CASE STUDY

BATCH- DS C30
SUBMITTED BY: SOWMYA CHERUKUWADA, SAKSHAM KUMAR

Problem:

In modern world, online marketing is gaining popularity, technology companies are exploring ways to improve their sales by analyzing customer behavior and gaining insights on product trends. And websites make it easier for customers to find the products they need without spending a lot of money. And obvious to say, the role of major data analysts is among the most sought-after job profiles over the decade.

For better business decisions, E-commerce websites find their way by tracking the number of clicks made customers and their time spent on websites searching for patterns within them. This type the data collected is called click stream data. And websites make it easier for customers to get the products they need without spending too much.

So, in this case study, we worked on click stream data to get information and make decisions on how E-commerce websites can improve their sales.

Objective

The purpose is to extract data and collect data from the e-commerce company's real-life data set.

Data

Here we use the public data. It is a click stream data of a cosmetic store. This Clickstream data contains all the logs of how a person navigated an e-commerce website. It also contains other

details such as customer time spent on all pages, number of clicks made, customer id, add items to cart etc.

The data are

https://e-commerce-events-ml.s3.amazonaws.com/2019-Oct.csv
https://e-commerce-events-ml.s3.amazonaws.com/2019-Nov.csv

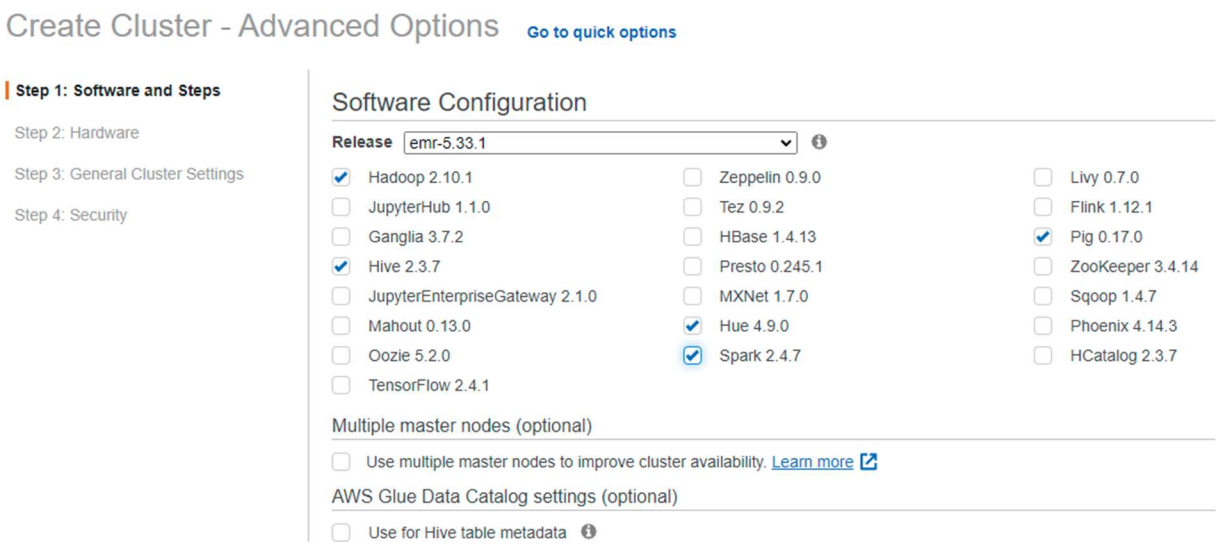
Here I follow the using the following steps

- Copying the data set into the HDFS:
    - First we launch an EMR cluster,
    - And move the data from the S3 bucket into the HDFS
- Creating the database and launching Hive queries on EMR cluster:
    - Creating the structure of the database,
    - Optimizing techniques to run the queries as efficiently as possible.
    - And then we showing the improvement of the performance after using optimization on any single query.
    - And then run Hive queries to answer the questions asked.
- Cleaning up
    - And last we drop the database,
    - And Terminate the cluster

EMR CLUSTER CREATION

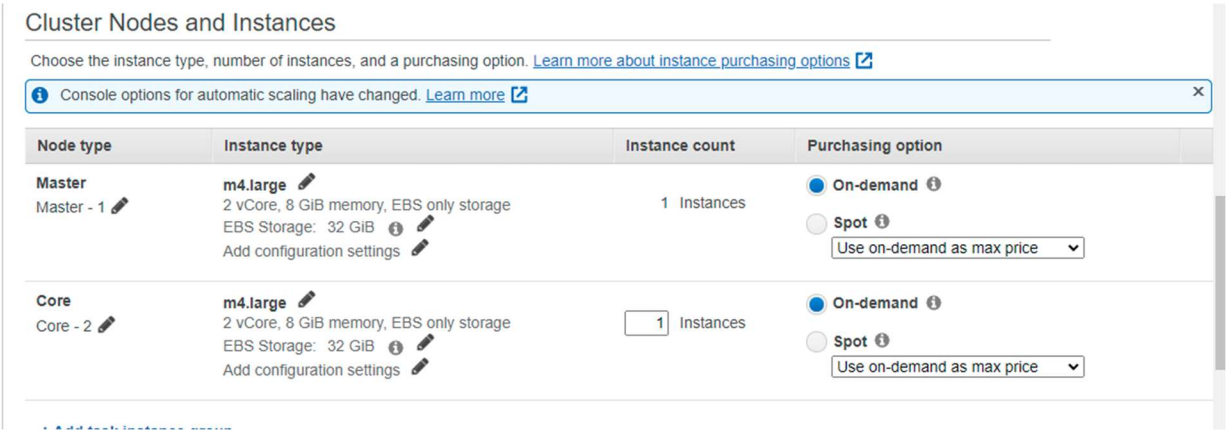
1.1EMR Cluster → Create cluster → Advanced Option → Release emr – 5.33.1

Here we choose Hadoop 2.10.1, Hive 2.3.7, Hue 4.9.0, Spark 2.4.7 and Pig 0.17.0



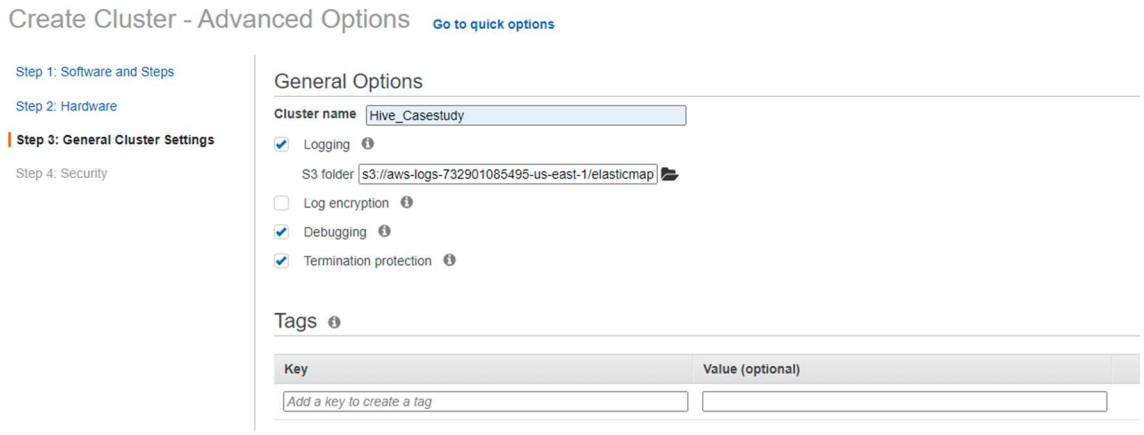
1.2 Hardware

Here we select instance m4.large foe both master and core node and for both instance count is 1.



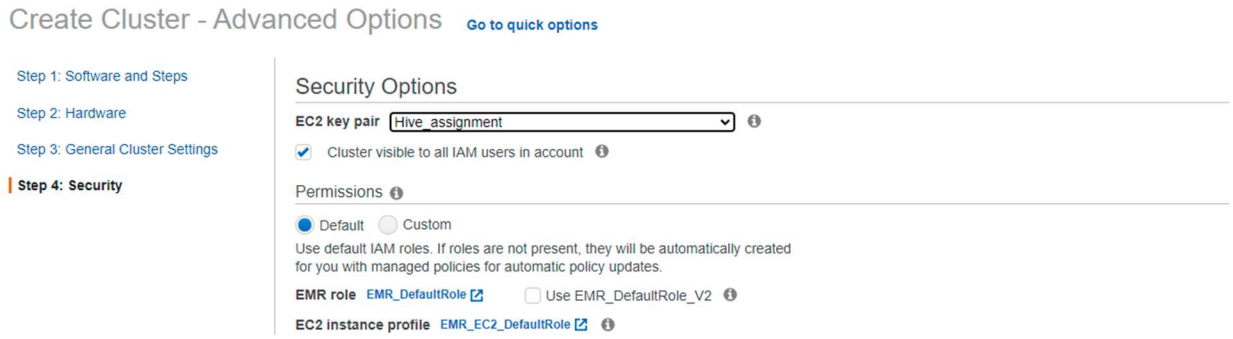
1.3 General Cluster Setting

In this setting we named the cluster as “Hive_Casestudy”.



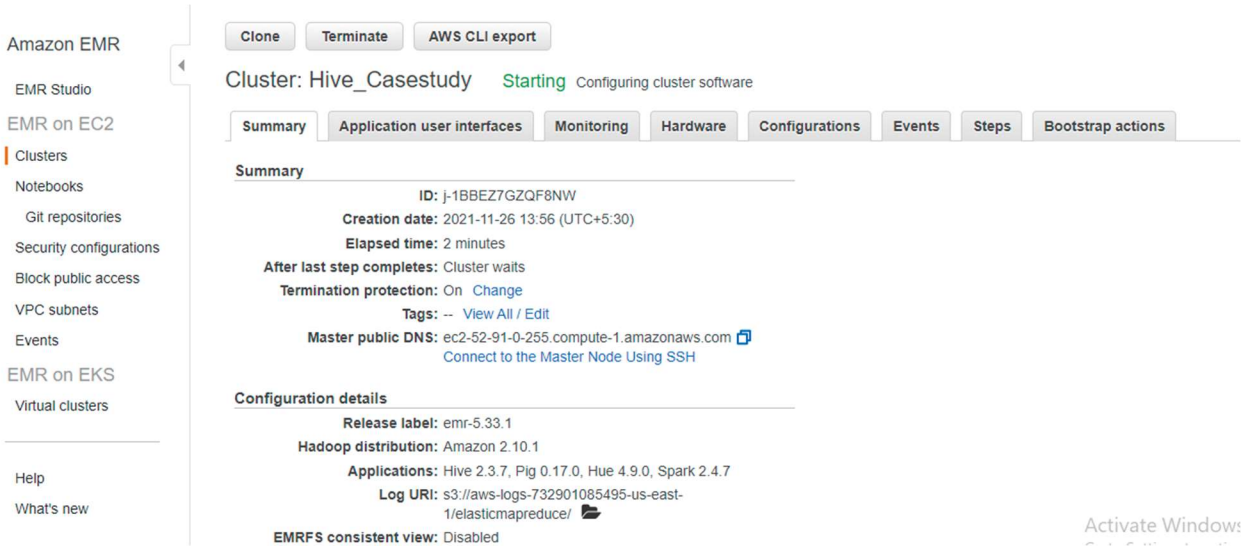
1.4 Security

For security we choose the EC2 key pair (which one created before the creation of EMR cluster) “Hive_assignment”.



1.5 After creation of cluster, in Security groups for Master, we edit inbound rule and add SSH type and source is anywhere IPV4.

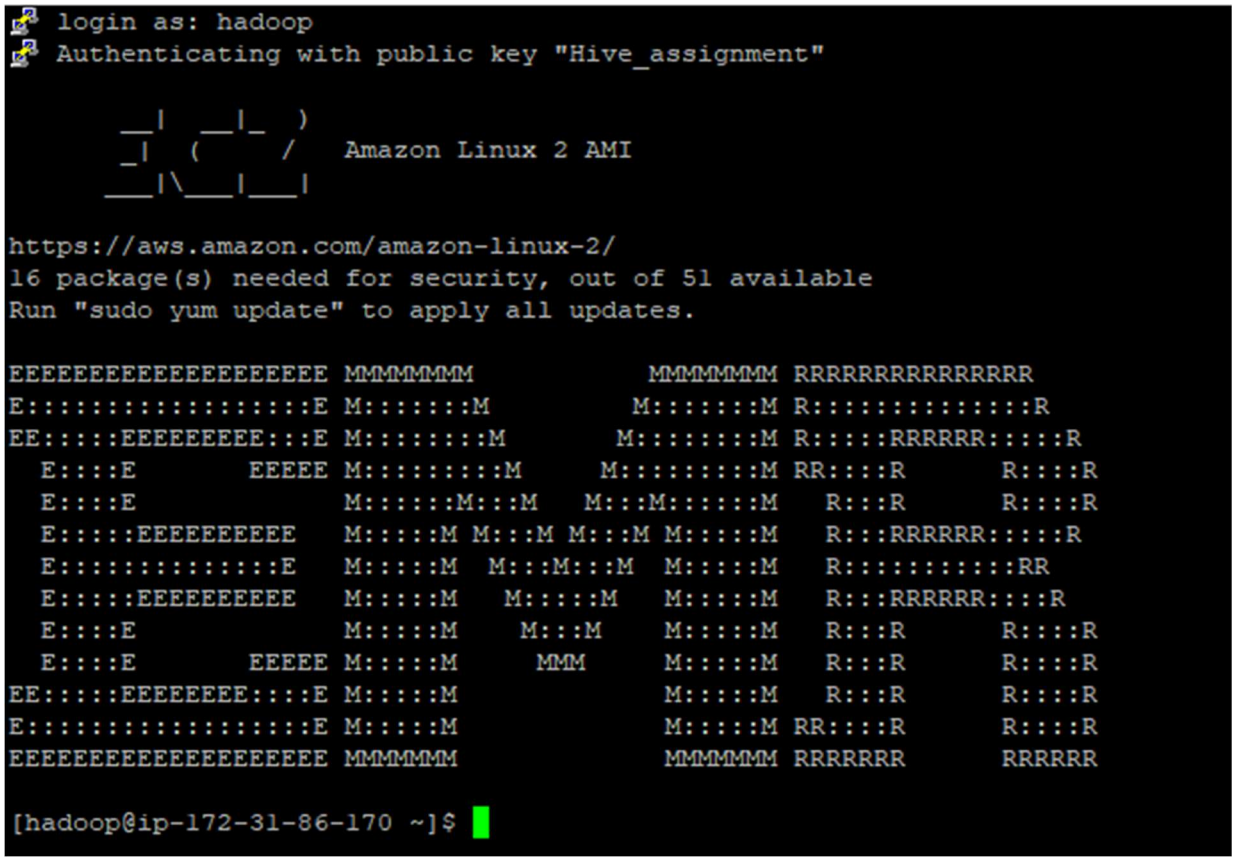
1.6 And then the cluster is ready for window.


Amazon EMR
EMR Studio
EMR on EC2
Clusters
Notebooks
Git repositories
Security configurations
Block public access
VPC subnets
Events
EMR on EKS
Virtual clusters
Help
What's new
Clone
Terminate
AWS CLI export
Cluster: Hive_Casestudy
Starting
Configuring cluster software
Summary
Application user interfaces
Monitoring
Hardware
Configurations
Events
Steps
Bootstrap actions
Summary
ID: j-1BBEZ7GZQF8NW
Creation date: 2021-11-26 13:56 (UTC+5:30)
Elapsed time: 2 minutes
After last step completes: Cluster waits
Termination protection: On Change
Tags: -- View All / Edit
Master public DNS: ec2-52-91-0-255.compute-1.amazonaws.com
Connect to the Master Node Using SSH
Configuration details
Release label: emr-5.33.1
Hadoop distribution: Amazon 2.10.1
Applications: Hive 2.3.7, Pig 0.17.0, Hue 4.9.0, Spark 2.4.7
Log URI: s3://aws-logs-732901085495-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled
Activate Windows

HADOOP query

2.1 Starting EMR cluster

- hadoop



2.2 Uploading the datasets

- wget https://e-commerce-events-ml.s3.amazonaws.com/2019-Oct.csv
- wget https://e-commerce-events-ml.s3.amazonaws.com/2019-Nov.csv

```
[hadoop@ip-172-31-86-170 ~]$ wget https://e-commerce-events-ml.s3.amazonaws.com/2019-Nov.csv ;
--2021-11-29 22:46:12--  https://e-commerce-events-ml.s3.amazonaws.com/2019-Nov.csv
Resolving e-commerce-events-ml.s3.amazonaws.com (e-commerce-events-ml.s3.amazonaws.com)... 52.217.33.100
Connecting to e-commerce-events-ml.s3.amazonaws.com (e-commerce-events-ml.s3.amazonaws.com)|52.217.33.100|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 545839412 (521M) [text/csv]
Saving to: '2019-Nov.csv'

100%[==========================================================================================================================>] 545,839,412 62.8MB/s   in 9.0s

2021-11-29 22:46:21 (57.7 MB/s) - '2019-Nov.csv' saved [545839412/545839412]
```

2.3 Creating new directory "casestudy"

- Hadoop fs –ls /
- Hadoop fs –mkdir /casestudy
- Hadoop fs –ls /

```
[hadoop@ip-172-31-86-170 ~]$ hadoop fs -ls /
Found 4 items
drwxr-xr-x   - hdfs hdfsadmingroup          0 2021-11-29 22:51 /apps
drwxrwxrwt   - hdfs hdfsadmingroup          0 2021-11-29 22:51 /tmp
drwxr-xr-x   - hdfs hdfsadmingroup          0 2021-11-29 22:51 /user
drwxr-xr-x   - hdfs hdfsadmingroup          0 2021-11-29 22:51 /var
[hadoop@ip-172-31-86-170 ~]$ hadoop fs -mkdir /casestudy
```

```
[hadoop@ip-172-31-86-170 ~]$ hadoop fs -ls /
Found 5 items
drwxr-xr-x   - hdfs   hdfsadmingroup          0 2021-11-29 22:51 /apps
drwxr-xr-x   - hadoop hdfsadmingroup          0 2021-11-29 22:51 /casestudy
drwxrwxrwt   - hdfs   hdfsadmingroup          0 2021-11-29 22:51 /tmp
drwxr-xr-x   - hdfs   hdfsadmingroup          0 2021-11-29 22:51 /user
drwxr-xr-x   - hdfs   hdfsadmingroup          0 2021-11-29 22:51 /var
[hadoop@ip-172-31-86-170 ~]$
```

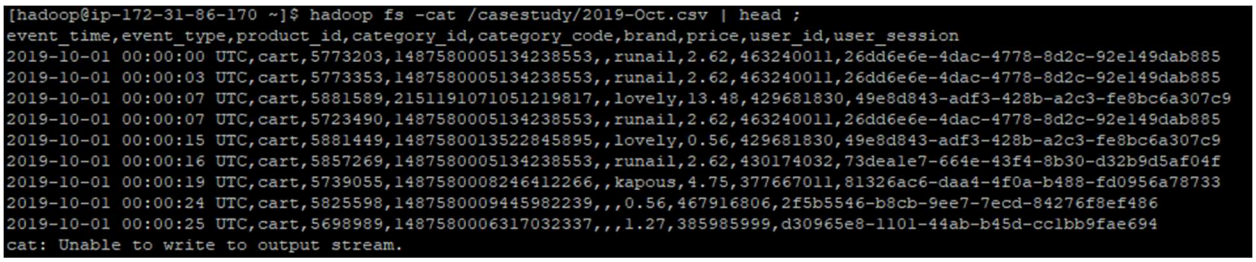
2.4 Loading data into this directory

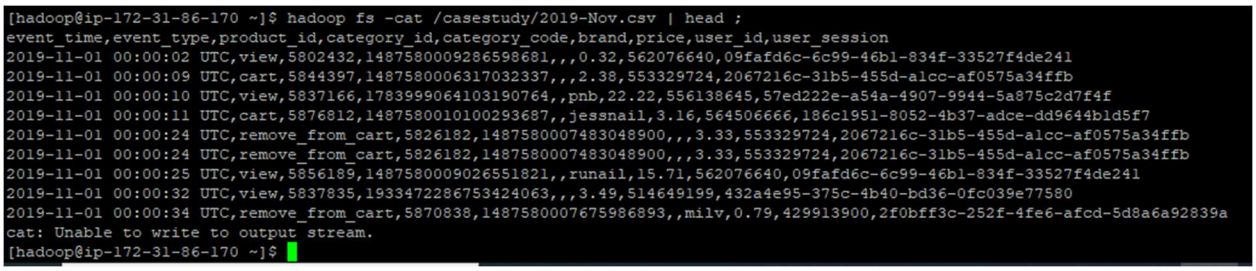
- Hadoop fs –put 2019-Oct.csv /casestudy
- Hadoop fs –put 2019-Nov.csv /casestudy

```
[hadoop@ip-172-31-86-170 ~]$ hadoop fs -put 2019-Nov.csv /casestudy ;
[hadoop@ip-172-31-86-170 ~]$ hadoop fs -put 2019-Oct.csv /casestudy ;
[hadoop@ip-172-31-86-170 ~]$
```

2.5 Checking the dataset

- Hadoop fs –cat /casestudy/2019-Oct.csv | head
- Hadoop fs –cat /casestudy/2019-Nov.csv | head





HIVE query

3.1 Launching Hive

- Hive

```
[hadoop@ip-172-31-86-170 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive> show databases ;
OK
default
Time taken: 0.752 seconds, Fetched: 1 row(s)
hive>
```

3.2 Creating new database – "new_assignment"

- CREATE DATABASE IF NOT EXISTS new_assignment ;
- SHOW DATABASES ;
- DESCRIBE DATABASE new_assignment ;

```
hive> CREATE DATABASE IF NOT EXISTS new_assignment ;
OK
Time taken: 0.449 seconds
hive> show databases ;
OK
default
new_assignment
Time taken: 0.087 seconds, Fetched: 2 row(s)
hive>
```

```
hive> DESCRIBE DATABASE new_assignment ;
OK
new_assignment          hdfs://ip-172-31-86-170.ec2.internal:8020/user/hive/warehouse/new_assignment.db hadoop  USER
Time taken: 0.058 seconds, Fetched: 1 row(s)
hive>
```

3.3 Creating external table – "retail"

Attribute_Name	Data_type	Description
event_time	timestamp	Time at which the event took place
event_type	string	Event type may be 'view', 'cart', 'remove_from_cart', 'purchase'
product_id	string	Unique identification of the product
category_id	string	Unique identification of the product category. Each product category contains several products
category_code	string	Name (if present) of the product category
brand	string	Name of the brand
price	float	Price of the product
user_id	bigint	Permanent user id
user_session	string	Identification for the user's session. Remains same for each user's session. It changes everytime the user returns back the website after a long pause

- CREATE EXTERNAL TABLE IF NOT EXISTS retail (event_time timestamp, event_typestring, 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 ("separatorChar" = "," , "quoteChar" = "\"", "escapeChar" = "\\") stored as textfile LOCATION '/casestudy' TBLPROPERTIES ("skip.header.line.count"="1") ;
- DESCRIBE retail ;

```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS retail (event_time timestamp, event_type string, product_id string, category_id string, category_code string, brand string, pr
ice decimal(10,3), user_id bigint, user_session string) ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' WITH SERDEPROPERTIES ("separatorChar" = "," , "quo
teChar" = "\"", "escapeChar" = "\\") stored as textfile LOCATION '/casestudy' TBLPROPERTIES ("skip.header.line.count"="1") ;
OK
Time taken: 0.333 seconds
```

```
hive> DESCRIBE retail ;
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: 0.098 seconds, Fetched: 9 row(s)
hive>
```

3.4 Loading the all dataset into external table “retail”

- LOAD DATA INPATH '/casestudy/2019-Oct.csv' INTO TABLE retail ;
- LOAD DATA INPATH '/casestudy/2019-Nov.csv' INTO TABLE retail ;

```
hive> LOAD DATA INPATH '/casestudy/2019-Oct.csv' INTO TABLE retail ;
Loading data to table default.retail
OK
Time taken: 1.576 seconds
hive>
```

```
hive> LOAD DATA INPATH '/casestudy/2019-Nov.csv' INTO TABLE retail ;
Loading data to table default.retail
OK
Time taken: 0.437 seconds
hive>
```

3.5 Checking the data in the table by month wise

- SELECT * FROM retail WHERE MONTH(event_time)=10 limit 5 ;
- SELECT * FROM retail WHERE MONTH(event_time)=11 limit 5 ;

```
hive> SELECT * FROM retail WHERE MONTH(event_time)=10 limit 5 ;
OK
2019-10-01 00:00:00 UTC cart    5773203 1487580005134238553             runail  2.62    463240011       26dd6e6e-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:03 UTC cart    5773353 1487580005134238553             runail  2.62    463240011       26dd6e6e-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:07 UTC cart    5881589 2151191071051219817             lovely  13.48   429681830       49e8d843-adf3-428b-a2c3-fe8bc6a307c9
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
Time taken: 1.875 seconds, Fetched: 5 row(s)
```

```
hive> SELECT * FROM retail WHERE MONTH(event_time)=11 limit 5 ;
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
2019-11-01 00:00:11 UTC cart    5876812 1487580010100293687             jessnail        3.16    564506666       186c1951-8052-4b37-adce-dd9644b1d5f7
2019-11-01 00:00:24 UTC remove_from_cart        5826182 1487580007483048900                     3.33    553329724       2067216c-31b5-455d-a1cc-af0575a34ffb
Time taken: 2.461 seconds, Fetched: 5 row(s)
```

3.6 Visualizing the table

- SELECT SUM(price) FROM retail WHERE MONTH(event_time)=10 AND event_type='purchase' ;

```
hive> SELECT SUM(price) FROM retail WHERE MONTH(event_time)=10 AND event_type='purchase' ;
Query ID = hadoop_20211129231018_cf096c3a-1103-43b9-958f-1d9204e6ff66
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_1638226296422_0002)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      2          2        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 64.38 s
--------------------------------------------------------------------------------
OK
1211538.4299997438
Time taken: 73.174 seconds, Fetched: 1 row(s)
```

- Since the execution time is very high for this (73.174 seconds) so we need to partition the table. First is for when event_type is “purchase” and the second is for month. This is because questions are from purchase and months.
- Also we clustered into 5 buckets for both using 'org.apache.hadoop.hive.serde2.OpenCSVSerde'.

3.7 Dynamics Partition

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

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

3.7.1 PARTITION 1: retail_1

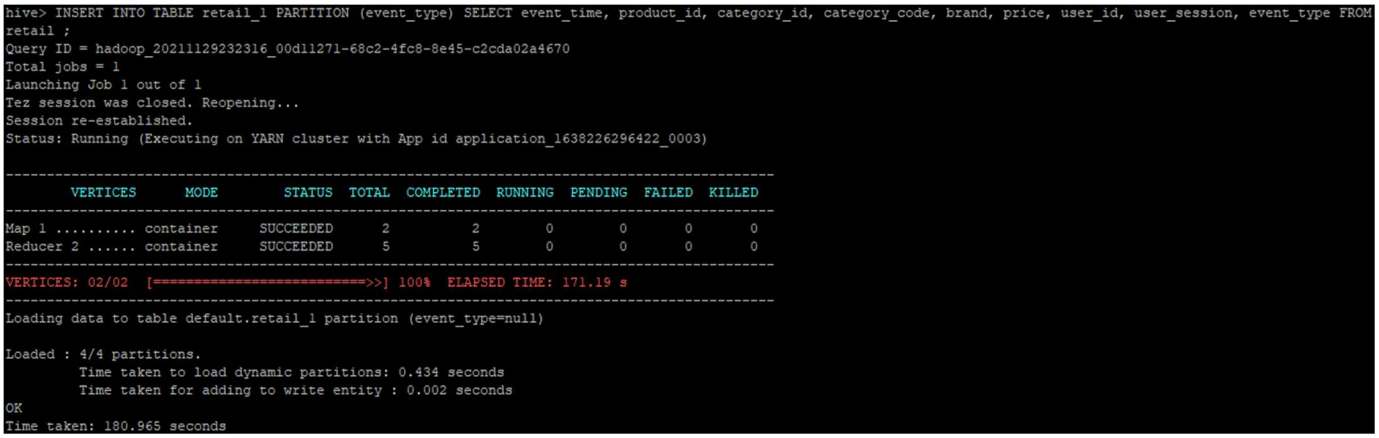
Partition on: event_type (there are 4 types and all questions are related to 'purchase')

- CREATE EXTERNAL TABLE IF NOT EXISTS retail_1 (event_time timestamp, product_id string, category_id string, category_code string, brand string, price decimal(10,3), user_id bigint, user_session string) PARTITIONED BY(event_type string) CLUSTERED BY (user_id) INTO 5 buckets ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS textfile ;
- DESCRIBE retail_1;

```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS retail_1 (event_time timestamp, product_id string, category_id string, category_code string, brand string, price decimal(10,3)
, user_id bigint, user_session string) PARTITIONED BY(event_type string) CLUSTERED BY (user_id) INTO 5 buckets ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSe
rde' STORED AS textfile ;
OK
Time taken: 0.093 seconds
```

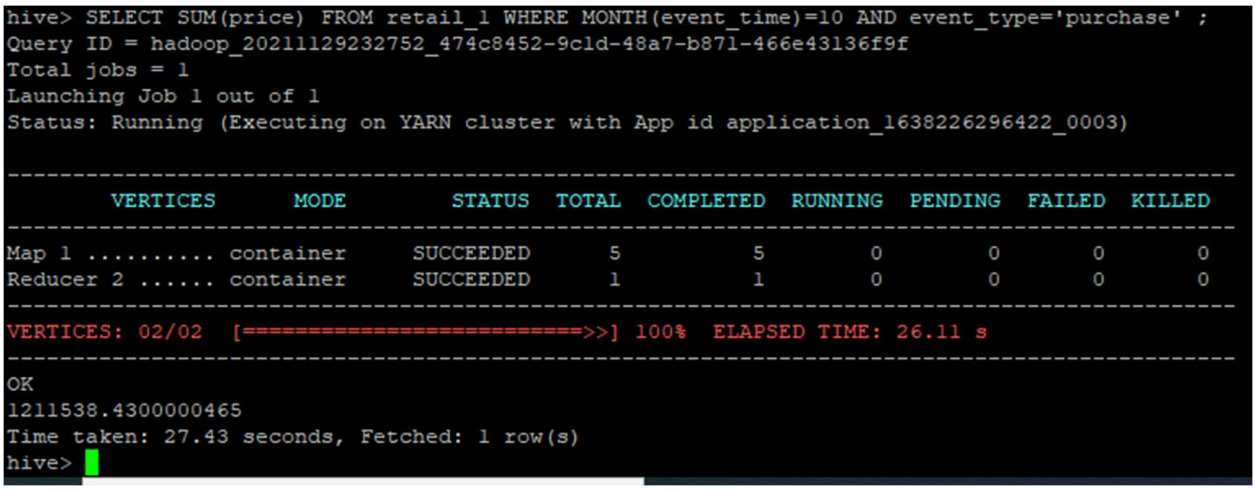
```
hive> DESCRIBE retail_1 ;
OK
event_time              string                  from deserializer
product_id              string                  from deserializer
category_id             string                  from deserializer
category_code           string                  from deserializer
brand                   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: 0.105 seconds, Fetched: 14 row(s)
hive>
```

- INSERT INTO TABLE retail_1 PARTITION (event_type) SELECT event_time, product_id, category_id, category_code, brand, price, user_id, user_session, event_type FROM retail ;



Now checking the execution time.

- SELECT SUM(price) FROM retail_part_1 WHERE MONTH(event_time)=10 AND event_type='purchase' ;



- Now it is ok because time is reduced to 27.43 seconds.

3.7.2 PARTITION TABLE 2: retail_2

Partition on: month

- CREATE EXTERNAL TABLE IF NOT EXISTS retail_2 (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) PARTITIONED BY(month int) CLUSTERED BY (brand) INTO 5 buckets ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS textfile ;

- Describe retail_2 ;

```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS retail_2 (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) PARTITIONED BY(month int) CLUSTERED BY (brand) INTO 5 buckets ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.
OpenCSVSerde' STORED AS textfile ;
OK
Time taken: 0.083 seconds
```

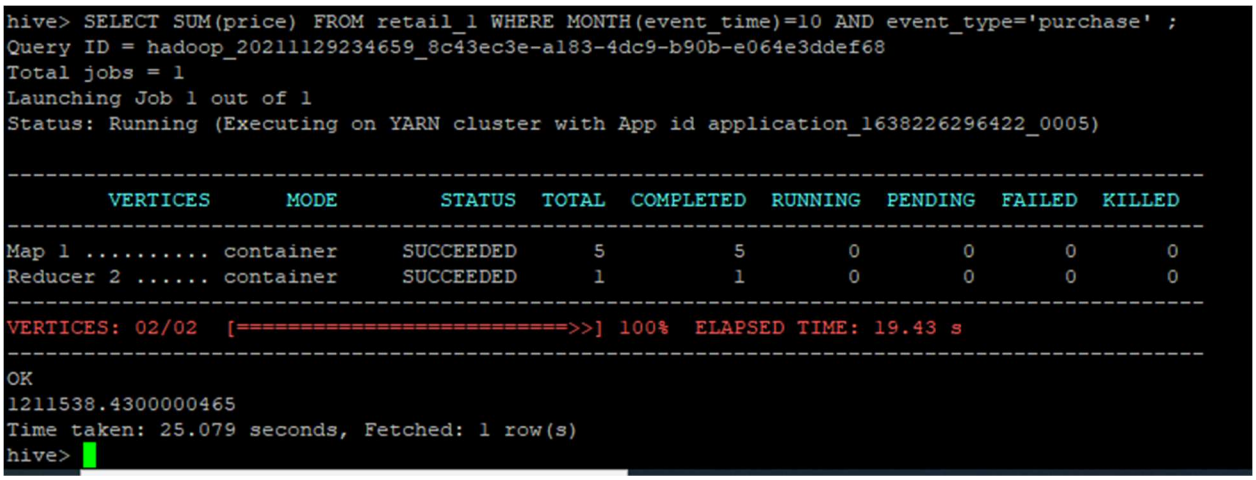
```
hive> DESCRIBE retail_2 ;
OK
event_time              string                  from deserializer
event_type              string                  from deserializer
product_id              string                  from deserializer
category_id             string                  from deserializer
category_code           string                  from deserializer
brand                   string                  from deserializer
price                   string                  from deserializer
user_id                 string                  from deserializer
user_session            string                  from deserializer
month                   int

# Partition Information
# col_name              data_type               comment

month                   int
Time taken: 0.065 seconds, Fetched: 15 row(s)
hive>
```

- INSERT INTO TABLE retail_2 PARTITION (month) SELECT event_time, event_type, product_id, category_id, category_code, brand, price, user_id, user_session, MONTH(CAST(REPLACE(event_time,'UTC','') AS timestamp)) FROM retail ;

```
hive> INSERT INTO TABLE retail_2 PARTITION (month) SELECT event_time, event_type, product_id, category_id, category_code, brand, price, user_id, user_session, MONTH(CAS
T(REPLACE(event_time,'UTC','') AS timestamp)) FROM retail ;
Query ID = hadoop_20211129233936_87486715-3d0b-4882-9779-71d9cf16454f
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1638226296422_0004)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..... container     SUCCEEDED      2          2        0        0       0       0
Reducer 2 ..... container     SUCCEEDED      5          5        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 193.32 s
--------------------------------------------------------------------------------
Loading data to table default.retail_2 partition (month=null)

Loaded : 2/2 partitions.
        Time taken to load dynamic partitions: 0.17 seconds
        Time taken for adding to write entity : 0.001 seconds
OK
Time taken: 194.465 seconds
```

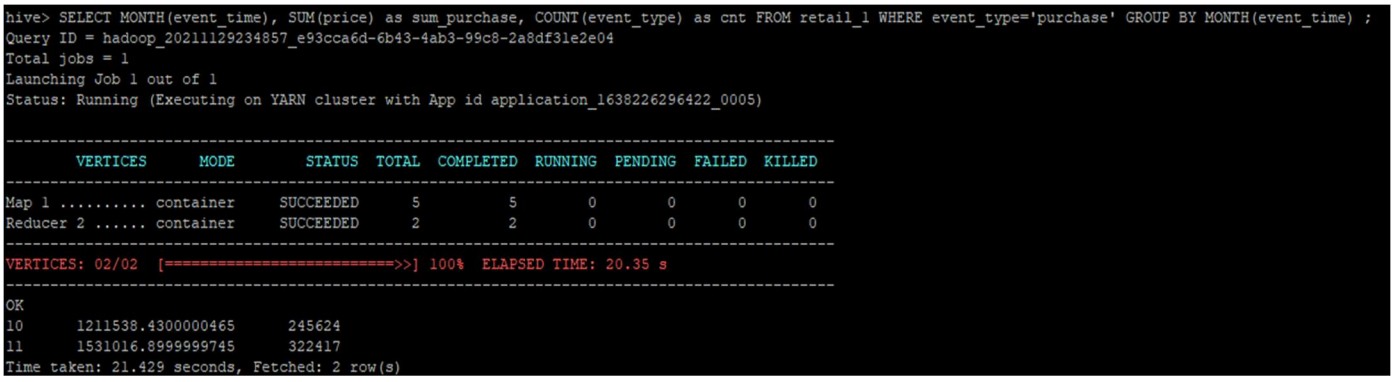
QUESTIONS

1. Find the total revenue generated due to purchases made in October.
   - SELECT SUM(price) FROM retail_1 WHERE MONTH(event_time)=10 AND event_type='purchase' ;



   - Total revenue generated in October is 1211538.43 due to purchases

2. Write a query to yield the total sum of purchases per month in a single output.
   - SELECT MONTH(event_time), SUM(price) as sum_purchase, COUNT(event_type) as cnt FROM retail_1 WHERE event_type='purchase' GROUP BY MONTH(event_time) ;



   - Clearly in the month of October total purchase is 245624, while in the month of November it goes to 322417. Hence the total purchases are increase in November. So the revenue is also increases in the month of November.

3. Write a query to find the change in revenue generated due to purchases from October to November.
   - WITH diff AS (SELECT SUM(CASE WHEN date_format(event_time,'MM')=10 THEN price ELSE 0 END) AS October, SUM(CASE WHEN date_format(event_time,'MM')=11 THEN price ELSE 0 END) AS November FROM retail_part_1 WHERE date_format(event_time,'MM') IN (10,11) AND event_type='purchase') SELECT October, November, (November - October) as Difference FROM diff ;

```
hive> WITH diff AS ( SELECT SUM(CASE WHEN date_format(event_time,'MM')=10 THEN price ELSE 0 END) AS October, SUM(CASE WHEN date_format(event_time,'MM')=11 THEN price EL
SE 0 END) AS November FROM retail_1 WHERE date_format(event_time,'MM') IN (10,11) AND event_type='purchase') SELECT October, November, (November - October) as Differenc
e FROM diff ;
Query ID = hadoop_20211129235050_956208ca-a574-429e-ac8c-dd001ccaf820
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1638226296422_0005)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      5          5        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 41.58 s
--------------------------------------------------------------------------------
OK
1211538.4300000465      1531016.8999999745      319478.469999928
Time taken: 42.315 seconds, Fetched: 1 row(s)
```

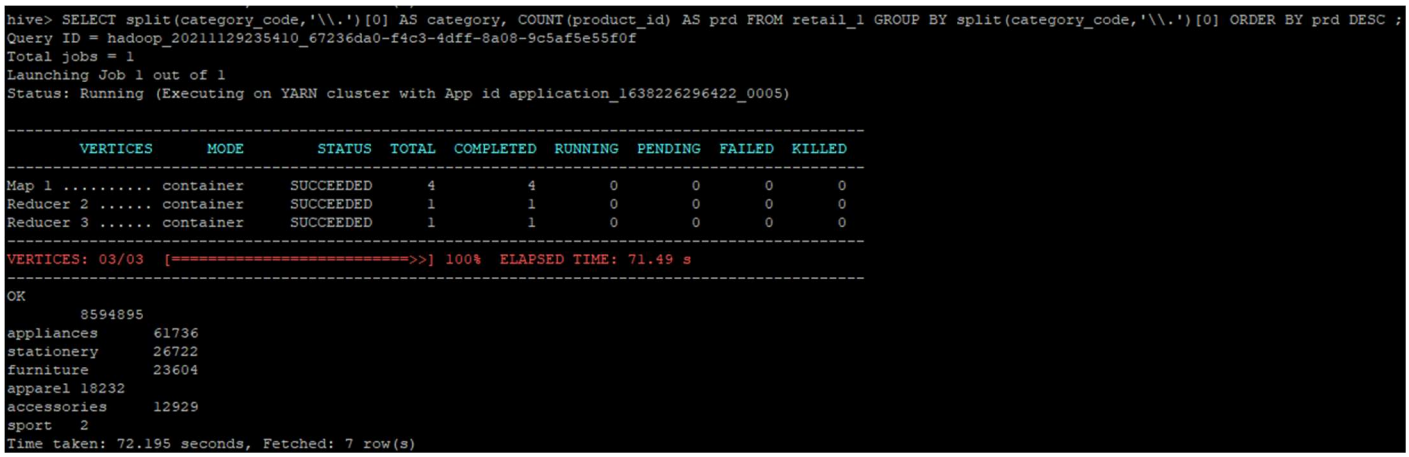
   - In the month of October total revenue is about 1211538.43 and in November it is 1531016.9, so revenue increases to about 31478.47.

4. Find distinct categories of products. Categories with null category code can be ignored.
   - SELECT DISTINCT split(category_code,'\\.')[0] AS category FROM retail_1 WHERE split(category_code,'\\.')[0]<>'' ;

```
hive> SELECT DISTINCT split(category_code,'\\.')[0] AS category FROM retail_1 WHERE split(category_code,'\\.')[0]<>'' ;
Query ID = hadoop_20211129235237_fbd23479-d1dd-4c23-9f6e-93d95286e57d
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1638226296422_0005)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      4          4        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 69.35 s
--------------------------------------------------------------------------------
OK
accessories
apparel
appliances
furniture
sport
stationery
Time taken: 70.078 seconds, Fetched: 6 row(s)
```

   - So for this there are 6 categories of products. These are “accessories”, “apparel”, “appliances”, “furniture”, “sport” and “stationery”.

5. Find the total number of products available under each category.
    - SELECT split(category_code,'\\.')[0] AS category, COUNT(product_id) AS prd FROM retail_1 GROUP BY split(category_code,'\\.')[0] ORDER BY prd DESC ;

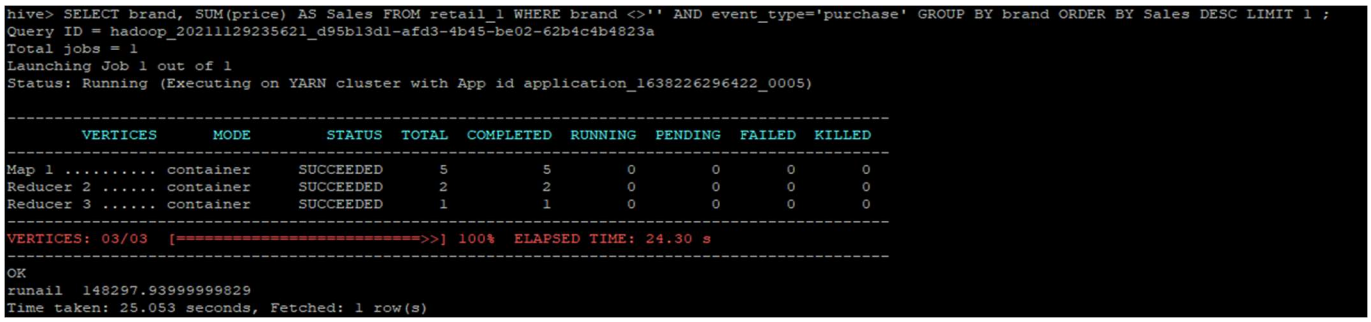


- In the categories “appliances”, “stationery”, “furniture”, “apparel”, “accessories” and “sport”.

Categories	*Number of products*
appliances	61736
stationery	26722
furniture	23604
apparel	18232
accessories	12929
sport	2

- And clearly appliance have the highest number of products while sports have only two products.

6. Which brand had the maximum sales in October and November combined?
    - SELECT brand, SUM(price) AS Sales FROM retail_1 WHERE brand <>'' AND event_type='purchase' GROUP BY brand ORDER BY Sales DESC LIMIT 1 ;



- “Runail” brand have the maximum sales in October and November combined.

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

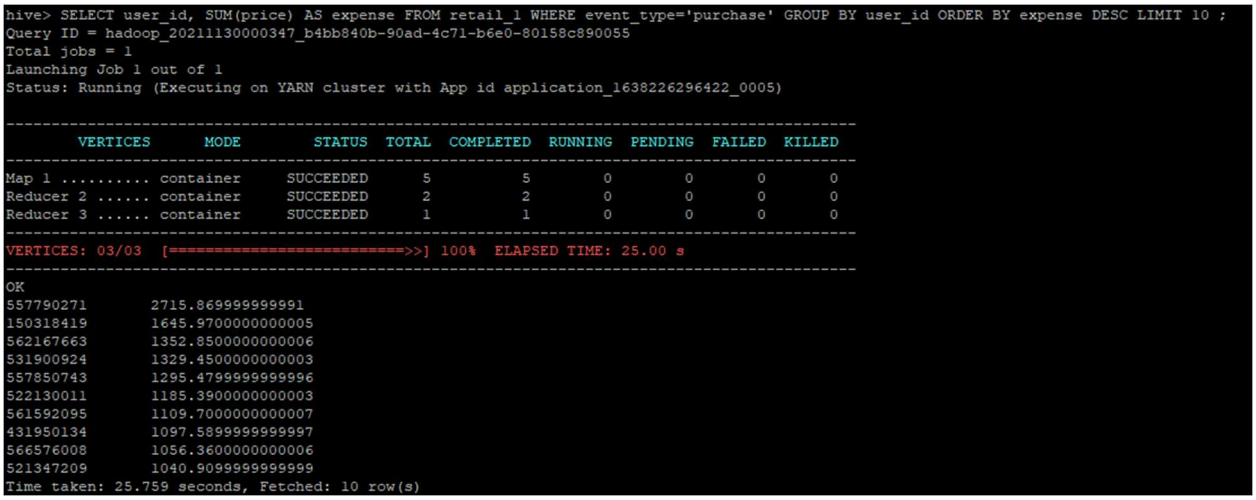
- WITH monthly_diff AS (SELECT brand, SUM(CASE WHEN date_format(event_time,'MM')=10 THEN price ELSE 0 END) AS October, SUM(CASE WHEN date_format(event_time,'MM')=11 THEN price ELSE 0 END) AS November FROM retail_1 WHERE event_type='purchase' GROUP BY brand) SELECT brand, October, November, (November-October) as Sales_diff FROM monthly_diff WHERE (November-October) >0 ORDER BY Sales_diff ;

```
hive> WITH monthly_diff AS ( SELECT brand, SUM(CASE WHEN date_format(event_time,'MM')=10 THEN price ELSE 0 END) AS October, SUM(CASE WHEN date_format(event_time,'MM')=1
1 THEN price ELSE 0 END) AS November FROM retail_1 WHERE event_type='purchase' GROUP BY brand) SELECT brand, October, November, (November-October) as Sales_diff FROM mo
nthly_diff WHERE (November-October) >0 ORDER BY Sales_diff ;
Query ID = hadoop_20211130000116_907780bd-fc94-437b-9e87-26da57bc39c4
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1638226296422_0005)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   container     SUCCEEDED      5          5        0        0       0       0
Reducer 2 ......   container     SUCCEEDED      2          2        0        0       0       0
Reducer 3 ......   container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 39.66 s
--------------------------------------------------------------------------------
OK
ovale   2.54    3.1     0.56
cosima  20.229999999999997      20.93   0.7000000000000028
grace   100.91999999999999      102.61000000000001      1.6900000000000261
```

```
swarovski       1887.9300000000003      3043.159999999997       1155.2299999999968
beauty-free     554.17  1782.8599999999988      1228.6899999999987
zeitun  708.66  2009.6299999999997       1300.9699999999998
joico   705.52  2015.1  1309.58
severina        4775.879999999992       6120.479999999991       1344.5999999999995
irisk   45591.95999999895       46946.03999999963       1354.0800000006784
oniq    8425.410000000003       9841.650000000003       1416.2399999999998
levrana 2243.5600000000004      3664.099999999999       1420.5399999999986
roubloff        3491.3600000000015      4913.769999999998       1422.4099999999962
smart   4457.260000000002       5902.140000000002       1444.88
shik    3341.2000000000007      4839.720000000001       1498.5200000000004
domix   10472.050000000017      12009.170000000004      1537.1199999999862
artex   2730.6399999999994      4327.249999999996       1596.609999999997
beautix 10493.949999999992      12222.949999999999      1729.0000000000073
milv    3904.9399999999914      5642.009999999992       1737.0700000000006
masura  31266.079999998878      33058.46999999912       1792.3900000002432
f.o.x   6624.229999999996       8577.279999999988       1953.049999999992
kapous  11927.159999999969      14093.079999999976      2165.9200000000073
concept 11032.14000000002       13380.400000000012      2348.259999999993
estel   21756.749999999978      24142.66999999998       2385.920000000002
kaypro  881.34  3268.7  2387.3599999999997
benovy  409.6199999999999       3259.970000000001       2850.3500000000013
italwax 21940.239999999965      24799.369999999933      2859.1299999999683
yoko    8756.909999999993       11707.879999999994      2950.9700000000001
haruyama        9390.690000000042       12352.910000000025      2962.219999999983
marathon        7280.749999999999       10273.1 2992.3500000000013
lovely  8704.379999999997       11939.059999999996      3234.6799999999985
bpw.style       11572.149999999867      14837.439999999828      3265.289999999961
staleks 8519.730000000003       11875.610000000006      3355.880000000003
freedecor       3421.7799999999897      7671.800000000008       4250.020000000019
runail  71539.28000000035       76758.66000000063       5219.380000000281
polarus 6013.719999999999       11371.93        5358.210000000001
cosmoprofi      8322.809999999998       14536.989999999983      6214.179999999986
jessnail        26287.839999999964      33345.22999999998       7057.390000000018
strong  29196.629999999994      38671.26999999999       9474.639999999996
ingarden        23161.38999999999       33566.20999999994       10404.819999999952
lianail 5892.8399999999965      16394.239999999932      10501.399999999936
uno     35302.02999999978       51039.7499999999        15737.720000000118
grattol 35445.53999999994       71472.70999999972       36027.16999999977
        474679.06000000564      619509.2400000023       144830.17999999668
Time taken: 40.437 seconds, Fetched: 161 row(s)
hive>
```

- 161 brands increased their sales from October to November

8. 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.
    - SELECT user_id, SUM(price) AS expense FROM retail_1 WHERE event_type='purchase' GROUP BY user_id ORDER BY expense DESC LIMIT 10 ;



```
hive> SELECT user_id, SUM(price) AS expense FROM retail_1 WHERE event_type='purchase' GROUP BY user_id ORDER BY expense DESC LIMIT 10 ;
Query ID = hadoop_20211130000347_b4bb840b-90ad-4c71-b6e0-80158c890055
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1638226296422_0005)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   container     SUCCEEDED      5          5        0        0       0       0
Reducer 2 ......   container     SUCCEEDED      2          2        0        0       0       0
Reducer 3 ......   container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 25.00 s
--------------------------------------------------------------------------------
OK
557790271       2715.869999999991
150318419       1645.9700000000005
562167663       1352.8500000000006
531900924       1329.4500000000003
557850743       1295.4799999999996
522130011       1185.3900000000003
561592095       1109.7000000000007
431950134       1097.5899999999997
566576008       1056.3600000000006
521347209       1040.9099999999999
Time taken: 25.759 seconds, Fetched: 10 row(s)
```

These top 10 users who spend the most. So They should be awarded.

Closing Query and Analysis

- First we drop the database from the hive.

```
hive> show databases  ;
OK
default
new_assignment
Time taken: 0.044 seconds, Fetched: 2 row(s)
hive> DROP DATABASE new_assignment ;
OK
Time taken: 0.111 seconds
hive> show databases ;
OK
default
Time taken: 0.024 seconds, Fetched: 1 row(s)
hive>
```

- And then we terminated cluster.