

AWS MACHINE LEARNING CERTIFICATION



DOMAIN #1: DATA ENGINEERING (20% EXAM)



AWS ML CERTIFICATION EXAM DOMAINS



Domain	% of Examination
Domain 1: Data Engineering	20%
Domain 2: Exploratory Data Analysis	24%
Domain 3: Modeling	36%
Domain 4: Machine Learning Implementation and Operations	20%
TOTAL	100%

Source: [https://d1.awsstatic.com/training-and-certification/docs-ml/AWS%20Certified%20Machine%20Learning%20-%20Specialty_Exam%20Guide%20\(1\).pdf](https://d1.awsstatic.com/training-and-certification/docs-ml/AWS%20Certified%20Machine%20Learning%20-%20Specialty_Exam%20Guide%20(1).pdf)



DOMAIN #1: WHERE ARE WE NOW!!?



SECTION #1: INTRODUCTION, DATA/ML LINGO, AWS DATA STORAGE

- What is Machine Learning and Artificial Intelligence?
- What is Amazon Web Services (AWS)?
- Artificial Intelligence and Machine learning Lingo (data types, Labeled vs. unlabeled, sagemaker groundtruth)
- structured vs. unstructured and database vs. data lake vs. data storage
- AWS Data Storage (Redshift, RDS, S3, DynamoDB)

SECTION #2: AMAZON S3

- Amazon S3 in Depth (partitions, tags)
- Amazon S3 Storage Tiers and Lifecycles
- Amazon S3 Encryption and Security
- Amazon S3 Encryption and Security – Part #2 (ACL, CloudWatch, CloudTrail, VPC)
- Additional Notes (Elasticsearch, ElastiCache, and Database vs. data warehouse)



DOMAIN #1: WHERE ARE WE NOW!!?



SECTION #3: AWS DATA MIGRATION, GLUE, PIPELINE, STEP AND BATCH

- AWS Glue (crawlers, features, built-in transformations etc)
- AWS Data pipeline
- AWS Data Migration Service (DMS)
- AWS Batch
- Step Function

WE ARE HERE!

SECTION #4: DATA STREAMING & KINESIS

- Kinesis Overview
- Kinesis Video Streams
- Kinesis Data Streams
- Kinesis Firehose
- Kinesis Analytics and Random Cut Forest



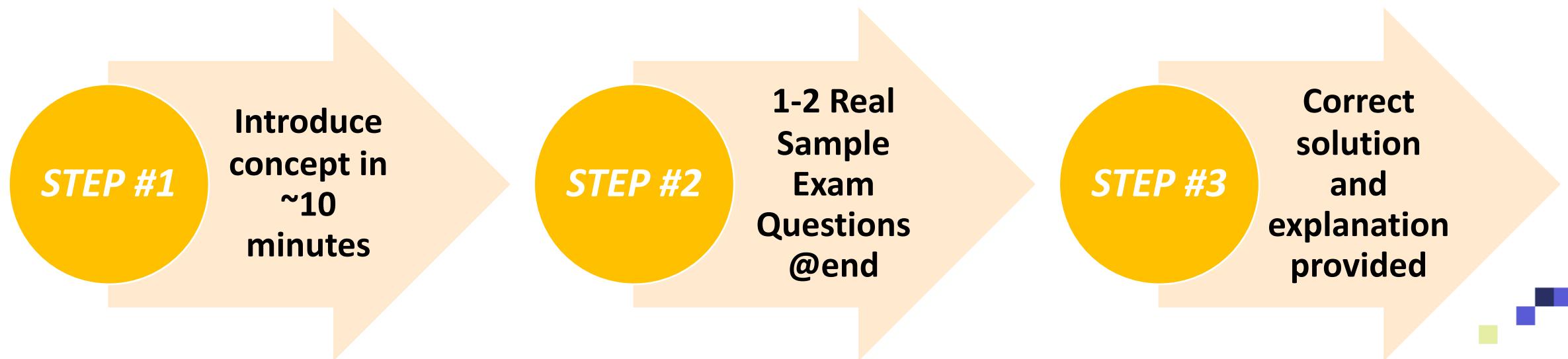
LECTURE DESIGN



- We know how hard it is to study for an exam especially if you have a busy schedule.
- This course is designed to be extremely on point and optimized to pass the exam.

No boring content. Zero unnecessary information.

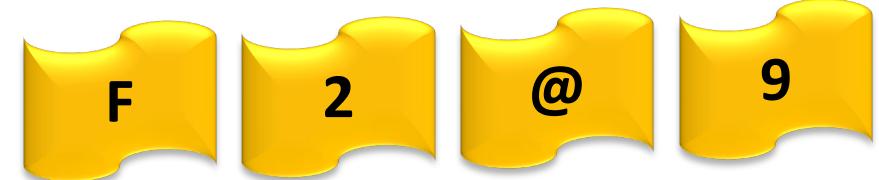
- Here's the lecture structure that we will follow:



RECALL OUR MINI CHALLENGE AND PRIZE!



- For those of you who will successfully complete the entire section and watch the videos till the end, they will receive a valuable prize!



AWS GLUE – Part #1



AWS GLUE: OVERVIEW

- AWS Glue is a fully managed service that enable users to extract, transform, and load (ETL) data.
- Data generated by AWS Glue could be fed into analytics tools such as Amazon QuickSight.
- AWS Glue steps:
 1. Setup AWS Glue and direct it to your stored data on AWS
 2. AWS Glue extracts metadata such as table definition and schema and put it in the AWS Glue Data Catalog.
 3. Once cataloged, you can query the data and perform ETL.



AWS GLUE: UNIQUE FEATURES



SEAMLESS INTEGRATION WITH AWS SERVICES

- AWS Glue works seamlessly with other AWS services which allow for easy integration.
- AWS Glue works well with stored data in Redshift, S3, and databases in Virtual Private Cloud (VPC) running on EC2.

MINIMUM CONFIGURATION HASSLE AND OPTIMIZED COST

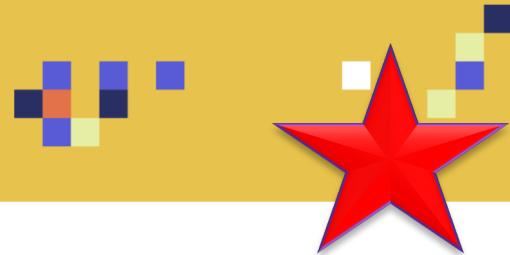
- AWS Glue offers pay per use service so no upfront cost or servers to manage.
- AWS Glue will take care of scaling and configuration in running the ETL jobs.

EASY TO USE

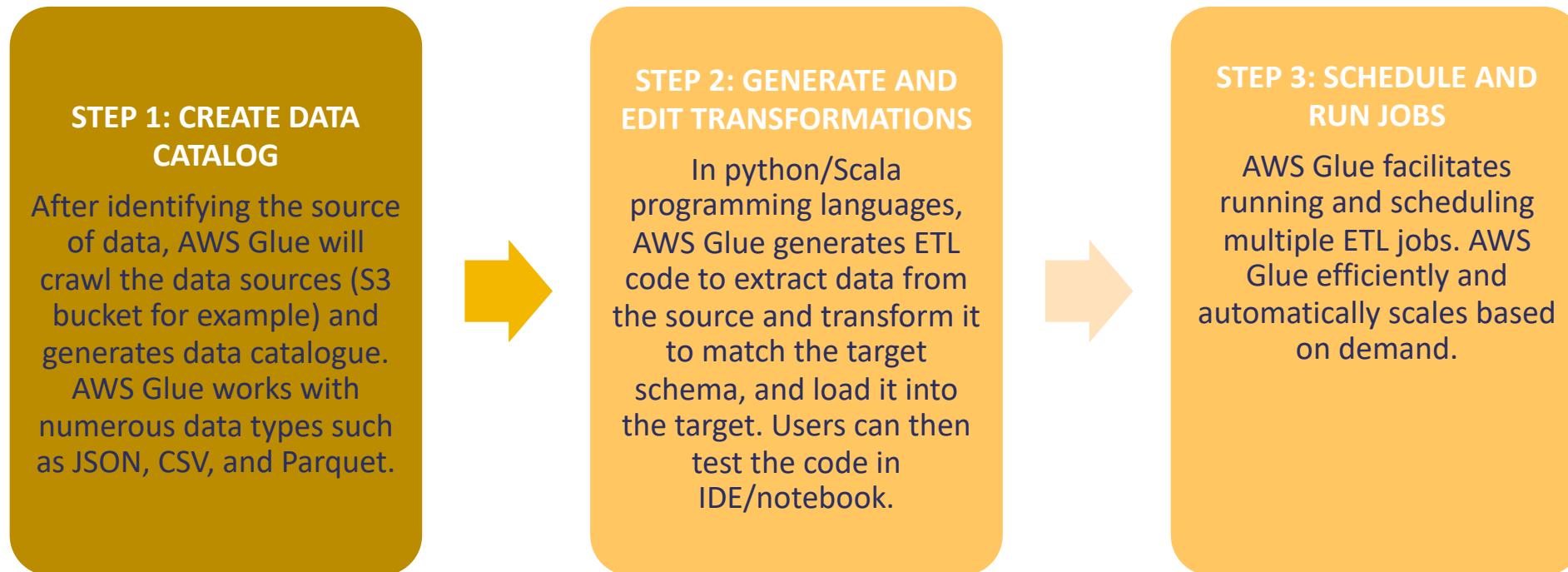
- AWS Glue crawls data storages, extract data schemas and transformations.
- AWS Glue generates data transformation codes on its own.



AWS GLUE: HOW DOES IT WORK?



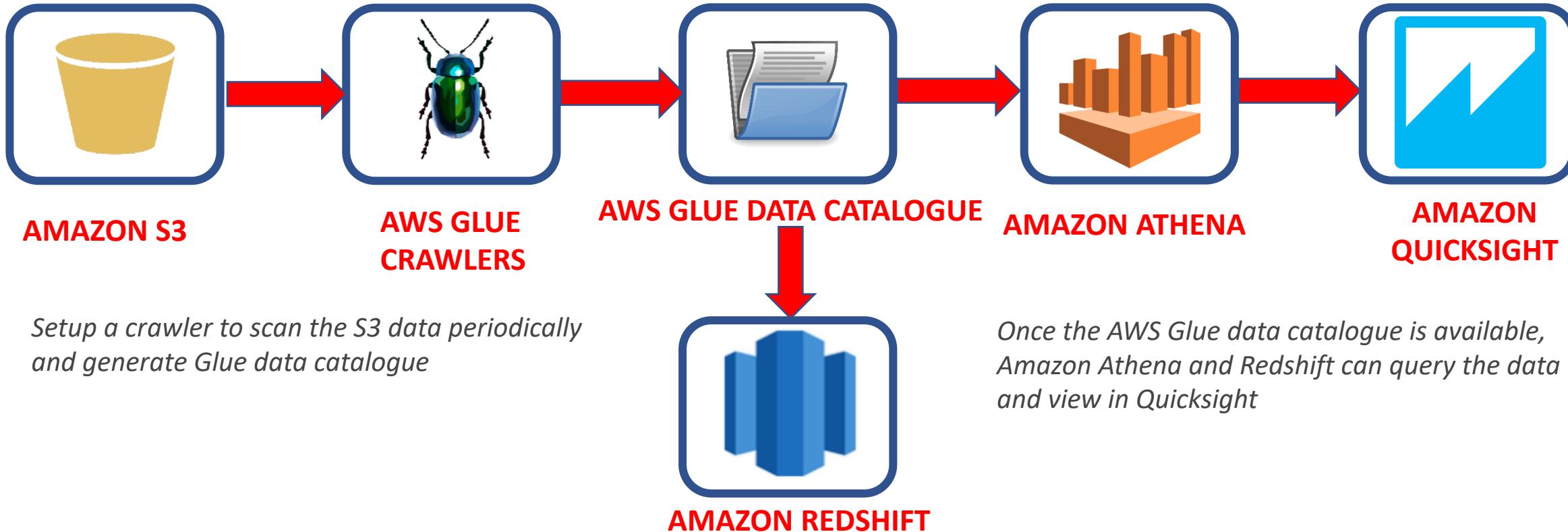
- AWS Glue is extremely easy to work with, users will have to first identify the data source/target.
- In python/Scala programming languages, AWS Glue generates ETL code to extract data from the source and transform it to match the target schema, and load it into the target.
- Users can then test the code in IDE/notebook.



AWS GLUE – Part #2



AWS GLUE: USE CASE



Setup a crawler to scan the S3 data periodically and generate Glue data catalogue

Once the AWS Glue data catalogue is available, Amazon Athena and Redshift can query the data and view in Quicksight

- Athena and AWS Glue Data Catalog work seamlessly together, AWS Glue can be used to create databases and tables (schema) so that Athena can use it to query the data.

Photo Credit: https://commons.wikimedia.org/wiki/File:AWS_Simple_Icons_Storage_Amazon_S3.svg

Photo Credit: <https://pixabay.com/illustrations/insect-insects-insect-perfection-4470664/>

Photo Credit: <https://commons.wikimedia.org/wiki/File:Document-open.svg>

Photo Credit: https://commons.wikimedia.org/wiki/File:Magnifying_glass_01.svg

Photo Credit: <http://pgfplots.net/tikz/examples/plot-markers/>

Photo Credit: <https://publicdomainvectors.org/en/free-clipart/Image-of-electricity-spark-orange-icon/31132.html>

AWS GLUE: WHAT ARE CRAWLERS?



- Crawlers are used to scan the stored data and infer schemas and partitions.
- Once the crawler scans the data, it generates/updates the table in the Data Catalog.
- AWS Glue will rely on this data catalogue in performing Extract, transform, and load (ETL) jobs.
- You will require setting up an IAM role so that the crawler is able to access the data storage and infer schemas.
- AWS Glue works with several data formats such as JSON, Parquet, CSV.
- You can run the Crawler on demand or based on a schedule



**AWS GLUE
CRAWLERS**



AWS GLUE: CRAWLERS AND CLASSIFIERS



The screenshot shows the AWS Glue service console. The left sidebar contains the following navigation items:

- aws Services
- AWS Glue
 - Data catalog
 - Databases
 - Tables** (highlighted with a red box)
 - Connections
- Crawlers
- Classifiers
- Settings

Below the sidebar, under ETL, the following items are listed:

- Workflows
- Jobs
- ML Transforms
- Triggers
- Dev endpoints
- Notebooks

Under Security, the following items are listed:

- Security configurations

Under Tutorials, the following items are listed:

- Add crawler
- Explore table
- Add job
- Resources (with a dropdown arrow)
- What's new (10+)

- A classifier reads the storage data and generates a schema if it finds one.
- AWS Glue provides built-in classifiers for several formats as shown below.

Built-In Classifiers in AWS Glue

AWS Glue provides built-in classifiers for various formats, including JSON, CSV, web logs, and many database systems.

If AWS Glue doesn't find a custom classifier that fits the input data format with 100 percent certainty, it invokes the built-in classifiers in the order shown in the following table. The built-in classifiers return a result to indicate whether the format matches (`certainty=1.0`) or does not match (`certainty=0.0`). The first classifier that has `certainty=1.0` provides the classification string and schema for a metadata table in your Data Catalog.

Classifier type	Classification string	Notes
Apache Avro	avro	Reads the schema at the beginning of the file to determine format.
Apache ORC	orc	Reads the file metadata to determine format.
Apache Parquet	parquet	Reads the schema at the end of the file to determine format.
JSON	json	Reads the beginning of the file to determine format.
Binary JSON	bson	Reads the beginning of the file to determine format.
XML	xml	Reads the beginning of the file to determine format. AWS Glue determines the table schema based on XML tags in the document. For information about creating a custom XML classifier to specify rows in the document, see Writing XML Custom Classifiers .
Amazon Ion	ion	Reads the beginning of the file to determine format.
Combined Apache log	combined_apache	Determines log formats through a grok pattern.
Apache log	apache	Determines log formats through a grok pattern.
Linux kernel log	linux_kernel	Determines log formats through a grok pattern.
Microsoft log	microsoft_log	Determines log formats through a grok pattern.
Ruby log	ruby_logger	Reads the beginning of the file to determine format.
Squid 3.x log	squid	Reads the beginning of the file to determine format.

MANAGING PARTITIONS IN AWS GLUE

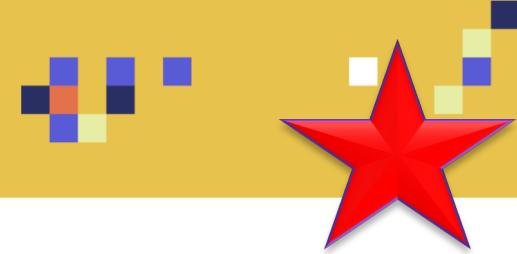


- Partitioning is critical for data organization.
- Proper portioning allow for an efficient query process.
- Once partitioned, AWS services such as Athena, Redshift, and Glue can utilize these partitions to filter through the data instead of scanning the entire bucket on S3.
- Partitioning allows for organizing data in a hierarchy directory structure as follows:

`s3://my_bucket/logs/year=2018/month=01/day=23/`

- AWS Glue Crawlers can rely on the partition structure to generate AWS Glue Data Catalog.
- The resulting partition columns can be queried using (1) AWS Glue ETL jobs and (2) Amazon Athena.

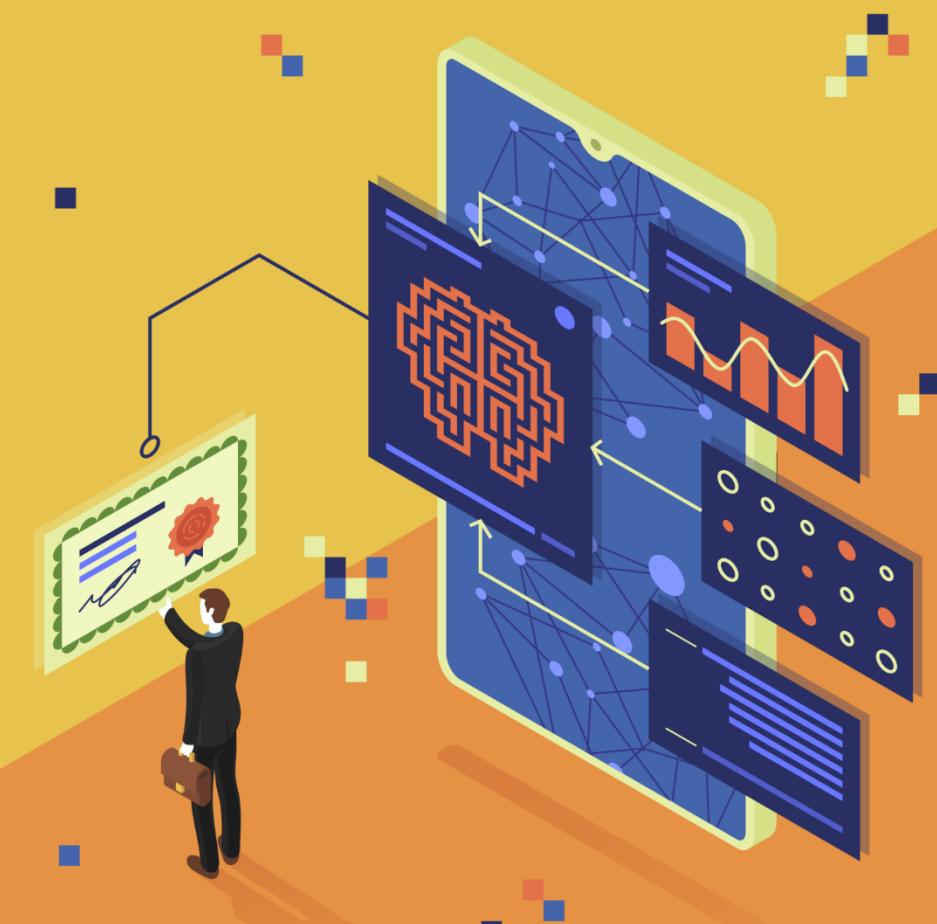
AWS GLUE BUILT-IN TRANSFORMATIONS



- AWS Glue offer several built-in transformations that could be called from an ETL script.
- Generally there are two types of transformations:
- **Bundled Transformations:**
 - DropFields Class
 - DropNullFields Class
 - Filter Class
 - Join Class
 - Map Class
 - MapToCollection Class
 - RenameField Class
 - ResolveChoice Class
- **Machine Learning Transformations:**
 - FindMatches ML: could be used to find matching data which enables you to find related products, customers, and places.
 - FindMatches ML could be used to find duplicate customers who have signed up more than once.

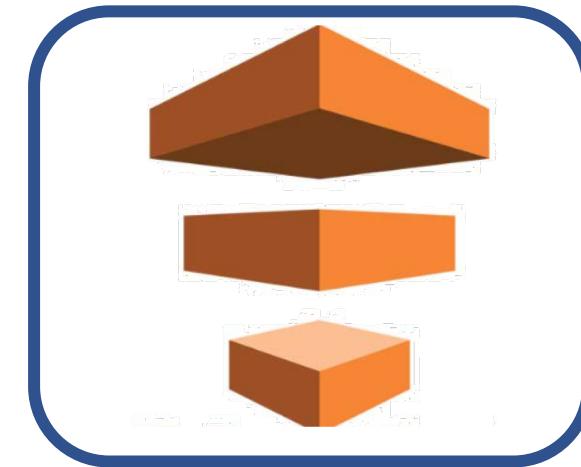


AWS DATA PIPELINE



AWS DATA PIPELINE

- AWS Data Pipeline is an **orchestration service** that allows AWS users to transfer data reliably and securely between various AWS compute and storage services.
- You can transfer data to Amazon S3, Amazon RDS, Amazon DynamoDB, and Amazon EMR.
- AWS manages the workflow so minimum maintenance is required.
- AWS guarantees resource availability and handles failures.



**AMAZON DATA
PIPELINE**

AWS DATA PIPELINE: EXAMPLE #1

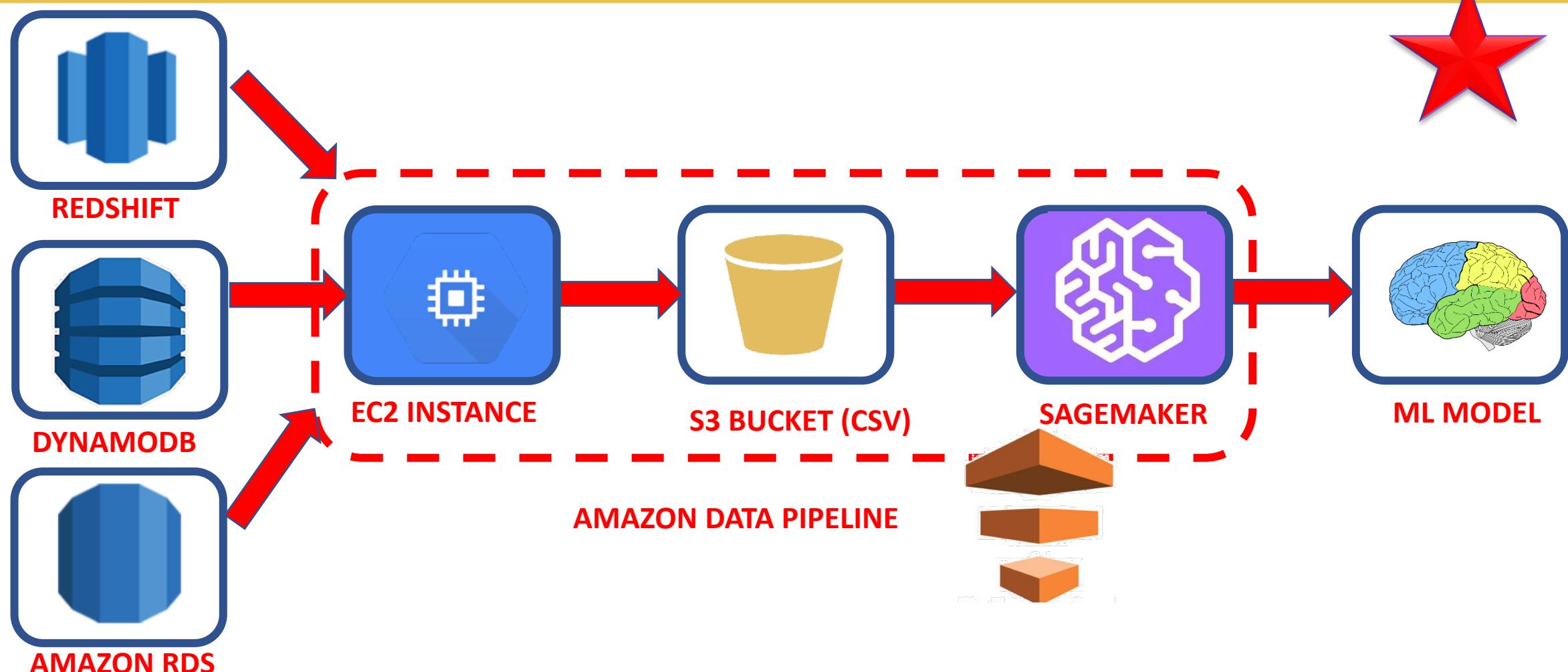


Photo Credit: https://en.wikipedia.org/wiki/File:User_icon_2.svg

<https://pixabay.com/illustrations/brain-lobes-neurology-human-body-1007686/>

<https://en.wikipedia.org/wiki/File:Google-Compute-Engine-Logo.svg>

AWS DATA PIPELINE: EXAMPLE#2

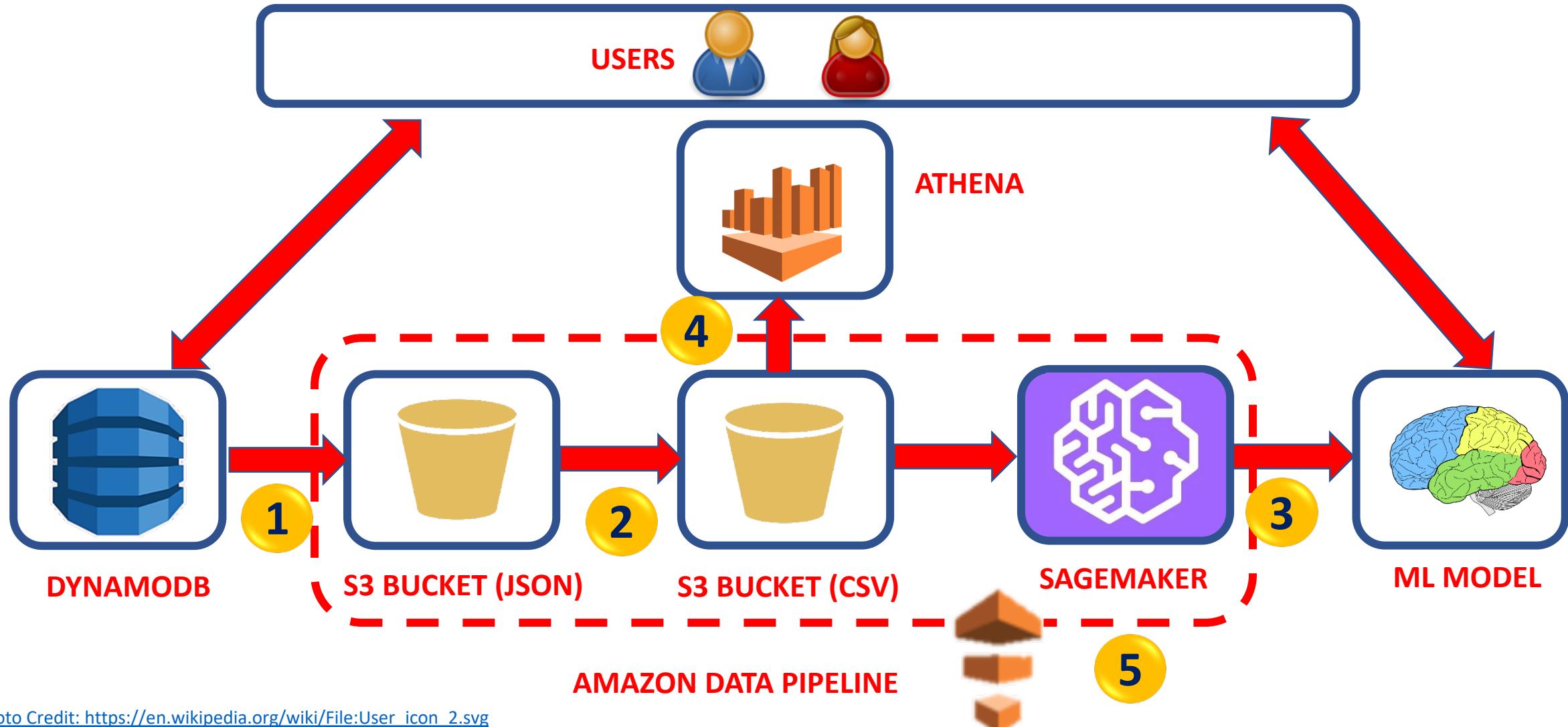
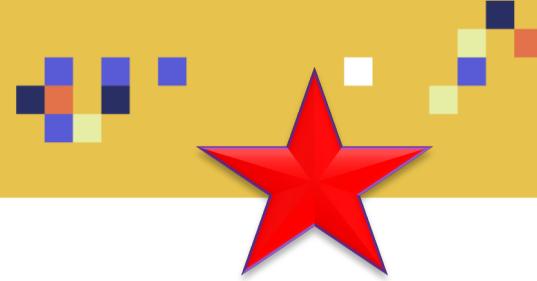


Photo Credit: https://en.wikipedia.org/wiki/File:User_icon_2.svg

<https://pixabay.com/illustrations/brain-lobes-neurology-human-body-1007686/>

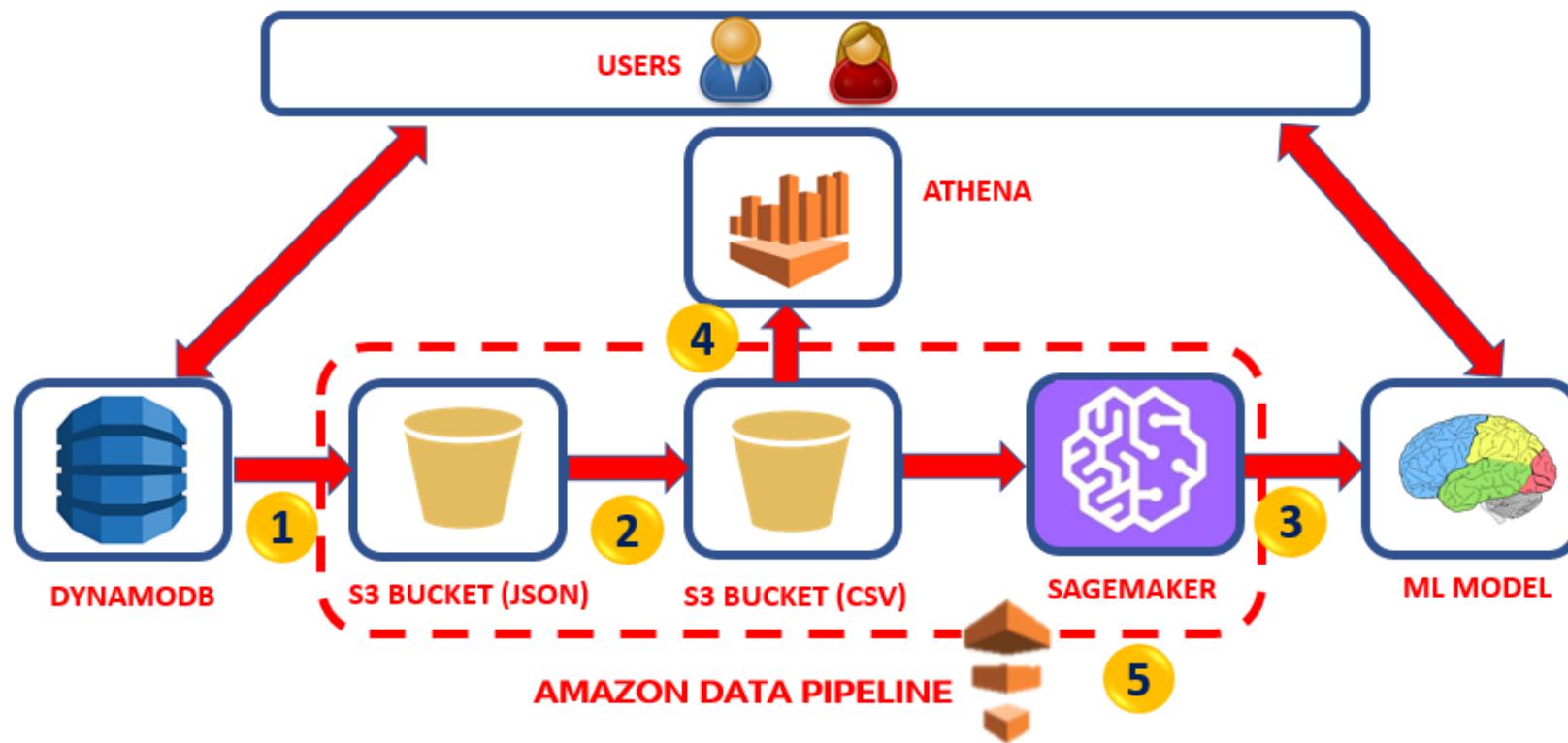
<https://aws.amazon.com/blogs/big-data/analyze-data-in-amazon-dynamodb-using-amazon-sagemaker-for-real-time-prediction/>

AWS DATA PIPELINE: EXAMPLE



The steps are as follows:

1. Data in DynamoDB is copied to Amazon S3 bucket (JSON) continuously as part of the Data Pipeline.
2. JSON files are then converted to CSV format to be used by Amazon SageMaker.
3. Amazon SageMaker consumes the data, trains the model, update endpoint for inference.
4. The data in CSV format is being queried (ad hoc) by Amazon Athena.
5. Data Pipeline manages the workflow as per the customer requirements.



WHEN SHOULD I USE AWS GLUE VS. DATA PIPELINE?



AWS GLUE:

- Glue is an ETL service that runs on a serverless Apache Spark environment.
- As a user, you do not have to configure or manage resources.
- Glue contains a data catalogue for ETL that could be used with Athena and Redshift Spectrum.
- AWS Glue ETL jobs uses Scala or Python.

AWS DATA PIPELINE:

- AWS Data Pipeline is a managed orchestration service.
- AWS Data pipeline offers more flexibility over EC2 instances that runs your code.
- If you are not using Apache Spark, its recommended to use AWS data pipeline.



AWS PIPELINE TEMPLATES



READYLY AVAILABLE TEMPLATES FOR
REDSHIFT, RDS, AND DYNAMODB

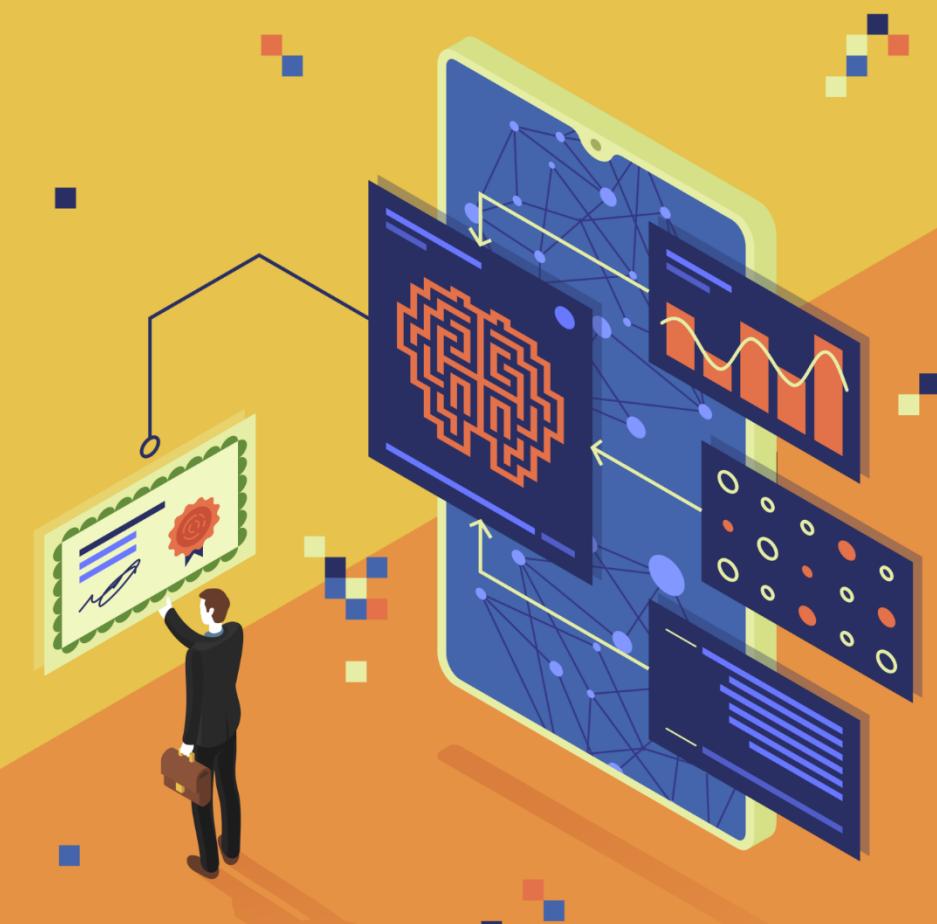
The screenshot shows the AWS Data Pipeline landing page. At the top is a dark header bar with a bell icon and the text "Ry". Below it is a large circular icon with a stylized orange and black design. The main title "AWS Data Pipeline" is centered above a brief description: "AWS Data Pipeline helps you move, integrate, and process data across AWS compute and storage resources, as well as your on-premises resources. AWS Data Pipeline supports integration of data and activities across multiple AWS regions." A blue "Get started now" button is at the bottom of this section. Below the description are three cards: "Define Data Nodes" (with a database icon), "Schedule Compute Activities" (with a computer monitor and gear icon), and "Activate & Monitor" (with a person icon). Each card has a brief description and a "Learn more" link.

The screenshot shows the "Create Pipeline" wizard. The first step, "Source", is selected. It contains fields for "Name" and "Description (optional)". Under "Source", there is a radio button for "Build using a template" which is selected, and a dropdown menu showing several options: "Choose...", "Choose...", "Getting Started", "AWS Command Line Interface (CLI) Templates", "DynamoDB Templates", "Elastic MapReduce (EMR) Templates", "RDS Templates", "Redshift Templates", and "Amazon Kinesis Templates". The "Full copy of RDS MySQL table to S3" option under "RDS Templates" is highlighted with a blue selection bar. Below the dropdown are sections for "Schedule", "Run", "Run every", "Starting", and "Ending". The "Starting" section shows a date and time input field set to "2019-12-01 00:49 UTC". At the bottom right is a "Next Step" button.

The screenshot shows the AWS Data Pipeline navigation bar. It includes links for "Additional Resources" (Data Pipeline Overview, Getting Set Up, FAQs, Pricing), "Support" (Forum, Documentation, Developer Tools, Support Center), and "Pipeline Configuration" (Logging, Enabled/Disabled). There is also a "Copy execution ID" link on the far right.

The screenshot shows the "Pipeline Configuration" step of the wizard. It has a "Logging" section with a radio button for "Enabled" which is selected. At the bottom right is a "Next Step" button.

AWS DATA MIGRATION SERVICE (DMS)



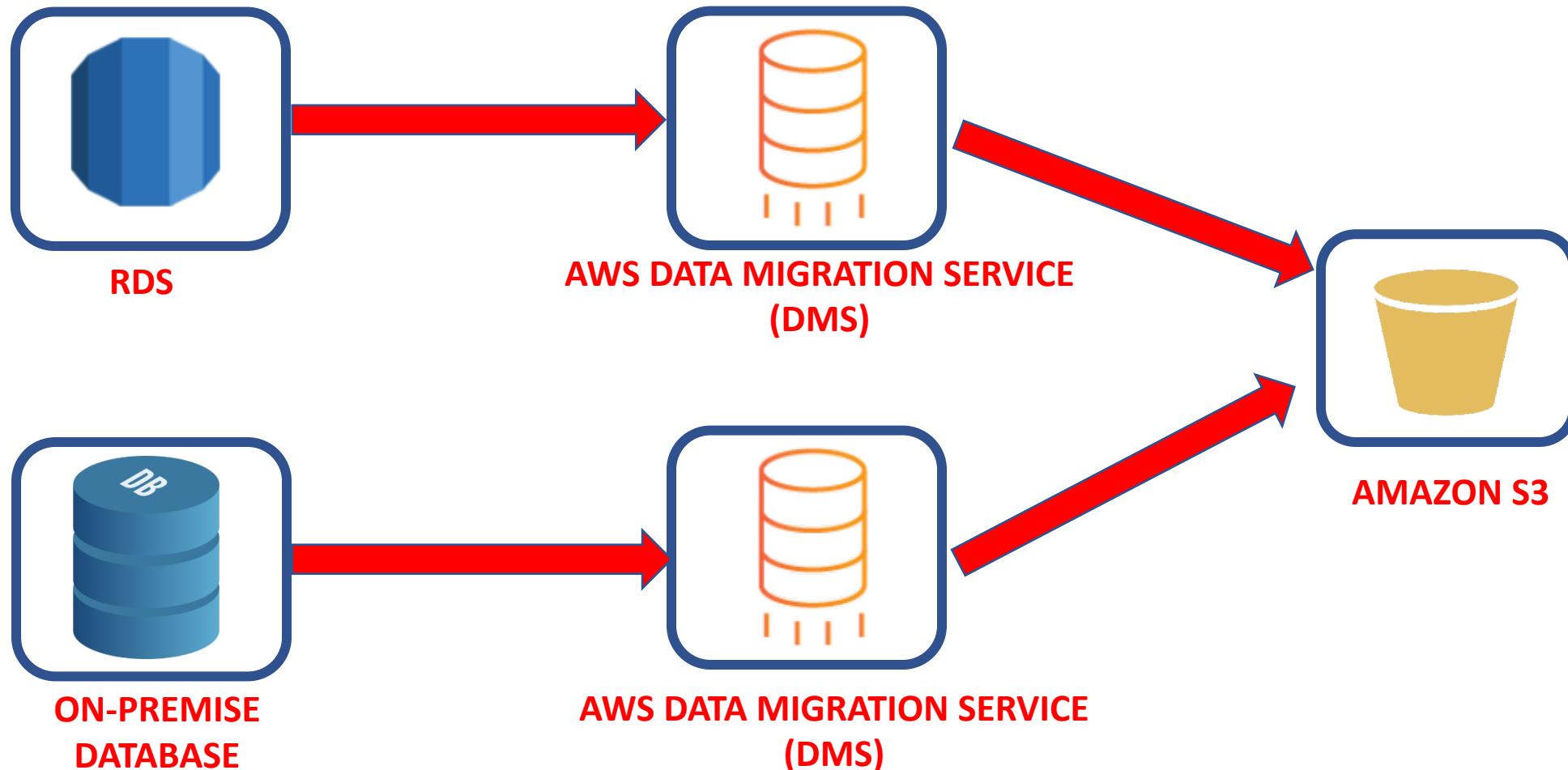
AWS DATA MIGRATION SERVICE (DMS)

- As the name stated, AWS Database Migration Service allows for database migration to AWS quickly.
- Source database remains functional during migration.
- Oracle to oracle migration or oracle to Aurora (RDS).
- Allows for database consolidation.
- Allows for data replication in a Data warehouse such as Amazon Redshift and S3.
- Watch Video: <https://aws.amazon.com/dms/>



**AWS DATA MIGRATION
SERVICE (DMS)**

AWS DATA MIGRATION SERVICE (DMS)



AWS DATA MIGRATION SERVICE (DMS)



SELECT THE SOURCE AND ENDPOINT

The screenshot shows the AWS DMS landing page. The main heading is "AWS Database Migration Service" with the subtext "Migrate your databases to AWS with minimal downtime". Below this, there's a section titled "How it works" featuring a diagram of three database instances (two on-premises and one in AWS) connected by dashed arrows over a world map, with a play button icon indicating a video tutorial. A text box explains that the service helps migrate databases quickly and securely, keeping the source database operational during the process. There are also sections for "Free DMS", "AWS Schema Conversion Tool", and "Use cases".

The screenshot shows the "Endpoint type" configuration screen. It includes two radio buttons: "Source endpoint" (selected) and "Target endpoint". The "Source endpoint" section describes it as allowing AWS DMS to read data from a database (on-premises or in the cloud) or from other data sources like Amazon S3. The "Target endpoint" section describes it as allowing AWS DMS to write data to a database or other data source. A red arrow points from the "Source endpoint" text in the landing page diagram to this configuration screen. The right side of the screenshot shows the "Endpoint configuration" section with fields for "Endpoint identifier" (set to "ProdEndpoint"), "Source engine" (set to "Choose an engine" with a search bar containing "aurora"), and a dropdown menu listing various database engines: aurora, aurora-postgresql, s3, db2, mariadb, azuredb, sqlserver, mongodb, mysql, and oracle.

WHEN SHOULD I USE AWS GLUE VS. AWS MIGRATION SERVICE?

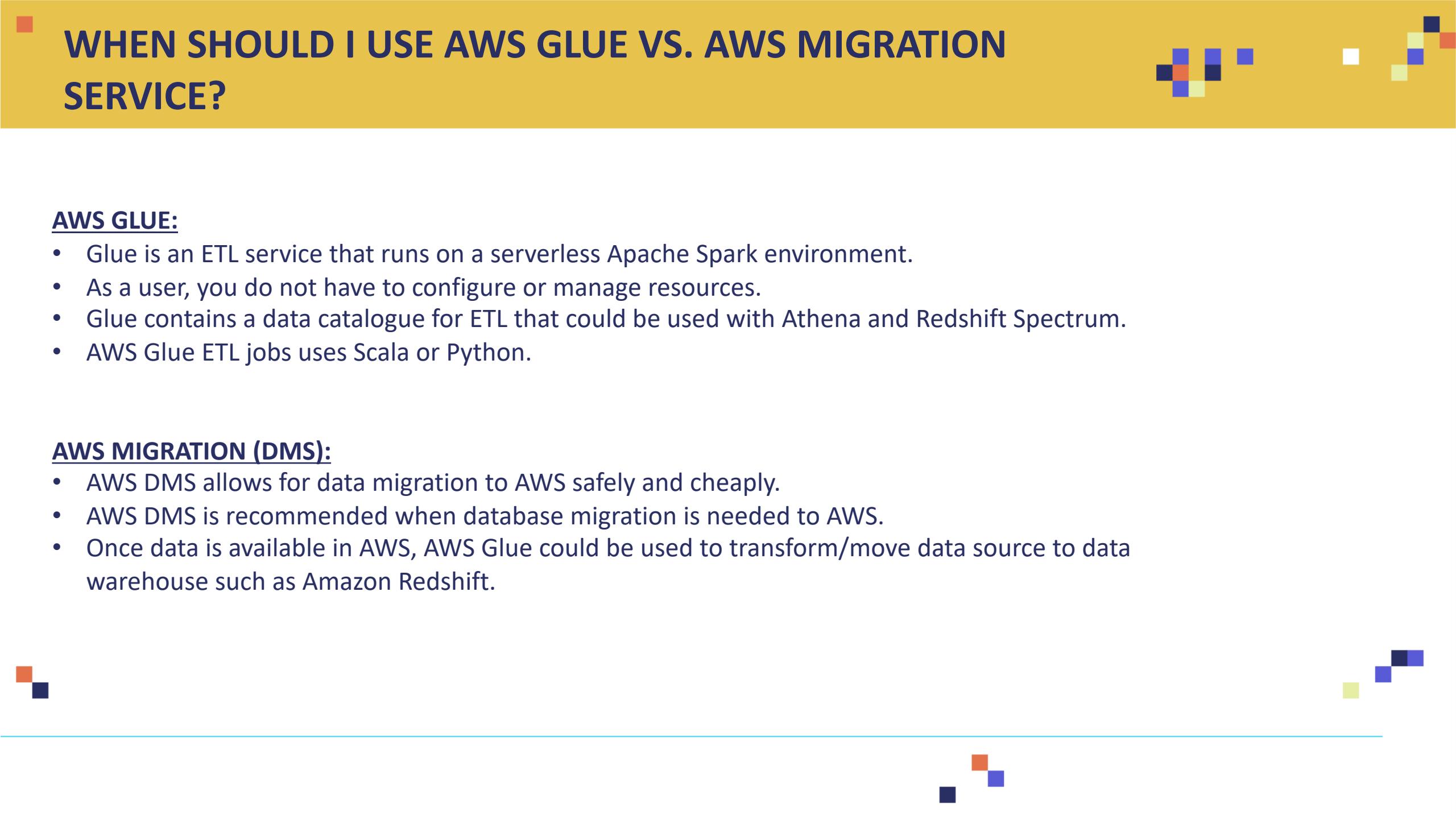


AWS GLUE:

- Glue is an ETL service that runs on a serverless Apache Spark environment.
- As a user, you do not have to configure or manage resources.
- Glue contains a data catalogue for ETL that could be used with Athena and Redshift Spectrum.
- AWS Glue ETL jobs uses Scala or Python.

AWS MIGRATION (DMS):

- AWS DMS allows for data migration to AWS safely and cheaply.
- AWS DMS is recommended when database migration is needed to AWS.
- Once data is available in AWS, AWS Glue could be used to transform/move data source to data warehouse such as Amazon Redshift.



AWS BATCH



AWS BATCH

- AWS Batch allows for running of batch computing jobs on AWS.
- AWS batch optimizes the type and number of compute resources based on volume.
- No management or hassle (serverless), AWS takes care of the batch computing software and resources.
- AWS Batch performs all the scheduling and execution of the batch using Amazon EC2 and Spot Instances.
- Users pay for the EC2 resources that the batch run on.
- AWS Batch jobs could be scheduled using CloudWatch Events
- AWS batch jobs could be orchestrated using AWS Step Functions



AWS BATCH

WHEN SHOULD I USE AWS GLUE VS. BATCH?



AWS GLUE:

- Glue is an ETL service that runs on a serverless Apache Spark environment.
- As a user, you do not have to configure or manage resources.
- Glue contains a data catalogue for ETL that could be used with Athena and Redshift Spectrum.
- AWS Glue ETL jobs uses Scala or Python.

AWS BATCH:

- Regardless of the type of the job, AWS Batch enables running of batch computing jobs on AWS.
- AWS Batch creates and manages the compute resources in the AWS account.
- AWS batch gives users full visibility and control over resources (EC2 instances).

For ETL use cases, use AWS Glue

For any other batch services (that might include ETL), use AWS Batch

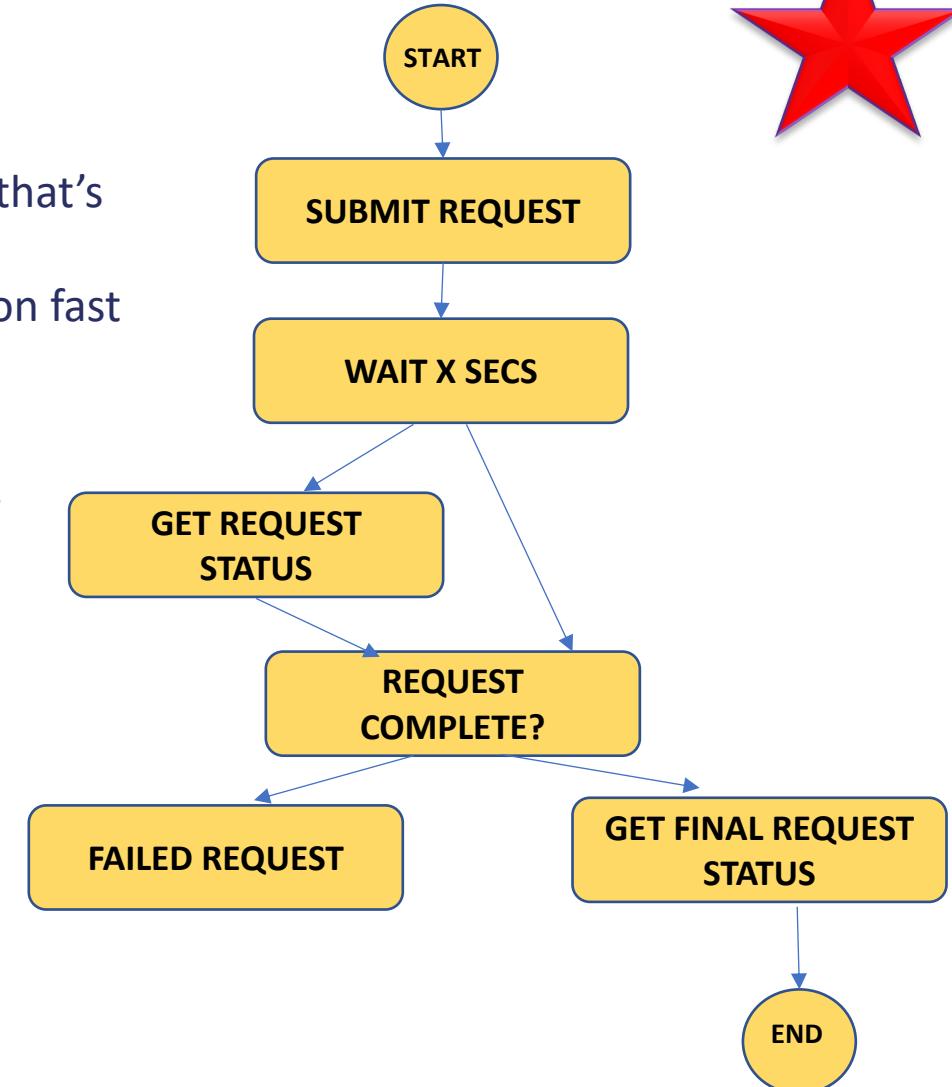


AWS STEP FUNCTION

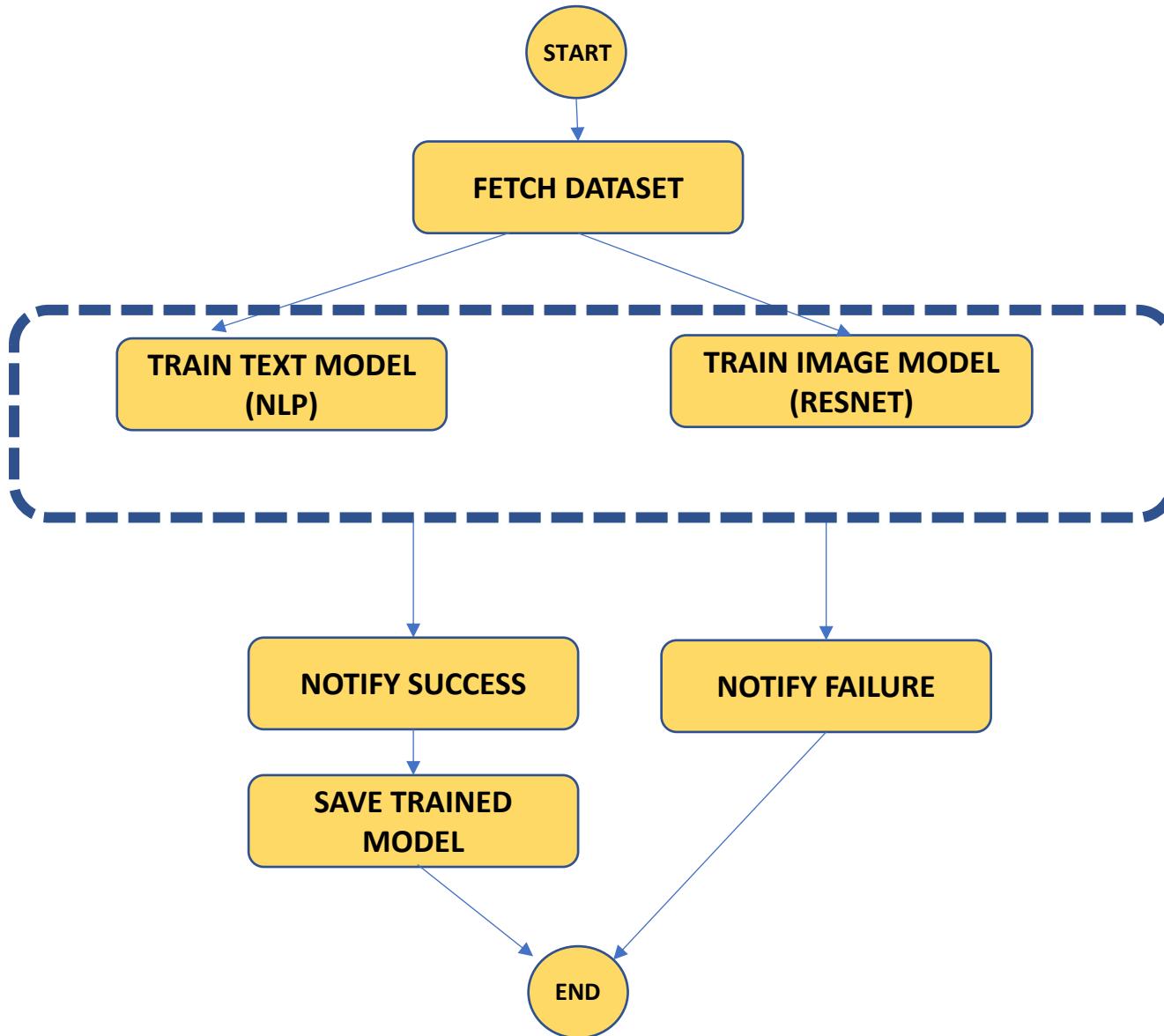


AMAZON AWS STEP FUNCTIONS

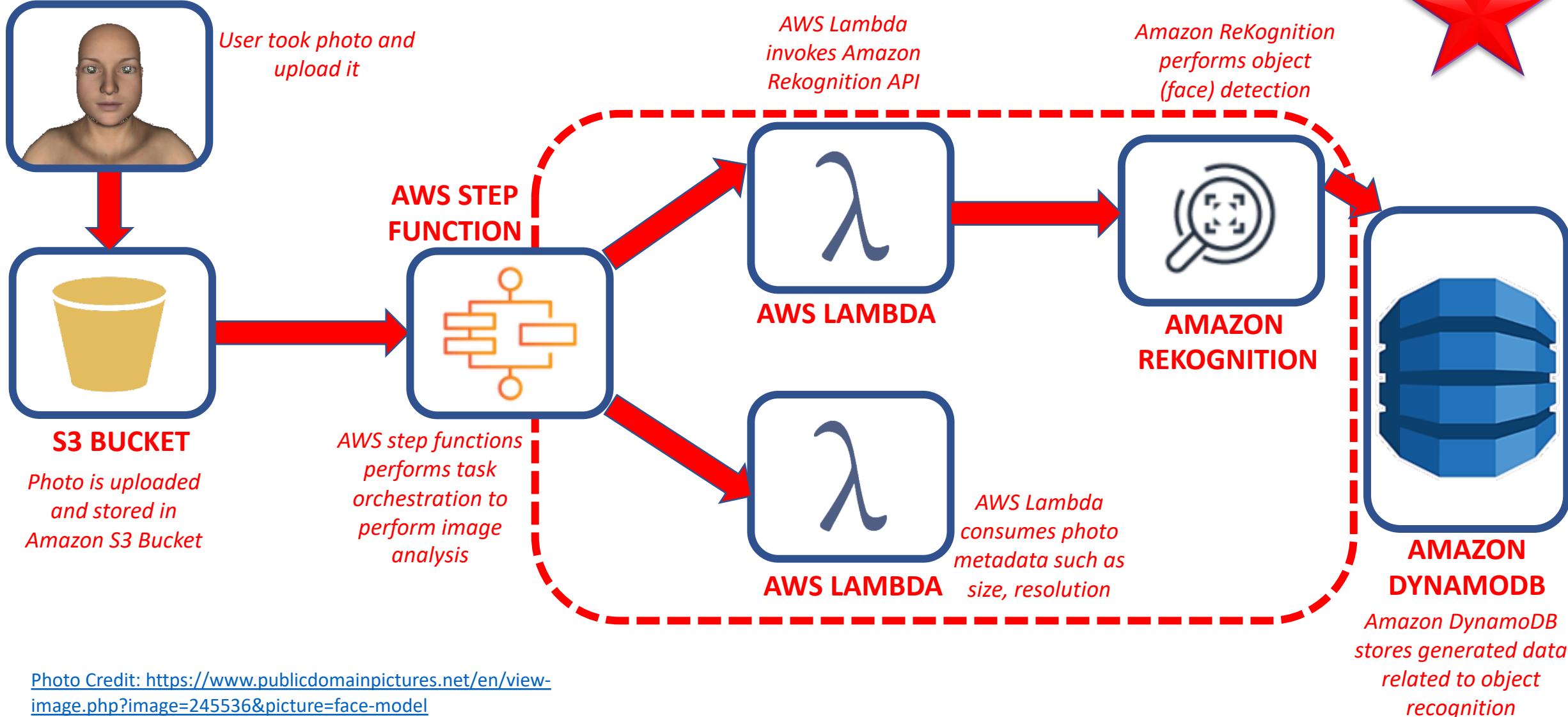
- AWS Step Functions allows for creating serverless workflows.
- Output from a step is fed as an input to the next step.
- AWS Step functions converts a workflow into a state machine diagram that's easy to debug and understand.
- AWS Step Functions allows for performing resilient workflow automation fast without writing code.
- It allows for advanced error handling and retrying mechanisms.
- Watch Video: <https://aws.amazon.com/step-functions/getting-started/>



AMAZON AWS STEP FUNCTIONS: MACHINE LEARNING EXAMPLE



AWS STEP FUNCTION: EXAMPLE #1



AWS STEP FUNCTION: EXAMPLE #2

