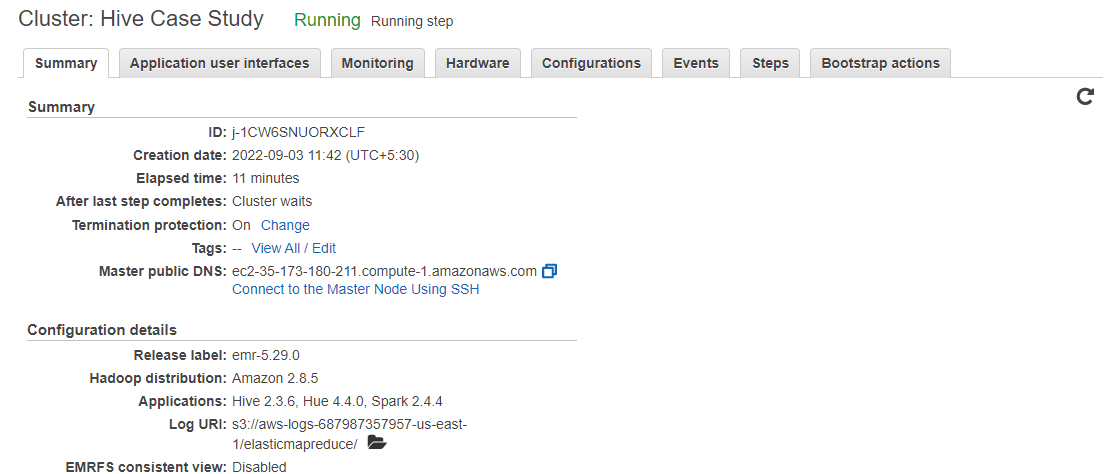
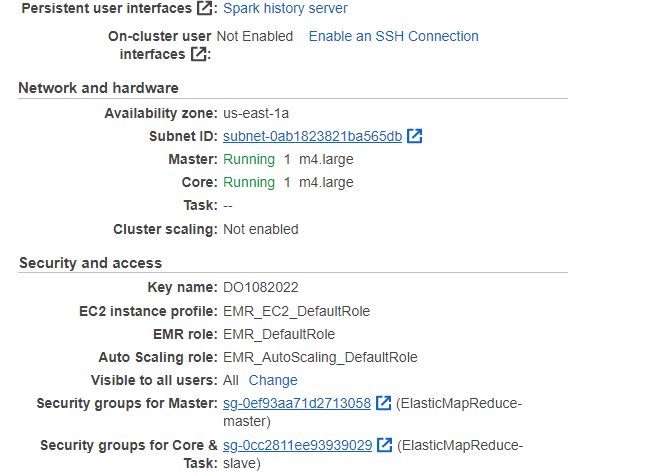
**HIVE CASE STUDY**

Parag Ranjan Behera

Creating Hive Case Study:

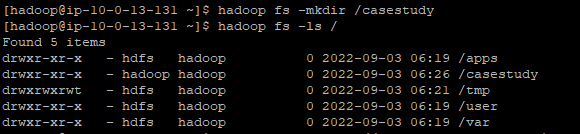
Step 1: Creating EMR Cluster





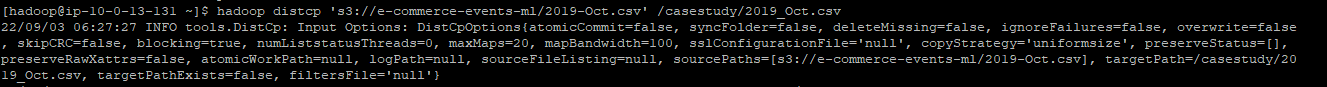
**Hadoop & Hive queries:-**

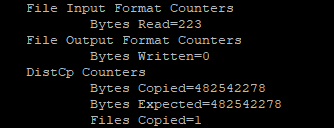
Creating a directory **“casestudy”**



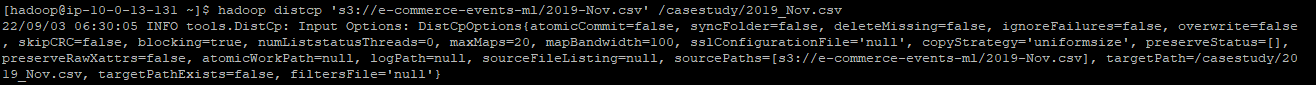
**Loading the datasets into HDFS from S3:**

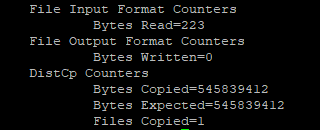
hadoop distcp 's3://e-commerce-events-ml/2019-Oct.csv' /casestudy/2019\_Oct.csv





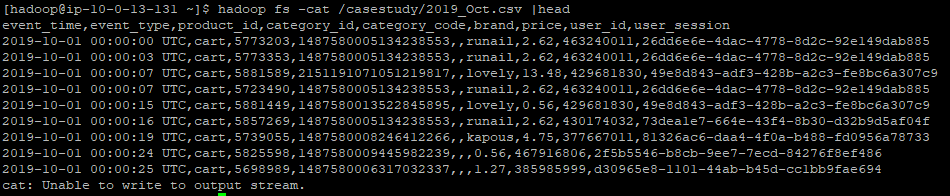
hadoop distcp 's3://e-commerce-events-ml/2019-Nov.csv' /casestudy/2019\_Nov.csv



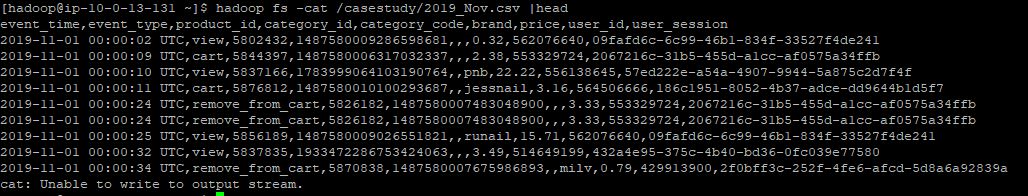


**Viewing the data**

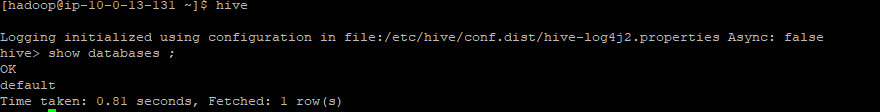
hadoop fs -cat /casestudy/2019\_Oct.csv | head



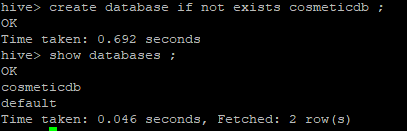
hadoop fs -cat /casestudy/2019\_Nov.csv | head

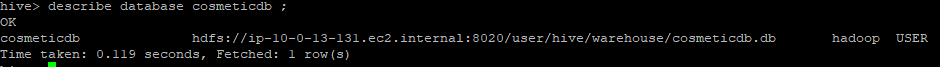


**Launching Hive:-**



**Creating a new Database “cosmeticdb” :-**

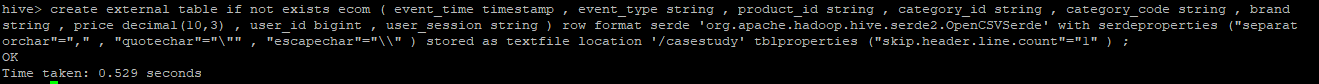




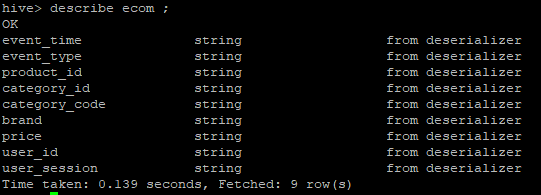
**Creating new table “ecom” :-**

hive > create external table if not exists ecom (event\_time timestamp, event\_type string,

product\_id string, category\_id string, category\_code string, brand string, price 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") ;



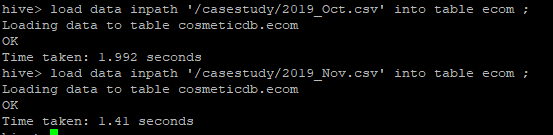
**hive> describe ecom ;**



**Loading data into table “ecom”:-**

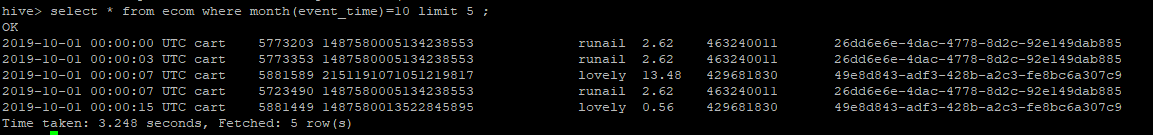
hive> load data inpath '/casestudy/2019\_Oct.csv' INTO TABLE ecom ;

hive> load data inpath '/casestudy/2019\_Nov.csv' INTO TABLE ecom;

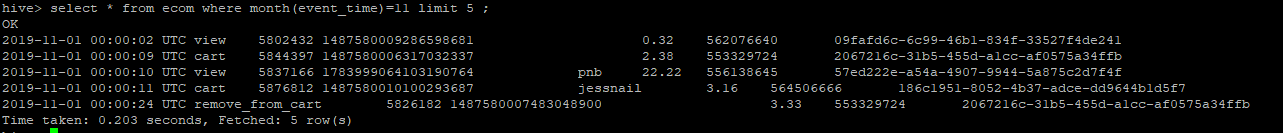


**Performing data check:**

hive> select \* from ecom where month(event\_time)=10 limit 5 ;



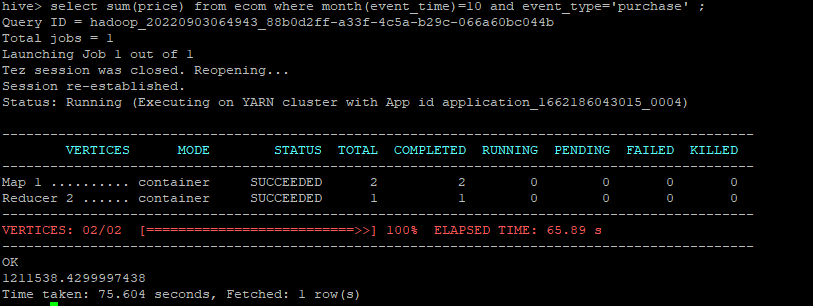
hive> select \* from ecom where month(event\_time)=11 limit 5 ;



**QUESTION 1:**

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

hive> select sum(price) from ecom where month(event\_time)=10 and event\_type='purchase' ;



Time Taken to execute the above query is 75.6 sec.

This is very high. Hence, to reduce this execution time, we will dynamically partition the table “ecom” and add buckets to create an optimised table.

**DYNAMIC PARTITIONING**

hive> set hive.exec.dynamic.partition=true;

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

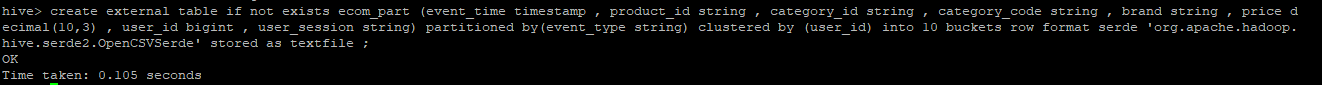
hive> set hive.enforce.bucketing=true ;



**PARTITION TABLE : ecom\_part**

Partition on : event\_type (there are 4 types and all questions are related to ‘purchase’)

hive> create external table if not exists ecom\_part (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 10 buckets row format serde 'org.apache.hadoop.hive.serde2.OpenCSVSerde' stored as textfile ;

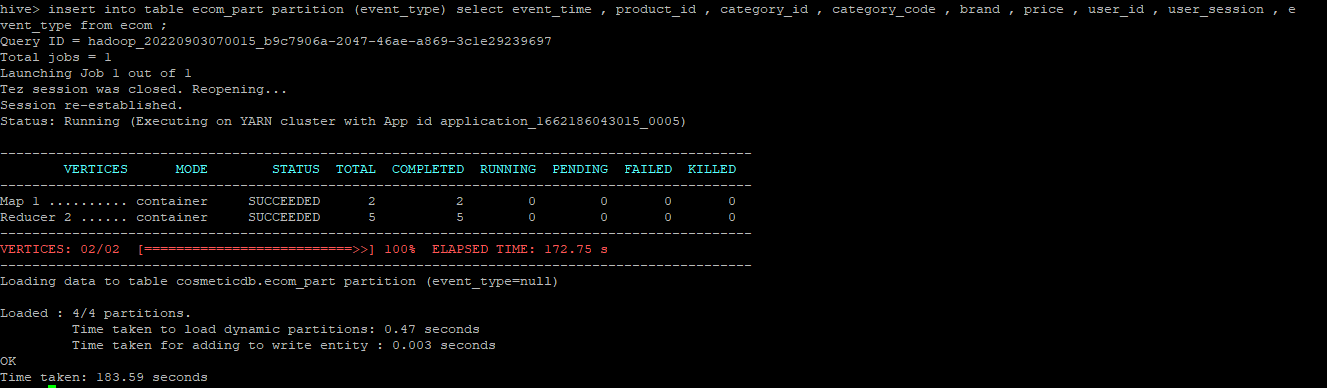


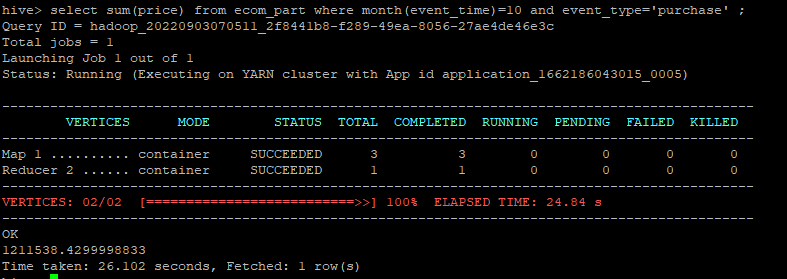
Hive> Describe ecom\_part ;



hive> insert into table ecom\_part partition (event\_type) select event\_time, product\_id,

category\_id, category\_code, brand, price, user\_id, user\_session, event\_type from ecom ;

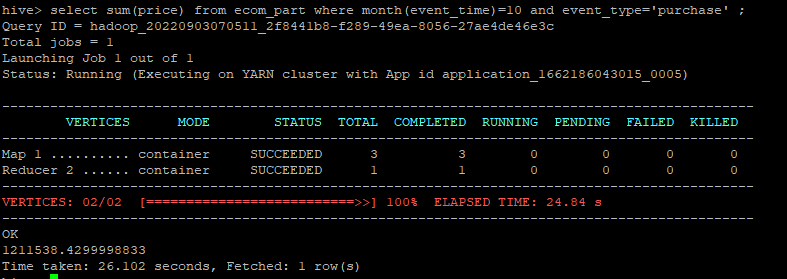




After partitioning the ecom table, the query time significantly improves to 26.1 sec. So we will use ecom\_part table for other queries as well.

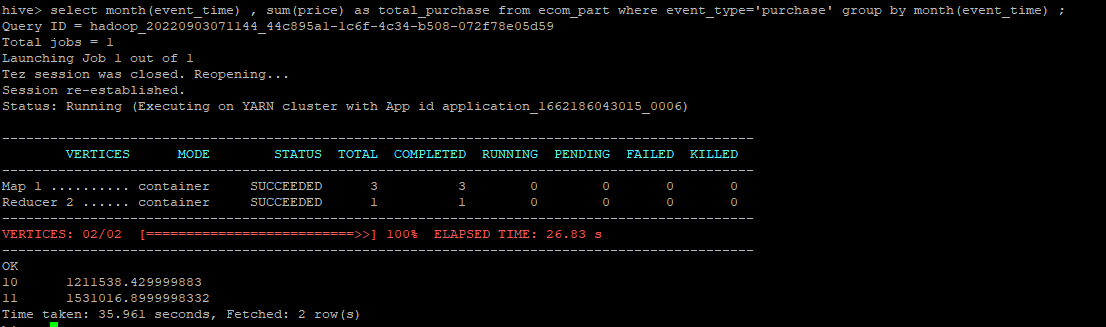
**Q1. Find the total revenue generated due to purchases made in October.**

hive> select sum(price) from ecom where month(event\_time)=10 and event\_type='purchase' ;



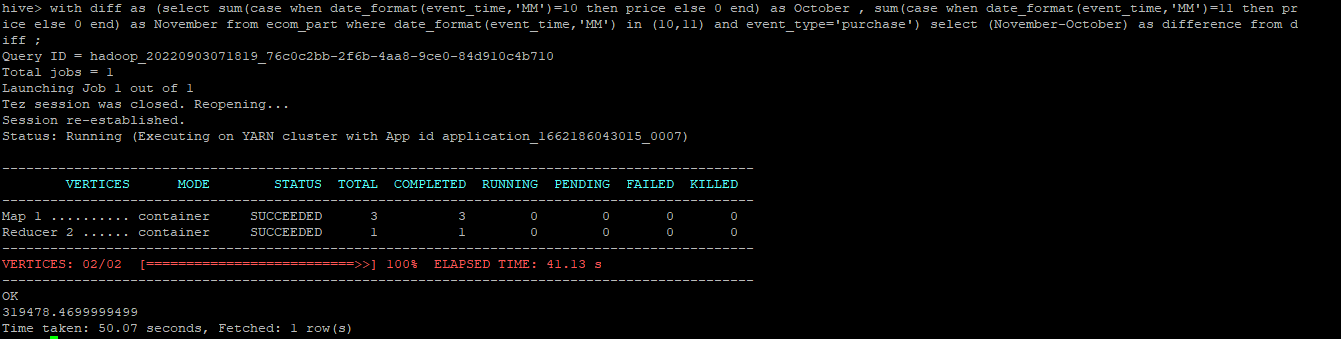
**Q2. Write a query to yield the total sum of purchases per month in a single output.**

Hive> select month(event\_time) , sum(price) as total\_purchase from ecom\_part where event\_type=’purchase’ group by month(event\_time) ;



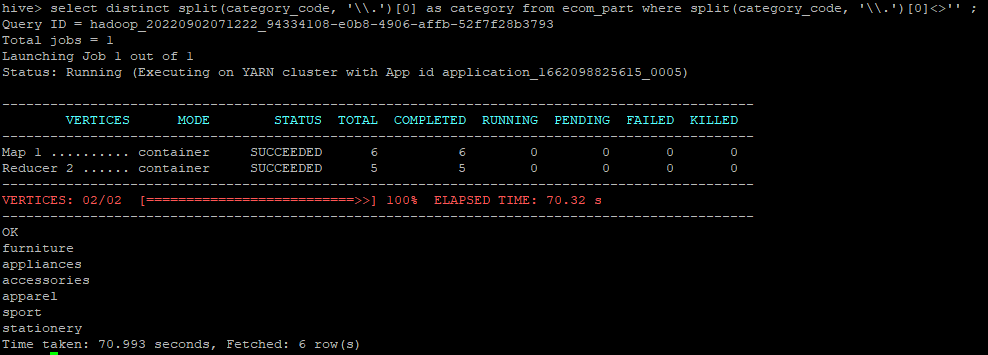
**Q3. Write a query to find the change in revenue generated due to purchases from October to November.**

Hive> 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 ecom\_part where date\_format(event\_time ,’MM’) in (10,11) and event\_type=’purchase’) select (November-October) as difference from diff ;



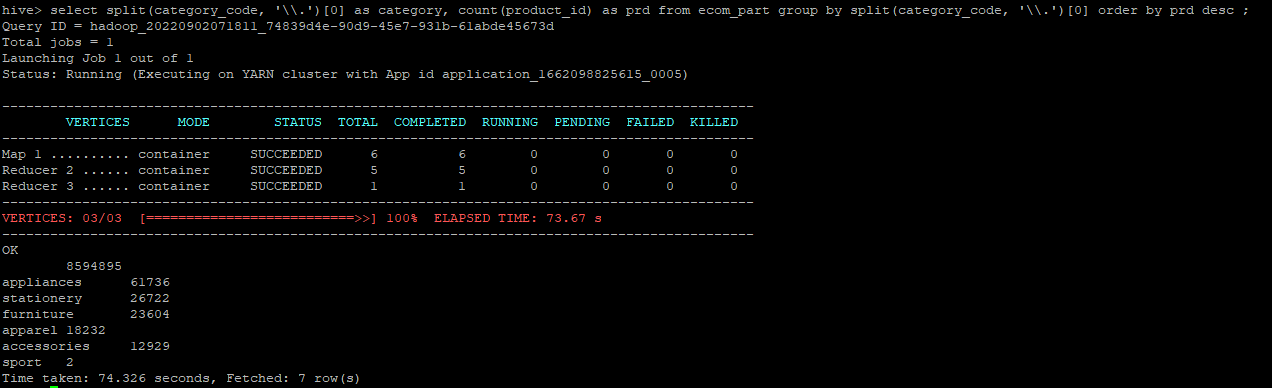
**Q4. Find distinct categories of products. Categories with null category code can be ignored.**

Hive> select distinct split(category\_code, ‘\\.’)[0] as category from ecom\_part where split(category\_code, ‘\\.’)[0]<>’’ ;



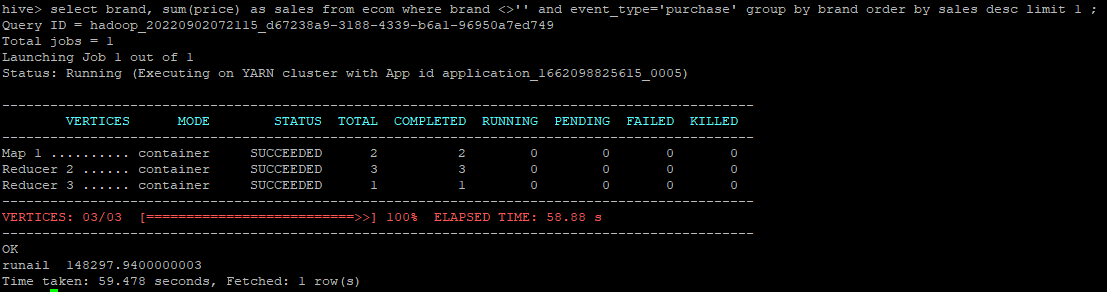
**Q5. Find the total number of products available under each category.**

Hive> select split(category\_code, ‘\\.’)[0] as category, count(product\_id) as prd from ecom\_part group by split(category\_code, ‘\\.’)[0] order by prd desc ;



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

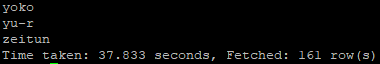
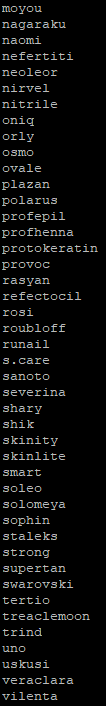
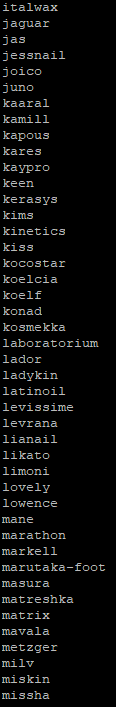
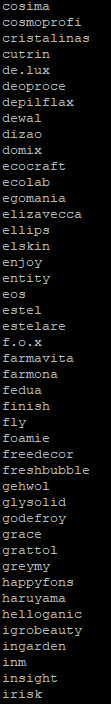
Hive> select brand, sum(price) as sales from ecom\_part where brand <>'' and event\_type='purchase' group by brand order by sales desc limit 1 ;



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

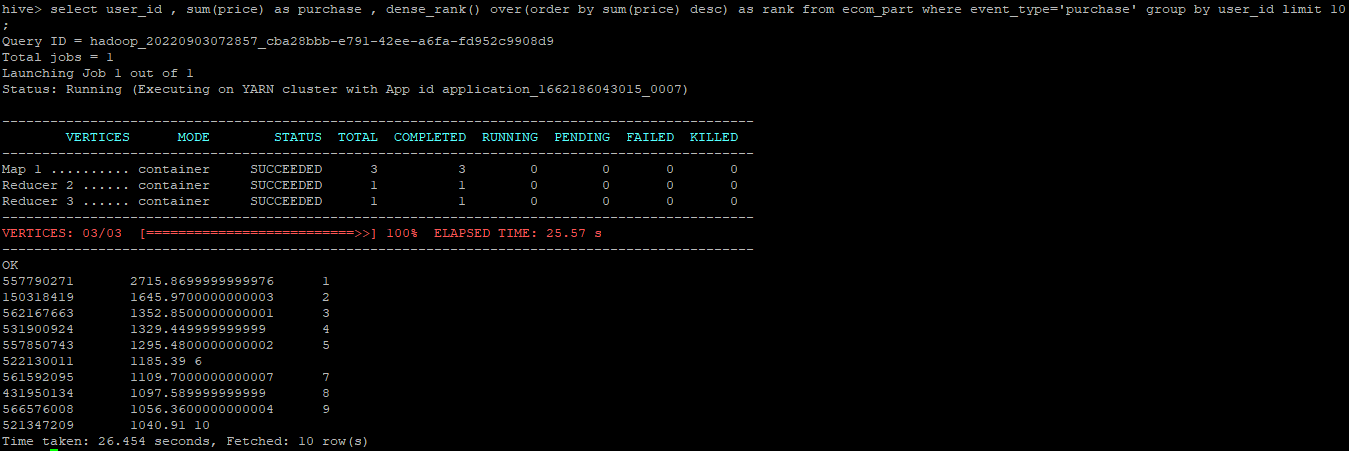
Hive>with monthly\_diff as ( select brand, sum(case when date\_format(event\_time,'MM')=10 then price else 0 end) as October, sum(case when date\_format(event\_time,'MM')=11 then price else 0 end) as November from ecom\_part where event\_type=‘purchase' group by brand) select brand from monthly\_diff where (November-October) >0 ;





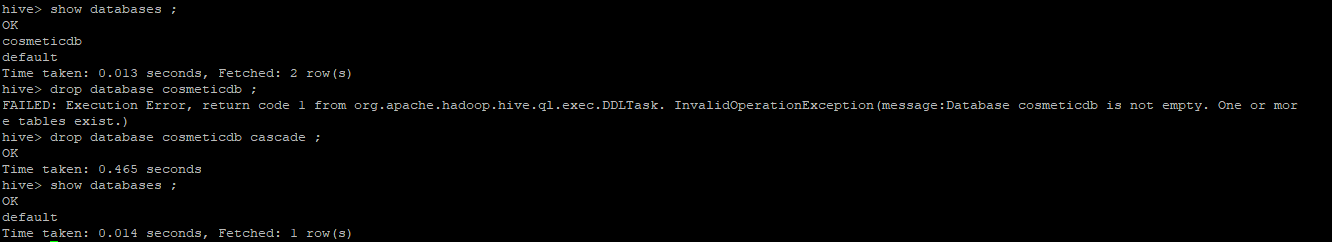
**Q8. Your company wants to reward the top 10 users of its website with a Golden Customer plan. Write a query to generate a list of top 10 users who spend the most.**

Hive> select user\_id , sum(price) as purchase, dense\_rank() over(order by sum(price) desc) as rank from ecom\_part where event\_type=’purchase’ group by user\_id limit 10 ;



**Cleaning Up:**

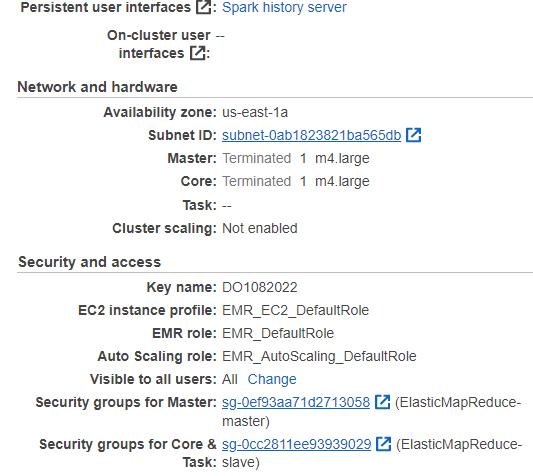
Once the analysis is completed, deleting the database and exit the hive.





Terminating the EMR Cluster:-





**Observations:**

1. The performance wise Partition is effective for low volume data. In our data set also we observed that performance rate increased when we use partitioning.
2. For larger data creating a bucketing gives you 2-3x better query performance than a non- bucket table.

**Insights:**

* 1. Depend on the data views and cart event type are more compared to purchase ones.
  2. The total revenue is high in November month than October month.
  3. Highest number of products available under appliances , stationary category.
  4. Runail brand has highest sales compared with other brands.
  5. Over all 43% brands increased their sales from October to November.
  6. The user\_id: 557790271 spent most in two months.