Here is a sample code block which you could add to your Databricks notebook to read an existing entity from the guid or qualified name that you could manually get from your Purview Studio account. This sample code is meant to illustrate the capabilities of leveraging the Apache Atlas API and corresponding pyapacheatlas classes to programmatically work with Purview.

import json

import os

# PyApacheAtlas packages

# Connect to Atlas via a Service Principal

from pyapacheatlas.auth import ServicePrincipalAuthentication

from pyapacheatlas.core import PurviewClient, AtlasEntity, AtlasProcess

if \_\_name\_\_ == "\_\_main\_\_":

"""

This sample provides an example of reading an existing entity

through the rest api / pyapacheatlas classes.

You need either the Guid of the entity or the qualified name and type name.

The schema of the response follows the /v2/entity/bulk GET operation

even if you are requesting only one entity by Guid.

https://atlas.apache.org/api/v2/json\_AtlasEntitiesWithExtInfo.html

The response of get\_entity will be a dict that has an "entities" key

that contains a list of the entities you requested.

"""

# Authenticate against your Atlas server

oauth = ServicePrincipalAuthentication(

tenant\_id=os.environ.get("TENANT\_ID", "Enter-tenant-id"),

client\_id=os.environ.get("CLIENT\_ID", "Enter-client-id"),

client\_secret=os.environ.get("CLIENT\_SECRET", "Enter-Client-secret")

)

client = PurviewClient(

account\_name = os.environ.get("PURVIEW\_NAME", "Enter-Purview-name"),

authentication=oauth

)

# When you know the GUID that you want to get

response = client.get\_entity(guid="Enter-guid")

print(json.dumps(response, indent=2))

# When you need to find multiple Guids and they all are the same type

#entities = client.get\_entity(

# qualifiedName=["qualifiedname1", "qualifiedname2", "qualifiedname3"],

# typeName="my\_type"

#)

#for entity in entities.get("entities"):

# print(json.dumps(entity, indent=2))