Azure RBAC & Databricks Unity Catalog

Table of Contents

[What is Azure role-based access control (Azure RBAC)? 2](#_heading=h.gjdgxs)

[What can I do with Azure RBAC? 2](#_heading=h.30j0zll)

[How Azure RBAC works 2](#_heading=h.1fob9te)

[Databricks 2](#_heading=h.3znysh7)

[Account Admins Vs Workspace Admins 3](#_heading=h.2et92p0)

[Account Admins Vs Metastore Admins Vs Workspace Admins 4](#_heading=h.tyjcwt)

[Data governance and secure data sharing 5](#_heading=h.3dy6vkm)

[What is a data lakehouse? 5](#_heading=h.1t3h5sf)

[Unity Catalog Introduction: 5](#_heading=h.4d34og8)

[Overview of Unity Catalog 6](#_heading=h.2s8eyo1)

[Data hierarchy in Unity Catalog: 7](#_heading=h.17dp8vu)

[Metastore: 7](#_heading=h.3rdcrjn)

[Catalog: 7](#_heading=h.26in1rg)

[Schema: 7](#_heading=h.lnxbz9)

[Tables: 7](#_heading=h.35nkun2)

[Managed Tables 8](#_heading=h.1ksv4uv)

[External Tables 8](#_heading=h.44sinio)

[Views: 8](#_heading=h.2jxsxqh)

[Comparison of between without & with UNITY CATALOG: 8](#_heading=h.z337ya)

[Automatic enablement of Unity Catalog 10](#_heading=h.3j2qqm3)

[Steps to Create Metastore: 10](#_heading=h.1y810tw)

[Prerequisites for creating Metastore 10](#_heading=h.4i7ojhp)

[How to Create Unity Catalog 14](#_heading=h.2xcytpi)

[a) Create storage credentials: 14](#_heading=h.1ci93xb)

[b) Create External Locations: 15](#_heading=h.3whwml4)

[c)Create Unity Catalog 16](#_heading=h.2bn6wsx)

[Resource quotas 17](#_heading=h.qsh70q)

### What is Azure role-based access control (Azure RBAC)?

Access management for cloud resources is a critical function for any organization that is using the cloud. Azure role-based access control (Azure RBAC) helps you manage who has access to Azure resources, what they can do with those resources, and what areas they have access to.

Azure RBAC is an authorization system built on [Azure Resource Manager](https://learn.microsoft.com/en-us/azure/azure-resource-manager/management/overview) that provides fine-grained access management to Azure resources.

### What can I do with Azure RBAC?

Here are some examples of what you can do with Azure RBAC:

* Allow one user to manage virtual machines in a subscription and another user to manage virtual networks
* Allow a DBA group to manage SQL databases in a subscription
* Allow a user to manage all resources in a resource group, such as virtual machines, websites, and subnets
* Allow an application to access all resources in a resource group

### How Azure RBAC works

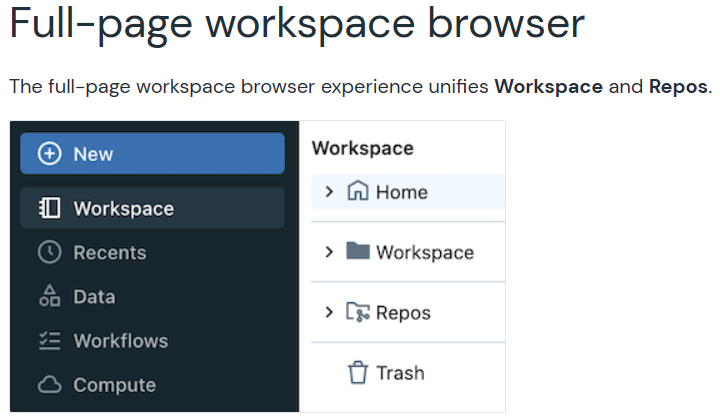
The way you control access to resources using Azure RBAC is to assign Azure roles. A role assignment consists of three elements: security principal, role definition, and scope.

Azure includes several [built-in roles](https://learn.microsoft.com/en-us/azure/role-based-access-control/built-in-roles) that you can use. For example, the [Virtual Machine Contributor](https://learn.microsoft.com/en-us/azure/role-based-access-control/built-in-roles#virtual-machine-contributor) role allows a user to create and manage virtual machines. If the built-in roles don't meet the specific needs of your organization, you can create your own [Azure custom roles](https://learn.microsoft.com/en-us/azure/role-based-access-control/custom-roles).

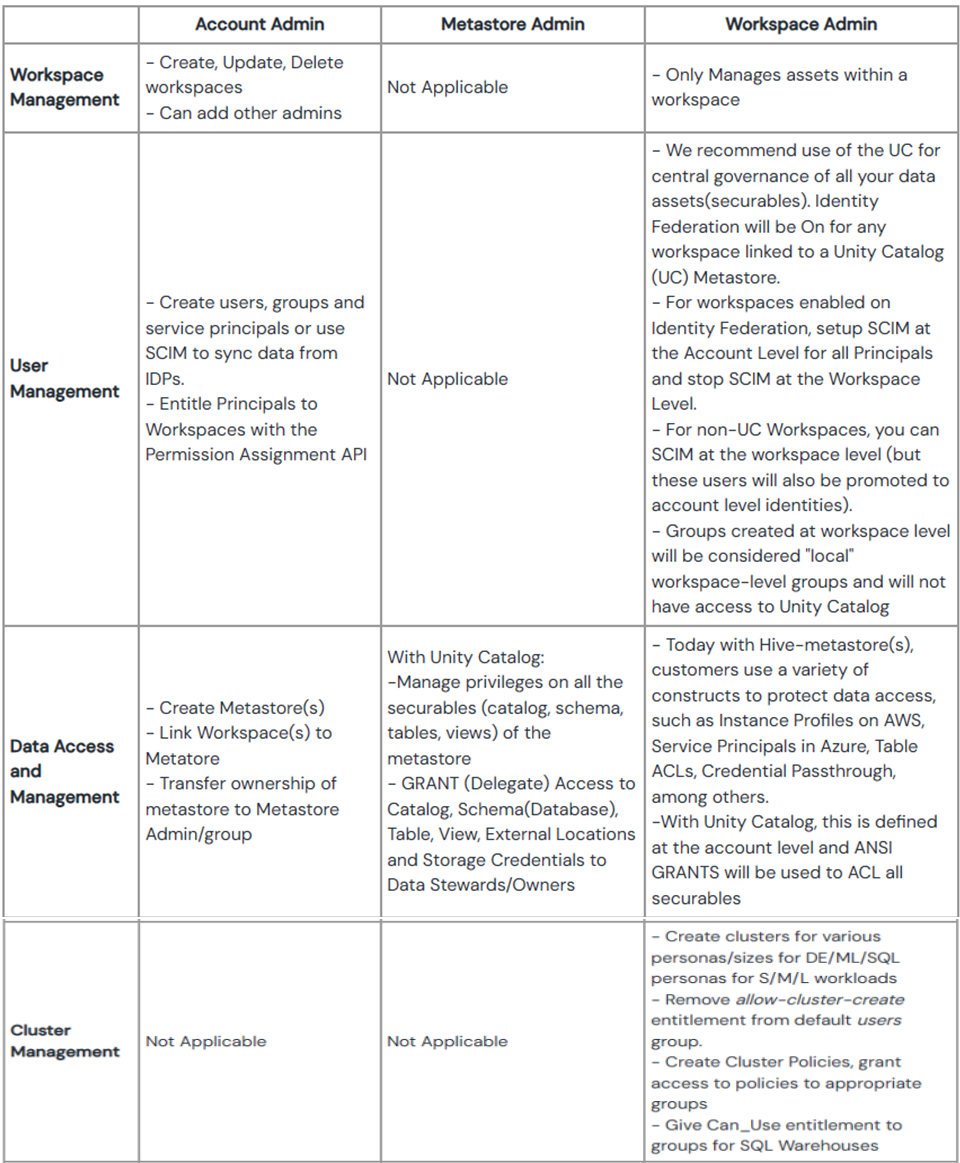
For more details please visit: [What is Azure role-based access control (Azure RBAC)? | Microsoft Learn](https://learn.microsoft.com/en-us/azure/role-based-access-control/overview)

### Databricks

Databricks is a unified, open analytics platform for building, deploying, sharing, and maintaining enterprise-grade data, analytics. The Databricks Data Intelligence Platform integrates with cloud storage and security in your cloud account, and manages and deploys cloud infrastructure on your behalf.



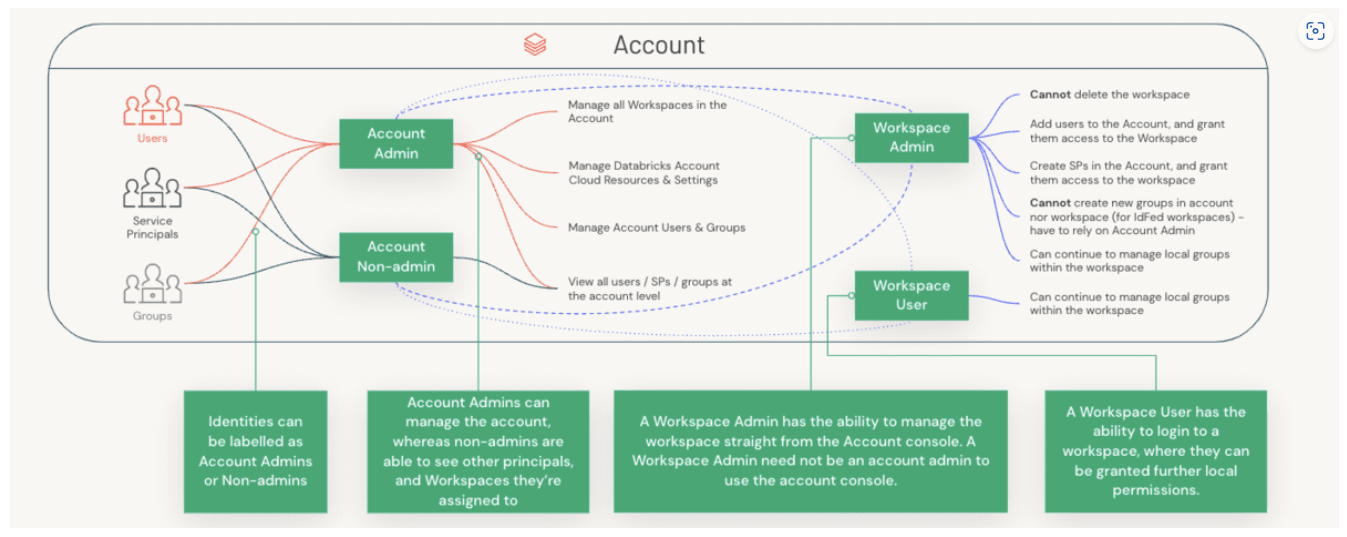
### Account Admins Vs Metastore Admins Vs Workspace Admins



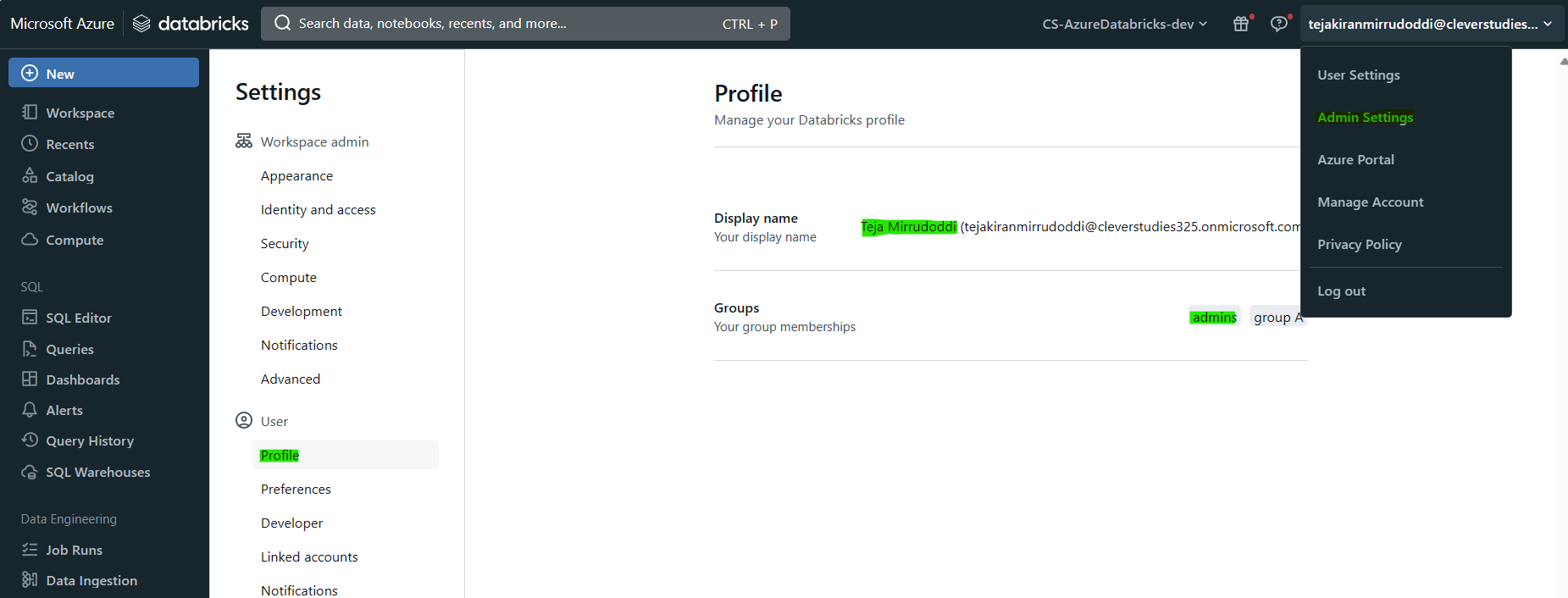
### 

### 

### Account Admins Vs Workspace Admins



**Where to check role of user in databricks**



### Data governance and secure data sharing

Unity Catalog provides a unified data governance model for the data lakehouse. Azure Databricks administrators can manage permissions for teams and individuals.

The lakehouse makes data sharing within your organization as simple as granting query access to a table or view. For sharing outside of your secure environment, Unity Catalog features a managed version of [Delta Sharing](https://learn.microsoft.com/en-us/azure/databricks/data-sharing/).

### What is a data lakehouse?

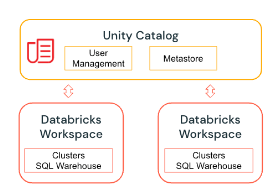
A data lakehouse is a data management system that combines the benefits of data lakes and data warehouses. [Data Lakehouse](https://learn.microsoft.com/en-us/azure/databricks/lakehouse/)

### 

### Unity Catalog Introduction:

Unity Catalog is a fine-grained governance solution for data on the Databricks platform. It helps simplify security and governance of your data by providing a central place to administer and audit data access.

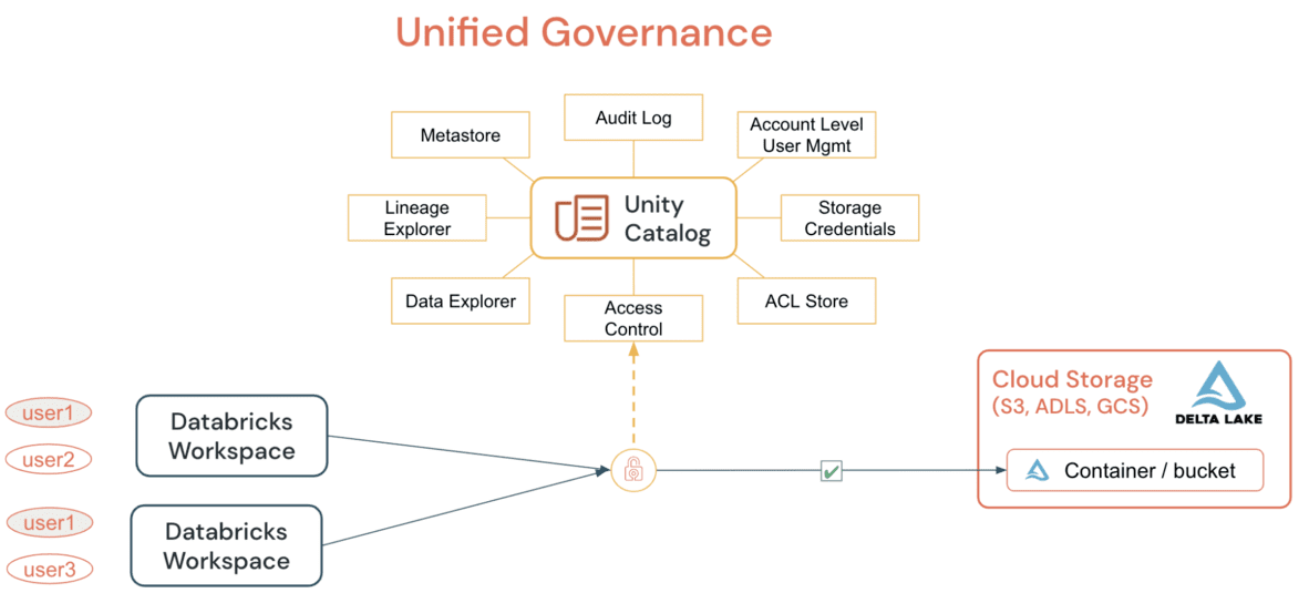
Unity Catalog provides centralized access control, auditing, lineage, and data discovery capabilities across Azure Databricks workspaces.



### Overview of Unity Catalog

Data Governance framework:

Unity Catalog (UC) enables organizations to adopt a common security model for tables and files for all workspaces under a single account, which was not possible before through simple GRANT statements.

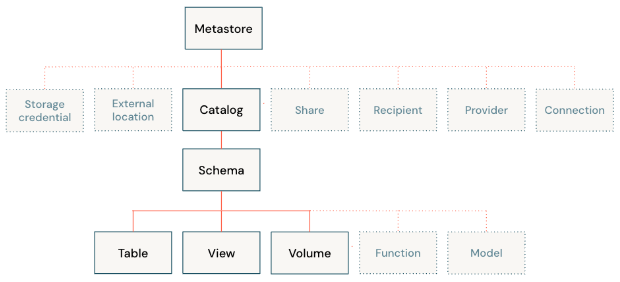


UC simplifies the job of an administrator (both at the account and workspace level) by centralizing the definitions, monitoring and discoverability of data across the metastore, and making it easy to securely share data irrespective of the number of workspaces that are attached to it. Utilizing the [Define Once, Secure Everywhere](https://docs.databricks.com/data-governance/unity-catalog/index.html#unity-catalog-preview) model, this has the added advantage of avoiding accidental data exposure in the scenario of a user's privileges inadvertently misrepresented in one workspace which may give them a backdoor to get to data that was not intended for their consumption.

All of this can be accomplished easily by utilizing [Account Level Identities](https://docs.databricks.com/data-governance/unity-catalog/manage-identities.html) and [Data Permissions](https://docs.databricks.com/data-governance/unity-catalog/data-permissions.html#data-permissions). [UC Audit Logging](https://docs.databricks.com/data-governance/unity-catalog/audit.html#unity-catalog-audit-log-events) allows full visibility into all actions by all users at all levels on all objects, and if you configure [verbose audit logging](https://docs.databricks.com/administration-guide/account-settings/audit-logs.html#configure-verbose-audit-logs), then each command executed, from a notebook or Databricks SQL, is captured. Access to securable can be granted by either a metastore admin, the owner of an object, or the owner of the catalog or schema that contains the object. It is recommended that the account-level admin delegate the metastore role by nominating a group to be the **metastore admins** whose sole purpose is granting the right access privileges.

For more details pls go through [Account Admins Vs Workspace Admin](https://www.databricks.com/blog/2022/08/26/databricks-workspace-administration-best-practices-for-account-workspace-and-metastore-admins.html)

### Data hierarchy in Unity Catalog:



### Metastore:

A metastore is the top-level container of objects in Unity Catalog. It registers metadata about data assets and the permissions that govern access to them. A metastore can optionally be configured with a managed storage location in an Azure Data Lake Storage Gen2 container in your Azure account

### Catalog:

A catalog is the first layer of Unity Catalog’s three-level namespace. It’s used to organize your data assets. Users can see all catalogs on which they have been assigned the USE CATALOG

### Schema:

A schema (also called a database) is the second layer of Unity Catalog’s three-level namespace. A schema organizes tables and views. Users can see all schemas on which they have been assigned the USE SCHEMA permission, along with the USE CATALOG permission on the schema’s parent catalog. To access or list a table or view in a schema, users must also have SELECT permission on the table or view.

### Tables:

A table resides in the third layer of Unity Catalog’s three-level namespace. It contains rows of data.

There are two types of Tables

### Managed Tables

Managed tables are the default way to create tables in Unity Catalog. By default, managed tables are stored in the root storage location that you configure when you create a metastore. You can optionally specify managed table storage locations at the catalog or schema levels, overriding the root storage location. Managed tables always use the Delta table format.

### External Tables

External tables are tables whose data lifecycle and file layout are not managed by Unity Catalog. Use external tables to register large amounts of existing data in Unity Catalog, or if you require direct access to the data using tools outside of Azure Databricks clusters or Databricks SQL warehouses.

### Views:

A view is a read-only object created from one or more tables and views in a metastore. It resides in the third layer of Unity Catalog’s three-level namespace. A view can be created from tables and other views in multiple schemas and catalogs

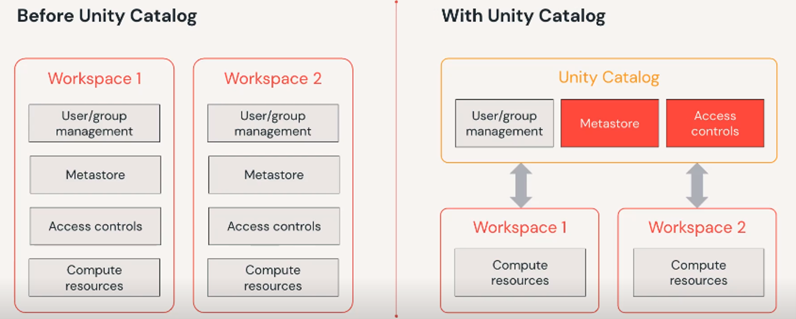
\*\*Official Document link: <https://learn.microsoft.com/en-us/azure/databricks/data-governance/unity-catalog/>

\*\*Scenario based Document links: <https://www.linkedin.com/pulse/comprehensive-guide-azure-databricks-unity-catalog-aritra-ghosh>

### Comparison between without & with UNITY CATALOG:

| Without Unity Catalog: | With Unity Catalog: |
| --- | --- |
| In a Databricks workspace without the Unity Catalog, each workspace functions independently, resembling separate rooms. User management, access control, and the Metastore, which stores metadata about data objects, are defined within the confines of each workspace. Essentially, what occurs in one workspace has no direct impact on the others.  Data resources, such as databases and tables, created in one workspace are restricted to that specific environment. These data objects remain invisible to other workspaces, leading to isolated sets of data resources within each workspace.  Security measures, including access control, are constrained by the boundaries of individual workspaces. If permissions are granted for accessing a database in Workspace A, these permissions do not automatically extend to Workspace B. Each workspace manages its security independently, limiting the scope of security measures to the confines of each distinct workspace. | Unity Catalog introduces centralized governance, serving as a master control center for user management, access control, and metadata storage. All users, databases, tables, and access permissions are concentrated within the Unity Catalog, providing a unified oversight mechanism.  Connected workspaces benefit from visibility across data objects. If a table is created in Workspace A, it becomes seamlessly accessible in Workspace B, fostering collaboration and enabling efficient resource sharing.  Unified access control ensures consistency across all workspaces, as permissions are centrally managed within the Unity Catalog. When access is granted to a specific database or table, that permission universally applies, establishing a standardized and cohesive security model.  Collaboration is enhanced as the Unity Catalog breaks down silos between workspaces. Teams can effortlessly share and collaborate on data resources without being confined by the limitations of individual workspace boundaries.  Efficient metadata management is achieved through the Unity Catalog’s Metastore, where metadata about data objects is stored centrally. This approach establishes a single source of truth for metadata, minimizing inconsistencies and simplifying the management and comprehension of the organization’s data landscape. |

In summary, using Databricks with the Unity Catalog provides a more centralized and streamlined approach to managing users, data objects, and access control. It promotes collaboration, visibility across workspaces, and a consistent security model, enhancing the overall efficiency and governance of data within the organization.



### Automatic enablement of Unity Catalog

Workspaces that were enabled automatically have the following properties:

* An automatically-provisioned Unity Catalog metastore (unless a Unity Catalog metastore already existed for the workspace region).
* Default privileges for workspace admins, such as the ability to create a catalog or an external database connection.
* No metastore admin (unless an existing Unity Catalog metastore was used and a metastore admin was already assigned).
* No metastore-level storage for managed tables and managed volumes (unless an existing Unity Catalog metastore with metastore-level storage was used).
* A *workspace catalog*, which, when originally provisioned, is named after your workspace.

All users in your workspace can create assets in the default schema in this catalog. By default, this catalog is *bound* to your workspace, which means that it can only be accessed through your workspace. Automatic provisioning of the workspace catalog at workspace creation is rolling out gradually across accounts.

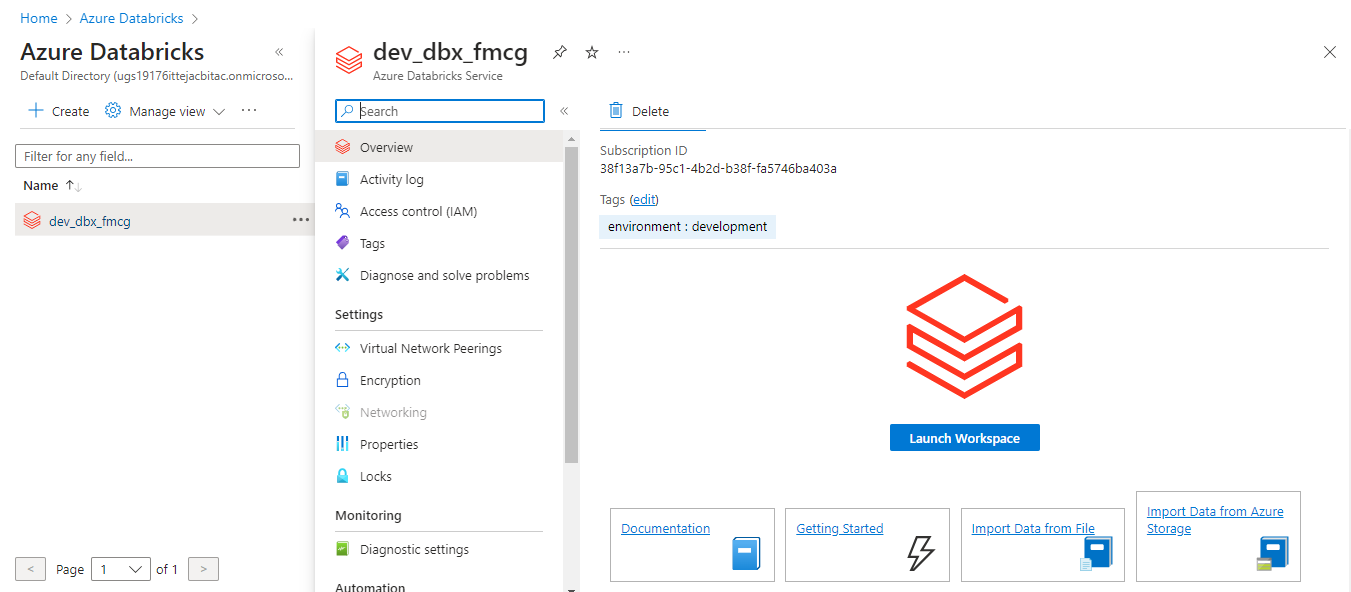
[Set up and manage Unity Catalog - Azure Databricks | Microsoft Learn](https://learn.microsoft.com/en-us/azure/databricks/data-governance/unity-catalog/get-started)

### Steps to Create Metastore:

### Prerequisites for creating Metastore

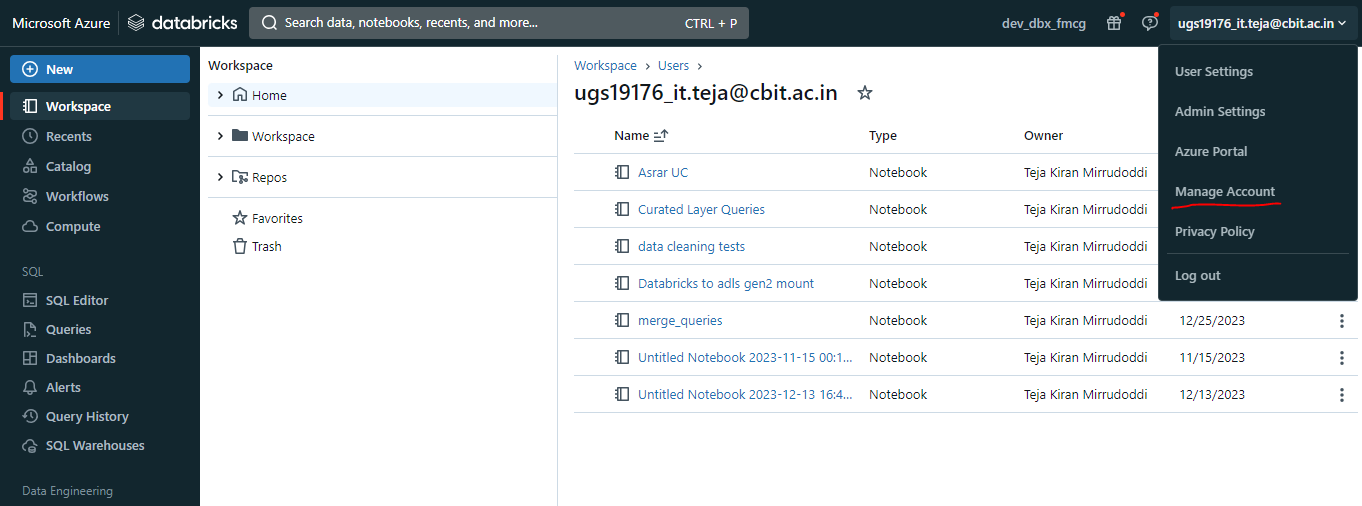
* Databricks account admin can only create a metastore and assign a metastore admin. In AAD, Global administrators are the default databricks account admins that can delegate it to other groups/users.
* A cluster to access Unity Catalog should set its access mode to either a single user or shared access mode and Databricks runtime version to Runtime: 11.1 (Scala 2.12, Spark 3.2.1) or higher.

-**Create a Databricks workspace & launch workspace**

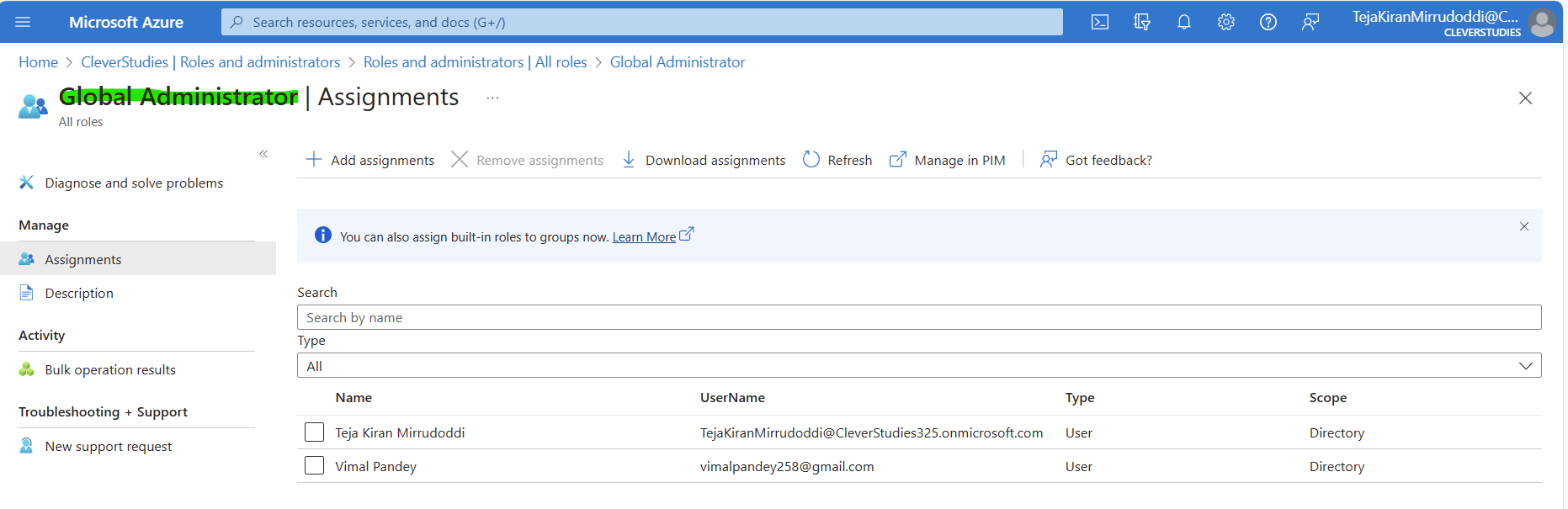


**-Navigate to MANAGE ACCOUNT on the top right inside the workspace.**

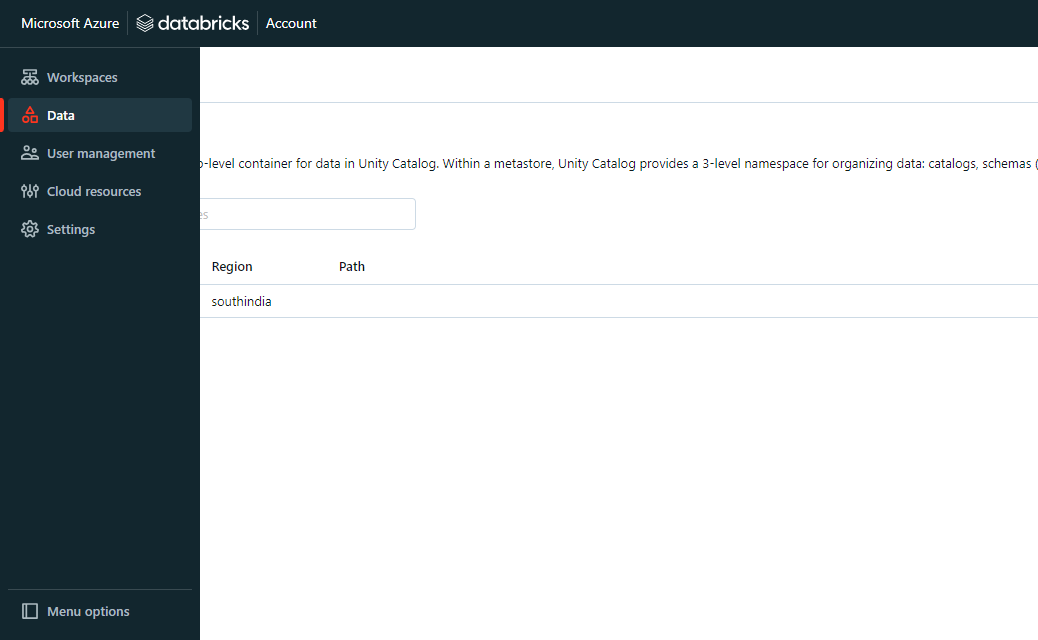
Navigate to MANAGE ACCOUNT on the top right inside the workspace.



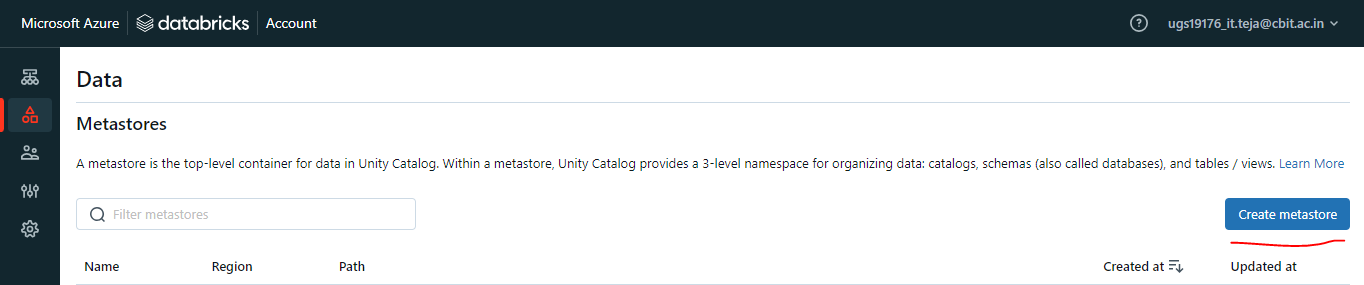
*Note: you need global Administrator access in your Azure account in order to get access to Manage account portal.*

**

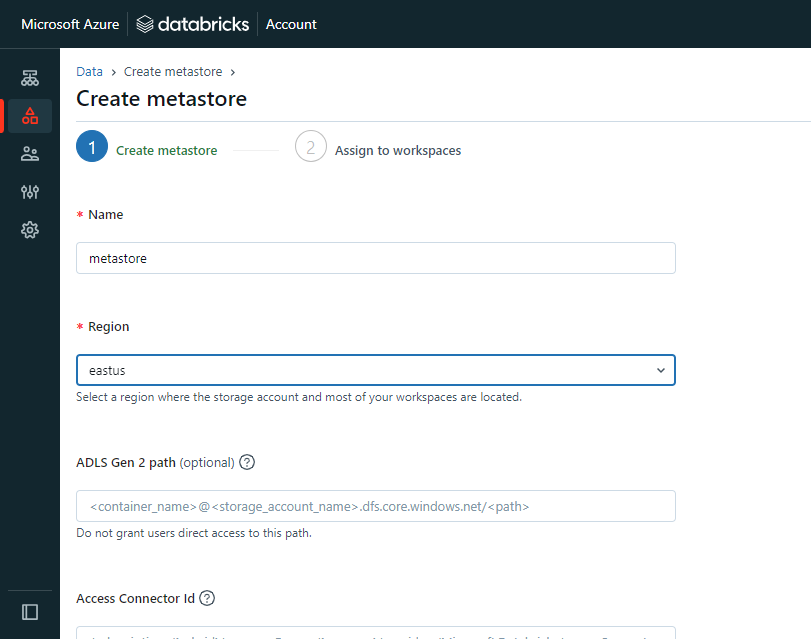
**-Go to DATA tab**



**-Select CREATE METASTORE option**



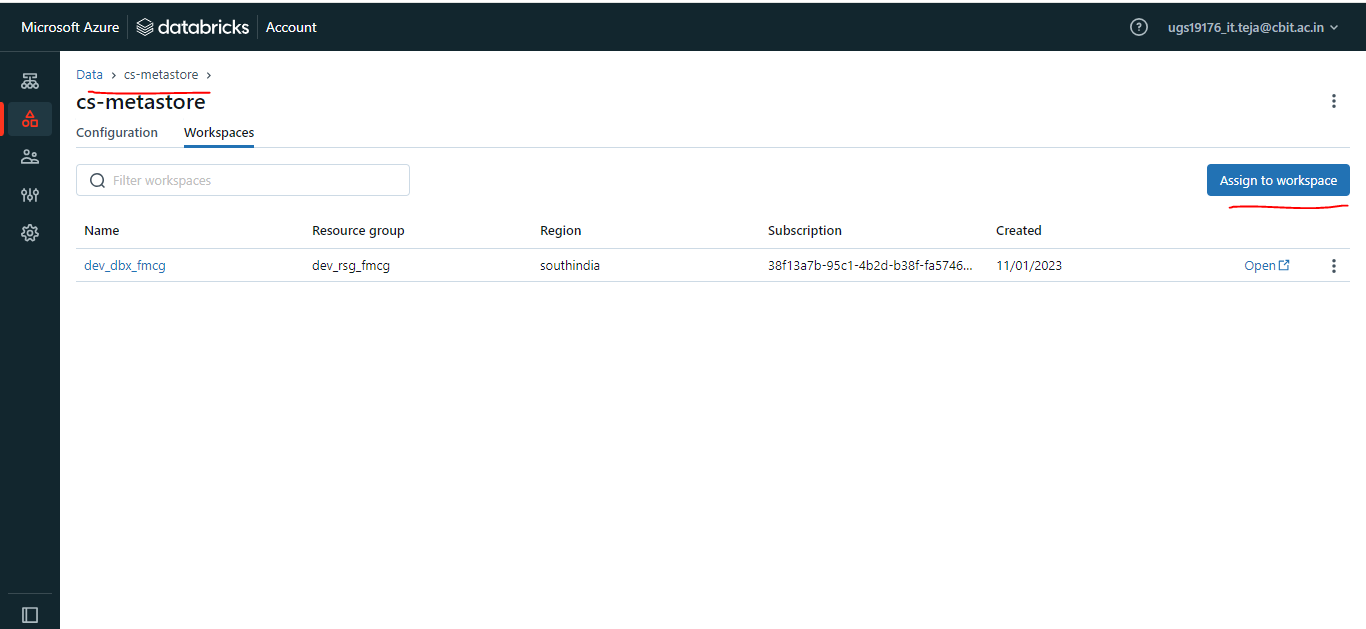
-Next fill out details of metastore ,like NAME & REGION(Only one metastore per region)



Adlsgen2 Path & access connector ID is optional

Select CREATE option below

**-Navigate to DATA in the left pane, click on the metastore & attach your databricks workspace to it.**



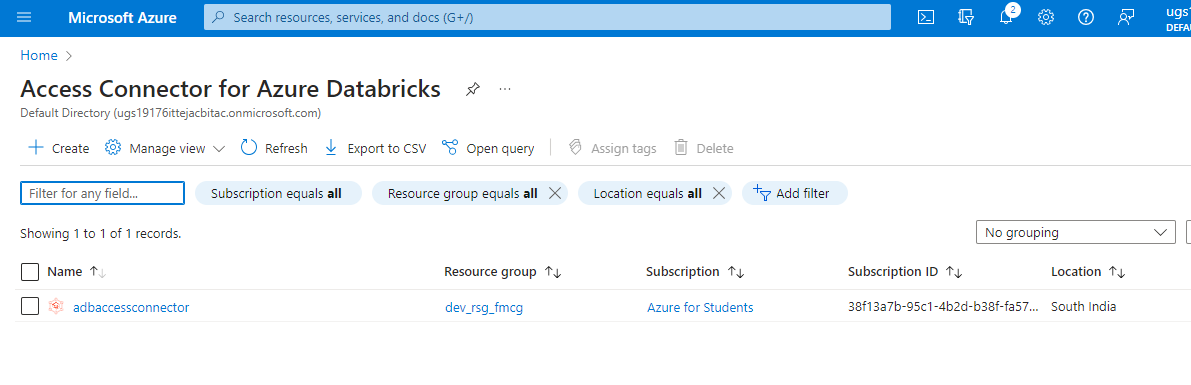
### How to Create Unity Catalog

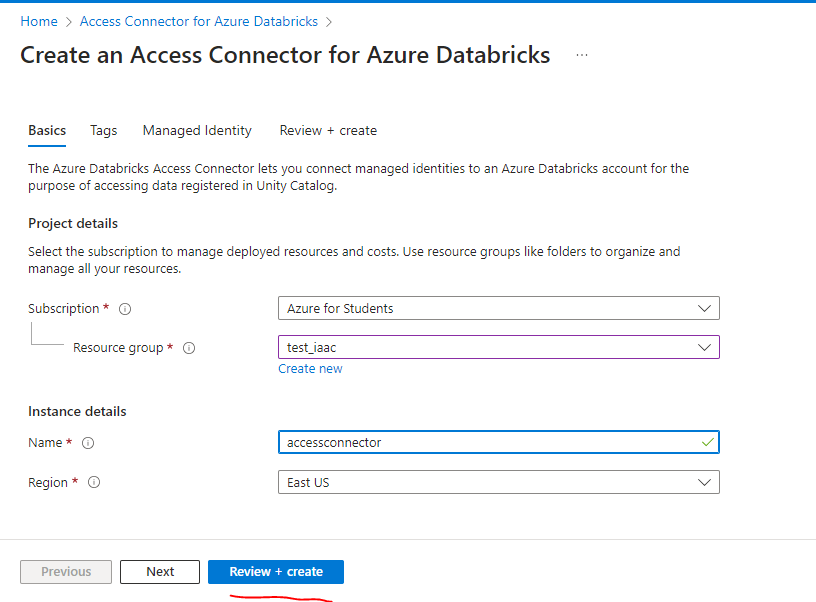
UNITY CATALOG:

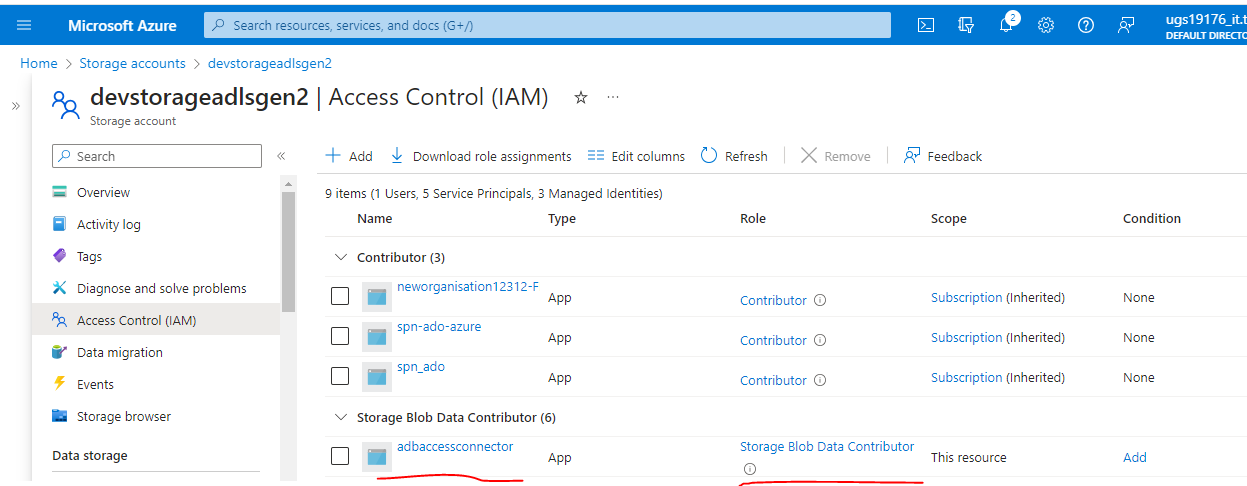
-login back to databricks workspace

### a) Create storage credentials:

-CREATE a "databricks access connector" resource in Azure & give that storage blob data contributor role.

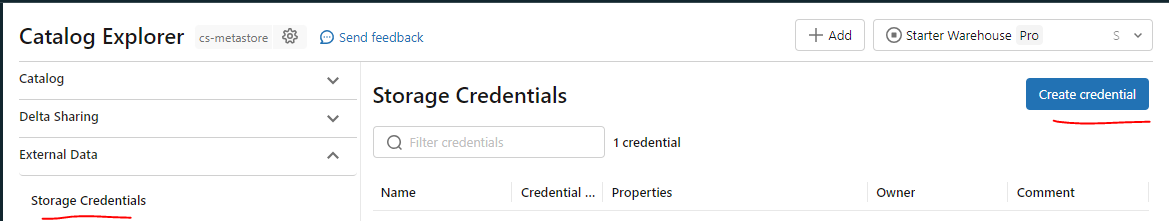




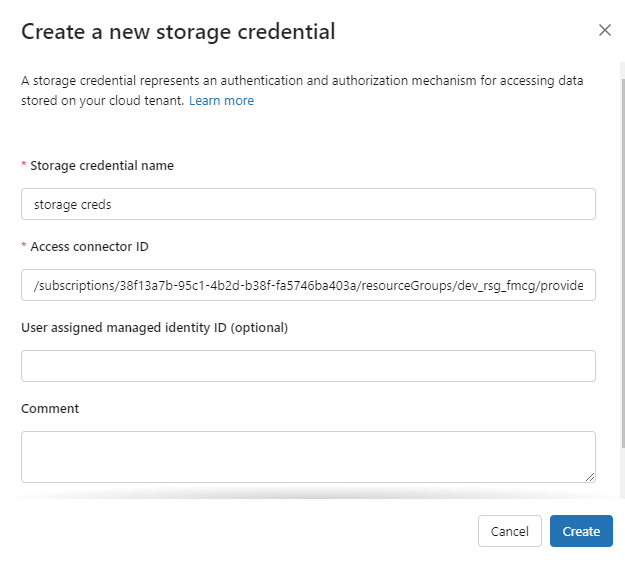


-Inside Databricks workspace CATALOG>EXTERNAL DATA(below)>STORAGE CREDENTIALS

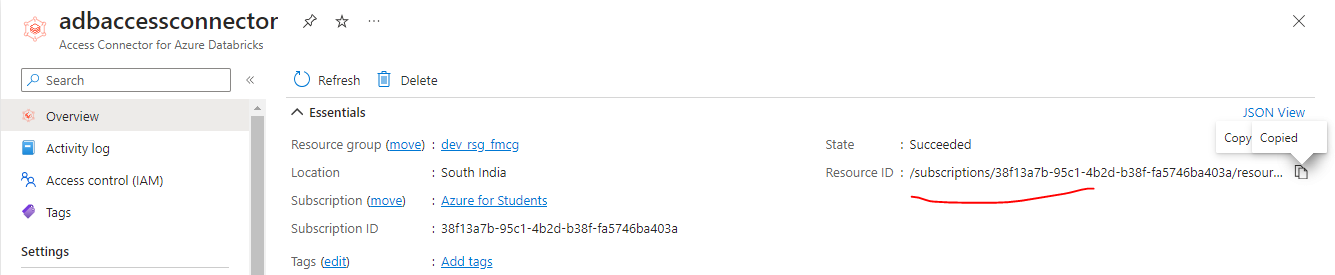
-Select CREATE CREDENTIAL



-Fill out details like STORAGE CREDENTIALS NAME & ACCESS CONNECTOR resource ID.

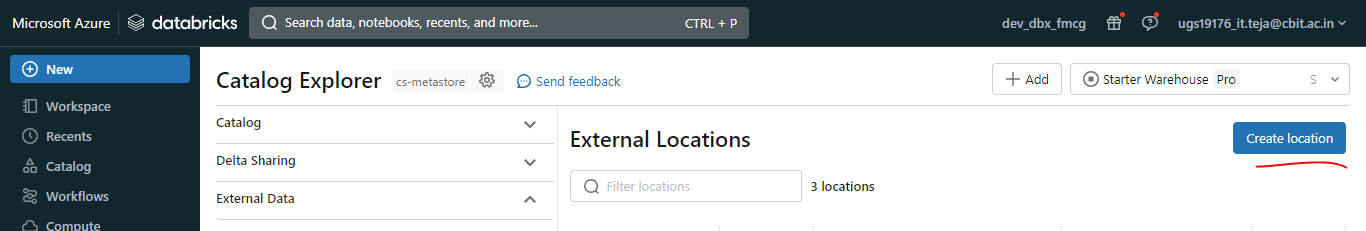


-Access connector ID you will find in databricks access connector resource

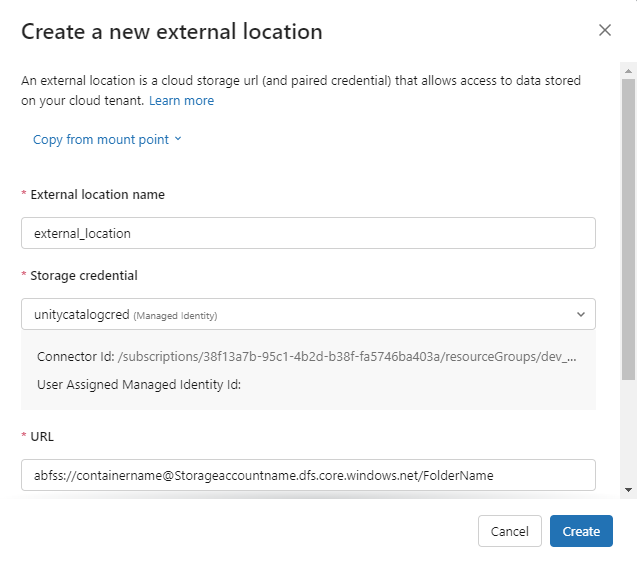


### b) Create External Locations:

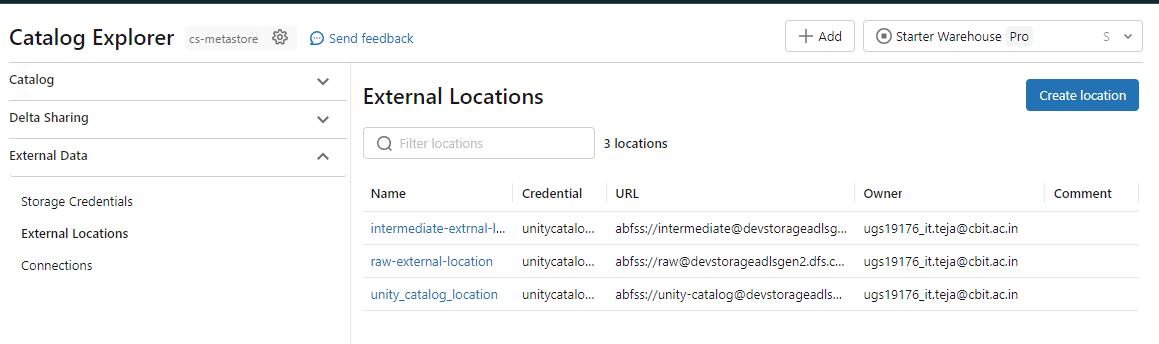
-Select External Locations option



-Fill out details like EXTERNAL LOCATION NAME, STORAGE CREDENTIAL USED & ADLS path URL in this format "abfss://containername@Storageaccountname.dfs.core.windows.net/FolderName"



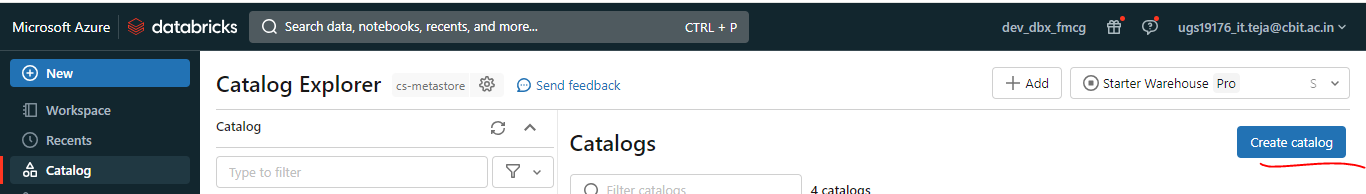
-you need to create external location for each schema example each external location for RAW, INTERMEDIATE, CURATED ,UNITY CATALOG(for external schema)



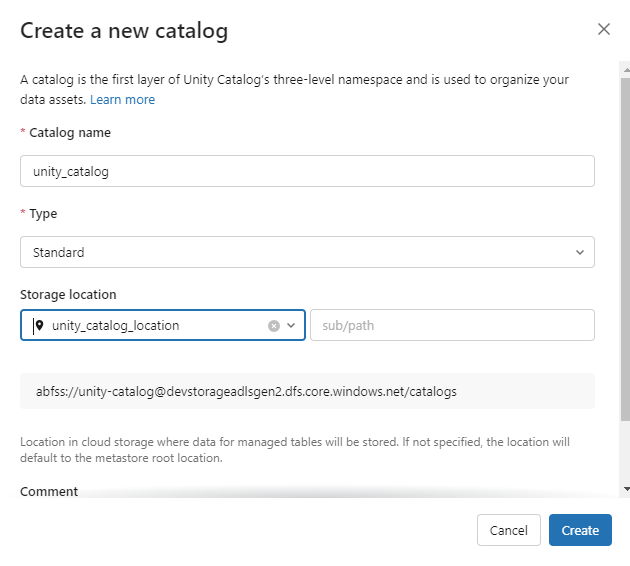
### c)Create Unity Catalog

-Navigate to CATALOG option from left pane

-Select CREATE CATALOG option



-Fill out the details like name,Type=Standard, & External Location(in adls where you want to store your external tables)



For more details, pls visit: [What is Unity Catalog? - Azure Databricks | Microsoft Learn](https://learn.microsoft.com/en-us/azure/databricks/data-governance/unity-catalog/)

### Resource quotas

Unity Catalog enforces resource quotas on all securable objects. Limits respect the same hierarchical organization throughout Unity Catalog. If you expect to exceed these resource limits, contact your Azure Databricks account team.

Quota values below are expressed relative to the parent (or grandparent) object in Unity Catalog.

