



Intro to AWS Data Lakes and Analytics

Immersion Day

David Bayard, Solutions Architect
Padmaja Suren, Solutions Architect
Zach Gardner, Solutions Architect

October 1, 2019

If you wish to do today's hands-on labs, please send an email to gardz@amazon.com

Agenda – Part 1

9:00 am (15 minutes)

- Introductions & Objectives

9:15 am (90 minutes)

- Cloud Data Lakes
 - What is a Cloud Data Lake?
 - Things you can do with a Cloud Data Lake
 - Building a Cloud Data Lake on AWS
- Lab: Building a Data Lake with S3 & AWS Glue & Athena

11:00 am (45 minutes)

- Cloud Data Warehouse
 - What is a Cloud Data Warehouse and how does it fit with a Cloud Data Lake?
 - Things you can do with a Cloud Data Warehouse
 - Building a Cloud Data Warehouse on AWS
- Lab: Extending the lake with Amazon Redshift

12:00 pm (30 minutes)

- Lunch

Agenda – Part 2

12:30 pm (30 minutes)

- Overview of Lake Formation
- Demonstration: Lake Formation

1:00 pm (75 minutes)

- Populating the Data Lake/Data Warehouse
 - Discussion of different complimentary approaches
 - Database Migration Service (DMS) and Schema Conversion Tool (SCT)
 - Lake Formation Blueprint and AWS Glue ETL
 - Streaming Data (Kinesis/Kafka)
 - 3rd party approaches (informatica, etc)

2:30 pm (30 minutes)

- Visualizing Data
 - Quicksight Overview
- Demonstration of Quicksight

3:00 pm (30 minutes)

- Wrap-up
- Clean-up of Lab Environments

Lab Preparation

If you wish to do today's hands-on labs, please send an email to
gardz@amazon.com

Introductions and Objectives

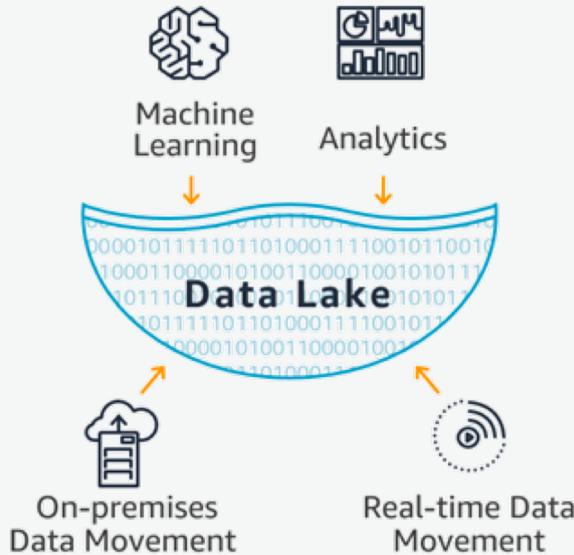
Q1: What is your level of experience with AWS?

Q2: What is your experience with data lakes and/or data warehousing?

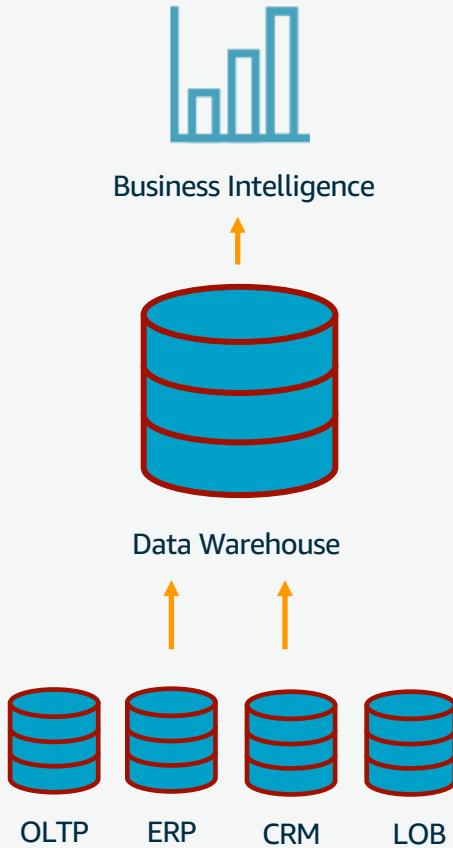
Cloud Data Lakes

What is a Data Lake?

- A **centralized repository** for both **structured** and **unstructured data**
- Store data **as-is** in **open-source file formats** to enable **direct analytics**



Traditionally, Analytics Looked Like This



Relational Data

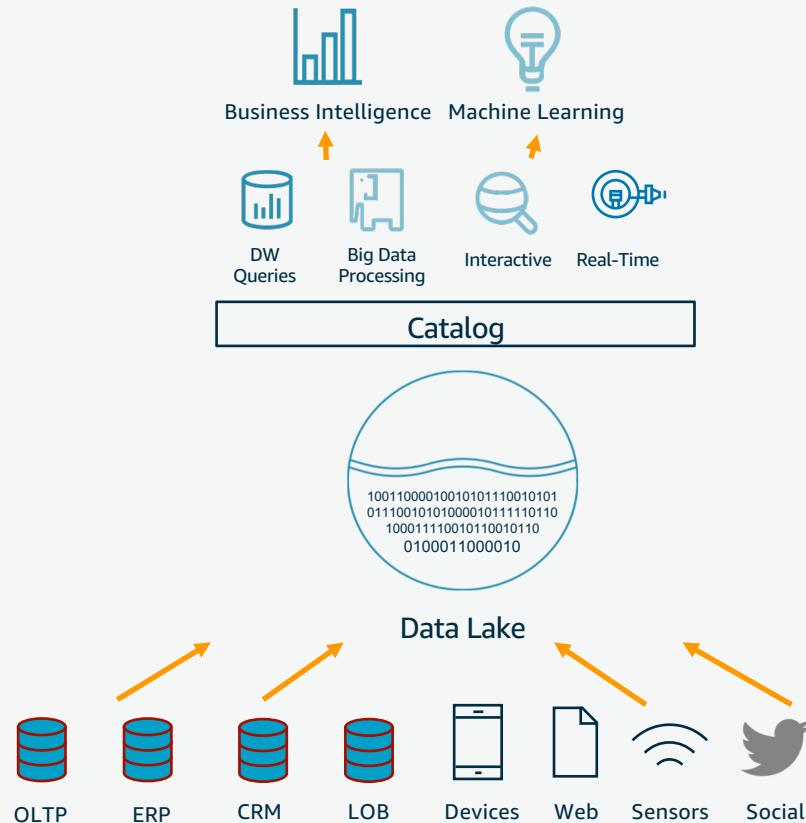
TBs Scale

Schema Defined Prior to Data Load

Operational and Ad Hoc Reporting

Large Initial Capex + \$\$K / TB/ Year

Data Lakes Extend the Traditional Approach



TB-PB-EBs Scale

All Data in one place, a Single Source of Truth

Relational and Non-Relational Data

Decouples (low cost) Storage and Compute

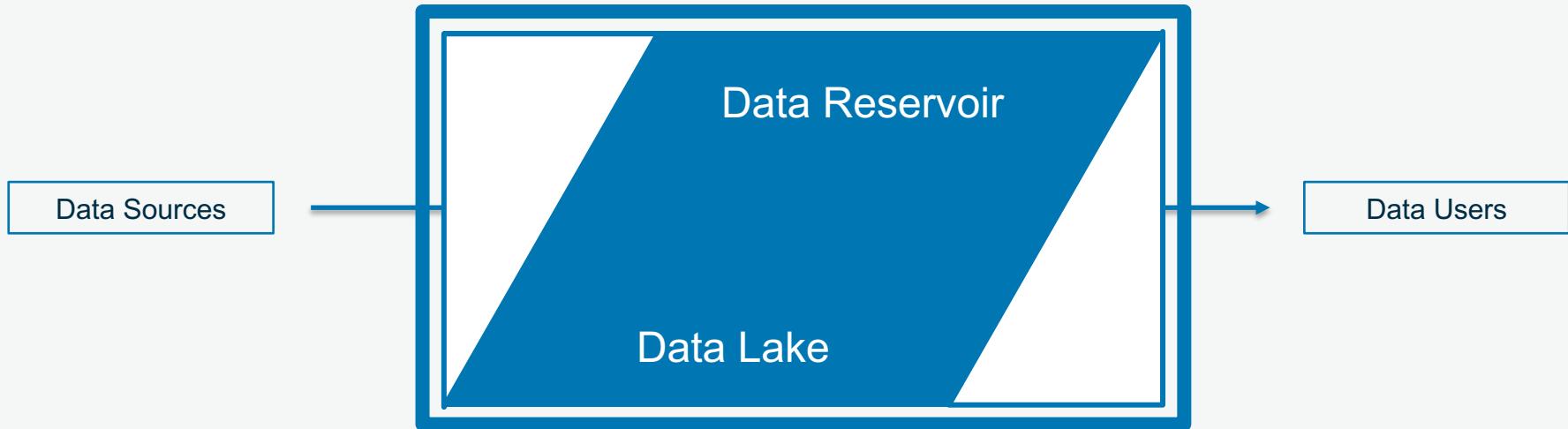
Schema on Read

Diverse Analytical Engines

Data Lakes and Data Reservoirs

Ingestion Triangle:
More work at the top.
Less work at the bottom.

Data Warehouse is a
kind of Data Reservoir



Data Lakes are optimized for large volumes of storage and ease of ingestion of any kind or flavor of data.

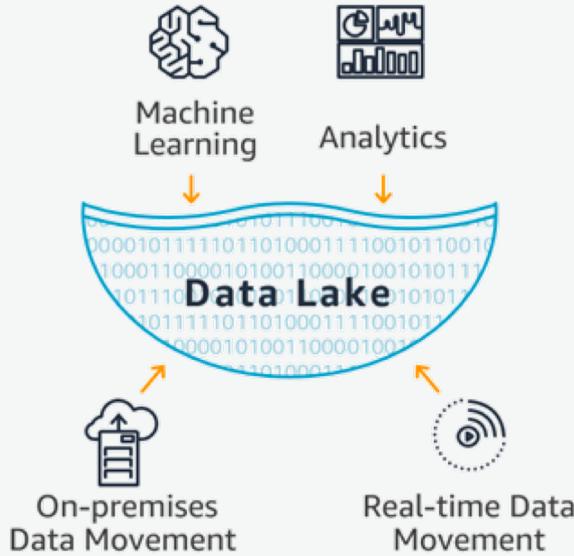
Data Warehouses are optimized for ease of retrieval of more critical and useful data.

Analytics Triangle:
Less work at the top.
More work at the bottom.

Why Data Lakes?

Why a Data Lake?

- Decouple **storage** from **compute**, allowing you to **scale**
- Enable **advanced analytics** across all of your data sources
- Reduce **complexity** in ETL and operational overhead
- Future **extensibility** as new database and analytics technologies are invented



Benefits of a Data Lake – All Data in One Place

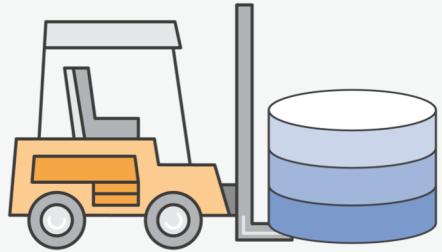


“Why is the data distributed in many locations? Where is the single source of truth ?”

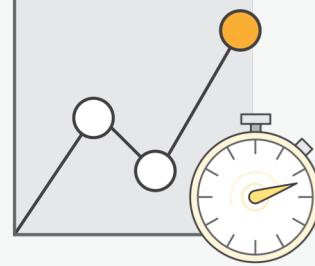


Store and analyze all of your data, from all of your sources, in one centralized location.

Benefits of a Data Lake – Quick Ingest

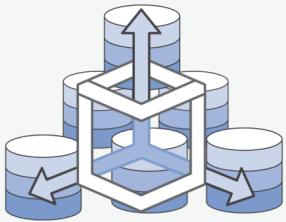


“How can I collect data quickly from various sources and store it efficiently?”

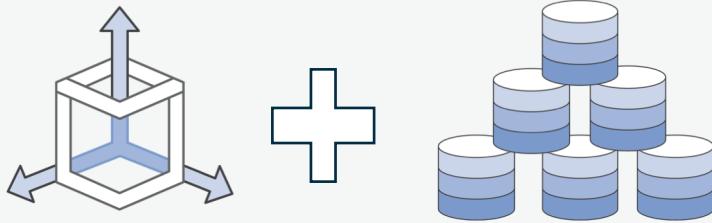


Quickly ingest data without needing to force it into a pre-defined schema.

Benefits of a Data Lake – Storage vs Compute



“How can I scale up with the volume of data being generated?”



Separating your storage and compute allows you to scale each component as required

Benefits of a Data Lake – Schema on Read



“Is there a way I can apply multiple analytics and processing frameworks to the same data?”



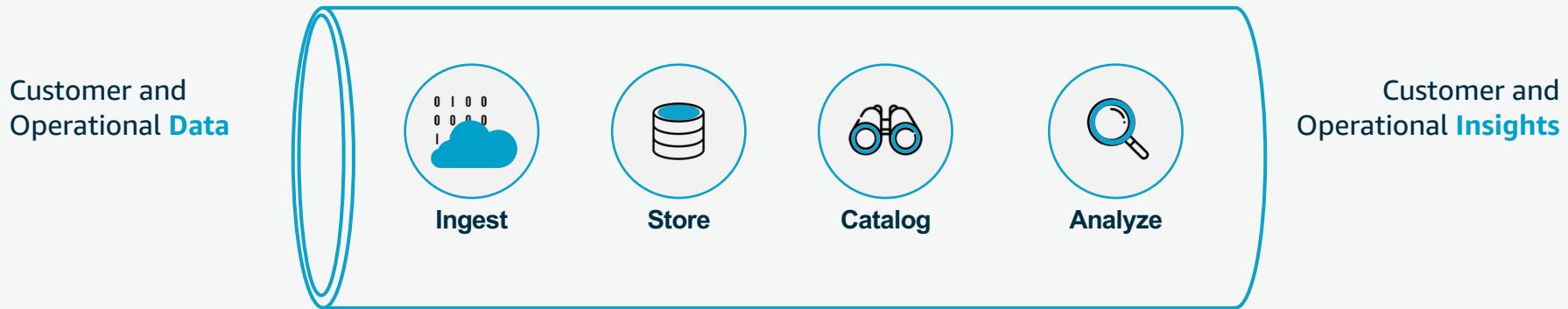
A Data Lake enables ad-hoc analysis by applying schemas on read, not write.

Building a Data Lake on AWS

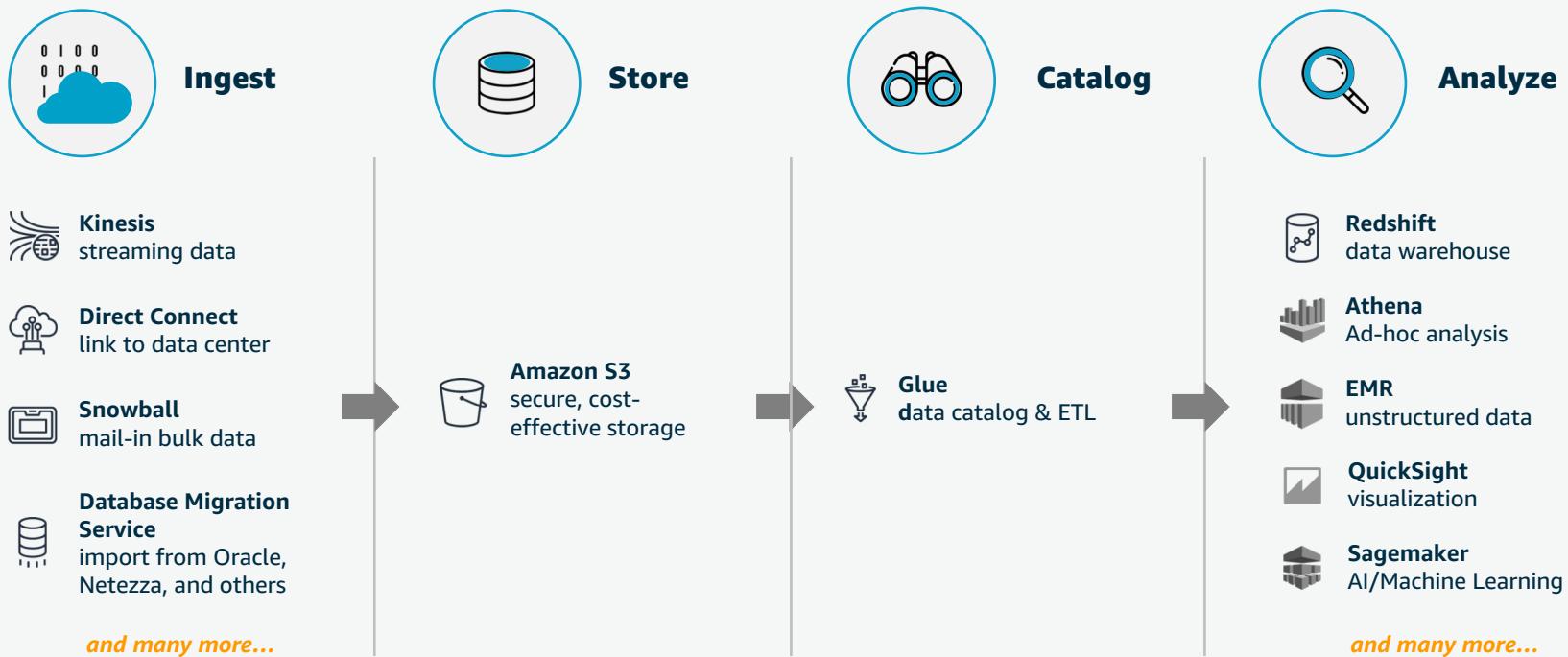
More data lakes & analytics on AWS than anywhere else



Realizing customer and operational insight from your data requires a robust Data Pipeline



Data Lake Reference Architecture





Amazon.com's vision is to be the earth's most customer-centric company; where people can find anything they want to buy online.

Challenge:

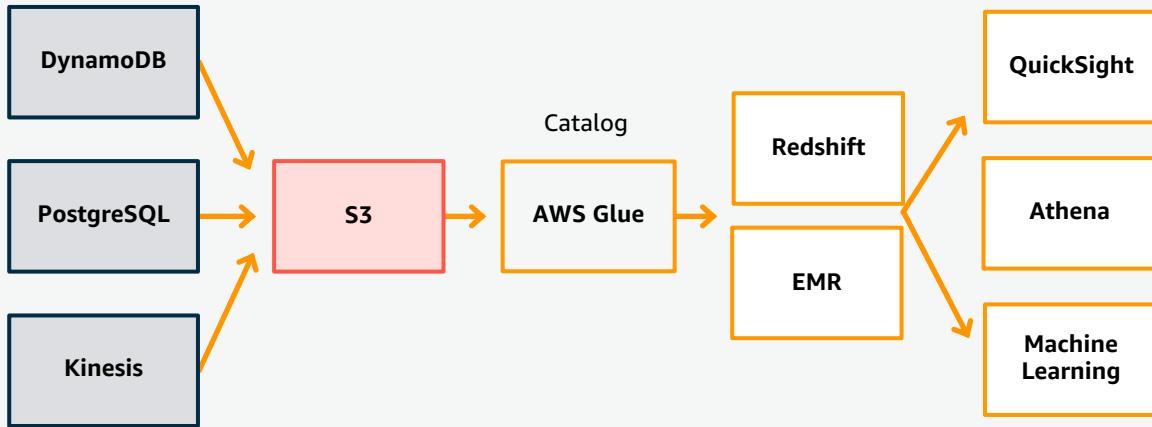
Load 500K+ transactions each day, and serve 300K+ queries/extracts each day from Amazon businesses (Amazon.com, Amazon Prime, Amazon Music, Amazon Alexa, Amazon Video, and Twitch).

Solution:

- Land data in S3 as a data lake
- Use Redshift as preferred SQL based analysis by business users, and EMR for machine learning



Amazon.com Uses AWS for Data Lakes & Analytics



- DynamoDB capturing all Amazon.com transactions
- Everything from DynamoDB, RDS PostgreSQL and Kinesis fed to a S3 data lake
- Glue used to catalog the data
- Redshift used for all SQL-based queries, and EMR for all machine learning and big data processing
- End-users use QuickSight for visualizations



Fox Film Entertainment is one of the largest movie studios in the world.

Challenge:

Processes 100+ of TB of data a day, 25k+ queries per day. They had a legacy data platform that struggled to scale, faced many outages, and had changing requirements from consumers who had many options.

Solution:

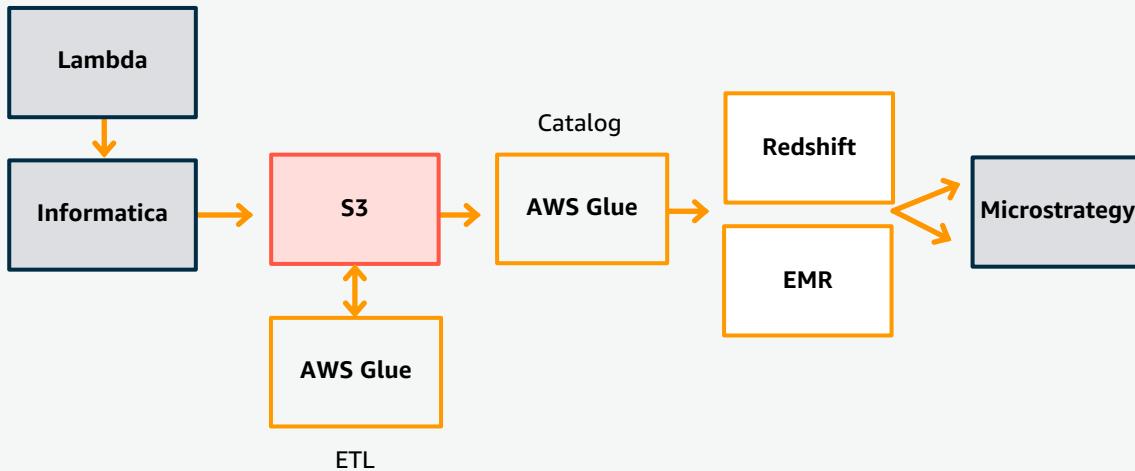
- Land data in S3 as a data lake
- Use Redshift and EMR as analytics engines to process data
- Saved 15–20% in overall costs (on-premises) with 30% of performance gain



Data Lake on AWS

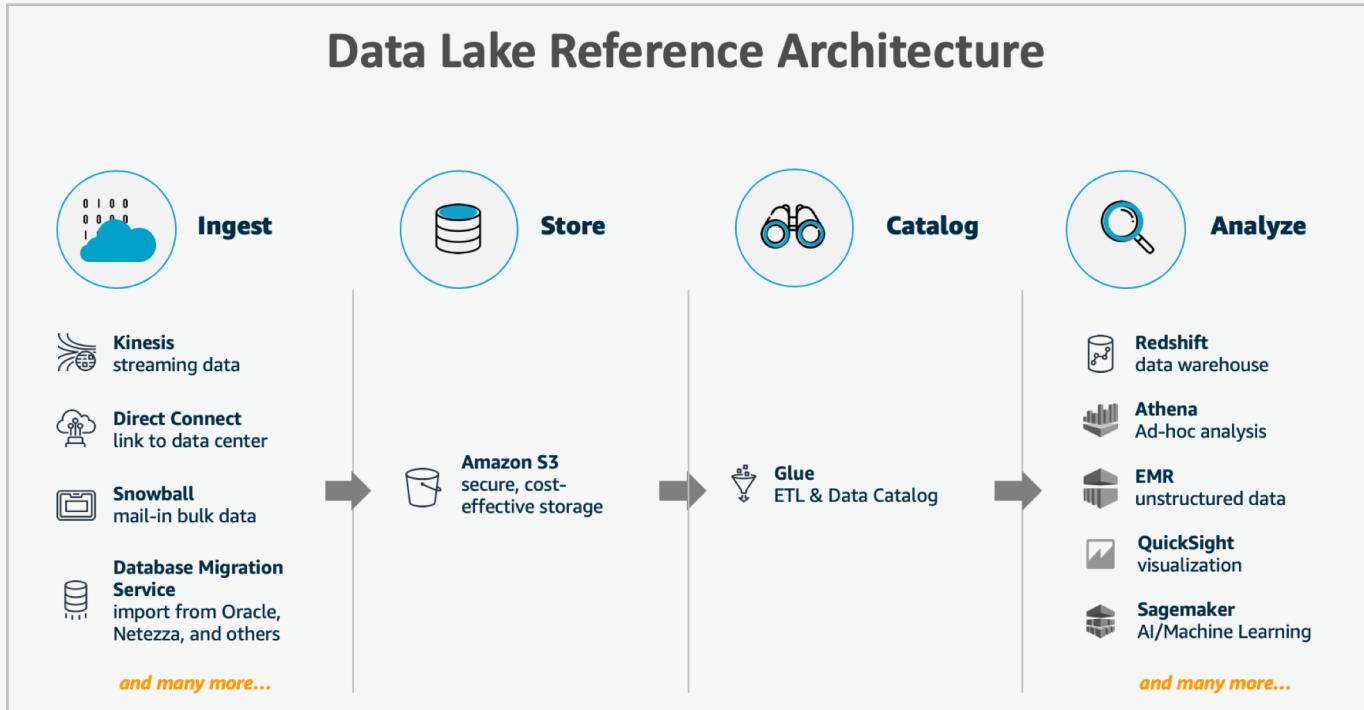


21st Century Fox Uses AWS for Data Lakes & Analytics



- Collect and ingest data with AWS Lambda for serverless scale and Informatica for data ingestion and transformation
- AWS Glue does ETL on data in S3 and provides a data catalog
- Redshift and EMR used as analytics engines
- Microstrategy used as a visualization tool

Amazon S3





Machine Learning

Amazon SageMaker
AWS Deep Learning AMIs
Amazon Rekognition
Amazon Lex
AWS DeepLens
Amazon Comprehend
Amazon Translate
Amazon Transcribe
Amazon Polly



Analytics

Amazon Athena
Amazon EMR
Amazon Redshift
Amazon Elasticsearch Service
Amazon Kinesis
Amazon QuickSight



On-premises Data Movement

AWS Direct Connect
AWS Snowball
AWS Snowmobile
AWS Database Migration Service



Real-time Data Movement

AWS IoT Core
Amazon Kinesis Data Firehose
Amazon Kinesis Data Streams
Amazon Kinesis Video Streams

Data Lakes start with
Amazon S3

Why Amazon S3 for a Data Lake?



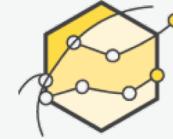
Durable

Designed for **11 9s** of durability



Available

Designed for **99.99%** availability



High performance

- Multiple upload
- Range GET



Easy to use

- Simple REST API
- AWS SDKs
- Read-after-create consistency
- Event notification
- Lifecycle policies



Scalable

- Store as much as you need
- Scale storage and compute independently
- No minimum usage commitments



Integrated

- Amazon EMR
- Amazon Redshift
- Amazon Athena
- AWS Glue
- Amazon DynamoDB
- More...

Lab Time

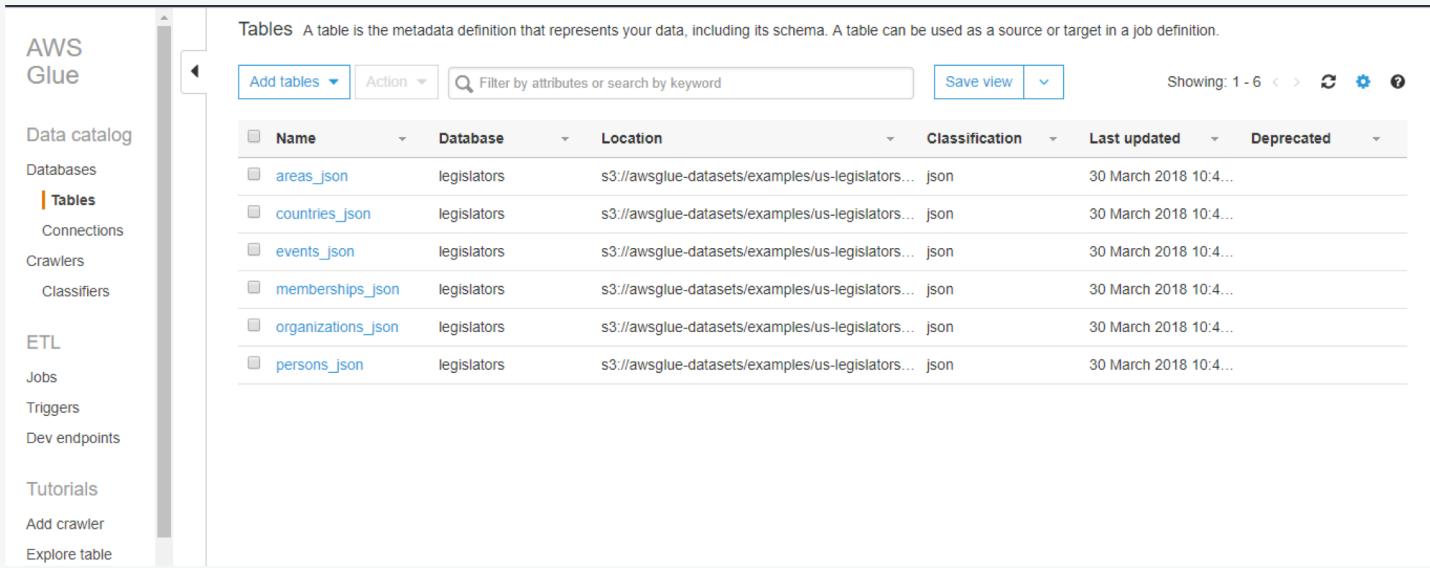
<https://github.com/dbayardAWS/intro-data-lake>

Please start Lab1.

Continue thru the **Upload the sample dataset to the Data Lake** step.

What can you do with a Data Lake?

Create a Central Data Catalog with AWS Glue



The screenshot shows the AWS Glue Data Catalog interface. On the left, a sidebar menu lists various categories: Data catalog, Databases, Tables (which is selected and highlighted in orange), Connections, Crawlers, Classifiers, ETL, Jobs, Triggers, Dev endpoints, Tutorials, Add crawler, and Explore table. The main content area is titled "Tables" with a sub-instruction: "A table is the metadata definition that represents your data, including its schema. A table can be used as a source or target in a job definition." Below this is a search bar with "Add tables" and "Action" buttons, a keyword search field, and a "Save view" dropdown. The table itself has columns: Name, Database, Location, Classification, Last updated, and Deprecated. It lists six tables: areas_json, countries_json, events_json, memberships_json, organizations_json, and persons_json, all associated with the "legislators" database and located at s3://awsglue-datasets/examples/us-legislators... json. All entries were last updated on March 30, 2018, at 10:42.

Name	Database	Location	Classification	Last updated	Deprecated
areas_json	legislators	s3://awsglue-datasets/examples/us-legislators... json		30 March 2018 10:4...	
countries_json	legislators	s3://awsglue-datasets/examples/us-legislators... json		30 March 2018 10:4...	
events_json	legislators	s3://awsglue-datasets/examples/us-legislators... json		30 March 2018 10:4...	
memberships_json	legislators	s3://awsglue-datasets/examples/us-legislators... json		30 March 2018 10:4...	
organizations_json	legislators	s3://awsglue-datasets/examples/us-legislators... json		30 March 2018 10:4...	
persons_json	legislators	s3://awsglue-datasets/examples/us-legislators... json		30 March 2018 10:4...	

Query Directly with Amazon Athena & Amazon Redshift

Athena **Query Editor** Saved Queries History Catalog Manager Settings Tutorial Help

DATABASE sampledb

TABLES Filter Tables... Add table... elb_logs

- timestamp (string)
- elbname (string)
- requestip (string)
- requestport (int)
- backendip (string)
- backendport (int)
- requestprocessingtime (double)
- backendprocessingtime (double)
- clientresponsetime (double)
- elbresponsecode (string)
- backendresponsecode (string)
- receivedbytes (bigint)
- sentbytes (bigint)
- requestverb (string)
- url (string)
- protocol (string)

ELB Select Query Sample query to view peak load ELBs during a particular timeframe

```
1 SELECT elbname, count(1) as num
2 FROM sampledb.elb_logs
3 Where elbresponsecode = '200'
4 GROUP BY elbname
5 ORDER BY num DESC limit 10;
```

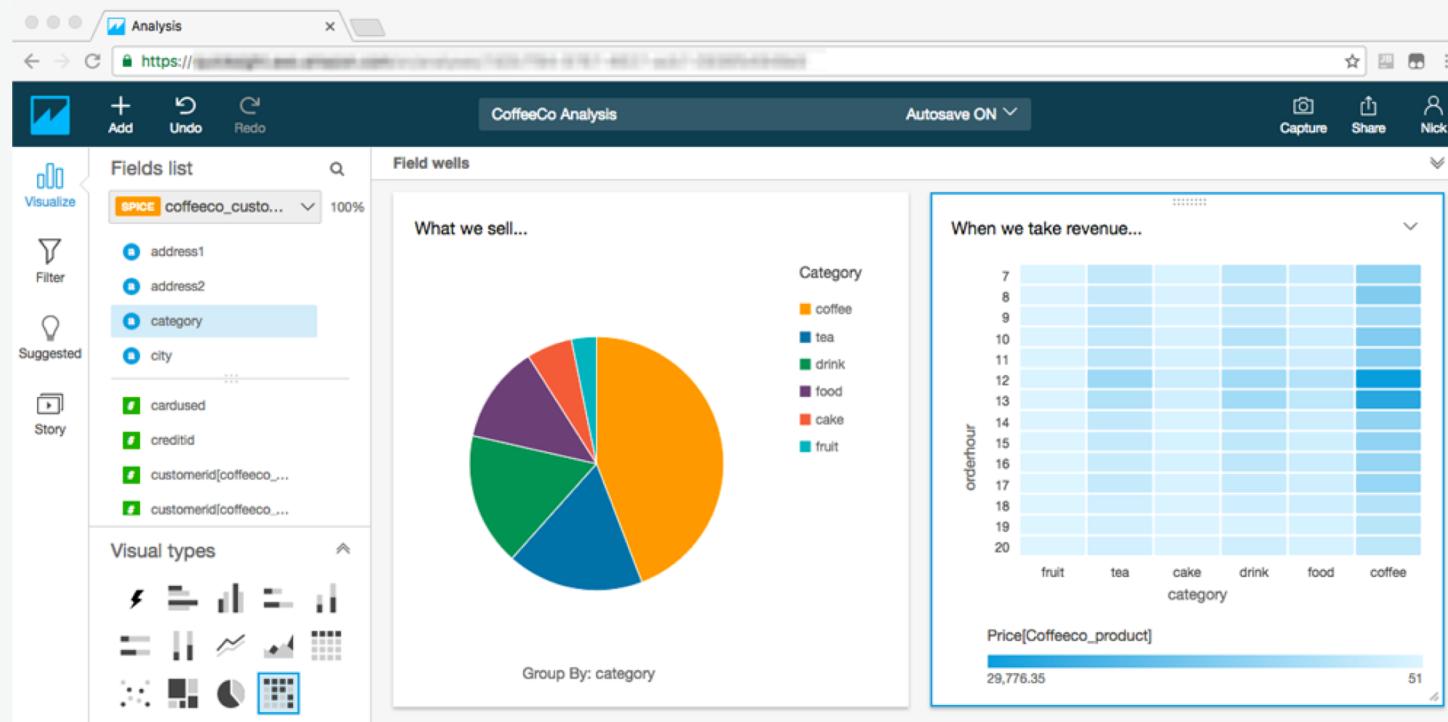
Use Ctrl + Enter to run query, Ctrl + Space to autocomplete

Run Query Save As Format Query New Query (Run time: 1.9 seconds, Data scanned: 826.54KB)

Results

	elbname	num
1	lb-demo	4108

Create Visualizations with Amazon QuickSight



Analyze with Hadoop on Amazon EMR

Create Cluster - Advanced Options [Go to quick options](#)

Step 1: Software and Steps

Step 2: Hardware

Step 3: General Cluster Settings

Step 4: Security

Software Configuration

Vendor Amazon MapR

Release emr-4.2.0  

<input checked="" type="checkbox"/> Hadoop 2.6.0	<input checked="" type="checkbox"/> Hive 1.0.0	<input type="checkbox"/> Mahout 0.11.0
<input checked="" type="checkbox"/> Zeppelin-Sandbox 0.5.5	<input type="checkbox"/> Hue 3.7.1	<input checked="" type="checkbox"/> Spark 1.5.2
<input type="checkbox"/> Ganglia 3.6.0	<input type="checkbox"/> Presto-Sandbox 0.125	<input type="checkbox"/> Oozie-Sandbox 4.2.0
<input type="checkbox"/> Pig 0.14.0		

Scala

```
val movieLensHomeDir = "s3://emr.examples/movieLens/"

val movies = sc.textFile(movieLensHomeDir + "movies.dat").map { line =>
    val fields = line.split("::")
    // format: (movieId, movieName)
    (fields(0).toInt, fields(1))
}.collect.toMap

val ratings = sc.textFile(movieLensHomeDir + "ratings.dat").map { line =>
    val fields = line.split("::")
    // format: (timestamp % 10, Rating(userId, movieId, rating))
    (fields(3).toLong % 10, Rating(fields(0).toInt, fields(1).toInt, fields(2).toDouble))
}
```

This is a live tutorial, you can run the code yourself. (Shift-Enter to Run)

Load Data into Table

```
import org.apache.commons.io.IOUtils
import java.net.URL
import java.util.concurrent.Charset
bonText: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[32] at parallelize at <console>:65
defined class Bank
bank: org.apache.spark.sql.DataFrame = [age: int, job: string, marital: string, education: string, balance: int]
```

Max Age

```
val maxAge = bank.select("age", "count(1) value").groupBy("age").orderBy("age").first().count(1).value
from bank
where age >= 30
group by age
order by age
```

Grouped Bar Chart

Single Value

Train ML Models with Amazon SageMaker

The screenshot shows the AWS Amazon SageMaker console. The left sidebar has a 'Jobs' section selected. The main area is titled 'Input data configuration' and shows settings for a 'train' channel. The 'Content type - optional' field is set to 'json'. Other fields include 'Compression type' (None), 'Record wrapper' (None), 'S3 data type' (S3Prefix), 'S3 data distribution type' (FullyReplicated), and 'S3 location' (s3://my-deepar-data/train-data). A 'Done' button is at the bottom right.

Services ▾ Resource Groups ▾ BurnerConsoleAccessClientRole...

Amazon SageMaker X

Dashboard Notebook instances Jobs Resources Models Endpoint configuration Endpoints

Input data configuration

Create up to 8 channels of input sources. If the algorithm you chose supports multiple input channels, you can specify those here. See [Algorithms Provided by Amazon SageMaker: Common Parameters](#)

train Edit Remove

Channel name train
Maximum of 64 alphanumeric characters. Can include hyphen (-), period (.), and underscore (_), but not spaces. Must be unique within a training job.

Content type - optional json

Compression type None Record wrapper None

S3 data type S3Prefix S3 data distribution type FullyReplicated

S3 location s3://my-deepar-data/train-data

Add channel Done

Feedback English (US)

© 2008 - 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Load into Downstream Services



Amazon Redshift

Run complex analytic queries against petabytes of structured data



Amazon Aurora

A MySQL and PostgreSQL compatible relational database built for the cloud



Amazon DynamoDB

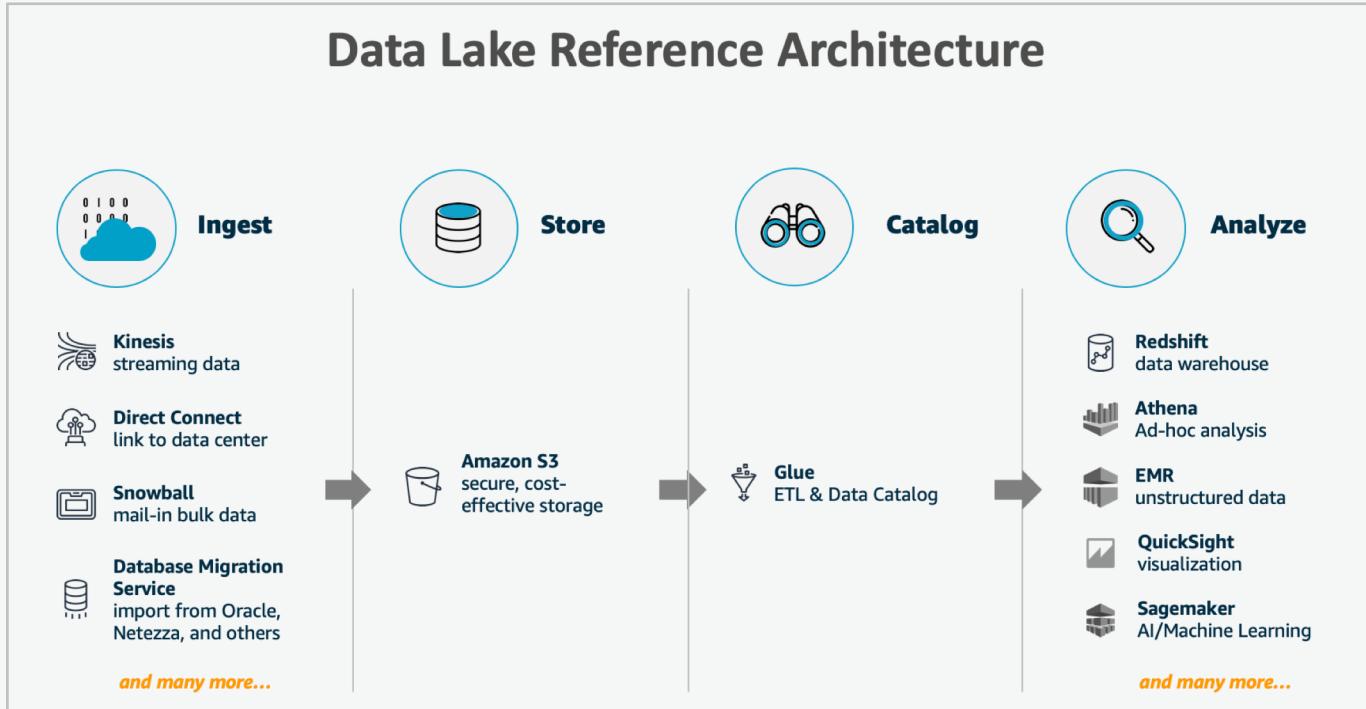
A NoSQL database service that delivers consistent, single-digit millisecond latency at any scale.



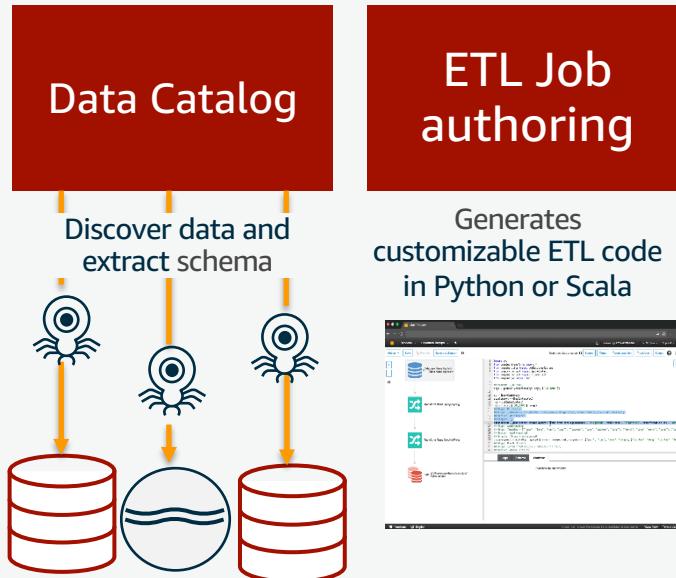
Amazon Elasticsearch

Delivers Elasticsearch's real-time analytics capabilities alongside the availability, scalability, and security that production workloads require.

AWS Glue



AWS Glue: Data Catalog & ETL Service



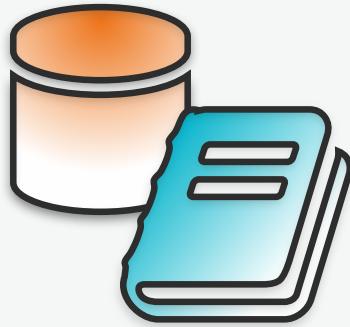
Automatically discovers data and stores schema

Data is immediately searchable, and available to extract, transform, and load (ETL)

Automatically generates customizable ETL code

Schedules and runs your ETL jobs

Serverless

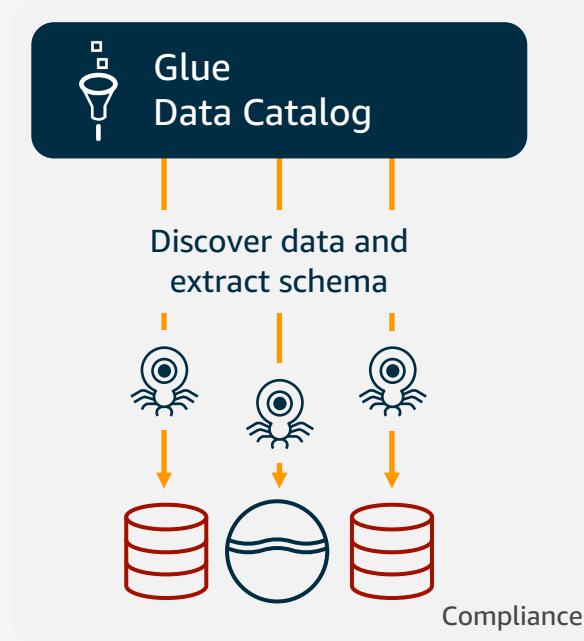


Glue data catalog

Discover and organize your data sets

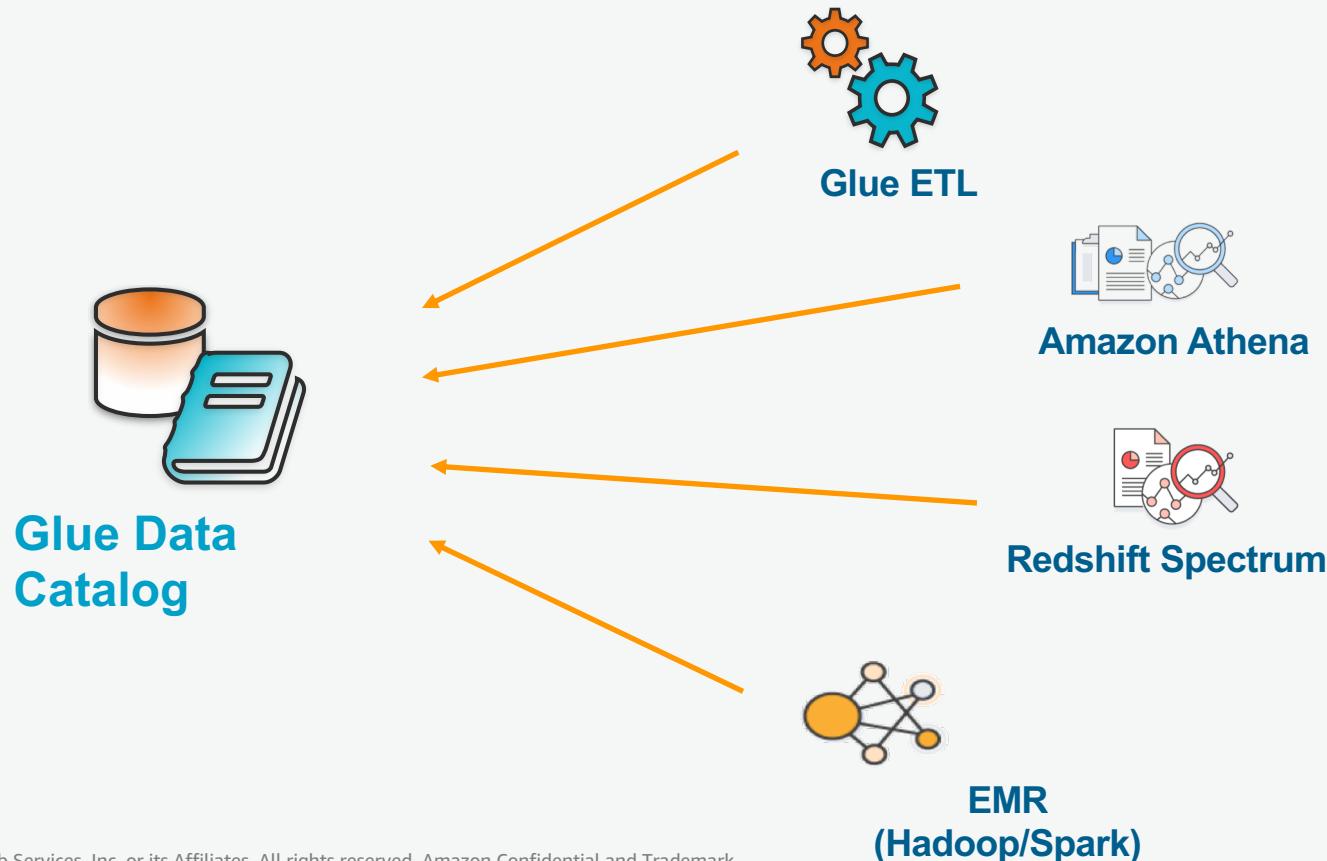
AWS Glue—Data Catalog

Make data discoverable



- Automatically discovers data and stores schema
- Catalog makes data searchable, and available for ETL
- Catalog contains table and job definitions
- Computes statistics to make queries efficient

Glue: Data Catalog – Queryable by Many Services



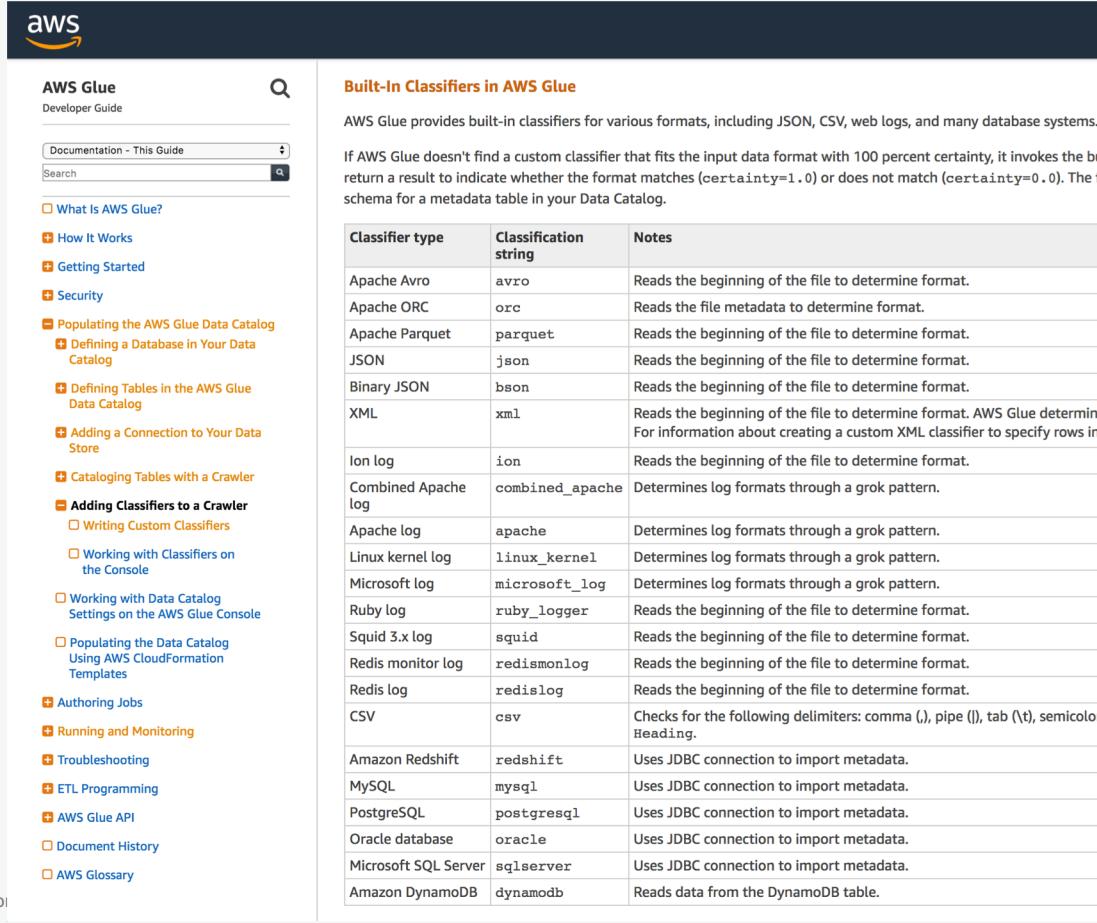
Glue: Data Catalog - Crawlers

The screenshot shows the AWS Glue Data Catalog - Crawlers page. On the left, there's a sidebar with navigation links for Services, Resource Groups, AWS Glue (selected), Data catalog, Databases, Tables, Connections, Crawlers (selected), Classifiers, ETL, Jobs, Triggers, Dev endpoints, Tutorials, Add a crawler, and Explore table. The main content area has a title 'Crawlers' with a sub-instruction: 'A crawler connects to a data store, progresses through a prioritized list of classifiers to determine the schema for your data, and then creates metadata tables in your data catalog.' Below this are buttons for 'Add crawler', 'Run crawler', and 'Action'. A table lists seven crawlers with columns: Name, Schedule, Status, Logs, Last runtime, Median runtime, Tables updated, and Tables added. The crawlers listed are DoubleClickCra..., FlightCrawling, NYC-Taxi-Craw..., Uber, adTechCrawler, and policeCrawler, all in Ready status.

Name	Schedule	Status	Logs	Last runtime	Median runtime	Tables updated	Tables added
DoubleClickCra...		Ready	Logs	15 secs	15 secs	0	1
FlightCrawling		Ready	Logs	20 secs	20 secs	0	1
NYC-Taxi-Craw...		Ready	Logs	13 secs	13 secs	0	4
Uber		Ready	Logs	16 secs	16 secs	0	1
adTechCrawler		Ready	Logs	20 secs	20 secs	0	1
policeCrawler		Ready	Logs	15 secs	15 secs	1	0

Features Include:

- Built-in classifiers
 - Detect file type
 - Extract schema
 - Identify partitions
- On-Demand or Scheduled Execution
- Build-your-own classifiers
 - Grok for ease of use



The screenshot shows the AWS Glue Developer Guide interface. The left sidebar contains a navigation tree with sections like 'What Is AWS Glue?', 'How It Works', 'Getting Started', 'Security', 'Populating the AWS Glue Data Catalog', 'Defining Tables in Your Data Catalog', 'Adding a Connection to Your Data Store', 'Cataloging Tables with a Crawler', 'Adding Classifiers to a Crawler', 'Working with Classifiers on the Console', 'Working with Data Catalog Settings on the AWS Glue Console', 'Populating the Data Catalog Using AWS CloudFormation Templates', 'Authoring Jobs', 'Running and Monitoring', 'Troubleshooting', 'ETL Programming', 'AWS Glue API', 'Document History', and 'AWS Glossary'. The main content area is titled 'Built-In Classifiers in AWS Glue' and describes built-in classifiers for various formats. A table lists the classifiers, their classification strings, and notes about how they determine file format.

Built-In Classifiers in AWS Glue

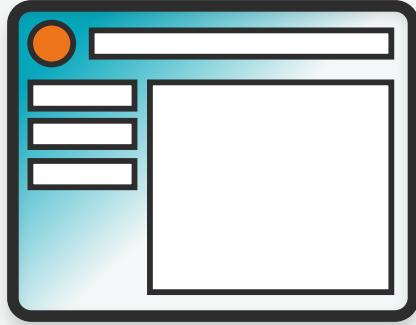
AWS Glue provides built-in classifiers for various formats, including JSON, CSV, web logs, and many database systems. If AWS Glue doesn't find a custom classifier that fits the input data format with 100 percent certainty, it invokes the built return a result to indicate whether the format matches (`certainty=1.0`) or does not match (`certainty=0.0`). The first schema for a metadata table in your Data Catalog.

Classifier type	Classification string	Notes
Apache Avro	avro	Reads the beginning of the file to determine format.
Apache ORC	orc	Reads the file metadata to determine format.
Apache Parquet	parquet	Reads the beginning 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 For information about creating a custom XML classifier to specify rows in th
Ion log	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.
Redis monitor log	redismonlog	Reads the beginning of the file to determine format.
Redis log	redislog	Reads the beginning of the file to determine format.
CSV	csv	Checks for the following delimiters: comma (,), pipe (), tab (\t), semicolon (;) Heading.
Amazon Redshift	redshift	Uses JDBC connection to import metadata.
MySQL	mysql	Uses JDBC connection to import metadata.
PostgreSQL	postgresql	Uses JDBC connection to import metadata.
Oracle database	oracle	Uses JDBC connection to import metadata.
Microsoft SQL Server	sqlserver	Uses JDBC connection to import metadata.
Amazon DynamoDB	dynamodb	Reads data from the DynamoDB table.

Ref: AWS Glue Developer Guide

© 2019, Amazon Web Services, Inc. or



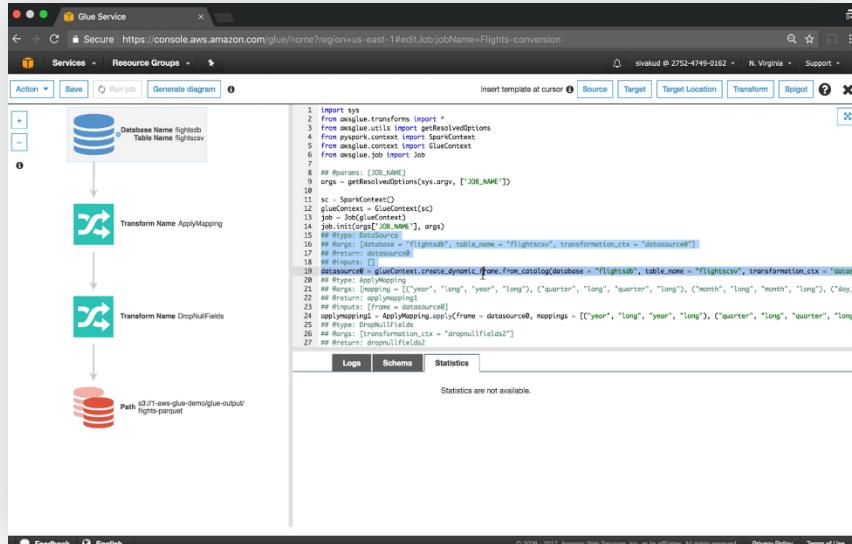


Glue ETL Jobs

Author and Deploy ETL Jobs Easily

AWS Glue—ETL Service

Make ETL scripting and deployment easy



- Automatically generates ETL code
- Code is customizable with Python and Spark
- Endpoints provided to edit, debug, test code
- Jobs are scheduled or event-based
- Serverless

AWS Glue: Job Authoring - Pick a Source

Job properties
GlueDemo

Data source

Data target

Schema

Review

Choose your data sources

Filter by attributes or search by keyword

Name	Database	Location	Classification
auroragluetestdb_sample_test_table	aurora_db	gluetestdb.sample_test_table	mysql

Choose from Manually-Defined or Crawler-Generated Sources:

- S3 Bucket
- RDBMS
- Redshift
- DynamoDB

AWS Glue: Job Authoring – Pick a Target

Write output results into an existing table.

Choose your data targets

Create tables in your data target
 Use tables in the data catalog and update your data target

Filter by attributes or search by keyword

Name	Database	Location	Classification
auroragluetestdb_sample_test_table	aurora_db	gluetestdb.sample_test_table	mysql

Create tables in your data target
 Use tables in the data catalog and update your data target

Data store
Amazon S3

Format
Parquet

Target path
s3://my-bucket/

Crawler-Defined or Manually-Created:

- S3 Bucket
- RDBMS
- Redshift

AWS Glue: Job Authoring – Code Generation

The screenshot shows the AWS Glue Job Authoring interface for creating a new job named "schema". On the left, there's a sidebar with "Job properties" (DeleteMe, Data source: city_baltimore, Data target: canonical, Schema, Review), and a main area titled "Add job: schema". The main area displays two tables: "Map to target" and "Target schema".

Map to target:

Column name	Data type	Map to target
crimedate	string	crimedate
crimetime	string	crime_time
crimecode	string	crimecode
location	string	location
description	string	description
inside/outside	string	-
weapon	string	weapon
post	bigint	-
district	string	district
neighborhood	string	neighborhood
location 1	string	-
premise	string	-
total incidents	bigint	total incidents

Target schema:

Column name	Data type
crimedate	string
crime_time	string
crimecode	string
location	string
description	string
weapon	string
district	string
neighborhood	string
total incidents	long

Arrows indicate the mapping from the "Map to target" columns to the corresponding columns in the "Target schema". The "Add column" button is located at the top right of the "Target schema" table.

Existing columns
in target

Can extend/add
new columns to
target

AWS Glue: Job Authoring – Code Generation

Add job: schema

Column name	Data type	Map to target	Column name	Data type
crimeidate	string	crimeidate	crimeidate	string
crimetime	string	crimetime	crime_time	string
crimecode	string	crimecode	crimecode	string
location	string	location	location	string
description	string	description	description	string
inside/outside	string	-	weapon	string
weapon	string	weapon	weapon	string
post	bigint	-	district	string
district	string	district	neighborhood	string
neighborhood	string	neighborhood	total incidents	long
location 1	string	-	neighborhood	string
premise	string	-	location 1	string
total incidents	bigint	total incidents	premise	string

Action ▾ Generate diagram

Database Name nytaxianalysis
Table Name city_baltimore

Transform Name ApplyMapping

Transform Name SelectFields

Database Name Incidents
Table Name canonical

```
1 import sys
2 from awsglue.transforms import *
3 from awsglue.utils import getResolvedOptions
4 from pyspark.context import SparkContext
5 from awsglue.context import GlueContext
6 from awsglue.job import Job
7
8 ## @params: [JOB_NAME]
9 args = getResolvedOptions(sys.argv, ['JOB_NAME'])
10
11 sc = SparkContext()
12 glueContext = GlueContext(sc)
13 job = Job(glueContext)
14 job.init(args['JOB_NAME'], args)
15 ## @type: DataSource
16 ## @args: [database = "nytaxianalysis", table_name = "city_baltimore", transformation_ctx = "datasource0"]
17 ## @return: datasource0
18 ## @inputs: []
19
```

Glue generates transformation graph and either *Python* or *Scala* Spark code

Apache Spark & AWS Glue ETL



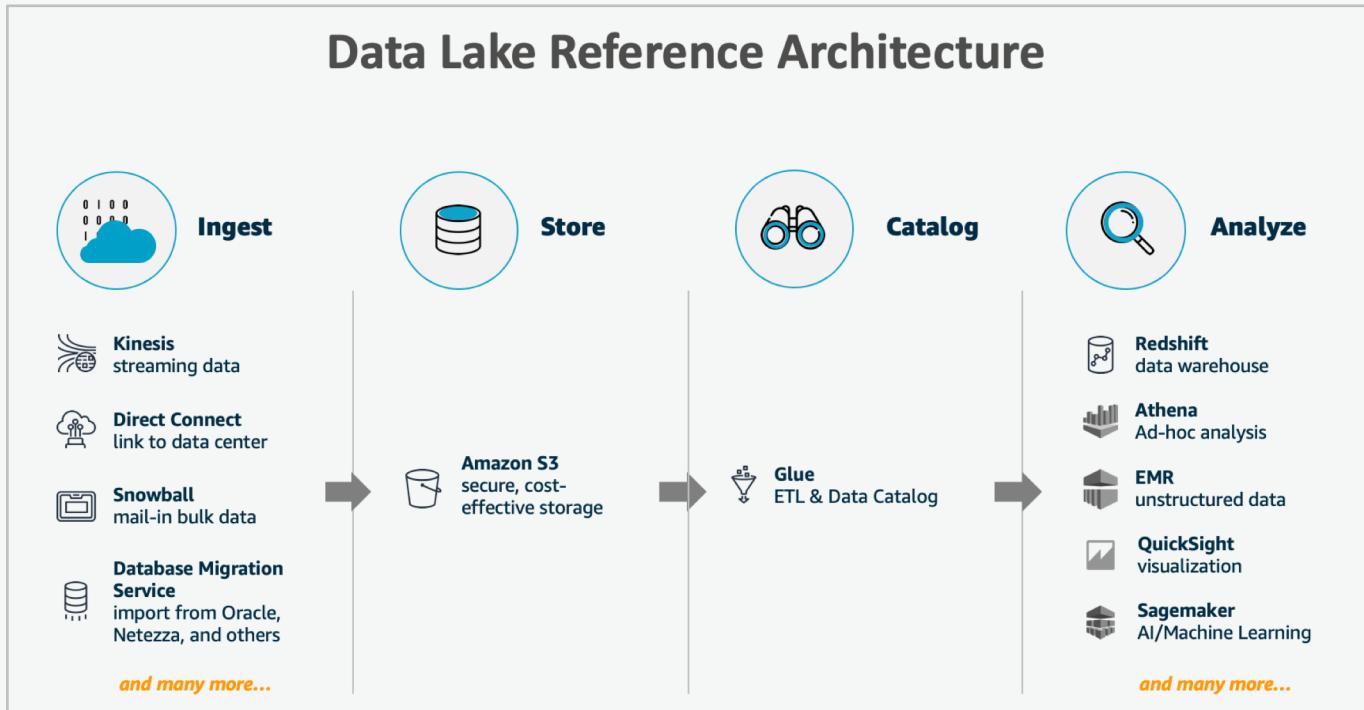
What is Apache Spark?

- Parallel, scale-out data processing engine
- Fault-tolerance built-in
- Flexible interface: Python scripting, SQL
- Rich eco-system: ML, Graph, analytics, ...

AWS Glue ETL libraries

- Integration: Data Catalog, job orchestration, code-generation, job bookmarks, S3, RDS
- ETL transforms, more connectors & formats
- New data structure: Dynamic frames

Amazon Athena



Amazon Athena—Interactive Analysis

Interactive query service to analyze data in Amazon S3 using standard SQL

No infrastructure to set up or manage and no data to load

Query Instantly



Zero setup cost; just point to S3 and start querying

Pay per query



Pay only for queries run; save 30–90% on per-query costs through compression

Open



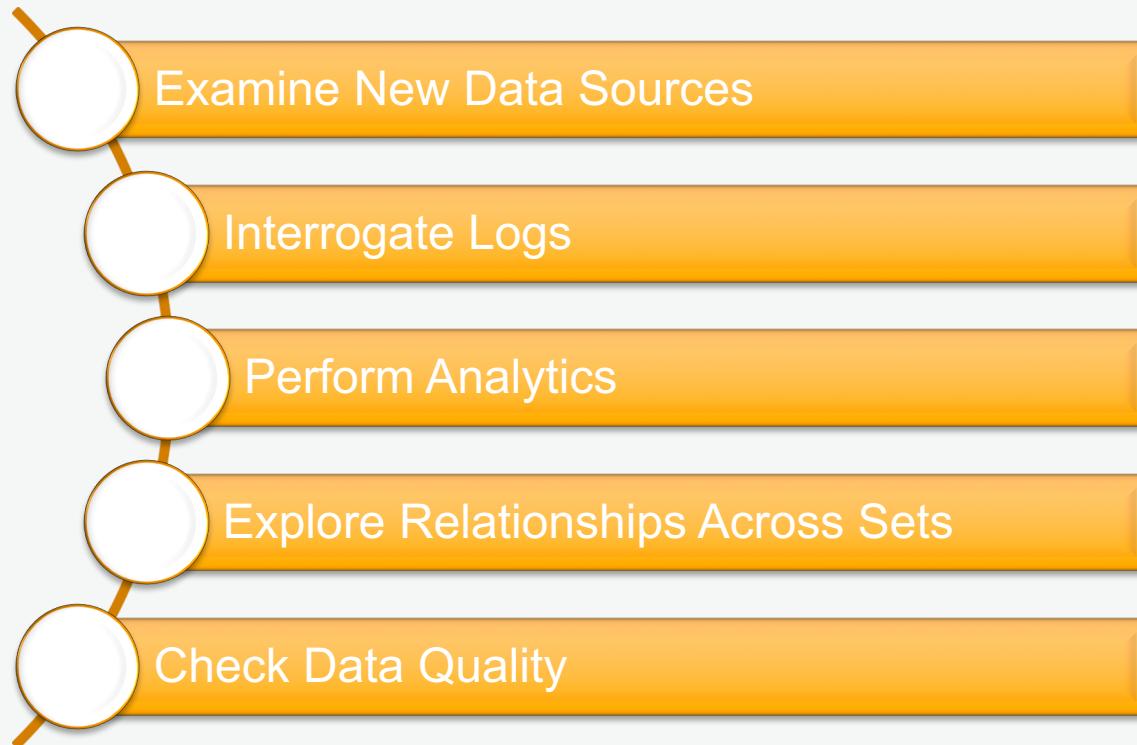
ANSI SQL interface, JDBC/ODBC drivers, multiple formats, compression types, and complex joins and data types

Easy

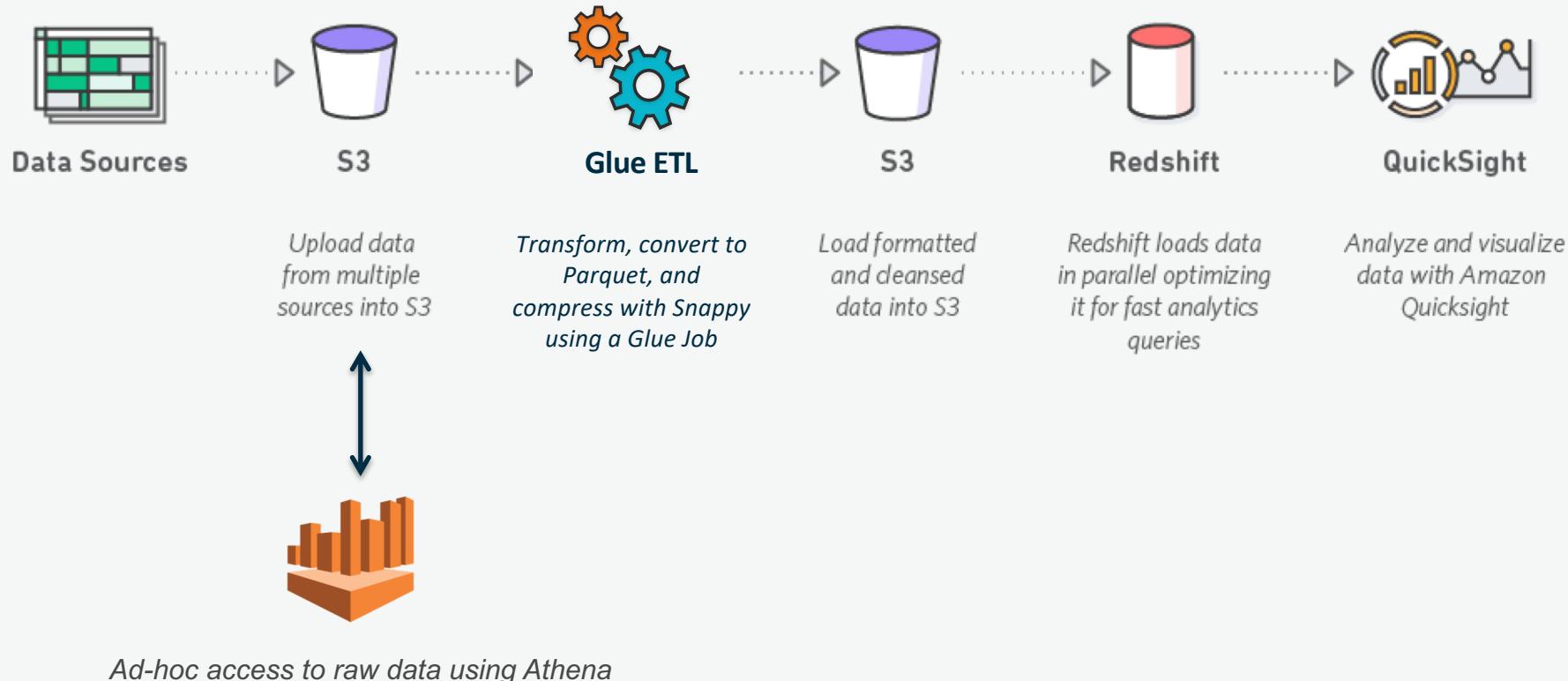


Serverless: zero infrastructure, zero administration
Integrated with QuickSight

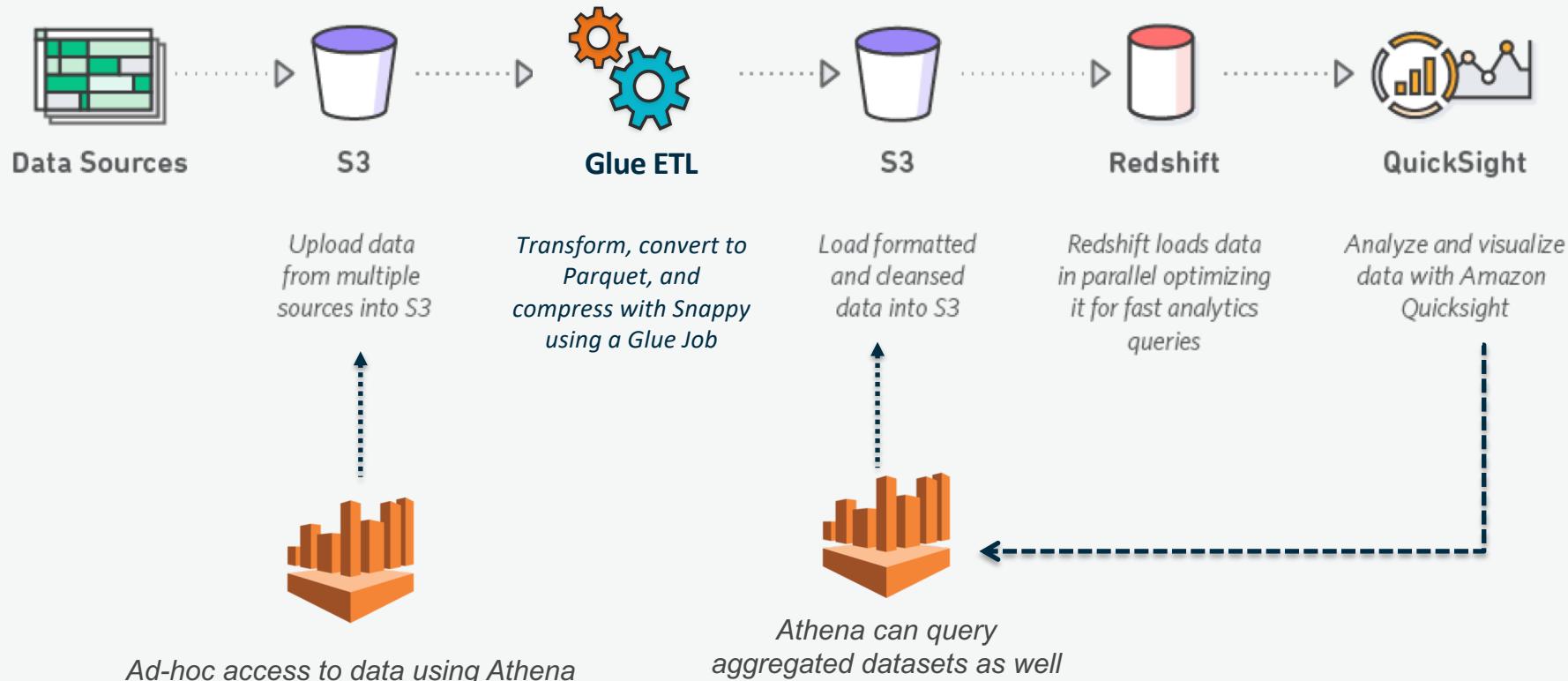
Examples of when you would query the Data Lake...



A Sample Pipeline

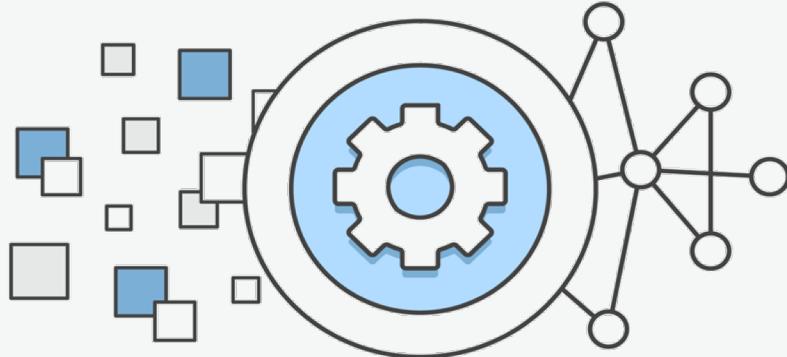


A Sample Pipeline



Athena is Serverless

- No Infrastructure or administration
- Zero Spin up time
- Transparent upgrades



Amazon Athena is Cost Effective

- Pay per query
- \$5 per TB scanned from S3
- DDL Queries and failed queries are free
- Save by using compression, columnar formats, partitions

Recap: The core of a Data Lake in AWS

Versatile
Compute
Layers



Athena



Amazon EMR



AWS Glue ETL



Amazon Redshift

Data Lake

Data &
Metadata



Amazon S3



AWS Glue
Data Catalog

Lab Time

<https://github.com/dbayardAWS/intro-data-lake>

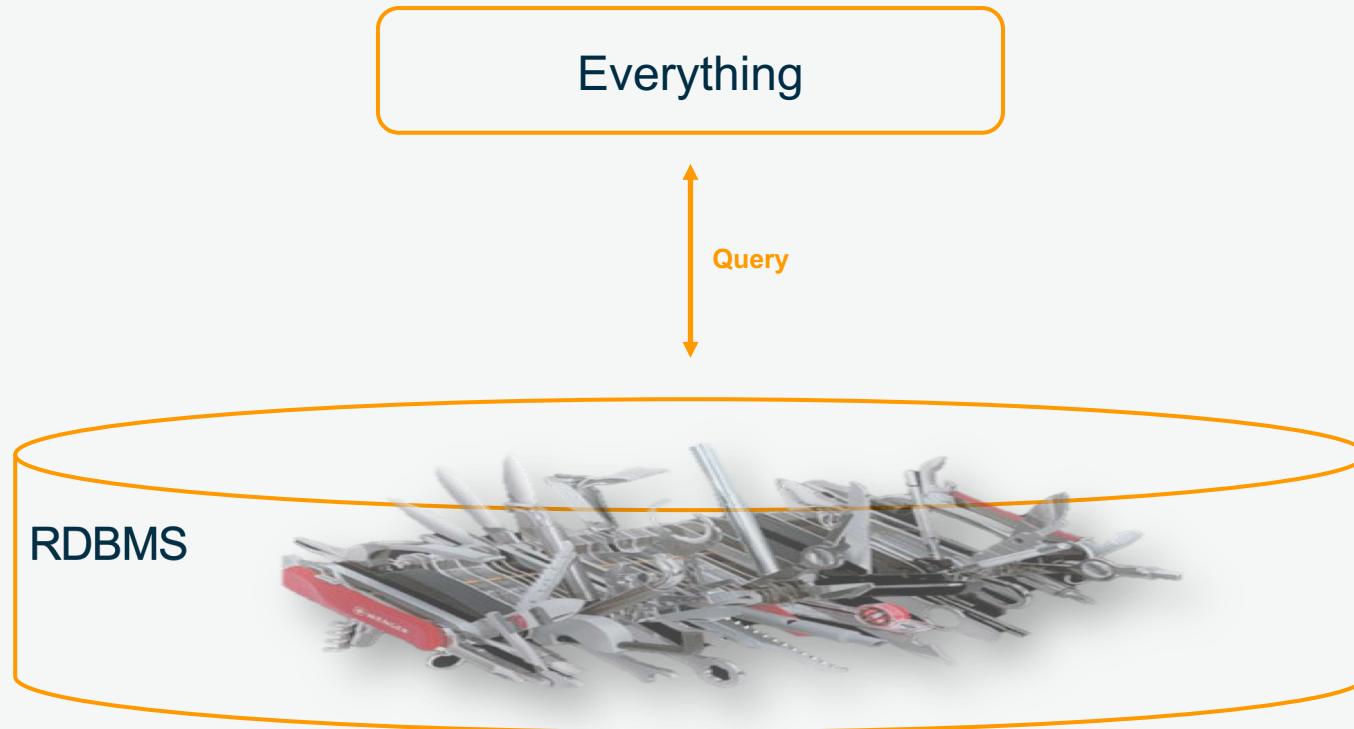
Please resume Lab1.

Your next step should be **Catalog our new dataset.**

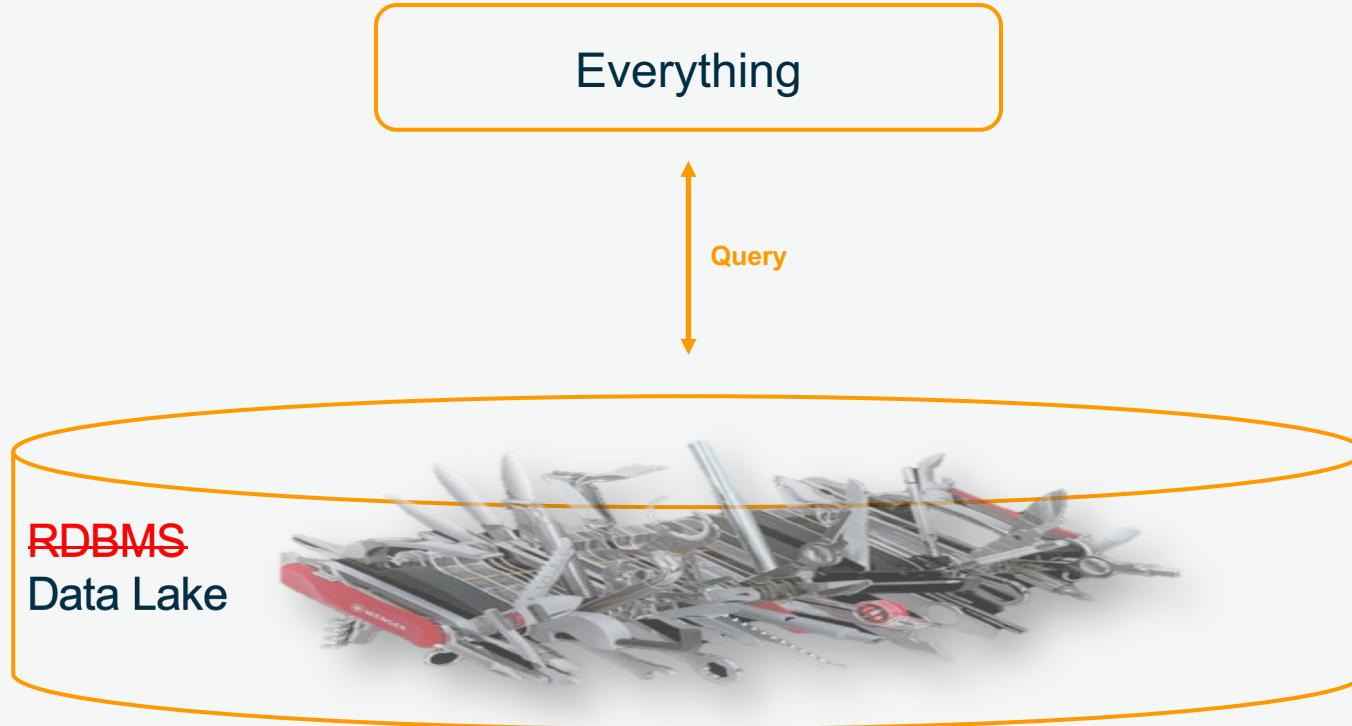
Finish Lab1 (the last exercise is optional).

Cloud Data Warehouse

Anti-Pattern



Also an Anti-Pattern



Data Lakes and Data Reservoirs

Ingestion Triangle:
More work at the top.
Less work at the bottom.

Data Warehouse is a
kind of Data Reservoir

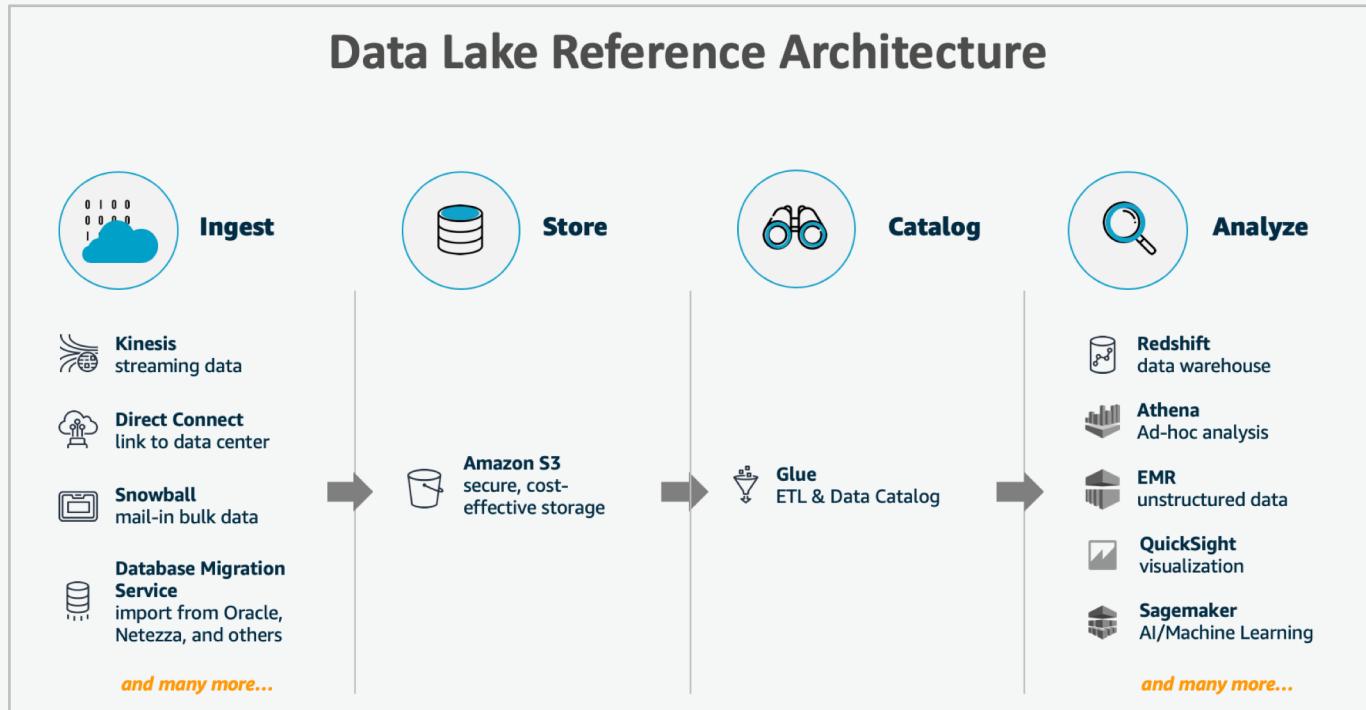


Data Lakes are optimized for large volumes of storage and ease of ingestion of any kind or flavor of data.

Data Warehouses are optimized for ease of retrieval of more critical and useful data.

Analytics Triangle:
Less work at the top.
More work at the bottom.

Amazon Redshift



Amazon Redshift is a cloud-native data warehouse that's...



Most popular

More than 15K customers use Redshift and process more than 2 EB of data per day



Fastest

2-3x faster than other cloud DW solutions



Most cost-effective

25% less than other solutions using on-demand pricing and 75% with reserved instances (RIs).



Integrated with the data lake

Query exabytes of data directly in open formats with no loading required

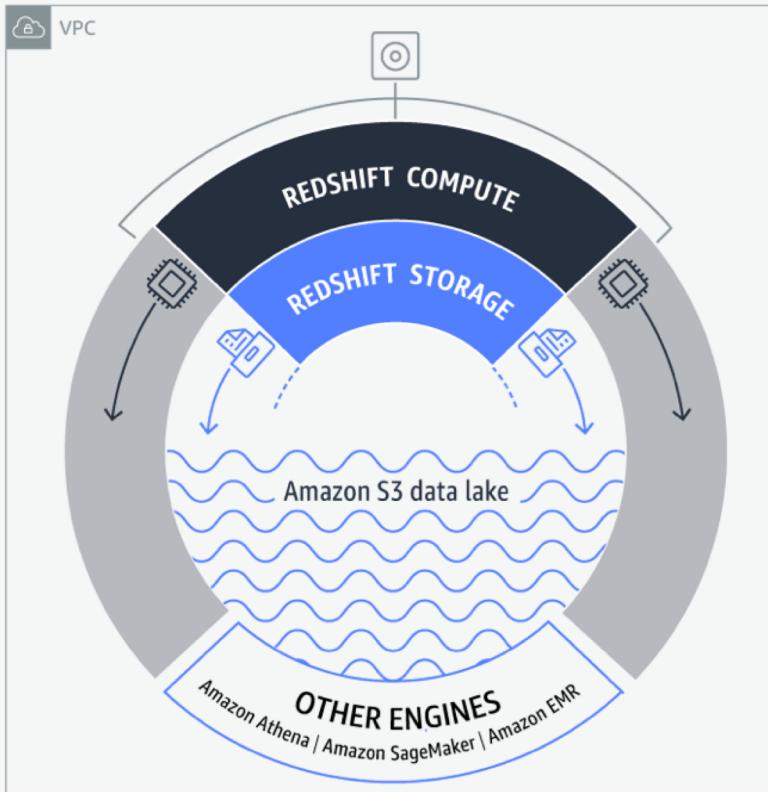
Lab Time

<https://github.com/dbayardAWS/intro-data-lake>

Please start Lab2.

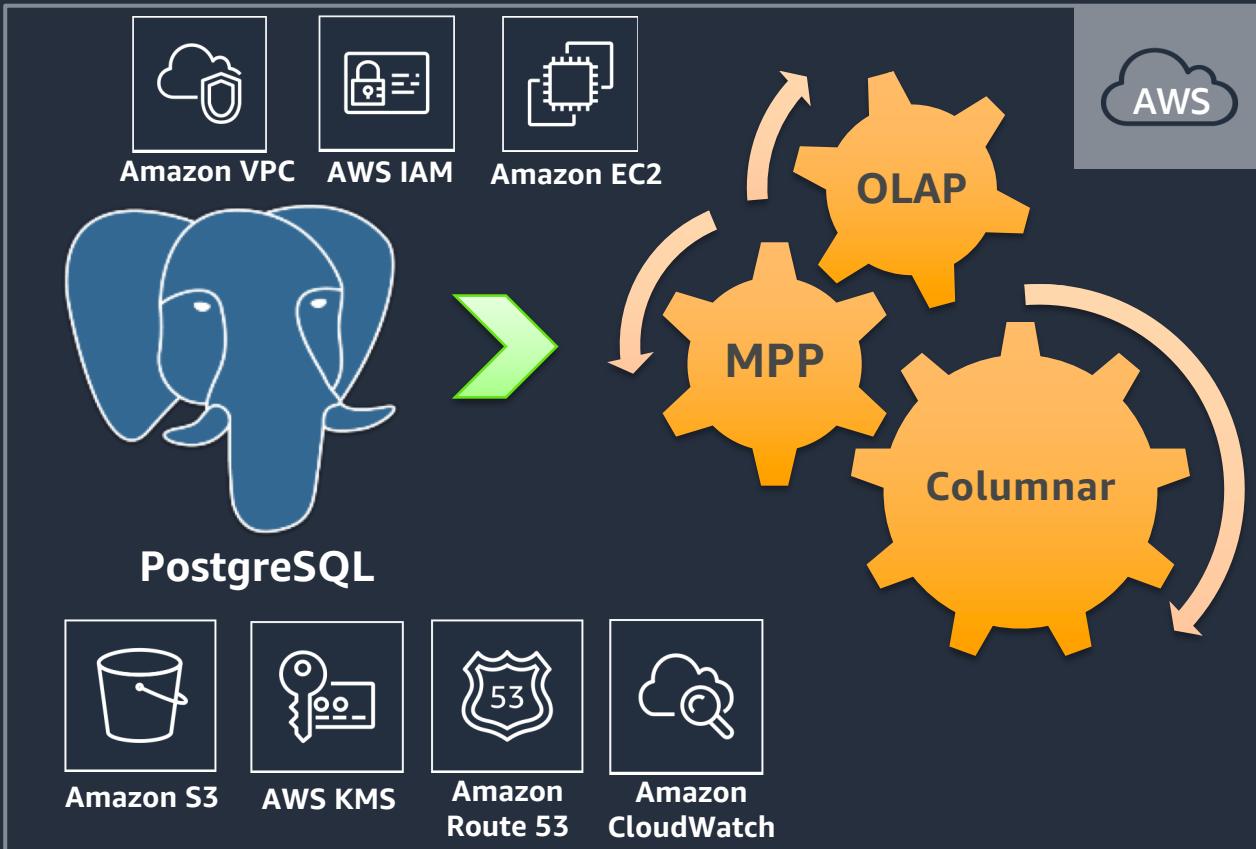
Continue thru the Provision a new Redshift Cluster step.

Redshift Conceptual Architecture





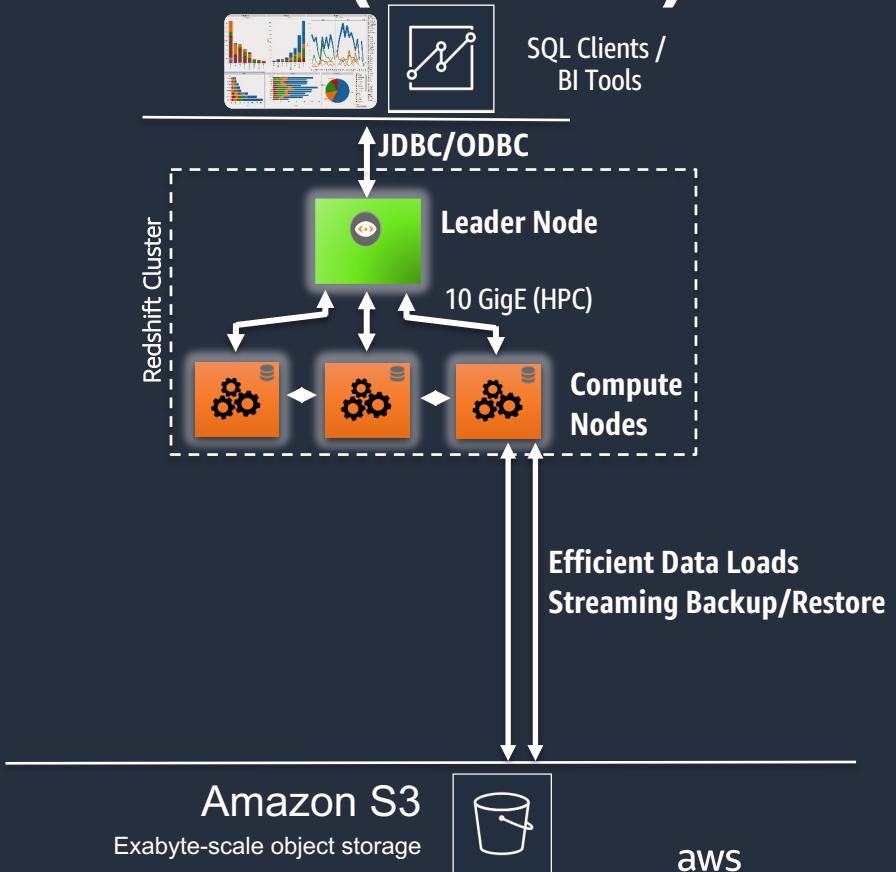
Redshift: What's Under the Hood?



Redshift Cluster Architecture (Classic)



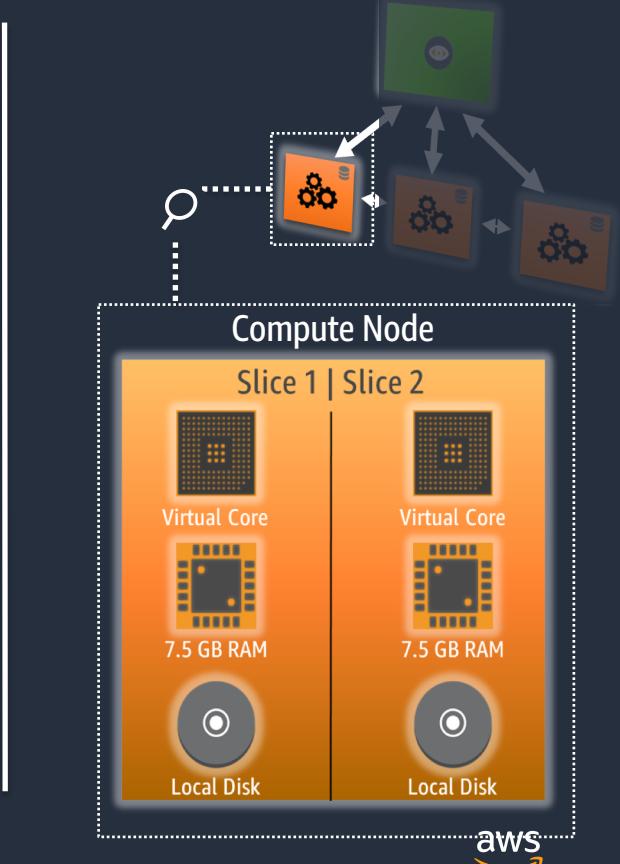
- Massively parallel, shared nothing architecture
- Streaming Backup/Restore from S3
- Leader node
 - SQL endpoint
 - Stores metadata
 - Coordinates parallel SQL processing; ML optimizations
 - Customers are not charged for the leader node for any cluster with two or more nodes
- Compute nodes
 - Local, columnar storage
 - Executes queries in parallel
 - Load, backup, restore





Compute Node: Under the Hood (Slices)

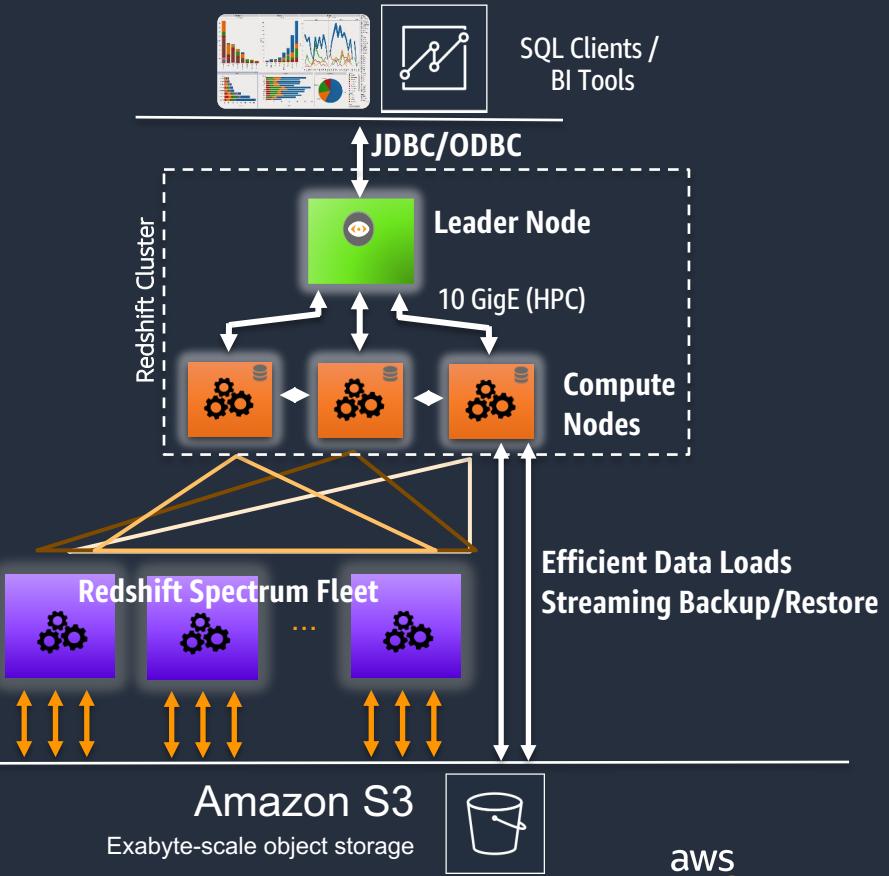
- A compute node is partitioned into either 2 or 16 *slices*; a slice can be thought of as a “virtual compute node”
- Each slice is allocated a portion of the compute node's memory and disk space, where it processes a portion of the workload assigned to the compute node by the leader node
- The leader node manages distributing data to the slices and apportions the workload for any queries or other database operations to the slices
- Slices work in parallel to complete operations





Redshift Cluster Architecture

- Massively parallel, shared nothing architecture
- Streaming Backup/Restore from S3
- Leader node
 - SQL endpoint
 - Stores metadata
 - Coordinates parallel SQL processing; ML optimizations
 - Customers are not charged for the leader node for any cluster with two or more nodes
- Compute nodes
 - Local, columnar storage
 - Executes queries in parallel
 - Load, backup, restore
- Amazon Redshift Spectrum nodes
 - Execute queries directly against data lake





Redshift Spectrum Overview

- Redshift Spectrum is a feature of Amazon Redshift that allows SQL queries to reference external data on Amazon S3 as they would any other table in Amazon Redshift
- Allows for querying of potentially Exabytes of data in an S3 data lake from within Amazon Redshift
- Data is queried in-place, so no loading of data into your Redshift cluster is required
- Powered by a fleet of Amazon Redshift Spectrum nodes

There is a soft limit/allowance of 10 spectrum nodes per Redshift cluster slice. So, a cluster with 20 slices will be allowed to leverage up to 200 Spectrum nodes

Run SQL queries directly against data in S3 using thousands of nodes



Fast @ Exabyte scale



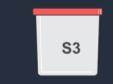
Elastic & highly available



On-demand, pay-per-query



High concurrency: Multiple clusters access same data



No ETL: Query data in-place using open file formats



I ❤️ SQL
Full Amazon Redshift SQL support

Lab Time

<https://github.com/dbayardAWS/intro-data-lake>

Please finish Lab2.

Lunch

We will start again at 12:30pm

Restart

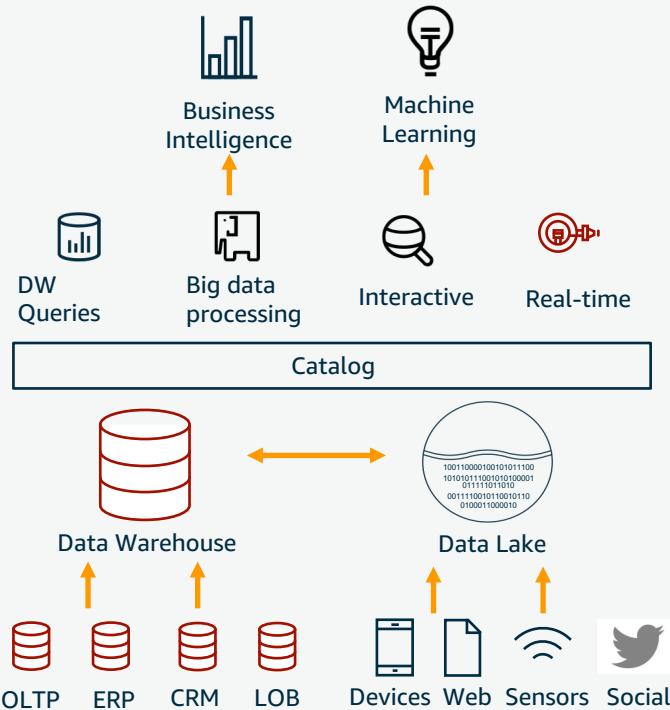
What did you think of the labs?
Any questions about the content so far?

AWS Lake Formation

Agenda

- Data Lake Process Steps
- AWS Lake Formation Overview
- AWS Lake Formation Demonstration

Why data lakes?



Data Lakes provide:

Relational and non-relational data

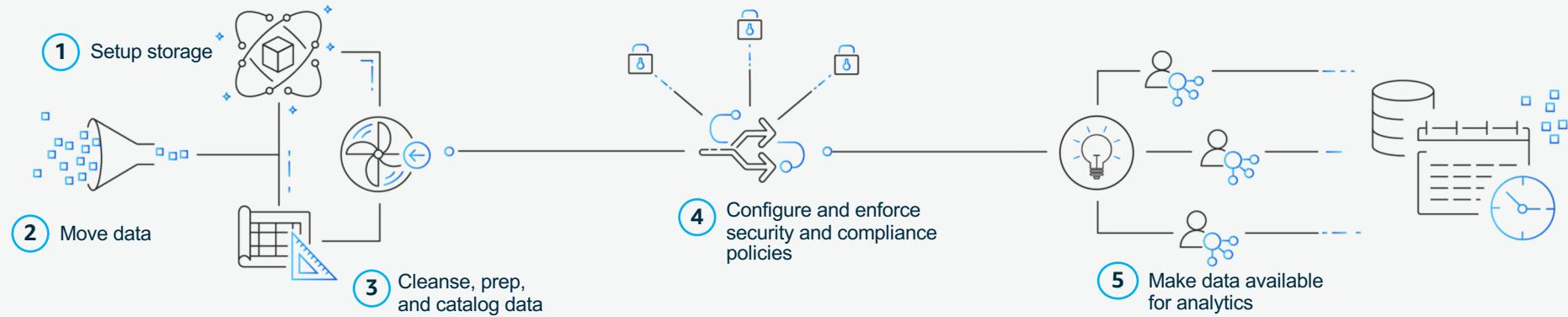
Scale-out to Exabytes

Diverse set of analytics and machine learning tools

Work on data without any data movement

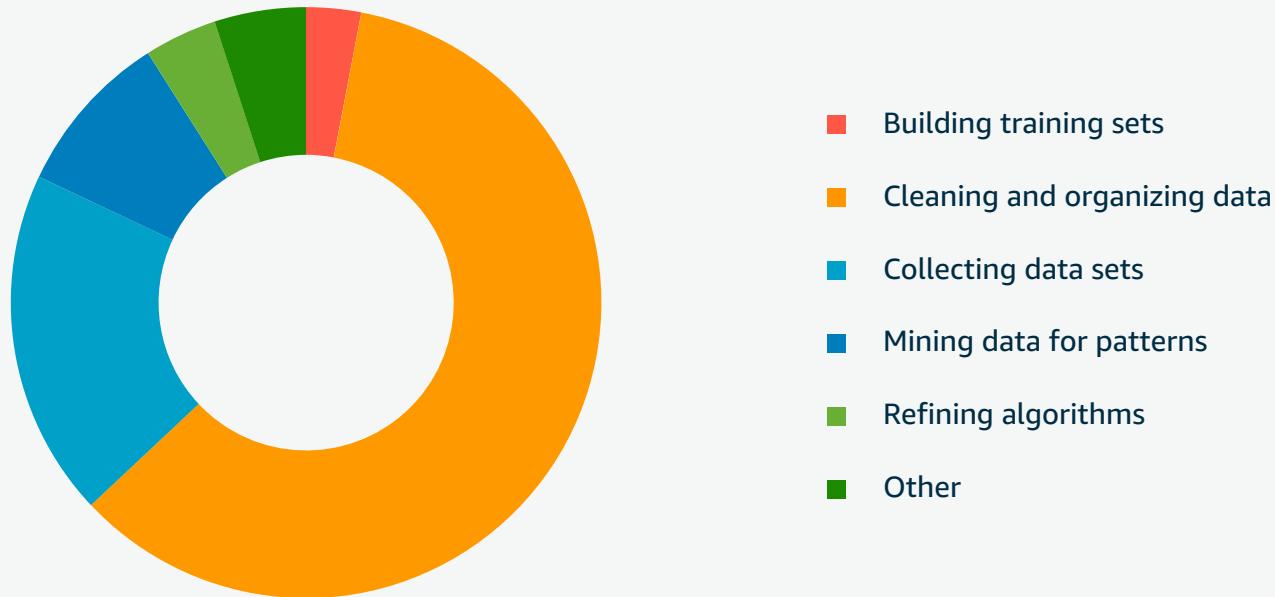
Designed for low cost storage and analytics

Typical steps of building a data lake



Data Preparation Accounts for ~80% of the Work

Building data lakes can still take months



Sample of steps required

Configure access from analytics services

Rinse and repeat for other:
data sets, users, and end-services

And more:

- manage and monitor ETL jobs
- update metadata catalog as data changes
- update policies across services as users and permissions change
- manually maintain cleansing scripts
- create audit processes for compliance

...

Manual | Error-prone | Time consuming

[Feedback](#) [English \(US\)](#)

© 2008 - 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

[Feedback](#) [English \(US\)](#)

© 2008 - 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

AWS Lake Formation

Build a secure data lake in days



Identify, ingest, clean, and transform data

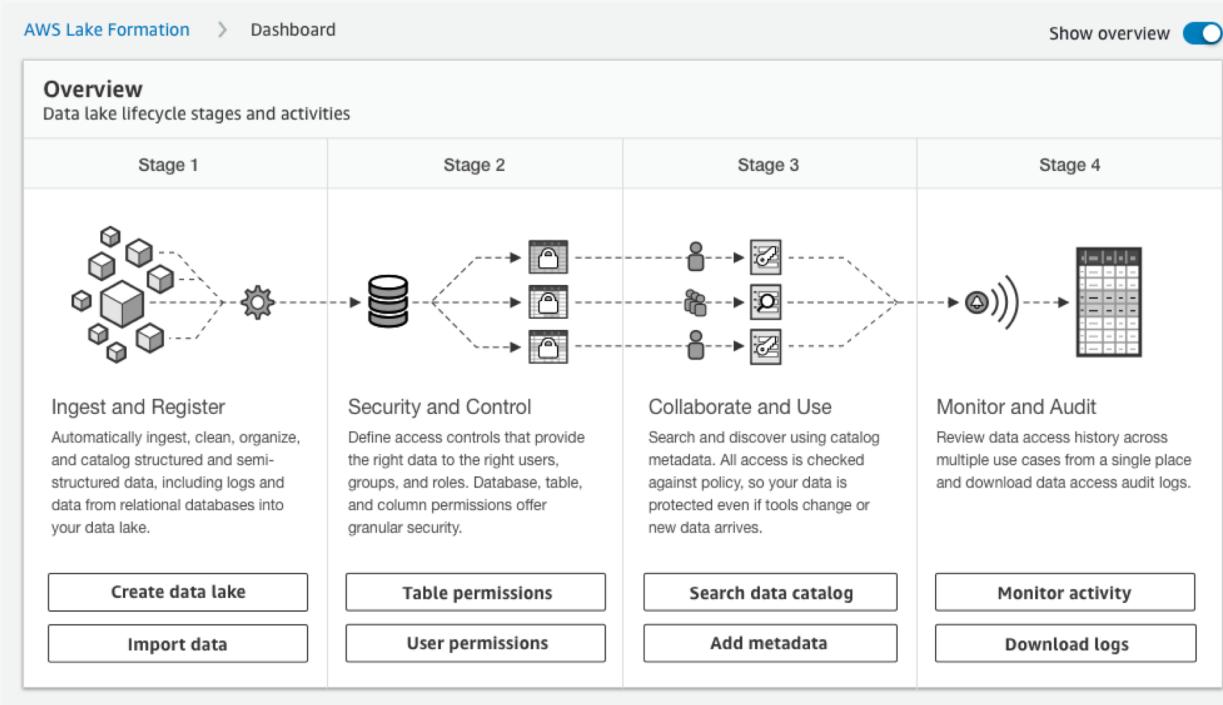


Enforce security policies across multiple services



Gain and manage new insights

How it works



Demonstration Data Set

Oracle SQL Developer File Edit View Navigate Run Source Team Tools Window Help SVCO 100% Mon Jul 8 11:45

Oracle SQL Developer : RDS_ORCL_Bayard_Master

Connections

- RDS_ORCL_Bayard_Master
 - Tables (Filtered)
 - Views
 - Indexes
 - Packages
 - Procedures
 - Functions
 - Operators
 - Queues
 - Queues Tables
 - Triggers
 - Types
 - Sequences
 - Materialized Views
 - Materialized View Logs
 - Synonyms

Reports

- All Reports
- Analytic View Reports
- Data Dictionary Reports
- Data Modeler Reports
- OLAP Reports
- TimesTen Reports
- User Defined Reports

Welcome Page RDS_ORCL_Bayard_Master

Worksheet Query Builder

```
select * from ecommerce order by yearly_amount_spent desc;
```

Query Result All Rows Fetched: 20 in 0.201 seconds

	TIME_ON_WEBSITE	YEARLY_AMOUNT_SPENT	ADDRESS	AVATARR	LENGTH_OF_MEMBER
1	34.47687762925054	637.102447915074	645 Martha Park Apt. 611, Jeffreychester, MN 67218-7250	FloralWhite	5.4935072013
2	37.190503800397956	605.061038804892	544 Alexander Heights Suite 768, North Johnview, MT 57912	LightSeaGreen	4.0645485504
3	37.53665330059473	599.4060920457634	14023 Rodriguez Passage, Port Jacobville, PR 37242-1057	MediumAquaMarine	4.4463083183
4	39.57766801952616	587.9510539684005	835 Frank Tunnel, Wrightmouth, MI 82180-9605	Violet	4.08262063295
5	36.72128267790313	581.8523440352177	1414 David Throughway, Port Jason, OH 22070-1220	SaddleBrown	3.1201787827
6	36.144666700041924	573.4158673313865	05302 Dunlap Ferry, New Stephaniehaven, MP 42268	Bisque	3.9185418391
7	37.534497341555735	570.2004089636196	860 Lee Key, West Debra, SD 97450-0495	Salmon	3.27343357774
8	37.37335885854755	549.9041461052942	Unit 6538 Box 8980, DPO AP 09026-4941	Aqua	4.43427343489
9	37.08792607098381	522.3374046069357	Unit 2413 Box 0347, DPO AA 07580-2652	Tomato	3.713209202
10	36.68377615286961	521.5721747578274	68388 Reyes Lights Suite 692, Josephbury, WV 92213-0247	DarkSlateBlue	4.6850172465
11	37.22580613162114	492.6060127179966	26104 Alexander Groves, Alexandriaport, WY 28244-9149	Tomato	2.4826077705
12	37.110597442120856	487.54750486747207	24645 Valerie Unions Suite 582, Cobbborough, DC 99414-7564	Bisque	4.1045432023
13	36.61995708279922	470.452733009554	7773 Powell Springs Suite 190, Samanthaland, ND 44358	DarkBlue	2.4945436466
14	36.213763093698624	461.7807421962299	49558 Ramirez Road Suite 399, Phillipstad, OH 35641-3238	Peru	3.35784684232
15	34.8940927514398	457.84769594494855	6362 Wilson Mountain, Johnsonfurt, GA 15169	PowderBlue	3.13613271648
16	38.24411459434352	452.3156754800354	64475 Andre Club Apt. 795, Port Dannytown, PW 63227	Cyan	1.5165755808
17	37.14516822352819	427.1993848953282	PSC 2734, Box 5255, APO AA 98456-7482	Brown	3.2028060715
18	37.42021557502538	408.6403510726275	6705 Miller Orchard Suite 186, Lake Shanestad, MO 75696-5051	RoyalBlue	4.0464231642
19	38.38513659413844	407.70454754954415	8982 Burton Row, Wilsonton, PW 88606	OliveDrab	2.4208061609
20	37.268958868297744	392.204933443264	4547 Archer Common, Diazchester, CA 06566-8576	DarkGreen	2.664034182



Step 1: Create a Lake Formation Database

The screenshot shows the AWS Lake Formation interface. The left sidebar has a tree view with 'Data catalog' expanded, showing 'Databases' (selected), 'Tables', and 'Settings'. Other collapsed sections include 'Register and ingest' (with 'Data lake locations', 'Blueprints', 'Crawlers', and 'Jobs'), and 'Permissions' (with 'Admins and database creators' and 'Data permissions'). The main area is titled 'Create database' under 'Database details'. It asks to 'Create a database in the Data Catalog.' A 'Name' field is present with placeholder text 'Enter a unique database name'. Below it, a note says names must contain letters (A-Z), numbers (0-9), hyphens (-), or underscores (_), and be less than 256 characters long. A 'Location - optional' section allows choosing an Amazon S3 location with a placeholder 'e.g.: s3://bucket/prefix/' and a 'Browse' button. An 'Description - optional' section has a placeholder 'Enter a description' and notes descriptions can be up to 2048 characters long. The top navigation bar includes 'Services', 'Resource Groups', a user icon, and regions 'N. Virginia' and 'Support'.

Create database

Database details
Create a database in the Data Catalog.

Name

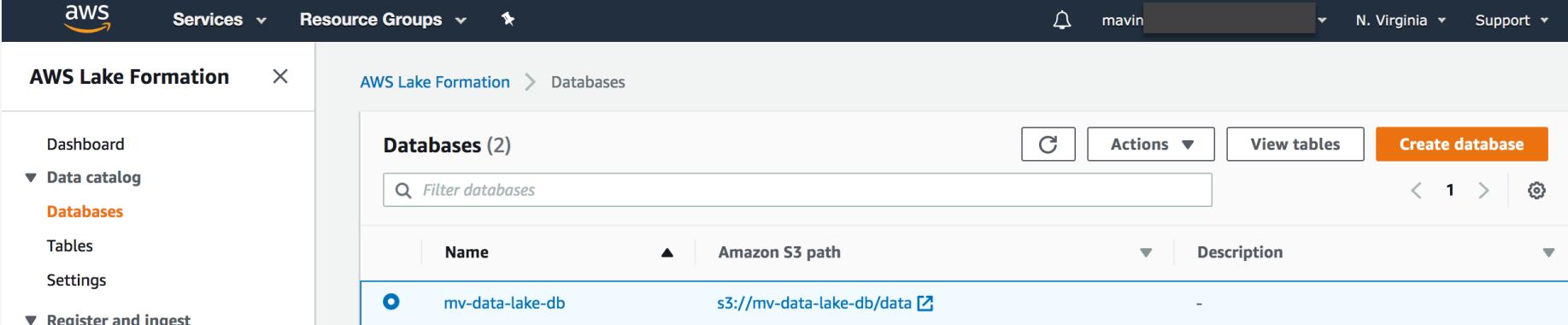
Names may contain letters (A-Z), numbers (0-9), hyphens (-), or underscores (_), and must be less than 256 characters long.

Location - optional
Choose an Amazon S3 location that corresponds to this database. Specifying a location eliminates the need to grant data location privileges on catalog table locations that are children of the database location.
 Browse

Description - optional

Descriptions can be up to 2048 characters long.

Step 1: Create a Lake Formation Database



The screenshot shows the AWS Lake Formation service interface. The left sidebar has a 'Data catalog' section with 'Databases' selected, which is highlighted in orange. The main content area is titled 'Databases (2)' and shows a table with one row. The table columns are 'Name', 'Amazon S3 path', and 'Description'. The single database listed is 'mv-data-lake-db' with the Amazon S3 path 's3://mv-data-lake-db/data'. There are buttons for 'Actions', 'View tables', and 'Create database'.

Name	Amazon S3 path	Description
mv-data-lake-db	s3://mv-data-lake-db/data	-

Step 2: Grant Privileges [Data Formation Database]

Grant permissions

Grant access permissions to specific users and roles.

IAM users and roles
Add one or more IAM users or roles.

Database
Add one or more databases.

Database permissions
Choose the specific access permissions to grant.

Select all Create table Alter Drop
 Grant all

Enabling this permission grants full access to the specified resources while still logging access requests based on the individual permission settings above. It is typically used for debugging. Specific individual permissions set above will take effect when Grant all is later removed.

Grantable permissions
Choose the specific permissions that may be granted to others.

Select all Create table Alter Drop
 Grant all

Enabling this permission grants full access to the specified resources while still logging access requests based on the individual permission settings above. It is typically used for debugging. Specific individual permissions set above will take effect when Grant all is later removed.

Cancel **Grant**

IAM users and roles
Add one or more IAM users or roles.

Users

- amplify-WKlzw User
- codeconfig-dave User
- dbayard User
- dbayardS3 User
- mavinson User

Roles

- Admin Role
- aws-quicksight-service-role-v0 Role
- AWSAmplifyExecutionRole-d25zeyqazpjx3 Role

Database
Add one or more databases.

mv-data-lake-db

Step 2: Grant Privileges [Data Formation Database]

Screenshot of the AWS Lake Formation Permissions page.

The left sidebar shows the AWS Lake Formation navigation menu:

- AWS Lake Formation
- Dashboard
- Data catalog
 - Databases
 - Tables
 - Settings
- Register and ingest
 - Data lake locations
 - Blueprints
 - Crawlers
 - Jobs
- Permissions
 - Admins and database creators
 - Data permissions** (highlighted)
 - Data locations

The main content area is titled "Data permissions (4)" and displays a table of permissions for the database "mv-data-lake-db".

Table Headers:

Principal	Principal type	Resource type	Resource	Permissions	Grantable
-----------	----------------	---------------	----------	-------------	-----------

Table Data:

Admin	IAM role	Database	mv-data-lake-db	All, Drop, Create table, Alter	All, Drop, Create table, Alter
WordpressBlueprintRole	IAM role	Database	mv-data-lake-db	All, Drop, Create table, Alter	All, Drop, Create table, Alter
AWSServiceRoleForLakeFormationDataAccess	IAM role	Database	mv-data-lake-db	All, Drop, Create table, Alter	All, Drop, Create table, Alter
mavinson	IAM user	Database	mv-data-lake-db	All, Drop, Create table, Alter	All, Drop, Create table, Alter

An orange oval highlights the last row of the table, which corresponds to the "mavinson" IAM user.

Step 3: Create a Data Lake Location

The screenshot shows the AWS Lake Formation interface. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, 'Resource Groups' dropdown, a bell icon, a user name 'mavins', and region 'N. Virginia' dropdown. Below the navigation is a breadcrumb trail: 'AWS Lake Formation > Data lake locations > Register location'. The main title is 'Register location'. The first section is 'Amazon S3 storage' with the sub-instruction 'Register Amazon S3 storage as your data lake.' Below it is 'Amazon S3 path' with a text input field containing 'e.g.: s3://bucket/prefix/' and a 'Browse' button. There's also an 'Audit location permissions' button. The next section is 'Existing location permissions - optional' with the note 'Registering the selected location may result in your users gaining access to data already at that location. Before registering the location, we recommend that you audit permissions on resources in the location.' The final section is 'IAM role' with the note 'To add or update data, Lake Formation needs read/write access to the chosen Amazon S3 path. Choose a role that you know has permission to do this, or choose the AWSServiceRoleForLakeFormationDataAccess service-linked role. When you register the first Amazon S3 path, the service-linked role and a new inline policy are created on your behalf. Lake Formation adds the first path to the inline policy and attaches it to the service-linked role. When you register subsequent paths, Lake Formation adds the path to the existing policy.' A dropdown menu is open, showing 'AWSServiceRoleForLakeFormationDataAccess' as the selected option. At the bottom are 'Cancel' and 'Register location' buttons.

AWS Lake Formation > Data lake locations > Register location

Register location

Amazon S3 storage
Register Amazon S3 storage as your data lake.

Amazon S3 path
Choose an Amazon S3 location for your data lake.

e.g.: s3://bucket/prefix/

Existing location permissions - optional
Registering the selected location may result in your users gaining access to data already at that location. Before registering the location, we recommend that you audit permissions on resources in the location.

IAM role
To add or update data, Lake Formation needs read/write access to the chosen Amazon S3 path. Choose a role that you know has permission to do this, or choose the **AWSServiceRoleForLakeFormationDataAccess** service-linked role. When you register the first Amazon S3 path, the service-linked role and a new inline policy are created on your behalf. Lake Formation adds the first path to the inline policy and attaches it to the service-linked role. When you register subsequent paths, Lake Formation adds the path to the existing policy.

▾

Step 3: Create a Data Lake Location

The screenshot shows the AWS Lake Formation interface. The left sidebar has a tree view with 'Data catalog' expanded, showing 'Databases', 'Tables', and 'Settings'. Under 'Register and ingest', 'Data lake locations' is selected, which is highlighted in orange. Other options include 'Blueprints', 'Crawlers' (with a blue arrow icon), and 'Jobs' (with a blue arrow icon). The main content area is titled 'Data lake locations (1)'. It includes a search bar with the placeholder 'Filter data lake storage', a 'Actions' dropdown menu, and an orange 'Register location' button. Below these are three filter dropdowns: 'Amazon S3 path', 'IAM role', and 'Last modified'. A table lists one data lake location: 's3://mv-data-lake/processed' (with a blue arrow icon), associated with the IAM role 'AWSServiceRoleForLakeFormatio...' (with a blue arrow icon), and last modified on 'Fri, 05 Jul 2019 19:48:36 GMT'. Navigation icons for back, forward, and refresh are also present.

Amazon S3 path	IAM role	Last modified
s3://mv-data-lake/processed	AWSServiceRoleForLakeFormatio...	Fri, 05 Jul 2019 19:48:36 GMT

Step 4: Create a Glue Database Connection

The screenshot shows the AWS Glue Connections page. On the left sidebar, under the 'Connections' section, there is a blue button labeled 'Edit'. The main content area displays the configuration for a JDBC connection named 'mv-database-connection'. The configuration includes:

Setting	Value
Type	JDBC
JDBC URL	jdbc:oracle:thin://@[REDACTED]aws.com:1521/mvdb
VPC Id	vpc-01a197cb538eccc09
Subnet	subnet-01e6a88ab30005f43
Security groups	sg-0220e49251f34b6b7
Require SSL connection	false
Description	
Username	master
Created	6 July 2019 12:26 AM UTC-4
Last modified	6 July 2019 12:59 AM UTC-4

Step 5: Create a Blueprint

You

1. Point us to the source
2. Tell us the location to load to in your data lake
3. Specify how often you want to load the data

Blueprints

1. Discover the source table(s) schema
2. Automatically convert to the target data format
3. Automatically partition the data based on the partitioning schema
4. Keep track of data that was already processed
5. You can customize any of the above

Step 5: Create a Blueprint

The screenshot shows the AWS Lake Formation console. The left sidebar is titled "AWS Lake Formation" and contains the following navigation items:

- Dashboard
- Data catalog** (selected)
- Databases
- Tables
- Settings
- Blueprints** (selected)
- Crawlers [2]
- Jobs [2]
- Permissions**
- Admins and database creators
- Data permissions
- Data locations

The main content area has a breadcrumb path: AWS Lake Formation > Blueprints. It features two sections: "Blueprint overview" and "Workflows".

Blueprint overview: Describes Blueprints as enabling data ingestion from common sources using automated workflows. It includes a "Database blueprints" section and a "Log file blueprints" section.

Database blueprints: Describes ingesting data from MySQL, PostgreSQL, Oracle, and SQL server databases to your data lake, either as bulk load snapshot, or incrementally load new data over time. A "Use blueprint" button is present.

Log file blueprints: Describes ingesting data from popular log file formats from AWS CloudTrail, Elastic Load Balancer, and Application Load Balancer logs.

Workflows (2): Shows a list of workflows. The table has columns: Name, Created on, Last updated, and Last run status. Two workflows are listed:

Name	Created on	Last updated	Last run status
lakeformationparq...	Sat, 06 Jul 2019 14:43:31 GMT	Sat, 06 Jul 2019 14:43:31 GMT	✓ COMPLETED
lakeformationdemo...	Sat, 06 Jul 2019 14:10:03 GMT	Sat, 06 Jul 2019 14:10:03 GMT	✓ COMPLETED

Actions buttons include a refresh icon, Actions dropdown, and a "Use blueprint" button.

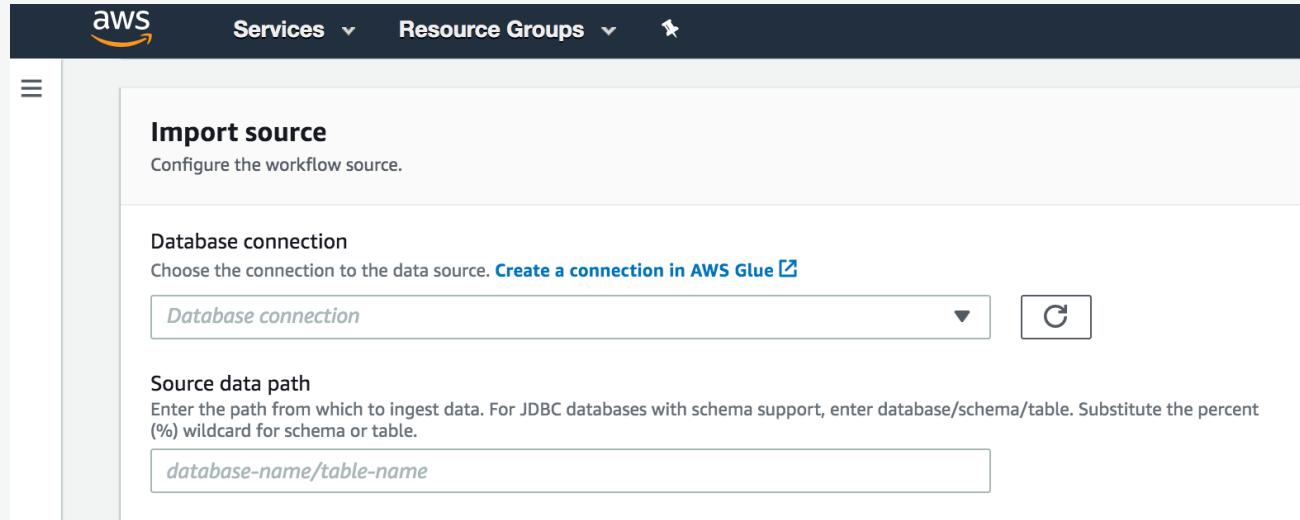
Step 5: Create a Blueprint (continued)

The screenshot shows the AWS Lake Formation interface. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, 'Resource Groups' dropdown, and a search icon. Below the navigation, the breadcrumb path is 'AWS Lake Formation > Blueprints > Use a blueprint'. The main title 'Use a blueprint' is centered above a section titled 'Blueprint type'. This section contains five options, each in its own box:

- Database snapshot** (selected): Bulk load data to your data lake from MySQL, PostgreSQL, Oracle, and Microsoft SQL Server databases.
- Incremental database**: Load new data to your data lake from MySQL, PostgreSQL, Oracle, and SQL Server databases.
- AWS CloudTrail**: Bulk load data from AWS CloudTrail sources.
- AWS ELB logs**: Load data from Elastic Load Balancer logs.
- AWS ALB logs**: Load data from Application Load Balancer logs.



Step 5: Create a Blueprint (continued)



The screenshot shows the 'Import source' configuration page in the AWS Glue console. The 'Database connection' section is expanded, showing a dropdown menu with 'Database connection' selected and a 'Create' button. The 'Source data path' section is also visible, with a text input field containing 'database-name/table-name'. A modal window titled 'Database connection' is displayed on the right, listing a single connection named 'mv-database-connection'.

Import source
Configure the workflow source.

Database connection
Choose the connection to the data source. [Create a connection in AWS Glue](#)

Database connection

Source data path
Enter the path from which to ingest data. For JDBC databases with schema support, enter database/schema/table. Substitute the percent (%) wildcard for schema or table.

database-name/table-name

Exclude patterns - optional
Specify a pattern to limit the workflow. Tables that match this pattern will be excluded.

Exclude pattern

Enter an exclude pattern

Add exclusion

Remove exclusion

Database connection
Choose the connection to the data source. [Create a connection in AWS Glue](#)

Database connection

Filter database connections

mv-database-connection

Step 5: Create a Blueprint (continued)

The screenshot shows the 'Import target' configuration screen in the AWS Data Pipeline console. The top navigation bar includes the AWS logo, 'Services' dropdown, 'Resource Groups' dropdown, and a search bar.

Import target
Configure the target of the workflow.

Target database
Choose a database in the Data Catalog. [Create database](#)

Data lake location
Choose where to import from your data lake storage locations.

Data format
Choose the output data format.

Import frequency
Schedule the workflow.

Frequency
Choose how often to run the workflow.

Target database
Choose a database in the Data Catalog. [Create database](#)

Data lake location
Choose where to import from your data lake storage locations.

IAM role: AWSLambdaRoleForLakeFormationDataAccess

Data format
Choose the output data format.

Frequency
Choose how often to run the workflow.

aws

Step 5: Create a Blueprint (continued)

aws Services Resource Groups ⚙

Import options
Configure the workflow.

Workflow name

Names may contain letters (A-Z), numbers (0-9), hyphens (-), or underscores (_), and must be less than 256 characters long.

IAM role

Table prefix
The table prefix that is used for catalog tables that are created.

Maximum capacity - optional
Sets the number of data processing units (DPUs) that can be allocated when this job runs. A DPU is a relative measure of processing power that consists of 4 vCPUs of compute capacity and 16 GB of memory.

Concurrency - optional
Sets the maximum number of concurrent runs that are allowed for this job. An error is returned when this threshold is reached. The default is 5.

Cancel **Create**



Lake Formation Setup and Ingestion is COMPLETE

AWS Services Resource Groups N. Virginia Support

AWS Lake Formation X

AWS Lake Formation > Tables

Tables (12)

Filter by tags and attributes or search by keyword

Create table using a crawler Create table

Name	Location	Classification	Last updated
june06_mvdb_master_runda...	s3://mv-data-lake/proces...	CSV	Sat, 06 Jul 2019 14:34:47 GMT
june06_mvdb_master_ecom...	s3://mv-data-lake/proces...	CSV	Sat, 06 Jul 2019 14:34:18 GMT
temp_june06_mvdb_master_...	s3://mv-data-lake/proces...	CSV	Sat, 06 Jul 2019 14:34:19 GMT
temp_june06_mvdb_master_...	s3://mv-data-lake/proces...	CSV	Sat, 06 Jul 2019 14:33:16 GMT
temp_parquet_mvdb_master...	s3://mv-data-lake/proces...	PARQUET	Sat, 06 Jul 2019 15:08:01 GMT
parquet_mvdb_master_ecom...	s3://mv-data-lake/proces...	PARQUET	Sat, 06 Jul 2019 15:08:01 GMT
temp_parquet_mvdb_master...	s3://mv-data-lake/proces...	PARQUET	Sat, 06 Jul 2019 15:07:06 GMT
parquet_mvdb_master_rund...	s3://mv-data-lake/proces...	PARQUET	Sat, 06 Jul 2019 15:08:30 GMT
june06_source_schema_mvdb...	mvdb.MASTER.ECOMMER...	oracle	Sat, 06 Jul 2019 14:18:27 GMT
june06_source_schema_mvdb...	mvdb.MASTER.RUNDATA...	oracle	Sat, 06 Jul 2019 14:18:27 GMT

Lake Formation Setup and Ingestion is COMPLETE



Services ▾

Resource Groups ▾



mavinsc

Global ▾

Support ▾

Amazon S3 > mv-data-lake > processed

Overview

Type a prefix and press Enter to search. Press ESC to clear.

Upload

+ Create folder

Download

Actions ▾

US East (N. Virginia)

Viewing 1 to 4

<input type="checkbox"/>	Name ▾	Last modified ▾	Size ▾	Storage class ▾
<input type="checkbox"/>	june06_mvdb_master_ecommerce	--	--	--
<input type="checkbox"/>	june06_mvdb_master_rundataevents	--	--	--
<input type="checkbox"/>	parquet_mvdb_master_ecommerce	--	--	--
<input type="checkbox"/>	parquet_mvdb_master_rundataevents	--	--	--

Viewing 1 to 4

Lake Formation Setup and Ingestion is COMPLETE

AWS Services Resource Groups ⚡

mavinson Global Support

Amazon S3 > mv-data-lake > processed > parquet_mvdb_master_ecommerce > version_0

Overview

Type a prefix and press Enter to search. Press ESC to clear.

Upload Create folder Download Actions US East (N. Virginia)

Viewing 1 to 21

<input type="checkbox"/> Name ▾	Last modified ▾	Size ▾	Storage class ▾
<input type="checkbox"/> _temporary	--	--	--
<input type="checkbox"/> part-00000-3a26c8f2-01d2-44b2-bfde-25b46d63389e-c000.snappy.parquet	Jul 6, 2019 11:07:37 AM GMT-0400	2.5 KB	Standard
<input type="checkbox"/> part-00001-3a26c8f2-01d2-44b2-bfde-25b46d63389e-c000.snappy.parquet	Jul 6, 2019 11:07:41 AM GMT-0400	2.5 KB	Standard
<input type="checkbox"/> part-00002-3a26c8f2-01d2-44b2-bfde-25b46d63389e-c000.snappy.parquet	Jul 6, 2019 11:07:41 AM GMT-0400	2.4 KB	Standard
<input type="checkbox"/> part-00003-3a26c8f2-01d2-44b2-bfde-25b46d63389e-c000.snappy.parquet	Jul 6, 2019 11:07:41 AM GMT-0400	2.5 KB	Standard
<input type="checkbox"/> part-00004-3a26c8f2-01d2-44b2-bfde-25b46d63389e-c000.snappy.parquet	Jul 6, 2019 11:07:53 AM GMT-0400	2.5 KB	Standard
<input type="checkbox"/> part-00005-3a26c8f2-01d2-44b2-bfde-25b46d63389e-c000 snappy parquet	Jul 6, 2019 11:07:56 AM GMT-0400	2.6 KB	Standard

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved. Amazon Confidential and Trademark

Grant Privileges to Lake Formation Tables

Grant permissions parquet_mvdb_master_ecommerce X

Grant access permissions to specific users and roles.

IAM users and roles
Add one or more IAM users or roles.

Column - optional
Grant permissions to:

Table permissions
Choose the specific access permissions to grant.
 Select all Alter Insert Drop
 Delete Select
 Grant all
Enabling this permission grants full access to the specified resources while still logging access requests based on the individual permission settings above. It is typically used for debugging. Specific individual permissions set above will take effect when Grant all is later removed.

Grantable permissions
Choose the specific permissions that may be granted to others.
 Select all Alter Insert Drop
 Delete Select
 Grant all
Enabling this permission grants full access to the specified resources while still logging access requests based on the individual permission settings above. It is typically used for debugging. Specific individual permissions set above will take effect when Grant all is later removed.

Grant table and column-level permissions

Column - optional
Grant permissions to:

 |



ademark

Grant Privileges to Lake Formation Tables

The screenshot shows the AWS Lake Formation Permissions page. The left sidebar navigation includes: Dashboard, Data catalog (Tables), Settings, Register and ingest (Data lake locations, Blueprints, Crawlers, Jobs), and Permissions. The main content area displays 'Data permissions (4)' for the database 'mv-data-lake-db' and table 'parquet_mvdb_master_ecommerce'. A search bar is at the top, followed by filter buttons for Database and Table. The table lists two entries for the principal 'mavinson' (IAM user). The first entry grants 'Select' permission on the column 'mv-data-lake-db.parquet_mvdb_master_ecommerce.*'. The second entry grants 'Drop, Insert, All, Delete, Alter' permission on the table 'parquet_mvdb_master_ecommerce'. Action buttons 'Clear', 'Revoke', and 'Grant' are available at the top right.

Principal	Principal type	Resource type	Resource	Permissions	Grantable
mavinson	IAM user	Column	mv-data-lake-db.parquet_mvdb_master_ecommerce.*	Select	Select
mavinson	IAM user	Table	parquet_mvdb_master_ecommerce	Drop, Insert, All, Delete, Alter	Drop, Insert, All, Delete, Alter

Welcome Page × RDS_ORCL_Bayard_Master ×

Worksheet Query Builder

```
select * from ecommerce order by yearly_amount_spent desc;
```

Query Result × All Rows Fetched: 20 in 0.201 seconds

SQL

	TIME_ON_WEBSITE	YEARLY_AMOUNT_SPENT	ADDRESS	AVATARR	LENGTH_OF_MEMBE
1	34.47687762925054	637.102447915074	645 Martha Park Apt. 611, Jeffreychester, MN 67218-7250	FloralWhite	5.4935072013
2	37.190503800397956	605.061038004892	544 Alexander Heights Suite 768, North Johnview, MT 57912	LightSeaGreen	4.0645485504
3	37.5366530059473	599.4060920457634	14023 Rodriguez Passage, Port Jacobville, PR 37242-1057	MediumAquaMarine	4.4463083183
4	39.57766801952616	587.9510539684005	835 Frank Tunnel, Wrightmouth, MI 82180-9605	Violet	4.08262063295
5	36.72128267790313	581.8523440352177	1414 David Throughway, Port Jason, OH 22070-1220	SaddleBrown	3.1201787827
6	36.14466670041924	573.4158673313865	05302 Dunlap Ferry, New Stephaniehaven, MP 42268	Bisque	3.9185418391
7	37.534497341555735	570.2004089636196	860 Lee Key, West Debra, SD 97450-0495	Salmon	3.2734357774
8	37.37335885854755	549.9041461052942	Unit 6538 Box 8980, DPO AA 09026-4941	Aqua	4.43427343489
9	37.08792607098381	522.3374046069357	Unit 2413 Box 0347, DPO AA 07580-2652	Tomato	3.713209202
10	36.68377615286961	521.5721747578274	68388 Reyes Lights Suite 692, Josephbury, WV 92213-0247	DarkSlateBlue	4.6850172465
11	37.22580613162114	492.6060127179966	26104 Alexander Groves, Alexandriaport, WY 28244-9149	Tomato	2.4826077705
12	37.110597442120856	487.5475486747207	24645 Valerie Unions Suite 582, Cobbborough, DC 99414-7564	Bisque	4.1045432023
13	36.61995708279922	470.452733009554	7773 Powell Springs Suite 190, Samanthaland, ND 44358	DarkBlue	2.4945436466
14	36.213763093698624	461.7807421962299	49558 Ramirez Road Suite 399, Phillipstad, OH 35641-3238	Peru	3.35784684232
15	34.8940927514398	457.84769594494855	6362 Wilson Mountain, Johnsonfurt, GA 15169	PowderBlue	3.13613271648
16	38.24411459434352	452.3156754800354	64475 Andre Club Apt. 795, Port Dannytown, PW 63227	Cyan	1.5165755808
17	37.14516822352819	427.1993848953282	PSC 2734, Box 5255, APO AA 98456-7482	Brown	3.2028060715
18	37.4021557502538	408.6403510726275	6705 Miller Orchard Suite 186, Lake Shanestad, MO 75696-5051	RoyalBlue	4.06464231642
19	38.38513659413844	407.70454754954415	8982 Burton Row, Wilsonton, PW 88606	OliveDrab	2.4208061609
20	37.268958868297744	392.2049334443264	4547 Archer Common, Diazchester, CA 06566-8576	DarkGreen	2.664034182

temp_parquet_mvdb_master_commerce

temp_parquet_mvdb_master_rundataevents

Views (0) Create view

You have not created any views. To create a view, run a query and click "Create view from query"

https://console.aws.amazon.com/athena/home?region=us-east-1#

mavinso N. Virginia Support

Settings Tutorial Help What's new 10+

Format query Clear

aws

Audit and Monitor in Real Time

Servi^{ces} ▾ Resource Groups ▾

mavini N. Virginia ▾ Support ▾

AWS Lake Formation X

Recent access activity (350)
Recent access activity for your data lake in AWS CloudTrail. Events can take several minutes to appear in CloudTrail and are limited to the last 90 days.

Filter events

Event name	Principal	Alert time
GetInternalTemporaryTableCredentials	mavinson	Mon, 08 Jul 2019 14:45:12 GMT
BatchGrantPermissions	mavinson	Mon, 08 Jul 2019 14:44:19 GMT
GetDataLakePrivileges	mavinson	Mon, 08 Jul 2019 14:44:15 GMT
BatchGrantPermissions	mavinson	Mon, 08 Jul 2019 14:44:08 GMT
GetDataLakePrivileges	mavinson	Mon, 08 Jul 2019 14:44:03 GMT
ListPermissions	mavinson	Mon, 08 Jul 2019 14:43:41 GMT
BatchGrantPermissions	mavinson	Mon, 08 Jul 2019 14:43:17 GMT
GetDataLakePrivileges	mavinson	Mon, 08 Jul 2019 14:43:11 GMT
BatchGrantPermissions	mavinson	Mon, 08 Jul 2019 14:43:00 GMT
GetDataLakePrivileges	mavinson	Mon, 08 Jul 2019 14:42:47 GMT
ListPermissions	mavinson	Mon, 08 Jul 2019 14:42:20 GMT

C View event

Dashboard

▼ Data catalog

Databases

Tables

Settings

▼ Register and ingest

Data lake locations

Blueprints

Crawlers

Jobs

▼ Permissions

Admins and database creators

Data permissions

Data locations

Audit and Monitor in Real Time

Event details

X

```
"eventTime": "2019-07-08T14:44:19Z",
"eventSource": "lakeformation.amazonaws.com",
"eventName": "BatchGrantPermissions",
"awsRegion": "us-east-1",
"sourceIPAddress": "72.21.196.66",
"userAgent": "signin.amazonaws.com",
"requestParameters": {
    "catalogId": "535356700373/datalake",
    "entries": [
        {
            "permissions": [
                "INSERT",
                "ALTER",
                "DROP",
                "DELETE"
            ],
            "permissionsWithGrantOption": [
                "INSERT",
                "ALTER",
                "DROP",
                "DELETE",
                "ALL"
            ],
            "principal": {
                "dataLakePrincipalIdentifier": "arn:aws:iam::535356700373:user/mavinson"
            },
            "resource": {
                "table": {
                    "name": "june06_mvdb_master_ecommerce",
                    "databaseName": "mv-data-lake-db"
                }
            }
        }
    ]
}
```



Customer interest



“We are very excited about the launch of AWS Lake Formation, which provides a **central point of control** to easily load, clean, secure, and catalog data from thousands of clients to our AWS-based data lake, **dramatically reducing our operational load**. ... Additionally, AWS Lake Formation will be **HIPAA compliant** from day one ...”

Aaron Symanski, CTO, Change Healthcare



“I can’t wait for my team to get our hands on AWS Lake Formation. With an enterprise-ready option like Lake Formation, we will be able to **spend more time deriving value from our data** rather than doing the heavy lifting involved in manually setting up and managing our data lake.”

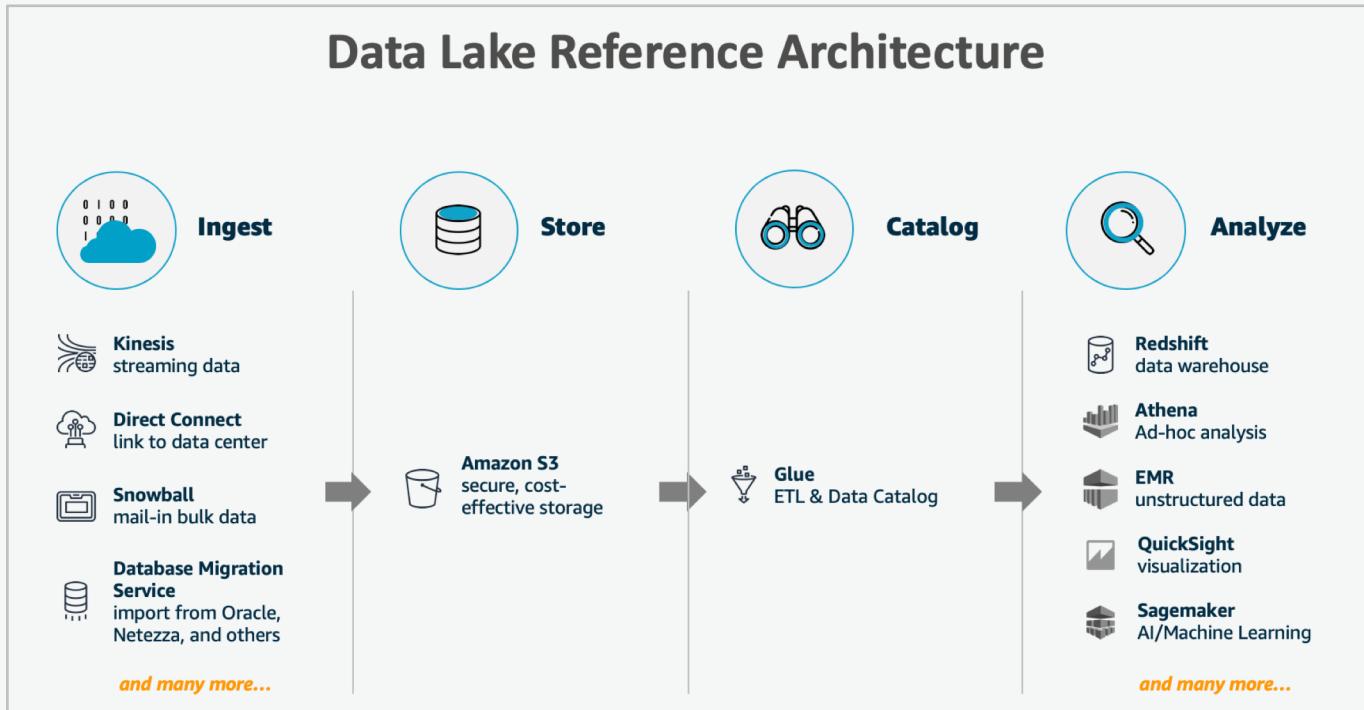
Joshua Couch, VP Engineering, Fender Digital

Thank you!



Populating the Data Lake/Warehouse

Data Ingestion



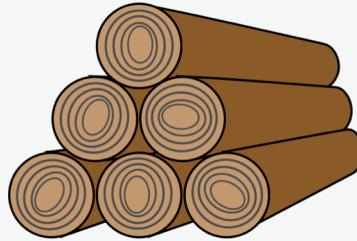
Data Sources



Databases



Streams



Logs



Files

Data Sources - Files



Files



Amazon S3

Files



Optimizing Transfers

- S3 Multi-Part Upload
- S3 Transfer Acceleration
- AWS Direct Connect

Available Services

- AWS DataSync
- AWS Transfer - SFTP
- AWS Snowball/Snowmobile

Data Sources - Streams



Streams



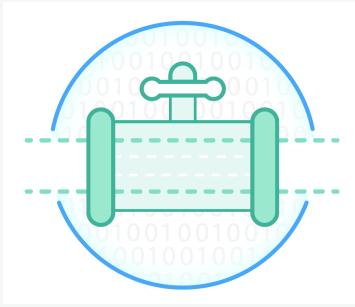
Amazon S3



Collecting and Analyzing

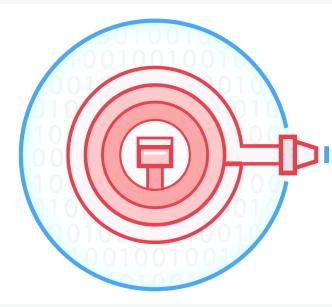
- Amazon Kinesis
- Amazon Managed Streaming for Kafka (MSK)
- Example: Clickstream Analytics

Amazon Kinesis - Stream Processing on AWS



Streams

- Capture streaming data for downstream processing
- Allow multiple processors to read streams at their own rate



Firehose

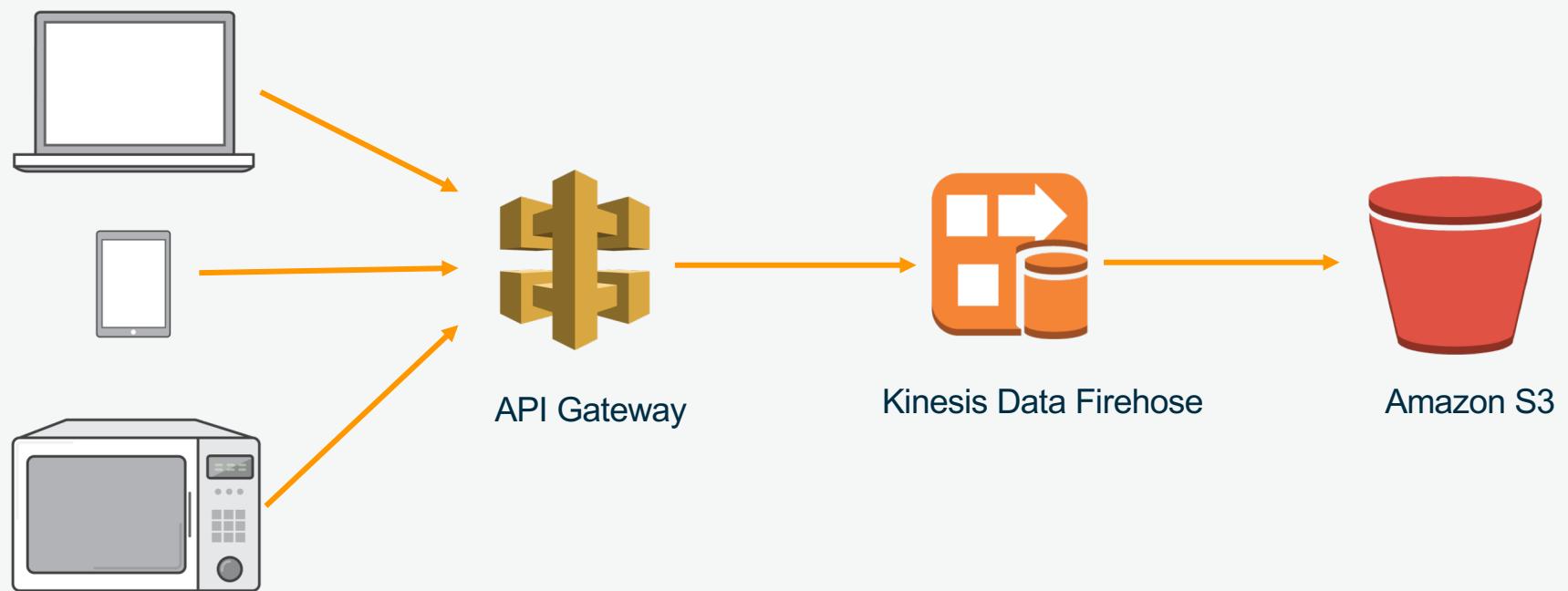
- Buffer records in a stream into a single output for more efficient storage
- Automatic flushing of buffer to S3, ElasticSearch, Redshift, or Splunk



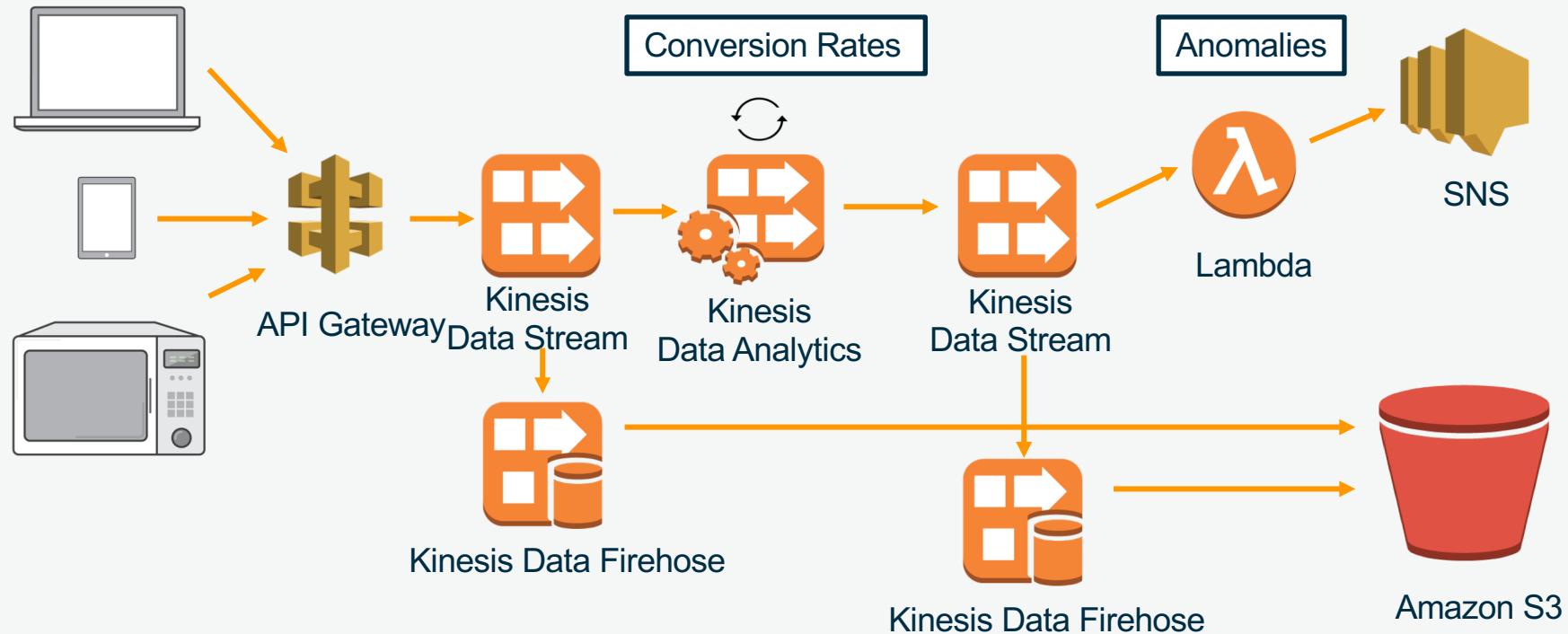
Analytics

- Create time windows over streams and perform aggregate operations using SQL
- Join together multiple streams and output to new streams

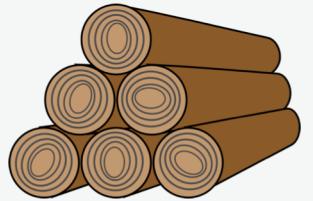
Clickstream



Clickstream with Real-Time Analytics



Data Sources - Logs

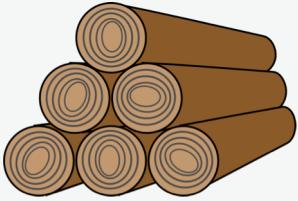


Logs



Amazon S3

Logs



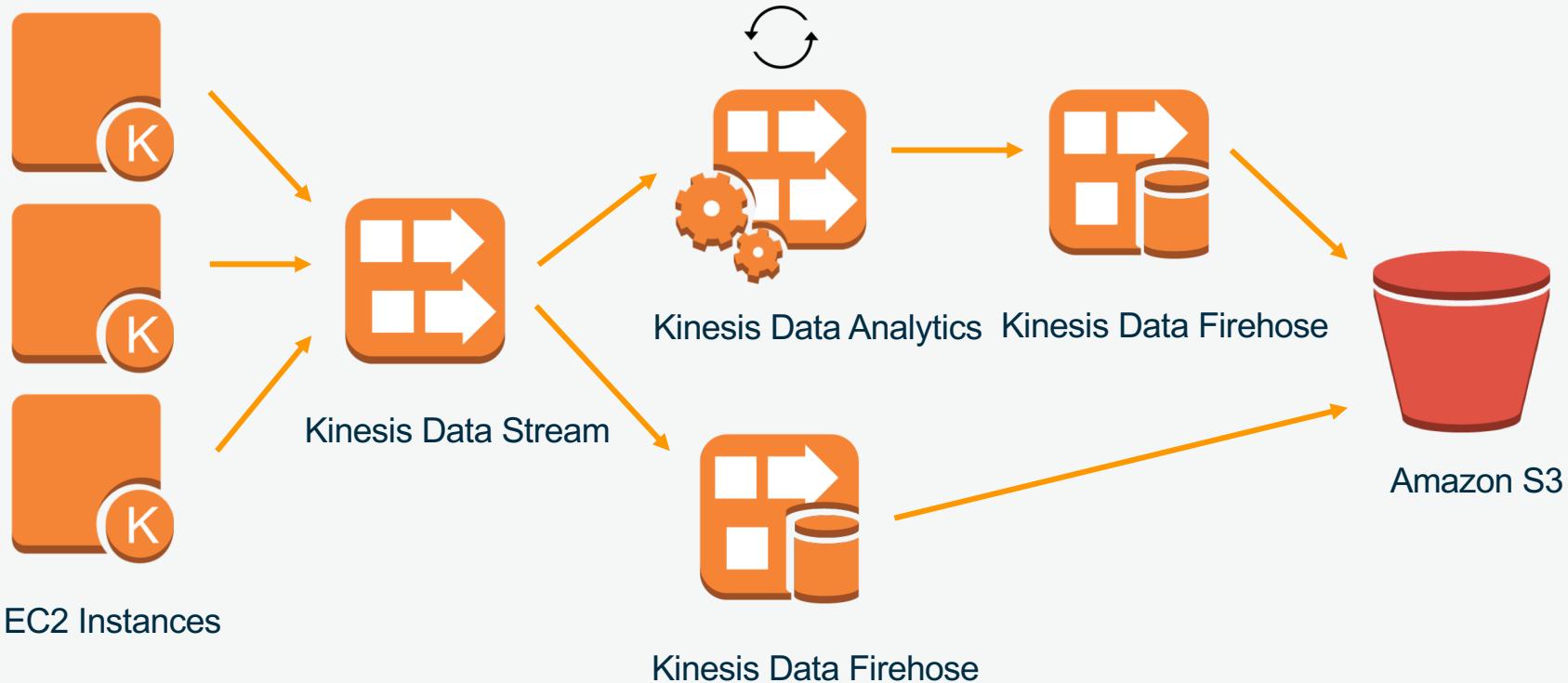
Collecting and Analyzing

- Amazon CloudWatch
- Amazon Kinesis
- Other Options

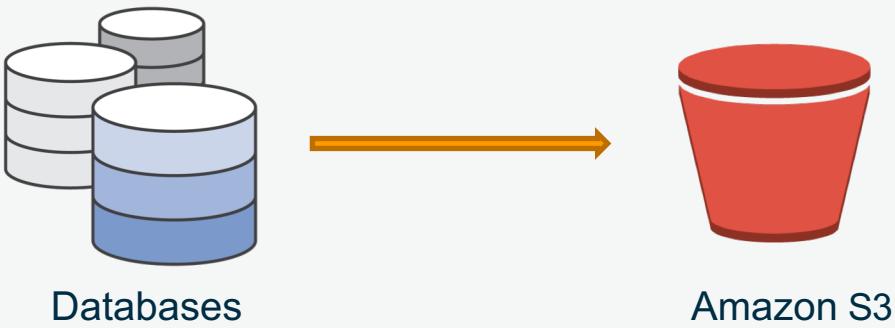
Logs – Kinesis Agent



Logs – Kinesis Agent (with Analytics)



Data Sources - Databases



Database Migration Service

AWS Database Migration Service (AWS DMS) easily and securely migrate **and/or replicate your databases and data warehouses to AWS**



AWS Schema Conversion Tool (AWS SCT) convert your commercial database and data warehouse schemas to open-source engines **or AWS-native services**, such as Amazon Aurora and Redshift

When to use DMS and SCT?

Modernize



Modernize your database tier –

- Commercial to open-source
- Commercial to Amazon Aurora

Modernize your Data Warehouse –

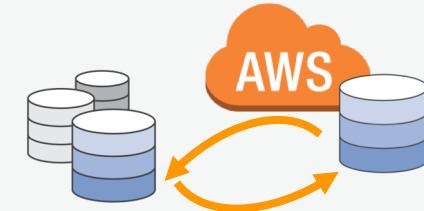
- Commercial to Redshift

Migrate



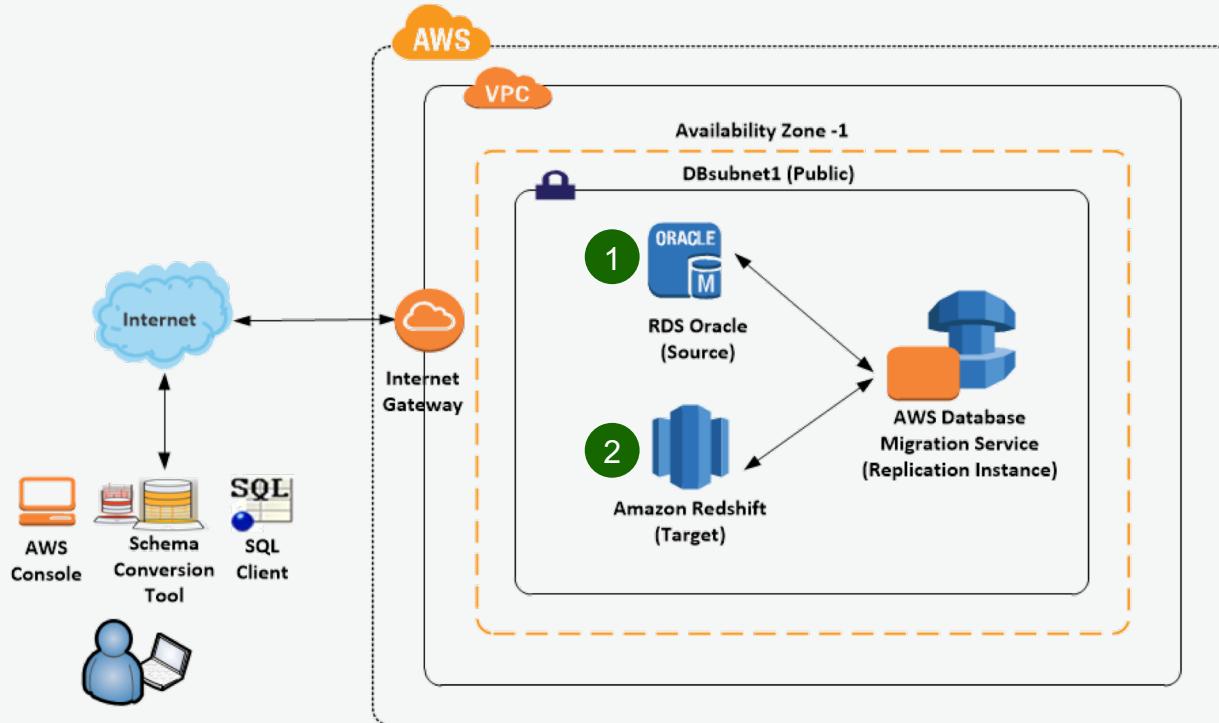
- Migrate business-critical applications
- Migrate from Classic to VPC
- Migrate data warehouse to Redshift
- Upgrade to a minor version

Replicate



- Create cross-regions Read Replicas
- **Run your analytics in the cloud**
- Keep your dev/test and production environment sync

SCT/DMS Demonstration Architecture



DMS Data Lake Use-Case: S3 as a Target

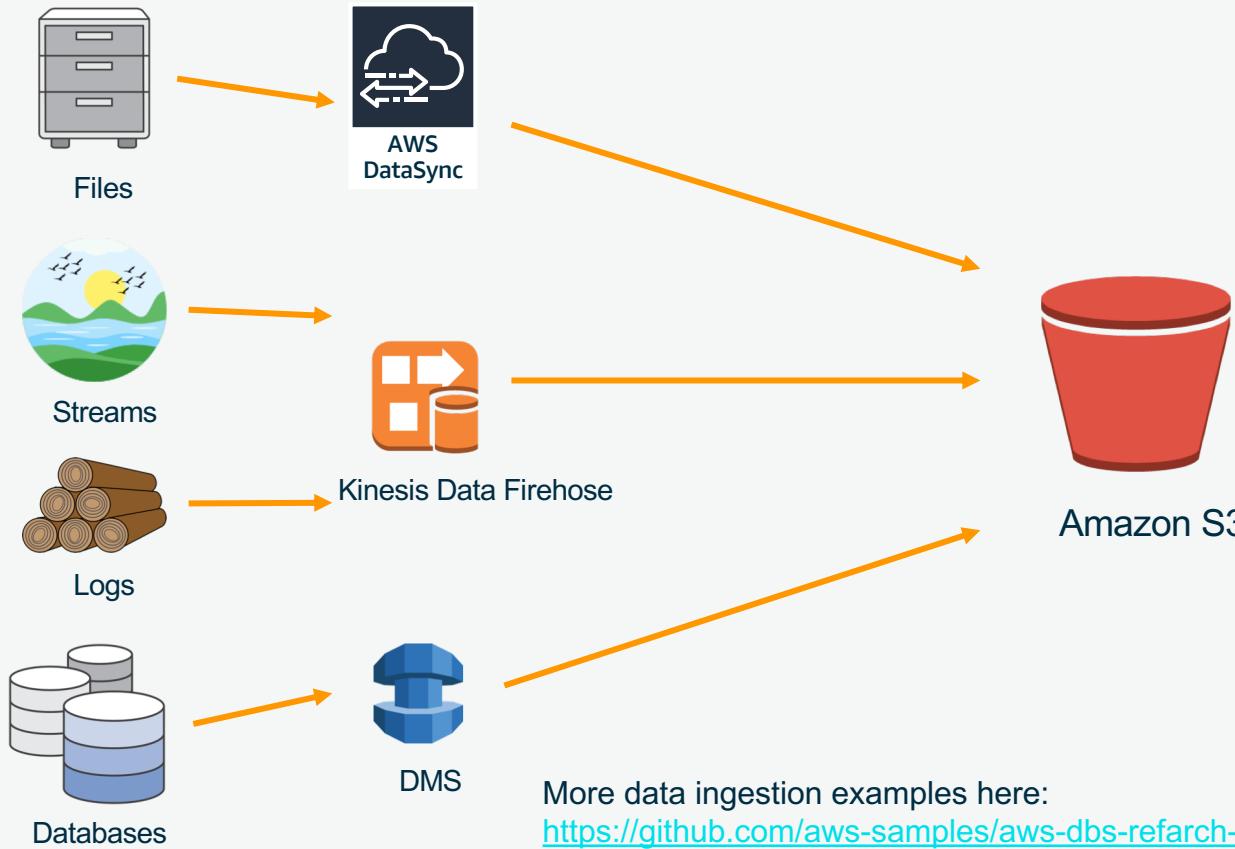
Bulk File

```
s3://mybucket/schemaName/tableName  
s3://mybucket/hr/employee  
  
/schemaName/tableName/LOAD001.csv  
/schemaName/tableName/LOAD002.csv  
/schemaName/tableName/LOAD003.csv  
...  
101,Smith,Bob,4-Jun-14,New York  
102,Smith,Bob,8-Oct-15,Los Angeles  
103,Smith,Bob,13-Mar-17,Dallas  
104,Smith,Bob,13-Mar-17,Dallas
```

Ongoing CDC Files

```
s3://mybucket/schemaName/tableName  
  
<time-stamp>.csv  
<time-stamp>.csv  
<time-stamp>.csv  
...  
  
I,101,Smith,Bob,4-Jun-14,New York  
U,101,Smith,Bob,8-Oct-15,Los Angeles  
U,101,Smith,Bob,13-Mar-17,Dallas  
D,101,Smith,Bob,13-Mar-17,Dallas
```

Summary - Ingestion



More data ingestion examples here:
<https://github.com/aws-samples/aws-dbs-refarch-datalake>

Architecture Examples



Nasdaq operates financial exchanges around the world, and processes large volumes of data.

Challenge:

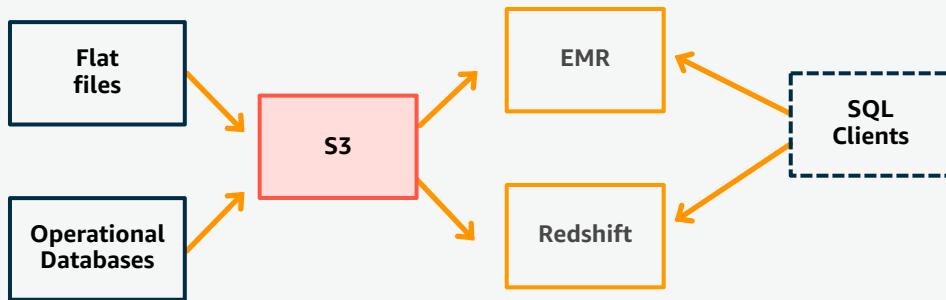
Nasdaq wanted to make their large historical data footprint available to analyze as a single dataset.

Solution:

- Use Amazon Redshift for interactive querying
- Use Amazon S3 as a Data Lake, and Presto on EMR to process historical data



Nasdaq Uses AWS to Build a Data Lake



Data from all 7 exchanges
operated by Nasdaq
(orders, quotes, trade executions)

- Migrate legacy on-premises warehouse to Amazon Redshift
- 4.8B rows inserted per trading day (orders, trades, quotes)
- Ingest data from multiple sources, validates, and stages in S3
- Redshift reads data out of S3 for fast queries
- Presto on EMR and S3 used for analysis of massive historical data set



Sysco is the leader in selling, marketing, and distributing food.

Challenge:

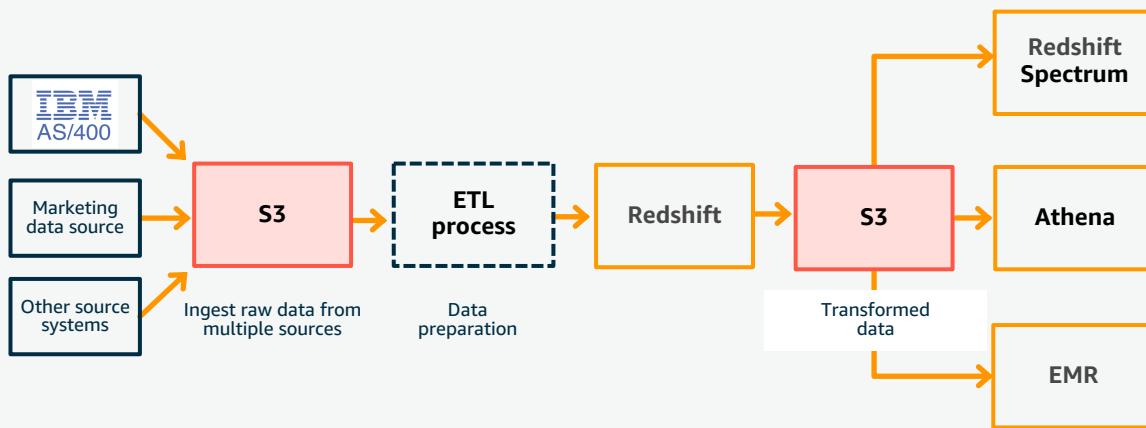
Large volumes of data in multiple systems. Also, high costs from maintaining on-premises EDW deployment.

Solution:

- Migrated their on-premises solution to the cloud with Redshift, S3, EMR, and Athena



Analytics on the Data Lake



- Sysco is the leader in selling, marketing, & distributing food
- Challenge: large volumes of data in multiple systems
- Consolidated data into a single S3 data lake
- Data scientists use EMR notebooks, Athena & Amazon Redshift Spectrum used by business users for reporting



Fox Film Entertainment is one of the largest movie studios in the world.

Challenge:

Processes 100+ of TB of data a day, 25k+ queries per day. They had a legacy data platform that struggled to scale, faced many outages, and had changing requirements from consumers who had many options.

Solution:

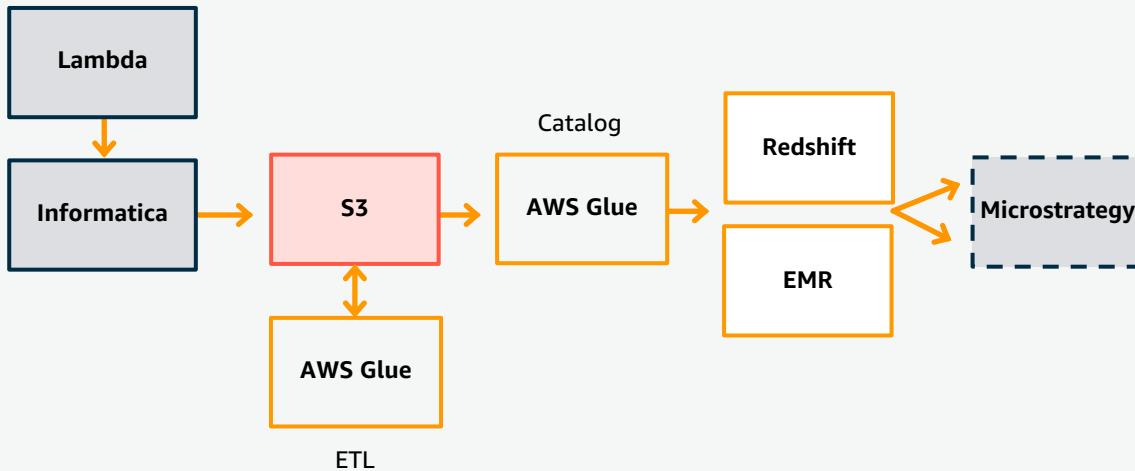
- Land data in S3 as a data lake
- Use Redshift and EMR as analytics engines to process data
- Saved 15–20% in overall costs (on-premises) with 30% of performance gain



Data Lake on AWS



21st Century Fox Uses AWS for Data Lakes & Analytics



- Collect and ingest data with AWS Lambda for serverless scale and Informatica for data ingestion and transformation
- AWS Glue does ETL on data in S3 and provides a data catalog
- Redshift and EMR used as analytics engines
- Microstrategy used as a visualization tool



Amazon.com's vision is to be the earth's most customer-centric company; where people can find anything they want to buy online.

Challenge:

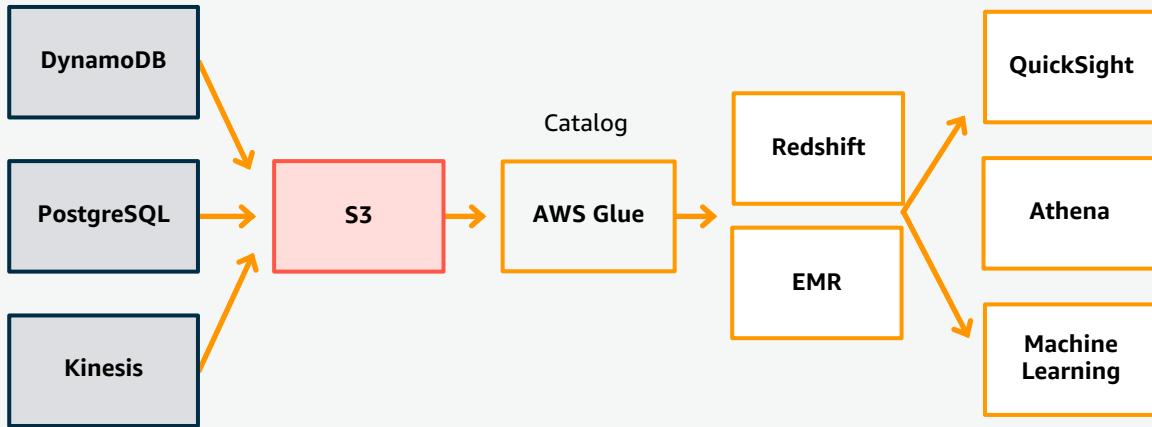
Load 500K+ transactions each day, and serve 300K+ queries/extracts each day from Amazon businesses (Amazon.com, Amazon Prime, Amazon Music, Amazon Alexa, Amazon Video, and Twitch).

Solution:

- Land data in S3 as a data lake
- Use Redshift as preferred SQL based analysis by business users, and EMR for machine learning



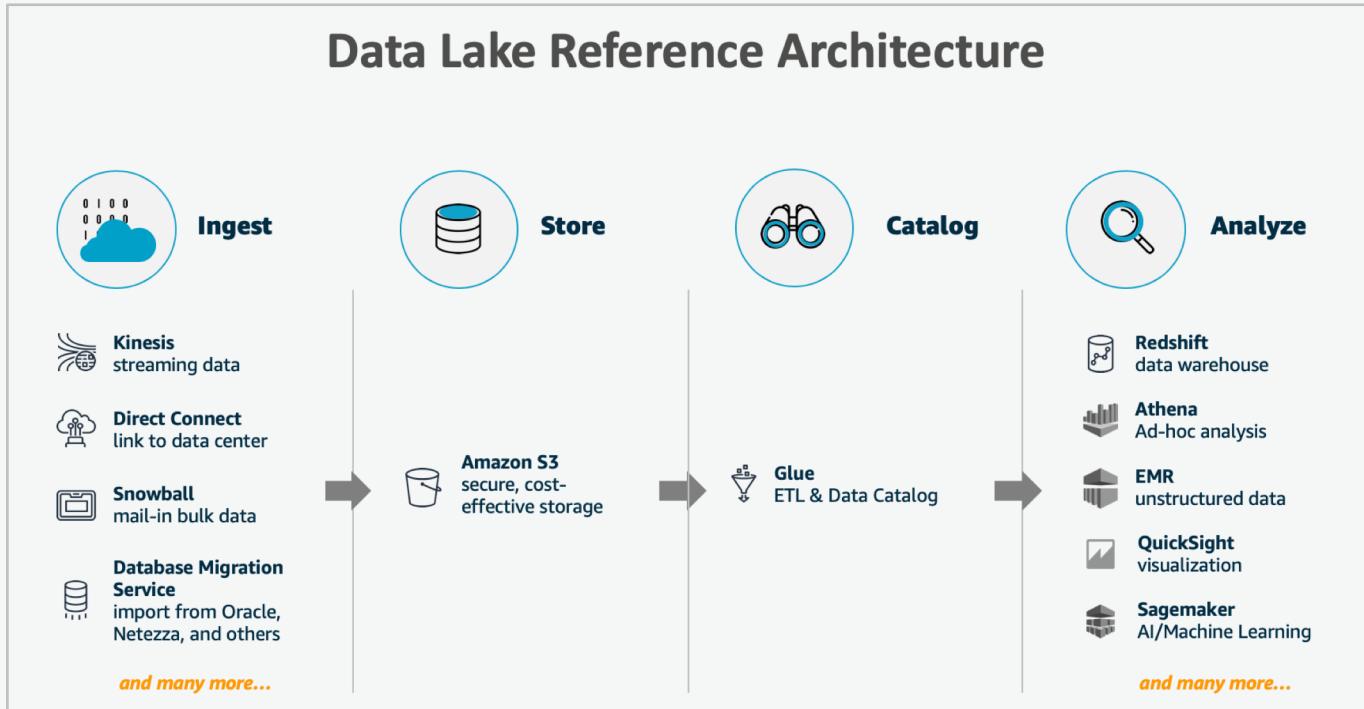
Amazon.com Uses AWS for Data Lakes & Analytics



- DynamoDB capturing all Amazon.com transactions
- Everything from DynamoDB, RDS PostgreSQL and Kinesis fed to a S3 data lake
- Glue used to catalog the data
- Redshift used for all SQL-based queries, and EMR for all machine learning and big data processing
- End-users use QuickSight for visualizations

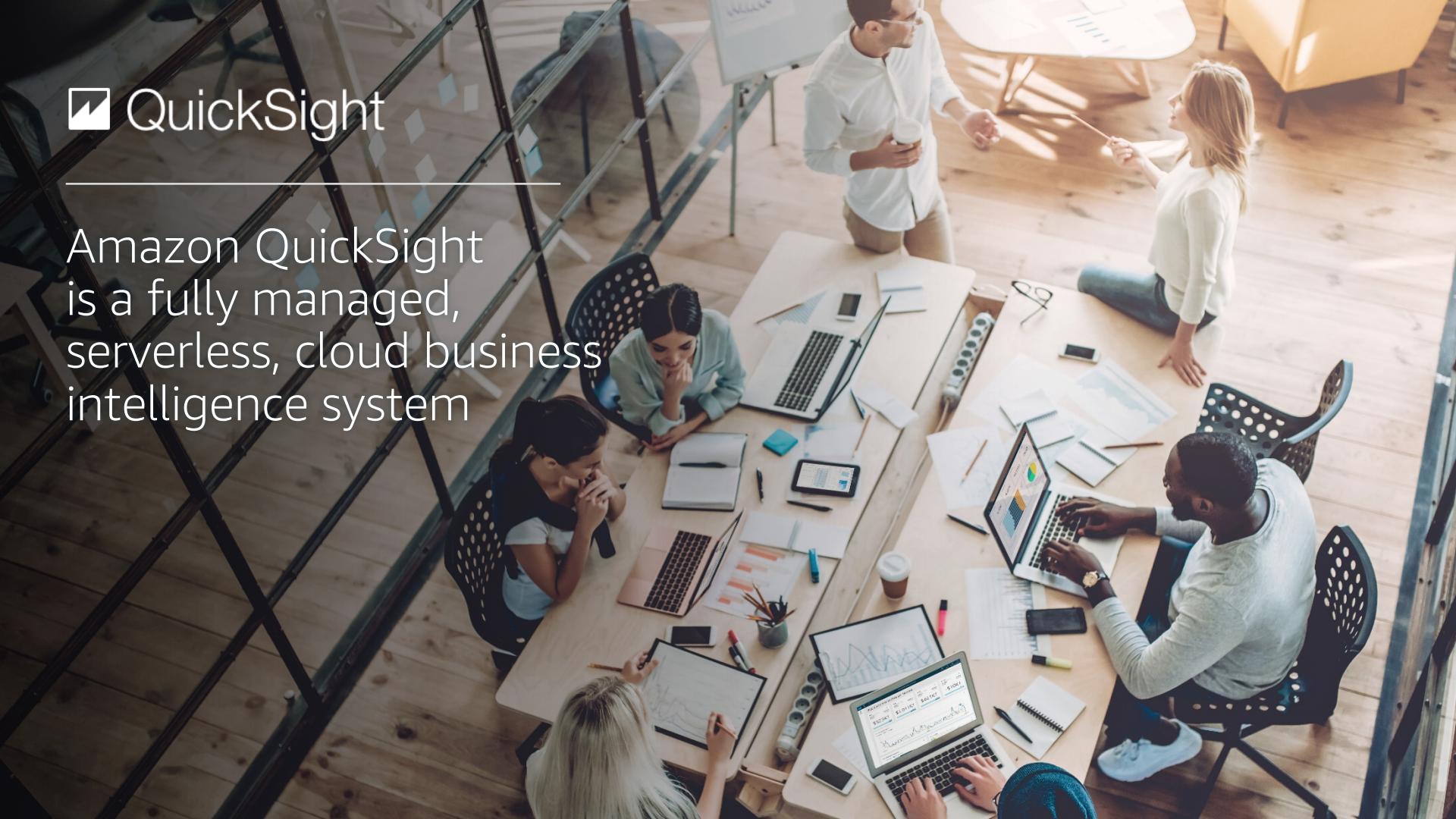
Visualizing Data

Amazon QuickSight





Amazon QuickSight
is a fully managed,
serverless, cloud business
intelligence system



Who Is QuickSight For?



Enterprise BI

Easily share interactive dashboards and insights across your organization at scale.



Developers/ISVs

Embed rich interactive analytics into your Own applications, websites and portals.



Analysts

Allow self-serve power users to easily connect to and explore their data.

Why QuickSight?



No Servers to Manage

QuickSight is a fully managed cloud service. There is no infrastructure to maintain or upgrade and no upfront costs.



Scalable

From 10 users to 10,000, QuickSight seamlessly grows with you with no need for additional servers or infrastructure.



Pay For What You Use

Instead of buying costly licenses for all of your users, QuickSight allows you to share dashboards and reports and only pay when users access them.



Fully integrated

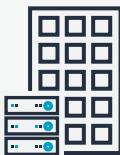
QuickSight integrates with your other AWS services and data sources giving you everything you need to build an end-to-end cloud analytics solution.

Connect to your data, wherever it is

QuickSight allows you to connect to AWS data sources, Private VPC subnets, on-premises and hosted databases and third party business applications.

On-premises

Securely connect to on-premise databases and flat files like Excel and CSV



- Excel
- CSV
- Teradata
- MySQL
- SQL Server
- PostgreSQL

In the cloud

Connect to hosted database, big data formats, and secure VPCs



- Redshift
- RDS
- S3
- Athena
- Aurora
- Teradata
- MySQL
- Presto
- Spark
- SQL Server
- Postgre SQL
- MariaDB
- Snowflake
- IoT Analytics



Applications

Connect directly to third party business applications

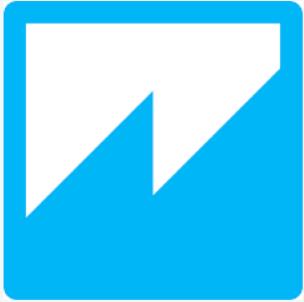


- Salesforce
- Square
- Adobe Analytics
- Jira
- ServiceNow
- Twitter
- Github



serviceNow

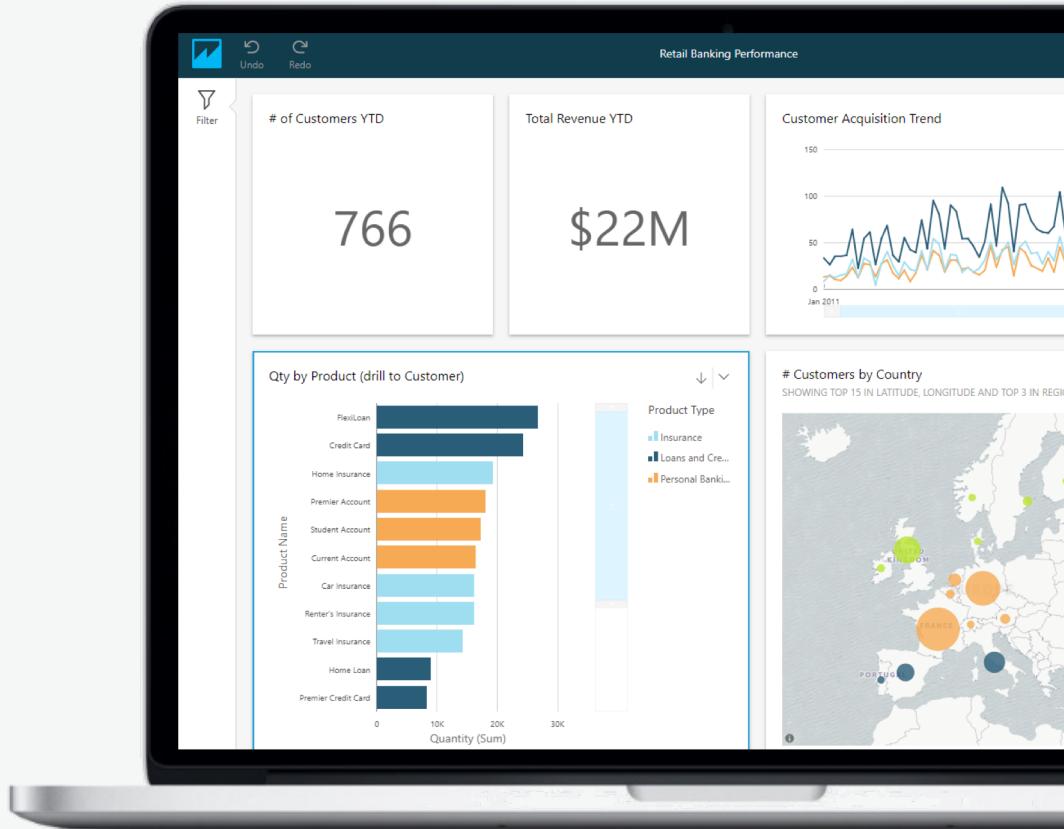




Demonstration

Create beautiful, interactive dashboards.

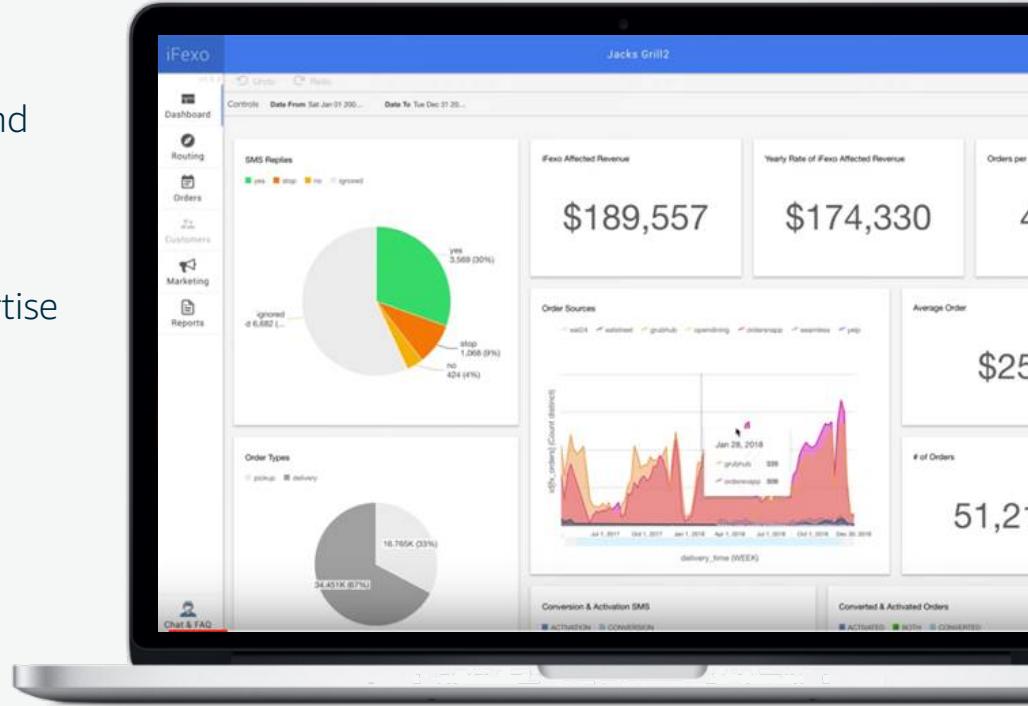
- Add rich interactivity like filters, drill downs, zooming, and more
- Blazing fast navigation
- Accessible on any device
- Data Refresh
- Publish to everyone with a click



Embedding Dashboards In Your Application

QuickSight allows you to seamlessly integrate interactive dashboards and analytics into your own applications

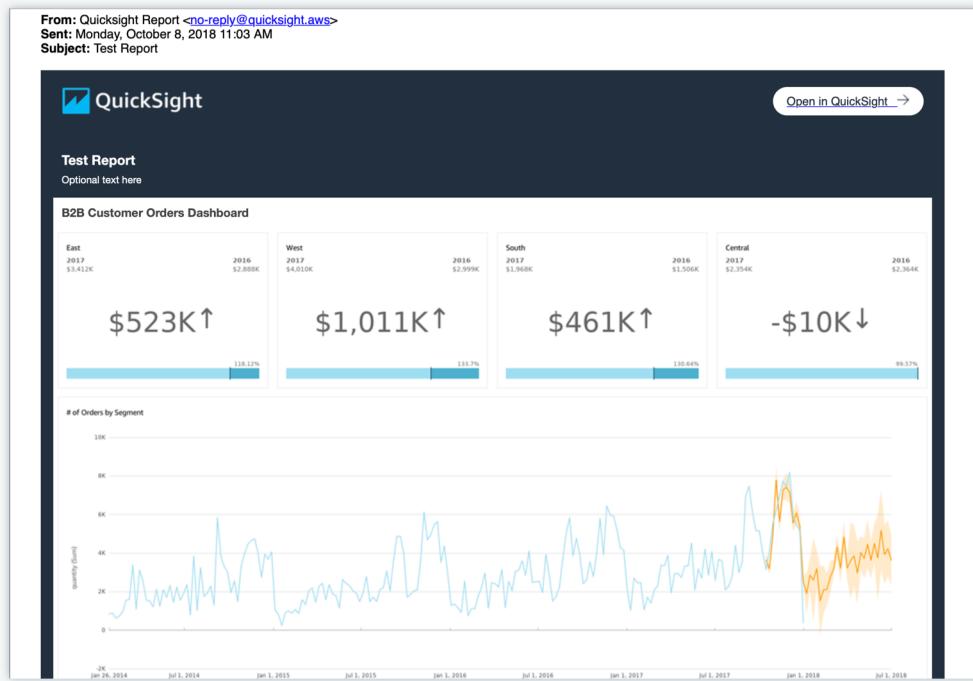
- Enhance your applications with rich analytics and dashboards
- Easy maintenance, no servers to manage
- Fast! No Custom development or domain expertise needed
 - Leverage new features as we add them
 - Utilizes Pay-per-Session Pricing.



Insights Delivered to Your Inbox

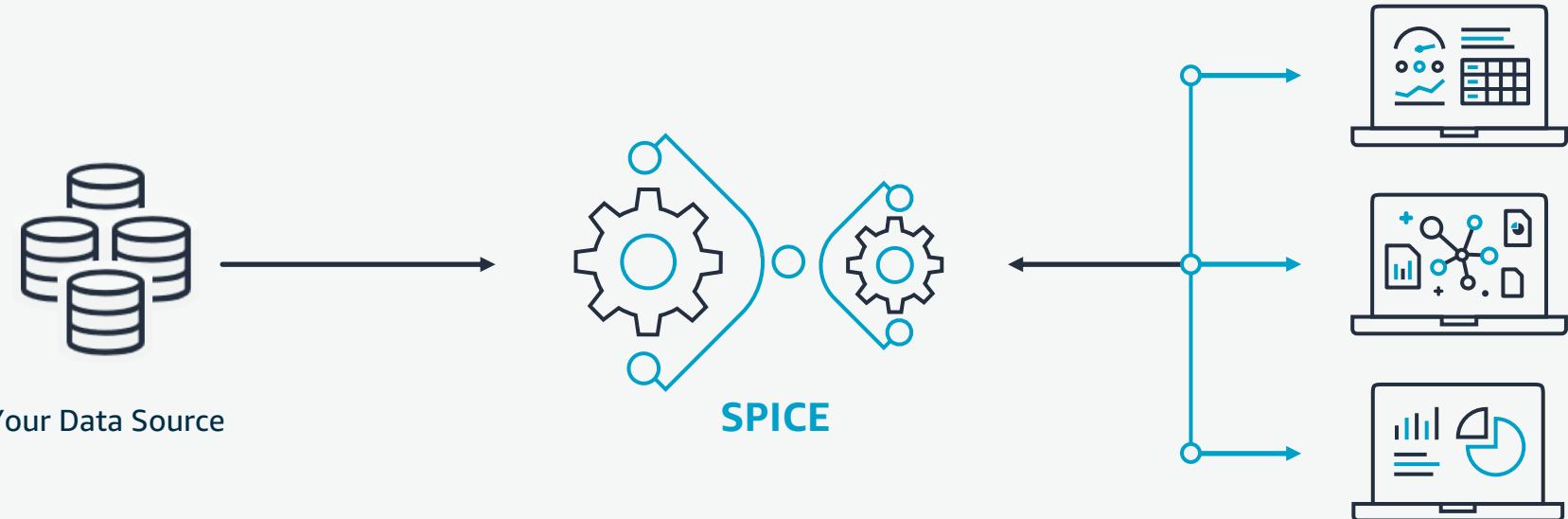
QuickSight lets you send report snapshots directly to your users inbox.

- Schedule email reports on a daily, weekly, or monthly basis
- Sent to individual users or groups
- Users can opt out of any report so they can focus on what's important.
- Uses Pay-per-Session Pricing



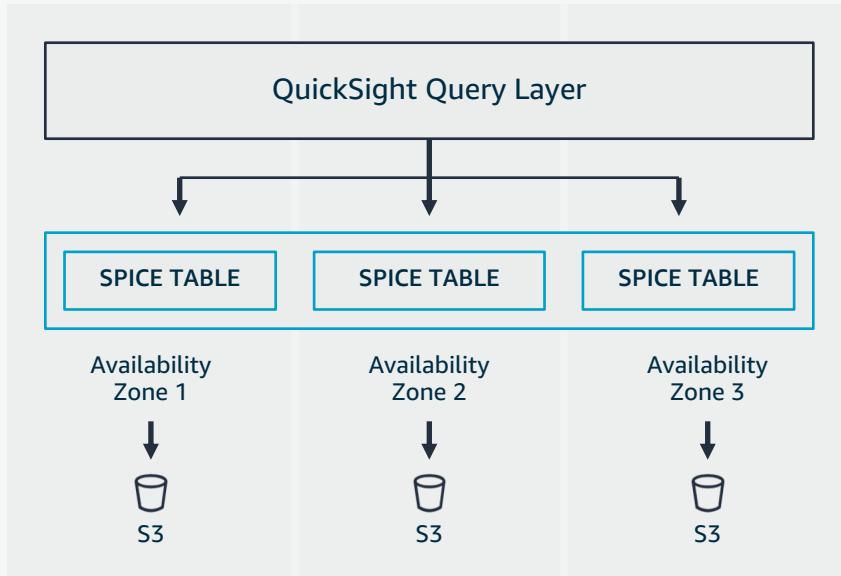
SPICE

QuickSight is powered by SPICE, a super-fast in-memory calculation engine that delivers performance and scale, regardless of how many users are active.



More About SPICE

Storing your data in SPICE protects your underlying data sources from end user activity, guaranteeing performance and saving you money.



Up to 10X faster (millisecond latency)

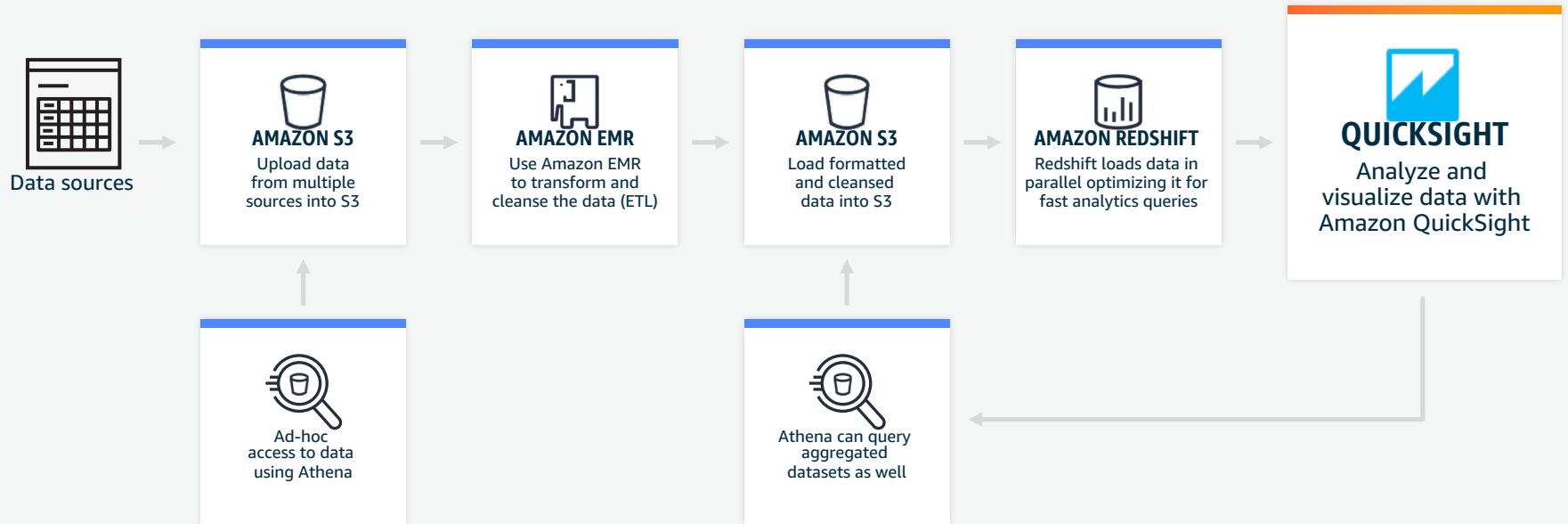
Fault-tolerant, self-healing

Support for high concurrency

Backed up in S3 (Write Ahead Log)

Instant failover with zero impact

ETL, Data Warehousing and Analytics with AWS



Use Case: NFL Next Gen Stats

NFL NEXT GEN STATS

HOME CHARTS STATS PLAYLIST GENERATOR FAQ

Player Search Logout

Game Center Passing Stats Rushing Stats Receiving Stats Defense Player Defense Team Nearest Defender Speed Distance Play

PASSING STATS

Glossary

Controls

Player [ALL] Season 2017 Season Type REG Week [ALL]

Team [ALL] Opponent [ALL] Quarter [ALL] Down [ALL]

Individual Passing Stats (Season)																						
Player	N...	Team	Season	TTT	Cmp	Att	Cmp %	Pass Yds	Pass TD	INT	Rating	Y/A	AV/A	AD/A	TW %	Sep	AVTS	LCAD	QBP	QBP...	ScrYds/Att	Exp
Jared Goff																						
Carson Wentz																						
Deshawn Watson																						
Jacoby Brissett																						
C.J. Beathard																						
Joe Flacco																						
Blaine Gabbert																						
Tom Brady																						
Brian Hoyer																						
Matthew Stafford																						



Customers



Customer name:

Auto Trader Group plc.

Company description:

Automotive marketplace

Data source:

S3/Athena

Use case:

50+ users and growing

Previous tools:

Spreadsheets and static graphs

Auth:

SAML-based SSO

Use case:

- Provide business dashboards to multiple teams from S3 data lake
- Equip analysts with easy, self-service dashboard building capability. Eliminate need to manually curate, process and build spreadsheets, saving 10% of an analyst's monthly efforts.



"Timely access to relevant data is the key to success in our business. Amazon QuickSight has enabled our analysts to move from static graphs to fast, interactive dashboards automatically updated with the latest data. QuickSight's native integration with Amazon Athena has allowed faster time to analysis and saved us effort involved in additional data curation. With the new QuickSight readers, we can extend interactive dashboards to the entire team, and Pay-per-Session pricing assures us we only pay when we use the product. The serverless nature of QuickSight aligns with our vision for the data platform, with no infrastructure management needed to scale deployments across the company! We look forward to enabling more use cases in QuickSight."



Jim Stamp, Head of Product – Data Engineering



As a native cloud service, QuickSight combines the speed, scalability, and security that our customers have come to depend on with the value and cost effectiveness you expect from AWS



Native AWS
service



No server licensing
or maintenance costs



Pay-as-you-go



Scalable,
fast, easy



No deployment
time

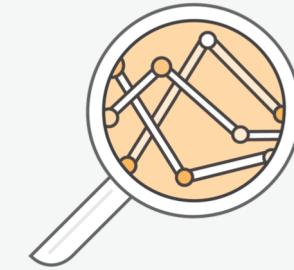
Try it free Today @
Quicksight.AWS

Wrap-up

Feedback survey: <http://bit.ly/2mjYBcO>

Summary

AWS helps unlock Data in three ways



Future-proofed for new data types

Ingest new data for differentiated experiences: image/video, social media, IoT sensors, and more

Single source of truth (aka “data lake”)

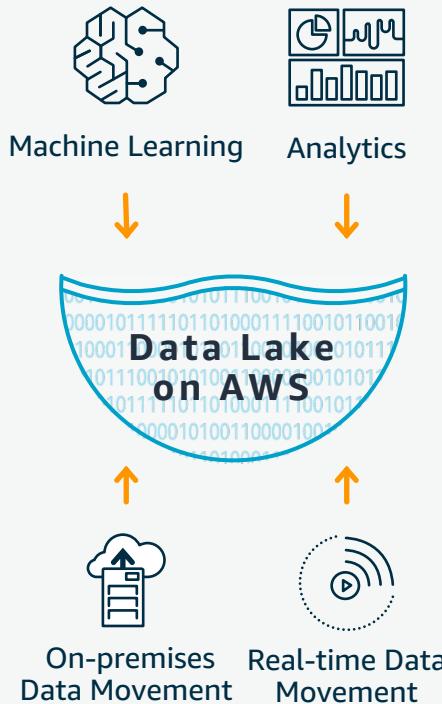
Secure and scalable backbone for all incoming data and all analytics – no more data siloes

The right analytics tool for every job

New analytics possibilities – machine learning, natural language processing, Hadoop-as-a-service, and more

Summary

Benefits of Data Lakes from AWS

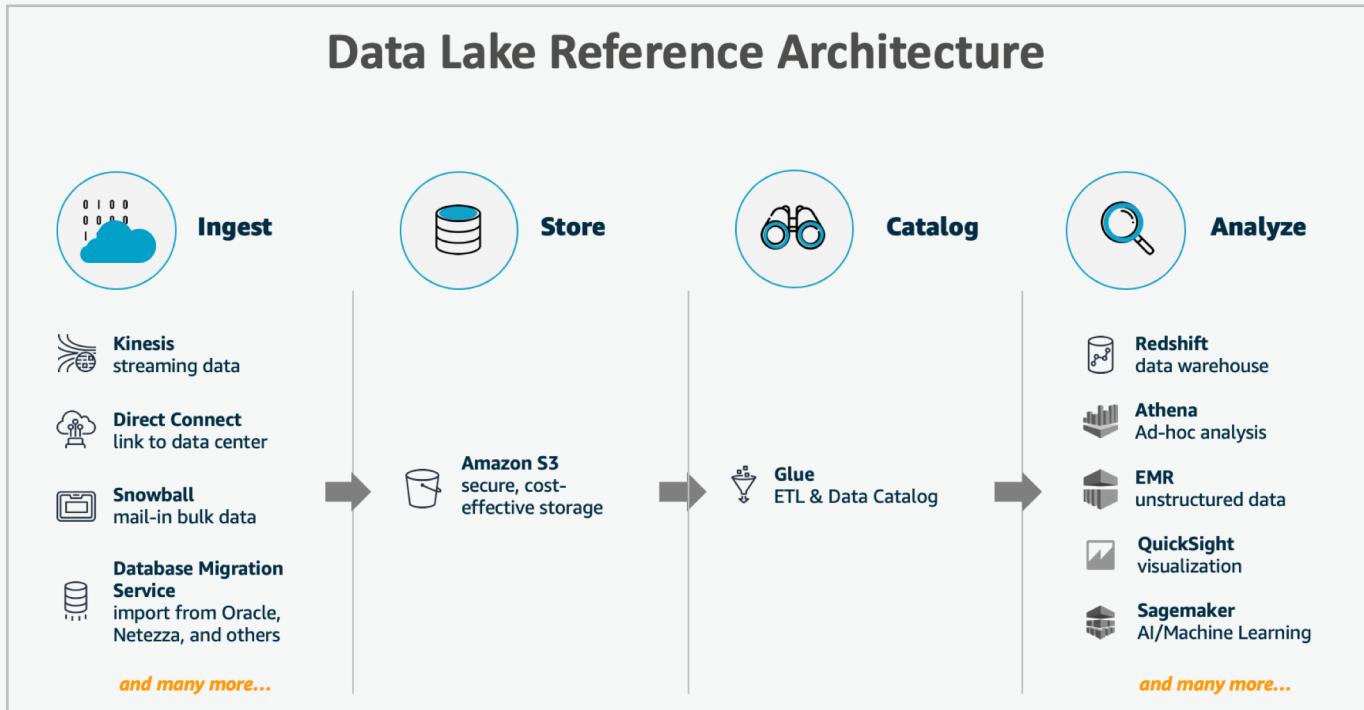


- Open and comprehensive
- Secure
- Scalable and durable
- Lowest cost

More Data Lakes & Analytics on AWS than Anywhere Else



Thank you!



Thank you!

Survey: <http://bit.ly/2mjYBcO>

