Migration Analysis Report

Overview

For a better understanding of migrating into Unity Catalog, this article will provide a great overview of the steps required for a smooth migration.

This article works as a complementary read to understand the different flavors of table migration.

Tables and Views

The workspace has **1433** EXTERNAL tables with DELTA format, in EXTERNAL storage.

Recommended migration options:

- To use DELTA format on Unity Catalog as a(n) EXTERNAL table, you must migrate using the SYNC approach.
- To use DELTA format on Unity Catalog as a(n) MANAGED table, you must migrate using the DEEP CLONE (keeps metadata)/CTAS approach.

The workspace has **79** MANAGED tables with DELTA format, in EXTERNAL storage.

Recommended migration options:

- To use DELTA format on Unity Catalog as a(n) EXTERNAL table, you must migrate using the 1. set spark.sts.sync.command.enableManagedTable=true; 2. SYNC command approach.
- To use DELTA format on Unity Catalog as a(n) MANAGED table, you must migrate using the DEEP CLONE (keeps hist)/CTAS approach.

The workspace has **27** MANAGED tables with DELTA format, in DBFS ROOT storage.

Recommended migration options:

- To use DELTA format on Unity Catalog as a(n) EXTERNAL table, you must migrate using the DEEP CLONE (keeps metadata)/CTAS approach.
- To use DELTA format on Unity Catalog as a(n) MANAGED table, you must migrate using the DEEP CLONE (keeps metadata)/CTAS approach.

The workspace has **12** VIEW .

Recommended migration options:

- CREATE VIEW using the view definition provided from hive_metastore.

The workspace has **8** EXTERNAL tables with PARQUET format, in EXTERNAL storage.

Recommended migration options:

- To use PARQUET format on Unity Catalog as a(n) EXTERNAL table, you must migrate using the SYNC approach.
- To use DELTA format on Unity Catalog as a(n) MANAGED table, you must migrate using the DEEP CLONE/CTAS approach.

The workspace has **1** EXTERNAL tables with CSV format, in EXTERNAL storage.

Recommended migration options:

- To use CSV/JSON/AVRO/ORC/TEXT/XML format on Unity Catalog as a(n) EXTERNAL table, you must migrate using the SYNC approach.
- To use DELTA format on Unity Catalog as a(n) MANAGED table, you must migrate using the CTAS approach.

The workspace has **1** EXTERNAL tables with CSV format, in Databricks Demo Dataset storage.

Recommended migration options:

- No specific recommendations available

The workspace has **1** EXTERNAL tables with CSV format, in DBFS ROOT storage.

Recommended migration options:

- To use CSV/JSON/AVRO/ORC/TEXT/XML format on Unity Catalog as a(n) EXTERNAL table, you must migrate using the CTAS approach.
- To use DELTA format on Unity Catalog as a(n) EXTERNAL table, you must migrate using the CTAS approach.
- To use DELTA format on Unity Catalog as a(n) MANAGED table, you must migrate using the CTAS approach.

The workspace has **1** MANAGED tables with PARQUET format, in DBFS ROOT storage.

Recommended migration options:

- To use PARQUET format on Unity Catalog as a(n) EXTERNAL table, you must migrate using the CTAS approach.
- To use DELTA format on Unity Catalog as a(n) EXTERNAL table, you must migrate using the DEEP CLONE/CTAS approach.
- To use DELTA format on Unity Catalog as a(n) MANAGED table, you must migrate using the DEEP CLONE/CTAS approach.

For more information on the migration approaches, refer to the documentation DEEP Clone, CTAS, SYNC, and CREATE View.

Important Guidelines for Data Migration

- SYNC requires UC external location to exist beforehand, pointing to the location or a parent location.
- SYNC forces maintaining the same table name. If table name change is required, use CTAS instead of SYNC.
- SYNC when the source table HMS type = MANAGED requires some extra steps. See steps above.
- SYNC with a table with mount location as source uses the real location (not the mnt point) for the new location. This is nice because we want to sunset those mount points.
- UNSUPPORTED Tables indicate that they may break downstream or upstream dependencies if not accounted for.
- All data in DBFS ROOT needs to be moved somewhere else.
- DEEP CLONing a parquet table creates a Delta Table, not parquet. This may not be what the customer wants in some cases. For example, if some external dependency assumes parquet format.
- Scala code to convert HMS managed to HMS external is in the Appendix section of this article. Just need to USE CATALOG hive_metastore; before running the Scala.please read.
- DEEP CLONE keeps metadata while >CTA loses it, creating a brand-new table.
- DEEP CLONE also preserves the existing partitioning scheme. If that is not the desired outcome, please use CTAS.
- If data is in ADLS or wasb, then CTAS is required (Azure only).
- If the HMS table is non-Delta+partitioned, and you register it as UC external, there will be a performance penalty because UC does not store external table partition metadata for non-Delta tables. More info here and here.

Storage Locations

- Your Workspace currently shows **1** external locations being used.
- Do not mount storage accounts to **DBFS** that are being used as External Locations.
    - As this is not governed by UC, you would allow your users to circumvent ACLs.
    - Mounting External Locations is blocked on shared, but not on single-user clusters.
    - Limit the roles/service principals that have access to buckets/accounts governed by External Locations in Unity Catalog.
- Follow public documentation requirements for setting up ADLS and external locations on your Workspace. Also, consider using Databricks Volumes.
- Do not use **EXTERNAL LOCATIONS** for anything other than creating tables and do not use path-based access for data science and other non-tabular data use cases.

Mount Points

- Your Workspace currently shows **8** locations being used as mount points.
- Databricks now recommends using Unity Catalog for managing all data access instead of the legacy mount access pattern. See Connect to cloud object storage using Unity Catalog.
- After migrating the tables, consider unmounting external data locations, as Databricks no longer supports mounting them to Databricks Filesystem.

Clusters, Jobs, Submits

Clusters

- **1** clusters(s) found with **cluster type not supported : LEGACY_PASSTHROUGH** setting.

Jobs

- **3** jobs(s) found with **not supported DBR: 8.1.x-scala2.12** setting.
- **2** jobs(s) found with **Uses azure service principal credentials config in Job cluster.** setting.
- **1** jobs(s) found with **not supported DBR: 10.4.x-scala2.12** setting.

Submit Run

- **1** submit run(s) found with **cluster type not supported : LEGACY_PASSTHROUGH** setting.

The Assessment Finding Index will explain and provide recommendations for each of the findings.

Other compute-related recommendations:

- Upgrade cluster runtime to latest LTS version (e.g. 14.3 LTS).
- Migrate users with GPU, Distributed ML, Spark Context, R type workloads to Assigned clusters. Implement cluster policies and pools to even out startup time and limit upper cost boundary.
- Upgrade SQL only users (and BI tools) to SQL Warehouses (much better SQL / Warehouse experience and lower cost).
- For users with single node python ML requirements, Shared Compute with `%pip install` library support or Personal Compute with pools and compute controls may provide a better experience and better manageability.
- For single node ML users on a crowded driver node of a large shared cluster, will get a better experience with Personal Compute policies combined with (warm) Compute pools.

Code compatibility

- The table *Workflow migration problems* in the UCX Assessment Main Dashboard will assist with verifying if workflows and dashboards are Unity Catalog compatible. It can be filtered on the path, problem code and workflow name.

  Each Row:

  - Points to a problem detected in the code using the code path, query or workflow & task reference and start/end line & column.
  - Explains the problem with a human-readable message and a code.

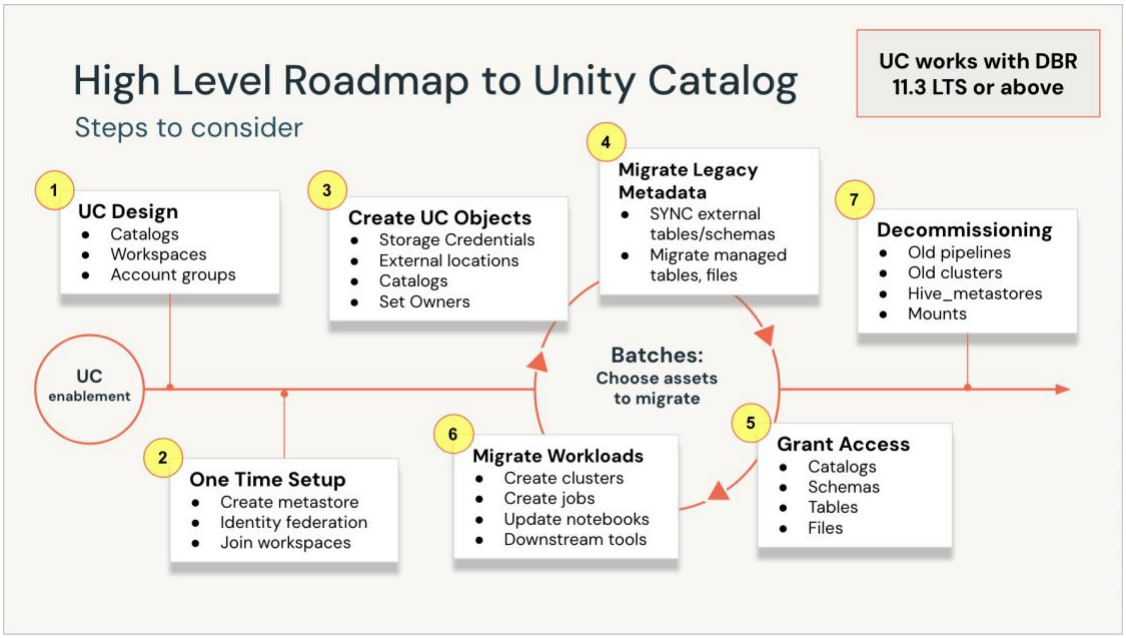
  For additional information related to code compatibility problems, refer to Linter message Codes

Permissions/Grants/Groups

- You have some grants that will require migration. You must reproduce those grants using UC permission mechanisms.
- Groups in your Workspace where the source is equal to Workspace will need to be migrated to the account level.
- Consider adopting a grant strategy that emphasizes assigning permissions to groups rather than individuals.
- Additionally, consider managing those groups at the account level and syncing them with your IDP through the SCIM API if it is not currently used.

Additional Guidance

High Level Roadmap to Unity Catalog



Automating Unity Catalog Upgrade Workflows with UCX Blog.

UC Migration, **Video Series** (final 8 videos) can provide some guidance about the migration processes. We highly encourage you to go through them.

In addition to the recommendations, we are sharing with you an **upgrade plan template** and a **matrix summary for your table scenarios**.