Jaya Jagannatha Kalia Santa

Batch Time Analysis of Transactional Data

DESCRIPTION

Lenodo is a multinational e-commerce organization that sells products directly to consumers. The database administrator exports the data every night in a CSV file, but this export functionality is unused. Lenodo wants to use this data to uncover insights about the most-sold item and the countries where customers have bought this item.

You are a data analytics consultant, and you're asked to provide valuable insights and statistics across products, brands, categories, segments to the marketing, product, sales, and procurement teams and inform them about which product has the highest amount of sales and which product and its marketing needs the most improvement. These statistics will help to run effective digital marketing campaigns. The scope of this project is limited to data engineering and analysis.

Objective:

To use AWS Big Data stack for data engineering to analyze transactions, uncover patterns, and share actionable insights

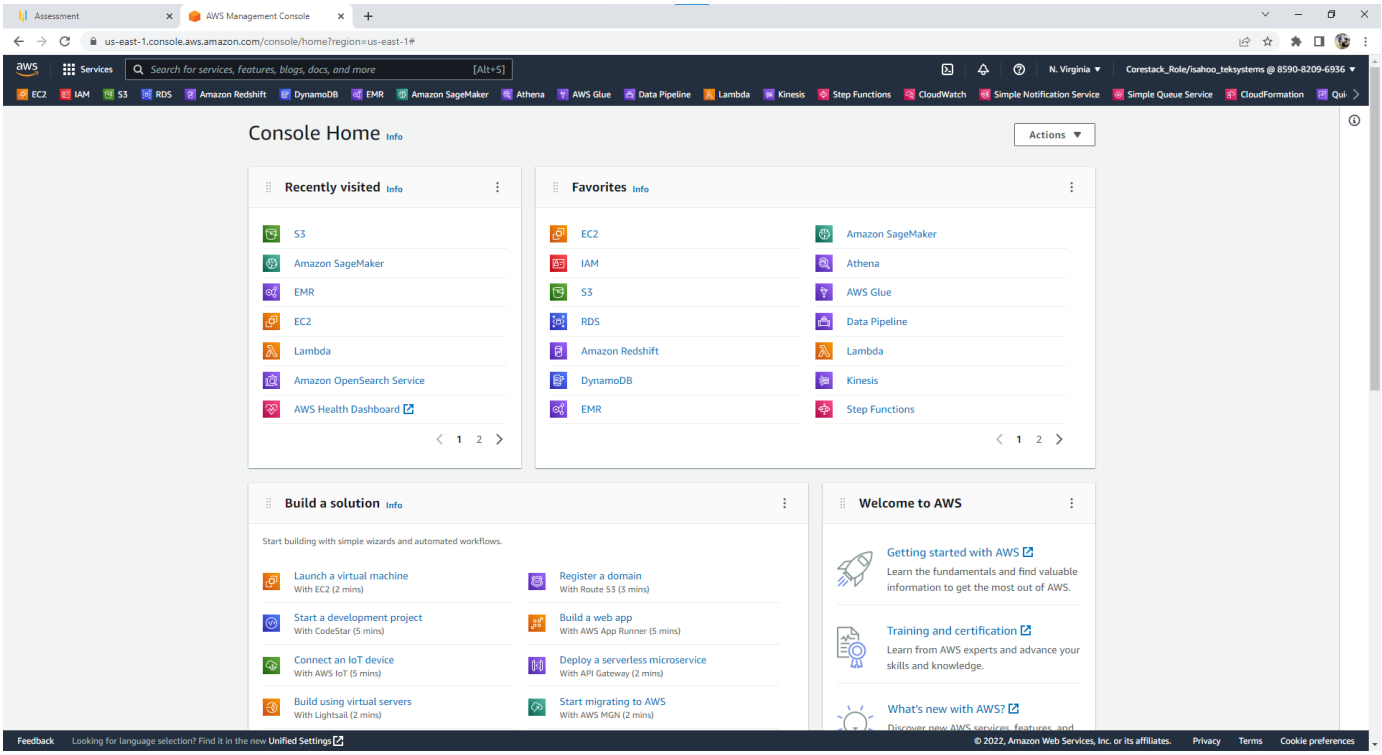
Steps to perform:

1. Create an S3 bucket with a unique name and upload the CSV file to the S3 bucket (ensure that the file is in UTF-8 format only)
2. Create a crawler to crawl the CSV data and generate a metadata catalog
3. Create a Glue job to transform the data into the Parquet format as CSV is not optimal for data warehouse queries
4. Add another crawler to crawl the Parquet data files to generate the metadata catalog of the Parquet file in order to query it with Athena
5. Query the data to identify the best-selling item and countries where customers have bought the most-sold item using Athena

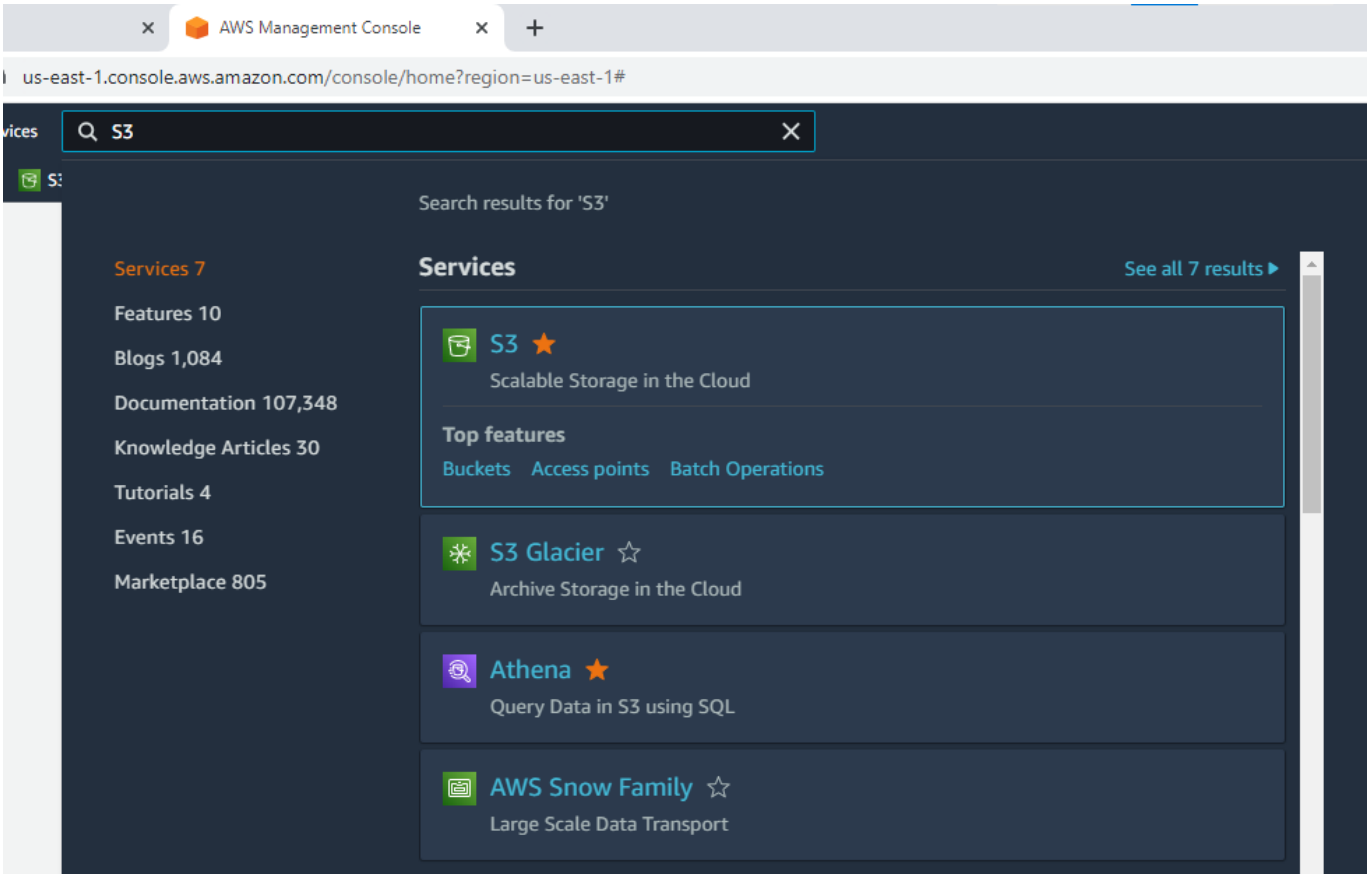
Submitted By: Ipsita Sahoo

Step 1: Create an S3 bucket and upload the csv file

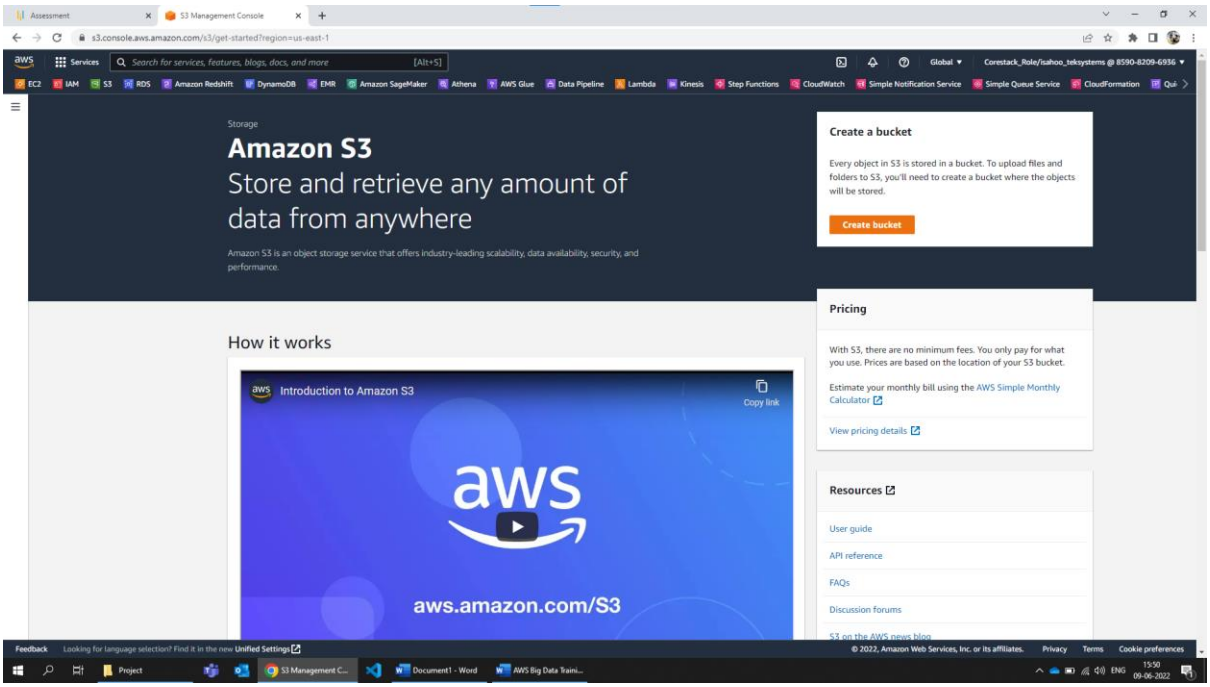
i. Open your AWS Management console home



ii. Search for S3 bucket

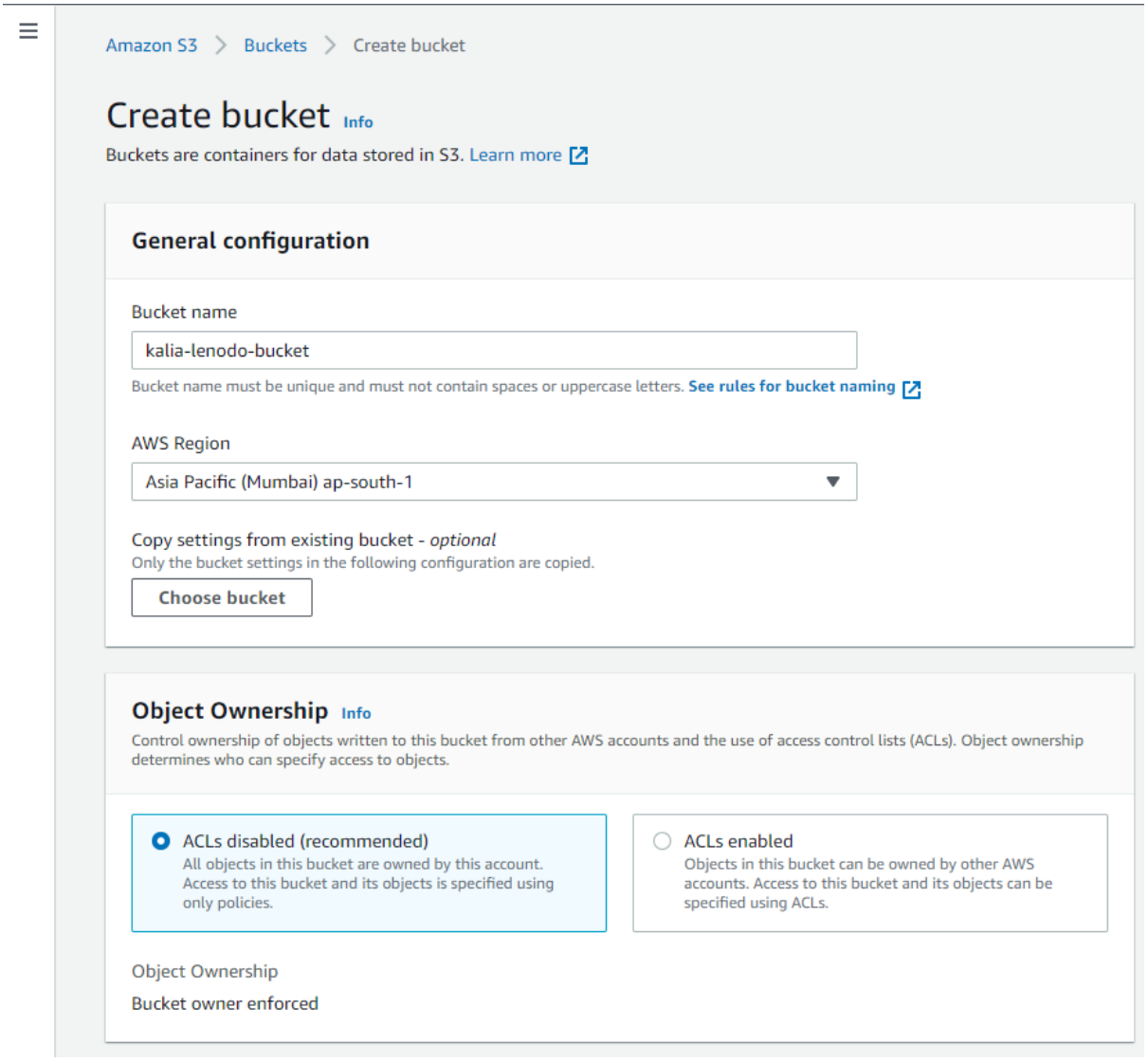


iii. Create a bucket

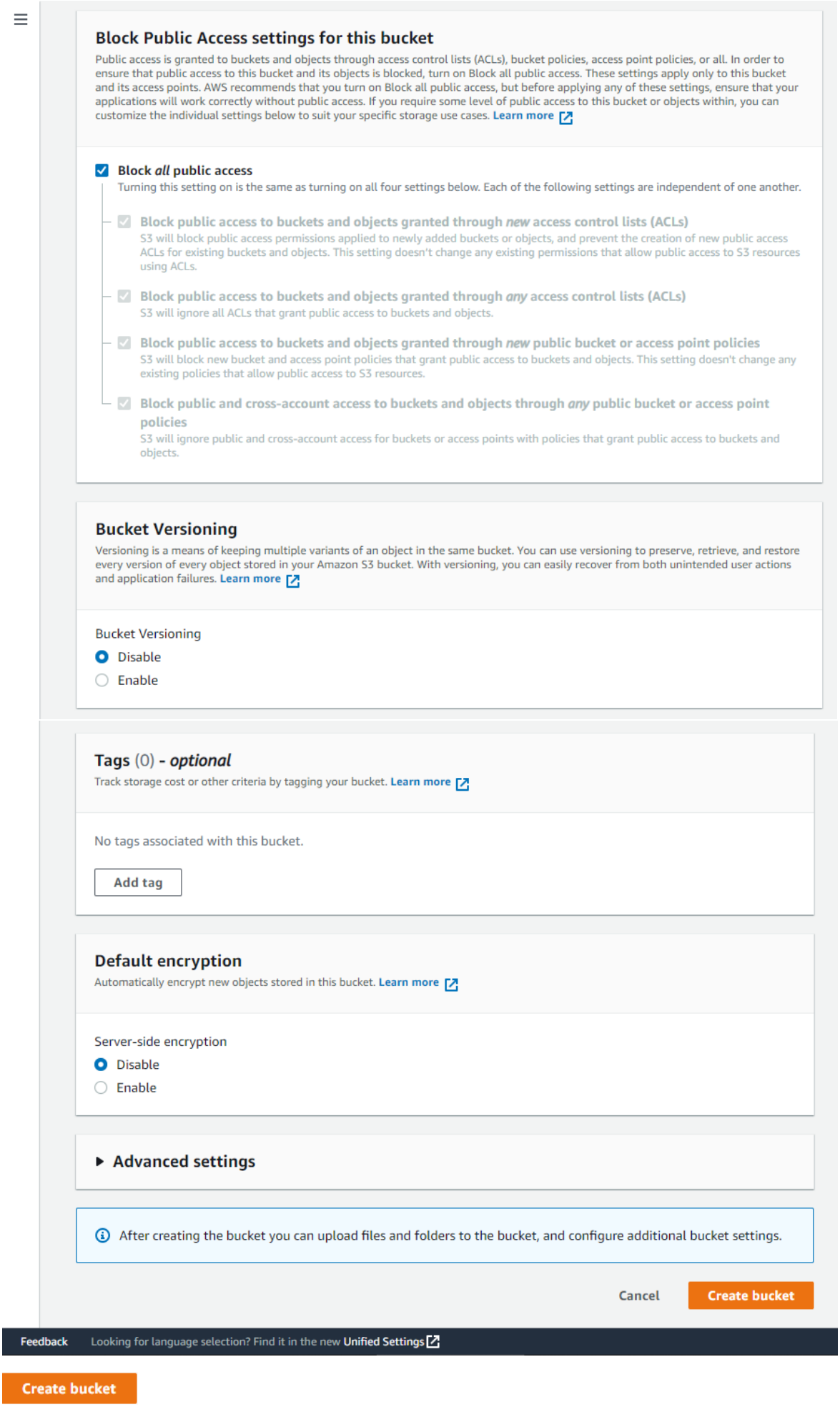


Create bucket

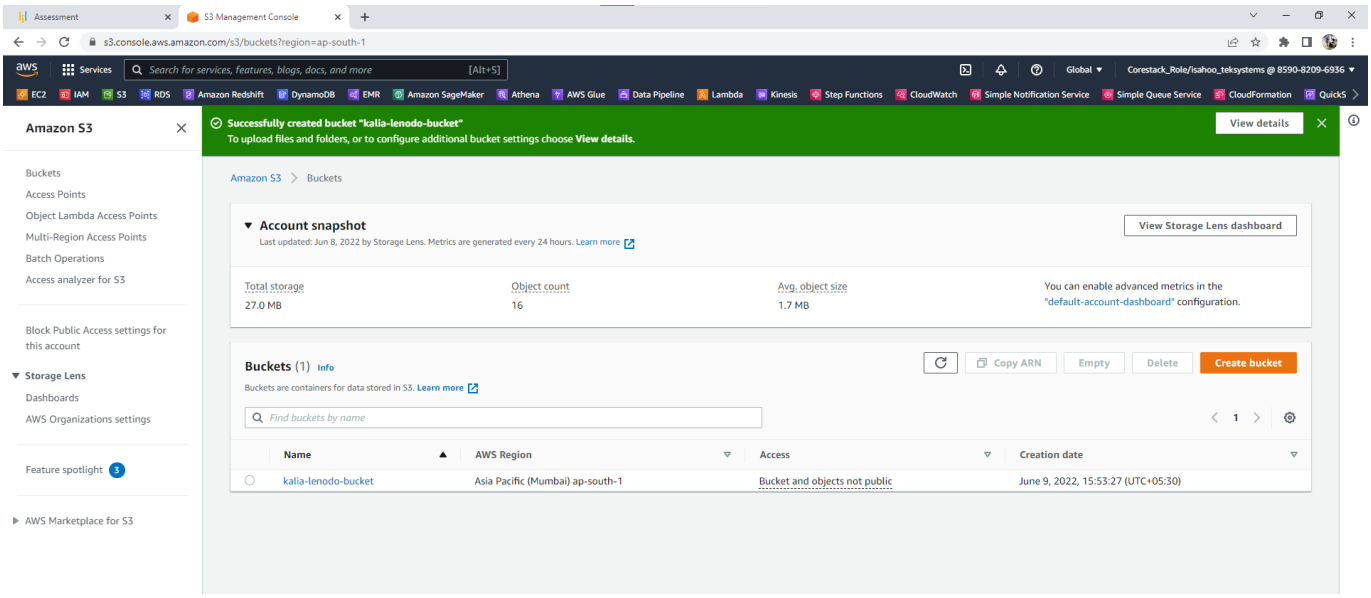
iv. Give the name to the bucket and if u want u can change the Region

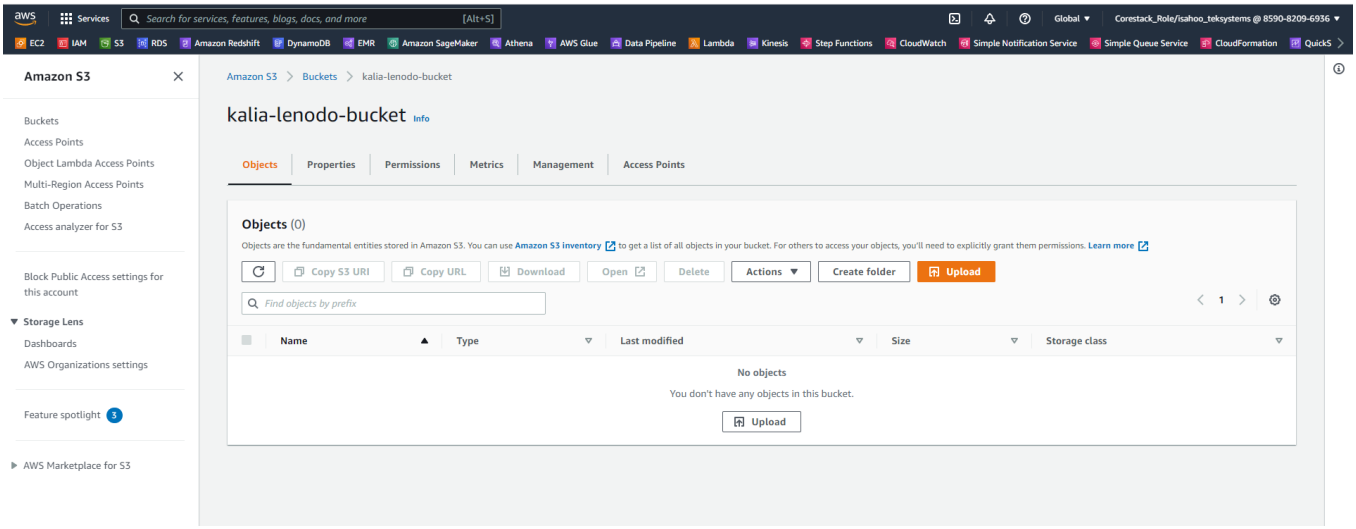


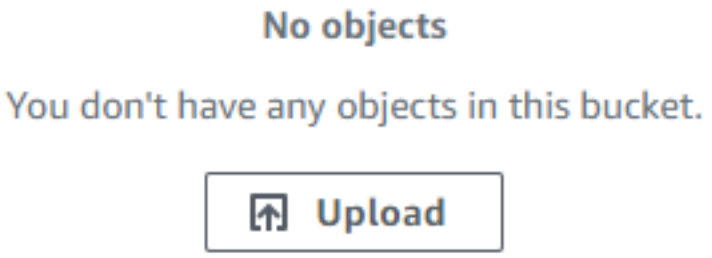
v. Keep all other options as default and create the bucket

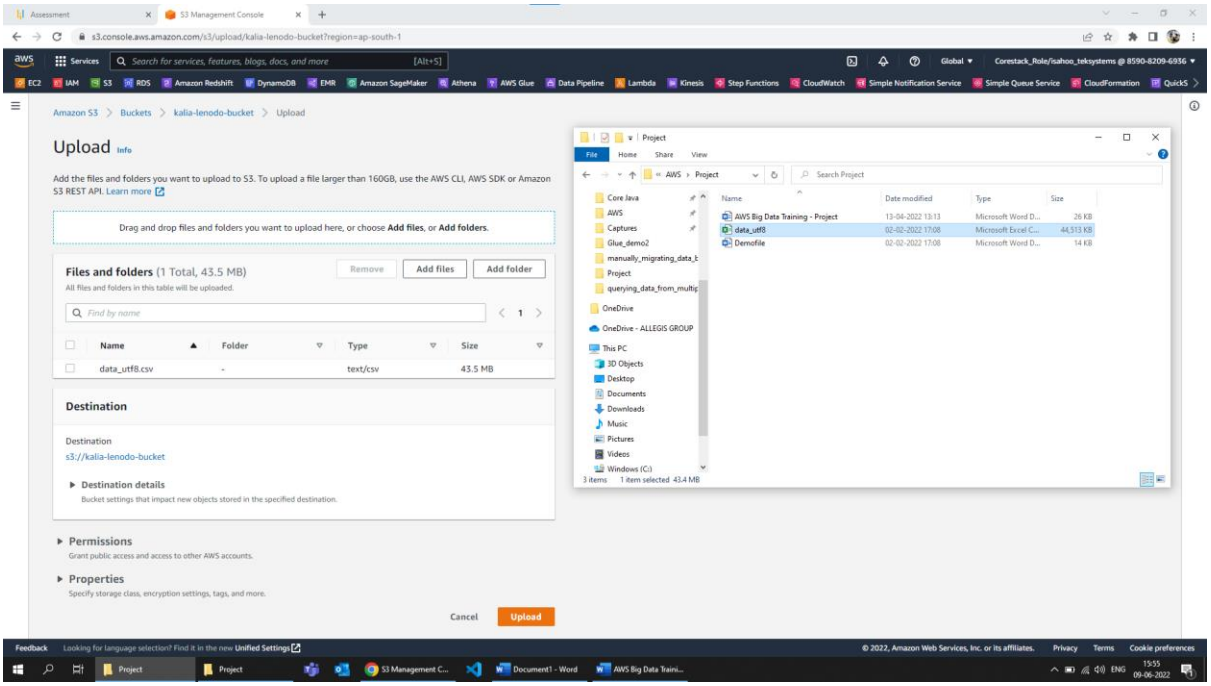


vi. Upload the file u want to work on in the bucket

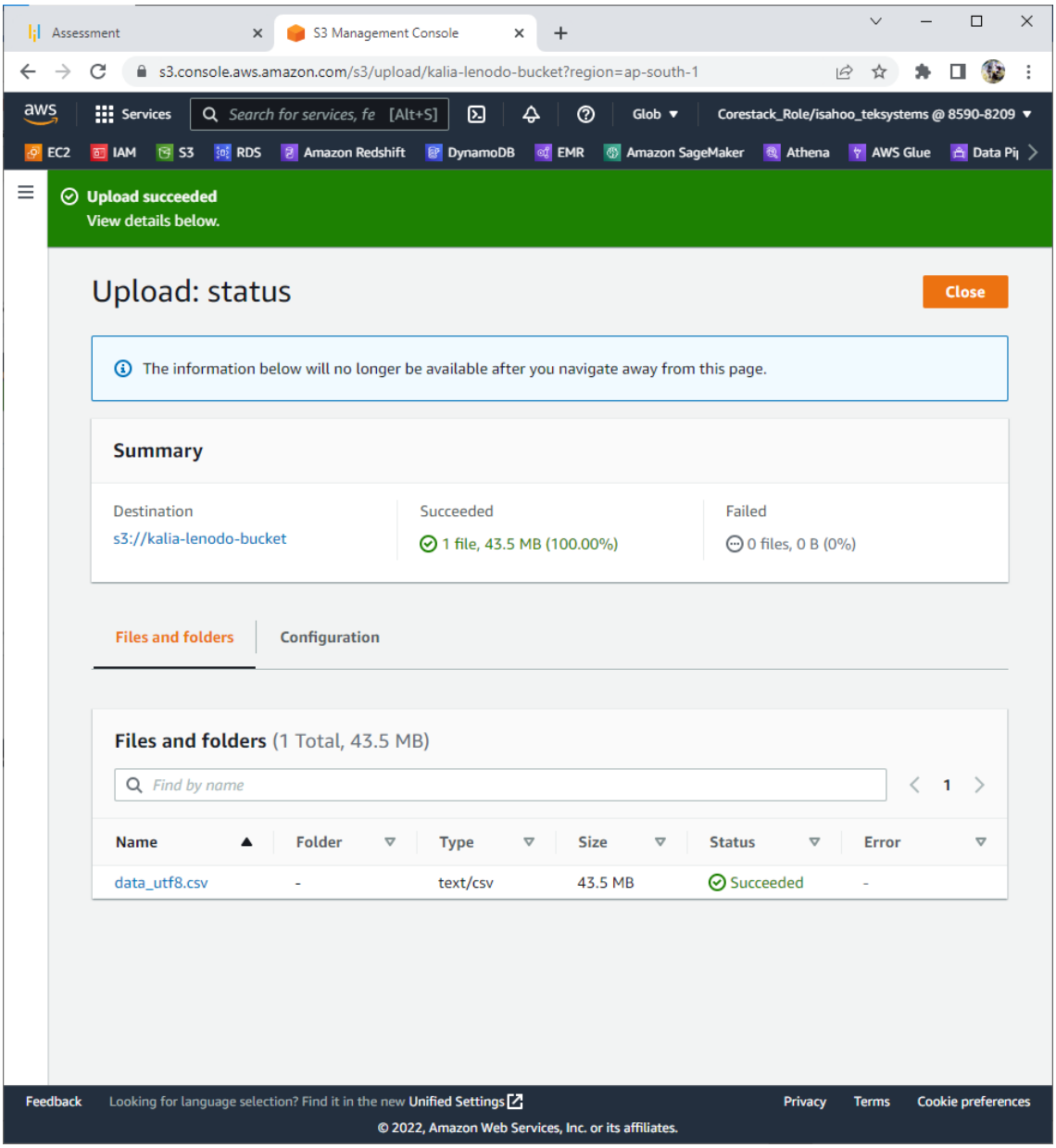






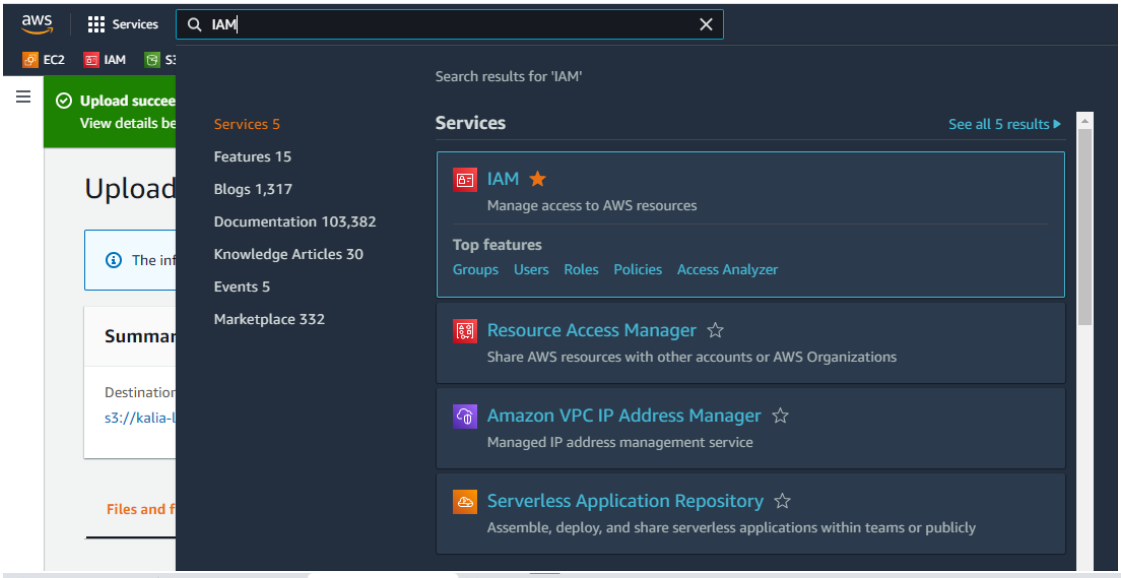

Amazon S3 > Buckets > kalia-lenodo-bucket > Upload
Upload
Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more
Drag and drop files and folders you want to upload here, or choose Add files, or Add folders.
Files and folders (1 Total, 43.5 MB)
All files and folders in this table will be uploaded.
Remove
Add files
Add folder
Find by name
Name
Folder
Type
Size
data_utf8.csv
-
text/csv
43.5 MB
Destination
Destination
s3://kalia-lenodo-bucket
Destination details
Bucket settings that impact new objects stored in the specified destination.
Permissions
Grant public access and access to other AWS accounts.
Properties
Specify storage class, encryption settings, tags, and more.
Cancel
Upload
Project
AWS Big Data Training - Project
data_utf8
Demofile
Feedback
© 2022, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

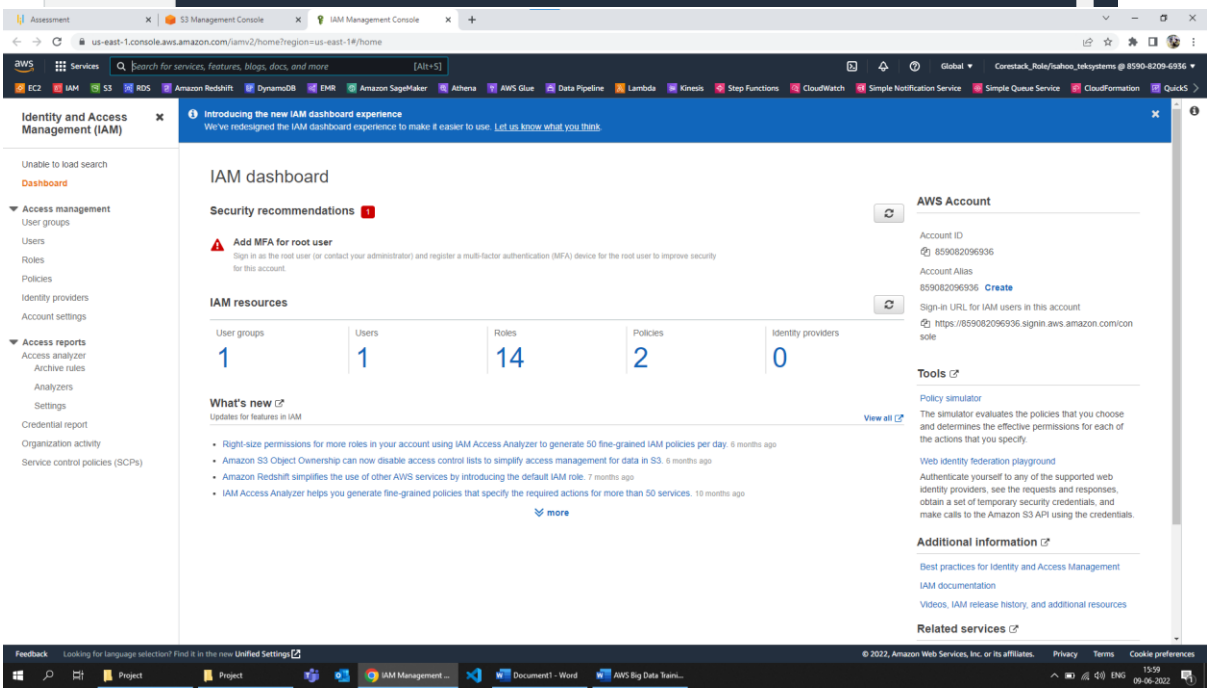
Upload


Assessment
S3 Management Console
s3.console.aws.amazon.com/s3/upload/kalia-lenodo-bucket?region=ap-south-1
Services
Search for services, fe [Alt+S]
Glob
Corestack_Role/isahoo_teksystems @ 8590-8209
EC2
IAM
S3
RDS
Amazon Redshift
DynamoDB
EMR
Amazon SageMaker
Athena
AWS Glue
Upload succeeded
View details below.
Upload: status
Close
The information below will no longer be available after you navigate away from this page.
Summary
Destination
s3://kalia-lenodo-bucket
Succeeded
1 file, 43.5 MB (100.00%)
Failed
0 files, 0 B (0%)
Files and folders
Configuration
Files and folders (1 Total, 43.5 MB)
Find by name
1
Name
Folder
Type
Size
Status
Error
data_utf8.csv
-
text/csv
43.5 MB
Succeeded
-
Feedback
Looking for language selection? Find it in the new Unified Settings
Privacy
Terms
Cookie preferences
© 2022, Amazon Web Services, Inc. or its affiliates.

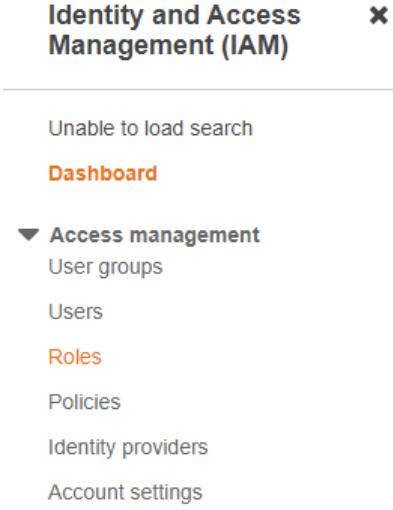
Step 2: Create an IAM role for Glue to access data from S3

i. Search for IAM on your Search bar

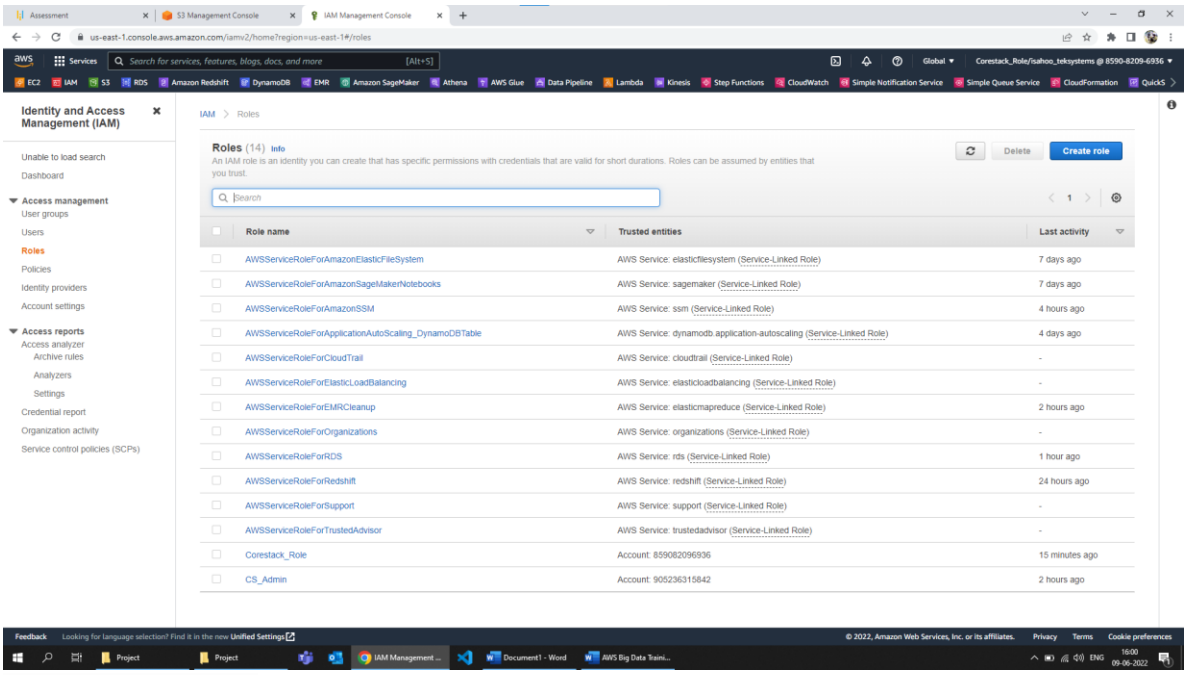




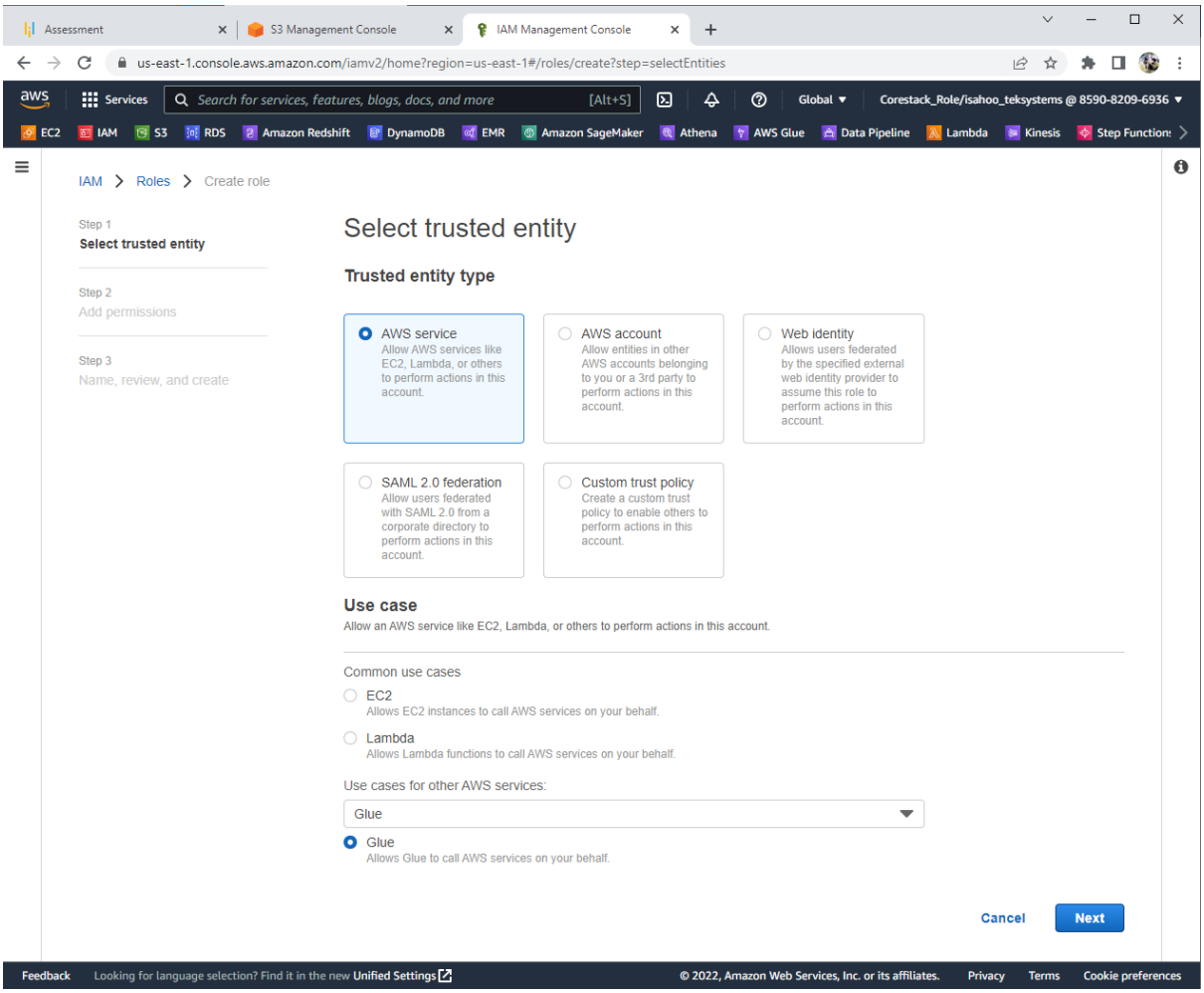
ii. Select Roles on the left pannel



iii. Click create role



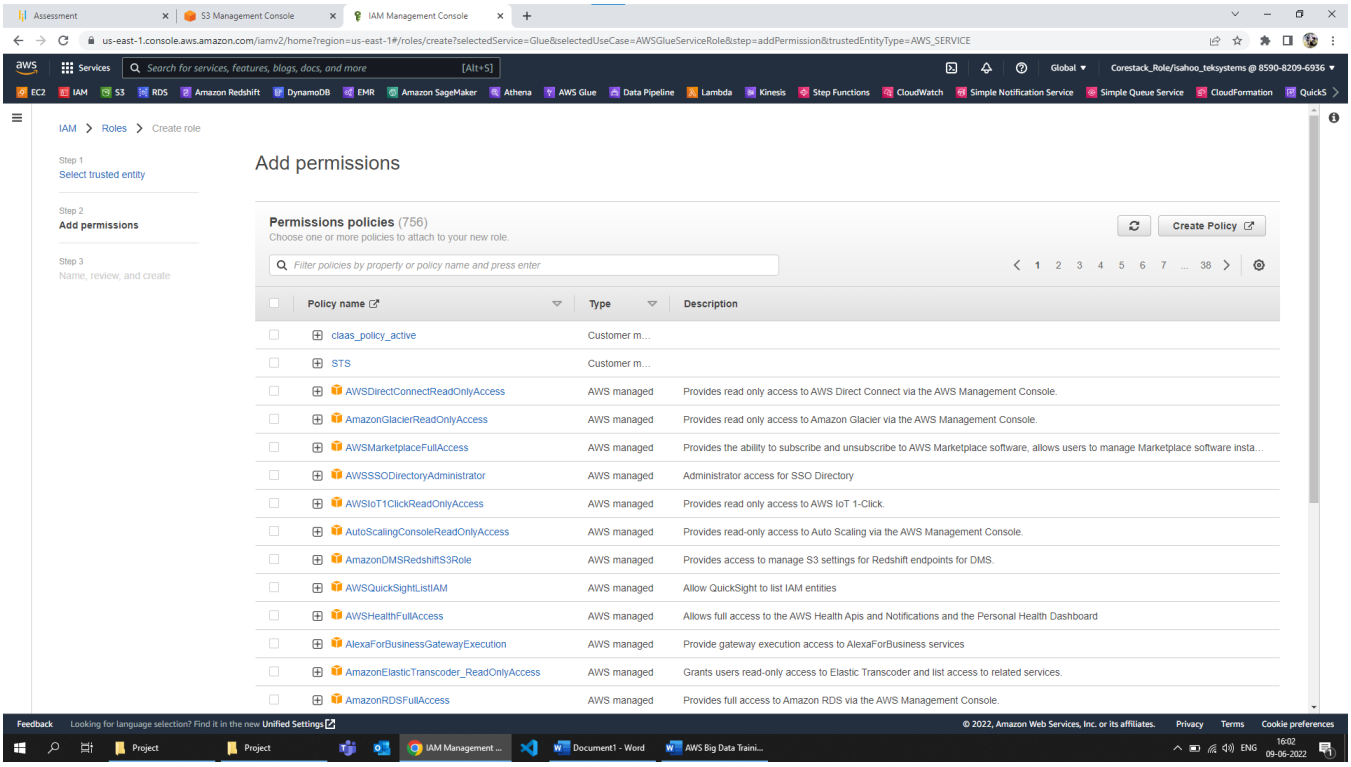
iv. Select AWS service and search for Glue and select it along with the radio button to allow Glue to call AWS services on your behalf and click next.

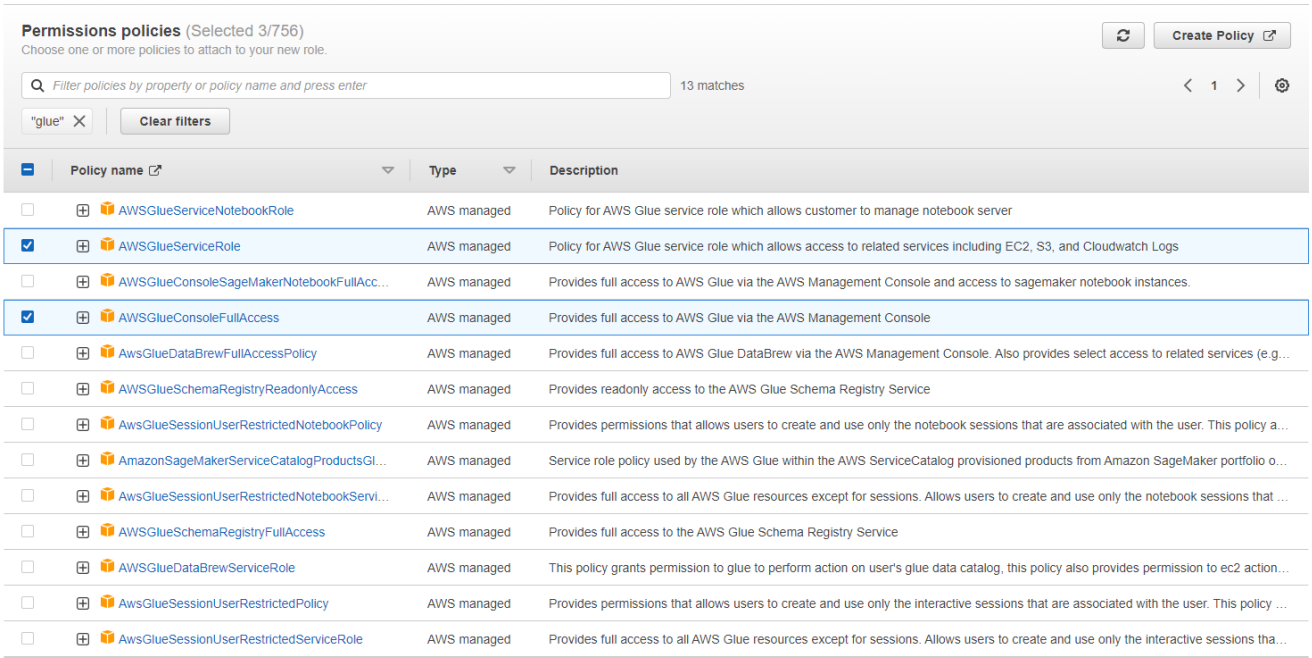


v. Search and select services as shown below you need to add:

a. S3 full access (To store and access data from S3 bucket)
b. Administrator access (Not really required but so as we do not face any issues further)
c. Glue Service role (Glue role)
d. Glue console full access (Glue role)

After done click next





Permissions policies (Selected 4/756)
Choose one or more policies to attach to your new role.

Create Policy

Filter policies by property or policy name and press enter — 35 matches — 1 2

"admin" | Clear filters

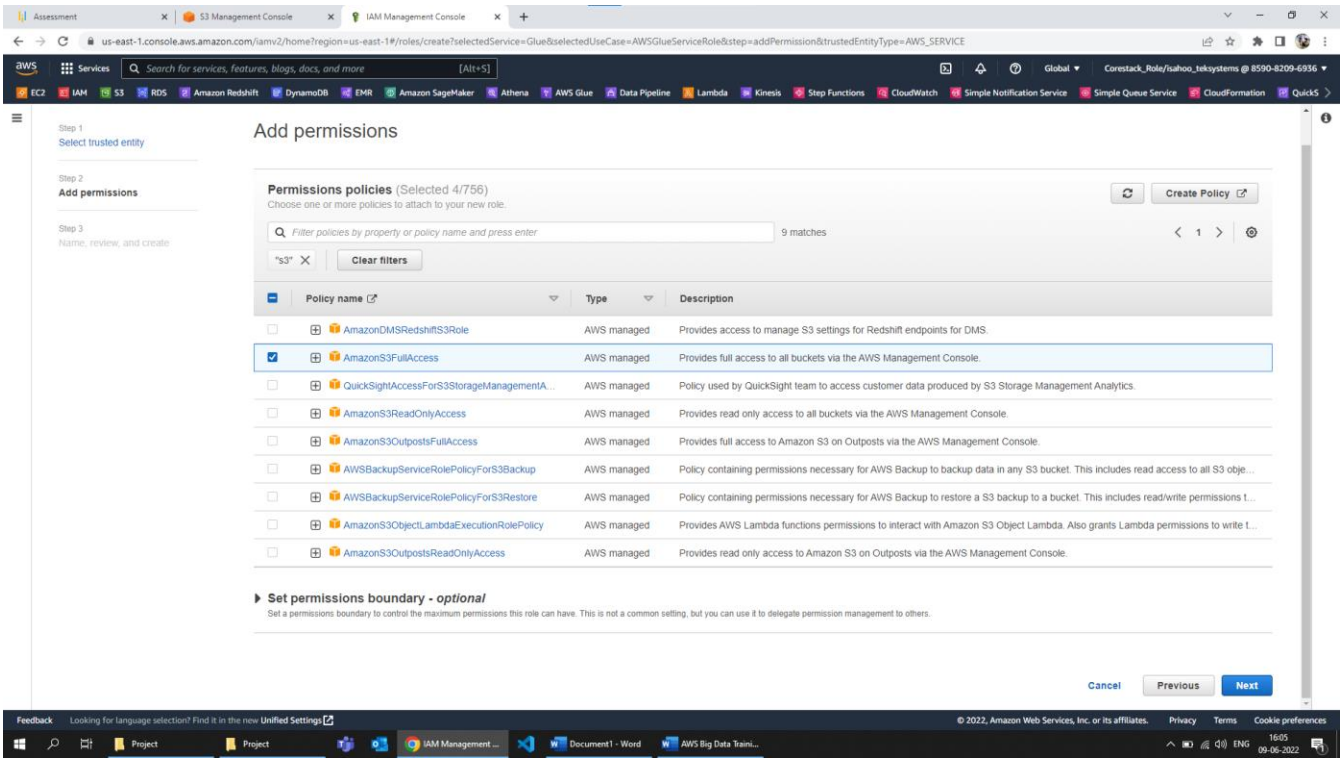
Policy name	Type	Description
AWSSSODirectoryAdministrator	AWS managed	Administrator access for SSO Directory
CloudWatchAgentAdminPolicy	AWS managed	Full permissions required to use AmazonCloudWatchAgent.
DatabaseAdministrator	AWS manag…	Grants full access permissions to AWS services and actions required to set up and configure AWS database services.
AWSSSOMasterAccountAdministrator	AWS managed	Provides access within AWS SSO to manage AWS Organizations master and member accounts and cloud application
AWSCloud9Administrator	AWS managed	Provides administrator access to AWS Cloud9.
AWSSSOMemberAccountAdministrator	AWS managed	Provides access within AWS SSO to manage AWS Organizations member accounts and cloud application
SystemAdministrator	AWS manag…	Grants full access permissions necessary for resources required for application and development operations.
AdministratorAccess	AWS manag…	Provides full access to AWS services and resources.

Permissions policies (Selected 4/756)
Choose one or more policies to attach to your new role.

Filter policies by property or policy name and press enter — 9 matches

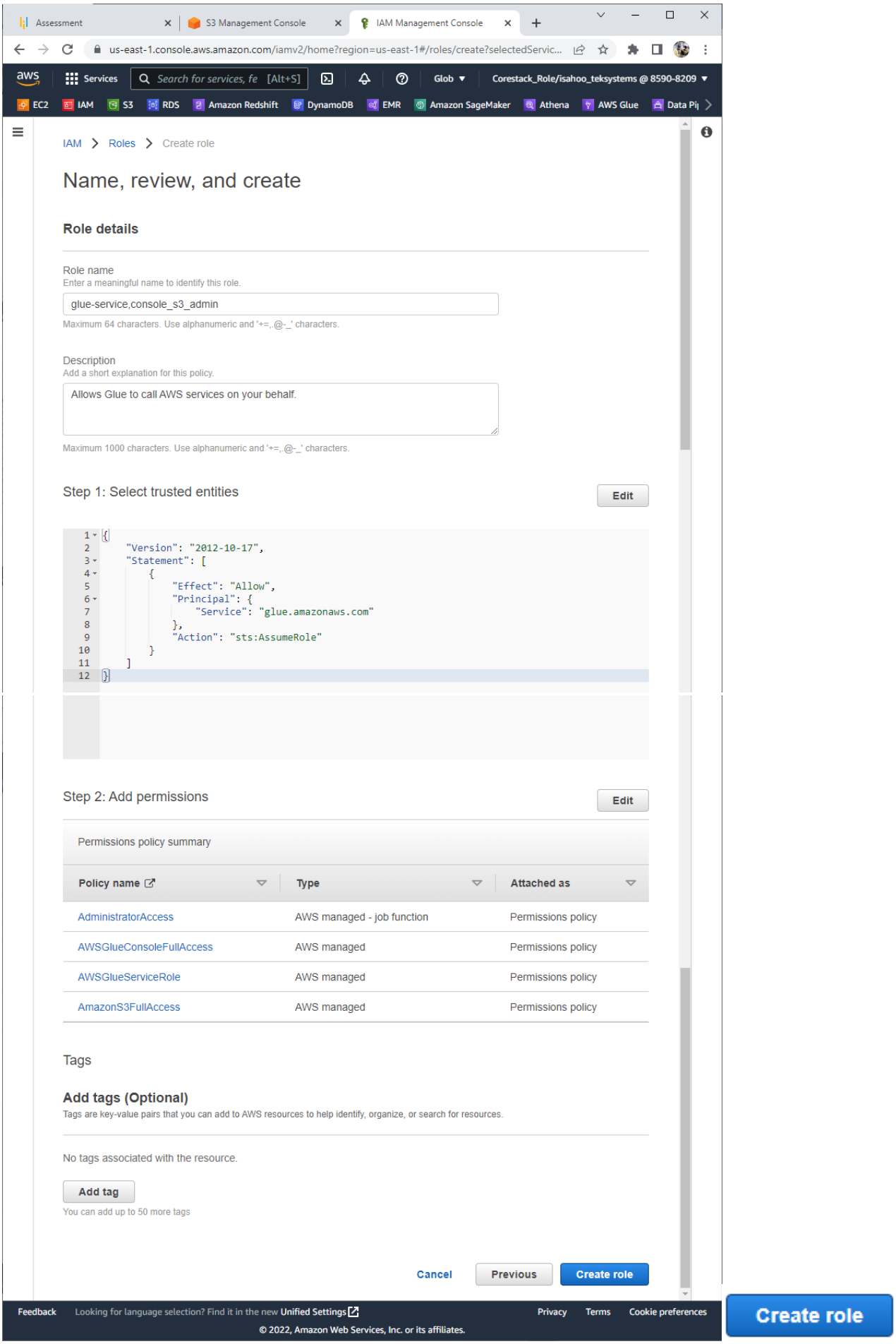
"s3" | Clear filters

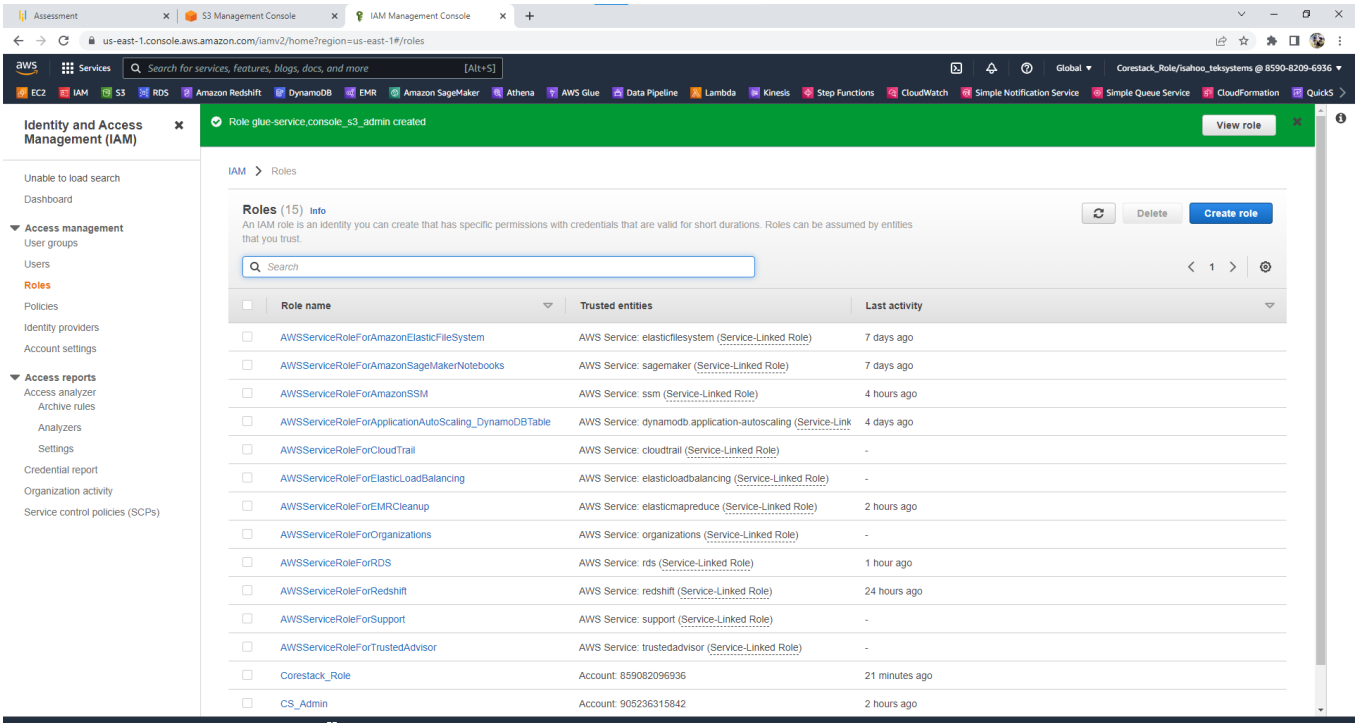
Policy name	Type	Description
AmazonDMSRedshiftS3Role	AWS managed	Provides access to manage S3 settings for Redshift endpoints for DMS.
AmazonS3FullAccess	AWS managed	Provides full access to all buckets via the AWS Management Console.

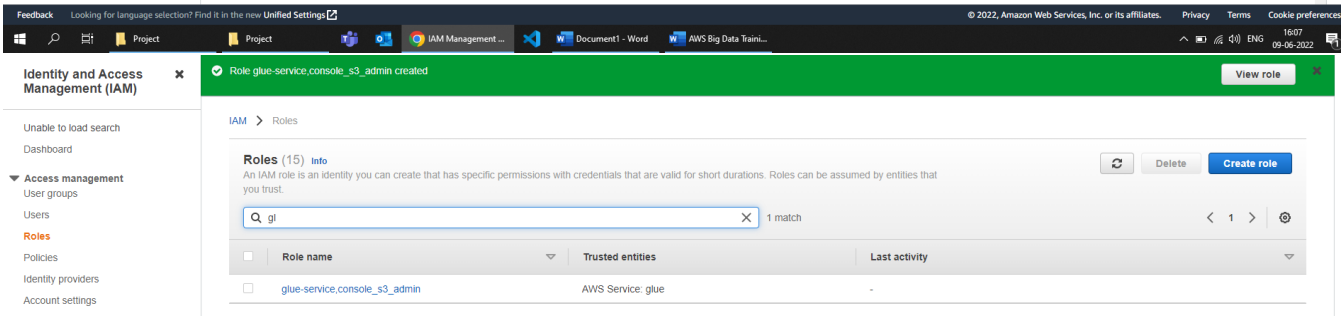


Next

vi. Give a name a to the role u created and click create role







IAM > Roles > glue-service,console_s3_admin

glue-service,console_s3_admin

Delete

Allows Glue to call AWS services on your behalf.

Summary

Edit

Creation date
June 09, 2022, 16:06 (UTC+05:30)

ARN
arn:aws:iam::859082096936:role/glue-service,console_s3_admin

Last activity
None

Maximum session duration
1 hour

Permissions | Trust relationships | Tags | Access Advisor | Revoke sessions

Permissions policies (4)

You can attach up to 10 managed policies.

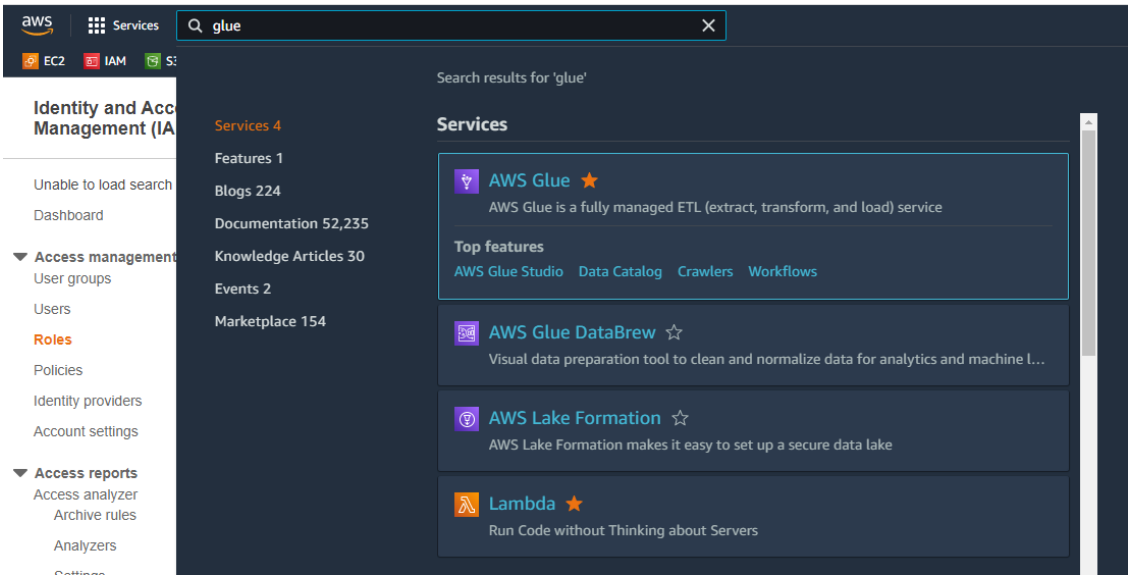
Simulate | Remove | Add permissions

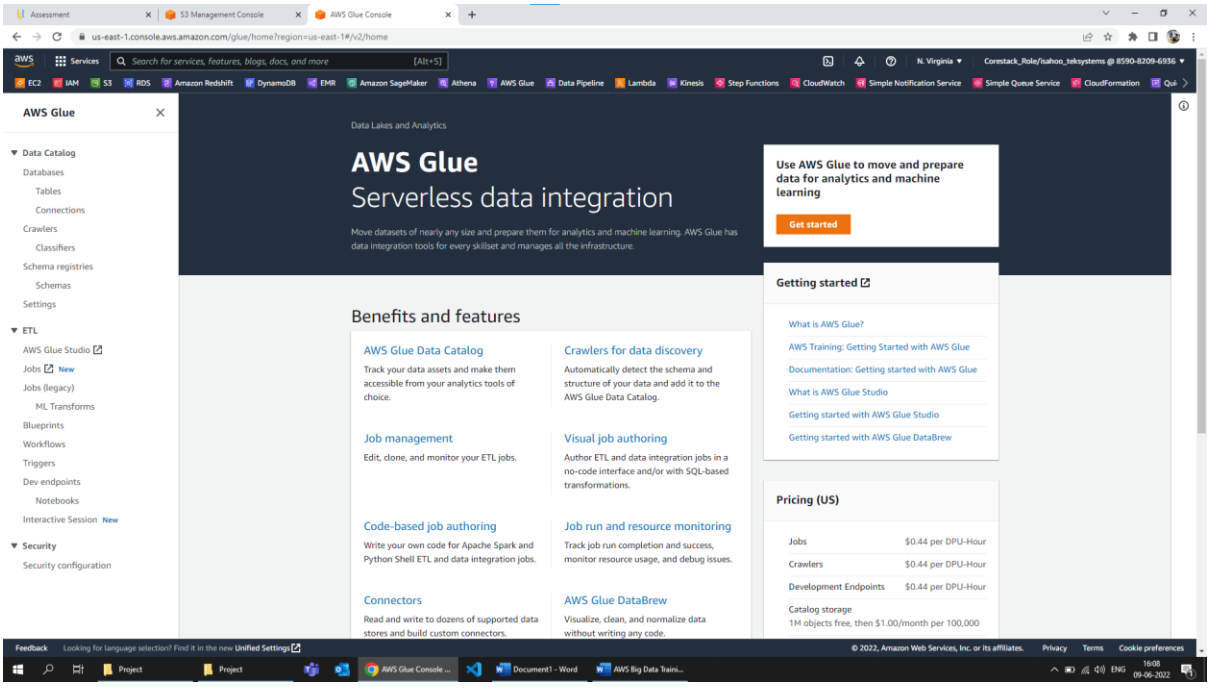
Filter policies by property or policy name and press enter

Policy name	Type	Description
AmazonS3FullAccess	AWS managed	Provides full access to all buckets via the AWS Management Console.
AWSGlueServiceRole	AWS managed	Policy for AWS Glue service role which allows access to related services including EC2, S3, and Cloudwatch Logs
AdministratorAccess	AWS managed - job function	Provides full access to AWS services and resources.
AWSGlueConsoleFullAccess	AWS managed	Provides full access to AWS Glue via the AWS Management Console

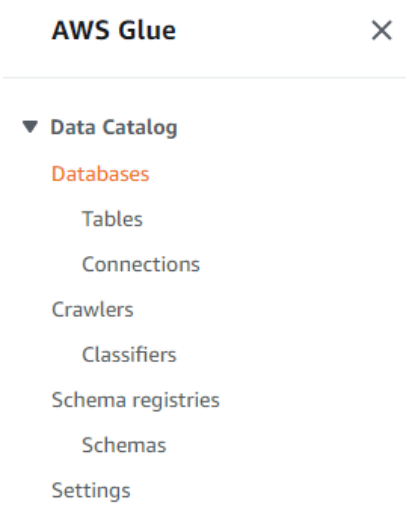
Step 3: Create a catalogue of the csv data

i. Search for Glue in the search bar

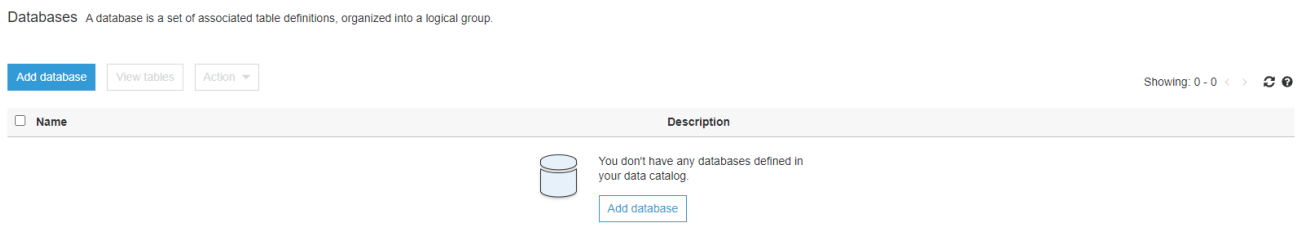


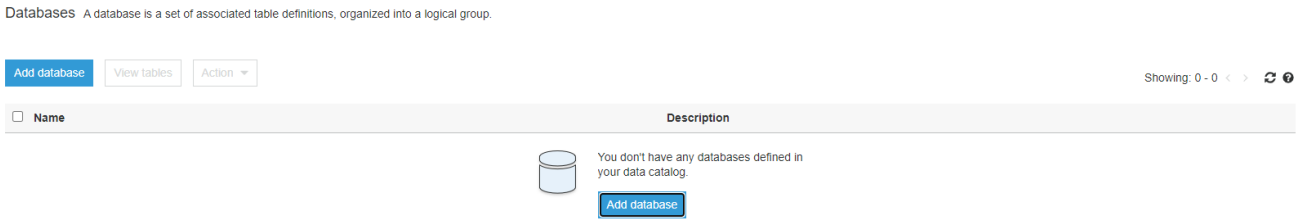


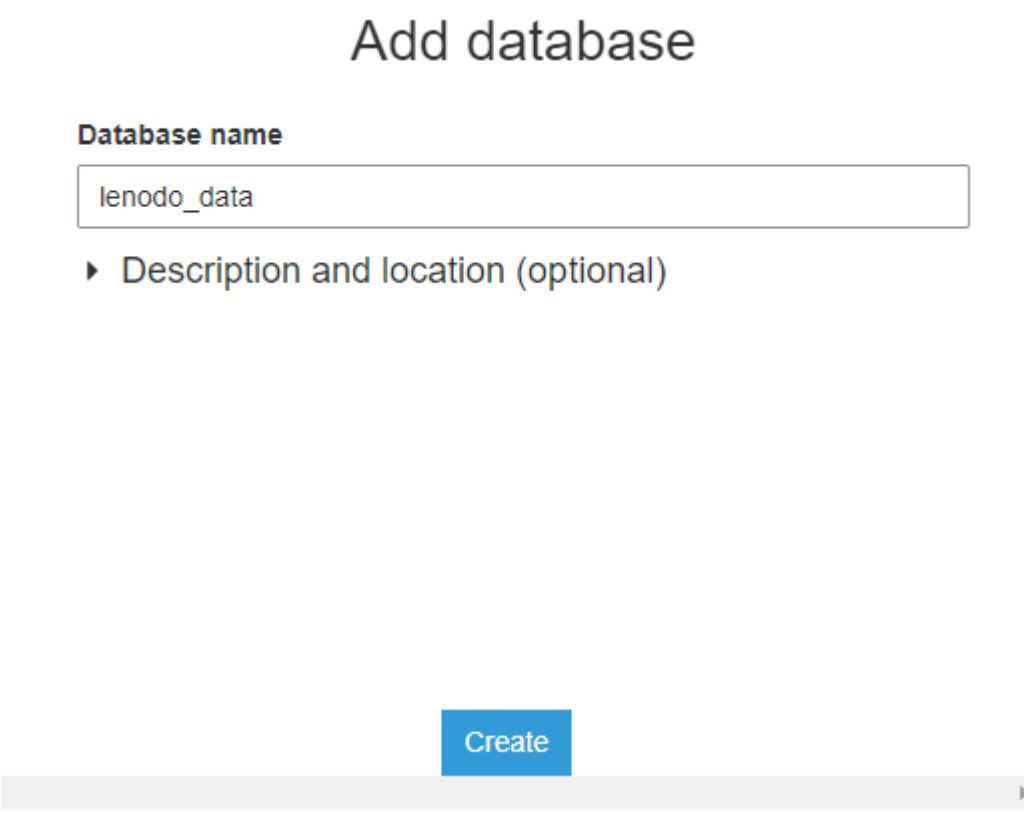
ii. Select databases in the left panel

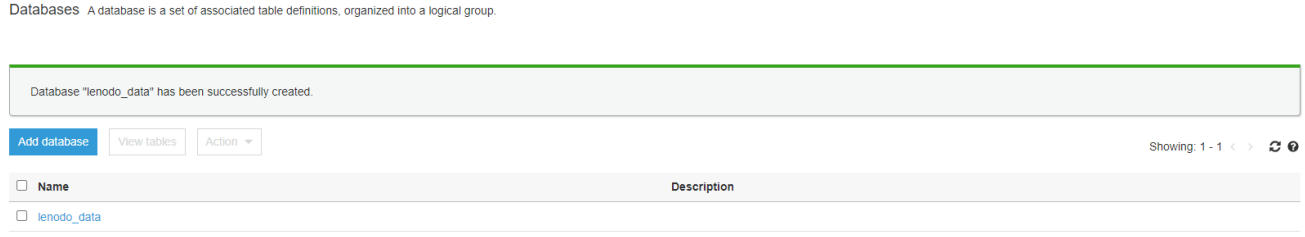


iii. Click add database give a name and click create

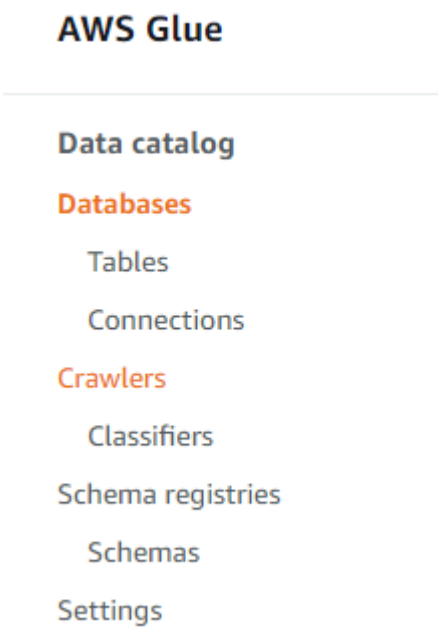




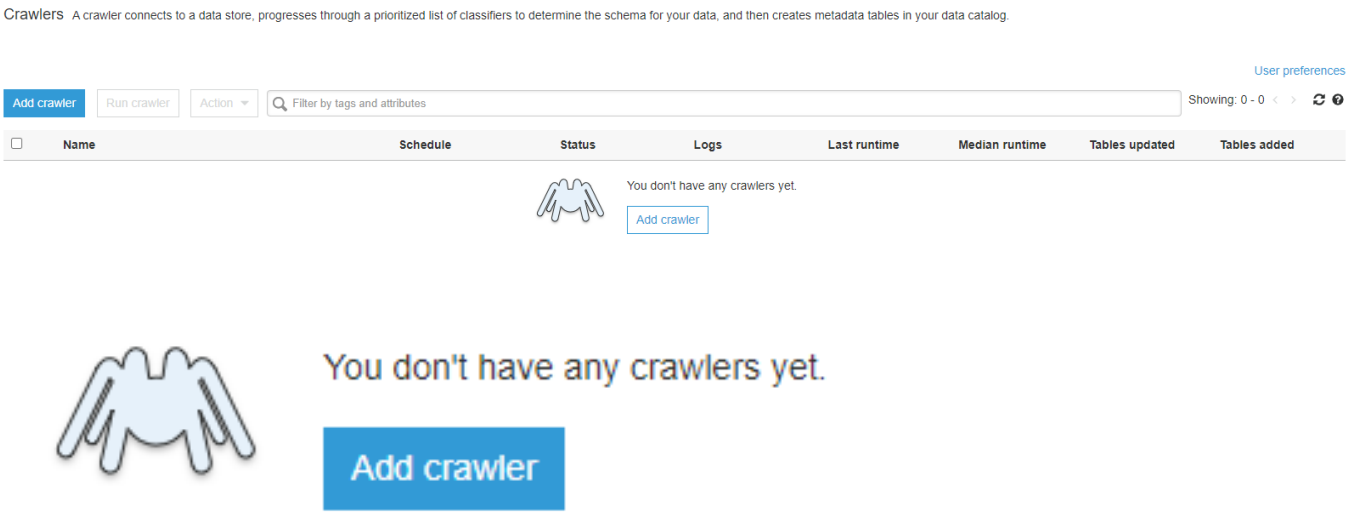




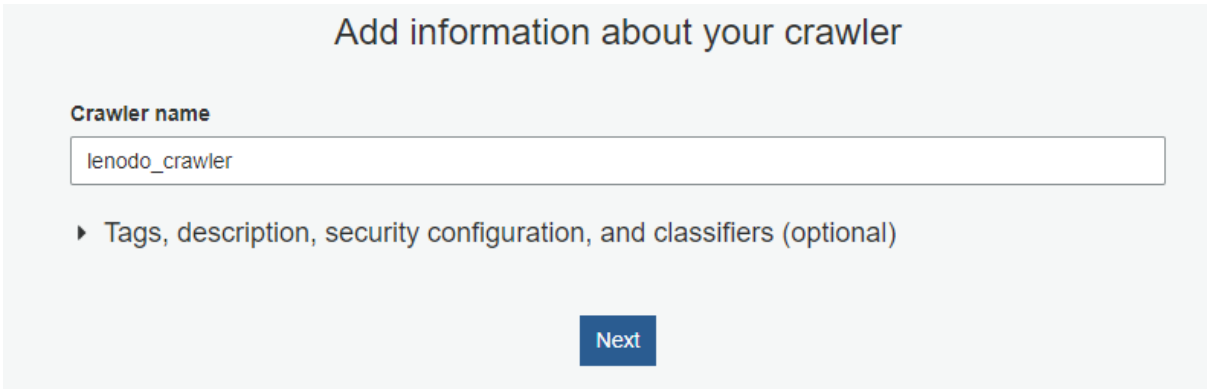
iv. Click on crawler on the left panel



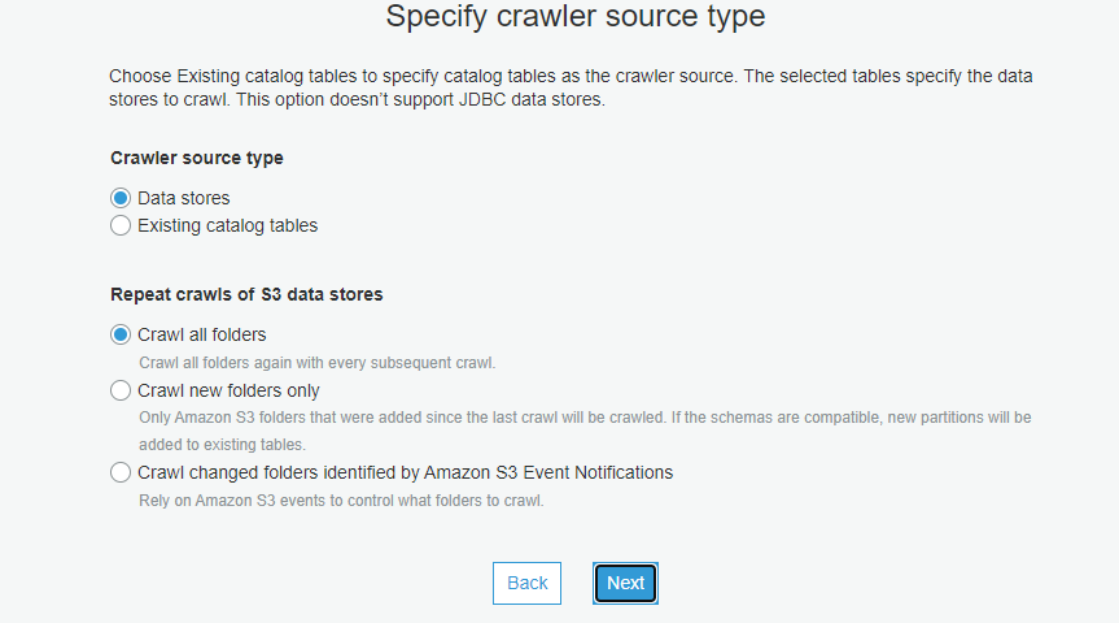
v. Select Add crawler



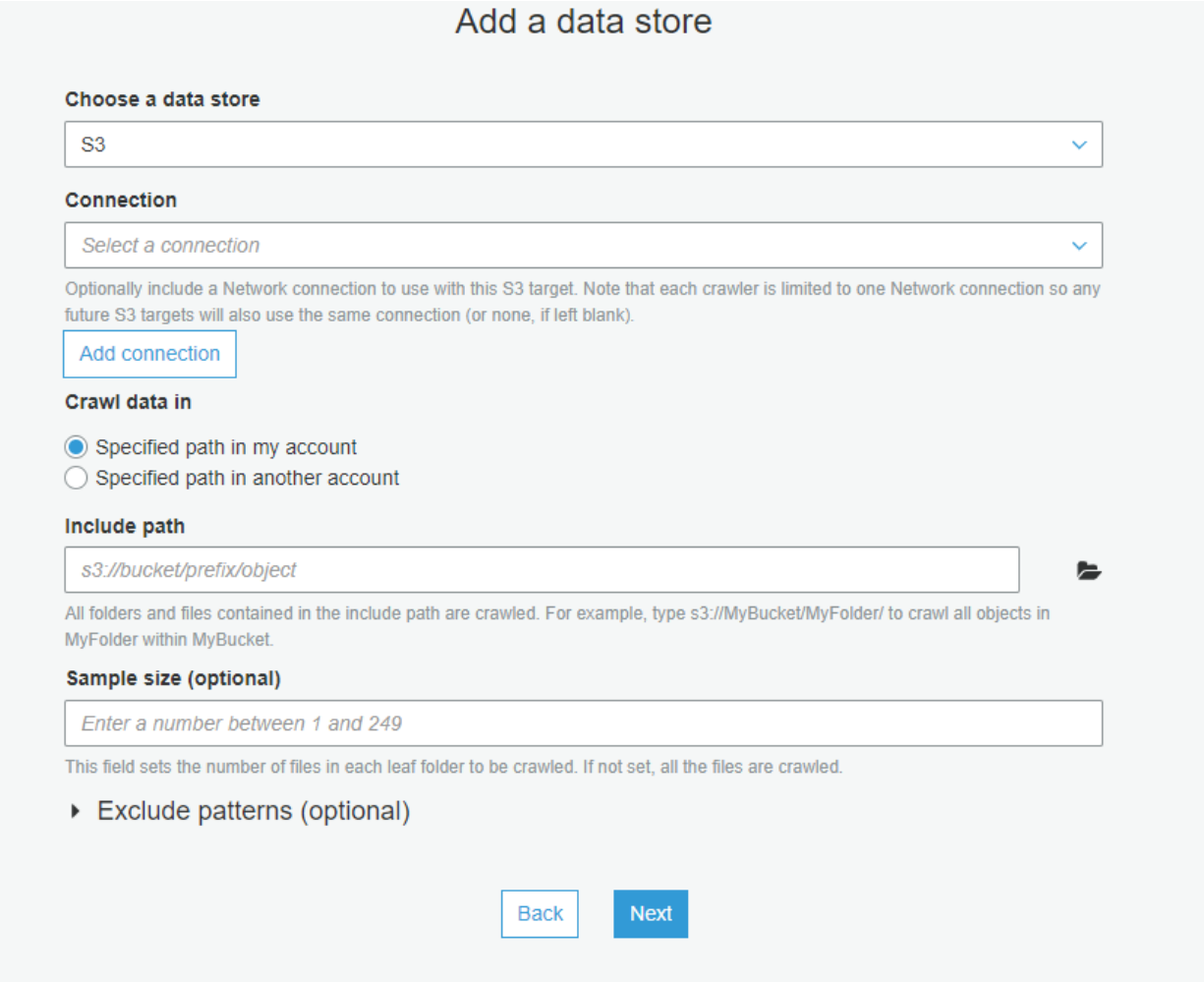
vi. Give the name and click next



vii. As we must crawl all files in the bucket (since we only have 1 csv file and nothing else in the folder currently) so go with Data Source option and to Crawl all folders which are basically the default selections and click next



viii. As our data source is in S3 so let it be S3, we must include the path to S3 so click on the folder to the right of Include Path choose the bucket (folder if it's in folder for my case it's in a bucket so I am selecting the bucket) and click select and click next to proceed.



Include path

s3://bucket/prefix/object

All folders and files contained in the include path are crawled. For example, type s3://MyBucket/MyFolder/ to crawl all objects in MyFolder within MyBucket.

Choose S3 path

- S3
  - kalia-lenodo-bucket
    - data_utf8.csv

Select

Add a data store

Choose a data store

S3

Connection

Select a connection

Optionally include a Network connection to use with this S3 target. Note that each crawler is limited to one Network connection so any future S3 targets will also use the same connection (or none, if left blank).

Add connection

Crawl data in

Specified path

Include path

s3://kalia-lenodo-bucket

All folders and files contained in the include path are crawled. For example, type s3://MyBucket/MyFolder/ to crawl all objects in MyFolder within MyBucket.

Sample size (optional)

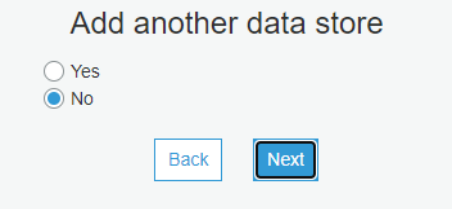
Enter a number between 1 and 249

This field sets the number of files in each leaf folder to be crawled. If not set, all the files are crawled.

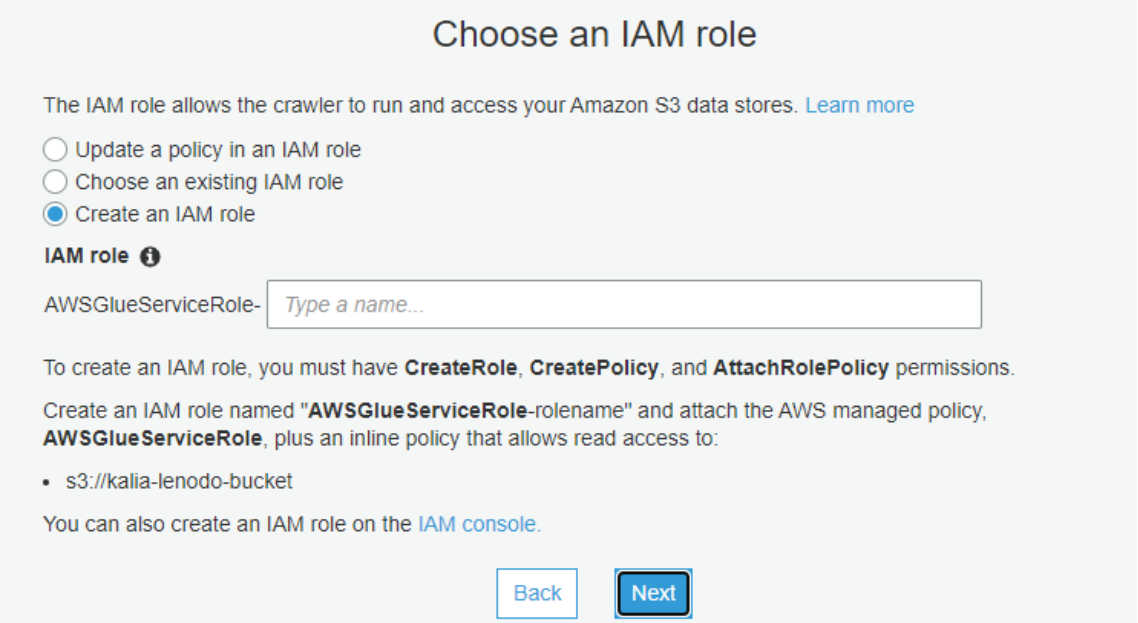
▸ Exclude patterns (optional)

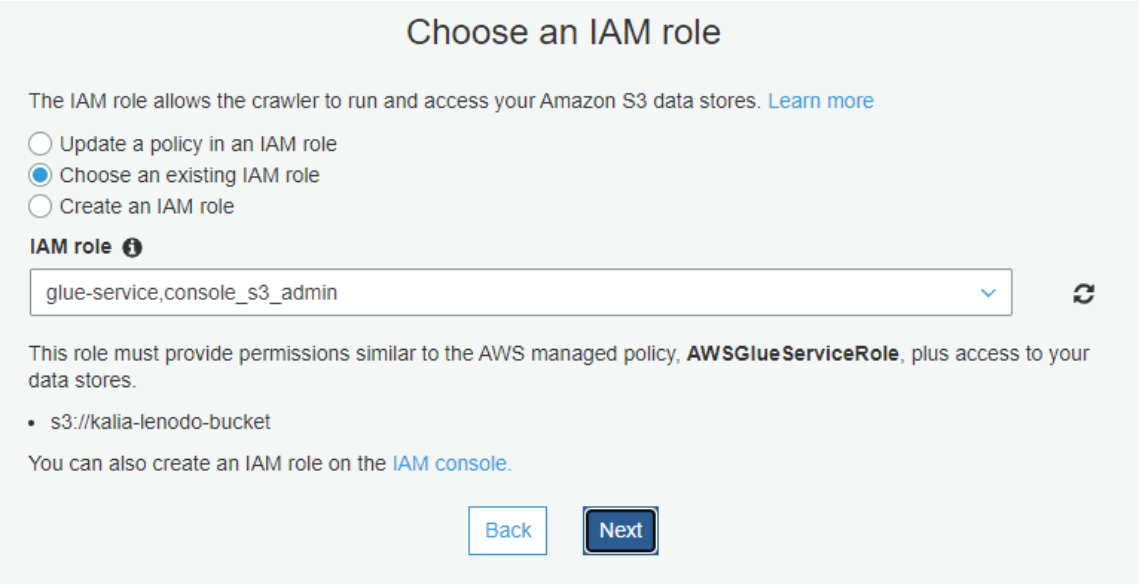
Back Next

ix. As I don't need any other additional data source I am selecting no and click next to proceed

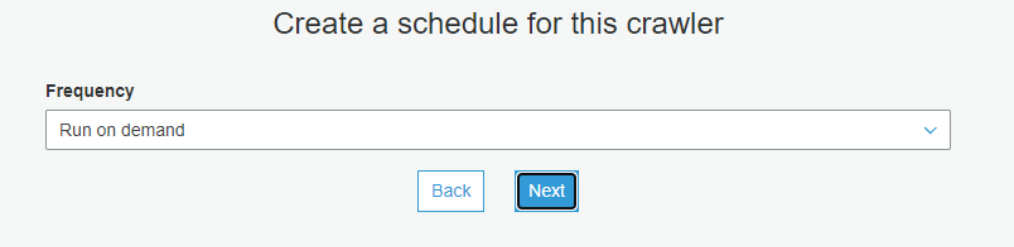


x. Select choose an existing IAM role as we have already created it, select the role and click next

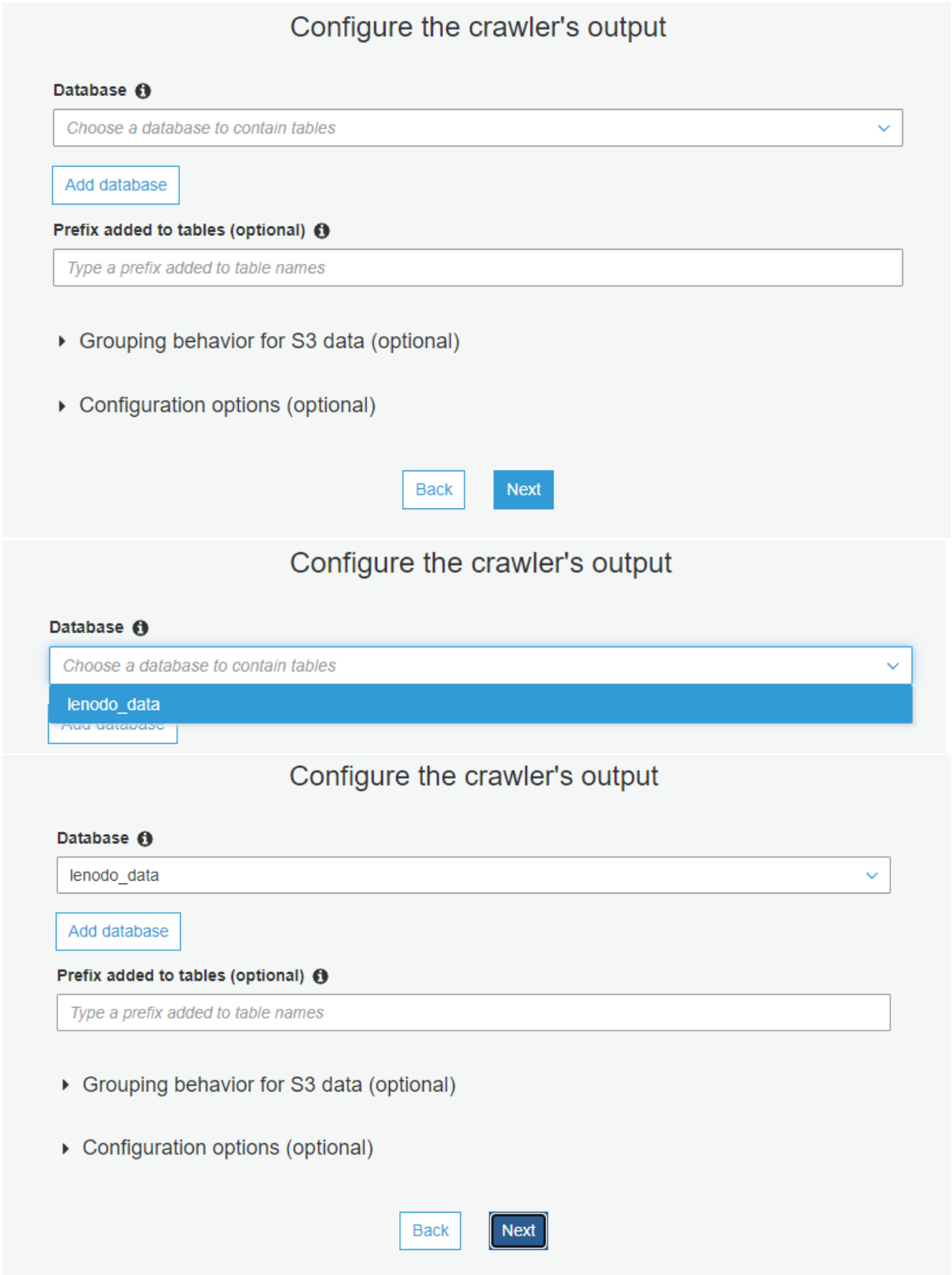




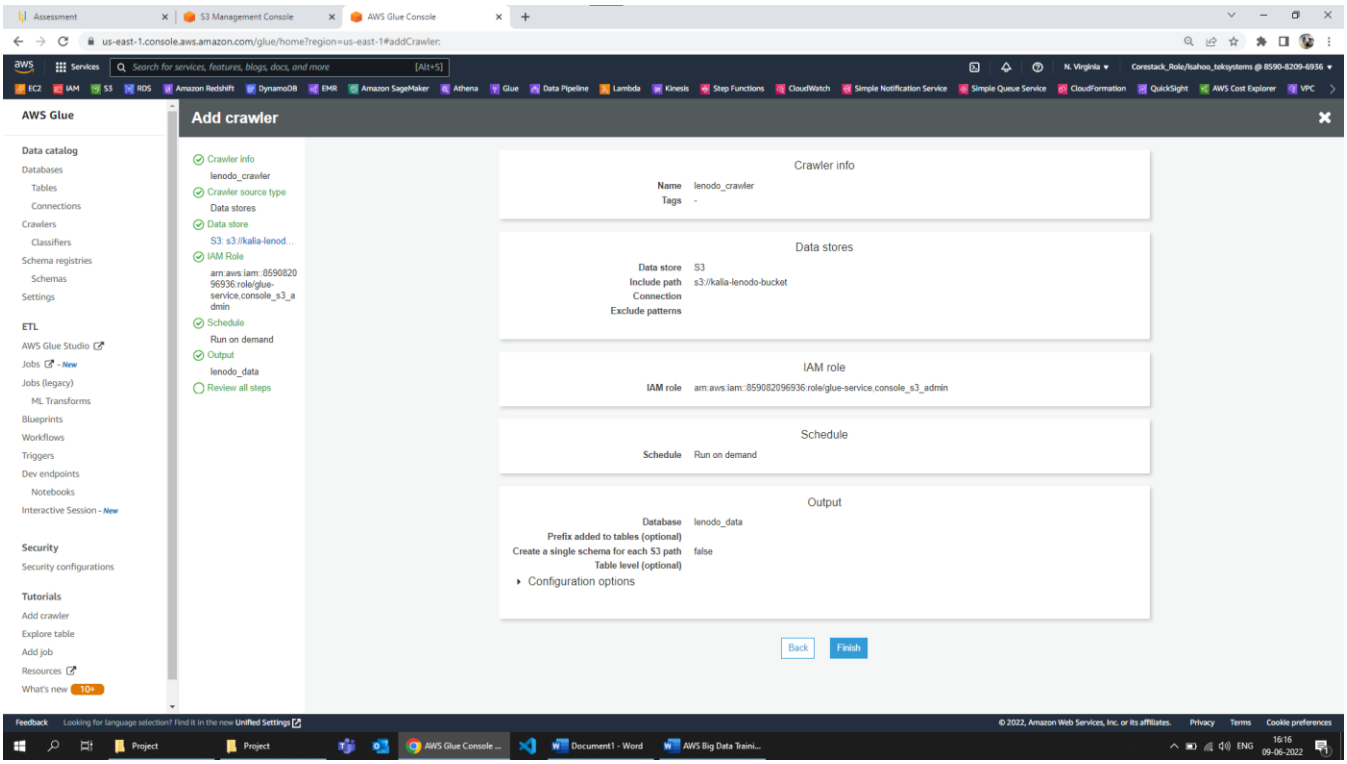
xi. We select Run on demand as frequency and click next

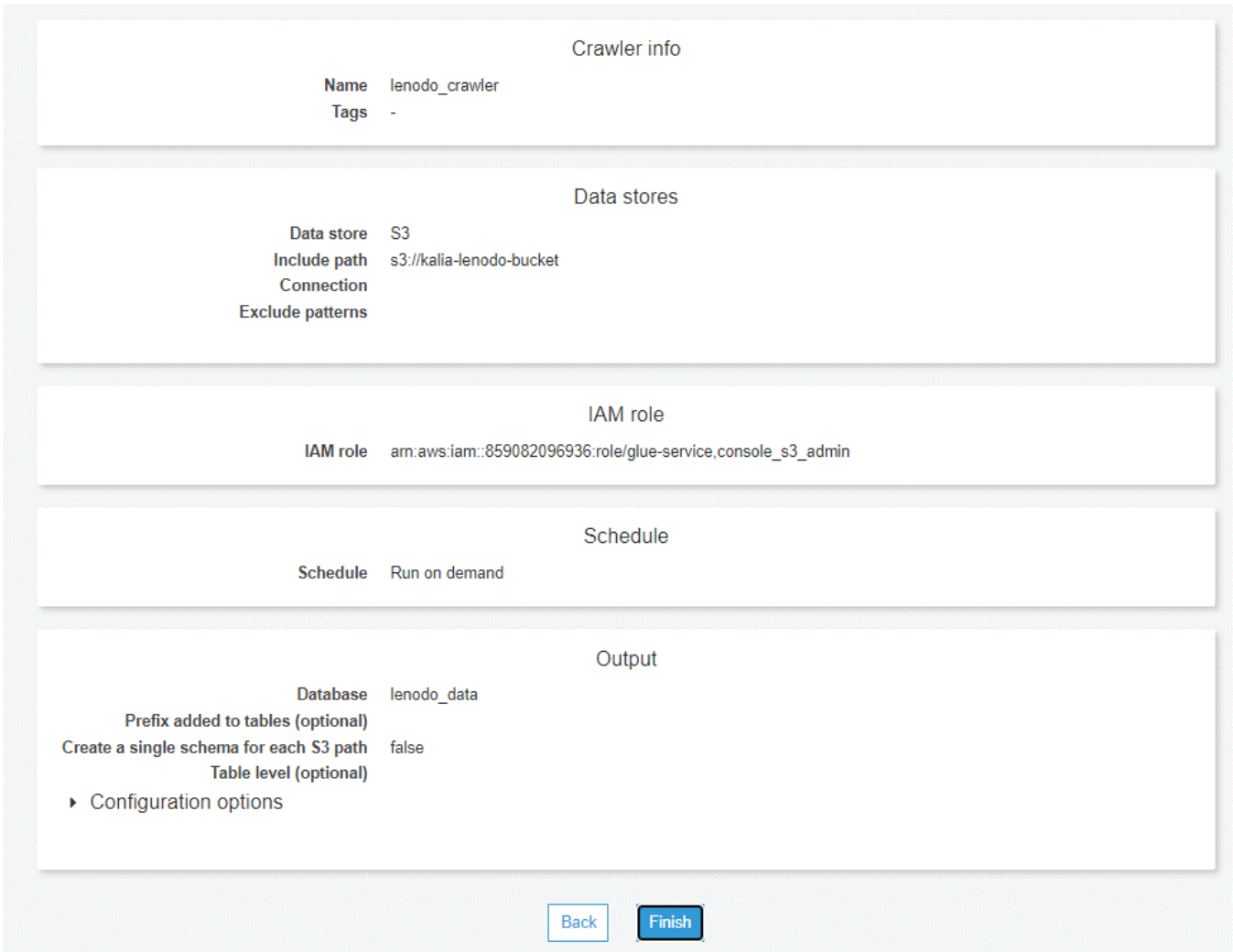


xii. Choose the database that we created earlier for crawler output and click next

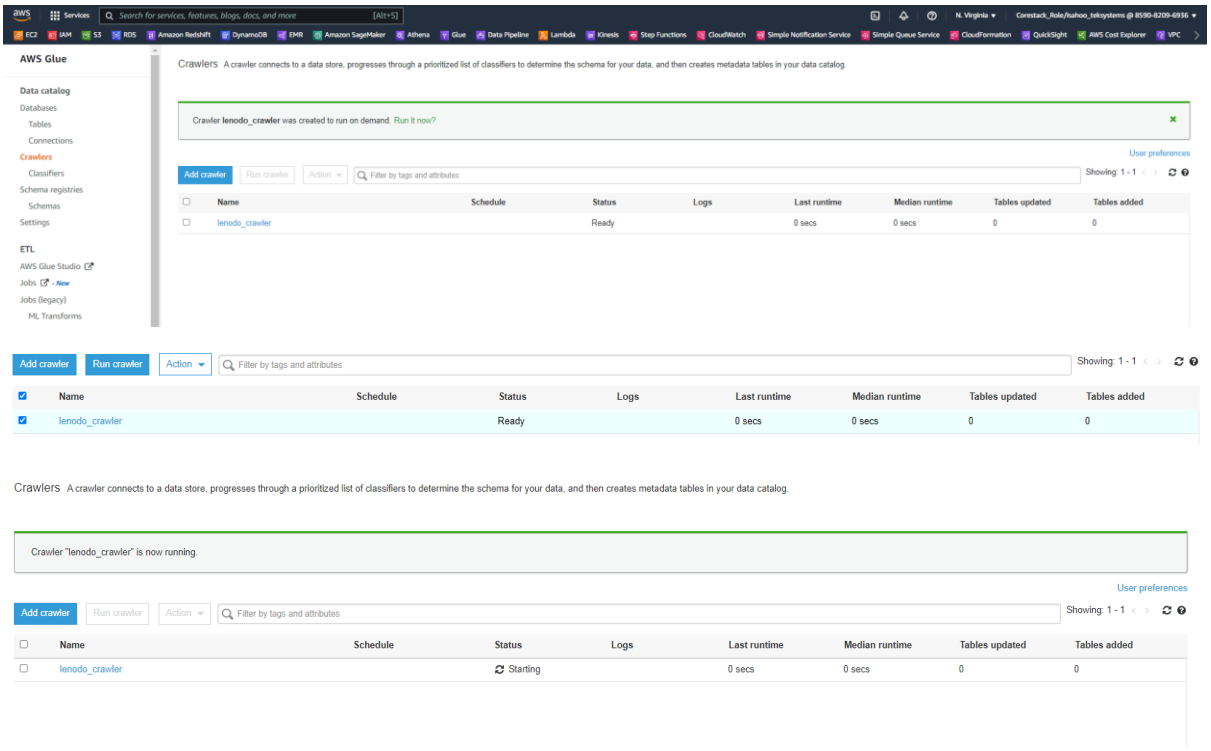


xiii. Click finish in the review step and it will redirect u to crawler dashboard page

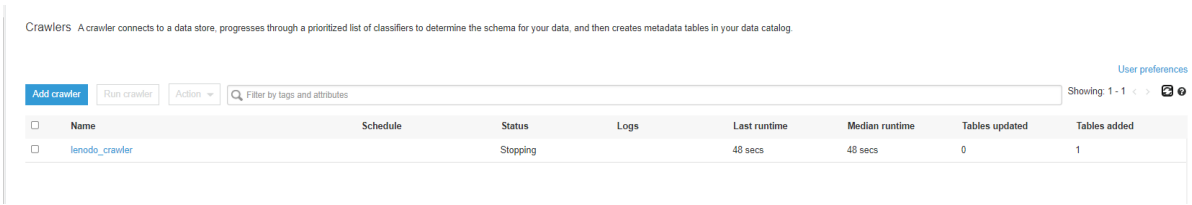


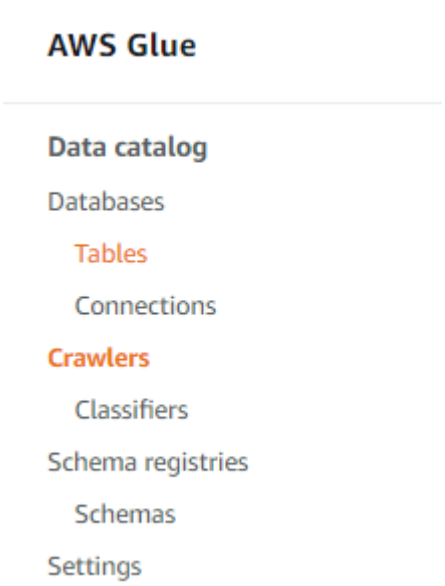


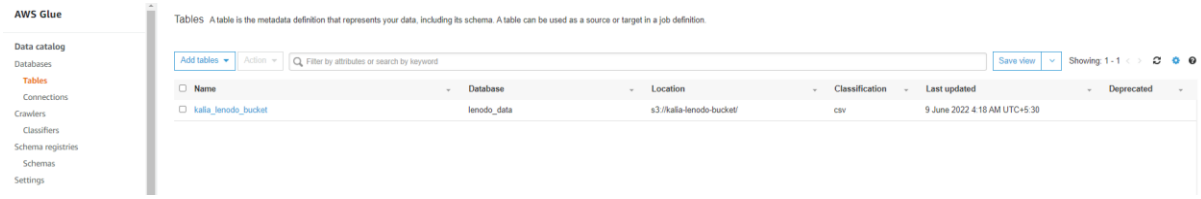
xiv. In the crawler section When the crawler gets created run it



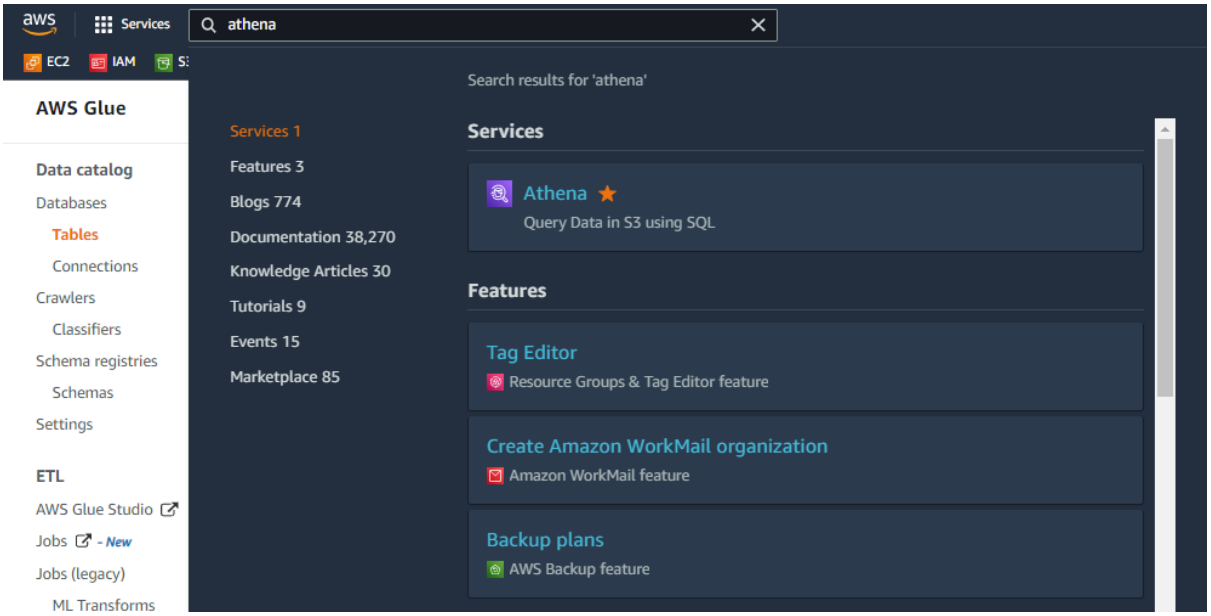
xv. When u see the table has got created u can move to tables section using the left panel and view your table

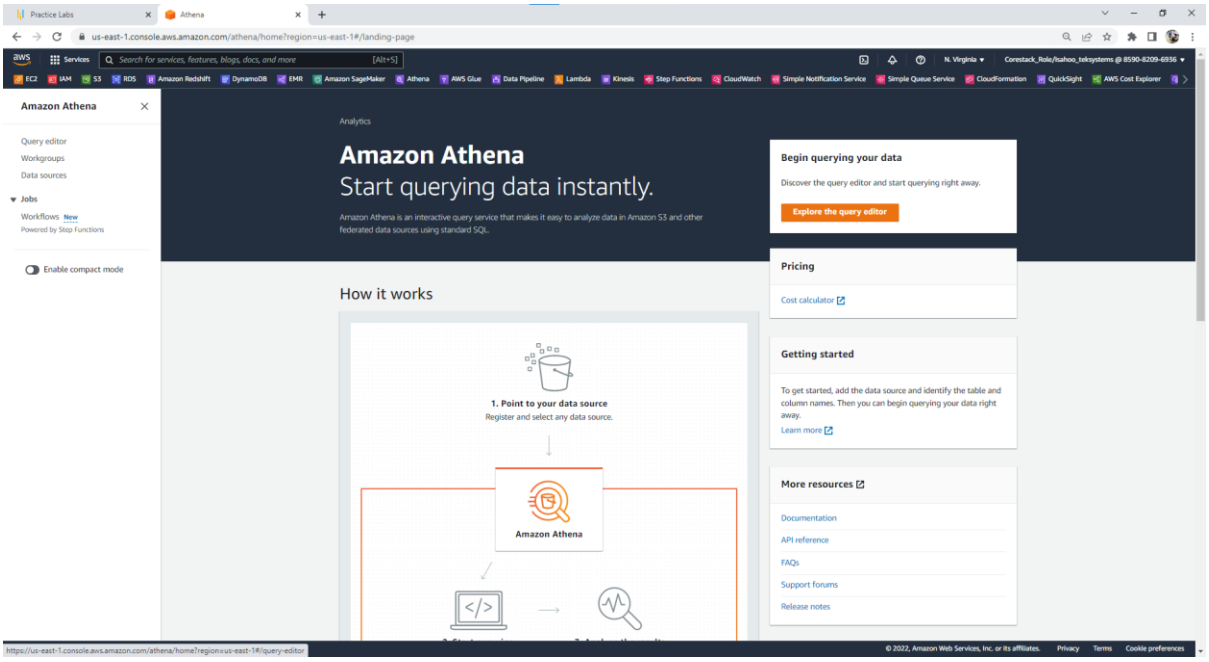


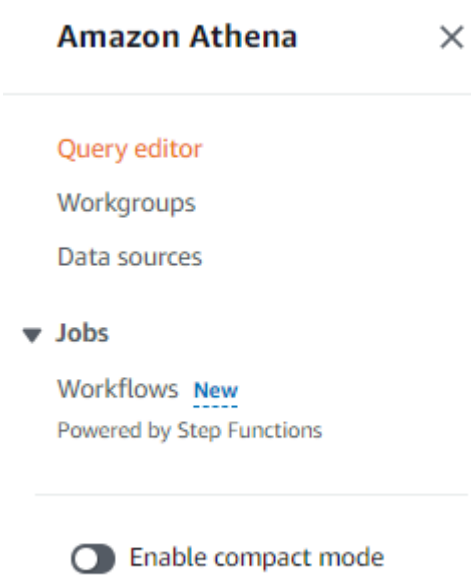




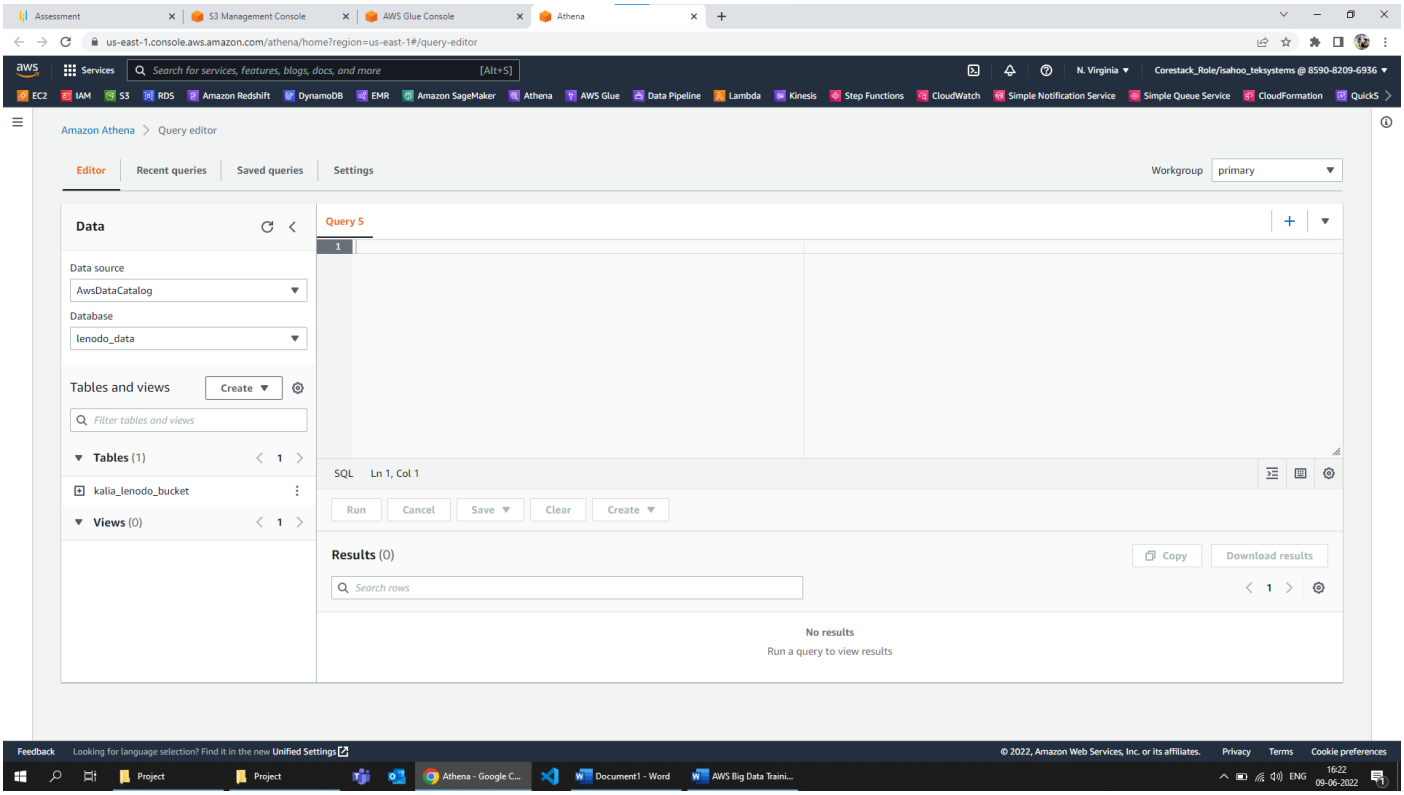
xvi. U can see the table preview using Athena open the search bar and search for Athena and select it choose Query Editor

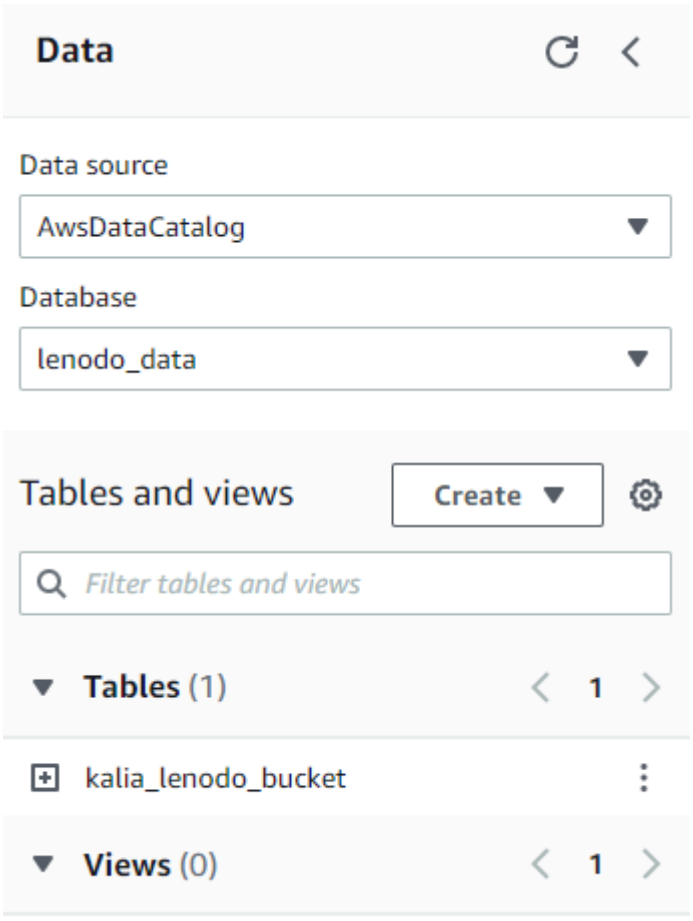


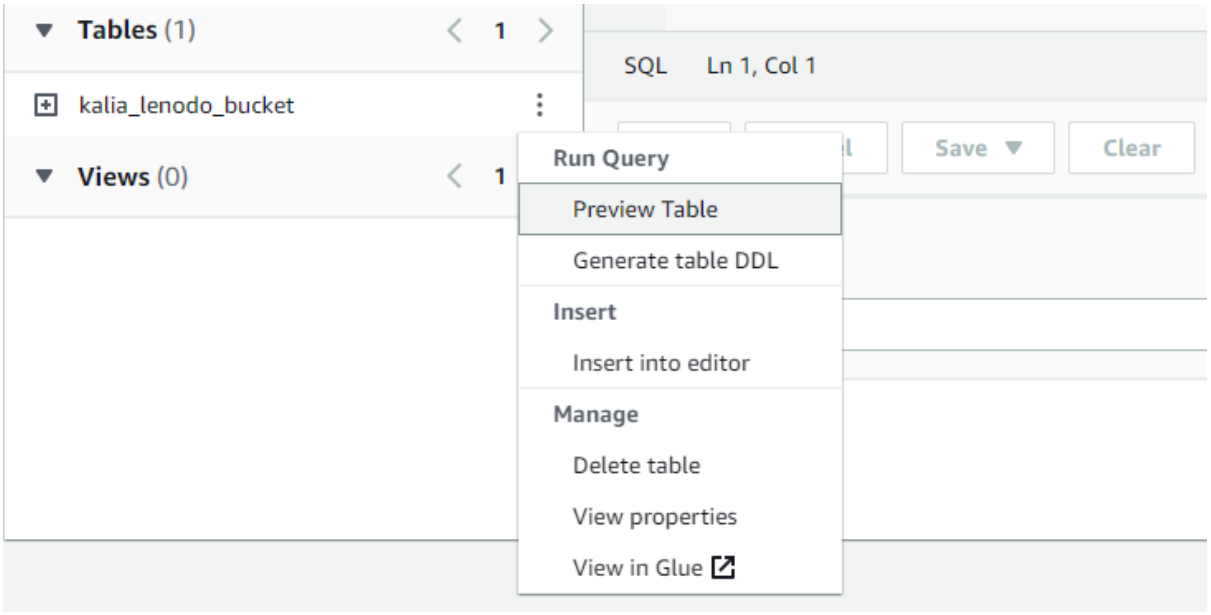




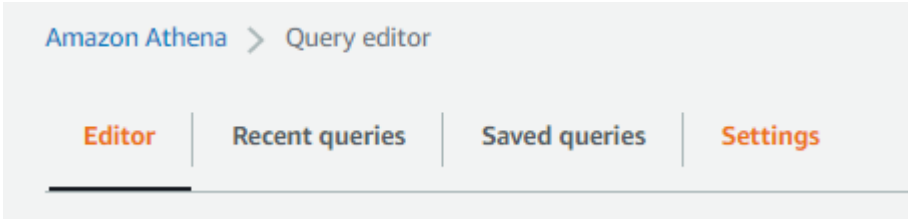
xvii. When the Athena editor will open up u can see that in the left side there is a Data Section we have a Data Source which is opened at AwsDataCatalog and the Database name is the same as the one we created before where our csv data catalogue table was created We can see that our table is present in the Tables list. Click on the 3 dots to the right of the table name and select preview table

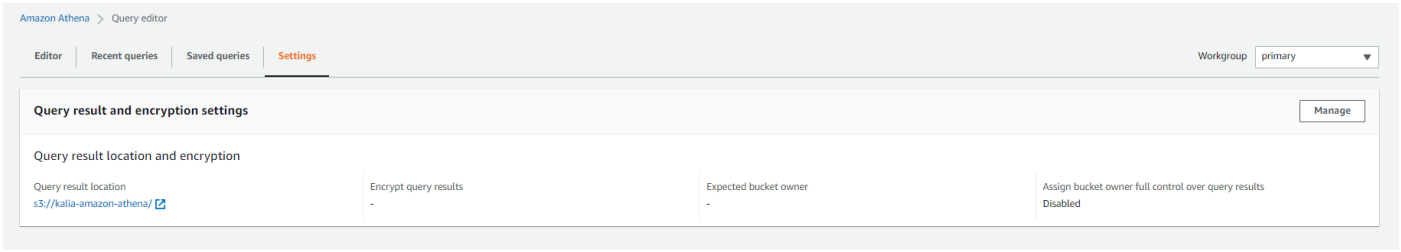


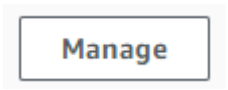




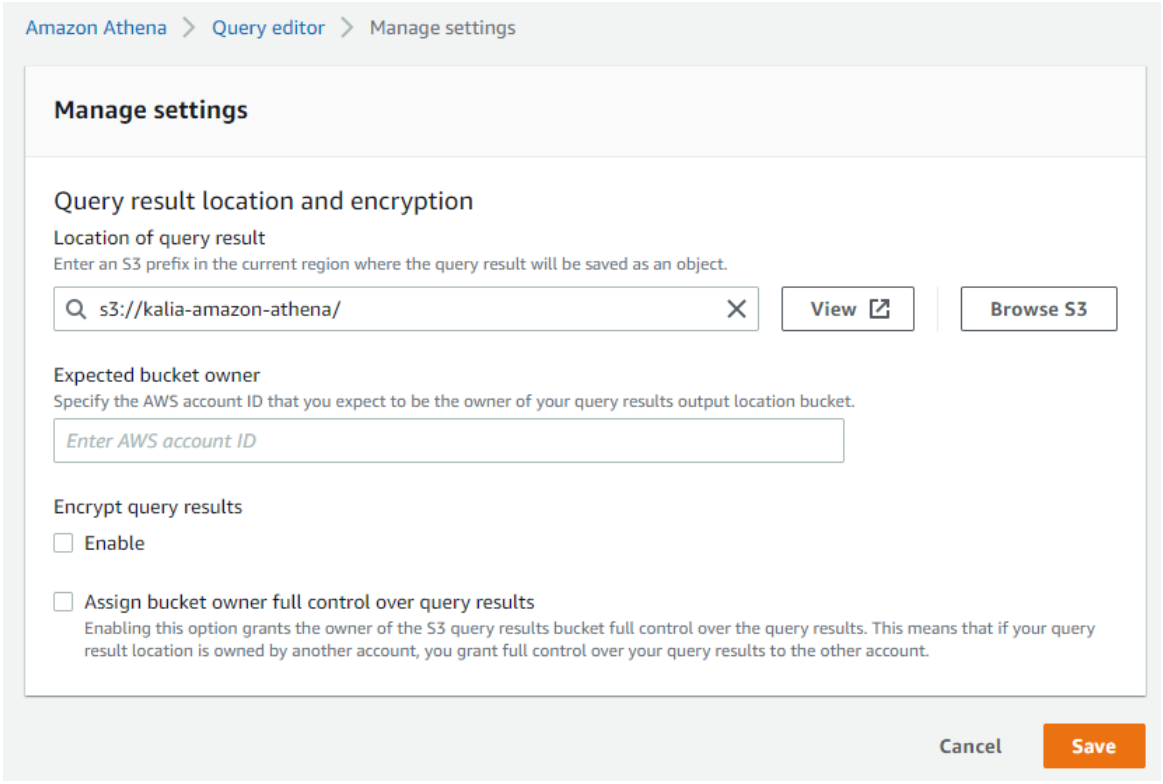
xviii. If u haven't used Athena before the query will load up on the screen but Run button will not be activated. To fix the issue create an Empty S3 bucket for Athena then return to Athena tab and click on settings in the same query editor and select Manage

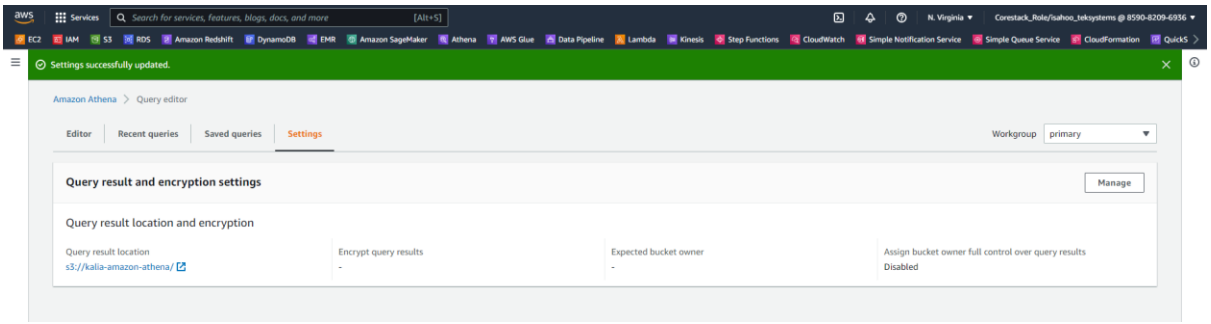




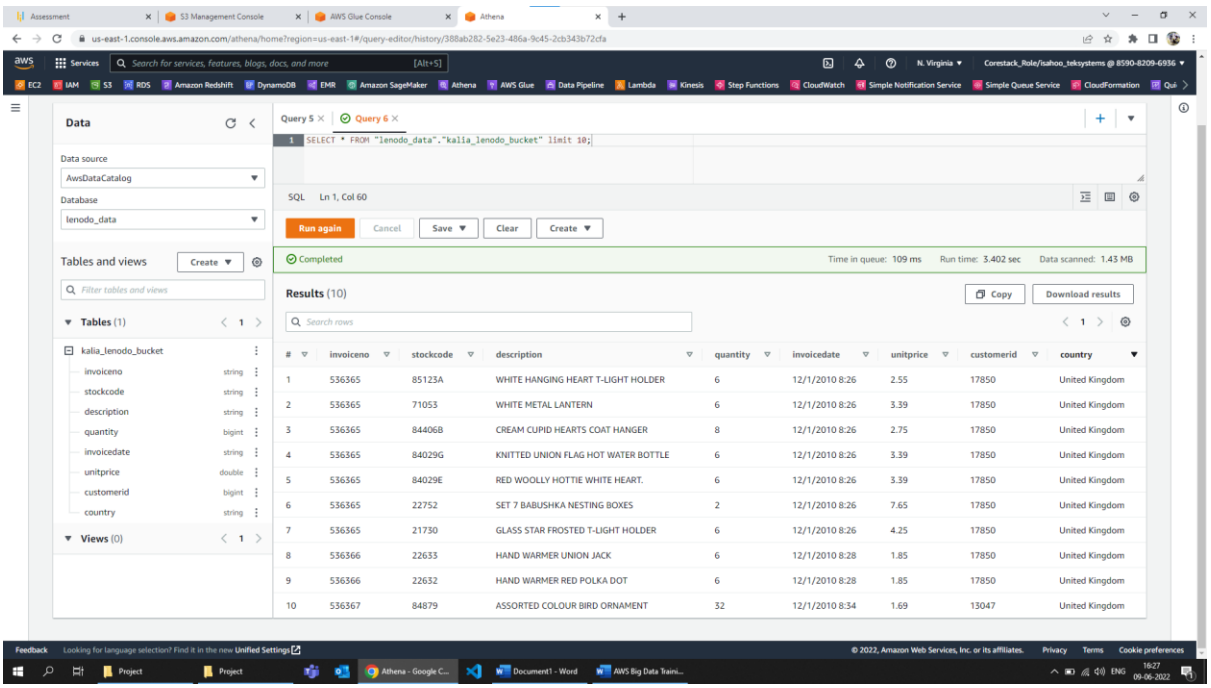


xix. In the query result and location click browse S3 and select the S3 bucket u created for Athena and click save return to the editor tab and now Run button must be activated



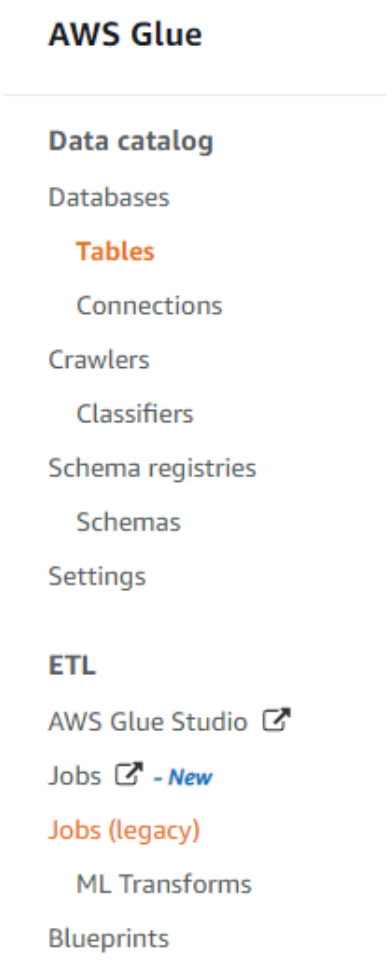


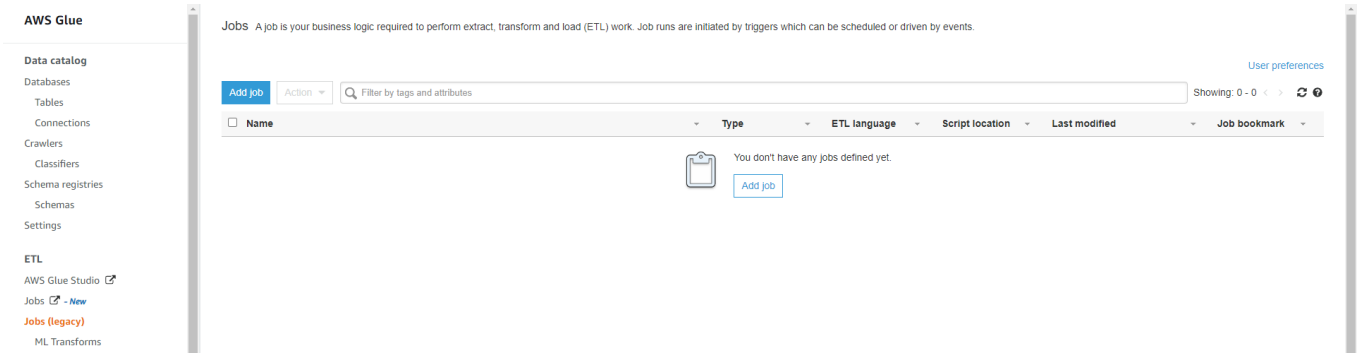
xx. Now u can select the query you want to run and select Run button to run the query

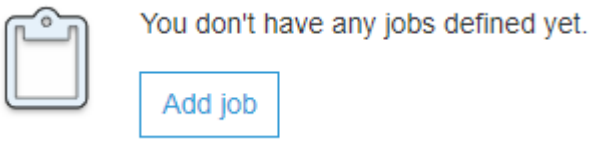


Step 4: Create a Glue job to transform the data into the Parquet format as CSV is not optimal for data warehouse queries

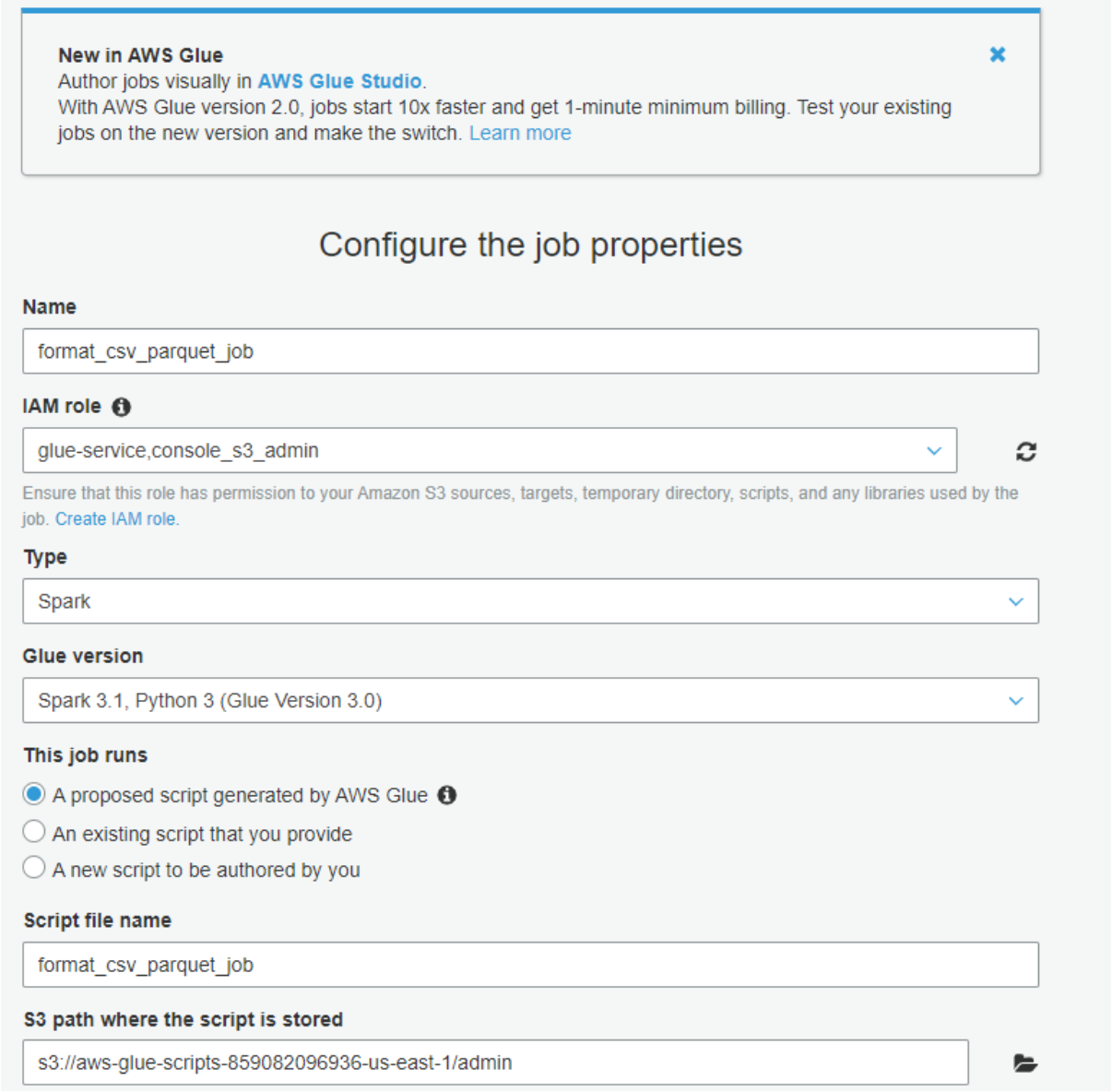
i. Come to the AWS Glue and under the left panel u can see an Option Jobs (legacy) under ETL(Extract Transform Load) subheading select it and click add jobs



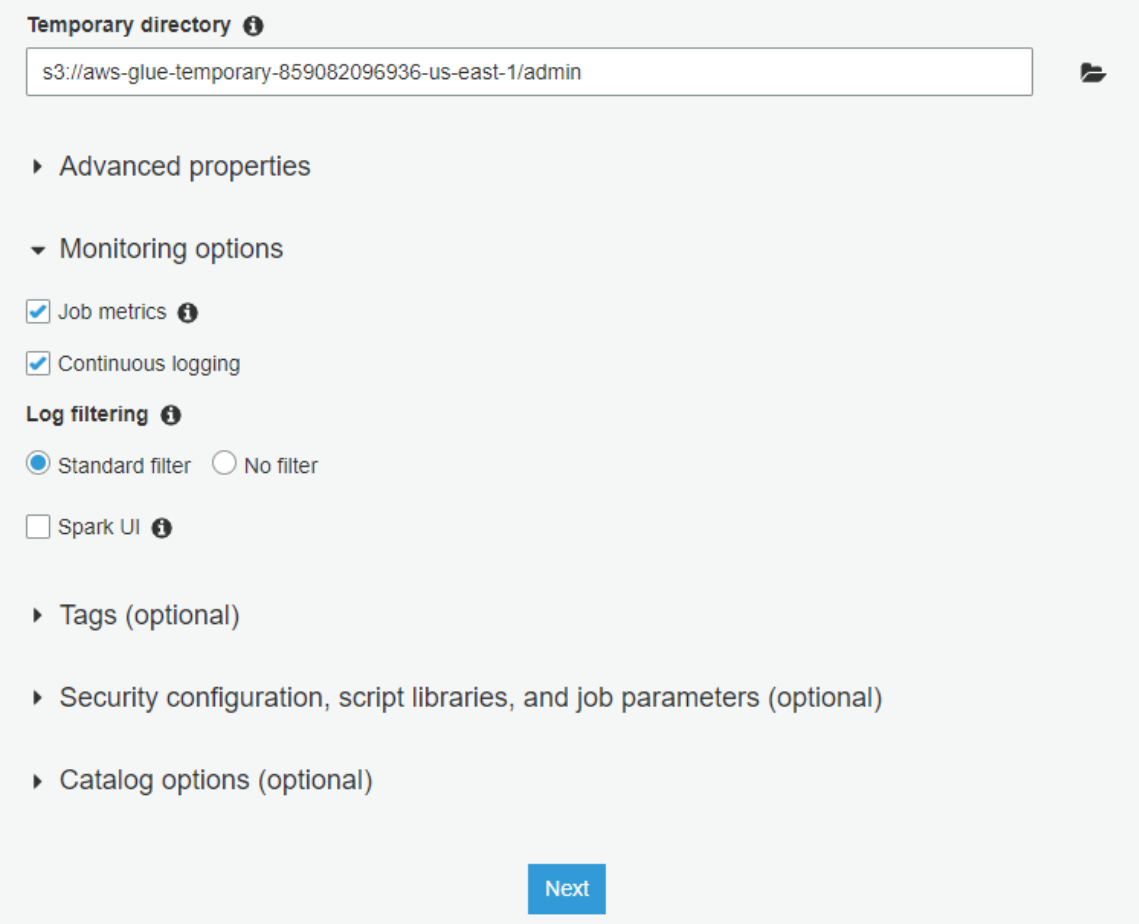




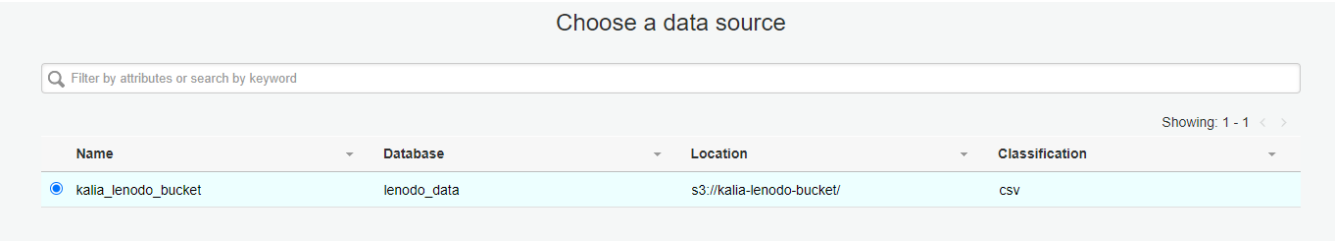
ii. Give name to the job, select the IAM role we created before, select the execution environment u want here we are going to go with type Spark and glue version Spark 3.1, Python 3 (Glue Version 3.0)



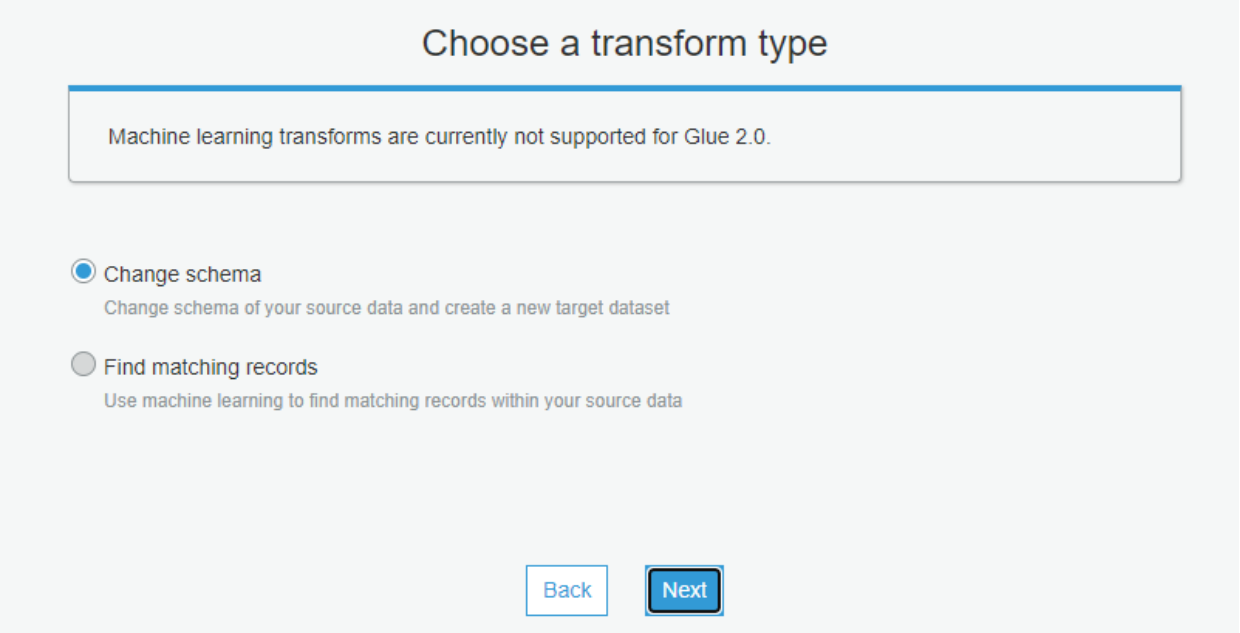
iii. Under monitoring subsection select job metrics and continuous logging and select standard filter and click next



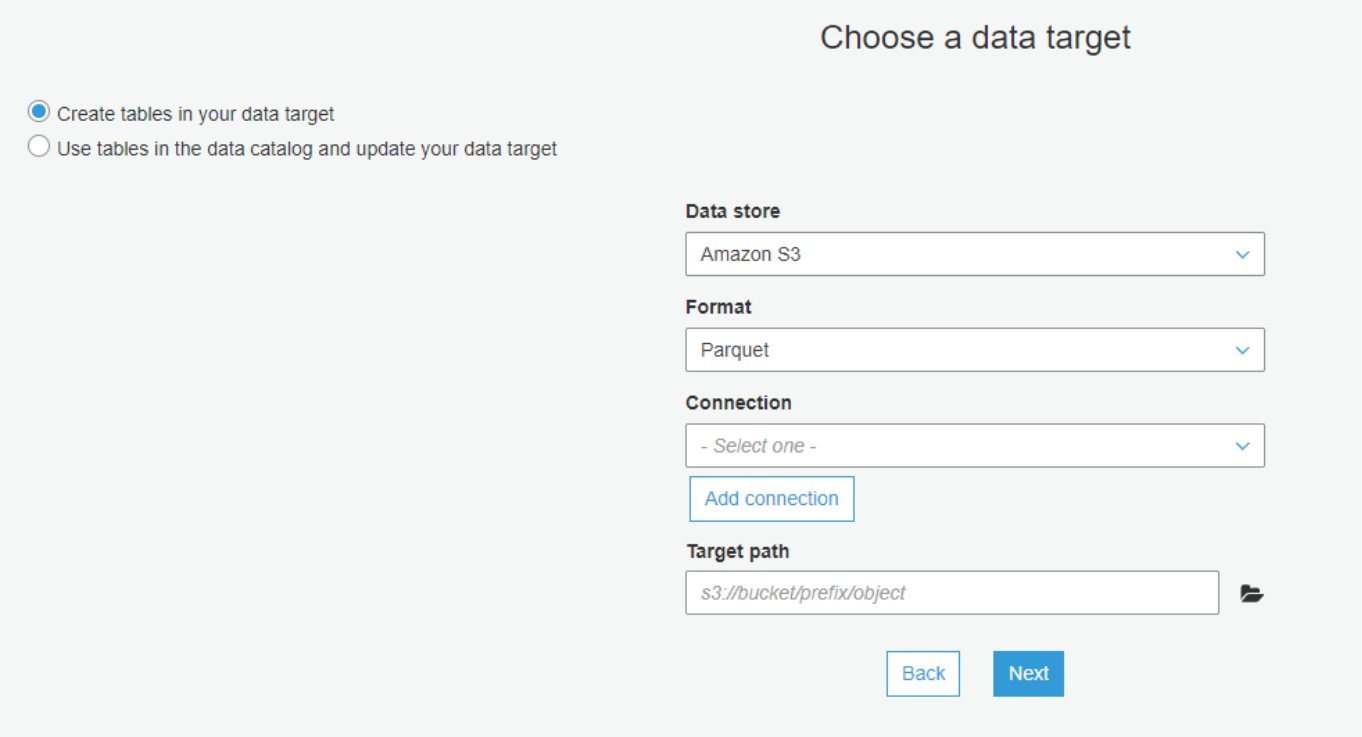
iv. In the data source select the one with the table and database that we want to convert click next



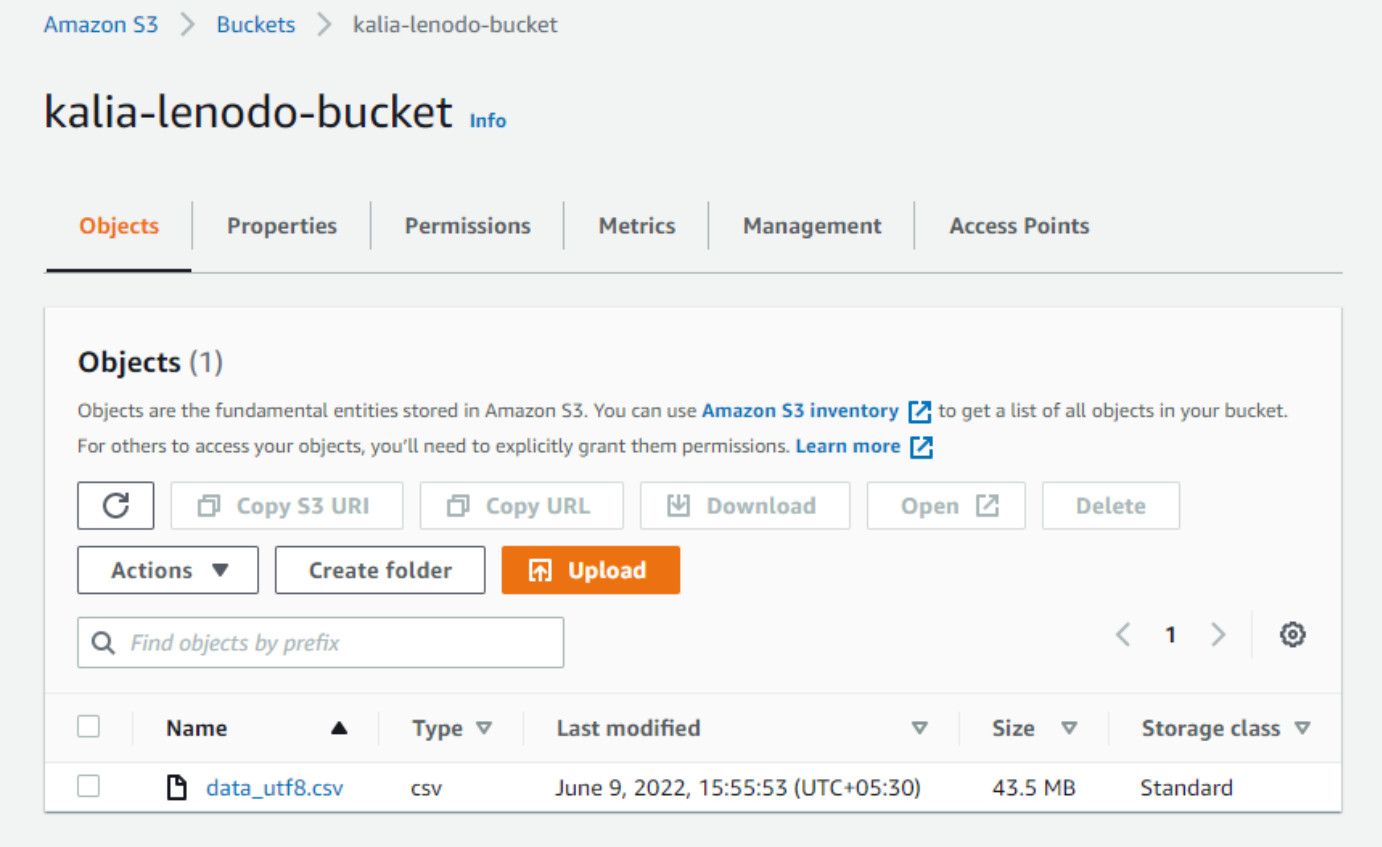
v. In transform type select change schema option and click next

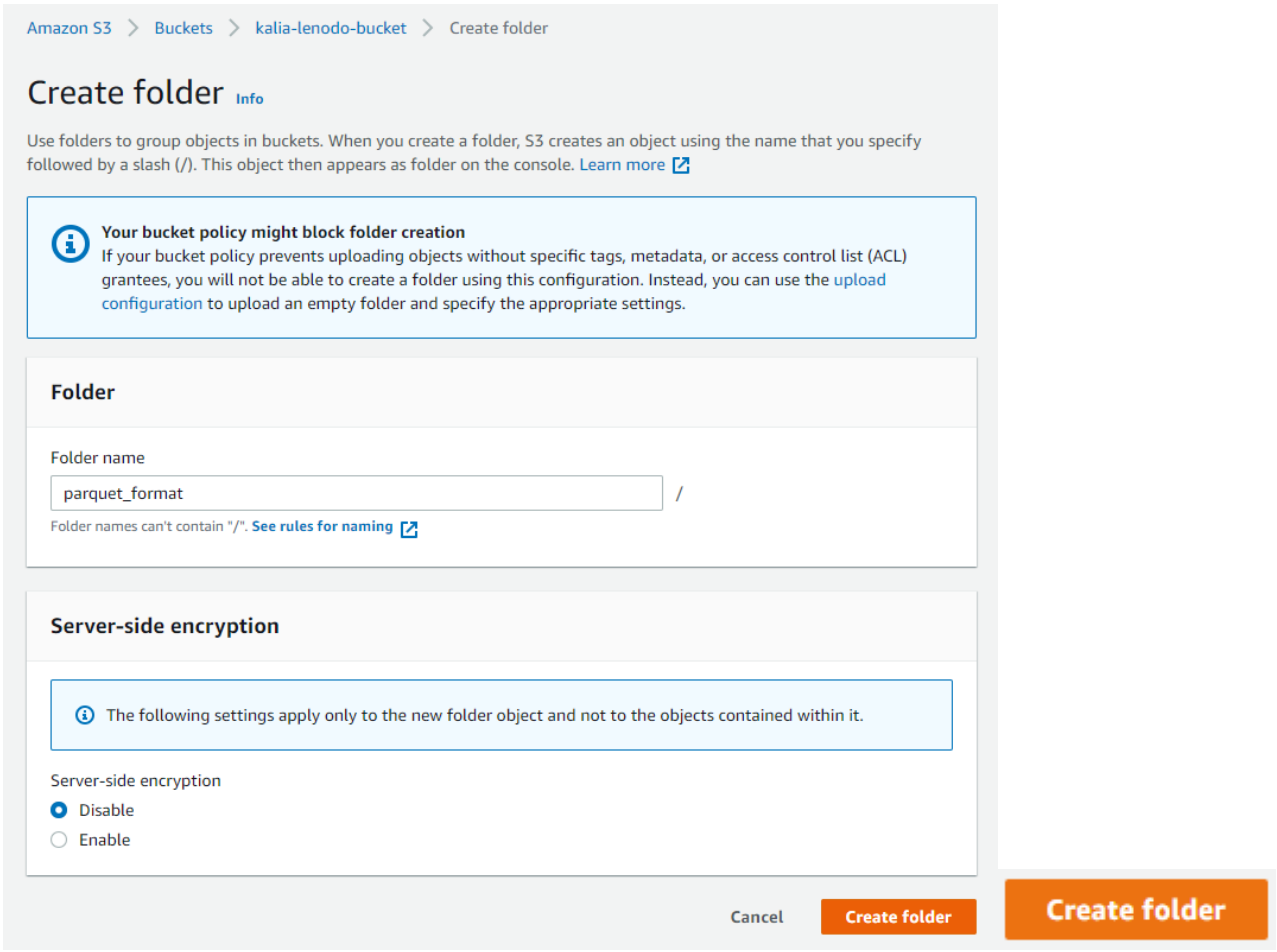


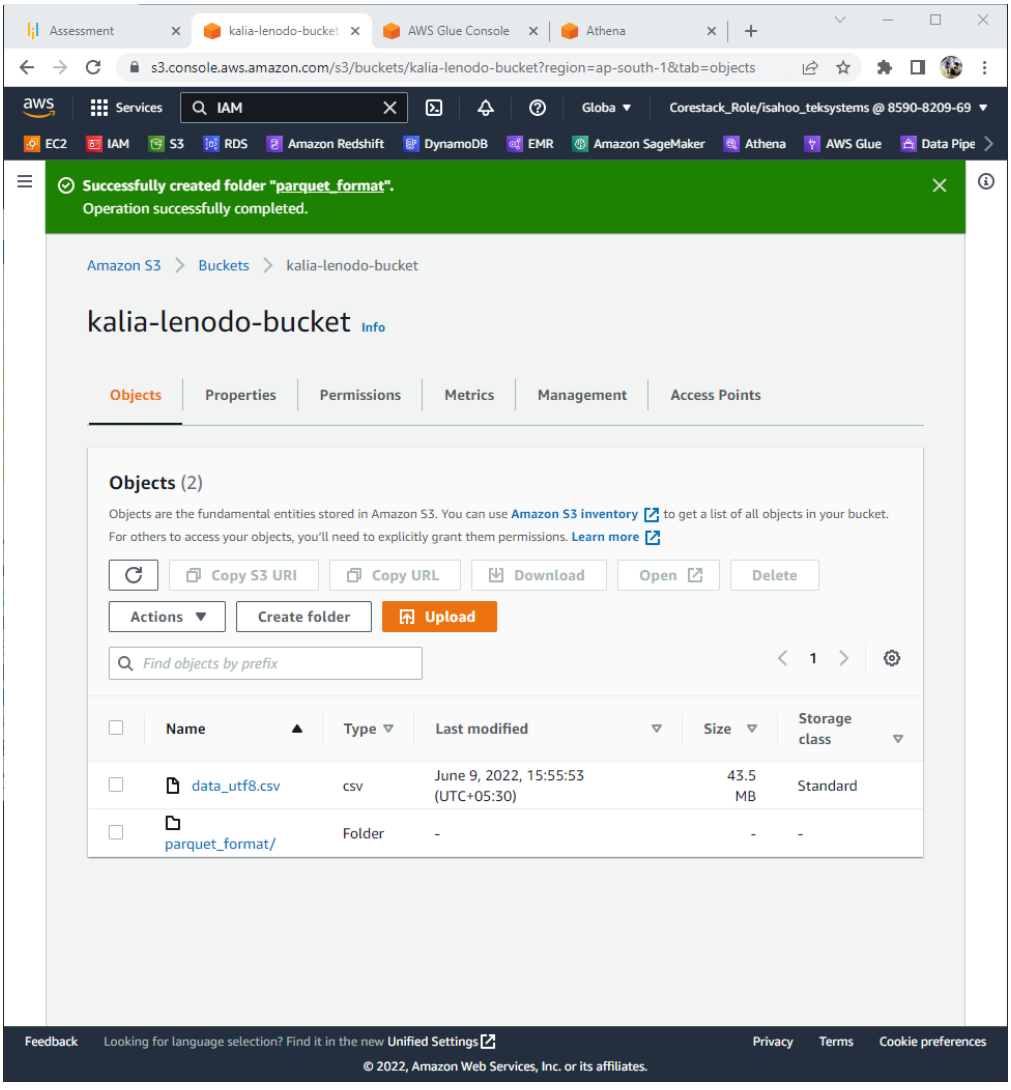
vi. In data target choose Data Source as S3 and format as Paraquet

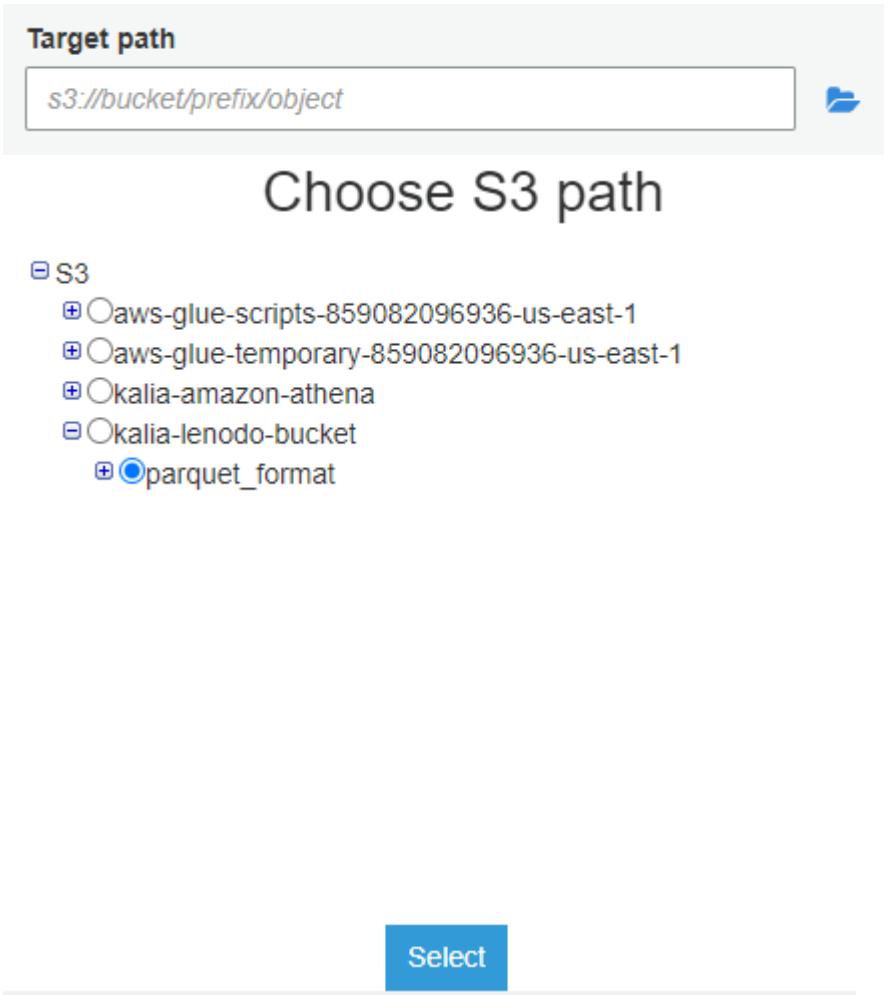


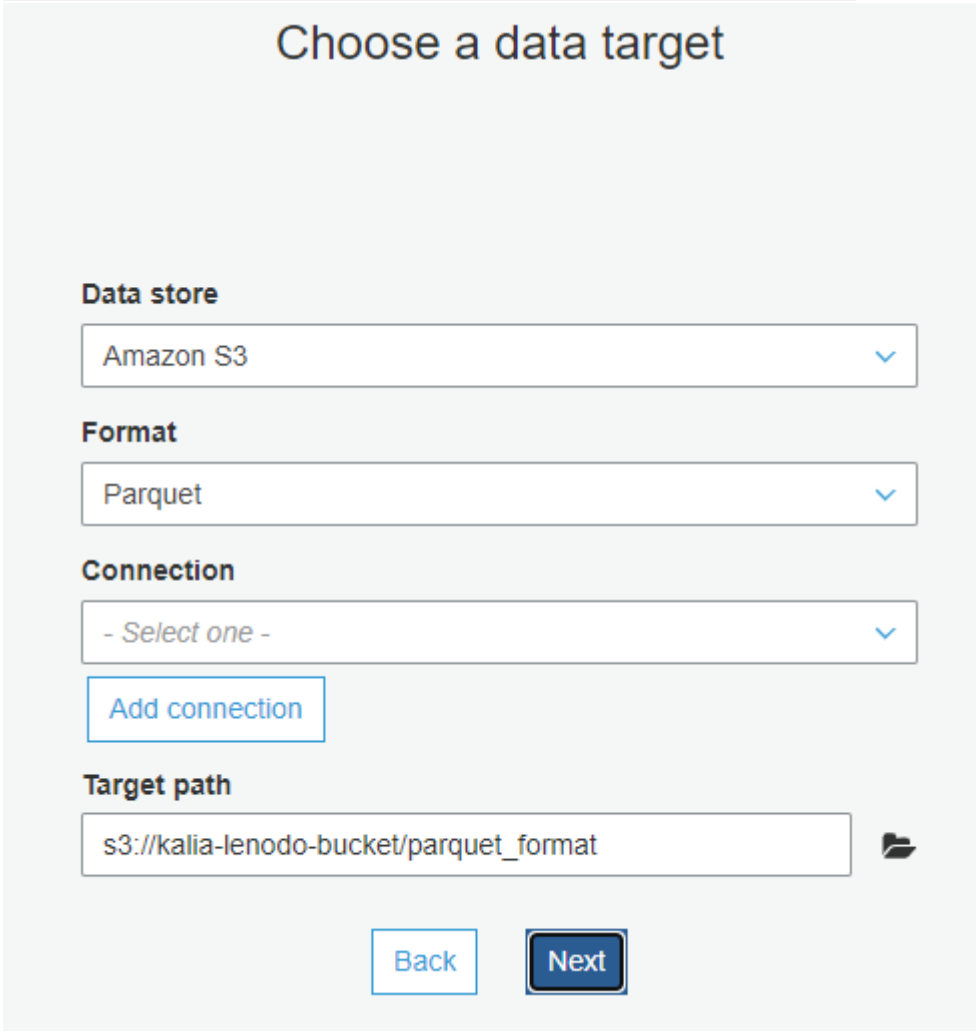
vii. For target path go to your S3 bucket and create a folder come back to glue tab here and update the target path to point to the new folder u created and click next



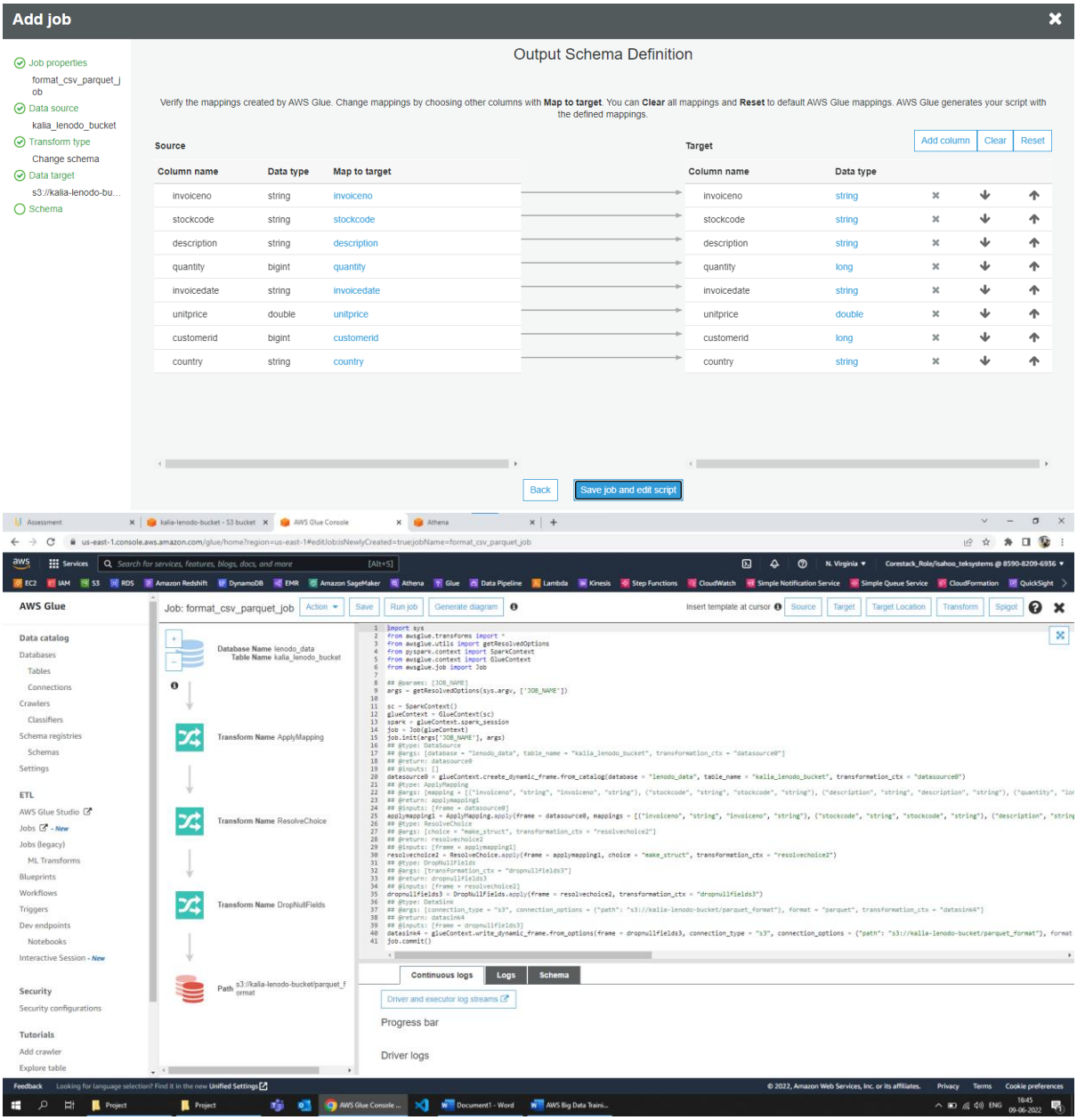

Amazon S3 > Buckets > kalia-lenodo-bucket > Create folder
Create folder Info
Use folders to group objects in buckets. When you create a folder, S3 creates an object using the name that you specify followed by a slash (/). This object then appears as folder on the console. Learn more
Your bucket policy might block folder creation
If your bucket policy prevents uploading objects without specific tags, metadata, or access control list (ACL) grantees, you will not be able to create a folder using this configuration. Instead, you can use the upload configuration to upload an empty folder and specify the appropriate settings.
Folder
Folder name
parquet_format
/
Folder names can't contain "/". See rules for naming
Server-side encryption
The following settings apply only to the new folder object and not to the objects contained within it.
Server-side encryption
Disable
Enable
Cancel
Create folder
Create folder


Successfully created folder "parquet_format".
Operation successfully completed.
Amazon S3 > Buckets > kalia-lenodo-bucket
kalia-lenodo-bucket Info
Objects
Properties
Permissions
Metrics
Management
Access Points
Objects (2)
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket.
For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix
Name
Type
Last modified
Size
Storage class
data_utf8.csv
csv
June 9, 2022, 15:55:53 (UTC+05:30)
43.5 MB
Standard
parquet_format/
Folder
-
-
-
Feedback
Looking for language selection? Find it in the new Unified Settings
Privacy
Terms
Cookie preferences
© 2022, Amazon Web Services, Inc. or its affiliates.


Target path
s3://bucket/prefix/object
Choose S3 path
S3
aws-glue-scripts-859082096936-us-east-1
aws-glue-temporary-859082096936-us-east-1
kalia-amazon-athena
kalia-lenodo-bucket
parquet_format
Select


Choose a data target
Data store
Amazon S3
Format
Parquet
Connection
- Select one -
Add connection
Target path
s3://kalia-lenodo-bucket/parquet_format
Back
Next

viii. You can see the conversion here click Save Job and Edit Script which will open up the code

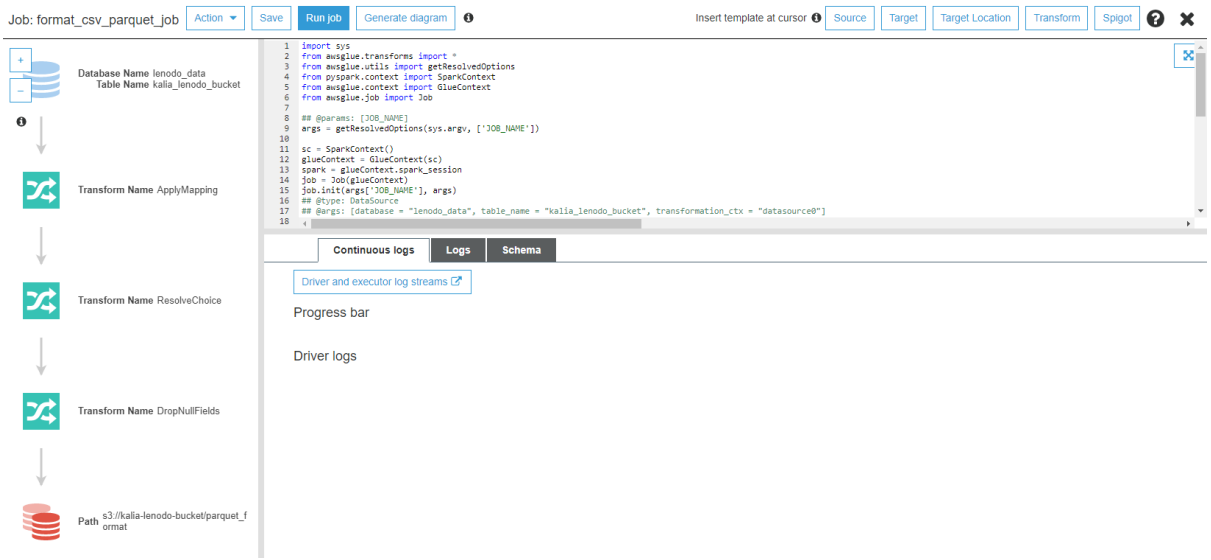


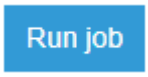
Code (Auto Generated) :

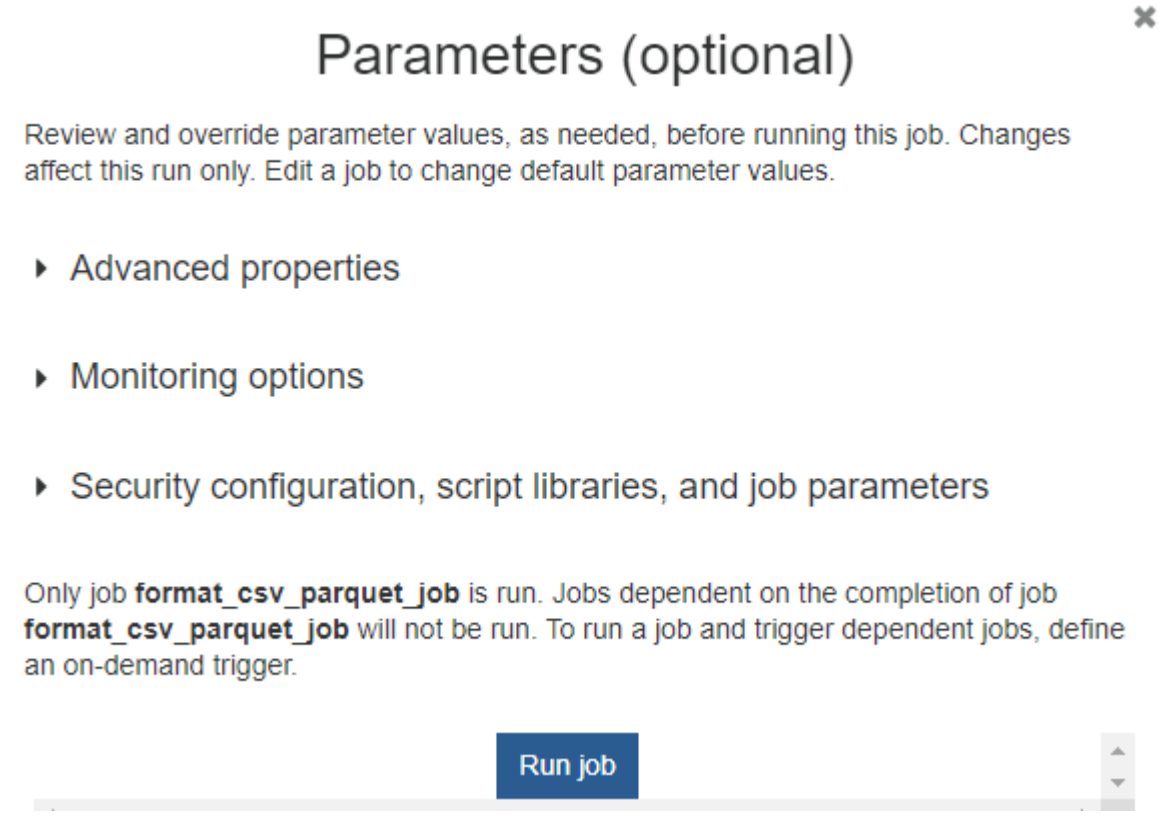
```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)
```

```
## @type: DataSource
## @args: [database = "lenodo_data", table_name = "kalia_lenodo_bucket",
transformation_ctx = "datasource0"]
## @return: datasource0
## @inputs: []
datasource0 = glueContext.create_dynamic_frame.from_catalog(database = "lenodo_data",
table_name = "kalia_lenodo_bucket", transformation_ctx = "datasource0")
## @type: ApplyMapping
## @args: [mapping = [("invoiceno", "string", "invoiceno", "string"), ("stockcode",
"string", "stockcode", "string"), ("description", "string", "description", "string"),
("quantity", "long", "quantity", "long"), ("invoicedate", "string", "invoicedate",
"string"), ("unitprice", "double", "unitprice", "double"), ("customerid", "long",
"customerid", "long"), ("country", "string", "country", "string")], transformation_ctx =
"applymapping1"]
## @return: applymapping1
## @inputs: [frame = datasource0]
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [("invoiceno",
"string", "invoiceno", "string"), ("stockcode", "string", "stockcode", "string"),
("description", "string", "description", "string"), ("quantity", "long", "quantity",
"long"), ("invoicedate", "string", "invoicedate", "string"), ("unitprice", "double",
"unitprice", "double"), ("customerid", "long", "customerid", "long"), ("country",
"string", "country", "string")], transformation_ctx = "applymapping1")
## @type: ResolveChoice
## @args: [choice = "make_struct", transformation_ctx = "resolvechoice2"]
## @return: resolvechoice2
## @inputs: [frame = applymapping1]
resolvechoice2 = ResolveChoice.apply(frame = applymapping1, choice = "make_struct",
transformation_ctx = "resolvechoice2")
## @type: DropNullFields
## @args: [transformation_ctx = "dropnullfields3"]
## @return: dropnullfields3
## @inputs: [frame = resolvechoice2]
dropnullfields3 = DropNullFields.apply(frame = resolvechoice2, transformation_ctx =
"dropnullfields3")
## @type: DataSink
## @args: [connection_type = "s3", connection_options = {"path": "s3://kalia-lenodo-
bucket/parquet_format"}, format = "parquet", transformation_ctx = "datasink4"]
## @return: datasink4
## @inputs: [frame = dropnullfields3]
datasink4 = glueContext.write_dynamic_frame.from_options(frame = dropnullfields3,
connection_type = "s3", connection_options = {"path": "s3://kalia-lenodo-
bucket/parquet_format"}, format = "parquet", transformation_ctx = "datasink4")
job.commit()
```

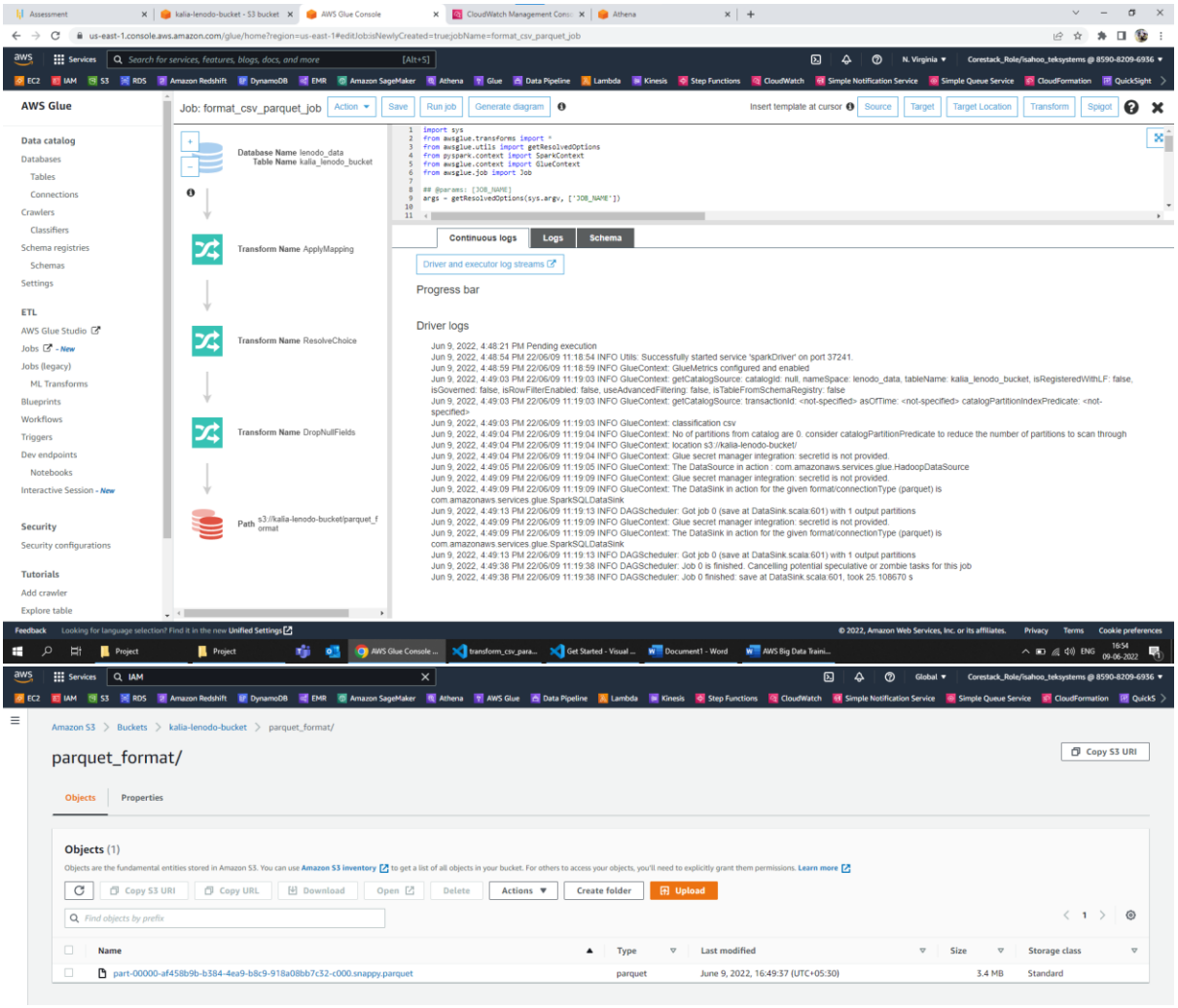
ix. Click Run Job → Run Job to run







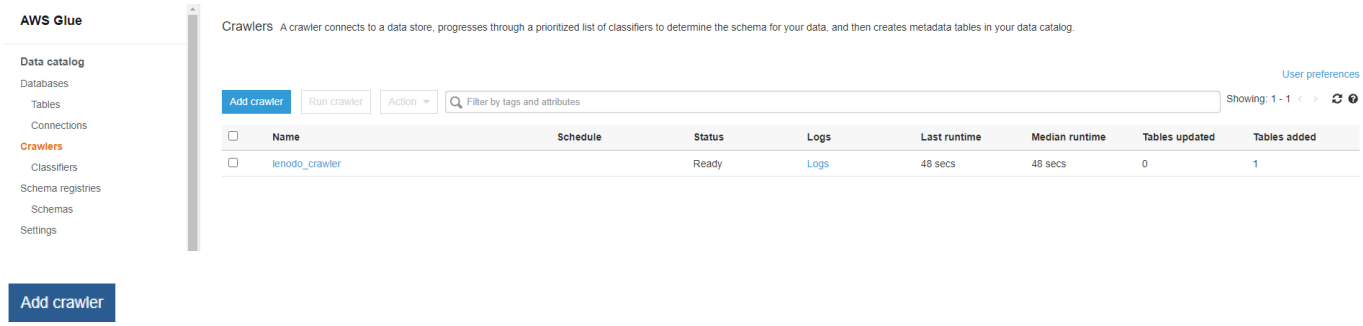
x. You can keep track of the driver logs when job is finished u can check the S3 bucket folder to see if the Paraquet format file was successfully generated

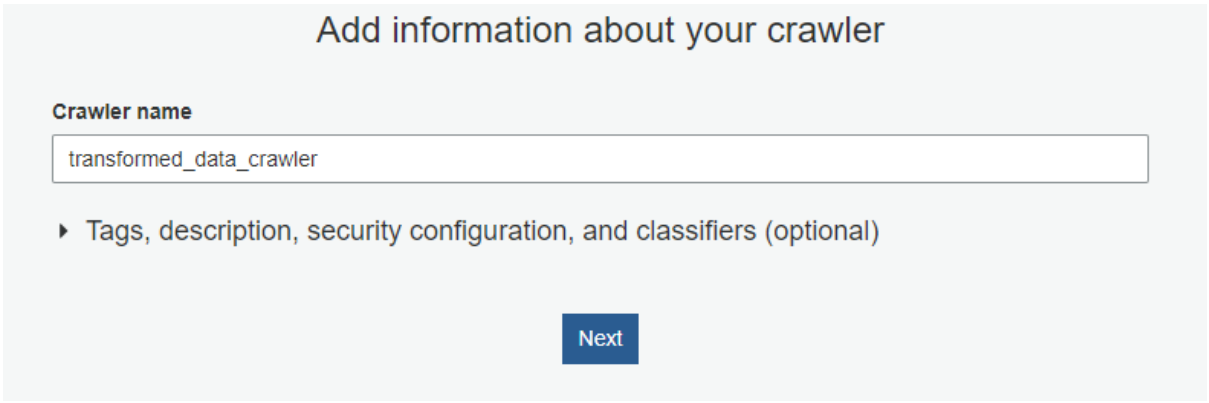


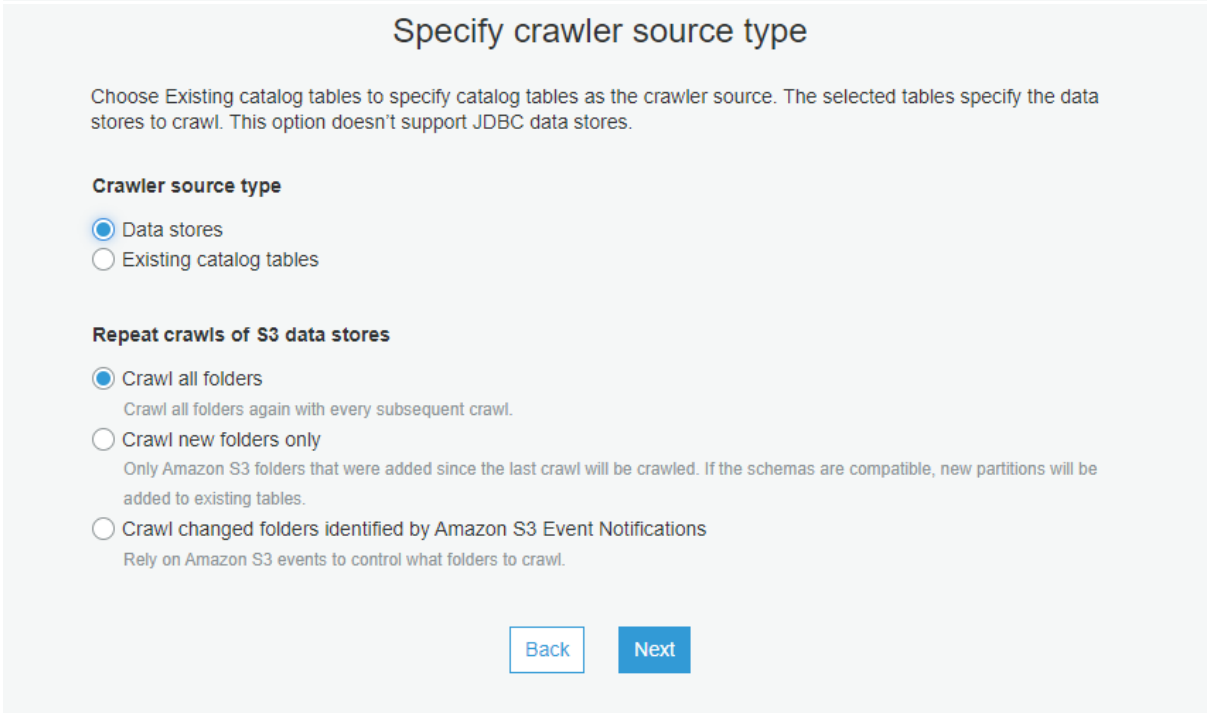
As u can see the data is generated successfully here

Step 5: Add another crawler to crawl the Parquet data files to generate the metadata catalog of the Parquet file in order to query it with Athena

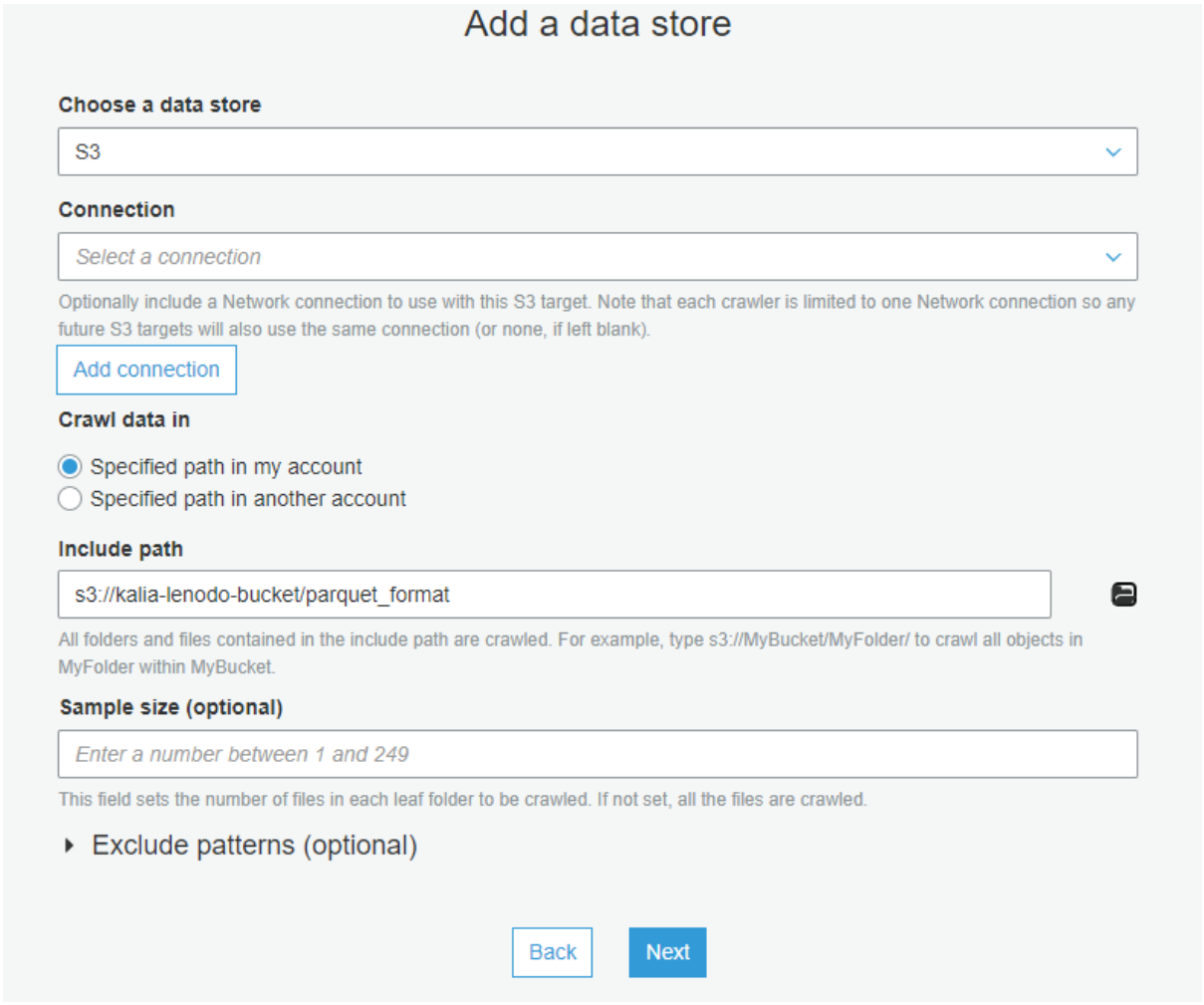
i. Open the Crawler section and add a crawler in the exact same way we did before

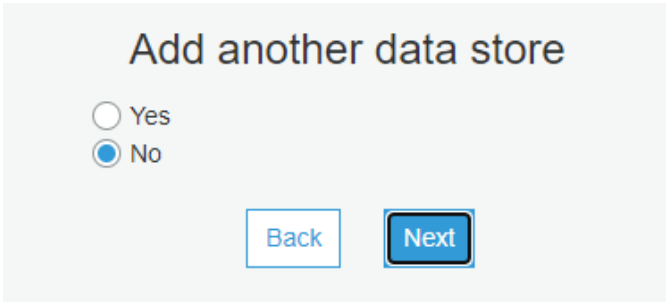


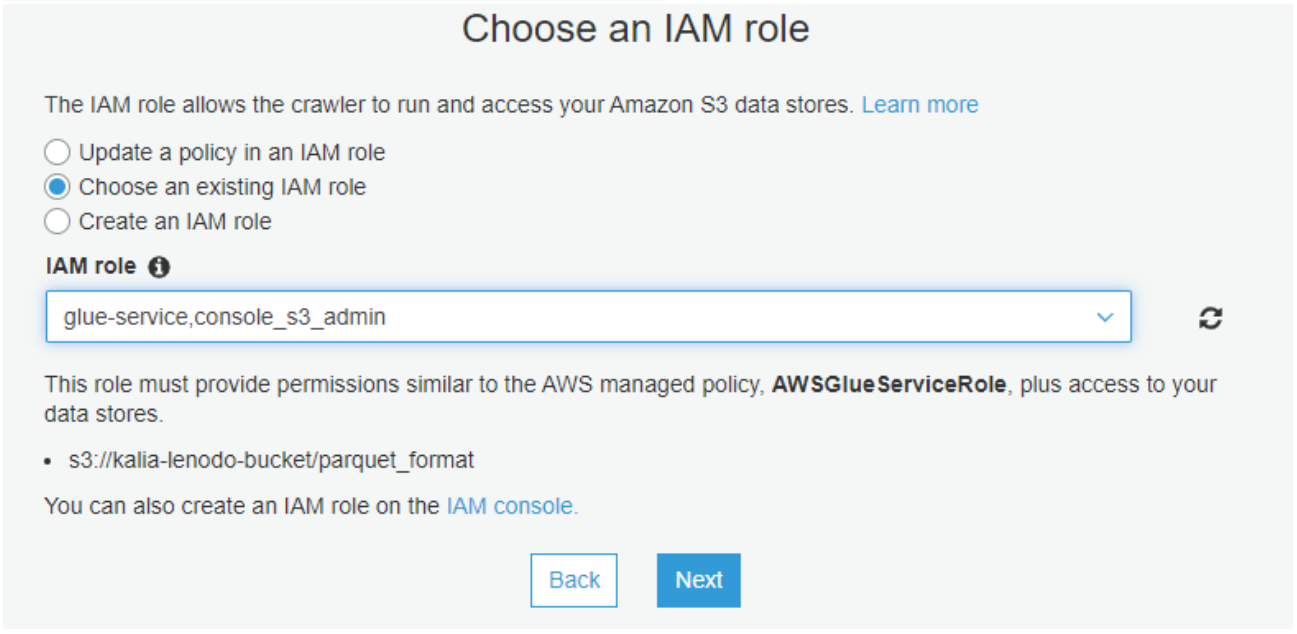


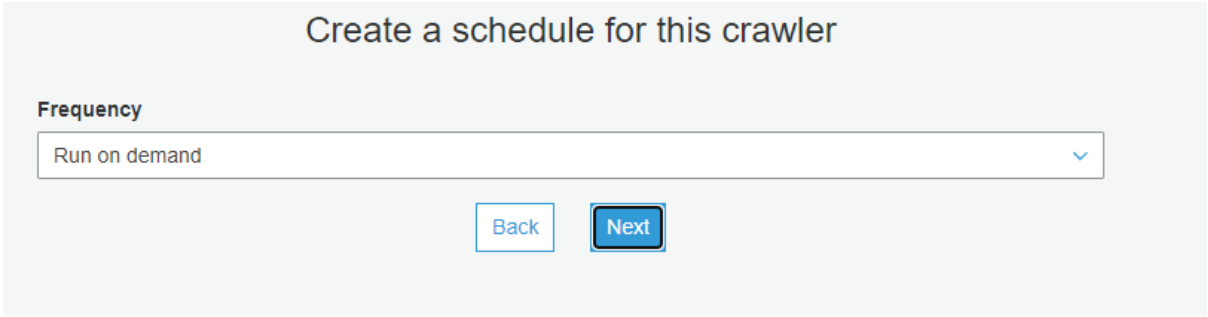


ii. Make sure u select the correct path in the Data Source it must point to the folder where u added the Parquet file using the Job

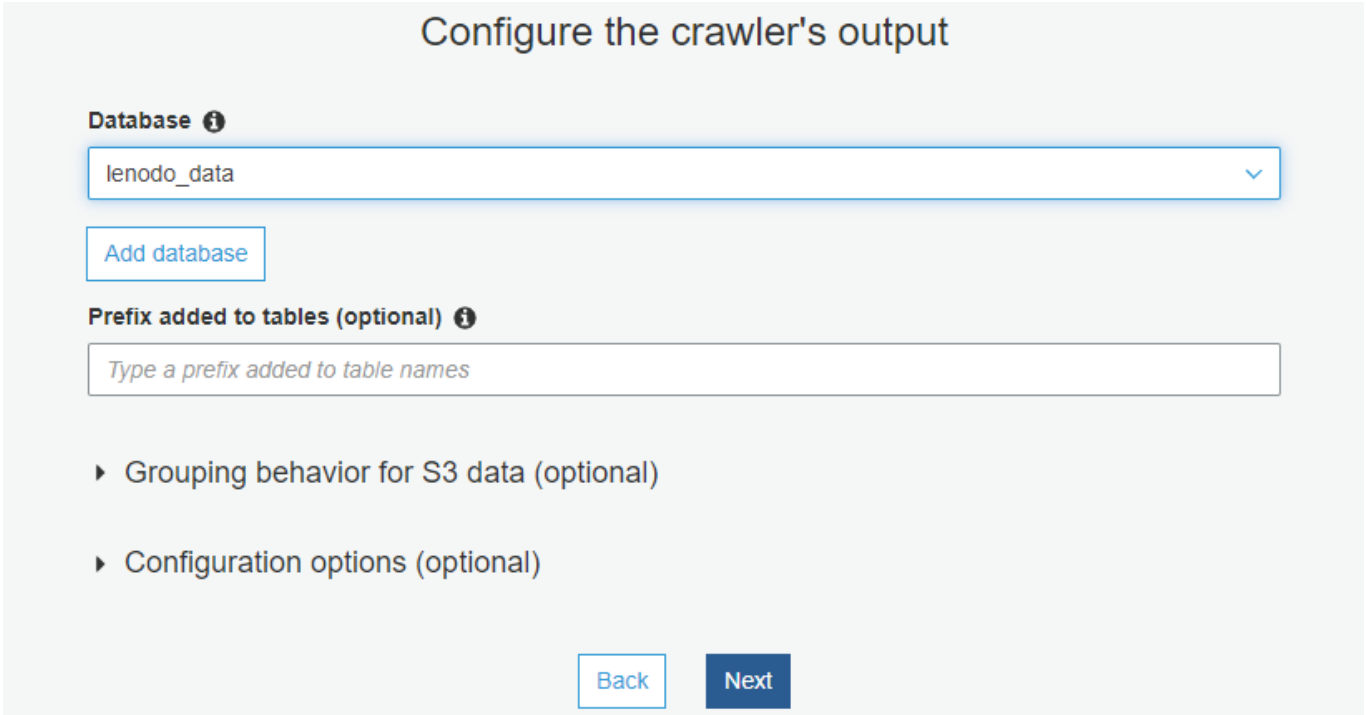


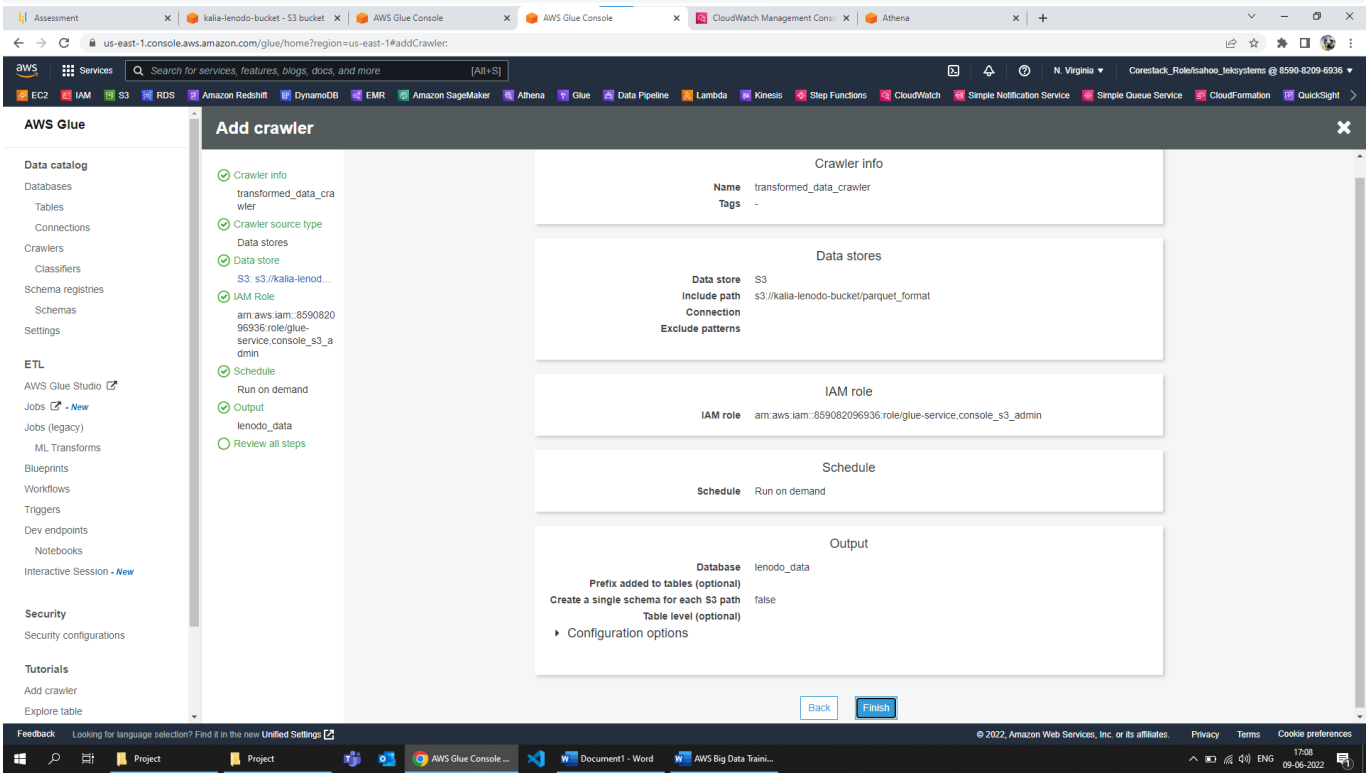




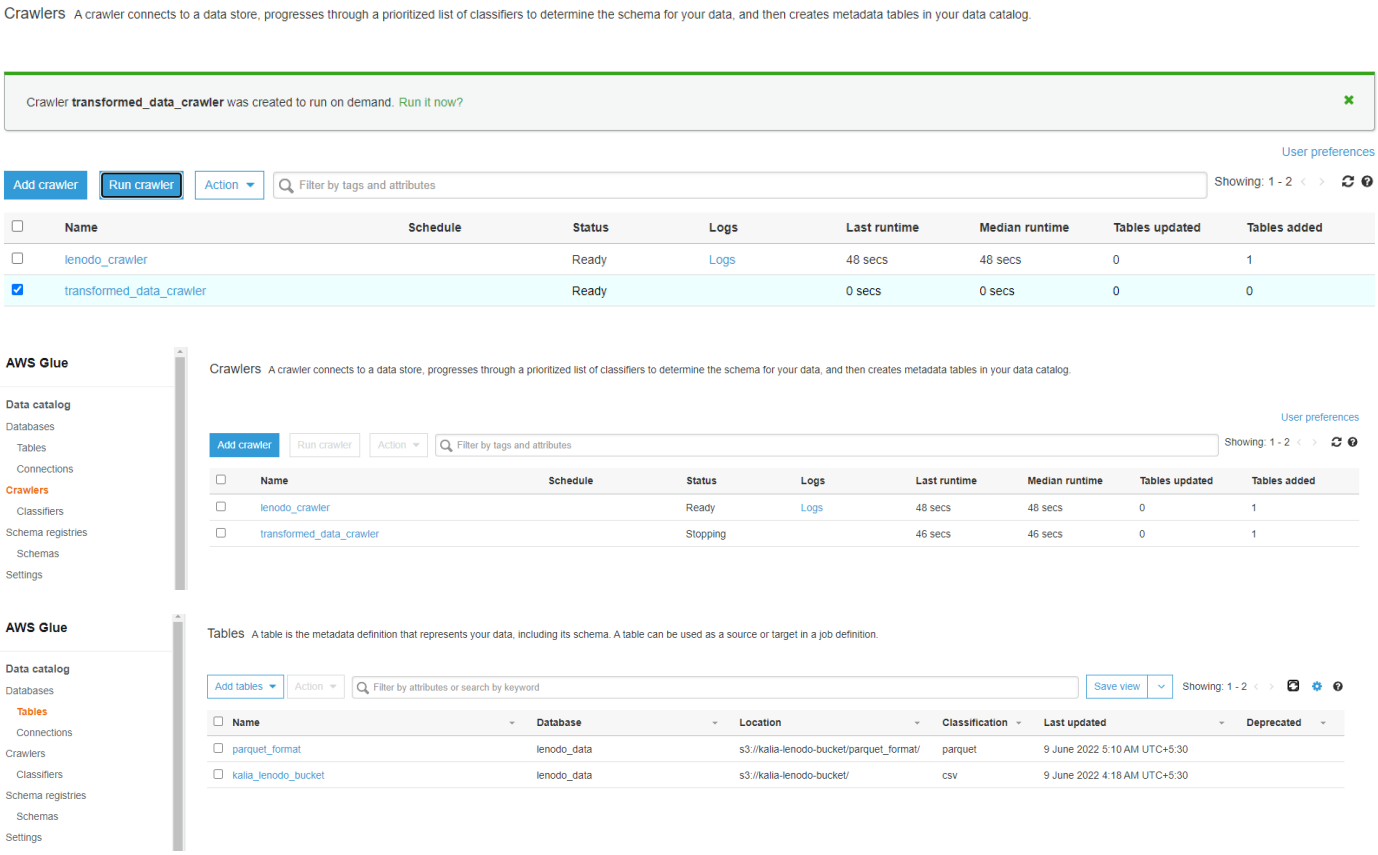


iii. You can choose a different database if u need here I will go with the one I was using (which we created before)

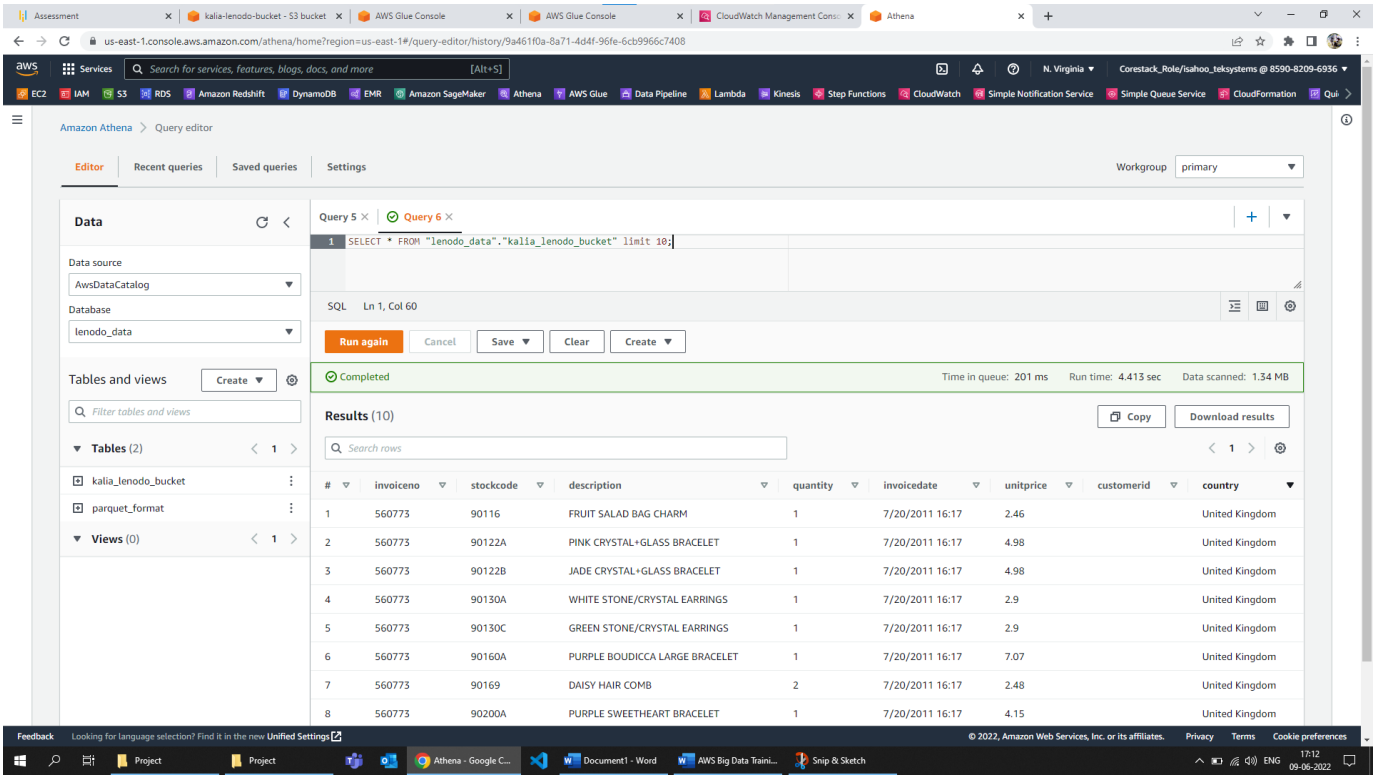


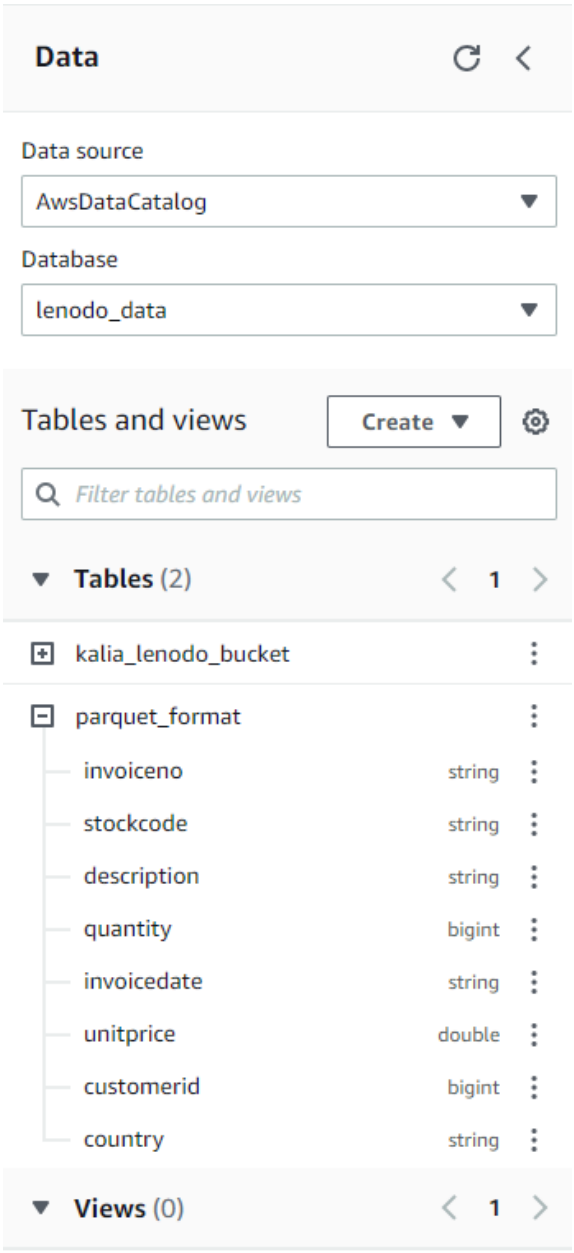


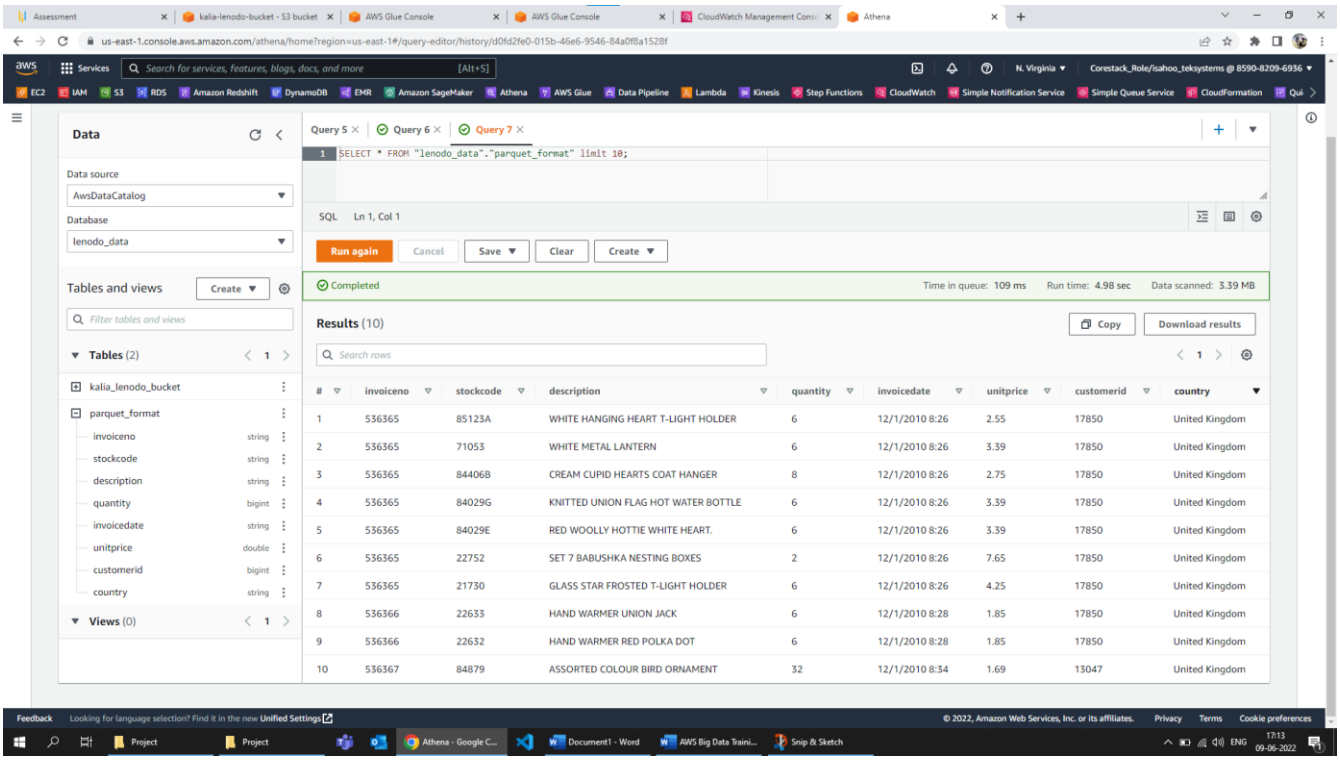
iv. Run the crawler and generate the catalog table.



v. U can go to Athena's query editor the refresh it to see the new table next u can preview the table



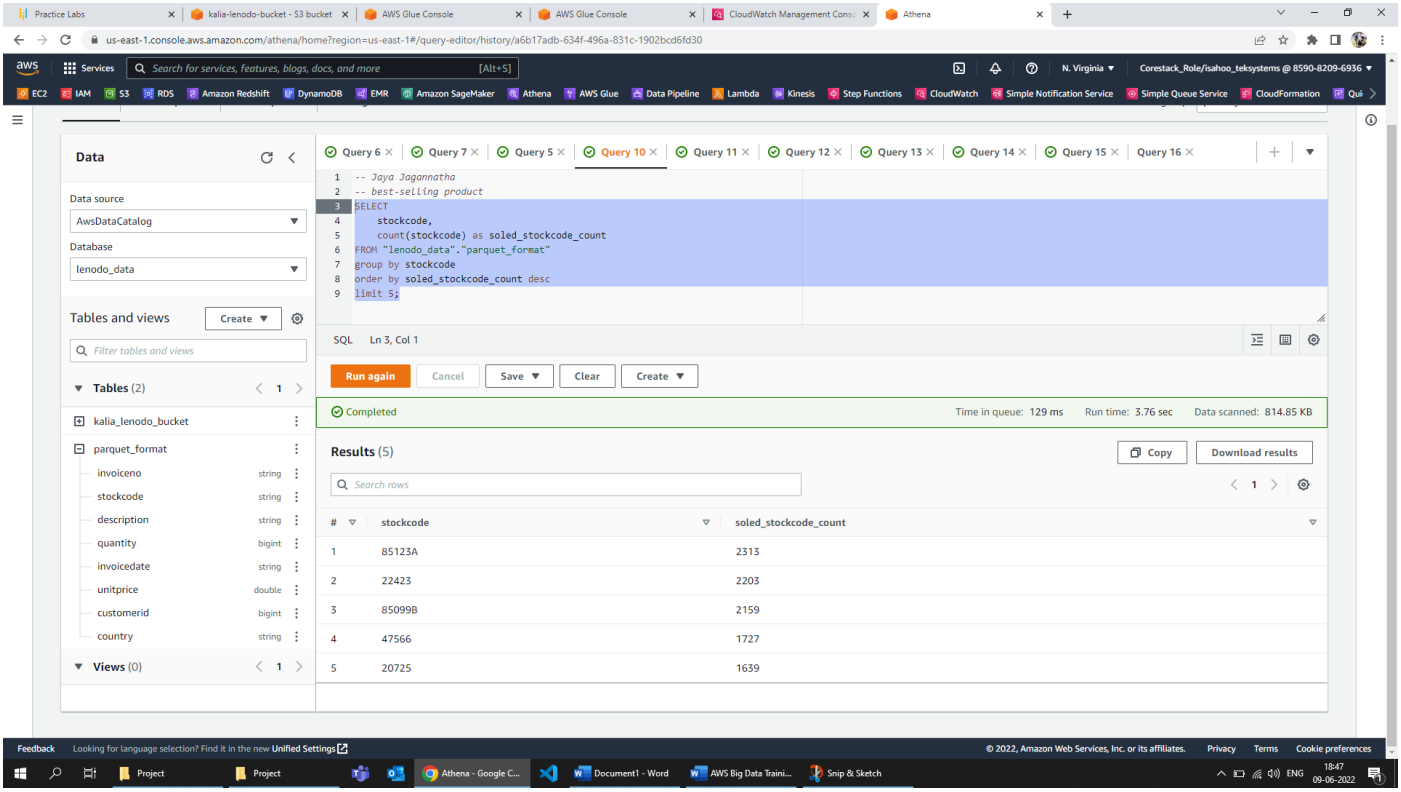

Data
Data source
AwsDataCatalog
Database
lenodo_data
Tables and views
Create
Filter tables and views
Tables (2)
1
kalia_lenodo_bucket
parquet_format
invoiceno string
stockcode string
description string
quantity bigint
invoicedate string
unitprice double
customerid bigint
country string
Views (0)
1


SELECT * FROM "lenodo_data"."parquet_format" limit 10;
Run again
Cancel
Save
Clear
Create
Completed
Time in queue: 109 ms
Run time: 4.98 sec
Data scanned: 3.39 MB
Results (10)
Copy
Download results
invoiceno stockcode description quantity invoicedate unitprice customerid country
1 536365 85123A WHITE HANGING HEART T-LIGHT HOLDER 6 12/1/2010 8:26 2.55 17850 United Kingdom
2 536365 71053 WHITE METAL LANTERN 6 12/1/2010 8:26 3.39 17850 United Kingdom
3 536365 84406B CREAM CUPID HEARTS COAT HANGER 8 12/1/2010 8:26 2.75 17850 United Kingdom
4 536365 84029G KNITTED UNION FLAG HOT WATER BOTTLE 6 12/1/2010 8:26 3.39 17850 United Kingdom
5 536365 84029E RED WOOLLY HOTTIE WHITE HEART. 6 12/1/2010 8:26 3.39 17850 United Kingdom
6 536365 22752 SET 7 BABUSHKA NESTING BOXES 2 12/1/2010 8:26 7.65 17850 United Kingdom
7 536365 21730 GLASS STAR FROSTED T-LIGHT HOLDER 6 12/1/2010 8:26 4.25 17850 United Kingdom
8 536366 22633 HAND WARMER UNION JACK 6 12/1/2010 8:28 1.85 17850 United Kingdom
9 536366 22632 HAND WARMER RED POLKA DOT 6 12/1/2010 8:28 1.85 17850 United Kingdom
10 536367 84879 ASSORTED COLOUR BIRD ORNAMENT 32 12/1/2010 8:34 1.69 13047 United Kingdom

Jaya Jagannatha

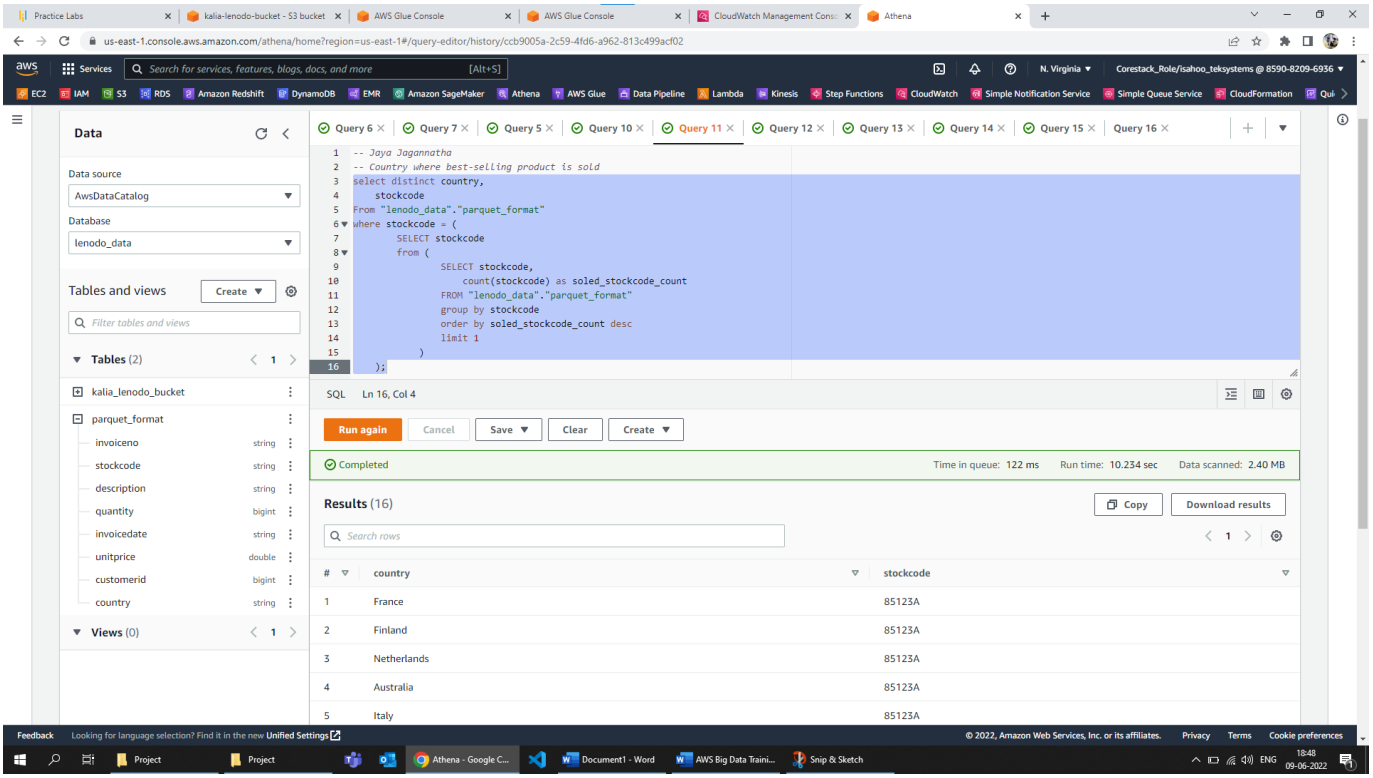
Step 6: Query the data to identify the best-selling item and countries where customers have bought the most-sold item using Athena

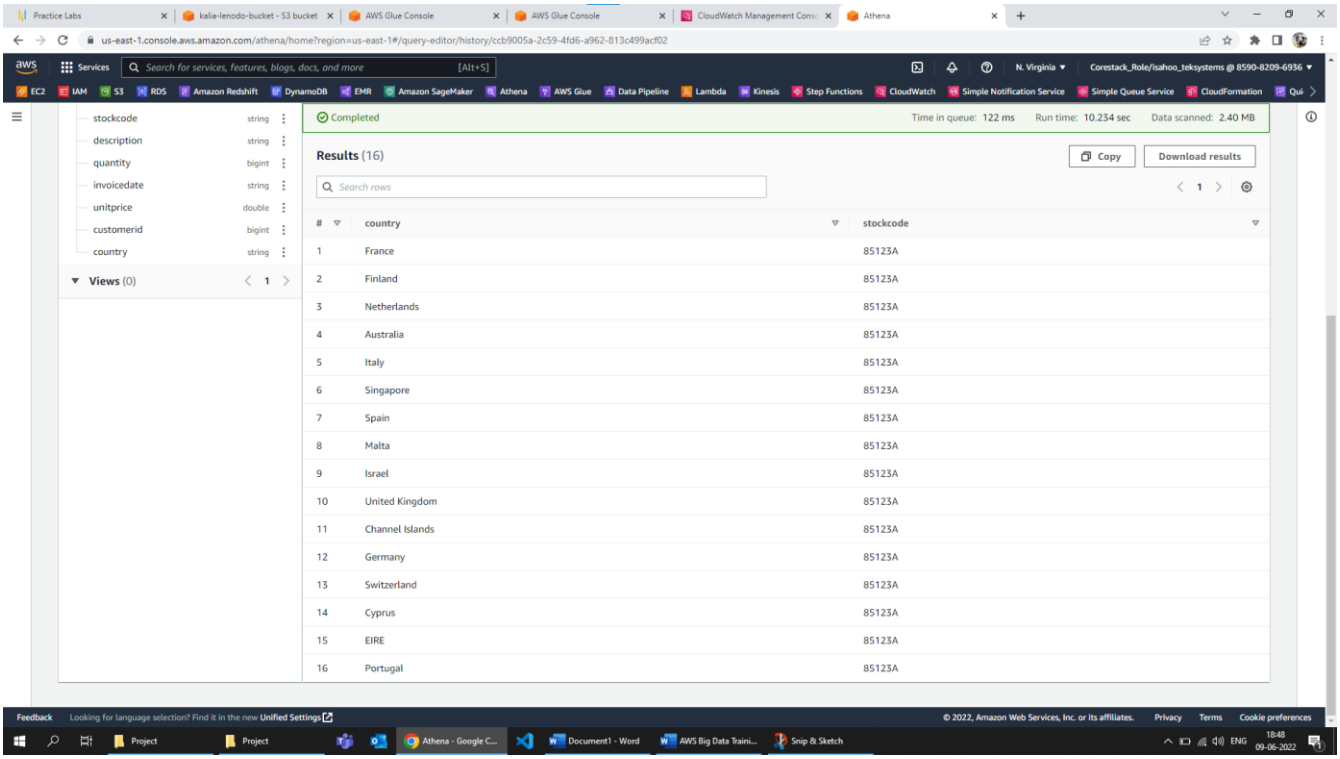
i. Best Selling Product



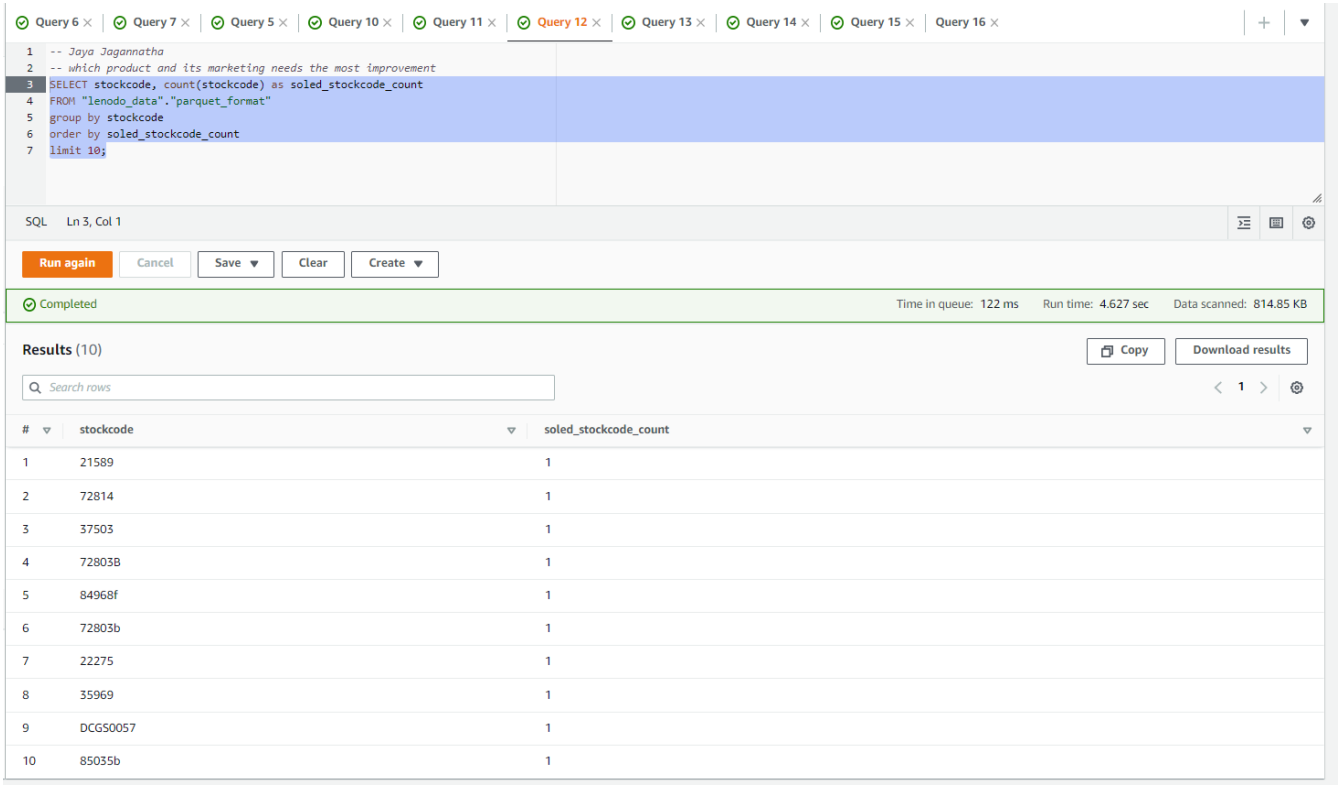
#	stockcode	soled_stockcode_count
1	85123A	2313

ii. In which countries the best selling product is sold

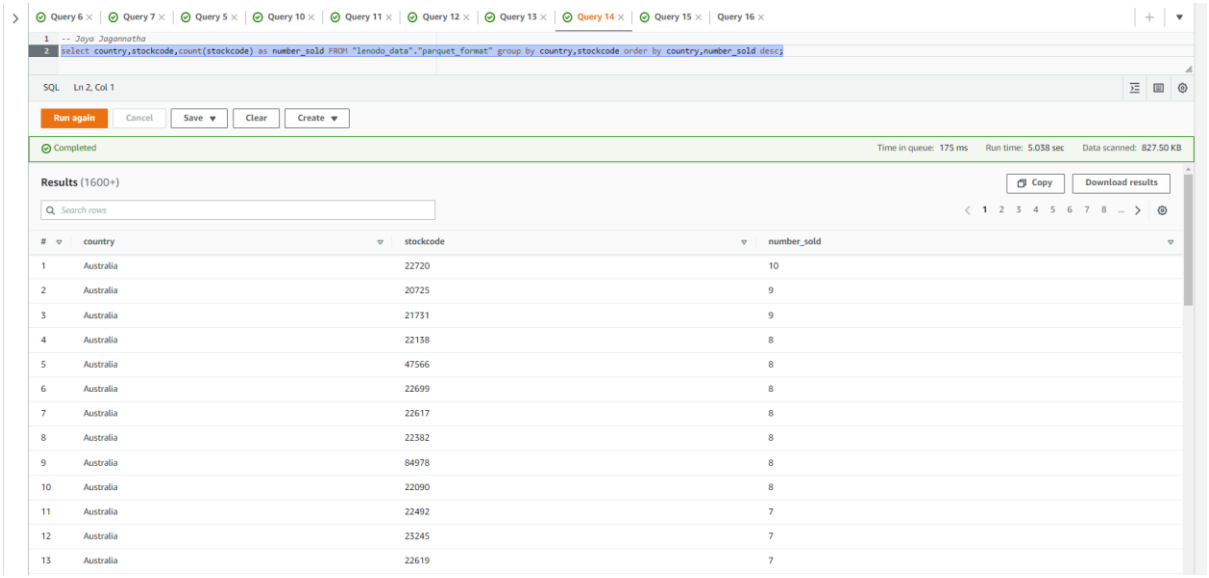




iii. Which products and its marketing needs most improvement (Showing only 5 here)



iv. Detailed view of which country has sale of which product(stockcode) and with which quantity where it's ordered by descending order. As the query result is very long I have just taken screenshot of top 13 rows



```
-- Jaya Jagannatha
select country,stockcode,count(stockcode) as number_sold FROM "lenodo_data"."parquet_format" group by country,stockcode order by country,number_sold desc;
```

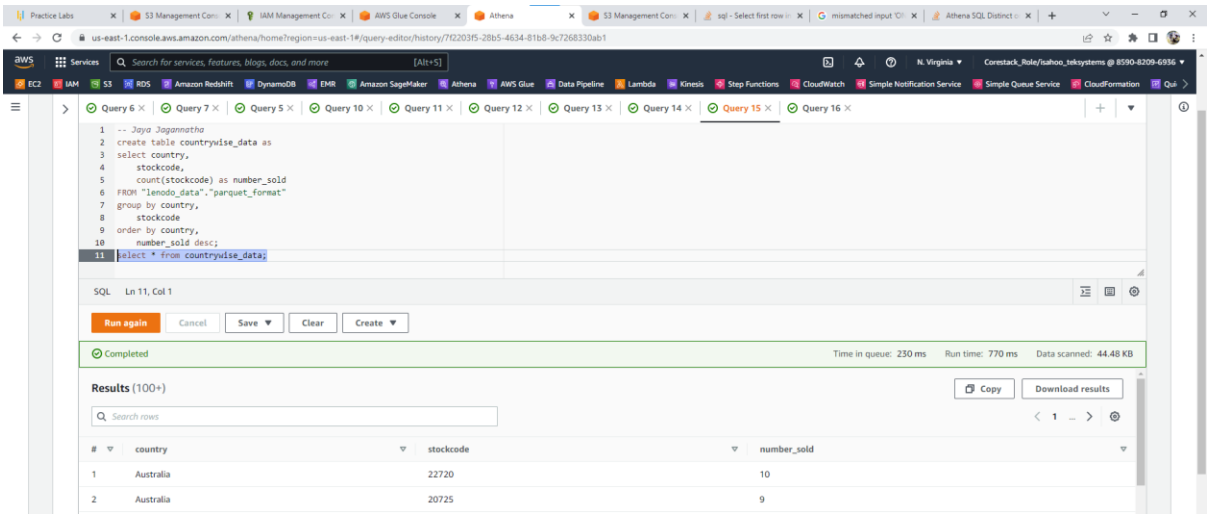
Run again | Cancel | Save | Clear | Create

Completed — Time in queue: 175 ms — Run time: 5.038 sec — Data scanned: 827.50 KB

Results (1600+) — Copy | Download results

#	country	stockcode	number_sold
1	Australia	22720	10
2	Australia	20725	9
3	Australia	21731	9
4	Australia	22138	8
5	Australia	47566	8
6	Australia	22699	8
7	Australia	22617	8
8	Australia	22382	8
9	Australia	84978	8
10	Australia	22090	8
11	Australia	22492	7
12	Australia	23245	7
13	Australia	22619	7

v. Creating a different table and storing this data for further analysis



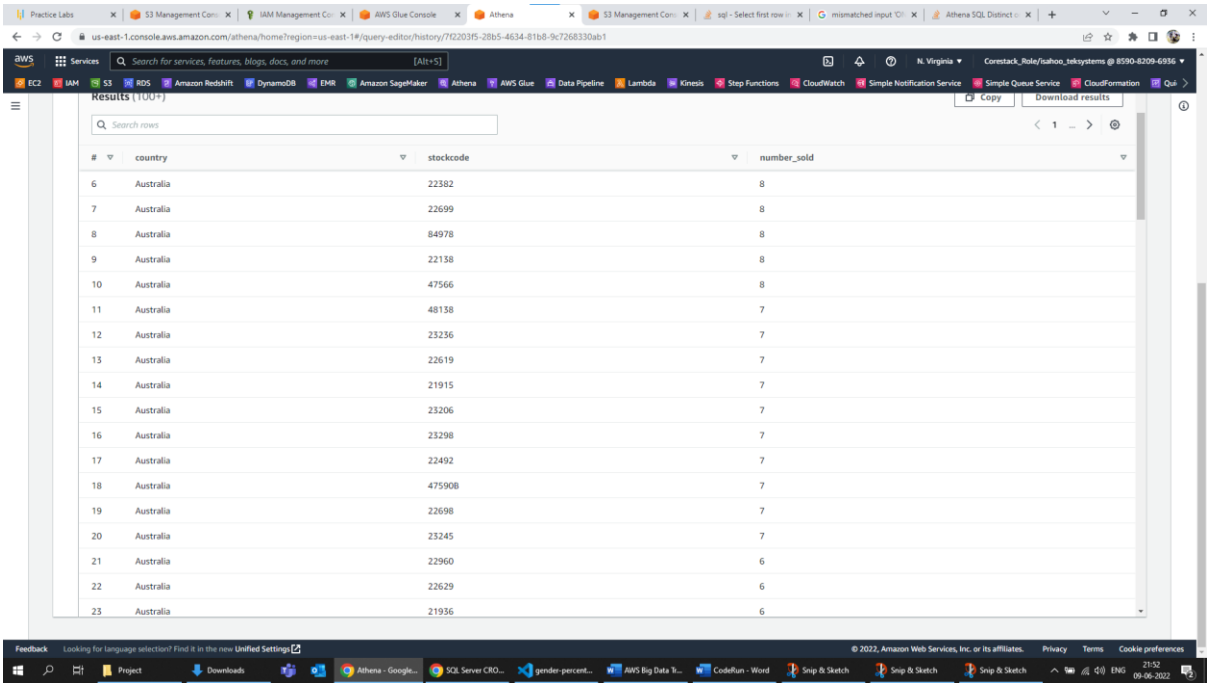
```
-- Jaya Jagannatha
create table countrywise_data as
select country,
    stockcode,
    count(stockcode) as number_sold
FROM "lenodo_data"."parquet_format"
group by country,
    stockcode
order by country,
    number_sold desc;
select * from countrywise_data;
```

Run again | Cancel | Save | Clear | Create

Completed — Time in queue: 230 ms — Run time: 770 ms — Data scanned: 44.48 KB

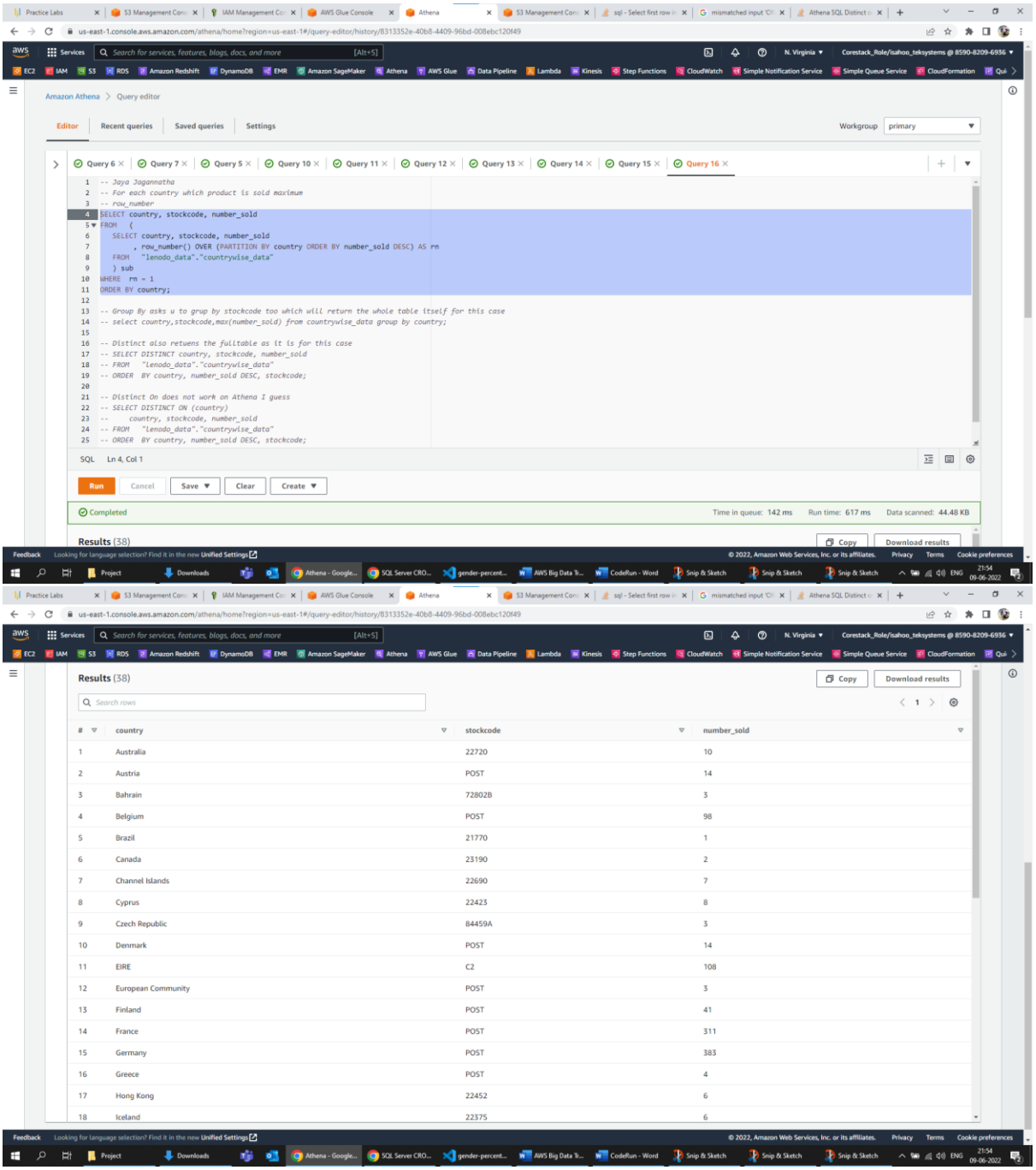
Results (100+) — Copy | Download results

#	country	stockcode	number_sold
1	Australia	22720	10
2	Australia	20725	9



#	country	stockcode	number_sold
6	Australia	22382	8
7	Australia	22699	8
8	Australia	84978	8
9	Australia	22138	8
10	Australia	47566	8
11	Australia	48138	7
12	Australia	23236	7
13	Australia	22619	7
14	Australia	21915	7
15	Australia	23206	7
16	Australia	23298	7
17	Australia	22492	7
18	Australia	47590B	7
19	Australia	22698	7
20	Australia	23245	7
21	Australia	22960	6
22	Australia	22629	6
23	Australia	21936	6

vi. For each country which is the maximum sold product with the count it is sold at and with its stockcode.



SQL Query:

```sql
-- Jaya Jagannatha Kalia Santa

-- csv data table
SELECT * FROM "lenodo_data"."kalia_lenodo_bucket" limit 10;

-- paraquet data table
SELECT * FROM "lenodo_data"."parquet_format" limit 10;
```

```sql
-- best-selling product
SELECT
    stockcode,
    count(stockcode) as soled_stockcode_count
FROM "lenodo_data"."parquet_format"
group by stockcode
order by soled_stockcode_count desc
limit 5;
```

```sql
-- Country where best-selling product is sold
select distinct country,
  stockcode
From "lenodo_data"."parquet_format"
where stockcode = (
    SELECT stockcode
    from (
        SELECT stockcode,
          count(stockcode) as soled_stockcode_count
        FROM "lenodo_data"."parquet_format"
        group by stockcode
        order by soled_stockcode_count desc
        limit 1
      )
  );
```

```sql
-- which product and its marketing needs the most improvement
SELECT stockcode, count(stockcode) as soled_stockcode_count
FROM "lenodo_data"."parquet_format"
group by stockcode
order by soled_stockcode_count;
```

```sql
-- For eachcountry what product sold for which quantity (for each country quantity is in
decending order)
select country,
  stockcode,
  count(stockcode) as number_sold
FROM "lenodo_data"."parquet_format"
group by country,
  stockcode
order by country,
  number_sold desc;

-- Put the data generated in previous query to a table
create table countrywise_data as
select country,
  stockcode,
  count(stockcode) as number_sold
FROM "lenodo_data"."parquet_format"
group by country,
  stockcode
order by country,
  number_sold desc;
-- Print the data using select
select * from countrywise_data;
```

```sql
-- Jaya Jagannatha
-- For each country which product is sold maximum
-- row_number works correctly
select country,stockcode,number_sold
from (
    select country,stockcode,number_sold,
    row_number() over (partition by country order by number_sold desc) as rn
    from "lenodo_data"."countrywise_data"
) sub
where rn = 1
order by country;

-- Group By asks u to grup by stockcode too which will return the whole table itself for
this case
-- select country,stockcode,max(number_sold) from countrywise_data group by country;

-- Distinct also retuens the fulltable as it is for this case
-- SELECT DISTINCT country, stockcode, number_sold
-- FROM   "lenodo_data"."countrywise_data"
-- ORDER  BY country, number_sold DESC, stockcode;

-- Distinct On does not work on Athena I guess
-- SELECT DISTINCT ON (country)
--     country, stockcode, number_sold
-- FROM   "lenodo_data"."countrywise_data"
-- ORDER  BY country, number_sold DESC, stockcode;
```