



Intro to AWS Data Lakes and Analytics

Immersion Day

David Bayard, Solutions Architect
John Hwang, Solutions Architect

October 2019

If you wish to do today's hands-on labs, please
send an email to johnhwan@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?
- Building a Cloud Data Lake on AWS
- Overview of AWS Lake Formation

Lab: AWS Lake Formation Workshop

11:00am (75 minutes) – Building a Cloud Data Lake on AWS, continued

Lab: Building a Data Lake with S3 & AWS Glue & Athena

12:15 pm (45 minutes) – Lunch

Agenda – Part 2

1:00 pm (60 minutes) - Cloud Data Warehouse

- What is a Cloud Data Warehouse and how does it fit with a Cloud Data Lake?

Lab: Extending the lake with Amazon Redshift

2:00 pm (45 minutes)- Populating the Data Lake/Data Warehouse

- Discussion of different complimentary ingest capabilities
 - 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)

Demonstration of DMS & SCT

3:00 pm (25 minutes) - Visualizing Data

- Quicksight Overview

Demonstration of Quicksight

3:25 pm (5 minutes) - Wrap-up

Lab Preparation

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

Introductions and Objectives

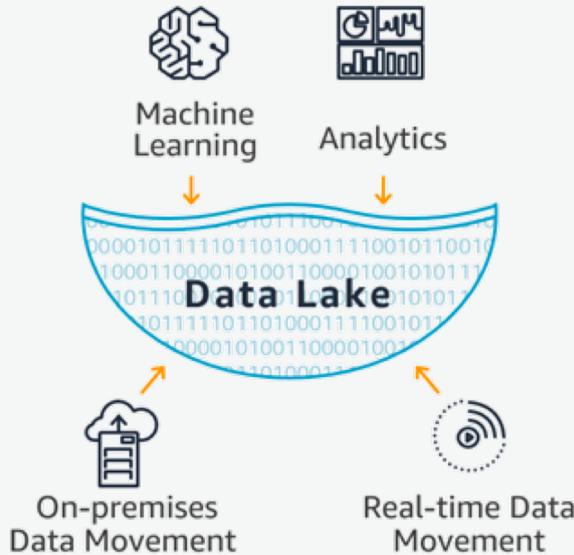
Q1: What is your level of experience with AWS?

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

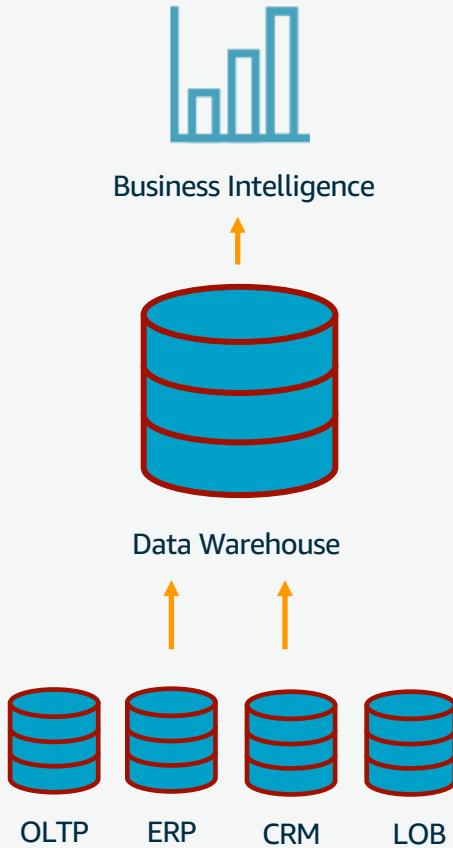
Cloud Data Lakes

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

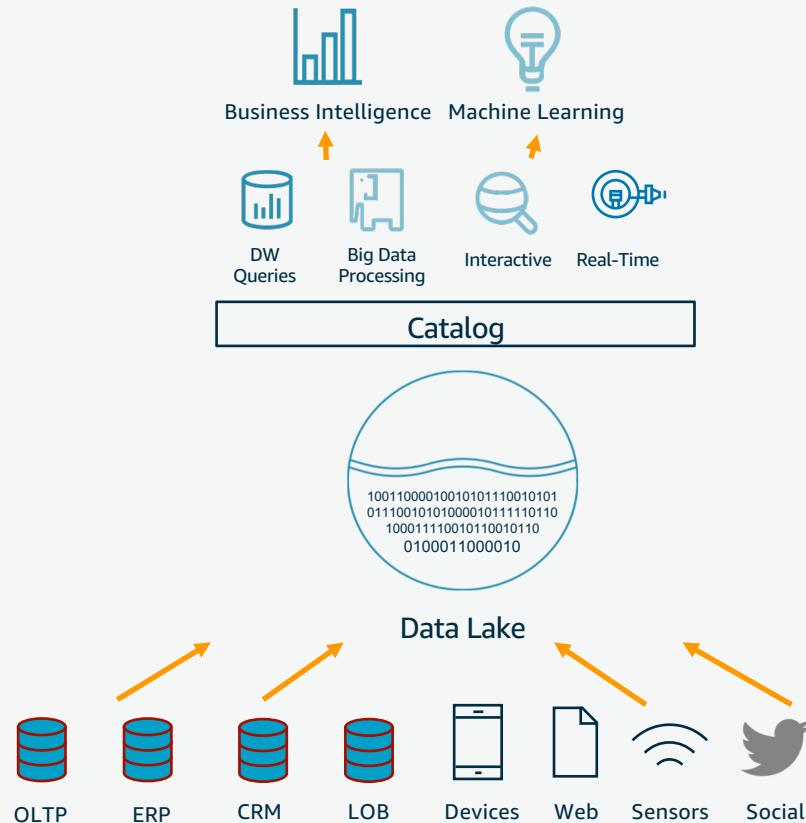
TBs Scale

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

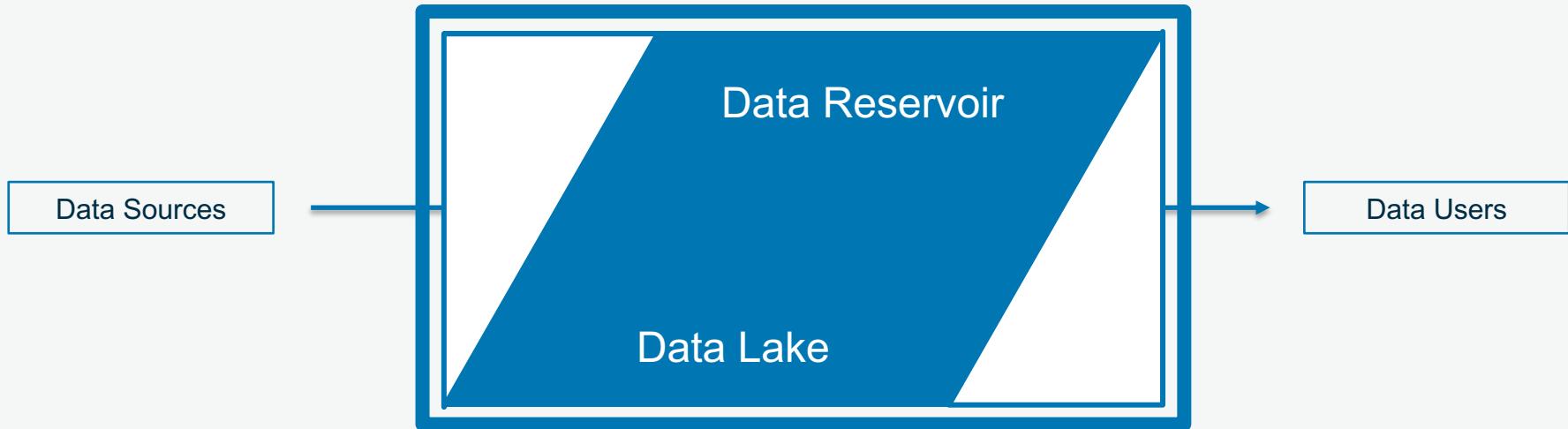
Schema on Read

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.

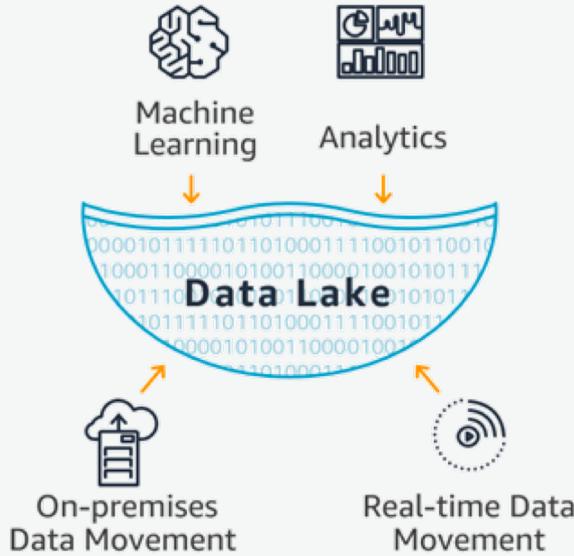
Data Warehouses are optimized for ease of retrieval of more critical and useful data.

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

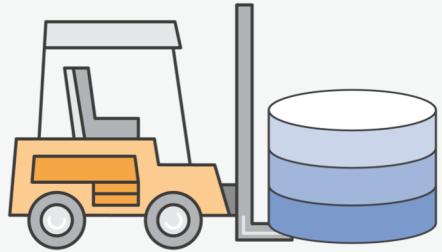


“Why is the data distributed in many locations? Where is the single source of truth ?”

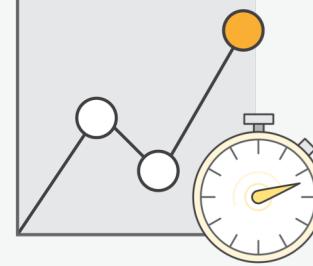


Store and analyze all of your data, from all of your sources, in one centralized location.

Benefits of a Data Lake – Quick Ingest

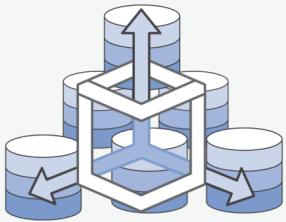


“How can I collect data quickly from various sources and store it efficiently?”

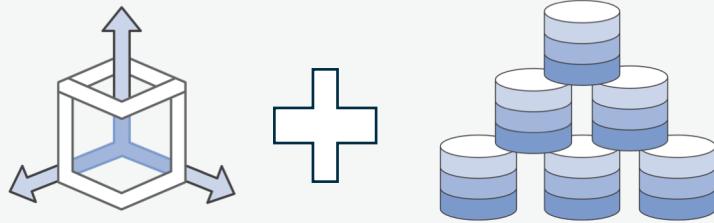


Quickly ingest data without needing to force it into a pre-defined schema.

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

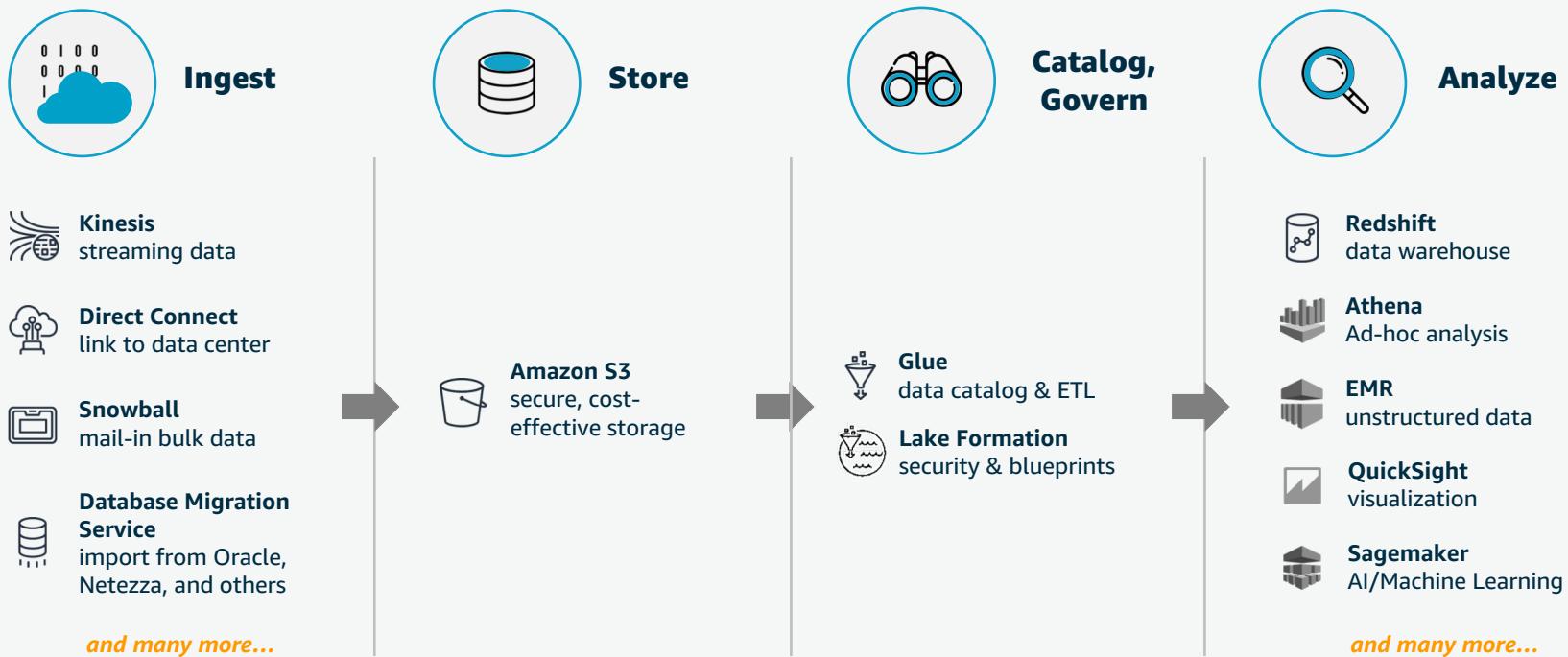
More data lakes & analytics on AWS than anywhere else



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



Data Lake Reference Architecture



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. Six rows are listed, all named with the suffix ".json" and belong to the "legislators" database:

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;
```

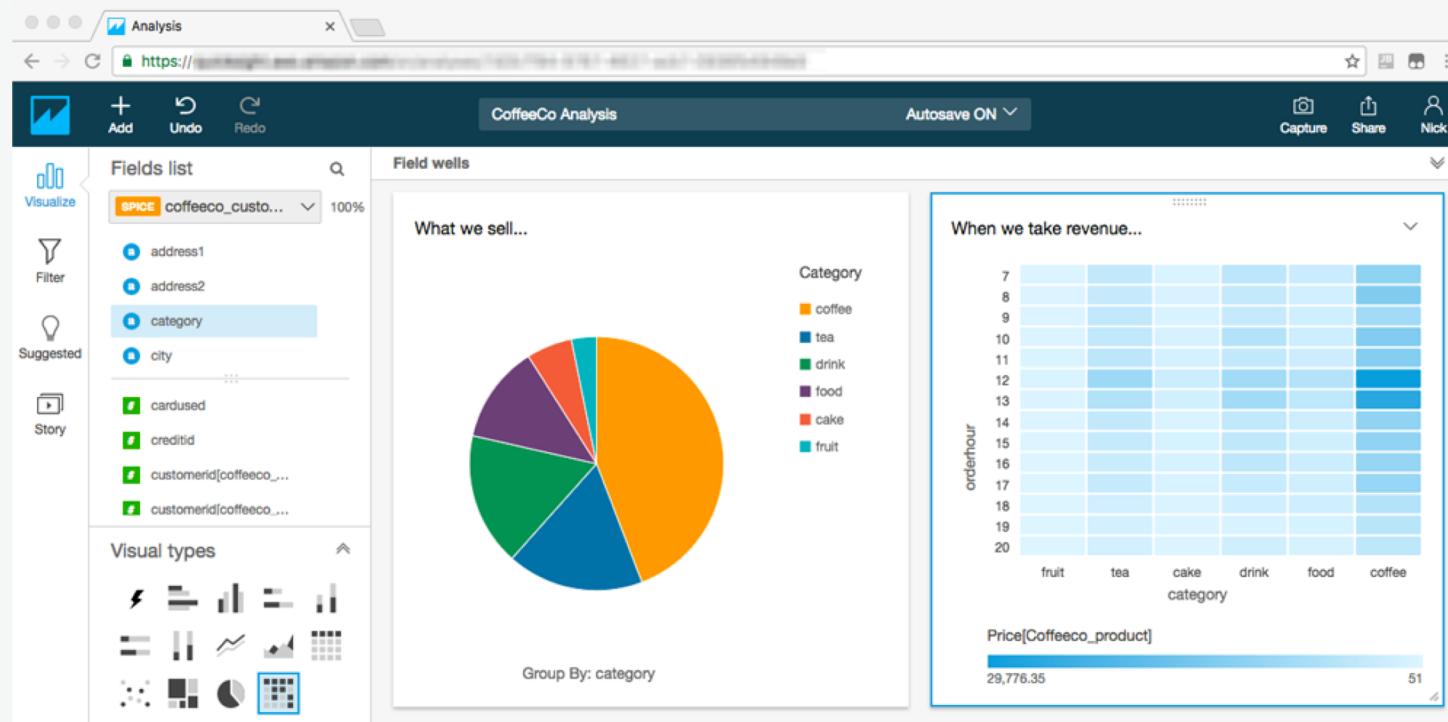
Use Ctrl + Enter to run query, Ctrl + Space to autocomplete

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

- Hadoop 2.6.0 Hive 1.0.0 Mahout 0.11.0
- Zeppelin-Sandbox 0.5.5 Hue 3.7.1 Spark 1.5.2 Oozie-Sandbox 4.2.0
- Ganglia 3.6.0 Presto-Sandbox 0.125 Oozie 4.2.0
- 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))
}
```

Zepplin

Notebook - Interpreter

Welcome to Zeppelin.

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

Task 1 seconds.

Load Data Into Table

```
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset
import org.apache.spark.sql.RDD
import org.apache.spark.sql.DataFrame
defined class org.apache.spark.sql.DataFrame = [age: int, job: string, marital: string, education: string, balance: int]
```

Task 5 seconds. (submitted)

FINISHED D II @

Ysql

select age, count(1) value

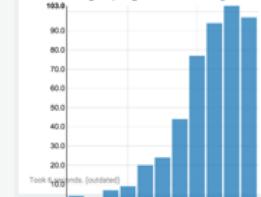
from bank

where age < 30

group by age

order by age

settings +



FINISHED D II @

Ysql

select age, count(1) value

from bank

where marital != \${maxAge-30}

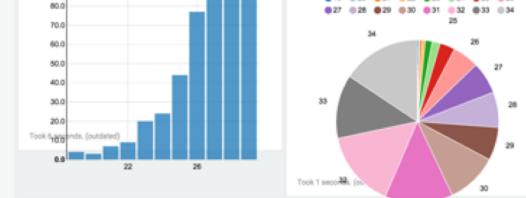
group by age

order by age

maxAge

30

settings +



FINISHED D II @

Ysql

select age, count(1) value

from bank

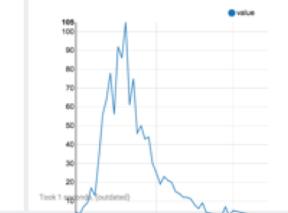
where marital = \${marital=single, single/divorced/married}

group by age

marital

single

settings +



Train ML Models with Amazon SageMaker

The screenshot shows the AWS Management Console interface for Amazon SageMaker. The left sidebar has a dark theme with the following navigation items:

- Dashboard
- Notebook instances
- Jobs** (highlighted in orange)
- Resources
- Models
- Endpoint configuration
- Endpoints

The main content area is titled "Input data configuration". It displays instructions: "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](#)".

A single channel named "train" is listed. The configuration for "train" includes:

- Channel name:** train
- 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

At the bottom of the configuration panel are "Edit" and "Remove" buttons, and a "Done" button.

At the very bottom of the page, there are "Feedback" and "English (US)" buttons, and a copyright notice: "© 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.



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:

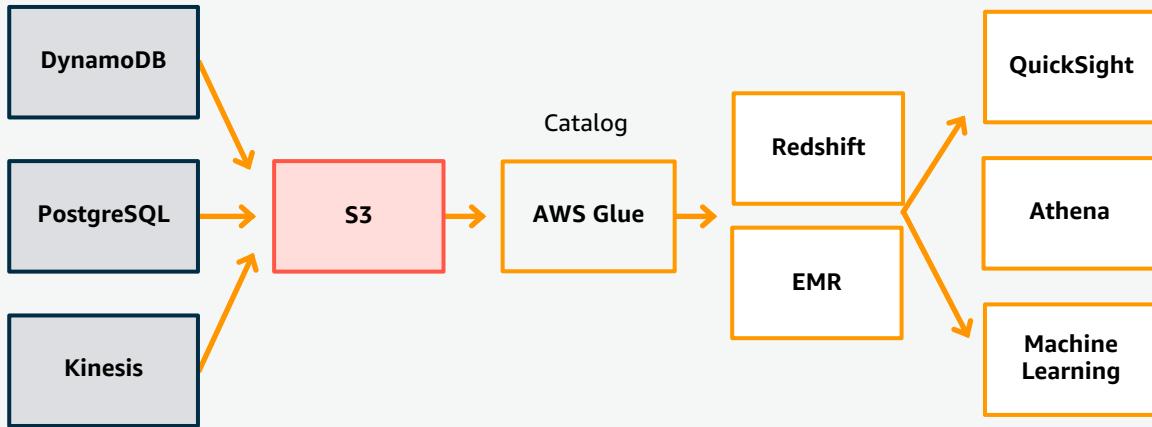
Load 500K+ transactions each day, and serve 300K+ queries/extracts each day from Amazon businesses (Amazon.com, Amazon Prime, Amazon Music, Amazon Alexa, Amazon Video, and Twitch).

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:

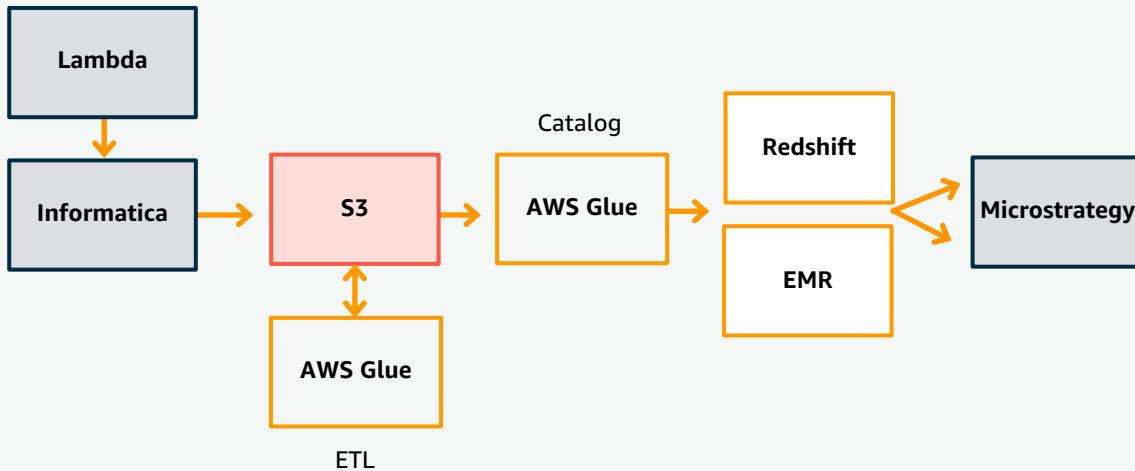
- Land data in S3 as a data lake
- Use Redshift and EMR as analytics engines to process data
- Saved 15–20% in overall costs (on-premises) with 30% of performance gain



Data Lake on AWS

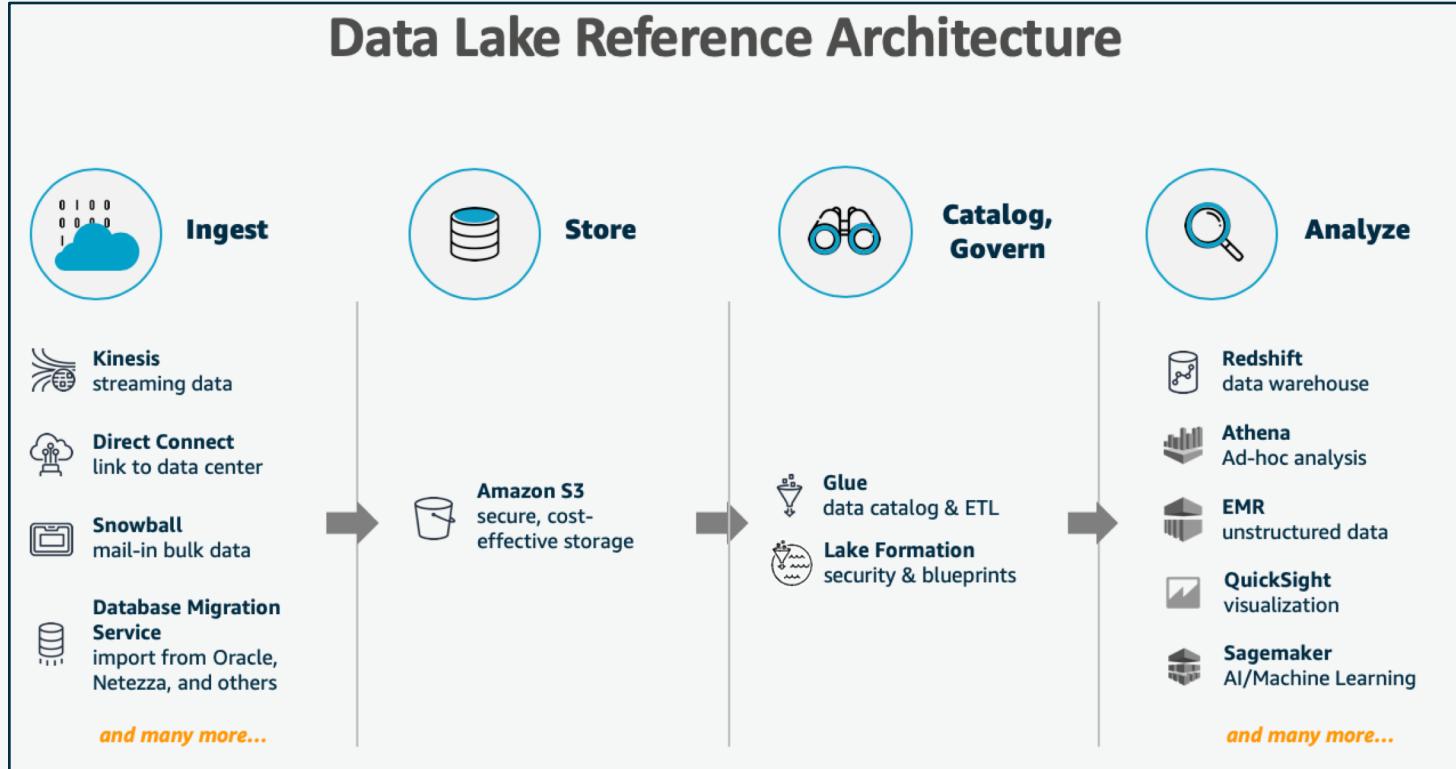


21st Century Fox Uses AWS for Data Lakes & Analytics



- 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

AWS Lake Formation



Lake Formation simplifies
building secure data lakes

Built on Amazon S3 a robust data lake infrastructure



Comprehensive set of **integrated tools** enable every user equally

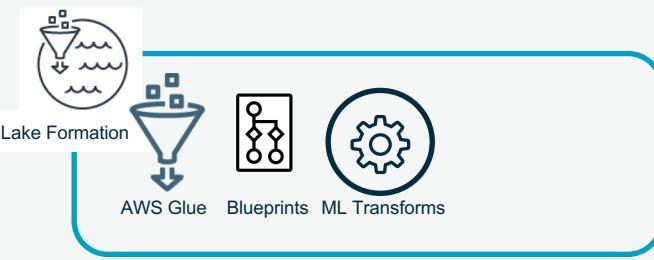


Cost effective, durable storage with global replication capabilities

Automates manual, repetitive, low value tasks



Comprehensive set of **integrated tools** enable every user equally



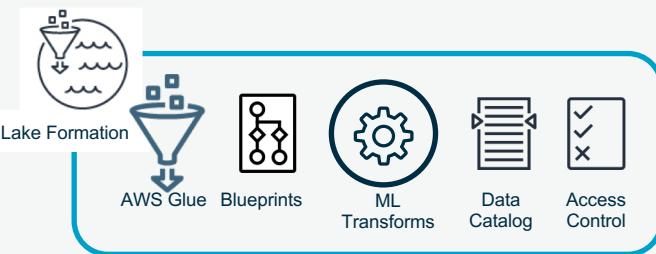
Simplified **ingest & cleaning** enables data engineers to build faster

Cost effective, durable storage with global replication capabilities

Provides a central locus of control



Comprehensive set of **integrated tools** enable every user equally



Centralized management of **fine grained permissions** empower security officers



Simplified **ingest & cleaning** enables data engineers to build faster

Cost effective, durable storage with global replication capabilities

AWS Lake Formation Pricing

No additional charges – Only pay for the underlying services used.

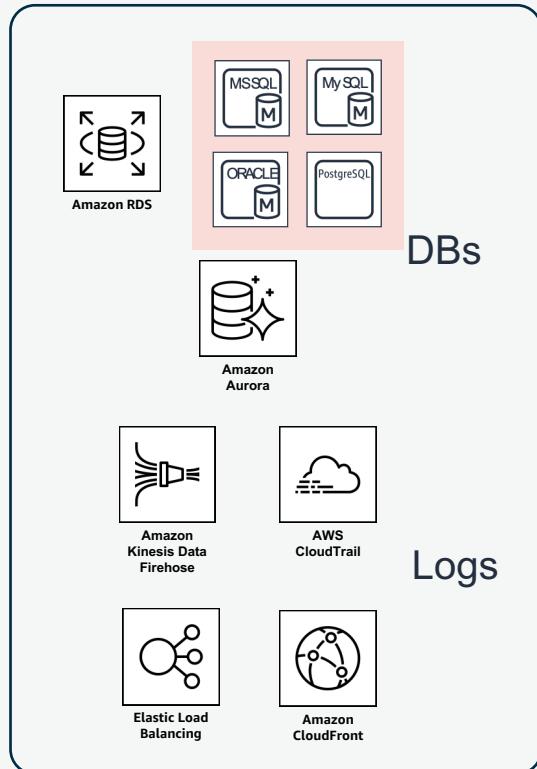
Lab Time

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

Please do Part 1.

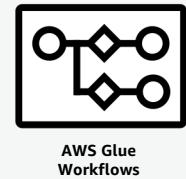
We'll do a few more Lake Formation slides while your blueprint is executing...

Easily load data into your data lake w/ blueprints



Prebuilt templates to serve common ingestion use cases

Automatically build **AWS Glue workflows**



AWS Glue **jobs** and **crawlers** discover, transform and structure data

Automatically populate the **Data Catalog**

Load data **incrementally** or in **full**

With blueprints

You

Point to data **source**

Specify data lake **location**

Specify data load **frequency**

Blueprints

Discover source table(s) schema

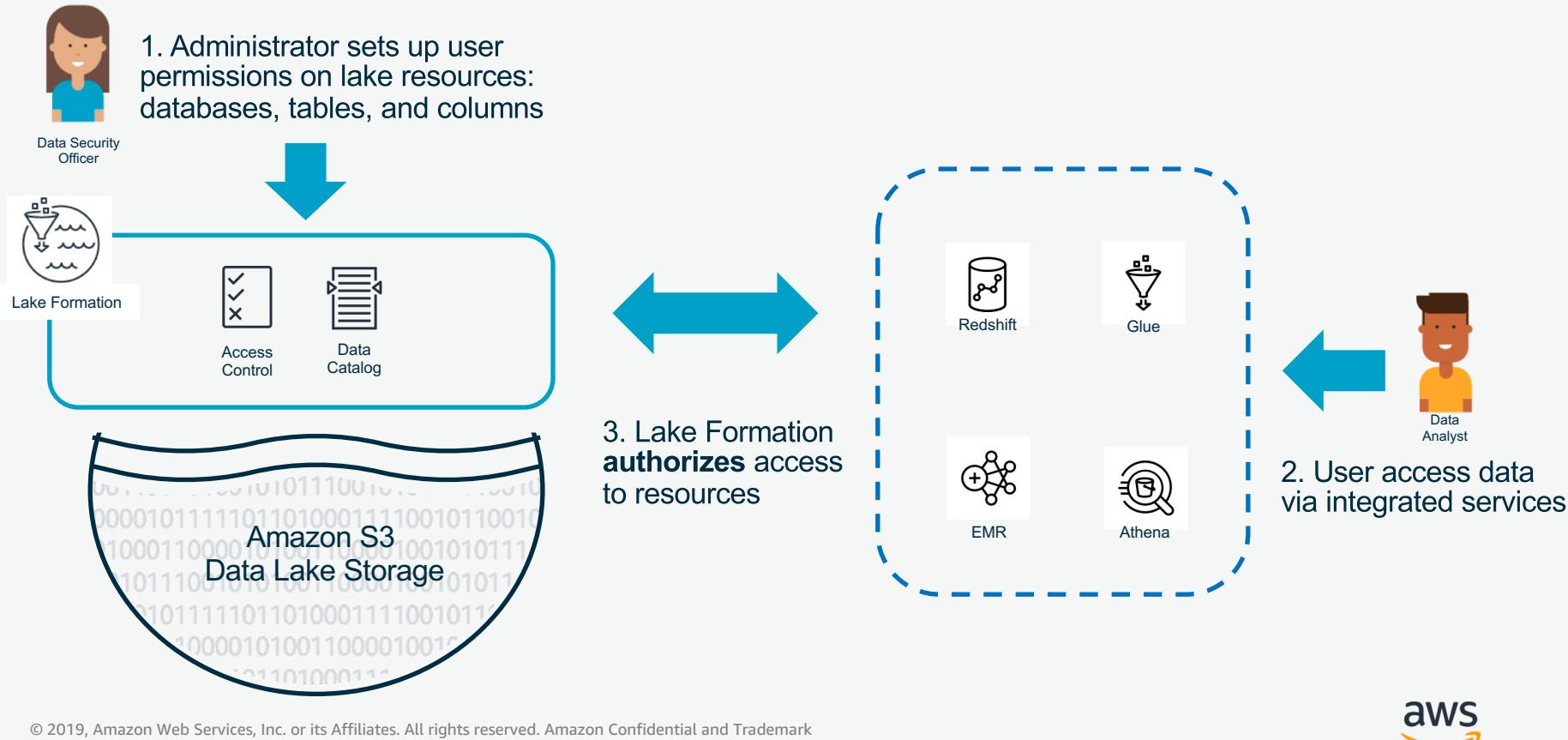
Convert to target data format

Partition data automatically

Track data that was already processed

Customize to your needs

Centralized permissions



Comprehensive portfolio of integrated tools

Compliant services honor
Lake Formation permissions



They guarantee that users
only see **tables & columns**
they have access to

All access is logged and
auditable

A screenshot of the AWS Athena Query Editor interface. The top navigation bar shows tabs for 'Athena' and 'Query Editor', along with links for 'Saved Queries', 'History', 'AWS Glue Data Catalog', 'Workgroup : primary', and account information ('datalake_user @ 7857-8929-2...'). The main area is divided into sections: 'Catalog' (set to 'Lake formation'), 'Database' (set to 'amazoncloudtrail'), and 'Tables (1)' (listing 'amazoncloudtrail_cloudtrail' as a 'Partitioned' table). Below these are sections for 'Views (0)' and 'Create view'. A central query editor window titled 'New query 1' contains the SQL command: 'SELECT * FROM "amazoncloudtrail"."amazoncloudtrail_cloudtrail" limit 10;'. Below the query are buttons for 'Run query', 'Save as', and 'Create'. The results section at the bottom displays a table with 10 rows of log data, each containing 'eventversion' and 'useridentity' fields. The AWS logo is visible in the bottom right corner of the slide.

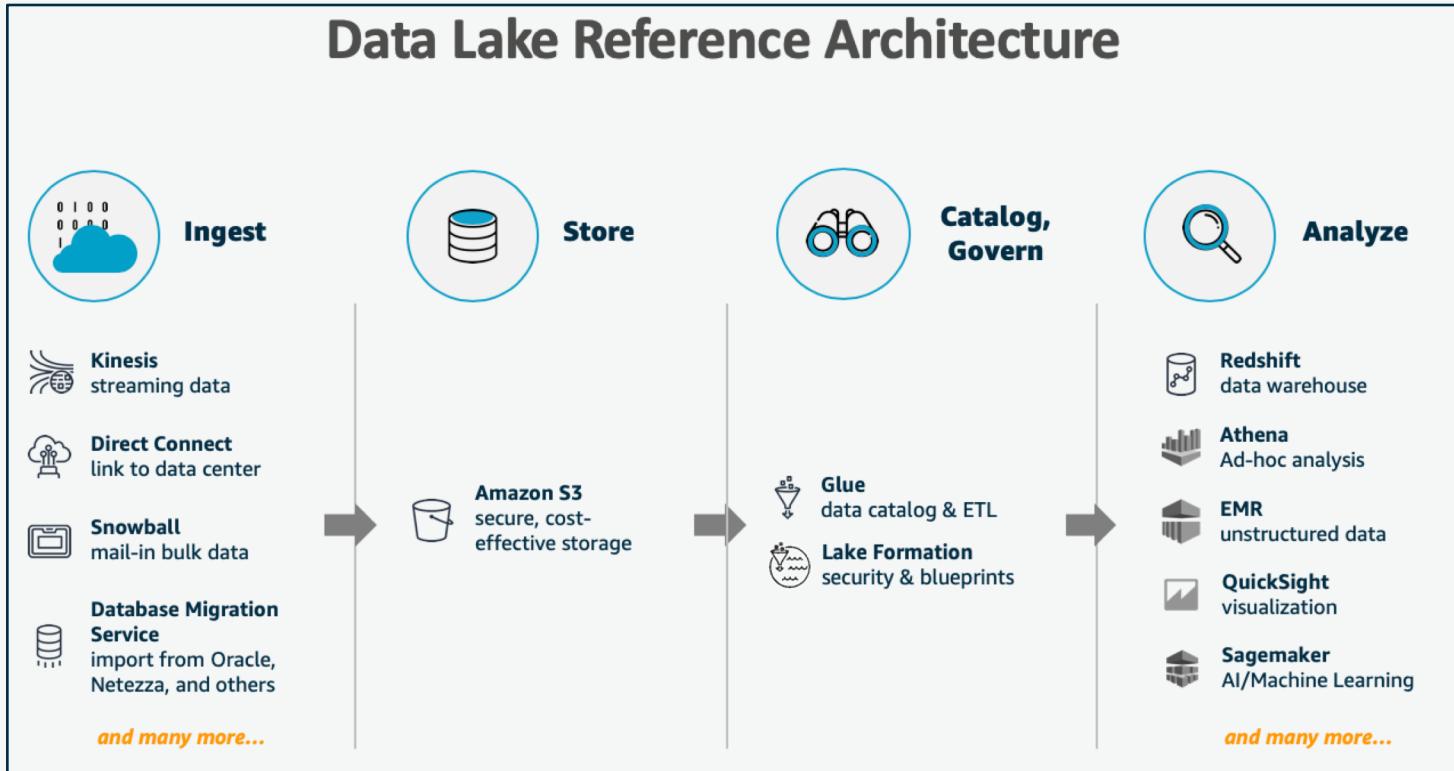
	eventversion	useridentity
1	1.05	{"type": "AssumedRole", "principalId": "AROA3N5FRCFAZFAY4O6R:palisade", "arn": "arn:aws:sts::785789292865:assumed-role/AwsS...
2	1.05	{"type": "AssumedRole", "principalId": "AROA3N5FRCFAZFAY4O6R:Meta31", "arn": "arn:aws:sts::785789292865:assumed-role/AwsS...
3	1.05	{"type": "AssumedRole", "principalId": "AROA3N5FRCFAZFAY4O6R:palisade", "arn": "arn:aws:sts::785789292865:assumed-role/AwsS...
4	1.05	{"type": "AssumedRole", "principalId": "AROA3N5FRCFAZFAY4O6R:palisade", "arn": "arn:aws:sts::785789292865:assumed-role/AwsS...
5	1.05	{"type": "AssumedRole", "principalId": "AROA3N5FRCFAZFAY4O6R:palisade", "arn": "arn:aws:sts::785789292865:assumed-role/AwsS...
6	1.05	{"type": "AssumedRole", "principalId": "AROA3N5FRCFAZFAY4O6R:Meta31", "arn": "arn:aws:sts::785789292865:assumed-role/AwsS...
7	1.05	{"type": "AssumedRole", "principalId": "AROA3N5FRCFAZFAY4O6R:palisade", "arn": "arn:aws:sts::785789292865:assumed-role/AwsS...
8	1.05	{"type": "AssumedRole", "principalId": "AROA3N5FRCFAZFAY4O6R:palisade", "arn": "arn:aws:sts::785789292865:assumed-role/AwsS...
9	1.05	{"type": "AssumedRole", "principalId": "AROA3N5FRCFAZFAY4O6R:palisade", "arn": "arn:aws:sts::785789292865:assumed-role/AwsS...
10	1.05	{"type": "AssumedRole", "principalId": "AROA3N5FRCFAZFAY4O6R:palisade", "arn": "arn:aws:sts::785789292865:assumed-role/AwsS...

Lab Time

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

Continue through the end of Part 1.

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



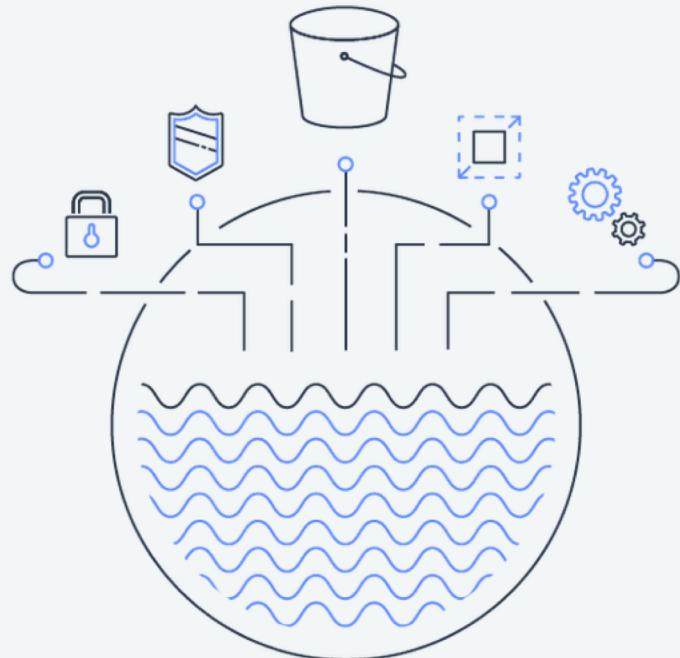
Real-time Data Movement

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

Data Lakes start with
Amazon S3

Why Build a data lake on Amazon S3?

Amazon S3 is designed for 99.999999999% (11 9s) of data durability. With that level of durability, you can expect that if you store 10,000,000 objects in Amazon S3, you should only expect to see an issue once every 10,000 years!



Security by design

Protect data with an infrastructure designed for the most data-sensitive organizations

Scalability on demand

Instantly scale up storage capacity, without lengthy resource procurement cycles

Durable against the failure of an entire AWS Availability Zone

Automatically store copies of data across a minimum of three Availability Zones (AZs). To provide fault tolerance, Availability Zones are separated by several miles—but no more than a hundred to ensure low latencies.

AWS services for analytics, HPC, AI, ML, and media data processing

Use AWS native services to run applications on your data lake

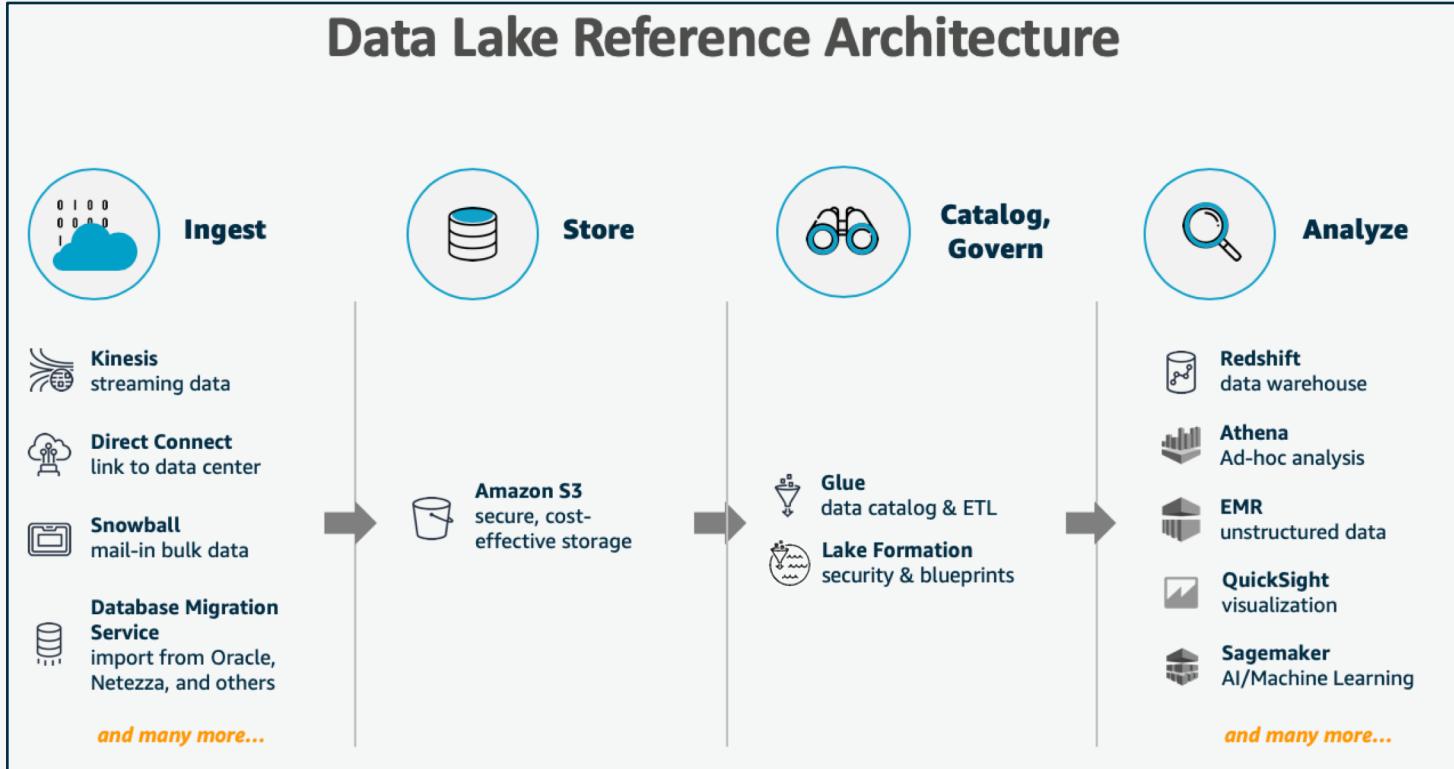
Integrations with third-party service providers

Bring preferred analytics platforms to your S3 data lake from the [APN](#).

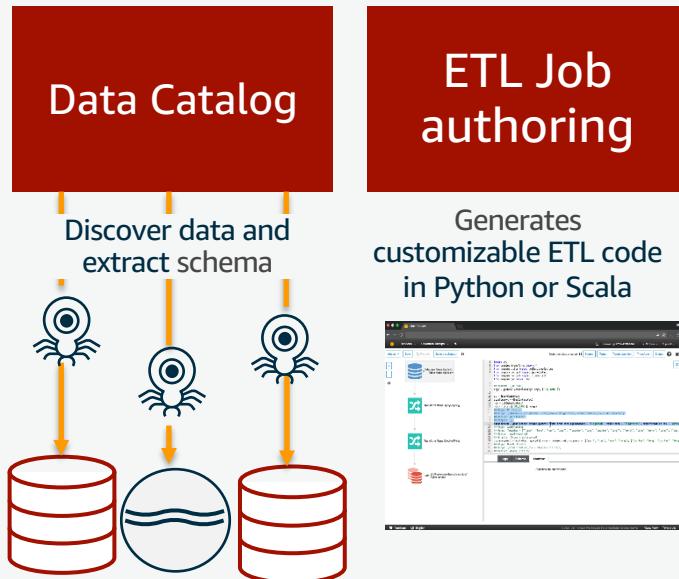
Wide range of data management features

Comprehensive flexibility to operate at an object level while managing at scale, configure access, enable cost efficiencies, and audit data across an S3 data lake.

AWS Glue



AWS Glue: Data Catalog & ETL Service



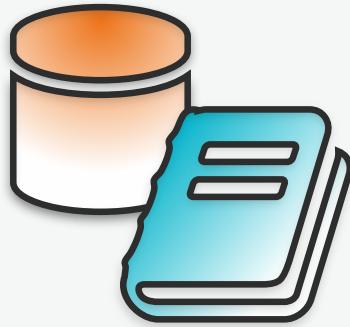
Automatically discovers data and stores schema

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

Automatically generates customizable ETL code

Schedules and runs your ETL jobs

Serverless

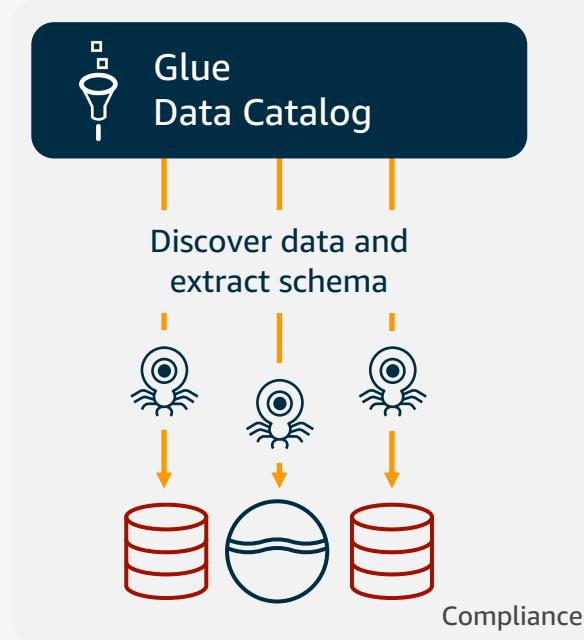


Glue data catalog

Discover and organize your data sets

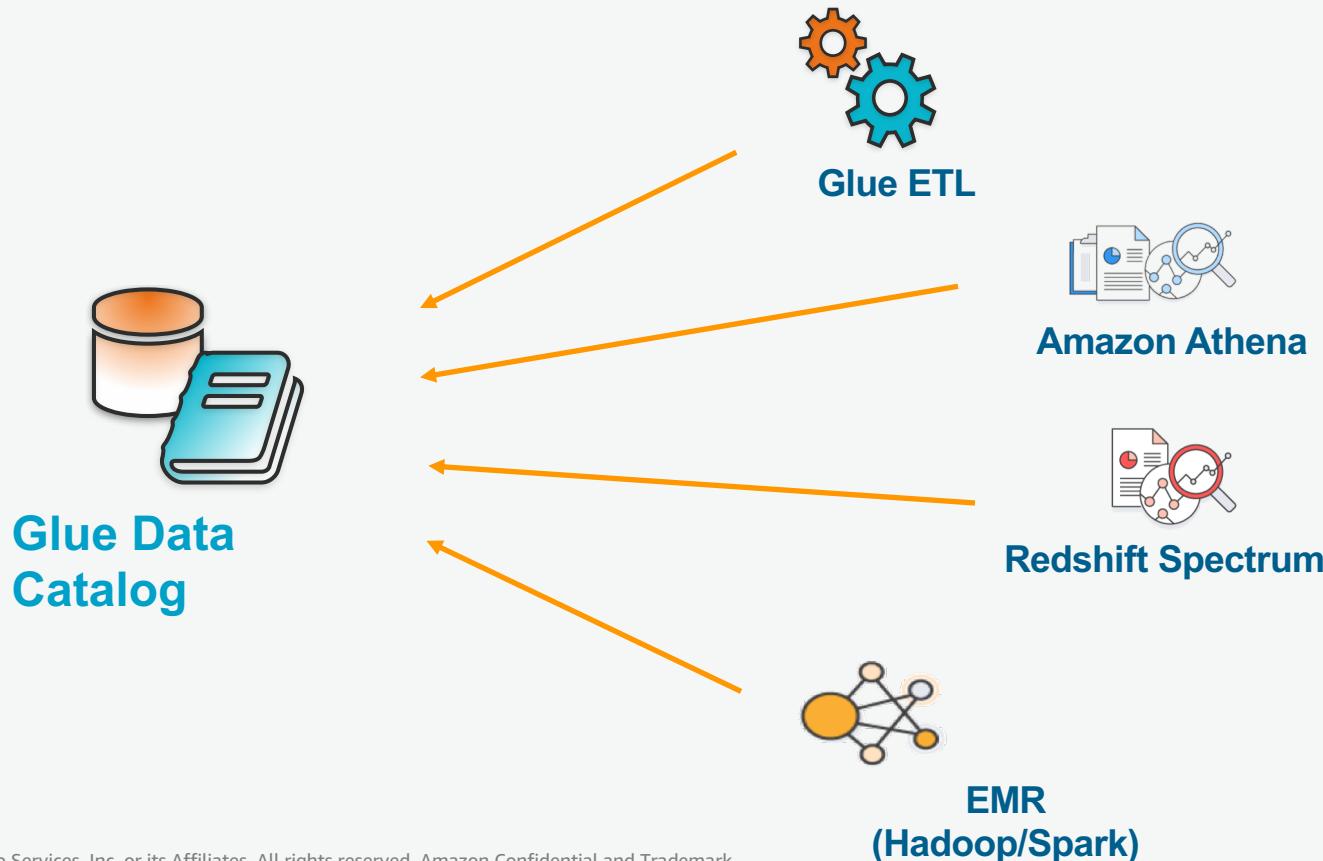
AWS Glue—Data Catalog

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



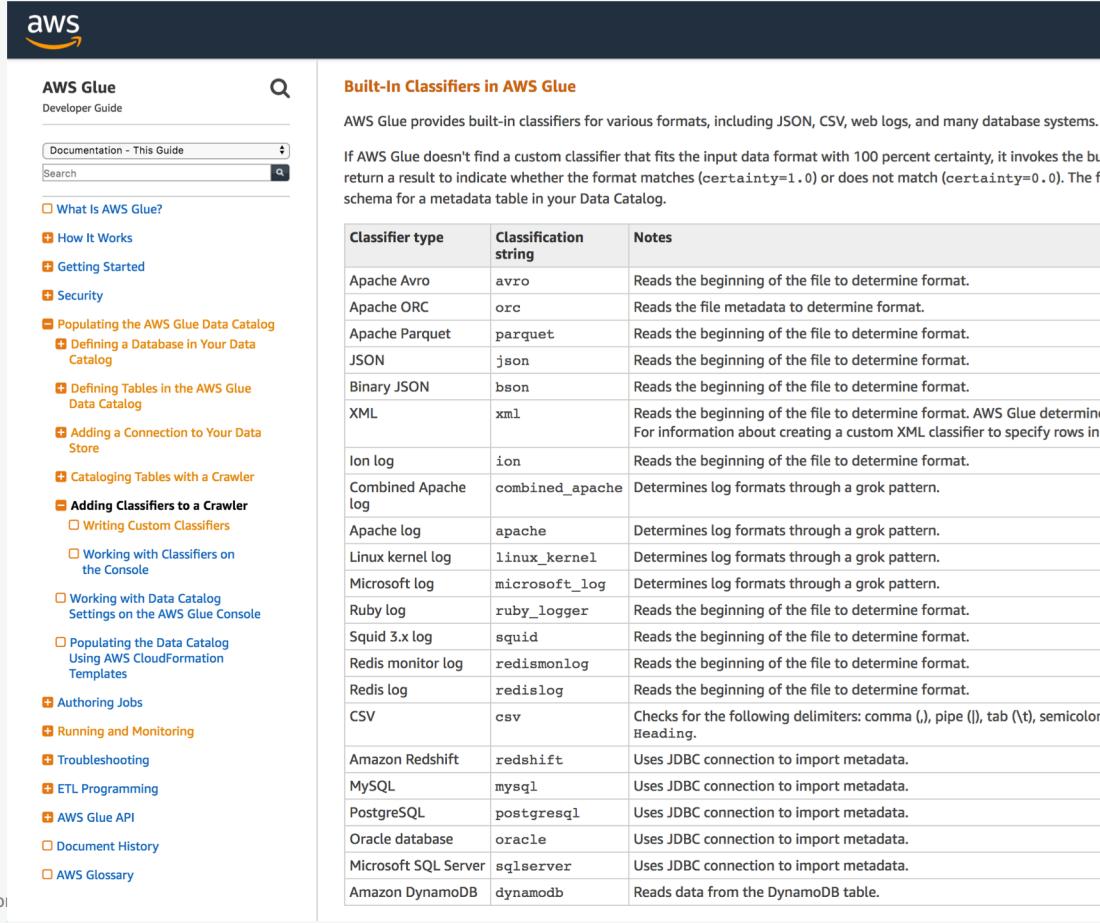
Glue: Data Catalog - Crawlers

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

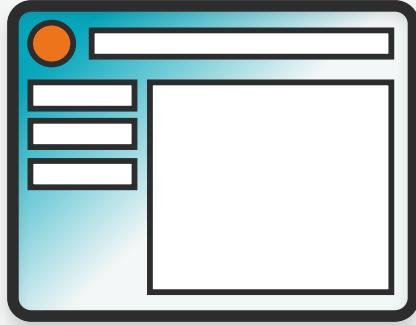
AWS Glue provides built-in classifiers for various formats, including JSON, CSV, web logs, and many database systems. If AWS Glue doesn't find a custom classifier that fits the input data format with 100 percent certainty, it invokes the built return a result to indicate whether the format matches (`certainty=1.0`)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



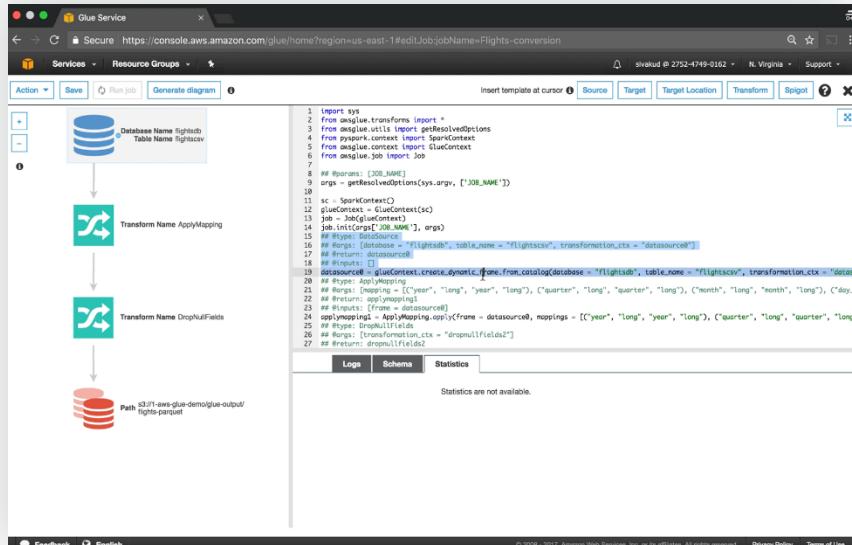


Glue ETL Jobs

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	crime_time	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?

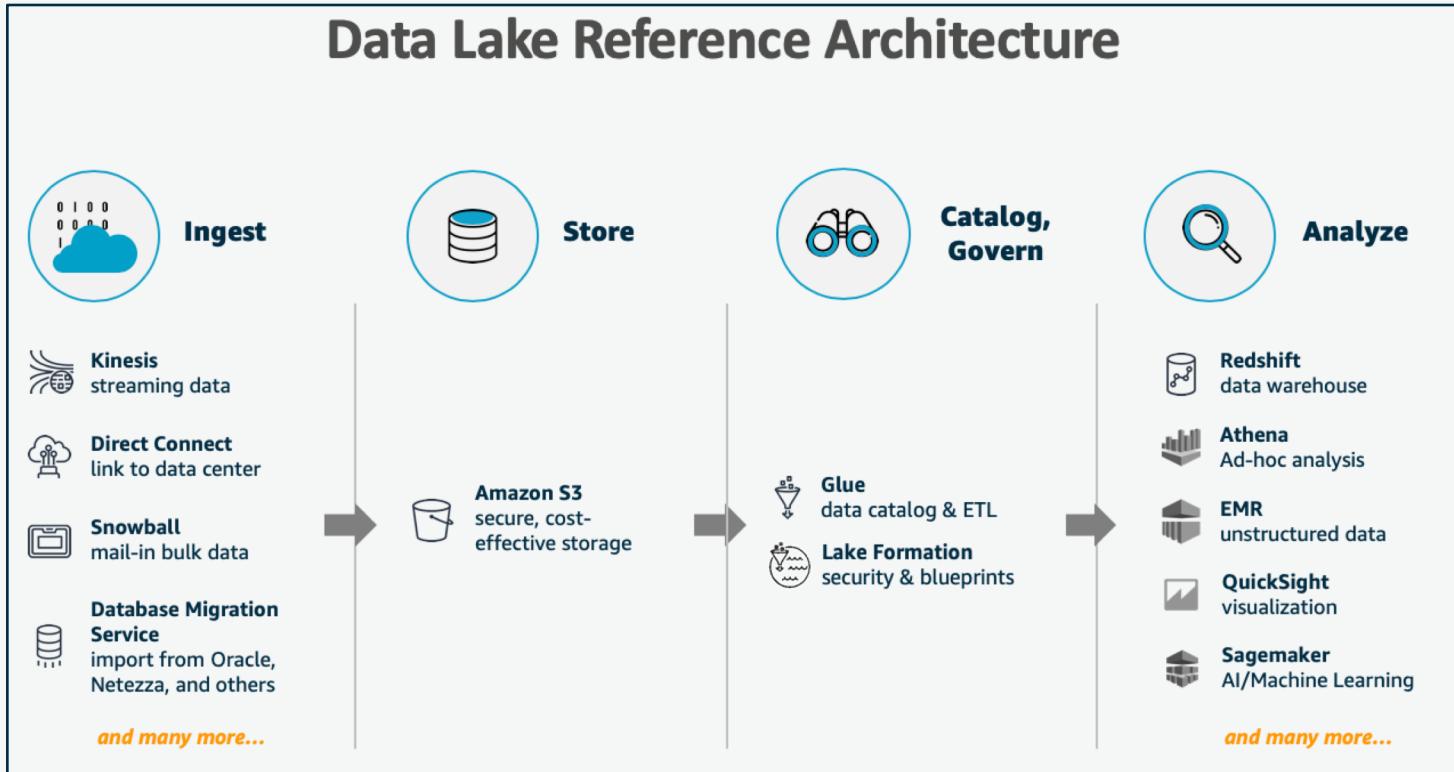
- Parallel, scale-out data processing engine
- Fault-tolerance built-in
- Flexible interface: Python scripting, SQL
- Rich eco-system: ML, Graph, analytics, ...

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

Data Lake Reference Architecture



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



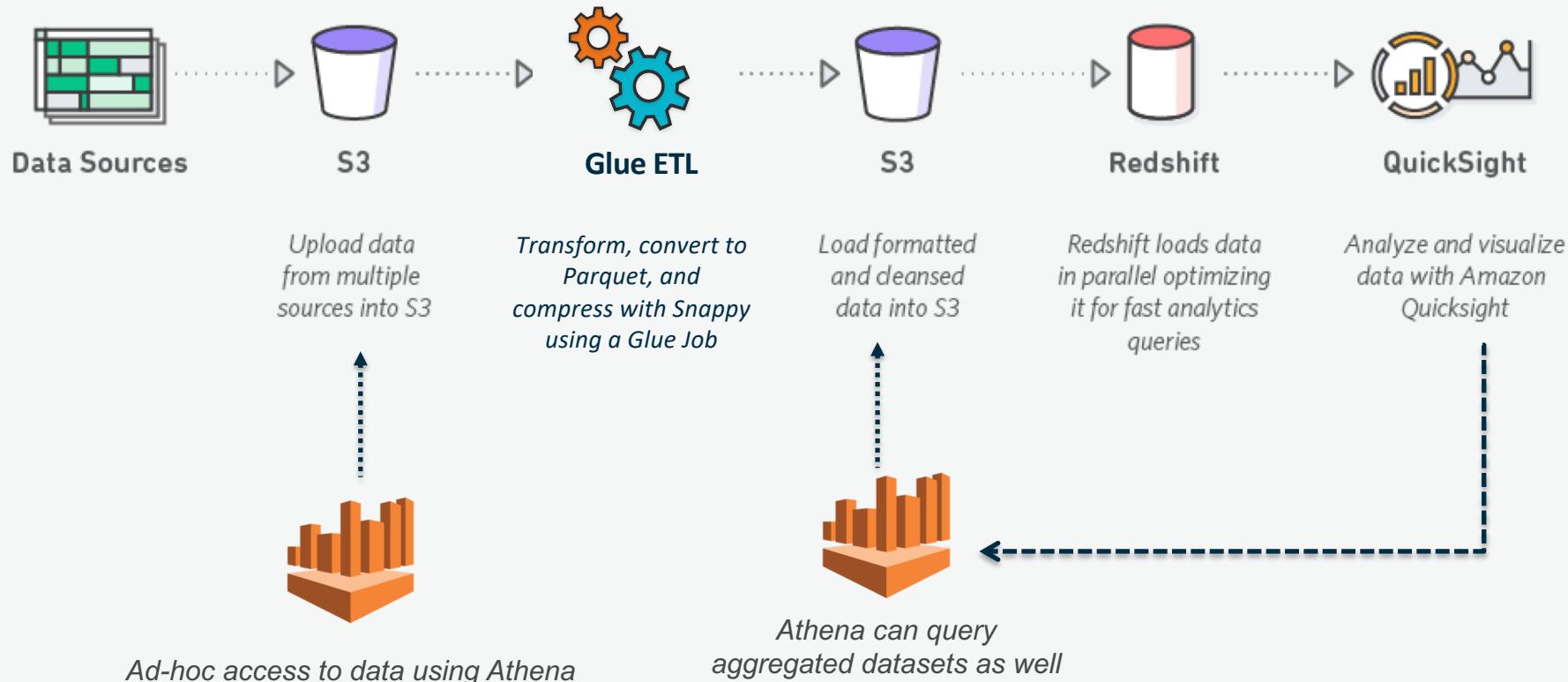
ANSI SQL interface, JDBC/ODBC drivers, multiple formats, compression types, and complex joins and data types

Easy



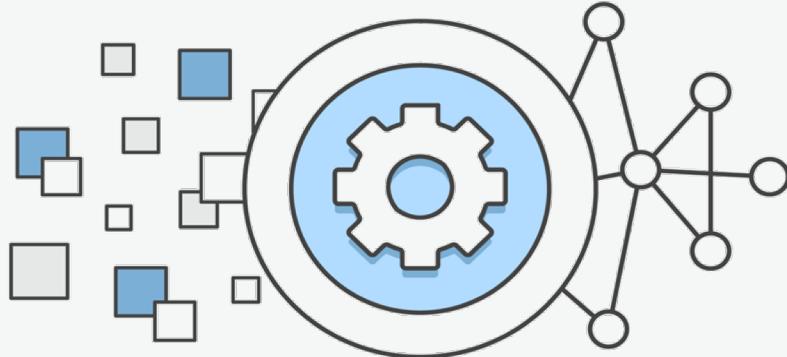
Serverless: zero infrastructure, zero administration
Integrated with QuickSight

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 continue thru the end of Part2.

Lunch

We will start again at 1:00pm

Restart

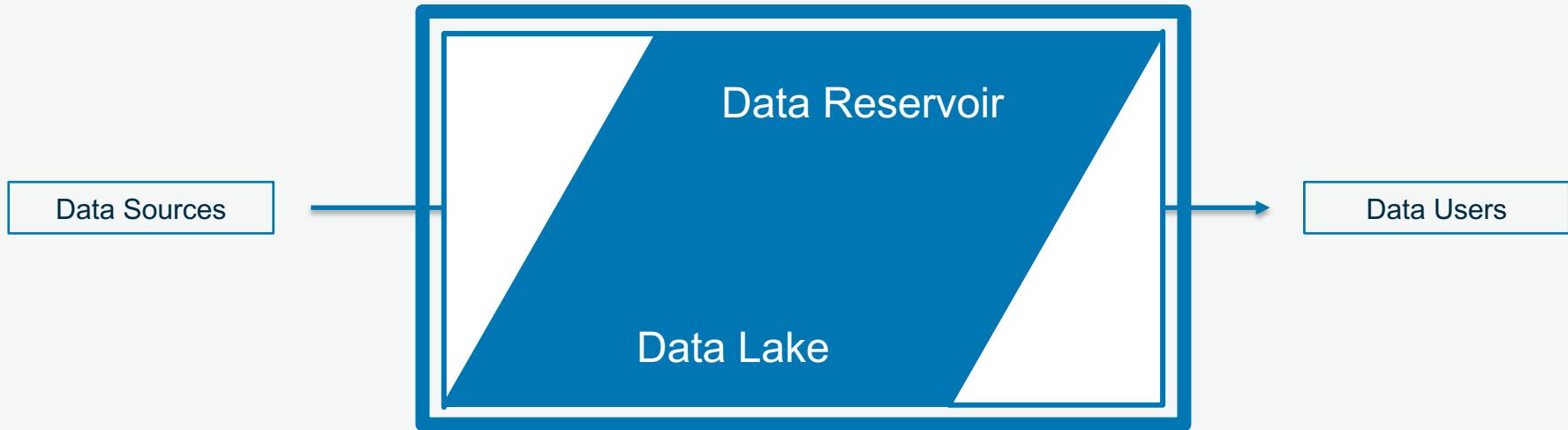
What did you think of the labs so far?
Any questions about the content so far?

Cloud Data Warehouse

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



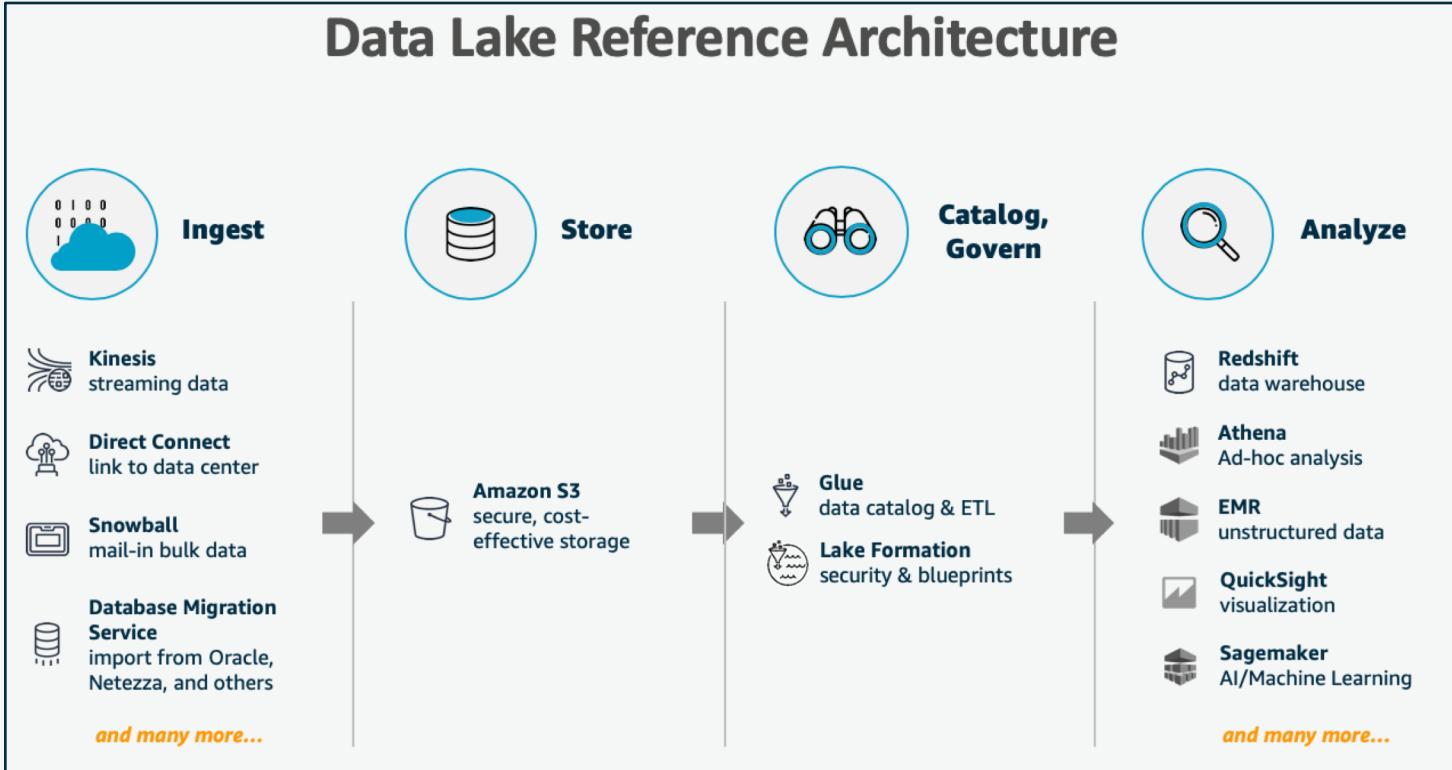
Data Lakes are optimized for large volumes of storage and ease of ingestion of any kind or flavor of data.

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

Data Lake Reference Architecture



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

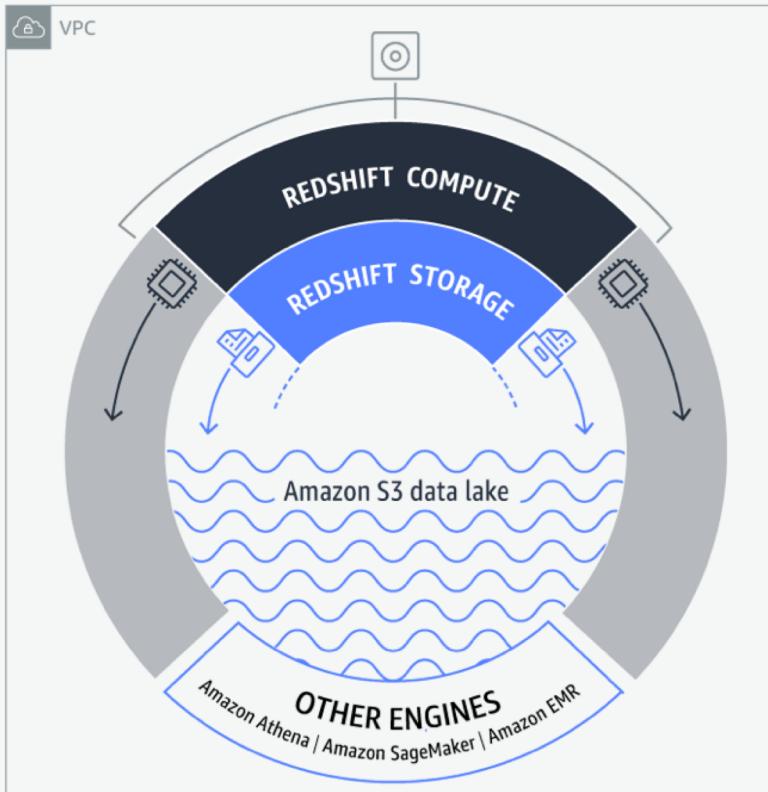
Lab Time

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

Please start Part 3.

Continue thru the Provision a new Redshift Cluster step.

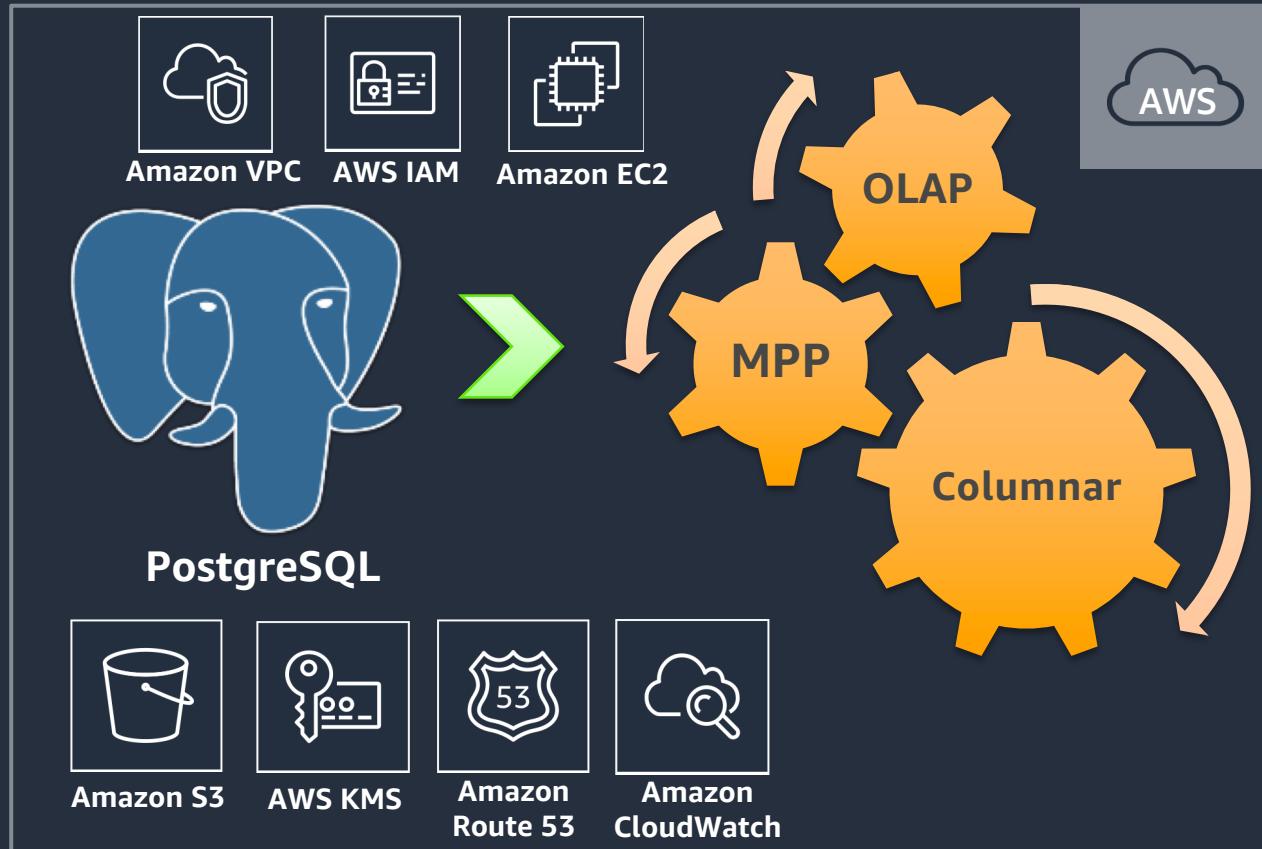
Redshift Conceptual Architecture



- Leader node creates query execution plans using machine learning and caches query results
- Compute scales to handle high concurrency and query the data lake directly
- Storage scales to exabytes of data in the Amazon S3 data lake in Parquet, ORC, and other open data formats



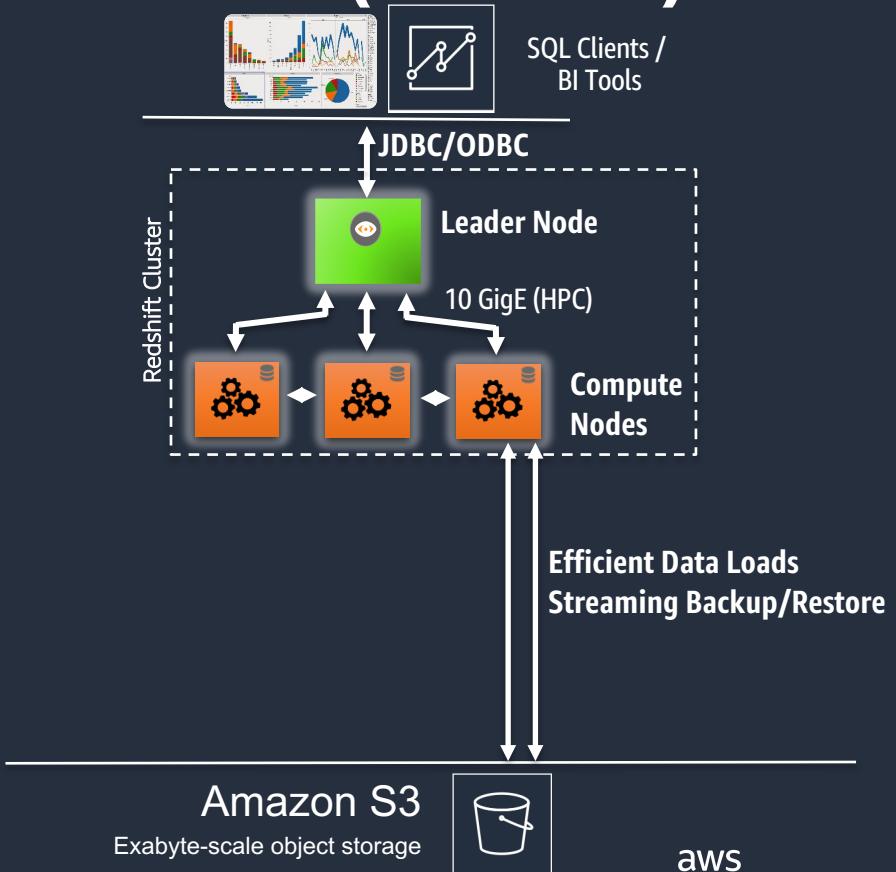
Redshift: What's Under the Hood?



Redshift Cluster Architecture (Classic)



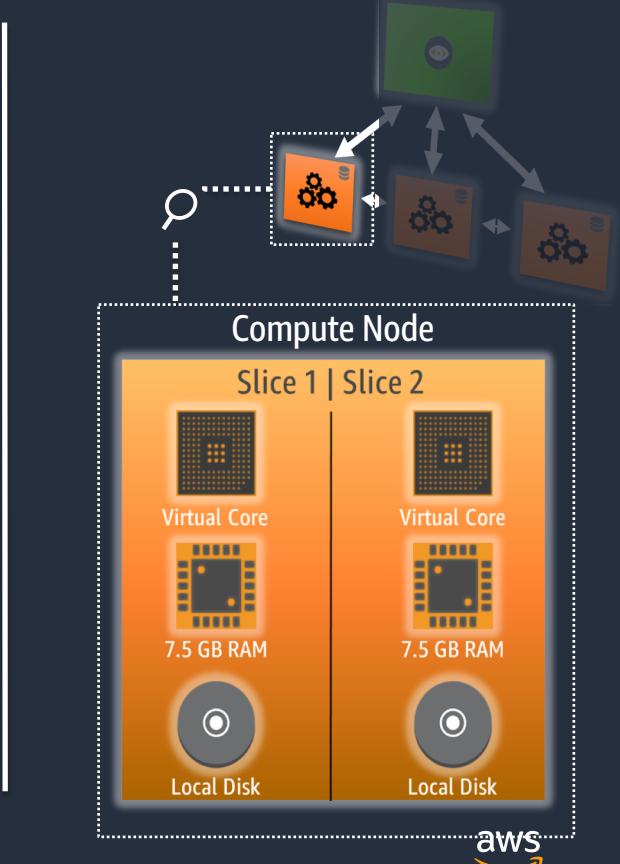
- Massively parallel, shared nothing architecture
- Streaming Backup/Restore from S3
- Leader node
 - SQL endpoint
 - Stores metadata
 - Coordinates parallel SQL processing; ML optimizations
 - Customers are not charged for the leader node for any cluster with two or more nodes
- Compute nodes
 - Local,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





Compute Node: Under the Hood (Slices)

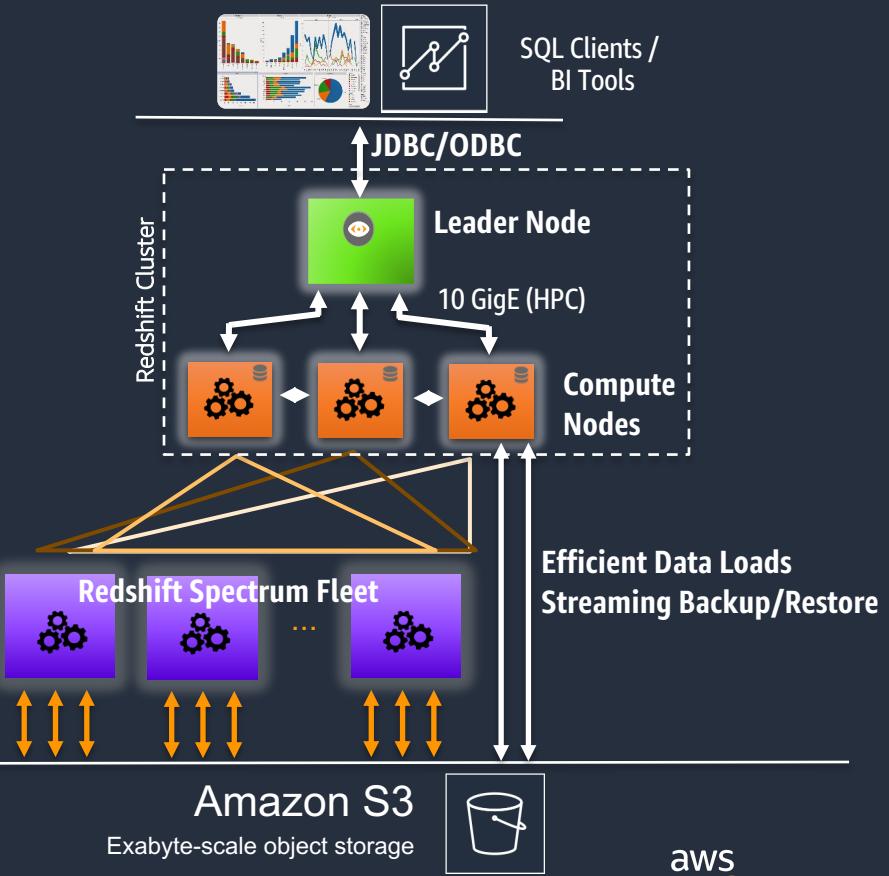
- A compute node is partitioned into either 2 or 16 *slices*; a slice can be thought of as a “virtual compute node”
- Each slice is allocated a portion of the compute node's memory and disk space, where it processes a portion of the workload assigned to the compute node by the leader node
- The leader 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



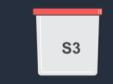
Elastic & highly available



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

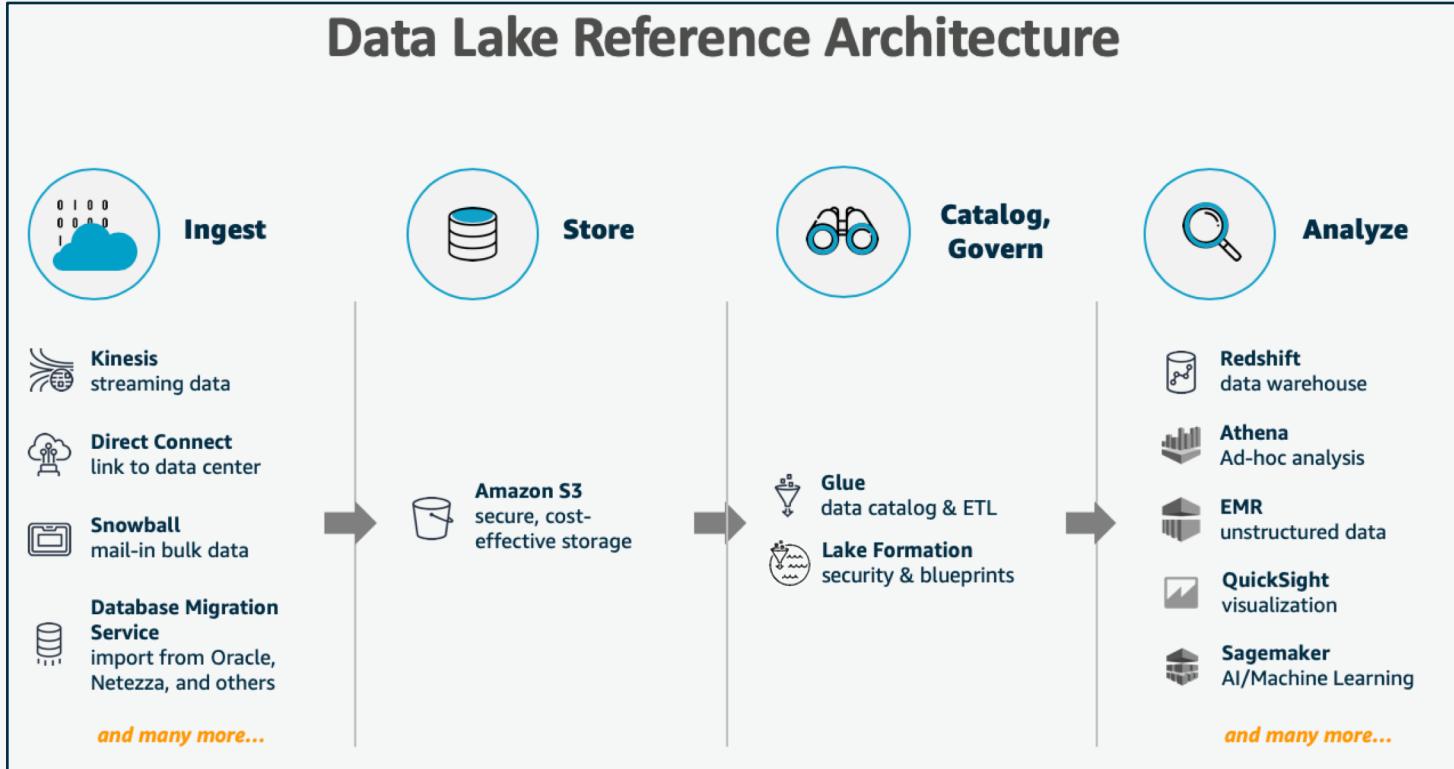
<https://github.com/dbayardAWS/intro-data-lake>

Please finish Part 3.

Populating the Data Lake/Warehouse

Data Ingestion

Data Lake Reference Architecture



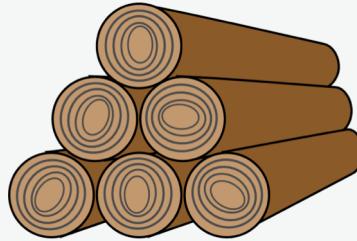
Data Sources



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

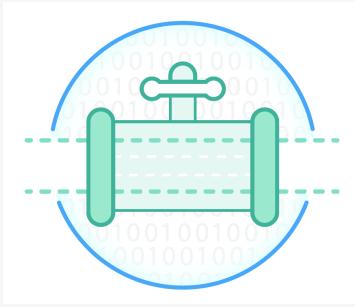
Streams



Collecting and Analyzing

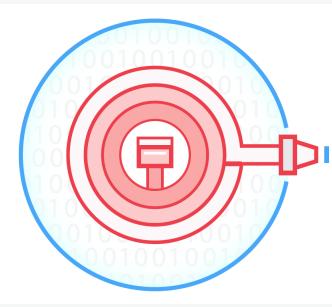
- Amazon Kinesis
- Amazon Managed Streaming for Kafka (MSK)
- Example: Clickstream Analytics

Amazon Kinesis - Stream Processing on AWS



Streams

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



Firehose

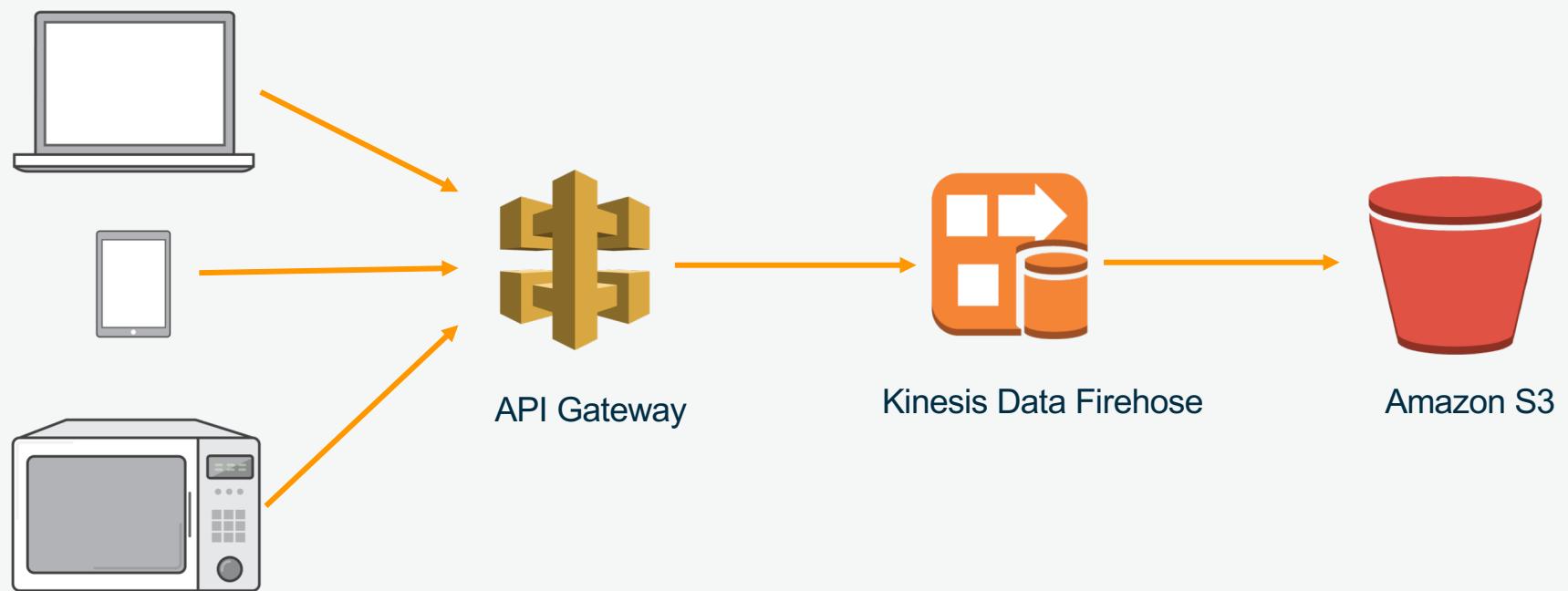
- Buffer records in a stream into a single output for more efficient storage
- Automatic flushing of buffer to S3, ElasticSearch, Redshift, or Splunk



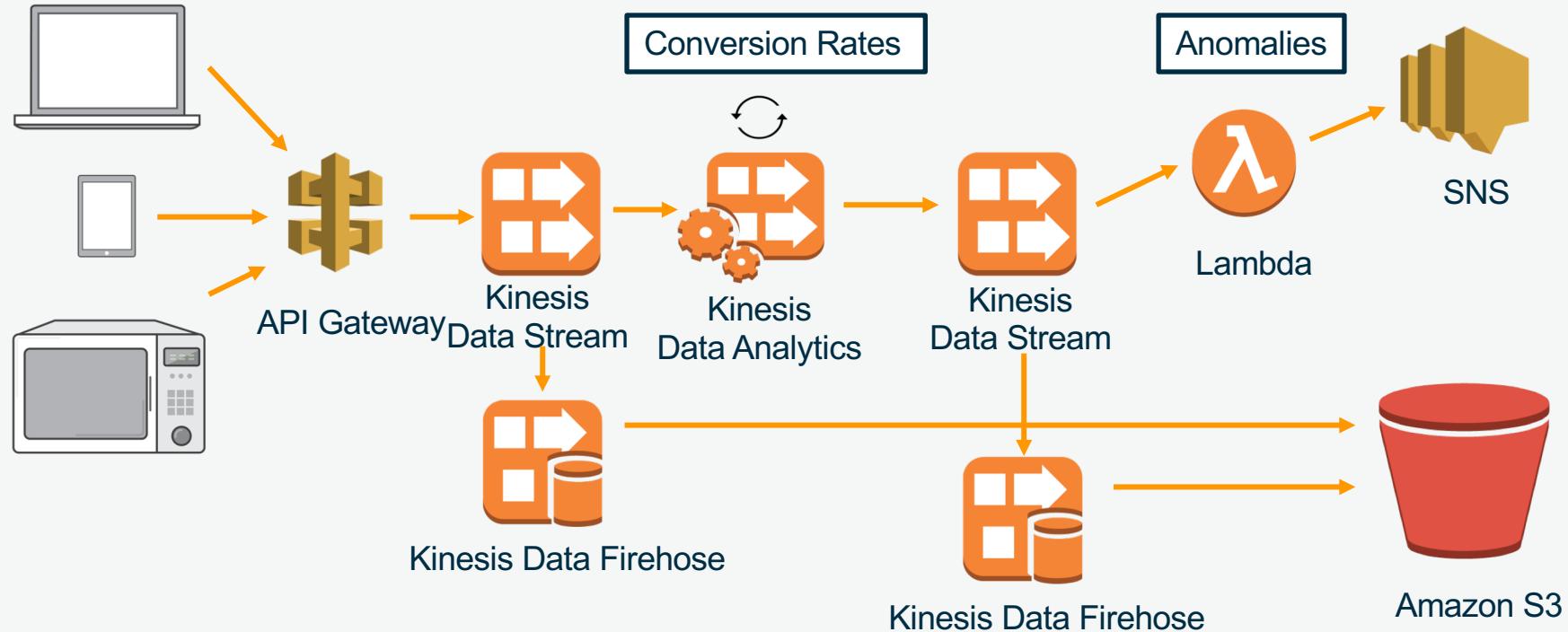
Analytics

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

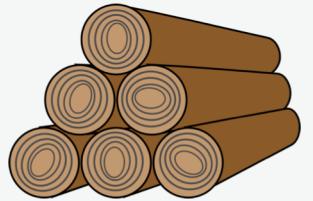
Clickstream



Clickstream with Real-Time Analytics



Data Sources - Logs

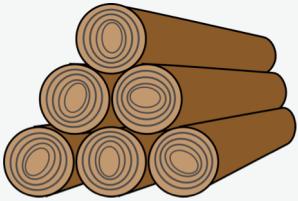


Logs



Amazon S3

Logs



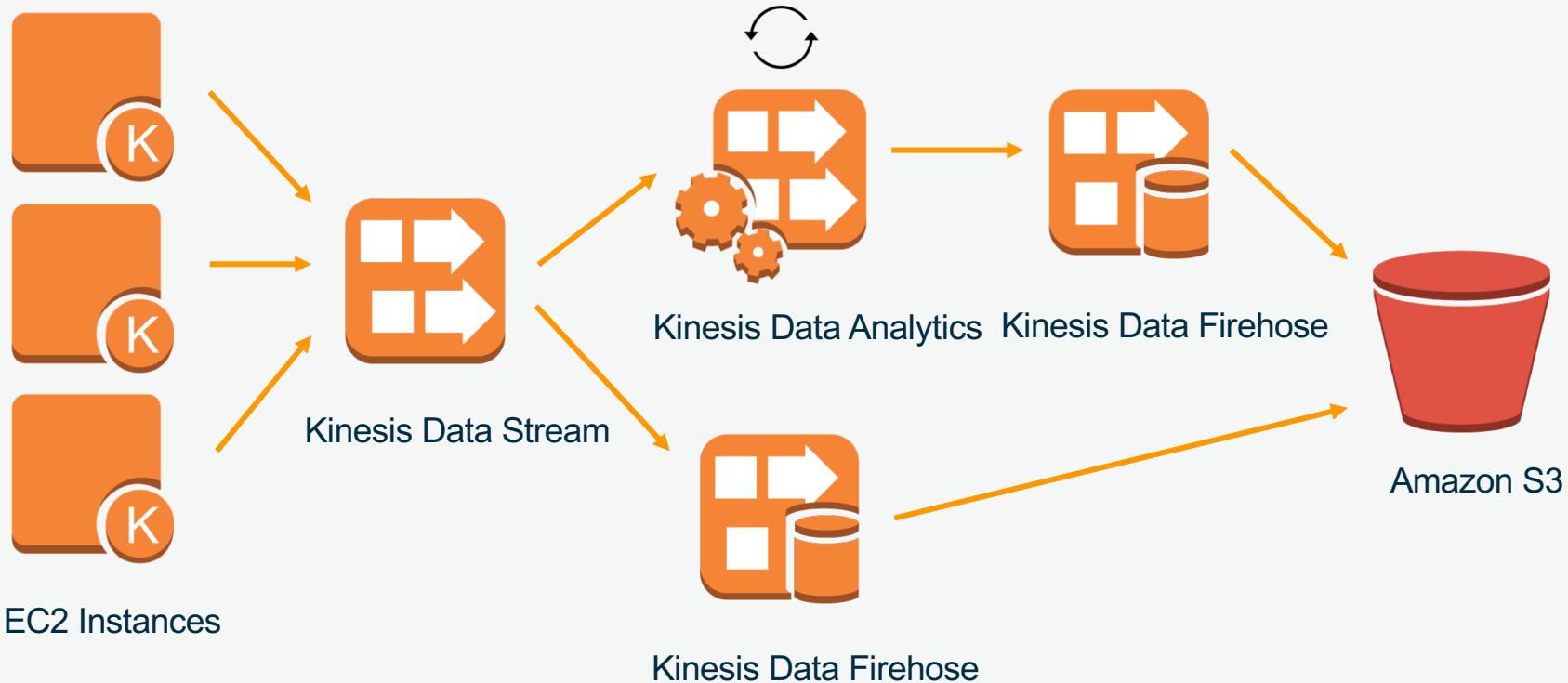
Collecting and Analyzing

- Amazon CloudWatch
- Amazon Kinesis
- Other Options

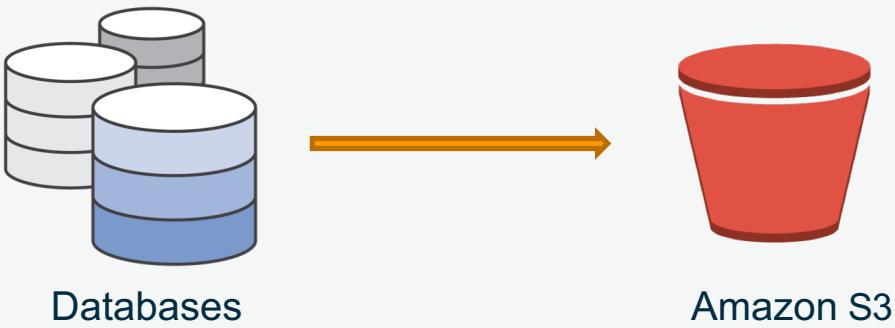
Logs – Kinesis Agent



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 –

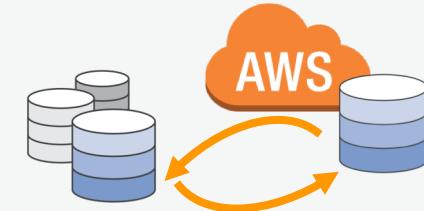
- Commercial to Redshift

Migrate



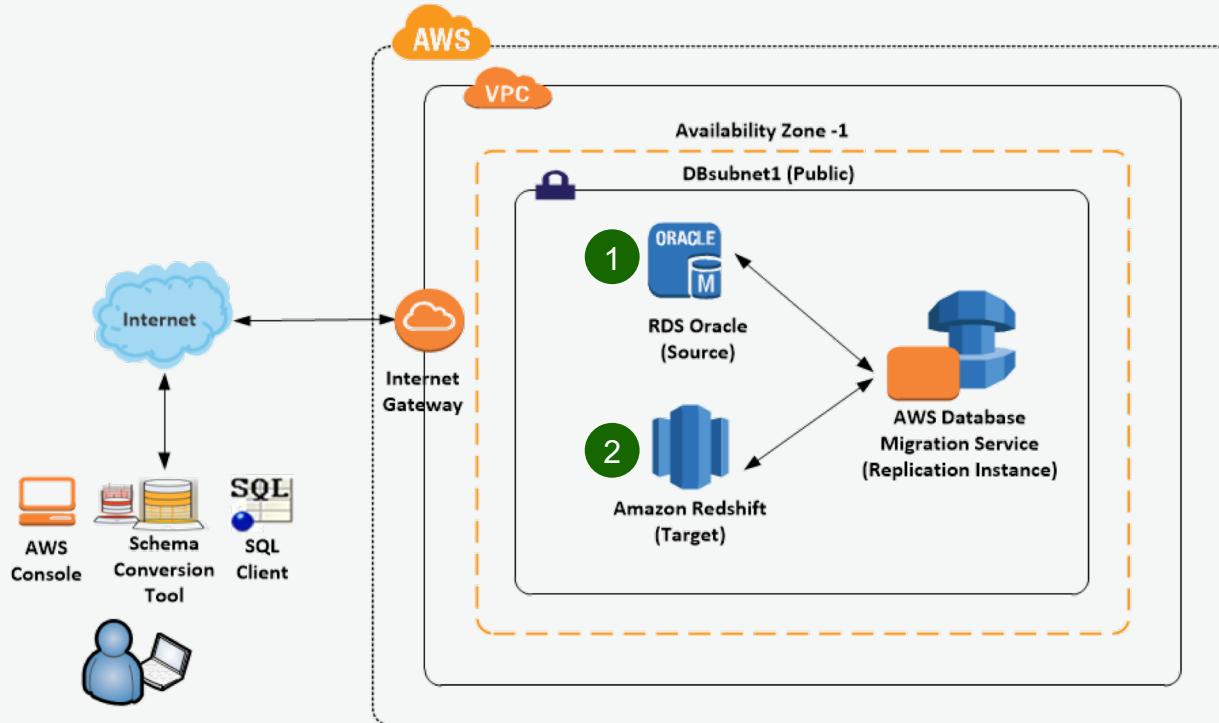
- Migrate business-critical applications
- Migrate from Classic to VPC
- Migrate data warehouse to Redshift
- Upgrade to a minor version

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

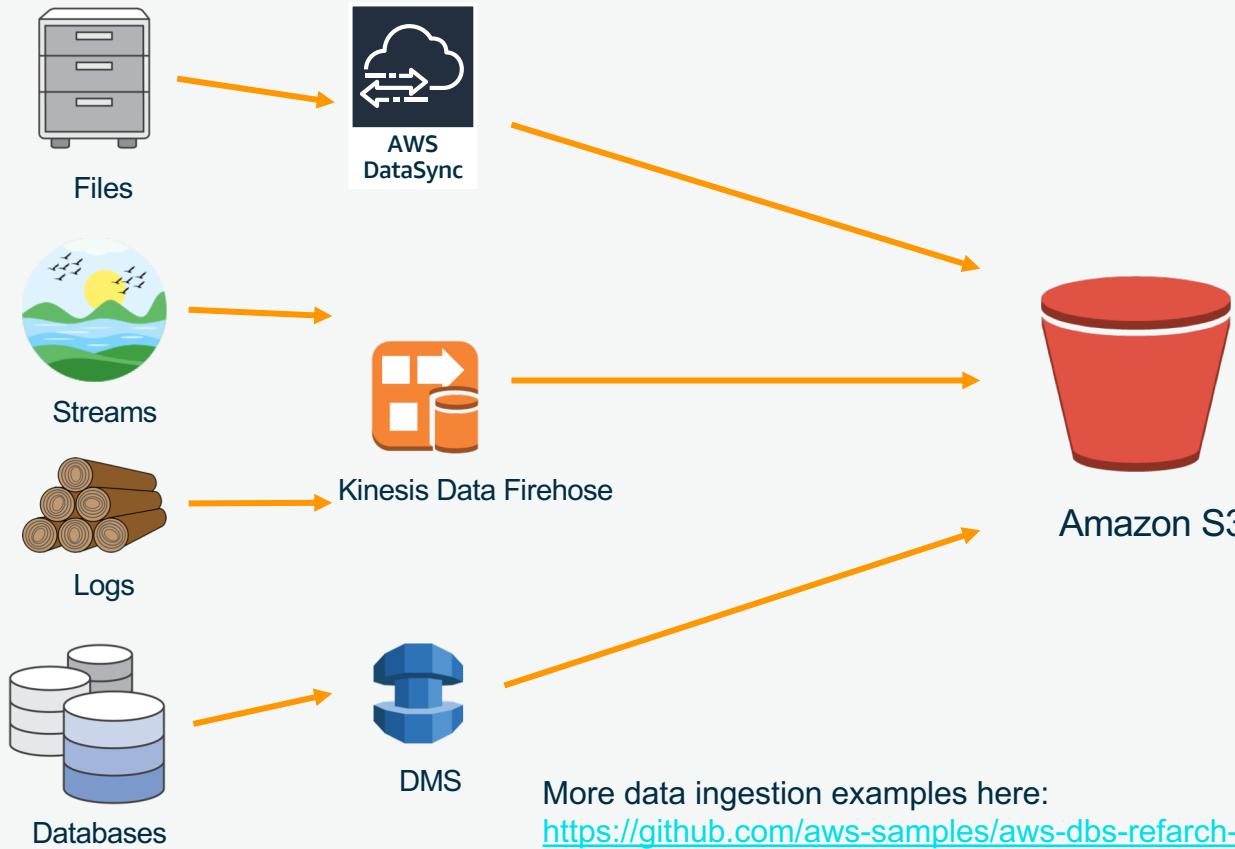
Bulk File

```
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:

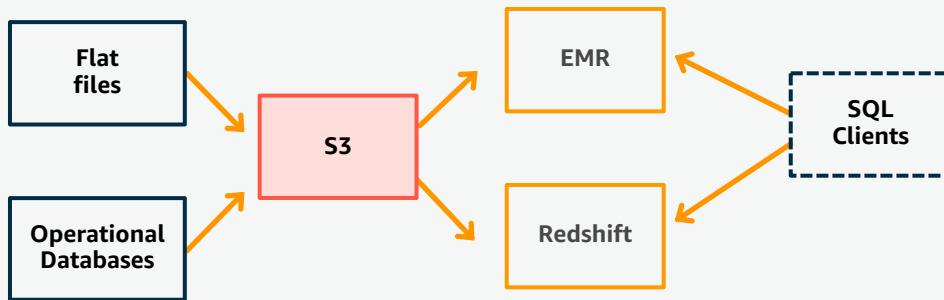
Nasdaq wanted to make their large historical data footprint available to analyze as a single dataset.

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:

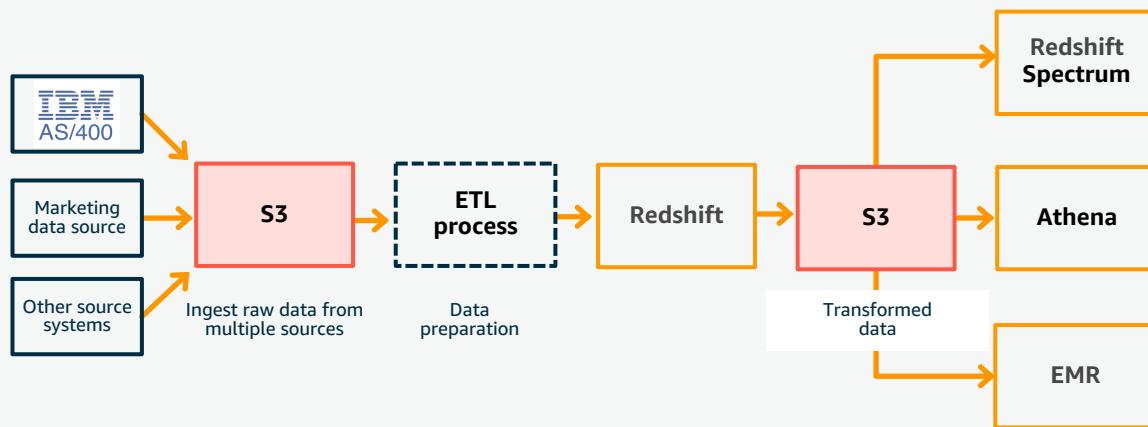
Large volumes of data in multiple systems. Also, high costs from maintaining on-premises EDW deployment.

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:

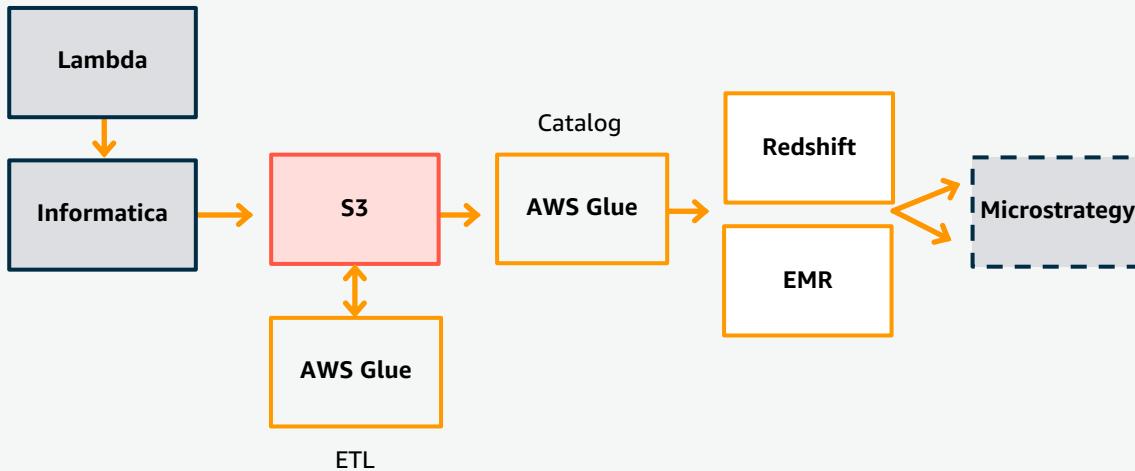
- Land data in S3 as a data lake
- Use Redshift and EMR as analytics engines to process data
- Saved 15–20% in overall costs (on-premises) with 30% of performance gain



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:

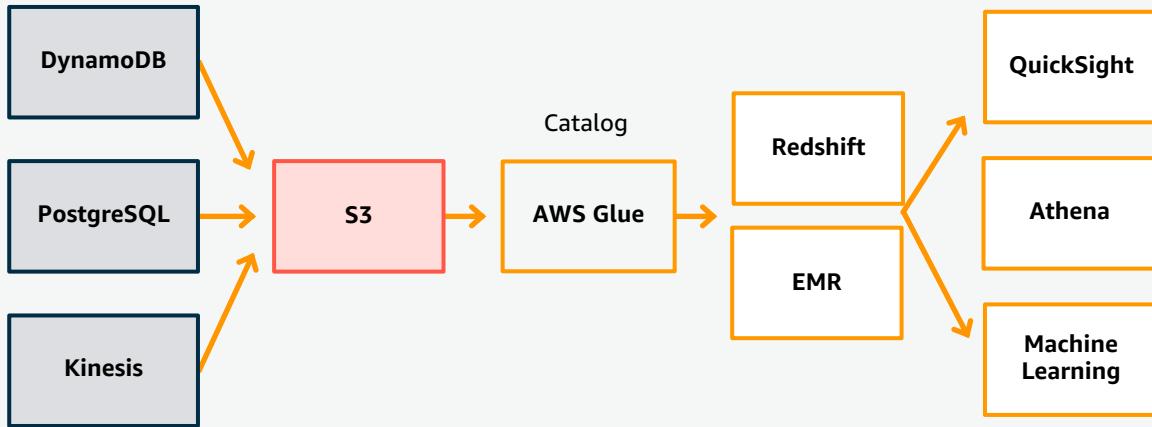
Load 500K+ transactions each day, and serve 300K+ queries/extracts each day from Amazon businesses (Amazon.com, Amazon Prime, Amazon Music, Amazon Alexa, Amazon Video, and Twitch).

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

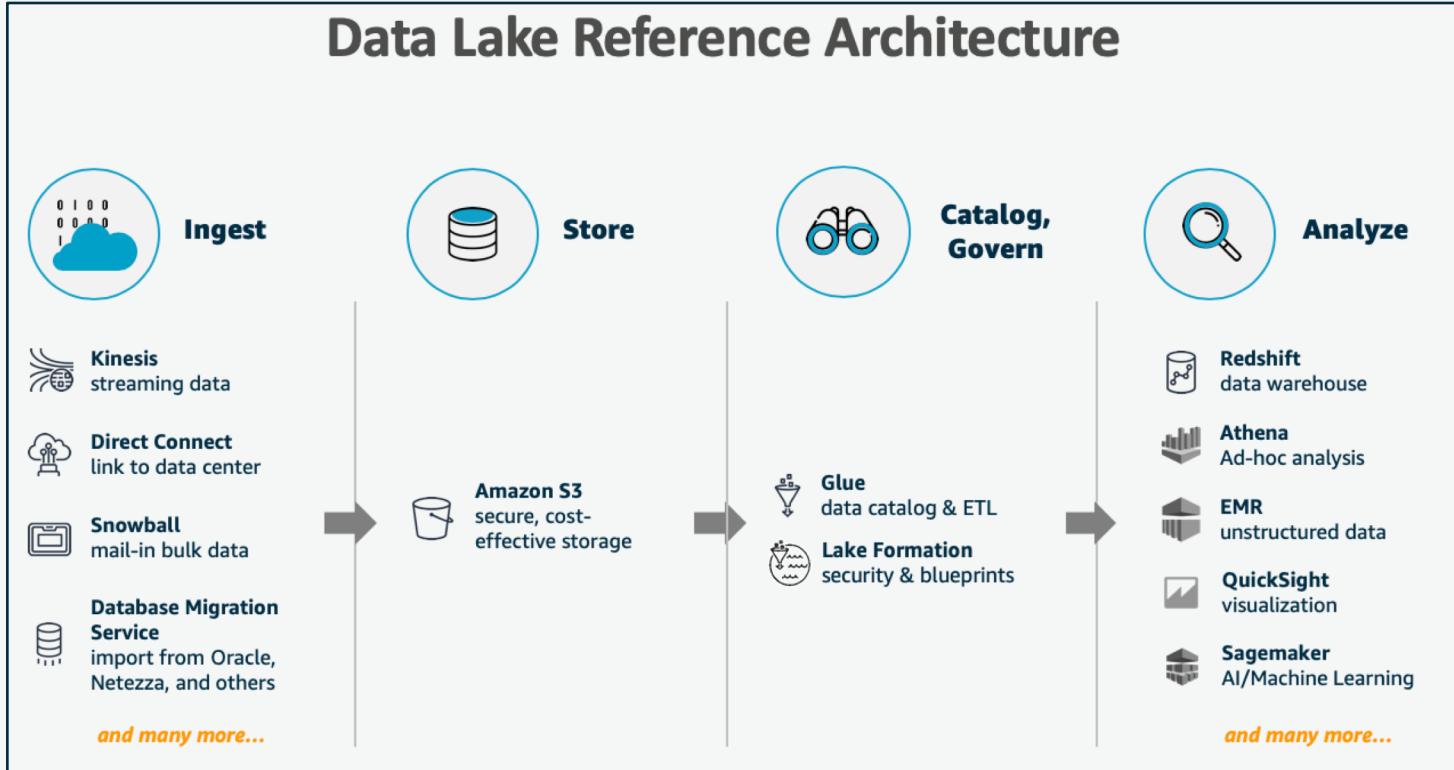


- DynamoDB capturing all Amazon.com transactions
- Everything from DynamoDB, RDS PostgreSQL and Kinesis fed to a S3 data lake
- Glue used to catalog the data
- Redshift used for all SQL-based queries, and EMR for all machine learning and big data processing
- End-users use QuickSight for visualizations

Visualizing Data

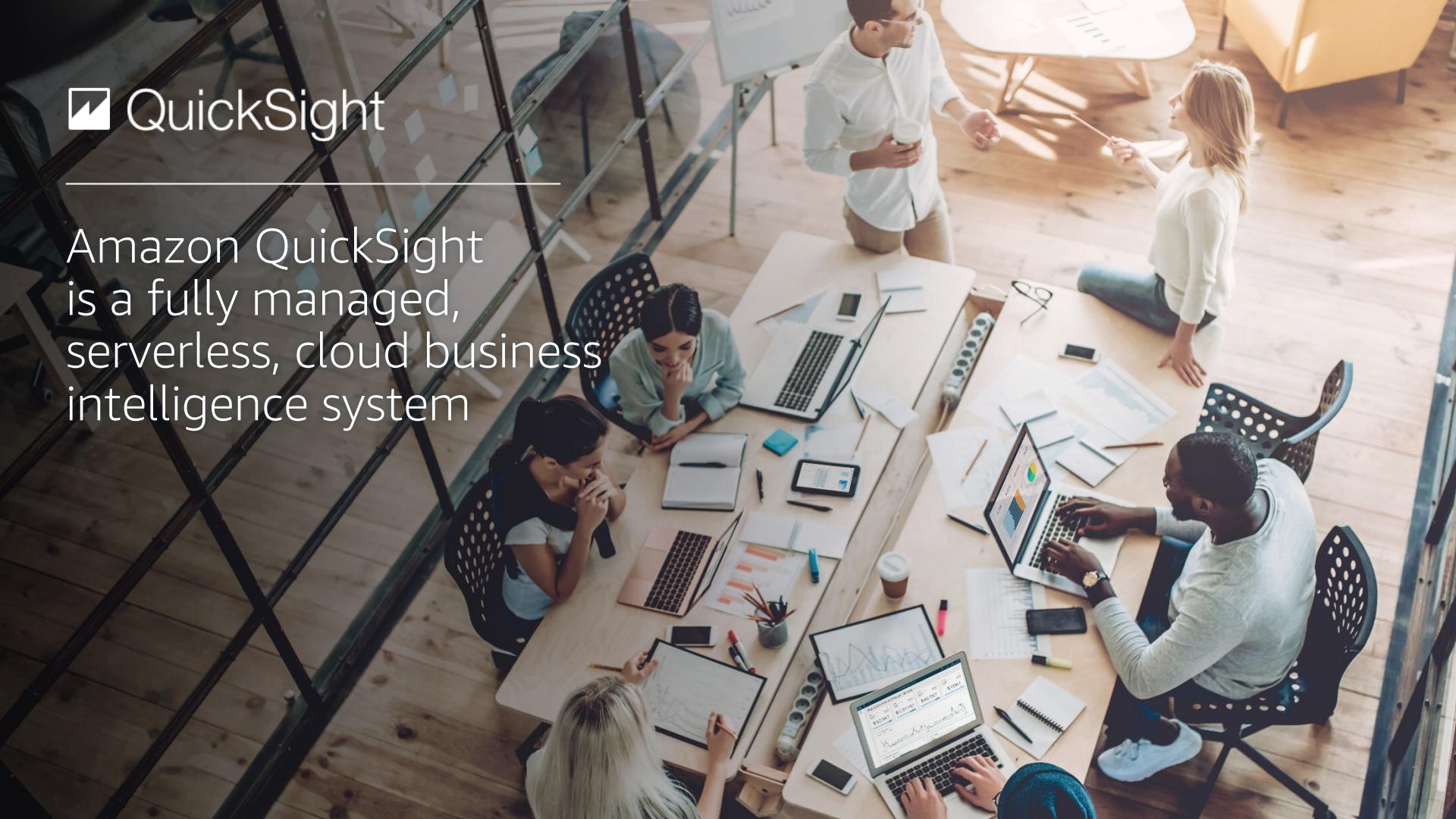
Amazon QuickSight

Data Lake Reference Architecture





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

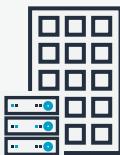
QuickSight integrates with your other AWS services and data sources giving you everything you need to build an end-to-end cloud analytics solution.

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



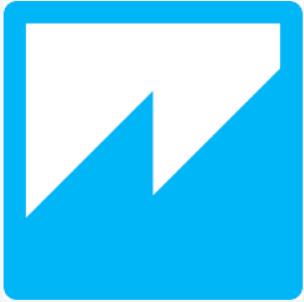
Applications

Connect directly to third party business applications



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

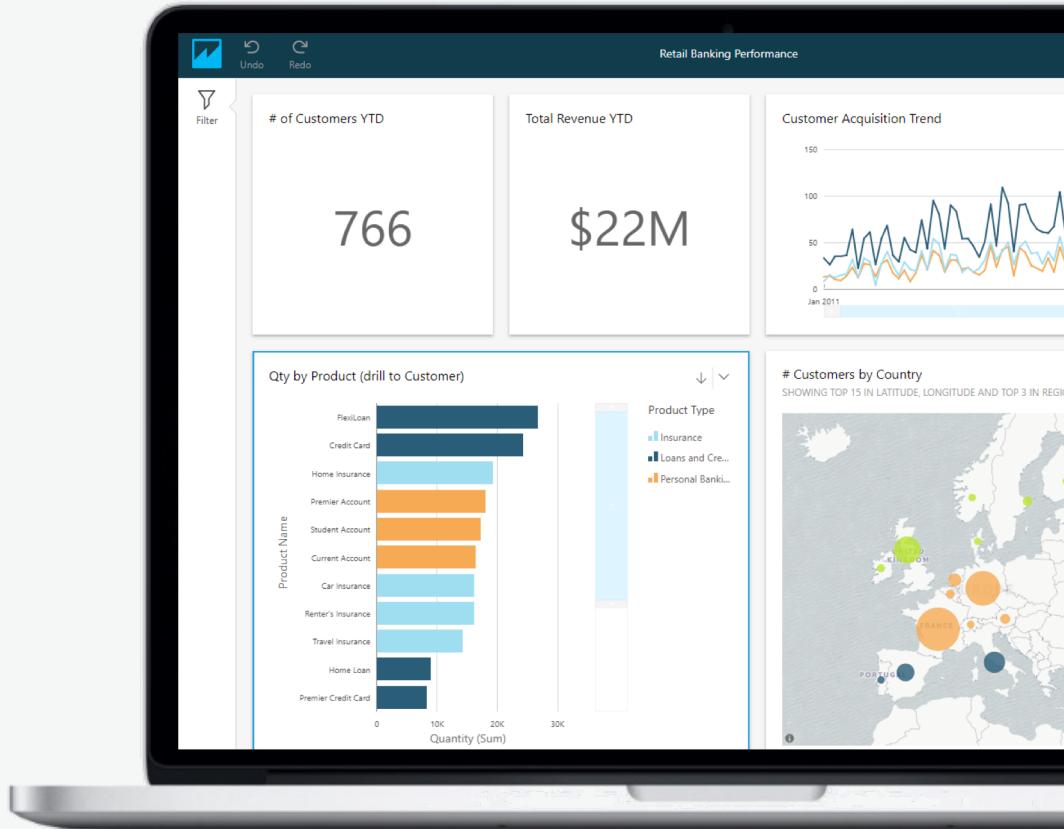




Demonstration

Create beautiful, interactive dashboards.

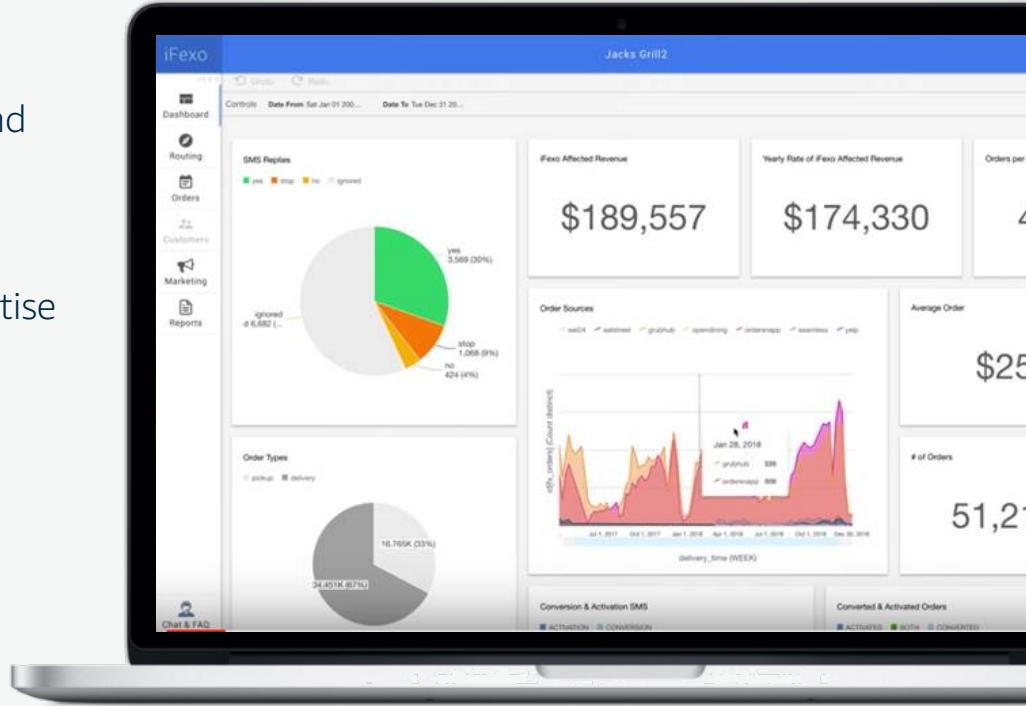
- Add rich interactivity like filters, drill downs, zooming, and more
- Blazing fast navigation
- Accessible on any device
- Data Refresh
- Publish to everyone with a click



Embedding Dashboards In Your Application

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

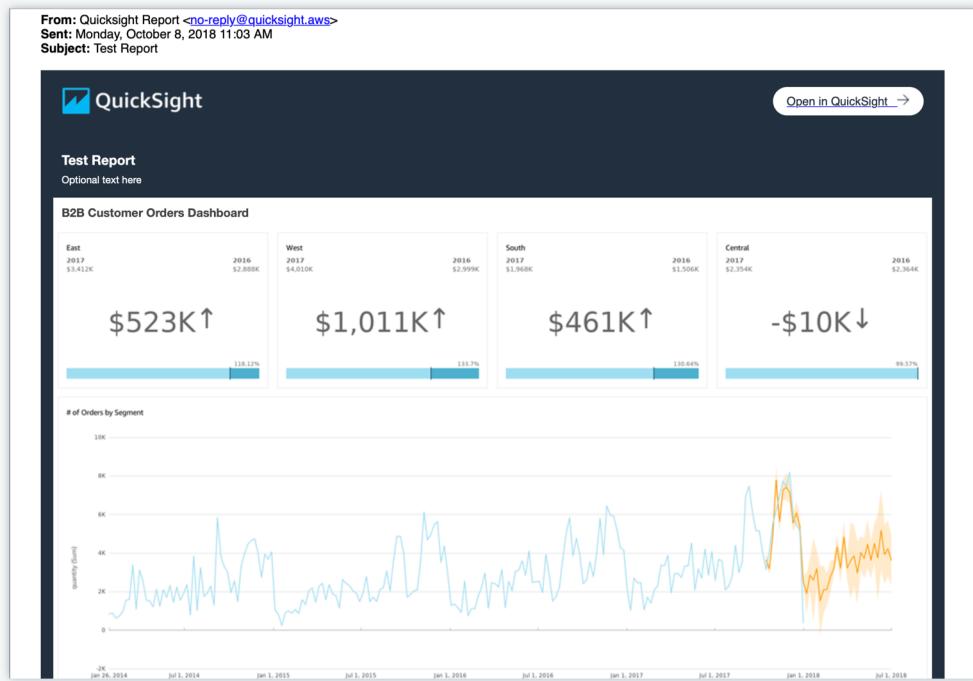
- Enhance your applications with rich analytics and dashboards
- Easy maintenance, no servers to manage
- Fast! No Custom development or domain expertise needed
 - Leverage new features as we add them
 - Utilizes Pay-per-Session Pricing.



Insights Delivered to Your Inbox

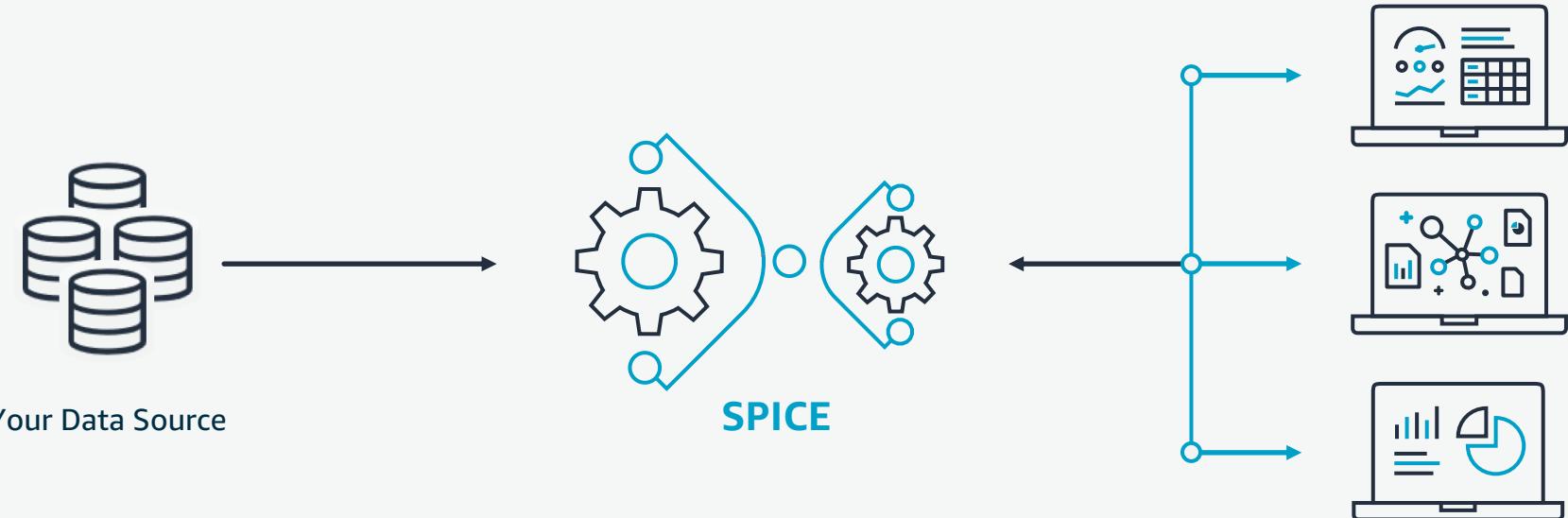
QuickSight lets you send report snapshots directly to your users inbox.

- 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



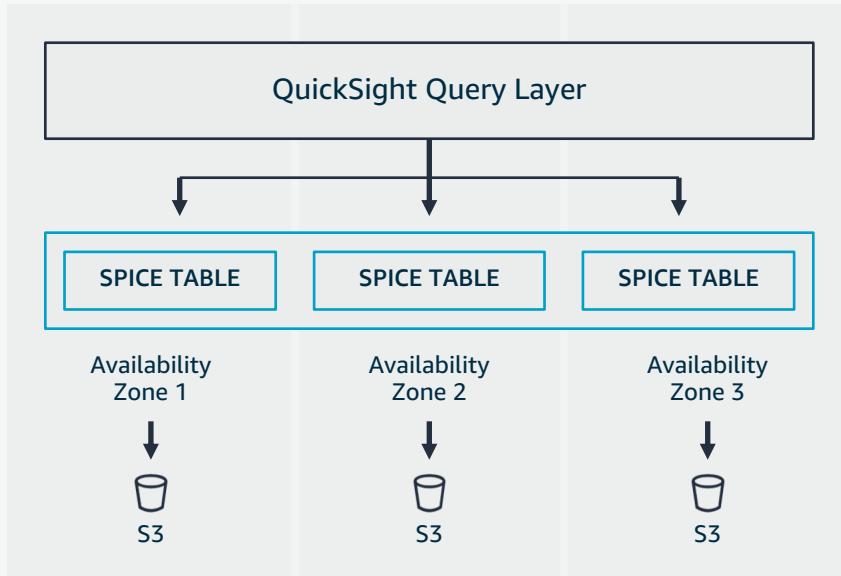
SPICE

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)

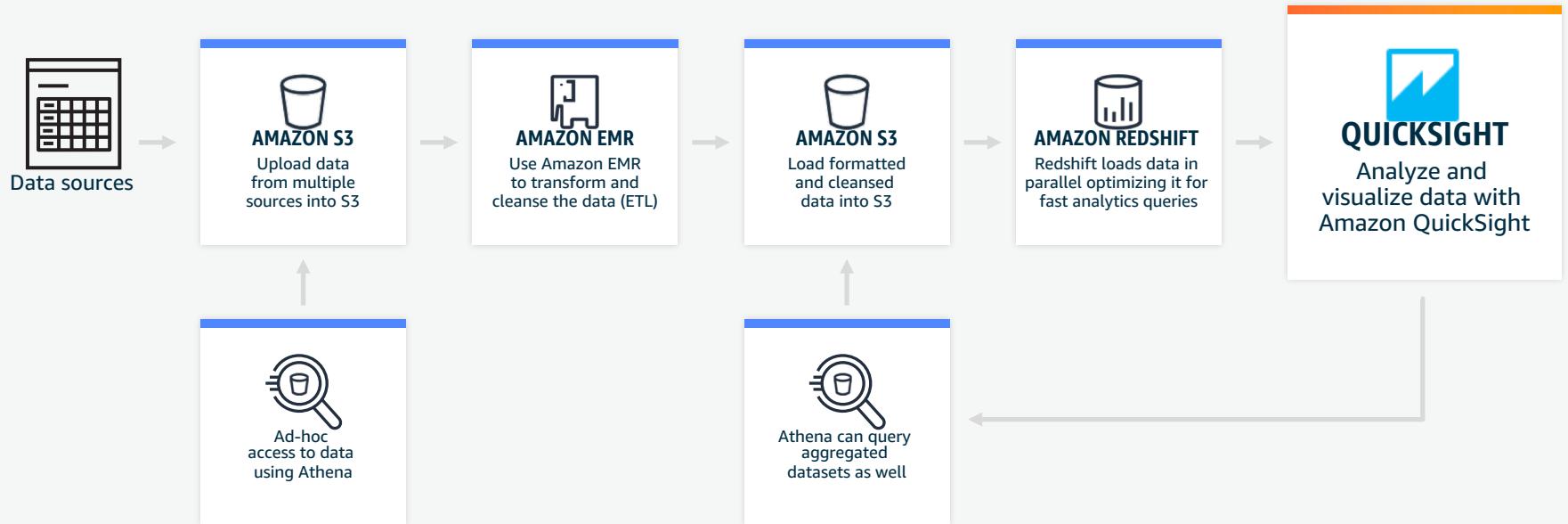
Fault-tolerant, self-healing

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

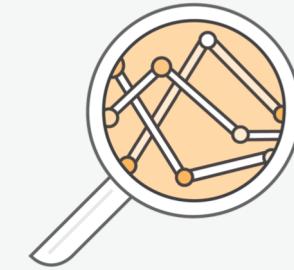
Try it free Today @
Quicksight.AWS

Wrap-up

Feedback survey: <http://bit.ly/2BNeeO7>

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”)

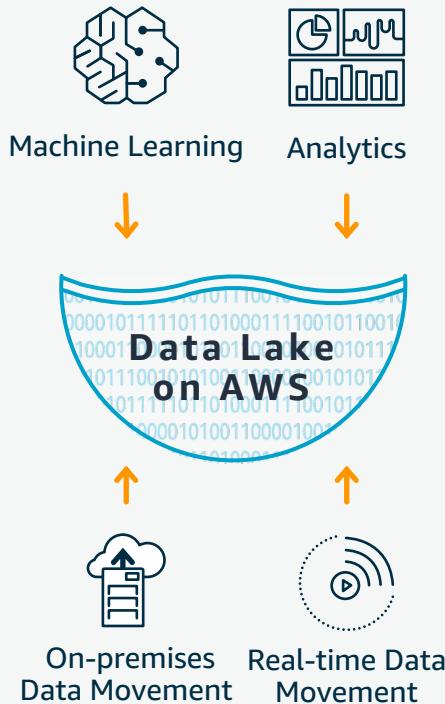
Secure and scalable backbone for all incoming data and all analytics – no more data siloes

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

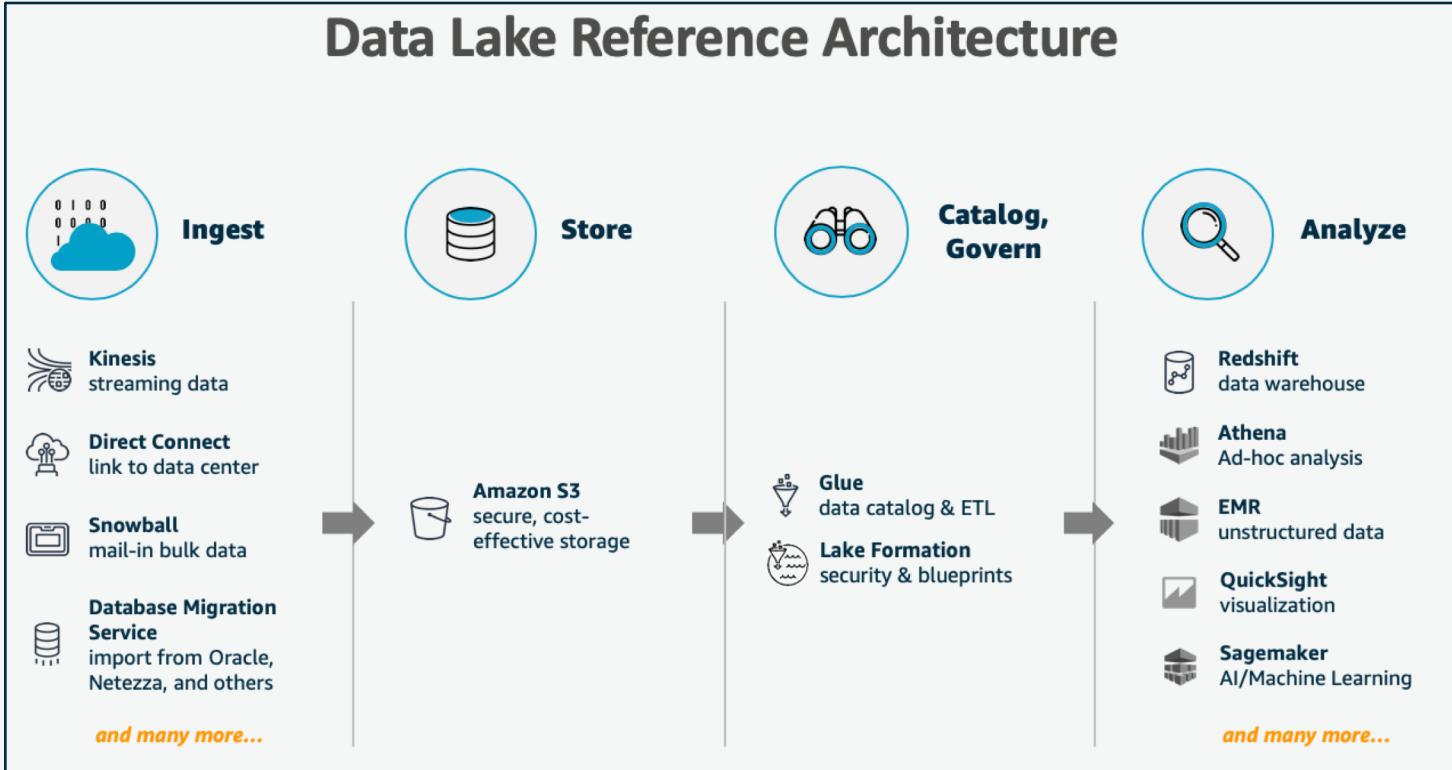


- Open and comprehensive
- Secure
- Scalable and durable
- Lowest cost

More Data Lakes & Analytics on AWS than Anywhere Else



Thank you!



Thank you!

Survey: <http://bit.ly/2BNeeO7>

