UPGRAD
DS41 BATCH 2022

HIVE CASE STUDY

BY
GARVITA SHARMA
YASH HIMANSHUBHAI PATEL
BIBEKANANDA SAHOO

BUSINESS INTELLIGENCE
AND DATA ANALYTICS

TABLE OF CONTENT

01 CREATING EMR CLUSTER

02 CREATE AND LOAD THE DATA IN THE S3 BUCKET

03 LOAD THE DATA FROM THE S3 BUCKET TO EMR CLUSTER

04 CREATING TABLE SCHEMA IN THE HIVE AND LOADING DATA INTO IT

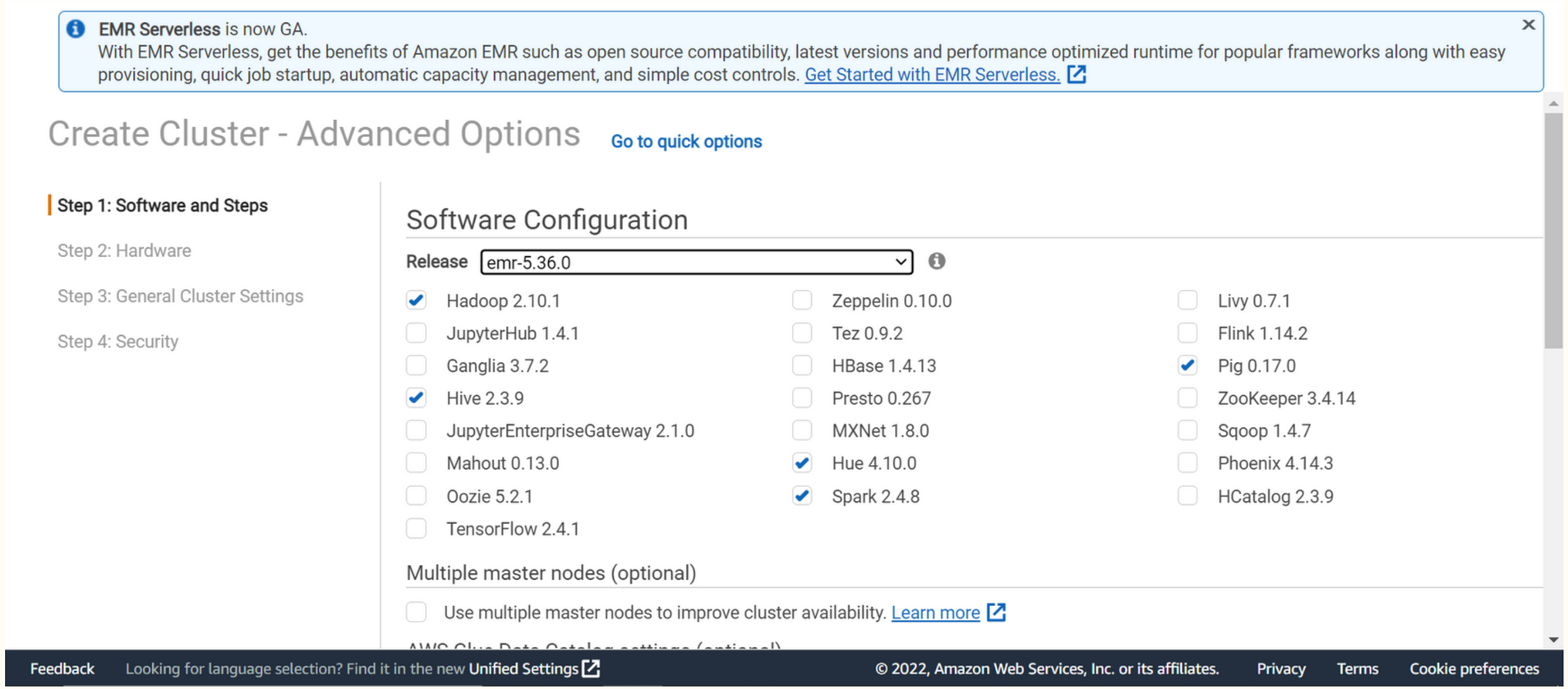
05 SOLVING PROBLEMS USING HQL QUERIES

06 TERMINATING THE CLUSTER

1. CREATING EMR CLUSTER

step 1: software and storage

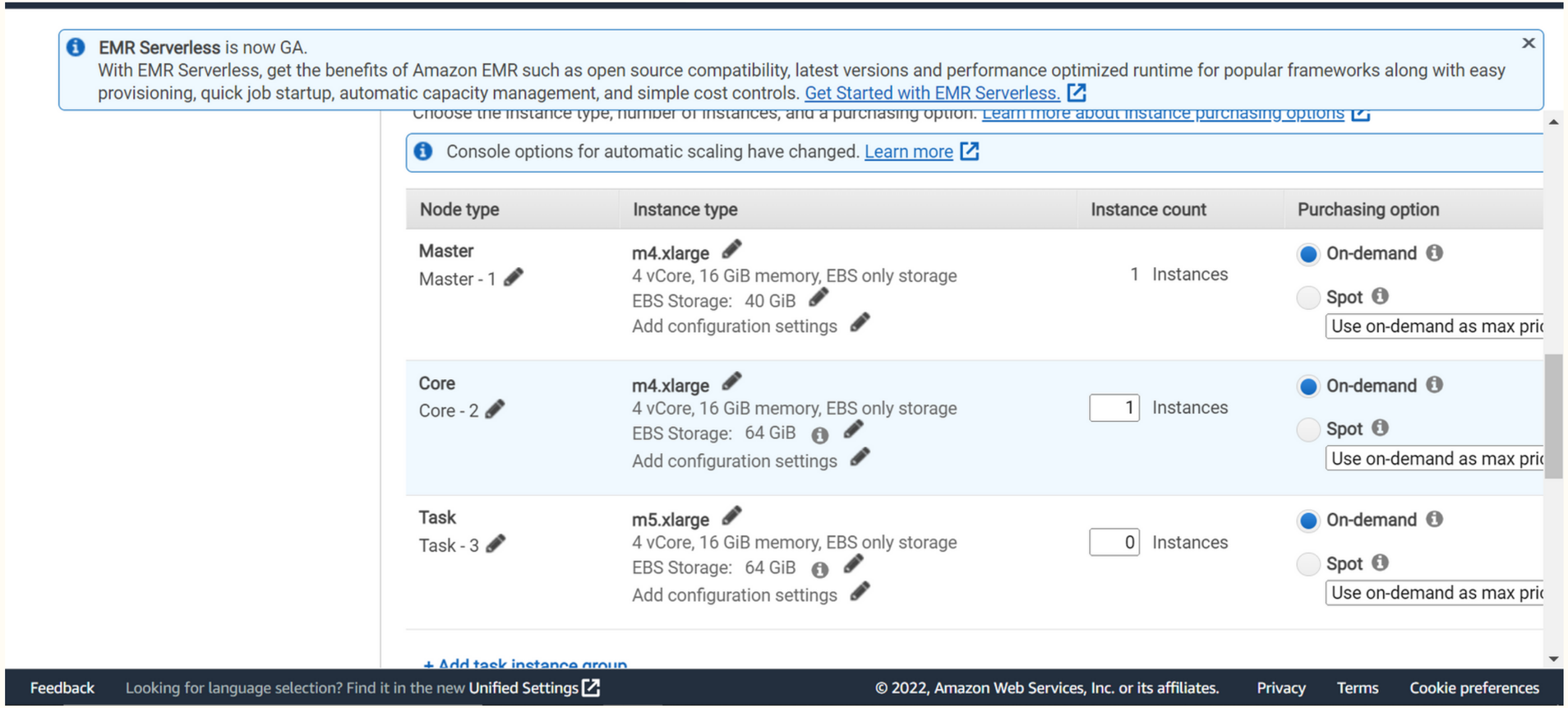
- we choose emr-5.36.0



1. CREATING EMR CLUSTER

step 2: Hardware

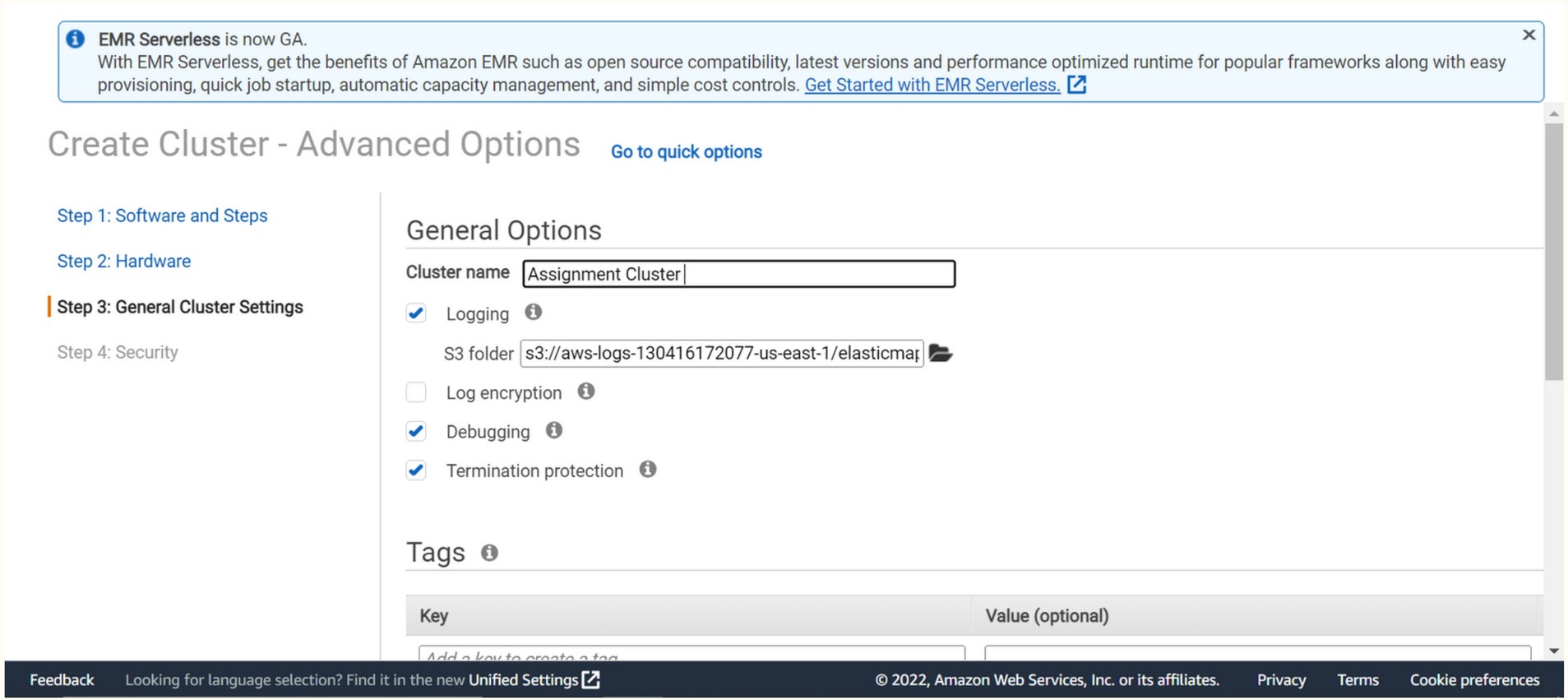
- we choose 1 instance for master note and 1 for core node/slave node



1. CREATING EMR CLUSTER

step 3: General Cluster settings

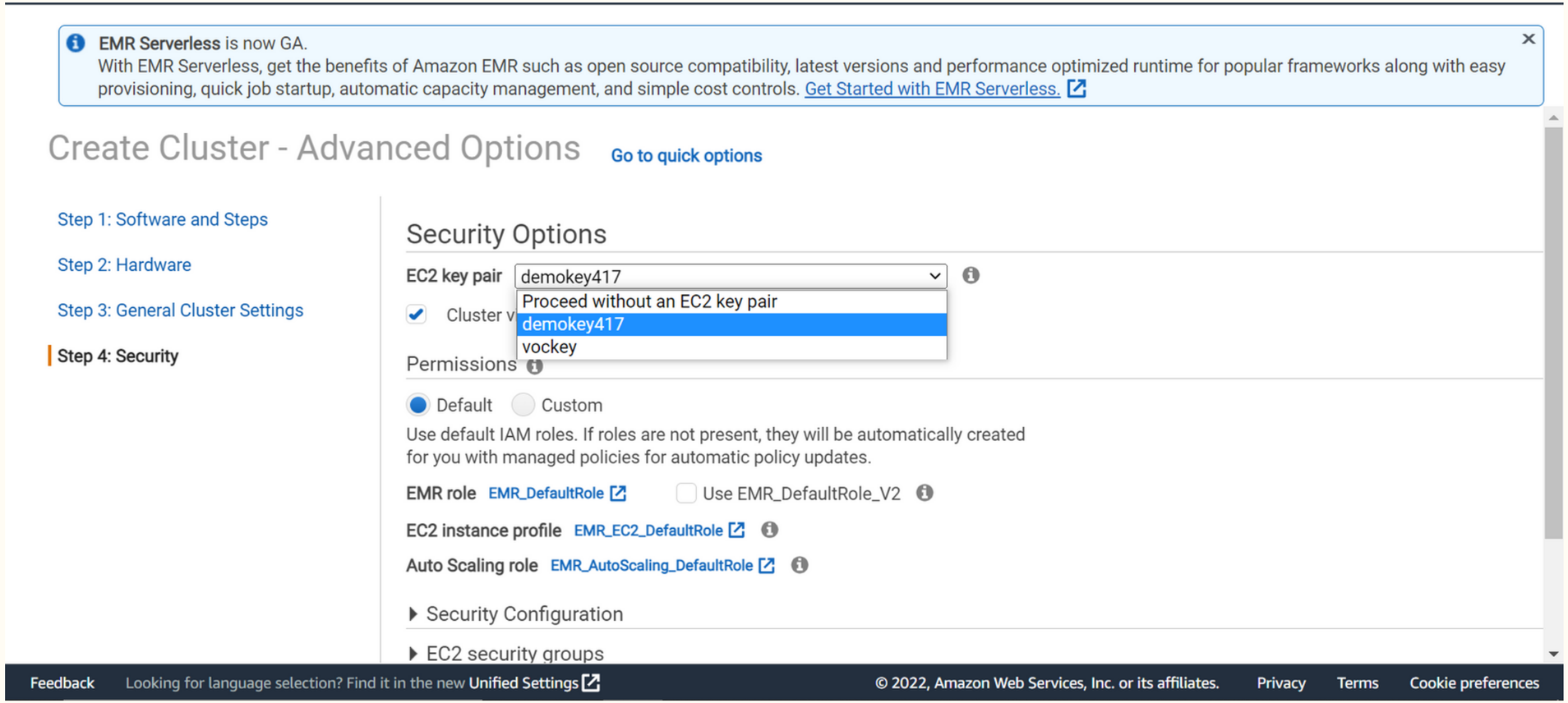
- The name of our cluster was Assignment cluster



1. CREATING EMR CLUSTER

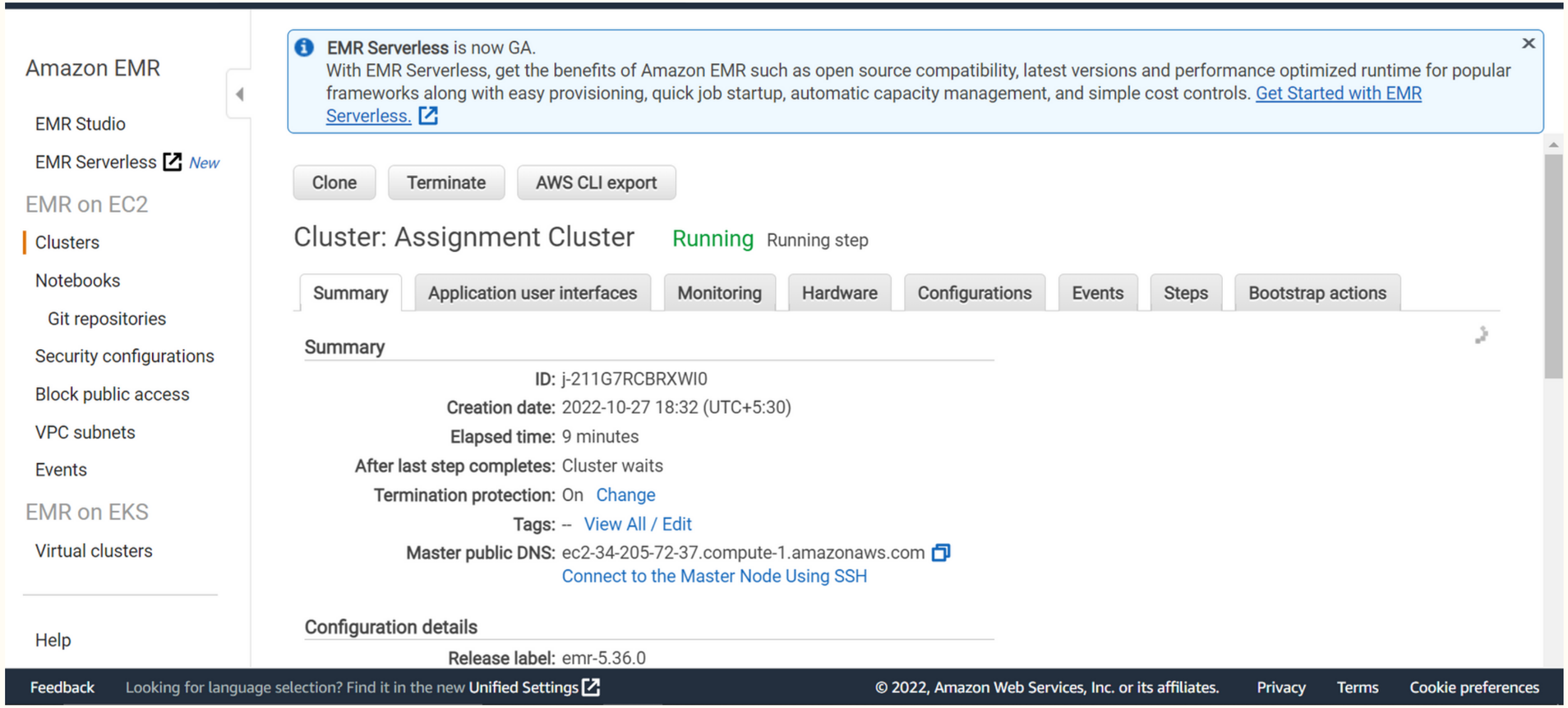
step 4: Security

- Provide the key pair whose name was "demokey417" and click create cluster.

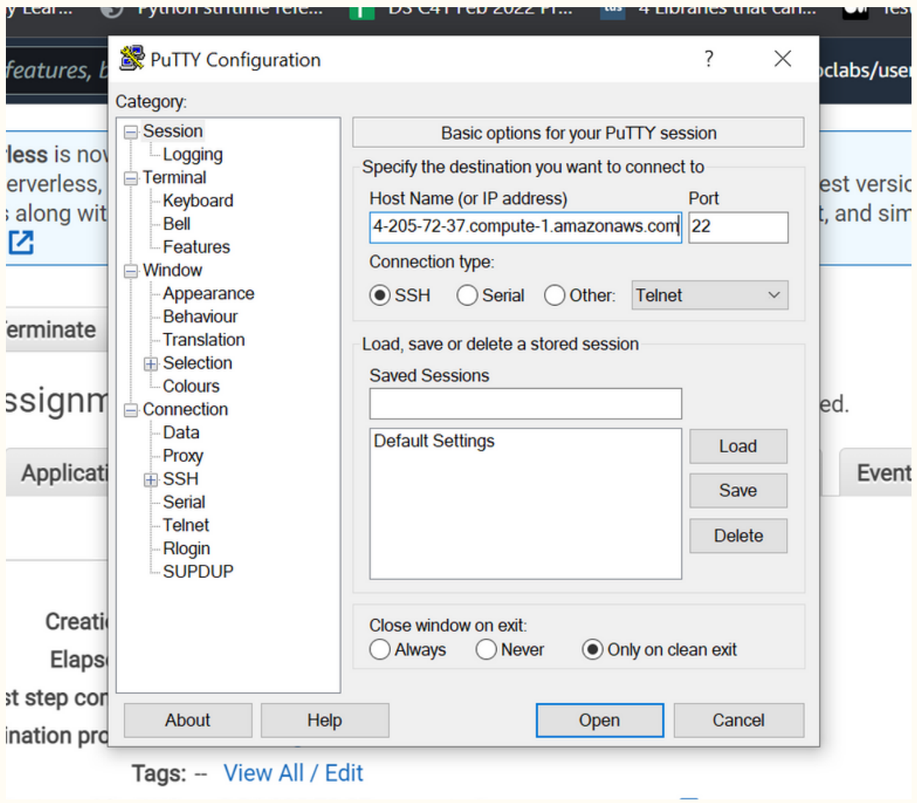


1. CREATING EMR CLUSTER

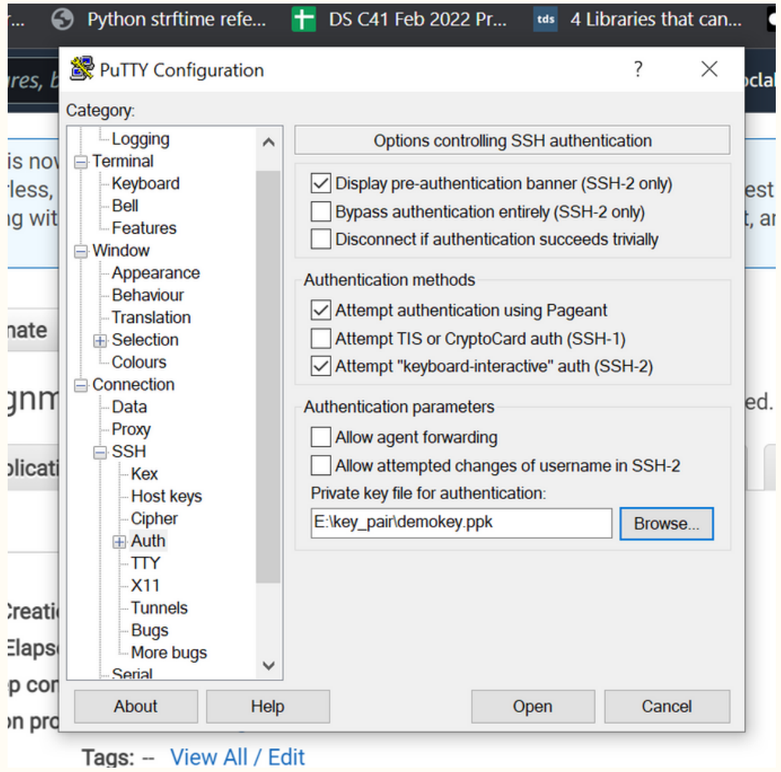
- Cluster is successfully created and running



1. CREATING EMR CLUSTER

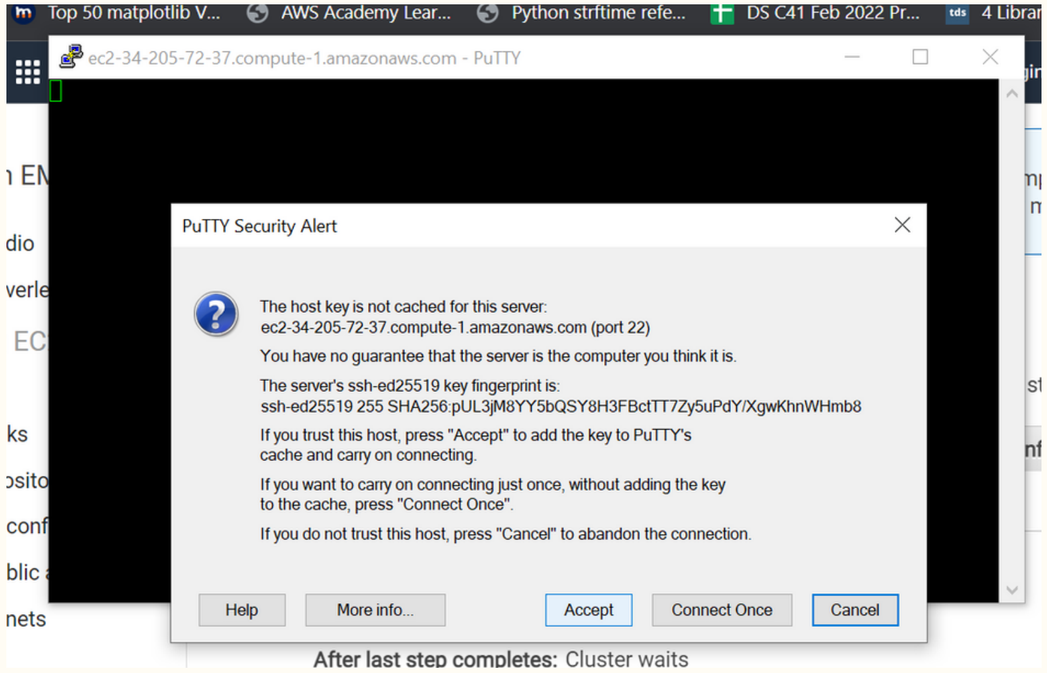


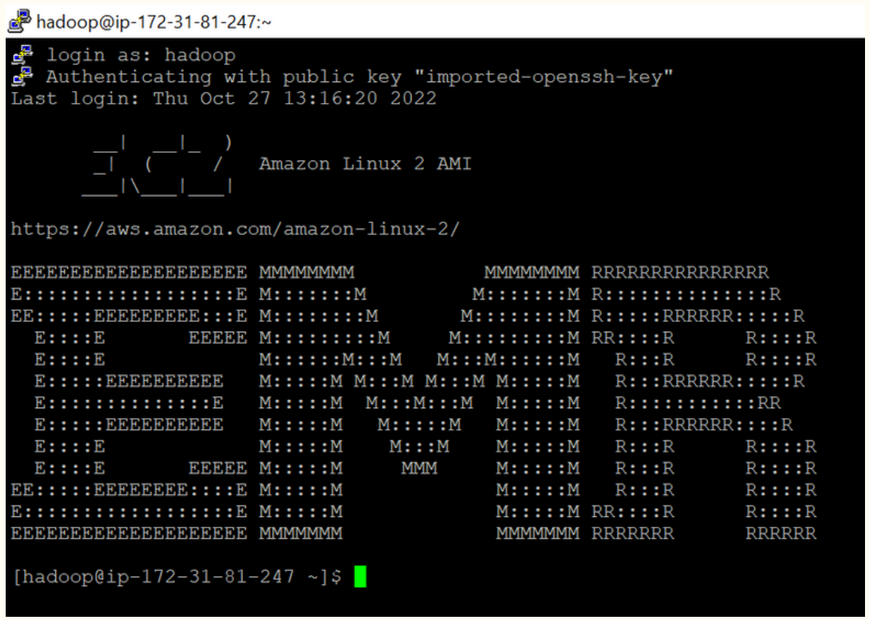
- The next step is to open Hadoop command-line interface.
- Copy the master public DNS and paste it on putty software
- Then go to ssh->auth-> and browse the PPK file to paste the file path.
- Then click open.



1. CREATING EMR CLUSTER

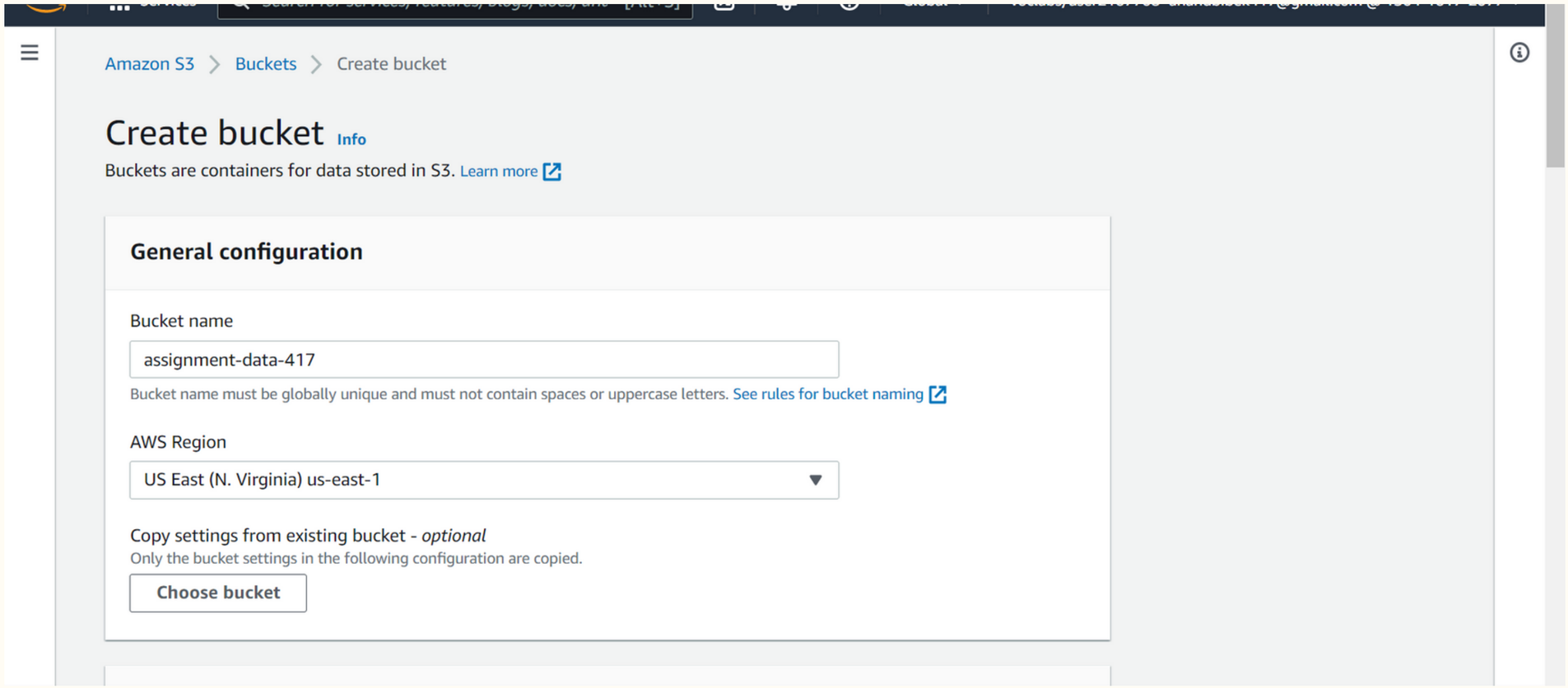
- Then click Accept.
- Then write Hadoop to log in.
- Then our EMR cluster CLI successfully opens.





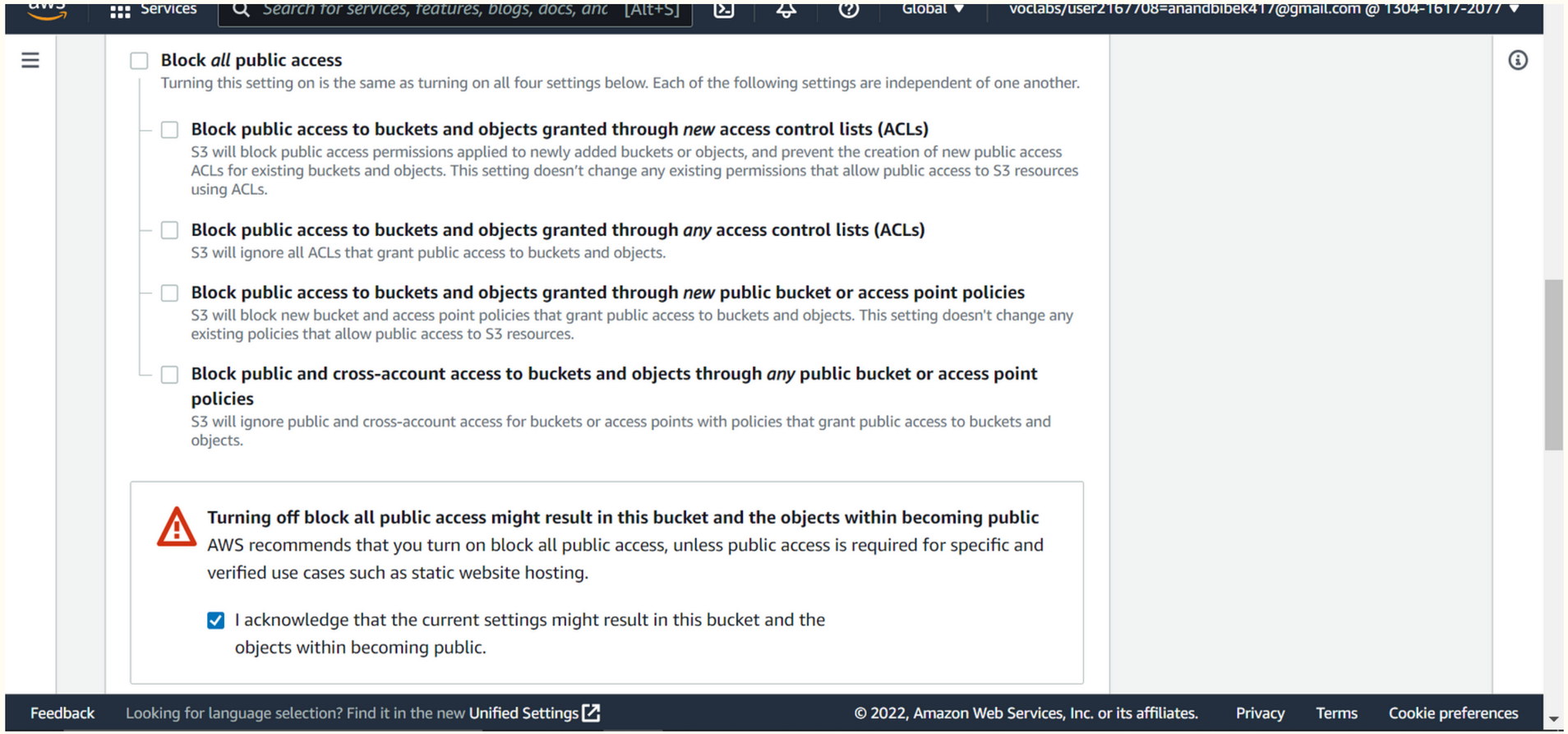
2. CREATING AN S3 BUCKET AND LOAD THE DATA INTO IT

- Creating a bucket with a unique name assignment-data-417.
- AWS Region set to US East(N. Virginia)



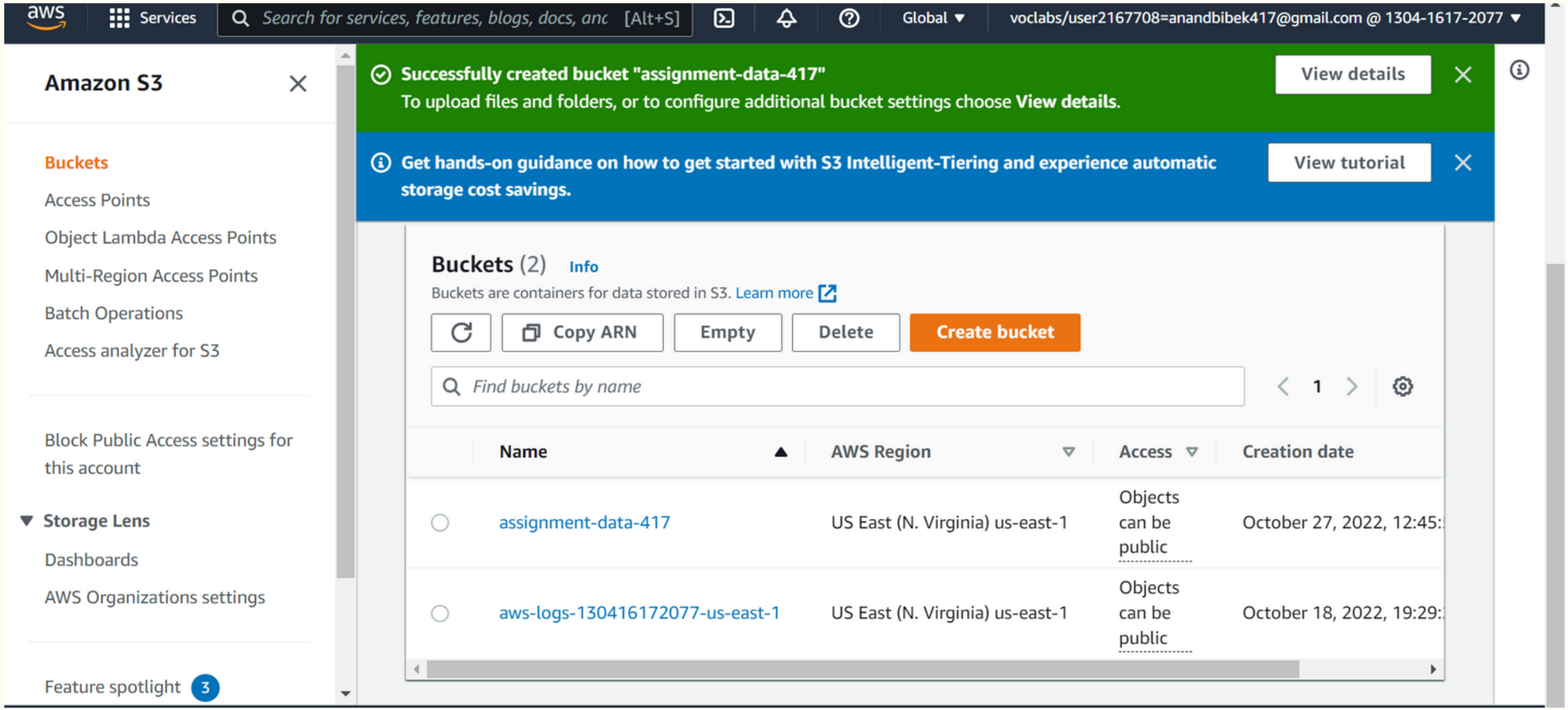
2. CREATING AN S3 BUCKET AND LOAD THE DATA INTO IT

- Untick the Block of all public access therefore anyone can assess the data.
- Then click create bucket.



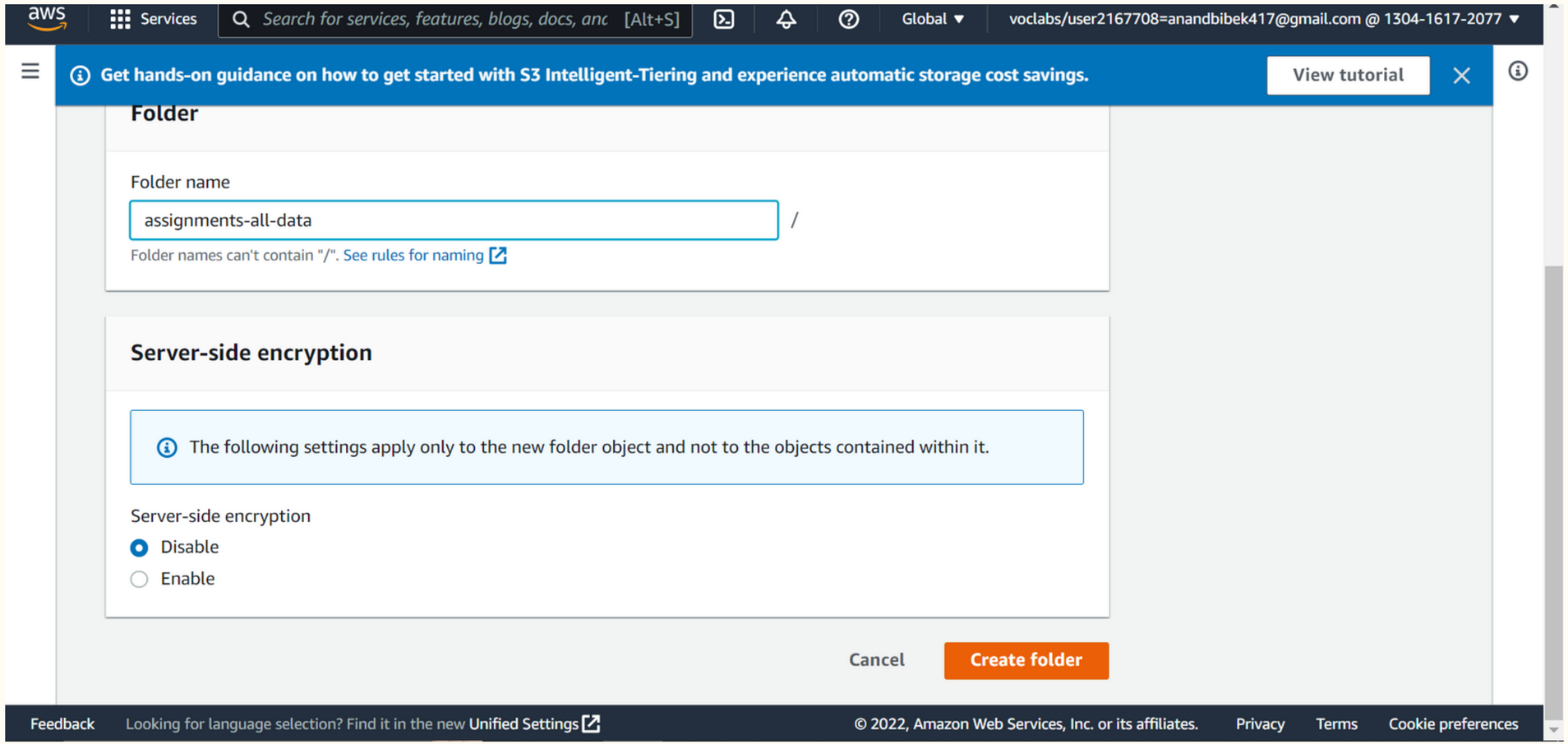
2. CREATING AN S3 BUCKET AND LOAD THE DATA INTO IT

- Successfully created the bucket assignment-data-417.



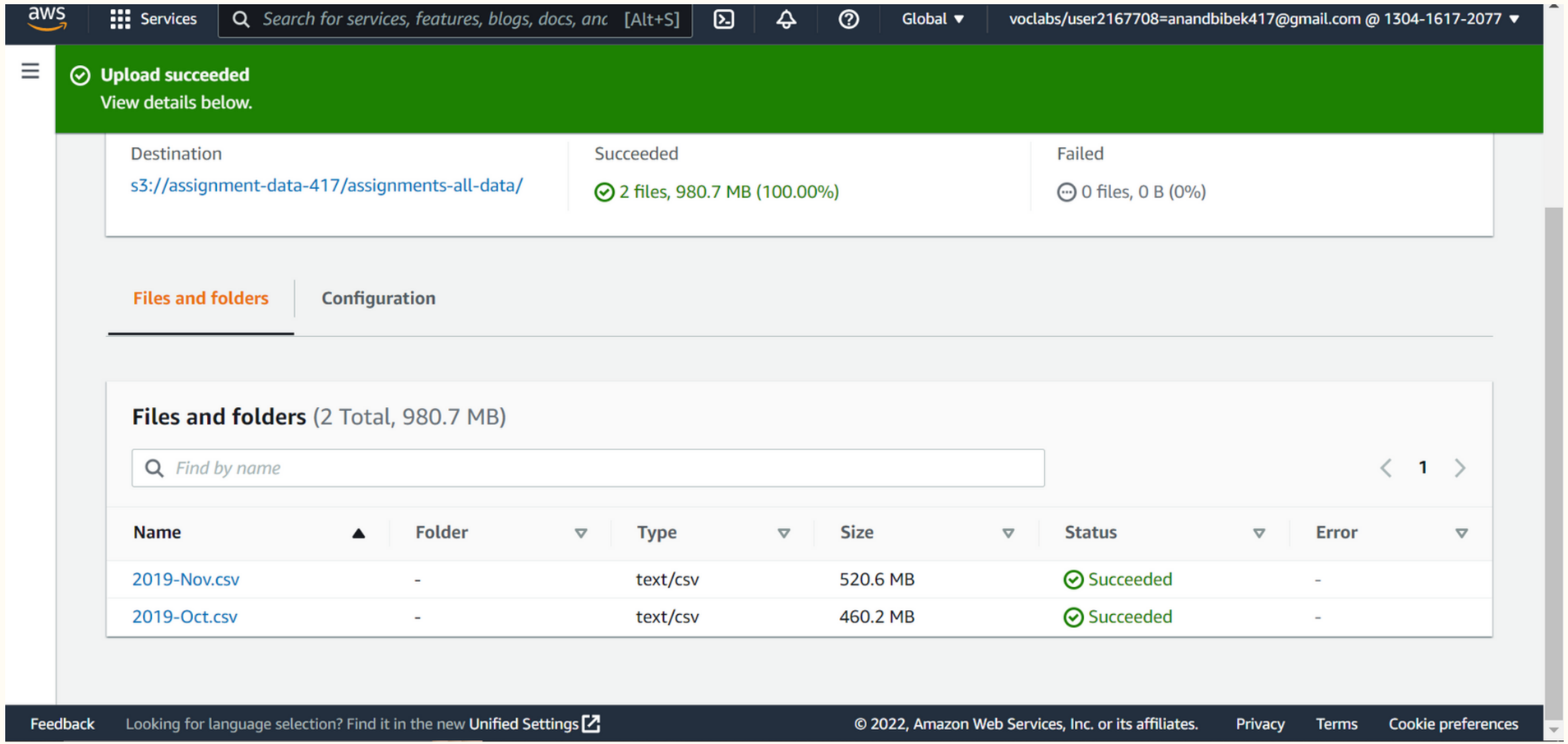
2. CREATING AN S3 BUCKET AND LOAD THE DATA INTO IT

- Creating a folder called assignment-all-data inside the bucket.



2. CREATING AN S3 BUCKET AND LOAD THE DATA INTO IT

- Uploading the data into the folder.



3. LOAD THE DATA FROM THE S3 BUCKET TO EMR CLUSTER

```
login as: hadoop
Authenticating with public key "imported-openssh-key"
Last login: Thu Oct 27 13:22:09 2022 from 223.176.65.131

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/

EEEEEEEEEEEEEEEEEEE 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
EEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-81-247 ~]$ ls
[hadoop@ip-172-31-81-247 ~]$ aws s3 cp s3://assignment-data-417/assignments-all-data/2019-Nov.csv .
download: s3://assignment-data-417/assignments-all-data/2019-Nov.csv to ./2019-Nov.csv
[hadoop@ip-172-31-81-247 ~]$ aws s3 cp s3://assignment-data-417/assignments-all-data/2019-Oct.csv .
download: s3://assignment-data-417/assignments-all-data/2019-Oct.csv to ./2019-Oct.csv
[hadoop@ip-172-31-81-247 ~]$
```

- Uploading the data from the s3 bucket to the Hadoop cluster using the below command
- aws s3 cp <s3_path> <instance_path>

- Using the ls command to check whether files are uploaded or not.

```
hadoop@ip-172-31-81-247:~
[hadoop@ip-172-31-81-247 ~]$ ls
2019-Nov.csv  2019-Oct.csv
[hadoop@ip-172-31-81-247 ~]$
```

4. CREATING TABLE SCHEMA IN THE HIVE AND LOADING DATA INTO IT

- step 1: creat database "assignment_db"

```
hadoop@ip-172-31-81-247:~
hive> create database assignment_db;
OK
Time taken: 1.088 seconds
hive> show databases ;
OK
assignment_db
default
Time taken: 0.245 seconds, Fetched: 2 row(s)
hive> use assignment_db ;
OK
```

- step 2: Make two temporary tables called temp_oct_data and temp_nov_data
- Then check tables are created or not using "show tables;" command.

```
hive> use assignment_db ;
OK
Time taken: 0.035 seconds
hive> create table temp_oct_data (event_time string,event_type string,product_id string,category_id string,category_code string,brand string,price string,use
r_id string,user_session string) row format delimited fields terminated by ',' lines terminated by '\n' stored as textfile ;
OK
Time taken: 0.616 seconds
hive> create table temp_nov_data (event_time string,event_type string,product_id string,category_id string,category_code string,brand string,price string,use
r_id string,user_session string) row format delimited fields terminated by ',' lines terminated by '\n' stored as textfile ;
OK
Time taken: 0.069 seconds
hive> show tables ;
OK
temp_nov_data
temp_oct_data
Time taken: 0.039 seconds, Fetched: 2 row(s)
```

4. CREATING TABLE SCHEMA IN THE HIVE AND LOADING DATA INTO IT

- step 3: Load the data into the table using the below code
- load data local inpath <hdfs path> into <table_name> ;
- Then checking the data is uploaded or not using the below code
- select * from <table_name> limit 5 ;

```
hive> load data local inpath "/home/hadoop/2019-Oct.csv" into table temp_oct_data ;
Loading data to table assignment_db.temp_oct_data
OK
Time taken: 1.858 seconds
hive> load data local inpath "/home/hadoop/2019-Nov.csv" into table temp_nov_data ;
Loading data to table assignment_db.temp_nov_data
OK
Time taken: 2.217 seconds
hive> select * from temp_oct_data limit 5 ;
OK
event_time      event_type      product_id      category_id     category_code   brand   price   user_id user_session
2019-10-01 00:00:00 UTC cart    5773203 1487580005134238553             runail  2.62    463240011       26dd6e6e-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:03 UTC cart    5773353 1487580005134238553             runail  2.62    463240011       26dd6e6e-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:07 UTC cart    5881589 2151191071051219817             lovely  13.48   429681830       49e8d843-adf3-428b-a2c3-fe8bc6a307c9
2019-10-01 00:00:07 UTC cart    5723490 1487580005134238553             runail  2.62    463240011       26dd6e6e-4dac-4778-8d2c-92e149dab885
Time taken: 1.551 seconds, Fetched: 5 row(s)
hive> select * from temp_nov_data limit 3 ;
OK
event_time      event_type      product_id      category_id     category_code   brand   price   user_id user_session
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
Time taken: 0.197 seconds, Fetched: 3 row(s)
hive>
```

- Data is successfully uploaded in the table.

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 1: Find the total revenue generated due to purchases made in October.
- Let's first solve this problem in a normal way using the below HQL query.
- select sum(price) as total_revenue from temp_oct_data where event_type='purchase' ;

```
hive> set hive.cli.print.header=true;
hive> use assignment_db ;
OK
Time taken: 0.779 seconds
hive> select sum(price) as total_revenue from temp_oct_data where event_type='purchase' ;
Query ID = hadoop_20221028072033_3de2f9a9-fce7-461d-a625-c0a4411e988a
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1666939584820_0004)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     11         11        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 27.48 s
--------------------------------------------------------------------------------
OK
total_revenue
1211538.4299999736
Time taken: 31.781 seconds, Fetched: 1 row(s)
hive>
```

- use the below command to show the headers.
- set hive.cli.print.header=true ;
- The query takes 27.48 sec
- Total revenue generated due to purchases was 121158.43

3. SOLVING PROBLEMS USING HQL QUERIES

- Question 1: Find the total revenue generated due to purchases made in October.
- Let's solve this problem using the partition concept.
- step 1: create another table named "par_purchase_oct" where we use a static partition on event_type='purchase'
- step 2: load the data into that table par_purchase_oct from the temporary table "temp_oct_data".

```
hive> create table par_purchase_oct (event_time string,product_id string,category_id string,category_code string,brand string,price string,user_id string,use
r_session string) partitioned by(event_type string) row format delimited fields terminated by ',' lines terminated by '\n' stored as textfile ;
OK
Time taken: 0.202 seconds
hive> insert into table par_purchase_oct partition(event_type='purchase') select event_time,product_id,category_id,category_code,brand,price,user_id,user_ses
sion from temp_oct_data   where event_type='purchase' ;
Query ID = hadoop_20221028072144_6cda4397-fefa-44fc-9e8f-60fb7089724a
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1666939584820_0004)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     11         11        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 01/01  [==========================>>] 100%  ELAPSED TIME: 26.26 s
----------------------------------------------------------------------------------------------
Loading data to table assignment_db.par_purchase_oct partition (event_type=purchase)
OK
event_time      product_id      category_id     category_code   brand   price   user_id user_session
Time taken: 27.716 seconds
```

- Data is succesfully loaded into the par_purchase_oct table.

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 1: Find the total revenue generated due to purchases made in October.
- step 3: Then run the same query using the par_purchase_oct table and analyze the time difference.
- select sum(price) as total_revenue from par_purchase_oct where event_type='purchase' ;

```
hive> select sum(price) as total_revenue from par_purchase_oct where event_type='purchase' ;
Query ID = hadoop_20221028072221_b9540f22-87d8-4d02-85e1-df5981e38aed
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1666939584820_0004)

----------------------------------------------------------------------------------------------
        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: 9.92 s
----------------------------------------------------------------------------------------------
OK
total_revenue
1211538.4299998377
Time taken: 10.492 seconds, Fetched: 1 row(s)
hive>
```

- we clearly saw a huge time difference after using the partition table
- previously the query takes 27.48sec but now it takes only 9.92sec to run.

- Let's check where the partition folder is stored.

```
[hadoop@ip-172-31-89-140 ~]$ hadoop fs -ls /user/hive/warehouse/assignment_db.db/par_purchase_oct
Found 1 items
drwxrwxrwt   - hadoop hdfsadmingroup          0 2022-10-29 15:32 /user/hive/warehouse/assignment_db.db/par_purchase_oct/event_type=purchase
[hadoop@ip-172-31-89-140 ~]$
```

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 2: Write a query to yield the total sum of purchases per month in a single output.
- lets first solve this in normal way.

```
hive> (select sum(price) from temp_oct_data where event_type='purchase')  union ( select sum(price) from temp_nov_data where event_type='purchase');
Query ID = hadoop_20221028072747_72510653-af1f-4c1f-a711-311cb46e4679
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1666939584820_0005)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     11         11        0        0       0       0
Map 5 .......... container     SUCCEEDED     11         11        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 4 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 6 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 05/05  [==========================>>] 100%  ELAPSED TIME: 32.89 s
--------------------------------------------------------------------------------
OK
1531016.8999999936
1211538.4299999734
```

```
(SELECT Sum(price) AS total_revenue_oct
 FROM  temp_oct_data
 WHERE  event_type = 'purchase')
 UNION
(SELECT Sum(price) AS total_revenue_nov
 FROM  temp_nov_data
 WHERE  event_type = 'purchase');
```

- It takes 32.89 sec
- Total revenue for October month was 1211538.43 and for November month 1531016.90

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 2: Write a query to yield the total sum of purchases per month in a single output.
- lets now solve this using partition tables.

```
hive> (select sum(price) from par_purchase_nov where event_type='purchase')  union ( select sum(price) from par_purchase_oct where event_type='purchase');
Query ID = hadoop_20221028073344_e3884543-18cc-4122-a4c6-79b16303a17d
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1666939584820_0005)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      4          4        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 4 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 6 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 05/05  [==========================>>] 100%  ELAPSED TIME: 22.46 s
--------------------------------------------------------------------------------
OK
1211538.4299998377
1531016.899999925
Time taken: 23.308 seconds, Fetched: 2 row(s)
```

```
(SELECT Sum(price) AS total_revenue_oct
 FROM   par_purchase_oct
 WHERE  event_type = 'purchase')
 UNION
(SELECT Sum(price) AS total_revenue_nov
 FROM   par_purchase_nov
 WHERE  event_type = 'purchase');
```

- A huge time difference we can see here after using partition tables it only takes 22.46sec.

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 3: Write a query to find the change in revenue generated due to purchases from October to November.
- step 1: first we solve this using the temporary tables and then solve this by partitioning tables and comparing the time difference.

HQL QUERY FOR THIS PROBLEM

```
WITH summery
    AS (SELECT *
        FROM   par_purchase_oct
        UNION ALL
        SELECT *
        FROM   par_purchase_nov),
    revenue_summery
    AS (SELECT Month(event_time) AS month,
               Sum(price)        AS total_revenue
        FROM   summery
        WHERE  event_type = 'purchase'
        GROUP  BY Month(event_time))
SELECT Max(total_revenue) - Min(total_revenue) AS change_in_revenue
FROM   revenue_summery;
```

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 3: Write a query to find the change in revenue generated due to purchases from October to November.

```
hive> with summery as (select * from temp_oct_data union all select * from temp_nov_data ), revenue_summery as (select month(event_time) as month ,sum(price)
 as total_revenue from summery where event_type='purchase'  group by month(event_time) ) select max(total_revenue)-min(total_revenue) as change_in_revenue fr
om revenue_summery ;
Query ID = hadoop_20221028134259_66d5a2ae-1223-48a4-aa3c-8e6f6c3fa390
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1666963707286_0001)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     11         11        0        0       0       0
Map 5 .......... container     SUCCEEDED     11         11        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      6          6        0        0       0       0
Reducer 4 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 30.58 s
--------------------------------------------------------------------------------
OK
change_in_revenue
319478.47000002
Time taken: 31.216 seconds, Fetched: 1 row(s)
```

```
hive> with summery as (select * from par_purchase_oct union all select * from par_purchase_nov ), revenue_summery as (select month(event_time) as month ,sum(
price) as total_revenue from summery where event_type='purchase'  group by month(event_time) ) select max(total_revenue)-min(total_revenue) as change_in_reve
nue from revenue_summery  ;
Query ID = hadoop_20221028134339_e19d7f97-7aec-4cfc-b602-d64d402c138e
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1666963707286_0001)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Map 5 .......... container     SUCCEEDED      4          4        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 4 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 27.19 s
--------------------------------------------------------------------------------
OK
change_in_revenue
319478.4700000873
```

- There is a clear time difference we can see while using the temporary tables the query takes 30.58sec and while using the partition table it takes 27.19sec.

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 4: Find distinct categories of products. Categories with null category code can be ignored.

- HQL QUERY FOR THIS PROBLEM
- select distinct(category_code) from temp_oct_data where category_code !="";
- lets solve this first in normal way using temporary table.

```
hive> select distinct(category_code) from temp_oct_data where category_code !="";
Query ID = hadoop_20221028143456_11db2807-47b8-4b15-843a-1f819599fe4b
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_1666963707286_0003)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     11         11        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      4          4        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 30.29 s
--------------------------------------------------------------------------------
OK
category_code
accessories.bag
appliances.environment.vacuum
appliances.personal.hair_cutter
sport.diving
category_code
apparel.glove
furniture.bathroom.bath
furniture.living_room.cabinet
stationery.cartrige
accessories.cosmetic_bag
appliances.environment.air_conditioner
furniture.living_room.chair
Time taken: 36.624 seconds, Fetched: 12 row(s)
```

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 4: Find distinct categories of products. Categories with null category code can be ignored.
- Let's use the bucketing concept for solving this problem
- Create another table called "category_bucket" where we apply bucketing on category_code and make 10 buckets.
- Then load the data into it from the temporary table.

```
hive> create table category_bucket (event_time string,event_type string,product_id string,category_id string,category_code string,brand string,price string,u
ser_id string,user_session string) clustered by (category_code) into 10 buckets row format delimited fields terminated by ',' lines terminated by '\n' stored
 as textfile ;
OK
Time taken: 0.059 seconds
hive> insert into table category_bucket select event_time,event_type,product_id,category_id,category_code,brand,price,user_id,user_session from temp_oct_data
 ;
Query ID = hadoop_20221028143724_2d896295-158f-4d39-82a1-df8cf03399ea
```

```
hadoop@ip-172-31-93-109:~
Query ID = hadoop_20221028143724_2d896295-158f-4d39-82a1-df8cf03399ea
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1666963707286_0003)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     11         11        0        0       0       0
Reducer 2 ...... container     SUCCEEDED     10         10        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 53.21 s
----------------------------------------------------------------------------------------------
Loading data to table assignment_db.category_bucket
OK
event_time      event_type      product_id      category_id     category_code   brand   price   user_id user_session
Time taken: 53.955 seconds
```

- Data loaded successfully in the category_bucket table .

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 4: Find distinct categories of products. Categories with null category code can be ignored.
- Now lets again run the query and saw the time difference.

```
Time taken: 53.955 seconds
hive> select distinct(category_code) from category_bucket where category_code !="";
Query ID = hadoop_20221028143824_43136284-a8b8-4e93-80f5-1fd760188f00
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1666963707286_0003)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     12         12        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      4          4        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 23.47 s
--------------------------------------------------------------------------------
OK
category_code
accessories.bag
appliances.environment.vacuum
appliances.personal.hair_cutter
sport.diving
category_code
apparel.glove
furniture.bathroom.bath
furniture.living_room.cabinet
stationery.cartrige
accessories.cosmetic_bag
appliances.environment.air_conditioner
furniture.living_room.chair
Time taken: 23.973 seconds, Fetched: 12 row(s)
hive>
```

- Previously it takes 30.29sec but if we use bucketing table then its uses 23.47sec.

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 5: Find the total number of products available under each category.
- HQL QUERY FOR THIS PROBLEM
- select category_code , count(product_id) as products_in_stuck from temp_oct_data where category_code!="" group by category_code ;

```
hive> select category_code , count(product_id) as products_in_stuck from temp_oct_data where category_code!="" group by category_code ;
Query ID = hadoop_20221028194108_adc6df77-5e11-4934-bf6f-ca27a25874a0
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1666984675793_0002)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     11         11        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      4          4        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 26.52 s
--------------------------------------------------------------------------------
OK
accessories.bag 5848
appliances.environment.vacuum   27732
appliances.personal.hair_cutter 828
sport.diving    2
category_code   1
apparel.glove   7430
furniture.bathroom.bath 5018
furniture.living_room.cabinet   6234
stationery.cartrige     13459
accessories.cosmetic_bag        642
appliances.environment.air_conditioner  161
furniture.living_room.chair     123
Time taken: 26.976 seconds, Fetched: 12 row(s)
hive>
```

- its takes 26.52 sec

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 6: Which brand had the maximum sales in October and November combined?

```
WITH summary AS
(
     SELECT *
     FROM   temp_oct_data
     UNION ALL
     SELECT *
     FROM   temp_nov_data )
 SELECT   brand,
      Sum(price) AS sales
  FROM     summary
  WHERE    brand!=""
  AND      event_type='purchase'
  GROUP BY brand
  ORDER BY sales DESC limit 1 ;
```

- Hive query for this problem
- The brand which has the maximum sales in October and November combined is "runail" with total sales of 148297.90

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 6: Which brand had the maximum sales in October and November combined?
- If we use temporary table then its takes 38.78 sec

```
hive> WITH summary AS (SELECT * FROM   temp_oct_data UNION ALL SELECT * FROM   temp_nov_data ) select brand,sum(price) as sales from summary where brand!=""
and event_type='purchase' group by brand order by sales desc limit 1 ;
Query ID = hadoop_20221030153840_e71b4334-a901-4338-8d9a-cde4cce37671
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1667143742921_0001)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     11         11        0        0       0       0
Map 5 .......... container     SUCCEEDED     11         11        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      6          6        0        0       0       0
Reducer 4 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 38.78 s
--------------------------------------------------------------------------------
OK
runail  148297.9400000001
```

- But if we use partition table then its only takes 26.12sec

```
hive> WITH summary AS (SELECT * FROM   par_purchase_oct UNION ALL SELECT * FROM   par_purchase_nov ) select brand,sum(price) as sales from summary where bran
d!="" and event_type='purchase' group by brand order by sales desc limit 1 ;
Query ID = hadoop_20221030153954_e3b5b72b-e199-44e2-9a67-7629fc8b8a82
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1667143742921_0001)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Map 5 .......... container     SUCCEEDED      4          4        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 4 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 26.12 s
--------------------------------------------------------------------------------
OK
runail  148297.94000000114
```

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 7: Which brands increased their sales from October to November?
- Hive query for this problem

```
WITH oct_summery
     AS (SELECT brand,
                Sum(price) AS oct_sales
         FROM   par_purchase_oct
         WHERE  event_type = 'purchase'
         GROUP  BY brand),
     nov_summery
     AS (SELECT brand,
                Sum(price) AS nov_sales
         FROM   par_purchase_nov
         WHERE  event_type = 'purchase'
         GROUP  BY brand),
     sales_summary
     AS (SELECT o.brand AS brand_name,
                oct_sales,
                nov_sales
         FROM   oct_summery o
                INNER JOIN nov_summery n
                        ON o.brand = n.brand)
SELECT brand_name,
   oct_sales,
   nov_sales
FROM   sales_summary
WHERE  nov_sales > oct_sales
   AND brand_name != "";
```

5. SOLVING PROBLEMS USING HQL QUERIES

- Question 7: Which brands increased their sales from October to November?

```
hive> with oct_summery as (select brand,sum(price) as oct_sales from par_purchase_oct where event_type='purchase' group by brand ) , nov_summery as ( select
brand,sum(price) as nov_sales from par_purchase_nov where event_type='purchase' group by brand ), sales_summary as ( select o.brand as brand_name ,oct_sales,
nov_sales from oct_summery o inner join nov_summery n on o.brand=n.brand) select brand_name,oct_sales,nov_sales from sales_summary where nov_sales > oct_sale
s and brand_name!="";
No Stats for assignment_db@par_purchase_oct, Columns: price, brand
No Stats for assignment_db@par_purchase_nov, Columns: price, brand
Query ID = hadoop_20221030160739_3fb818bf-dd89-4367-ac2f-fe29c83c6ded
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_1667143742921_0003)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Map 3 .......... container     SUCCEEDED      4          4        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 4 ...... container     SUCCEEDED      2          2        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 26.59 s
--------------------------------------------------------------------------------
OK
art-visage      2092.710000000001       2997.8000000000043
artex   2730.6400000000003      4327.25
batiste 772.3999999999999       874.1699999999998
beautix 10493.949999999975      12222.949999999997
beautyblender   78.74000000000001       109.41
bioaqua 942.8900000000001       1398.1200000000001
biore   60.650000000000006      90.31
blixz   38.95   63.400000000000006
browxenna       14331.369999999997      14916.729999999998
carmex  145.08  243.36000000000004
concept 11032.13999999993       13380.399999999994
cutrin  299.37  367.61999999999995
deoproce        316.84  329.17
domix   10472.049999999981      12009.169999999975
ecocraft        41.160000000000004      241.95
ecolab  262.85  1214.3000000000006
```

- The query takes 26.59sec and here we use the partition tables.

5. SOLVING PROBLEMS USING HQL QUERIES

- 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 the top 10 users who spend the most.
- Hive query for this problem

```
WITH summary AS
(
        SELECT *
        FROM   par_purchase_oct
        UNION ALL
        SELECT *
        FROM   par_purchase_nov)
SELECT   user_id,
         Sum(price) AS total_spend
FROM     summary
WHERE    event_type='purchase'
GROUP BY user_id
ORDER BY total_spend DESC limit 10 ;
```

5. SOLVING PROBLEMS USING HQL QUERIES

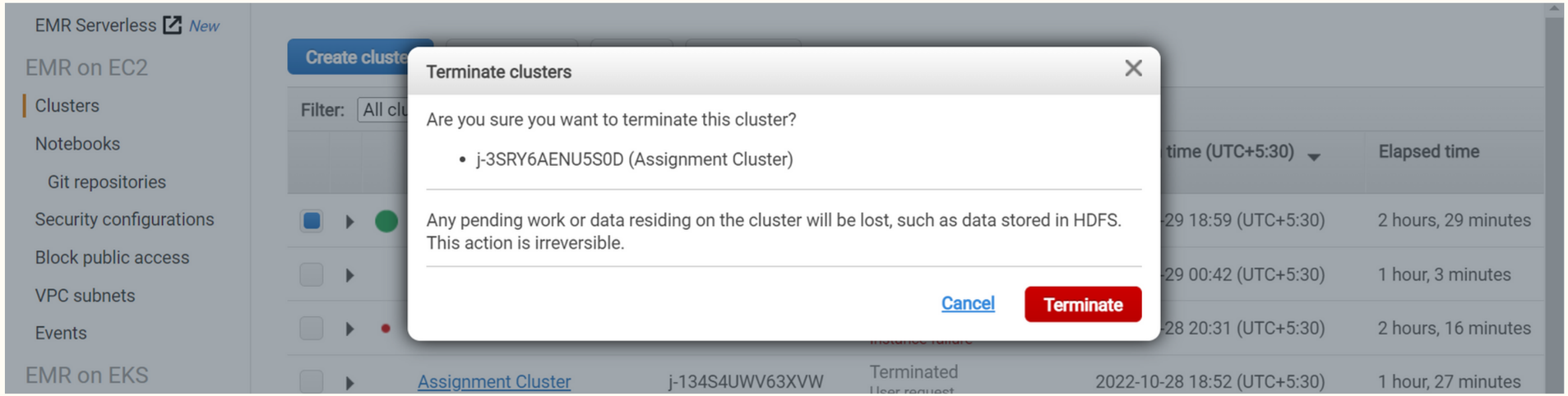
- Question 8: Your company wants to reward the top 10 users of its website with a Golden Customer plan. Write a query to generate a list of the top 10 users who spend the most.

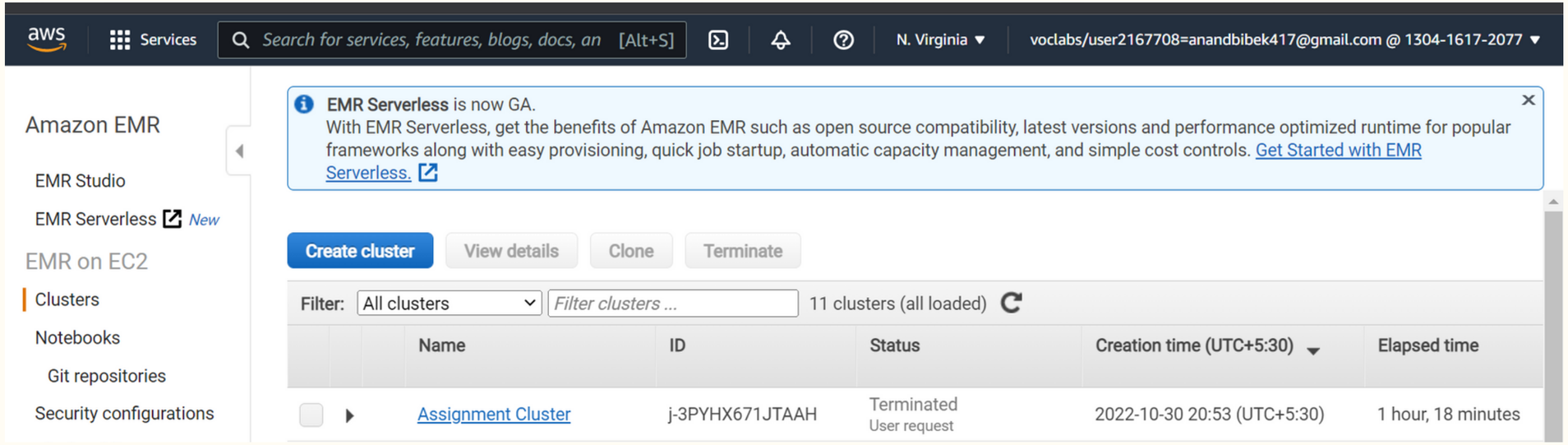
```
hive> WITH summary AS (SELECT * FROM   par_purchase_oct UNION ALL SELECT * FROM   par_purchase_nov) select user_id,sum(price) as total_spend from summary whe
re event_type='purchase' group by user_id order by total_spend desc limit 10 ;
Query ID = hadoop_20221029153855_0fa89cdc-8de8-4900-bc49-4080926aa564
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1667050494736_0005)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Map 5 .......... container     SUCCEEDED      4          4        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 4 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 24.00 s
--------------------------------------------------------------------------------
OK
user_id total_spend
557790271       2715.8699999999935
150318419       1645.9700000000005
562167663       1352.8500000000001
531900924       1329.4499999999998
557850743       1295.48
522130011       1185.3899999999996
561592095       1109.6999999999996
431950134       1097.5899999999997
566576008       1056.3600000000017
521347209       1040.9099999999999
Time taken: 24.588 seconds, Fetched: 10 row(s)
hive>
```

- The query takes 24.00sec

6. TERMINATING THE CLUSTER





THANK YOU