



Graph API Module Tutorial & Setup

How to use the Graph API module in your own Azure Synapse workspace

1. Create a linked service to Graph API

- In your Azure, 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” and select Microsoft Graph -> select Application Permissions -> choose the “User.Read.All” permission, and click on the “Add permissions” button.
- Now go to “certificates & secrets” and add a new client secret -> copy that value.

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes the Microsoft Azure logo and a search bar. The breadcrumb trail indicates the path: Home > test_test_Cortoso > ar-syn-oea-cisdedemo4. The main heading is "ar-syn-oea-cisdedemo4 | API permissions". A left-hand sidebar contains navigation links: Overview, Quickstart, Integration assistant, Manage (Branding, Authentication, Certificates & secrets, Token configuration, API permissions), Expose an API, App roles, Owners, and Roles and administrators | Preview. The "API permissions" link is selected. The main content area shows a message about "Admin consent required" and a section titled "Configured permissions". Below this, there is a table of permissions:

API / Permissions name	Type	Description	Admin consent req...	Status
▼ Microsoft Graph (2)				
User.Read	Delegated	Sign in and read user profile	No	Granted for test_test_Co...
User.Read.All	Application	Read all users' full profiles	Yes	Granted for test_test_Co...

Below the table, there is a link: "To view and manage permissions and user consent, try Enterprise applications."

- Now in your Synapse Studio, go to “linked services” under “Manage”, and add a REST service for Graph API -> select AAD Service Principal as the Authentication type.

The screenshot shows the Microsoft Azure Synapse Studio interface. The top navigation bar includes the Microsoft Azure logo and the Synapse Analytics logo. The breadcrumb trail indicates the path: syn-oea-cisdedemo4. The main heading is "Edit linked service (REST)". The left-hand sidebar contains navigation links: Analytics pools, SQL pools, Apache Spark pools, External connections, Linked services, Azure Purview (Preview), Integration, Triggers, Integration runtimes, Security, Access control, Credentials, Managed private endpoints, Code libraries, Workspace packages, Source control, and Git configuration. The "Linked services" link is selected. The main content area shows a list of linked services with columns: Name, Type, and Related. The "GraphAPIv1" service is selected. The right-hand sidebar shows the configuration for the "GraphAPIv1" service, including fields for Name, Description, Connect via integration runtime, Base URL, Authentication type, Service principal ID, Service principal key, Tenant, AAD resource, Azure cloud type, Server Certificate Validation, Auth headers, and Annotations. The "Authentication type" is set to "AAD Service Principal". The "Service principal ID" is "a591da39-1566-478a-9712-945f10798f6d". The "Service principal key" is "*****". The "Tenant" is "178ab4...". The "AAD resource" is "https://graph.microsoft.com/". The "Azure cloud type" is "workspace's cloud type". The "Server Certificate Validation" is set to "Enable". The "Auth headers" is set to "New". The "Annotations" is set to "New". At the bottom right, there are buttons for "Save", "Test connection", and "Cancel".

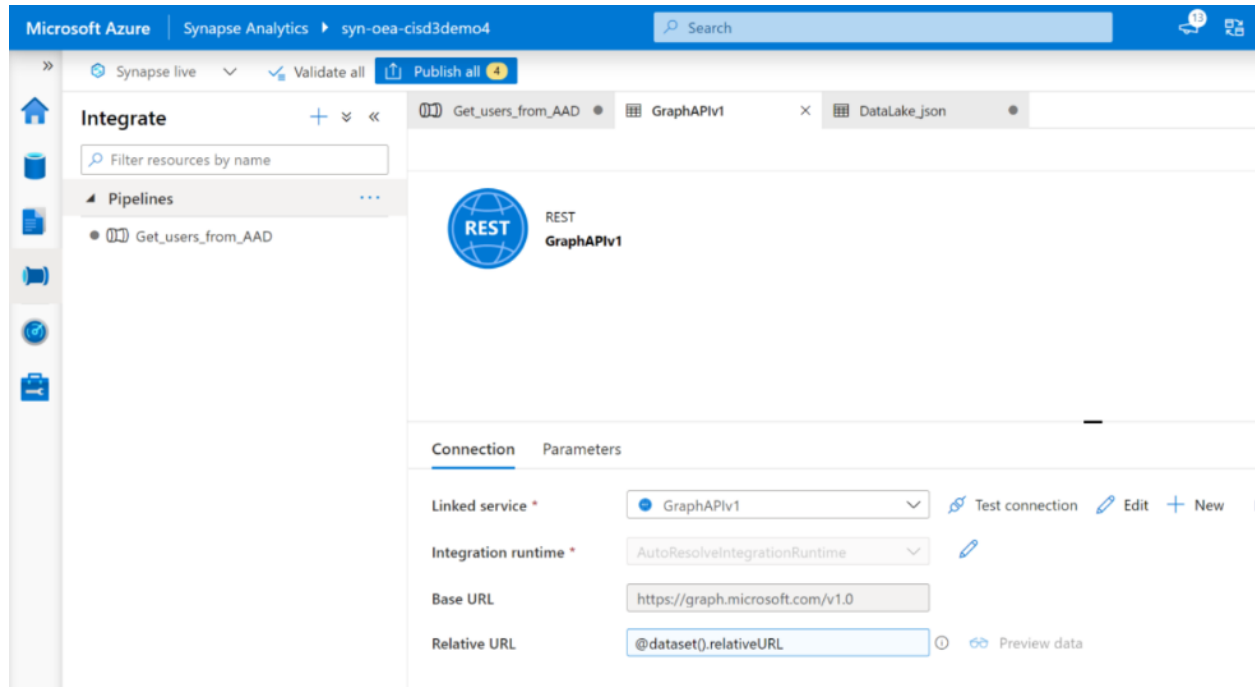
- Under the base URL and AAD resource fields, type in: <https://graph.microsoft.com/>
- Under “Service principal key”, paste the value of the secret you copied previously.
- Under “Service principal ID”, enter the “Application (client) ID” of the app registration created in the previous step (go to the Overview section of the app registration).

2. Notes

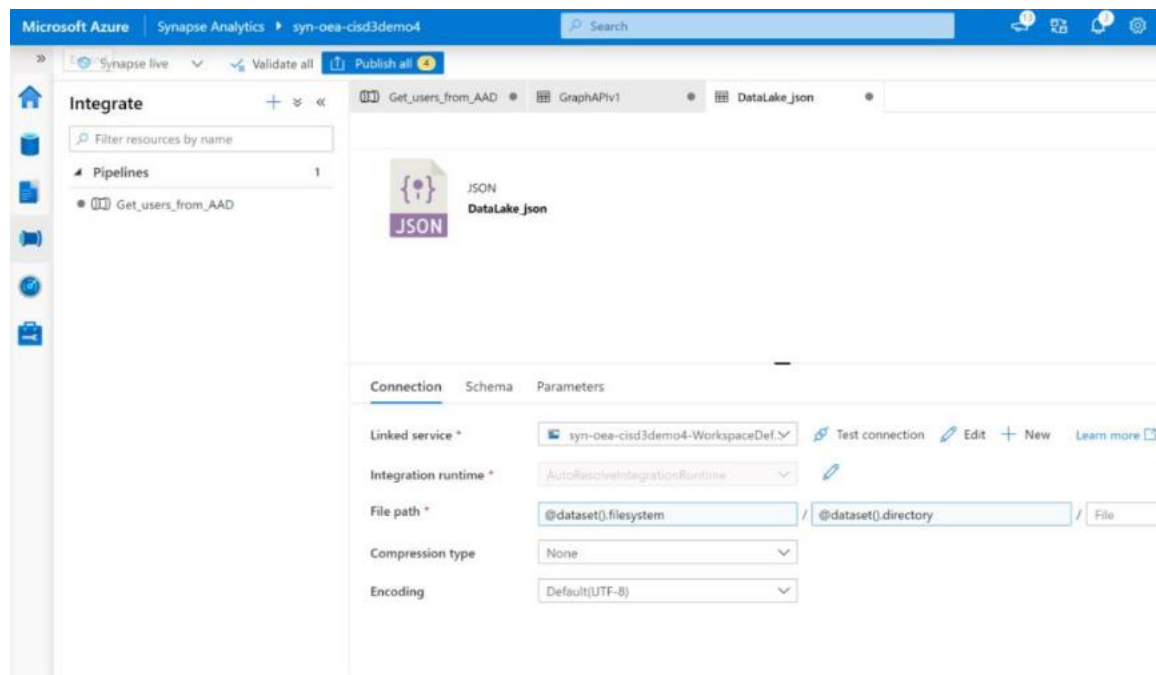
- Make sure that you’re Synapse environment is working within the OEA framework (i.e. the initial setup).
- Before moving forward in this tutorial, you can choose to either proceed with step 3 or step 4 (step 3 explains the process of creating a Graph pipeline from scratch; step 4 uses the sample datasets and pipeline template provided in this module). You should start with step 4 to familiarize yourself with the pipeline, and easily use the sample notebook.
 - If you want to use your own Graph API data, you can start with step 3. To use the notebook provided, you will have to make sure that the pipeline is landing data in stage1np under a folder “GraphAPI” -> the JSON files being landed in this folder should be “users.json”, “m365_app_user_detail.json”, and “teams_activity_user_details.json” in order to run the notebook seamlessly (the queries used for these three files can be found in the GitHub dataset folder of this module).
 - You will also have to sign in to <https://sds.microsoft.com> to use your own data (because of the default hashing of userPrincipalNames); go to the admin center and login -> go to Settings -> Org Settings -> “Services” page. Select “Reports”. Uncheck the statement “Display concealed user, group, and site names in all reports,” and then save your changes. [Activity Reports in the Microsoft 365 admin center - Microsoft 365 admin | Microsoft Docs.](#)

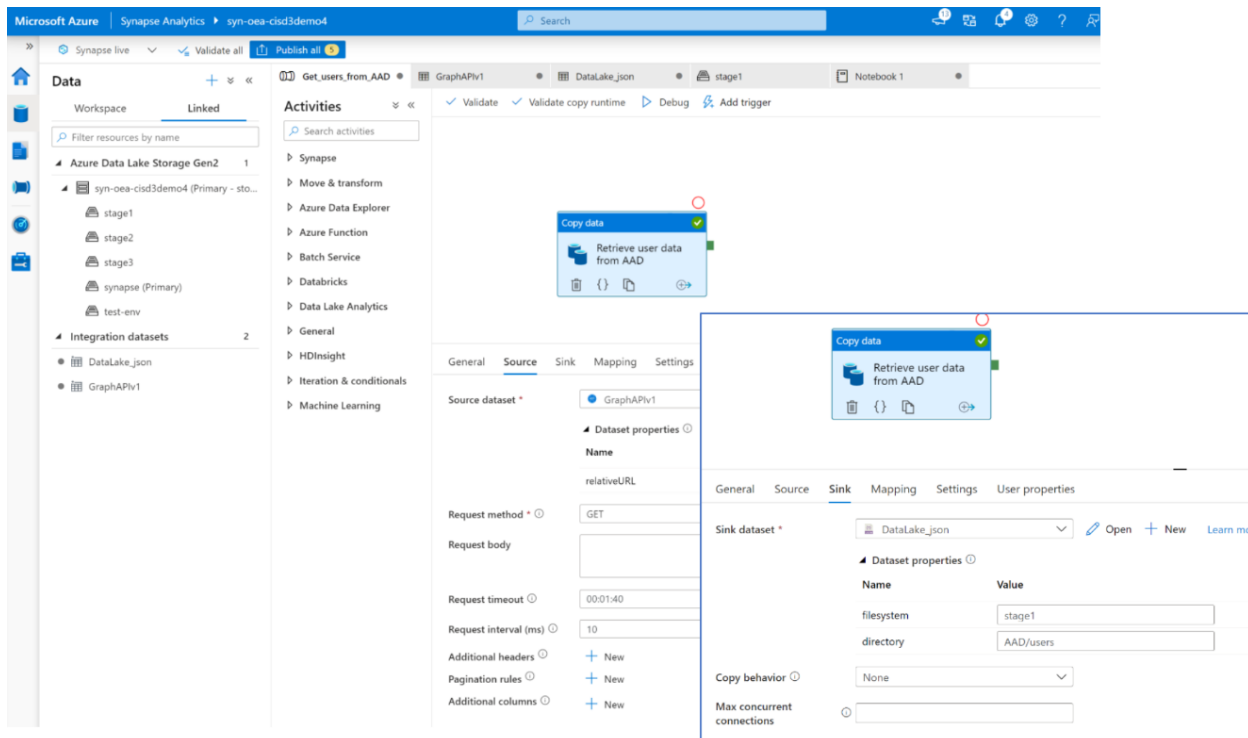
3. Creating the Graph pipeline from scratch within your Synapse environment

- Create a new pipeline under “Integrate” -> create a new REST dataset as the source, referring to the Graph API linked service you created.



- Create a sink dataset going to the data lake as a JSON.

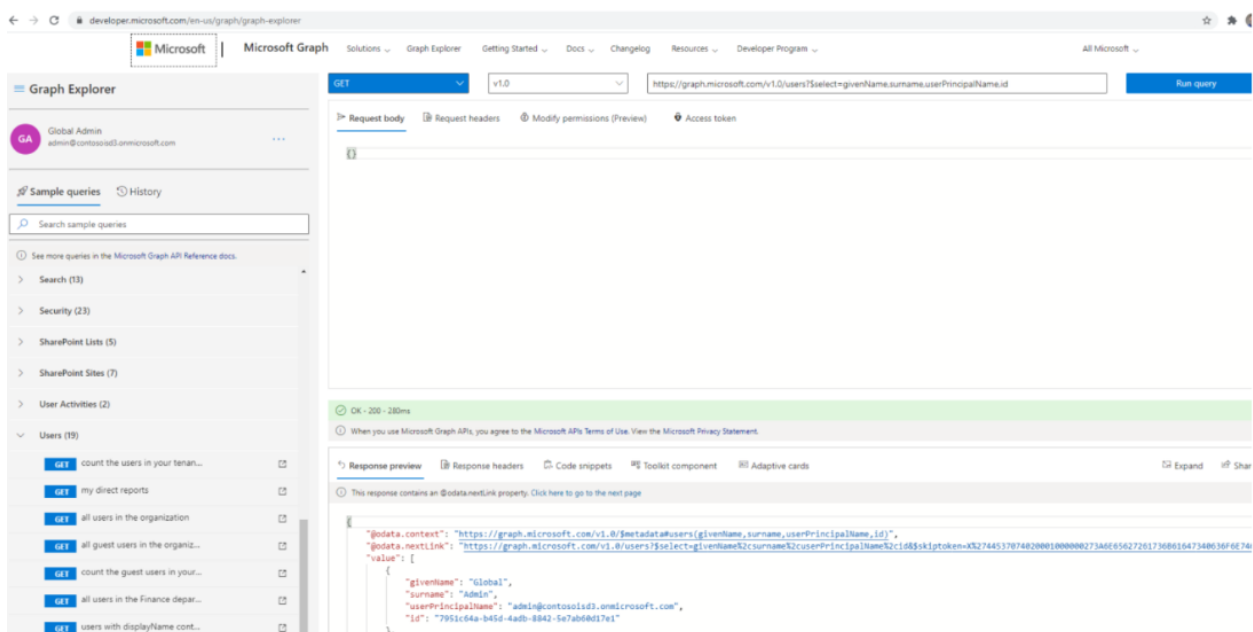




- You can reduce the data returned by selecting specific attributes in the relativeURL, like this: `users?$select=givenName,surname,userPrincipalName,id`

Make sure you put this relativeURL in the “dynamic contents” section in order to retrieve the data from that query.

- Use the Graph Explorer to try it out, and see the data the query is pulling (found at <https://developer.microsoft.com/en-us/graph/graph-explorer>):



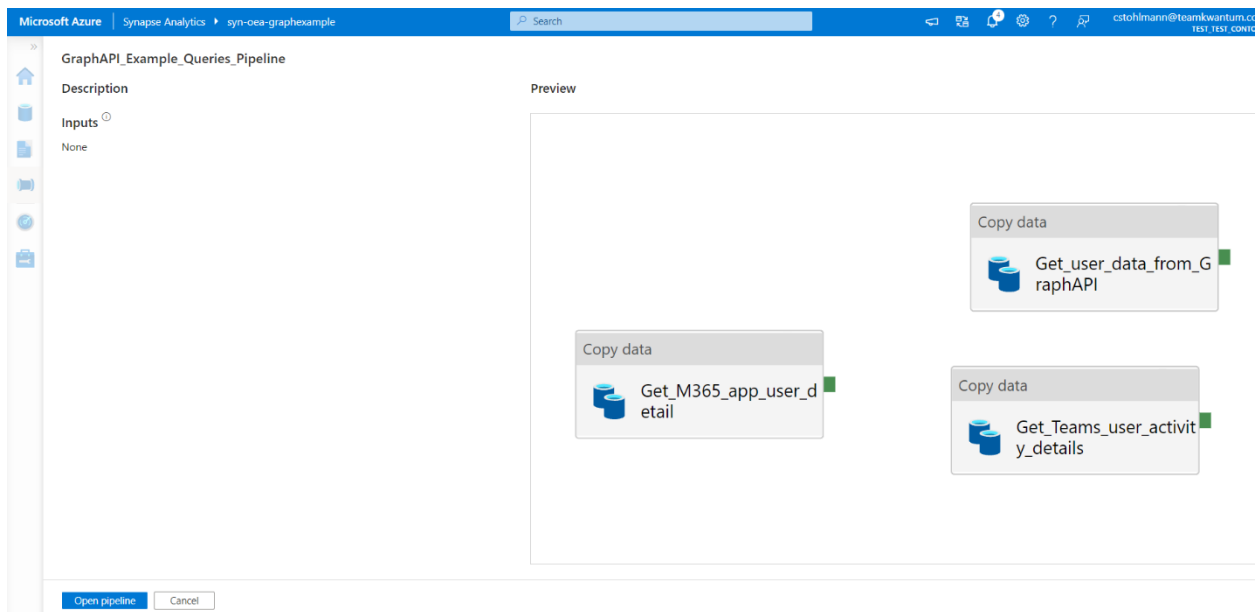
- If you run the pipeline and get a failure because of lack of access like this:

```
{ "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\": [ ] }
```

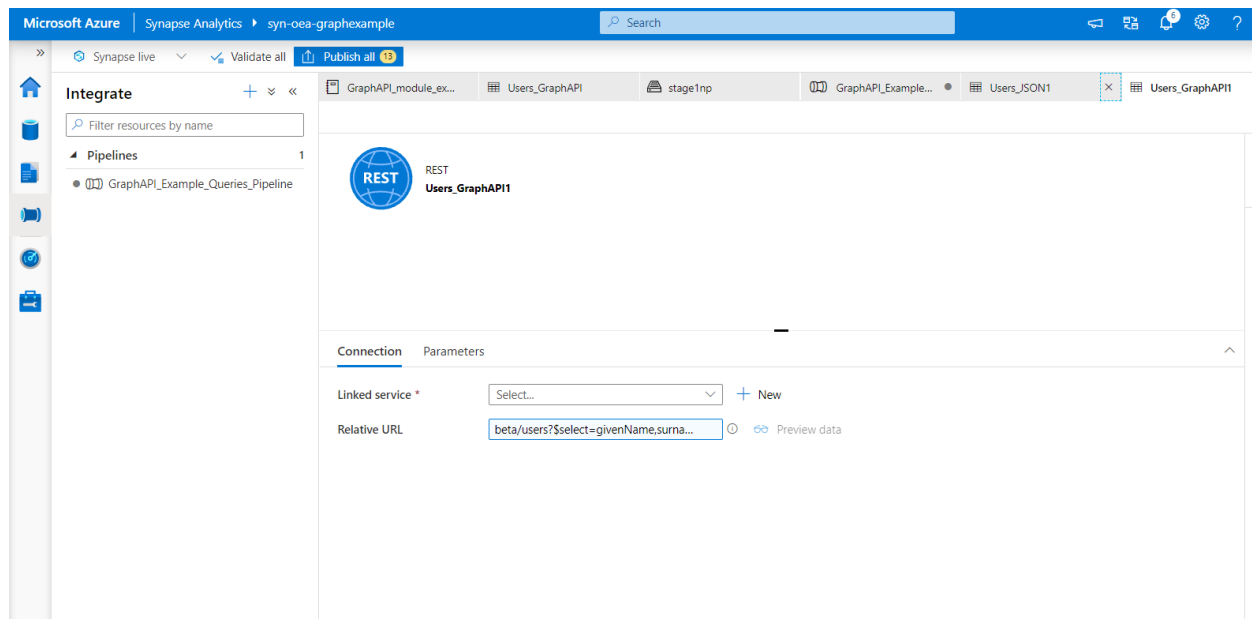
You'll need to double check the app registration you created and the API permissions to it.

4. Using the pipeline template in Synapse for the Graph API

- First download the zip file under “pipelines” in the Graph module, and the sample JSON files under “datasets”.
- Now go to your Synapse environment and navigate to “stage1np” under “data” and “linked” in your Gen2 data lake -> create a new file named “GraphAPI” -> upload the datasets under this folder.
- Next, under “Integrate” and select “Add resource” -> then select “Import from pipeline template” -> select the pipeline zip you downloaded.



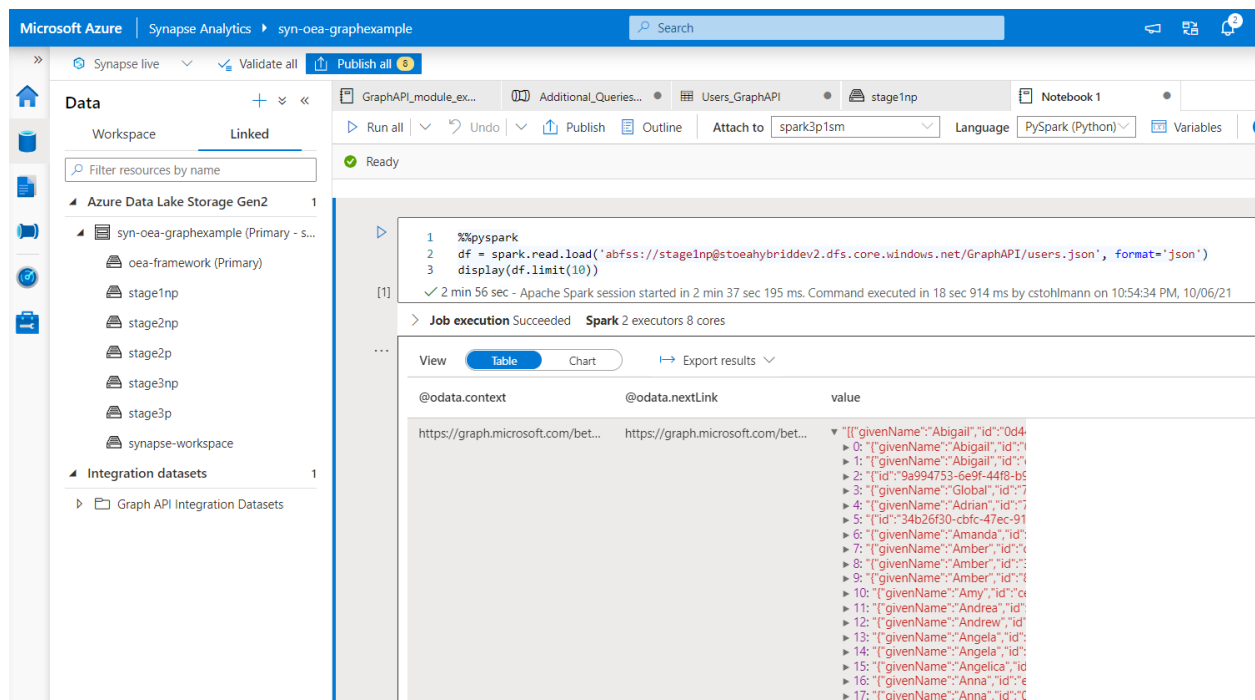
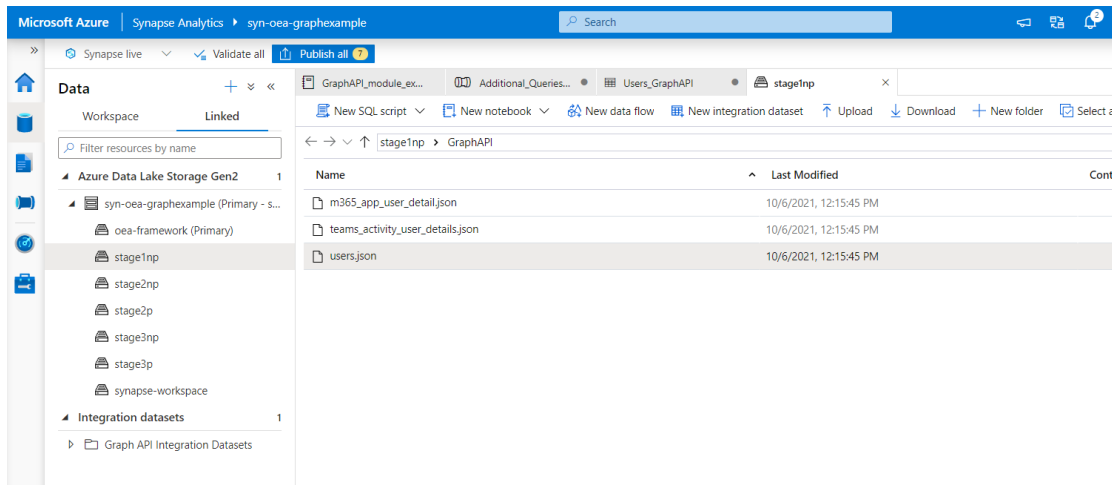
- Now you will just have to connect the REST data source to your linked service: for each of the “copy data” activities, go to “source” -> click “open” -> then select your linked service created in the step 1, pointing to the Graph API.



- After you’ve done this for each activity, you can manually trigger the pipeline to run.

5. View Data in Stage 1np

- 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”.



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

6. Execute the Notebook

- Download the notebook from this Graph module -> import the notebook under “Develop”
- Make sure you change the code in the code block “Provision storage accounts”, so that you’re using your own storage account name.
- Attach to your spark pool and now run this notebook:

The screenshot shows the Microsoft Azure Synapse Analytics interface. The top bar includes the Microsoft Azure logo, Synapse Analytics, and the workspace name 'syn-oea-graphexample'. A search bar is also present. The left sidebar shows the 'Develop' environment with a 'Notebooks' section containing 'GraphAPI_module_example_notebook'. The main area displays the notebook content, which includes a title 'Graph API Module Example Notebook', a description, and a code block for 'Provision storage accounts'. The code block contains Python code for setting up Spark SQL types and functions, and defining storage accounts. The notebook is attached to a Spark pool named 'spark3p1sm' and the language is set to 'PySpark (Python)'. The status bar at the bottom indicates that the Apache Spark session started in 2 min 23 sec 906 ms and the command executed in 156 ms on 10/06/21.

Microsoft Azure | Synapse Analytics | syn-oea-graphexample

Develop

Filter resources by name

Notebooks

GraphAPI_module_example_notebook

Graph API Module Example Notebook

This notebook creates 3 tables (users, m365_app_user_detail and teams_activity_user_details) into a new Spark database called graphapi.

+ Code + Markdown

Provision storage accounts

The storage account variable has to be changed to the name of the storage account associated with your Azure resource group.

```
1 from pyspark.sql.types import StructType, StructField, StringType, IntegerType, DoubleType, ArrayType
2 from pyspark.sql.functions import *
3 from pyspark.sql.window import Window
4
5
6 # data lake and container information
7 storage_account = 'stoeahybriddev2'
8 use_test_env = False
9
10 if use_test_env:
11     stage1np = 'abfss://test-env@' + storage_account + '.dfs.core.windows.net/stage1np'
12     stage2np = 'abfss://test-env@' + storage_account + '.dfs.core.windows.net/stage2np'
13 else:
14     stage1np = 'abfss://stage1np@' + storage_account + '.dfs.core.windows.net'
15     stage2np = 'abfss://stage2np@' + storage_account + '.dfs.core.windows.net'
```

[1] ✓ - Apache Spark session started in 2 min 23 sec 906 ms. Command executed in 156 ms on 10/06/21

7. View Data in Stage 2np

- Now you're able to query the data from the newly created spark database.

The screenshot displays the Microsoft Azure Synapse Analytics interface. The top navigation bar shows 'Microsoft Azure | Synapse Analytics | syn-oea-graphexample'. The left sidebar contains a 'Data' section with a 'Workspace' and 'Linked' tab. Under 'Databases', there are two databases: 'default (Spark)' and 'graphapi (Spark)'. The 'graphapi (Spark)' database is expanded, showing a 'Tables' folder with three tables: 'm365_app_user_detail', 'teams_activity_user_details', and 'users'. The 'users' table is selected. The main pane shows a SQL query in a script editor:

```
1 SELECT TOP (100) [surname]
2 , [givenName]
3 , [userPrincipalName]
4 , [id]
5 FROM [graphapi].[dbo].[users]
```

The query is executed, and the results are displayed in a table view. The table has four columns: 'surname', 'givenName', 'userPrincipalName', and 'id'. The results show 10 rows of user data.

surname	givenName	userPrincipalName	id
Long	Abigail	abigaillong3@contosoisd3.onmi...	0d444980-3123-46f4-85d3-6f49e4619ea5
Smith	Abigail	abigailsmith35@contosoisd3.on...	c19234e7-a816-41af-9524-4be4092fc559
Admin	Global	admin@contosoisd3.onmicrosoft...	7951c64a-b45d-4adb-8842-5e7ab60d17e1
Jordan	Adrian	adrianjordan15@contosoisd3.on...	7a2ca37f-d1f3-4be8-8abe-8711d4eea8d6
(NULL)	(NULL)	agundapaneni_microsoft.com#E...	34b26f30-cbfc-47ec-9131-27fef4433705
Long	Amanda	amandalong47@contosoisd3.on...	154f2b2c-9e73-4ecc-917e-06c0e440ea15
Berger	Amber	amberberger79@contosoisd3.on...	dbc10b91-8d2c-4cf2-ac08-985ad3b4d6b4
Buchanan	Amber	amberbuchanan5@contosoisd3...	3f901837-788d-407b-b6e5-3dd9018011fe