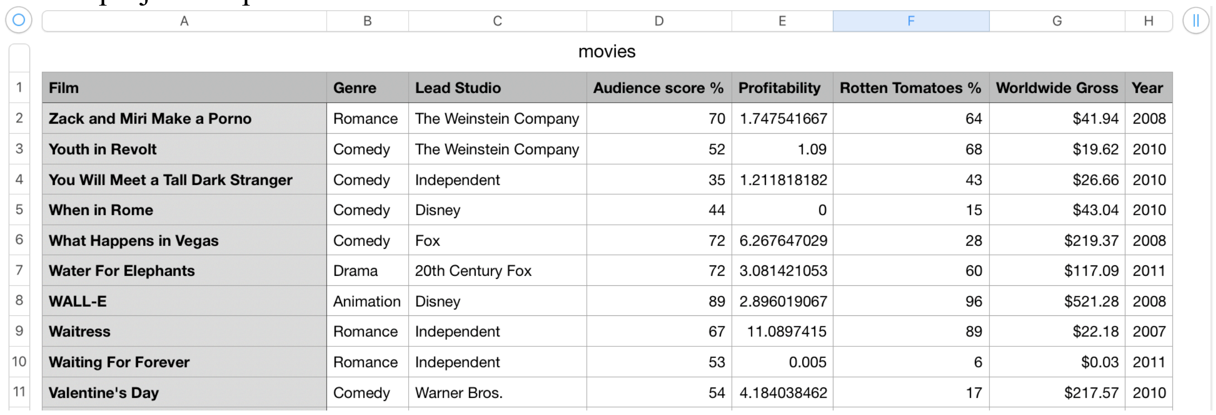
Using AWS Cloud Computing Services
Sonam Dawani

This project is to present the use of Amazon Web Services. The following Amazon Web Services are used in this project:

- S3
- AWS Glue
  - Crawler
  - ETL Job
- AWS Lambda
- AWS CloudWatch
- AWS Athena
- AWS SageMaker

The code and scripts implemented in this project are uploaded on Github.
https://github.com/SONAMDAWANI/UsingAWS_DataAnalysis

For this project simple data is used: movies.csv

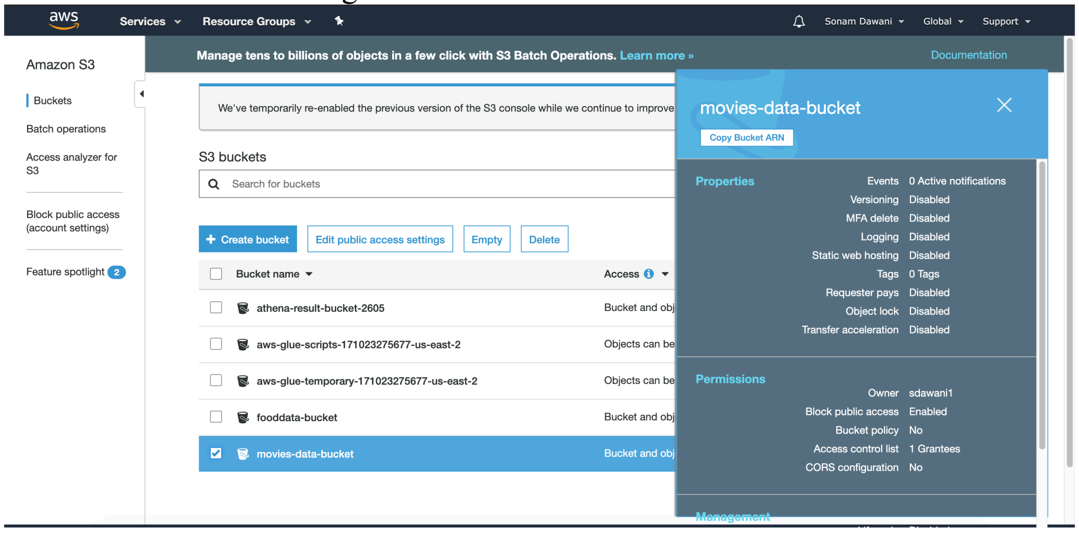


movies

	A	B	C	D	E	F	G	H
1	**Film**	**Genre**	**Lead Studio**	**Audience score %**	**Profitability**	**Rotten Tomatoes %**	**Worldwide Gross**	**Year**
2	**Zack and Miri Make a Porno**	Romance	The Weinstein Company	70	1.747541667	64	$41.94	2008
3	**Youth in Revolt**	Comedy	The Weinstein Company	52	1.09	68	$19.62	2010
4	**You Will Meet a Tall Dark Stranger**	Comedy	Independent	35	1.211818182	43	$26.66	2010
5	**When in Rome**	Comedy	Disney	44	0	15	$43.04	2010
6	**What Happens in Vegas**	Comedy	Fox	72	6.267647029	28	$219.37	2008
7	**Water For Elephants**	Drama	20th Century Fox	72	3.081421053	60	$117.09	2011
8	**WALL-E**	Animation	Disney	89	2.896019067	96	$521.28	2008
9	**Waitress**	Romance	Independent	67	11.0897415	89	$22.18	2007
10	**Waiting For Forever**	Romance	Independent	53	0.005	6	$0.03	2011
11	**Valentine's Day**	Comedy	Warner Bros.	54	4.184038462	17	$217.57	2010

Using S3 Buckets

Created bucket in S3 storage 'movies-data-bucket'



Manage tens to billions of objects in a few click with S3 Batch Operations. Learn more »

We've temporarily re-enabled the previous version of the S3 console while we continue to improve

S3 buckets

Bucket name	Access
athena-result-bucket-2605	Bucket and obj
aws-glue-scripts-171023275677-us-east-2	Objects can be
aws-glue-temporary-171023275677-us-east-2	Objects can be
fooddata-bucket	Bucket and obj
movies-data-bucket	Bucket and obj

movies-data-bucket

Properties

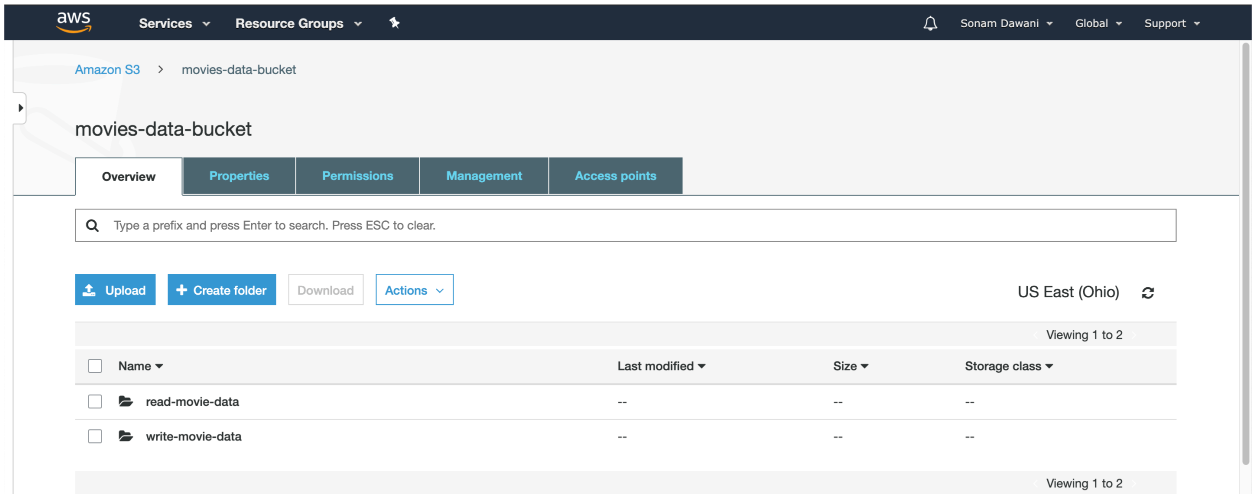
Events	0 Active notifications
Versioning	Disabled
MFA delete	Disabled
Logging	Disabled
Static web hosting	Disabled
Tags	0 Tags
Requester pays	Disabled
Object lock	Disabled
Transfer acceleration	Disabled

Permissions

Owner	sdawani1
Block public access	Enabled
Bucket policy	No
Access control list	1 Grantees
CORS configuration	No

Management

Created read and write folders in the 'movies-data-bucket' bucket.



movies-data-bucket

Viewing 1 to 2

Name	Last modified	Size	Storage class
read-movie-data	--	--	--
write-movie-data	--	--	--

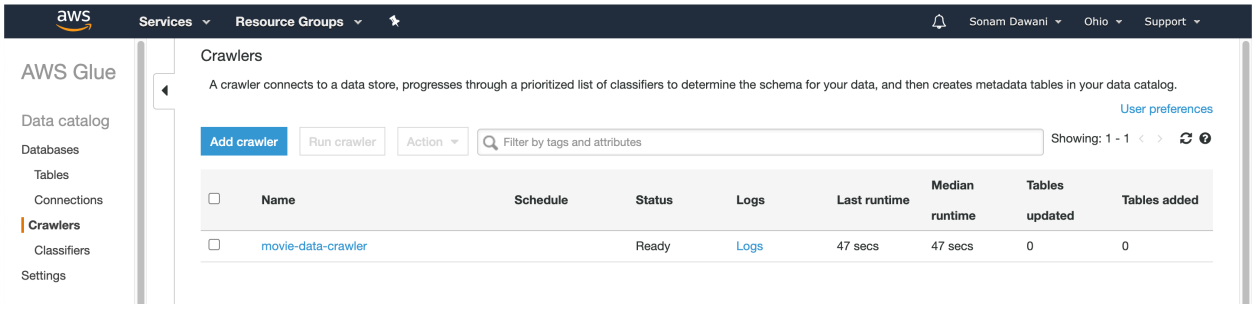
Viewing 1 to 2

The read folder contains the original csv file, whereas write data is initially empty. The result after ETL transformation will be stored in the write folder.

Using AWS Glue

- ### Crawler

First created a crawler which connects to the data storage (here the S3 bucket) and then creates metadata tables.

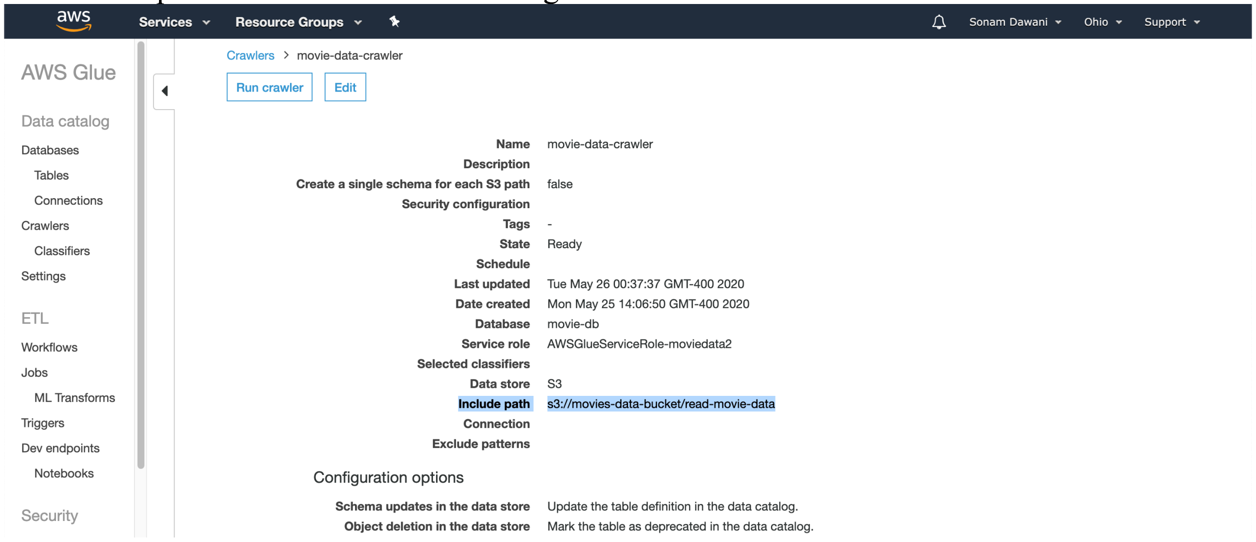


Crawlers

A crawler connects to a data store, progresses through a prioritized list of classifiers to determine the schema for your data, and then creates metadata tables in your data catalog.

Name	Schedule	Status	Logs	Last runtime	Median runtime	Tables updated	Tables added
movie-data-crawler		Ready	Logs	47 secs	47 secs	0	0

With the required IAM role and data storage information the crawler is created.

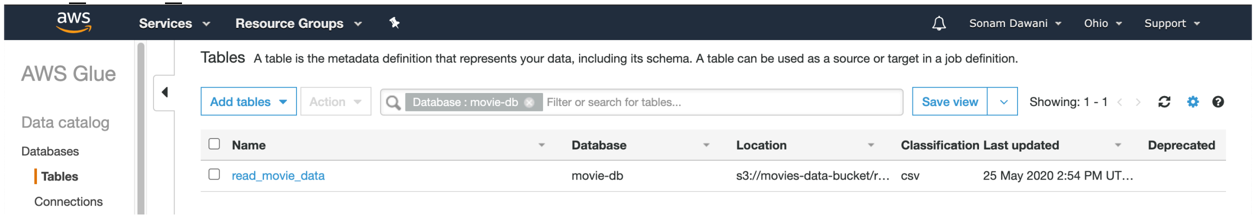


Name	movie-data-crawler
Description	
Create a single schema for each S3 path	false
Security configuration	
Tags	-
State	Ready
Schedule	
Last updated	Tue May 26 00:37:37 GMT-400 2020
Date created	Mon May 25 14:06:50 GMT-400 2020
Database	movie-db
Service role	AWSGlueServiceRole-moviedata2
Selected classifiers	
Data store	S3
Include path	s3://movies-data-bucket/read-movie-data
Connection	
Exclude patterns	

Configuration options

Schema updates in the data store	Update the table definition in the data catalog.
Object deletion in the data store	Mark the table as deprecated in the data catalog.

The execution of crawler above created the specified database ‘movies-db’ with ‘read_movie_data’ table in it.

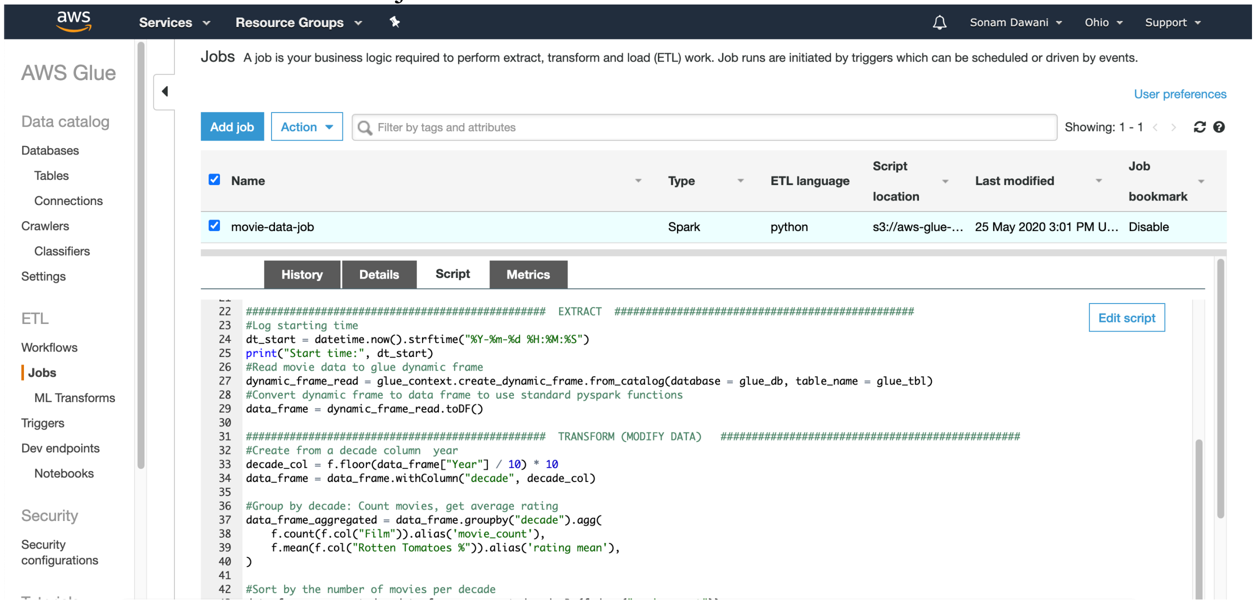


Tables A table is the metadata definition that represents your data, including its schema. A table can be used as a source or target in a job definition.

Name	Database	Location	Classification	Last updated	Deprecated
read_movie_data	movie-db	s3://movies-data-bucket/r…	csv	25 May 2020 2:54 PM UT…	

- **<u>ETL Job</u>**

Created run On-Demand ETL job.



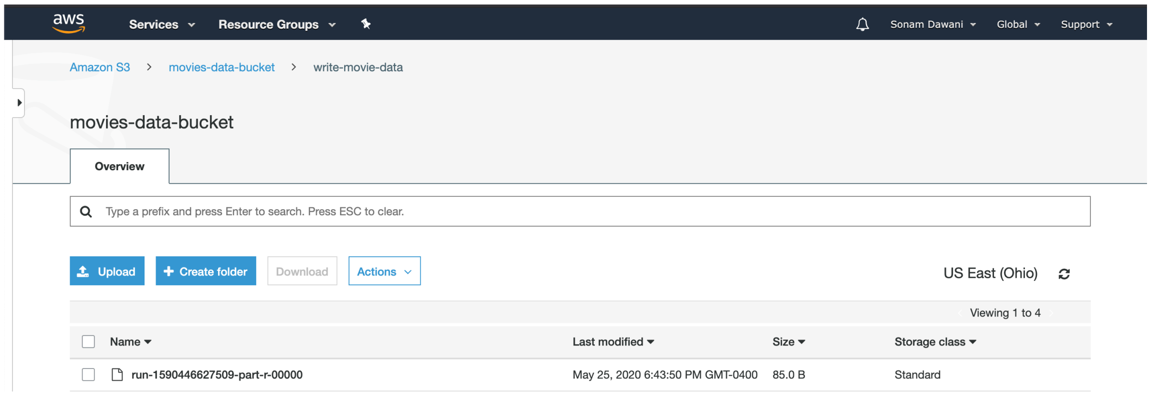
Jobs A job is your business logic required to perform extract, transform and load (ETL) work. Job runs are initiated by triggers which can be scheduled or driven by events.

Name	Type	ETL language	Script location	Last modified	Job bookmark
movie-data-job	Spark	python	s3://aws-glue-…	25 May 2020 3:01 PM U…	Disable

```
###############################################  EXTRACT  ###############################################
#Log starting time
dt_start = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
print("Start time:", dt_start)
#Read movie data to glue dynamic frame
dynamic_frame_read = glue_context.create_dynamic_frame.from_catalog(database = glue_db, table_name = glue_tbl)
#Convert dynamic frame to data frame to use standard pyspark functions
data_frame = dynamic_frame_read.toDF()

###############################################  TRANSFORM (MODIFY DATA)   ###############################################
#Create from a decade column  year
decade_col = f.floor(data_frame["Year"] / 10) * 10
data_frame = data_frame.withColumn("decade", decade_col)

#Group by decade: Count movies, get average rating
data_frame_aggregated = data_frame.groupby("decade").agg(
    f.count(f.col("Film")).alias('movie_count'),
    f.mean(f.col("Rotten Tomatoes %")).alias('rating mean'),
)

#Sort by the number of movies per decade
```

The script written for this job extract the ‘read_movie_data’ in a dataframe, performs transformation steps and load the resultant dataframe as CSV in the ‘write-movie-data’ folder of the S3 bucket.

In the transformation steps the movies are group by decades and mean rating of each decade is calculated.



Name	Last modified	Size	Storage class
run-1590446627509-part-r-00000	May 25, 2020 6:43:50 PM GMT-0400	85.0 B	Standard

The transformed data:

run-1590446627509-part-r-00000

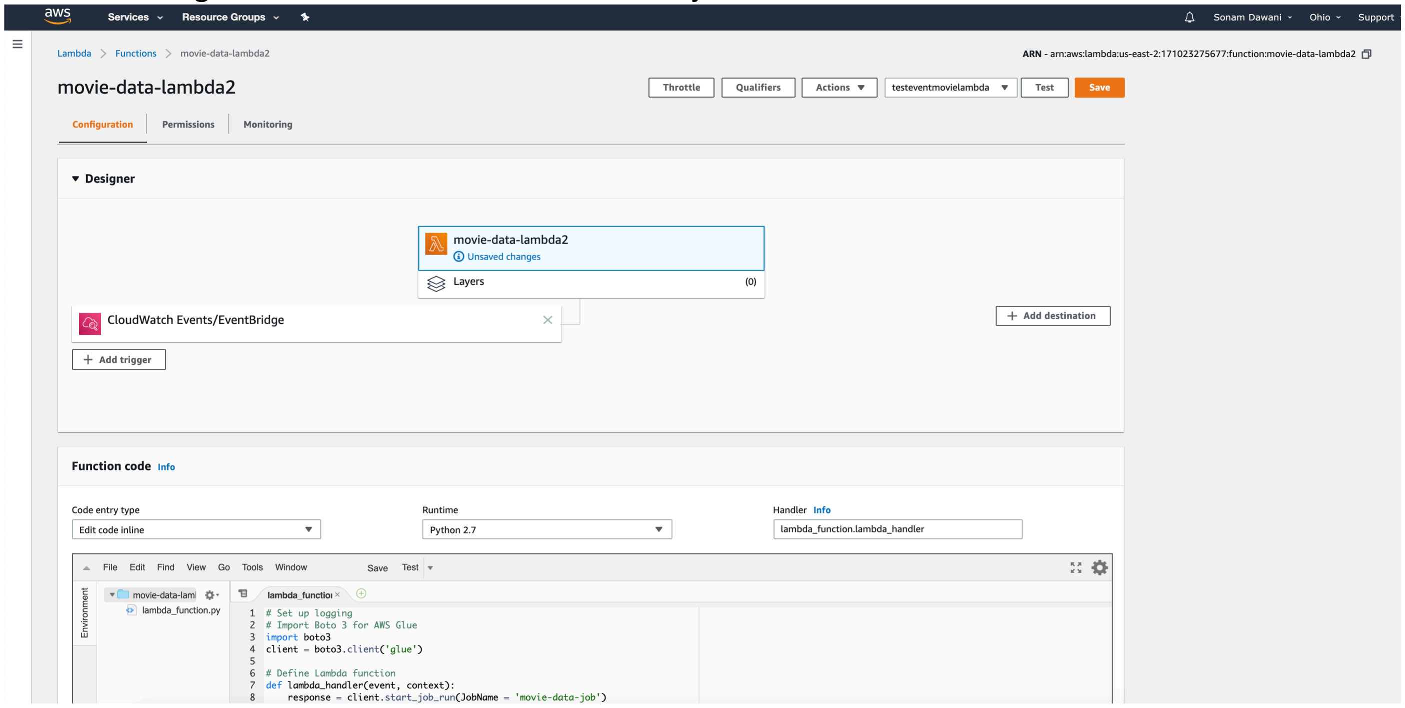
decade	movie_count	rating mean
2010	34	42.73529411764706
2000	43	49.83720930232558

Using AWS Lambda

AWS Lambda is used there to automate the process of running ETL job after the crawler execution is completed.

This is done in two steps:

- Creating the lambda function to start the ETL job

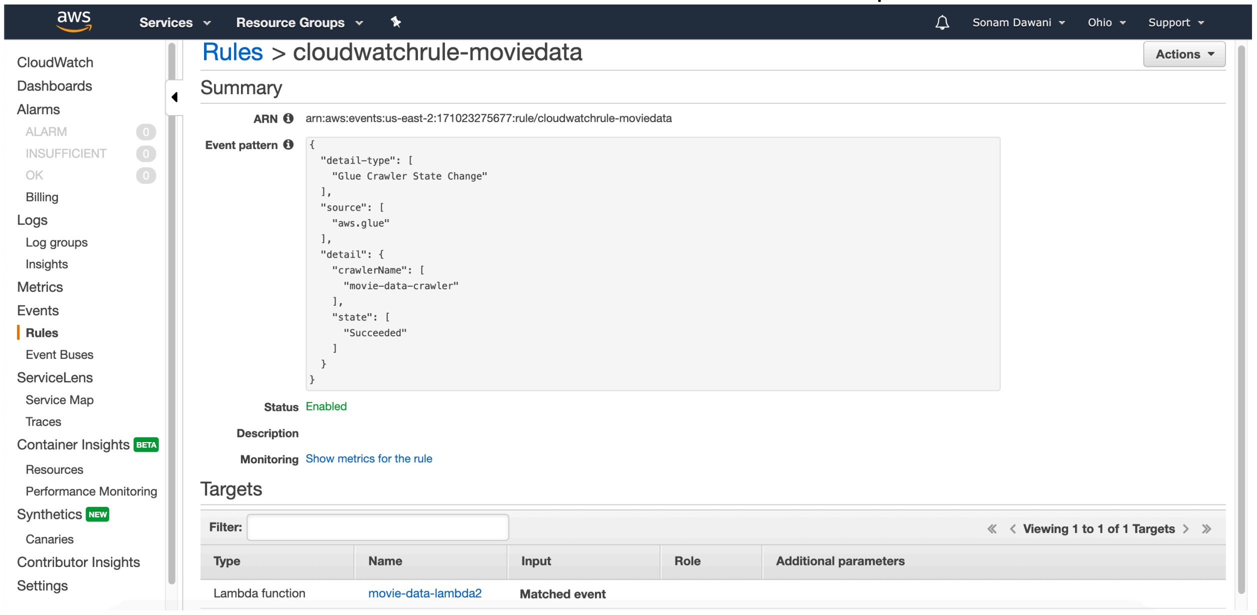


Code entry type: Edit code inline
Runtime: Python 2.7
Handler: lambda_function.lambda_handler

```
# Set up logging
# Import Boto 3 for AWS Glue
import boto3
client = boto3.client('glue')

# Define Lambda function
def lambda_handler(event, context):
    response = client.start_job_run(JobName = 'movie-data-job')
```

- Adding rule in CloudWatch

Created a rule in CloudWatch for the event of crawler. This rule is set to execute the lambda function once the execution of 'movie-data-crawler' is completed.



Rules > cloudwatchrule-moviedata

Summary

ARN arn:aws:events:us-east-2:171023275677:rule/cloudwatchrule-moviedata

Event pattern

```
{
  "detail-type": [
    "Glue Crawler State Change"
  ],
  "source": [
    "aws.glue"
  ],
  "detail": {
    "crawlerName": [
      "movie-data-crawler"
    ],
    "state": [
      "Succeeded"
    ]
  }
}
```

Status Enabled

Description

Monitoring Show metrics for the rule

Targets

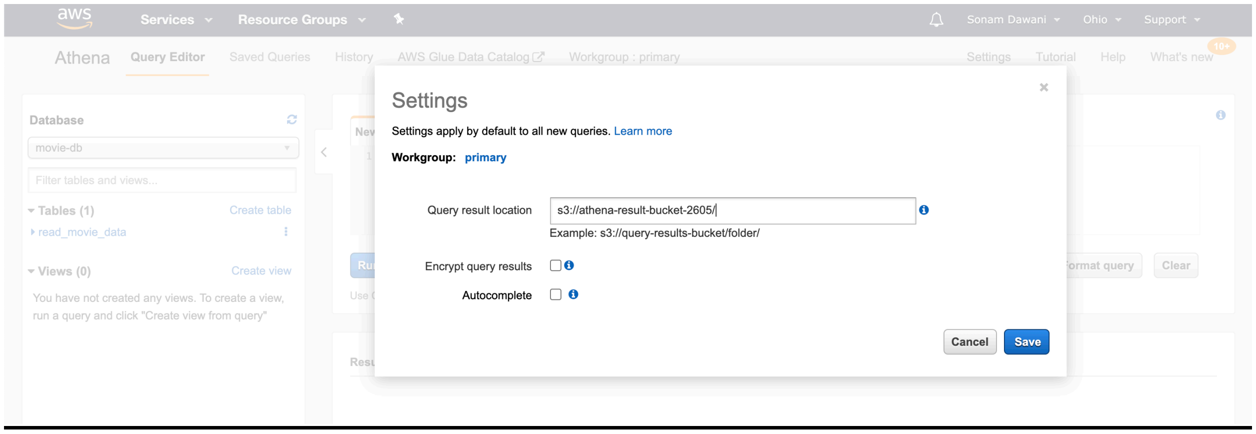
Filter: Viewing 1 to 1 of 1 Targets

Type	Name	Input	Role	Additional parameters
Lambda function	movie-data-lambda2	Matched event		

The steps above required to have **IAM roles** with some specific policies. Some of the IAM roles required to add Inline policies.

Using AWS Athena

For analysis purpose the database table was queries using AWS Athena. The results were stored in S3 bucket.



Settings

Settings apply by default to all new queries. Learn more

Workgroup: primary

Query result location s3://athena-result-bucket-2605/

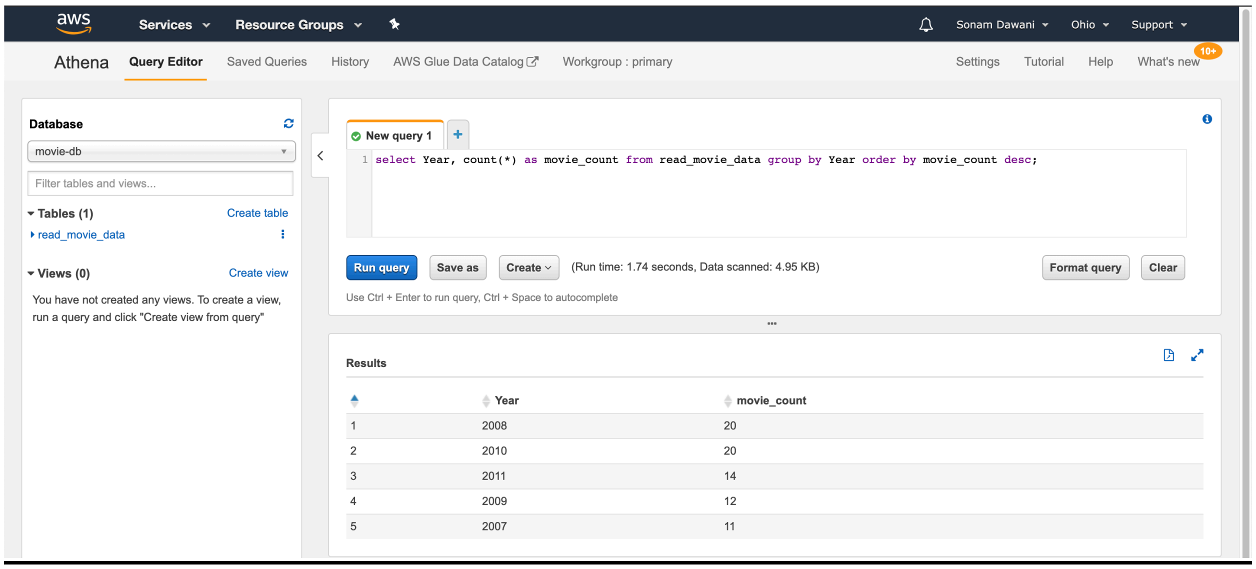
Example: s3://query-results-bucket/folder/

Encrypt query results ☐

Autocomplete ☐

Cancel Save

Sample of querying data:



You have not created any views. To create a view, run a query and click "Create view from query"

```
select Year, count(*) as movie_count from read_movie_data group by Year order by movie_count desc;
```

(Run time: 1.74 seconds, Data scanned: 4.95 KB)

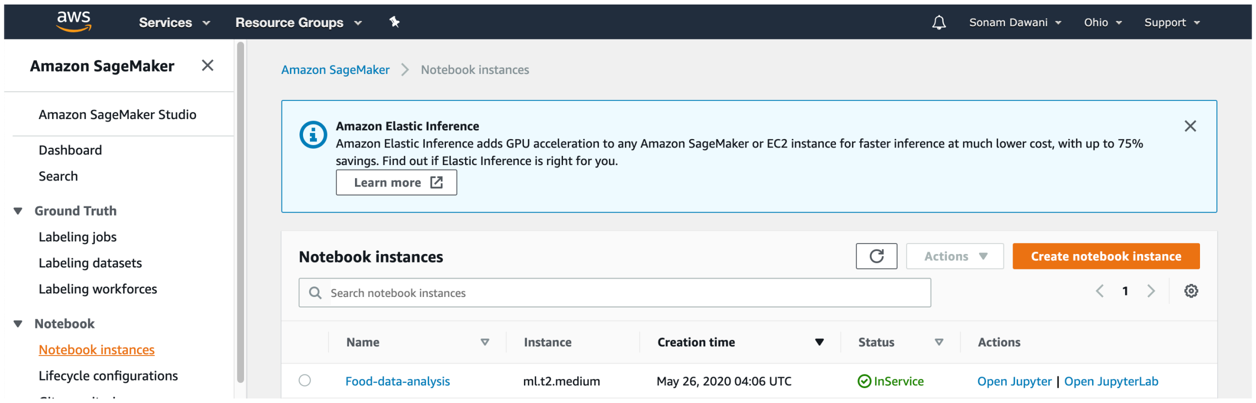
Use Ctrl + Enter to run query, Ctrl + Space to autocomplete

Results

	Year	movie_count
1	2008	20
2	2010	20
3	2011	14
4	2009	12
5	2007	11

Using AWS SageMaker

For SageMaker activity I used different dataset.
The dataset used here is food data containing recipe name, time to cook, nutritional values, ingredients etc. This data is collected from Food.com



Amazon Elastic Inference
Amazon Elastic Inference adds GPU acceleration to any Amazon SageMaker or EC2 instance for faster inference at much lower cost, with up to 75% savings. Find out if Elastic Inference is right for you.

Notebook instances

Name	Instance	Creation time	Status	Actions
Food-data-analysis	ml.t2.medium	May 26, 2020 04:06 UTC	InService	Open Jupyter | Open JupyterLab

The notebook contains:

- **Data reading from S3 bucket**
- Preprocessing
- Data Exploration
- Adding cuisine data using transfer learning
- Clustering of recipes using PCA and k-means
- Regression to predict nutritional value using Gradient Boosting
- Apriori / Market Basket Analysis

The github repo contains the PDF version of the notebook.
https://github.com/SONAMDAWANI/UsingAWS_DataAnalysis