Sachin Chandrashekhar - Data Engineering Hub

🎯 Follow me on Twitter: https://x.com/AWSCloudSachin

DATA ENGINEERING HUB

Disclaimer: This document and the supporting artifacts are copyrighted and strictly for personal use only.

This document is reserved for people enrolled into **The Ultimate RADE™ Program by Sachin Chandrashekhar.**

Please do not share this document or upload publicly, it is intended for personal use only.

If you've obtained this document for free on a website that is not the course's website or on social media or public repositories, please reach out to legal@dataengineeringhub.in.

I reiterate – You must go through the self-paced sections 14 and 15 to get Glue in entirety, though I have tried to cover as much as I can through this hackathon.

Self-paced Glue Sections are in great depth which you won't find any where else.

"Main" Components of Glue:

1) Glue Catalog
2) Glue Crawler
3) Glue Workflows (optional to learn)
4) Glue ETL – Three modes –
    a. Interactive Code Notebook
    b. Glue Script
        i. Glue Python shell.
        ii. Glue Spark
    c. Glue Visual – Ignore this

We have seen Glue catalog in Athena. We will see again today

We have talked about Crawler but we will see it live today.

We have already seen Glue Python Shell and have done hands on.

We will do Glue Spark job today.

So, what's our architecture for Glue Spark?

Remember this one architecture for your interviews -,

we get data into S3 and then for ETL, we can either use Lambda, Glue Python Shell, ECS Fargate, Athena, or EMR etc.

Here, we will use Glue pyspark to do the ETL and then load curated data back into S3 curated layer.

From s3 curated layer, you will then load data into Redshift table using COPY command in a stored procedure that you saw in the redshift hackathon.

Orchestration will be like this –

Data comes into S3, S3 event triggers lambda which triggers step function.

Step function triggers Glue Spark job, then after this completes, next step in step function triggers Redshift Stored proc that runs the COPY command which loads the curated data from S3 into the redshift table.

So, essentially,

Source - S3 (Data Lake) – raw layer

Event driven orchestration – Triggers – Step function (Full orchestration)

Step function triggers – (Lambda/ Glue/ ECS Fargate / EMR / Athena) for ETL

Transformed data is stored back in S3 – curated layer

Then via Step function, run stored proc in Redshift to

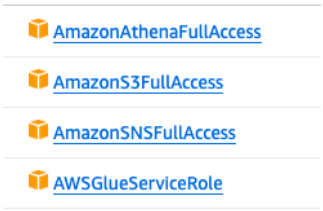
1) Truncate stage table
2) COPY S3 data into stage table
3) Upsert Stage to core table

SNS for any failure notification

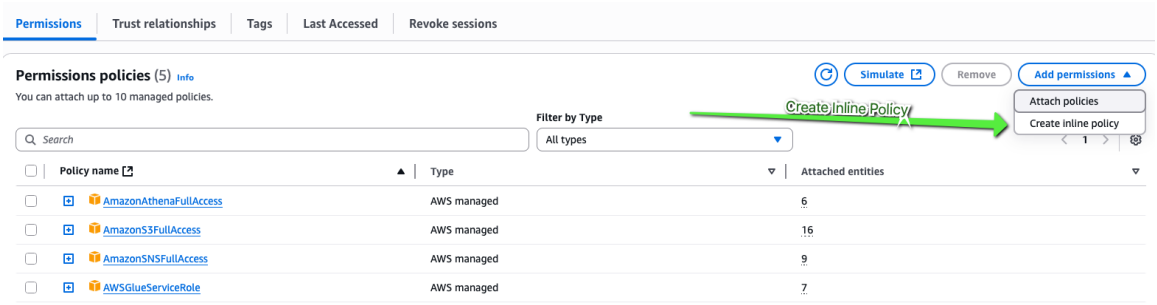
Give Access to the S3 data through Glue catalog/Athena to Data

Scientists ; Give Access to Redshift table to Data Analysts

Let's start by creating an IAM role "GlueServiceRole" created for Glue with appropriate permissions.



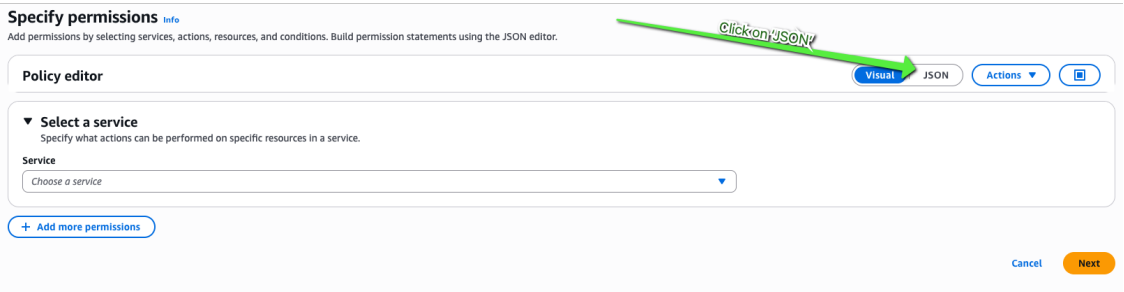
Next, create an inline policy after you have created a role with the above policies. On the same role, create an inline policy.



Sachin Chandrashekhar - Data Engineering Hub

🎯 Follow me on Twitter: https://x.com/AWSCloudSachin

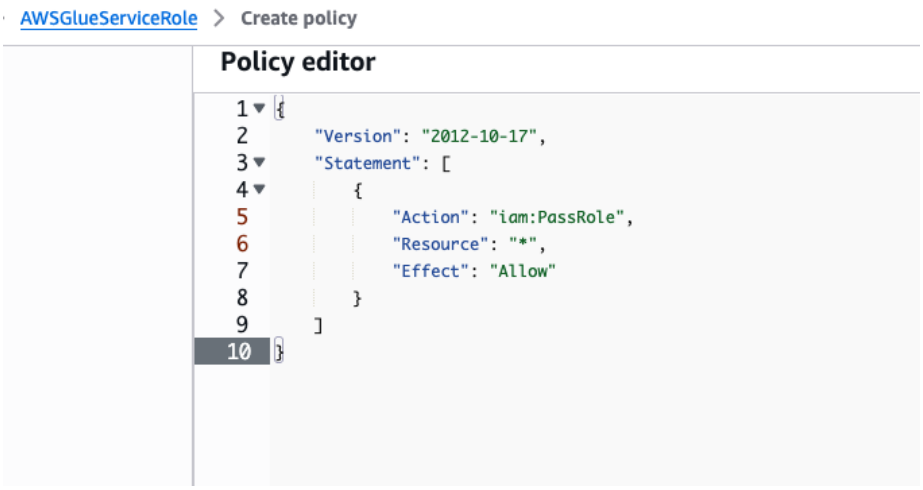
Next,



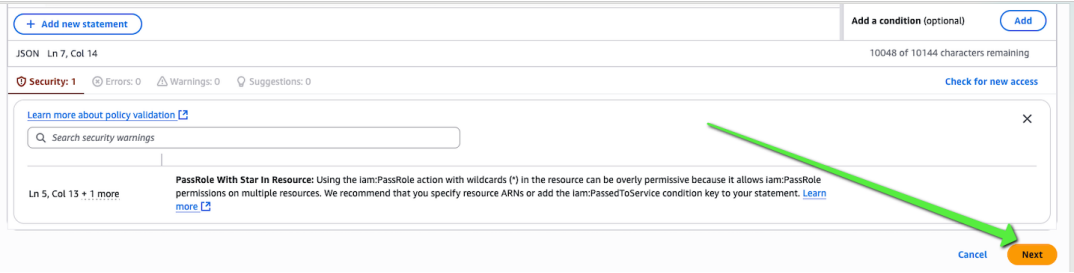
Copy the below

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "iam:PassRole",
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

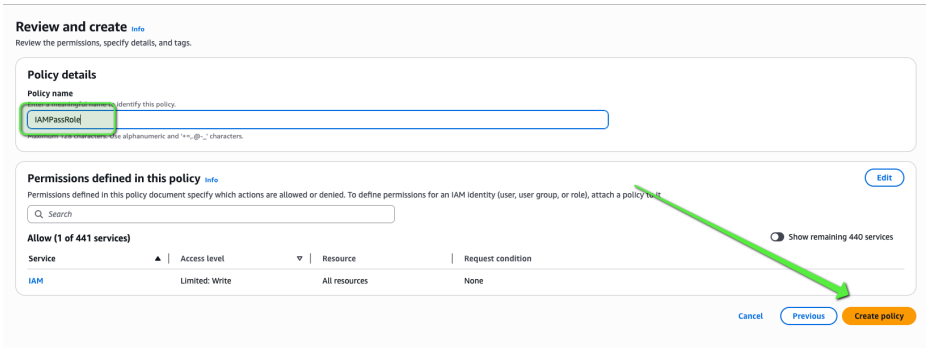
Paste the above below:



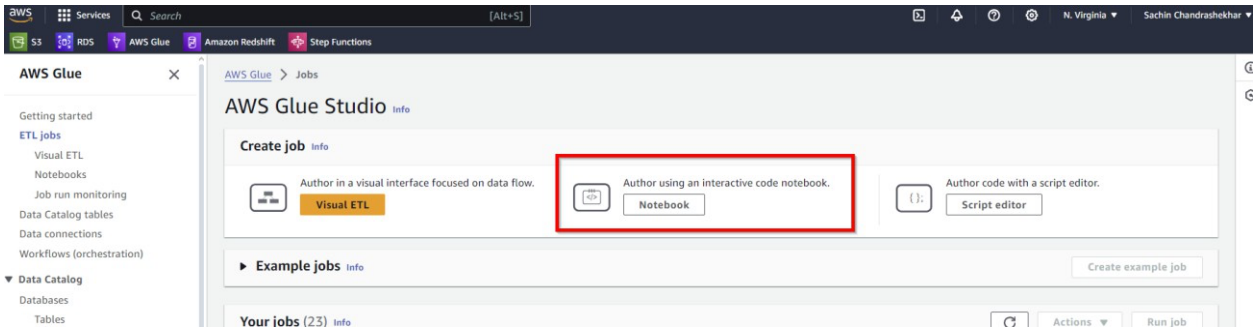
Click on Next



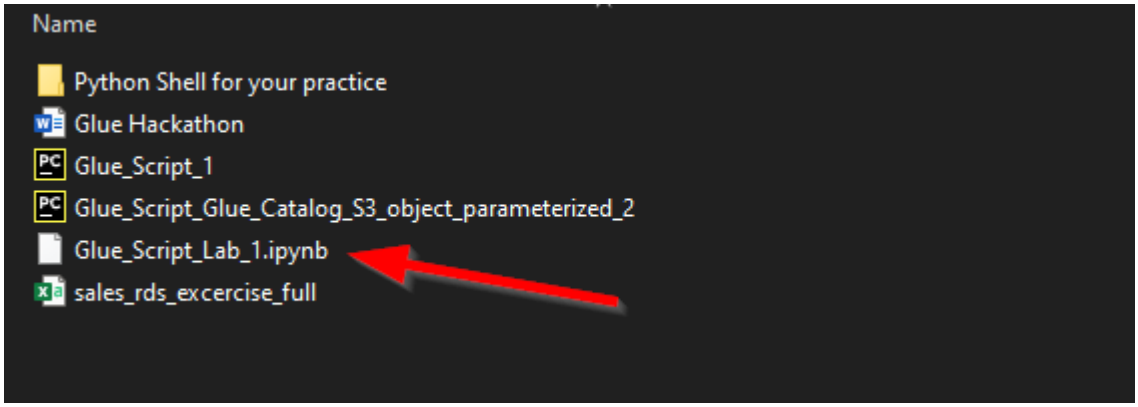
Provide name and create

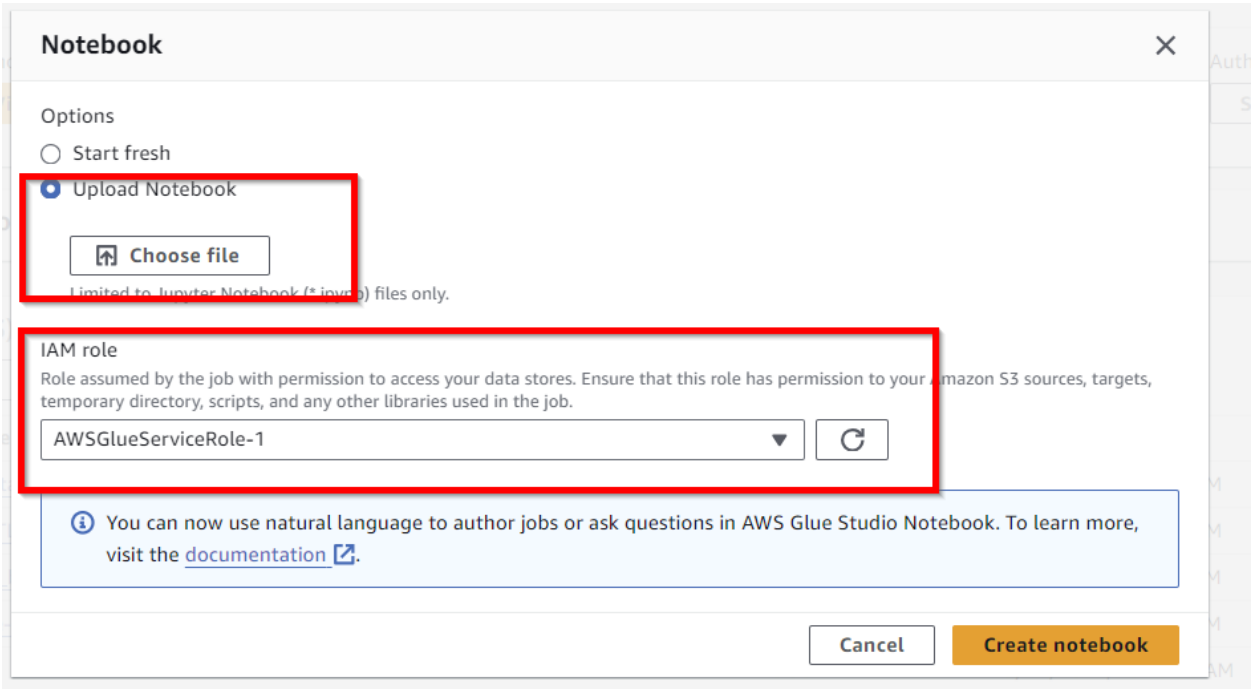


Let's NOW start with Glue Interactive Notebook



Use the Notebook provided Glue_Script_Lab_1.ipynb to upload for your practice.





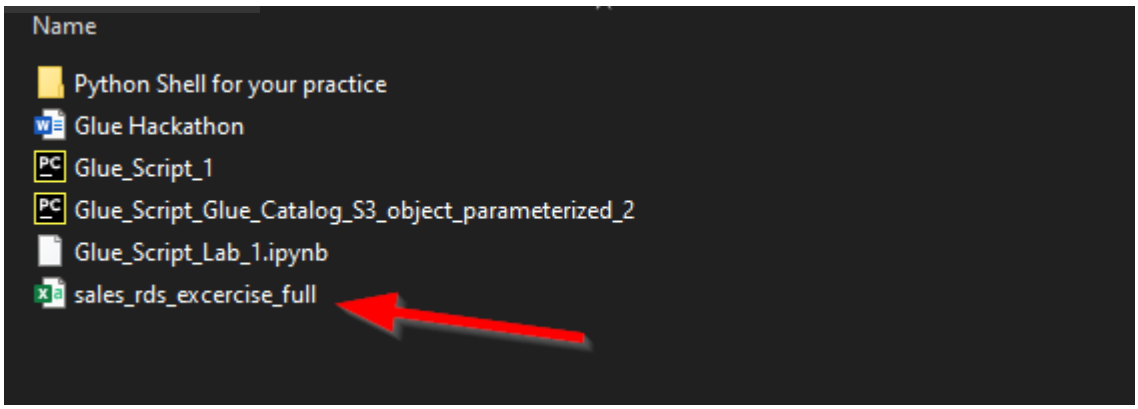
We will use the Sales file we have always been using – Make sure it is uploaded in your bucket.

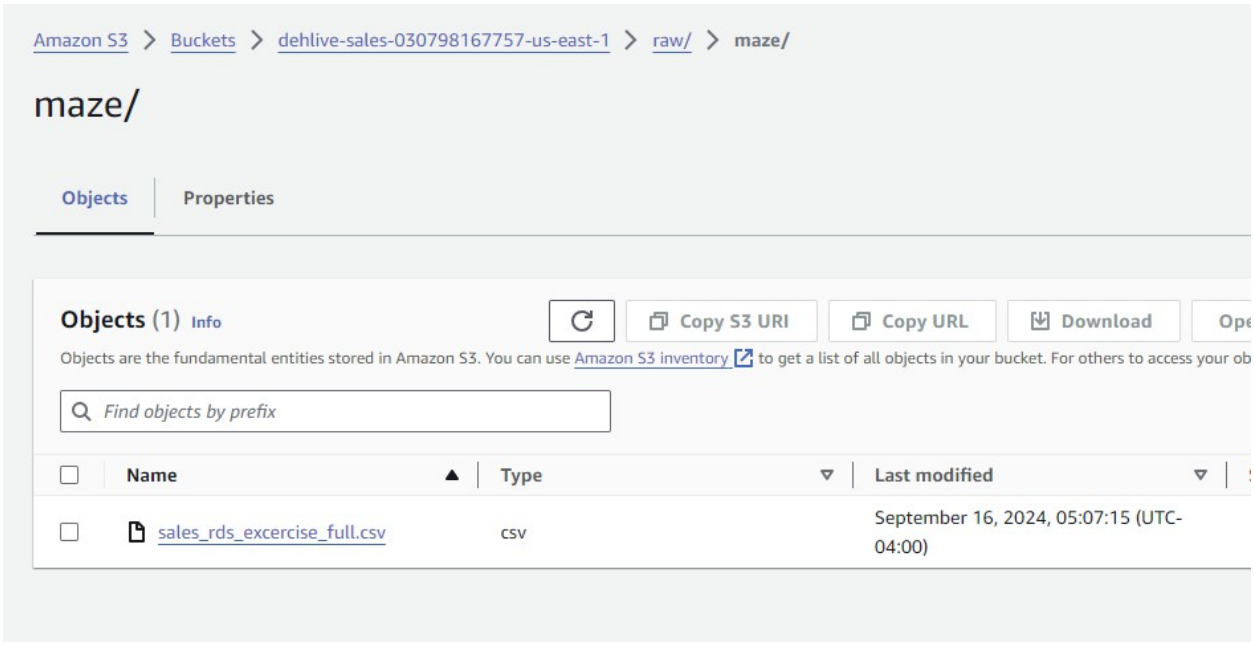
s3://dehlive-sales-<your account number>-us-east-1/raw/maze/sales_rds_excercise_full.csv

Else upload using the file I provided in the hackathon zip.

Sachin Chandrashekhar - Data Engineering Hub

🎯 Follow me on Twitter: https://x.com/AWSCloudSachin





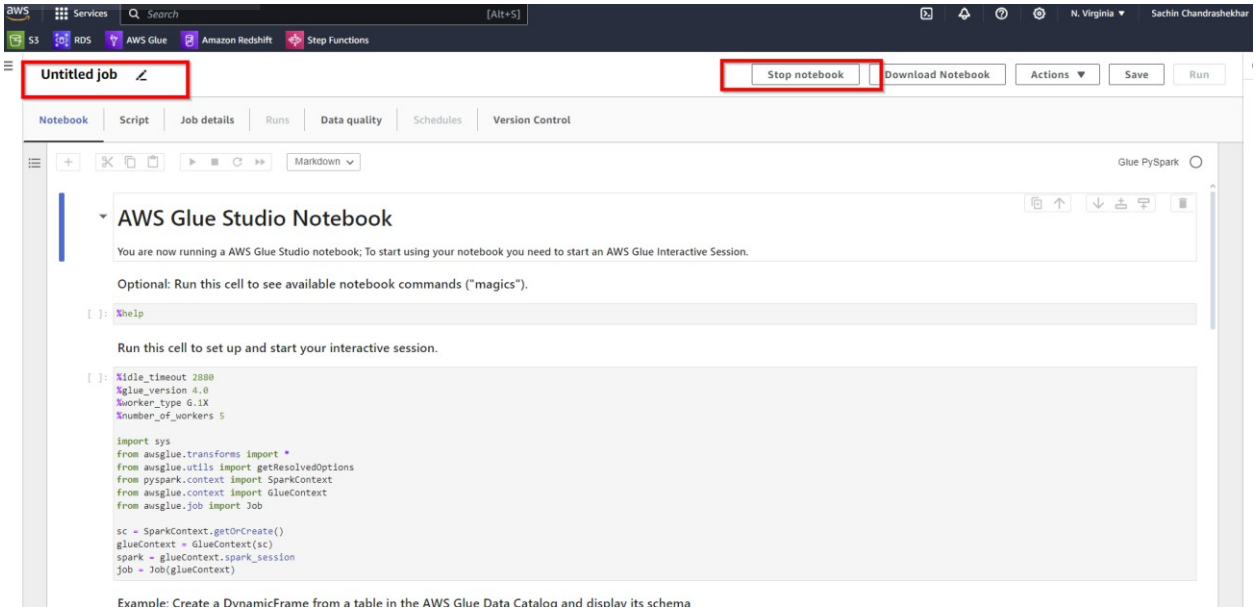
Spinning up notebooks is very expensive.

Rename the job appropriately and STOP NOTEBOOK WHEN DONE.

Keep a reminder in your phone to stop notebooks.

Sachin Chandrashekhar - Data Engineering Hub

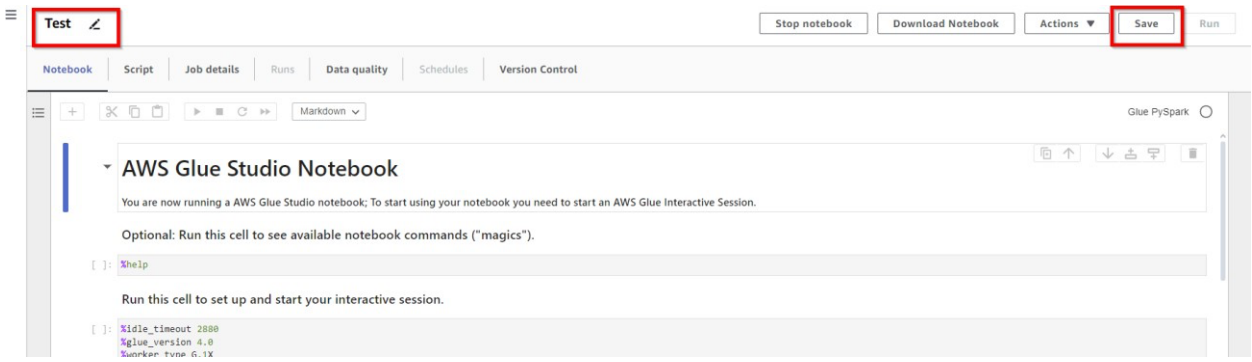
🎯 Follow me on Twitter: https://x.com/AWSCloudSachin



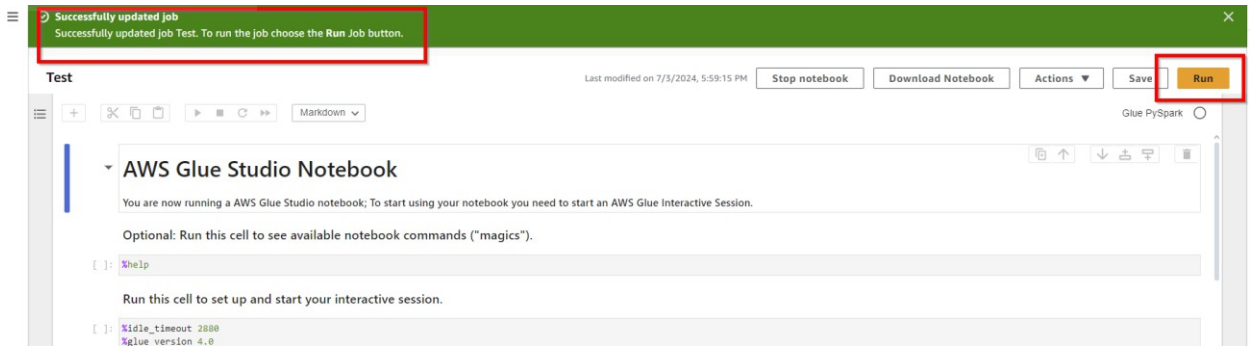
Understand what Spark , SparkContext , GlueContext and Spark session are and what the above boilerplate code is

Reference the file path to update the notebook code.

Renamed as shown below and “Save” the job.

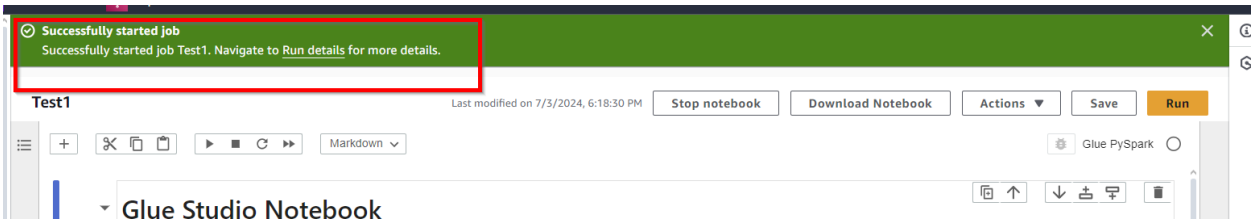


Why do we use Notebook? For iterative development. First step we do is iteratively develop our glue code through notebooks

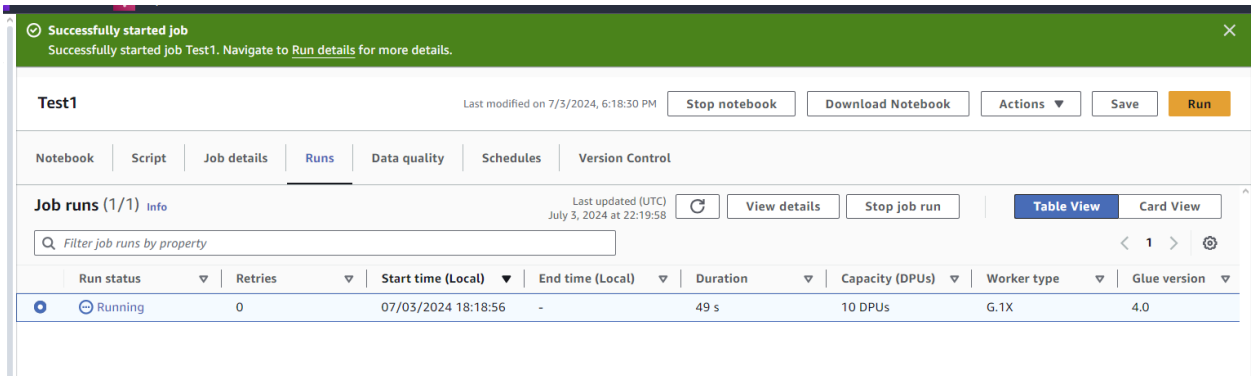


After you “save” it, you can just “Run” then job.

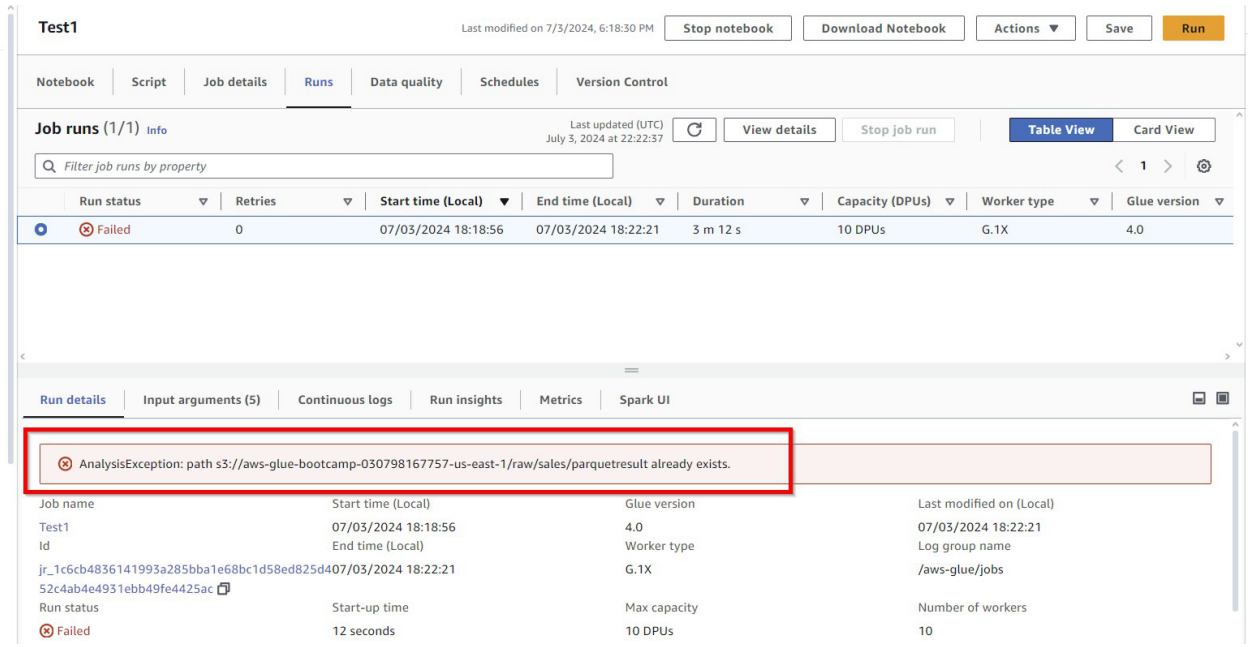
Show’s like this after you run – Click on “Run Details”



Run Details:

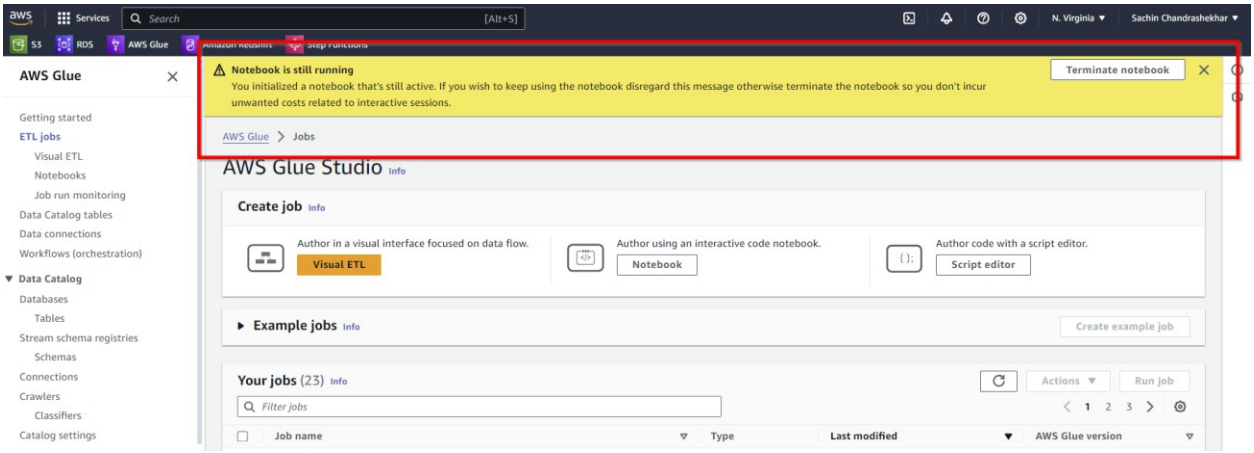


If the job fails, you get the error like this:



If your job fails, see why it fails – perhaps due to fill missing the referenced path or something else; update the code and run the job again.

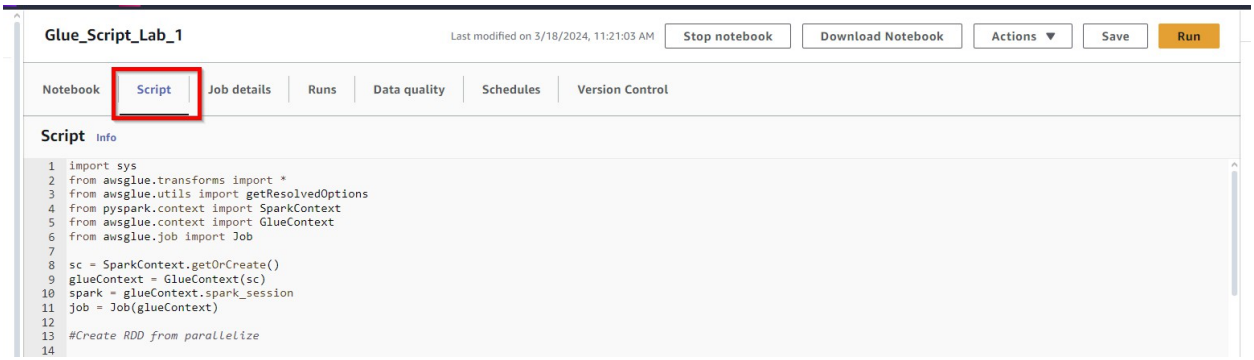
If you keep the notebook running, Glue warns you of that to terminate.



Sachin Chandrashekhar - Data Engineering Hub

🎯 Follow me on Twitter: https://x.com/AWSCloudSachin

So, after the iterative development is done, and you are confident of running this job as a script (if your organization needs the jobs to be in form of scripts) then simply get the script by clicking on “Script”



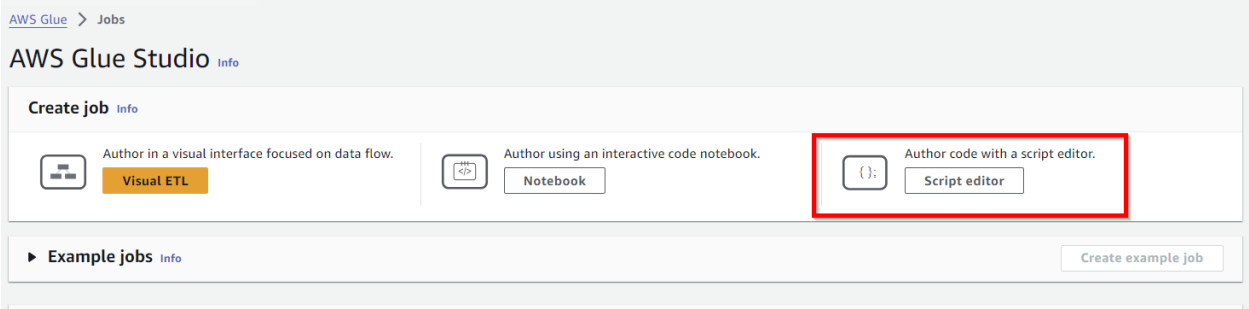
The entire notebook code is moved to the script.

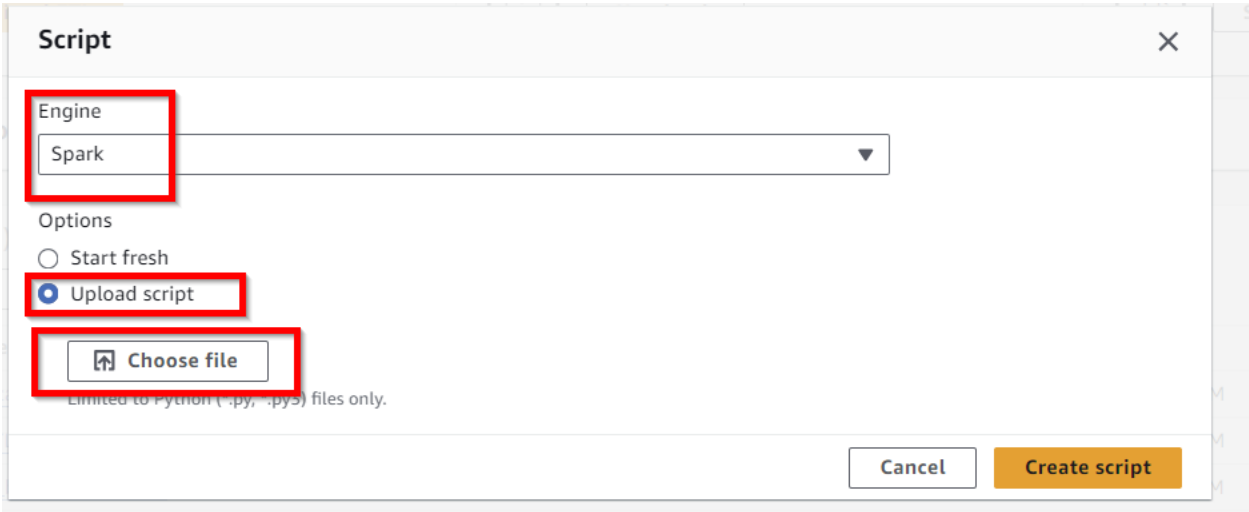
You can then edit this script to remove your print/ debugging statements. I have already done that for you and have saved the code from script.

You don’t have to follow the below screenshots during the class.

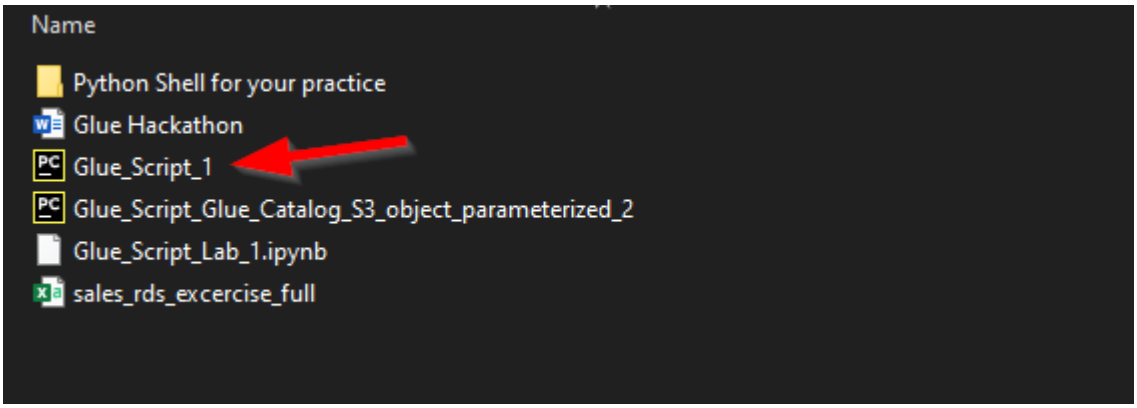
Now, you can create a glue job based on script. – So far we have seen notebook based job.

Choose the third option below as highlighted.



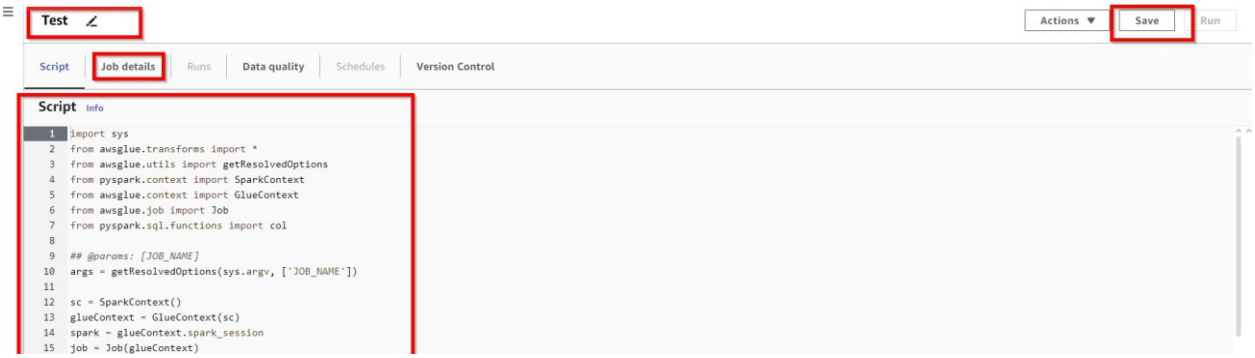


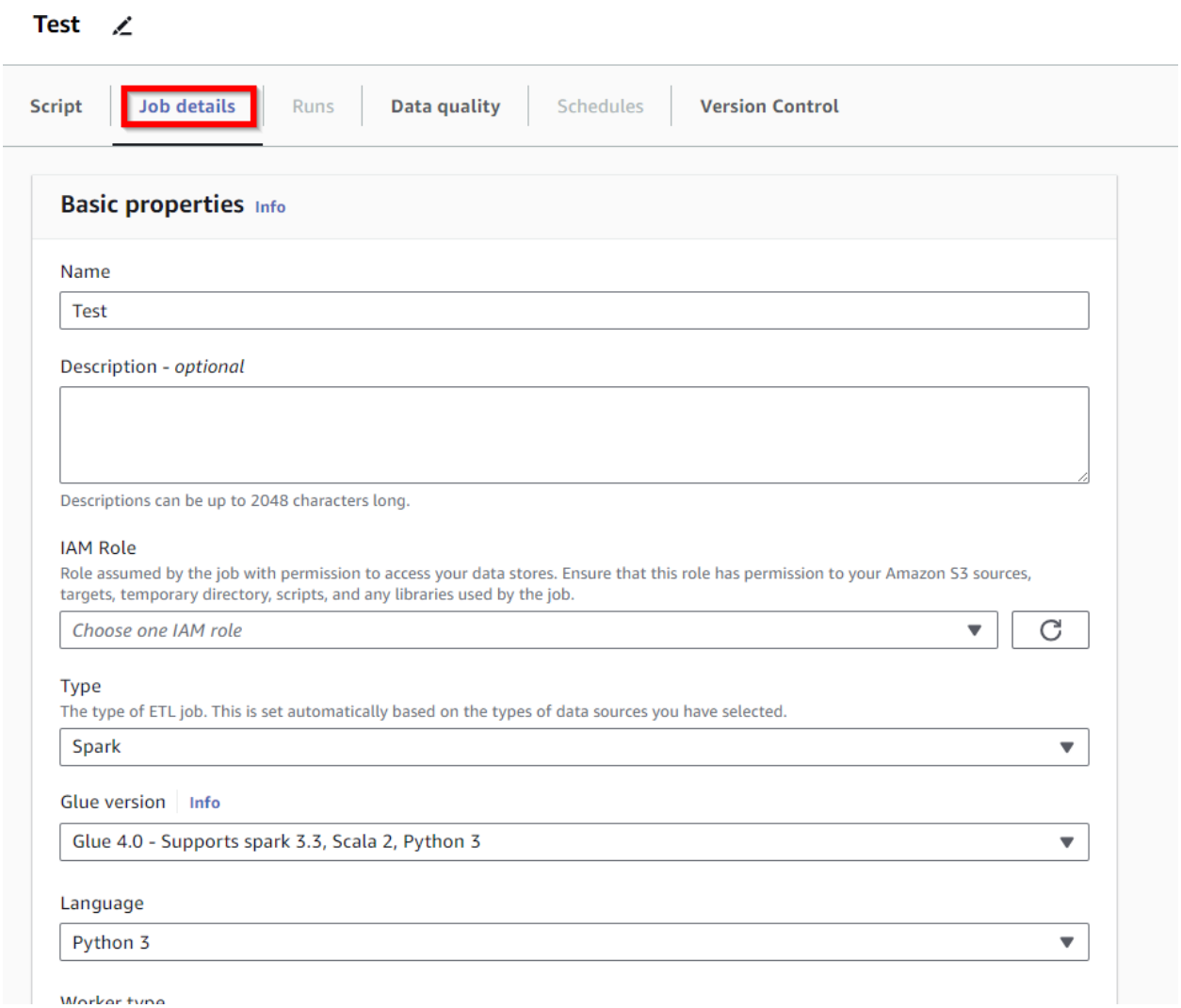
Use Glue_Script_1.py from the zip file.



Script shows up below

Rename the job to whatever you like. Then, save and then go to Job details.





Don’t do the above during the class ; you may continue with this script in your free time.

Sachin Chandrashekhar - Data Engineering Hub

🎯 Follow me on Twitter: https://x.com/AWSCloudSachin

For now, we will use a better use case for script-based coding now.

So, what's our architecture? Again, As discussed in redshift hackathon, we get data into S3 and then for ETL, we can use Lambda, Glue Python Shell, Athena, EMR, ECS Fargate, etc.

Here, we will use Glue pyspark to do the ETL and then load curated data back into S3 curated layer.

From s3 curated layer, you will then load data into Redshift table using COPY command in a stored procedure that you saw in the redshift hackathon.

Orchestration will be like this –

Data comes into S3, S3 event triggers lambda and this in turn triggers step function.

Step function triggers Glue Spark job, then after this completes, next step in step function triggers Redshift Stored proc that runs the COPY command which loads the curated data from S3 into the redshift table.

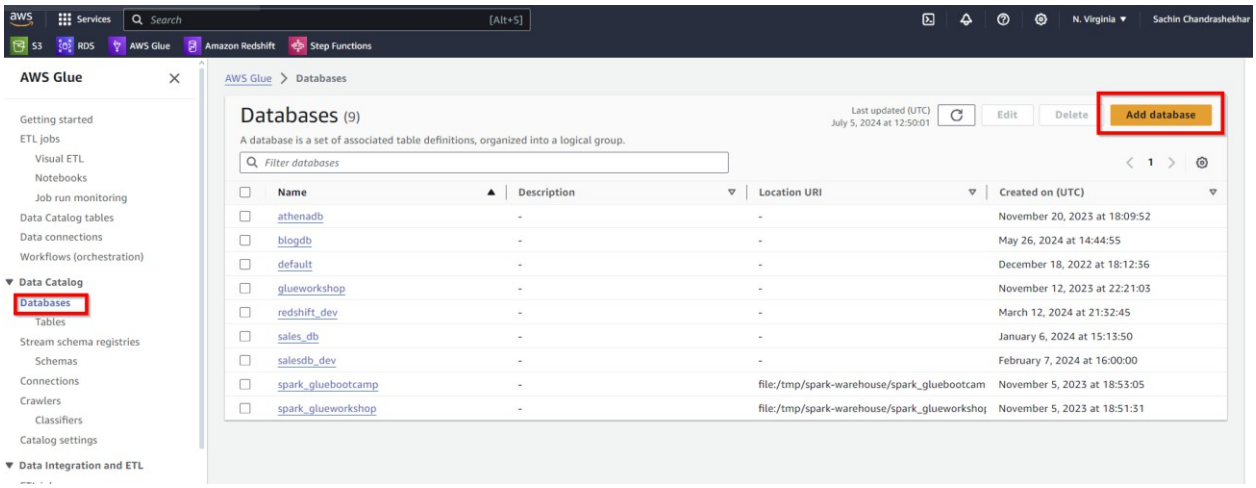
For now, we will see how we can use Glue spark to read data from S3 and create curated data back into S3.

We will just do the ETL using Glue and not load to Redshift - that you have already seen and done in Redshift hackathon.

So, Use the sales file and let's create glue crawler to create a glue catalog object.

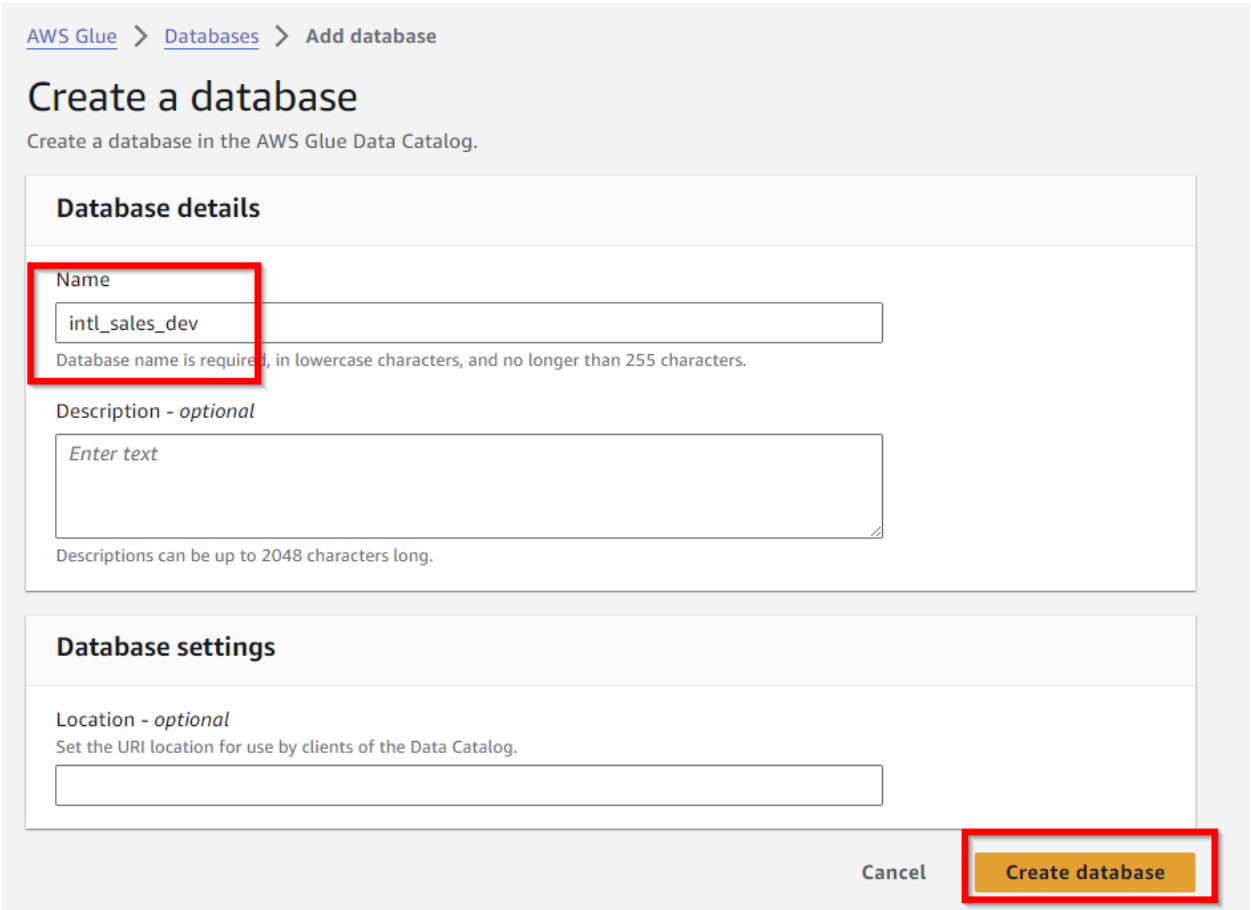
For that, what's the first step?

We need a database. Let's create that. You did this from Athena console in the past if you remember In Athena Hackathon.
Now we will create one directly in the console.



Use this database name – intl_sales_dev

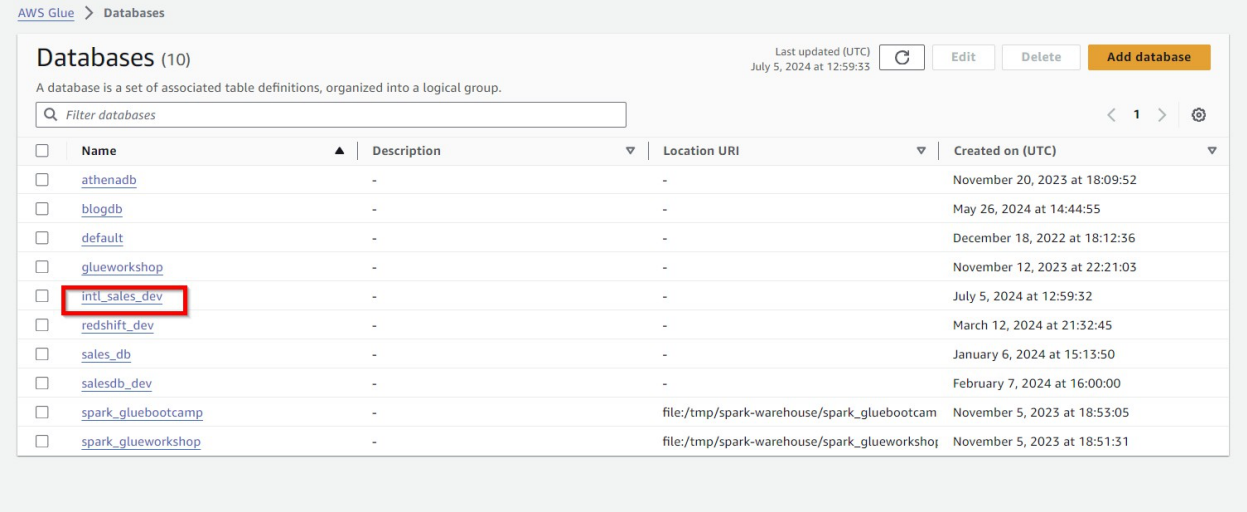
- Indicates database holds international sales data and database is in development environment.



Sachin Chandrashekhar - Data Engineering Hub

🎯 Follow me on Twitter: https://x.com/AWSCloudSachin

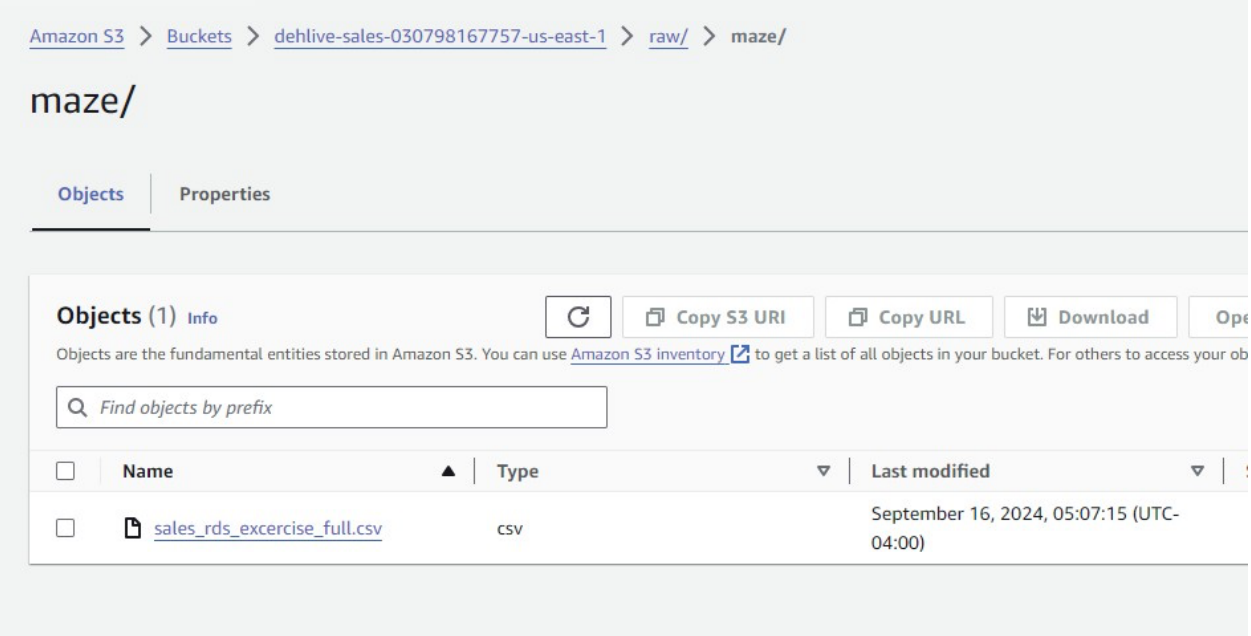
Shows up here –



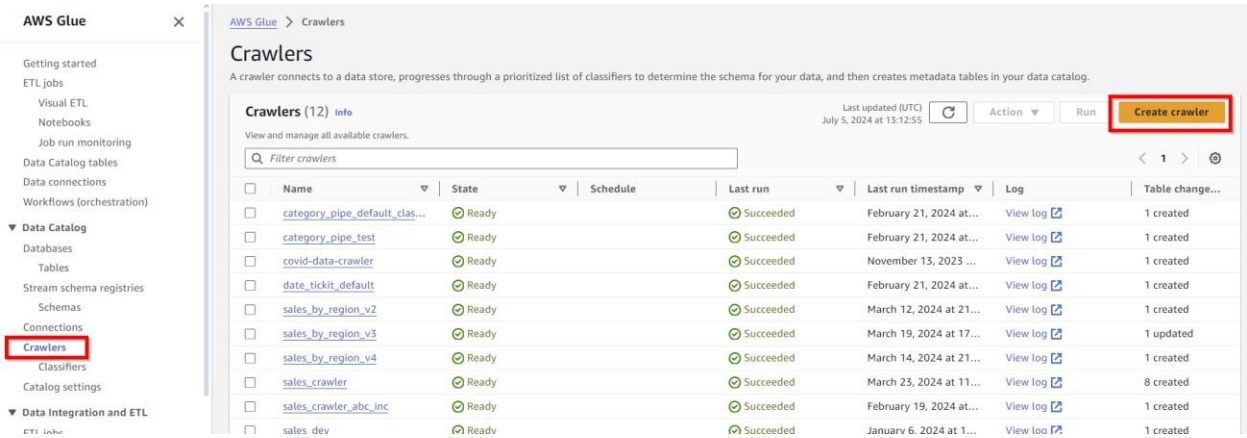
Now, we need to crawl the data file with the help of a crawler so that it can automatically create a glue catalog object for us.

So, first, let's ensure we have the file uploaded.

I have uploaded it an existing bucket and under raw folder – why raw? It's a raw file from source.

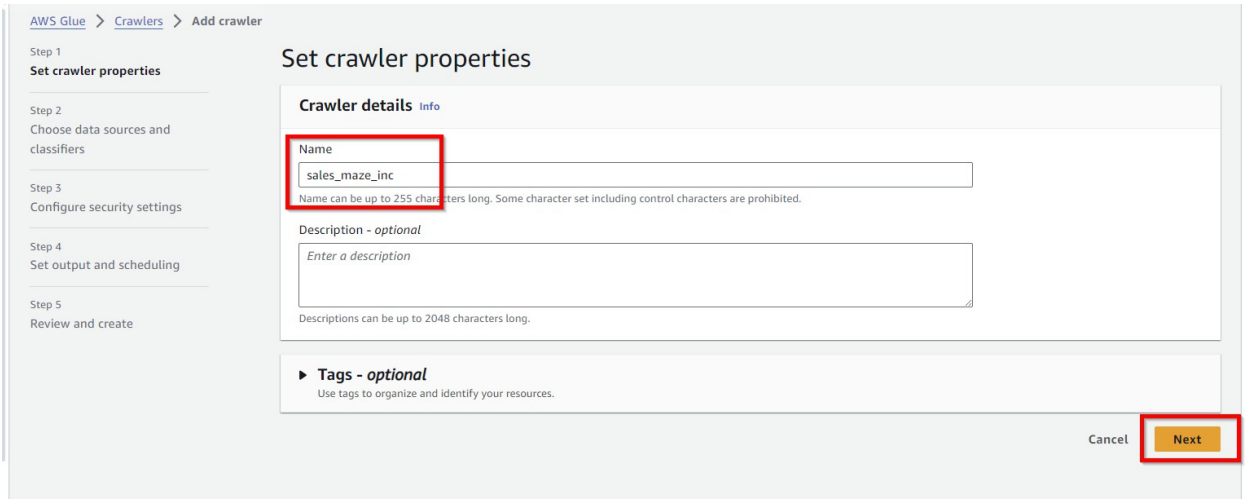


Now, let's create a crawler.



Enter 'sales_maze_inc'

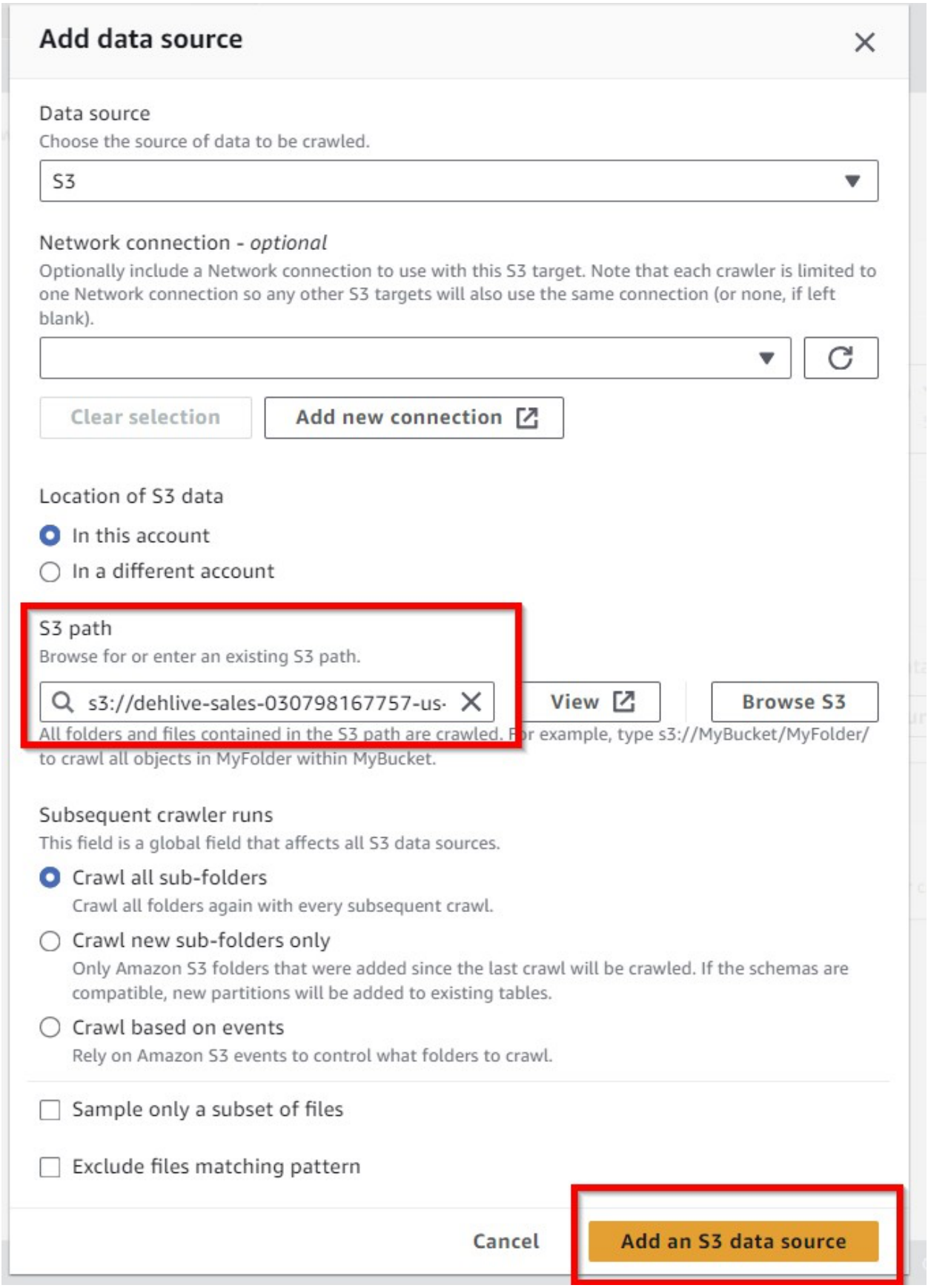
(indicates sales file from maze inc company)

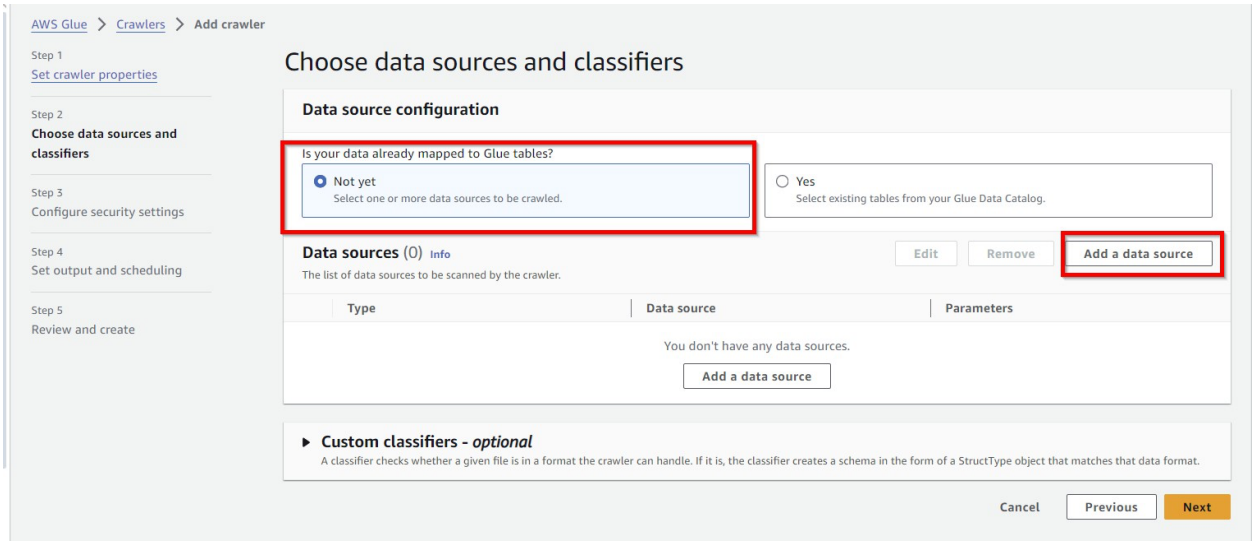


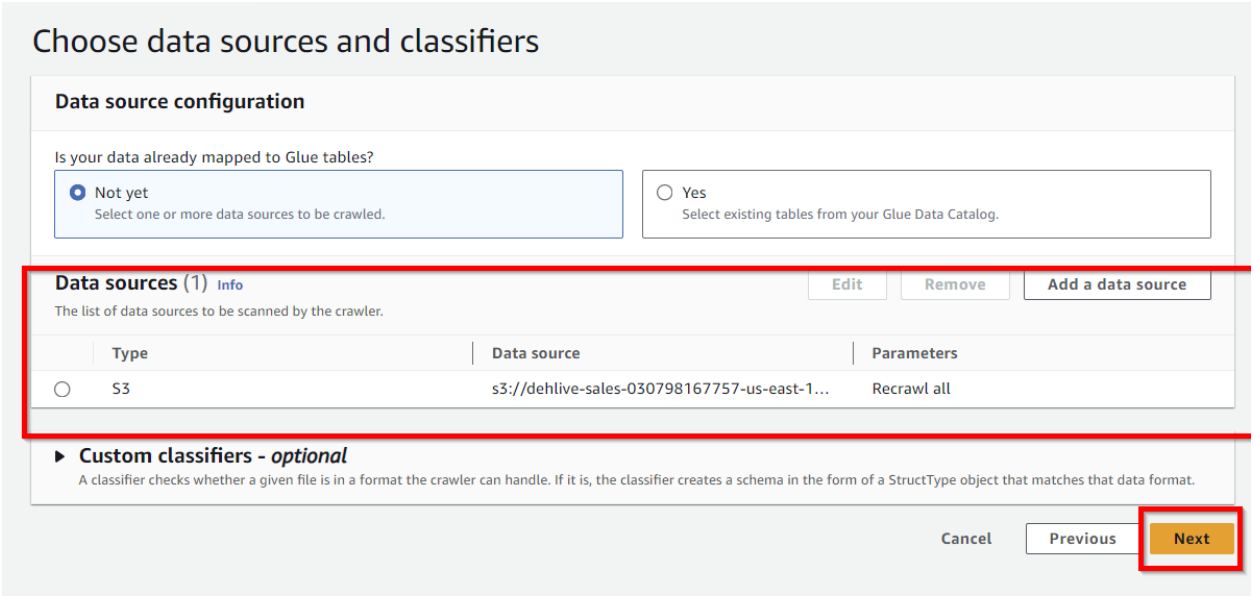
Now, we need to add a data source.

Enter just the path and not the file name.

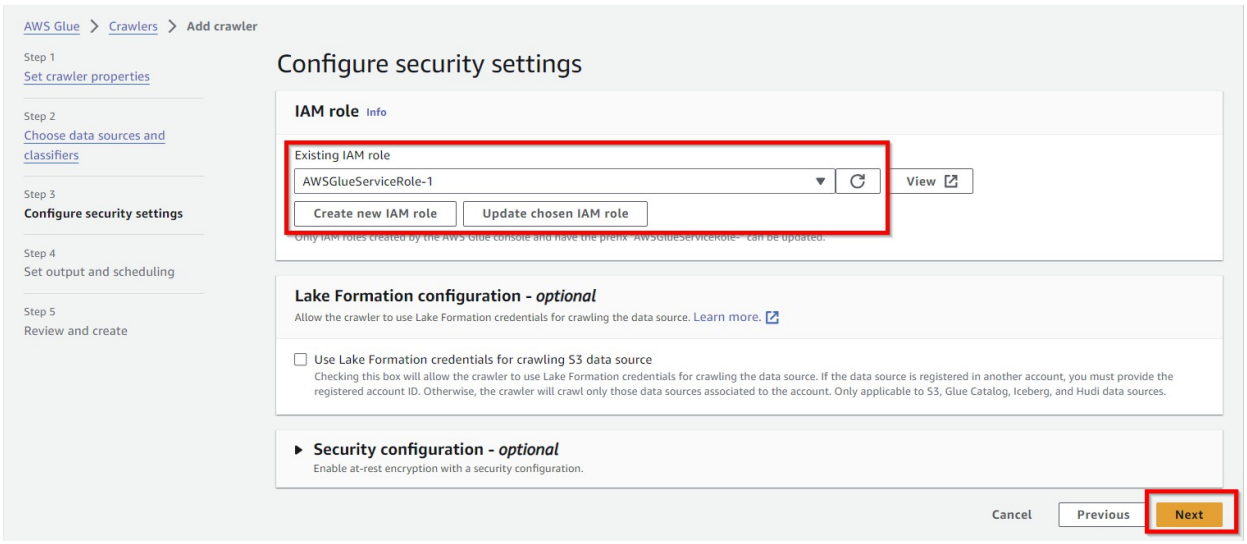
In my case it's s3://dehlive-sales-<your account number>-us-east-1/raw/maze/







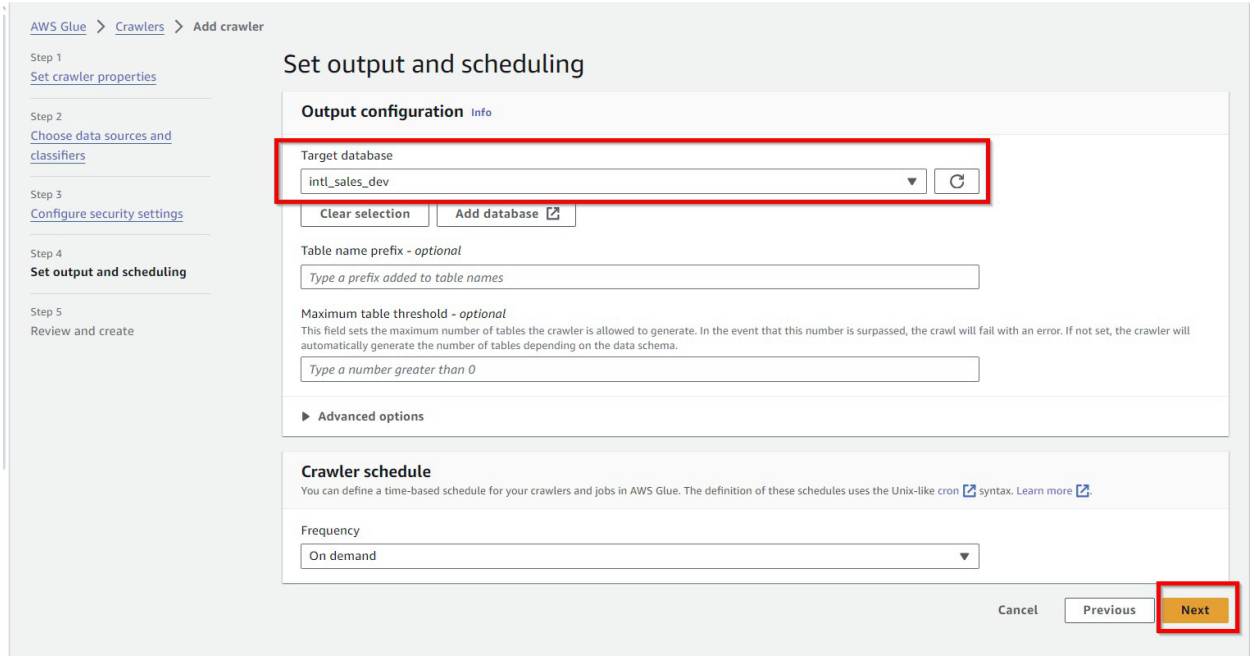
Next choose "your" Glue role that you created a few mins before and click next.



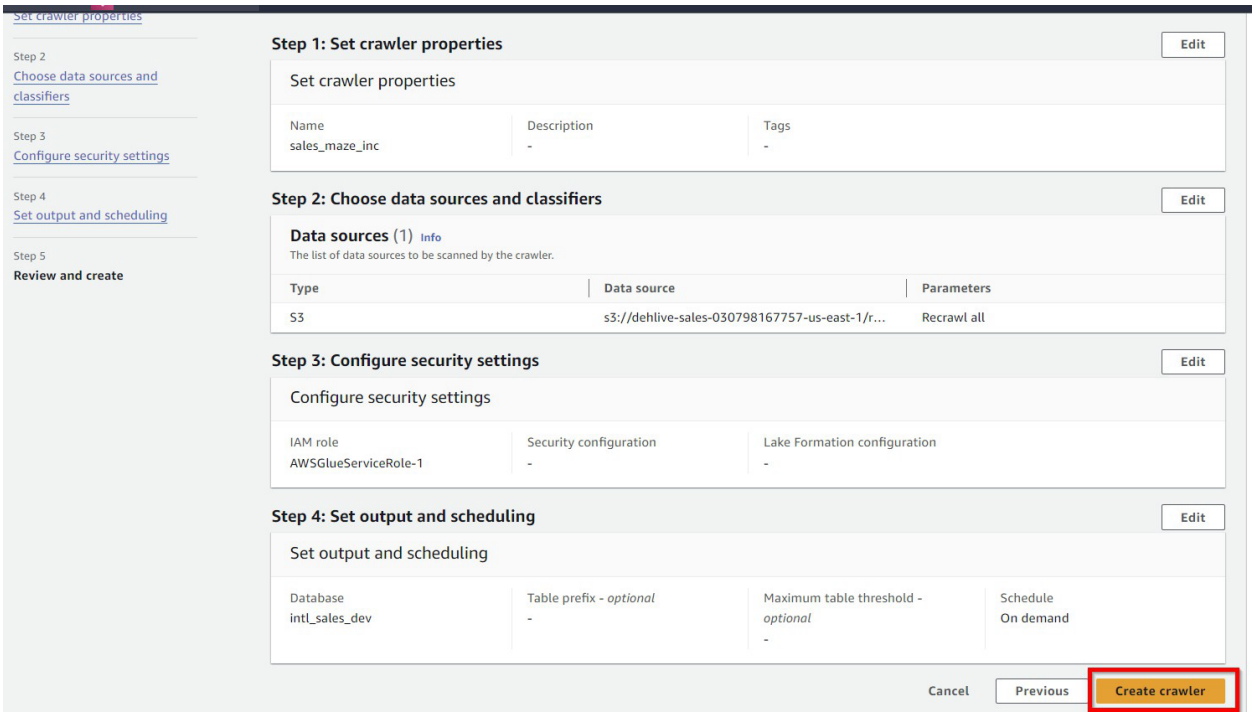
Sachin Chandrashekhar - Data Engineering Hub

🎯 Follow me on Twitter: https://x.com/AWSCloudSachin

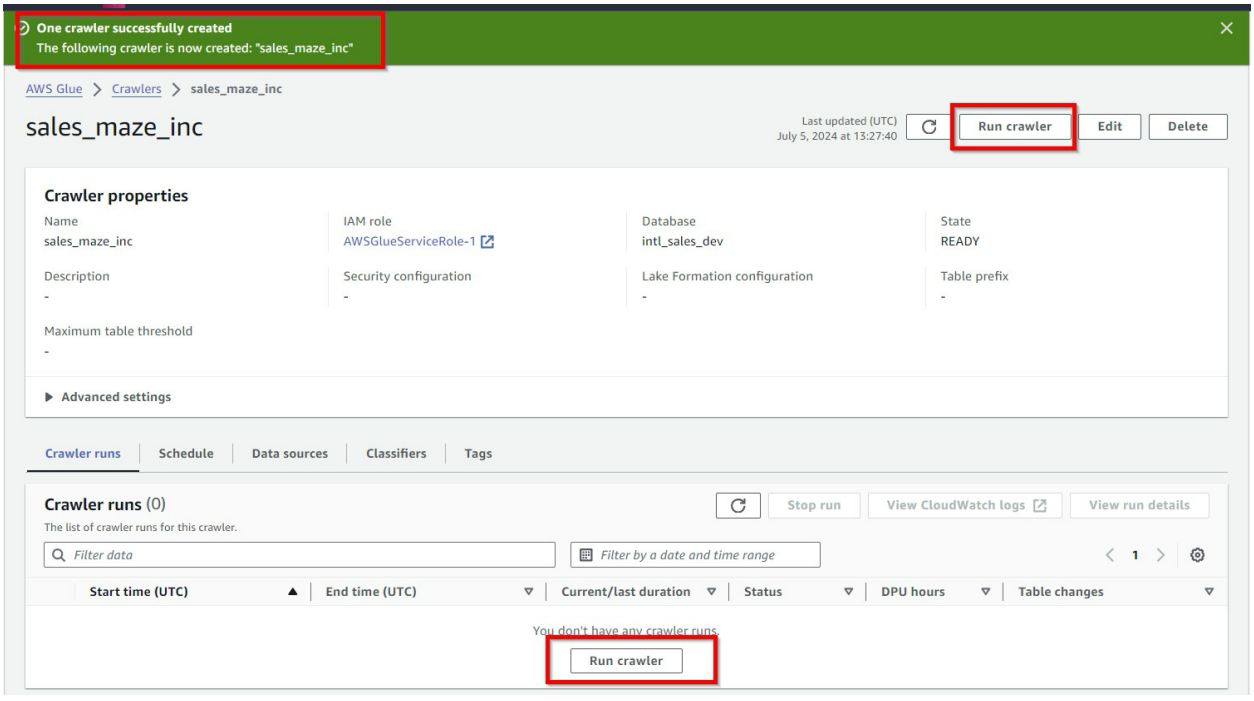
Next, choose the database you created before and click on next.



Finally review and create the crawler.

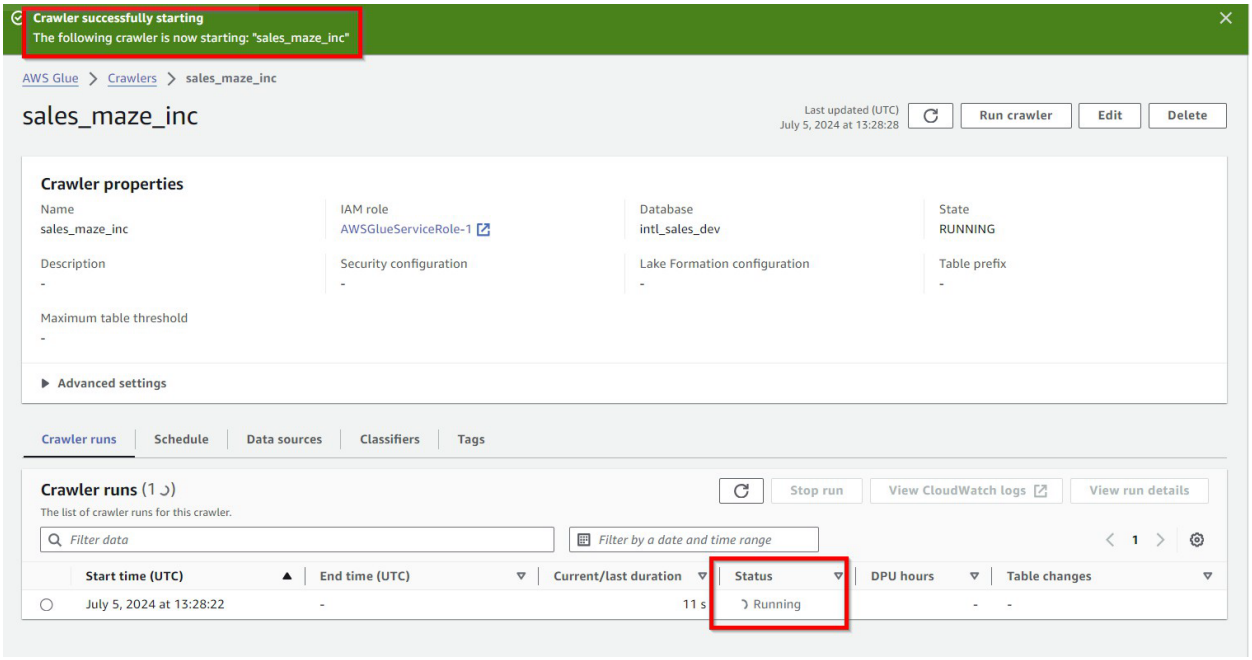


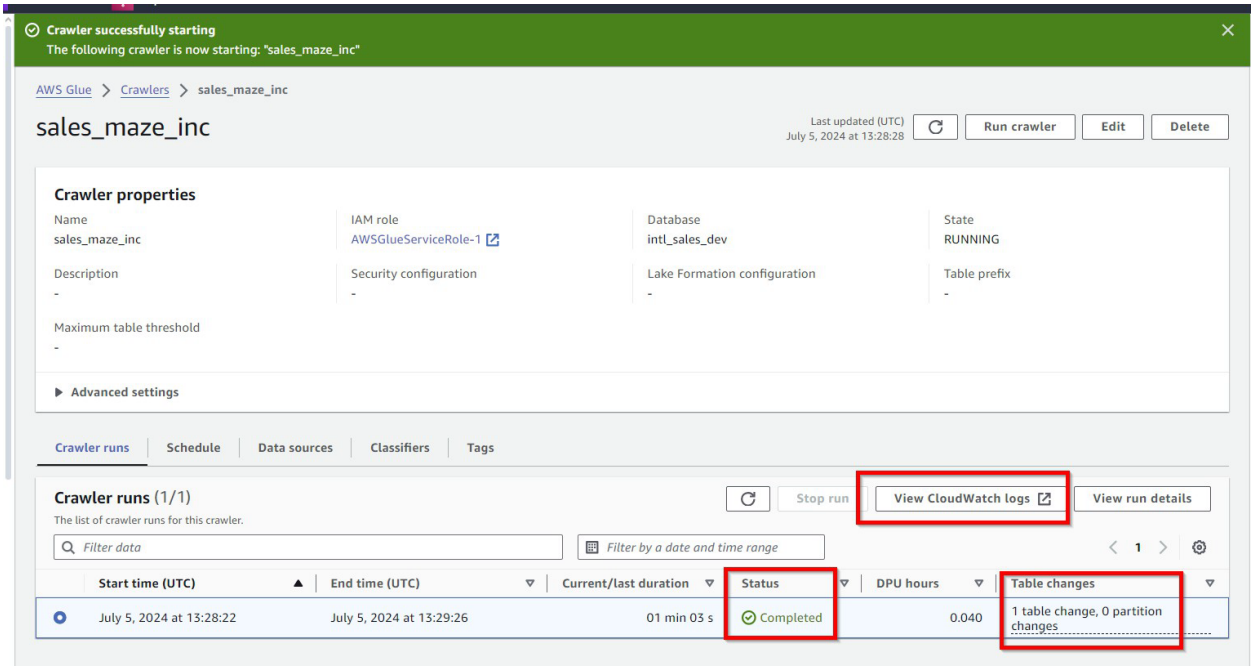
Now, run the crawler –



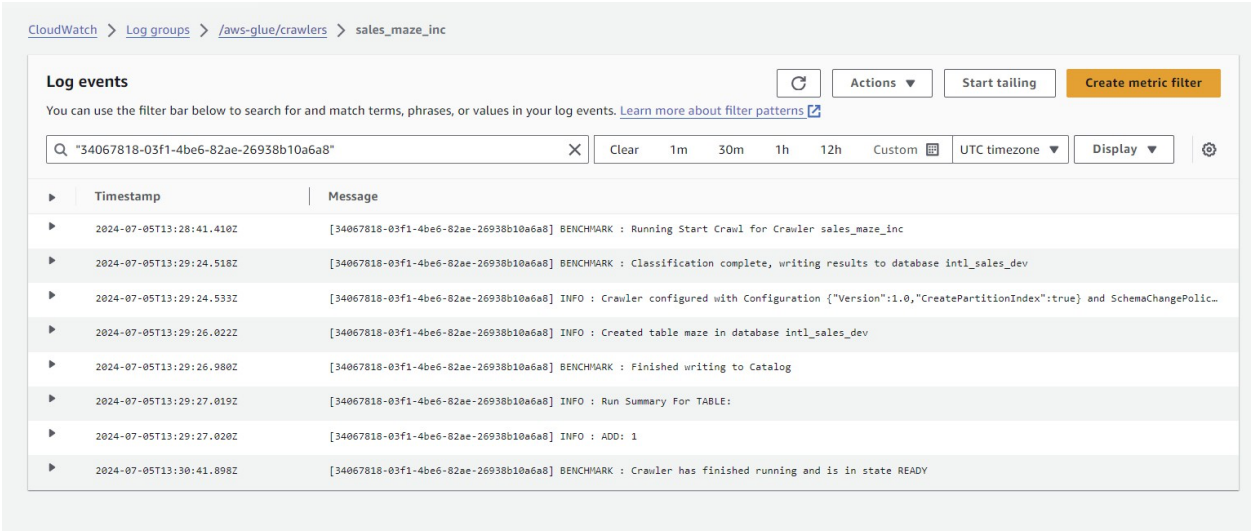
Sachin Chandrashekhar - Data Engineering Hub

🎯 Follow me on Twitter: https://x.com/AWSCloudSachin

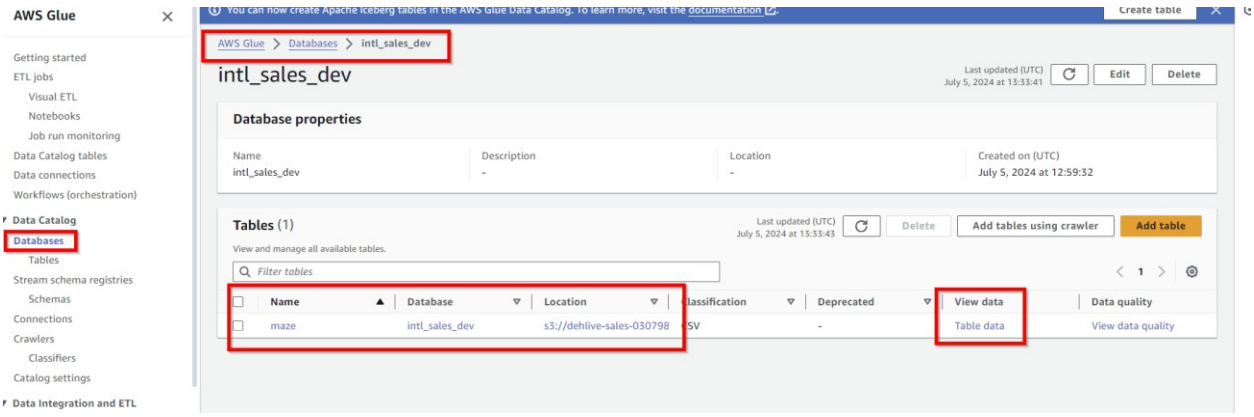




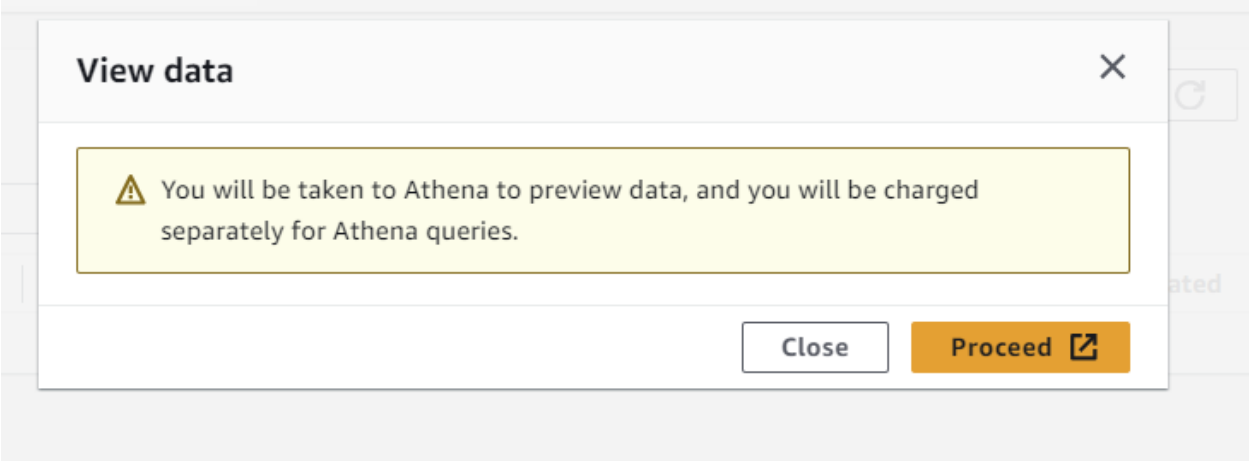
Click on View CloudWatch logs to see the logs –



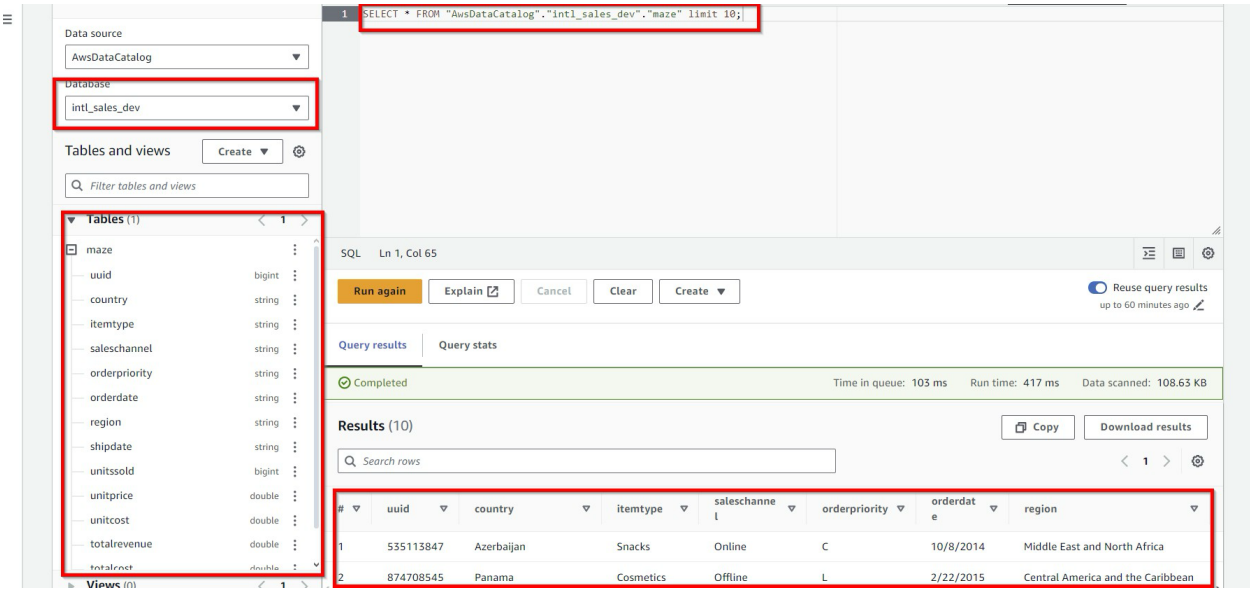
Now, go back to databases and check if table is created – and then click on table data –



Click on proceed below



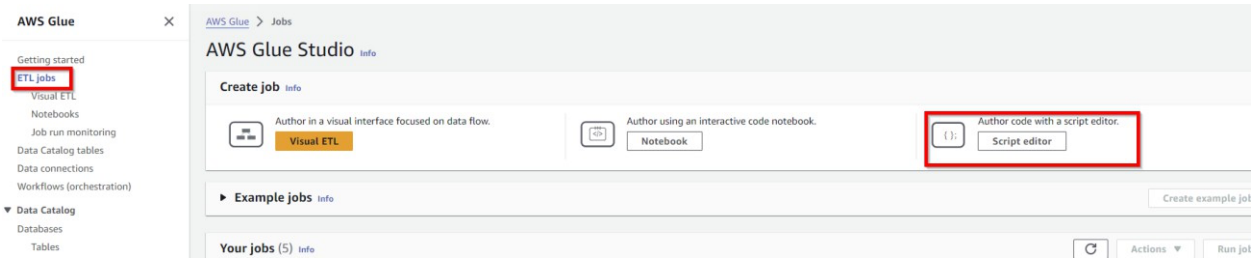
It takes to Athena where it has already fired the query to preview data and you can see the table created by choosing the database you created before. It also shows the data –

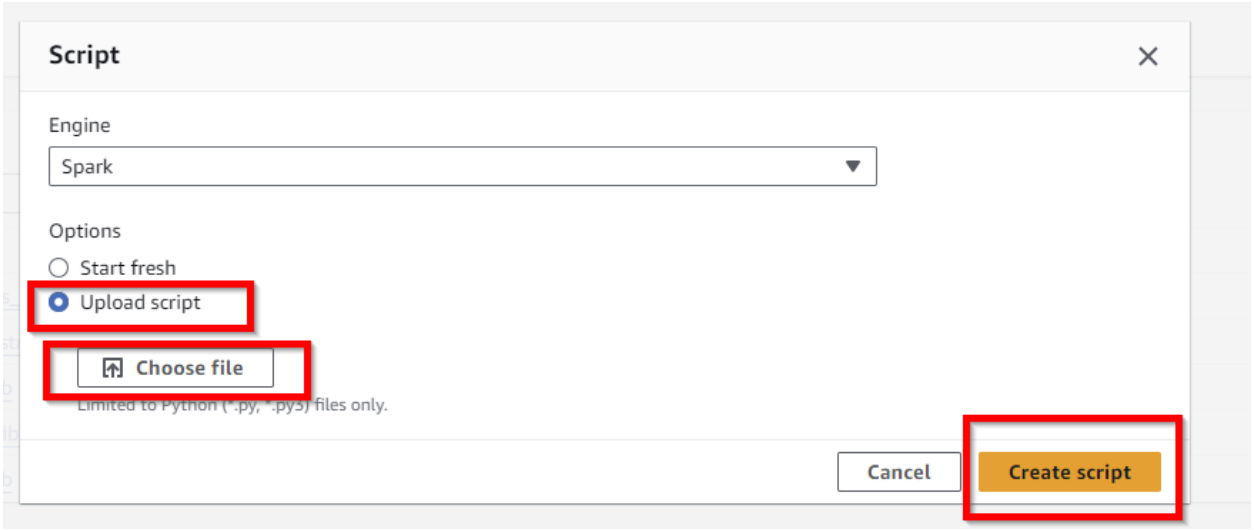


Now, since we have validated data looks good and we are able to query via data catalog object table "maze", it's time to create the Glue ETL script.

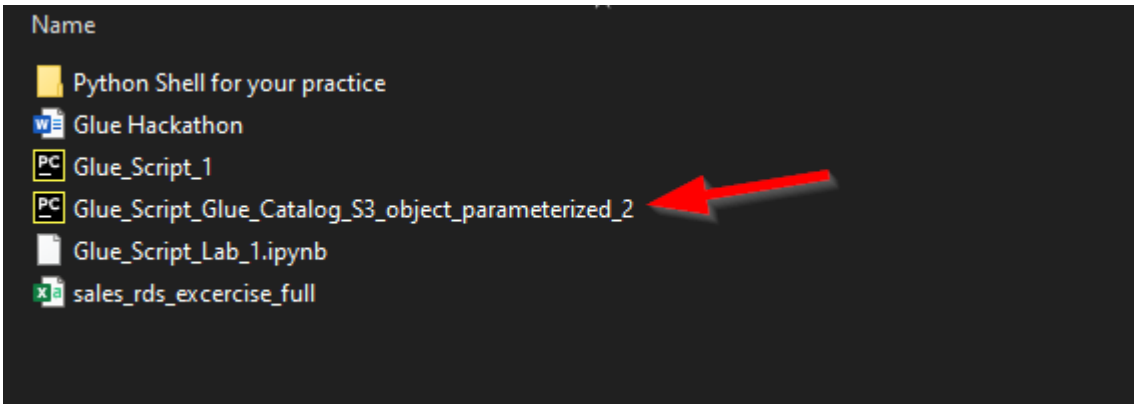
Btw, data is in S3 and we did not create table and then load data

We loaded data in data lake, then created table on top of the file. This is ELT and not ETL

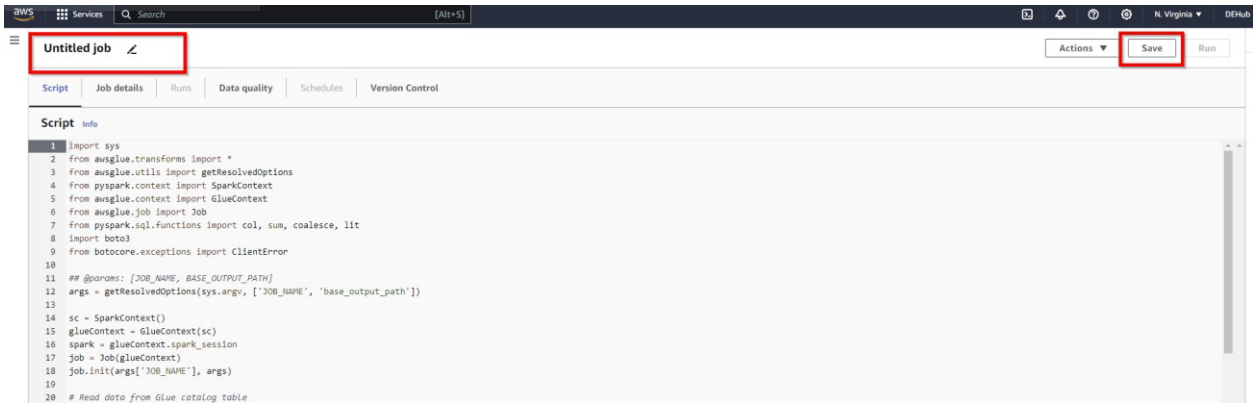




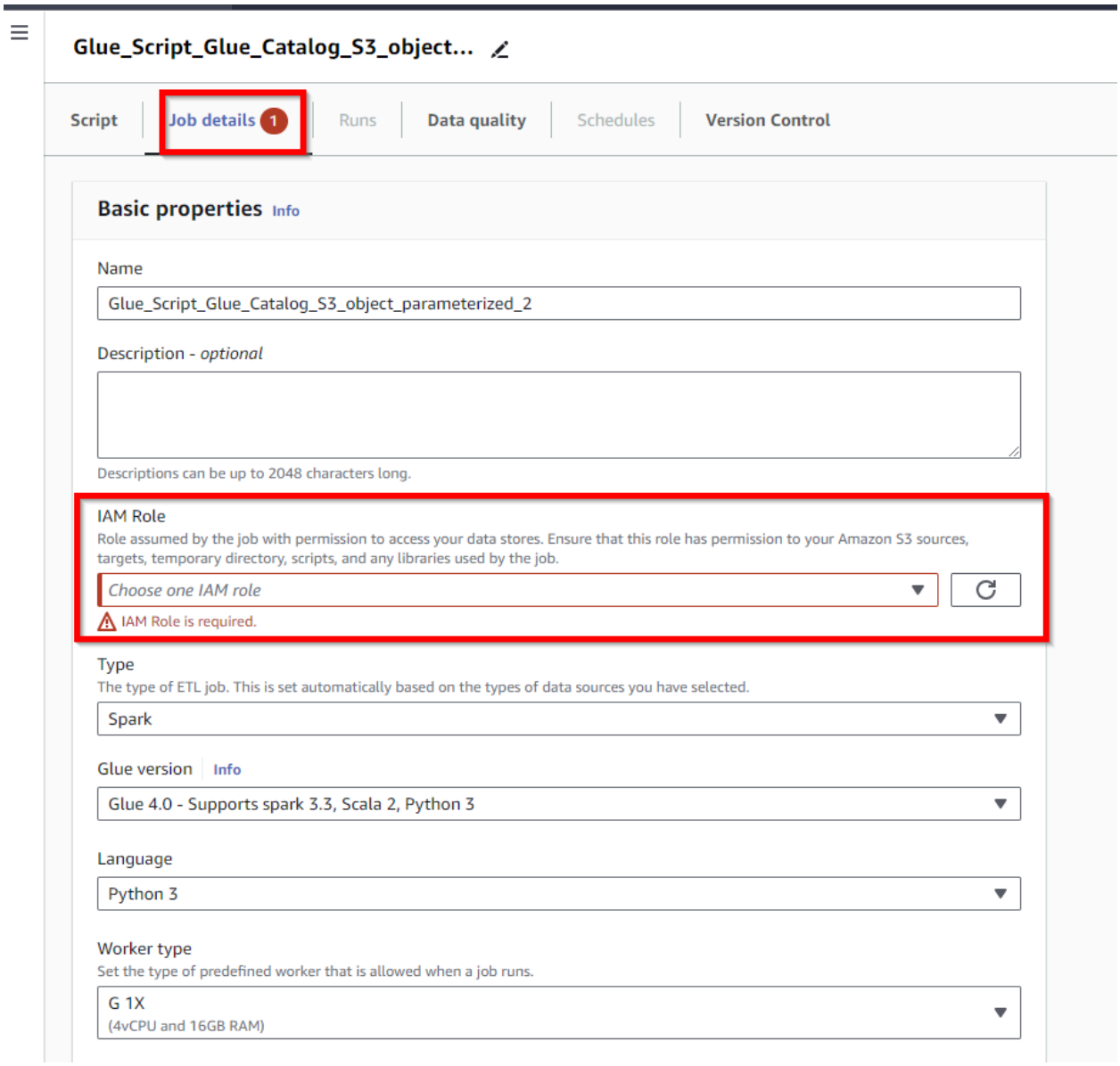
Use file Glue_Script_Glue_Catalog_S3_object_parameterized_2.py I shared.



Rename the job & Save.



Now choose the IAM role

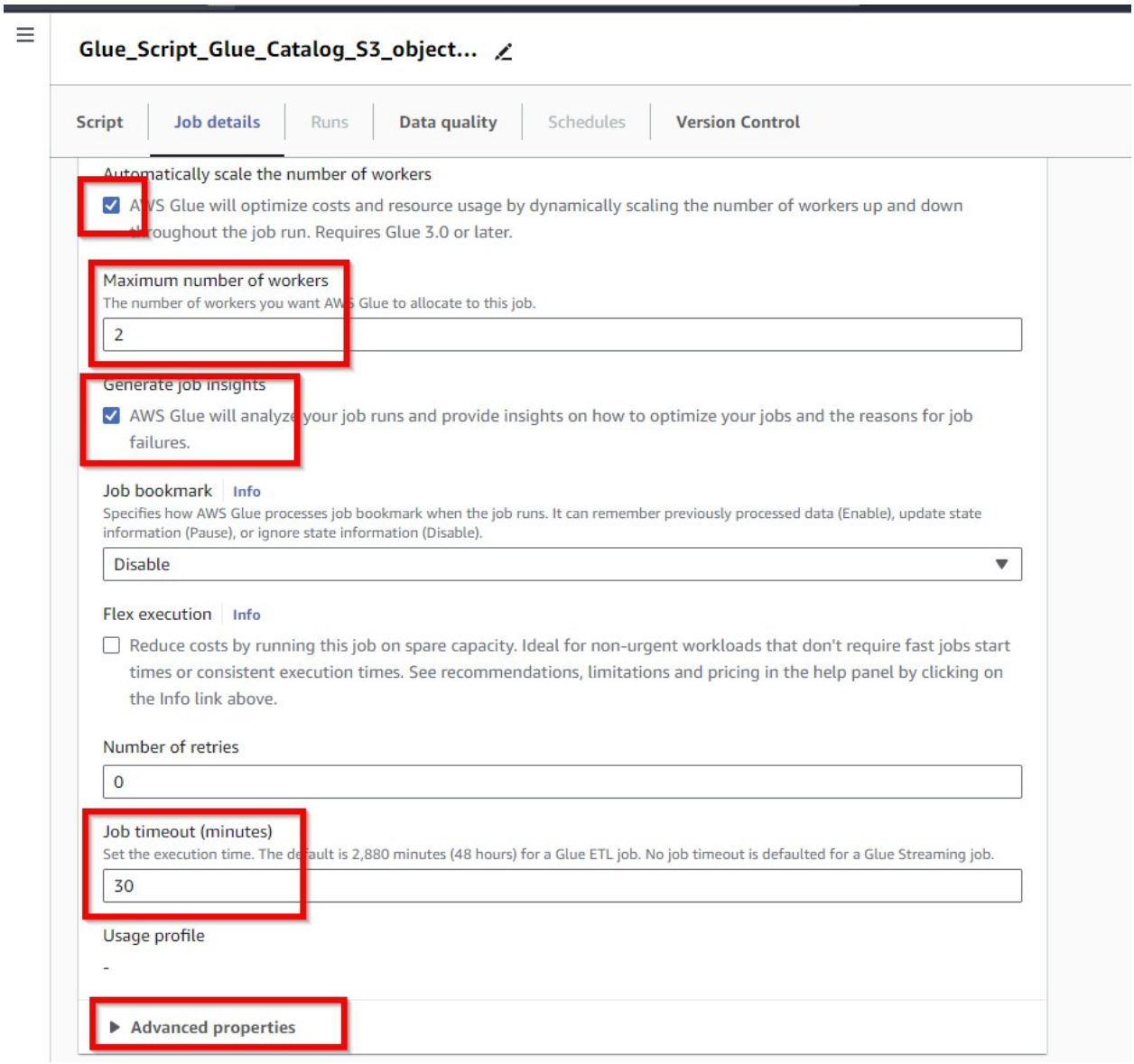


Next scroll down and choose these settings.

We need to reduce workers from 10 to 2 to save costs. Also, we don't need so many for the small data we have.

We can choose dynamic scaling of workers – Why? Interview Question - How do you save money with Glue – this way!

And though glue jobs can run for 48 hours, reduce it to 30 mins or the appropriate amount you know per your experience and size of data – so that you know if the job fails and doesn't continue running if there's any looping because of bad coding.



Lastly expand Advanced properties shown above and create a Job parameter "--curated_path"

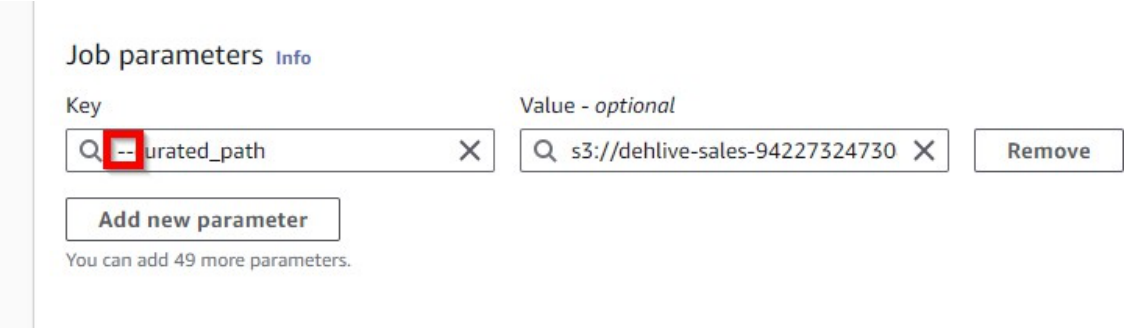
So, this is how we pass arguments to Glue job which can be referenced in the script.

Here, we pass the curated path to the job. Note the two hyphens -- to be added as a prefix to the parameter name.

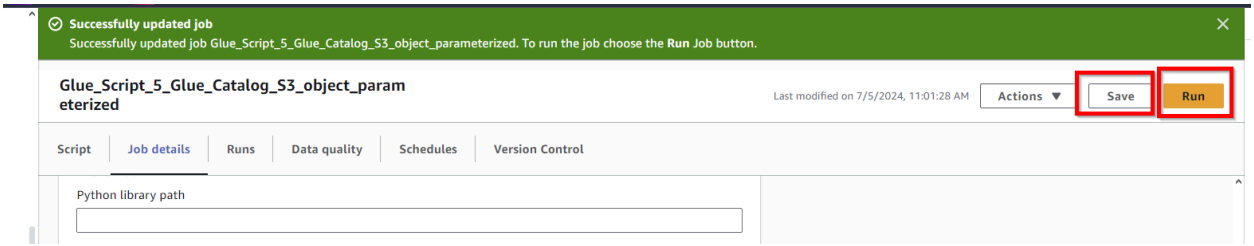
The value of parameter in my case is

s3://dehlive-sales-<youraccount number >-us-east-1/curated/maze

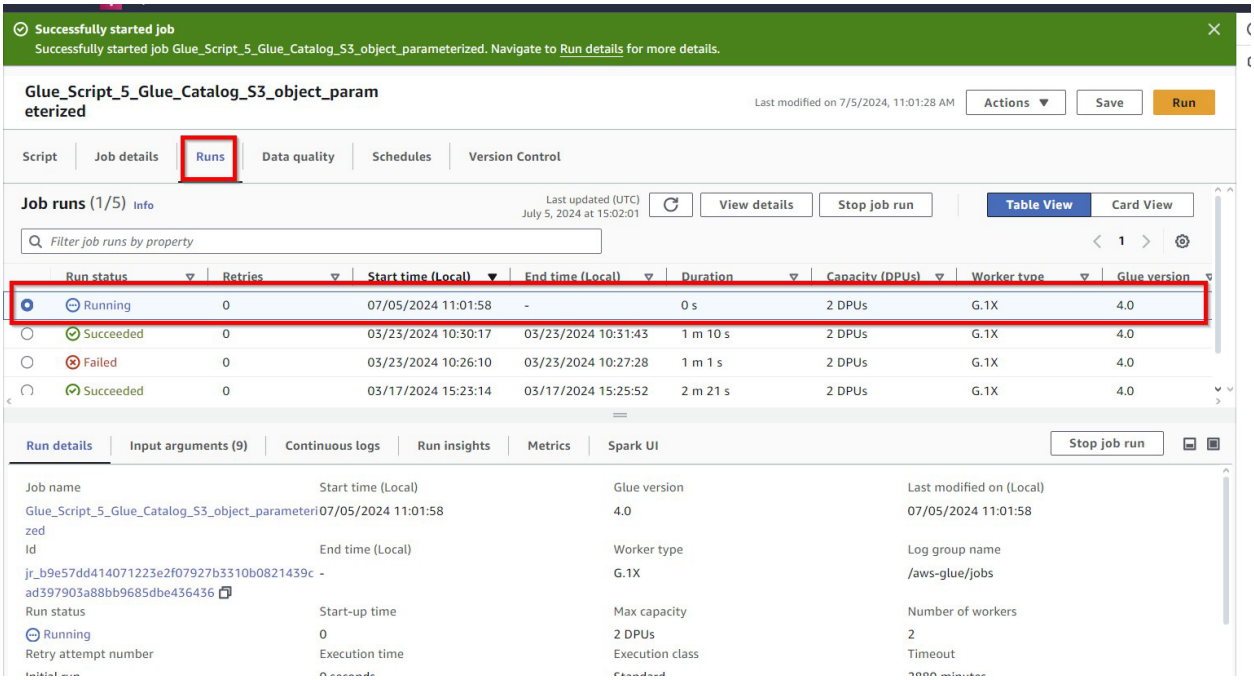
It looks like this.



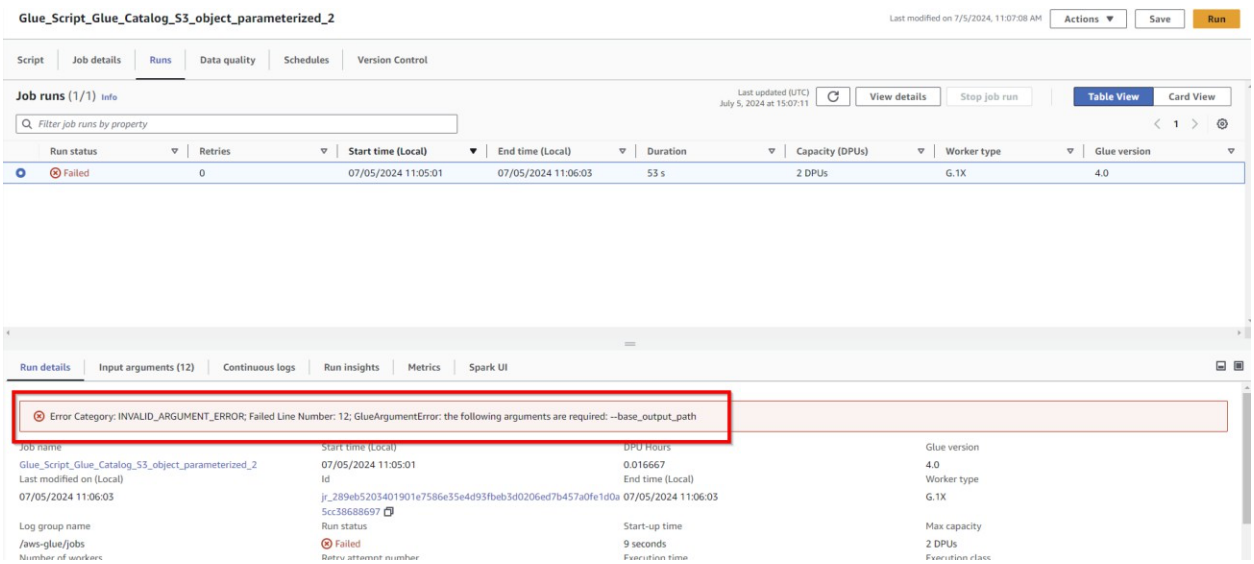
Finally save the job and run it.

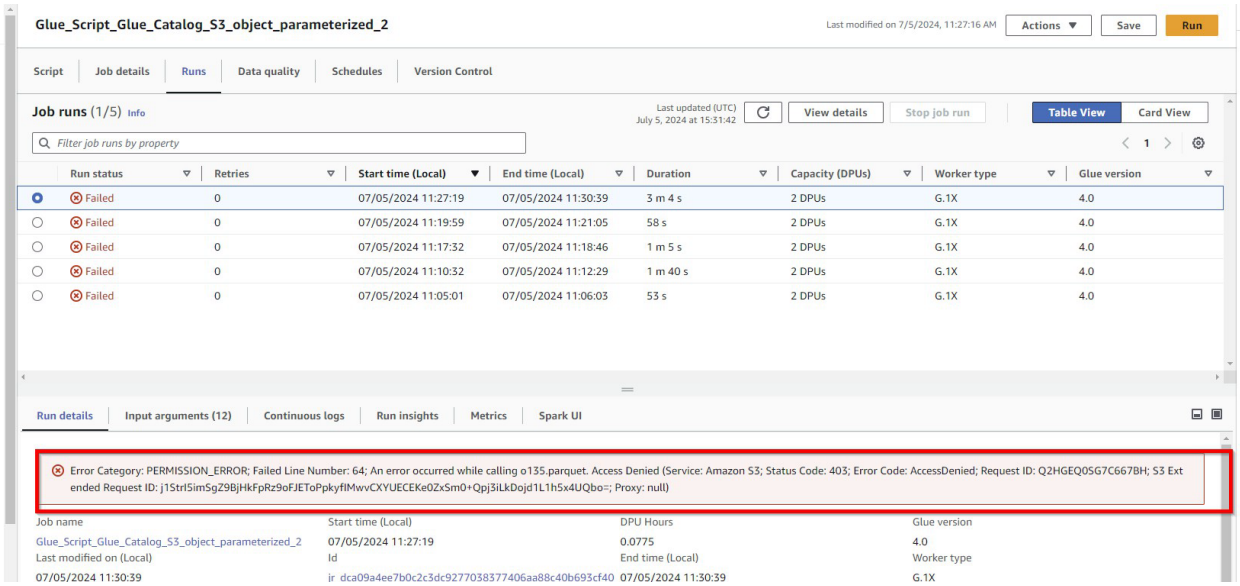


Go to Runs and you see it running.



If it fails, you will see the reason here –

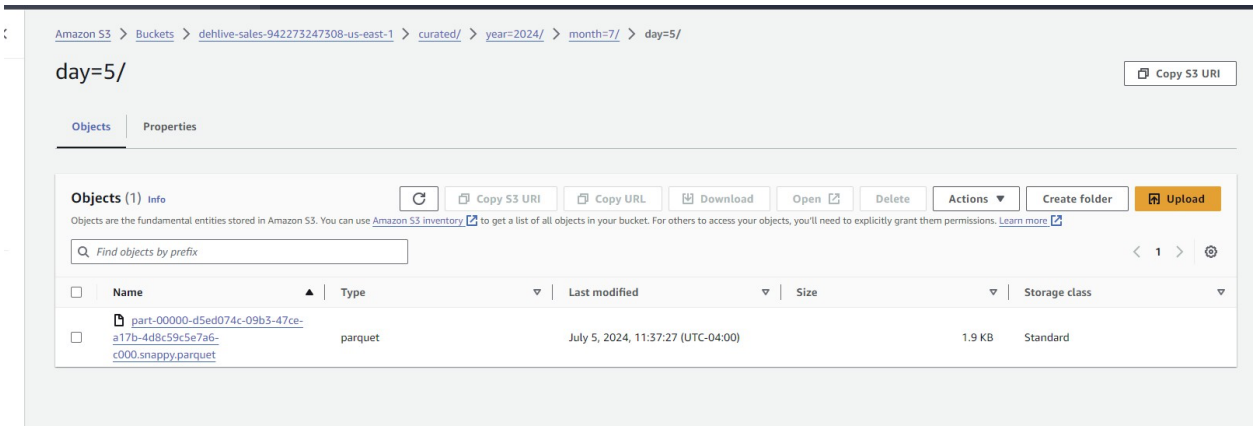




The above indicates S3 access is missing in the Glue role.

Add the S3 full access policy to the role and run again.

It goes thru and you get the curated data as below.



Again, what's our architecture for Glue Spark? As discussed in redshift hackathon, we get data into S3 and then for ETL, we can use Lambda, Glue Python Shell, ECS Fargate , Athena, EMR etc.

Here, we will use Glue pyspark to do the ETL and then load curated data back into S3 curated layer.

From s3 curated layer, you will then load data into Redshift table using COPY command in a stored procedure that you saw in the redshift hackathon.

Orchestration will be like this –

Data comes into S3, S3 event triggers Lambda which inturn triggers step function.

Step function triggers Glue Spark job, then after this completes, next step in step function triggers Redshift Stored proc that runs the COPY command which loads the curated data from S3 into the redshift table.

Now,

There are 4-5 things more that you can try out.

Go through my glue self-paced section and try out the following things

1) Incremental processing of data using bookmarks
2) Glue python shell jobs- you have already done hands on using this before
3) Glue Visual ETL – optional – also in the below mentioned resource.
4) Glue workflows – optional – this is not step-functions. – also, in the below mentioned resource.

Lastly, this is a very good resource – takes about 10-12 hours but is well worth the grind.

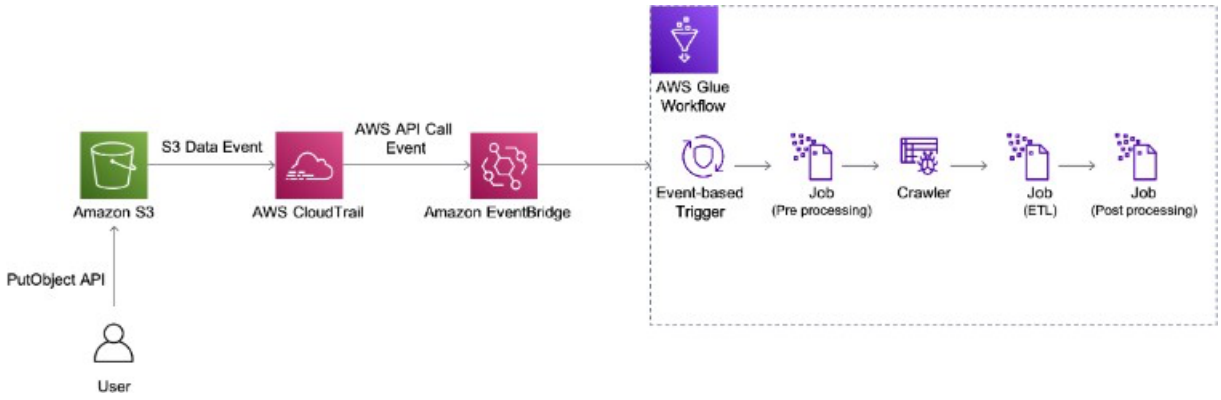
AWS Glue Workshop by AWS

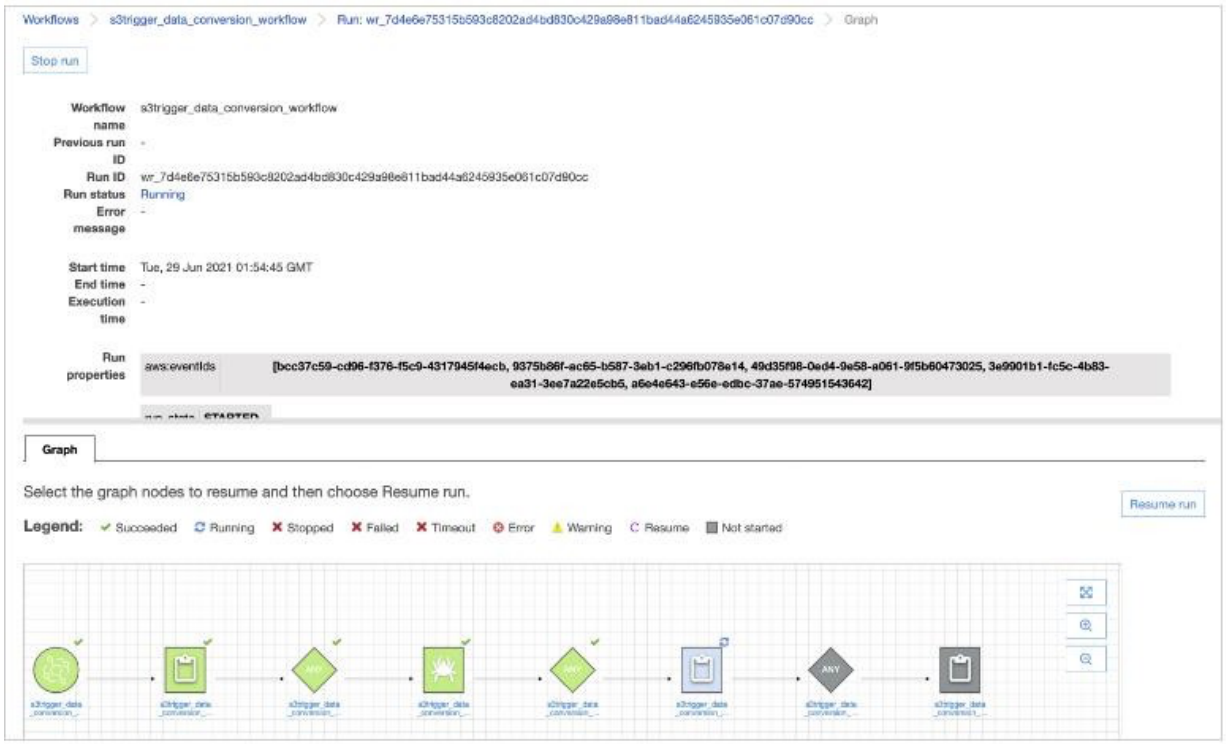
https://catalog.us-east-1.prod.workshops.aws/workshops/ee59d21b-4cb8-4b3d-a629-24537cf37bb5/en-US

Optional if you want to practice Glue Workflows – Why optional? We already know Step functions and Airflow by now and Even though some companies use Glue workflows, the functionality is limited – you can only use glue crawlers and glue jobs in the workflow – you cannot combine these with lambda or redshift or athena, in Glue Workflows. etc. Hence I would say skip it – some companies do use this though.

This uses **Glue Workflows,** Glue crawler, Glue ETL and eventbridge.

https://aws.amazon.com/blogs/big-data/build-a-serverless-event-driven-workflow-with-aws-glue-and-amazon-eventbridge/





I highly recommend trying this architecture out –

S3 events trigger -> eventbridge Rule -> step function -> Glue Job + Redshift + SNS

This could be your capstone project.

- Create an end to end project.
- Create CDK code and deploy
- Upload in Git
- Create a LinkedIn or Medium.com Article.
- Tag me ☺
- Always promote yourself everywhere!

Also, this can go into your resume for sure. – you can say I implemented this framework in my company.

https://aws.amazon.com/blogs/big-data/identify-source-schema-changes-using-aws-glue/

If you are still hungry for more, this gets you hands on with another service called Appflow (used to pull data from SaaS applications like JIRA, Salesforce, Service now) and then also makes use of Glue Databrew (which I have not taught – I am sure you can figure this one out)

https://aws.amazon.com/blogs/big-data/empower-your-jira-data-in-a-data-lake-with-amazon-appflow-and-aws-glue/

Curated by: Sachin Chandrashekhar

Founder – Data Engineering Hub

🎯 Follow me on Twitter: https://x.com/AWSCloudSachin