



Jan Gazda

Posted on Aug 25, 2020 • Updated on Aug 31, 2020

AWS Glue first experience - part 1 - How to run your code?

#aws #datascience #python #serverless

My AWS Glue journey (5 Part Series)

- 1 **AWS Glue first experience - part 1 - How to run your code?**
- 2 AWS Glue first experience - part 2 - Dependencies and guts
- 3 AWS Glue first experience - part 3 - Arguments & Logging
- 4 AWS Glue first experience - part 4 - Deployment & packaging
- 5 AWS Glue first experience - part 5 - Glue Workflow, monitoring an...

My first project with AWS Glue

I have received an assignment to help build a data lake with a data pipeline using AWS Glue.

It was my first exposure to this service, and there were many challenges along the way worth sharing.

AWS Glue is a serverless, fully managed extract, transform, and load (ETL) service to prepare and load data for analytics. Supports data stored in Amazon Aurora and all other Amazon RDS engines, Amazon Redshift, and Amazon S3, as well as common database engines and databases running on Amazon EC2. Helps you to construct a Data Catalog using Crawlers and pre-built Classifiers then suggests schemas and transformations and generates the code.

The goal of the project

The data lake followed typical 3 zone architecture: Raw, Refined and Curated.

Data for raw and refined zones are stored in S3 bucket while curated data is written to PostgreSQL database running on AWS Aurora.

The idea was to process and transform data incoming from 4 different data sources. All data sources dropped data into a raw zone S3 bucket. Where they have been picked up by individual Glue jobs.

The glue jobs itself have been orchestrated using Glue Workflow feature.

First steps

Since we have identified our data sources and ETL zones it was time to write 8 Glue jobs. Understand four for jobs per transition: 4x `raw to refined`, 4x `refined to curated`.

Glue supports multiple run times: Apache Spark with Scala, Apache spark with Python - PySpark, Python shell - pure python (3.6 by the time of writing) interpreter.

We will stick with Python and use PySpark and python shell.

AWS Glue provides an extension (soft wrapper) around `pyspark.sql.context` and adds Glue specific features such as `DynamicFrame`, etc. To provide maximum portability I'm going to avoid using AWS Glue specific features.

Challenge number 1: How to run your code?

As a python developer, I'm used to splitting the code into modules and packages to prevent large files.

The Glue documentation is pretty straight forward in pointing you to use the pre-generated code or write it using an online editor built-in AWS Glue Console.

Apart from that, there is a short page about [providing your scripts](#).

Most of the documentation shows the examples within the console and our code was expected to be deployed via [Terraform](#).

The Glue Job code requires a script file to be stored in an S3 bucket.

Then you have to point your Terraform resource: `aws_glue_job` to the `script_location` which contains an S3 URL to your file eg. `s3://code-bucket/glue-job.py`


We can now write the code in 1 file, which is enough for small ETL scripts based purely on Spark. In the next episode, we are going to explore how to deal with dependencies.

The code for the examples in this article can be found in my GitHub repository [aws-glue-monorepo-style](#)

My AWS Glue journey (5 Part Series)


- 1 **AWS Glue first experience - part 1 - How to run your code?**
- 2 AWS Glue first experience - part 2 - Dependencies and guts
- 3 AWS Glue first experience - part 3 - Arguments & Logging
- 4 AWS Glue first experience - part 4 - Deployment & packaging
- 5 AWS Glue first experience - part 5 - Glue Workflow, monitoring an...

Top comments (4) ↕

- 


Luminita Färnström • Oct 16 '20

⋮

✕ Really interesting post. We are about to start our journey with Glue, so your post got us a few heads up. I wonder: would you recommend using Workflows or, if one has (for now) quite simple workflows, stick to jobs chained by triggers?
- 

Jan Gazda 🌟 • Nov 26 '20

⋮

✕ Hi Luminita, that really depends. In fact, Workflow is just a chain of triggers and jobs. The benefit of the Workflow is access to the state via Workflow properties.
- 

Adrian Abreu Gonzalez • Nov 30 '20

⋮

✕ Beautiful post series, I've reading a lot of documentation lately and your articles are by far the most serious in this.
I'd like to know a few things also:

- Do you make use of the spark crawlers moving on your raw zone? My idea was to make use of dynamic frames against s3 and resolve schemas conflicts prior converting the data to dataframes.
- Would you include support for the latest aws glue docker image? I'm currently using it and allowed me to launch the aws glue tests.

Thans for everything



Jan Gazda  • Dec 21 '20



Hi Adrian,

We did not make use of crawlers for moving data from raw zone. Let me know how your idea went.

I'm aware of the latest glue container, I did not have the opportunity to try it out yet but I can include it as soon as I do because it's quite important part of the dev flow.

[Code of Conduct](#) • [Report abuse](#)


Update Your DEV Experience Level:

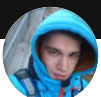
Content

What is your approximate experience level (1-5)?

1 Novice	2 Beginner	3 Mid-level	4 Advanced	5 Expert
-------------	---------------	----------------	---------------	-------------

This will not be displayed on your profile or anywhere publicly. It helps gently determine what content you are shown along with tags you follow etc.

Go to [your customization settings](#) to nudge your home feed to show content more relevant to your developer experience level. 



Jan Gazda

I solve problems, usually with Python. Organize PyAmsterdam python meetups.

LOCATION

Amsterdam

WORK

Freelance engineer

JOINED

Jul 10, 2020

More from [Jan Gazda](#)

AWS Glue first experience - part 5 - Glue Workflow, monitoring and rants

[#aws](#) [#datascience](#) [#python](#) [#serverless](#)

AWS Glue first experience - part 4 - Deployment & packaging

[#aws](#) [#python](#) [#datascience](#) [#serverless](#)

AWS Glue first experience - part 3 - Arguments & Logging

[#aws](#) [#python](#) [#datascience](#) [#serverless](#)
