## Setting up a pipeline to pull user data from Graph API

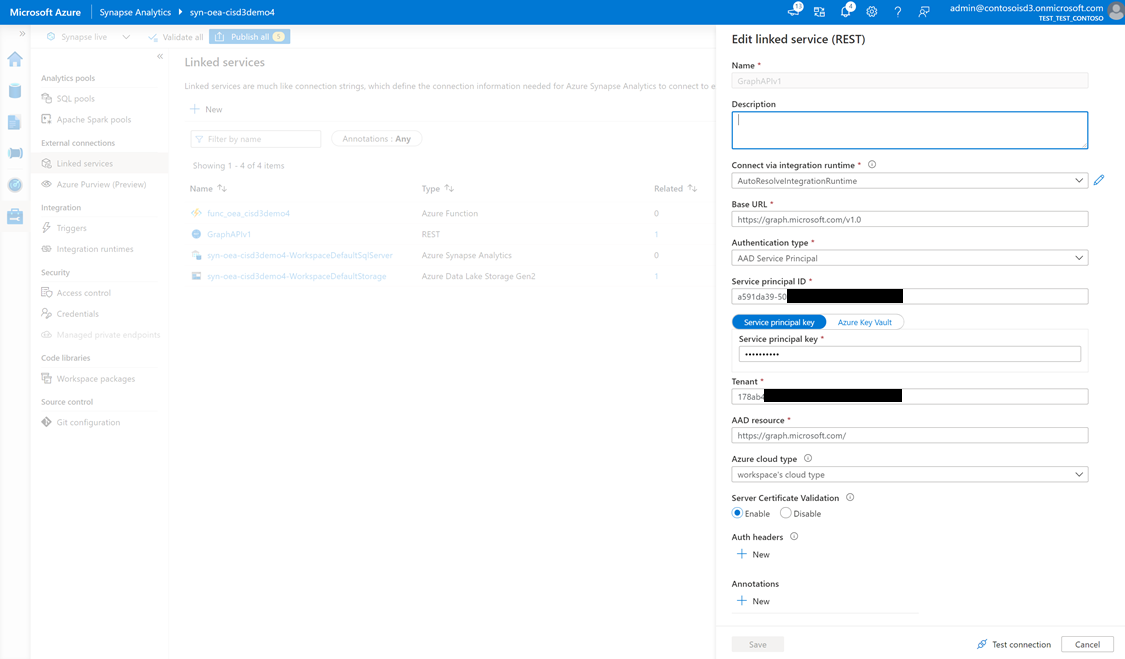
1) create linked service to GraphAPI  
First go to AAD -> App registrations -> New registration, and create a new app registration specifically for accessing the Graph API from Synapse.  
Then click Add Permission, then select Microsoft Graph, then select Application permissions, then choose the User.Read.All permission and click on the Add permissions button.

Now go to certificates & secrets and add a new client secret, and copy the value.

Graphical user interface, text, application, email

Description automatically generated

Now in synapse studio, go to linked services and add a REST service for Graph API and select AAD Service Principal as the Authentication type.



Under service principal key, paste the value of the secret you copied in the previous step.  
Under Service principal ID enter the “Application (client) ID” of the app registration created in the previous step (go to the Overview section of the app registration).

2) create a pipeline, and then create a new REST dataset as the source, referring to the GraphAPIv1 linked service

Graphical user interface, text

Description automatically generated

3) create a sink dataset going to the data lake

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, application

Description automatically generated

You can reduce the data returned by selecting specific attributes in the relativeURL, like this:  
users?$select=givenName,surname,userPrincipalName,id

Use the Graph Explorer to try it out:

Graphical user interface, text, application, email

Description automatically generated

If you run the pipeline and get a failure because of lack of access like this:

{ "errorCode": "2200", "message": "Failure happened on 'Source' side. ErrorCode=RestSourceCallFailed,'Type=Microsoft.DataTransfer.Common.Shared.HybridDeliveryException,Message=The HttpStatusCode 403 indicates failure.\nRequest URL: <https://graph.microsoft.com/v1.0/users?>$top=3\nResponse payload:{\"error\":{\"code\":\"Authorization\_RequestDenied\",\"message\":\"Insufficient privileges to complete the operation.\",\"innerError\":{\"date\":\"2021-05-28T19:31:00\",\"request-id\":\"abc221e5-a4d1-4845-8a5d-c26353b99af3\",\"client-request-id\":\"abc221e5-a4d1-4845-8a5d-c26353b99af3\"}}},Source=Microsoft.DataTransfer.ClientLibrary,'", "failureType": "UserError", "target": "Copy data1", "details": [] }

you'll need to double check the app registration you created and the API permissions assigned to it.

You can then go into Synapse and navigate to your data lake and see the json file. Right click on it and select Load to dataframe, then run the Synapse notebook.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

Now you need to process this user data and write it to stage 2 in a more usable way, and also create a spark db that can be easily queried from Power BI.

Run this notebook to do that:

Graphical user interface, text, application

Description automatically generated

Here’s the script (you have to change the url values to use your storage account name):

%%pyspark

from pyspark.sql.types import StructType, StructField, StringType, IntegerType, DoubleType, ArrayType

from pyspark.sql.functions import explode

user\_schema = StructType(fields=[

    StructField('value', ArrayType(

        StructType([

            StructField('surname', StringType(), False),

            StructField('givenName', StringType(), False),

            StructField('userPrincipalName', StringType(), False),

            StructField('id', StringType(), False)

        ])

    ))

])

df = spark.read.load('abfss://stage1@stoeacisd3demo4.dfs.core.windows.net/AAD/users', format='json', schema=user\_schema)

df = df.select(explode('value').alias('exploded\_values')).select("exploded\_values.\*")

display(df.limit(10))

df.write.format('parquet').mode('overwrite').save('abfss://stage2@stoeacisd3demo4.dfs.core.windows.net/AAD/users')

# Create spark db to allow for access to the data in the data lake via SQL on-demand.

spark.sql('CREATE DATABASE IF NOT EXISTS AAD')

spark.sql("create table if not exists AAD.users using PARQUET location 'abfss://stage2@stoeacisd3demo4.dfs.core.windows.net/AAD/users'")

Now you’re able to query the user data from the spark db:

Graphical user interface, application, table

Description automatically generated