GLUE

GLUE

step1-> step2->step3->step4

AWS Glue is a serverless Spark-based ETL service.

- **Serverless (with constraints)**
    - No cluster management
    - Limited Spark configs & runtime control
    - You trade **control** for **operational simplicity**
- **Custom ETL Jobs (Spark under the hood)**
    - Written in PySpark / Scala
    - Triggered by schedule, event, or Airflow
    - Best for **standardized, repeatable ETL**
- **Glue Data Catalog (Metadata Layer)**
    - Central schema registry for S3 data
    - Shared by Athena, Redshift Spectrum, EMR
    - Glue ≠ Athena, Glue = metadata + ETL

GLUE ETL

- **ETL Development**
  - Auto code generation via Glue Studio (optional)
  - Custom ETL jobs written in **PySpark or Scala**
- **Compute Engine**
  - Spark-based distributed processing
  - Supports Python and Scala workloads
- **Security & Encryption**
  - Server-side encryption **at rest**
  - SSL encryption **in transit**
- **Job Triggers**
  - Event-driven execution (e.g. S3 events)
  - Scheduled runs or orchestrated by **Airflow**
- **Performance Scaling**
  - Scale compute by provisioning additional **DPUs**
  - Vertical scaling model

GLUE ETL

- **Data Processing**
  - Data transformation, data cleansing, data enrichment
  - Auto-generated ETL code (editable)
  - Custom Spark scripts (PySpark / Scala)
- **Targets & Integration**
  - Output targets: **S3**, **JDBC sources**, **Glue Data Catalog (GDC)**
- **Execution Model**
  - Fully managed ETL service
  - Jobs run on a **serverless Spark platform**
- **Cost Model**
  - Cost-effective, **pay only for resources consumed**
  - No idle cluster cost

Glue ETL – Schema & Catalog Management

- **Schema & Partition Management**
  - ETL scripts can update table schema and partitions
  - Automatically add new partitions
- **Partition Updates**
  - Enable `UpdateCatalog` and define `partitionKeys` in the script
- **Schema Evolution**
  - Update table schema using `enableUpdateCatalog` and `updateBehavior`
- **Table Creation**
  - Create new tables via ETL scripts
  - Use `enableUpdateCatalog` / `updateBehavior` with `setCatalogInfo`
- **Limitations**
  - S3 data sources only
  - Supported formats: **JSON, CSV, Avro, Parquet**
  - Parquet requires additional handling
  - Nested schema support is limited

Glue and S3 Partitions

Partition Discovery

- Glue Crawler extracts partitions based on **S3 directory structure**
- data/parquet/ingest_date=20230728/adefeafa001.parquet
- data/parquet/ingest_date=20230728/adefeafa002.parquet
- data/parquet/ingest_date=20230728/adefeafa003.parquet

Crawler Behavior

- Connects to the specified S3 location
- Recursively explores directory structure
- Infers partitions from folder names
- Reads file metadata to infer schema

Key Assumption

- **Folder names represent partition keys**
- Partition format: `key=value`

Athena

What is Athena?

Interactive query service for Amazon S3 (SQL-based)

- Query data directly from S3
- **No data loading required**

Serverless

- No infrastructure or cluster management
- Pay per query based on data scanned

Supported Data Formats

- CSV
- JSON
- ORC
- Parquet
- AVRO

Supported Compression Formats

- Snappy
- Zlib
- LZO
- GZip

Athena Workgroup

- **Logical grouping for Athena usage**
  - Organize users, teams, applications, and workloads into workgroups
- **Access & Cost Control**
  - Control query access at the workgroup level
  - Track and manage query **costs** by workgroup
- **AWS Integration**
  - Integrates with **IAM**, **CloudWatch**, and **SNS**
- **Each Workgroup Can Have**
  - Query history
  - Data scan limits
  - IAM policies
  - Encryption settings
  - Query result location (S3)

Athena Cost

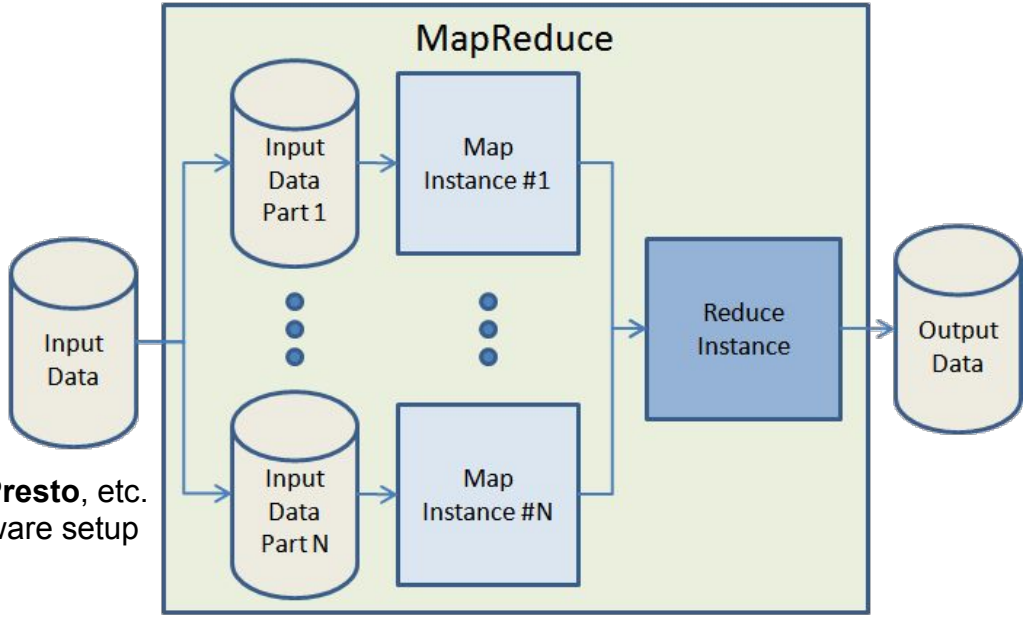
- **Pay-as-you-go**
  - Charged based on **data scanned**
  - **No charge for DDL statements**
    (CREATE / ALTER / DROP)
- **Cost Optimization**
  - Use **columnar formats** for significant savings
  - Recommended formats:
    - **Parquet**
    - **ORC**
- **Cost & Performance Benefits**
  - Reduce scanned data by **30%–90%**
  - Better query performance with less data scanned

EMR

Ec2, emr(ec2 s), glue(emr + serveless)

What is EMR?

- **Amazon Elastic MapReduce (EMR)**
  - Managed big data platform on **EC2**
- **Managed Hadoop Ecosystem**
  - Supports **Hadoop**, **Spark**, **Hive**, **HBase**, **Presto**, etc.
  - You manage clusters, AWS manages software setup
- **Runs on EC2 Instances**
  - Full control over instance types, cluster size, and lifecycle
  - Long-running or transient clusters
- **AWS Integrations**
  - Native integration with **S3**, **IAM**, **CloudWatch**
  - Works with Glue Data Catalog



EMR Cluster Architecture

Master Node

- Manages the cluster and coordinates jobs
- Tracks task status and monitors cluster health
- Single EC2 instance
- Leader node (control plane)

Core Nodes

- Store data in **HDFS**
- Execute processing tasks
- Required for multi-node clusters (at least one)
- Can scale up or down

Task Nodes

- Execute processing tasks only
- **Do not store HDFS data**
- Optional
- Safe to add or remove without data loss

EMR Serverless

- **Serverless execution for EMR workloads**
  - No cluster creation or management
  - Focus on job execution only
- **Runtime Selection**
  - Choose EMR release and runtime:(Spark, Hive, Presto)
- **Job Submission**
  - Submit queries or scripts via **job run requests**
  - Each job runs independently
- **Capacity Management**
  - EMR automatically manages underlying compute capacity
  - You can configure:
    - Default worker size
    - Pre-initialized capacity
- **Auto Scaling**
  - EMR calculates required resources per job
  - Workers are provisioned and released automatically
  - No need to estimate cluster size
- **Availability**
  - Runs within a single region
  - Capacity spans multiple Availability Zones (AZs)

Airflow

step1->step2->step3

What is Airflow?

- **Workflow orchestration tool**
  - Defines, schedules, and monitors data pipelines
- **DAG-based**
  - Pipelines are defined as **Directed Acyclic Graphs (DAGs)**
  - Task dependencies are explicit
- **Not a processing engine**
  - Does **not** process data itself
  - Orchestrates jobs running on:
    - Glue
    - EMR / EMR Serverless
    - Athena
    - Spark / SQL / Python jobs
- **Scheduling & Monitoring**
  - Time-based or event-based scheduling
  - Task retries, failure handling, alerting
- **AWS Integration**
  - Commonly used with **MWAA (Managed Airflow)**
  - Integrates with IAM, CloudWatch, SNS