End-to-End Fisheries ETL Pipeline Using AWS Services

Overview and objectives

Throughout this course, I have completed hands-on labs, where you used the features of different AWS services to practice ingesting large datasets, transforming them, and extracting information from them.

In this capstone project, I was challenged to build a solution that uses many AWS services that are familiar to me without being given step-by-step guidance. Specific sections of the assignment were challenging.

The dataset

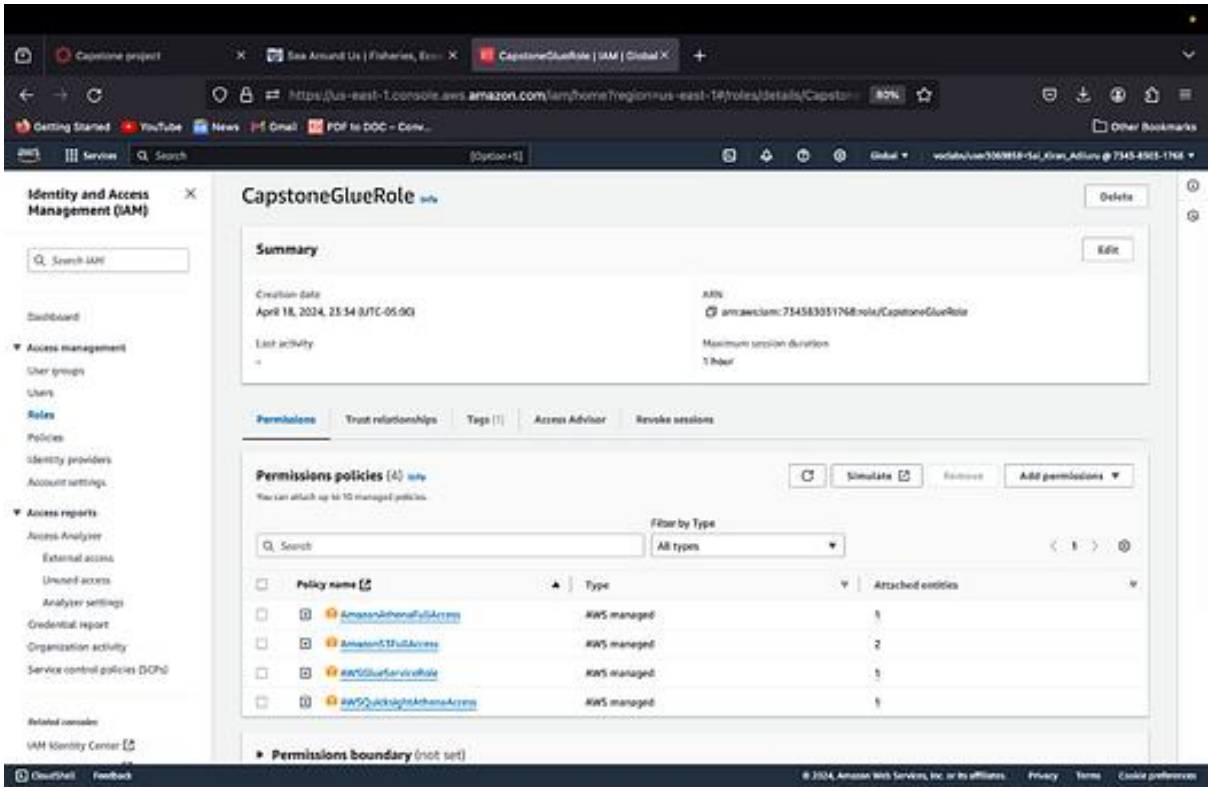
The Sea Around Us website provides a dataset with extensive historical information about fisheries in all parts of every ocean globally. The data includes information about yearly fishery catches from 1950 to 2018.

I worked with three data files from the Sea Around Us website:

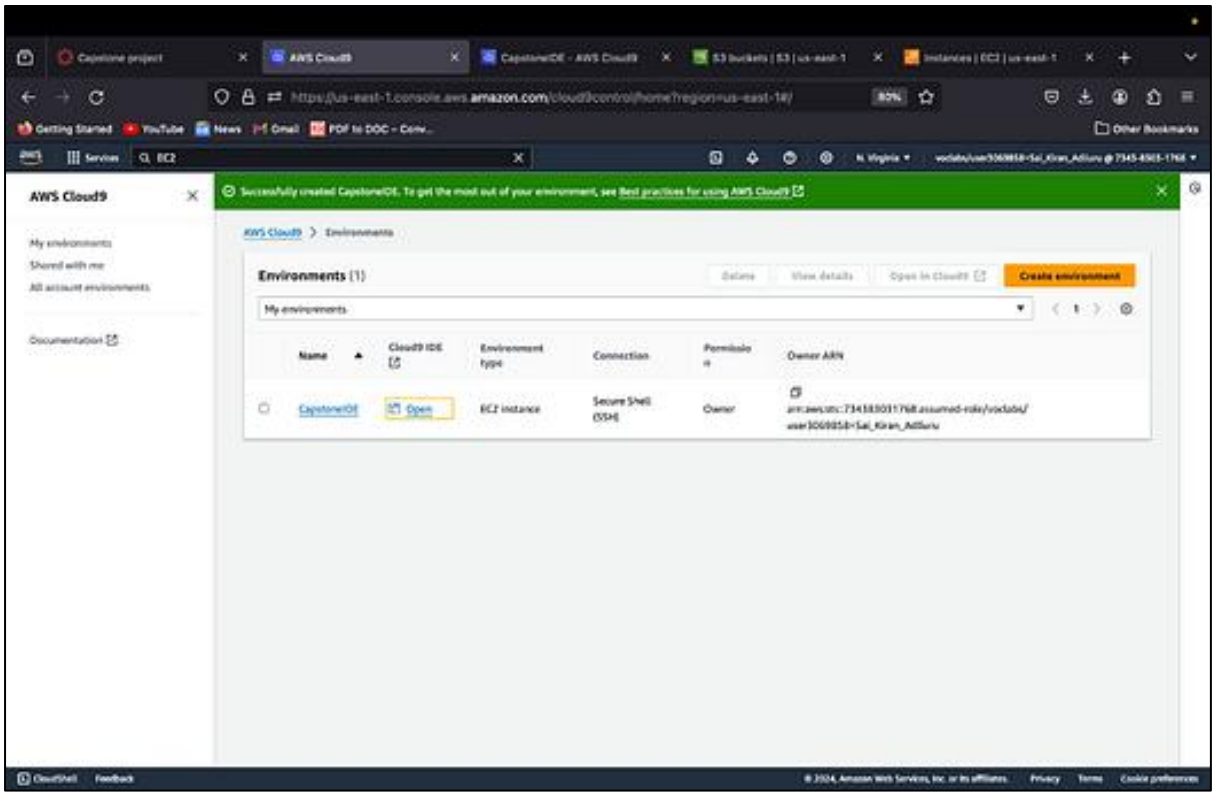
- The first file contains data from *all open seas areas*.
- The second file contains data from a *single open seas area* in the Pacific ocean, referred to as *Pacific, Western Central*, which is not far from Fiji and many other countries.
- The third file contains data from the *EEZ* of a single country (Fiji), which is near the Pacific, Western Central open seas area.

Configuired the development environment

I set up your development environment, by taking a closer look into the IAM roles provided.

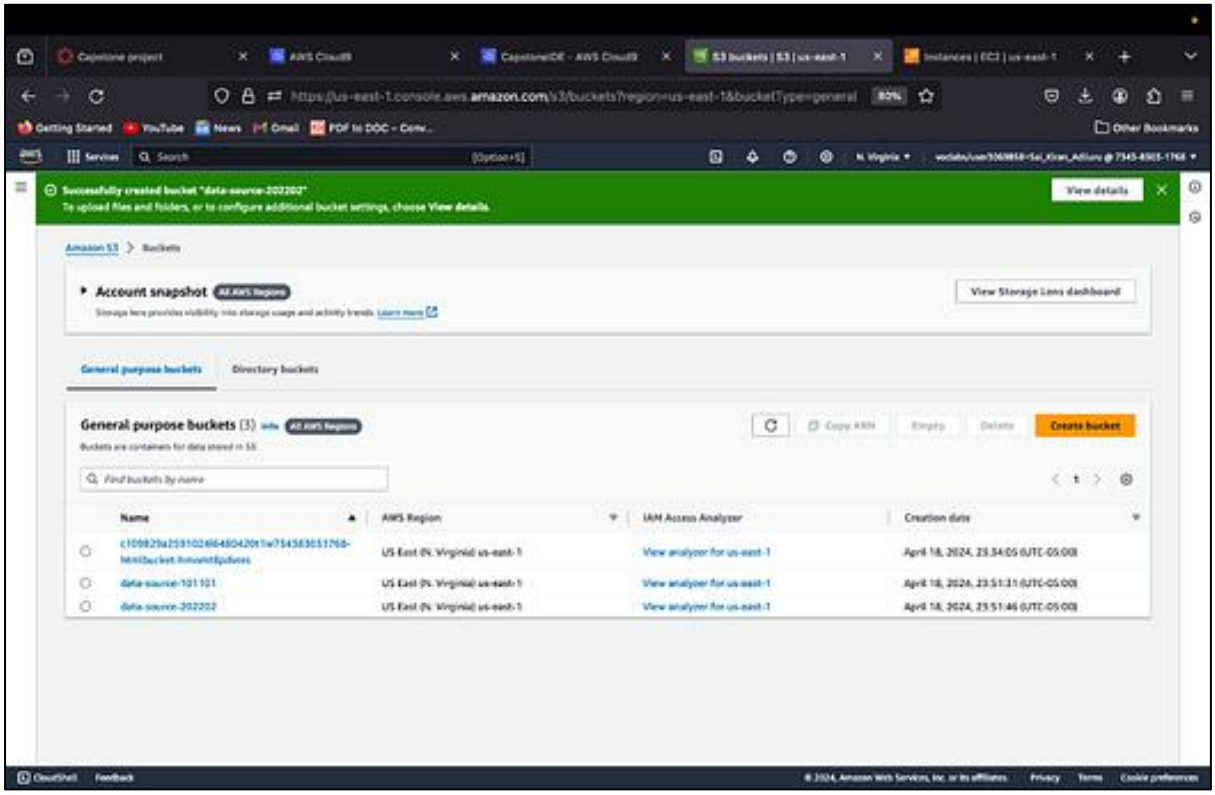


I created an AWS Cloud9 environment, with the name CapstoneIDE



And a new EC2 instance for the environment, and used a t2.micro instance. Deployed the instance to support *SSH* connections to the *Capstone VPC*, in the *Capstone public subnet.*

Created two S3 buckets in US-East regions. With the bucket names as follows : Data-Source-101101 & Query-Results-101101.



Downloaded the three .csv source data files, by the following commands in the terminal of your AWS Cloud9 IDE:

For the first file :

wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACDENG-1-91570/lab-capstone/s3/SAU-GLOBAL-1-v48-0.csv

Second file:

wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACDENG-1-91570/lab-capstone/s3/SAU-HighSeas-71-v48-0.csv

Third file:

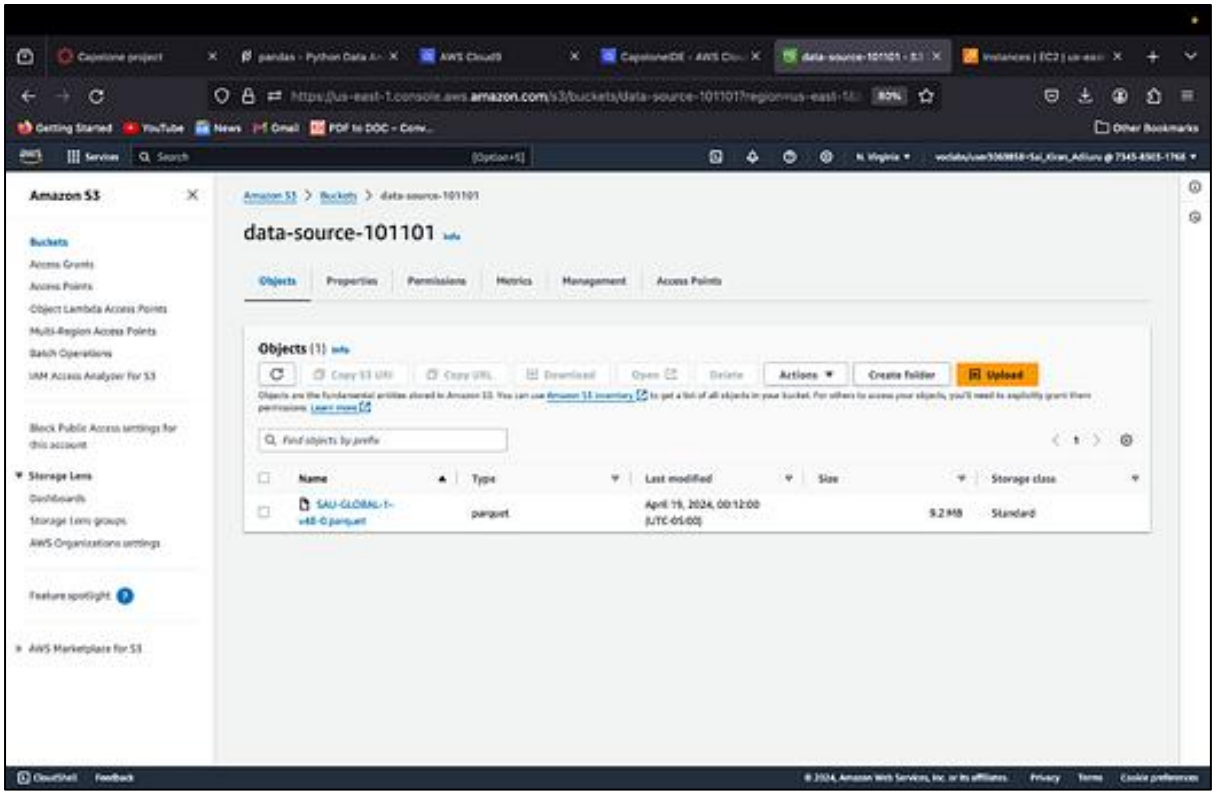
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACDENG-1-91570/lab-capstone/s3/SAU-EEZ-242-v48-0.csv

converted the csv files to parquet files using python in CloudIDE , using Pandas library to read the CSV file, and fastparquet are installed using Pip3 . Fastparquet as the backend for writing Parquet files.

```
import pandas as pd


df = pd.read_csv('SAU-GLOBAL-1-v48-0.csv')
gf= pd.read_csv('SAU-HighSeas-71-v48-0.csv')


df.to_parquet('SAU-GLOBAL-1-v48-0.parquet')
gf.to_parquet('SAU-HighSeas-71-v48-0.parquet')


exit()
```

To upload the files into S3 bucket the I used the following command in the cloud9 terminal

```
aws s3 cp SAU-GLOBAL-1-v48–0.parquet s3://data-source-101101/
```

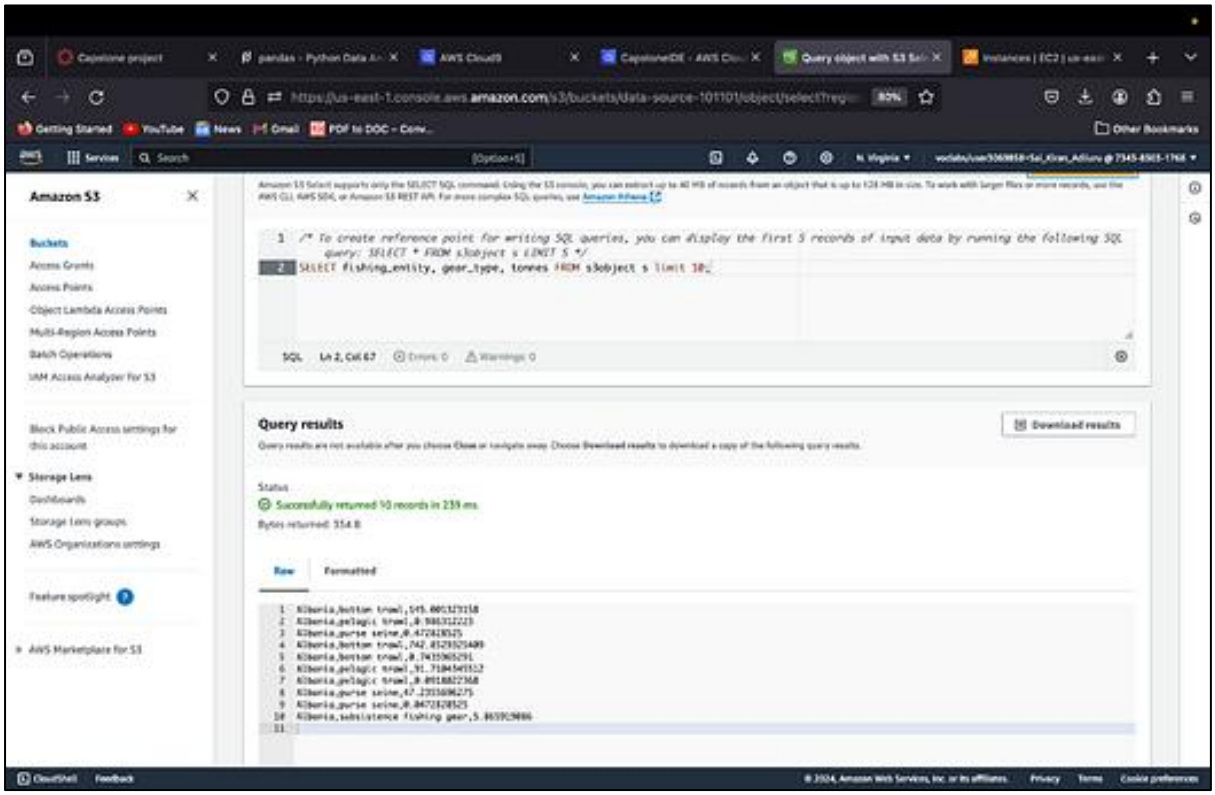


Queried a single file with S3 Select

used the S3 Select feature to run the following query on the single data source file.

The command used was -

SELECT fishing_entity, gear_type, tonnes FROM s3object s limit 10;

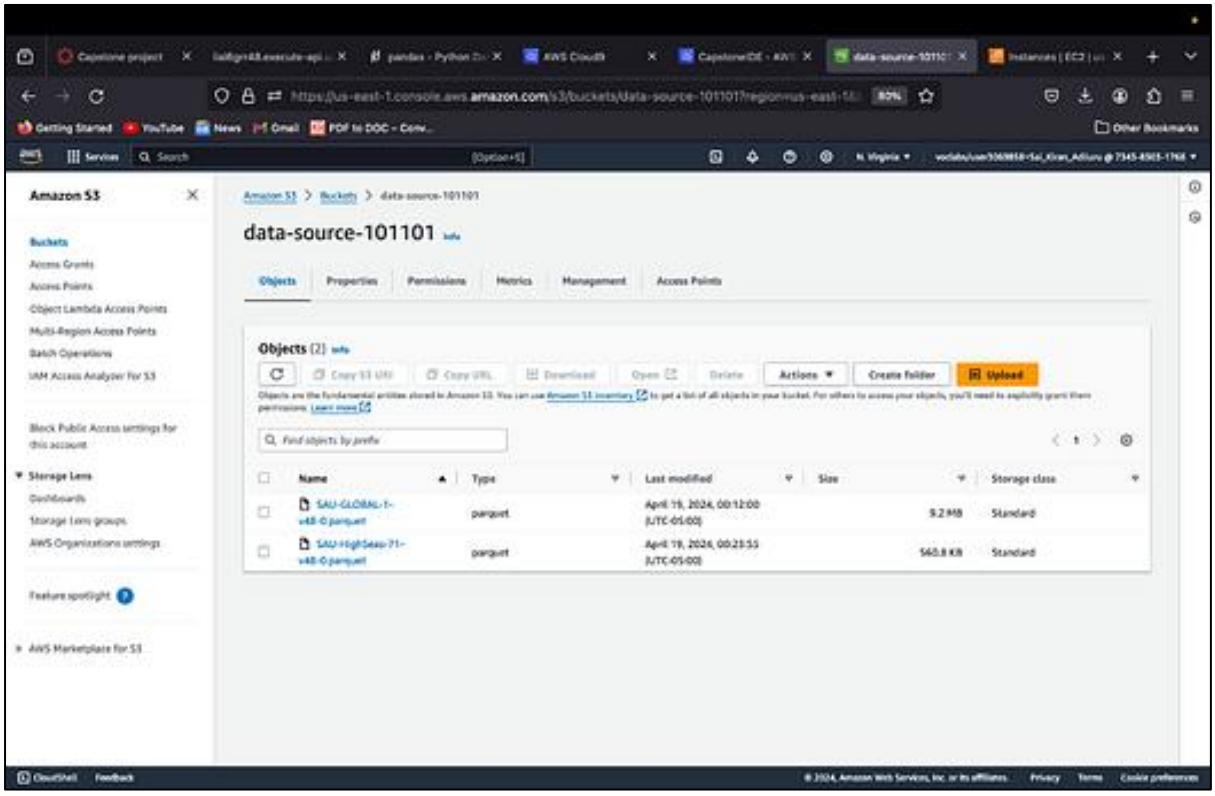


Used an AWS Glue crawler and queryed multiple files with Athena

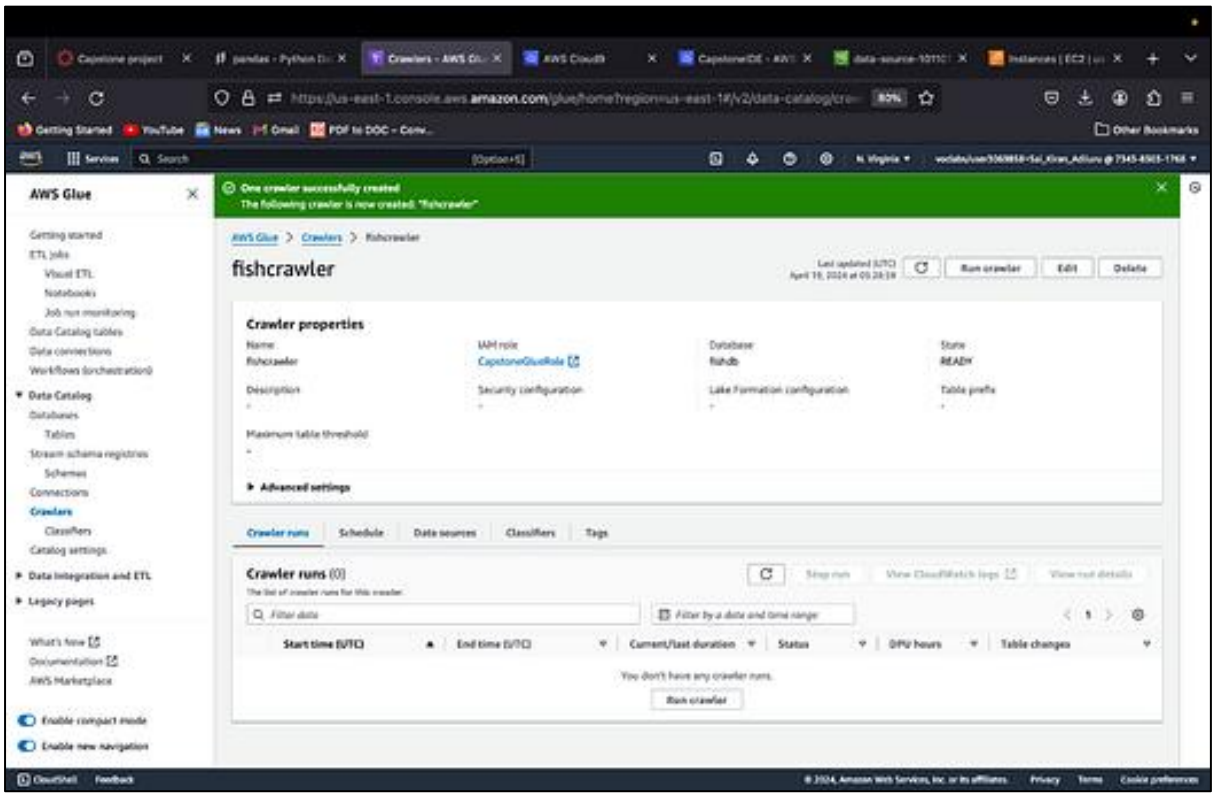
query that I ran in the previous task works well for a single data file. But if I needed to query a larger dataset that consists of more than one file. To do this I configured an AWS Glue crawler to discover the structure of the data and then used Athena to query the data.

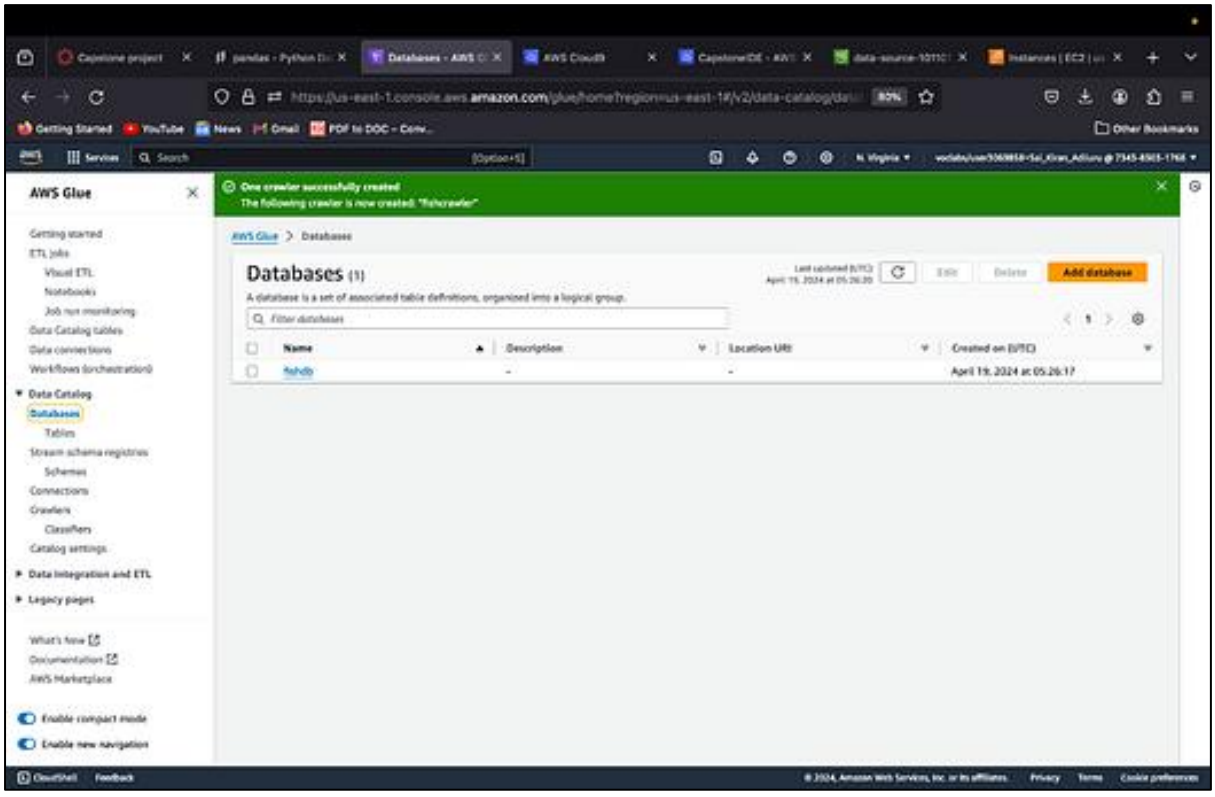
Because there is only one file in the bucket to upload the second file i used the following command

aws s3 cp *SAU-HighSeas-71-v48–0.parquet* s3://data-source-101101/

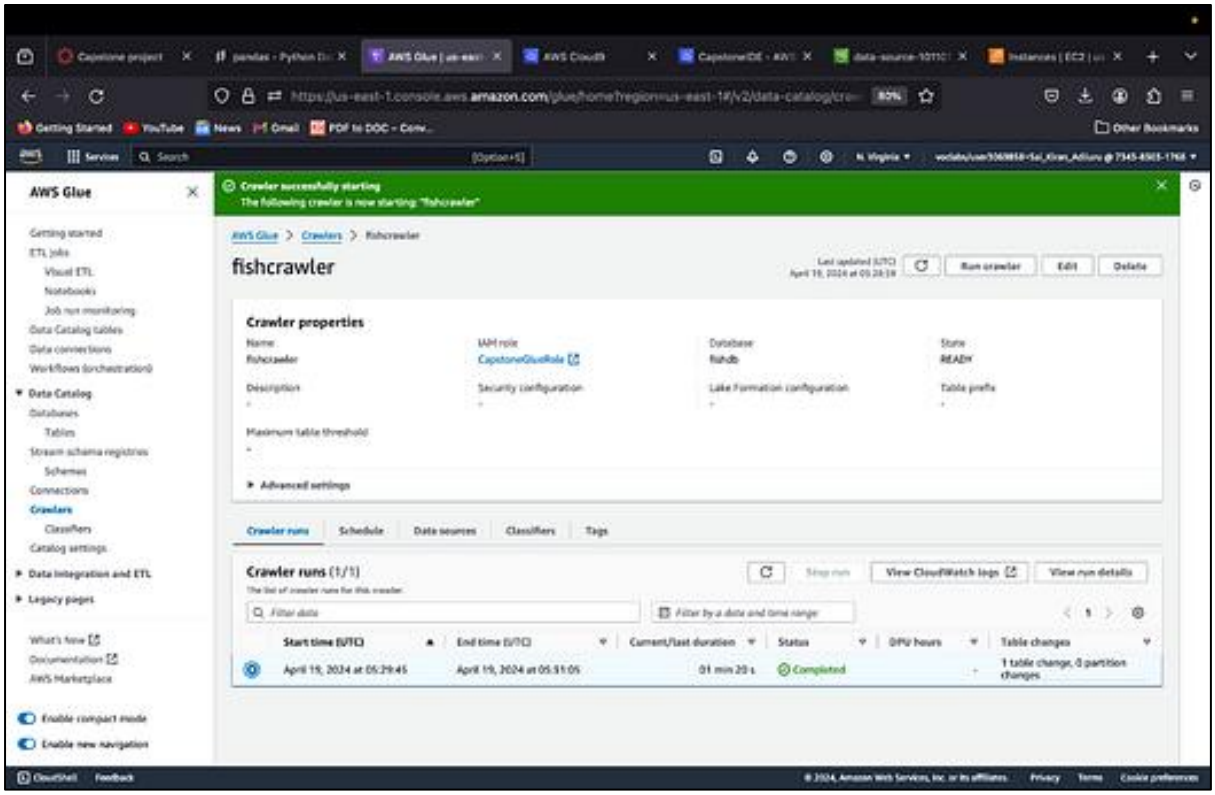


Next was creation of database and crawler with the IAM roles that was created in the start of the project.





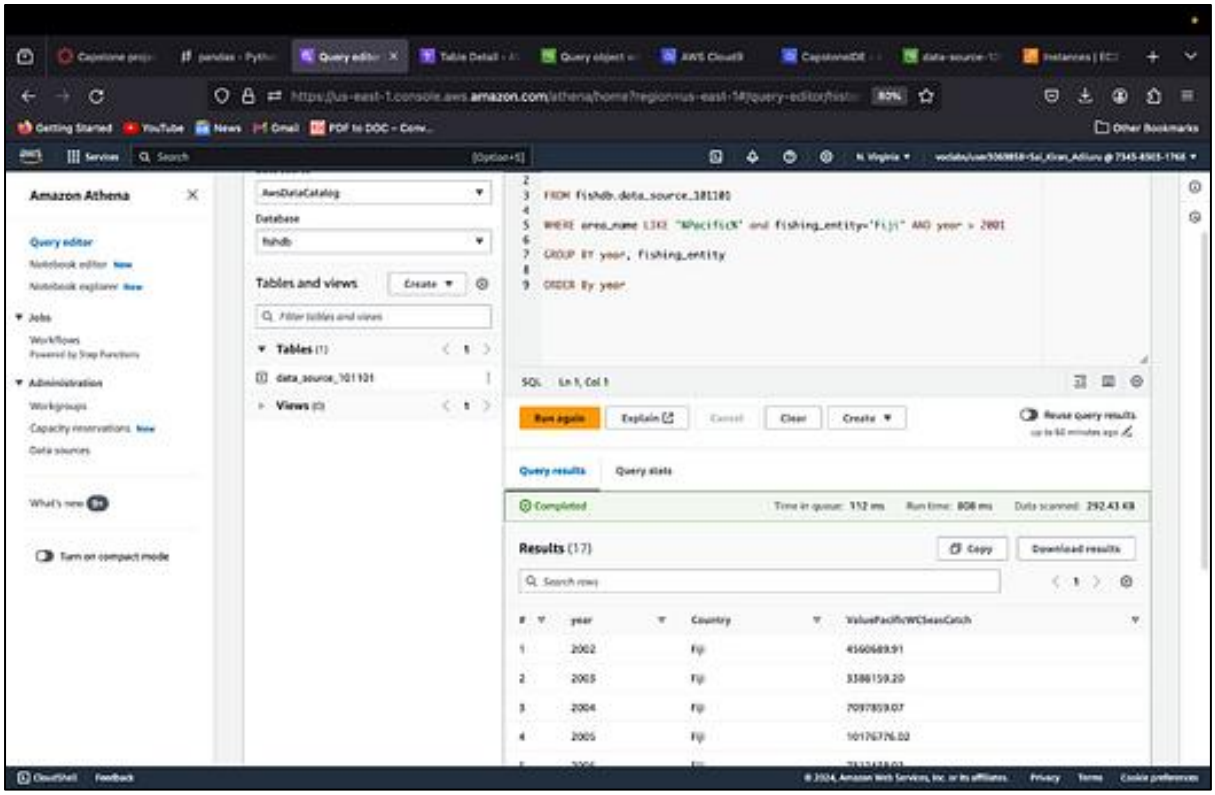
Ran the crawler to create a table that contains metadata in the AWS Glue database.



Configured the Athena Query Editor to output data to the *query-results* bucket.

Ran a command to find the value in US dollars of all fish caught by the country Fiji from the Pacific, Western Central high seas area since 2001, organized by year, used the following query

```
SELECT year, fishing_entity AS Country, CAST(CAST(SUM(landed_value) AS DOUBLE) AS DECIMAL(38,2)) AS ValuePacificWCSeasCatch

FROM fishdb.data_source_101101

WHERE area_name LIKE '%Pacific%' and fishing_entity='Fiji' AND year > 2001

GROUP BY year, fishing_entity

ORDER By year
```



Transformed the third file and adding it to the dataset

First created a backup file for the third file , the command was a follows

cp SAU-EEZ-242-v48-0.csv SAU-EEZ-242-v48-0-old.csv

The column names of other files didn't match with the third file , so the column names in file had to changed so the commands are as follows

Python

import pandas as pd

data_location = 'SAU-EEZ-242-v48-0-old.csv'

df = pd.read_csv(data_location)

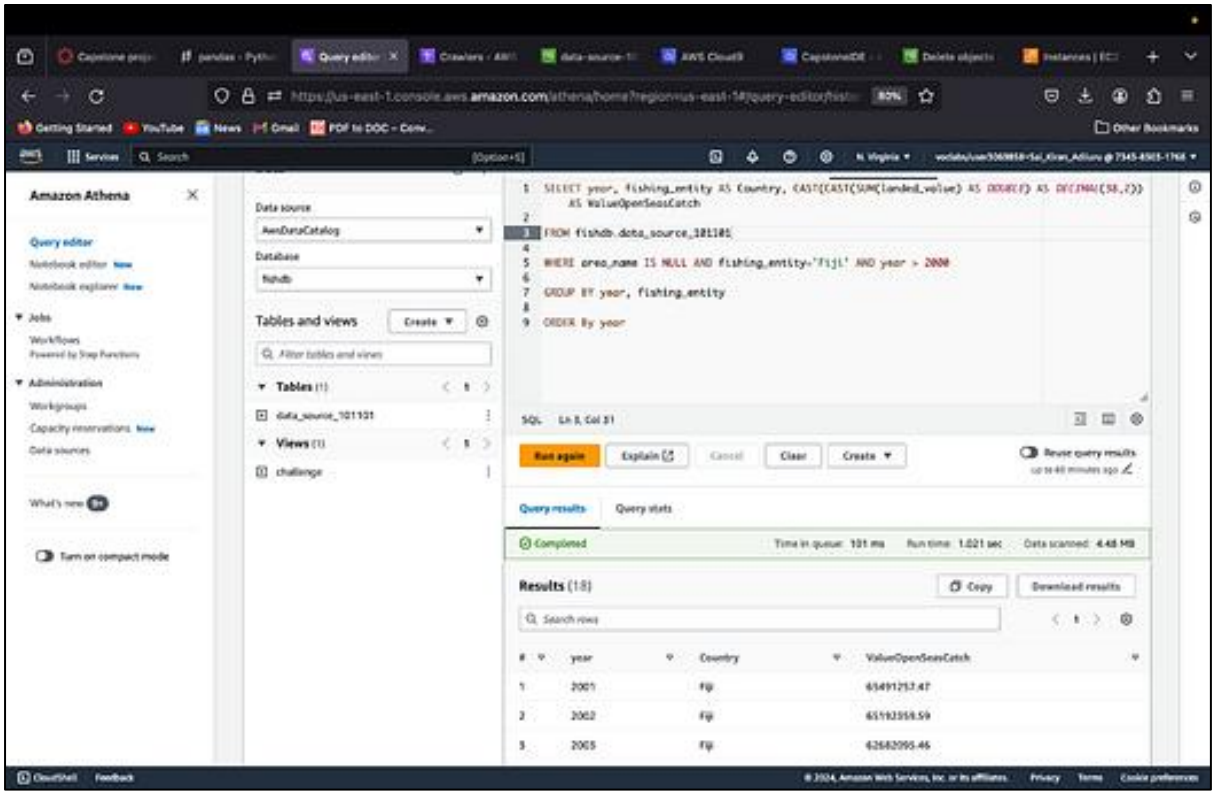
df.rename(columns = {"fish_name": "common_name", "country": "fishing_entity"}, inplace = True)

```
df.to_csv('SAU-EEZ-242-v48-0.csv', header=True, index=False)

df.to_parquet('SAU-EEZ-242-v48-0.parquet')

df.to_parquet('SAU-EEZ-242-v48-0.parquet')

exit()
```

Now uploaded the updated file to the S3 bucket as I did in the previous steps. Ran the glue crawler to update the table meta data.

To get the results and insights i used AWS Athena , I used to following commands for the results.

```
SELECT year, fishing_entity AS Country, CAST(CAST(SUM(landed_value) AS DOUBLE) AS DECIMAL(38,2)) AS ValueOpenSeasCatch

FROM fishdb.data_source_101101

WHERE area_name IS NULL AND fishing_entity='Fiji' AND year > 2000

GROUP BY year, fishing_entity

ORDER By year
```



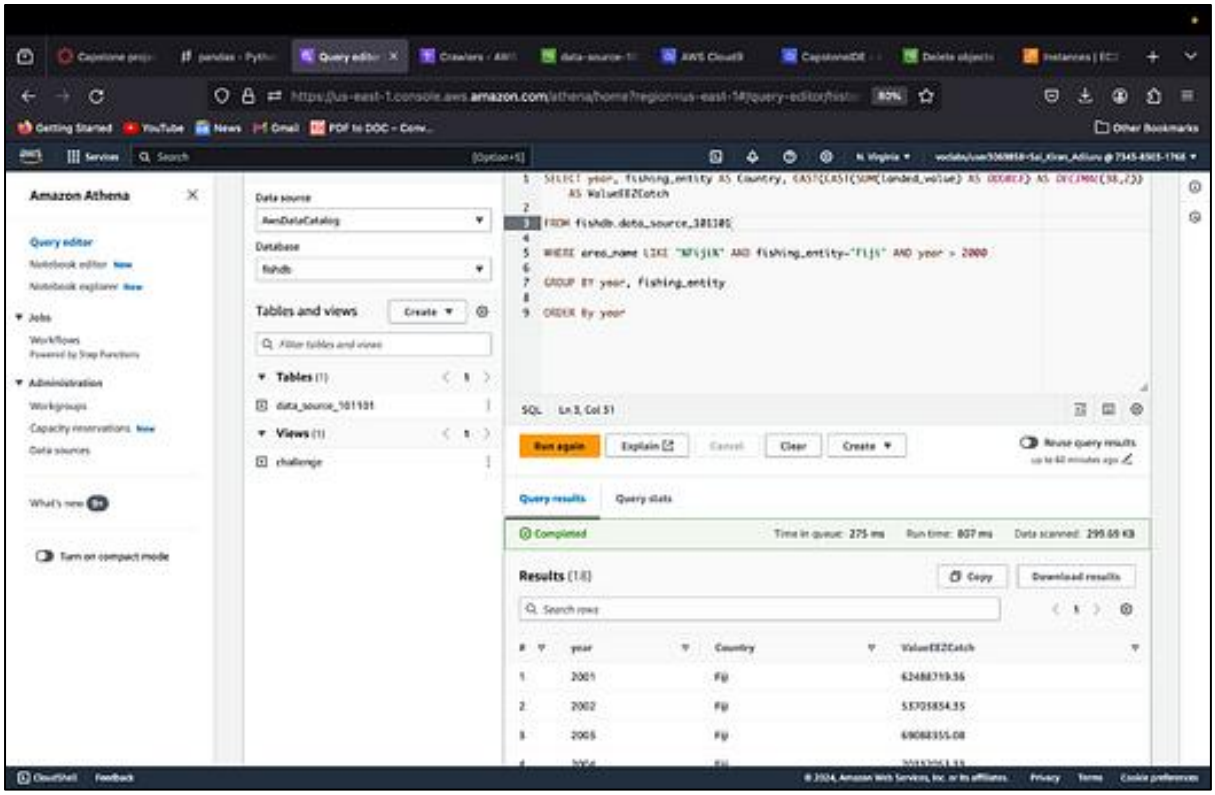
SELECT year, fishing_entity AS Country, CAST(CAST(SUM(landed_value) AS DOUBLE) AS DECIMAL(38,2)) AS ValueEEZCatch

FROM fishdb.data_source_101101

WHERE area_name LIKE '%Fiji%' AND fishing_entity='Fiji' AND year > 2000

GROUP BY year, fishing_entity

ORDER By year



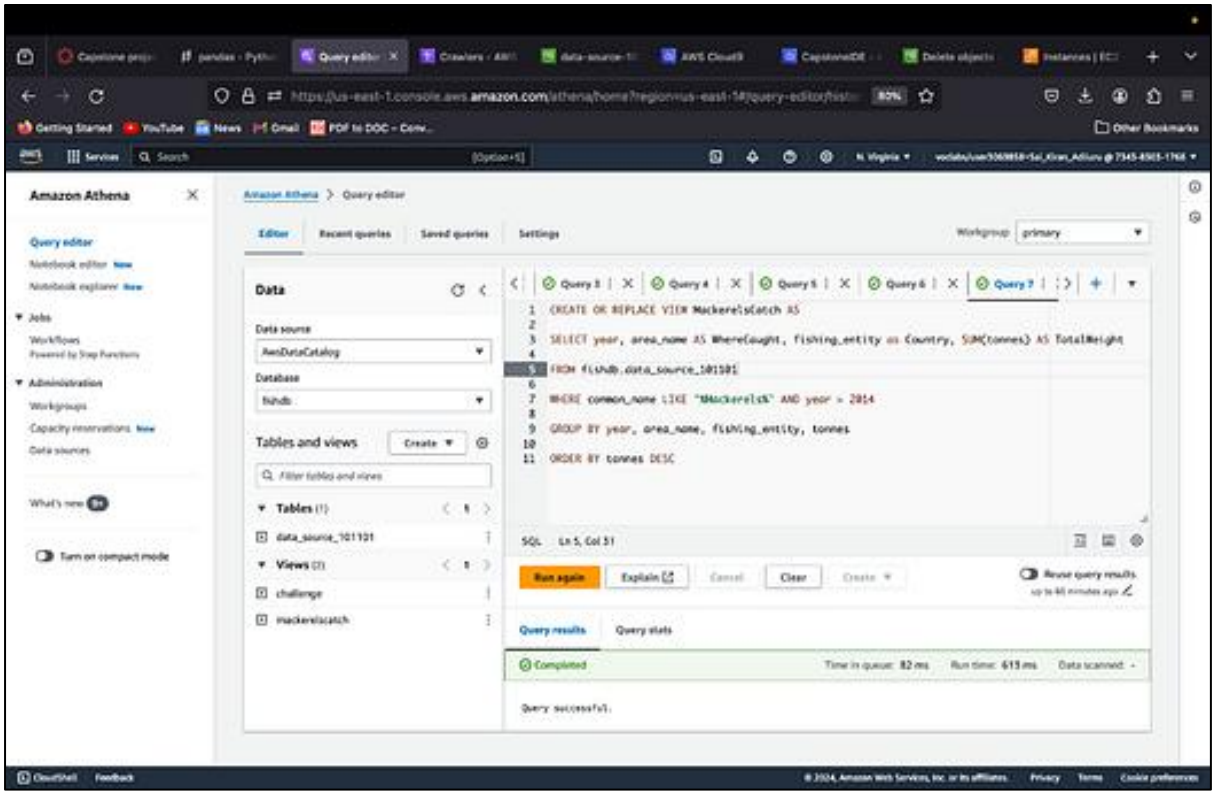
SELECT year, fishing_entity AS Country, CAST(CAST(SUM(landed_value) AS DOUBLE) AS DECIMAL(38,2)) AS ValueEEZAndOpenSeasCatch

FROM fishdb.data_source_101101

WHERE (area_name LIKE '%Fiji%' OR area_name IS NULL) AND fishing_entity='Fiji' AND year > 2000

GROUP BY year, fishing_entity

ORDER By year



The conclusion, data is formatted well and the AWS Glue crawler properly updated the metadata table, then the results that I got from queries in this steps should add up to the results that you get from the third query.