

# What's New in AWS Lake Formation?

**Adnan Hasan**

WW GTM Specialist



# What to expect

**01**

**UNDERSTAND**

AWS Lake Formation  
strategy

**02**

**LEARN**

What's new

**03**

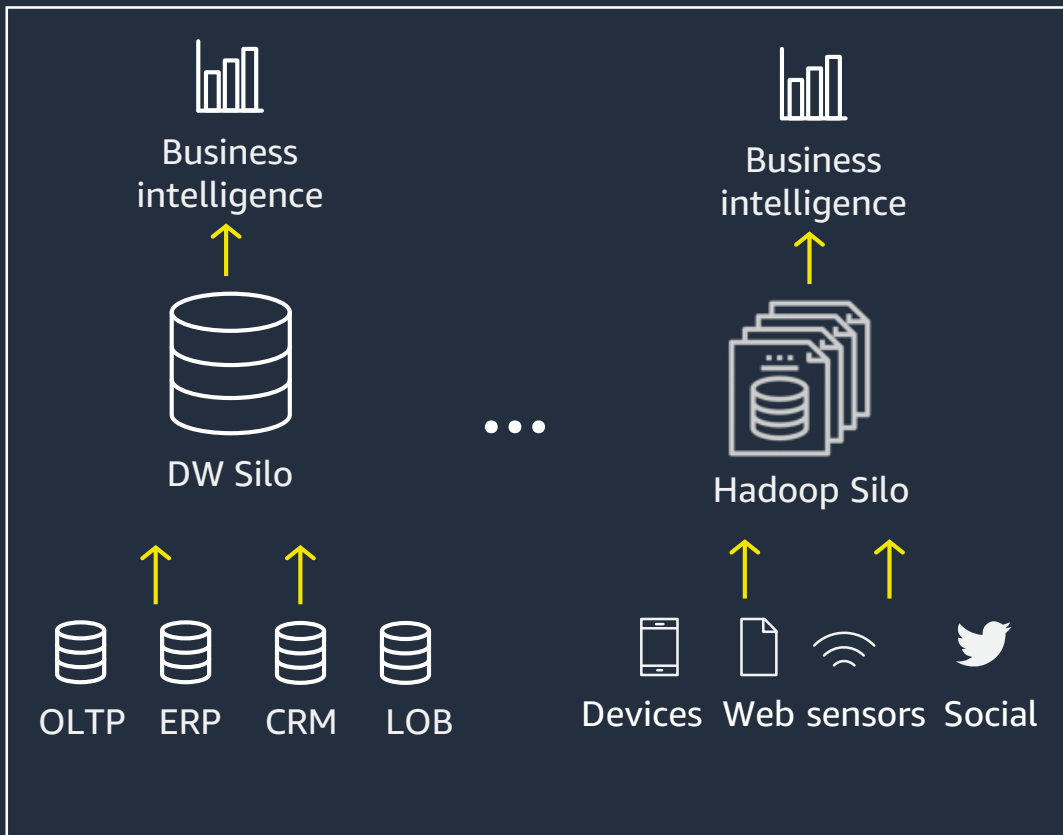
**APPLY**

Get started today



# Customers are moving from...

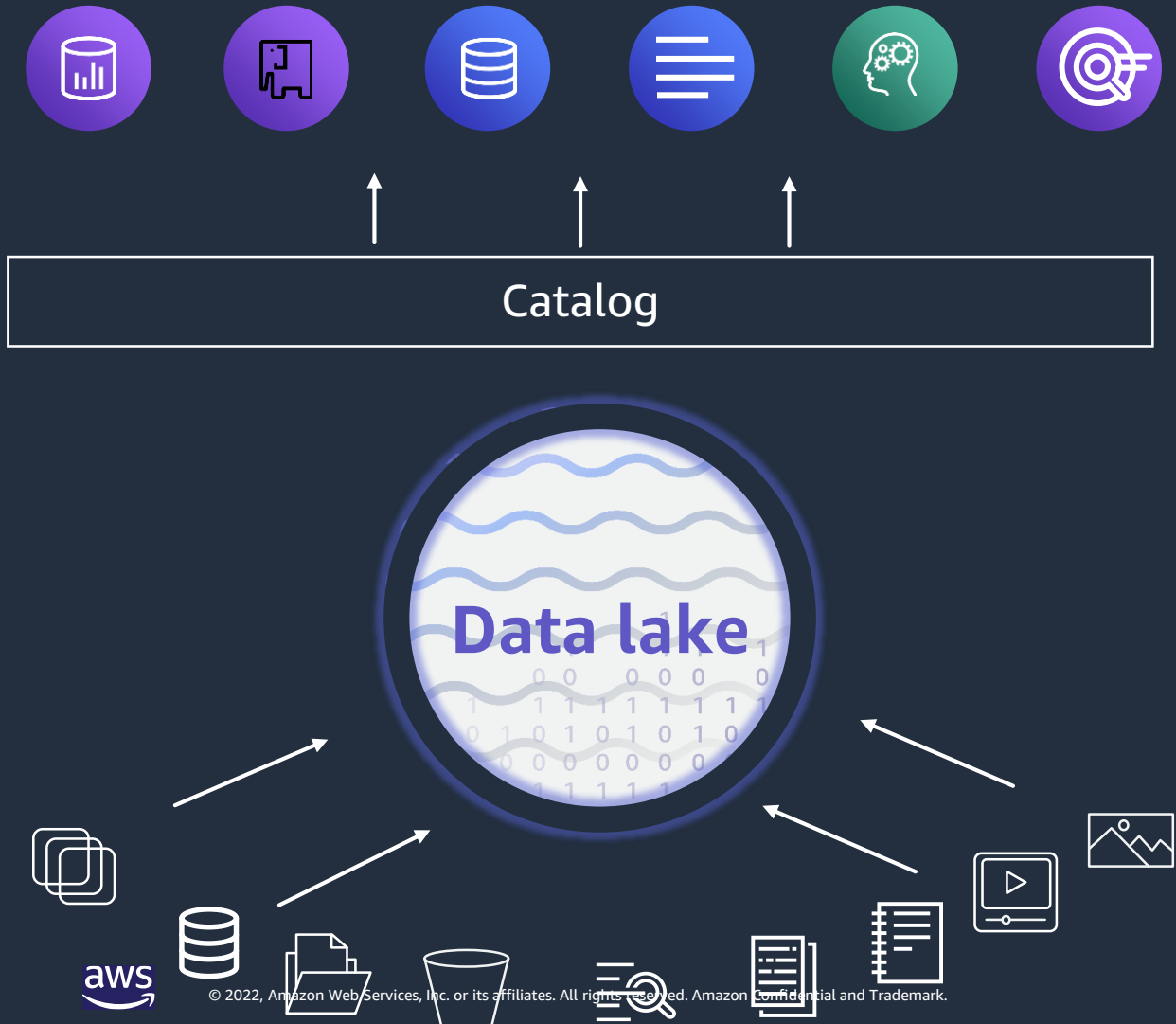
## Siloed data



## Modern data architectures



# What is a Modern data lake..



Store all your data

Cost effectively scale storage to EBs

Decouple storage from compute

Pay-as-you-go analytical and ML engines

Process data in-place

# Amazon S3: Most popular choice to build data lakes



# Common Data Lake Challenges We Hear From Our Customers

"Sharing and searching data is difficult"

"I just want to get access to the data I need"

"Difficult to meet all requirements across differing business units"

"I wish to focus on innovating with data, not on maintaining and administering a data lake"

"My data science team should easily find the datasets they seek and have the ability to share them with others"

"My team needs to own datasets, pipelines and repositories that are isolated from other teams"

"Why doesn't our organization treat data as a product?"

"Current data architecture is complex and monolithic and slow to change"

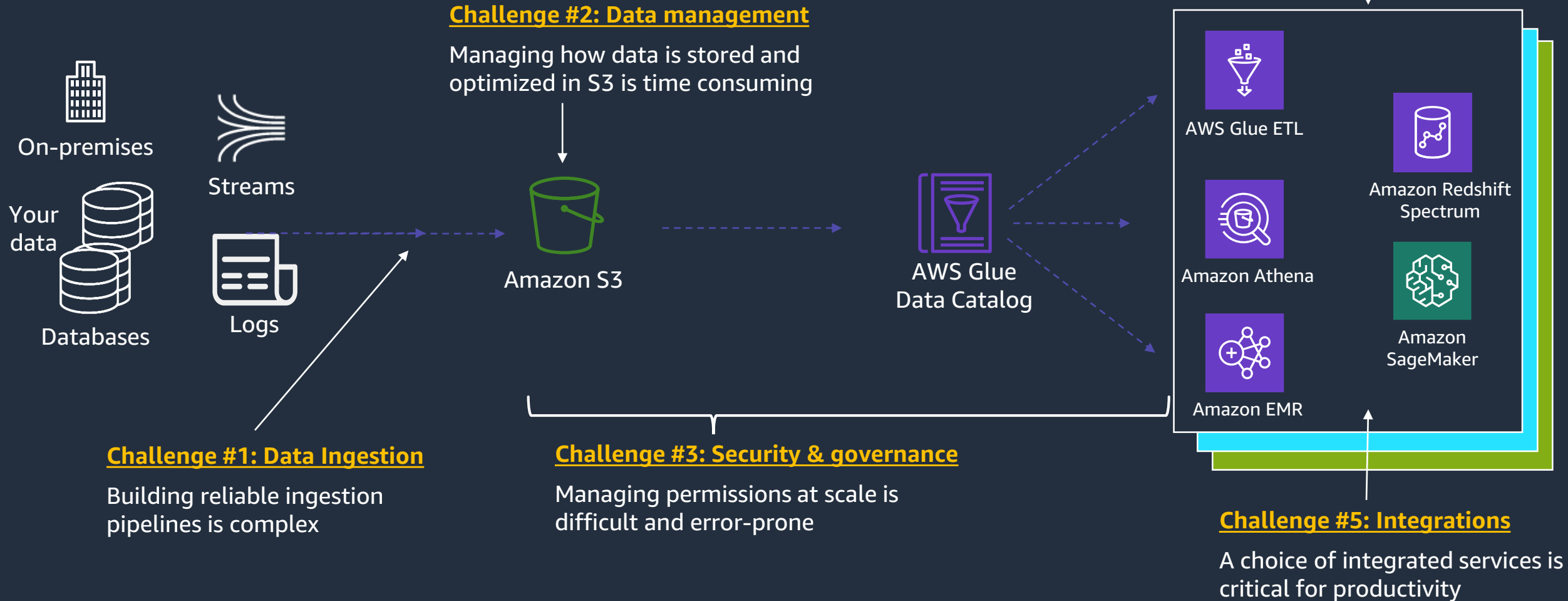
"If I share data, I've lost control"

"There is a mis-match between executive leadership goals and business line deliverables and incentives"

"Our internal policies on what can be shared unclear and there is lack of incentive to share"

"Need to create a model to support sharing from both producers and consumers of data"

# Challenges in building data lakes



# The quest for FLAIR data principles

- F – Findability. The ability to view which data assets are available, access metadata including ownership and data classification, and other mandatory attributes for data governance & compliance.
- L – Lineage. The ability to find data origin, ability to trace data back, understand and visualizing data as it flows from data sources to consumption.
- A – Accessibility. The ability to request a security credential granting an entitlement to access the data asset. Also requires a networking infrastructure to facilitate efficient access.
- I – Interoperability. Data is stored in a format which will be accessible to most, if not all, internal processing systems
- R – Reusability. Data is registered with a known schema, and attribution of the data source is clear. May encompass MDM concepts.

***None of these features require a centralised or co-located storage model***

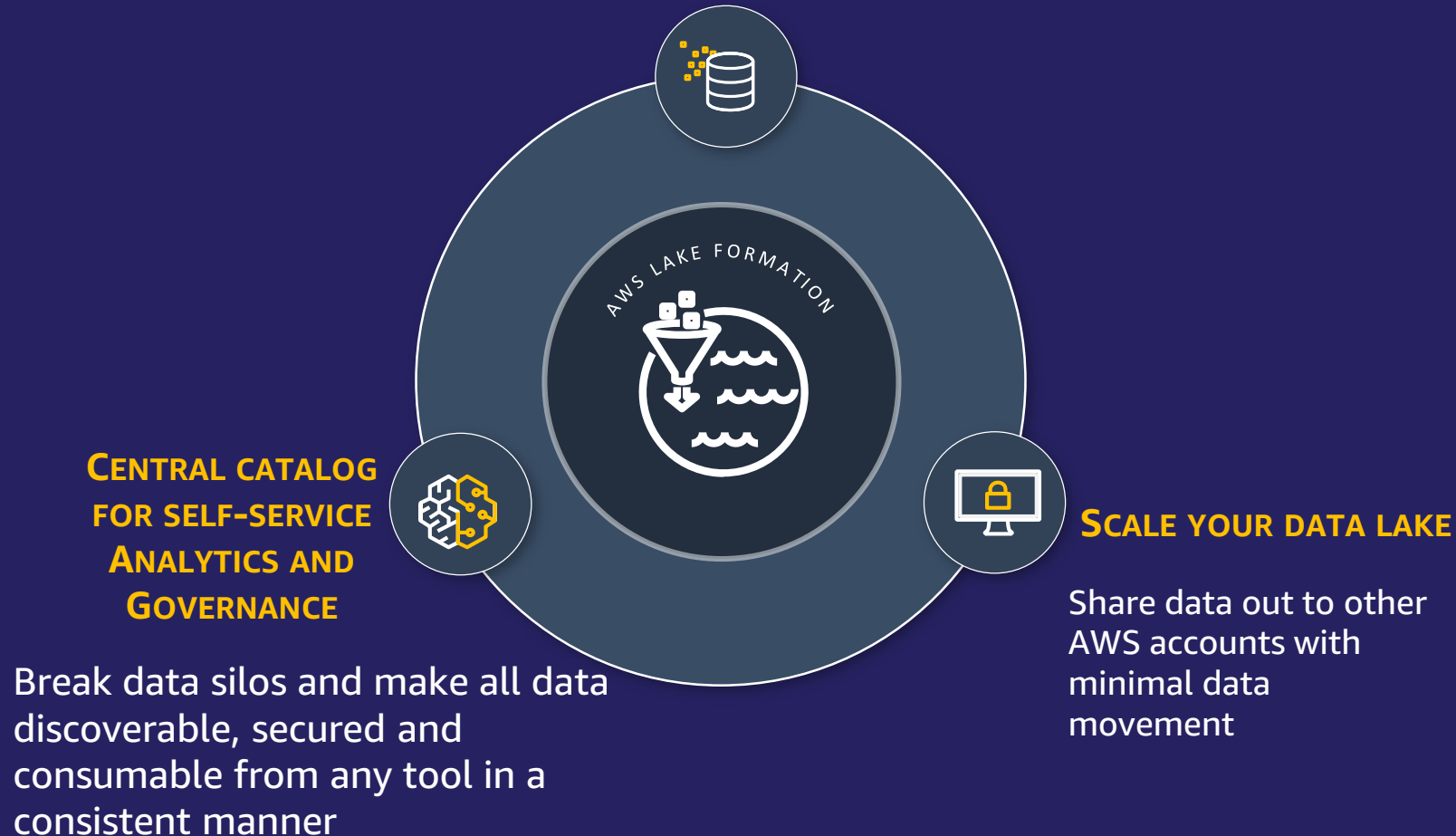


# AWS Lake Formation

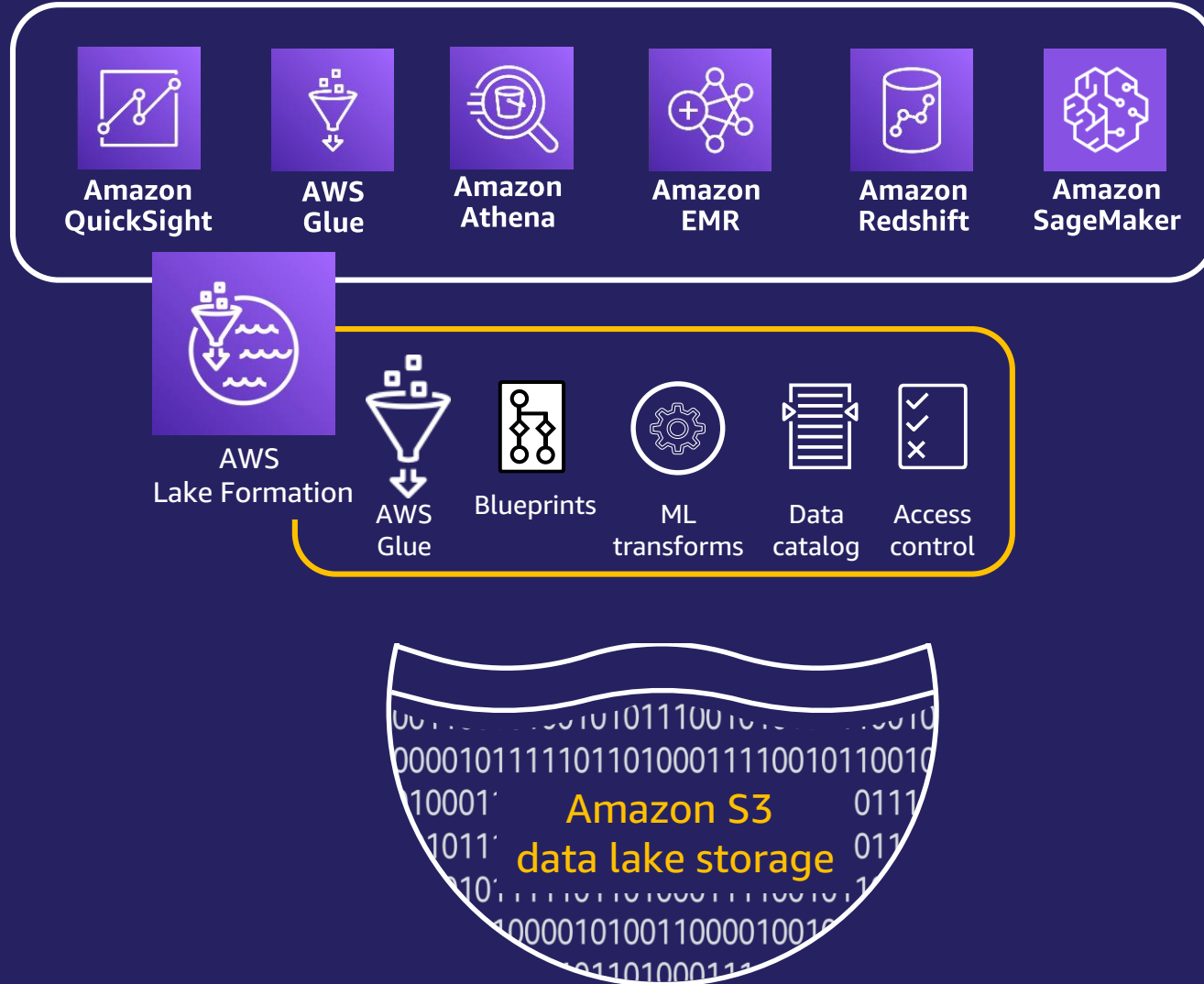
A fully managed serverless service that allows you to **build, clean, secure & operationalize** data lakes in days

## DATABASE-LIKE FEATURES

Simplified ingestion and cleanup with ACID transaction, database-like permissions including row/cell level security, query performance improvement with auto compaction



# AWS Lake Formation: The foundation for data lakes



Single place to manage access

Open file formats

Efficient sharing

Ecosystem of integrated tools

Cost effective



# Thousands of customers use AWS Lake Formation

MELIÀ HOTELS INTERNATIONAL

Santander

CollegeBoard

香港 CI

Clickatell

NU SKIN

FANDUEL

toast

CURVO

Houghton  
Mifflin  
Harcourt

REALOGY

... managing petabytes  
of data

100 SCHOLASTIC

Alcon

State Farm®

GoDaddy

MACROMILL GROUP

agibank®  
Pra você, é agora.

amazon

D2C

BW | Best Western.  
Hotels & Resorts

Onefootball

OSU

MODSY

free

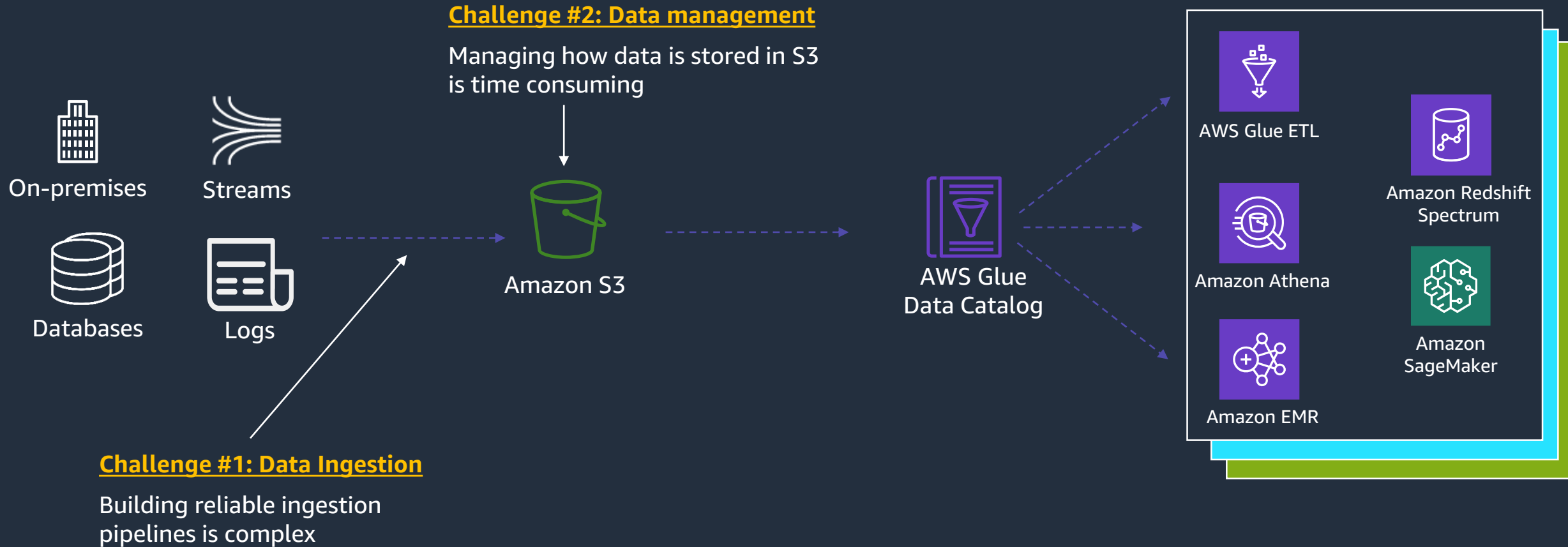
MOIA

Juvo

aws

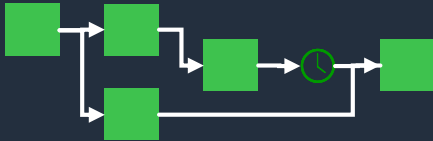
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

# Challenge: Data ingestion & management



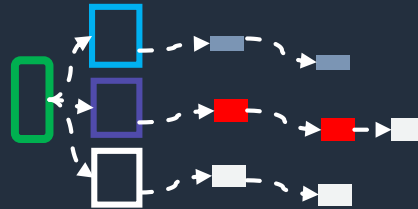
# Why is data ingestion and management hard?

## CONTINUOUS UPDATES



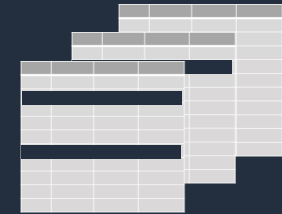
COMPLEX ETL  
DELAYS IN DATA FRESHNESS  
EXPENSIVE, BRITTLE &  
ERROR-PRONE

## INCONSISTENT PERFORMANCE



DATA STORED HOW IT  
ARRIVED  
LOTS OF SMALL FILES  
PARTITION UPDATES

## COMPLYING WITH REGULATIONS



DIFFICULT TO FIND  
NEEDLE IN VERY LARGE  
HAYSTACK

# AWS Lake Formation Governed Tables

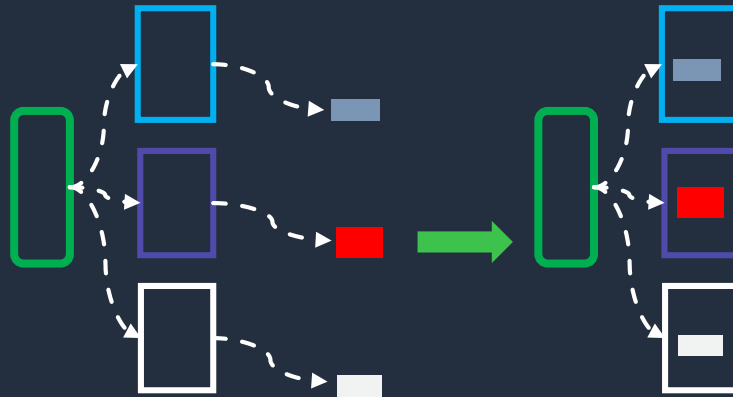
SIMPLIFY DATA INGESTION AND DATA MANAGEMENT



## ACID transactions

Consistent across tasks  
Insert, update, delete  
Converge batch & real-time

Reliable



## Storage optimization

Auto-compact small files  
Push-down filters  
Reduce data scan

Performant



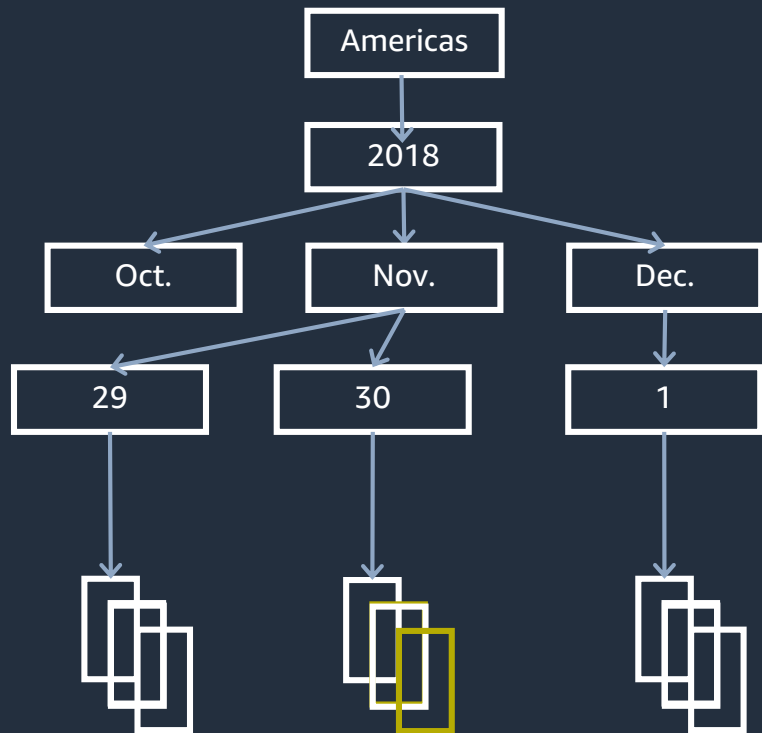
## Time travel

Data history  
Reproduce experiments  
Audit changed data

Versioned

# Governed Tables: Under the hood

*Apache Hive-style tables  
organize data in partitions*



Path

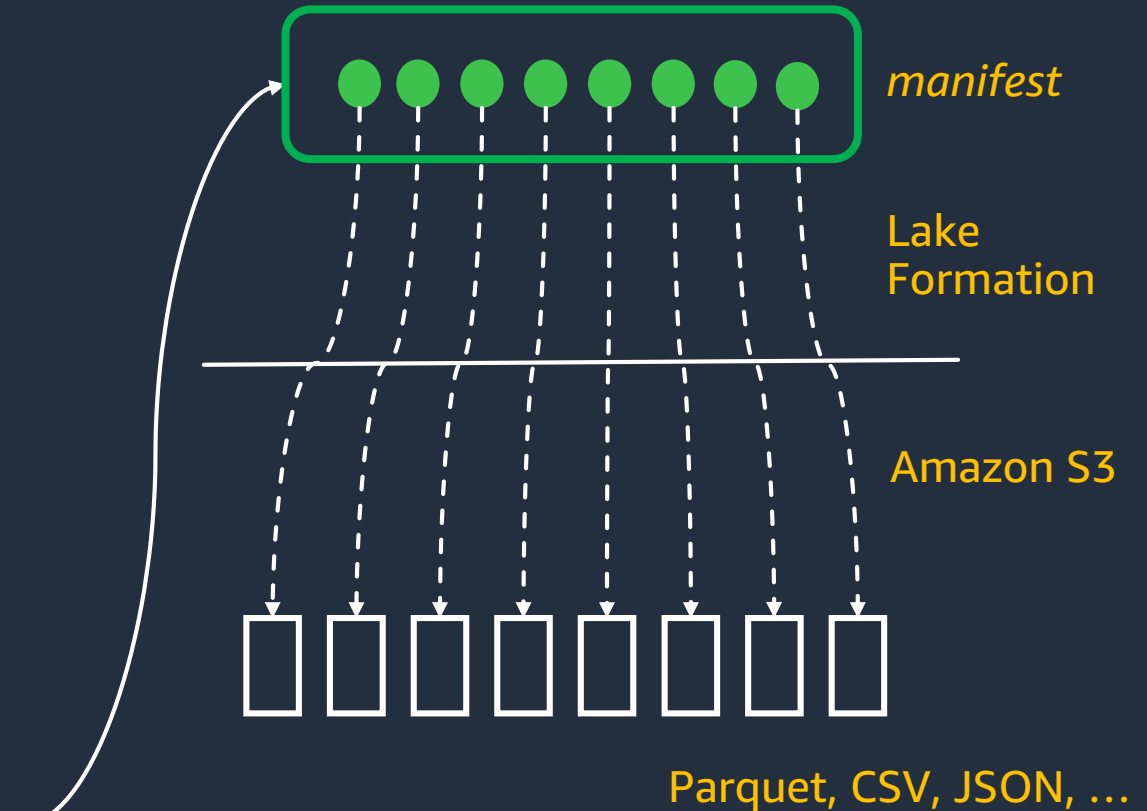
File

S3://IOTDeviceData/region=Americas/year=2021/month=Nov/day=30/data1.csv  
 S3://IOTDeviceData/region=Americas/year=2021/month=Nov/day=30/data2.csv  
 S3://IOTDeviceData/region=Americas/year=2021/month=Nov/day=30/data3.csv



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

## Governed Table

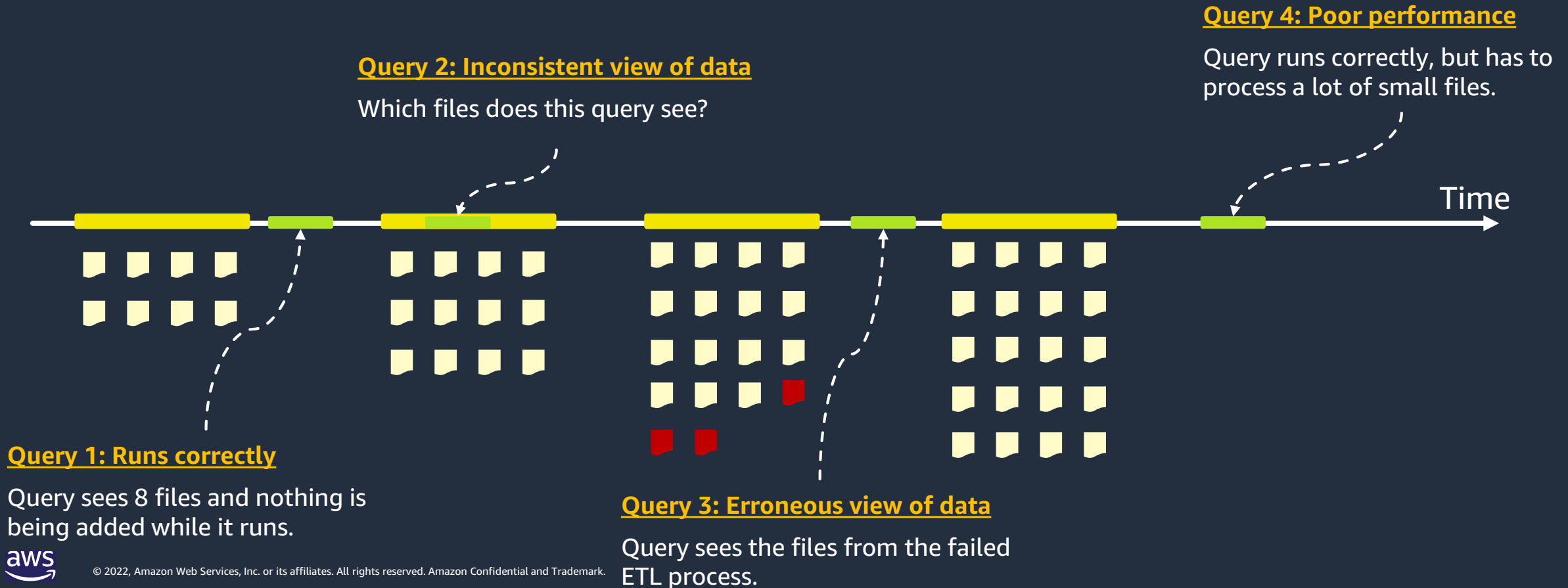


**Import and export: Metadata only**

# Without transactions and storage optimization

Ingest data with **ETL process** that may fail sometimes

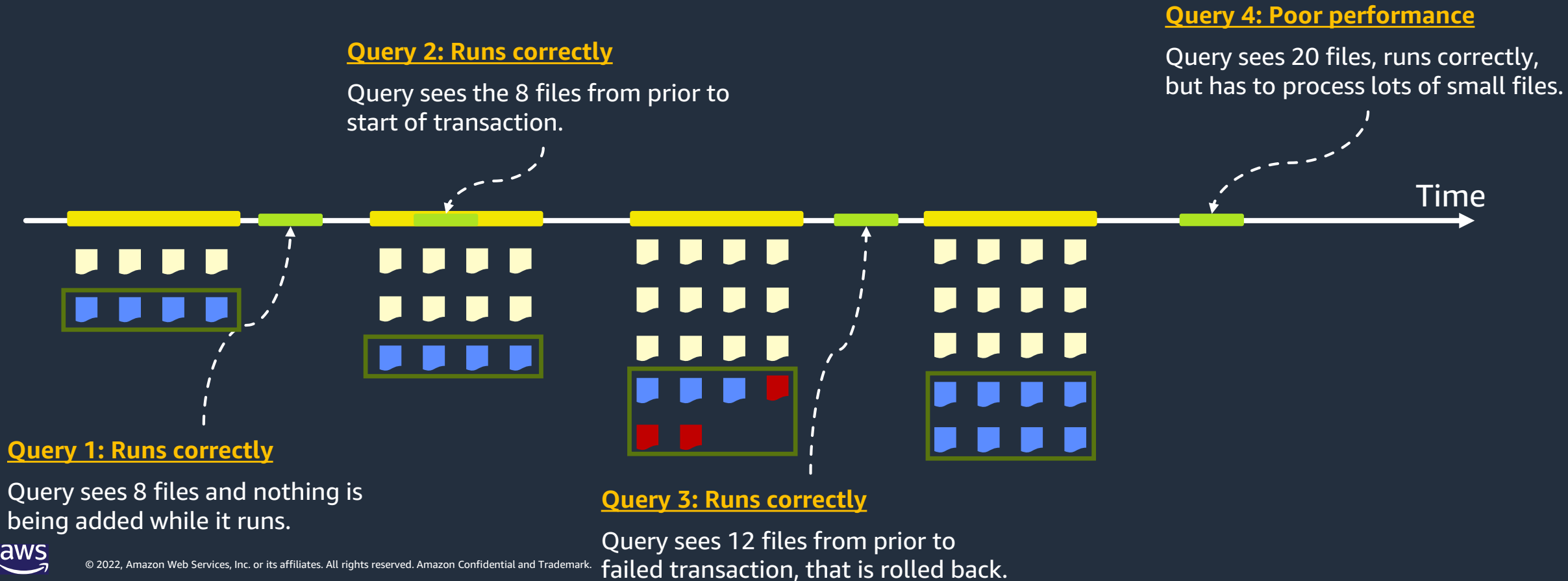
Users run **queries** in an ad-hoc manner





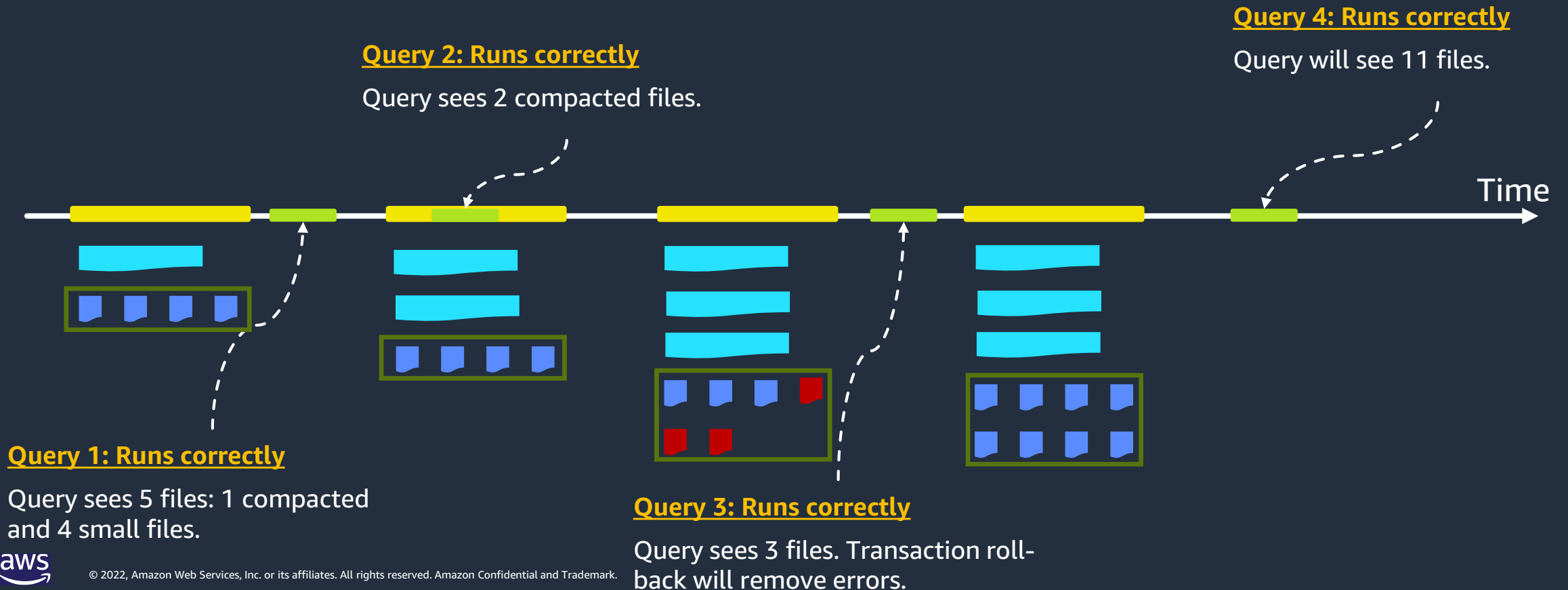
# With transactions

ETL process uses Governed Tables transactions to ingest data and recover from errors



# And, with storage optimization

Governed tables **automatically compact small files** into larger files



# Storage optimizer benefits: File sizes

## S3 tables without compaction

Name	Type	Last modified	Size
<a href="#">run-1605877752823-part-block-0-0-r-00003-snappy-077077191262.f4eaa44652472b8281284f15d2ef0c36bb313i4.parquet</a>	parquet	November 20, 2020, 05:09 (UTC-08:00)	2.9 KB
<a href="#">run-1605880546874-part-block-0-0-r-00003-snappy-077077191262.7d5d9913df33126cc4bec998cb854afa6ecf6a.parquet</a>	parquet	November 20, 2020, 05:55 (UTC-08:00)	2.9 KB
<a href="#">run-1605883429534-part-block-0-0-r-00003-snappy-077077191262.70ee9ea4b8fcd0516dbc66875cb76d2c2ba26b2.parquet</a>	parquet	November 20, 2020, 06:43 (UTC-08:00)	3.1 KB
<a href="#">run-1605886358271-part-block-0-0-r-00003-snappy-077077191262.bd437632fe3d45ec27f2cec22005f126c264aaa.parquet</a>	parquet	November 20, 2020, 07:32 (UTC-08:00)	3.1 KB
<a href="#">run-1605889215078-part-block-0-0-r-00003-snappy-077077191262.38452a0ff4cbf6c92da82af4550bab1f82af1a3.parquet</a>	parquet	November 20, 2020, 08:20 (UTC-08:00)	3.1 KB
<a href="#">run-1605892074557-part-block-0-0-r-00003-snappy-077077191262.d1310b44a6c8c6e1f60e734e54371ab8fc0698b.parquet</a>	parquet	November 20, 2020, 09:07 (UTC-08:00)	3.1 KB
<a href="#">run-1605894952359-part-block-0-0-r-00003-snappy-077077191262.68699c84ef82287d8ee7b18de4850cae8b6cec4.parquet</a>	parquet	November 20, 2020, 09:55 (UTC-08:00)	3.1 KB
<a href="#">run-1605897818406-part-block-0-0-r-00003-snappy-077077191262.9e51396e6c2dbe58d533c6c668831812316e06b.parquet</a>	parquet	November 20, 2020, 10:43 (UTC-08:00)	3.1 KB
<a href="#">run-1605900720793-part-block-0-0-r-00003-snappy-077077191262.c58f9f454dbce049e5092dd551693fe1af3fd5d.parquet</a>	parquet	November 20, 2020, 11:32 (UTC-08:00)	3.1 KB
<a href="#">run-1605903571944-part-block-0-0-r-00003-snappy-077077191262.70ee9ea4b8fcd0516dbc66875cb76d2c2bab2.parquet</a>	parquet	November 20, 2020, 12:19 (UTC-08:00)	3.1 KB

## Governed tables with automatic compaction

Name	Type	Last modified	Size
<a href="#">run-1605877752823-part-block-0-0-r-00012-snappy-077077191262.f4eaa44652472b8281284f15d2ef0c36bb34.parquet</a>	parquet	November 20, 2020, 05:09 (UTC-08:00)	26.7 MB
<a href="#">run-1605880546874-part-block-0-0-r-00012-snappy-077077191262.7d5d9913df33126cc4bec998cb854afa6ecf6a.parquet</a>	parquet	November 20, 2020, 05:56 (UTC-08:00)	29.3 MB
<a href="#">run-1605883429534-part-block-0-0-r-00012-snappy-077077191262.70ee9ea4b8fcd0516dbc66875cb76d2c2bab2.parquet</a>	parquet	November 20, 2020, 06:44 (UTC-08:00)	31.3 MB
<a href="#">run-1605886358271-part-block-0-0-r-00012-snappy-077077191262.bd437632fe3d45ec27f2cec22005f126c264a.parquet</a>	parquet	November 20, 2020, 07:33 (UTC-08:00)	35.9 MB
<a href="#">run-1605889215078-part-block-0-0-r-00012-snappy-077077191262.38452a0ff4cbf6c92da82af4550bab1f82af1a3.parquet</a>	parquet	November 20, 2020, 08:21 (UTC-08:00)	38.1 MB
<a href="#">run-1605892074557-part-block-0-0-r-00012-snappy-077077191262.d1310b44a6c8c6e1f60e734e54371ab8fc08b.parquet</a>	parquet	November 20, 2020, 09:08 (UTC-08:00)	38.7 MB
<a href="#">run-1605894952359-part-block-0-0-r-00012-snappy-077077191262.68699c84ef82287d8ee7b18de4850cae8b6c4.parquet</a>	parquet	November 20, 2020, 09:56 (UTC-08:00)	39.3 MB
<a href="#">run-1605897818406-part-block-0-0-r-00012-snappy-077077191262.9e51396e6c2dbe58d533c6c6688318123106b.parquet</a>	parquet	November 20, 2020, 10:44 (UTC-08:00)	39.8 MB
<a href="#">run-1605900720793-part-block-0-0-r-00012-snappy-077077191262.c58f9f454dbce049e5092dd551693fe1af3fd5d.parquet</a>	parquet	November 20, 2020, 11:32 (UTC-08:00)	40.4 MB
<a href="#">run-1605903571944-part-block-0-0-r-00012-snappy-077077191262.70ee9ea4b8fcd0516dbc66875cb76d2c2bab2.parquet</a>	parquet	November 20, 2020, 12:19 (UTC-08:00)	40.4 MB

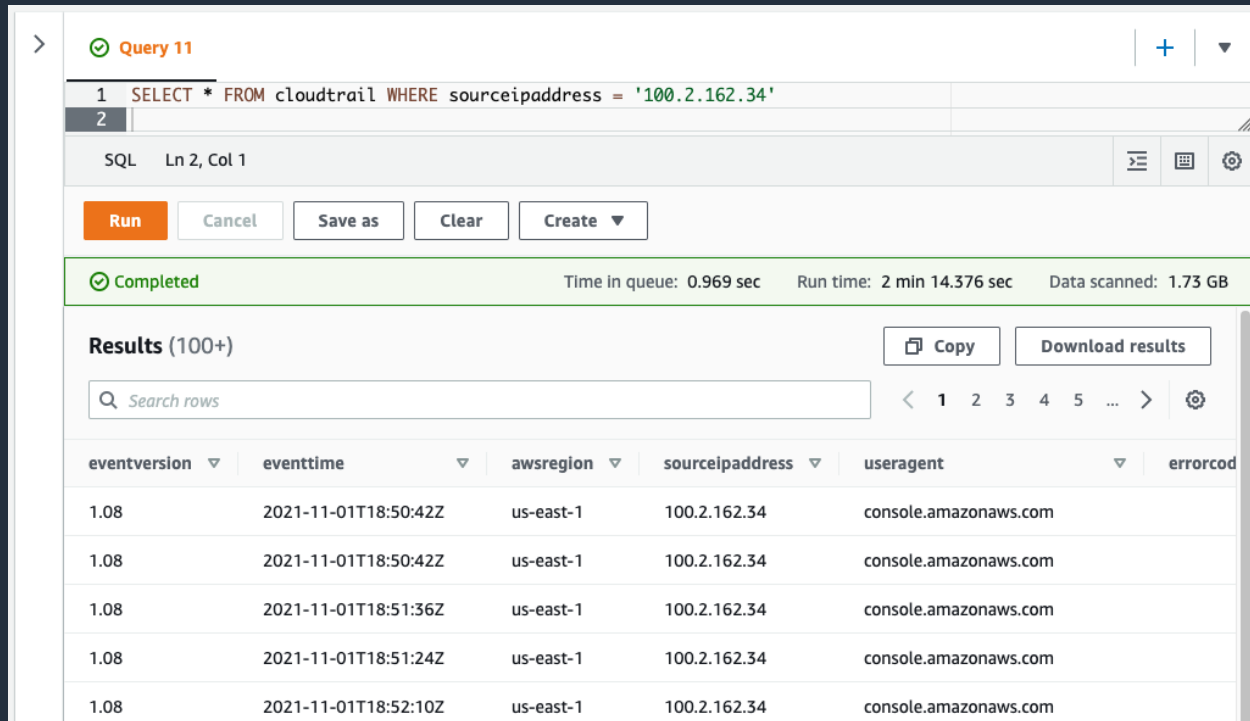
Smaller files means performance limited by I/O requests

Optimally sized for faster query performance



# Storage optimizer benefits: Performance

## Query performance without compaction



Query 11

```
1 SELECT * FROM cloudtrail WHERE sourceipaddress = '100.2.162.34'
```

SQL Ln 2, Col 1

Run Cancel Save as Clear Create

Completed Time in queue: 0.969 sec Run time: 2 min 14.376 sec Data scanned: 1.73 GB

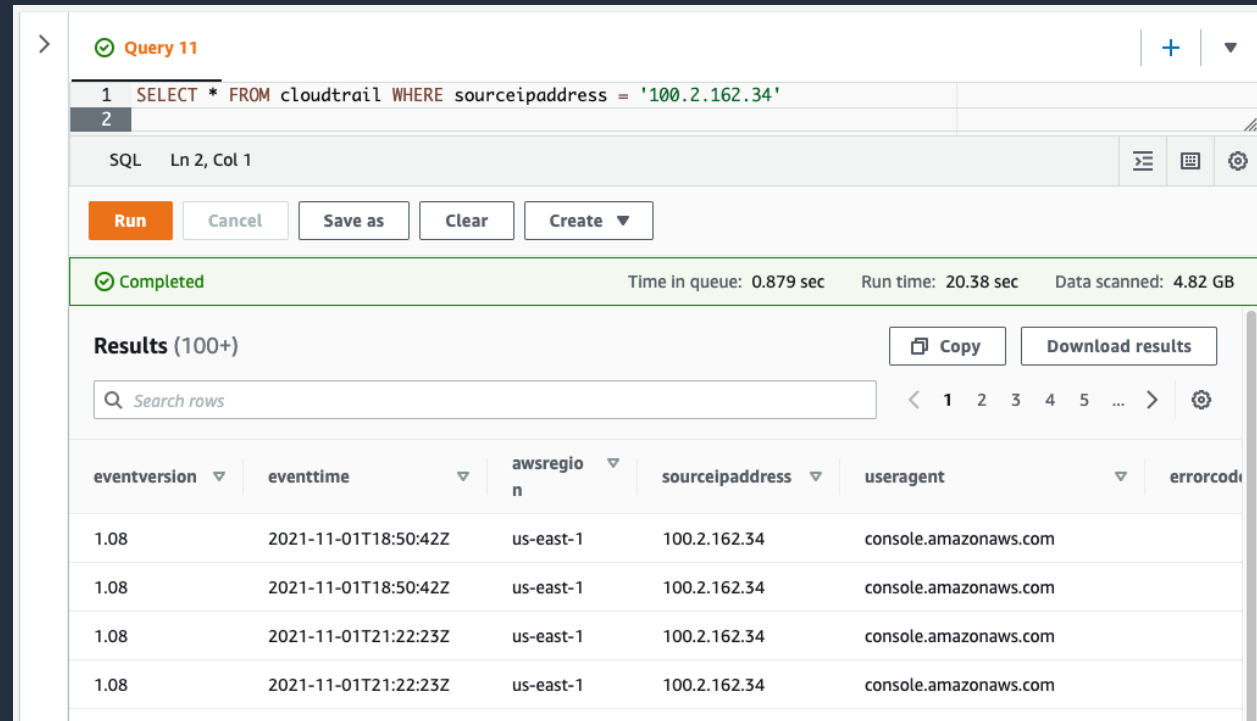
Results (100+) Copy Download results

Search rows

eventversion	eventtime	awsregion	sourceipaddress	useragent	errorcode
1.08	2021-11-01T18:50:42Z	us-east-1	100.2.162.34	console.amazonaws.com	
1.08	2021-11-01T18:50:42Z	us-east-1	100.2.162.34	console.amazonaws.com	
1.08	2021-11-01T18:51:36Z	us-east-1	100.2.162.34	console.amazonaws.com	
1.08	2021-11-01T18:51:24Z	us-east-1	100.2.162.34	console.amazonaws.com	
1.08	2021-11-01T18:52:10Z	us-east-1	100.2.162.34	console.amazonaws.com	

Run time: 2 min 14.376 sec

## Query performance with automatic compaction



Query 11

```
1 SELECT * FROM cloudtrail WHERE sourceipaddress = '100.2.162.34'
```

SQL Ln 2, Col 1

Run Cancel Save as Clear Create

Completed Time in queue: 0.879 sec Run time: 20.38 sec Data scanned: 4.82 GB

Results (100+) Copy Download results

Search rows

eventversion	eventtime	awsregion	sourceipaddress	useragent	errorcode
1.08	2021-11-01T18:50:42Z	us-east-1	100.2.162.34	console.amazonaws.com	
1.08	2021-11-01T18:50:42Z	us-east-1	100.2.162.34	console.amazonaws.com	
1.08	2021-11-01T18:50:42Z	us-east-1	100.2.162.34	console.amazonaws.com	
1.08	2021-11-01T21:22:23Z	us-east-1	100.2.162.34	console.amazonaws.com	
1.08	2021-11-01T21:22:23Z	us-east-1	100.2.162.34	console.amazonaws.com	

Run time: 20.38 sec

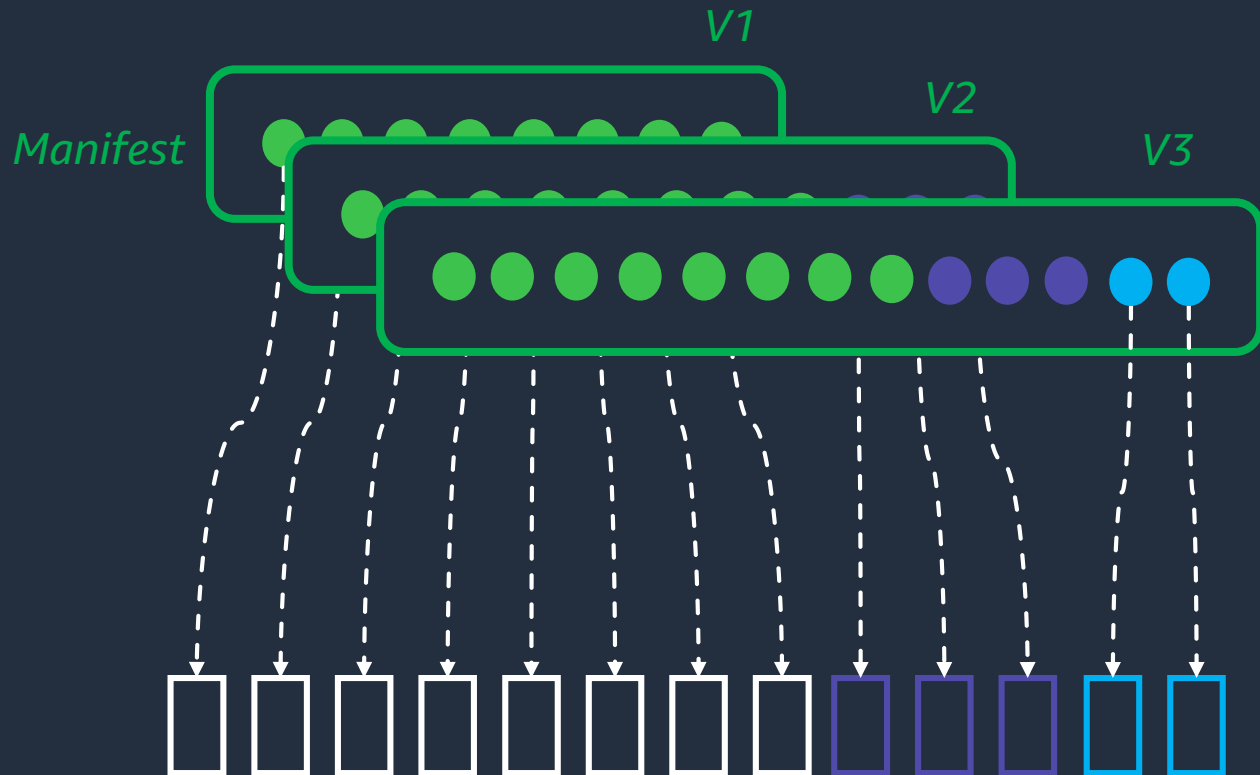
Smaller files means performance limited by I/O requests

Optimally sized for faster query performance

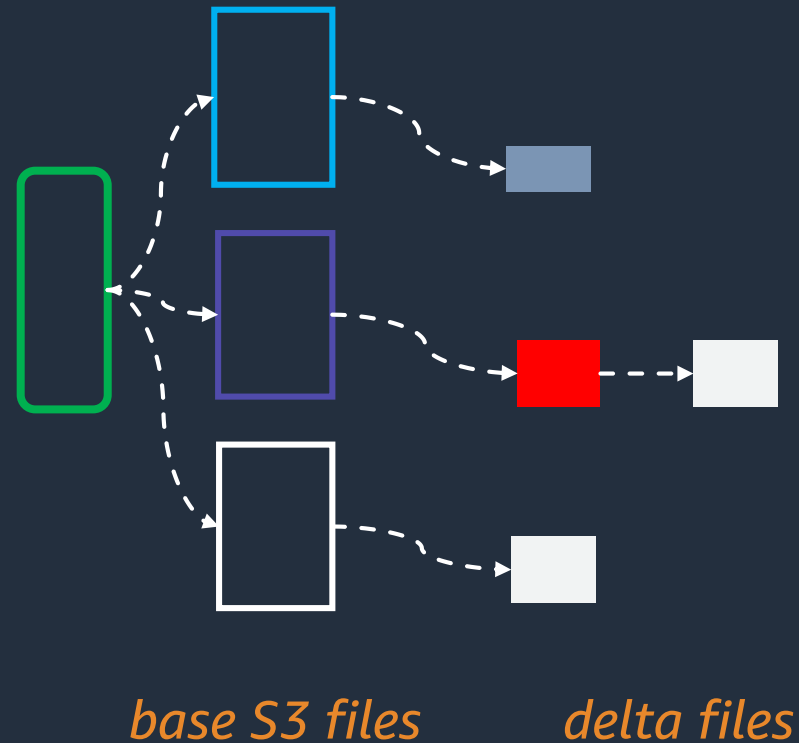


# Governed Tables: Versions and deltas

Time travel to previous version is instant



Manifest-based txn API:  
Add or remove files



Row-based txn API:  
Insert, update, delete rows  
delta files contain edits

# Example: Time travel

Query 11

```
1 SELECT eventtime, eventsource, eventid, useragent
2 FROM cloudtrail
3 WHERE sourceipaddress = '100.2.162.34'
4 ORDER BY eventtime DESC LIMIT 10
```

SQL Ln 5, Col 1

Run Cancel Save as Clear Create

Completed Time in queue: 0.894 sec Run time: 37.137 sec Data scanned: 3.29 GB

Results (10) Copy Download results

Search rows

eventtime	eventsource	eventid
2021-11-04T03:55:24Z	athena.amazonaws.com	0f4846ad-ae59-4765-b267-6be5fc2c41d9
2021-11-04T03:55:04Z	athena.amazonaws.com	60ea1097-c6c0-448a-b618-d3ef3ade0473
2021-11-04T03:54:39Z	athena.amazonaws.com	8bb55fe7-2af7-4f77-89b8-33cc8f3a7e67
2021-11-04T03:47:39Z	kinesis.amazonaws.com	572b1b46-f280-4d68-b764-74f0ac2f05ef

Query 1: Result of query now

Query 11

```
1 SELECT eventtime, eventsource, eventid, useragent
2 FROM cloudtrail FOR SYSTEM_TIME AS OF TIMESTAMP '2021-10-30 18:30:42Z'
3 WHERE sourceipaddress = '100.2.162.34'
4 ORDER BY eventtime DESC LIMIT 10
```

SQL Ln 5, Col 1

Run Cancel Save as Clear Create

Completed Time in queue: 1.03 sec Run time: 29.554 sec Data scanned: 52.70 MB

Results (10) Copy Download results

Search rows

eventtime	eventsource	eventid
2021-10-30T18:29:51Z	athena.amazonaws.com	f8e059e1-4f35-4096-b967-b408fcfad1c0
2021-10-30T18:29:36Z	lakeformation.amazonaws.com	b8470e6a-3857-4ebf-8dd7-3fc96aa8cb5b
2021-10-30T18:29:16Z	glue.amazonaws.com	c4f61b0f-c828-485c-bb24-e14fba825a96
2021-10-30T18:29:16Z	glue.amazonaws.com	5a452019-3392-4cac-88f7-3eb810a7e1ed

Query 2: Result of query "as of" last week



# Transactions simplify development...



*“... Transactional ETL processes are an important part of how we ensure data integrity and ... required additional development time and complexity. We’re excited about AWS Lake Formation Transactions’ ability to simplify our ETL and reduce the overall effort needed to produce trustworthy data in our data lake.”*

Rob Hruska  
Engineering Director  
Hudl

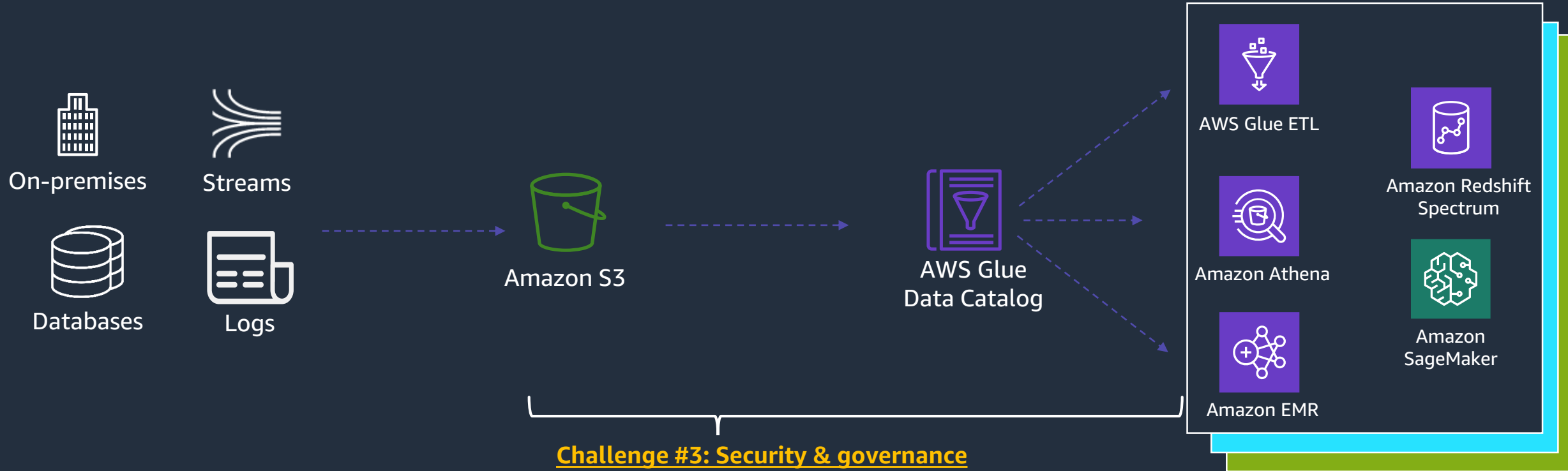


“PowerBuy decided to forego traditional database-based architecture in favor of a data lake using AWS Lake Formation Governed Tables. Governed Tables make it easy to insert, update and delete data for all of our PowerBuy products using highly scalable ACID transactions. With Governed Tables we can release new products quicker like PowerBuy AI and PowerBuy Dashboard without worrying about scaling...”

Thu Truong  
CTO and Co-Founder  
PowerBuy



# Challenge: Security and governance

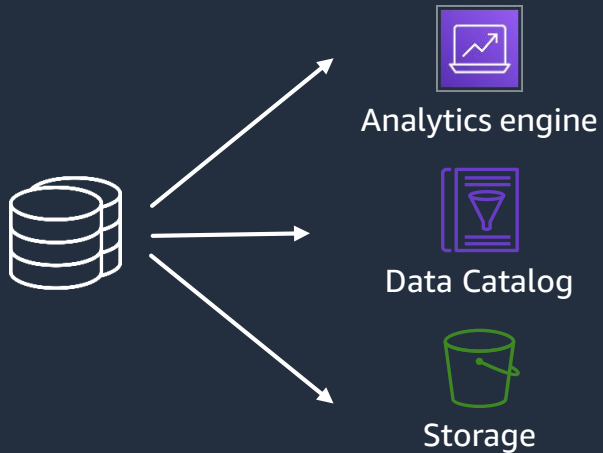


Managing unified fine-grained permissions at scale is error-prone.



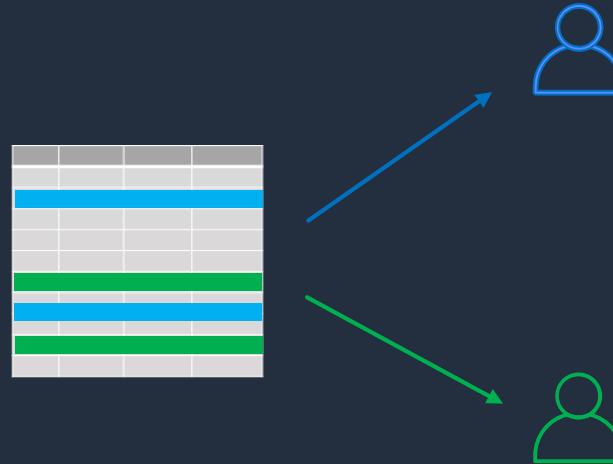
# Why is securing data lakes hard?

## Unifying permissions across the data lake stack



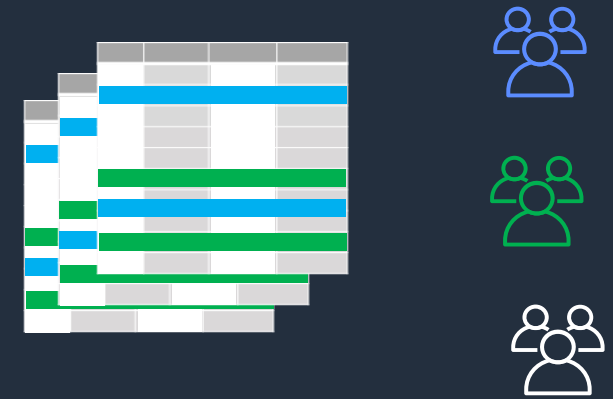
Split storage, metadata, & compute  
Each system has different permissions  
Syncing permissions is error-prone

## Enforcing fine-grained permissions to restrict access



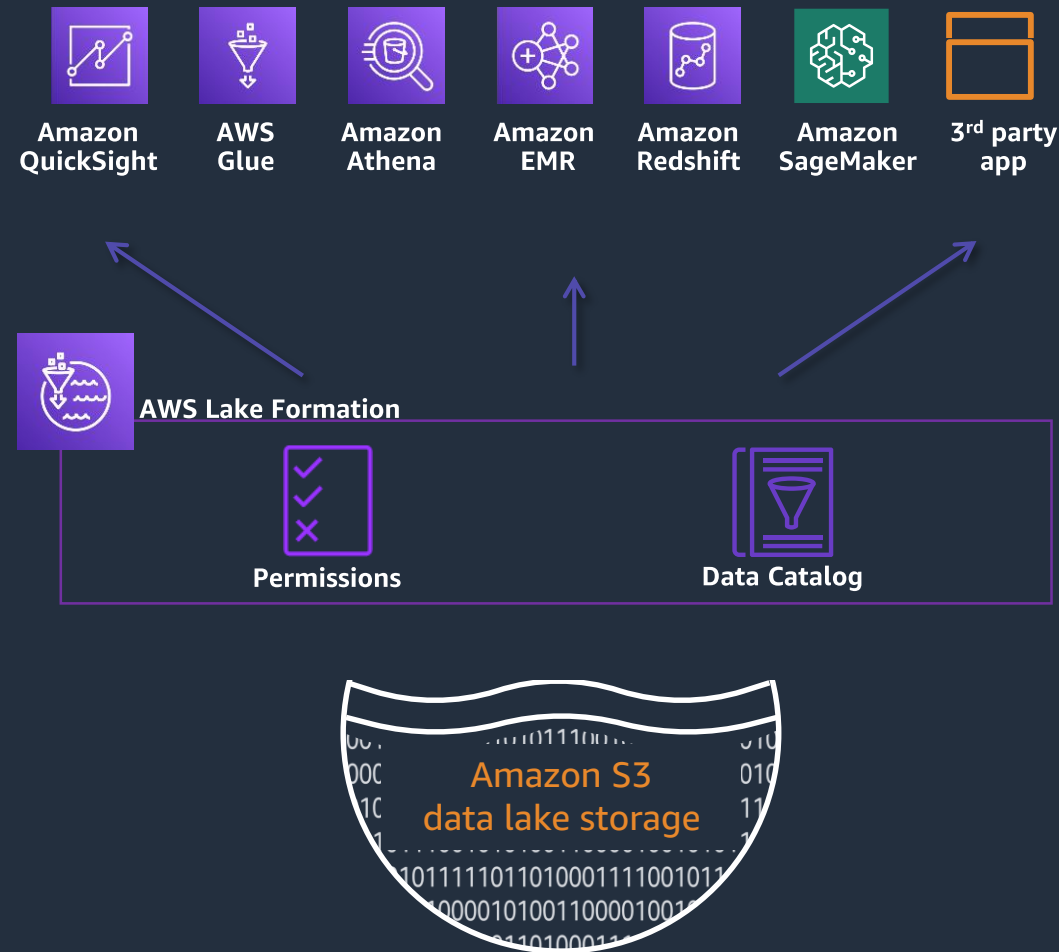
Data lakes contain a lot of data  
Users should only access portions  
Slice and dice the data into portions

## Scalable permissions to manage data and users



1,000s of DBs and tables  
10,000s of users  
New data sets added constantly

# AWS Lake Formation permissions model



DB style fine-grained permissions

Fine-grained permissions on catalog resources

S3 access managed by permission on resources

LF-Tag based access control (LF-TBAC) to scale

Integrated with services and tools

Easy to audit permissions and access

# AWS Lake Formation permissions on:

Columns

Blue	Blue	Blue	Blue	Blue
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray

Specify include or exclude list

Rows

Blue	Blue	Blue	Blue	Blue
Gray	Gray	Gray	Gray	Gray
Blue	Blue	Blue	Blue	Blue
Blue	Blue	Blue	Blue	Blue
Blue	Blue	Blue	Blue	Blue
Gray	Gray	Gray	Gray	Gray
Gray	Gray	Gray	Gray	Gray
Blue	Blue	Blue	Blue	Blue
Gray	Gray	Gray	Gray	Gray
Blue	Blue	Blue	Blue	Blue
Blue	Blue	Blue	Blue	Blue
Gray	Gray	Gray	Gray	Gray

Specify row filter with PartiQL

Cells

Blue	Blue	Blue	Blue	Blue
Gray	Gray	Gray	Gray	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Gray	Gray	Gray	Gray
Gray	Gray	Gray	Gray	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Gray	Gray	Gray	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Blue	Gray	Blue	Gray
Gray	Gray	Gray	Gray	Gray

Combine column and row filters

# Setting up permissions is simple

## Principals

- ☒ **IAM users and roles**  
Users or roles from this AWS account.
- ☐ **SAML users and groups**  
SAML users and group or QuickSight ARNs.
- ☐ **External accounts**  
AWS accounts or AWS organizations outside of this account.

### IAM users and roles

Add one or more IAM users or roles.

Choose IAM principals to add

BusinessUser X  
Role

DataAnalyst2 X  
User

Specify the principal:

Select IAM users and roles

Bring your own SAML users and groups

Specify the resource / columns

Select the right level of permissions

## Databases

Select one or more databases.

Choose databases

cloudtrail X  
106567286946

Load more

### Tables - optional

Select one or more tables.

Choose tables

cloudtrail\_logs\_awslogs X  
106567286946

## Table permissions

### Table permissions

Choose specific access permissions to grant.

- ☒ Select
- ☐ Insert
- ☐ Delete
- ☒ Describe
- ☒ Alter
- ☒ Drop

☐ Super

This permission is the union of all the individual permissions to the left, and supersedes them.

## Data permissions

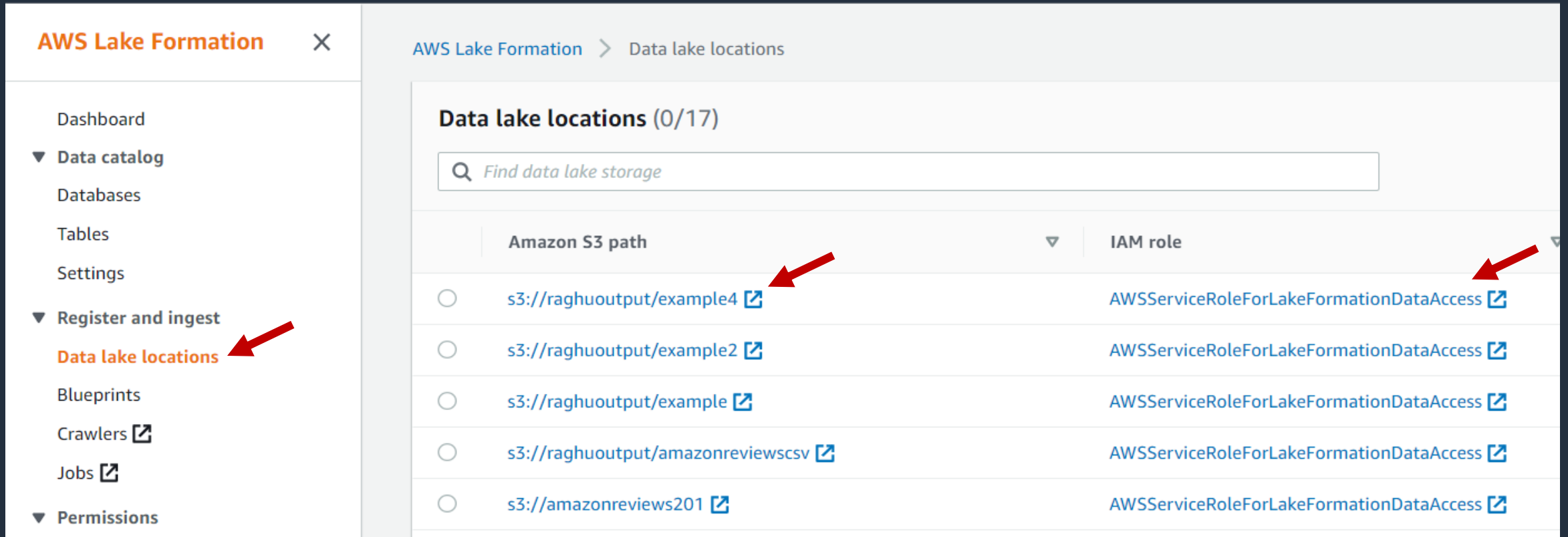
☒ **All data access**  
Grant access to all data without any restrictions.

☐ **Column-based access**  
Grant data access to specific columns only.



# AWS Lake Formation: Managing S3 access

AWS Lake Formation can vend credentials for **registered locations**

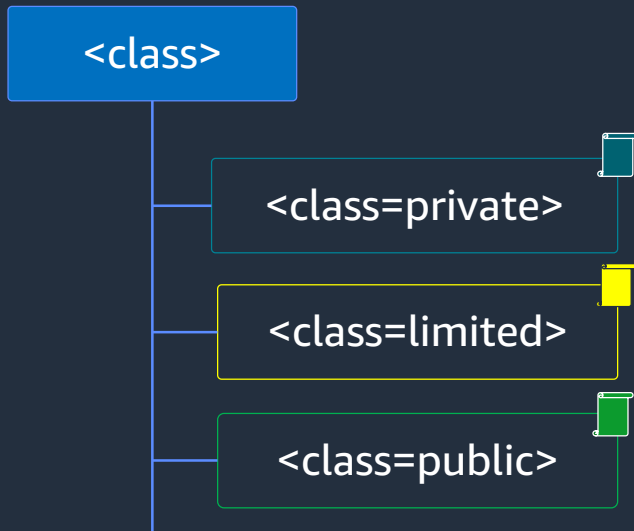


The screenshot displays the AWS Lake Formation console interface. On the left, a navigation sidebar includes links to Dashboard, Data catalog, Databases, Tables, Settings, Register and ingest, Data lake locations (highlighted with a red arrow), Blueprints, Crawlers, Jobs, and Permissions. The main content area is titled 'Data lake locations (0/17)' and features a search bar with the placeholder text 'Find data lake storage'. Below the search bar is a table with two columns: 'Amazon S3 path' and 'IAM role'. The table lists five registered locations, each with a radio button, an S3 path, and an IAM role. Red arrows point to the first row's S3 path and IAM role. The IAM role for all locations is 'AWSServiceRoleForLakeFormationDataAccess'.

	Amazon S3 path	IAM role
<input type="radio"/>	s3://raghuoutput/example4 <a href="#">↗</a>	AWSServiceRoleForLakeFormationDataAccess <a href="#">↗</a>
<input type="radio"/>	s3://raghuoutput/example2 <a href="#">↗</a>	AWSServiceRoleForLakeFormationDataAccess <a href="#">↗</a>
<input type="radio"/>	s3://raghuoutput/example <a href="#">↗</a>	AWSServiceRoleForLakeFormationDataAccess <a href="#">↗</a>
<input type="radio"/>	s3://raghuoutput/amazonreviewscsv <a href="#">↗</a>	AWSServiceRoleForLakeFormationDataAccess <a href="#">↗</a>
<input type="radio"/>	s3://amazonreviews201 <a href="#">↗</a>	AWSServiceRoleForLakeFormationDataAccess <a href="#">↗</a>

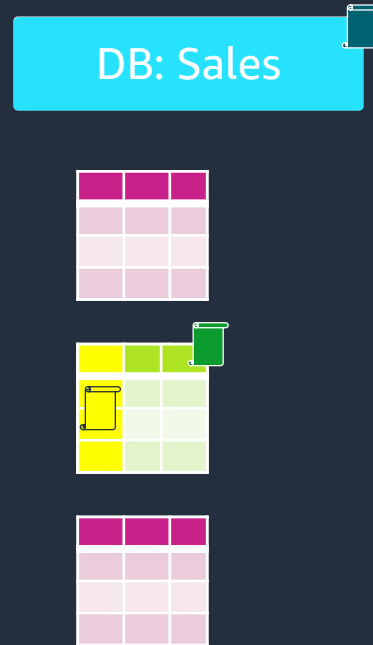
# LF – Tag Based Access Control

## Define LF-Tags



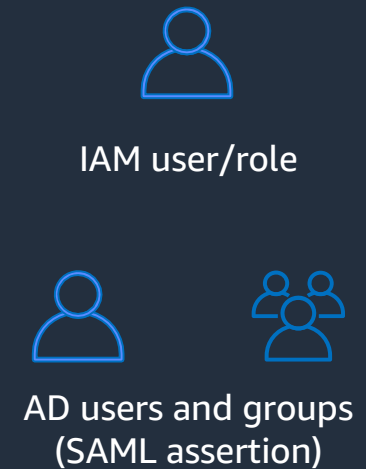
Specify who can assign LF-Tags and values

## Assign LF-Tags to resources



Tag databases, tables, columns  
LF-Tags are hierarchical and may be overridden

## Create policies on LF-Tags



Scale by applying permission on LF-Tags

# Example: Using LF-TBAC

**Target principals**  
Select the principals or accounts to grant permissions to.

☒ **IAM users and roles**  
Users or roles from this AWS account.

☐ **SAML users and groups**  
SAML users and group or QuickSight ARNs.

☐ **External accounts**  
AWS accounts or AWS organizations outside of this account.

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

Choose IAM principals to add ▼

dataAdmin X  
User

**LF-Tags or catalog resources**

☒ **Resources matched by LF-Tags (recommended)**  
Manage permissions indirectly for resources or data matched by a specific set of LF-Tags.

☐ **Named data catalog resources**  
Manager permissions for specific databases or tables, in addition to fine-grained data access.

**Key** **Values**

Q Confidentiality X Choose LF-tag values ▼ Remove

Confidential X

Q Departments X Choose LF-tag values ▼ Remove

Marketing X

Add LF-Tag

Specify the principal  
Specify the tag expression  
Specify the row-level filter  
PartiQL expression

**Database permissions**

Database permissions  
Choose specific access permissions to grant.

☒ Create table ☐ Alter ☐ Drop ☐ Super

☒ Describe

This permission is the union of all the individual permissions to the left, and supersedes them.

**Table permissions**

Table permissions  
Choose specific access permissions to grant.

☒ Select ☒ Insert ☐ Delete ☐ Super

☐ Describe ☐ Alter ☐ Drop

This permission is the union of all the individual permissions to the left, and supersedes them.

Cancel Grant

# Securing data lakes...



*"We found AWS Lake Formation easy to use to build and secure our data lake. Without AWS Lake Formation, we would have to make constant access policy updates to Amazon S3 when we added more users and data . . . With the adoption of AWS Lake Formation, we are able to . . . reduce Amazon S3 policy edits by over 90% . . ."*

Hisatoshi Imaoka  
Tech. Lead Data Infrastructure  
freee K.K.



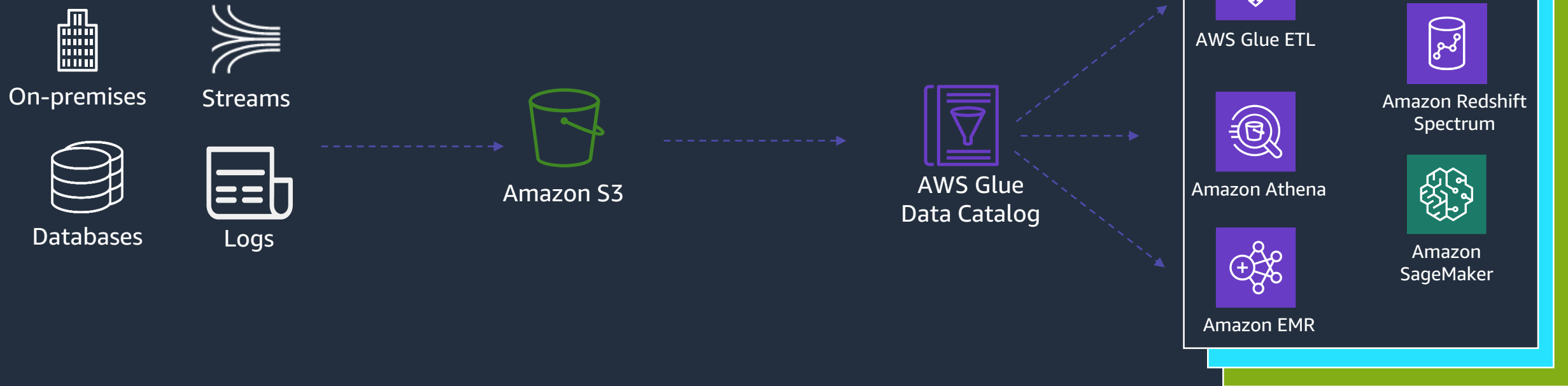
*"AWS Lake Formation enables us to create a secure Data Lake with fine-grained controls on our user's personal information. Our Analysts can now deliver much needed insights, much faster without compromising the governance and security policies."*

Damian Grech  
Data Engineering, Sr. Manager  
FanDuel





# Challenge: Data sharing

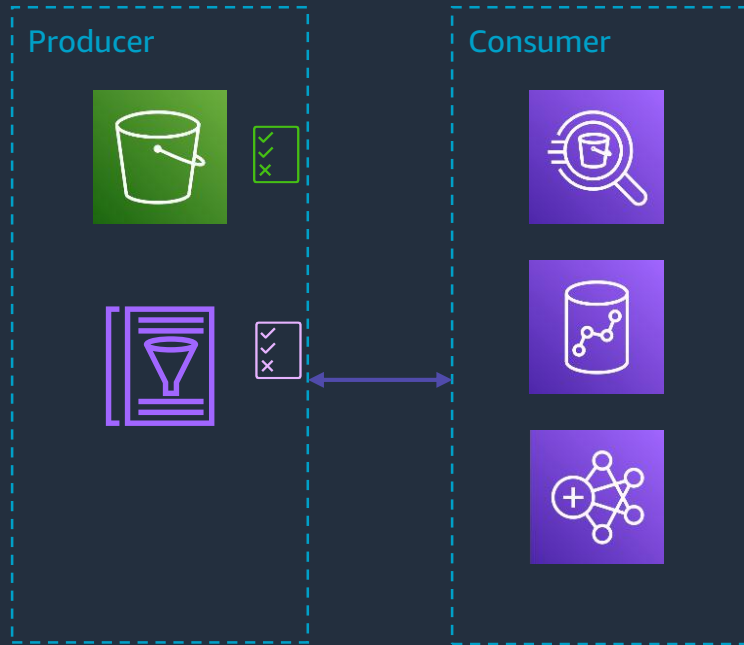


## Challenge #4: Data sharing

Sharing across accounts and organizations is cumbersome.

# Why is sharing data across accounts hard?

## To share data...



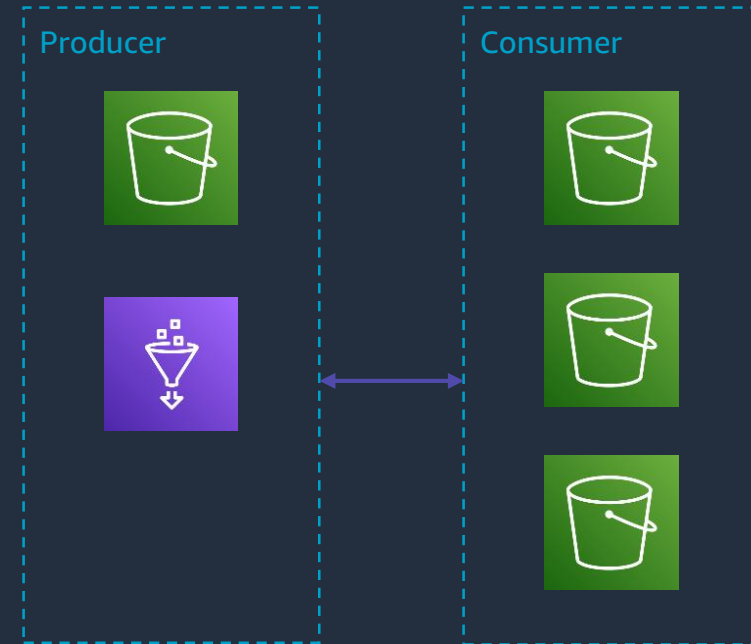
**S3 and IAM policies**

**Limited by service support**

**Lacks discoverability**

**Policy size limits (coarse grained)**

## Duplicating data

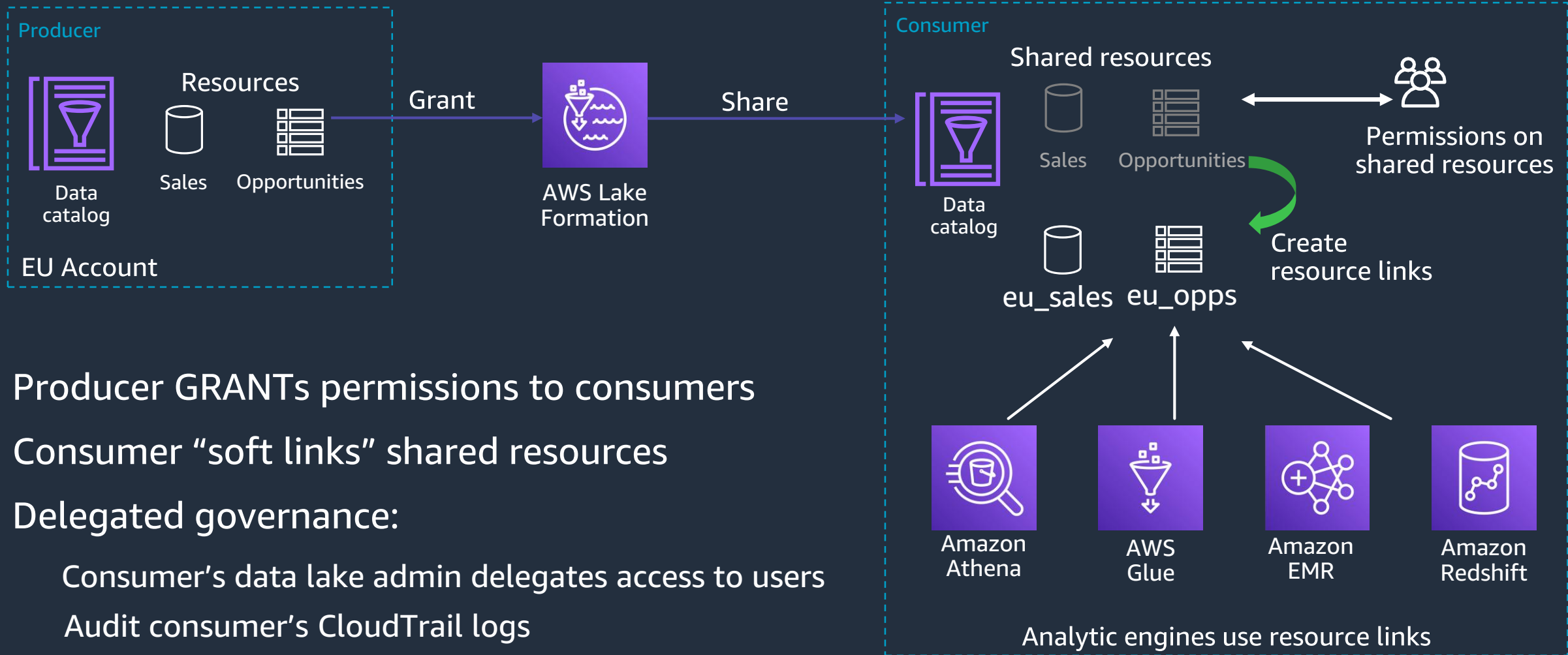


**ETL pipelines**

**Multiple redacted copies**

**Expensive, brittle, and error-prone**

# AWS Lake Formation cross-account sharing



Producer GRANTs permissions to consumers

Consumer “soft links” shared resources

Delegated governance:

- Consumer’s data lake admin delegates access to users

- Audit consumer’s CloudTrail logs

Optional centralized auditing with event forwarding



# Sharing data lakes across accounts



*"... We are building a **hub-and-spoke architecture** using AWS Lake Formation where data producers can **publish** their sharable data to a centralized data catalog and data subscribers can **request access** to that data from the centralized data catalog."*

Charles Beadnall  
CTO  
GoDaddy



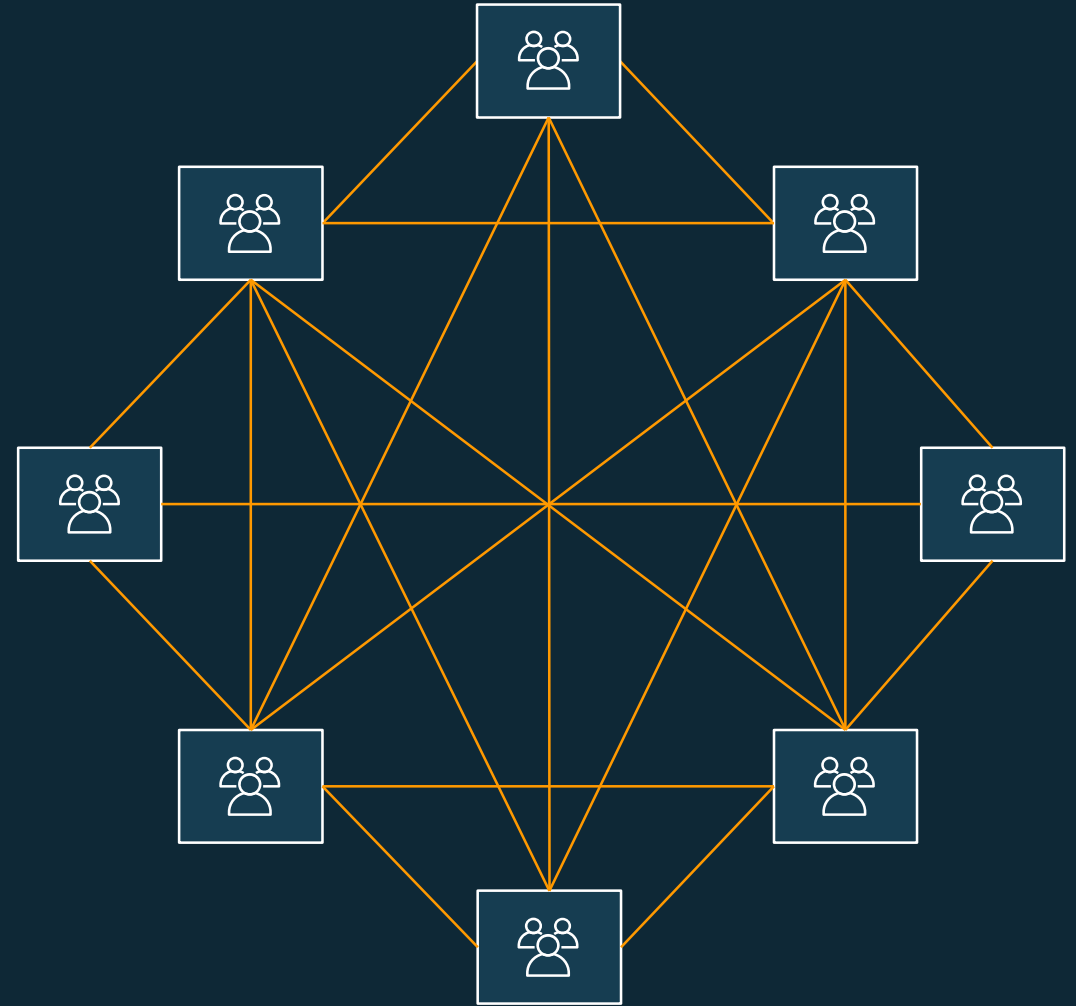
*"... To work on the data lake with **multiple AWS accounts**, we need a central metadata catalog ... Using the **cross account database/table sharing** of **AWS Lake Formation**, we are able to **achieve** our **goal easily**, without affecting the existing workload ... the ability to manage data by **column enables more sophisticated data management**."*

Rinichiro Nagatomo  
Technical lead  
Data management platform  
D2C Inc.



# Why Data Mesh?

- Encourage data-driven **agility**
- Support domain-local **governance** through **lightweight** centralized policy
- Isolate data resources with clear **accountability**
- Consider data a **product** which can exist in **any system**



# Data Mesh Organisational Principles

Allow data producers to **create the data model** and contract for the data they own

Provide efficient means to safely **find, publish, and securely exchange** data

Create a **single model for identity** & data classification

Build **logging, monitoring, auditing** into this environment

**Mandate** this system's use for data sharing using resource share

**Set organisational goals** on adoption, and create an inspection mechanism

**Publicise success** of data sharing between teams to create a entitlement policy

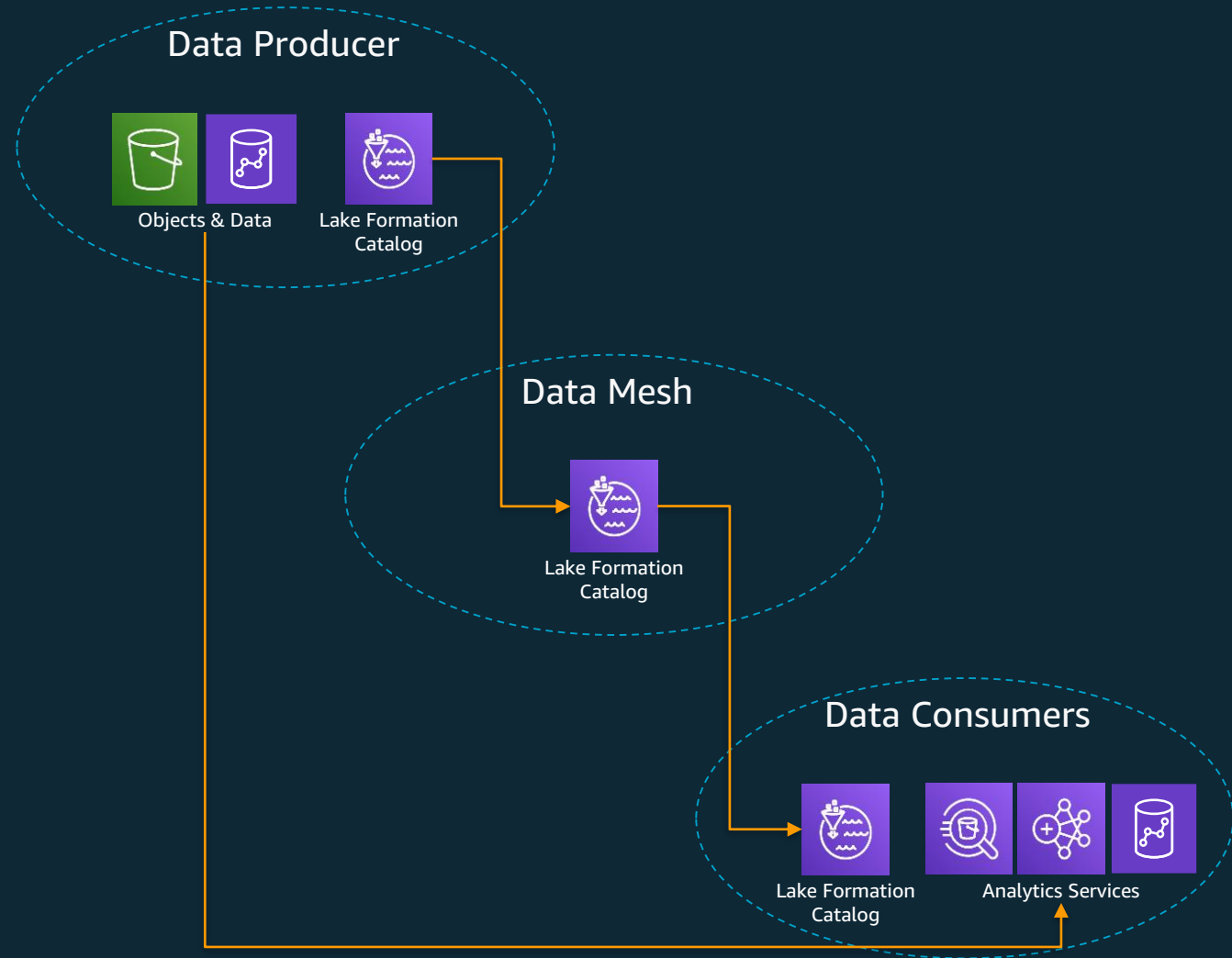
# Data Mesh - Core Concepts

A Data Mesh features **Data Domains** as nodes, which exist in data lake accounts

A Data Producer contributes one or more **Data Products** to a central catalog in a **Data Mesh account**

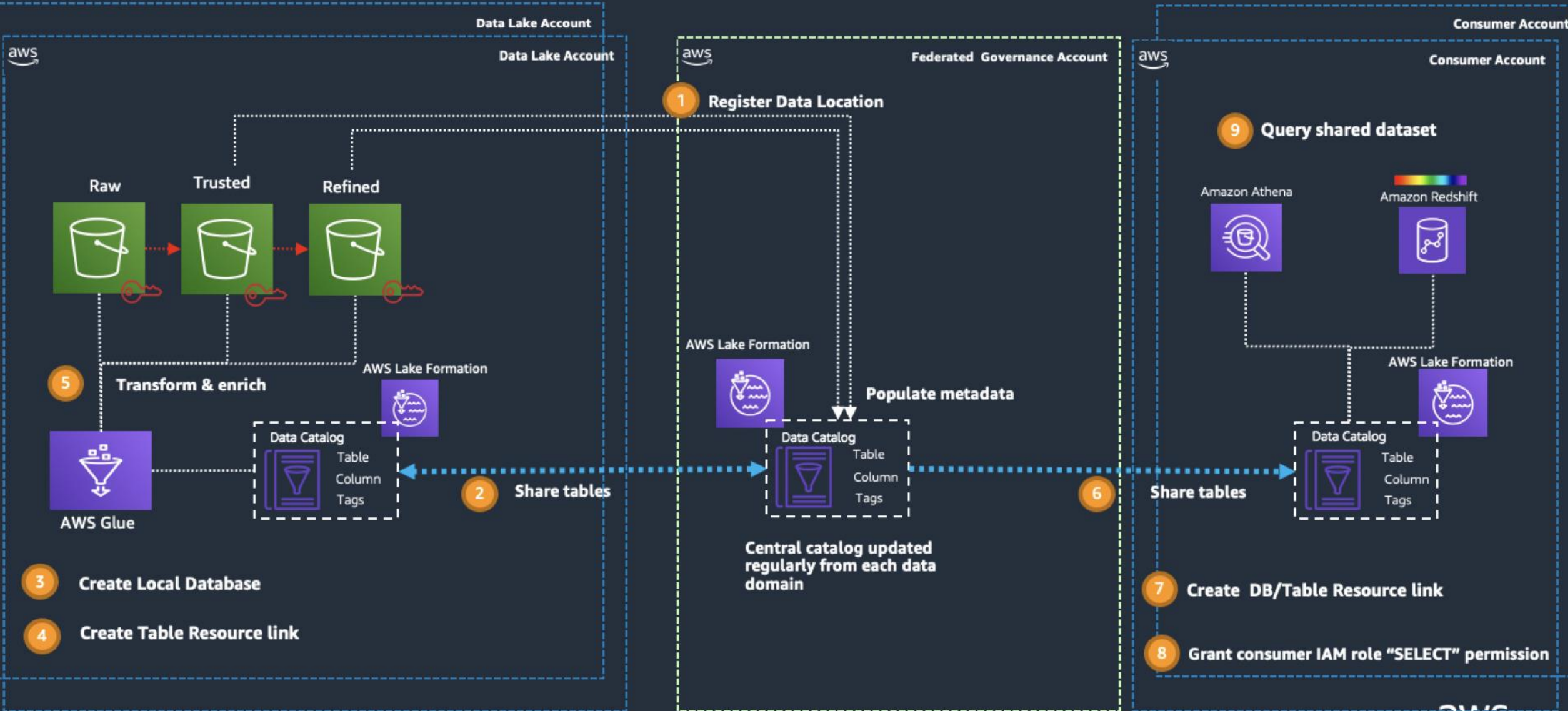
**Federated data Governance** is applied to how data products are shared – delivering discoverable **metadata** and auditability

A Data Consumer searches for catalog and gains access to a Data Product by accepting a **Resource Share** via the **Data Mesh pattern**





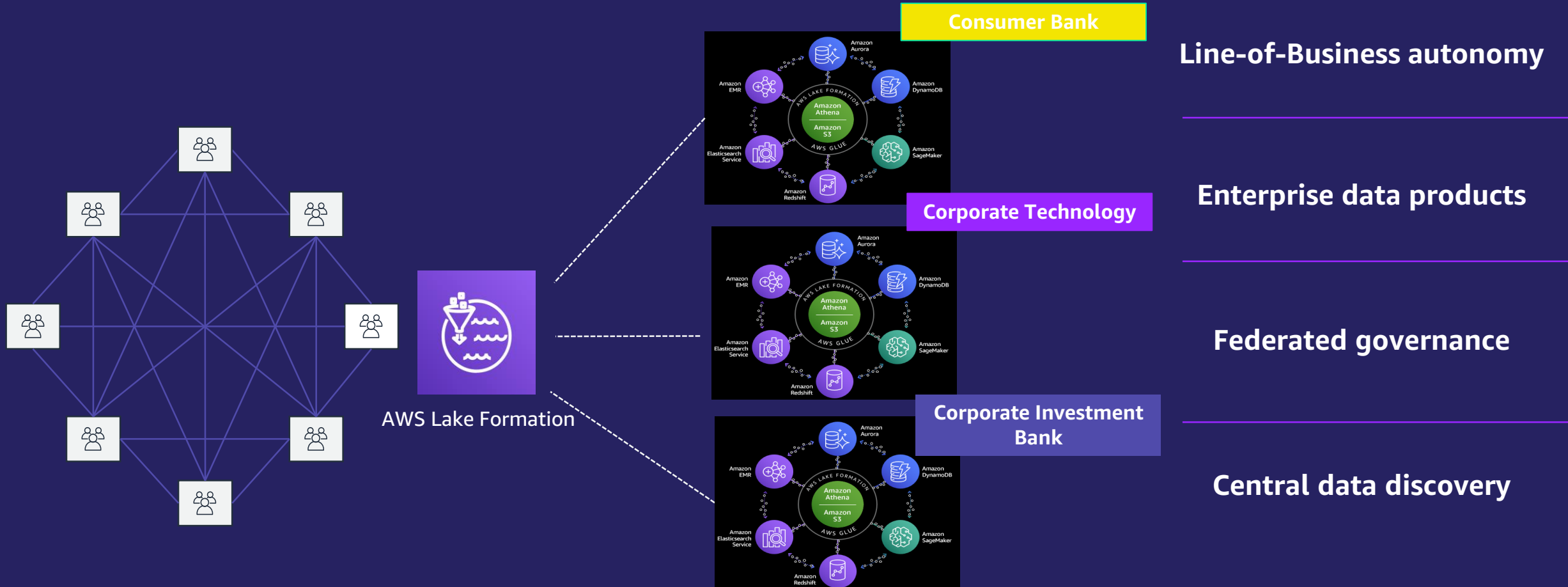
# Data Mesh on AWS – Central Governance



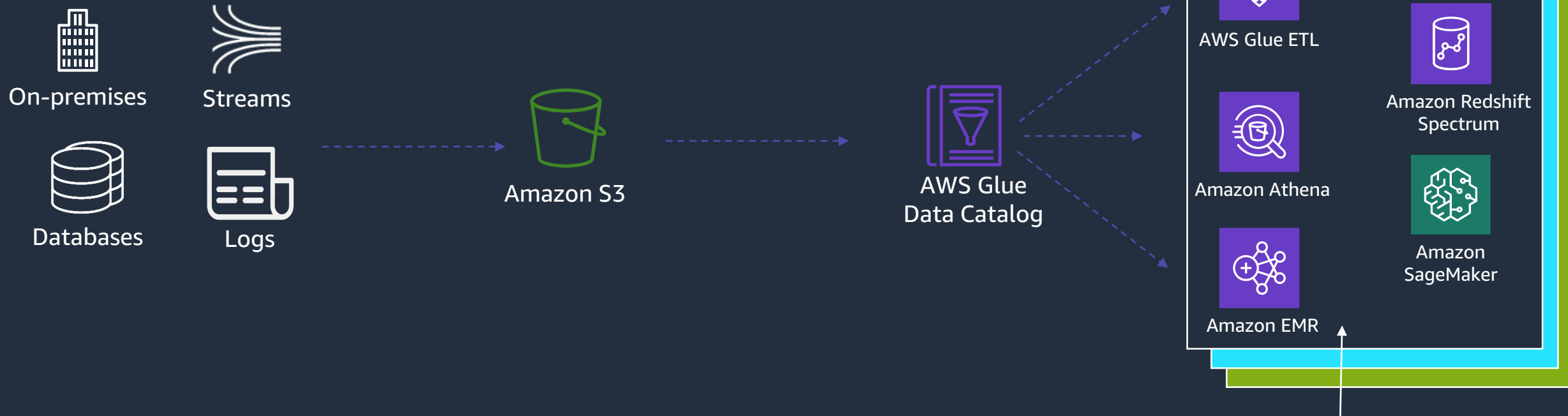


# Next evolution of our data lake

Extending the data lake following the data mesh design pattern



# Challenge: Integrations

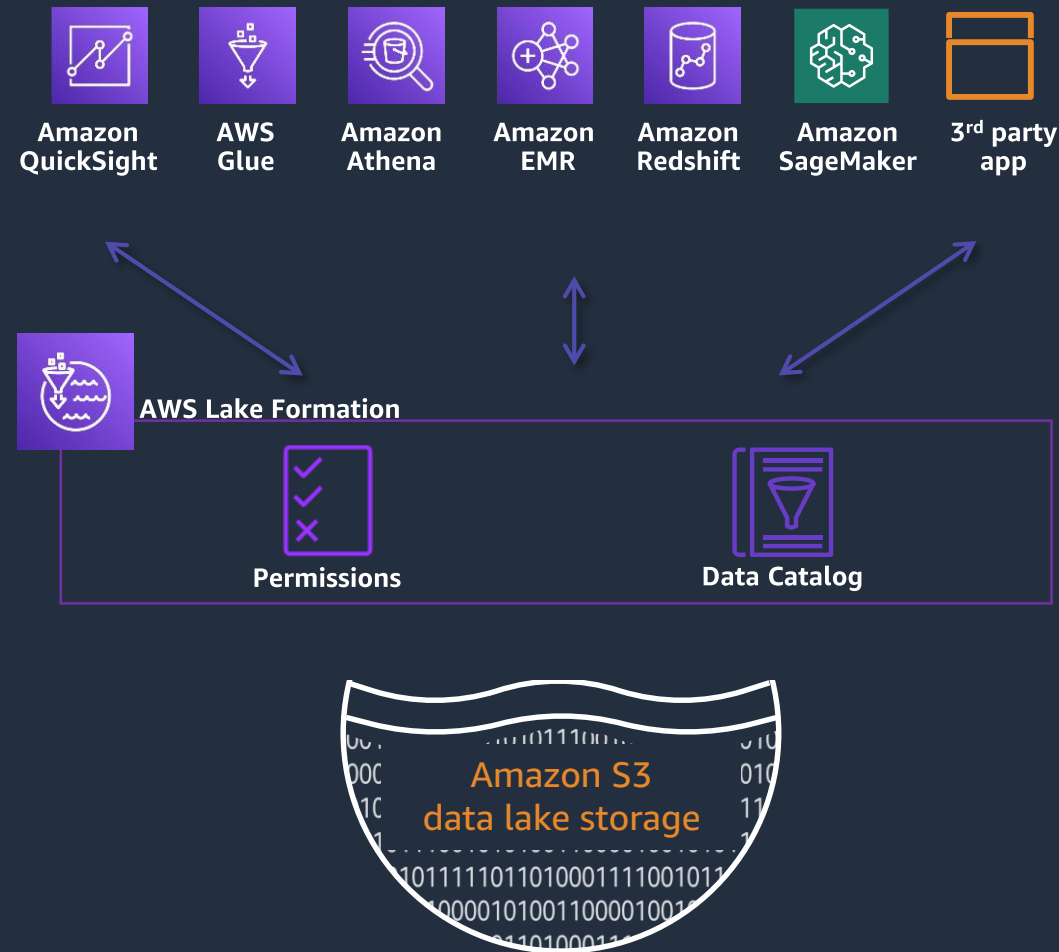


## Challenge #5: Integrations

A large set of integrated services is critical for productivity.



# AWS Lake Formation integrations



## Two integration options

### 1) Credential vending APIs

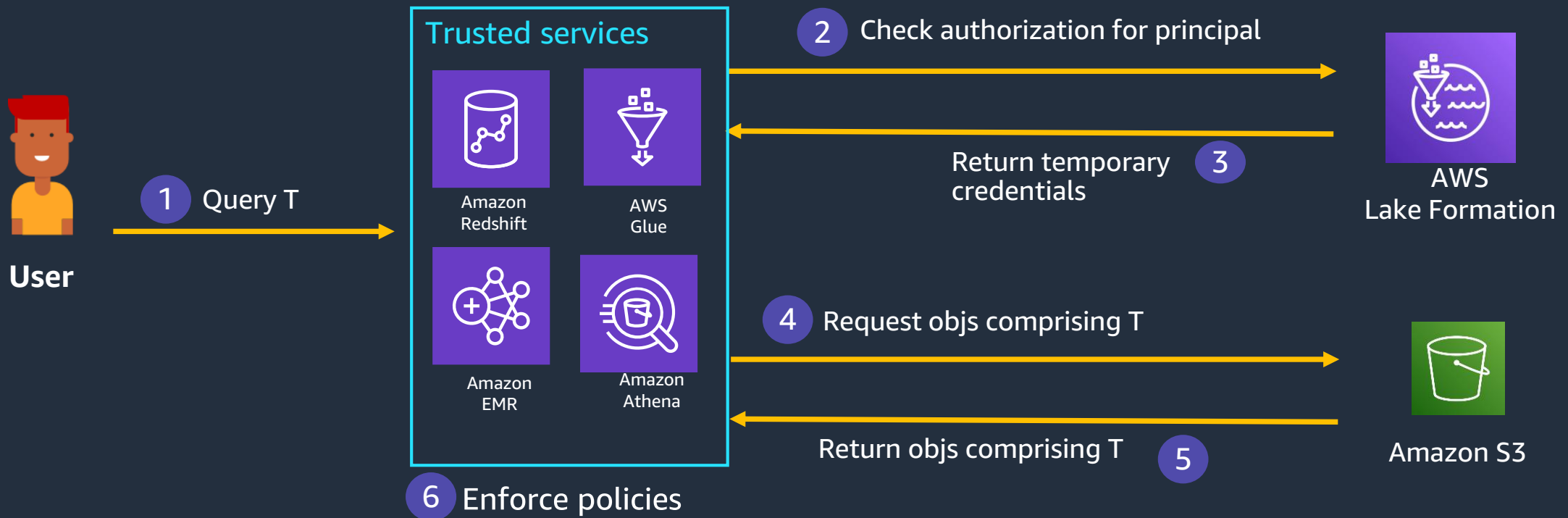
Distributed enforcement w/ fail close

### 2) Universal Data Access APIs

Centralized enforcement  
Simplified integrations

# Integration: Credential vending APIs

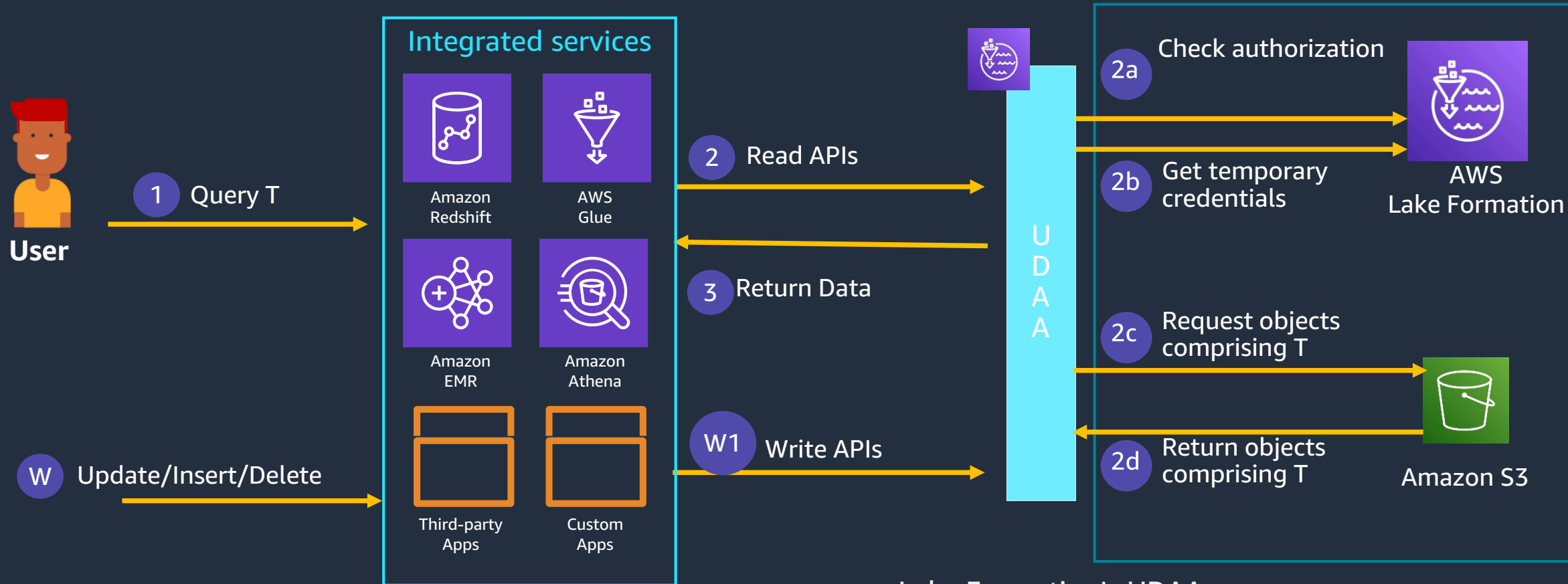
AWS Lake Formation manages access **to registered locations**



Trusted services enforce Lake Formation policies  
(distributed enforcement with fail close )

# Integration: Unified Data Access APIs

AWS Lake Formation consistently **reads and writes** to the table



Lake Formation's UDAA  
consistently enforces all policies

# Lake Formation 3<sup>rd</sup> party integrations



AWS Lake Formation

Policy store

Access authorization

Permission enforcement

↔ Integrate with external policy managers

↔ Authorize access and vend credentials to external engines

↔ Consistently enforce permissions for integrated engines

# Apply

## Get started





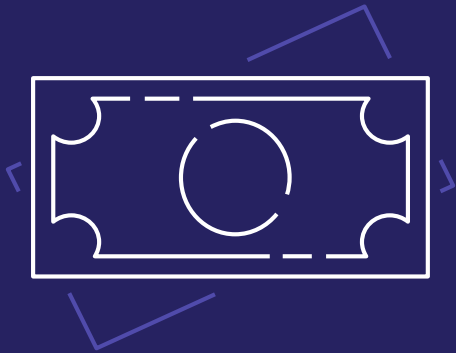
# Get the new storage API from AWS Lake Formation

<https://docs.aws.amazon.com/lake-formation/latest/dg/what-is-lake-formation.html#lake-formation-features>



# Lake Formation adoption trends

## Data Lake Migrations to AWS



Goals

TCO reduction, simplified security & management from start

Common attach use cases

**New data lake migrations:** on-prem Hadoop, Hive, Spark to EMR

**New data warehouse migrations:** legacy DW to Redshift

## Modernize Existing



Enable centralized security/governance, performance, data classification

**Modernize existing data lake:** Customers using Amazon S3 with Glue Catalog, and not using Lake Formation

## Federated Data Lakes (Mesh)



Increased collaboration, data silo reduction & Innovation

**Federate data lakes:** Customers looking to share data across data silos, in a well-governed but distributed data mesh pattern



# Data Lake Security & Governance program

## What's included?



### Free Two day Security/Sharing Envisioning workshop

Get introduced to AWS Lake Formation security, governance & sharing benefits and review security envisioning workshop. Workshop provides hands-on implementation guidance to unified data lake security via features like fine-grained permissions, data classification, tag based access and data sharing. Workshop is lead by Analytics Solution architects or partner SMEs. A quick assessment of customer's current security and sharing architecture is performed with recommendations on future-state architecture.



### Proof-of-concept or MVP support

Customer interested in POC or production implementation can get expert advise from Data Labs team or consultants from Pro-Serve and partners.



### Customer Training

Get hands-on access to immersion day labs that detail best practices around configuring data lakes security and sharing using AWS Lake Formation



### Modernization Incentives

\*POC acceleration credits, Data Labs, EDP Credits, or Must-Win credits may apply



# Amazon Redshift & AWS Lake Formation better together

Allows Redshift to enable access to all data



## Supported use cases & roadmap

Lake Formation attach Use Case	Supported	Roadmap
Fine-grained access for S3 data	IAM Role – Yes	GA
	SAML user/group -No	-
Redshift Share in Lake Formation	roadmap	Yes
Governed table access from Redshift	roadmap	GA
Native Redshift tables in Lake Formation	roadmap	Yes

## Why include Lake Formation?

- Simplified S3 Security– no need to manage complex S3 access policies
- Enable fine-grained access for S3 data accessible via spectrum
- No need to move all data in Redshift, handle changes in S3 data (inserts, updates, deletes) including time travel
- No additional cost- Improve overall price/performance ratio
- Handle changes in S3 data (inserts, updates, deletes)
- Enable data versioning (time travel) and transactional capability in the data lake
- Enable Redshift data access from other query engines like Athena, EMR via Redshift live share in Lake Formation (upcoming)



# ZS Associates save time with AWS Lake Formation storage API

EFFICIENT, TRUSTWORTHY SOLUTION PROVIDES INSIGHTS FROM DATA FOR FASTER DECISION-MAKING

## Challenge

ZS Associates needed to restrict access to data in a table to users based on their region/country. To accomplish this “row-level security,” ZS was creating a large number of Amazon Redshift “materialized views” that do a JOIN between the external table and a physical user-mapping table. However, this consumed large Redshift physical storage since all the TBs worth of data are in S3 and the “materialized views.” This physical storage consumption appended approximately 15% expenditure to ZS’s regular Redshift tally, since the company was using RA3 nodes with managed storage.

## Solution

With the AWS Lake Formation data-filtering feature, ZS no longer has to load TBs worth of data in Redshift to enforce fine-grained permissions. The table has a “country” column that helps identify the rows specific to a country.

## Result

The new storage API not only helped ZS reduce storage costs, but also reduced its development efforts by approximately 70%. ZS now uses this fully managed AWS service to create a reliable, performance-efficient, and secure solution to enable fine-grained access control on its data lake.



**Rustum Virani**

Director of Cloud Services

ZS Associates India Pvt.  
Ltd.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.



# Amazon EMR & AWS Lake Formation better together

## Simplified Data Lake Permissions



### EMR Migration Scenarios

### Use LF Security

Ranger/Atlas (on-prem) to EMR (Lift and shift)	No
Ranger/Atlas (on-prem) to Lake Formation Security	Yes (partner solution)
Spark(on-prem) to Spark on EMR - SAML notebook	Yes
Spark(on-prem) to Spark on EMR – Step Jobs	On roadmap
Hive/Presto on EMR	On roadmap

## Why include Lake Formation?

- Simplified S3 Security– no need to manage complex S3 access policies
- Enable fine-grained access for S3 data accessible via EMR running Spark, Hive, and Presto
- Migrate existing Atlas or Ranger policies to Lake Formation
- Handle changes in S3 data (inserts, updates, deletes) via Lake Formation Governed table. Opportunity to migrate Databricks' delta tables to Lake Formation.
- Enable data versioning (time travel) and transactional capability in the data lake



# AWS Glue & AWS Lake Formation better together

## Simplified ETL Modernization



### Glue Integration

Glue Catalog

In Lake Formation Console – Glue Jobs, Glue Crawlers

Lake Formation Blueprints invoke Glue Workflows

Machine Learning Transforms – built on Glue Api

Integration with Governed Table Transactions

## Compelling Reasons to Include Lake Formation

- Simplify Glue Ingestion and ETL pipeline by using Governed table transactions
- Enable reliability and consistency during data ingestion
- Leverage auto compaction in Governed tables, no need to develop and execute manual ETL operation
- Simplify permission management and discovery of Glue resources in Lake Formation console
- Augment Glue's coarse grained permissions with Lake Formation fine-grained permissions.



# More data lakes & analytics than anywhere else

TENS OF THOUSANDS OF DATA LAKES RUN ON AWS ACROSS ALL INDUSTRIES

AMGEN

BMW  
GROUP

CHANGE  
HEALTHCARE

COMCAST

Continental

coursera



Discovery  
CHANNEL

duolingo

EA

ENGIE



expedia  
group

experian

Fannie Mae

FICO

airbnb

FOURSQUARE

INCHCAPE  
SHIPPING SERVICES

INNOVO

intuit

INVISTA

ironSource

Klarna

Liberty Mutual  
INSURANCE

mercado  
libre

Movable Ink



GRUBHUB



docomo

New Relic

NuData Security  
mastercard

OLX

PADDYPOWER  
betfair

PennyMac  
Correspondent

Pinterest

REDFIN

robinhood

ROBLOX



SIEMENS

slack

Snap Inc.

Sysco

Goldman  
Sachs

ANA

theTradeDesk

TOYOTA

Vanguard

vyaire  
MEDICAL

yelp

Zillow

ZipRecruiter



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.



# Thank you!