Unity Catalog Upgrade

Account Level Configuration

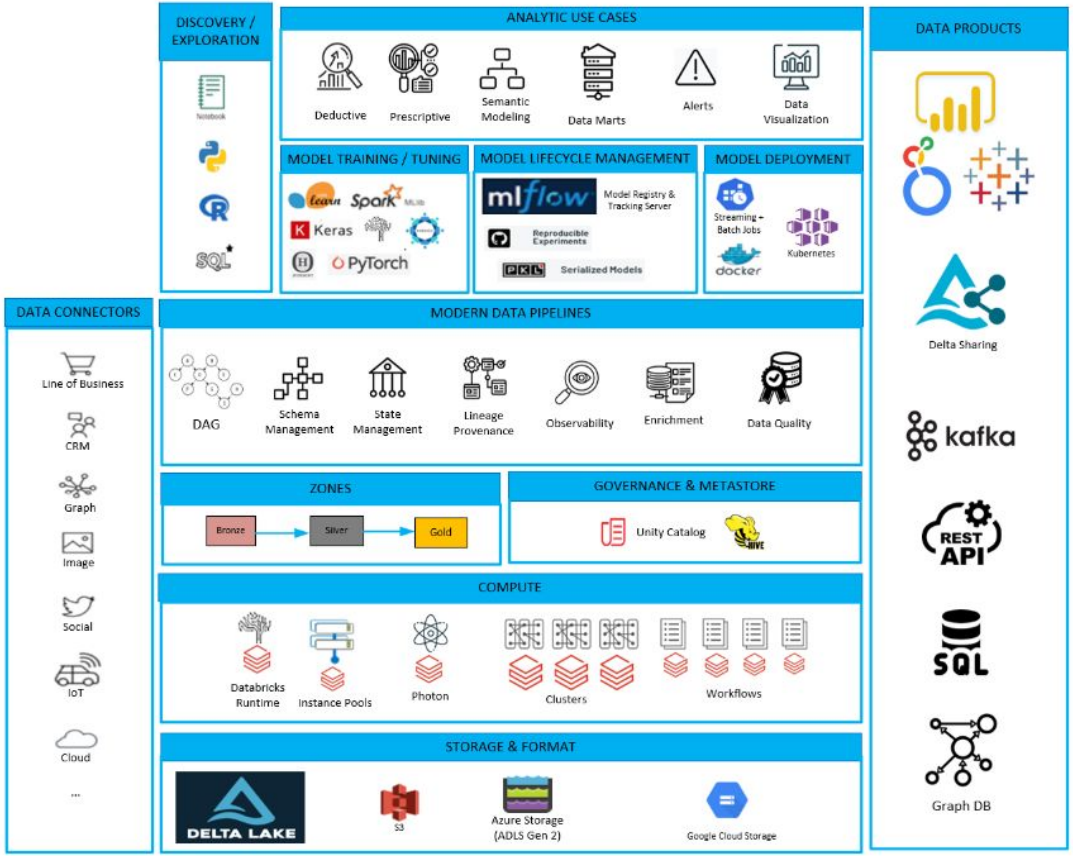
OBJECTIVES

The purpose of today's discussion is to go over best practices around Databricks Account Level Configuration and plan the following:

- Databricks Workspace Planning, Environments
- Designing the Unity Catalog Metastore(s)
- Cloud Storage, Credentials, External Locations, Volumes
- Data Federation Requirements
- Auditing

Record the decisions made and incorporate into diagrams and design documents for the project deliverables

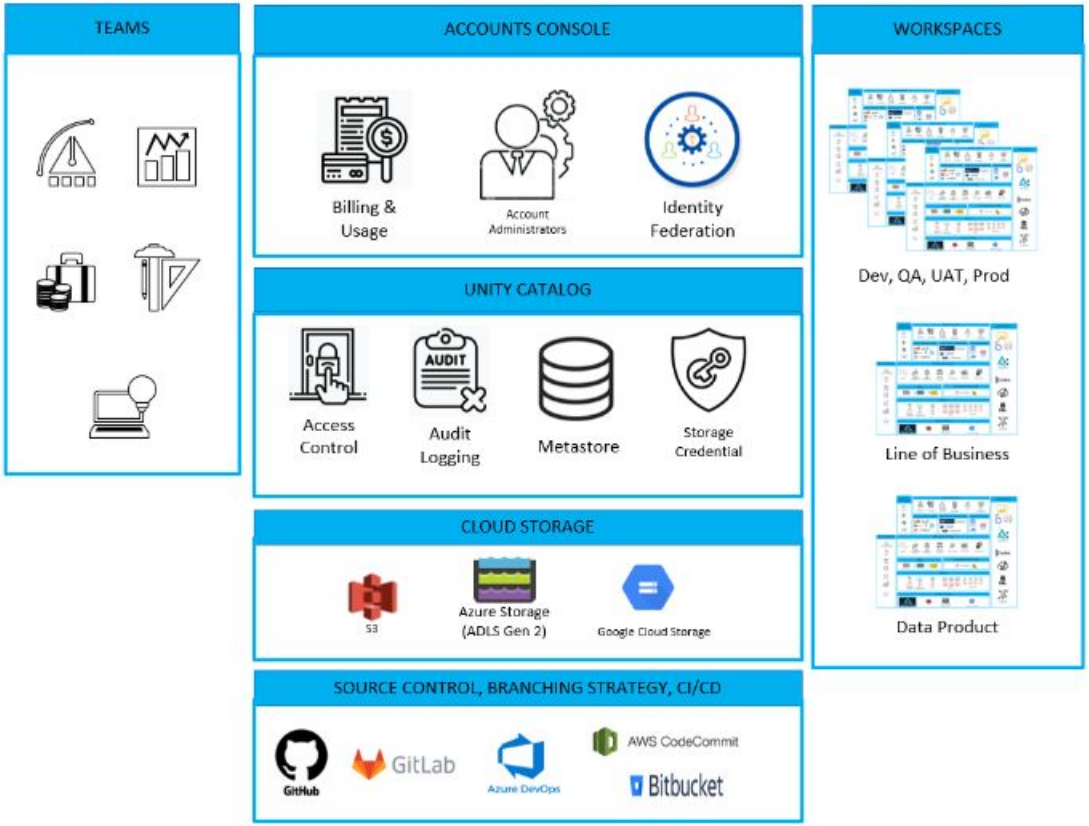
The Databricks Lakehouse



- Unified Data & AI Platform combining the strengths of Data Lakes and Warehouses.
- Support any Data and Processing workloads
- Build any Data Product
  - AI Models & Algorithms
  - Data Visualization Tools
  - Data Sharing Platforms
  - Data APIs
  - Data Management Tools
  - Data-driven Products and Services
  - Data Monetization Platforms
- Scalable, Flexible, Agile
- Make more informed decisions and achieve better business outcomes

The Databricks Lakehouse allows you to INNOVATE.

Databricks in Production

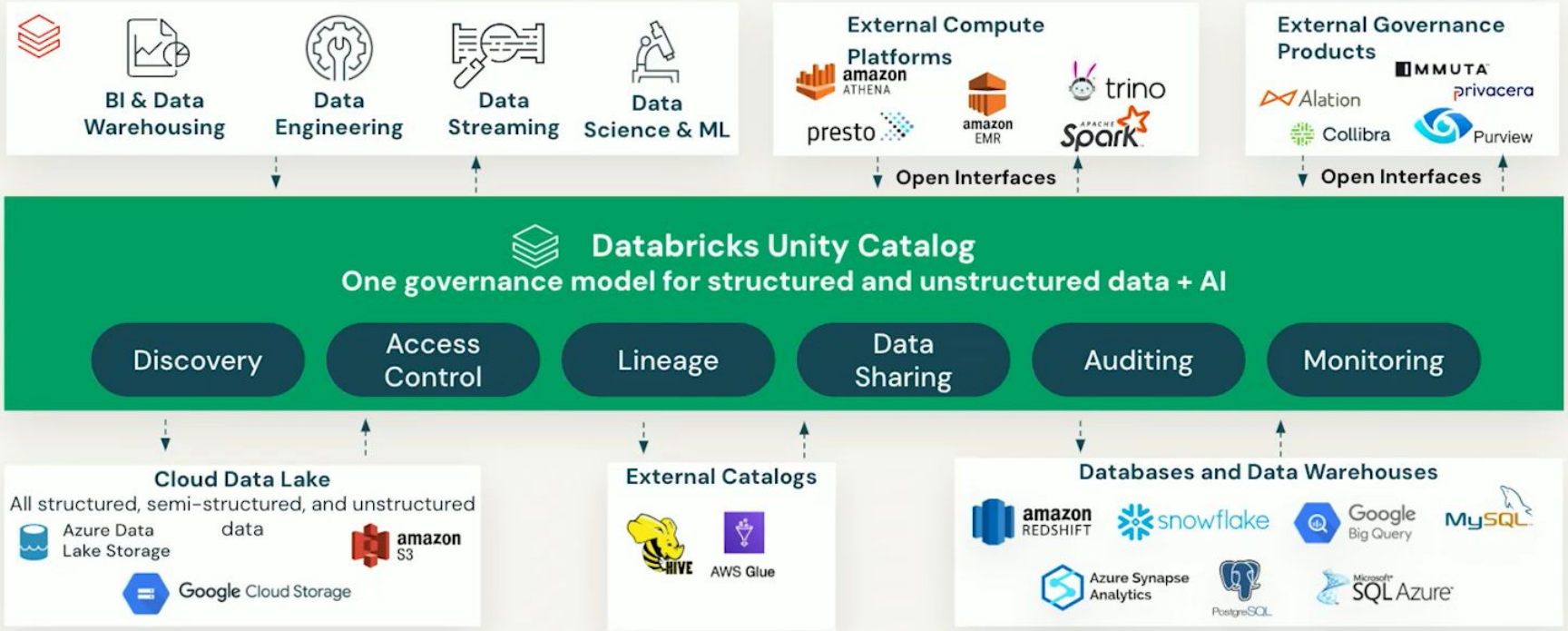


- Support multiple Teams and Environments
- Deliver Data Products
- Governance
- Code Quality
- Data Quality
- Security
- Administration & Monitoring
- Scalability
- Cost Management
- Continuous Integration & Deployment (CI/CD)
- Regulatory Compliance

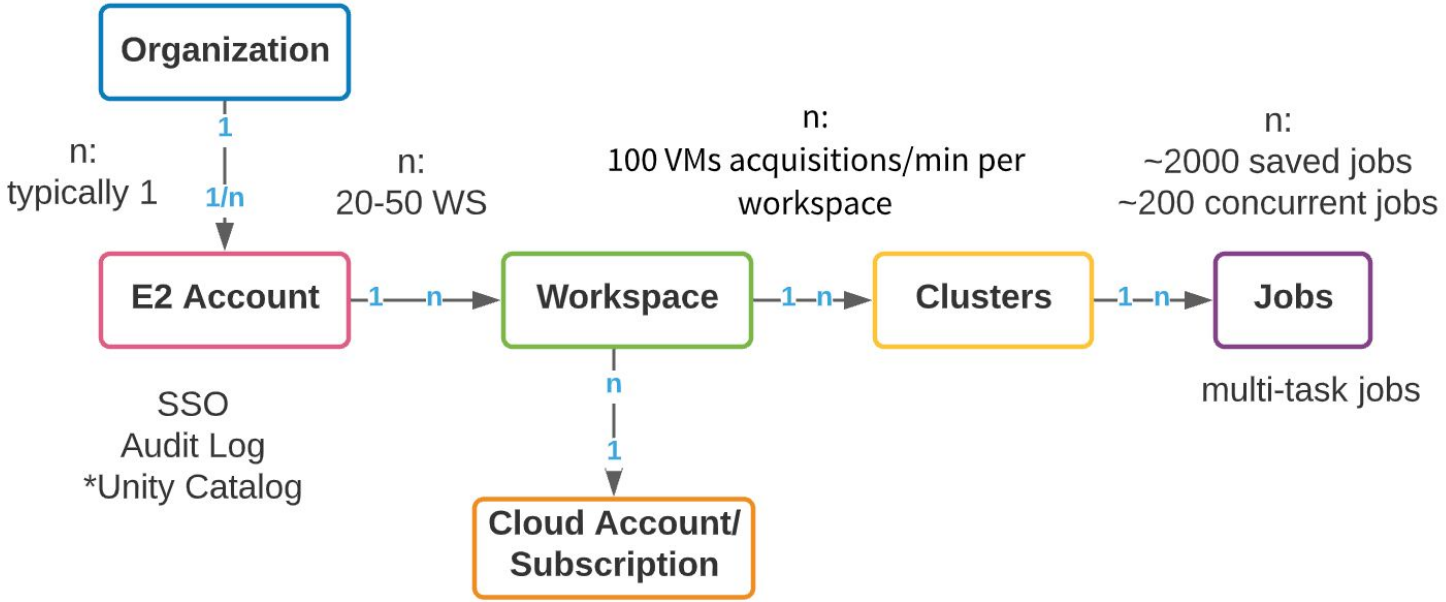
The Production Databricks Lakehouse allows you to DELIVER VALUE in a reliable, secure and repeatable way.

Unity Catalog

Unified governance for Data & AI



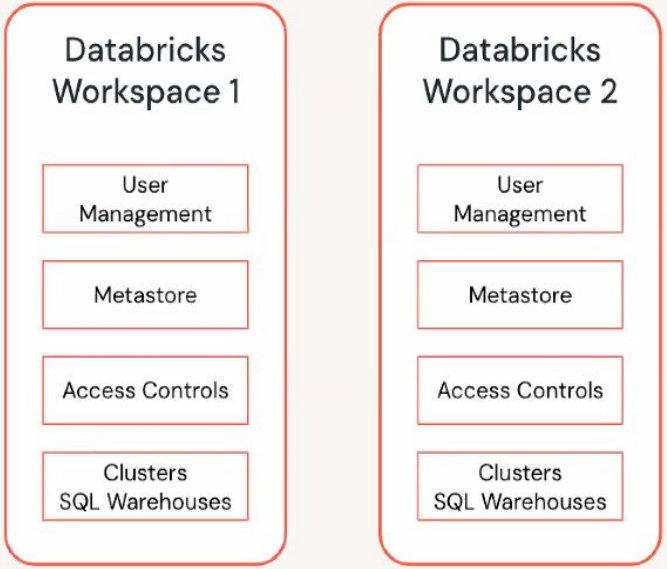
Databricks High Level Object Model



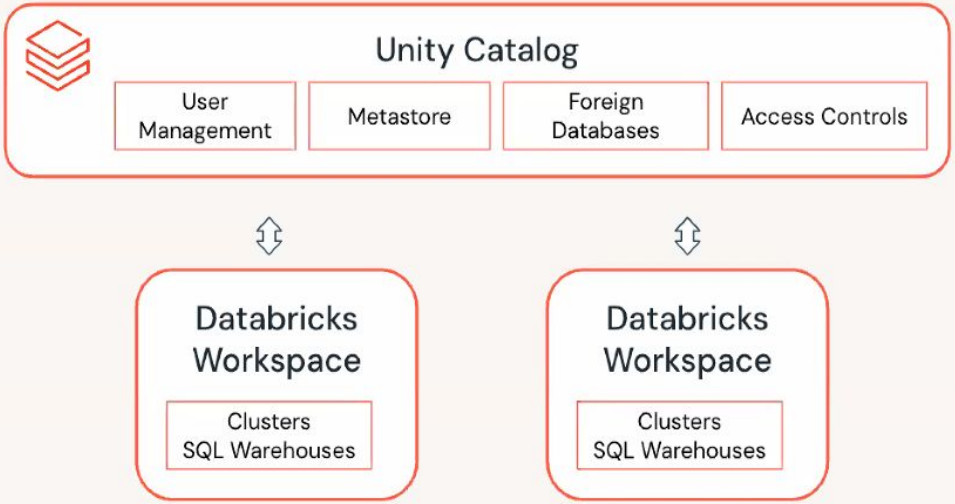
All your metadata, in one place

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

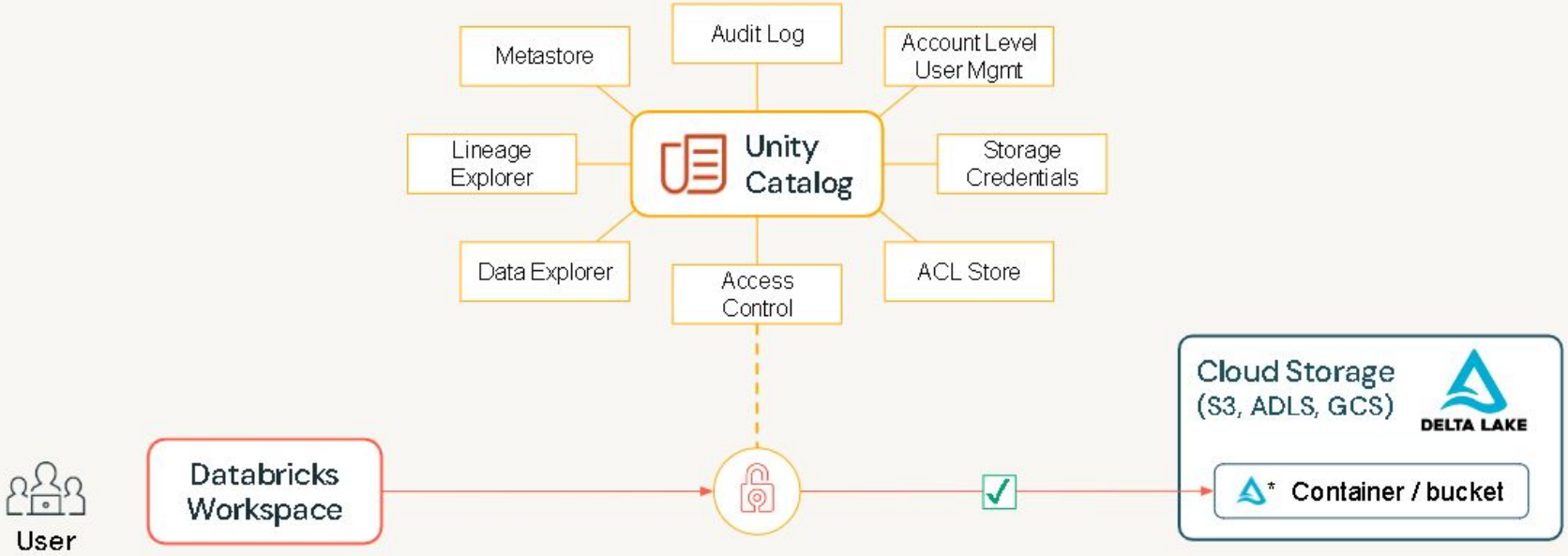
Without Unity Catalog



With Unity Catalog



Unity Catalog –Architecture



* Unity Catalog will support any data format (table or raw files)

Some Important Definitions

Databricks Account: A Databricks account is the top-level container for all your Databricks resources, including workspaces, clusters, jobs, and notebooks.

Workspace: An isolated environment for accessing all of your Databricks assets. A workspace organizes objects (notebooks, libraries, and experiments) into folders, and provides access to data objects and computational resources. Connected to a meastore for metadata.

Metastore: A regional service that contains catalogs and metadata. Used to manage metadata for file and table sources.
Pre-Unity Catalog analog: Hive Metastore

Catalog - A logical collection of schemas and tables within a metastore. Provides isolation and grouping of data.

Schema - A namespace within a catalog that contains tables, views and functions. Provides an additional layer of isolation.

Managed Table - A table created in Unity Catalog without specifying an external location, so the data is stored in the managed location. Metadata is managed by Unity Catalog.
Pre-Unity Catalog analog: Hive Metastore Managed Tables (DBFS)

External Table - A table defined in Unity Catalog by specifying an external location path where the data is stored outside Unity Catalog's managed storage. Metadata still managed by Unity Catalog.
Pre-Unity Catalog analog: Hive Metastore External Tables

Credential: An abstraction of cloud provider access credentials like IAM roles that allows Unity Catalog to securely access storage locations. Encapsulates an authentication and authorization mechanism for accessing data stored on your cloud tenant.
Pre-Unity Catalog analog: Spark Configuration, Instance Profiles.

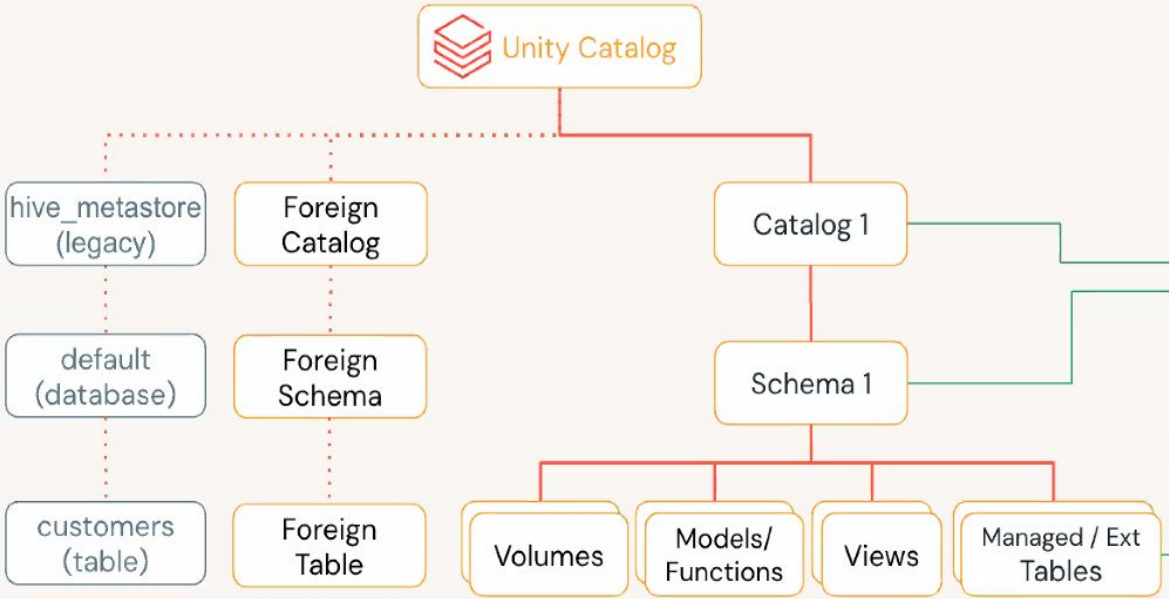
Managed Location: A default storage location in an S3 bucket in your AWS account that is associated with a metastore, catalog, or schema when they are created. Managed tables and managed volumes store data and metadata files in managed storage.
Pre-Unity Catalog analog: DBFS (Databricks File System) Paths.

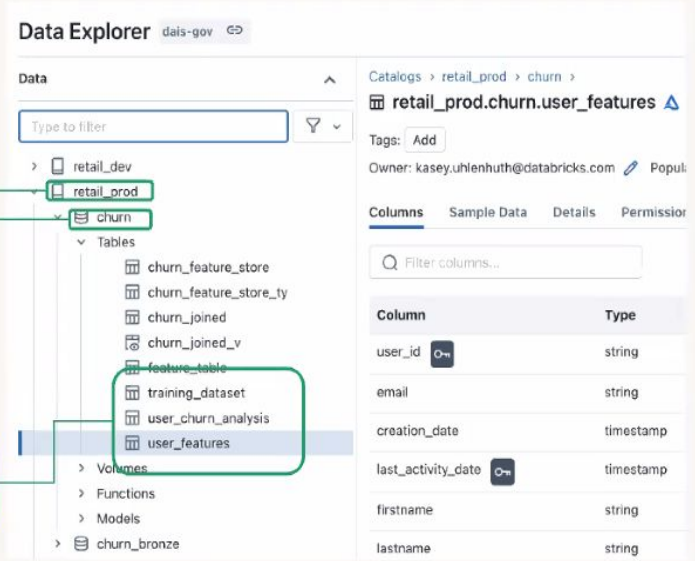
External Location: Combines a cloud storage path with a storage credential that authorizes access to the cloud storage path. It is used for creating external tables only when you require direct access to the data outside of Databricks clusters or Databricks SQL warehouses.
Pre-Unity Catalog analog: Mount Points, s3a paths, Spark Conf.

Volume - Enables accessing files in a storage location through POSIX commands while applying Unity Catalog governance. Can be created on managed or external locations.
Pre-Unity Catalog analog: DBFS (Databricks File System) Paths.

Governed namespace across file and database sources

Access legacy metastore and foreign databases powered by Lakehouse Federation



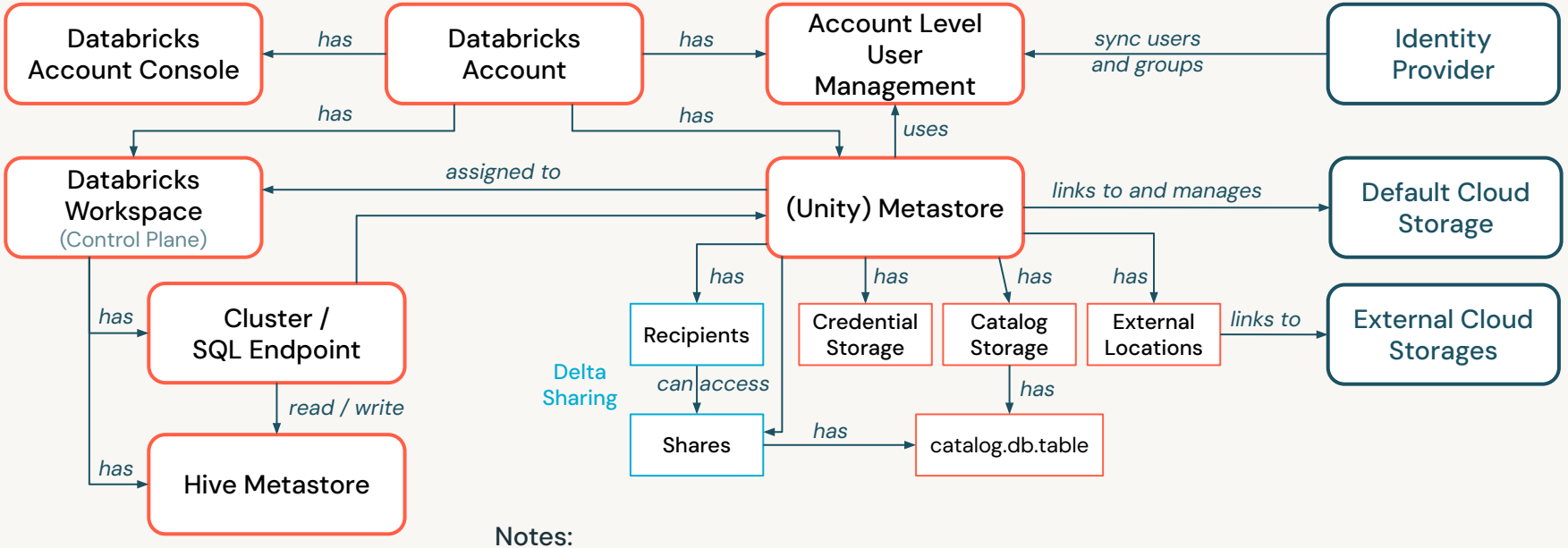



```
SELECT * FROM main.paul.red_wine;  -- <catalog>.<database>.<table>

SELECT * FROM hive_metastore.default.customers;

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

Object Relations with Unity Catalog (UC)

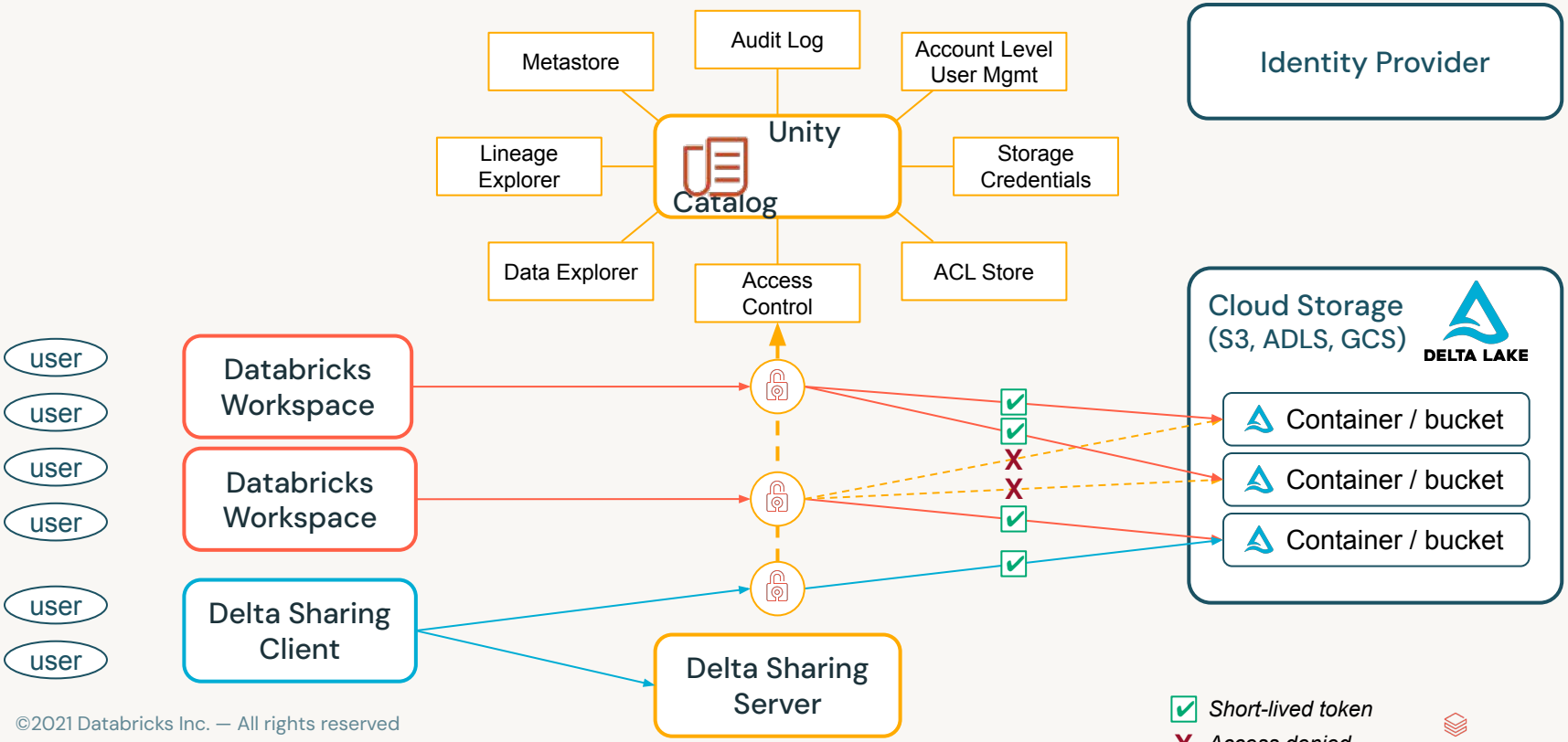


Notes:

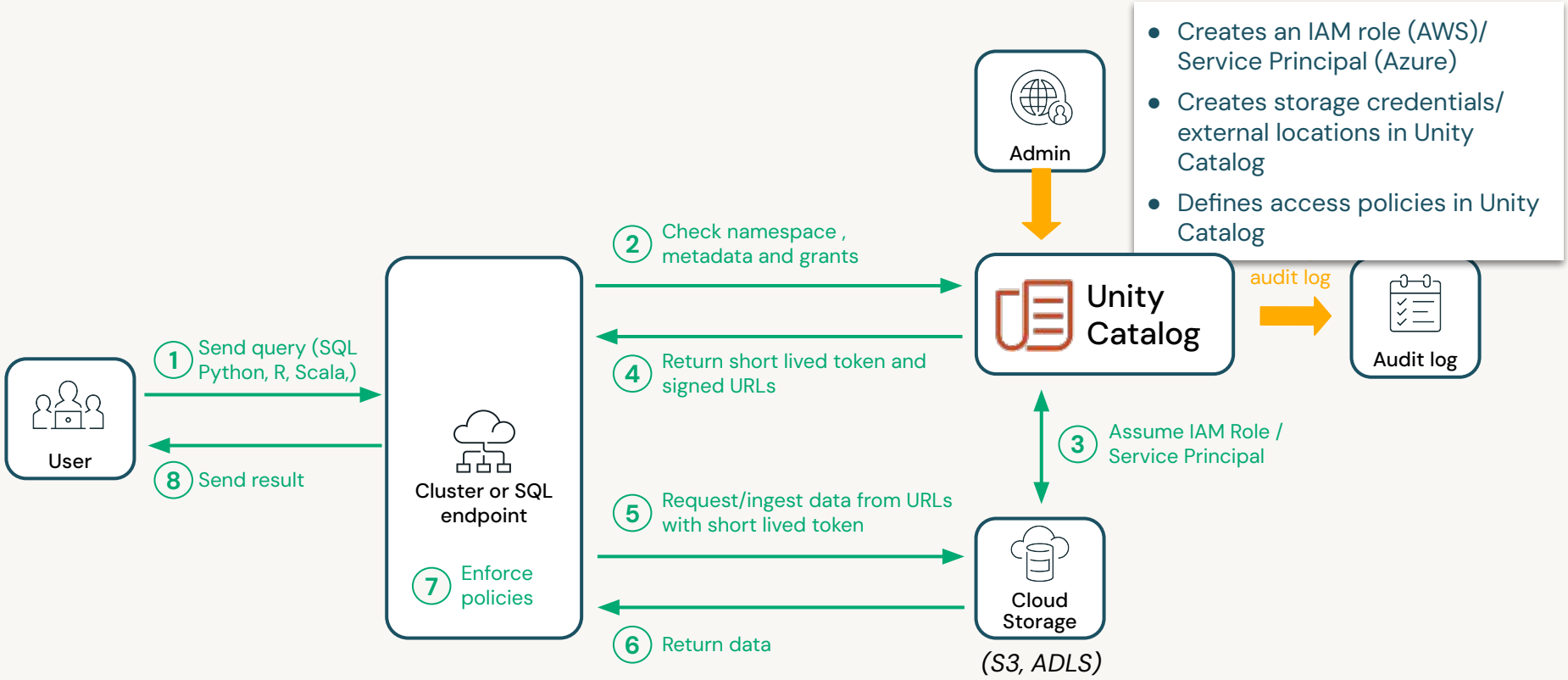
- Using Unity Catalog is optional. Workspaces can still use a Hive Metastore only
- There can be more than one Unity Metastore (UC) per Databricks Account (e.g. for regional isolation or for isolation of lines of business)
- Every workspace can only attach to one UC Metastore, however one Unity Metastore can be assigned to several Workspaces

Databricks Unity Catalog

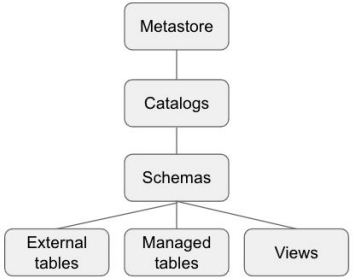
Centralized Governance

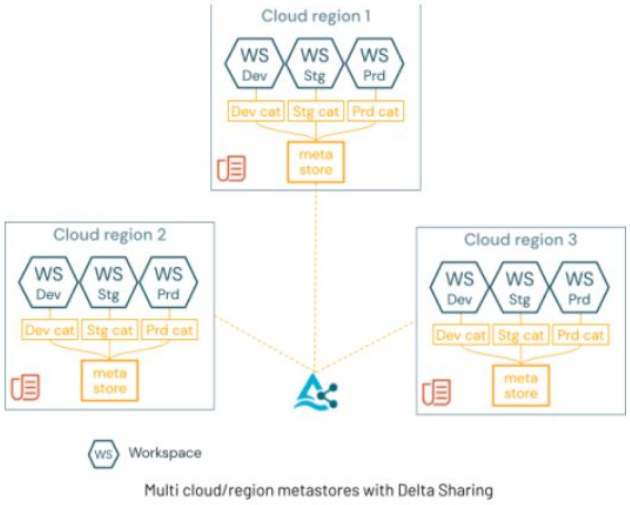


Life of a query with Unity Catalog



Unity Catalog Metastore





Multi cloud/region metastores with Delta Sharing

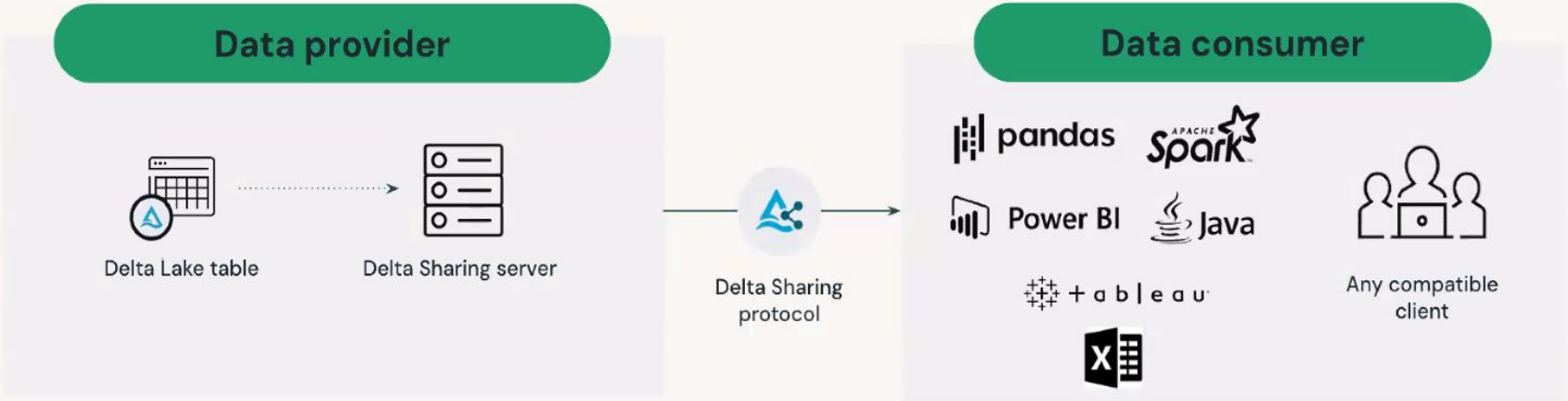
- Top-level container of objects in Unity Catalog
- Stores data assets (tables & views) and the permissions that govern access to them
- Metastores are assigned to Databricks Workspaces to control which workloads use each metastore
- Unity Catalog allows for a single metastore per region and only allows you to use that metastore in its assigned region.
- If you have multiple regions using Databricks, you will have multiple metastores
- To share data between metastores, use Databricks-managed Delta Sharing (a.k.a. Databricks-to-Databricks Delta Sharing)
- Each metastore is configured with a root storage location, used for managed tables

Metastore Recommendations

- Ensure that no users have direct access to the UC root storage location. This could allow a user to bypass access controls in UC and disrupt auditability.
- Do not reuse a bucket that is your current (or previous) DBFS root file system.

Delta Sharing

An open standard for data sharing

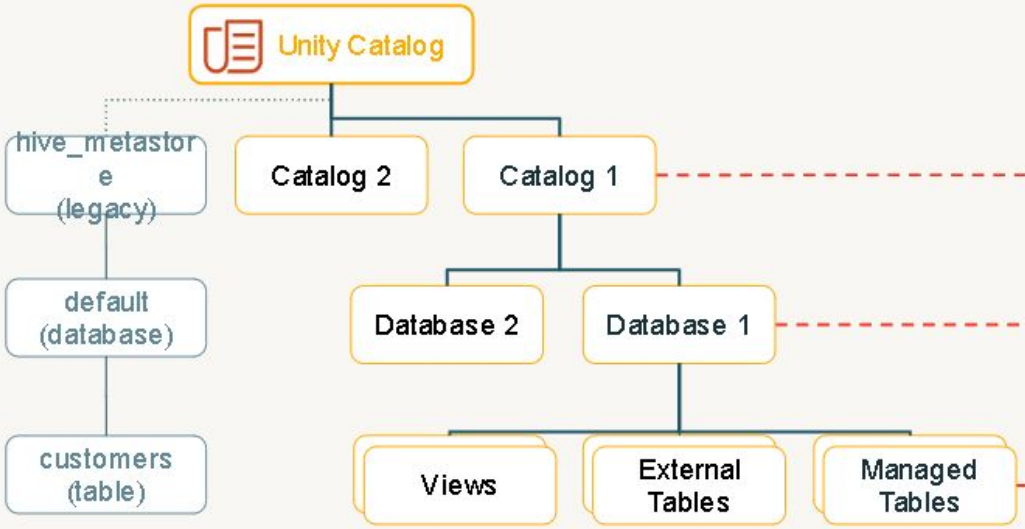


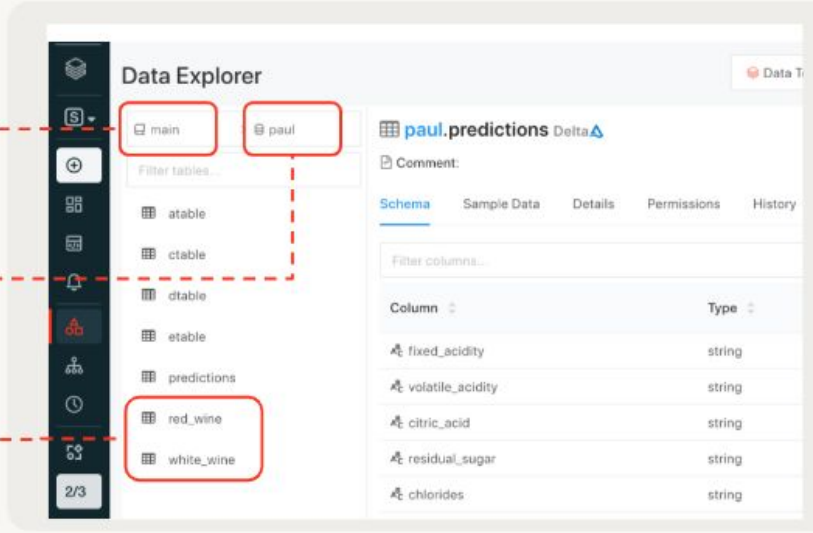
Cross-platform sharing via open protocol

Share existing Delta tables with no replication

Three level namespace

Seamless access to your existing metastores






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

SELECT * FROM hive_metastore.default.customers;
```

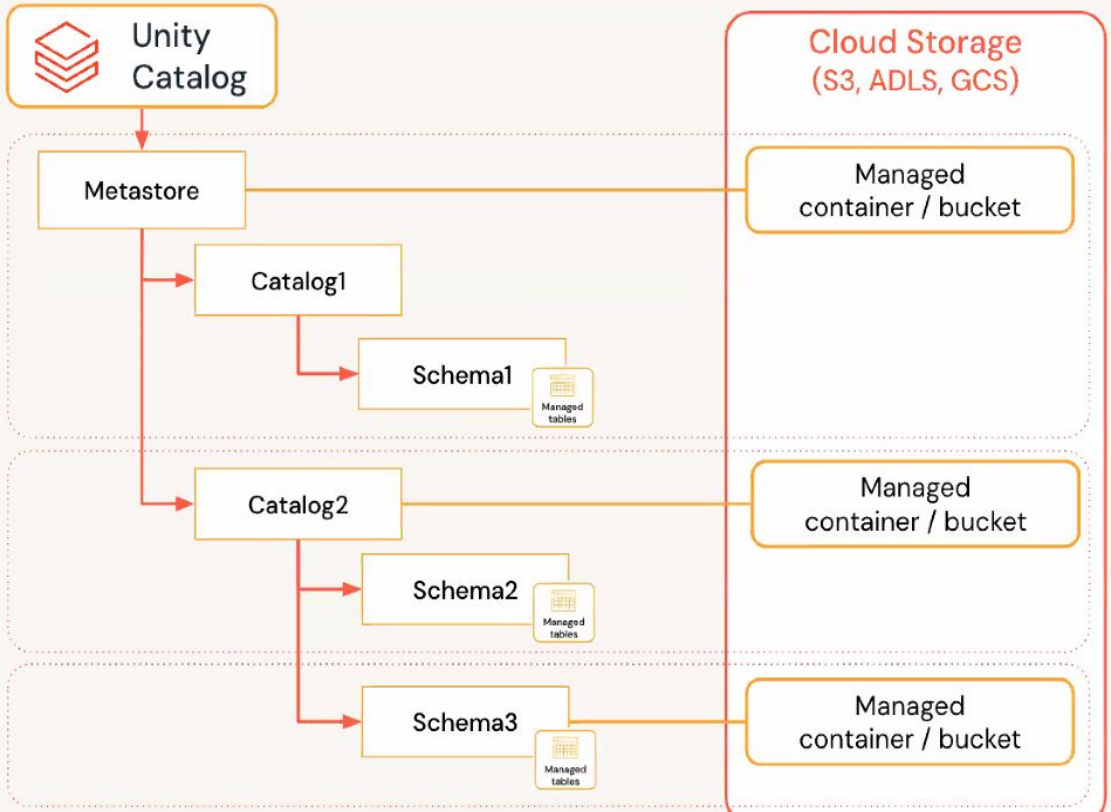
Default access to storage by catalog or schema

Use managed data sources for data isolation or cost allocation

1) Store at the **metastore**

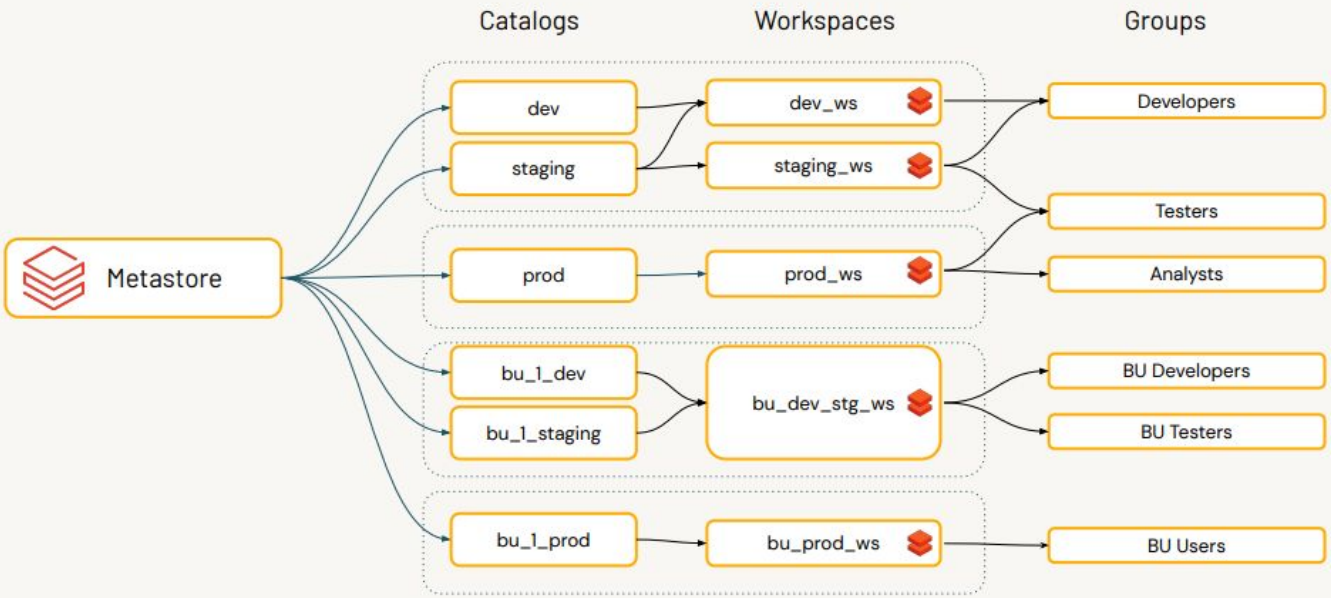
2) Store at the **catalog**

3) Store at the **schema**



Access data from specified environments only

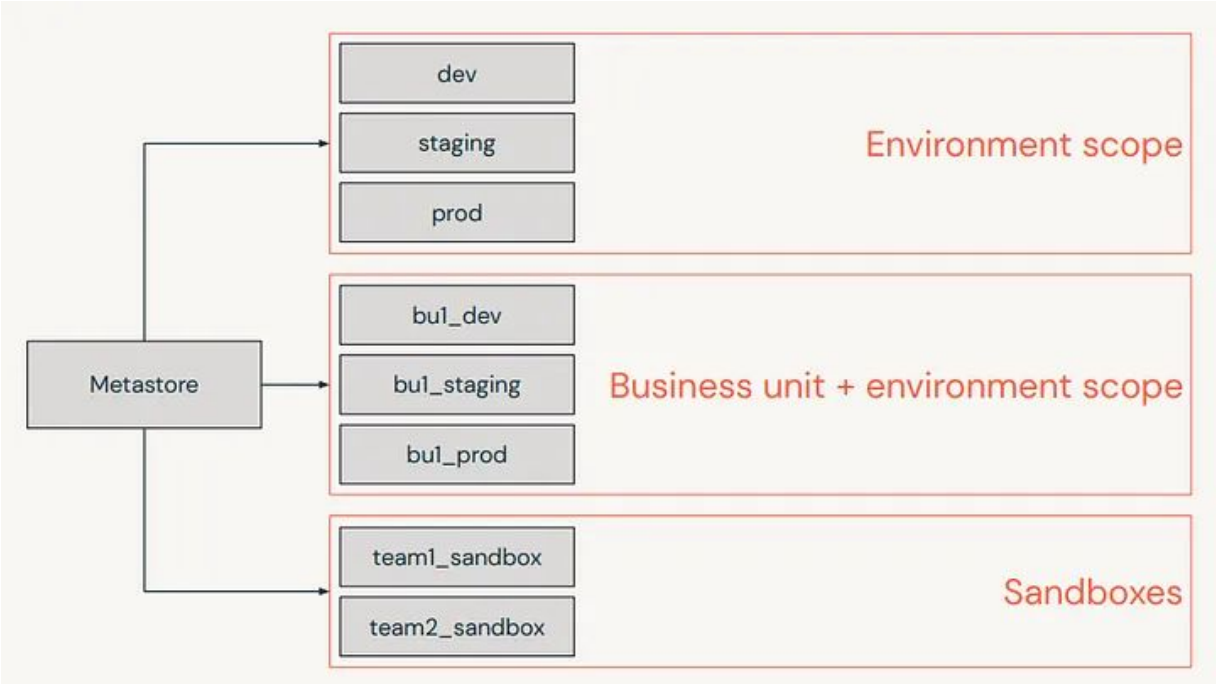
Restrict data access by environment or purpose



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

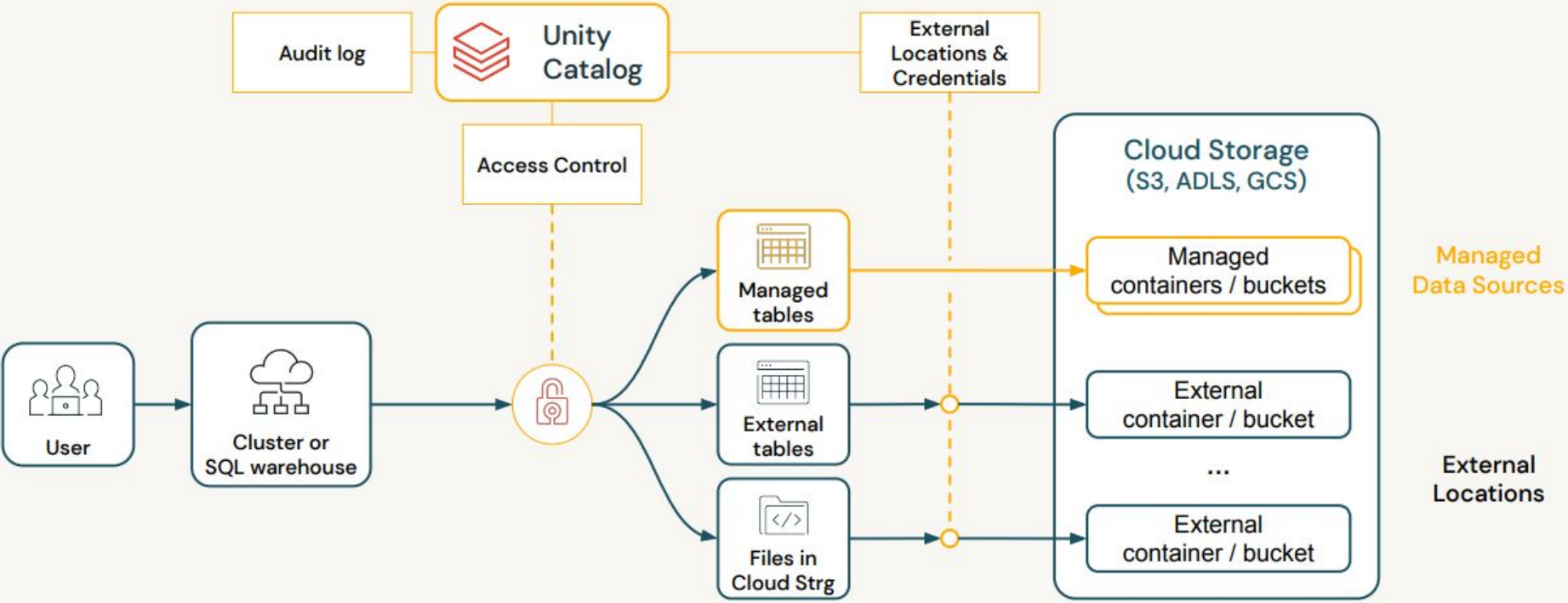
Metastore Catalog Data Segregation

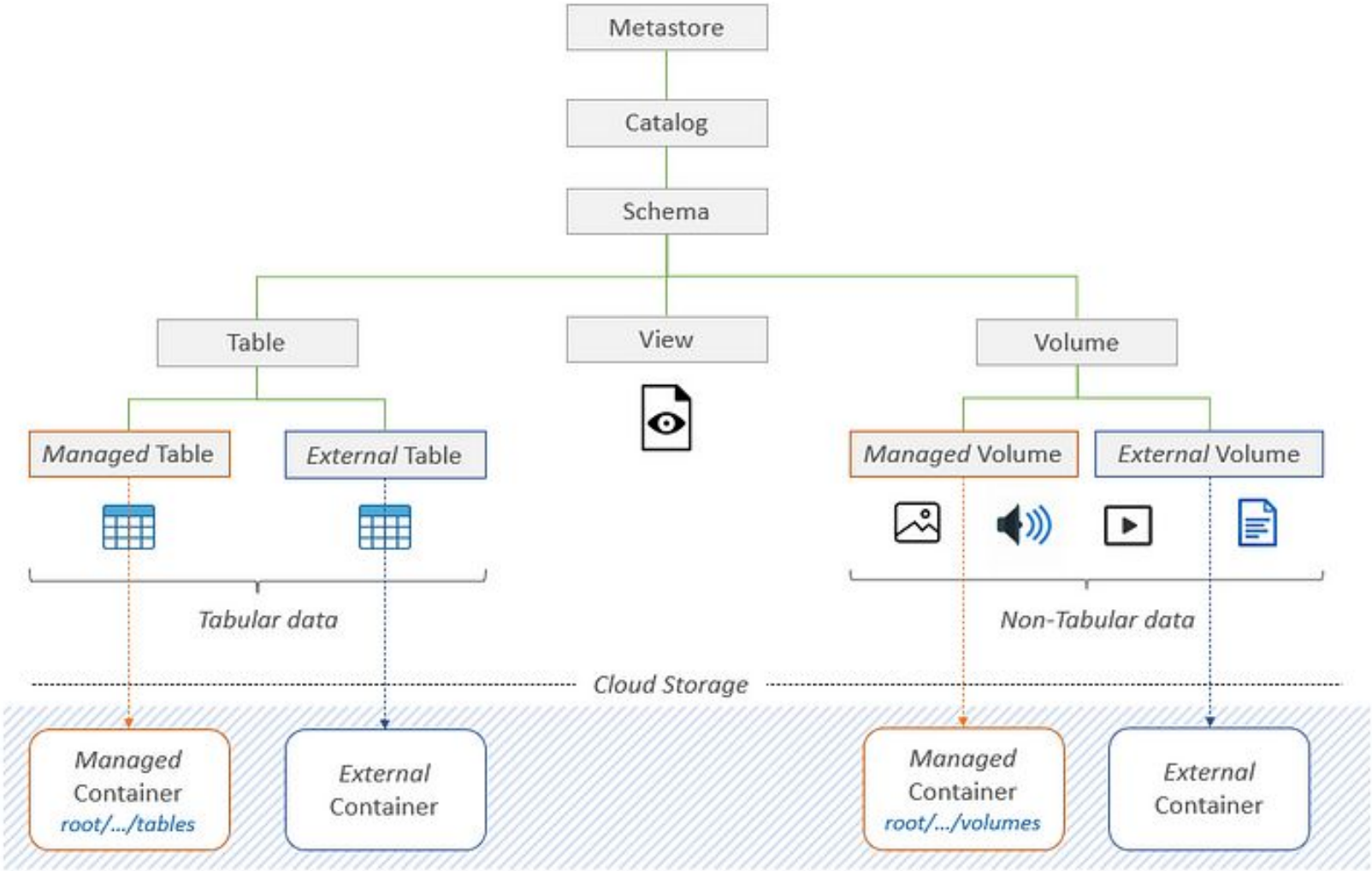
- Segregate your data at the catalog level.
- Group your data based on their type or use case.
- Create a separate catalog for each team in your organization, to ensure that each team has access to nly the data they need and there is no confusion about who owns what data .



Managed Data Sources & External Locations

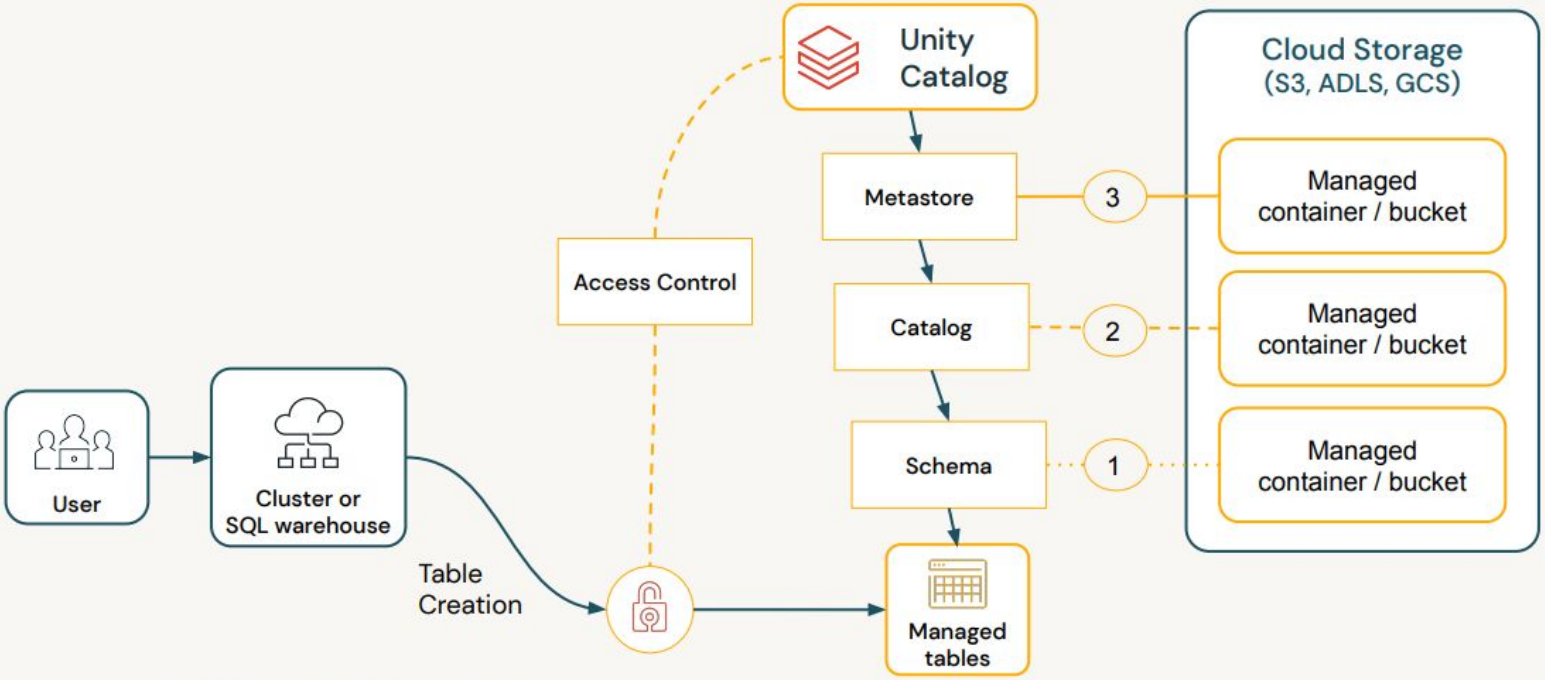
Simplify data access management across clouds




Metastore
Catalog
Schema
Table
View
Volume
Managed Table
External Table
Managed Volume
External Volume
Tabular data
Non-Tabular data
Cloud Storage
Managed Container root/.../tables
External Container
Managed Container root/.../volumes
External Container

Managed Data Sources for Catalogs and Schemas

Default access to managed data for data isolation



In Preview

Volumes in Unity Catalog

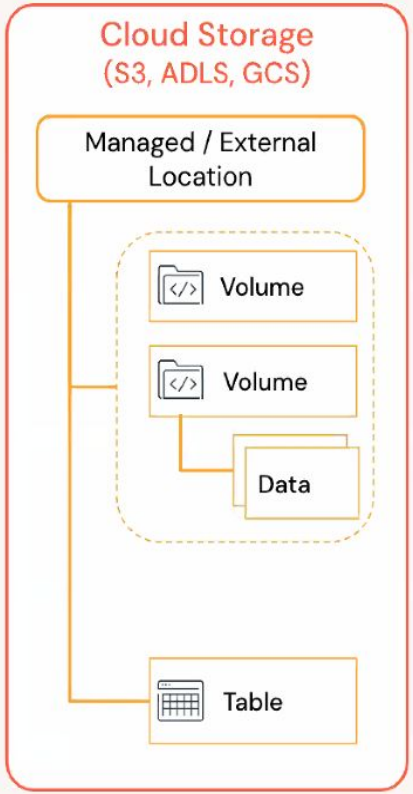
Access, store, organize and process files with Unity Catalog governance

- Volumes can be accessed by POSIX commands

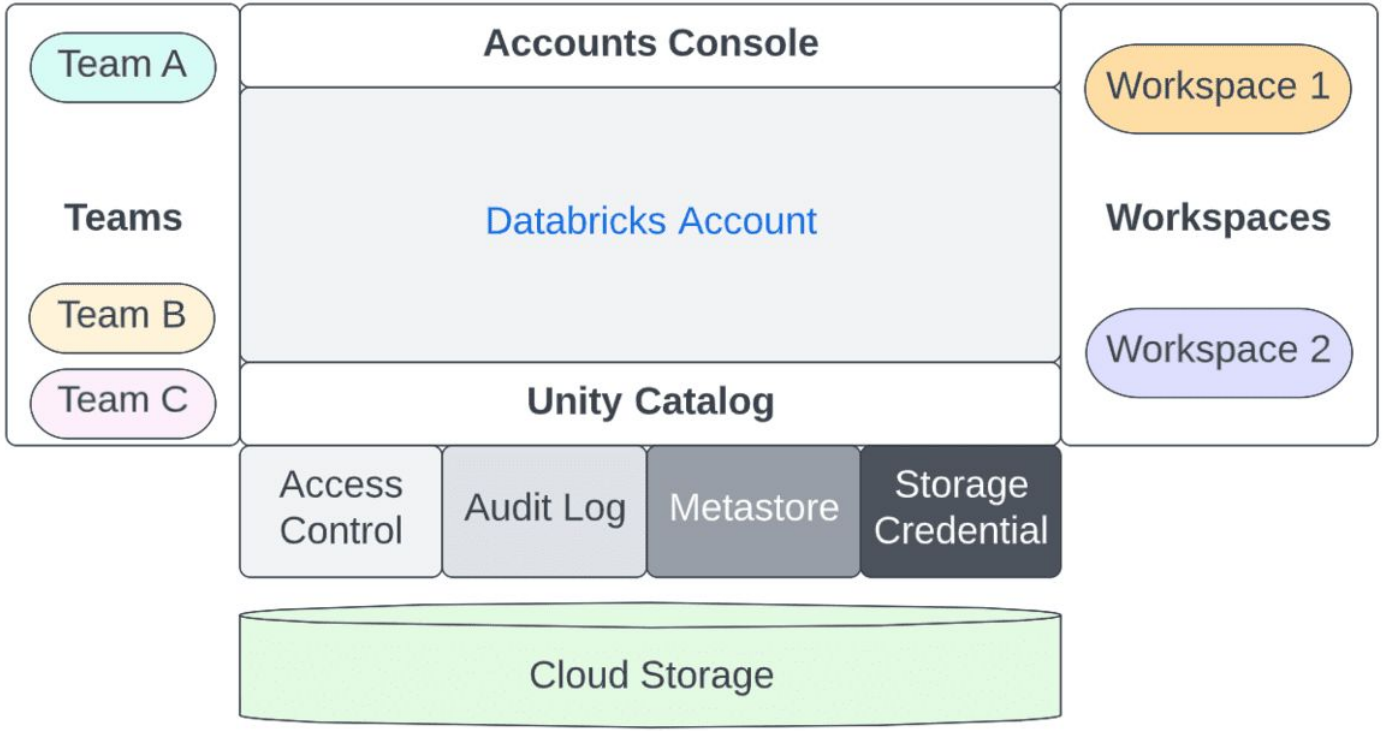
```
dbutils.fs.ls("s3://my_external_location/Volumes/volume123")

ls /Volumes/volume123
```

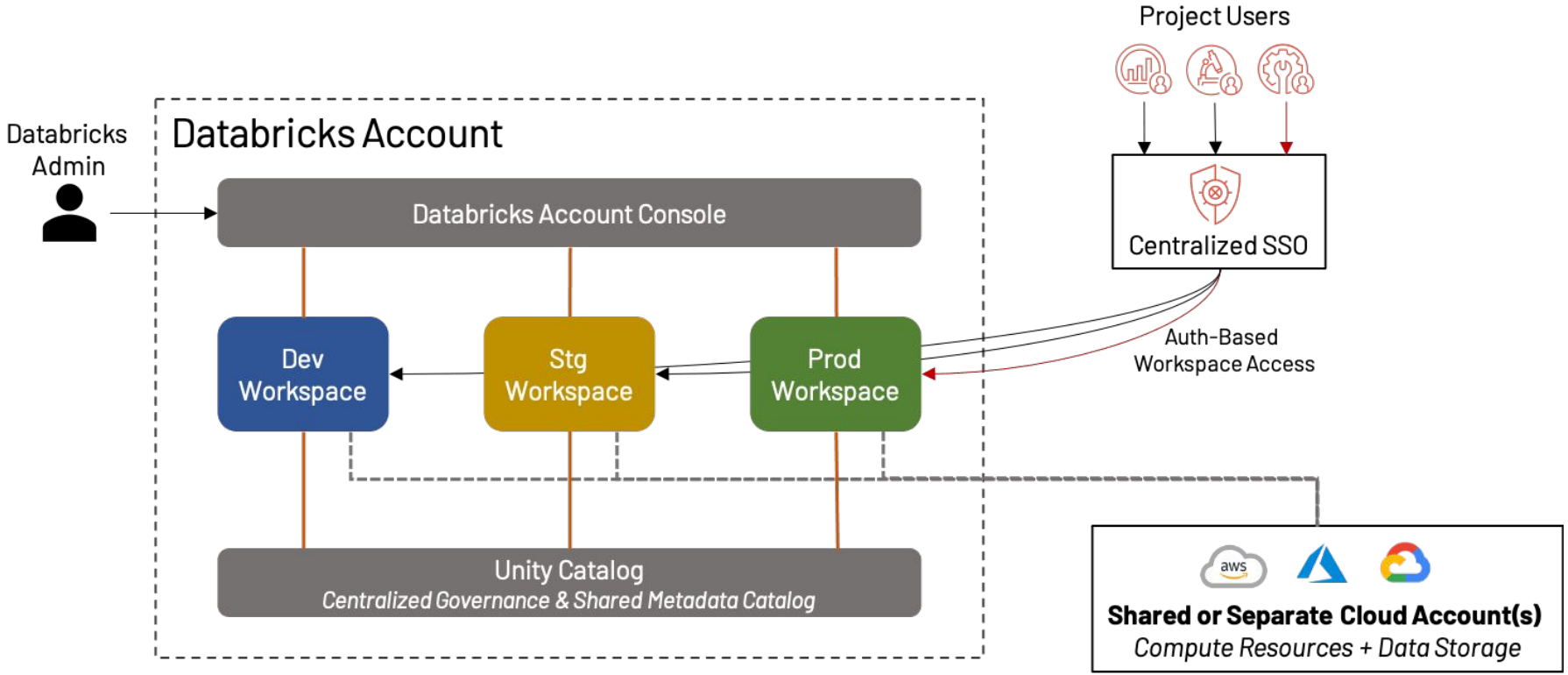
- Volumes are created under Managed or External Locations and show up in UC Lineage
- Volumes add governance over non-tabular data sets
  - Unstructured data, e.g., image, audio, video, or PDF files, used for ML
  - Semi-structured training, validation, test data sets, used in ML model training
  - Raw 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



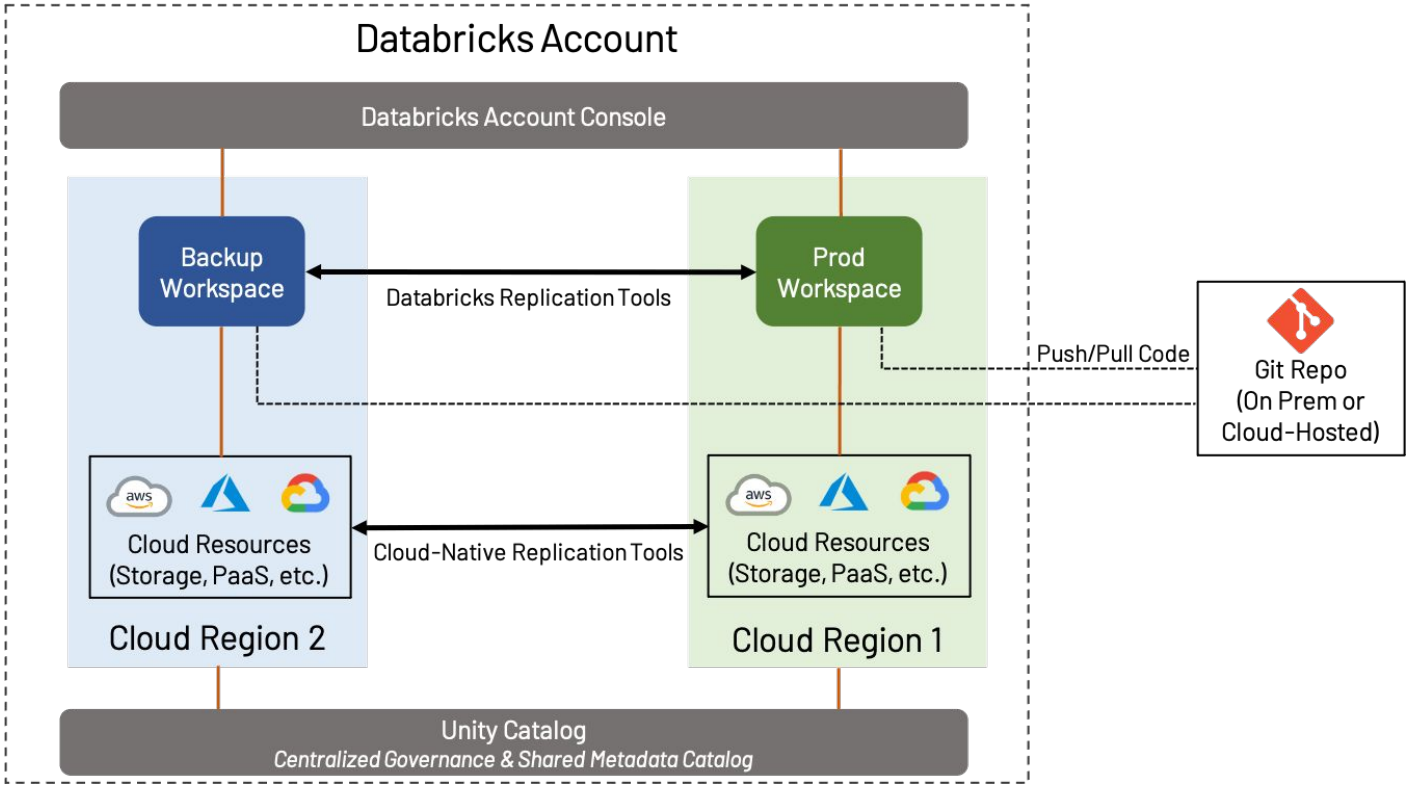
Account Artifacts



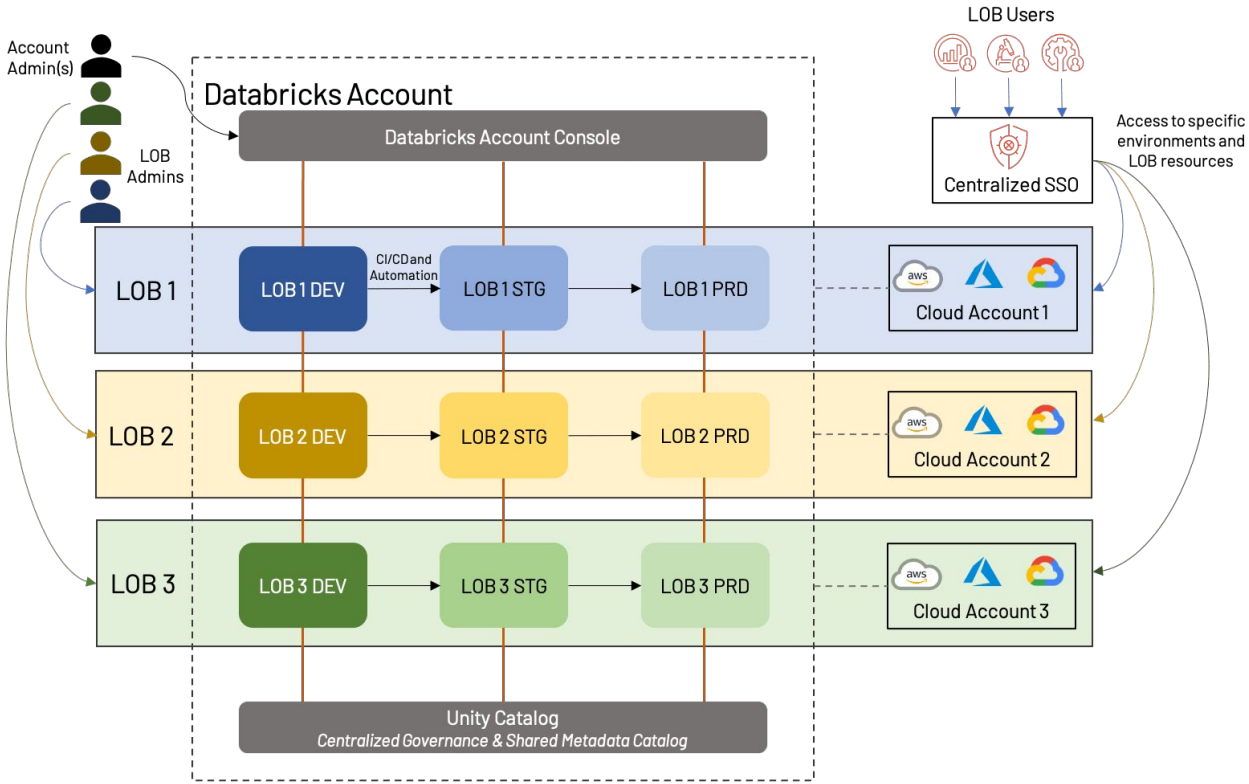
Per-Environment Workspaces



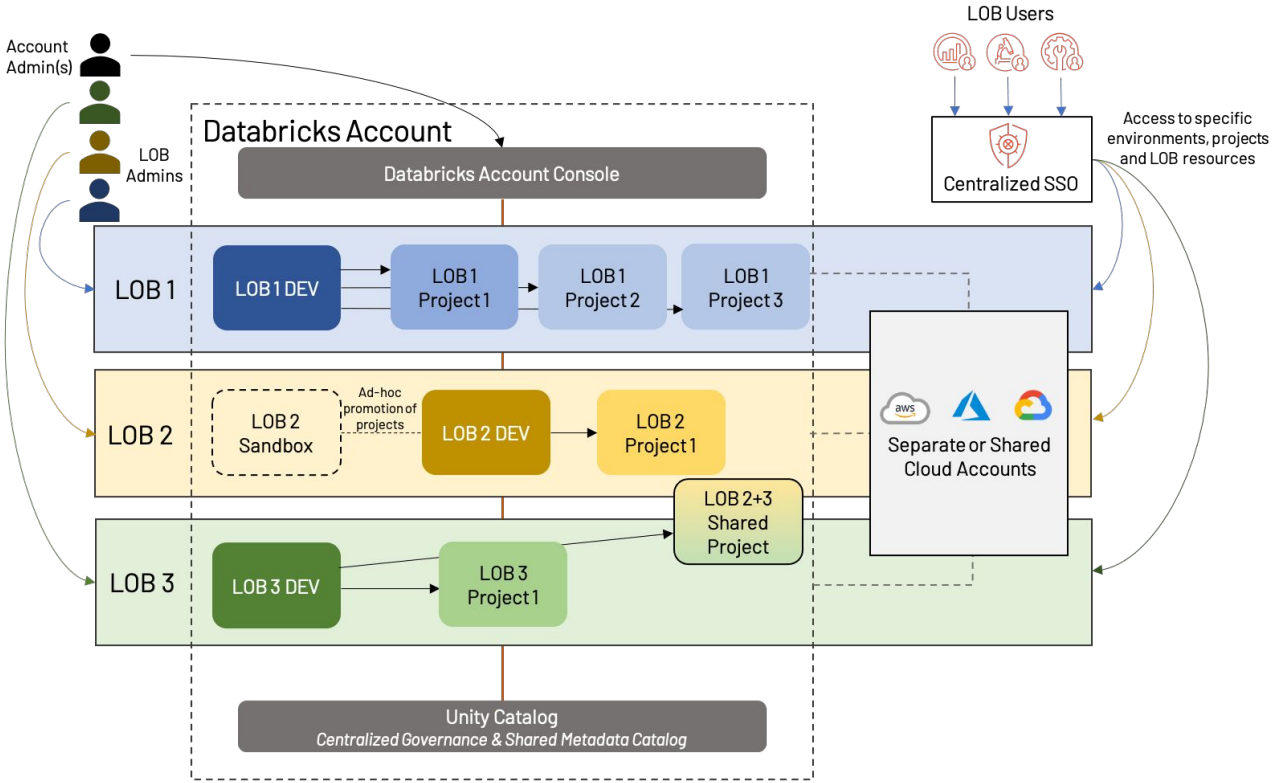
DR and Regional Backup



Line of Business Isolation



Data Product Isolation



External Locations and Storage Credentials

- Allow Unity Catalog to read and write data on your cloud tenant on behalf of your users, including:
  - Creating, reading from and writing to External Tables
  - Creating a Managed or External Table from files stored on your cloud tenant
  - Inserting records into tables from files stored on your cloud tenant
  - Directly exploring data files stored on your cloud tenant
- Storage Credential encapsulates a long-term cloud credential that provides access to cloud storage (e.g. in AWS IAM role to access S3 buckets)
- External Location combines a cloud storage path with a storage credential in order to authorize access to the cloud storage path. Subject to UC access-control policies.

```
CREATE EXTERNAL LOCATION [IF NOT EXISTS] <location_name>
URL 's3://<bucket_path>'
WITH ([STORAGE] CREDENTIAL <storage_credential_name>)
[COMMENT <comment_string>];
```

External Locations and Storage Credential

- Use external locations rather than using storage credentials directly.
- Minimize the number of users with direct access to a bucket being used as an external location. This limits users from bypassing access control in a UC metastore and disrupting auditability.
- Do not mount storage accounts to DBFS that are being used as external locations.
- Migrate mounts on cloud storage locations to external locations within UC using Data Explorer.
- Ensure that Block Public Access is enabled on your storage locations.
- Do not enable Soft delete for blobs.
- Use one External Location to contain all External Tables in one Catalog.
- Use one External Location to contain all External Tables in one bucket/container.
- Use one External Location per user for user shares.
- Use one External Location per team for team shares.
- Use one Storage Credential per team or per Catalog (bucket/container).
- Maintain all External Locations at the same hierarchical depth with any single S3 bucket or ADLS account.
- Save compute cost by moving External Tables using cloud-native commands.

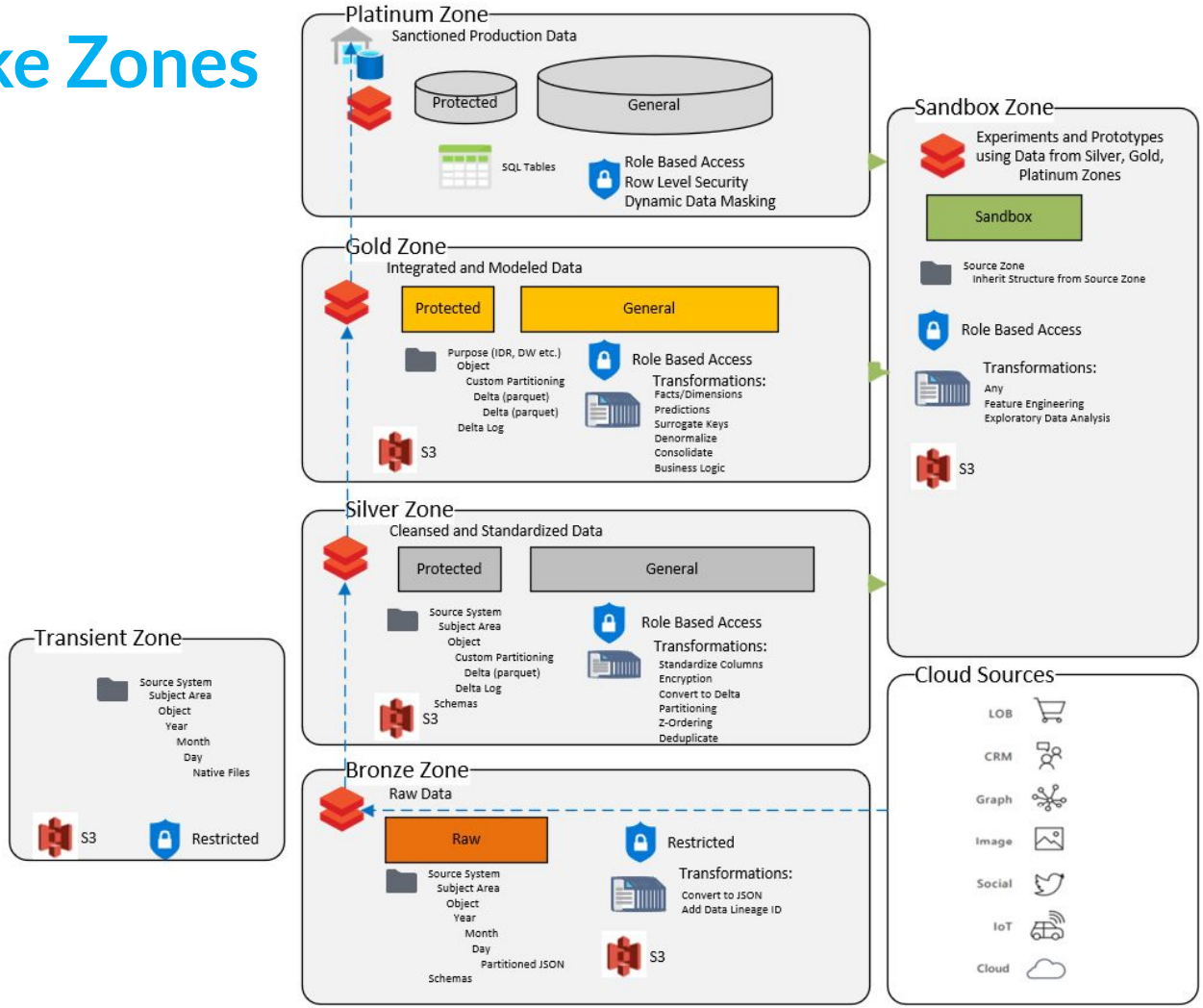
```
resource "databricks_storage_credential" "example" {
  name = "example-credential"
  aws_iam_role {
    role_arn = "arn:aws:iam::123456789012:role/example"
  }
}
```

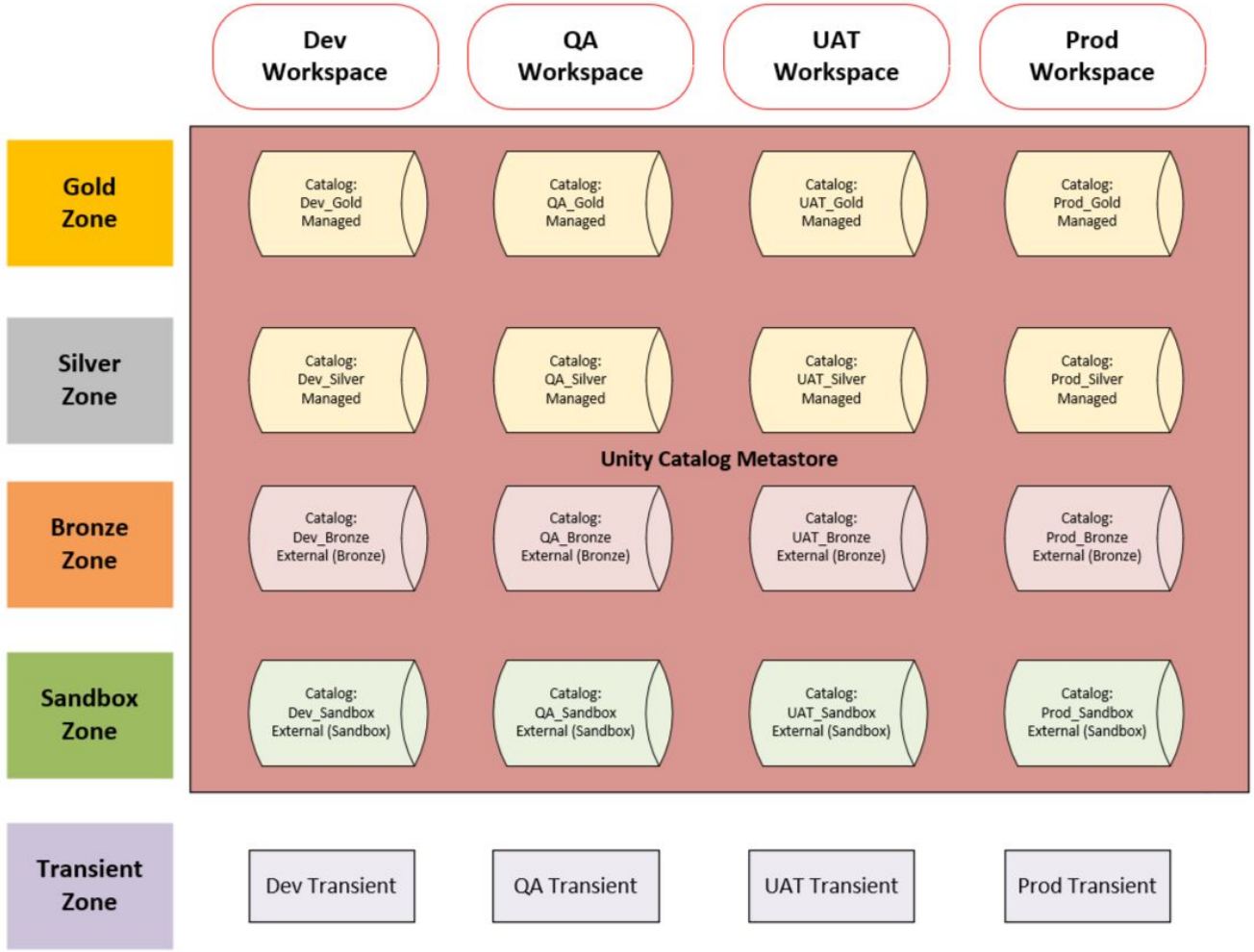
```
from databricks.sdk import WorkspaceClient
from databricks.sdk.service import catalog

w = WorkspaceClient()

storage_credential = w.storage_credentials.create(
    name='example-credential',
    aws_iam_role=catalog.AwsIamRole(role_arn='arn:aws:iam::123456789012:role/example'),
    comment="created via SDK")
```

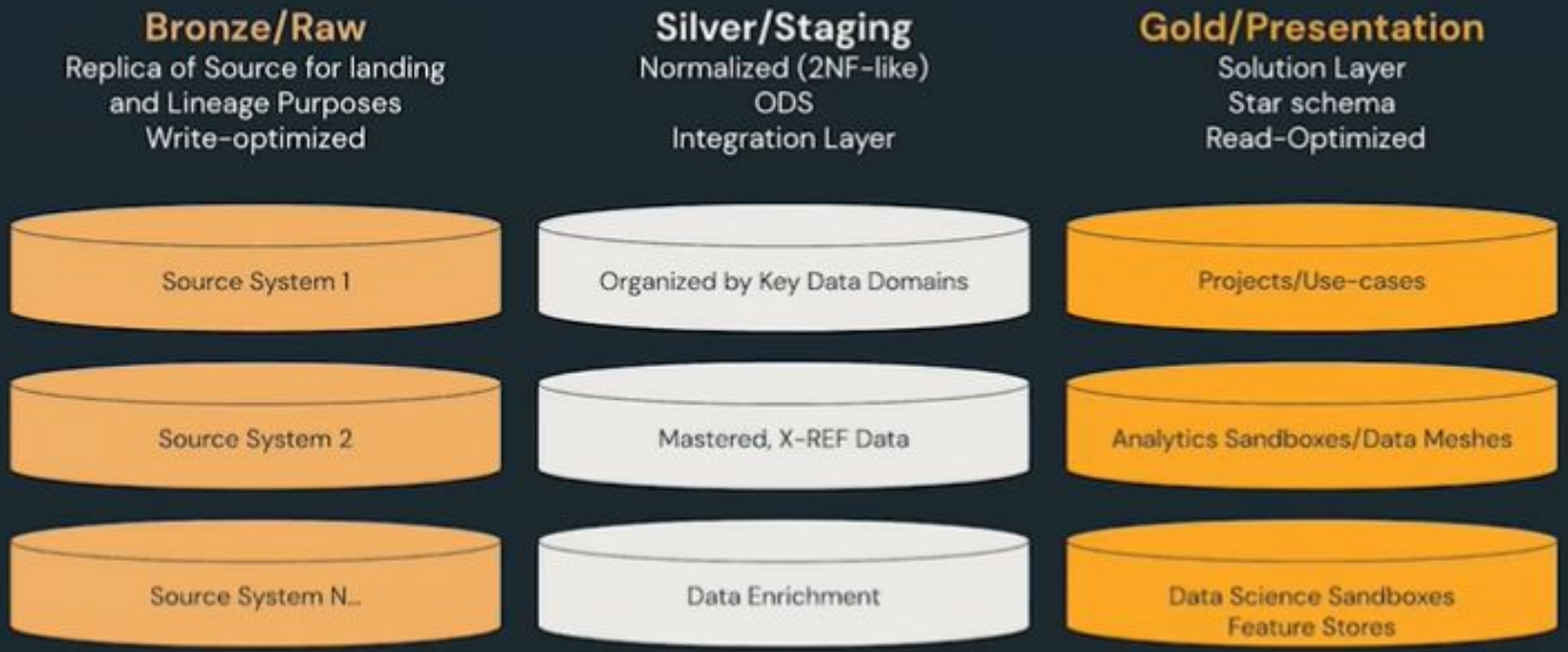
Data Lake Zones



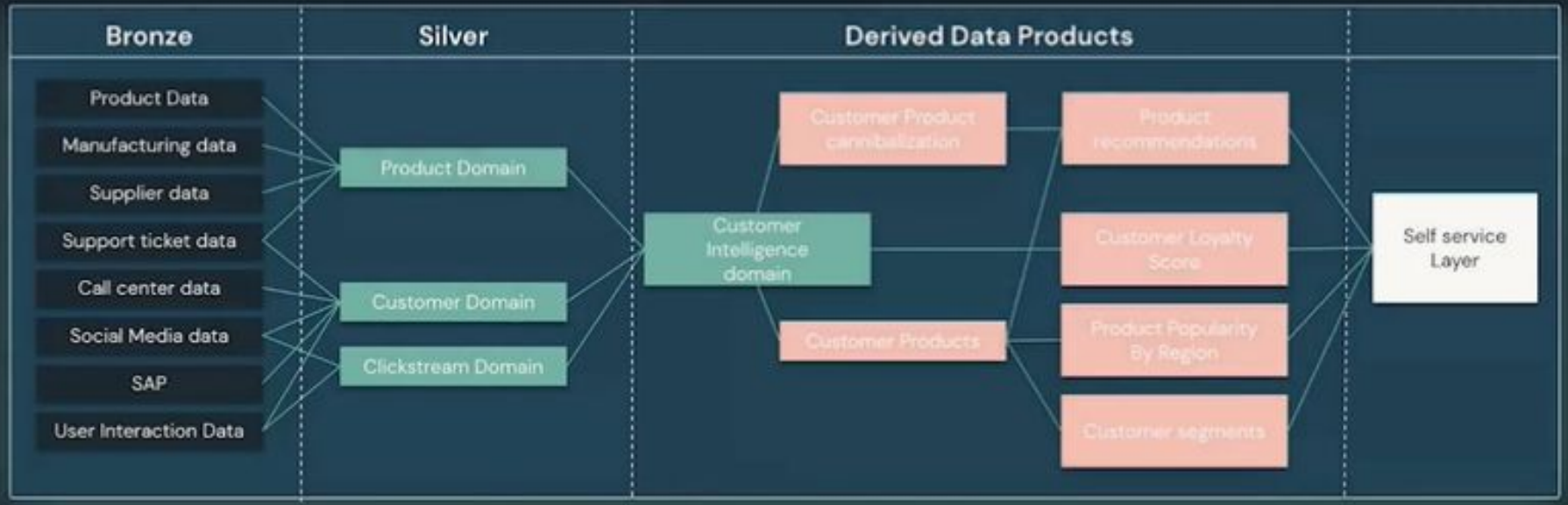

Dev Workspace
QA Workspace
UAT Workspace
Prod Workspace
Gold Zone
Silver Zone
Bronze Zone
Sandbox Zone
Transient Zone
Catalog: Dev_Gold Managed
Catalog: QA_Gold Managed
Catalog: UAT_Gold Managed
Catalog: Prod_Gold Managed
Catalog: Dev_Silver Managed
Catalog: QA_Silver Managed
Catalog: UAT_Silver Managed
Catalog: Prod_Silver Managed
Unity Catalog Metastore
Catalog: Dev_Bronze External (Bronze)
Catalog: QA_Bronze External (Bronze)
Catalog: UAT_Bronze External (Bronze)
Catalog: Prod_Bronze External (Bronze)
Catalog: Dev_Sandbox External (Sandbox)
Catalog: QA_Sandbox External (Sandbox)
Catalog: UAT_Sandbox External (Sandbox)
Catalog: Prod_Sandbox External (Sandbox)
Dev Transient
QA Transient
UAT Transient
Prod Transient

Lakehouse Medallion Zones

Best practices



Data lakehouse mesh architecture in practice



Data Lake Zone Recommendations

- Use the widely accepted medallion zones for data enrichment.
- The Lakehouse should be comprised of multiple storage accounts (e.g. S3, ADLS Gen2, or GCS).
    - These should be used as security boundaries and user access granted via passthrough from Azure Active Directory or OTKA.
    - Create Databricks Groups corresponding to each of the Zones and subzones of the Data Lake.
- To support Disaster Recovery and failover to another Cloud region, storage accounts should use geo-redundant storage.

Workspace Recommendations

- Plan Databricks Workspace layout (but strive to keep it as simple as possible).
- Consider using a cross-workspace centralized model registry, codebase, and metastore/catalog to aid collaboration.
- Minimize the number of top-level accounts (both cloud provider and Databricks) where possible.
- Create workspaces only when separation is necessary for compliance, isolation, or geographical constraints.
- Decide on an isolation strategy to support long-term flexibility without undue complexity. Implement guidelines before bringing workloads onto your Lakehouse.
- Automate cloud processes, including SSO/SCIM, Infrastructure-as-Code, CI/CD Pipelines and Repos, cloud backup and monitoring.
- Establish a COE team for central governance of an enterprise-wide strategy, where repeatable aspects of data and machine learning pipeline is templatized and automated so that different data teams can use self-service capabilities with guardrails in place.
- The COE Team should be lightweight but critical hub for data teams and should view itself as an enabler–maintaining documentation, Run Books, how-tos, FAQs to educate users.
- Workspaces should support and perform Audit Log Delivery, compliance (e.g. HIPAA, PCI etc.), exfiltration controls, use of ACLs and user controls and regular review of all the above.

Initial UC Setup

Enable UC

Identify Databricks Account ID

Configure S3 Bucket

Create & Modify IAM Role

Create & Attach IAM Policy

Outcome: Enables Unity Catalog

Create Metastore

1. Log in to the account console.
2. Click on the Data icon.
3. Click on "Create Metastore".
4. Enter the following:
- A name for the metastore.
- The region where you want to deploy the metastore. This must be in the same region as the workspaces you want to use to access the data. Make sure that this matches the region of the storage bucket you created earlier.
- The S3 bucket path (you can omit s3://) and IAM role name for the bucket and role you created earlier.
5. Click "Create".
6. When prompted, select workspaces to link to the metastore.

Assign Workspaces

1. As an account admin, log in to the account console.
2. Click on the Data icon.
3. Click on the metastore name.
4. Click on the "Workspaces" tab.
5. Click on "Assign to workspaces".
6. Select one or more workspaces. You can type part of the workspace name to filter the list.
7. Click "Assign".
8. On the confirmation dialog, click "Enable".

Add Principals

Add Users, Groups, and Service Principals:

1. Log in to the Databricks account console.
2. Click on the "Admin Console" icon.
3. Click on "Users", "Groups", or "Service Principals" depending on what you want to add.
4. Click on "Add User", "Add Group", or "Add Service Principal".
5. Fill in the required information and click "Add".

Unity Catalog as Default

Unlock the full potential of your Lakehouse environment

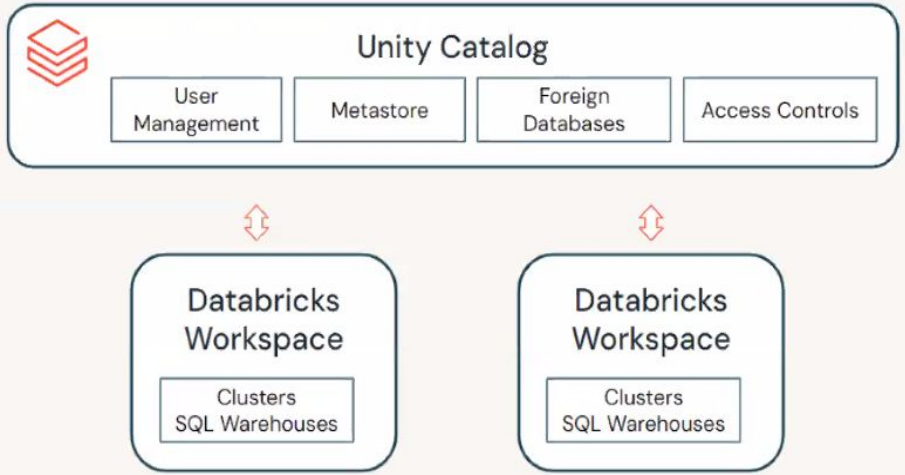
Leverage the benefits of a native governance solution and simplify onboarding

Automatically created catalog per workspace to get started with Unity Catalog

Benefit from unified governance across workspaces

Leverage the latest enhancements in Databricks Lakehouse Platform

GA Q3 on AWS and Azure



Account Level Configuration: Action Items

01	**Account Level Design Decisions**	• Environment/Team Separation (workspace vs. catalogs) • UC Catalogs List, Naming Conventions, Paths • UC Schemas/Databases • External Locations Strategy
02	**Storage Configuration**	• Create Storage Credentials • Create External Locations • Replacement for Mount Points
03	**Catalog Creation**	• Define default S3 Path • Include Foreign Catalogs
04	**Schema Creation**	• Define default S3 Path
05	**Volume Creation**	• Replacement for Mount Points • Replacement for DBFS

UC Friction Points and Complexity

- Hive Metastore Assets: Tables/Views/ACLS - External/DBFS Root Tables
- Mount Points
- Active Clusters - Init Scripts/Libraries etc.
- DLT Pipelines
- ML Models/Feature Stores
- Local Groups - Overlapping/Conflicting

Activation

Assign a metastore to the workspace

1. Attach UC Metastore to Workspace
2. Beware of the “default cluster type” challenge
3. Create CATALOG for Workspace (optional)
4. Bind Workspace to new CATALOG (optional)
5. Create Workspace Admin group at Account Level
6. Set Workspace Admin group as the CATALOG owner

Challenges with Activation

What works what doesn't

1. Account admin credentials are required for Metastore Creation
2. Azure: we need a member of the "Global AD Admin" group
3. Enabling UC automatically turns on "Identity Federation" for the cluster
4. Enabling UC can interfere with existing automated jobs:
   a. If the cluster DBR is 10 or later
   b. If the cluster is created using an API call (API, ADF, Terraform, ETC)
   c. A cluster Access Mode has to be specified in the API call:
      "data_security_mode": "NONE"

Upgrade Plan

- Assessment Results
- Upgrade Plan Worksheet