**AWS GLUE**

**1. Introduction to AWS Glue**

AWS Glue is a serverless data integration service that allows you to discover, prepare, move, and integrate data from multiple sources for analytics, machine learning, and application development.

**2. Key Steps Performed in AWS Glue**

**🔹 Step 1: Set Up IAM Role**

* Create an IAM role with policies for:
  + S3 access(AWSS3FULLACCESS)
  + CloudWatch Logs
  + AWS Glue permissions (e.g., AWSGlueServiceRole)
* Trust relationship should include the Glue service:  
  "Service": "glue.amazonaws.com",

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**🔹 Step 2: Create or Use an Existing S3 Bucket**

* Upload your data sources (CSV, JSON, Parquet, etc.).
* This will be used for both input and output data in ETL jobs.

A screenshot of a computer

AI-generated content may be incorrect.

**🔹 Step 3: Set Up Glue Data Catalog**

* **Crawlers**: Automatically scan your data and create tables.
  + Configure:
    - Data store location (S3, JDBC, etc.)
    - IAM role
    - Schedule (optional)
    - Output database

A screenshot of a computer

AI-generated content may be incorrect.

**🔹 Step 4: Create a Glue Job**

* Choose the source and target tables.
* Select the job type: Spark (for ETL) or Python shell (for lightweight scripting).
* Configure:
  + Script file name
  + Glue version
  + Worker type and number of workers (e.g., G.1X or G.2X)
  + Job bookmark (to handle incremental loads)

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

**🔹 Step 5: Edit and Customize the ETL Script**

* AWS Glue auto-generates PySpark code.
* You can customize:
  + Transformations (e.g., DropFields, RenameField, ResolveChoice)
  + Business logic using DynamicFrame and DataFrame APIs

**🔹 Step 6: Test and Run the Job**

* Validate script syntax.
* Monitor the job via:
  + AWS Console
  + CloudWatch logs
  + Glue job metrics

**🔹 Step 7: Monitor and Debug**

* Use Glue Console → Jobs → Logs to inspect execution logs.
* Use metrics dashboard to monitor:
  + Success/failure rate
  + Execution time
  + Data read/write volumes

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**🔹 Step 8: Schedule Jobs with Triggers**

* Create triggers to:
  + Run jobs on a schedule
  + Chain jobs based on completion
  + React to AWS events (event-based triggers)

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**RUN AWS GLUE ON LOCAL MACHIN USING DOCKER**

[**https://docs.aws.amazon.com/glue/latest/dg/aws-glue-programming-etl-libraries.html**](https://docs.aws.amazon.com/glue/latest/dg/aws-glue-programming-etl-libraries.html)

**To test scripts locally:**

**🔸 Step 1: Install Docker**

**🔸 Step 2: Pull the Glue ETL Docker Image**

**bash**

**CopyEdit**

**docker pull amazon/aws-glue-libs:glue\_libs\_4.0.0\_image\_01**

**🔸 Step 3: Run Docker Container**

**bash**

**CopyEdit**

**docker run -it amazon/aws-glue-libs:glue\_libs\_4.0.0\_image\_01 /bin/bash**

**🔸 Step 4: Set Up Local Python Environment**

* **Glue libraries are pre-installed.**
* **Use Jupyter or VSCode to run .py scripts with DynamicFrame and DataFrame.**

**🔸 Step 5: Test Your Glue Scripts**

* **Run your ETL scripts locally before deploying to AWS Glue.**
* **Connect to local or mock S3 and JDBC endpoints if required.**

**For more:** [**AWS Glue Local Development**](https://docs.aws.amazon.com/glue/latest/dg/aws-glue-programming-etl-libraries.html)

**Serverless steps**

**Install node.js**

**Install aws cli**

**Type below code in command prompt for severless connection**

**Aws configure**

**AWS\_ACCESS\_KEY\_ID=**AKIARQ36HR3PFA4GF2ND

**AWS\_SECRET\_ACCESS\_KEY=**qR3MklLB6oWi3rWO+NGZGZWeS0/VPKWMzt3t72q6

**serverless deploy**

**.env file**

DB\_NAME=bhaygesh-database

GLUE\_ROLE\_ARN=arn:aws:iam::104954564318:role/bhagyesh-role

CRAWLER\_TARGET\_PATH=s3://bhagyesh-s3-bucket/input/product/

CRAWLER\_NAME=bhagyesh-crawler-v3

ACCOUNT=104954564318

AWS\_ACCESS\_KEY= AKIARQ36HR3PFA4GF2ND

AWS\_SECRET\_KEY= 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

**Serverless.yml**

service: sls-create-glue-crawler

useDotenv: true

provider:

  name: aws

  region: ap-south-1

  stackTags:

    product: datateam

    env: qa

    created-date: 2025-05-28

    team: data

    customer-impact: false

    terraform: false

resources:

  Resources:

    GlueDatabase:

      Type: AWS::Glue::Database

      Properties:

        CatalogId: ${env:ACCOUNT}

        DatabaseInput:

          Name: ${env:DB\_NAME}

    TableGlueCrawler:

      Type: AWS::Glue::Crawler

      Properties:

        DatabaseName: ${env:DB\_NAME}

        Name: ${env:CRAWLER\_NAME}

        RecrawlPolicy:

          RecrawlBehavior: CRAWL\_EVERYTHING

        Role: ${env:GLUE\_ROLE\_ARN}

        SchemaChangePolicy:

          DeleteBehavior: DEPRECATE\_IN\_DATABASE

        Targets:

          S3Targets:

            - Path: ${env:CRAWLER\_TARGET\_PATH}