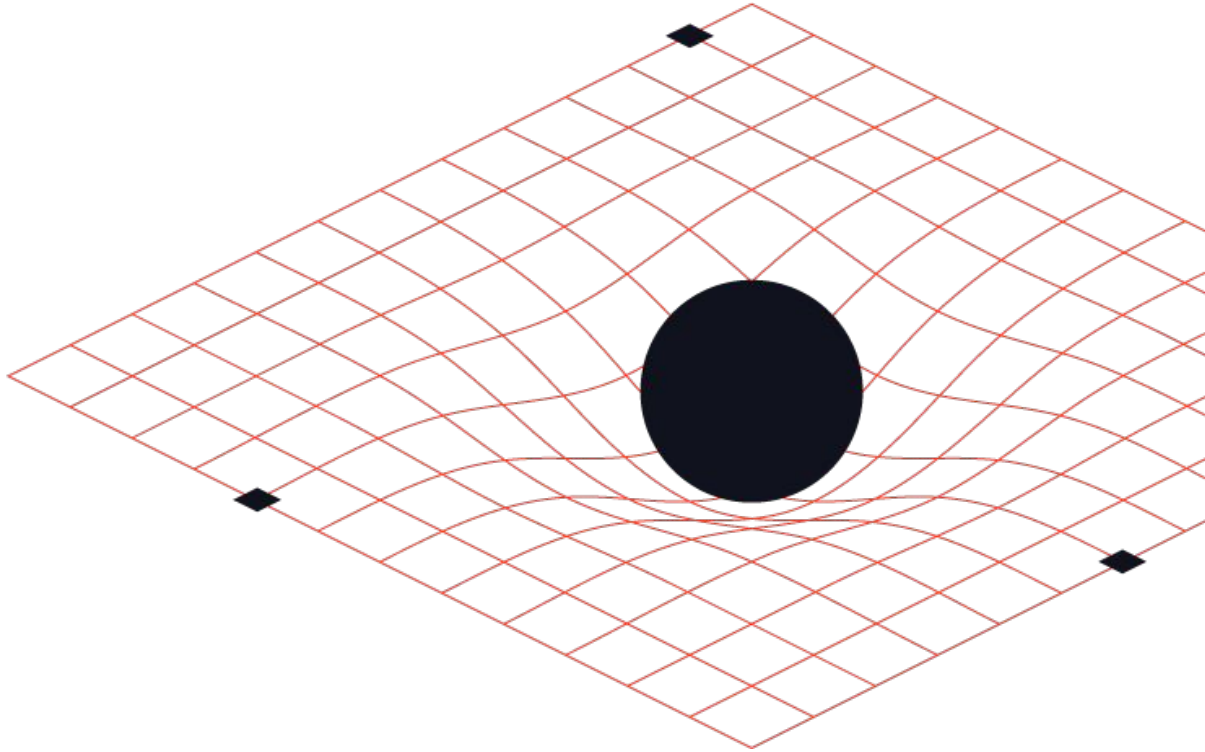
TECH SUMMIT

Unity Catalog Best Practices and Patterns from accumulated Shared Technical Services experiences



How to make the most out of UC

Shared Technical Services

Agenda

UC Best Practices and Patterns from accumulated STS experiences

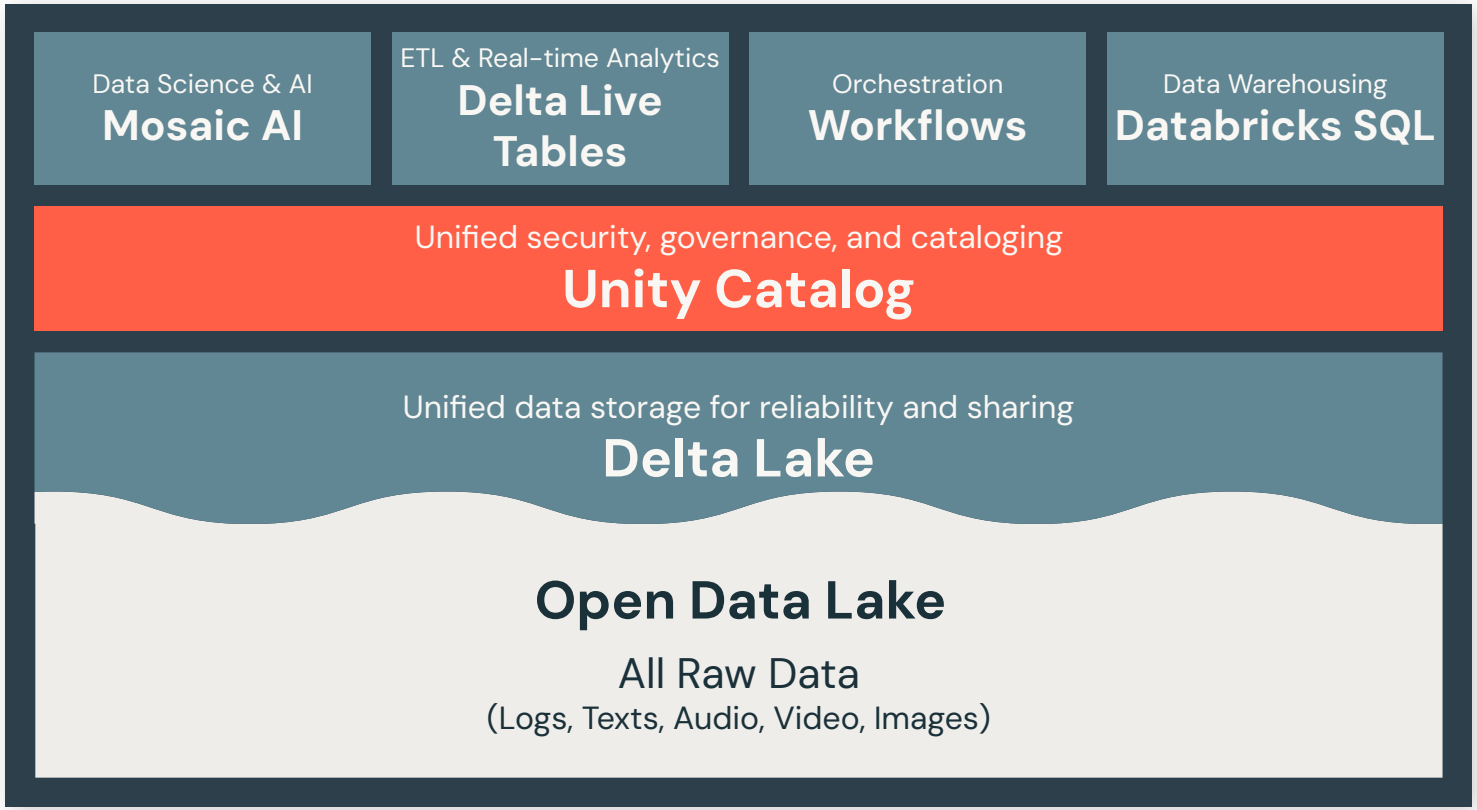
- What is Shared Technical Services?
- Unity Catalog Overview
- Key considerations when planning your environment
  - Administrative roles
  - Logical Data Isolation
  - Physical Data Isolation
  - Fine grained access control
  - System Tables

`#uc-is-awesome`

Unity Catalog Overview

What is Unity Catalog?

Unified governance for all data and AI assets

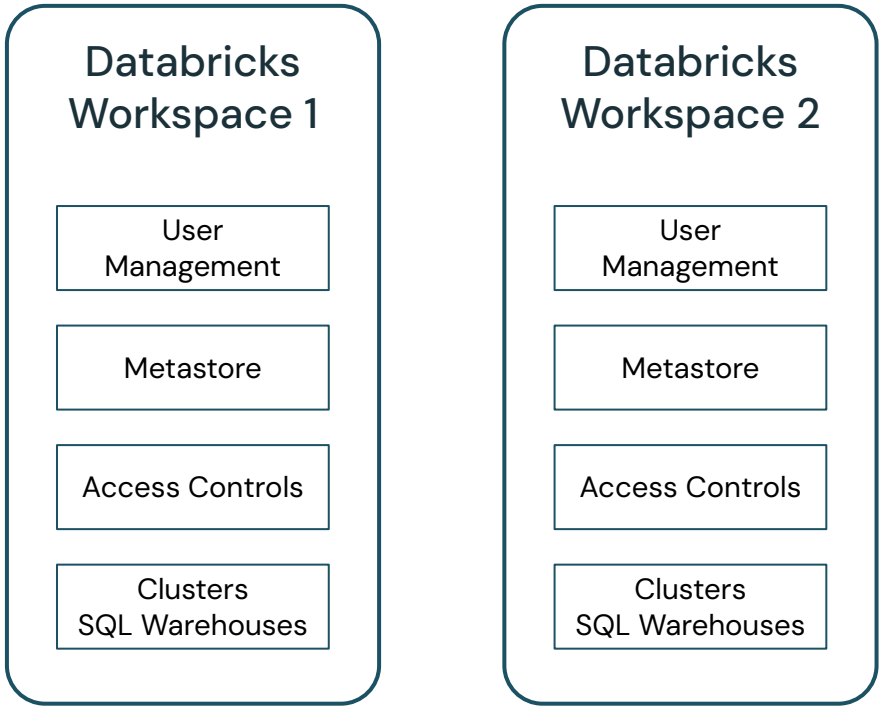


- Centralized governance and access control.
- Centralized data search and discovery.
- Data access auditing.
- Data lineage.
- Delta Sharing.

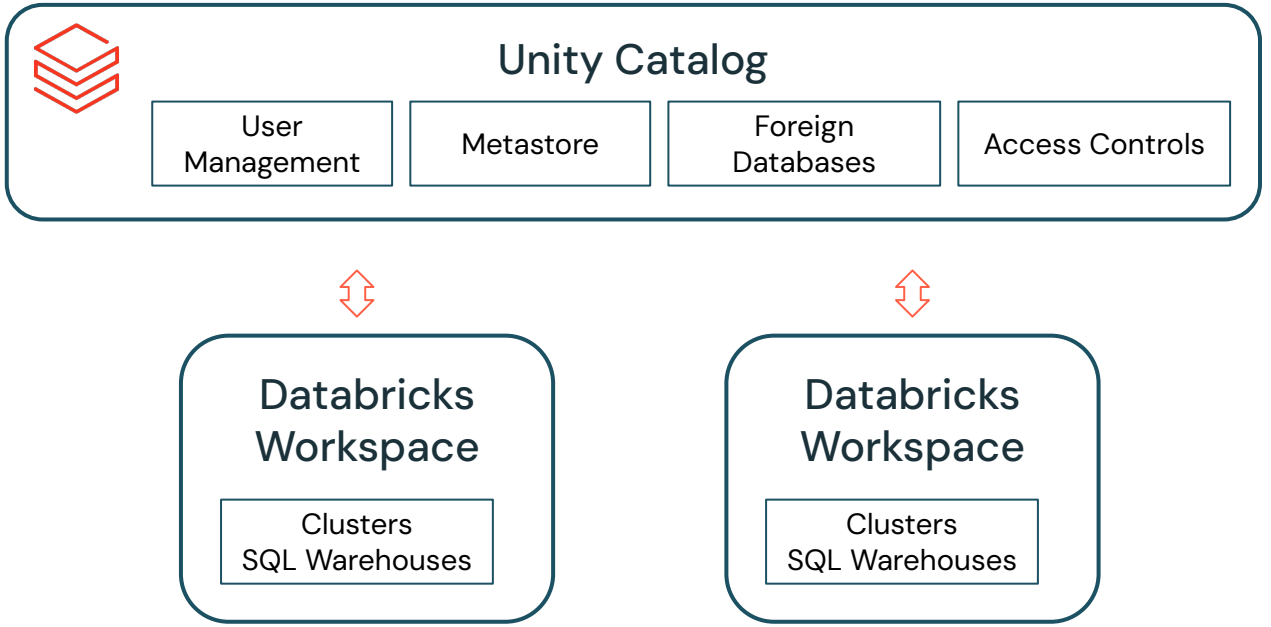
All your metadata, in one place

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

Without Unity Catalog

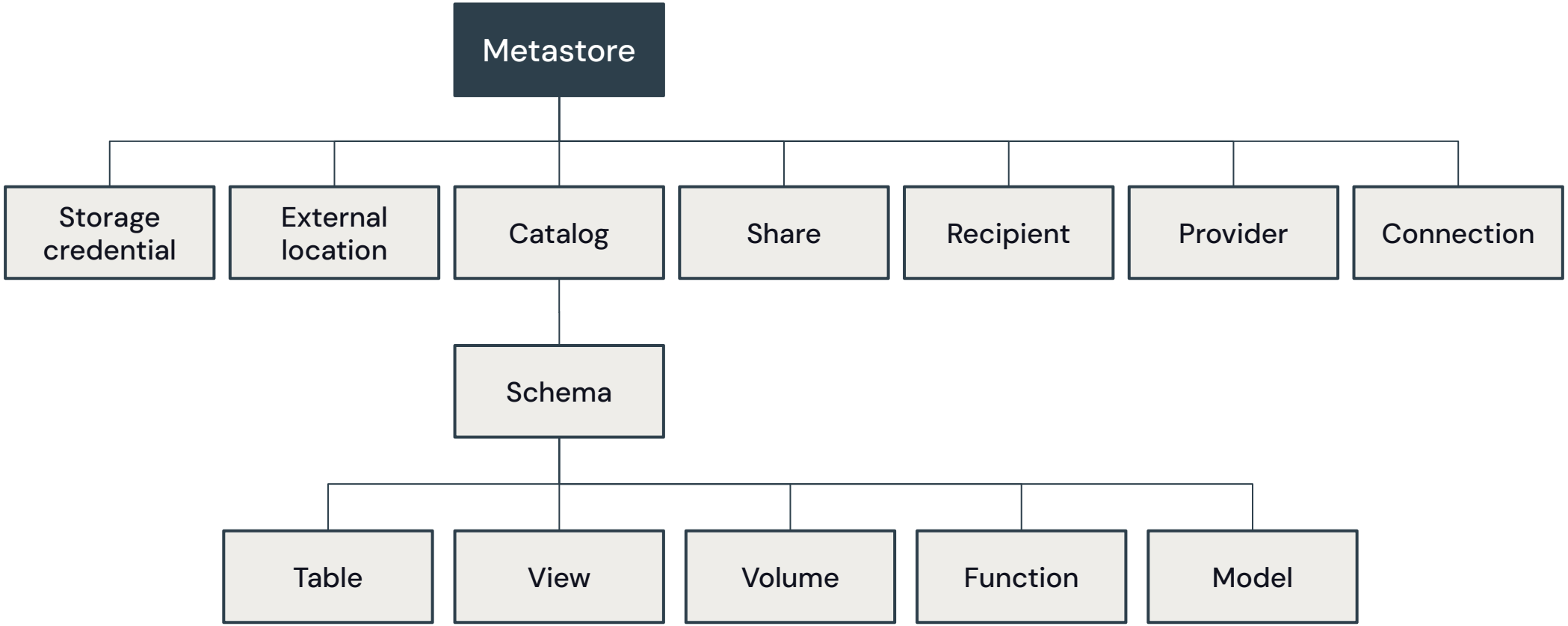


With Unity Catalog



	w/ UC	w/ Hive Metastore
Governance across workspaces (Access controls, metastore, Identity management)	✅	❌ (Governance at workspace level)
Automated table and column level lineage in real-time	✅	❌
Row Filters and Column Masking (use standard SQL functions for fine-grained access controls)	✅	❌
Volumes (Govern non tabular data/arbitrary files - image, video, PDF etc.)	✅	❌
Lakehouse Federation (Discover, govern and query external databases, data warehouses, Hive Metastore, AWS Glue)	✅	❌
Lakehouse Monitoring (Monitor quality of data and ML models with auto generated alerts and dashboards)	✅	❌
Delta Sharing (Share files, tables, ML models, notebooks across clouds, regions and data platforms)	✅	❌
Databricks Marketplace	✅	❌
Databricks Cleanroom (Collaboration on data in a privacy-preserving manner)	✅	❌
Databricks Data Room (aka Genie)	✅	❌
Databricks Assistant / LakehouseIQ (Knowledge engine on the lakehouse to get insights in natural language)	✅	❌ (Degraded experience)
Governance for AI (manage and govern ML Models, Feature store)	✅	❌
System Tables (end to end observability for Billing, Auditing, Lineage, Marketplace analytics)	✅	❌
Lakehouse Apps	✅	❌
Databricks Connect V2	✅	❌
Materialized Views	✅	❌
AI Powered Predictive optimization (optimizes your query plans and data layout for peak performance, intelligently balancing read vs. write performance)	✅	❌
Streaming Tables	✅	❌
HMS Interface (Query data registered in UC using other data platforms such as Amazon Athena, Presto, Trino,EMR)	✅	✅
Vector Search for indexing	✅	❌
Serverless workflows	✅	❌

Unity Catalog hierarchy



What should we consider to make the most out of UC?

Key considerations

- Who will be working on Databricks? (BUs, personas, etc…)
- Do they have different levels of access/permissions?
- How will we arrange our data logically? (SDLC, teams, projects, etc…)
- Where will we store our data physically?

Administrative roles in Databricks

Administrators

Think carefully before giving admin access

Account Admin

- Create workspaces
- Create & configure metastores
- Create users, groups & service principals
- Grant users access to workspaces
- Set billing budget threshold alerts

Metastore Admin

- Create CATALOG, CREDENTIAL, EXTERNAL LOCATION
- Create SHARE, RECIPIENT
- Change OWNER of any securable object (indirect access to data)

Workspace Admin

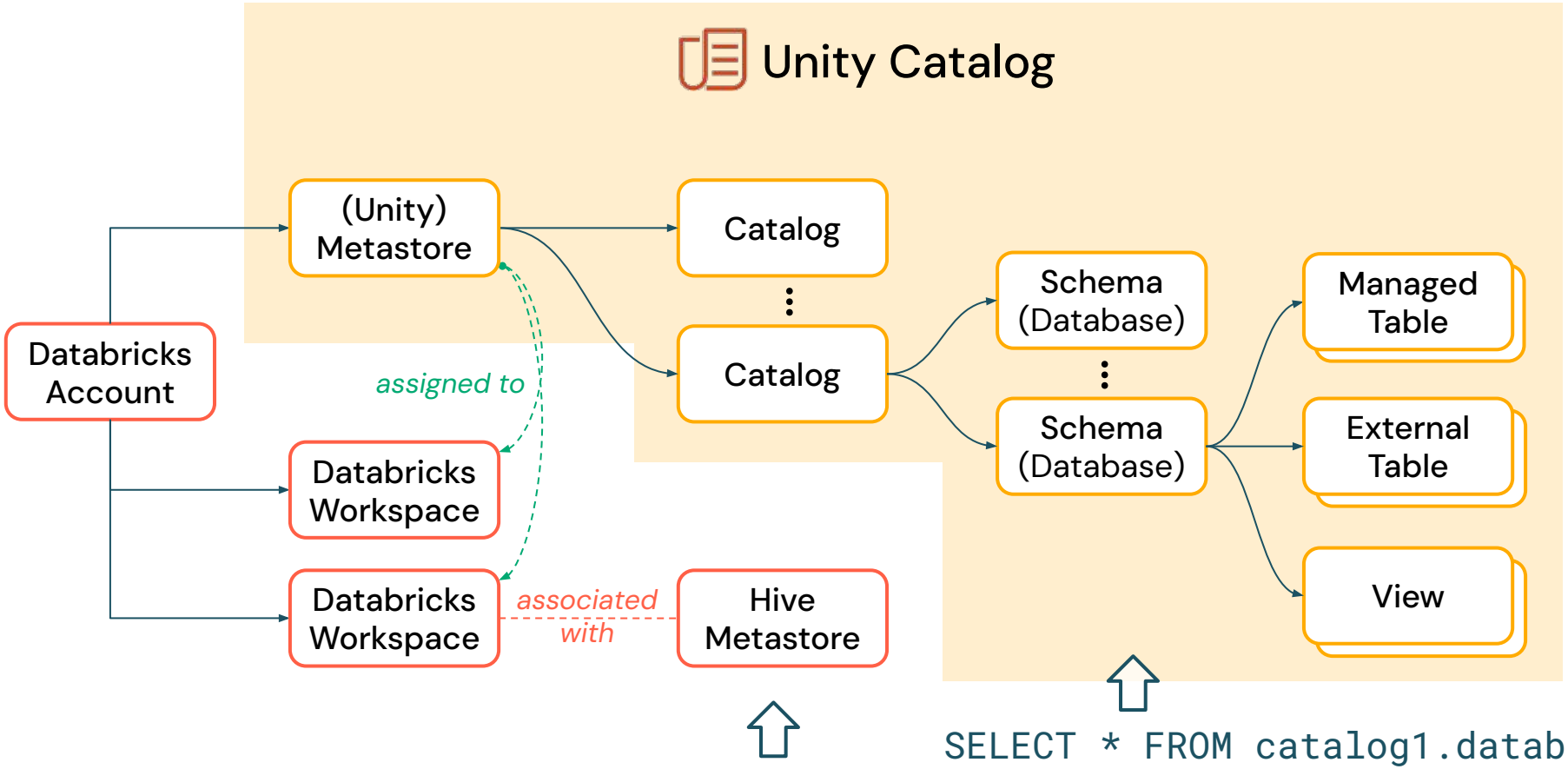
- Add users and groups to workspace
- Create clusters & cluster policies
- Change OWNER of clusters, workflows, notebooks, queries, dashboards

Data Owner

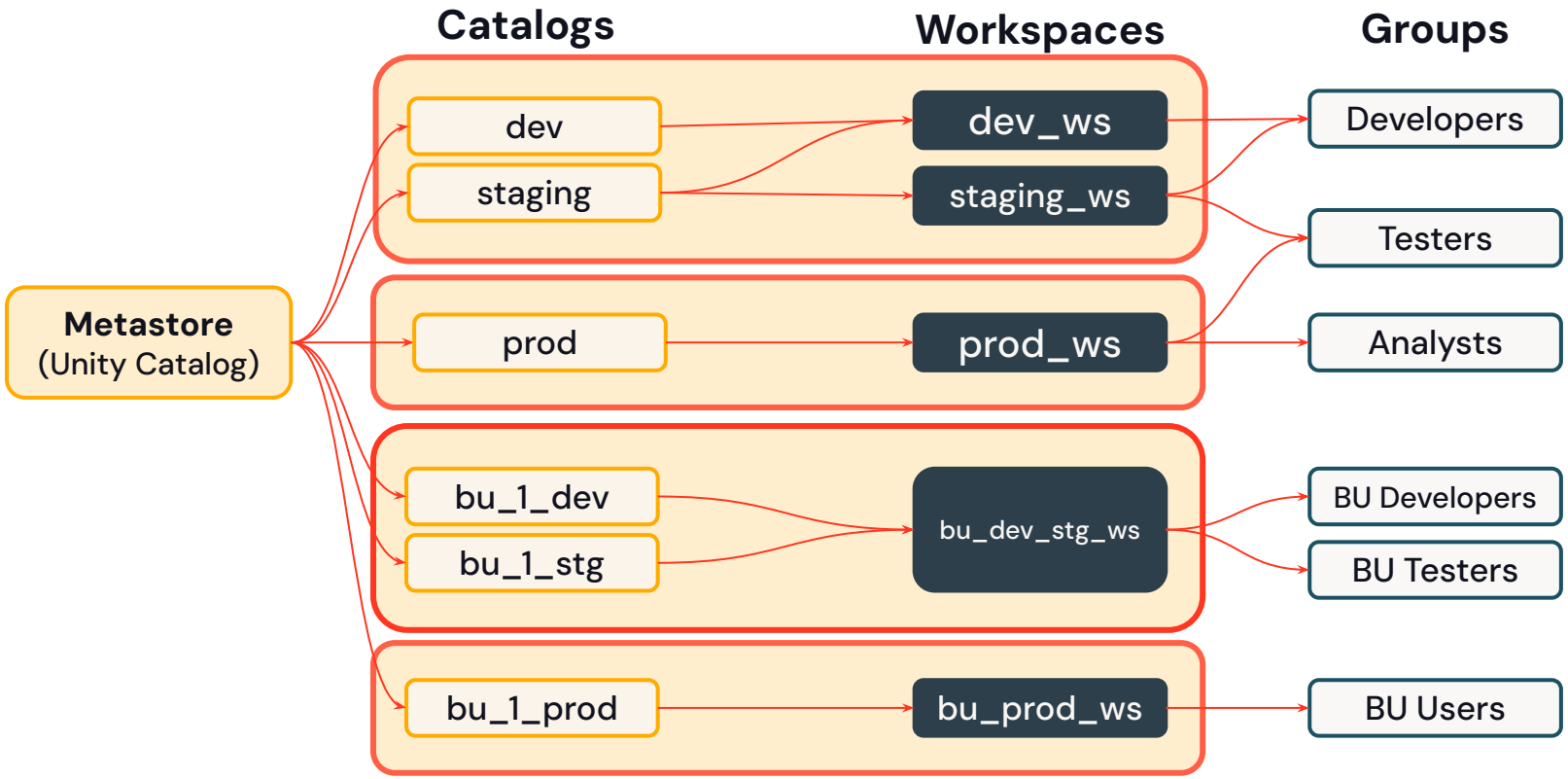
- Change ownership of securable object (CATALOG, SCHEMA, TABLE, VIEW, etc.)
- Grant any privilege to any principal

Logical data isolation layout

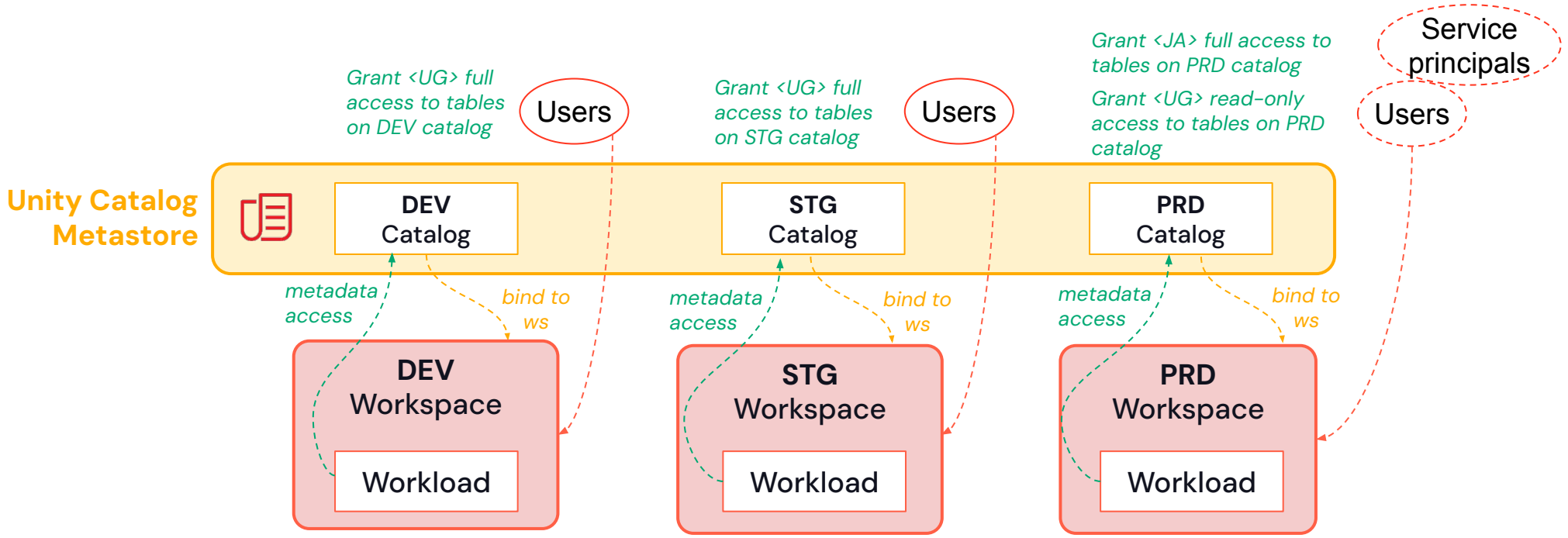
Three level namespace



Common patterns for logical data isolation



Logical Isolation Example



<JA> Service Principal to execute jobs in PRD
<UG> User Groups working in DEV, STG and PRD

Physical data isolation layout

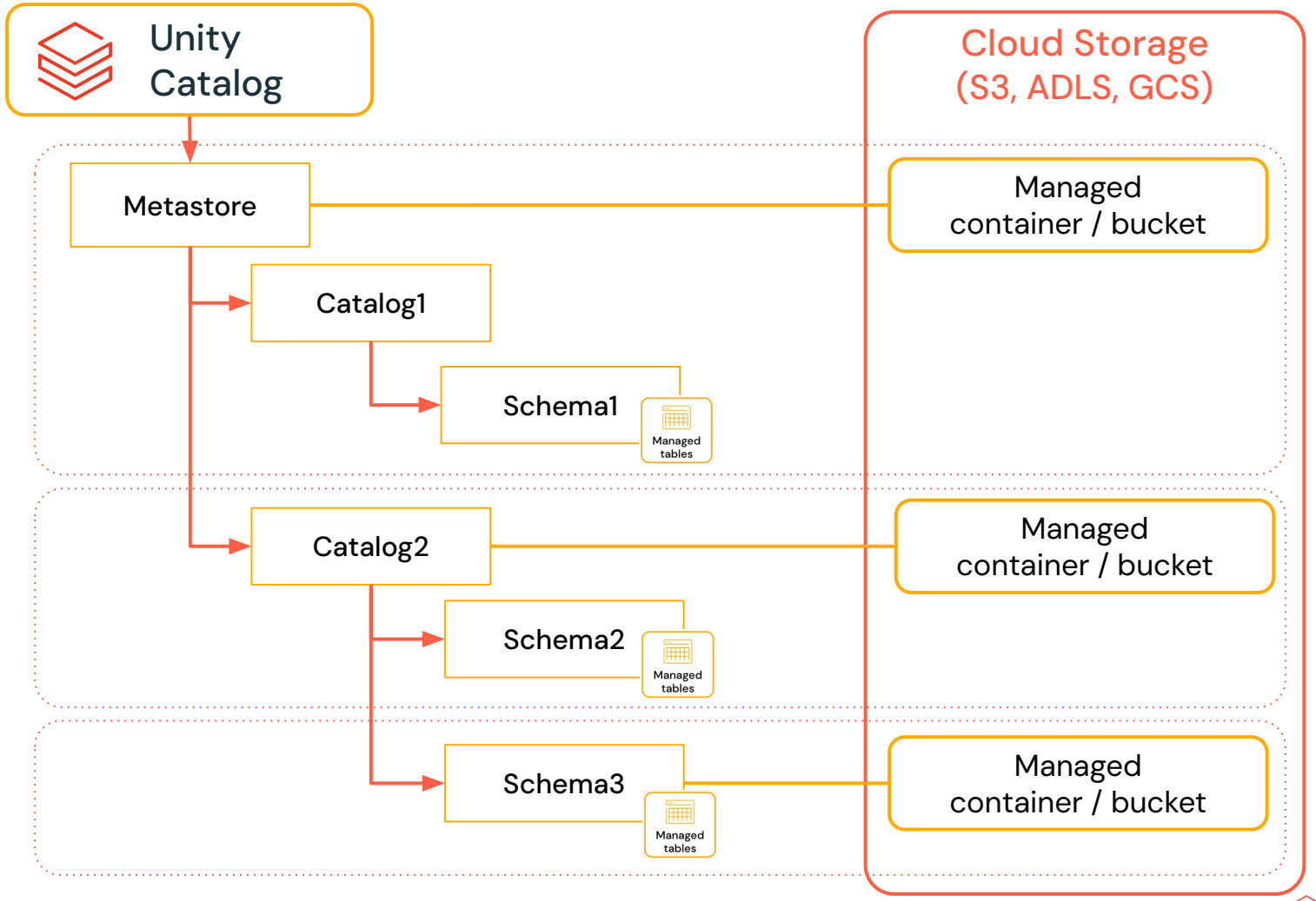
Data isolation design

Physical isolation

1) Store at the **metastore**

2) Store at the **catalog**

3) Store at the **schema**

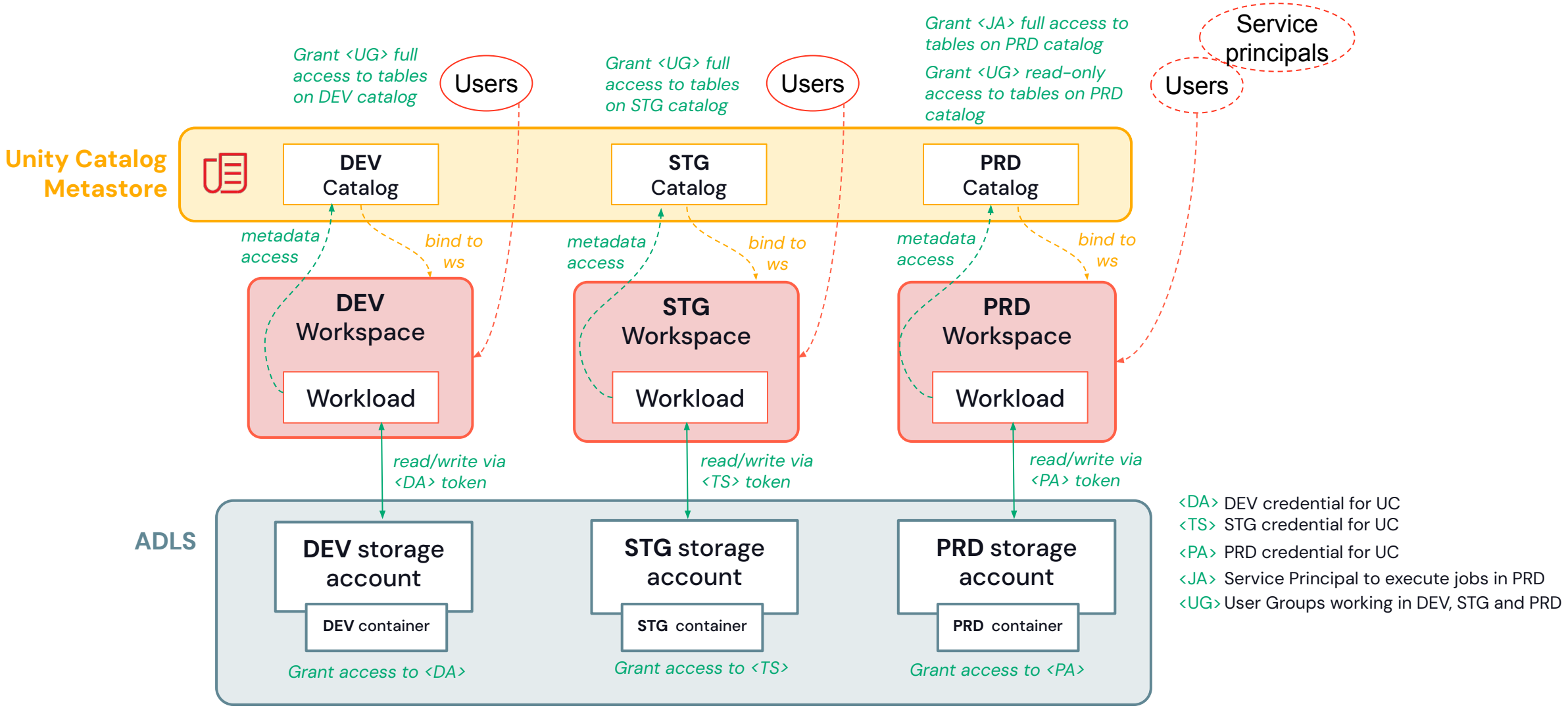


Comparison of Managed and External UC Tables

Consider the benefits of Managed tables

Characteristic	Managed	**Unmanaged** (a.k.a. "External")
Table's **Type** Property Value	"MANAGED"	"EXTERNAL"
DROP Table Behavior	**Deletes** the associated data • Generally what business analysts (SQL users) expect	Does not delete the data • May be helpful in certain use cases
Create Table Syntax	`CREATE TABLE [<catalog>.][<schema>.]<table> <column_specification>;`	`CREATE TABLE [<catalog>.][<schema>.]<table> <column_specification> LOCATION 'abfss:/cont@stacct.dfs.core.windows.net';`
Data File Location	Whichever is found first: • Location specified for the database (if specified) • Location specified for the catalog (if specified) • Metastore default storage location (if configured)	The path specified by the LOCATION keyword in your create table statement
Performance Optimizations	**Auto Tune** (In Preview)	Manually managed by the customer
Data Format Support	DELTA	DELTA, CSV, JSON, AVRO, PARQUET, ORC, TEXT

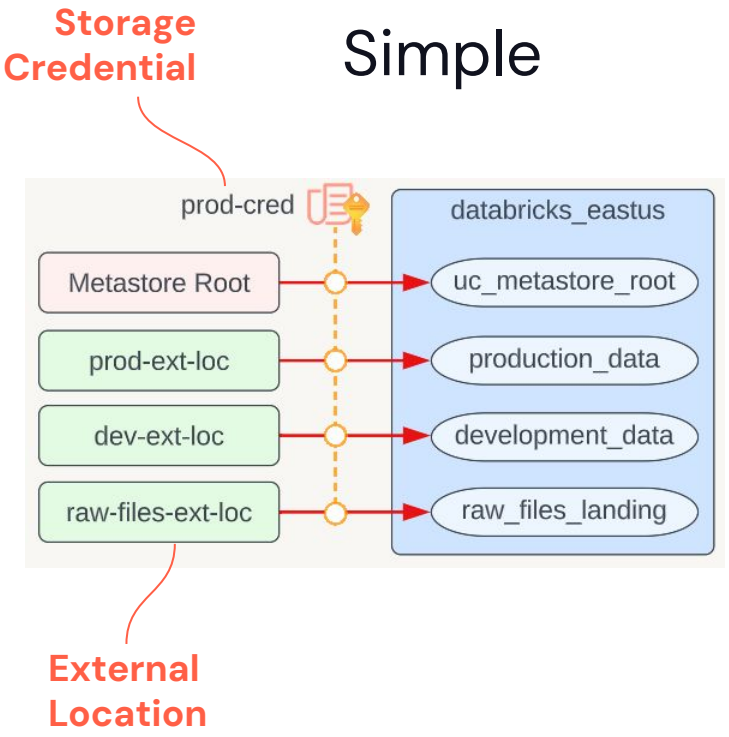
Physical Data Isolation



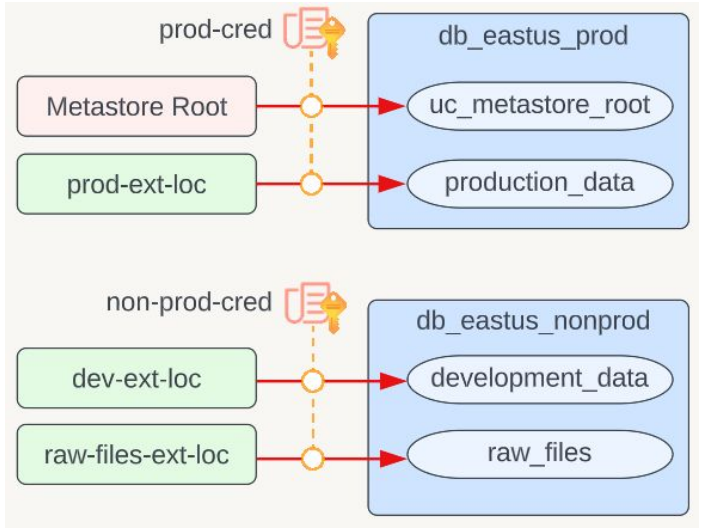
External location patterns

Your governance requirements drive the pattern

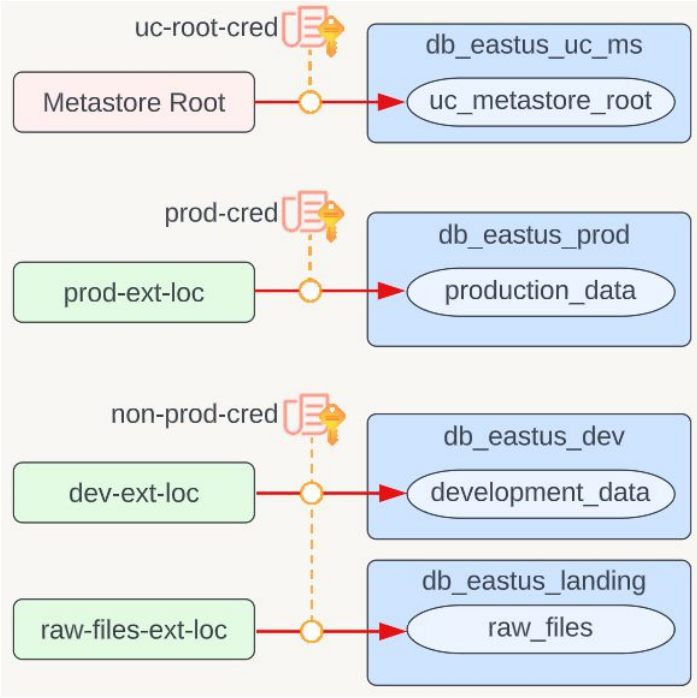
Simple



Prod/Non-Prod



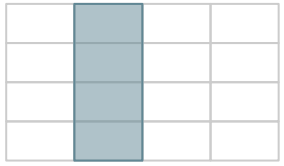
As Necessary



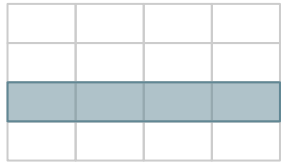
Fine-grained access control

Dynamic Views

Fine-grained access control



Limit access to columns



Obscure data

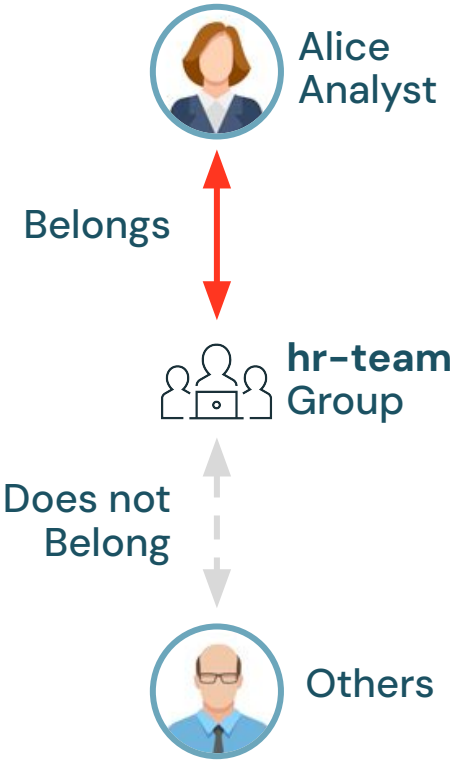
●●●●●●@databricks.com

Data Masking

Can be conditional on user or group membership through Databricks-provided functions

Row Level Security and Column Level Masking

Fine grained access control to your data based on conditions



	Name	SSN	Email	Salary	Region
1	Nancy Gibson	666-84-2234	ngibs02@example.com	84500.00	Americas
2	Andrew Roberts	900-33-2748	arobe01@example.com	110450.00	Americas
3	Nobu Yagawa	921-40-2534	nobu@example.com	98000.00	Asia Pacific
4	Ines Drechsler	966-25-0367	idrec01@example.com	156500.00	Europe Middle East
5	Arif Handal	962-62-8977	ahand0302@example.com	82500.00	Europe Middle East

Mask Column Data

	Name	SSN	Email	Salary	Region
1	Nancy Gibson	REDACTED	*****@example.com	null	Americas
2	Andrew Roberts	REDACTED	*****@example.com	null	Americas
3					
4					
5					

Filter Rows

Row and Column Security in Preview

System tables

Audit with system tables

- Actions performed against the metastore are captured in audit logs
- You can access these logs through system tables, specifically system.access.audit
- System tables must be enabled by an account admin using the UC REST API
- Leverage Databricks SQL for querying, visualization and alerting

```
SELECT
action_name as `EVENT`,
event_time as `WHEN`,
IFNULL(request_params.full_name_arg, 'Non-specific') AS `TABLE ACCESSED`,
IFNULL(request_params.commandText,'GET table') AS `QUERY TEXT`
FROM system.access.audit
WHERE user_identity.email = '{{User}}'
AND action_name IN ('createTable','commandSubmit','getTable','deleteTable')
AND datediff(now(), event_date) < 1
ORDER BY event_date DESC;
```

Questions?