



Amazon Web Services Data Engineering Immersion Day

Lab 2. ETL with AWS Glue

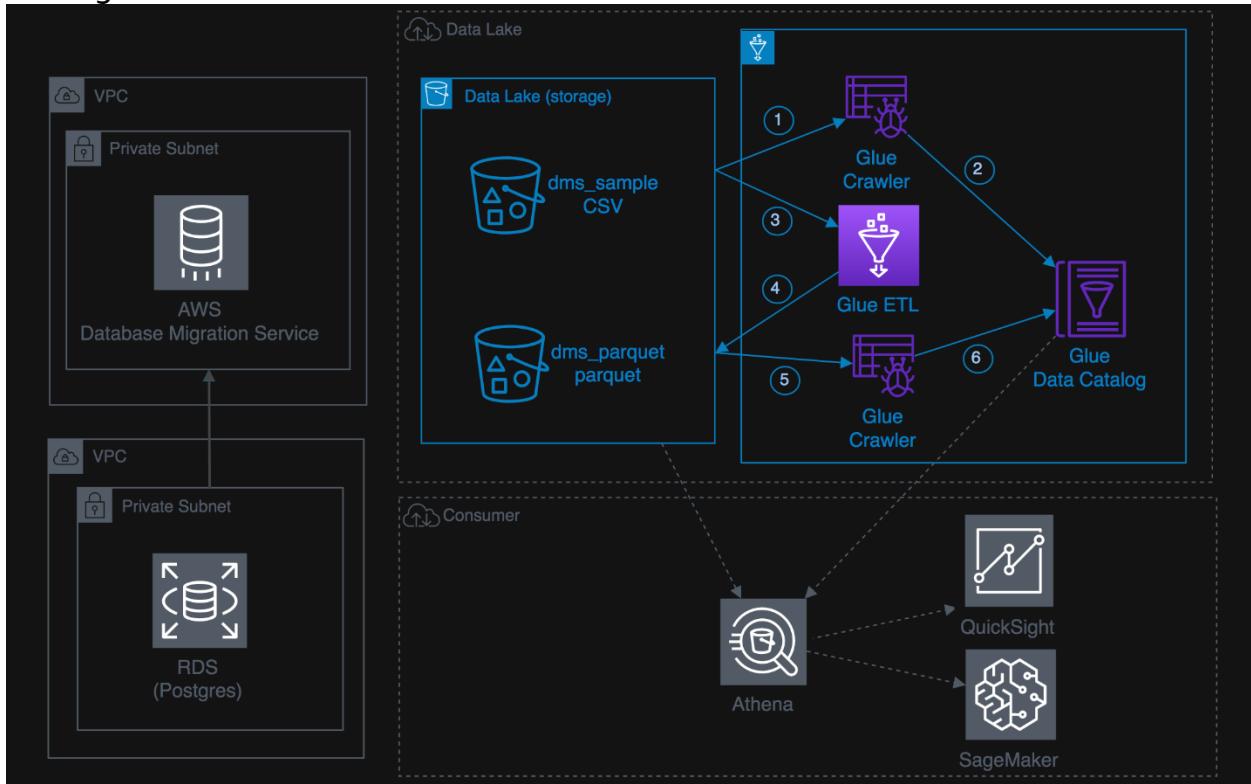
August 2020

Table of Contents

<i>Introduction</i>	2
<i>Get Started Using the Lab Environment</i>	3
PART A: Data Validation and ETL	6
Create Glue Crawler for initial full load data	6
Data Validation Exercise	10
Data ETL Exercise	11
Create Glue Crawler for Parquet Files	16
PART B: Glue Job Bookmark (Optional):	19
Step 1: Create Glue Crawler for ongoing replication (CDC Data)	19
Step 2: Create a Glue Job with Bookmark Enabled	23
Step 3: Create Glue crawler for Parquet data in S3	25
Step 4: Generate CDC data and to observe bookmark functionality	28
PART C: Glue Workflows (Optional, self-paced)	29
Overview:	29
Creating and Running Workflows:.....	29

Introduction

This lab will give you an understanding of the AWS Glue – a fully managed data catalog and ETL service



Prerequisites

1. Completed Lab 1. Hydrating the Data Lake with DMS

Tasks Completed in this Lab:

In this lab you will be completing the following tasks. You can choose to complete only **Part-(A)** to move to next lab where tables can be queried using Amazon Athena and Visualize with Amazon Quicksight

1. [PART-\(A\): Data Validation and ETL](#)
2. [PART- \(B\): Glue Job Bookmark Functionality\(Optional\)](#)
3. [PART- \(C\): Glue Workflows\(Optional\)](#)

The Lab is also available - <https://aws-dataengineering-day.workshop.aws/>

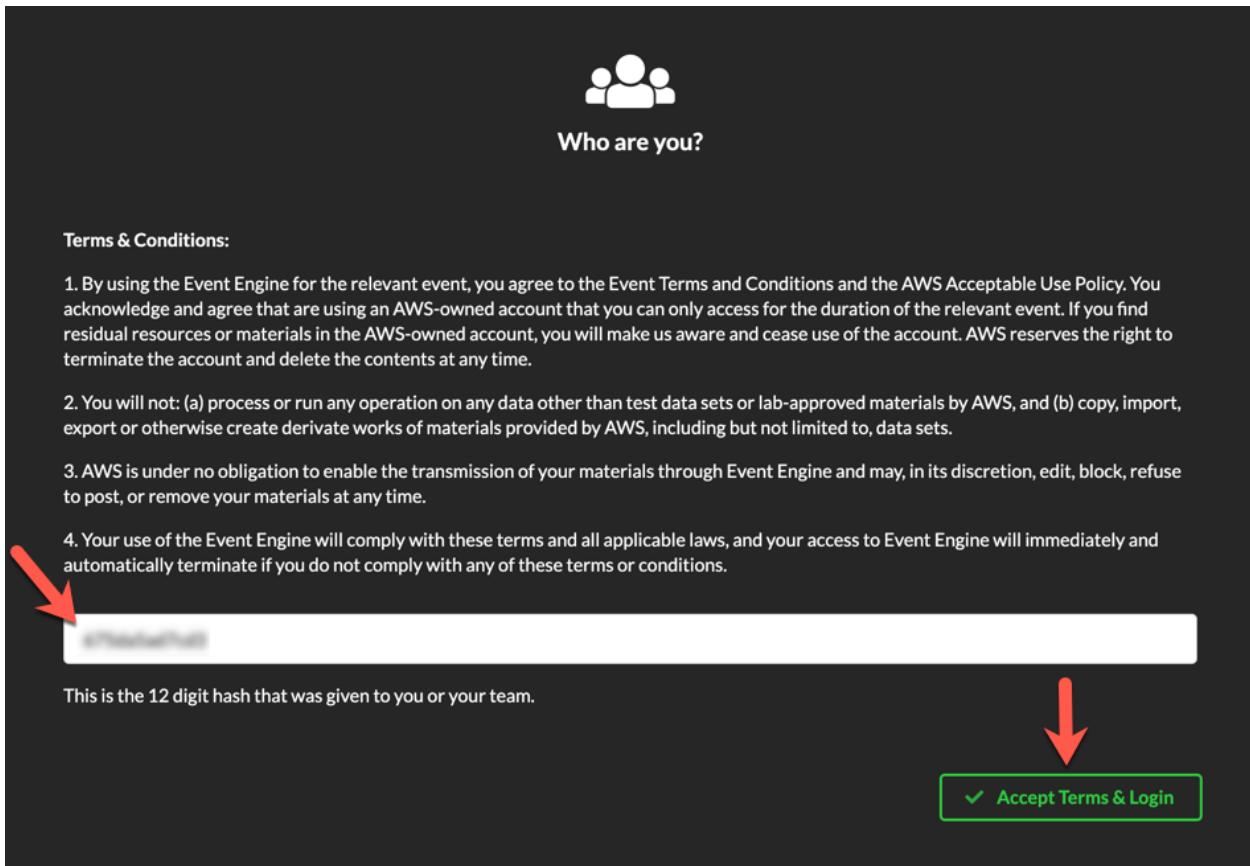
Get Started Using the Lab Environment

Please skip this section if you are running the lab on your own AWS account.

Today, you are attending a formal event and you will have been sent your access details beforehand. If in the future you might want to perform these labs in your own AWS environment by yourself, you can follow instructions on GitHub - <https://github.com/aws-samples/data-engineering-for-aws-immersion-day>.

A 12-character access code (or 'hash') is the access code that grants you permission to use a dedicated AWS account for the purposes of this workshop.

1. Go to <https://dashboard.eventengine.run/>, enter the access code and click Proceed:



2. On the Team Dashboard web page you will see a set of parameters that you will need during the labs. Best to save them to a text file locally, alternatively you can always go to this page to review them. Replace the parameters with the corresponding values from here where indicated in subsequent labs:

Lab 2. ETL with AWS Glue

Because you're at a formal event, some AWS resources have been pre-deployed for your convenience, for example:

- The source database connection in RDS DB Info module

RDS DB Info

Outputs:

No outputs defined

Readme

- S3 Bucket, IAM role for the Glue lab etc

Environment Setup

Outputs:

S3 Bucket name
mod-3fccddd609114925-dmslabs3bucket-1ngcgzzcnd15u

BusinessAnalystUser
mod-3fccddd609114925-BusinessAnalystUser-MBOXFZLQLOXX

DMSLabRoleS3 ARN
arn:aws:iam::377243295828:role/mod-3fccddd609114925-DMSLabRoleS3-O2VT1RSN43SG

Glue Lab Role
mod-3fccddd609114925-GlueLabRole-YLTJA13WW6WT

S3BucketWorkgroupA
mod-3fccddd609114925-s3bucketworkgroupa-tbon3m1mkunh

S3BucketWorkgroupB
mod-3fccddd609114925-s3bucketworkgroupb-18ygl8nfp8ead

WorkgroupManagerUser
mod-3fccddd609114925-WorkgroupManagerUser-5IVE0UQNIBG4

Readme

3. On the Team Dashboard, please click AWS Console to log into the AWS Management Console:

Team Dashboard

Event

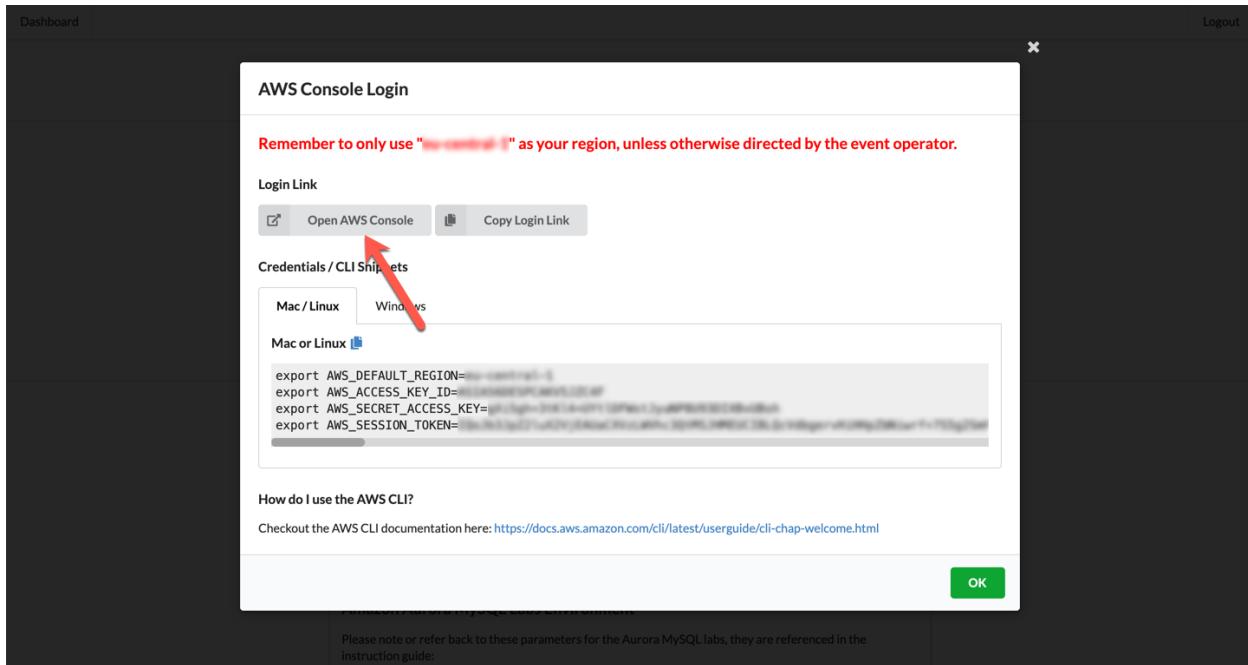
AWS Console SSH Key

Event: Data Engineering Immersion Day - Test
Team Name:

Event ID: d2302d4ae9ff4ea2857846b74f7de7e2
Team ID: 1c2f7ad7ec044b0b8276f917c5983133

Lab 2. ETL with AWS Glue

4. Click Open Console. For the purposes of this workshop, you will not need to use command line and API access credentials:



Once you have completed these steps, you can continue with the rest of this lab.

Lab 2. ETL with AWS Glue

PART A: Data Validation and ETL

Create Glue Crawler for initial full load data

1. Navigate to the AWS Glue service:

<https://console.aws.amazon.com/glue/home?region=us-east-1>

The screenshot shows the AWS Glue service console. In the top navigation bar, there is a search bar with the text "glue". Below the search bar, a list of services is shown, with "AWS Glue" being the first item. The "AWS Glue" entry is described as "A fully managed ETL (extract, transform, and load) service". There are also other entries like "AWS Lake Formation" and "S3" and "EC2". At the bottom of the list, there is a link to "All services".

2. On the AWS Glue menu, select **Crawlers**.

The screenshot shows the "Crawlers" page under the AWS Glue service. The left sidebar has a "Crawlers" section selected. The main area displays a table with columns: Name, Schedule, Status, Logs, Last runtime, Median runtime, Tables updated, and Tables added. A message says "You don't have any crawlers yet." and there is a blue "Add crawler" button.

3. Click **Add crawler**.
4. Enter **glue-lab-crawler** as the crawler name for initial data load.
5. Optionally, enter the description. This should also be descriptive and easily recognized and Click **Next**.

The screenshot shows the "Add crawler" step 1. On the left, there is a sidebar with options: "Crawler info" (selected), "Crawler source type", "Data store", "IAM Role", "Schedule", "Output", and "Review all steps". The main area has a title "Add information about your crawler". It contains a "Crawler name" field with "glue-lab-crawler" entered, and a note "Tags, description, security configuration, and classifiers (optional)". At the bottom right is a "Next" button.

6. Choose **Crawler Source Type** as **Data Stores** and Click **Next**

The screenshot shows the "Add crawler" step 2. On the left, the "Crawler source type" option is selected in the sidebar. The main area has a title "Specify crawler source type" and a note "Choose Existing catalog tables to specify catalog tables as the crawler source. The selected tables specify the data stores to crawl. This option doesn't support JDBC data stores." It contains a "Crawler source type" section with "Data stores" selected. At the bottom right are "Back" and "Next" buttons.

7. On the **Add a data store** page, make the following selections:

Lab 2. ETL with AWS Glue

- a. For Choose a data store, click the drop-down box and select **S3**.
- b. For Crawl data in, select **Specified path in my account**.
- c. For Include path, browse to the target folder for your DMS initial export from Lab 1, e.g., **s3://dmslab-student-dmslabs3bucket-woti4bf73cw3/tickets**

8. Click **Next**.

Add crawler

Crawler info
glue-lab-crawler

Crawler source type

Data stores

Data store
S3: s3://dmslab-stu...

IAM Role

Schedule

Output

Review all steps

Add a data store

Choose a data store
S3

Crawl data in
 Specified path

Include path
s3://dmslab-student-dmslabs3bucket-1xby1wp8fe8iq/tickets

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

Exclude patterns (optional)

Back Next

Chosen data stores
S3: s3://dmslab-stud...

9. On the **Add another data store page**, select **No.** and Click **Next**.

Add crawler

Crawler info
glue-lab-crawler

Crawler source type

Data stores

Data store
S3: s3://dmslab-stu...

IAM Role

Schedule

Output

Review all steps

Add another data store

Yes
 No

Back Next

Chosen data stores
S3: s3://dmslab-stud...

10. On the **Choose an IAM role** page, make the following selections:

- a. Select **Choose an existing IAM role**.
- b. For **IAM role**, select **<stackname>-GlueLabRole-<RandomString>** pre-created for you. For example “**dmslab-student-GlueLabRole-ZOQDII7JTBUM**”

11. Click **Next**.

Add crawler

Crawler info
glue-lab-crawler

Crawler source type

Data stores

Data store
S3: s3://dmslab-stu...

IAM Role

Schedule

Output

Review all steps

Choose an IAM role

The IAM role allows the crawler to run and access your Amazon S3 data stores. [Learn more](#)

Update a policy in an IAM role
 Choose an existing IAM role
 Create an IAM role

IAM role [?](#)
dmslab-student-GlueLabRole-ZOQDII7JTBUM

This role must provide permissions similar to the AWS managed policy, **AWSGlueServiceRole**, plus access to your data stores.
• s3://dmslab-student-dmslabs3bucket-woti4bf73cw3

You can also create an IAM role on the [IAM console](#).

Back Next

Lab 2. ETL with AWS Glue

12. On the Create a schedule for this crawler page, for Frequency, select **Run on demand** and Click **Next**.

Add crawler

Create a schedule for this crawler

Frequency: Run on demand

Back Next

Crawler info: glue-lab-crawler
Crawler source type: Data stores
Data store: S3: s3://dmslab-stu...
IAM Role: arn:aws:iam::341259728059:role/service-role/dmslab-student-GlueLabRole-ZQQDII7JTBUM
Schedule: Run on demand
Output:
Review all steps:

13. On the Configure the crawler's output page, click **Add database** to create a new database for our Glue Catalogue.

Add crawler

Configure the crawler's output

Database: Choose a database to contain tables
Add database (highlighted)
Prefix added to tables (optional): Type a prefix added to table names

Grouping behavior for S3 data (optional)
Configuration options (optional)

Back Next

Crawler info: glue-lab-crawler
Crawler source type: Data stores
Data store: S3: s3://dmslab-stu...
IAM Role: arn:aws:iam::341259728059:role/service-role/dmslab-student-GlueLabRole-ZQQDII7JTBUM
Schedule: Run on demand
Output:
Review all steps:

14. Enter **ticketdata** as your database name and click **create**

Add crawler

Add database

Database name: ticketdata
Description and location (optional)

Create

Grouping behavior for S3 data (optional)
Configuration options (optional)

Back Next

Crawler info: glue-lab-crawler
Crawler source type: Data stores
Data store: S3: s3://dmslab-stu...
IAM Role: arn:aws:iam::341259728059:role/service-role/dmslab-student-GlueLabRole-ZQQDII7JTBUM
Schedule: Run on demand
Output:
Review all steps:

15. For **Prefix added to tables (optional)**, leave the field empty.

16. For **Configuration options (optional)**, select **Add new columns only** and keep the remaining default configuration options and Click **Next**.

Lab 2. ETL with AWS Glue

Add crawler

Configure the crawler's output

Database: ticketdata

Prefix added to tables (optional): Type a prefix added to table names

Grouping behavior for S3 data (optional)

Configuration options (optional)

During the crawler run, all schema changes are logged.

When the crawler detects schema changes in the data store, how should AWS Glue handle table updates in the data catalog?

- Update the table definition in the data catalog.
- Add new columns only.
- Ignore the change and don't update the table in the data catalog.

How should AWS Glue handle deleted objects in the data store?

- Delete tables and partitions from the data catalog.
- Ignore the change and don't update the table in the data catalog.
- Mark the table as deprecated in the data catalog.

Back **Next**

17. Review the summary page noting the Include path and Database output and Click **Finish**. The crawler is now ready to run.

Edit crawler

Crawler info

- Name: glue-lab-crawler
- Description: DMS Lab full initial load crawler
- Tags: -

Data stores

- Data store: S3
- Include path: s3://dmslab-student-dmslab-dsbucket-xg1hdq80bs
- Exclude patterns: -

IAM role

- IAM role: arn:aws:iam::665953140268:role/service-role/dmslab-student-GlueLabRole-14R6WFBWGZ4MB

Schedule

- Schedule: Run on demand

Output

- Database: ticketdata
- Prefix added to tables (optional): ticketdata
- Create a single schema for each S3 path: false

Configuration options

Back **Finish**

18. Click **Run it now**.

AWS Glue

Crawlers

Crawler glue-lab-crawler was created to run on demand. **Run it now!**

Name	Schedule	Catalog type	Status	Logs	Last runtime	Median runtime	Tables updated	Tables added
glue-lab-crawler	Glue	Ready	0 secs	0 secs	0	0	0	0

User preferences

Crawler will change status from starting to stopping, wait until crawler comes back to ready state (the process will take a few minutes), you can see that it has created 15 tables.

Lab 2. ETL with AWS Glue

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.

Crawler "glue-lab-crawler" completed and made the following changes: 15 tables created, 0 tables updated. See the tables created in database ticketdata.

Name	Schedule	Catalog type	Status	Logs	Last runtime	Median runtime	Tables updated	Tables added
glue-lab-crawler		Glue	Ready	Logs	1 min	1 min	0	15

19. In the AWS Glue navigation pane, click **Databases > Tables**. You can also click the **ticketdata** database to browse the tables.

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

Name	Database	Location	Classification	Last updated	Deprecated
mib_data	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
name_data	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
nfl_data	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
nfl_stadium_data	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
person	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:48 PM ...	
player	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
seat	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
seat_type	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
sport_division	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
sport_league	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
sport_location	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
sport_team	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
sporting_event	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
sporting_event_ticket	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	
ticket_purchase_hist	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM ...	

Data Validation Exercise

1. Within the Tables section of your **ticketdata** database, click the person table.

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

Name	Database	Location	Classification	Last updated	Deprecated
mib_data	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM UTC-5	
name_data	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM UTC-5	
nfl_data	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM UTC-5	
nfl_stadium_data	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM UTC-5	
person	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:48 PM UTC-5	
player	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM UTC-5	
seat	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM UTC-5	
seat_type	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM UTC-5	
sport_division	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM UTC-5	
sport_league	ticketdata	s3://dmslab-student-dm... csv		10 January 2020 1:37 PM UTC-5	

You may have noticed that some tables (such as person) have column headers such as col0,col1,col2,col3. In absence of headers or when the crawler cannot determine the header type, default column headers are specified.

This exercise uses the person table in an example of how to resolve this issue.

2. Click **Edit Schema** on the top right side.

Lab 2. ETL with AWS Glue

AWS Glue Tables > person Edit table Delete table Last updated 10 Jan 2020 Table Version (Current version) ▾ View properties Compare versions Edit schema

Table properties

Name	person
Description	ticketdata
Database	ticketdata
Classification	csv
Location	s3://dmrslab-student-dmstables3bucket-xg1hdyqf0bs/tickets/dmr_sample/person/
Connection	
Deprecated	No
Last updated	Fri Jan 10 13:37:23 GMT-500 2020
Input format	org.apache.hadoop.mapred.TextInputFormat
Output format	org.apache.hadoop.hive.io.HiveIgnoreKeyTextOutputFormat
Serde serialization lib	org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
Serde parameters	field.delim = , sizeKey 366585890 objectCount 1 UPDATED_BY_CRAWLER glue-lab-crawler CrawlerSchemaSerializerVersion 1.0 recordCount 9164647 averageRecordSize 40 CrawlerSchemaDeserializerVersion 1.0 compressionType none columnsOrdered true areColumnsQuoted false delimiter , typeOfData file

Schema

Column name	Data type	Partition key	Comment
1 col0	string		
2 col1	string		
3 col2	string		
4 col3	string		

3. In the Edit Schema section, double-click **col0** (column name) to open edit mode. Type “id” as the column name.
4. Repeat the preceding step to change the remaining column names to match those shown in the following figure.

AWS Glue Tables > person Add column Last updated 10 Jan 2020 Table Version (Current version) ▾ Edit schema Cancel Save Showing: 1 - 4 of 4 ▶ ▷

Column name	Data type	Key	Comment
1 id	string		X
2 full_name	string		X
3 last_name	string		X
4 first_name	string		X

5. Click **Save**.

Data ETL Exercise

1. Go to the Glue console, in the left navigation pane, under **ETL** click **Jobs**, and then click **Add job**.

AWS Glue Data catalog Databases Tables Connections Crawlers Classifiers Settings ETL Jobs A job is your business logic required to perform extract, transform and load (ETL) work. Job runs are initiated by triggers which can be scheduled or driven by events. User preferences Showing: 0 - 0 ▶ ▷

Add job Action ▾ Filter by attributes

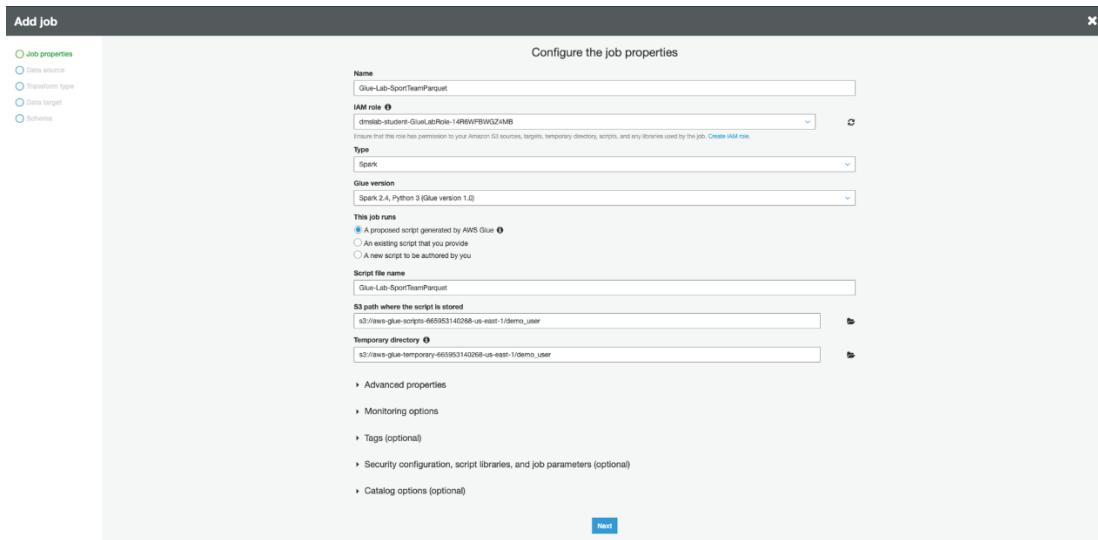
Name	ETL language	Script location	Last modified	Job bookmark
You don't have any jobs defined yet.				

Add job

2. On the Job properties page, make the following selections:
 - a. For **Name**, type “Glue-Lab-SportTeamParquet”

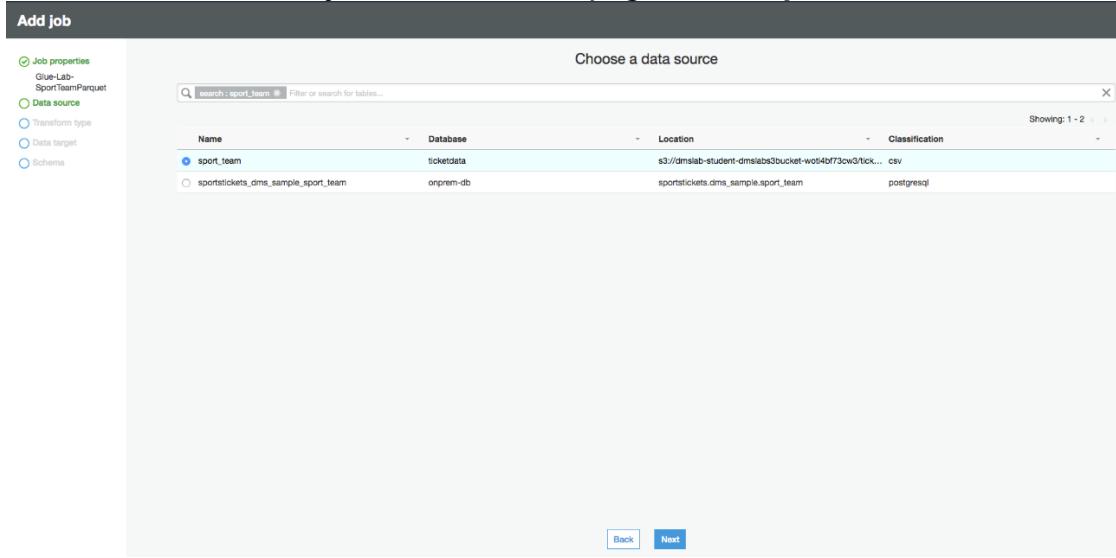
Lab 2. ETL with AWS Glue

- b. For **IAM role**, choose existing role e.g. "xxxx-GlueLabRole-xxx"
- c. For **Type**, Select "Spark"
- d. For **Glue Version**, select "Spark 2.4, Python 3(Glue version 1.0)" or whichever is the latest version
- e. For **This job runs**, select "A proposed script generated by AWS Glue".
- f. For **Script file name**, type **Glue-Lab-SportTeamParquet**.
- g. For **S3 path where the script is stored**, provide a unique Amazon S3 path to store the scripts. (You can keep the default for this lab.)
- h. For **Temporary directory**, provide a unique Amazon S3 directory for a temporary directory. (You can keep the default for this lab.)



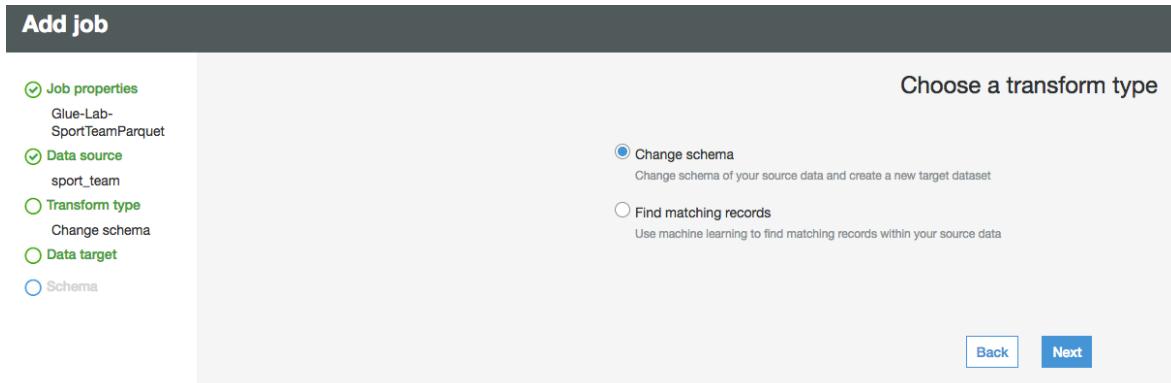
3. Click **Next**

4. On the Choose your data sources page, select **sport_team** and Click **Next**.



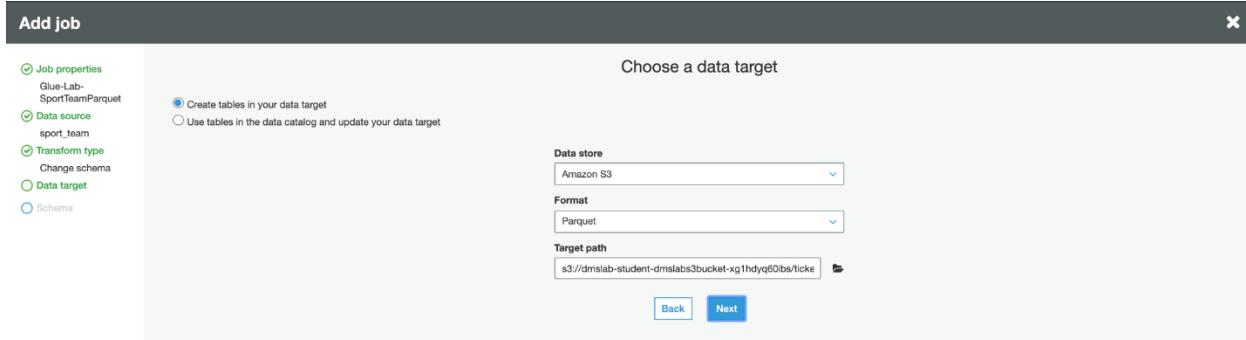
5. On the **Choose a transformation type** page, select **change schema**

Lab 2. ETL with AWS Glue

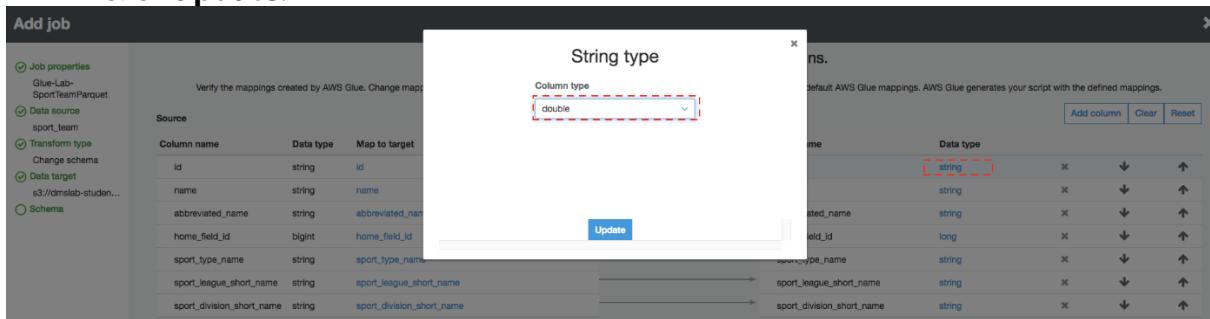


6. On the Choose your data targets page, select **Create tables in your data target**.
7. For Data store, select **Amazon S3**.
8. For Format, select **Parquet**.
9. For **Target path**, click the **folder icon** and choose the s3 bucket, then append **/tickets/dms_parquet/sport_team** to it, making the target path look like **s3://xxx-dmslab-student-dmslabs3bucket-xxx/tickets/dms_parquet/sport_team** – Glue will create necessary folders

10. Click **Next**.



11. Click the target **Data type** to edit the schema mapping for the **id** column. In **String type** pop-up window Select **double** from **Column type** drop down and click **update**.



Lab 2. ETL with AWS Glue

Map the source columns to target columns.

Verify the mappings created by AWS Glue. Change mappings by choosing other columns with **Map to target**. You can **Clear** all mappings and **Reset** to default AWS Glue mappings. AWS Glue generates your script with the defined mappings.

Source		Target			
Column name	Data type	Map to target	Column name	Data type	
<code>id</code>	string	<code>id</code>	<code>id</code>	double	X ↓ ↑
<code>name</code>	string	<code>name</code>	<code>name</code>	string	X ↓ ↑
<code>abbreviated_name</code>	string	<code>abbreviated_name</code>	<code>abbreviated_name</code>	string	X ↓ ↑
<code>home_field_id</code>	bigint	<code>home_field_id</code>	<code>home_field_id</code>	long	X ↓ ↑
<code>sport_type_name</code>	string	<code>sport_type_name</code>	<code>sport_type_name</code>	string	X ↓ ↑
<code>sport_league_short_name</code>	string	<code>sport_league_short_name</code>	<code>sport_league_short_name</code>	string	X ↓ ↑
<code>sport_division_short_name</code>	string	<code>sport_division_short_name</code>	<code>sport_division_short_name</code>	string	X ↓ ↑

Add column Clear Reset

Back Save job and edit script

12. Click **Save job and edit script**.

13. View the job. (This screen provides you with the ability to customize this script as required.) Click **Save** and then **Run Job**.

Job: Glue-Lab-SportTeamParquet Action **Save** Run job Generate diagram

Insert template at cursor Source Target Target Location Transform Spigot

```

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 # @params: {JOB_NAME}
9 args = getResolvedOptions(sys.argv, ['JOB_NAME'])
10
11 sc = SparkContext()
12 glueContext = GlueContext(sc)
13 spark = glueContext.spark_session
14 job = Job(glueContext)
15 job.init(args['JOB_NAME'], args)
16 args["datasource0"] = "titledata"
17 # @args: {datasource0: "titledata", table_name: "sport_team", transformation_ctx: "datasource0"}
18 # @return: datasource0
19 # @inputs: []
20 datasource0 = glueContext.create_dynamic_frame.from_catalog(database = "titledata", table_name = "sport_team", transformation_ctx = "datasource0")
21 # @type: ApplyMapping
22 # @args: {mapping: [{"id": "string", "id": "double"}, {"name": "string", "name": "string"}, {"abbreviated_name": "string", "abbreviated_name": "string"}, {"home_field_id": "long", "home_field_id": "long"}, {"sport_type_name": "string", "sport_type_name": "string"}, {"sport_league_short_name": "string", "sport_league_short_name": "string"}, {"sport_division_short_name": "string", "sport_division_short_name": "string"}]
23 # @return: applymapping1
24 applymapping1 = datasource0.apply_mapping(mapping = mapping, transformation_ctx = "applymapping1")
25 # @type: ResolveChoice
26 # @args: {choice_struct: "mke_struct", transformation_ctx: "resolvechoice1"}
27 choice_struct = "mke_struct"
28 # @return: resolvechoice1
29 # @inputs: [frame = applymapping1]
30 resolvechoice1 = ResolveChoice.apply(frame = applymapping1, choice = "mke_struct", transformation_ctx = "resolvechoice1")
31 # @type: DropNullFields
32 # @args: {dropnullfields3: "dropnullfields3"}
33 # @return: dropnullfields3
34

```

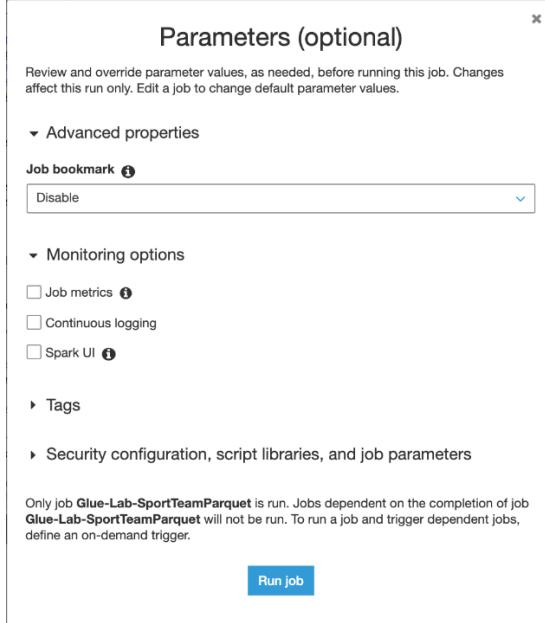
Log Schema

s3://dimalab-student-dimalab280c
Path ket-w0t4bf3c3w3h/tickets/dms_parquet/sport_team

14. In **Parameters** option,

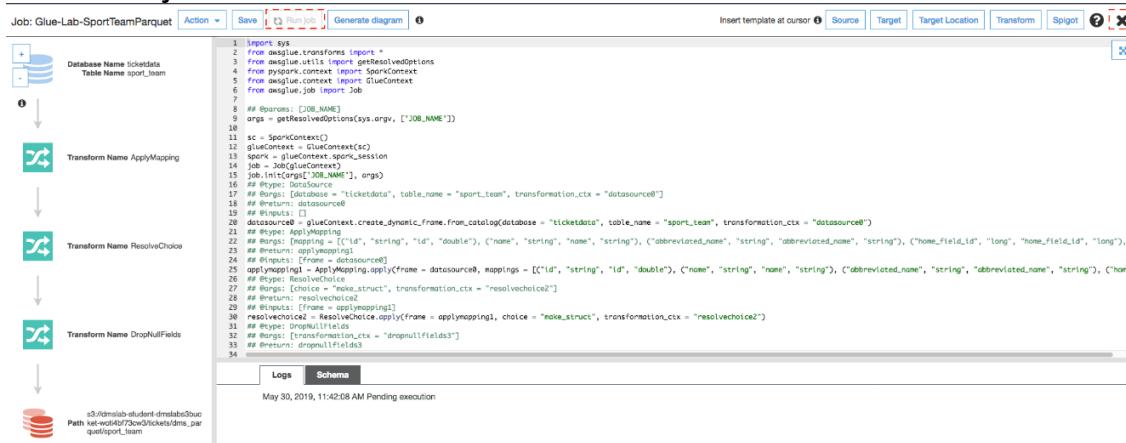
- you can leave **Job bookmark** as **Disable**. AWS Glue tracks data that has already been processed during a previous run of an ETL job by persisting state information from the job run.
- You can leave the **Job metrics** option **Unchecked**. You can collect metrics about AWS Glue jobs and visualize them on the AWS Glue with job metrics.

Lab 2. ETL with AWS Glue



15. Click Run Job

16. You will see job in now running as Run job button became greyed out. Click the cross button located in top right corner to close the window to return to the ETL jobs.



17. Click your job to view history and verify that it ran successfully.

Name	Type	ETL language	Script location	Last modified	Job bookmark									
Glue-Lab-SportTeamParquet	Spark	python	s3://aws-glue-scri...	11 March 2020 3:17 PM UTC-7	Disable									
History	Details	Script	Metrics											
View run metrics					Showing: 1 - 1 < >									
Run ID	Retry attempt	Run status	Error	Logs	Error logs	Glue version	Maximum capacity	Triggered by	Start time	End time	Execution time	Timeout	Delay	Job run input
jr_e37b56aa03cd3...	-	Running		Logs	Error logs	1.0	5		11 Mar...	0 secs	2880 mins			s3://aws-glue-tem...

(Optional) If you plan to continue with other labs outside of this event (for example, the [Athena lab](#)), you'll have to complete the rest of this section to create more ETL Jobs to transform additional tables to parquet, changing their schema as per

Lab 2. ETL with AWS Glue

instructions below., otherwise you can proceed to section “**Create Glue Crawler for Parquet Files**”

To enable us to join data, we will also update the target data types in the schema. If below **Table1** is indicating need Schema changes as “Yes”. Refer **Table 2** find out column which need to changes with source and target data type during ETL job creation.

Table 1:

Job Name & Script Filename	Source Table	S3 Target Path	Need Schema Change?
Glue-Lab-SportLocationParquet	sport_location	tickets/dms_parquet/sport_location	No
Glue-Lab-SportingEventParquet	sporting_event	tickets/dms_parquet/sporting_event	Yes
Glue-Lab-SportingEventTicketParquet	sporting_event_ticket	tickets/dms_parquet/sporting_event_ticket	Yes
Glue-Lab-PersonParquet	person	tickets/dms_parquet/person	Yes

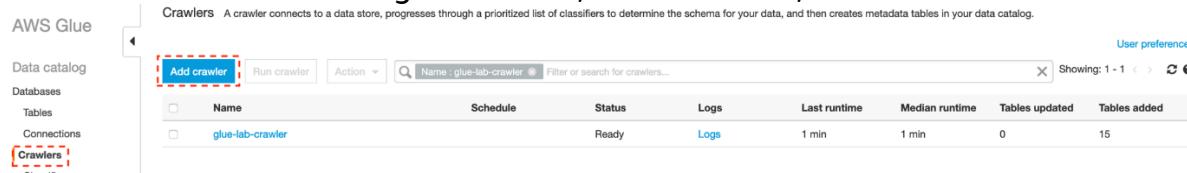
Table 2:

Job Name	Table	Column	Source Data Type	Target Data Type
Glue-Lab-SportingEventParquet	sporting_event	start_date_time	STRING	TIMESTAMP
Glue-Lab-SportingEventParquet	sporting_event	start_date	STRING	DATE
Glue-Lab-SportingEventTicketParquet	sporting_event_ticket	id	STRING	DOUBLE
Glue-Lab-SportingEventTicketParquet	sporting_event_ticket	sporting_event_id	STRING	DOUBLE
Glue-Lab-SportingEventTicketParquet	sporting_event_ticket	ticketholder_id	STRING	DOUBLE
Glue-Lab-PersonParquet	person	id	STRING	DOUBLE

Once these jobs have completed, we can create a crawler to index these parquet files.

Create Glue Crawler for Parquet Files

1. In the AWS Glue navigation menu, click **Crawlers**, and then click **Add crawler**.



2. For Crawler name, type “glue-lab-parquet-crawler” and Click Next.

The screenshot shows the 'Add crawler' configuration page. On the left, there are tabs for 'Crawler info' (selected), 'Crawler source type' (set to 'Data Stores'), 'Data stores', 'IAM Role', and 'Schedule'. The main area has a heading 'Add information about your crawler' and a 'Crawler name' field containing 'glue-lab-parquet-crawler'. Below it is a note: 'Tags, description, security configuration, and classifiers (optional)'. At the bottom is a 'Next' button.

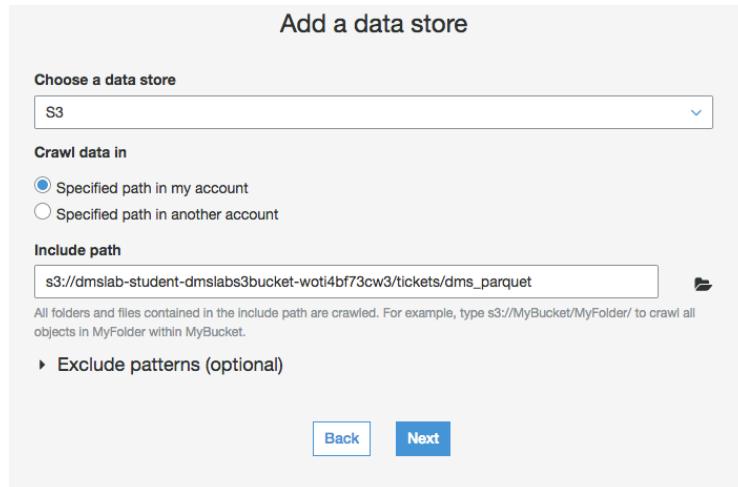
3. In next screen Specify crawler source type, select Data Stores as choice for Crawler source type and click Next.

4. In Add a data store screen

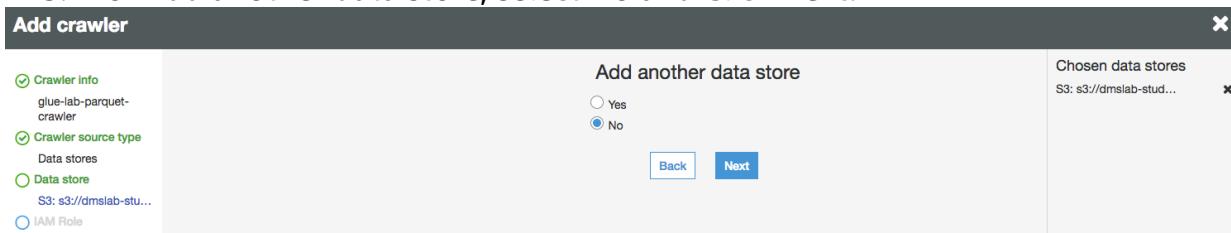
- a. For Choose a data store, select “S3”.
- b. For Crawl data in, select “Specified path in account”.

Lab 2. ETL with AWS Glue

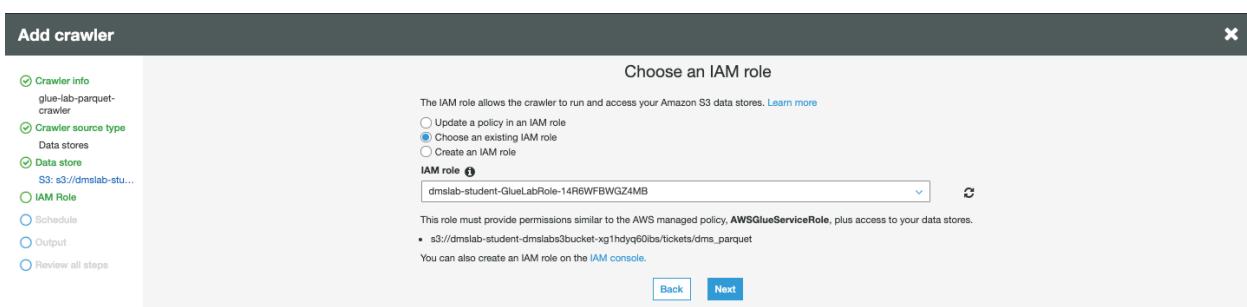
- c. For Include path, specify the S3 Path (Parent Parquet folder) that contains the nested parquet files e.g., s3://xxx-dmslabs3bucket-xxx/tickets/dms_parquet
- d. Click **Next**.



5. For Add another data store, select **No** and Click **Next**.

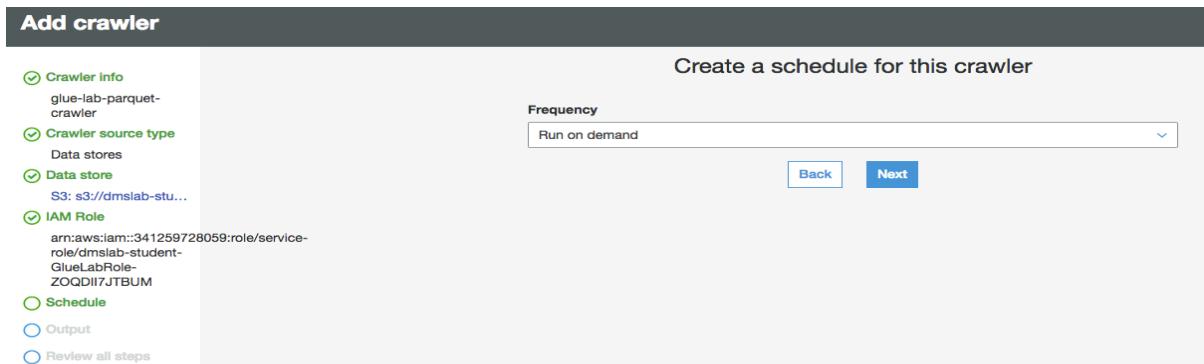


6. On the Choose an IAM role page, select **Choose an existing IAM role**.
For IAM role, select the existing role "xxx-GlueLabRole-xxx" and Click **Next**.

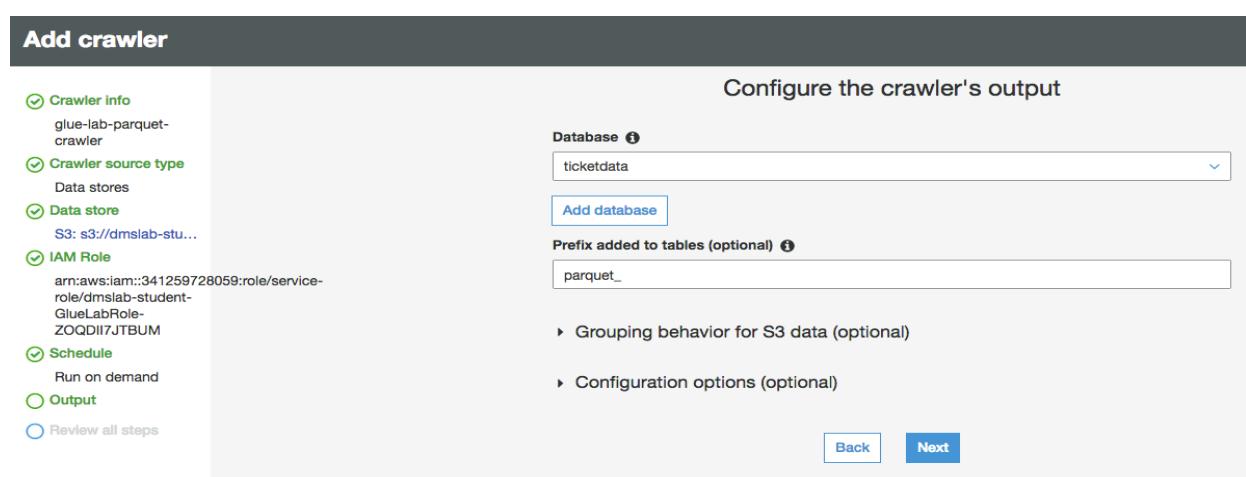


7. For Frequency, select "Run On Demand" and Click **Next**.

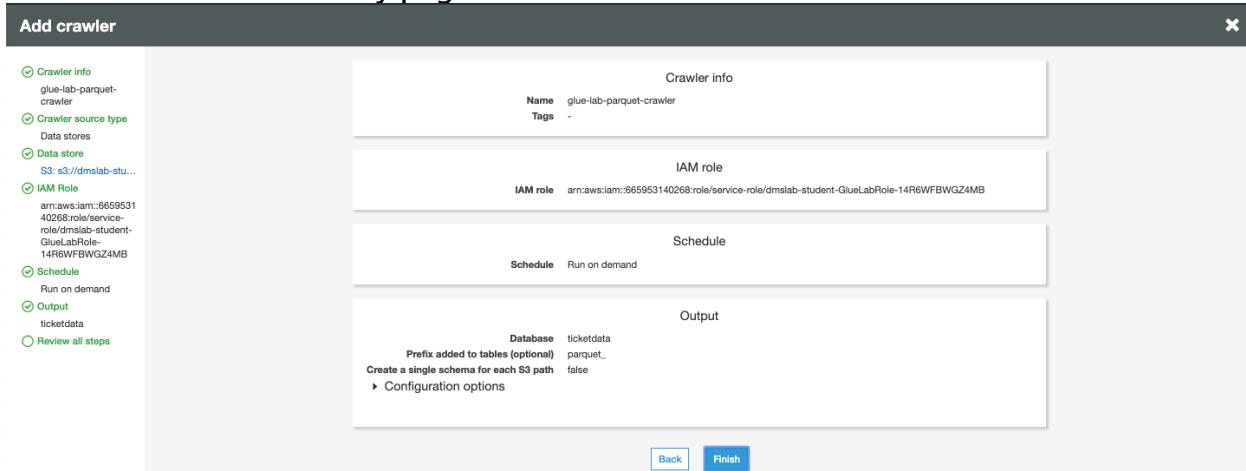
Lab 2. ETL with AWS Glue



8. For the crawler's output database, choose your existing database which you created earlier e.g. "**ticketdata**"
9. For the **Prefix added to tables** (optional), type "**parquet_**"



10. Review the summary page and click **Finish**.



11. On the notification bar, click **Run it now**. Once your crawler has finished running, you should report that tables were added, 1 to 5, depending on how many parquet ETL conversions you set up in the previous section

Lab 2. ETL with AWS Glue

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.

Crawler "glue-lab-parquet-crawler" completed and made the following changes: 5 tables created, 0 tables updated. See the tables created in database ticketdata.

User preferences

Showing: 1 - 3

Name	Schedule	Catalog type	Status	Logs	Last runtime	Median runtime	Tables updated	Tables added
glue-lab-cdc-cra...		Glue	Ready	Logs	1 min	1 min	0	2
glue-lab-crawler		Glue	Ready	Logs	1 min	1 min	0	15
glue-lab-parquet...		Glue	Ready	Logs	1 min	1 min	0	5

Confirm you can see the tables:

1. In the left navigation pane, click **Tables**.
2. Add the filter "parquet" to return the newly created tables.

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

Add tables Action Database : ticketdata Filter or search for tables...

Name	Database	Location	Classification	Last updated	Deprecated
mib_data	ticketdata	s3://dmslab-student-dmslabs3buck...	csv	10 January 2020 1:37 PM UTC-5	
name_data	ticketdata	s3://dmslab-student-dmslabs3buck...	csv	10 January 2020 1:37 PM UTC-5	
nfl_data	ticketdata	s3://dmslab-student-dmslabs3buck...	csv	10 January 2020 1:37 PM UTC-5	
nfl_stadium_data	ticketdata	s3://dmslab-student-dmslabs3buck...	csv	10 January 2020 1:37 PM UTC-5	
parquet_person	ticketdata	s3://dmslab-student-dmslabs3buck...	parquet	23 January 2020 1:49 PM UTC-5	
parquet_person_annotation	ticketdata	s3://dmslab-student-dmslabs3buck...	parquet	23 January 2020 1:49 PM UTC-5	
parquet_sport_team	ticketdata	s3://dmslab-student-dmslabs3buck...	parquet	23 January 2020 1:49 PM UTC-5	
parquet_sporting_event	ticketdata	s3://dmslab-student-dmslabs3buck...	parquet	23 January 2020 1:49 PM UTC-5	
parquet_sporting_event_ticket	ticketdata	s3://dmslab-student-dmslabs3buck...	parquet	23 January 2020 1:49 PM UTC-5	
person	ticketdata	s3://dmslab-student-dmslabs3buck...	csv	10 January 2020 1:48 PM UTC-5	

PART B: Glue Job Bookmark (Optional):

****Pre-requisite: Completion of CDC part of DMS Lab ****

Step 1: Create Glue Crawler for ongoing replication (CDC Data)

Now, let's repeat this process to load the data from change data capture.

1. On the AWS Glue menu, select 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.

User preferences

Showing: 0 - 0

Name	Schedule	Status	Logs	Last runtime	Median runtime	Tables updated	Tables added
You don't have any crawlers yet.							

Add crawler

2. Click **Add crawler**.
3. Enter the crawler name for ongoing replication. This name should be descriptive and easily recognized (e.g., "**glue-lab-cdc-crawler**").
4. Optionally, enter the description. This should also be descriptive and easily recognized and Click **Next**.

Lab 2. ETL with AWS Glue

Add crawler

Add information about your crawler

Crawler name
glue-lab-cdc-crawler

Tags, description, security configuration, and classifiers (optional)

Next

5. Choose Data Stores as Crawler Source Type and Click Next

Add crawler

Specify crawler source type

Choose Existing catalog tables to specify catalog tables as the crawler source. The selected tables specify the data stores to crawl. This option doesn't support JDBC data stores.

Crawler source type
 Data stores
 Existing catalog tables

Back Next

The screenshot shows the 'Specify crawler source type' step. On the left, there's a sidebar with options: Crawler info (checked), Crawler source type (checked), Data stores (unchecked), Data store (checked), IAM Role (unchecked), Schedule (unchecked), Output (unchecked), and Review all steps (unchecked). On the right, it says 'Specify crawler source type' and 'Choose Existing catalog tables to specify catalog tables as the crawler source. The selected tables specify the data stores to crawl. This option doesn't support JDBC data stores.' Below that is a radio button group for 'Crawler source type' with 'Data stores' selected. At the bottom are 'Back' and 'Next' buttons.

6. On the Add a data store page, make the following selections:

- For **Choose a data store**, click the drop-down box and select **S3**.
- For **Crawl data in**, select **Specified path in my account**.
- For **Include path**, enter the target folder for your DMS ongoing replication, e.g., "**s3://xxx-dmslabs3bucket-xxx/cdc/dms_sample**"

7. Click Next.

Add crawler

Add a data store

Choose a data store
S3

Crawl data in
 Specified path in my account
 Specified path in another account

Include path
s3://dmslab-student-dmslabs3bucket-wotl4bf73cw3/cdc/dms_sample

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

Exclude patterns (optional)

Back Next

The screenshot shows the 'Add a data store' step. On the left, there's a sidebar with options: Crawler info (checked), Crawler source type (checked), Data stores (unchecked), Data store (checked), IAM Role (unchecked), Schedule (unchecked), Output (unchecked), and Review all steps (unchecked). On the right, it says 'Add a data store' and 'Choose a data store' with 'S3' selected. Below that is a section for 'Crawl data in' with 'Specified path in my account' selected. Under 'Include path', there's a text input with 's3://dmslab-student-dmslabs3bucket-wotl4bf73cw3/cdc/dms_sample'. A note below says 'All folder and files contained in the include path are crawled. For example, type s3://MyBucket/MyFolder/ to crawl all objects in MyFolder within MyBucket.'. At the bottom are 'Back' and 'Next' buttons.

8. On the Add another data store page, select No and Click Next.

Add crawler

Add another data store

Chosen data stores
S3: s3://dmslab-stud...

Yes

 No

Back Next

The screenshot shows the 'Add another data store' step. On the left, there's a sidebar with options: Crawler info (checked), Crawler source type (checked), Data stores (unchecked), Data store (checked), IAM Role (unchecked), Schedule (unchecked), Output (unchecked), and Review all steps (unchecked). On the right, it says 'Add another data store' and has a radio button group for 'Yes' and 'No', with 'No' selected. At the bottom are 'Back' and 'Next' buttons. To the right of the main area, there's a panel titled 'Chosen data stores' with 'S3: s3://dmslab-stud...' listed.

9. On the Choose an IAM role page, make the following selections:

- Select **Choose an existing IAM role**.

Lab 2. ETL with AWS Glue

- b. For IAM role, select **xxx-GlueLabRole-xxx**. E.g. "dmslab-student-GlueLabRole-ZOQDII7JTBUM"

10. Click **Next**.

Choose an IAM role

The IAM role allows the crawler to run and access your Amazon S3 data stores. [Learn more](#)

Update a policy in an IAM role
 Choose an existing IAM role
 Create an IAM role

IAM role [dmslab-student-GlueLabRole-14R6WFBWGZ4MB](#)

This role must provide permissions similar to the AWS managed policy, [AWSGlueServiceRole](#), plus access to your data stores.

- s3://dmslab-student-dmslabs3bucket-xg1hdqg0lbt/cdc/dms_sample

You can also create an IAM role on the [IAM console](#).

Back Next

11. On the Create a schedule for this crawler page, for Frequency, select **Run on demand** and Click **Next**.

Create a schedule for this crawler

Frequency [Run on demand](#)

Back Next

12. On the Configure the crawler's output page, select the existing **Database** for crawler output (e.g., "ticketdata").

13. For **Prefix added to tables (optional)**, specify "cdc_".

14. For Configuration options (optional), keep the default selections and click **Next**.

Configure the crawler's output

Database [ticketdata](#)

Add database

Prefix added to tables (optional) [Type a prefix added to table names](#)

Grouping behavior for S3 data (optional)

Configuration options (optional)

When the crawler detects schema changes in the data store, how should AWS Glue handle table updates in the data catalog?

Update the table definition in the data catalog.
 Add new columns only.
 Ignore the change and don't update the table in the data catalog. [i](#)

Update all new and existing partitions with metadata from the table. [i](#)

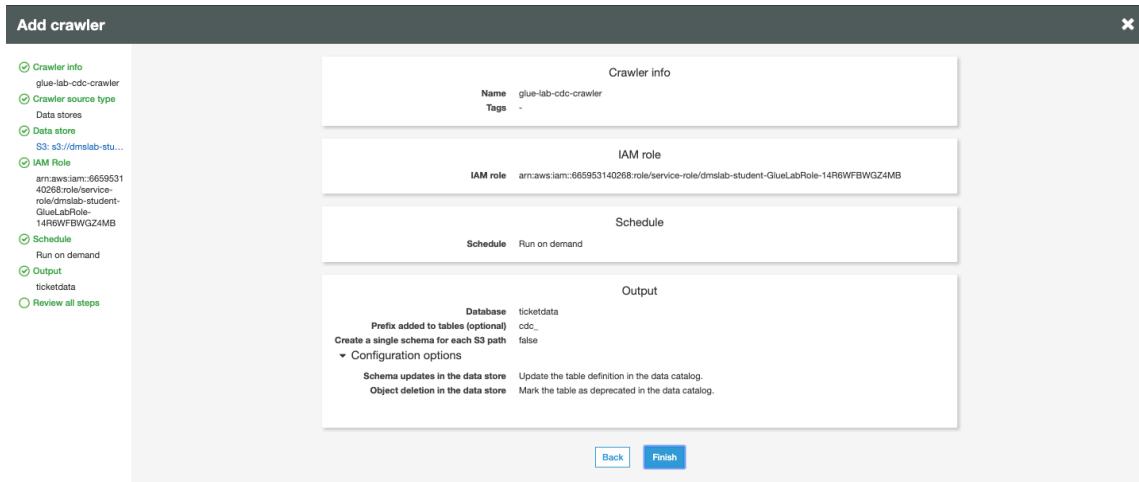
How should AWS Glue handle deleted objects in the data store?

Delete tables and partitions from the data catalog.
 Ignore the change and don't update the table in the data catalog.
 Mark the table as deprecated in the data catalog. [i](#)

Back Next

15. Review the summary page noting the **Include path** and **Database target** and Click **Finish**. The crawler is now ready to run.

Lab 2. ETL with AWS Glue



16. Click Run it now.

AWS Glue

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

Name	Schedule	Catalog type	Status	Logs	Last runtime	Median runtime	Tables updated	Tables added
glue-lab-cdc-cra...	Glue	Ready	0 secs	0 secs	0	0	0	0
glue-lab-crawler	Glue	Ready	Logs	1 min	1 min	0	0	15

17. When the crawler is completed, you can see it has "Status" as Ready, Crawler will change status from starting to stopping, wait until crawler comes back to ready state, you can see that it has created 2 tables.

AWS Glue

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

Name	Schedule	Catalog type	Status	Logs	Last runtime	Median runtime	Tables updated	Tables added
glue-lab-cdc-cra...	Glue	Ready	Logs	1 min	1 min	0	0	2
glue-lab-crawler	Glue	Ready	Logs	1 min	1 min	0	0	15

18. Click the database name (e.g., "ticketdata") to browse the tables. Specify "cdc" as the filter to list only newly imported tables.

AWS Glue

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

Name	Database	Location	Classification	Last updated	Deprecated
cdc_sporting_event_ticket	ticketdata	s3://dmslab-student-dmslabs3buck...	csv	23 January 2020 4:38 PM UTC-5	
cdc_ticket_purchase_hist	ticketdata	s3://dmslab-student-dmslabs3buck...	csv	23 January 2020 4:38 PM UTC-5	
mlb_data	ticketdata	s3://dmslab-student-dmslabs3buck...	csv	10 January 2020 1:37 PM UTC-5	
name_data	ticketdata	s3://dmslab-student-dmslabs3buck...	csv	10 January 2020 1:37 PM UTC-5	
nfl_data	ticketdata	s3://dmslab-student-dmslabs3buck...	csv	10 January 2020 1:37 PM UTC-5	
nfl_stadium_data	ticketdata	s3://dmslab-student-dmslabs3buck...	csv	10 January 2020 1:37 PM UTC-5	
parquet_person	ticketdata	s3://dmslab-student-dmslabs3buck...	parquet	23 January 2020 1:49 PM UTC-5	
parquet_sport_location	ticketdata	s3://dmslab-student-dmslabs3buck...	parquet	23 January 2020 1:49 PM UTC-5	
parquet_sport_team	ticketdata	s3://dmslab-student-dmslabs3buck...	parquet	23 January 2020 1:49 PM UTC-5	

Lab 2. ETL with AWS Glue

Step 2: Create a Glue Job with Bookmark Enabled

1. On the left-hand side of Glue Console, click on Jobs and then Click on Add Job

The screenshot shows the AWS Glue Jobs console. On the left, there's a sidebar with 'AWS Glue' at the top, followed by 'Data catalog', 'Databases', and 'Tables'. In the main area, there's a heading 'Jobs' with a sub-instruction: 'A job is your business logic required to perform extract, transform and load (ETL) work. Job runs are initiated by triggers which can be scheduled or driven by events.' Below this is a search bar with 'Add job' and 'Action' buttons, and a 'Filter by tags and attributes' dropdown. At the bottom, there's a table header with columns 'Name', 'Type', 'ETL language', 'Script location', 'Last modified', and 'Job bookmark'. A note says 'Showing 1 - 9'.

2. On the Job properties page, make the following selections:
 - a. For **Name**, type **Glue-Lab-TicketHistory-Parquet-with-bookmark**.
 - b. For **IAM role**, choose existing role "xxx-GlueLabRole-xxx"
 - c. For **Type**, Select **Spark**
 - d. For **Glue Version**, select **Spark 2.4, Python 3 (Glue version 1.0)** or whichever is the latest version
 - e. For **This job runs**, select **A proposed script generated by AWS Glue**.
 - f. For **Script file name**, type **Glue-Lab-TicketHistory-Parquet-with-bookmark**.
 - g. For **S3 path where the script is stored**, provide a unique Amazon S3 path to store the scripts. (You can keep the default for this lab.)
 - h. For **Temporary directory**, provide a unique Amazon S3 directory for a temporary directory. (You can keep the default for this lab.)
3. Expand the **Advanced properties** section. For Job bookmark, select **Enable** from the drop-down option.
4. Expand on the **Monitoring** options, enable **Job metrics**.
5. Click **Next**

The screenshot shows the 'Configure the job properties' dialog box. On the left, there's a sidebar with 'Job properties' selected, showing options like 'Glue Lab', 'TicketHistory', 'Parquet-with-bookmark', 'Data source', 'Transform type', 'Data target', and 'Regions'. The main area has sections for 'Name' (set to 'Glue-Lab-TicketHistory-Parquet-with-bookmark'), 'IAM role' (set to 'arn:aws:iam::123456789012:role/GlueLabRole-123456789012'), 'Type' (set to 'Spark'), 'Glue version' (set to 'Spark 2.4, Python 3 (Glue Version 1.0)'), 'This job runs' (radio button selected for 'A proposed script generated by AWS Glue'), 'Script file name' (set to 'Glue-Lab-TicketHistory-Parquet-with-bookmark'), 'S3 path where the script is stored' (set to 's3://aws-glue-scripts-66593142098-us-east-1/debutut'), 'Temporary directory' (set to 's3://aws-glue-temporary-66593142098-us-east-1/debutut'). Below these are sections for 'Advanced properties' (Job bookmark set to 'Enable') and 'Monitoring options' (Job metrics checked). At the bottom, there's a 'Tags (optional)' section.

6. In **Choose a data source**, select **cdc_ticket_purchase_hist** as we are generating new data entries for **ticket_purchase_hist** table. Click **Next**

Lab 2. ETL with AWS Glue

Add job

Choose a data source

Name	Database	Location	Classification
bookmark_parquet_ticket_purchase_history	ticketdata	s3://dmslab-student-dmlabs3bucket-xg1hydg0ba/cdc_bookmark/ticket...	parquet
cdc_sporting_event_ticket	ticketdata	s3://dmslab-student-dmlabs3bucket-xg1hydg0ba/cdc/dms_sample/sp...	csv
cdc_ticket_purchase_hist	ticketdata	s3://dmslab-student-dmlabs3bucket-xg1hydg0ba/cdc/dms_sample/tic...	csv
clickstream_dts	processed-data	s3://rewinddataset-dsbatch/Clickstream_data/	json
csv_clickstream_data	processed-data	s3://processed-dsbatch/Clickstream_data/	csv

7. In Choose a transform type, select Change Schema and Click Next

Add job

Choose a transform type

Change schema
Change schema of your source data and create a new target dataset.

Find matching records
Use machine learning to find matching records within your source data.

Back **Next**

8. In Choose a data target:

- For Data store: select amazon S3
- Format: parquet
- Target path: s3://xxx-dmlabs3bucket-xxx/cdc_bookmark/ticket_purchase_history_data/
- Click Next

Add job

Choose a data target

Create tables in your data target
 Use tables in the data catalog and update your data target

Data store: Amazon S3
Format: Parquet
Target path: 1hydg0ba/cdc_bookmark/ticket_purchase_history_data/

Back **Next**

9. In map the source columns to target columns window, leave everything default and Click on Save job and edit script.

Add job

Map the source columns to target columns.

Verify the mappings created by AWS Glue. Change mappings by choosing other columns with Map to target. You can Clear all mappings and Reset to default AWS Glue mappings. AWS Glue generates your script with the defined mappings.

Source	Column name	Data type	Map to target	Target	Column name	Data type
	op	string	op		op	string
	sporting_event_ticket_id	string	sporting_event_ticket_id		sporting_event_ticket_id	string
	purchased_by_id	string	purchased_by_id		purchased_by_id	string
	transaction_date_time	string	transaction_date_time		transaction_date_time	string
	transferred_from_id	string	transferred_from_id		transferred_from_id	string
	purchase_price	double	purchase_price		purchase_price	double

Back **Save job and edit script**

10. In the next window, review the job script and click on Run job. Click on close mark on the top right of the window to close the screen.

Lab 2. ETL with AWS Glue

```

Job: Glue-Lab-TicketHistory-Parquet-with-bookmark
Action Save Run job Generate diagram
Database Name ticketdata
Table Name cdc_ticket_purchase_hist
Transform Name ApplyMapping
Transform Name ResolveChoice
Transform Name DropNullFields
s3://dmslab-student-dmslabs3bucket-xg1hdyq60ibs/partning_hyphenated_bookmark_ticket_purchase_history.parquet
Partitions: 1
Last modified: 2020-01-27T10:10:00Z
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. # Initialize [JOB NAME]
9. args = getResolvedOptions(sys.argv, ['JOB_NAME'])
10.
11. sc = SparkContext()
12. glueContext = GlueContext(sc)
13. spark = glueContext.sparkSession
14. job = Job(glueContext)
15. job.init(args['JOB_NAME'], args)
16.
17. ## Merge [datasource = "ticketdata", table_name = "cdc.ticket.purchase_hist", transformation_ctx = "datasource0"]
18. ## Inputs: []
19. ## Outputs: []
20. ## Resolved: [glueContext.create_dynamic_frame.from_catalog(database = "ticketdata", table_name = "cdc.ticket.purchase_hist", transformation_ctx = "datasource0")]
21. ## Reuse: ApplyMapping
22. ## Merge: [mapping = ["op", "op", "string"], "sporting_event_ticket_id", "string", "purchased_by_id", "string"], ("transaction_data_time", "string"), ("transaction_data_time", "string")
23. ## Inputs: [frame = datasource0]
24. ## Outputs: [mapping.apply(frame = datasource0), mappings = [{"op": "op", "op": "string"}, "sporting_event_ticket_id", "string", "purchased_by_id", "string"], ("transaction_data_time", "string"), ("transaction_data_time", "string")]
25. ## Reuse: ResolveChoice
26. ## Inputs: [frame = datasource0]
27. ## Outputs: [frame = datasource0], transformation_ctx = "resolvechoice2"
28. ## Reuse: resolvechoice2
29. ## Inputs: [frame = datasource0]
30. ## Outputs: [frame = datasource0], transformation_ctx = "dropnullfields3"]
31. ## Reuse: DropNullFields
32. ## Inputs: [frame = datasource0], transformation_ctx = "dropnullfields3"]
33. ## Outputs: dropnullfields3
34. ## Reuse: dropnullfields3
35. dropnullfields3 = DropNullFields.apply(frame = resolvechoice2, transformation_ctx = "dropnullfields3")
36. ## Inputs: [frame = dropnullfields3]
37. ## Outputs: [frame = dropnullfields3], connection_options = {"path": "s3://dmslab-student-dmslabs3bucket-xg1hdyq60ibs/cdc_bookmark/ticket_purchase_history/data/"}, format = "parquet", transformation_ctx = "datasink4"
38. ## Reuse: datasink4
39. ## Inputs: [frame = dropnullfields3]
40. datasink4 = glueContext.write_dynamic_frame.frame_options(frame = dropnullfields3, connection_type = "s3", connection_options = {"path": "s3://dmslab-student-dmslabs3bucket-xg1hdyq60ibs/cdc_bookmark/ticket_purchase_history/data/"}, format = "parquet", transformation_ctx = "datasink4")
41. job.commit()

```

11. Once the job finishes its run, check the **S3 bucket** for the parquet partitioned data.

Name	Last modified	Size	Storage class
part-00000-49bea7fc-2ac1-4787-b431-9e16f5e24a3f-c000.snappy.parquet	Jan 24, 2020 7:03:16 PM GMT-0500	1.1 MB	Standard
part-00001-49bea7fc-2ac1-4787-b431-9e16f5e24a3f-c000.snappy.parquet	Jan 24, 2020 7:03:16 PM GMT-0500	1.2 MB	Standard

Step 3: Create Glue crawler for Parquet data in S3

- Once you have the data in S3 bucket, navigate to **Glue Console** and now we will crawl the parquet data in S3 to create data catalog.
- Click on Add crawler**

- In crawler configuration window, provide crawler name as **glue_lab_cdc_bookmark_crawler** and Click **Next**.

- In Add a data store:**
 - For **Choose a data store**, select **S3**

Lab 2. ETL with AWS Glue

- b. For the path, provide this: s3:// xxx-dmslabs3bucket-xxx and append /cdc_bookmark/ticket_purchase_history/.
6. Click on **Next**

Add a data store

Choose a data store
S3

Crawl data in
 Specified path
Include path
s3://dmslab-student-dmslabs3bucket-xg1hdyg60bs/cdc_bookmark/ticket_purchase_history/
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.
Exclude patterns (optional)

Back Next

7. For **Add another data store**, select **No** and click **Next**.

Add another data store

Yes
 No

Back Next

8. In **Choose an IAM role**, select **Choose an existing IAM role** and select the role that you created as part of the DMS_Student Lab. (for eg, this role name looks something like this: dmslab-student-GlueLabRole-<random-alphanumeric-characters>)

Choose an IAM role

The IAM role allows the crawler to run and access your Amazon S3 data stores. [Learn more](#)

Update a policy in an IAM role
 Choose an existing IAM role
 Create an IAM role

IAM role

dmslab-student-GlueLabRole-14RBWFBWGZ4MB

This role must provide permissions similar to the AWS managed policy, **AWSGlueServiceRole**, plus access to your data stores.
s3://dmslab-student-dmslabs3bucket-xg1hdyg60bs/cdc_bookmark/ticket_purchase_history/
You can also create an IAM role on the [IAM console](#).

Back Next

9. For setting the **frequency** in create a schedule for this crawler, select "Run on demand". Click **Next**

Create a schedule for this crawler

Frequency
Run on demand

Back Next

10. For the crawler's output:

- For Database, select "**ticketdata**" database.
- Optionally, add prefix to the newly created tables for easy identification. Provide the prefix as "**bookmark_parquet_**"
- Click **Next**

Configure the crawler's output

Database
ticketdata

Add database

Prefix added to tables (optional)
bookmark_parquet_

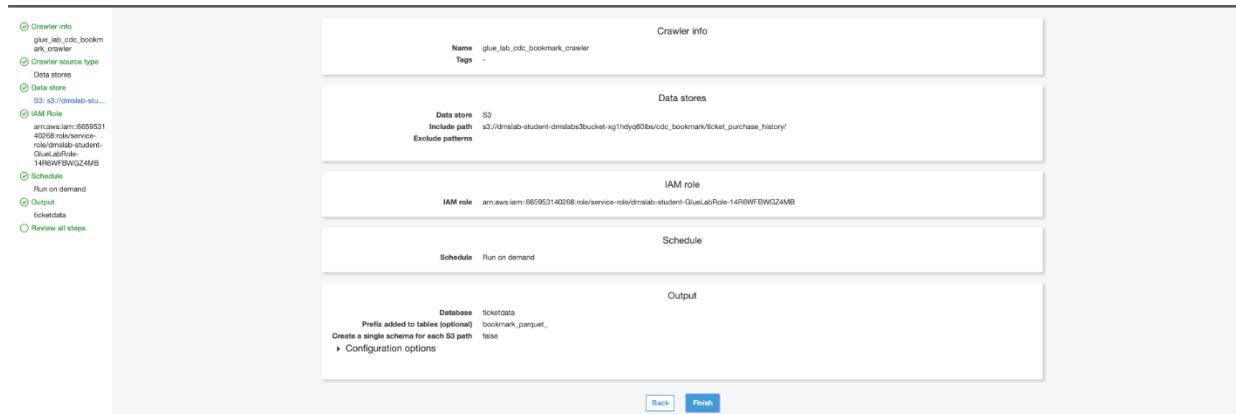
Grouping behavior for S3 data (optional)

Configuration options (optional)

Back Next

11. Review all the details and click on **Finish**. Next, **run the crawler**.

Lab 2. ETL with AWS Glue



12. After the crawler finishes running, click on Databases, select “**ticketdata**” and view tables in this database. You will find the newly created table as **“bookmark_parquet_ticket_purchase_history”**

Name	Database	Location	Classification	Last updated	Deprecated
bookmark_parquet_ticket_purchase_history	ticketdata	s3://dmslab-student-dm.../bucket-xg1hdygq0b...	parquet	24 January 2020 7:14 PM UTC-5	
cdc_sporting_event_ticket	ticketdata	s3://dmslab-student-dm.../bucket-xg1hdygq0b...	csv	24 January 2020 5:13 PM UTC-5	
cdc_ticket_purchase_hist	ticketdata	s3://dmslab-student-dm.../bucket-xg1hdygq0b...	csv	24 January 2020 5:13 PM UTC-5	
mnb_data	ticketdata	s3://dmslab-student-dm.../bucket-xg1hdygq0b...	csv	10 January 2020 1:37 PM UTC-5	

13. Once the table is created, click on Action and from dropdown select View Data.

If it's the first time you are using Athena in your AWS Account, click **Get Started**

Amazon Athena

Amazon Athena is a fast, cost-effective, interactive query service that makes it easy to analyze petabytes of data in S3 with no data warehouses or clusters to manage.

Get Started

Getting started guide

Before you run your first query, you need to set up a query result location in Amazon S3. Learn more

Then click **set up a query result location in Amazon S3** at the top

Sources Workgroup : primary

In the pop-up window in the **Query result location** field, enter your s3 bucket location followed by /, so that it looks like **s3://xxx-dmsslabs3bucket-xxx/** and

click **Save**

Settings

Settings apply by default to all new queries. [Learn more](#)

Workgroup: [primary](#)

Query result location



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

Encrypt query results



Autocomplete



[Cancel](#)

[Save](#)

To select some rows from the table, try running:

```
SELECT * FROM  
"ticketdata"."bookmark_parquet_ticket_purchase_history_data" limit  
10;
```

To get a row count, run:

```
SELECT count(*) as recordcount FROM  
"ticketdata"."bookmark_parquet_ticket_purchase_history_data";
```

Before moving on to next step, note the rowcount.

[Step 4: Generate CDC data and to observe bookmark functionality](#)

Ask your instructor generate more CDC data at source database, if you ran the instructor setup on your own, then make sure to follow "**Generate the CDC Data**" section from instructor prelab.

1. To make sure the new data has been successfully generated, check the S3 bucket for cdc data, you will see new files generated. Note the time when the files were generated.

Lab 2. ETL with AWS Glue

The screenshot shows the Amazon S3 console with the path: Amazon S3 > dmslab-student-dmslabs3bucket-xg1hdyyq60bs > cdc_bookmark > ticket_purchase_history > data. The 'Overview' tab is selected. A search bar at the top says 'Type a prefix and press Enter to search. Press ESC to clear.' Below it are buttons for 'Upload', '+ Create folder', 'Download', and 'Actions'. The main area lists files with columns for 'Name', 'Last modified', 'Size', and 'Storage class'. The 'Name' column lists file names like 'part-00000-05202004-2fc-47c2-0249-d9100b7edad-c000.snappy.parquet' through 'part-00002-01666723-315b-45fb-8630-c3cd2e1519a-c000.snappy.parquet'. The 'Last modified' column shows dates from Jan 24, 2020 to Jan 25, 2020. The 'Size' column shows file sizes ranging from 9.3 kB to 1.9 MB. The 'Storage class' column shows all entries as 'Standard'. An orange box highlights the first four files in the list.

Name	Last modified	Size	Storage class
part-00000-05202004-2fc-47c2-0249-d9100b7edad-c000.snappy.parquet	Jan 24, 2020 9:20:13 PM GMT+0500	9.3 kB	Standard
part-00000-49bea7fc-2ac1-4787-b421-9e105ea142f-c000.snappy.parquet	Jan 24, 2020 7:03:16 PM GMT+0500	1.1 MB	Standard
part-00000-d1666723-315b-45fb-8630-c3cd2e1519a-c000.snappy.parquet	Jan 25, 2020 11:24:29 PM GMT+0500	1.7 MB	Standard
part-00000-efcc00ff-d440-44bc-8230-c3cd2e1519a-c000.snappy.parquet	Jan 25, 2020 10:24:27 PM GMT+0500	7.2 kB	Standard
part-00000-00d20004-2fc-47c2-0249-d9100b7edad-c000.snappy.parquet	Jan 24, 2020 9:20:13 PM GMT+0500	66.5 kB	Standard
part-00000-49bea7fc-2ac1-4787-b421-9e105ea142f-c000.snappy.parquet	Jan 24, 2020 7:03:16 PM GMT+0500	1.2 MB	Standard
part-00001-d1666723-315b-45fb-8630-c3cd2e1519a-c000.snappy.parquet	Jan 25, 2020 11:24:29 PM GMT+0500	1.7 MB	Standard
part-00000-05202004-2fc-47c2-0249-d9100b7edad-c000.snappy.parquet	Jan 24, 2020 9:20:15 PM GMT+0500	1.7 MB	Standard
part-00002-01666723-315b-45fb-8630-c3cd2e1519a-c000.snappy.parquet	Jan 25, 2020 11:24:19 PM GMT+0500	1.9 MB	Standard

2. Rerun the Glue job **Glue-Lab-TicketHistory-Parquet-with-bookmark** you created in Step 2
3. Go to the Athena Console, and rerun the following query to notice the increase in row count:

```
SELECT count(*) as recordcount FROM  
"ticketdata"."bookmark_parquet_ticket_purchase_history_data";
```

To review the latest transactions, run:

```
SELECT * FROM  
"ticketdata"."bookmark_parquet_ticket_purchase_history_data" order  
by transaction_date_time desc limit 100;
```

PART C: Glue Workflows (Optional, self-paced)

****Pre-requisite before creating workflow** - completed Part B**

Overview:

In AWS Glue, you can use workflows to create and visualize complex extract, transform, and load (ETL) activities involving multiple crawlers, jobs, and triggers. Each workflow manages the execution and monitoring of all its components. As a workflow runs each component, it records execution progress and status, providing you with an overview of the larger task and the details of each step. The AWS Glue console provides a visual representation of a workflow as a graph.

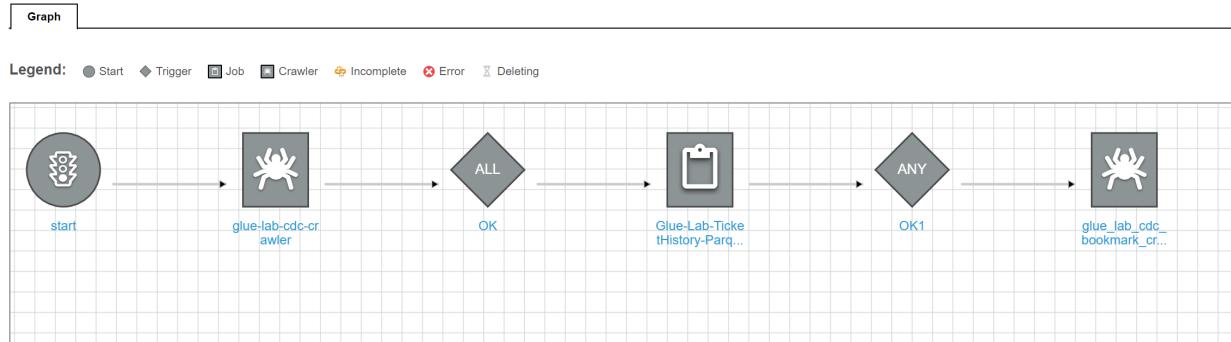
Creating and Running Workflows:

Above mentioned Part A (ETL with Glue) and Part B (Glue Job Bookmarks) can be created and executed using workflows. Complex ETL jobs involving multiple crawlers and jobs can also be created and executed using workflows in an automated fashion. Below is a simple example to demonstrate how to create and run workflows.

Lab 2. ETL with AWS Glue

Try creating a new Glue Workflow to string together the two Crawlers and one Job from part B as follows:

On-demand trigger -> glue-lab-cdc-crawler -> Glue-Lab-TicketHistory-Parquet-with-bookmark -> glue_lab_cdc_bookmark_crawler



To create a workflow:

1. Navigate to **AWS Glue Console** and under **ETL**, click on **Workflows**. Then Click on **Add Workflow**.

The screenshot shows the AWS Glue Workflows console interface. On the left, a sidebar lists categories: Data catalog, Databases, Tables, Connections, Crawlers, Classifiers, Settings, ETL (which is selected), Workflows (highlighted in orange), Jobs, and ML Transforms. The main area is titled "Workflows (0)" and contains the following information:

- A brief description: "A workflow is an orchestration used to visualize and manage the relationship and execution of multiple triggers, jobs and crawlers."
- Buttons: "Add workflow" (highlighted in blue), "Actions", and a search bar "Filter workflows".
- A table header with columns: Name, Last run, Last run status, and Last modified.
- A message: "No workflows" and a button: "Add a new ETL workflow".

2. Give the workflow name as "**Workflow_tickethistory**". Provide a description (optional) and click on **Add Workflow** to create it.
3. Click on the **workflow** and scroll to the bottom of the page. You will see an option **Add Trigger**. Click on that button.

The screenshot shows the "Details" tab for the "Workflow_MLB_Data" workflow. The left sidebar remains the same as the previous screenshot. The main area displays the workflow details:

- A brief description: "A workflow is an orchestration used to visualize and manage the relationship and execution of multiple triggers, jobs and crawlers."
- Buttons: "Add workflow" (disabled), "Actions", and a search bar "Filter workflows".
- A table with columns: Name, Last run, Last run status, and Last modified. It shows one entry: "Workflow_MLB_Data" with a creation date of "Sun, 26 Jan 2020 05:12:03 GMT".
- A message: "The workflow is empty" and a button: "Add trigger".

4. In **Add Trigger** window, From Clone Existing and Add New options, click on **Add New**.
 - a. Provide **Name** as "**trigger1**"
 - b. Provide a **description**: Trigger to start workflow

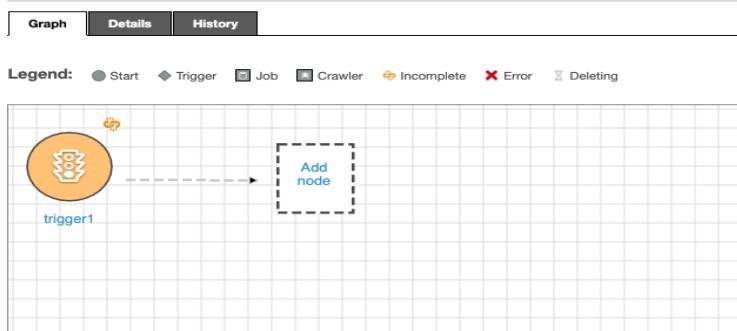
c. **Trigger type: On-demand.**

d. Click on **Add**

Triggers are used to initiate the workflow and there are multiple ways to invoke the trigger. Any scheduled operation or any event can activate the trigger which in turn starts the workflow

The screenshot shows the 'Add trigger' dialog box. At the top, there are two buttons: 'Clone existing' and 'Add new'. The 'Add new' button is highlighted. Below these are fields for 'Name' (containing 'trigger1') and 'Description (optional)' (containing 'Trigger to start the workflow'). Under 'Trigger type', the 'On demand' option is selected. At the bottom right are 'Cancel' and 'Add' buttons.

- Click on **trigger1** to add a **new node**. New Node can be a crawler or job, depending upon the workflow you want to build.



- Click on **Add node**, a new window to add jobs or crawlers will open. Select the Crawler **glue-lab-cdc-crawler**, then **Add**.
- Click on the crawler and **Add Trigger** provide the following:
 - Name:** trigger2
 - Description:** Trigger to execute job
 - Trigger type:** Event
 - Trigger logic:** Start after ALL watched event. This will make sure that job starts once Glue Crawler finishes.
 - Click Add**

Lab 2. ETL with AWS Glue

The screenshot shows the 'Add trigger' dialog box. At the top, there are two buttons: 'Clone existing' and 'Add new'. The 'Add new' button is highlighted. Below it, the 'Name' field contains 'trigger2'. The 'Description (optional)' field has the text 'Trigger to execute crawler'. Under 'Trigger type', the 'Event' radio button is selected. In the 'Trigger logic' section, the 'Start after ANY watched event' radio button is selected. At the bottom right are 'Cancel' and 'Add' buttons, with 'Add' being the active one.

8. After **trigger2** is added to workflow, Click on **Add node**, select job **Glue-Lab-TicketHistory-Parquet-with-bookmark**, click **Add**.
9. Click on the job and **Add Trigger** provide the following:
 - a. **Name: trigger3**
 - b. **Description: Trigger to execute crawler**
 - c. **Trigger type: Event**
 - d. **Trigger logic: Start after ANY watched event.** This will make sure that crawler starts once Glue job finishes processing of ALL data.
 - e. Click **Add**
10. Click on **Add node**, Select the Crawler **glue_lab_cdc_bookmark_crawler**, then **Add**.
11. Select your workflow, click on **Actions->Run** and this will start the first trigger "trigger1"
12. Once the workflow is completed, you will observe that glue job and crawlers have been successfully executed.

Congratulations!! You have successfully completed this lab