



# AWS Certified Data Engineer - Associate



# Types of Data



# Structured Data



**Organized In A  
Predefined Manor**



**Organized In A  
Predefined Manor**

**Searchable**



**Organized In A  
Predefined Manor**

**Searchable**

**Stored In Tabular  
Formats**



**Organized In A  
Predefined Manor**

**Searchable**

**Stored In Tabular  
Formats**

**Fields Are Descrete  
And Defined**



# Examples Of Structured Data



# Examples Of Structured Data

MySQL



# Examples Of Structured Data

**MySQL**

**PostgreSQL**



# Examples Of Structured Data

**MySQL**

**PostgreSQL**

**Amazon  
Redshift**



# Examples Of Structured Data

**MySQL**

**PostgreSQL**

**Amazon  
Redshift**

**Spreadsheets**



**Structured Data**

**UnStructured  
Data**



**Lacks a Predefined  
Manor or Organisation**



**Lacks a Predefined  
Manor or Organisation**

**Not Easily Searchable**



**Lacks a Predefined  
Manor or Organisation**

**Not Easily Searchable**

**Wide Variety of Data  
Types**



**Lacks a Predefined  
Manor or Organisation**

**Not Easily Searchable**

**Wide Variety of Data  
Types**

**Wide Variety of  
Formats**



# Examples Of Un- Structured Data



# Examples Of Un- Structured Data

**Text  
Documents**



# Examples Of Un- Structured Data

**Text  
Documents**

**Multi-  
media files**



# Examples Of Un- Structured Data

**Text  
Documents**

**Multi-  
media files**

**Social  
Media Post**



# Examples Of Un- Structured Data

**Text  
Documents**

**Multi-  
media files**

**Social  
Media Post**

**Emails**



**Structured Data**

**UnStructured  
Data**

**Semi-Structured  
Data**



# **Elements of Both Structured and Unstructured Data**



**Elements of Both  
Structured and  
Unstructured Data**

**Does Not Neatly Fit  
Tabular Format**



**Elements of Both  
Structured and  
Unstructured Data**

**Does Not Neatly Fit  
Tabular Format**

**Has Some  
Organizational  
Properties**



**Elements of Both  
Structured and  
Unstructured Data**

**Does Not Neatly Fit  
Tabular Format**

**Has Some  
Organizational  
Properties**

**Uses Tags/Markers To  
Separate Elements**



# Examples Of Un- Structured Data



# Examples Of Un- Structured Data

## XML Files



# Examples Of Un- Structured Data

**XML Files**

**JSON**



# Examples Of Un- Structured Data

**XML Files**

**JSON**

**NoSQL  
Databases**



# Examples Of Un- Structured Data

**XML Files**

**JSON**

**NoSQL  
Databases**

**HTML**



# Properties of Data



# **Three V's**

**Volume**

**Velocity**

**Variety**



# Volume of Data





**Scale or Quantity**  
of Data Being Produced



# Challenges

---



# Challenges

---

Storage  
Management



# Challenges

---

Storage  
Management

Processing  
Power



# Challenges

---

Storage  
Management

Processing  
Power

Data  
Integration



# **Velocity of Data**





**Speed**  
at which Data Being Produced



# Challenges

---



# Challenges

---

**Real-Time  
Processing**



# Challenges

---

**Real-Time  
Processing**

**Latency**



# Variety of Data





**Diversity**  
of data types, formats, and sources



# Challenges

---



# Challenges

---

**Data  
Integration**



# Challenges

---

Data  
Integration

Data Quality



# Challenges

---

Data  
Integration

Data Quality

Interoperability



**Data Warehouse**  
**Data Lakes**  
**Data Lakehouses**



# Data Warehouse



**Centralized repository that stores structured  
data from multiple sources**

---



**Centralized repository that stores structured data from multiple sources**

---

**01    Schema-on-Write**



# **Centralized repository that stores structured data from multiple sources**

---

**01** Schema-on-Write

**02** Stores structured data



# **Centralized repository that stores structured data from multiple sources**

---

**01** Schema-on-Write

**02** Stores structured data

**03** fast query performance



# **Centralized repository that stores structured data from multiple sources**

---

**01** Schema-on-Write

**02** Stores structured data

**03** fast query performance

**04** Ensures data integrity and consistency



# Examples

## Amazon Redshift



# Examples

Amazon Redshift  
Snowflake



# Examples

Amazon Redshift

Snowflake

Google BigQuery



# Data Lake



**Centralized repository to store data,  
both structured and unstructured, at  
any scale**

---



**Centralized repository to store data,  
both structured and unstructured, at  
any scale**

---

**01** Schema-on-Read



**Centralized repository to store data,  
both structured and unstructured, at  
any scale**

---

**01** Schema-on-Read

**02** Stores all data



**Centralized repository to store data,  
both structured and unstructured, at  
any scale**

---

**01** Schema-on-Read

**02** Stores all data

**03** Highly Scalable



# **Centralized repository to store data, both structured and unstructured, at any scale**

---

**01** Schema-on-Read

**02** Stores all data

**03** Highly Scalable

**04** Allows For Data  
Exploration and  
Experimentation



# Examples

# Apache Hadoop



# Examples

# Apache Hadoop

## S3



# Examples

Apache Hadoop

S3

Microsoft Azure  
Data Lake Storage



# Data Lakehouse



**Combines elements of data lakes and  
data warehouses to provide a unified  
data platform**

---



**Combines elements of data lakes and  
data warehouses to provide a unified  
data platform**

---

- 01** Stores Structured and Unstructured Data



# **Combines elements of data lakes and data warehouses to provide a unified data platform**

---

**01**

**Stores Structured and Unstructured Data**

**02**

**Schema-on-Read and Schema-on-Write**



# **Combines elements of data lakes and data warehouses to provide a unified data platform**

---

**01**

**Stores Structured and Unstructured Data**

**02**

**Schema-on-Read and Schema-on-Write**

**03**

**Real-Time Data Analytics**



# **Combines elements of data lakes and data warehouses to provide a unified data platform**

---

- 01** Stores Structured and Unstructured Data
- 02** Schema-on-Read and Schema-on-Write
- 03** Real-Time Data Analytics
- 04** Minimizes data movement and duplication



# Examples



# **Examples**

# **Databricks Lakehouse Platform**



# **Examples**

**Databricks Lakehouse  
Platform**

**Google BigLake**



# **Key Differences**



<b>Feature</b>	<b>Data Warehouse</b>	<b>Data Lake</b>	<b>Data Lakehouse</b>
Data Type	Structured	Structured, Semi-Structured, Unstructured	Structured, Semi-Structured, Unstructured
Schema	Schema-on-Write	Schema-on-Read	Schema-on-Read and Schema-on-Write
Purpose	BI and Reporting	Data Science and Big Data Analytics	BI, Reporting, and Advanced Analytics
Performance	High for structured queries	Varies, but not optimized for queries	High for both structured and unstructured queries
Cost	Higher due to processing and storage	Lower per GB, but can increase with data growth	Moderate, balancing both data lakes and warehouses
Data Governance	Strong	Less defined	Combines strengths of both



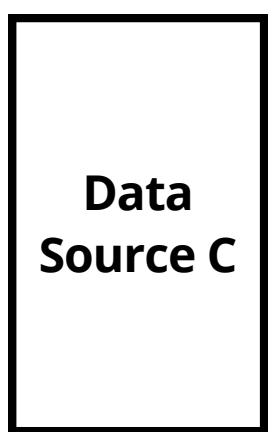
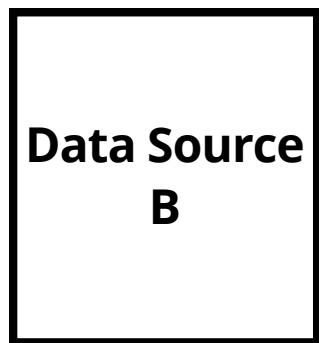
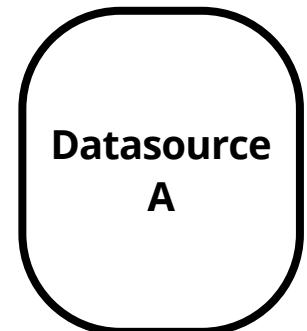
# Data Mesh



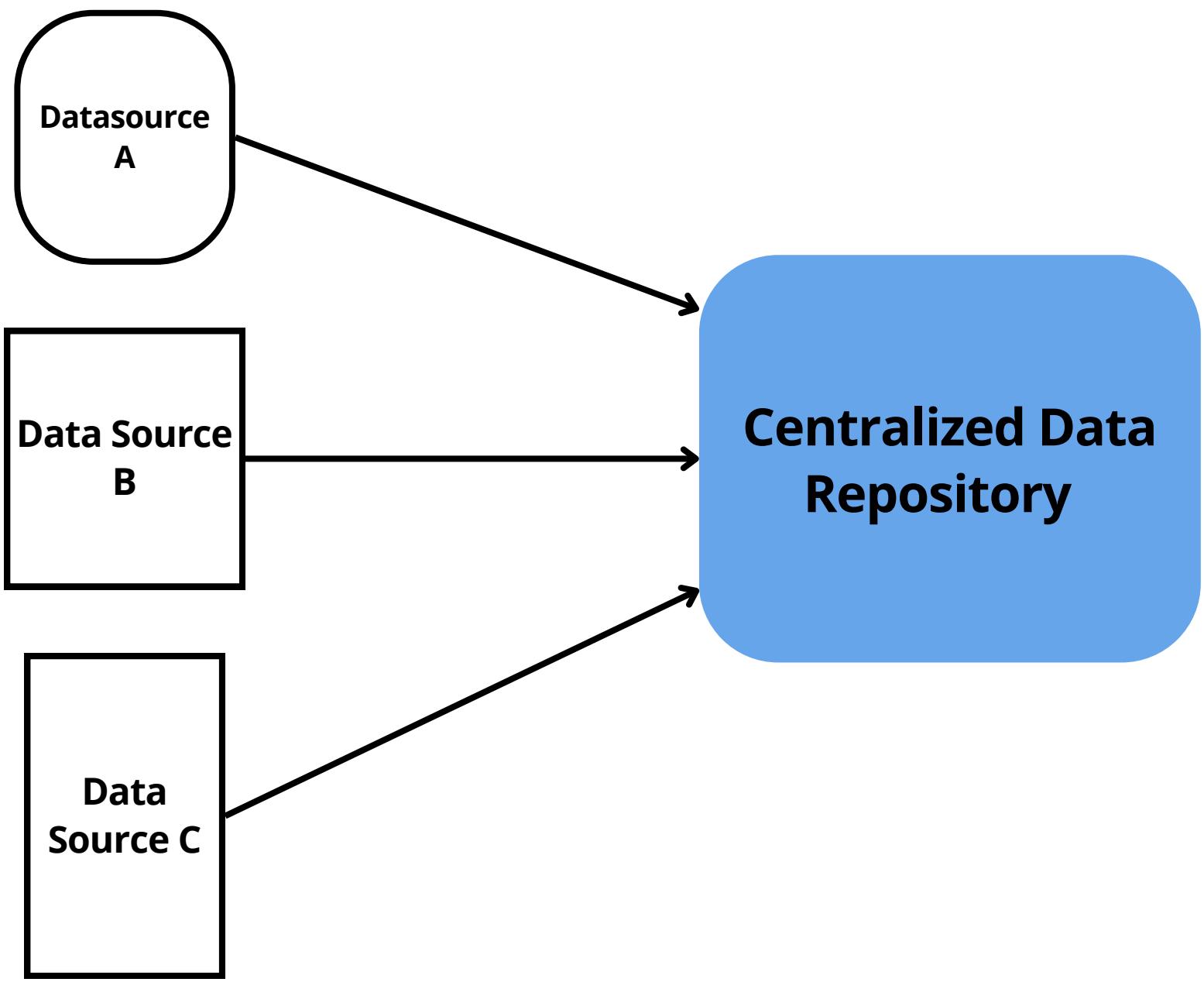
# Before Data Meshes



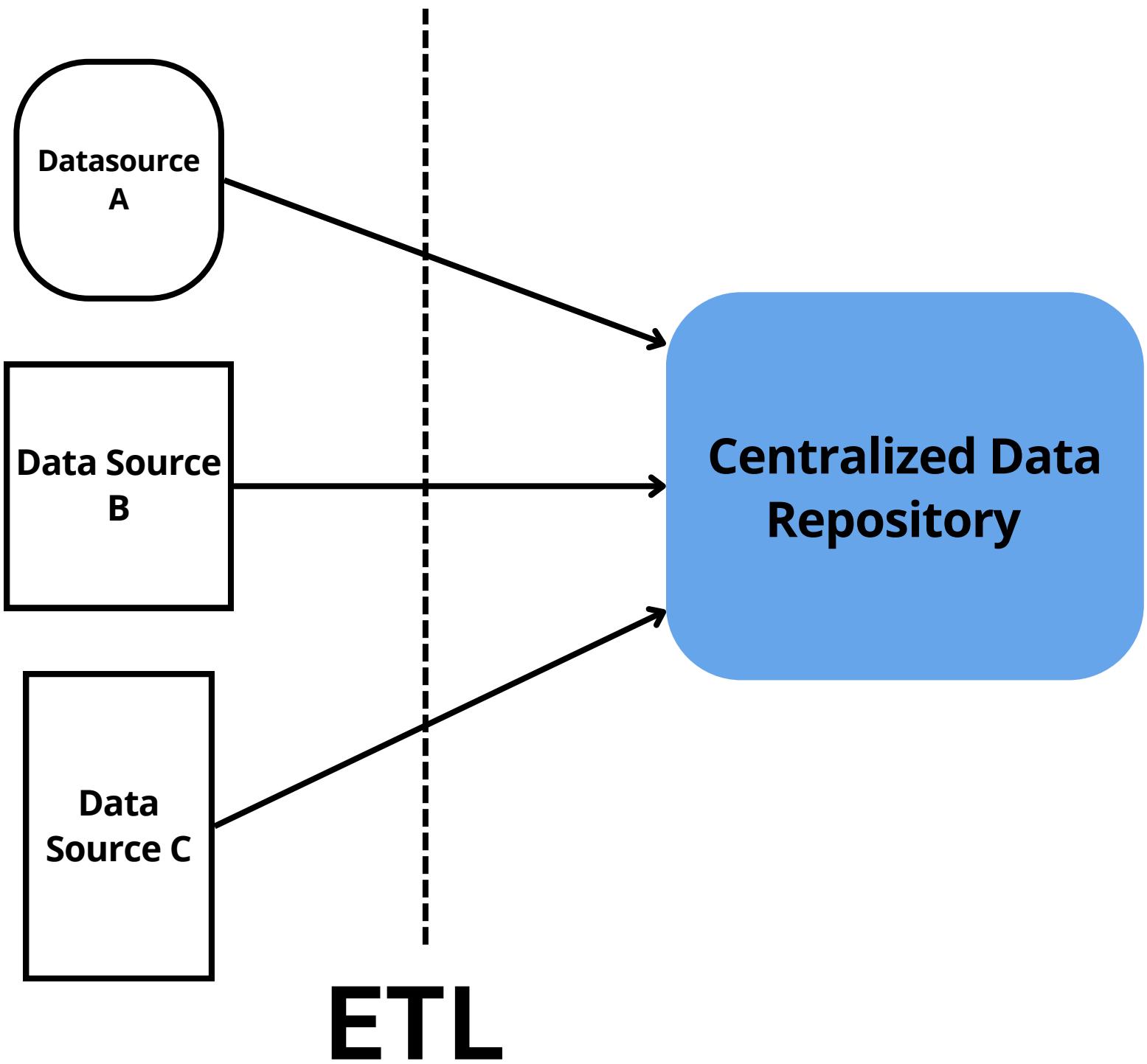
# Before Data Meshes



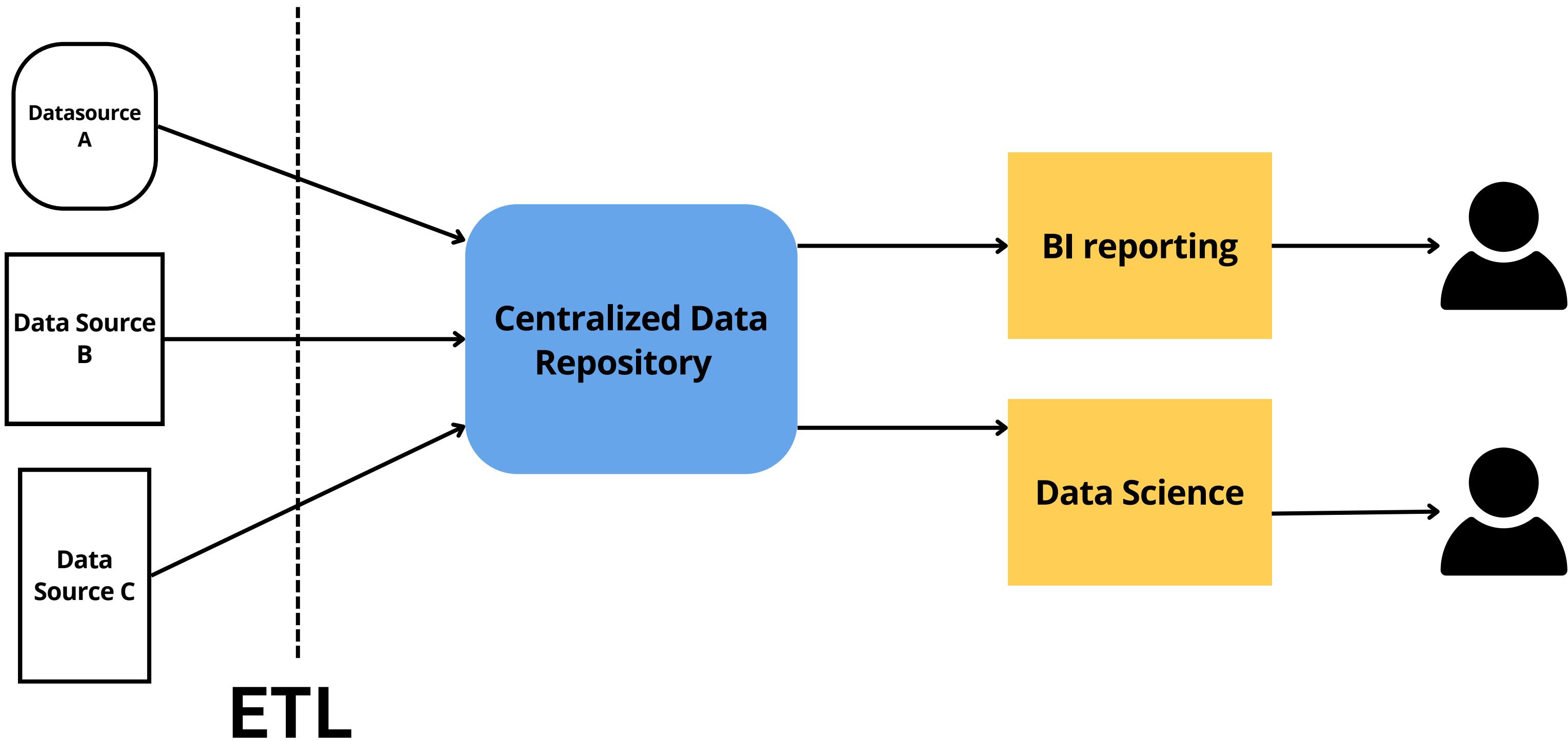
# Before Data Meshes



# Before Data Meshes



# Before Data Meshes



# Challenges

---

**Siloed data  
team**



# Challenges

---

Siloed data  
team

Slow  
responsiveness  
to change



# Challenges

---

Siloed data  
team

Slow  
responsiveness  
to change

Reduced  
Accuracy



# Data Mesh

---

# Data Mesh

---

**Is a decentralized approach to managing and accessing large-scale data in an organization.**

# Data Mesh

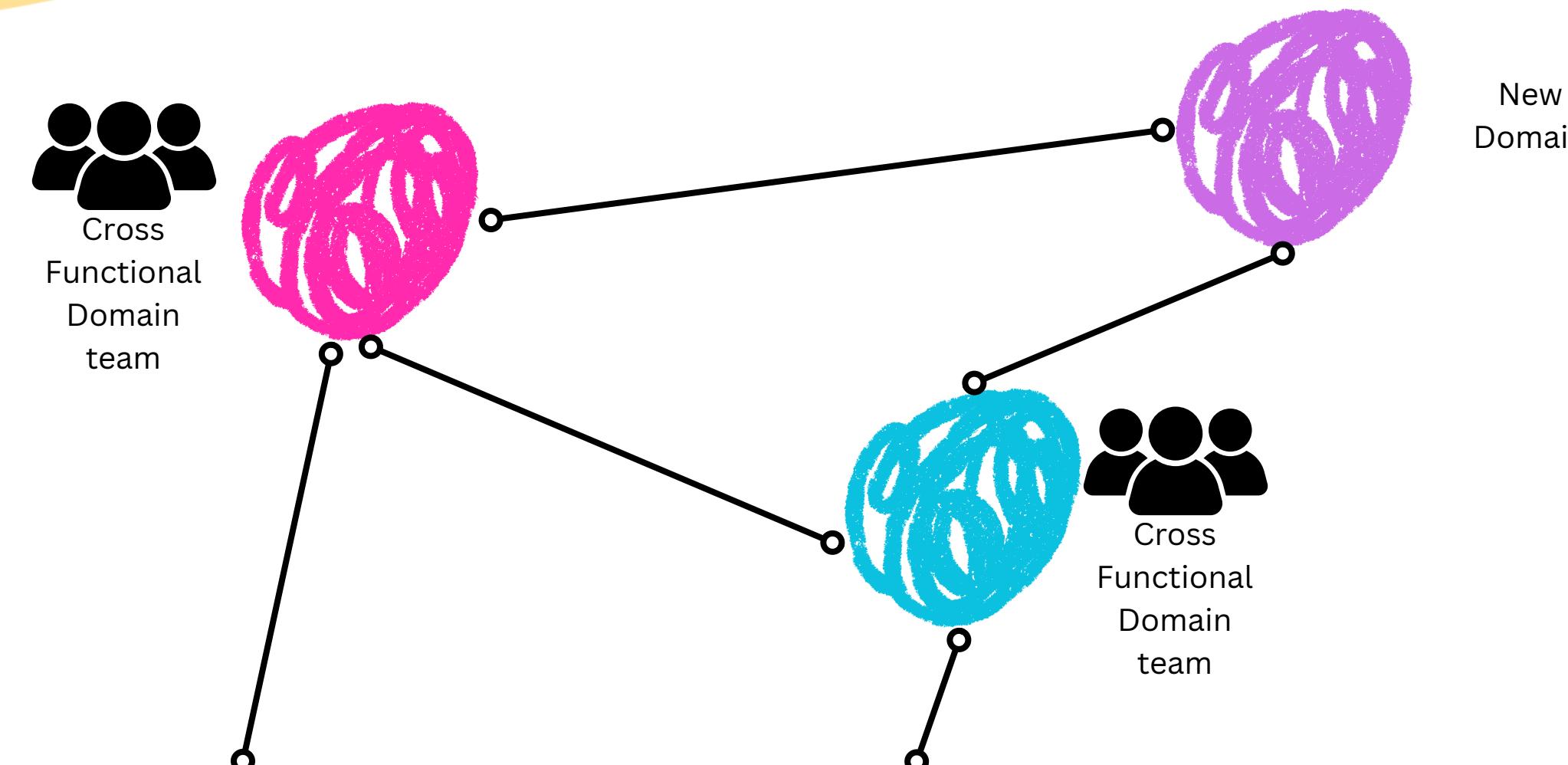
---

**Is a decentralized approach to managing and accessing large-scale data in an organization.**

**It aims to address some of the limitations of traditional centralized data architecture models by treating data as a product and fostering domain-oriented data ownership.**

# Global Governance

## Enables Interoperability



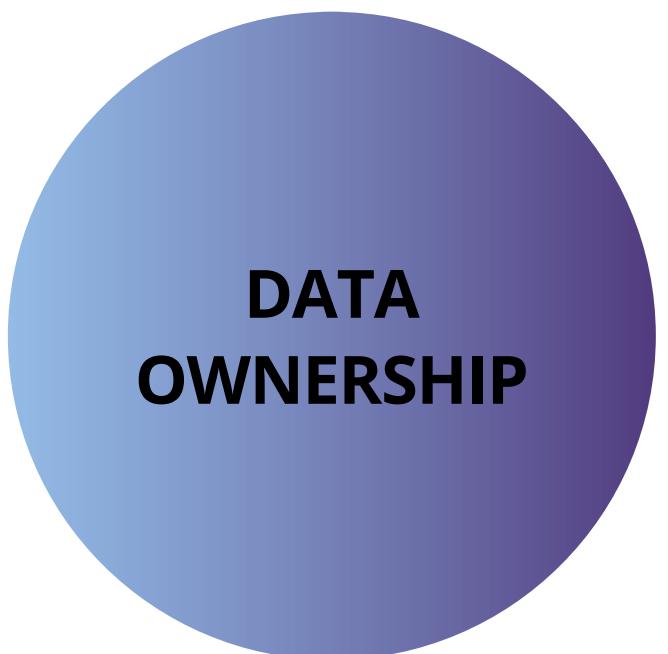
Data Infra As a Platform  
(Storage, Pipeline, Catalog, Access Control)



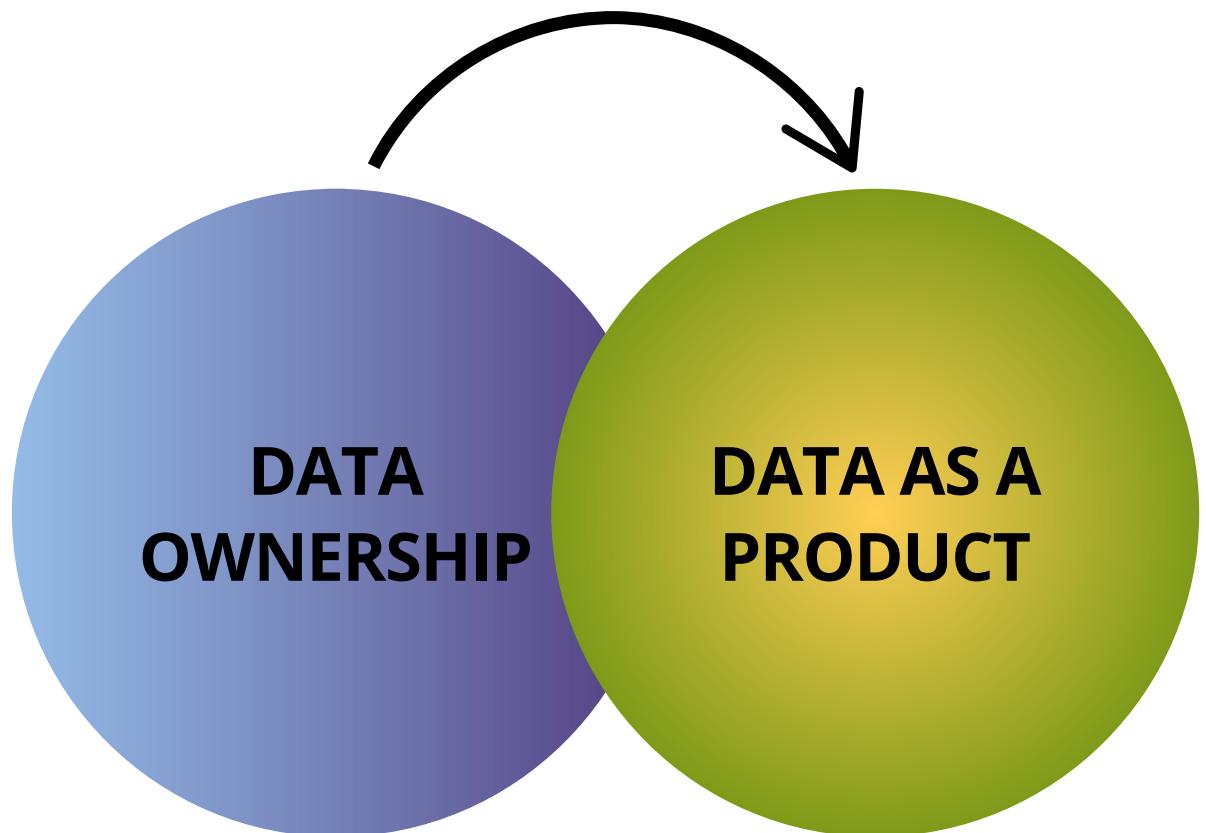
# Data Mesh Principles



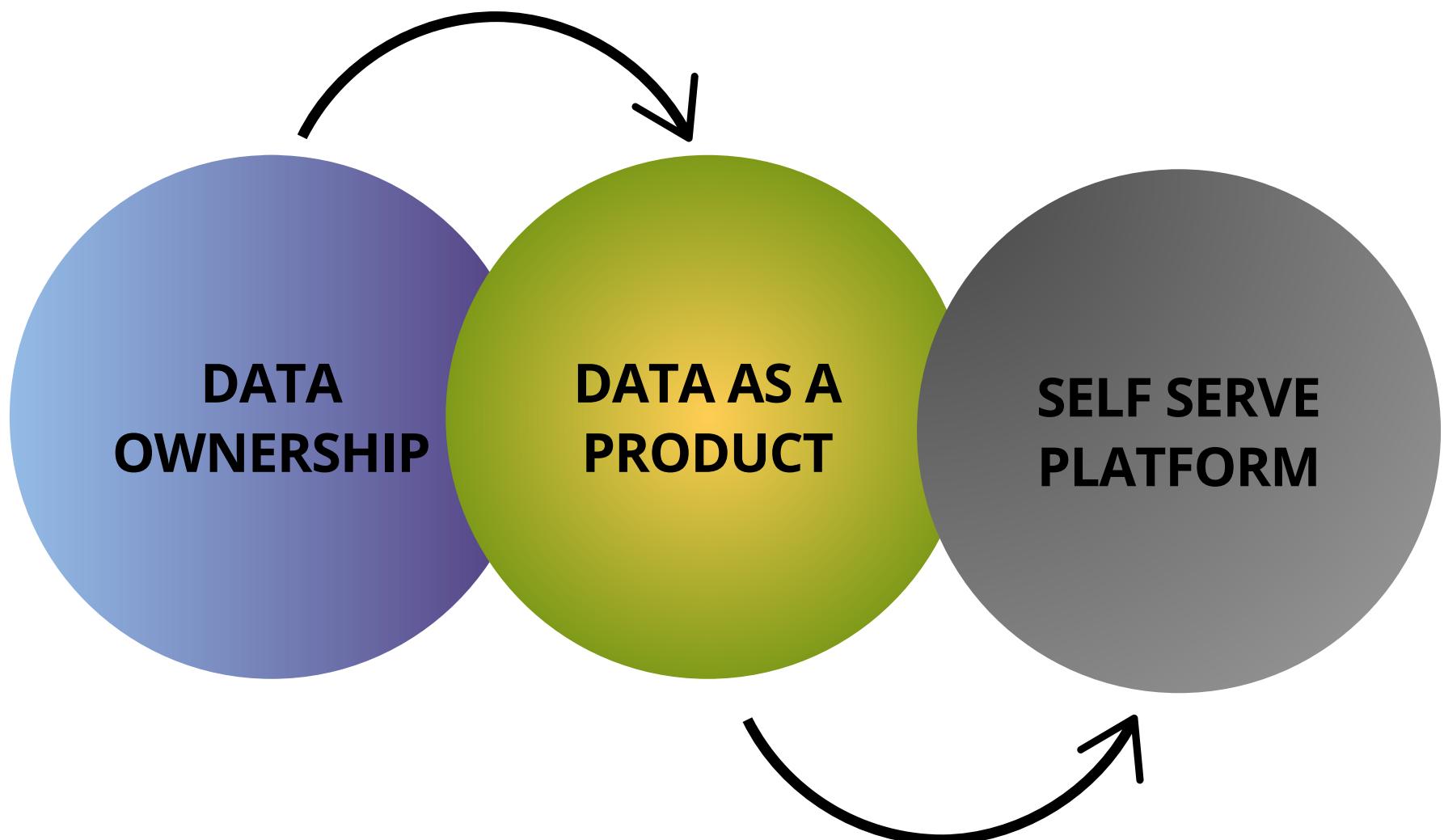
# Data Mesh Principles



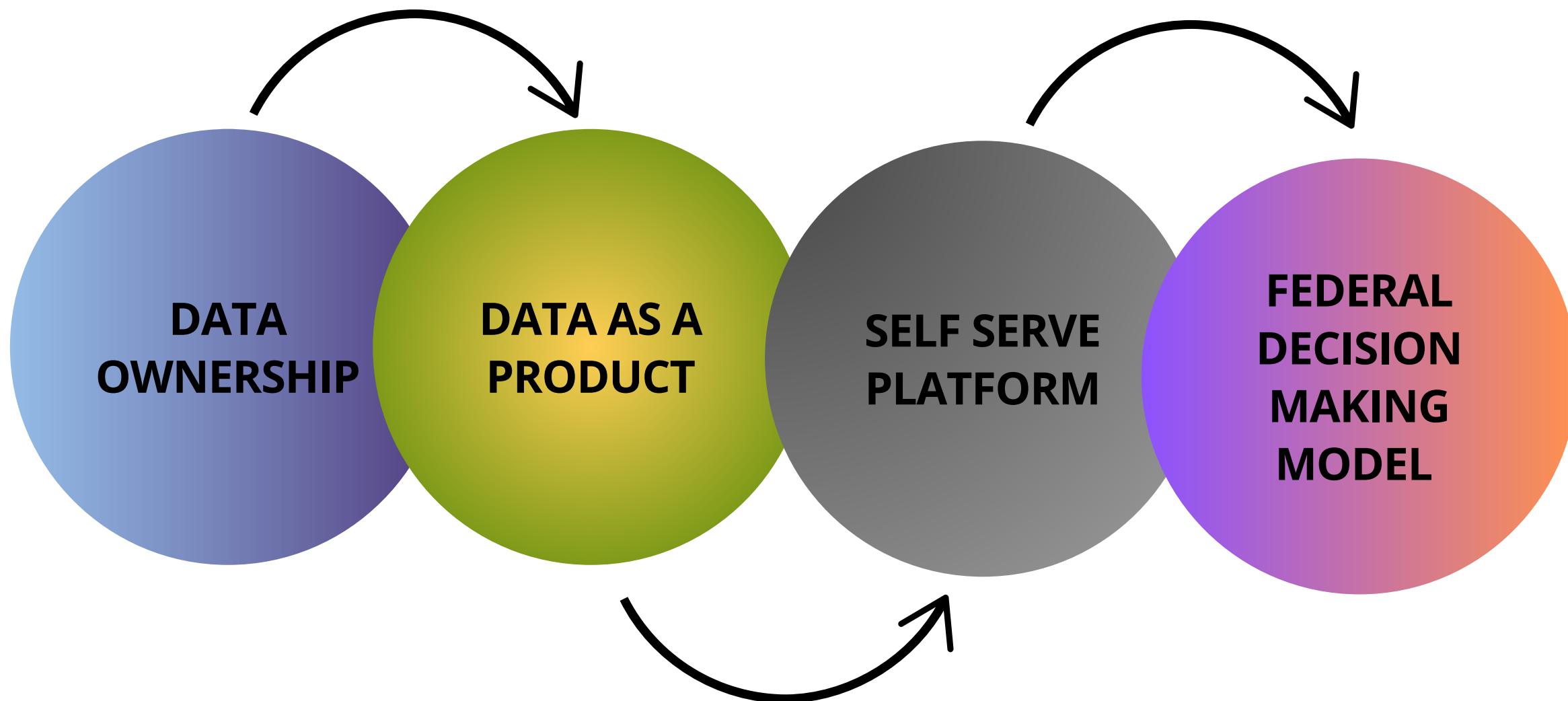
# Data Mesh Principles



# Data Mesh Principles



# Data Mesh Principles



# Orchestration



# Orchestration

---

# Orchestration

---

**Refers to the automated coordination  
and management of complex data  
workflows and processes.**

# Orchestration

---

**Refers to the automated coordination and management of complex data workflows and processes.**

**Streamline data pipelines by scheduling, monitoring, and managing dependencies**

# Examples



# **Examples**

# **Apache Airflow**



# Examples

# Apache Airflow

# AWS Step Functions



# Common Data Formats



**Define how data is structured, stored,  
and exchanged between systems.**

---



**Define how data is structured, stored,  
and exchanged between systems.**

---

**01** CSV (Comma-Separated  
Values)



# **Define how data is structured, stored, and exchanged between systems.**

---

**01** CSV (Comma-Separated  
Values)

**02** JSON (JavaScript Object  
Notation)



# **Define how data is structured, stored, and exchanged between systems.**

---

**01** CSV (Comma-Separated Values)

**02** JSON (JavaScript Object Notation)

**03** XML (eXtensible Markup Language)



# **Define how data is structured, stored, and exchanged between systems.**

---

**01** CSV (Comma-Separated Values)

**02** JSON (JavaScript Object Notation)

**03** XML (eXtensible Markup Language)

**04** Apache Parquet



# Common Data Sources



**Data sources are the origins from which  
data is collected or obtained for  
analysis, processing, and storage.**

---



**Data sources are the origins from which  
data is collected or obtained for  
analysis, processing, and storage.**

---

## 01 Databases



**Data sources are the origins from which  
data is collected or obtained for  
analysis, processing, and storage.**

---

**01** Databases

**02** Data Warehouses



**Data sources are the origins from which  
data is collected or obtained for  
analysis, processing, and storage.**

---

**01** Databases

**02** Data Warehouses

**03** Data Lakes



**Data sources are the origins from which data is collected or obtained for analysis, processing, and storage.**

---

**01** Databases

**02** Data Warehouses

**03** Data Lakes

**04** Flat Files



# Data Modeling



**Data modeling is the process of creating a visual representation of a complex system's data structure and organization.**

---



**Data modeling is the process of creating a visual representation of a complex system's data structure and organization.**

---

## 01 Conceptual Data Model



**Data modeling is the process of creating a visual representation of a complex system's data structure and organization.**

---

**01** Conceptual Data Model

**02** Logical Data Model



**Data modeling is the process of creating a visual representation of a complex system's data structure and organization.**

---

**01** Conceptual Data Model

**02** Logical Data Model

**03** Physical Data Model



# Data Lineage



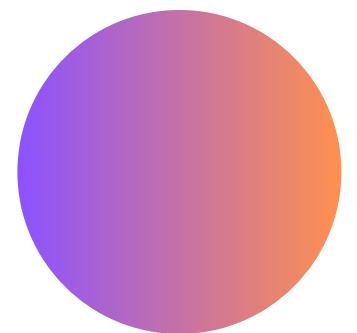
**Involves tracking the flow of data  
through an organization's systems and  
processes.**

---



**Involves tracking the flow of data through an organization's systems and processes.**

---

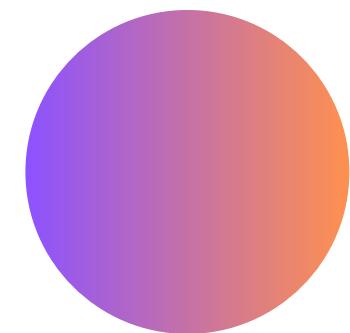


Datasource A

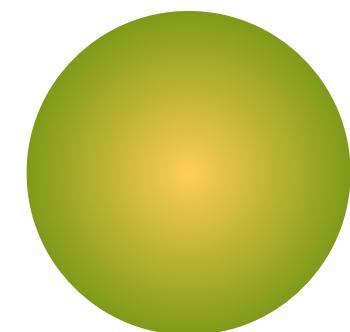


**Involves tracking the flow of data through an organization's systems and processes.**

---



Datasource A

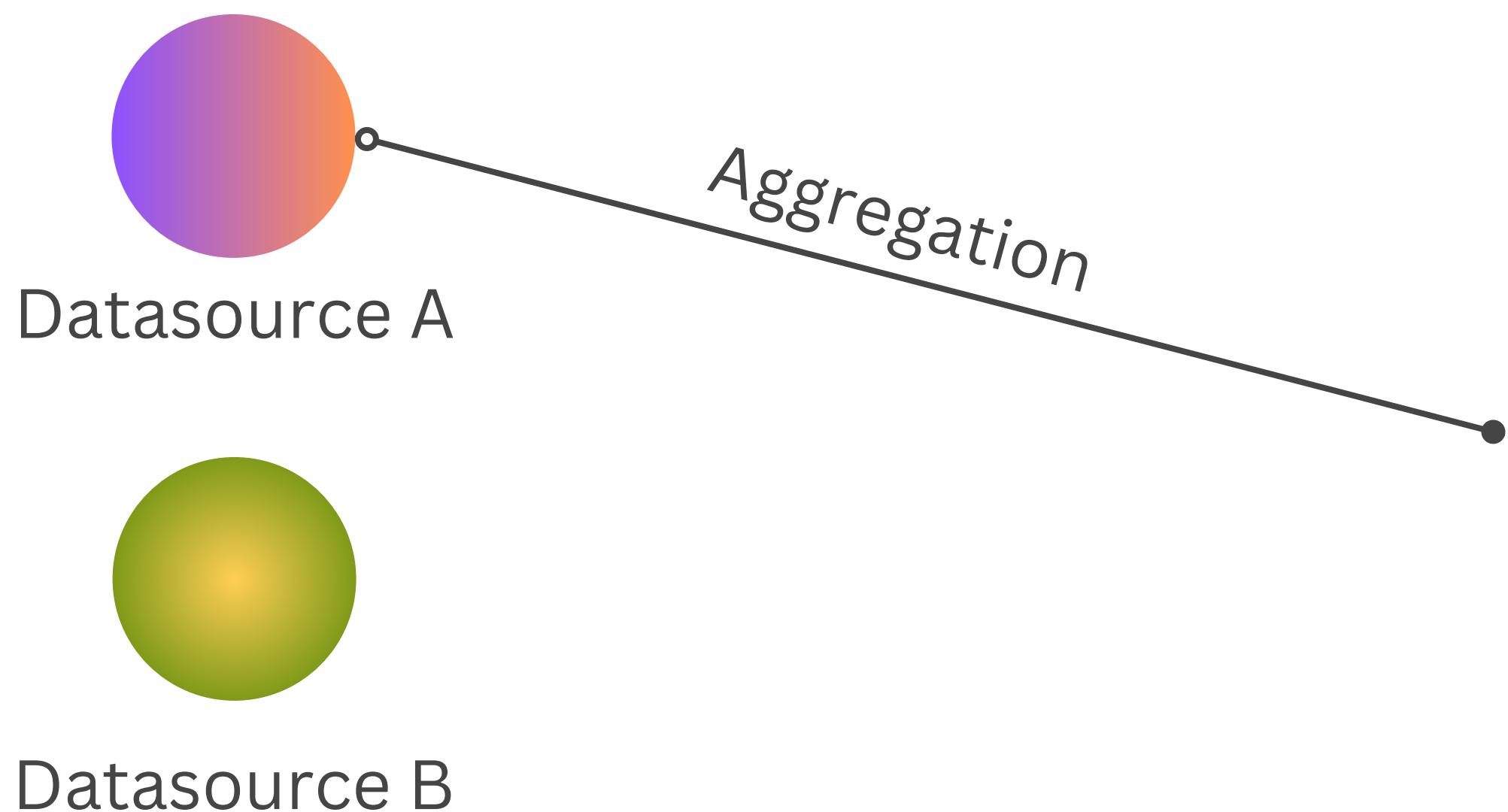


Datasource B



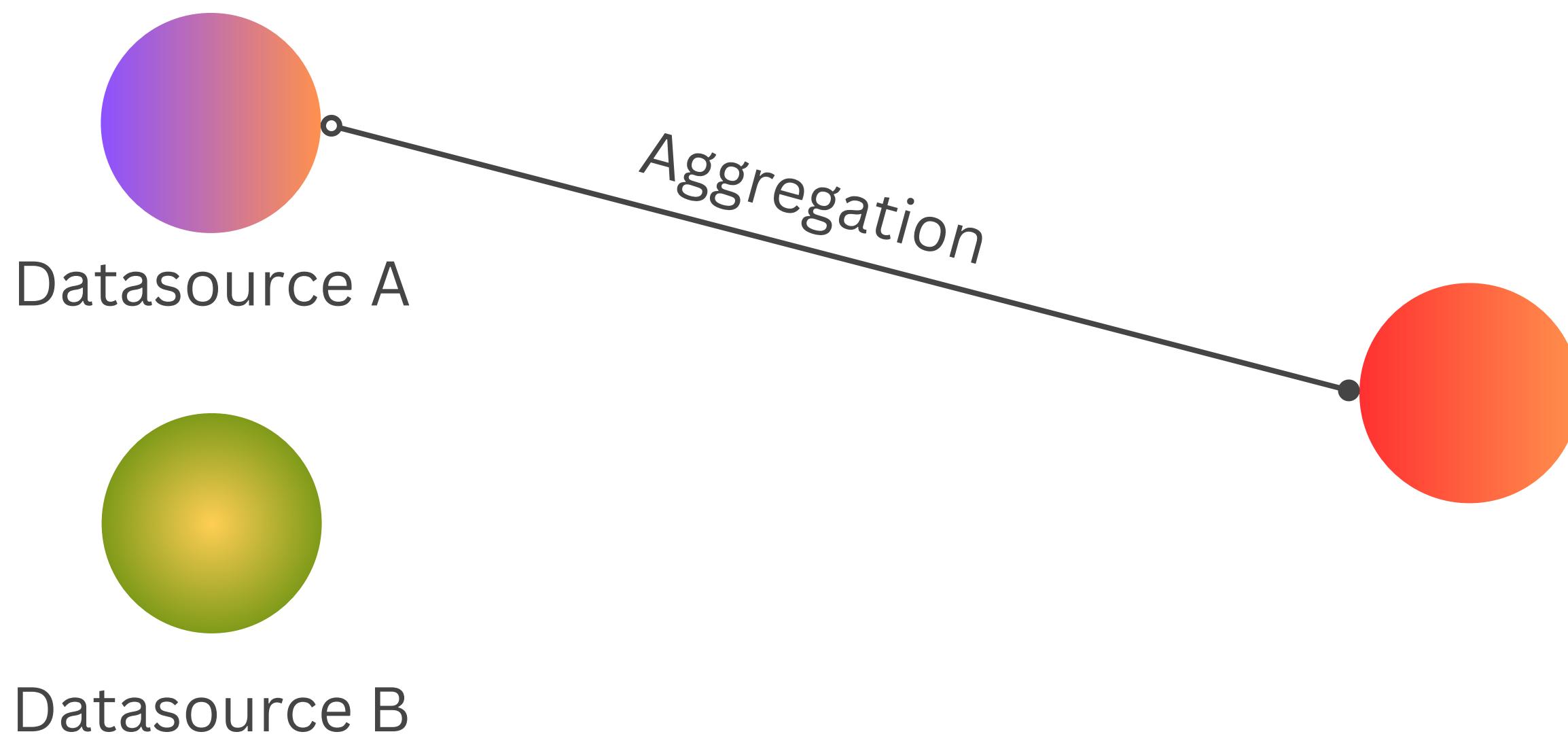
**Involves tracking the flow of data through an organization's systems and processes.**

---



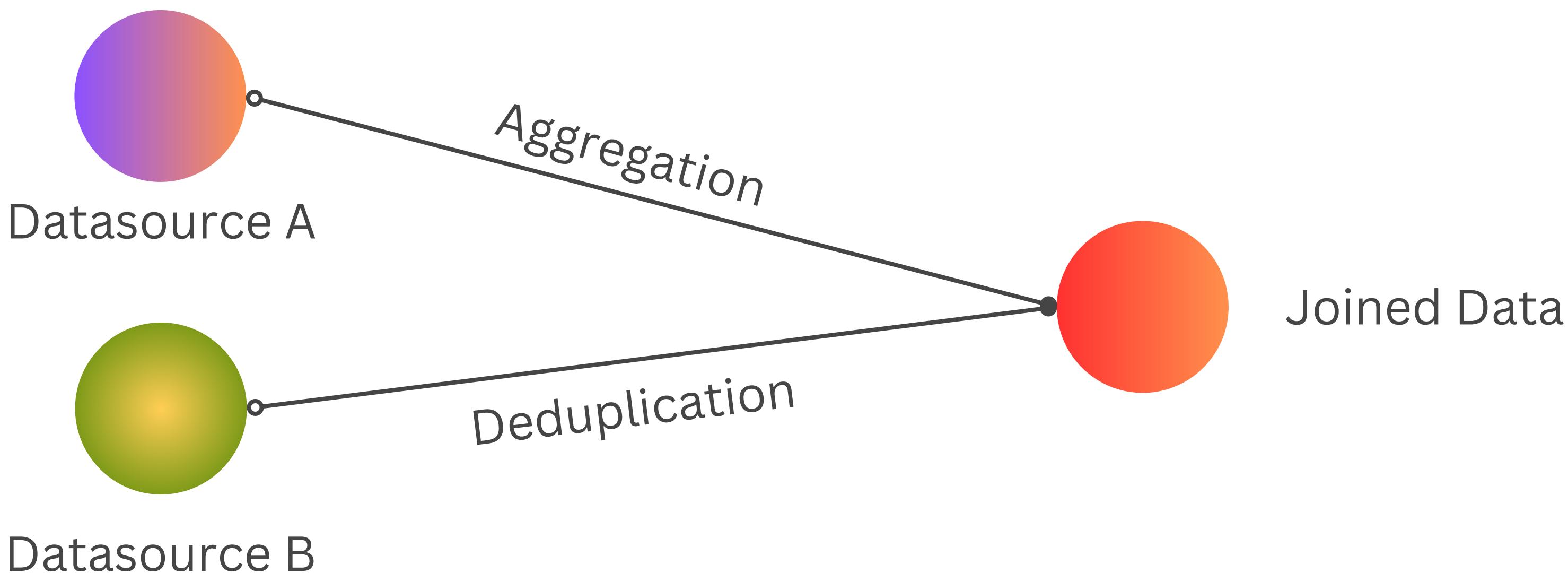
**Involves tracking the flow of data through an organization's systems and processes.**

---



**Involves tracking the flow of data through an organization's systems and processes.**

---



# Schema Evolution



**Process of managing changes to the schema of a database or data structure over time**

---



# **Process of managing changes to the schema of a database or data structure over time**

---

**Firstname**

**Lastname**

---

John

Doe

Jane

Done



# **Process of managing changes to the schema of a database or data structure over time**

---

<b>Firstname</b>	<b>Lastname</b>	
John	Doe	
Jane	Done	
<b>Add Fullname</b>		



# Process of managing changes to the schema of a database or data structure over time

---

Firstname	Lastname
John	Doe
Jane	Done



Add Fullname

Firstname	Lastname	Fullscreen
John	Doe	John Doe
Jane	Done	Jane Doe



# Data Sampling



**A statistical technique used to select a subset of data from a larger dataset**

---



**A statistical technique used to select a subset of data from a larger dataset**

---

## **01 Population and Sample**



**A statistical technique used to select a subset of data from a larger dataset**

---

**01** Population and Sample

**02** Sampling Frame



**A statistical technique used to select a subset of data from a larger dataset**

---

**01** Population and Sample

**02** Sampling Frame

**03** Sampling Error



**A statistical technique used to select a subset of data from a larger dataset**

---

**01** Population and Sample

**02** Sampling Frame

**03** Sampling Error

**04** Sample Size



# Data Skewness



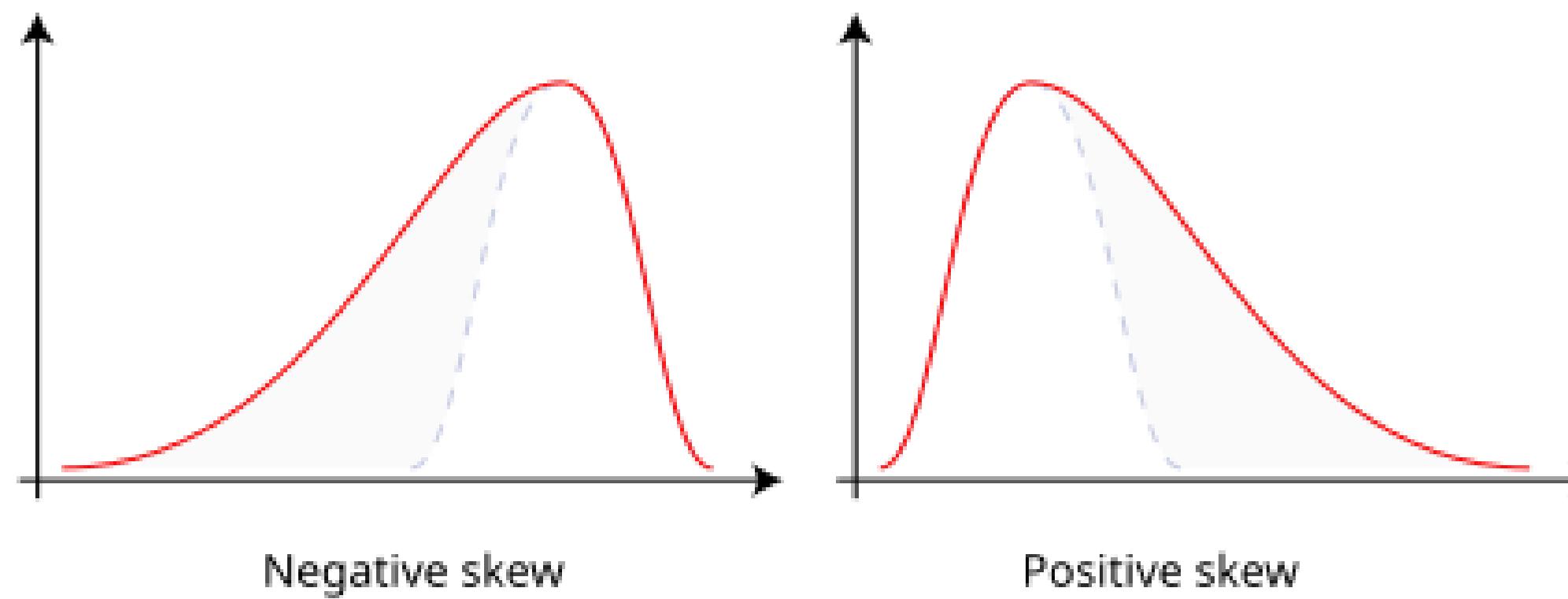
**Refers to the asymmetry in the distribution of data values around the mean**

---



**Refers to the asymmetry in the distribution of data values around the mean**

---



<https://en.wikipedia.org/wiki/Skewness>



# Data Validation and Profiling



# Data Validation

To ensure that data is accurate, consistent, and meets predefined criteria or rules.

# Data Profiling

To analyze and understand the characteristics, structure, and quality of data within a dataset.



# A Word On SQL



The  
certification  
will expect you  
to know SQL



# AWS Glue



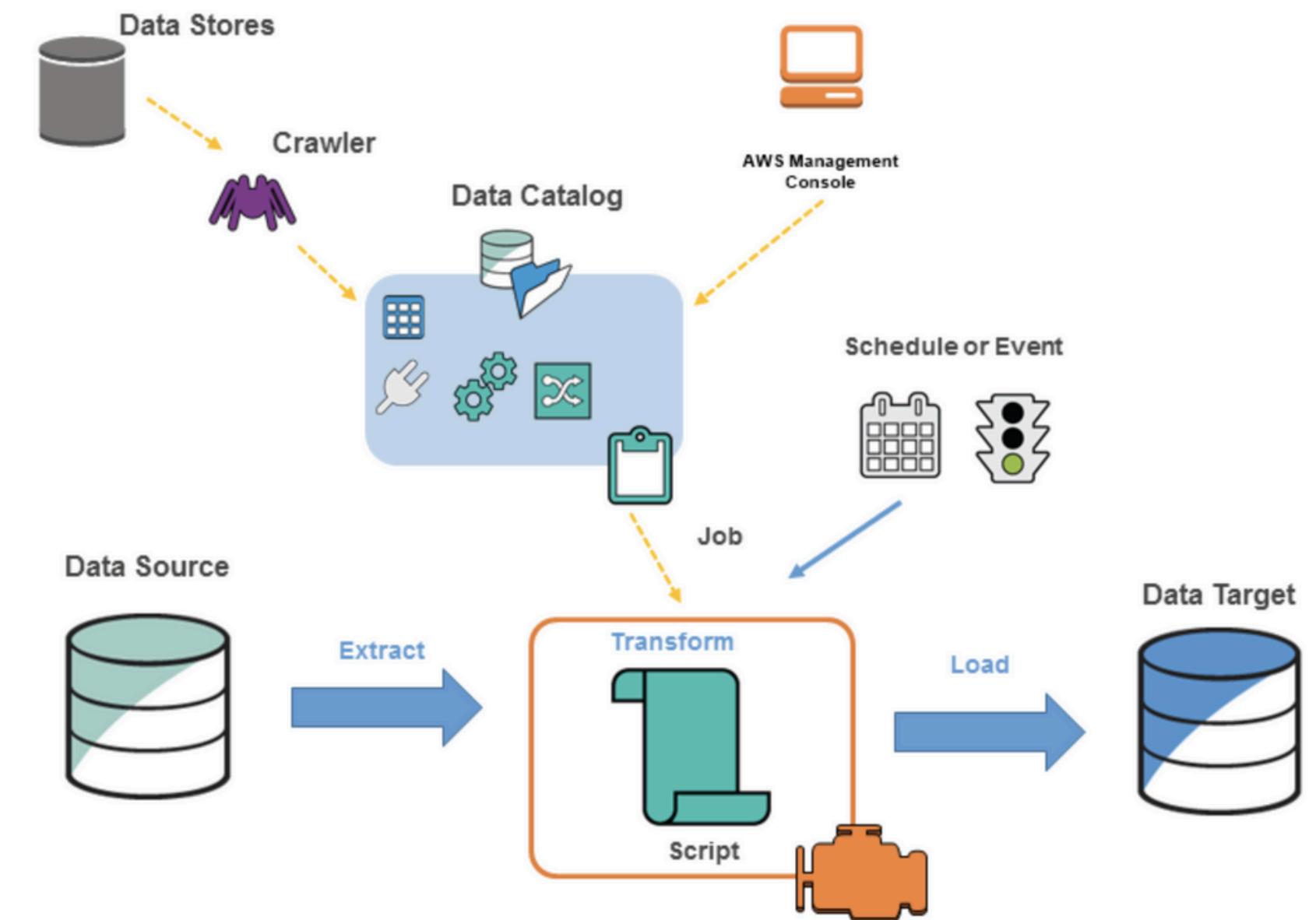
# What is AWS Glue?

Fully Managed ETL Service

A Spark ETL Engine

Consists of a Central Metadata Repository - Glue Data Catalog

Flexible Scheduler



# Why use AWS Glue?

AWS Glue offers a fully managed serverless ETL Tool. This removes the overhead, and barriers to entry, when there is a requirement for a ETL service in AWS.



# AWS Glue - Setup Work



# AWS Glue Data Catalog



# Glue Data Catalog

## Persistent Metadata Store

It is a managed service that lets you store, annotate, and share metadata which can be used to query and transform data

One AWS Glue Data Catalog per AWS region

Identity and Access Management (IAM) policies control access

Can be used for data governance

Data Location

Schema

Data Types

Data Classification

Examples of Meta Data



# AWS Glue Databases



# AWS Glue Databases

**A set of associated Data Catalog table definitions organized into a logical group.**

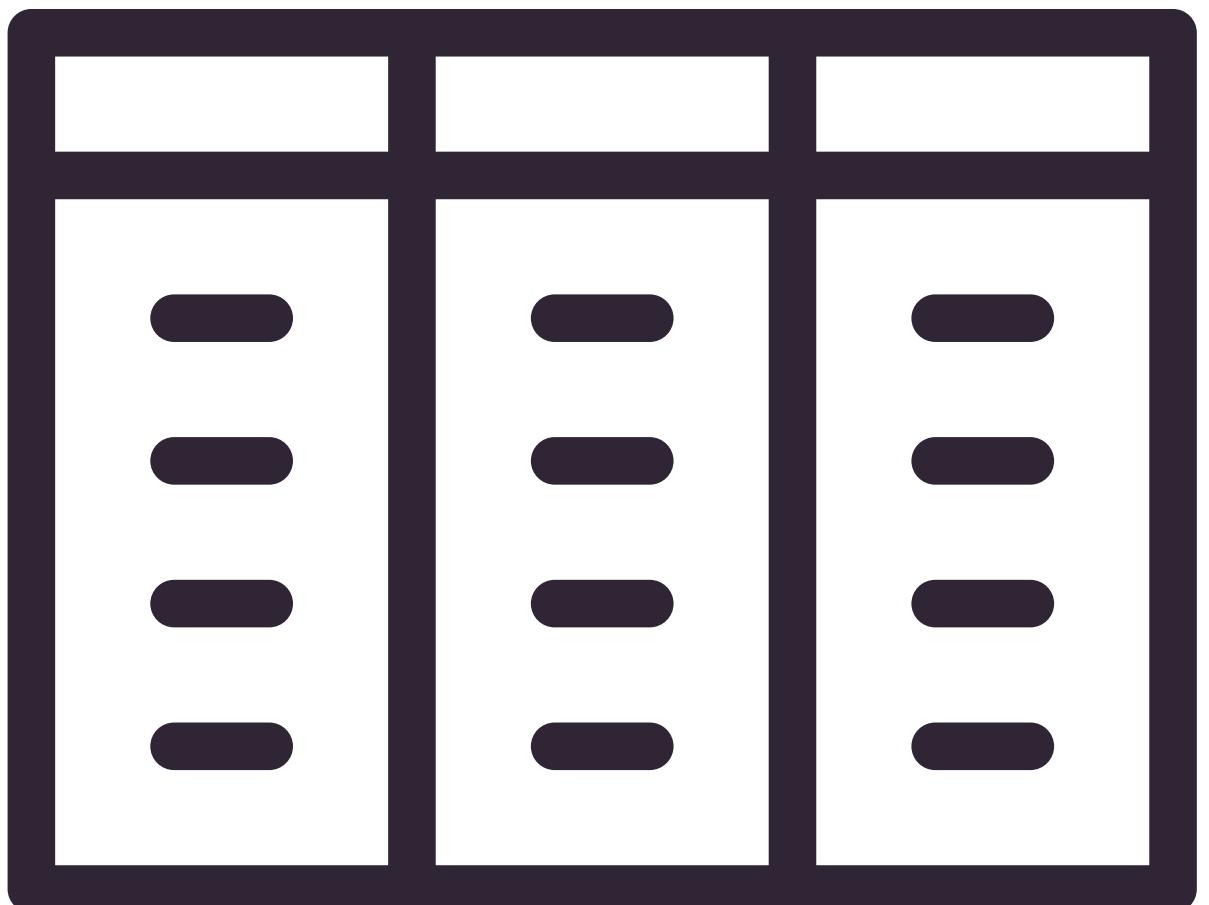


# AWS Glue Tables



# AWS Glue Tables

The metadata definition that represents your data. The data resides in its original store. This is just a representation of the schema.

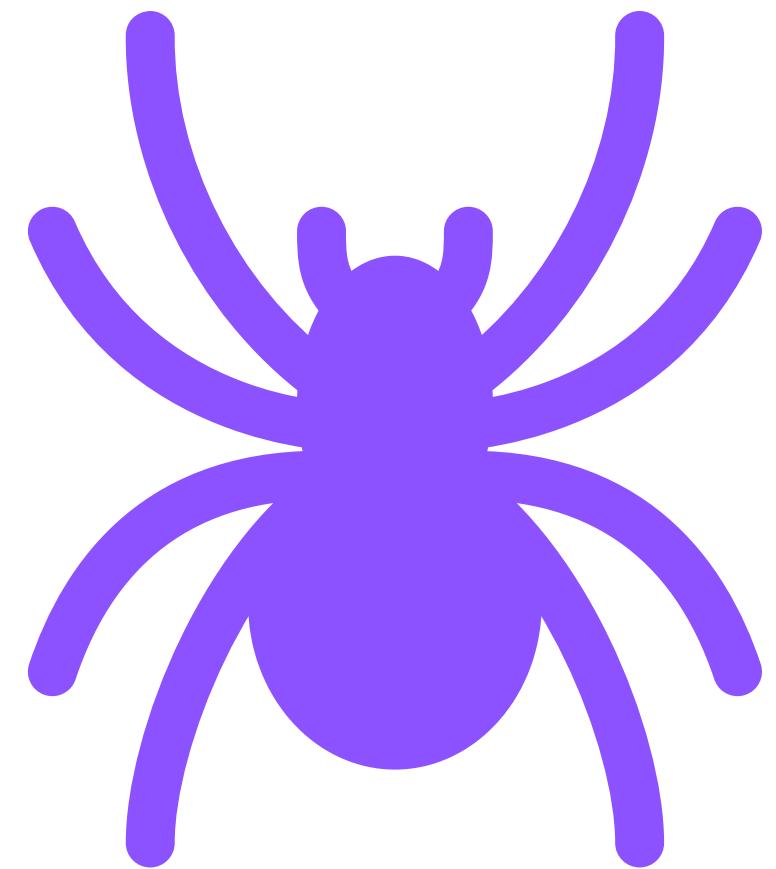


# AWS GLUE CRAWLERS



# AWS Glue Crawler

**A program that connects to a data store (source or target), progresses through a prioritized list of classifiers to determine the schema for your data, and then creates metadata tables in the AWS Glue Data Catalog.**

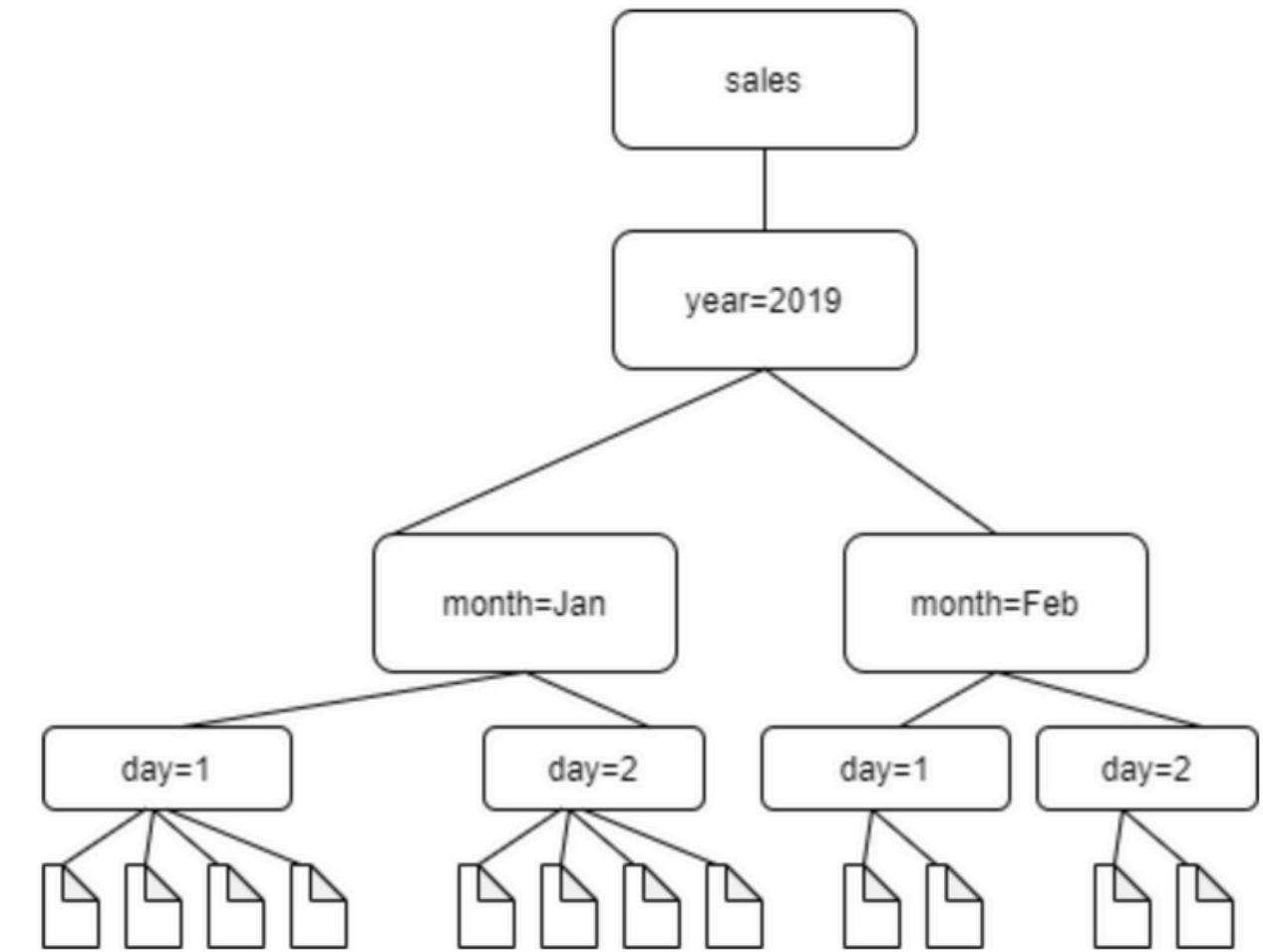


# PARTITIONS IN AWS



# PARTITIONS

**Folders where data is stored on S3, which are physical entities, are mapped to partitions, which are logical entities i.e. Columns in the Glue table.**



`s3://sales/year=2019/month=Jan/day=1`  
`s3://sales/year=2019/month=Jan/day=2`  
`s3://sales/year=2019/month=Feb/day=1`  
`s3://sales/year=2019/month=Feb/day=2`

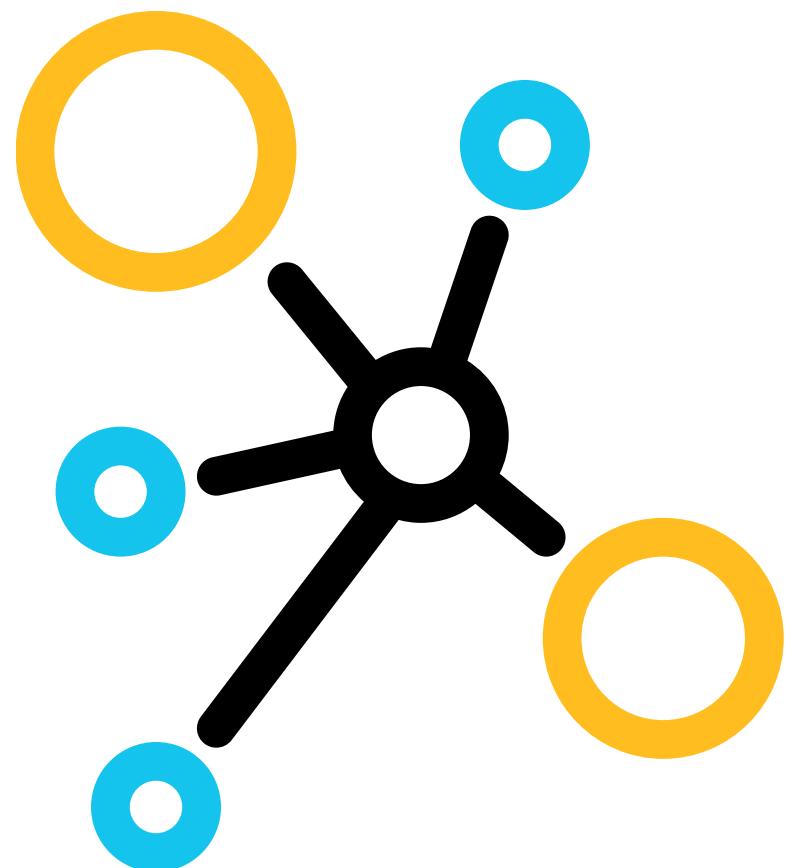


# AWS Glue Connections



# AWS Glue Connections

A Data Catalog object that contains the properties that are required to connect to a particular data store.



# AWS Glue ETL



# AWS Glue ETL

**AWS Glue ETL supports extracting data from various sources, transforming it to meet your business needs, and loading it into a destination of your choice.**



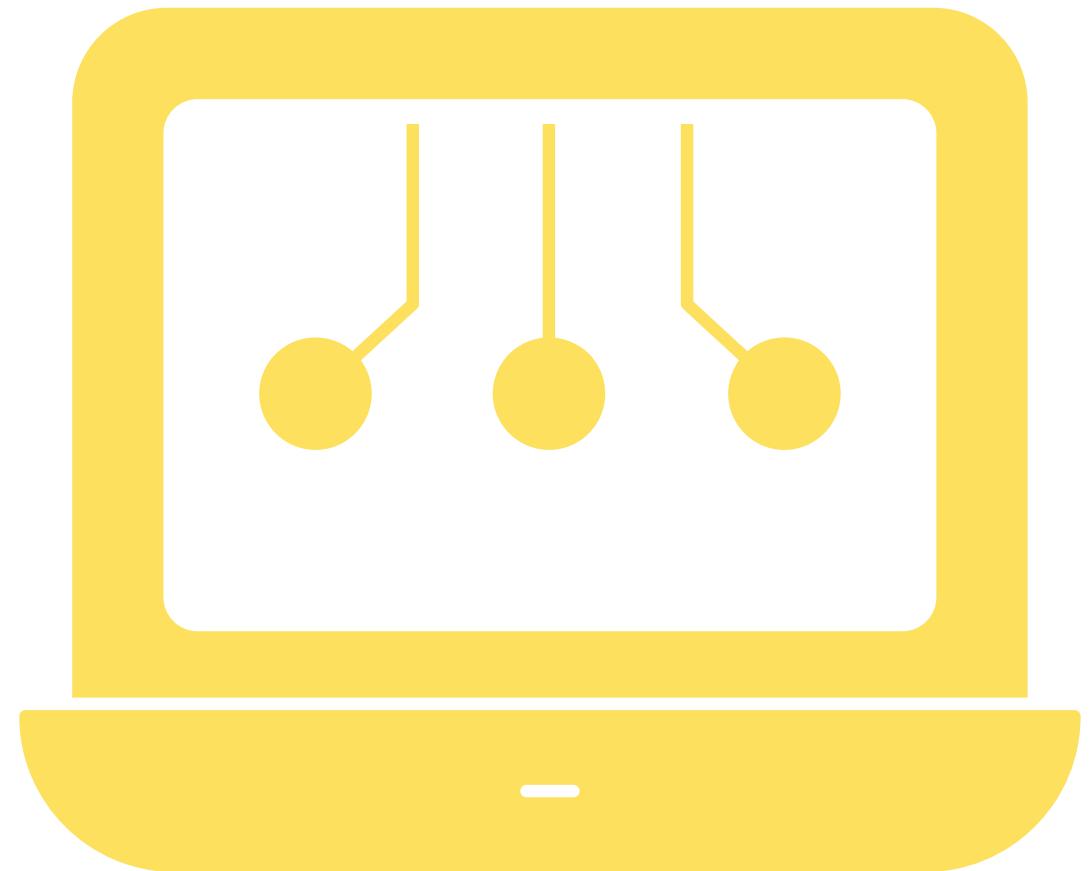
# AWS Glue ETL Engine

Apache Spark engine to distribute  
big data workloads across worker  
nodes



# AWS Glue DPUs

**1 DPU is equivalent to 4 vCPUs and  
16 GB memory.**



# AWS Glue Bookmarks

**Tracks data that has already been processed during a previous run of an ETL job by persisting state information from the job run**

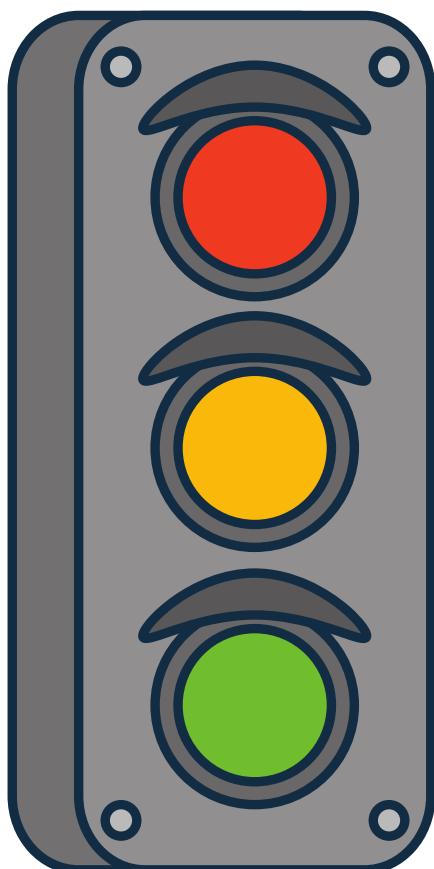
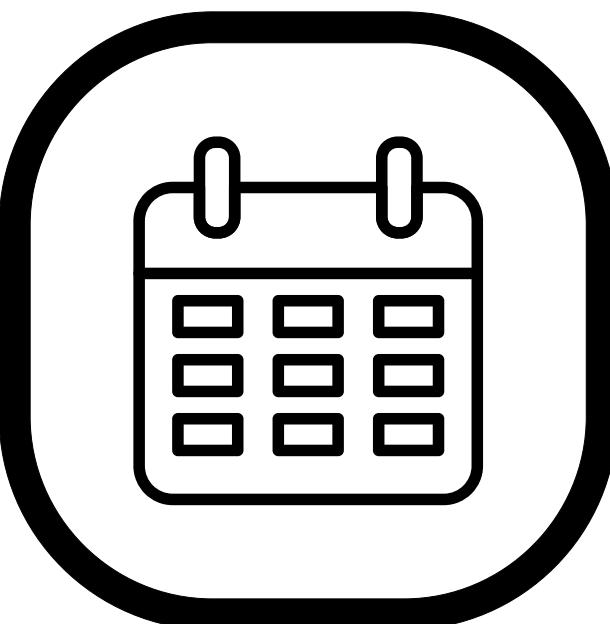


# AWS Glue Scheduling



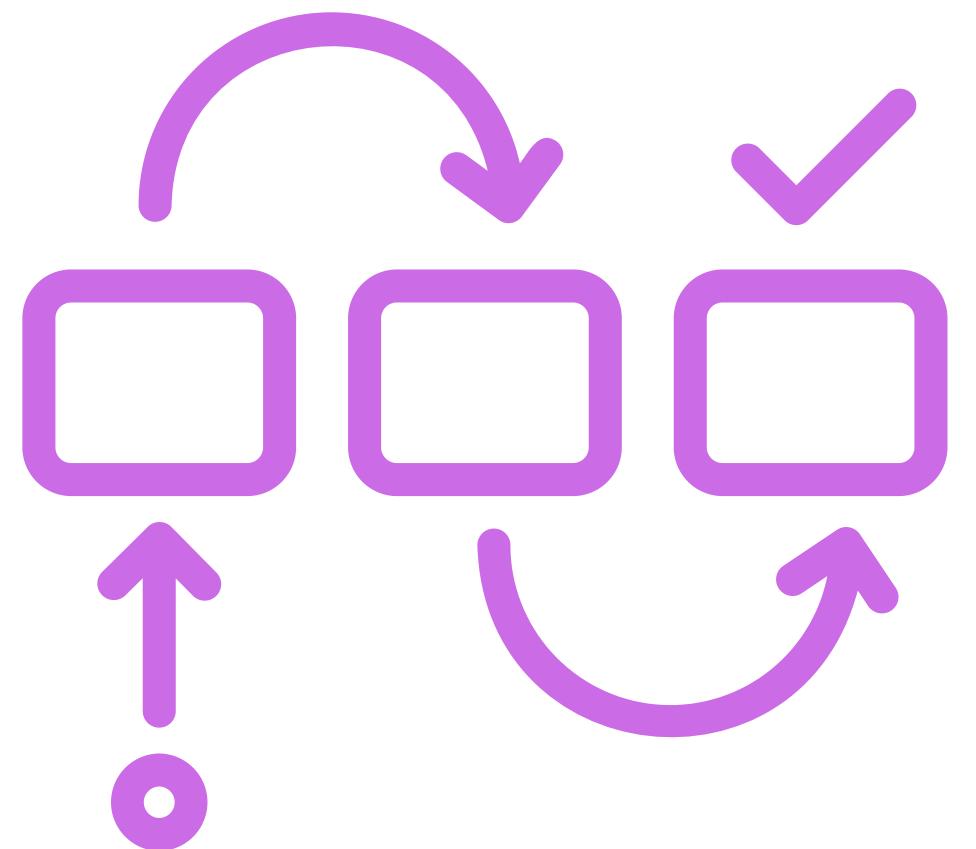
# AWS Glue Triggers

Initiates an ETL job. Triggers can be defined based on a scheduled time or an event.



# AWS Glue Workflow

Create and visualize complex extract, transform, and load (ETL) activities involving multiple crawlers, jobs, and triggers



# AWS Glue Scheduling (Others)

Apache Airflow

AWS Step Functions

Amazon Event Bridge



# AWS Glue Data Quality



# AWS GLUE DATA Quality

**Monitor the quality of your data  
by Data Quality Definition  
Language (DQDL) using DeeQu**



# AWS Glue Data Brew



# AWS GLUE DATA Brew

**Visual data preparation tool that  
makes it easier for data analysts  
and data scientists to clean and  
normalize data**



# Amazon Athena



# What is Athena?

---



# What is Athena?

---

01

**Interactive query service that makes it easy to analyze data in Amazon S3 using standard SQL**



# What is Athena?

---

01

**Interactive query service that makes it easy to analyze data in Amazon S3 using standard SQL**

02

**Serverless - Pay as You Go**



# What is Athena?

---

01

**Interactive query service that makes it easy to analyze data in Amazon S3 using standard SQL**

02

**Serverless - Pay as You Go**

03

**Presto/Trino to run SQL Queries**



# What is Athena?

---

**01**

**Interactive query service that makes it easy to analyze data in Amazon S3 using standard SQL**

**03**

**Presto/Trino to run SQL Queries**

**02**

**Serverless - Pay as You Go**

**04**

**Integrated with AWS Glue Data Catalog**



# What is Presto/Trino?

---



# What is Presto/Trino?

---

**01** Developed By Facebook



# What is Presto/Trino?

---

01

Developed By Facebook

02

Distributed system that  
runs on Hadoop



# What is Presto/Trino?

---

**01**

**Developed By Facebook**

**02**

**Distributed system that  
runs on Hadoop**

**03**

**One coordinator node working in  
synch with multiple worker nodes**



# What is Presto/Trino?

---

**01**

**Developed By Facebook**

**03**

**One coordinator node working in  
synch with multiple worker nodes**

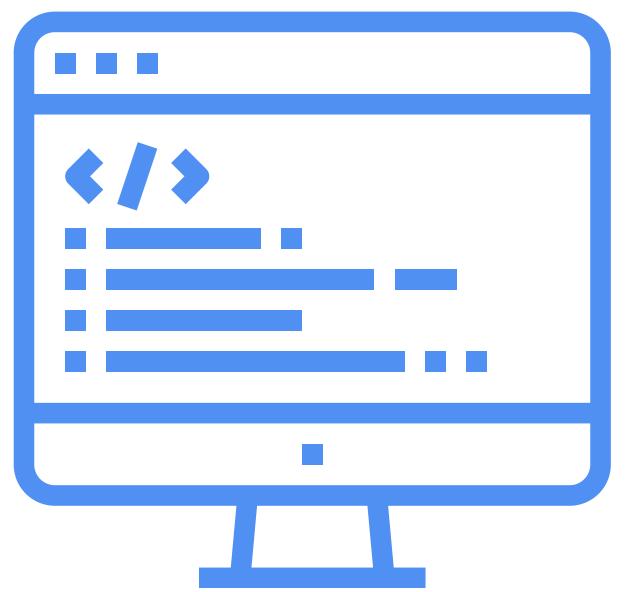
**02**

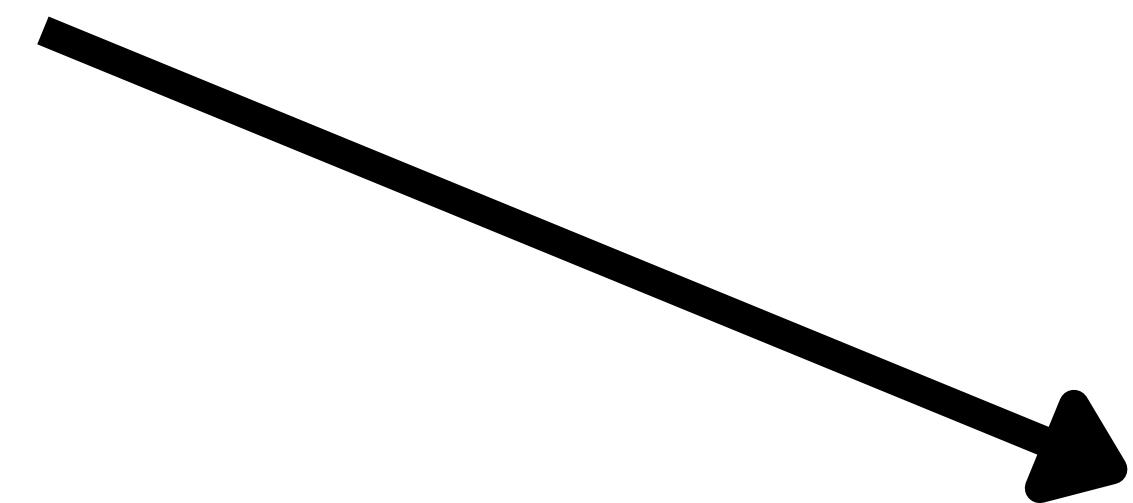
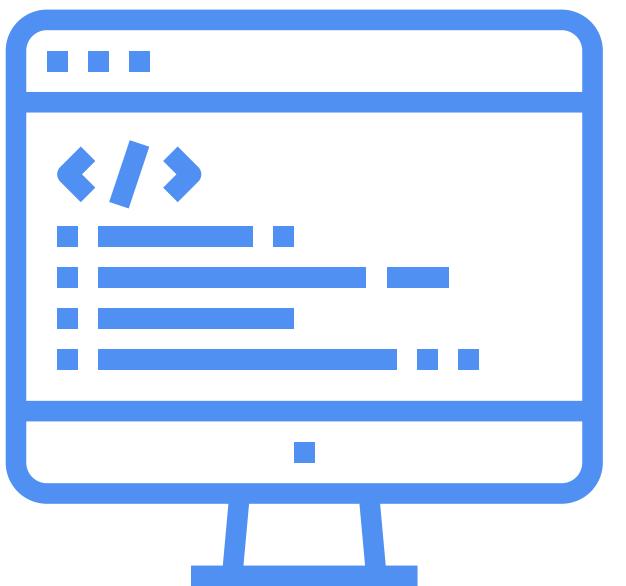
**Distributed system that  
runs on Hadoop**

**04**

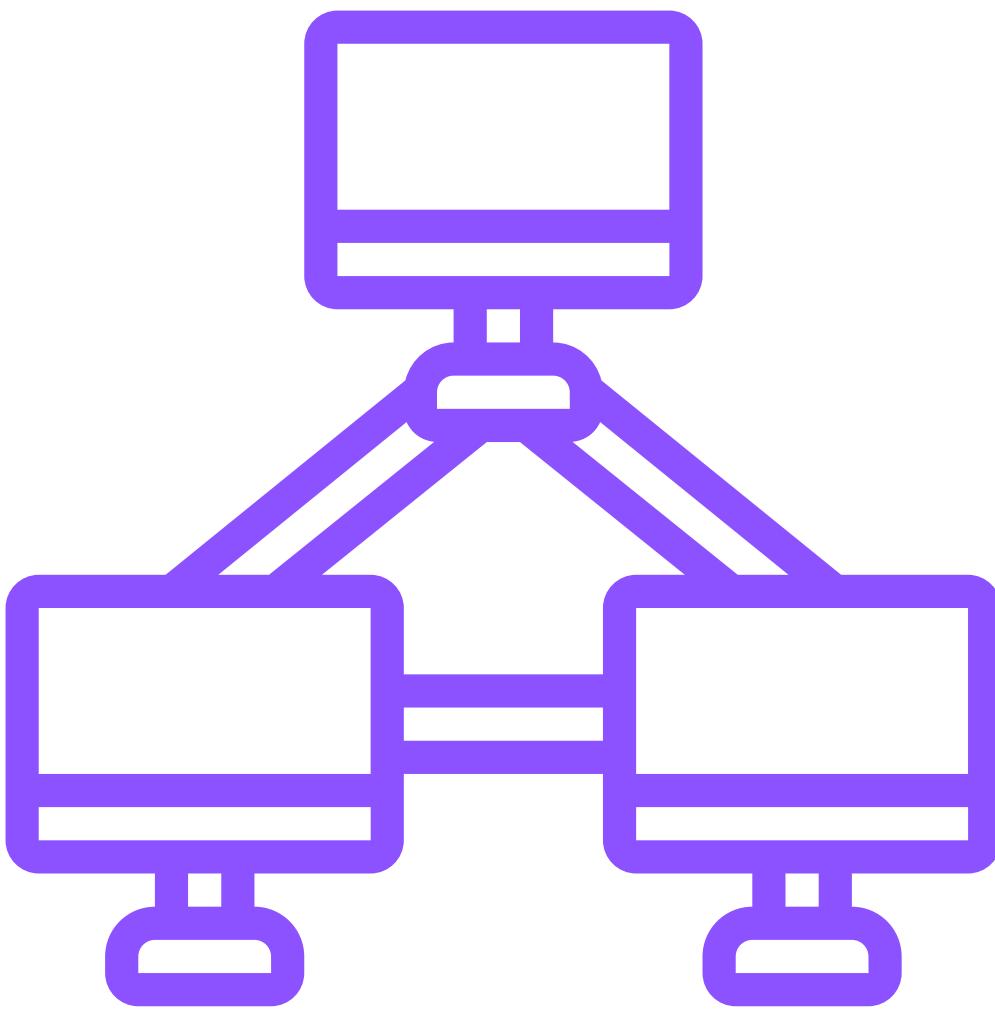
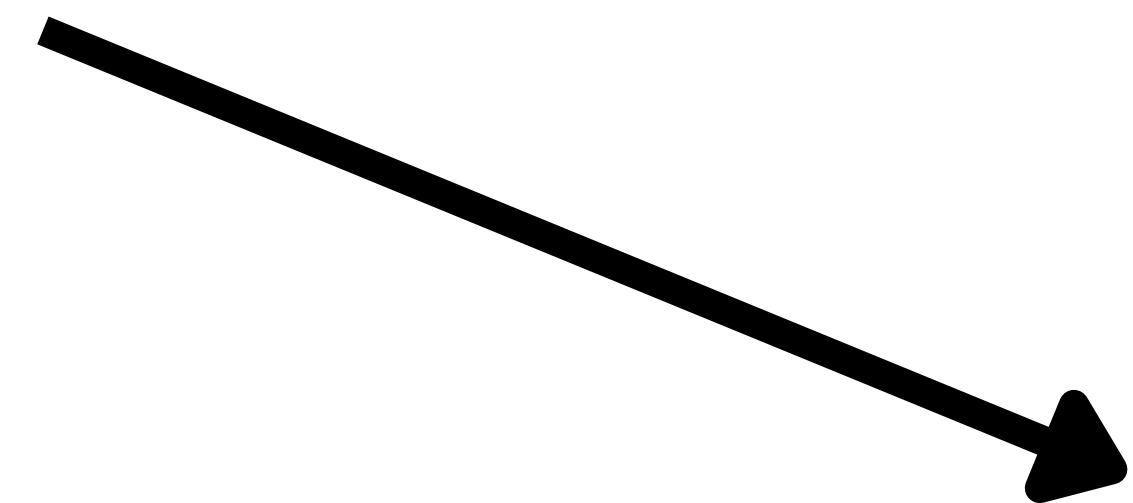
**Runs SQL Queries**



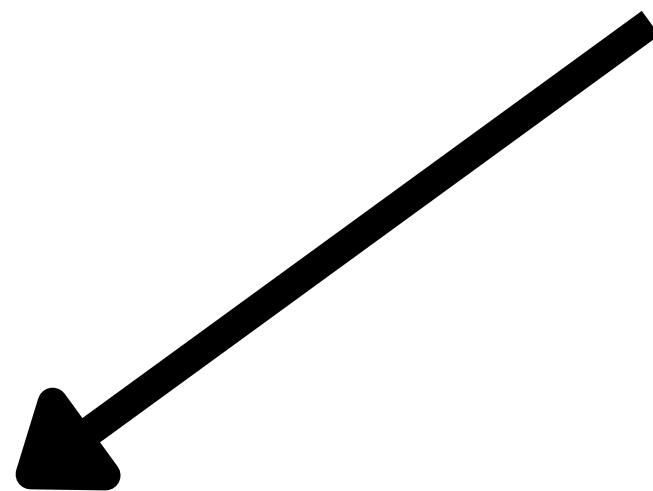
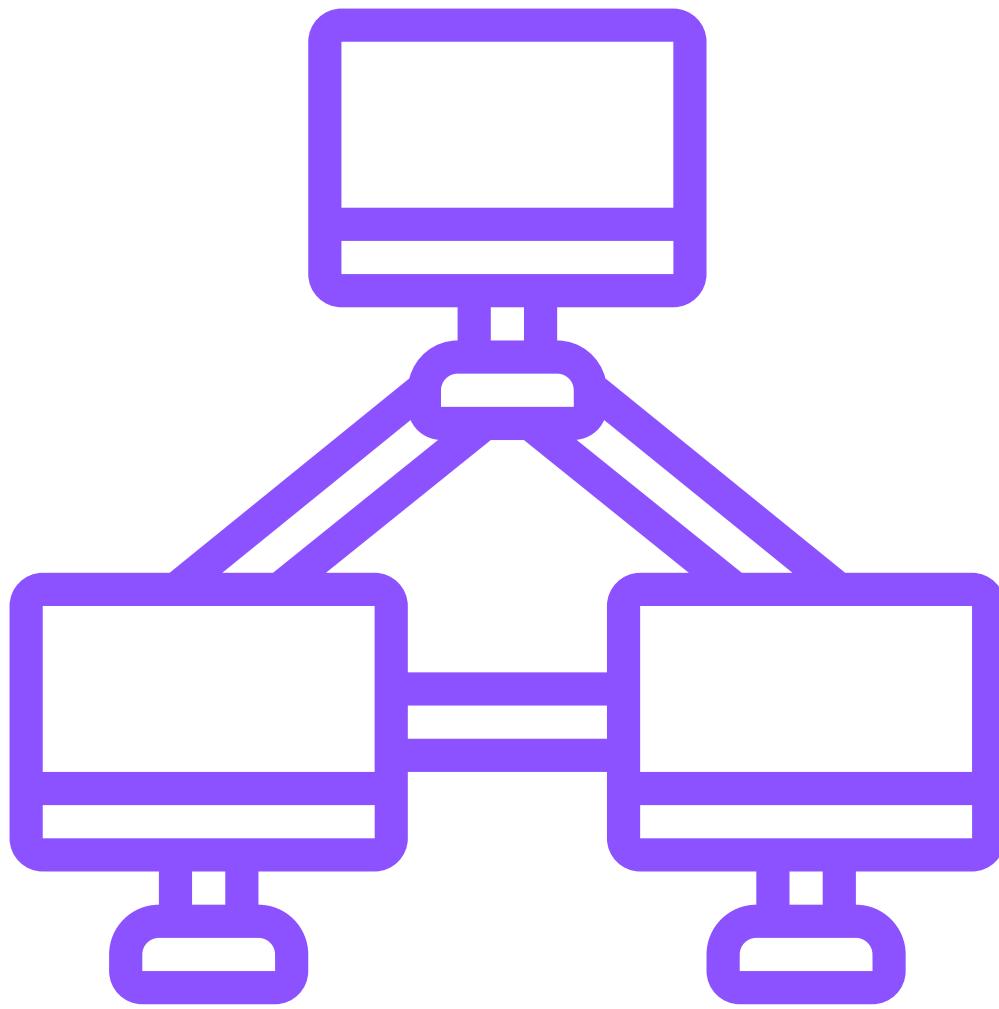
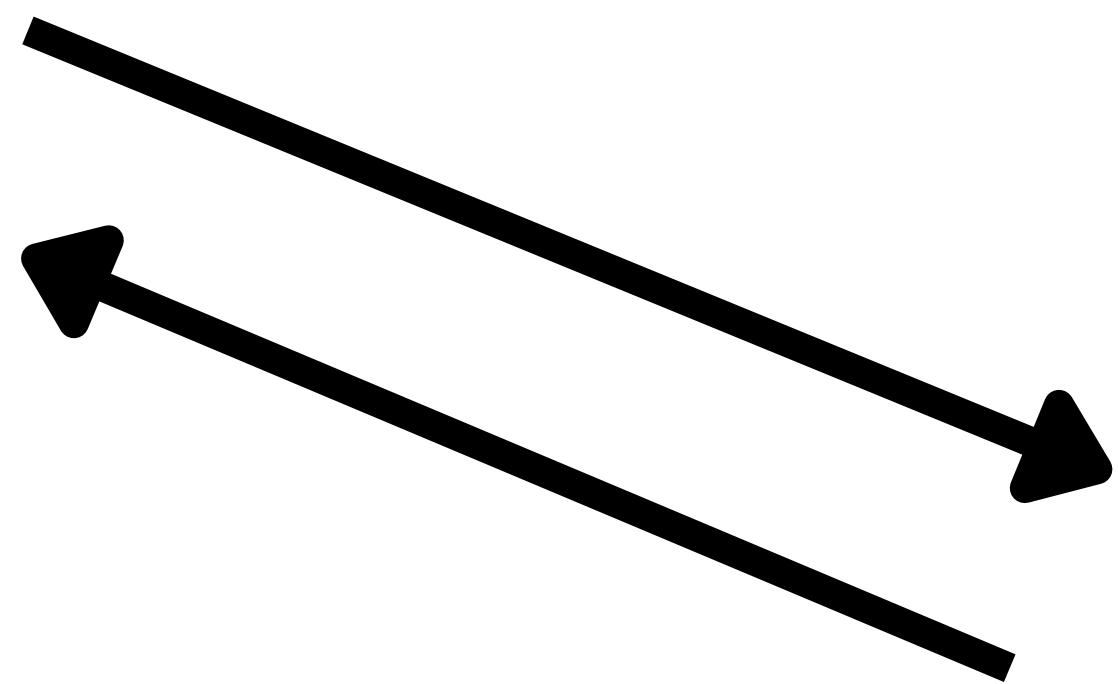




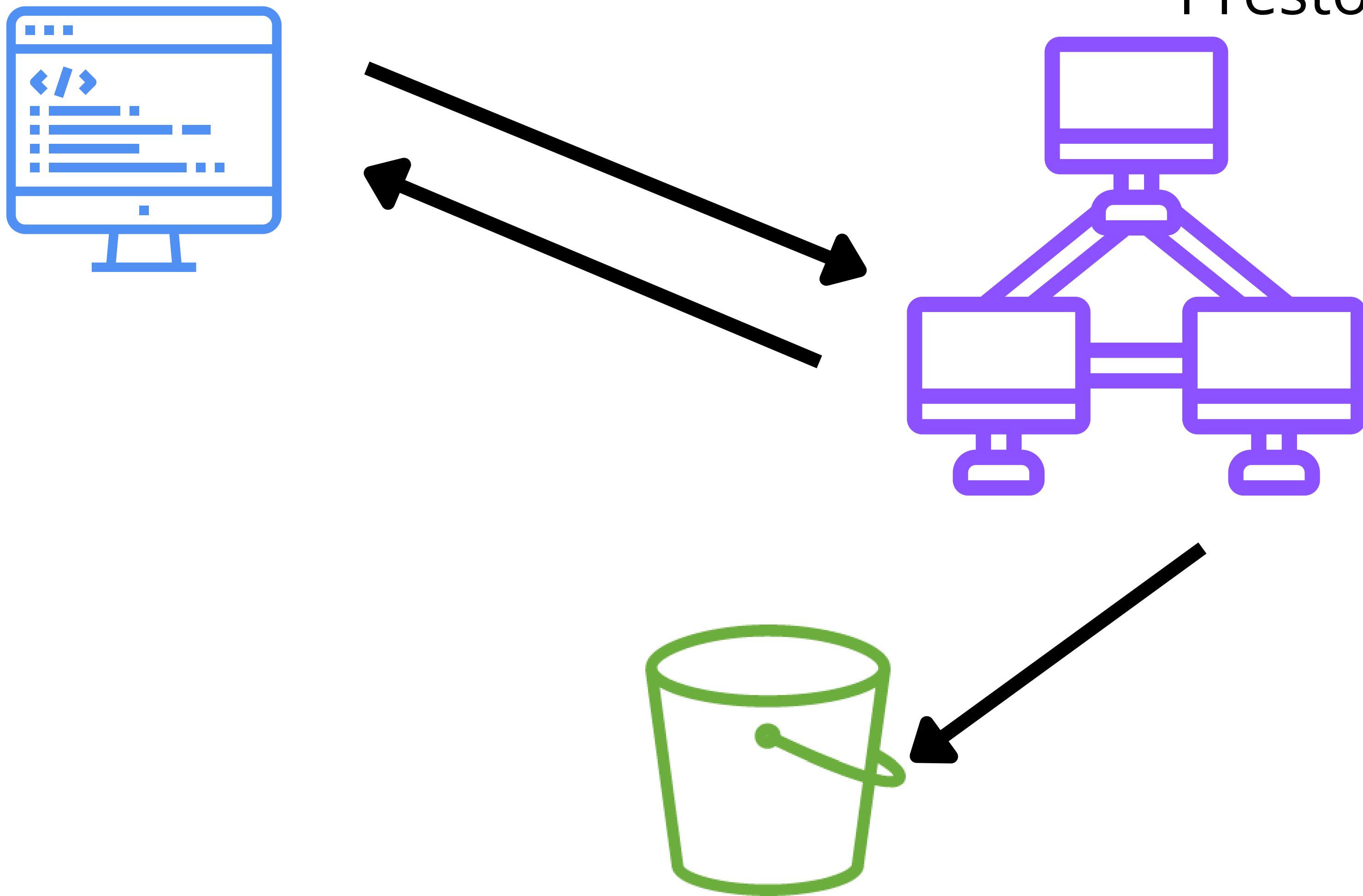
# Presto Engine



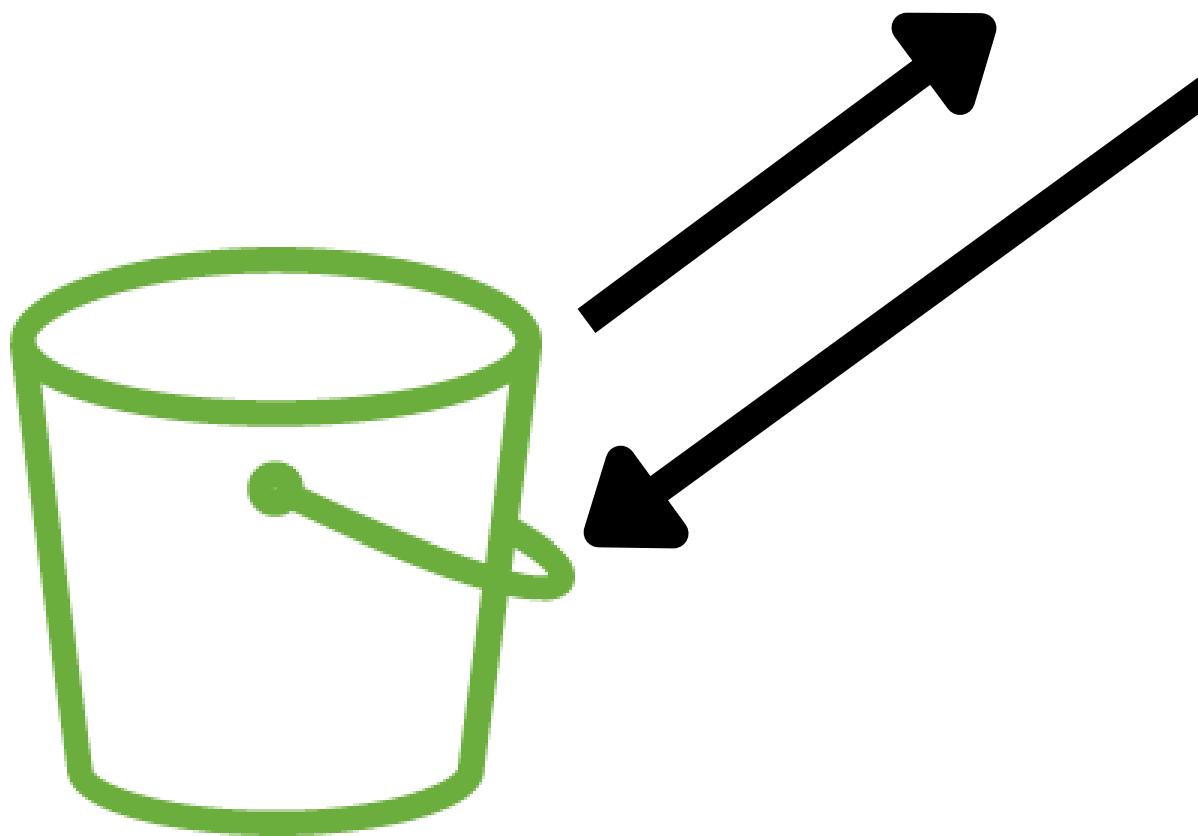
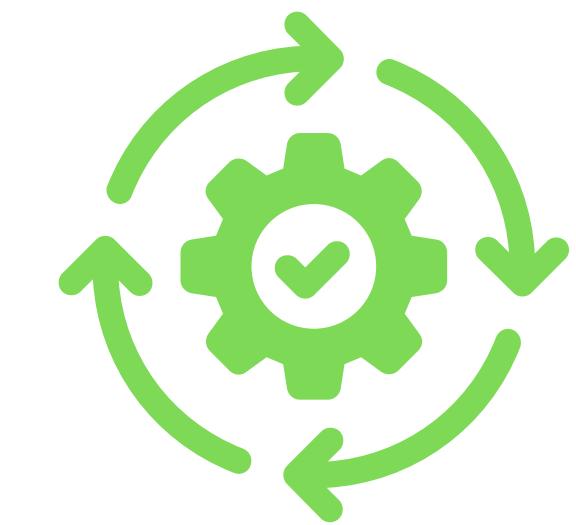
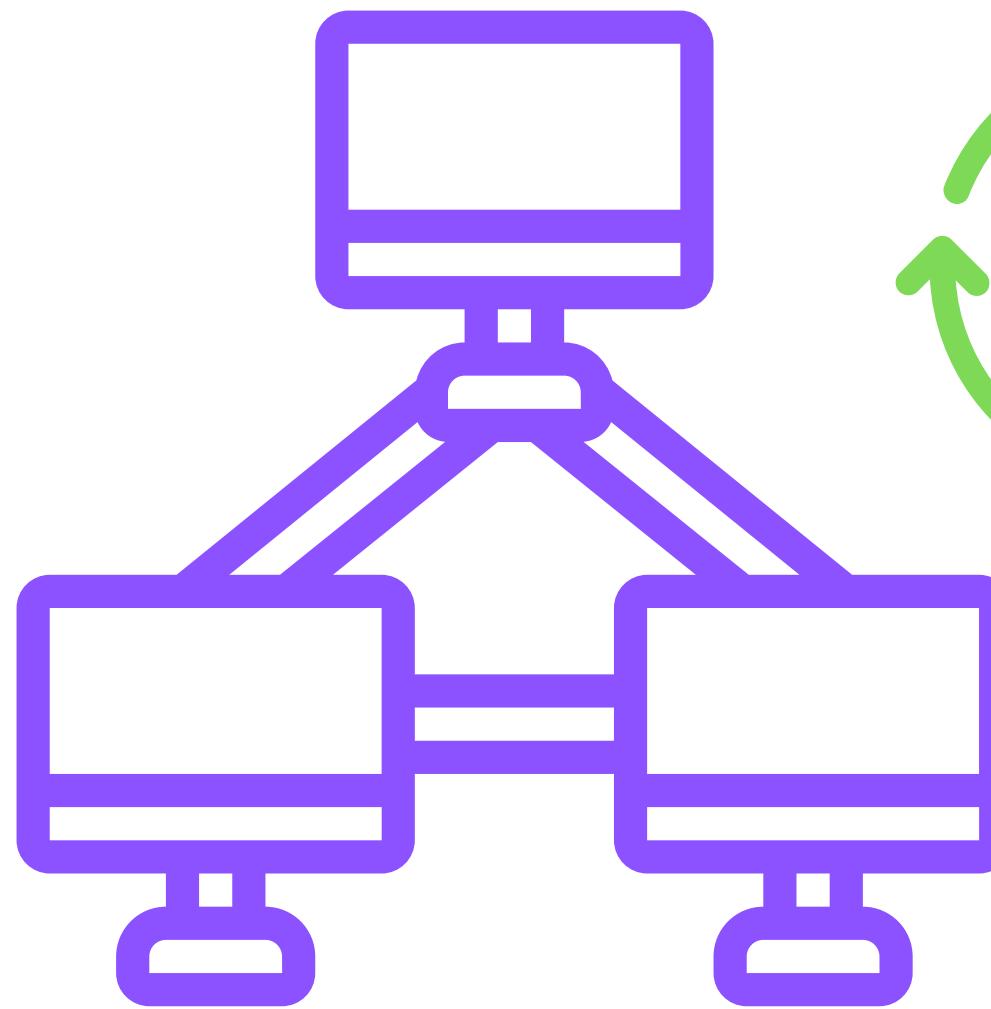
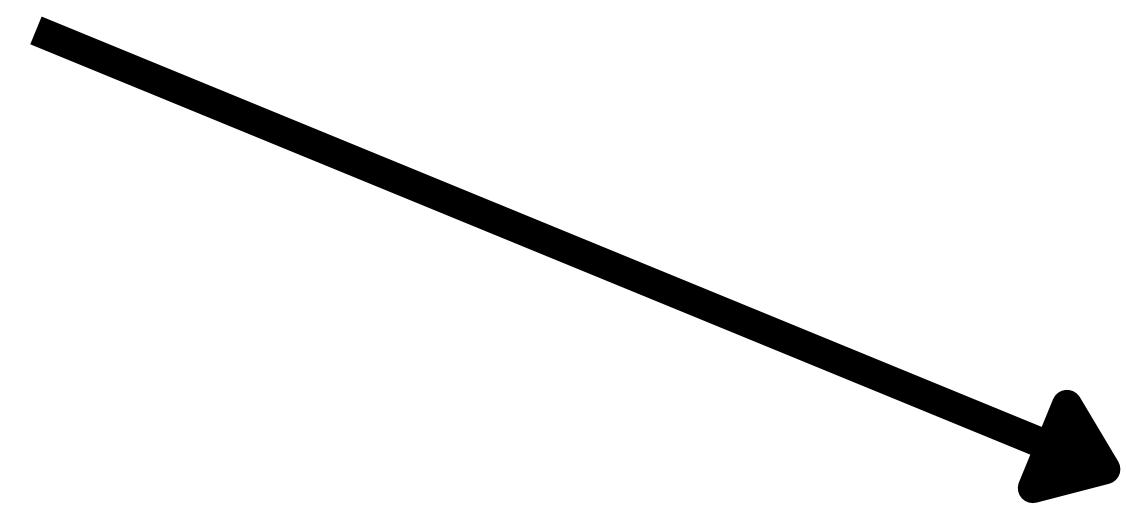
# Presto Engine



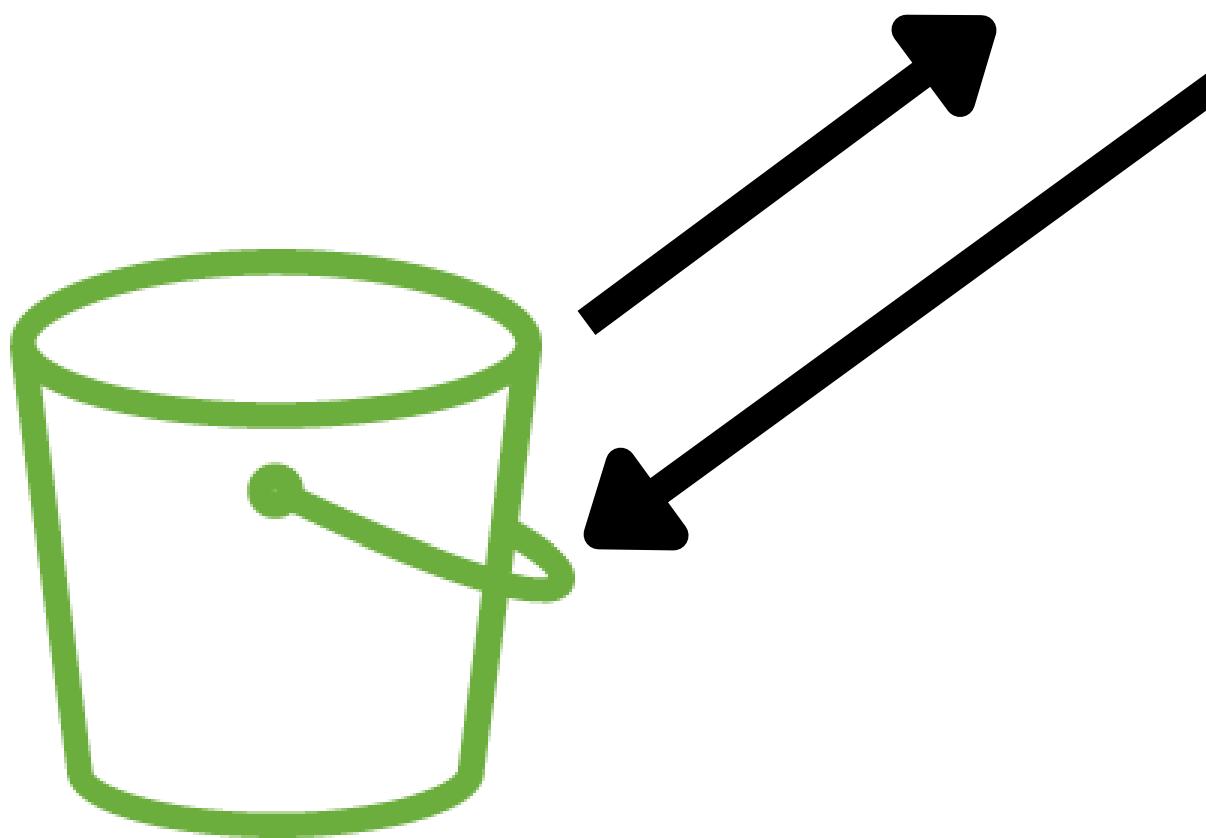
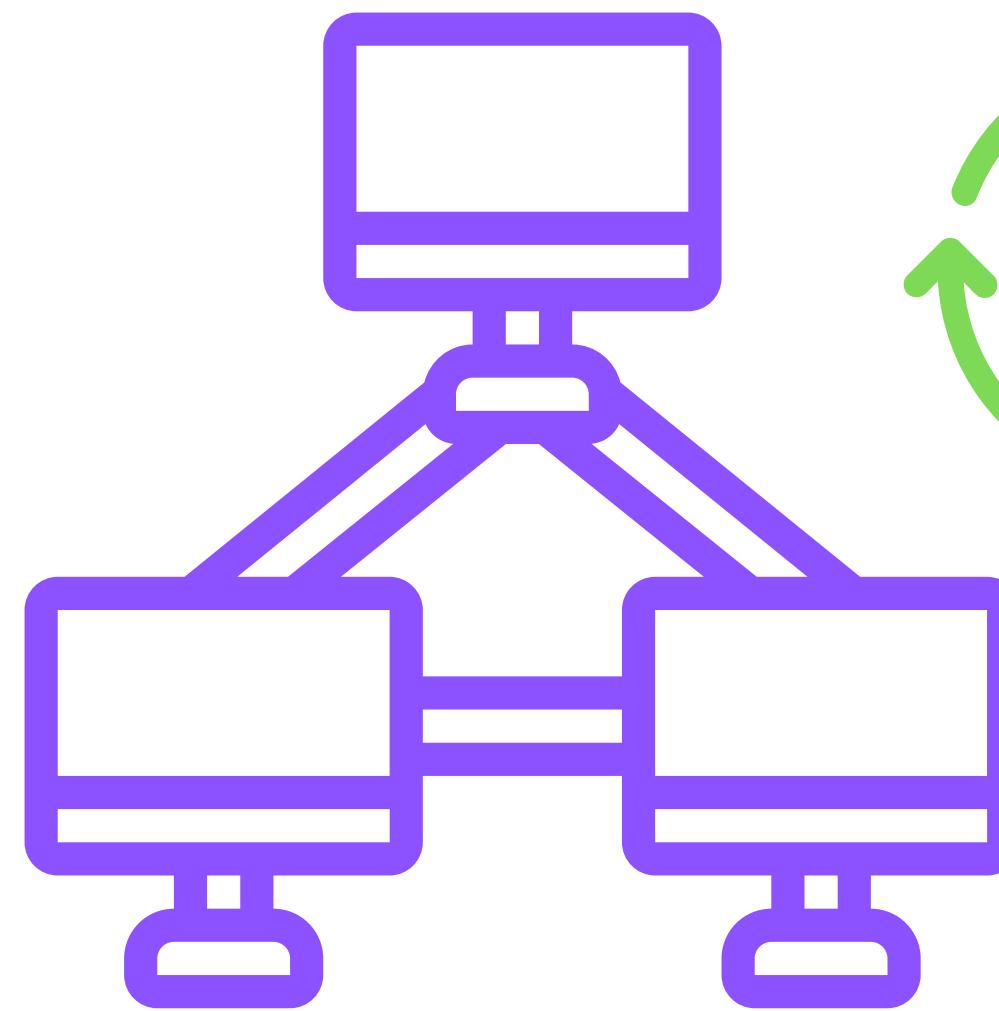
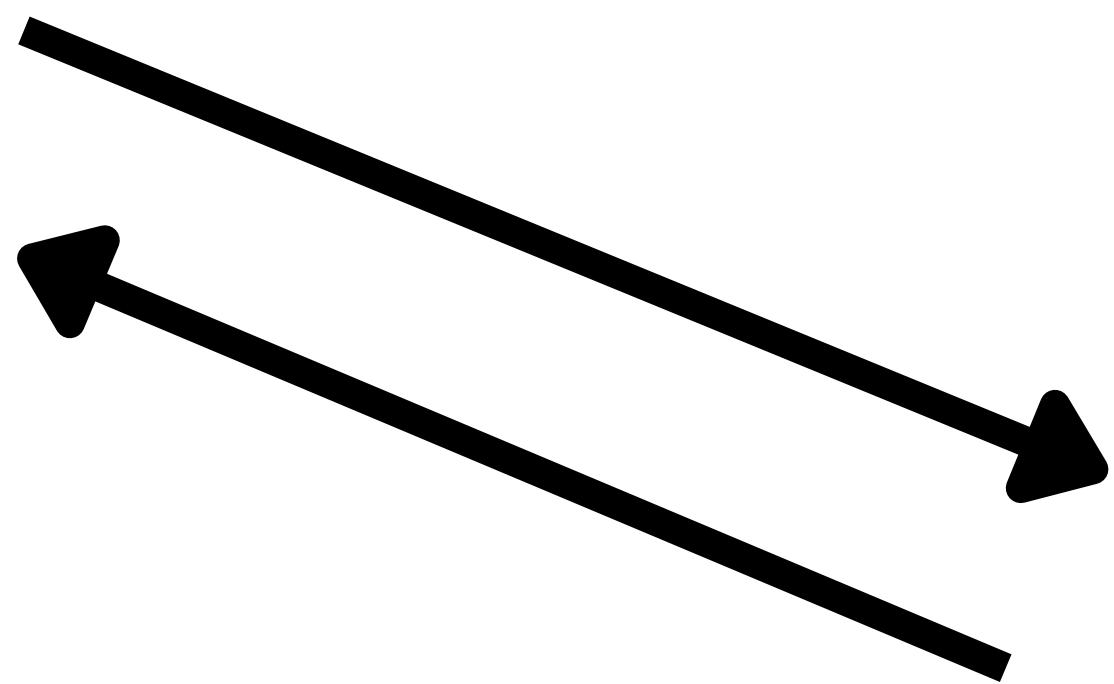
# Presto Engine



# Presto Engine



# Presto Engine



# Amazon Redshift



# Amazon Redshift

A fast, fully managed, petabyte-scale data warehouse

Designed for Online Analytical Processing (OLTP)

Queried using Structured Query Language (SQL)



# Amazon Redshift Common Use Cases

---



# Amazon Redshift Common Use Cases

---

## 01      Analytical workloads



# Amazon Redshift Common Use Cases

---

**01**

Analytical workloads

**02**

Unifying Data Warehouse



# Amazon Redshift Common Use Cases

---

**01**      **Analytical workloads**

**02**      **Unifying Data Warehouse**

**03**      **Stock Analysis**



# Amazon Redshift Common Use Cases

---

**01**

**Analytical workloads**

**02**

**Unifying Data Warehouse**

**03**

**Stock Analysis**

**04**

**Social Trends**

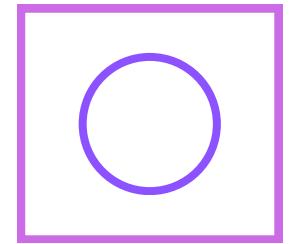


# Amazon Redshift Architecture

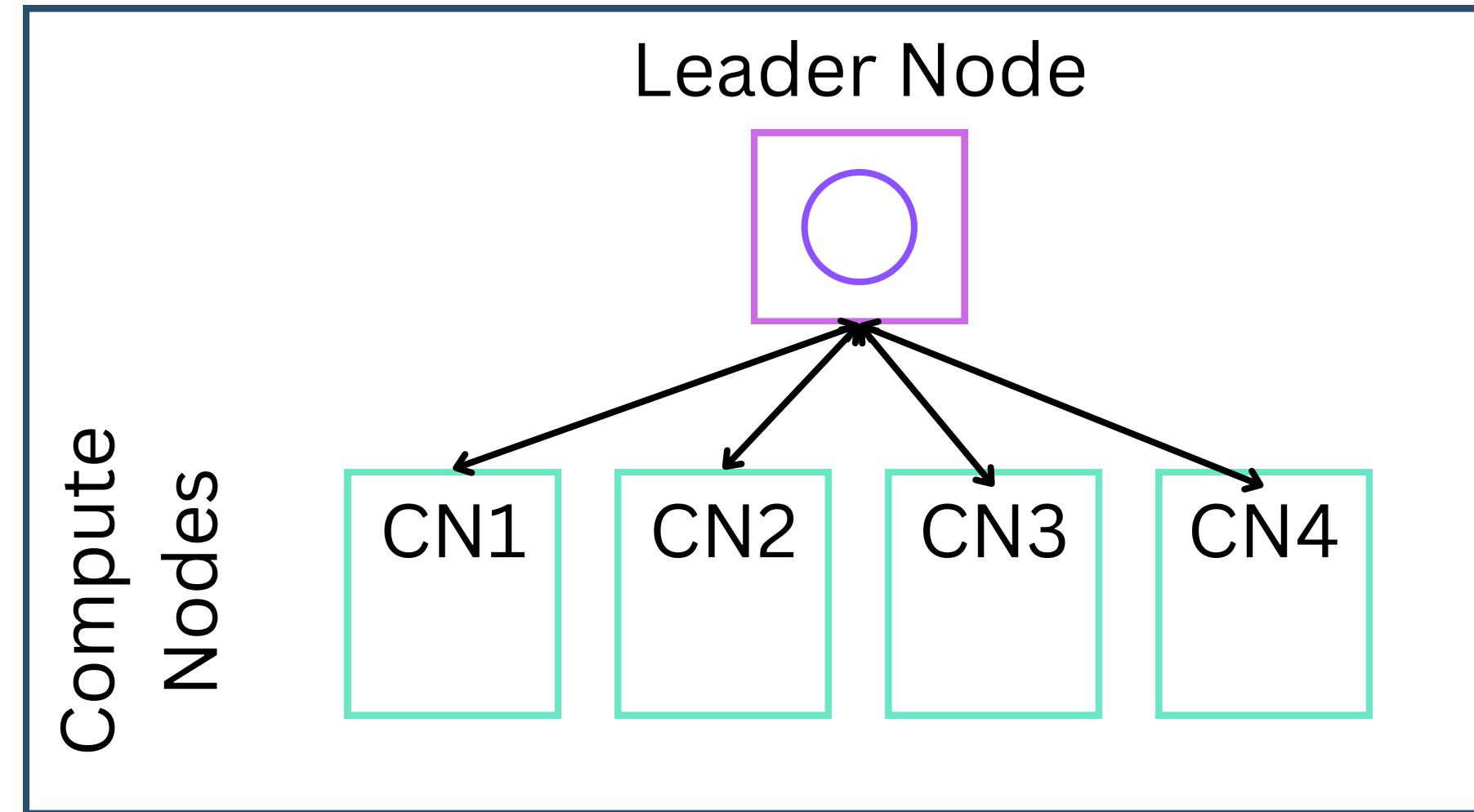


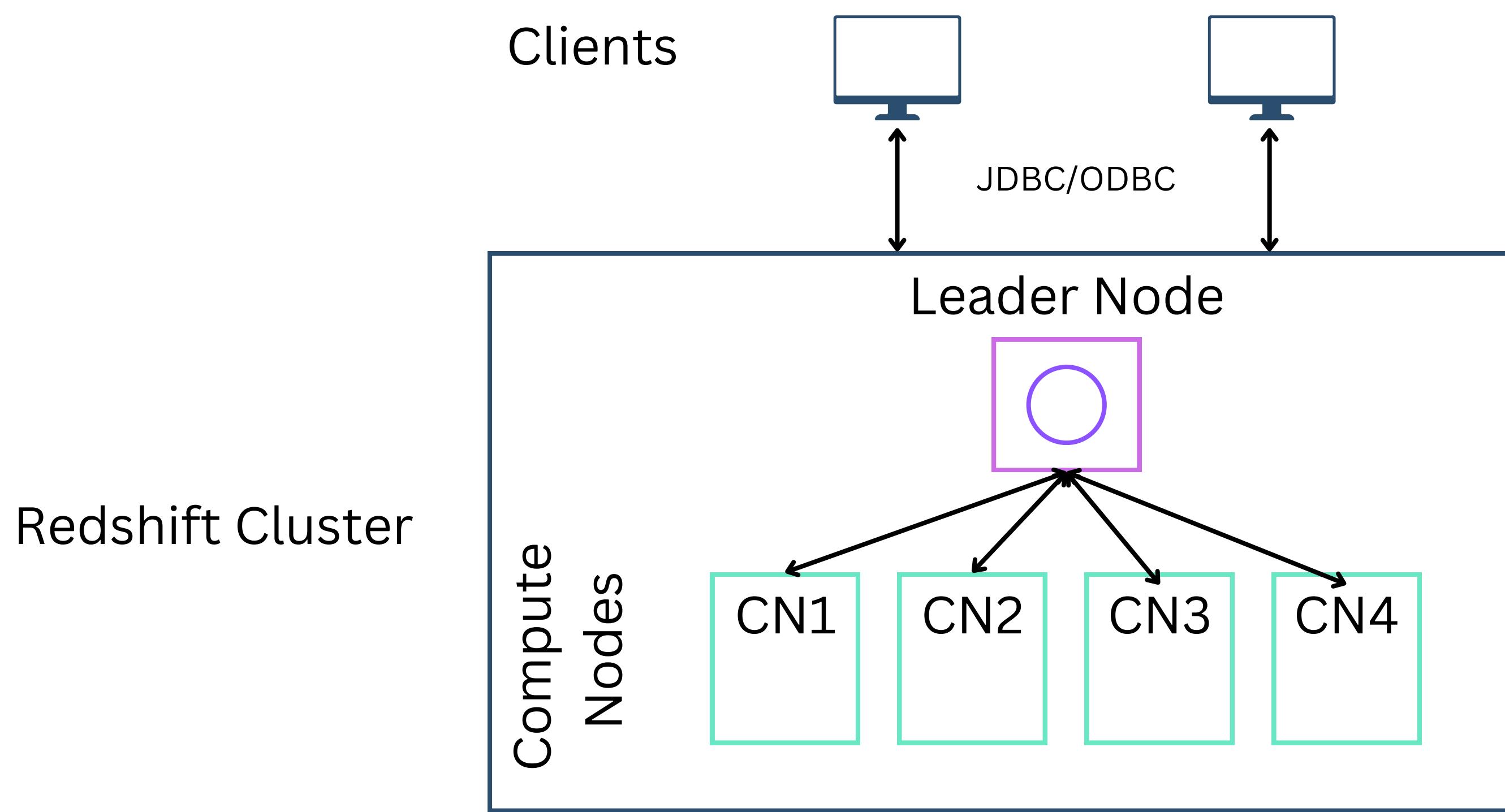
Redshift Cluster

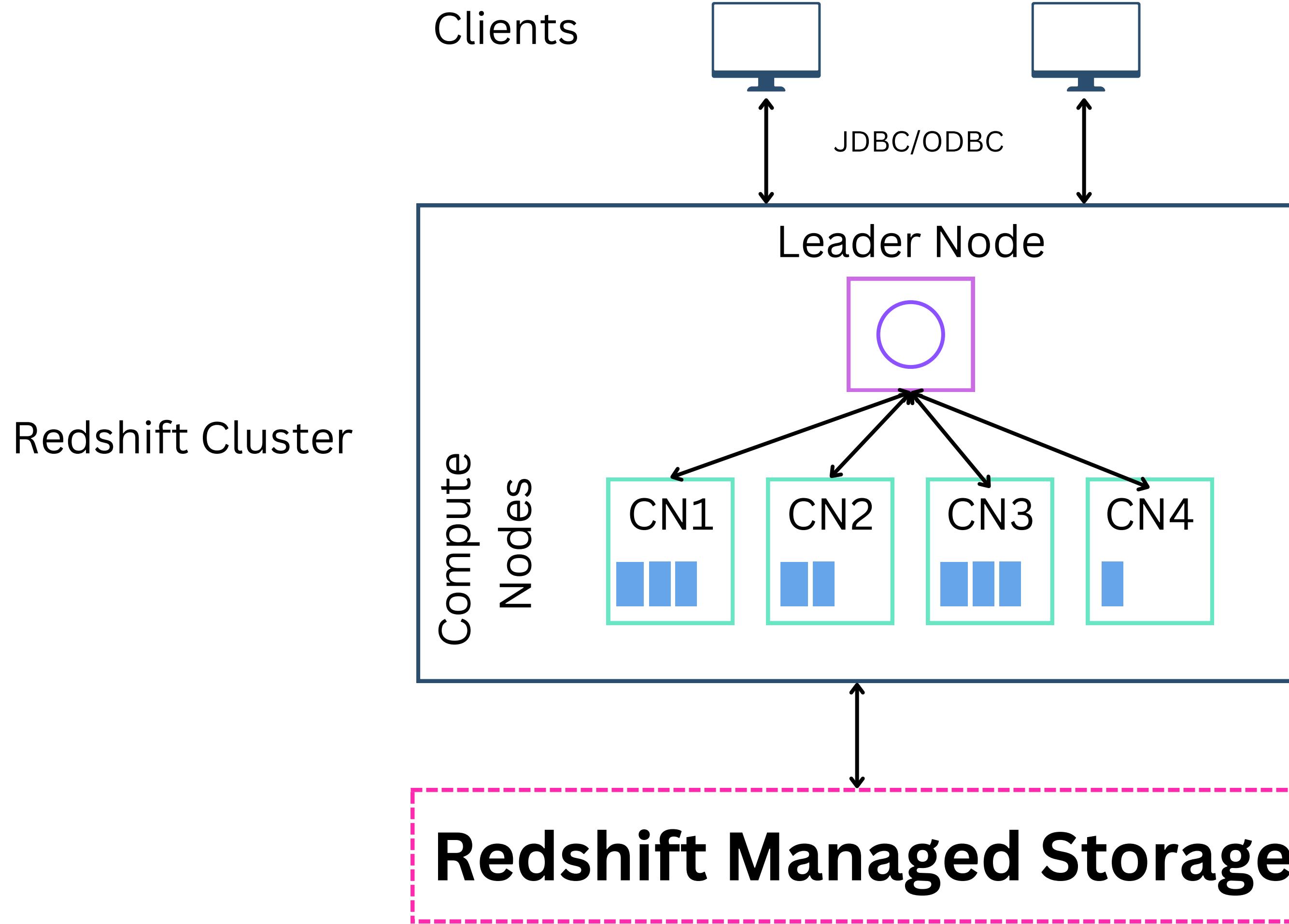
Leader Node



Redshift Cluster







# **Amazon Redshift Durability and Scalability**



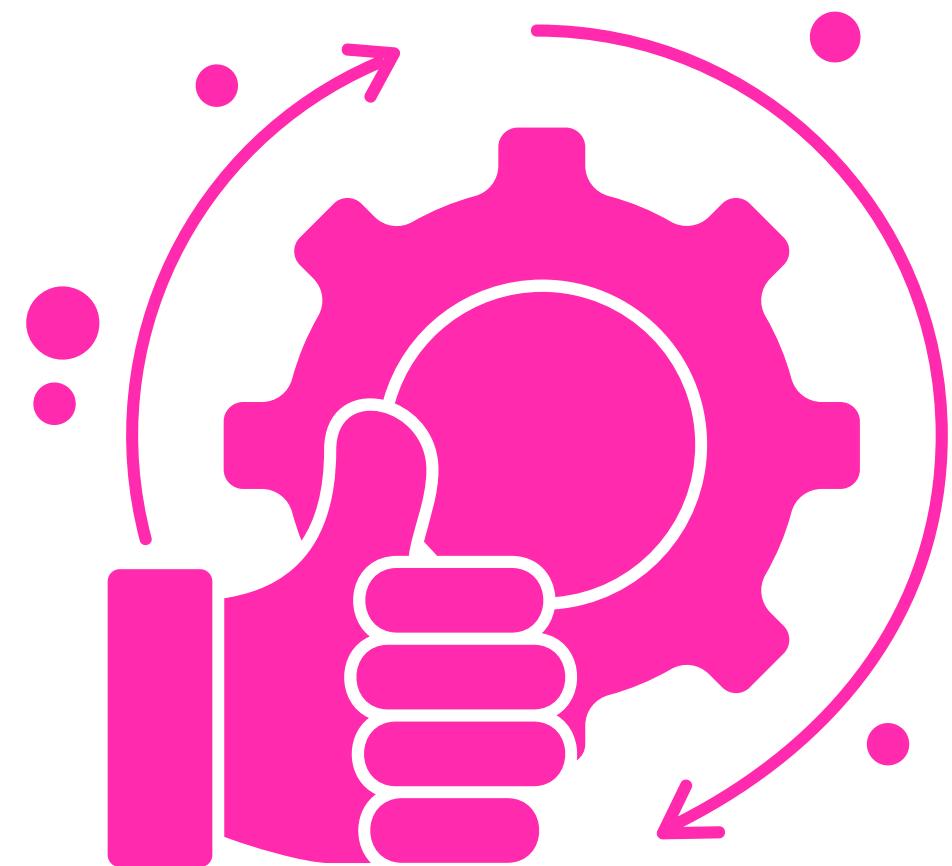
# Durability

**Data is replicated within a Cluster**

**Backed up to S3 with automated snapshots**

**S3 snapshot Asynchronously replication to other AWS regions**

**Multi AZ on RA Node Types**

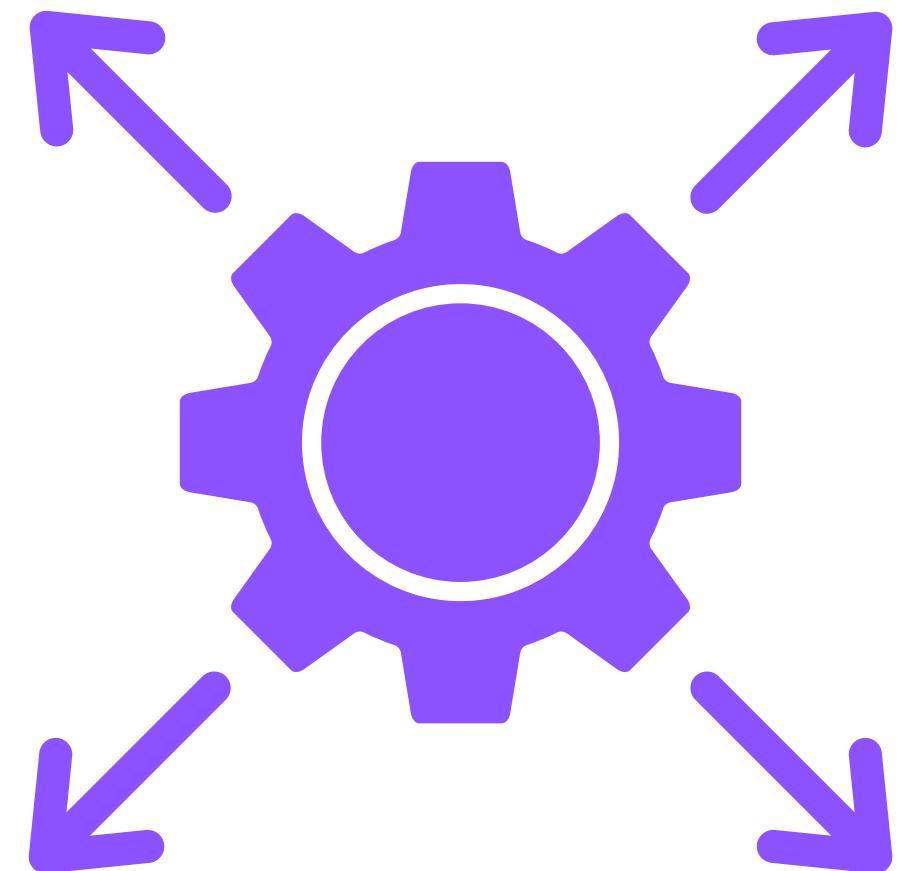


# Scalability

Elastic Resizing - Few mins Downtime

Classic Resize - Hours/Days Downtime

Snapshot and Restore - Near Zero Downtime



# Amazon Redshift Distribution



# **There are four possible row distribution styles**

---

## **01      Auto Distribution**



# **There are four possible row distribution styles**

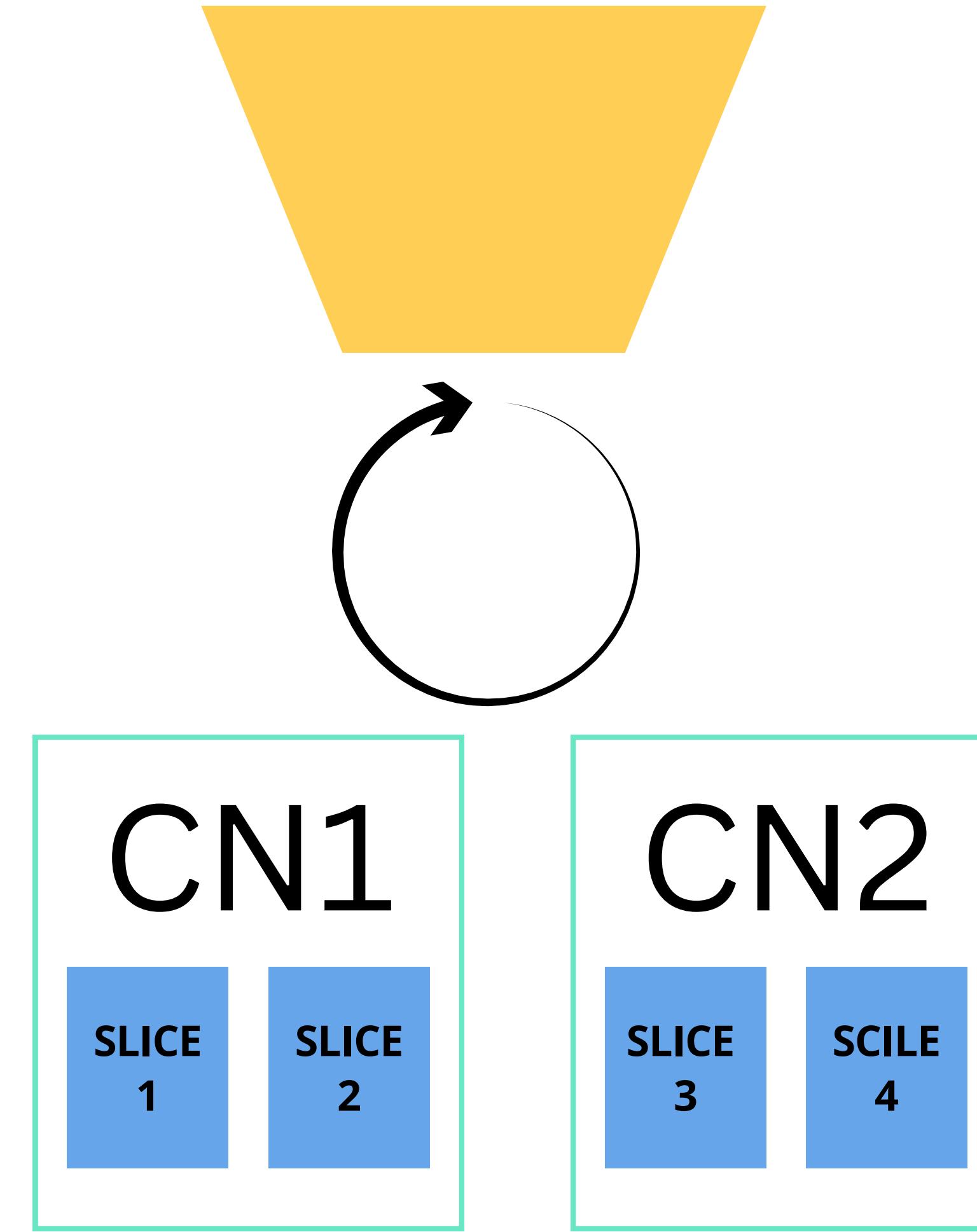
---

**01**      Auto Distribution

**02**      Even Distribution



# Even



# **There are four possible row distribution styles**

---

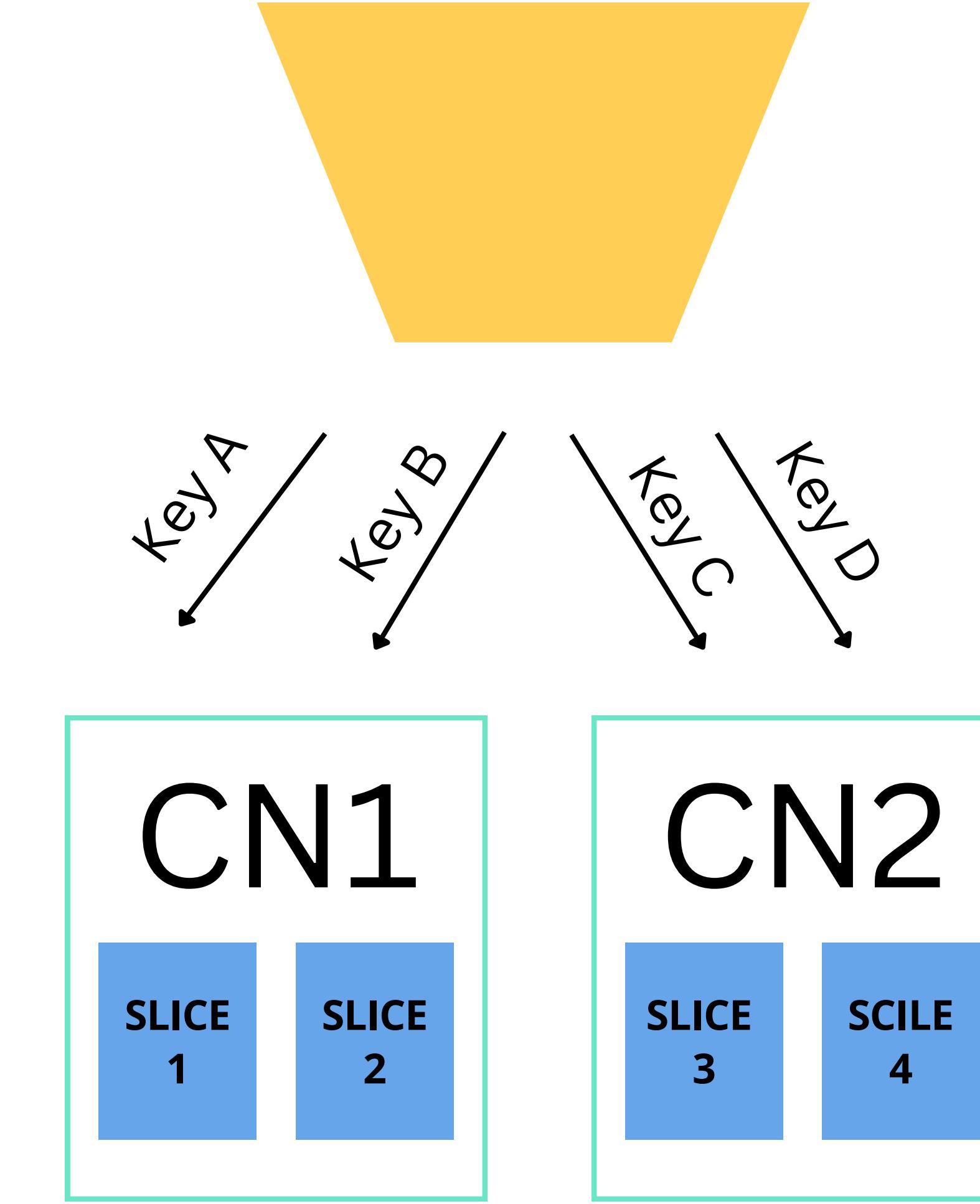
**01**      Auto Distribution

**02**      Even Distribution

**03**      Key Distribution



# KEY



# **There are four possible row distribution styles**

---

**01**      Auto Distribution

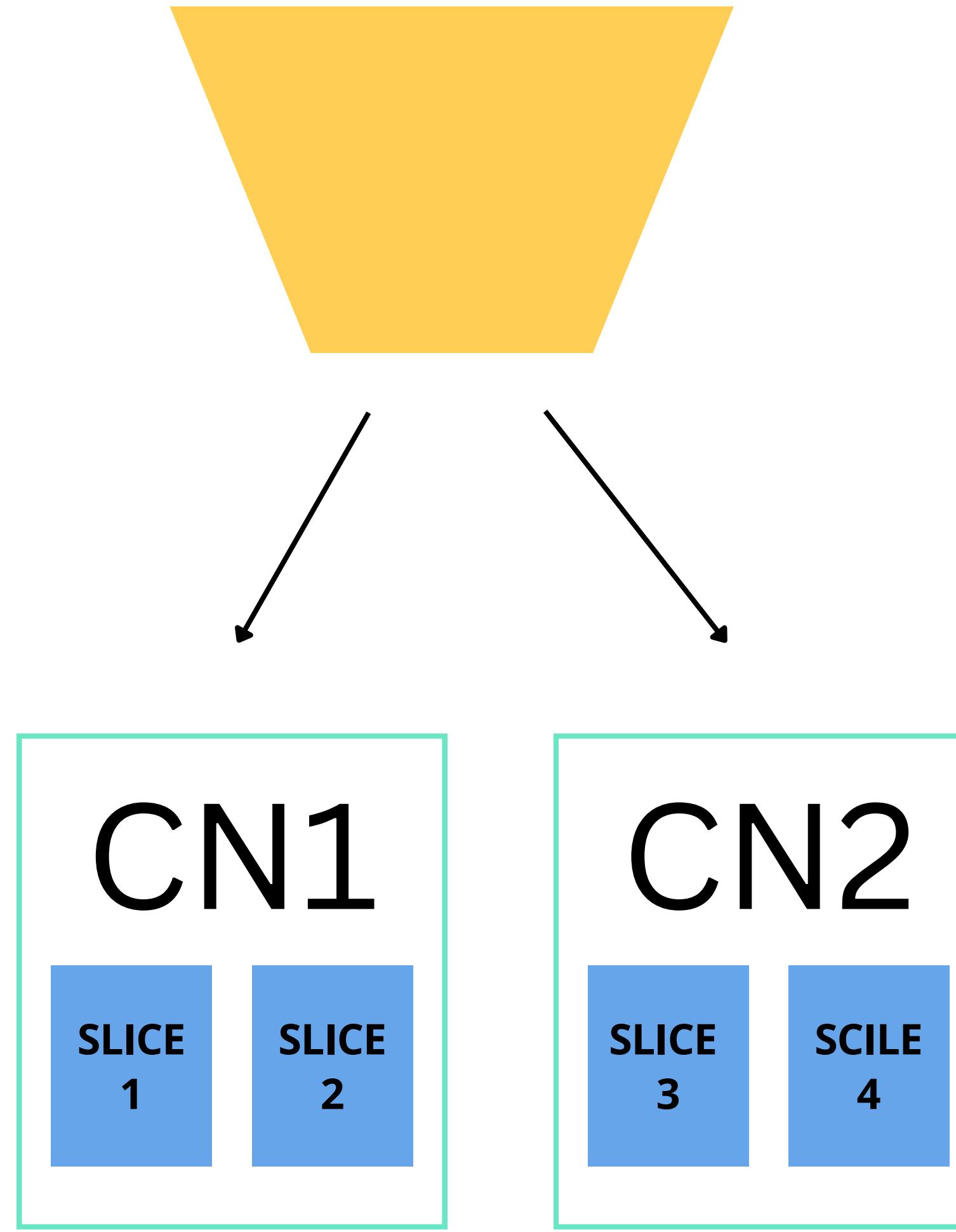
**02**      Even Distribution

**03**      Key Distribution

**04**      All Distribution



# ALL



# **Amazon Redshift**

## **Data Import/Export**



# The Copy Command

Loads data into a table from data files or  
from an Amazon DynamoDB table

COPY command can decrypt the data and  
unzip files

Parallelisation functionality

Auto Copy from S3 feature



# Zero-ETL

**From Aurora RDS to Amazon Redshift**

**Data is automatically ingested from Aurora**

**No infrastructure to manage**



# Streaming

Amazon Kinesis

Amazon MSK



# **Copying Data between Amazon Clusters in different regions**

---



# Copying Data between Amazon Clusters in different regions

---

01

Create a KMS key in the destination  
region



# Copying Data between Amazon Clusters in different regions

---

01

Create a KMS key in the destination region

02

Specify a unique name for your snapshot copy grant in the destination region



# Copying Data between Amazon Clusters in different regions

---

01

Create a KMS key in the destination region

02

Specify a unique name for your snapshot copy grant in the destination region

03

Specify the KMS Key ID for which you are creating for the Copy grant in your destination region



# Copying Data between Amazon Clusters in different regions

---

01

Create a KMS key in the destination region

02

Specify a unique name for your snapshot copy grant in the destination region

03

Specify the KMS Key ID for which you are creating for the Copy grant in your destination region

04

In the source region enable copying of snapshots to the copy grant you just created.



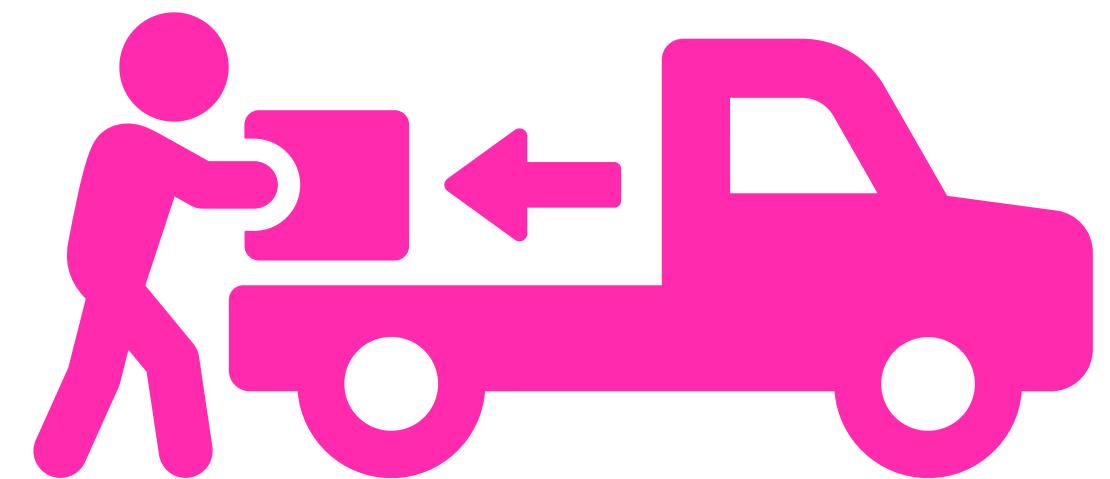
# Unload Command

**Data Exported from Amazon Redshift using  
the UNLOAD command**

**Unloads the result of a query to S3**

**Unload to text, JSON, or Apache Parquet**

**Server side encryption supported**



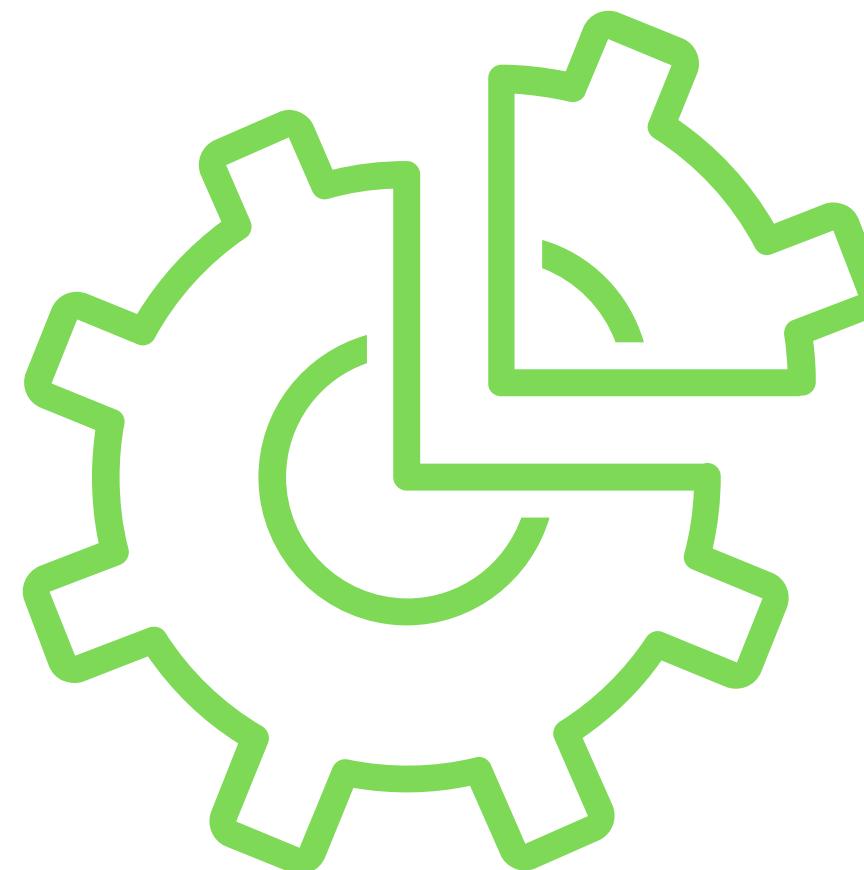
# Integrations

S3

DynamoDB

EMR

EC2



# Amazon Redshift Vacuum Command



# Vacuum command recovers space from deleted rows

---



# Vacuum command recovers space from deleted rows

---

01

Vacuum Full



# Vacuum command recovers space from deleted rows

---

01

Vacuum Full

02

Vacuum Delete Only



# Vacuum command recovers space from deleted rows

---

- 01 Vacuum Full
- 02 Vacuum Delete Only
- 03 Vacuum Sort Only



# Vacuum command recovers space from deleted rows

---

01

Vacuum Full

02

Vacuum Delete Only

03

Vacuum Sort Only

04

Vacuum ReIndex



# **Amazon Redshift Workload Management**



# Workload Management (WLM)

**When users run queries in Amazon Redshift, the queries are routed to query queues**

**Each query queue contains a number of query slots**

**Each queue is allocated a portion of the cluster's available memory.**

**A queue's memory is divided among the queue's query slots.**

**(WLM) enables users to flexibly manage priorities**



# **Manual WLM**

**One default queue with a concurrency level of 5**

**Super user queue with a currency level of 1**

**Up to 8 queues can be created with Max concurrency 50**



# **Short query Acceleration (SQA)**

**Prioritizes selected short-running queries ahead of longer-running queries**

**SQA runs short-running queries in a dedicated space**



# **Concurrency Scaling Feature**

**Support thousands of concurrent users and concurrent queries**

**Automatically adds additional cluster capacity to process an increase in both read and write queries**

**Manage which queries are sent to the concurrency-scaling cluster by configuring WLM queues**



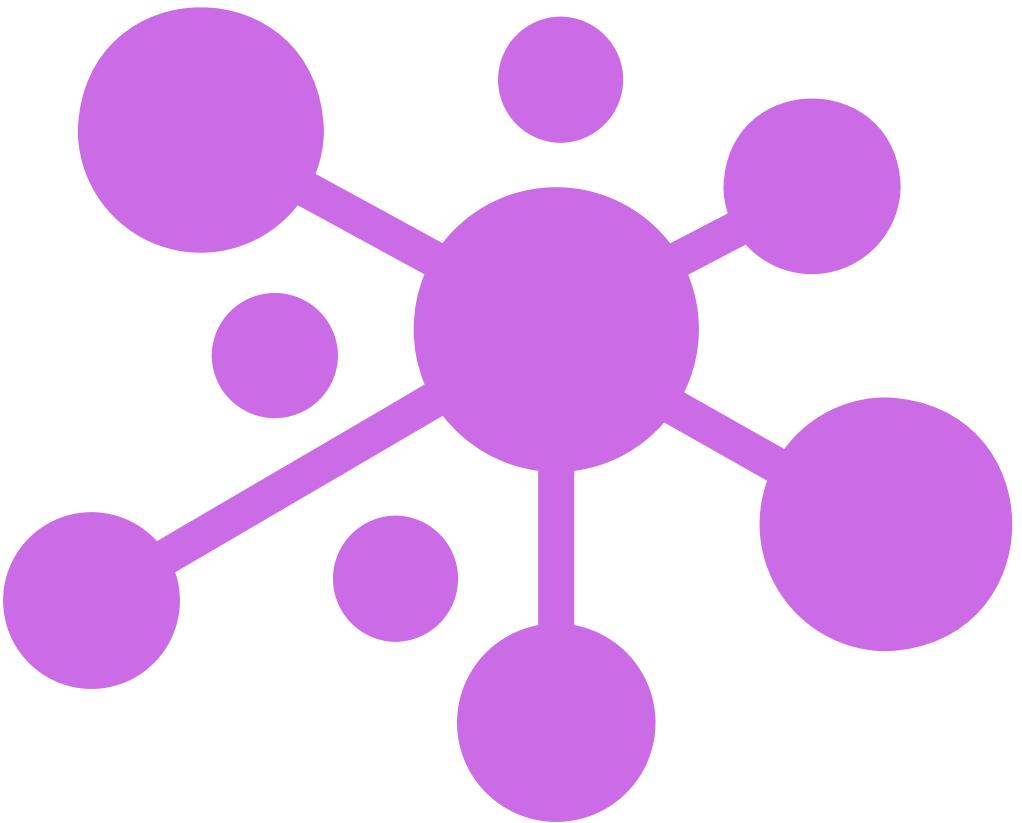
# **Amazon Redshift Other Things You Need To Know**



# RA3 Nodes

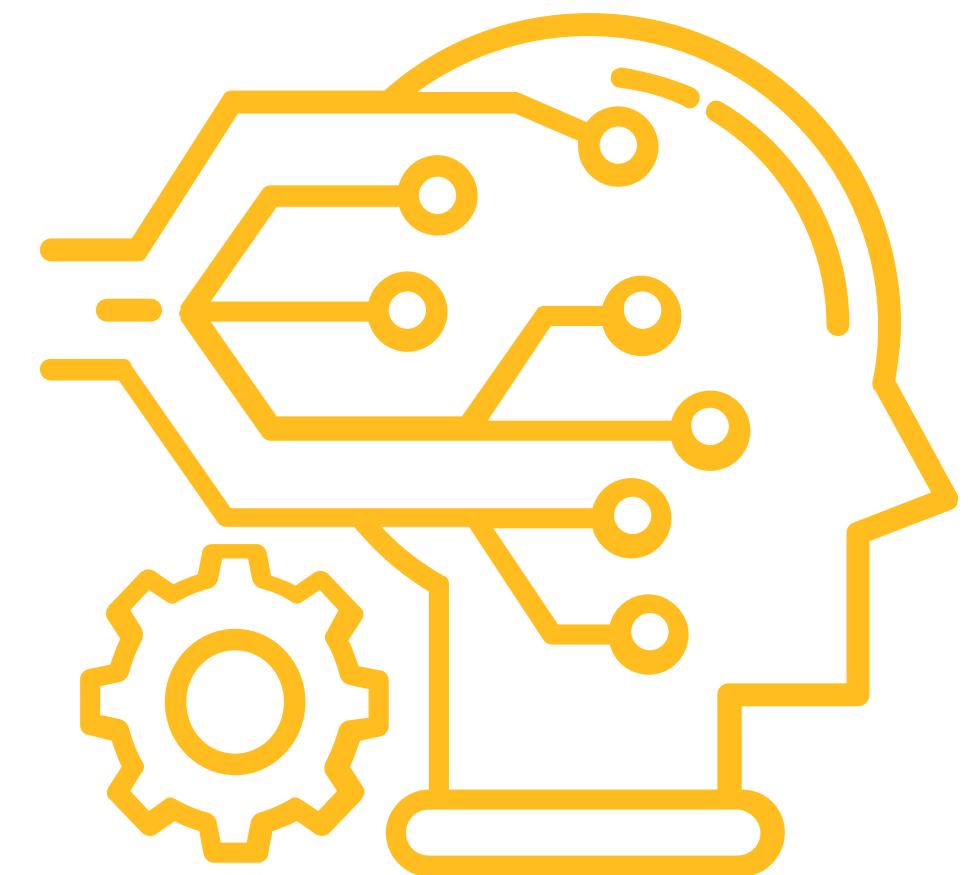
**Allow for compute to be independently scaled from storage**

**Only pay for the managed storage that is used**

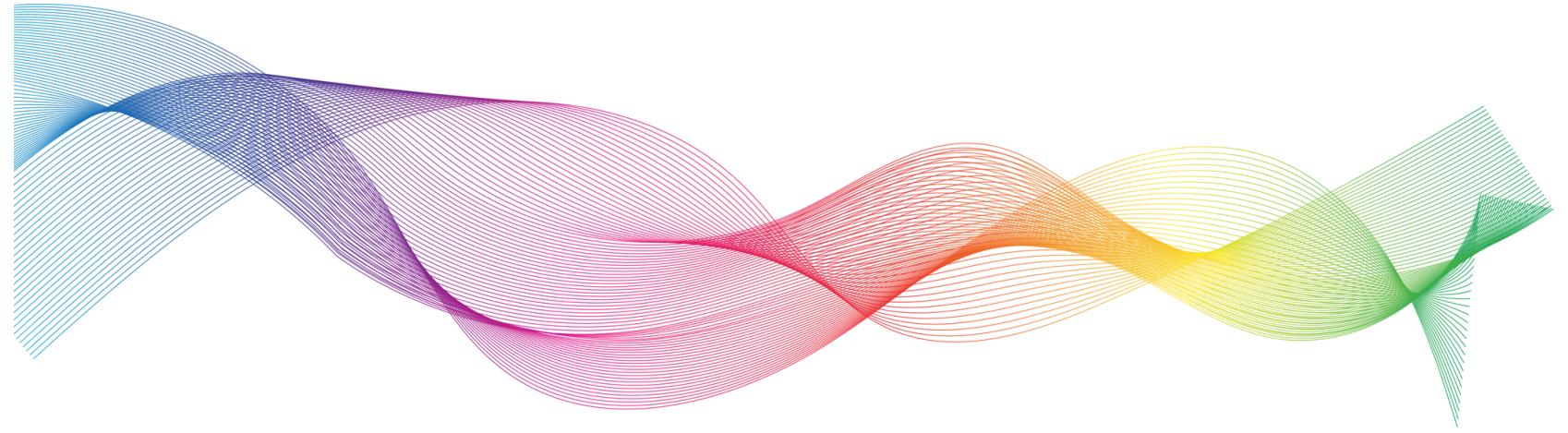


# Redshift Machine Learning

Machine learning models can be created by issuing SQL commands



# Amazon Redshift Spectrum

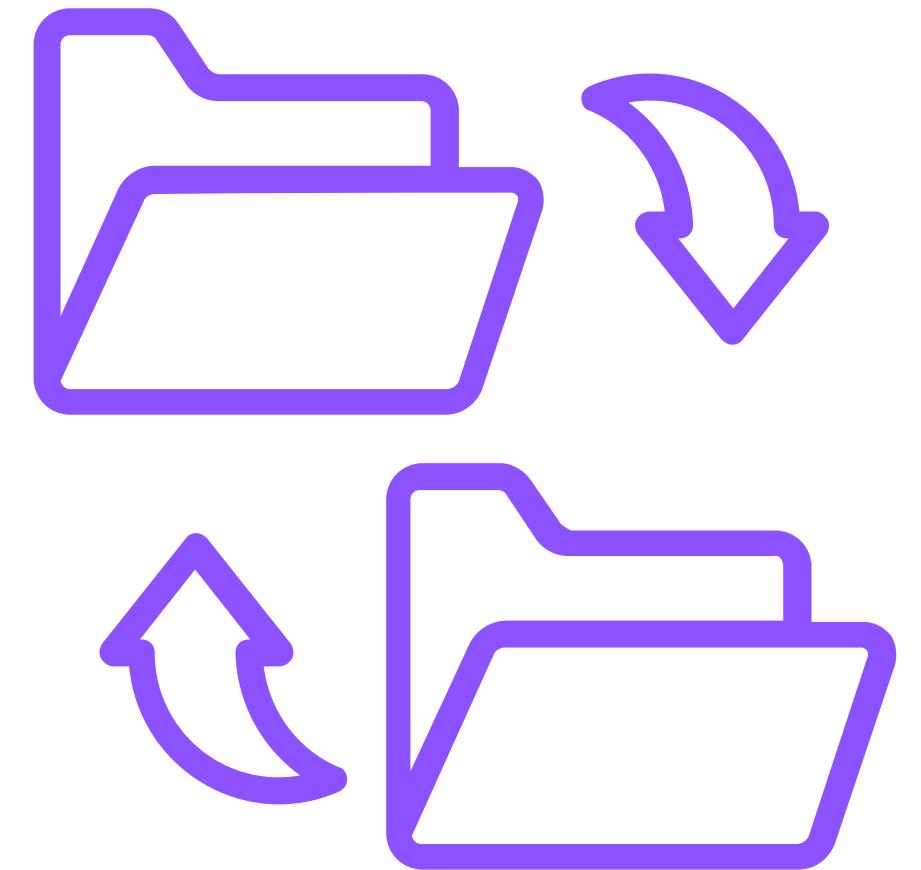


**Data held in S3 can be queried by a dedicated Amazon Redshift server which is independent of your Amazon Redshift cluster**



# Redshift Data Sharing / Data Shares

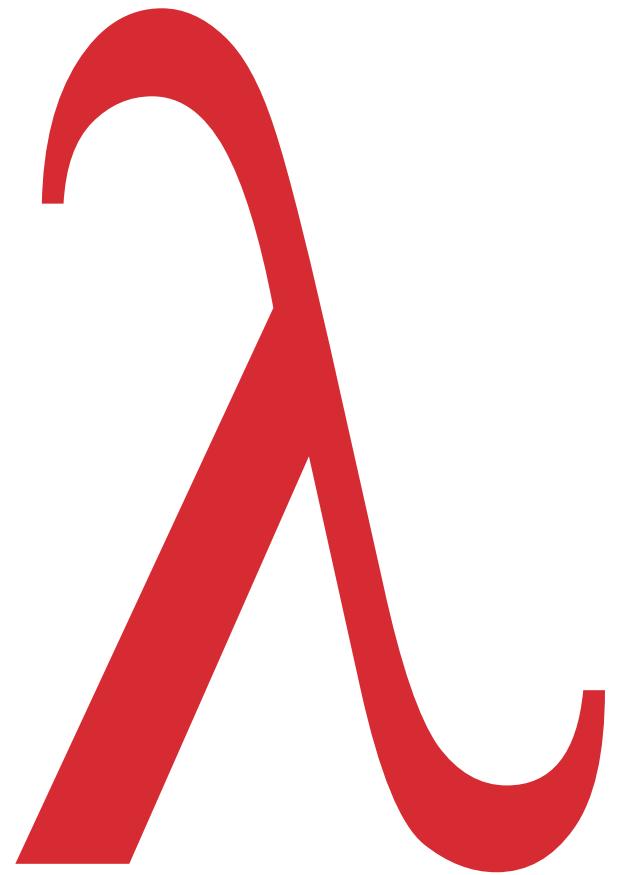
**Live data across Amazon Redshift Clusters,  
Workgroups, AWS accounts and AWS  
Regions**



# Lambda UDF

**Custom AWS Lambda Function inside SQL  
syntax**

**Redshift must have permission to invoke  
these lambda functions**



# Redshift Federated Queries

**Allow your Redshift Cluster to access data  
from Amazon RDS instances**

**This is done by creating an external schema  
in Amazon Redshift**



# Redshift Serverless

**No Servers need to be provisioned**

**Automatically scales for your workloads  
based on Redshift Processing Units (RPUs)**

**There are no WLM or public endpoints**



# Amazon EMR

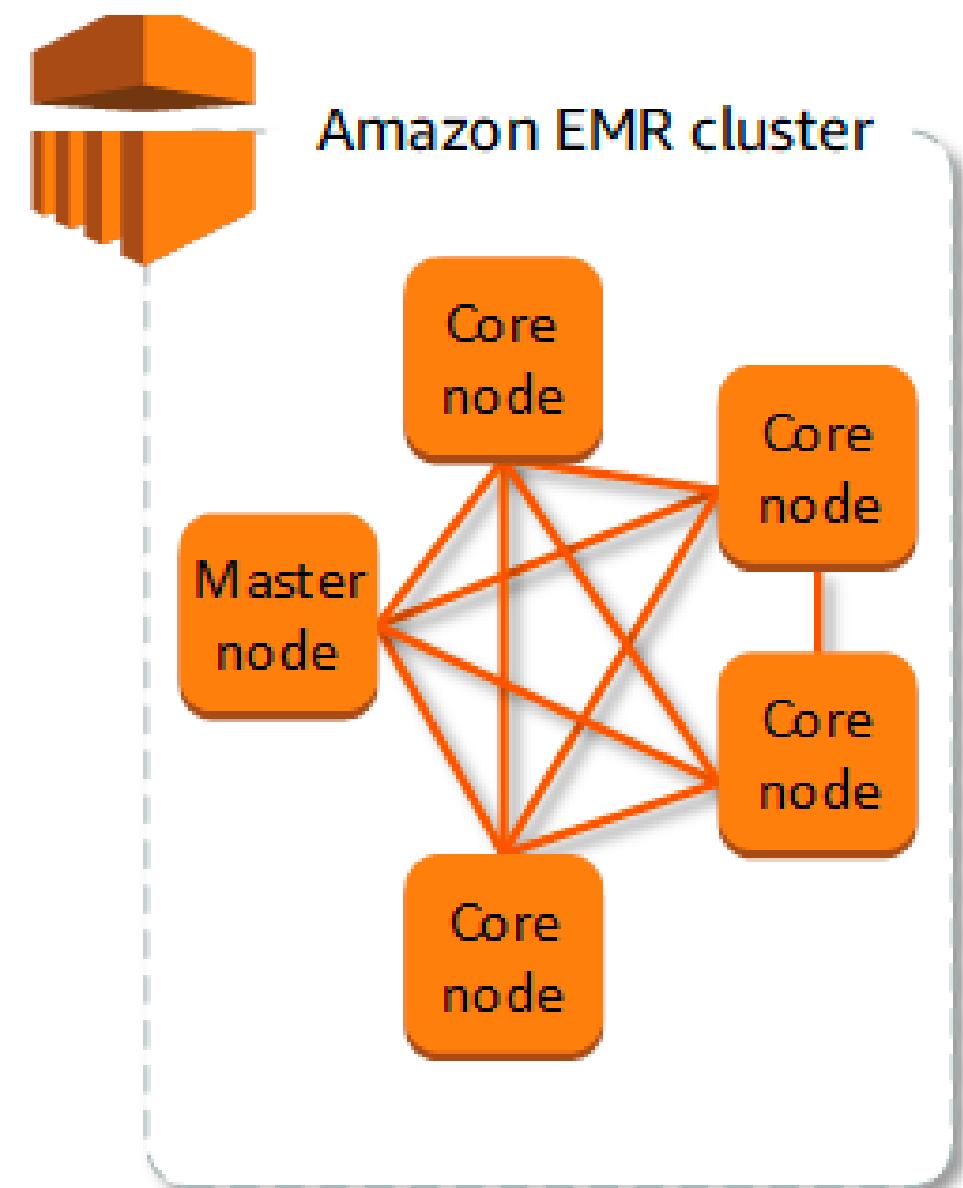


# What is AWS EMR?

**Master node:** A node that manages the cluster by running software components to coordinate the distribution of data and tasks among other nodes

**Core node:** A node with software components that run tasks and store data in the Hadoop Distributed File System (HDFS) on your cluster. Multi-node clusters have at least one core node.

**Task node:** A node with software components that only runs tasks and does not store data in HDFS. Task nodes are optional.



# What is AWS EMR? continued...

## Data Processing Frameworks

Engine used to process and analyze data

Different frameworks are available for different kinds of processing needs

The main processing frameworks available for Amazon EMR are Hadoop MapReduce and Spark

## Cluster Resource Management

The resource management layer is responsible for managing cluster resources and scheduling the jobs for processing data.

By default, Amazon EMR uses YARN (Yet Another Resource Negotiator).

## Storage

Hadoop Distributed File System (HDFS) is a distributed, scalable file system for Hadoop.

Using the EMR File System (EMRFS), Amazon EMR extends Hadoop to add the ability to directly access data stored in Amazon S3 as if it were a file system like HDFS.

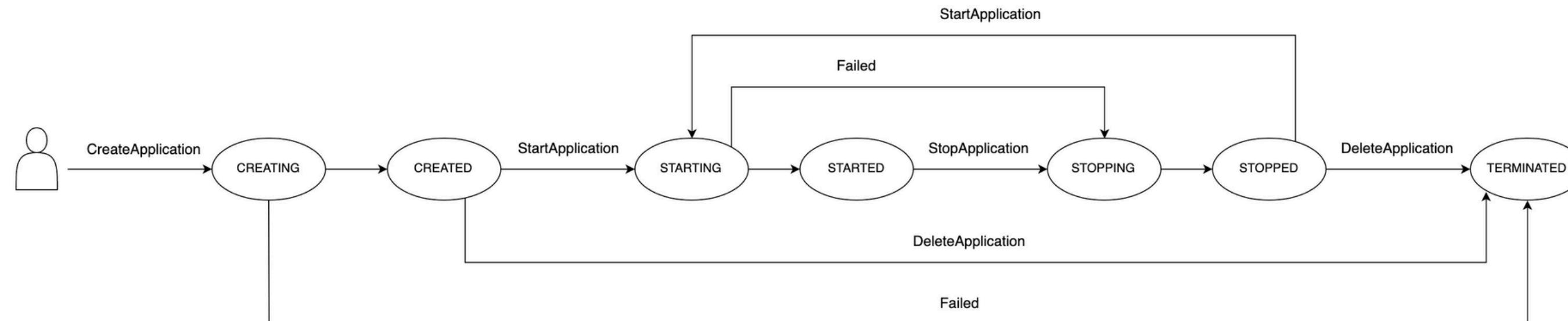
The local file system refers to a locally connected disk.



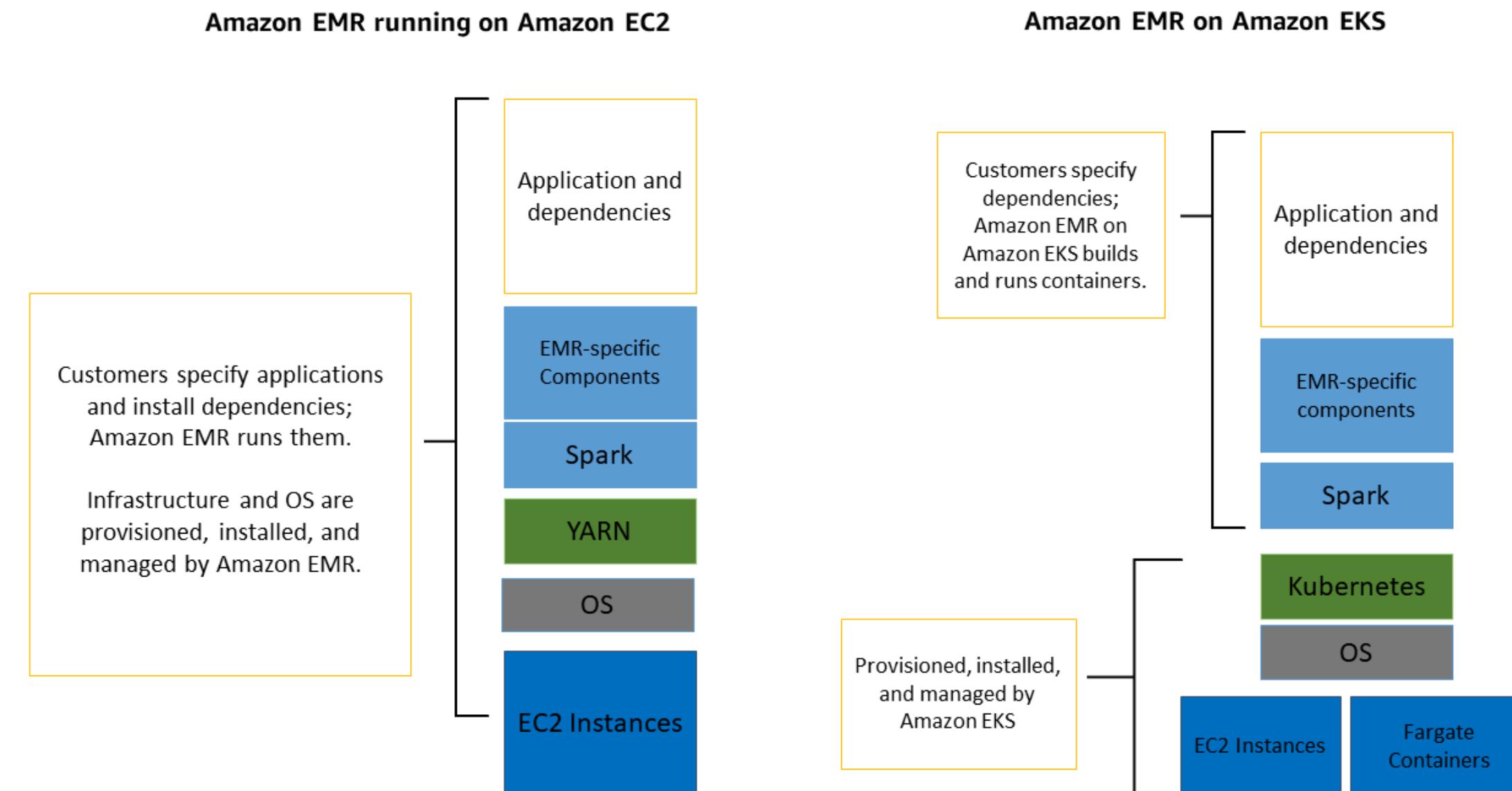
# EMR SERVERLESS

Provisions EMR in a severless environment

Supports Hive, Spark, Presto

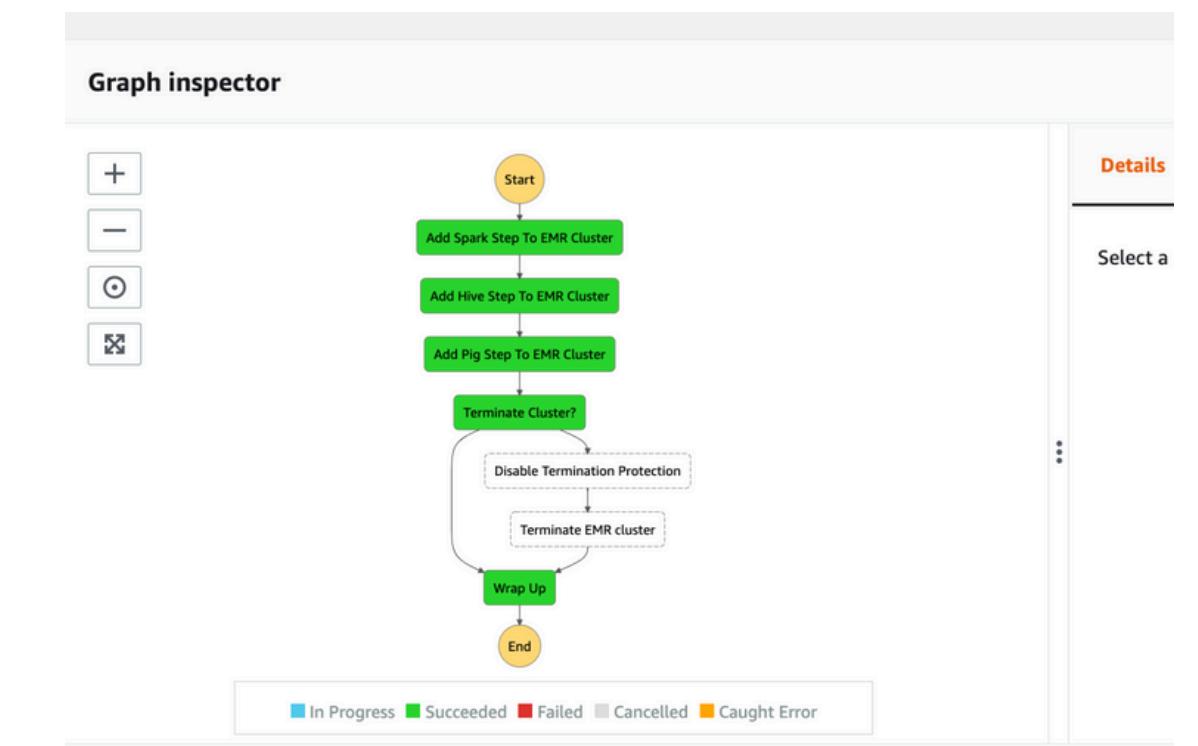


# EMR on EKS



# Step Function Orchestration

**AWS Step Functions is a low-code, visual workflow service that developers use to build distributed applications, automate IT and business processes, and build data and machine learning pipelines using AWS services. Workflows manage failures, retries, parallelization, service integrations, and observability so developers can focus on higher-value business logic.**



# Amazon EMR and Data Engineering



# Amazon EMR and Data Engineering

Big Data Processing For Ingestion

Big Data Processing for Analytics

Big Data Steaming



# Amazon Kinesis



# Amazon Kinesis

**Real Time Streaming Solution which allows you to ingest, buffer, and process data**

**Fully managed and runs your streaming applications without requiring you to manage any infrastructure.**

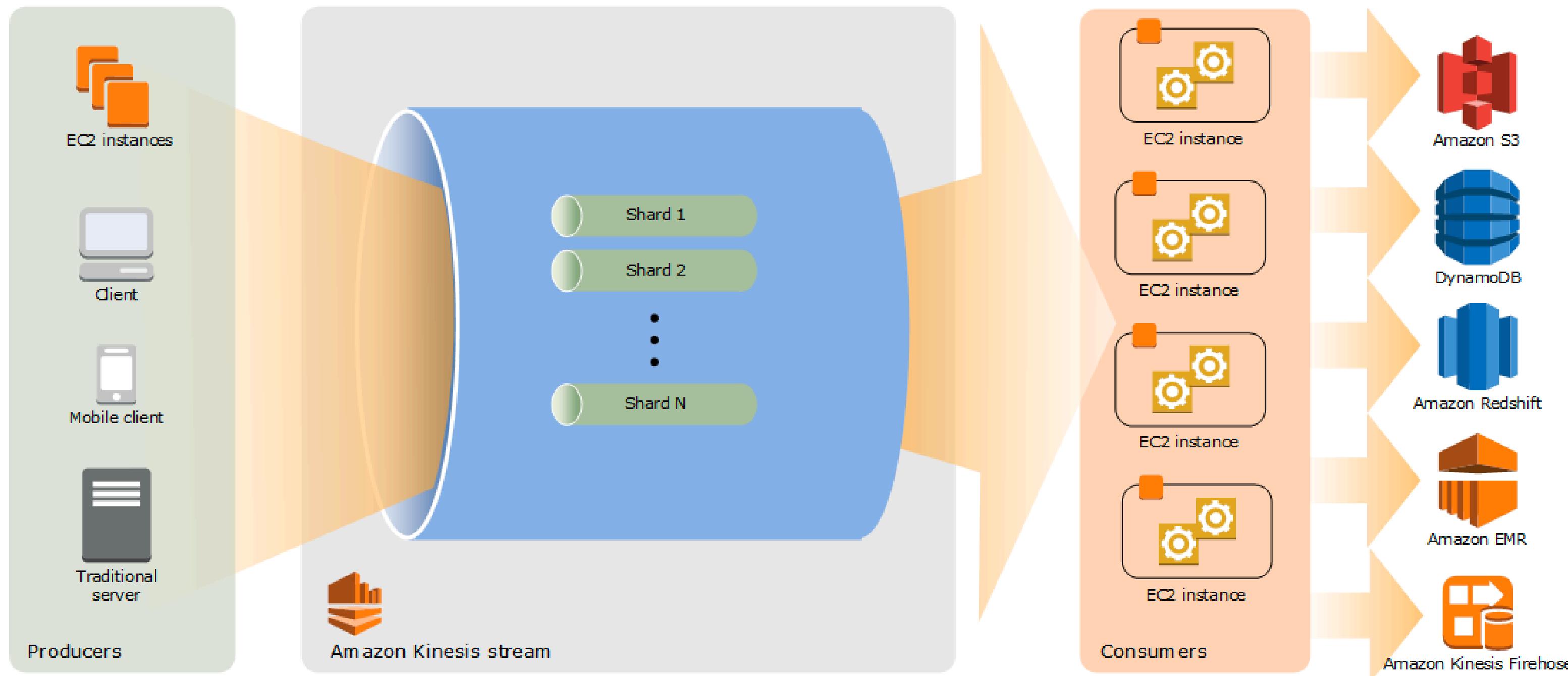
**Amazon Kinesis is scalable. It can handle any amount of streaming data and process data from hundreds of thousands of sources with very low latencies.**



# Amazon Kinesis Data Streams



# Amazon Kinesis Data Streams



# Amazon Kinesis Data Streams

**Data Stream:** A Kinesis data stream is a set of shards. Each shard has a sequence of data records. Each data record has a sequence number that is assigned by Kinesis Data Streams.

**Data Record:** A data record is the unit of data stored in a Kinesis data stream. Data records are composed of a sequence number, a partition key, and a data blob, which is an immutable sequence of bytes. Kinesis Data Streams does not inspect, interpret, or change the data in the blob in any way. A data blob can be up to 1 MB.

**Partition Key:** A partition key is used to group data by shard within a stream. Kinesis Data Streams segregates the data records belonging to a stream into multiple shards. It uses the partition key that is associated with each data record to determine which shard a given data record belongs to.



# Amazon Kinesis Data Streams ctn..

**Sequence Number:** Each data record has a sequence number that is unique per partition-key within its shard. Kinesis Data Streams assigns the sequence number after you write to the stream with `client.putRecords` or `client.putRecord`.

**Producers:** Producers put records into Amazon Kinesis Data Streams. For example, a web server sending log data to a stream is a producer.

**Consumers:** Consumers get records from Amazon Kinesis Data Streams and process them. These consumers are known as Amazon Kinesis Data Streams Application.



# Amazon Kinesis Data Streams KCL

The **Kinesis Consumer Library (KCL)** takes care of many complex tasks associated with distributed processing and allows you to focus on the record processing logic including;

- Connects to the data stream
- Enumerates the shards within the data stream
- Uses leases to coordinate shard associations with its workers
- Instantiates a record processor for every shard it manages
- Pulls data records from the data stream
- Pushes the records to the corresponding record processor
- Checkpoints processed records
- Balances shard-worker associations (leases) when the worker instance count changes or when the data stream is resharded (shards are split or merged)

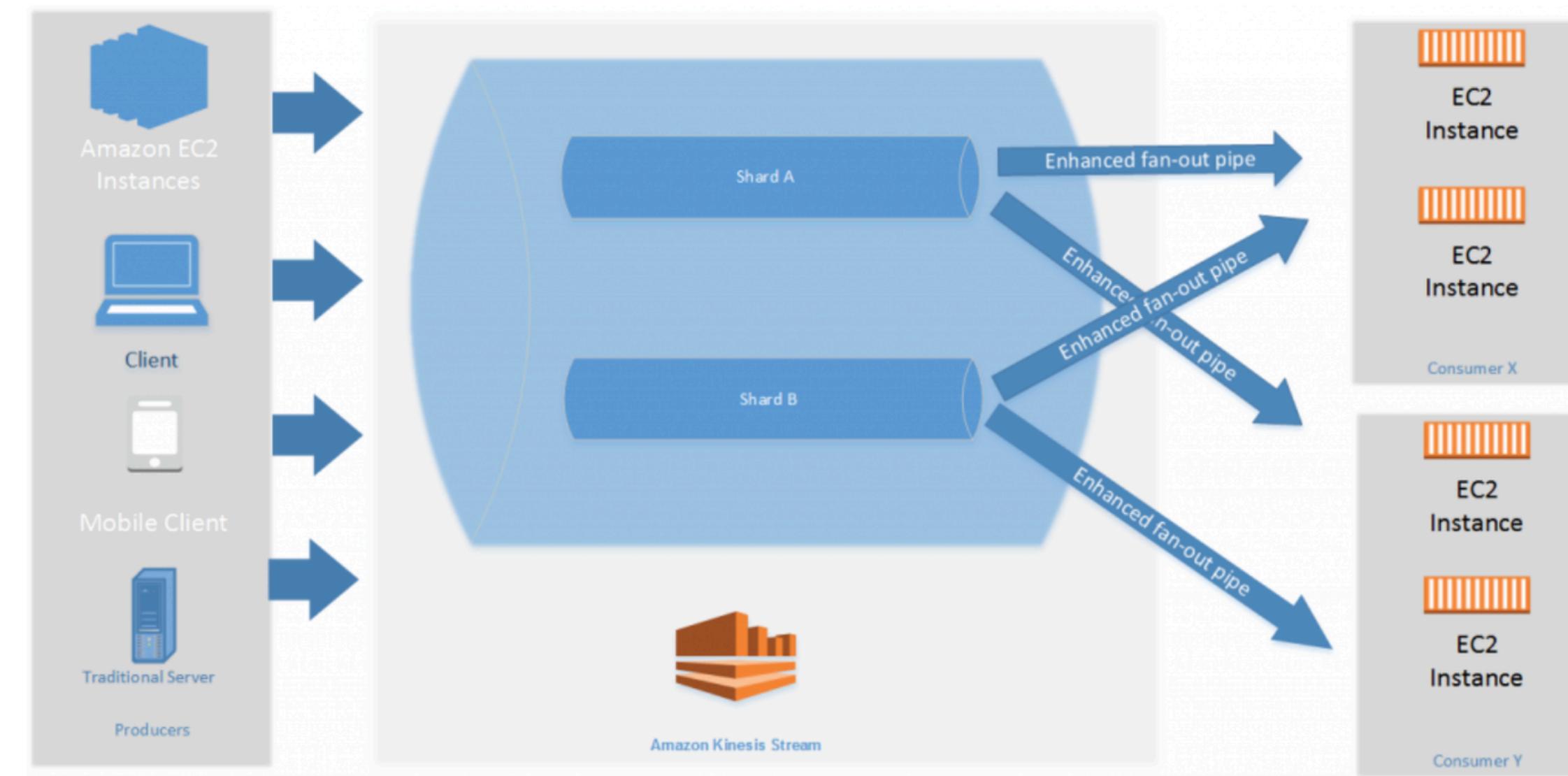


# Amazon Kinesis Data Streams ctn..

SDK	KPL
The SDK can call the 'PutRecord' and 'PutRecords' api	The KPL can write to one or more Kinesis Data Streams with automatic and configurable retries
PutRecord operation allows a single data record within an API call	Collects records and utilizes PutRecords API to write multiple records to multiple shards per request
PutRecords operation allows multiple data records within an API call up to 500 records	Aggregate records to increase payload size and throughput
Each record in the request can be as large as 1 MiB, up to a limit of 5 MiB	Only available in Java



# Kinesis Data Streams Enhanced Fanout

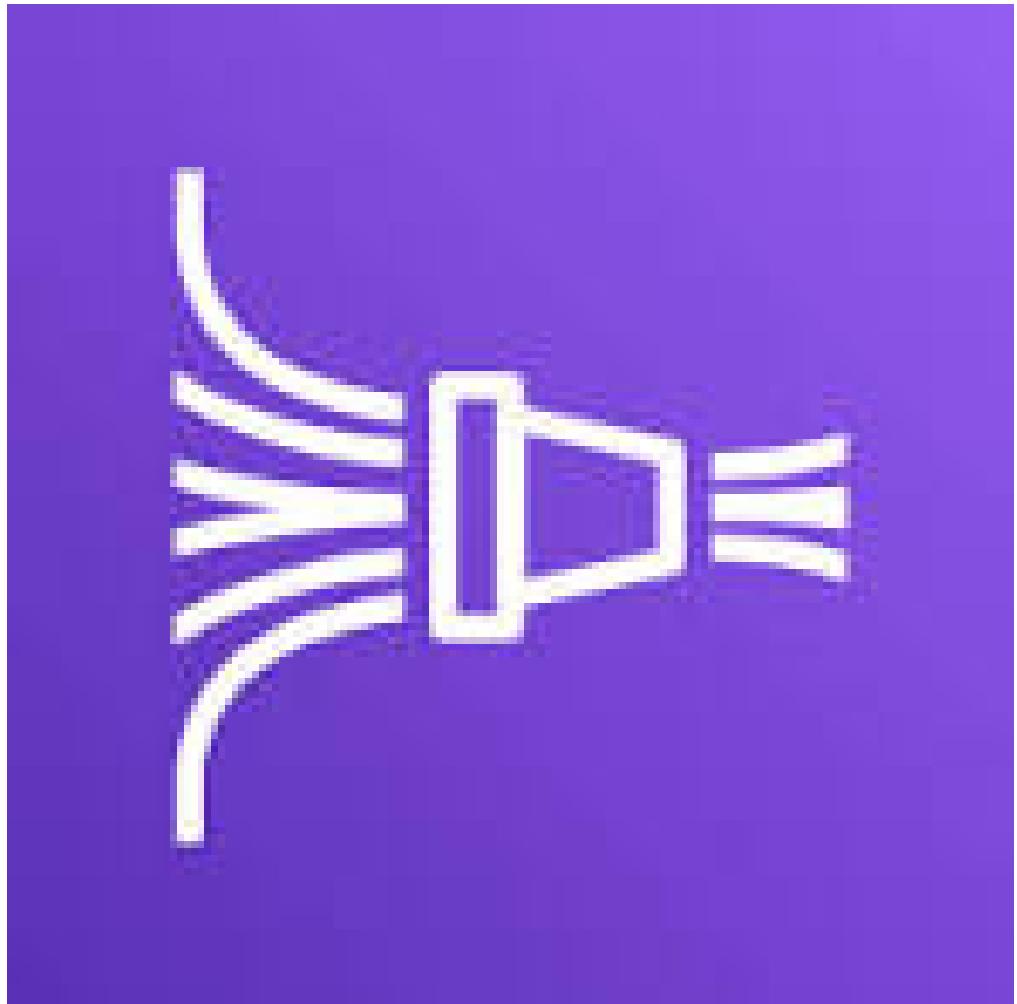


# Amazon Kinesis FireHose



# Kinesis FireHose

**A Fully managed service for delivering real-time streaming data to destinations such as Amazon Simple Storage Service (Amazon S3), Amazon Redshift, Amazon OpenSearch Service...**



# **Amazon Managed Service For Apache Flink**



# Amazon Managed Service For Apache Flink

**Transform and analyze streaming data in  
real time with Apache Flink**

**The service handles the underlying  
infrastructure for your Apache Flink  
applications**



# Amazon MSK



# Amazon MSK

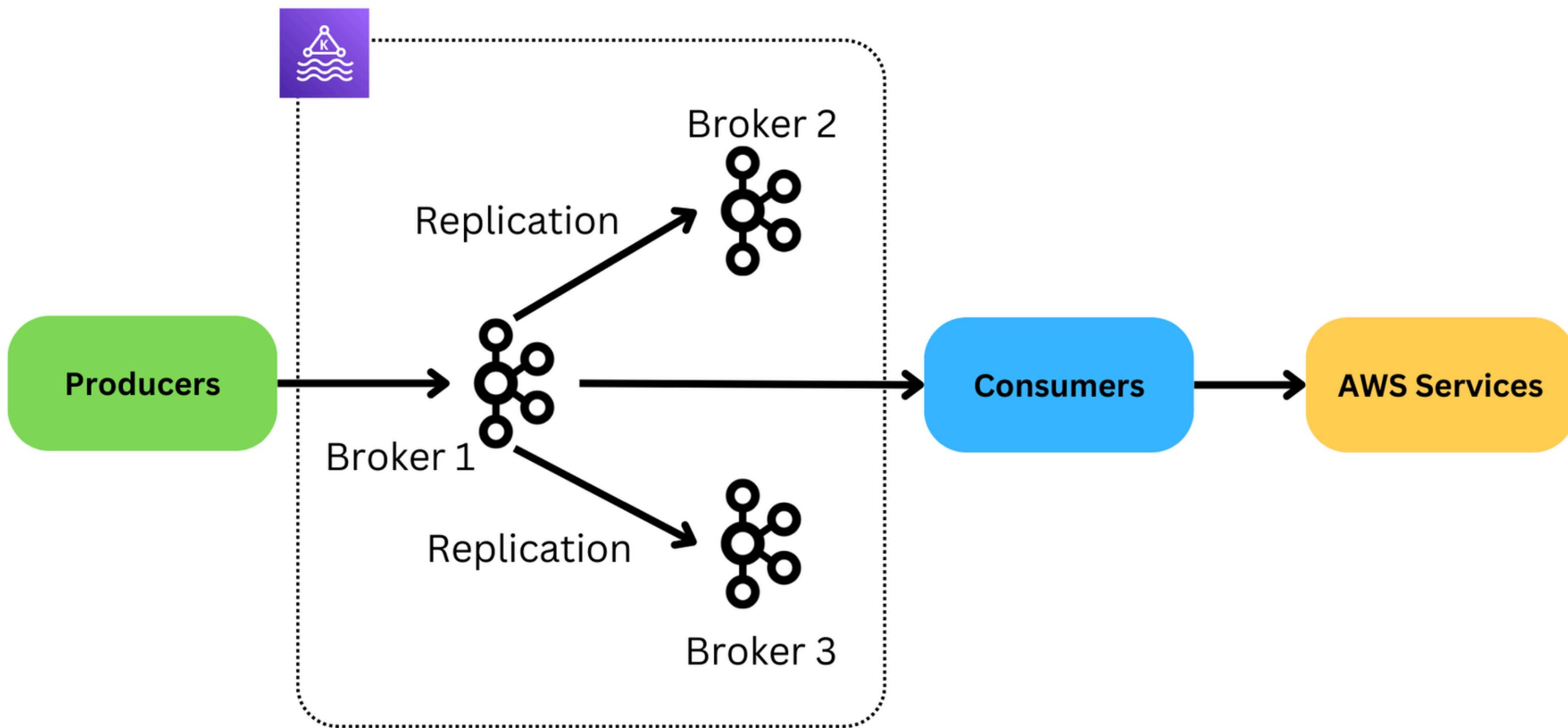
**Amazon MSK manages the Apache Kafka infrastructure and operations**

**It provides the control-plane operations for creating, updating, and deleting Kafka clusters**

**Amazon MSK runs open-source versions of Apache Kafka**



# Amazon MSK Architecture



# Amazon MSK Vs Kinesis



# Amazon MSK Vs Kinesis

MSK	Kineses Data Streams
Up to 10MB	1 MB message Size
Kafka topics with partitions	Data Streams with Shards
Add partitions to a topic only	Shard splitting and merging
Plain text of TLS	TLS Inflight encrytpion
KMS at rest encyption	KMS at rest
Security: Mutual TLS & KAFKA ACLs, SASL/SCRAM & KAFKA ACLS, IAM	Security: IAM



# Amazon OpenSearch



# What is Amazon OpenSearch?

OpenSearch is an open-source, distributed search, Visualization, and analytics engine, which is a fork of the popular Elasticsearch and Kibana software.

Amazon OpenSearch Service is a managed service for deploying, operating, and scaling OpenSearch clusters.



# OpenSearch Use Cases

Log and Event Data Analysis

Full-Text Search

Real-Time Application Monitoring

Business Analytics



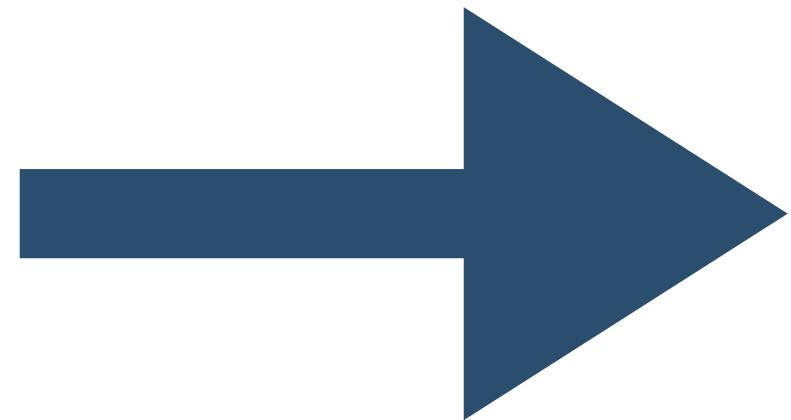
# OpenSearch Concepts

## Documents

A document is the basic unit of data that can be indexed

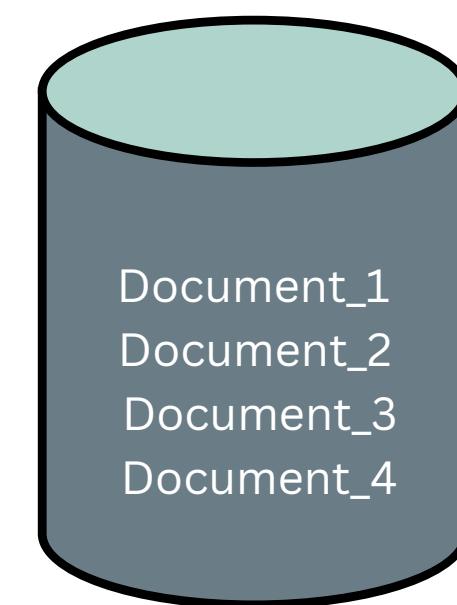
A document is represented in JSON

```
[{"director": "Baird, Stuart",  
 "genre": [  
     "Action",  
     "Crime",  
     "Thriller"  
 ],  
 "year": 1998,  
 "actor": [  
     "Downey Jr., Robert",  
     "Jones, Tommy Lee"  
 ],  
 "title": "U.S. Marshals"}]
```



## Indices

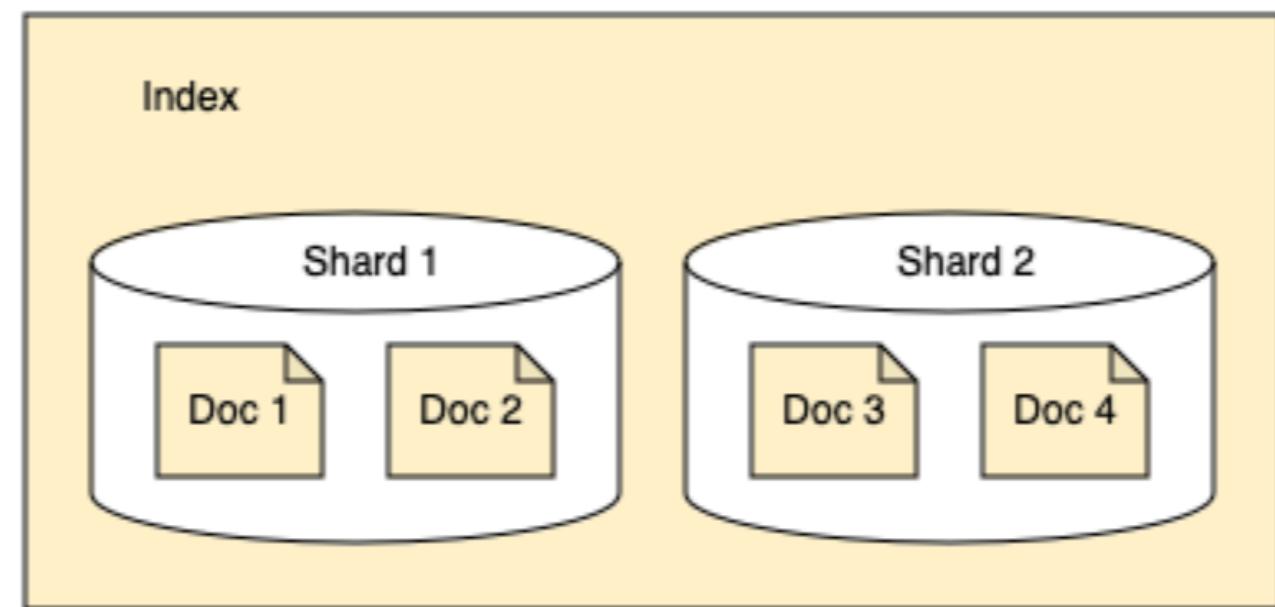
An index is a logical namespace that stores a collection of documents



# OpenSearch Indexes

An Index is made up of shards

Documents are hashed onto a shard



## Intro to OpenSearch

Introduction to OpenSearch

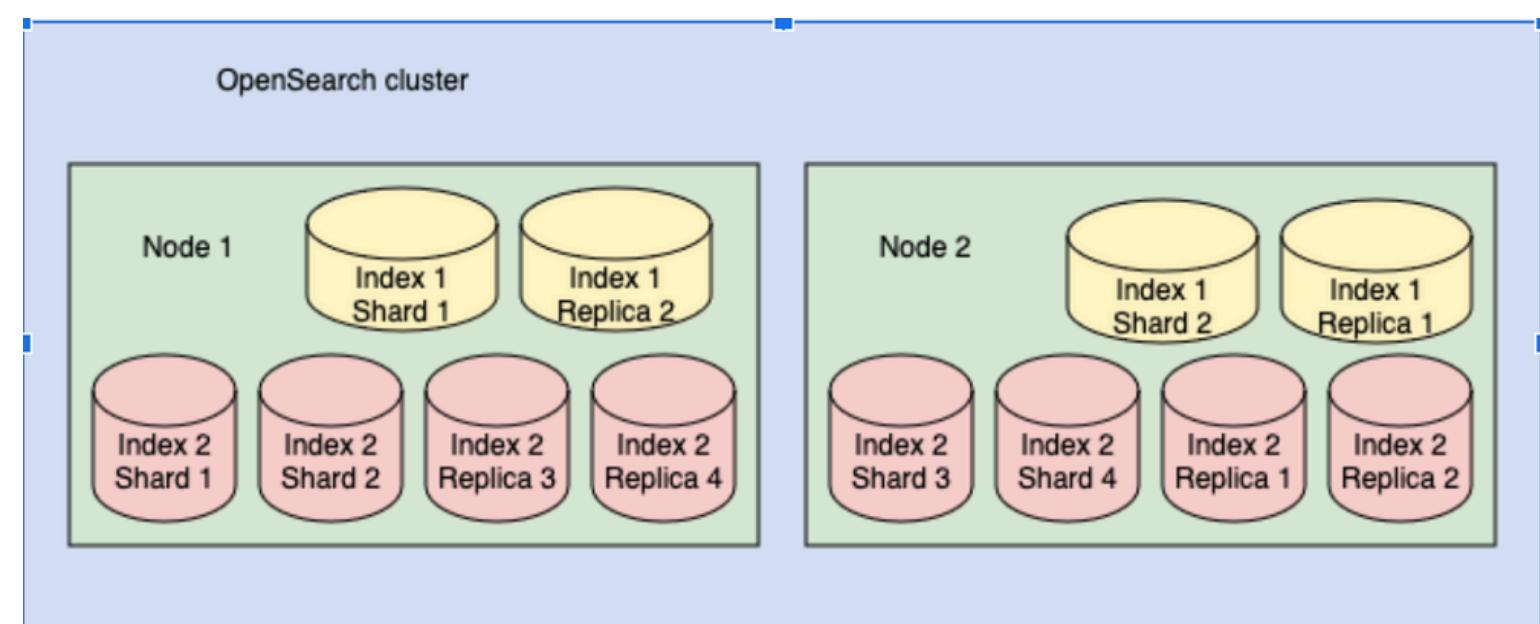
 OpenSearch Documentation / Aug 20



# Primary Shards and Replicas

**Primary shards are the main shards that contain the original data of an index**

**Replica shards are exact copies of primary shards. They serve two main purposes: providing high availability and load balancing for search operations**



# **Calculate Storage Capacity for an Index**

**Size of Source Data \* (1+ Number of replicas) \* 1.45**



# JVM Pressure

**Typically, JVM pressure on a cluster is caused when the shards are unbalanced, and/or there are too many shards. To fix this issue a user should off load data which is not being used to S3 or other storage services, and delete indices which are not being used.**



# Amazon QuickSight



# What is Amazon Quicksight?

A cloud-based business intelligence (BI) service

Create and publish interactive dashboards



# Quicksight Use Cases

**Business Reporting**

**Data Exploration**

**Operational Analytics**

**Customer Insights**



# Key Features

**Data Visualization and Dashboards**

**Machine Learning Insights**

**Interactive Exploration**

**Access and Control**

**Scalable and Serverless**

**Data Connectivity**

**SPICE Engine**

**Mobile Access**



# Integrations

**Amazon Redshift**

**On premise databases**

**Amazon RDS**

**SaaS applications**

**Amazon Athena**

**Cloud Database eg Snowflake**

**Amazon S3**

**Flat file uploads**



# Accessing Quicksight using a VPC

Create a VPC Endpoint for Quicksight

Configure Security Groups

Enable VPC Access in QuickSight



# Amazon DynamoDB



# Amazon DynamoDB Intro



# Amazon DynamoDB

**Fully managed Serverless NoSQL database.**

**Can handle large volumes of data and high request rates**

**Manage data in a key-value and document format**

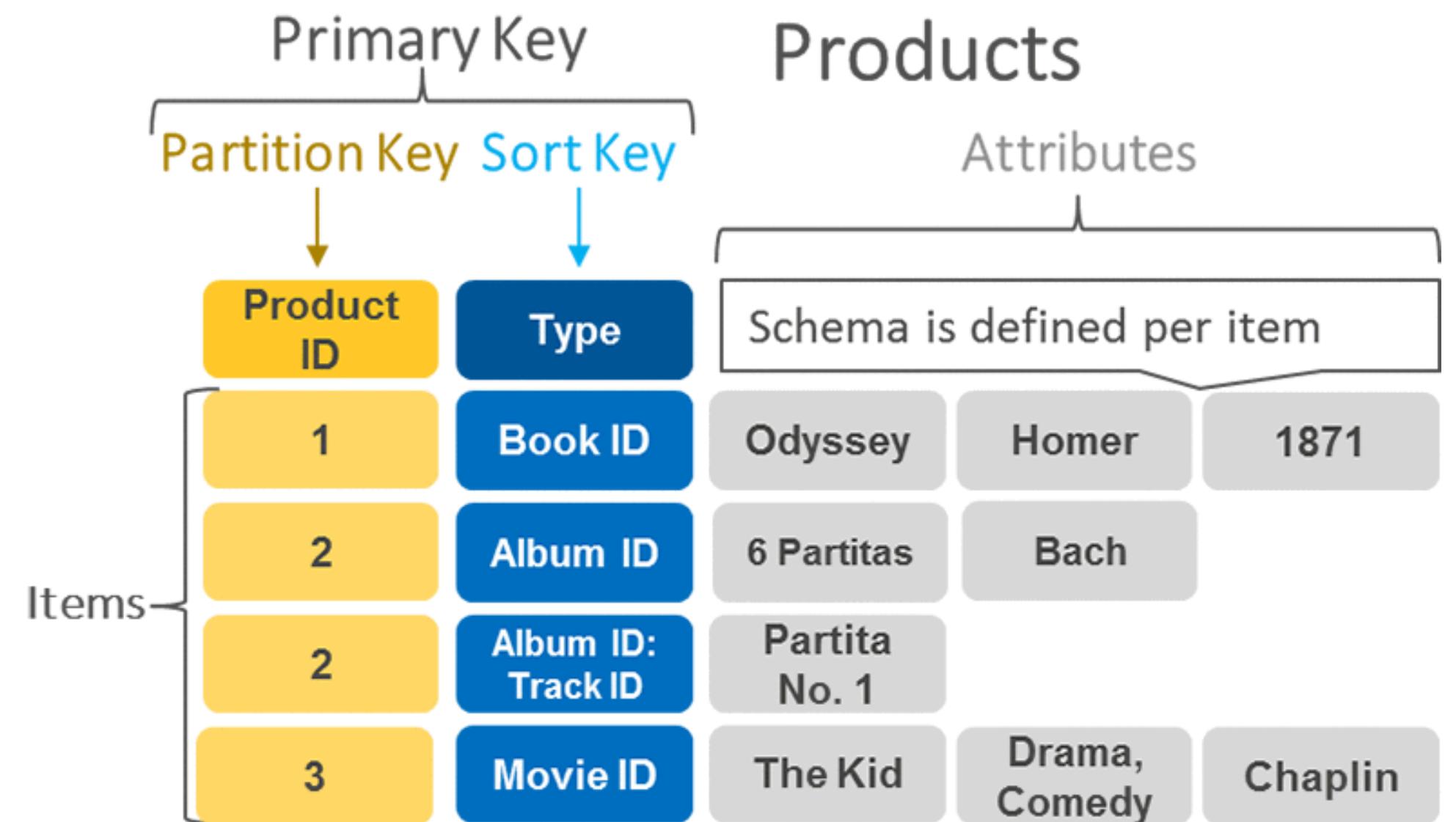


# Partition Keys, Sorts Keys, and Primary Keys

The Partition Key is a unique identifier for an item

The Sort Key allows you to store multiple items with the same partition key

Attributes are basic units of information



# **Amazon DynamoDB Indexes**



# Amazon DynamoDB Indexes

**Global Secondary Indexes (GSI) and Local Secondary Indexes (LSI)** are mechanisms to create alternative query patterns on your data.

Global Secondary Indexes allow you to create an index with a partition key and an optional sort key that can be different from those used in the table's primary key.

Local Secondary Indexes allow you to create an index with the same partition key as the base table but a different sort key. LSIs are "local" because the index is scoped to items that share the same partition key.



# Choosing Between GSI and LSI

- Use GSI if:
  - You need a different partition key for your query patterns.
  - You want to scale the throughput independently from the base table.
  - You require indexing across the entire table, without the constraints of a shared partition key.
- Use LSI if:
  - You want to query on a different attribute using the same partition key.
  - You need consistent reads and want to avoid the additional costs of maintaining a separate index.



# DynamoDB Read Capacity Units (RCU) and Write Capacity Units (WRU)

**Read Capacity Units (RCUs) and Write Capacity Units (WCUs)** are used to measure the throughput capacity required for read and write operations on your tables. These units help manage the performance and cost of database operations by allowing you to specify and pay for the level of throughput you require.



# DynamoDB Read Capacity Units (RCU)

A Read Capacity Unit (RCU) represents one strongly consistent read per second, or two eventually consistent reads per second, for an item up to 4 KB in size.

## Examples

- Strongly Consistent Read of an Item 4 KB or Less:
  - Consumes 1 RCU.
  - Example: Reading an item of size 3 KB using strongly consistent read consumes 1 RCU.
- Eventually Consistent Read of an Item 4 KB or Less:
  - Consumes 0.5 RCU.
- Reading an item of size 2 KB using eventually consistent read consumes 0.5 RCU

Reading Larger Items: Each RCU can handle an item up to 4 KB. If the item is larger, it consumes additional RCUs.



# DynamoDB Write Capacity Units

A Write Capacity Unit (WCU) represents one write per second for an item up to 1 KB in size.

## Examples

- Writing an Item 1 KB or Less:
  - Consumes 1 WCU.
- Writing an item of size 500 bytes
  - Consumes 1 WCU.

**Writing Larger Items:** Each WCU can handle an item up to 1 KB. If the item is larger, it consumes additional WCUs.

## Examples

- Writing a 3 KB item would consume 3 WCUs ( $3 \text{ KB} / 1 \text{ KB} = 3$ ).



# DynamoDB Modes

## Provisioned Mode

You specify the number of RCUs and WCUs needed based on expected load. This mode is cost-effective when you have predictable traffic patterns.

## On-Demand Mode

DynamoDB automatically adjusts capacity to accommodate workloads as they scale up or down. You pay for the actual read and write requests you use, making it ideal for unpredictable or spiky workloads.



# PartiQI

DynamoDB supports PartiQL, which is a SQL-compatible query language that allows used to select, insert, update, and delete data in Amazon DynamoDB. PartiQL provides SQL-compatible query access across multiple data stores containing structured data, semistructured data, and nested data.



# DynamoDB Accelerator (DAX)

**Fully managed, in-memory caching service for DynamoDB.**  
**It significantly improves the performance of read-intensive applications by providing fast in-memory access to DynamoDB tables, reducing read response times from milliseconds to microseconds, even at millions of requests per second.**

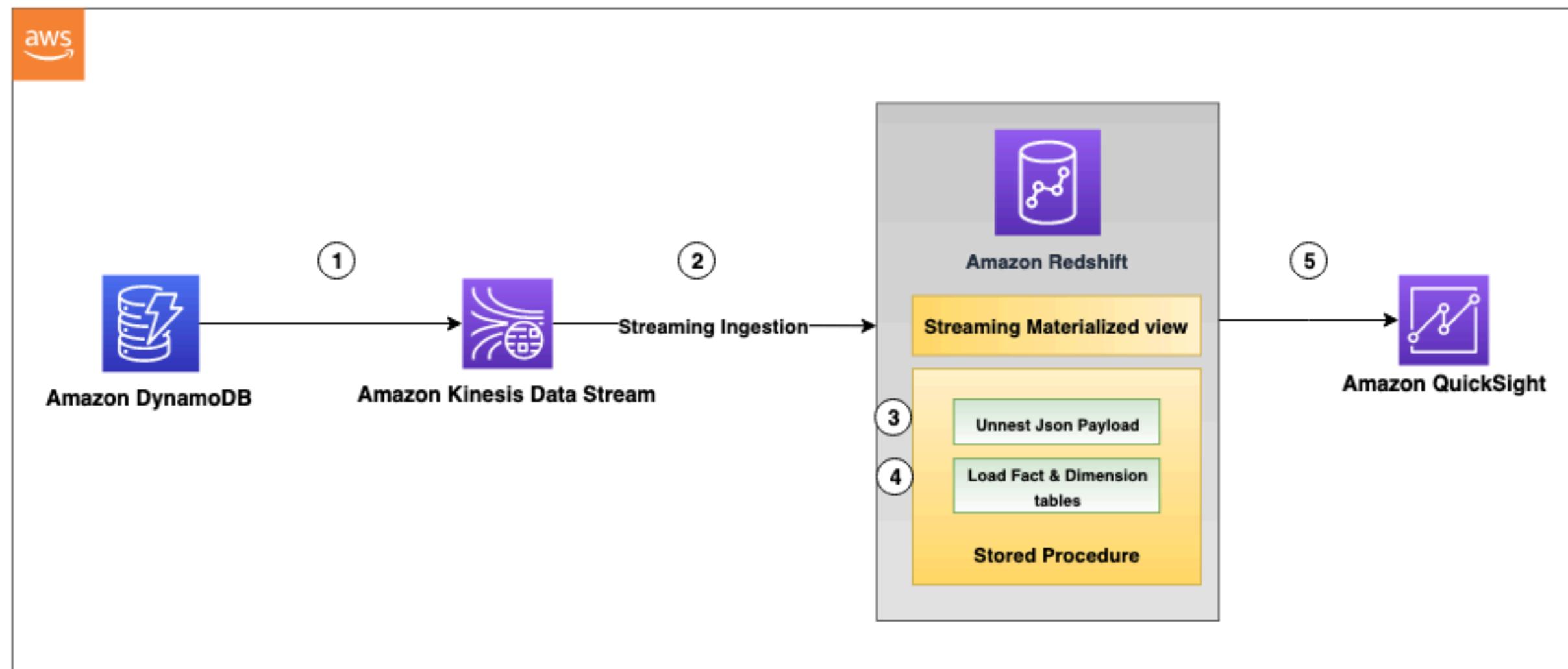


# DynamoDB Streams

**DynamoDB Streams** is a feature of Amazon DynamoDB that captures a time-ordered sequence of item-level changes (such as inserts, updates, and deletes) in a DynamoDB table. When enabled on a table, DynamoDB Streams records these changes and provides them as a stream of data records, which can be consumed by various applications and services for further processing.



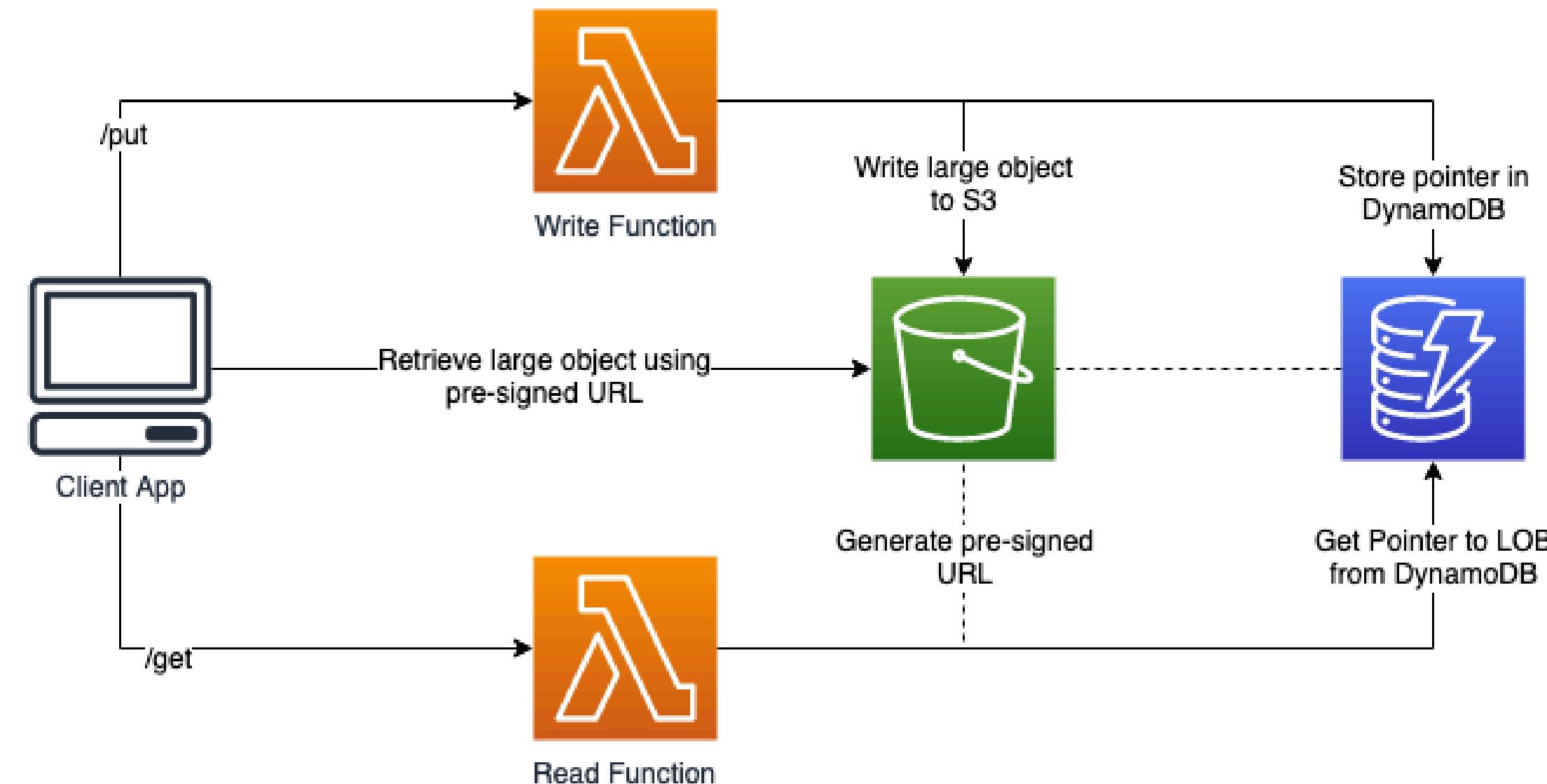
# DynamoDB Data Engineering



<https://aws.amazon.com/blogs/big-data/near-real-time-analytics-using-amazon-redshift-streaming-ingestion-with-amazon-kinesis-data-streams-and-amazon-dynamodb/>



# DynamoDB Data Engineering and S3 Patterns



<https://aws.amazon.com/blogs/database/large-object-storage-strategies-for-amazon-dynamodb/> <https://aws.amazon.com/blogs/database/amazon-dynamodb-can-now-import-amazon-s3-data-into-a-new-table/>



# Amazon Lambda



# Amazon Lambda

**AWS Lambda is a serverless computing service. It allows developers to run code without provisioning or managing servers. With AWS Lambda, you can execute code in response to events.**



# Amazon Lambda For Data Engineering



# Lambda Data Engineering Use Cases

Real-time Data Ingestion

Database Replication

Batch Processing

Log Processing

Data Transformation

Orchestration

Trigger-Based Processing



# Benefits of Using AWS Lambda for Data Engineering

Scalability

Cost Efficiency

Reduced Operational Overhead

Integration with AWS Ecosystem

Event-Driven



# Containers



# **What is a container?**

**A lightweight, standalone, and executable package that includes everything needed to run a piece of software: code, runtime, system tools, libraries, and settings.**



# **Containers and AWS**



# AWS Container Services

Amazon Elastic Container Service (ECS)

AWS Fargate

Amazon Elastic Kubernetes Service (EKS)



# AWS Container Services

Amazon Elastic Container Service (ECS)

AWS Fargate

Amazon Elastic Kubernetes Service (EKS)



# **Containers on AWS And Data Engineering**



# Benefits of Using Containers for Data Engineering

Portability

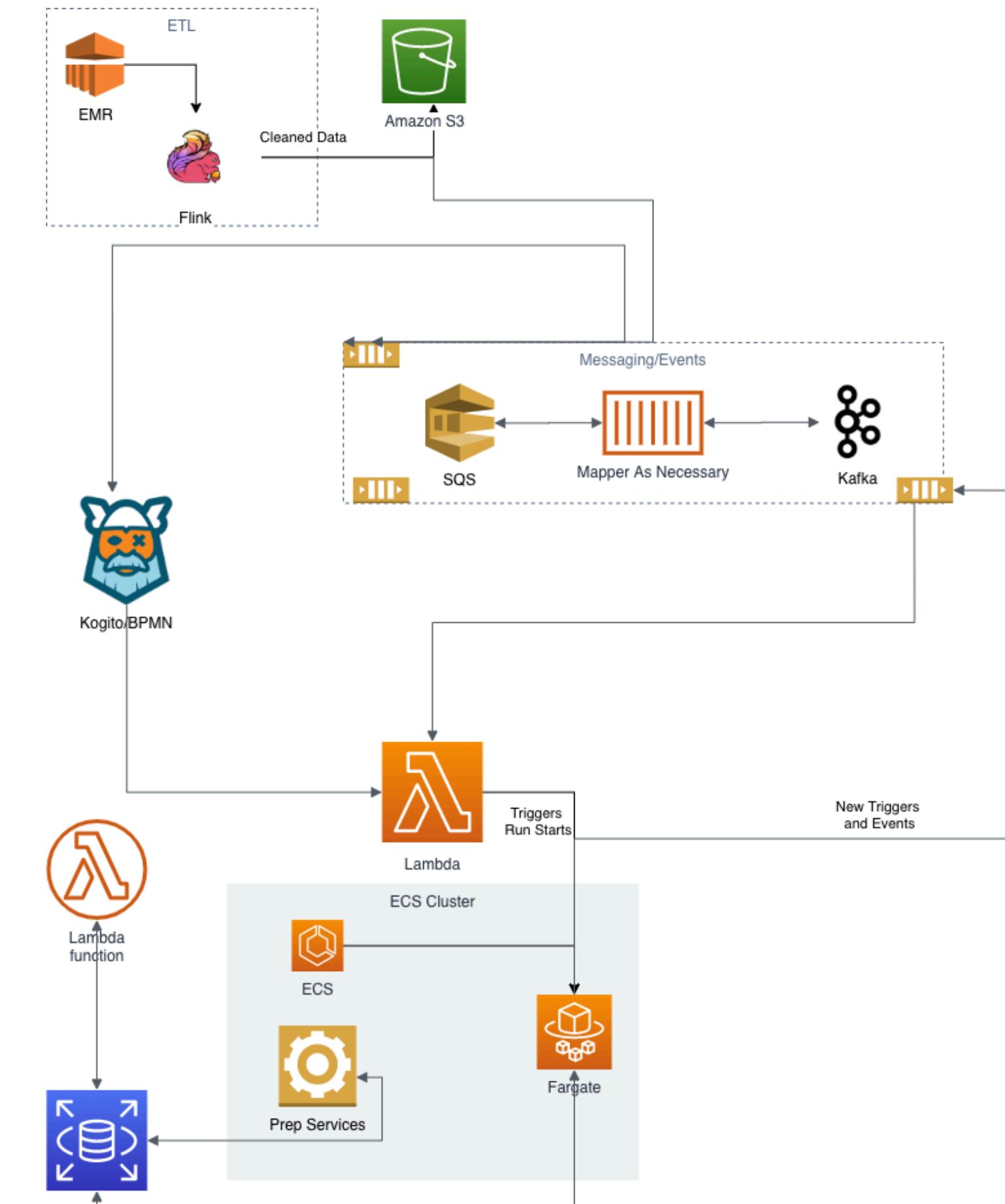
Efficiency

Scalability

Isolation



- 1. Cheaper**
- 2. More reliable**
- 3. Easy debugging**
- 4. Suited for scale**



<https://aws.amazon.com/blogs/containers/how-taloflow-saved-60-by-moving-their-data-pipeline-to-aws-fargate-spot/>



# Amazon S3



# Amazon S3

**Amazon S3 (Simple Storage Service) is a scalable, high-speed, web-based cloud storage service.**

**It is designed for storing and retrieving any amount of data at any time, from anywhere on the web.**



**99.99% Availability and 99.999999999%**  
**Durability**



# S3 and Data Engineering

**Data Ingestion and Storage**

**Data Processing and Transformation**

**Data Warehousing and Analytics**

**Machine Learning and Data Science**

**Data Backup and Archiving**

**Event-Driven Data Processing**

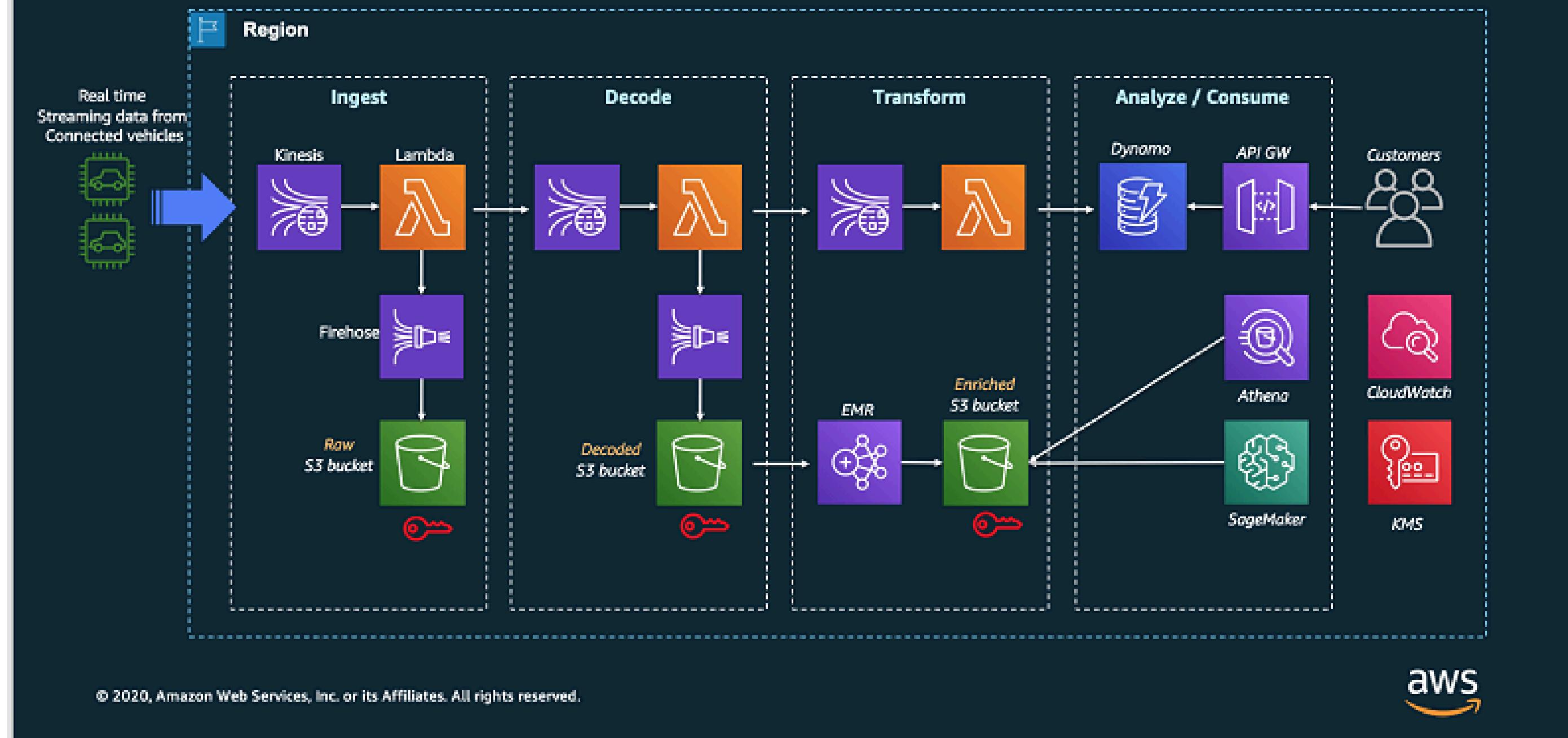


# S3 and Data Lakes

A data lake is a centralized repository that allows you to store all your structured and unstructured data at any scale. With a data lake, you can store your data as-is, without having to first structure the data, and run different types of analytics—from dashboards and visualizations to big data processing, real-time analytics, and machine learning—to guide better decisions.



# Toyota Connected Data Lake Architecture



<https://aws.amazon.com/blogs/big-data/enhancing-customer-safety-by-leveraging-the-scalable-secure-and-cost-optimized-toyota-connected-data-lake/>



# S3 LifeCycle Policy

**S3 Lifecycle policies allow you to manage the lifecycle of objects stored in Amazon S3. These policies define actions that Amazon S3 can perform on your objects during their lifetime, such as transitioning them to a different storage class or automatically deleting them. Lifecycle policies help optimize storage costs and manage data according to business requirements.**



# Amazon Lakeformation

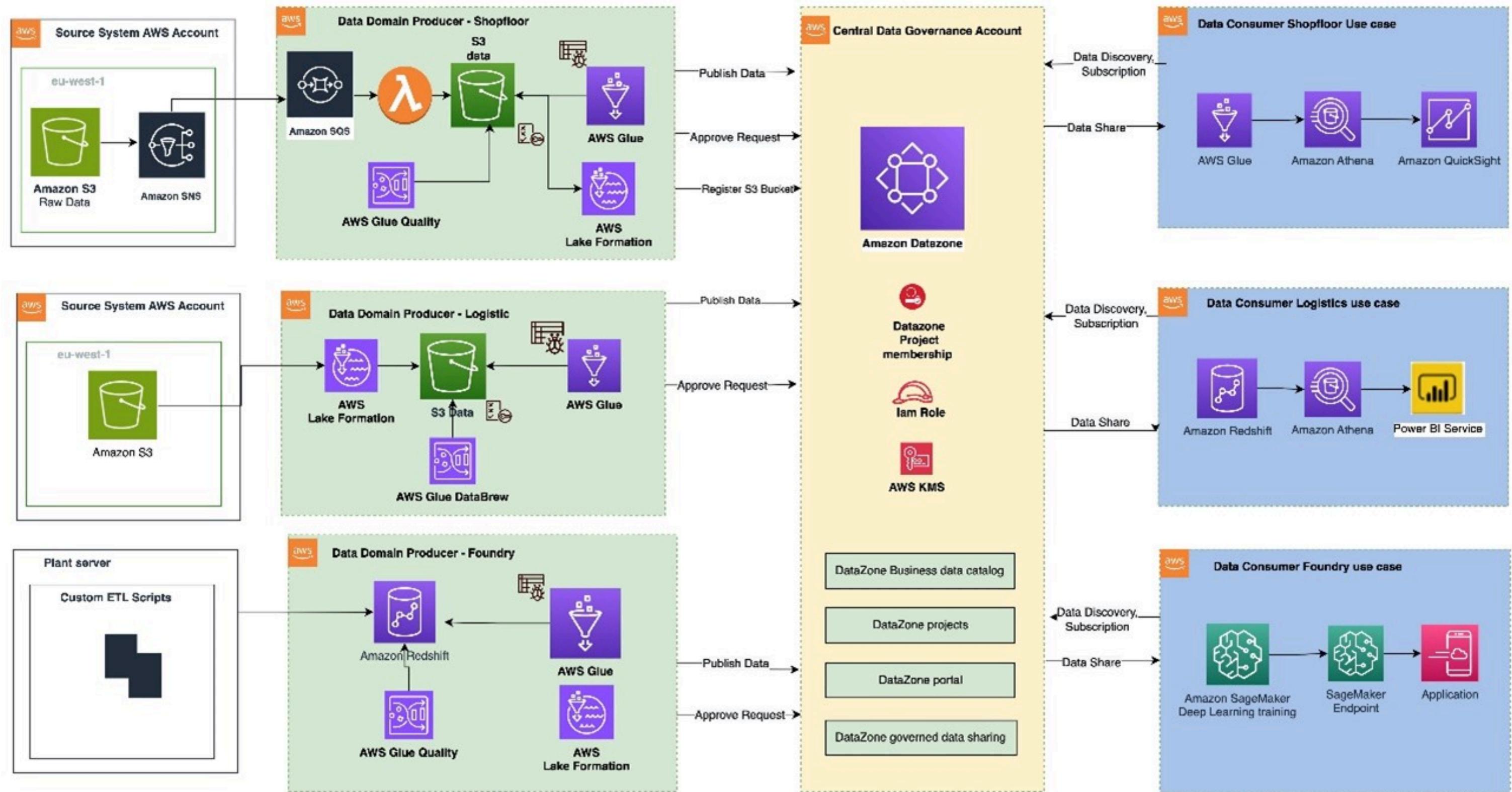


# Amazon Lakeformation

**Simplifies the process of setting up a secure data lake.**

**Automates the steps involved in setting up a data lake, including data ingestion, storage, cataloging, transformation, and security**





<https://aws.amazon.com/blogs/big-data/how-volkswagen-streamlined-access-to-data-across-multiple-data-lakes-using-amazon-datazone-part-1/>



# More Amazon Databases

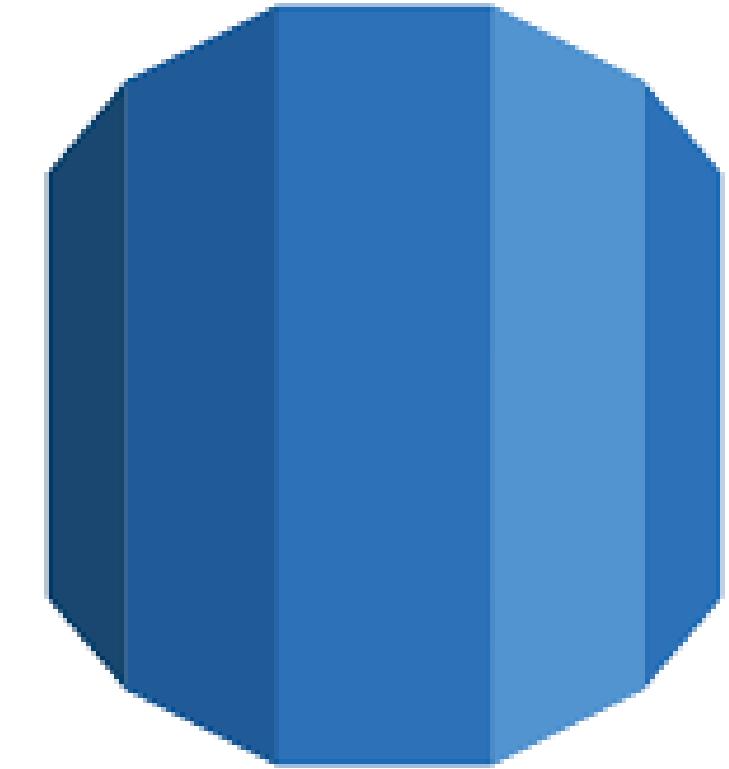


# Amazon RDS



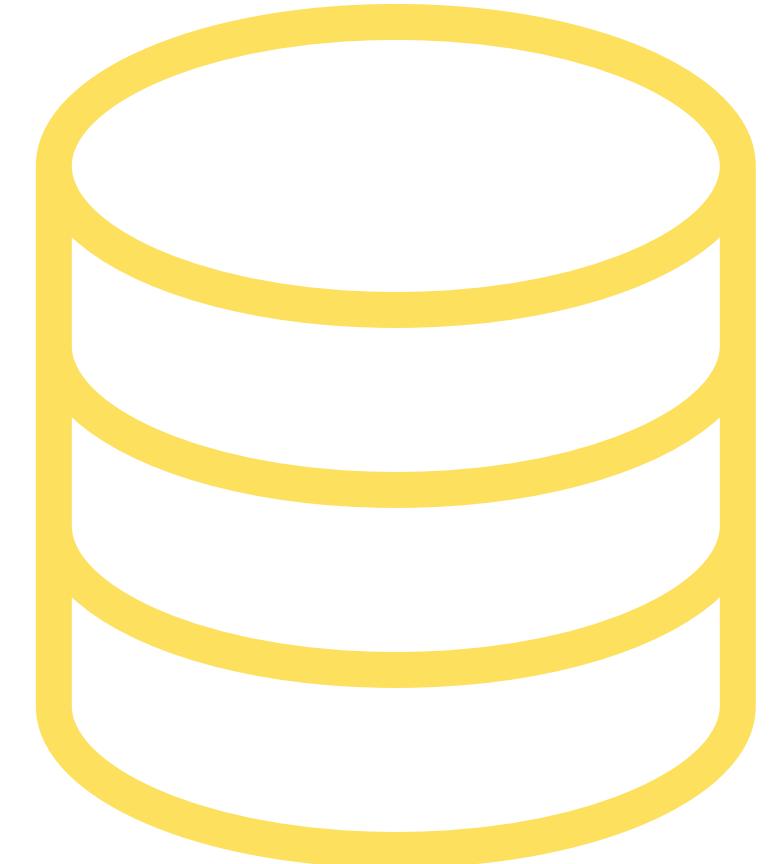
# Amazon RDS

**Amazon RDS (Relational Database Service) is a managed database service provided by Amazon Web Services (AWS) that simplifies the process of setting up, operating, and scaling relational databases in the cloud.**



# Amazon RDS Engines

- Amazon Aurora (compatible with MySQL and PostgreSQL)
- MySQL
- PostgreSQL
- MariaDB
- Oracle
- Microsoft SQL Server



# Amazon Timestream



# Amazon Timestream

**Amazon Timestream is a fully managed, scalable, and serverless time series database service. It is designed specifically for storing and analyzing time series data.**

**Common use cases for time series data include monitoring and observability, Internet of Things (IoT) applications, industrial telemetry, and real-time analytics.**



# Amazon Neptune



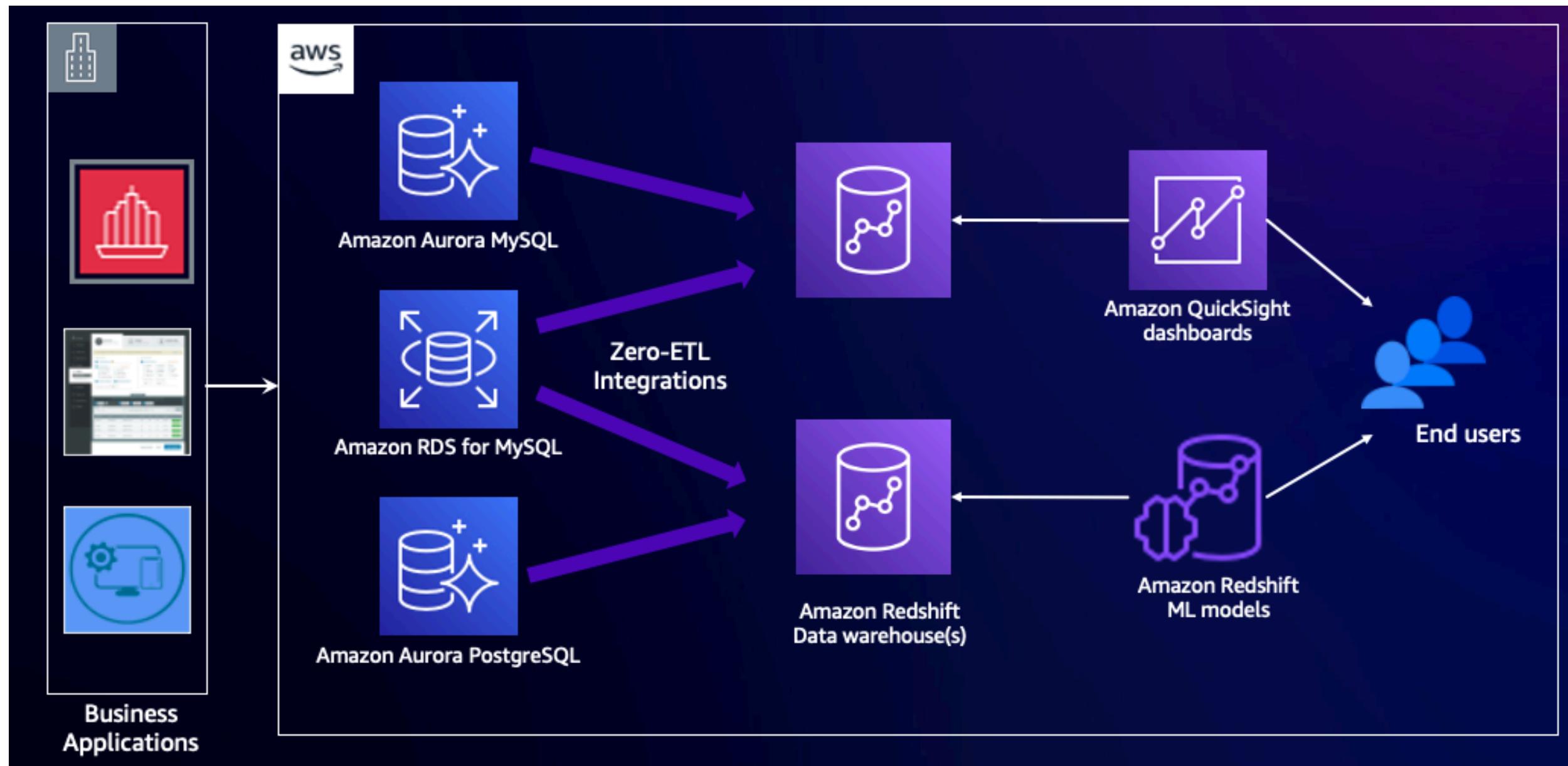
# Amazon Neptune

**Amazon Neptune** is a fully managed graph database service. It is designed for applications that require highly connected datasets and can efficiently store and navigate relationships between data.



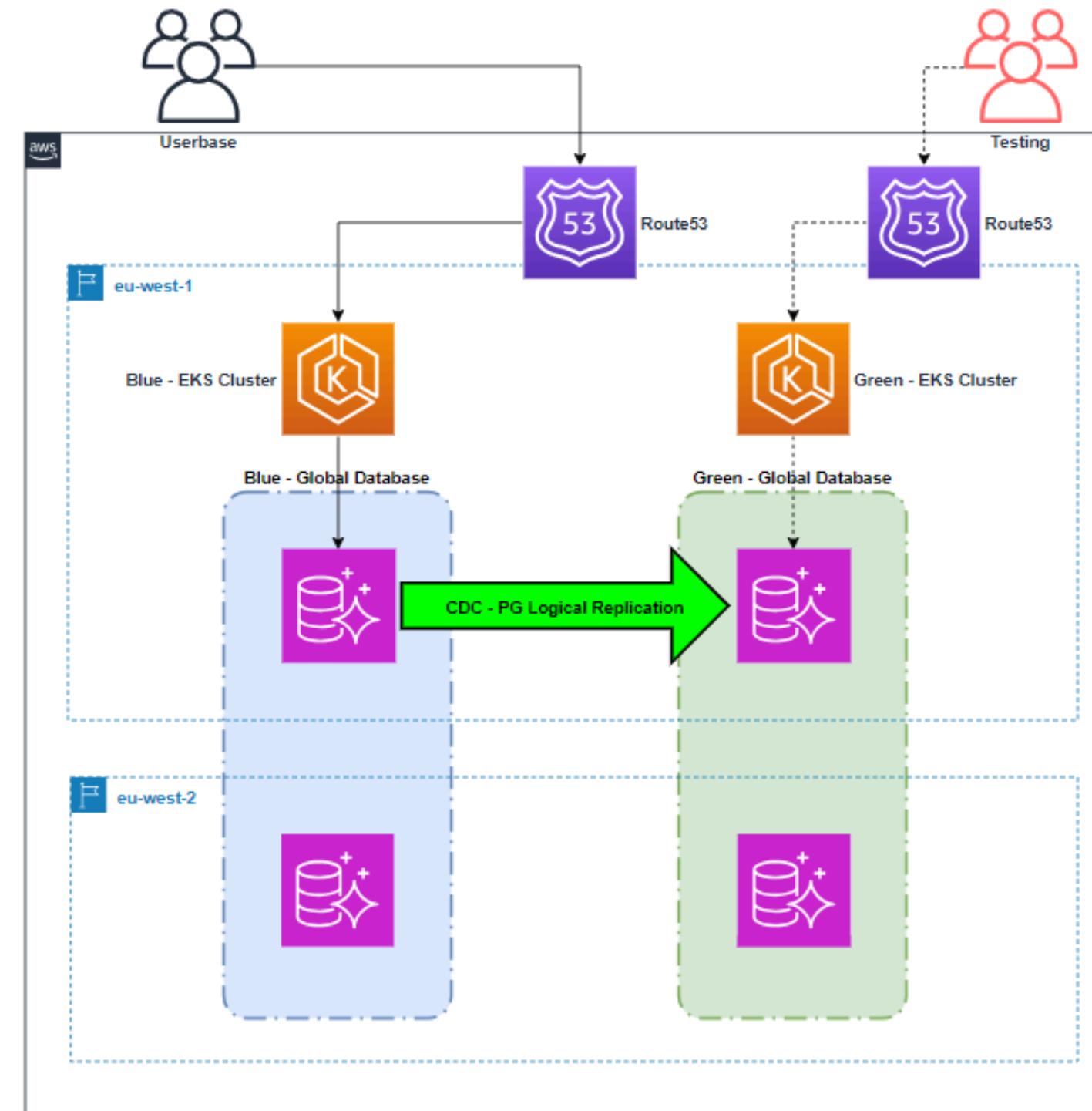
# AWS Databases and Data Engineering





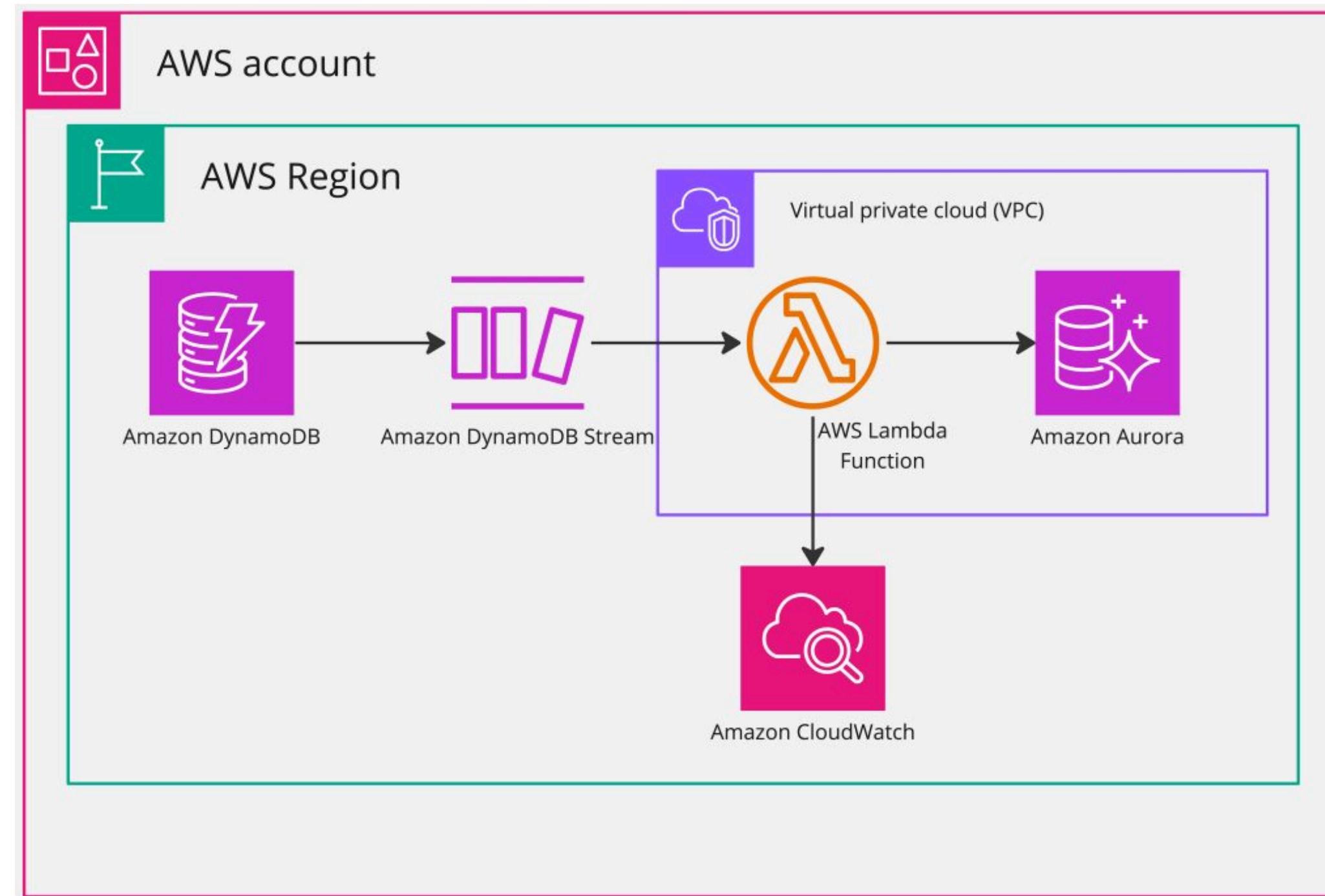
<https://aws.amazon.com/blogs/big-data/unlock-insights-on-amazon-rds-for-mysql-data-with-zero-etl-integration-to-amazon-redshift/>





<https://aws.amazon.com/blogs/database/how-london-stock-exchange-group-optimised-blue-green-deployments-for-amazon-aurora-postgresql-global-database/>





<https://aws.amazon.com/blogs/database/continuously-replicate-amazon-dynamodb-changes-to-amazon-aurora-postgresql-using-aws-lambda/>



# AWS Application Discovery Service



# AWS Application Discovery Service

**AWS Application Discovery Service** is a tool provided by Amazon Web Services (AWS) that helps organizations plan migration projects by gathering information about on-premises data centers. It automatically collects detailed information about your on-premises servers, applications, dependencies, and resource utilization.



# **Key Features of AWS Application Discovery Service**

**Agent or Agent-less**

**Application Dependency Mapping**

**Integration with AWS Migration Hub**



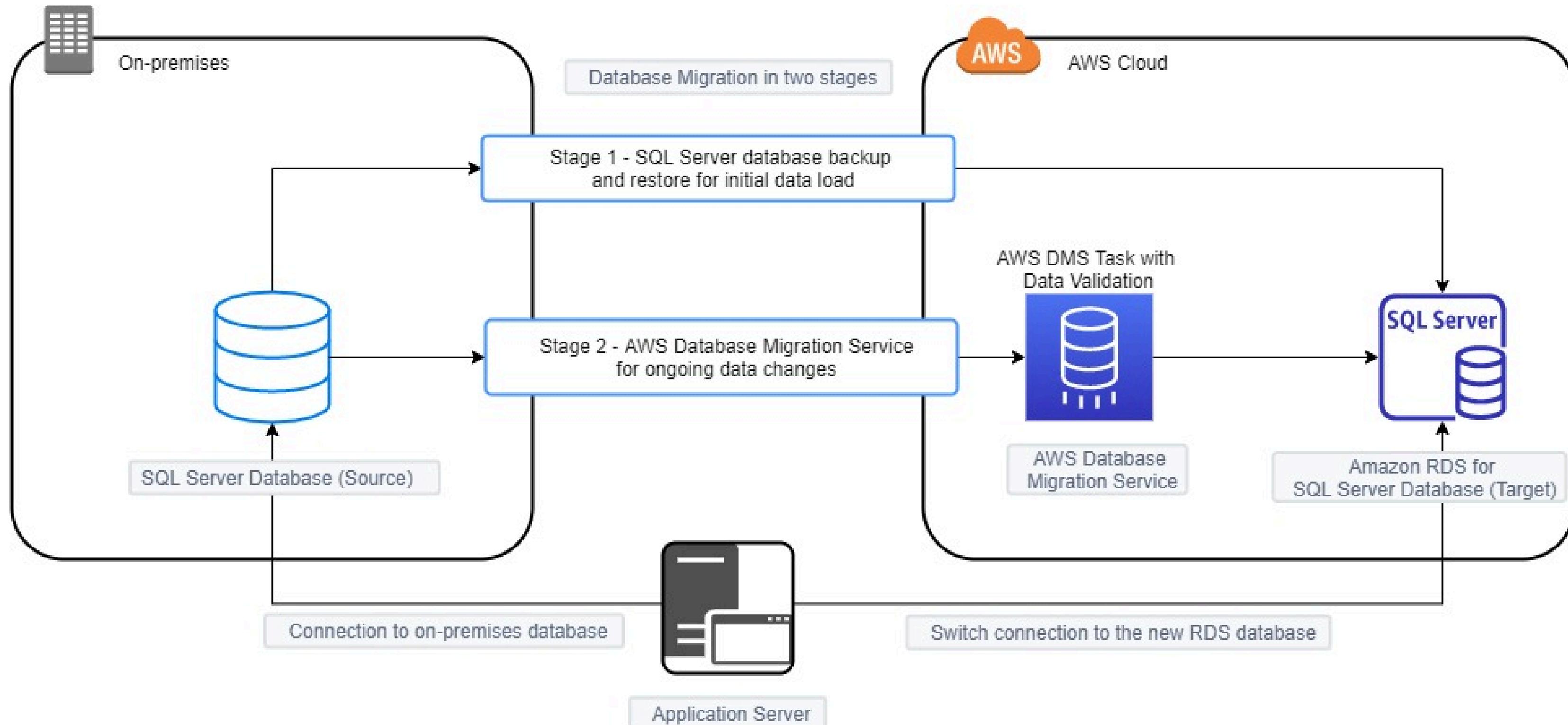
# AWS Database Migration Service



# AWS Database Migration Service

**AWS Database Migration Service (AWS DMS) is a managed service that facilitates the migration of databases to AWS. It supports the migration of data between different database engines, making it an ideal tool for both homogeneous migrations and heterogeneous migrations.**





<https://aws.amazon.com/blogs/database/migrating-your-sql-server-database-to-amazon-rds-for-sql-server-using-aws-dms/>



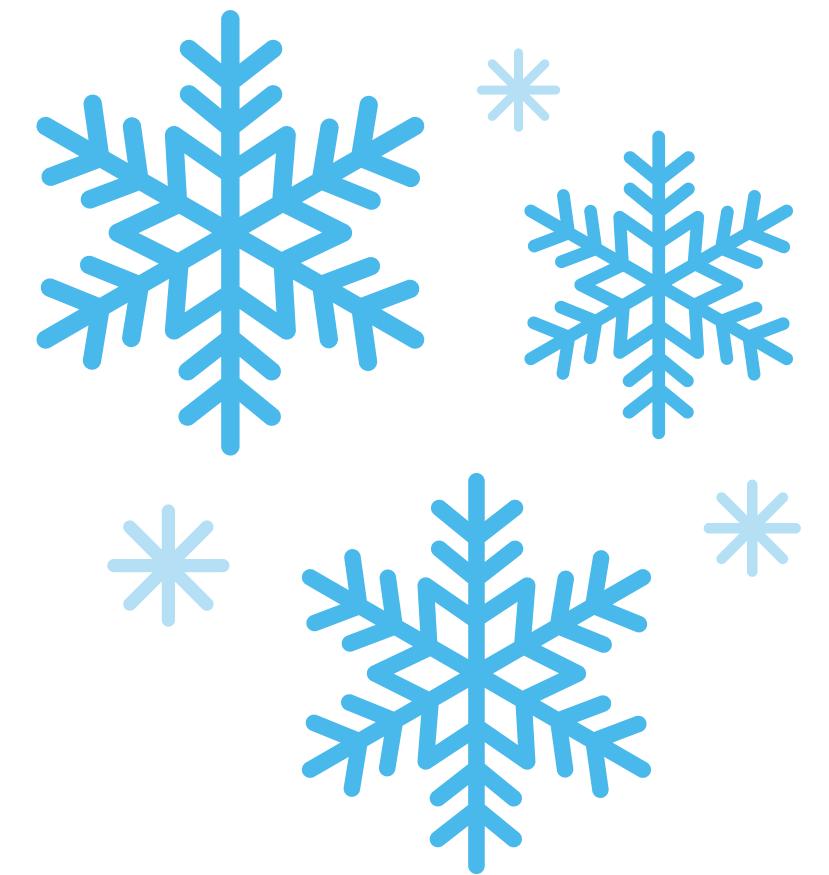
# AWS Snow Family



# AWS Snow Family

The AWS Snow Family is a collection of physical devices designed to transport large amounts of data to and from AWS.

These devices are particularly useful in scenarios where transferring data over the internet is impractical due to its volume, cost, or time constraints.



# Members of the AWS Snow Family

## AWS Snowcone

The smallest and most portable member of the Snow Family, AWS Snowcone is designed for use cases that require small data transfers or edge computing capabilities. It provides 8 terabytes (TB) of usable storage and can operate in challenging environments, including remote locations with limited connectivity.



# Members of the AWS Snow Family ctn..

## AWS Snowball Edge

**Snowball Edge Storage Optimized:** This variant is designed for data transfer and storage with 80 TB of usable storage capacity. It also supports optional compute capabilities for edge applications.



# Members of the AWS Snow Family ctn..

**Snowball Edge Compute Optimized:** In addition to data transfer and storage, this variant includes more powerful computing resources, including GPUs for machine learning, analytics, and processing of high-resolution video and images at the edge.



# AWS Application Integration

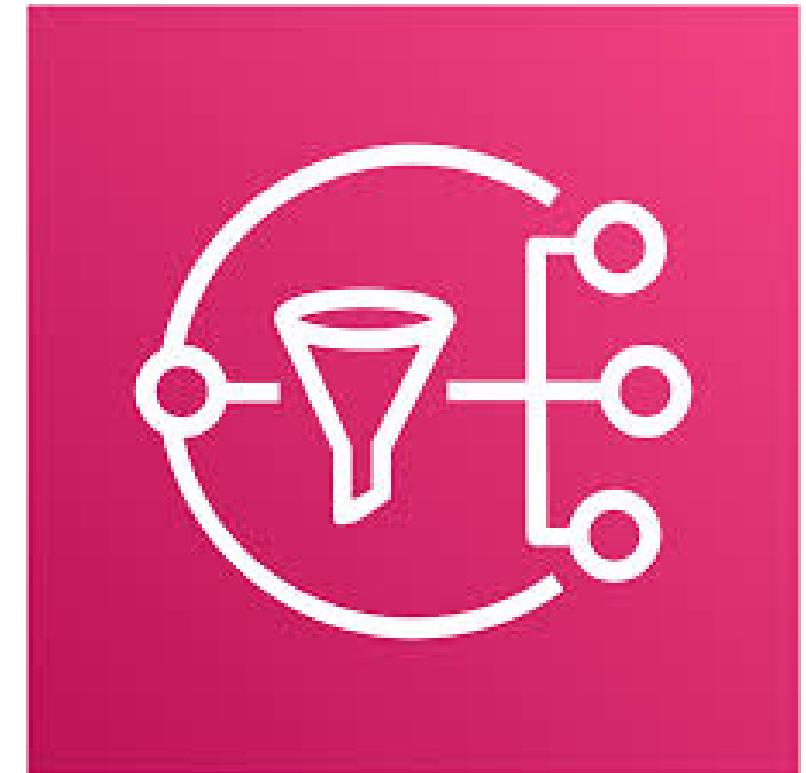


# AWS SNS

**Amazon Simple Notification Service (Amazon SNS) is a fully managed messaging service.**

**It enables the delivery of messages from publishers to subscribers.**

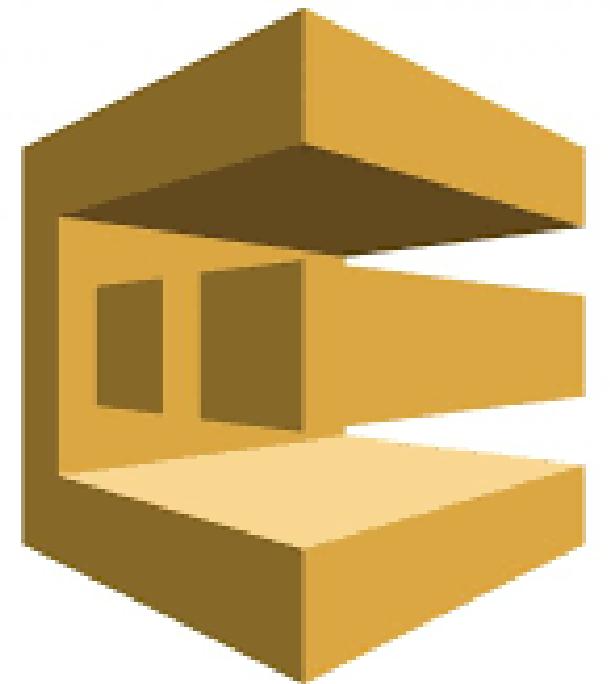
**SNS supports a variety of messaging patterns, including fan-out, pub/sub (publish/subscribe), and point-to-point messaging.**

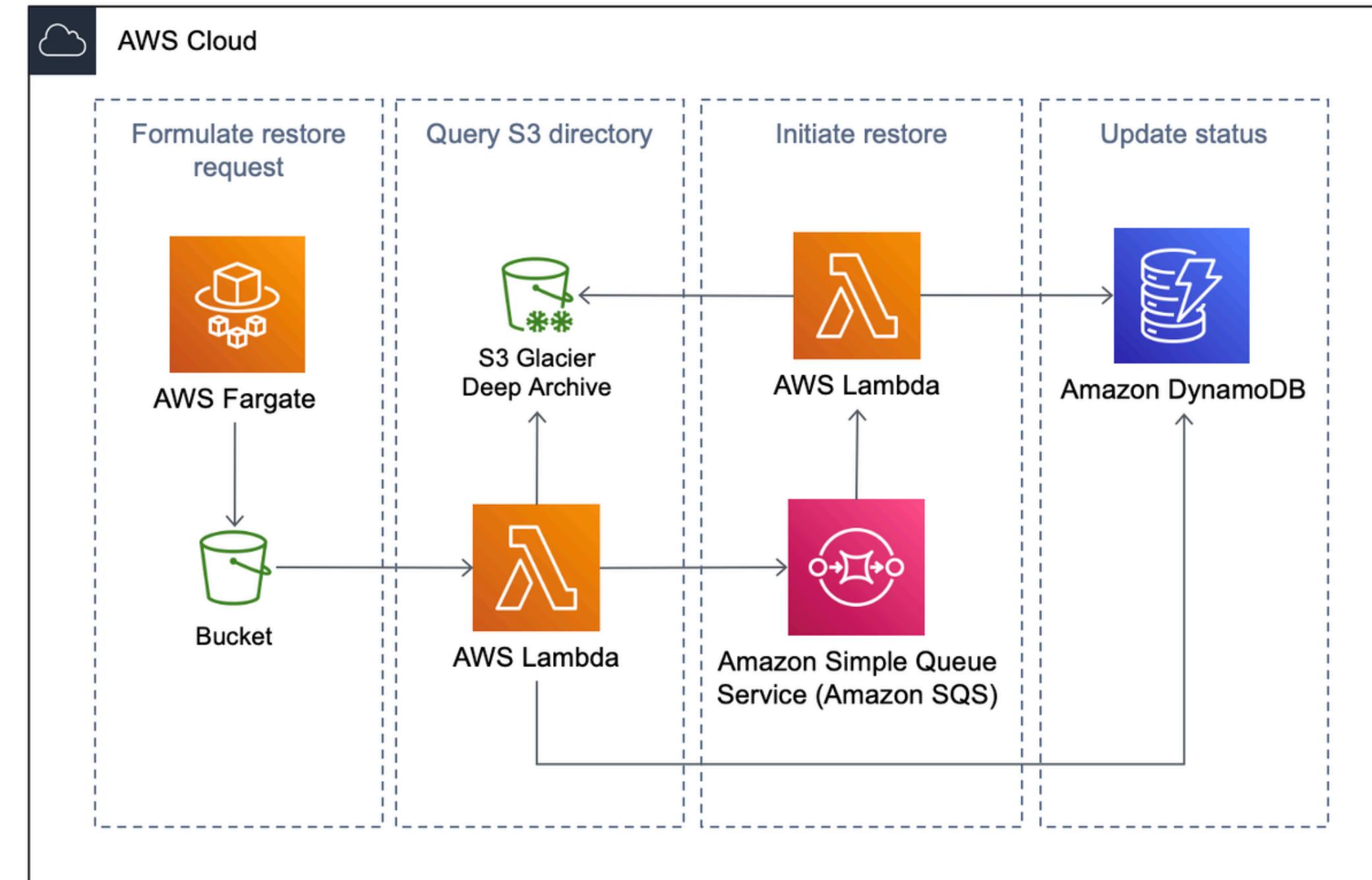


# AWS SQS

**It enables decoupling and scaling of microservices, distributed systems, and serverless applications.**

**SQS allows you to send, store, and receive messages between software components, ensuring that these components can operate independently and reliably.**





<https://aws.amazon.com/blogs/compute/implementing-aws-lambda-error-handling-patterns/>



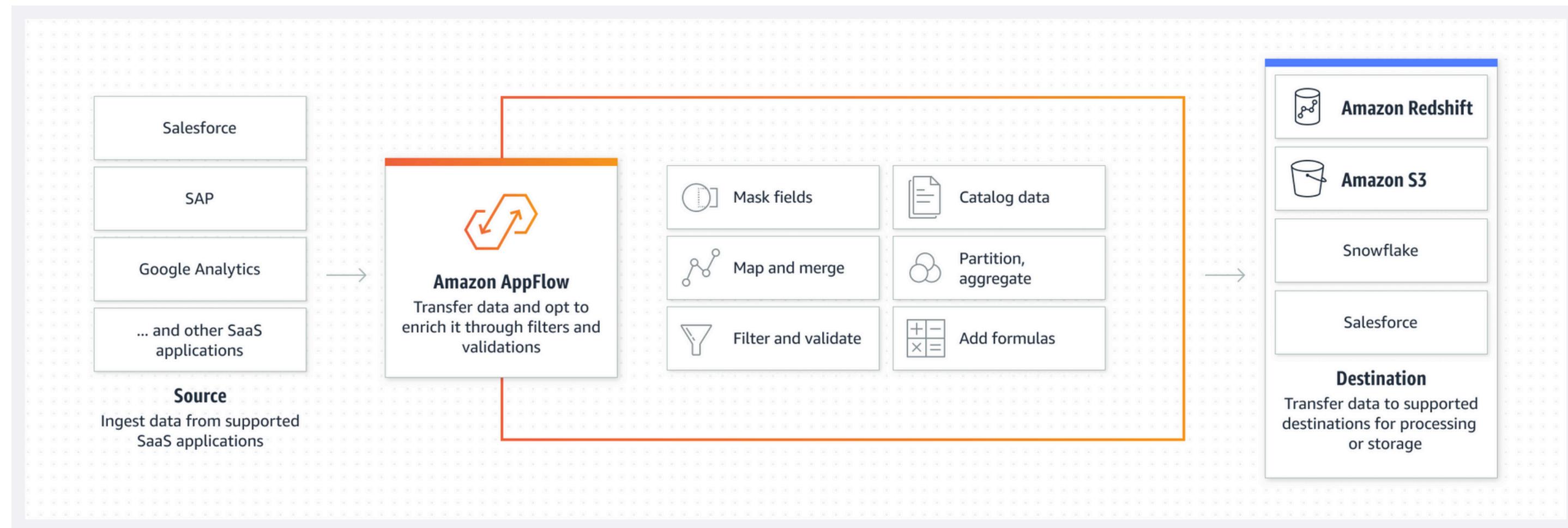
# Amazon Appflow

**Amazon AppFlow is a fully managed integration service that allows you to securely transfer data between AWS services and Software-as-a-Service (SaaS) applications.**



**It enables bidirectional data flows and is designed to simplify data integration offering built-in transformation and filtering capabilities.**





<https://aws.amazon.com/appflow/>



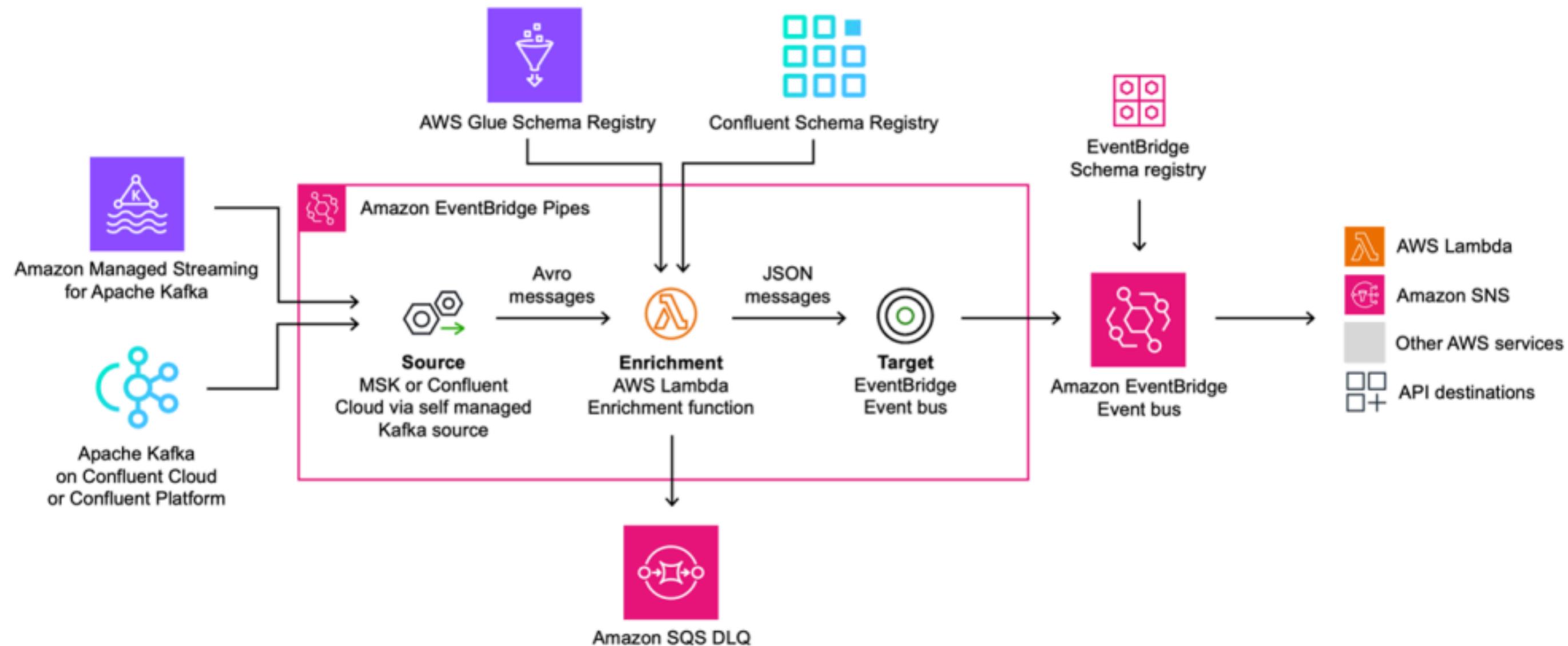
# Amazon Event Bridge

**Amazon EventBridge is a serverless event bus service that facilitates event-driven architectures by allowing applications to react to events from a variety of sources.**



**EventBridge enables seamless integration and orchestration between different services, applications, and systems by providing a robust platform for event-driven communication.**





<https://aws.amazon.com/blogs/compute/converting-apache-kafka-events-from-avro-to-json-using-eventbridge-pipes/>



# Amazon SageMaker



# Amazon SageMaker

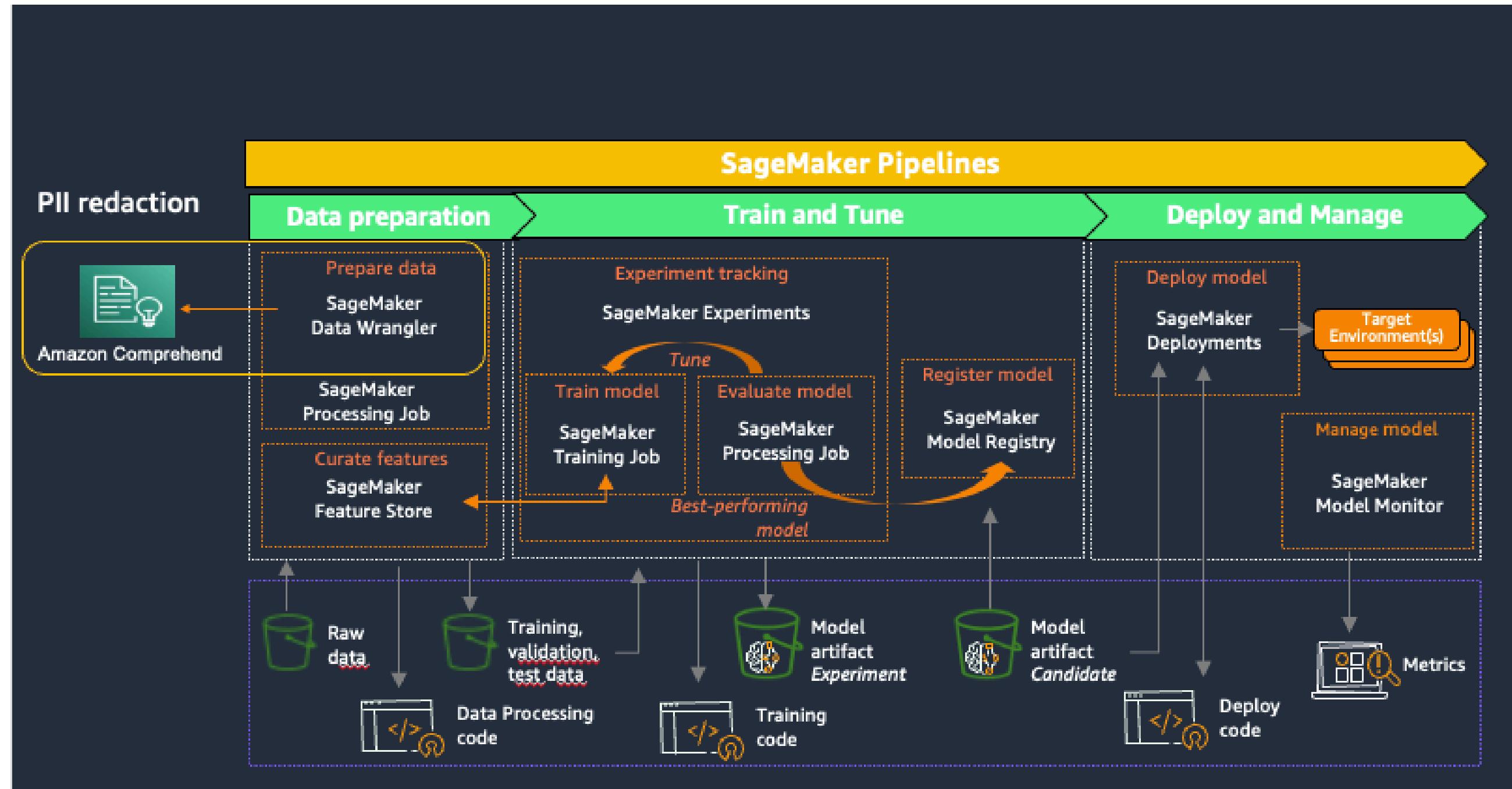
**Amazon SageMaker is a fully managed machine learning (ML) service that simplifies the process of building, training, and deploying machine learning models. While SageMaker is primarily known for its capabilities in ML model development, it also offers several features and tools that are valuable for data engineering tasks.**



# Amazon SageMaker Data Wrangler

**A tool for visual data preparation and transformation. It simplifies the process of cleaning, transforming, and analyzing data without needing extensive coding. Data Wrangler integrates with SageMaker's notebook instances, enabling users to perform data transformations interactively and prepare data for training and analysis**





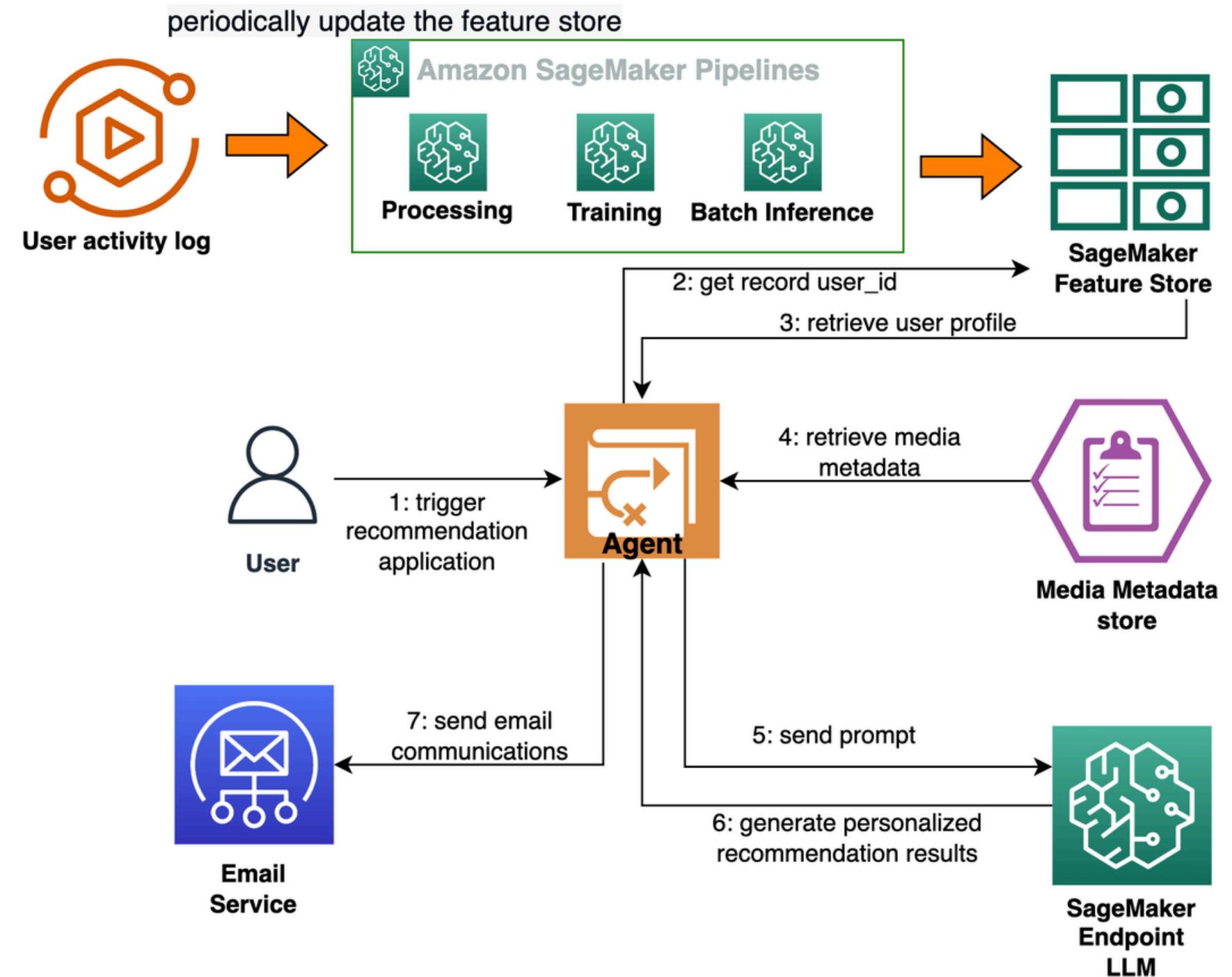
<https://aws.amazon.com/blogs/machine-learning/automatically-redact pii-for-machine-learning-using-amazon-sagemaker-data-wrangler/>



# Amazon SageMaker Feature Store

- A repository for storing and managing features used in machine learning models. It enables you to create, store, and retrieve features efficiently, ensuring consistency and reducing duplication across ML projects.





<https://aws.amazon.com/blogs/machine-learning/personalize-your-generative-ai-applications-with-amazon-sagemaker-feature-store/>

