



RADE™ Glue Hackathon

www.sachinchandrashekhar.com

Sachin Chandrashekhar - Data Engineering Hub

👉 Follow me on Twitter: <https://x.com/AWSCloudSachin>

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 anywhere 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.

Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>

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

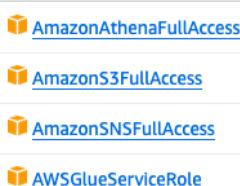
- 1) Truncate stage table
- 2) COPY S3 data into stage table
- 3) Upsert Stage to core table

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.

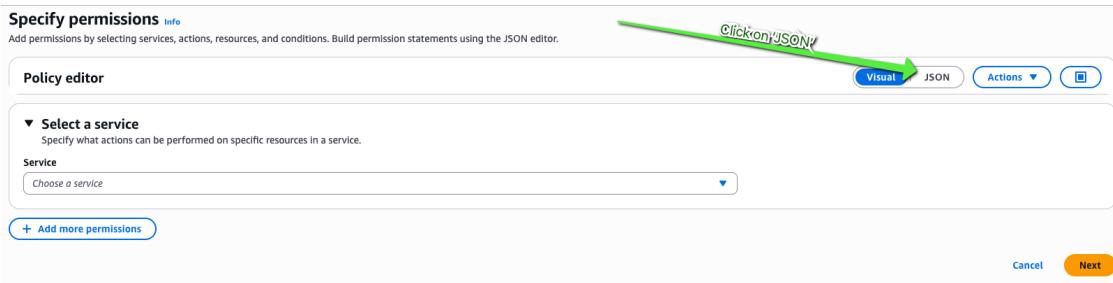
The screenshot shows the AWS IAM Role Permissions page for the "AWSGlueServiceRole". The "Permissions" tab is selected. In the top right corner, there is a green arrow pointing to the "Create inline policy" button. The "Permissions policies" section shows four managed policies attached to the role. The "Attached entities" table lists the number of entities each policy is attached to: 6 for AmazonAthenaFullAccess, 16 for AmazonS3FullAccess, 9 for AmazonSNSFullAccess, and 7 for AWSGlueServiceRole.

Attached entities
6
16
9
7

Sachin Chandrashekhar - Data Engineering Hub

Follow me on Twitter: <https://x.com/AWSCloudSachin>

Next,



Specify permissions Info

Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.

Policy editor

▼ **Select a service**
Specify what actions can be performed on specific resources in a service.

Service

+ Add more permissions

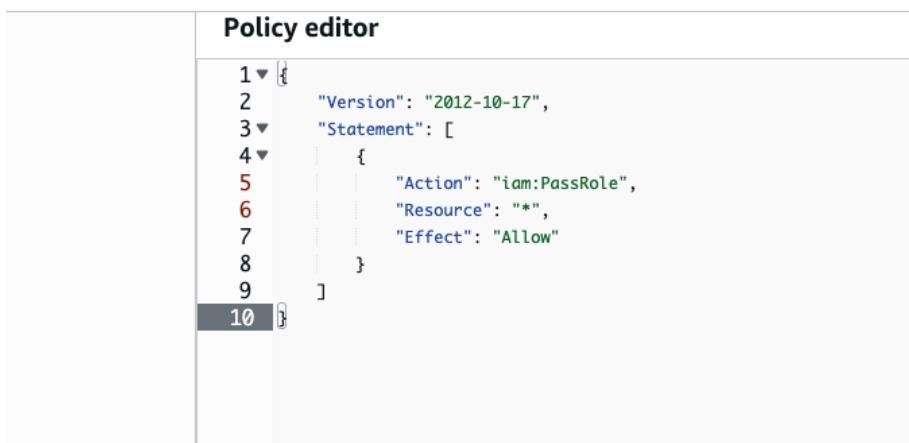
Visual JSON Actions ▾

Copy the below

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

Paste the above below:

· [AWSGlueServiceRole](#) > Create policy



Policy editor

```

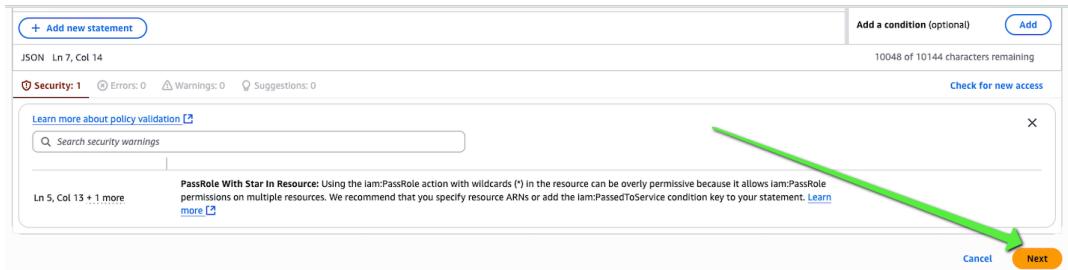
1  {
2      "Version": "2012-10-17",
3      "Statement": [
4          {
5              "Action": "iam:PassRole",
6              "Resource": "*",
7              "Effect": "Allow"
8          }
9      ]
10 }
```

Sachin Chandrashekhar - Data Engineering Hub

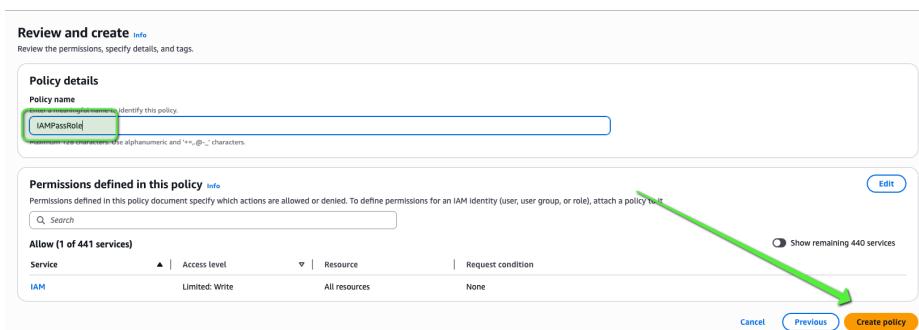
 Follow me on Twitter: <https://x.com/AWSCloudSachin>



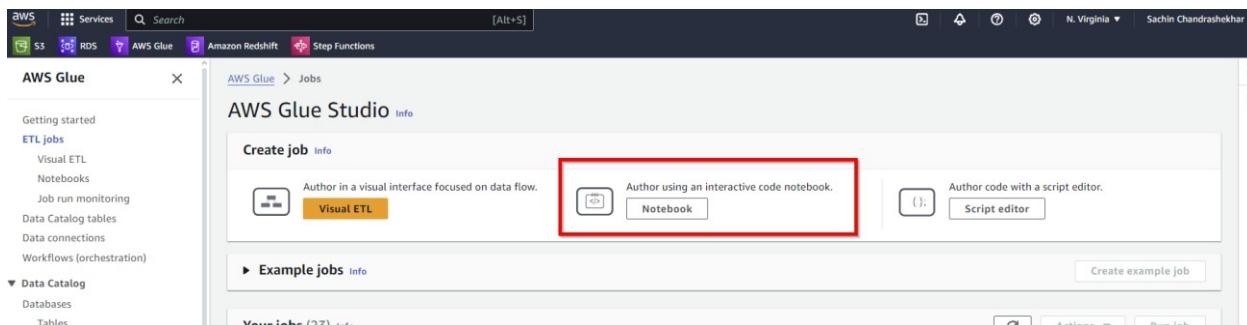
Click on Next



Provide name and create



Let's NOW start with Glue Interactive Notebook

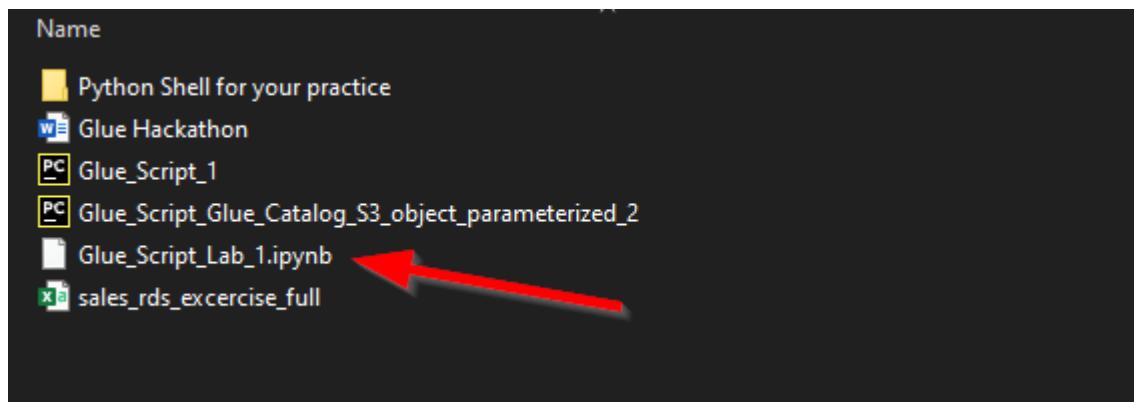


Use the Notebook provided Glue_Script_Lab_1.ipynb to upload for your practice.

Not For Distribution ©Sachin Chandrashekhar www.dataengineeringhub.in

Sachin Chandrashekhar - Data Engineering Hub

Follow me on Twitter: <https://x.com/AWSCloudSachin>



The screenshot shows the 'Create notebook' dialog in AWS Glue Studio. The 'Notebook' tab is selected. The dialog includes the following fields:

- Options:** Radio buttons for "Start fresh" and "Upload Notebook". The "Upload Notebook" option is selected and highlighted with a red box.
- Choose file:** A button to select a file, also highlighted with a red box. Below it, a note says "Limited to Jupyter Notebook (*.ipynb) files only."
- IAM role:** A dropdown menu set to "AWSGlueServiceRole-1", which is also highlighted with a red box.
- Information message:** A box containing the text: "You can now use natural language to author jobs or ask questions in AWS Glue Studio Notebook. To learn more, visit the [documentation](#)." This box is also highlighted with a red border.
- Buttons:** "Cancel" and "Create notebook" at the bottom right.

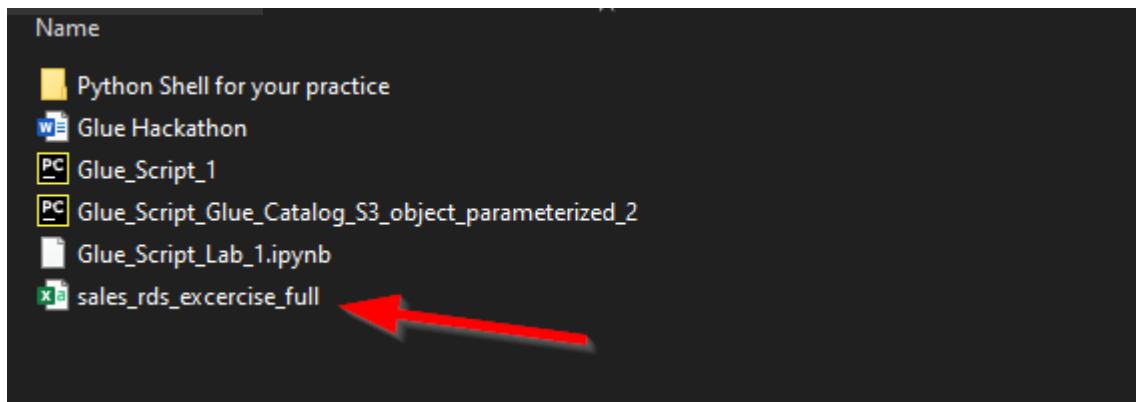
We will use the Sales file we have always been using – Make sure it is uploaded in your bucket.

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



Amazon S3 > Buckets > dehlive-sales-030798167757-us-east-1 > raw/ > maze/

maze/

Objects Properties

Objects (1) Info

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 ob

Find objects by prefix

<input type="checkbox"/>	Name	Type	Last modified
<input type="checkbox"/>	sales_rds_exercise_full.csv	csv	September 16, 2024, 05:07:15 (UTC-04:00)

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>

The screenshot shows the AWS Glue Studio Notebook interface. At the top, there's a navigation bar with 'Services' and various AWS services like S3, RDS, AWS Glue, Amazon Redshift, and Step Functions. Below the navigation bar is a toolbar with 'Stop notebook' (highlighted with a red box), 'Download Notebook', 'Actions', 'Save', and 'Run'. The main area is titled 'AWS Glue Studio Notebook' and contains a help section with magic commands like %help, boilerplate code for setting up a Spark session, and an example of creating a DynamicFrame. The code editor shows a single cell with the path 's3://aws-glue-bootcamp-030798167757-us-east-1/raw/sales/input/sales_rds_exercise.csv' highlighted with a red box.

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

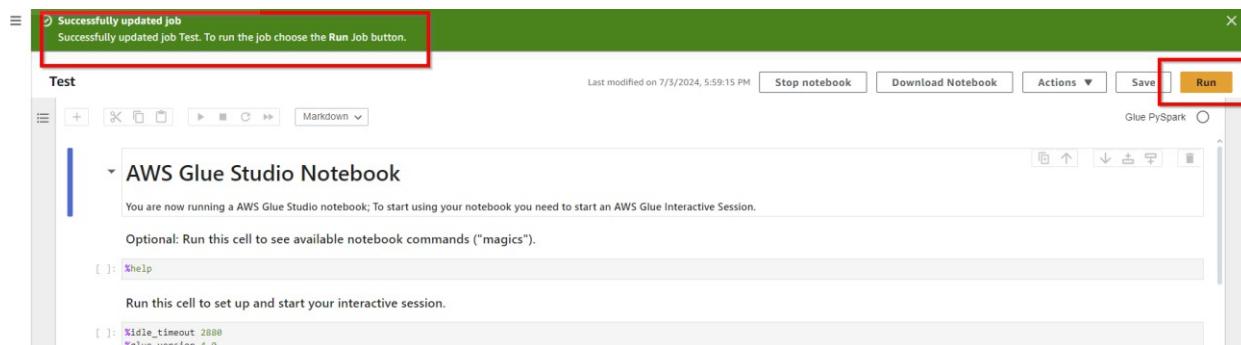
Reference the file path to update the notebook code.

The screenshot shows the AWS Glue Studio Notebook code editor. A cell in the code editor has the file path 's3://aws-glue-bootcamp-030798167757-us-east-1/raw/sales/input/sales_rds_exercise.csv' highlighted with a red box. The code also includes imports for spark, sparkContext, and glueContext.

Renamed as shown below and “Save” the job.

The screenshot shows the AWS Glue Studio Notebook interface with the 'Test' tab selected. At the top, there's a toolbar with 'Stop notebook', 'Download Notebook', 'Actions', 'Save' (highlighted with a red box), and 'Run'. The main area is titled 'AWS Glue Studio Notebook' and contains a help section with magic commands like %help, boilerplate code for setting up a Spark session, and an example of creating a DynamicFrame. The code editor shows a single cell with the path 's3://aws-glue-bootcamp-030798167757-us-east-1/raw/sales/input/sales_rds_exercise.csv' highlighted with a red box.

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



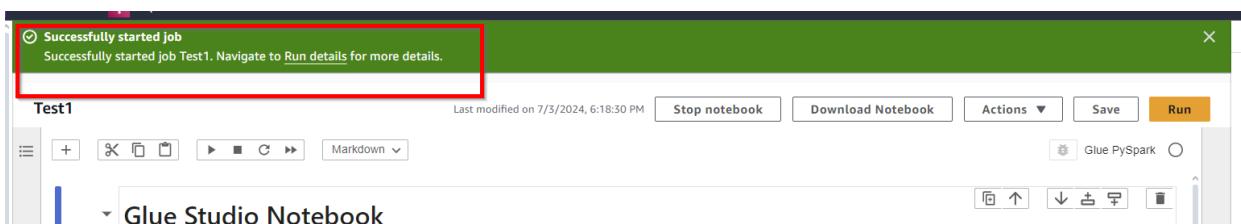
Successfully updated job
Successfully updated job Test. To run the job choose the Run Job button.

Last modified on 7/3/2024, 5:59:15 PM Stop notebook Download Notebook Actions Save Run

AWS Glue Studio Notebook
You are now running a AWS Glue Studio notebook; To start using your notebook you need to start an AWS Glue Interactive Session.
Optional: Run this cell to see available notebook commands ("magics").
[]: %%help
Run this cell to set up and start your interactive session.
[]: %idle_timeout 2880
glue version 4.0

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

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

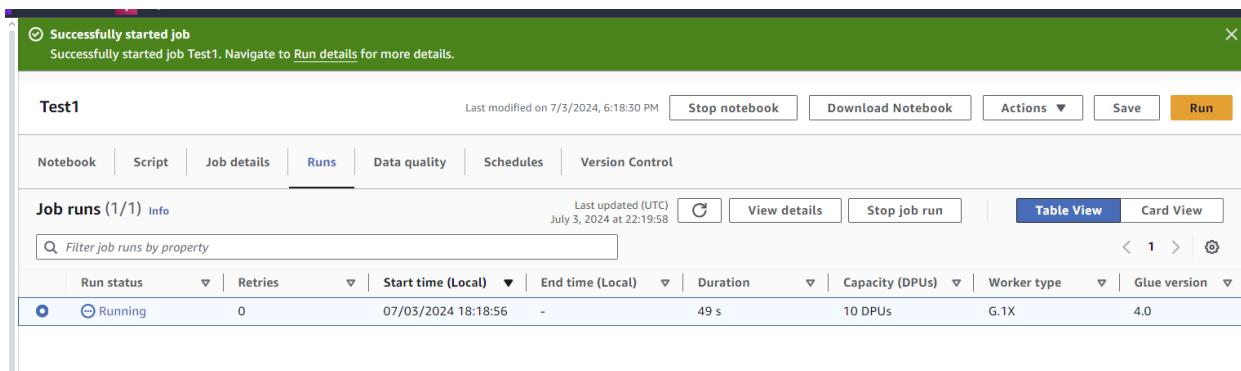


Successfully started job
Successfully started job Test1. Navigate to Run details for more details.

Last modified on 7/3/2024, 6:18:30 PM Stop notebook Download Notebook Actions Save Run

Glue Studio Notebook

Run Details:



Successfully started job
Successfully started job Test1. Navigate to Run details for more details.

Last modified on 7/3/2024, 6:18:30 PM Stop notebook Download Notebook Actions Save Run

Test1

Notebook Script Job details **Runs** Data quality Schedules Version Control

Job runs (1/1) info Last updated (UTC)
July 3, 2024 at 22:19:58

Run status	Retries	Start time (Local)	End time (Local)	Duration	Capacity (DPUs)	Worker type	Glue version
Running	0	07/03/2024 18:18:56	-	49 s	10 DPUs	G.1X	4.0

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

Sachin Chandrashekhar - Data Engineering Hub

Follow me on Twitter: <https://x.com/AWSCloudSachin>

Test1

Last modified on 7/3/2024, 6:18:30 PM

Runs

Job runs (1/1) Info Last updated (UTC) July 3, 2024 at 22:22:37

Run status	Retries	Start time (Local)	End time (Local)	Duration	Capacity (DPU)	Worker type	Glue version
Failed	0	07/03/2024 18:18:56	07/03/2024 18:22:21	3 m 12 s	10 DPU	G.1X	4.0

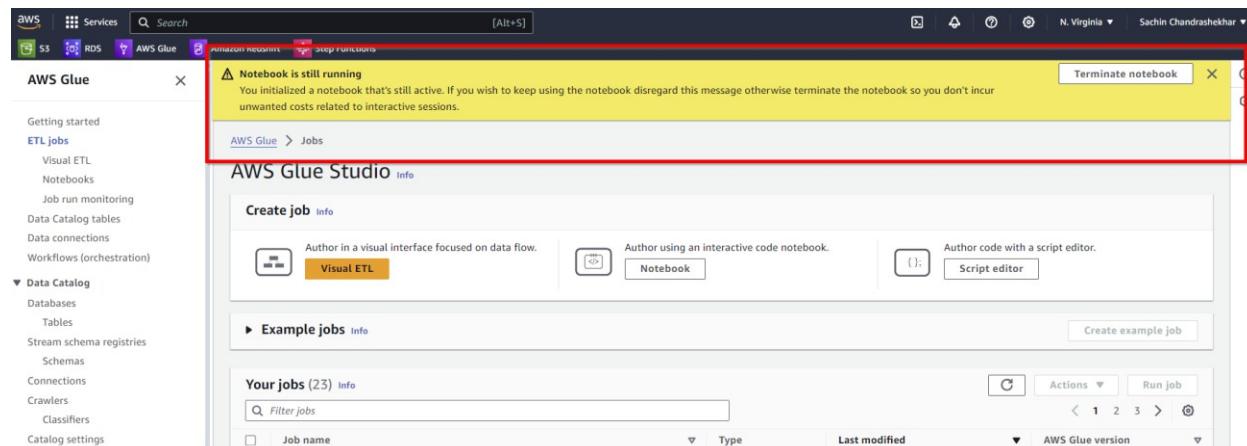
Run details

AnalysisException: path s3://aws-glue-bootcamp-030798167757-us-east-1/raw/sales/parquetresult already exists.

Job name	Start time (Local)	Glue version	Last modified on (Local)
Test1	07/03/2024 18:18:56	4.0	07/03/2024 18:22:21
Id	End time (Local)	Worker type	Log group name
jr_1c6cb4836141995a285bba1e68bc1d58ed825d407/03/2024 18:22:21		G.1X	/aws-glue/jobs
52c4ab4e4931ebb49fe4425ac			
Run status	Start-up time	Max capacity	Number of workers
Failed	12 seconds	10 DPU	10

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.



Notebook is still running.
You initialized a notebook that's still active. If you wish to keep using the notebook disregard this message otherwise terminate the notebook so you don't incur unwanted costs related to interactive sessions.

AWS Glue Studio

Create job

Author in a visual interface focused on data flow. **Visual ETL**

Author using an interactive code notebook. **Notebook**

Author code with a script editor. **Script editor**

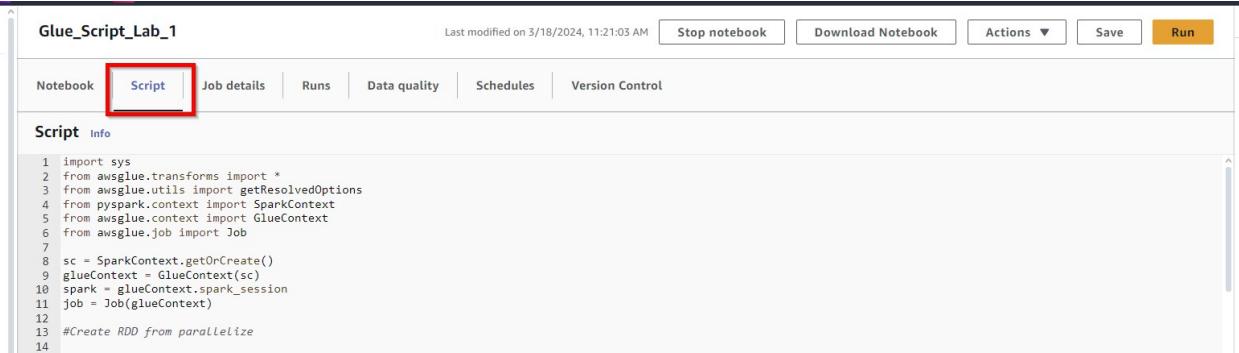
Your jobs (23)

Actions

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 screenshot shows the AWS Glue Script Lab 1 interface. At the top, there are tabs: Notebook, **Script**, Job details, Runs, Data quality, Schedules, and Version Control. The 'Script' tab is highlighted with a red box. Below the tabs, there is a section titled 'Script Info' containing the following Python code:

```

1 import sys
2 from awsglue.transforms import *
3 from awsglue.utils import getResolvedOptions
4 from pyspark.context import SparkContext
5 from awsglue.context import GlueContext
6 from awsglue.job import Job
7
8 sc = SparkContext.getOrCreate()
9 glueContext = GlueContext(sc)
10 spark = glueContext.spark_session
11 job = Job(glueContext)
12
13 #Create RDD from parallelize
14

```

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.

AWS Glue > Jobs

AWS Glue Studio Info

Create job Info

Author in a visual interface focused on data flow.

Visual ETL

Author using an interactive code notebook.

Notebook

Author code with a script editor.

{}; Script editor

Example jobs Info

Create example job

Script

Engine

Spark

Options

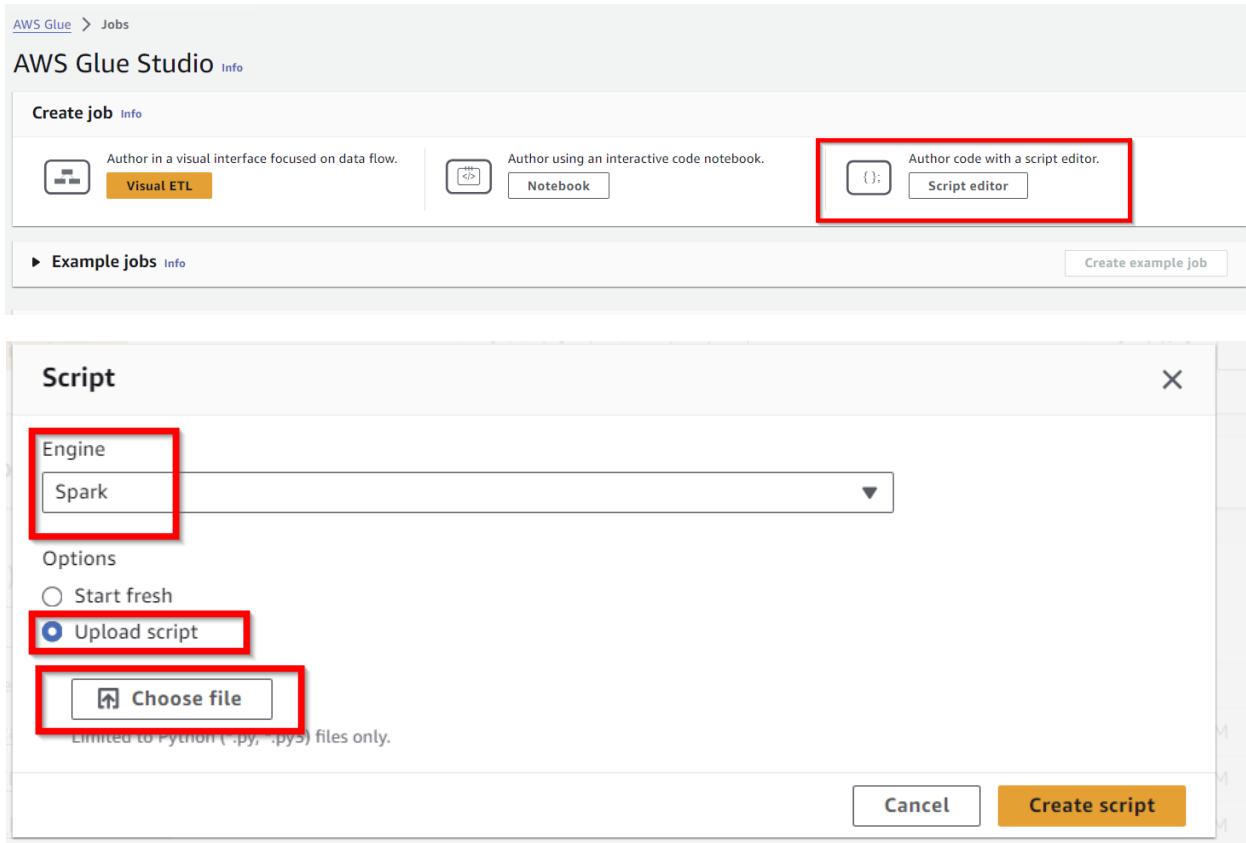
Start fresh

Upload script

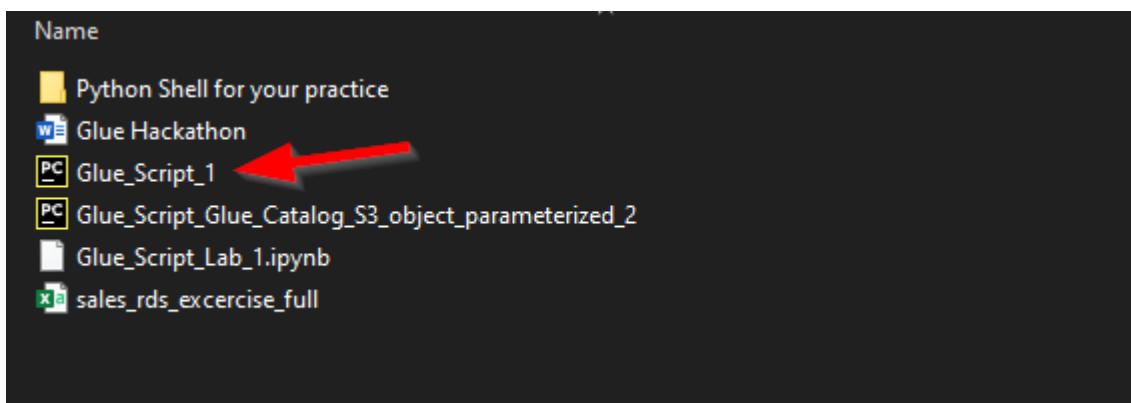
Choose file

Limited to Python (.py, .pyo) files only.

Cancel Create script



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.

Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>

Test 

Actions ▾  Run

Script  Job details Runs Data quality Schedules Version Control

Script Info

```

1 import sys
2 from awsglue.transforms import *
3 from awsglue.utils import getResolvedOptions
4 from pyspark.context import SparkContext
5 from awsglue.context import GlueContext
6 from awsglue.job import Job
7 from pyspark.sql.functions import col
8
9 ## @params: [JOB_NAME]
10 args = getResolvedOptions(sys.argv, ['JOB_NAME'])
11
12 sc = SparkContext()
13 glueContext = GlueContext(sc)
14 spark = glueContext.spark_session
15 job = Job(glueContext)

```

Test 

Script  Runs Data quality Schedules Version Control

Basic properties 

Name

Description - *optional*

Descriptions can be up to 2048 characters long.

IAM Role

Type

Glue version 

Language

Worker type

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.

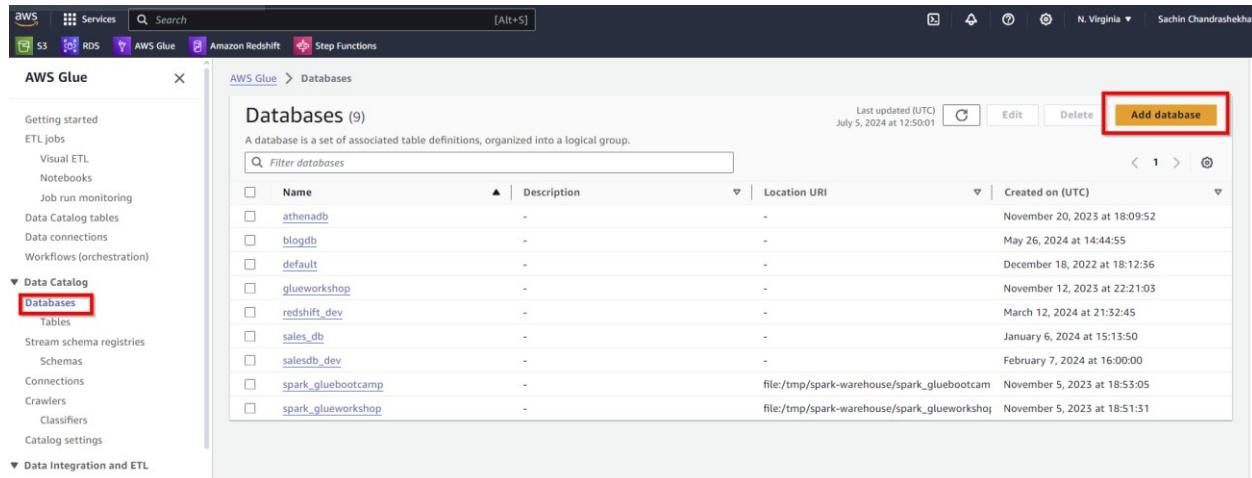
For that, what's the first step?

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.

Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>



AWS Glue > Databases

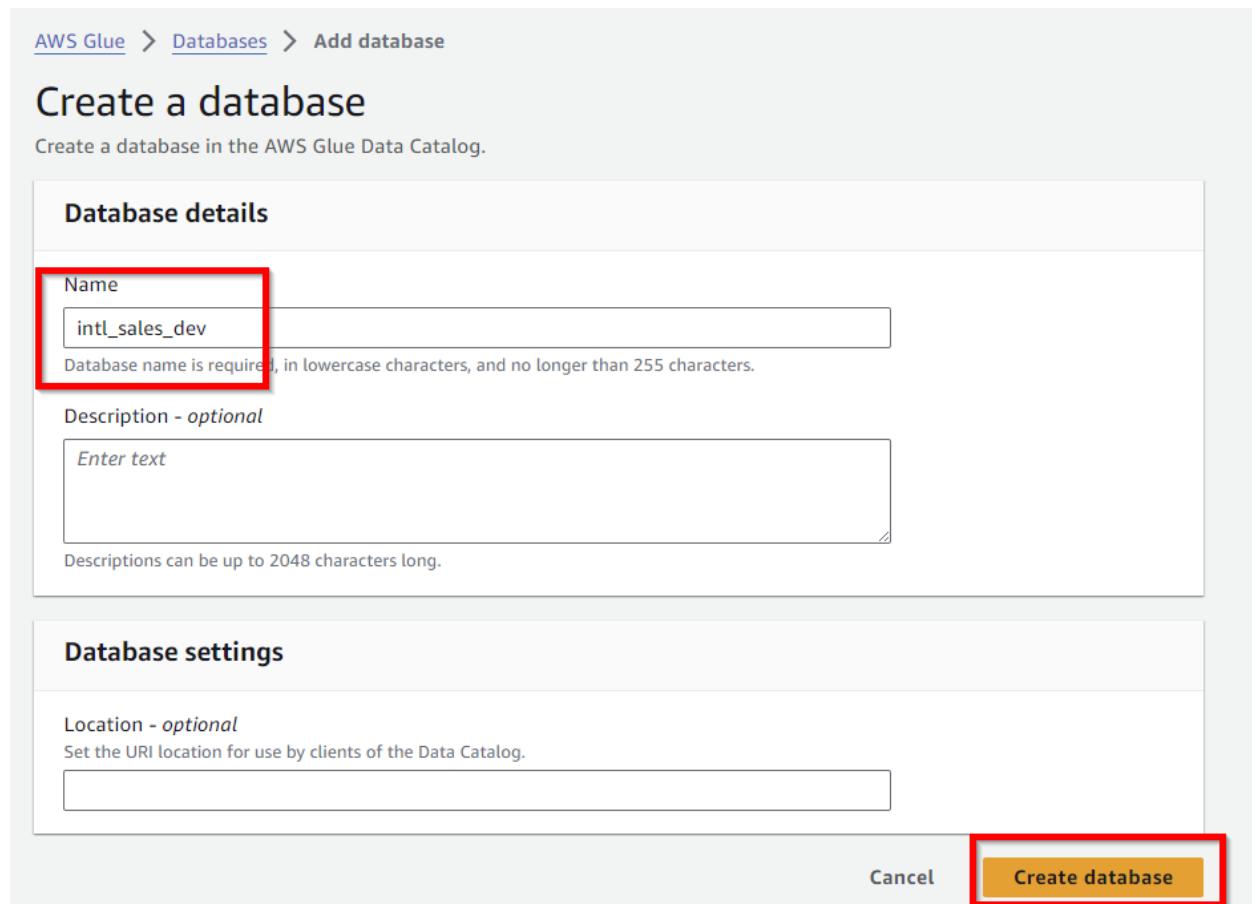
Databases (9)

Name	Description	Location URI	Created on (UTC)
athenadb	-	-	November 20, 2023 at 18:09:52
blogdb	-	-	May 26, 2024 at 14:44:55
default	-	-	December 18, 2022 at 18:12:36
glueworkshop	-	-	November 12, 2023 at 22:21:03
redshift_dev	-	-	March 12, 2024 at 21:32:45
sales_db	-	-	January 6, 2024 at 15:13:50
salesdb_dev	-	-	February 7, 2024 at 16:00:00
spark_gluebootcamp	-	file:/tmp/spark-warehouse/spark_gluebootcamp	November 5, 2023 at 18:53:05
spark_glueworkshop	-	file:/tmp/spark-warehouse/spark_glueworkshop	November 5, 2023 at 18:51:51

Add database

Use this database name – **intl_sales_dev**

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



AWS Glue > Databases > Add database

Create a database

Create a database in the AWS Glue Data Catalog.

Database details

Name: **intl_sales_dev** (highlighted with a red box)

Database name is required, in lowercase characters, and no longer than 255 characters.

Description - optional

Enter text

Descriptions can be up to 2048 characters long.

Database settings

Location - optional

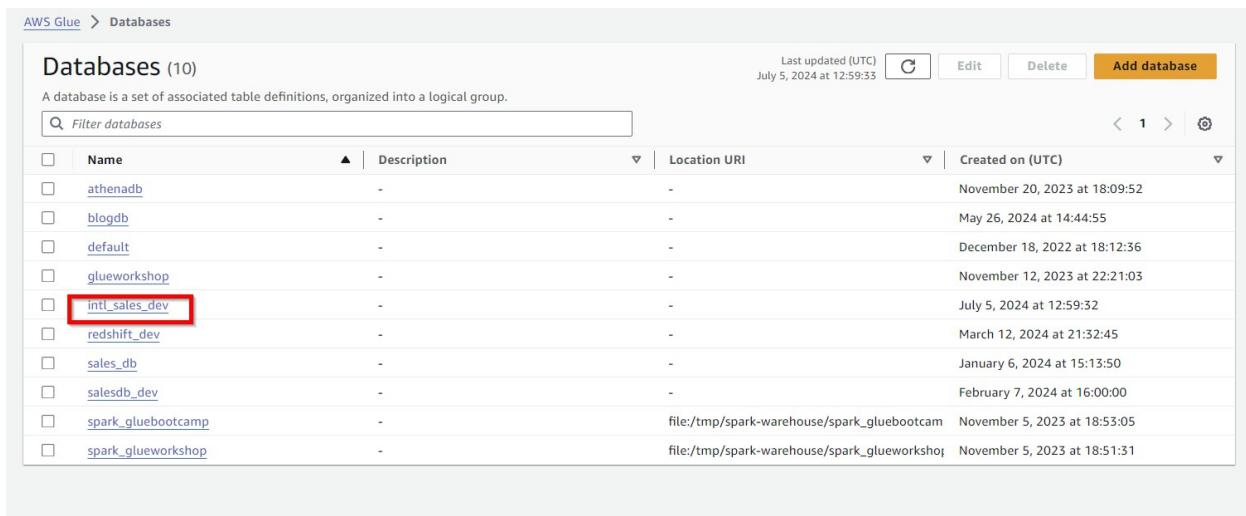
Set the URI location for use by clients of the Data Catalog.

Create database (highlighted with a red box)

Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>

Shows up here –



AWS Glue > Databases

Databases (10)

A database is a set of associated table definitions, organized into a logical group.

Name	Description	Location URI	Created on (UTC)
athenadb	-	-	November 20, 2023 at 18:09:52
blogdb	-	-	May 26, 2024 at 14:44:55
default	-	-	December 18, 2022 at 18:12:36
glueworkshop	-	-	November 12, 2023 at 22:21:03
intl_sales_dev	-	-	July 5, 2024 at 12:59:32
redshift_dev	-	-	March 12, 2024 at 21:32:45
sales_db	-	-	January 6, 2024 at 15:13:50
salesdb_dev	-	-	February 7, 2024 at 16:00:00
spark_gluebootcamp	-	file:/tmp/spark-warehouse/spark_gluebootcam	November 5, 2023 at 18:53:05
spark_glueworkshop	-	file:/tmp/spark-warehouse/spark_glueworkshop	November 5, 2023 at 18:51:31

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.

Amazon S3 > Buckets > dehlive-sales-030798167757-us-east-1 > raw/ > maze/

maze/

Objects Properties

Objects (1) Info

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 ob

Find objects by prefix

<input type="checkbox"/>	Name	Type	Last modified
<input type="checkbox"/>	sales_rds_excercise_full.csv	csv	September 16, 2024, 05:07:15 (UTC-04:00)

Now, let's create a crawler.

AWS Glue

- Getting started
- ETL jobs
- Visual ETL
- Notebooks
- Job run monitoring
- Data Catalog tables
- Data connections
- Workflows (orchestration)
- Crawlers** (highlighted with a red box)
- Classifiers
- Catalog settings
- Data Integration and ETL**
- ETL jobs

AWS Glue > Crawlers

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.

Crawlers (12) Info

View and manage all available crawlers.

Filter crawlers

<input type="checkbox"/>	Name	State	Schedule	Last run	Last run timestamp	Log	Table change...
<input type="checkbox"/>	category_pipe_default_clas...	Ready		Succeeded	February 21, 2024 at...	View log	1 created
<input type="checkbox"/>	category_pipe_test	Ready		Succeeded	February 21, 2024 at...	View log	1 created
<input type="checkbox"/>	covid-data-crawler	Ready		Succeeded	November 13, 2023 ...	View log	1 created
<input type="checkbox"/>	date_tickit_default	Ready		Succeeded	February 21, 2024 at...	View log	1 created
<input type="checkbox"/>	sales_by_region_v2	Ready		Succeeded	March 12, 2024 at 21...	View log	1 created
<input type="checkbox"/>	sales_by_region_v3	Ready		Succeeded	March 19, 2024 at 17...	View log	1 updated
<input type="checkbox"/>	sales_by_region_v4	Ready		Succeeded	March 14, 2024 at 21...	View log	1 created
<input type="checkbox"/>	sales_crawler	Ready		Succeeded	March 23, 2024 at 11...	View log	8 created
<input type="checkbox"/>	sales_crawler_abc_inc	Ready		Succeeded	February 19, 2024 at...	View log	1 created
<input type="checkbox"/>	sales dev	Ready		Succeeded	January 6, 2024 at 1...	View log	1 created

[Create crawler](#)

Enter 'sales_maze_inc'

(indicates sales file from maze inc company)

Sachin Chandrashekhar - Data Engineering Hub

 LinkedIn: <https://www.linkedin.com/in/sachincw/>

**Step 1
Set crawler properties**
Step 2
 Choose data sources and classifiers

Step 3
 Configure security settings

Step 4
 Set output and scheduling

Step 5
 Review and create

Set crawler properties
Crawler details Info
Name

Name can be up to 255 characters long. Some character set including control characters are prohibited.
Description - optional

Descriptions can be up to 2048 characters long.
► Tags - optional
Use tags to organize and identify your resources.
Cancel
Next

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/

Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>

Add data source X

Data source
Choose the source of data to be crawled.

S3 ▼

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

C

Clear selection Add new connection

Location of S3 data

In this account
 In a different account

S3 path
Browse for or enter an existing S3 path.
 View Browse S3

All folders and files contained in the S3 path are crawled. For example, type s3://MyBucket/MyFolder/ to crawl all objects in MyFolder within MyBucket.

Subsequent crawler runs
This field is a global field that affects all S3 data sources.

Crawl all sub-folders
Crawl all folders again with every subsequent crawl.

Crawl new sub-folders only
Only Amazon S3 folders that were added since the last crawl will be crawled. If the schemas are compatible, new partitions will be added to existing tables.

Crawl based on events
Rely on Amazon S3 events to control what folders to crawl.

Sample only a subset of files

Exclude files matching pattern

Cancel Add an S3 data source

AWS Glue > Crawlers > Add crawler

Step 1 Set crawler properties

Step 2 Choose data sources and classifiers

Step 3 Configure security settings

Step 4 Set output and scheduling

Step 5 Review and create

Choose data sources and classifiers

Data source configuration

Is your data already mapped to Glue tables?

Not yet Select one or more data sources to be crawled.

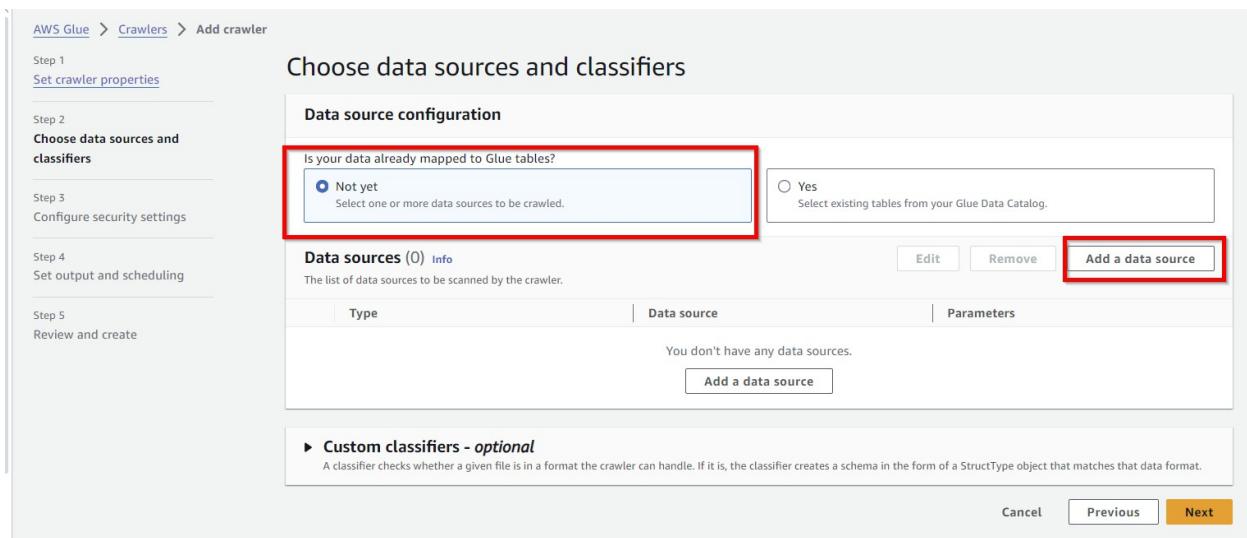
Yes Select existing tables from your Glue Data Catalog.

Data sources (0) Info The list of data sources to be scanned by the crawler.

Type	Data source	Parameters
You don't have any data sources.		
Add a data source		

Custom classifiers - optional A classifier checks whether a given file is in a format the crawler can handle. If it is, the classifier creates a schema in the form of a StructType object that matches that data format.

[Cancel](#) [Previous](#) **Next**



Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>

Choose data sources and classifiers

Data source configuration

Is your data already mapped to Glue tables?

<input checked="" type="radio"/> Not yet Select one or more data sources to be crawled.	<input type="radio"/> Yes Select existing tables from your Glue Data Catalog.
--	--

Data sources (1) Info

The list of data sources to be scanned by the crawler.

Type	Data source	Parameters
<input type="radio"/> S3	s3://dehlive-sales-030798167757-us-east-1...	Recrawl all

Custom classifiers - optional
A classifier checks whether a given file is in a format the crawler can handle. If it is, the classifier creates a schema in the form of a StructType object that matches that data format.

Cancel Previous **Next**

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

AWS Glue > Crawlers > Add crawler

Step 1 Set crawler properties

Step 2 Choose data sources and classifiers

Configure security settings

IAM role Info

Existing IAM role: AWSGlueServiceRole-1

Lake Formation configuration - optional
Allow the crawler to use Lake Formation credentials for crawling the data source. Learn more. [i]

Use Lake Formation credentials for crawling S3 data source
Checking this box will allow the crawler to use Lake Formation credentials for crawling the data source. If the data source is registered in another account, you must provide the registered account ID. Otherwise, the crawler will crawl only those data sources associated to the account. Only applicable to S3, Glue Catalog, Iceberg, and Hudi data sources.

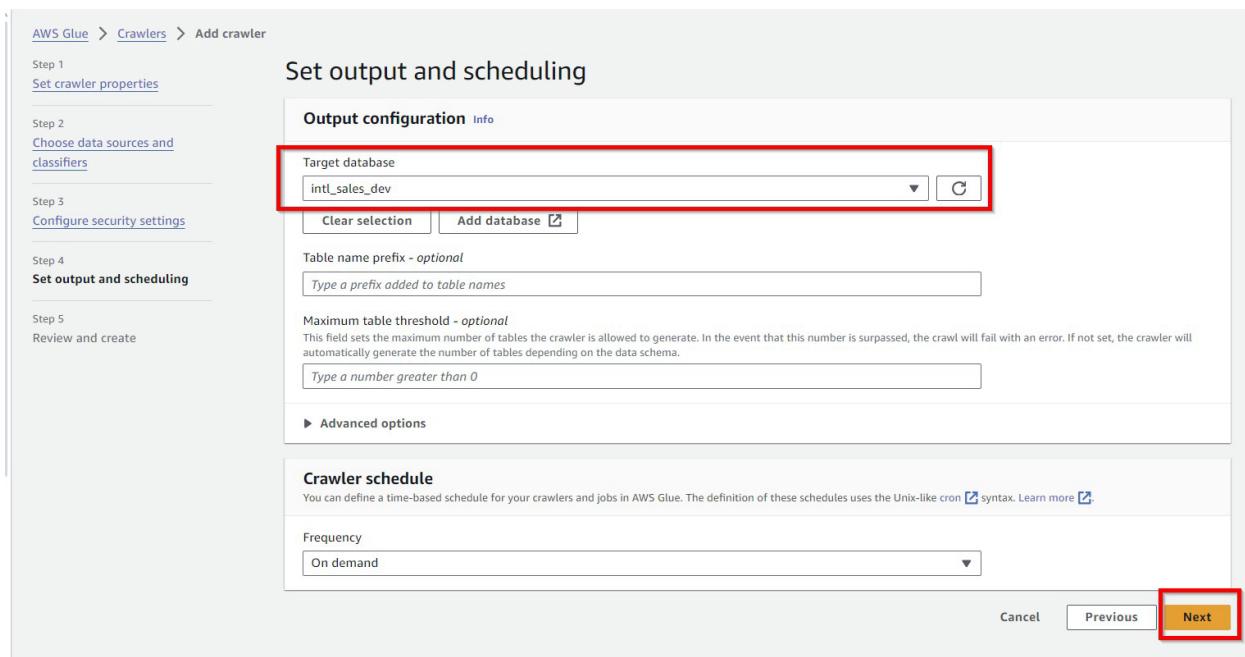
Security configuration - optional
Enable at-rest encryption with a security configuration.

Cancel Previous **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.



AWS Glue > Crawlers > Add crawler

Step 1 Set crawler properties

Step 2 Choose data sources and classifiers

Step 3 Configure security settings

Step 4 Set output and scheduling

Step 5 Review and create

Set output and scheduling

Output configuration Info

Target database Add database

Table name prefix - optional

Maximum table threshold - optional
This field sets the maximum number of tables the crawler is allowed to generate. In the event that this number is surpassed, the crawl will fail with an error. If not set, the crawler will automatically generate the number of tables depending on the data schema.

Crawler schedule
You can define a time-based schedule for your crawlers and jobs in AWS Glue. The definition of these schedules uses the Unix-like cron syntax. Learn more

Frequency

Finally review and create the crawler.

Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>

[Set crawler properties](#)

Step 1: Set crawler properties

Set crawler properties

Name	sales_maze_inc	Description	-	Tags
------	----------------	-------------	---	------

Step 2: Choose data sources and classifiers

Data sources (1) Info
 The list of data sources to be scanned by the crawler.

Type	Data source	Parameters
S3	s3://dehlive-sales-030798167757-us-east-1/r...	Recrawl all

Step 3: Configure security settings

Configure security settings

IAM role	AWSGlueServiceRole-1	Security configuration	Lake Formation configuration
----------	----------------------	------------------------	------------------------------

Step 4: Set output and scheduling

Set output and scheduling

Database	intl_sales_dev	Table prefix - <small>optional</small>	Maximum table threshold - <small>optional</small>	Schedule
		-	-	On demand

[Cancel](#) [Previous](#) [Create crawler](#)

Now, run the crawler –

One crawler successfully created
 The following crawler is now created: "sales_maze_inc"

[AWS Glue](#) > [Crawlers](#) > sales_maze_inc

sales_maze_inc

Last updated (UTC) July 5, 2024 at 13:27:40 [Run crawler](#) [Edit](#) [Delete](#)

Crawler properties

Name	sales_maze_inc	IAM role	AWSGlueServiceRole-1	Database	intl_sales_dev	State	READY
Description	-	Security configuration	-	Lake Formation configuration	-	Table prefix	-
Maximum table threshold	-						

[Advanced settings](#)

[Crawler runs](#) [Schedule](#) [Data sources](#) [Classifiers](#) [Tags](#)

Crawler runs (0)
 The list of crawler runs for this crawler.

Start time (UTC)	End time (UTC)	Current/last duration	Status	DPU hours	Table changes
You don't have any crawler runs.					

[Run crawler](#)

Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>

Crawler successfully starting
The following crawler is now starting: "sales_maze_inc"

AWS Glue > Crawlers > sales_maze_inc

sales_maze_inc

Last updated (UTC)
July 5, 2024 at 13:28:28

Run crawler Edit Delete

Crawler properties															
Name sales_maze_inc	IAM role AWSGlueServiceRole-1	Database intl_sales_dev	State RUNNING												
Description -	Security configuration -	Lake Formation configuration -	Table prefix -												
Maximum table threshold -															
▶ Advanced settings															
Crawler runs Schedule Data sources Classifiers Tags															
Crawler runs (1) The list of crawler runs for this crawler. <table border="1"> <thead> <tr> <th>Start time (UTC)</th> <th>End time (UTC)</th> <th>Current/last duration</th> <th>Status</th> <th>DPU hours</th> <th>Table changes</th> </tr> </thead> <tbody> <tr> <td>July 5, 2024 at 13:28:22</td> <td>-</td> <td>11 s</td> <td>Running</td> <td>-</td> <td>-</td> </tr> </tbody> </table>				Start time (UTC)	End time (UTC)	Current/last duration	Status	DPU hours	Table changes	July 5, 2024 at 13:28:22	-	11 s	Running	-	-
Start time (UTC)	End time (UTC)	Current/last duration	Status	DPU hours	Table changes										
July 5, 2024 at 13:28:22	-	11 s	Running	-	-										

Crawler successfully starting
The following crawler is now starting: "sales_maze_inc"

AWS Glue > Crawlers > sales_maze_inc

sales_maze_inc

Last updated (UTC)
July 5, 2024 at 13:28:28

Run crawler Edit Delete

Crawler properties															
Name sales_maze_inc	IAM role AWSGlueServiceRole-1	Database intl_sales_dev	State RUNNING												
Description -	Security configuration -	Lake Formation configuration -	Table prefix -												
Maximum table threshold -															
▶ Advanced settings															
Crawler runs Schedule Data sources Classifiers Tags															
Crawler runs (1/1) The list of crawler runs for this crawler. <table border="1"> <thead> <tr> <th>Start time (UTC)</th> <th>End time (UTC)</th> <th>Current/last duration</th> <th>Status</th> <th>DPU hours</th> <th>Table changes</th> </tr> </thead> <tbody> <tr> <td>July 5, 2024 at 13:28:22</td> <td>July 5, 2024 at 13:29:26</td> <td>01 min 03 s</td> <td>Completed</td> <td>0.040</td> <td>1 table change, 0 partition changes</td> </tr> </tbody> </table>				Start time (UTC)	End time (UTC)	Current/last duration	Status	DPU hours	Table changes	July 5, 2024 at 13:28:22	July 5, 2024 at 13:29:26	01 min 03 s	Completed	0.040	1 table change, 0 partition changes
Start time (UTC)	End time (UTC)	Current/last duration	Status	DPU hours	Table changes										
July 5, 2024 at 13:28:22	July 5, 2024 at 13:29:26	01 min 03 s	Completed	0.040	1 table change, 0 partition changes										

Click on View CloudWatch logs to see the logs –

Sachin Chandrashekhar - Data Engineering Hub

Follow me on Twitter: <https://x.com/AWSCloudSachin>

CloudWatch > Log groups > /aws-glue/crawlers > sales_maze_inc

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

Q "34067818-03f1-4be6-82ae-26938b10a6a8" X Clear 1m 30m 1h 12h Custom UTC timezone Display ⚡

Timestamp	Message
2024-07-05T13:28:41.410Z	[34067818-03f1-4be6-82ae-26938b10a6a8] BENCHMARK : Running Start Crawl for Crawler sales_maze_inc
2024-07-05T13:29:24.518Z	[34067818-03f1-4be6-82ae-26938b10a6a8] BENCHMARK : Classification complete, writing results to database intl_sales_dev
2024-07-05T13:29:24.533Z	[34067818-03f1-4be6-82ae-26938b10a6a8] INFO : Crawler configured with Configuration {"Version":1.0,"CreatePartitionIndex":true} and SchemaChangePolicy...
2024-07-05T13:29:26.022Z	[34067818-03f1-4be6-82ae-26938b10a6a8] INFO : Created table maze in database intl_sales_dev
2024-07-05T13:29:26.980Z	[34067818-03f1-4be6-82ae-26938b10a6a8] BENCHMARK : Finished writing to Catalog
2024-07-05T13:29:27.019Z	[34067818-03f1-4be6-82ae-26938b10a6a8] INFO : Run Summary For TABLE:
2024-07-05T13:29:27.020Z	[34067818-03f1-4be6-82ae-26938b10a6a8] INFO : ADD: 1
2024-07-05T13:30:41.898Z	[34067818-03f1-4be6-82ae-26938b10a6a8] BENCHMARK : Crawler has finished running and is in state READY

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

AWS Glue

Getting started ETL jobs Visual ETL Notebooks Job run monitoring Data Catalog tables Data connections Workflows (orchestration)

Data Catalog Databases Tables Stream schema registries Schemas Connections Crawlers Classifiers Catalog settings Data Integration and ETL

AWS Glue > Databases > intl_sales_dev

intl_sales_dev

Database properties

Name: intl_sales_dev Description: - Location: - Created on (UTC): July 5, 2024 at 12:59:32

Tables (1)

Last updated (UTC): July 5, 2024 at 15:33:45

Add table

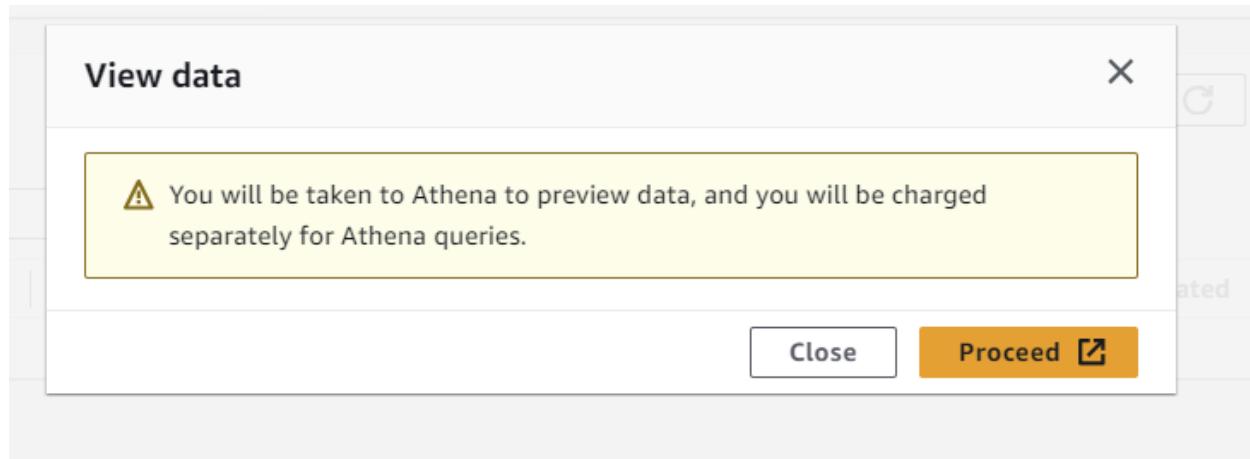
Filter tables

Name	Database	Location	Classification	Deprecated	View data	Data quality
maze	intl_sales_dev	s3://dehlive-sales-030798	SV	-	Table data	View data quality

Click on proceed below

Sachin Chandrashekhar - Data Engineering Hub

Follow me on Twitter: <https://x.com/AWSCloudSachin>



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 –

#	uuid	country	itemtype	saleschannel	orderpriority	orderdate	region
1	535113847	Azerbaijan	Snacks	Online	C	10/8/2014	Middle East and North Africa
2	874708545	Panama	Cosmetics	Offline	L	2/22/2015	Central America and the Caribbean

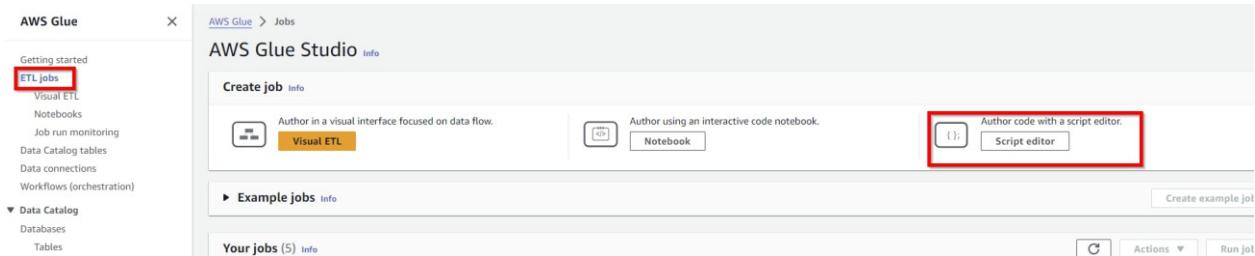
Now, since we have validated data looks good and we are able to query via data catalog object table "maze", it's time to create the Glue ETL script.

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

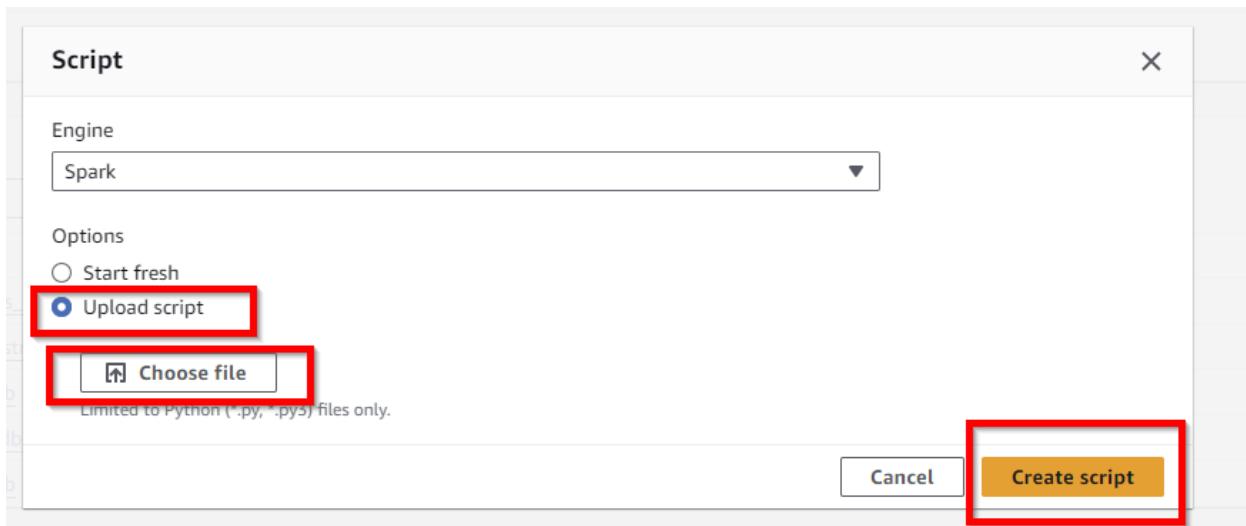
We loaded data in data lake, then created table on top of the file. This is ELT and not ETL.

Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>

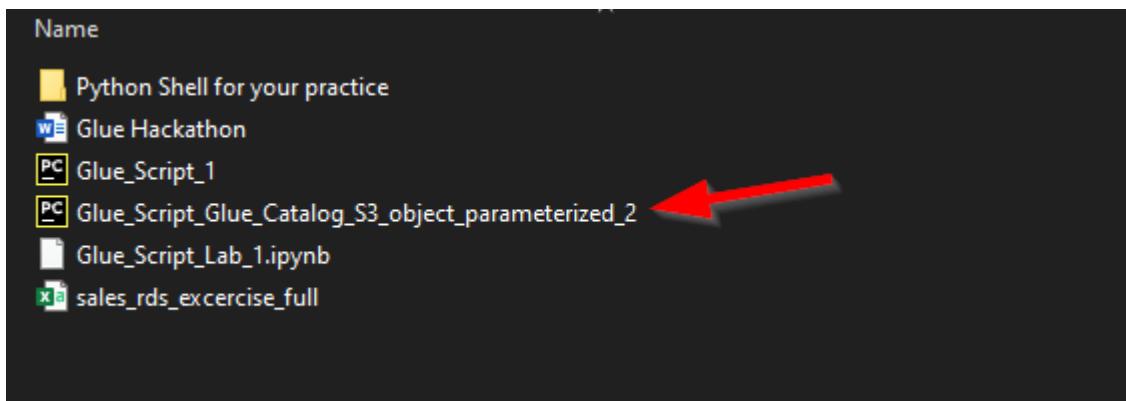


AWS Glue Studio interface showing the 'Create job' section. It displays three options: 'Visual ETL' (selected), 'Notebook', and 'Script editor'. The 'Script editor' option is highlighted with a red box.



Script creation dialog box. Under 'Engine', 'Spark' is selected. Under 'Options', 'Upload script' is selected (radio button highlighted with a red box). A 'Choose file' button is also highlighted with a red box. A note below says 'Limited to Python (.py, .py3) files only.' At the bottom right, the 'Create script' button is highlighted with a red box.

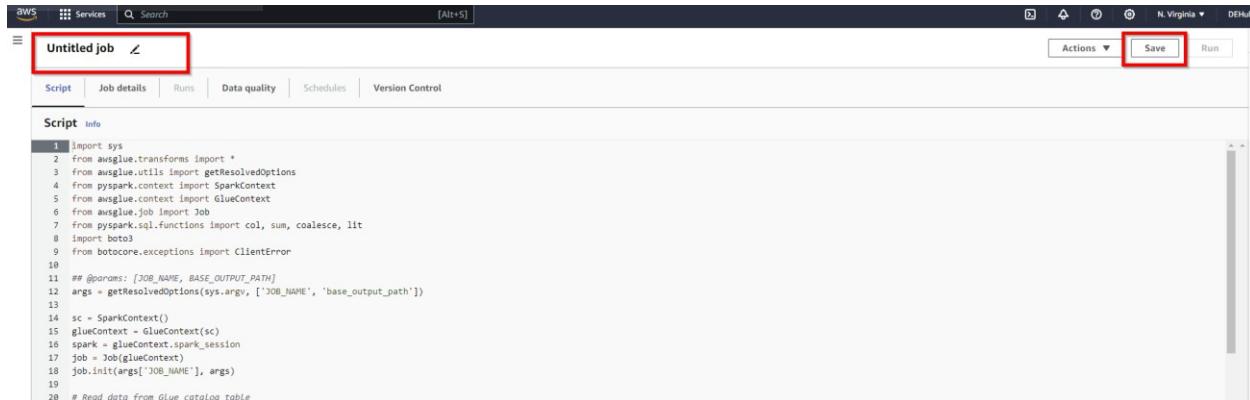
Use file `Glue_Script_Glue_Catalog_S3_object_parameterized_2.py` I shared.



Rename the job & Save.

Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>



```

1 import sys
2 from awsglue.transforms import *
3 from awsglue.utils import getResolvedOptions
4 from pyspark.context import SparkContext
5 from awsglue.context import GlueContext
6 from awsglue.job import Job
7 from pyspark.sql.functions import col, sum, coalesce, lit
8 import boto3
9 from botocore.exceptions import ClientError
10
11 ## @params: {JOB_NAME, BASE_OUTPUT_PATH}
12 args = getResolvedOptions(sys.argv, ['JOB_NAME', 'base_output_path'])
13
14 sc = SparkContext()
15 glueContext = GlueContext(sc)
16 spark = glueContext.spark_session
17 job = Job(glueContext)
18 job.init(args['JOB_NAME'], args)
19
20 # Read data from Glue catalog table

```

Now choose the IAM role

Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>

Glue_Script_Glue_Catalog_S3_object...

[Script](#)
[Job details 1](#)
[Runs](#)
[Data quality](#)
[Schedules](#)
[Version Control](#)

Basic properties [Info](#)

Name

Description - optional

Descriptions can be up to 2048 characters long.

IAM Role

Role assumed by the job with permission to access your data stores. Ensure that this role has permission to your Amazon S3 sources, targets, temporary directory, scripts, and any libraries used by the job.



⚠ IAM Role is required.

Type

The type of ETL job. This is set automatically based on the types of data sources you have selected.

Glue version [Info](#)

Language

Worker type

Set the type of predefined worker that is allowed when a job runs.

(4vCPU and 16GB RAM)

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.

Sachin Chandrashekhar - Data Engineering Hub



Follow me on Twitter: <https://x.com/AWSCloudSachin>

Glue_Script_Glue_Catalog_S3_object...

Script **Job details** Runs Data quality Schedules Version Control

Automatically scale the number of workers

AWS Glue will optimize costs and resource usage by dynamically scaling the number of workers up and down throughout the job run. Requires Glue 3.0 or later.

Maximum number of workers

The number of workers you want AWS Glue to allocate to this job.

Generate job insights

AWS Glue will analyze your job runs and provide insights on how to optimize your jobs and the reasons for job failures.

Job bookmark | [Info](#)

Specifies how AWS Glue processes job bookmark when the job runs. It can remember previously processed data (Enable), update state information (Pause), or ignore state information (Disable).

Flex execution | [Info](#)

Reduce costs by running this job on spare capacity. Ideal for non-urgent workloads that don't require fast jobs start times or consistent execution times. See recommendations, limitations and pricing in the help panel by clicking on the Info link above.

Number of retries

Job timeout (minutes)

Set the execution time. The default is 2,880 minutes (48 hours) for a Glue ETL job. No job timeout is defaulted for a Glue Streaming job.

Usage profile

-

► Advanced properties

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

Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>



It looks like this.

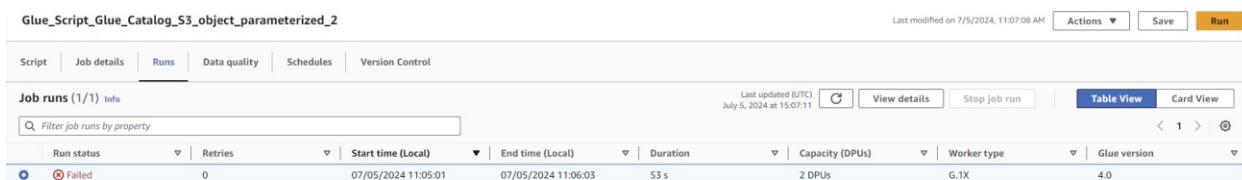
Finally save the job and run it.

Go to Runs and you see it running.

Sachin Chandrashekhar - Data Engineering Hub

Follow me on Twitter: <https://x.com/AWSCloudSachin>

If it fails, you will see the reason here –



Glue_Script_Glue_Catalog_S3_object_parameterized_2

Last modified on 7/5/2024, 11:07:08 AM Actions ▾ Save Run

Script Job details Runs Data quality Schedules Version Control

Job runs (1/1) Info Last updated (UTC) July 5, 2024 at 15:07:11 View details Stop job run Table View Card View < 1 > ⌂

Filter job runs by property

Run status	Retries	Start time (Local)	End time (Local)	Duration	Capacity (DPU)	Worker type	Glue version
Failed	0	07/05/2024 11:05:01	07/05/2024 11:06:03	53 s	2 DPU	G.1X	4.0



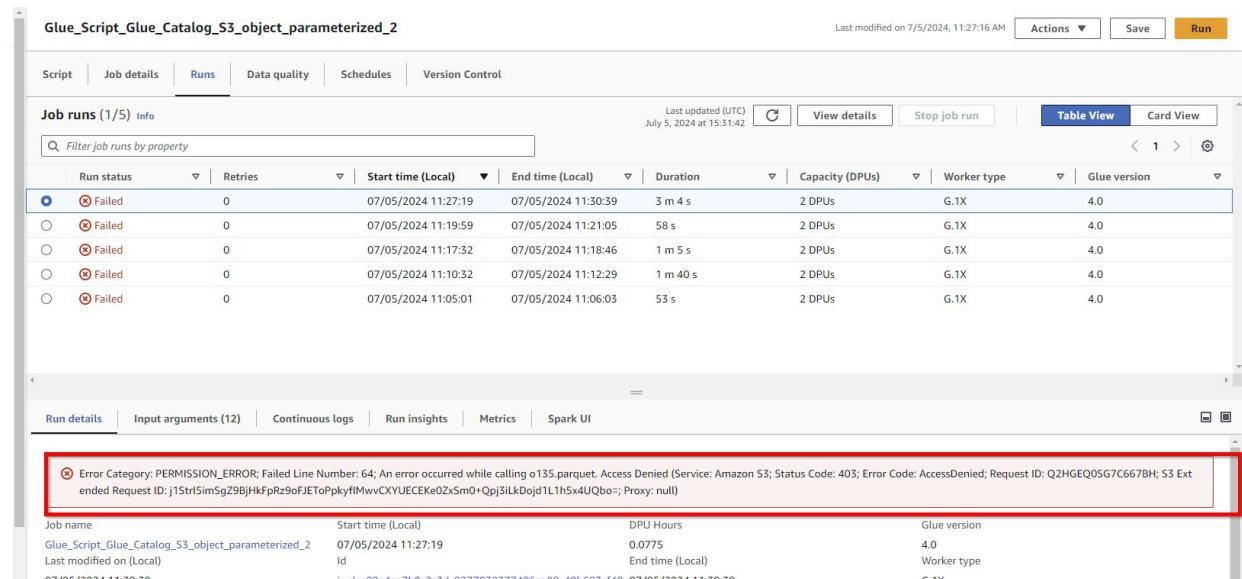
Glue_Script_Glue_Catalog_S3_object_parameterized_2

Last modified on 7/5/2024, 11:06:03 Actions ▾ Save Run

Run details Input arguments (12) Continuous logs Run insights Metrics Spark UI

Error Category: INVALID_ARGUMENT_ERROR; Failed Line Number: 12; GlueArgumentError: the following arguments are required: --base_output_path

Job name	Start time (Local)	DPU Hours	Glue version
Glue_Script_Glue_Catalog_S3_object_parameterized_2	07/05/2024 11:05:01	0.016667	4.0
Last modified on (Local)	Id	End time (Local)	Worker type
07/05/2024 11:06:03	jr_289eb5203401901e7586e35e4d93fbeb3d0206ed7b457a0fe1d0a	07/05/2024 11:06:03	G.1X
Log group name	Run status	Start-up time	Max capacity
/aws-glue/jobs	Failed	9 seconds	2 DPU
Number of workers	Retries attempt number	Execution time	Execution class



Glue_Script_Glue_Catalog_S3_object_parameterized_2

Last modified on 7/5/2024, 11:27:16 AM Actions ▾ Save Run

Script Job details Runs Data quality Schedules Version Control

Job runs (1/5) Info Last updated (UTC) July 5, 2024 at 15:31:42 View details Stop job run Table View Card View < 1 > ⌂

Filter job runs by property

Run status	Retries	Start time (Local)	End time (Local)	Duration	Capacity (DPU)	Worker type	Glue version
Failed	0	07/05/2024 11:27:19	07/05/2024 11:30:39	3 m 4 s	2 DPU	G.1X	4.0
Failed	0	07/05/2024 11:19:59	07/05/2024 11:21:05	58 s	2 DPU	G.1X	4.0
Failed	0	07/05/2024 11:17:32	07/05/2024 11:18:46	1 m 5 s	2 DPU	G.1X	4.0
Failed	0	07/05/2024 11:10:32	07/05/2024 11:12:29	1 m 40 s	2 DPU	G.1X	4.0
Failed	0	07/05/2024 11:05:01	07/05/2024 11:06:03	53 s	2 DPU	G.1X	4.0

Run details Input arguments (12) Continuous logs Run insights Metrics Spark UI

Error Category: PERMISSION_DENIED; Failed Line Number: 64; An error occurred while calling o135.parquet. Access Denied (Service: Amazon S3; Status Code: 403; Error Code: AccessDenied; Request ID: Q2HGEQ0SG7C667BH; S3 Extended Request ID: j1Str5imSgZ9BjHkFp29fJEToPpkylMkwvCYUECEke0ZxSm0+Qpj5IkDqjdL1h5x4UQbo=; Proxy: null)

Job name	Start time (Local)	DPU Hours	Glue version
Glue_Script_Glue_Catalog_S3_object_parameterized_2	07/05/2024 11:27:19	0.0775	4.0
Last modified on (Local)	Id	End time (Local)	Worker type
07/05/2024 11:30:39	ir_dca09a4ee7b0c2c3dc9277038377406aa88c40b693cf40	07/05/2024 11:30:39	G.1X

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.

Sachin Chandrashekhar - Data Engineering Hub

Follow me on Twitter: <https://x.com/AWSCloudSachin>

Amazon S3 > Buckets > dehlive-sales-942273247308-us-east-1 > curated/ > year=2024/ > month=7/ > day=5/
day=5/

[Copy S3 URI](#)

Objects Properties

Objects (1) info

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](#)

Find objects by prefix

[Actions](#) [Create folder](#) [Upload](#)

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	part-00000-d5ed074c-09b3-47ce-a17b-4d8c59c5e7a6-c000.snappy.parquet	parquet	July 5, 2024, 11:37:27 (UTC-04:00)	1.9 KB	Standard

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.

Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>

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.

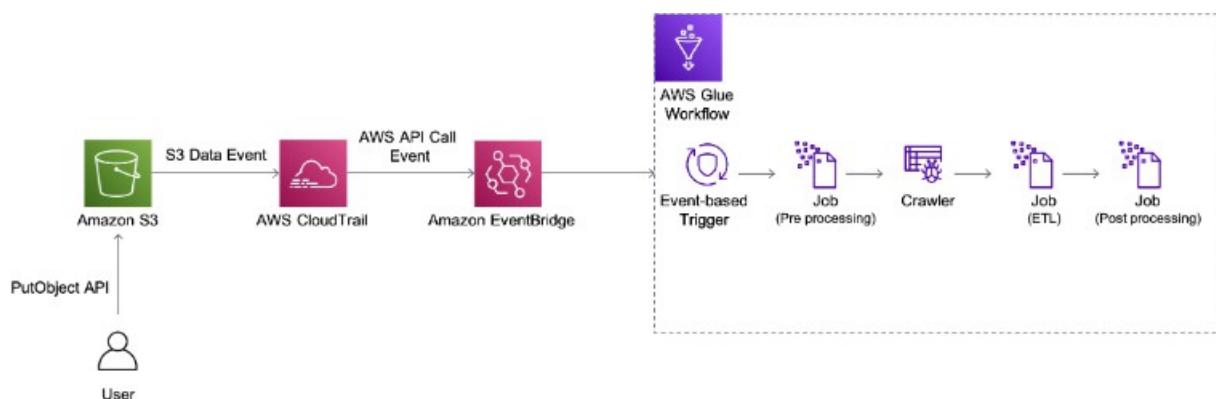
AWS Glue Workshop by AWS

<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.

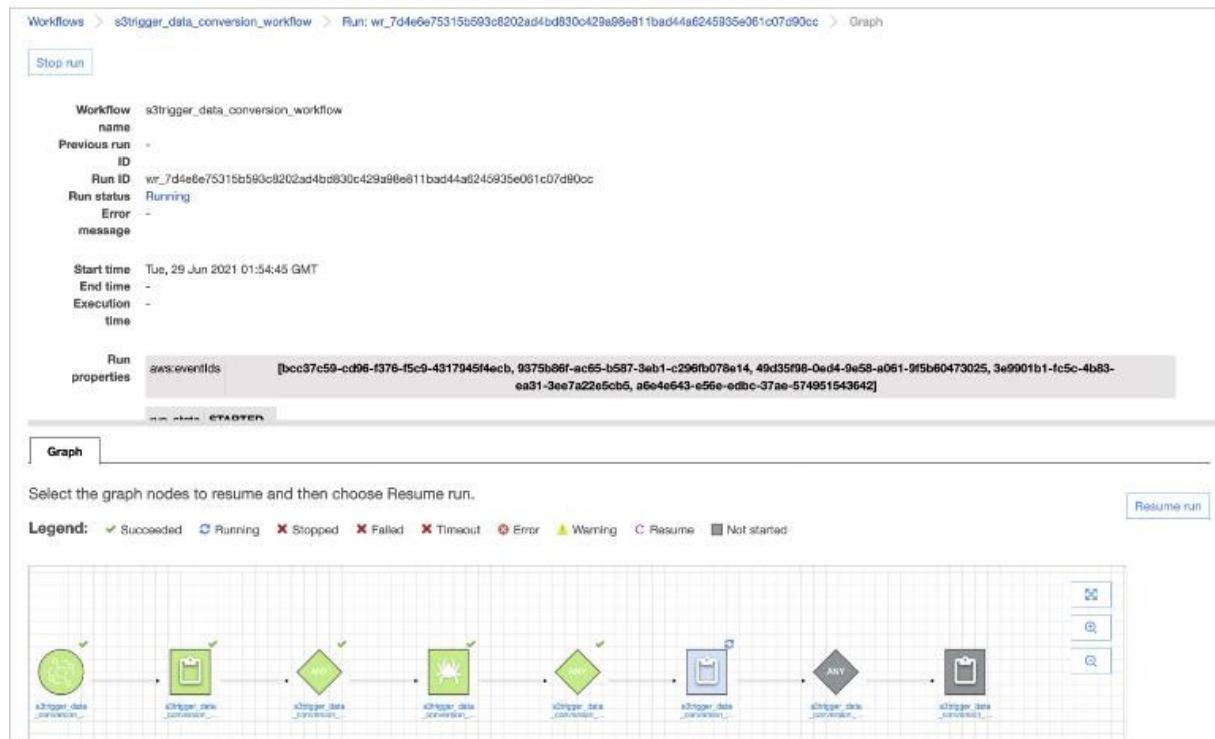
This uses **Glue Workflows**, Glue crawler, Glue ETL and eventbridge.

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



Sachin Chandrashekhar - Data Engineering Hub

 Follow me on Twitter: <https://x.com/AWSCloudSachin>



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>