ADF v2.0 (Azure Data factory) samples

This document contains the following ADF v2.0 samples:

* [ADF v2.0 Read Blob storage and upload to SQL Server](#Blob_storage_to_sql_server_upload)
* [ADF v2.0 with Dataset and Pipeline Parameters](#Dataset_and_pipeline_parameters)
* [ADF v2.0 Call Databricks with Parameters](#Call_Databricks_with_Parameters)
* [ADF v2.0 Pipeline Security/Authentication](#Pipeline_Security_and_Auhtentication)
* [ADF v2.0 Event based Pipeline Trigger](#Event_based_Pipeline_Trigger)
* [ADF v2.0 Pull data from on-prem database and load it into Azure SQL database](#Upload_onprem_data_to_Azure_SQL_Database)

ADF v2.0 Read Blob storage and upload to SQL Server

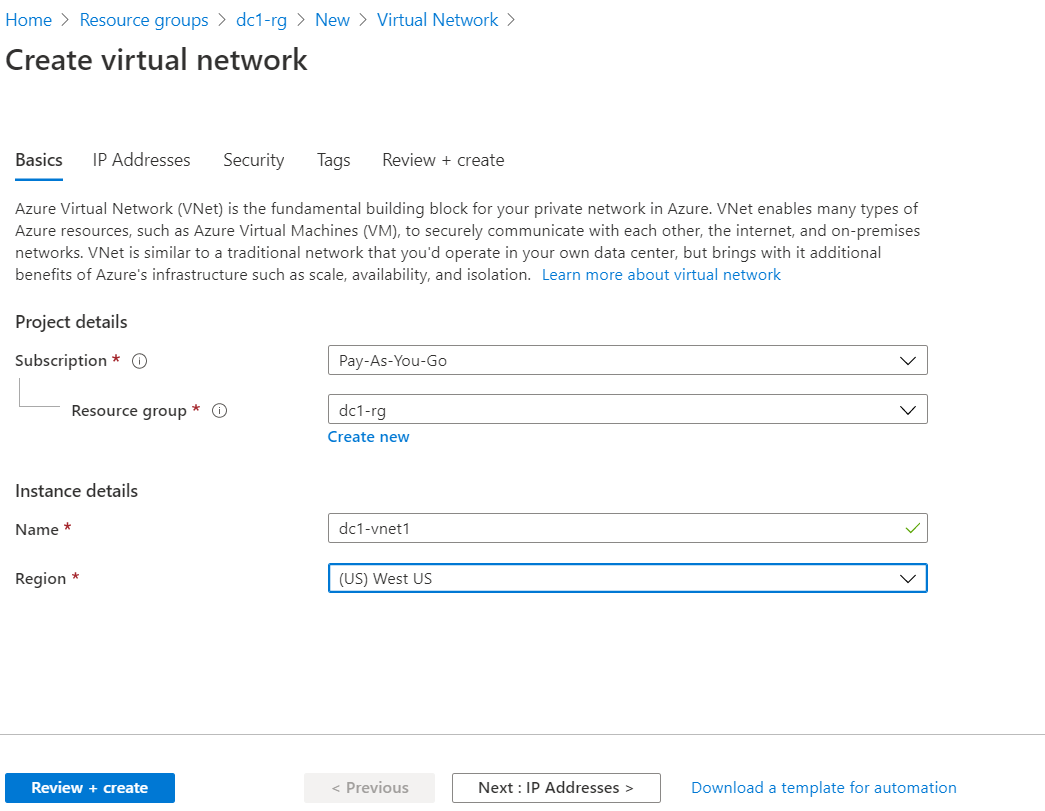
Friday, July 24, 2020

3:22 PM

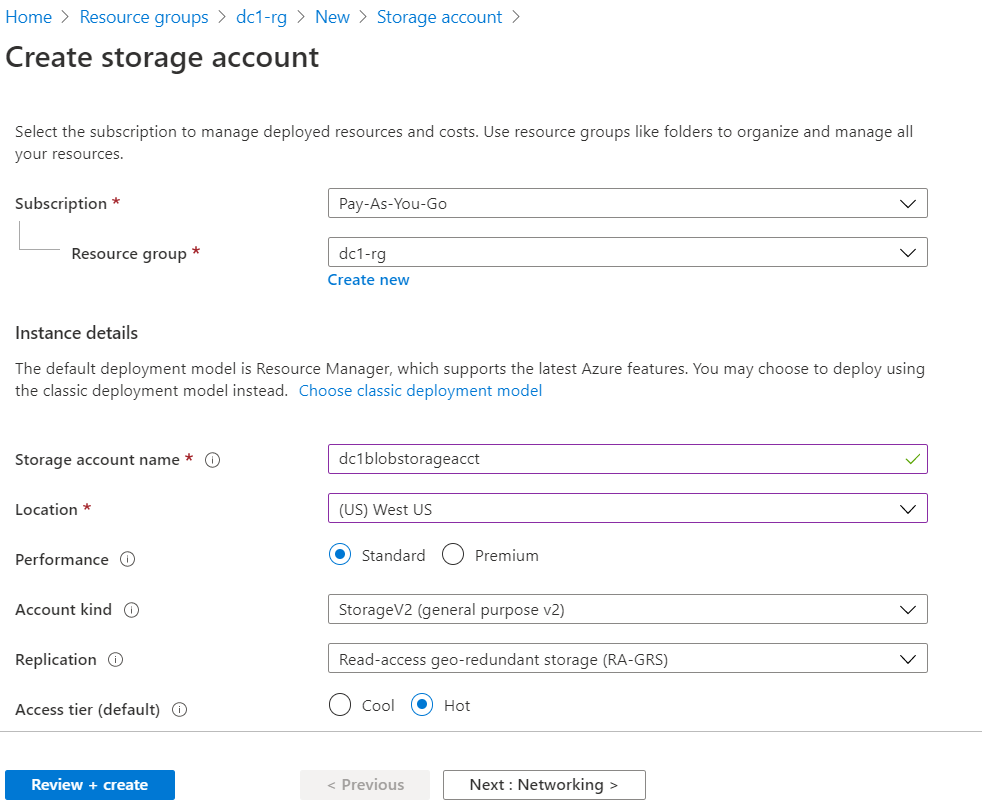
Create Azure Data Factory v2



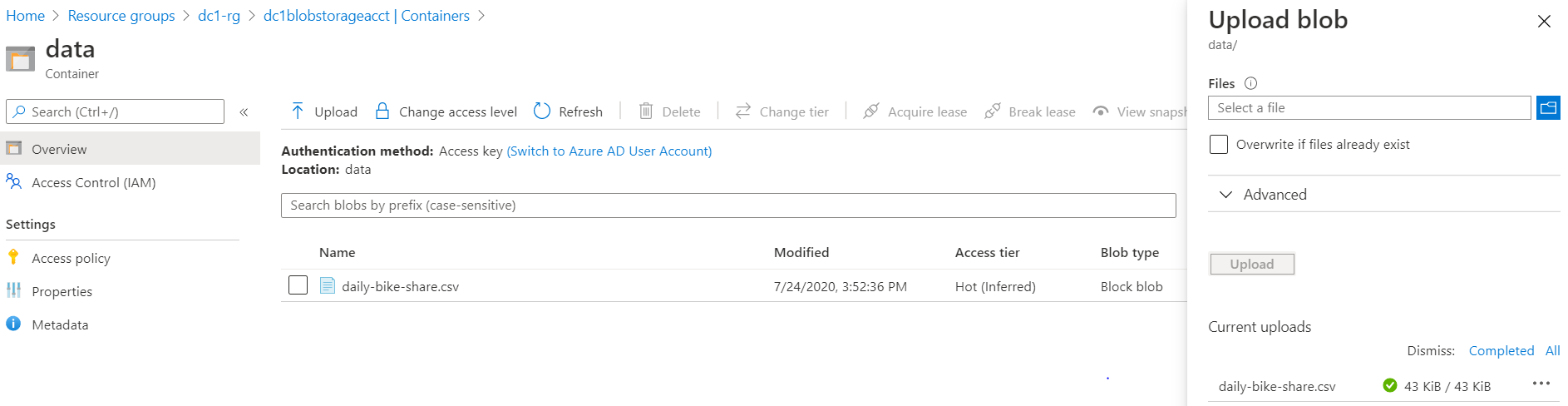
Create a virtual network



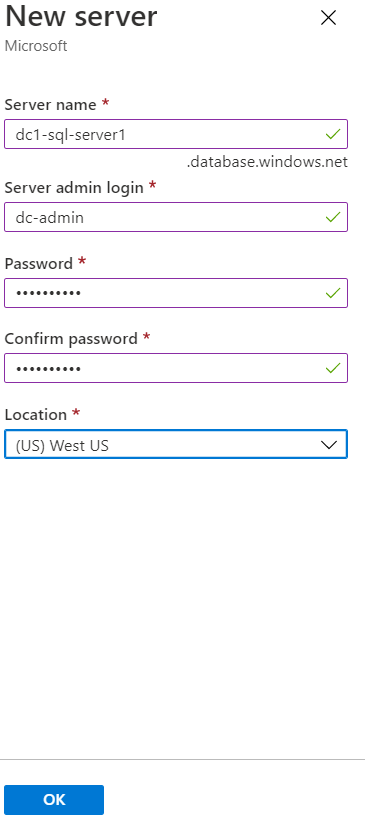
Create a blob storage account



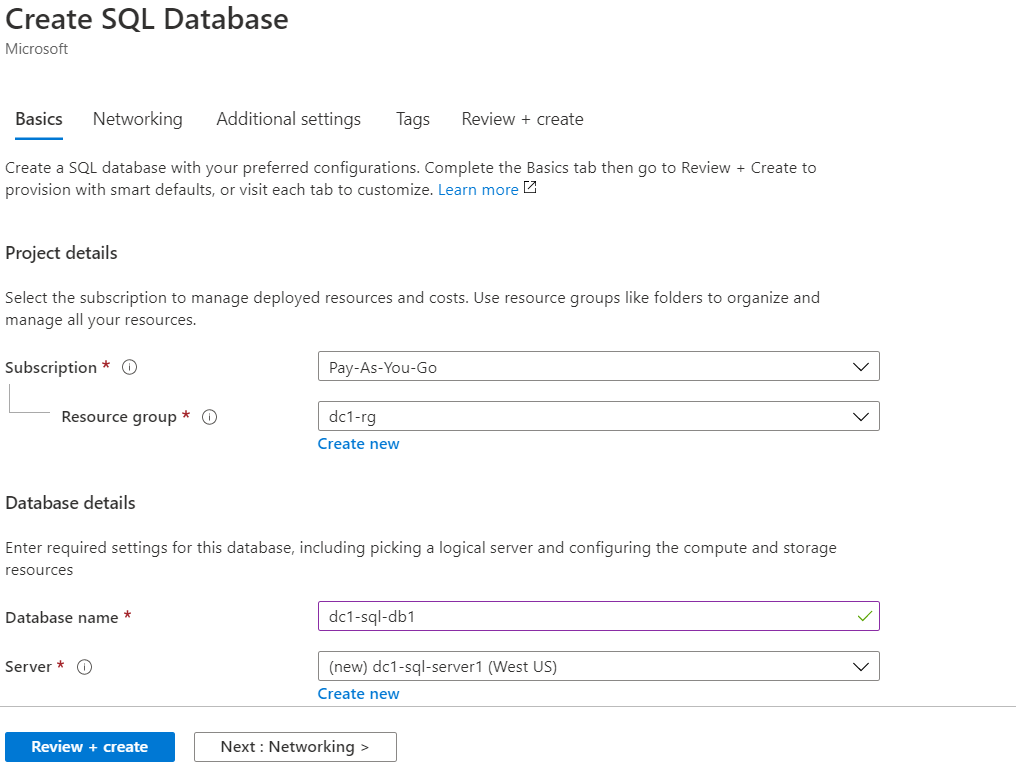
Load a CSV file into the storage



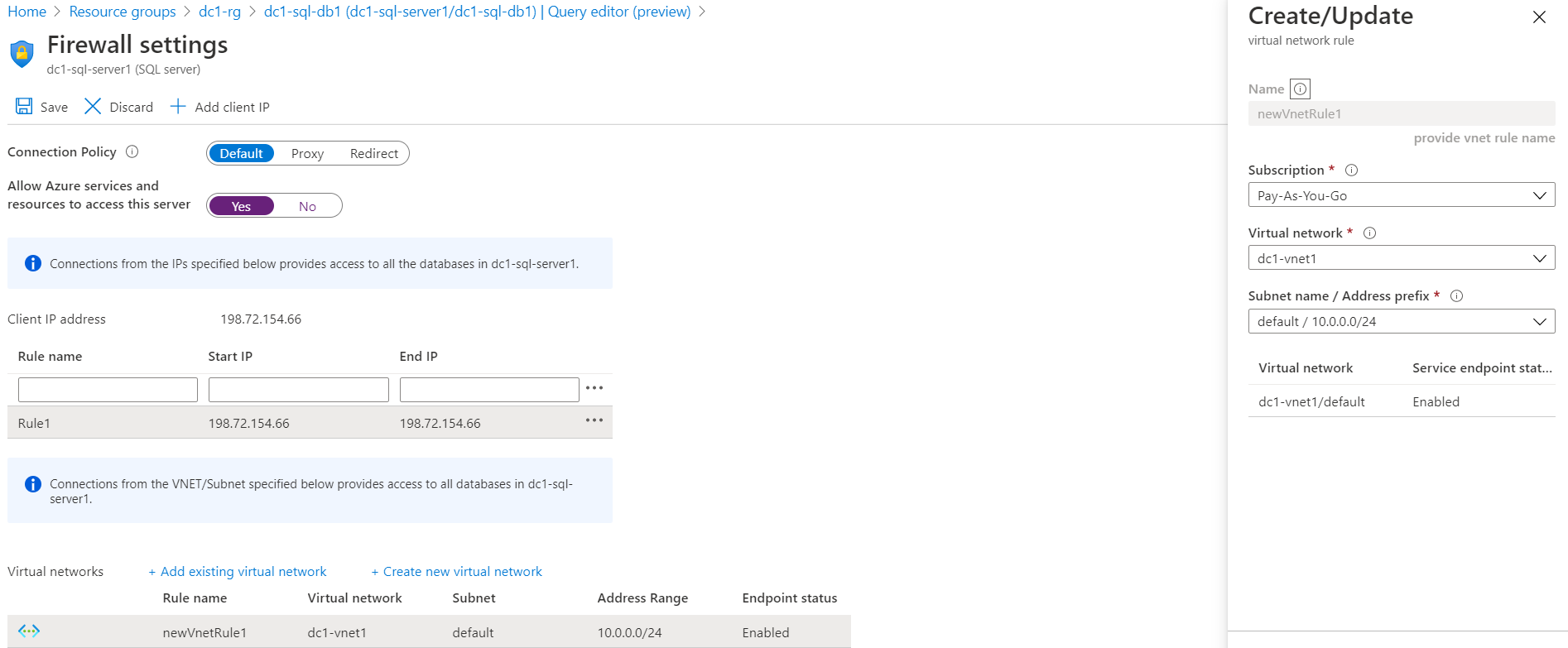
Create a SQL Server instance



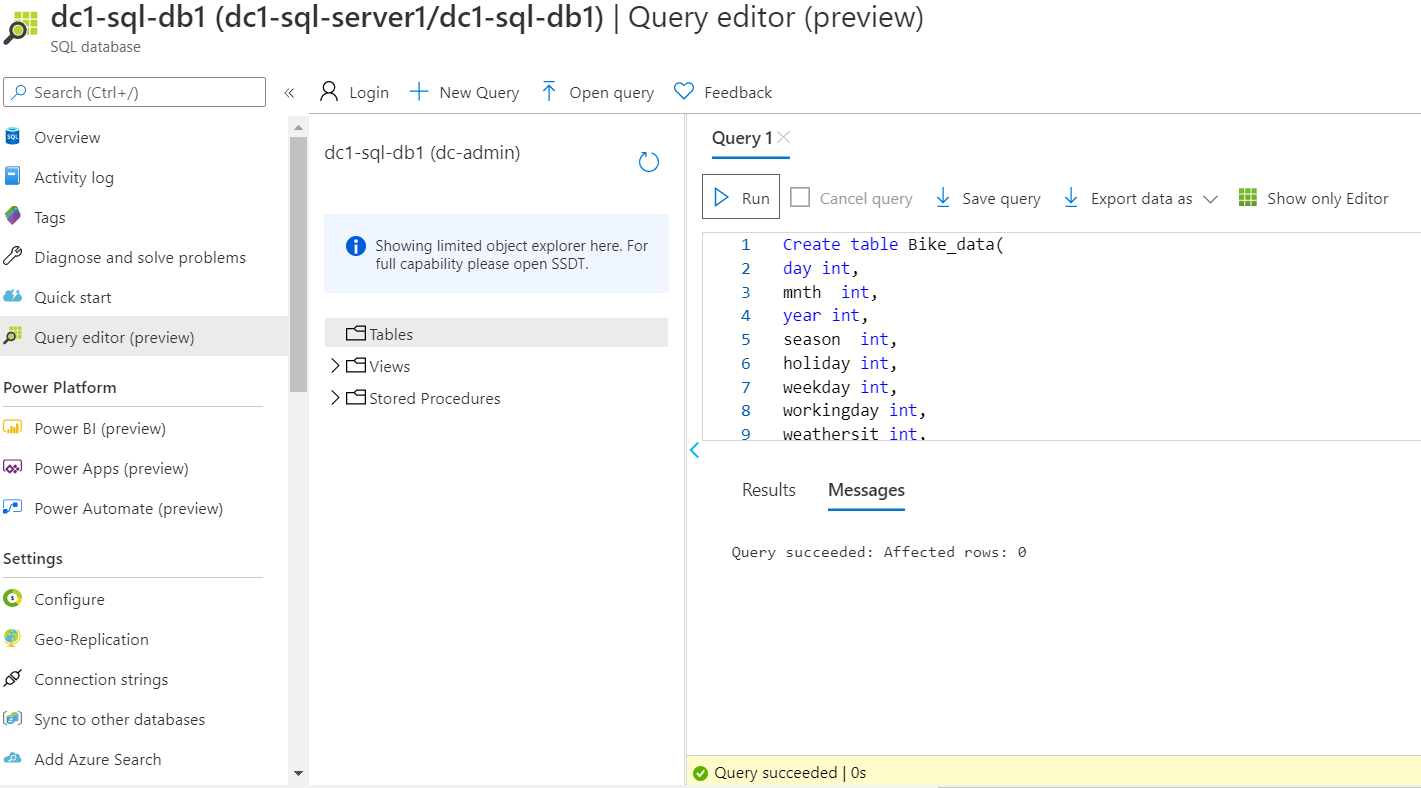
Create an Azure SQL database



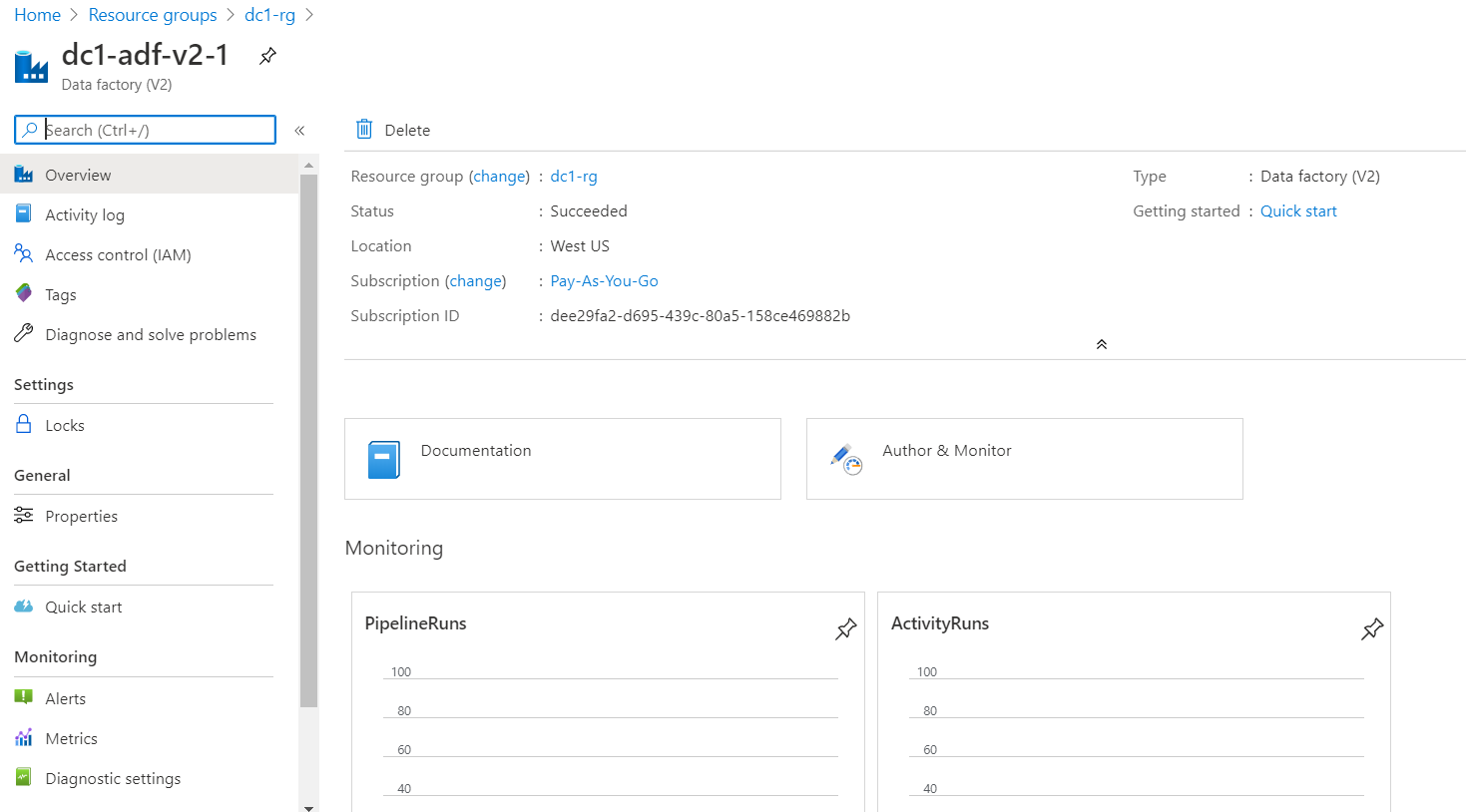
Allow SQL Interactive query to access database



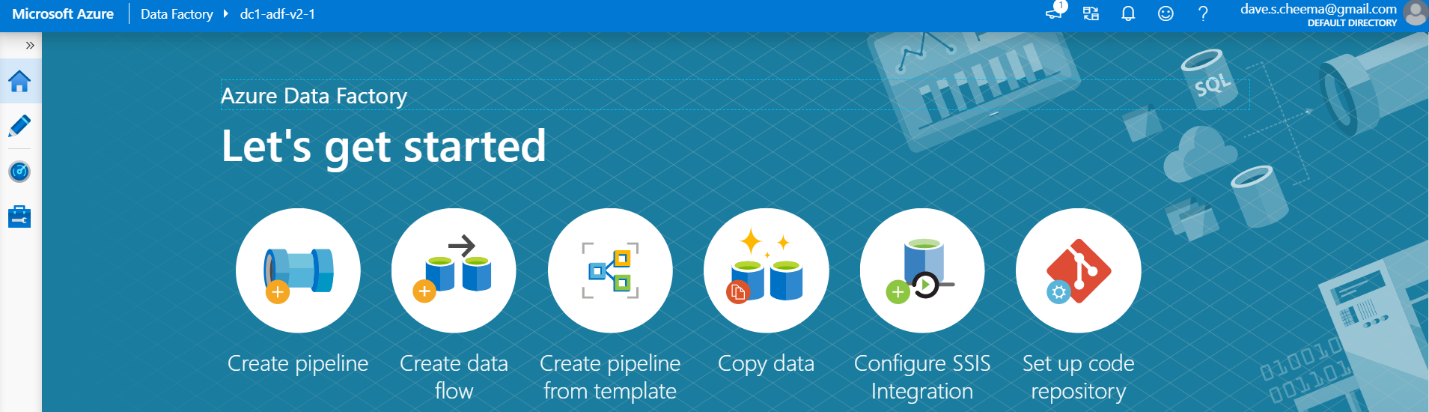
Using Interactive query, create a table where the blob data is to be loaded



Open up Azure Data Factory instance and click on Author & Monitor



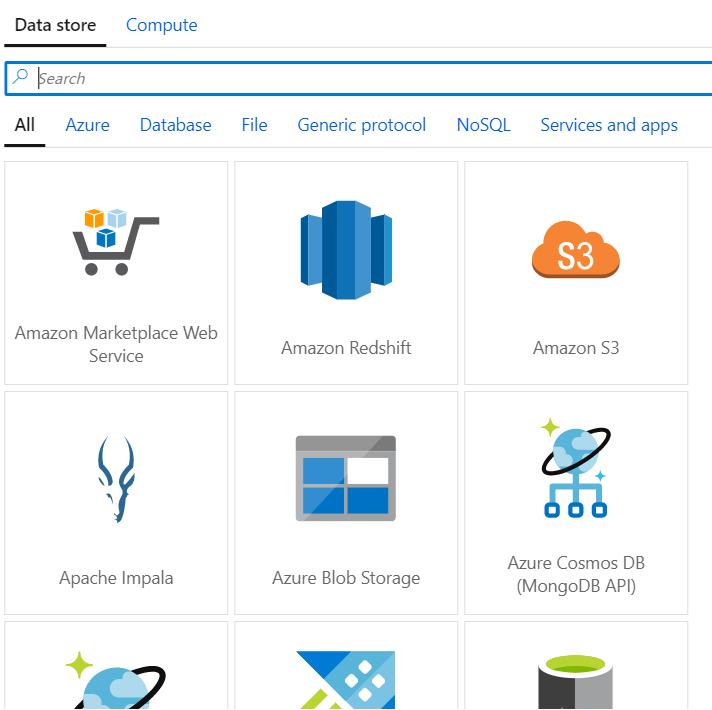
You will come to the screen below:



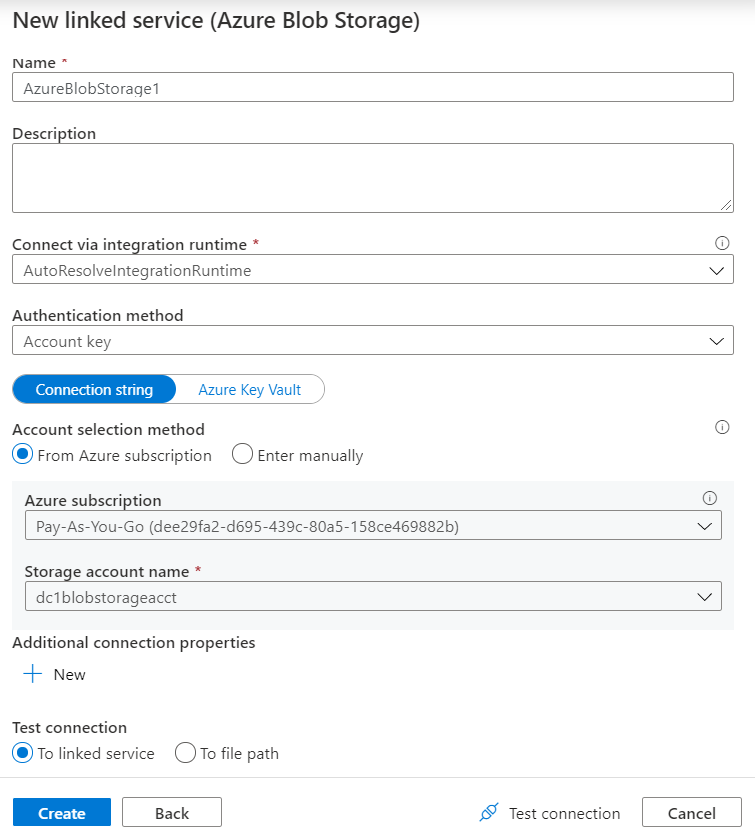
Create Link services for the source and target destination

Link Service to source blob container

Click on New (+)



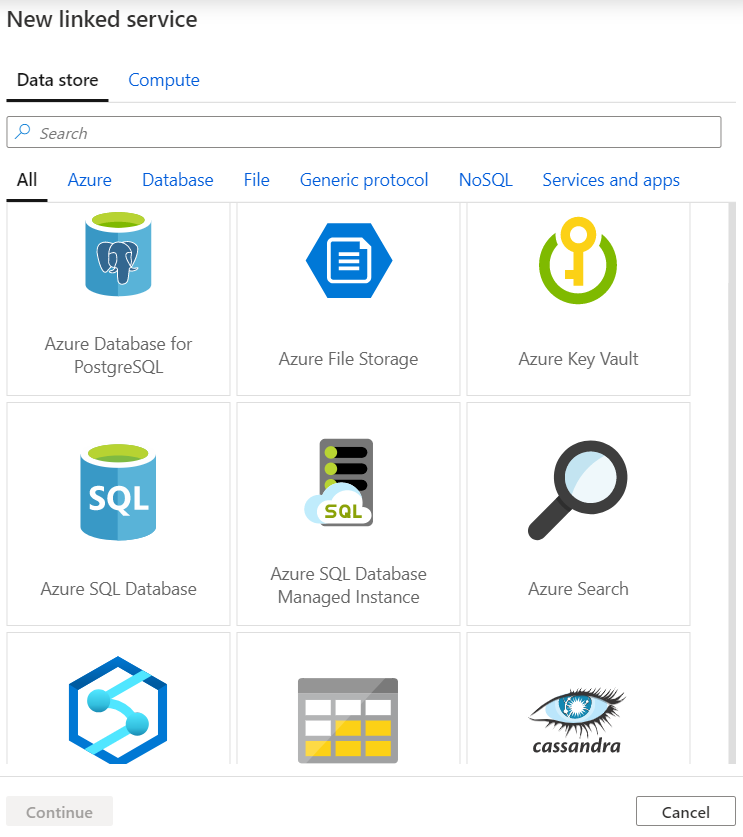
Click on Azure Blob Storage icon and click on Continue button at the bottom (not shown)



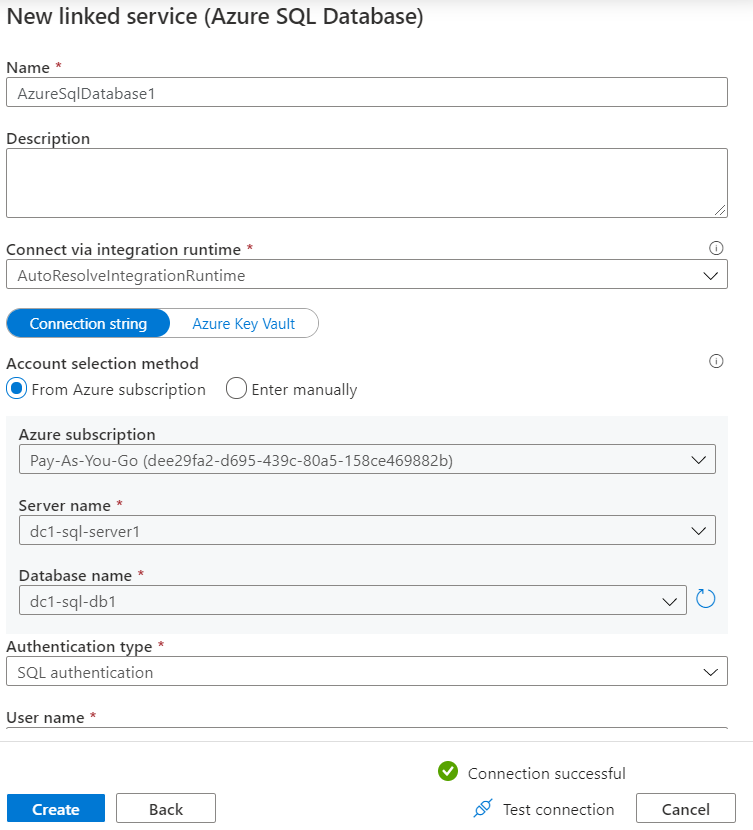
Fill out the details and click on test connection. You should get Successful connection test message

Click Create button

Click on New (+) to create a second Link service for the database and table



This time click on Azure SQL Database icon and click on Continue button at the bottom

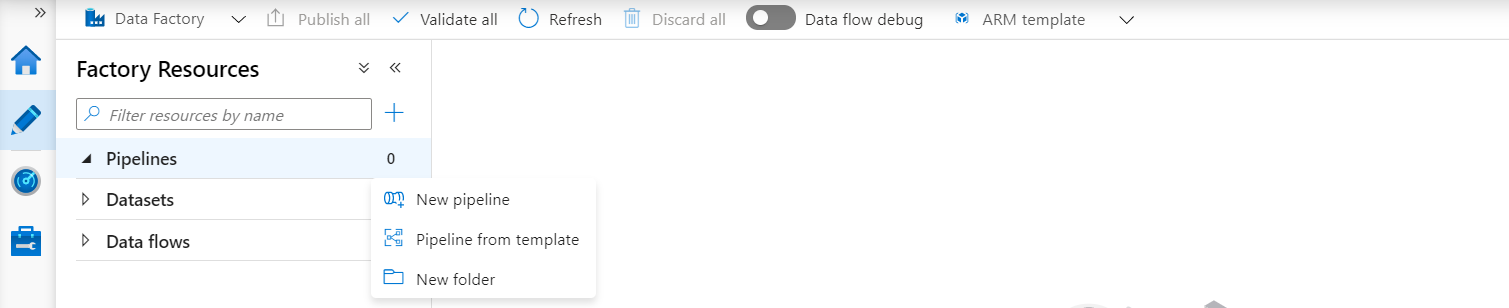


Fill in all the details and click on Test connection link.

Upon successful connection message, click on Create

You should see two Linked services in your Linked services screen

Now create a new Pipeline



Click on New pipeline button

Under Factory Resources, click on Context menu on Pipelines

Under Activities, expand Move & transform, drag Copy data option the workspace

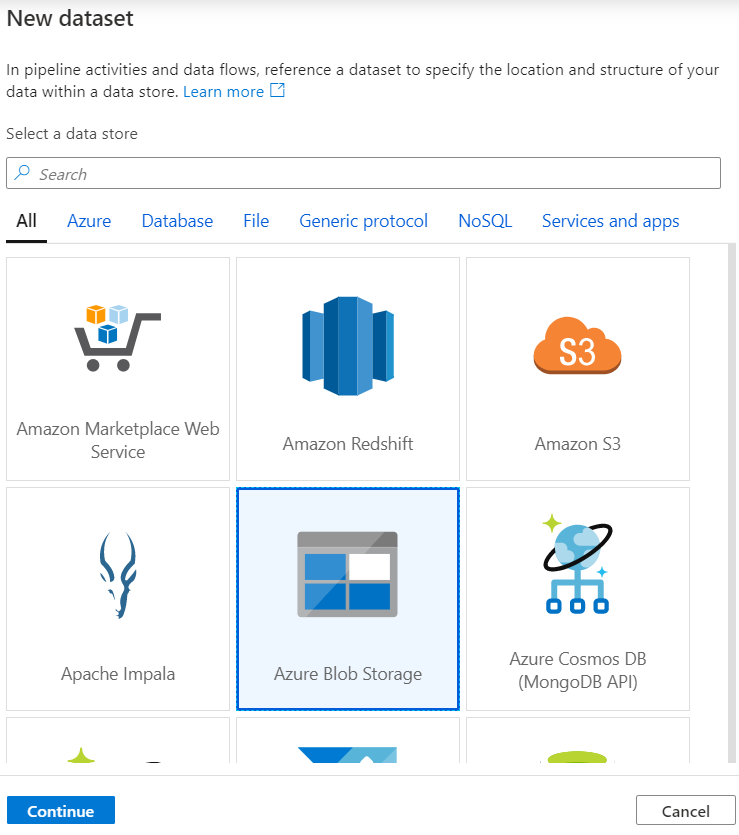
At the bottom half of the workspace, define/configure Source and Sink options

Source (blob storage) and Sink will be the table into which the data will be moved and reside

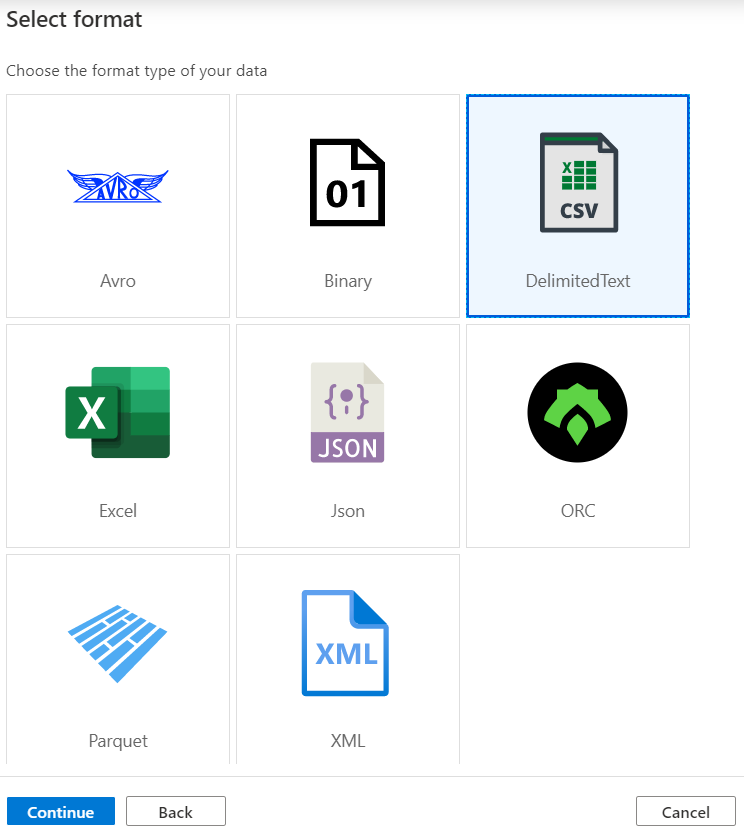
Source

Click on Source

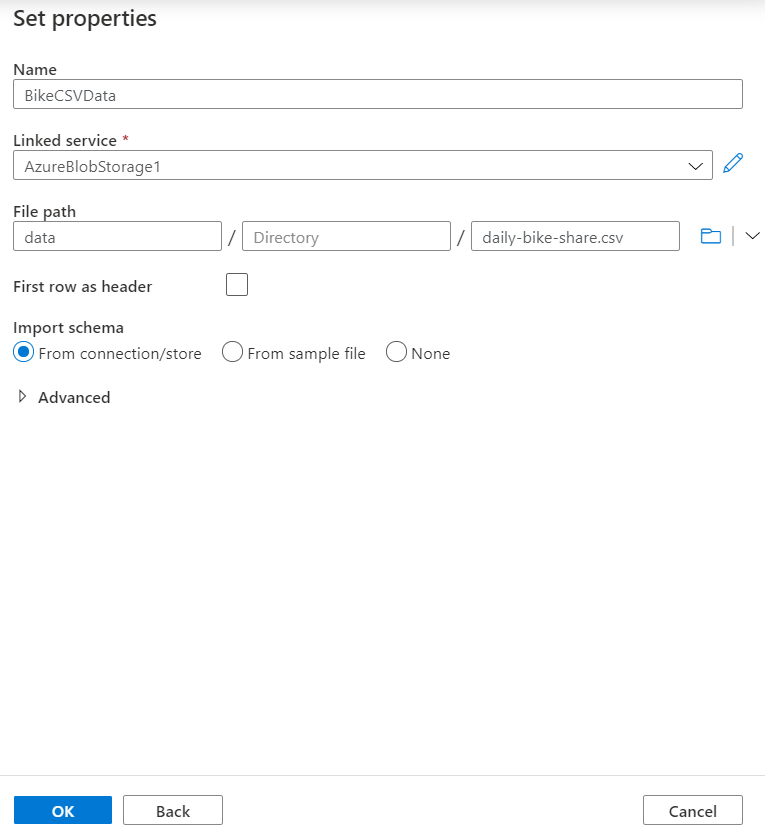
Clink on New (+)



Click on Azure Blob Storage icon and click on Continue button at the bottom



On the Select Format screen, select CSV and click on Continue



On the Set Properties screen

Pick the Linked service you created earlier

On the File path, click on the folder icon and select the appropriate file

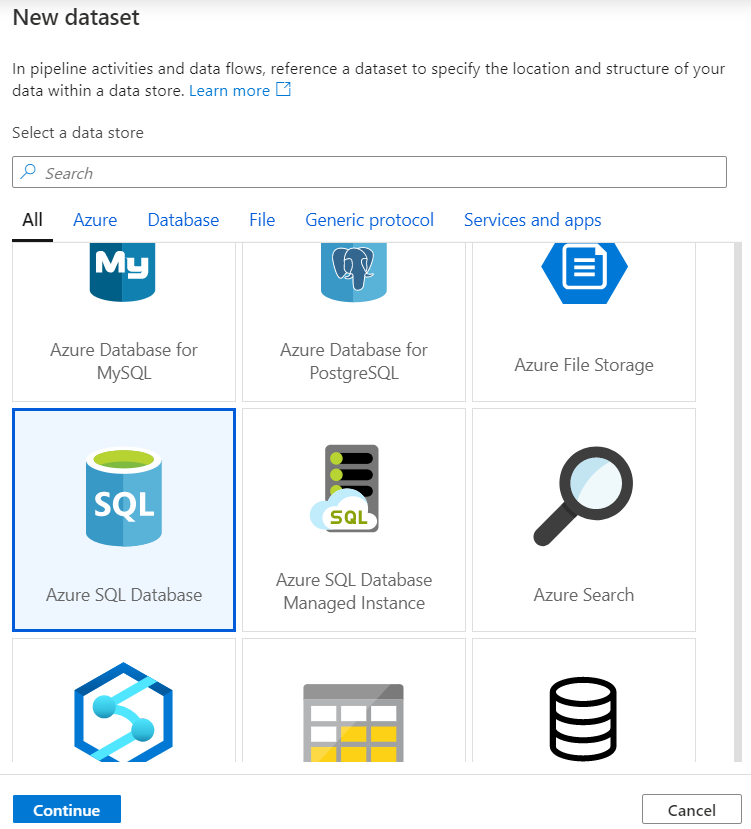
Click OK

The Source path is set. You can Preview data, if you like

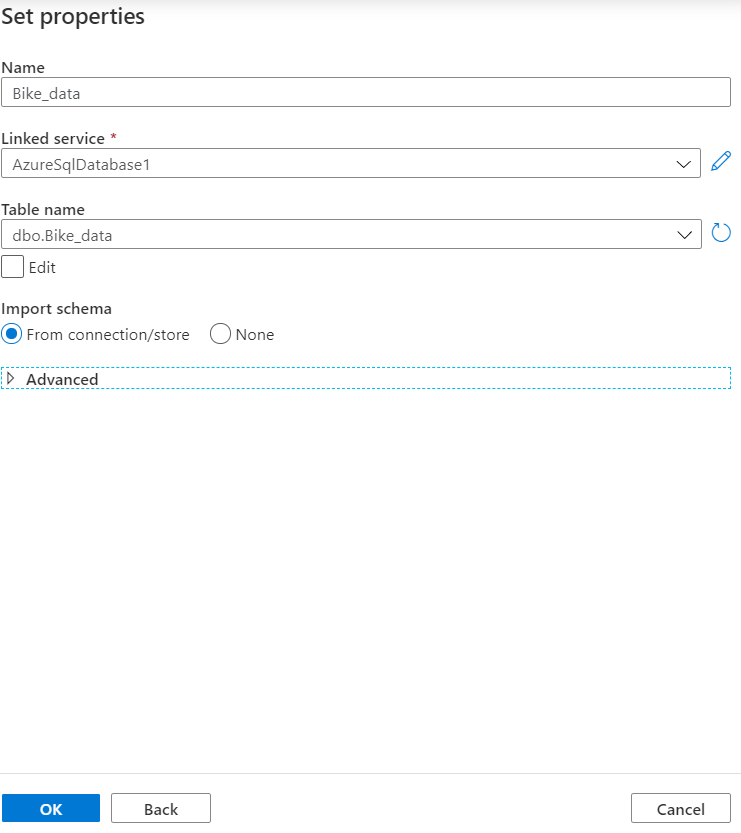
Sink

Click on Sink menu option

Clink on New (+)



Select Azure SQL Database and click on Continue



Enter Name

Select Linked service from the dropdown list

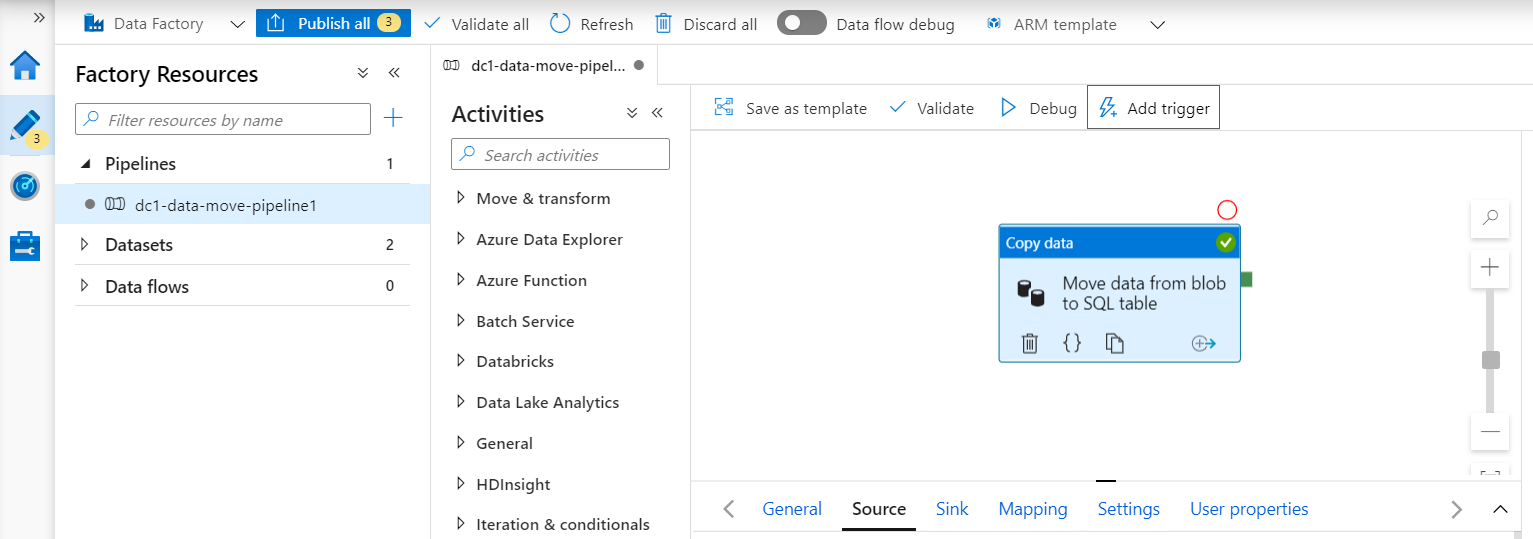
Select the table

Click OK

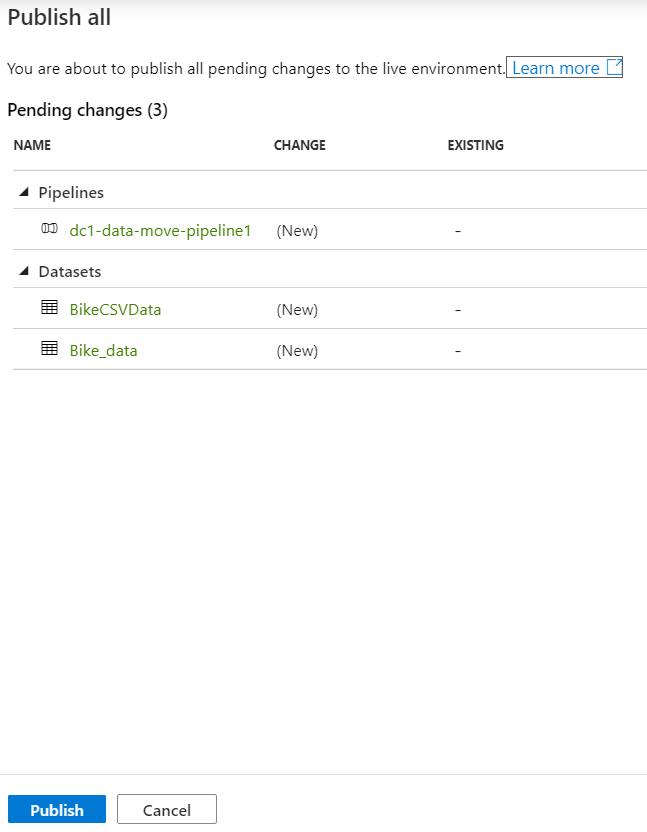
On the Source tab, you can also skip rows (1st header row, for example)

You can also, debug the pipeline to ensure it works as intended

You have to Publish the Pipeline before you can run it

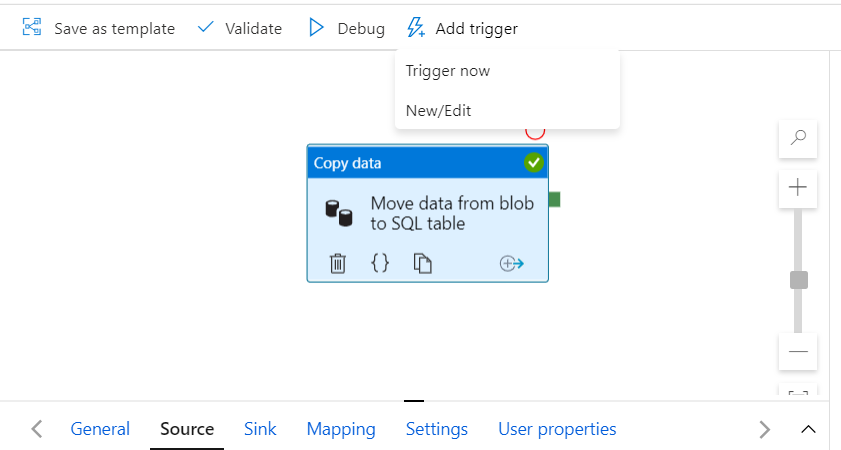


Click on Publish all

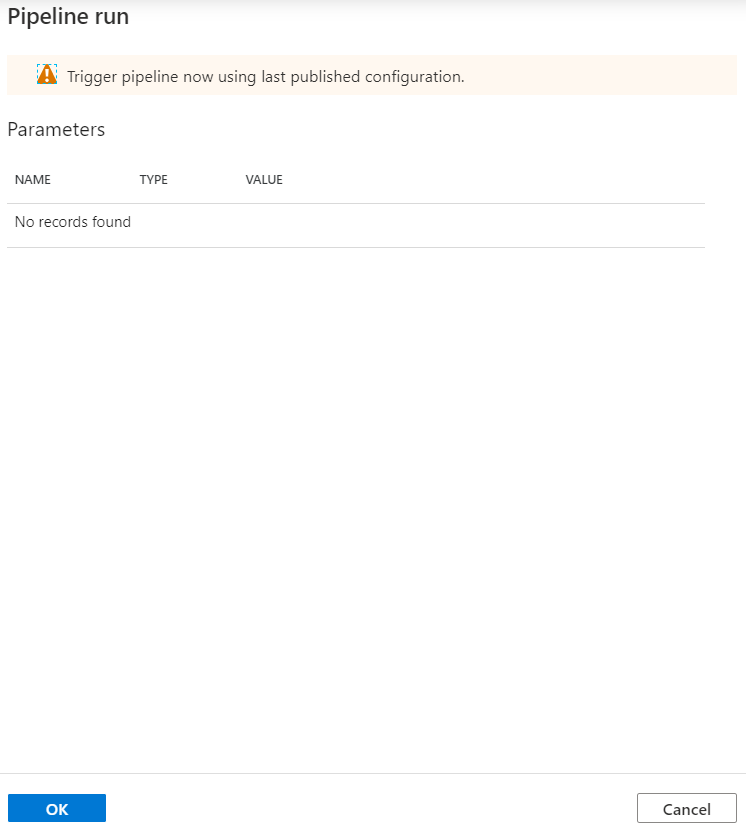


Click on Publish

When all resources are published, click on Trigger. You can run it immediately and you also schedule the job using a comprehensive Scheduler

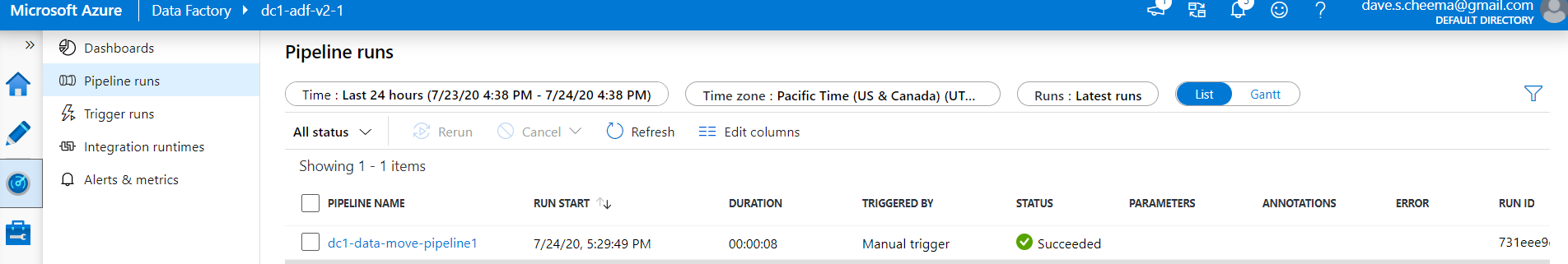


For this exercise, click on Trigger now option



Click OK

When the pipeline job ends, you can view and review its outcome by clicking on the Monitor icon on the left side of the screen



For example, it took 8 seconds to move 1492 rows from the Azure Blob storage into Azure SQL database table

You can also log into your Azure SQL database and verify the results.

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

ADF v2.0 with Dataset and Pipeline Parameters

Saturday, September 12, 2020

8:44 PM

Data copy using parameters and variables

Note: In this documentation, we’ll copy data from a blob storage and copy it into a SQL database

Create Azure Data factory 2.0 service in Azure portal

Create a storage account

Create a container in the storage account

Upload the source data file into the container

Create a SQL database instance

Go to Firewalls and Virtual networks of the SQL server

Add client IP

Go to the SQL database instance

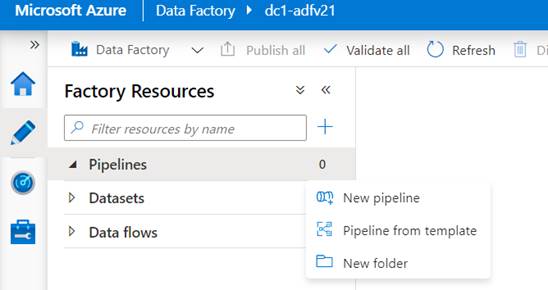
Click on Query editor and login to SQL database using credentials you created during SQL database creation

Create the target table that matches the source data schema and format

Go back to Azure Data factory

Click on Author & Monitor, in the middle of the screen, on the right-side panel

Click on Author icon on the left panel, click on Pipelines, click on New pipeline



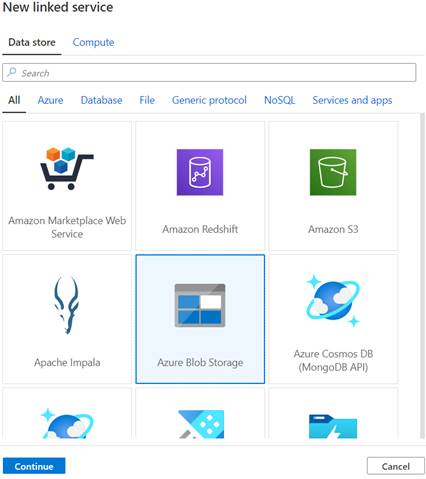
Give it meaningful name

Click on Manage icon on the left menu, it will show Linked services panel

Click +New

(You have to define a linked service for each input and output source/target)

Select Data Store à click on Blob storage icon and click on Continue

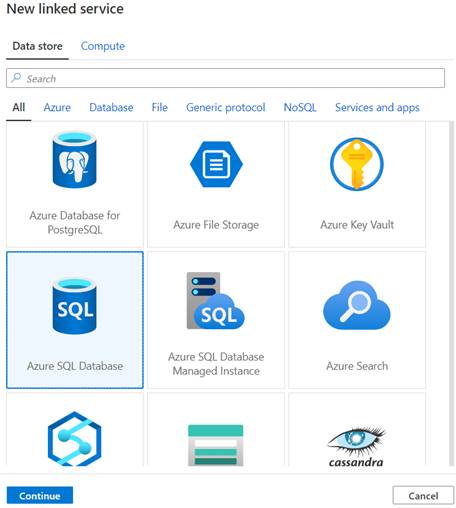


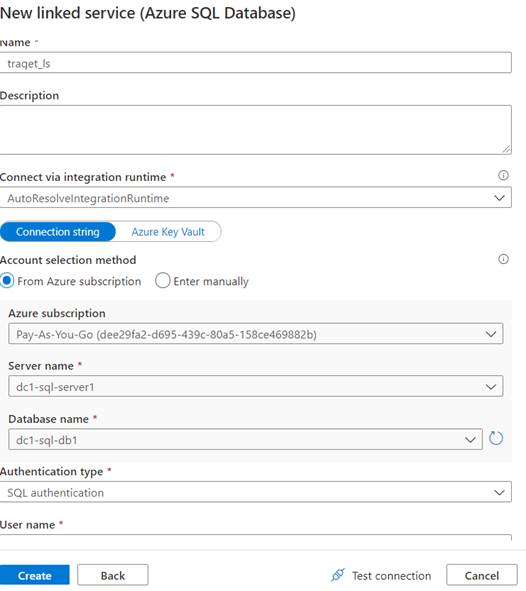
Select the Blob storage location and credentials and test Connection.

If connection test is successful, click on Create and a linked service for the source data will be created

Now create a second linked service for the target (e.g., a SQL database) by clicking on +New on the Linked Services screen

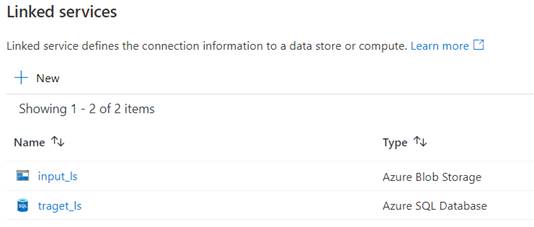
Select Azure SQL database on the Data Store screen and click Continue





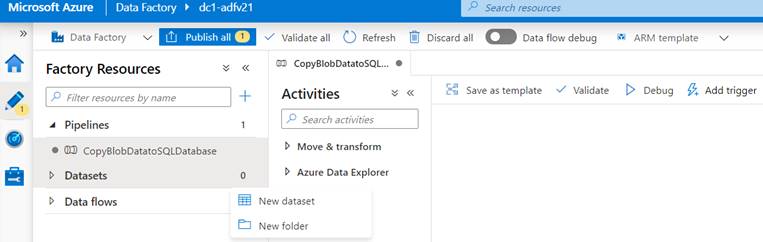
Enter target database credentials and Test connection

If test connection successful, click Create

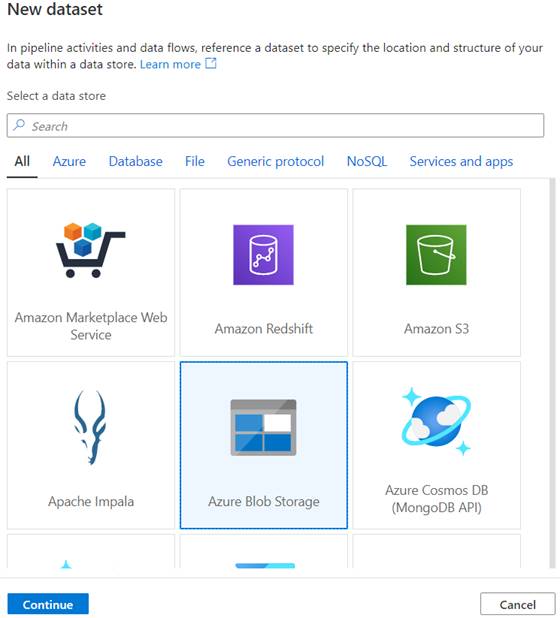


At this point, you should have two (input\_ls and target\_ls) linked services

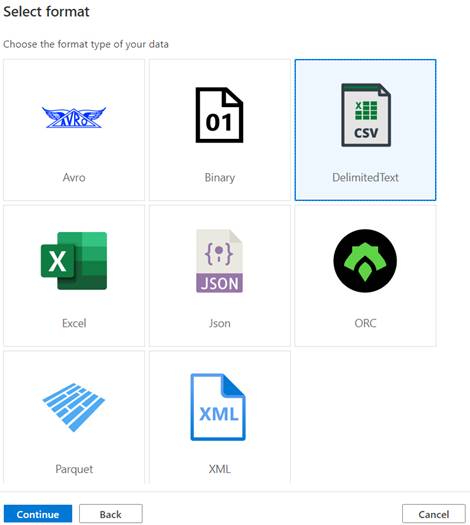
Now create two Datasets (source\_ds and target\_ds) by clicking on the Author icon on the left of the screen



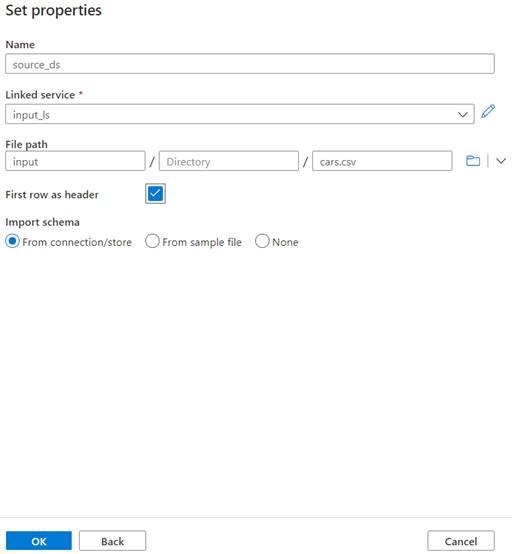
Click on New dataset



Click on Azure Blob Storage and click on Continue



Select CSV (source data format) on the Select format screen and click on Continue



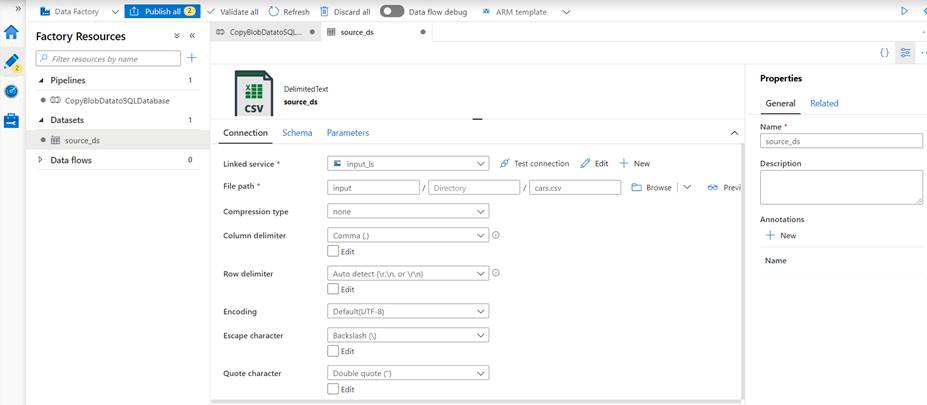
Give it a name, e.g., source\_ds.

Select Linked service, input\_ls, in this case

Select File Path by clicking on the Folder icon and the source file

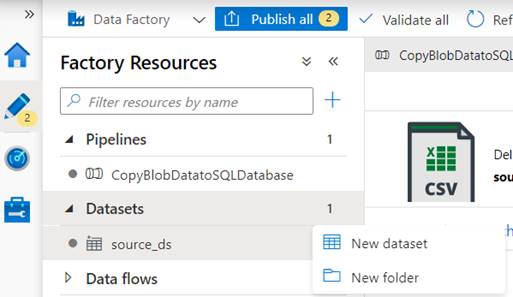
Check on First row as header, if source file contains headers

Click on OK



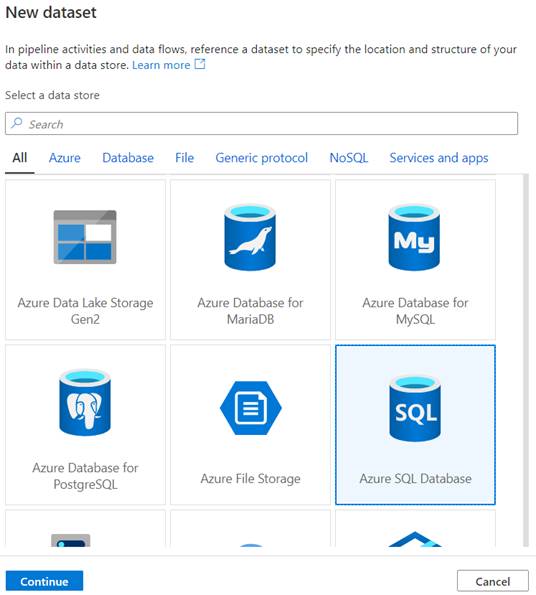
You can preview source data by clicking on the Preview

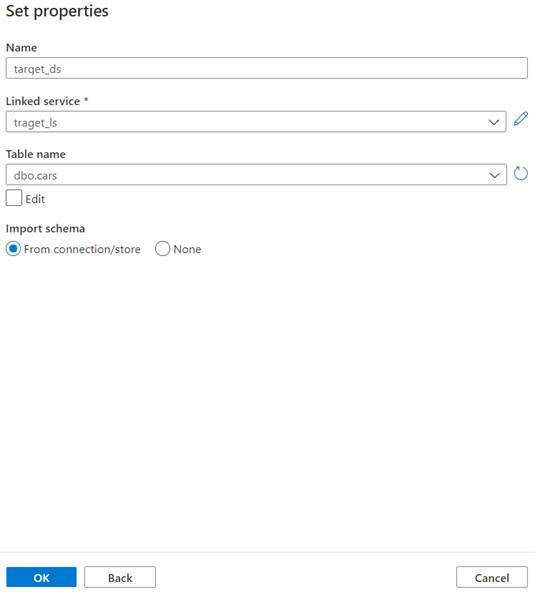
Create another Dataset for the target



Click on Datasets à New dataset

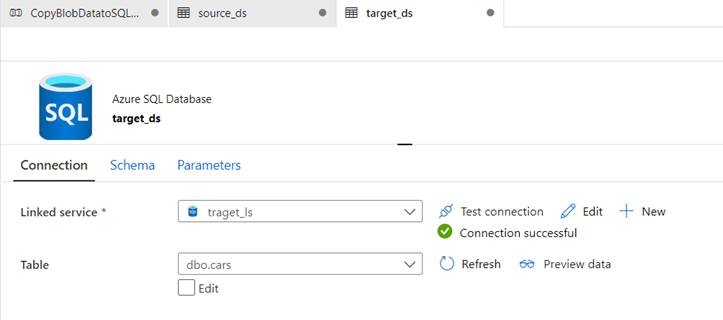
Select Azure SQL Database on New dataset screen and click on Continue





On the Set properties screen, give it a name, e.g., target\_ds, select Linked service, e.g., target\_ls, and select Table name for the target

Click on OK and you should see:

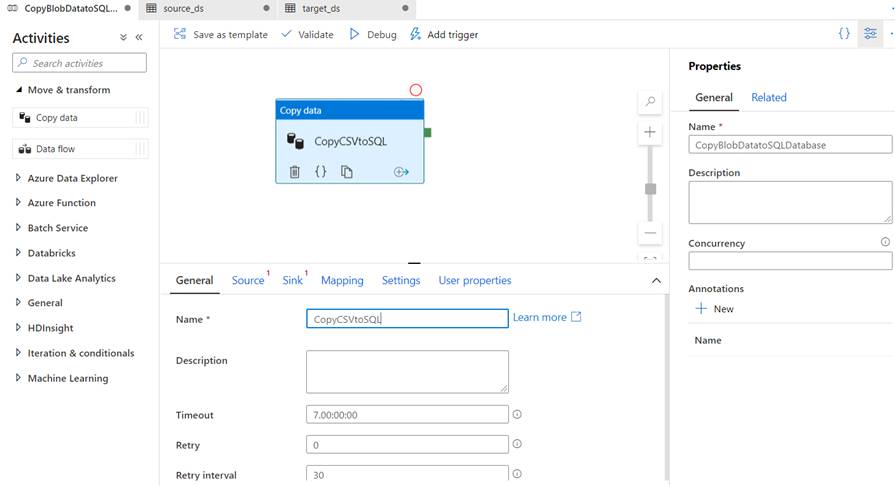


Click on the Pipeline instance you had created earlier

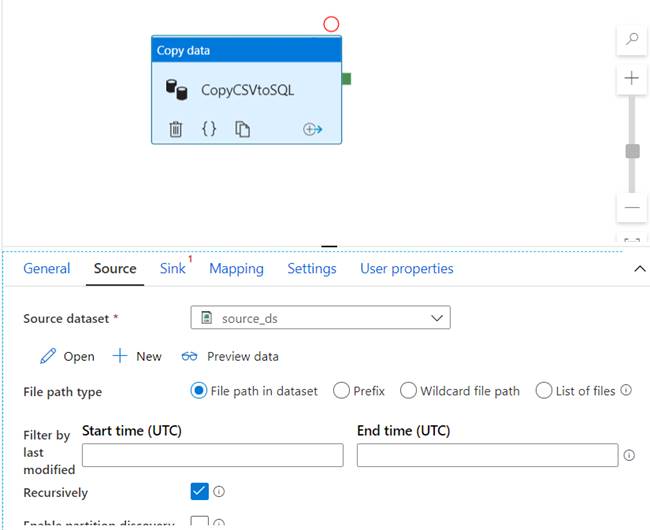
Under Activities, click on Move & transform à drag Copy data to the main panel

Click on the Copy data activity. The bottom half of the screen will show the details of the Copy activity and configuration settings

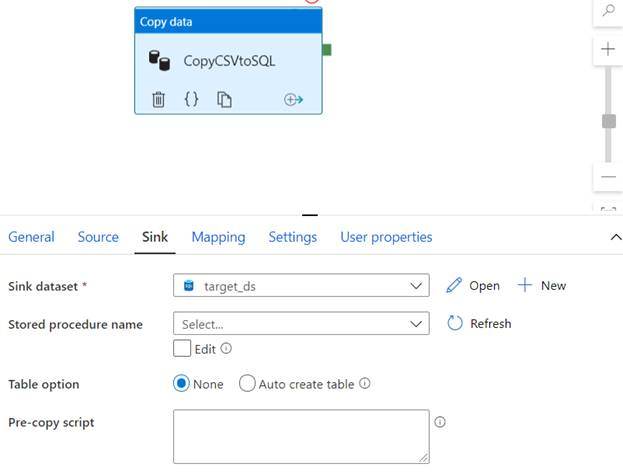
Give it a name, e.g., CopyCSVtoSQL



Click on the Source (menu item)

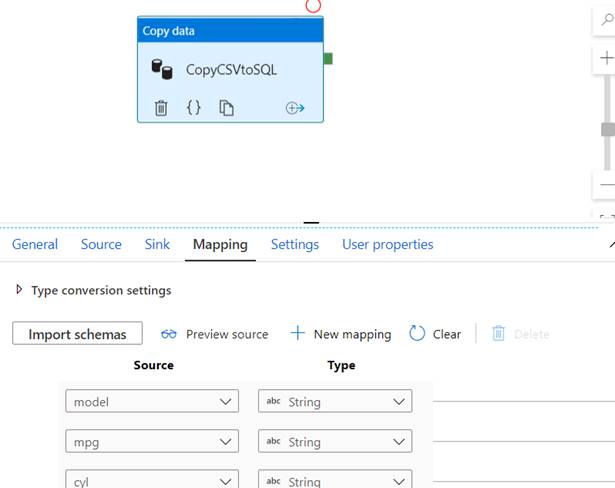


Select Source dataset, e.g., source\_ds. You can Preview data here



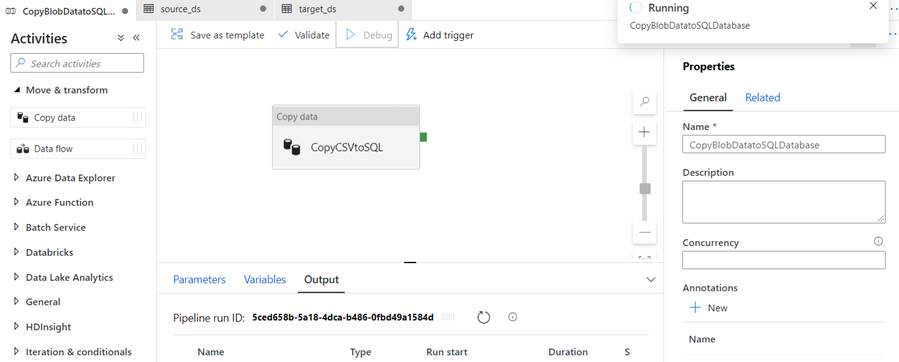
Click on Sink menu item and select Sink dataset

You can click on Mapping to ensure that source and target are mapped as expected

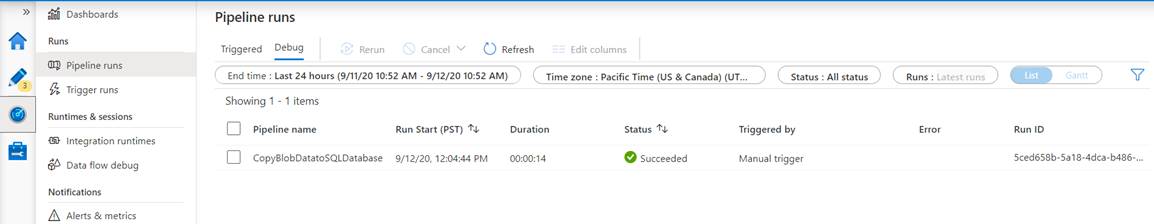


Now you’re ready to copy data. You can Publish pipeline items and click on trigger. However, it is recommended that you run it thru the debugger before publishing it.

Click on Debug



When the Debug run is completed, you can view the run results by clicking on Monitor icon on the left



You can view the Status, Error and other details.

**Parameters**

There are two ways (levels) to leverage parameters: Dataset level and Pipeline level

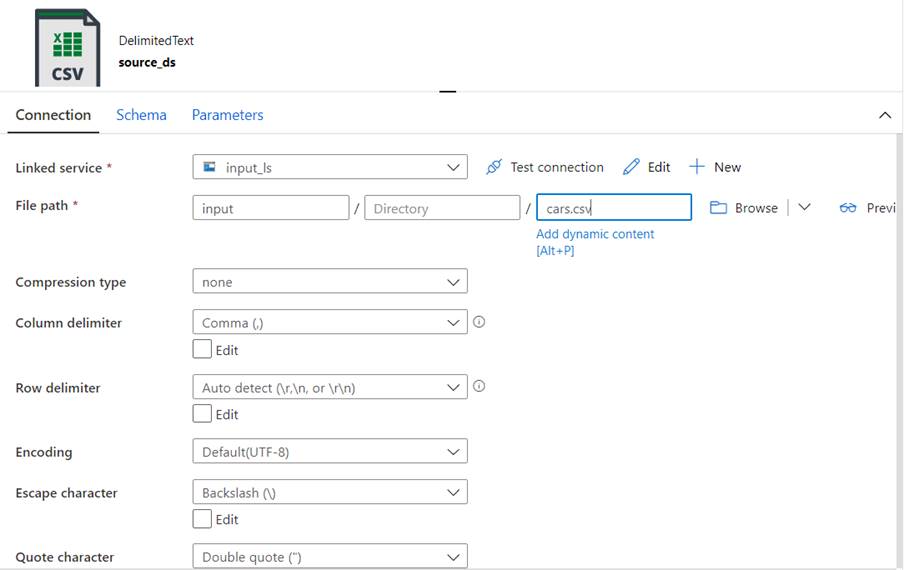
**Dataset level parameters**

Double click on the dataset, e.g., source\_ds



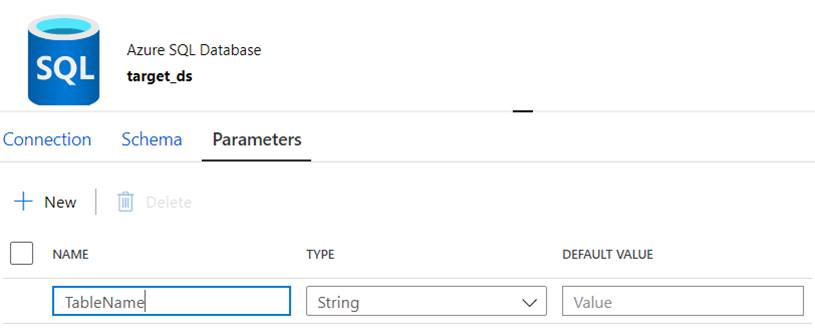
Click on Parameters à click on + New

Enter the name of the parameter, e.g., FileName



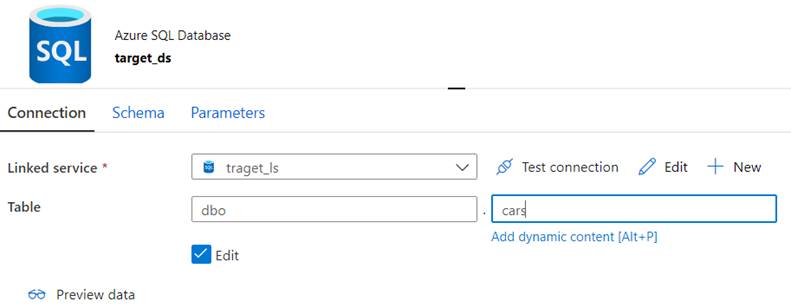
Clear out the file name, e.g., cars.csv. You will see Add dynamic content (Alt+P)

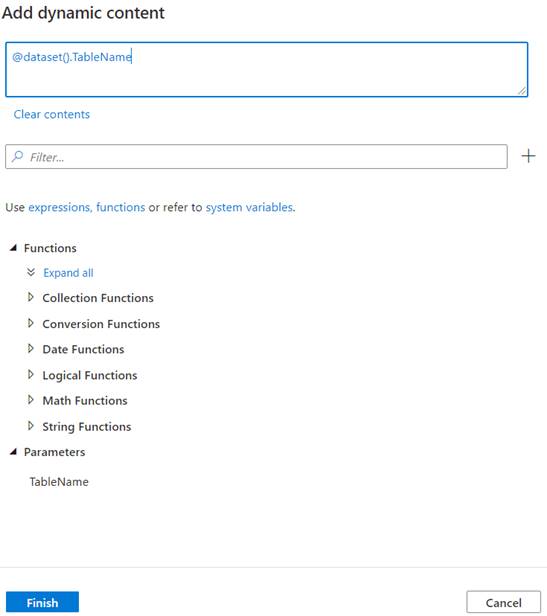
Click on Add dynamic content (Alt+P)



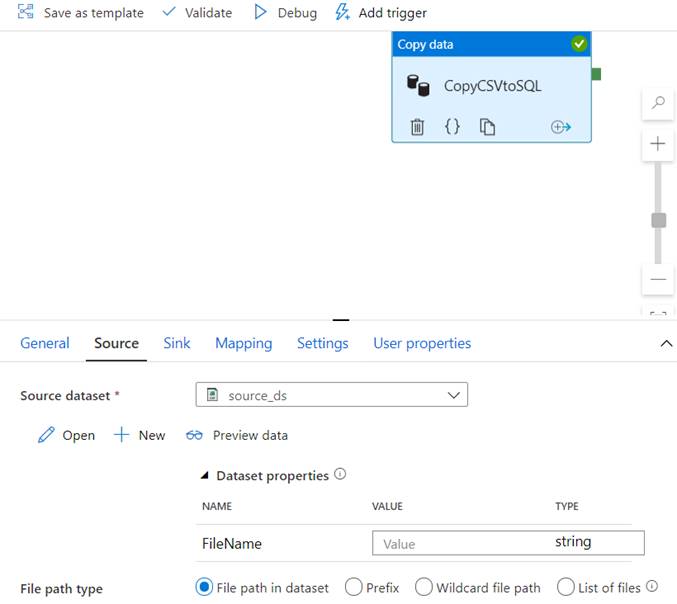
Click on Parameters and enter parameter name, e.g., TableName

Click on Connection menu item, check on Edit, clear out the table name, cars and click on Add dynamic content (Alt+P)





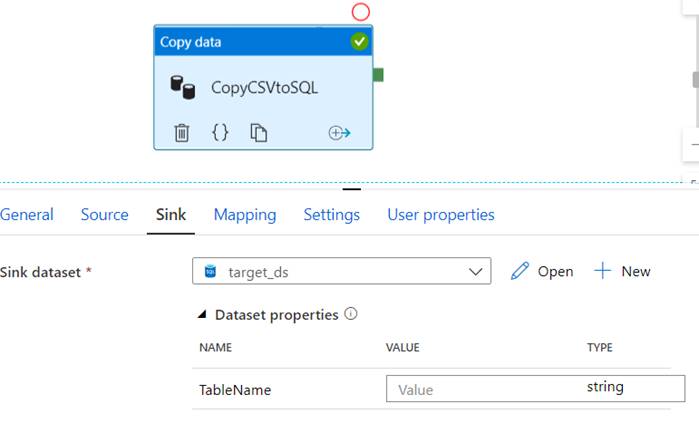
Go to Parameters section and click TableName and click on Finish



Click on Copy data pipeline

At the bottom half of the screen, click on Source menu item. You will see DataSet properties. And below it is FileName and empty Value.

Enter the name of the file name that you want to copy in the Value field, e.g., cars.csv



Next click on the Sink menu item and type in the name of the target table in the Value field, e.g., cars

Then, you can click on Mapping menu item to ensure source to target mapping. If you don’t see to correct mapping, click on Import schemas. You’re ready to run it in Debug mode.

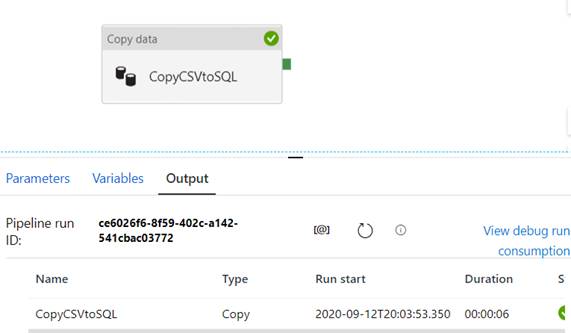
Truncate target table to ensure it is empty to avoid any confusion

Click on Debug

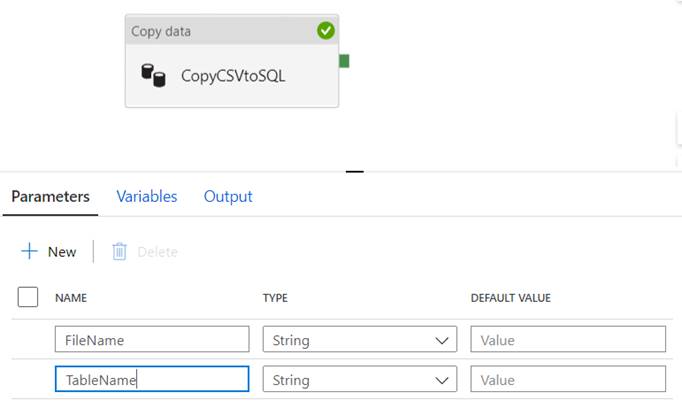
Verify the result by clicking on the Monitor icon on the left side and view the output of your run

**Pipeline level parameters**

Click on away from the pipeline instance



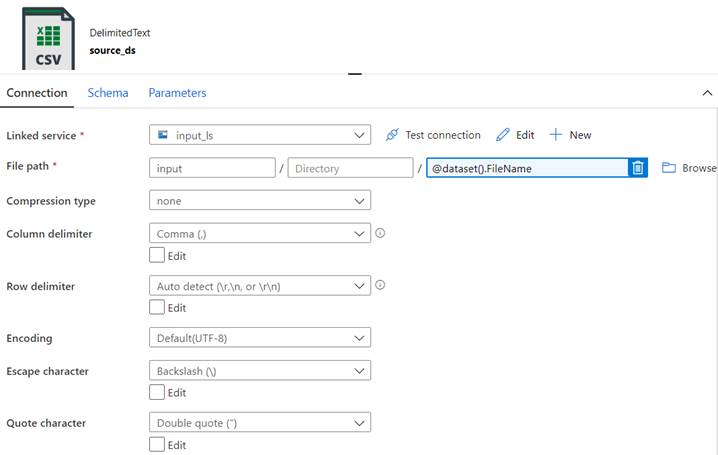
Click on Parameters



Enter the names of the parameters in the list

Go to Datasets menu option on the left side and click on it

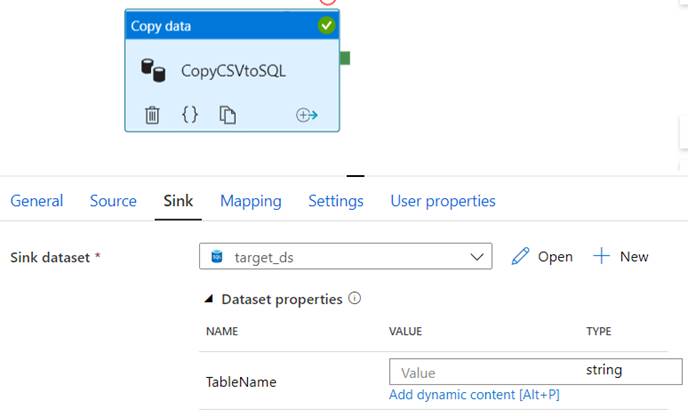
Select your source dataset,



Click on Copy data pipeline instance

In the bottom section of the panel, click in the Value column of the FileName

Remove @dataset().FileName value and click on FileName in the Parameters section



Click on the Sink menu item, click out the value in the TableName value column and click on Add dynamic content (ALT+P)

Click on the TableName in the Parameters section and click on Finish

Click on the Copy data pipeline instance and click on Debug

Provide the names of the FileName and TableName and click on OK

Wait for the run to finish, click on Monitor icon on the left side and view/review the results of your run

**Copy a different file to a different database table using parameters**

Load source file into the blob

Create a target table in the target database

In the pipeline run, click on Import schemas. It will ask for the pipeline parameters (FileName and TableName)

Provide the source file name and the target table name, e.g., planes.csv and planes

Click on the Copy data pipeline instance, click on Debug and provide the names of the parameters (planes.csv and planes) and click on OK

Wait for the run to finish, click on Monitor icon on the left side and view/review the results of your run

**Backup a database table to a CSV file**

Drag a second, BackupTableData (BTD), Copy data activity from Activities

Connect BTD to the original Copy data activity pipeline instance

Click on BTD

Setup the source and sink properties and configurations, but this time reverse of the original settings

Ensure that mappings in BTD and the original Copy data pipeline instance are consistent

**Publish Pipeline and datasets**

Before you can deploy a pipeline, you must publish the pipeline and the datasets

The you use triggers to kick off a pipeline activity. BTW, ADF 2.0 has very comprehensive trigging capabilities

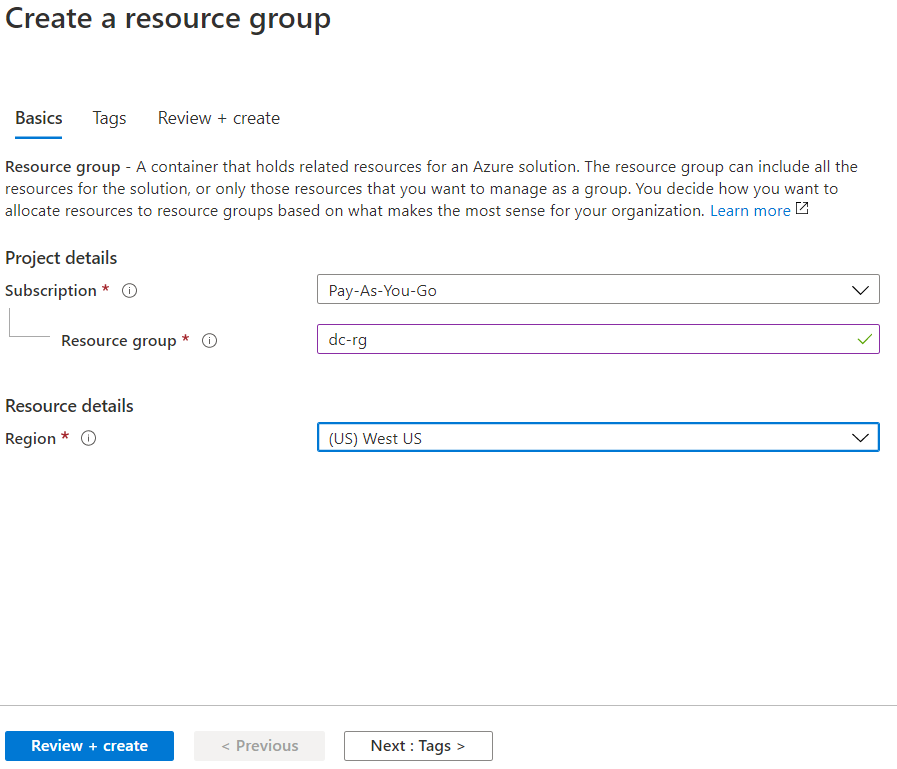
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

ADF v2.0 Call Databricks with Parameters

Monday, September 21, 2020

6:47 PM

1. Create a resource group in Azure portal



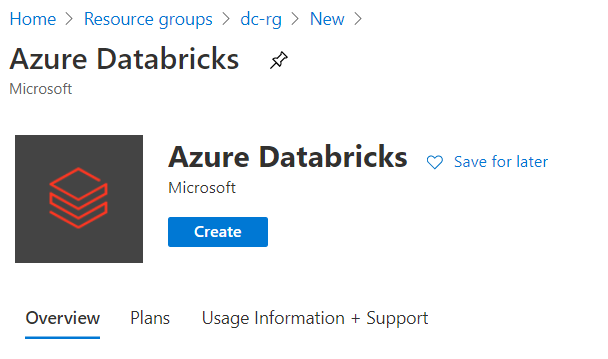
Enter Resource group name, select Region and click Review + Create

Go to the Resource group details

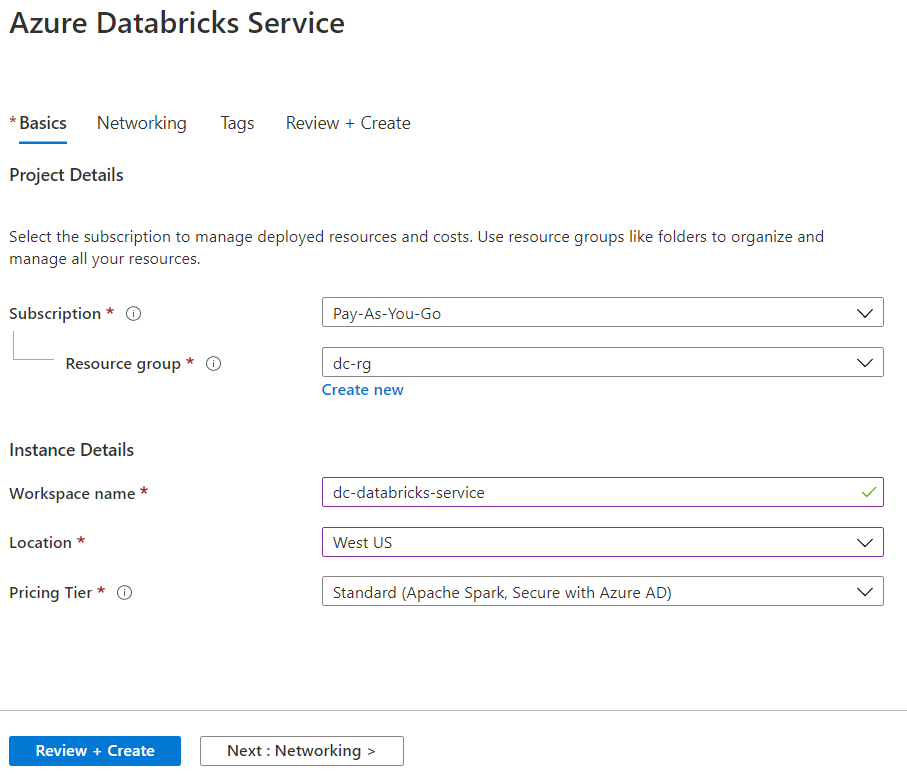


Click on + Add

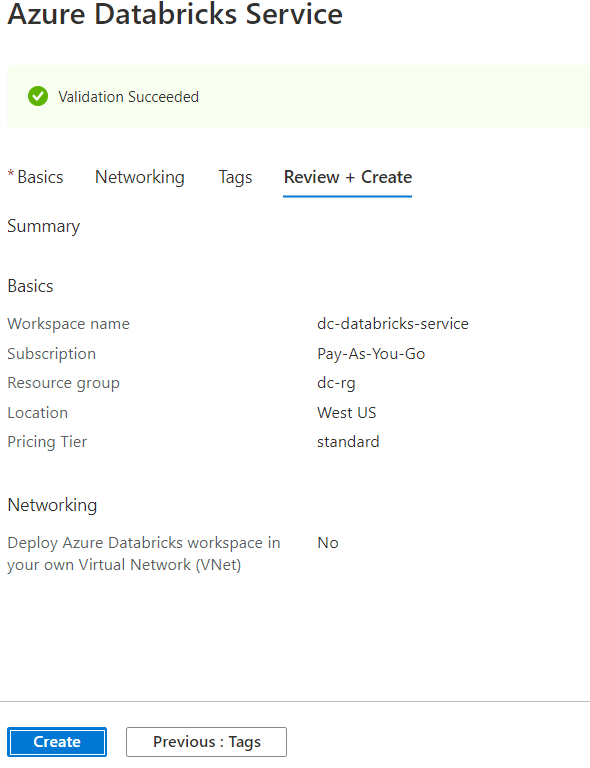
Search for Azure Databricks. When found, click on it



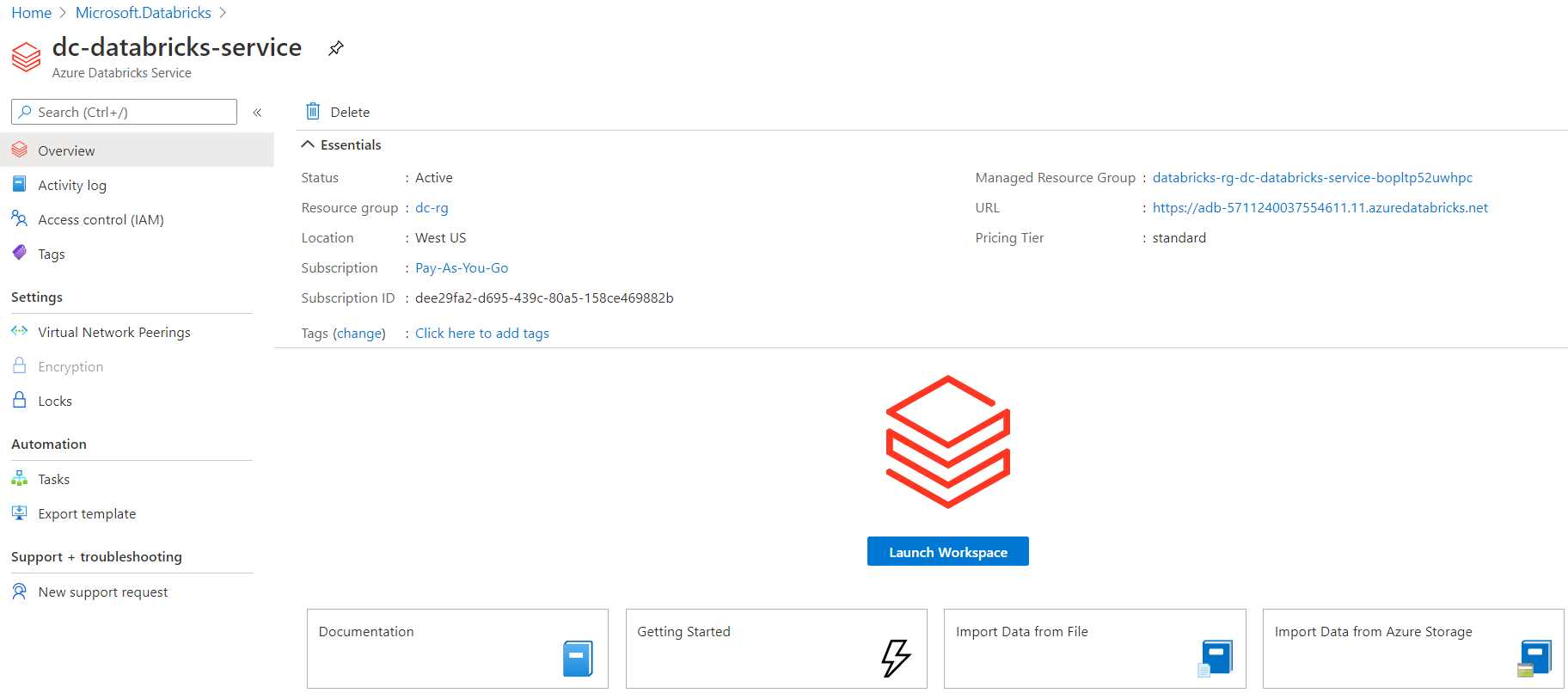
Create on Create



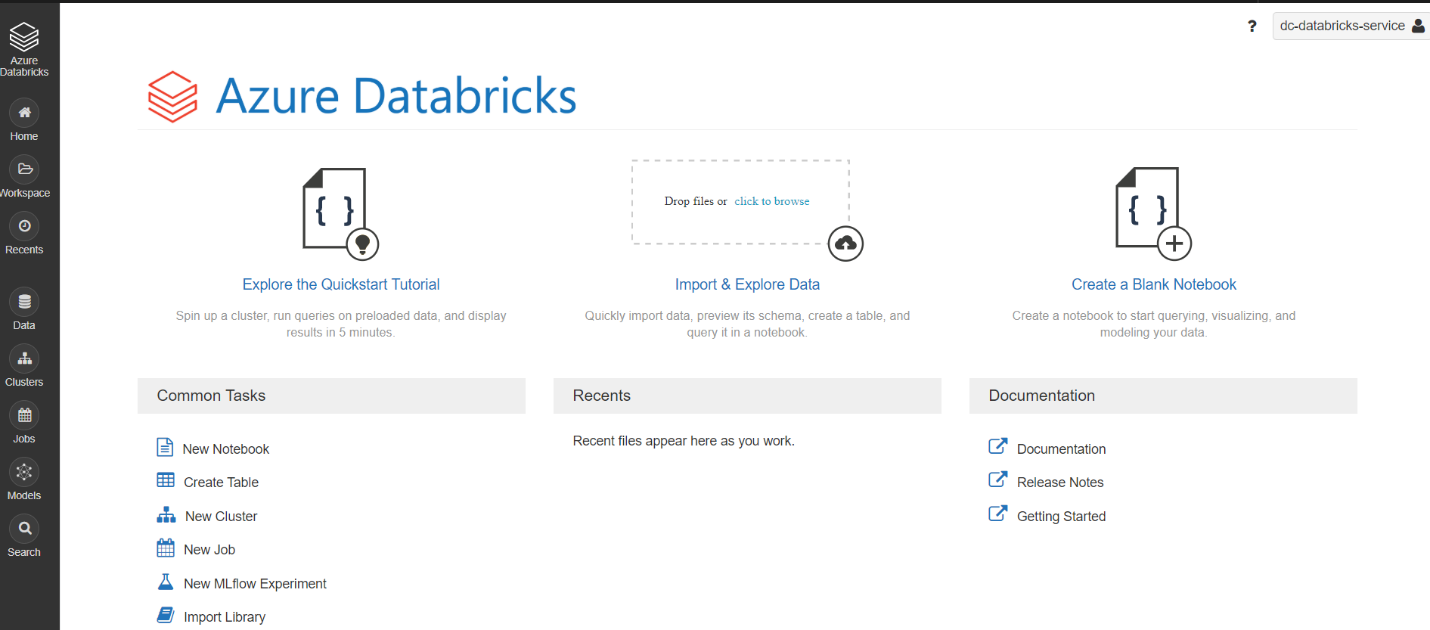
Select the Resource group, Workspace name, Location (should be the same as the resource group) and click on Review + Create



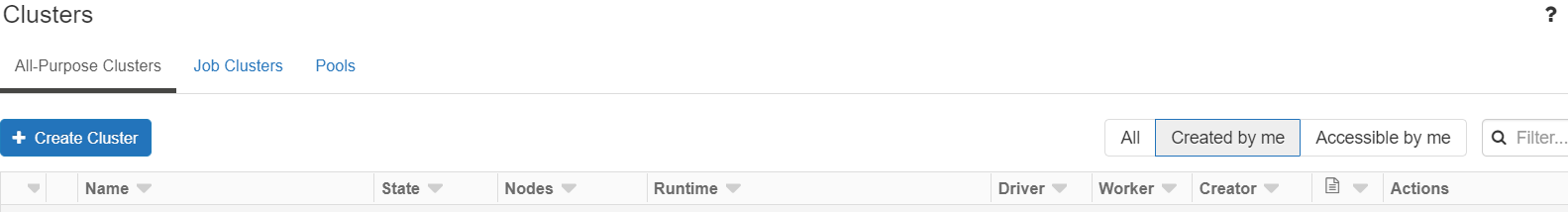
Click on Create



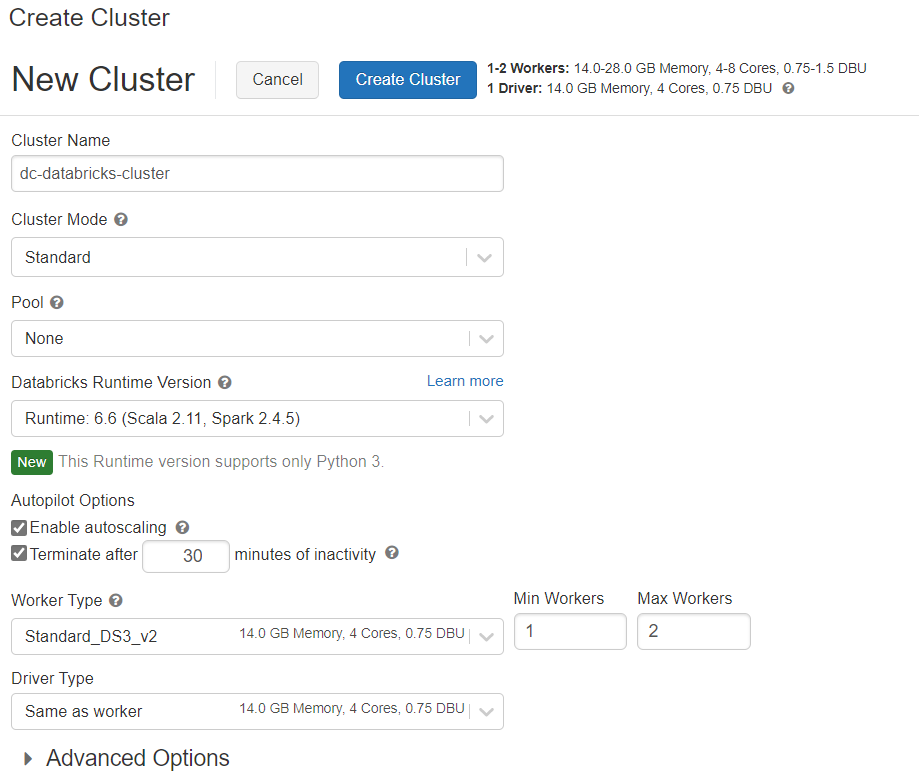
Click on Launch Workspace (in the middle of right panel)



Click on Clusters on the left side menu



Click on + Create Cluster

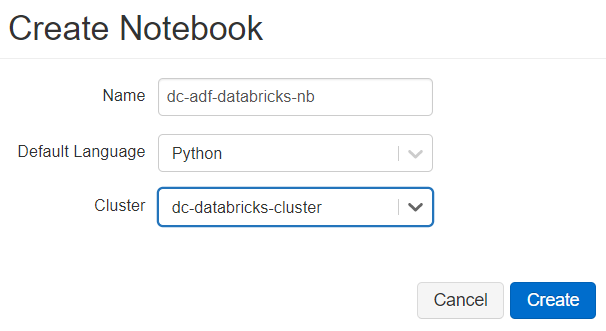


Enter Cluster Name, select Databricks Runtime Version, enter Terminate after \_\_\_\_ minutes of inactivity

Select Worker type, Min Workers and Max Workers

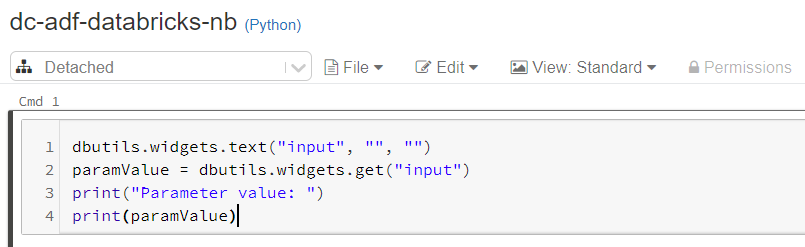
Click on Create Cluster (at the top)

After the cluster is created (takes about 20 minutes), click on Workspace --> Create --> Notebook



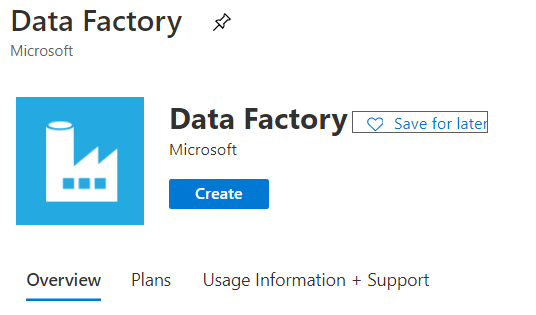
Enter Name, select Default Language, Cluster and click on Create

The script below shows how to receive and interpret Parameters

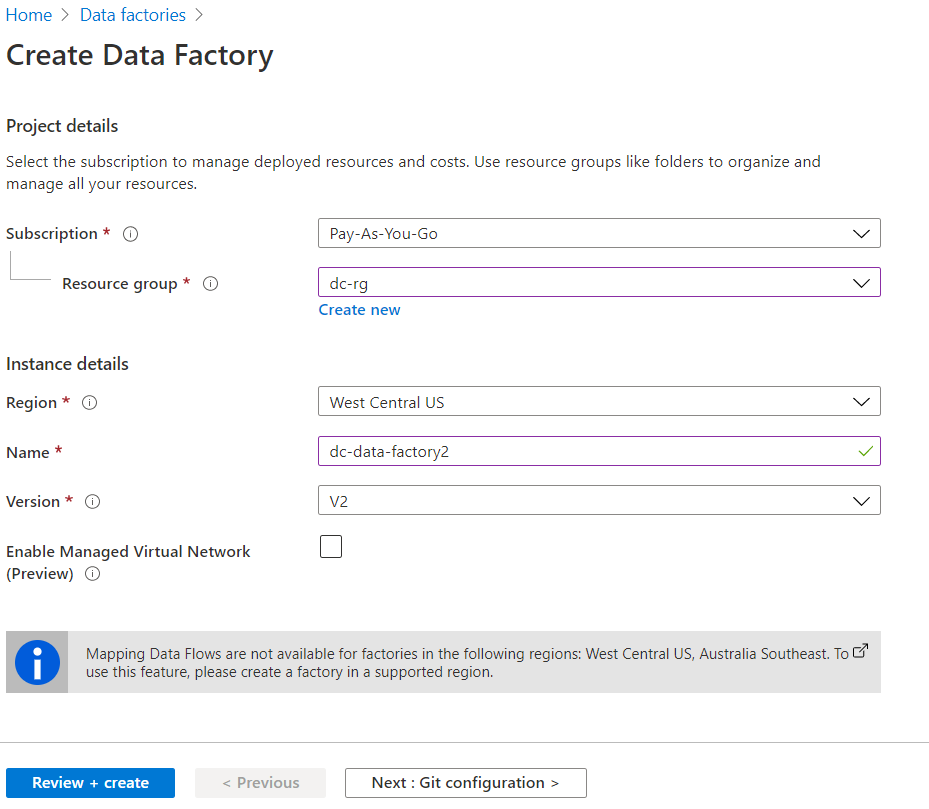


Now go back to Azure portal and create an instance of Azure Data factory v2.0

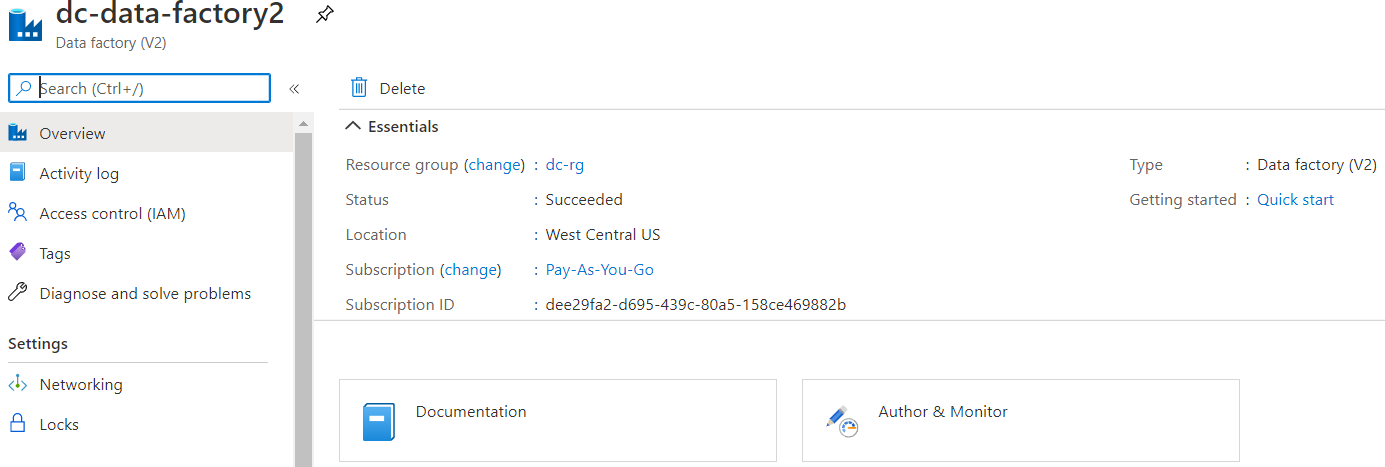
Search for Data factory and click on it



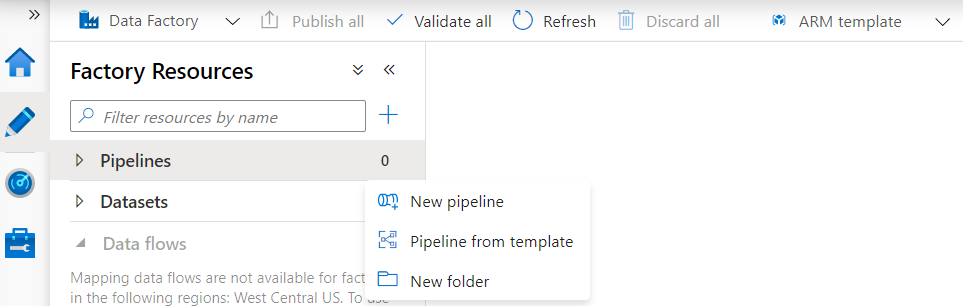
Click on Create



Click on Review + create

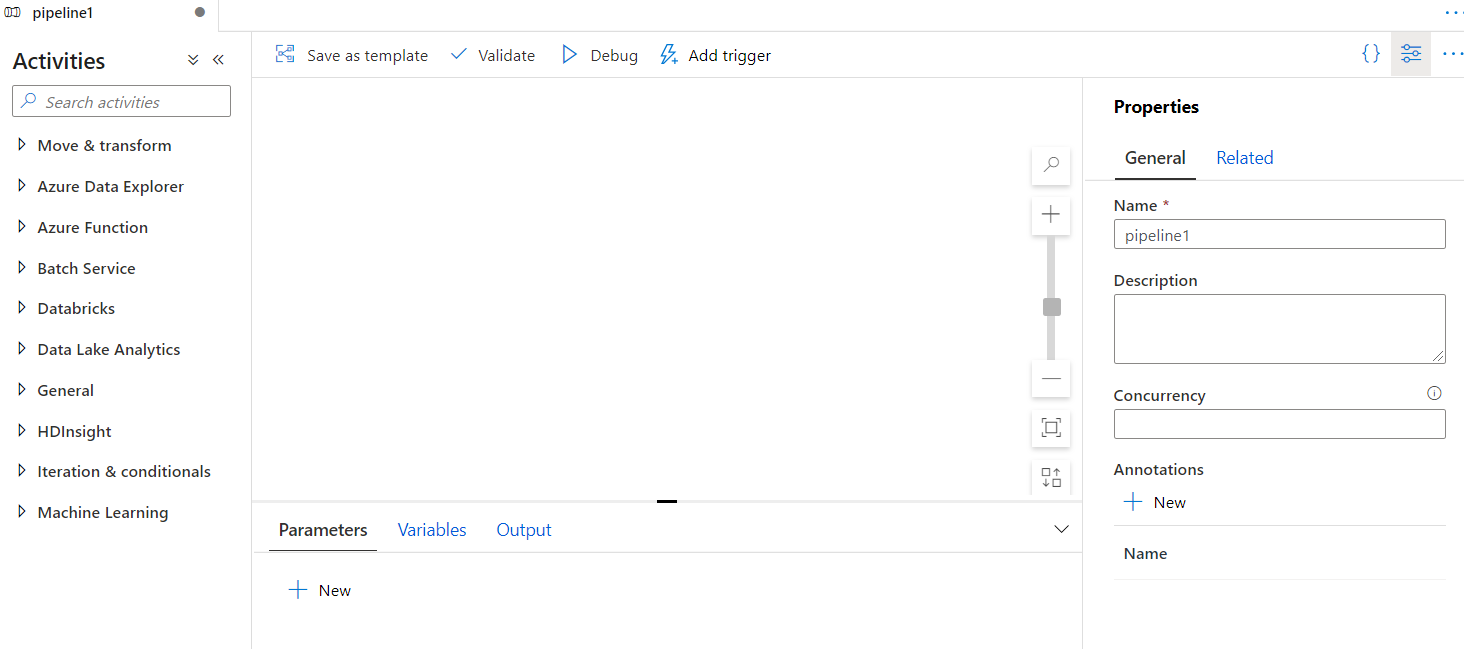


Click on Author & Monitor (in the center of right panel)

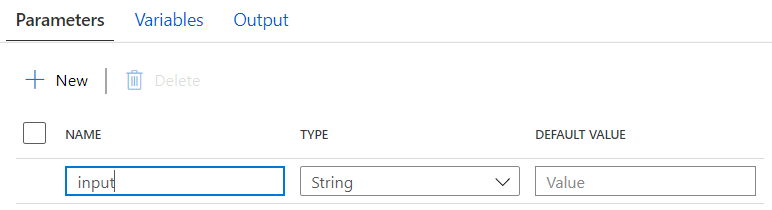


Click on Author --> Pipelines --> New pipeline

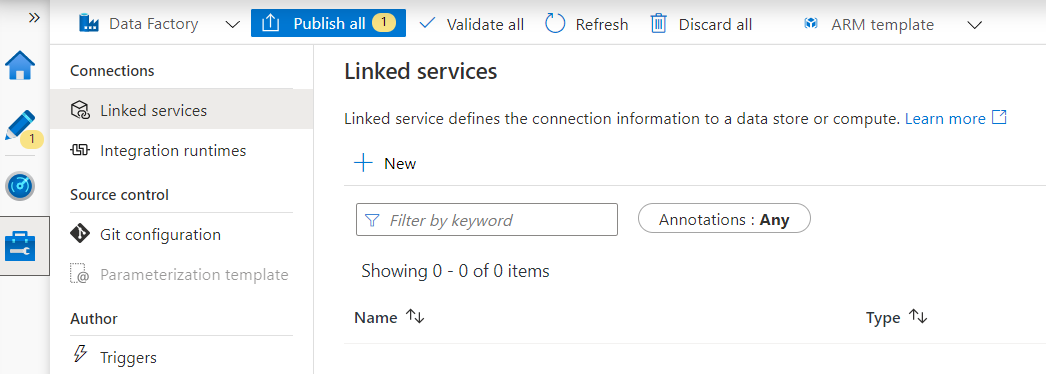
Note: in the exercise, we'll be sending simple parameter named input, of type string from ADF to Databricks. To do this, look at the bottom half of the right side panel



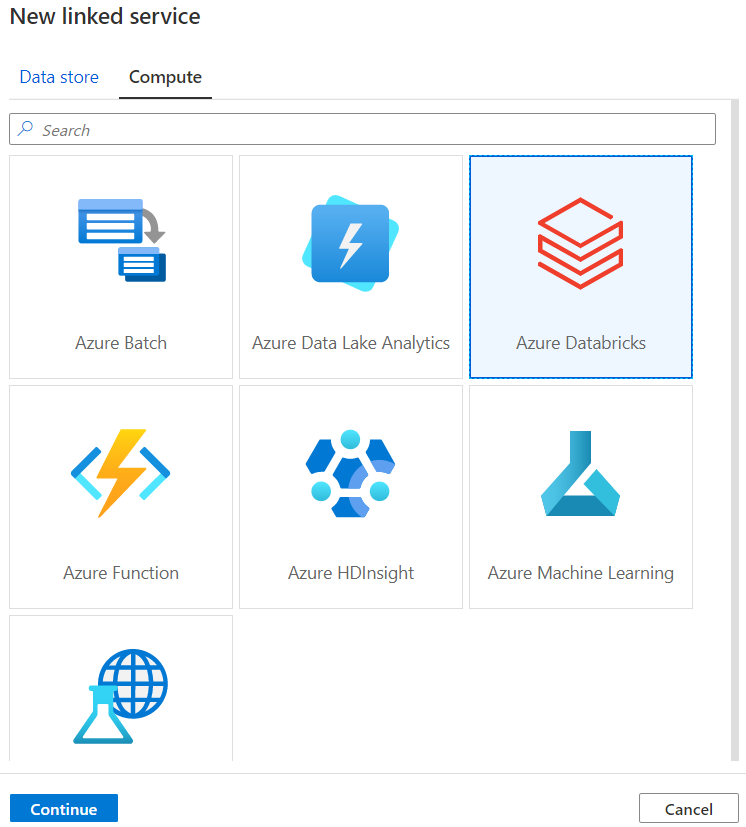
Under Parameters, click on + New



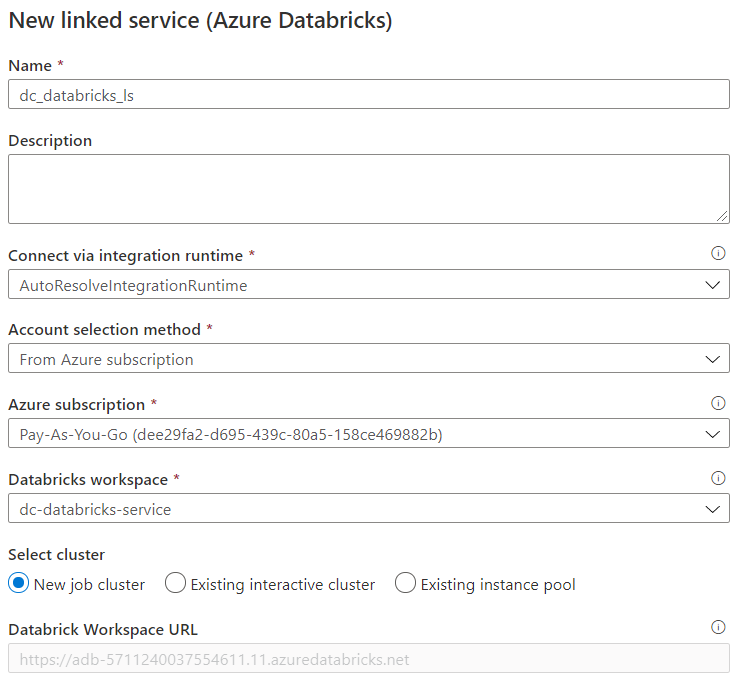
Now click on Manage menu item, on the left side

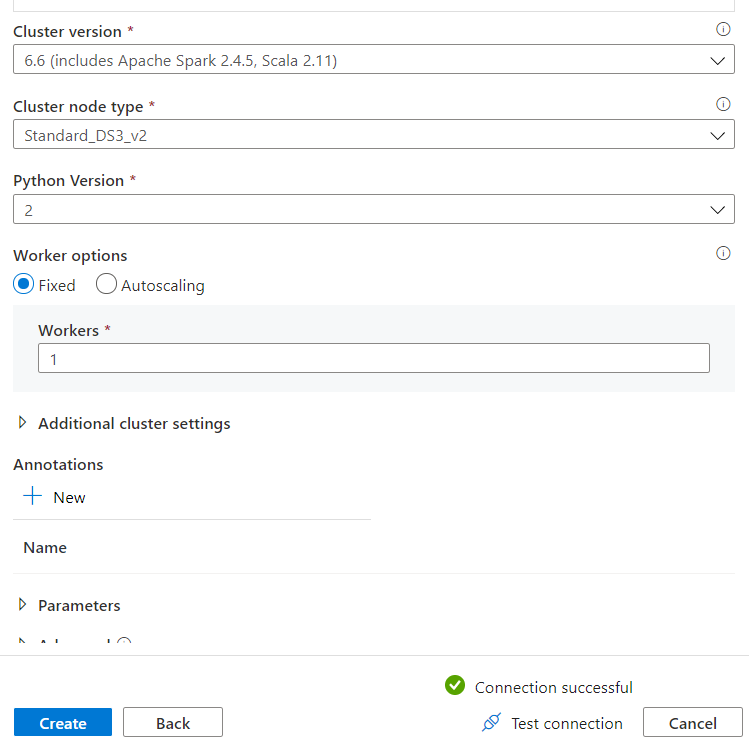


Click on + New under Linked services



Click on Azure Databricks and click on Continue

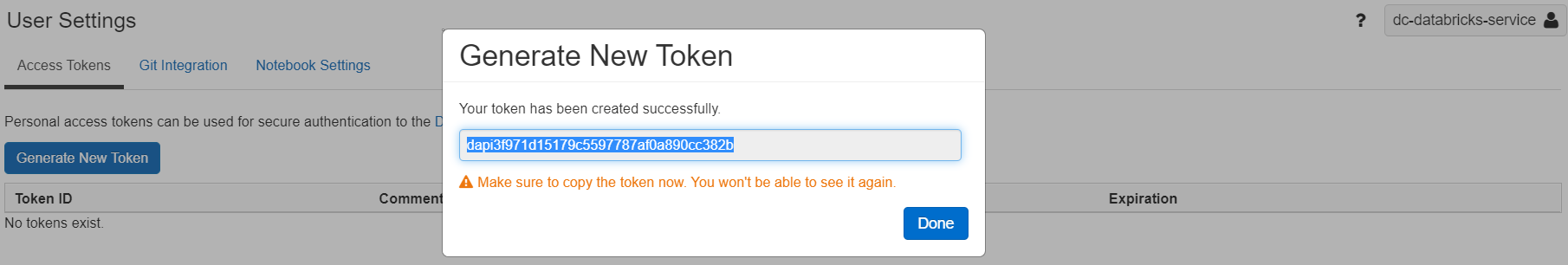




Enter Name and the rest of the parameter values

You will get Access token from the Databricks session user settings

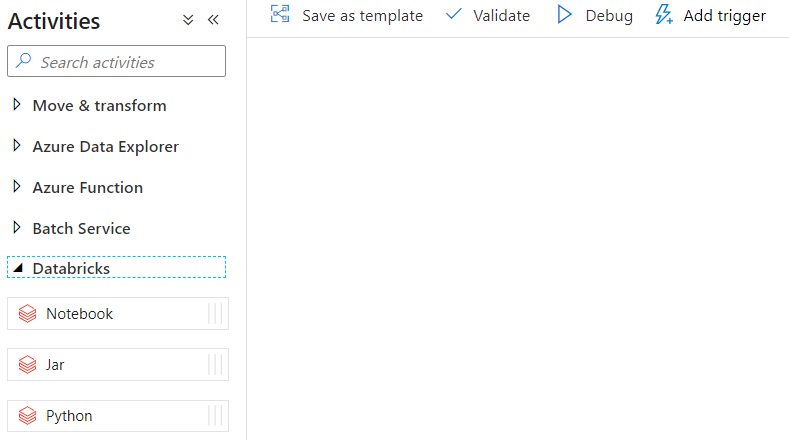
Click on user icon, at the top left corner, user settings, click on generate token, enter description and click on Generate



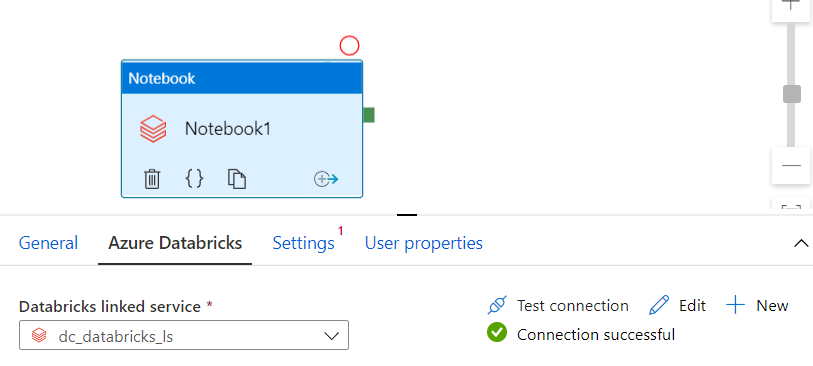
Save the token and enter it in the Access token field in the Linked service panel

Click on Create

Click on Author menu item and select your pipeline, e.g., pipeline1

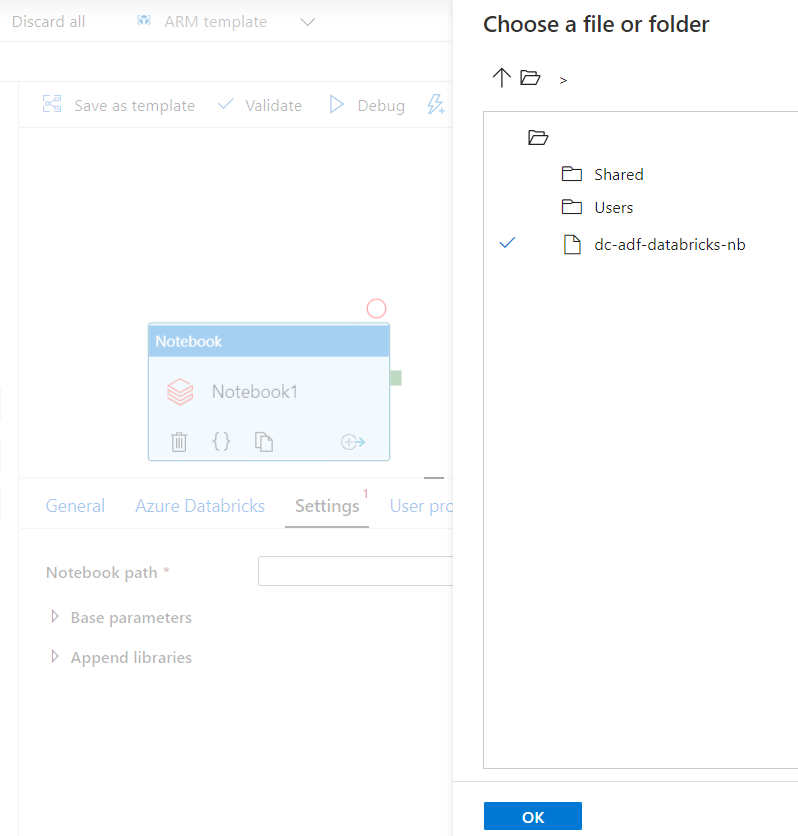


Under Activities, click on Databricks. Select and drag the Notebook icon and drop it in the pipeline panel

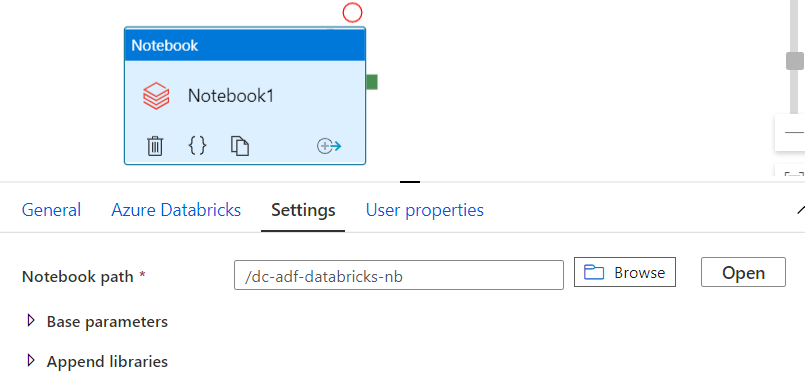


Click Azure Databricks and select Databricks linked service, Test connection

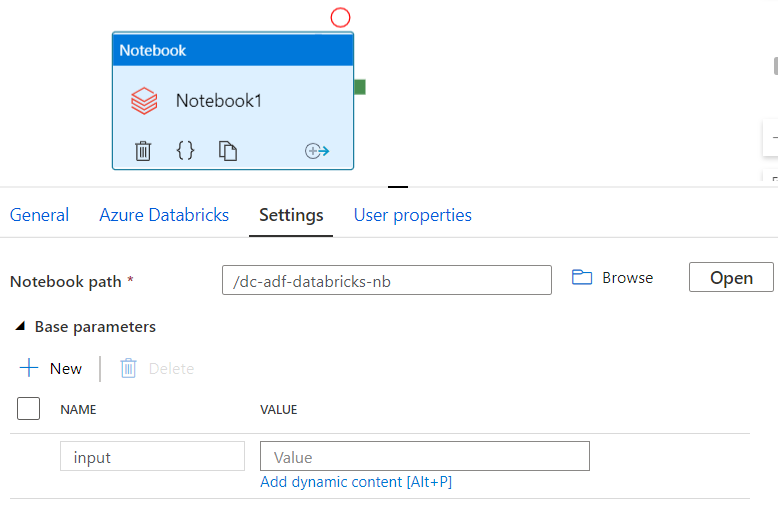
Click on Settings



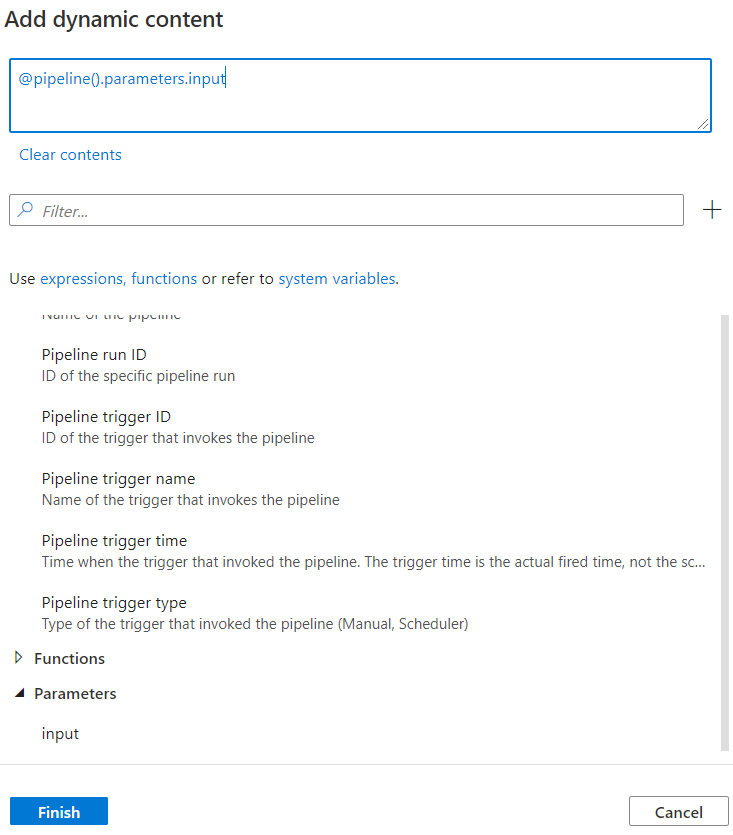
Click on the folder icon, follow it till the notebook is found. Click on Notebook and click on OK



Click on Base parameters

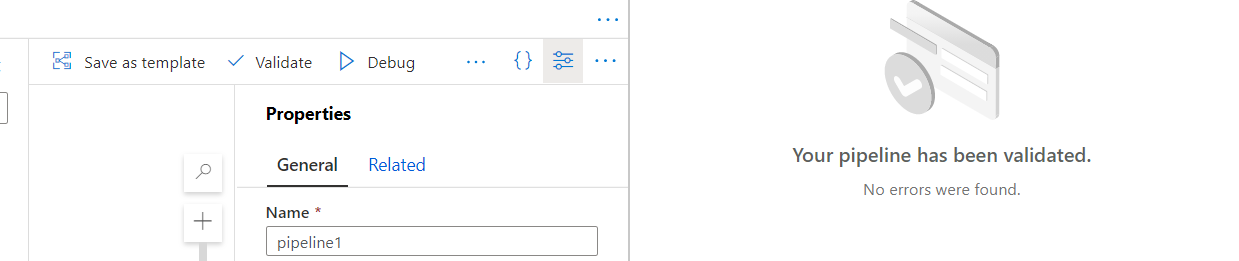


Enter input in the Name field and tab into the Value field. You will see Add dynamic content [Alt+P]. Click on it

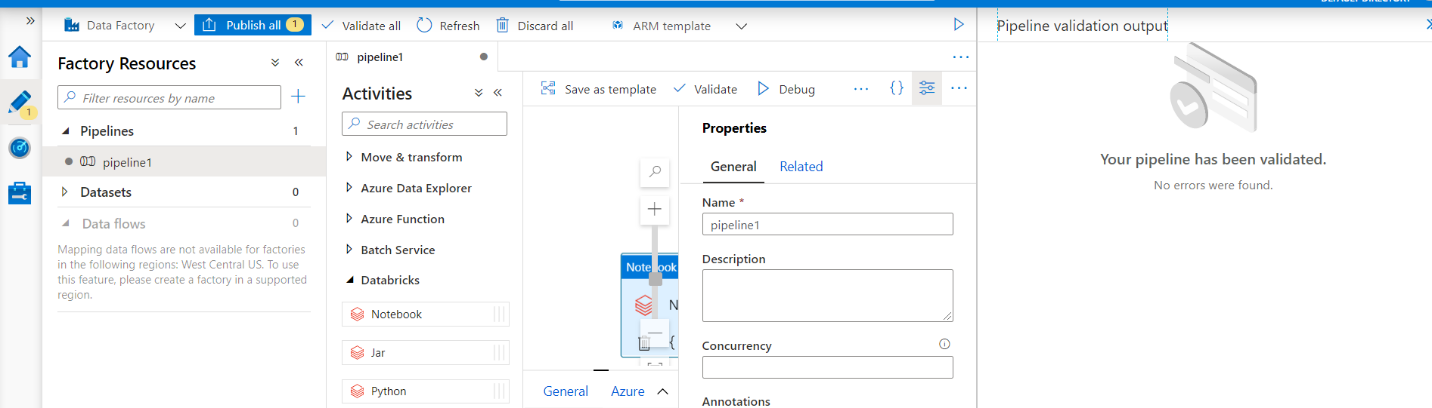


Go to Parameters section and click on input and you will the value in the contents field

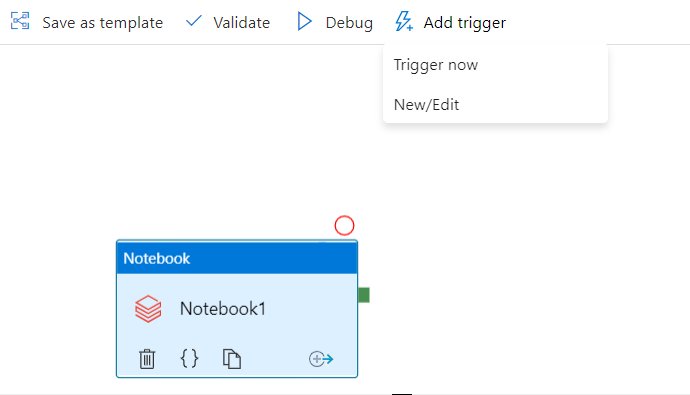
Click on Finish



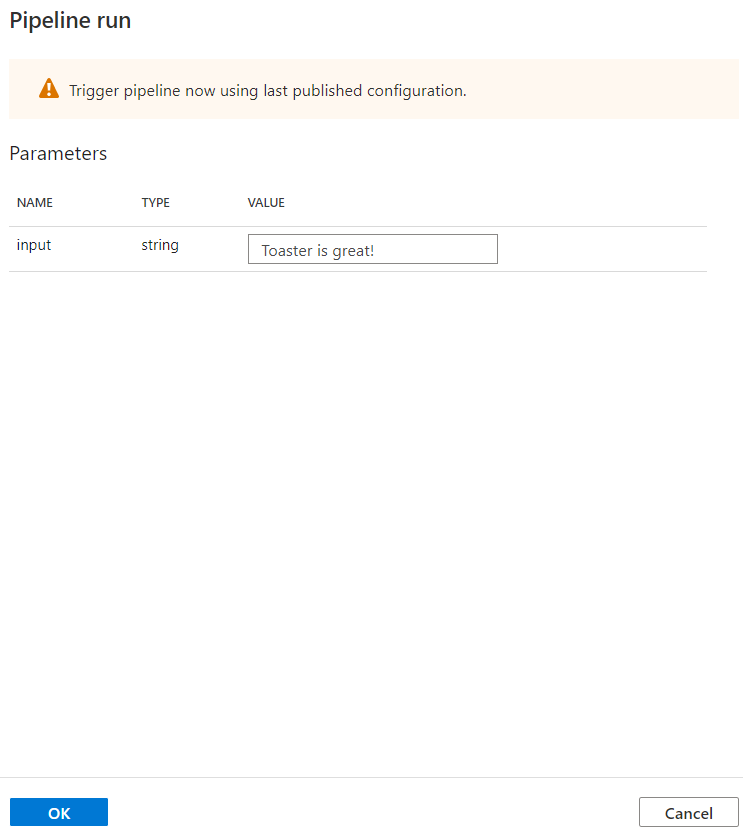
Click on Validate menu item



Click on Publish and click on Publish in the detail panel

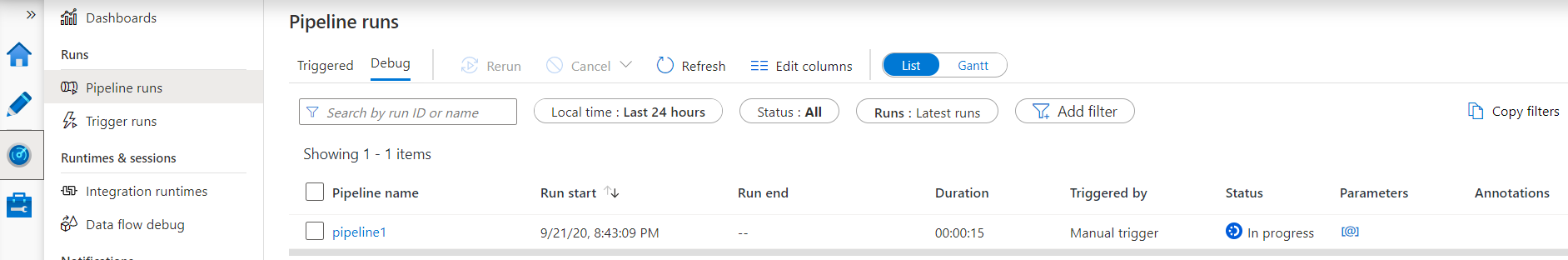


Click Add trigger --> click on Trigger now

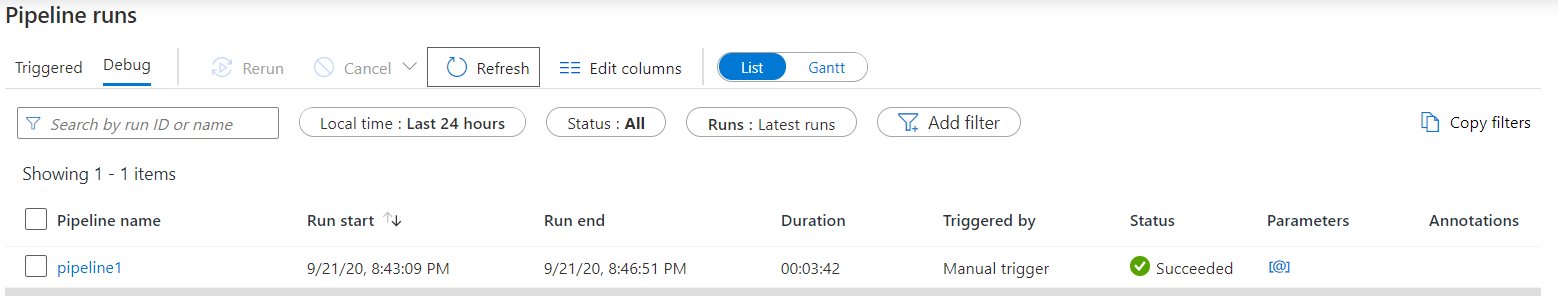


Enter some value in the input parameter and click on OK

To see the progress of the job, click on Monitor menu item on the left side and you will be the status if your job running. You click on Refresh icon.



When don, you should see it as shown below:



You can try various options



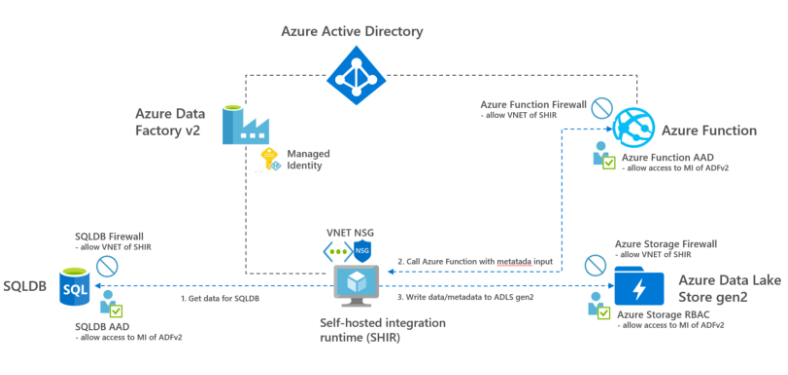
For example, you can try different options in Select cluster field. You can try Worker options and number of Workers nodes.

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

ADF v2.0 Pipeline Security/Authentication

Sunday, September 13, 2020

10:42 AM



Azure Data Factory v2.0 Architecture

**ADF v2.0 and related components Authentication and Authorization**

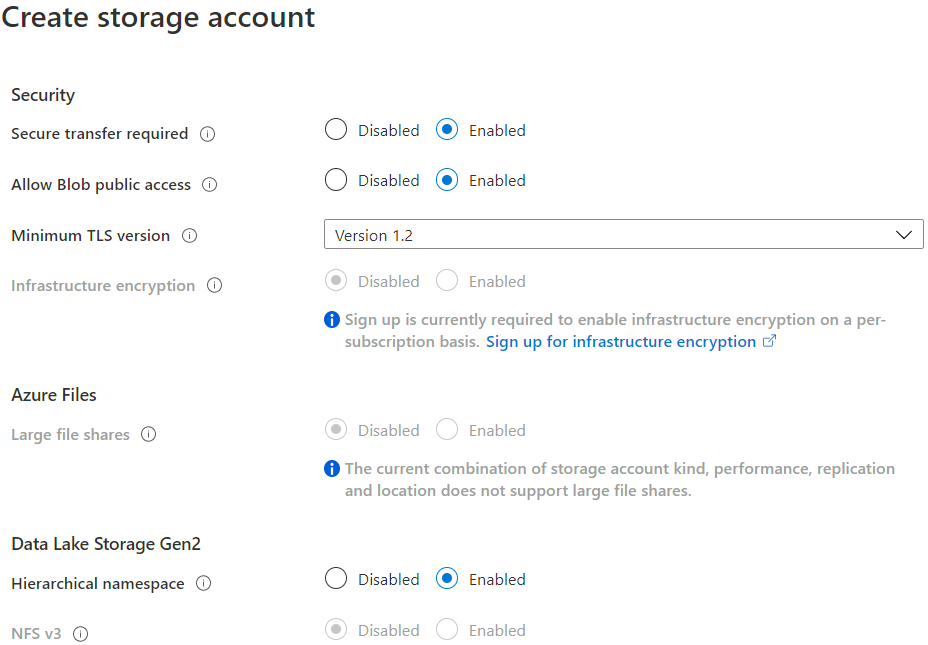
**Three ways:**

1. Account key
2. Service principal
3. Managed Identity

**Account key**

Create Azure Data Factory (ADFv2) instance

Create an ADLS gen2 storage account. Be sure to enable Data Lake Storage Gen2 option



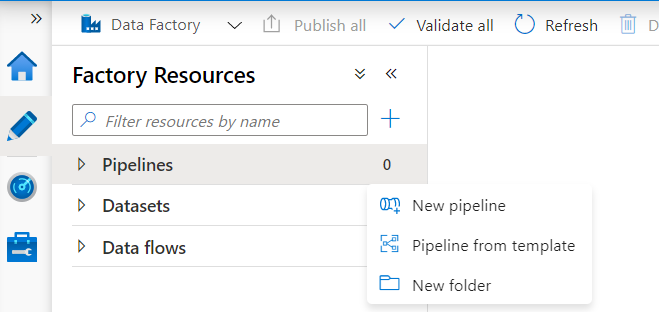
Create two (source and target) folders under Containers

Upload a file (any type, e.g., .txt) to the source folder

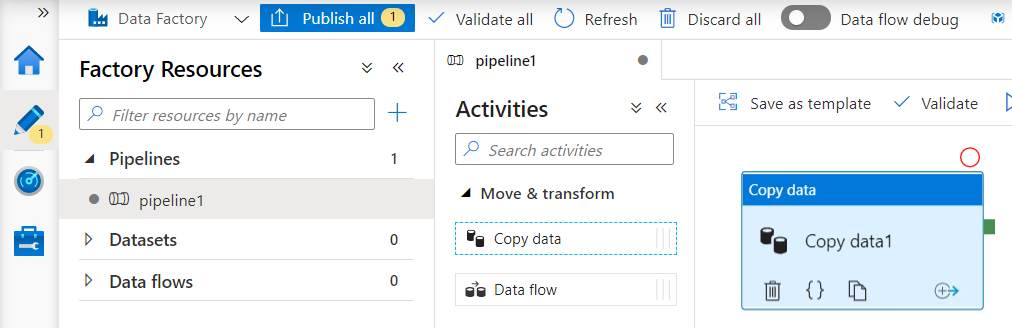
Create an instance of Azure Data factory (ADFv2)

Click on Author & Monitor (in the middle of the page)

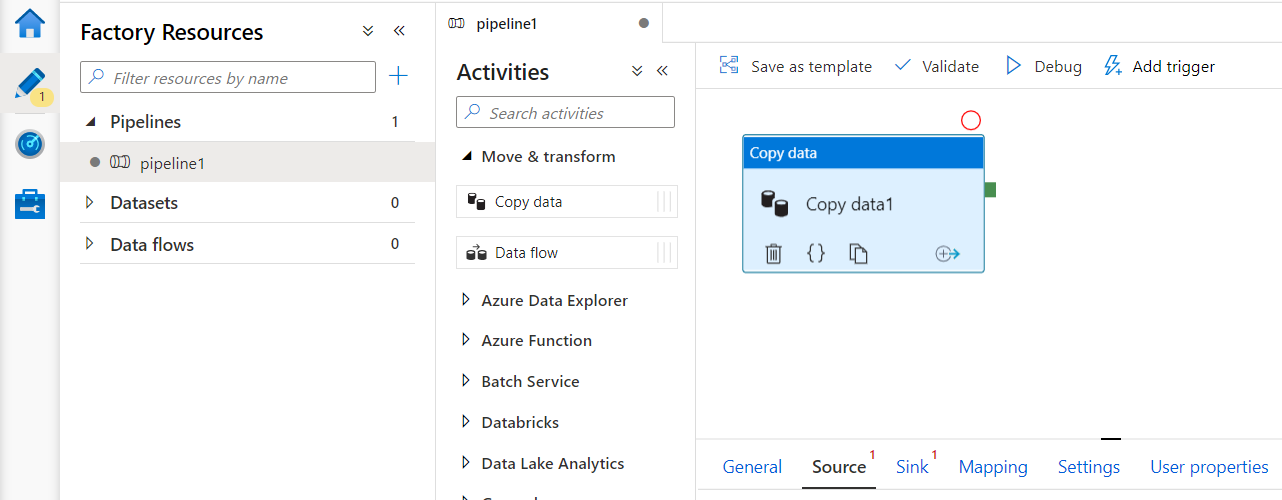
Click on Author --> Pipelines --> New pipeline



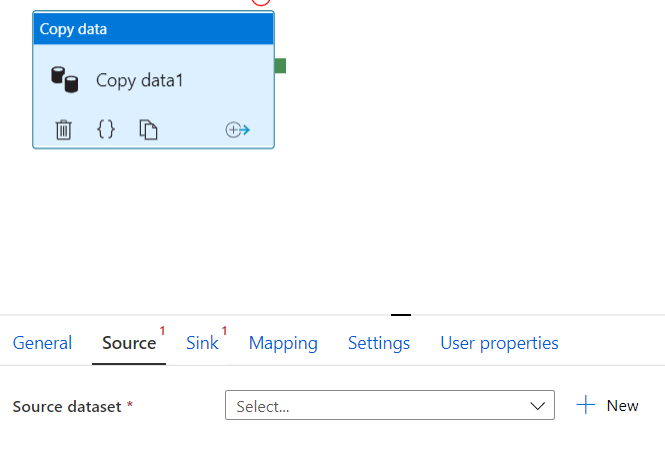
Under Activities, click on Move & transform, select Copy data activity, drag and drop it on to the canvas



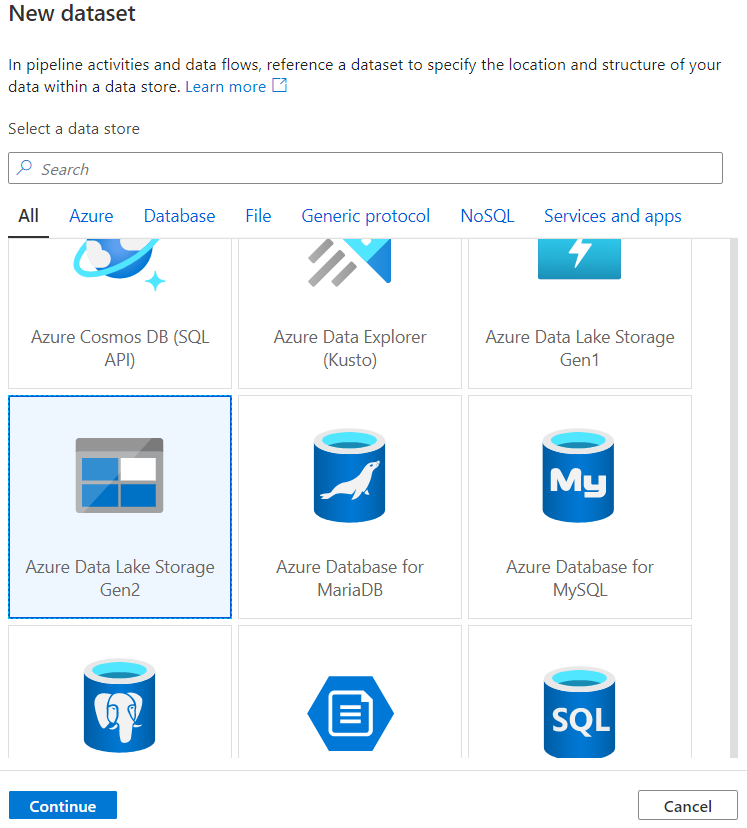
Click on Copy data activity, the bottom half of the canvas will have new menu items



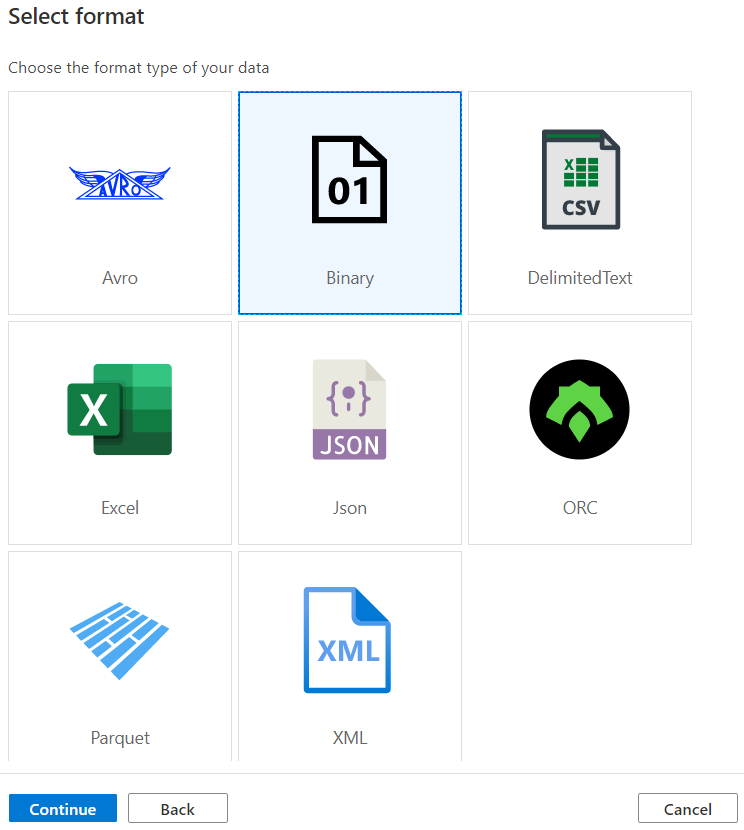
Click Source menu item



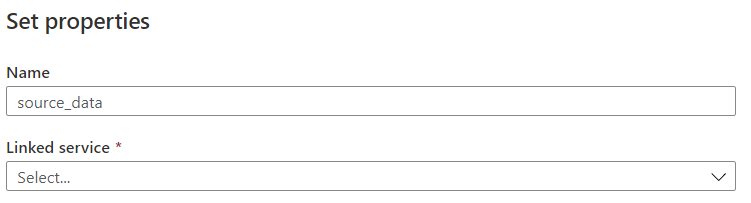
Click on + New



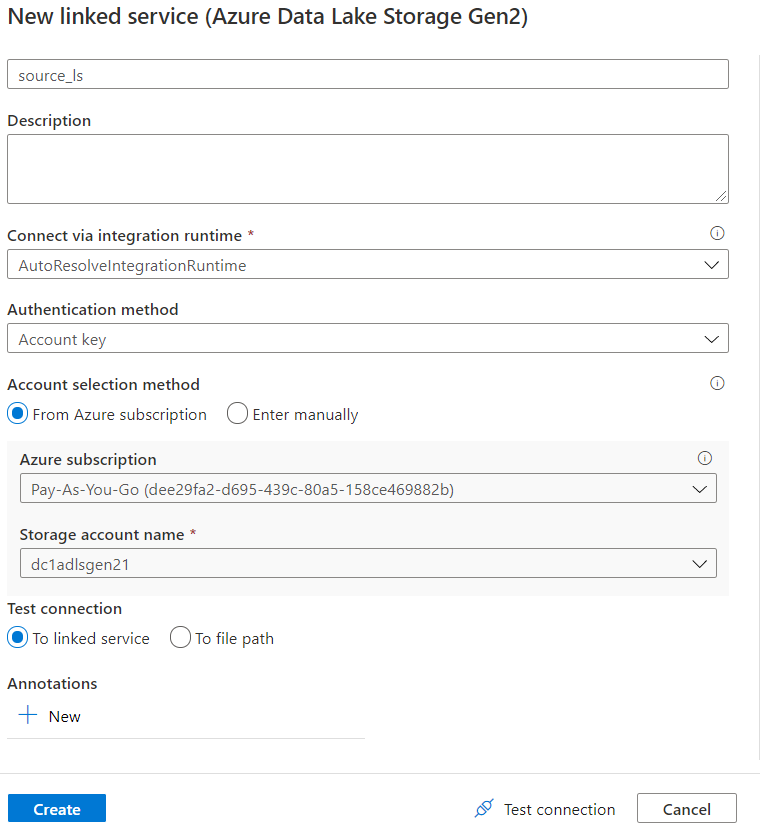
Click on Azure Data Lake Storage Gen2 and click on Continue



Click on Binay (don't worry about the data type, for now) and click on Continue

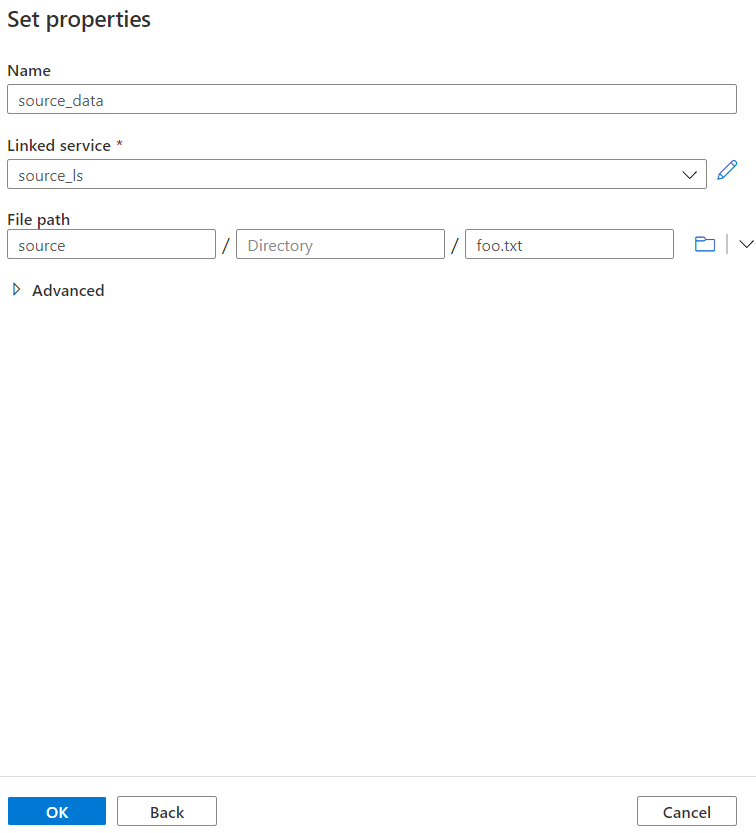


Enter a meaningful name in the Name and select New in the Linked Service



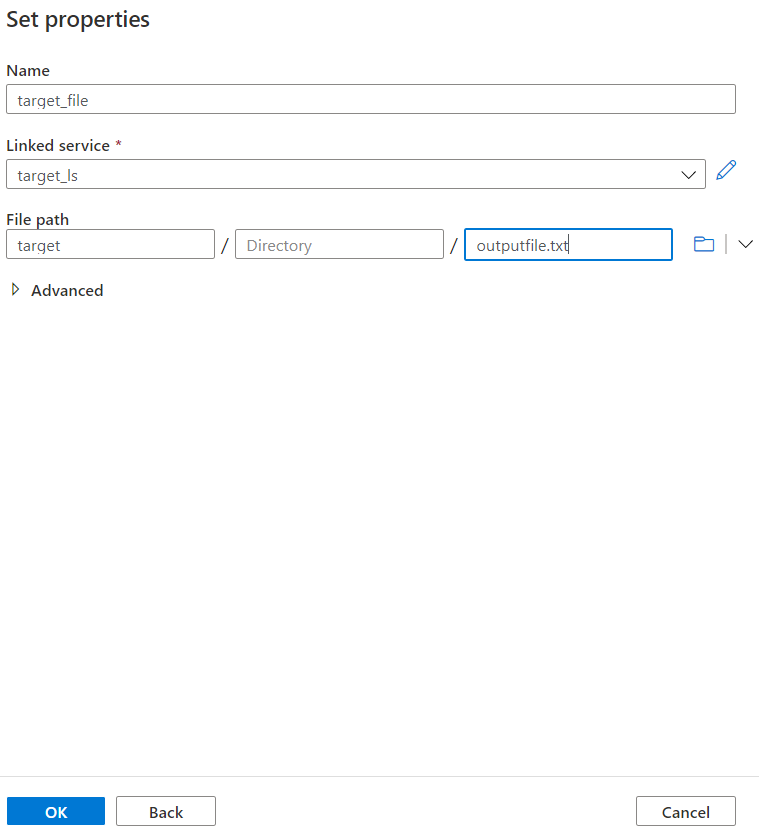
Enter appropriate values in the settings and click on Test connection

If Test connection successful, click on Create



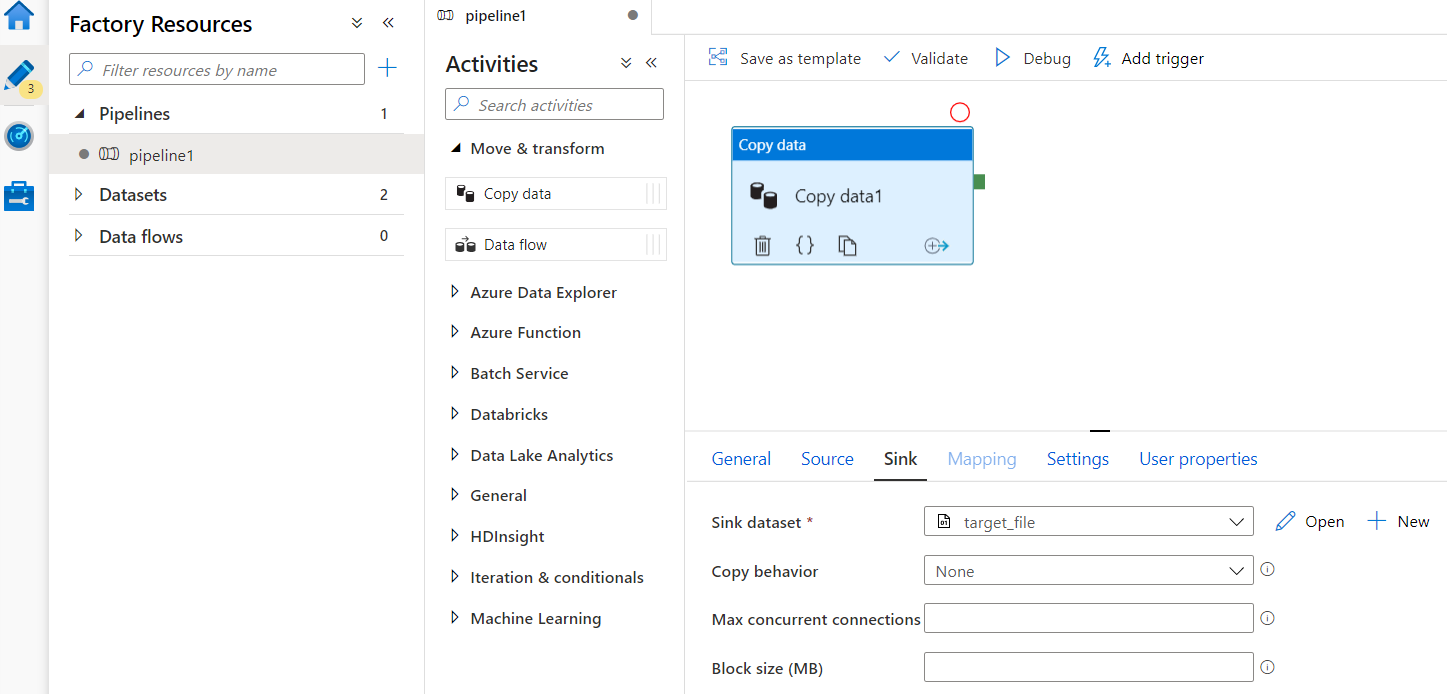
Click on the folder icon and select the source file and click on OK

For the Sink option, do the same steps, in the Set properties, pick the target folder and manually enter the output file name (output.txt, for example)



Click on OK

When done, click on Debug and watch results:

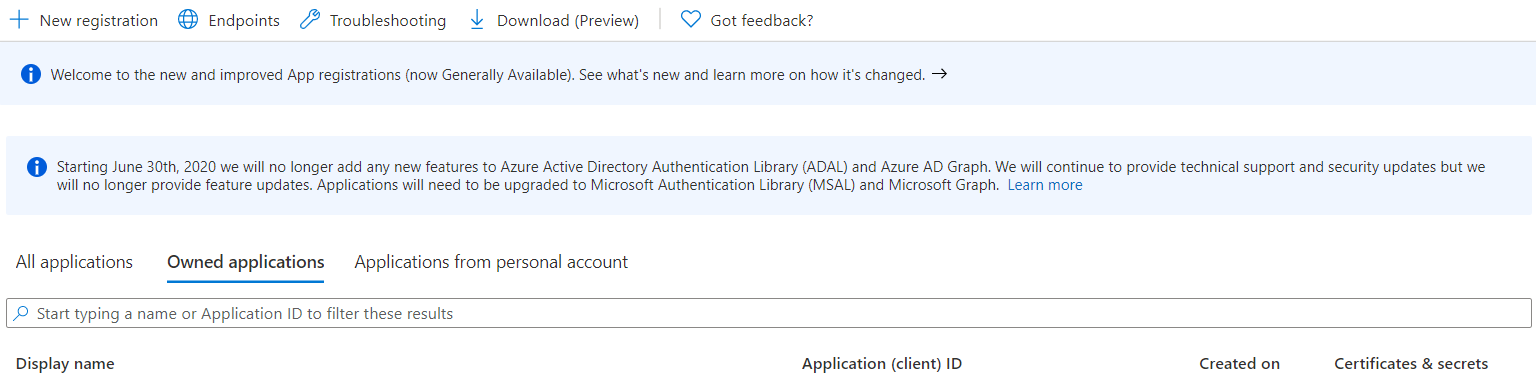


If successful, verify the existence of output.txt file in the target folder in the ADLS Gen2 storage account.

**Service Principal**

Create service principal

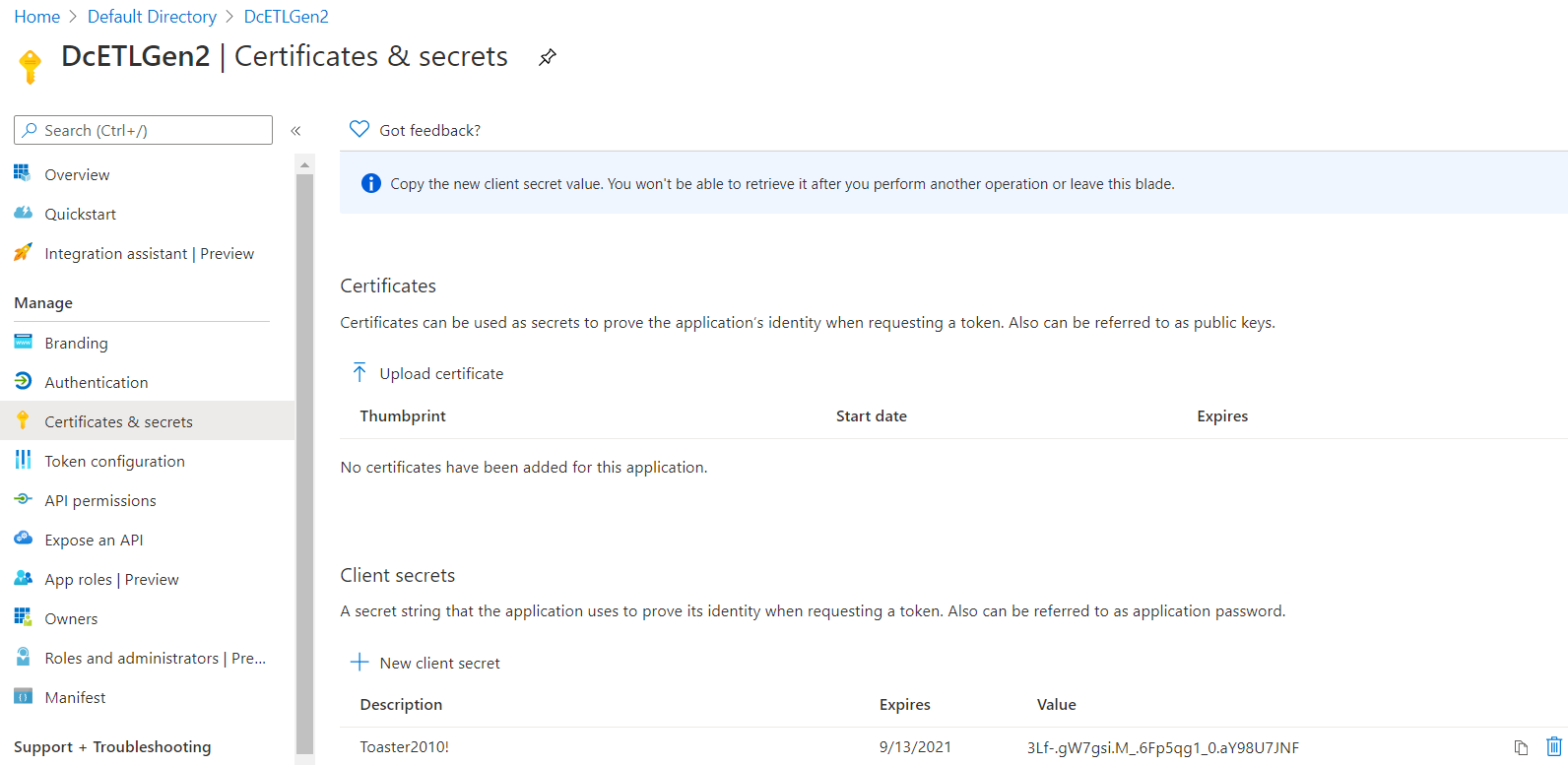
Go to Azure Active Directory, click on App Registrations



Click on + registration



Add Description and Expire click on Add button



Click on Copy icon at the bottom, just before the Delete button.

Save the copied Secret to a separate storage

Copy

Display name: DcETLGen2

Application (client) ID: dbe739d3-f17f-4130-a7d9-45cea942b84d

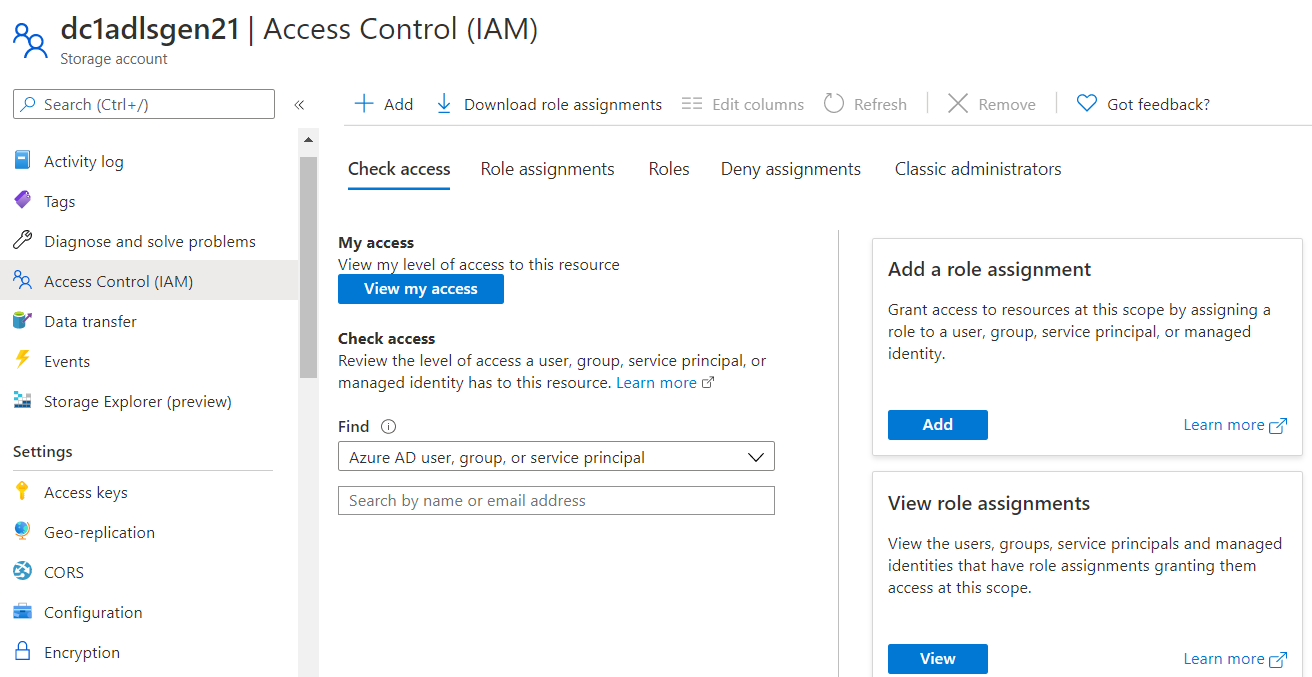
Directory (tenant) ID: 6bdf9621-fbe5-401d-9c2e-8c9ed7555112

Object ID: 5009ffd5-ddba-480e-be77-37f86734a17b

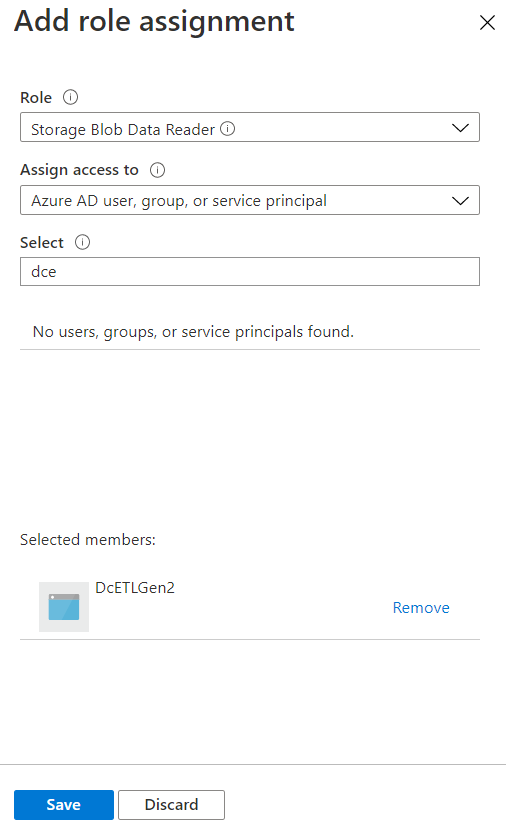
Password: <*password*>

Go to the storage account that you created earlier

At the account level, click on Access Control (IAM)



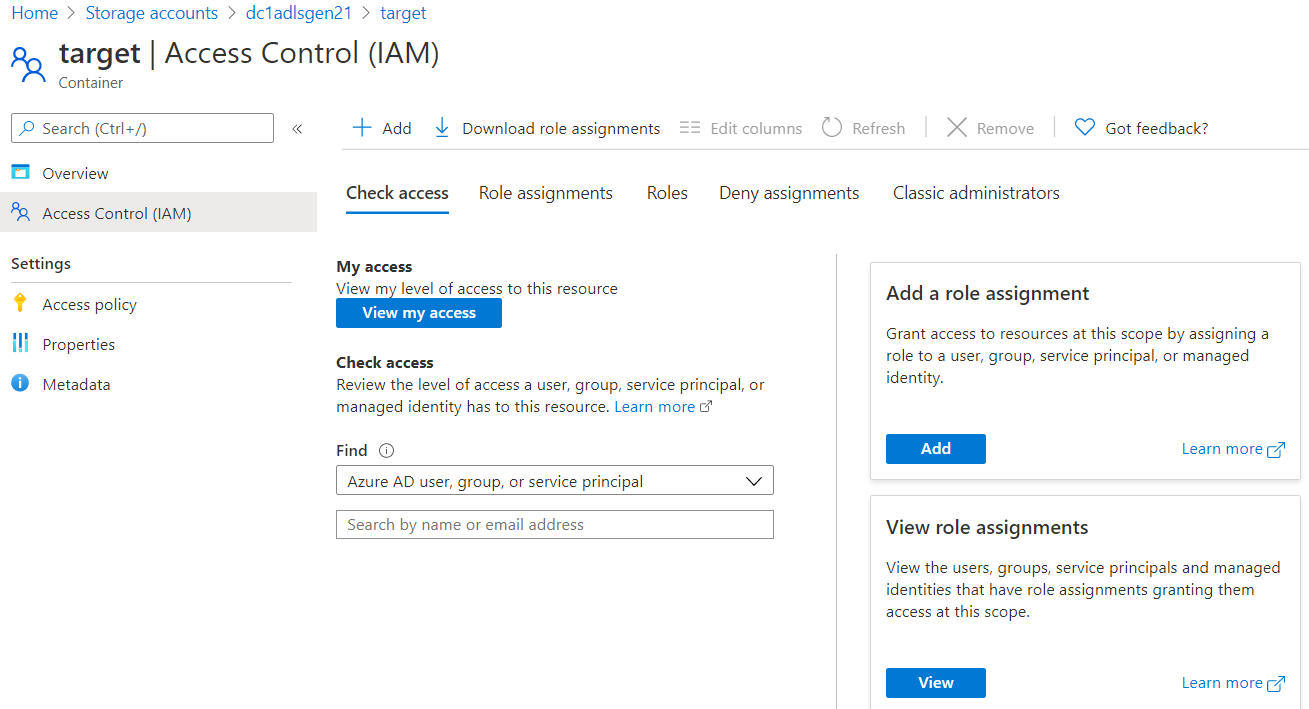
Click on + Add, click on Add in the Add a role assignment tile



Select Storage Blob Data reader in the Role field

Search for the App registration you created in the App registration of the Azure Active Directory, click on it

Click on the Save button



Select on the target container in the containers

Click on + Add, click on Add button in the Add a role assignment tile

Select Storage Blob Data Contributor in the Role field

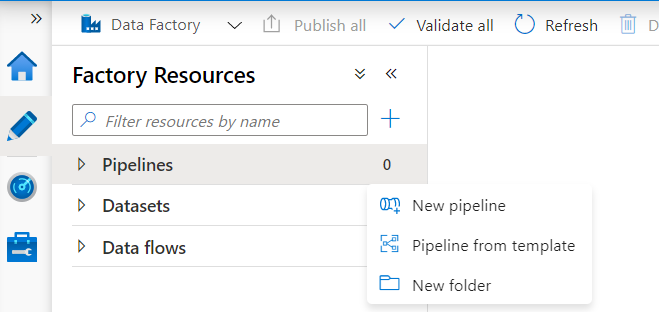
Search for the App registration you created in the App registration of the Azure Active Directory, click on it

Click on the Save button

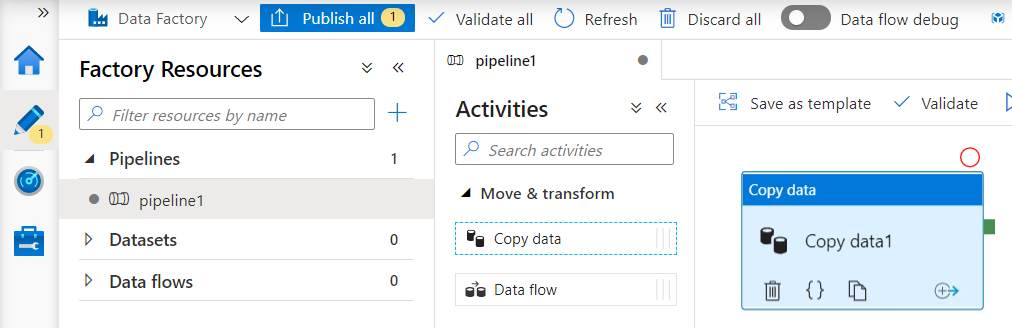
Create an instance of Azure Data factory (ADFv2)

Click on Author & Monitor (in the middle of the page)

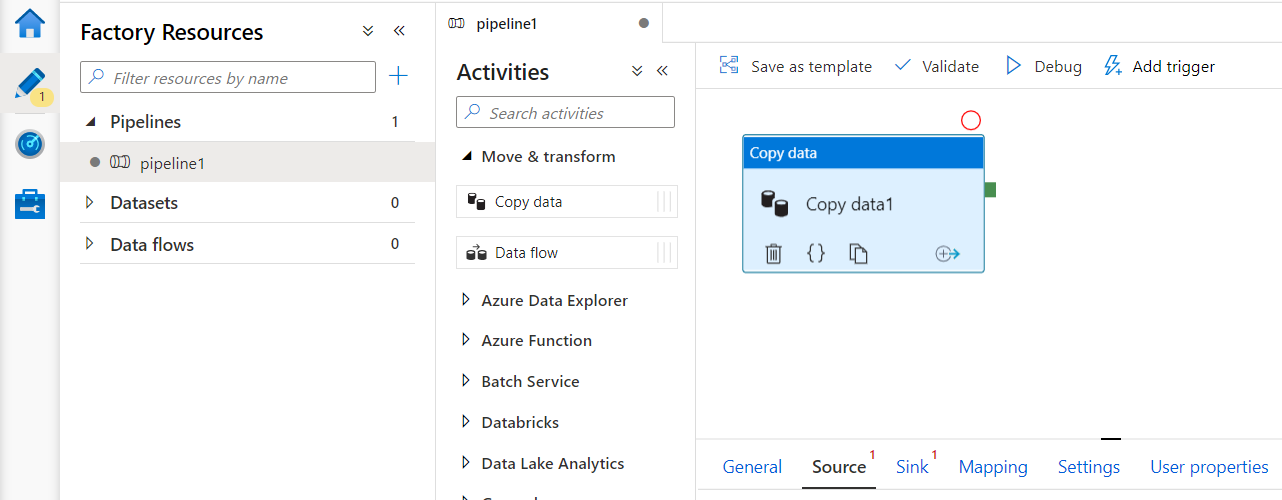
Click on Author --> Pipelines --> New pipeline



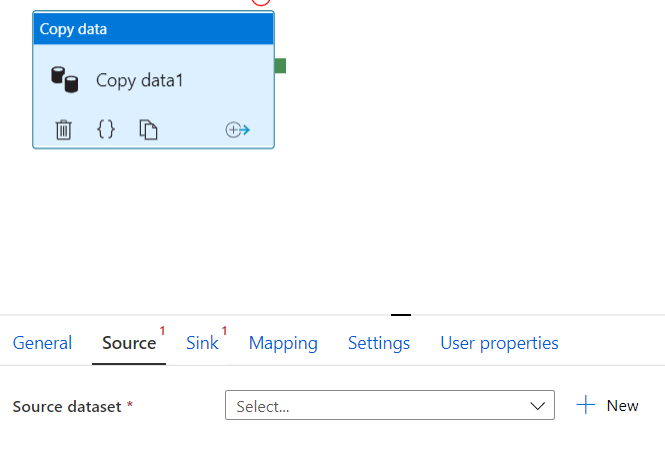
Under Activities, click on Move & transform, select Copy data activity, drag and drop it on to the canvas



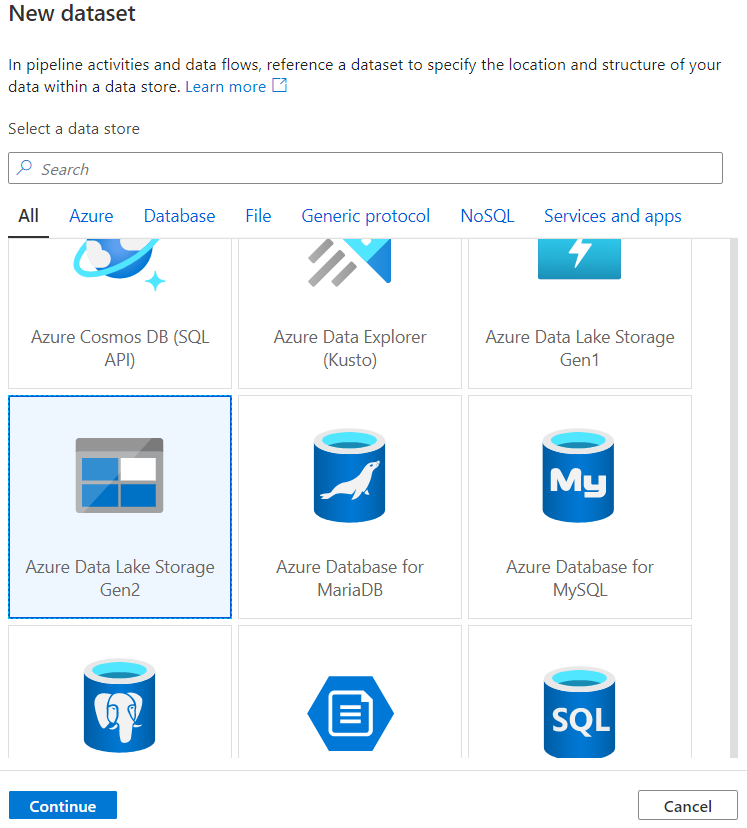
Click on Copy data activity, the bottom half of the canvas will have new menu items



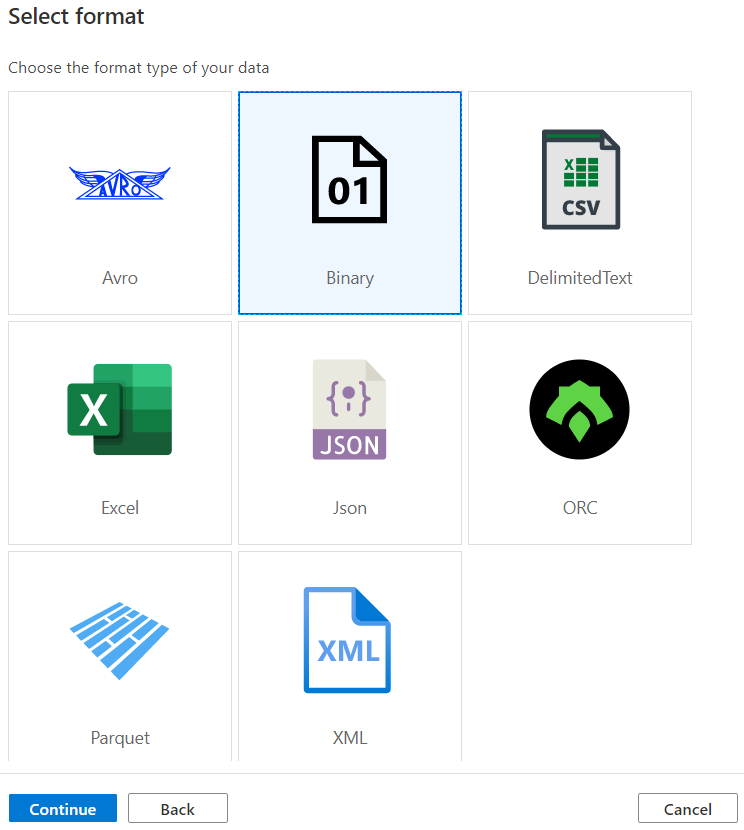
Click Source menu item



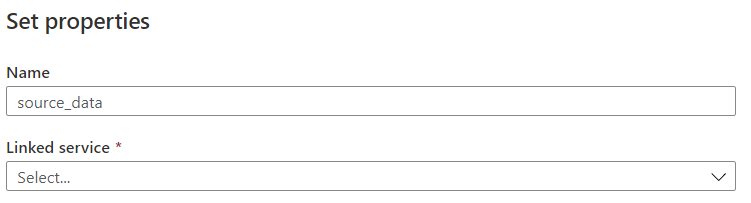
Click on + New



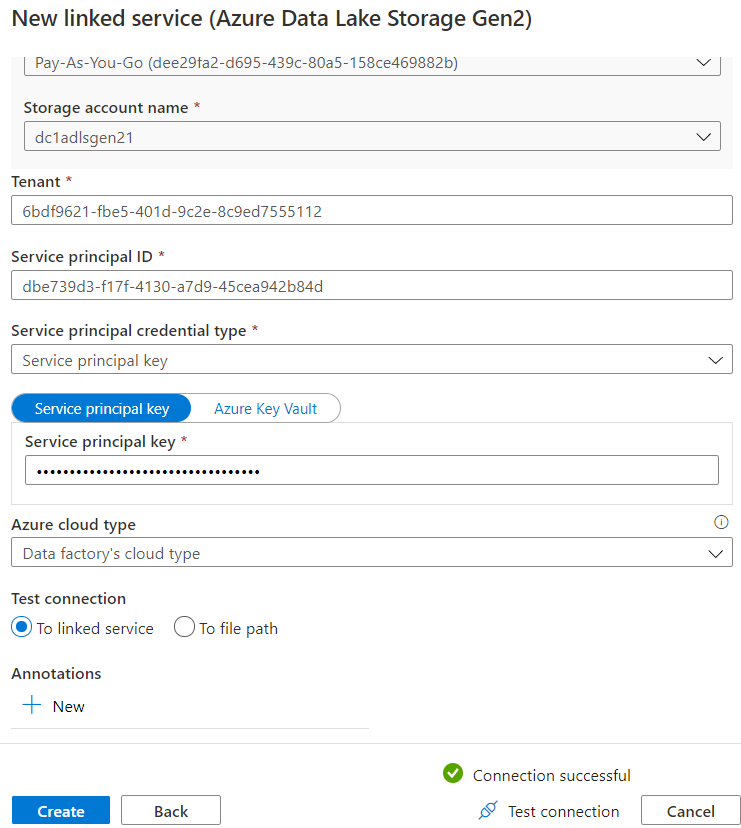
Click on Azure Data Lake Storage Gen2 and click on Continue



Click on Binary (don't worry about the data type, for now) and click on Continue



Enter a meaningful name in the Name and select New in the Linked Service



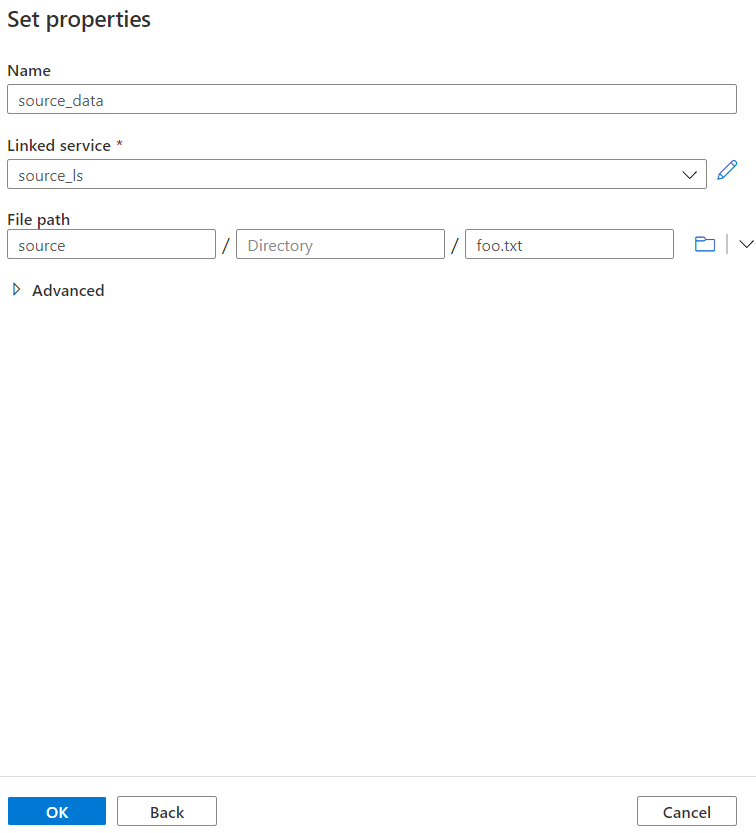
Enter appropriate values in the settings and click on Test connection

Note: Application (client) ID will be entered in Service principal ID

Directory (tenant) ID will be entered in Tenant field

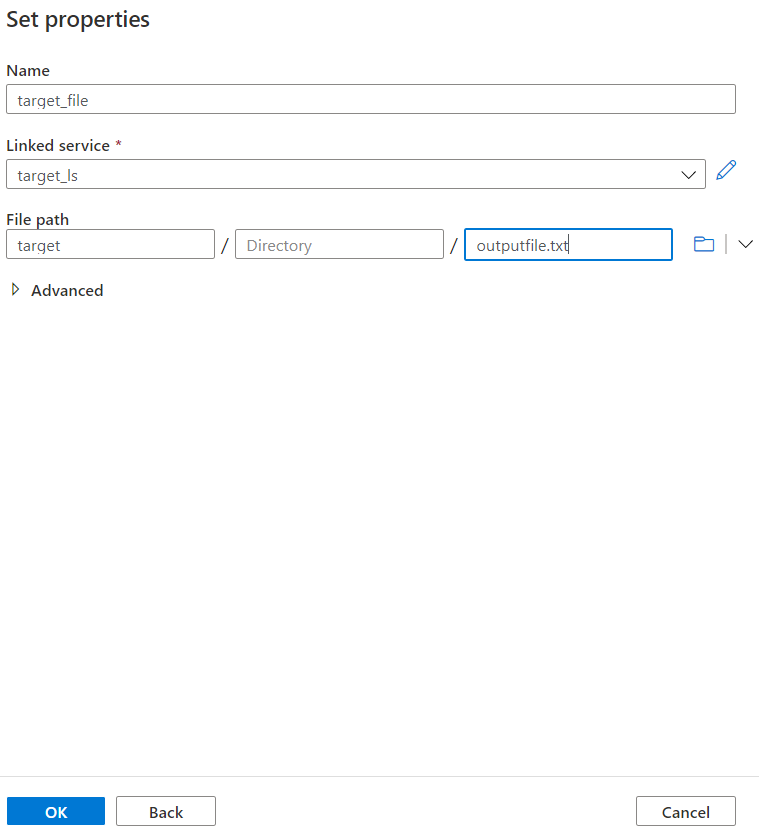
Secret/password will be entered in the Service principal key field

If Test connection successful, click on Create



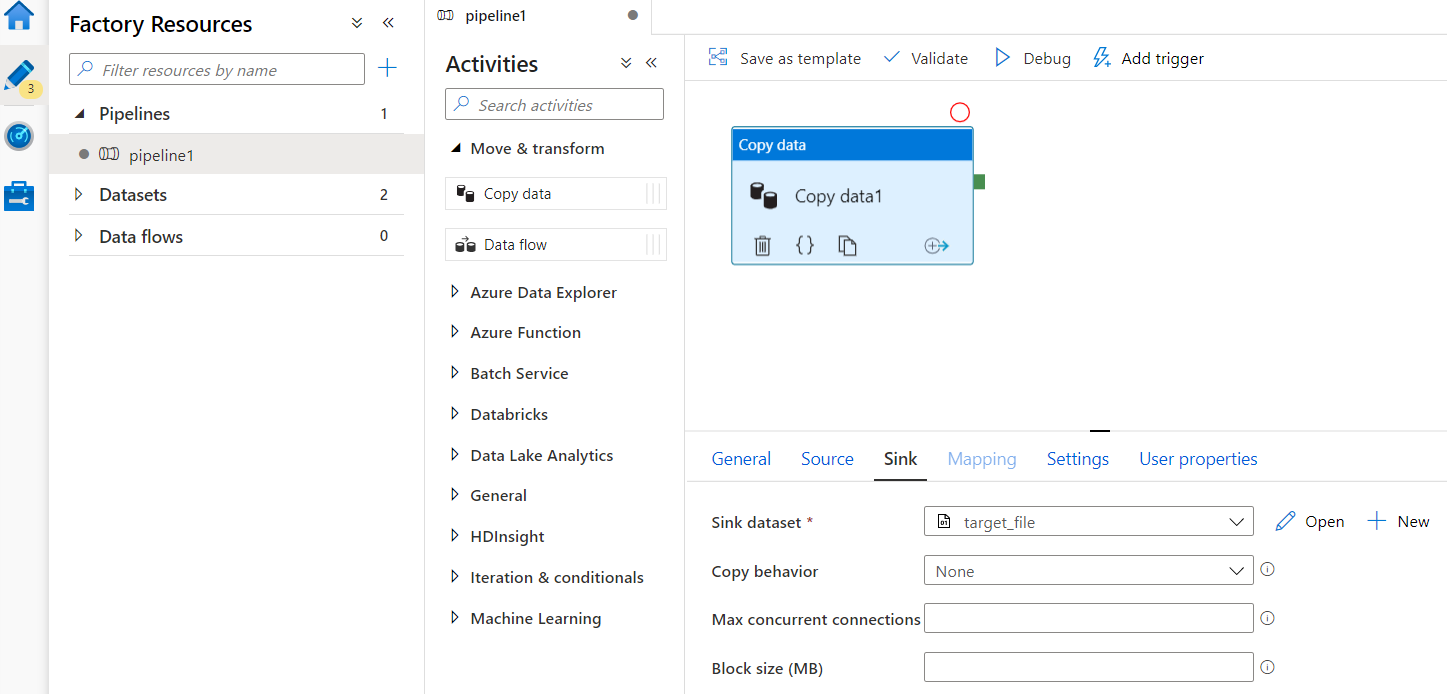
Click on the folder icon and select the source file and click on OK

For the Sink option, do the same steps, in the Set properties, pick the target folder and manually enter the output file name (output.txt, for example)



Click on OK

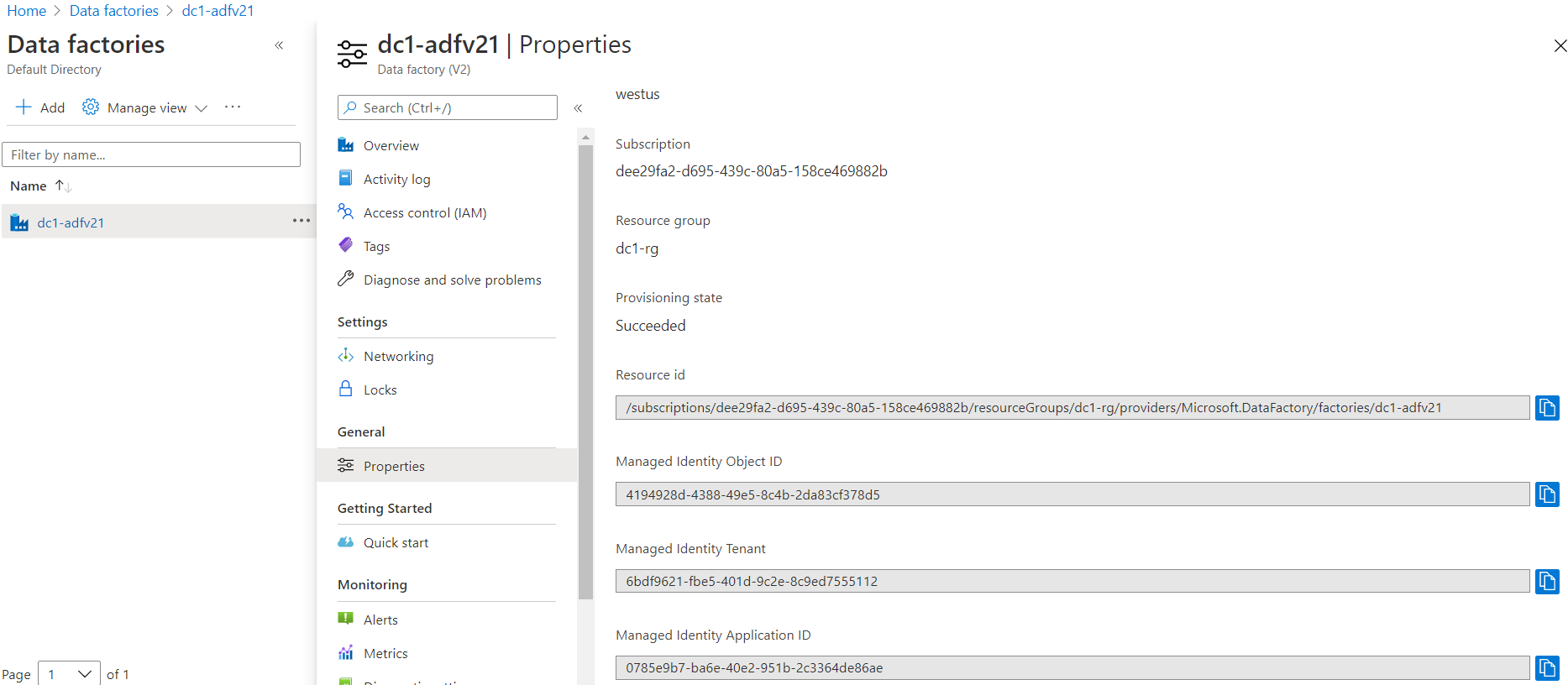
When done, click on Debug and watch results:



If successful, verify the existence of output.txt file in the target folder in the ADLS Gen2 storage account.

**Managed Identity**

Create an instance of Azure Data factory (ADFv2)

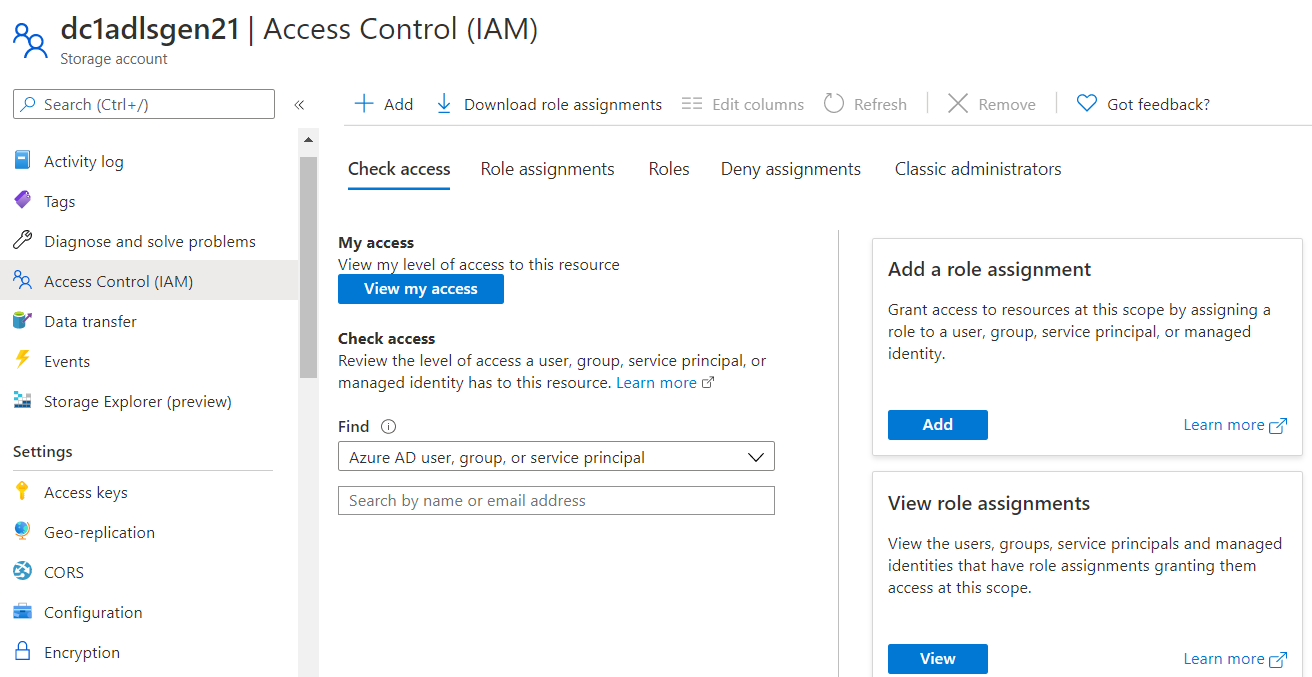


Click on Properties under General category

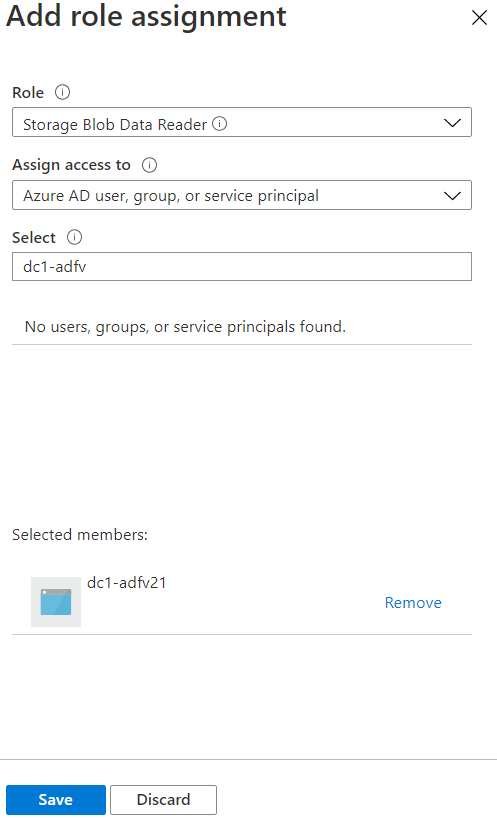
Copy Resource id, Managed identity Object ID, Managed identity Tenant, and Managed identity Application ID

Go to the storage account that you created earlier

At the account level, click on Access Control (IAM)



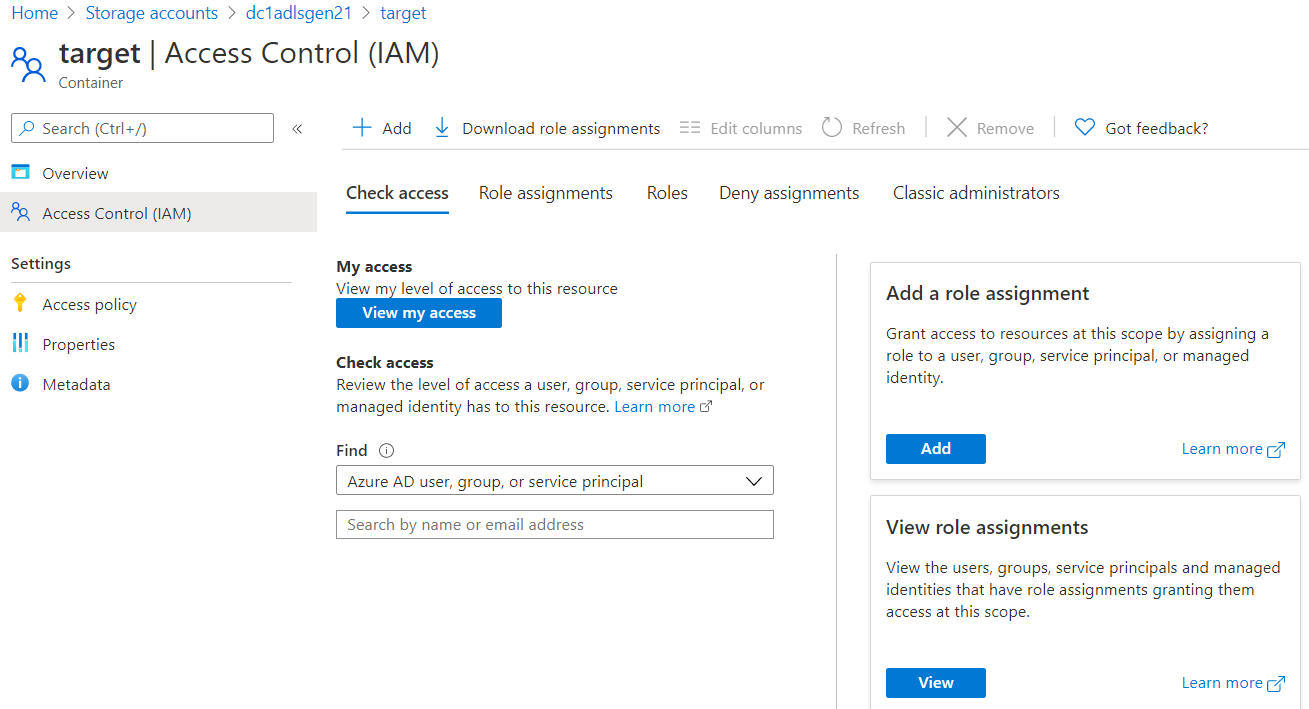
Click on + Add, click on Add in the Add a role assignment tile

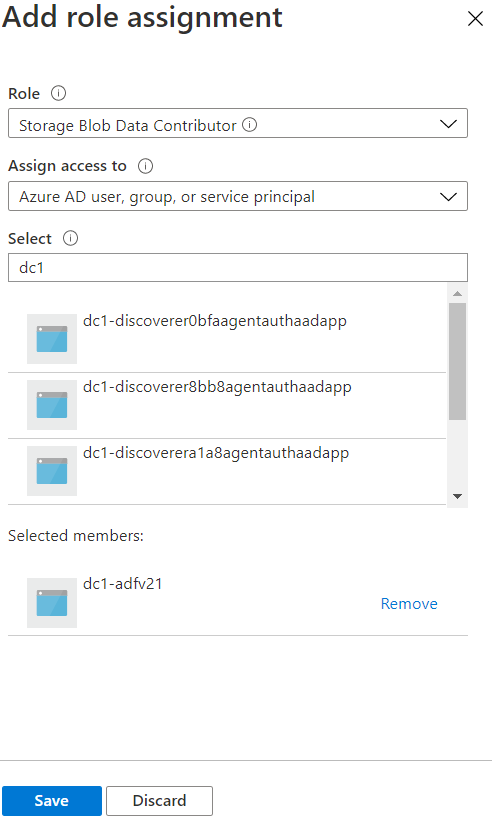


Select Storage Blob Data reader in the Role field

Enter the name of the Azure Data Factory instance, e.g., dc1-adfv21

Click on the Save button





Select on the target container in the containers

Click on + Add, click on Add button in the Add a role assignment tile

Select Storage Blob Data Contributor in the Role field

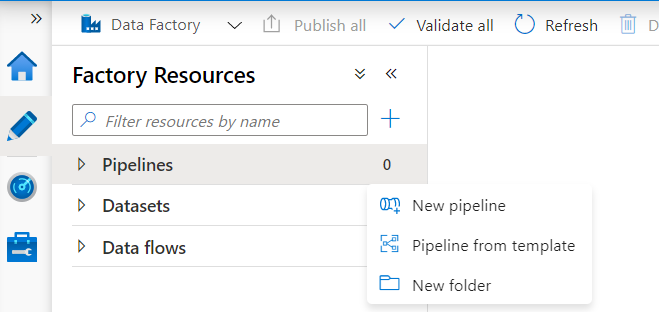
Enter the name of the Azure Data Factory instance, e.g., dc1-adfv21

Click on the Save button

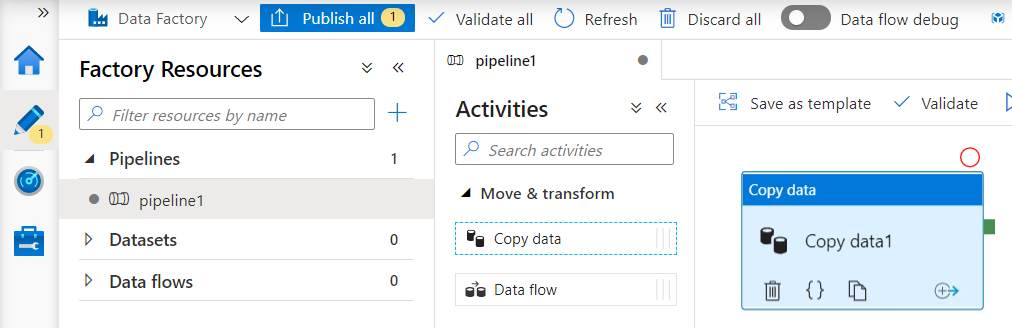
Go to the Azure Data factory v2 instance

Click on Author & Monitor (in the middle of the page)

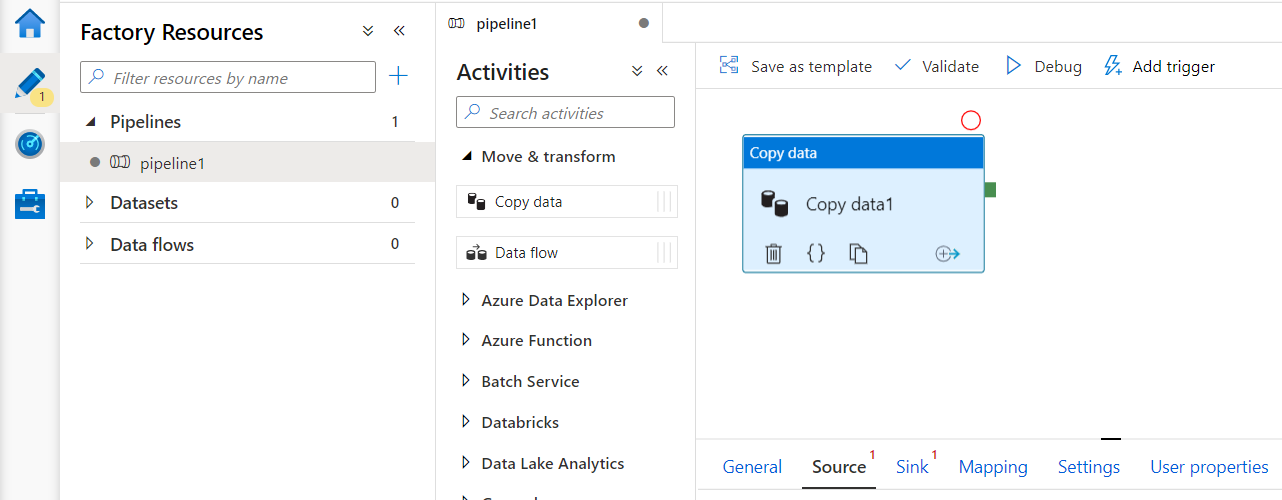
Click on Author --> Pipelines --> New pipeline



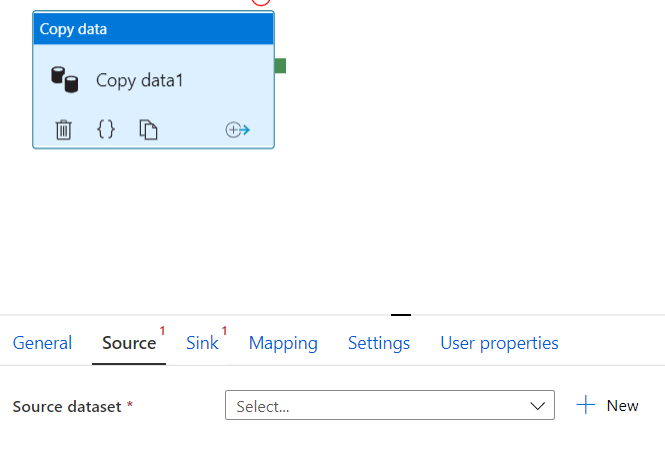
Under Activities, click on Move & transform, select Copy data activity, drag and drop it on to the canvas



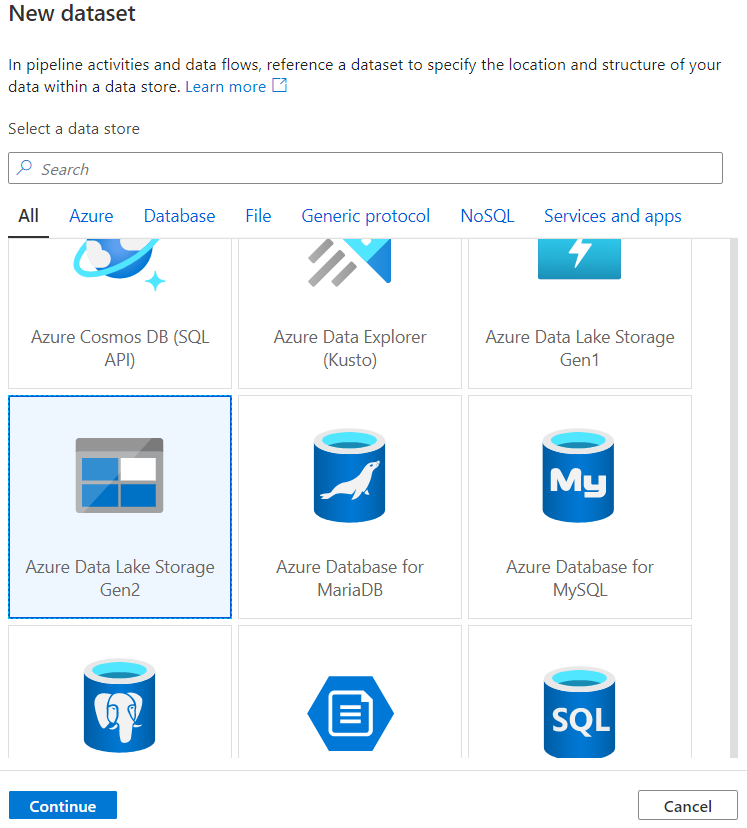
Click on Copy data activity, the bottom half of the canvas will have new menu items



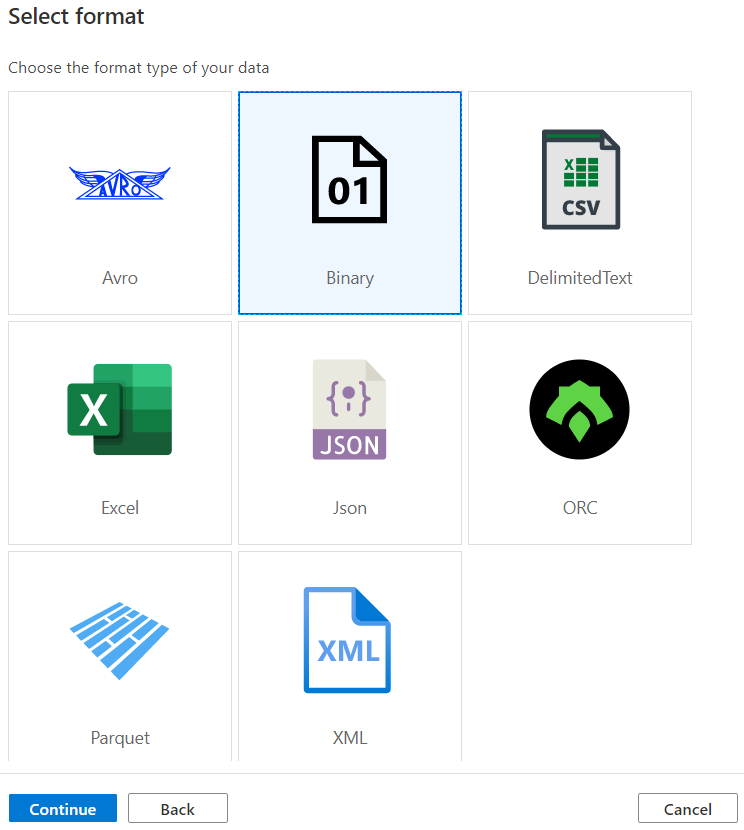
Click Source menu item



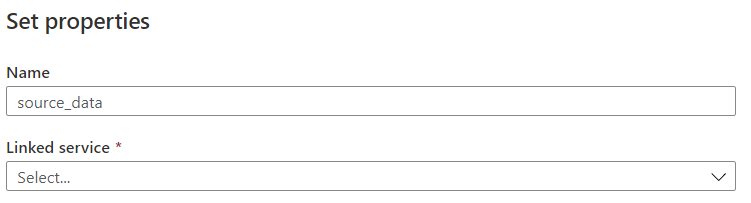
Click on + New



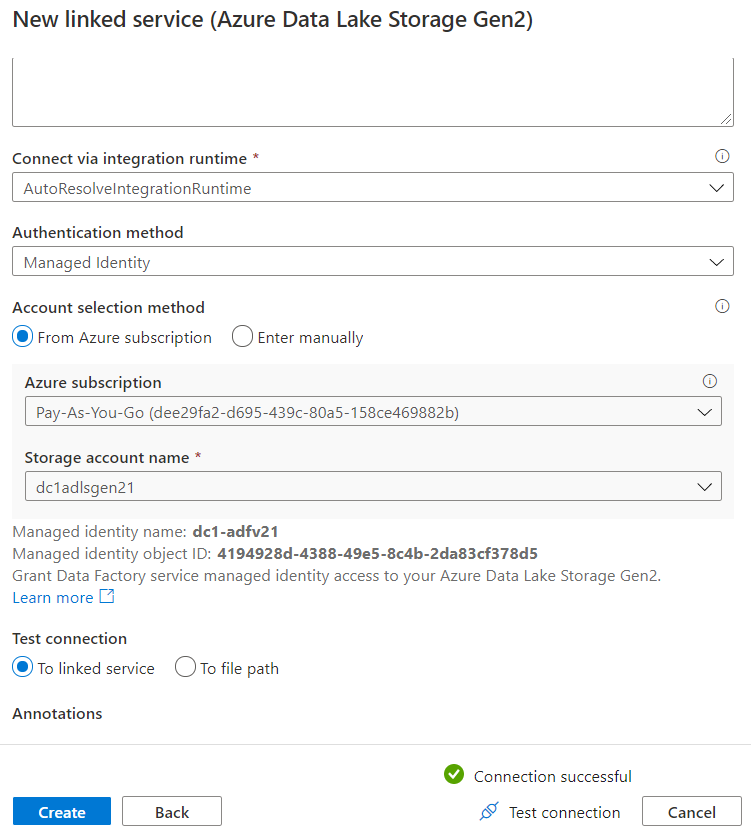
Click on Azure Data Lake Storage Gen2 and click on Continue



Click on Binary (don't worry about the data type, for now) and click on Continue

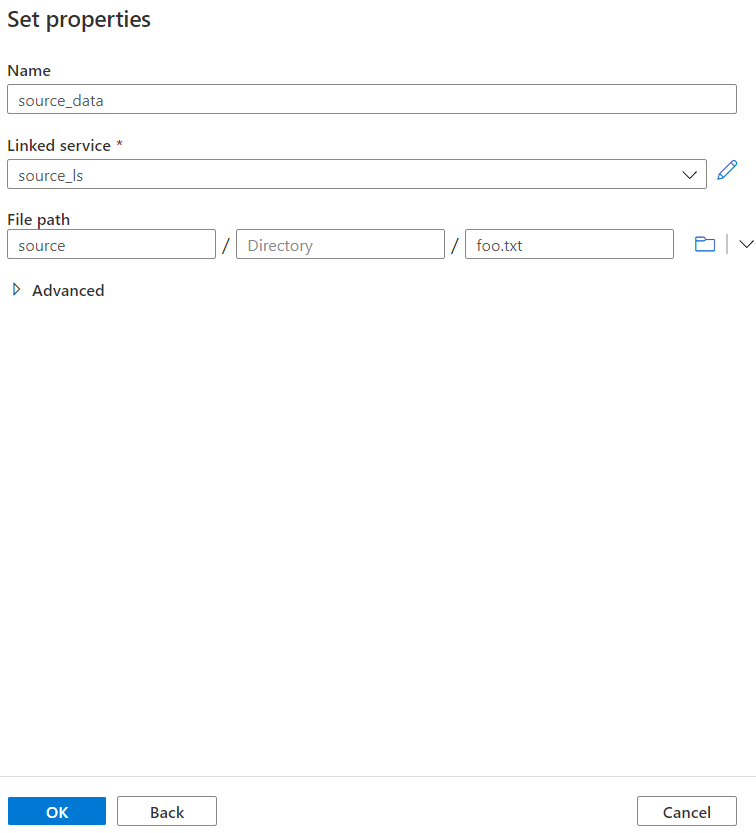


Enter a meaningful name in the Name and select New in the Linked Service



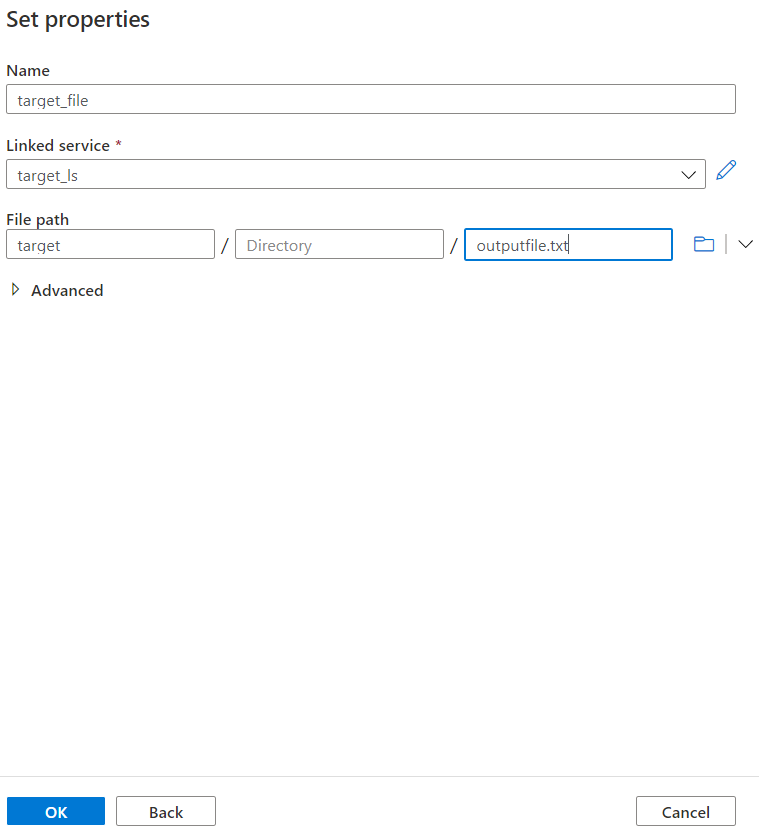
Enter appropriate values in the settings and click on Test connection

If Test connection successful, click on Create



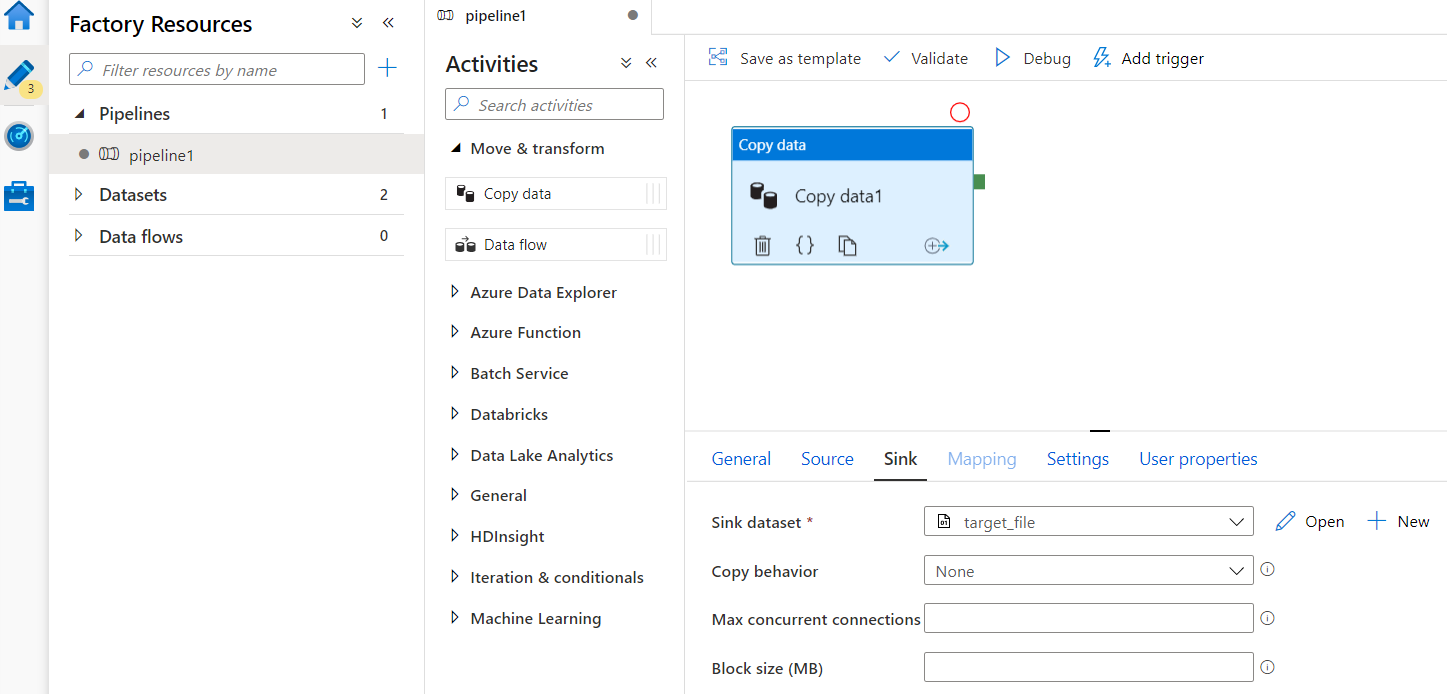
Click on the folder icon and select the source file and click on OK

For the Sink option, **do the same steps**, in the Set properties, pick the target folder and manually enter the output file name (output.txt, for example)



Click on OK

When done, click on Debug and watch results:



If successful, verify the existence of output.txt file in the target folder in the ADLS Gen2 storage account.

**References:**

* How to secure your Azure Data Factory pipeline <<https://towardsdatascience.com/how-to-secure-your-azure-data-factory-pipeline-e2450502cd43>>
* Security considerations for data movement in Azure Data Factory <<https://docs.microsoft.com/en-us/azure/data-factory/data-movement-security-considerations>>
* Azure security baseline for Azure Data Factory <<https://docs.microsoft.com/en-us/azure/data-factory/security-baseline>>
* AZURE DATA FACTORY SECURITY & AUTHENTICATION <https://www.industry-era.com/images/pdf/Azure%20data%20Factory-Security.pdf>
* Azure Data Factory Interview Questions and Answer <<https://intellipaat.com/blog/interview-question/azure-data-factory-interview-questions/>>

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

ADF v2.0 Event based Pipeline Trigger

Thursday, October 8, 2020

1:11 PM

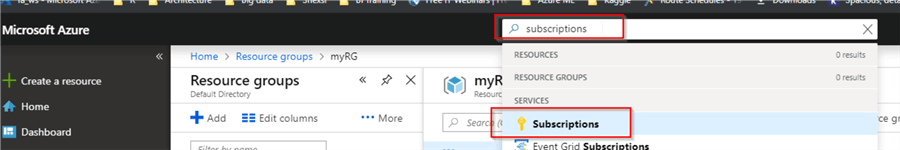
**Prerequisite**

Be sure to Register Azure Event Grid resource provider to your subscription before creating an event trigger

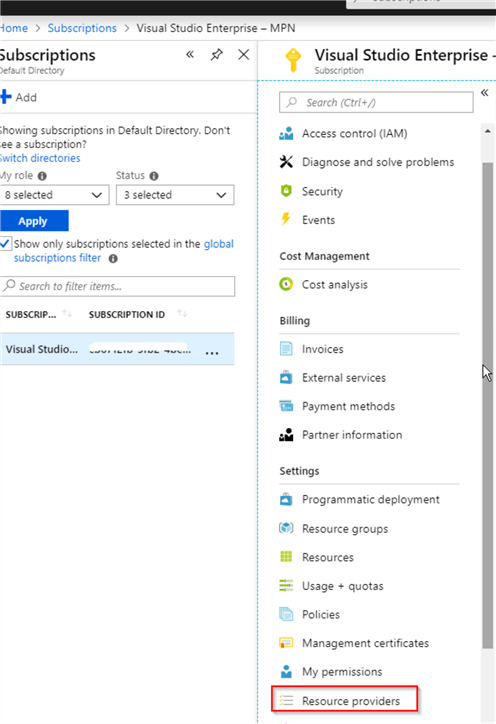
Event-based triggers in Azure Data Factory

Event-based triggers start pipelines in response to file deposit and removal events on Azure Blob Storage. This feature leverages [Azure Event Grid](https://docs.microsoft.com/en-ca/azure/event-grid/overview) functionality, so we need to follow the below steps to enable Azure Event Grid for our subscription:

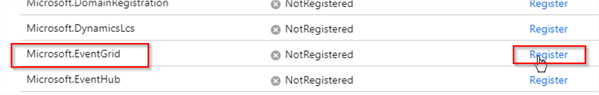
* Open Azure Portal, type 'subscriptions' in the top search box and select 'Subscriptions' menu:



* Choose your subscription name and scroll down the menu panel to select 'Resource providers' command:

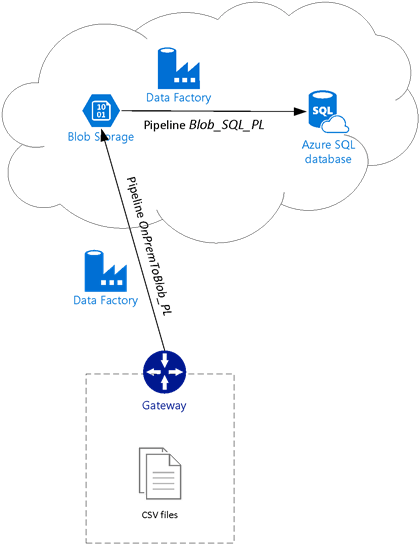


* Scroll down the provider list to find 'Microsoft.EventGrid' provider and click 'Register' button:



Data flow description in Azure Data Factory

In earlier posts dedicated to file transfer pipelines (see [Transfer On-Premises Files to Azure Blob Storage](https://www.mssqltips.com/sqlservertip/5928/transfer-onpremises-files-to-azure-blob-storage/)), we created a blob storage account, hosting container csvfiles and built pipeline OnPremToBlob\_PL, which transferred CSV files into that container. Then we built pipeline Blob \_SQL\_PL to bring those files from blob storage into Azure SQL Database and executed both pipelines manually. In this exercise, we will create an event trigger to initiate pipeline Blob\_SQL\_PL automatically, in response to file deposit events into the csvfiles container and use pipeline OnPremToBlob\_PL to deposit the CSV files into this container, as shown in this diagram:



The pipeline Blob\_SQL\_PL should be kicked-off automatically, in response to each file drop event and will transfer data related to that specific file. Therefore, in our case we should see three executions, matching the count of CSV files.

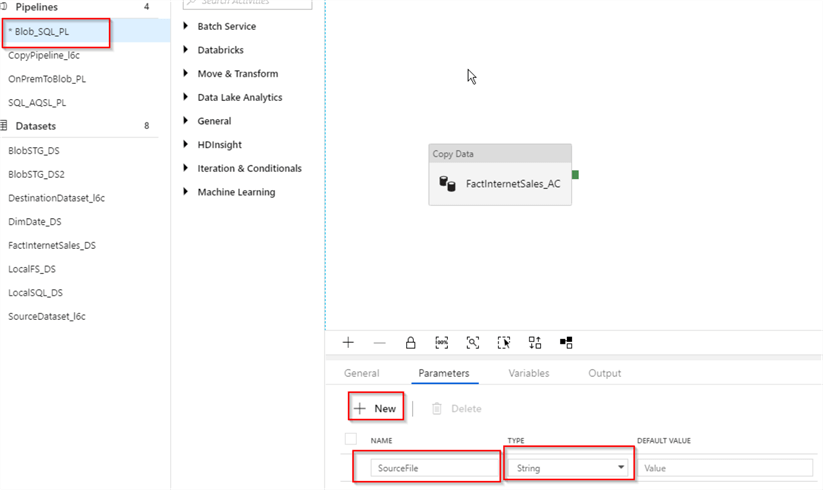
The pipeline Blob\_SQL\_PL we have built so far, transfers data for all CSV files located in its source container. However, I want to make this trigger a bit smarter to initiate each execution with specific parameters, indicating which file has caused this execution, so the pipeline transfers data for the related file only.

So, let's parameterize pipeline Blob\_SQL\_PL and its source dataset, before we proceed further.

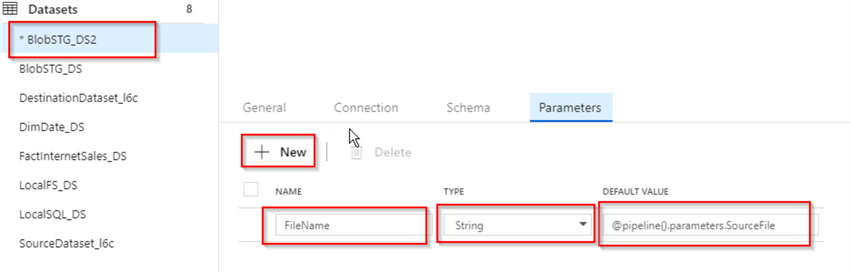
Parameterization of the pipeline and its dataset

We learned how to add pipeline parameters and tie them to trigger settings in the previous posts. This time we will parameterize the pipeline and its source dataset and link these parameters together, please see below required steps:

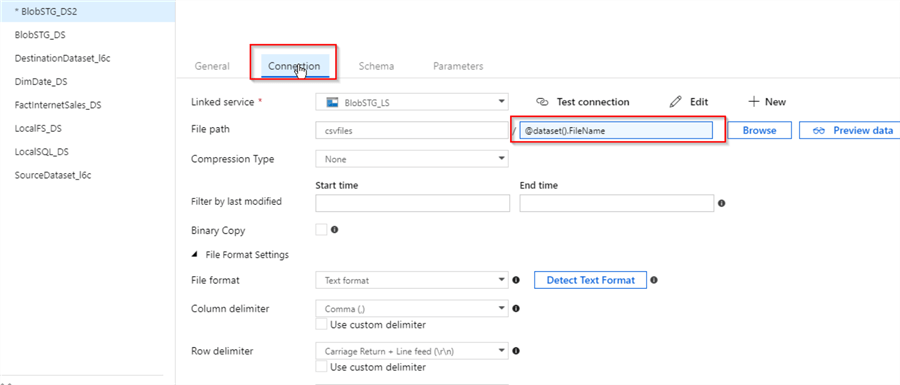
* Select pipeline Blob\_SQL\_PL, switch to 'Parameters' tab and add string type parameter SourceFile:



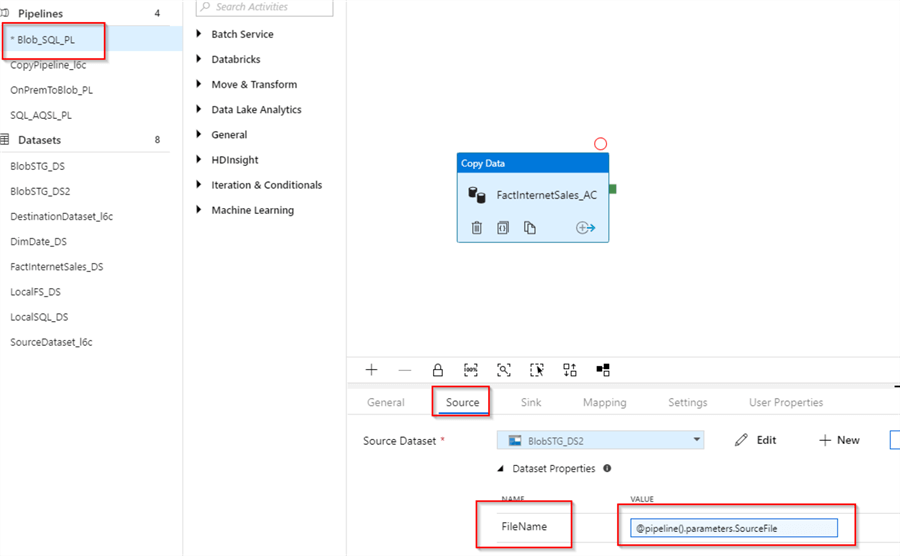
* We can refer to the pipeline parameters using expression like @pipeline().parameters.XXXX, where the last part of the expression represents the parameter name. Accordingly, we can use an expression @pipeline().parameters.SourceFile to refer to SourceFile parameter we created in the previous step. So, let's select dataset BlobSTG\_DS2, switch to the 'Parameters' tab, add string type parameter FileName and assign the expression @pipeline().parameters.SourceFile as the default value for this parameter:



* To ensure that each execution instance of the pipeline transfers only the file specific to that execution, we need to pass the source file name from the trigger via the dataset parameter we just created. To refer to dataset parameter FileName, we can use an expression @dataset().FileName, so let's switch to the Connection tab and add that expression into the file name related part of the 'File path' text box:



* Next, to link the dataset parameter FileName to the pipeline parameter SourceFile, let's select activity FactInternetSales\_AC, switch to 'Source' tab and assign expression @pipeline().parameters.SourceFile to the FileName parameter under 'Dataset Parameters' section:

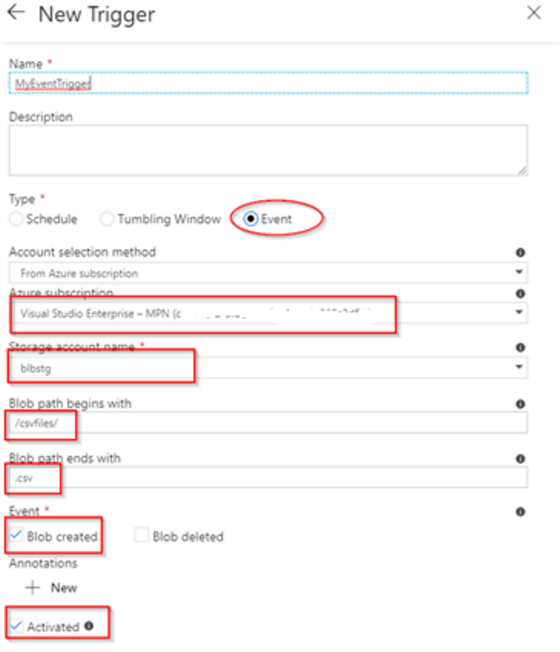


* Finally, let's publish the changes, using the 'Publish All' button.

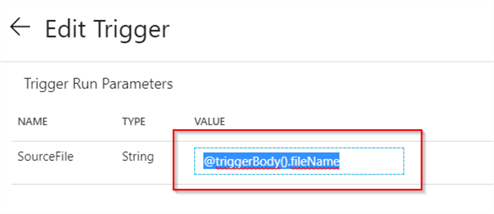
Creating event-based trigger in Azure Data Factory

Now that we have prepared pipeline 'Blob\_SQL\_PL' to receive settings from the trigger, let's proceed with that event trigger's configuration, as follows:

* Select pipeline 'Blob\_SQL\_PL', click 'New/Edit' command under Trigger menu and choose 'New trigger' from drop-down list
* Assign the trigger name ('MyEventTrigger' in this example) and select event trigger type
* Next few steps are related to blob storage where we are expecting the file drops. Select your Azure subscription from drop-down list, as well as storage account name ('blbstg' in this example)
* Specify container name, enclosed by '/' character ('/csvfiles/' in this example) in the 'Blob path begins with' field
* The 'Blob path ends with' field can be used to enter file names, but in our case we want to include all CSV files, located in csvfiles container, so we'll specify only file extension '.csv'
* Check 'Blob created' checkbox, to indicate that trigger needs to be fired in response to file arrival events into the above-mentioned container. Please note that, we could also set file removal as a triggering condition, however that would probably not be suitable for file copy pipelines. Here's how would the 'New Trigger' window looks after all these settings:



* An event based trigger can pass two different system variables to the related pipeline- @triggerBody().fileName and @triggerBody().folderPath, which represents the name and folder of the file. In this case, we're only interested in passing @triggerBody().fileName, so let's assign this expression to the pipeline's SourceFile parameter:



* Finally, let's publish the changes using 'Publish All' button.

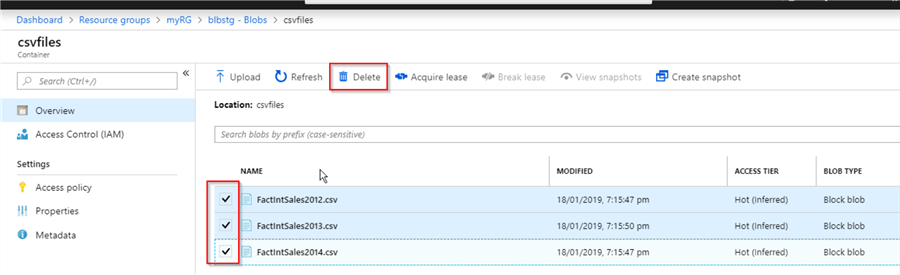
Here's is the JSON code for the new event-based trigger:

{ "name": "MyEventTrigger",  
 "properties": {  
 "runtimeState": "Started",  
 "pipelines": [  
 {  
 "pipelineReference": {  
 "referenceName": "Blob\_SQL\_PL",  
 "type": "PipelineReference"  
 },  
 "parameters": {  
 "SourceFile": "@triggerBody().fileName"  
 }  
 }  
 ],  
 "type": "BlobEventsTrigger",  
 "typeProperties": {  
 "blobPathBeginsWith": "/csvfiles/",  
 "scope": "/subscriptions/cb6712fb-5fb2-4be3-b071-1e369c3d5499/resourceGroups/myRG/providers/Microsoft.Storage/storageAccounts/blbstg",  
 "events": [  
 "Microsoft.Storage.BlobCreated"  
 ]  
 }  
 }  
}

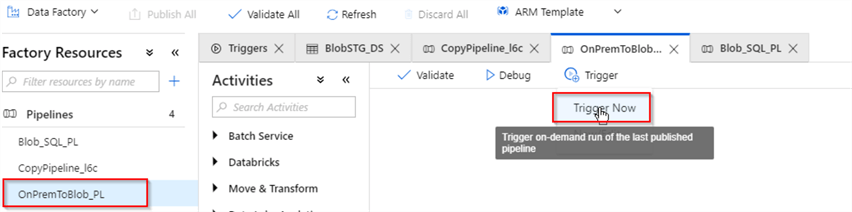
You can also find JSON files for pipeline 'Blob\_SQL\_PL'.

Validating trigger execution in Azure Data Factory

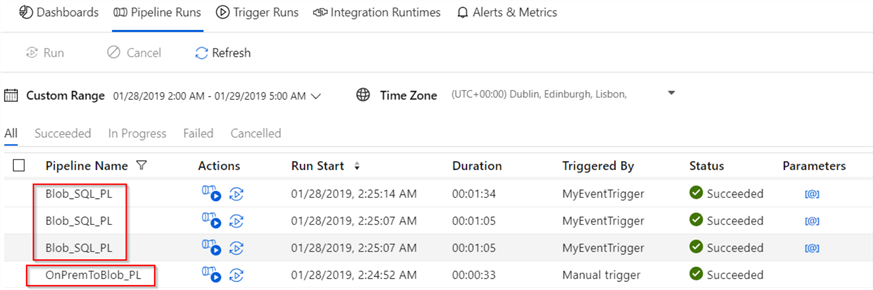
To validate our newly created event trigger, let's open the Blob Storage page and container csvfiles inside it, select all three files and remove them, using the 'Delete' button:



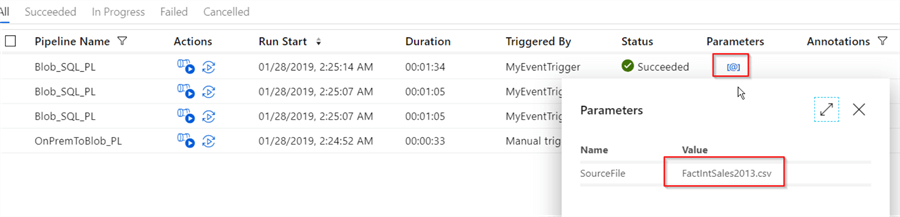
Now, let's trigger OnPremToBlob\_PL pipeline manually, to deposit the CSV files into the blob storage again:



Once execution of the OnPremToBlob\_PL pipeline has finished, we can open ADF monitoring page and examine the execution results, here is the screenshot:



We can also examine file name parameter passed to each execution, using the 'Parameters' button:



*From <*[*https://www.mssqltips.com/sqlservertip/6063/create-event-based-trigger-in-azure-data-factory/*](https://www.mssqltips.com/sqlservertip/6063/create-event-based-trigger-in-azure-data-factory/)*>*

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

ADF v2.0 Pull data from on-prem database and load it into Azure SQL database

Monday, January 11, 2021

7:00 PM

Download and install [Download Microsoft Integration Runtime from Official Microsoft Download Center](https://nam06.safelinks.protection.outlook.com/?url=https%3A%2F%2Fwww.microsoft.com%2Fen-us%2Fdownload%2Fdetails.aspx%3Fid%3D39717&data=04%7C01%7Cshgopath%40microsoft.com%7Ca8170fc91933477b877a08d8b668a6e5%7C72f988bf86f141af91ab2d7cd011db47%7C1%7C0%7C637459907878008456%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C1000&sdata=soiB5WJ%2F%2FeoWo1AGIigEyzY6drQGa9%2BHuBgCL1mOVTE%3D&reserved=0) on **your local machine**

Go to your Azure portal and create an instance of Azure Data Factory v2.0

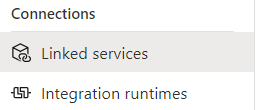
In the Overview details screen, click on Author & Monitor



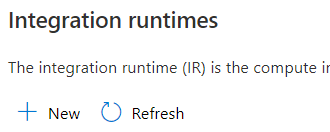
Click on Manage menu item on the left side of the screen



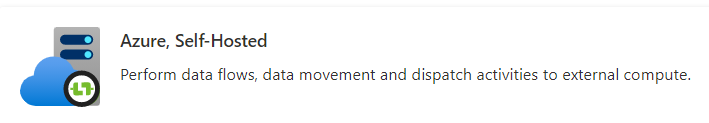
Click on Integration Runtimes



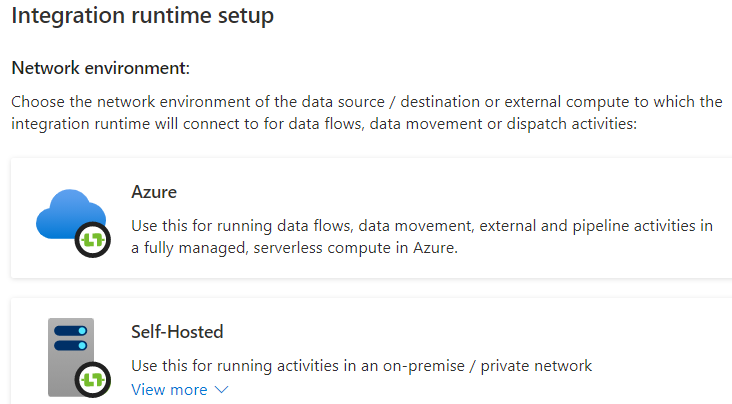
Click on + New link



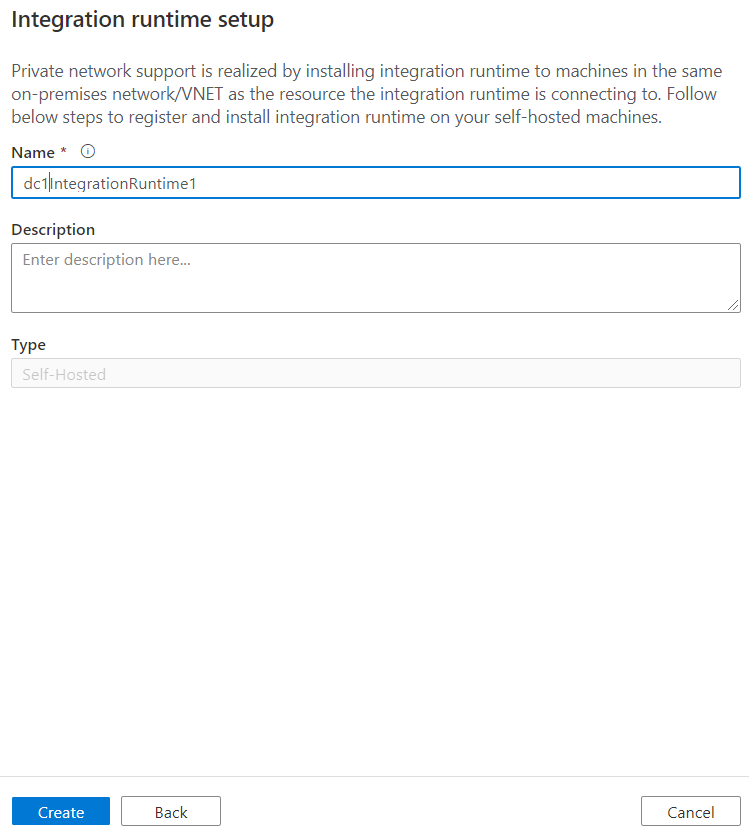
Click on Azure, Self-Hosted option



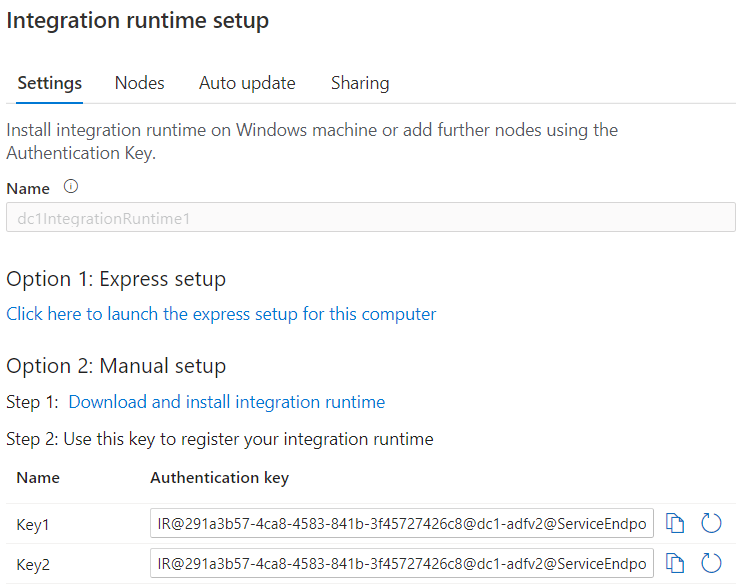
Click on Continue



Click on Self-Hosted and click on Continue



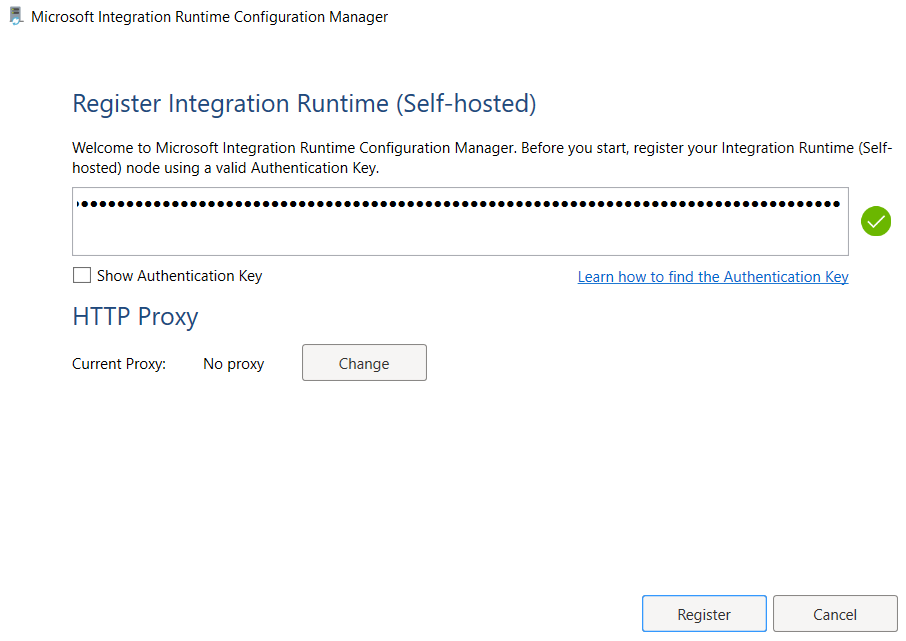
Enter a name in the Name field and click on Create



Coy the Key1 value and save it. You will need it later on.

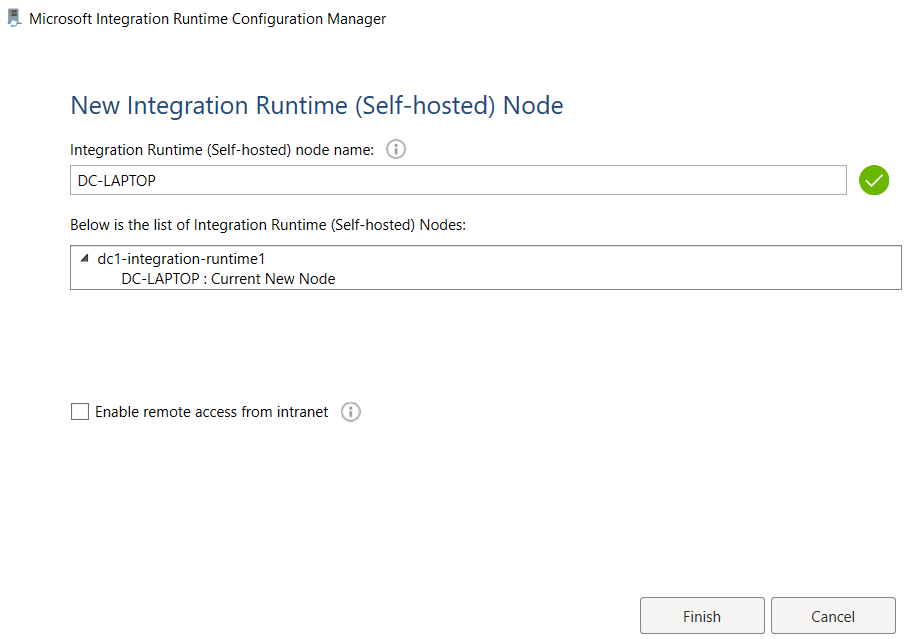
Download and Install integration runtime (Option 2: Manual Setup)

When Integration Runtime (Self-hosted) installation completes, you will come to the Register Integration Runtime (Self-hosted) dialog box.



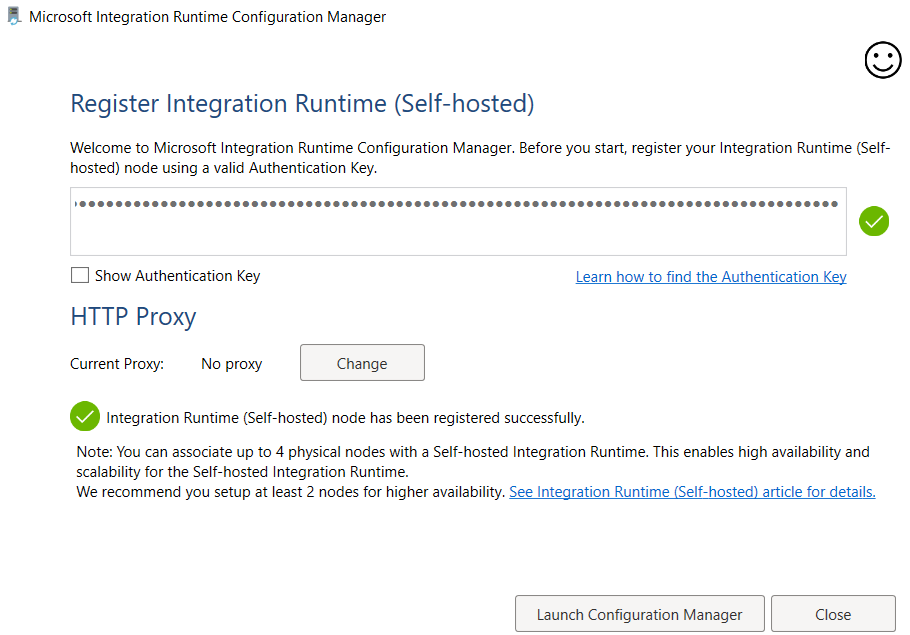
Paste the Key value you saved earlier into the registration box and click on Register button at the bottom

Upon successful registration, you will see the sceen shown below

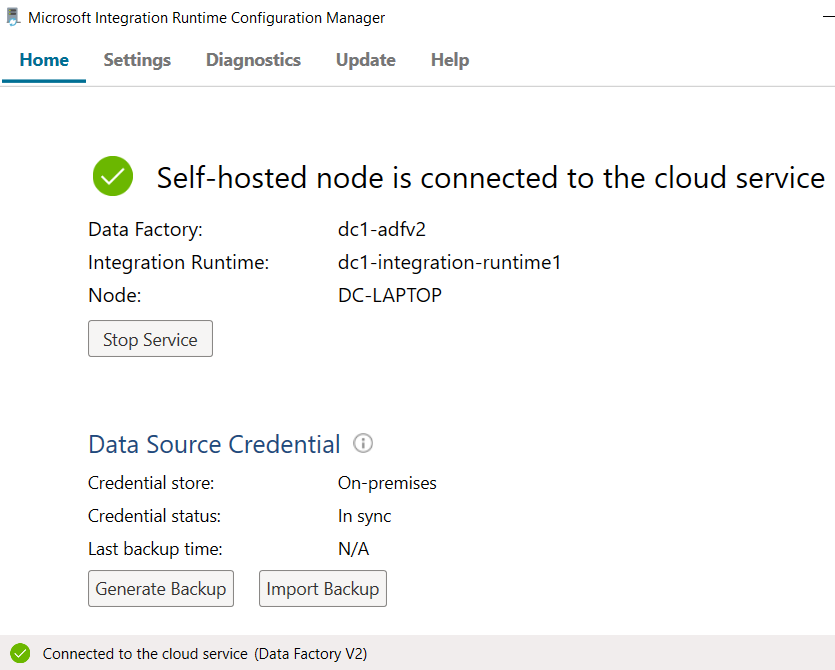


Click on Finish

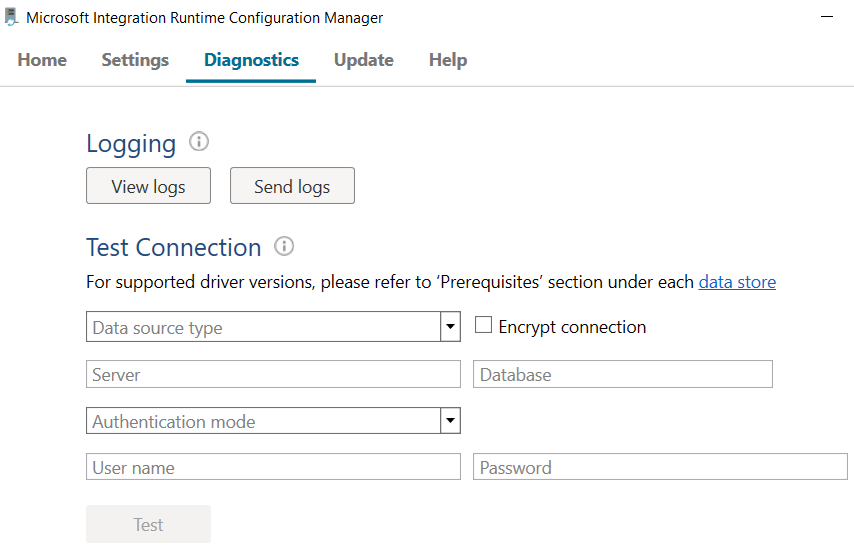
Upon completion, click on Launch Configuration Manager

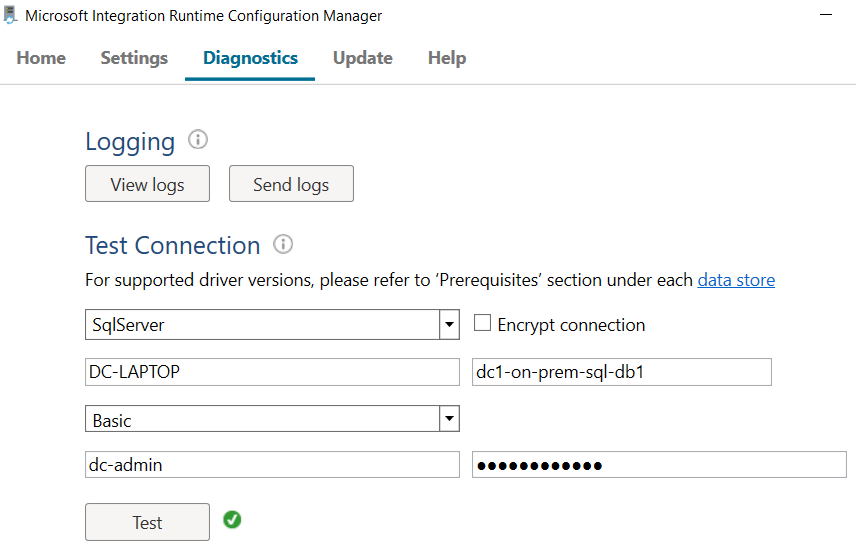


You will see the screen shown below:



Click on Diagnostics menu item at the top

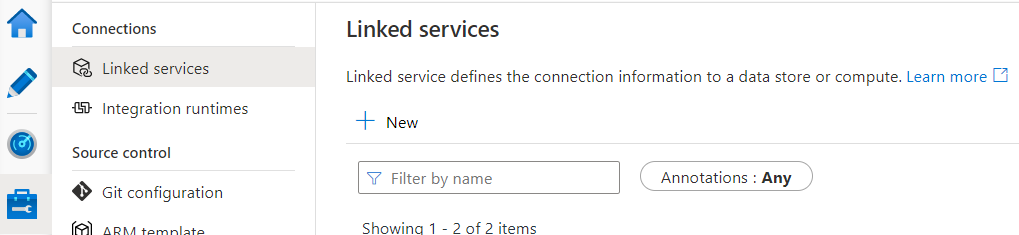


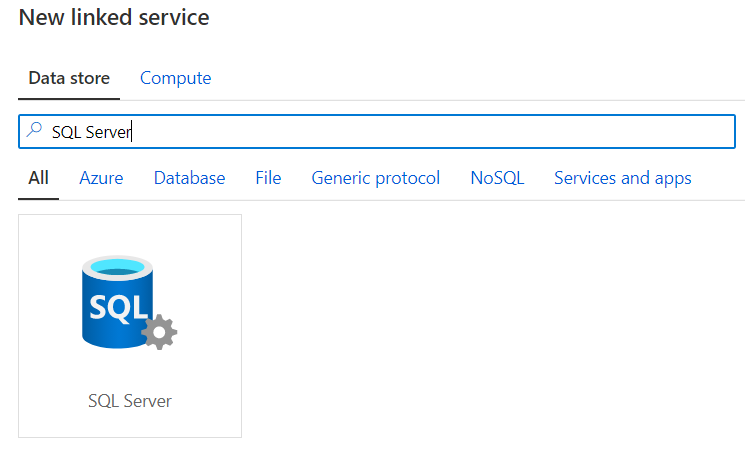


Fill the screen with appropriate values and click on Test. If you see Green checkmark next to Test button, the Integration Runtime setup is complete.

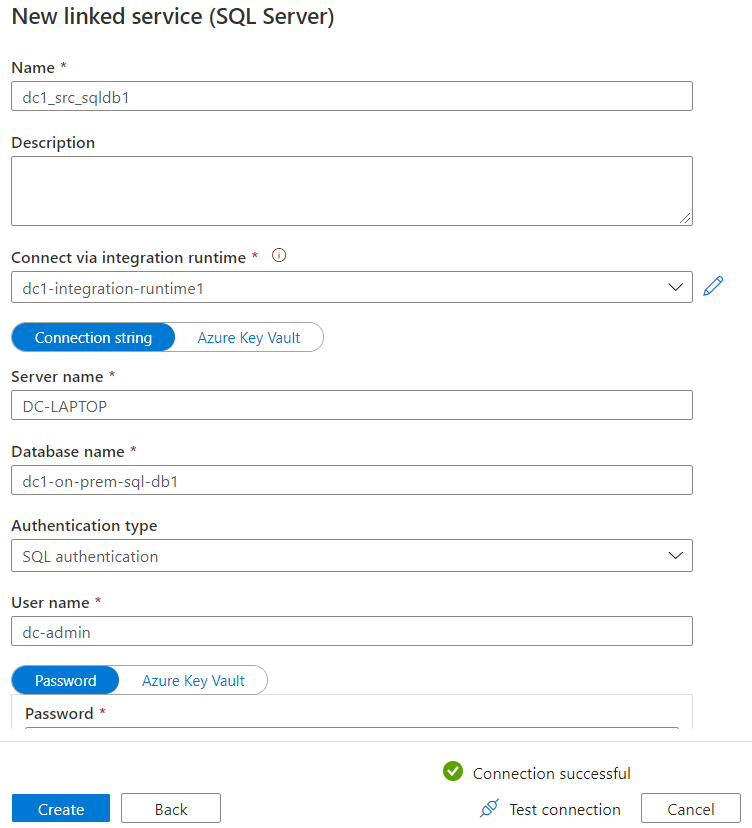
Go back to your Azure portal and go to the Author & Monitor of your ADFv2 instance you created earlier.

Click on the Manage icon on the left --> Linked services --> + New





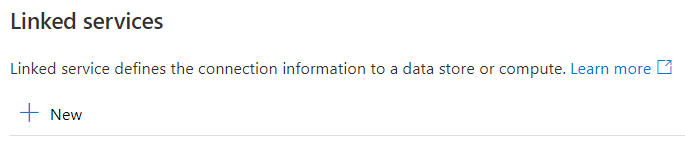
Search for SQL Server, select it and click on Continue button at the bottom



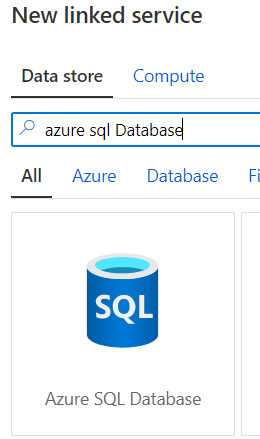
Fill out all the fields. Please pay special attention to **Connect via integration runtime. Make sure you select the Integration Runtime you created earliear.**

Click in Test Connection. If all goes well, you will see Green checkmark Connection successful.

Click on Create.

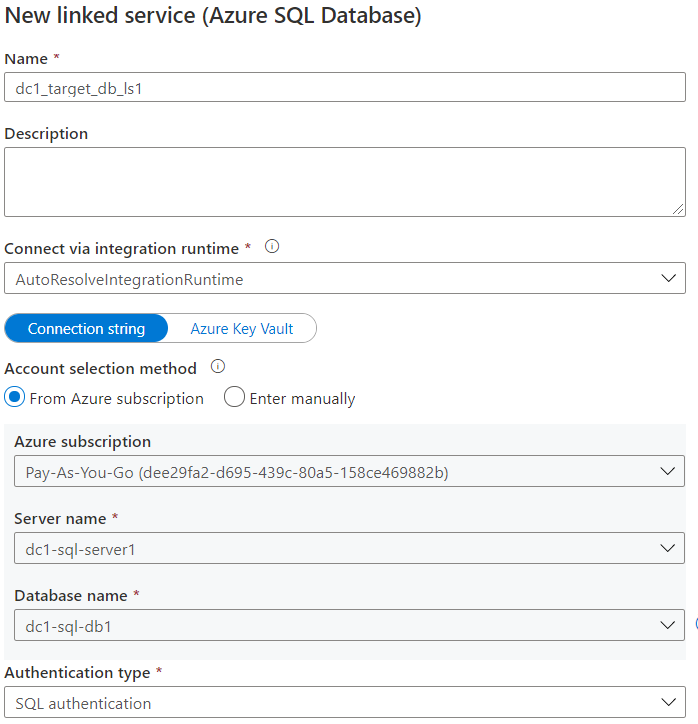


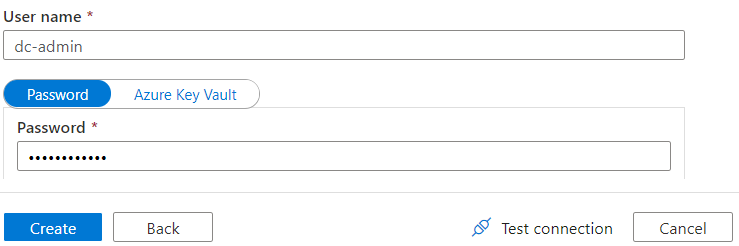
Click on the + New link on the Linked services screen to create the target linked service



Search for Azure SQL Database and select it.

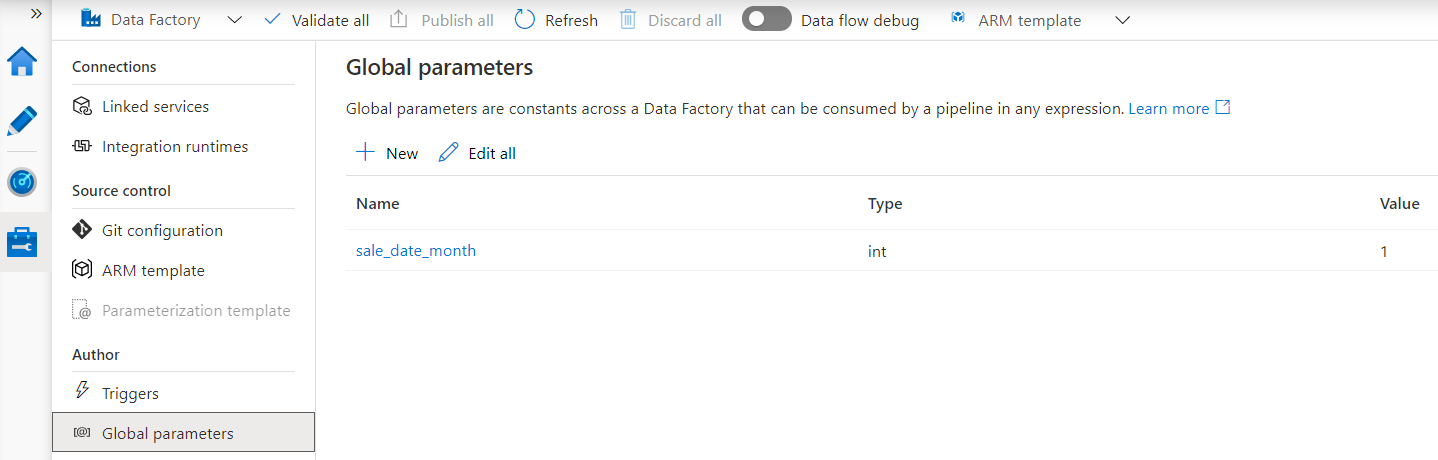
Click on Continue





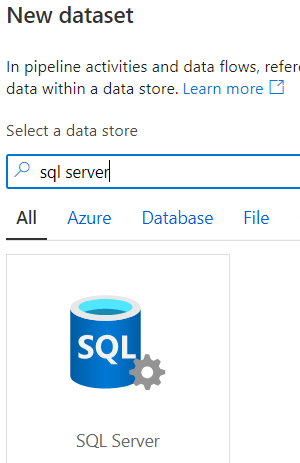
Click on Test Connection. If successful, you will see a Green checkmark next to it. Click on Create

You can create Global variables, should you choose to do so

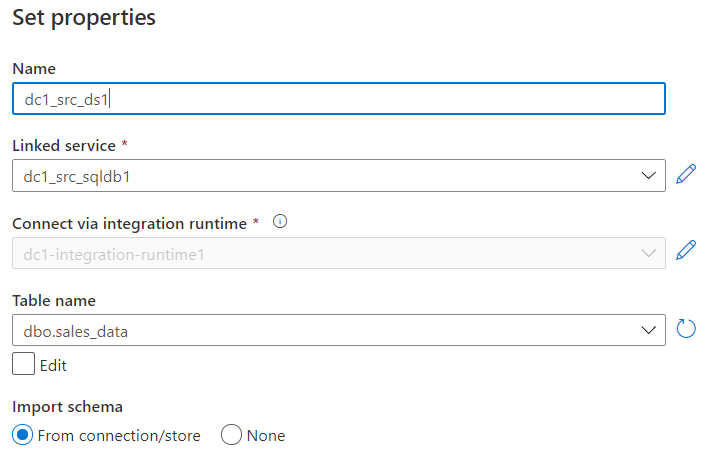


Click on Author (pencil) icon

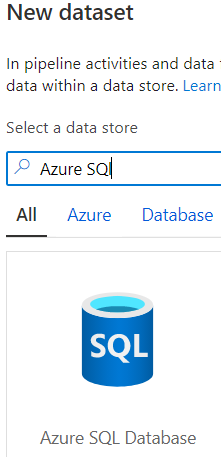
Right click on Datasets, click on New dataset



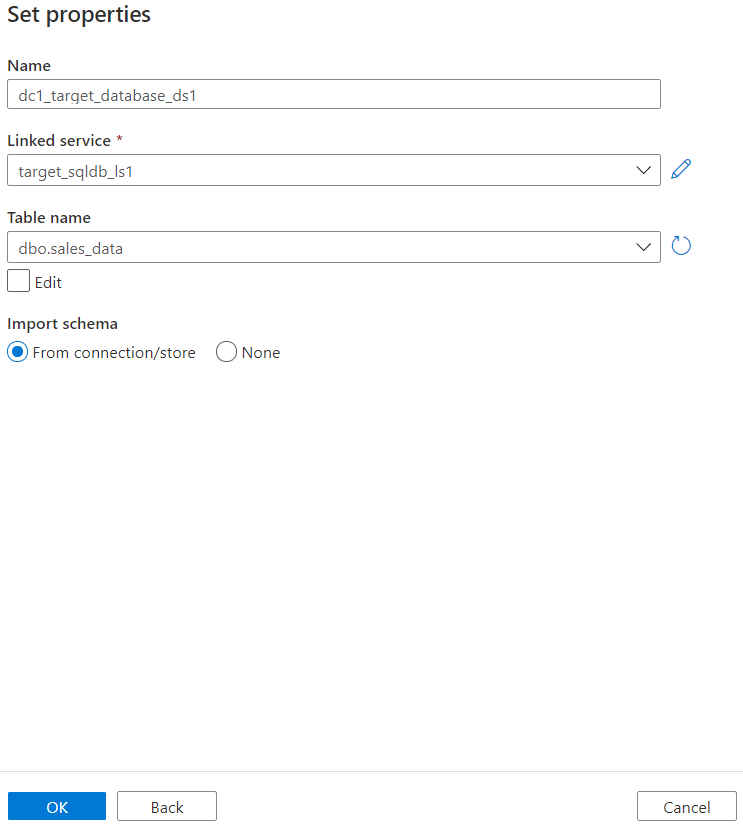
Select SQL Server and click on Continue



Click on OK at the bottom of the screen



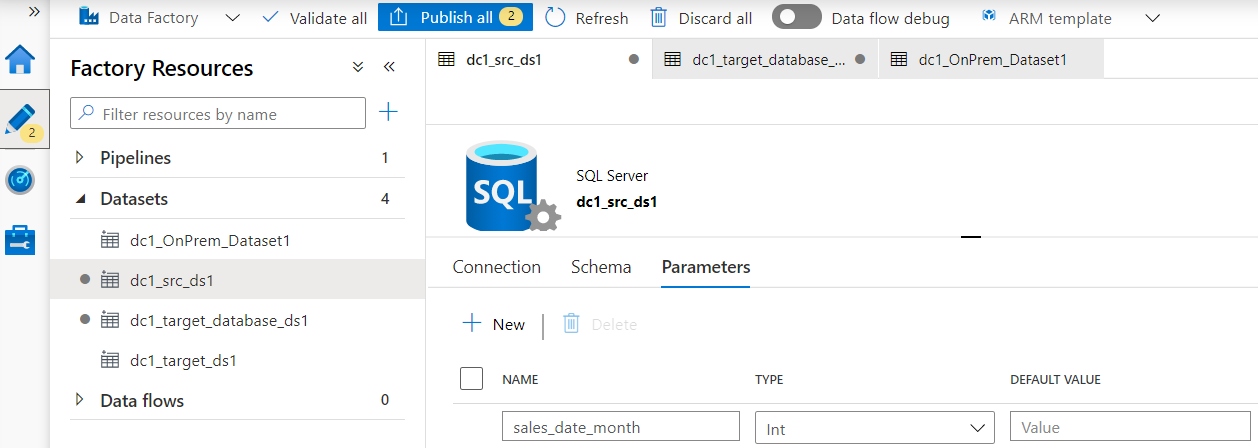
Search for Azure SQL Database, select it and click on Continue at the bottom



Enter the name, target database linked service and the name of the target table.

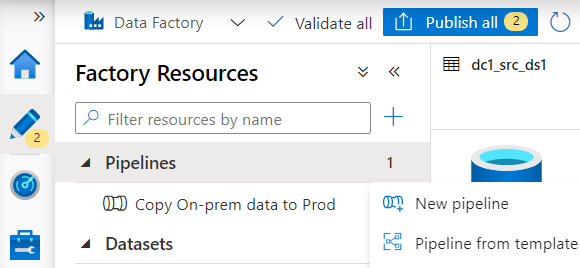
Click on OK

You can create Parameter(s), if required as shown below:

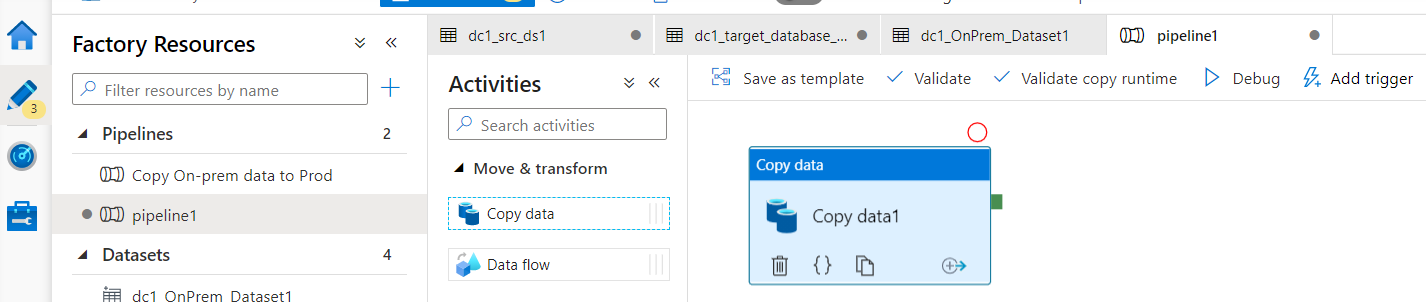


Now it is tie to create a pipeline.

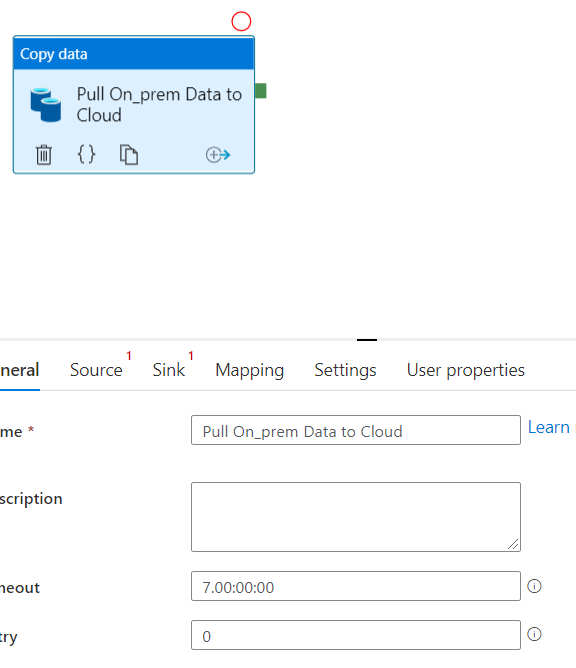
Click on Author (pencil) icon, right mouse click on Pipelines, and click on New Pipeline



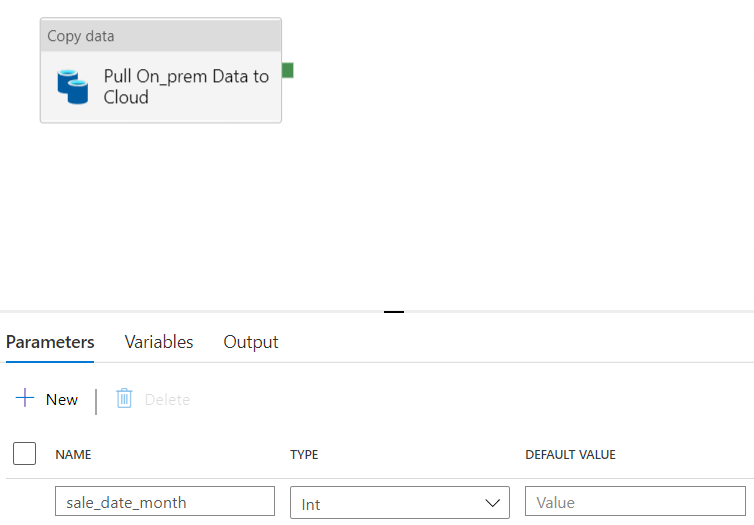
You will see:



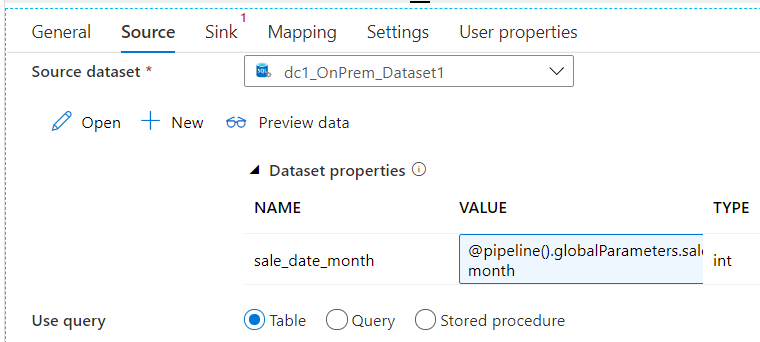
Give it a more meaningful name



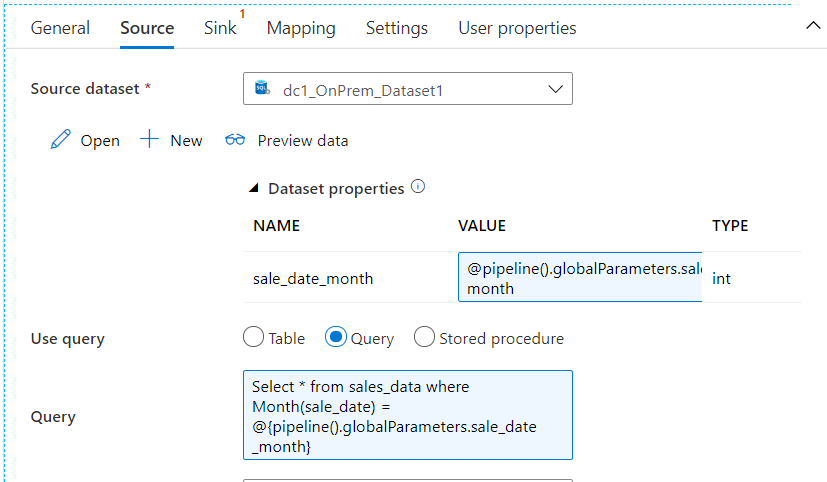
Click in the area below pipeline (Copy data) icon and you will see:



Click on Parameters menu item and click on + New to create a parameter as shown above



Click on Dynamic content

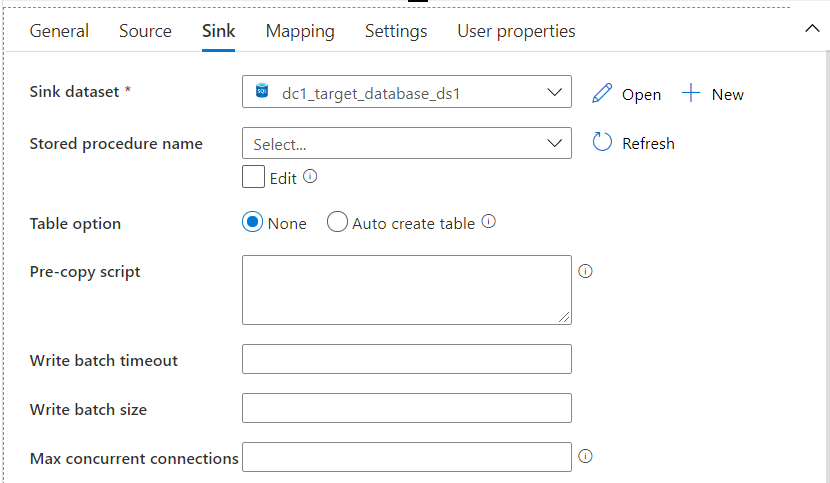


Click on Source Menu item at the bottom half of the screen

Select Source dataset, dynamic value of the parameter field, e.g., sale\_date\_month

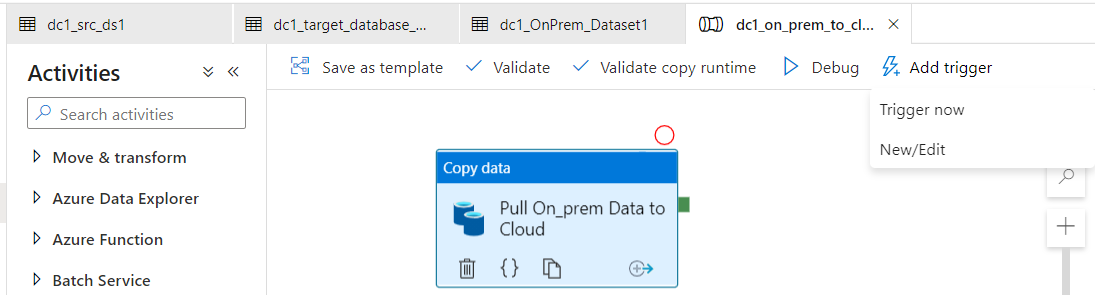
Create query in the query box. **Please sure to note the parameter format.**

Configure Sink as shown below:



After the Source and the Sink are setup, Click on **Publish all**.

After all components are published, click on Add trigger, click on Trigger now



That is it. You can verifiy the results on the Azure SQL Database.