

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

The screenshot displays the AWS Management Console interface for an Amazon RDS instance. The browser address bar shows the URL: `us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#database-id=database-1;is-cluster=false`. The console header includes the AWS logo, a search bar, and the user's profile information (N. Virginia, voclabs/user3050229=nitin.data1997@gmail.com @ 8913-7736-4898).

The left-hand navigation pane lists various RDS services, with 'Databases' selected. The main content area shows the configuration for 'database-1'. At the top, there are buttons for 'Refresh', 'Modify', and 'Actions'. Below this is a 'Summary' section with a table of key attributes:

DB identifier	Status	Role	Engine	Recommendations
database-1	⌚ Backing-up	Instance	MySQL Community	
CPU	Class	Current activity	Region & AZ	
3.37%	db.t3.micro	0 Connections	us-east-1a	

Below the summary, there are tabs for 'Connectivity & security', 'Monitoring', 'Logs & events', 'Configuration', 'Zero-ETL integrations', 'Maintenance & backups', and 'Tags'. The 'Connectivity & security' tab is active, showing a table with three columns: 'Endpoint & port', 'Networking', and 'Security'.

Endpoint & port	Networking	Security
Endpoint database-1.cn4wwwocm81j.us-east-1.rds.amazonaws.com	Availability Zone us-east-1a	VPC security groups default (sg-0ff09fae7981d99ff)
Port 3306	VPC vpc-080da7f02b96705e4	Active
	Subnet group	Publicly accessible Yes

The bottom of the screen shows the Windows taskbar with various application icons and the system clock indicating 11:39 AM on ENG.

2. EMR creation

- Including Apache Sqoop, Apache Hbase, Hadoop

The screenshot displays the AWS Management Console interface for an Amazon EMR cluster. The browser address bar shows the URL: `us-east-1.console.aws.amazon.com/emr/home?region=us-east-1#/clusterDetails/j-22BG06D05ZFKL`. The console header includes the AWS logo, a search bar, and navigation links for Services, Regions (N. Virginia), and the user profile (voclabs/user3050229=nitin.data1997@gmail.com @ 8913-7736-4898).

The main content area is titled "Mapreduce_Assignment" and shows it was updated less than a minute ago. Action buttons include "Relaunch to update", "Terminate", "Clone in AWS CLI", and "Clone".

Summary

Cluster info	Applications	Cluster management	Status and time
Cluster ID j-22BG06D05ZFKL	Amazon EMR version emr-5.30.1	Log destination in Amazon S3 aws-logs-891377364898-us-east-1/elasticmapreduce	Status Starting
Cluster configuration Instance groups	Installed applications HBase 1.4.13, Hadoop 2.8.5, Sqoop 1.4.7	Primary node public DNS ec2-44-197-187-60.compute-1.amazonaws.com Connect to the Primary node using SSH Connect to the Primary node using SSM	Creation time March 04, 2024, 11:33 (UTC+05:30)
Capacity 1 Primary 0 Core 0 Task			Elapsed time 5 minutes, 53 seconds

Properties | Bootstrap actions | Instances (Hardware) | Steps | Applications | Configurations | Monitoring | Events | Tags (0)

Cluster logs [Info](#)

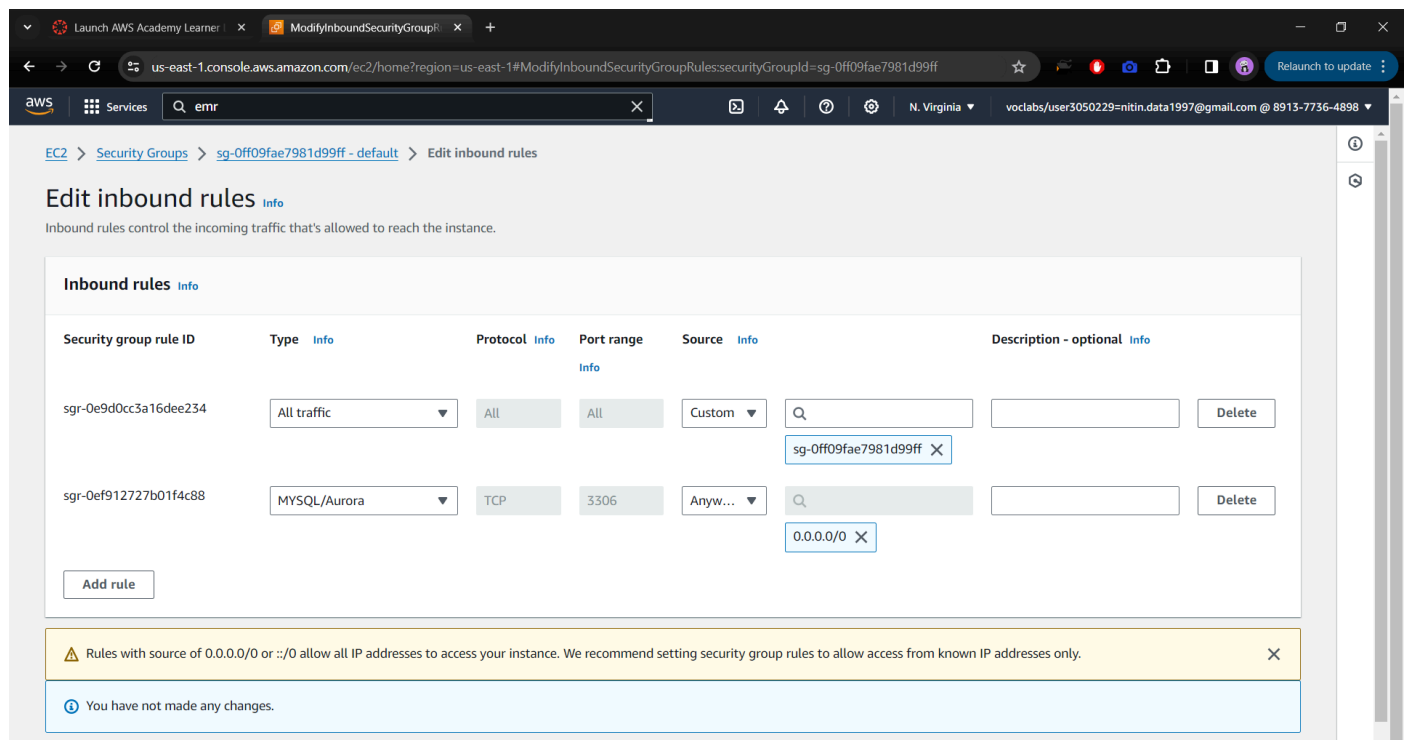
Archive log files to Amazon S3 Turned on	Encryption for logs Turned off
Amazon S3 location s3://aws-logs-891377364898-us-east-1/elasticmapreduce/	

Cluster termination [Info](#) [Edit](#)

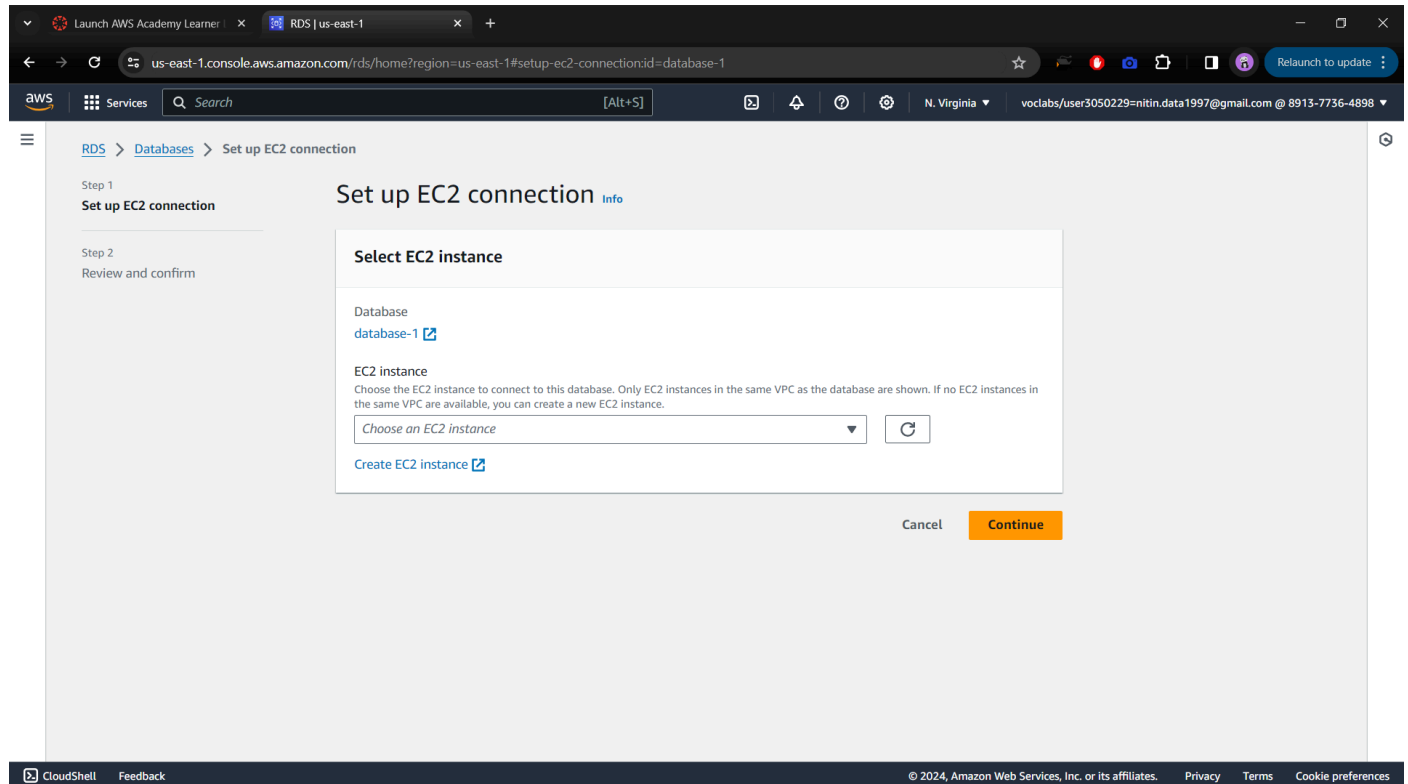
Termination option Manually terminate cluster	Idle time -
Termination protection Off	

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.cn4wwwocm81j.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.

 hadoop@ip-172-31-7-137:~

```
EE:::EEEEEEEE::E M:::M M:::M R:::R R:::R  
E:::E EEEE M:::M M:::M R:::RRRRRR:::R  
E:::E M:::M M:::M M:::M R:::R R:::R  
E:::EEEEEEEE M:::M M:::M M:::M R:::RRRRRR:::R  
E:::EEEE M:::M M:::M M:::M R:::R R:::R  
E:::E M:::M M:::M M:::M R:::RRRRRR:::R  
E:::E EEEE M:::M MMM M:::M R:::R R:::R  
EE:::EEEEEEEE::E M:::M M:::M R:::R R:::R  
E:::EEEEEEEE M:::M M:::M M:::M R:::R R:::R  
EEEEEEEEEEEEEEEE MMMMMM MMMMMM RRRRRR RRRRR
```

```
[hadoop@ip-172-31-7-137 ~]$ mysql -h database-1.cn4wvccm81j.us-east-1.rds.amazonaws.com -P 3306 --user admin -p  
Enter password:  
Welcome to the MariaDB monitor. Commands end with ; or \g.  
Your MySQL connection id is 24  
Server version: 8.0.28 Source distribution  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type 'C' to clear the current input statement.  
  
MySQL [(none)]> create database taxi_yellow  
->  
Query OK, 1 row affected (0.01 sec)  
  
MySQL [(none)]> use taxi_yellow  
Database changed  
MySQL [(taxi yellow)]> 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_charge DOUBLE,  
-> mta_tax DOUBLE,  
-> tip_amount DOUBLE,  
-> tolls_amount DOUBLE,  
-> improvement_surcharge DOUBLE,  
-> total_amount DOUBLE,  
-> congestion_surcharge DOUBLE,  
-> airport_fee DOUBLE
```

Following code was used to create database table:

-> Use taxi_yellow

```
->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:

```
hadoop@ip-172-31-7-137:~$
Bye
[hadoop@ip-172-31-7-137 ~]$ wget "https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-01.csv"
"https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-01.csv": Scheme missing.
[hadoop@ip-172-31-7-137 ~]$ ^Cet "https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-01.csv"
[hadoop@ip-172-31-7-137 ~]$ wget "https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-01.csv"
--2024-03-04 06:16:58-- https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-01.csv
Resolving nyc-tlc-upgrad.s3.amazonaws.com (nyc-tlc-upgrad.s3.amazonaws.com)... 54.231.135.97, 52.216.179.147, 52.217.174.121, ...
Connecting to nyc-tlc-upgrad.s3.amazonaws.com (nyc-tlc-upgrad.s3.amazonaws.com)|54.231.135.97|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 914029540 (872M) [text/csv]
Saving to: 'yellow_tripdata_2017-01.csv'

100%[=====>] 914,029,540 34.0MB/s in 25s

2024-03-04 06:17:24 (34.3 MB/s) - 'yellow_tripdata_2017-01.csv' saved [914029540/914029540]

[hadoop@ip-172-31-7-137 ~]$ wget "https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-02.csv"
--2024-03-04 06:17:39-- https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-02.csv
Resolving nyc-tlc-upgrad.s3.amazonaws.com (nyc-tlc-upgrad.s3.amazonaws.com)... 54.231.198.1, 52.217.123.41, 16.182.64.137, ...
Connecting to nyc-tlc-upgrad.s3.amazonaws.com (nyc-tlc-upgrad.s3.amazonaws.com)|54.231.198.1|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 863487050 (823M) [text/csv]
Saving to: 'yellow_tripdata_2017-02.csv'

100%[=====>] 863,487,050 38.7MB/s in 22s

2024-03-04 06:18:02 (36.8 MB/s) - 'yellow_tripdata_2017-02.csv' saved [863487050/863487050]

[hadoop@ip-172-31-7-137 ~]$ pwd
/home/hadoop
[hadoop@ip-172-31-7-137 ~]$ mysql -h database-1.cn4wwocm8lj.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MySQL connection id is 26
Server version: 8.0.28 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> use taxi_yellow
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [taxi_yellow]> select count(*) from trips;
+-----+
| count(*) |
+-----+
|          0 |
+-----+
1 row in set (0.00 sec)
```

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

```
```sql
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 named "trips".

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

Query OK, 9710820 rows affected, 65535 warnings (2 min 33.04 sec)
Records: 9710820 Deleted: 0 Skipped: 0 Warnings: 19421640

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

Query OK, 9169775 rows affected, 65535 warnings (2 min 41.22 sec)
Records: 9169775 Deleted: 0 Skipped: 0 Warnings: 18339550

MySQL [taxi_yellow]>
MySQL [taxi_yellow]> select count(*) from trips;
+-----+
| count(*) |
+-----+
| 18880595 |
+-----+
1 row in set (1 min 4.40 sec)
```

After Importing the final values in dataset is around : 18880595

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

```
> select count (*) from trips;
```

```
> select * from trips limit 5;
```

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

Query OK, 9710820 rows affected, 65535 warnings (2 min 33.04 sec)
Records: 9710820 Deleted: 0 Skipped: 0 Warnings: 19421640

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

Query OK, 9169775 rows affected, 65535 warnings (2 min 41.22 sec)
Records: 9169775 Deleted: 0 Skipped: 0 Warnings: 18339550

MySQL [taxi_yellow]>
MySQL [taxi_yellow]> select count(*) from trips;
+-----+
| count(*) |
+-----+
| 18880595 |
+-----+

1 row in set (1 min 4.40 sec)
```

```

VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance | RatecodeID | store_and_fwd_flag | PULocationID | DOLocationID | payment_type | fare_amount | extra | mta_tax | tip_amount | tolls_amount | improvement_surcharge | total_amount | Airport_fee
1 | 2017-01-01 00:32:05 | 2017-01-01 00:37:48 | 1 | 1.2 | 1 | N | 140 | 236 | 2 | 6.5 | 0 | 0 | 0 | 0 | 0.3 | 7.8 | 0
1 | 2017-01-01 00:43:25 | 2017-01-01 00:47:42 | 1 | 0.7 | 1 | N | 237 | 140 | 2 | 5 | 0 | 0 | 0 | 0 | 0.3 | 6.3 | 0
1 | 2017-01-01 00:49:10 | 2017-01-01 00:53:53 | 1 | 0.8 | 1 | N | 140 | 237 | 2 | 5.5 | 0 | 0 | 0 | 0 | 0.3 | 6.8 | 0
1 | 2017-01-01 00:36:42 | 2017-01-01 00:41:09 | 1 | 1.1 | 1 | N | 41 | 42 | 2 | 6 | 0 | 0 | 0 | 0 | 0.3 | 7.3 | 0
1 | 2017-01-01 00:07:41 | 2017-01-01 00:18:16 | 1 | 3 | 1 | N | 48 | 263 | 2 | 11 | 0 | 0 | 0 | 0 | 0.3 | 12.3 | 0
5 rows in set (0.03 sec)

```