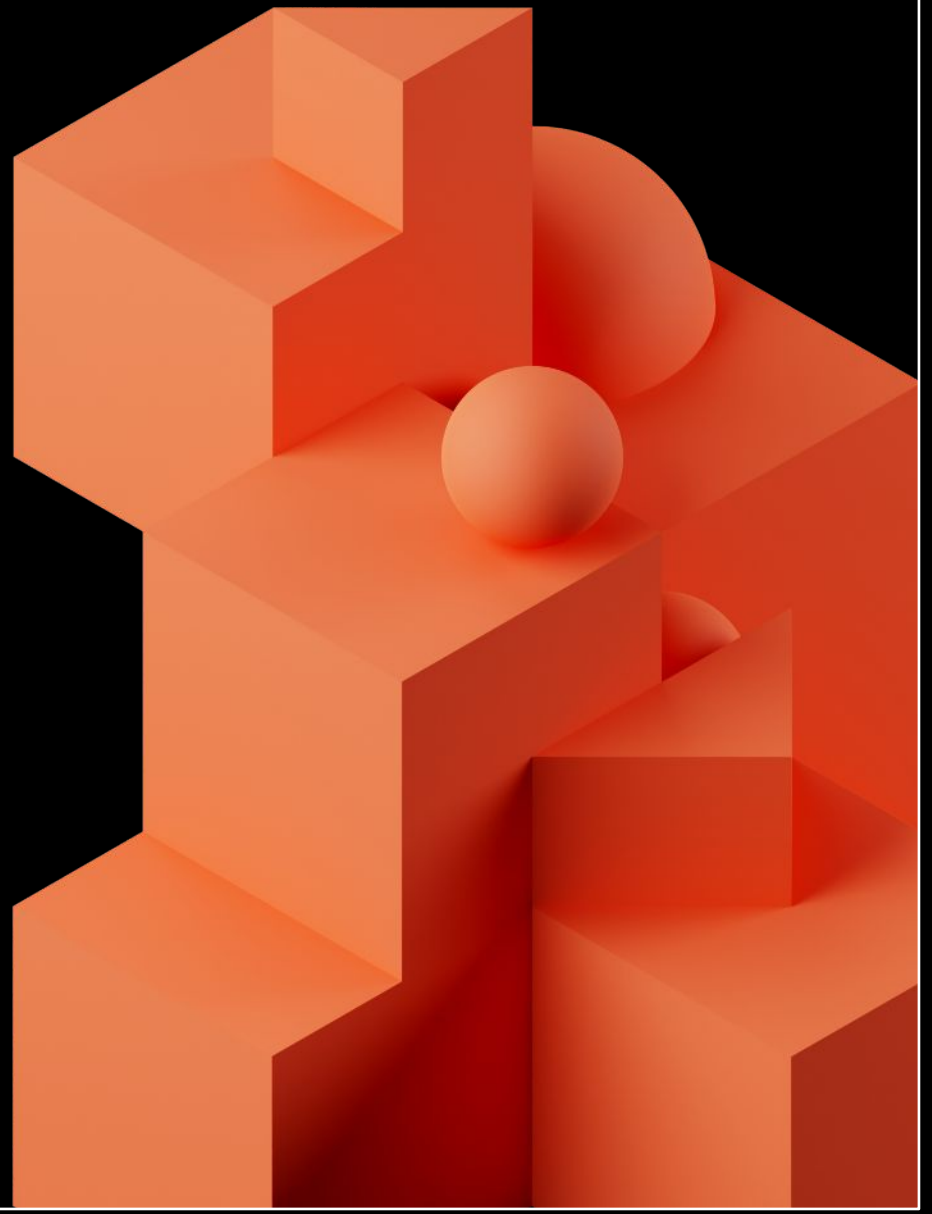
DATA+AI SUMMIT
BY databricks

A Technical Deep Dive into Unity Catalog's Practitioner Playbook



Zeashan Pappa
Ifi Derekli

Product safe harbor statement

- This information is provided to outline Databricks' general product direction and is **for informational purposes only.** Customers who purchase Databricks services should make their purchase decisions relying solely upon services, features, and functions that are currently available. Unreleased features or functionality described in forward-looking statements are subject to change at Databricks discretion and may not be delivered as planned or at all.

Zeashan Pappa

Product Leader - Data Governance

- 22+ years of experience
- 6 years of experience building with Databricks
- 10 years of experience building scalable data platforms
- Product leader working across field teams and product management
- Has less hair in real life
- Constantly bothers Ifi

Ifi Derekli

Field Engineering Manager &
Unity Catalog Specialist

- 13+ years of experience with big data platforms
- 3 years of experience on Databricks
- 6 years of security & governance focus
- Patient with Zeashan and able to keep up with his crazy speed of thought

Agenda

- Unity Catalog and Cloud Providers
- Register data with Unity Catalog
- Secure your data
- Discover your data with Search and Lineage
- Audit your data
- Open data sharing
- Upgrade to Unity Catalog
- Architecture Patterns
- Demo

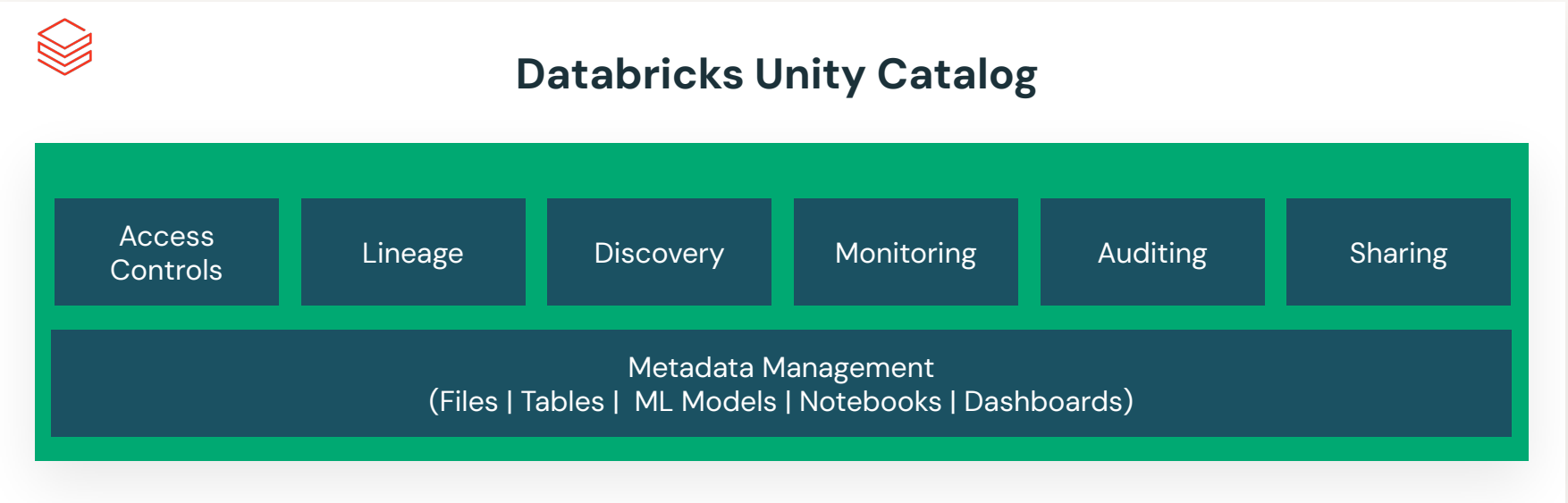
Databricks Unity Catalog
Unified governance for data and AI

Unified visibility into data and AI

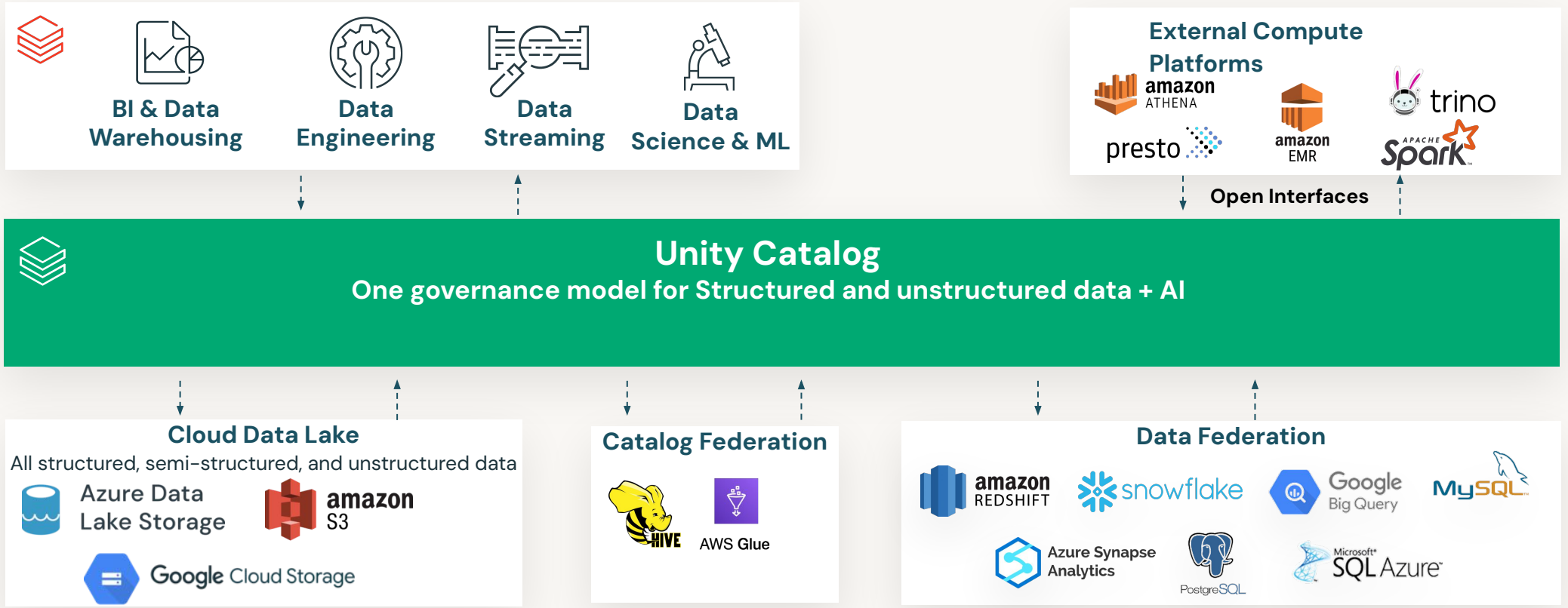
Single permission model for data and AI

AI-powered monitoring and observability

Open data sharing



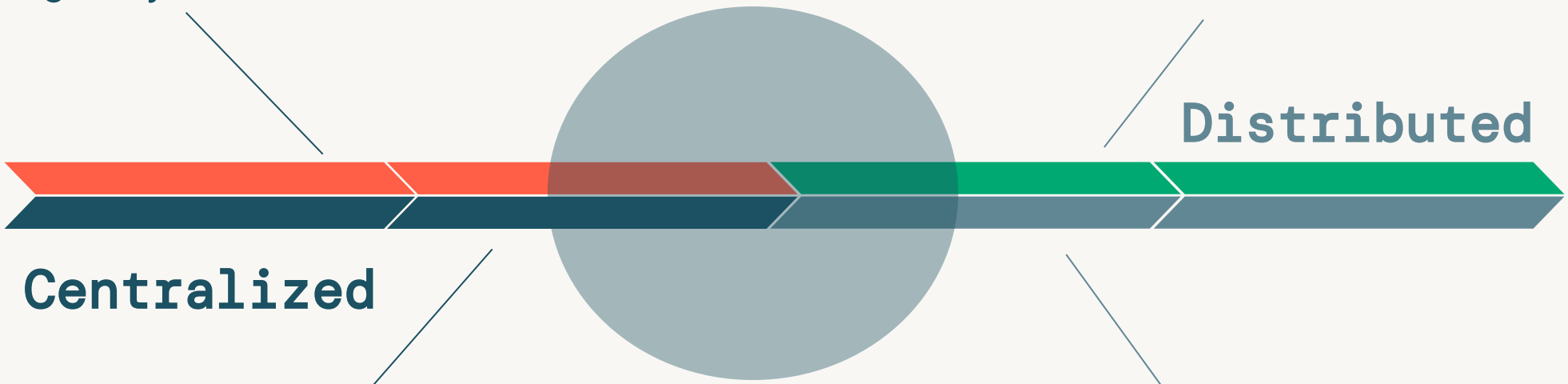
Databricks Lakehouse unifies data and AI governance



Spectrum of Organizational Governance

Production of data artifacts managed by a central team

Domain driven production of data artifacts

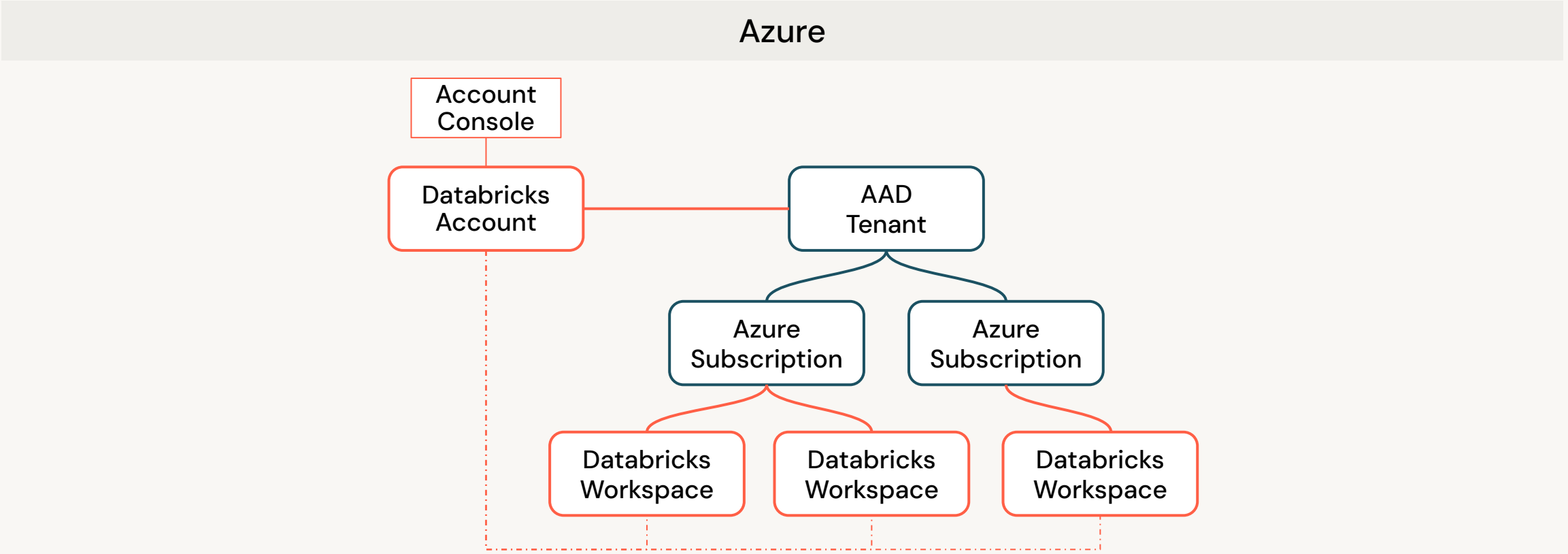


Governance policies applied and enforced by a central team

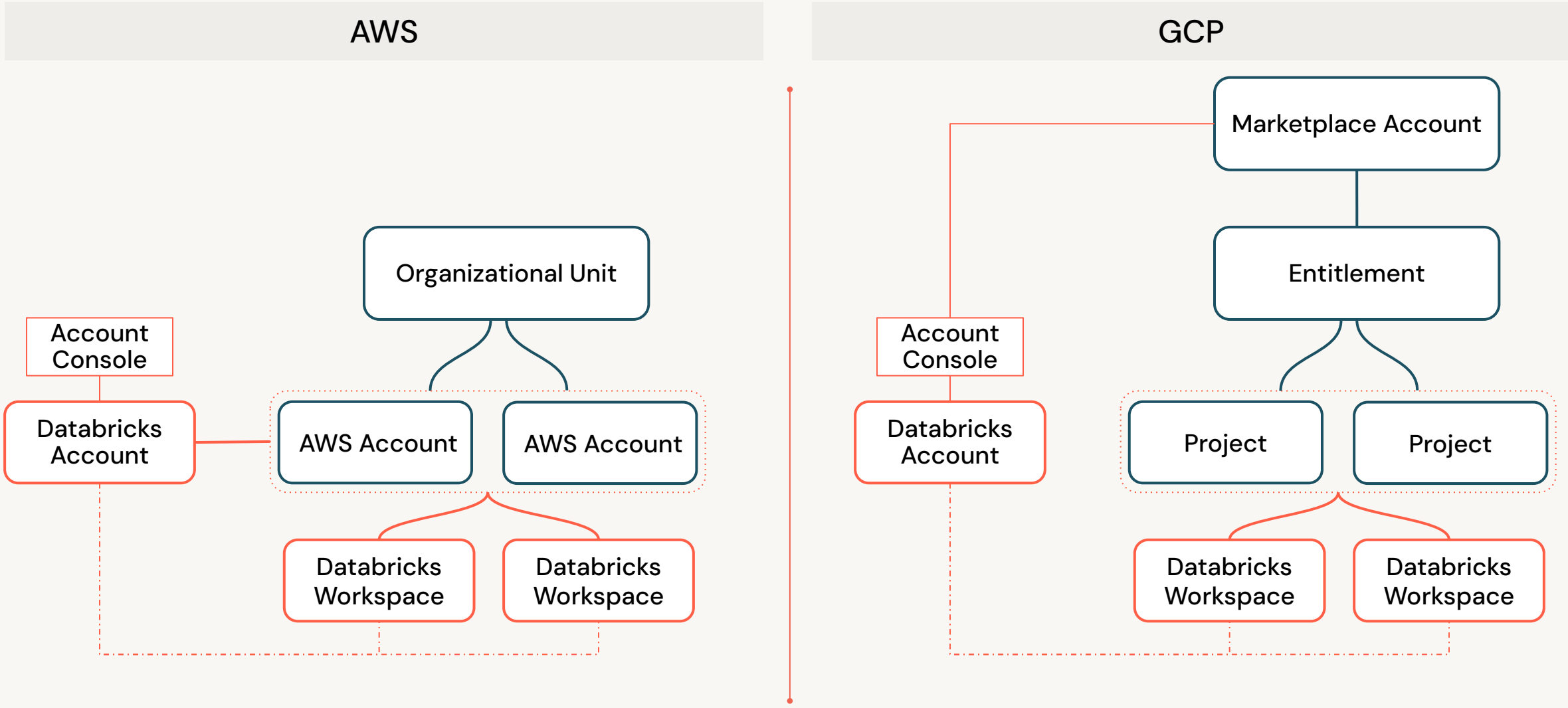
Entitlements on data owned by domain teams

Unity Catalog and Cloud Providers

Databricks Accounts and Cloud Providers



Databricks Accounts and Cloud Providers



Unity Catalog and Cloud Constructs

	AWS	Azure	GCP
Databricks Account	Accounts	Tenant	Marketplace Account
Metastore	Region	Region	Region
Catalog	*Account**	*Subscription**	*Project**
Storage Location	S3 Bucket	ADLS Account	GCS Bucket
Credential	IAM Role	Managed Identity	Service Account

* *Minimum one, more are optional*

Key Roles in Unity Catalog

Assign roles to groups

Account Admin Group

- Can create workspaces *(*in Azure any contributor can)*
- Can create & configure metastores
- Can create users, groups, & service principals
- Can create credentials
- Can grant users access to workspaces
- *Recommended: Platform Ops or Central Gov Team*

Metastore Admin Group

- Can create CATALOG, EXTERNAL LOCATION
- Can create SHARE, RECIPIENT
- Can change OWNER of any securable object
- *Recommended: Platform Ops or Central Gov Team*
- *Recommended: Delegate to Data Owners*

Workspace Admin Group

- Can add users and groups to workspace
- Can create clusters & cluster policies
- Can change OWNER of clusters, workflows, notebooks, queries, dashboards
- *Recommended: IT/Platform/DevOps team*
- *Recommended: Regular Service Principal Audits*

Data Owner Group

- Can change ownership of securable object (CATALOG, SCHEMA, TABLE, VIEW, etc.)
- Can grant any privilege to any principal
- *Recommended: BU Gov Team or Central Gov Team*

Best Practices for working with Cloud Constructs

Organizational patterns dictate usage

Centralized

- Data and compute is typically stored in **central subscriptions / accounts / projects** in the the cloud, by SDLC scope (e.g. DEV vs PROD).
- One team controls cloud **infrastructure** → One team controls creation of **credentials and external locations.**
- **One** team **produces data** → **One** team manages **production pipelines**.
- **One** team owns **governance and access** → **One** team will administer and **own Metastore and Catalogs** and manage permissions.
- ***Central team* carries administrative responsibilities both in the cloud and in Databricks.**

Distributed

- **Each team has their own subscription(s) / account(s) / project(s)** in the cloud with their own storage and compute for data isolation and cost allocation → each team owns their own **catalog(s) and workspace(s).**
- **Many or one** team controls cloud **infrastructure** → **Many or one** team controls creation of **credentials and external locations.**
- **Many** teams **produce data** → **each** team manages their **own workspaces / pipelines.**
- ***Each team* carries *shared* administrative responsibilities both in the cloud and in Databricks.**

Register data with Unity Catalog

PREVIEW

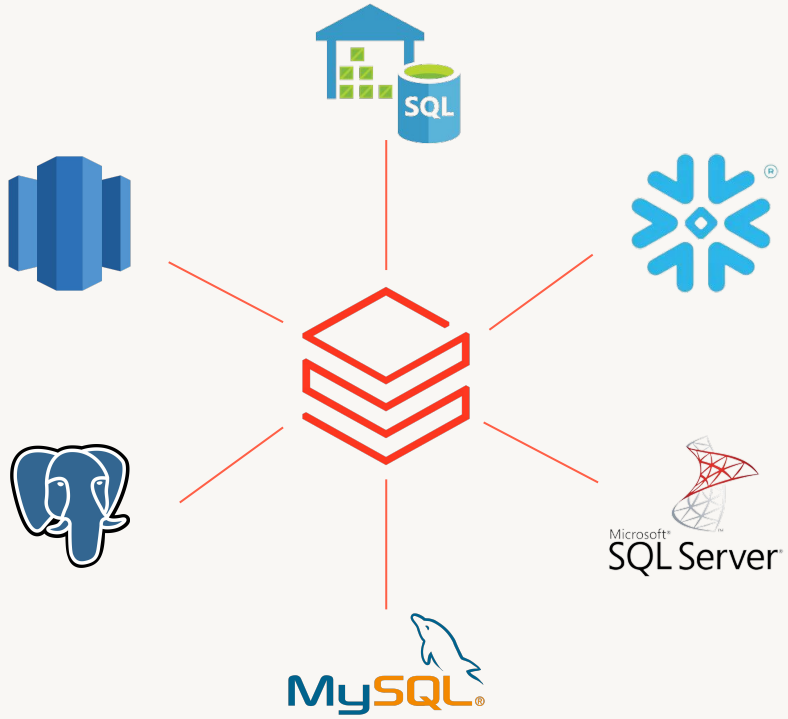
Query Federation

Unify your entire data estate with lakehouse

Query Federation provides **one single point of secure access to all your data** - no matter where it lives - and one way to access, catalog, govern, and query all your data - **no ingestion required**.

- **Unified permission controls**
- Intelligent pushdown optimizations
- Accelerated query performance with Materialized Views
- Support for R/O operations today

```
CREATE FOREIGN CATALOG <catalog_name>
USING CONNECTION <connection_name>
OPTIONS (database '<remote_database>')

SELECT * FROM <catalog_name>.<schema_name>.<table_name>
```



Fundamental Concepts

Working with file based data sources

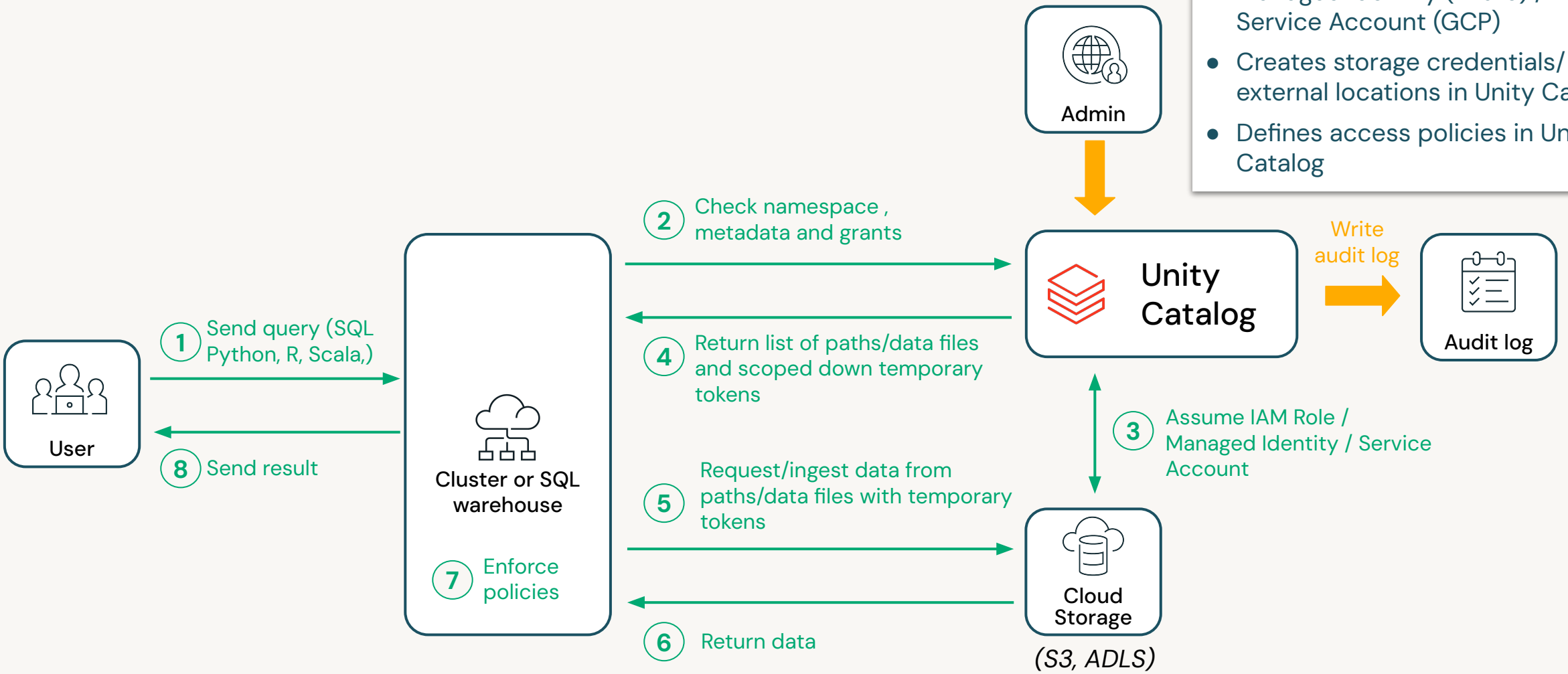
- Credentials
  - Cloud provider credential to connect to storage
- External Locations
  - Storage location used for external tables, external volumes, or arbitrary files, or default managed location for a catalog or schema
- Managed / External Tables
  - Tabular data stored in managed or external locations
- Managed / External Volumes
  - Arbitrary file container inside a managed or external location

Working with databases

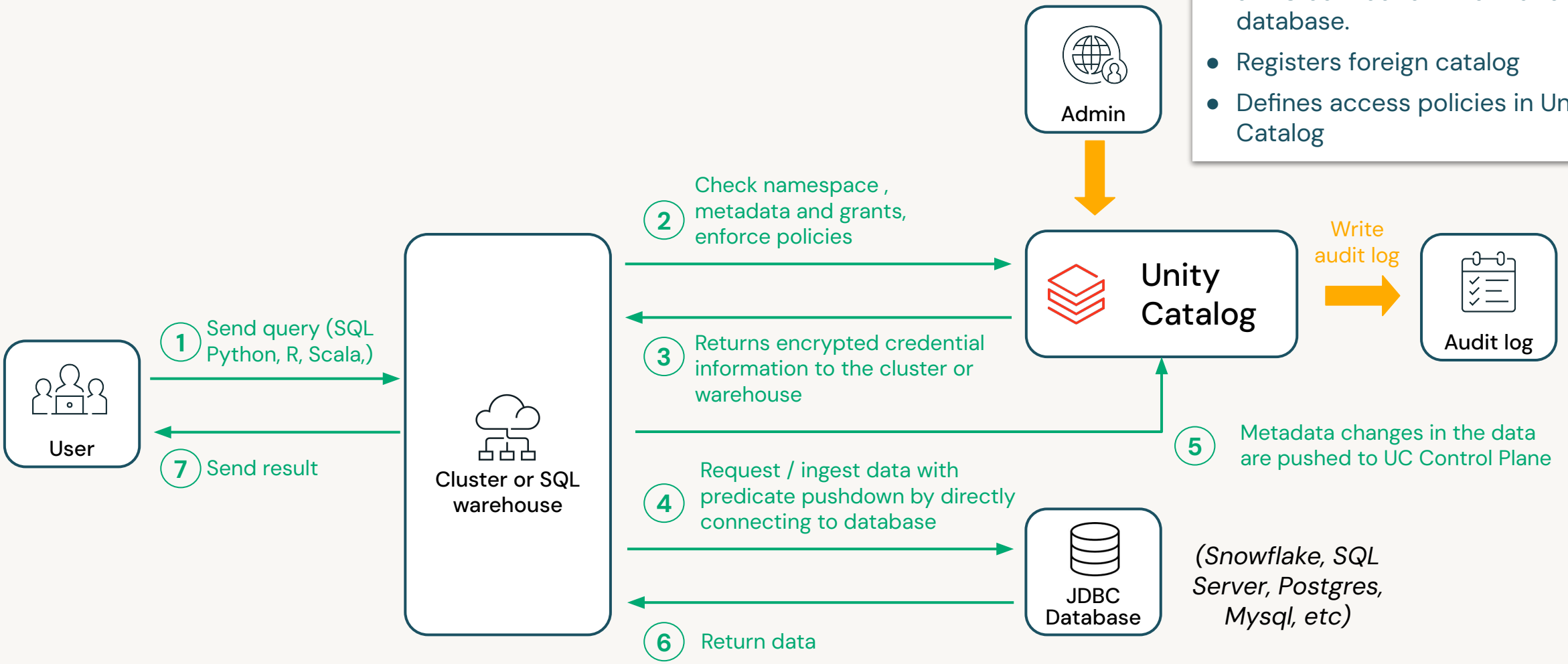
- Connections
  - Credential and connection information to connect to an external database
- Foreign Catalogs
  - A catalog that represents an external database in UC and can be queried alongside managed data sources and file sources

Querying file based data sources with Unity

- Creates an IAM role (AWS)/ Managed Identity (Azure) / Service Account (GCP)
- Creates storage credentials/ external locations in Unity Catalog
- Defines access policies in Unity Catalog



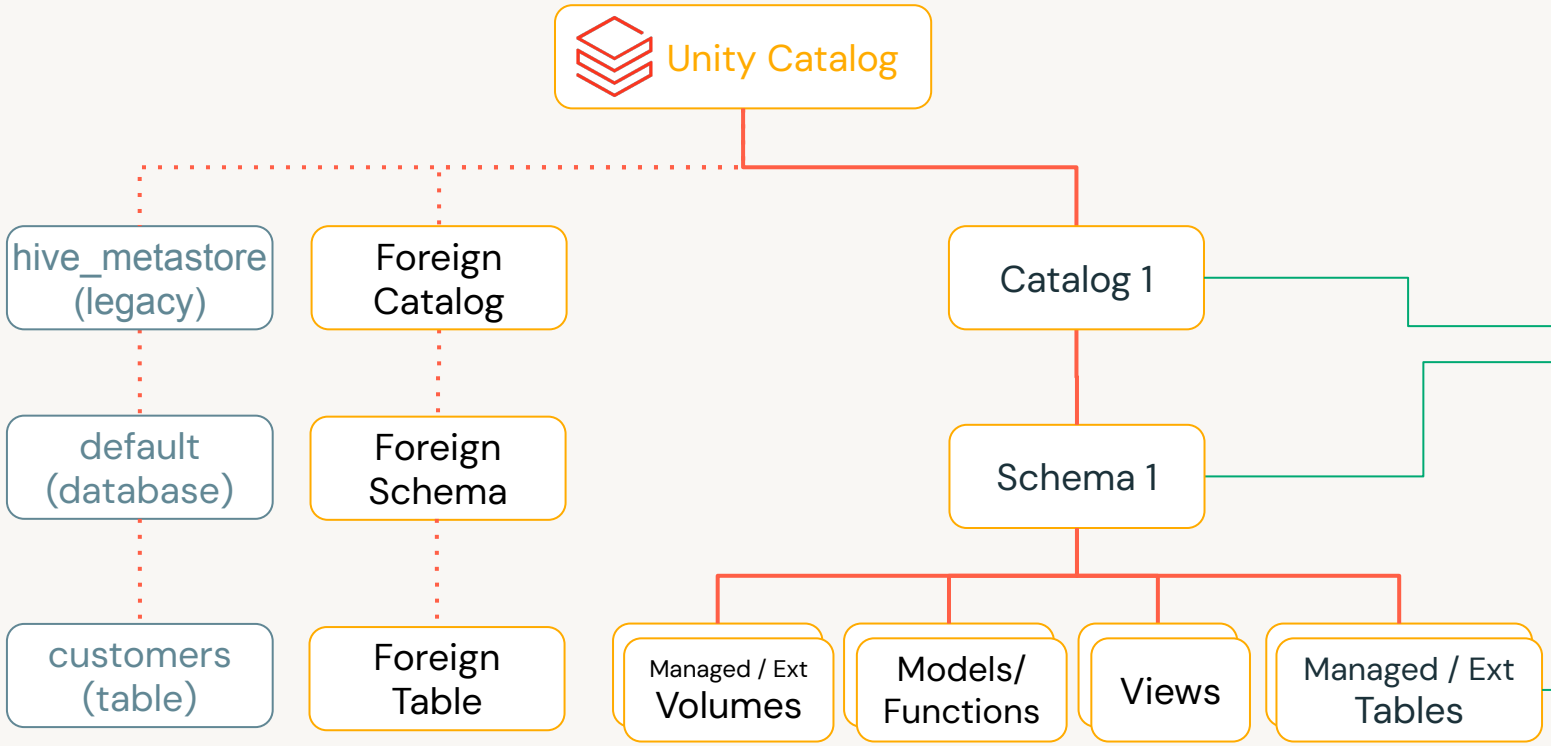
Querying database sources with Unity

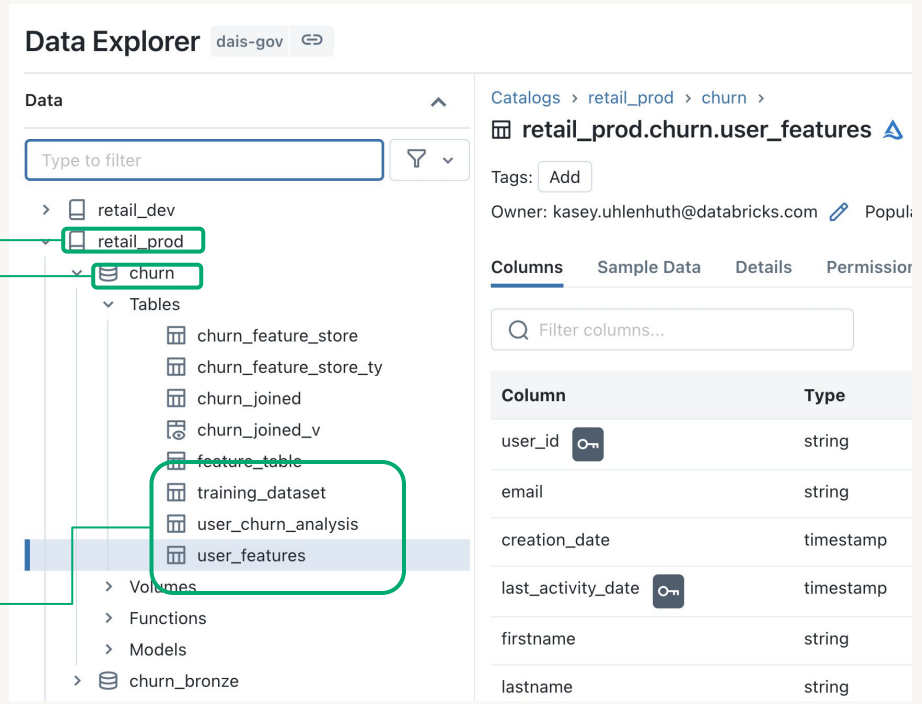


- Defines Connection object with JDBC connection information to database.
- Registers foreign catalog
- Defines access policies in Unity Catalog

Governed namespace across file and database sources

Access legacy metastore and foreign databases powered by Query Federation `






```
SELECT * FROM main.paul.red_wine;  -- <catalog>.<database>.<table>

SELECT * FROM hive_metastore.default.customers;

SELECT * FROM snowflake_warehouse.some_schema.some_table;
```

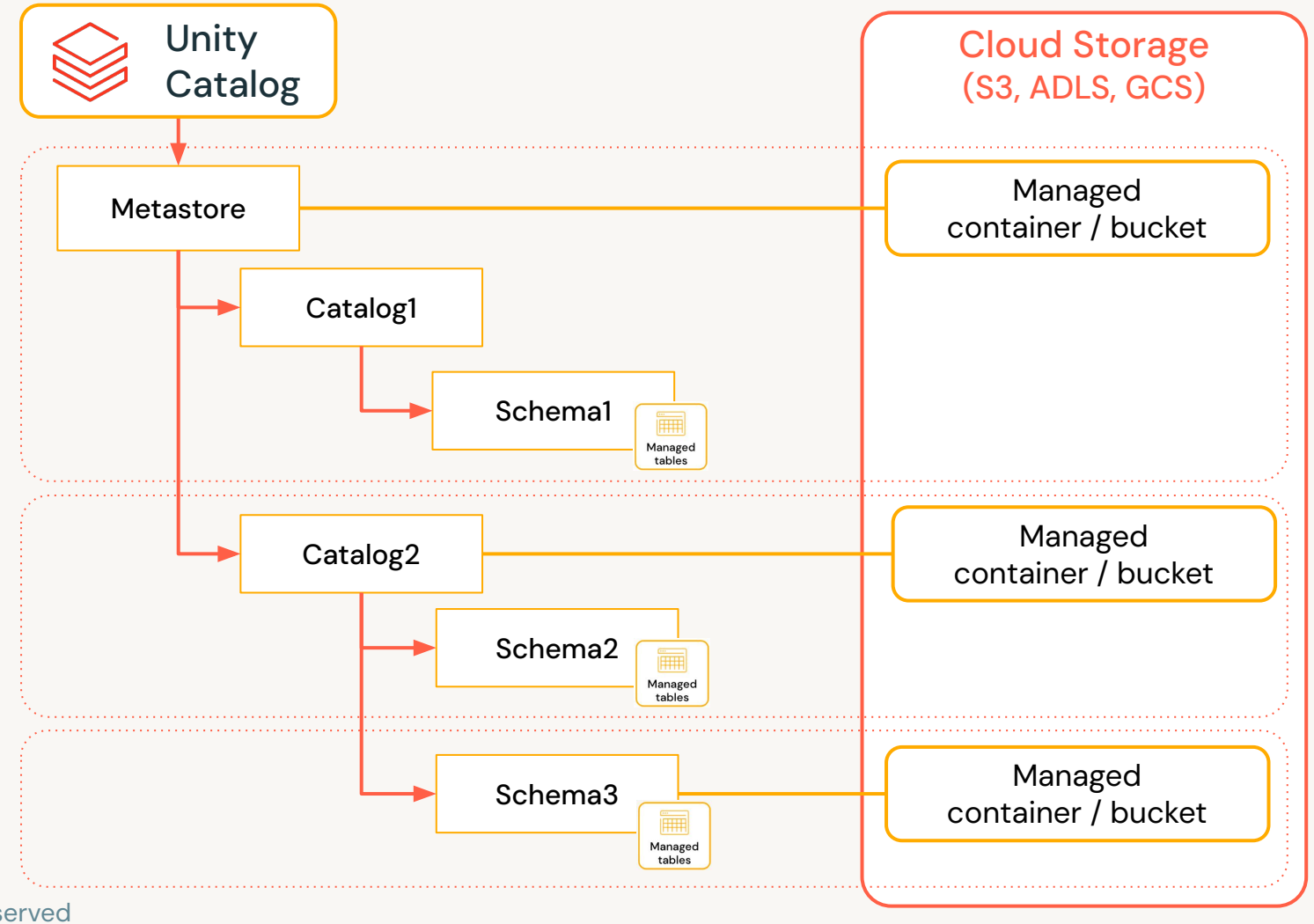
Isolation between file based data sources

Use managed data sources for data isolation or cost allocation

1) Store at the **metastore**

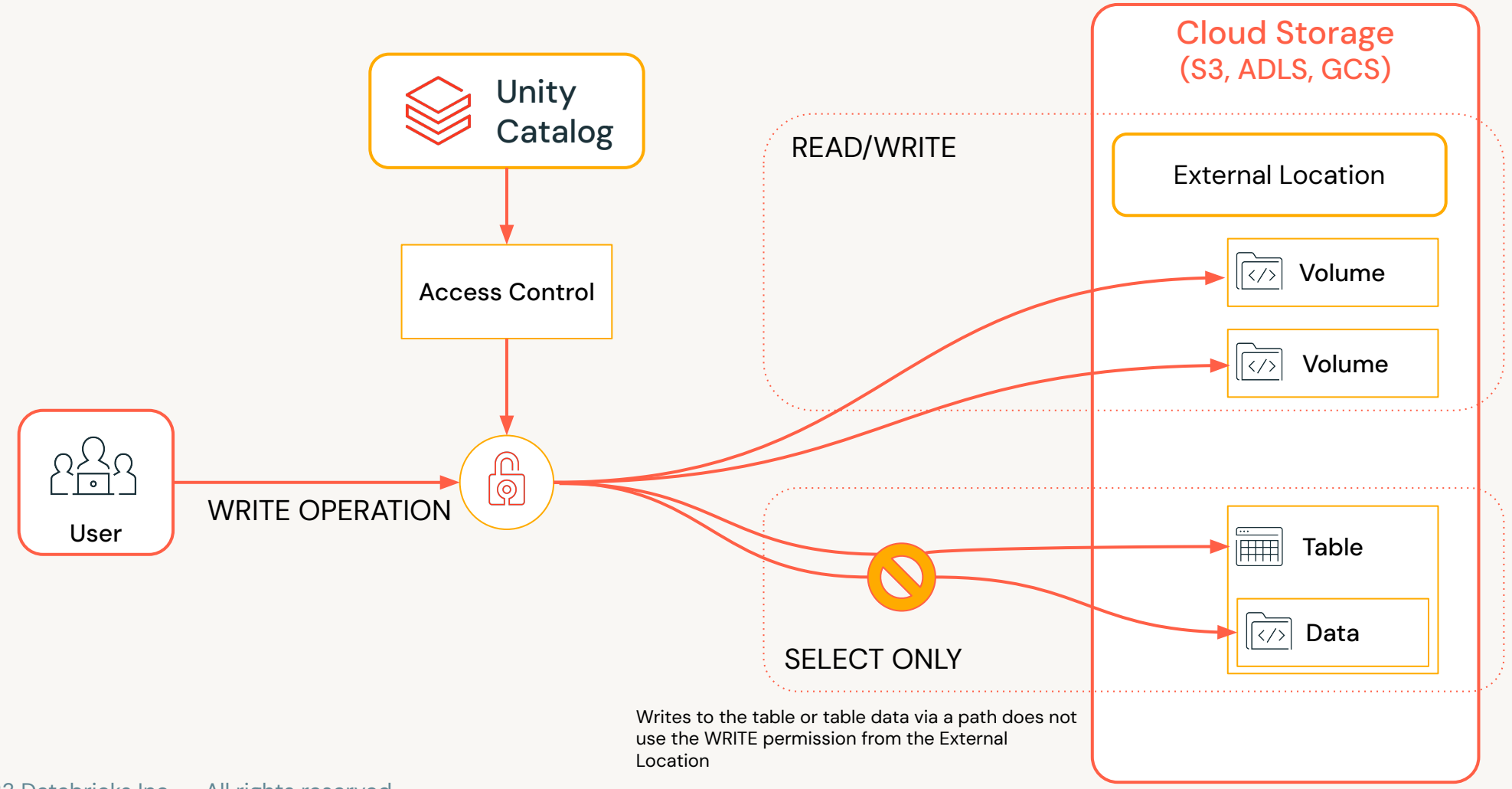
2) Store at the **catalog**

3) Store at the **schema**



Multiple ACL trees for flexible governance

Govern external tables and file based data source access separately



PREVIEW

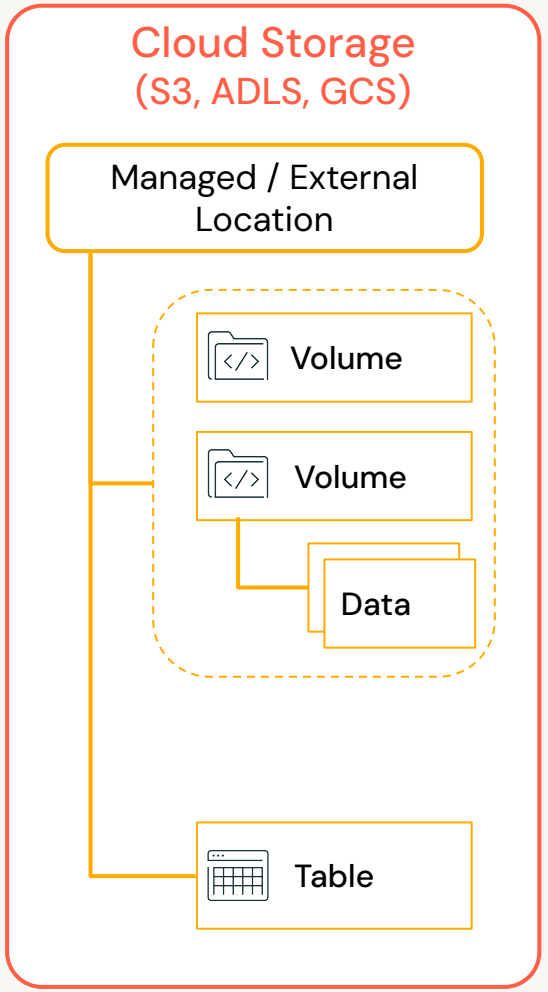
Volumes in Unity Catalog

Access, store, organize and process files with Unity Catalog governance

- Volumes can be accessed by some POSIX commands

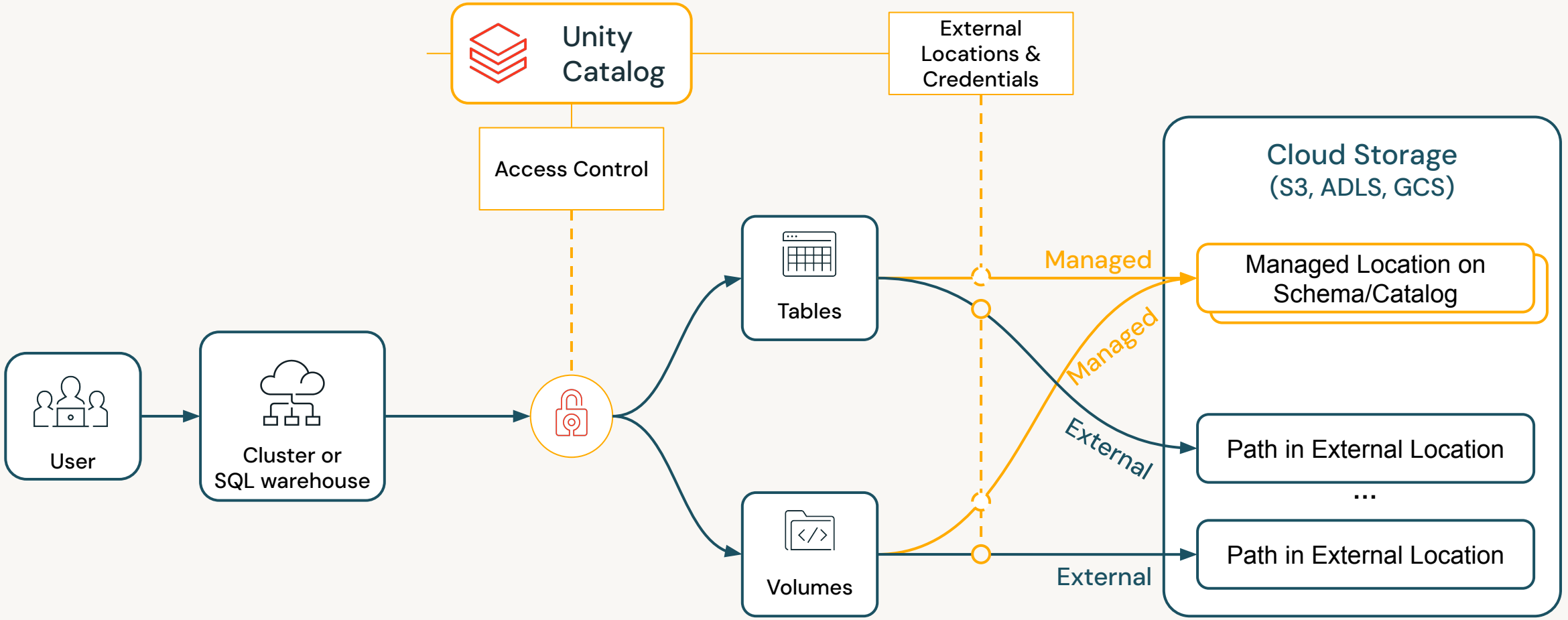
```
dbutils.fs.ls("s3://my_external_location/Volumes/catalog/schema/volume123")

ls /Volumes/catalog/schema/volume123
```

- Volumes are created under Managed or External Locations and show up in UC Lineage
- Volumes add governance over non-tabular data sets
  - Unstructured data, e.g., image, audio, video, or PDF files, used for ML
  - Semi-structured training, validation, test data sets, used in ML model training
  - Raw data files used for ad-hoc or early stage data exploration, or saved outputs
  - Library or config files used across workspaces
  - Operational data, e.g., logging or checkpointing output files
- Tables are registered in Managed / External Locations, not in Volumes



Defining file based data sources in Unity

Simplify data access management across clouds

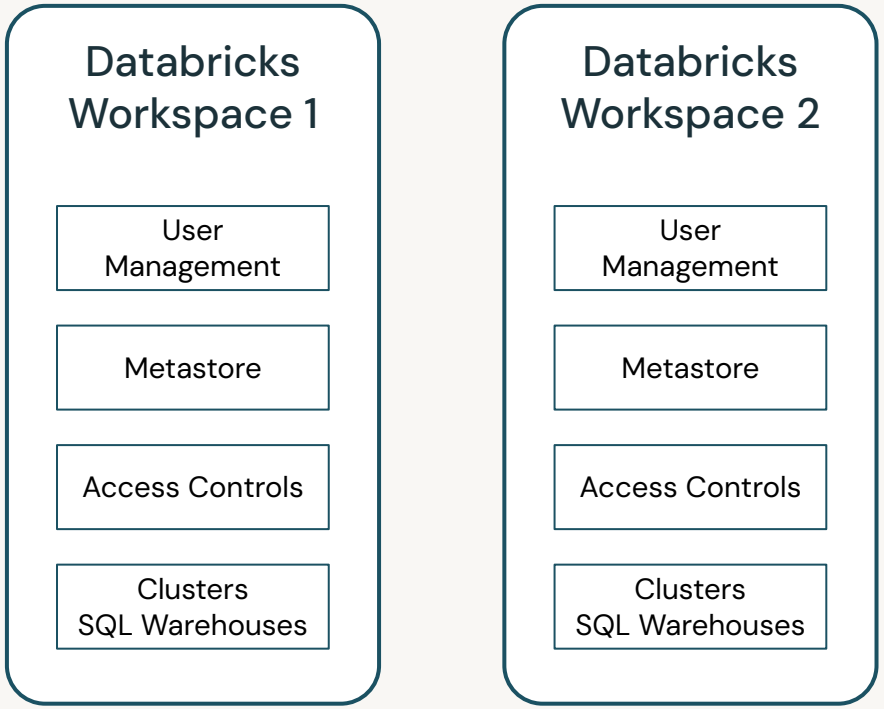


Secure your Data

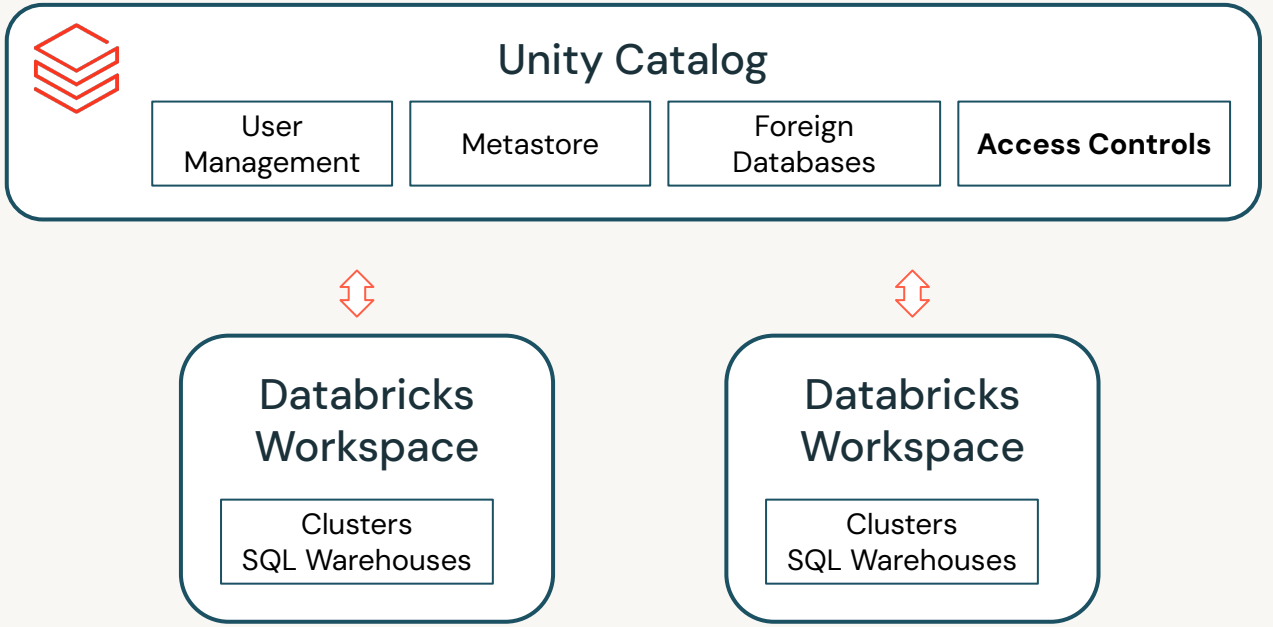
All your metadata, in one place

One metadata layer across file and database sources **superpowers** governance

Without Unity Catalog



With Unity Catalog



Centralized Access Controls

Centrally grant and manage access permissions across workloads and foreign databases

Using ANSI SQL DCL

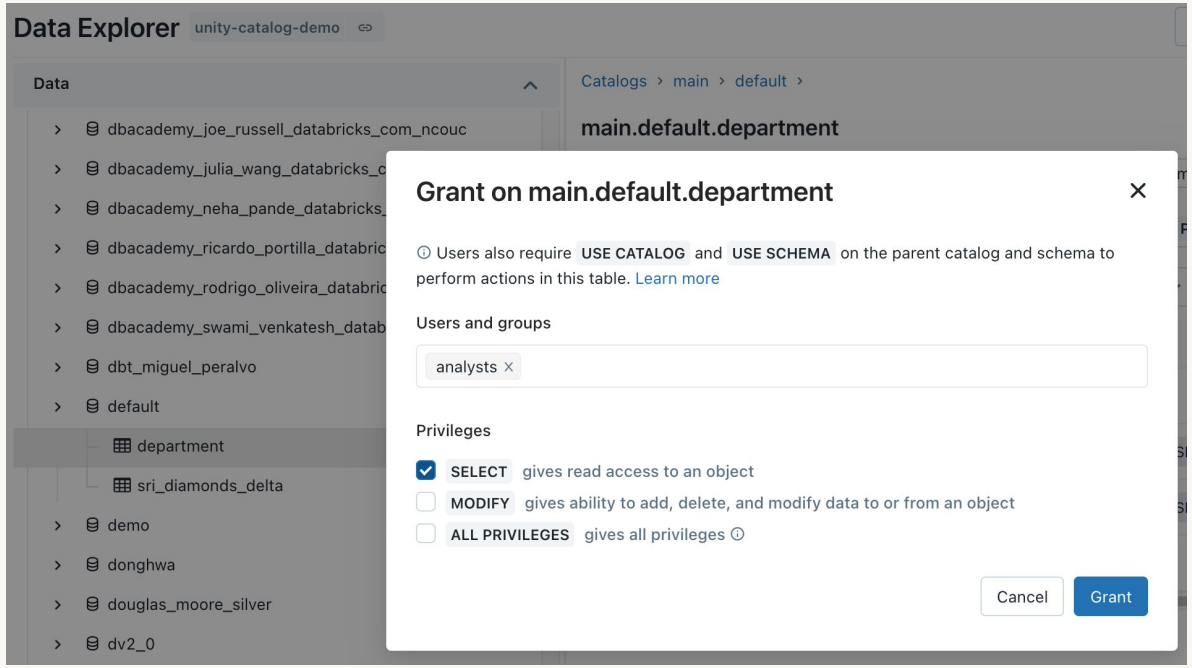
```
GRANT <privilege> ON <securable_type>
<securable_name> TO `<principal>`

GRANT SELECT ON iot.events TO engineers
```

Choose permission level

'Table'= collection of files in S3/ADLS

Sync groups from your identity provider

Using UI



PREVIEW

Row Level Security and Column Level Masking

Provide differential fine grained access to file based datasets and foreign tables

Only show specific rows

```
CREATE FUNCTION <name> ( <parameter_name >
<parameter_type> .. )
RETURN {filter clause whose output must be a boolean}
```

```
CREATE FUNCTION us_filter(region STRING)
RETURN IF(IS_MEMBER('admin'), true, region="US");

ALTER TABLE sales SET ROW FILTER us_filter ON region;
```

Test for group membership

Assign reusable filter to table

Specify filter predicates

Mask or redact sensitive columns

```
CREATE FUNCTION <name> (<parameter_name>,
<parameter_type>, [, <column>...])
RETURN {expression with the same type as the first
parameter}
```

```
CREATE FUNCTION ssn_mask(ssn STRING)
RETURN IF(IS_MEMBER('admin'), ssn, "****");

ALTER TABLE users ALTER COLUMN table_ssn SET MASK
ssn_mask;
```

Test for group membership

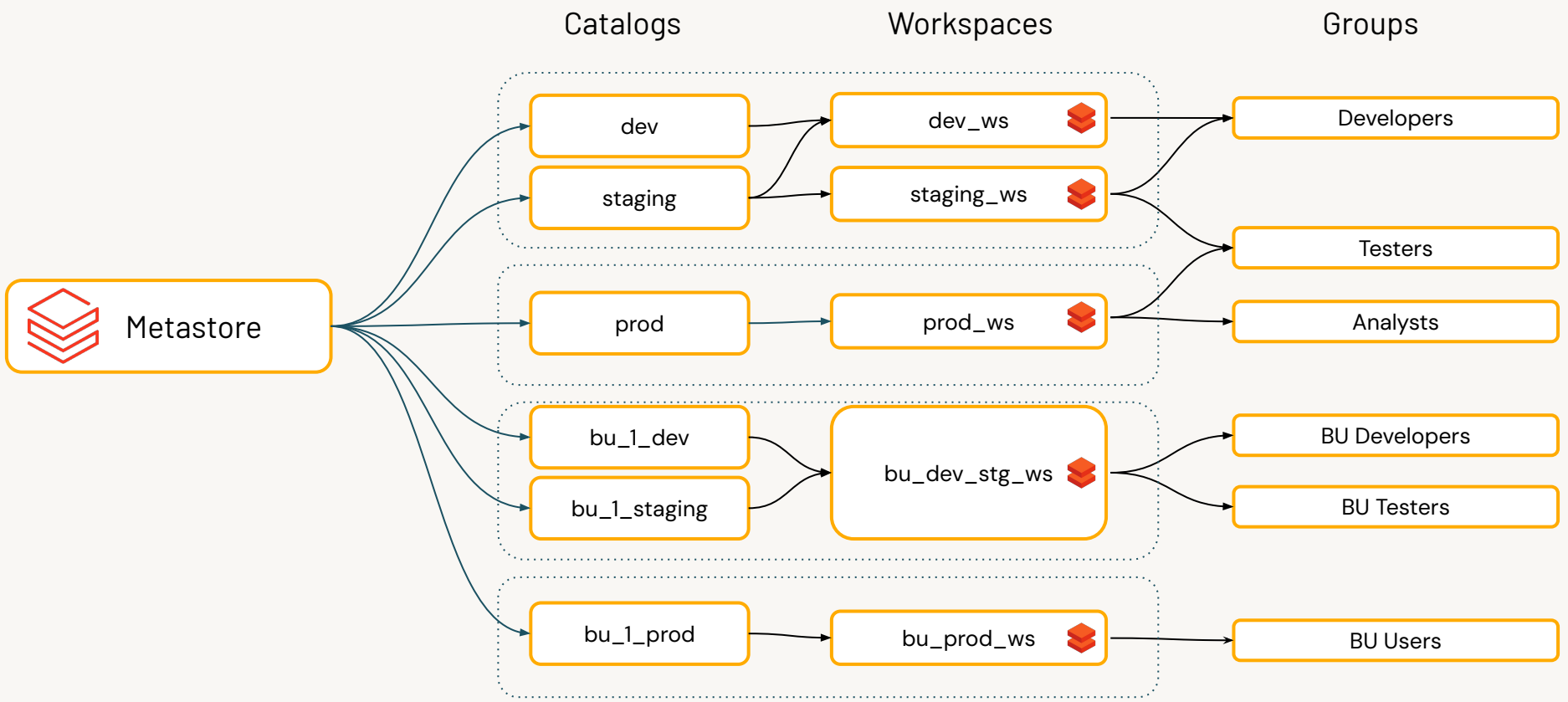
Assign reusable mask to column

Specify mask or function to mask

PREVIEW

Access data from specified environments only

Restrict catalog access by environment or purpose



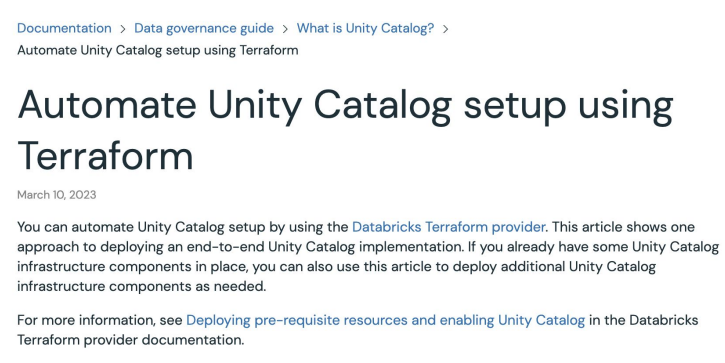
Access to data and availability of data can be isolated across workspaces and groups

High Leverage Governance with Terraform & APIs

Use data-sec-ops, policies as code patterns to scale your efforts

- Privileges for UC objects can be managed programmatically using our Terraform provider, especially for teams already using Terraform
- This will pair naturally with the management of the UC objects (Metastore, Catalog, Assignments etc.) themselves.

(If not already using Terraform, maybe now is a good time!)


Documentation > Data governance guide > What is Unity Catalog? >
Automate Unity Catalog setup using Terraform

Automate Unity Catalog setup using Terraform

March 10, 2023

You can automate Unity Catalog setup by using the Databricks Terraform provider. This article shows one approach to deploying an end-to-end Unity Catalog implementation. If you already have some Unity Catalog infrastructure components in place, you can also use this article to deploy additional Unity Catalog infrastructure components as needed.

For more information, see Deploying pre-requisite resources and enabling Unity Catalog in the Databricks Terraform provider documentation.

```
resource "databricks_grants" "sandbox" {
  provider = databricks.workspace
  catalog = databricks_catalog.sandbox.name
  grant {
    principal = "Data Scientists"
    privileges = ["USAGE", "CREATE"]
  }
  grant {
    principal = "Data Engineers"
    privileges = ["USAGE"]
  }
}
```

Discover your data with search and lineage

Why is data lineage important?

Compliance

- **Regulatory** requirements to verify data lineage
- Track the **spread of sensitive data** across datasets

Discovery

- Understand **context** and **trustworthiness** of data before using it in analytics
- **Prevent duplicative** work and data

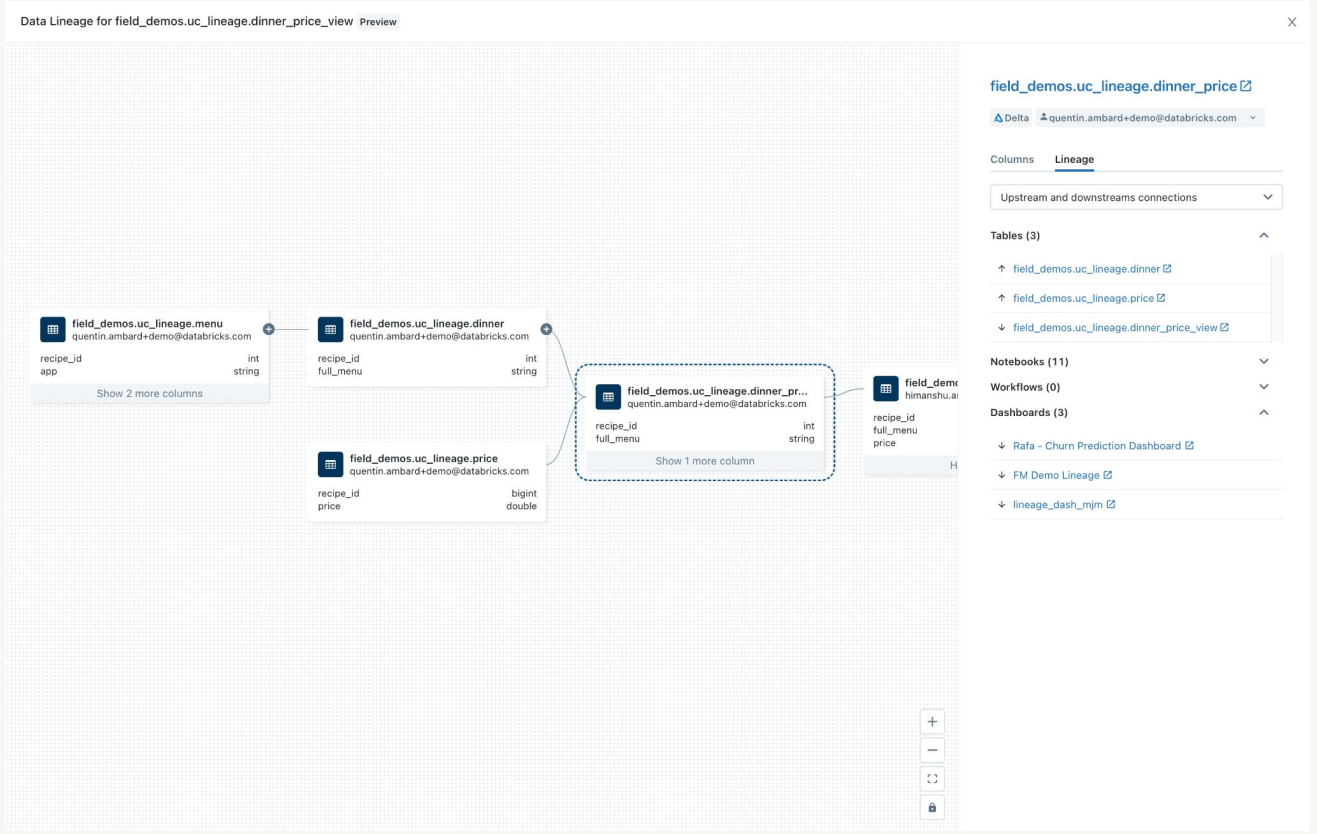
Observability

- Track down **issues / discrepancies** in reports by tracing back the data
- Analyze **impact of proposed changes** to downstream reports e.g. column deprecation

Automated lineage for all workloads

End-to-end visibility into how data flows and consumed in your organization

- Auto-capture runtime data lineage on a Databricks cluster or SQL warehouse
- Leverage common permission model from Unity Catalog
- Lineage across tables, columns, dashboards, workflows, notebooks, files, external sources, and models
- *Recommendation: Upgrade to Unity Catalog!*

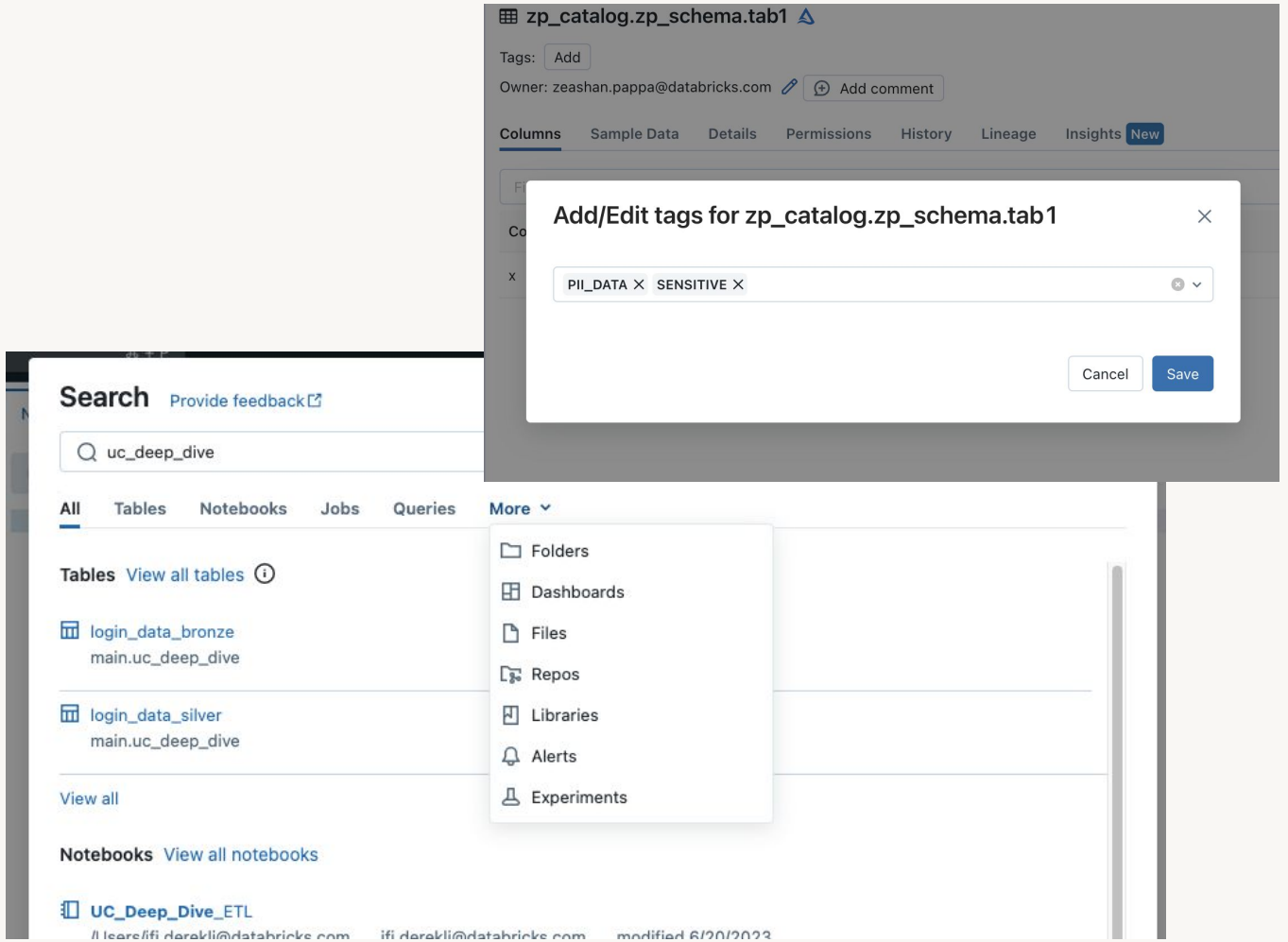


Built-in search and discovery

Accelerate time to value with low latency data discovery

- Unified UI to search for data assets stored in Unity Catalog
- Leverage common permission model from Unity Catalog
- Tag Column, Table, Schema, Catalog objects in UC
- Search for objects on tags

- *Recommendation: Use comments and Tag your Data Assets on Ingest*



Audit your data

System Tables: Object Metadata

Answer questions about the state of objects in the catalog

What tables are in the sales catalog?

```
SELECT table_name
FROM system.information_schema.tables
WHERE table_catalog="sales"
AND table_schema!="information_schema";
```

Who has access to this table?

```
SELECT grantee, table_name, privilege_type
FROM system.information_schema.table_privileges
WHERE table_name = "login_data_silver";
```

`

Who last updated the gold tables and when?

```
SELECT table_name, last_altered_by, last_altered
FROM system.information_schema.tables
WHERE table_schema = "churn_gold"
ORDER BY 1, 3 DESC;
```

Who owns this gold table?

```
SELECT table_owner
FROM system.information_schema.tables
WHERE table_catalog = "retail_prod" AND table_schema =
"churn_gold" AND table_name = "churn_features";
```


PREVIEW

System Tables: Audit Logs

Near-realtime, see who accessed what, and when

Who accesses this table the most?

```
SELECT user_identity.email, count(*)
FROM system.operational_data.audit_logs
WHERE request_params.table_full_name = "main.uc_deep_dive.login_data_silver"
AND service_name = "unityCatalog"
AND action_name = "generateTemporaryTableCredential"
GROUP BY 1 ORDER BY 2 DESC LIMIT 1;
```

Who deleted this table?

```
SELECT user_identity.email
FROM system.operational_data.audit_logs
WHERE request_params.full_name_arg =
"main.uc_deep_dive.login_data_silver"
AND service_name = "unityCatalog"
AND action_name = "deleteTable";
```

What has this user accessed in the last 24 hours?

```
SELECT request_params.table_full_name
FROM system.operational_data.audit_logs
WHERE user_identity.email = "ifi.derekli@databricks.com"
AND service_name = "unityCatalog"
AND action_name = "generateTemporaryTableCredential"
AND datediff(now(), created_at) < 1;
```

What tables does this user access most frequently?

```
SELECT request_params.table_full_name, count(*)
FROM system.operational_data.audit_logs
WHERE user_identity.email = "ifi.derekli@databricks.com"
AND service_name = "unityCatalog"
AND action_name = "generateTemporaryTableCredential"
GROUP BY 1 ORDER BY 2 DESC LIMIT 1;
```


PREVIEW

System Tables: Billing Logs

Understand cost allocation across your data estate

What is the daily trend in DBU consumption?

```
SELECT date(created_on) as `Date`, sum(dbus) as `DBUs Consumed`
   FROM system.operational_data.billing_logs
GROUP BY date(created_on)
ORDER BY date(created_on) ASC;
```

How many DBUs of each SKU have been used so far this month?

```
SELECT sku as `SKU`, sum(dbus) as `DBUs`
   FROM system.operational_data.billing_logs
WHERE
   month(created_on) = month(CURRENT_DATE)
GROUP BY sku
ORDER BY `DBUs` DESC;
```

Which 10 users consumed the most DBUs?

```
SELECT tags.creator as `User`, sum(dbus) as `DBUs`
   FROM system.operational_data.billing_logs
GROUP BY tags.creator
ORDER BY `DBUs` DESC
LIMIT 10;
```

Which Jobs consumed the most DBUs?

```
SELECT tags.JobId as `Job ID`, sum(dbus) as `DBUs`
   FROM system.operational_data.billing_logs
GROUP BY `Job ID`;
```

PREVIEW

System Tables: Lineage Data

Query upstream and downstream sources in one place

What tables are sourced from this table?

```
SELECT DISTINCT target_table_full_name
FROM system.access.table_lineage
WHERE source_table_name = "login_data_bronze";
```

What user queries read from this table?

```
SELECT DISTINCT entity_type, entity_id,
source_table_full_name
FROM system.access.table_lineage
WHERE source_table_name = "login_data_silver";
```

Open data sharing powered by Unity Catalog

Data sharing & collaboration

Accelerate innovation and open new business practices

Delta Sharing

Open sharing between organizations

Databricks Marketplace

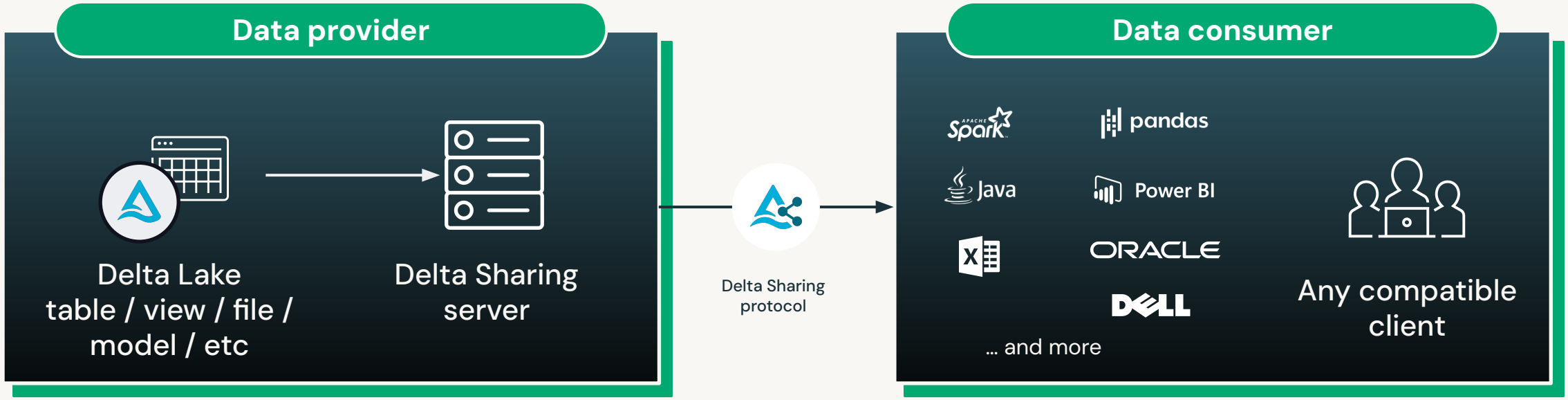
Open Marketplace for all your data, AI, and applications

Databricks Clean Room

Privacy-safe computing and collaboration

Delta Sharing

An open standard for secure sharing of tables, views, files, models, and more



Share cross-platform w/ open protocol

Share data with no replication

Databricks Marketplace

Open marketplace for data, analytics, & AI.

Datasets

Notebooks

Dashboards

ML models

Solutions accelerators

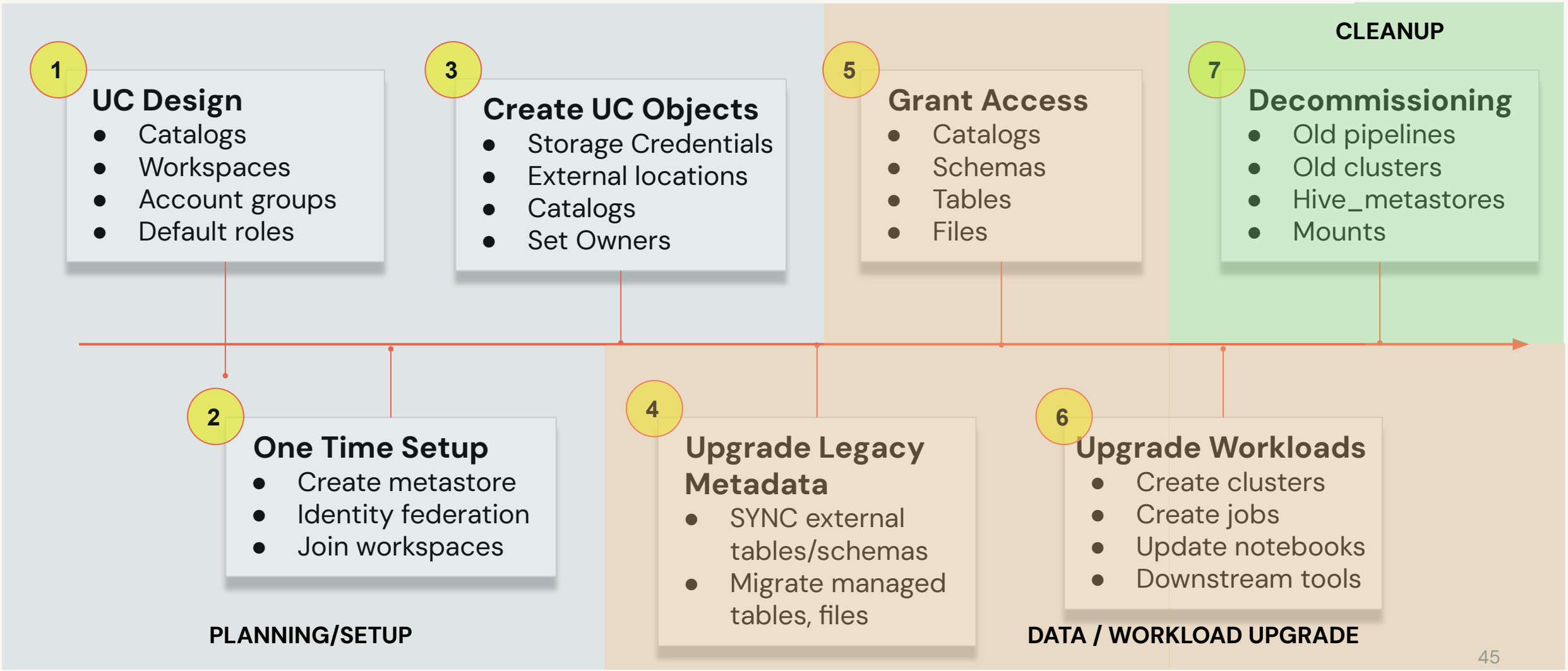
Data applications

Powered by Delta Sharing

Upgrade to Unity Catalog

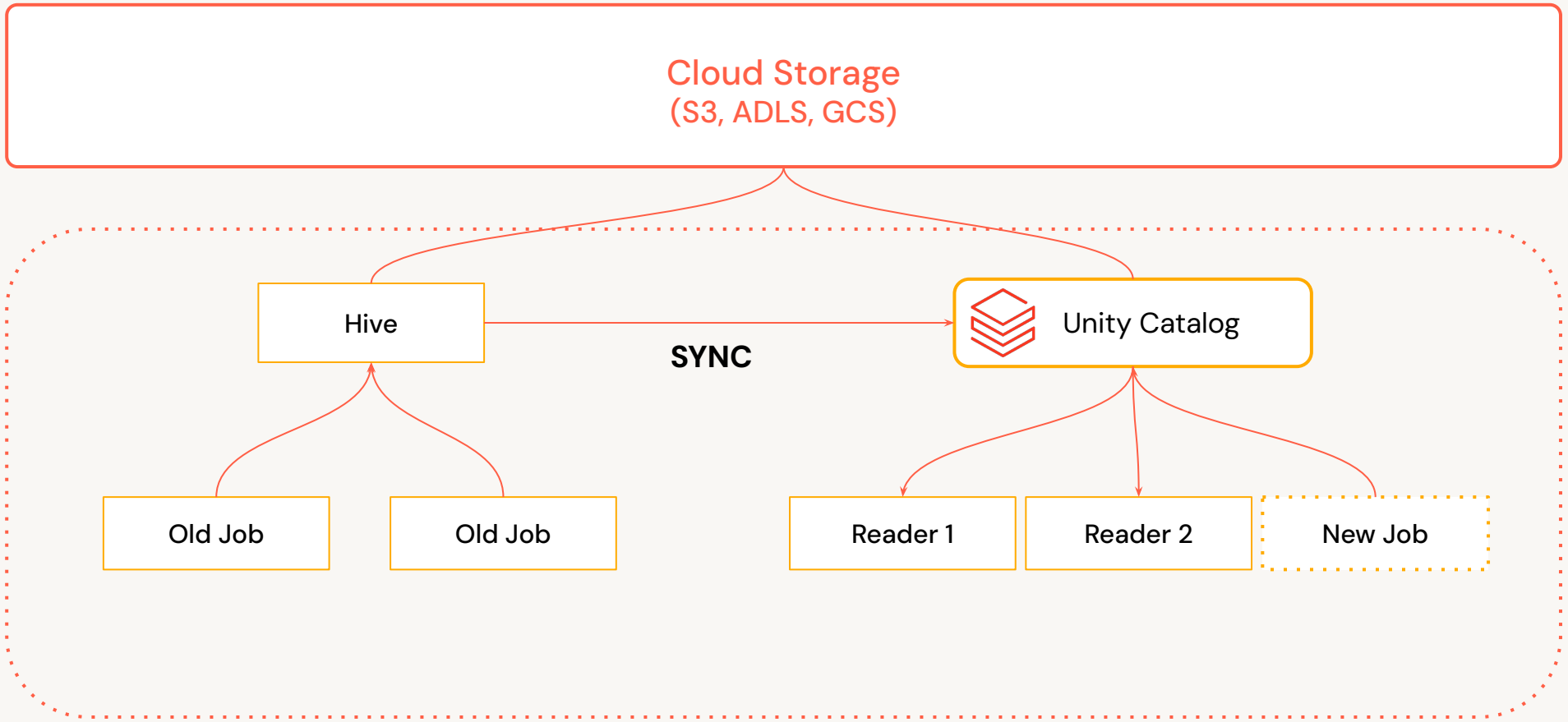
High Level Roadmap to Unity Catalog

Steps to consider for a full upgrade



Keep your jobs

Bring your readers



Upgrading Hive tables to Unity

Managed & External tables - use SYNC command

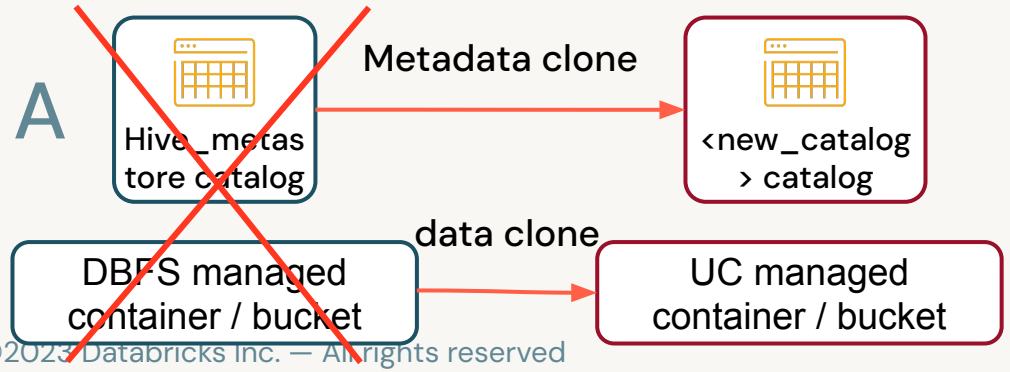
- Run multiple times to pull changes from the hive/glue database into Unity over time
  - Use a job for long term synchronization
- Use the DRY RUN option to test the sync without making any changes to the target table.
- Works on Hive Managed Tables where schema locations are defined.

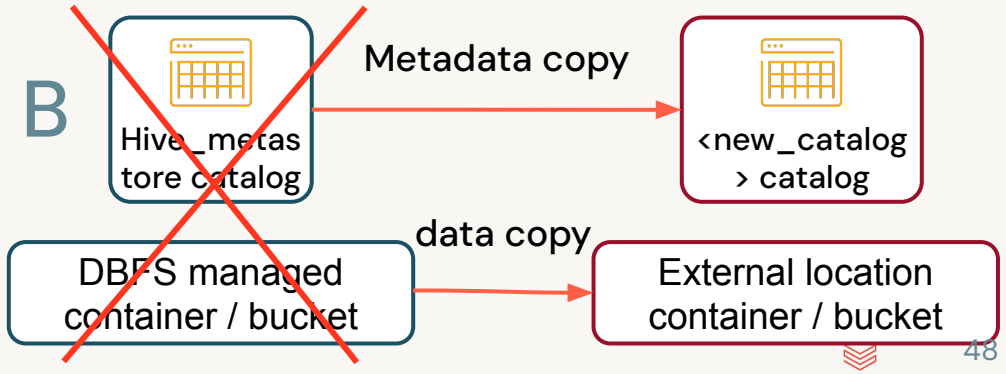
```
SYNC SCHEMA hive_metastore.my_db TO SCHEMA main.my_db_uc DRY RUN

SYNC TABLE hive_metastore.my_db.my_tbl TO TABLE main.my_db_uc.my_tbl
```

Moving Managed Hive tables to Unity

Optional or if in DBFS root - CTAS/CLONE

```
1  // A. Managed Delta -> Managed Delta
2  CREATE TABLE <new_catalog>.<new_schema>.<new_table> CLONE
3  hive_metastore.<old_schema>.<old_table>;
4  // B. Managed non-Delta -> External non-Delta
5  CREATE TABLE <new_catalog>.<new_schema>.<new_table> LOCATION <..> AS SELECT * FROM
6  hive_metastore.<old_schema>.<old_table>;
7  // A+B. Once fully upgraded and tested, drop hive table
8  DROP TABLE hive_metastore.<old_schema>.<old_table>;
```



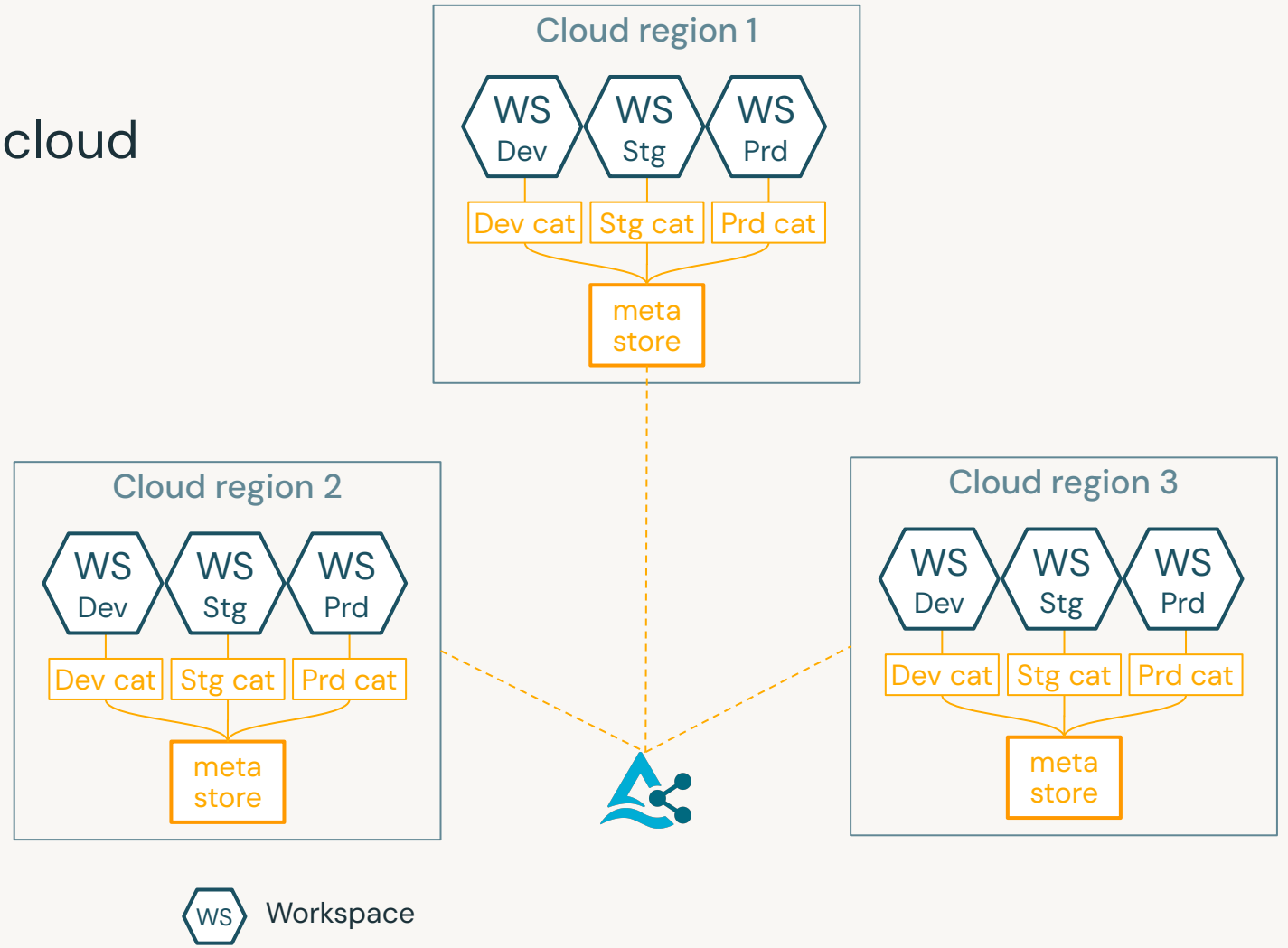


Architectural Patterns

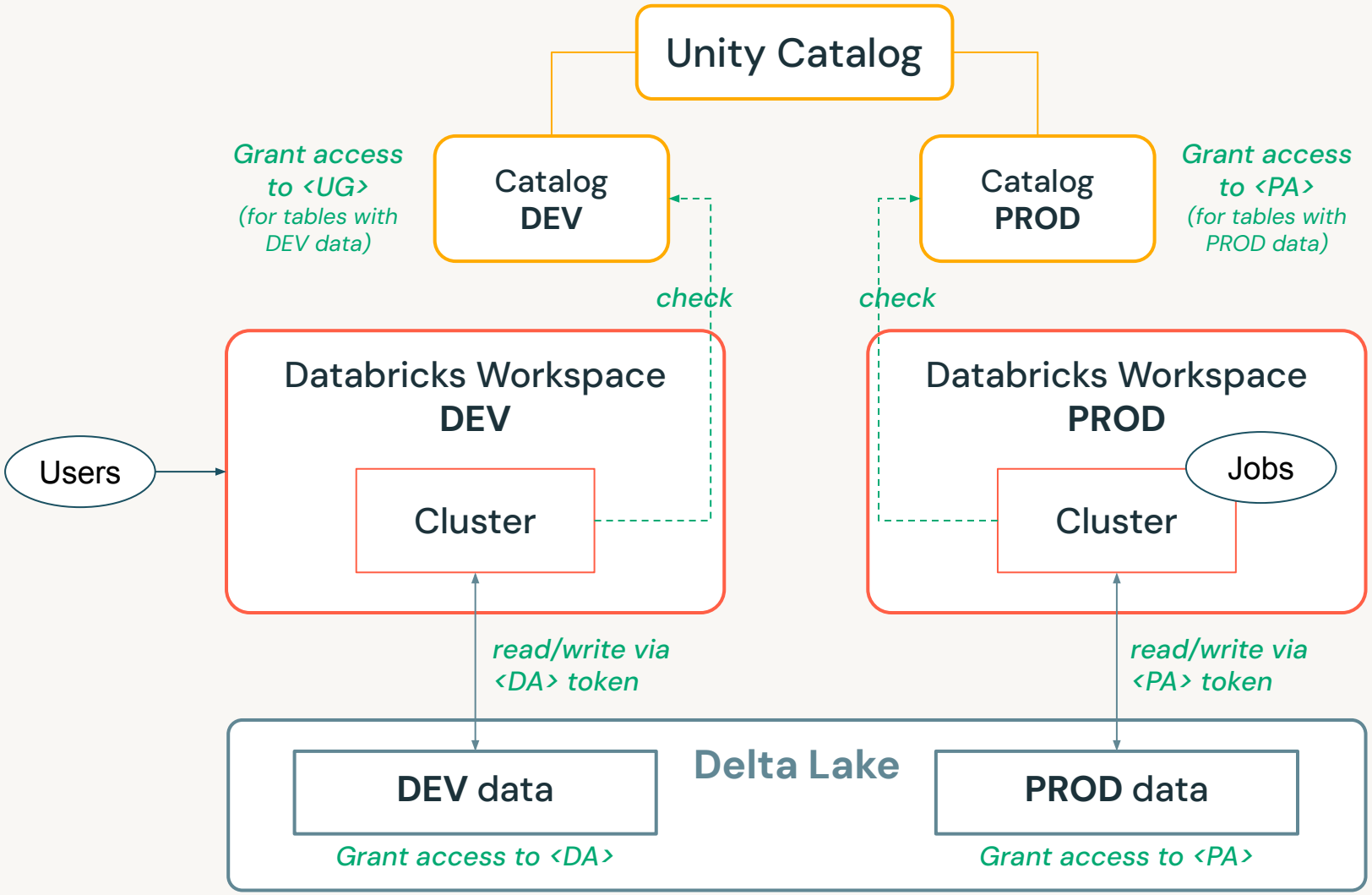
Topology: multi-region / multi-cloud UC

Powered by Delta Sharing

- Metastore boundary = region / cloud (due to latency, cost)
- *Single region Metastore for all SDLC scopes and business units*
- *Leverage workspace-catalog binding as needed*
- *Use Databricks-to-Databricks Delta Sharing between cloud regions and cloud providers*



Software Development Lifecycle setup w/ UC



Note:

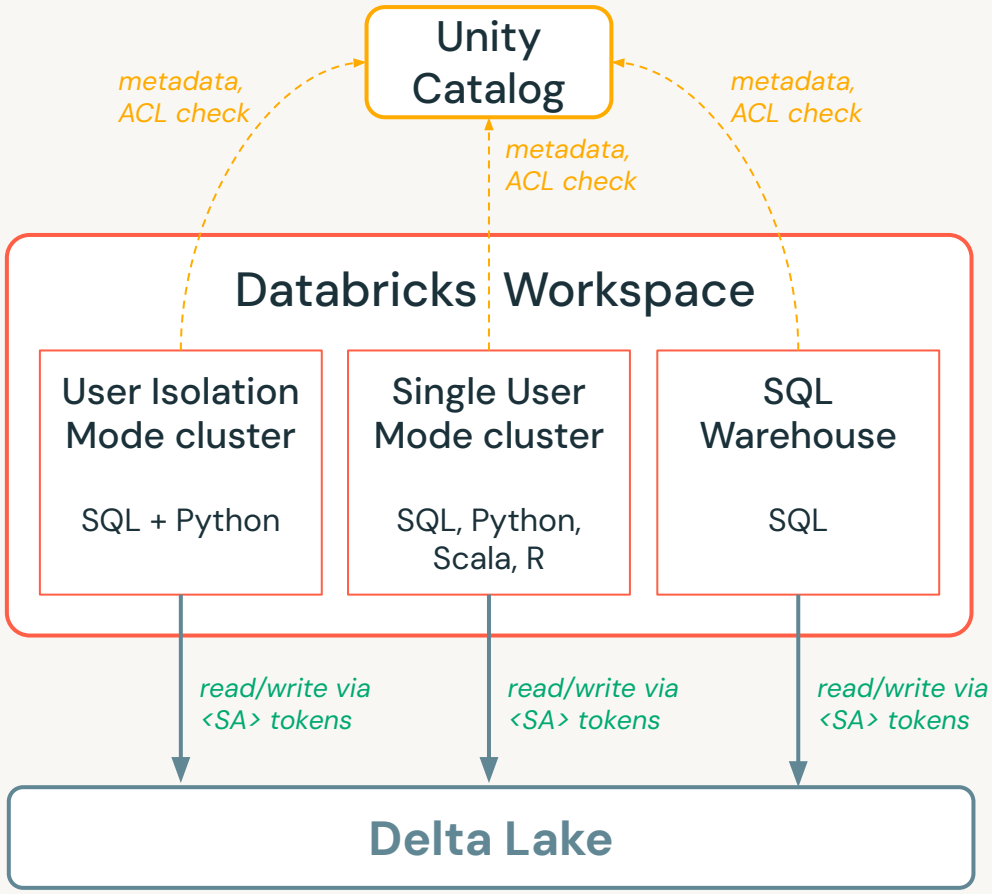
- One of the reasons to have different Workspaces for DEV and PROD is that they could reside in different VNets/VPCs. This is independent of UC, but leads to a setup as it is shown here.

<DA> DEV System Account (Service Principal, Instance Profile, Service Account)

<PA> PROD System Account (Service Principal, Instance Profile, Service Account)

<UG> User Group (Developers, Data Engineers, Data Scientists)

Clusters/endpoints with Unity Catalog



<SA> System Account
(Service Principal or Managed Identity for Azure, IAM Role for AWS, Service Account for GCP)

Standard clusters with User Isolation mode

- Use User isolation mode for general workloads (ETL, data exploration, …) using SQL and Python
- Multiple users can work on the same cluster

Standard clusters with Single User mode

- Use Single User mode for Scala users and for Data Scientists
- ML Runtime with MLflow is supported
- Only the owner can execute code on this cluster, so for notebook collaboration, co-workers can see everything, but they cannot execute cells.
- Limitations
  - Access to views requires access to the underlying table
  - Dynamic views (e.g. for row- /column-level security) are not supported

SQL Warehouse

- Use SQL Warehouses for Business Analysts either using Databricks SQL Editor or external BI tools like Power BI, Tableau, …

Note: Multi-user support for ML Runtimes and MLFlow is under development

Demo

Demo Scenario

HighTech Company - analyzing app user access patterns

John
Data Engineer
Owns pipeline, wants lineage
Upgrade readers, then writers

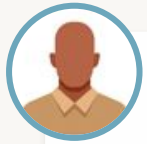
→ **Upgrade, Lineage**

Alice
BI Analyst
Wants to analyze data across platforms
Join with federated query to Snowflake

→ **Federation**

Emily
Unified Governance and Security
Protect PII, comply with GDPR and Reg
Apply RLS/CLS

→ **RLS / CLS**



Marc
Platform Owner
Cares about cost trends and chargeback
Leverage System tables

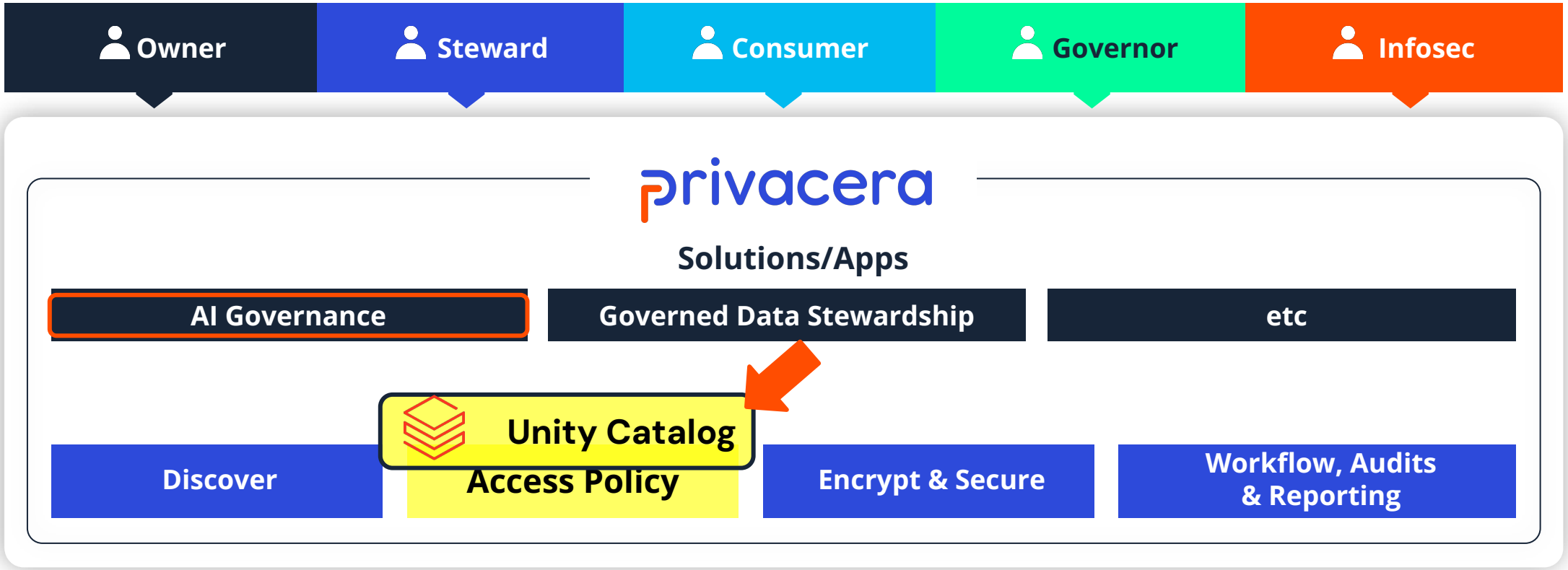
→ **System Tables**

privacera

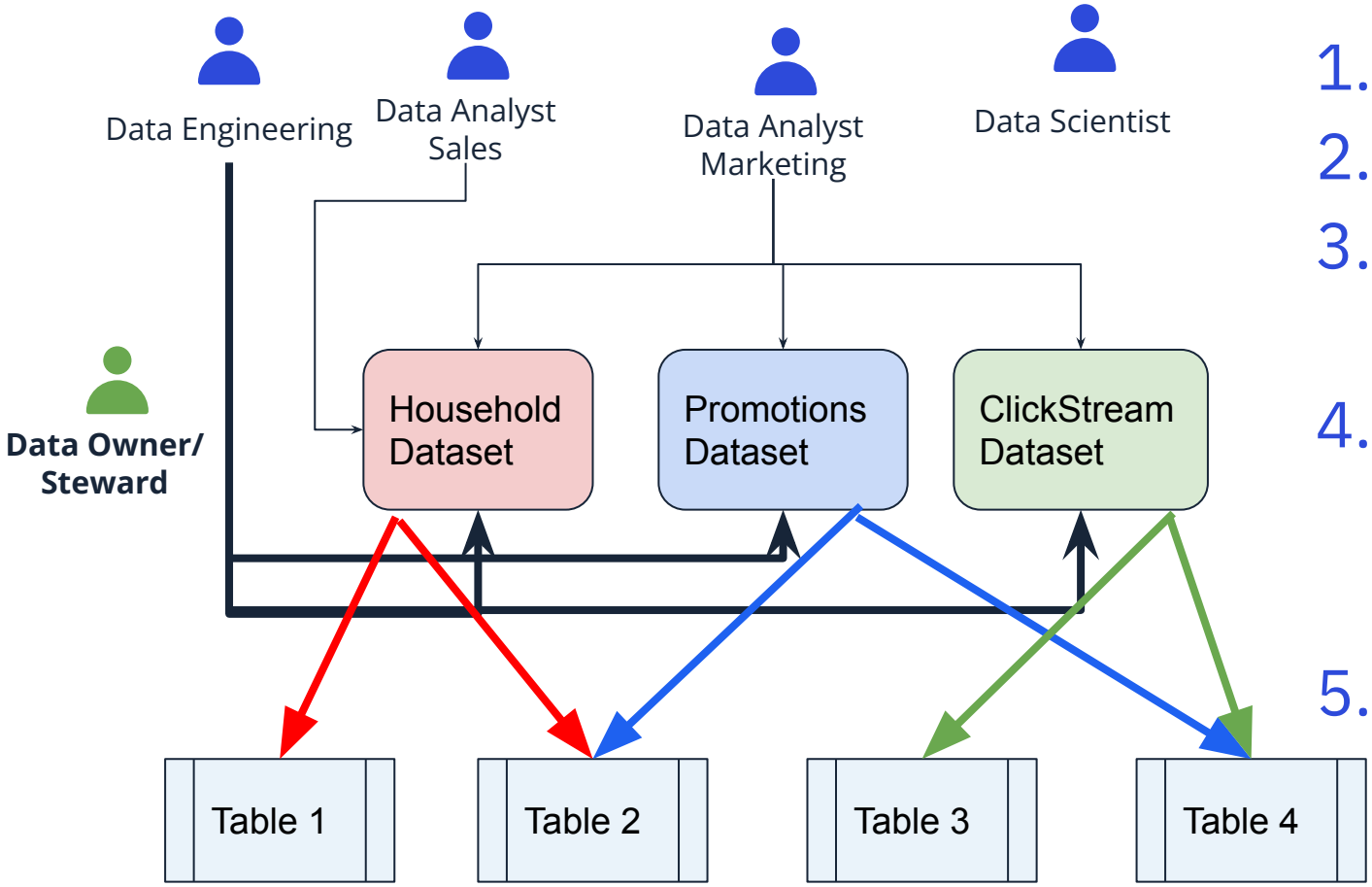
- Extending Unity Catalog
- Athena Federation to Unity Catalog

Don Bosco Durai
Co-founder & CTO - Privacera
Creator of Apache Ranger

Single Pane of Glass Data Access & Security Governance

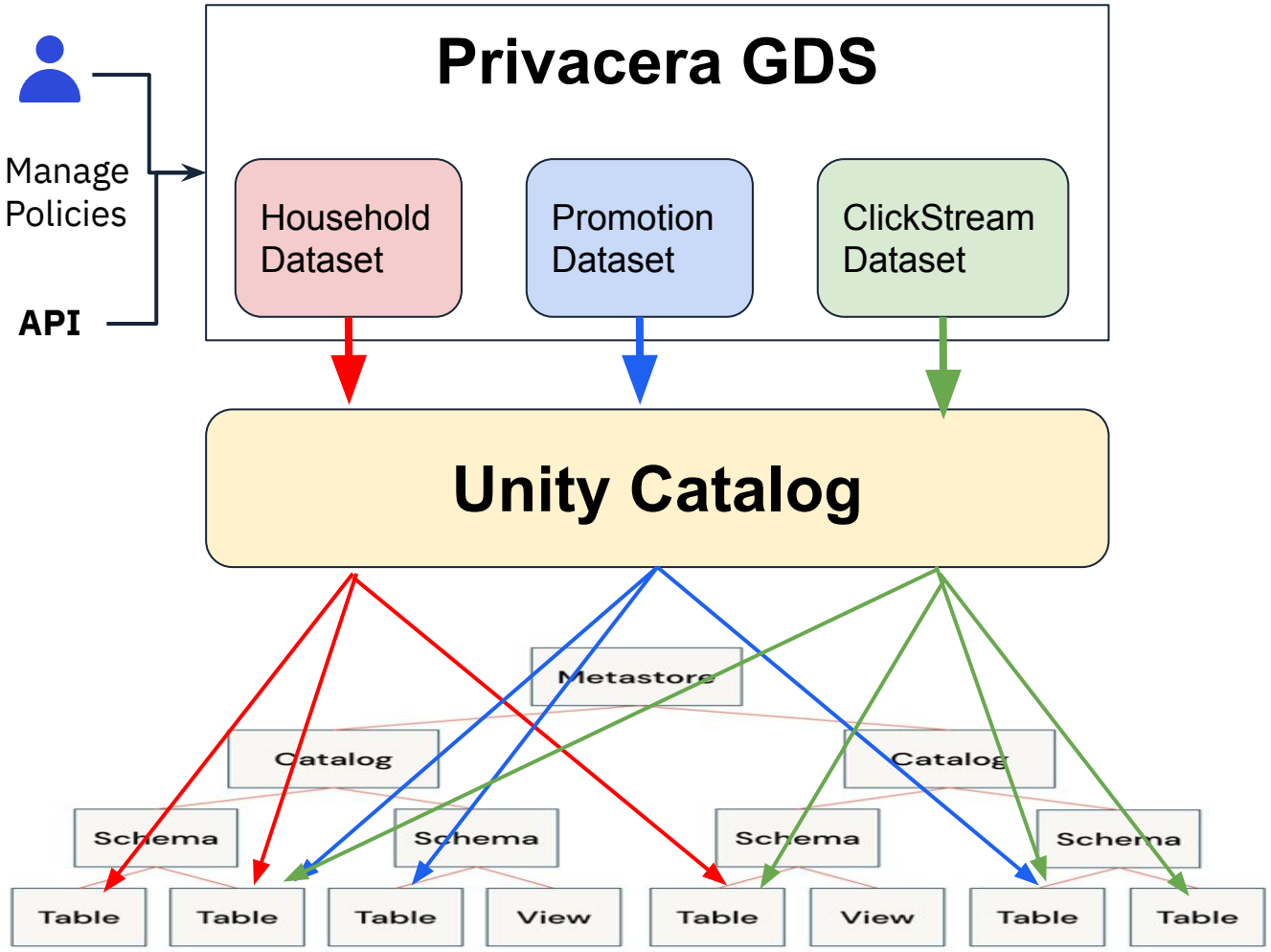


Privacera - Governed Data Stewardship (GDS)



1. Simplified Governance Policies
2. Manage DataSets than Resources
3. Self Service for Data Analysts and Data Scientist
4. Transparent enforcement of fine-grained access control based on purpose
5. Grant permissions for datasets

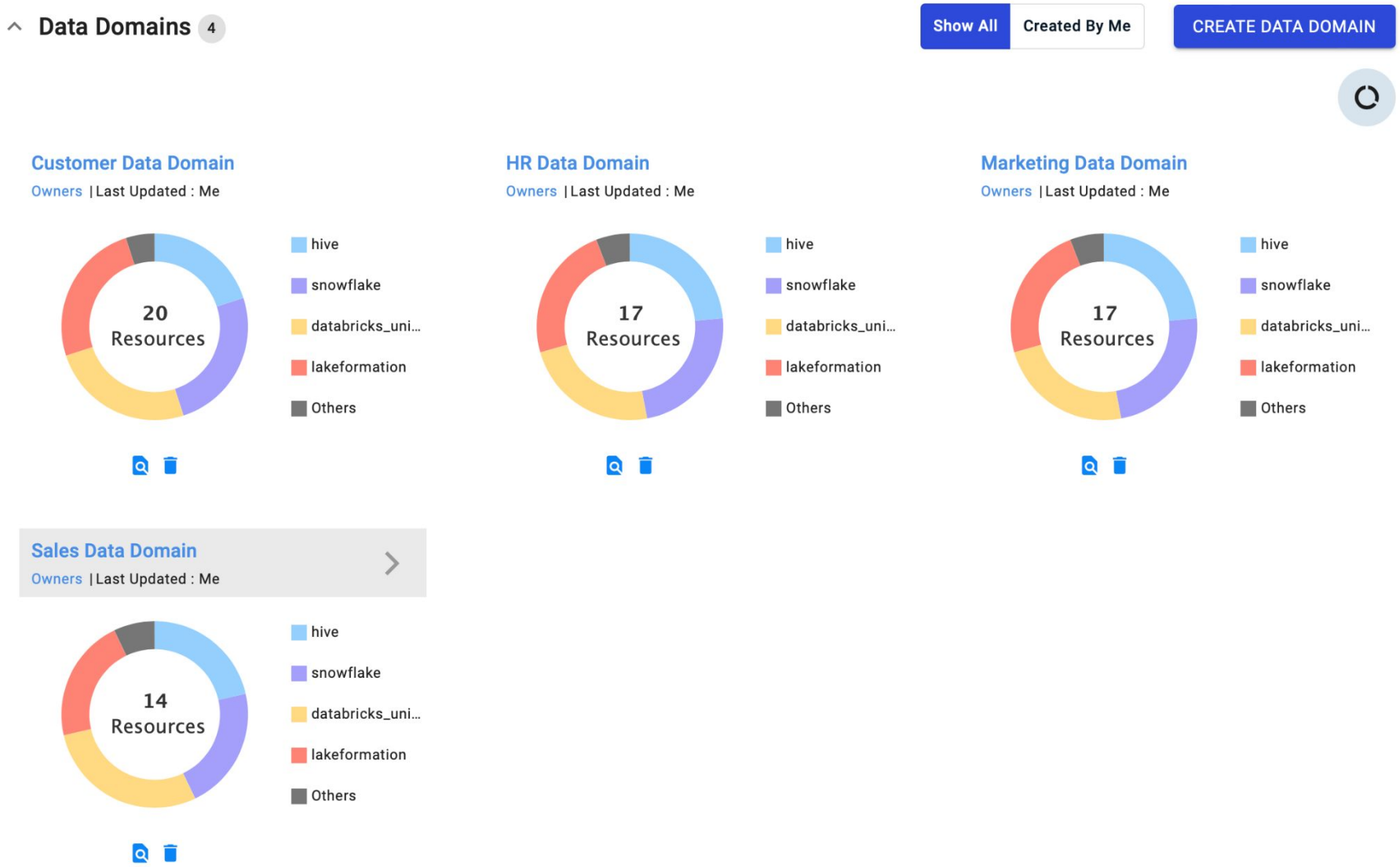
Unity Catalog + Privacera = Enterprise Scale Governance



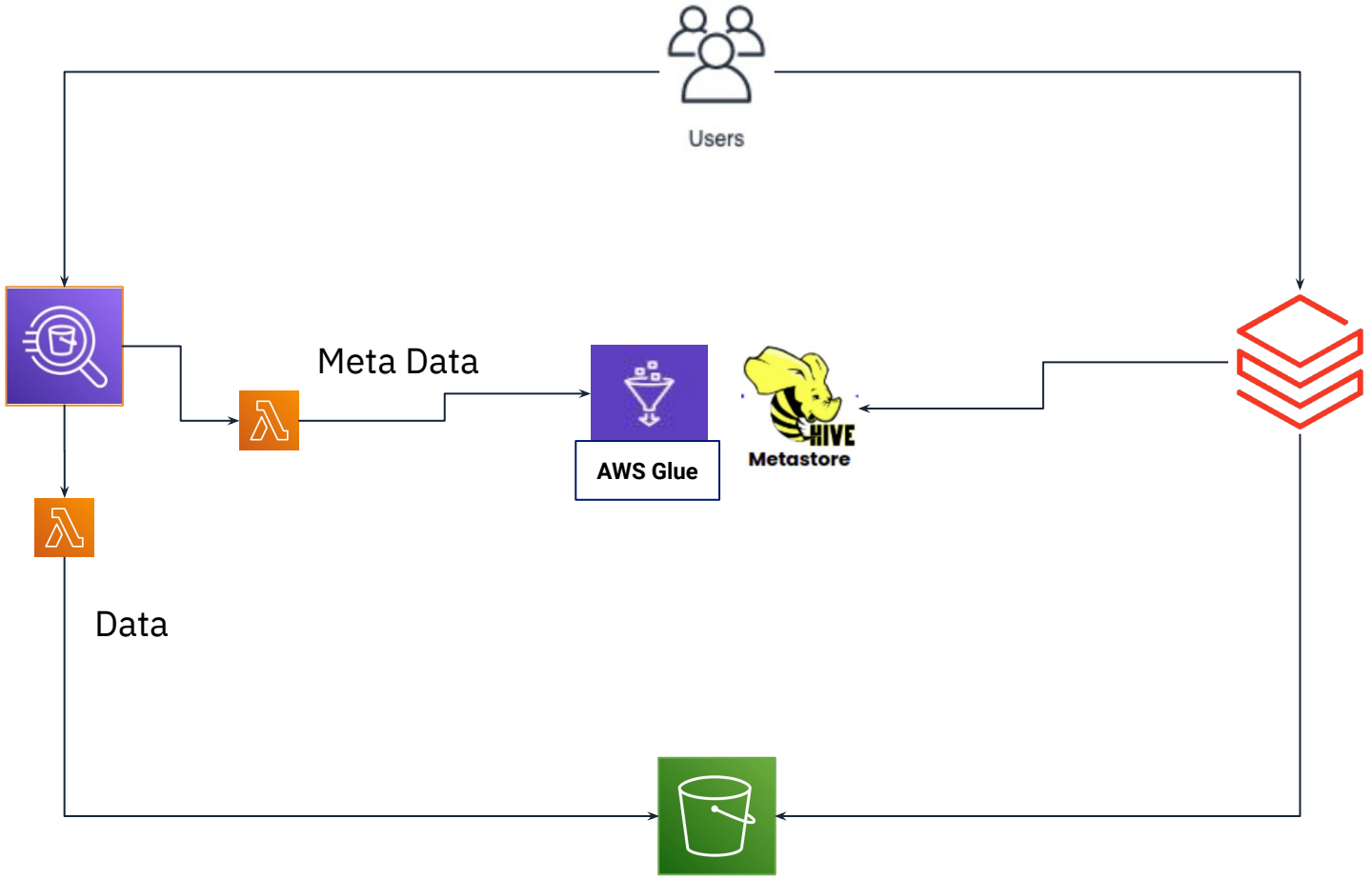
- 10+ Line Of Businesses
- 1 to 50+ of data catalogs
- 10K to 100K+ of tables
- 100 to 1000s scheduled jobs
- Peta Bytes of storage files
- Regional and Global compliance policies
- Governed and secure sharing capabilities

privacera

Governed Data Stewardship - GDS

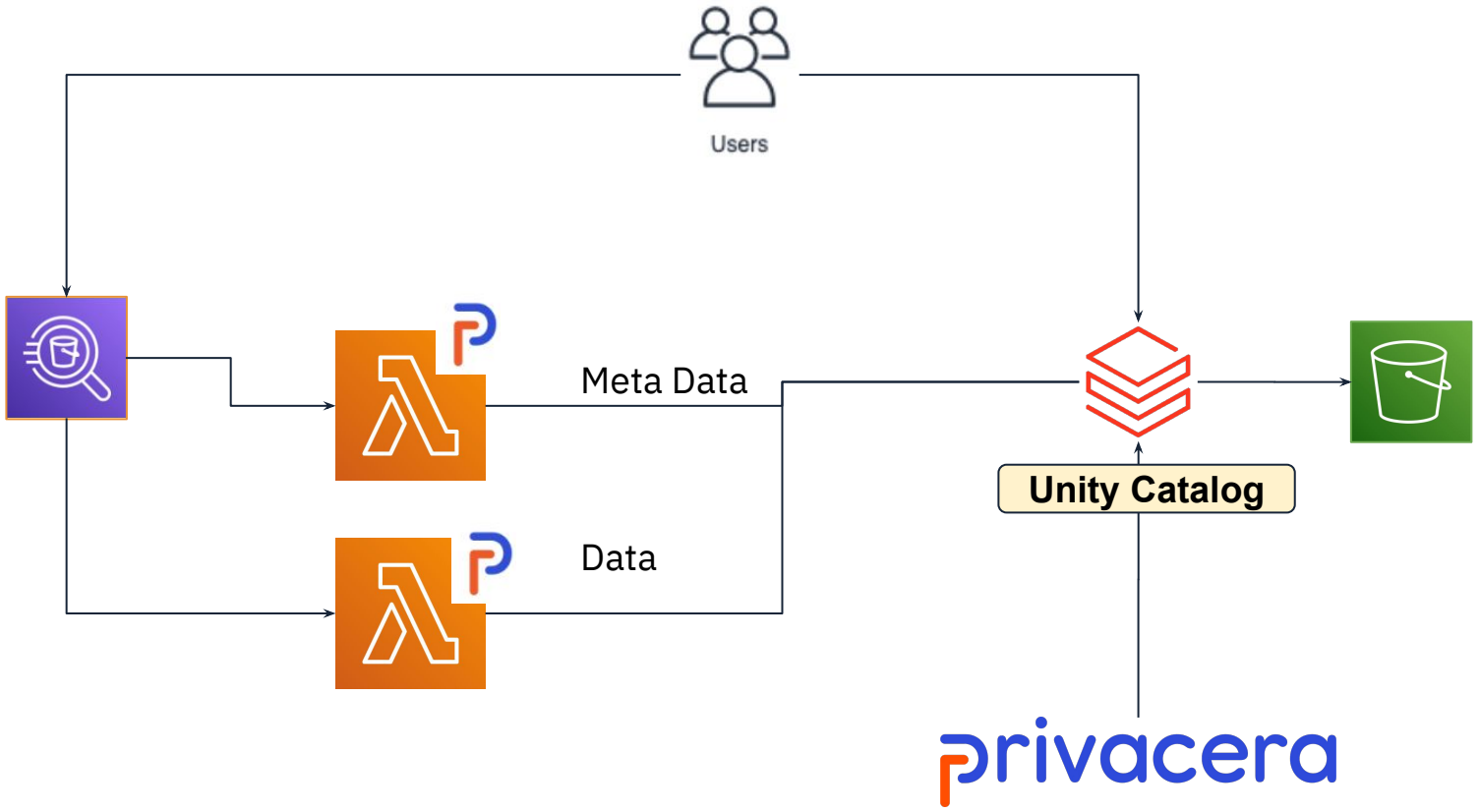


Athena -> HMS/Glue <- Databricks



- Athena and Databricks get meta data from HMS/Glue
- Athena and Databricks get data from S3

Athena -> Databricks Unity Catalog



- Athena using lambda gets metadata and data from Unity Catalog
- Privacera pushes policy to Unity Catalog
- Privacera's Lambda plugin enforces the same policy in Athena

Learn more at the summit!

Databricks Events App

Tells us what you think

- We kindly request your valuable feedback on this session.
- Please take a moment to rate and share your thoughts about it.
- You can conveniently provide your feedback and rating through the **Mobile App**.

What to do next?

- Discover more related sessions in the mobile app!
- Visit the Demo Booth: Experience innovation firsthand!
- More Activities: Engage and connect further at the Databricks Zone!

Get trained and certified

- Discover the Learning Hub at the Databricks Zone!
- Get certified at the event!
- Visit our Databricks Academy website for more training, courses and workshops! www.databricks.com

Questions?