AWS Glue vs. Data Pipeline

NOTE: This content will be available as a video course soon! We are making it available as a Learning Path resource in the meantime to help you prepare for the newly updated AWS Certified Solutions Architect - Associate (SAA-C03) exam.

Introduction

Hello and welcome to this course, which will talk about some of the core data processing services on AWS.

Working with data is often an ugly experience. Data needs cleaning, validation, transforming, and enriching. This is where ETL, or extract, transform, and load, processing comes into play. AWS offers a few ETL services that will properly categorize, clean and enrich your data. Two of the main ETL services are Amazon EMR and AWS Glue. In this course, I'll be comparing the two services, and discussing ways to make ETL processes more automated and repeatable.

My name is Alana Layton, and I am an AWS content creator here at Cloud Academy. Feel free to connect with me to ask any questions using the details shown on the screen, alternatively you can always get in touch with us here at Cloud Academy by sending an e-mail to support@cloudacademy.com where one of our Cloud experts will reply to your question.

Who should attend this course?

This course has been created for those who are implementing and managing ETL on AWS, and for those who are looking to take an AWS certification, such as the AWS Certified Solutions Architect – Associate Certification.

Learning Objectives

By the end of this course, you will have a greater understanding of ETL processes on AWS, including:

- What AWS Glue is and how it works
- How AWS Glue compares to Amazon EMR
- How to make ETL processes more automated and repeatable using orchestration services such as AWS Data Pipeline, AWS Glue Workflows, and AWS Step Functions

Prerequisites

In this course, I will provide introductory information on AWS Glue. However, to get the most of this course, you should have an understanding of Amazon EMR and Amazon EC2. For more information on these services please see our existing content, titled:

Compute Fundamentals for AWS
https://cloudacademy.com/course/compute-fundamentals-for-aws/

Introduction to Amazon Elastic Map Reduce
https://cloudacademy.com/course/intro-amazon-elastic-map-reduce-emr-1115/

Feedback

Feedback on our courses here at Cloud Academy is valuable to both us as trainers and any students looking to take the same course in the future. If you have any feedback, positive or negative, it would be greatly appreciated if you could contact support@cloudacademy.com.

Please note that, at the time of writing this content, all course information was accurate. AWS implements hundreds of updates every month as part of its ongoing drive to innovate and enhance its services.

As a result, minor discrepancies may appear in the course content over time. Here at Cloud Academy, we strive to keep our content up to date in order to provide the best training available. So, if you notice any information that is outdated, please contact support@cloudacademy.com. This will allow us to update the course during its next release cycle.

Thank you!

AWS Glue Primer

Hello and welcome to this lecture which is meant to provide you with a high-level overview of the AWS Glue service.

AWS Glue historically was only an ETL service. Since then, the service has turned into a suite of data integration tools. Now, AWS Glue is made up of four different services:

1. Glue Data Catalog

2. Glue Studio

3. Glue DataBrew, and

4. Glue Elastic Views. Glue Elastic Views is out of scope for this content, so I won't be talking about it in this lecture. If you're interested in Glue Elastic Views, I will link a course specifically for that topic.

Let's go through each one of these Glue services.

AWS Glue Data Catalog

First, there's Glue Data Catalog. AWS defines this as a central metadata repository. This means that it stores data about your data. This includes information like data format, data location, and schema. Here's how it works:

You upload your data to storage like Amazon S3, or a database like Amazon DynamoDB, Amazon Redshift, or Amazon RDS. From there, you can use a Glue Crawler to connect to your data source, parse through your data, and then infer the column name and data type for all of your data. The Crawler does this by using Classifiers, which actually reads the data from your storage. You can use built-in Classifiers or custom Classifiers you write to identify your schema.

Once it infers the schema, it will create a new catalog table with information about the schema, the metadata, and where the source data is stored. You can have many tables filled with schema data from multiple sources. These tables are housed in what's called a database.

Note, that your data still lives in the location where you originally uploaded it, but now you also have a representation of the schema and metadata for that data in the catalog tables. This means your code doesn't necessarily need to know where the data is stored, and can reference the Data Catalog for this information instead.

Glue ETL Jobs in Glue Studio

You can then use the Data Catalog to create ETL jobs if needed. When you create a Glue ETL job, you need three things:

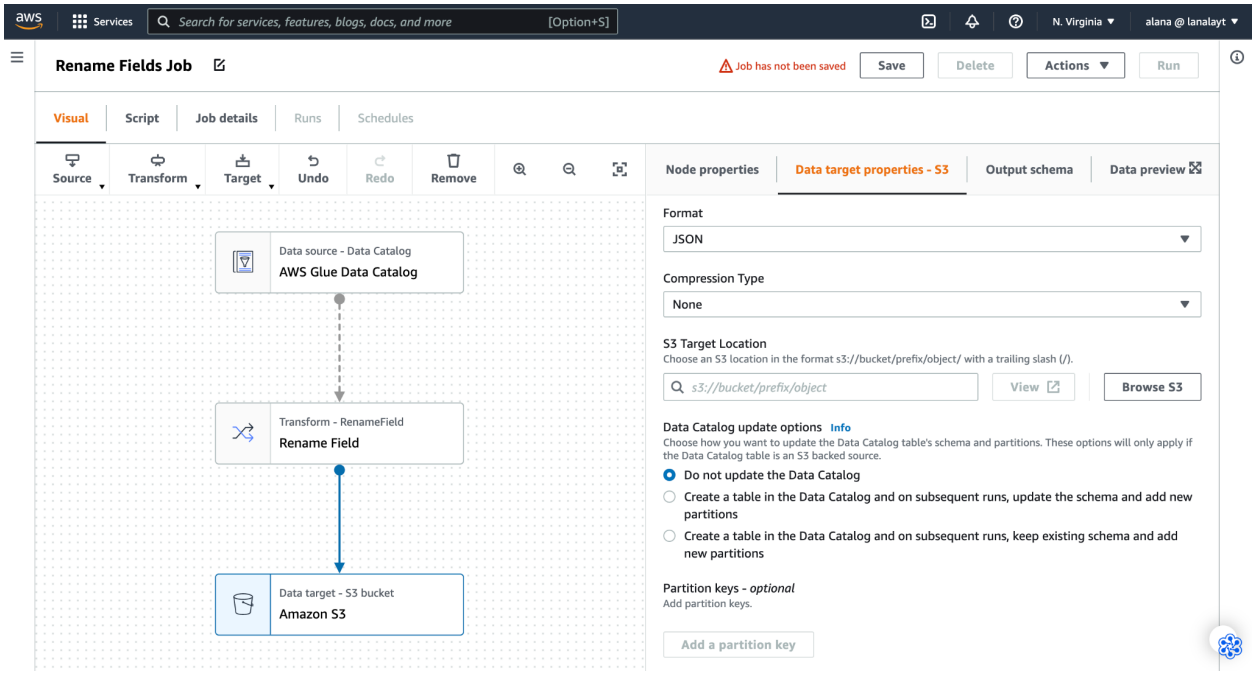
1. **A data source.** This could be the Data Catalog, Amazon S3, Amazon Kinesis, Redshift, RDS, DynamoDB or another JDBC source.

2. You need a **transformation script**. Glue will take data from your source and process it according to a transformation script you write.

3. Then, you need a **target**. Glue will export the output to a target of your choice, such as the Data Catalog, Amazon S3, Redshift or a JDBC source.

In the console, you create, submit, and monitor ETL jobs in Glue Studio. To create jobs, you can choose between programmatic and no-code development.

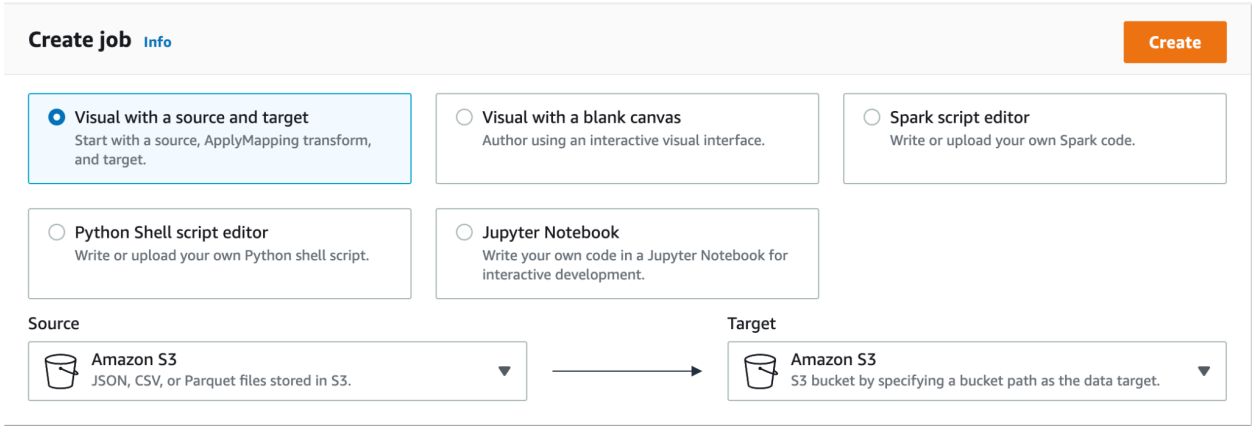
The no-code option enables you to use a visual interface to create graphical relationships between your source, transformation scripts, and your destination.



For example, in this screen, I'm using the Data Catalog as my source. For my transformation script, I'm using a built-in script called Rename Field, that renames a key in my data set to another name. Then, I'm outputting the transformation to an Amazon S3 bucket. I can additionally choose to update my Data Catalog or not.

You can create pretty complex relationships and graphs between services without coding at all, and Glue will generate the Apache Spark code for you behind the scenes. The built-in transformation scripts in Glue Studio are very limited, however you can create custom transformation scripts using Python or Scala as well.

If you're a data engineer or someone familiar with writing ETL scripts, this graphical interface might feel clunky to you. So instead of using this no-code option, you can choose to use the Spark script editor, the Python shell script editor, or the built-in Jupyter Notebook interface to create Python or Scala job scripts. These Glue Studio programmatic options provide real power and flexibility to the user.



Glue DataBrew

A few years ago, Glue released another transformation tool called Glue DataBrew. On the surface, DataBrew looks extremely similar to Glue Studio. So, what is GlueDataBrew?

Glue DataBrew is a true no-code service for transforming data. Here's how it works:

You first **upload your data**. You can upload it directly to the service, or connect to other data sources like Amazon S3, Amazon Aurora, Amazon Redshift, Glue Data Catalog, or other JDBC Connections. It can additionally connect to AppFlow, Data Exchange and Snowflake.

Once you upload your data, you can preview your data in a visual interface. From there you can **choose from hundreds of built-in transformations**. Some of these transformations include formatting your data, modifying columns, working with duplicate or missing values, encoding data, and more.

Once you apply your transformation, you can **store the output in Amazon S3**. Note that Amazon S3 is the only place you can store your transformed data.

Glue DataBrew vs Glue Studio

So if both of these services provide transformations, function in similar ways, and if Glue Data Studio also provides some no-code options, which service do you use?

Well, there are a four main differences between the two that might help you distinguish when to use each service:

1. No-Code vs. Custom Code

Glue DataBrew is a no-code tool. Unlike Glue Studio, you can't write your own custom code for transformations even if you wanted to. However, that means that DataBrew provides a lot more options for built-in transformations. DataBrew has over 250+ built-in transformations, while Glue

Studio has around 10. These transformations are different as well. Glue Studio built-in transformations focus mostly on ETL, while DataBrew's transformations mostly prepare data for machine learning.

2. Different Tools for Different Users

These services are meant for different audiences. Glue Studio is meant for ETL engineers and is focused on ETL itself, while Glue DataBrew is mostly for business analysts and data scientists that may not have coding experience. You don't need specialized expertise to transform data with DataBrew.

3. Programmatic Creation of ETL Jobs

Both services provide a graphical interface for visualizing your transformations. Glue Studio, however, is the only option that provides programmatic opportunities for working with ETL through Jupyter notebooks and shell scripts.

4. Data Profiling

DataBrew has a profiling feature, which enables you to get statistics about your data. For example, with profiling, you can get information about how many rows you have in your data set or how many unique values you have in each column. Glue Studio does not have a data profiling feature.

Summary

In summary, here are quick definitions for each tool:

- Glue Data Catalog is a central metadata repository.
- Glue DataBrew is an easy way to transform data for all levels of technical expertise, while providing data profiling statistics.
- Glue Studio provides the ability to create custom ETL jobs with transformation scripts written in Python or Scala.

That's it for this one - see you next time!

Amazon EMR vs. AWS Glue for ETL

In this lecture, I'll be comparing Amazon EMR vs AWS Glue. Before we get deeper into the two services, it's important to note that Amazon EMR does have multiple deployment options. You can use EMR on EC2, on EKS, or use EMR serverless. In this video, I'll be focusing mostly on EMR on EC2, with some mentions to EMR serverless.

With that being said, our next fight for the evening is: AWS Glue taking on Amazon EMR. Who will win?

In one corner, we have Amazon EMR, a big data platform that's designed not only for ETL, but also for machine learning and data analysis.

In the other corner, we have AWS Glue, a data integration service that provides Glue Studio for ETL. It also includes a Data Catalog, Glue DataBrew for no-code transformations, and Glue Elastic Views.

Let's look at these two services from three different perspectives:

1. Ease of use
2. Pricing
3. Limitations

Ease of Use

Ease of use for any tool in AWS is often inversely related to control. Tools that AWS says are " easy to use" generally provide the user with less control over the service. The same is true the other way around, tools that provide a lot of control are typically more complex to use.

You can see this clearly with EMR and Glue. For example, EMR on EC2 provides maximum control over the service. You can optimize, manage, and scale your cluster and compute nodes. You can take advantage of EC2 instance types, sizes, and pricing options such as Spot, Reservations, and Savings Plans. You can install a wide range of open source tools, such as Hive, Presto, HBase, Spark, and more to fit your use case. And you can choose how long you run your EMR cluster. It could be a longer running cluster that is available 24/7, or it could be a transient cluster that's provisioned, runs the proposed jobs, and then terminates soon after. You have control over all of it.

However, the freedom of choice can make the service more complex to manage. Configuring and maintaining the engine and the cluster can be a full time job. You may need to dedicate resources in your engineering teams to manage this underlying infrastructure.

With Glue, your choices decrease because it is serverless. You no longer get to manage the underlying EC2 instances and storage. All cluster, node, and engine maintenance disappears. From the infrastructure maintenance perspective, it is simpler. However, that also means you no longer get to choose EC2 instance types, sizes, or pricing options. And you also no longer get to choose from a range of open source engines. Glue can only run your ETL jobs in an Apache Spark environment. The other factor is that Glue terminates as soon as your job finishes executing, so support for longer-running clusters is not possible.

Pricing

The convenience of serverless is helpful, but there is a price for this convenience. Glue is, at face value, more expensive than EMR on EC2. This is a common tradeoff in AWS, where you have to decide if the convenience of not having to configure and manage a cluster is worth it. However, before you go with the cheaper option, you have to factor in additional costs with EMR, such as what it takes to maintain a cluster. You might need to factor in the cost of a cluster administrator into your total cost analysis. With all costs considered, you might even find that Glue may be cheaper in the long run.

Another factor to consider is that Glue terminates as soon as the job executes. With Glue, you only pay for the time it runs. If you have longer-running EMR clusters, you will pay for idle time where the cluster is sitting there, not performing any work.

Limitations

With Glue, there are three limitations you need to be aware of:

1. It has a default limitation on how much CPU and RAM you can use for your jobs. The biggest worker type you can use currently has 8 vCPU and 32 GB of RAM. The number of workers you can scale up to in Glue by default is 100. So if you need more performance than Glue can provide to you, EMR is the better choice.

2. Glue is a true ETL service. While it can do light machine learning analysis and can be paired with Amazon Athena for data analysis, Amazon EMR outperforms Glue for both Machine Learning and with data analysis using engines like Presto.

3. Ultimately, if your workload requires any other engine other than Spark, you should use EMR.

EMR Serverless vs. Glue

EMR on EC2 vs Glue is a fairly straightforward comparison. However, the differences between EMR Serverless and Glue are a little less obvious. Both EMR Serverless and Glue require no infrastructure maintenance and are more expensive than EMR on EC2. The biggest difference is the use case. Like EMR on EC2, you can use EMR Serverless for use cases beyond ETL. With Glue, while it does support light machine learning transformations, it is mostly considered an ETL tool. Because of this, Glue Studio offers more ETL tooling that EMR serverless doesn't natively support, such as a graphical ETL interface, built-in scheduling, and the ability to build pipelines from Glue components.

If you already use EMR and have pre-existing Spark or Hive jobs, it may be worthwhile to consider running these jobs on EMR serverless. This lets you use a familiar tool without the maintenance of cluster management.

Summary

Ultimately, if you need flexibility with how you manage the engine or the underlying infrastructure, EMR on EC2 is best for you. Otherwise, if you need to run short-lived jobs that will run in an Apache Spark environment, Glue or EMR Serverless will save you time by managing the infrastructure for you. That's it for this one - see you next time!

Orchestrating ETL Workflows

ETL pipelines can be complex. You may be running multiple ETL jobs at once, at varying intervals of time (maybe hourly, daily, weekly), involving multiple AWS services. It's important you have a service that not only triggers your pipeline to run, but also automates the movement between services, while also handling basic retry logic and error handling.

To do this, you can use orchestration services. There are three main orchestration services that can be used in combination with ETL services, like Amazon EMR and AWS Glue. These services are:

- AWS Data Pipeline
- AWS Step Functions
- And surprisingly, AWS Glue. AWS glue has its own orchestration tool called Glue Workflows

AWS Glue

Let's start with the simplest of the three options: Glue Workflows. Glue Workflows provides a visual editor to create relationships between your Glue components, such as your Triggers, Crawlers and your Glue ETL jobs.

For example, let's say I create a Workflow. This Workflow will first start with a trigger. I can trigger based on a schedule or an event. I want this Workflow to be triggered daily at 12:00.

Once the workflow is triggered, it will kick off a job to do some light pre-processing of the data. After that is successful, I'll have a crawler crawl the optimized data set. Once the crawler finishes running, I can then run ETL on that data.

Glue will run this workflow every day at 12:00 without my intervention, completely automating my pipeline.

Glue Workflows has only one drawback: it is very simplistic and can only be integrated with Glue tools. If you use other AWS services within your pipeline, and not just Glue, consider using a service that has better service integration, such as Data Pipeline or AWS Step Functions.

There is no extra cost to Glue Workflows, however, you will pay for the Crawlers, the ETL jobs, and the Data Catalog requests that Workflows triggers on your behalf.

AWS Data Pipeline

Next, there's Data Pipeline. Its sole purpose is to coordinate data processing from one service to another without human intervention.

The service itself is pretty bare bones, and because of this, it's very simple in nature. A data pipeline is made up of three core components:

1. **Data nodes**: these are storage locations where you house your input data and output data. Data nodes can be S3, Redshift, DynamoDB, RDS or a JDBC connection.

2. **Activities**: this is the work that you want the pipeline to perform on your data. This could be a CopyActivity, that copies data to another location, it could be a SQL activity, that runs a SQL query on a database, or it could be an EMR activity, such as running an EMR cluster, or running a Hive query or Pig script on an EMR cluster.

3. **Preconditions**: these are conditional statements that must be true before an activity can run. For example, you can check whether a data node exists, or run a custom shell script before your activity runs.

Data Pipeline also has retry functionality built-in to the service. You can configure up to 5 retries per activity. The pipeline won't report failure until it goes through the number of retries you set. The higher the number, the longer it will take.

While AWS DataPipeline is simple to get started with, you may find that there are some limitations. For example, DataPipeline has limited data sources. While you might be able to hack around this, you may want to consider using a service called AWS Step Functions for further AWS service integration.

AWS Step Functions

This leads us to the last orchestration service: AWS Step Functions. While AWS Step functions isn't purpose-built for working with data, it does work well with most general workflows. This generic nature provides more flexibility to the user.

With Step Functions, you can integrate with far more services, such as AWS Lambda, API Gateway, Athena, and more. You can call over 200 AWS services from your Step Functions workflows. It additionally can support pipelines that use Amazon EMR and AWS Glue, whereas DataPipeline only supports EMR.

Step Functions coordinates the navigation among services in a serverless workflow and manages retries and errors. It is more robust than DataPipeline in terms of configuration, providing the ability to not only perform tasks, but also embed simple logic for execution in your pipeline. This enables you to make choices between multiple states, pass data between services, use parallel execution, and implement delays in your pipeline.

To get a feel of how it works let's draw a quick example that uses Step Functions to orchestrate Glue ETL jobs. In this example, I upload my data to Amazon S3, which triggers Step Functions to run. Step Functions first signals to Lambda to validate my data in S3, to ensure that it is the right data type and schema.

If the validation is successful, the data is moved to a staging folder. If the validation fails, it moves to an error folder and sends you a notification using Amazon SNS.

For the successfully validated data, an AWS Glue Crawler runs to infer the schema. Step Functions then triggers a Glue ETL job to run, transforming the file into a different format. Once the glue job is complete, it stores the outputted data in the transformed folder in Amazon S3. I then receive an SNS message stating the ETL job has successfully finished.

This is just an example of what you can do with Step Functions. You can build far more complex ETL processes that include a wide range of AWS services and logic.

Summary

In summary, AWS Glue Workflows is great for creating workflows between different Glue Components. However, it's not best if you need orchestration that includes other AWS services.

While AWS Data Pipeline is a simplistic way of getting started with building data pipelines on AWS, it does have rigid limitations on what services integrate with it. AWS Step Functions has far greater integration with AWS and provides more sophisticated logic when building a pipeline. That's it for this one - see you next time!

Summary

In this course, I talked a lot about AWS ETL services and AWS orchestration services. At this point, you should have a better understanding of AWS Glue and how it differs from Amazon EMR. You

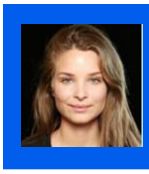
should also be familiar with the different orchestration services you can use with both AWS Glue and Amazon EMR.

If you would like to get some hands-on experience with some of the services I talked about today, then take a look at the following labs titled:

Setting up a Simple Data Lake in AWS
https://cloudacademy.com/lab/setting-simple-data-lake-aws/

Integrating Services with an AWS Step Functions State Machine
https://cloudacademy.com/lab/integrating-services-with-aws-step-functions-state-machine/

Once again, my name is Alana Layton and I hope you've enjoyed our time together. If you have any feedback, positive or negative, please contact us at support@cloudacademy.com. Your feedback is greatly appreciated. Thank you and till next time!



Alana Layton, Sr. AWS Content Creator
alana.layton@cloudacademy.com