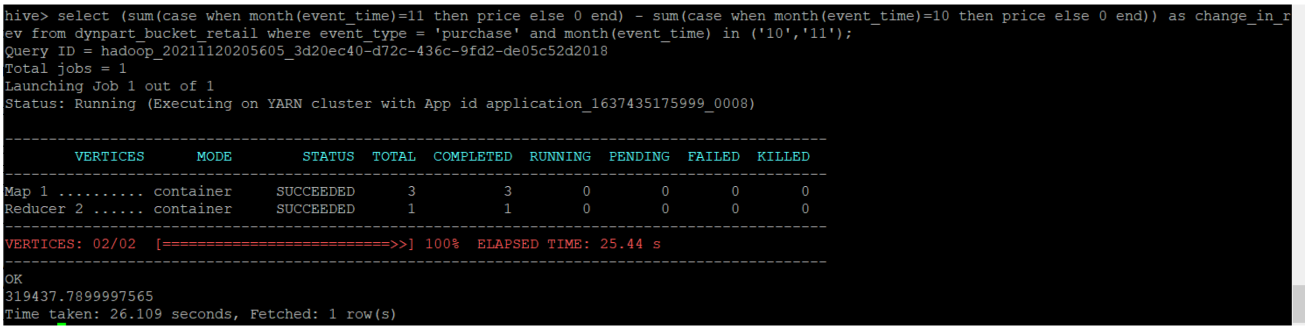


```
hive> select (sum(case when month(event_time)=11 then price else 0 end) - sum(case when month(event_time)=10 then price else 0 end)) as change_in_
ev from retail where event_type = 'purchase' and month(event_time) in ('10','11');
Query ID = hadoop_20211120205309_820e6dba-af39-438b-8830-a126b905d03f
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_1637435175999_0008)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      5          5        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 102.63 s
----------------------------------------------------------------------------------------------
OK
319478.4700003781
Time taken: 112.009 seconds, Fetched: 1 row(s)
```

<u>With Optimization (Dynpart_bucket_retail Table)</u>

Command:

select (sum(case when month(event_time)=11 then price else 0 end) - sum(case when month(event_time)=10 then price else 0 end)) as change_in_rev from dynpart_bucket_retail where event_type = 'purchase' and month(event_time) in ('10','11');



```
hive> select (sum(case when month(event_time)=11 then price else 0 end) - sum(case when month(event_time)=10 then price else 0 end)) as change_in_r
ev from dynpart_bucket_retail where event_type = 'purchase' and month(event_time) in ('10','11');
Query ID = hadoop_20211120205605_3d20ec40-d72c-436c-9fd2-de05c52d2018
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1637435175999_0008)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 25.44 s
----------------------------------------------------------------------------------------------
OK
319437.7899997565
Time taken: 26.109 seconds, Fetched: 1 row(s)
```

Insights:

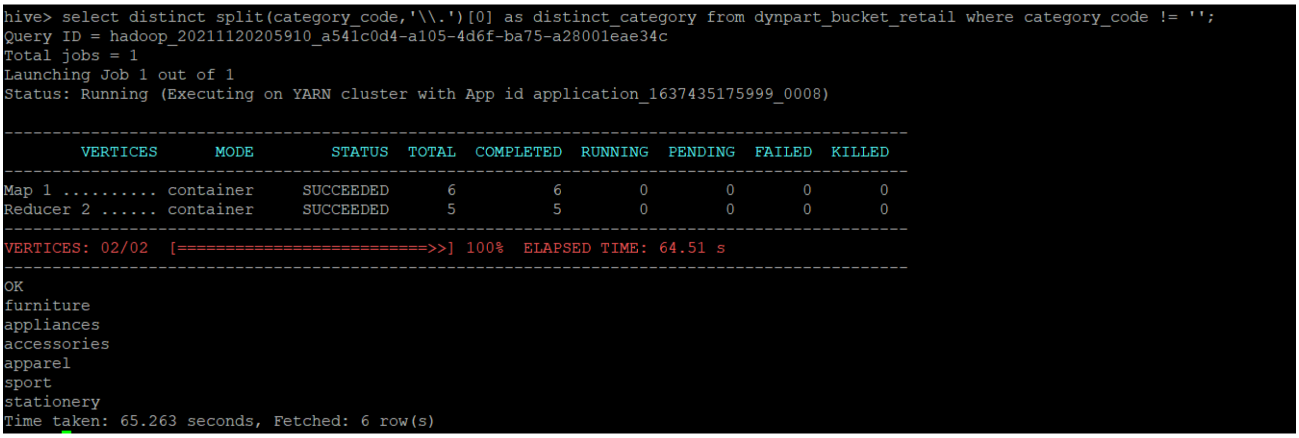
1. In this next question, we will find out on the differences in the revenue generated based on purchases for the month of October and November.
2. The difference of revenue generated between the month of October and November 2019 are 319478.47 +/-.
3. It took 112.009 second for the Retail table to read the data from the above query, however only 26.109 seconds time taken from the optimization table. Again, the Optimized table performed faster in this analysis. Hence, for the next remaining questions, we will be using the optimized tables to run the queries.

Question 4: Find distinct categories of products. Categories with null category code can be ignored.

With Optimization (Dynpart_bucket_retail Table)

Command:

select distinct split(category_code,'\\.')[0] as distinct_category from dynpart_bucket_retail where category_code != '';



```
hive> select distinct split(category_code,'\\.')[0] as distinct_category from dynpart_bucket_retail where category_code != '';
Query ID = hadoop_20211120205910_a541c0d4-a105-4d6f-ba75-a28001eae34c
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1637435175999_0008)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      6          6        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      5          5        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 64.51 s
----------------------------------------------------------------------------------------------
OK
furniture
appliances
accessories
apparel
sport
stationery
Time taken: 65.263 seconds, Fetched: 6 row(s)
```

Insights:

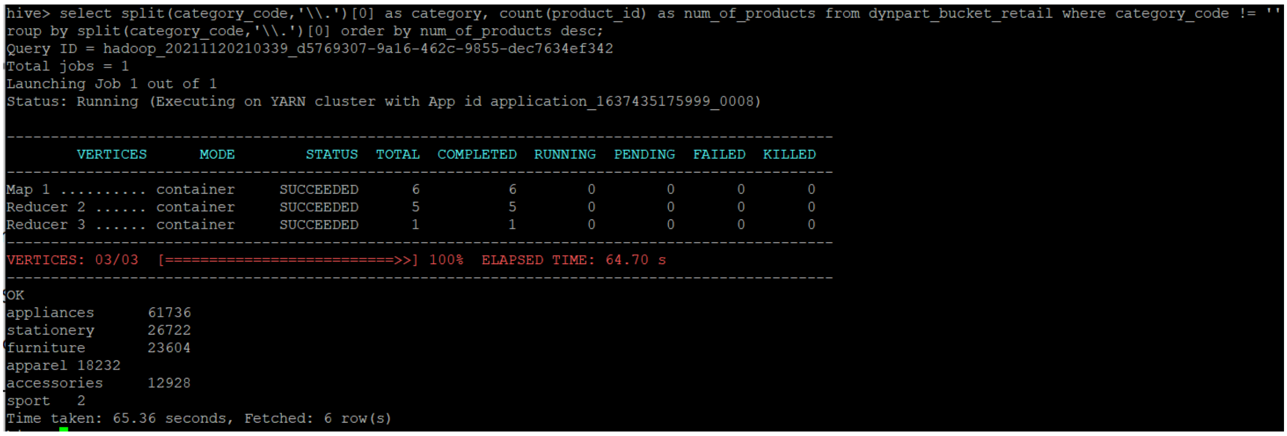
1. The time take to retrieve the data to execute the query is 65.263 seconds.
2. This question required us to find out the categories present in the data, and there are total of 6 categories available, which are:
    - Furniture
    - Appliances
    - Accessories
    - Sport
    - Apparel
    - Stationery

Question 5: Find the total number of products available under each category

<u>With Optimization (Dynpart_bucket_retail Table)</u>

Command:

select split(category_code,'\\.')[0] as category, count(product_id) as num_of_products from dynpart_bucket_retail where category_code != '' group by split(category_code,'\\.')[0] order by num_of_products desc;



```
hive> select split(category_code,'\\.')[0] as category, count(product_id) as num_of_products from dynpart_bucket_retail where category_code != ''
roup by split(category_code,'\\.')[0] order by num_of_products desc;
Query ID = hadoop_20211120210339_d5769307-9a16-462c-9855-dec7634ef342
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1637435175999_0008)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      6          6        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      5          5        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 64.70 s
----------------------------------------------------------------------------------------------
OK
appliances      61736
stationery      26722
furniture       23604
apparel 18232
accessories     12928
sport   2
Time taken: 65.36 seconds, Fetched: 6 row(s)
```

Insights:

1. Based from the categories listed from the previous question, we are required to find the total number of products in each of the 6 categories.
2. To execute this query, it took 65.36 seconds.
3. And the output are as follows:

No.	Category	Number of Products
1.	Appliances	61736
2.	Stationery	26722
3.	Furniture	23604
4.	Apparel	18232
5.	Accessories	12928
6.	Sports	2

Appliances lead the total number of products with 61736, followed closely by Stationery and Furniture. Surprisingly Sports only consists of 2 products.

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

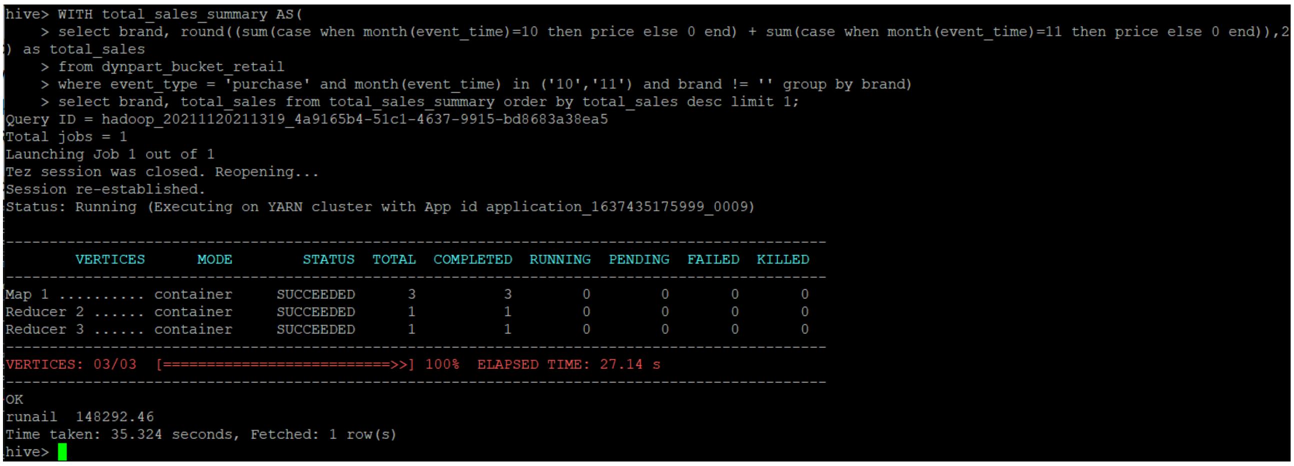
With Optimization (Dynpart_bucket_retail Table)

Command:

WITH total_sales_summary AS(

select brand, round((sum(case when month(event_time)=10 then price else 0 end) + sum(case when month(event_time)=11 then price else 0 end)),2) as total_sales from dynpart_bucket_retail where event_type = ‘purchase’ and month(event_time) in (‘10’,’11’) and brand != ‘’ group by brand)

select brand, total_sales from total_sales_summary order by total_sales desc limit 1;



```
hive> WITH total_sales_summary AS(
    > select brand, round((sum(case when month(event_time)=10 then price else 0 end) + sum(case when month(event_time)=11 then price else 0 end)),2
) as total_sales
    > from dynpart_bucket_retail
    > where event_type = 'purchase' and month(event_time) in ('10','11') and brand != '' group by brand)
    > select brand, total_sales from total_sales_summary order by total_sales desc limit 1;
Query ID = hadoop_20211120211319_4a9165b4-51c1-4637-9915-bd8683a38ea5
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_1637435175999_0009)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 27.14 s
----------------------------------------------------------------------------------------------
OK
runail  148292.46
Time taken: 35.324 seconds, Fetched: 1 row(s)
hive> 
```

Insights:

1. In this question, we are required to find the name of the brand which had the total number of sales for BOTH month of October and November (combine).
2. The time taken to retrieve the data from the above query is only 35.324 seconds.
3. It turns out that brand Runail possessed the highest sales for the month of October AND November.

Question 7: Which brands increased their sales from October to November?

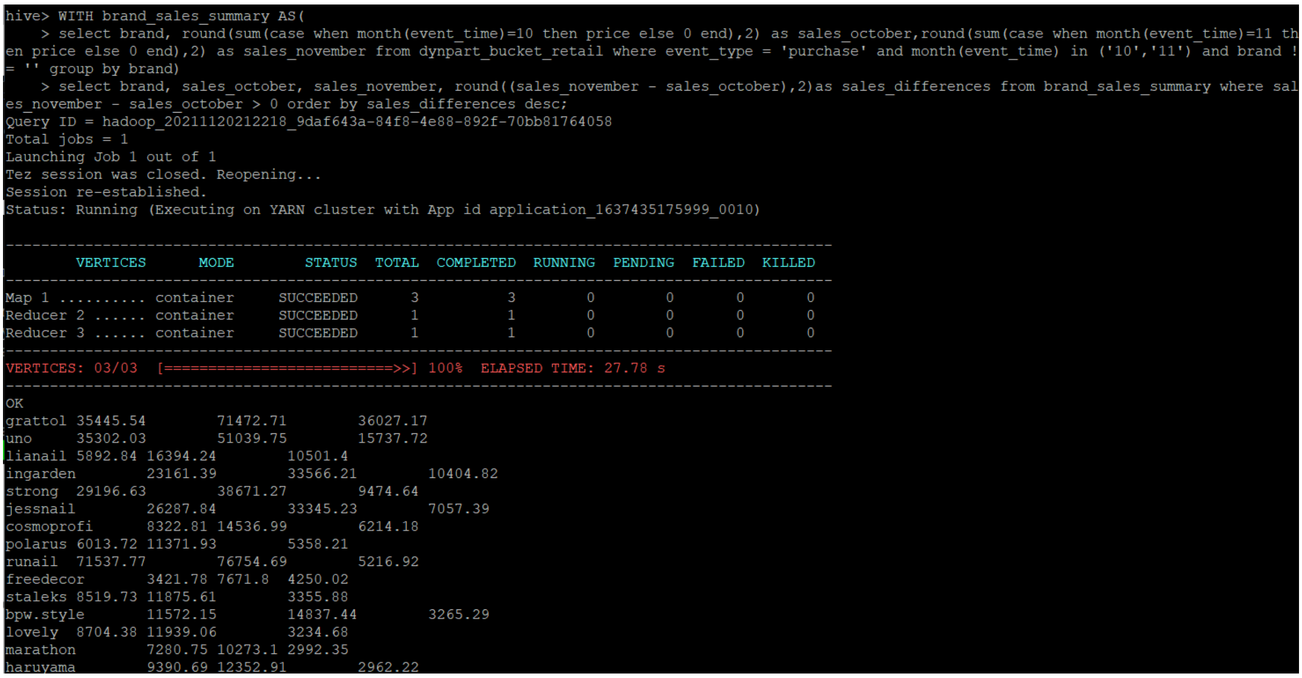
With Optimization (Dynpart_bucket_retail Table)

Command:

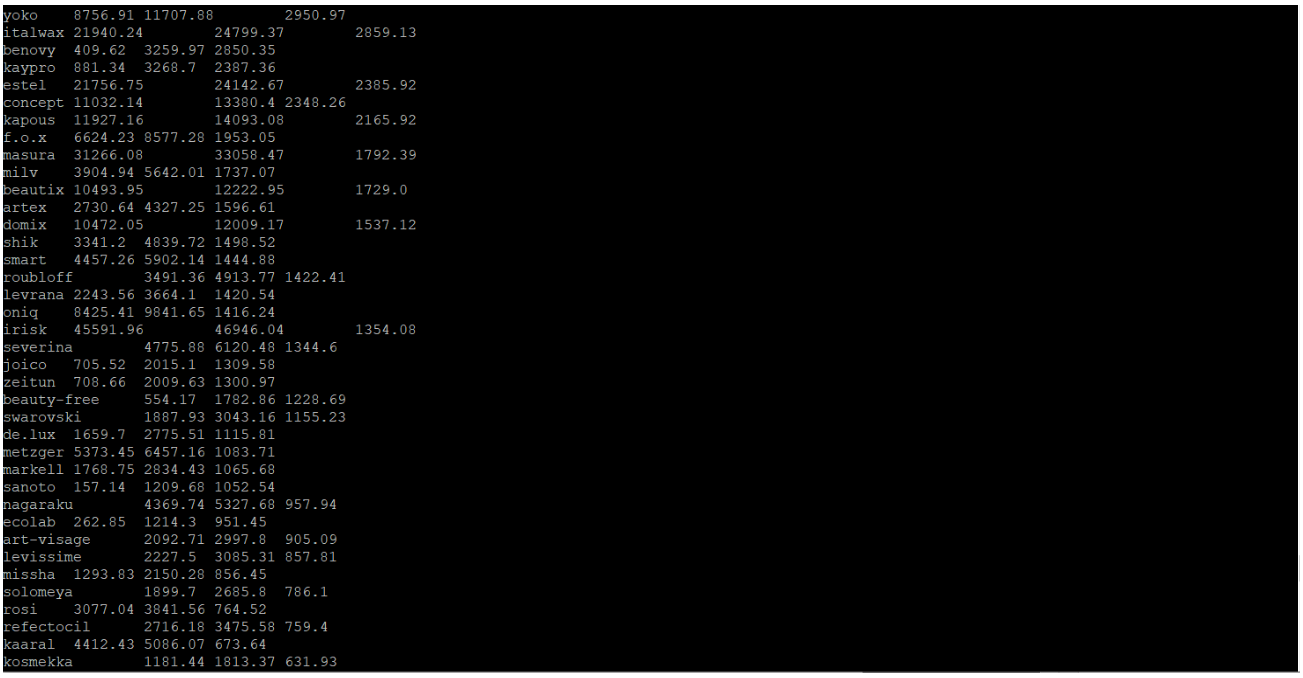
WITH brand_sales_summary AS(

select brand, round(sum(case when month(event_time)=10 then price else 0 end),2) as sales_october,round(sum(case when month(event_time)=11 then price else 0 end),2) as sales_november from dynpart_bucket_retail where event_type = ‘purchase’ and month(event_time) in (‘10’,’11’) and brand != ‘’ group by brand)

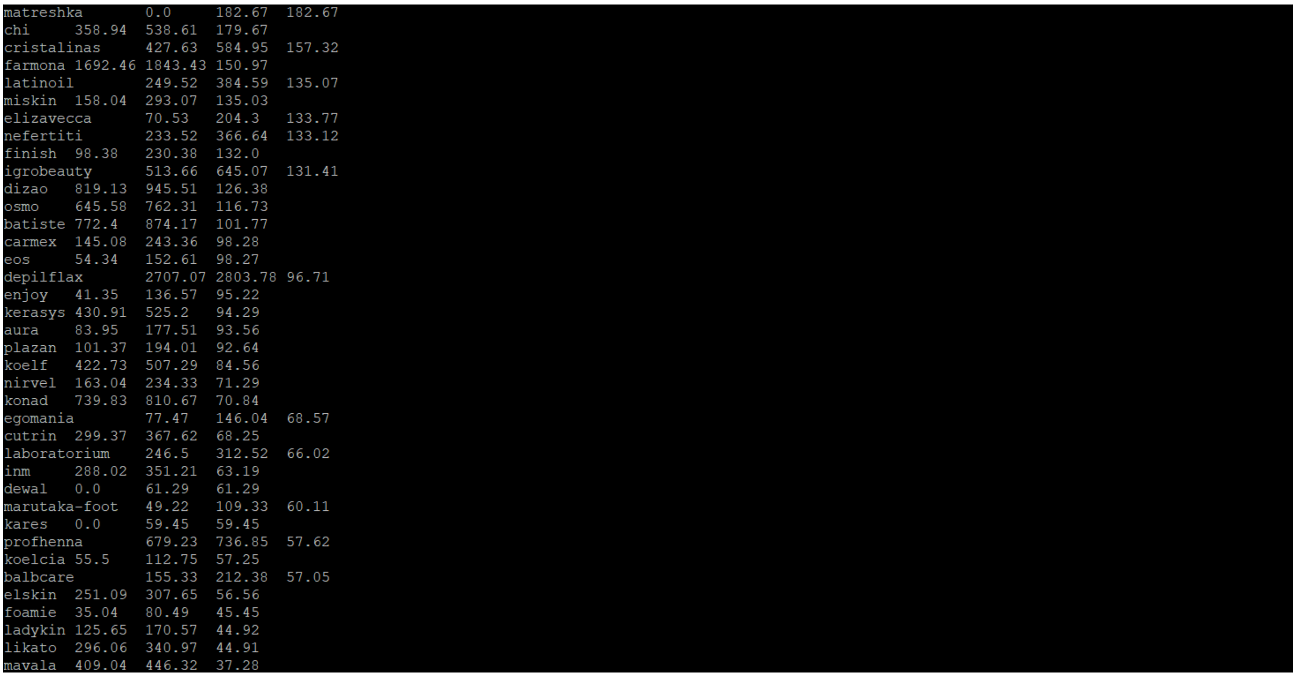
select brand, sales_october, sales_november, round((sales_november – sales_october),2)as sales_differences from brand_sales_summary where sales_november – sales_october > 0 order by sales_differences desc;



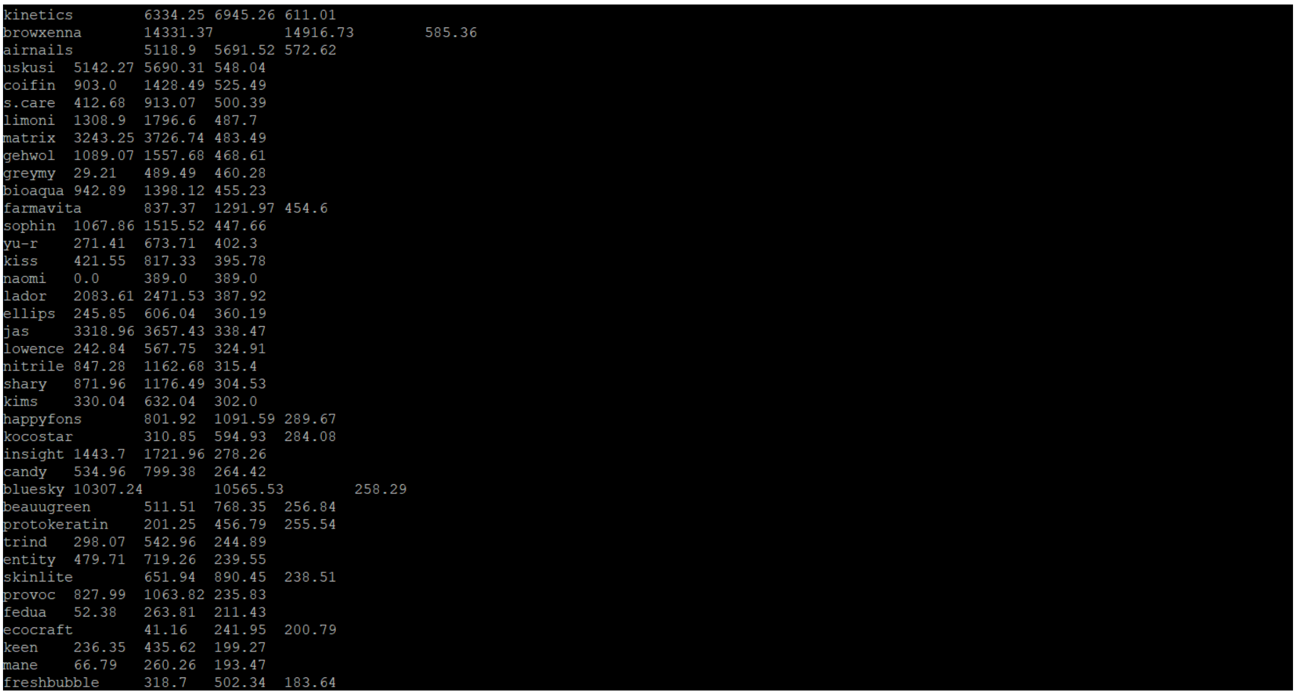
```
hive> WITH brand_sales_summary AS(
    > select brand, round(sum(case when month(event_time)=10 then price else 0 end),2) as sales_october,round(sum(case when month(event_time)=11 th
en price else 0 end),2) as sales_november from dynpart_bucket_retail where event_type = 'purchase' and month(event_time) in ('10','11') and brand !
= '' group by brand)
    > select brand, sales_october, sales_november, round((sales_november - sales_october),2)as sales_differences from brand_sales_summary where sal
es_november - sales_october > 0 order by sales_differences desc;
Query ID = hadoop_20211120212218_9daf643a-84f8-4e88-892f-70bb81764058
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_1637435175999_0010)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 27.78 s
----------------------------------------------------------------------------------------------
OK
grattol 35445.54        71472.71        36027.17
uno     35302.03        51039.75        15737.72
lianail 5892.84 16394.24        10501.4
ingarden        23161.39        33566.21        10404.82
strong  29196.63        38671.27        9474.64
jessnail        26287.84        33345.23        7057.39
cosmoprofi      8322.81 14536.99        6214.18
polarus 6013.72 11371.93        5358.21
runail  71537.77        76754.69        5216.92
freedecor       3421.78 7671.8  4250.02
staleks 8519.73 11875.61        3355.88
bpw.style       11572.15        14837.44        3265.29
lovely  8704.38 11939.06        3234.68
marathon        7280.75 10273.1 2992.35
haruyama        9390.69 12352.91        2962.22
```



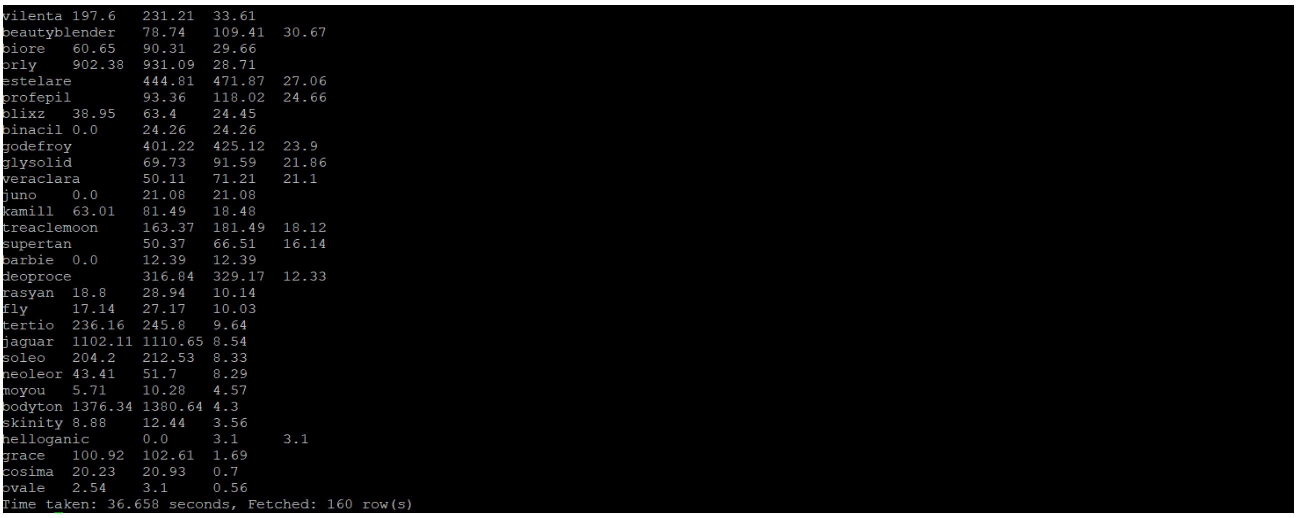
```
yoko    8756.91 11707.88        2950.97
italwax 21940.24        24799.37        2859.13
benovy  409.62  3259.97 2850.35
kaypro  881.34  3268.7  2387.36
estel   21756.75        24142.67        2385.92
concept 11032.14        13380.4 2348.26
kapous  11927.16        14093.08        2165.92
f.o.x   6624.23 8577.28 1953.05
masura  31266.08        33058.47        1792.39
milv    3904.94 5642.01 1737.07
beautix 10493.95        12222.95        1729.0
artex   2730.64 4327.25 1596.61
domix   10472.05        12009.17        1537.12
shik    3341.2  4839.72 1498.52
smart   4457.26 5902.14 1444.88
roubloff        3491.36 4913.77 1422.41
levrana 2243.56 3664.1  1420.54
oniq    8425.41 9841.65 1416.24
irisk   45591.96        46946.04        1354.08
severina        4775.88 6120.48 1344.6
joico   705.52  2015.1  1309.58
zeitun  708.66  2009.63 1300.97
beauty-free     554.17  1782.86 1228.69
swarovski       1887.93 3043.16 1155.23
de.lux  1659.7  2775.51 1115.81
metzger 5373.45 6457.16 1083.71
markell 1768.75 2834.43 1065.68
sanoto  157.14  1209.68 1052.54
nagaraku        4369.74 5327.68 957.94
ecolab  262.85  1214.3  951.45
art-visage      2092.71 2997.8  905.09
levissime       2227.5  3085.31 857.81
missha  1293.83 2150.28 856.45
solomeya        1899.7  2685.8  786.1
rosi    3077.04 3841.56 764.52
refectocil      2716.18 3475.58 759.4
kaaral  4412.43 5086.07 673.64
kosmekka        1181.44 1813.37 631.93
```



```
matreshka       0.0     182.67  182.67
chi     358.94  538.61  179.67
cristalinas     427.63  584.95  157.32
farmona 1692.46 1843.43 150.97
latinoil        249.52  384.59  135.07
miskin  158.04  293.07  135.03
elizavecca      70.53   204.3   133.77
nefertiti       233.52  366.64  133.12
finish  98.38   230.38  132.0
igrobeauty      513.66  645.07  131.41
dizao   819.13  945.51  126.38
osmo    645.58  762.31  116.73
batiste 772.4   874.17  101.77
carmex  145.08  243.36  98.28
eos     54.34   152.61  98.27
depilflax       2707.07 2803.78 96.71
enjoy   41.35   136.57  95.22
kerasys 430.91  525.2   94.29
aura    83.95   177.51  93.56
plazan  101.37  194.01  92.64
koelf   422.73  507.29  84.56
nirvel  163.04  234.33  71.29
konad   739.83  810.67  70.84
egomania        77.47   146.04  68.57
cutrin  299.37  367.62  68.25
laboratorium    246.5   312.52  66.02
inm     288.02  351.21  63.19
dewal   0.0     61.29   61.29
marutaka-foot   49.22   109.33  60.11
kares   0.0     59.45   59.45
profhenna       679.23  736.85  57.62
koelcia 55.5    112.75  57.25
balbcare        155.33  212.38  57.05
elskin  251.09  307.65  56.56
foamie  35.04   80.49   45.45
ladykin 125.65  170.57  44.92
likato  296.06  340.97  44.91
mavala  409.04  446.32  37.28
```



```
kinetics        6334.25 6945.26 611.01
browxenna       14331.37        14916.73        585.36
airnails        5118.9  5691.52 572.62
uskusi  5142.27 5690.31 548.04
coifin  903.0   1428.49 525.49
s.care  412.68  913.07  500.39
limoni  1308.9  1796.6  487.7
matrix  3243.25 3726.74 483.49
gehwol  1089.07 1557.68 468.61
greymy  29.21   489.49  460.28
bioaqua 942.89  1398.12 455.23
farmavita       837.37  1291.97 454.6
sophin  1067.86 1515.52 447.66
yu-r    271.41  673.71  402.3
kiss    421.55  817.33  395.78
naomi   0.0     389.0   389.0
lador   2083.61 2471.53 387.92
ellips  245.85  606.04  360.19
jas     3318.96 3657.43 338.47
lowence 242.84  567.75  324.91
nitrile 847.28  1162.68 315.4
shary   871.96  1176.49 304.53
kims    330.04  632.04  302.0
happyfons       801.92  1091.59 289.67
kocostar        310.85  594.93  284.08
insight 1443.7  1721.96 278.26
candy   534.96  799.38  264.42
bluesky 10307.24        10565.53        258.29
beauugreen      511.51  768.35  256.84
protokeratin    201.25  456.79  255.54
trind   298.07  542.96  244.89
entity  479.71  719.26  239.55
skinlite        651.94  890.45  238.51
provoc  827.99  1063.82 235.83
fedua   52.38   263.81  211.43
ecocraft        41.16   241.95  200.79
keen    236.35  435.62  199.27
mane    66.79   260.26  193.47
freshbubble     318.7   502.34  183.64
```



```
vilenta 197.6   231.21  33.61
beautyblender   78.74   109.41  30.67
biore   60.65   90.31   29.66
orly    902.38  931.09  28.71
estelare        444.81  471.87  27.06
profepil        93.36   118.02  24.66
blixz   38.95   63.4    24.45
binacil 0.0     24.26   24.26
godefroy        401.22  425.12  23.9
glysolid        69.73   91.59   21.86
veraclara       50.11   71.21   21.1
juno    0.0     21.08   21.08
kamill  63.01   81.49   18.48
treaclemoon     163.37  181.49  18.12
supertan        50.37   66.51   16.14
barbie  0.0     12.39   12.39
deoproce        316.84  329.17  12.33
rasyan  18.8    28.94   10.14
fly     17.14   27.17   10.03
tertio  236.16  245.8   9.64
jaguar  1102.11 1110.65 8.54
soleo   204.2   212.53  8.33
neoleor 43.41   51.7    8.29
moyou   5.71    10.28   4.57
bodyton 1376.34 1380.64 4.3
skinity 8.88    12.44   3.56
helloganic      0.0     3.1     3.1
grace   100.92  102.61  1.69
cosima  20.23   20.93   0.7
ovale   2.54    3.1     0.56
Time taken: 36.658 seconds, Fetched: 160 row(s)
```

Insights:

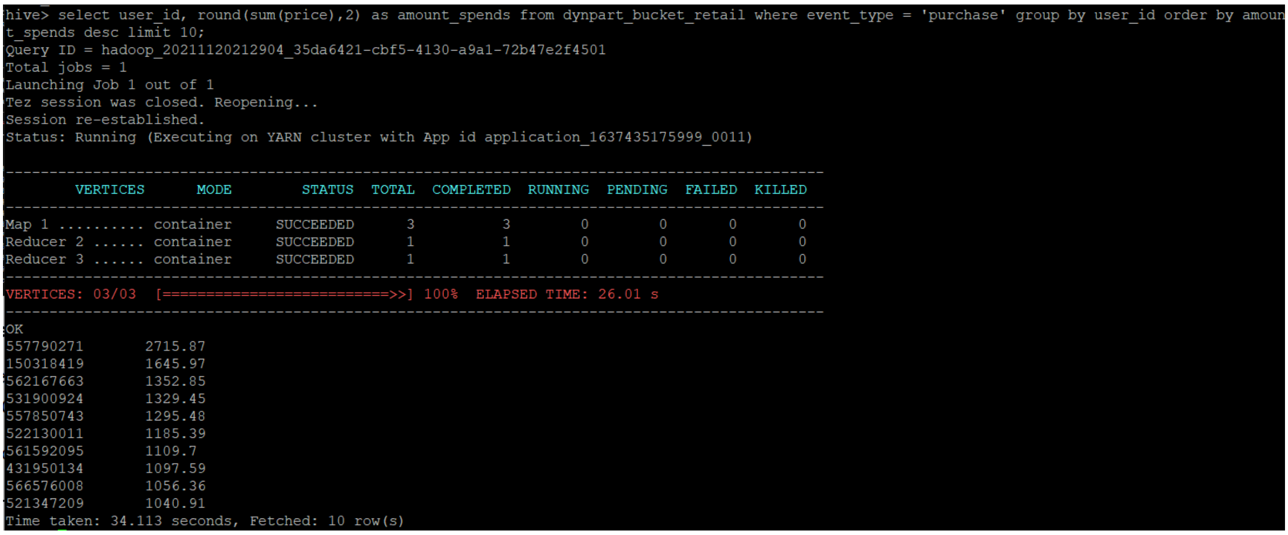
1. To answer this question, we need to find the name of the brand which has increased their sales from the month of October and November.
2. It took only 36.658 seconds to retrieve the data from the Optimized table.
3. From the output, we can see that there are the total of 160 rows, which means, there are the total of 160 brands which has increased their sales in the subsequent month.
4. Grattol if the leading brand with the total increment of 2x which is 36027.17 increment in both of the month, followed closely by Uno brand with 15737.32 and Lianail with 10501.40. The least brand which has the lowest differences in Sales for the month of October and November are Ovale with 0.56, Cosima 0.70 and Grace with 1.69.

Question 8: Your company wants to reward the top 10 users of its websites with a golden customer plan. Write a query to generate a list of top 10 users who spend the most.

With Optimization (Dynpart_bucket_retail Table)

Command:

select user_id, round(sum(price),2) as amount_spends from dynpart_bucket_retail where event_type = ‘purchase’ group by user_id order by amount_spends desc limit 10;



```
hive> select user_id, round(sum(price),2) as amount_spends from dynpart_bucket_retail where event_type = 'purchase' group by user_id order by amoun
t_spends desc limit 10;
Query ID = hadoop_20211120212904_35da6421-cbf5-4130-a9a1-72b47e2f4501
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_1637435175999_0011)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 26.01 s
----------------------------------------------------------------------------------------------
OK
557790271       2715.87
150318419       1645.97
562167663       1352.85
531900924       1329.45
557850743       1295.48
522130011       1185.39
561592095       1109.7
431950134       1097.59
566576008       1056.36
521347209       1040.91
Time taken: 34.113 seconds, Fetched: 10 row(s)
```

Insights:

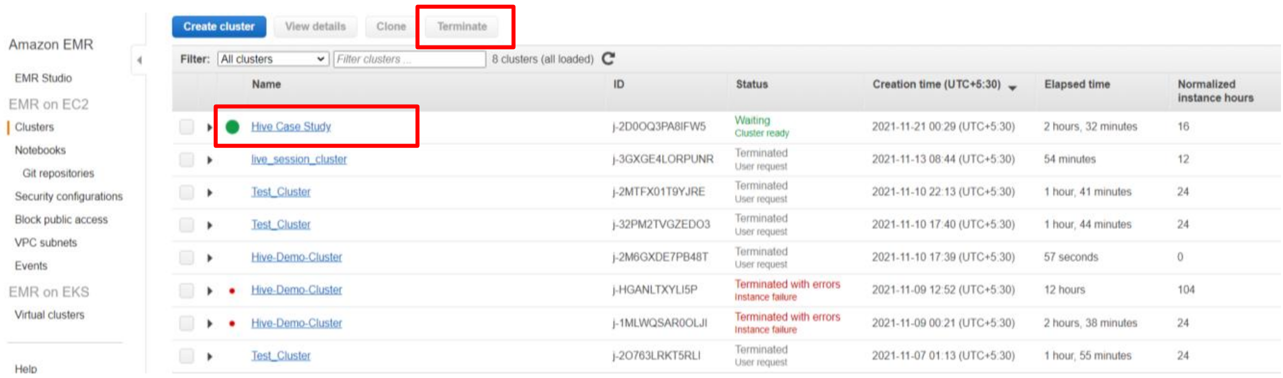
1. We need to find out the Top 10 users who spend the most on the website, which will won the Golden Customer Plan from our company.
2. The time taken for this query to be completed is 34.113 seconds.
3. Based from the output, we can see that the Top user spend 2715.87, followed by 1645.97 and 1352.85.

We have completed all of the questions, which needs to be run on the system, and we can conclude that Partitioning and Bucketing helps in increasing the performance by providing faster analysis on the query despite the huge amount of data provided. And it is efficient and convenient, as well as easy to code despite the volume of data loaded.

STEPS: TERMINATING EMR CLUSTER

It is very important to terminate the cluster, because of the charges that will be charged into our account. Thus, we followed the below procedure to complete the above process:

On AWS Services, type EMR and go back to the main page of our Cluster.

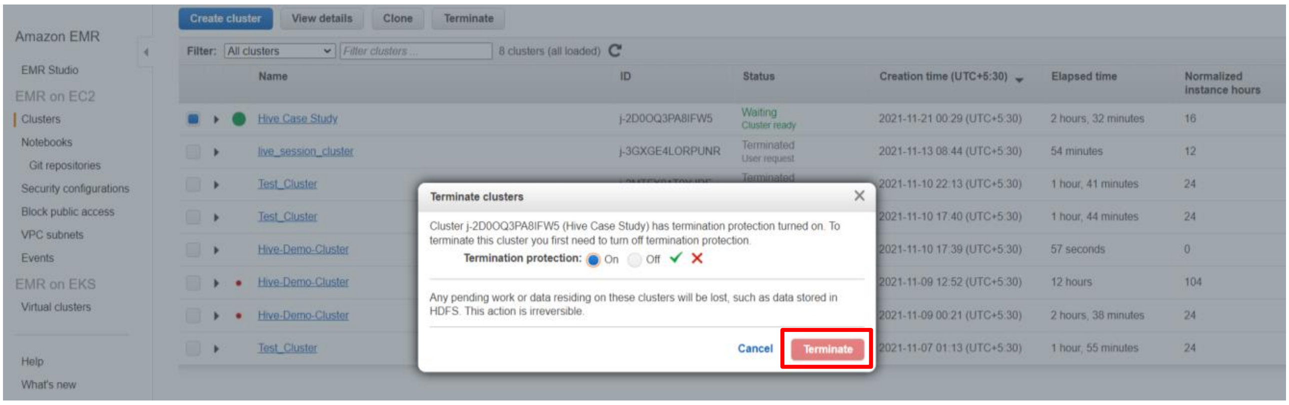


Filter: All clusters | Filter clusters ... | 8 clusters (all loaded)

Name	ID	Status	Creation time (UTC+5:30)	Elapsed time	Normalized instance hours
● Hive Case Study	j-2D0OQ3PA8IFW5	Waiting Cluster ready	2021-11-21 00:29 (UTC+5:30)	2 hours, 32 minutes	16
live_session_cluster	j-3GXGE4LORPUNR	Terminated User request	2021-11-13 08:44 (UTC+5:30)	54 minutes	12
Test_Cluster	j-2MTFX01T9YJRE	Terminated User request	2021-11-10 22:13 (UTC+5:30)	1 hour, 41 minutes	24
Test_Cluster	j-32PM2TVGZEDO3	Terminated User request	2021-11-10 17:40 (UTC+5:30)	1 hour, 44 minutes	24
Hive-Demo-Cluster	j-2M6GXDE7PB48T	Terminated User request	2021-11-10 17:39 (UTC+5:30)	57 seconds	0
● Hive-Demo-Cluster	j-HGANLTXYLI5P	Terminated with errors Instance failure	2021-11-09 12:52 (UTC+5:30)	12 hours	104
● Hive-Demo-Cluster	j-1MLWQSAR0OLJI	Terminated with errors Instance failure	2021-11-09 00:21 (UTC+5:30)	2 hours, 38 minutes	24
Test_Cluster	j-2O763LRKT5RLI	Terminated User request	2021-11-07 01:13 (UTC+5:30)	1 hour, 55 minutes	24

Tick on our cluster, in this case 'Hive_Case_Study' cluster > click on Terminate.

A screen will pop up, we have unticked the 'On' and switch to 'Off' to allow the termination procedure to be completed, then click on 'Terminate'.



Terminate clusters

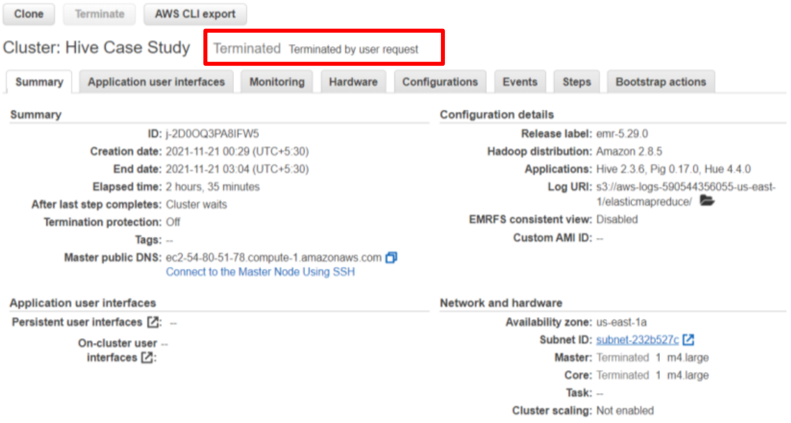
Cluster j-2D0OQ3PA8IFW5 (Hive Case Study) has termination protection turned on. To terminate this cluster you first need to turn off termination protection.

Termination protection: ● On ○ Off ✓ ✗

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

Cancel | Terminate

Our cluster and both of our Master and Core node has been terminated and with that we have completed ur hive Analysis on ClickStream dataset successfully.



Cluster: Hive Case Study Terminated Terminated by user request

Summary

ID: j-2D0OQ3PA8IFW5
Creation date: 2021-11-21 00:29 (UTC+5:30)
End date: 2021-11-21 03:04 (UTC+5:30)
Elapsed time: 2 hours, 35 minutes
After last step completes: Cluster waits
Termination protection: Off
Tags: --
Master public DNS: ec2-54-80-51-78.compute-1.amazonaws.com
Connect to the Master Node Using SSH

Configuration details

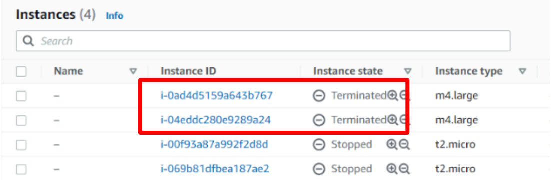
Release label: emr-5.29.0
Hadoop distribution: Amazon 2.8.5
Applications: Hive 2.3.6, Pig 0.17.0, Hue 4.4.0
Log URI: s3://aws-logs-590544356055-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled
Custom AMI ID: --

Application user interfaces

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

Network and hardware

Availability zone: us-east-1a
Subnet ID: subnet-232b527c
Master: Terminated 1 m4.large
Core: Terminated 1 m4.large
Task: --
Cluster scaling: Not enabled



Instances (4) Info

Search

□	Name	Instance ID	Instance state	Instance type
□	–	i-0ad4d5159a643b767	⊖ Terminated	m4.large
□	–	i-04eddc280e9289a24	⊖ Terminated	m4.large
□	–	i-00f93a87a992f2d8d	⊖ Stopped	t2.micro
□	–	i-069b81dfbea187ae2	⊖ Stopped	t2.micro