Hive Case Study

E-Commerce Sales Review

By,
Antara Chatterji
Satvik Yadav

PROBLEM STATEMENT :

Tech companies are trying to analyse the customer behaviour and gain insights about the product trends. This helps the companies to make the products easily available for the customers and thus increasing their sales. As a big data analyst, one needs to extract the data and gather insights from real time data of an e-commerce company.

Objective: We need to gather the insights from the clickstream data so we can extract insights from the customer behaviour.

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:

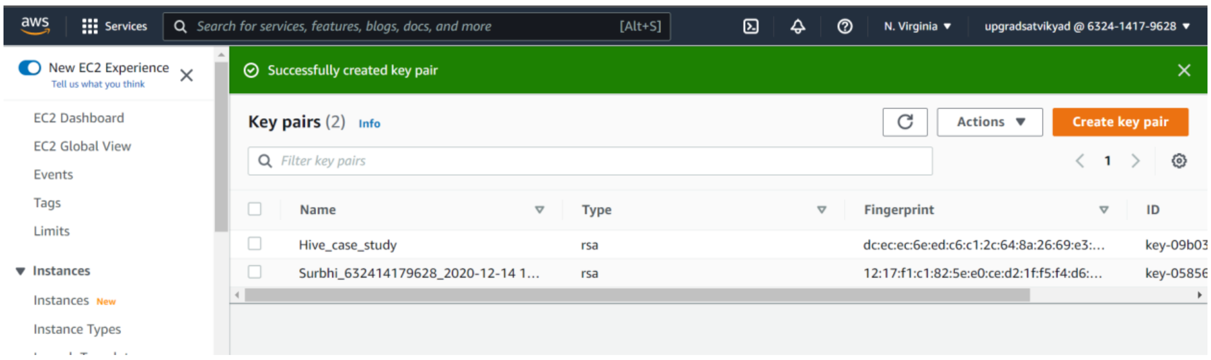
- Create the structure of your database,
- Use optimized techniques to run your queries as efficiently as possible
- Show the improvement of the performance after using optimization on any single query.
- Run Hive queries to answer the questions given below.

Cleaning up

- Drop your database, and
- Terminate your cluster

Launch the EMR cluster:

- To Launch the EMR cluster
- Create a key-pair and download the .PEM/.PPK file

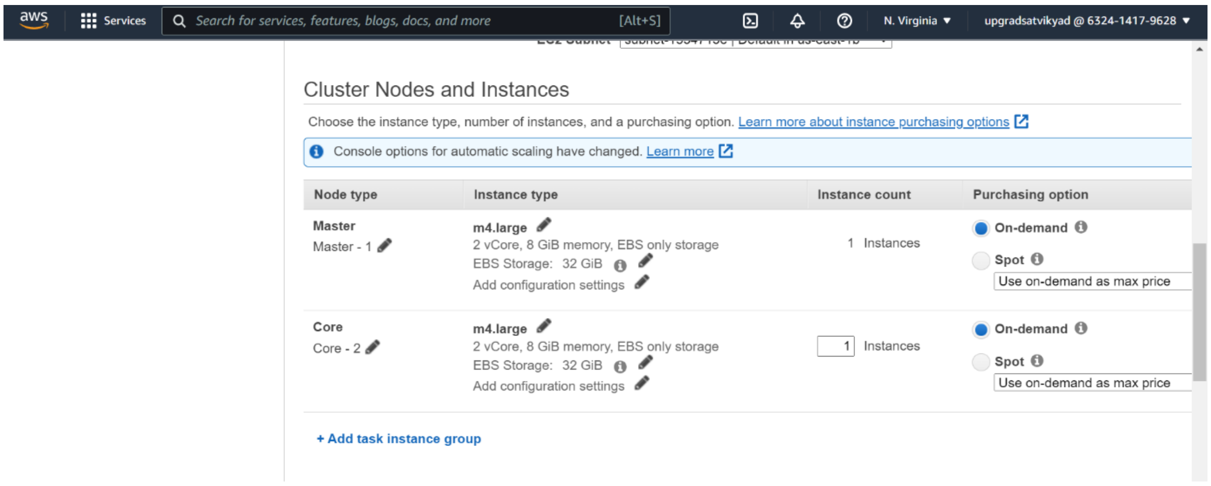


Successfully created key pair

Key pairs (2) Info

Name	Type	Fingerprint	ID
Hive_case_study	rsa	dc:ec:ec:6e:ed:c6:c1:2c:64:8a:26:69:e3:...	key-09b03
Surbhi_632414179628_2020-12-14 1...	rsa	12:17:f1:c1:82:5e:e0:ce:d2:1f:f5:f4:d6:...	key-05856

- Now we need to create EMR cluster. While creating EMR cluster we need to select **m4.large** for both Master and Core node of single instance.


Cluster Nodes and Instances

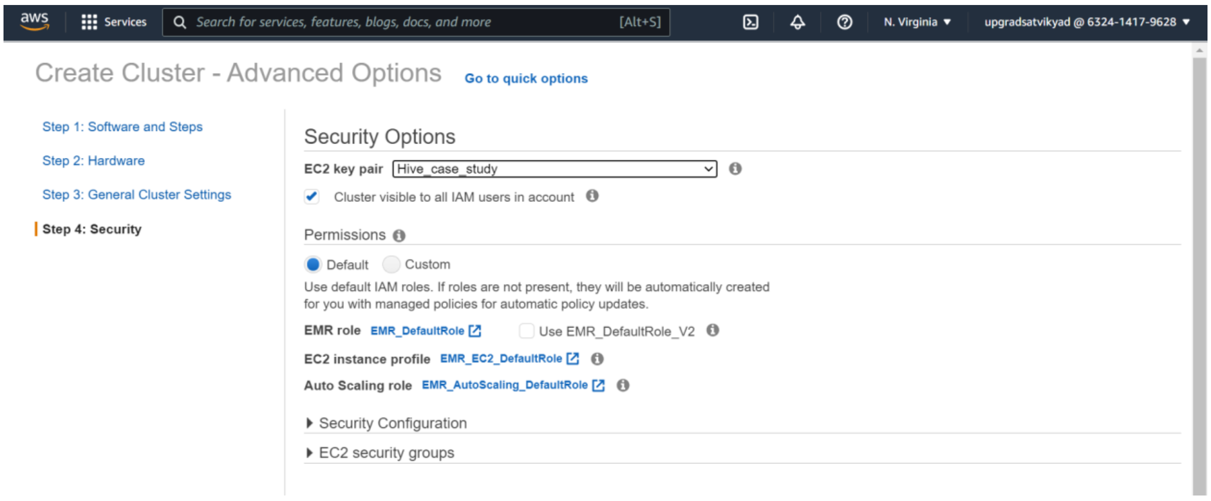
Choose the instance type, number of instances, and a purchasing option. Learn more about instance purchasing options

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

- Then we need to select the correct key pair from the dropdown.


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_case_study

✓ Cluster visible to all IAM users in account

Permissions

[✓ Default] Custom

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

EMR role: EMR_DefaultRole   ☐ Use EMR_DefaultRole_V2

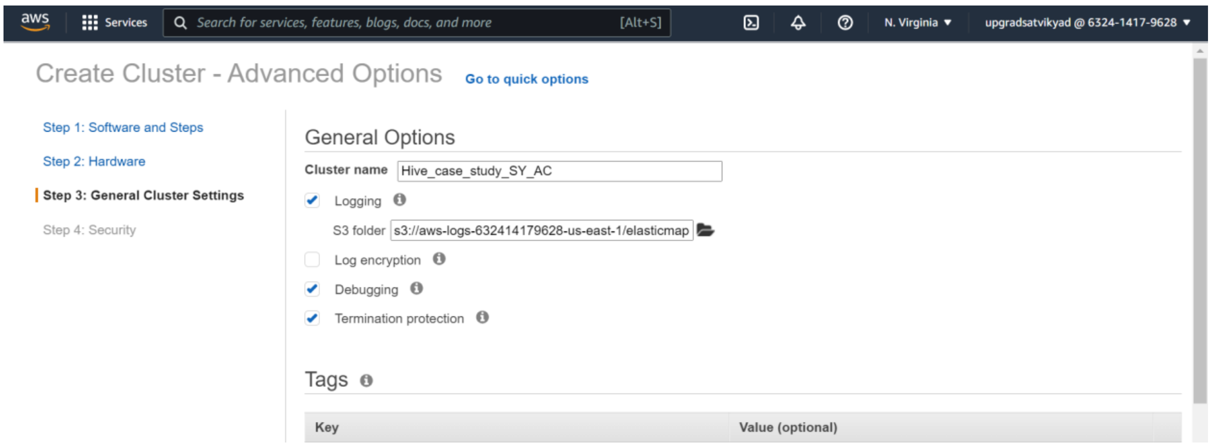
EC2 instance profile: EMR_EC2_DefaultRole

Auto Scaling role: EMR_AutoScaling_DefaultRole

Security Configuration

EC2 security groups

- We need to provide a relevant cluster name while creating the cluster.


Create Cluster - Advanced Options   Go to quick options

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

General Options

Cluster name: Hive_case_study_SY_AC

✓ Logging

S3 folder: s3://aws-logs-632414179628-us-east-1/elasticmap

☐ Log encryption

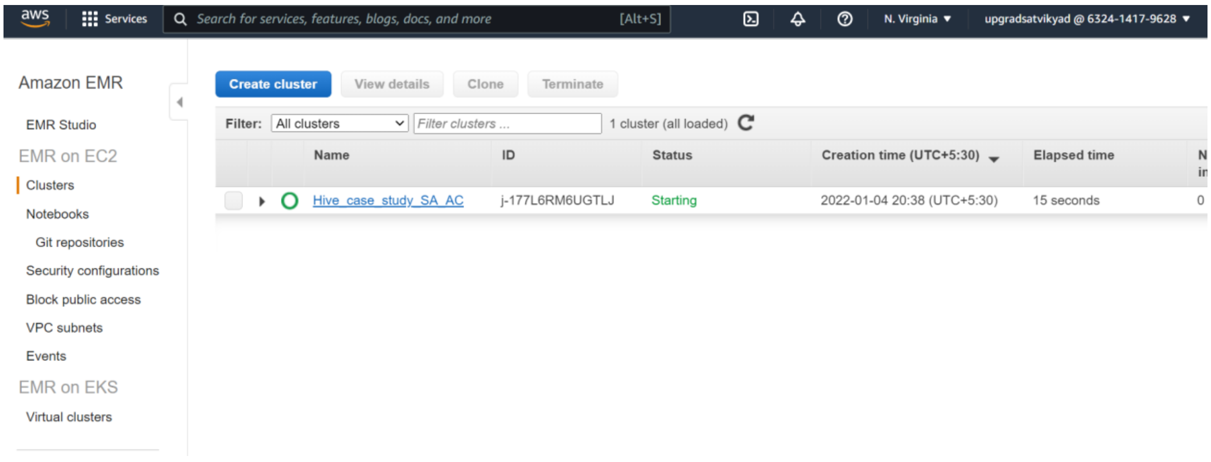
✓ Debugging

✓ Termination protection

Tags

Key	Value (optional)

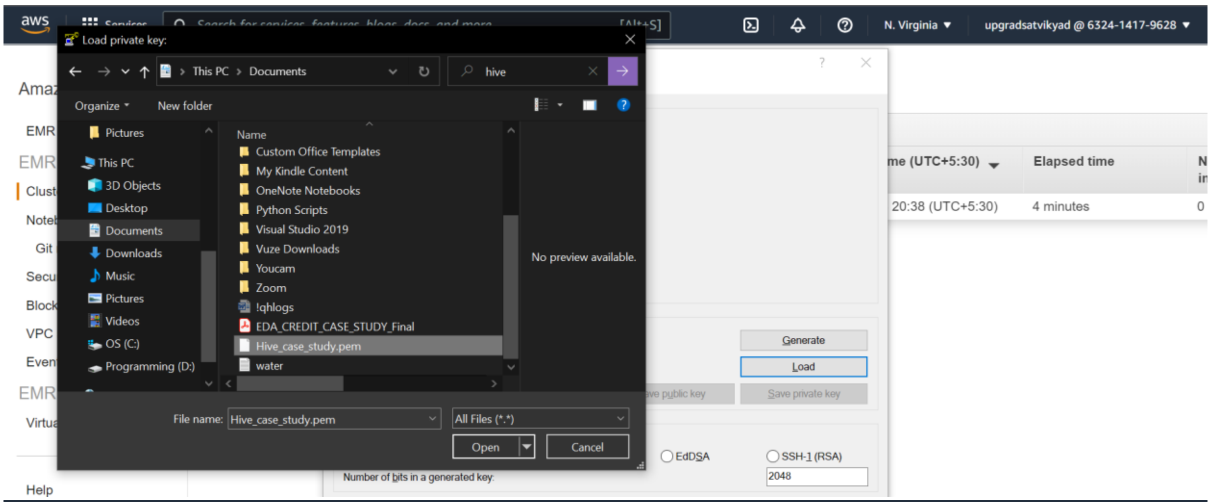
- Now the cluster is created and is in the waiting stage. Now we need to move data from S3 to HDFS.



	Name	ID	Status	Creation time (UTC+5:30)	Elapsed time
	Hive_case_study_SA_AC	j-177L6RM6UGTLJ	Starting	2022-01-04 20:38 (UTC+5:30)	15 seconds

Connect to EMR cluster:

- We need to open PuTTY for windows and open the downloaded .pem file and save the private key which is in .ppk extension.

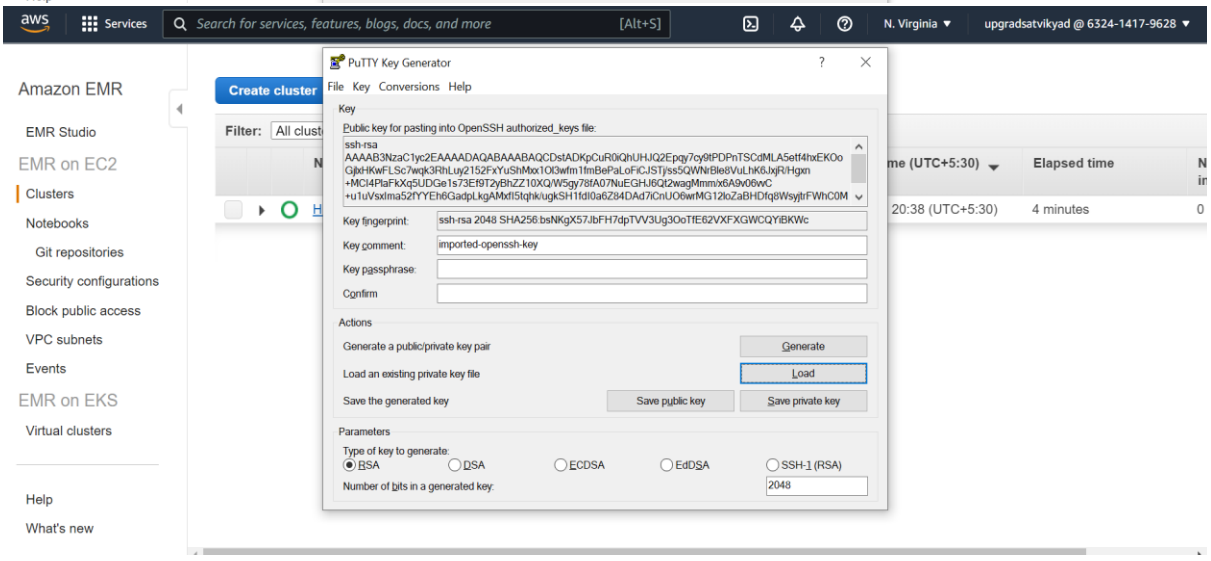


Load private key:

This PC > Documents

Custom Office Templates
My Kindle Content
OneNote Notebooks
Python Scripts
Visual Studio 2019
Vuze Downloads
Youcam
Zoom
!qhlogs
EDA_CREDIT_CASE_STUDY_Final
Hive_case_study.pem
water

File name: Hive_case_study.pem
All Files (*.*)



PuTTY Key Generator

Public key for pasting into OpenSSH authorized_keys file:

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCDstADKpCuR0iQhUHJQ2Epqy7cy9tPDPnTSCdMLA5etf4hxEKOo
GjlxHKwFLSc7wqk3RhLuy2152FxYuShMxx1Ol3wfm1fmBePaLoFiCJSTj/ss5QWNrBle8VuLhK6JxjR/Hgxn
+MCl4PlaFkXq5UDGe1s73Ef9T2yBhZZ10XQ/W5gy78fA07NuEGHJ6Qt2wagMmm/x6A9v06wC
+u1uVsxIma52fYYEh6GadpLkgAMxfI5tqhk/ugkSH1fdI0a6Z84DAd7iCnUO6wrMG12loZaBHDfq8WsyjtrFWhC0M
```

Key fingerprint: ssh-rsa 2048 SHA256:bsNKgX57JbFH7dpTVV3Ug3OoTfE62VXFXGWCQYiBKWc

Key comment: imported-openssh-key

Key passphrase:

Confirm:

Actions

Generate a public/private key pair — Generate

Load an existing private key file — Load

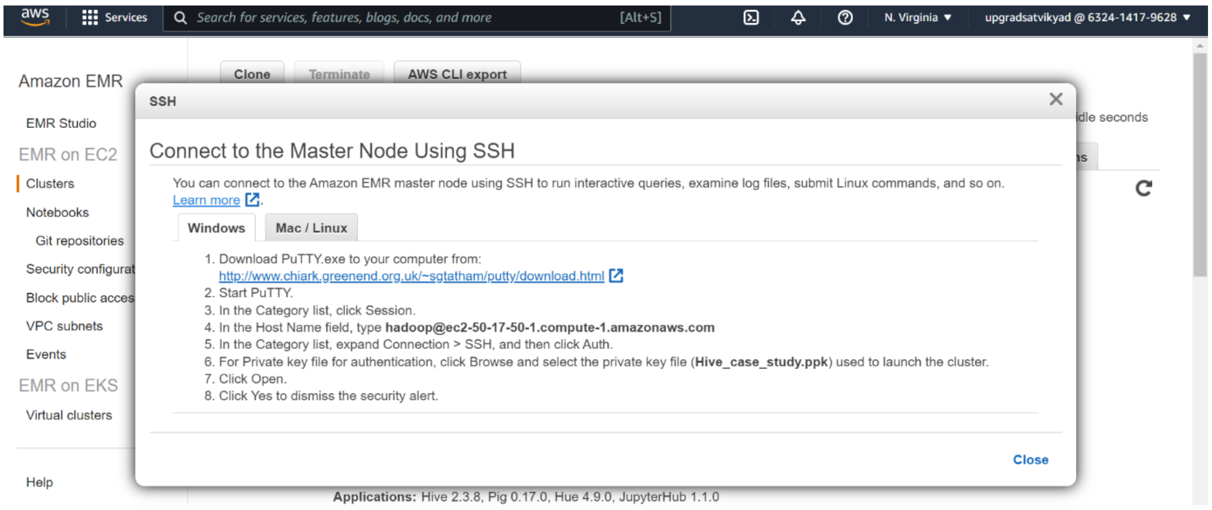
Save the generated key — Save public key — Save private key

Parameters

Type of key to generate: [✓ RSA] DSA ECDSA EdDSA SSH-1 (RSA)

Number of bits in a generated key: 2048

- When the cluster is in running state we click on Master public DNS.



SSH

Connect to the Master Node Using SSH

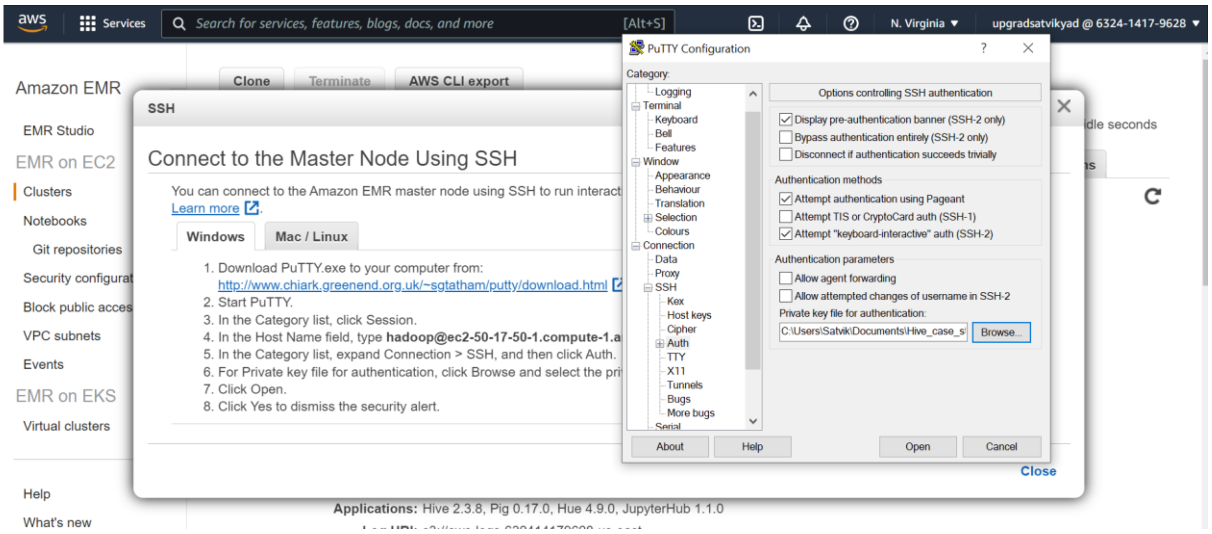
You can connect to the Amazon EMR master node using SSH to run interactive queries, examine log files, submit Linux commands, and so on. Learn more.

Windows | Mac / Linux

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@ec2-50-17-50-1.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.

Applications: Hive 2.3.8, Pig 0.17.0, Hue 4.9.0, JupyterHub 1.1.0

- We need to open the PuTTY application and provide the Host Name(Master Node DNS) and then click on Connection>SSH>Auth, browse to the private key file location. Select the .ppk file and connect to the master node.



PuTTY Configuration

Category:

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

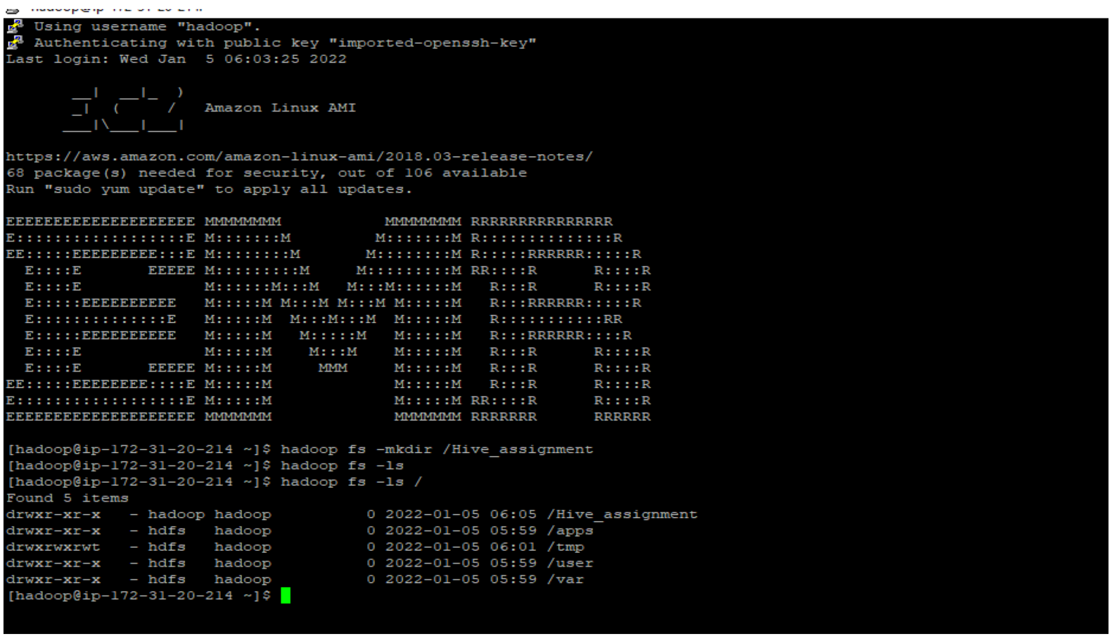
- [ ] Allow agent forwarding
- [ ] Allow attempted changes of username in SSH-2

Private key file for authentication: C:\Users\Satvik\Documents\Hive_case_s

Connection to Hadoop is successful.

Load the data sets into HDFS from S3:

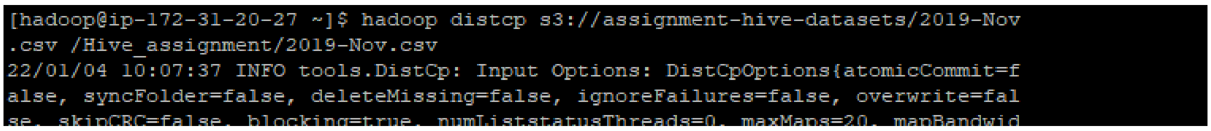
Create a directory named ‘Hive_assignment’ in Hadoop.



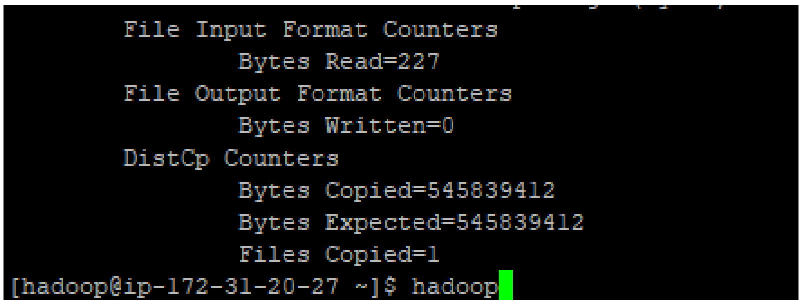
```
Using username "hadoop".
Authenticating with public key "imported-openssh-key"
Last login: Wed Jan  5 06:03:25 2022

       __|  __|_  )
       _|  (     /   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-172-31-20-214 ~]$ hadoop fs -mkdir /Hive_assignment
[hadoop@ip-172-31-20-214 ~]$ hadoop fs -ls
[hadoop@ip-172-31-20-214 ~]$ hadoop fs -ls /
Found 5 items
drwxr-xr-x   - hadoop hadoop          0 2022-01-05 06:05 /Hive_assignment
drwxr-xr-x   - hdfs   hadoop          0 2022-01-05 05:59 /apps
drwxrwxrwt   - hdfs   hadoop          0 2022-01-05 06:01 /tmp
drwxr-xr-x   - hdfs   hadoop          0 2022-01-05 05:59 /user
drwxr-xr-x   - hdfs   hadoop          0 2022-01-05 05:59 /var
[hadoop@ip-172-31-20-214 ~]$
```

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://assignment-hive-datasets/2019-Nov.csv/ Hive_assignment/2019-Nov.csv**

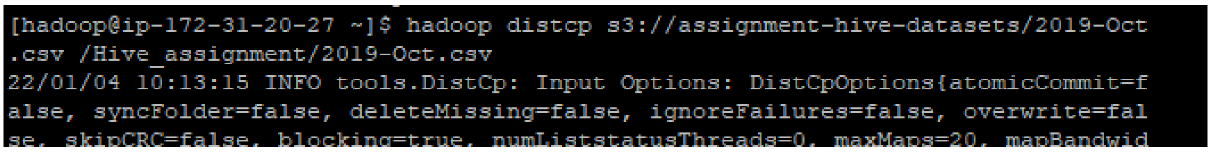


```
[hadoop@ip-172-31-20-27 ~]$ hadoop distcp s3://assignment-hive-datasets/2019-Nov
.csv /Hive_assignment/2019-Nov.csv
22/01/04 10:07:37 INFO tools.DistCp: Input Options: DistCpOptions{atomicCommit=f
alse, syncFolder=false, deleteMissing=false, ignoreFailures=false, overwrite=fal
se, skipCRC=false, blocking=true, numListstatusThreads=0, maxMaps=20, mapBandwid
```

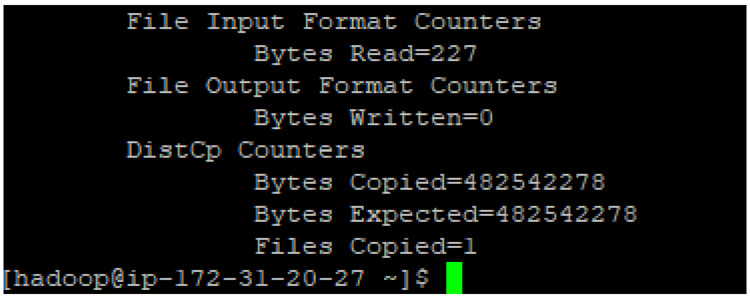


```
        File Input Format Counters
                Bytes Read=227
        File Output Format Counters
                Bytes Written=0
        DistCp Counters
                Bytes Copied=545839412
                Bytes Expected=545839412
                Files Copied=1
[hadoop@ip-172-31-20-27 ~]$ hadoop
```

Command: hadoop distcp s3://assignment-hive-datasets/2019-Oct.csv/ Hive_assignment/2019-Oct.csv

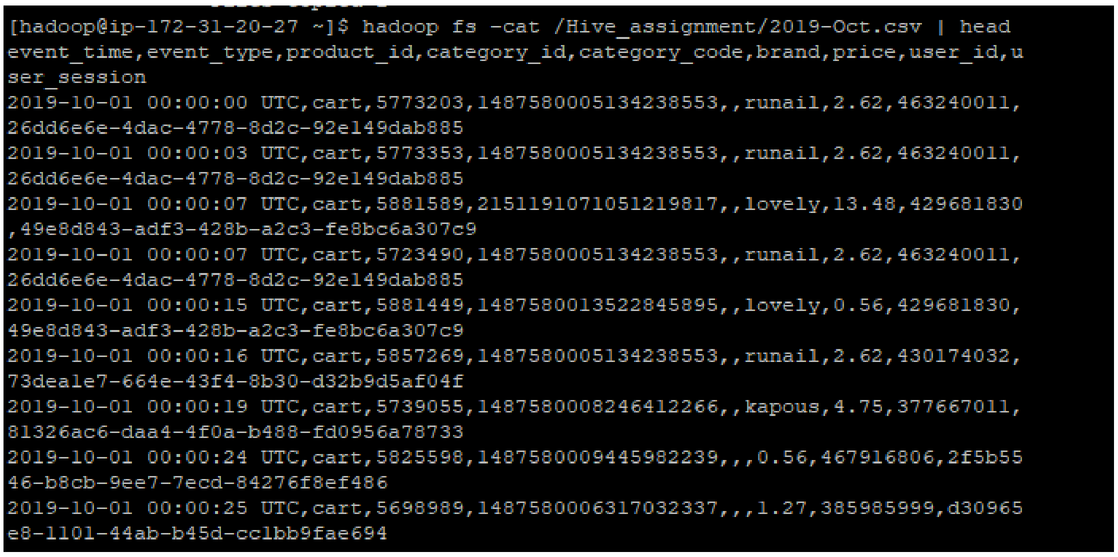


```
[hadoop@ip-172-31-20-27 ~]$ hadoop distcp s3://assignment-hive-datasets/2019-Oct
.csv /Hive_assignment/2019-Oct.csv
22/01/04 10:13:15 INFO tools.DistCp: Input Options: DistCpOptions{atomicCommit=f
alse, syncFolder=false, deleteMissing=false, ignoreFailures=false, overwrite=fal
se, skipCRC=false, blocking=true, numListstatusThreads=0, maxMaps=20, mapBandwid
```



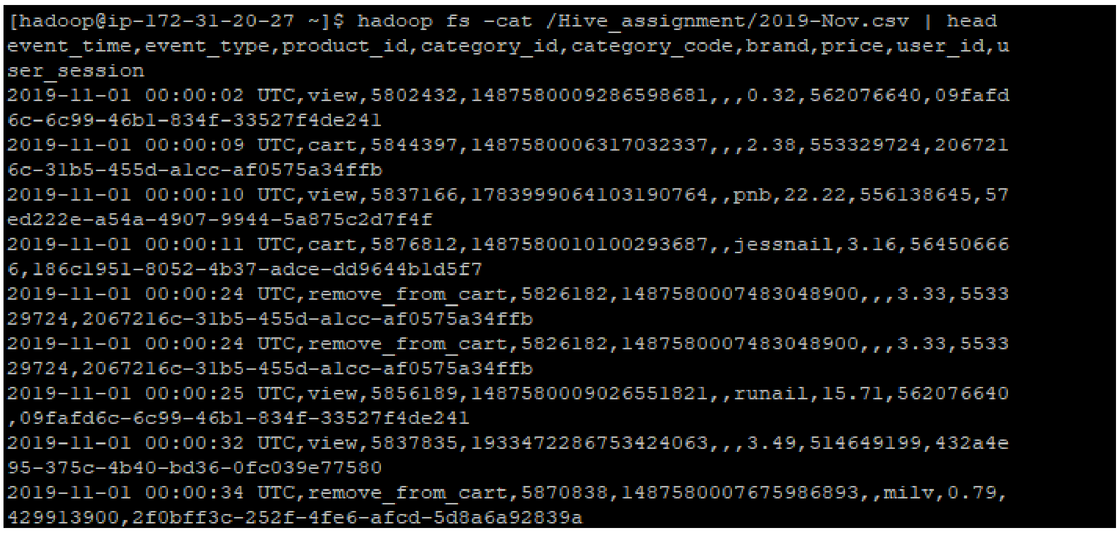
```
        File Input Format Counters
                Bytes Read=227
        File Output Format Counters
                Bytes Written=0
        DistCp Counters
                Bytes Copied=482542278
                Bytes Expected=482542278
                Files Copied=1
[hadoop@ip-172-31-20-27 ~]$
```

View the data in HDFS by executing below commands:
Oct 2019 Sales data:



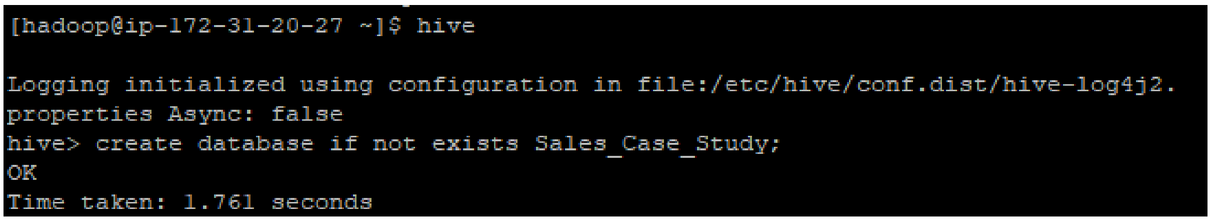
```
[hadoop@ip-172-31-20-27 ~]$ hadoop fs -cat /Hive_assignment/2019-Oct.csv | head
event_time,event_type,product_id,category_id,category_code,brand,price,user_id,u
ser_session
2019-10-01 00:00:00 UTC,cart,5773203,1487580005134238553,,runail,2.62,463240011,
26dd6e6e-4dac-4778-8d2c-92e149dab885
2019-10-01 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,
73deale7-664e-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,,,0.56,467916806,2f5b55
46-b8cb-9ee7-7ecd-84276f8ef486
2019-10-01 00:00:25 UTC,cart,5698989,1487580006317032337,,,1.27,385985999,d30965
e8-1101-44ab-b45d-cc1bb9fae694
```

Nov 2019 Sales Data:



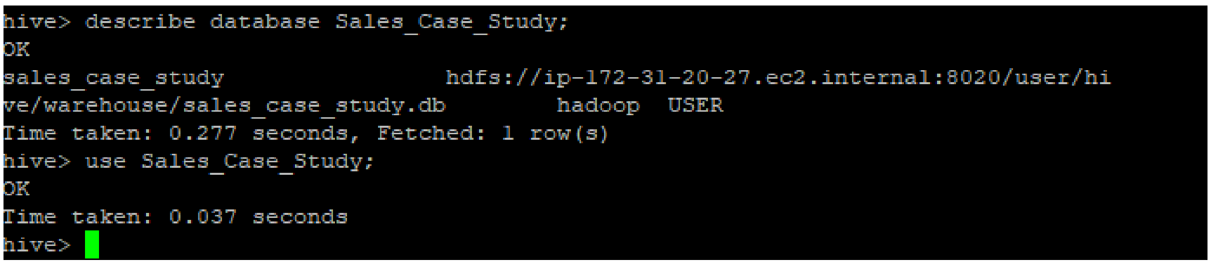
```
[hadoop@ip-172-31-20-27 ~]$ hadoop fs -cat /Hive_assignment/2019-Nov.csv | head
event_time,event_type,product_id,category_id,category_code,brand,price,user_id,u
ser_session
2019-11-01 00:00:02 UTC,view,5802432,1487580009286598681,,,0.32,562076640,09fafd
6c-6c99-46b1-834f-33527f4de241
2019-11-01 00:00:09 UTC,cart,5844397,1487580006317032337,,,2.38,553329724,206721
6c-31b5-455d-a1cc-af0575a34ffb
2019-11-01 00:00:10 UTC,view,5837166,1783999064103190764,,pnb,22.22,556138645,57
ed222e-a54a-4907-9944-5a875c2d7f4f
2019-11-01 00:00:11 UTC,cart,5876812,1487580010100293687,,jessnail,3.16,56450666
6,186c1951-8052-4b37-adce-dd9644b1d5f7
2019-11-01 00:00:24 UTC,remove_from_cart,5826182,1487580007483048900,,,3.33,5533
29724,2067216c-31b5-455d-a1cc-af0575a34ffb
2019-11-01 00:00:24 UTC,remove_from_cart,5826182,1487580007483048900,,,3.33,5533
29724,2067216c-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,,,3.49,514649199,432a4e
95-375c-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 the data, it’s time to Set the data in Hive - Launch Hive :



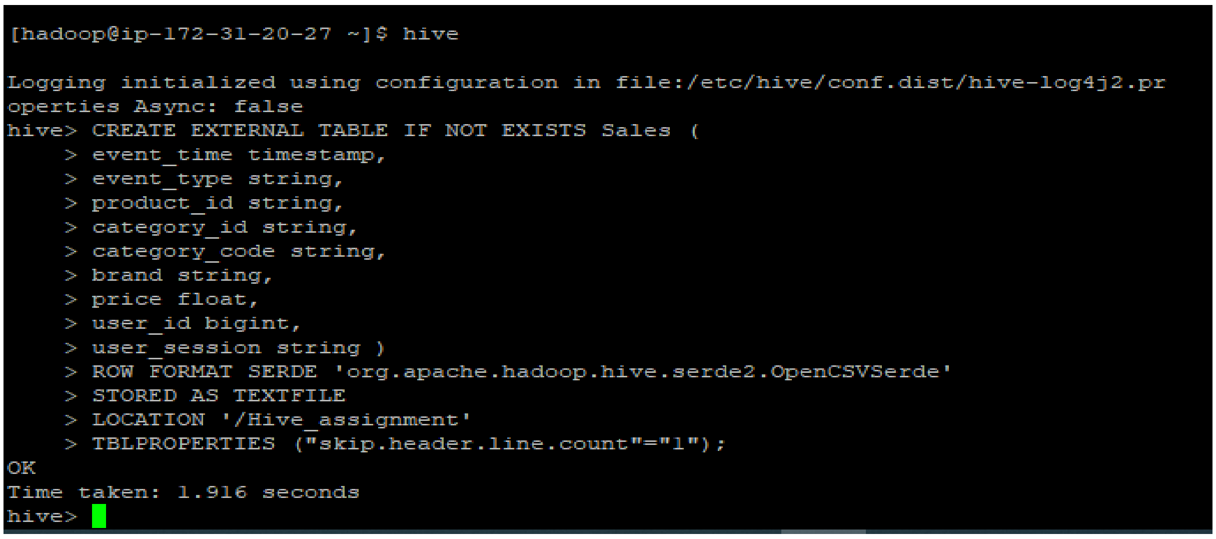
```
[hadoop@ip-172-31-20-27 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.
properties Async: false
hive> create database if not exists Sales_Case_Study;
OK
Time taken: 1.761 seconds
```

Creating database, creating tables :



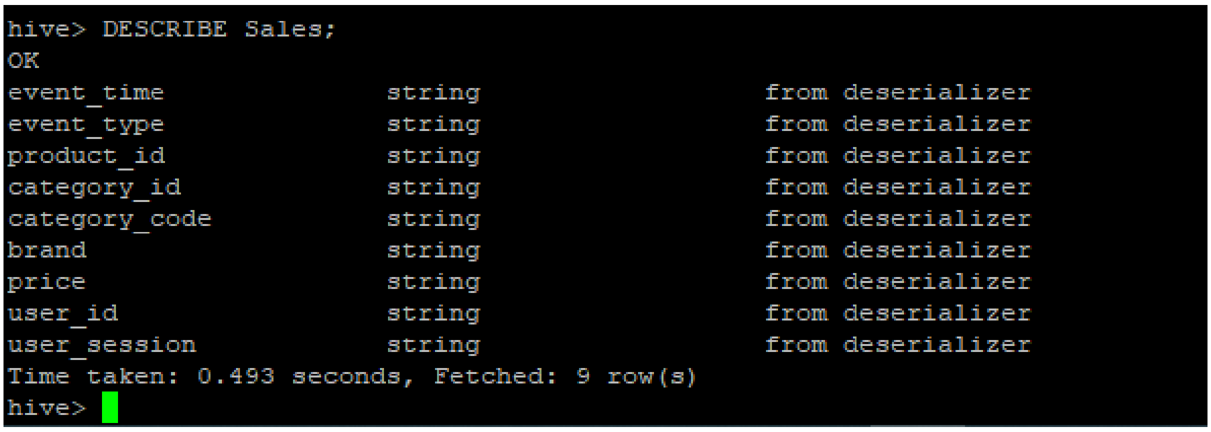
```
hive> describe database Sales_Case_Study;
OK
sales_case_study                hdfs://ip-172-31-20-27.ec2.internal:8020/user/hi
ve/warehouse/sales_case_study.db        hadoop  USER
Time taken: 0.277 seconds, Fetched: 1 row(s)
hive> use Sales_Case_Study;
OK
Time taken: 0.037 seconds
hive>
```

Creating a table from the raw data given by taking care of the data dictionary:



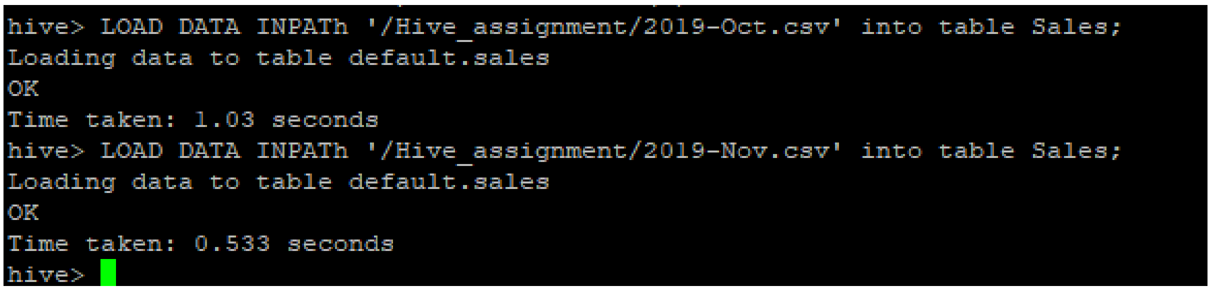
```
[hadoop@ip-172-31-20-27 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.pr
operties Async: false
hive> CREATE EXTERNAL TABLE IF NOT EXISTS Sales (
    > 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: 1.916 seconds
hive>
```

The Table schema:



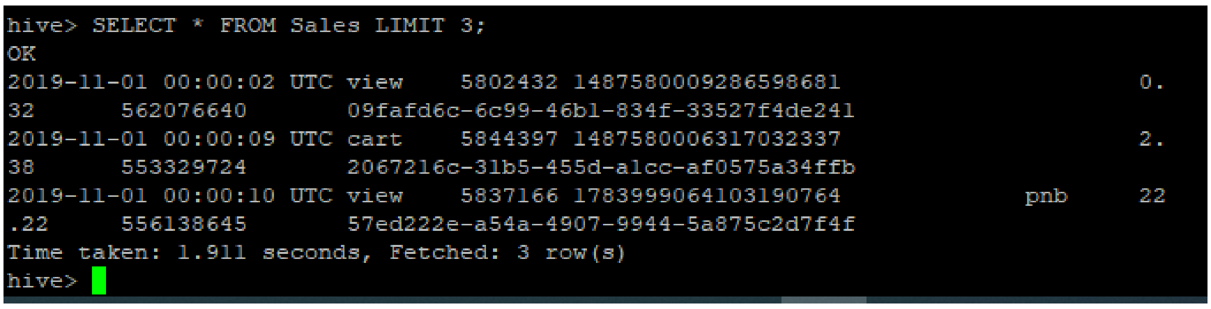
```
hive> DESCRIBE Sales;
OK
event_time              string                  from deserializer
event_type              string                  from deserializer
product_id              string                  from deserializer
category_id             string                  from deserializer
category_code           string                  from deserializer
brand                   string                  from deserializer
price                   string                  from deserializer
user_id                 string                  from deserializer
user_session            string                  from deserializer
Time taken: 0.493 seconds, Fetched: 9 row(s)
hive>
```

Loading data from both the files into this table:



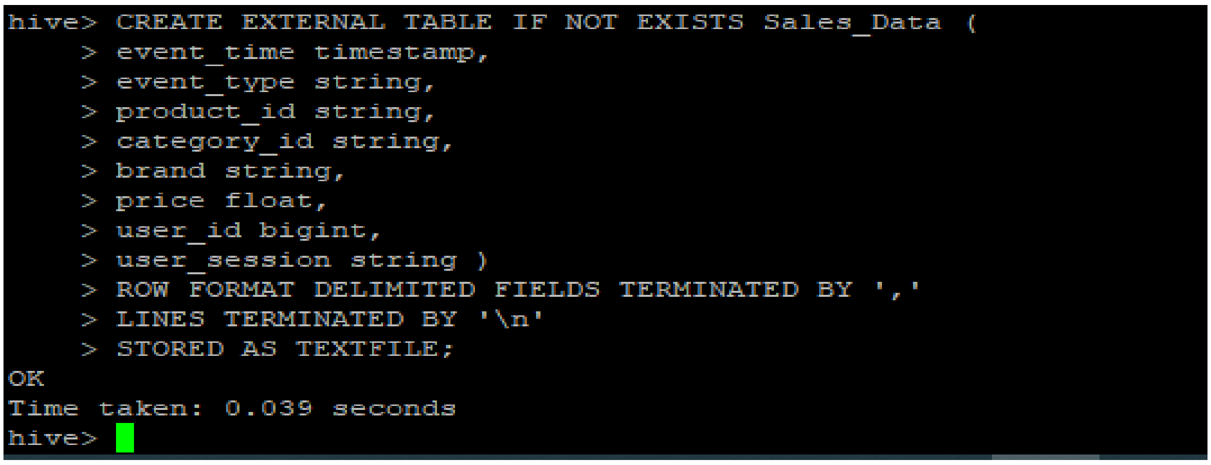
```
hive> LOAD DATA INPATh '/Hive_assignment/2019-Oct.csv' into table Sales;
Loading data to table default.sales
OK
Time taken: 1.03 seconds
hive> LOAD DATA INPATh '/Hive_assignment/2019-Nov.csv' into table Sales;
Loading data to table default.sales
OK
Time taken: 0.533 seconds
hive>
```

Viewing the data after loading them into the table:



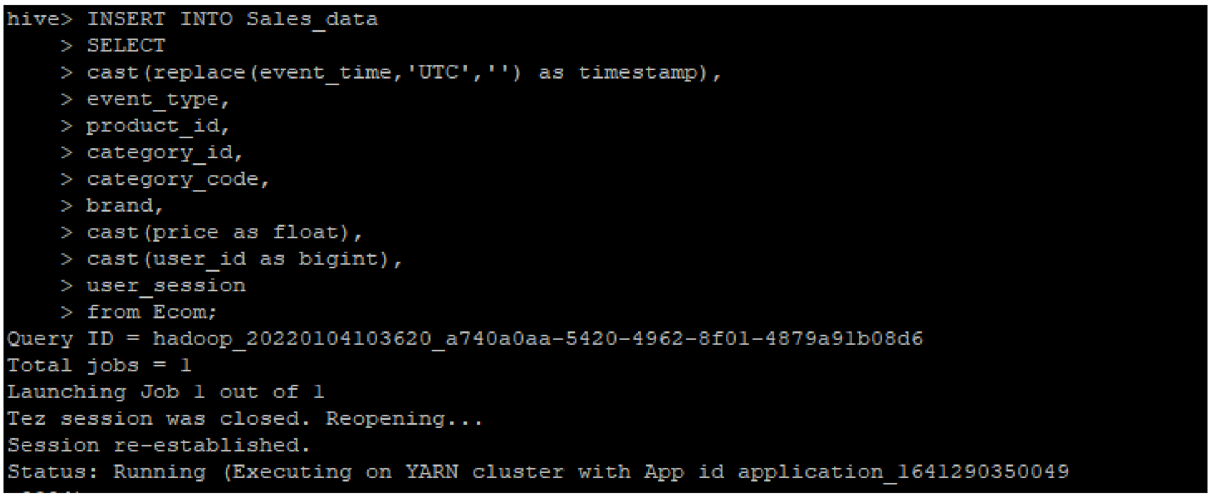
```
hive> SELECT * FROM Sales LIMIT 3;
OK
2019-11-01 00:00:02 UTC view    5802432 1487580009286598681                   0.
32      562076640       09fafd6c-6c99-46b1-834f-33527f4de241
2019-11-01 00:00:09 UTC cart    5844397 1487580006317032337                   2.
38      553329724       2067216c-31b5-455d-a1cc-af0575a34ffb
2019-11-01 00:00:10 UTC view    5837166 1783999064103190764           pnb     22
.22     556138645       57ed222e-a54a-4907-9944-5a875c2d7f4f
Time taken: 1.911 seconds, Fetched: 3 row(s)
hive>
```

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



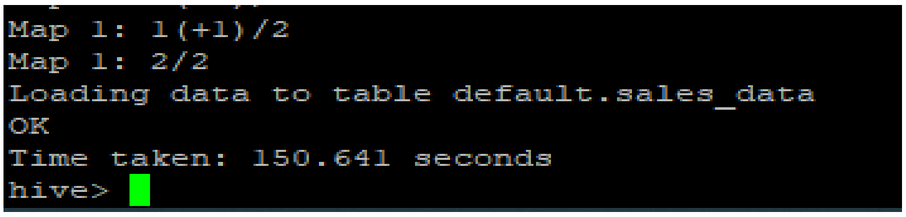
```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS Sales_Data (
    > event_time timestamp,
    > event_type string,
    > product_id string,
    > category_id string,
    > brand string,
    > price float,
    > user_id bigint,
    > user_session string )
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > LINES TERMINATED BY '\n'
    > STORED AS TEXTFILE;
OK
Time taken: 0.039 seconds
hive>
```

Inserting the data into this table (Sales_Data).



```
hive> INSERT INTO Sales_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_20220104103620_a740a0aa-5420-4962-8f01-4879a91b08d6
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_1641290350049
```

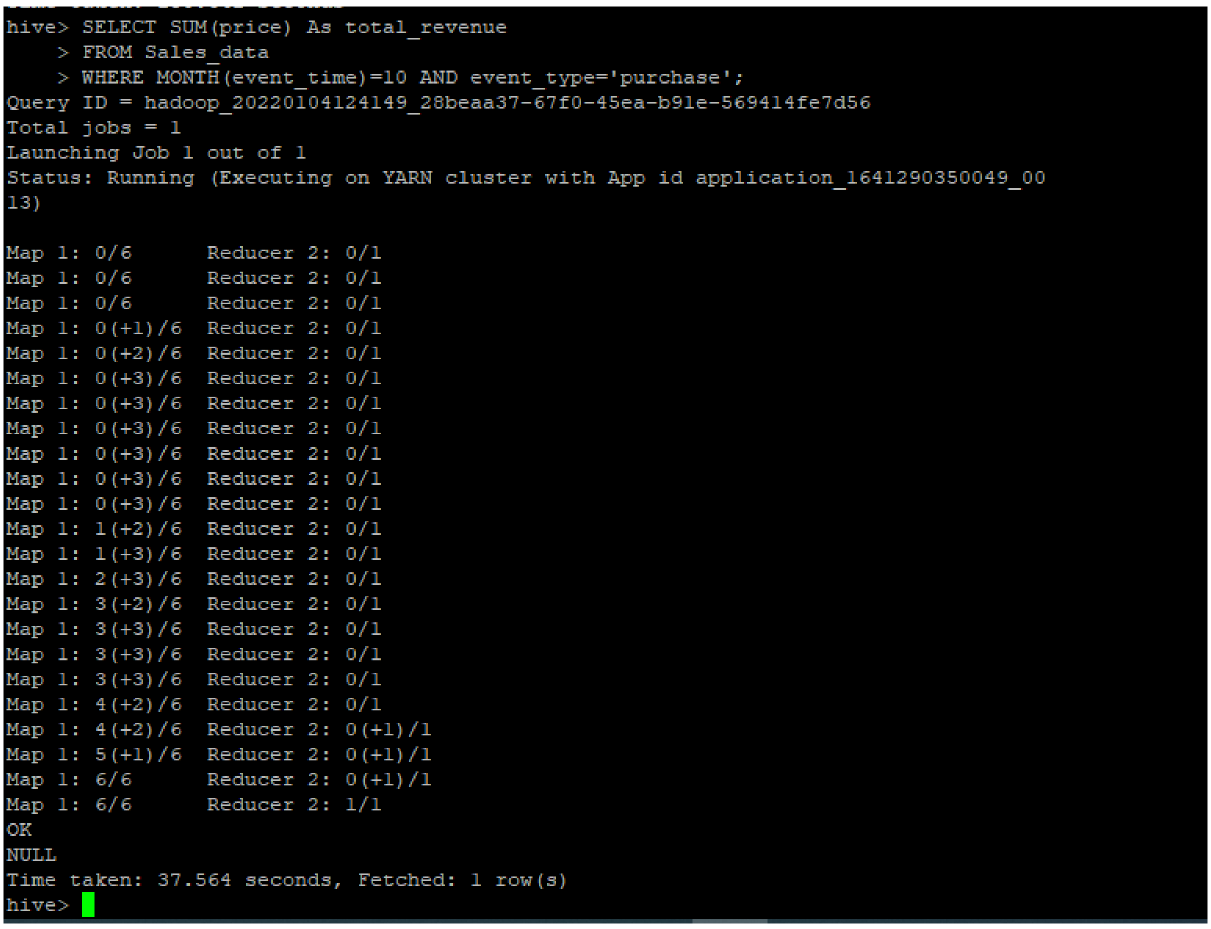
Data loaded successfully:



```
Map 1: 1(+1)/2
Map 1: 2/2
Loading data to table default.sales_data
OK
Time taken: 150.641 seconds
hive>
```

Running the first query in order to check the total time taken by the query to run successfully without partitioning:

Q1. Find the total revenue generated due to purchases made in October. (using the Sales_data table)



```
hive> SELECT SUM(price) As total_revenue
    > FROM Sales_data
    > WHERE MONTH(event_time)=10 AND event_type='purchase';
Query ID = hadoop_20220104124149_28beaa37-67f0-45ea-b91e-569414fe7d56
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641290350049_00
13)

Map 1: 0/6      Reducer 2: 0/1
Map 1: 0/6      Reducer 2: 0/1
Map 1: 0/6      Reducer 2: 0/1
Map 1: 0(+1)/6  Reducer 2: 0/1
Map 1: 0(+2)/6  Reducer 2: 0/1
Map 1: 0(+3)/6  Reducer 2: 0/1
Map 1: 0(+3)/6  Reducer 2: 0/1
Map 1: 0(+3)/6  Reducer 2: 0/1
Map 1: 0(+3)/6  Reducer 2: 0/1
Map 1: 0(+3)/6  Reducer 2: 0/1
Map 1: 0(+3)/6  Reducer 2: 0/1
Map 1: 1(+2)/6  Reducer 2: 0/1
Map 1: 1(+3)/6  Reducer 2: 0/1
Map 1: 2(+3)/6  Reducer 2: 0/1
Map 1: 3(+2)/6  Reducer 2: 0/1
Map 1: 3(+3)/6  Reducer 2: 0/1
Map 1: 3(+3)/6  Reducer 2: 0/1
Map 1: 3(+3)/6  Reducer 2: 0/1
Map 1: 4(+2)/6  Reducer 2: 0/1
Map 1: 4(+2)/6  Reducer 2: 0(+1)/1
Map 1: 5(+1)/6  Reducer 2: 0(+1)/1
Map 1: 6/6      Reducer 2: 0(+1)/1
Map 1: 6/6      Reducer 2: 1/1
OK
NULL
Time taken: 37.564 seconds, Fetched: 1 row(s)
hive>
```

SELECT SUM(price) AS total_revenue

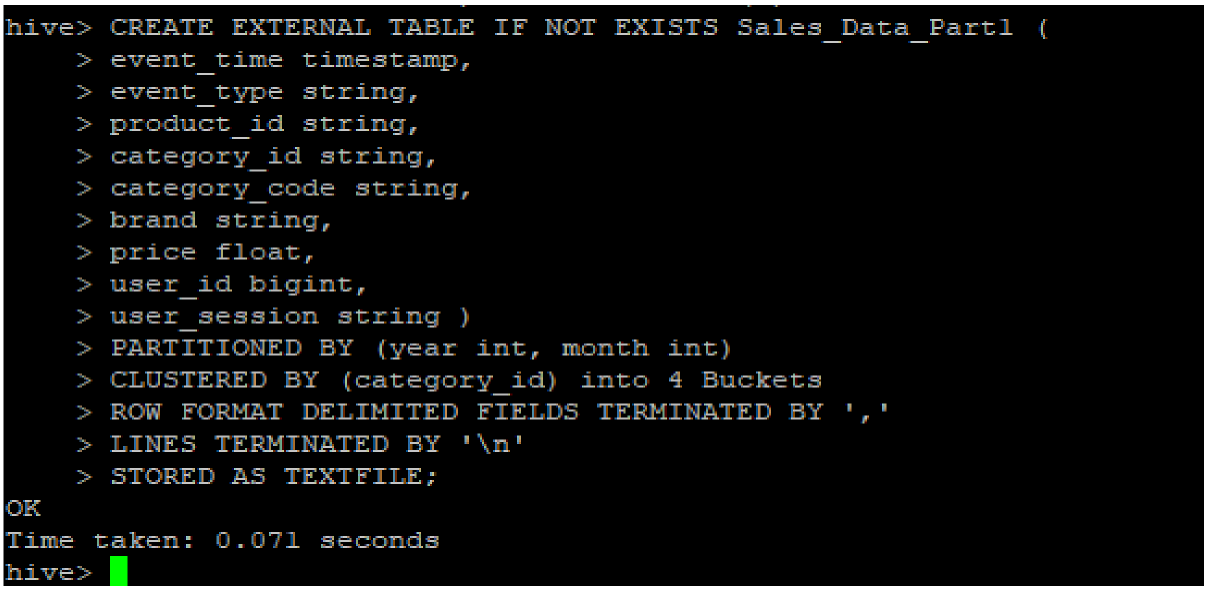
FROM Sales_data

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

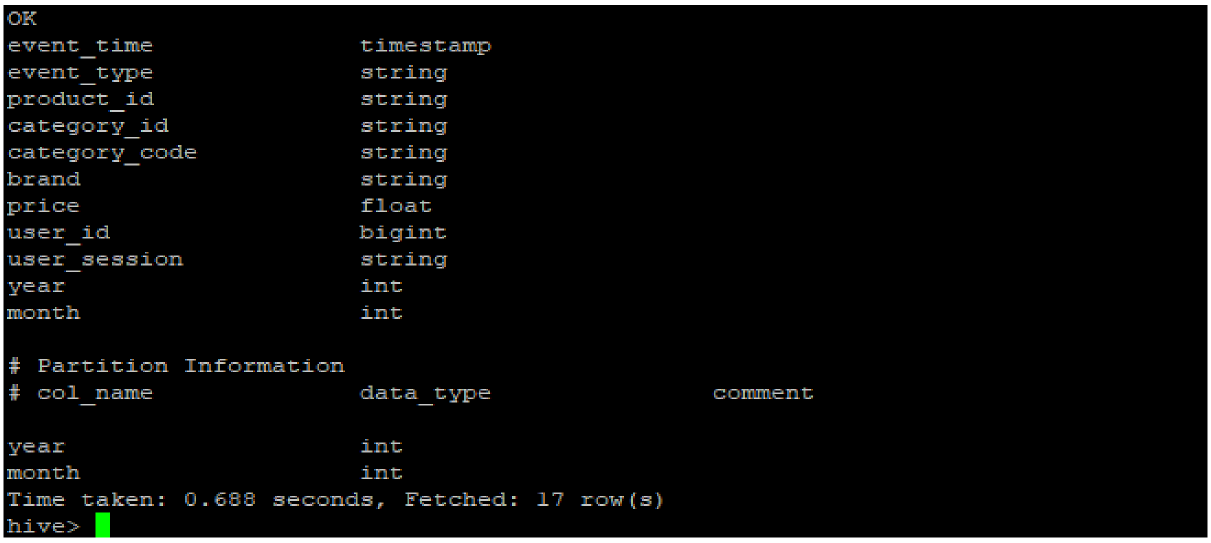
Time Taken by the entire query without table optimization (i.e. without partitioning) : 37.564 seconds

Note: Once the base table is created, we need to optimize the table for quick query result through partitioning and bucketing. Our first optimized table name is Sales_data_part1.

Now we will be enabling Dynamic Partitioning and creating a partitioned table with buckets.

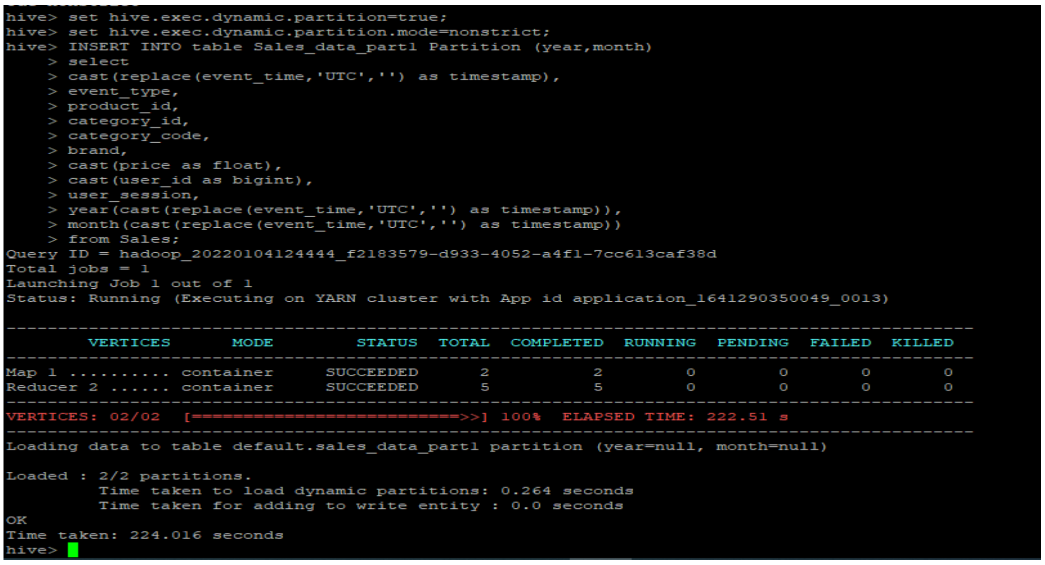


```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS Sales_Data_Part1 (
    > event_time timestamp,
    > event_type string,
    > product_id string,
    > category_id string,
    > category_code string,
    > brand string,
    > price float,
    > user_id bigint,
    > user_session string )
    > PARTITIONED 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: 0.071 seconds
hive>
```



```
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
Time taken: 0.688 seconds, Fetched: 17 row(s)
hive>
```

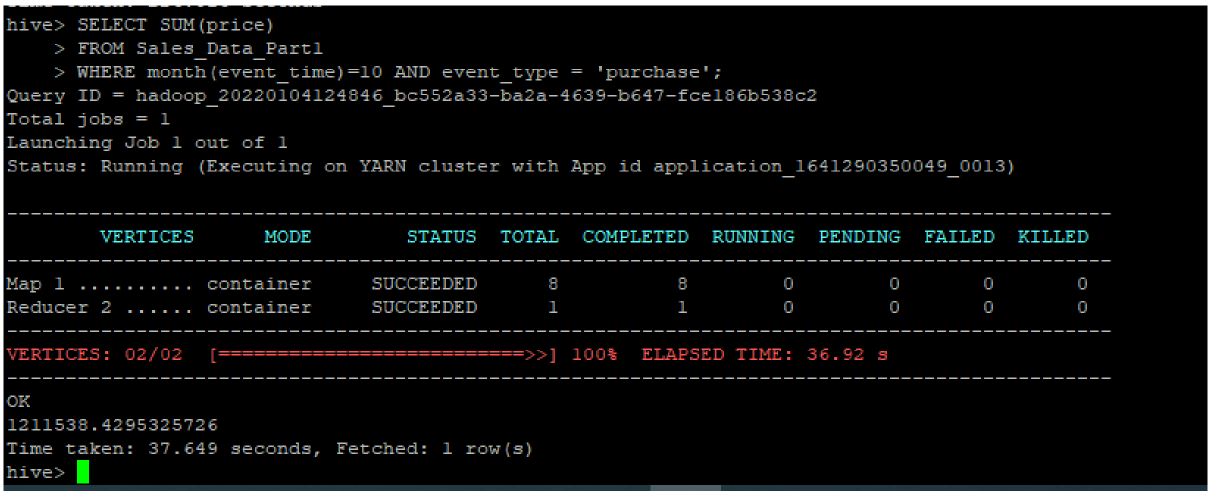
The new optimised table has been created. We will now insert the data into this table.



```
hive> set hive.exec.dynamic.partition=true;
hive> set hive.exec.dynamic.partition.mode=nonstrict;
hive> INSERT INTO table Sales_data_part1 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 Sales;
Query ID = hadoop_20220104124444_f2183579-d933-4052-a4f1-7cc613caf38d
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641290350049_0013)

----------------------------------------------------------------------------------------------
        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: 222.51 s
----------------------------------------------------------------------------------------------
Loading data to table default.sales_data_part1 partition (year=null, month=null)

Loaded : 2/2 partitions.
         Time taken to load dynamic partitions: 0.264 seconds
         Time taken for adding to write entity : 0.0 seconds
OK
Time taken: 224.016 seconds
hive>
```

We will now test this table by running the same query (Q1) in this optimised table and note the time taken.

Q1. Find the total revenue generated due to purchases made in October. (using the Sales_data_part1 table)



```
hive> SELECT SUM(price)
    > FROM Sales_Data_Part1
    > WHERE month(event_time)=10 AND event_type = 'purchase';
Query ID = hadoop_20220104124846_bc552a33-ba2a-4639-b647-fce186b538c2
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641290350049_0013)

----------------------------------------------------------------------------------------------
        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: 36.92 s
----------------------------------------------------------------------------------------------
OK
1211538.4295325726
Time taken: 37.649 seconds, Fetched: 1 row(s)
hive>
```

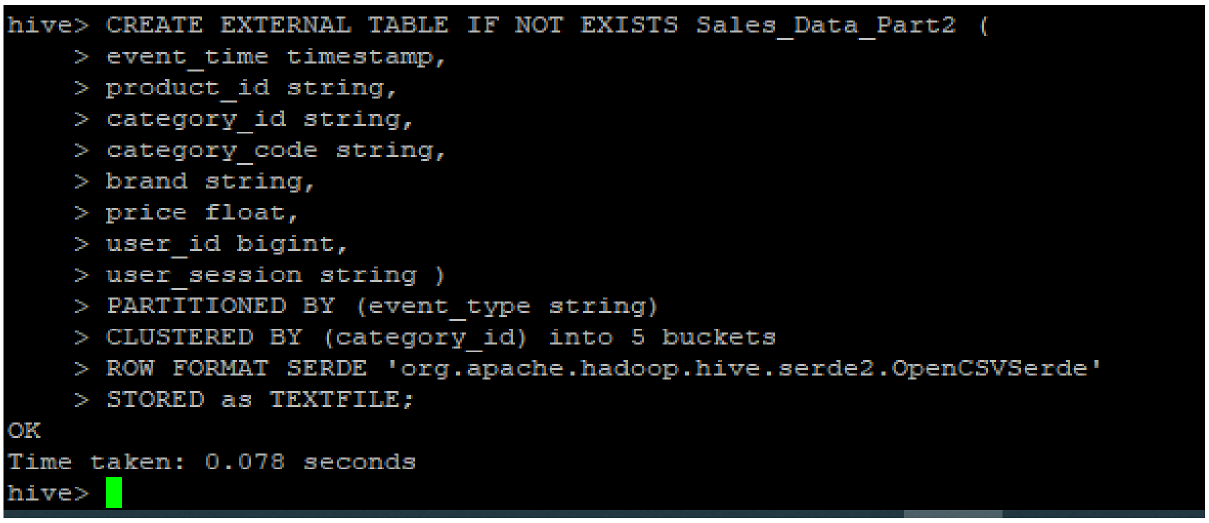
SELECT SUM(price) AS total_revenue

FROM Sales_data_part1

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

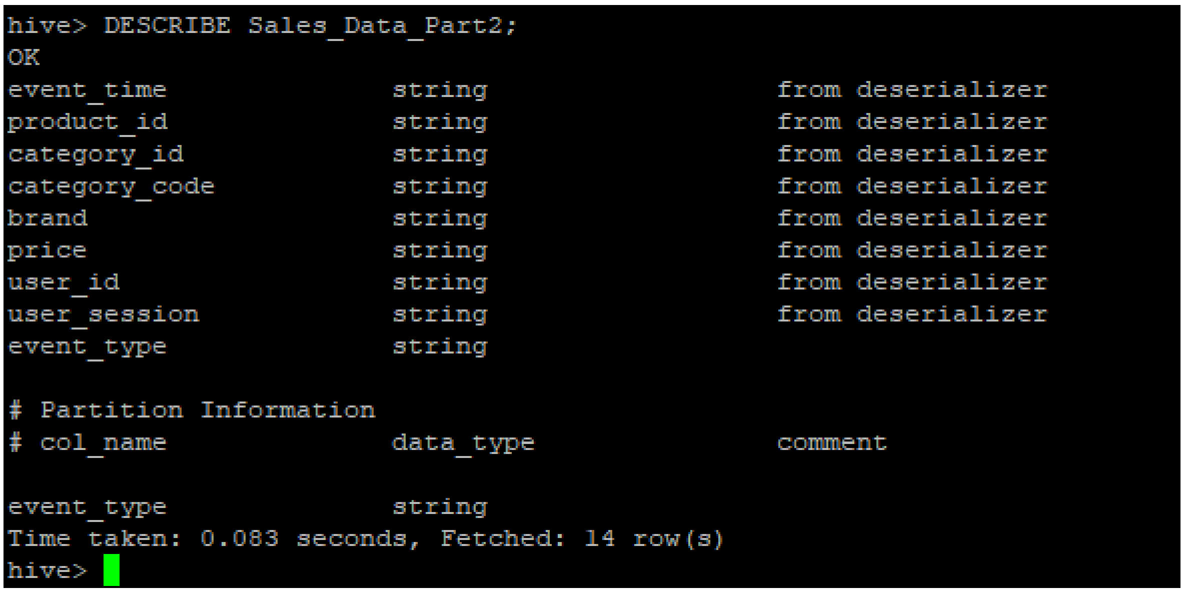
Time Taken by the entire query after dynamic partitioning of the table : 37.649 seconds

Enabling second approach for Dynamic Partitioning and creating another partitioned table with buckets.



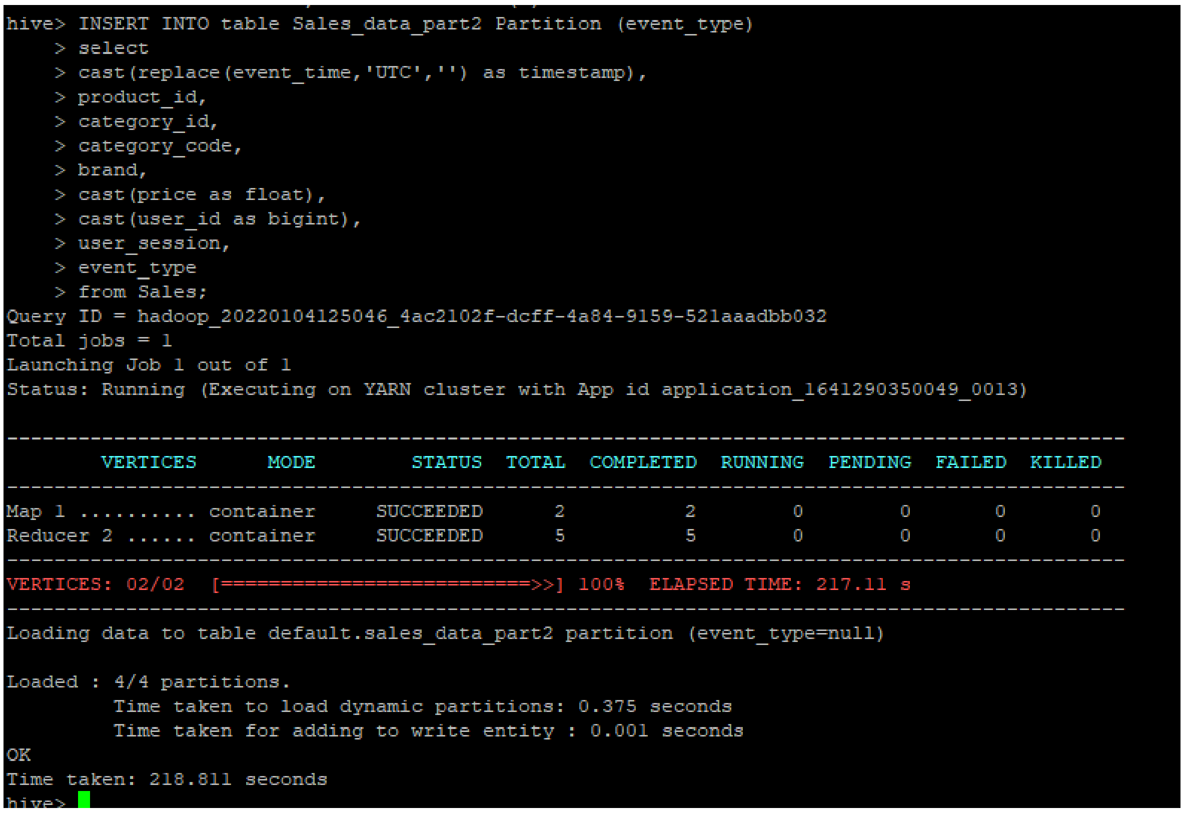
```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS Sales_Data_Part2 (
    > event_time timestamp,
    > product_id string,
    > category_id string,
    > category_code string,
    > brand string,
    > price float,
    > user_id bigint,
    > user_session string )
    > PARTITIONED BY (event_type string)
    > CLUSTERED BY (category_id) into 5 buckets
    > ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
    > STORED as TEXTFILE;
OK
Time taken: 0.078 seconds
hive>
```

The Second Partitioned table has been created . This table is named as Sales_Data_Part2.



```
hive> DESCRIBE Sales_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: 0.083 seconds, Fetched: 14 row(s)
hive>
```

We will now insert the data into this table:



```
hive> INSERT INTO table Sales_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 Sales;
Query ID = hadoop_20220104125046_4ac2102f-dcff-4a84-9159-521aaadbb032
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641290350049_0013)

----------------------------------------------------------------------------------------------
        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: 217.11 s
----------------------------------------------------------------------------------------------
Loading data to table default.sales_data_part2 partition (event_type=null)

Loaded : 4/4 partitions.
         Time taken to load dynamic partitions: 0.375 seconds
         Time taken for adding to write entity : 0.001 seconds
OK
Time taken: 218.811 seconds
hive>
```

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

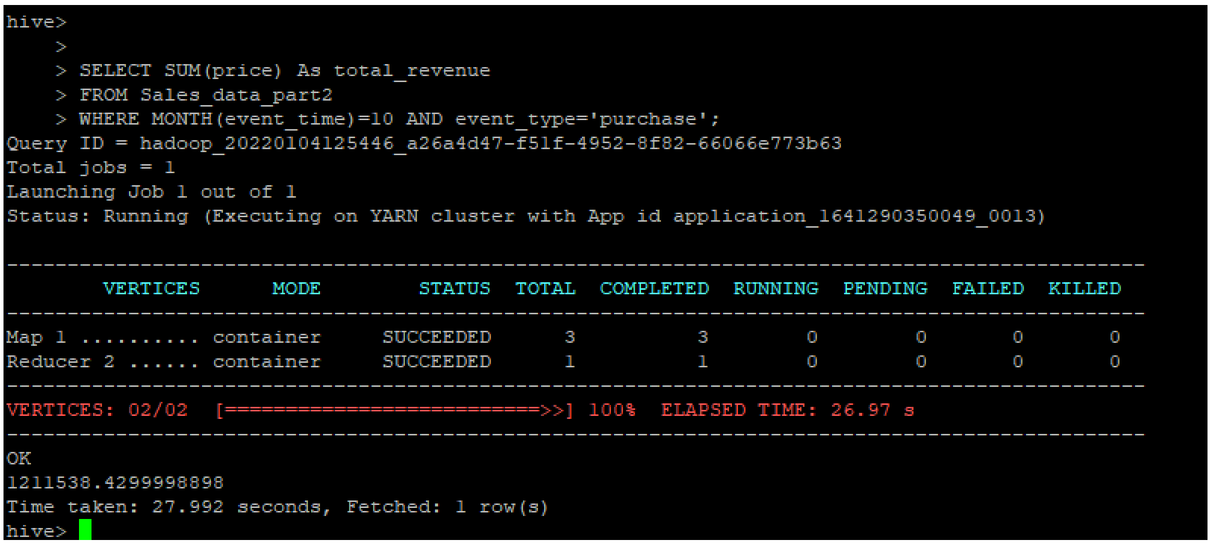
Q1. Find the total revenue generated due to purchases made in October. (using the Sales_Data_Part2 table)

SELECT SUM(price) AS total_revenue

FROM Sales_data_Part2

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

Time Taken by the entire query after dynamic partitioning of the table : 27.992 seconds

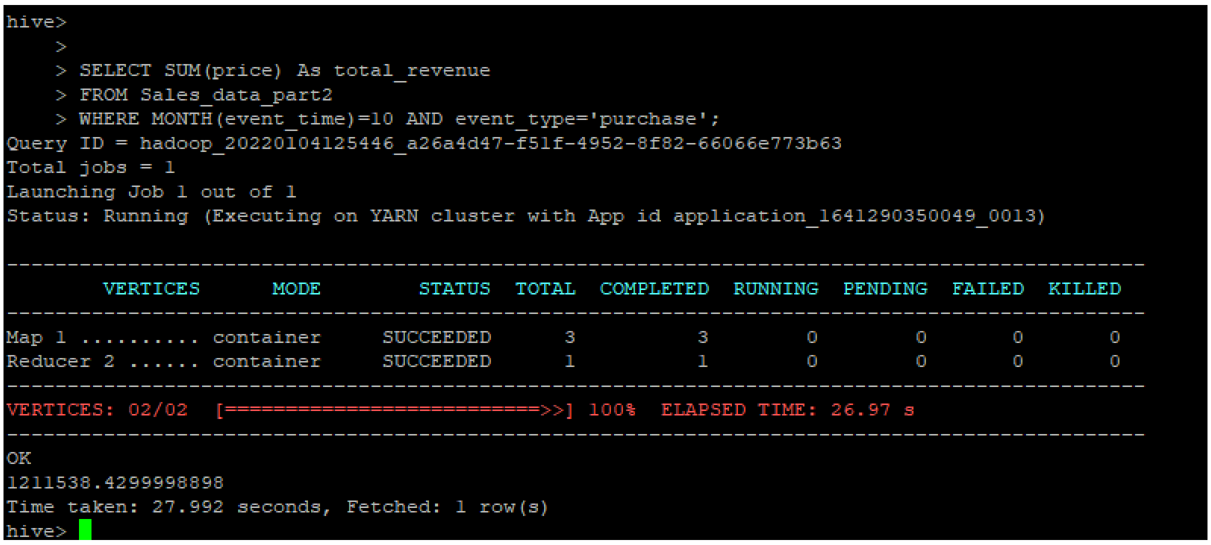


```
hive>
    >
    > SELECT SUM(price) As total_revenue
    > FROM Sales_data_part2
    > WHERE MONTH(event_time)=10 AND event_type='purchase';
Query ID = hadoop_20220104125446_a26a4d47-f51f-4952-8f82-66066e773b63
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641290350049_0013)

----------------------------------------------------------------------------------------------
        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: 26.97 s
----------------------------------------------------------------------------------------------
OK
1211538.4299998898
Time taken: 27.992 seconds, Fetched: 1 row(s)
hive>
```

<u>CONCLUSION</u> : It can be observed that, by Partition by over 'event_type' and clustering by 'category_id' we get the most optimized output of the query.

Note: The following questions are being solved using the most optimized table i.e. Sales_Data_Part2:

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

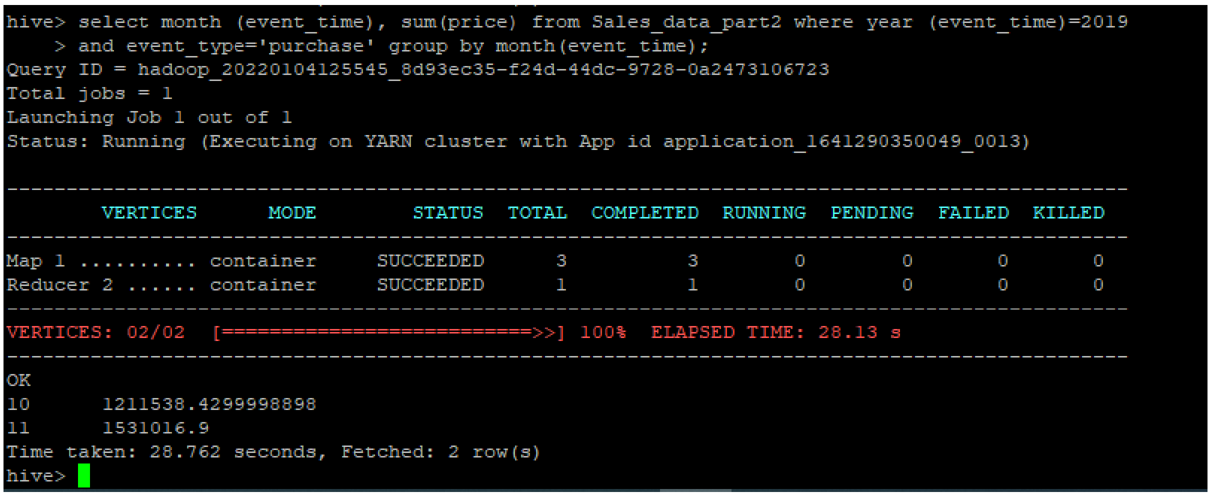


```
hive>
    >
    > SELECT SUM(price) As total_revenue
    > FROM Sales_data_part2
    > WHERE MONTH(event_time)=10 AND event_type='purchase';
Query ID = hadoop_20220104125446_a26a4d47-f51f-4952-8f82-66066e773b63
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641290350049_0013)

----------------------------------------------------------------------------------------------
        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: 26.97 s
----------------------------------------------------------------------------------------------
OK
1211538.4299998898
Time taken: 27.992 seconds, Fetched: 1 row(s)
hive>
```

SELECT SUM(price) As total_revenue
FROM Sales_data_part2
WHERE MONTH(event_time)=10 AND event_type='purchase';

OUTPUT : Total Revenue = 1211538.4299

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 Sales_data_part2 where year (event_time)=2019
    > and event_type='purchase' group by month(event_time);
Query ID = hadoop_20220104125545_8d93ec35-f24d-44dc-9728-0a2473106723
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641290350049_0013)

----------------------------------------------------------------------------------------------
        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: 28.13 s
----------------------------------------------------------------------------------------------
OK
10      1211538.4299998898
11      1531016.9
Time taken: 28.762 seconds, Fetched: 2 row(s)
hive>
```

SELECT month (event_time), sum(price)

FROM Sales_data_part2

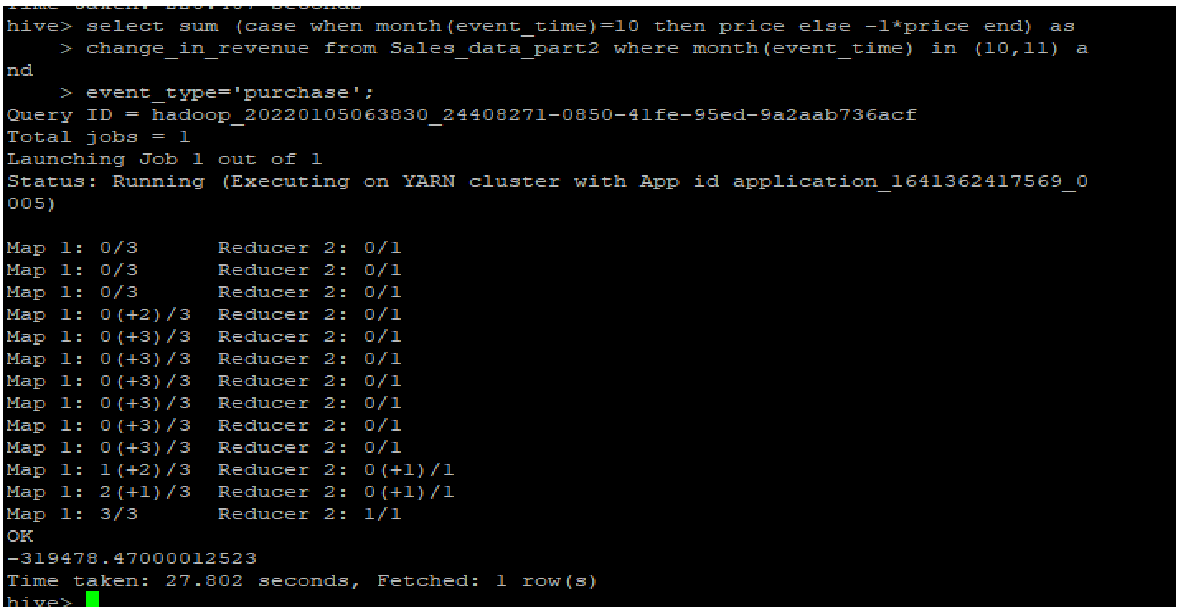
WHERE year (event_time)=2019 AND event_type='purchase'

GROUP BY month(event_time);

OUTPUT:

event_time (month)	Total_Purchase
10	1211538.4299
11	1531016.9

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



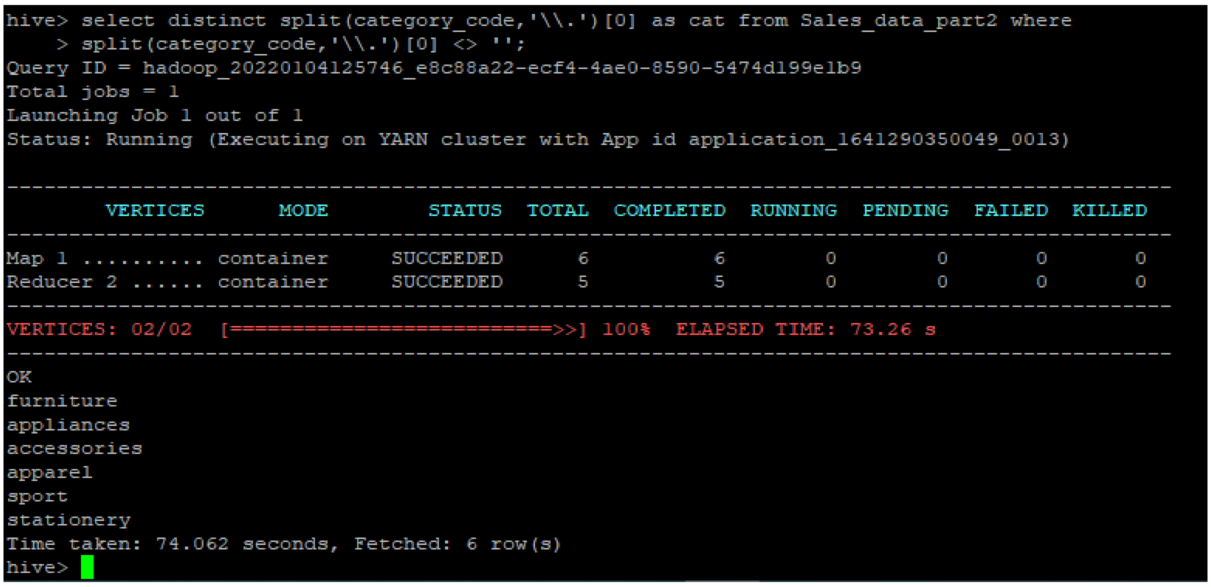
```
hive> select sum (case when month(event_time)=10 then price else -1*price end) as
    > change_in_revenue from Sales_data_part2 where month(event_time) in (10,11) a
nd
    > event_type='purchase';
Query ID = hadoop_20220105063830_24408271-0850-41fe-95ed-9a2aab736acf
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641362417569_0
005)

Map 1: 0/3      Reducer 2: 0/1
Map 1: 0/3      Reducer 2: 0/1
Map 1: 0/3      Reducer 2: 0/1
Map 1: 0(+2)/3  Reducer 2: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1
Map 1: 1(+2)/3  Reducer 2: 0(+1)/1
Map 1: 2(+1)/3  Reducer 2: 0(+1)/1
Map 1: 3/3      Reducer 2: 1/1
OK
-319478.47000012523
Time taken: 27.802 seconds, Fetched: 1 row(s)
hive>
```

SELECT sum(case when month(event_time)=10 then price else -1*price end) as change_in_revenue FROM Sales_data_part2

WHERE month(event_time) in (10,11) and event_type='purchase';

OUTPUT: -319478.4700

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



```
hive> select distinct split(category_code,'\\.')[0] as cat from Sales_data_part2 where
    > split(category_code,'\\.')[0] <> '';
Query ID = hadoop_20220104125746_e8c88a22-ecf4-4ae0-8590-5474d199e1b9
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641290350049_0013)

----------------------------------------------------------------------------------------------
        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: 73.26 s
----------------------------------------------------------------------------------------------
OK
furniture
appliances
accessories
apparel
sport
stationery
Time taken: 74.062 seconds, Fetched: 6 row(s)
hive>
```

SELECT DISTINCT split(category_code,'\\.')[0] as cat

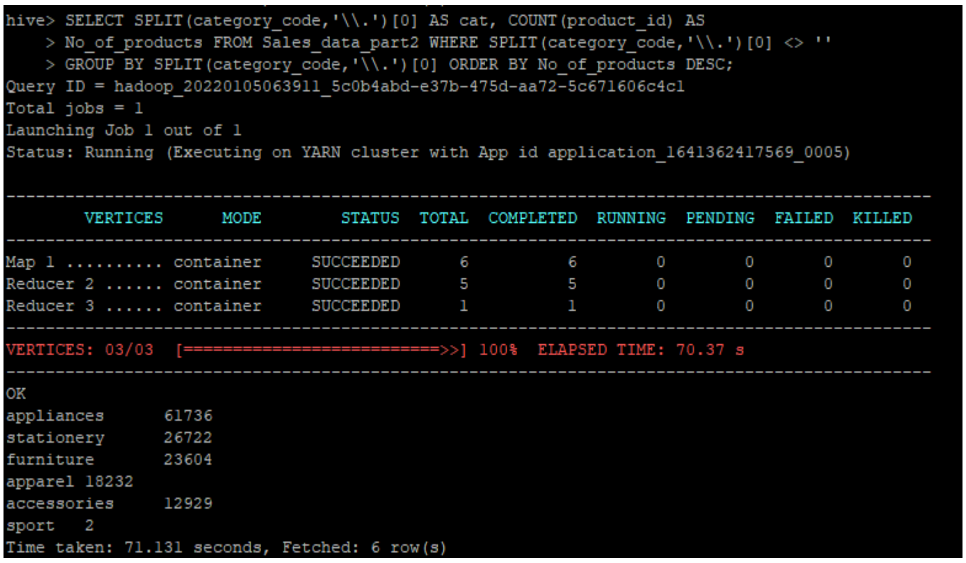
FROM Sales_data_part2

WHERE split(category_code,'\\.')[0] <> '';

OUTPUT:

Categories of Product
Furniture
Appliances
Accessories
Apparel
Sport
Stationery

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



```
hive> SELECT SPLIT(category_code,'\\.')[0] AS cat, COUNT(product_id) AS
    > No_of_products FROM Sales_data_part2 WHERE SPLIT(category_code,'\\.')[0] <> ''
    > GROUP BY SPLIT(category_code,'\\.')[0] ORDER BY No_of_products DESC;
Query ID = hadoop_20220105063911_5c0b4abd-e37b-475d-aa72-5c671606c4c1
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641362417569_0005)

----------------------------------------------------------------------------------------------
        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: 70.37 s
----------------------------------------------------------------------------------------------
OK
appliances      61736
stationery      26722
furniture       23604
apparel 18232
accessories     12929
sport   2
Time taken: 71.131 seconds, Fetched: 6 row(s)
```

SELECT SPLIT(category_code,'\\.')[0] AS cat, COUNT(product_id) AS

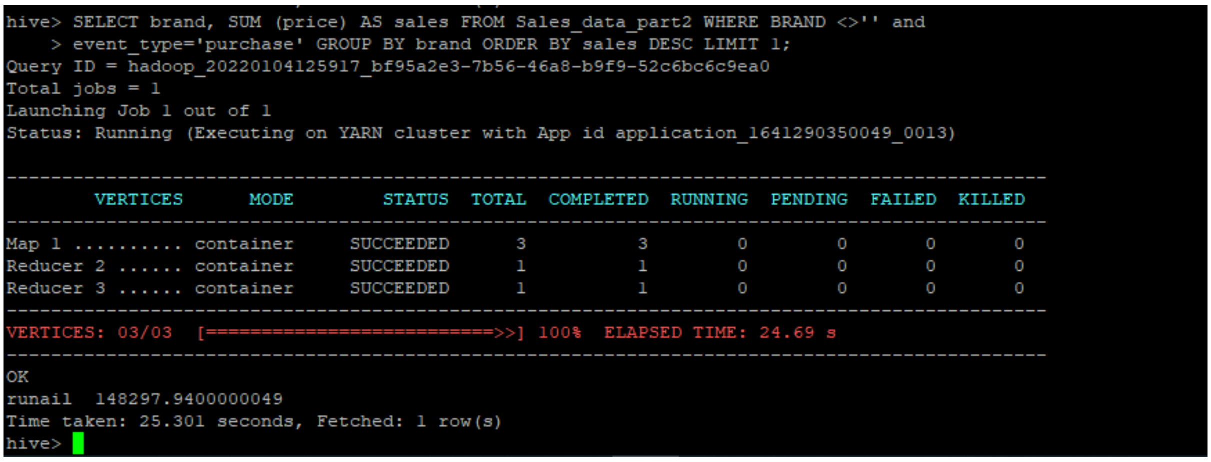
No_of_products FROM Sales_data_part2 WHERE SPLIT(category_code,'\\.')[0] <> ''

GROUP BY SPLIT(category_code,'\\.')[0] ORDER BY No_of_products DESC;

OUTPUT:

appliances	61736
Stationary	26722
Furniture	23604
Apparel	18232
Accessories	12929
Sport	2

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



```
hive> SELECT brand, SUM (price) AS sales FROM Sales_data_part2 WHERE BRAND <>'' and
    > event_type='purchase' GROUP BY brand ORDER BY sales DESC LIMIT 1;
Query ID = hadoop_20220104125917_bf95a2e3-7b56-46a8-b9f9-52c6bc6c9ea0
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641290350049_0013)

----------------------------------------------------------------------------------------------
        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: 24.69 s
----------------------------------------------------------------------------------------------
OK
runail  148297.9400000049
Time taken: 25.301 seconds, Fetched: 1 row(s)
hive>
```

SELECT brand, SUM (price) AS sales

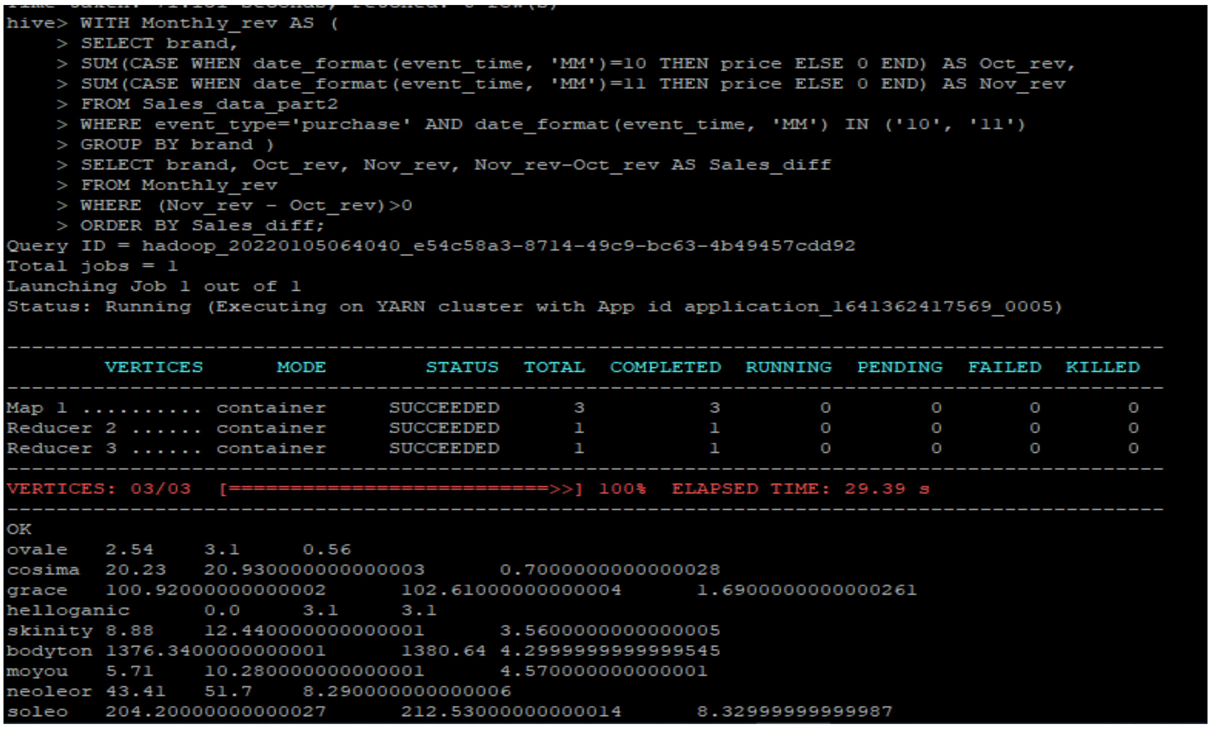
FROM Sales_data_part2

WHERE BRAND <>'' and event_type='purchase' GROUP BY brand ORDER BY sales DESC LIMIT 1;

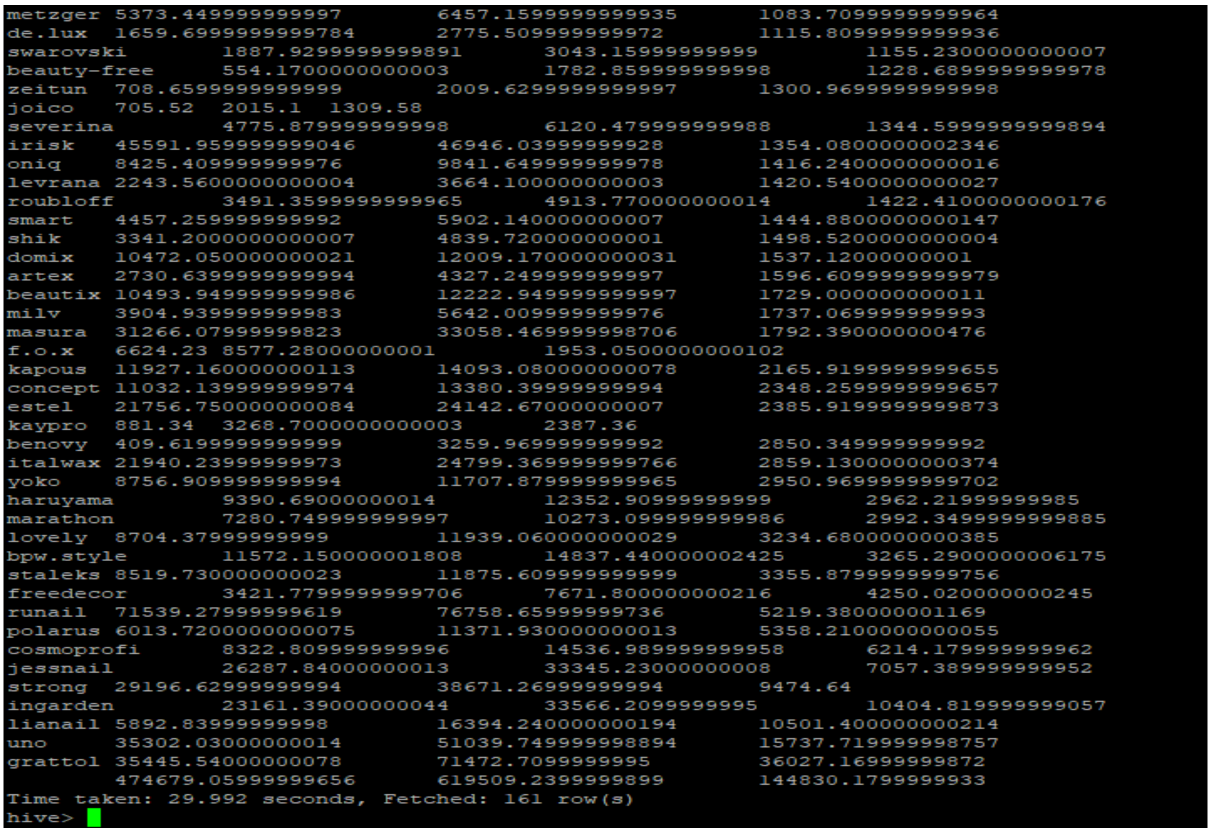
OUTPUT:

Brand Name	Max_Sales
runall	148297.94

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 Sales_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_20220105064040_e54c58a3-8714-49c9-bc63-4b49457cdd92
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641362417569_0005)

----------------------------------------------------------------------------------------------
        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: 29.39 s
----------------------------------------------------------------------------------------------
OK
ovale   2.54    3.1     0.56
cosima  20.23   20.930000000000003      0.7000000000000028
grace   100.92000000000002      102.61000000000004      1.6900000000000261
helloganic      0.0     3.1     3.1
skinity 8.88    12.440000000000001      3.5600000000000005
bodyton 1376.3400000000001      1380.64 4.2999999999999545
moyou   5.71    10.280000000000001      4.570000000000001
neoleor 43.41   51.7    8.290000000000006
soleo   204.20000000000027      212.53000000000014      8.329999999999987
```



```
metzger 5373.449999999997       6457.159999999935       1083.7099999999964
de.lux  1659.6999999999784      2775.509999999972       1115.8099999999936
swarovski       1887.9299999999891      3043.15999999999        1155.2300000000007
beauty-free     554.1700000000003       1782.859999999998       1228.6899999999978
zeitun  708.6599999999999       2009.6299999999997      1300.9699999999998
joico   705.52  2015.1  1309.58
severina        4775.879999999998       6120.479999999988       1344.5999999999894
irisk   45591.959999999046      46946.03999999928       1354.080000002346
oniq    8425.409999999976       9841.649999999978       1416.240000000016
levrana 2243.5600000000004      3664.100000000003       1420.5400000000027
roubloff        3491.3599999999965      4913.770000000014       1422.4100000000176
smart   4457.259999999992       5902.140000000007       1444.8800000000147
shik    3341.2000000000007      4839.720000000001       1498.5200000000004
domix   10472.050000000021      12009.170000000031      1537.1200000000001
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      2165.919999999655
concept 11032.139999999974      13380.39999999994       2348.259999999657
estel   21756.750000000084      24142.67000000007       2385.919999999873
kaypro  881.34  3268.7000000000003      2387.36
benovy  409.6199999999999       3259.969999999992       2850.349999999992
italwax 21940.23999999973       24799.369999999766      2859.130000000374
yoko    8756.909999999994       11707.879999999965      2950.969999999702
haruyama        9390.69000000014        12352.90999999999       2962.21999999985
marathon        7280.749999999997       10273.099999999986      2992.3499999999885
lovely  8704.37999999999        11939.060000000029      3234.680000000385
bpw.style       11572.150000001808      14837.440000002425      3265.290000006175
staleks 8519.730000000023       11875.609999999999      3355.8799999999756
freedecor       3421.7799999999706      7671.800000000216       4250.020000000245
runail  71539.27999999619       76758.65999999736       5219.38000001169
polarus 6013.7200000000075      11371.930000000013      5358.210000000055
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: 29.992 seconds, Fetched: 161 row(s)
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 Sales_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

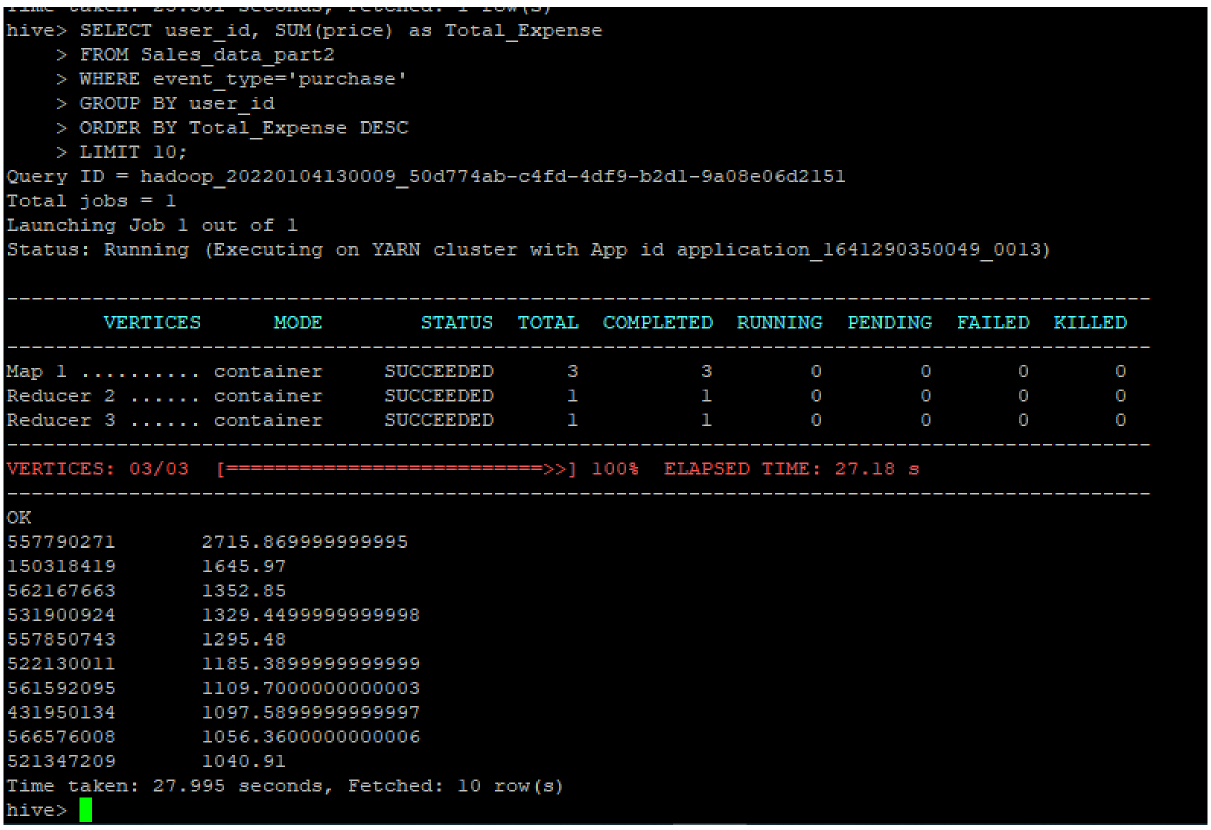
FROM Monthly_rev

WHERE (Nov_rev - Oct_rev)>0

ORDER BY Sales_diff;

OUTPUT: Total 161 rows returned (refer to the output screen)

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 Sales_data_part2
    > WHERE event_type='purchase'
    > GROUP BY user_id
    > ORDER BY Total_Expense DESC
    > LIMIT 10;
Query ID = hadoop_20220104130009_50d774ab-c4fd-4df9-b2d1-9a08e06d2151
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641290350049_0013)

----------------------------------------------------------------------------------------------
        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: 27.18 s
----------------------------------------------------------------------------------------------
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: 27.995 seconds, Fetched: 10 row(s)
hive>
```

SELECT user_id, SUM(price) as Total_Expense

FROM Sales_data_part2

WHERE event_type='purchase'

```
GROUP BY user_id

ORDER BY Total_Expense DESC

LIMIT 10;
```

OUTPUT:

User_id	Total_Amt_Spent
557790271	2715.87
150318415	1645.97
562167663	1382.85
531900924	1329.4499
557850743	1295.48
522130011	1329.4499
561592095	1109.70000
431950134	1097.58999
566576008	1056.36000
521347209	1040.91