Purpose of Mounting

Provides a unified access layer across multiple clusters within same workspace.

Eliminate the need for repeatedly authenticating to Azure storage.

Ensures a consistent environment for data access and collaboration.

Mounted storage can be accessed by all users without reconfiguring access.

Key vault

Key vault will be storing our passwords and secret.

Three sections in key vault

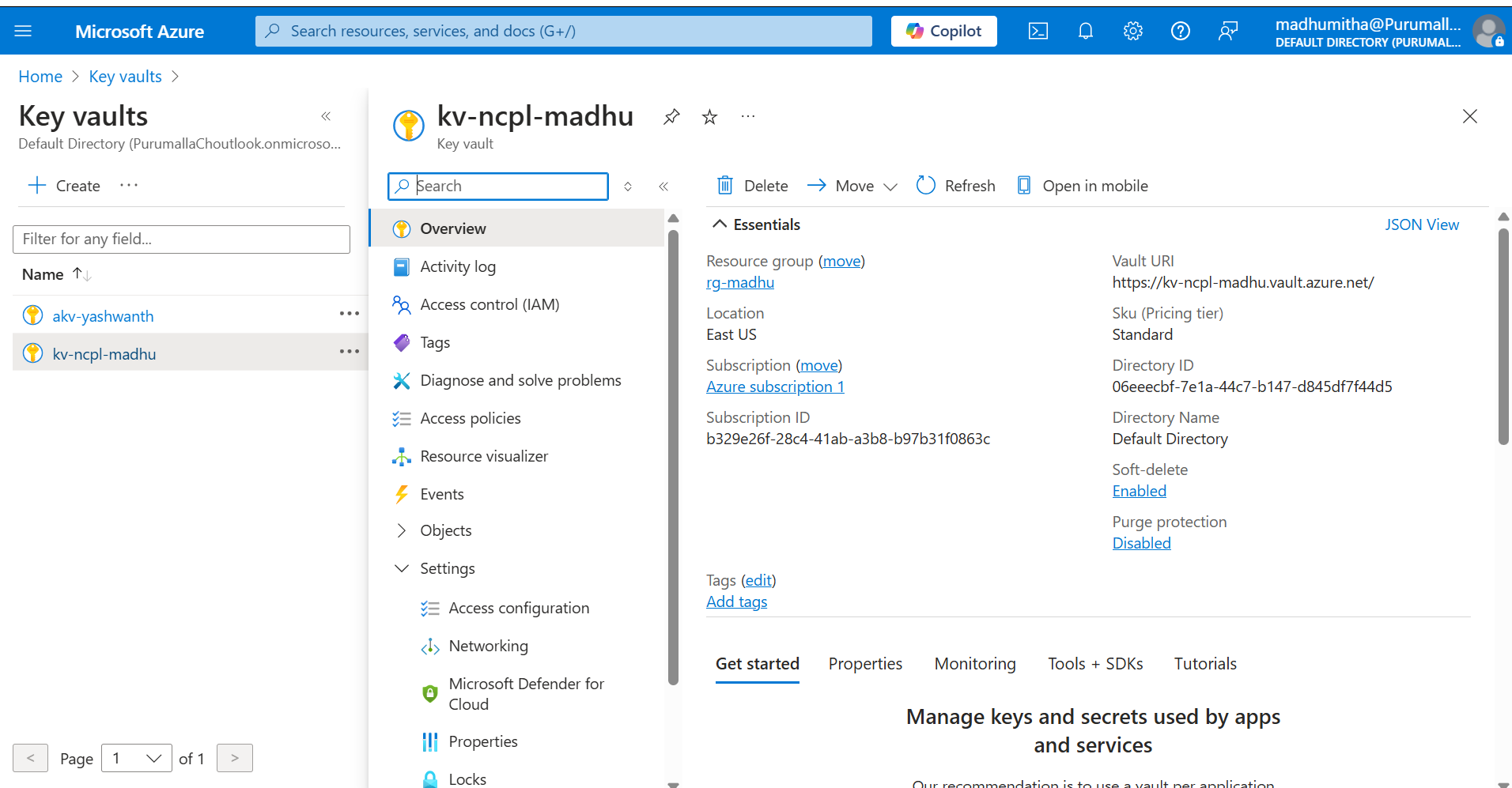
1.Keys – Generate key and use that key for disk encryption. It is used by infrastructure team.

2.Secrets – To store passwords or connection details. Secrets are used by Data Engineers

3.Certificate – Used for SSL/TLS encryption, typically for web applications.

**Create key vault**

Create resources-> add key vault name-> Enable vault access policy in access configuration-> Create



To set up secret scope and connect to Key vault

IN blob storage->Under security+networking-> select SAS-> Enable service, container, object in allowed resource type-> Generate SAS token

Go to resource-> key vault->click secret->create a secret->name it as sas-key->In secrete value paste SAS URL ->create

**Mount Blob Storage Using Secret Scope**

Open databricks, remove up to .net in URL.

Add #secrets/createScope

Give scope name as -aksconnection

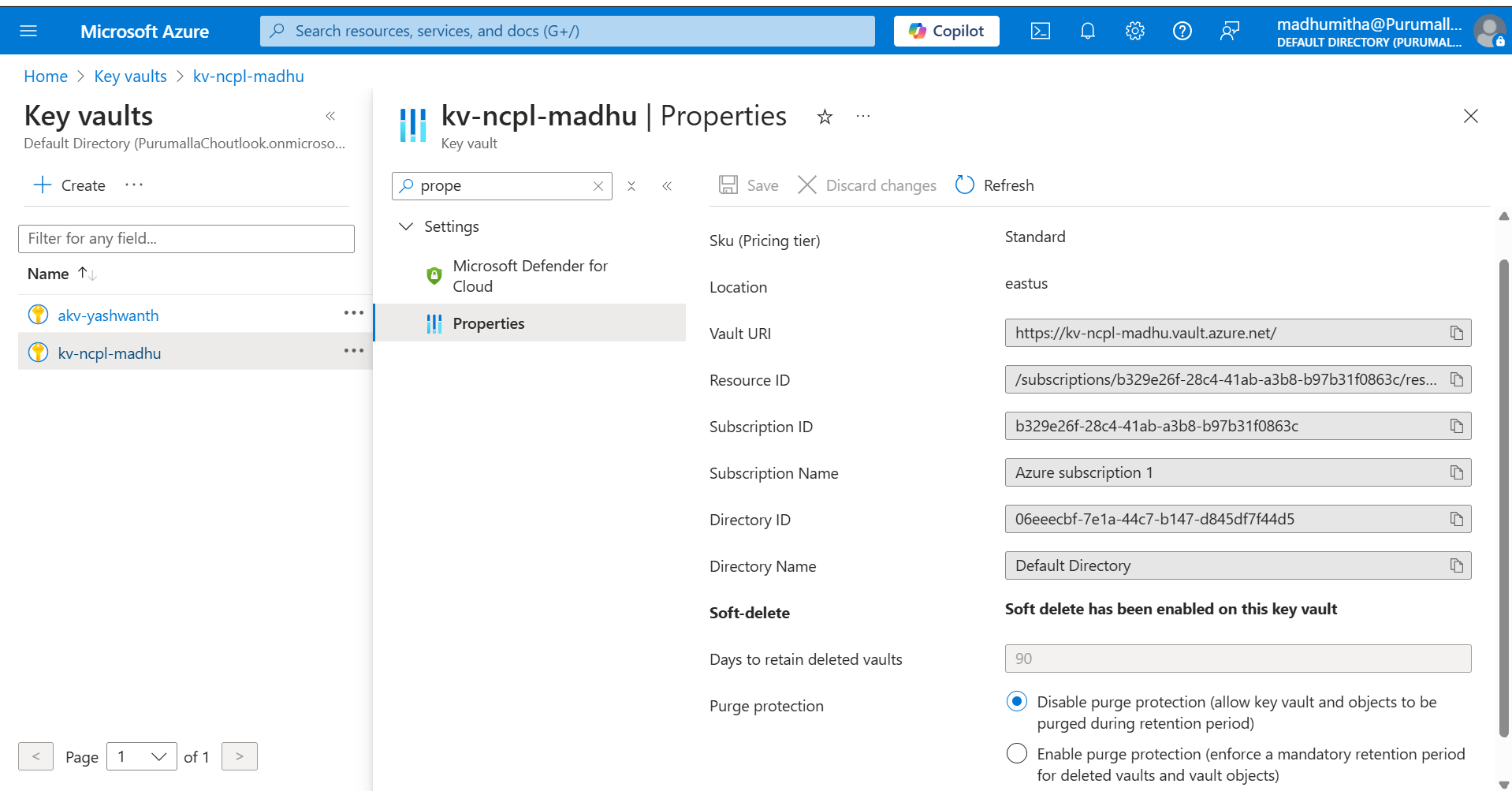
Manage priniciple – All workspace users

In Azure key vault -> Properties-> Copy vault URI and resource id

DNS name – vault URI(https://kv-ncpl-madhu.vault.azure.net/)

Resource ID – ResourceID(/subscriptions/b329e26f-28c4-41ab-a3b8-b97b31f0863c/resourceGroups/rg-madhu/providers/Microsoft.KeyVault/vaults/kv-ncpl-madhu)

Create it



Navigate to databricks

Mount the blob storage in databricks.

Add container name as edw, storage account name as ncplmadhublob, scope name as akvconnection and key as sas-key.

**Command 1**

dbutils.fs.mount(

  source = "wasbs://edw@ncplmadhublob.blob.core.windows.net",

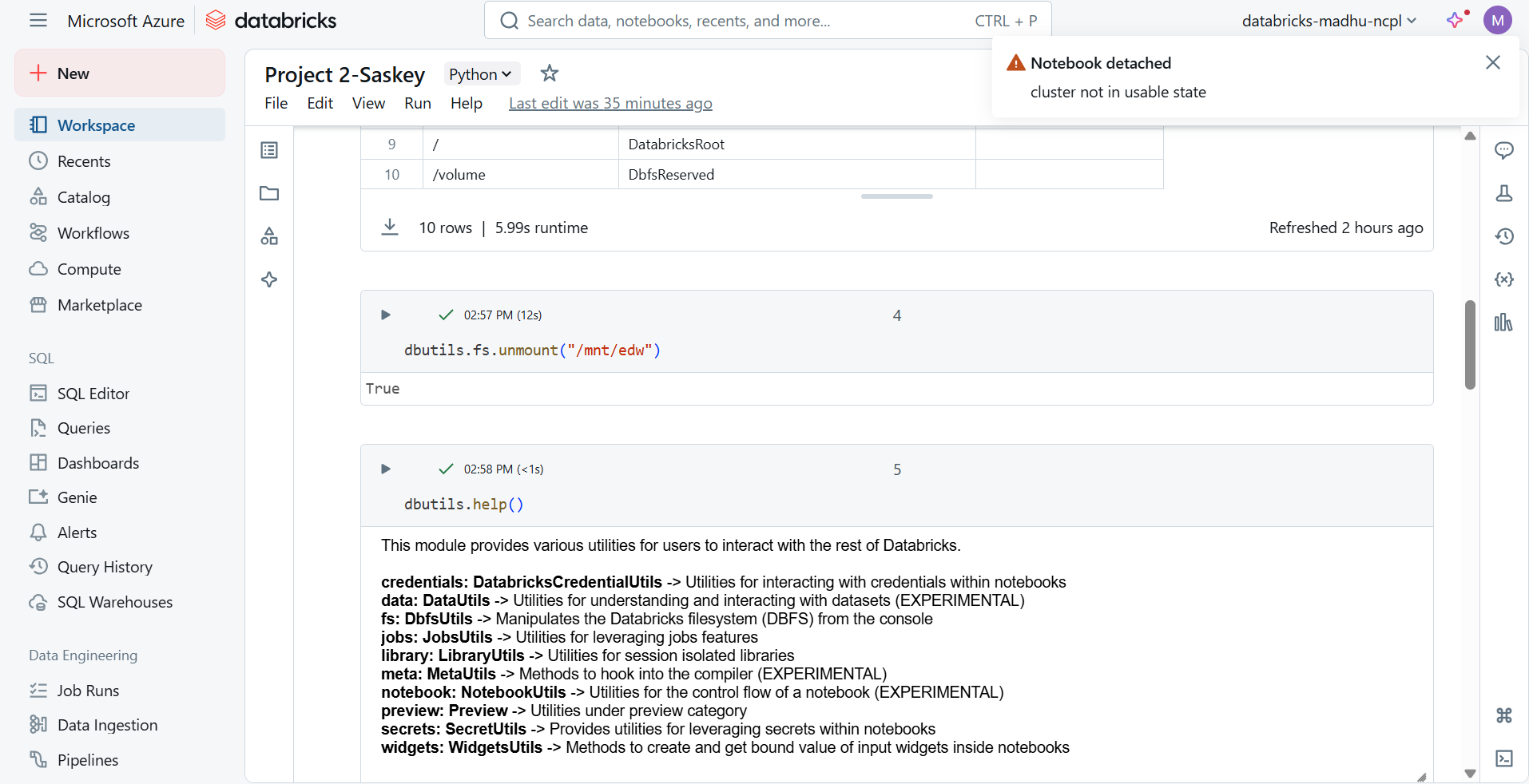
  mount\_point = "/mnt/edw",

  extra\_configs = {"fs.azure.sas.edw.ncplmadhublob.blob.core.windows.net":dbutils.secrets.get(scope = "akvconnection", key = "sas-key")})

The above command will show error if you mounted the edw container earlier. To avoid the conflict unmount the container by using the below command

**Command 2**

dbutils.fs.unmount(“/mnt/edw”)

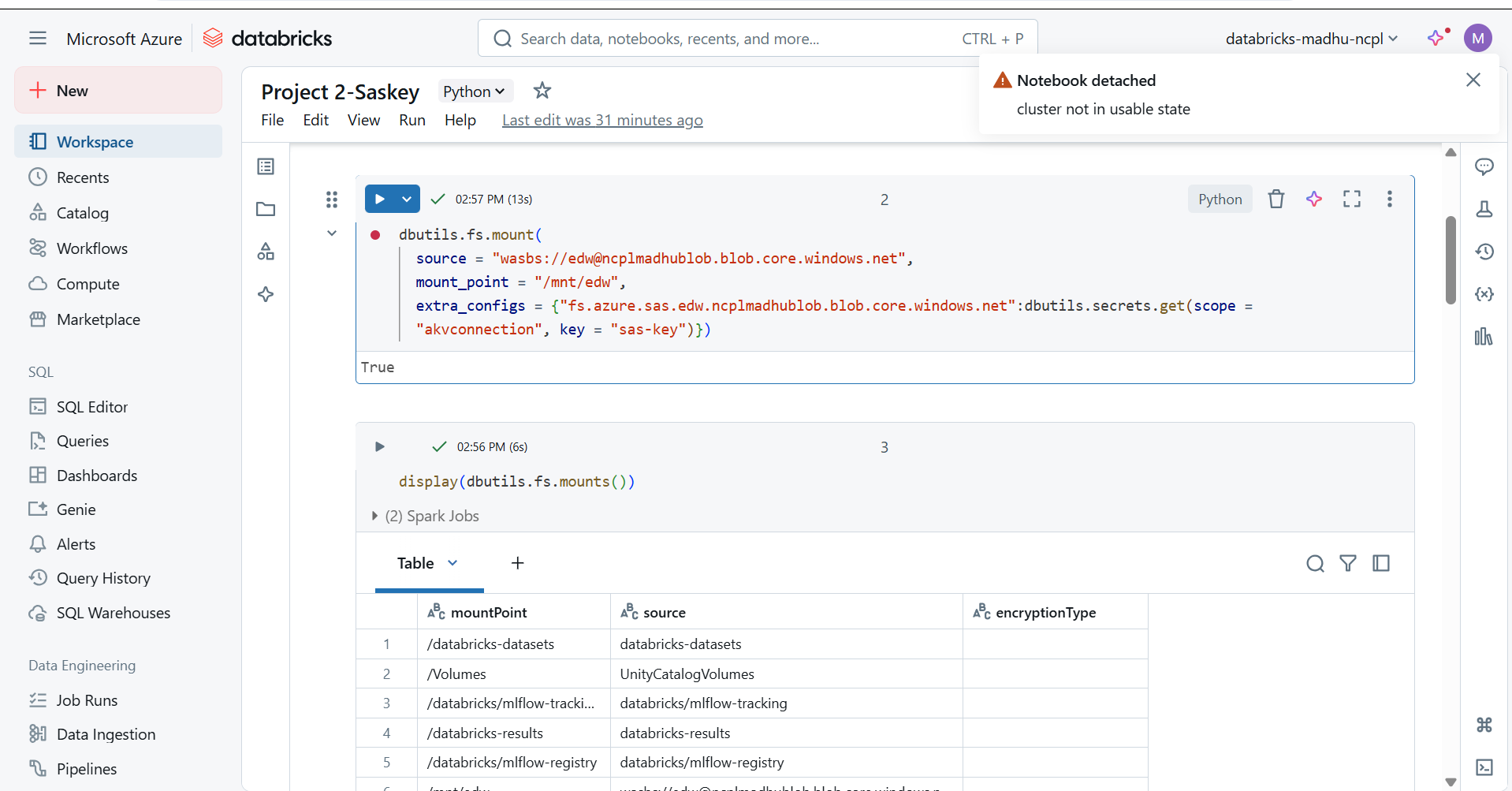


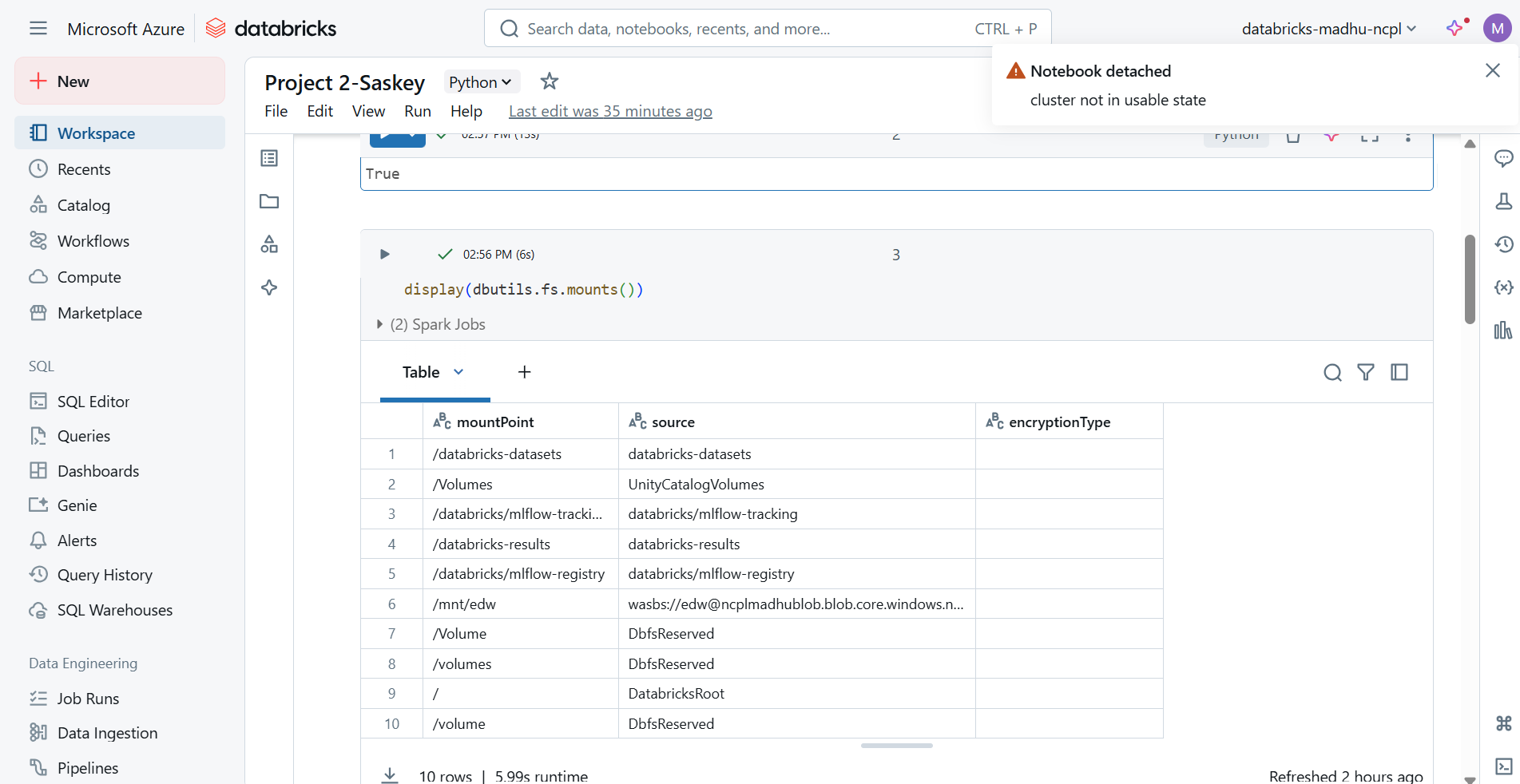
Then again use the mount command

Command 3

Display(dbutils.fs.mount())

The above command will show mounted source in mountpoint.



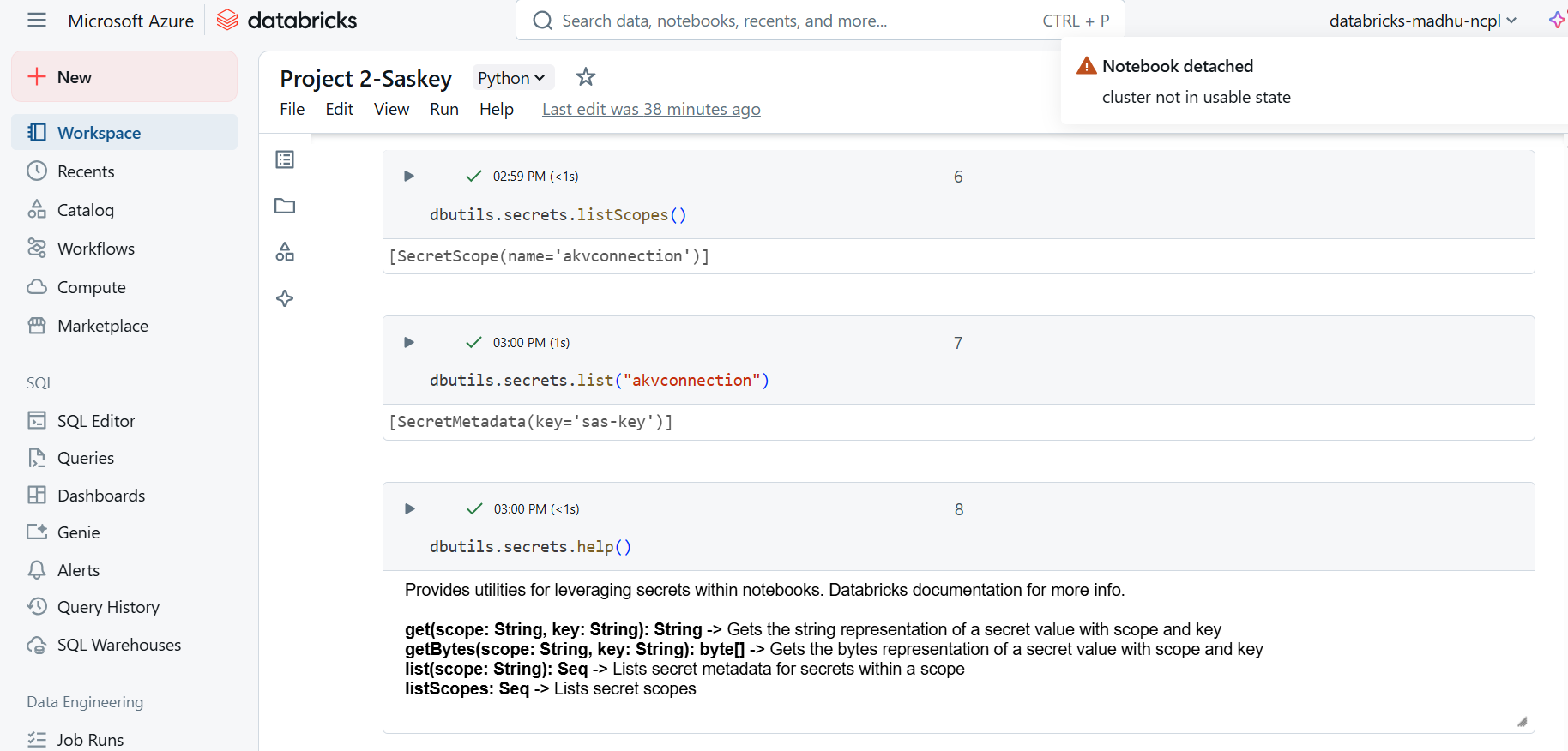


Command 4

dbutils.secrets.help() – it will provide utilities in secrets within the notebooks.

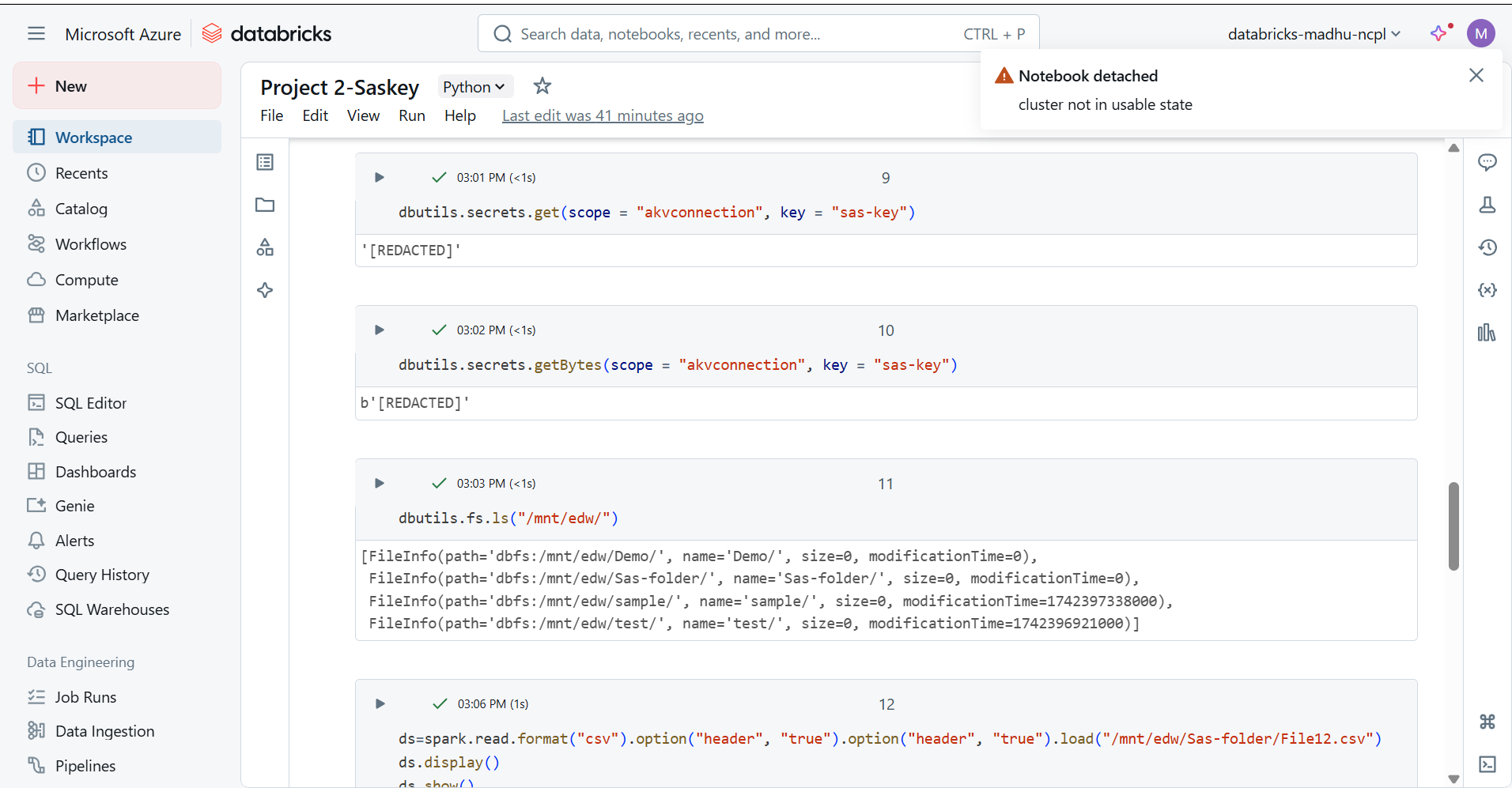
dbutils.secrets.list(“akvconnection”)- Lists the secret metadata for serets within a scope

dbutils.secret.listScopes() -> It will list the secret scopes

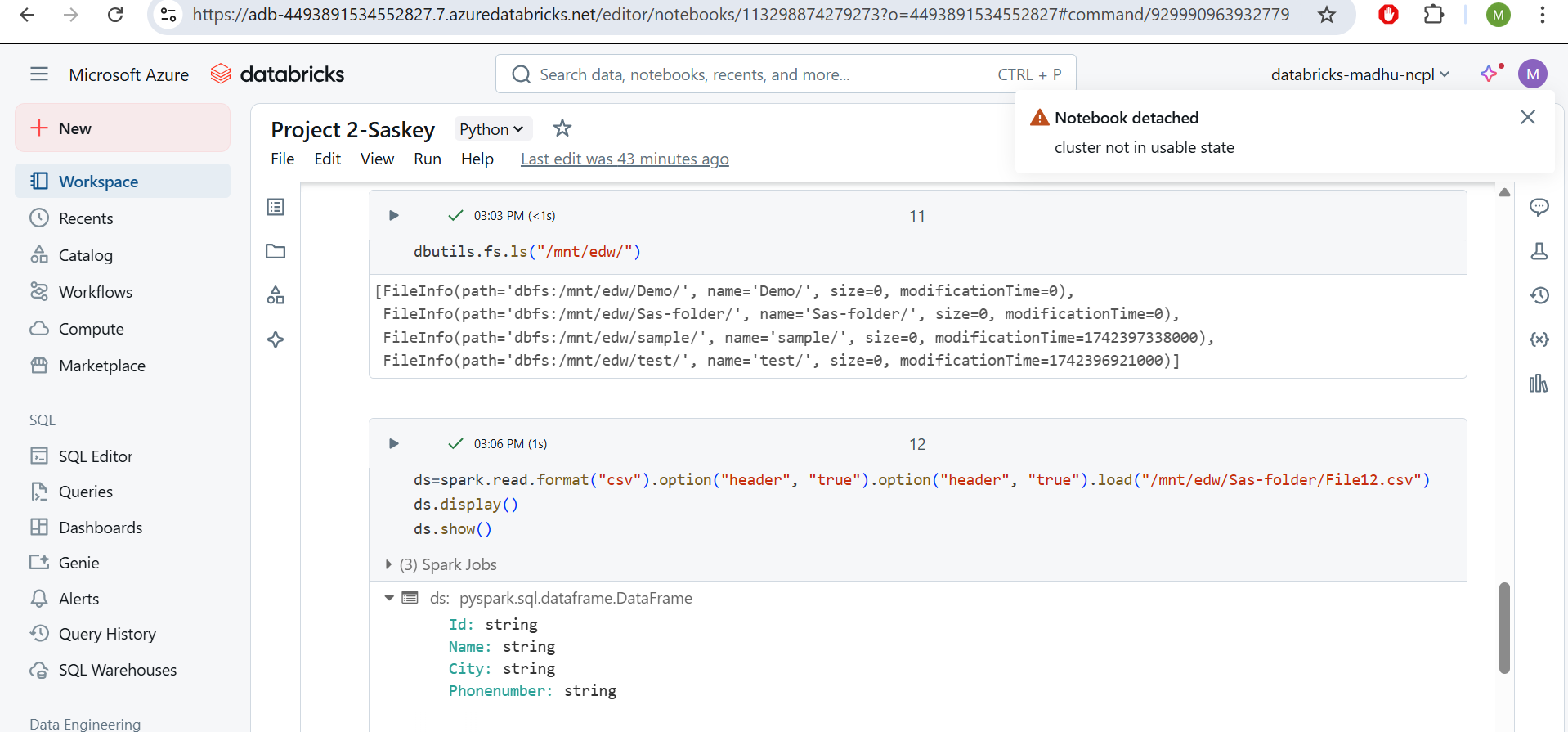


dbutils.secrets.get(scope = "akvconnection", key = "sas-key") – Gets the string representation of secret value with scope and key

dbutils.secrets.getBytes(scope = "akvconnection", key = "sas-key") – Gets the byte representation of secret vault with scope and key



dbutils.fs.ls("/mnt/edw/") – It will display the list of files in edw container



To read a CSV file into a Spark DataFrame and display its contents in Databricks, th below command is used

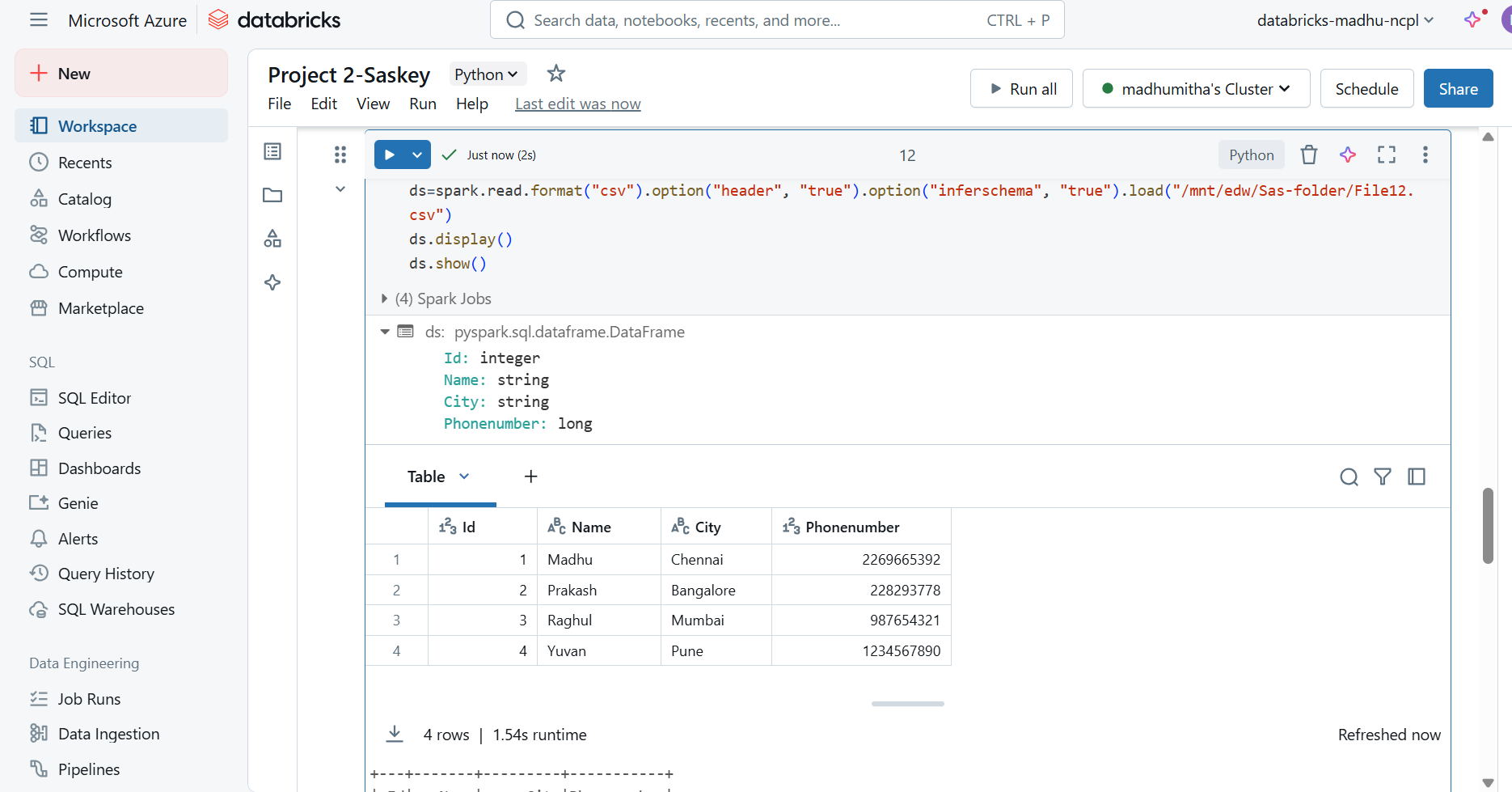
ds=spark.read.format("csv").option("header", "true").option("header", "true").load("/mnt/edw/Sas-folder/File12.csv")

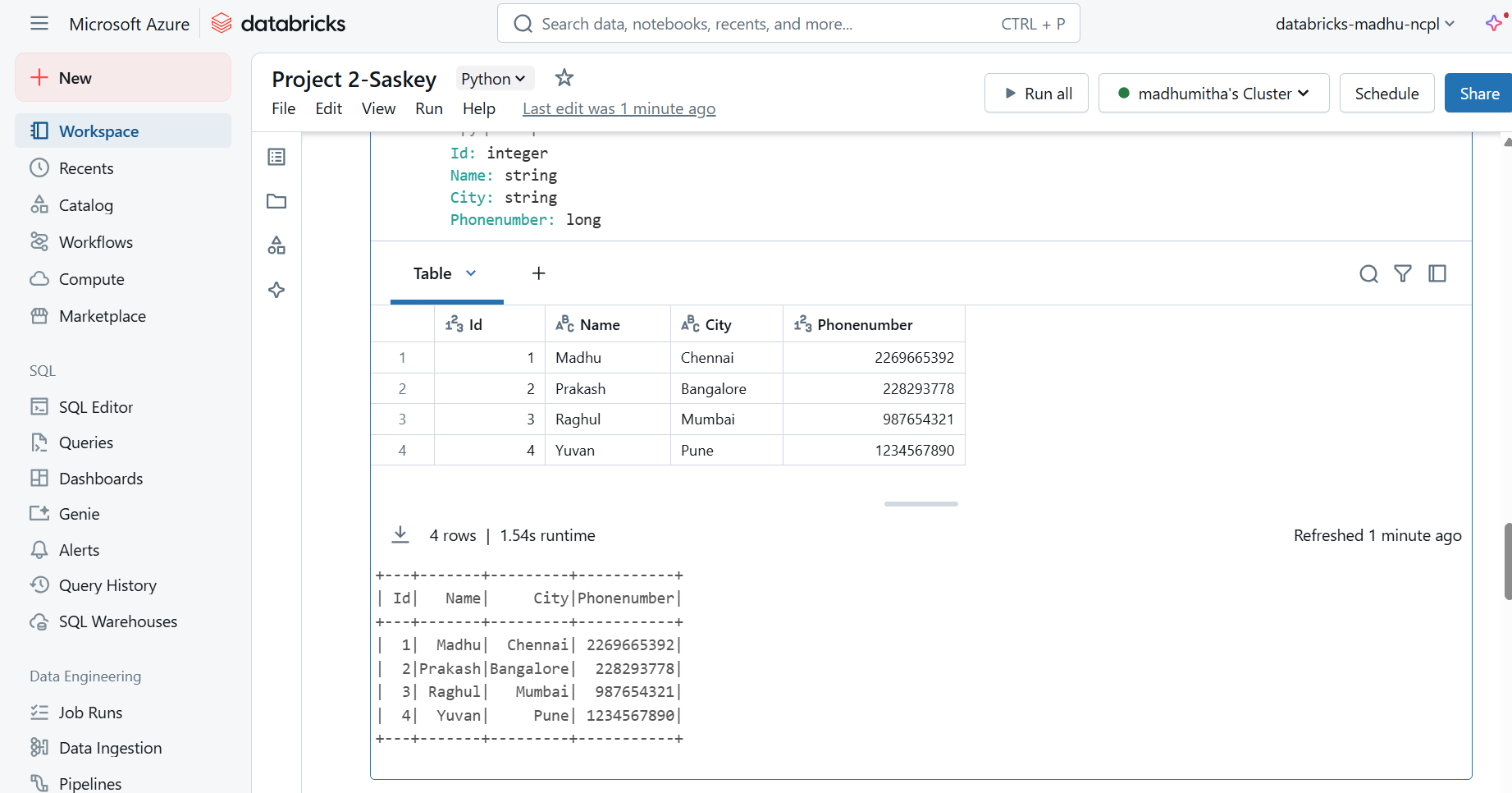
ds.display()

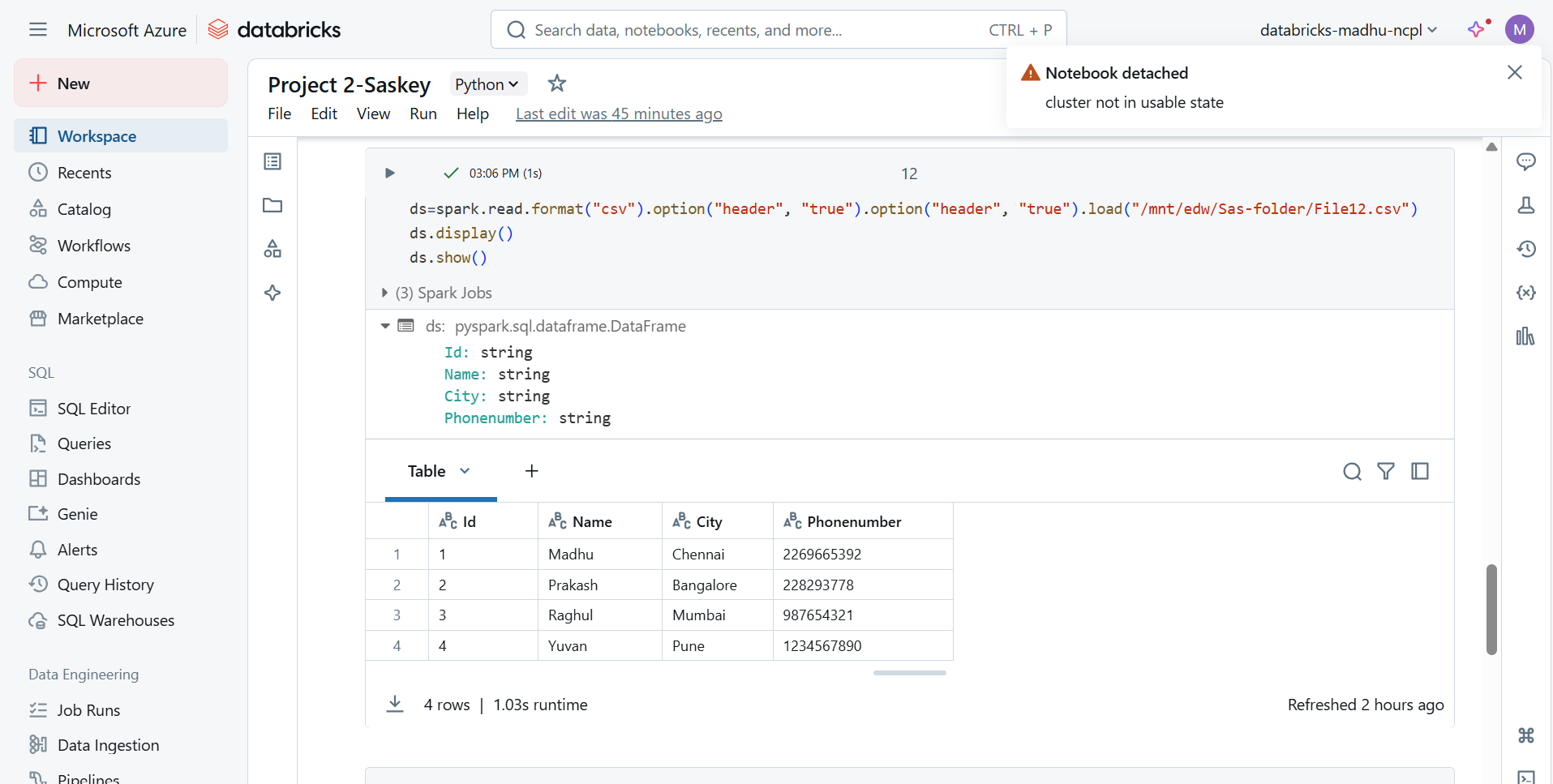
ds.show()

If we use header, all the column datatypes will be generated as strings to avoid this error, we use inferschema

Inferschema- Read the schema and decide the correct datatype







ds.display() – It will show the output in a proper tabular format in UI

ds.show() – It won’t show the output in proper tabular format