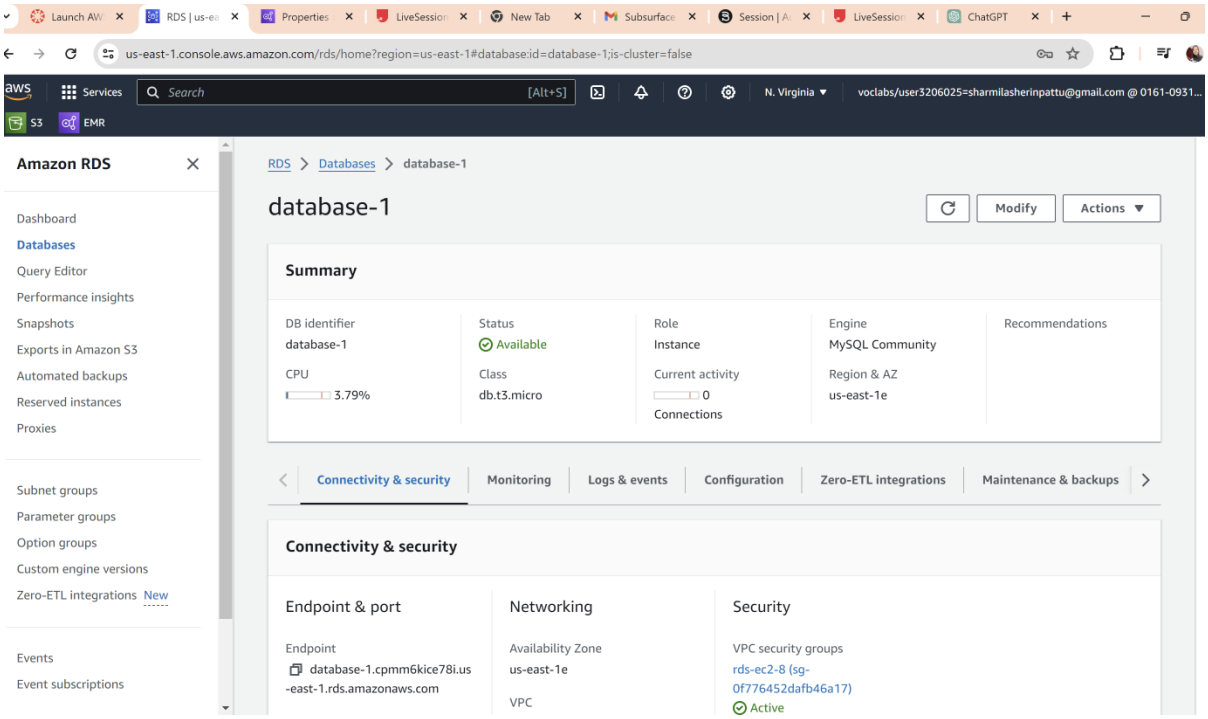
Task 1: AWS Environment Setup and Data Upload

Objective: Creating an RDS instance in AWS and uploading specific data files.

Instructions:

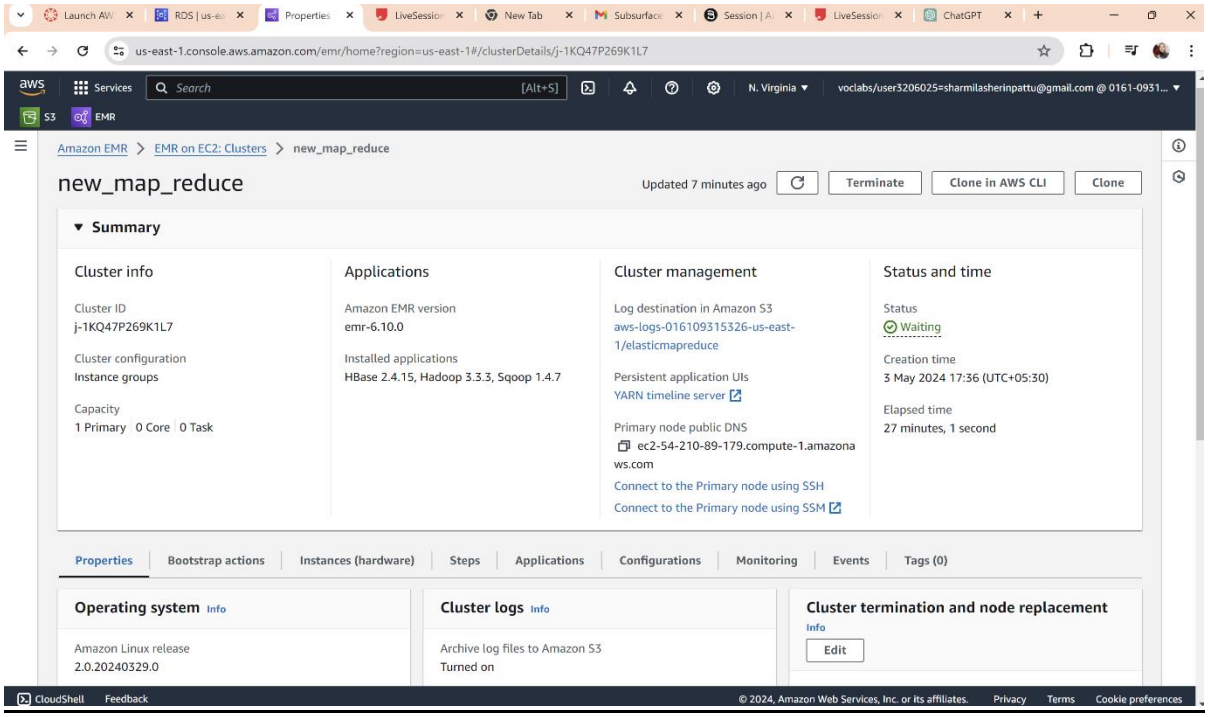
1. Begin by setting up an AWS environment and configuring an RDS instance.
2. Utilize the provided AWS account credentials.
3. Upload only two files, yellow_tripdata_2017-01.csv and yellow_tripdata_2017-02.csv, from the dataset due to its size.
4. Ensured a suitable schema is created for the datasets to facilitate their upload to the RDS instance.

1.RDS instance creation in AWS



2. EMR creation

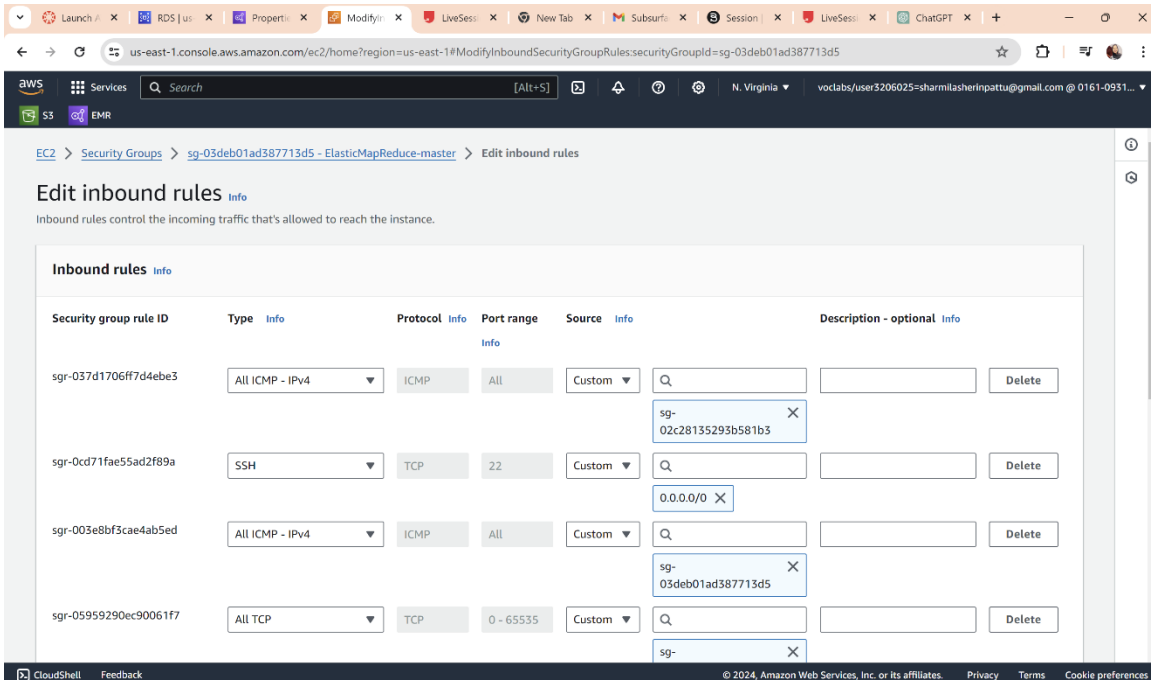
- **Including Apache Sqoop, Apache Hbase, Hadoop**



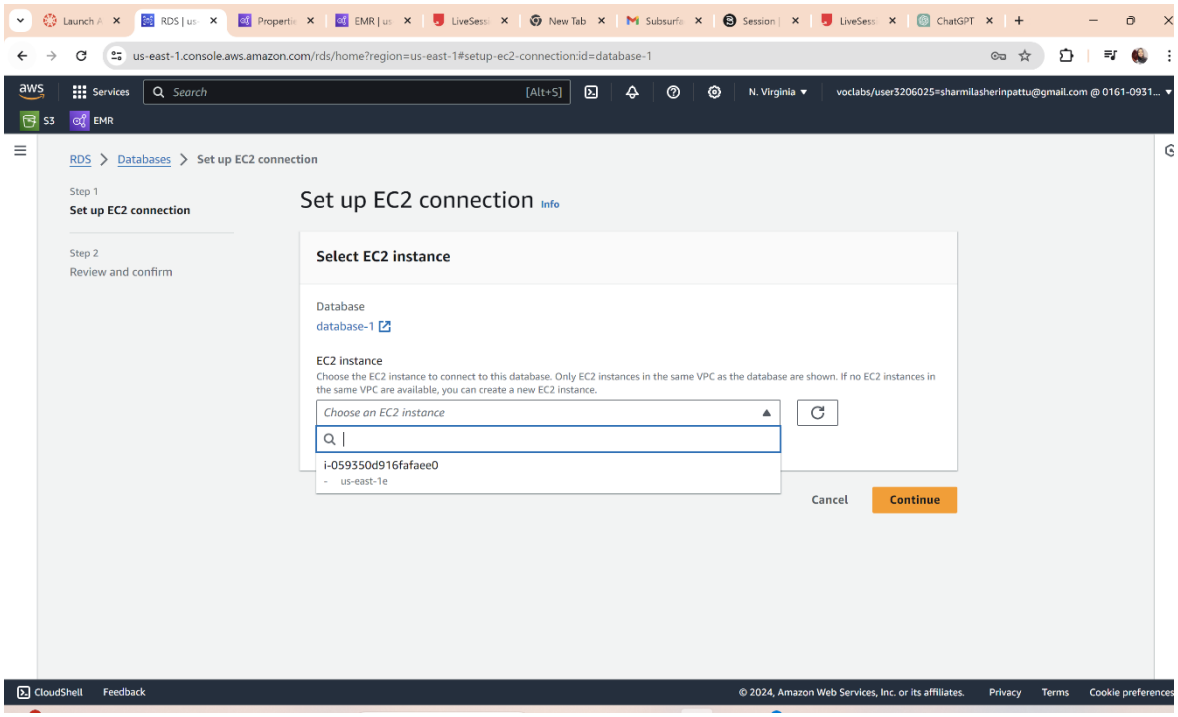
3: Connecting the RDS instance with the EMR instance

- To connect the RDS instance with the EMR instance, I adjusted the security group settings.
- Here's what I did: Accessed the AWS Management Console. Navigated to the EC2 section and selected "Security Groups".
- Identified the security group associated with the RDS instance.
- Edited the inbound rules to permit traffic from the EMR instance.

- This entailed specifying either the EMR instance's security group ID or its IP address range, along with the relevant port for database connectivity (such as 3306 for MySQL). Saved the modifications to the security group.
- Through these adjustments, I ensured secure connectivity between the EMR and RDS instances, facilitating data processing and analysis tasks.



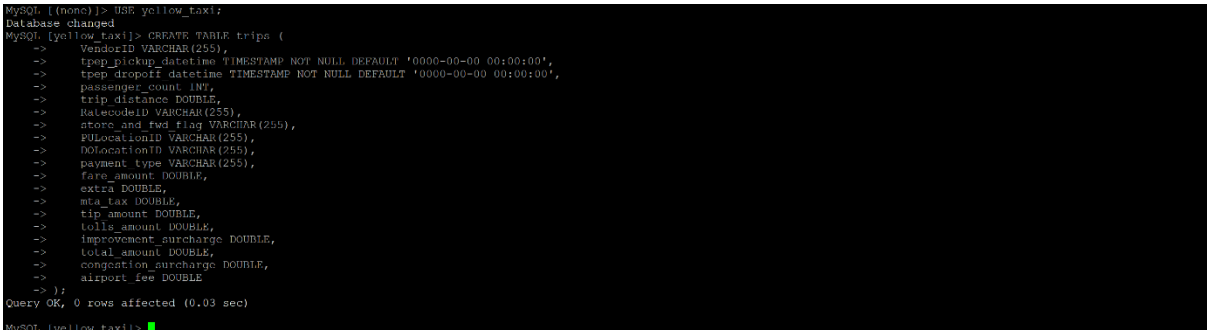
- Then we click on ‘Action’ button on RDS menu and then‘Set up EC2 connection’.



- To access the RDS instance through the EMR instance, we used the following command: **“mysql -h database-1.cpmm6kice78i.us-east-1.rds.amazonaws.com -P 3306 -u admin -p “**
- Upon executing the command, we were prompted to enter the password. After providing the password, the login process was completed successfully.

Following code was used to create database table:

```
-> CREATE DATABASE yellow_taxi;
USE yellow_taxi;
CREATE TABLE trips (
    VendorID VARCHAR(255),
    tpep_pickup_datetime TIMESTAMP NOT NULL DEFAULT '0000-00-00 00:00:00',
    tpep_dropoff_datetime TIMESTAMP NOT NULL DEFAULT '0000-00-00 00:00:00',
    passenger_count INT,
    trip_distance DOUBLE,
    RatecodeID VARCHAR(255),
    store_and_fwd_flag VARCHAR(255),
    PULocationID VARCHAR(255),
    DOLocationID VARCHAR(255),
    payment_type VARCHAR(255),
    fare_amount DOUBLE,
    extra DOUBLE,
    mta_tax DOUBLE,
    tip_amount DOUBLE,
    tolls_amount DOUBLE,
     improvement_surcharge DOUBLE,
    total_amount DOUBLE,
    congestion_surcharge DOUBLE,
    airport_fee DOUBLE
   );
```

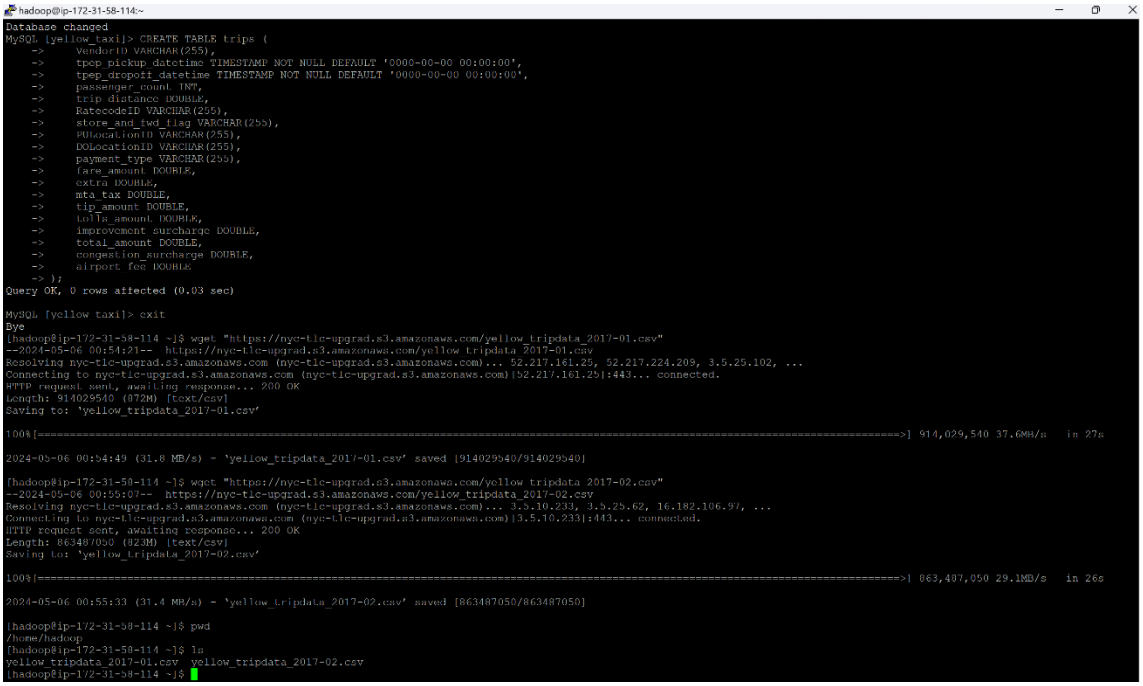


To download the necessary CSV files, I executed the following commands:

wget "https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-01.csv"
wget "https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-02.csv"

These commands fetched the specified CSV files from the provided URLs.

Just to showcase that table data was 0 before importing data to database:

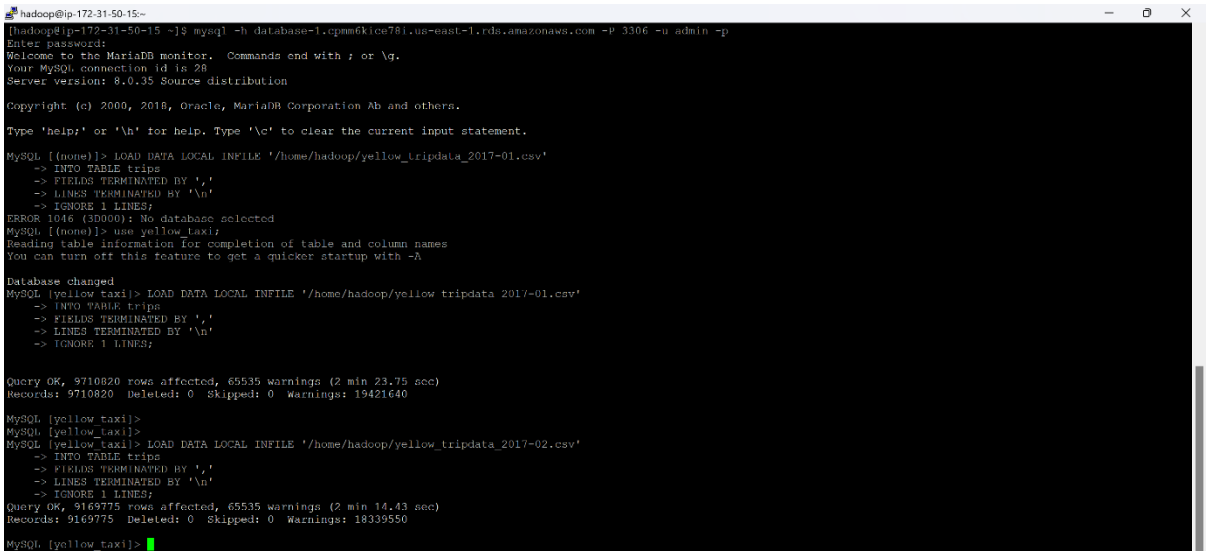


To load data into the MySQL table, I logged in and executed the following SQL commands:

LOAD DATA LOCAL INFILE '/home/hadoop/yellow_tripdata_2017-01.csv'
INTO TABLE trips
FIELDS TERMINATED BY ' , '
LINES TERMINATED BY ' \n '
IGNORE 1 LINES;

LOAD DATA LOCAL INFILE '/home/hadoop/yellow_tripdata_2017-02.csv'
INTO TABLE trips
FIELDS TERMINATED BY ' , '
LINES TERMINATED BY ' \n '
IGNORE 1 LINES;

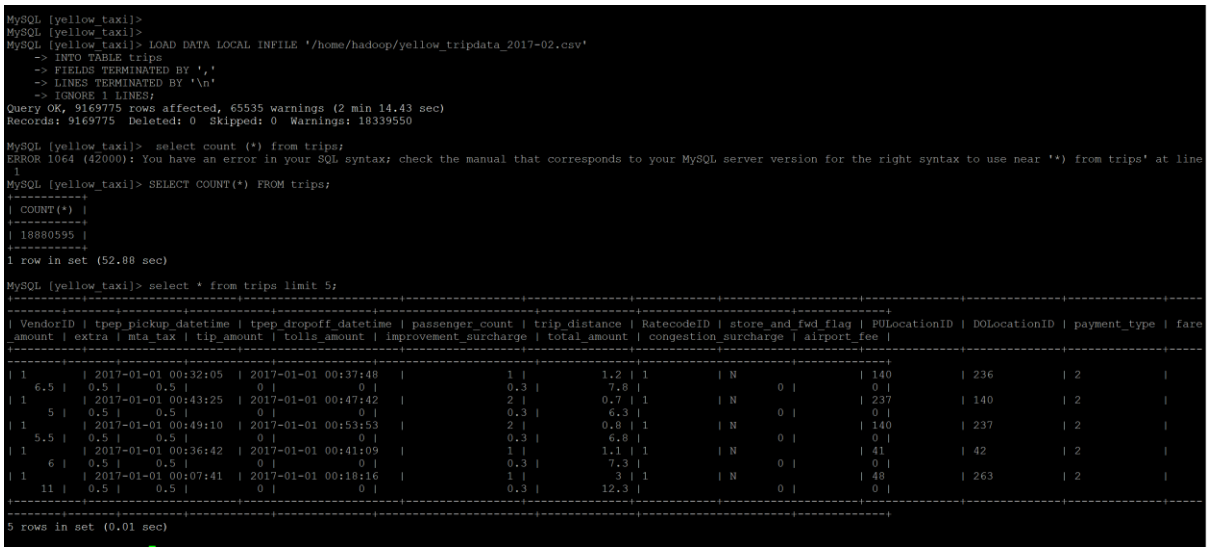
These commands imported the data from the specified CSV files into the MySQL table “trips”



Confirming that data is loaded: to do this, we run simple SQL queries:

> select count (*) from trips;

> select * from trips limit 5;



After Importing the final values in dataset is around : 18880595