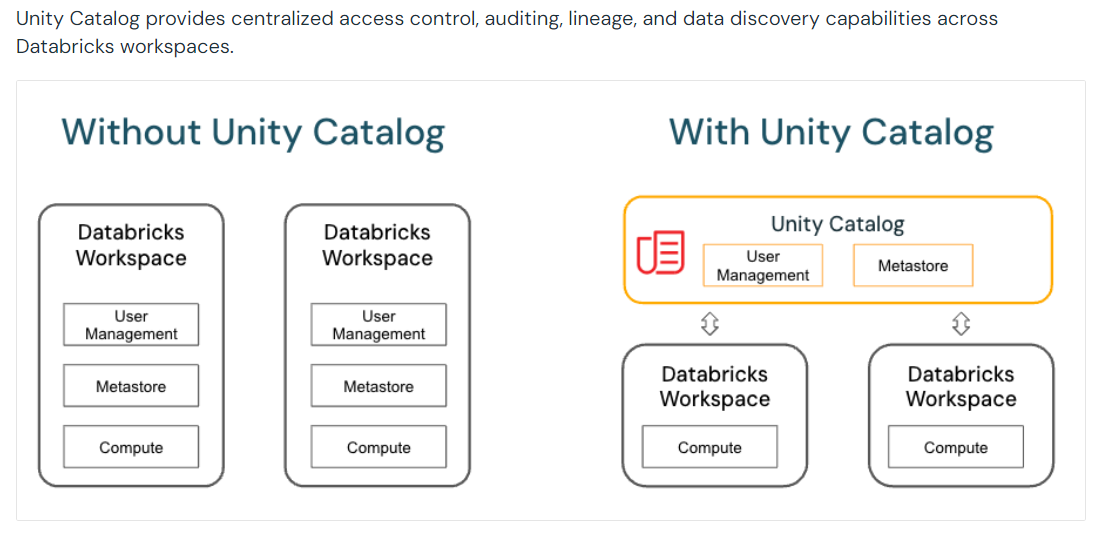
Unity catalog in Azure databricks

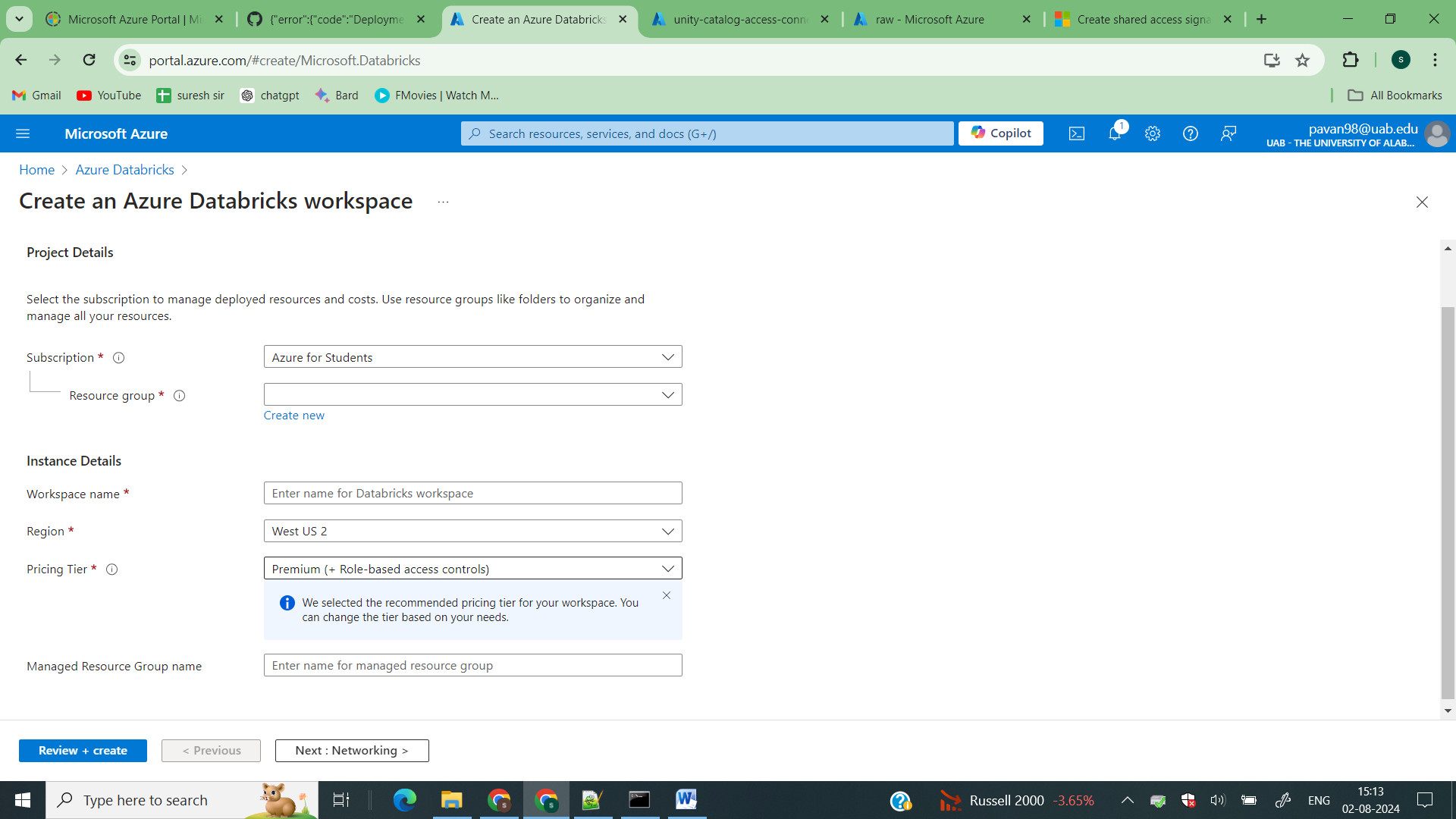


With unity catalog all the data governance comes under on roof.

Previously we have all the metastore and the permissions access also assigned to one workspace.

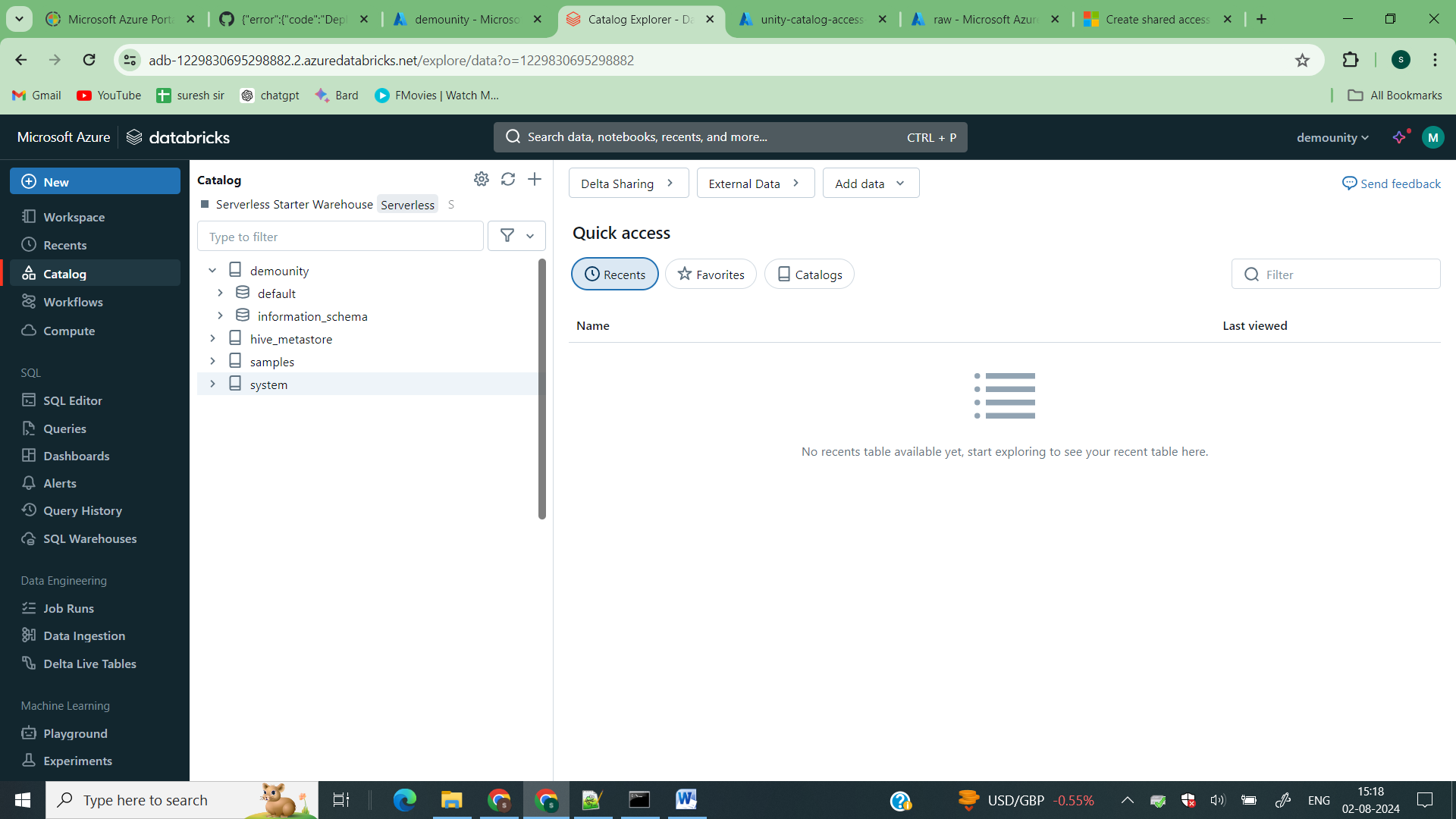
We might have any workspaces but we can manage all the data governance in one space

You should have one metastore for each region in which you have workspaces. Typically, a metastore is created automatically when you create a Databricks workspace in a region for the first time.



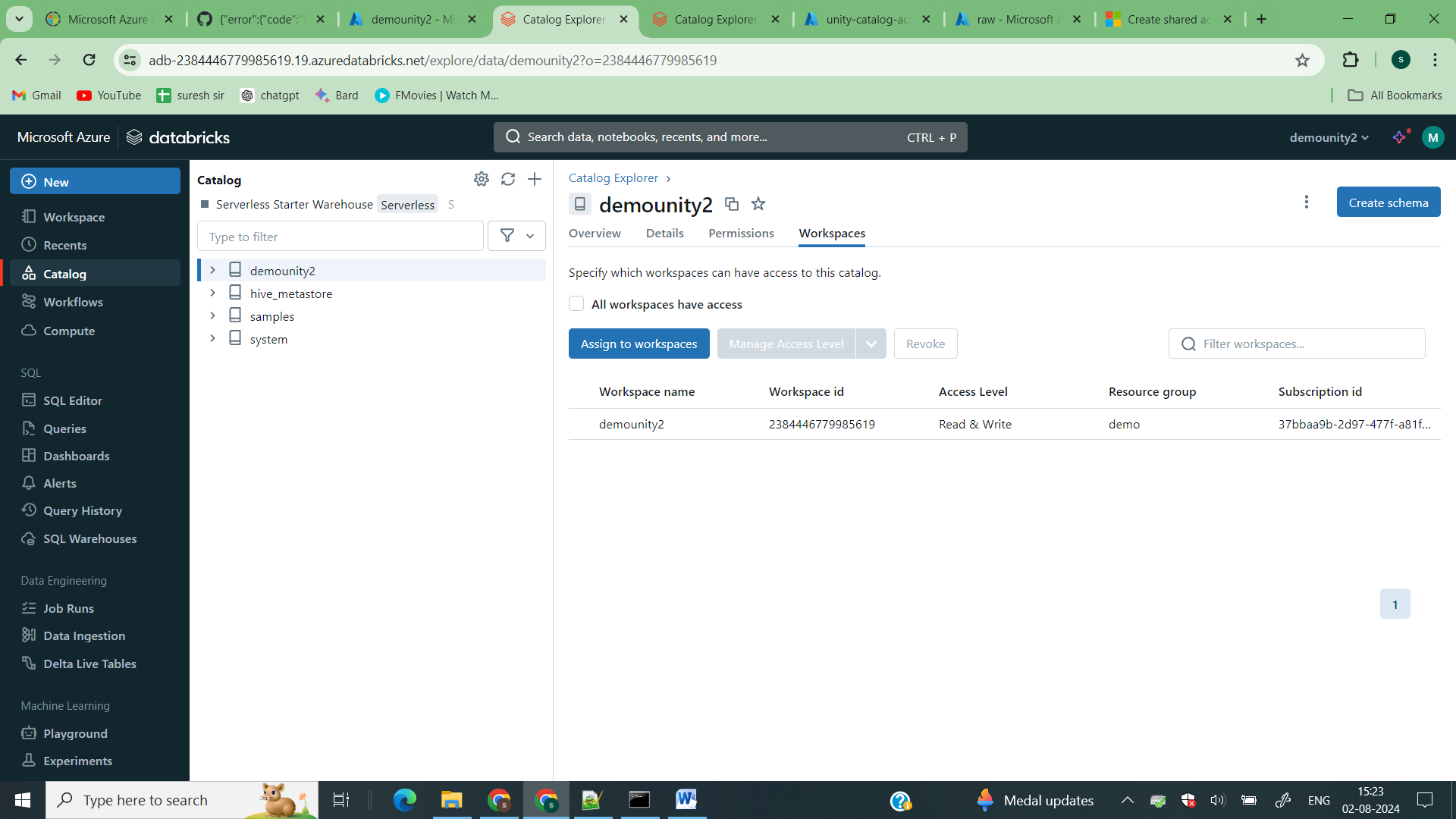
How to share catalogs from one workspace to another workspace

Step1 Create a databricks



If you see in the catalog by default a catalog with your databricks workspace is available and all the metastore information will be handled by azure.

Step2 I created another databricks workspace demounity2



If you see I have default catalog available for this workspace.

%sql

CREATE SCHEMA IF NOT EXISTS demounity2.MEC\_STD\_DB

COMMENT 'Schema for storing standard data';

CREATE TABLE IF NOT EXISTS demounity2.MEC\_STD\_DB.contact (

    contact\_id INT,

    first\_name STRING,

    last\_name STRING,

    email STRING

) USING DELTA;

INSERT INTO demounity2.MEC\_STD\_DB.contact (contact\_id, first\_name, last\_name, email) VALUES

(1, 'John', 'Doe', 'john.doe@example.com'),

(2, 'Jane', 'Smith', 'jane.smith@example.com'),

(3, 'Alice', 'Johnson', 'alice.johnson@example.com'),

(4, 'Bob', 'Brown', 'bob.brown@example.com'),

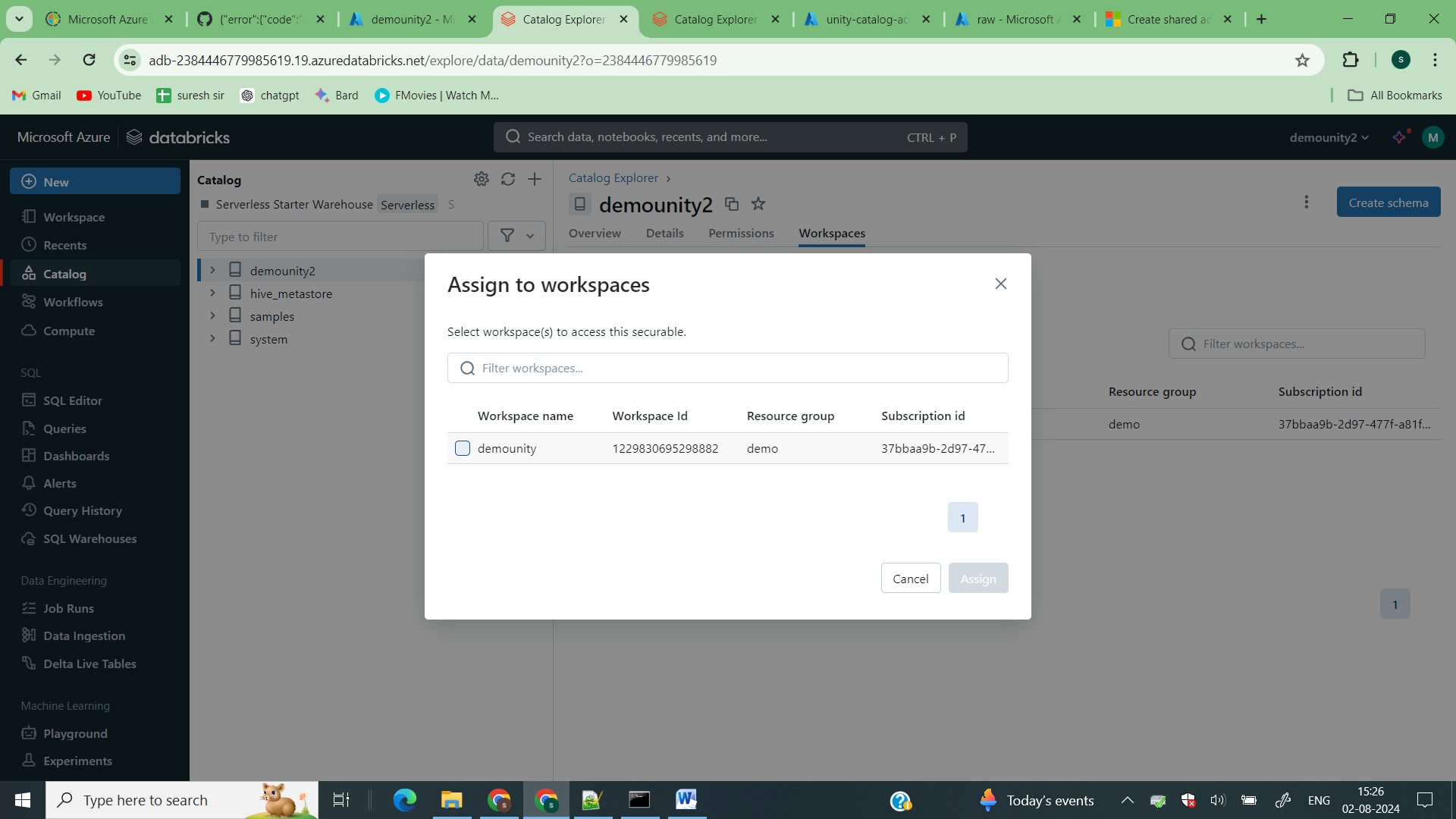
(5, 'Charlie', 'Davis', 'charlie.davis@example.com');

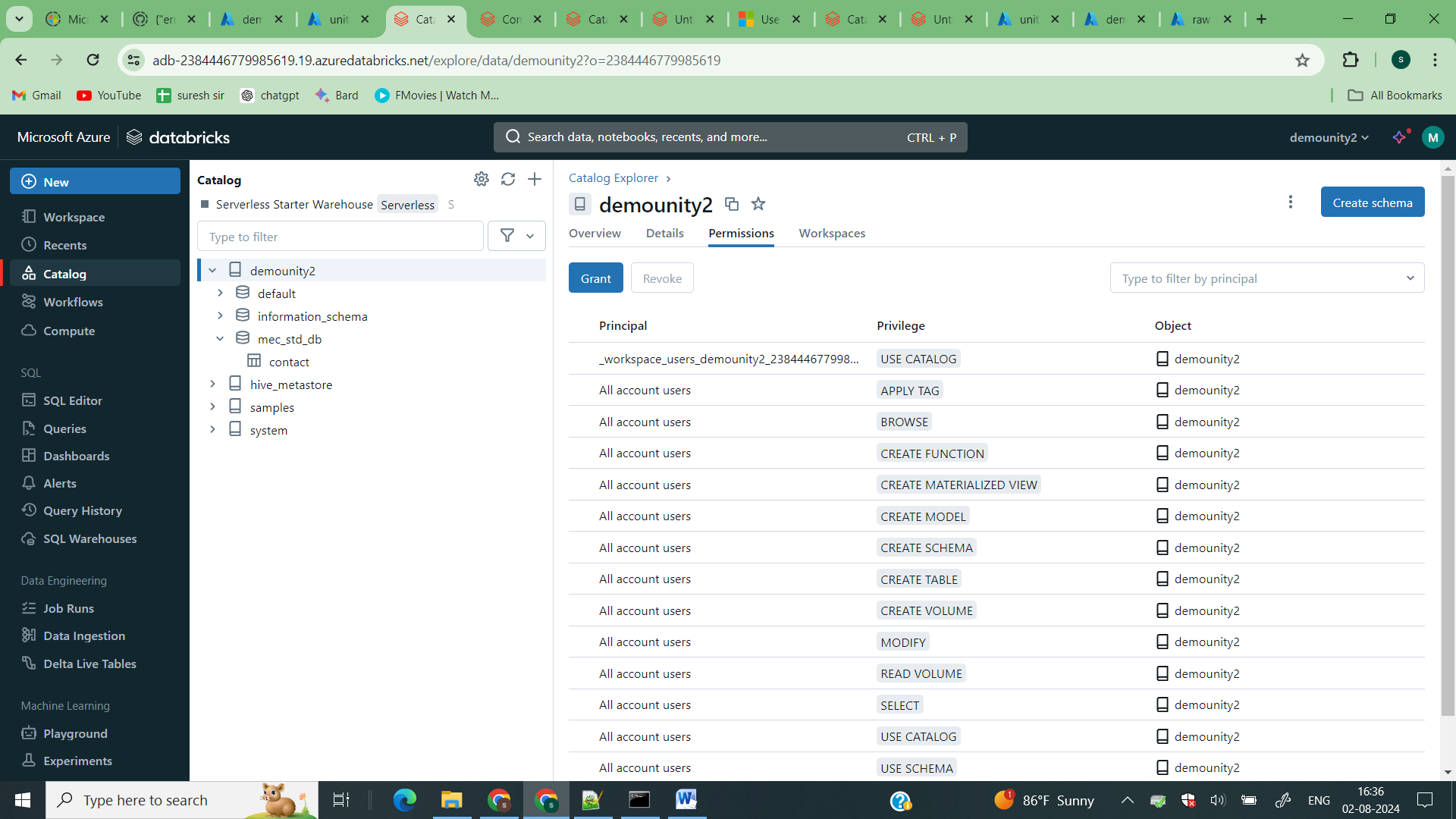
Run this ecript to create schema and tables

Step3 – Managing catalogs to workspace level

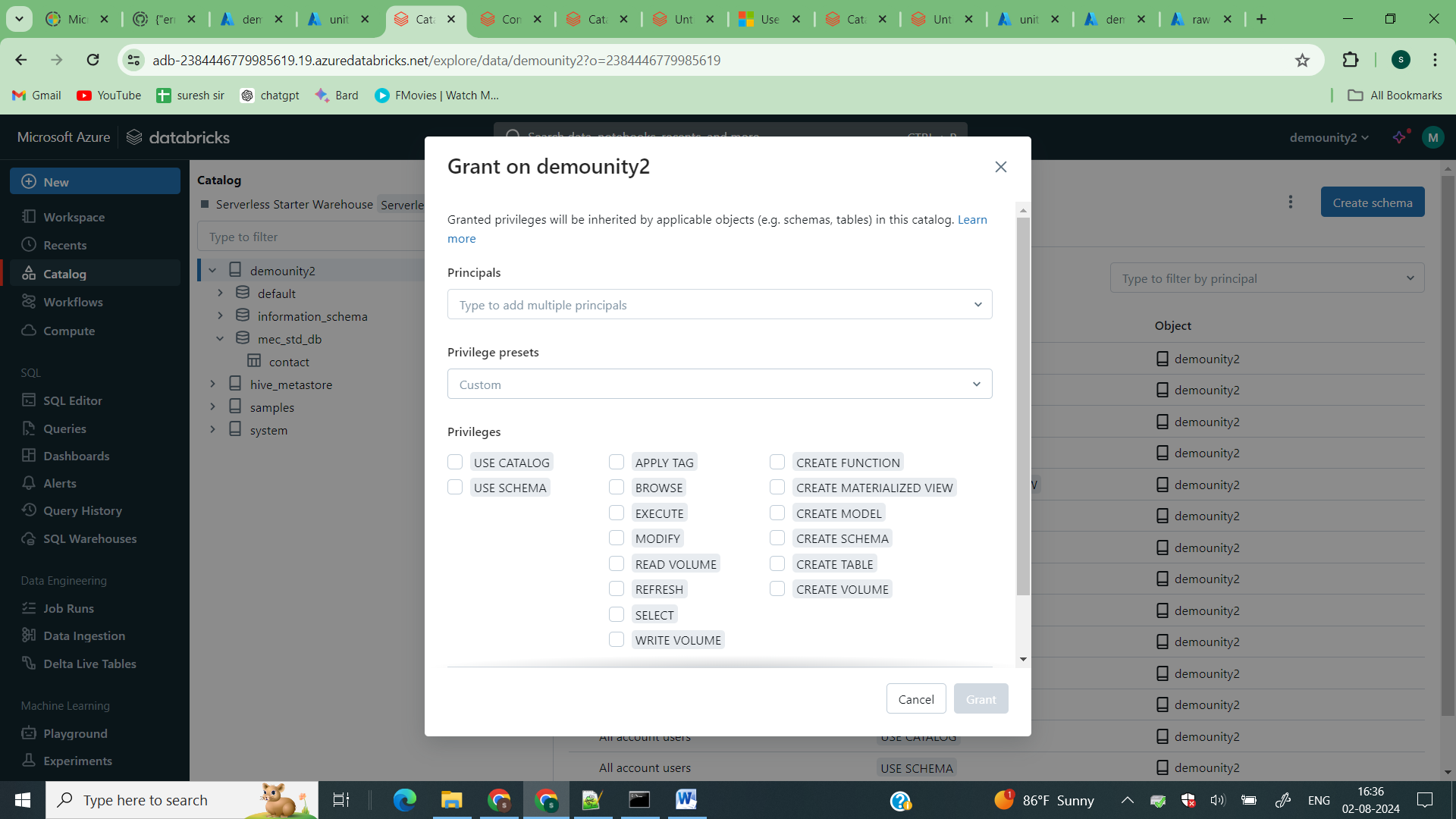
Now, If I want this catalog to present in another workspace I can do it from here

Click on assign to workspace

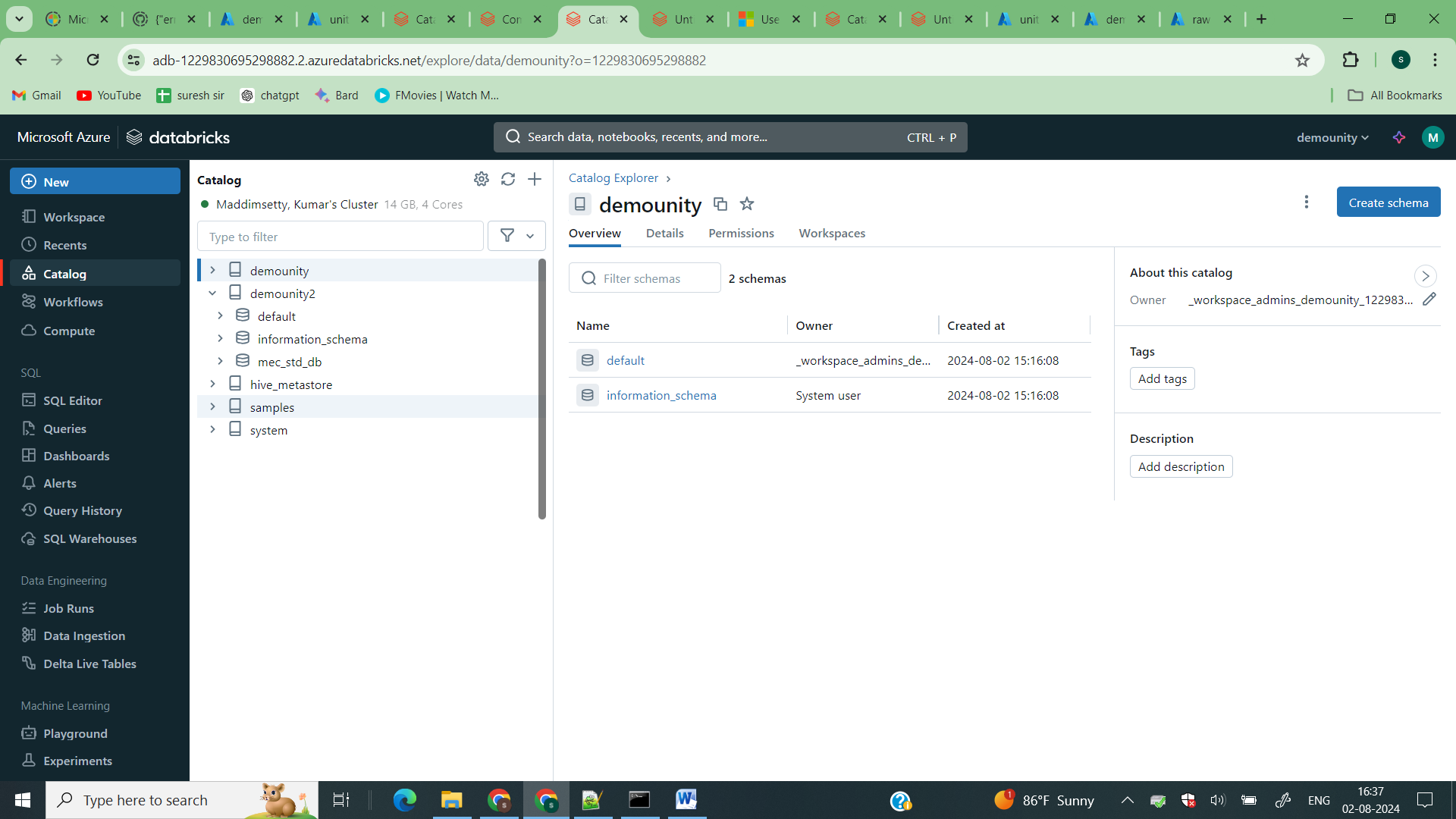




Click on permissions



Grant permission to users or user group and give priviliges

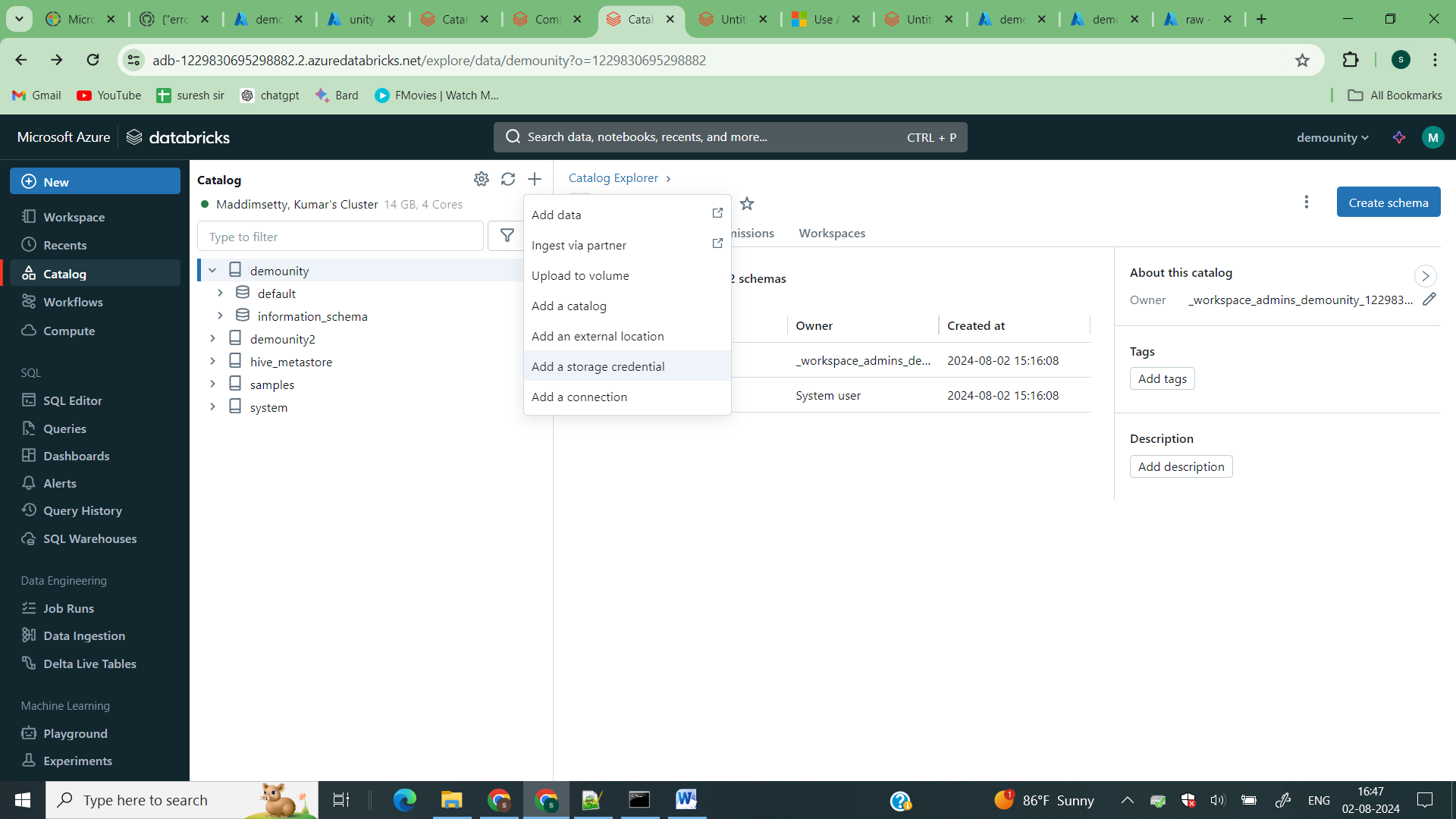


Now in the catalogs of demounity you can see the demounity2 catalog

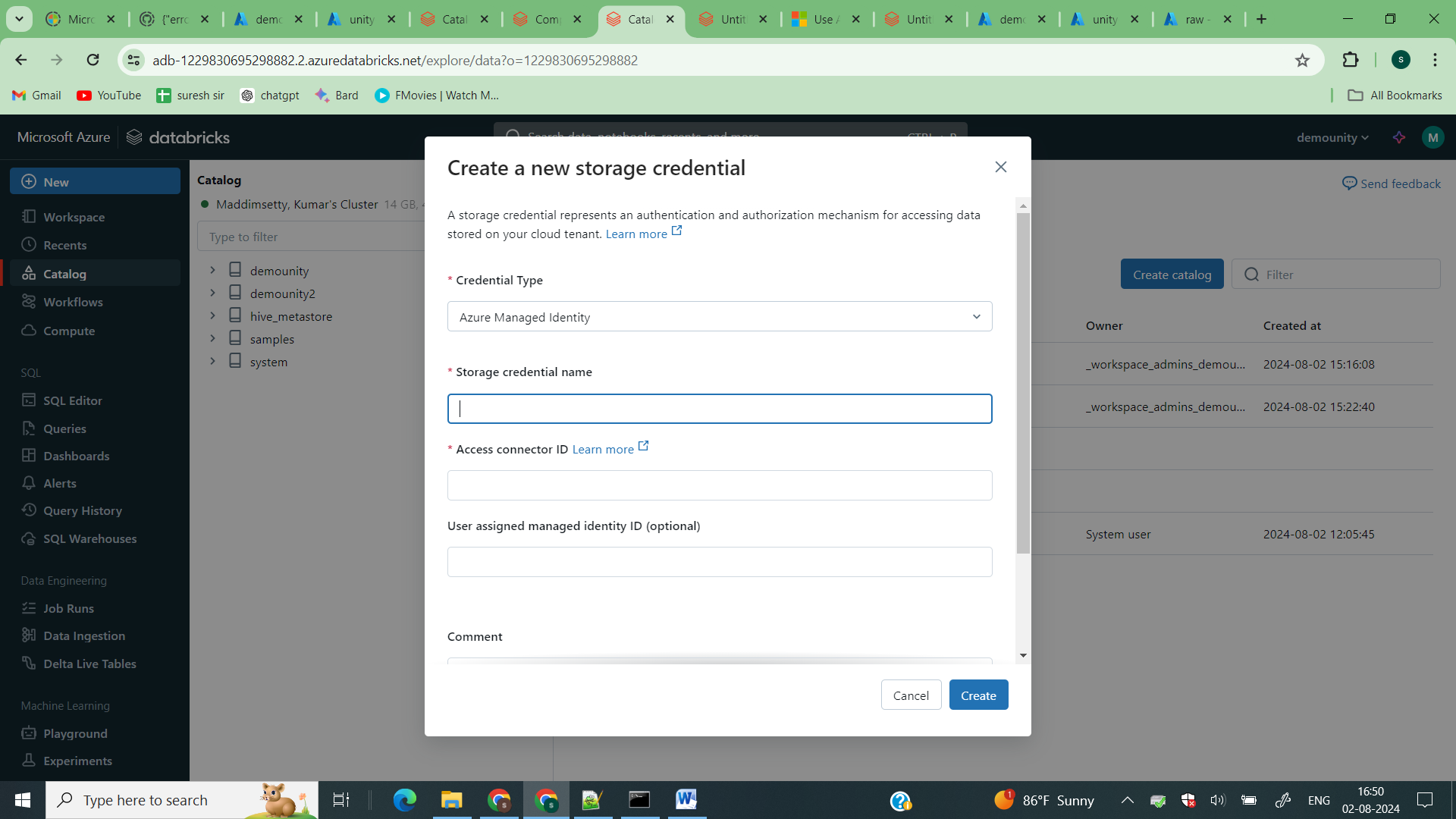
So this is how you can assign the catalogs to respective workspaces like u will map dev catalogs to dev workspaces and respectively. And you can access the data also

Lets see how to create our own catalog pointing to external location.

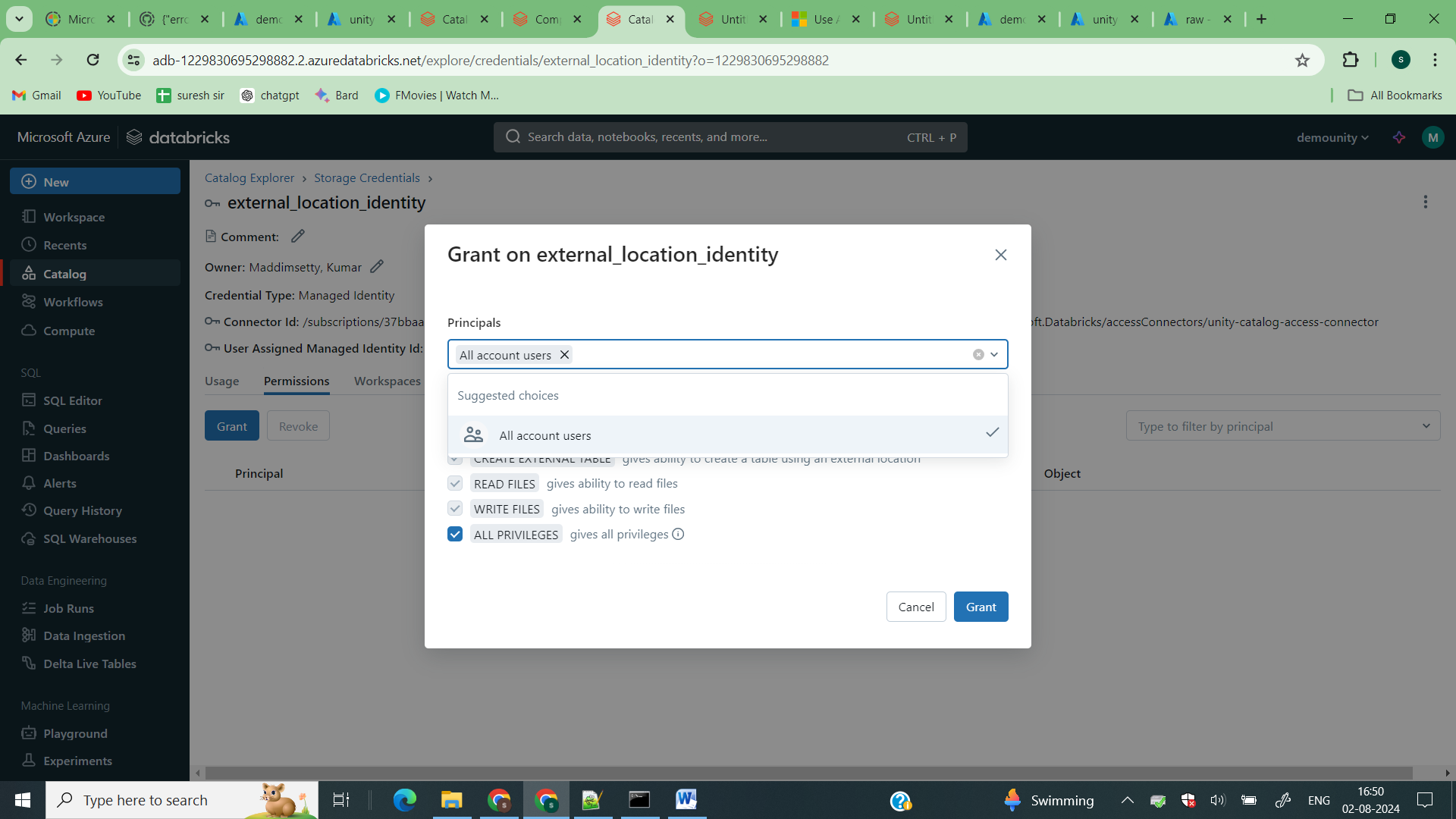
1st you need to create a credantials.



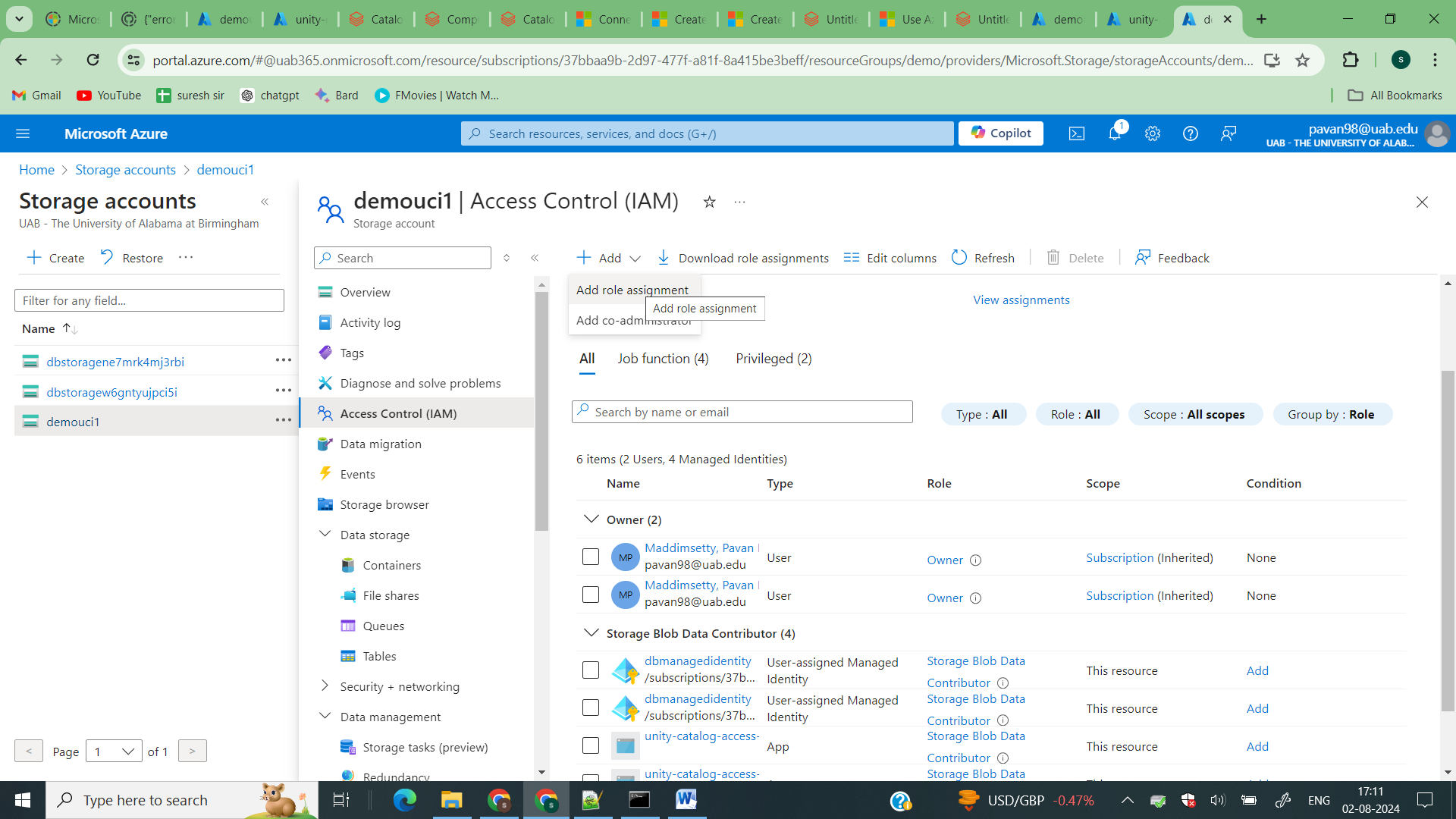
Click on add storage credential



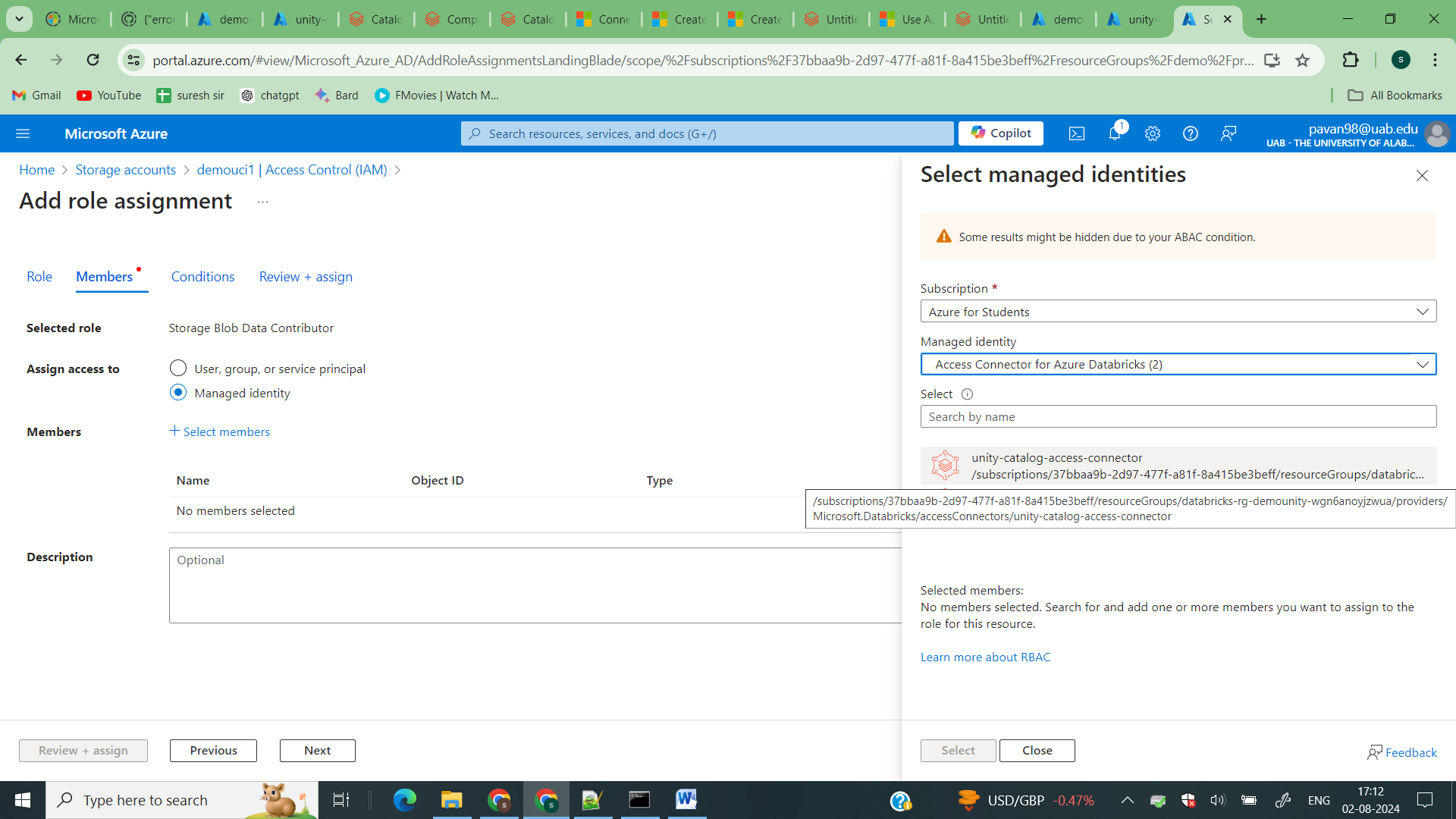
Enter the resourceid of your default databricks access connector in access connector id. And create

 Now grant permissions to the credantials.

Now add the databricks access connector to the adls to access

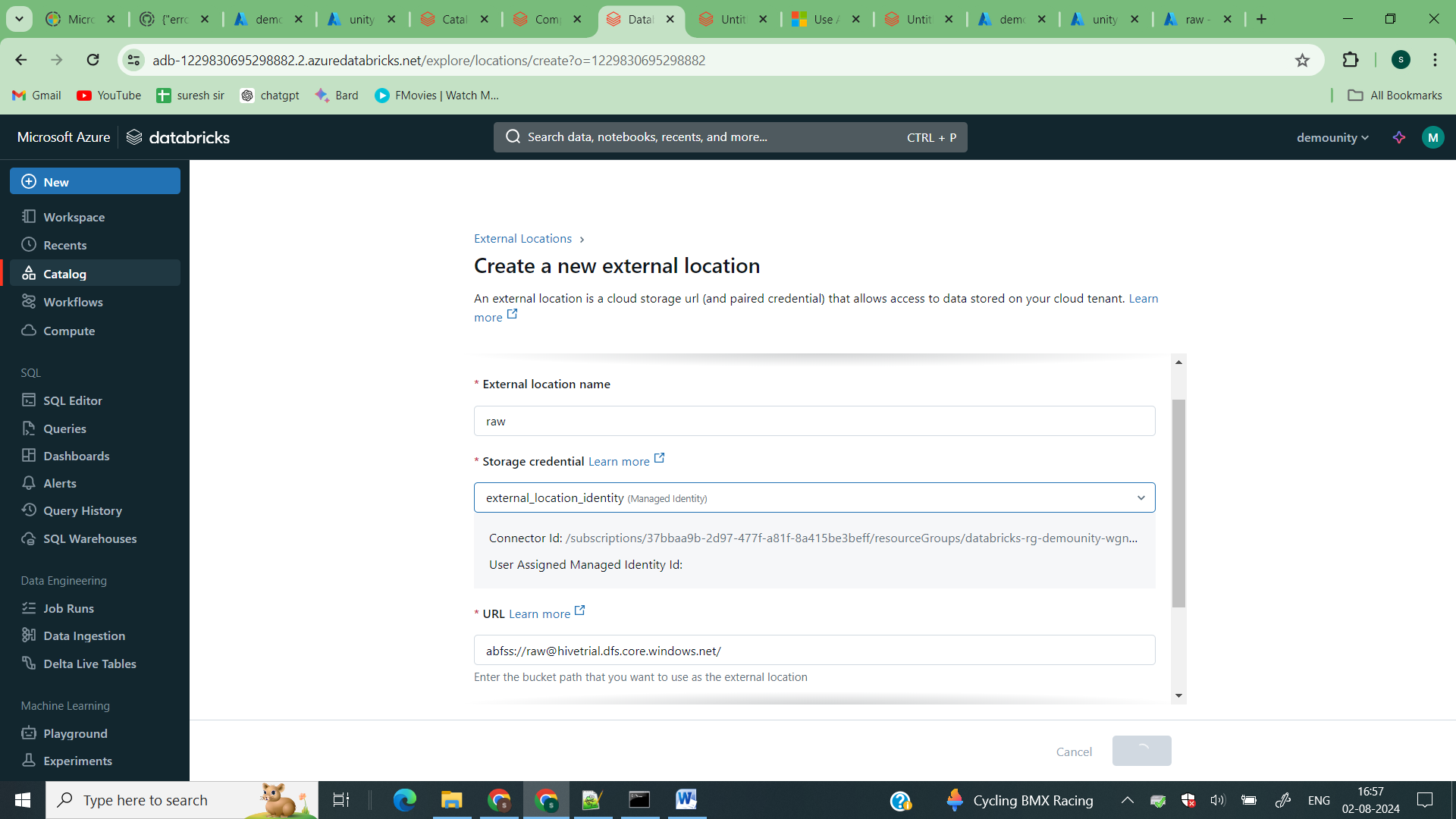


Open your external connection and add role assignement and select storage blob connector role.

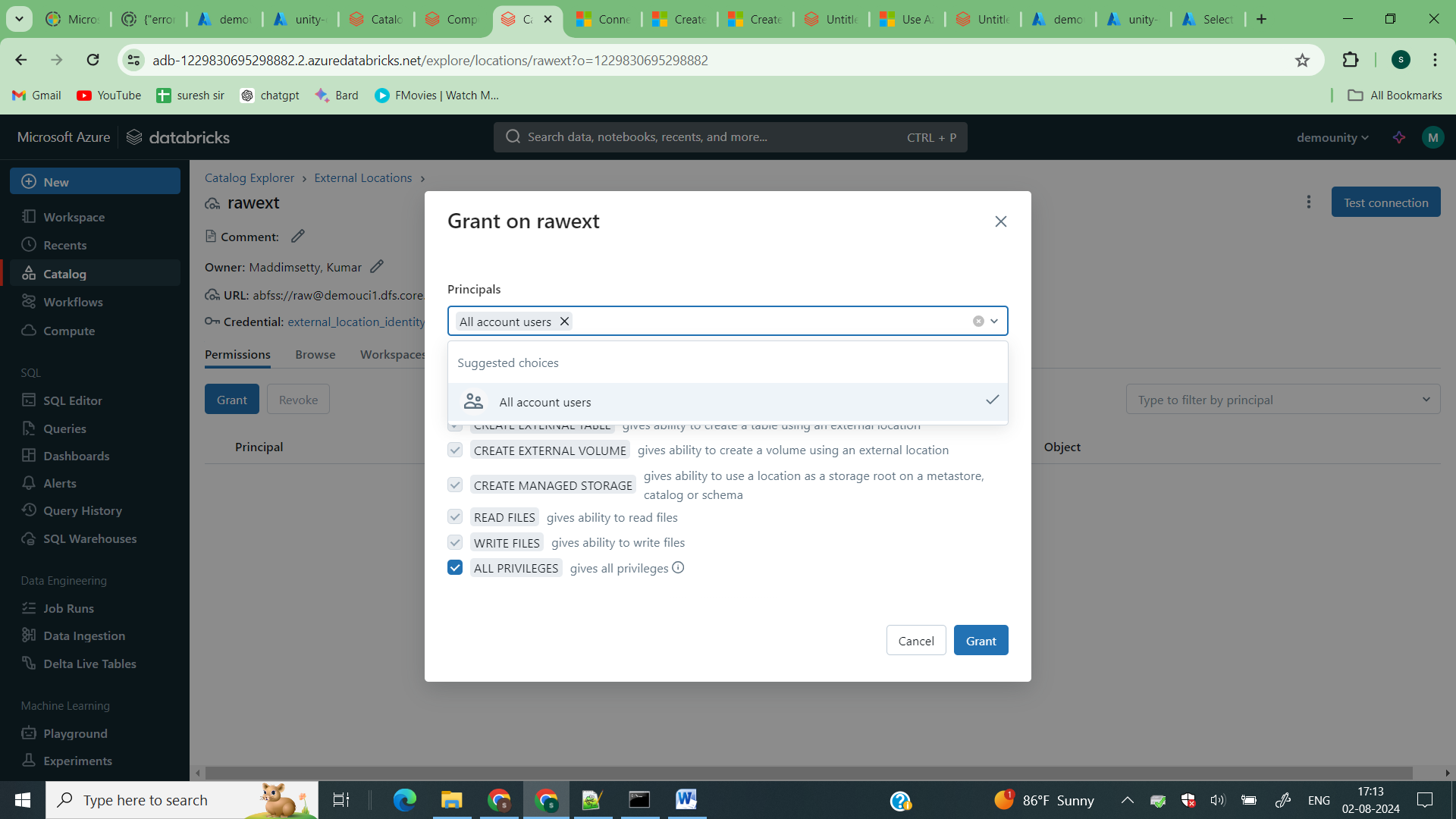


Add your access connector

Now again click on create new external location

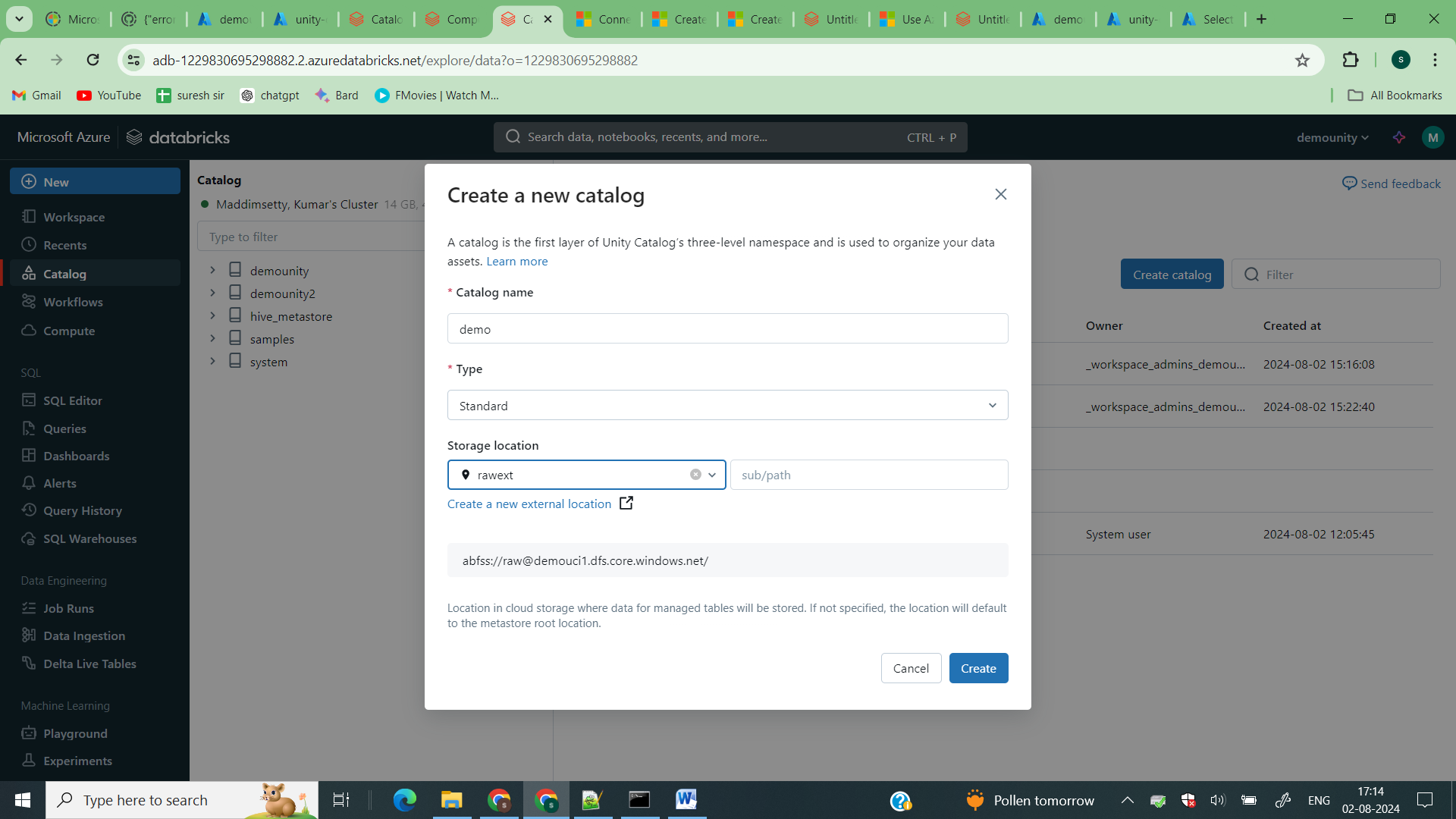


Click on create

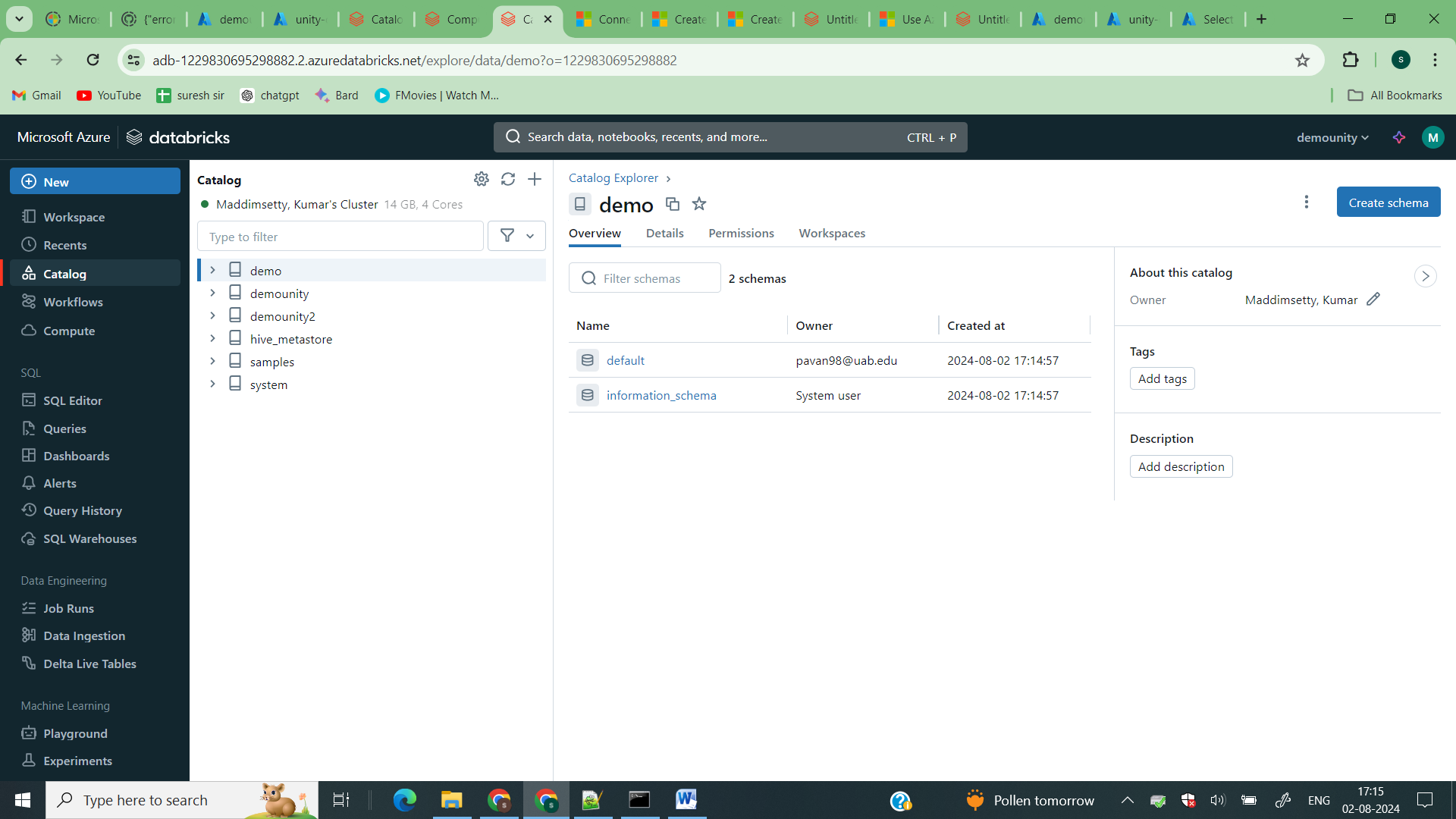


Then grant permission

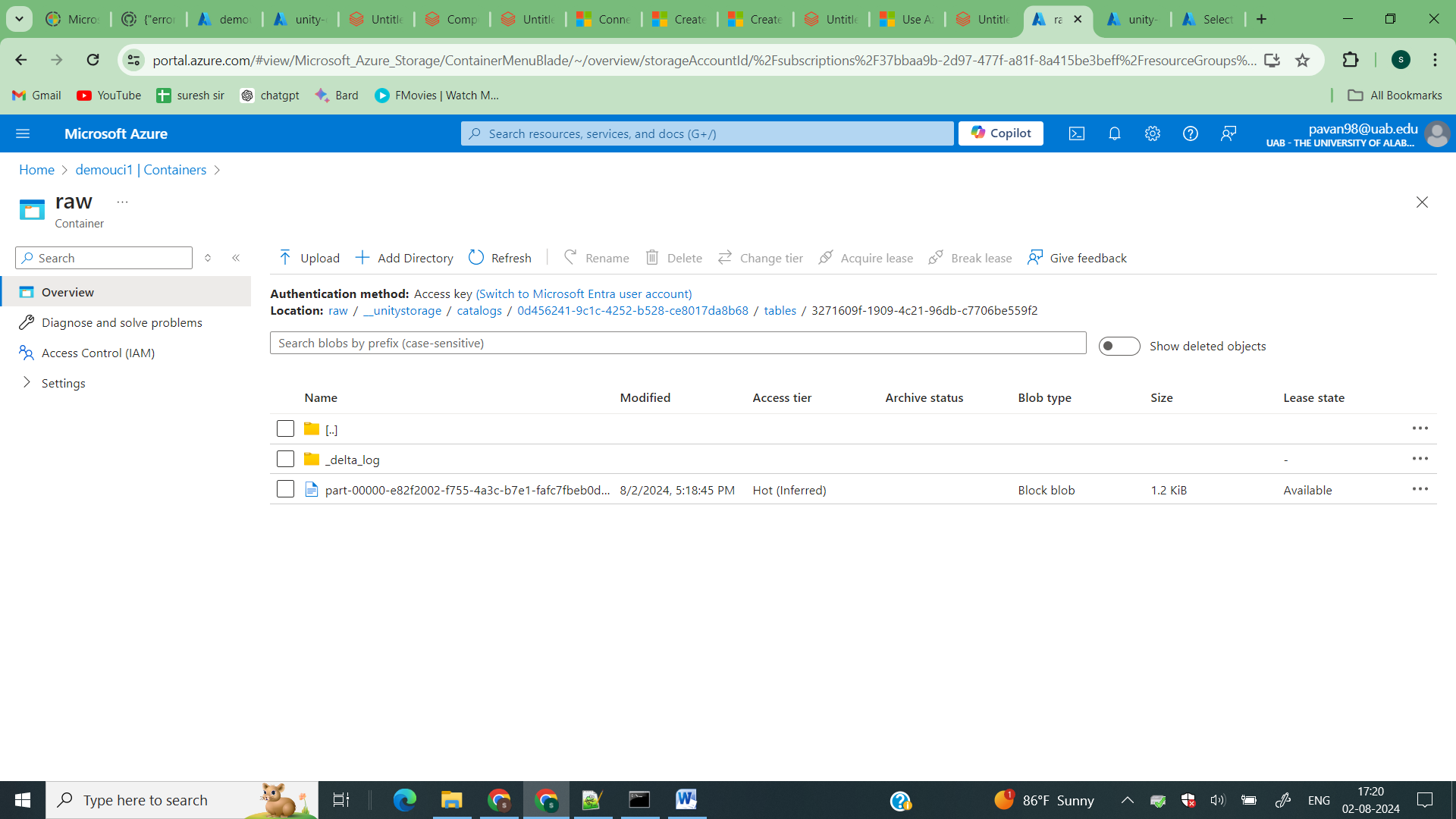
Now click on Add catalog



Select your external location

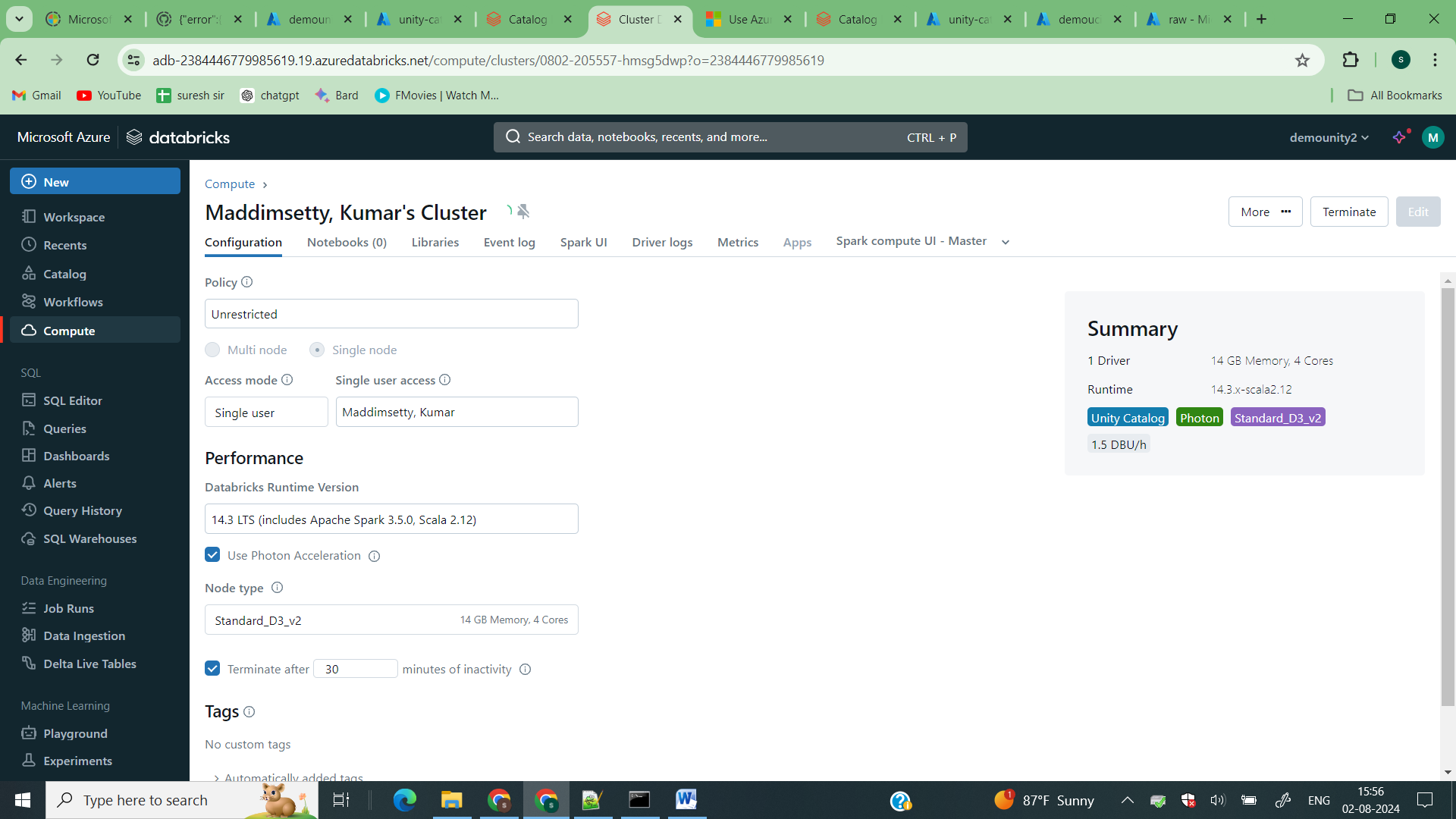


Now your external catalog is created lets create schema and tables in this external location.

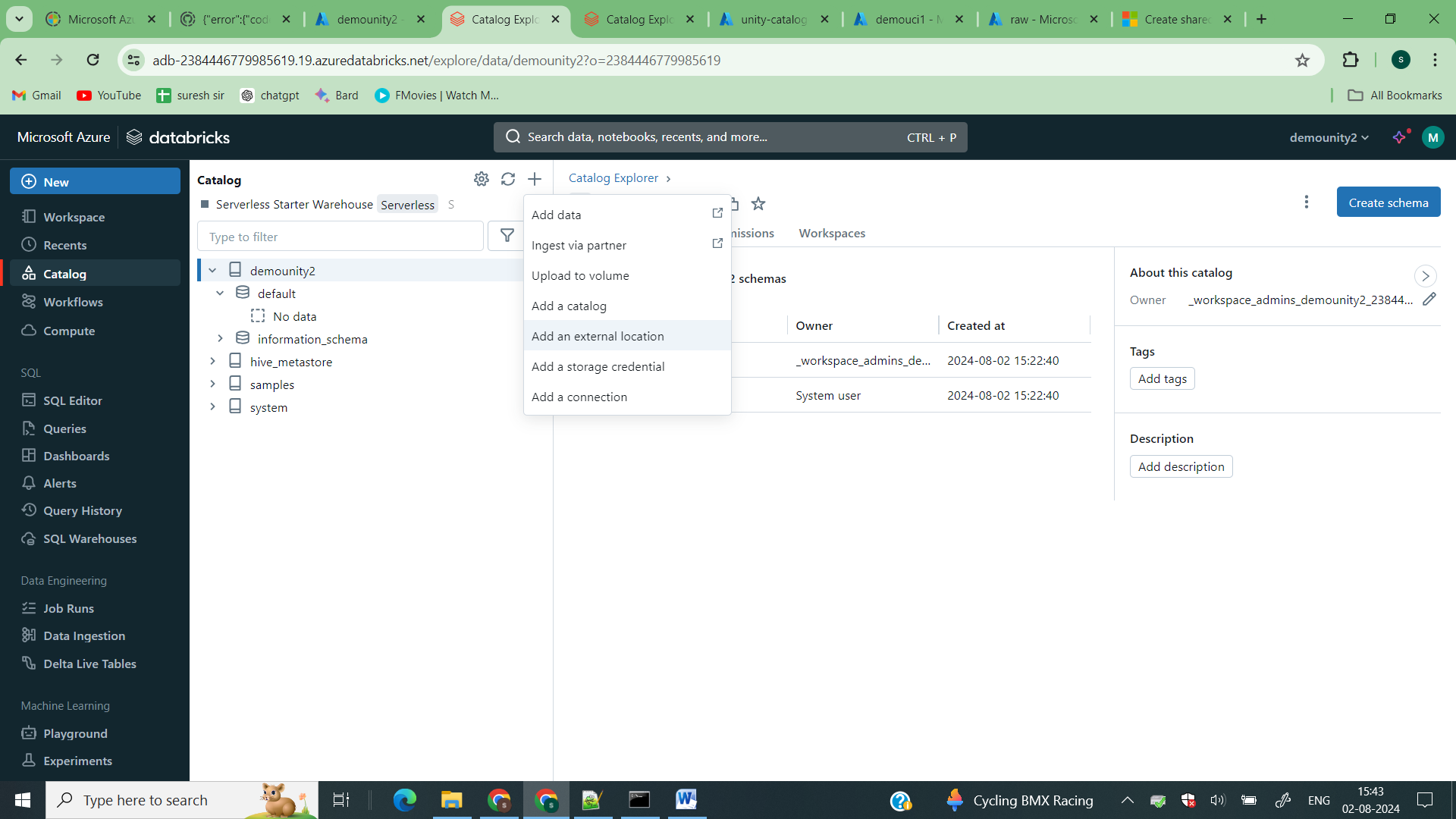


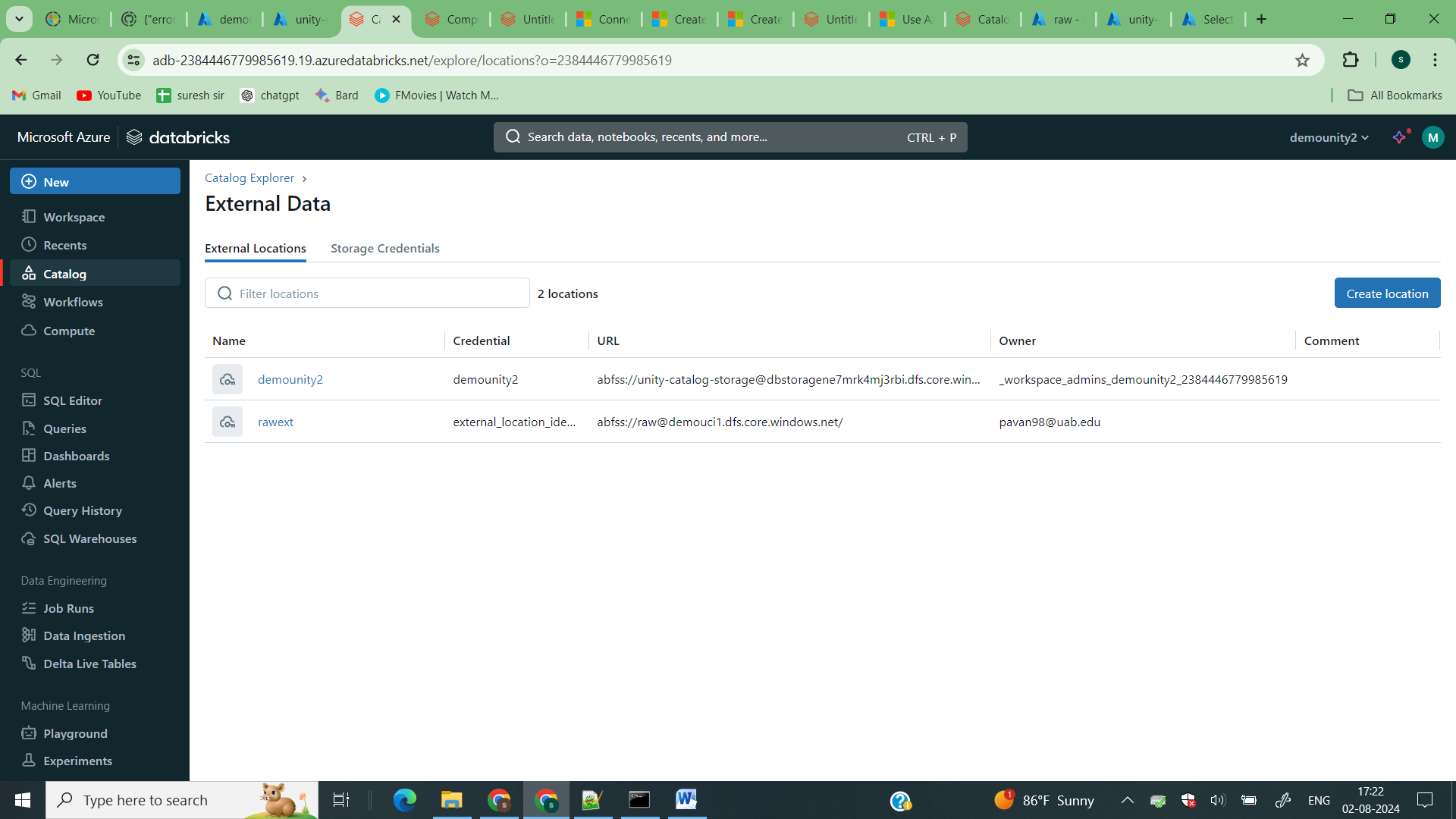
If you go to your storage account this is how the delta tables are created

Creating a cluster



How to add external locations in unity catalog



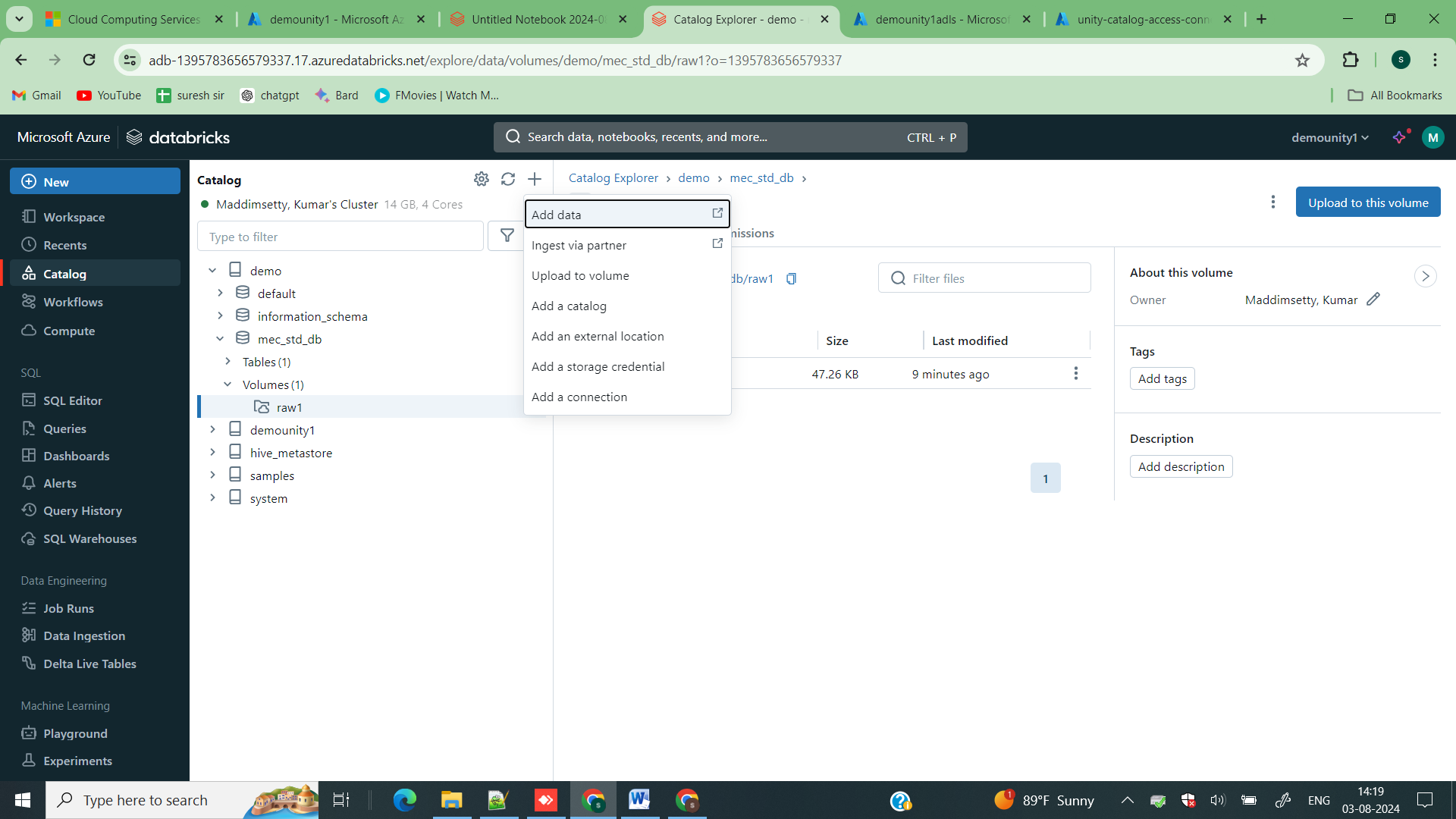


If you see in your 2nd databricks workspace u can see all the external locations catalog and everything all your workspaces if you want u can restrict the access to the workspace you click on catalog and go to workspaces tab there you can manage.

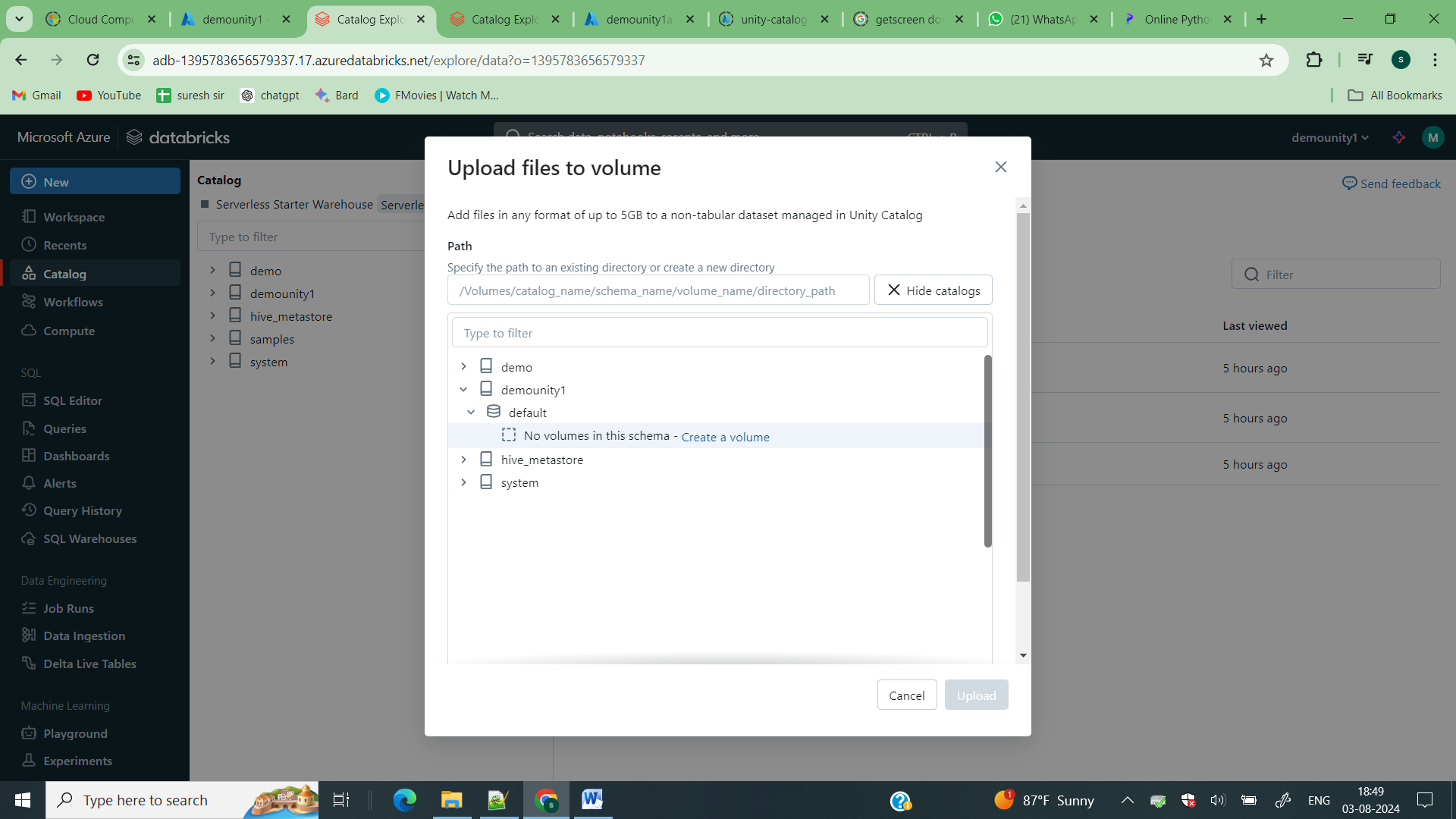
Usage Of Volumes

Before unity catalog we need to mount to adls using the abss:// path everywhere

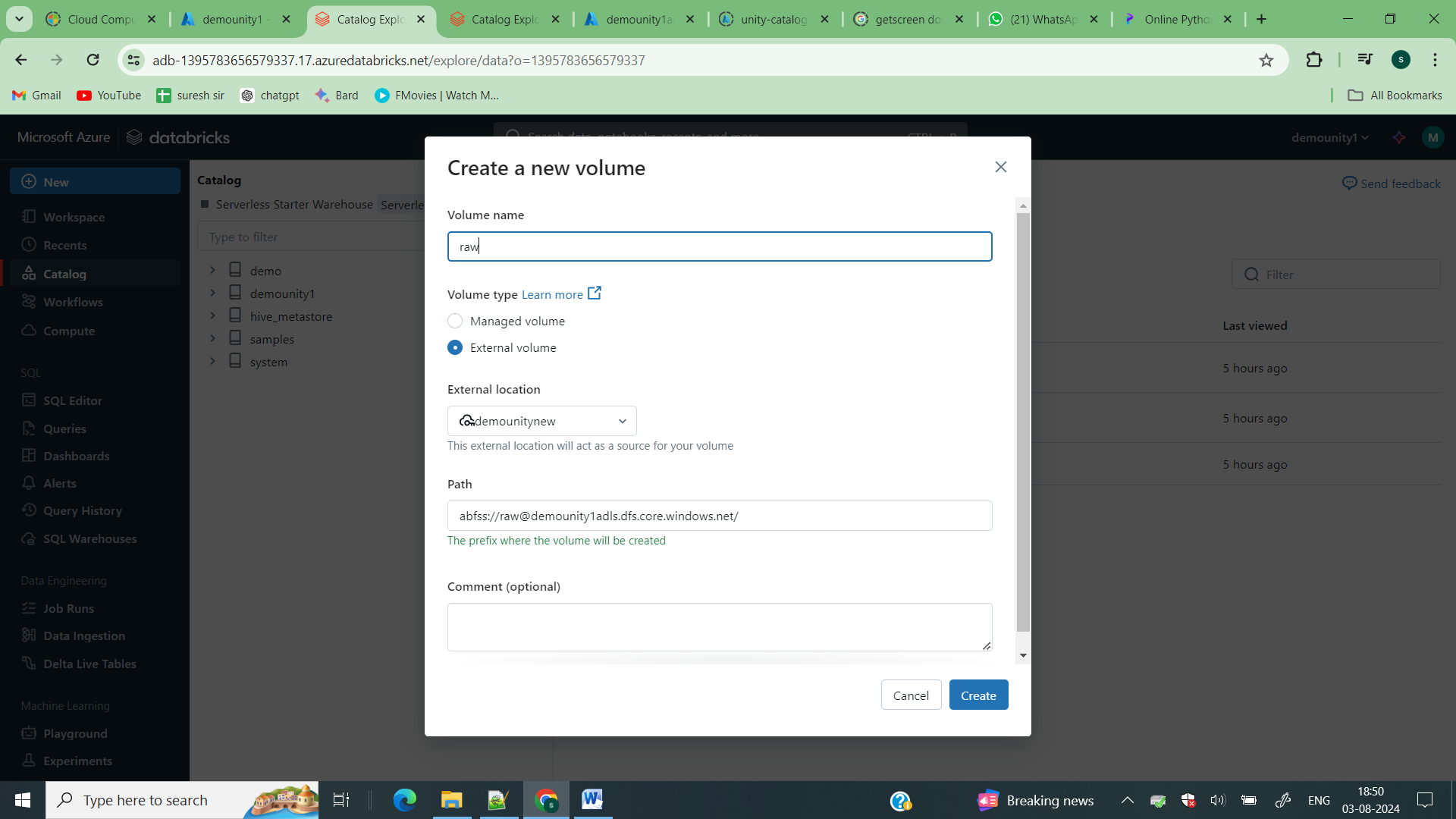
But now we can create volumes and we can access the files and upload the files directly from databricks to adls



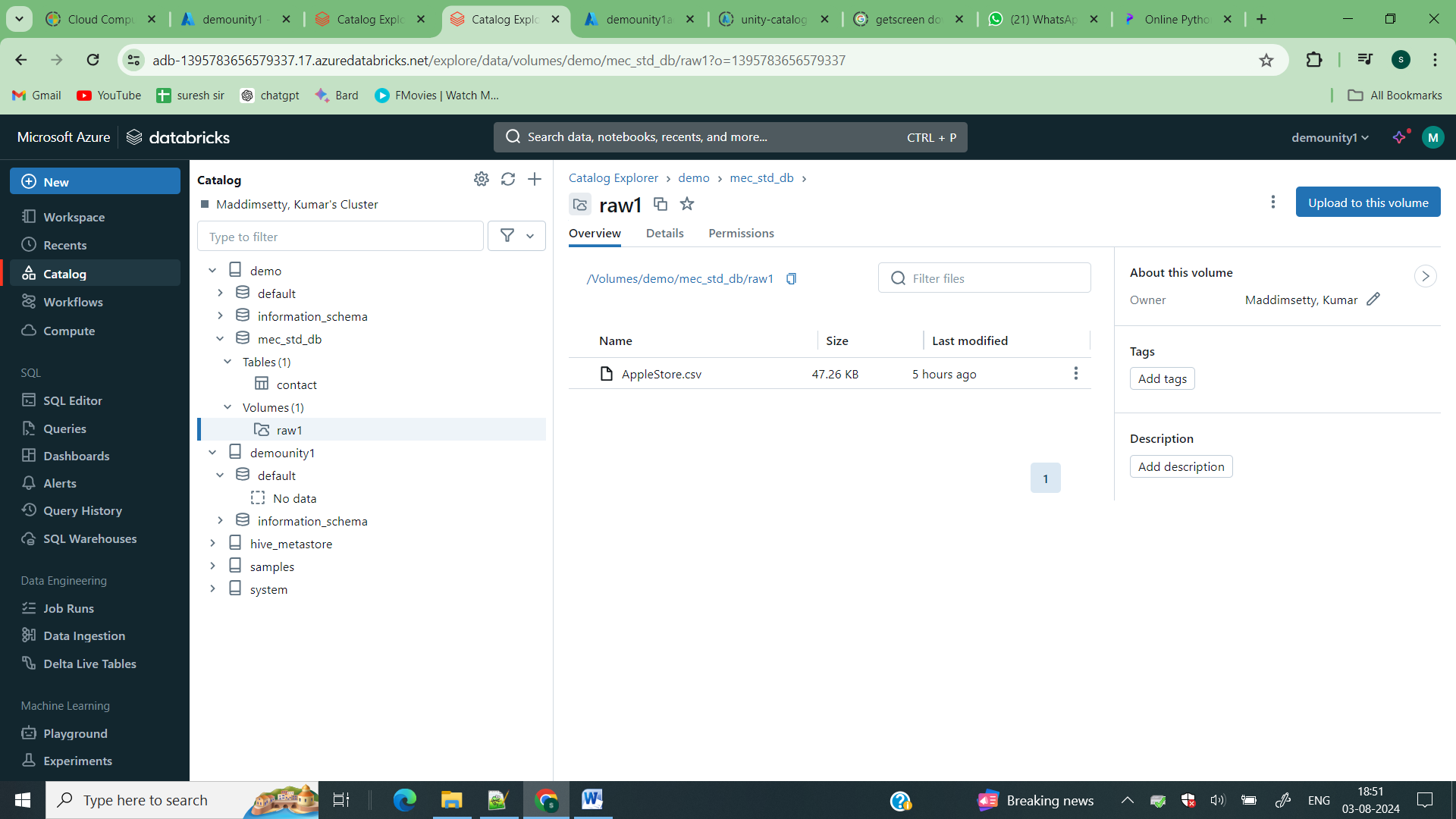
Click on upload to volume



Click on create volume.



Select your external location and click on create



So you can access all the files in your adls directly from volumes and you can governance it by using setting the permissions.

df = spark.read.format("csv").load("/Volumes/demo/mec\_std\_db/raw1/AppleStore.csv")

df.show()

instead of using dbfss: you can directly use the volumes to read the files.