**SUMMARY – DAY19**

**Name : Tejaswini Gokanakonda**

**Roll No: DE142**

**Date : 04-12-2024**

## **Unity Catalog: Unified Governance for Data and AI**

Unity Catalog is a governance layer provided by Databricks for managing data and AI assets in a secure, centralized, and organized manner on the Databricks Lakehouse platform. It streamlines policy enforcement, access control, auditing, and lineage tracking across multiple workspaces and integrates seamlessly with cloud storage systems like Azure Data Lake.

### **Why Unity Catalog?**

Organizations deal with sprawling datasets spread across multiple cloud environments and tools. Unity Catalog unifies governance by:

1. Centralizing access controls for seamless management.
2. Providing comprehensive audit logs and data lineage tracking for compliance.
3. Enhancing operational visibility with built-in discovery and monitoring tools

**Features of Unity Catalog**

### **1. Centralized Access Control**

* **Unified Policy Management:** Define data access policies once and enforce them uniformly across all connected workspaces.
* **Simplified Permissions:** Use ANSI SQL-based syntax to manage permissions.
* **Granularity:** Control access at every level: catalogs, schemas (databases), tables, views, and more.

### **2. Standards-Compliant Security**

* Built on ANSI SQL, Unity Catalog’s permission structure integrates with existing data lake standards.
* **Supports fine-grained access control with familiar SQL syntax:**
  + Example: GRANT SELECT ON TABLE sales\_data TO user1;
* **Administrators can secure assets at multiple levels:**
  + Catalog level.
  + Schema (database) level.
  + Table or view level.

### **3. Auditing and Lineage**

* **Built-In Audit Logs:**
  + Automatically records user actions (e.g., data access, query executions).
  + Helps meet compliance and security requirements.
* **Lineage Tracking:**
  + Visualizes data origin and transformations.
  + Tracks dependencies between datasets and workflows.
  + Critical for impact analysis during schema updates or debugging.

### **4. Data Discovery**

* **Search and Tagging:**
  + Allows tagging and documenting data assets with metadata.
  + Empowers users to search and locate datasets easily within a workspace.
* **Enhanced Usability:**
  + Facilitates collaboration by providing clear documentation and discoverability.

### **5. System Tables (Preview)**

* **Operational Insights:**
  + Built-in tables allow querying of operational data, including:
    - Audit logs.
    - Billable resource usage.
    - Lineage tracking.
* **Central Monitoring:**
  + Analyze patterns of resource usage and governance compliance.

## **Core Components of Unity Catalog**

Unity Catalog introduces concepts for effectively governing data stored in Databricks and cloud object storage:

### **1. Storage Credentials**

* Credentials encapsulate secure cloud storage access (e.g., Azure Managed Identity).
* Used for both managed and external data access.

### **2. External Locations**

* References storage credentials and specifies the exact cloud storage path.
* Example: Assign credentials to a specific Azure Data Lake folder.

### 

### **3. Managed Storage Locations**

* Designated storage areas within Unity Catalog for managed tables and volumes.
* Automatically manages the lifecycle of these datasets, ensuring governance and compliance.

### **4. Volumes**

* Specialized storage objects for non-tabular data, such as files in various formats.
* Organized into managed or external categories based on governance needs.

## **Unity Catalog Object Model: A Hierarchical Structure**

Unity Catalog organizes data assets using a three-level namespace: catalog.schema.table. This structure ensures scalability and logical separation of data assets.

### **1. Metastore**

* **Definition:** Top-level container for metadata and permissions.
* **Purpose:**
  + Registers metadata for all data assets (tables, views, volumes).
  + Associates permissions with data.
* **Best Practice:**
  + Create one metastore per Azure region.
  + Attach a single metastore to multiple workspaces within the same region.

### **2. Catalog**

* **Definition:** The first level of the namespace hierarchy.
* **Purpose:**
  + Groups schemas (databases) logically.
  + Assign access controls for broader organizational divisions.
* **Example Use Case:** Separate catalogs for Finance, Marketing, and Operations teams.

### **3. Schema (Database)**

* **Definition:** The second level of the namespace hierarchy.
* **Purpose:**
  + Groups tables and views within a catalog.
* **Access Requirements:**
  + Users need USE SCHEMA and USE CATALOG permissions for access.

### **4. Tables, Views, and Volumes**

* **Tables:**
  + Contain structured, tabular data.
  + **Two types:**
    1. **Managed Tables:**
       - Fully governed and stored by Unity Catalog.
       - Automatically deleted within 30 days when dropped.
    2. **External Tables:**
       - Registered to Unity Catalog but lifecycle is managed externally.
       - Supports formats like CSV, Parquet, JSON, and more.
* **Views:**
  + Read-only representations derived from one or more tables/views.
  + Support dynamic views for enforcing row- and column-level security.
* **Volumes:**
  + Store non-tabular data files.
  + Managed or external, based on governance and accessibility needs.

## **Identity and Access Management**

**Unity Catalog integrates Azure Databricks identity management to enforce permissions:**

* **Users:** Individual access to data and assets.
* **Service Principals:** Applications or automated processes requiring access.
* **Groups:** Simplify management by grouping users with similar permissions.

### **Identity Federation**

* Allows users, service principals, and groups to access Unity Catalog across multiple workspaces.
* Ensures seamless data governance in collaborative environments.

## **Governance Strategies**

1. **Managed Storage:**
   * Provides automated management for tables and volumes.
   * Ensures compliance by segregating data at different levels (metastore, catalog, schema).
2. **External Locations:**
   * Flexible access to existing cloud storage paths.
   * Ideal for large-scale integration with legacy systems or external tools.

## **Use Cases and Benefits**

### 1. **Centralized Data Governance**

* Define once, enforce everywhere.
* Simplifies policy management for large organizations.

### **2. Lineage and Auditing for Compliance**

* Automatically tracks data usage and transformations.
* Enhances debugging and regulatory reporting.

### **3. Collaboration Across Teams**

* Logical data isolation via catalogs and schemas.
* Enables cross-departmental usage without compromising security.

### **4. Scalability**

* Designed for multi-region, multi-workspace deployments.
* Supports hybrid use cases with managed and external data.

### **Real-World Example**

**An e-commerce company using Azure Databricks:**

* **Catalogs:** Separate datasets for Product, Sales, and Marketing teams.
* **Schemas:** Organize transactional data (sales tables) and reference data (customer details).
* **Lineage:** Tracks how marketing analytics depend on sales and product data.
* **Auditing:** Ensures regulatory compliance by logging access to customer data.