Saikat Chowdhury
Daivanshu Gandhi

Hive Case Study

E-Commerce Sales Review

Problem Statement: The tech companies are exploring different ways to improve their sales by analysing customer behaviour and gaining insights about product trends, due to the increasing popularity in online sales. Therefore, as a big data analyst, we are expected to extract data and gather insights from a real-life data set of an e-commerce company.

Objective: Need to analyse and gain insights about the clickstream data from a website so that we can extract insights about the customers behaviour.

The steps involved in the entire process are as follows:

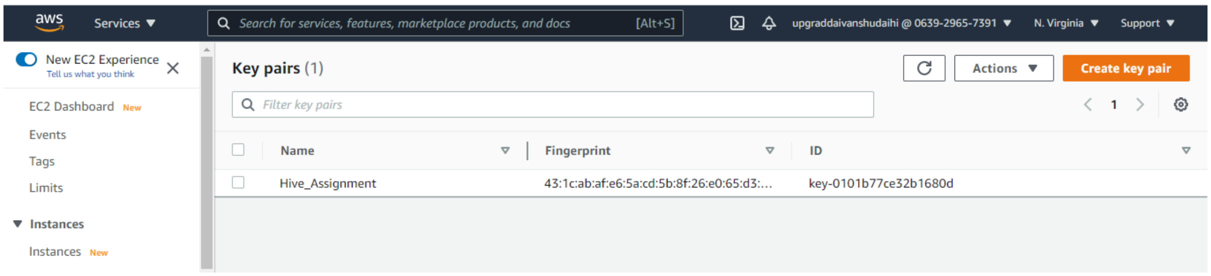
- Copying the data set into the HDFS:
  - Launch an EMR cluster that utilizes the Hive services, and
  - Move the data from the S3 bucket into the HDFS
- Creating the database and launching Hive queries on your EMR cluster:
  - Create the structure of your database,
  - Use optimized techniques to run 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.

Launch the EMR Cluster.

In order to launch an EMR Cluster, the following steps were followed:

- Create a key-pair and download the PEM/ PPK file.

Creation of Key Pair:

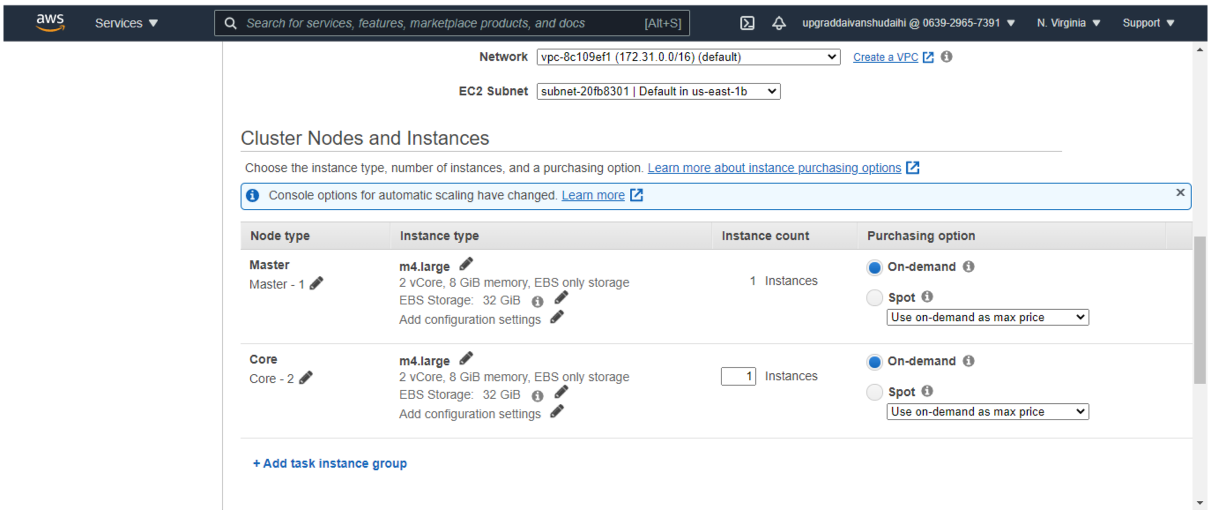


Key pairs (1)

Name	Fingerprint	ID
Hive_Assignment	43:1c:ab:af:e6:5a:cd:5b:8f:26:e0:65:d3:...	key-0101b77ce32b1680d

31/05/2021

- After creation of Key-pair, now we need create an EMR cluster. While creating an EMR cluster need to make sure that we are selecting **m4.large** Master and Core node of single instances. Also need to select the correct key-pair in the security option


aws
Services
Search for services, features, marketplace products, and docs [Alt+S]
upgraddaivanshudaihi @ 0639-2965-7391
N. Virginia
Support

Network vpc-8c109ef1 (172.31.0.0/16) (default) Create a VPC

EC2 Subnet subnet-20fb8301 | Default in us-east-1b

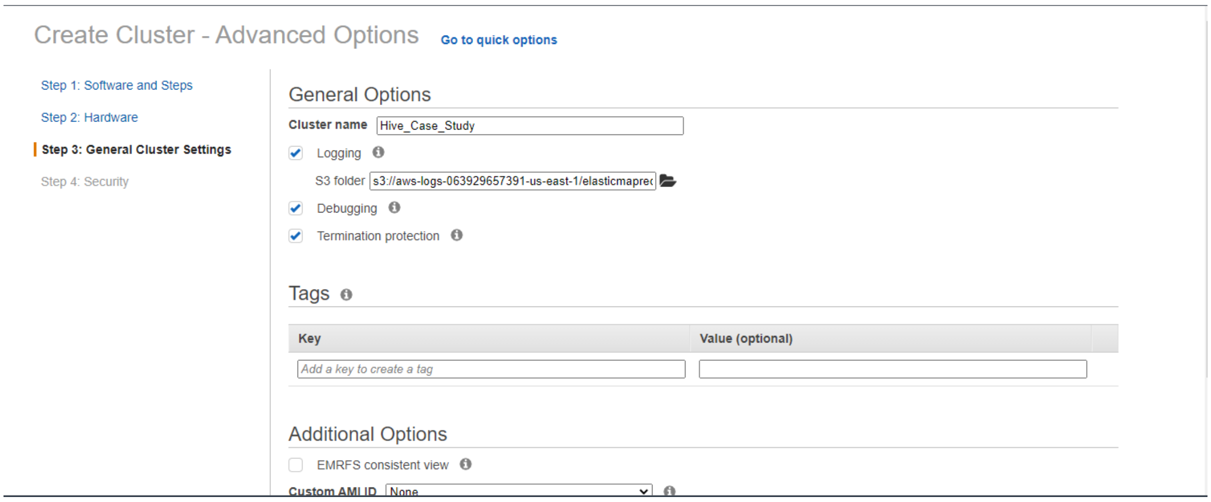
Cluster Nodes and Instances

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

+ Add task instance group


Create Cluster - Advanced Options Go to quick options

General Options

Cluster name Hive_Case_Study

✓ Logging

S3 folder s3://aws-logs-063929657391-us-east-1/elasticmapre

✓ Debugging

✓ Termination protection

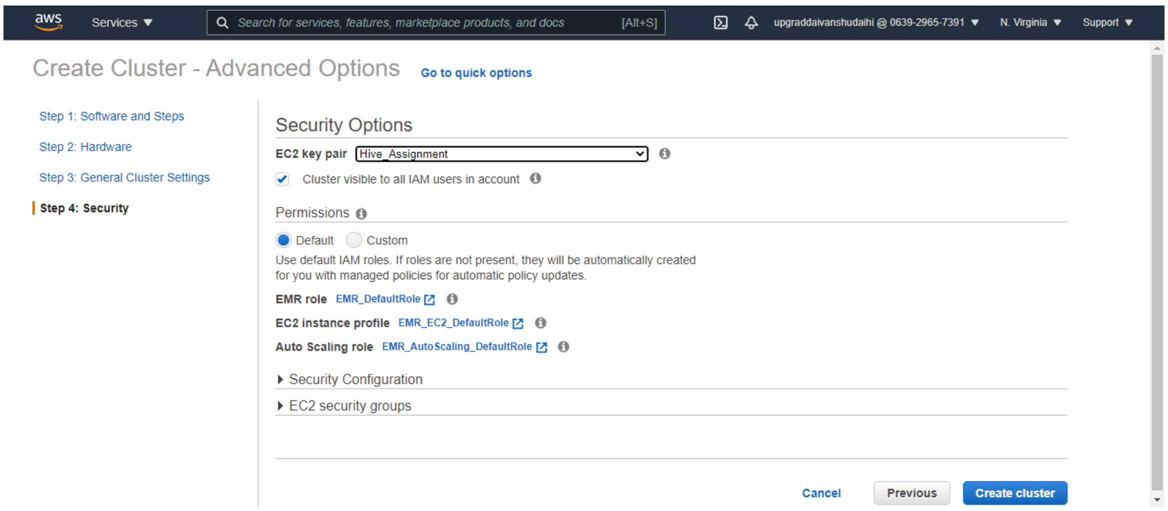
Tags

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

Additional Options

☐ EMRFS consistent view

Custom AMI ID None


aws
Services
Search for services, features, marketplace products, and docs [Alt+S]
upgraddaivanshudaihi @ 0639-2965-7391
N. Virginia
Support

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 Hive_Assignment

✓ 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

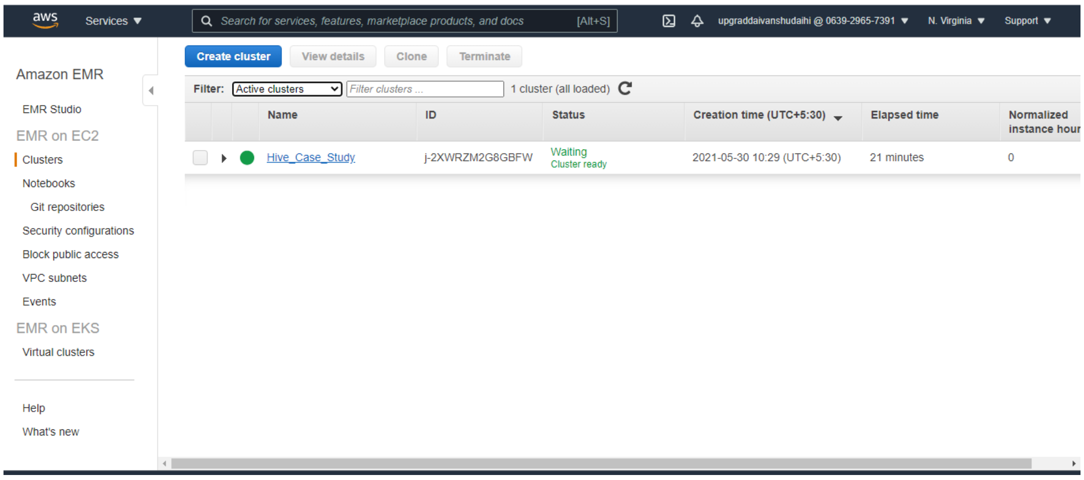
EC2 instance profile EMR_EC2_DefaultRole

Auto Scaling role EMR_AutoScaling_DefaultRole

Security Configuration

EC2 security groups

- Now the cluster is created and we are ready to move to the next stage i.e moving data from S3 to HDFS.

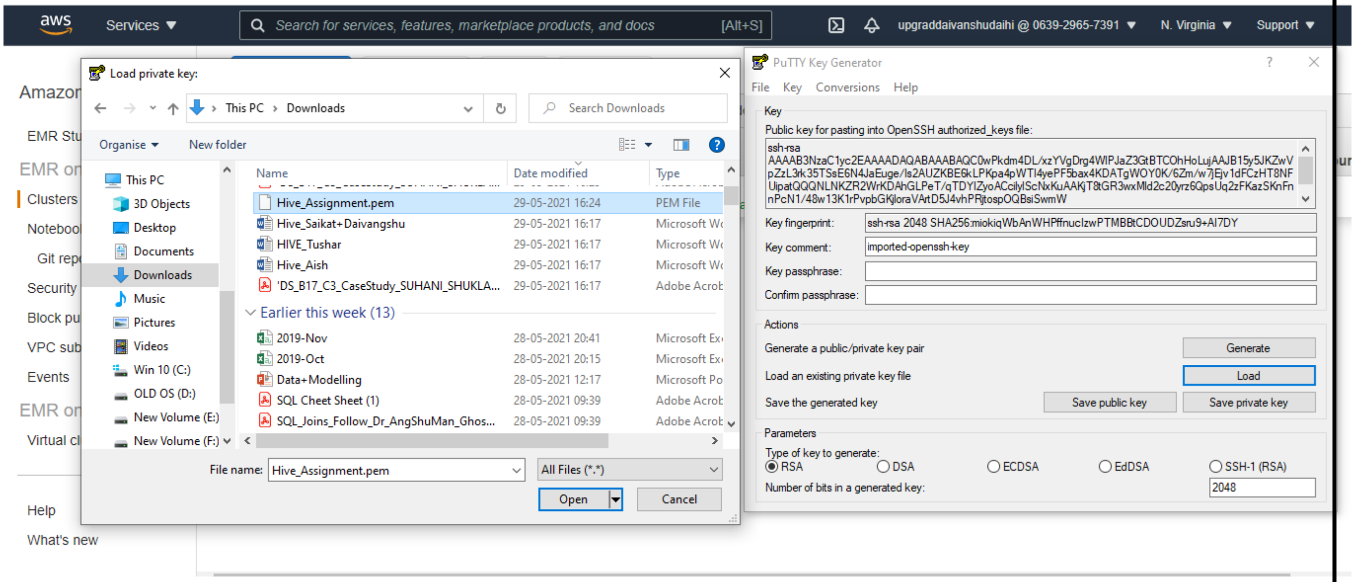


Filter: Active clusters | Filter clusters ... | 1 cluster (all loaded)

Name	ID	Status	Creation time (UTC+5:30)	Elapsed time	Normalized instance hour
Hive_Case_Study	j-2XWRZM2G8GBFW	Waiting Cluster ready	2021-05-30 10:29 (UTC+5:30)	21 minutes	0

Connecting to EMR cluster:

- Here need to open PuTTYgen application for windows machine and we have to Load the previously downloaded .pem Key-pair file and save the private key which is in the extension .ppk



Name	Date modified	Type
Hive_Assignment.pem	29-05-2021 16:24	PEM File
Hive_Saikat+Daivangshu	29-05-2021 16:17	Microsoft W
HIVE_Tushar	29-05-2021 16:17	Microsoft W
Hive_Aish	29-05-2021 16:17	Microsoft W
'DS_B17_C3_CaseStudy_SUHANI_SHUKLA...	29-05-2021 16:17	Adobe Acrob
Earlier this week (13)		
2019-Nov	28-05-2021 20:41	Microsoft Ex
2019-Oct	28-05-2021 20:15	Microsoft Ex
Data+Modelling	28-05-2021 12:17	Microsoft Po
SQL Cheet Sheet (1)	28-05-2021 09:39	Adobe Acrob
SQL_Joins_Follow_Dr_AngShuMan_Ghos...	28-05-2021 09:39	Adobe Acrob

File name: Hive_Assignment.pem | All Files (*.*)

Key
Public key for pasting into OpenSSH authorized_keys file:

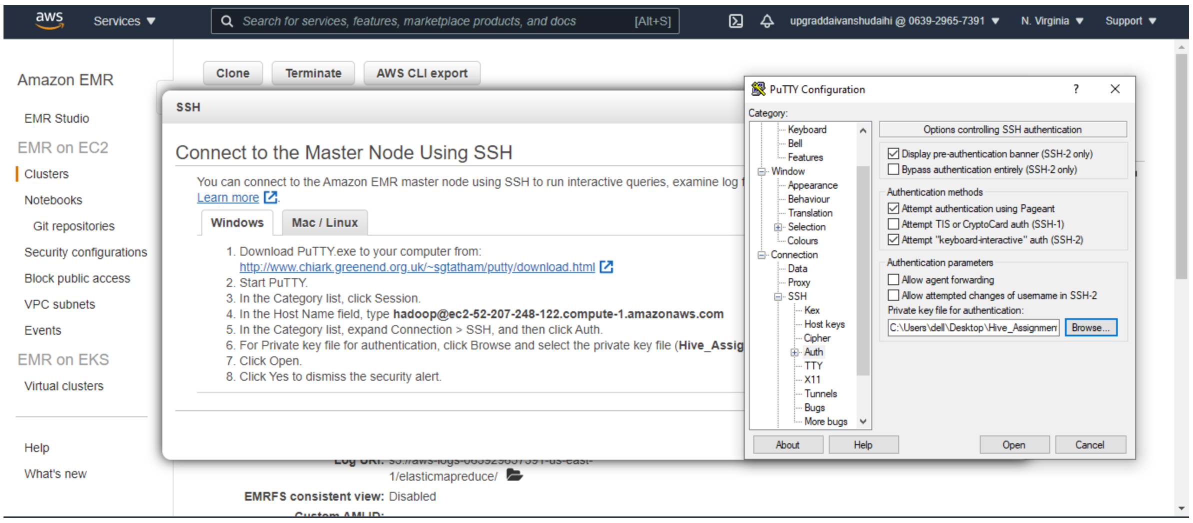
```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQC0wPkdm4DL/xzYVgDrg4WIPJaZ3GtBTCOhHoLujAAJB15y5JKZwV
pZzL3rk35TSsE6N4JaEuge/ls2AUZKBE6kLPKpa4pWTI4yePF5bax4KDATgWOY0K/6Zm/w7jEjv1dFCzHT8NF
UipatQQQNLNKZR2WrKDAhGLPeT/qTDYIZyoACcilylScNxKuAAKjT8tGR3wxMld2c20yrz6QpsUq2zFKazSKnFn
nPcN1/48w13K1rPvpbGKjloraVArtD5J4vhPRjtospOQBsiSwmW
```

Key fingerprint: ssh-rsa 2048 SHA256:miokiqWbAnWHPffnuclzwPTMBBtCDOUDZsru9+AI7DY
Key comment: imported-openssh-key
Key passphrase:
Confirm passphrase:

Actions
Generate a public/private key pair
Load an existing private key file
Save the generated key

Parameters
Type of key to generate:
[✓ RSA] DSA ECDSA EdDSA SSH-1 (RSA)
Number of bits in a generated key: 2048

Once cluster in running state we have to click on Master public DNS. 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. Now we need to open Putty and connect to the master node by selecting the .ppk file.



Connect to the Master Node Using SSH

You can connect to the Amazon EMR master node using SSH to run interactive queries, examine log f
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@ec2-52-207-248-122.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_Assig**
7. Click Open.
8. Click Yes to dismiss the security alert.

Log URI: s3://aws-logs-063929657391-us-east-1/elasticmapreduce/

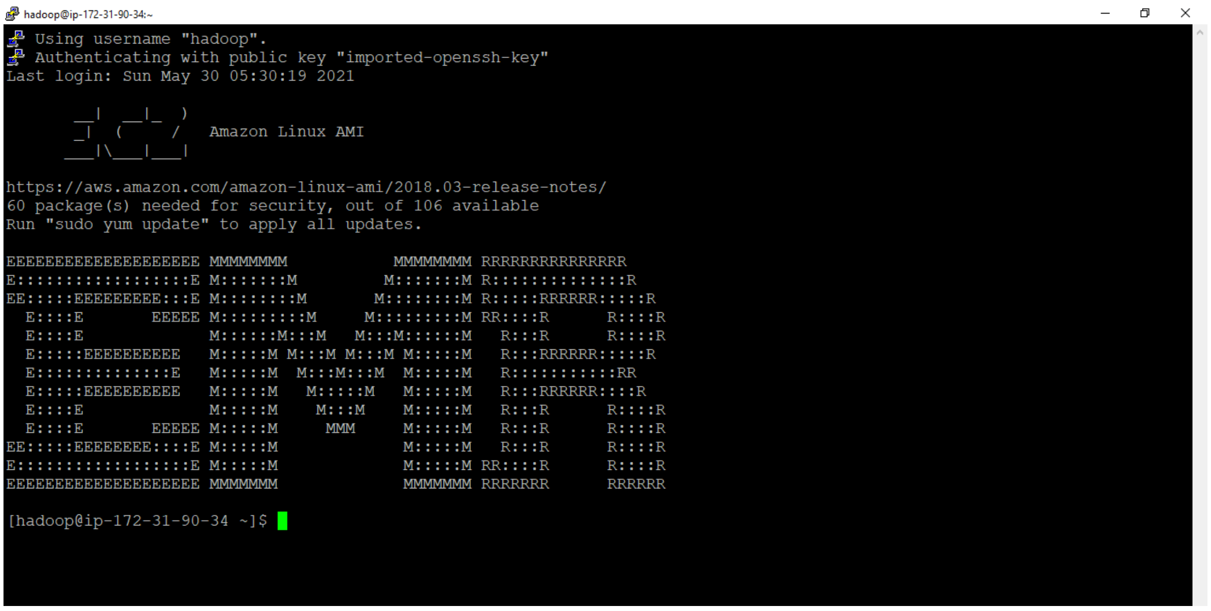
EMRFS consistent view: Disabled

[✓] 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\dell\Desktop\Hive_Assignmen

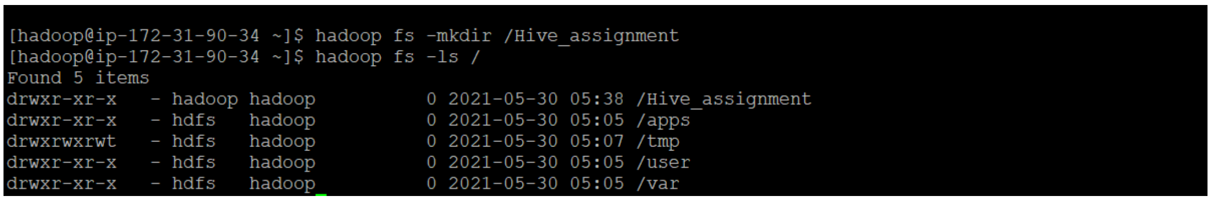
Connection to hadoop is successful:



```
Using username "hadoop".
Authenticating with public key "imported-openssh-key"
Last login: Sun May 30 05:30:19 2021

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

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
60 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-172-31-90-34 ~]$
```

Load the data sets into HDFS from S3:

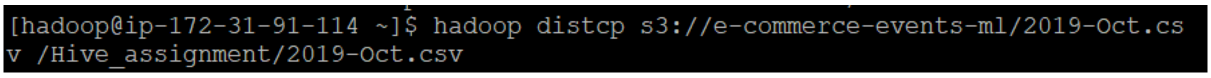
- Create a directory named 'Hive_assignment' in Hadoop.



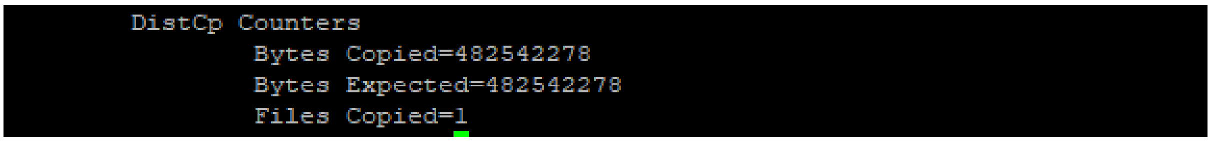
```
[hadoop@ip-172-31-90-34 ~]$ hadoop fs -mkdir /Hive_assignment
[hadoop@ip-172-31-90-34 ~]$ hadoop fs -ls /
Found 5 items
drwxr-xr-x   - hadoop hadoop          0 2021-05-30 05:38 /Hive_assignment
drwxr-xr-x   - hdfs   hadoop          0 2021-05-30 05:05 /apps
drwxrwxrwt   - hdfs   hadoop          0 2021-05-30 05:07 /tmp
drwxr-xr-x   - hdfs   hadoop          0 2021-05-30 05:05 /user
drwxr-xr-x   - hdfs   hadoop          0 2021-05-30 05:05 /var
```

- Move the data from the s3 buckets to the HDFS using the distributed copy command.

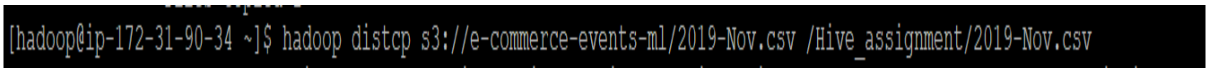
  Loading the s3 public data set to created directory “Hive_assignment” in hadoop .
  Command: **hadoop distcp 's3://e-commerce-events-ml/2019-Oct.csv' / Hive_assignment /2019-Oct.csv**
  hadoop distcp 's3://e-commerce-events-ml/2019-Nov.csv'/Hive_assignment/2019-Nov.csv



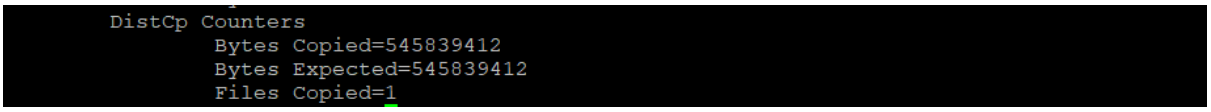
```
[hadoop@ip-172-31-91-114 ~]$ hadoop distcp s3://e-commerce-events-ml/2019-Oct.cs
v /Hive_assignment/2019-Oct.csv
```



```
        DistCp Counters
                Bytes Copied=482542278
                Bytes Expected=482542278
                Files Copied=1
```

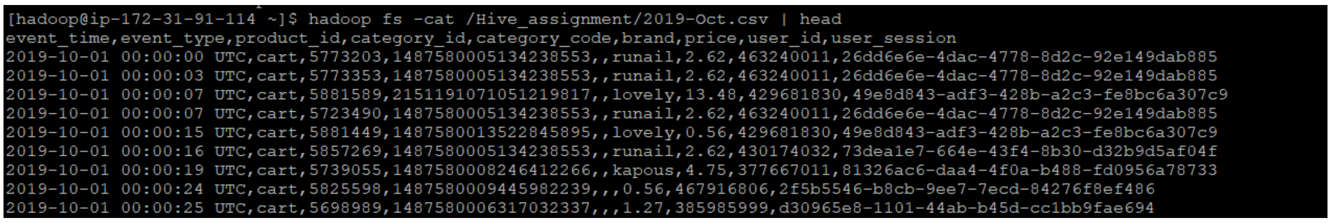


```
[hadoop@ip-172-31-90-34 ~]$ hadoop distcp s3://e-commerce-events-ml/2019-Nov.csv /Hive_assignment/2019-Nov.csv
```

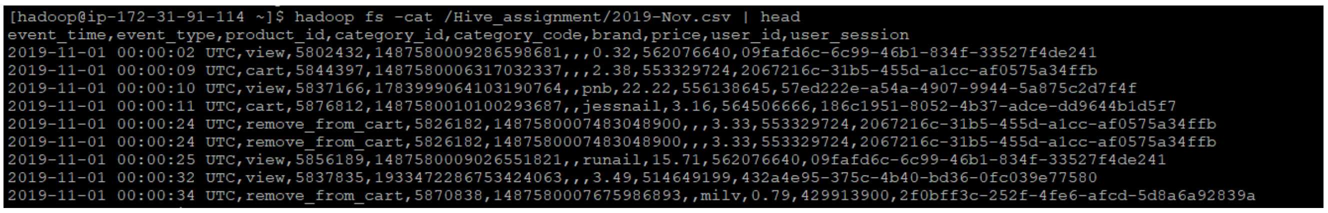


```
        DistCp Counters
                Bytes Copied=545839412
                Bytes Expected=545839412
                Files Copied=1
```

- View the data in HDFS by executing below commands

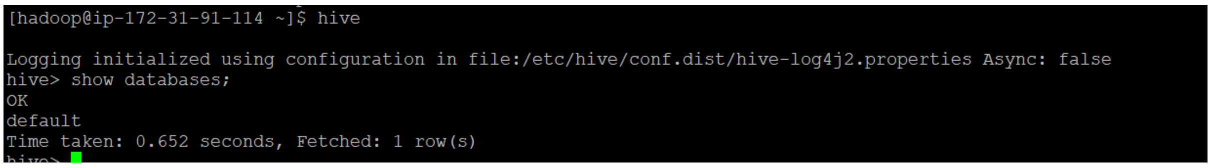


```
[hadoop@ip-172-31-91-114 ~]$ hadoop fs -cat /Hive_assignment/2019-Oct.csv | head
event_time,event_type,product_id,category_id,category_code,brand,price,user_id,user_session
2019-10-01 00:00:00 UTC,cart,5773203,1487580005134238553,,runail,2.62,463240011,26dd6e6e-4dac-4778-8d2c-92e149dab885
2019-10-01 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
```



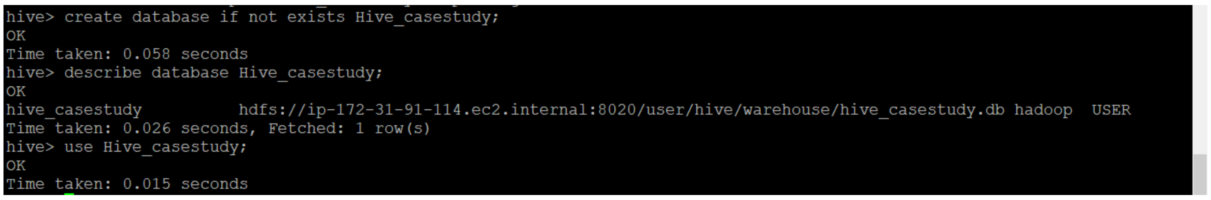
```
[hadoop@ip-172-31-91-114 ~]$ hadoop fs -cat /Hive_assignment/2019-Nov.csv | head
event_time,event_type,product_id,category_id,category_code,brand,price,user_id,user_session
2019-11-01 00:00:02 UTC,view,5802432,1487580009286598681,,,0.32,562076640,09fafd6c-6c99-46b1-834f-33527f4de241
2019-11-01 00:00:09 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-af0575a34ffb
2019-11-01 00:00:24 UTC,remove_from_cart,5826182,1487580007483048900,,,3.33,553329724,2067216c-31b5-455d-a1cc-af0575a34ffb
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-5d8a6a92839a
```

- After successfully adding data, it’s time to Set the data in Hive
- Launch Hive



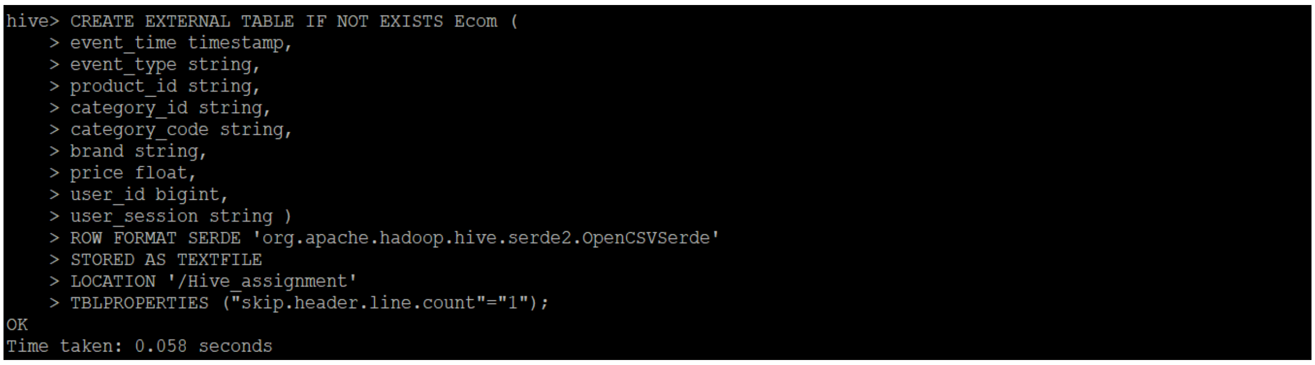
```
[hadoop@ip-172-31-91-114 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive> show databases;
OK
default
Time taken: 0.652 seconds, Fetched: 1 row(s)
hive>
```

- Creating database, creating tables

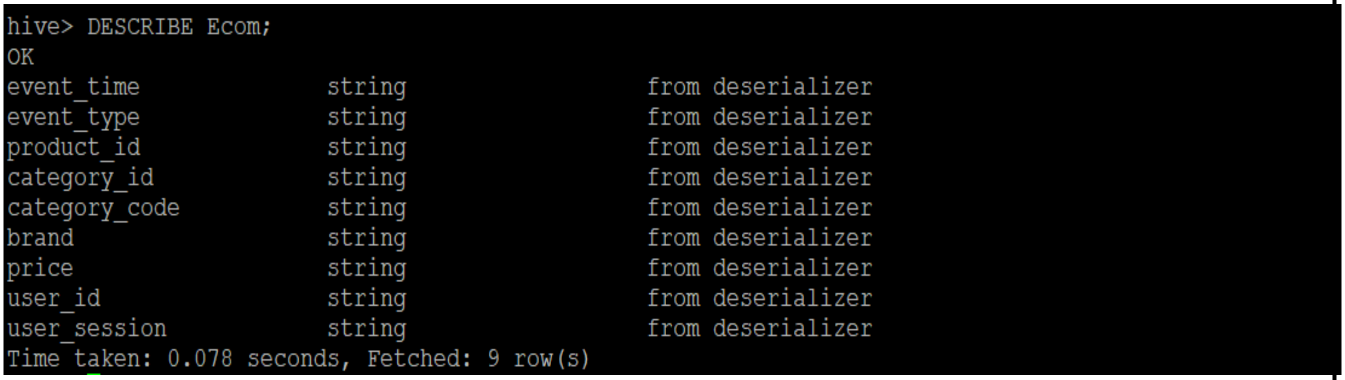


```
hive> create database if not exists Hive_casestudy;
OK
Time taken: 0.058 seconds
hive> describe database Hive_casestudy;
OK
hive_casestudy          hdfs://ip-172-31-91-114.ec2.internal:8020/user/hive/warehouse/hive_casestudy.db hadoop  USER
Time taken: 0.026 seconds, Fetched: 1 row(s)
hive> use Hive_casestudy;
OK
Time taken: 0.015 seconds
```

- Creating a table from the raw data by taking care of the data dictionary.

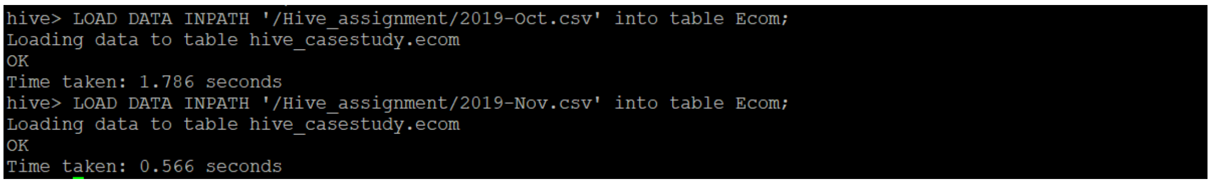


```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS Ecom (
    > event_time timestamp,
    > event_type string,
    > product_id string,
    > category_id string,
    > category_code string,
    > brand string,
    > price float,
    > user_id bigint,
    > user_session string )
    > ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
    > STORED AS TEXTFILE
    > LOCATION '/Hive_assignment'
    > TBLPROPERTIES ("skip.header.line.count"="1");
OK
Time taken: 0.058 seconds
```



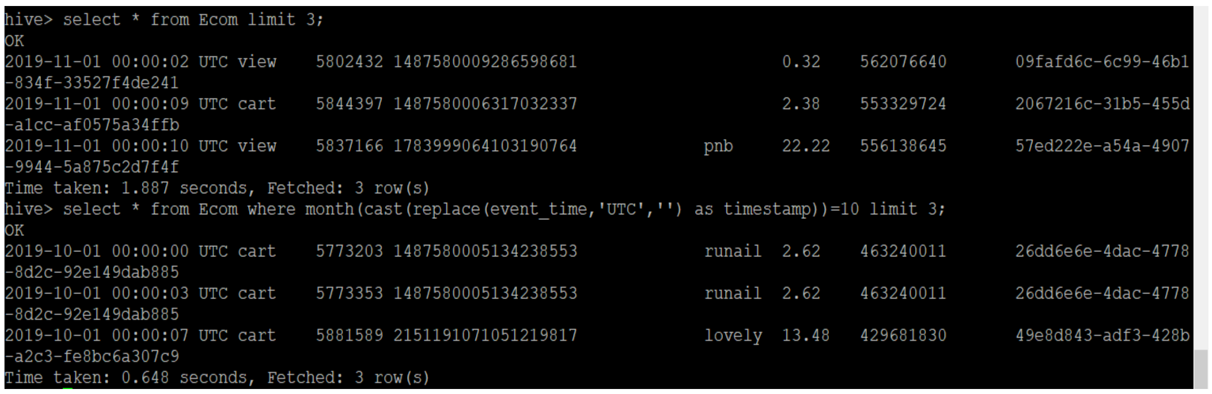
```
hive> DESCRIBE Ecom;
OK
event_time              string                  from deserializer
event_type              string                  from deserializer
product_id              string                  from deserializer
category_id             string                  from deserializer
category_code           string                  from deserializer
brand                   string                  from deserializer
price                   string                  from deserializer
user_id                 string                  from deserializer
user_session            string                  from deserializer
Time taken: 0.078 seconds, Fetched: 9 row(s)
```

- Loading data from both files into this table.



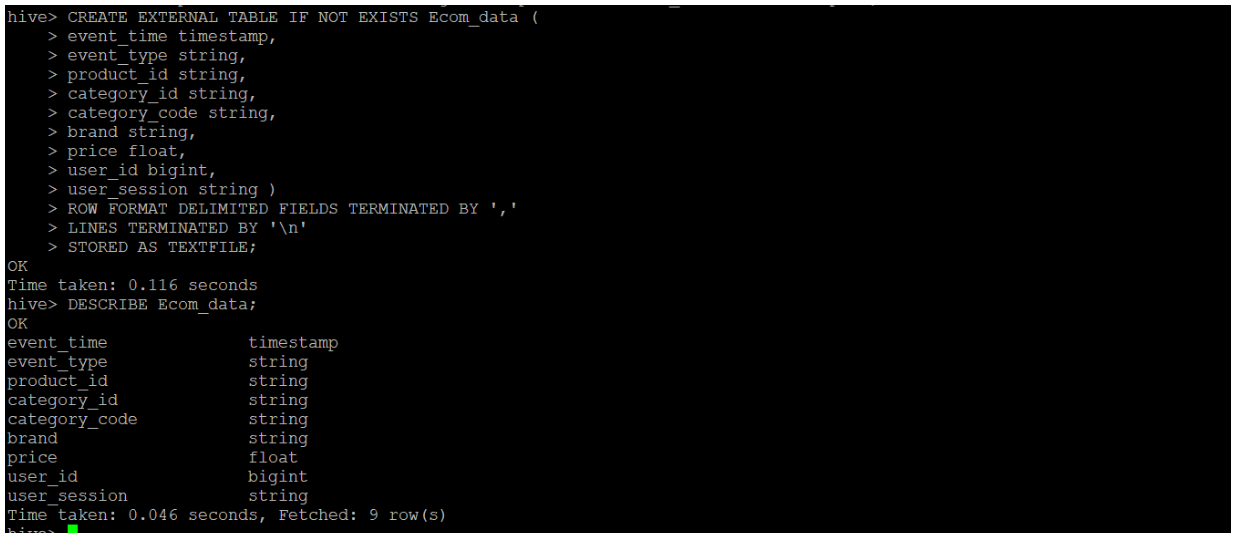
```
hive> LOAD DATA INPATH '/Hive_assignment/2019-Oct.csv' into table Ecom;
Loading data to table hive_casestudy.ecom
OK
Time taken: 1.786 seconds
hive> LOAD DATA INPATH '/Hive_assignment/2019-Nov.csv' into table Ecom;
Loading data to table hive_casestudy.ecom
OK
Time taken: 0.566 seconds
```

- Checking the data after loading them into tables



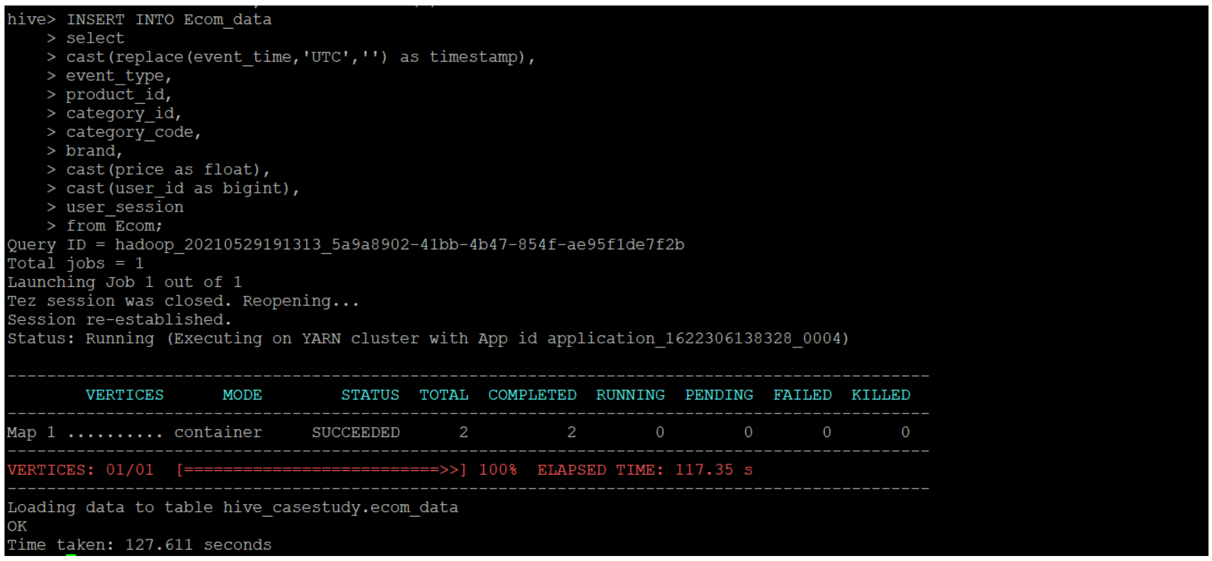
```
hive> select * from Ecom limit 3;
OK
2019-11-01 00:00:02 UTC view    5802432 1487580009286598681             0.32    562076640       09fafd6c-6c99-46b1
-834f-33527f4de241
2019-11-01 00:00:09 UTC cart    5844397 1487580006317032337             2.38    553329724       2067216c-31b5-455d
-a1cc-af0575a34ffb
2019-11-01 00:00:10 UTC view    5837166 1783999064103190764     pnb     22.22   556138645       57ed222e-a54a-4907
-9944-5a875c2d7f4f
Time taken: 1.887 seconds, Fetched: 3 row(s)
hive> select * from Ecom where month(cast(replace(event_time,'UTC','') as timestamp))=10 limit 3;
OK
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
Time taken: 0.648 seconds, Fetched: 3 row(s)
```

- Creating table for data analysis with data in proper format.



```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS Ecom_data (
    > 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 DELIMITED FIELDS TERMINATED BY ','
    > LINES TERMINATED BY '\n'
    > STORED AS TEXTFILE;
OK
Time taken: 0.116 seconds
hive> DESCRIBE Ecom_data;
OK
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
Time taken: 0.046 seconds, Fetched: 9 row(s)
```

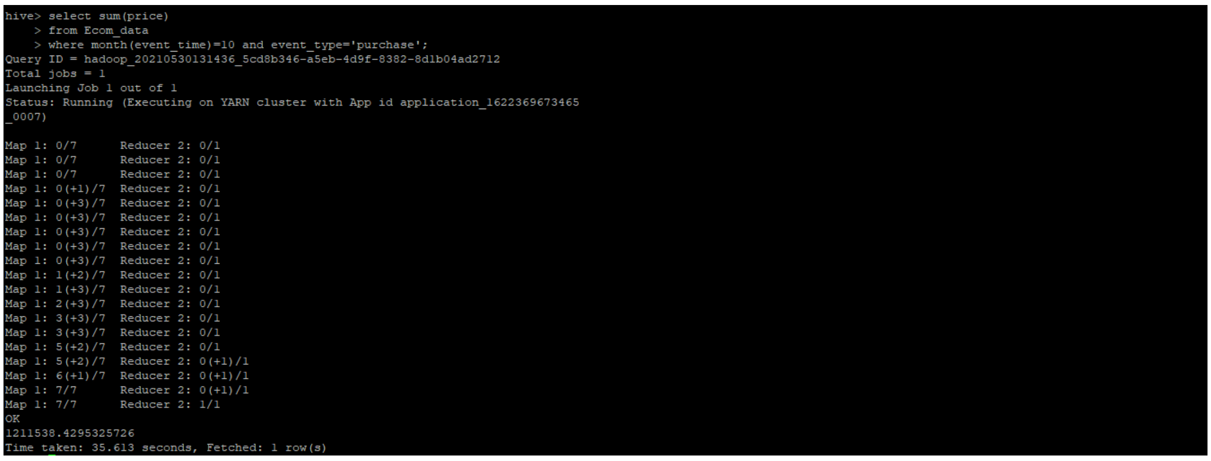
- Inserting data into this table.



```
hive> INSERT INTO Ecom_data
    > select
    > cast(replace(event_time,'UTC','') as timestamp),
    > event_type,
    > product_id,
    > category_id,
    > category_code,
    > brand,
    > cast(price as float),
    > cast(user_id as bigint),
    > user_session
    > from Ecom;
Query ID = hadoop_20210529191313_5a9a8902-41bb-4b47-854f-ae95f1de7f2b
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_1622306138328_0004)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      2          2        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 01/01  [==========================>>] 100%  ELAPSED TIME: 117.35 s
----------------------------------------------------------------------------------------------
Loading data to table hive_casestudy.ecom_data
OK
Time taken: 127.611 seconds
```

Saikat Chowdhury
Daivanshu Gandhi

Q: Find the total revenue generated due to purchases made in October. (using the Ecom_data table)

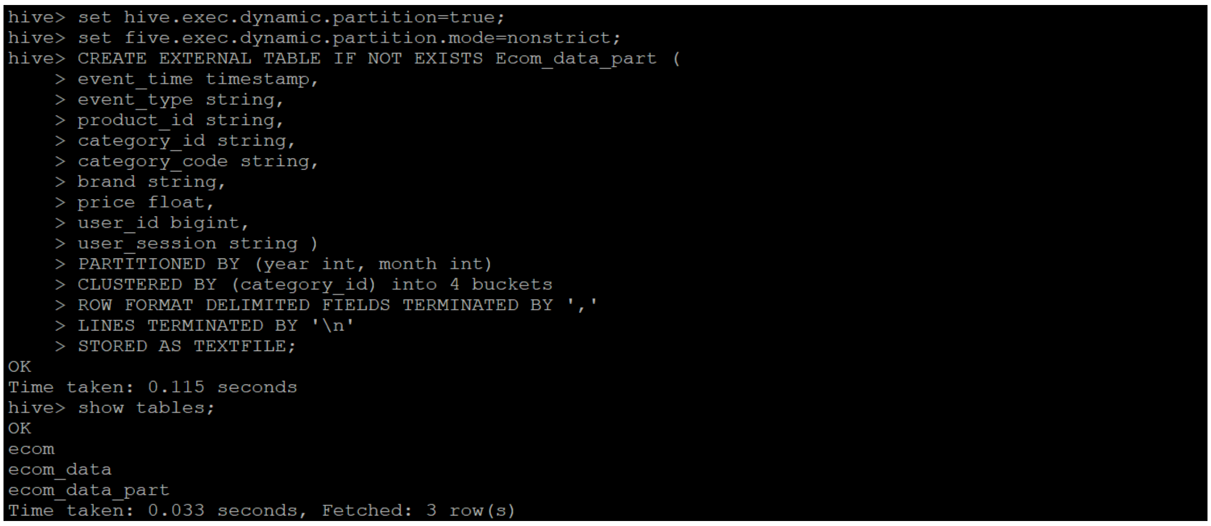


```
hive> select sum(price)
    > from Ecom_data
    > where month(event_time)=10 and event_type='purchase';
Query ID = hadoop_20210530131436_5cd8b346-a5eb-4d9f-8382-8d1b04ad2712
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1622369673465
_0007)

Map 1: 0/7      Reducer 2: 0/1
Map 1: 0/7      Reducer 2: 0/1
Map 1: 0/7      Reducer 2: 0/1
Map 1: 0(+1)/7  Reducer 2: 0/1
Map 1: 0(+3)/7  Reducer 2: 0/1
Map 1: 0(+3)/7  Reducer 2: 0/1
Map 1: 0(+3)/7  Reducer 2: 0/1
Map 1: 0(+3)/7  Reducer 2: 0/1
Map 1: 0(+3)/7  Reducer 2: 0/1
Map 1: 1(+2)/7  Reducer 2: 0/1
Map 1: 1(+3)/7  Reducer 2: 0/1
Map 1: 2(+3)/7  Reducer 2: 0/1
Map 1: 3(+3)/7  Reducer 2: 0/1
Map 1: 3(+3)/7  Reducer 2: 0/1
Map 1: 5(+2)/7  Reducer 2: 0/1
Map 1: 5(+2)/7  Reducer 2: 0(+1)/1
Map 1: 6(+1)/7  Reducer 2: 0(+1)/1
Map 1: 7/7      Reducer 2: 0(+1)/1
Map 1: 7/7      Reducer 2: 1/1
OK
1211538.4295325726
Time taken: 35.613 seconds, Fetched: 1 row(s)
```

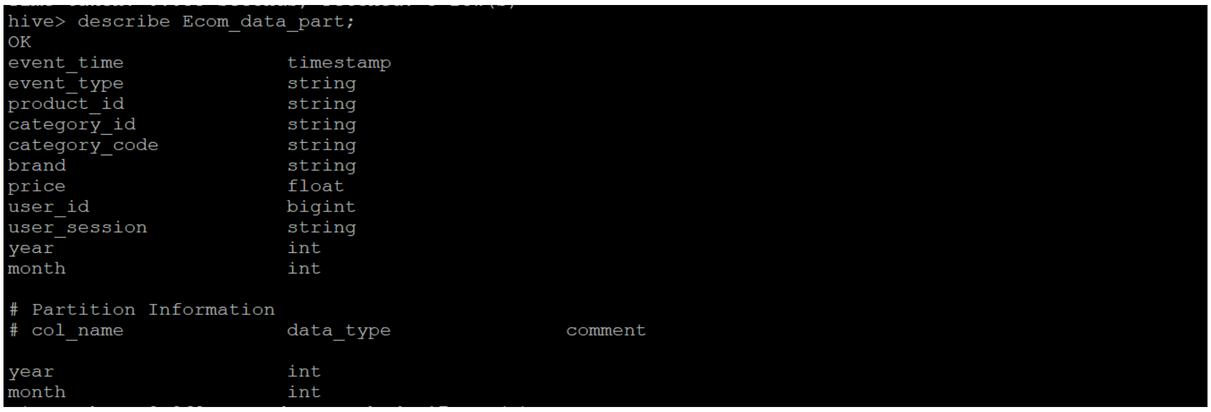
```
SELECT SUM(price)

FROM ecomm_data

WHERE MONTH(event_time)=10 AND event_type='purchase';
```

Time taken= 35 seconds.

Once the base table is created, we need to optimize the table for quick query result through partitioning and bucketing. Our optimized table name is Ecom_data_part.
Now we will Enabling Dynamic Partitioning and creating a partitioned table with buckets.

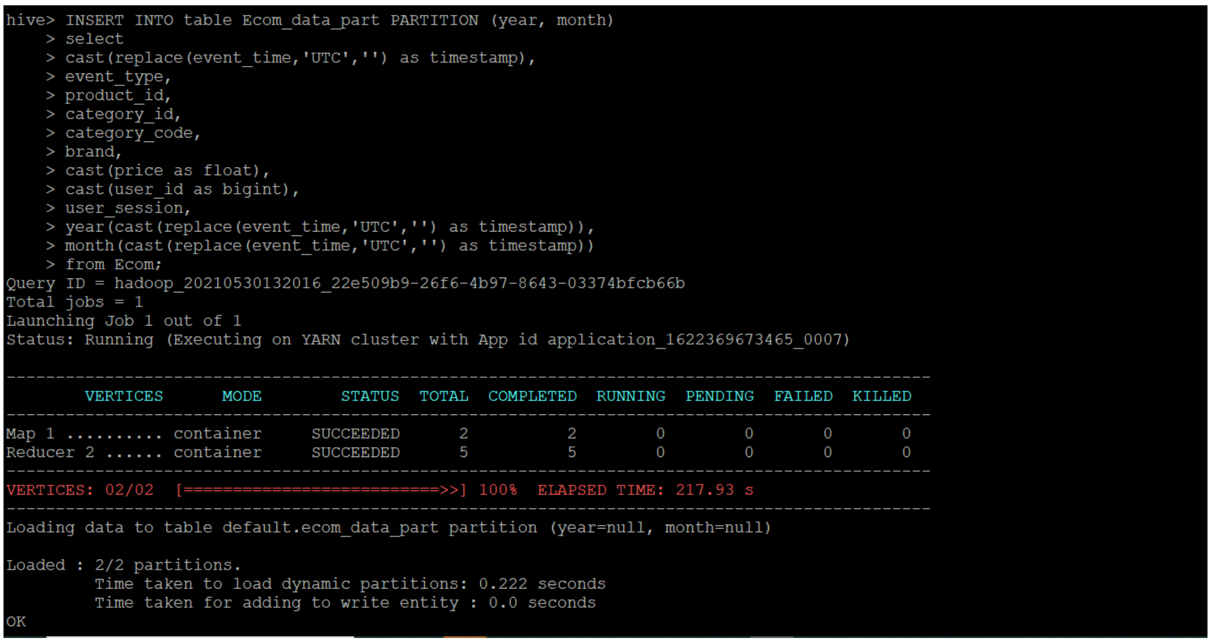


```
hive> set hive.exec.dynamic.partition=true;
hive> set five.exec.dynamic.partition.mode=nonstrict;
hive> CREATE EXTERNAL TABLE IF NOT EXISTS Ecom_data_part (
    > event_time timestamp,
    > event_type string,
    > product_id string,
    > category_id string,
    > category_code string,
    > brand string,
    > price float,
    > user_id bigint,
    > user_session string )
    > PARTITIONED BY (year int, month int)
    > CLUSTERED BY (category_id) into 4 buckets
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > LINES TERMINATED BY '\n'
    > STORED AS TEXTFILE;
OK
Time taken: 0.115 seconds
hive> show tables;
OK
ecom
ecom_data
ecom_data_part
Time taken: 0.033 seconds, Fetched: 3 row(s)
```



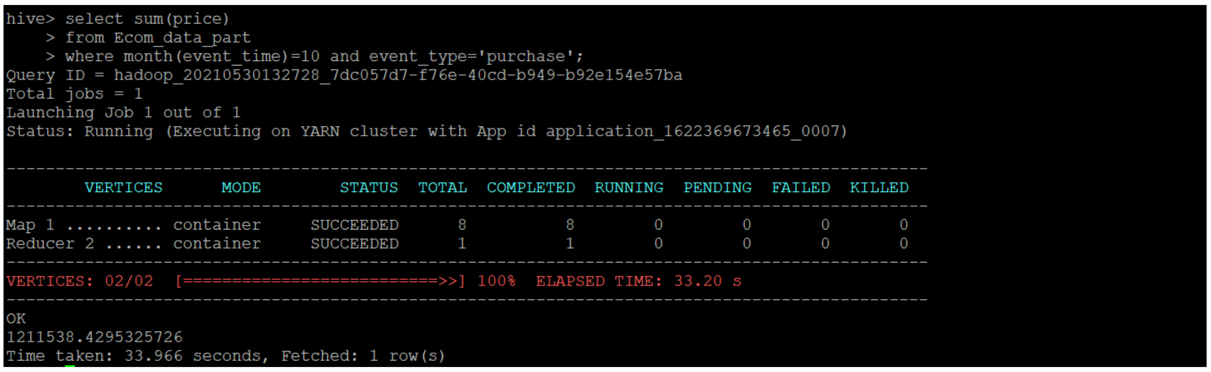
```
hive> describe Ecom_data_part;
OK
event_time              timestamp
event_type              string
product_id              string
category_id             string
category_code           string
brand                   string
price                   float
user_id                 bigint
user_session            string
year                    int
month                   int

# Partition Information
# col_name              data_type               comment

year                    int
month                   int
```

The new optimised table has been created. We will now insert the data into this table.



```
hive> INSERT INTO table Ecom_data_part PARTITION (year, month)
    > select
    > cast(replace(event_time,'UTC','') as timestamp),
    > event_type,
    > product_id,
    > category_id,
    > category_code,
    > brand,
    > cast(price as float),
    > cast(user_id as bigint),
    > user_session,
    > year(cast(replace(event_time,'UTC','') as timestamp)),
    > month(cast(replace(event_time,'UTC','') as timestamp))
    > from Ecom;
Query ID = hadoop_20210530132016_22e509b9-26f6-4b97-8643-03374bfcb66b
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1622369673465_0007)

----------------------------------------------------------------------------------------------
        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: 217.93 s
----------------------------------------------------------------------------------------------
Loading data to table default.ecom_data_part partition (year=null, month=null)

Loaded : 2/2 partitions.
         Time taken to load dynamic partitions: 0.222 seconds
         Time taken for adding to write entity : 0.0 seconds
OK
```

We will now test this table by running the same query in this optimised table and note the time taken.



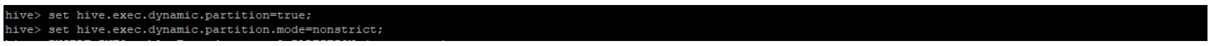
```
hive> select sum(price)
    > from Ecom_data_part
    > where month(event_time)=10 and event_type='purchase';
Query ID = hadoop_20210530132728_7dc057d7-f76e-40cd-b949-b92e154e57ba
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1622369673465_0007)

----------------------------------------------------------------------------------------------
        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: 33.20 s
----------------------------------------------------------------------------------------------
OK
1211538.4295325726
Time taken: 33.966 seconds, Fetched: 1 row(s)
```

```
SELECT SUM(price) as total_revenue

FROM ecomm_data_part
```

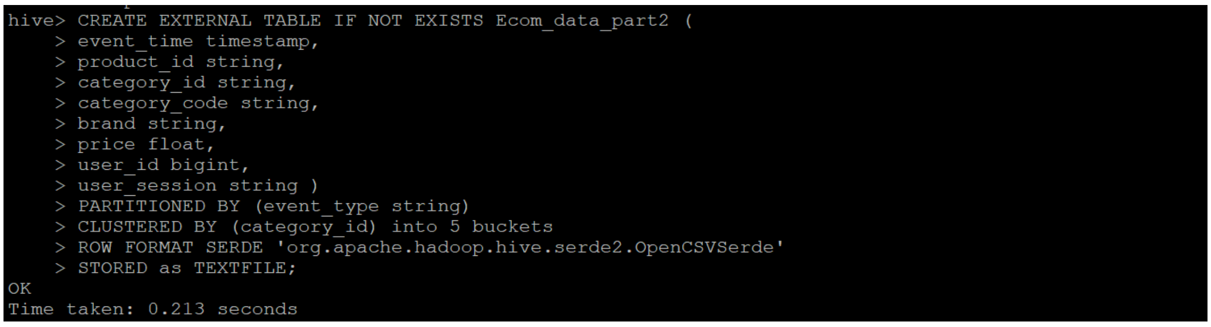
`WHERE MONTH(event_time)=10 AND event_type='purchase';`

Time taken= 33.96 seconds.

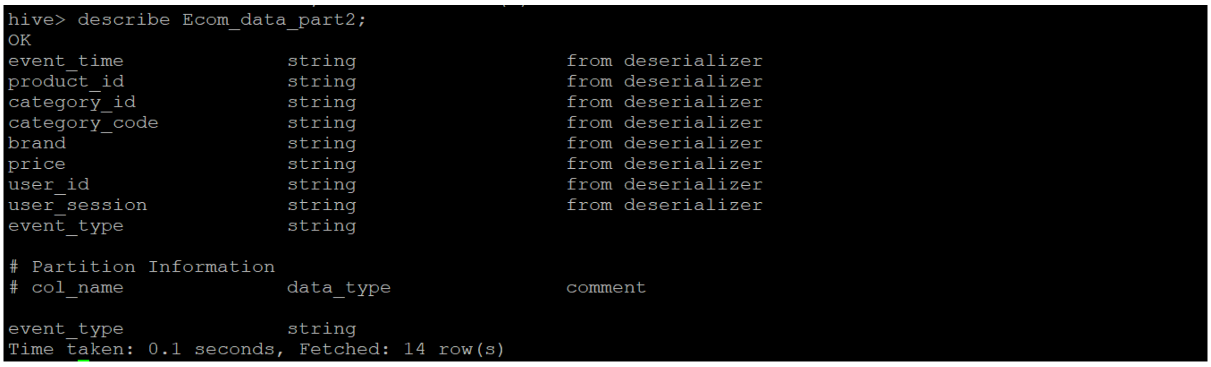
Enabling Second Approach for Dynamic Partitioning and creating a partitioned table with buckets.



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

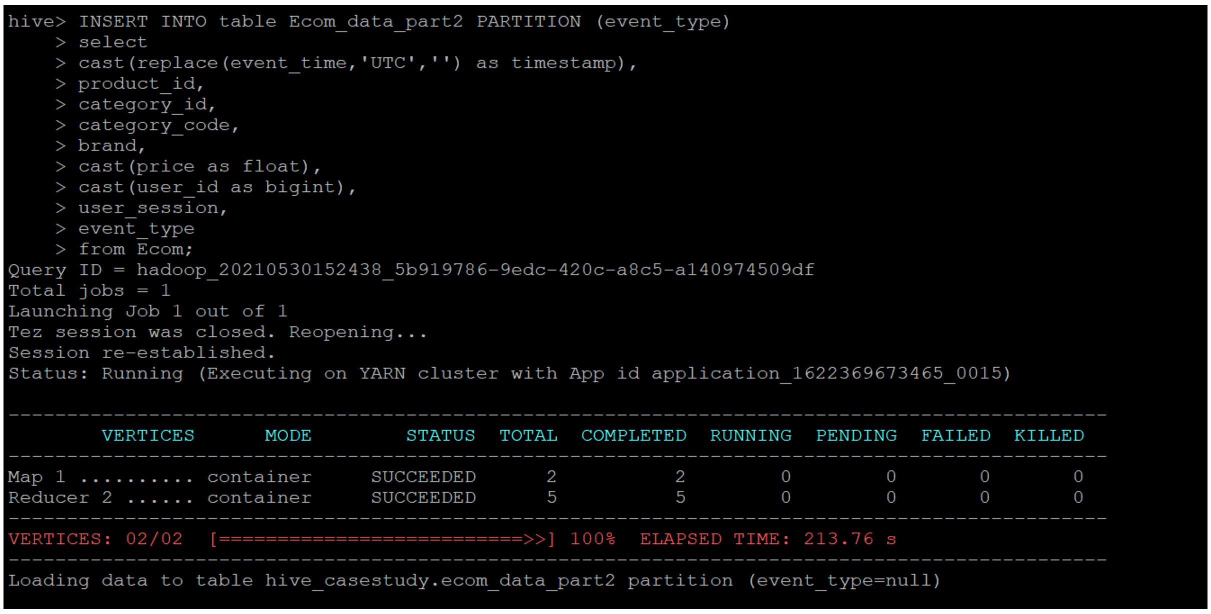


```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS Ecom_data_part2 (
    > event_time timestamp,
    > product_id string,
    > category_id string,
    > category_code string,
    > brand string,
    > price float,
    > user_id bigint,
    > user_session string )
    > PARTITIONED BY (event_type string)
    > CLUSTERED BY (category_id) into 5 buckets
    > ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
    > STORED as TEXTFILE;
OK
Time taken: 0.213 seconds
```

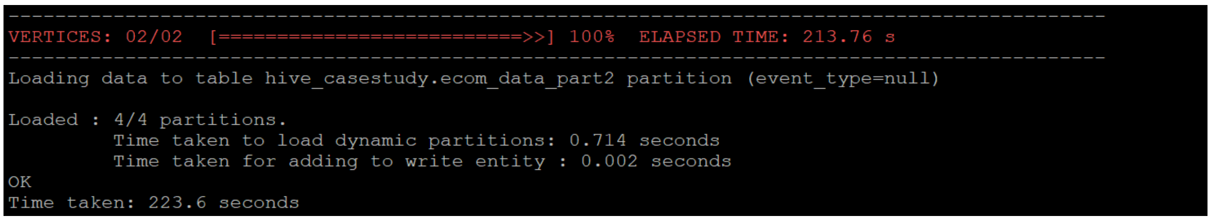


```
hive> describe Ecom_data_part2;
OK
event_time              string                  from deserializer
product_id              string                  from deserializer
category_id             string                  from deserializer
category_code           string                  from deserializer
brand                   string                  from deserializer
price                   string                  from deserializer
user_id                 string                  from deserializer
user_session            string                  from deserializer
event_type              string

# Partition Information
# col_name              data_type               comment

event_type              string
Time taken: 0.1 seconds, Fetched: 14 row(s)
```

The new optimised table has been created. We will now insert the data into this table.

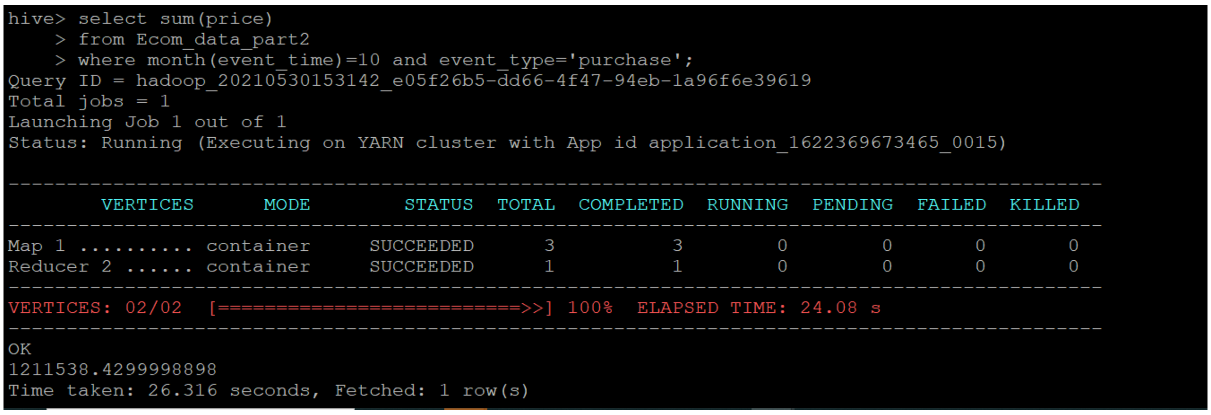


```
hive> INSERT INTO table Ecom_data_part2 PARTITION (event_type)
    > select
    > cast(replace(event_time,'UTC','') as timestamp),
    > product_id,
    > category_id,
    > category_code,
    > brand,
    > cast(price as float),
    > cast(user_id as bigint),
    > user_session,
    > event_type
    > from Ecom;
Query ID = hadoop_20210530152438_5b919786-9edc-420c-a8c5-a140974509df
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_1622369673465_0015)

----------------------------------------------------------------------------------------------
        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: 213.76 s
----------------------------------------------------------------------------------------------
Loading data to table hive_casestudy.ecom_data_part2 partition (event_type=null)
```



```
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 213.76 s
----------------------------------------------------------------------------------------------
Loading data to table hive_casestudy.ecom_data_part2 partition (event_type=null)

Loaded : 4/4 partitions.
         Time taken to load dynamic partitions: 0.714 seconds
         Time taken for adding to write entity : 0.002 seconds
OK
Time taken: 223.6 seconds
```

We will now test this table by running the same query in this optimised table and note the time taken.



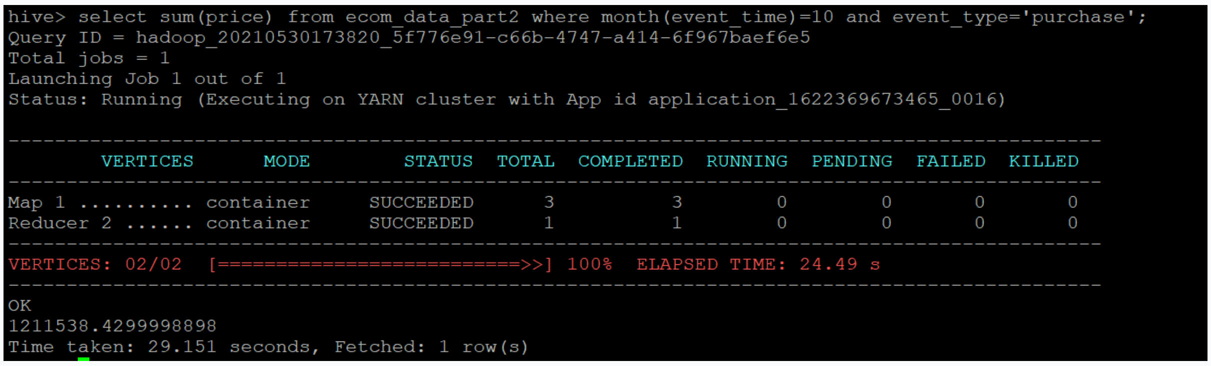
```
hive> select sum(price)
    > from Ecom_data_part2
    > where month(event_time)=10 and event_type='purchase';
Query ID = hadoop_20210530153142_e05f26b5-dd66-4f47-94eb-1a96f6e39619
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1622369673465_0015)

----------------------------------------------------------------------------------------------
        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: 24.08 s
----------------------------------------------------------------------------------------------
OK
1211538.4299998898
Time taken: 26.316 seconds, Fetched: 1 row(s)
```

```
SELECT SUM(price) as total_revenue

FROM ecomm_data_part2

WHERE MONTH(event_time)=10 AND event_type='purchase';
```

Time taken= 26.316 seconds.

- So here we find that By **Partition by** over 'event_type' and **clustering by** 'category_id' we get the most optimized output of the query.

Q1. Find the total revenue generated due to purchases made in October.

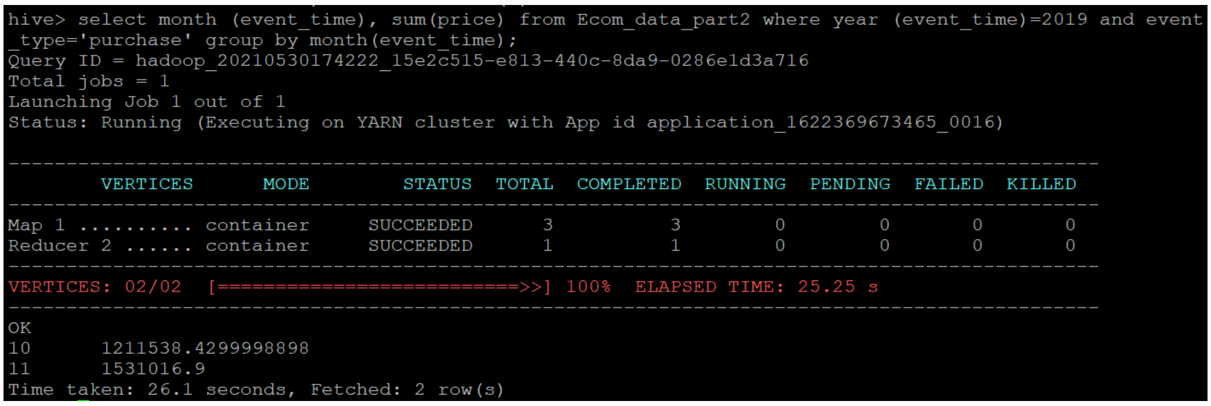


```
hive> select sum(price) from ecom_data_part2 where month(event_time)=10 and event_type='purchase';
Query ID = hadoop_20210530173820_5f776e91-c66b-4747-a414-6f967baef6e5
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1622369673465_0016)

----------------------------------------------------------------------------------------------
        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: 24.49 s
----------------------------------------------------------------------------------------------
OK
1211538.4299998898
Time taken: 29.151 seconds, Fetched: 1 row(s)
```

Query: select sum(price) from ecom_data_part where month(event_time)=10 and event_type='purchase';

Note :- The screenshot 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

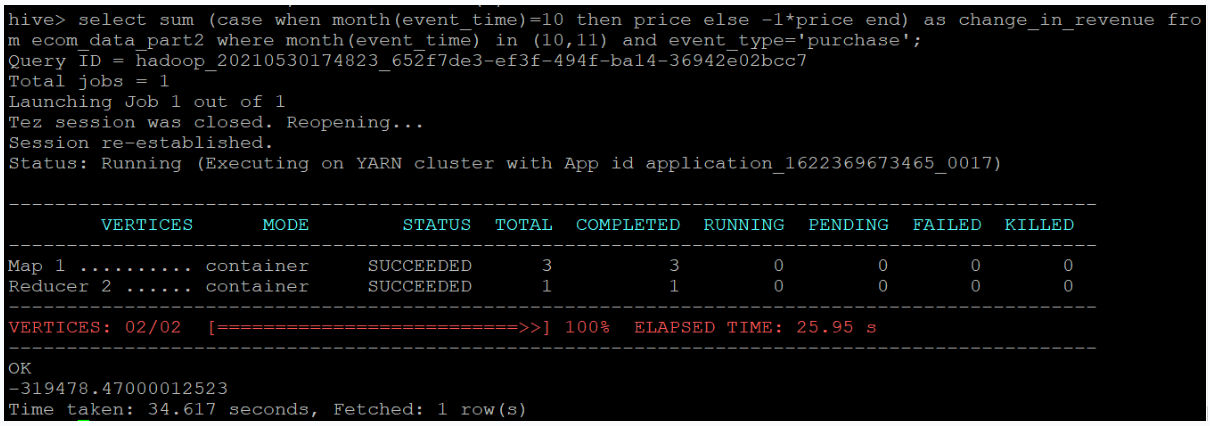
Q2. Write a query to yield the total sum of purchases per month in a single output.



```
hive> select month (event_time), sum(price) from Ecom_data_part2 where year (event_time)=2019 and event
_type='purchase' group by month(event_time);
Query ID = hadoop_20210530174222_15e2c515-e813-440c-8da9-0286e1d3a716
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1622369673465_0016)

----------------------------------------------------------------------------------------------
        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: 25.25 s
----------------------------------------------------------------------------------------------
OK
10      1211538.4299998898
11      1531016.9
Time taken: 26.1 seconds, Fetched: 2 row(s)
```

Query: select month (event_time), sum(price) from Ecom_data_part2 where year (event_time)=2019 and event_type='purchase' group by month(event_time);

Q3. Write a query to find the change in revenue generated due to purchases from October to November.

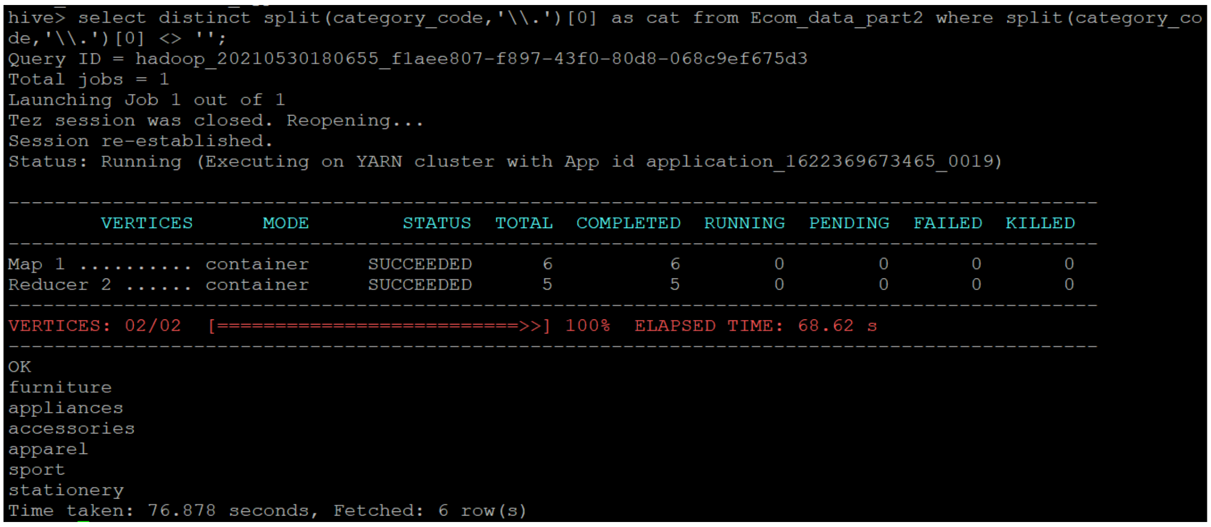


```
hive> select sum (case when month(event_time)=10 then price else -1*price end) as change_in_revenue fro
m ecom_data_part2 where month(event_time) in (10,11) and event_type='purchase';
Query ID = hadoop_20210530174823_652f7de3-ef3f-494f-ba14-36942e02bcc7
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1622369673465_0017)

----------------------------------------------------------------------------------------------
        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: 25.95 s
----------------------------------------------------------------------------------------------
OK
-319478.47000012523
Time taken: 34.617 seconds, Fetched: 1 row(s)
```

Query: select sum (case when month(event_time)=10 then price else -1*price end) as change_in_revenue from ecom_data_part2 where month(event_time) in (10,11) and event_type='purchase';

Q4. Find distinct categories of products. Categories with null category code can be ignored.

Result: Total categories: 500 categories

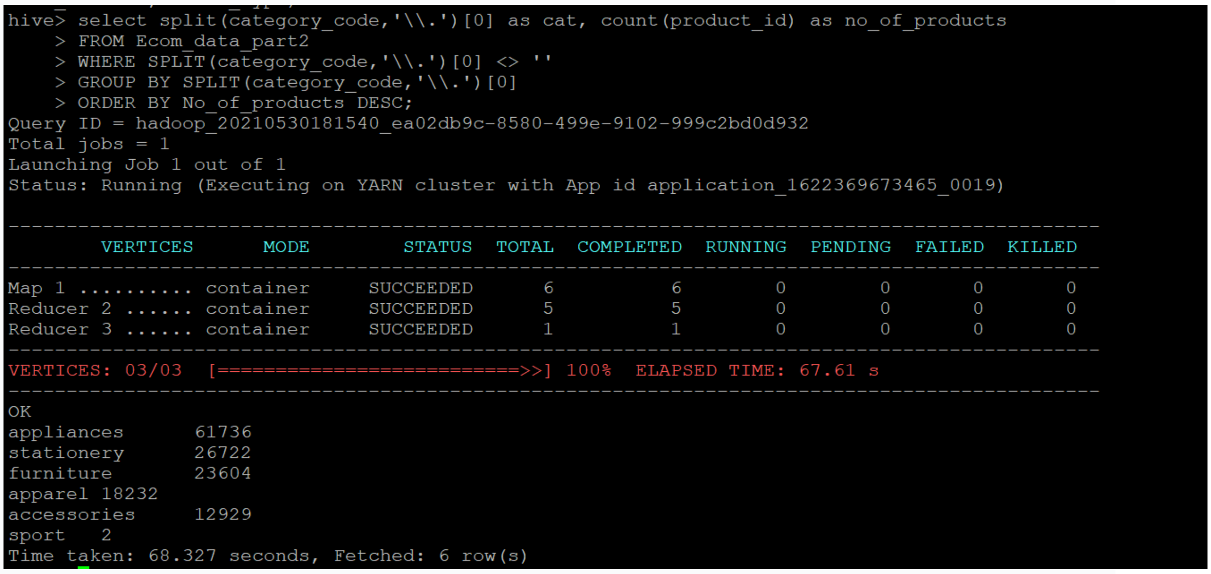


```
hive> select distinct split(category_code,'\\.')[0] as cat from Ecom_data_part2 where split(category_co
de,'\\.')[0] <> '';
Query ID = hadoop_20210530180655_f1aee807-f897-43f0-80d8-068c9ef675d3
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1622369673465_0019)

----------------------------------------------------------------------------------------------
        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: 68.62 s
----------------------------------------------------------------------------------------------
OK
furniture
appliances
accessories
apparel
sport
stationery
Time taken: 76.878 seconds, Fetched: 6 row(s)
```

Query: select distinct split(category_code,’\\.’)[0] as cat from Ecom_data_part2 where split(category_code,’\\.’)[0] <> ‘’;

Q5. Find the total number of products available under each category.

ans: 6 products available

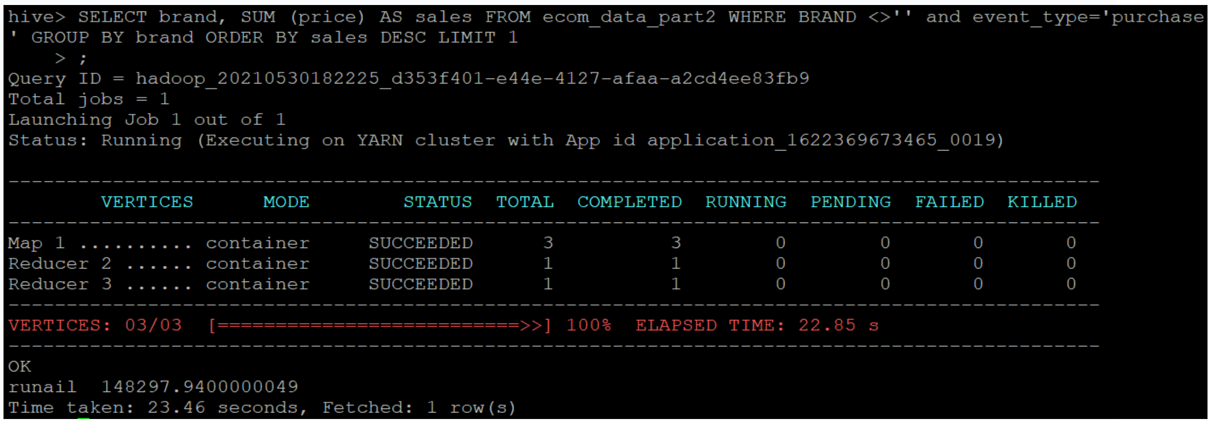


```
hive> select split(category_code,'\\.')[0] as cat, count(product_id) as no_of_products
    > FROM Ecom_data_part2
    > WHERE SPLIT(category_code,'\\.')[0] <> ''
    > GROUP BY SPLIT(category_code,'\\.')[0]
    > ORDER BY No_of_products DESC;
Query ID = hadoop_20210530181540_ea02db9c-8580-499e-9102-999c2bd0d932
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1622369673465_0019)

----------------------------------------------------------------------------------------------
        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: 67.61 s
----------------------------------------------------------------------------------------------
OK
appliances      61736
stationery      26722
furniture       23604
apparel 18232
accessories     12929
sport   2
Time taken: 68.327 seconds, Fetched: 6 row(s)
```

Query: SELECT SPLIT(category_code,'\\.')[0] AS cat, COUNT(product_id) AS No_of_products FROM Ecom_data_part2 WHERE SPLIT(category_code,'\\.')[0] <> '' GROUP BY SPLIT(category_code,'\\.')[0] ORDER BY No_of_products DESC;

Saikat Chowdhury
Daivanshu Gandhi

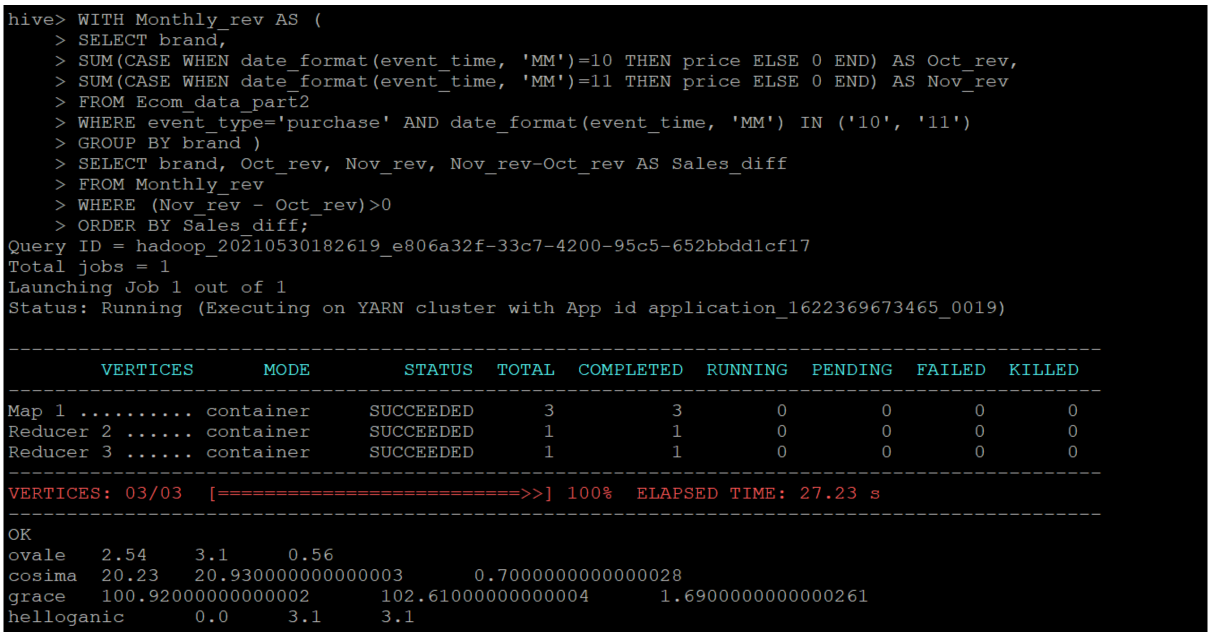
Q6. Which brand had the maximum sales in October and November combined?



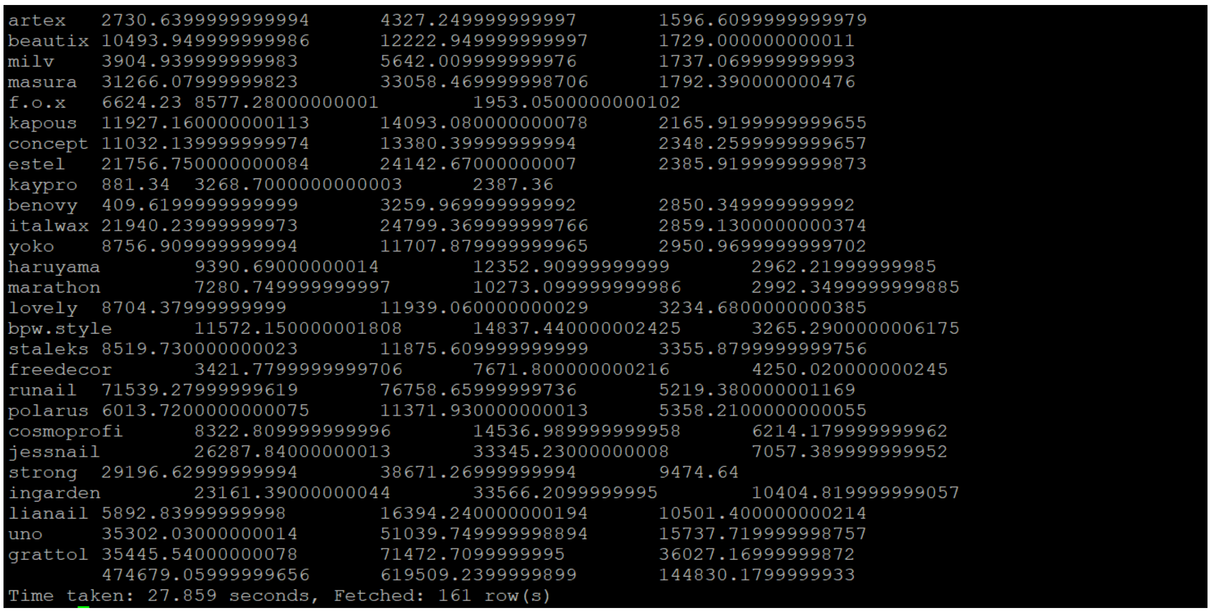
```
hive> SELECT brand, SUM (price) AS sales FROM ecom_data_part2 WHERE BRAND <>'' and event_type='purchase
' GROUP BY brand ORDER BY sales DESC LIMIT 1
    > ;
Query ID = hadoop_20210530182225_d353f401-e44e-4127-afaa-a2cd4ee83fb9
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1622369673465_0019)

----------------------------------------------------------------------------------------------
        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: 22.85 s
----------------------------------------------------------------------------------------------
OK
runail  148297.9400000049
Time taken: 23.46 seconds, Fetched: 1 row(s)
```

Query: SELECT brand, SUM (price) AS sales FROM ecom_data_part2 WHERE BRAND <>'' and event_type='purchase' GROUP BY brand ORDER BY sales DESC LIMIT 1;

Q7. Which brands increased their sales from October to November?



```
hive> WITH Monthly_rev AS (
    > SELECT brand,
    > SUM(CASE WHEN date_format(event_time, 'MM')=10 THEN price ELSE 0 END) AS Oct_rev,
    > SUM(CASE WHEN date_format(event_time, 'MM')=11 THEN price ELSE 0 END) AS Nov_rev
    > FROM Ecom_data_part2
    > WHERE event_type='purchase' AND date_format(event_time, 'MM') IN ('10', '11')
    > GROUP BY brand )
    > SELECT brand, Oct_rev, Nov_rev, Nov_rev-Oct_rev AS Sales_diff
    > FROM Monthly_rev
    > WHERE (Nov_rev - Oct_rev)>0
    > ORDER BY Sales_diff;
Query ID = hadoop_20210530182619_e806a32f-33c7-4200-95c5-652bbdd1cf17
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1622369673465_0019)

----------------------------------------------------------------------------------------------
        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: 27.23 s
----------------------------------------------------------------------------------------------
OK
ovale   2.54    3.1     0.56
cosima  20.23   20.930000000000003      0.7000000000000028
grace   100.92000000000002      102.61000000000004      1.6900000000000261
hellogenic      0.0     3.1     3.1
```

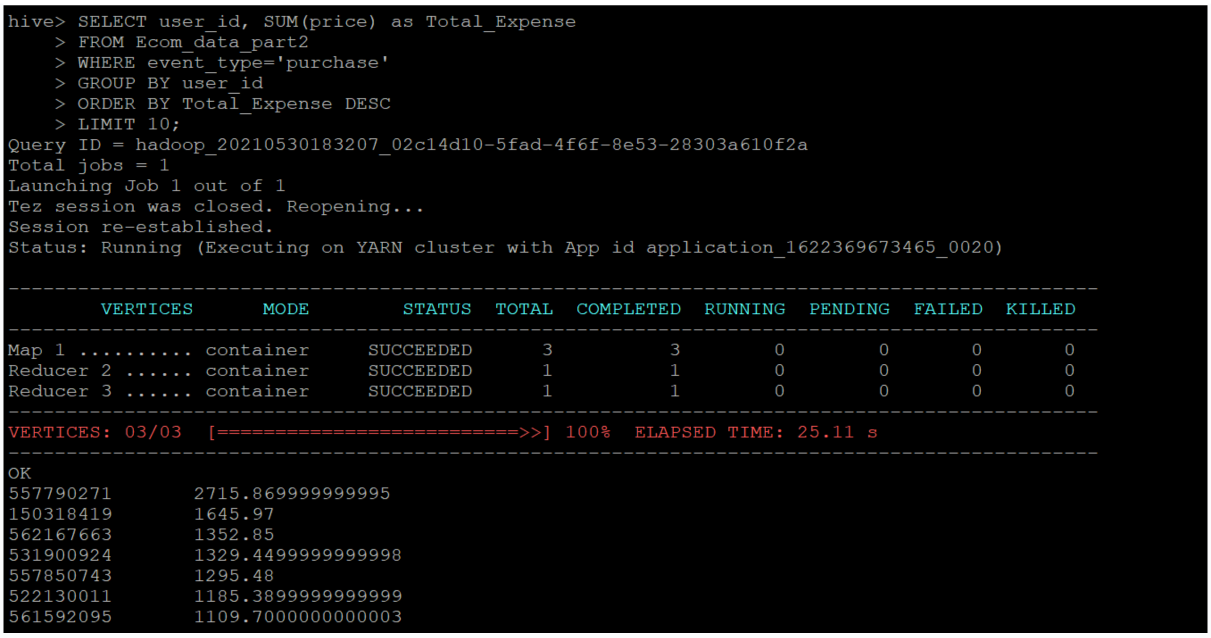


```
artex   2730.6399999999994      4327.249999999997       1596.6099999999979
beautix 10493.949999999986      12222.949999999997      1729.000000000011
milv    3904.939999999983       5642.009999999976       1737.069999999993
masura  31266.07999999823       33058.469999998706      1792.390000000476
f.o.x   6624.23 8577.28000000001        1953.0500000000102
kapous  11927.160000000113      14093.080000000078      2165.9199999999655
concept 11032.139999999974      13380.39999999994       2348.2599999999657
estel   21756.750000000084      24142.67000000007       2385.9199999999873
kaypro  881.34  3268.7000000000003      2387.36
benovy  409.6199999999999       3259.969999999992       2850.349999999992
italwax 21940.23999999973       24799.369999999766      2859.1300000000374
yoko    8756.909999999994       11707.879999999965      2950.9699999999702
haruyama        9390.69000000014        12352.90999999999       2962.21999999985
marathon        7280.749999999997       10273.099999999986      2992.3499999999885
lovely  8704.37999999999        11939.060000000029      3234.6800000000385
bpw.style       11572.150000001808      14837.440000002425      3265.2900000006175
staleks 8519.730000000023       11875.609999999999      3355.8799999999756
freedecor       3421.7799999999706      7671.800000000216       4250.020000000245
runail  71539.27999999619       76758.65999999736       5219.380000001169
polarus 6013.7200000000075      11371.930000000013      5358.2100000000055
cosmoprofi      8322.809999999996       14536.989999999958      6214.179999999962
jessnail        26287.84000000013       33345.23000000008       7057.389999999952
strong  29196.62999999994       38671.26999999994       9474.64
ingarden        23161.39000000044       33566.2099999995        10404.819999999057
lianail 5892.83999999998        16394.240000000194      10501.400000000214
uno     35302.03000000014       51039.749999998894      15737.719999998757
grattol 35445.54000000078       71472.7099999995        36027.1699999872
        474679.05999999656      619509.2399999899       144830.1799999933
Time taken: 27.859 seconds, Fetched: 161 row(s)
```

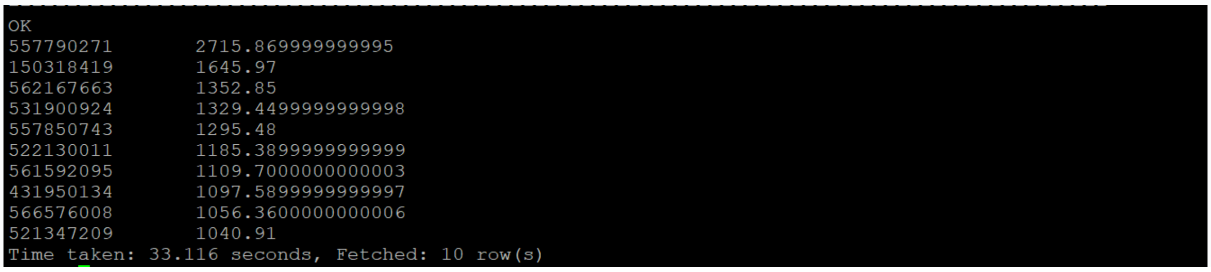
Query: WITH Monthly_rev AS (
SELECT brand,
SUM(CASE WHEN date_format(event_time, 'MM')=10 THEN price ELSE 0 END) AS Oct_rev,
SUM(CASE WHEN date_format(event_time, 'MM')=11 THEN price ELSE 0 END) AS Nov_rev
FROM Ecom_data_part2
WHERE event_type='purchase' AND date_format(event_time, 'MM') IN ('10', '11')
GROUP BY brand)
SELECT brand, Oct_rev, Nov_rev, Nov_rev-Oct_rev AS Sales_diff
FROM Monthly_rev
WHERE (Nov_rev - Oct_rev)>0
ORDER BY Sales_diff;

So from the above data we can see that there are total of 161 brands increasing from October to November.

Q8. Your company wants to reward the top 10 users of its website with a Golden Customer plan. Write a query to generate a list of top 10 users who spend the most.



```
hive> SELECT user_id, SUM(price) as Total_Expense
    > FROM Ecom_data_part2
    > WHERE event_type='purchase'
    > GROUP BY user_id
    > ORDER BY Total_Expense DESC
    > LIMIT 10;
Query ID = hadoop_20210530183207_02c14d10-5fad-4f6f-8e53-28303a610f2a
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_1622369673465_0020)

----------------------------------------------------------------------------------------------
        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: 25.11 s
----------------------------------------------------------------------------------------------
OK
557790271       2715.869999999995
150318419       1645.97
562167663       1352.85
531900924       1329.4499999999998
557850743       1295.48
522130011       1185.3899999999999
561592095       1109.7000000000003
```



```
OK
557790271       2715.869999999995
150318419       1645.97
562167663       1352.85
531900924       1329.4499999999998
557850743       1295.48
522130011       1185.3899999999999
561592095       1109.7000000000003
431950134       1097.5899999999997
566576008       1056.3600000000006
521347209       1040.91
Time taken: 33.116 seconds, Fetched: 10 row(s)
```

Query: SELECT user_id, SUM(price) as Total_Expense

FROM Ecom_data_part2

WHERE event_type='purchase'

GROUP BY user_id

ORDER BY Total_Expense DESC

LIMIT 10;