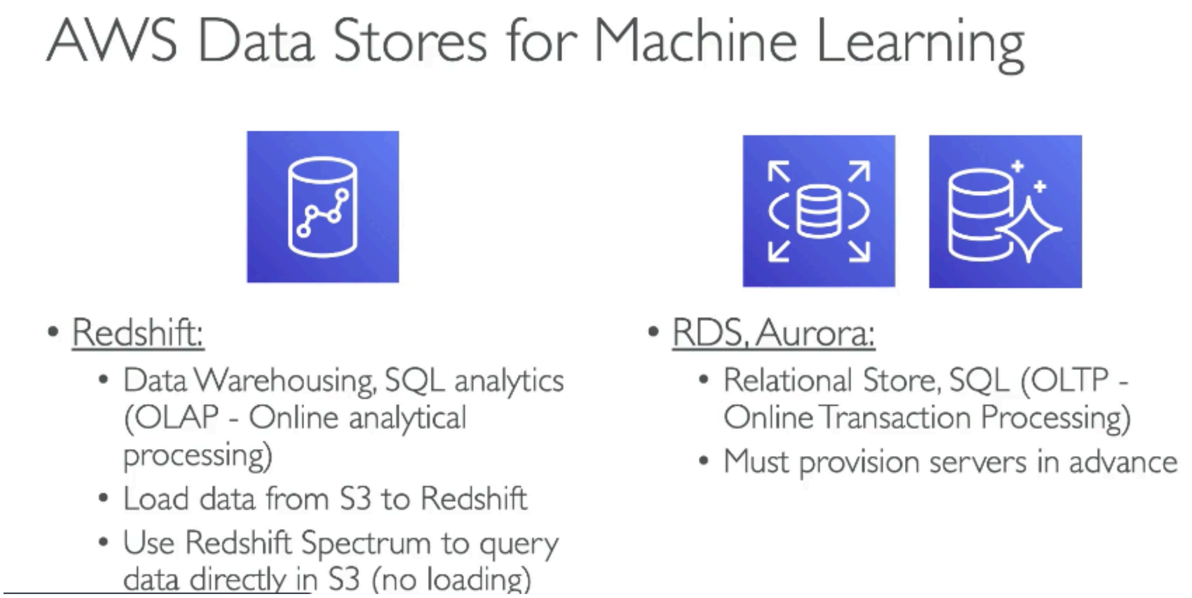
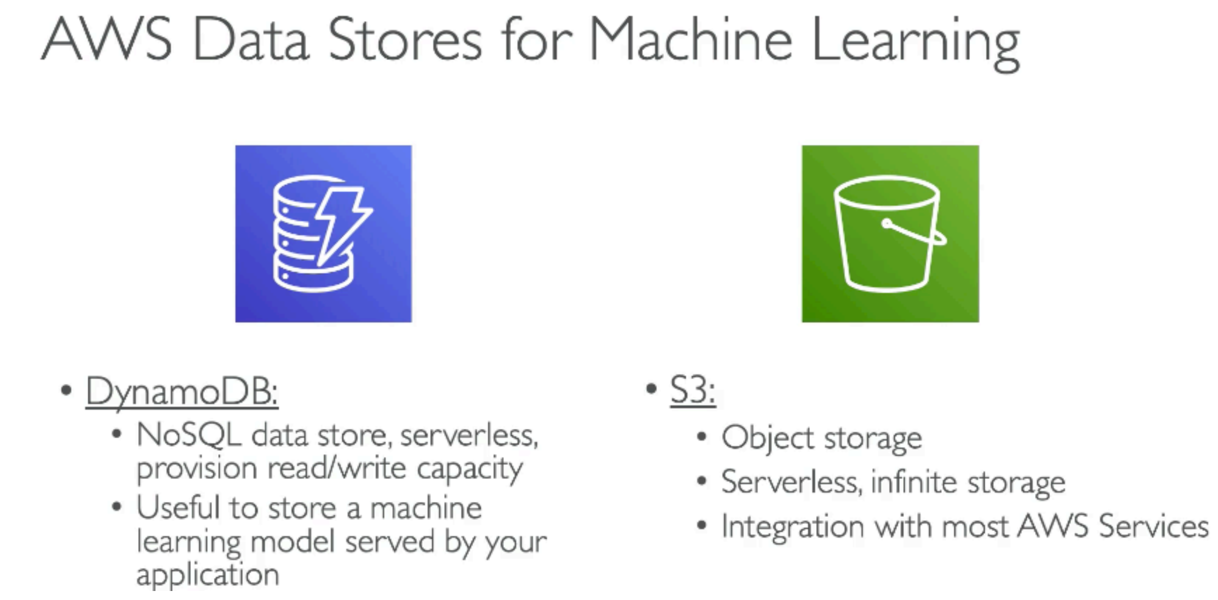
Udemy - 1 - Data Engineering - part 2

AWS Data Stores in Machine Learning



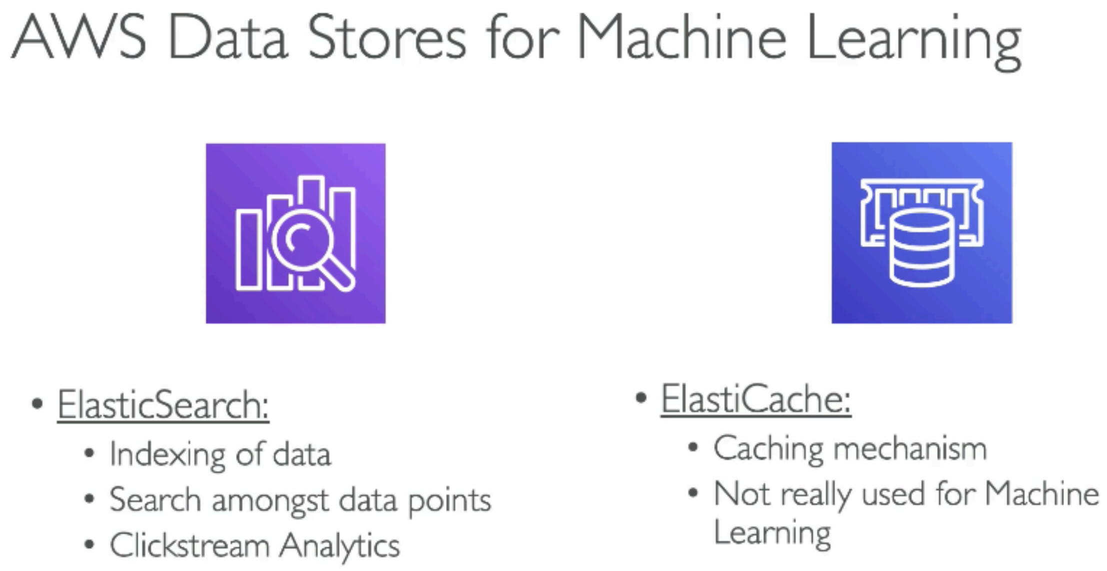
AWS Data Stores for Machine Learning

- Redshift:
  - Data Warehousing, SQL analytics (OLAP - Online analytical processing)
  - Load data from S3 to Redshift
  - Use Redshift Spectrum to query data directly in S3 (no loading)
- RDS, Aurora:
  - Relational Store, SQL (OLTP - Online Transaction Processing)
  - Must provision servers in advance



AWS Data Stores for Machine Learning

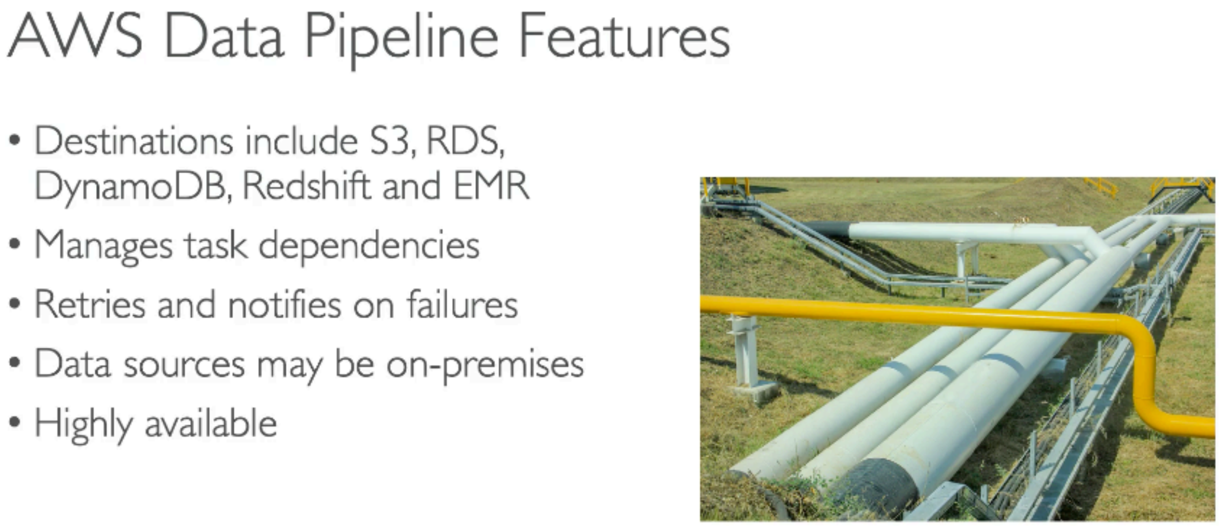
- DynamoDB:
  - NoSQL data store, serverless, provision read/write capacity
  - Useful to store a machine learning model served by your application
- S3:
  - Object storage
  - Serverless, infinite storage
  - Integration with most AWS Services



AWS Data Stores for Machine Learning

- ElasticSearch:
  - Indexing of data
  - Search amongst data points
  - Clickstream Analytics
- ElastiCache:
  - Caching mechanism
  - Not really used for Machine Learning

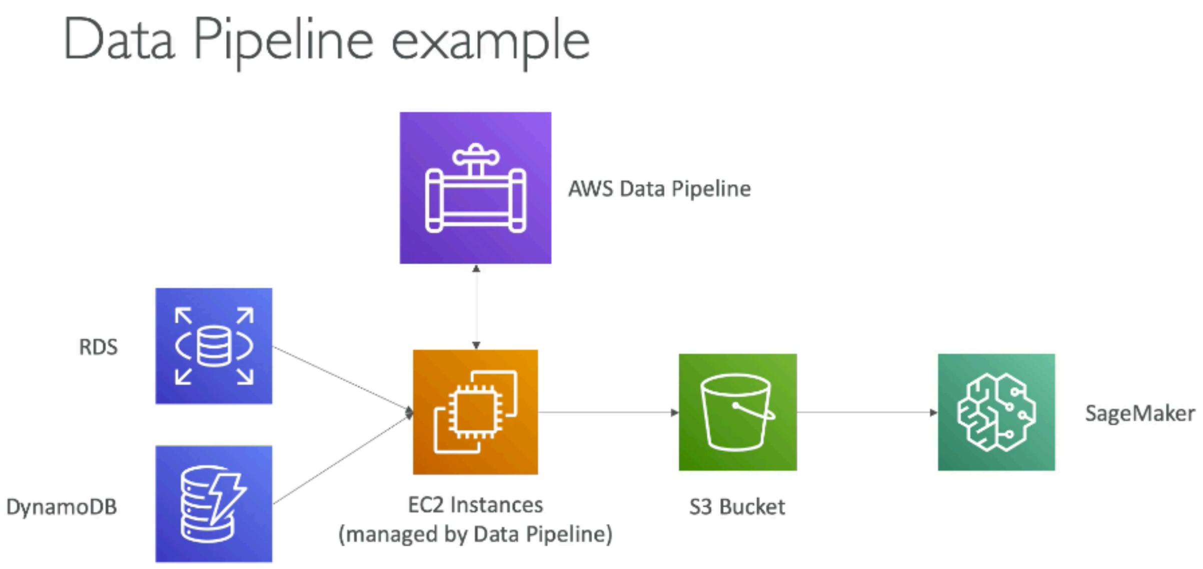
AWS Data Pipelines
It is just an **ORCHESTRATOR**. EC2 instances will be handling the compute



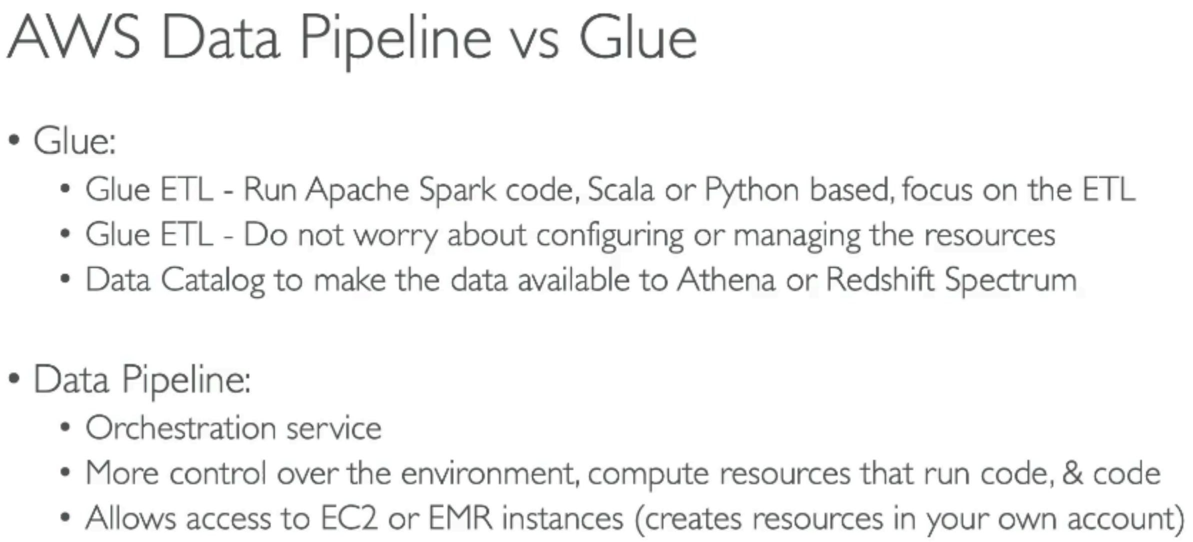
AWS Data Pipeline Features

- Destinations include S3, RDS, DynamoDB, Redshift and EMR
- Manages task dependencies
- Retries and notifies on failures
- Data sources may be on-premises
- Highly available

Why would we use a Data Pipeline?
- Orchestrate and Move data from RDS to S3



Data Pipeline example

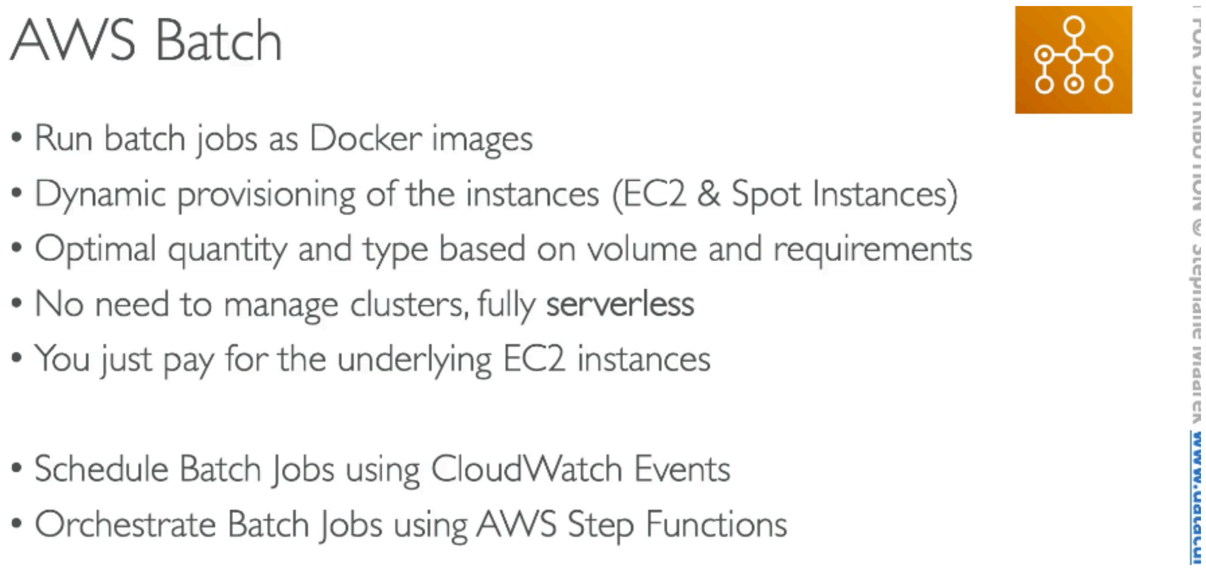


AWS Data Pipeline vs Glue

- Glue:
  - Glue ETL - Run Apache Spark code, Scala or Python based, focus on the ETL
  - Glue ETL - Do not worry about configuring or managing the resources
  - Data Catalog to make the data available to Athena or Redshift Spectrum
- Data Pipeline:
  - Orchestration service
  - More control over the environment, compute resources that run code, & code
  - Allows access to EC2 or EMR instances (creates resources in your own account)

Both are ETL services
 - **Glue** is more Apache Spark focused, ETL focused with Transform
 - **Data Pipeline** gives us a bit more control, run on EC2 or EMR instances from within our account, gives us a bit more control.

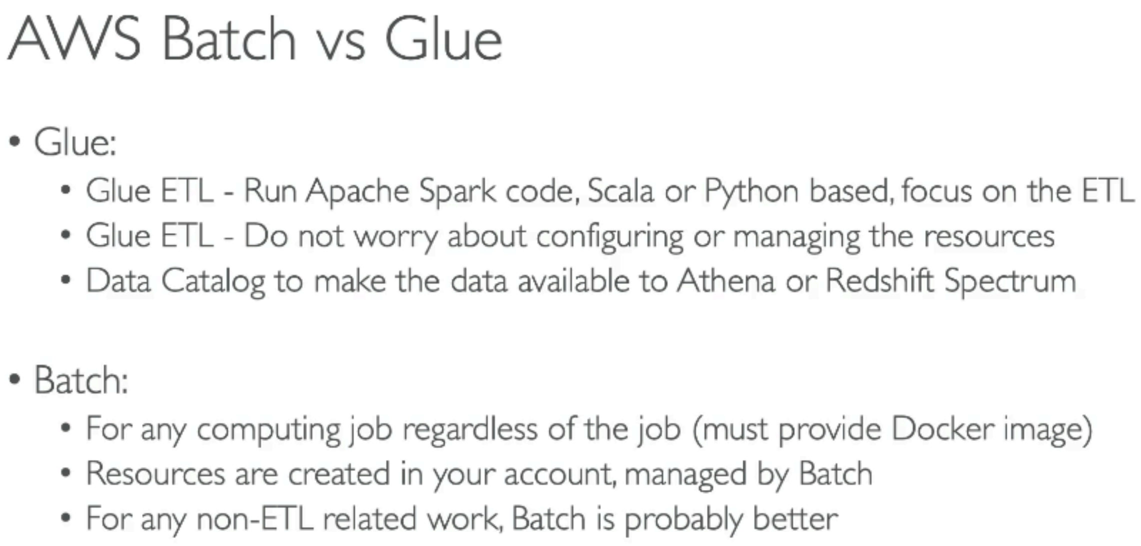
AWS Batch



AWS Batch

- Run batch jobs as Docker images
- Dynamic provisioning of the instances (EC2 & Spot Instances)
- Optimal quantity and type based on volume and requirements
- No need to manage clusters, fully **serverless**
- You just pay for the underlying EC2 instances

- Schedule Batch Jobs using CloudWatch Events
- Orchestrate Batch Jobs using AWS Step Functions

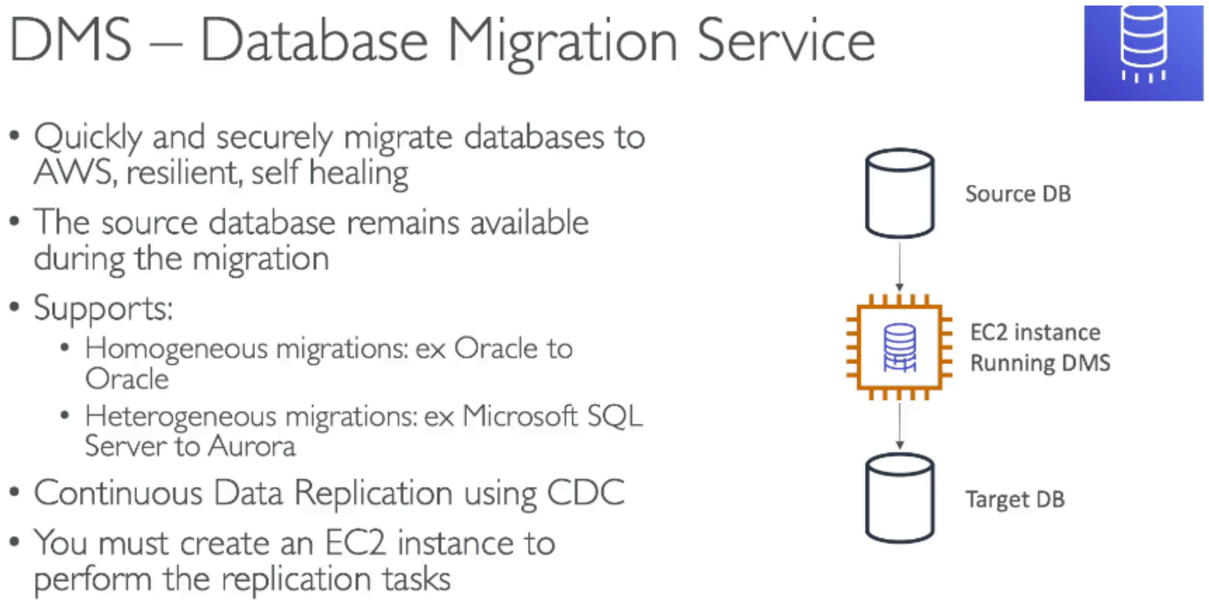


AWS Batch vs Glue

- Glue:
  - Glue ETL - Run Apache Spark code, Scala or Python based, focus on the ETL
  - Glue ETL - Do not worry about configuring or managing the resources
  - Data Catalog to make the data available to Athena or Redshift Spectrum

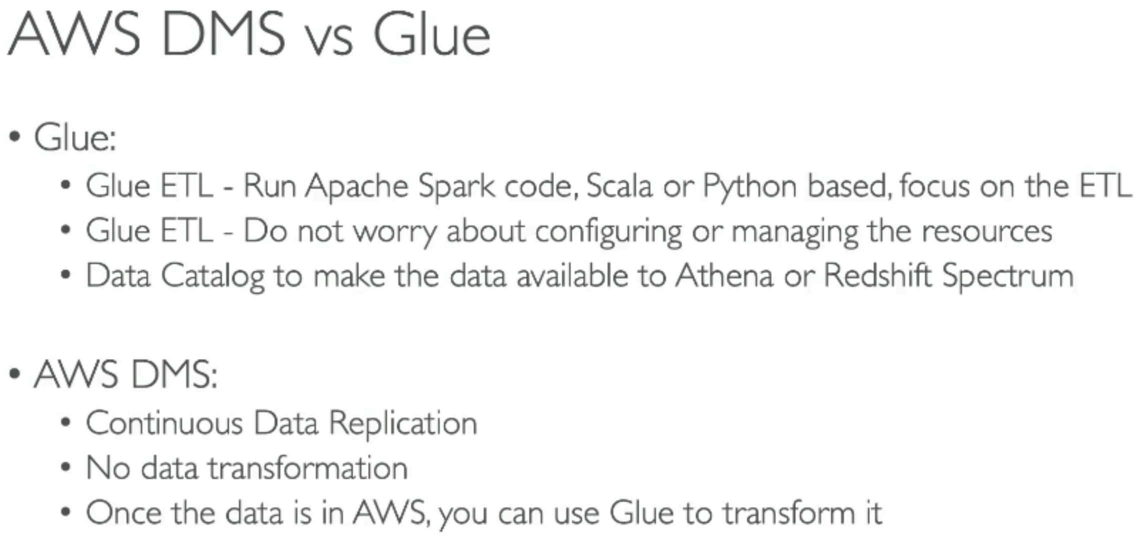
- Batch:
  - For any computing job regardless of the job (must provide Docker image)
  - Resources are created in your account, managed by Batch
  - For any non-ETL related work, Batch is probably better

DMS - Database Migration Service



DMS – Database Migration Service

- Quickly and securely migrate databases to AWS, resilient, self healing
- The source database remains available during the migration
- Supports:
  - Homogeneous migrations: ex Oracle to Oracle
  - Heterogeneous migrations: ex Microsoft SQL Server to Aurora
- Continuous Data Replication using CDC
- You must create an EC2 instance to perform the replication tasks



AWS DMS vs Glue

- Glue:
  - Glue ETL - Run Apache Spark code, Scala or Python based, focus on the ETL
  - Glue ETL - Do not worry about configuring or managing the resources
  - Data Catalog to make the data available to Athena or Redshift Spectrum
- AWS DMS:
  - Continuous Data Replication
  - No data transformation
  - Once the data is in AWS, you can use Glue to transform it

AWS Step Functions

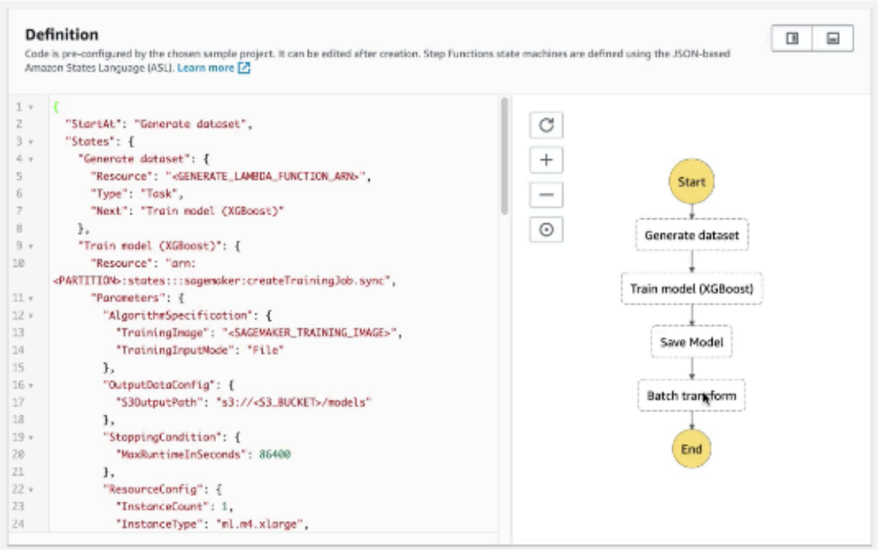
Define/design workflow

AWS Step Functions

- Use to design workflows
- Easy visualizations
- Advanced Error Handling and Retry mechanism outside the code
- Audit of the history of workflows
- Ability to "Wait" for an arbitrary amount of time
- Max execution time of a State Machine is 1 year

Example:

Step Functions – Examples
Train a Machine Learning Model

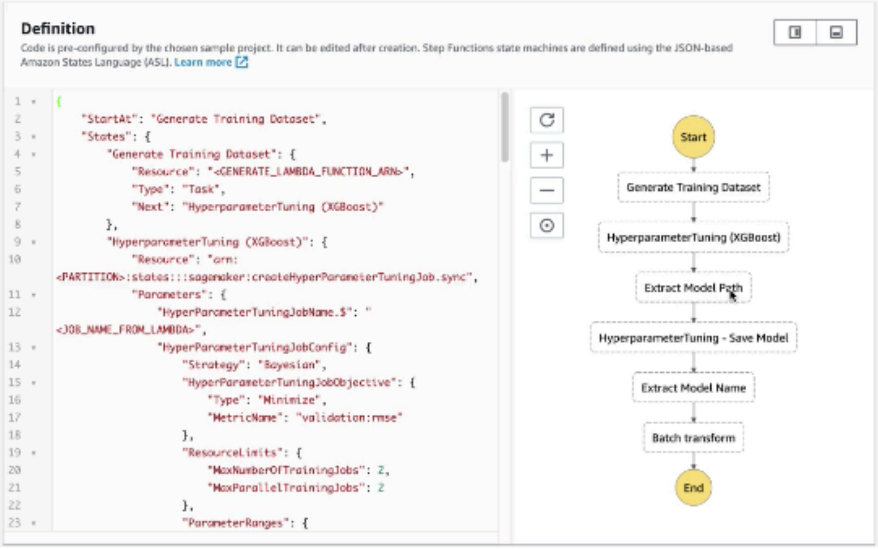


Definition

Code is pre-configured by the chosen sample project. It can be edited after creation. Step Functions state machines are defined using the JSON-based Amazon States Language (ASL). Learn more

```
{
  "StartAt": "Generate dataset",
  "States": {
    "Generate dataset": {
      "Resource": "<GENERATE_LAMBDA_FUNCTION_ARN>",
      "Type": "Task",
      "Next": "Train model (XGBoost)"
    },
    "Train model (XGBoost)": {
      "Resource": "arn:
<PARTITION>:states:::sagemaker:createTrainingJob.sync",
      "Parameters": {
        "AlgorithmSpecification": {
          "TrainingImage": "<SAGEMAKER_TRAINING_IMAGE>",
          "TrainingInputMode": "File"
        },
        "OutputDataConfig": {
          "S3OutputPath": "s3://<S3_BUCKET>/models"
        },
        "StoppingCondition": {
          "MaxRuntimeInSeconds": 86400
        },
        "ResourceConfig": {
          "InstanceCount": 1,
          "InstanceType": "ml.m4.xlarge",
```

Step Functions – Examples
Tune a Machine Learning Model



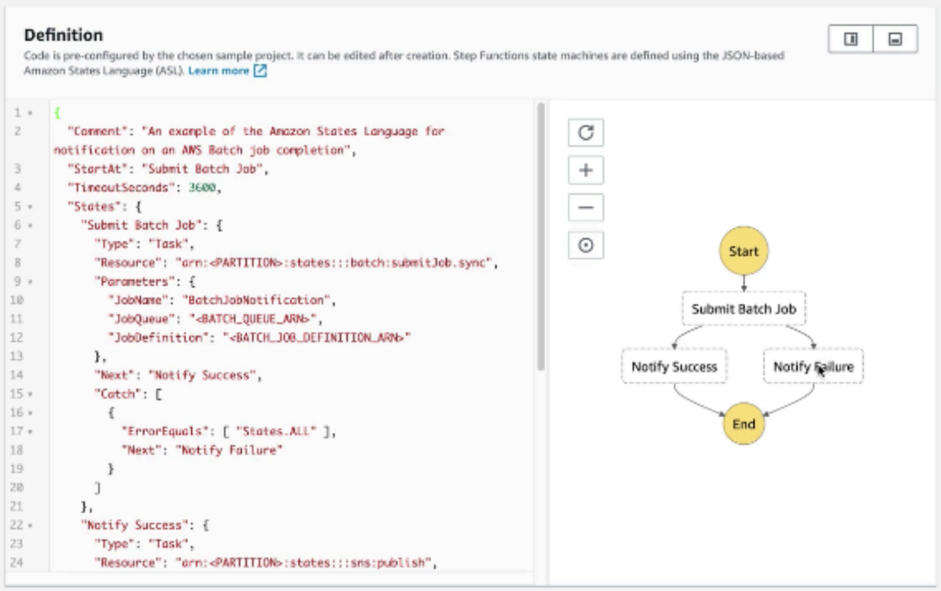
Definition

Code is pre-configured by the chosen sample project. It can be edited after creation. Step Functions state machines are defined using the JSON-based Amazon States Language (ASL). Learn more

```
{
    "StartAt": "Generate Training Dataset",
    "States": {
        "Generate Training Dataset": {
            "Resource": "<GENERATE_LAMBDA_FUNCTION_ARN>",
            "Type": "Task",
            "Next": "HyperparameterTuning (XGBoost)"
        },
        "HyperparameterTuning (XGBoost)": {
            "Resource": "arn:
<PARTITION>:states:::sagemaker:createHyperParameterTuningJob.sync",
            "Parameters": {
                "HyperParameterTuningJobName.$": "
<JOB_NAME_FROM_LAMBDA>",
                "HyperParameterTuningJobConfig": {
                    "Strategy": "Bayesian",
                    "HyperParameterTuningJobObjective": {
                        "Type": "Minimize",
                        "MetricName": "validation:rmse"
                    },
                    "ResourceLimits": {
                        "MaxNumberOfTrainingJobs": 2,
                        "MaxParallelTrainingJobs": 2
                    },
                    "ParameterRanges": {
```

Start
Generate Training Dataset
HyperparameterTuning (XGBoost)
Extract Model Path
HyperparameterTuning - Save Model
Extract Model Name
Batch transform
End

Step Functions – Examples
Manage a Batch Job



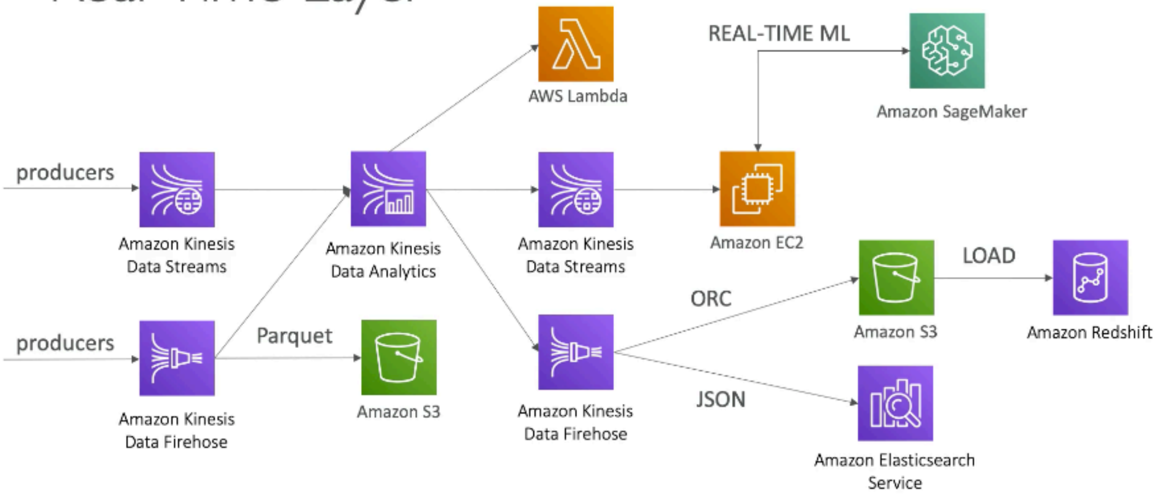
Definition

Code is pre-configured by the chosen sample project. It can be edited after creation. Step Functions state machines are defined using the JSON-based Amazon States Language (ASL). Learn more

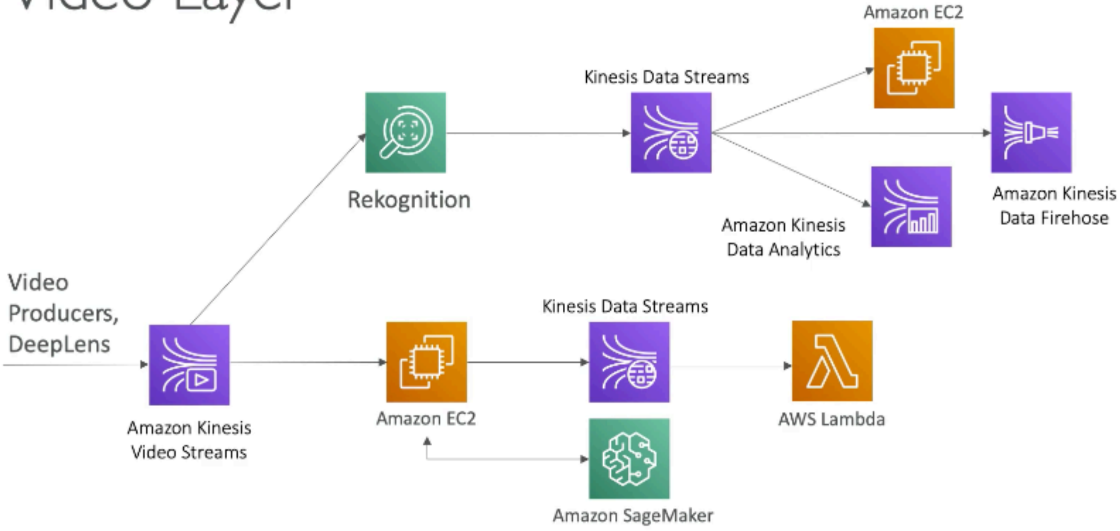
```
{
  "Comment": "An example of the Amazon States Language for
notification on an AWS Batch job completion",
  "StartAt": "Submit Batch Job",
  "TimeoutSeconds": 3600,
  "States": {
    "Submit Batch Job": {
      "Type": "Task",
      "Resource": "arn:<PARTITION>:states:::batch:submitJob.sync",
      "Parameters": {
        "JobName": "BatchJobNotification",
        "JobQueue": "<BATCH_QUEUE_ARN>",
        "JobDefinition": "<BATCH_JOB_DEFINITION_ARN>"
      },
      "Next": "Notify Success",
      "Catch": [
        {
          "ErrorEquals": [ "States.ALL" ],
          "Next": "Notify Failure"
        }
      ]
    },
    "Notify Success": {
      "Type": "Task",
      "Resource": "arn:<PARTITION>:states:::sns:publish",
```

Summarizing All these services - Data Engineering Pipelines

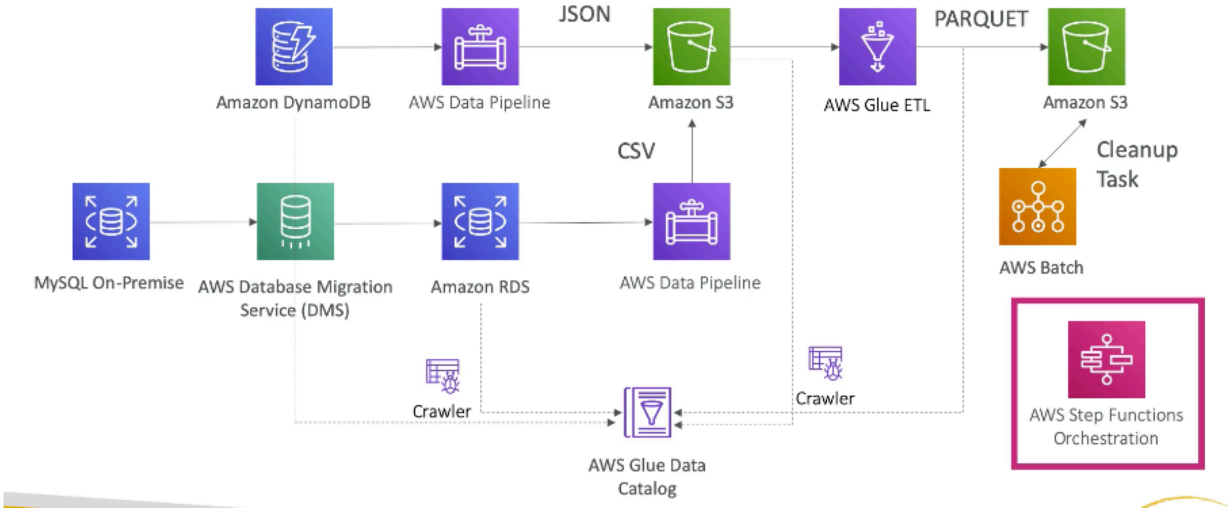
Full Data Engineering Pipeline
Real-Time Layer



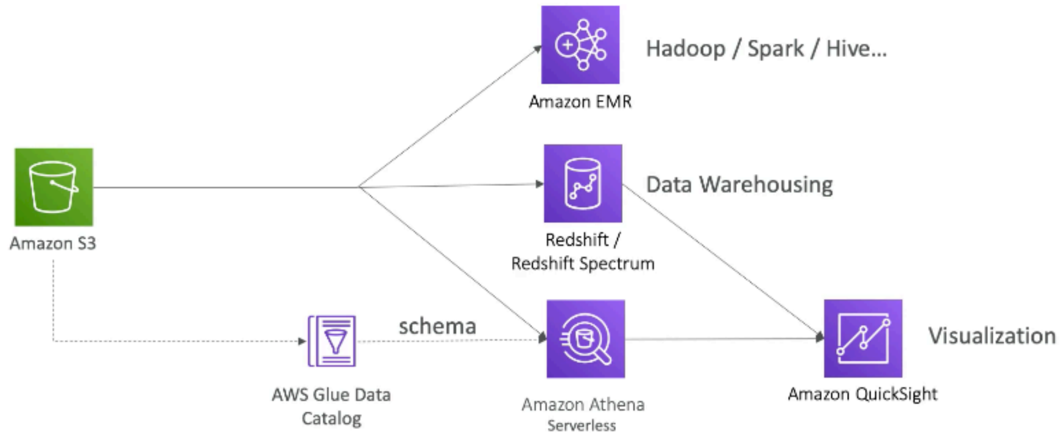
Full Data Engineering Pipeline
Video Layer

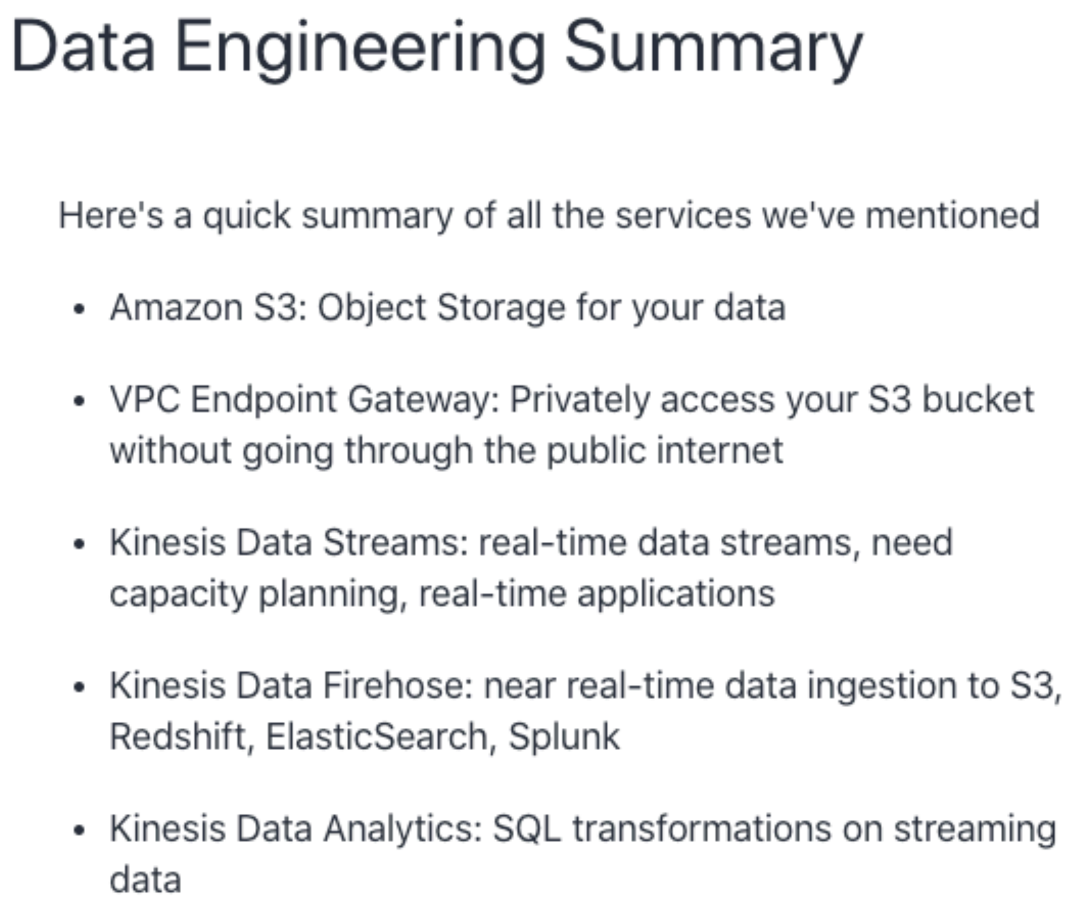


Full Data Engineering Pipeline
Batch Layer



Full Data Engineering Pipeline
Analytics layer



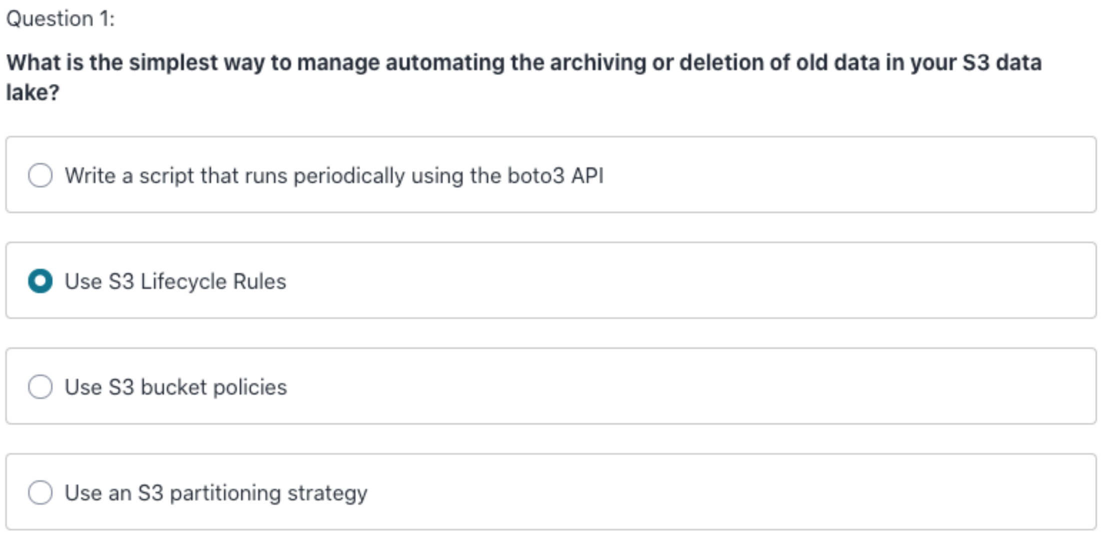


Data Engineering Summary

Here's a quick summary of all the services we've mentioned

- Amazon S3: Object Storage for your data
- VPC Endpoint Gateway: Privately access your S3 bucket without going through the public internet
- Kinesis Data Streams: real-time data streams, need capacity planning, real-time applications
- Kinesis Data Firehose: near real-time data ingestion to S3, Redshift, ElasticSearch, Splunk
- Kinesis Data Analytics: SQL transformations on streaming data

Quiz - Data Engineering



Question 1:

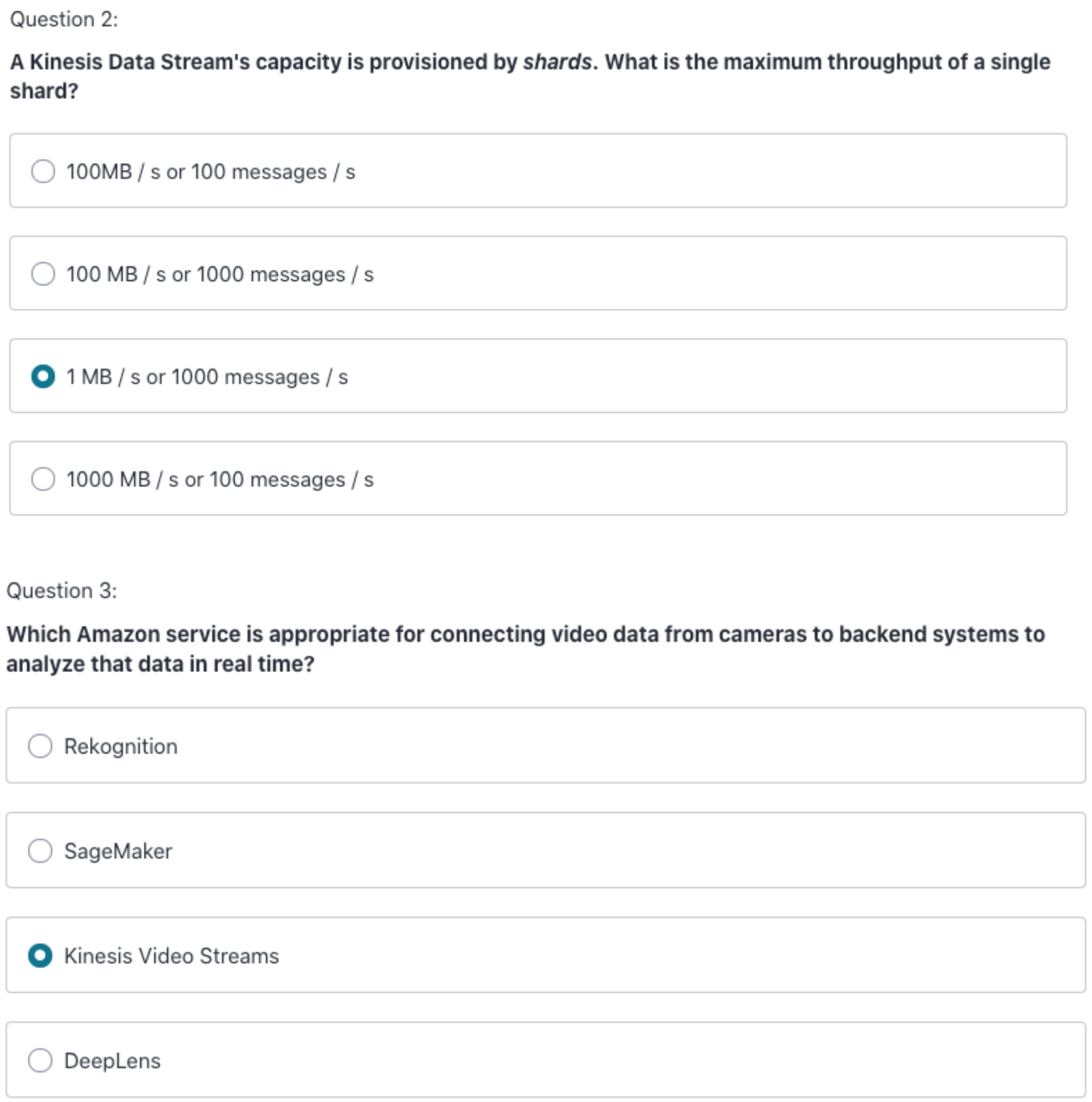
What is the simplest way to manage automating the archiving or deletion of old data in your S3 data lake?

○ Write a script that runs periodically using the boto3 API

● Use S3 Lifecycle Rules

○ Use S3 bucket policies

○ Use an S3 partitioning strategy



Question 2:

A Kinesis Data Stream's capacity is provisioned by *shards*. What is the maximum throughput of a single shard?

○ 100MB / s or 100 messages / s

○ 100 MB / s or 1000 messages / s

● 1 MB / s or 1000 messages / s

○ 1000 MB / s or 100 messages / s

Question 3:

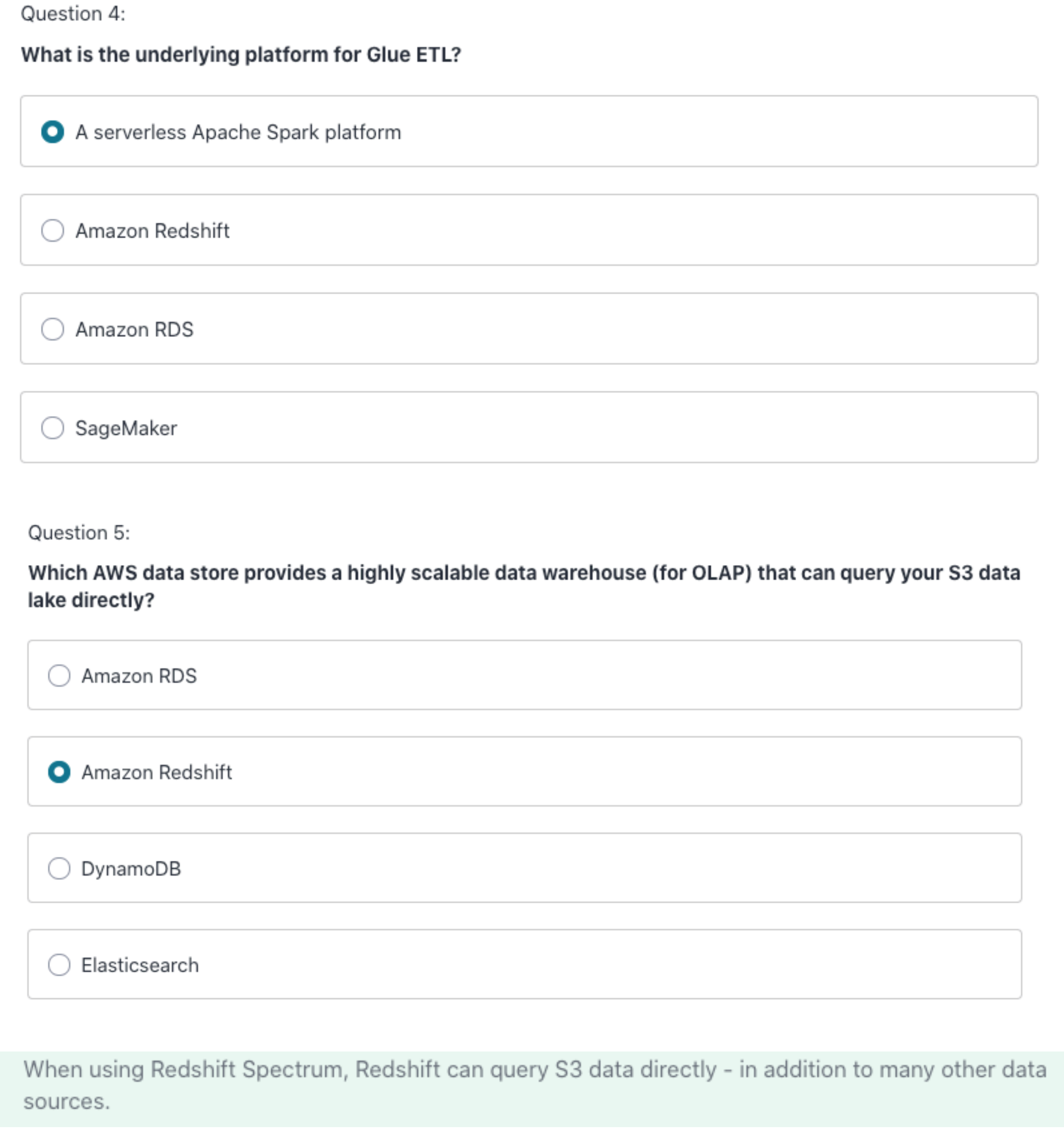
Which Amazon service is appropriate for connecting video data from cameras to backend systems to analyze that data in real time?

○ Rekognition

○ SageMaker

● Kinesis Video Streams

○ DeepLens



Question 4:

What is the underlying platform for Glue ETL?

● A serverless Apache Spark platform

○ Amazon Redshift

○ Amazon RDS

○ SageMaker

Question 5:

Which AWS data store provides a highly scalable data warehouse (for OLAP) that can query your S3 data lake directly?

○ Amazon RDS

● Amazon Redshift

○ DynamoDB

○ Elasticsearch

When using Redshift Spectrum, Redshift can query S3 data directly - in addition to many other data sources.