E-Commerce Sales Hive Case Study

Contents

Problem Statement: 2

Objective 2

Steps Involved 2

Create EMR and Load the data to HDFS for Analysis 2

- Create Key Pair 2
- Create EMR Cluster 3
- Create S3 Bucket and upload the Data 4
- Enable SSH Connection and Connect Putty 5
- Setup the directory 5
- Starting Hive: 7
- Applying Optimization Techniques Columns Partitioning and Bucketing: 9

Questions and Queries to Answers: 14

Dropping the Database and Terminating the Cluster 27

- Drop Database: 27
- Terminate EMR: 27

Problem Statement:

A tech companies is exploring ways to improve their sales. They want to start by analysing customer behaviour and gaining insights about product trends.
Here, the role of big data analysts is among the most sought-after to gain the insights from abundance of data.

Objective

Through this assignment, as a big data analyst, we will extract data and gather insights from a real-life data set of an e-commerce company. We will analyse and gain insights about the clickstream data from a website so that we can extract insights about the customers behaviour.

Steps Involved

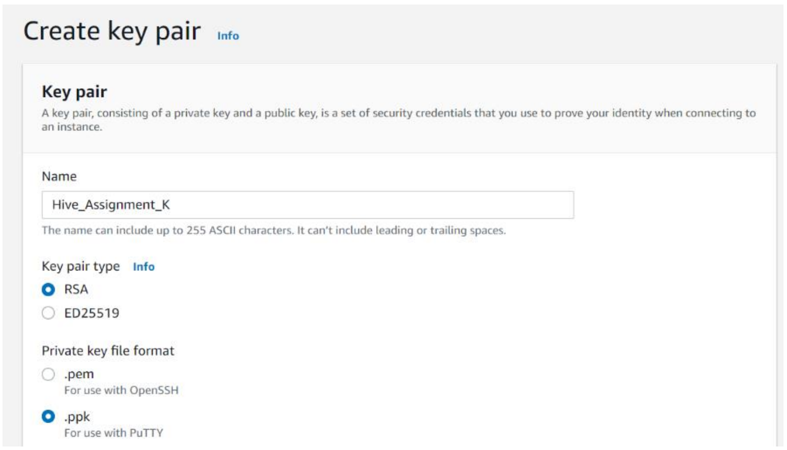
The steps involved in the entire process are as follows:

- ✓ Create the Key pair in AWS
- ✓ Create S3 bucket and copy the csv data files into s3 bucket
- ✓ Create the EMR cluster and launch EMR that utilizes the Hive services
- ✓ Move the data from the S3 bucket into the HDFS
- ✓ Creating the database and launching Hive queries on EMR cluster
- ✓ Creating the structure of database
- ✓ Use partitioning and bucketing to run your queries as efficiently as possible
- ✓ Run Hive queries to verify the analysis on given dataset.
- ✓ Drop the database, and
- ✓ Terminate the cluster

Create EMR and Load the data to HDFS for Analysis

Create Key Pair

- Open EC2 Management Console and select Create Key Pair. Enter a name for Key Pair, keeping the type as “RSA” amd file format as “.ppk” and Select Create Key Pair. The Key Pair is downloaded.



Key pair

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

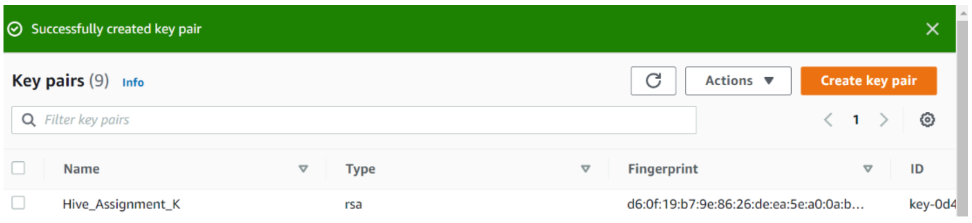
Name: Hive_Assignment_K

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

Key pair type: [✓ RSA] ED25519

Private key file format: .pem (For use with OpenSSH) [✓ .ppk (For use with PuTTY)]

- A new key pair “Hive_Assignment_K” has been created to be used for this case study.

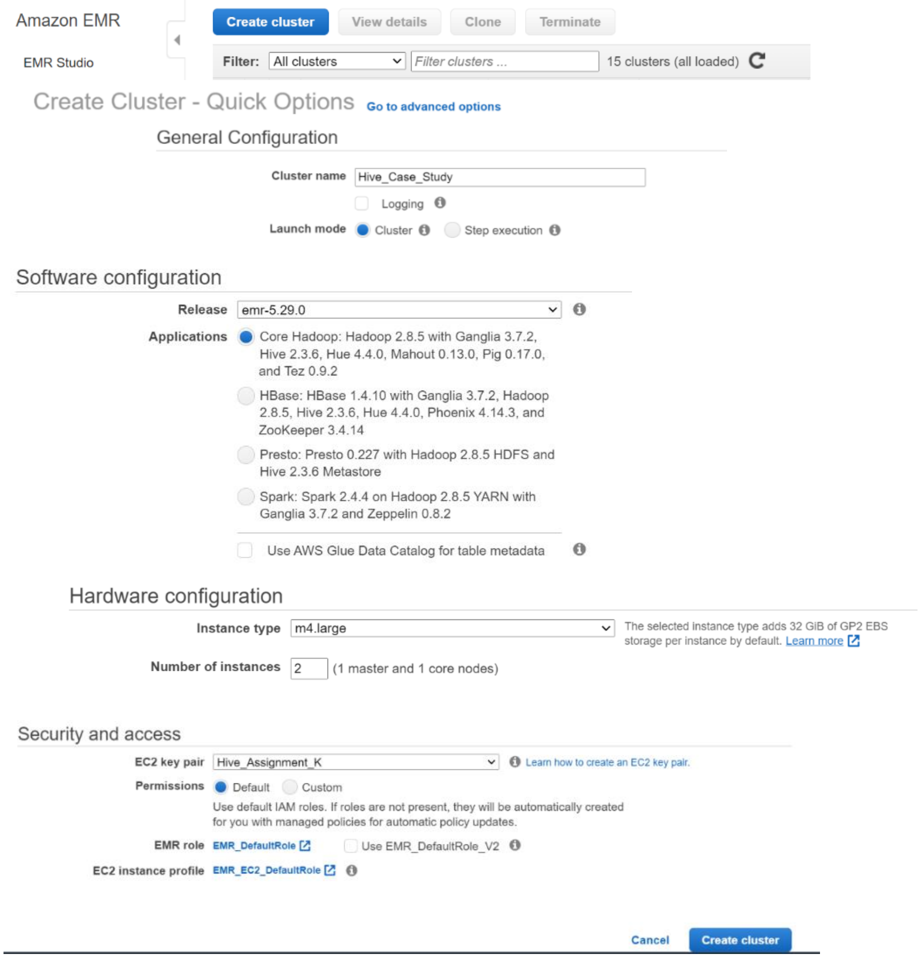

Successfully created key pair

Key pairs (9) Info

	Name	Type	Fingerprint	ID
☐	Hive_Assignment_K	rsa	d6:0f:19:b7:9e:86:26:de:ea:5e:a0:0a:b...	key-0d4

Create EMR Cluster

- Go to EMR > Select Create Cluster. Selected the General Configuration, Software Configuration, and Hardware Configuration as shown in below images. Select "Hive_Assignment_K" in EC2 key pair. Click "Create Cluster".



Create Cluster - Quick Options

General Configuration

Cluster name: Hive_Case_Study
☐ Logging
Launch mode: [✓ Cluster] Step execution

Software configuration

Release: emr-5.29.0
Applications:
[✓ Core Hadoop: Hadoop 2.8.5 with Ganglia 3.7.2, Hive 2.3.6, Hue 4.4.0, Mahout 0.13.0, Pig 0.17.0, and Tez 0.9.2]
HBase: HBase 1.4.10 with Ganglia 3.7.2, Hadoop 2.8.5, Hive 2.3.6, Hue 4.4.0, Phoenix 4.14.3, and ZooKeeper 3.4.14
Presto: Presto 0.227 with Hadoop 2.8.5 HDFS and Hive 2.3.6 Metastore
Spark: Spark 2.4.4 on Hadoop 2.8.5 YARN with Ganglia 3.7.2 and Zeppelin 0.8.2
☐ Use AWS Glue Data Catalog for table metadata

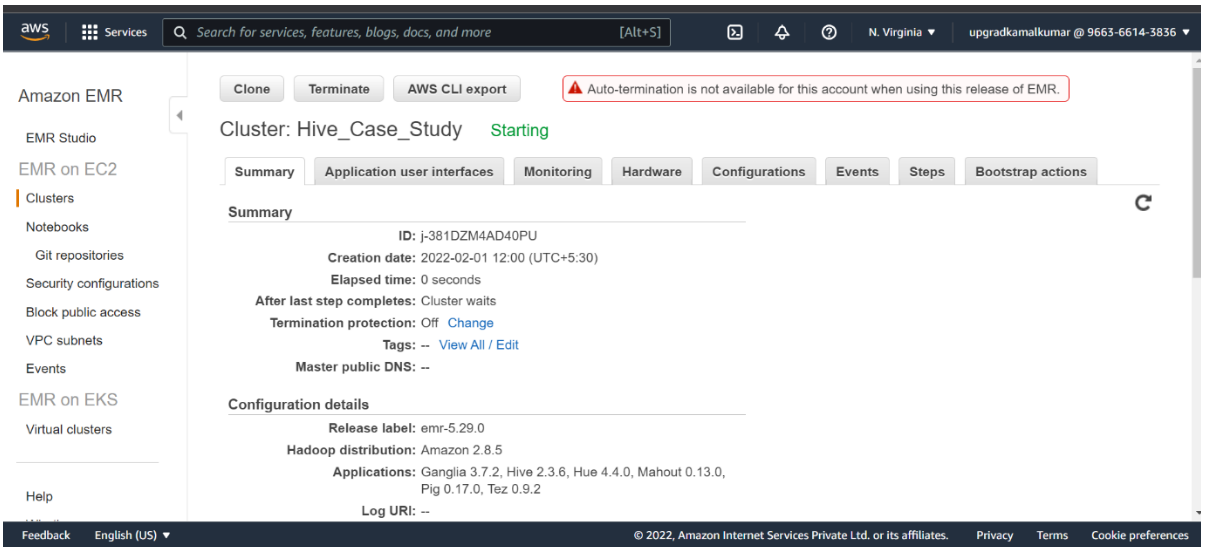
Hardware configuration

Instance type: m4.large — The selected instance type adds 32 GiB of GP2 EBS storage per instance by default. Learn more
Number of instances: 2 (1 master and 1 core nodes)

Security and access

EC2 key pair: Hive_Assignment_K — Learn how to create an EC2 key pair.
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

- The EMR Cluster creation and configuration begins in the backend. The Starting symbol beside Cluster name indicates it.



Clone | Terminate | AWS CLI export | Auto-termination is not available for this account when using this release of EMR.

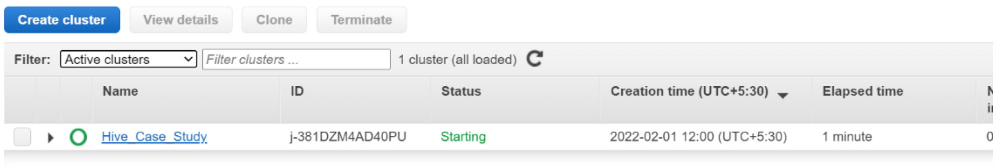
Cluster: Hive_Case_Study Starting

Summary

ID: j-381DZM4AD40PU
Creation date: 2022-02-01 12:00 (UTC+5:30)
Elapsed time: 0 seconds
After last step completes: Cluster waits
Termination protection: Off Change
Tags: -- View All / Edit
Master public DNS: --

Configuration details

Release label: emr-5.29.0
Hadoop distribution: Amazon 2.8.5
Applications: Ganglia 3.7.2, Hive 2.3.6, Hue 4.4.0, Mahout 0.13.0, Pig 0.17.0, Tez 0.9.2
Log URI: --

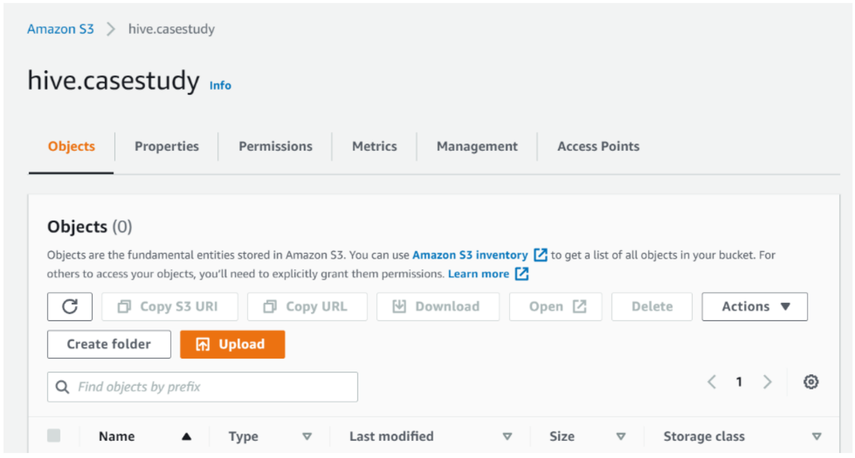


Filter: Active clusters | Filter clusters ... | 1 cluster (all loaded)

Name	ID	Status	Creation time (UTC+5:30)	Elapsed time
Hive_Case_Study	j-381DZM4AD40PU	Starting	2022-02-01 12:00 (UTC+5:30)	1 minute

Create S3 Bucket and upload the Data

- Go to Amazon S3 > Select Create Bucket > Enter Bucket Name "hive.casestudy" > Unselect "Block *all* public access" > Select Create Bucket.



Amazon S3 > hive.casestudy

hive.casestudy Info

Objects (0)

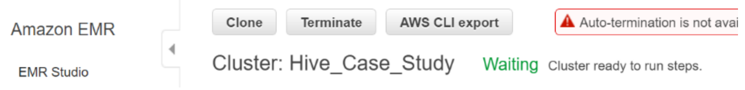
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

- Upload the data files "2019-Oct" and "2019-Nov" that are provided by UpGrad.

- After the files are uploaded, Copy S3 URL of both the uploaded files:
    - s3://hive.casestudy/2019-Oct.csv
    - s3://hive.casestudy/2019-Nov.csv

Enable SSH Connection and Connect Putty

- Go to Amazon EMR > Open the newly created active cluster after it is in waiting >



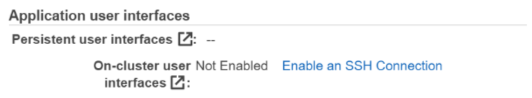
Amazon EMR

EMR Studio

Clone | Terminate | AWS CLI export | Auto-termination is not avai

Cluster: Hive_Case_Study Waiting Cluster ready to run steps.

- Scroll down to Application User > Select Enable an SSH Connection > Copy the Host Name Field "hadoop@ec2-54-196-3-223.compute-1.amazonaws.com"



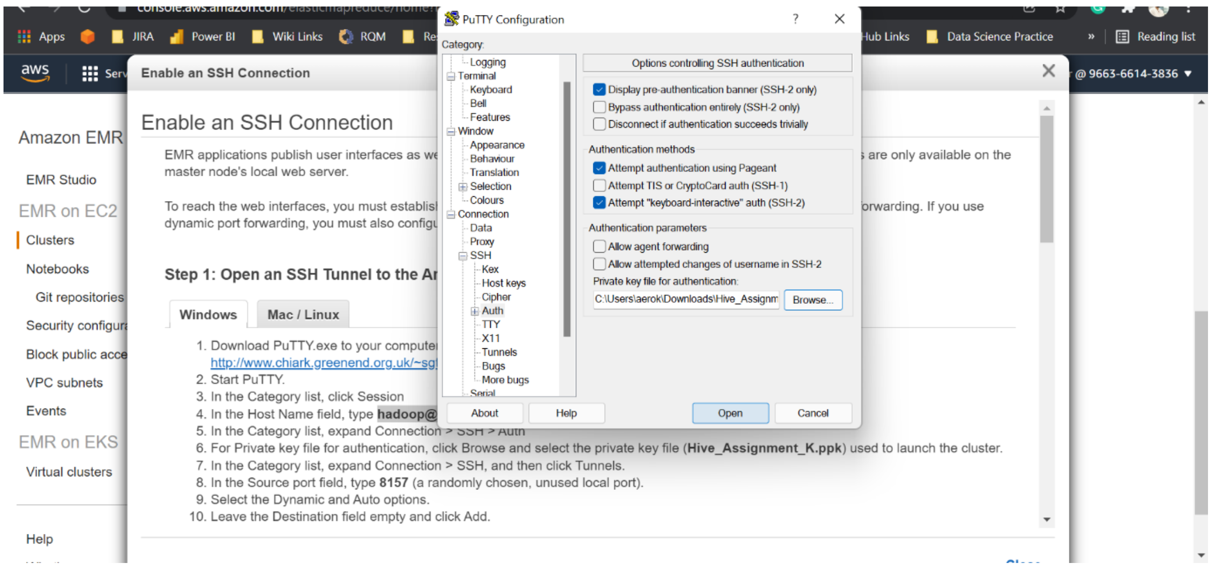
Application user interfaces

Persistent user interfaces: --

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

- Open Putty > Enter the host name hadoop@ec2-54-196-3-223.compute-1.amazonaws.com and select the key pair.

- Once cluster in running state we have to click on Master public DNS. 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.



Enable an SSH Connection

EMR applications publish user interfaces as we … are only available on the master node's local web server.

To reach the web interfaces, you must establis… dynamic port forwarding, you must also configu… orwarding. If you use

Step 1: Open an SSH Tunnel to the Ar

Windows | Mac / Linux

1. Download PuTTY.exe to your compute… http://www.chiark.greenend.org.uk/~sg…
2. Start PuTTY.
3. In the Category list, click Session
4. In the Host Name field, type **hadoop@**…
5. In the Category list, expand Connection > SSH > Auth
6. For Private key file for authentication, click Browse and select the private key file (**Hive_Assignment_K.ppk**) used to launch the cluster.
7. In the Category list, expand Connection > SSH, and then click Tunnels.
8. In the Source port field, type **8157** (a randomly chosen, unused local port).
9. Select the Dynamic and Auto options.
10. Leave the Destination field empty and click Add.

PuTTY Configuration

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)

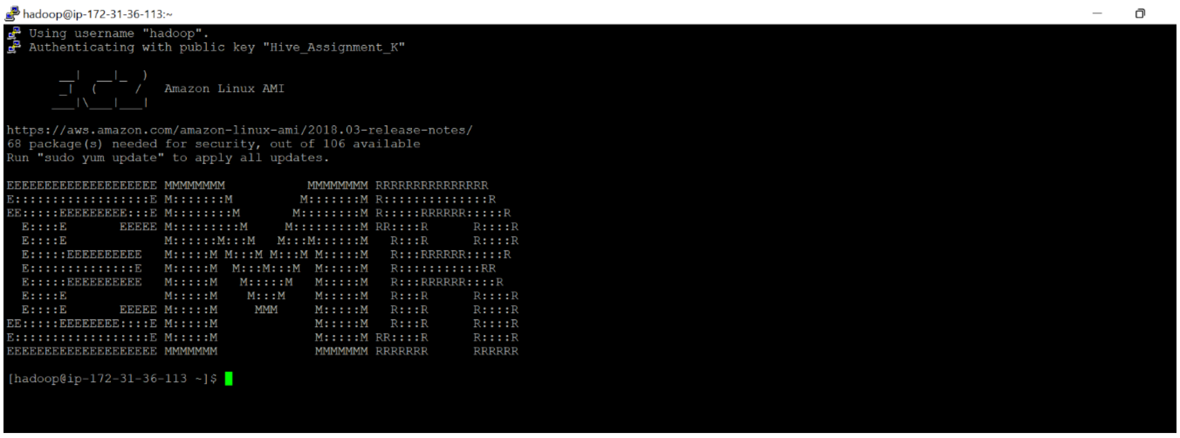
Authentication parameters

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

Private key file for authentication: C:\Users\aerok\Downloads\Hive_Assignm

Setup the directory

- EMR Opens, and below screen reflects that Connection to Hadoop is successful:



```
hadoop@ip-172-31-36-113:~
Using username "hadoop".
Authenticating with public key "Hive_Assignment_K"

       __|  __|_  )
       _|  (     /   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-172-31-36-113 ~]$
```

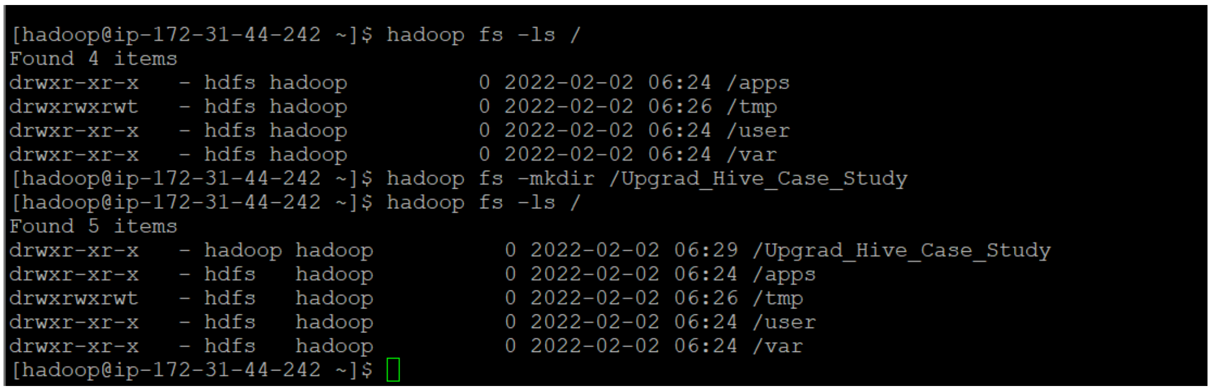
- Check the existing directories, and create a directory named ‘Upgrad_Hive_Case_Study’ in Hadoop. > ‘Upgrad_Hive_Case_Study’ is now created in Hadoop:

  Command:

  hadoop fs -ls /

  hadoop fs -mkdir /Upgrad_Hive_Case_Study

  hadoop fs -ls /



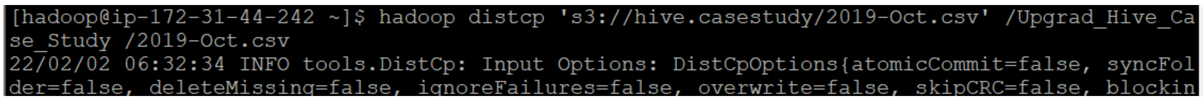
```
[hadoop@ip-172-31-44-242 ~]$ hadoop fs -ls /
Found 4 items
drwxr-xr-x   - hdfs hadoop          0 2022-02-02 06:24 /apps
drwxrwxrwt   - hdfs hadoop          0 2022-02-02 06:26 /tmp
drwxr-xr-x   - hdfs hadoop          0 2022-02-02 06:24 /user
drwxr-xr-x   - hdfs hadoop          0 2022-02-02 06:24 /var
[hadoop@ip-172-31-44-242 ~]$ hadoop fs -mkdir /Upgrad_Hive_Case_Study
[hadoop@ip-172-31-44-242 ~]$ hadoop fs -ls /
Found 5 items
drwxr-xr-x   - hadoop hadoop          0 2022-02-02 06:29 /Upgrad_Hive_Case_Study
drwxr-xr-x   - hdfs   hadoop          0 2022-02-02 06:24 /apps
drwxrwxrwt   - hdfs   hadoop          0 2022-02-02 06:26 /tmp
drwxr-xr-x   - hdfs   hadoop          0 2022-02-02 06:24 /user
drwxr-xr-x   - hdfs   hadoop          0 2022-02-02 06:24 /var
[hadoop@ip-172-31-44-242 ~]$
```

- Move the data from the s3 buckets to the HDFS using the distributed copy command. Loading the s3 public data set to created directory “Upgrad_Hive_Case_Study” in hadoop .

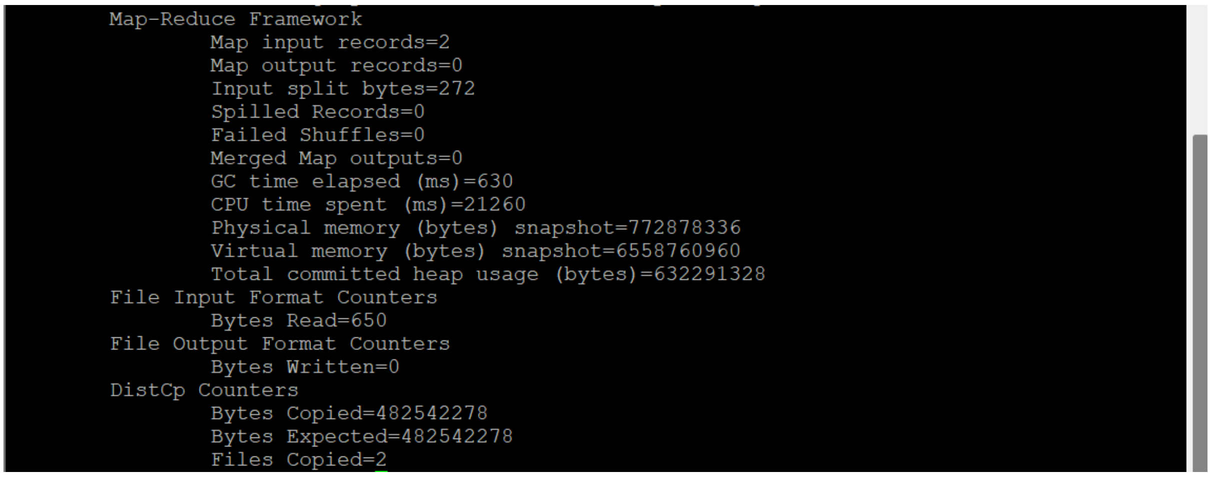
  Command:

  hadoop distcp 's3://hive.casestudy/2019-Oct.csv' /Upgrad_Hive_Case_Study /2019-Oct.csv

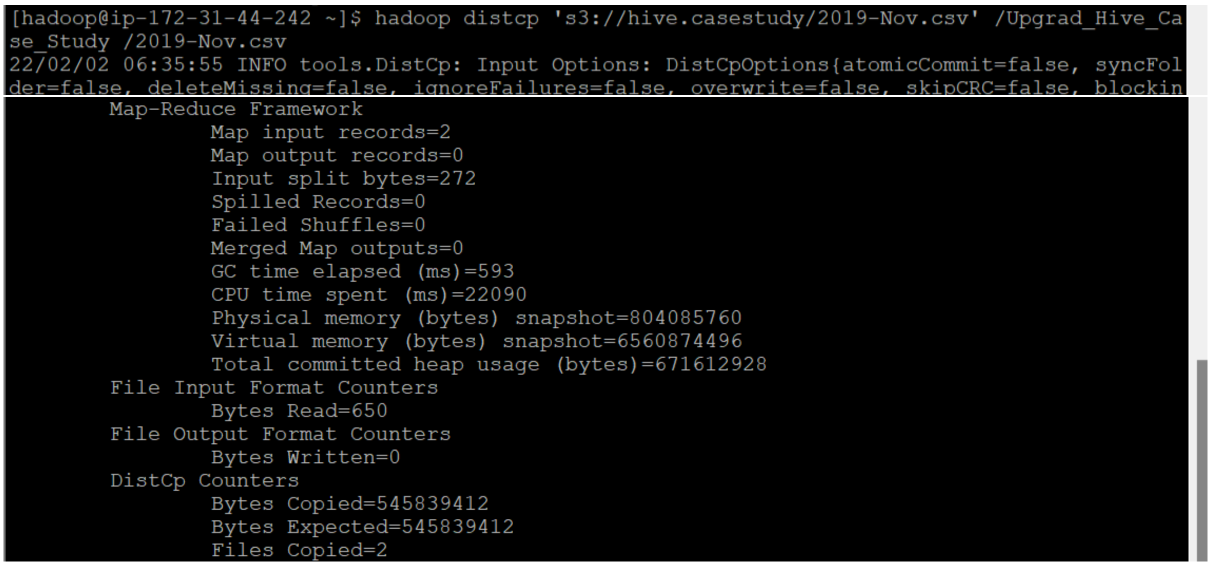
  hadoop distcp 's3://hive.casestudy/2019-Nov.csv' /Upgrad_Hive_Case_Study /2019-Nov.csv



```
[hadoop@ip-172-31-44-242 ~]$ hadoop distcp 's3://hive.casestudy/2019-Oct.csv' /Upgrad_Hive_Ca
se_Study /2019-Oct.csv
22/02/02 06:32:34 INFO tools.DistCp: Input Options: DistCpOptions{atomicCommit=false, syncFol
der=false, deleteMissing=false, ignoreFailures=false, overwrite=false, skipCRC=false, blockin
```



```
        Map-Reduce Framework
                Map input records=2
                Map output records=0
                Input split bytes=272
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=630
                CPU time spent (ms)=21260
                Physical memory (bytes) snapshot=772878336
                Virtual memory (bytes) snapshot=6558760960
                Total committed heap usage (bytes)=632291328
        File Input Format Counters
                Bytes Read=650
        File Output Format Counters
                Bytes Written=0
        DistCp Counters
                Bytes Copied=482542278
                Bytes Expected=482542278
                Files Copied=2
```



```
[hadoop@ip-172-31-44-242 ~]$ hadoop distcp 's3://hive.casestudy/2019-Nov.csv' /Upgrad_Hive_Ca
se_Study /2019-Nov.csv
22/02/02 06:35:55 INFO tools.DistCp: Input Options: DistCpOptions{atomicCommit=false, syncFol
der=false, deleteMissing=false, ignoreFailures=false, overwrite=false, skipCRC=false, blockin
        Map-Reduce Framework
                Map input records=2
                Map output records=0
                Input split bytes=272
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=593
                CPU time spent (ms)=22090
                Physical memory (bytes) snapshot=804085760
                Virtual memory (bytes) snapshot=6560874496
                Total committed heap usage (bytes)=671612928
        File Input Format Counters
                Bytes Read=650
        File Output Format Counters
                Bytes Written=0
        DistCp Counters
                Bytes Copied=545839412
                Bytes Expected=545839412
                Files Copied=2
```

Starting Hive:

- It's time to Set the data in Hive,
- Launch Hive > show databases > CREATE a DATABASE "hive_data" > Check if Database is created.

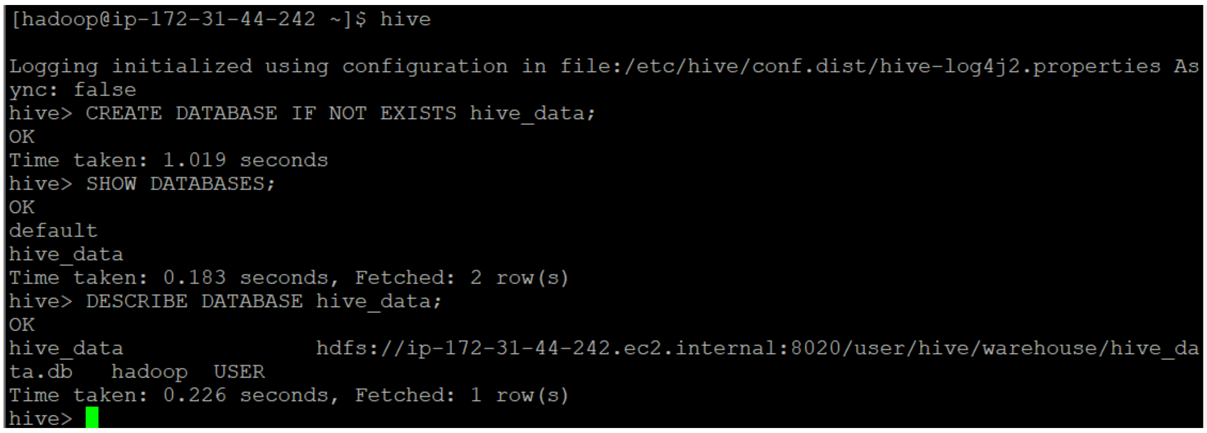
  Codes:

  Hive

  CREATE DATABASE IF NOT EXISTS hive_data;

  SHOW DATABASES;

  DESCRIBE DATABASE hive_data;



```
[hadoop@ip-172-31-44-242 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties As
ync: false
hive> CREATE DATABASE IF NOT EXISTS hive_data;
OK
Time taken: 1.019 seconds
hive> SHOW DATABASES;
OK
default
hive_data
Time taken: 0.183 seconds, Fetched: 2 row(s)
hive> DESCRIBE DATABASE hive_data;
OK
hive_data               hdfs://ip-172-31-44-242.ec2.internal:8020/user/hive/warehouse/hive_da
ta.db   hadoop  USER
Time taken: 0.226 seconds, Fetched: 1 row(s)
hive>
```

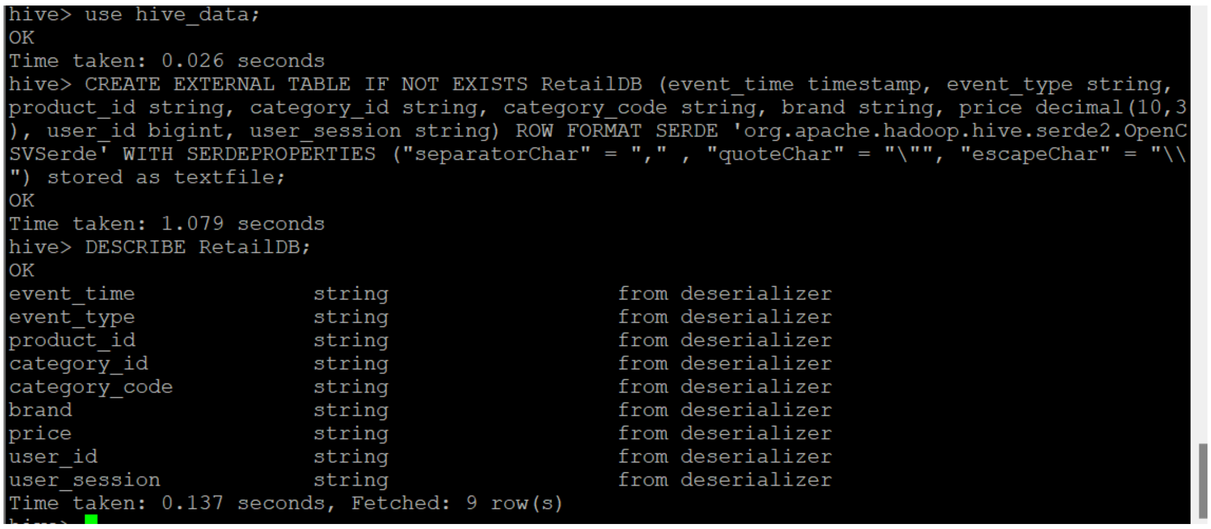
Use the database and create an external table “RetailDB”:

Query:

use hive_data;

CREATE EXTERNAL TABLE IF NOT EXISTS RetailDB (event_time timestamp, event_type string, product_id string, category_id string, category_code string, brand string, price decimal(10,3), user_id bigint, user_session string) ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' WITH SERDEPROPERTIES ("separator-Char" = "," , "quoteChar" = "\"", "escapeChar" = "\\") stored as textfile

DESCRIBE RetailDB;



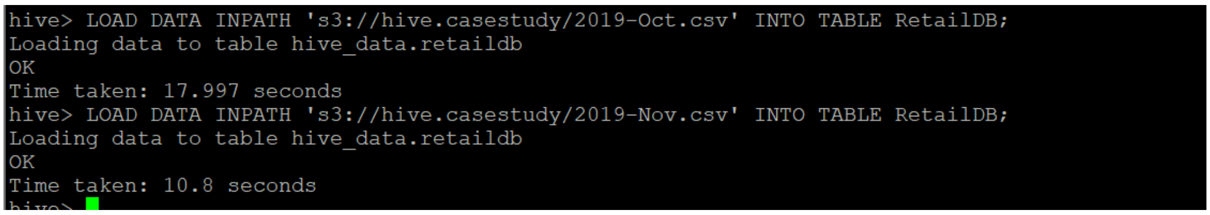
```
hive> use hive_data;
OK
Time taken: 0.026 seconds
hive> CREATE EXTERNAL TABLE IF NOT EXISTS RetailDB (event_time timestamp, event_type string,
product_id string, category_id string, category_code string, brand string, price decimal(10,3
), user_id bigint, user_session string) ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenC
SVSerde' WITH SERDEPROPERTIES ("separatorChar" = "," , "quoteChar" = "\"", "escapeChar" = "\\
") stored as textfile;
OK
Time taken: 1.079 seconds
hive> DESCRIBE RetailDB;
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: 0.137 seconds, Fetched: 9 row(s)
```

- Load the input Data into “RetailDB” table:

Query:

LOAD DATA INPATH ‘s3://hive.casestudy/2019-Oct.csv’ INTO TABLE RetailDB;

LOAD DATA INPATH ‘s3://hive.casestudy/2019-Nov.csv’ INTO TABLE RetailDB;



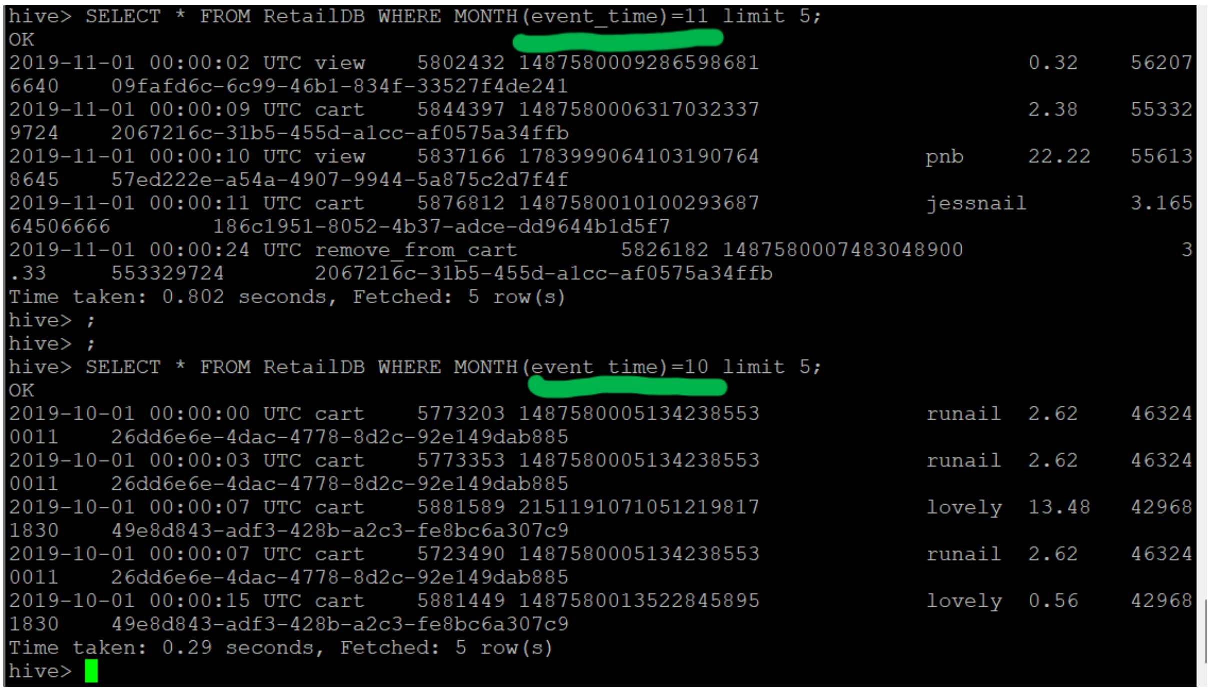
```
hive> LOAD DATA INPATH 's3://hive.casestudy/2019-Oct.csv' INTO TABLE RetailDB;
Loading data to table hive_data.retaildb
OK
Time taken: 17.997 seconds
hive> LOAD DATA INPATH 's3://hive.casestudy/2019-Nov.csv' INTO TABLE RetailDB;
Loading data to table hive_data.retaildb
OK
Time taken: 10.8 seconds
hive>
```

- Test the loaded data and check if data of November and October are loaded:

Query:

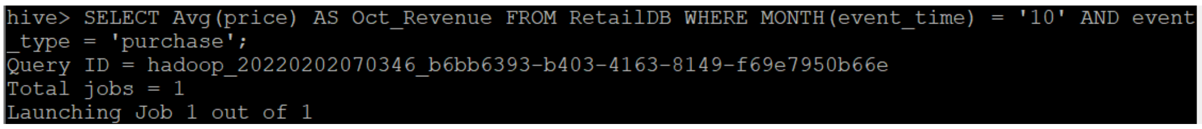
SELECT * FROM RetailDB WHERE MONTH(event_time)=11 limit 5;

SELECT * FROM RetailDB WHERE MONTH(event_time)=10 limit 5;

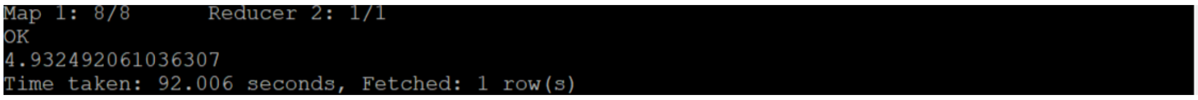


```
hive> SELECT * FROM RetailDB WHERE MONTH(event_time)=11 limit 5;
OK
2019-11-01 00:00:02 UTC view    5802432 1487580009286598681             0.32    56207
6640    09fafd6c-6c99-46b1-834f-33527f4de241
2019-11-01 00:00:09 UTC cart    5844397 1487580006317032337             2.38    55332
9724    2067216c-31b5-455d-a1cc-af0575a34ffb
2019-11-01 00:00:10 UTC view    5837166 1783999064103190764     pnb     22.22   55613
8645    57ed222e-a54a-4907-9944-5a875c2d7f4f
2019-11-01 00:00:11 UTC cart    5876812 1487580010100293687     jessnail        3.165
64506666        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
Time taken: 0.802 seconds, Fetched: 5 row(s)
hive> ;
hive> ;
hive> SELECT * FROM RetailDB WHERE MONTH(event_time)=10 limit 5;
OK
2019-10-01 00:00:00 UTC cart    5773203 1487580005134238553     runail  2.62    46324
0011    26dd6e6e-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:03 UTC cart    5773353 1487580005134238553     runail  2.62    46324
0011    26dd6e6e-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:07 UTC cart    5881589 2151191071051219817     lovely  13.48   42968
1830    49e8d843-adf3-428b-a2c3-fe8bc6a307c9
2019-10-01 00:00:07 UTC cart    5723490 1487580005134238553     runail  2.62    46324
0011    26dd6e6e-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:15 UTC cart    5881449 1487580013522845895     lovely  0.56    42968
1830    49e8d843-adf3-428b-a2c3-fe8bc6a307c9
Time taken: 0.29 seconds, Fetched: 5 row(s)
hive>
```

Further, randomly check if query works for aggregate functions:



```
hive> SELECT Avg(price) AS Oct_Revenue FROM RetailDB WHERE MONTH(event_time) = '10' AND event
_type = 'purchase';
Query ID = hadoop_20220202070346_b6bb6393-b403-4163-8149-f69e7950b66e
Total jobs = 1
Launching Job 1 out of 1
```

➢ 

```
Map 1: 8/8      Reducer 2: 1/1
OK
4.932492061036307
Time taken: 92.006 seconds, Fetched: 1 row(s)
```

Applying Optimization Techniques Columns Partitioning and Bucketing:

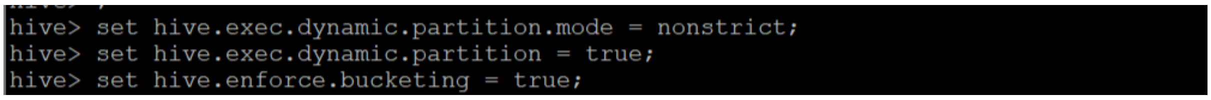
- Start with enabling the dynamic partitioning and bucketing:

Code:

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

set hive.exec.dynamic.partition = true;

set hive.enforce.bucketing = true;



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

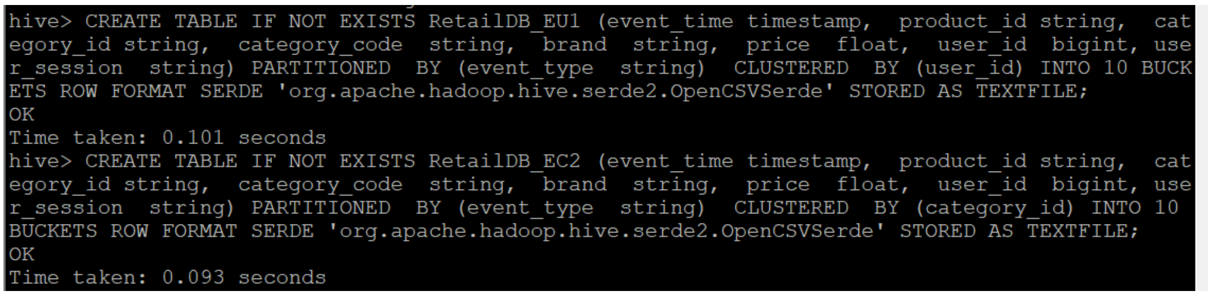
Let’s create and test two different combinations of partitioning and bucketing:

- Create a Table “RetailDB_EU1”, PARTITIONED BY(event_type string) CLUSTERED BY (user _id) INTO 10 buckets
- Create a Table “RetailDB_EC2”, PARTITIONED BY(event_type string) CLUSTERED BY (category _id) INTO 10 buckets

Query:

CREATE TABLE IF NOT EXISTS RetailDB_EU1 (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 (user_id) INTO 10 BUCKETS ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS TEXTFILE;

CREATE TABLE IF NOT EXISTS RetailDB_EC2 (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 10 BUCKETS ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS TEXTFILE;



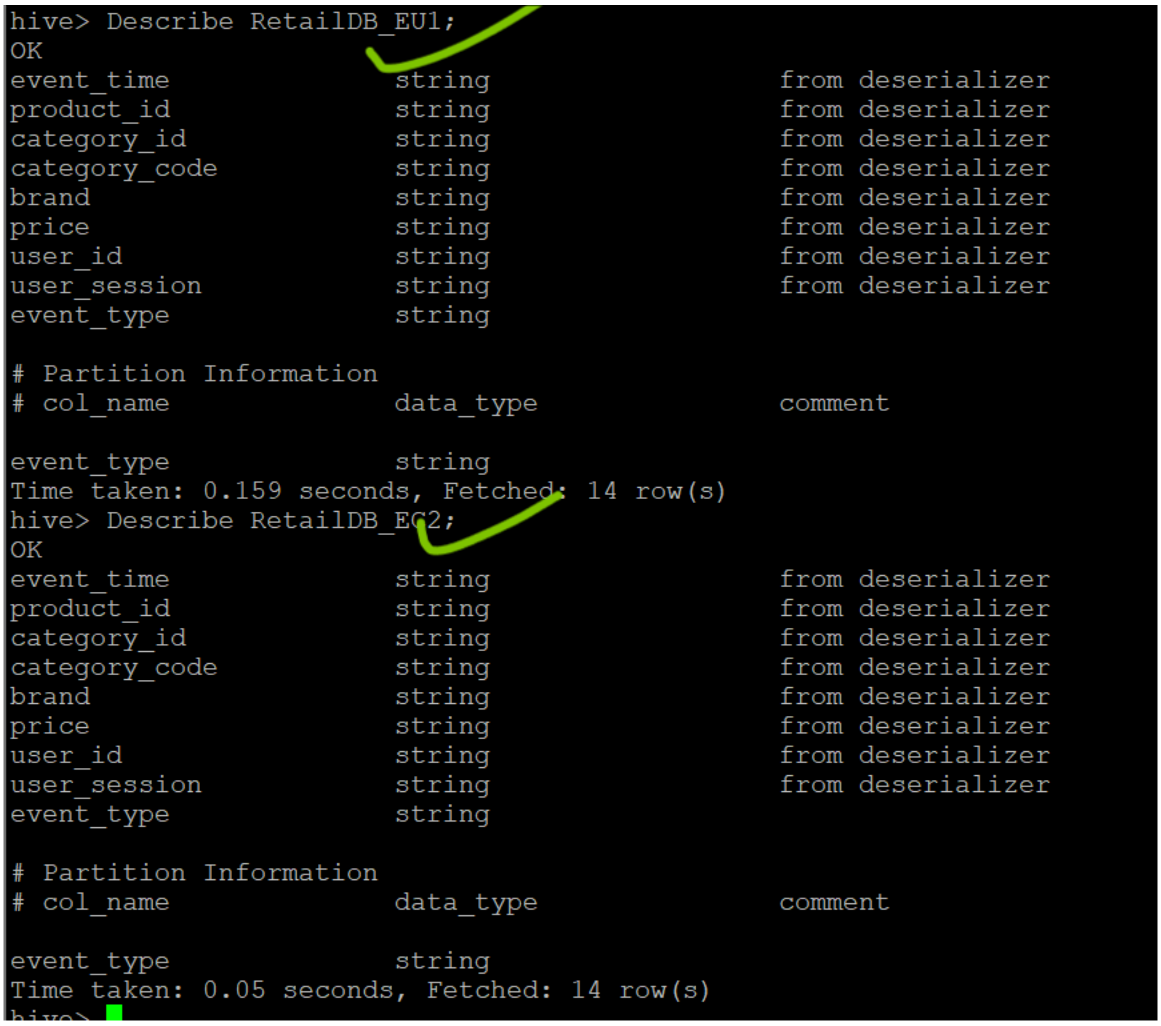
```
hive> CREATE TABLE IF NOT EXISTS RetailDB_EU1 (event_time timestamp,  product_id string,  cat
egory_id string,  category_code  string,  brand  string,  price  float,  user_id  bigint, use
r_session  string) PARTITIONED  BY (event_type  string)  CLUSTERED  BY (user_id) INTO 10 BUCK
ETS ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS TEXTFILE;
OK
Time taken: 0.101 seconds
hive> CREATE TABLE IF NOT EXISTS RetailDB_EC2 (event_time timestamp,  product_id string,  cat
egory_id string,  category_code  string,  brand  string,  price  float,  user_id  bigint, use
r_session  string) PARTITIONED  BY (event_type  string)  CLUSTERED  BY (category_id) INTO 10
BUCKETS ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS TEXTFILE;
OK
Time taken: 0.093 seconds
```

Check Properties of both Tables:

Query:

Describe RetailDB_EU1;

Describe RetailDB_EC2;



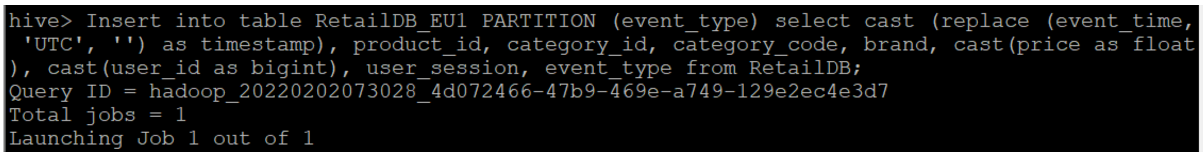
```
hive> Describe RetailDB_EU1;
OK
event_time              string                  from deserializer
product_id              string                  from deserializer
category_id             string                  from deserializer
category_code           string                  from deserializer
brand                   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: 0.159 seconds, Fetched: 14 row(s)
hive> Describe RetailDB_EC2;
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: 0.05 seconds, Fetched: 14 row(s)
hive>
```

- Load the Data into the tables:

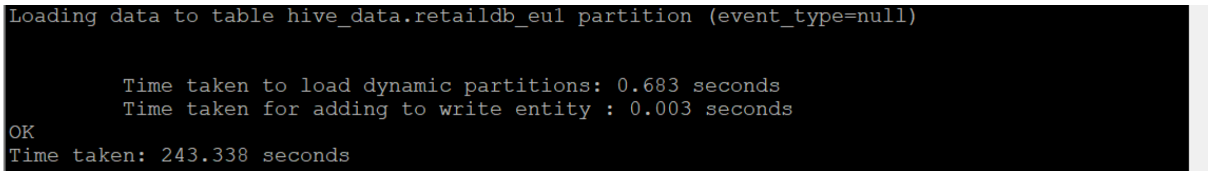
Query:

Insert into table RetailDB_EU1 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 RetailDB;

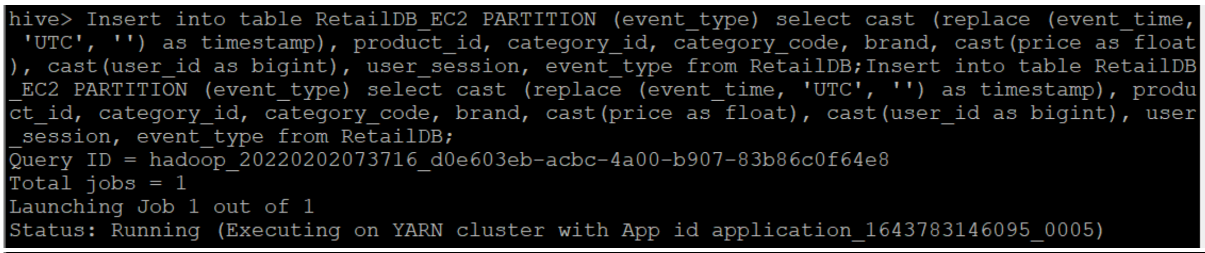
Insert into table RetailDB_EC2 PARTITION (event_type) select cast (replace (event_time, 'UTC', '') as timestamp), product_id, category_id, category_code, brand, cast(price as float), cast(user_id as bigint), user_session, event_type from RetailDB;



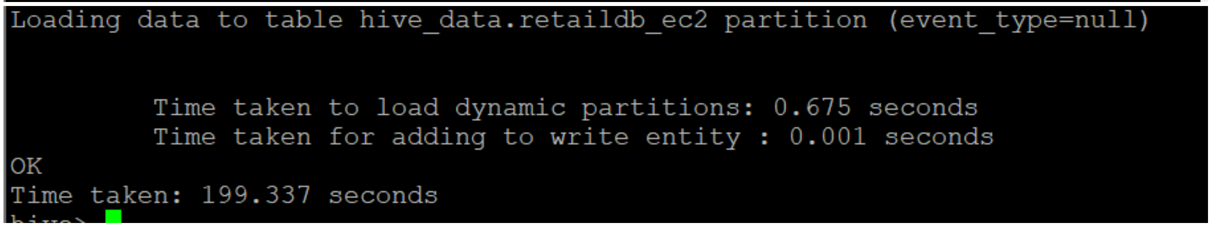
```
hive> Insert into table RetailDB_EU1 PARTITION (event_type) select cast (replace (event_time,
 'UTC', '') as timestamp), product_id, category_id, category_code, brand, cast(price as float
), cast(user_id as bigint), user_session, event_type from RetailDB;
Query ID = hadoop_20220202073028_4d072466-47b9-469e-a749-129e2ec4e3d7
Total jobs = 1
Launching Job 1 out of 1
```



```
Loading data to table hive_data.retaildb_eu1 partition (event_type=null)


         Time taken to load dynamic partitions: 0.683 seconds
         Time taken for adding to write entity : 0.003 seconds
OK
Time taken: 243.338 seconds
```



```
hive> Insert into table RetailDB_EC2 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 RetailDB;Insert into table RetailDB
_EC2 PARTITION (event_type) select cast (replace (event_time, 'UTC', '') as timestamp), produ
ct_id, category_id, category_code, brand, cast(price as float), cast(user_id as bigint), user
_session, event_type from RetailDB;
Query ID = hadoop_20220202073716_d0e603eb-acbc-4a00-b907-83b86c0f64e8
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1643783146095_0005)
```

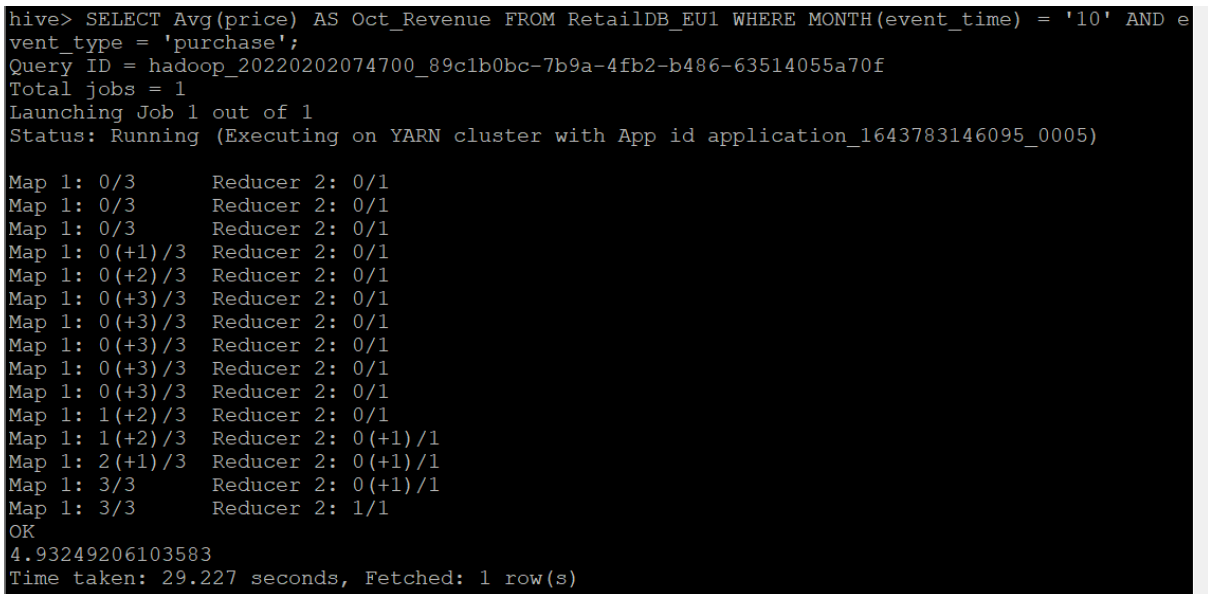


```
Loading data to table hive_data.retaildb_ec2 partition (event_type=null)


         Time taken to load dynamic partitions: 0.675 seconds
         Time taken for adding to write entity : 0.001 seconds
OK
Time taken: 199.337 seconds
```

Further, we did run some basic query in both and found the second option to return the output in lesser time. Hence, we will continue with only second option.

Query:

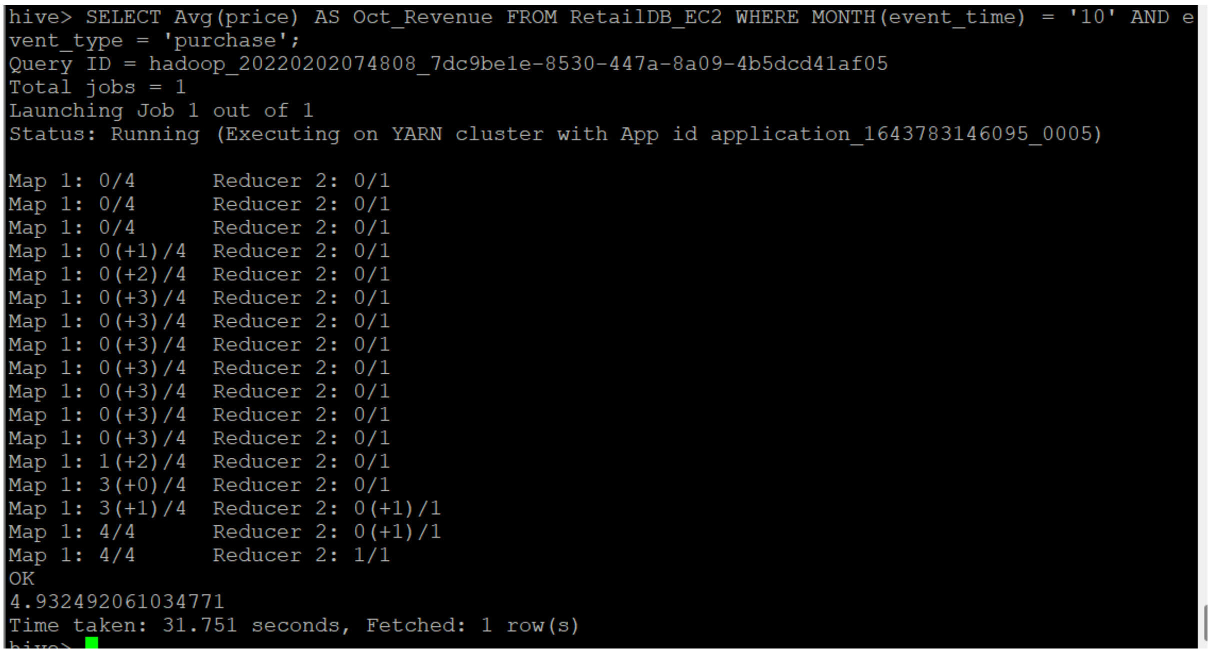
SELECT Avg(price) AS Oct_Revenue FROM RetailDB_EU1 WHERE MONTH(event_time) = '10' AND event_type = 'purchase';



```
hive> SELECT Avg(price) AS Oct_Revenue FROM RetailDB_EU1 WHERE MONTH(event_time) = '10' AND e
vent_type = 'purchase';
Query ID = hadoop_20220202074700_89c1b0bc-7b9a-4fb2-b486-63514055a70f
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1643783146095_0005)

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(+1)/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: 1(+2)/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: 0(+1)/1
Map 1: 3/3      Reducer 2: 1/1
OK
4.93249206103583
Time taken: 29.227 seconds, Fetched: 1 row(s)
```

Query:

SELECT Avg(price) AS Oct_Revenue FROM RetailDB_EC2 WHERE MONTH(event_time) = '10' AND event_type = 'purchase';



```
hive> SELECT Avg(price) AS Oct_Revenue FROM RetailDB_EC2 WHERE MONTH(event_time) = '10' AND e
vent_type = 'purchase';
Query ID = hadoop_20220202074808_7dc9be1e-8530-447a-8a09-4b5dcd41af05
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1643783146095_0005)

Map 1: 0/4      Reducer 2: 0/1
Map 1: 0/4      Reducer 2: 0/1
Map 1: 0/4      Reducer 2: 0/1
Map 1: 0(+1)/4  Reducer 2: 0/1
Map 1: 0(+2)/4  Reducer 2: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1
Map 1: 1(+2)/4  Reducer 2: 0/1
Map 1: 3(+0)/4  Reducer 2: 0/1
Map 1: 3(+1)/4  Reducer 2: 0(+1)/1
Map 1: 4/4      Reducer 2: 0(+1)/1
Map 1: 4/4      Reducer 2: 1/1
OK
4.932492061034771
Time taken: 31.751 seconds, Fetched: 1 row(s)
hive>
```

So, we see, that time taken by RetailDB_EU1 is slightly lesser than RetailDB_EC2.

Hence, we will go ahead with RetailDB_EU1 table for our further analysis:

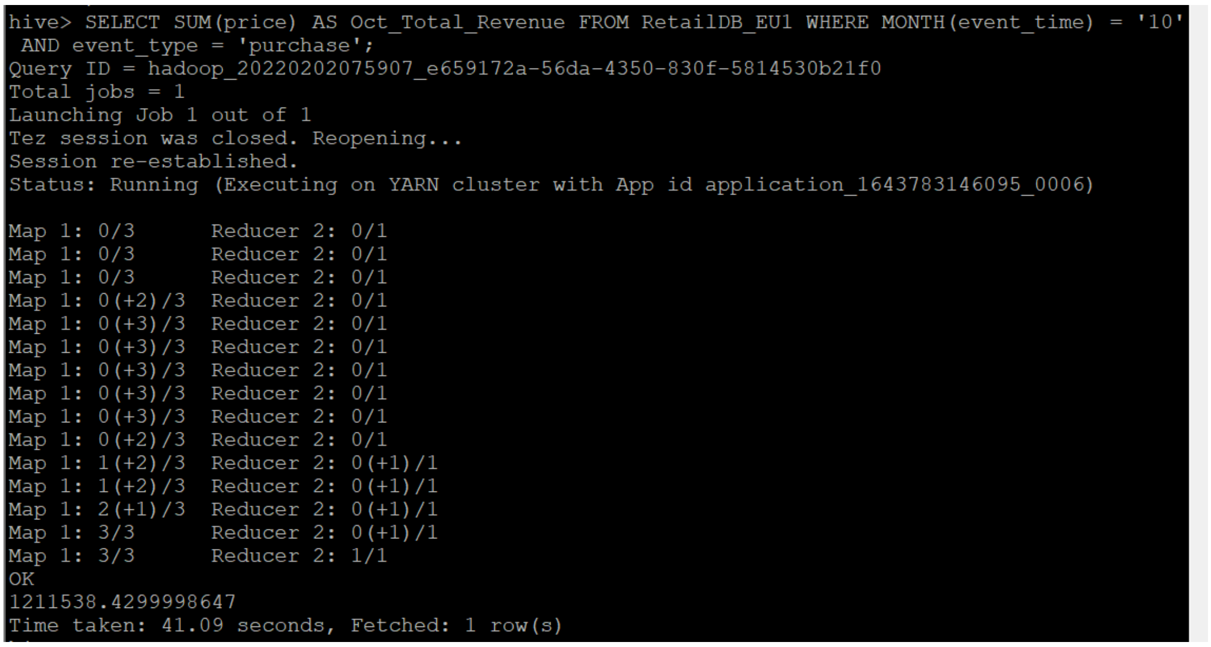
Questions and Queries to Answers:

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

Query:

```
SELECT sum(price)

FROM RetailDB_EU1

WHERE month(event_time)=10 and event_type='purchase';
```

Answer:



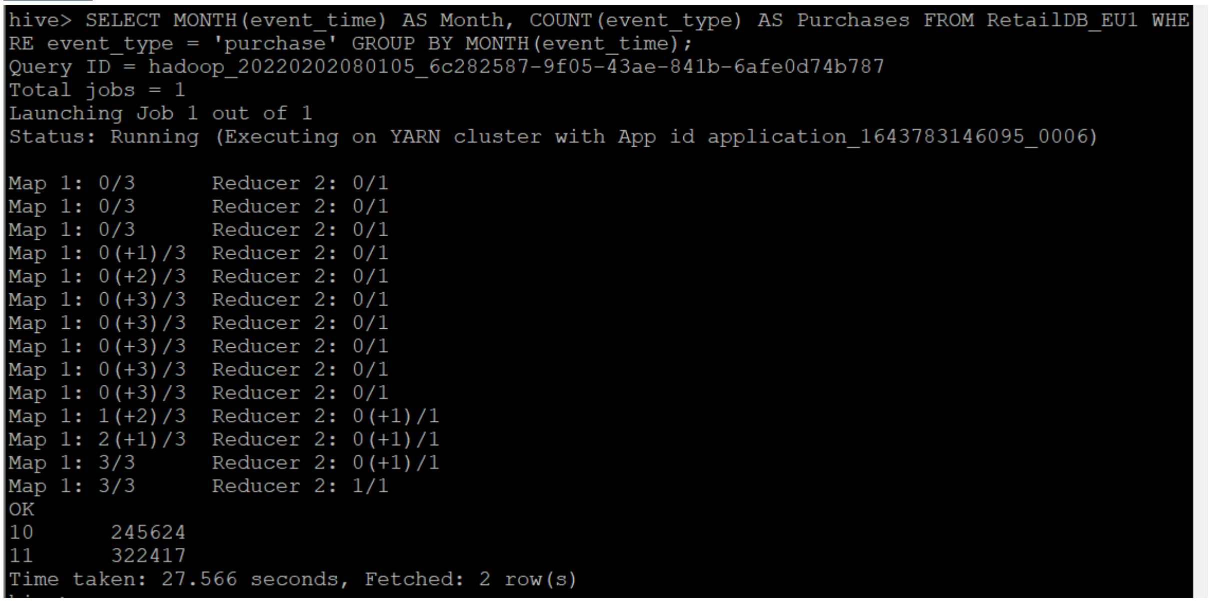
```
hive> SELECT SUM(price) AS Oct_Total_Revenue FROM RetailDB_EU1 WHERE MONTH(event_time) = '10'
 AND event_type = 'purchase';
Query ID = hadoop_20220202075907_e659172a-56da-4350-830f-5814530b21f0
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1643783146095_0006)

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(+2)/3  Reducer 2: 0/1
Map 1: 1(+2)/3  Reducer 2: 0(+1)/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: 0(+1)/1
Map 1: 3/3      Reducer 2: 1/1
OK
1211538.4299998647
Time taken: 41.09 seconds, Fetched: 1 row(s)
```

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

Query:

```
SELECT MONTH(event_time) AS Month, COUNT(event_type) AS Purchases
FROM RetailDB_EU1
WHERE event_type = 'purchase'
GROUP BY MONTH(event_time);
```

Answer:



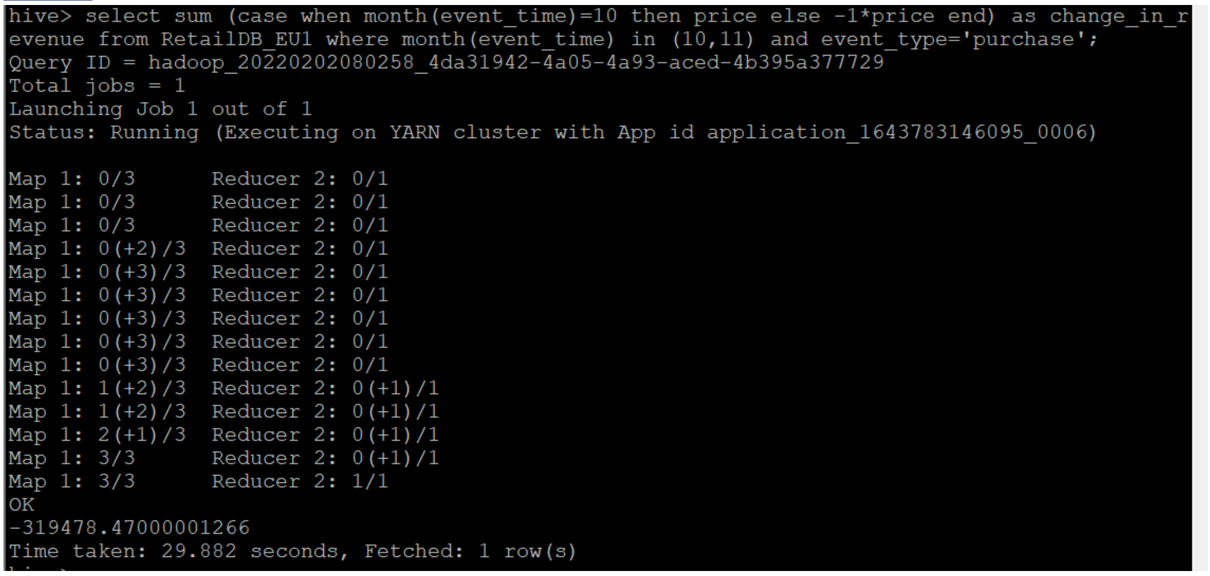
```
hive> SELECT MONTH(event_time) AS Month, COUNT(event_type) AS Purchases FROM RetailDB_EU1 WHE
RE event_type = 'purchase' GROUP BY MONTH(event_time);
Query ID = hadoop_20220202080105_6c282587-9f05-43ae-841b-6afe0d74b787
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1643783146095_0006)

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(+1)/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: 1(+2)/3  Reducer 2: 0(+1)/1
Map 1: 2(+1)/3  Reducer 2: 0(+1)/1
Map 1: 3/3      Reducer 2: 0(+1)/1
Map 1: 3/3      Reducer 2: 1/1
OK
10      245624
11      322417
Time taken: 27.566 seconds, Fetched: 2 row(s)
```

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

Query:

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

FROM RetailDB_EU1

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

Answer:



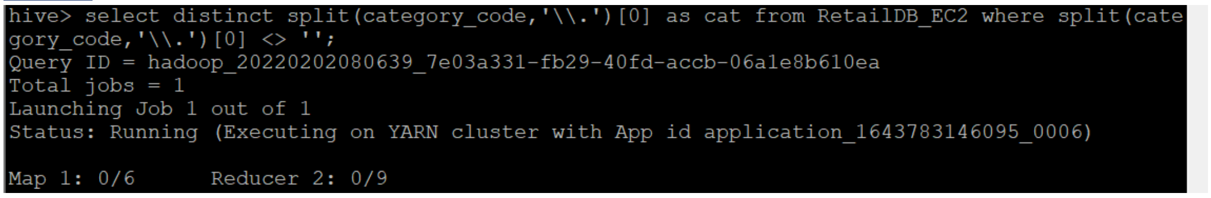
```
hive> select sum (case when month(event_time)=10 then price else -1*price end) as change_in_r
evenue from RetailDB_EU1 where month(event_time) in (10,11) and event_type='purchase';
Query ID = hadoop_20220202080258_4da31942-4a05-4a93-aced-4b395a377729
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1643783146095_0006)

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: 1(+2)/3  Reducer 2: 0(+1)/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: 0(+1)/1
Map 1: 3/3      Reducer 2: 1/1
OK
-319478.47000001266
Time taken: 29.882 seconds, Fetched: 1 row(s)
```

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

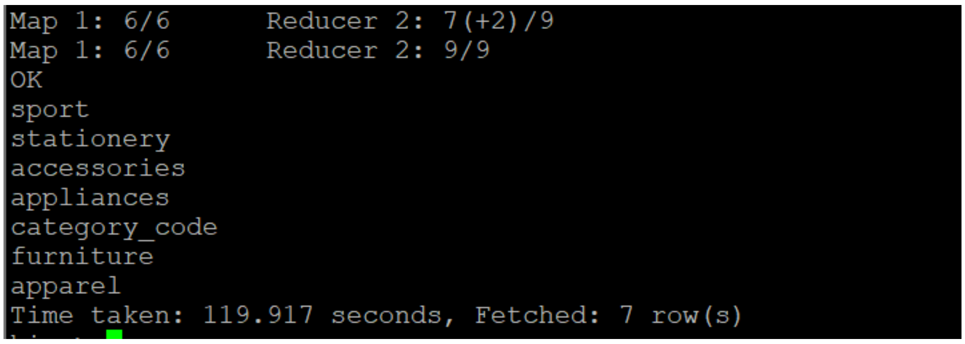
Query:

```
SELECT distinct split(category_code,’\\.’)[0] as cat

FROM RetailDB_EU1

WHERE split(category_code,’\\.’)[0] <> ‘’;
```

Answer:



```
hive> select distinct split(category_code,'\\.')[0] as cat from RetailDB_EC2 where split(cate
gory_code,'\\.')[0] <> '';
Query ID = hadoop_20220202080639_7e03a331-fb29-40fd-accb-06a1e8b610ea
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1643783146095_0006)

Map 1: 0/6      Reducer 2: 0/9
```



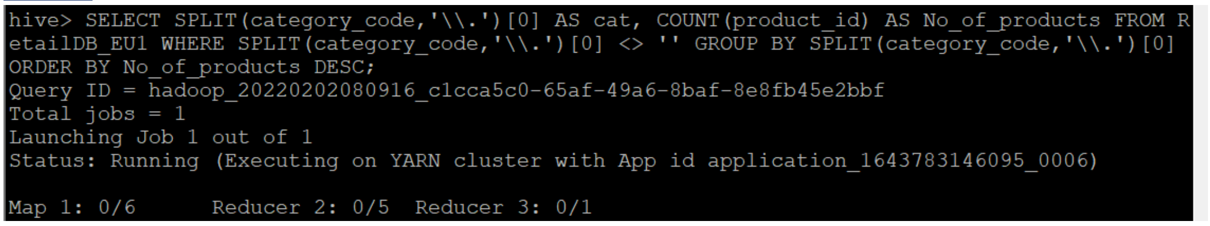
```
Map 1: 6/6      Reducer 2: 7(+2)/9
Map 1: 6/6      Reducer 2: 9/9
OK
sport
stationery
accessories
appliances
category_code
furniture
apparel
Time taken: 119.917 seconds, Fetched: 7 row(s)
```

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

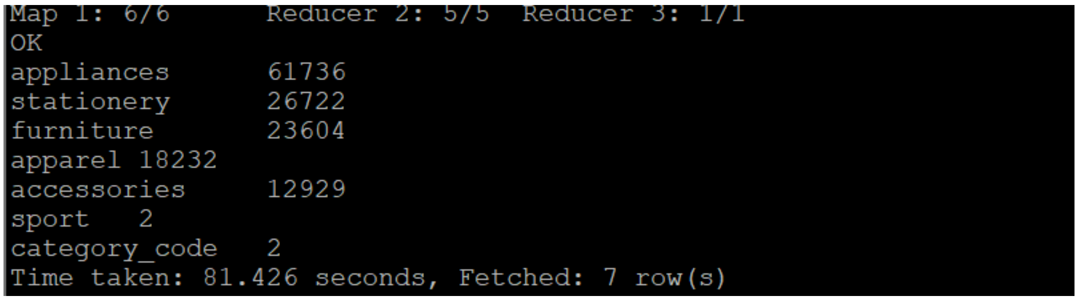
Query:

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

FROM RetailDB_EU1

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

GROUP BY SPLIT(category_code,'\\.')[0]

ORDER BY No_of_products DESC;
```

Answer:



```
hive> SELECT SPLIT(category_code,'\\.')[0] AS cat, COUNT(product_id) AS No_of_products FROM R
etailDB_EU1 WHERE SPLIT(category_code,'\\.')[0] <> '' GROUP BY SPLIT(category_code,'\\.')[0]
ORDER BY No_of_products DESC;
Query ID = hadoop_20220202080916_c1cca5c0-65af-49a6-8baf-8e8fb45e2bbf
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1643783146095_0006)

Map 1: 0/6      Reducer 2: 0/5  Reducer 3: 0/1
```



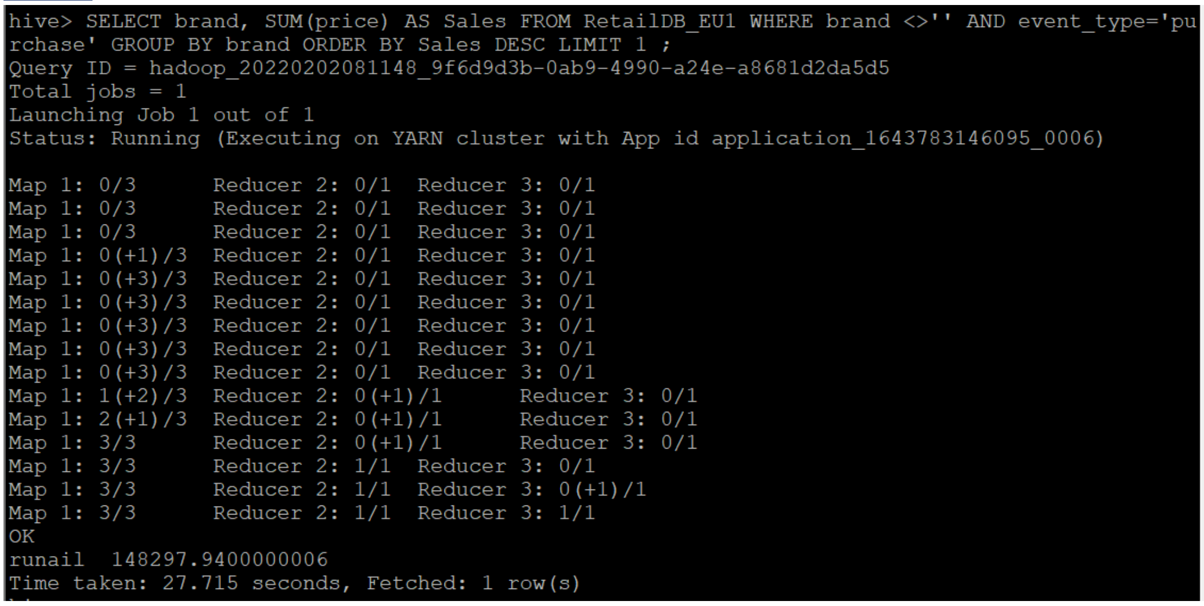
```
Map 1: 6/6      Reducer 2: 5/5  Reducer 3: 1/1
OK
appliances      61736
stationery      26722
furniture       23604
apparel 18232
accessories     12929
sport   2
category_code   2
Time taken: 81.426 seconds, Fetched: 7 row(s)
```

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

QUERY:

```
SELECT brand, SUM(price) AS Sales

FROM RetailDB_EU1

WHERE brand <>'' AND event_type='purchase'

GROUP BY brand

ORDER BY Sales DESC LIMIT 1 ;
```

Answer:



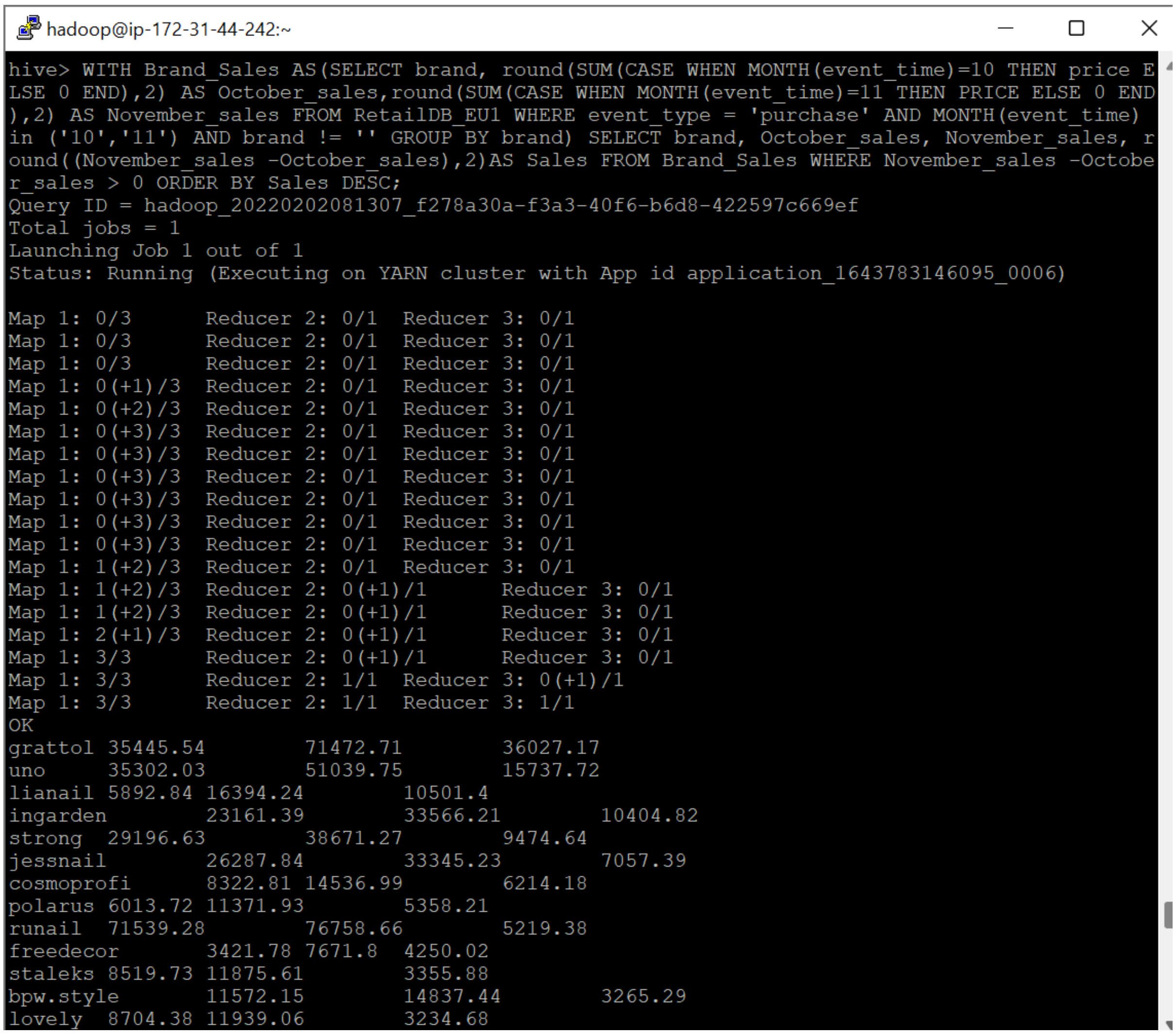
```
hive> SELECT brand, SUM(price) AS Sales FROM RetailDB_EU1 WHERE brand <>'' AND event_type='pu
rchase' GROUP BY brand ORDER BY Sales DESC LIMIT 1 ;
Query ID = hadoop_20220202081148_9f6d9d3b-0ab9-4990-a24e-a8681d2da5d5
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1643783146095_0006)

Map 1: 0/3      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0/3      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0/3      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+1)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 1(+2)/3  Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 2(+1)/3  Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 3/3      Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 3/3      Reducer 2: 1/1  Reducer 3: 0/1
Map 1: 3/3      Reducer 2: 1/1  Reducer 3: 0(+1)/1
Map 1: 3/3      Reducer 2: 1/1  Reducer 3: 1/1
OK
runail  148297.9400000006
Time taken: 27.715 seconds, Fetched: 1 row(s)
```

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

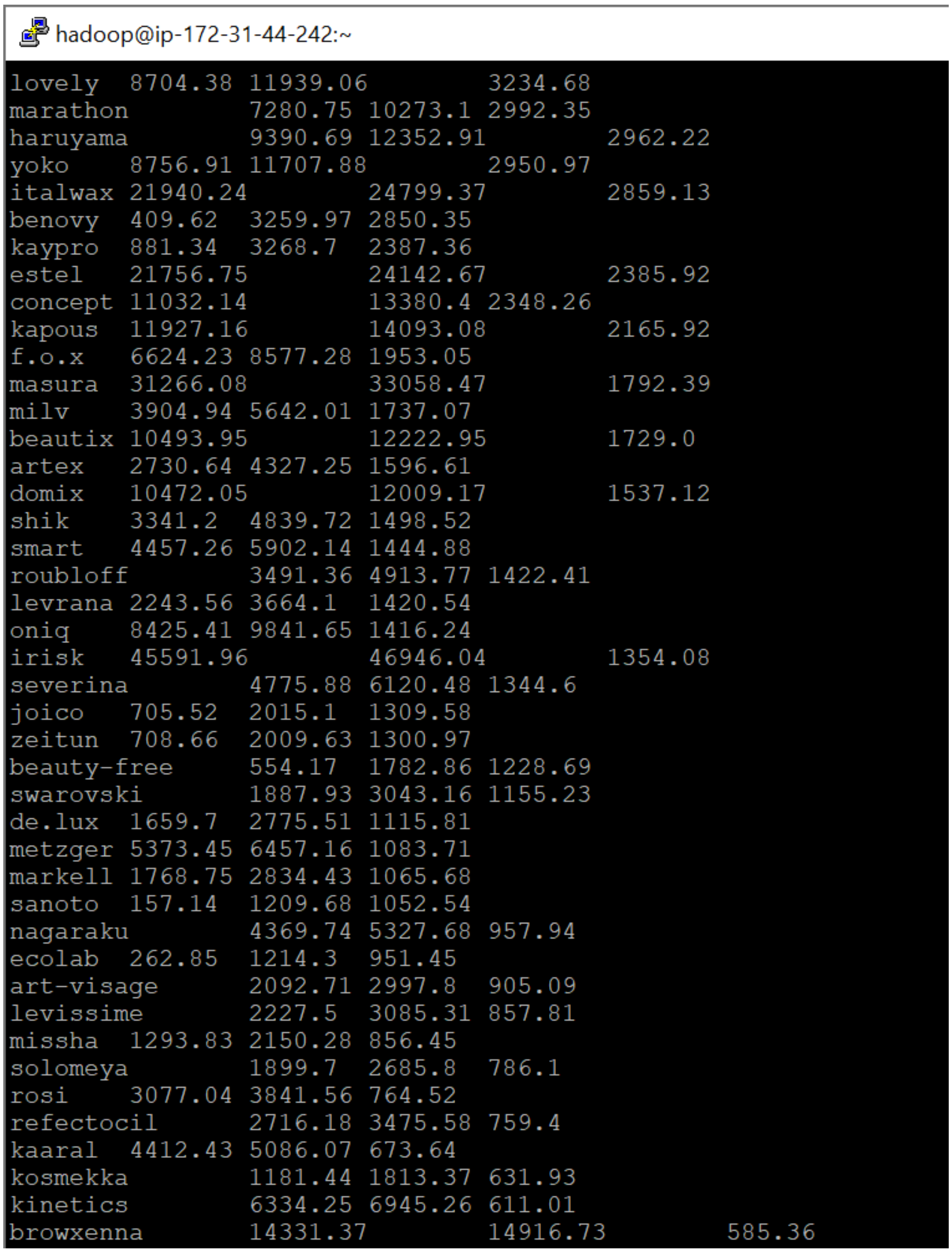
QUERY:

```
WITH Brand_Sales AS

      (SELECT brand, round(SUM(CASE WHEN MONTH(event_time)=10 THEN price
      ELSE 0 END),2) AS October_sales, round(SUM(CASE WHEN
      MONTH(event_time)=11 THEN PRICE ELSE 0 END),2) AS November_sales

      FROM RetailDB_EU1

      WHERE event_type = 'purchase' AND MONTH(event_time) in ('10','11')
      AND brand != ''

      GROUP BY brand)

SELECT brand, October_sales, November_sales, round((November_sales -
October_sales),2)AS Sales

FROM Brand_Sales

WHERE November_sales -October_sales > 0

ORDER BY Sales DESC;
```

Answer:

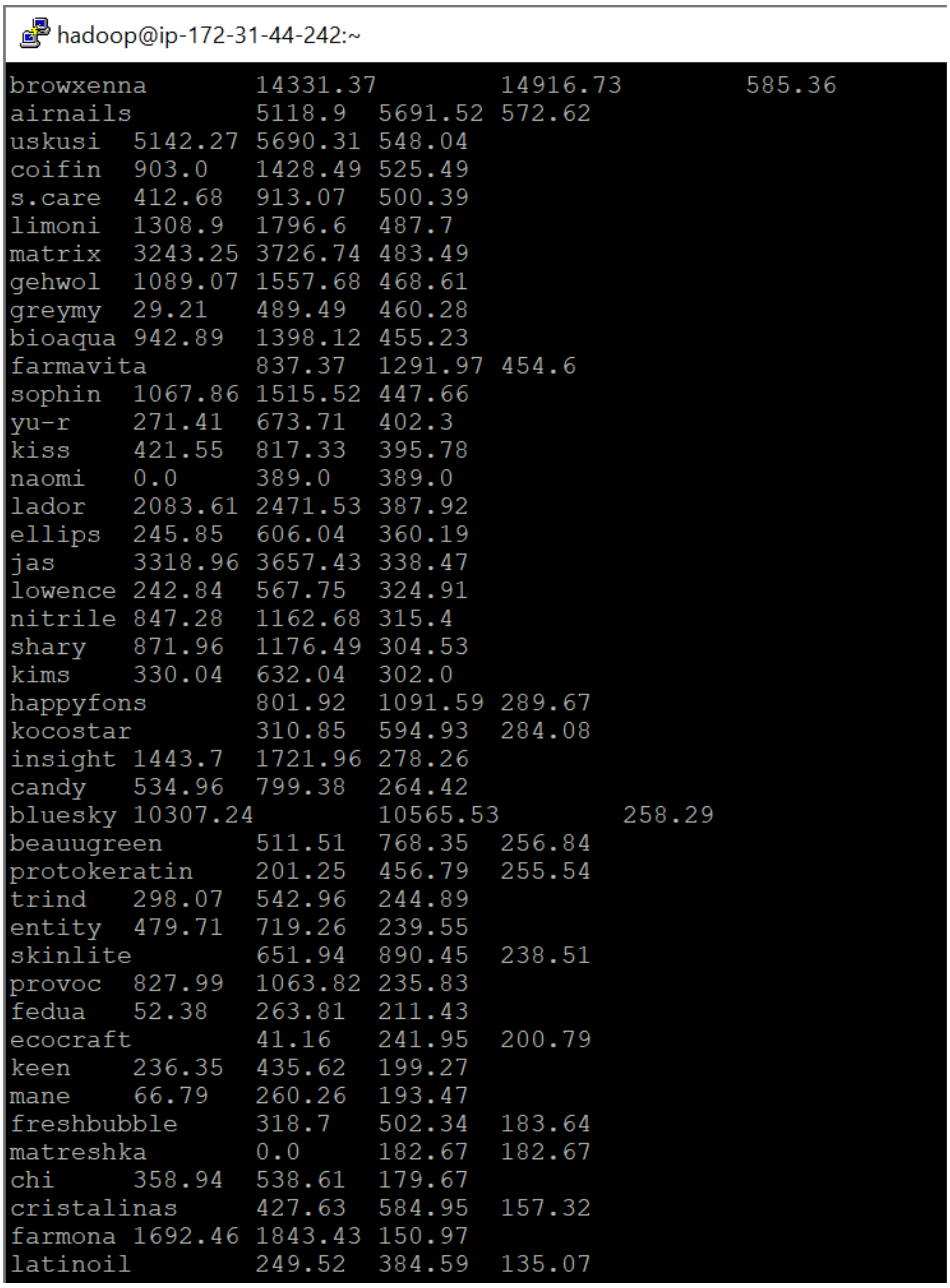

hadoop@ip-172-31-44-242:~

```
hive> WITH Brand_Sales AS(SELECT brand, round(SUM(CASE WHEN MONTH(event_time)=10 THEN price E
LSE 0 END),2) AS October_sales,round(SUM(CASE WHEN MONTH(event_time)=11 THEN PRICE ELSE 0 END
),2) AS November_sales FROM RetailDB_EU1 WHERE event_type = 'purchase' AND MONTH(event_time)
in ('10','11') AND brand != '' GROUP BY brand) SELECT brand, October_sales, November_sales, r
ound((November_sales -October_sales),2)AS Sales FROM Brand_Sales WHERE November_sales -Octobe
r_sales > 0 ORDER BY Sales DESC;
Query ID = hadoop_20220202081307_f278a30a-f3a3-40f6-b6d8-422597c669ef
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1643783146095_0006)

Map 1: 0/3      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0/3      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0/3      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+1)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+2)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 1(+2)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 1(+2)/3  Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 1(+2)/3  Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 2(+1)/3  Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 3/3      Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 3/3      Reducer 2: 1/1  Reducer 3: 0(+1)/1
Map 1: 3/3      Reducer 2: 1/1  Reducer 3: 1/1
OK
grattol 35445.54        71472.71        36027.17
uno     35302.03        51039.75        15737.72
lianail 5892.84 16394.24        10501.4
ingarden        23161.39        33566.21        10404.82
strong  29196.63        38671.27        9474.64
jessnail        26287.84        33345.23        7057.39
cosmoprofi      8322.81 14536.99        6214.18
polarus 6013.72 11371.93        5358.21
runail  71539.28        76758.66        5219.38
freedecor       3421.78 7671.8  4250.02
staleks 8519.73 11875.61        3355.88
bpw.style       11572.15        14837.44        3265.29
lovely  8704.38 11939.06        3234.68
```

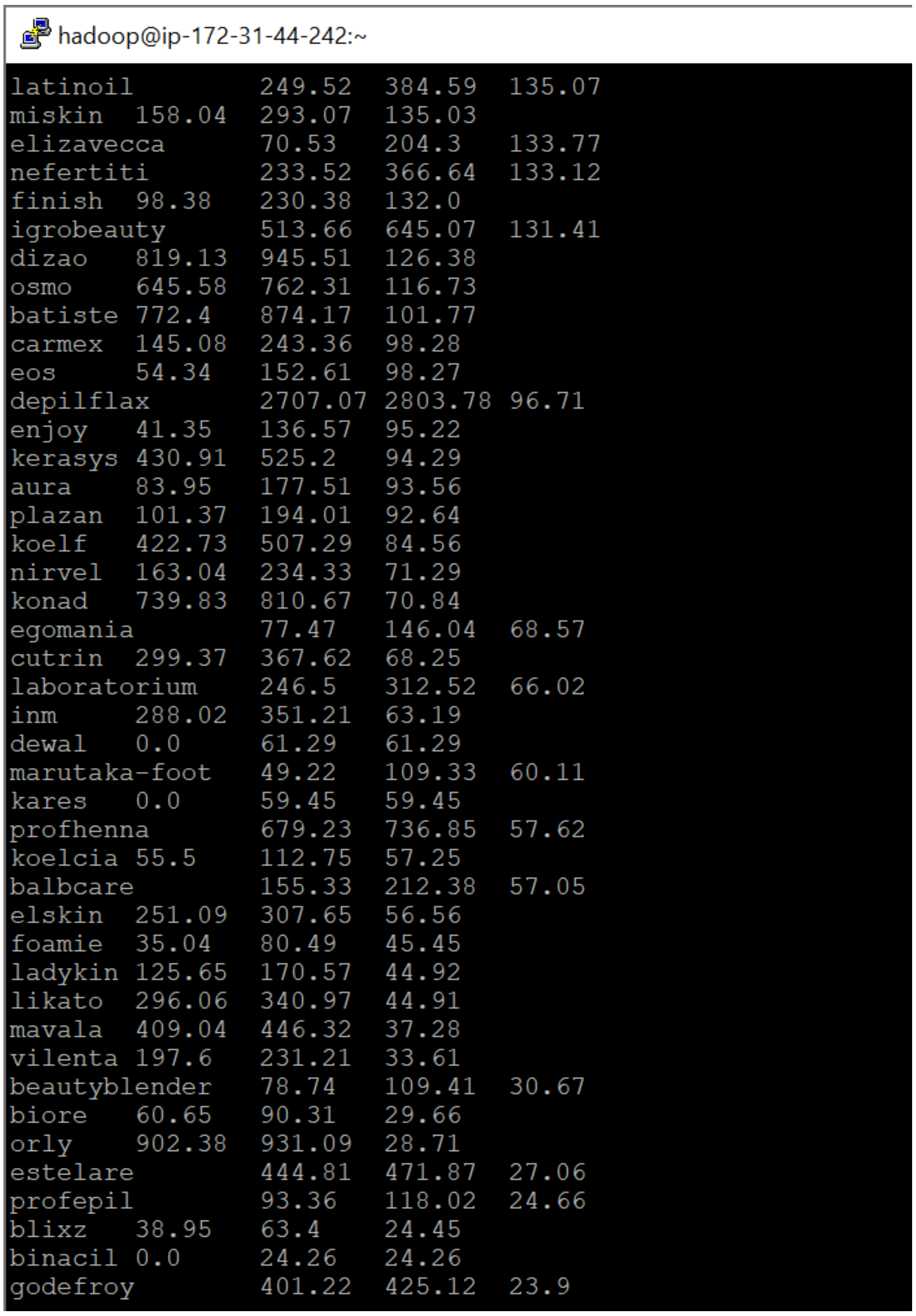


hadoop@ip-172-31-44-242:~

```
lovely  8704.38 11939.06        3234.68
marathon        7280.75 10273.1 2992.35
haruyama        9390.69 12352.91        2962.22
yoko    8756.91 11707.88        2950.97
italwax 21940.24        24799.37        2859.13
benovy  409.62  3259.97 2850.35
kaypro  881.34  3268.7  2387.36
estel   21756.75        24142.67        2385.92
concept 11032.14        13380.4 2348.26
kapous  11927.16        14093.08        2165.92
f.o.x   6624.23 8577.28 1953.05
masura  31266.08        33058.47        1792.39
milv    3904.94 5642.01 1737.07
beautix 10493.95        12222.95        1729.0
artex   2730.64 4327.25 1596.61
domix   10472.05        12009.17        1537.12
shik    3341.2  4839.72 1498.52
smart   4457.26 5902.14 1444.88
roubloff        3491.36 4913.77 1422.41
levrana 2243.56 3664.1  1420.54
oniq    8425.41 9841.65 1416.24
irisk   45591.96        46946.04        1354.08
severina        4775.88 6120.48 1344.6
joico   705.52  2015.1  1309.58
zeitun  708.66  2009.63 1300.97
beauty-free     554.17  1782.86 1228.69
swarovski       1887.93 3043.16 1155.23
de.lux  1659.7  2775.51 1115.81
metzger 5373.45 6457.16 1083.71
markell 1768.75 2834.43 1065.68
sanoto  157.14  1209.68 1052.54
nagaraku        4369.74 5327.68 957.94
ecolab  262.85  1214.3  951.45
art-visage      2092.71 2997.8  905.09
levissime       2227.5  3085.31 857.81
missha  1293.83 2150.28 856.45
solomeya        1899.7  2685.8  786.1
rosi    3077.04 3841.56 764.52
refectocil      2716.18 3475.58 759.4
kaaral  4412.43 5086.07 673.64
kosmekka        1181.44 1813.37 631.93
kinetics        6334.25 6945.26 611.01
browxenna       14331.37        14916.73        585.36
```

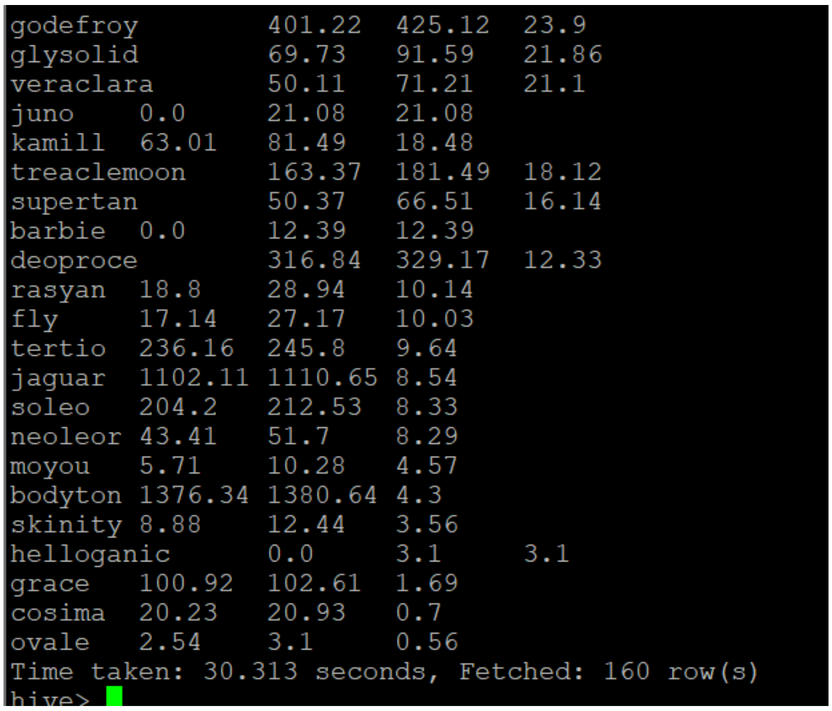

hadoop@ip-172-31-44-242:~

```
browxenna       14331.37        14916.73        585.36
airnails        5118.9  5691.52 572.62
uskusi  5142.27 5690.31 548.04
coifin  903.0   1428.49 525.49
s.care  412.68  913.07  500.39
limoni  1308.9  1796.6  487.7
matrix  3243.25 3726.74 483.49
gehwol  1089.07 1557.68 468.61
greymy  29.21   489.49  460.28
bioaqua 942.89  1398.12 455.23
farmavita       837.37  1291.97 454.6
sophin  1067.86 1515.52 447.66
yu-r    271.41  673.71  402.3
kiss    421.55  817.33  395.78
naomi   0.0     389.0   389.0
lador   2083.61 2471.53 387.92
ellips  245.85  606.04  360.19
jas     3318.96 3657.43 338.47
lowence 242.84  567.75  324.91
nitrile 847.28  1162.68 315.4
shary   871.96  1176.49 304.53
kims    330.04  632.04  302.0
happyfons       801.92  1091.59 289.67
kocostar        310.85  594.93  284.08
insight 1443.7  1721.96 278.26
candy   534.96  799.38  264.42
bluesky 10307.24        10565.53        258.29
beauugreen      511.51  768.35  256.84
protokeratin    201.25  456.79  255.54
trind   298.07  542.96  244.89
entity  479.71  719.26  239.55
skinlite        651.94  890.45  238.51
provoc  827.99  1063.82 235.83
fedua   52.38   263.81  211.43
ecocraft        41.16   241.95  200.79
keen    236.35  435.62  199.27
mane    66.79   260.26  193.47
freshbubble     318.7   502.34  183.64
matreshka       0.0     182.67  182.67
chi     358.94  538.61  179.67
cristalinas     427.63  584.95  157.32
farmona 1692.46 1843.43 150.97
latinoil        249.52  384.59  135.07
```



hadoop@ip-172-31-44-242:~

```
latinoil        249.52  384.59  135.07
miskin  158.04  293.07  135.03
elizavecca      70.53   204.3   133.77
nefertiti       233.52  366.64  133.12
finish  98.38   230.38  132.0
igrobeauty      513.66  645.07  131.41
dizao   819.13  945.51  126.38
osmo    645.58  762.31  116.73
batiste 772.4   874.17  101.77
carmex  145.08  243.36  98.28
eos     54.34   152.61  98.27
depilflax       2707.07 2803.78 96.71
enjoy   41.35   136.57  95.22
kerasys 430.91  525.2   94.29
aura    83.95   177.51  93.56
plazan  101.37  194.01  92.64
koelf   422.73  507.29  84.56
nirvel  163.04  234.33  71.29
konad   739.83  810.67  70.84
egomania        77.47   146.04  68.57
cutrin  299.37  367.62  68.25
laboratorium    246.5   312.52  66.02
inm     288.02  351.21  63.19
dewal   0.0     61.29   61.29
marutaka-foot   49.22   109.33  60.11
kares   0.0     59.45   59.45
profhenna       679.23  736.85  57.62
koelcia 55.5    112.75  57.25
balbcare        155.33  212.38  57.05
elskin  251.09  307.65  56.56
foamie  35.04   80.49   45.45
ladykin 125.65  170.57  44.92
likato  296.06  340.97  44.91
mavala  409.04  446.32  37.28
vilenta 197.6   231.21  33.61
beautyblender   78.74   109.41  30.67
biore   60.65   90.31   29.66
orly    902.38  931.09  28.71
estelare        444.81  471.87  27.06
profepil        93.36   118.02  24.66
blixz   38.95   63.4    24.45
binacil 0.0     24.26   24.26
godefroy        401.22  425.12  23.9
```



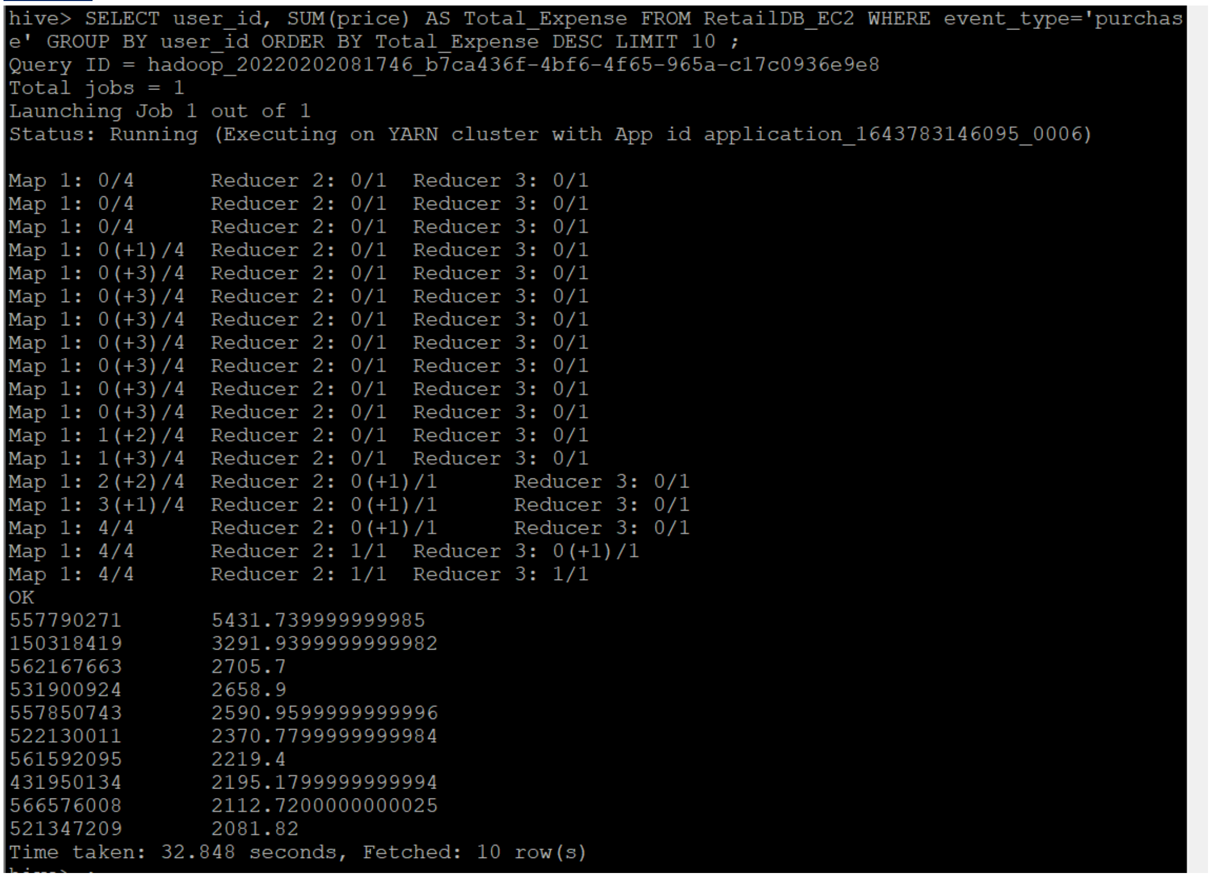
```
godefroy        401.22  425.12  23.9
glysolid        69.73   91.59   21.86
veraclara       50.11   71.21   21.1
juno    0.0     21.08   21.08
kamill  63.01   81.49   18.48
treaclemoon     163.37  181.49  18.12
supertan        50.37   66.51   16.14
barbie  0.0     12.39   12.39
deoproce        316.84  329.17  12.33
rasyan  18.8    28.94   10.14
fly     17.14   27.17   10.03
tertio  236.16  245.8   9.64
jaguar  1102.11 1110.65 8.54
soleo   204.2   212.53  8.33
neoleor 43.41   51.7    8.29
moyou   5.71    10.28   4.57
bodyton 1376.34 1380.64 4.3
skinity 8.88    12.44   3.56
helloganic      0.0     3.1     3.1
grace   100.92  102.61  1.69
cosima  20.23   20.93   0.7
ovale   2.54    3.1     0.56
Time taken: 30.313 seconds, Fetched: 160 row(s)
hive>
```

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.

Query:

```
SELECT user_id, SUM(price) AS Total_Expense

FROM RetailDB_EU1

WHERE event_type='purchase'

GROUP BY user_id

ORDER BY Total_Expense DESC LIMIT 10 ;
```

Answer:



```
hive> SELECT user_id, SUM(price) AS Total_Expense FROM RetailDB_EC2 WHERE event_type='purchas
e' GROUP BY user_id ORDER BY Total_Expense DESC LIMIT 10 ;
Query ID = hadoop_20220202081746_b7ca436f-4bf6-4f65-965a-c17c0936e9e8
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1643783146095_0006)

Map 1: 0/4      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0/4      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0/4      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+1)/4  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/4  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 1(+2)/4  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 1(+3)/4  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 2(+2)/4  Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 3(+1)/4  Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 4/4      Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 4/4      Reducer 2: 1/1  Reducer 3: 0(+1)/1
Map 1: 4/4      Reducer 2: 1/1  Reducer 3: 1/1
OK
557790271       5431.739999999985
150318419       3291.9399999999982
562167663       2705.7
531900924       2658.9
557850743       2590.9599999999996
522130011       2370.7799999999984
561592095       2219.4
431950134       2195.1799999999994
566576008       2112.7200000000025
521347209       2081.82
Time taken: 32.848 seconds, Fetched: 10 row(s)
```

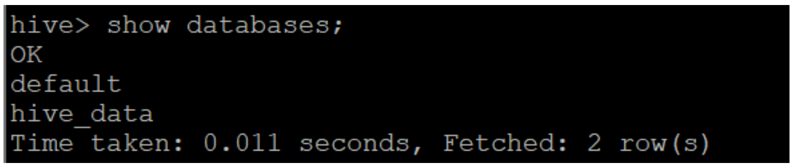
Dropping the Database and Terminating the Cluster

Drop Database:

- Run the below queries to check out database:

  Query:

  `Show databases;`

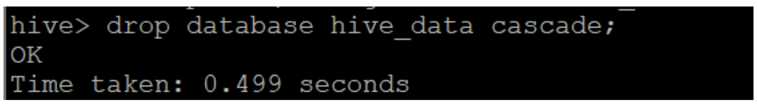


```
hive> show databases;
OK
default
hive_data
Time taken: 0.011 seconds, Fetched: 2 row(s)
```

Drop the "hive_data" database (Use cascade to drop the tables inside before dropping the database):

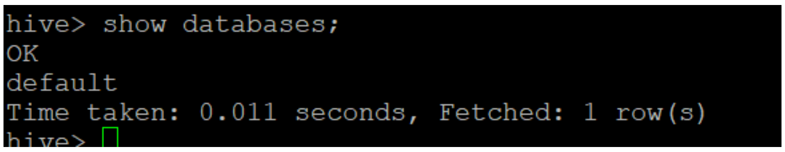
Query:

Drop database hive_data cascade;



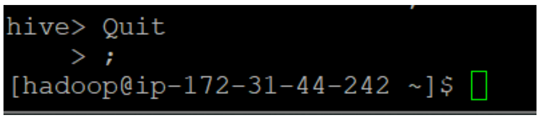
```
hive> drop database hive_data cascade;
OK
Time taken: 0.499 seconds
```

- Reconfirm if database is dropped using (Show databases;) :



```
hive> show databases;
OK
default
Time taken: 0.011 seconds, Fetched: 1 row(s)
hive>
```

- Quit Hive:

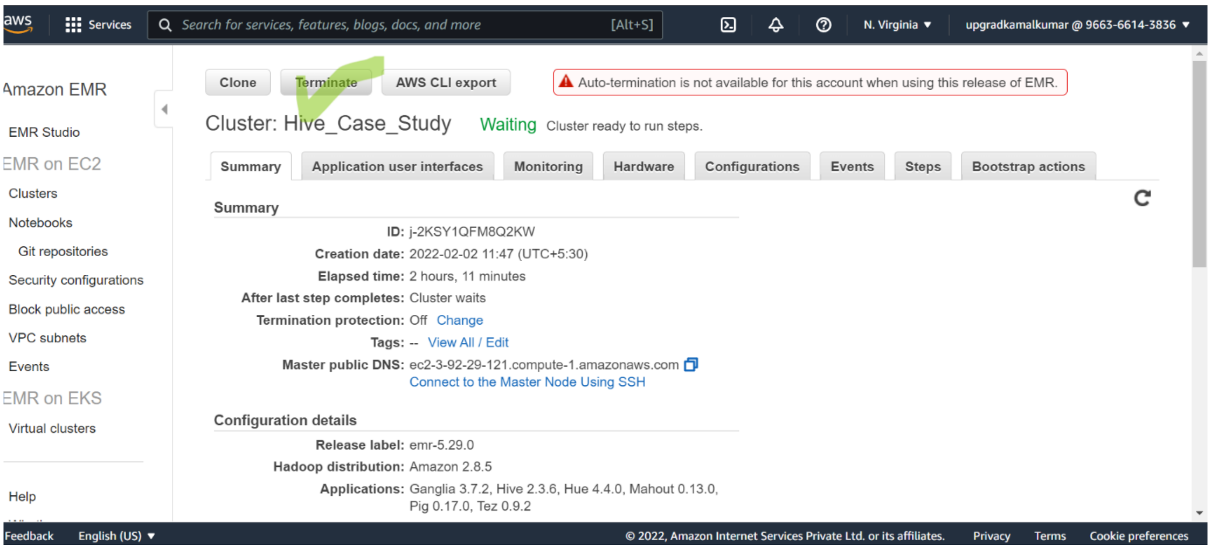


```
hive> Quit
    > ;
[hadoop@ip-172-31-44-242 ~]$
```

- Type "Exit" to quit from putty.

Terminate EMR:

- Select Terminate > Select Terminate in the pop-up window >Status changes to Terminating.



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

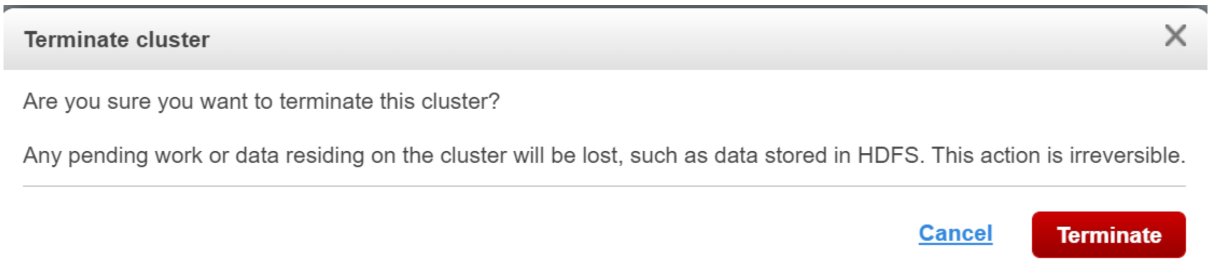
Cluster: Hive_Case_Study Waiting Cluster ready to run steps.

Summary

ID: j-2KSY1QFM8Q2KW
Creation date: 2022-02-02 11:47 (UTC+5:30)
Elapsed time: 2 hours, 11 minutes
After last step completes: Cluster waits
Termination protection: Off Change
Tags: -- View All / Edit
Master public DNS: ec2-3-92-29-121.compute-1.amazonaws.com
Connect to the Master Node Using SSH

Configuration details

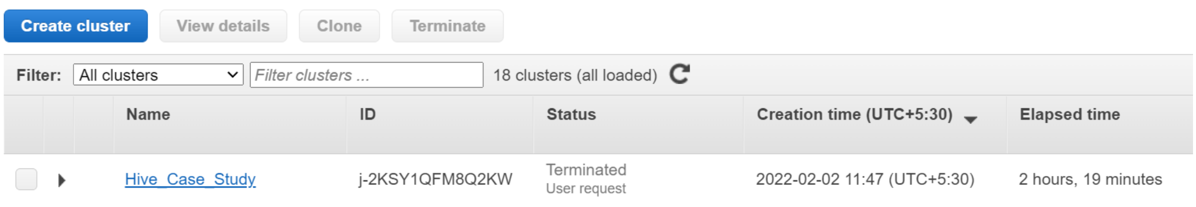
Release label: emr-5.29.0
Hadoop distribution: Amazon 2.8.5
Applications: Ganglia 3.7.2, Hive 2.3.6, Hue 4.4.0, Mahout 0.13.0, Pig 0.17.0, Tez 0.9.2



Terminate cluster

Are you sure you want to terminate this cluster?

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



Filter: All clusters | Filter clusters ... | 18 clusters (all loaded)

Name	ID	Status	Creation time (UTC+5:30)	Elapsed time
Hive_Case_Study	j-2KSY1QFM8Q2KW	Terminated User request	2022-02-02 11:47 (UTC+5:30)	2 hours, 19 minutes

Cluster is Terminated!