Data Governance z Databricks Unity Catalog - raport z wdrożenia

Tomasz Krawczyk

Data Architect/ Data Engineer

SQL Day

16 edycja konferencji SQLDay

Data Community

13-15 maja 2024, WROCŁAW + ONLINE

partner platynowy

lingaro

partner złoty

DXC TECHNOLOGY

VOLVO

software one

partner srebrny

accenture

Nordcloud an IBM Company

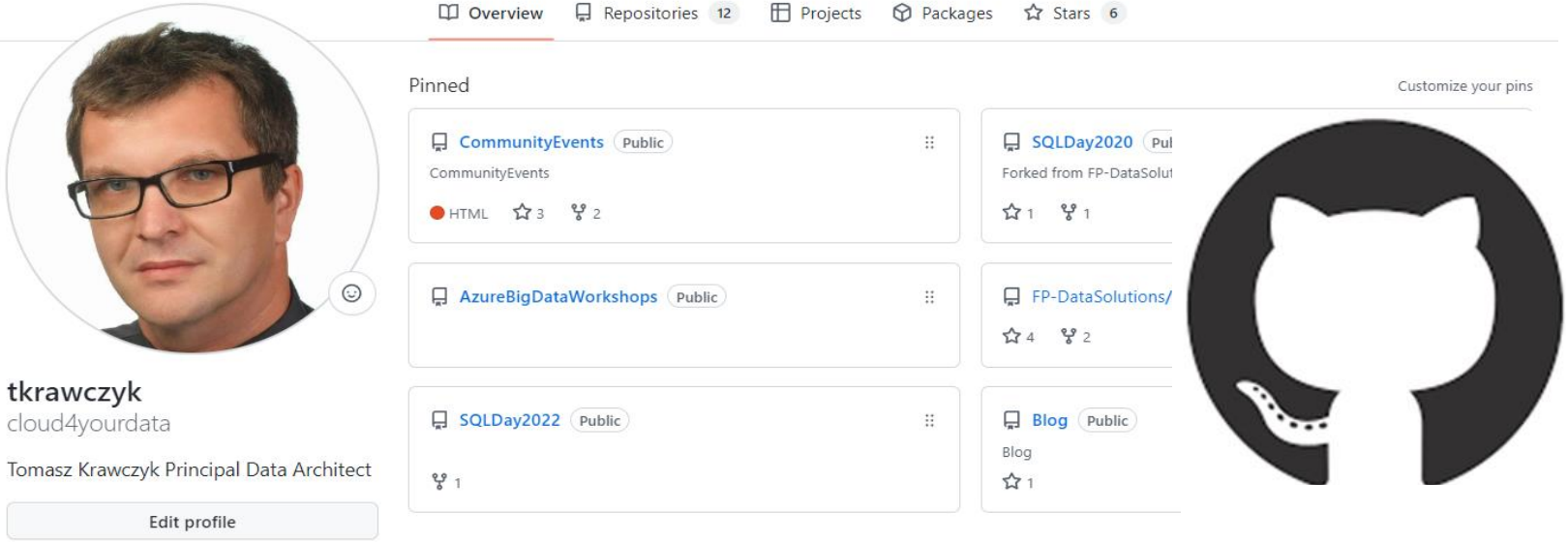
Capgemini

TECHNOLOGY INNOVATION DATA KNOWLEDGE hdk

elitmind think bright

C&F

Data
Community



Data Architect/Data Engineer

Future Processing Data Solutions (>10 years)

Data Lakehouse Enthusiast

https://github.com/fp-datasolutions

https://github.com/cloud4yourdata/CommunityEvents

tkrawczyk@future-processing.com

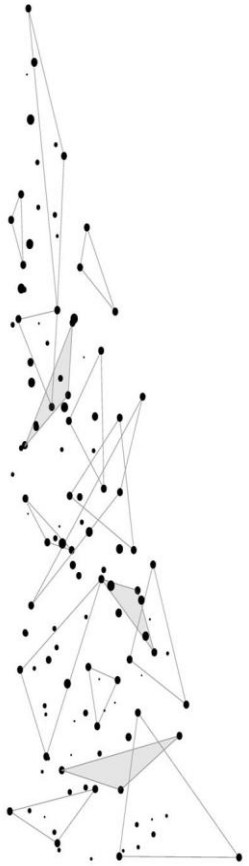
tomasz.k.krawczyk@gmail.com

Agenda

- Overview of Data Governance
- Databrick Unity Catalog and Data Governance
  - Unity Catalog Key Concepts
- Migration to Unity Catalog
  - Migration Strategy
    - Assess current data architecture
    - Prepare the Unity Catalog environment
    - Migration …
- Unity Catalog - Benefits
- Demo(s)
- Q&A

Data Governance

What is Data Governance?



"**Data Governance** is a **principled approach** to **managing data** during its **life cycle**, from acquisition to use to disposal."

"You can think of **Data Governance** as introducing **Rules**, **Processes**, and **Accountability** that allow the organization to better manage the **availability, usability, security and integrity** of **corporate data sources**"

DAMA-DMBOK2

Data Governance by Databricks

Four key functional areas:

Data Access Control

Control who has access to which data

Data Access Audit

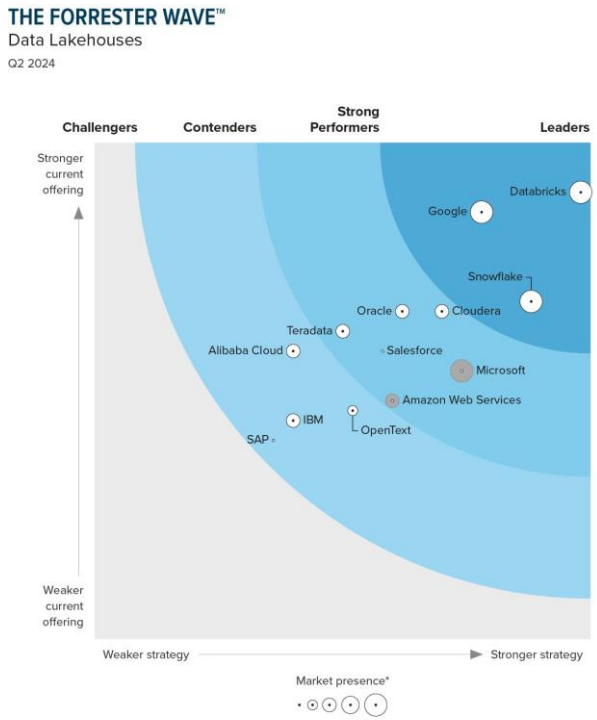
Capture and record all access to data

Data Lineage

Capture upstream sources and downstream consumers

Data Discovery

Ability to search for and discover authorized assets

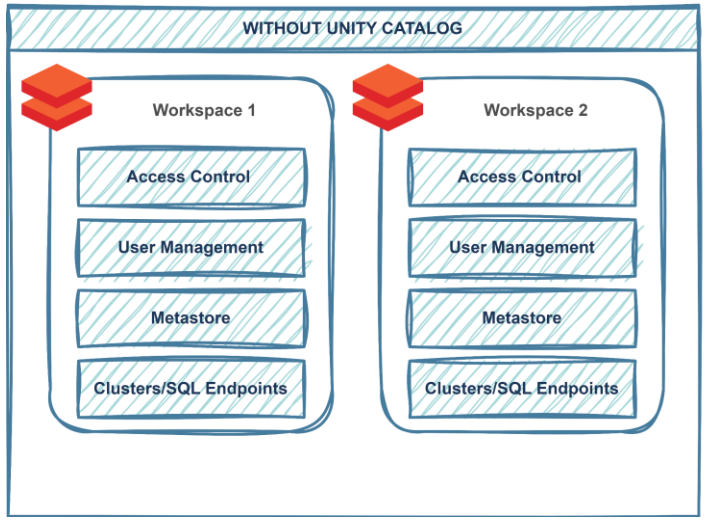


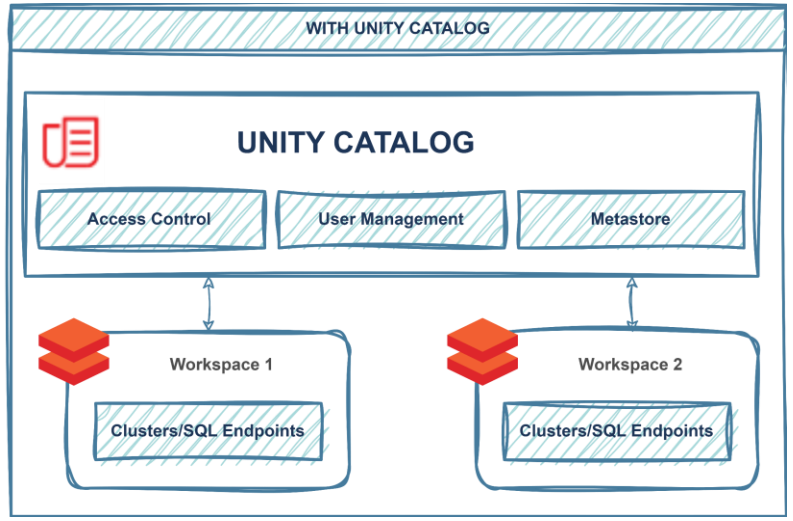
Data Governance by Databricks

Challenges (before Unity Catalog):

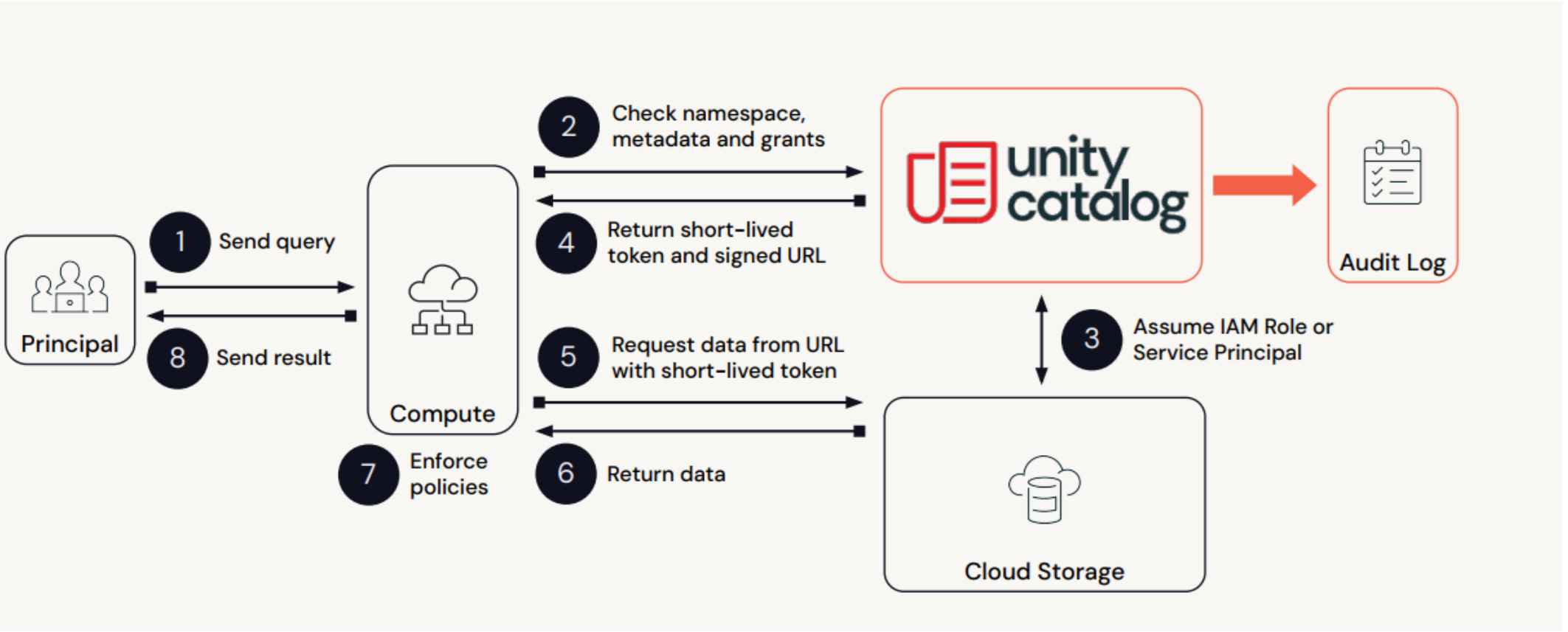
- **Complexity due to Permissions on Files**
- **Permissions on Objects (Databases, Tables, Views,…)**
- **Permissions on Columns, Rows**
- **Permissions on ML models, dashboards, features**
- **Auditing**
- **Local permission management**

Unity Catalog provides **centralized access control**, **auditing**, **lineage**, and **data discovery** capabilities across Databricks workspaces.



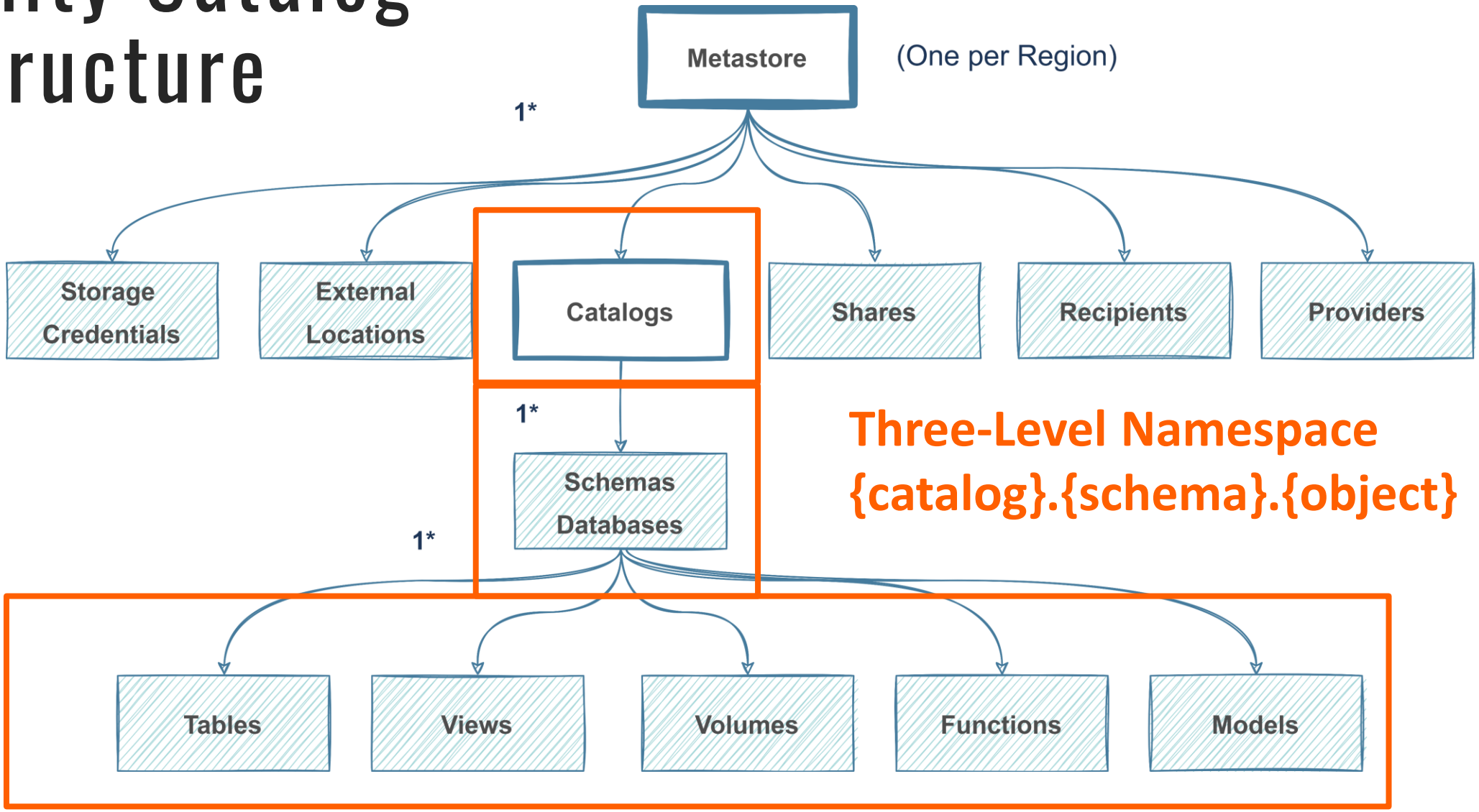


UC- Query Lifecycle



Source: Databrikcs

Unity Catalog Structure



Three-Level Namespace
{catalog}.{schema}.{object}

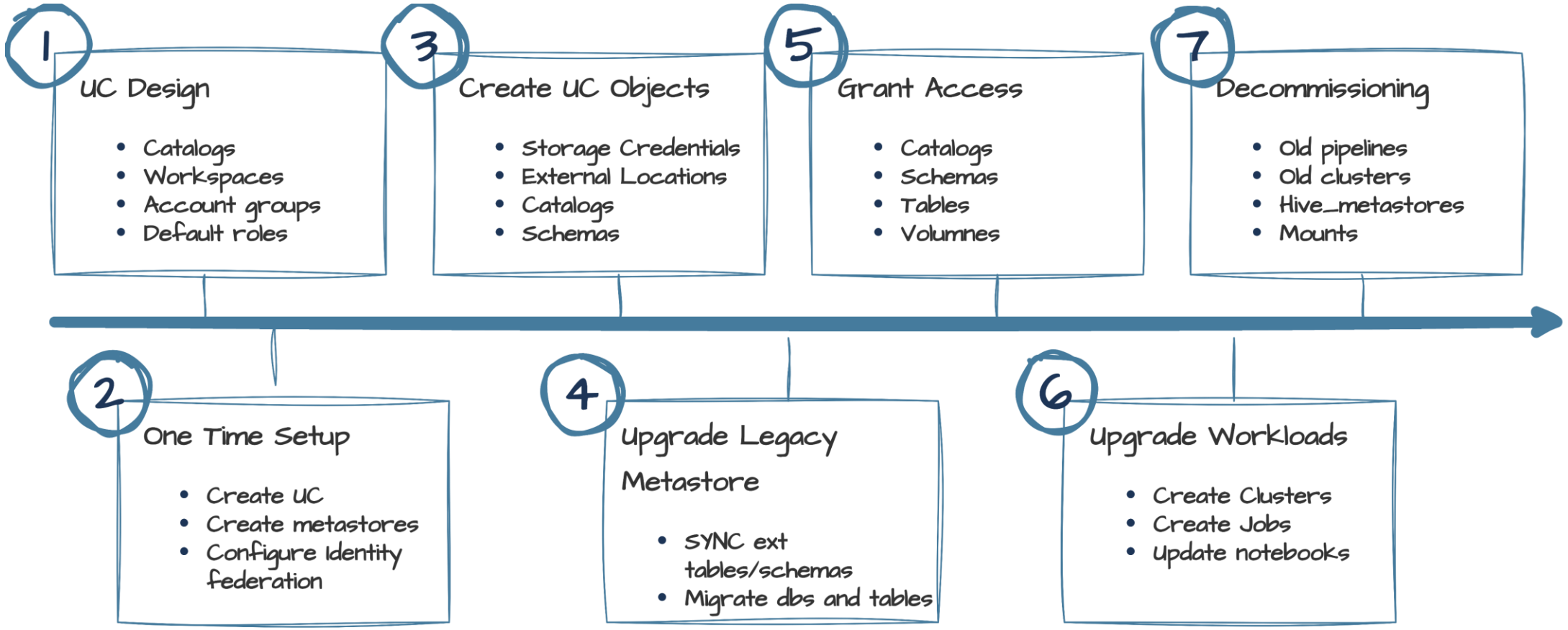
Migration to Unity Catalog

Strategy for migrating to Unity Catalog:

- **Assess current data architecture**
- Define your goals and requirements
- Plan the migration
- **Prepare the Unity Catalog environment(s)**
- **Migrate the data and processes**
- Test the migration
- Optimize the performance
- Monitor and maintain the Unity Catalog environment

High Level Roadmap to Unity Catalog

Steps to consider for a full upgrade



Assess data architecture

Assess data architecture:

- **Identify the data sources/stores**
- **Identify the data pipelines**
- Identify the data consumers
- Analyze the performance
- Analyze the scalability
- **Analyze the security**
- Analyze the compliance (GDPR, HIPAA)

Useful Tools:

- UCX
  - https://github.com/databrickslabs/ucx
- Databricks REST API
  - https://docs.databricks.com/api/workspace/introduction
- Databricks SDK
  - https://docs.databricks.com/en/dev-tools/sdk-python.html
- Databricks CLI
  - https://docs.databricks.com/en/dev-tools/cli/index.html

Migrate to Unity Catalog

Migration approaches:

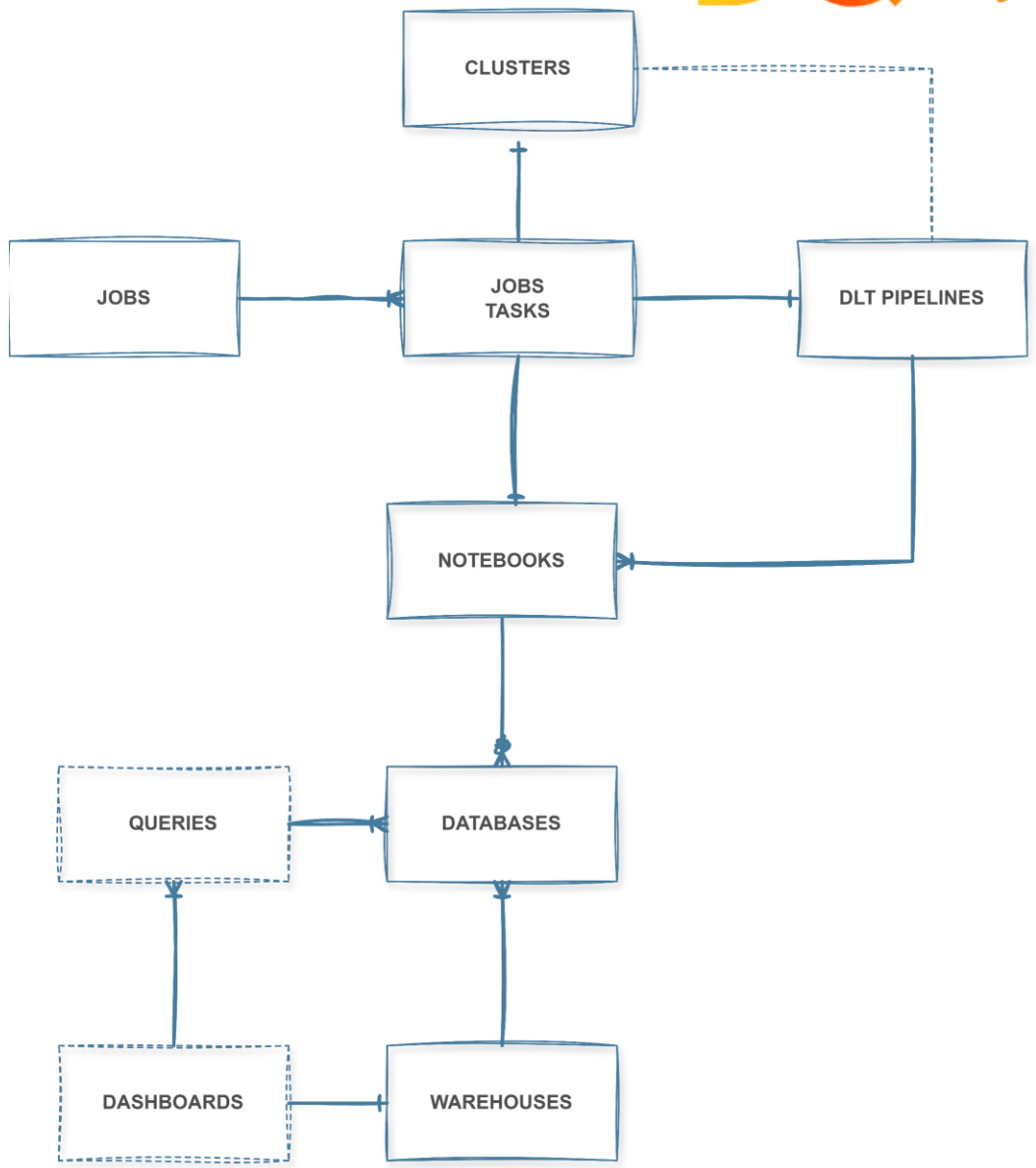
- **“All in one”**

  This approach involves migrating all the data and processes at once.

- **“Small steps method”**
- This method involves migrating data and applications in stages or increments. This allows for testing and validation after each step, reducing the risk of downtime or data loss.

Challenges:

- **Objects dependencies**



Assess data architecture

- **Databases:**
  - *SHOW DATABASES*
- **Jobs**
  - Databricks SDK
- **DLT Pipeline**
  - Databricks SDK
- **Notebooks**
  - Git Repo
- **Notebooks-Databases**
  - Python Script

```python
print("Generating audit log for Jobs...")
w = sdk.WorkspaceClient(host=db_host,token=db_token)
jobs = []
for j in w.jobs.list():
    ji = w.jobs.get(j.job_id).as_dict()
    jdetails = json.dumps(ji)
```

```python
print("Generating audit log for DLTPipelines...")
w = sdk.WorkspaceClient(host=db_host,token=db_token)
pipelines =[]
for p in w.pipelines.list_pipelines():
    pipeline = w.pipelines.get(p.pipeline_id)
    pipeline_info = pipeline.as_dict()
    pipeline_info_json = json.dumps(pipeline_info)
```

```python
def search_file(file_path, databases:list):
    result =[]
    content = None
    with open(file_path,'r') as f:
        content = f.read()
    #print(content)
    if content is not None:
        for db in databases:
            if content.find(db)>0:
                result.append([db,normalize_notebook_path(file_path)])
    return result
```

Databricks Unity Catalog on Azure

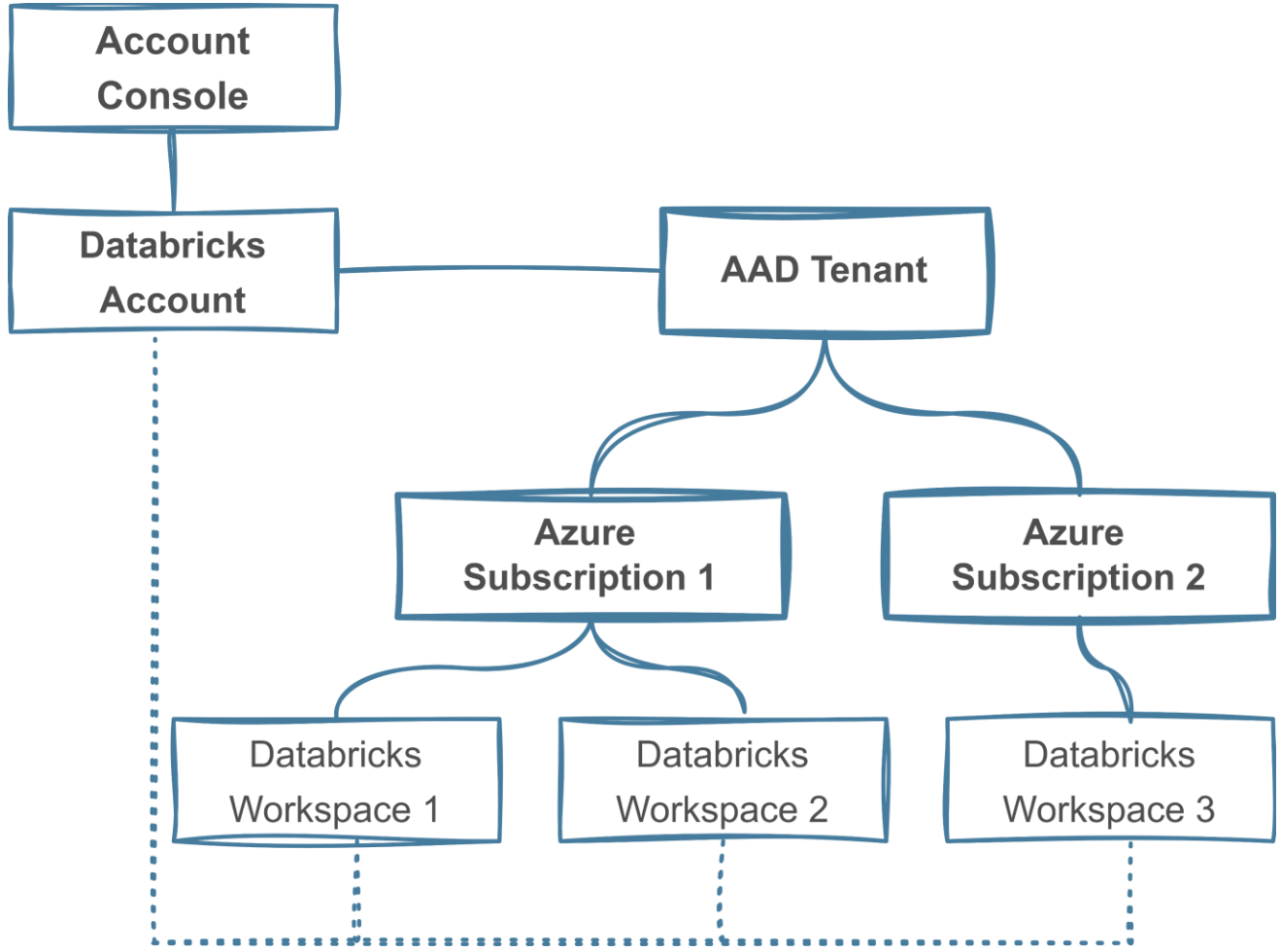
Account Console:

https://accounts.azuredatabricks.net/

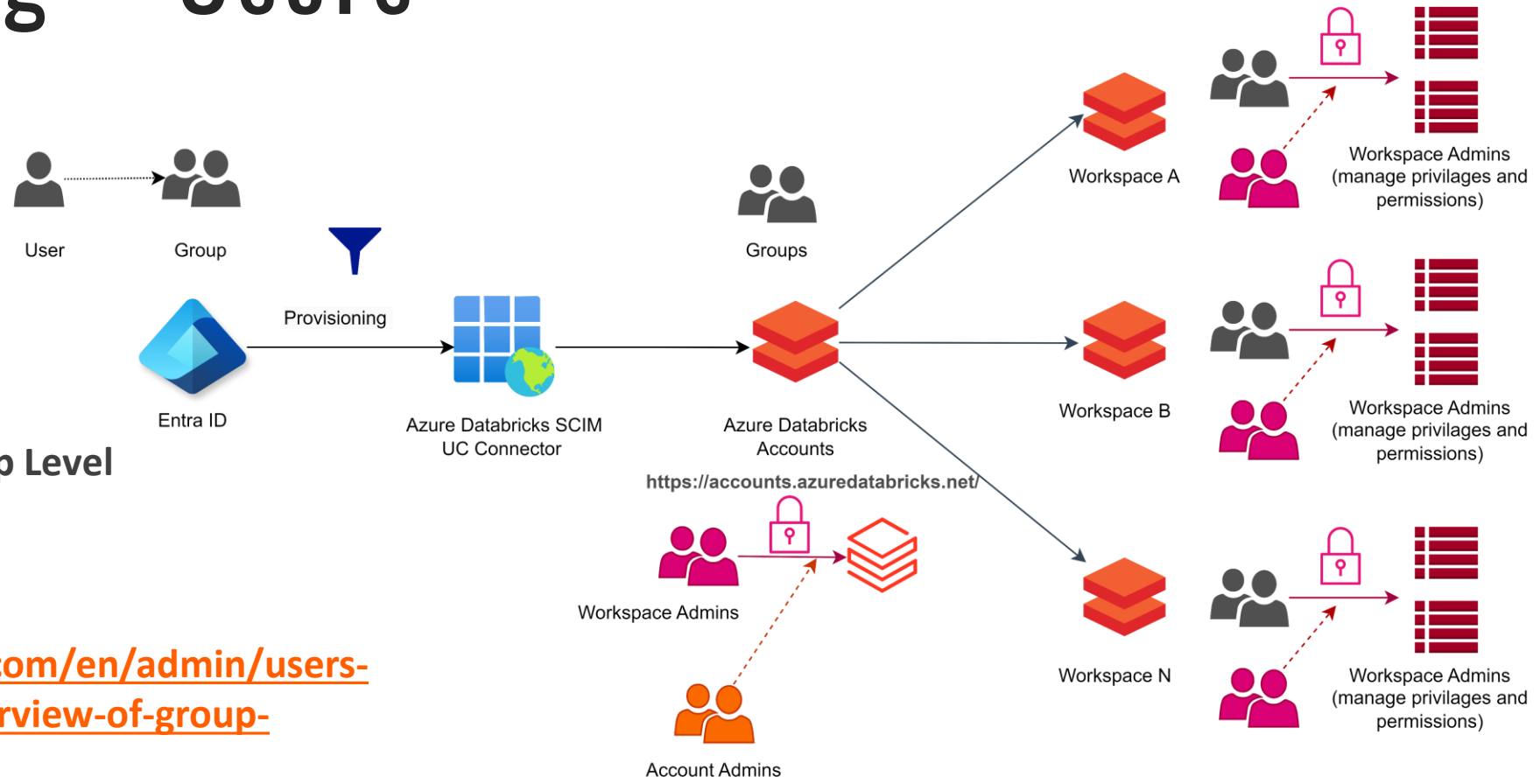
Azure AD Global Administrator role

Access Connector for Azure Databricks
- is an Azure resource that lets you connect managed identities to an Azure Databricks account.

Managed storage
- location in an Azure Data Lake Storage Gen2 container to store data and metadata



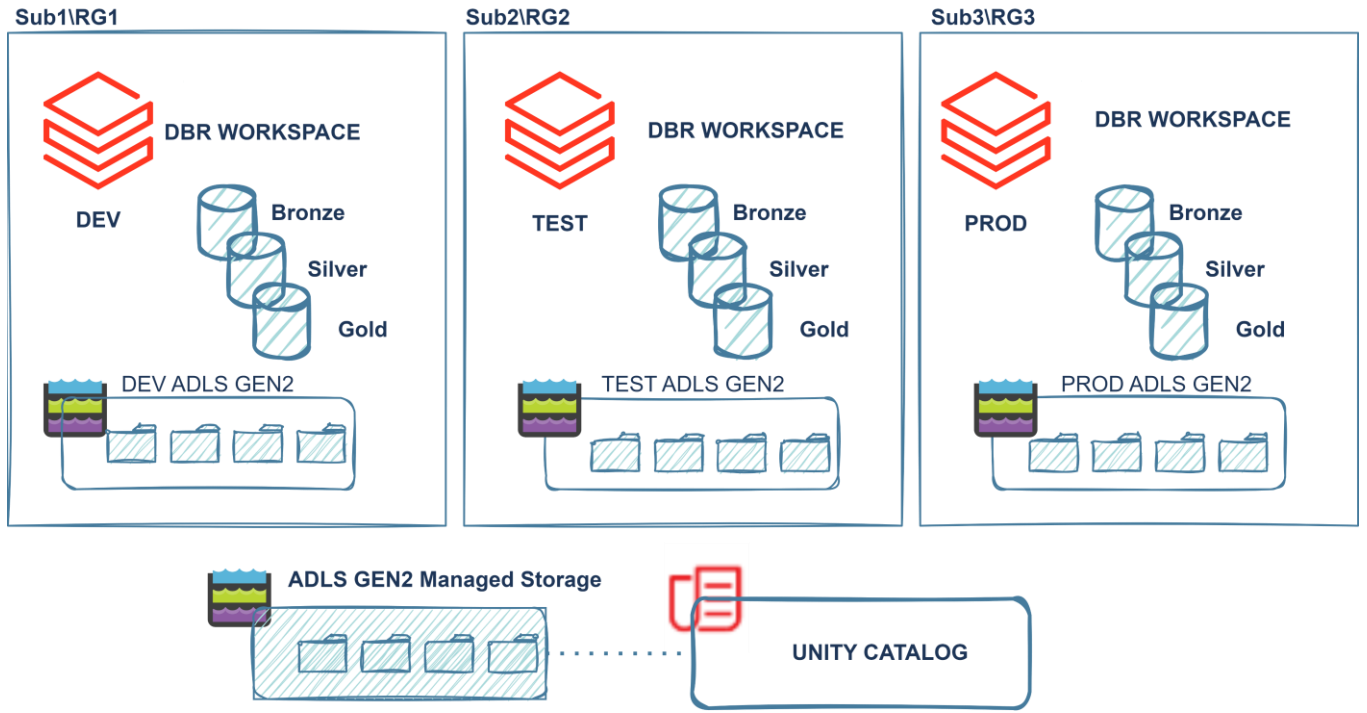
Unity Catalog – Users Management



- ENTRA ID – Groups and Users
- Permissions only on the Group Level
- Account Admins
- Workspace Admins
- No Local Groups
  - https://docs.databricks.com/en/admin/users-groups/groups.html#overview-of-group-management
  - IS_ACCOUNT_GROUP_MEMBER (account level)
  - ~~IS_MEMBER~~ (workspace local group)

Unity Catalog DBR Workspaces

- **DBR Workspaces**
  - Technical (Dev, Uat, Prod …)
  - Business (Sales, Marketing…)
- DBR Workspace
  - Storage
    - Containers(e.g. Silver, Gold)
    - Mounting points
    - Databases
- **Migration Strategy**
  - Use existing structure and resources
  - Create new



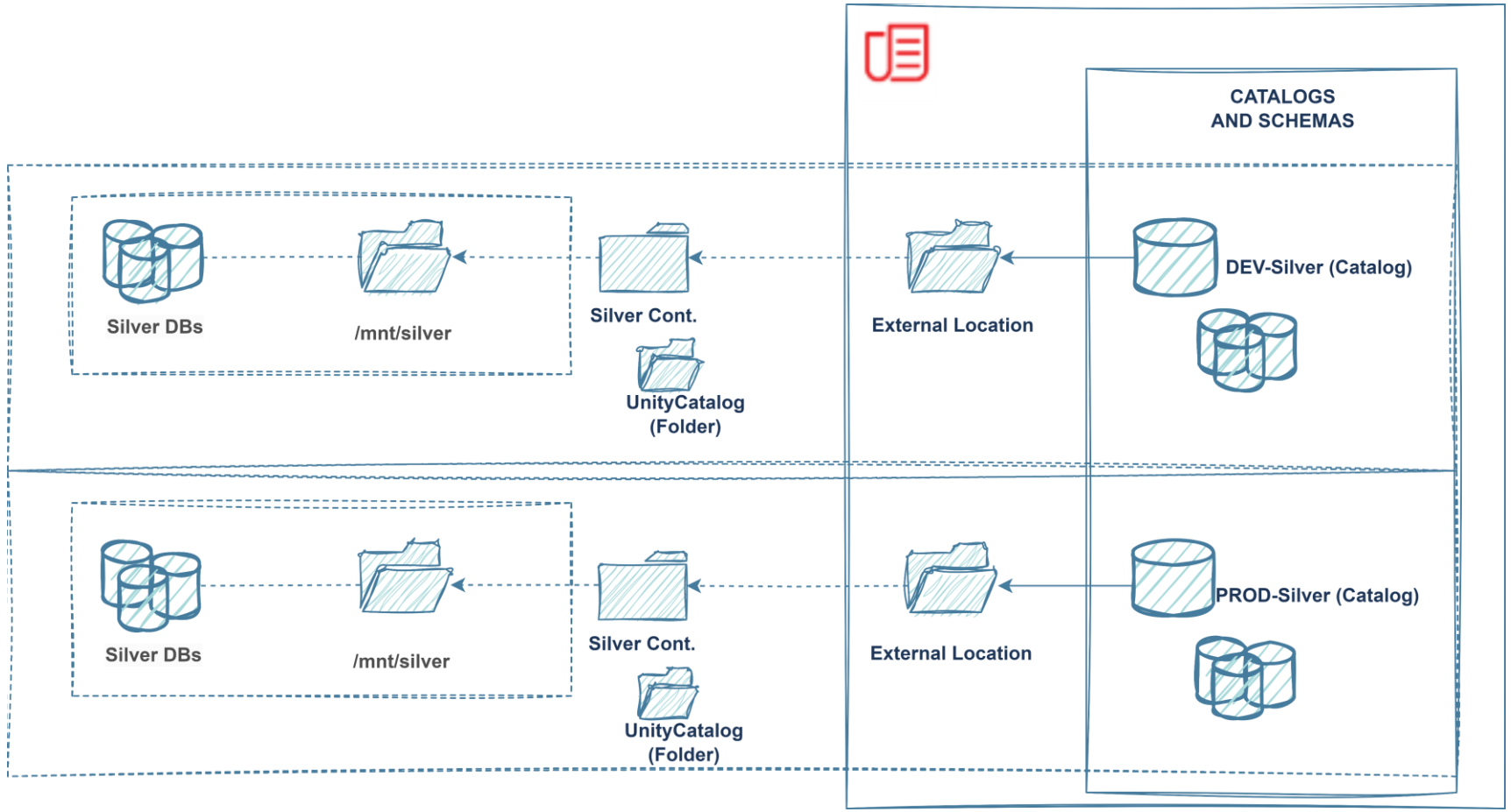
Prepare the Unity Catalog Structure

OLD WORLD

- Storage
  - Containers
- DBR mounting point
  - One per Zone*
  - Databases
    - Location /mnt/

UNITY CATALOG

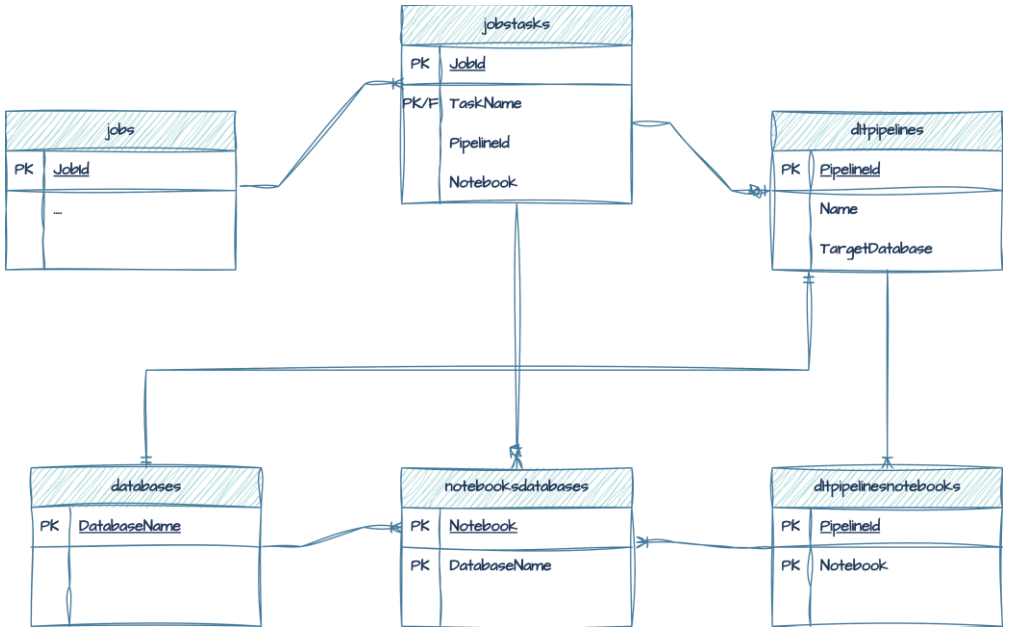
- Storage
  - Container
    - Folder Unity Catalog
- DBR external location
  - Catalog (external location)
  - DBS in Catalog



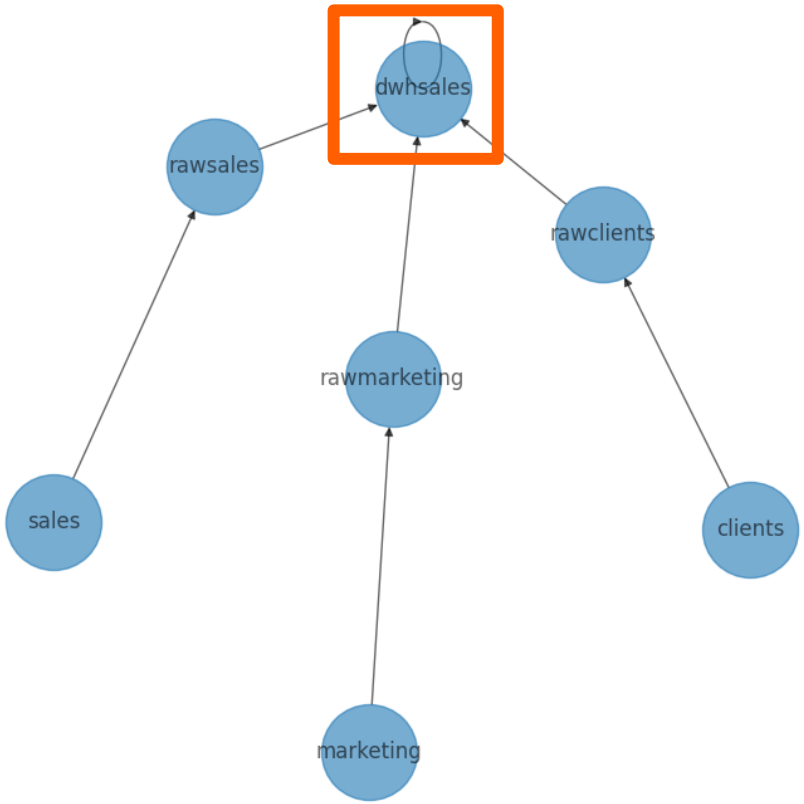
Migration to Unity Catalog

databricks

Databricks SDK for Python



Assessment/Audit Objects



Databases Dependencies Graph

UNITY CATALOG

- **Three-Level Namespace**
  - **{catalog}.{schema}.{object}**
- Access to Hive metastore objects
  hive_metastore.{DBName}.{ObjectName}

Data Migration

Tools:

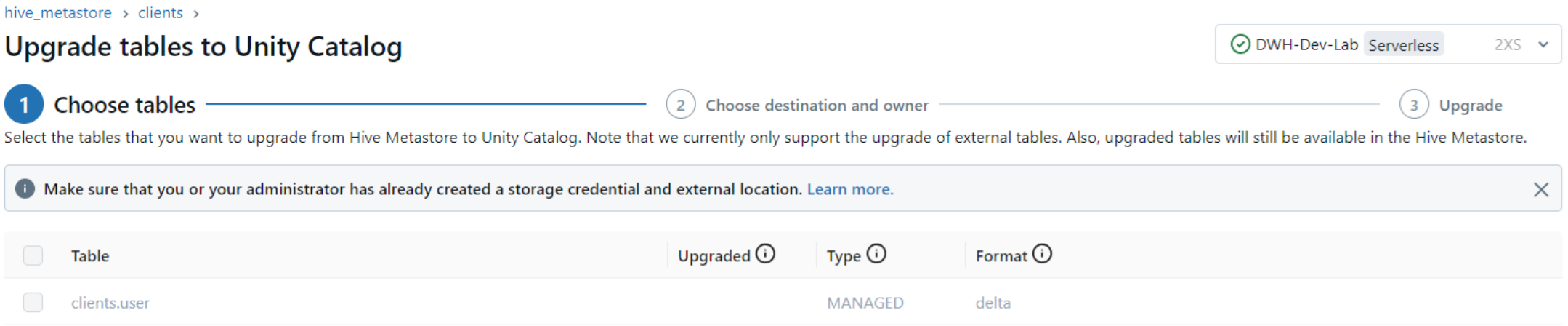
- UCX
  - https://github.com/databrickslabs/ucx
- Unity Catalog upgrade wizard
- SYNC SQL command
- CREATE TABLE CLONE SQL command

Copy

```
CREATE TABLE [IF NOT EXISTS] table_name
   [SHALLOW | DEEP] CLONE source_table_name [TBLPROPERTIES clause] [LOCATION path]
```

Copy

```
[CREATE OR] REPLACE TABLE table_name
   [SHALLOW | DEEP] CLONE source_table_name [TBLPROPERTIES clause] [LOCATION path]
```



Processes Migration

Compute:

- Migrate existing all-purpose cluster
  - Lakeguard –isolation for shared clusters

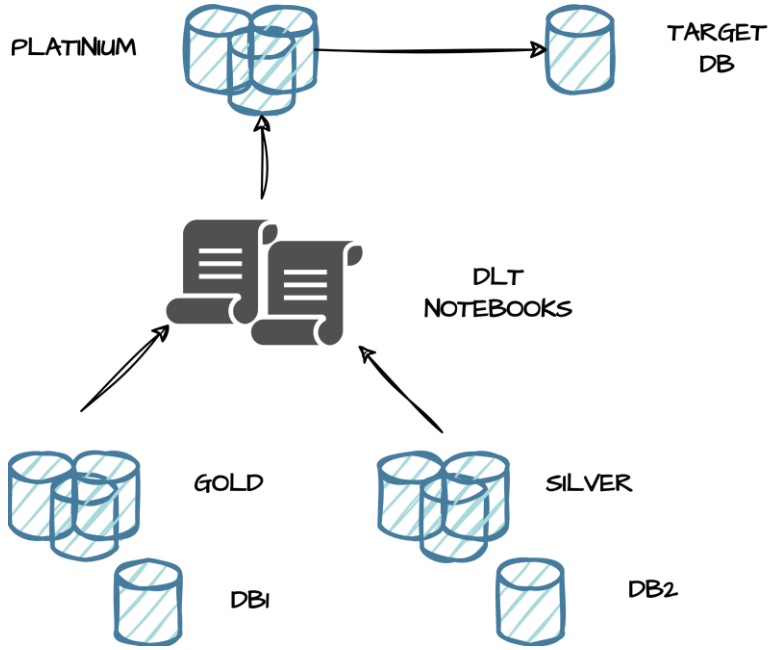
Processes:

- **Jobs /Workflows**
  - **Notebooks/Scripts/Queries…**
    - **Three level namespace**
- **Delta Live Tables**
  - **Notebooks**
  - **Three level namespace**

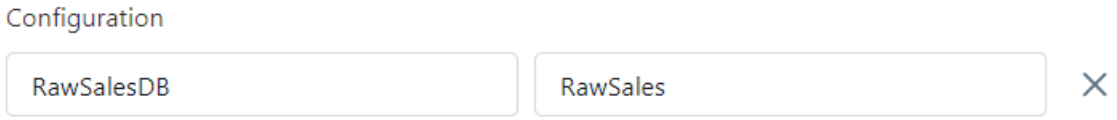
Spark configuration settings:

`spark.databricks.sql.initial.catalog.name`

(overrides the default catalog for a specific cluster)



Pipelines/Notebooks Parametrization



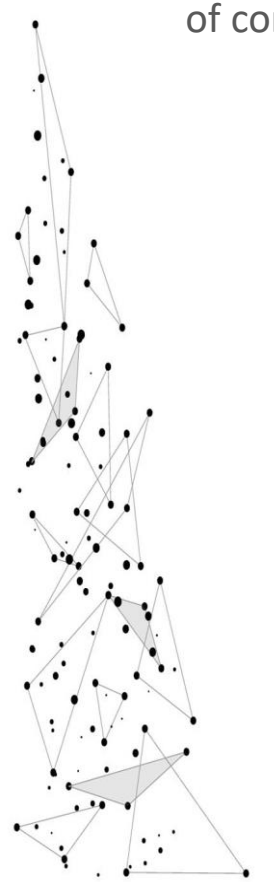
Processes Migration

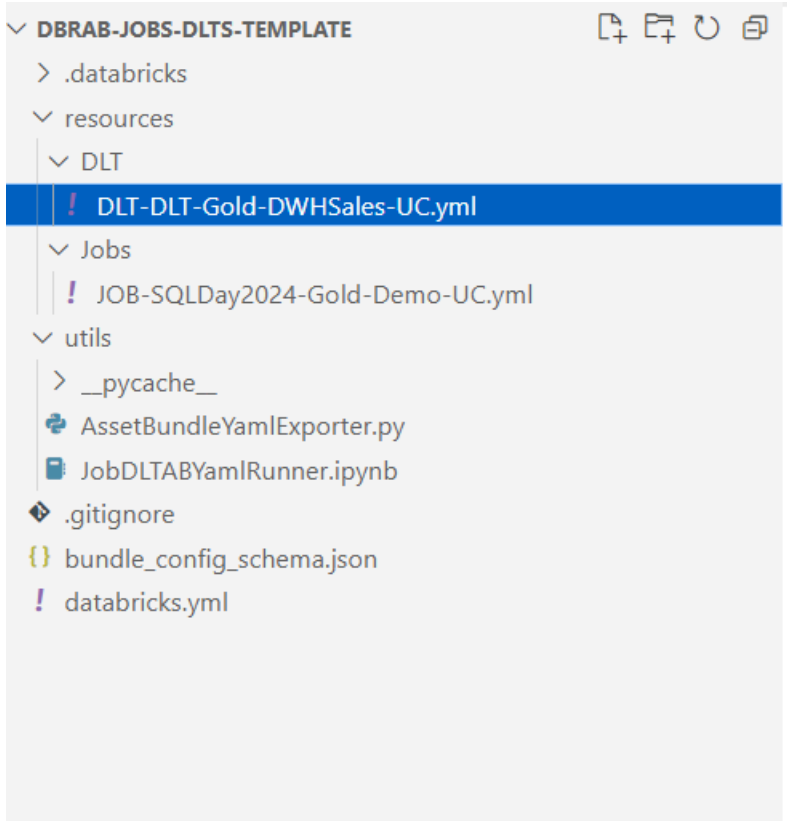
Databricks Asset Bundles

Databricks Asset Bundles (DABs) are a new tool for streamlining the development of complex data, analytics, and ML projects for the Databricks platform.

databricks

Databricks SDK for Python



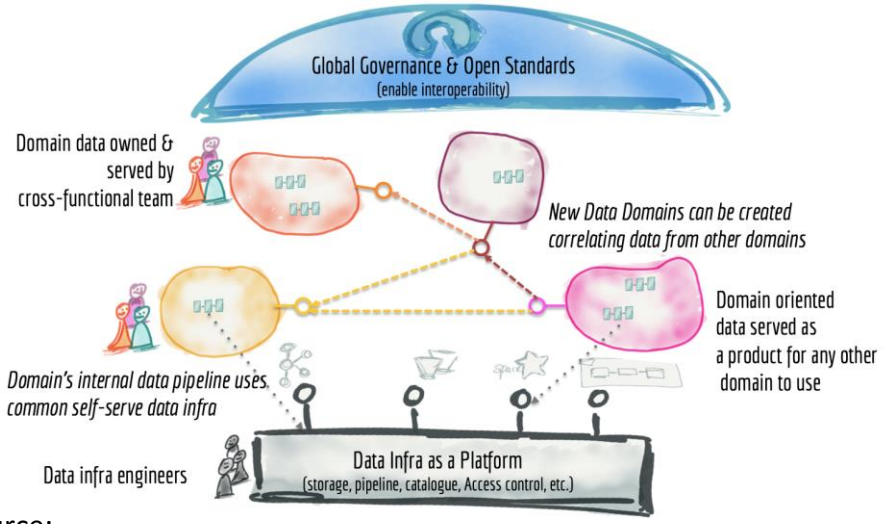


resources > DLT > DLT-DLT-Gold-DWHSales-UC.yml

```yaml
resources:
  pipelines:
    DLT-DLT-Gold-DWHSales-UC:
      catalog: ${bundle.target}_gold
      channel: PREVIEW
      clusters:
      - label: default
        num_workers: 1
      configuration:
        Param1: rawmarketing
        spark.databricks.sql.initial.catalog.name: ${bundle.target}_silver
      continuous: false
      development: true
      edition: CORE
      libraries:
      - notebook:
          path: /Users/tkrawczyk@future-processing.com/SQLDay2024/Code/DLT/Gold/DWH
      name: DLT-Gold-DWHSales-[UC]
      photon: false
      target: DWHSales
```

Unity Catalog -Benefits

- **Centralized data management**
  - Unity Catalog provides a single place to manage all your data assets, simplifying data discovery, access, and governance.
- **Improved collaboration**
  - With a unified catalog, teams can easily share and access data across the organization, promoting collaboration and reducing data silos.
- **Enhanced security**
  - Unity Catalog offers robust security features, such as column-level access control and data encryption, to ensure that your data is protected.
- **Better data quality**
  - By centralizing data management, Unity Catalog helps ensure that data is consistent, accurate, and up-to-date, improving data quality and reliability.
- **Increased productivity**
  - Unity Catalog simplifies data management tasks, such as data discovery, access, and integration, freeing up time for more value-added activities.



Source:

https://martinfowler.com/articles/data-monolith-to-mesh.html#TheParadigmShiftTowardsADataMesh

Centralized:

- Governance policies applied by a central team
- Production of data artifacts managed by a central team

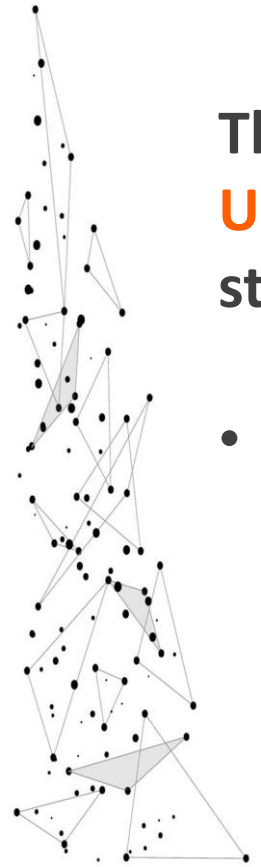
Distributed:

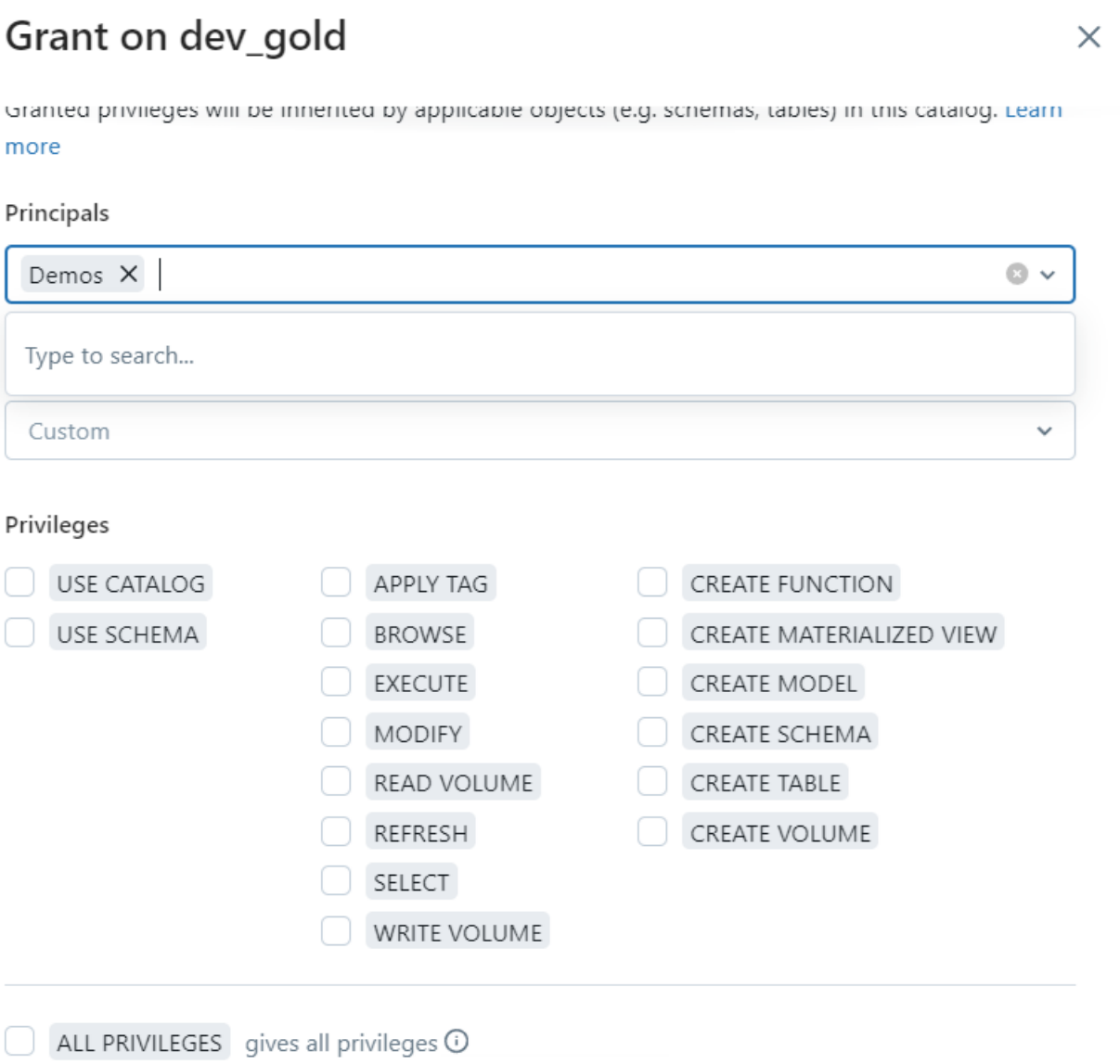
- Domain driven production of data artifacts
- Entitlements on data owned by domain teams

Unity Catalog – Permissions

The security model in Databricks Unity Catalog uses ANSI SQL standards.

- **GRANT/REVOKE** *privilege_types* ON *securable_object* TO *principal*





Unity Catalog -Security Row filters and column masking

- **IS_ACCOUNT_GROUP_MEMBER (** account level)
- ~~**IS_MEMBER** (~~workspace local group)
- **ROW FILTER FUNCTION**

```
CREATE FUNCTION us_filter(region STRING)
RETURN IF(IS_ACCOUNT_GROUP_MEMBER('admin'), true, region='US');
```

SQL

```
ALTER TABLE <table_name> SET ROW FILTER <function_name> ON (<column_name>, ...);
```

- **COLUMN MASKING FUNCTION**

```
CREATE FUNCTION ssn_mask(ssn STRING)
  RETURN IF(IS_ACCOUNT_GROUP_MEMBER('admin'), ssn, '****');
```

SQL Copy

```
ALTER TABLE <table_name> ALTER COLUMN <col_name> SET MASK <mask_func_name> [USING COLUMNS <additional_columns>];
```

Table

Name	Age	Country
John	34	US
Eva	33	UK
Jenny	32	US

All data

USUsers Group
ViewSensitveData Group

Table

Name	Age	Country
John	34	US
Jenny	32	US

Not in ViewSensitveData Group

Table

Name	Age	Country
****	34	US
****	32	US

UKUsers Group
ViewSensitveData Group

Table

Name	Age	Country
Eva	33	UK

Not in ViewSensitveData Group

Table

Name	Age	Country
****	33	UK

Limitations:

Delta Live Tables materialized views and streaming tables don’t support row filters or column masks.

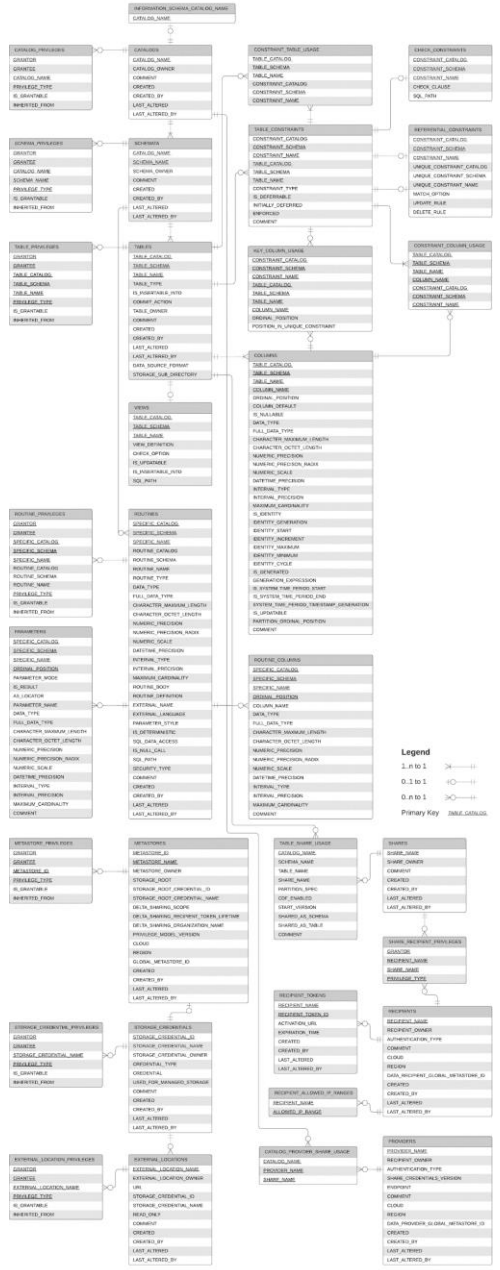
Unity Catalog –Auditing

The *INFORMATION_SCHEMA* is a SQL standard based schema, provided in every catalog created on **Unity Catalog**.

```
SELECT * FROM information_schema.catalogs;

SELECT * FROM information_schema.catalog_privileges;

SELECT * FROM information_schema.tables;
```

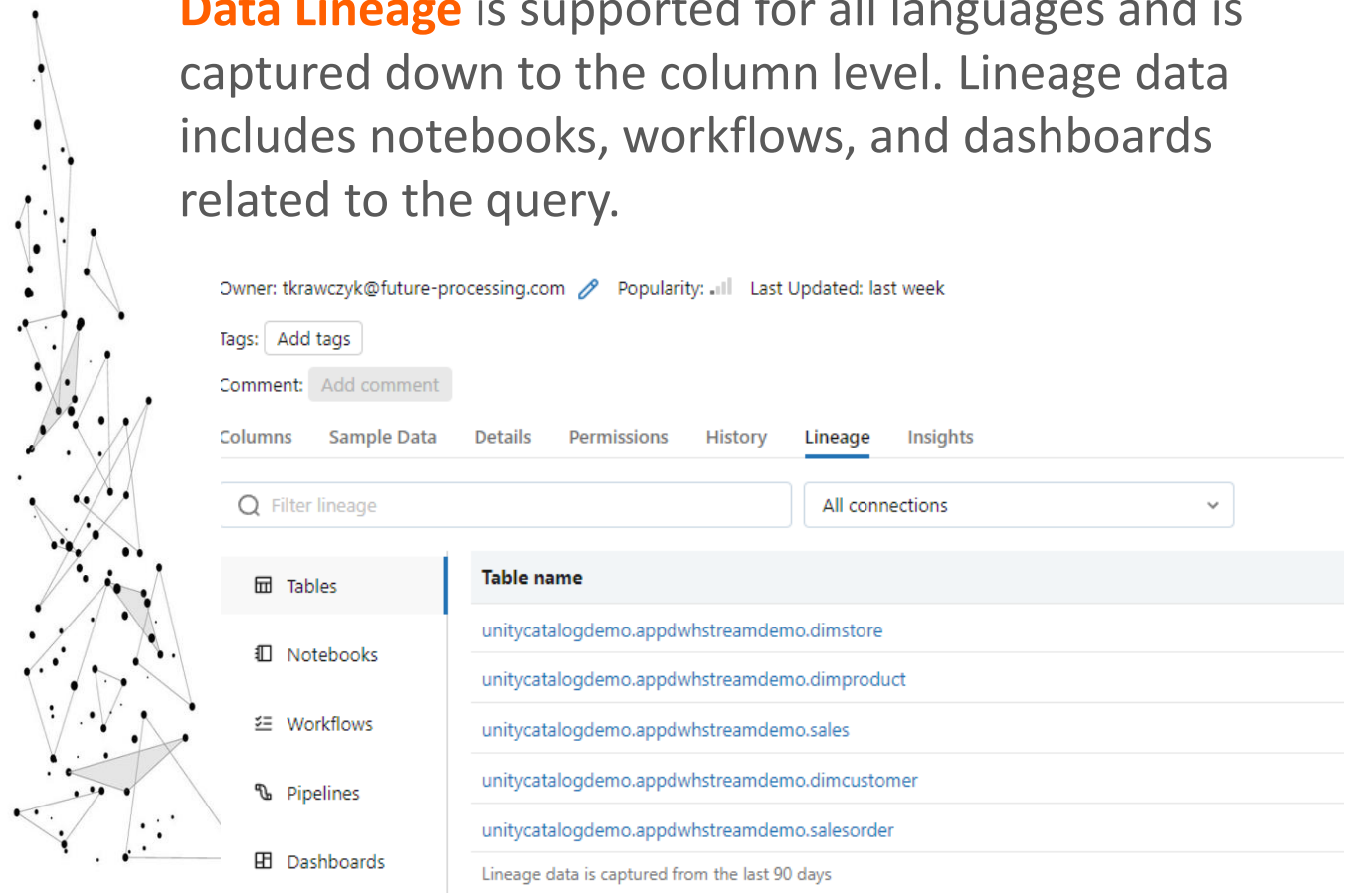


System tables are an Databricks-hosted analytical store of your account's operational data

- **Audit logs**: Located at system.access.audit.
- **Billable usage logs**: Located at system.billing.usage.
- **Pricing table**: Located at system.billing.list_prices.
- **Table and column lineage**: Both tables located under system.access.
- **Marketplace listing access**: Located at system.marketplace.listing_access_events.

Unity Catalog Data Lineage

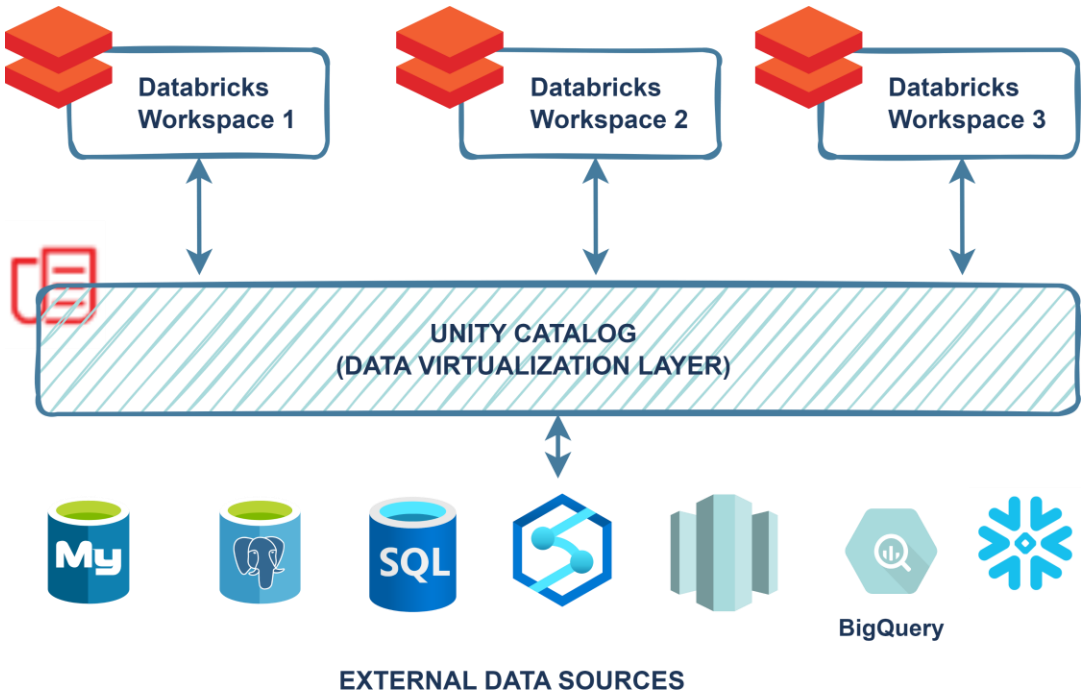
Data Lineage is supported for all languages and is captured down to the column level. Lineage data includes notebooks, workflows, and dashboards related to the query.



Unity Catalog – Lakehouse Query Federation

Lakehouse Query Federation provides one single secure access to all your data.
Supported data sources:

- **MySQL ,PostgreSQL,Amazon Redshift, Snowflake, Azure SQL Database, Azure Synapse, Google's BigQuery …**
- **MS Fabric - Microsoft Fabric Community Conference**



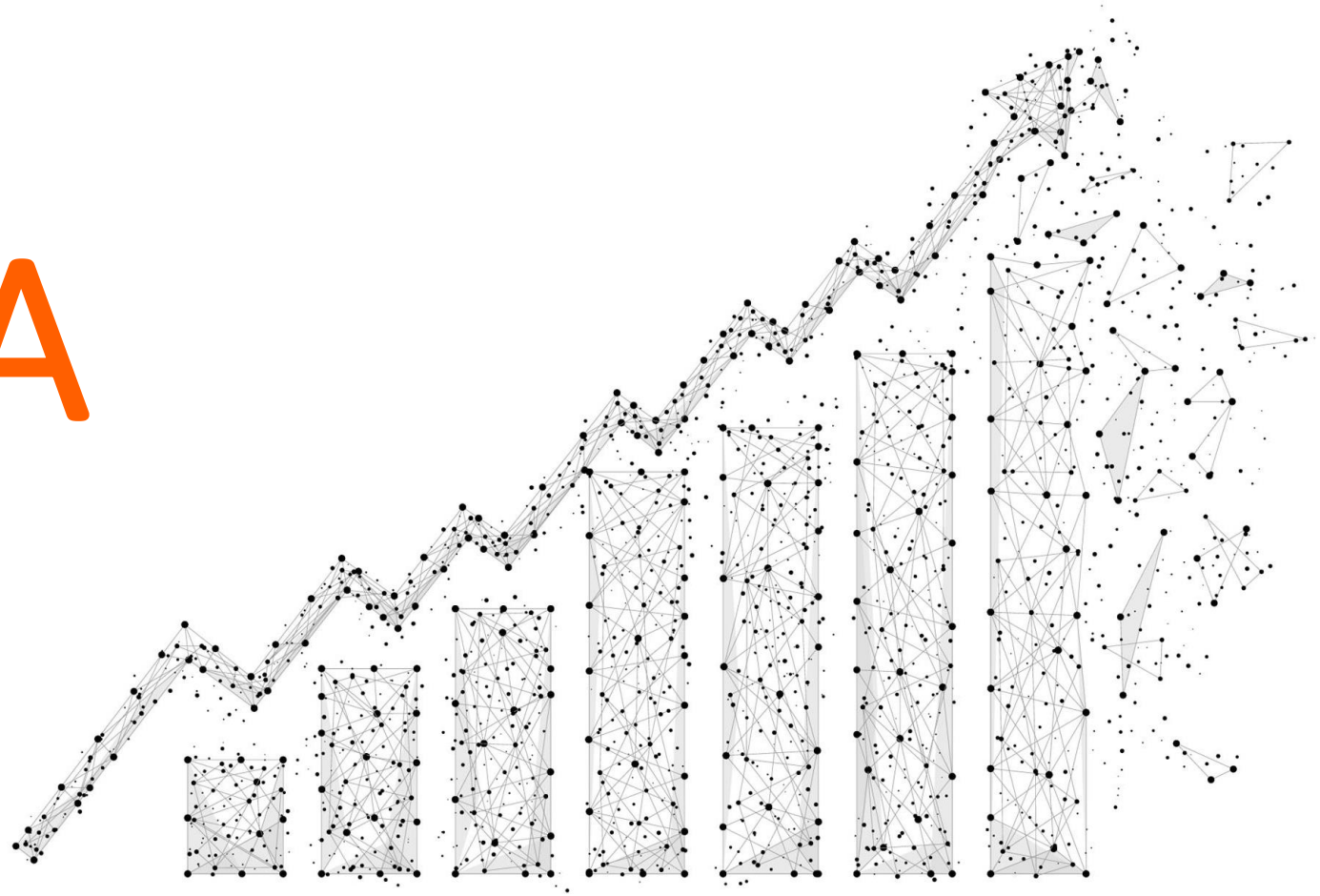
Unity Catalog provides:

- **Unified permission controls**
- **Intelligent pushdown optimizations**
- Accelerated query performance with Materialized view
- Support for R/O operations

```
CREATE FOREIGN CATALOG [IF NOT EXISTS] <catalog-name> USING CONNECTION <connection-name>
OPTIONS (database '<database-name>');
```

DEMO TIME

Q & A

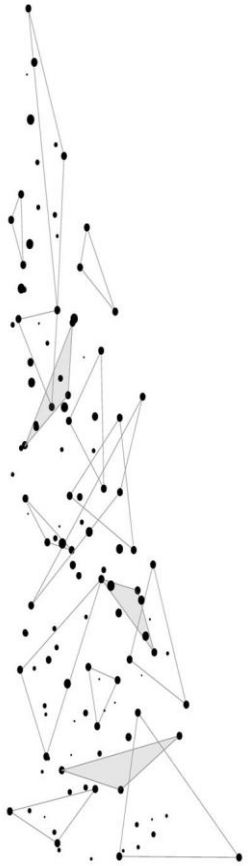


Resources

- https://www.databricks.com/product/unity-catalog
- https://www.databricks.com/resources/demos/tutorials?itm_data=demo_center
- https://learn.microsoft.com/en-us/azure/databricks/sql/language-manual/sql-ref-information-schema
- https://learn.microsoft.com/en-us/azure/databricks/administration-guide/system-tables/
- https://www.databricks.com/resources/demos/tutorials?itm_data=demo_center

THANK YOU!

tkrawczyk@future-processing.com
tomasz.k.krawczyk@gmail.com



Data
Community