Hive Case Study- DA Track

Submitted by
1. Sukeerthi G
2. Shubham Aggarwal

E-Commerce Hive Case Study

Problem Statement:

With online sales gaining popularity, tech companies are exploring ways to improve their sales by analyzing customer behavior and gaining insights about product trends. Furthermore, the websites make it easier for customers to find the products they require without much scavenging. Needless to say, the role of big data analysts is among the most sought-after job profiles of this decade. Therefore, as part of this assignment, it is expected to extract data and gather insights from a real-life data set of an e-commerce company.

Dataset:

The dataset used for this case study is available in the following links,

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

Attribute Description:

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 every time the user returns back the website after a long pause

Table 1 Attribute Description

Importing the data to HDFS:

Step 1 : Creating S3 bucket and loading the dataset:

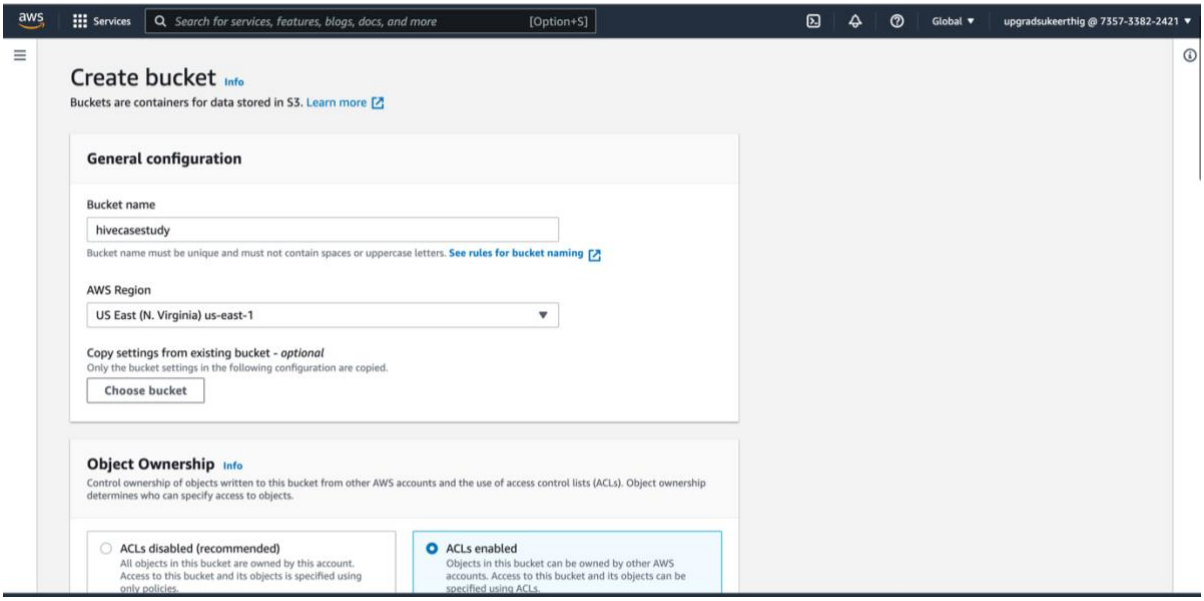

Figure 1 Bucket Creation

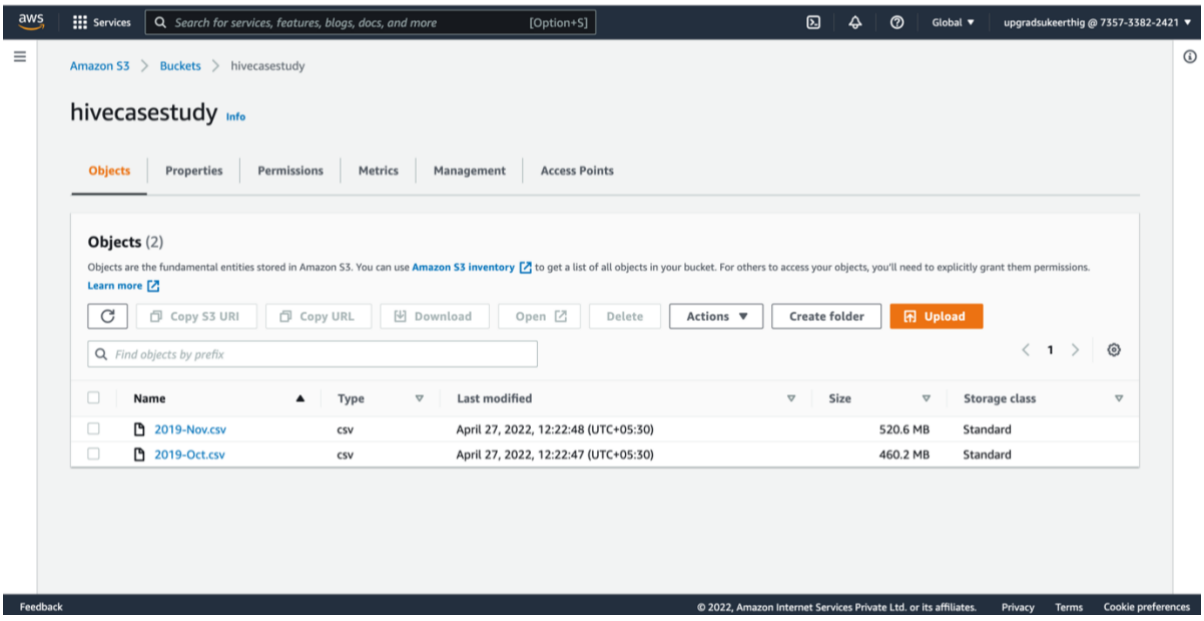

Figure 2 Uploading data into the S3 bucket

Step 2: Creation of Key-Pair:

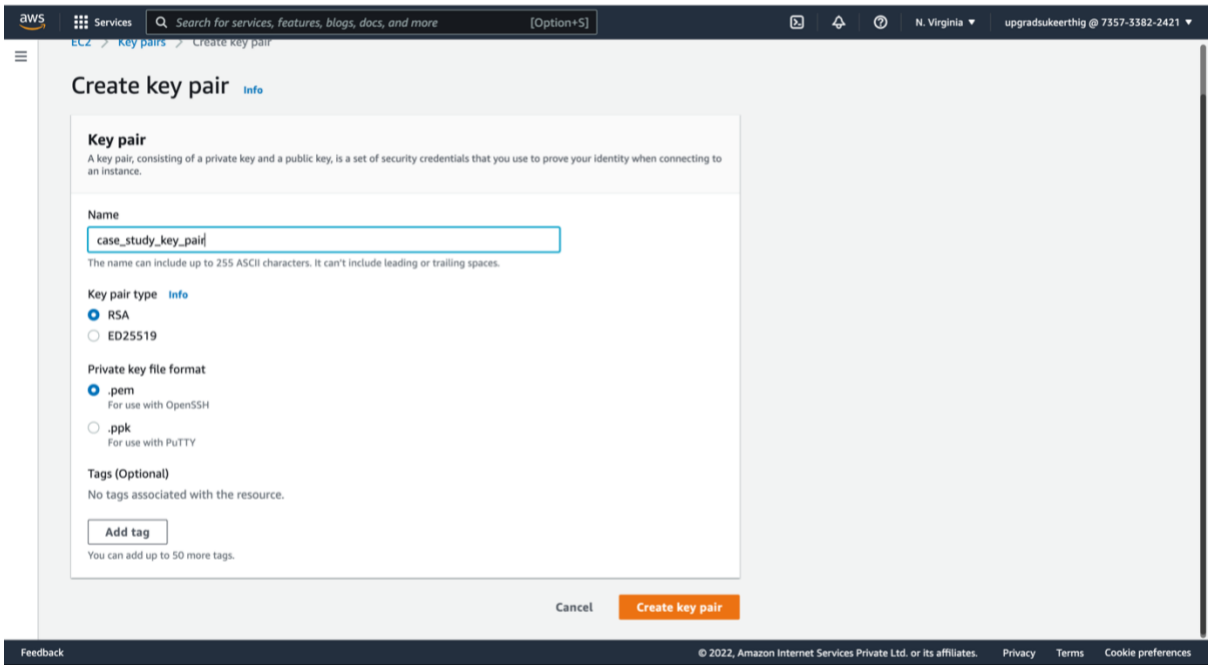

Figure 3 Key-pair configurations

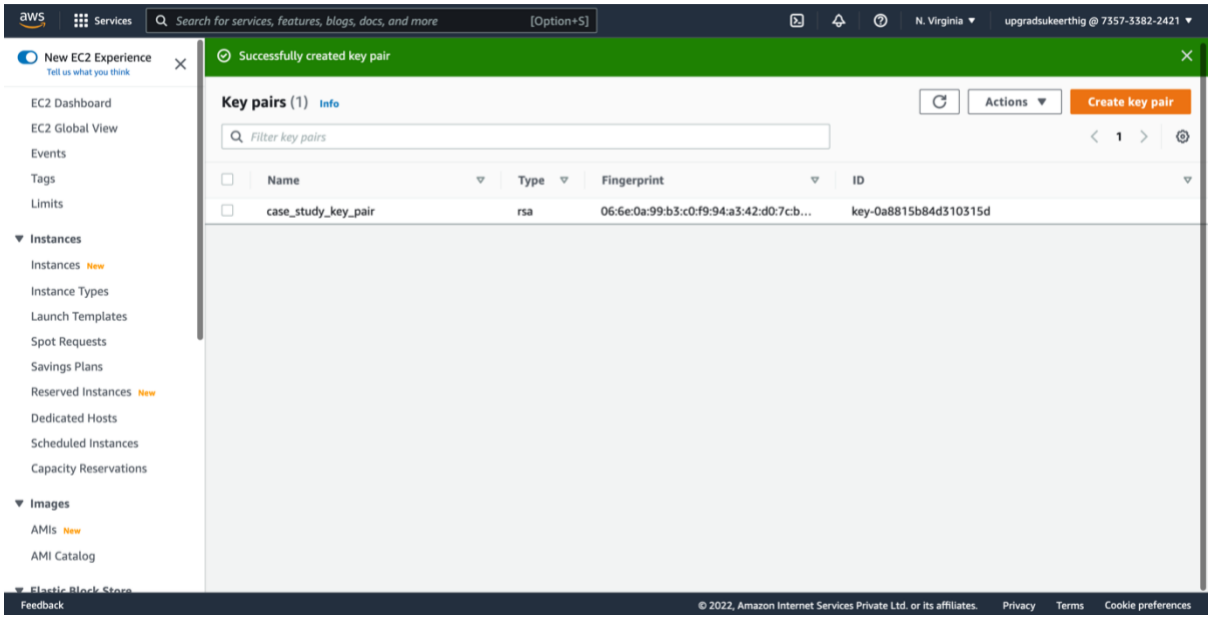

Figure 4 Key-Pair creation

Step 3: Creation of EMR Cluster:

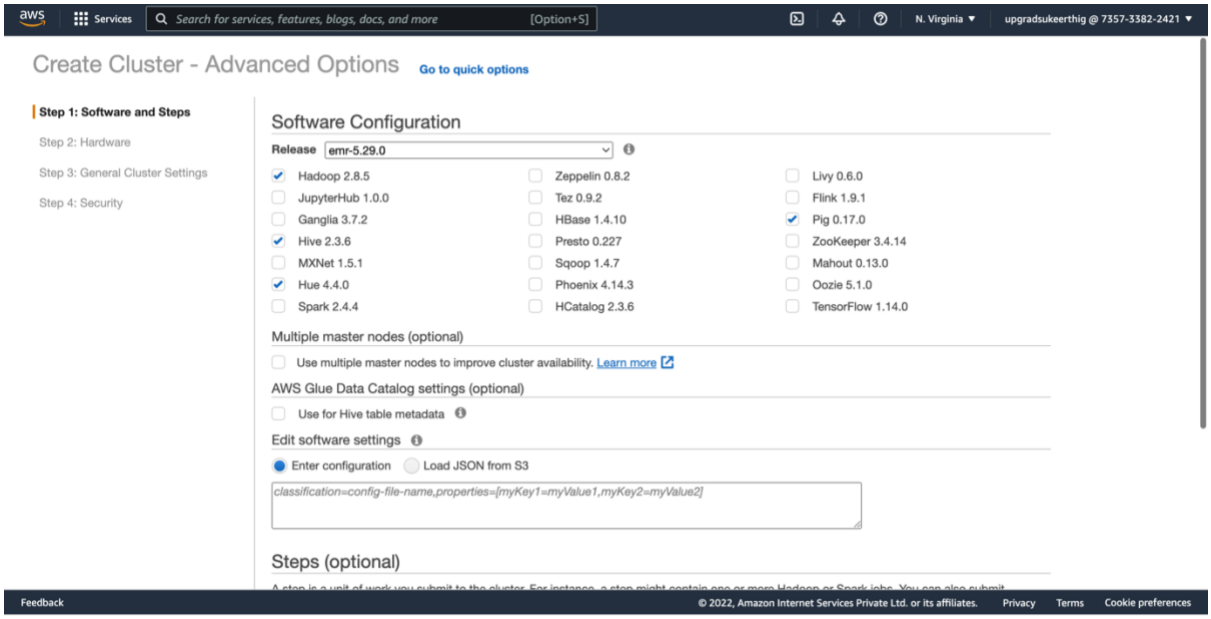

Figure 5 Software Configuration

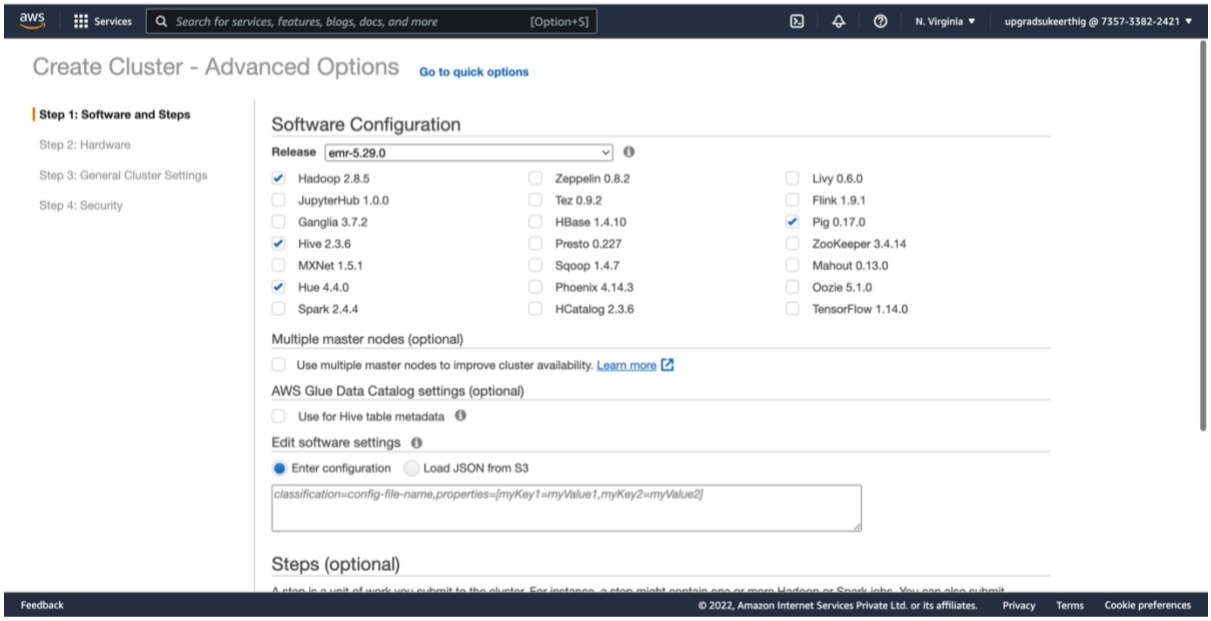

Figure 6 Advanced Options

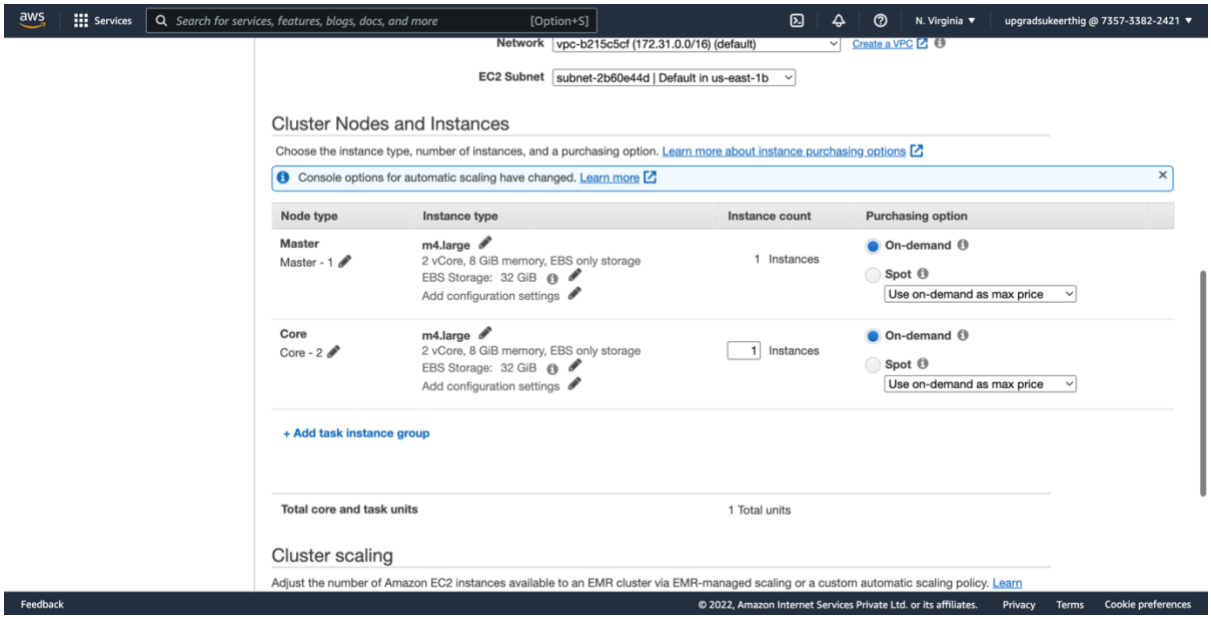

Figure 7 Cluster node hardware settings

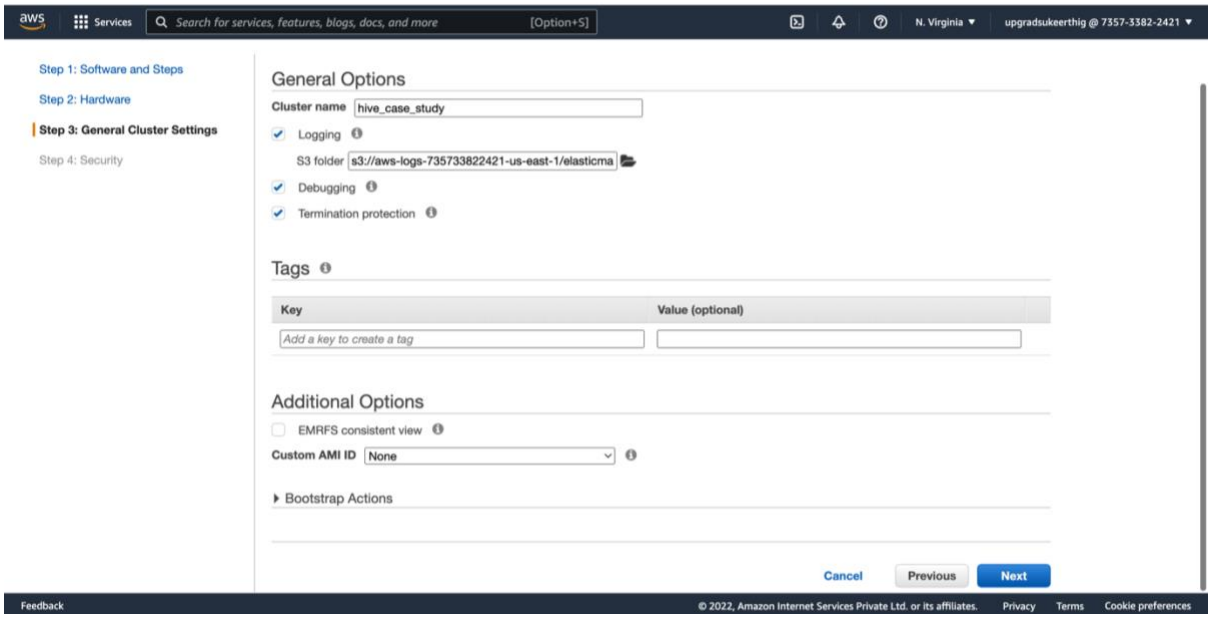

Figure 8 General cluster settings

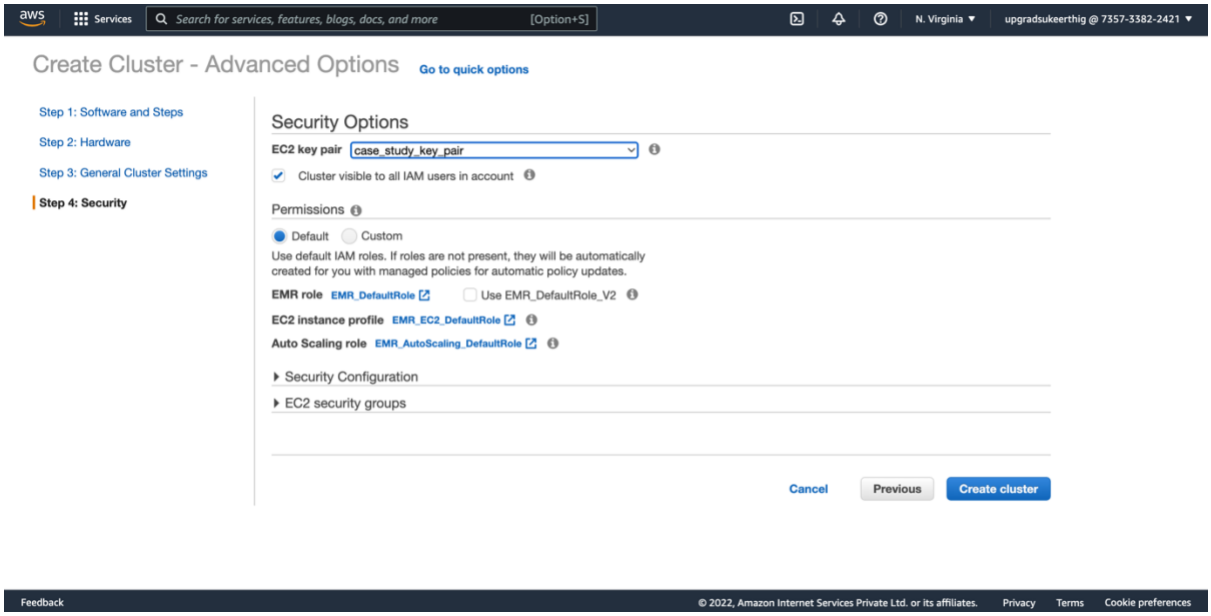

Figure 9 Security Options

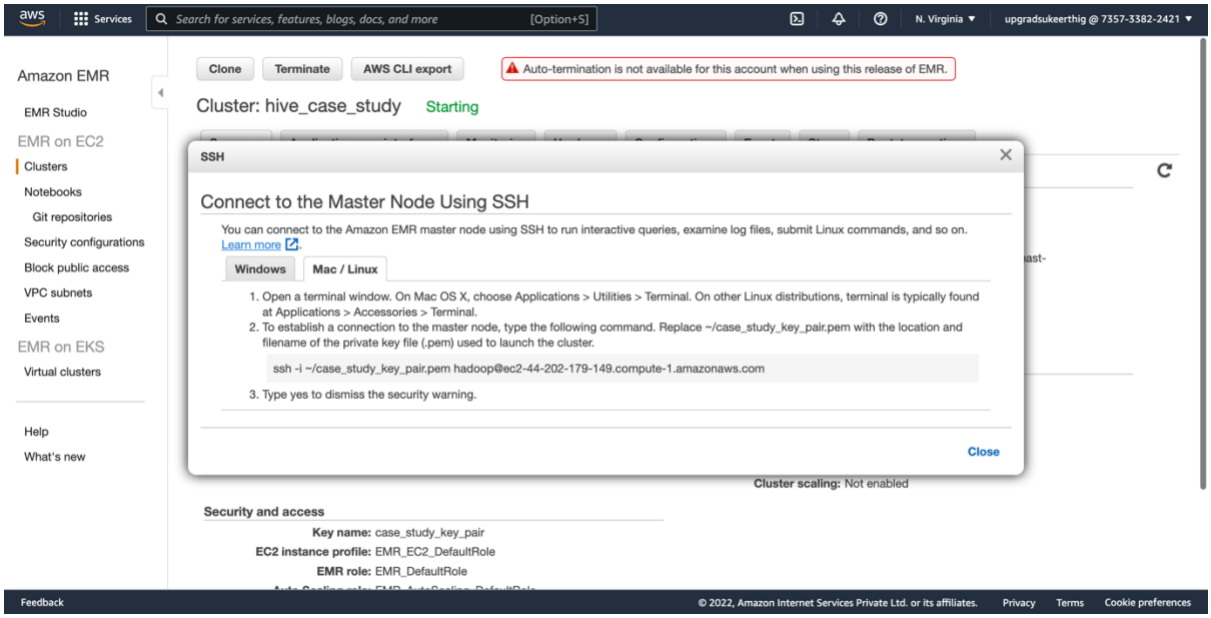

Figure 10 SSH command for connecting with the master node

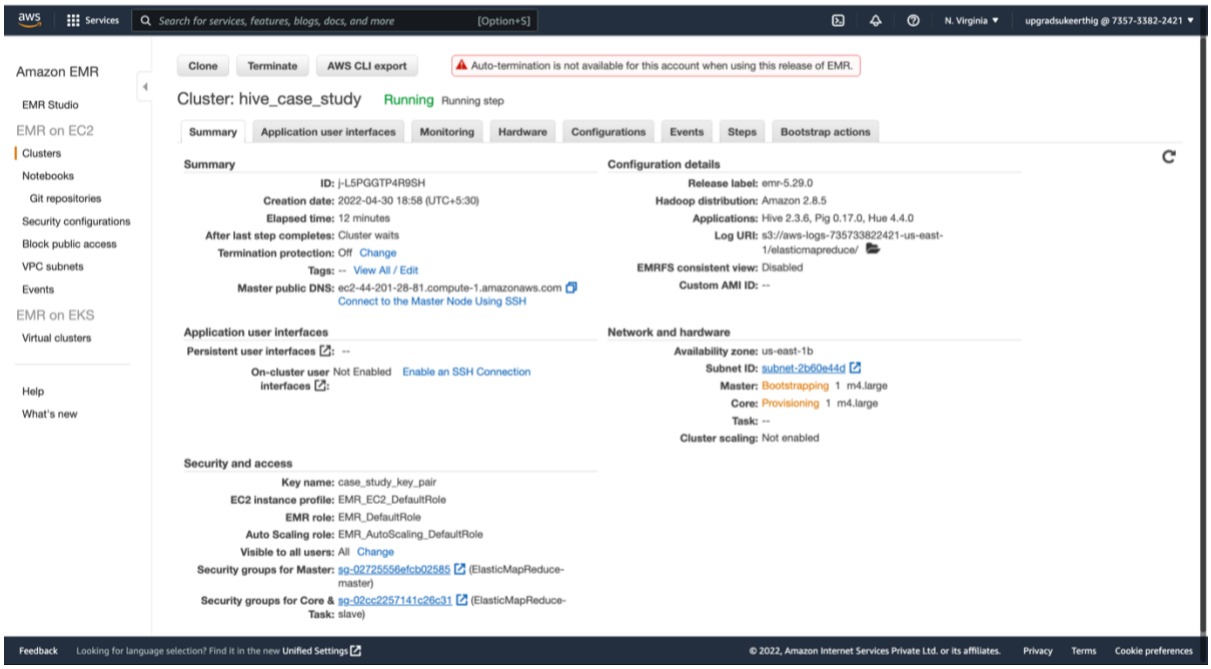

Figure 11 Cluster running

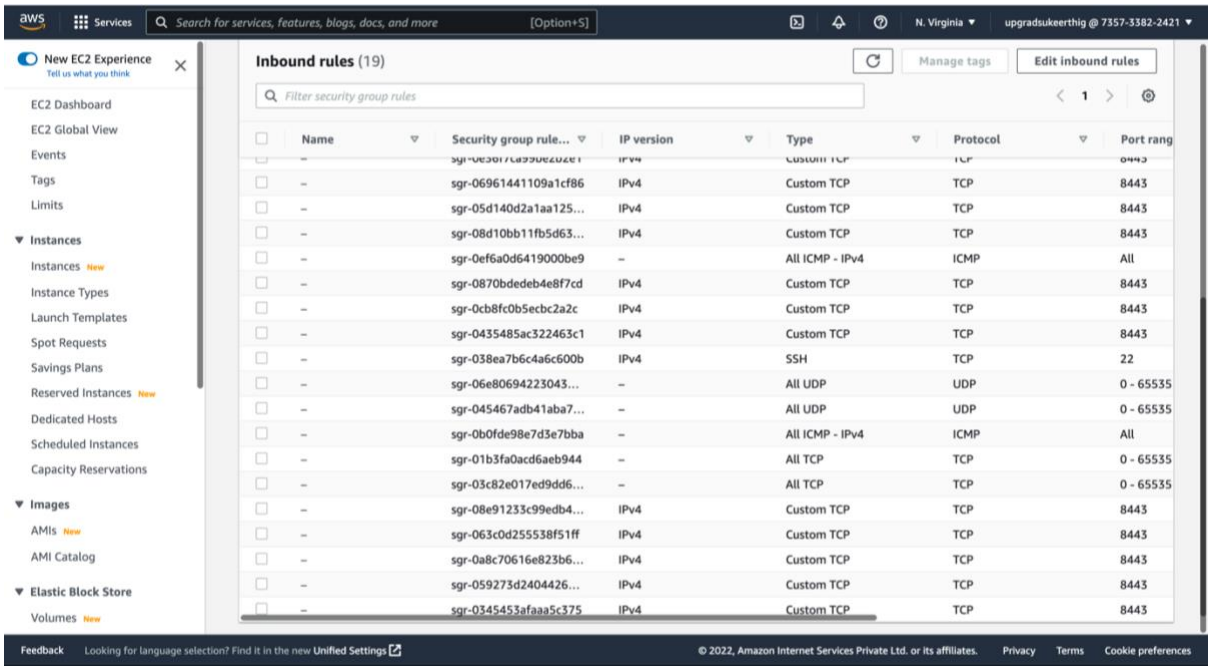

Figure 12 Inbound rules already updated to SSH with TCP at port 22 in the past

Step 4: Connecting to the master node through terminal window

Command:

cd /Users/sukeerthig/Downloads

chmod 400 case_study_key_pair.pem

ssh -i case_study_key_pair.pem hadoop@ec2-44-201-28-81.compute-1.amazonaws.com

Figure 13 Successfully connecting to the master node

Command:

hadoop fs -mkdir /hive_case_study

hadoop fs -ls /

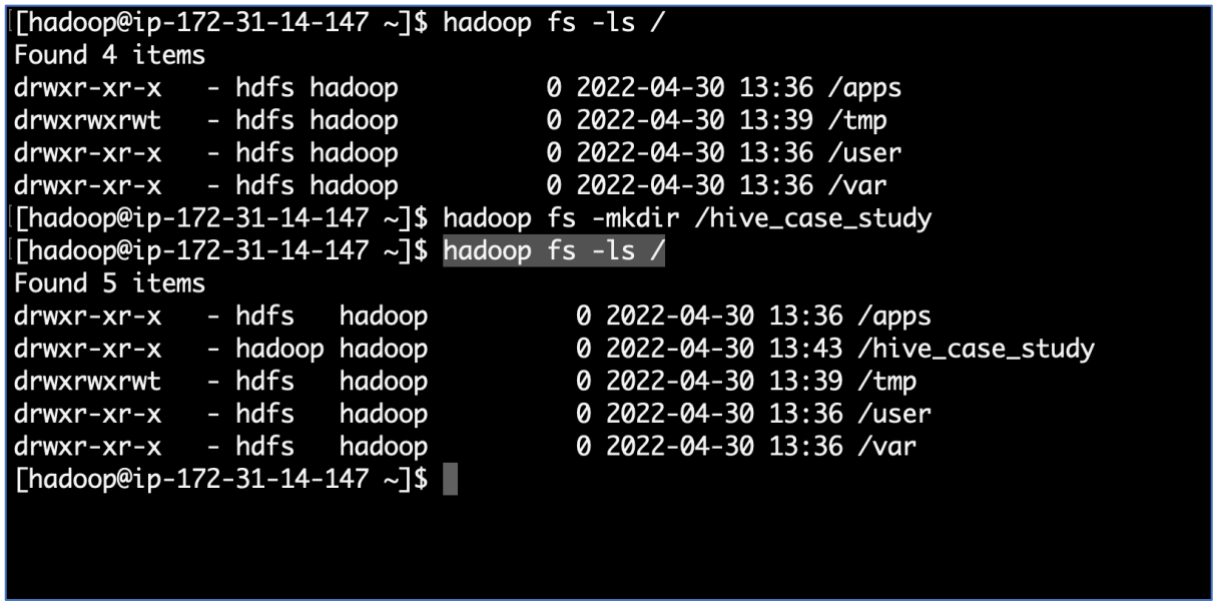

Figure 14 Creating directory for the case study

Command:

hadoop distcp s3://hivecasestudy/2019-Nov.csv /hive_case_study

hadoop distcp s3://hivecasestudy/2019-Oct.csv /hive_case_study

hadoop fs -ls /hive_case_study

```
[hadoop@ip-172-31-14-147 ~]$ hadoop fs -ls /hive_case_study
Found 2 items
-rw-r--r--   1 hadoop hadoop  545839412 2022-04-30 13:51 /hive_case_study/2019-Nov.csv
-rw-r--r--   1 hadoop hadoop  482542278 2022-04-30 13:52 /hive_case_study/2019-Oct.csv
[hadoop@ip-172-31-14-147 ~]$
```

Figure 15 Moving the data from S3 bucket to HDFS

Creating Suitable Hive tables:

Step 1: Creation of database

Command:

CREATE DATABASE IF NOT EXISTS clickstream_data ;

SHOW DATABASES ;

```
[hadoop@ip-172-31-14-147 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive> SHOW DATABASES ;
OK
default
Time taken: 1.087 seconds, Fetched: 1 row(s)
hive> CREATE DATABASE IF NOT EXISTS clickstream_data ;
OK
Time taken: 0.783 seconds
hive> SHOW DATABASES ;
OK
clickstream_data
default
Time taken: 0.023 seconds, Fetched: 2 row(s)
hive>
```

Figure 16 Database creation

Step 2: Creating and inserting the data into the table

Command:

USE clickstream_data ;

CREATE TABLE IF NOT EXISTS ecommerce (event_time timestamp, event_type string, product_id string, category_id string, category_code string, brand string, price float, user_id bigint, user_session string) ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS TEXTFILE LOCATION '/hive_case_study' TBLPROPERTIES ('serialization.null.format'='','skip.header.line.count'='1');

SET hive.cli.print.header=true;

SELECT * FROM ecommerce LIMIT 5 ;

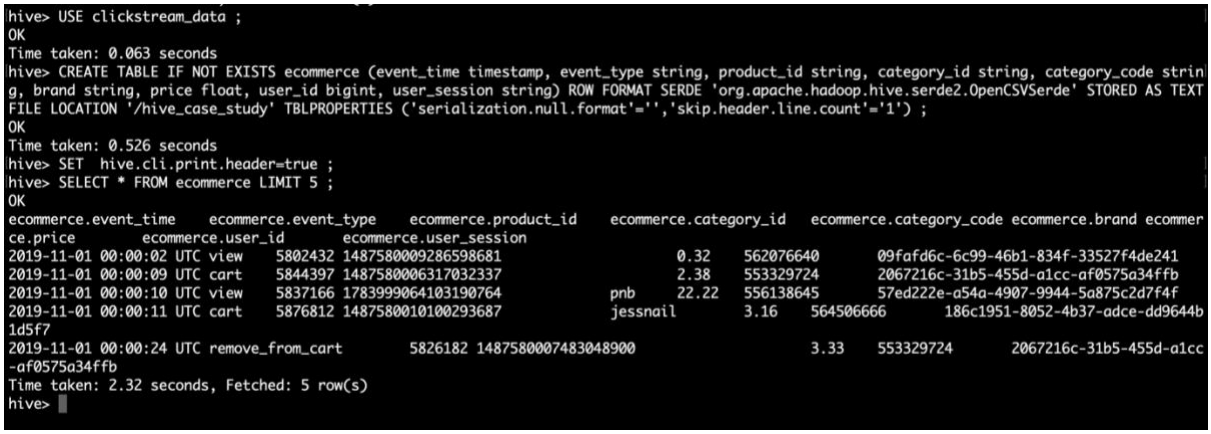

Figure 17 Creating and uploading data into the table

Optimization Techniques

Command:

SET hive.exec.dynamic.partition =true;

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

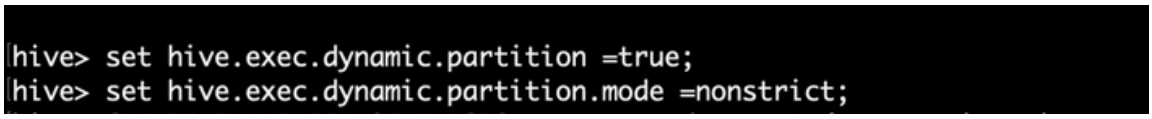

Figure 18 Creation of partitioned table based on event_type

Step 1 : Creating partitioned and bucketed table and inserting data into the table:

For a lesser querying time this table was created with partition done based on ‘event_type’ and clustered/ bucketed based on price.

Command:

CREATE TABLE IF NOT EXISTS ecommerce_bucketed (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 (price) INTO 30 BUCKETS ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde';

INSERT INTO TABLE ecommerce_bucketed PARTITION(event_type) SELECT event_time, product_id, category_id, category_code, brand,price, user_id, user_session, event_type FROM ecommerce ;

```
hive> CREATE TABLE IF NOT EXISTS ecommerce_bucketed (event_time timestamp, product_id string, category_id string, category_code string, brand s
tring, price float, user_id bigint, user_session string) PARTITIONED BY (event_type string) CLUSTERED BY (price) INTO 30 BUCKETS ROW FORMAT SER
DE 'org.apache.hadoop.hive.serde2.OpenCSVSerde';
OK
Time taken: 0.121 seconds
hive> INSERT INTO TABLE ecommerce_bucketed PARTITION(event_type) SELECT event_time, product_id, category_id, category_code, brand,price, user_i
d, user_session, event_type  FROM ecommerce ;
Query ID = hadoop_20220430151100_ee4ec6a5-5cc3-4367-87e9-e1ea01c8c39c
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_1651325890766_0006)

--------------------------------------------------------------------------------
        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: 172.33 s
--------------------------------------------------------------------------------
Loading data to table clickstream_data.ecommerce_bucketed partition (event_type=null)

Loaded : 4/4 partitions.
        Time taken to load dynamic partitions: 0.429 seconds
        Time taken for adding to write entity : 0.002 seconds
OK
event_time      product_id      category_id     category_code   brand   price   user_id user_session    event_type
Time taken: 184.578 seconds
hive>
```

Figure 19 Creation and insertion of data into bucketed table

Command:

SELECT * FROM ecommerce_bucketed LIMIT 5 ;

```
hive> SELECT * FROM ecommerce_bucketed LIMIT 5 ;
OK
ecommerce_bucketed.event_time   ecommerce_bucketed.product_id   ecommerce_bucketed.category_id  ecommerce_bucketed.category_code        ecommer
ce_bucketed.brand       ecommerce_bucketed.price        ecommerce_bucketed.user_id      ecommerce_bucketed.user_session ecommerce_bucketed.even
t_type
2019-10-08 07:00:58 UTC 5864603 1487580013086638258             italwax 7.94    557958553       5f4f15b4-ff56-4bfa-8a59-d0e6be5bd57c    cart
2019-10-10 08:13:48 UTC 5695629 1487580013841613016             estel   6.59    558716501       dc31944b-dd09-4f00-87a8-5244300fbb7e    cart
2019-10-09 08:32:15 UTC 5858175 1487580011585077370                     3.89    558362705       3f3a3061-5e9f-4a29-b29e-3e4d0e8000be    cart
2019-10-10 18:44:23 UTC 5869143 1783999063314661546             cosmoprofi      7.94    549195612       93620f57-af00-42b9-b3a8-40b8fdb286d1  c
art
2019-10-11 05:49:56 UTC 5690991 1998040850688377703             cnd     8.57    558528420       66ad7c3b-025f-4cf1-b4b1-9fa9be0c6af0    cart
Time taken: 0.216 seconds, Fetched: 5 row(s)
hive>
```

Figure 20 Checking the data in the bucketed table

Step 2: Comparing query performance between the tables and utilizing the optimized table for analysis

Analysis: 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.

3.1 Using the unoptimized table (flat table)

Command:

WITH Customer_Rank AS(SELECT user_id AS Customer, ROUND(SUM(price),2) AS Expenditure, RANK() OVER(ORDER BY ROUND(SUM(price),2) DESC) AS Rank FROM ecommerce WHERE event_type = 'purchase' GROUP BY user_id) SELECT Customer, Expenditure, Rank FROM Customer_Rank WHERE Rank <=10;

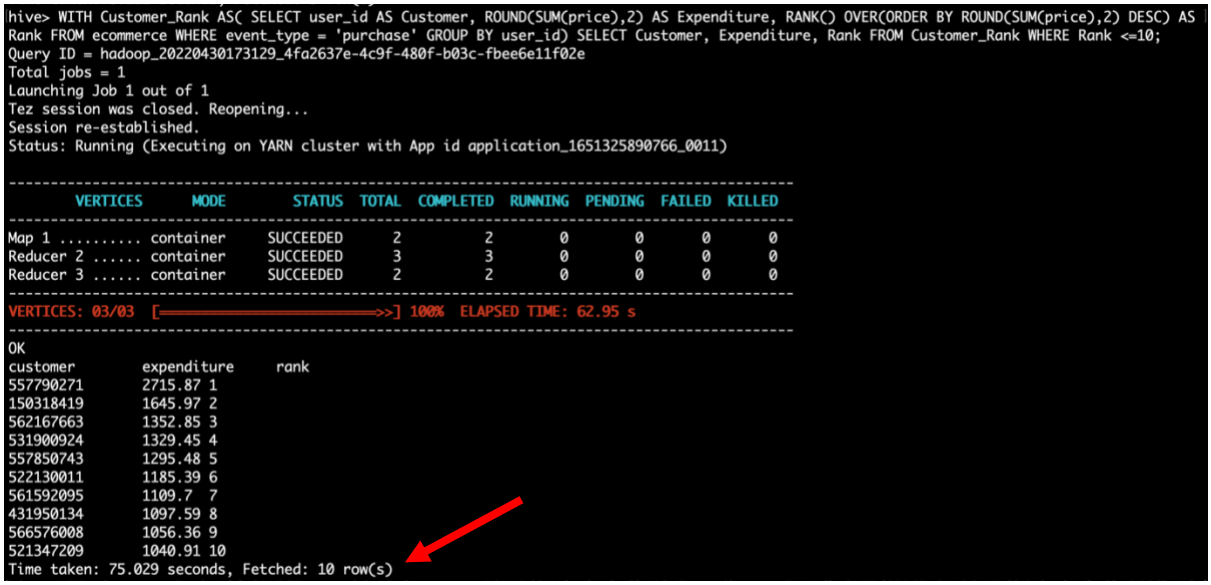

Figure 21 Analysis using the flat table

3.2 Using Bucketed table

Command:

WITH Customer_Rank AS(SELECT user_id AS Customer, ROUND(SUM(price),2) AS Expenditure, RANK() OVER(ORDER BY ROUND(SUM(price),2) DESC) AS Rank FROM ecommerce_bucketed WHERE event_type = 'purchase' GROUP BY user_id) SELECT Customer, Expenditure, Rank FROM Customer_Rank WHERE Rank <=10;

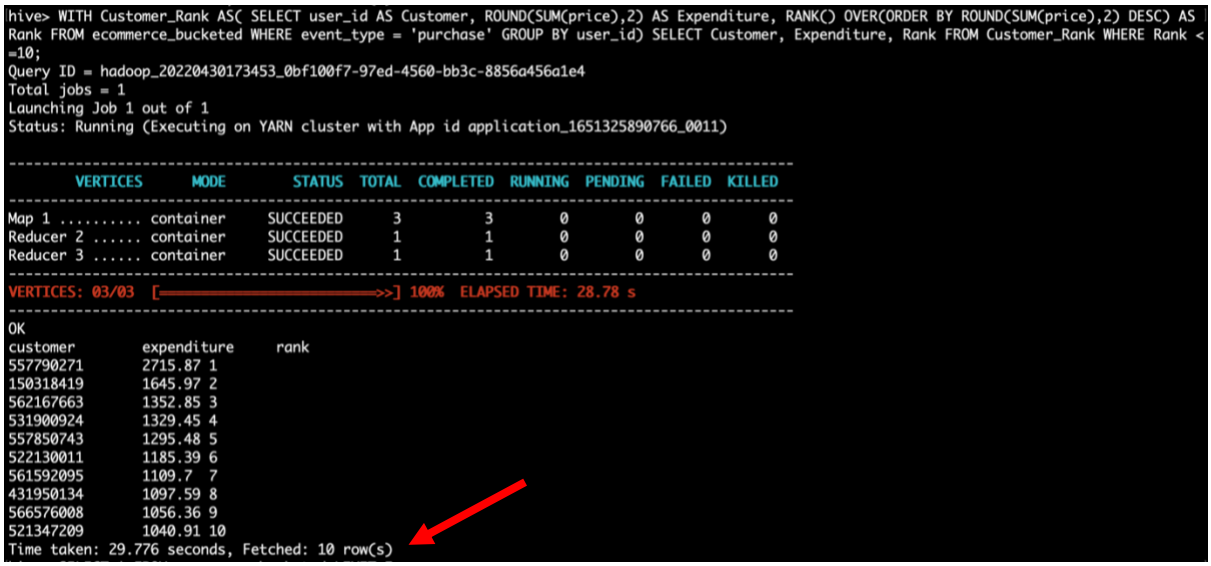

Figure 22 Analysis using bucketed table

Table	Table Name	Time Taken (in Seconds)
Flat table	ecommerce	75.029
Bucketed Table	ecommerce_bucketed	29.776

Table 2 Time taken by the tables to fetch the columns

Inference:

We can see that the query time for bucketed table is much lower than the unoptimized table and hence the Bucketed table is the optimized table giving faster results.

Hence, the right choice of columns for partitioning and bucketing leads to a better query time. Here, we have chosen ‘event_type’ as partitioning parameter due to its low cardinality and better performance and ‘price’ as the clustering parameter.

Analysis using Hive queries

Question 1:

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

Query:

SELECT ROUND(SUM(price),2) AS total_revenue_october FROM ecommerce_bucketed WHERE event_type = 'purchase' GROUP BY month(event_time) HAVING month(event_time) = 10;

```
hive> SELECT ROUND(SUM(price),2) AS total_revenue_october FROM ecommerce_bucketed WHERE event_type = 'purchase' GROUP BY month(event_time) HAVI
NG month(event_time) = 10;
Query ID = hadoop_20220430175523_34b1a069-8628-443c-84dd-8a8450a5a0d1
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_1651325890766_0012)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ......   container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 27.00 s
--------------------------------------------------------------------------------
OK
total_revenue_october
1211538.43
Time taken: 38.036 seconds, Fetched: 1 row(s)
hive>
```

Figure 23 Query for finding revenue in October

Answer:

Total revenue generated due to purchases in October was 1211538.43.

Question 2:

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

Query:

SELECT CASE WHEN (month(event_time) ==10) THEN 'Oct' ELSE 'Nov' END AS Month, ROUND(SUM(price),2) AS total FROM ecommerce_bucketed WHERE event_type = 'purchase' GROUP BY month(event_time);

```
hive> SELECT CASE WHEN (month(event_time) ==10) THEN 'Oct' ELSE 'Nov' END AS Month, ROUND(SUM(price),2) AS total FROM ecommerce_bucketed WHERE
event_type = 'purchase' GROUP BY month(event_time);
Query ID = hadoop_20220430175939_785962d2-3753-4268-98e4-3760af061702
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1651325890766_0012)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ......   container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 27.97 s
--------------------------------------------------------------------------------
OK
month   total
Oct     1211538.43
Nov     1531016.9
Time taken: 28.937 seconds, Fetched: 2 row(s)
hive>
```

Figure 24 Query to find sum purchases for October and November

Answer:

The sum of purchases for the month of October is 1211538.43 whereas for November is 1531016.9. Hence, the revenue due to 'purchases' in November is more than that of October.

Question 3:

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

Query:

SELECT (Rev_Nov - Rev_Oct) AS difference FROM (SELECT ROUND(SUM(CASE WHEN month(event_time)=10 THEN price ELSE 0 END),2) AS Rev_Oct, ROUND(SUM(CASE WHEN month(event_time)=11 THEN price ELSE 0 END),2) AS Rev_Nov FROM ecommerce_bucketed WHERE event_type='purchase') AS rev;

```
hive> SELECT (Rev_Nov - Rev_Oct) AS difference FROM (SELECT ROUND(SUM(CASE WHEN month(event_time)=10 THEN price ELSE 0
END),2) AS Rev_Oct, ROUND(SUM(CASE WHEN month(event_time)=11 THEN price ELSE 0 END),2) AS Rev_Nov FROM ecommerce_bucket
ed WHERE event_type='purchase') AS rev;
Query ID = hadoop_20220501091327_eaf82e01-c5f5-423a-aa6b-d61b3f0b370c
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1651395421072_0003)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 26.46 s
--------------------------------------------------------------------------------
OK
difference
319478.47
Time taken: 27.922 seconds, Fetched: 1 row(s)
hive>
```

Figure 25 Query to find change in revenue from October to November

Answer:

The difference between sales between October and November is 319478.47.

Question 4:

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

Query:

SELECT DISTINCT SPLIT (category_code,'\\.')[0] AS Category_List FROM ecommerce_bucketed WHERE category_code !='';

```
hive> SELECT DISTINCT SPLIT (category_code,'\\.')[0] AS Category_List FROM ecommerce_bucketed WHERE category_code !='';
Query ID = hadoop_20220504074820_162b859e-2502-4c66-af3d-8083c4540657
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1651647952134_0004)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      6          6        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      5          5        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 75.82 s
--------------------------------------------------------------------------------
OK
category_list
furniture
appliances
accessories
apparel
sport
stationery
Time taken: 76.763 seconds, Fetched: 6 row(s)
hive>
```

Figure 26 Query to find distinct categories of products

Answer:

category_list
furniture
appliances
accessories
apparel
sport
stationery

Question 5:

Find the total number of products available under each category.

Query:

SELECT SPLIT(category_code,'\\.')[0] AS Category, COUNT(product_id) AS Product_count FROM ecommerce_bucketed WHERE category_code !='' GROUP BY SPLIT(category_code,'\\.')[0] ORDER BY Product_count DESC;

```
hive> SELECT SPLIT(category_code,'\\.')[0] AS Category, COUNT(product_id) AS Product_count FROM ecommerce_bucketed WHERE category_
code !='' GROUP BY SPLIT(category_code,'\\.')[0] ORDER BY Product_count DESC;
Query ID = hadoop_20220504074416_5a69c56d-b7d2-4f6b-894f-2c2aa1854398
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1651647952134_0004)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      6          6        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      5          5        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 70.61 s
--------------------------------------------------------------------------------
OK
category        product_count
appliances      61736
stationery      26722
furniture       23604
apparel 18232
accessories     12929
sport   2
Time taken: 71.391 seconds, Fetched: 6 row(s)
hive>
```

Figure 27 Query to find total number of products available under each category

Answer:

category	product_count
appliances	61736
stationery	26722
furniture	23604
apparel	18232
accessories	12929
sport	2

Question 6:

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

Query:

SELECT brand, ROUND(SUM(price),2) AS total_sales FROM ecommerce_bucketed WHERE brand != '' AND event_type = 'purchase' GROUP BY brand ORDER BY total_sales DESC LIMIT 1;

```
hive> SELECT brand, ROUND(SUM(price),2) AS total_sales FROM ecommerce_bucketed WHERE brand != '' AND event_type = 'purc
hase' GROUP BY brand ORDER BY total_sales DESC LIMIT 1;
Query ID = hadoop_20220501100708_51d2f59d-0bf4-4793-a2d9-1022b5a81d95
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1651395421072_0005)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ......   container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ......   container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 25.43 s
--------------------------------------------------------------------------------
OK
brand   total_sales
runail  148297.94
Time taken: 26.09 seconds, Fetched: 1 row(s)
hive>
```

Figure 28 Query to find brand with maximum sales in both the months

Answer:

brand total_sales
runail 148297.94

Question 7:

Which brands increased their sales from October to November?

Query:

SELECT oct.brand as Brand, ROUND((nov.Nov_sale - oct.Oct_sale),2) AS increase_in_sale FROM (SELECT brand, SUM(price) as Oct_sale FROM ecommerce_bucketed WHERE event_type = 'purchase' AND brand !='' AND month(event_time) = 10 GROUP BY brand) oct JOIN (SELECT brand, SUM(price) as Nov_sale FROM ecommerce_bucketed WHERE event_type = 'purchase' AND brand !='' AND month(event_time) = 11 GROUP BY brand) nov ON oct.brand = nov.brand WHERE nov.Nov_sale > oct.Oct_sale ORDER BY increase_in_sale DESC;

```
hive> SELECT oct.brand as Brand, ROUND((nov.Nov_sale - oct.Oct_sale),2) AS increase_in_sale FROM (SELECT brand, SUM(pri
ce) as Oct_sale FROM ecommerce_bucketed WHERE event_type = 'purchase' AND brand != '' AND month(event_time) = 10 GROUP
BY brand) oct JOIN (SELECT brand, SUM(price) as Nov_sale FROM ecommerce_bucketed WHERE event_type = 'purchase' AND bran
d != '' AND month(event_time) = 11 GROUP BY brand) nov ON oct.brand = nov.brand WHERE nov.Nov_sale > oct.Oct_sale ORDER
 BY increase_in_sale DESC;
Query ID = hadoop_20220501101345_06469e06-dbef-46b9-b681-c41c92a3d3ae
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_1651395421072_0006)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Map 5 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 4 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 6 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 06/06  [==========================>>] 100%  ELAPSED TIME: 35.82 s
--------------------------------------------------------------------------------
```

Figure 29 Query to find the brands that increase sales

```
----------------------------
OK
brand   increase_in_sale
grattol 36027.17
uno     15737.72
lianail 10501.4
ingarden        10404.82
strong  9474.64
jessnail        7057.39
cosmoprofi      6214.18
polarus 5358.21
runail  5219.38
freedecor       4250.02
staleks 3355.88
bpw.style       3265.29
lovely  3234.68
marathon        2992.35
haruyama        2962.22
yoko    2950.97
italwax 2859.13
benovy  2850.35
kaypro  2387.36
estel   2385.92
concept 2348.26
kapous  2165.92
f.o.x   1953.05
masura  1792.39
milv    1737.07
beautix 1729.0
artex   1596.61
domix   1537.12
shik    1498.52
smart   1444.88
roubloff        1422.41
levrana 1420.54
oniq    1416.24
irisk   1354.08
severina        1344.6
joico   1309.58
zeitun  1300.97
beauty-free     1228.69
swarovski       1155.23
de.lux  1115.81
metzger 1083.71
markell 1065.68
sanoto  1052.54
nagaraku        957.94
ecolab  951.45
art-visage      905.09
levissime       857.81
missha  856.45
solomeya        786.1
rosi    764.52
refectocil      759.4
kaaral  673.64
kosmekka        631.93
kinetics        611.01
browxenna       585.36
airnails        572.62
uskusi  548.04
coifin  525.49
s.care  500.39
limoni  487.7
matrix  483.49
gehwol  468.61
greymy  460.28
bioaqua 455.23
farmavita       454.6
sophin  447.66
yu-r    402.3
kiss    395.78
lador   387.92
ellips  360.19
jas     338.47
lowence 324.91
nitrile 315.4
shary   304.53
kims    302.0
happyfons       289.67
kocostar        284.08
insight 278.26
candy   264.42
bluesky 258.29
beauugreen      256.84
protokeratin    255.54
trind   244.89
entity  239.55
skinlite        238.51
provoc  235.83
fedua   211.43
ecocraft        200.79
keen    199.27
mane    193.47
freshbubble     183.64
chi     179.67
cristalinas     157.32
farmona 150.97
latinoil        135.07
miskin  135.03
elizavecca      133.77
nefertiti       133.12
finish  132.0
igrobeauty      131.41
dizao   126.38
osmo    116.73
batiste 101.77
carmex  98.28
eos     98.27
depilflax       96.71
enjoy   95.22
kerasys 94.29
aura    93.56
plazan  92.64
koelf   84.56
nirvel  71.29
konad   70.84
egomania        68.57
cutrin  68.25
laboratorium    66.02
inm     63.19
marutaka-foot   60.11
profhenna       57.62
koelcia 57.25
balbcare        57.05
elskin  56.56
foamie  45.45
ladykin 44.92
likato  44.91
mavala  37.28
vilenta 33.61
beautyblender   30.67
biore   29.66
orly    28.71
estelare        27.06
profepil        24.66
blixz   24.45
godefroy        23.9
glysolid        21.86
veraclara       21.1
kamill  18.48
treaclemoon     18.12
supertan        16.14
deoproce        12.33
rasyan  10.14
fly     10.03
tertio  9.64
jaguar  8.54
soleo   8.33
neoleor 8.29
moyou   4.57
bodyton 4.3
skinity 3.56
grace   1.69
cosima  0.7
ovale   0.56
```

Answer:

The brand with highest increase in sales from October to November is Grattol with 36027.17

Question 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.

Query:

WITH Customer_Rank AS(SELECT user_id AS Customer, ROUND(SUM(price),2) AS Expenditure, RANK() OVER(ORDER BY ROUND(SUM(price),2) DESC) AS Rank FROM ecommerce_bucketed WHERE event_type = 'purchase' GROUP BY user_id) SELECT Customer, Expenditure, Rank FROM Customer_Rank WHERE Rank <=10;

```
hive> WITH Customer_Rank AS( SELECT user_id AS Customer, ROUND(SUM(price),2) AS Expenditure, RANK() OVER(ORDER BY ROUND(SUM(price),2) DESC) AS
Rank FROM ecommerce_bucketed WHERE event_type = 'purchase' GROUP BY user_id) SELECT Customer, Expenditure, Rank FROM Customer_Rank WHERE Rank <
=10;
Query ID = hadoop_20220430173453_0bf100f7-97ed-4560-bb3c-8856a456a1e4
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1651325890766_0011)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 28.78 s
--------------------------------------------------------------------------------
OK
customer        expenditure     rank
557790271       2715.87 1
150318419       1645.97 2
562167663       1352.85 3
531900924       1329.45 4
557850743       1295.48 5
522130011       1185.39 6
561592095       1109.7  7
431950134       1097.59 8
566576008       1056.36 9
521347209       1040.91 10
Time taken: 29.776 seconds, Fetched: 10 row(s)
```

Figure 30 Query to find top 10 users

Answer:

customer	expenditure	rank
557790271	2715.87	1
150318419	1645.97	2
562167663	1352.85	3
531900924	1329.45	4
557850743	1295.48	5
522130011	1185.39	6
561592095	1109.7	7
431950134	1097.59	8
566576008	1056.36	9

521347209 1040.91 10

Dropping tables, database and terminating the cluster

Step 1: Dropping tables and database

Command:

SHOW TABLES ;

DROP TABLE ecommerce ;

DROP TABLE ecommerce_bucketed ;

SHOW DATABASES ;

DROP DATABASE clickstream_data ;

```
hive> SHOW TABLES ;
OK
ecommerce
ecommerce_bucketed
Time taken: 0.027 seconds, Fetched: 2 row(s)
hive> DROP TABLE ecommerce ;
OK
Time taken: 0.382 seconds
hive> DROP TABLE ecommerce_bucketed ;
OK
Time taken: 0.161 seconds
hive> SHOW TABLES ;
OK
Time taken: 0.023 seconds
hive> SHOW DATABASES ;
OK
clickstream_data
default
Time taken: 0.03 seconds, Fetched: 2 row(s)
hive> DROP DATABASE clickstream_data ;
OK
Time taken: 0.194 seconds
hive> SHOW DATABASES ;
OK
default
Time taken: 0.019 seconds, Fetched: 1 row(s)
hive> EXIT ;
```

Figure 31 Dropping the tables and database

We have successfully dropped all the tables created and the database and only left with the default database.

Step 2: Terminating the cluster

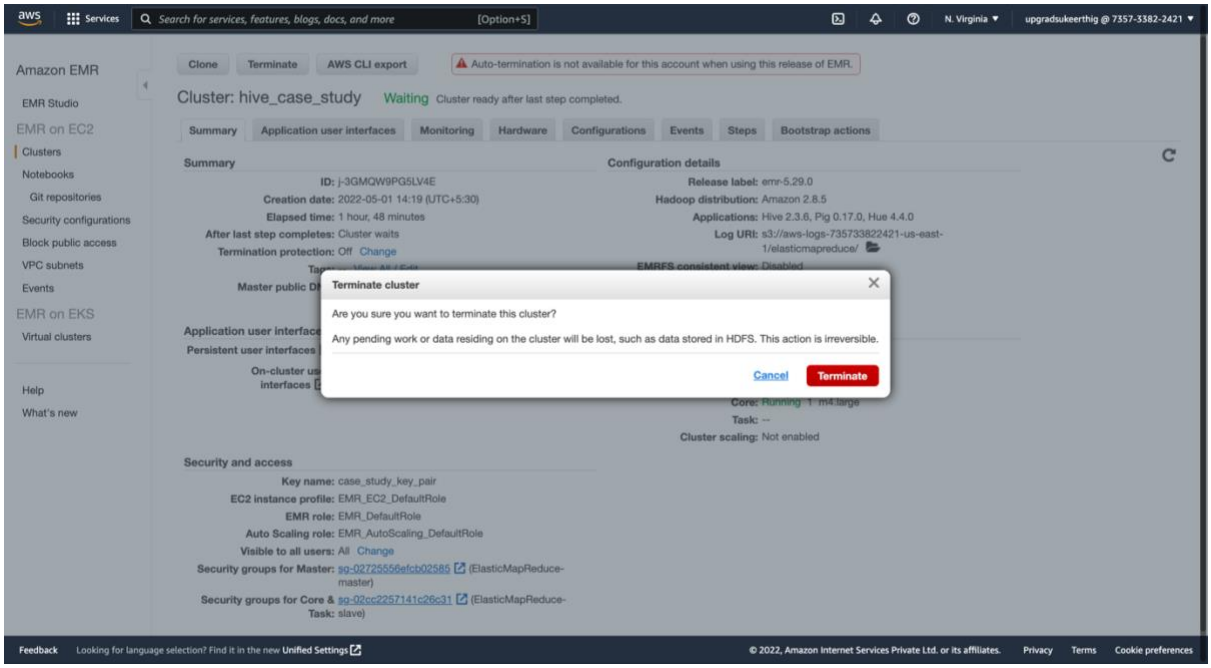

Figure 32 Cluster termination

```
hive>
Broadcast message from root@ip-172-31-4-238
        (unknown) at 10:43 ...

The system is going down for power off NOW!
Connection to ec2-34-201-32-127.compute-1.amazonaws.com closed by remote host.
Connection to ec2-34-201-32-127.compute-1.amazonaws.com closed.
(base) Sukeerthis-MacBook-Pro:Downloads sukeerthig$
```

Figure 33 Closing the connection

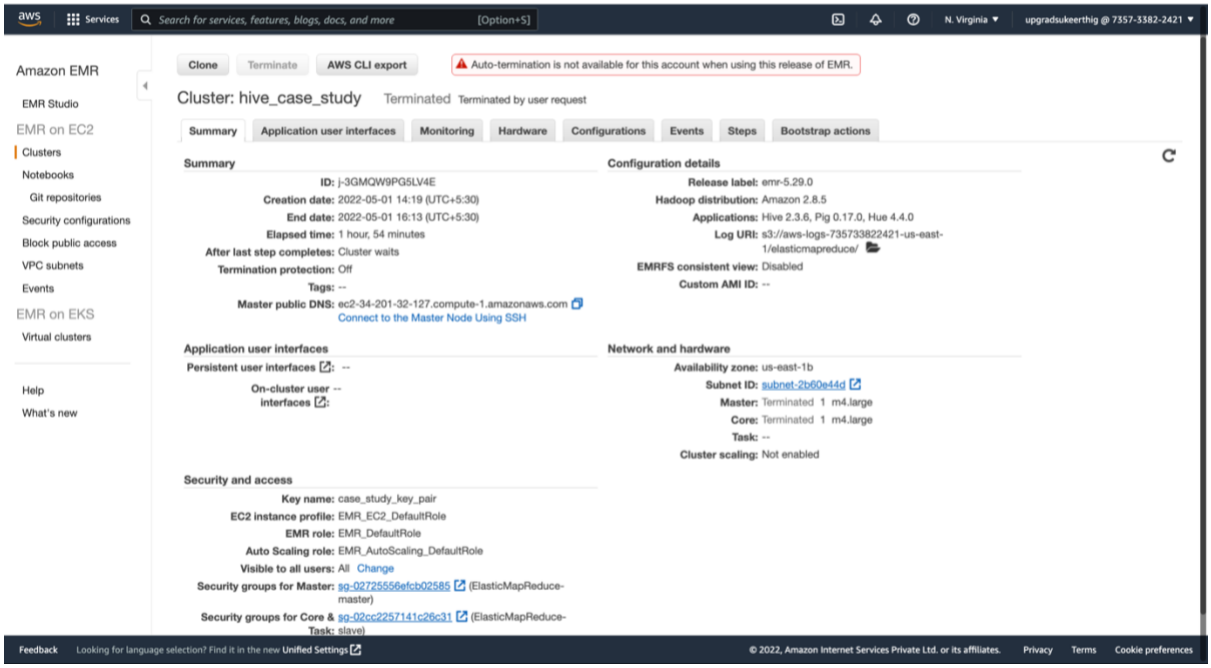

Figure 34 Cluster successfully terminated