Unity Catalog Governance & Access Strategy for MicroStrategy Reporting

\*\*Version\*\*: 2.0

\*\*Date\*\*: June 23, 2025

\*\*Prepared By\*\*: <PlatformGovernanceTeam> - Data Engineering Team

\*\*Audience\*\*: Product Owners, Architects, <CloudAdminTeam>, App Teams

# 1. Overview

This document outlines the approved strategy for Unity Catalog creation, schema governance, and cross-team access within Databricks to support MicroStrategy reporting capabilities. It balances centralized governance by <PlatformGovernanceTeam> with autonomous table-level development by App Teams. This strategy is approved by Architecture and designed to evolve with future enterprise needs.

# 2. Ownership and Responsibilities

Roles and ownership of key components are outlined below:

|  |  |  |
| --- | --- | --- |
| Component | Owned/Performed By | Description |
| Unity Catalog Creation | <PlatformGovernanceTeam> | Created and governed centrally for all capabilities |
| Schema Creation | <PlatformGovernanceTeam> | Created as per naming convention, even if used by App Teams |
| Table Creation | App Teams | App teams create/write to tables inside their assigned schema |
| Service Principal Creation | App Teams | Created with proper tags and naming conventions |
| SP Access Grant | <CloudAdminTeam> (on approval) | Access granted based on approval from <PlatformGovernanceTeam> |
| Governance & Audit | <PlatformGovernanceTeam> | Quarterly review of access and tagging compliance |

# 3. Governance Philosophy

Unity Catalog and schemas are centrally governed by <PlatformGovernanceTeam> to ensure naming consistency, environment segregation, and controlled access. App Teams have freedom to develop and manage their own tables within assigned schemas, but must follow defined approval workflows and tagging standards.

# 4. Schema Naming Convention

Schemas follow a standard naming pattern based on capability and environment. For example:

- `reporting\_customer360\_dev`

- `cleansed\_<another\_capability>\_prod`

# 5. Service Principal Strategy

Key points for SP usage across Unity Catalogs:

- Each App Team creates its own SP per environment.

- SP must be tagged with `OwnerTeam`, `Capability`, `Environment`, etc.

- SP access to schema must be approved by <PlatformGovernanceTeam>.

- <CloudAdminTeam> will grant `USAGE`, `CREATE`, `SELECT`, and `INSERT` to the approved schema.

# 6. Multi-Catalog Access by SP

A single SP can be granted access to multiple Unity Catalogs across environments, provided explicit access grants are approved by <PlatformGovernanceTeam>. No automatic cross-catalog access is allowed.

Example SQL:

GRANT USAGE ON CATALOG uc\_mstr\_customer360 TO `sp\_customer360\_prod`;  
GRANT USAGE ON SCHEMA uc\_mstr\_customer360.reporting\_customer360\_prod TO `sp\_customer360\_prod`;  
GRANT CREATE, SELECT, INSERT ON SCHEMA uc\_mstr\_customer360.reporting\_customer360\_prod TO `sp\_customer360\_prod`;

# 7. Tagging Strategy

All resources must be tagged to ensure traceability and cost attribution.

|  |  |
| --- | --- |
| Tag Key | Description |
| OwnerTeam | Team responsible for the SP or resource |
| Capability | MicroStrategy capability name |
| Environment | Environment: DEV, DISC, or PROD |
| CreatedBy | Entity who provisioned the resource |
| Purpose | Usage purpose: reporting, ingestion, etc. |
| DataSensitivity | Data classification: PII, Confidential, etc. |

# 8. Checklist for Onboarding New Capability

The following checklist must be completed before development begins:

- [ ] App Team submits request for new capability

- [ ] <PlatformGovernanceTeam> create Unity Catalog and schema

- [ ] App Team creates SP and tags it appropriately

- [ ] App Team submits SP ID to <PlatformGovernanceTeam>

- [ ] <PlatformGovernanceTeam> approve and submit to <CloudAdminTeam>

- [ ] <CloudAdminTeam> grant required schema access

- [ ] App Team can now write tables inside assigned schema

# 9. Disclaimer

This document reflects the current governance model and architectural decisions as of June 23, 2025. These practices are subject to refinement as the platform evolves and more use cases emerge. Any changes will be discussed, documented, and approved by the architecture and governance teams.