A PROJECT REPORT

***on***

**“**Data Migration and Transformation Tool for Amazon NO SQL Data Bases**”**

***Submitted to***

GUVI GEEK NETWORK

In partial fulfilment of the requirements for the award CCM

**IN**

**MASTER DATA ENGINEERING COURSE**

***By***

**PALUKURI SRINIVAS**

**E5 – Data Engineering batch**

*Under the esteemed guidance of*

**DE mentors,**

**“GUVI”**

****

**GIVEN PROJECT STATEMENT**

**Project 1:**

|  |  |
| --- | --- |
| **Project Title** | **Data Migration and Transformation Tool for Amazon NO SQL Data Bases** |
| **Technologies** | **Python/PySpark,Requests,Zipfile,boto3,pandas,sqlalchemy, Amazon S3,Amazon NO SQL Data Bases** |

**Problem Statement:**

You have a [URL](https://www.sec.gov/edgar/sec-api-documentation) that points to a zip file. The zip file contains multiple JSON files. The JSON files contain multiple documents with various data structures. Your goal is to download the zip file from the URL, extract the data from the JSON files, store it in Amazon S3, and load it into Amazon RDS. You want to use Python or PySpark to perform these tasks. You may use any libraries or tools that are necessary to complete the task.

**Approach:**

To extract the data from a zip file that is available at a URL and load it into Amazon S3 and Amazon RDS (NoSQL), you can follow these steps:

1. Use the requests library to download the zip file from the URL.
2. Use the zipfile module to extract the data from the zip file.
3. Use the boto3 library or PySpark to store the data in Amazon S3.
4. Use the pandas library and sqlalchemy or PySpark to load the data from S3 into Amazon RDS (NoSQL).

**Results:**

The result of following these steps should be that the data from the zip file is extracted and stored in a list of dictionaries (if you are using Python) or a DataFrame (if you are using PySpark). Each dictionary or DataFrame row will represent a document from one of the JSON files in the zip file.

The data in the list or DataFrame will then be stored in Amazon S3 as JSON files. You will be able to access these JSON files using the boto3 library or the Amazon S3 web interface.

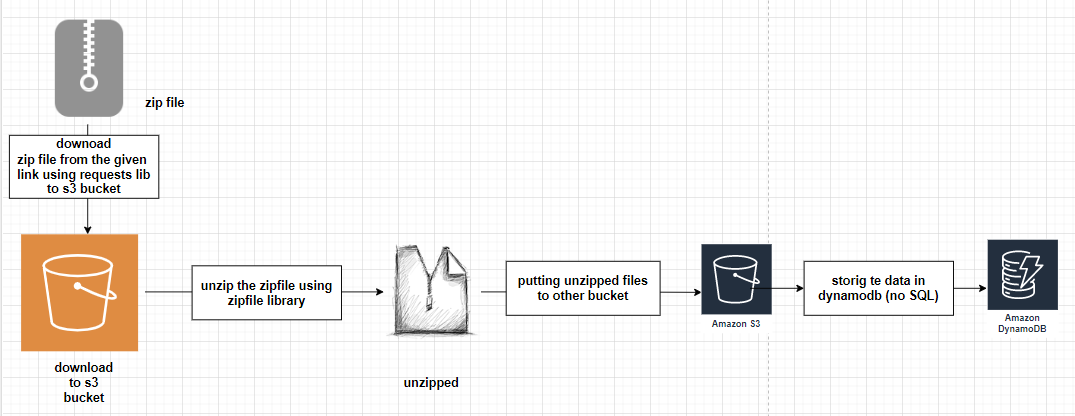
The data from the JSON files will also be loaded into Amazon RDS (NoSQL). You will be able to access the data in RDS using SQL queries. The data will be stored in a table in RDS, and the schema of the table will be determined by the structure of the JSON documents.

I hope this helps. Let me know if you have any further questions or need more assistance.

Project description:

Data Migration and Transformation Tool for Amazon NOSQL Data Storages.

Project explained in below diagram:



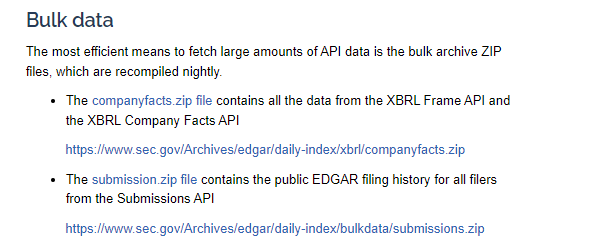
Given link for downloading the zip file : <https://www.sec.gov/edgar/sec-api-documentation>

By following this link we can get this page.



After reaching this page we can see the option two options for downloading the zip files as shown in the below picture we can download any one for completing this project.

I have selected the submissions.zip because it contains the each organization fillings submissions.

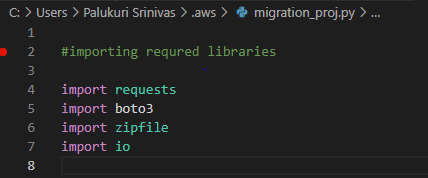


Selected link for downloading: <https://www.sec.gov/Archives/edgar/daily-index/bulkdata/submissions.zip>

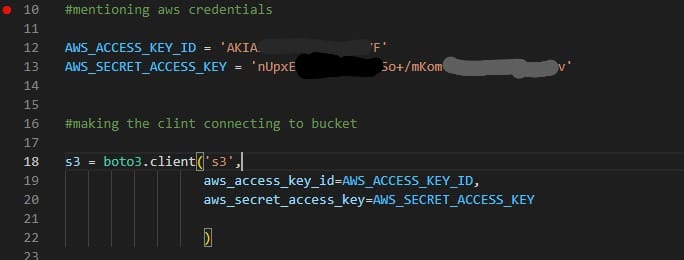
So now first part of the project: downloading the ZIPFILE through given link using requests library.

* Importing the required libraries

1. requests: for downloading the zip file
2. boto3: a python SDK for AWS
3. zipfile: for unzipping the zipped file
4. io : to do input and output operations

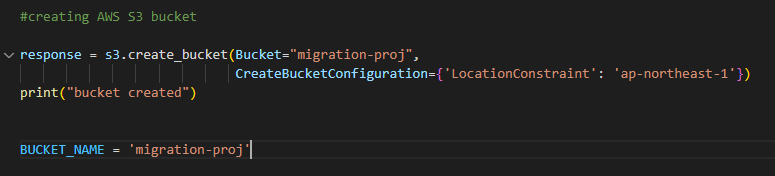


Motioning AWS credentials to connect the AWS resources to python ide (VS code)



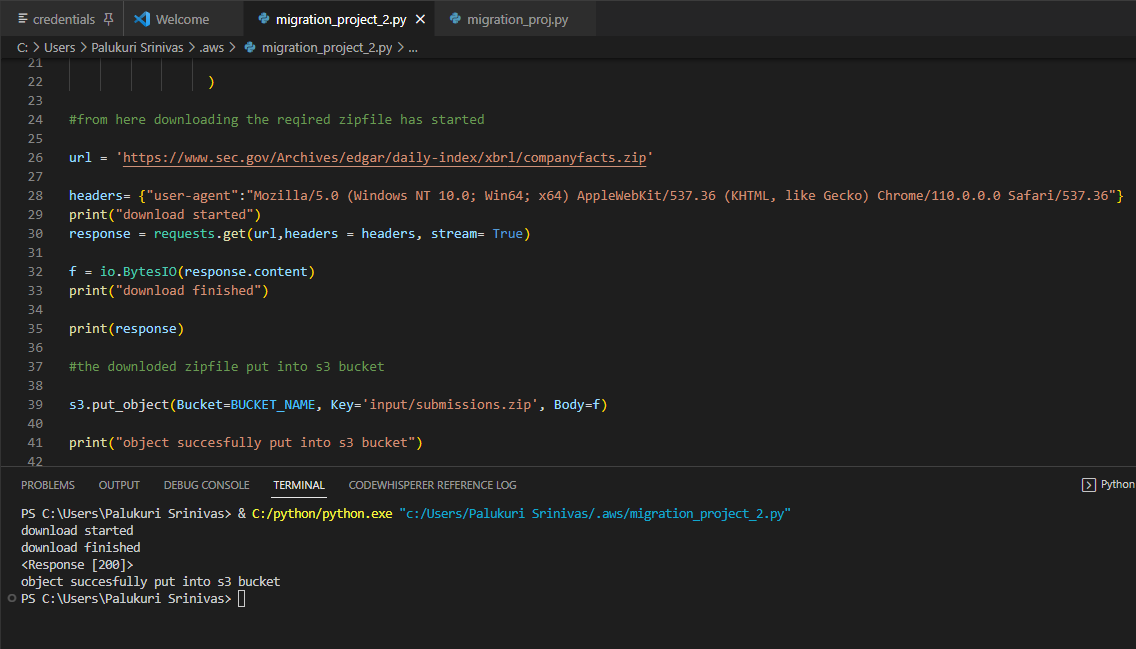
Observe the above picture we have given the we have mentioned the **AWS\_ACCESS\_KEY\_ID** and **AWS\_SECRET\_ACCESS\_KEY** to connect the AWS resources.

* Created the AWS S3 bucket for downloading the zip file.

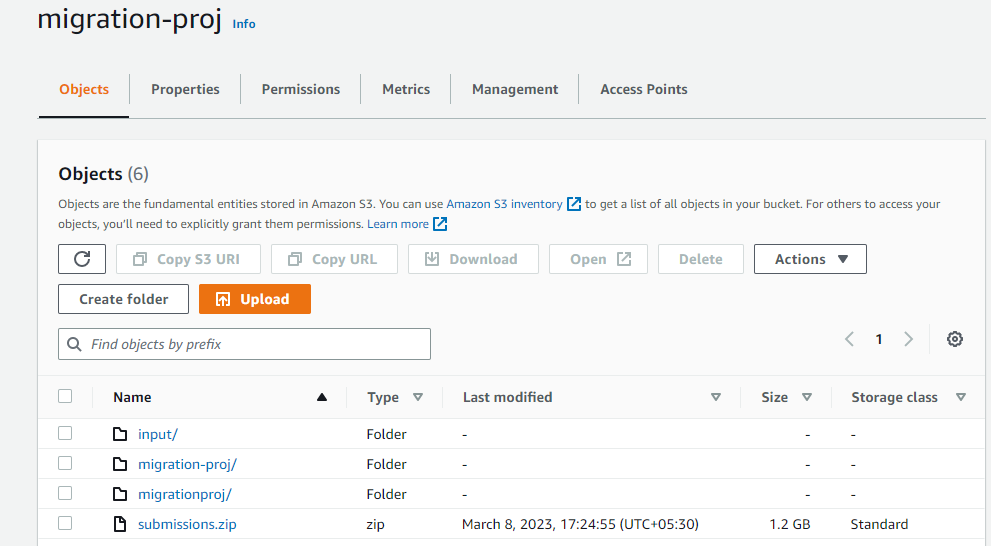


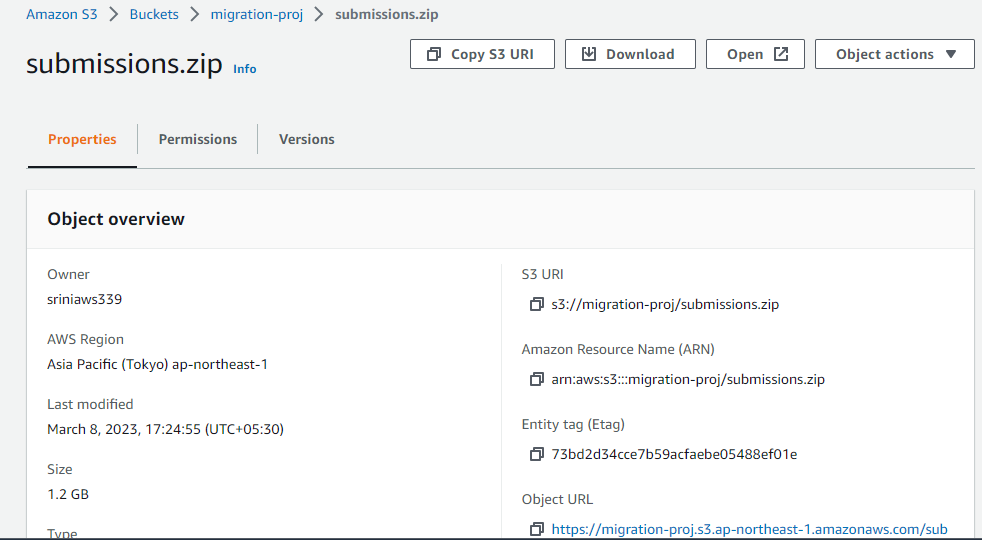
Note: bucket name must be unique in your region so select the unique name to overcome the invalid location constraint error while executing.

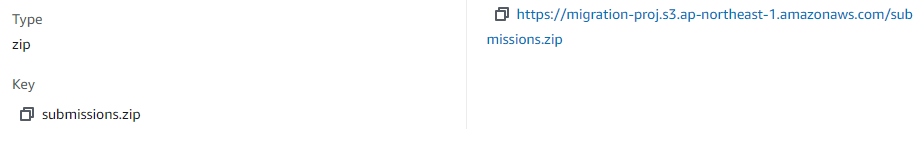
* Downloading the zip file into AWS S3 bucket using requests library and using boto3 putting the downloaded file in to S3 bucket.



The mentioned zip file successfully downloaded and put into mentioned S3 bucket: result picture attached below

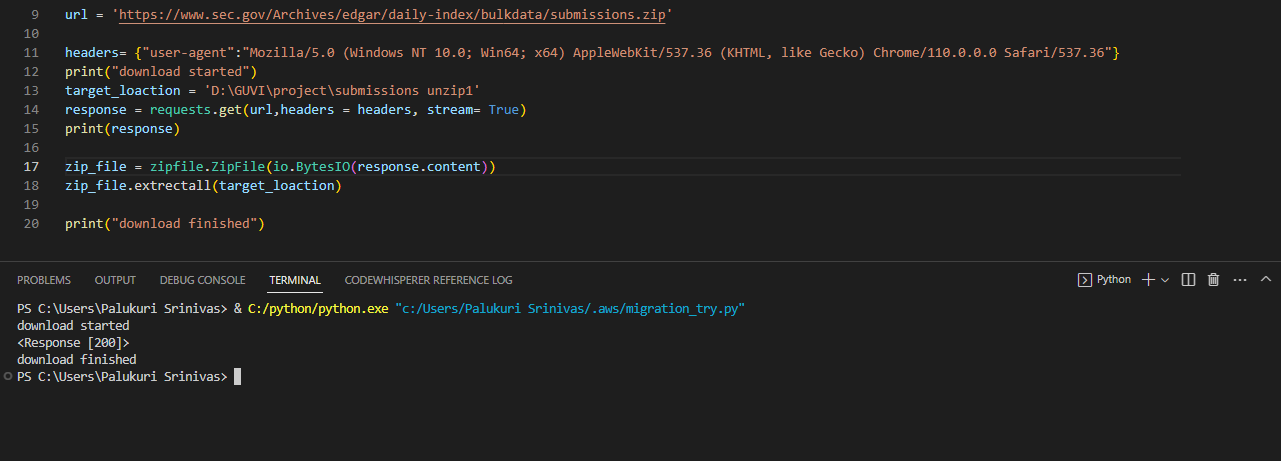




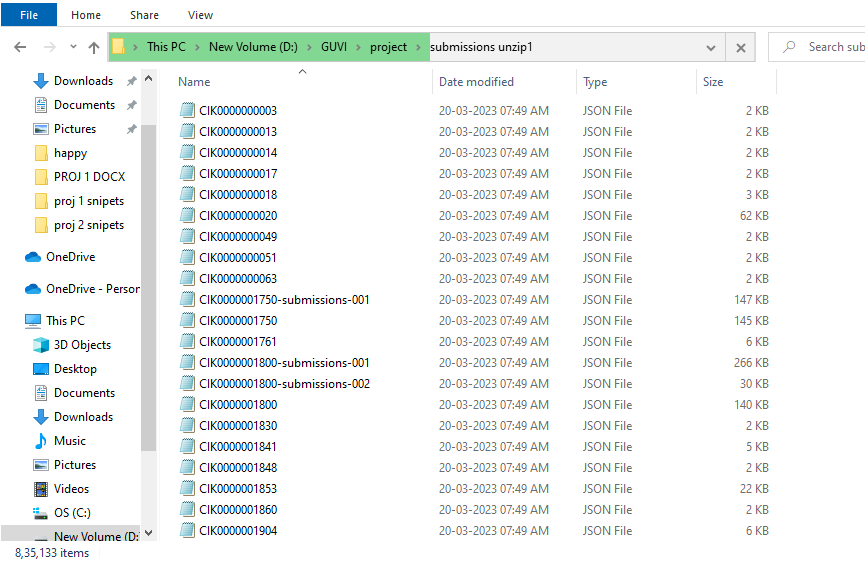


Unzipped the files using the zipfile module:

Tried to unzip the file using the zip file it took more than 8hrs of time but task has not completed so then I wanted to check what present in that zip file so downloaded to local storage and extracted using extractall



Files downloaded and extracted to local storage.

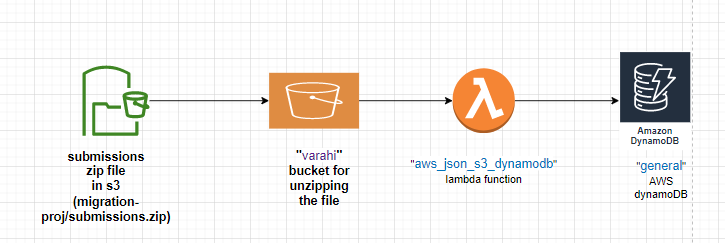


After downloading the file to my local storage then I can know that it containing 835133 (8lakhs of json files)

NOTE: Here we have an issue for unzipping, the zip file contains more than 8 lakh of files, its taking more time to unzip and im using AWS free tier account then I have asked the DE MENTOR during Project doubt sessions as per his suggestion to unzip 100 no of json files are enough to complete the project. But we should have the basic idea to unzip all the files in the zipfile. As per mentor suggestion completed the project by unzipping 150 no of json files.

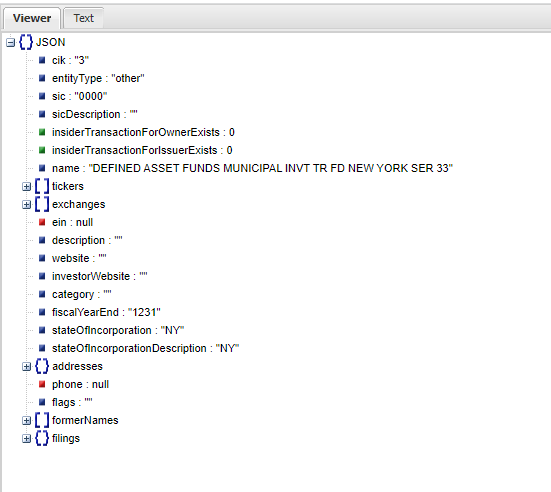
Our strategy to store the json files to dynamoDB from the zip file:

Creating the AWS lambda function which is triggered by S3 “varahi” bucket, then the lambda function will post the data to Dynamo DB “general” as a record.



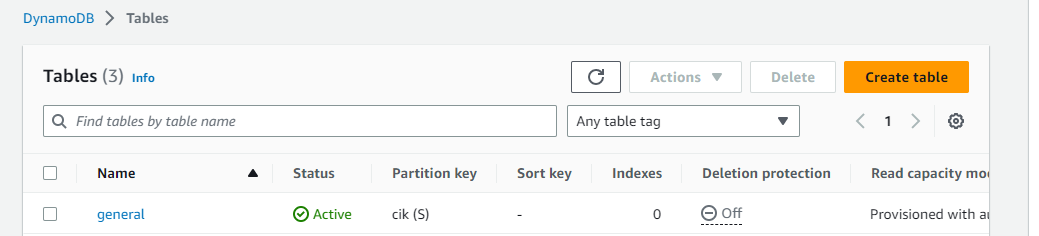
I have checked the content present in the json in json viewer.

I got the result as shown in below picture and verified many json files then I can find the partition key to create the dynamoDB is “cik” data type is “string” I have observed the many files similar data type and partition key found .

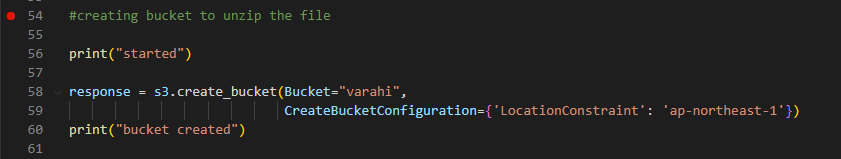


This is that json files containing.

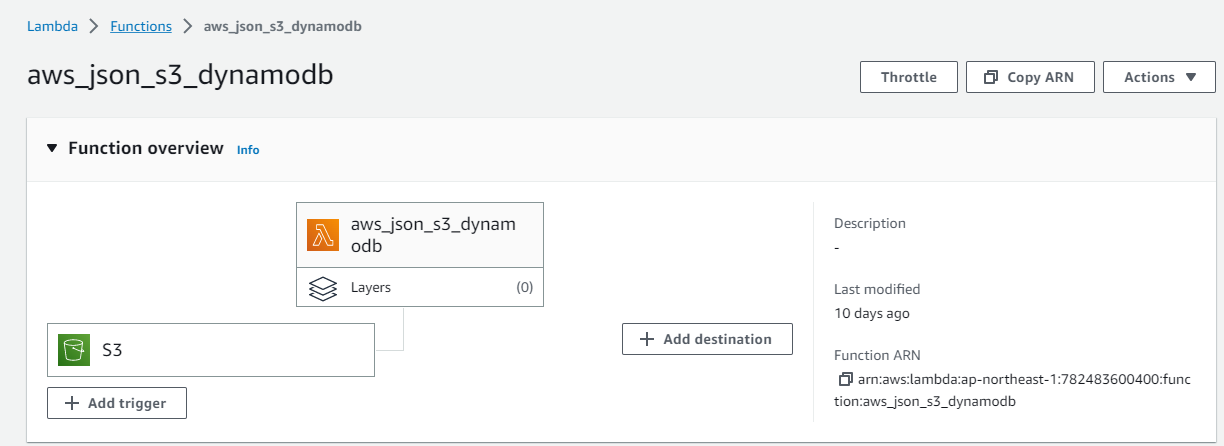
* Creation of dynamo DB for saving the Json data which is present in unzipped files.
* DynamoDB name : general created with partition key as “cik” in “string” data type.

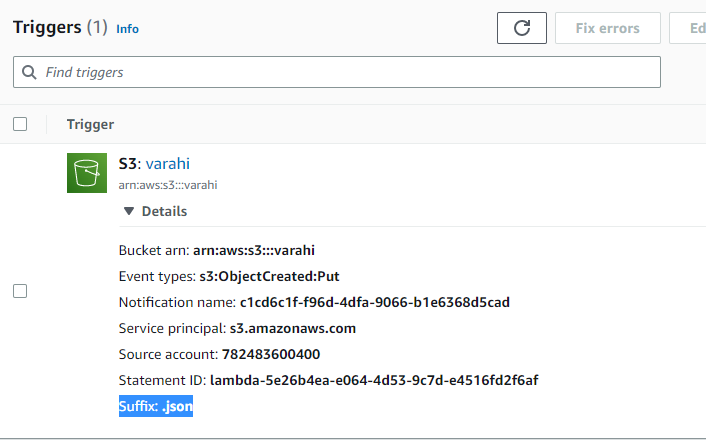


* Creation of S3 bucket for unzipping, s3 bucket name: “varahi”

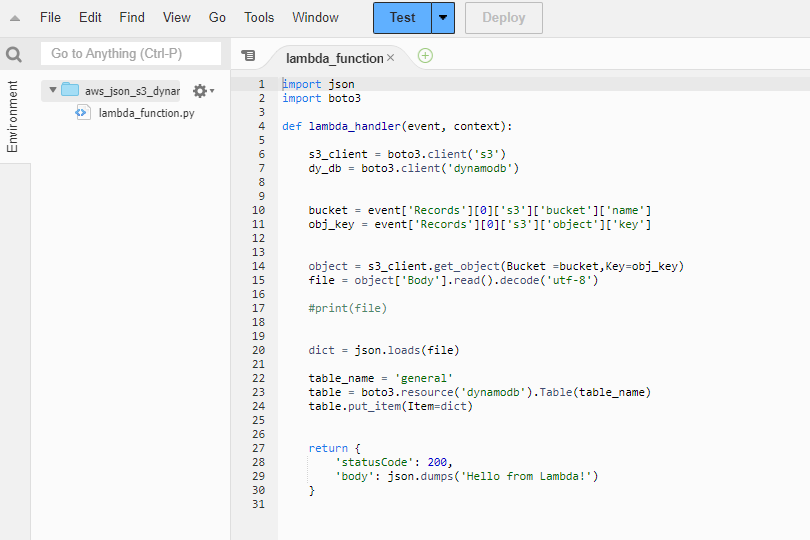


* Creation of AWS lambda function with name “aws\_json\_s3\_dynamodb”



Added the trigger to lambda function as s3 bucket “varahi” , and Suffix: .json : so that any json file added to “varahi” bucket which have partition key as “cik” in “string” data type can be added to “general” dynamoDB table as an row.

Used lambda code given below:



In the above code we can see :

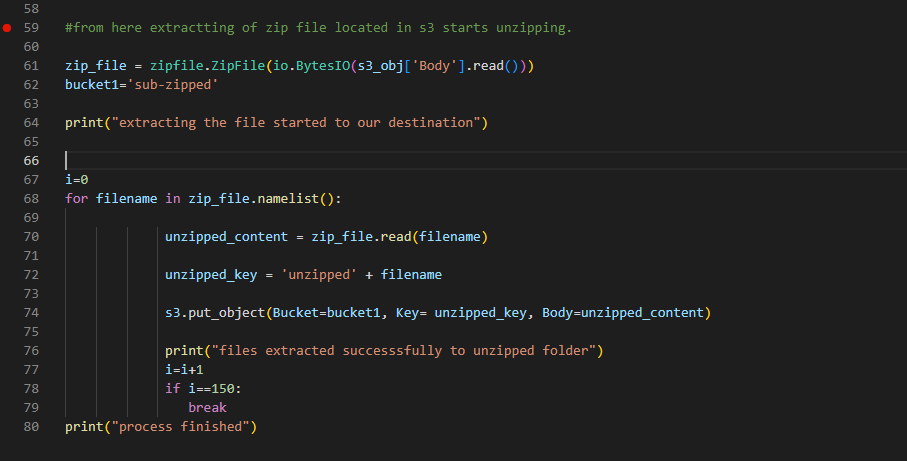
1. Imported the required libraries.
2. Created S3 client.
3. Created the DynamoDB client.
4. Getting the bucket name and object key by pain event .
5. Retrieving object with bucket name and object key.
6. Reading the content to file variable.
7. Converting the file variable from json to dictionary (as per project statement).
8. Inserting the data to dynamo Db using Put item method.

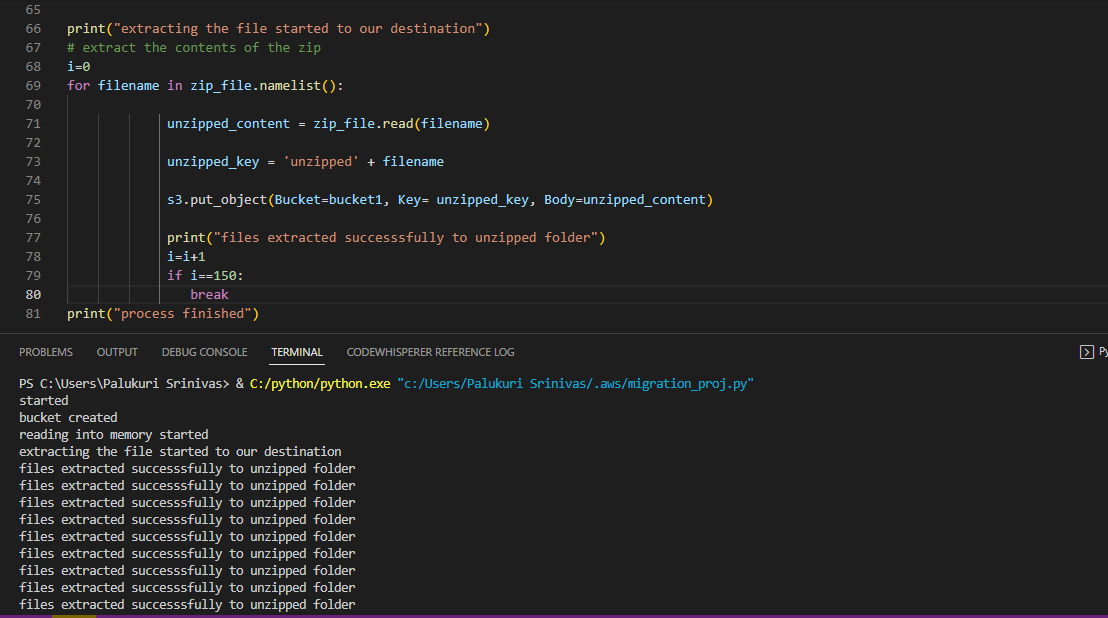
So to complete our process

1. DynamoDB table created
2. S3 bucket created for unzipping and triggering the lambda function
3. Lambda function also created with required code

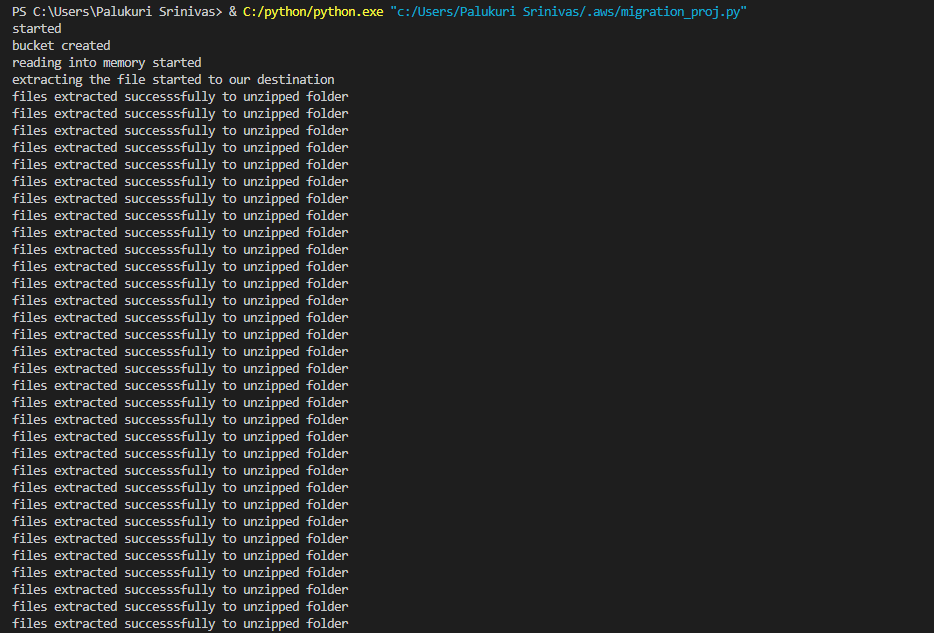
Now when any file added to “varahi” we know any Json file have partition key a “cik” with “string”data type it will be added to dynamo db as new record.

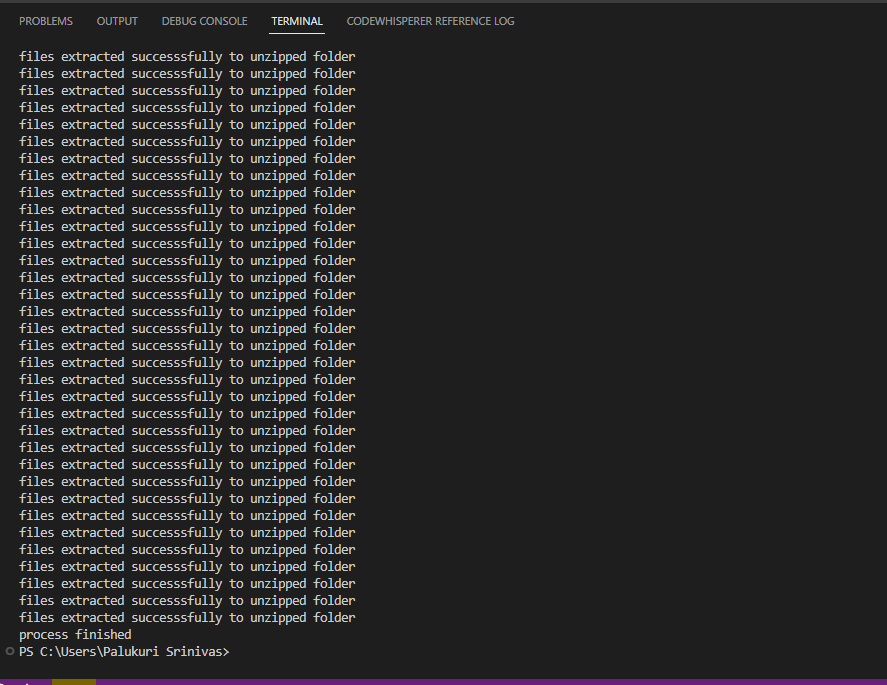
* Unzipping the “Submissions.zip” file to “varahi” setting limit as 150 nos, code snippet attached below



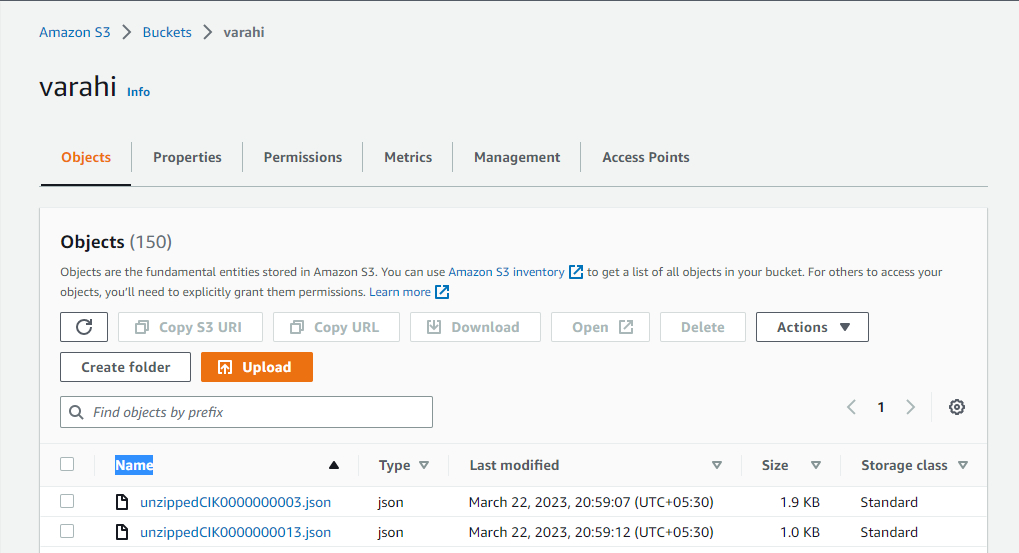


Results attached below for executing the above code:

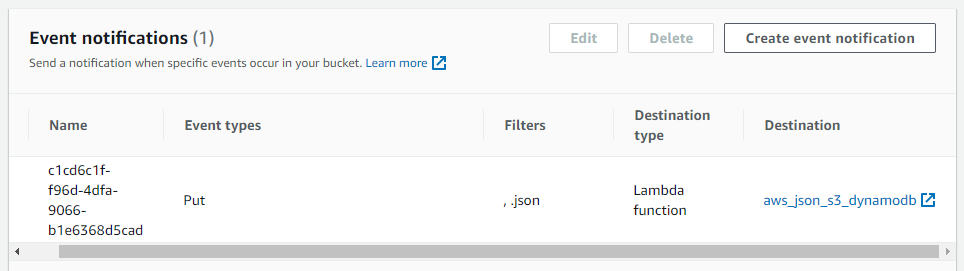




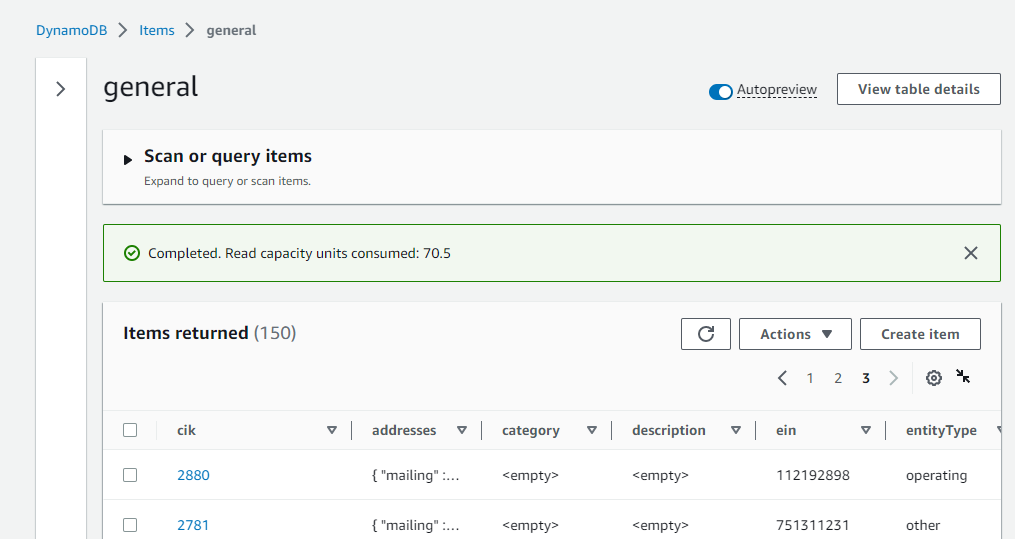
Unzipped files to varahi bucket :



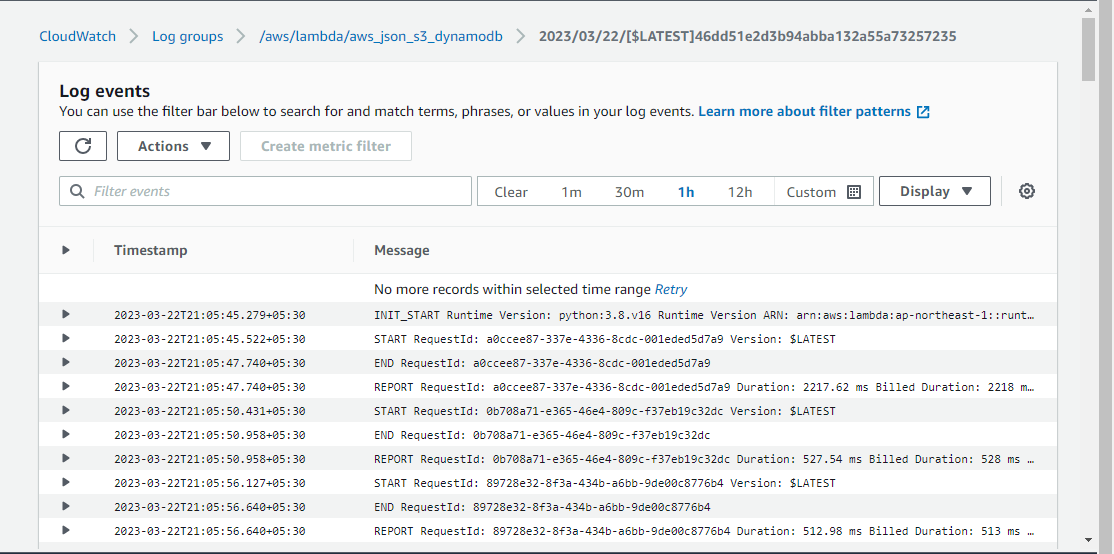
We can see the destination as lambda function “aws\_json\_s3\_dynamodb”

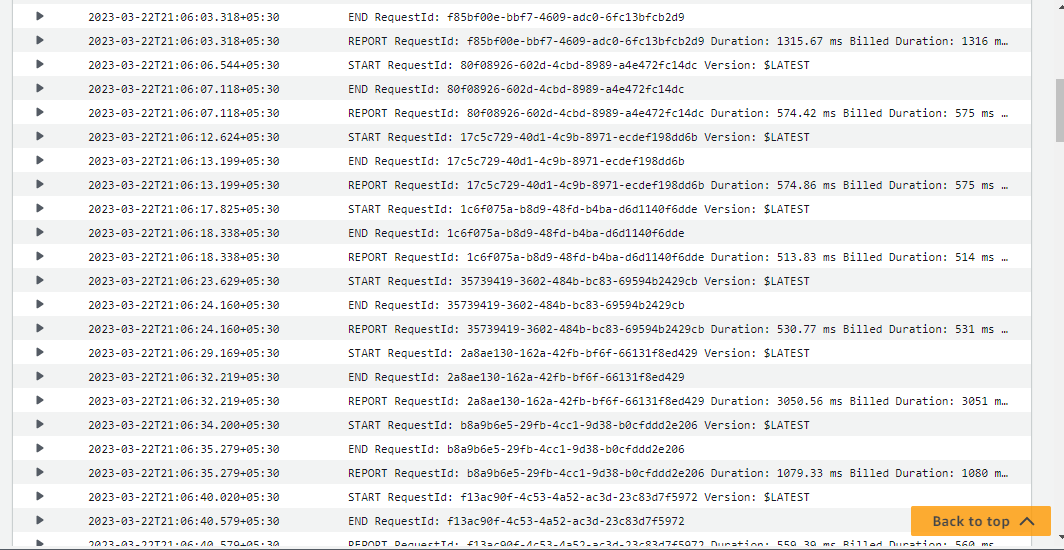


Finally as a results the json files updated to dynamoDB as a records (Rows) snippets attached below. We can find the 150 rows updated to “general” dynamoDB.



Cloud watch logs samples attached below as a snippet:





Note: in the above unzipping code we have make limit for 150nos if we remove that each file will be unzipped and uploaded to dynamoDB records.

CONCLUSION:

* As per project statement we have downloaded the zip file using.
* Zip file is unzipped using zip file python module.
* Using boto3 module uploaded to S3 bucket.
* The json files converted to dictionary and updated to dynamoDB using boto3 SDK

Task completed.

