

Rajiya Casestudy AWS

Task 1

Using mpg dataset create linear regression between any two input columns and develop Machine Learning model and deploy on Flask.

Running the app.py on command prompt

```
Anaconda Prompt (Anaconda3)
(base) C:\Users\rproddatur>cd C:\Users\rproddatur\Desktop\rajiya casestudy
(base) C:\Users\rproddatur\Desktop\rajiya casestudy>python app.py
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with windowsapi reloader
* Debugger is active!
* Debugger PIN: 176-988-005
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [01/Dec/2021 18:33:12] "[37mGET / HTTP/1.1" 200 -
C:\Users\rproddatur\Anaconda3\lib\site-packages\sklearn\utils\validation.py:63: FutureWarning: Arrays of bytes/strings i
s being converted to decimal numbers if dtype='numeric'. This behavior is deprecated in 0.24 and will be removed in 1.1
(renaming of 0.26). Please convert your data to numeric values explicitly instead.
  return f(*args, **kwargs)
127.0.0.1 - - [01/Dec/2021 18:33:52] "[37mPOST /predict HTTP/1.1" 200 -
forrtl: error (200): program aborting due to control-C event
Image                PC                Routine              Line          Source
libifcoremd.dll      00007FFF12F3B58      Unknown              Unknown       Unknown
KERNELBASE.dll       00007FF87ED2B9D3      Unknown              Unknown       Unknown
KERNEL32.DLL         00007FF87F497034      Unknown              Unknown       Unknown
ntdll.dll            00007FF881342651      Unknown              Unknown       Unknown
forrtl: error (200): program aborting due to control-C event
Image                PC                Routine              Line          Source
libifcoremd.dll      00007FFF12F3B58      Unknown              Unknown       Unknown
KERNELBASE.dll       00007FF87ED2B9D3      Unknown              Unknown       Unknown
KERNEL32.DLL         00007FF87F497034      Unknown              Unknown       Unknown
ntdll.dll            00007FF881342651      Unknown              Unknown       Unknown
```

Predict the horsepower on the basis of mpg and cylinders

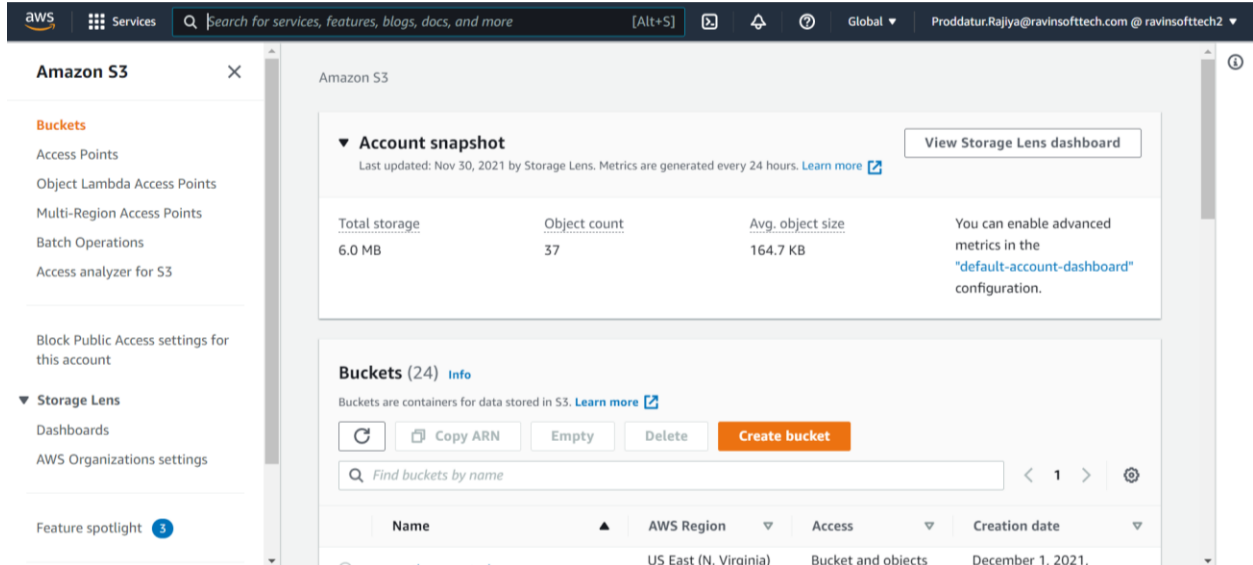
Predict the horspower on the basis of mpg and cylinders

The predicted price of the house is \$ 1855.56

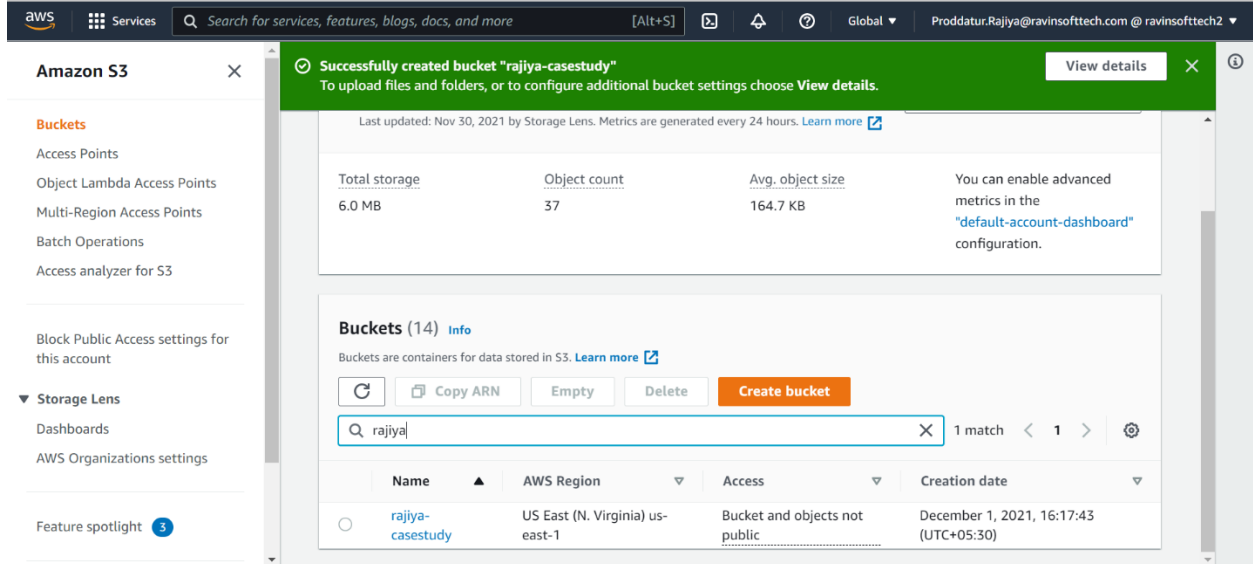
Task 2

AWS Task:

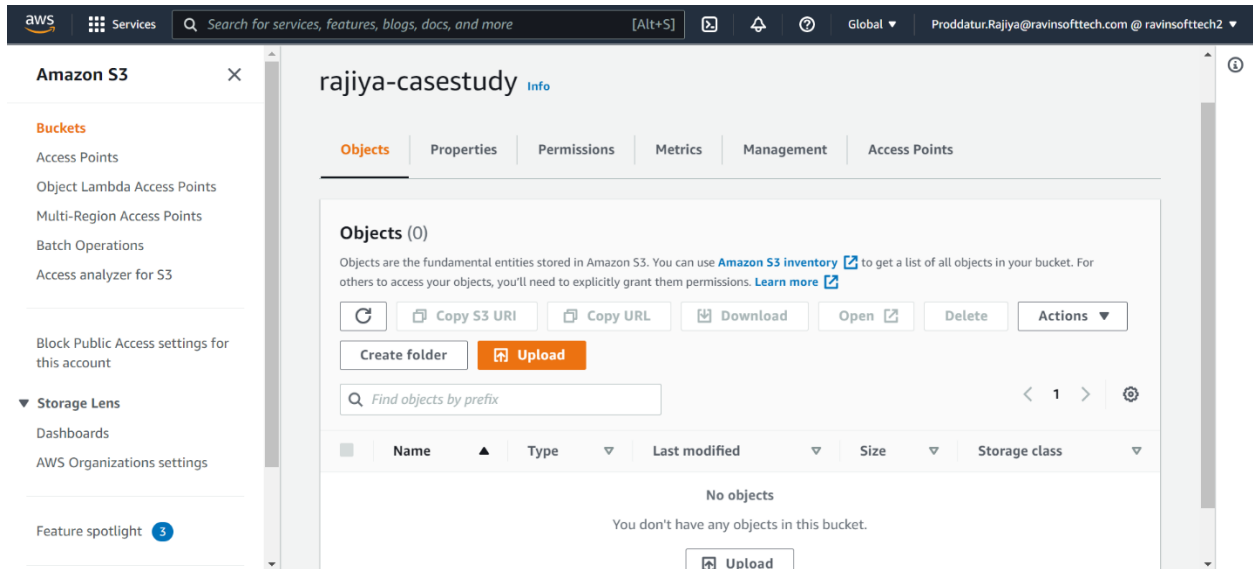
In amazon s3 page we go to “Buckets”



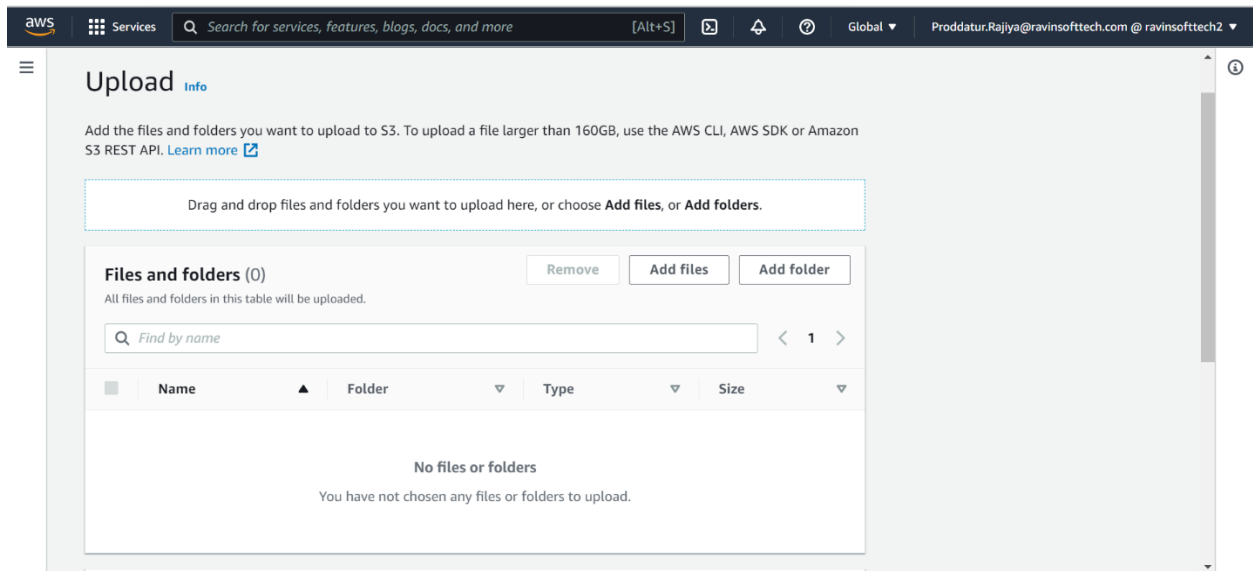
In the above page by clicking “Create bucket” we created s3 bucket with name as “rajiya-casestudy”



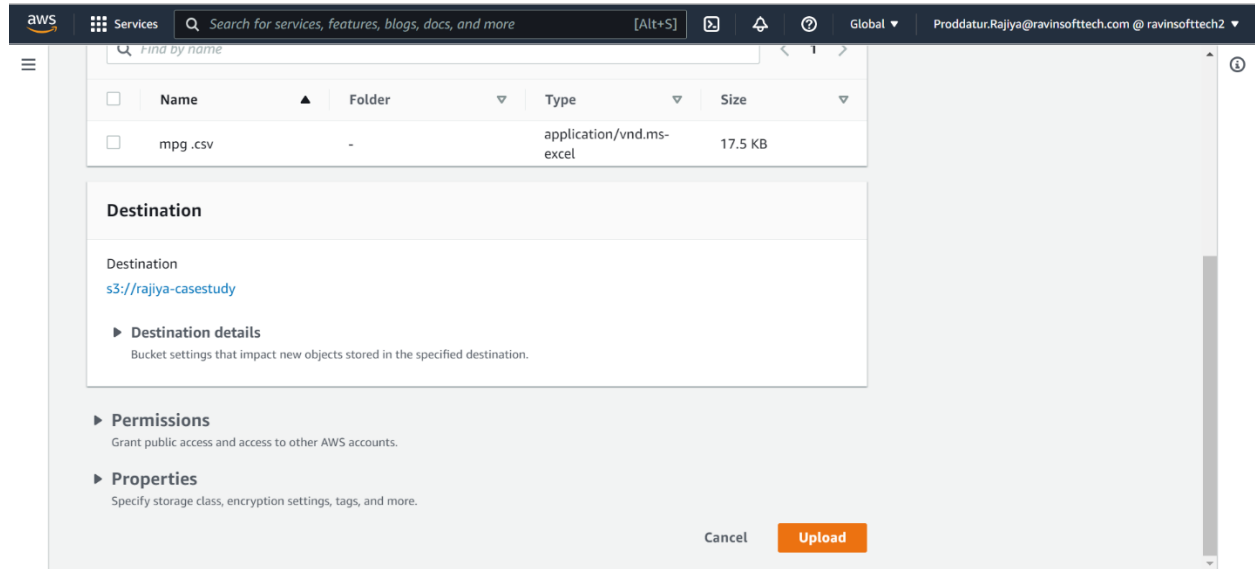
Now we can start uploading files in to s3 bucket



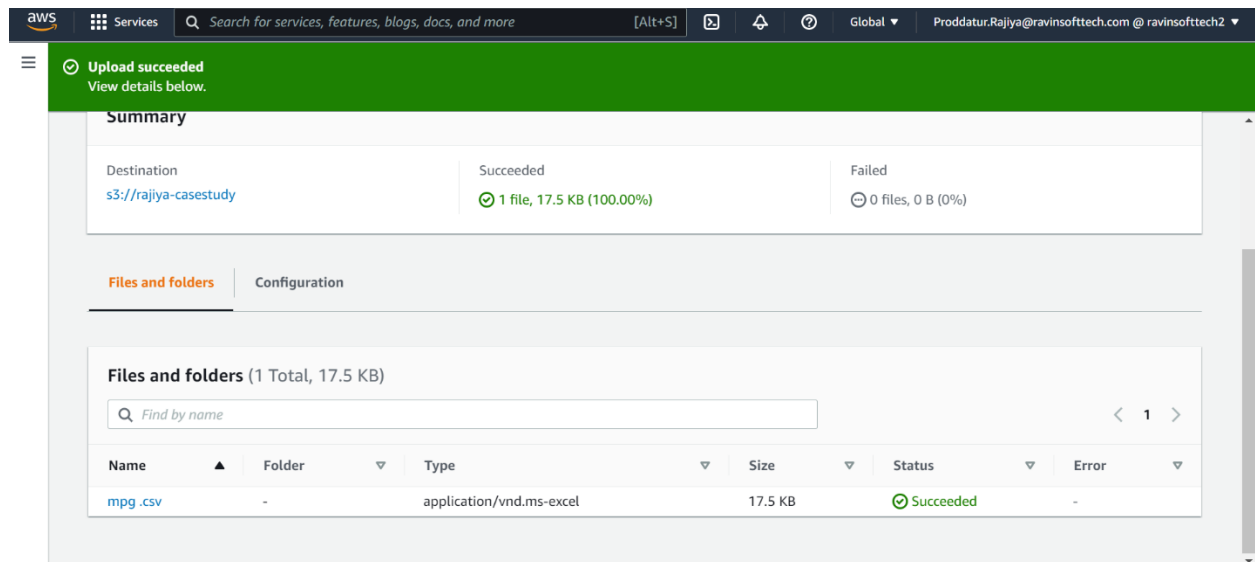
By clicking on “Upload” button the following page will open



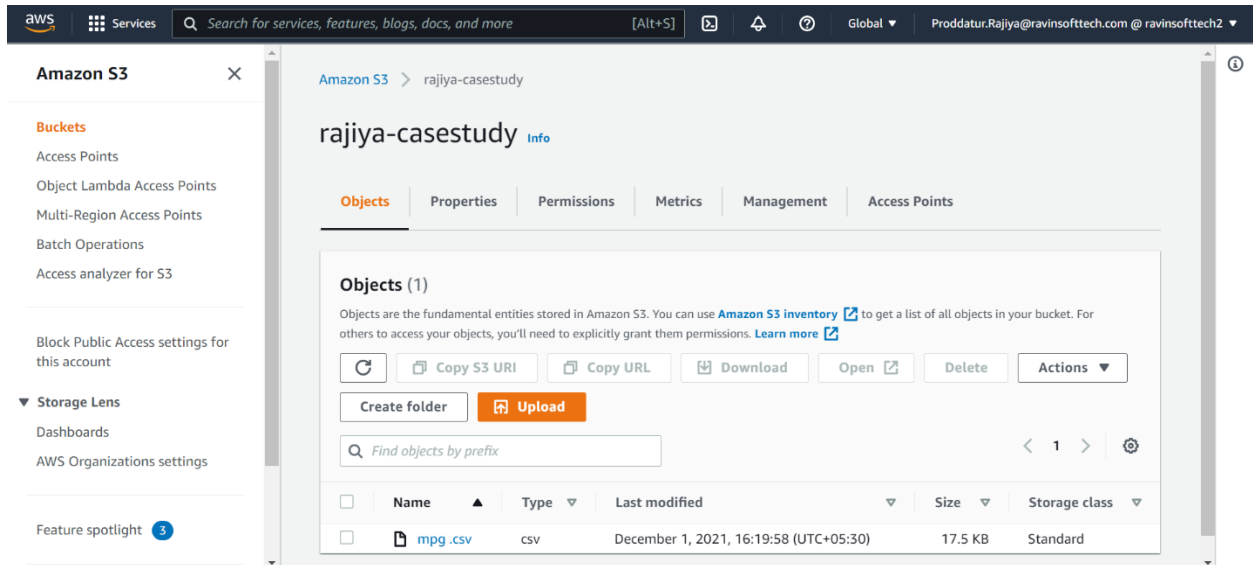
We can upload by clicking on “Add files” or “Drag and drop the files”



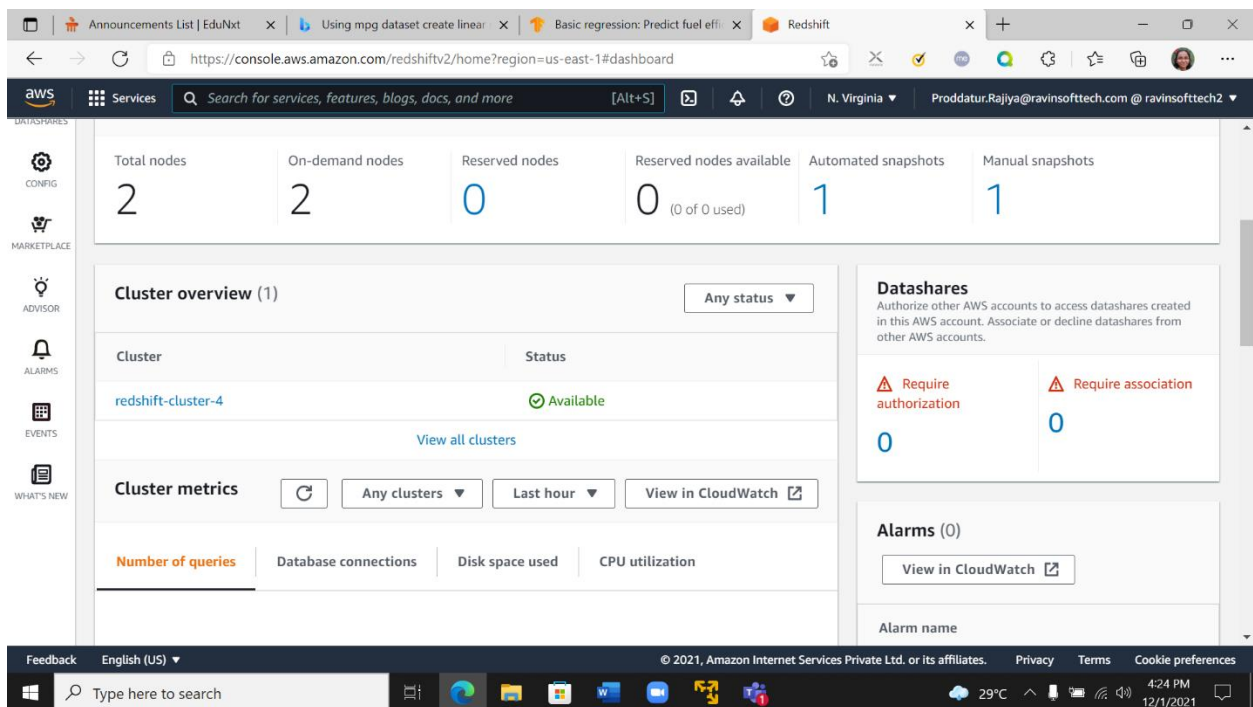
By clicking on the “Upload” button the files will get uploaded



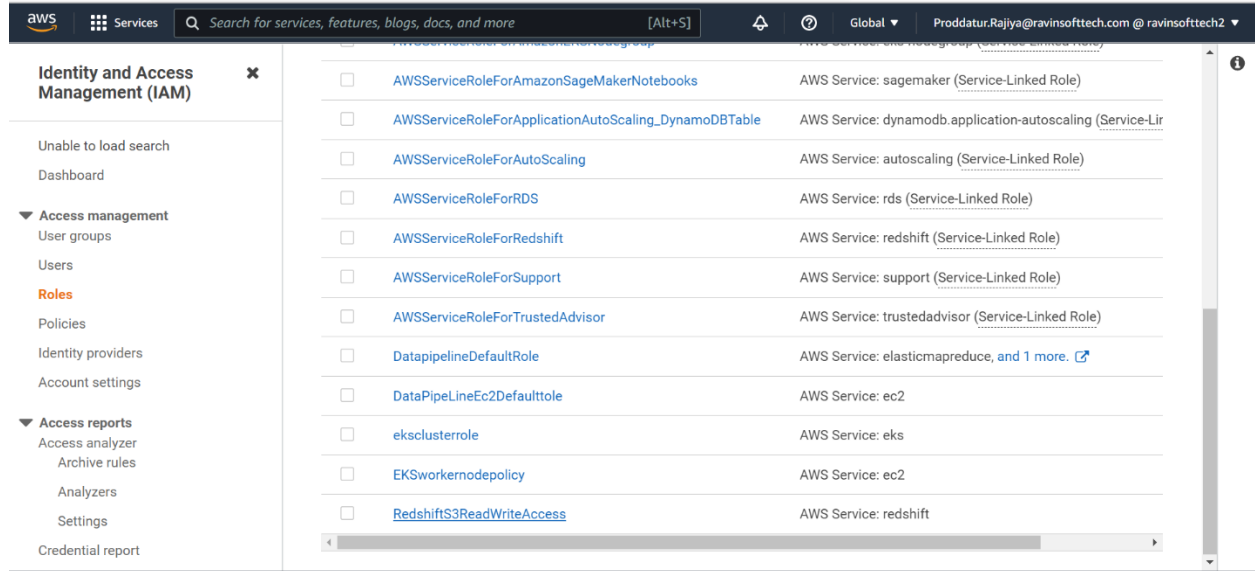
Now we can copy the location of the bucket for further use.



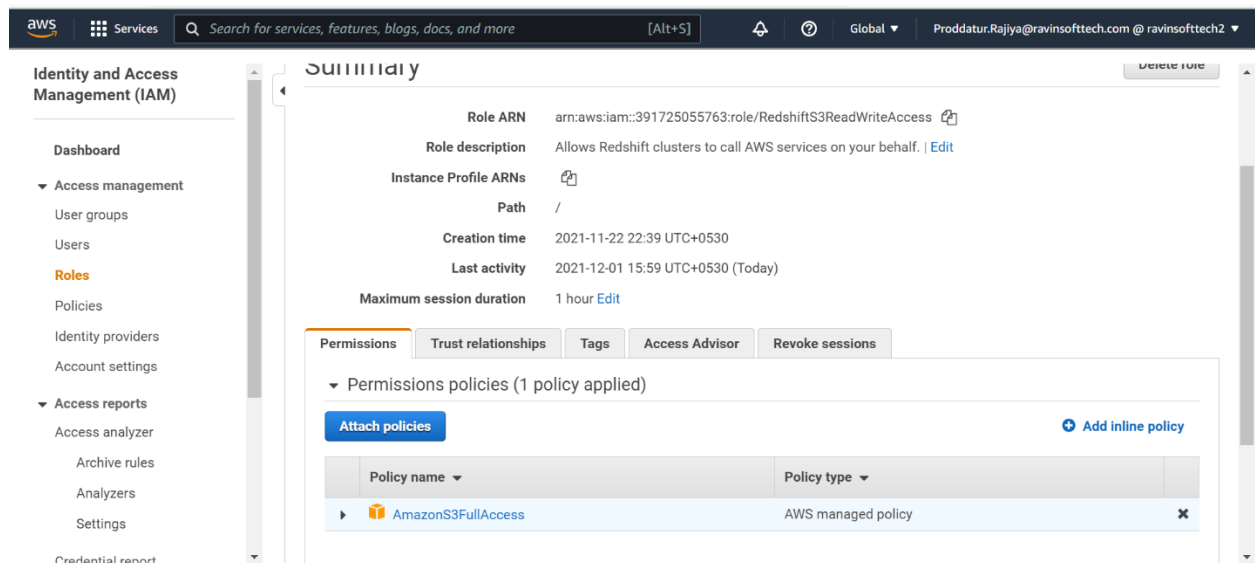
Now we need to go to “Amazon Redshift” and create a cluster
As we already have a cluster we are making use of it



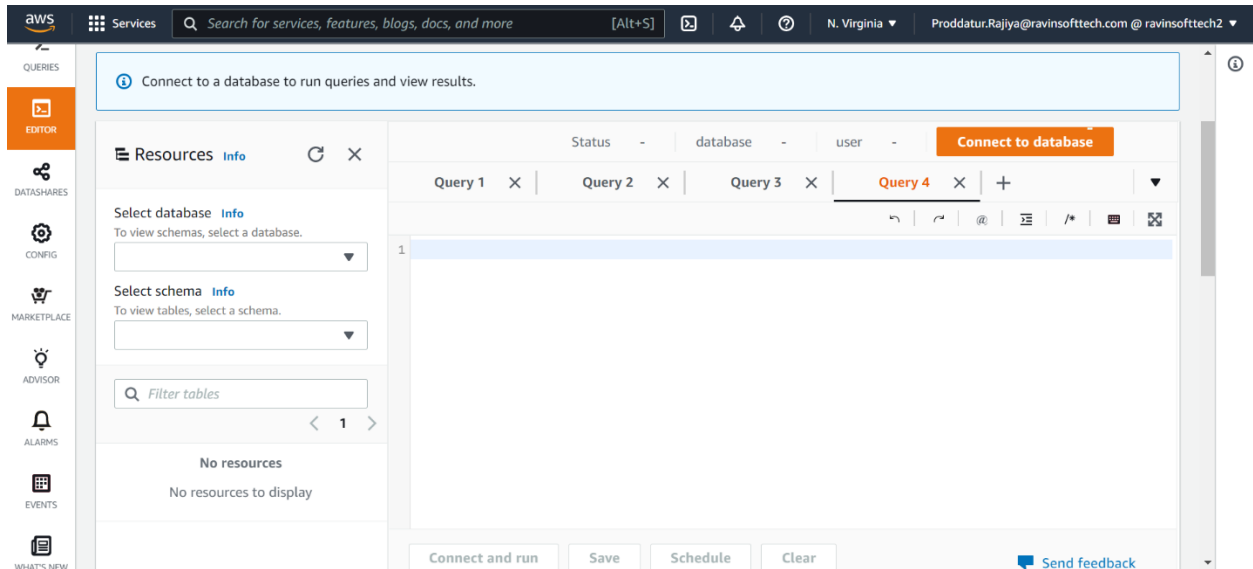
Opening the “iam console” and choosing the created “RedshiftS3ReadWriteAccess”



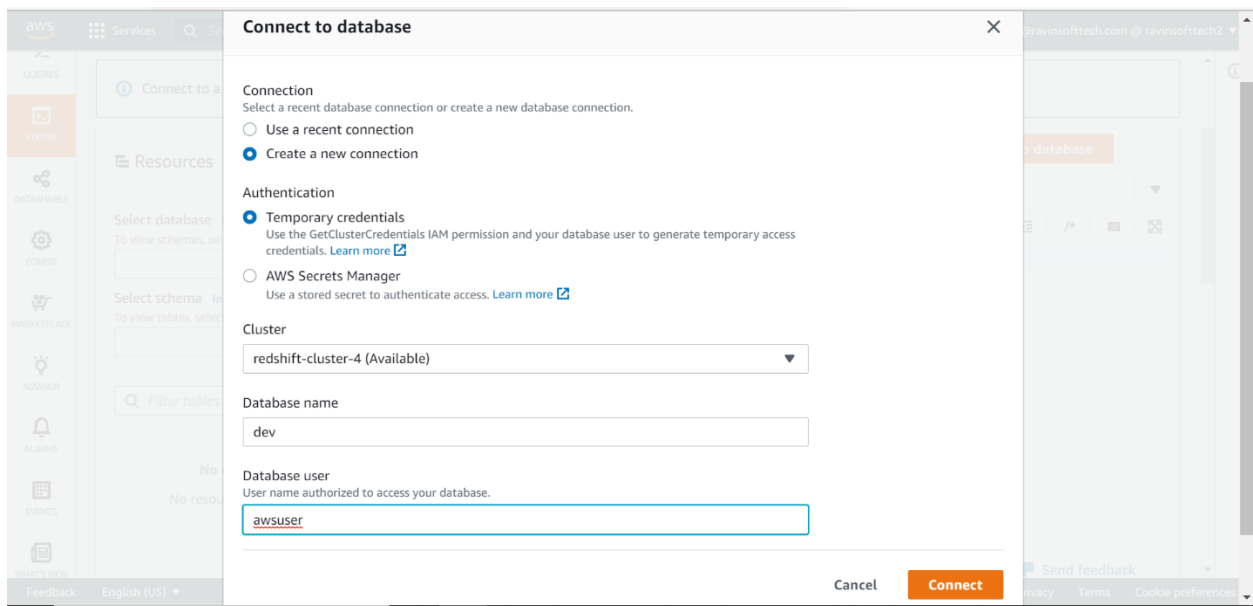
By clicking on it we get the following page and copy the “Role ARN” for further use



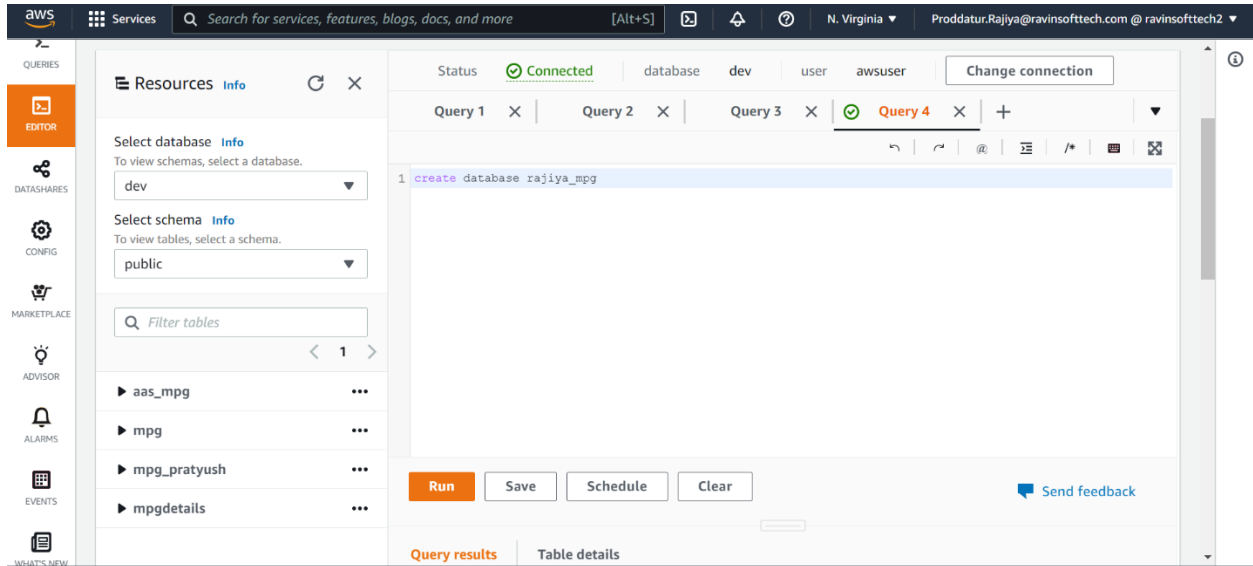
Now go the Amazon Redshift console in “Editors” open query



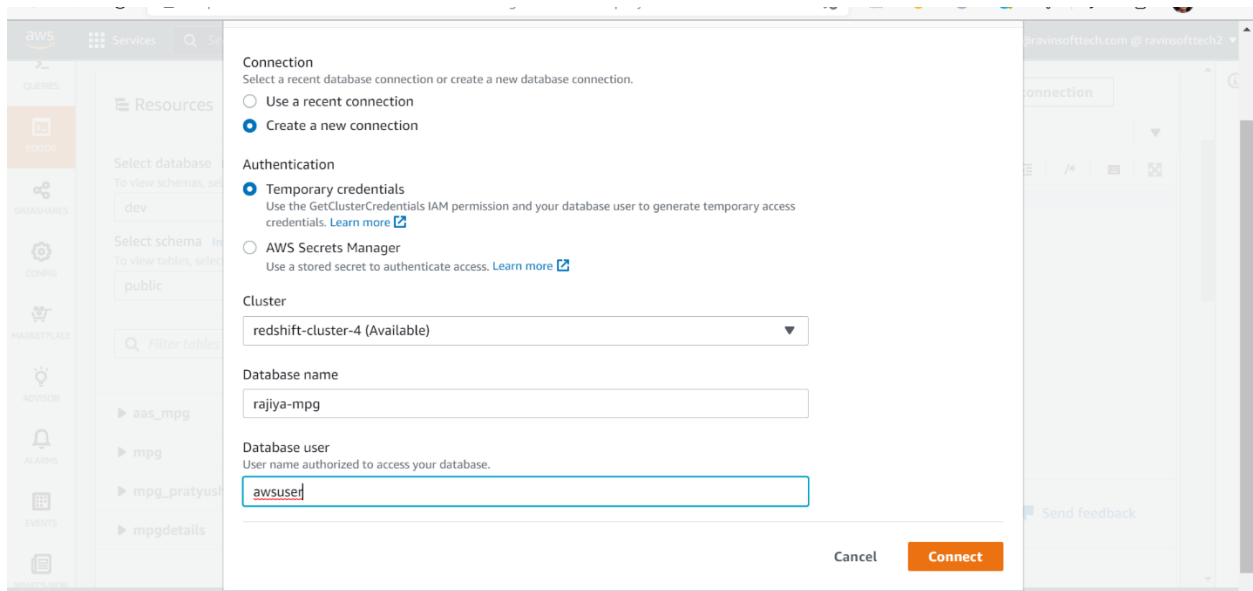
Connect to the database



Creating my own database



Connecting to my own database



Creating the table in redshift editor

The screenshot shows the AWS Redshift console editor interface. The top navigation bar includes the AWS logo, a search bar, and the user's profile. The left sidebar contains various navigation options like 'QUERIES', 'EDITOR', 'DATASHARES', 'CONFIG', 'MARKETPLACE', 'ADVISOR', 'ALARMS', and 'EVENTS'. The main content area is divided into two panes. The left pane, titled 'Resources', shows the 'Select database' dropdown set to 'rajiya_mpg' and the 'Select schema' dropdown set to 'public'. Below these, there is a 'Filter tables' search bar and a list of tables, with 'mpg' selected. The right pane, titled 'Query 1', contains a SQL statement to create a table named 'mpg' with the following columns: 'mpg float', 'cylinders int', 'displacement float', 'horsepower float', 'weight float', 'acceleration float', 'model_year int', 'origin int', and 'car_name VARCHAR(50)'. The status bar at the bottom indicates 'Connected' and 'Change connection' options. The bottom of the editor has buttons for 'Run', 'Save', 'Schedule', and 'Clear', along with a 'Send feedback' link.

```
1 CREATE TABLE mpg
2 (
3     mpg float,
4     cylinders int,
5     displacement float,
6     horsepower float,
7     weight float,
8     acceleration float,
9     model_year int,
10    origin int,
11    car_name VARCHAR(50)
12 );
```

performing copy operation

The screenshot shows the AWS Redshift console editor interface, similar to the previous one. The left pane shows the 'Select database' dropdown set to 'rajiya_mpg' and the 'Select schema' dropdown set to 'public'. The 'Filter tables' search bar is empty, and the 'mpg' table is still selected. The right pane, titled 'Query 2', contains a SQL statement to perform a copy operation from an S3 bucket to the 'mpg' table. The status bar at the bottom indicates 'Connected' and 'Change connection' options. The bottom of the editor has buttons for 'Run', 'Save', 'Schedule', and 'Clear', along with a 'Send feedback' link.

```
1 copy mpg from 's3://rajiya-casestudy/mpg .csv'
2 credentials 'aws_iam_role=arn:aws:iam::391725055763:role/RedshiftS3ReadWriteAccess'
3 delimiter ',' region 'us-east-1'
4 IGNOREHEADER 1;
```

Viewing the schema

The screenshot shows the AWS Glue console interface. At the top, there's a navigation bar with the AWS logo, 'Services' link, a search bar, and user information. The main content area displays the 'mpg' table schema. It includes a 'Filter data' search bar, a table with columns 'Columns', 'Type', 'Nullable', 'Length', and 'Precision', and a 'Show schema' button.

| Columns | Type | Nullable | Length | Precision |
|--------------|--------|----------|--------|-----------|
| mpg | float8 | true | 17 | 17 |
| cylinders | int4 | true | 10 | 10 |
| displacement | float8 | true | 17 | 17 |
| horsepower | float8 | true | 17 | 17 |
| weight | float8 | true | 17 | 17 |
| acceleration | float8 | true | 17 | 17 |
| model_year | int4 | true | 10 | 10 |
| origin | int4 | true | 10 | 10 |

Previewing the data

The screenshot shows the AWS Glue console interface with the 'mpg' table data previewed. It includes a 'Search rows' search bar, a table with columns 'mpg', 'cylinders', 'displacement', 'horsepower', 'weight', and 'acceleration', and a 'Preview data' button.

| mpg | cylinders | displacement | horsepower | weight | acceleration |
|-----|-----------|--------------|------------|--------|--------------|
| 18 | 8 | 307 | 130 | 3504 | 12 |
| 15 | 8 | 350 | 165 | 3693 | 11.5 |
| 18 | 8 | 318 | 150 | 3436 | 11 |
| 16 | 8 | 304 | 150 | 3433 | 12 |
| 17 | 8 | 302 | 140 | 3449 | 10.5 |