AZURE PROJECT

DATA MIGRATION PIPELINE

BY MEHMAAM & HUZAIFA

Table of Contents

On - Premises SQL Server 5
SQL setup and Installation: 5
Azure Data Factory 14
Create Account 14
Storage Account 18
Part 1: Data Ingestion 18
SQL Server & Azure Data Factory Integration: 18
Part 2: Data Transformation 48
Azure Data Factory & Azure Databricks Integration and Transformation 48
Part 3: Data Loading 59
Azure Databricks & Azure Synapse Analytics Integration and Loading 59

Mehmaam & Huzaifa

Contacts

Syed Muhammad Mehmaam

I am a Data Engineer with a couple of years of experience. I have worked on various projects related to Cloud Data Engineering and Data Scientist.

If you'd like to get in touch:

Linkedin : https://www.linkedin.com/in/muhammad-mehmaam/
Github : https://github.com/Mehmaam99
Mail : muhammadmehmaam@gmail.com

Muhammad Huzaifa Imtiaz

I'm an Azure Cloud Data Engineer with a few years of hands-on experience. I've had my fair share of projects in the exciting world of cloud data engineering.

If you ever want to chat or connect, don't hesitate to reach out!

Linkedin : https://www.linkedin.com/in/huzaifai/
Github : https://github.com/Huzaifa116
Mail : datas.huzaifa@gmail.com

Mehmaam & Huzaifa

On-Permises SQL Server

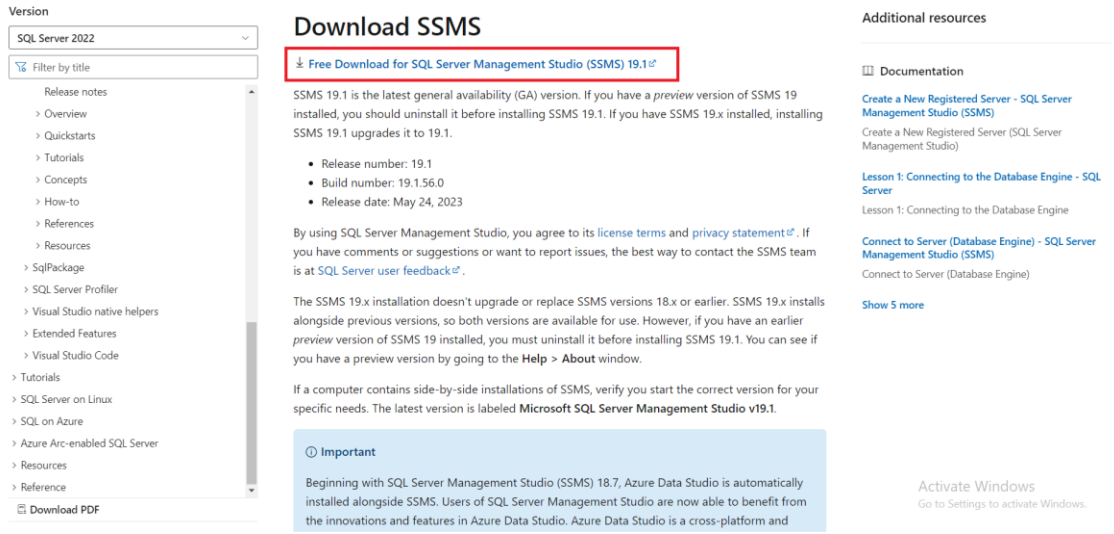
Azure End to End Migration Project

On - Premises SQL Server

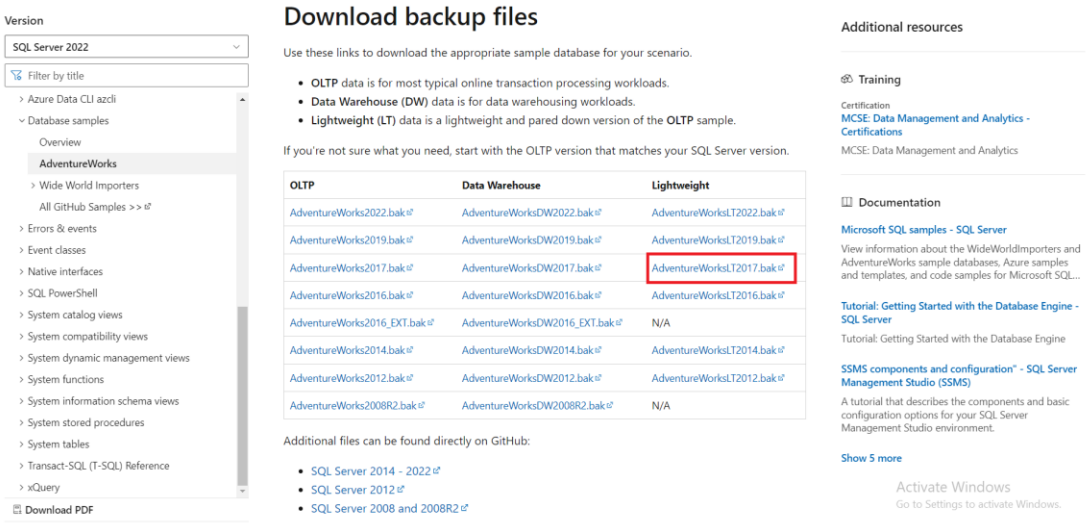
SQL setup and Installation:

1. Download SQL Server https://www.microsoft.com/en-us/sql-server/sql-server-downloads

2. Install SSMS https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16&redirectedfrom=MSDN

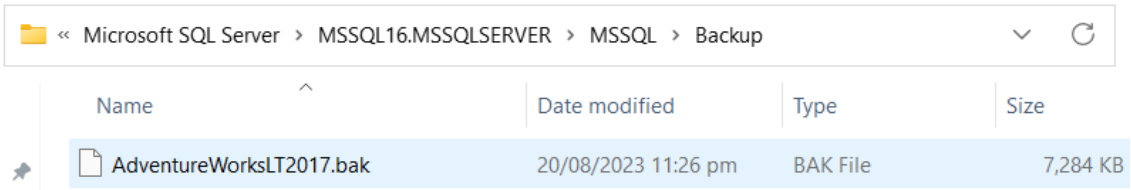


3. Download light weight data
   https://learn.microsoft.com/en-us/sql/samples/adventureworks-install-configure?view=sql-server-ver16&tabs=ssms



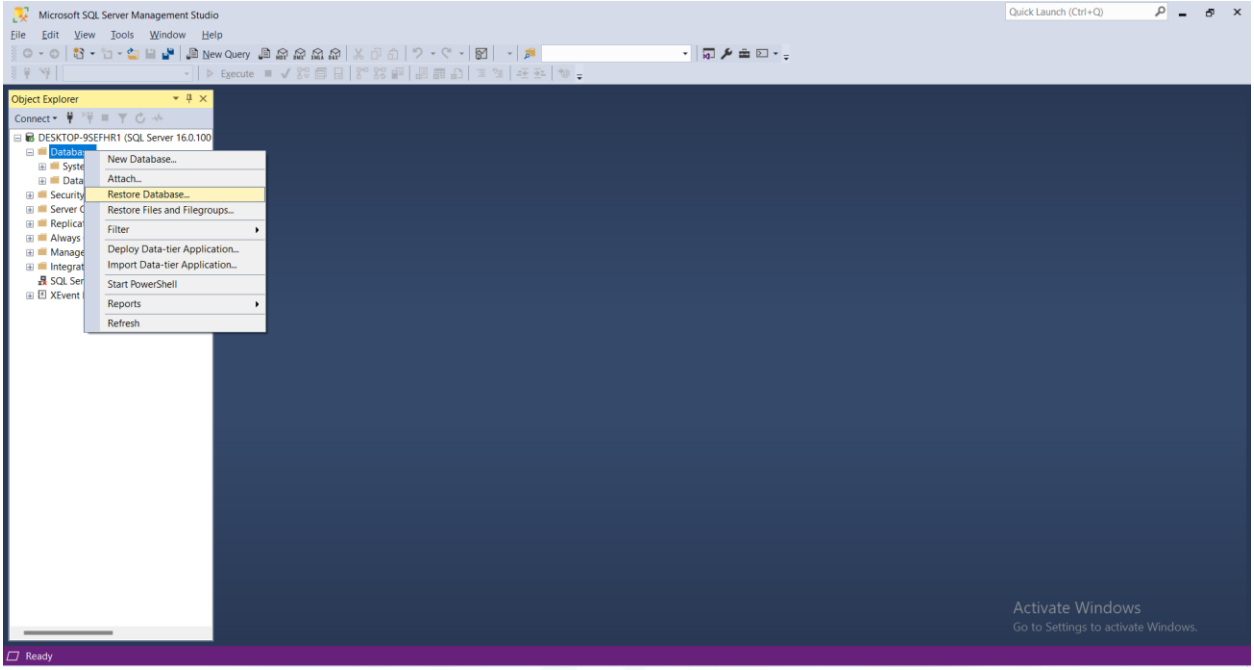
4. Move into this location

   C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\Backup

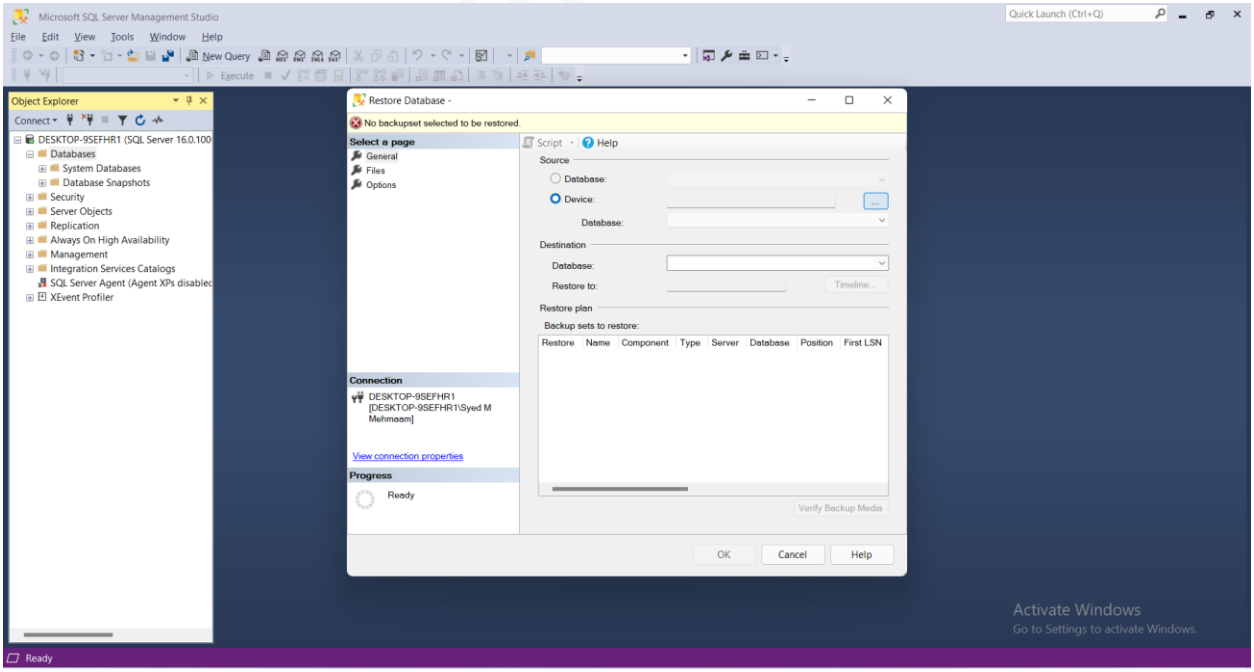


5. Restore Database:
   Steps to restore database and authentication in sql server:

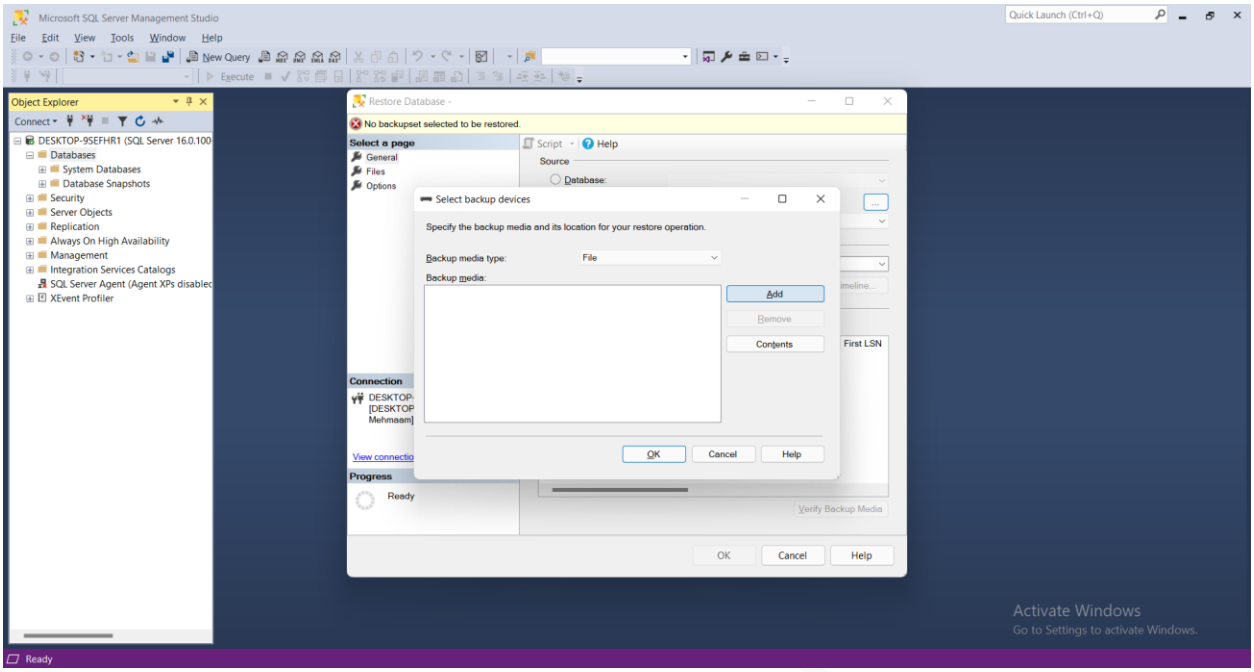
Step 1:



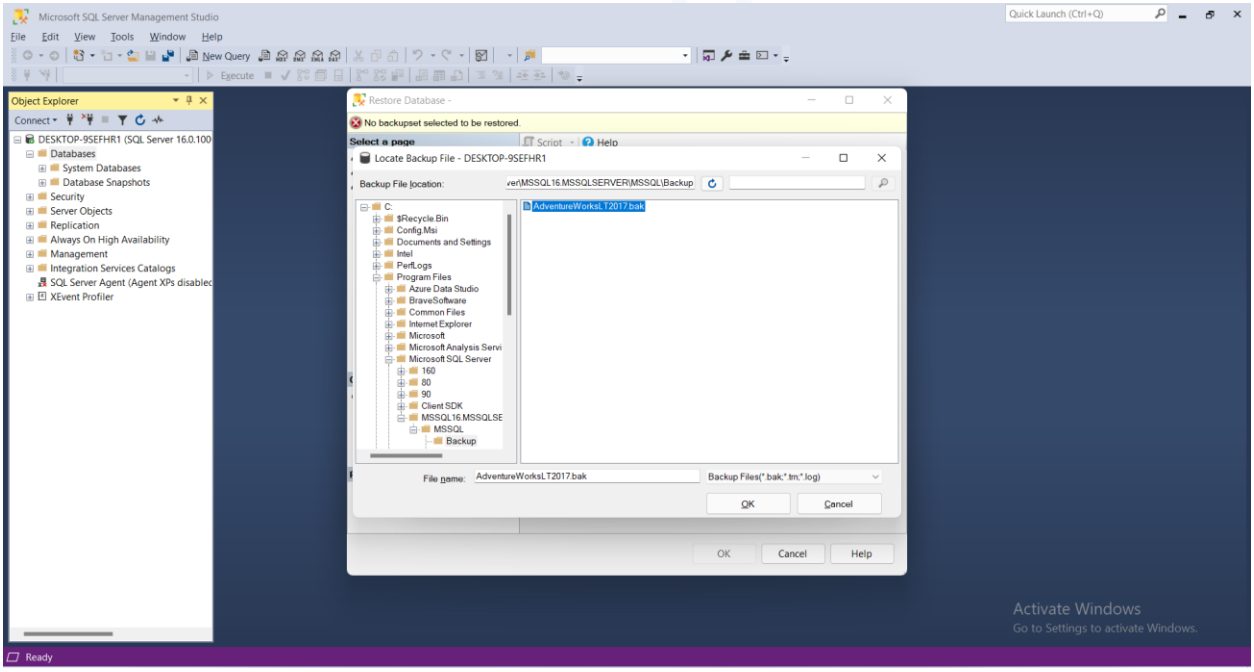
Step 2:



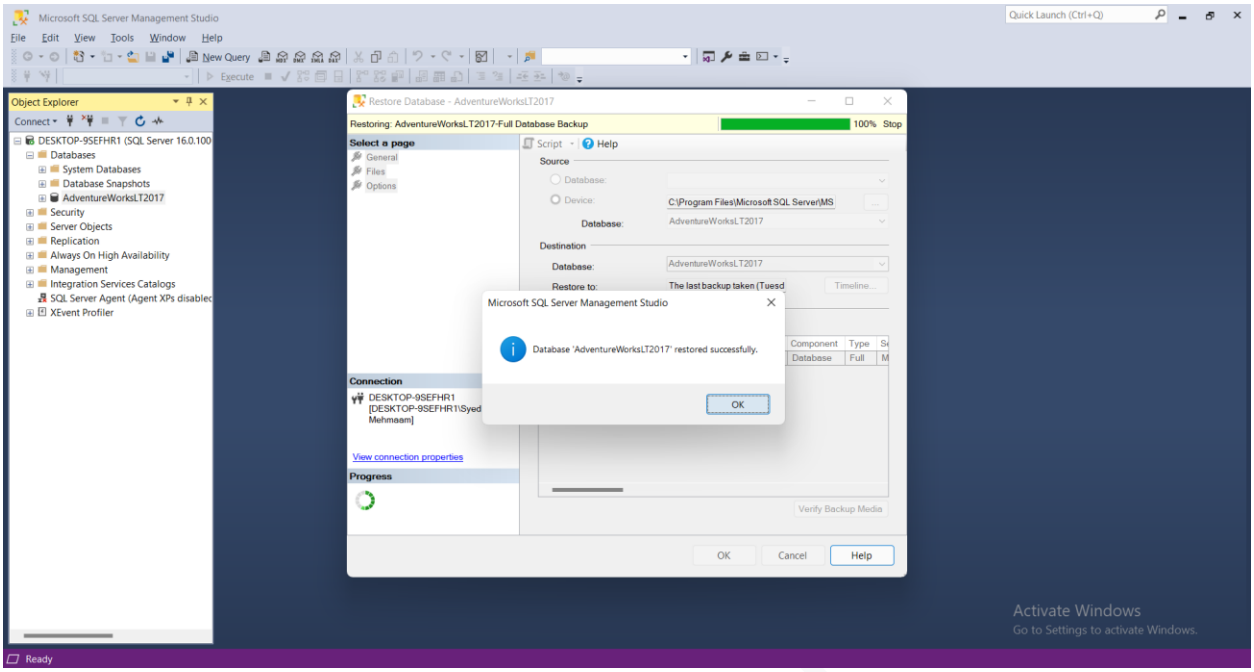
Step 3:


Microsoft SQL Server Management Studio
File Edit View Tools Window Help
New Query
Execute
Quick Launch (Ctrl+Q)
Object Explorer
Connect
DESKTOP-9SEFHR1 (SQL Server 16.0.100
Databases
System Databases
Database Snapshots
Security
Server Objects
Replication
Always On High Availability
Management
Integration Services Catalogs
SQL Server Agent (Agent XPs disablec
XEvent Profiler
Restore Database -
No backupset selected to be restored.
Select a page
General
Files
Options
Script
Help
Source
Database:
Select backup devices
Specify the backup media and its location for your restore operation.
Backup media type:
File
Backup media:
Add
Remove
Contents
OK
Cancel
Help
First LSN
Connection
View connectio
Progress
Ready
Verify Backup Media
OK
Cancel
Help
Activate Windows
Go to Settings to activate Windows.
Ready

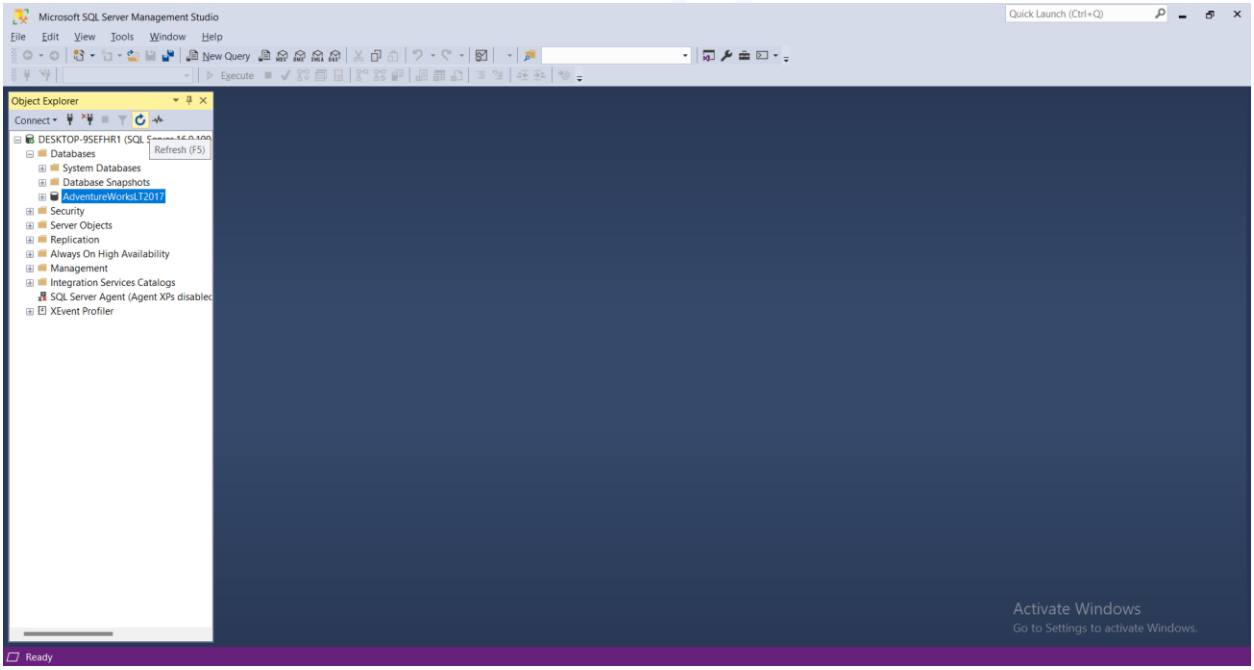
Step 4:


Microsoft SQL Server Management Studio
File Edit View Tools Window Help
New Query
Execute
Quick Launch (Ctrl+Q)
Object Explorer
Connect
DESKTOP-9SEFHR1 (SQL Server 16.0.100
Databases
System Databases
Database Snapshots
Security
Server Objects
Replication
Always On High Availability
Management
Integration Services Catalogs
SQL Server Agent (Agent XPs disablec
XEvent Profiler
Restore Database -
No backupset selected to be restored.
Select a page
Script
Help
Locate Backup File - DESKTOP-9SEFHR1
Backup File location:
ver\MSSQL16.MSSQLSERVER\MSSQL\Backup
C:
$Recycle.Bin
Config.Msi
Documents and Settings
Intel
PerfLogs
Program Files
Azure Data Studio
BraveSoftware
Common Files
Internet Explorer
Microsoft
Microsoft Analysis Servi
Microsoft SQL Server
160
80
90
Client SDK
MSSQL16.MSSQLSE
MSSQL
Backup
AdventureWorksLT2017.bak
File name:
AdventureWorksLT2017.bak
Backup Files(*.bak;*.trn;*.log)
OK
Cancel
OK
Cancel
Help
Activate Windows
Go to Settings to activate Windows.
Ready

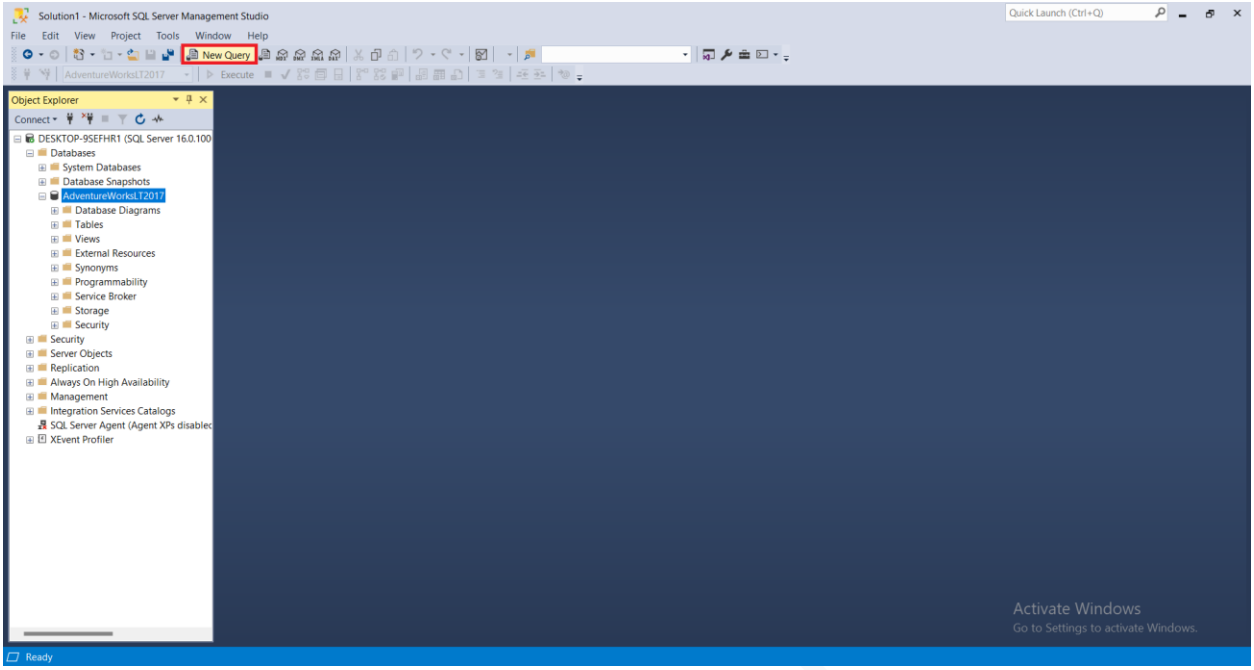
Step 5:


Microsoft SQL Server Management Studio
File Edit View Tools Window Help
New Query
Execute
Quick Launch (Ctrl+Q)
Object Explorer
Connect
DESKTOP-9SEFHR1 (SQL Server 16.0.100
Databases
System Databases
Database Snapshots
AdventureWorksLT2017
Security
Server Objects
Replication
Always On High Availability
Management
Integration Services Catalogs
SQL Server Agent (Agent XPs disablec
XEvent Profiler
Restore Database - AdventureWorksLT2017
Restoring: AdventureWorksLT2017-Full Database Backup
100% Stop
Select a page
General
Files
Options
Script
Help
Source
Database:
Device:
C:\Program Files\Microsoft SQL Server\MS
Database:
AdventureWorksLT2017
Destination
Database:
AdventureWorksLT2017
Restore to:
The last backup taken (Tuesd
Timeline...
Microsoft SQL Server Management Studio
Database 'AdventureWorksLT2017' restored successfully.
OK
Component
Type
Database
Full
Connection
DESKTOP-9SEFHR1
[DESKTOP-9SEFHR1\Syed
Mehmaam]
View connection properties
Progress
Verify Backup Media
OK
Cancel
Help
Activate Windows
Go to Settings to activate Windows.
Ready

Step 6:

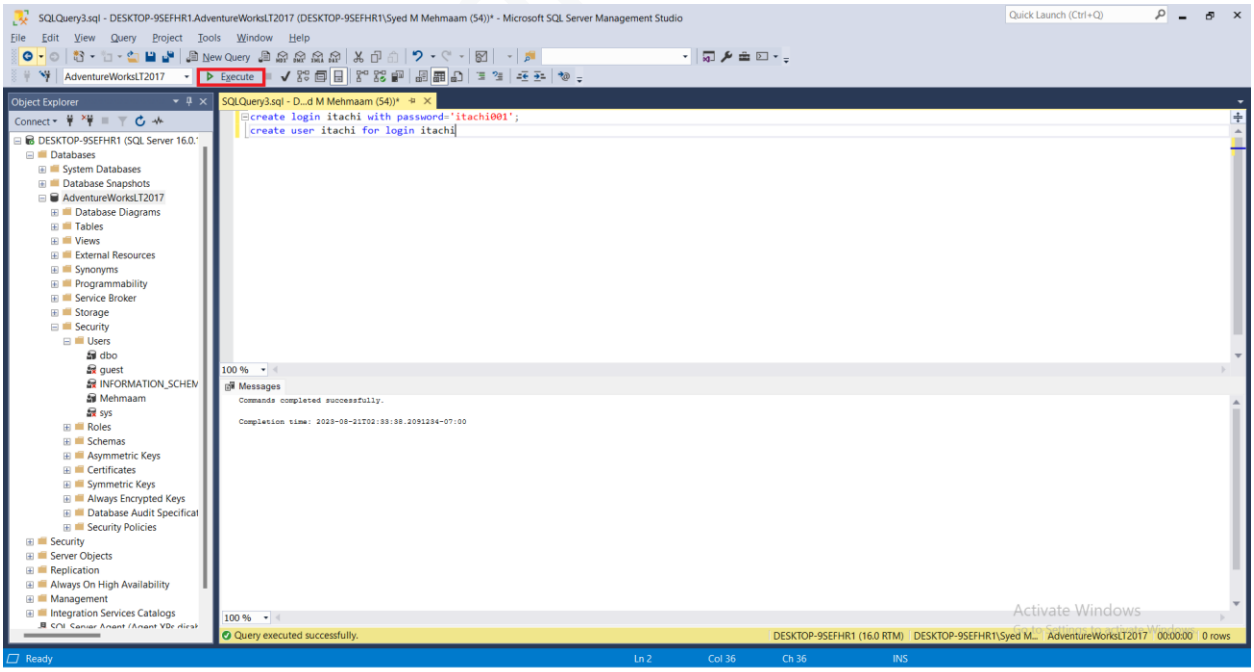

Microsoft SQL Server Management Studio
File Edit View Tools Window Help
New Query
Execute
Quick Launch (Ctrl+Q)
Object Explorer
Connect
DESKTOP-9SEFHR1 (SQL
Refresh (F5)
Databases
System Databases
Database Snapshots
AdventureWorksLT2017
Security
Server Objects
Replication
Always On High Availability
Management
Integration Services Catalogs
SQL Server Agent (Agent XPs disablec
XEvent Profiler
Activate Windows
Go to Settings to activate Windows.
Ready

Step 7:

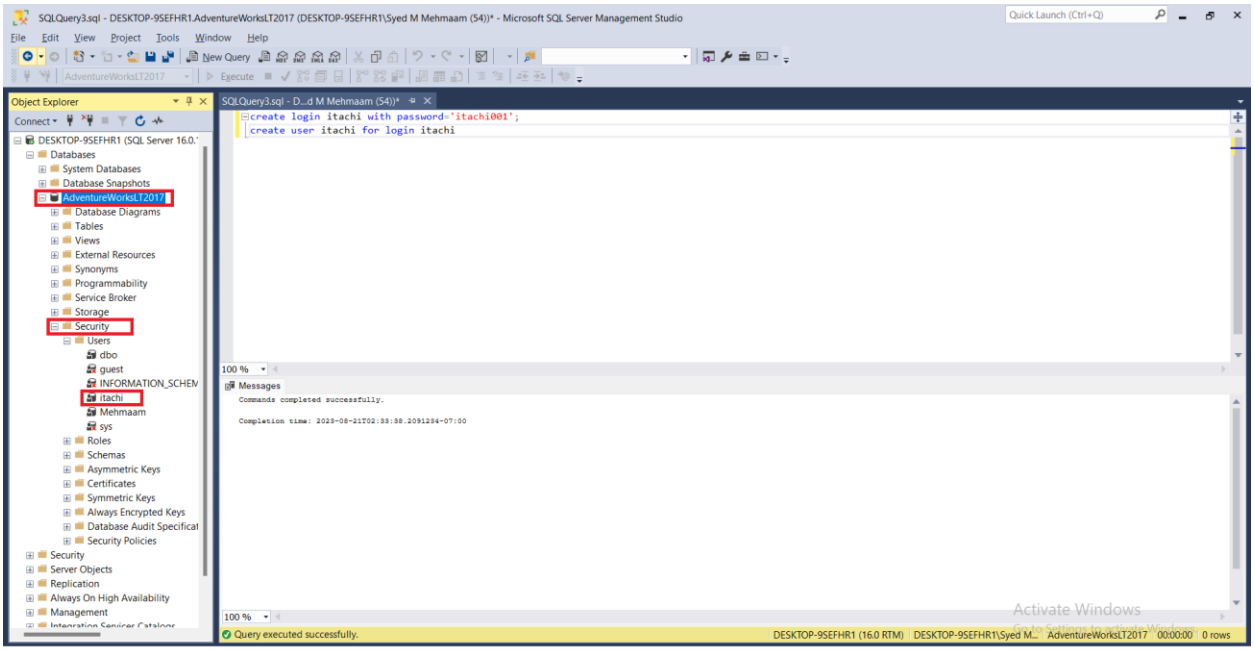


Step 8:

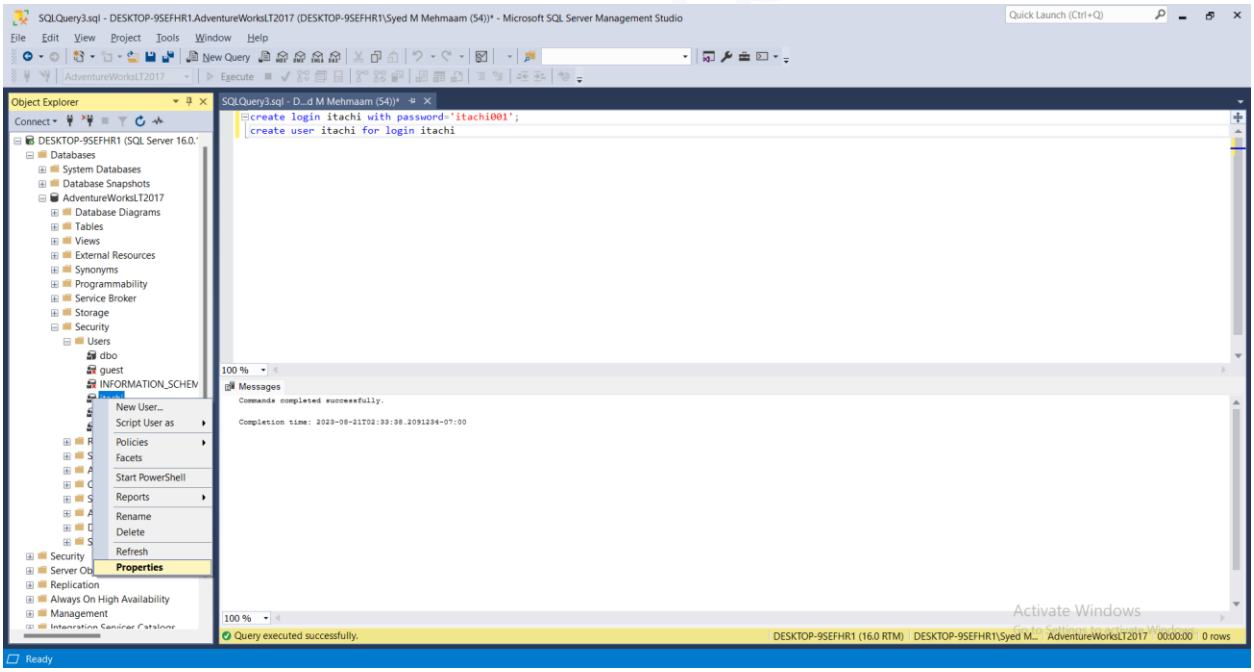
Create a username and password, then execute.
create login **username** with password='password12345';
create user **username** for login **username.**



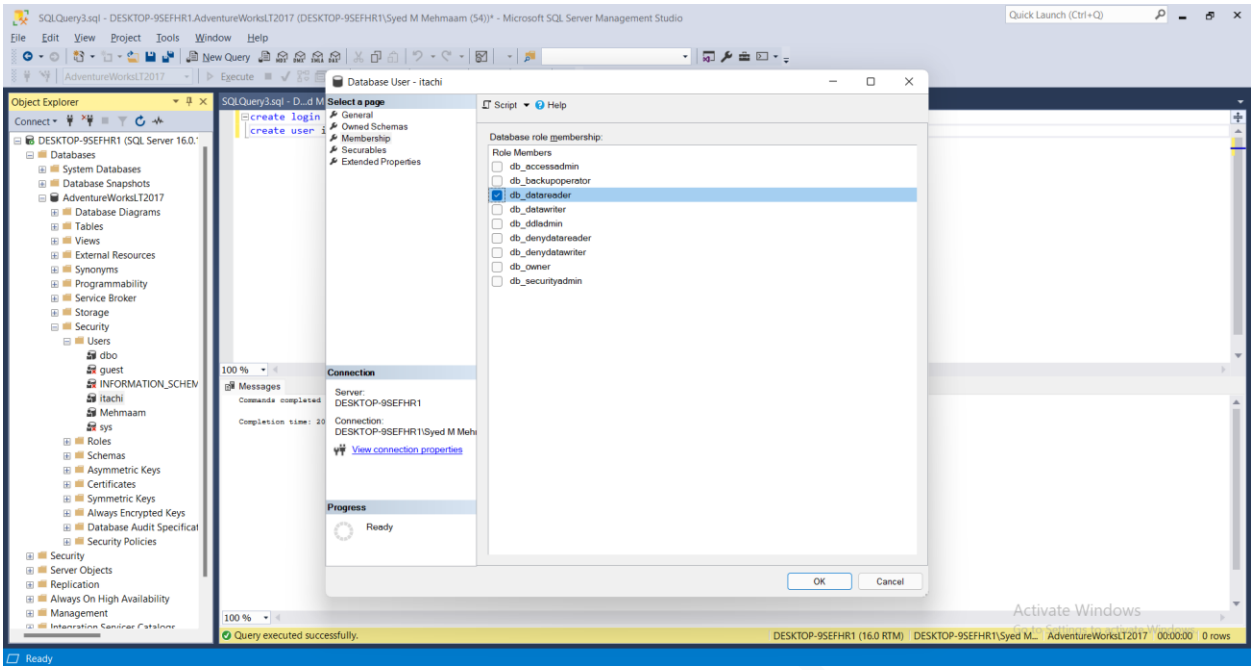
Step 9:



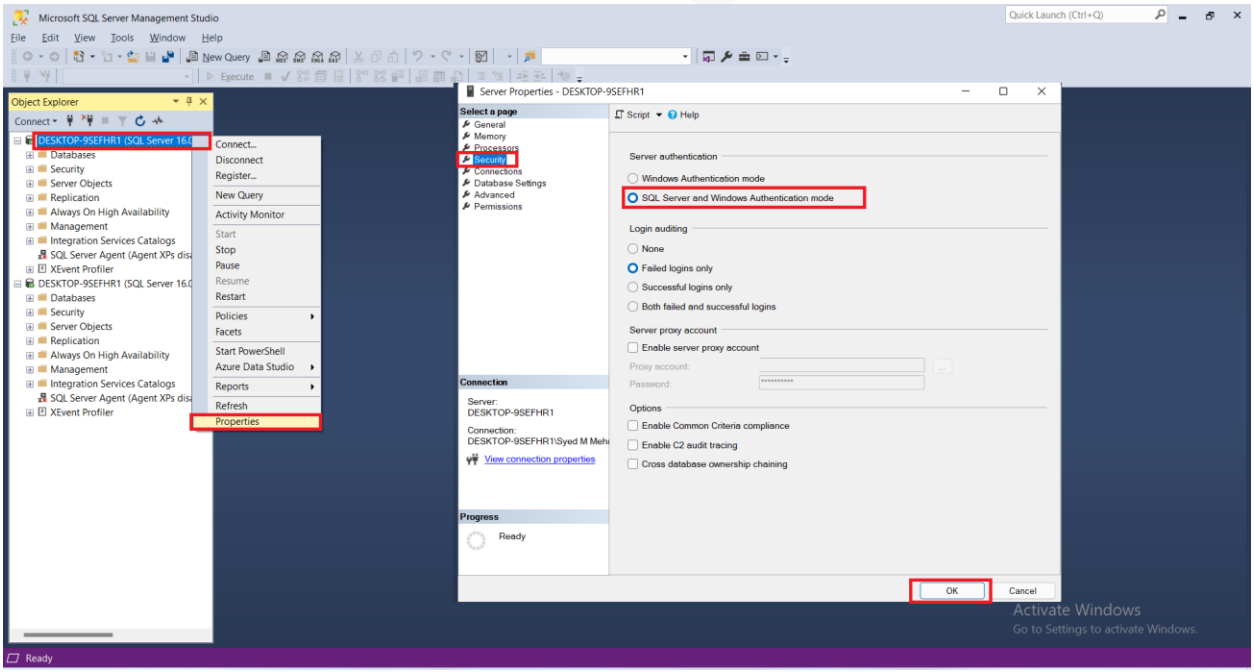
Step 10:



Step 11:



Additional Step:

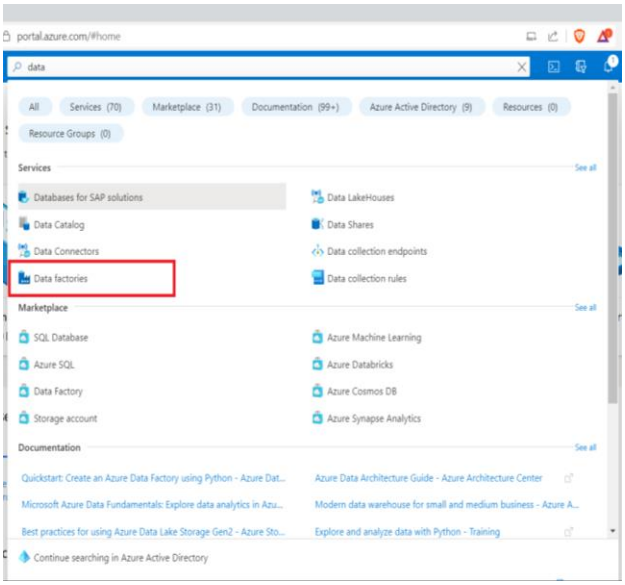


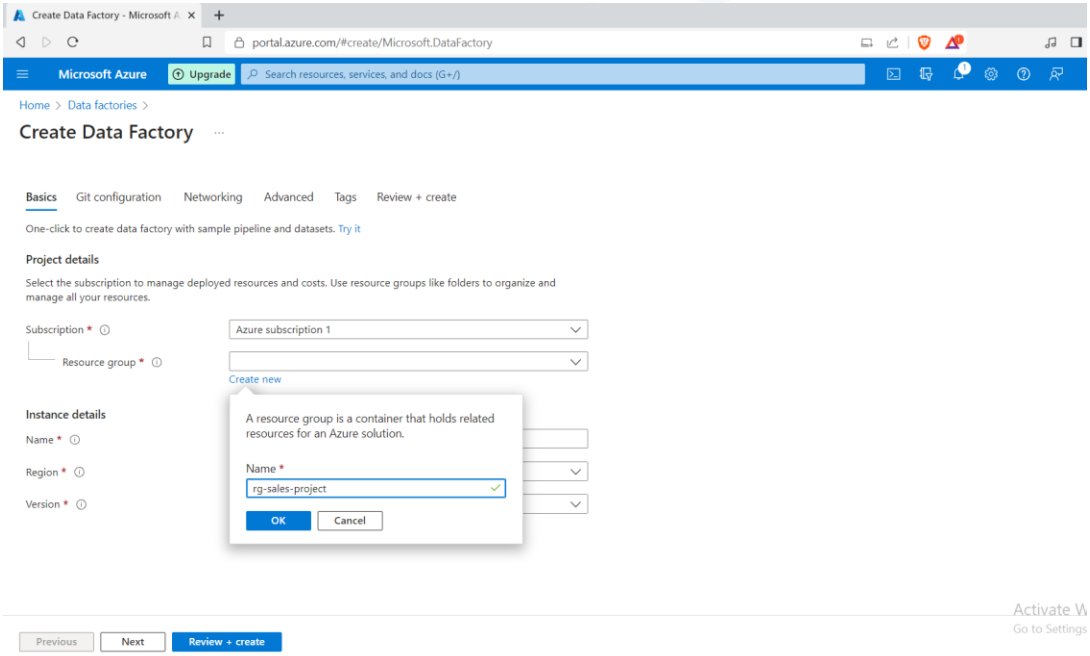
Azure Data Factory

Azure Data Factory

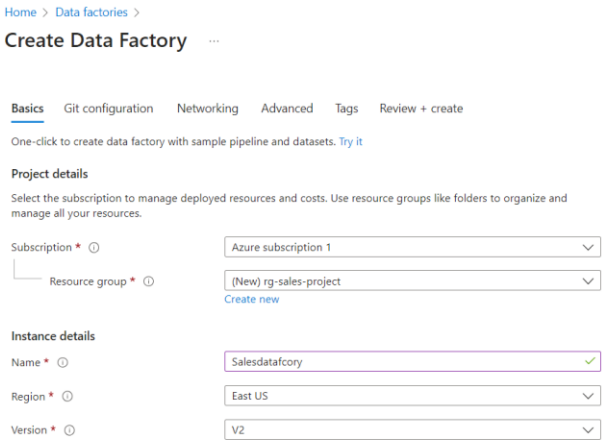
Create Account

Create a Data Factory Service and assign it to a resource group.

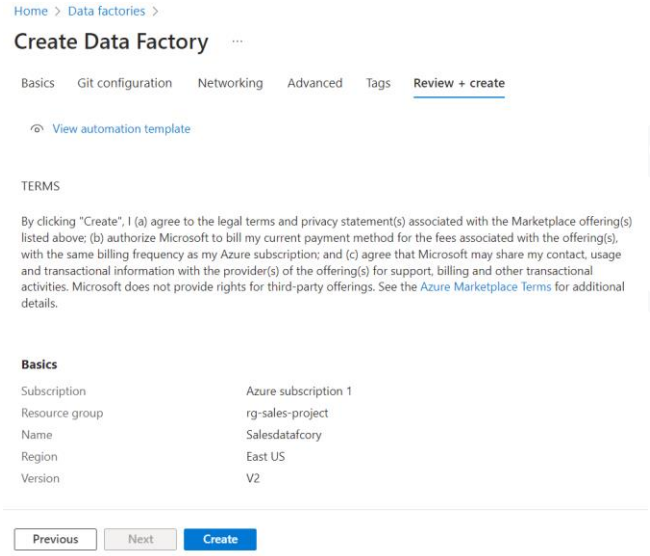




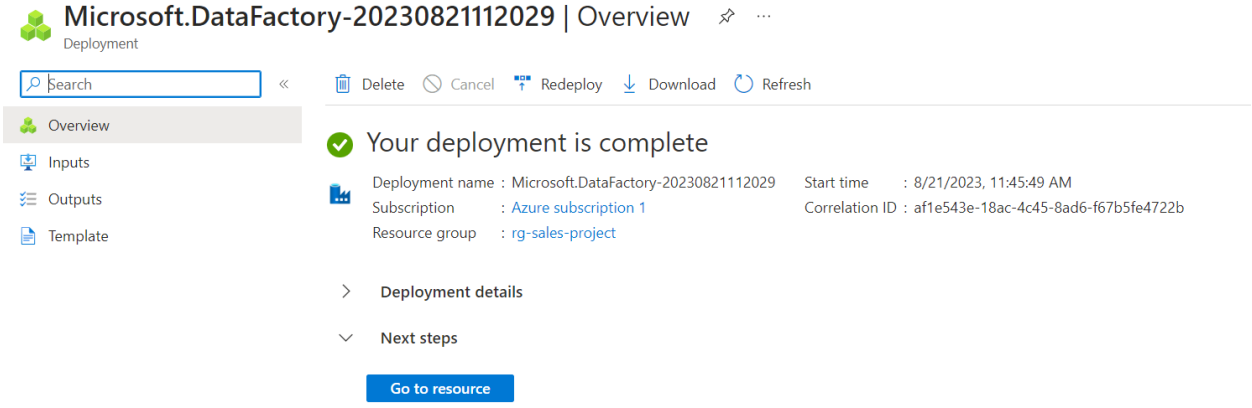
& Huzaifa



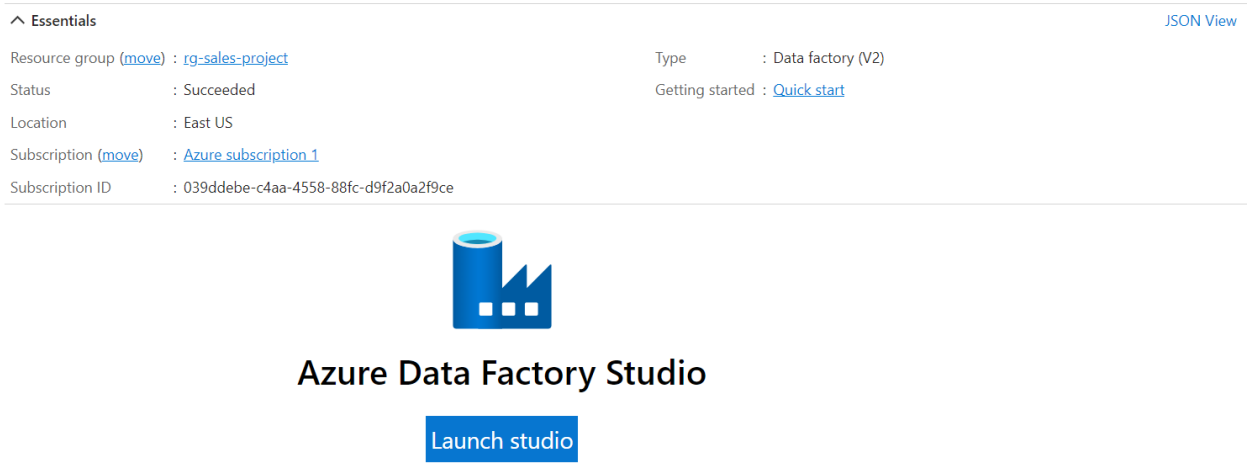
All the settings will remain the same, just click “**Review + Create**”, and after that, it will be the same just click on create.



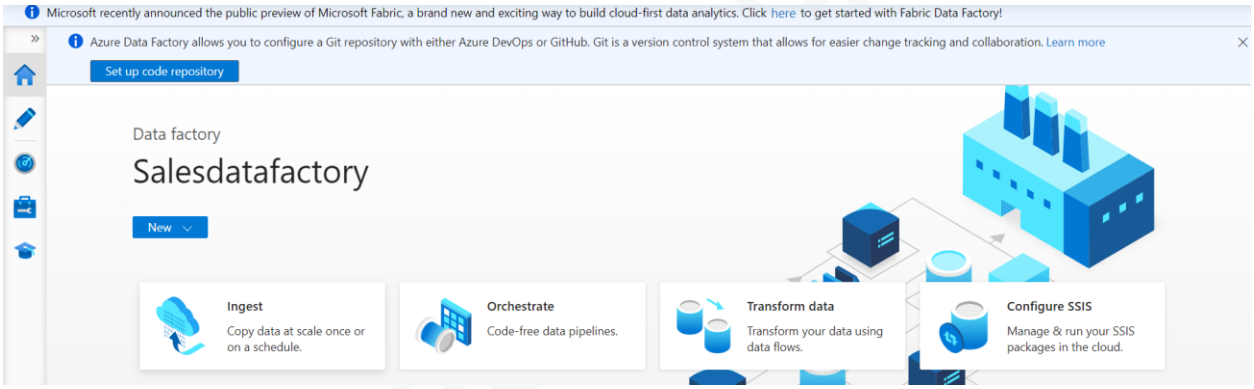
After clicking on **“create”**, it will appear like that simply click on “**go to resource**”.



Click on Launch studio.



The Data Factory UI will appear as follows.



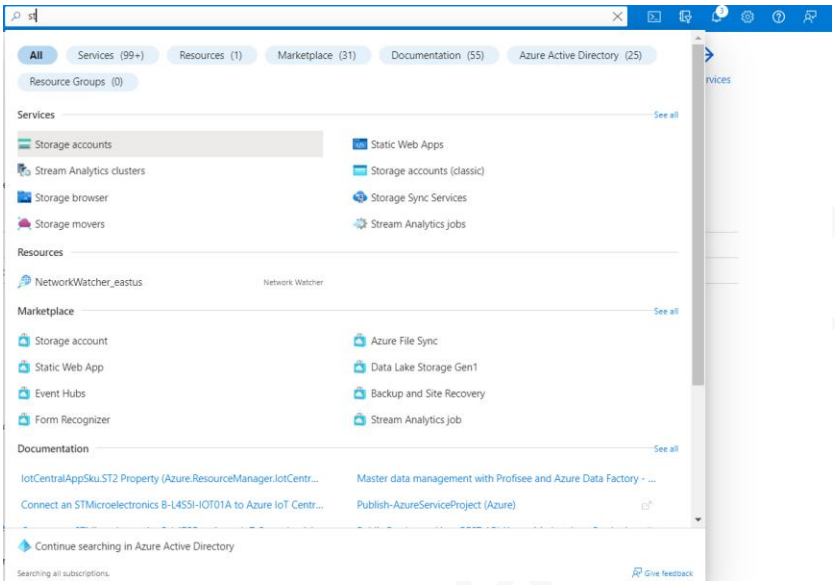
Storage Account

Storage Account

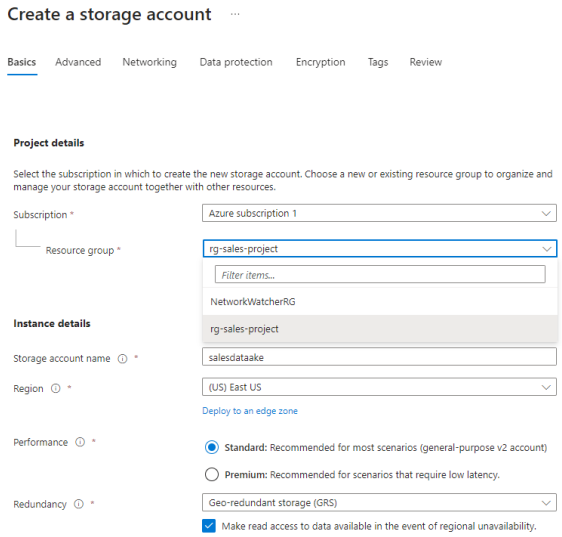
Part 1: Data Ingestion

SQL Server & Azure Data Factory Integration:

The Data Factory account has been created now, we need to create a Data Lake Gen2 storage account.

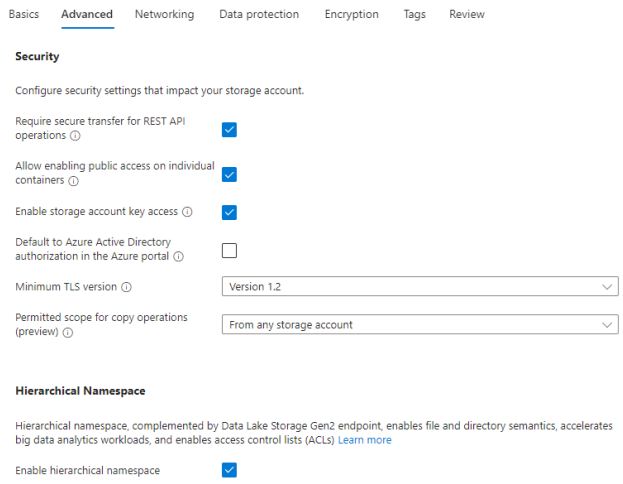


Ensure that you select the existing resource group.

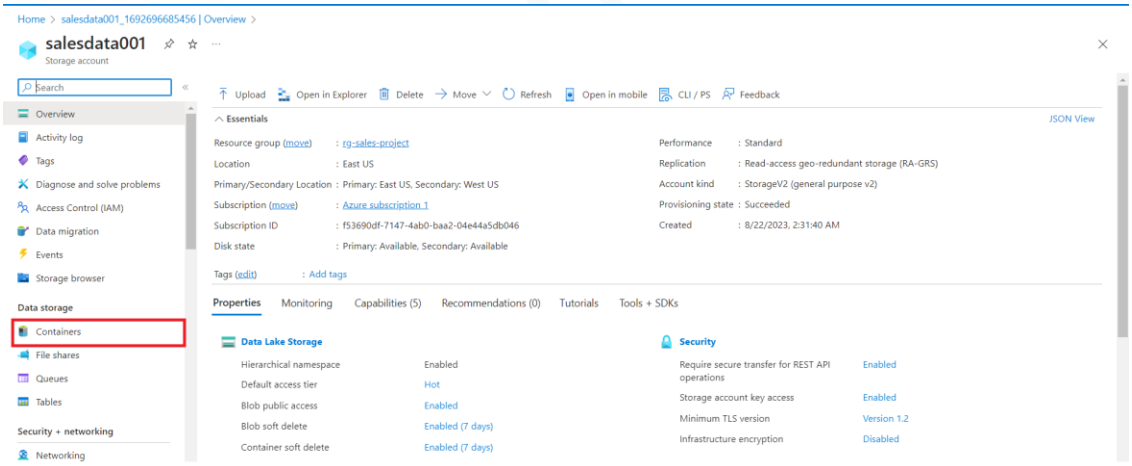


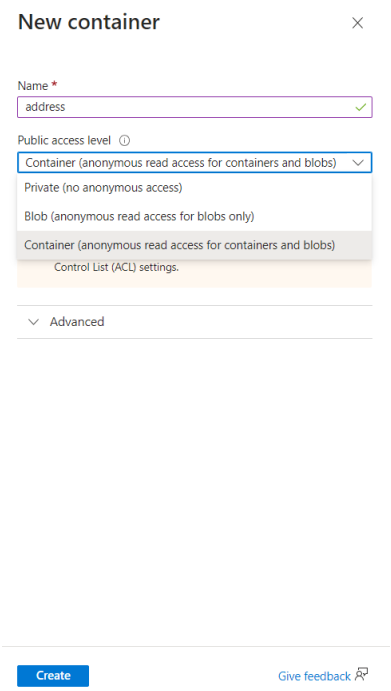
In the advanced section, ensure that you enable the hierarchical namespace by default, if it's not enabled, a hierarchical namespace blob storage account will be created.

All Networking, Data protection, Encryption and Tags will remain the same. Then, click on '**Review + create**' and follow the same steps as for the Data Factory resources.

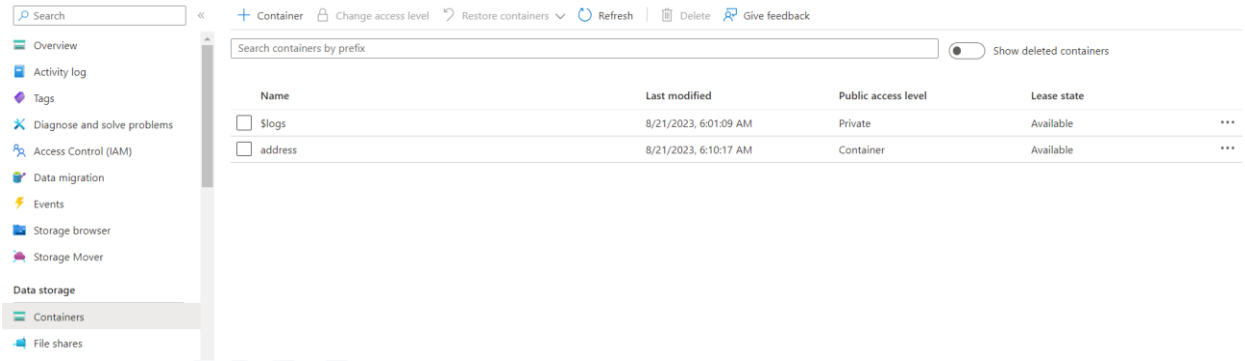


The Data Lake Gen2 UI should look like this. Then, click on "**Containers",** create a new container, and ensure that the container has public-level access set to '**container' (allowing anonymous read access for containers and blobs).** Finally, click on "Create".

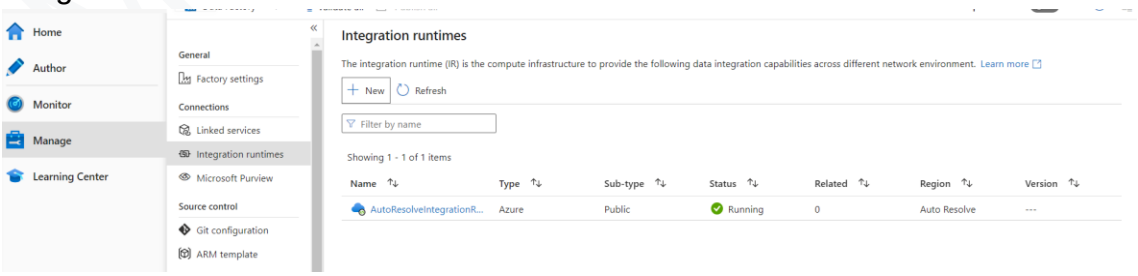




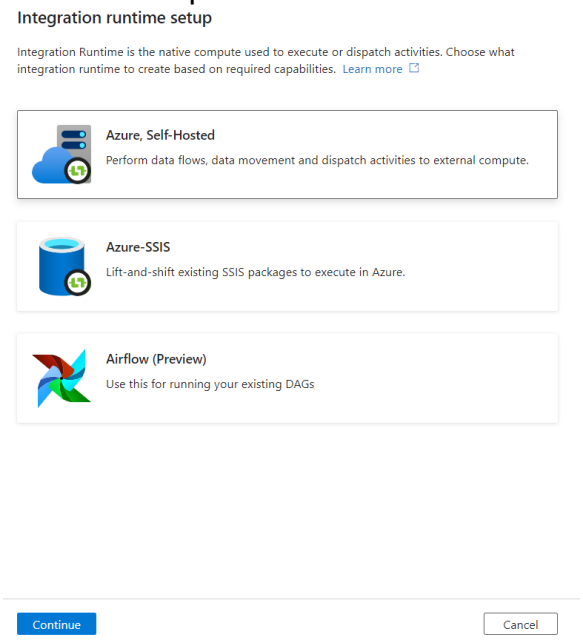
After that, the container would be created.

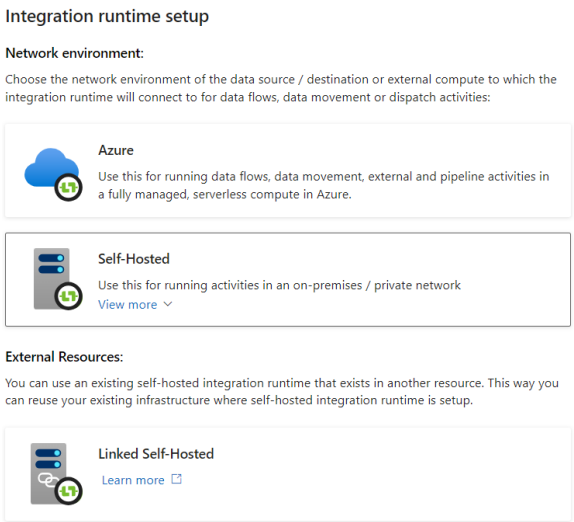


Click on Azure Data Factory Studio, then go to Manage → Integration Runtimes → New Integration Runtime.

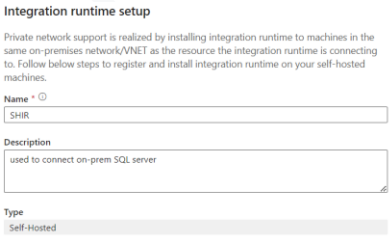


Next, choose '**Azure**' and '**Self-Hosted**,' then click '**Continue**.' Afterward, select '**Self-Hosted**' and proceed.

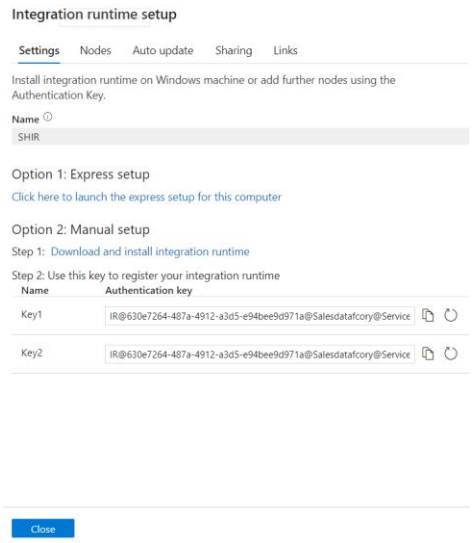




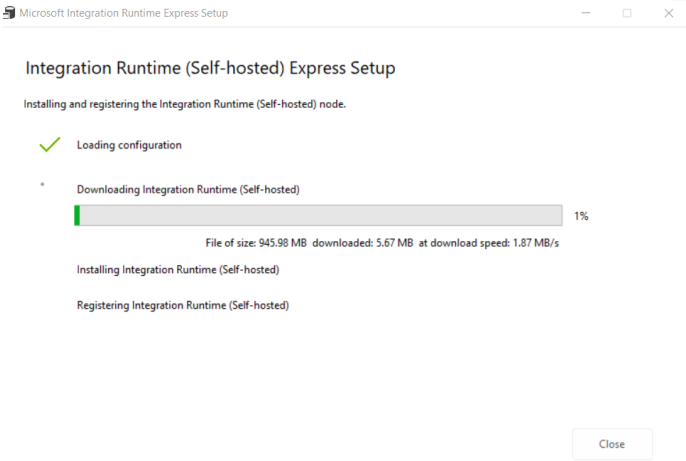
Next, provide a name for the Integration Runtime, include a description if necessary, and then click on **'create'**.



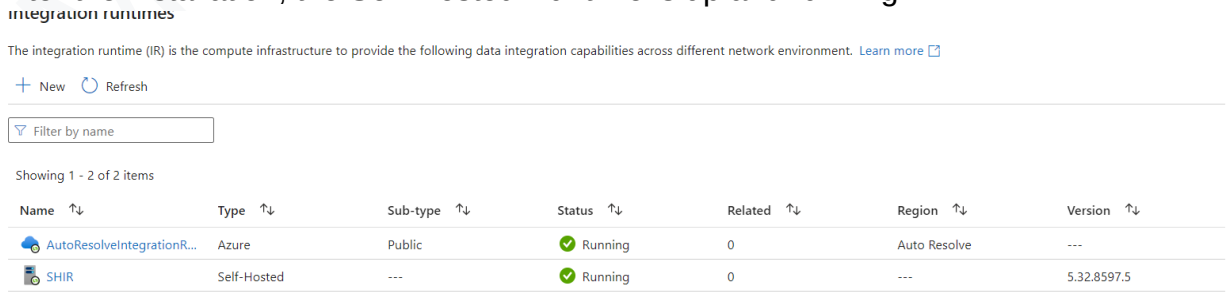
Now, there are two options: click on Option 1, Express Setup.



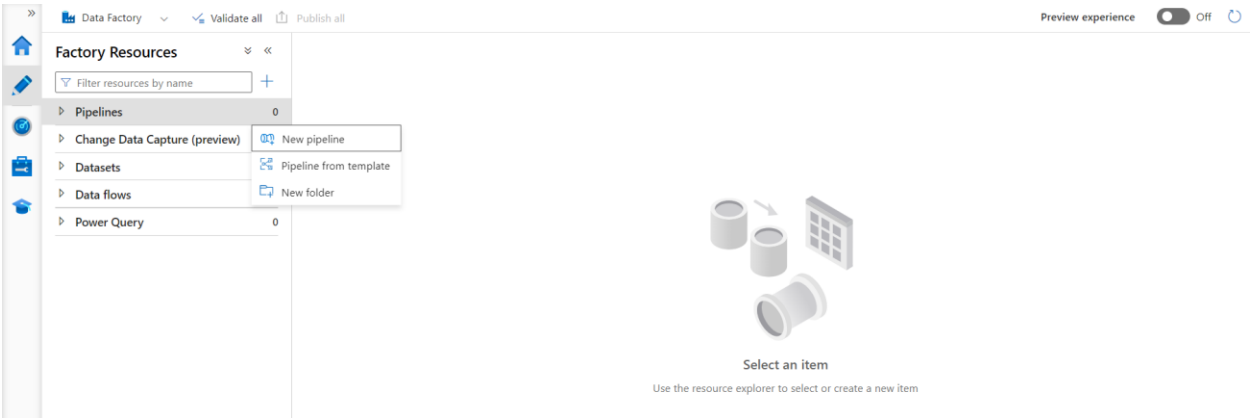
After clicking on the option, the application will be downloaded to your local computer, and when you run the application, the Microsoft Integration Runtime will be installed on your computer.



After the installation, the Self-Hosted Runtime is up and running.

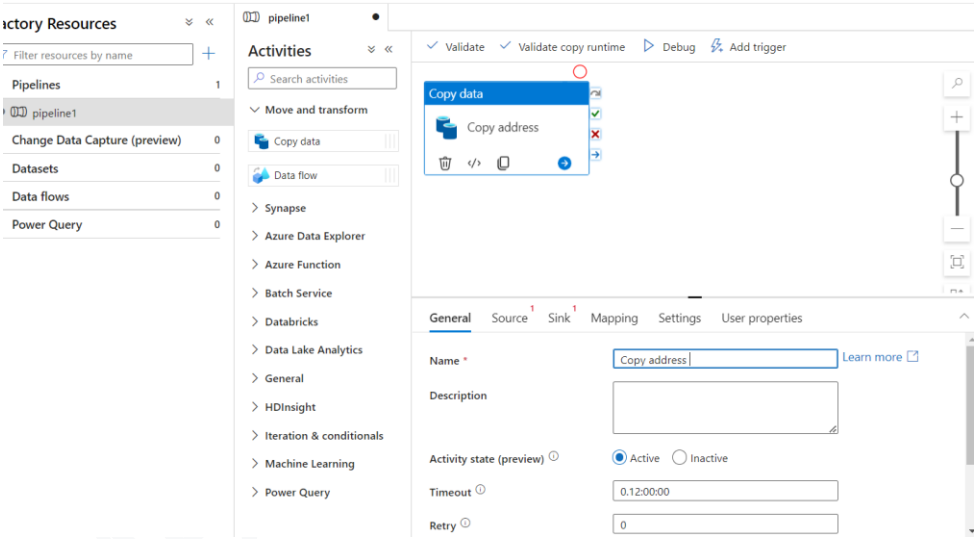


Go to the **Author** section and create a new pipeline.

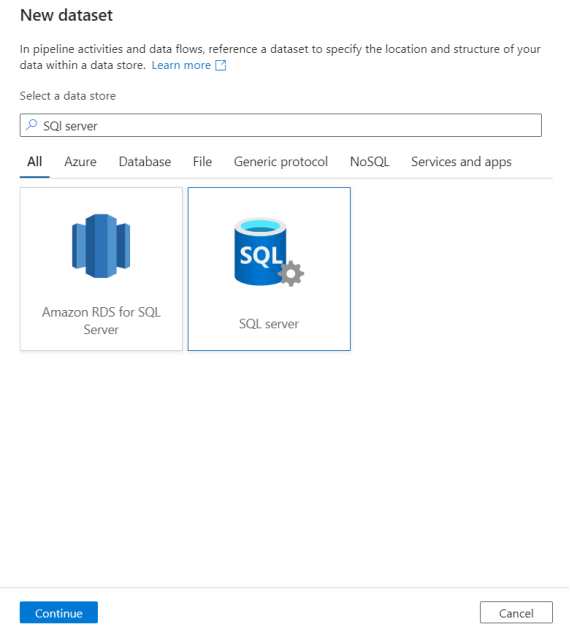


For Single Table:

Copy the address table from the on-premises SQL server to Azure Data Lake Gen2. To do this, select '**Move and Transform**,' drag **'Copy Data**' onto the canvas, and give it a name.

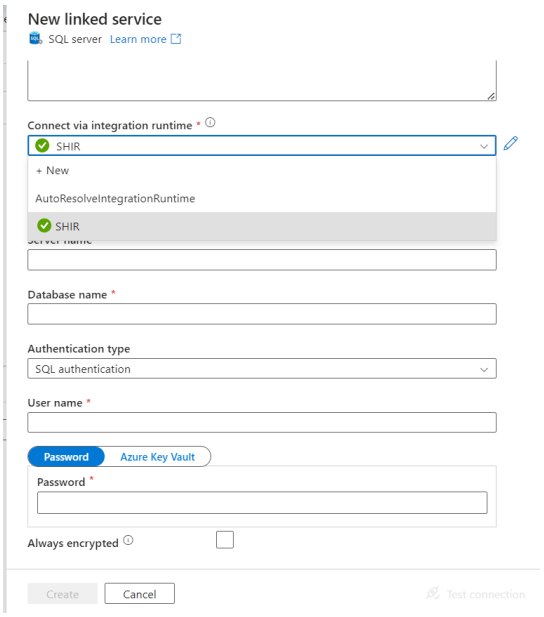


Click on '**Source**' → '**New**' → 'Add source to your pipeline for data ingestion.' Use SQL Server because the given data is on-premises here. Click on '**Continue**.'

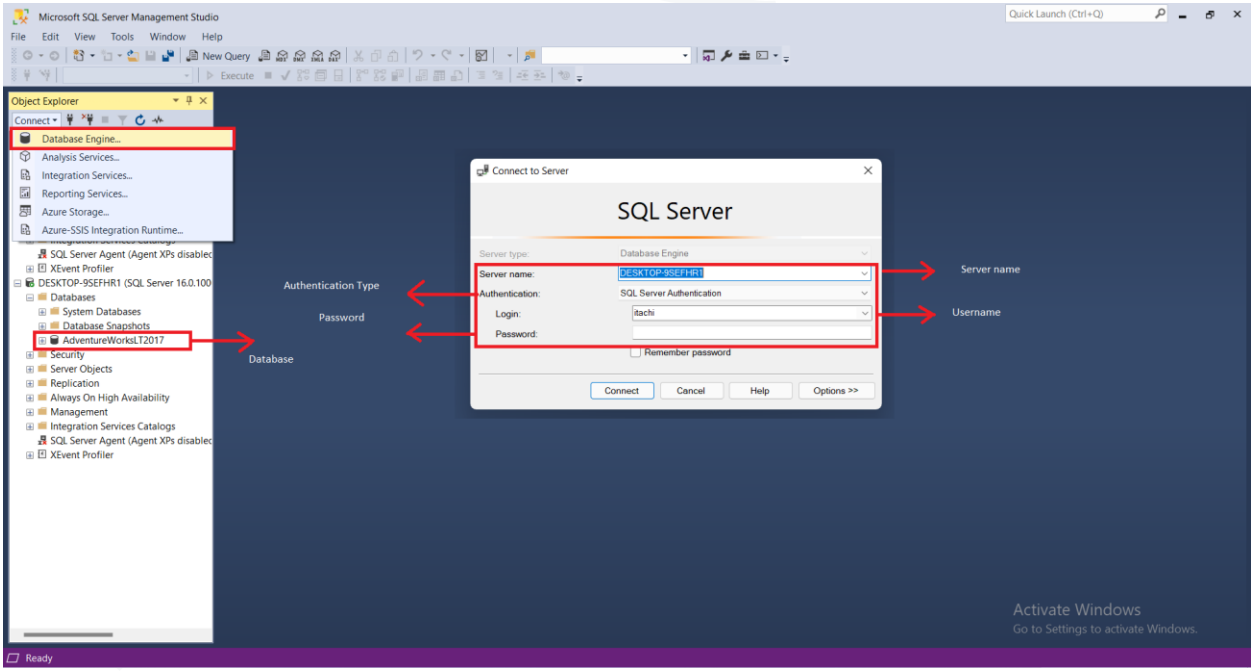


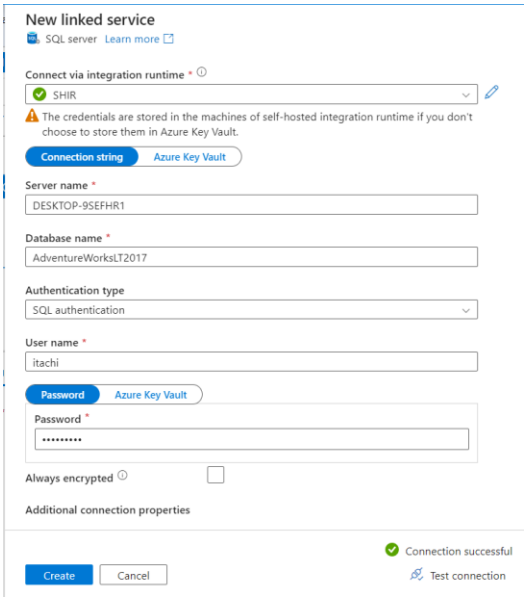
Give it a name, then select **'New'** in the linked service.

Connect via the integration to Suntime→SHIR.

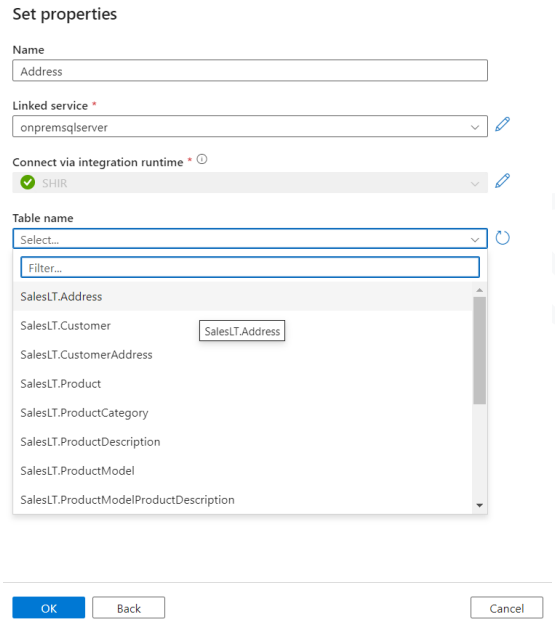


Fill in the following details.

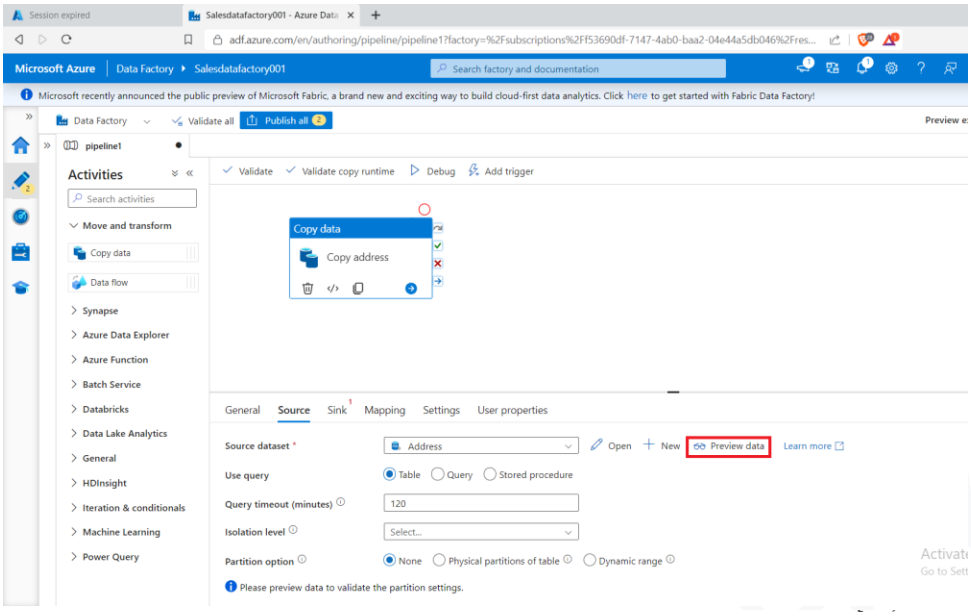




Click on '**Create**' and in the '**Table name**' field, enter '**SalesLT.Address**', then click '**OK**'.



Preview the data.



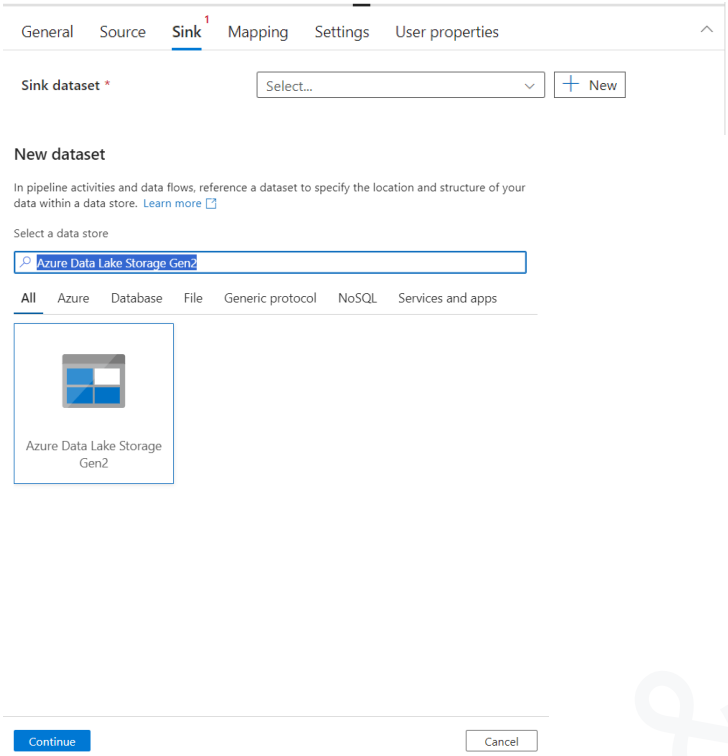
Preview data

Linked service: onpremsqlserver

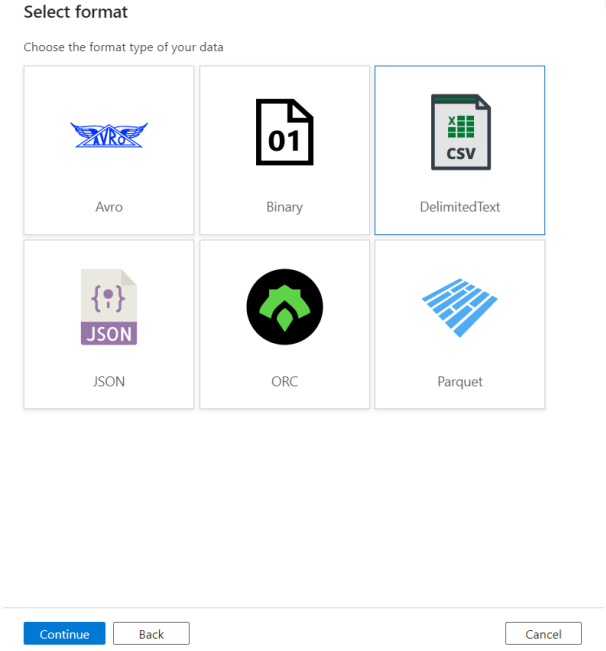
Object: SalesLT.Address

	AddressID	AddressLine1	AddressLine2	City	StateProvince	CountryRegion	PostalCode	rowgu
1	9	8713 Yosemite Ct.		Bothell	Washington	United States	98011	268af 76d7- 9441- 144fd
2	11	1318 Lasalle Street		Bothell	Washington	United States	98011	981b3 aca2-- 9a96- fb670
3	25	9178 Jumping St.		Dallas	Texas	United States	75201	c8df3 48f0-4 a8dd- 14a67
4	28	9228 Via Del Sol		Phoenix	Arizona	United States	85004	12ae5 fc3e-4 9b92- 3b970
								84a95

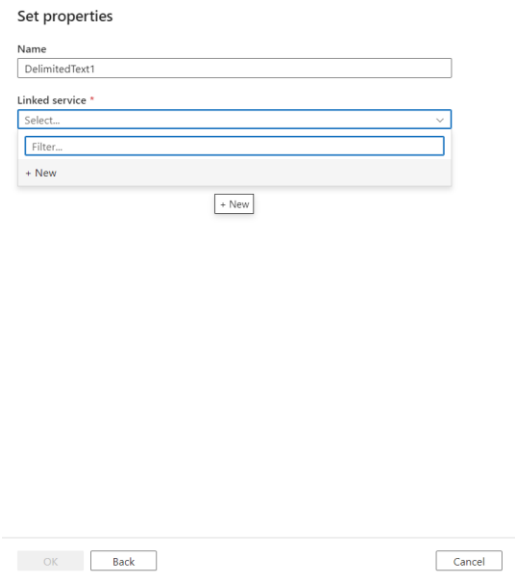
Go to Sink → New → Data Lake Gen2 → Continue.



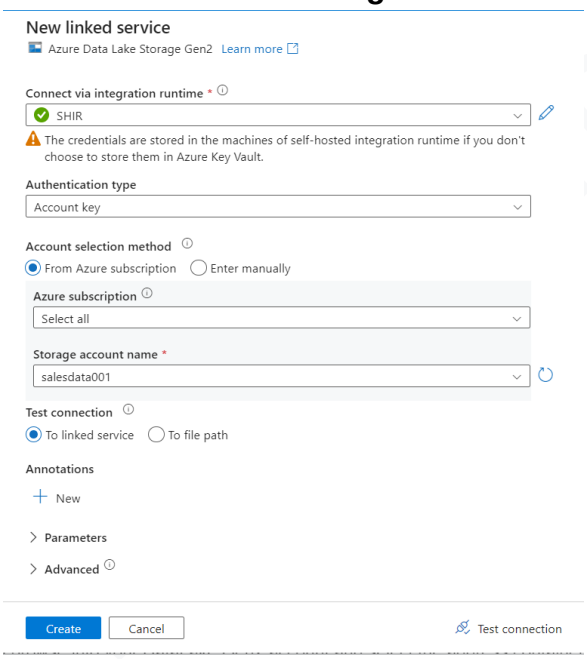
Select '**Delimited Text'** (it's a text format) and then continue.



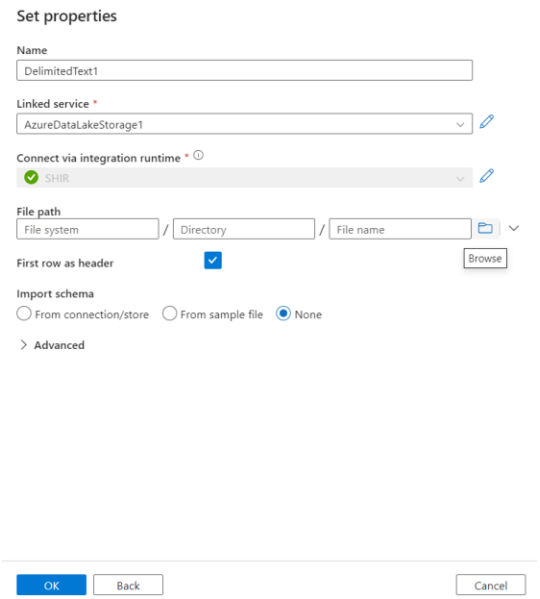
Create a new linked service.



In the **'New Linked Service'** section, navigate to **'Connect via Integration Runtime'** and choose **'Self-Hosted Integration Runtime (SHIR)'**.

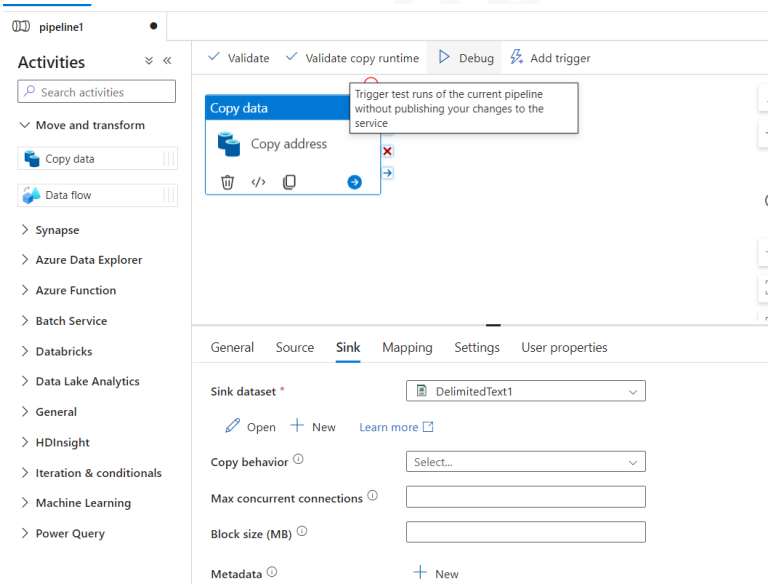


For **'Authentication Type'**, select **'Account Key'**.
Under **'Account Selection Method'**, choose **'From Azure Subscription'**.
Enter the **'Storage Account Name'** as the name of your Data Lake Gen2 account, then click **'Create'**.
Make sure to select 'SHIR'.

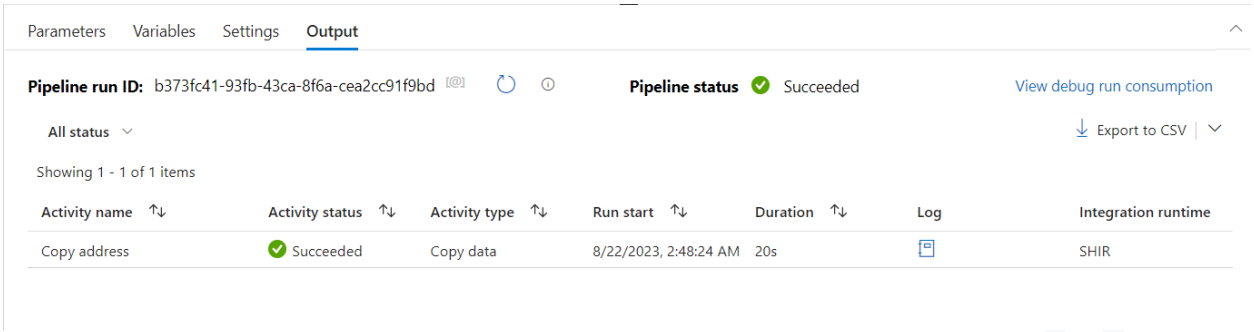


Click **'Browse'** in your Data Lake Gen2 account, then navigate to the **'address'** container, and click **'OK'**.

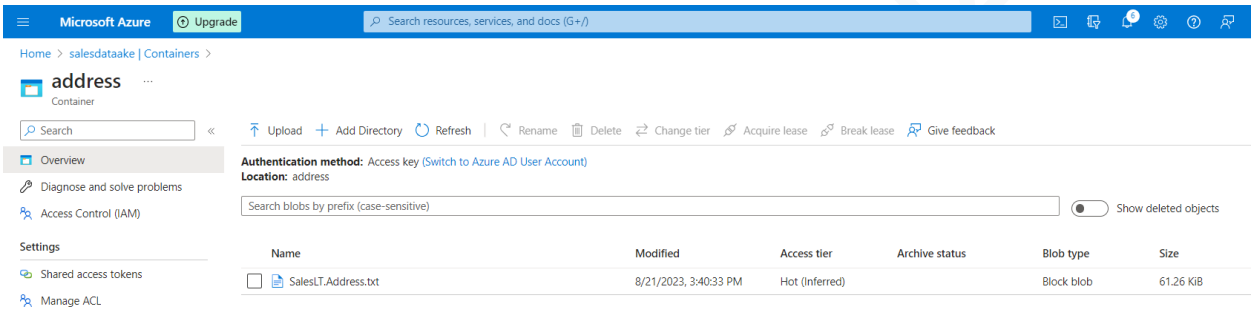
Click on **'Debug'** to run your pipeline.



Check the activity status to determine if the pipeline was successful or not.



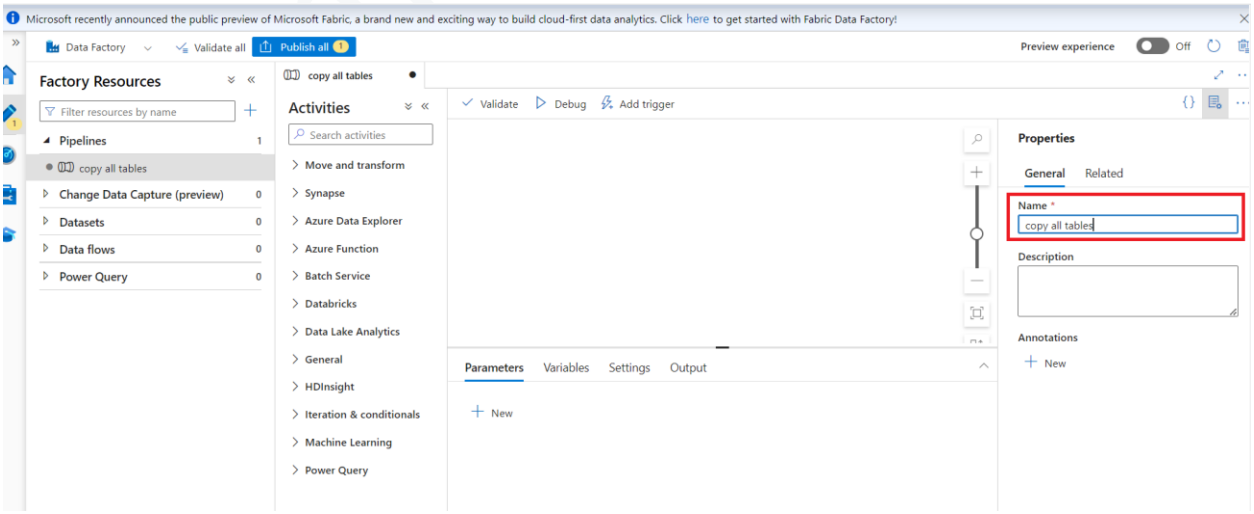
Once the pipeline runs successfully, the SalesLT.Address file will be copied into an address container in an Azure Data Lake Gen2 account.



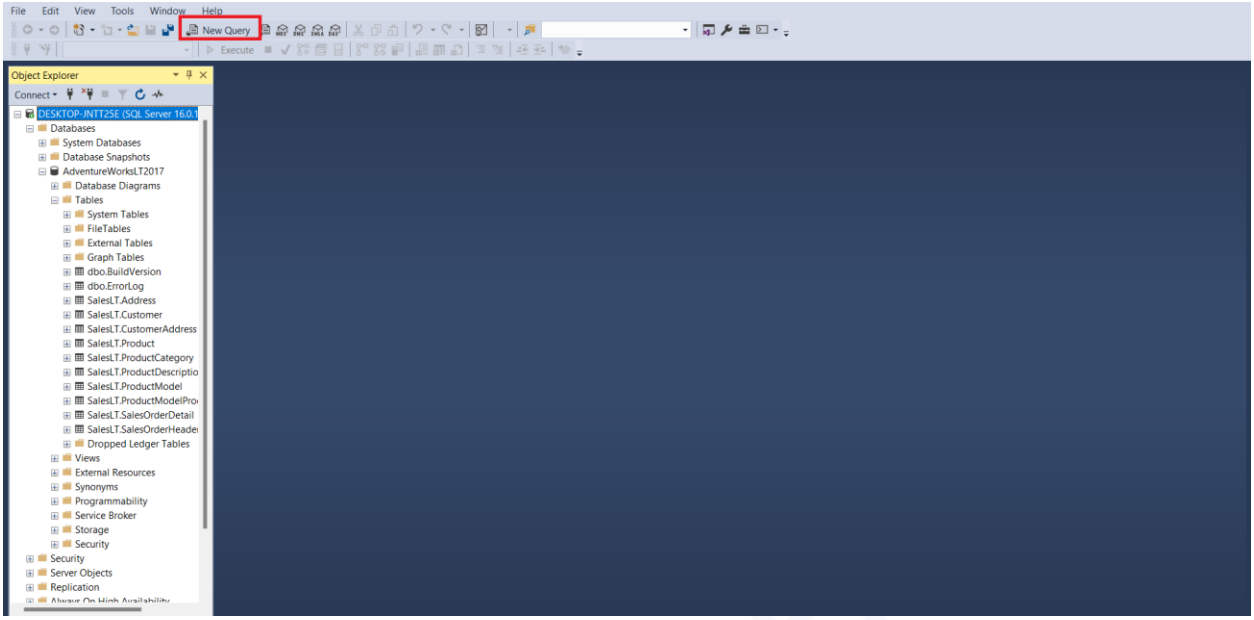
we have to ingest all the tables from on-prem SQL server to Azure one by one.

For Multiple Tables:

Create a new pipeline and assign it a name.

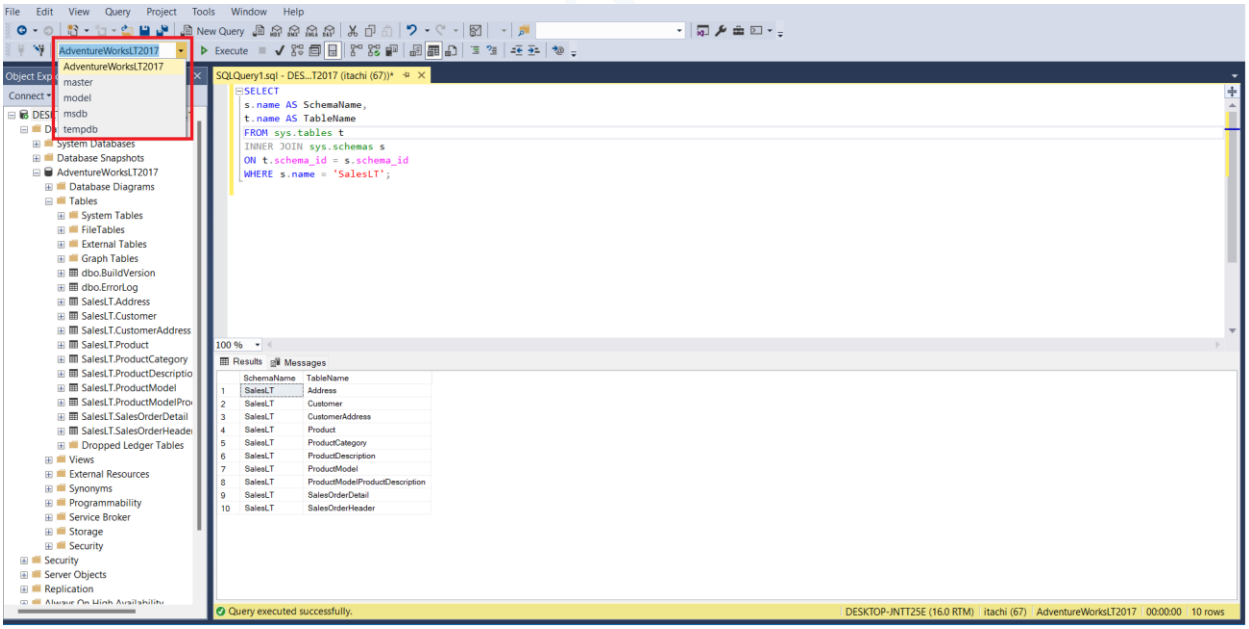


Go to SQL Server management studio → new query.



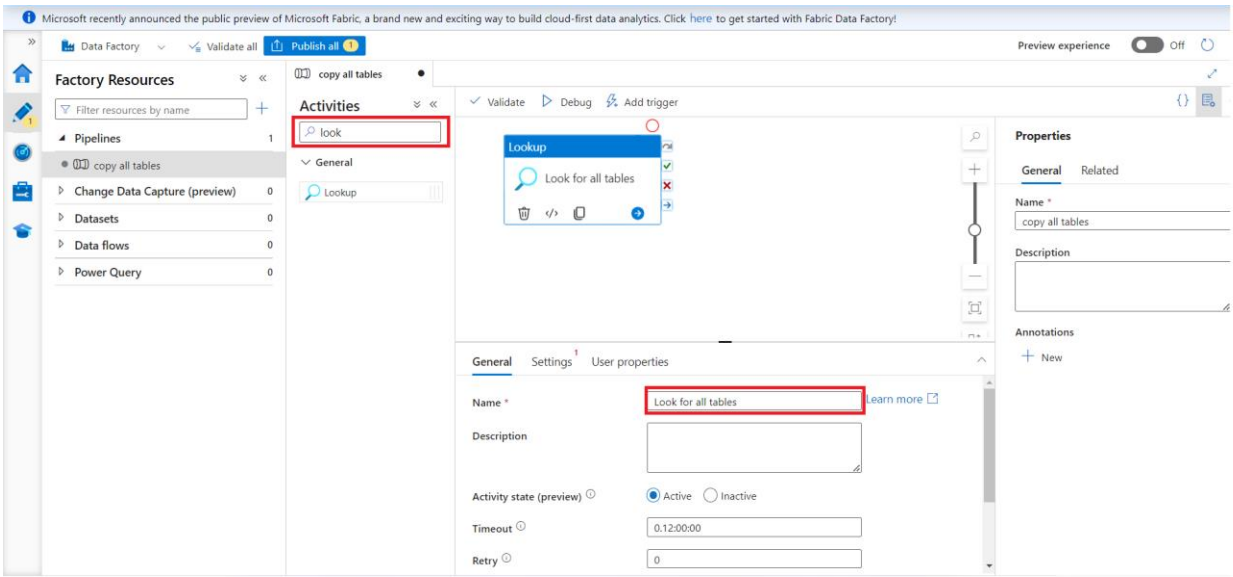
Write a query to select all tables from the SalesLT schema in the AdventureWorksLT2017 database.

NOTE: This query is intended for a lookup activity to check its executability.

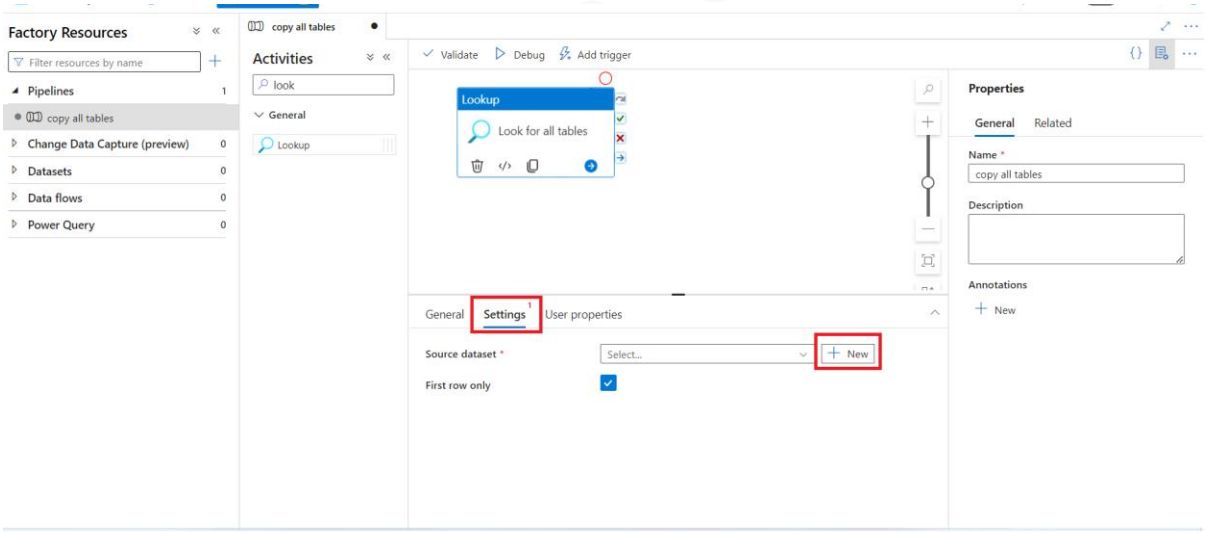


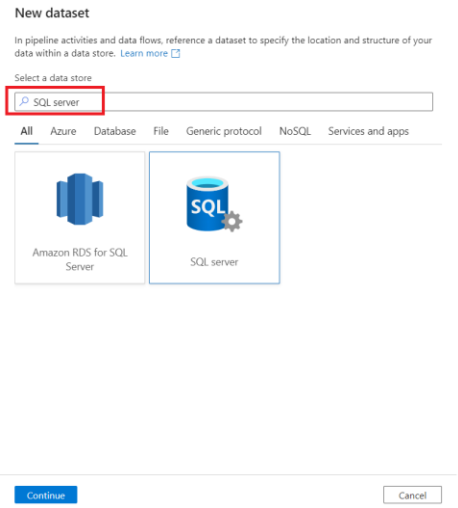
Navigate to Azure Data Factory Studio, go to Activities, search for the Lookup activity, drag it onto the canvas, and provide it with a name.

Official Documentation for Lookup:
https://learn.microsoft.com/en-us/azure/data-factory/control-flow-lookup-activity

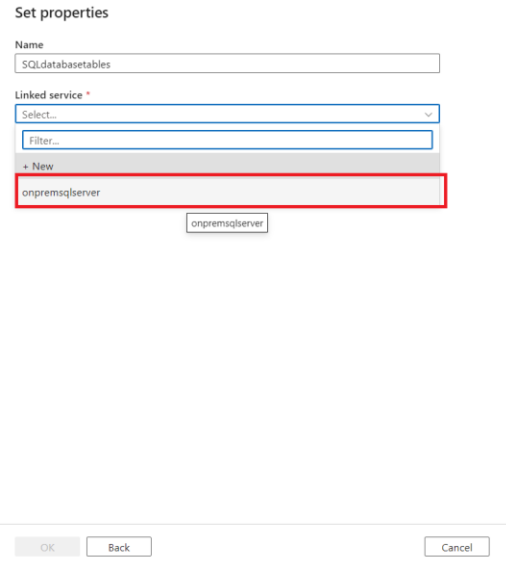


Go to Settings, then select New, followed by SQL Server, and finally, continue.

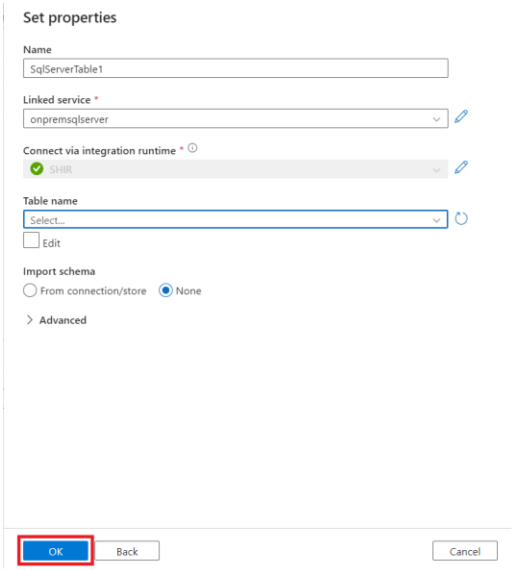




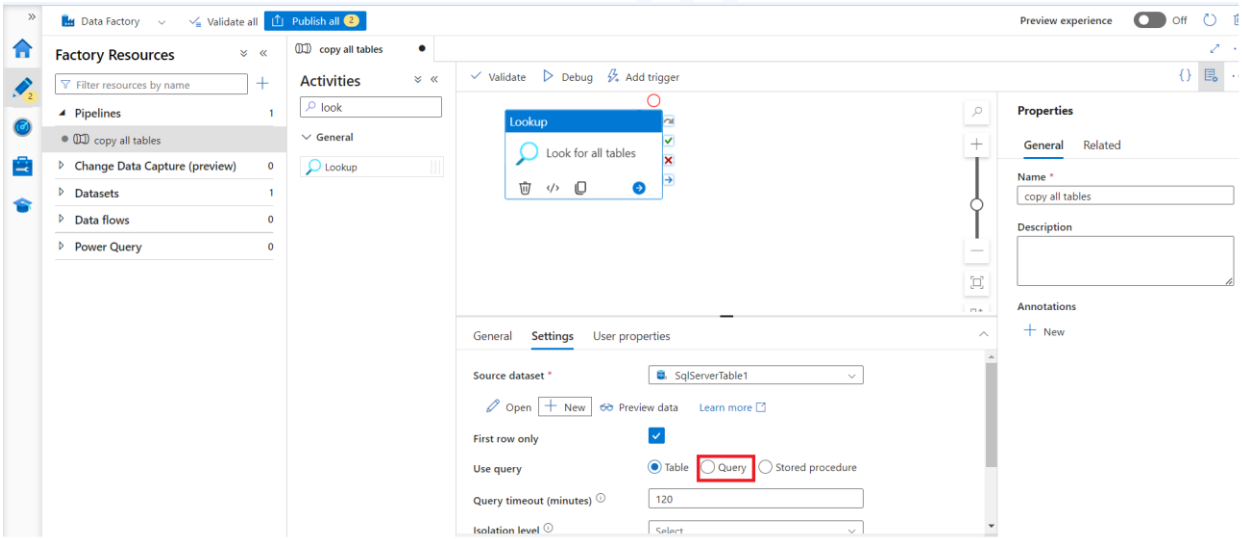
Give it a name, and in the linked service section, select the SQL Server that was created before, then click OK.



In this case, we are copying all the tables, so just click **'OK'**.



In the settings, click on **'Select Query.'**

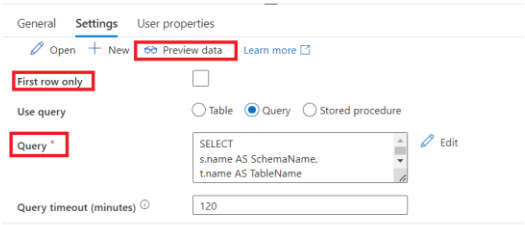


In the query editor, replicate the same query to retrieve all the SalesLT tables, unchecking the **'First row only'** option, and then click on **'Preview Data'**.

Query:

```
SELECT
s.name AS SchemaName,
t.name AS TableName
FROM sys.tables t
INNER JOIN sys.schemas s
```

```
ON t.schema_id = s.schema_id
WHERE s.name = 'SalesLT';
```



It previews all Table Names and their corresponding Schema Names.

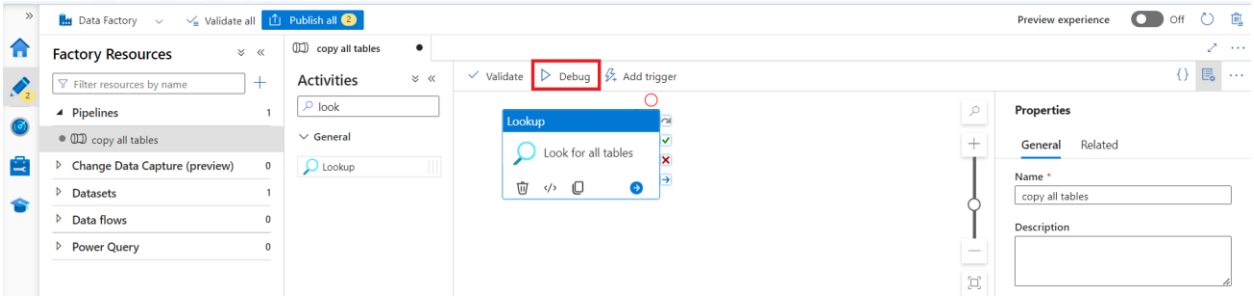
Preview data

Linked service: onpremsqlserver

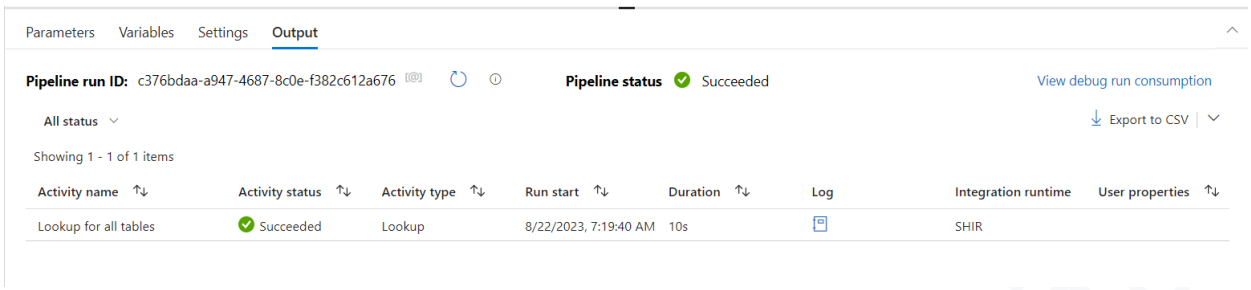
Object:

	SchemaName	TableName
1	SalesLT	Address
2	SalesLT	Customer
3	SalesLT	CustomerAddress
4	SalesLT	Product
5	SalesLT	ProductCategory
6	SalesLT	ProductDescription
7	SalesLT	ProductModel
8	SalesLT	ProductModelProductDescription
9	SalesLT	SalesOrderDetail
10	SalesLT	SalesOrderHeader

Please debug the pipeline to identify and fix any issues.



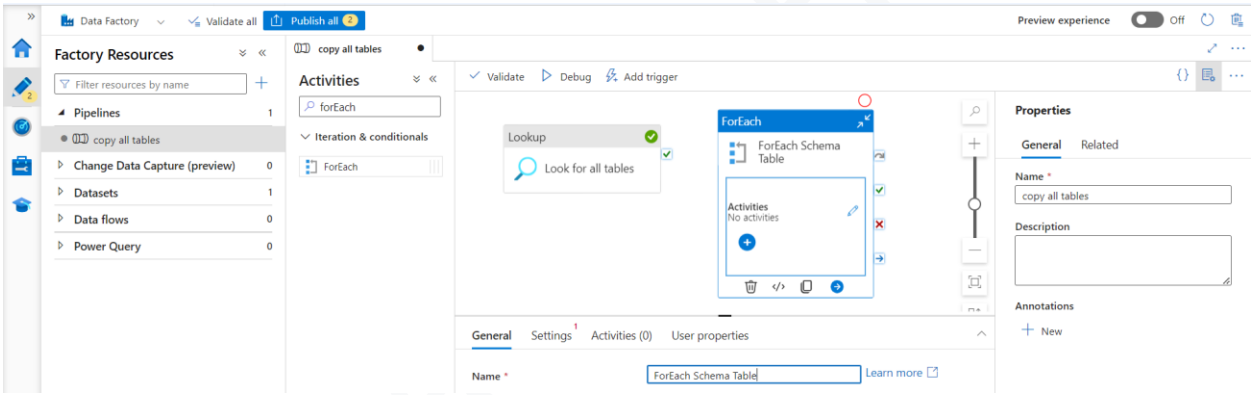
Check the activity status of the lookup activity. Has it succeeded or not?



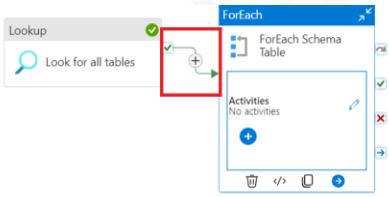
Use the ForEach activity to copy all the tables.

Official Documentation for the ForEach Activity:
https://learn.microsoft.com/en-us/azure/data-factory/control-flow-for-each-activity

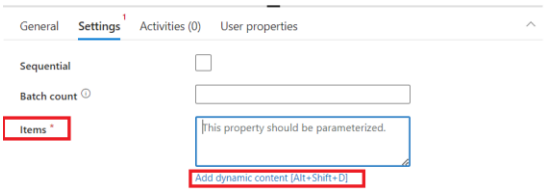
Search for the ForEach activity, drag it to the canvas, and name it.



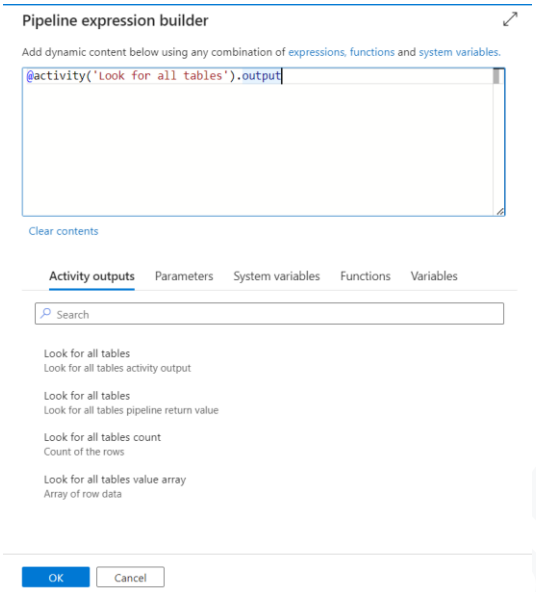
Connect the lookup activity to the ForEach Schema Table on success, which means that the ForEach activity will run when the lookup activity is successful.



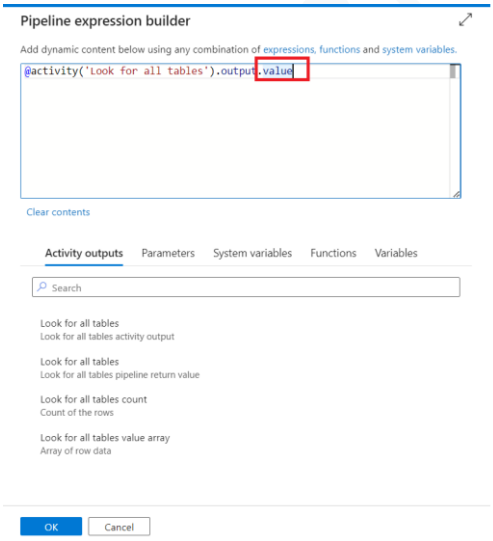
Navigate to the Settings → Items section and click on **'Add dynamic content.'**



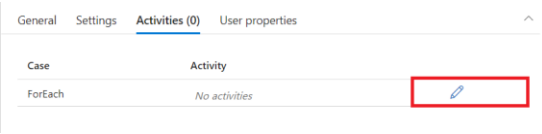
In Activity outputs, look for all the tables (using a dynamic expression).

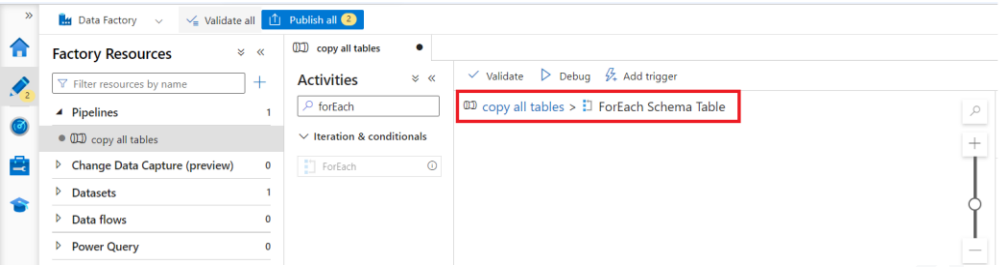


Add **'.value'** to the given dynamic expression, then click OK.

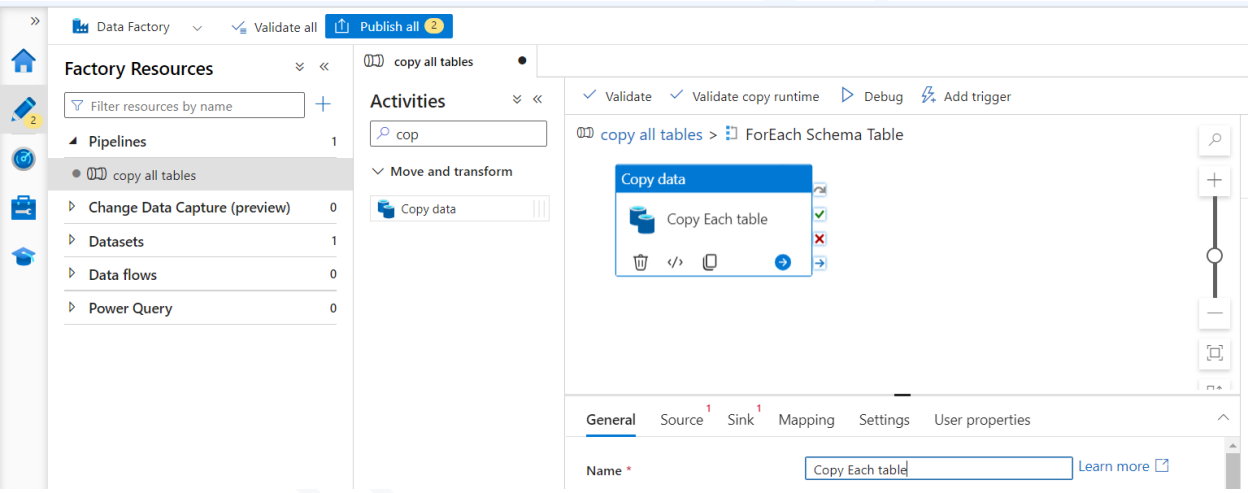


Navigate to the **'Activities'** section and then select the pencil icon.

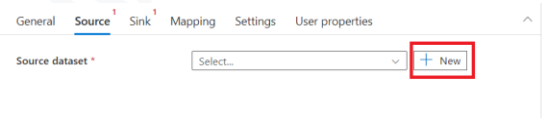


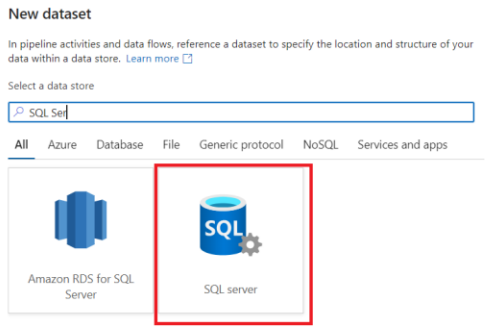


Search for the **'Copy Data Activity,'** drag it onto a blank canvas, and give it a name.

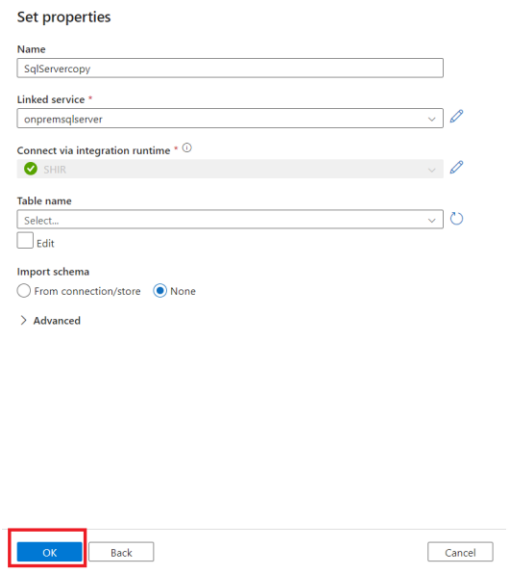


Click on Source → New → SQL Server → Continue → Linked Devices → Select Server Name → Click OK.

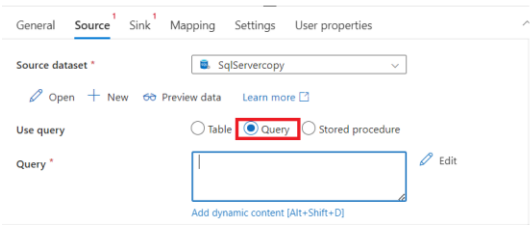




Provide the name and select the linked service **'onpremsqlserver,'** and confirm.



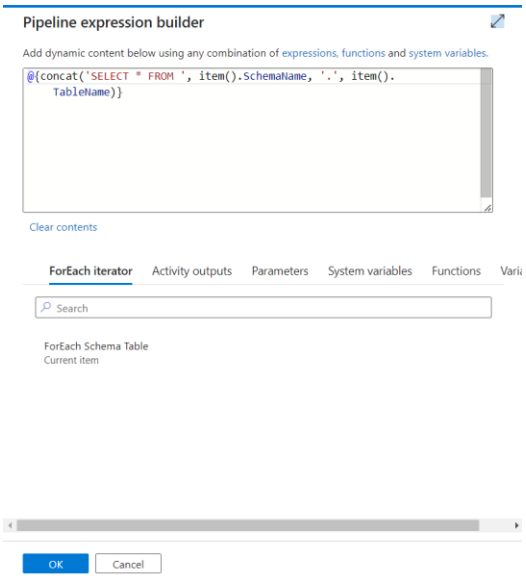
In the source section, select the **'Query'** option, then click the **'Add Dynamic content'** button, write the provided script, and finally, click **'OK'**.



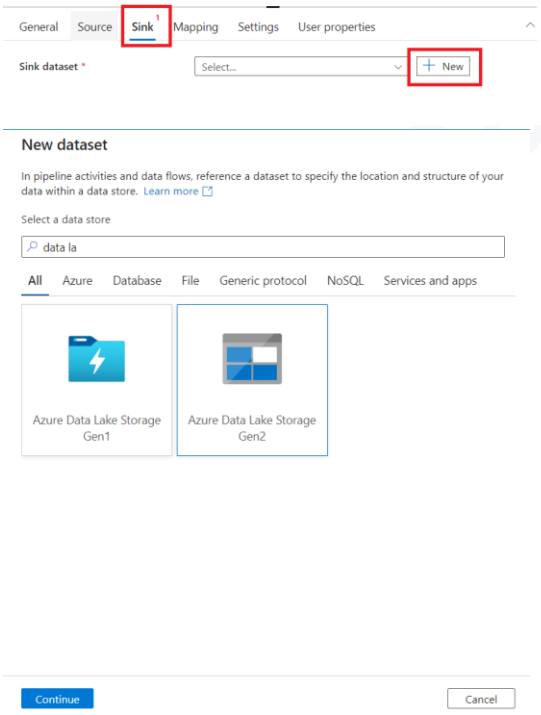
Script:

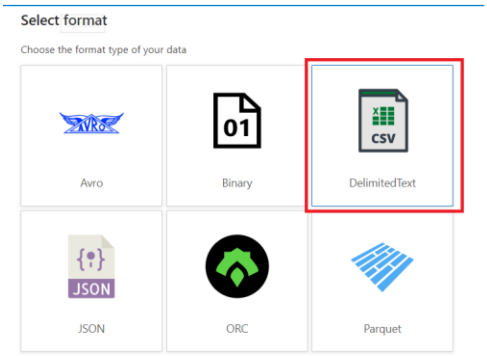
```
@{concat('SELECT * FROM ', item().SchemaName, '.', item().TableName)}
```

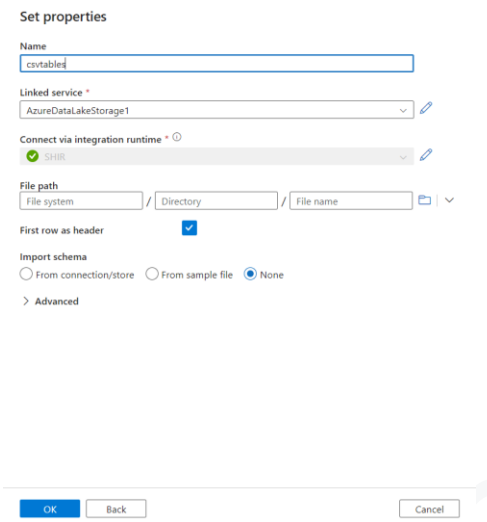
The given script is concatenating items, schema name, and table name.

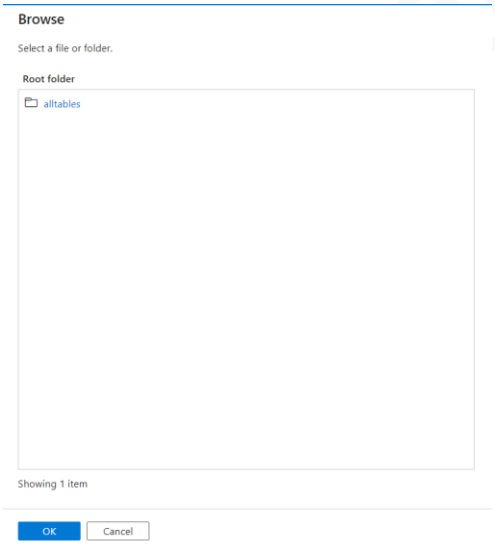


In the Sink section, click "**New**" → "**Azure Data Lake Gen 2**" → "**Continue**" → "**Delimited Text**" → "**Continue**" → "**Linked Service**" → select "**Azure Data Lake Storage**" → "**Browse**" → "**Container**" → "**OK**."



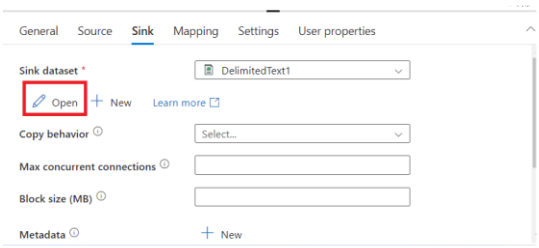

Select format
Choose the format type of your data
Avro
Binary
DelimitedText
JSON
ORC
Parquet


Set properties
Name
csvtables
Linked service *
AzureDataLakeStorage1
Connect via integration runtime *
SHIR
File path
File system
Directory
File name
First row as header
Import schema
From connection/store
From sample file
None
Advanced
OK
Back
Cancel

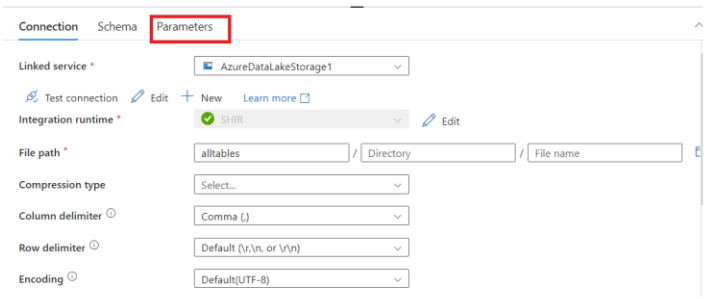

Browse
Select a file or folder.
Root folder
alltables
Showing 1 item
OK
Cancel

Sam & Huzaifa

Note:
To configure this folder structure in a data lake (e.g., alltables/SalesLT/Address.csv), we require two important pieces of information: a Table Name and a Schema Name.
In the Sink, click on **'Open.'**

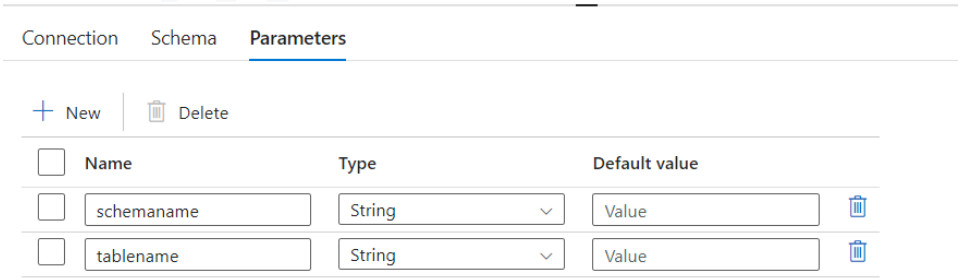


Click on **'Parameters'** and create two new parameters.

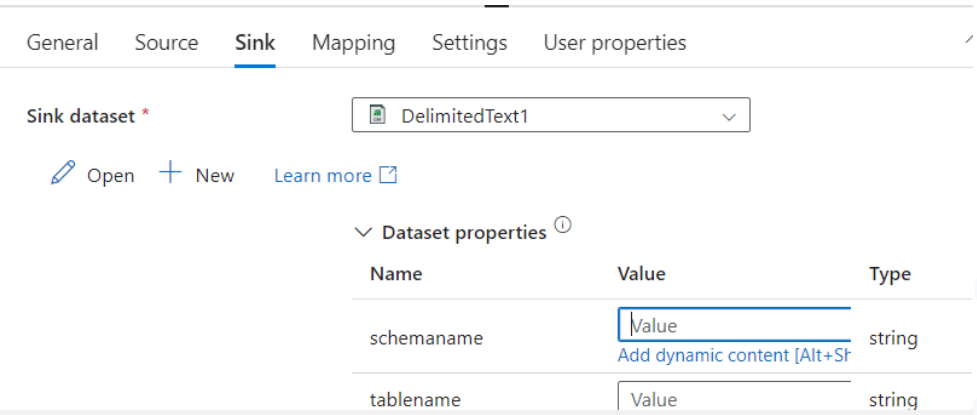


Click on **'New'** to create two new parameters.

Give the name for the parameter schemaname and tablename.



Return to the Sink section, where two new parameters have been added. Add the values by clicking on **"Add Dynamic Content,"** then write the script, and finally, click **"OK"** to confirm.



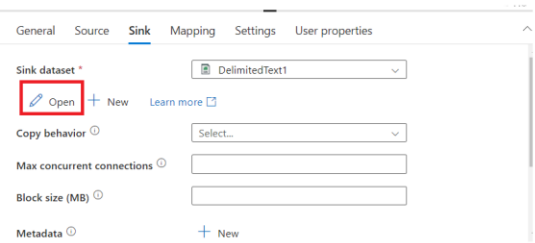
Script for schema name:

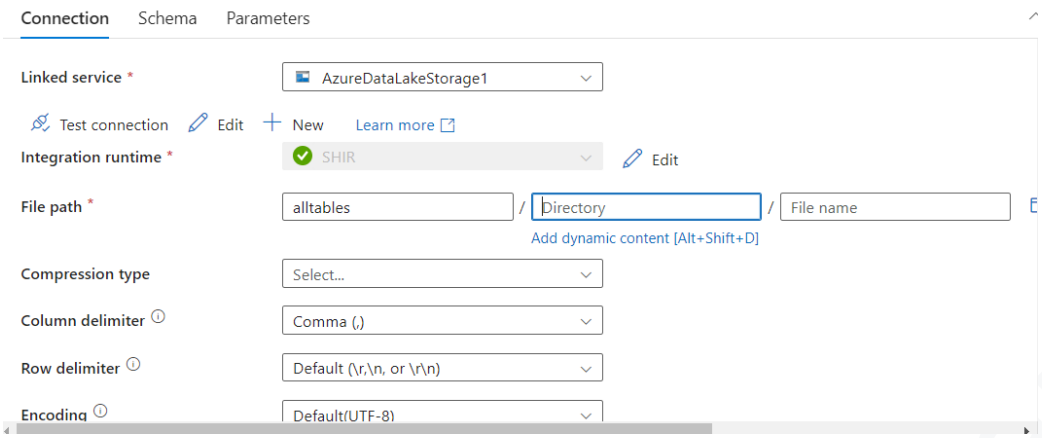
```
@item().SchemaName
```

Script for table name:

```
@item().TableName
```

In the Sink section, click on **"Open,"** then go to **"Connection,"** and select **"File Path."** Add dynamic content by writing the given script as per your requirements.





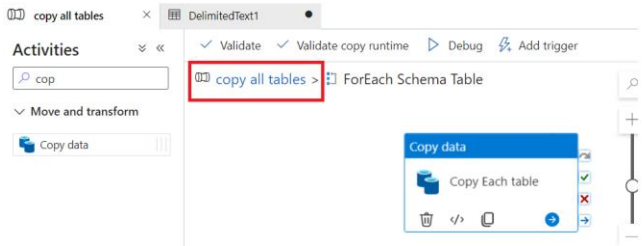
Script for Directory:

```
@{concat(dataset().schemaname, '/',dataset().tablename)}
```

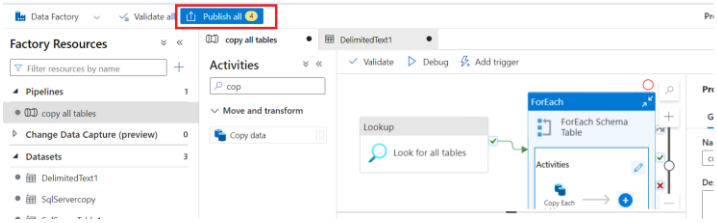
Script for File name:

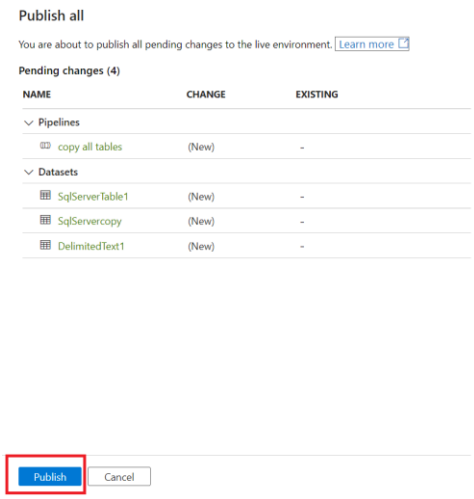
```
@{concat(dataset().tablename, '.csv')}
```

With all the configurations successfully set up in the copy data pipeline, proceed to the outer pipeline. Click on the "Copy All Tables" option to continue.

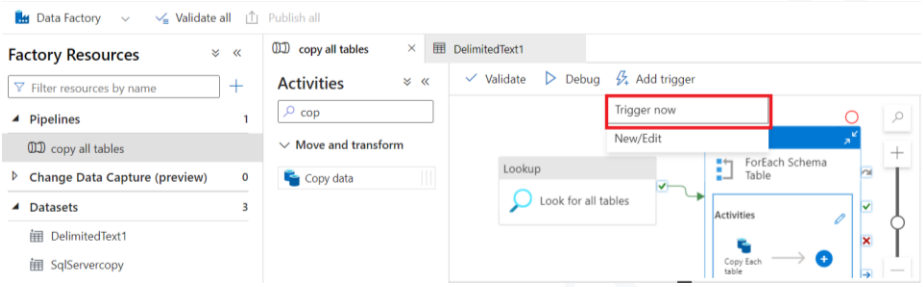


After making the necessary adjustments, then publish all the changes by clicking on the **"Publish"** button.

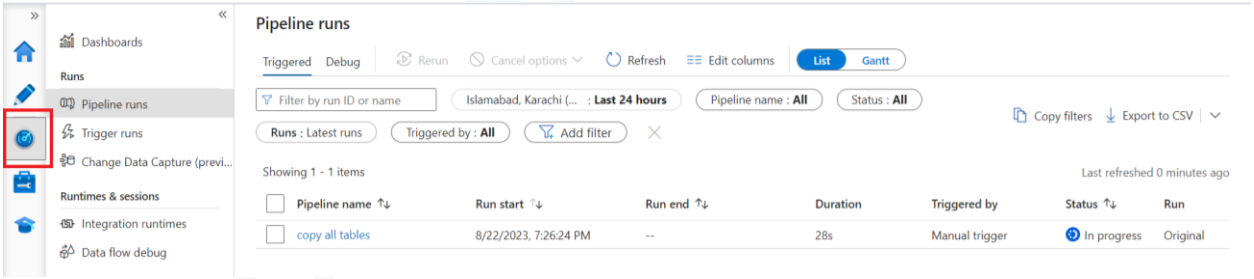


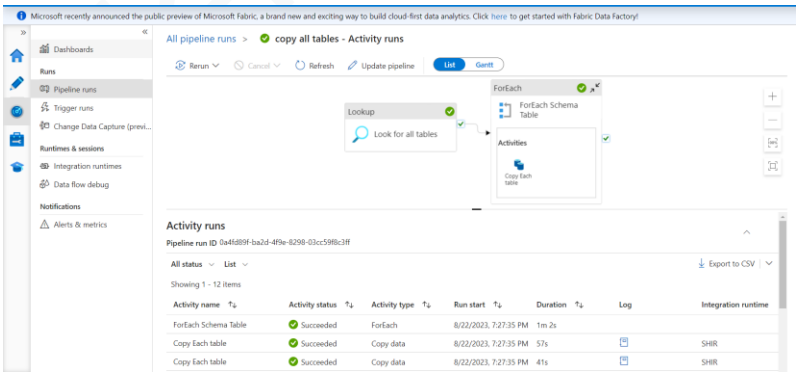


To run the pipeline, click on **"Trigger Now"**.

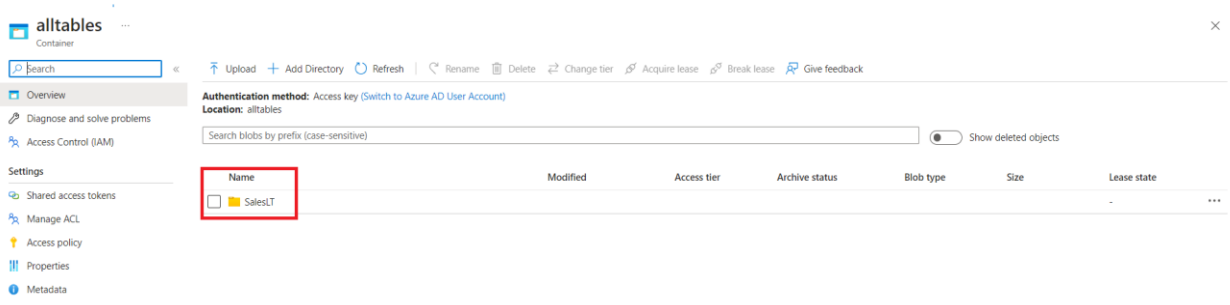


To view the pipeline status, click on the **"Monitor"** tab.

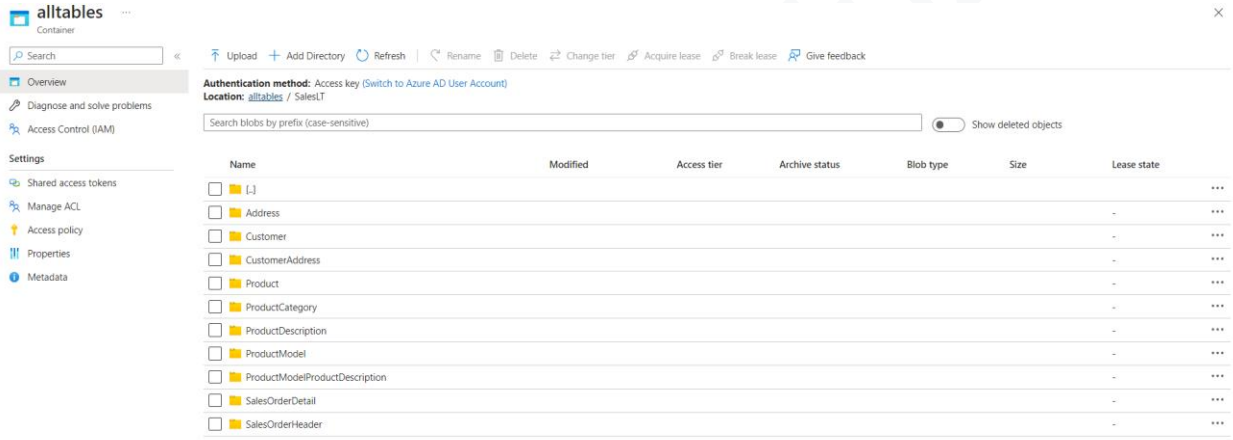




Navigate to the Data Lake Gen2 account, go to the **"All tables"** container, and you'll find the **"SalesLT"** directory added inside the container.



Inside the SalesLT directory, you have successfully ingested all the tables from the on-premises SQL server.



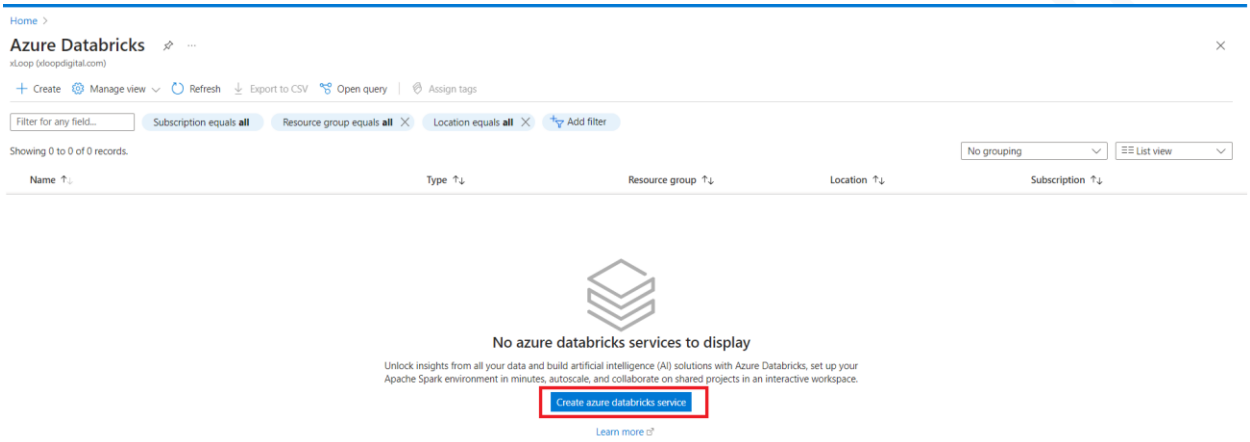
So congratulations for that we have completed our First part **“Data Ingestion”**. 😀

Part 2: Data Transformation

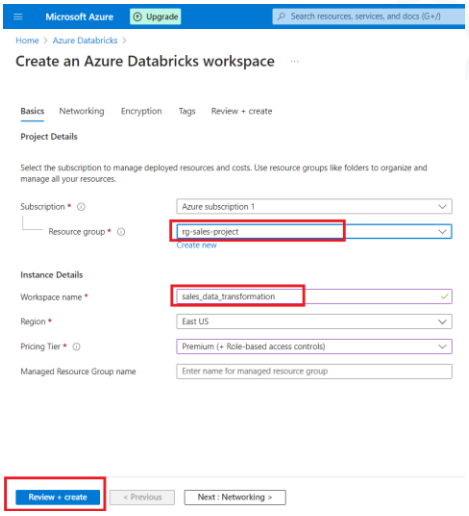
Azure Data Factory & Azure Databricks Integration and Transformation

For Data Transformation we have to create an Azure Databricks service.
Documentation:
https://learn.microsoft.com/en-us/azure/databricks/introduction/

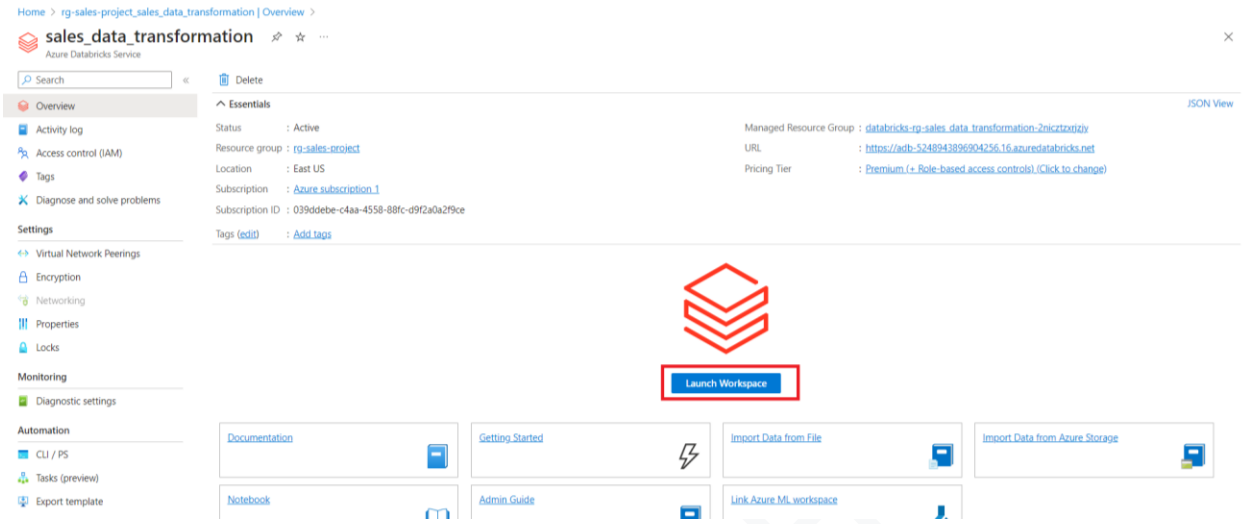
Create a new Azure Databricks service



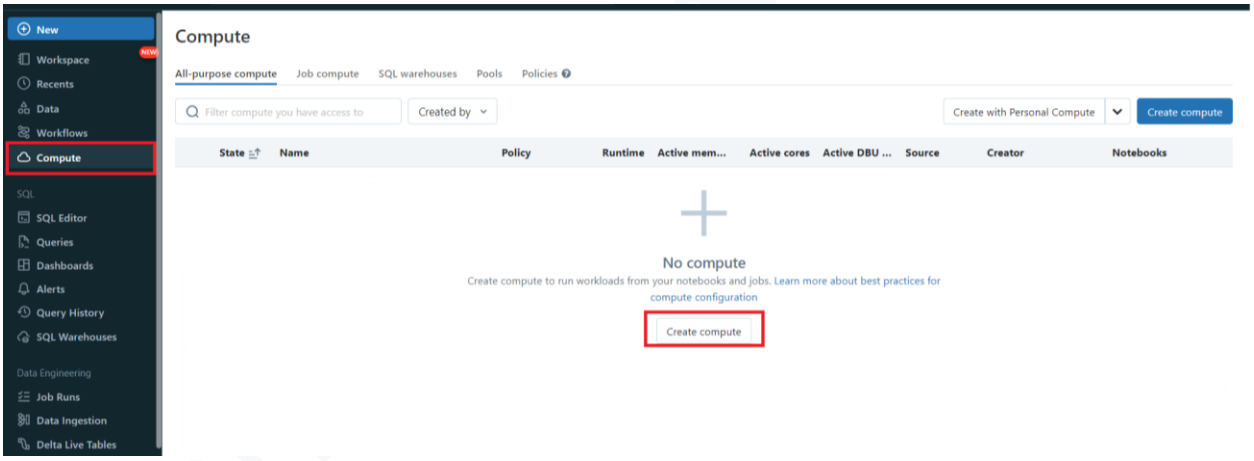
Select the existing resource group that was used in Data Ingestion and ensure all settings remain the same. Then, click on **'Review + create.'**



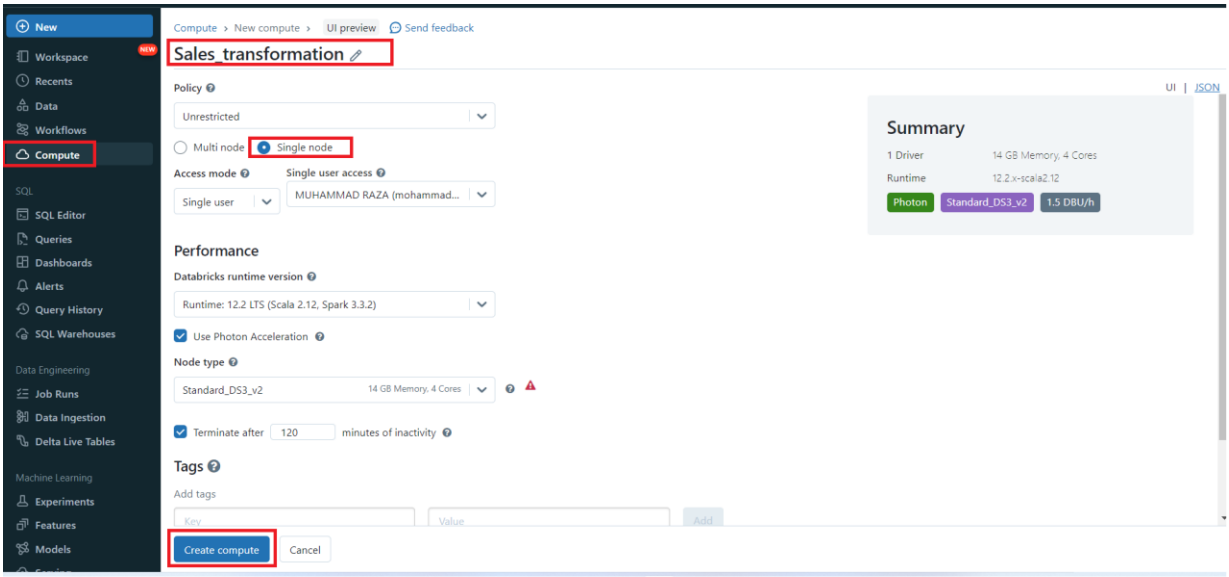
Next, navigate to **'Resources,'** and inside your Databricks service account, click on **'Launch Workspace.'**



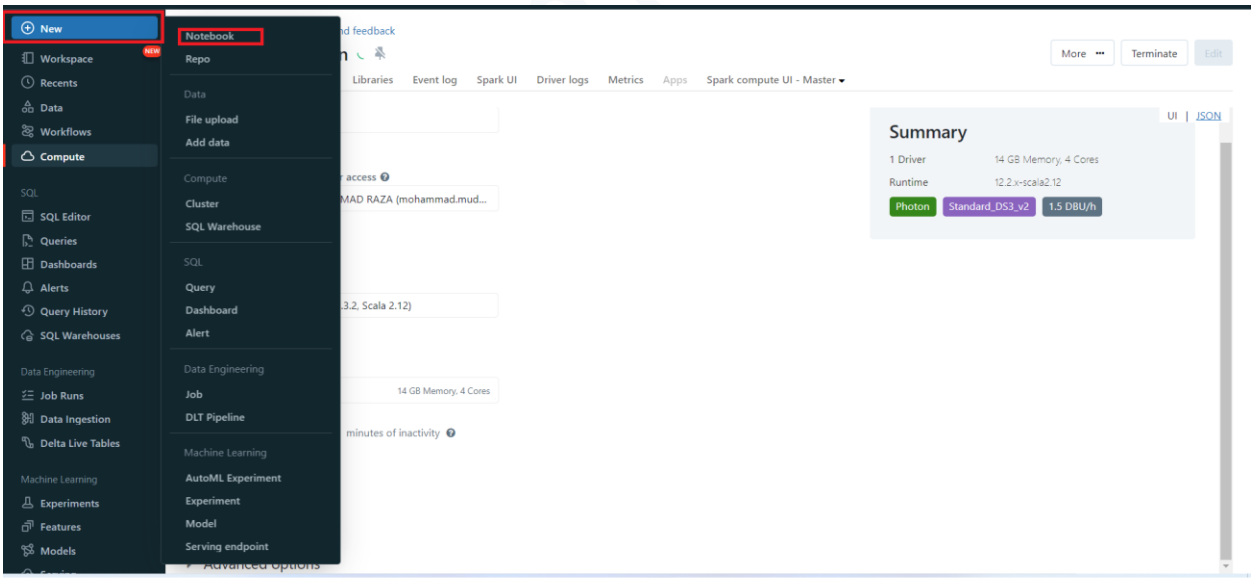
First of all, to perform data transformation, we need to create a Spark cluster. Go to the compute section and click on **'Create Compute'**.



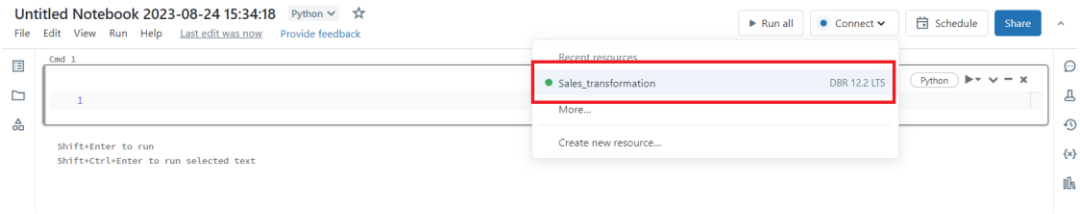
Provide your cluster name and please note that if you are using Azure on a free trial or Azure_subscription1, select a single node. Otherwise, you will need to upgrade your subscription to pay-as-you-go. All the configurations will remain the same. Finally, click on **'Create Compute'**.



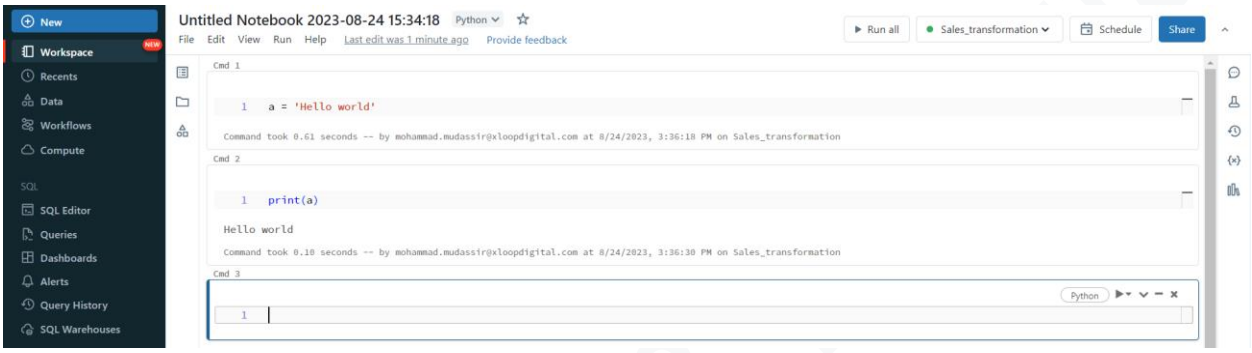
Now, create the new notebook for running the Data Transformation code.



Select the cluster that you have created.

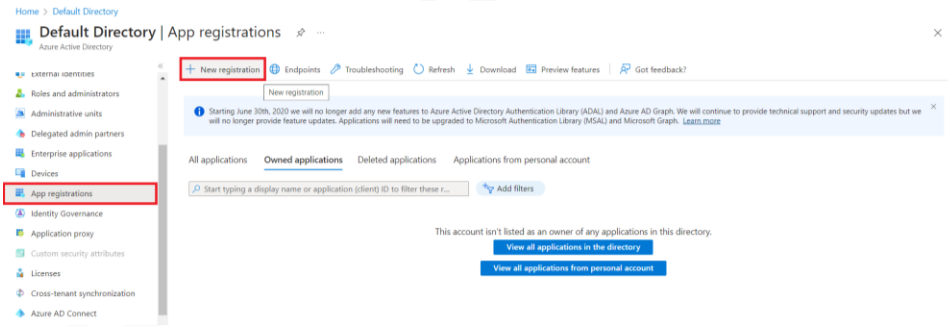


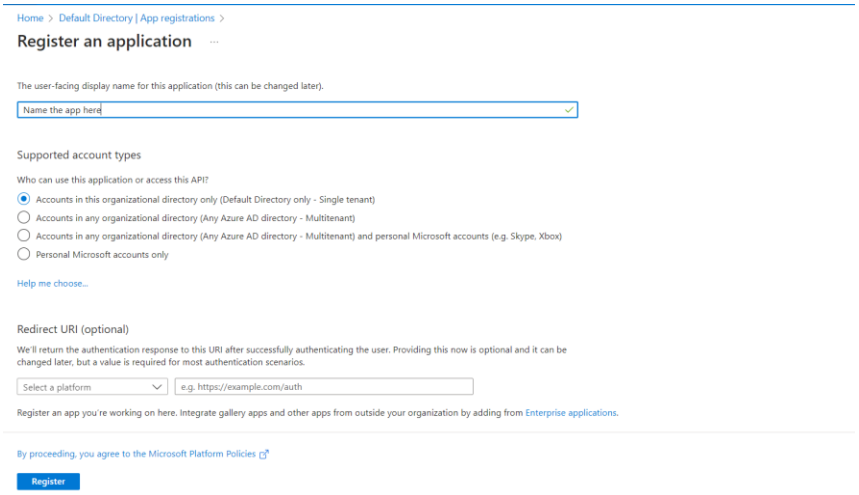
Write a basic Python code to check if the cluster is functioning or not.



Access to Databricks to Azure Data Lake Storage:

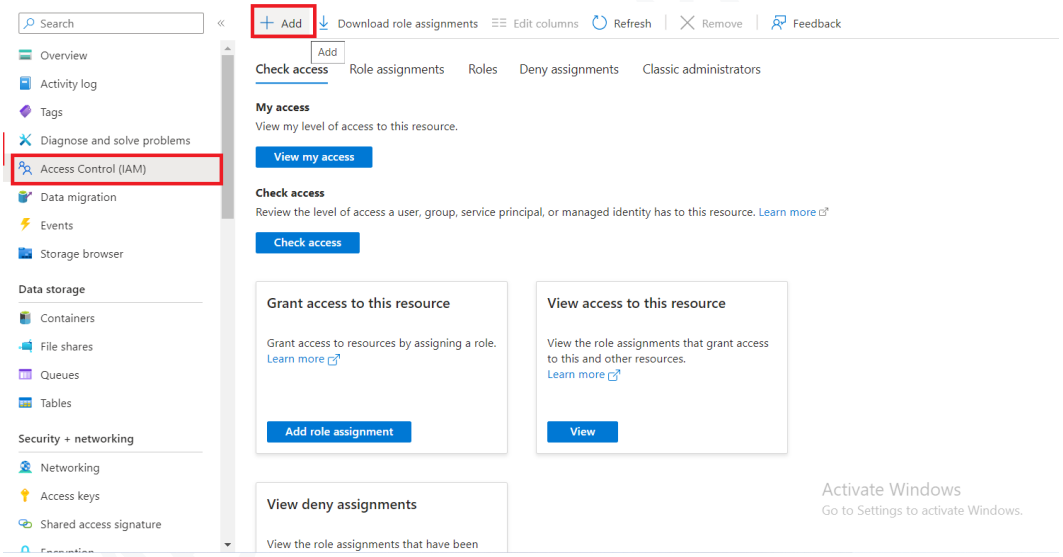
Step 1:
Go to Azure Active Directory → App Registration → New registration → Name it → Register.

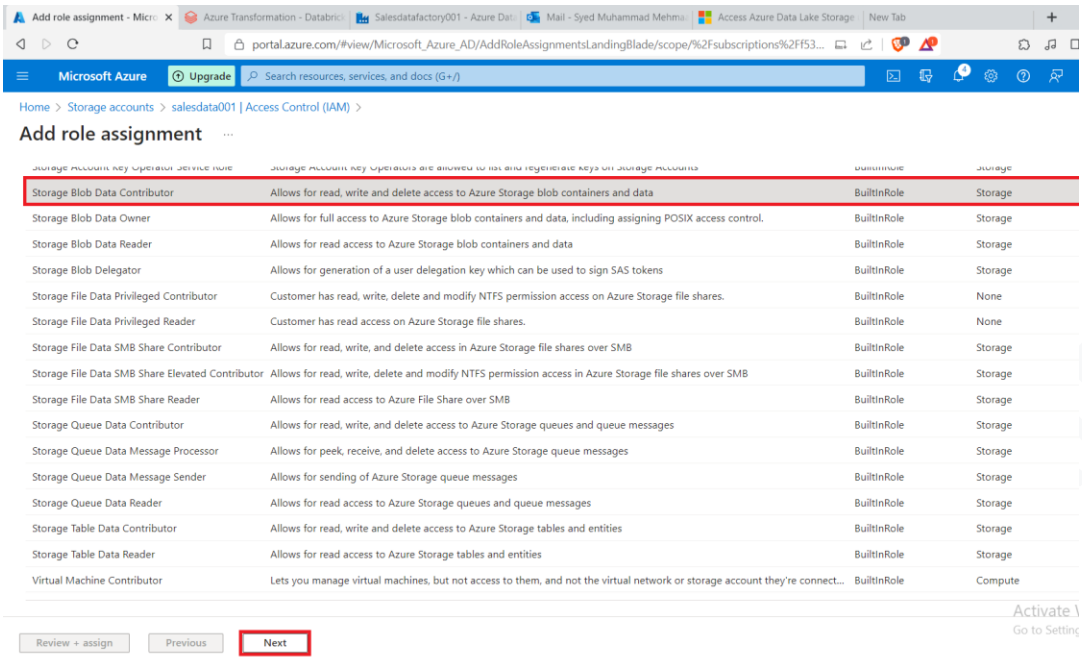


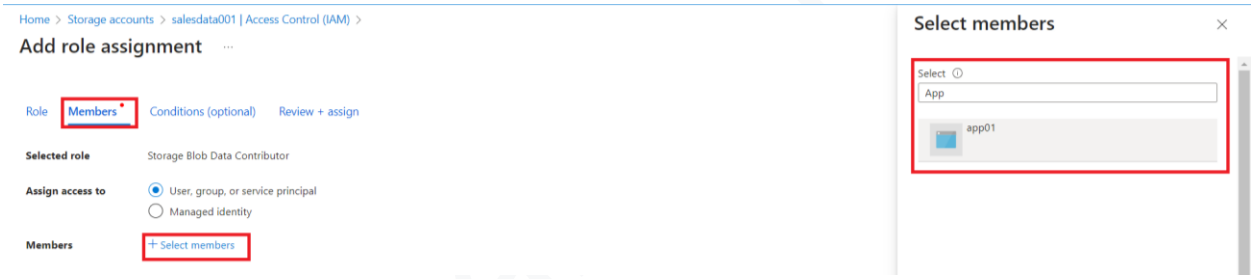


Step 2:

Go to Storage Account → IAM Access Control → Add role assignment → Storage Blob Data Contributor → Next → Members → select members → select the app registered in step 1 → Review and Assign.

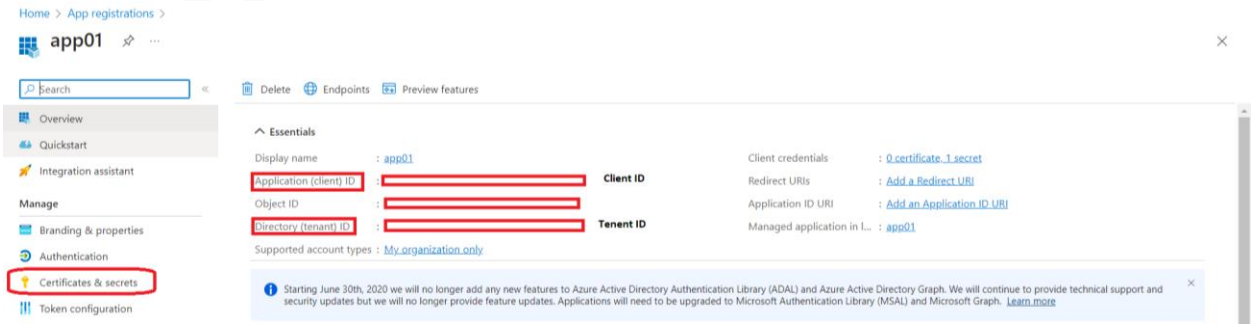


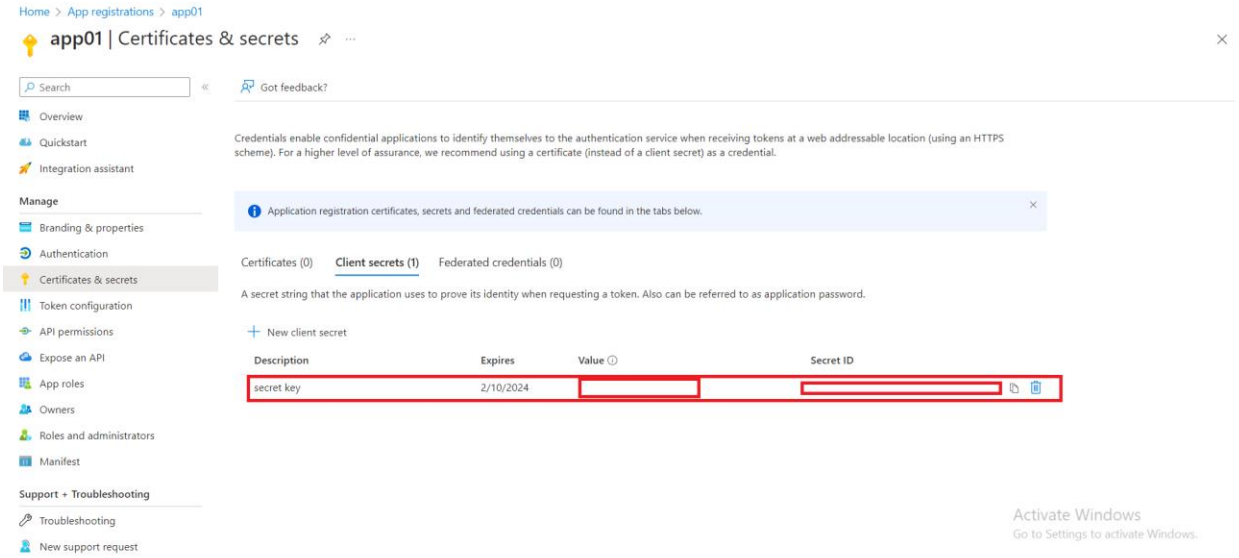




Connect Databricks notebook with Data Factory:

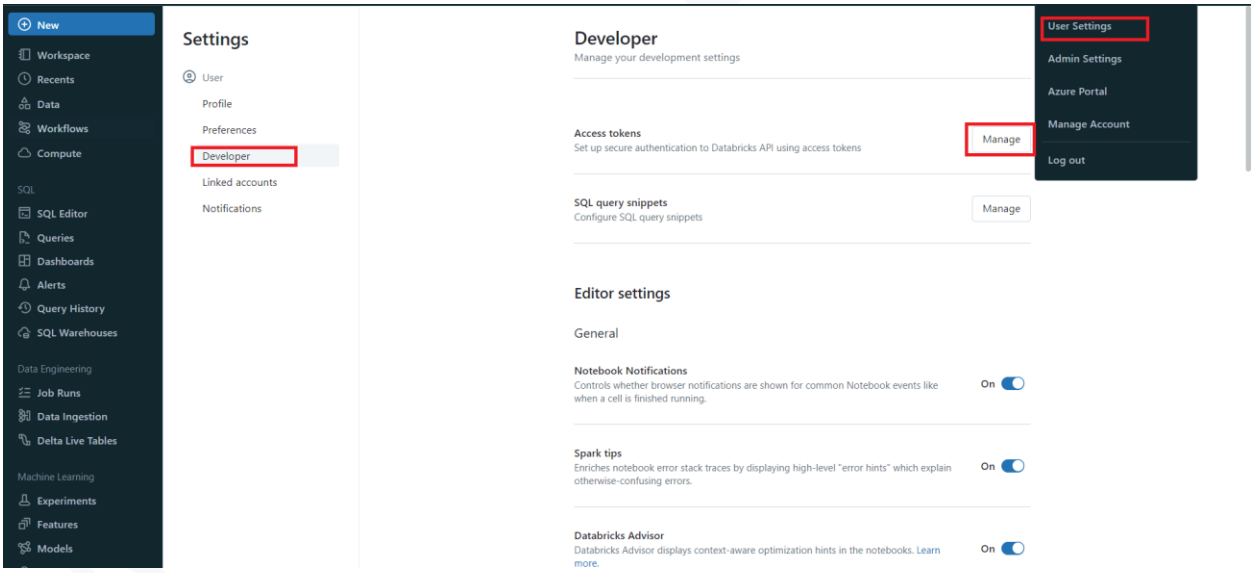
Put the client ID, tenant ID, and secret key into the **configuration.ipynb** file, which is available in the repository.



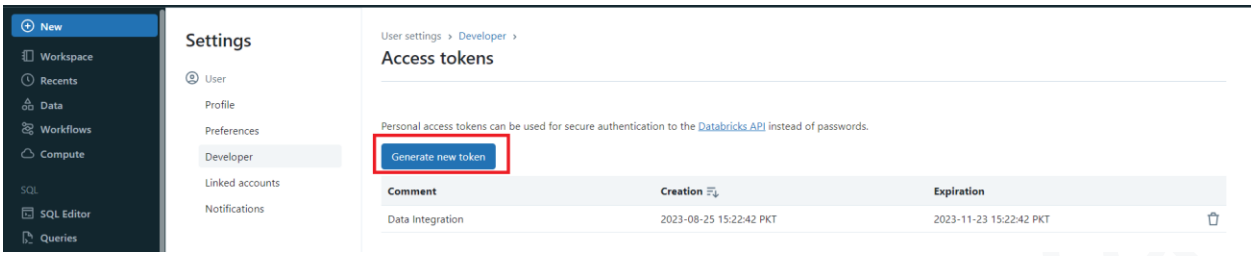


Databricks pipeline configuration:

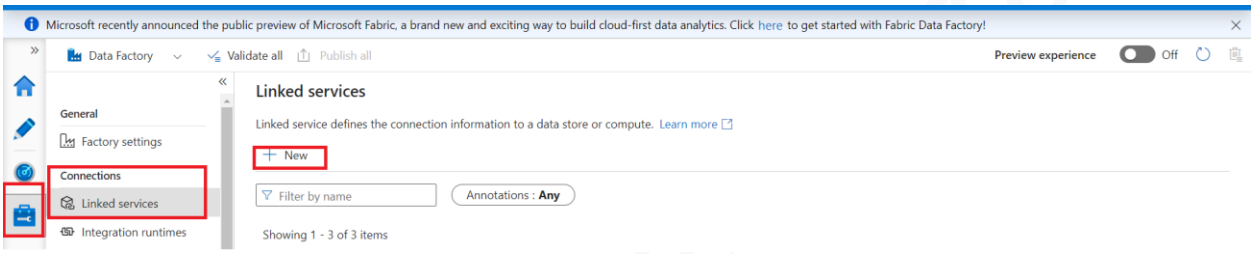
After performing the data transformation, integrate it into the pipeline by navigating to User Settings -> Developer. Within the Developer section, click on **'Manage.'**"



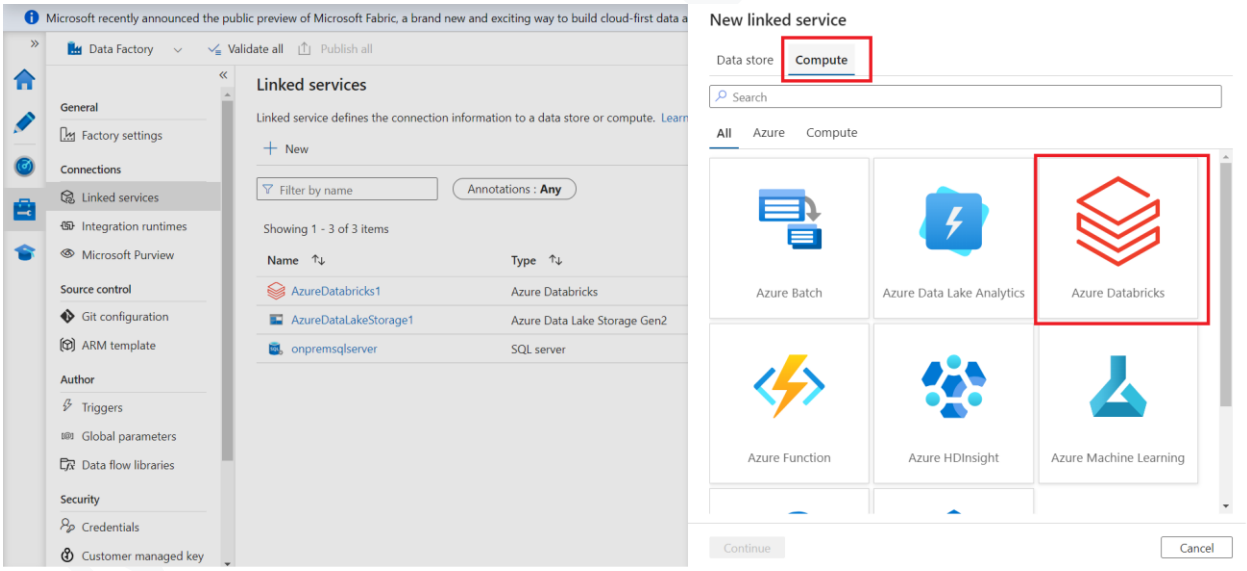
Next, click on **'Generate new Token'.**



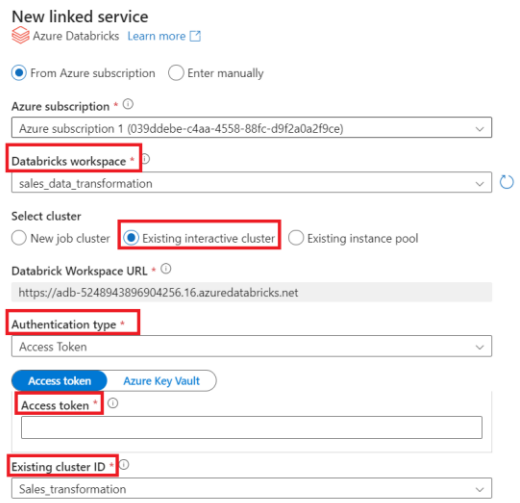
Next, open Azure Data Factory Studio, go to the **'Manage'** tab, navigate to **'Linked Services,'** and then click on **'New.'**



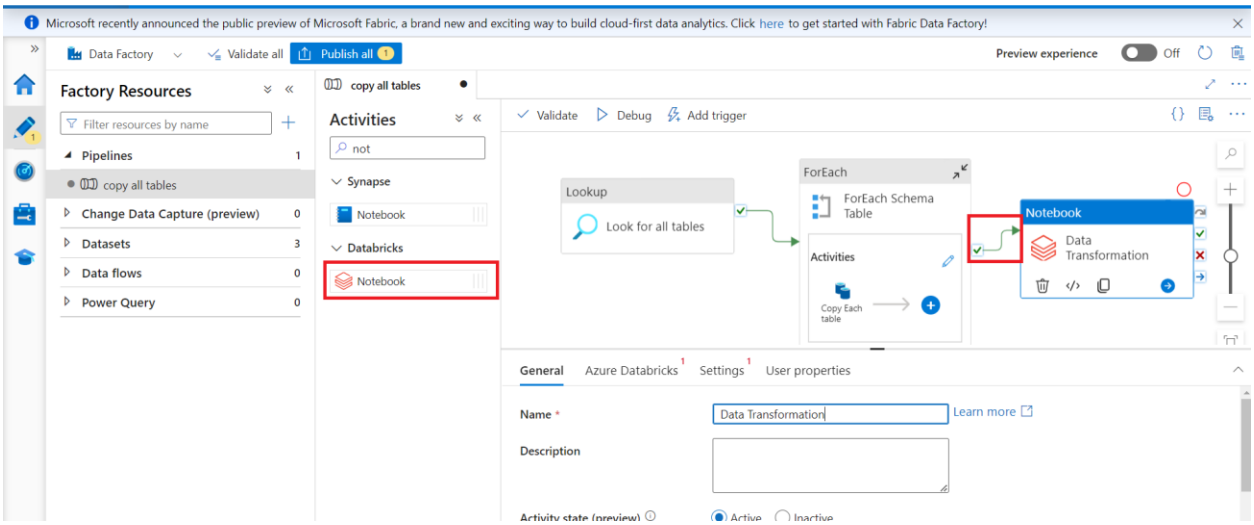
Afterward, click on **"Compute"** and choose **"Azure Databricks"** from the options.



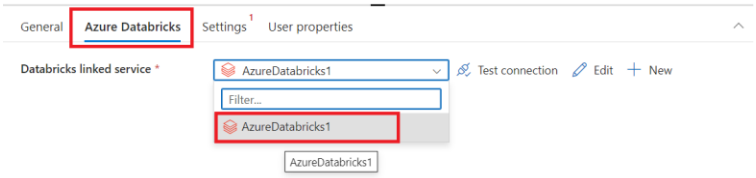
In the linked service setup, choose **"Databricks Workspace,"** then select **"Existing Interactive Cluster."** For the authentication type, opt for **"Access Token."** Proceed by pasting the Access Token and the Existing Cluster ID. Finally, click on the **"Create"** button.



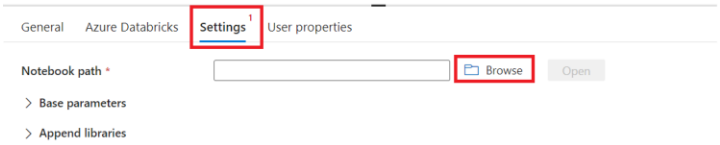
Next, within the Activities section, locate the notebook activity and drag it onto the blank canvas. Connect this activity with the ForEach Activity.



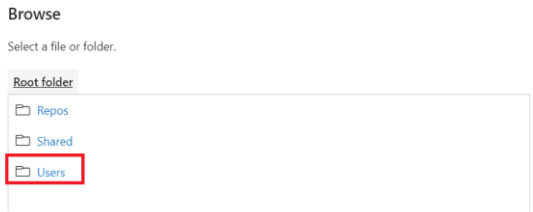
Afterward, click on **"Azure Databricks"** and select the linked service that was created earlier.



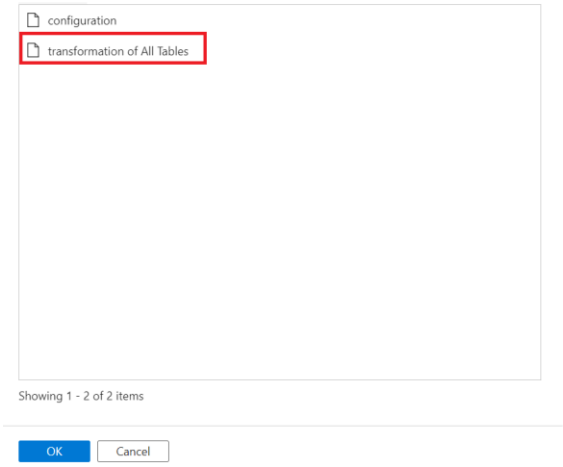
Then click on **"Settings"** and proceed to click on **"Browse."**



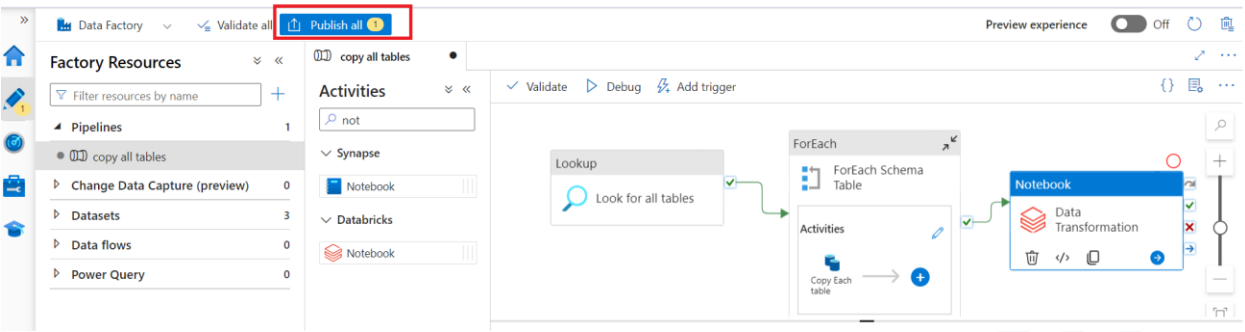
Then click on Users.



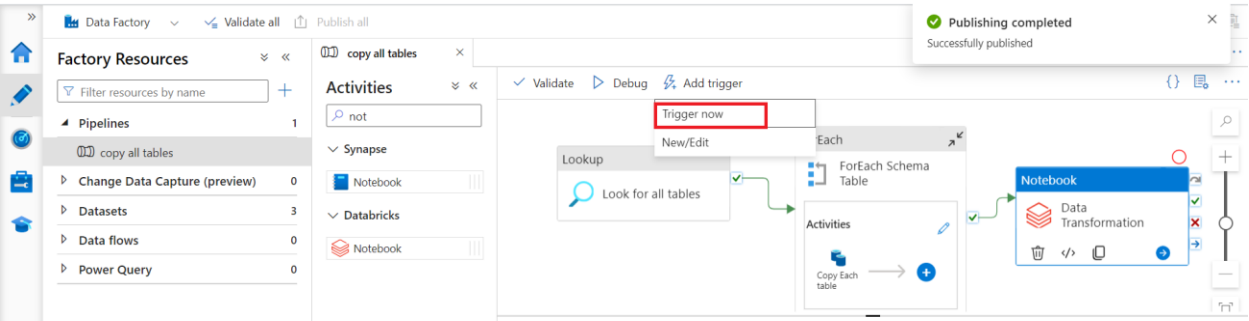
Next, choose your username and select the notebook for transforming all tables. After that, click on **"OK."**



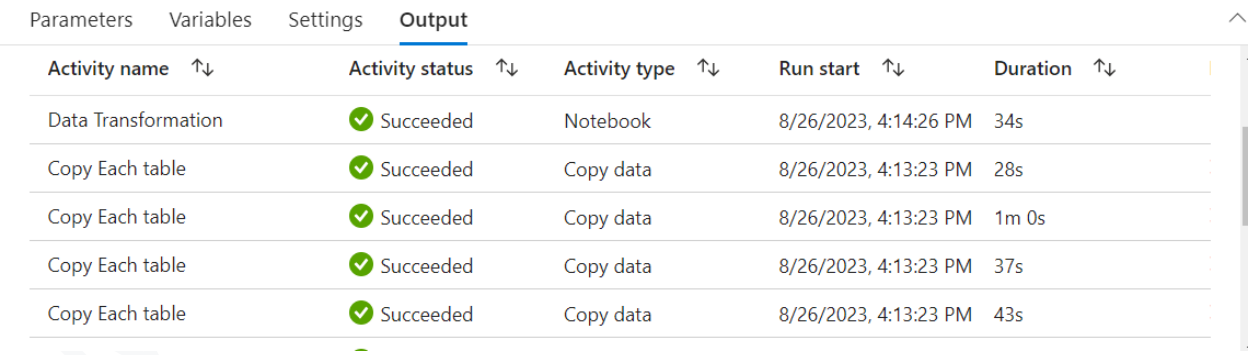
After configuring Databricks in your pipeline, make sure to publish the changes.



Once the changes are published, click on **"Add Trigger"** and then select **"Trigger Now."**



As you can see, we have successfully configured Databricks in our pipeline.



Parameters Variables Settings **Output**

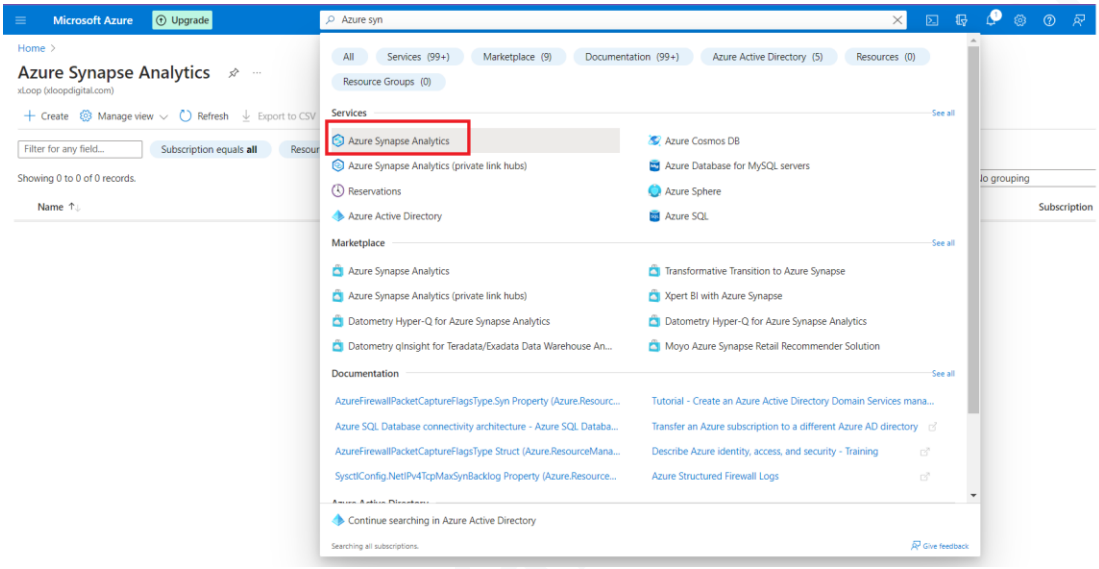
Activity name	Activity status	Activity type	Run start	Duration
Data Transformation	Succeeded	Notebook	8/26/2023, 4:14:26 PM	34s
Copy Each table	Succeeded	Copy data	8/26/2023, 4:13:23 PM	28s
Copy Each table	Succeeded	Copy data	8/26/2023, 4:13:23 PM	1m 0s
Copy Each table	Succeeded	Copy data	8/26/2023, 4:13:23 PM	37s
Copy Each table	Succeeded	Copy data	8/26/2023, 4:13:23 PM	43s

Part 3: Data Loading

Azure Databricks & Azure Synapse Analytics Integration and Loading

To load our transformed data into Azure Synapse Analytics, we need to create a new **"Azure Synapse Analytics"** linked service.

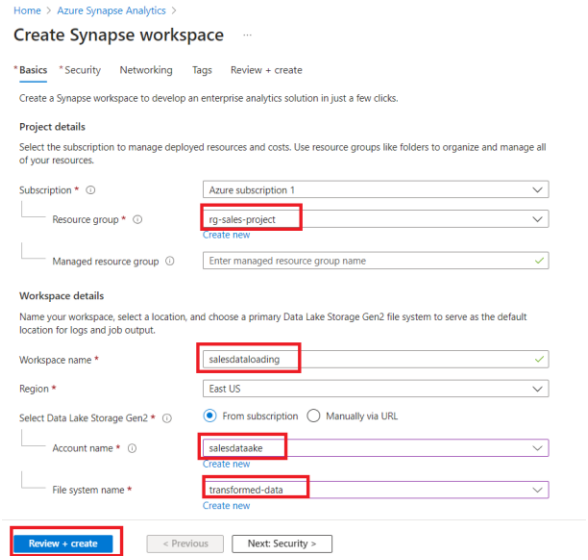
In the Azure Portal, search for **"Azure Synapse Analytics."**



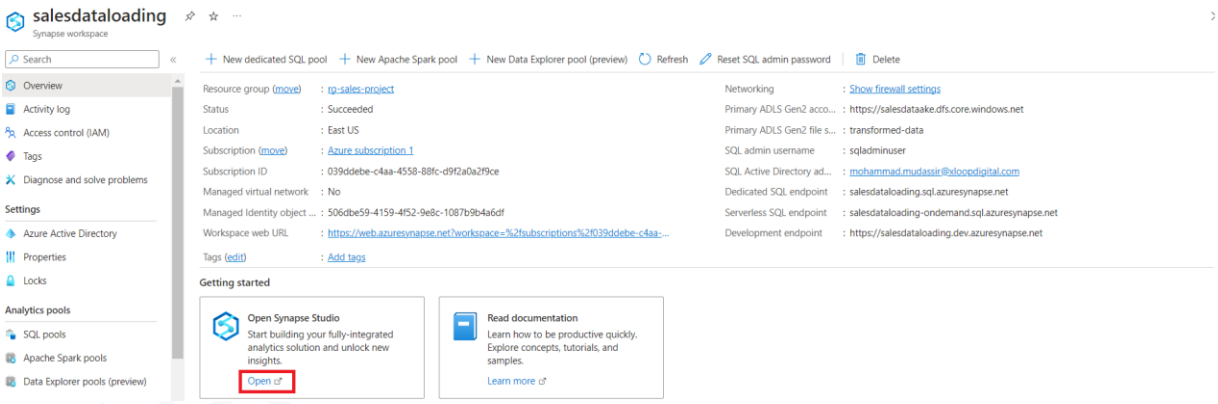
To create a new Synapse Analytics workspace, follow these steps:

1. Navigate to the Azure portal.
2. Click on "Create a resource."
3. Search for **"Azure Synapse Analytics"** and select it.
4. In the **"Azure Synapse Analytics"** configuration panel:
    a. Choose the existing resource group.
    b. Enter a unique workspace name.
    c. For **"Account name,"** select the existing DataLake Gen2 account name.
    d. Specify the directory where you save the transformed data.
5. Click on **"Review + create"** to proceed with the creation.

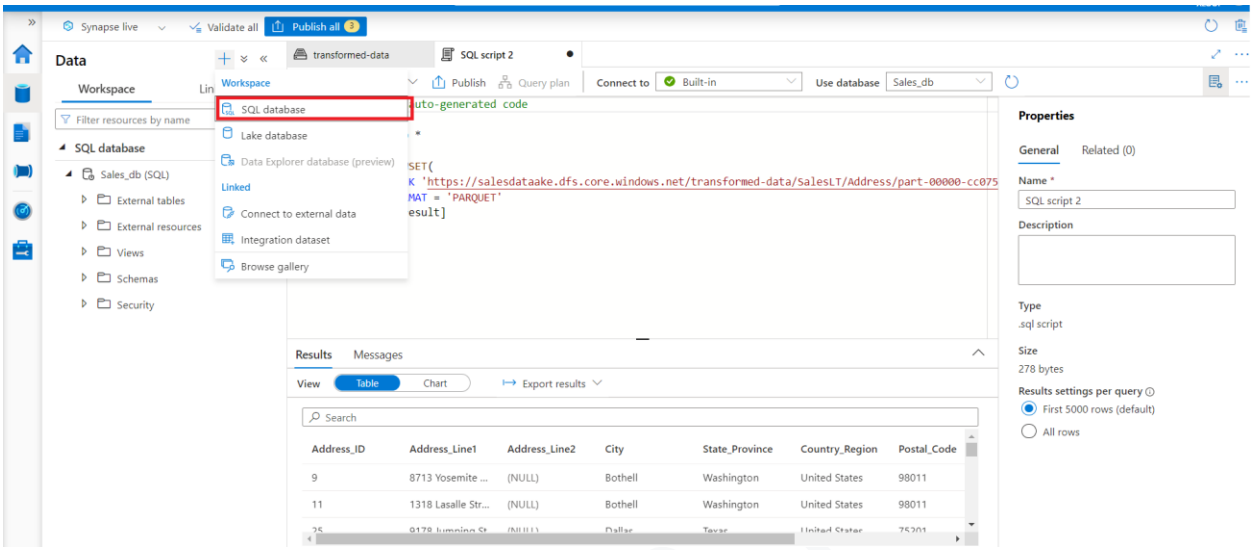
Review the details, and if everything looks correct, click **"Create"** to create your Synapse Analytics workspace.



Next, access your Synapse Analytics resource and open your Azure Synapse Analytics workspace.



In the Azure Synapse Analytics workspace, navigate to the Data section, click on the **"+"** icon, and create a SQL Database. Provide a name for your database during the creation process.



Serverless SQL Pools:

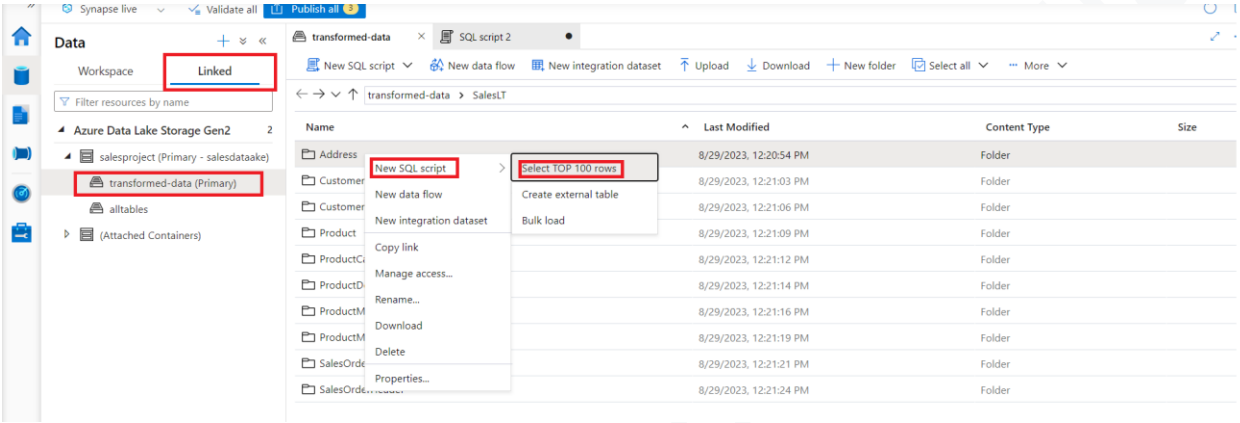
(1) Serverless SQL pools in Azure Synapse Analytics allow you to query data on-demand without provisioning or managing dedicated resources.
(2) This is suitable for scenarios where you have intermittent or unpredictable workloads and want to avoid the overhead of managing dedicated resources.
(3) You pay only for the data processed during your queries, making it cost-effective for sporadic or ad-hoc analytical tasks.
(4) It's well-suited for exploration, discovery, and data analysis tasks.

Dedicated SQL Pools:

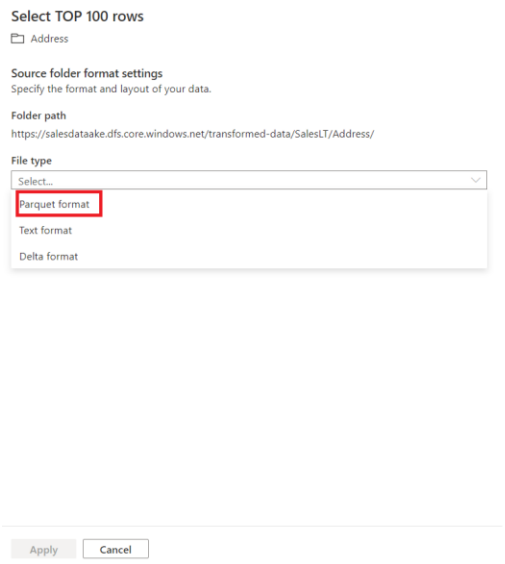
(1) Dedicated SQL pools involve provisioning dedicated resources (compute power and storage) for your analytical workloads.
(2) This option is more suitable for predictable and consistent workloads where you need higher performance and guaranteed resources.
(3) Dedicated pools are better suited for complex analytics, large-scale data transformations, and regular reporting tasks.

(4) You have more control over the resource allocation, making it possible to fine-tune performance for specific queries and workloads.
(5) However, you are responsible for managing and monitoring the dedicated resources, which might involve more administrative overhead.

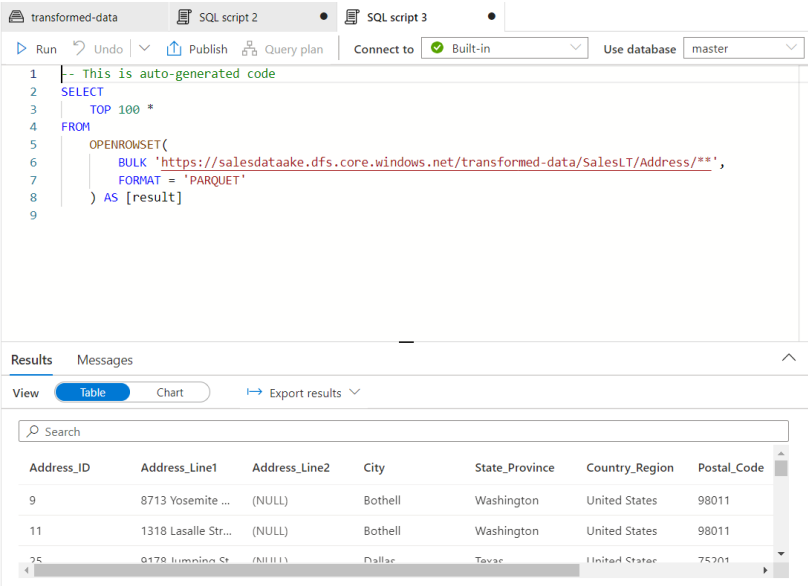
In the **"Linked"** tab, select the transformed data directory and then choose the **"Address table."** Right-click on the **"Address table"** and select **"New SQL Script"** followed by **"Top 100 rows."**



Choose the file format as Delta, and then click on **"Apply"** to confirm your selection.



Once the query has been executed, you will be able to view the output.



Next, go to the **"Develop"** section and click on the **"+"** icon to create a new SQL script. Then, paste the script provided below.

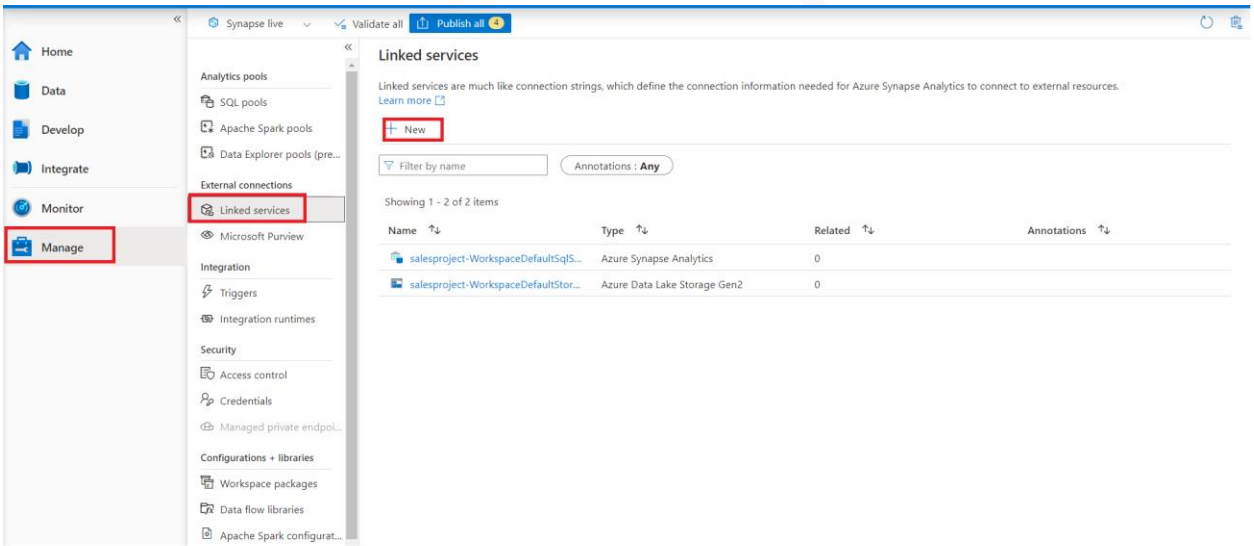
Script:

```
-- Use the Sales_db database for this script
USE Sales_db;
GO

-- Create or alter a stored procedure named SQL_script_solo
CREATE OR ALTER PROC SQL_script_solo @ViewName nvarchar(100)
AS
BEGIN

-- Declare a variable to store the SQL statement
DECLARE @statement VARCHAR(MAX);

-- Build the SQL statement to create or alter a view
SET @statement = N'CREATE OR ALTER VIEW ' + QUOTENAME(@ViewName)
+ ' AS
    SELECT
         *
    FROM
        OPENROWSET(
```

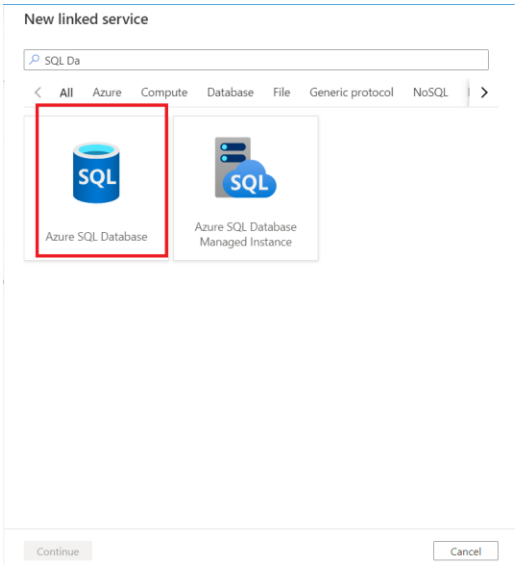
Mehmaam & Huzaifa

```
            BULK
''https://salesdatalake.dfs.core.windows.net/transformed-
data/SalesLT/'+ @ViewName+ '/'',
            FORMAT = ''DELTA''
        ) AS [result]
    '

-- Execute the dynamically generated SQL statement
EXEC (@statement);

END
GO

-- End of the script
```

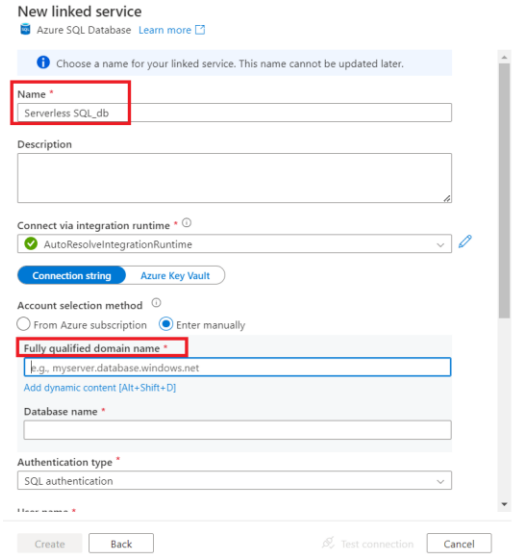
In the **"Manage"** tab, select the **"Linked service"** option, and then click on **"New"** to create a new Linked service.



For Serverless SQL Database, select it, and then click on **"Continue."**

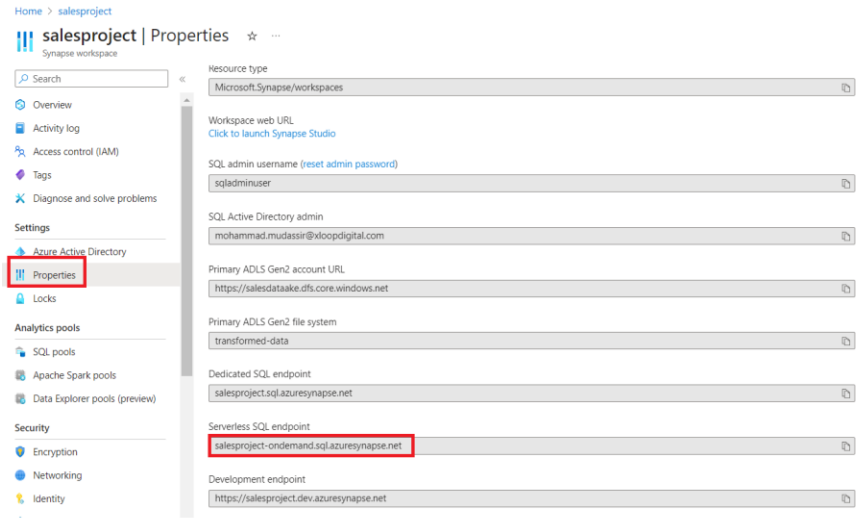


Provide a name for the configuration and enter the Fully Qualified Domain Name (FQDN) you obtained from the Synapse account properties.

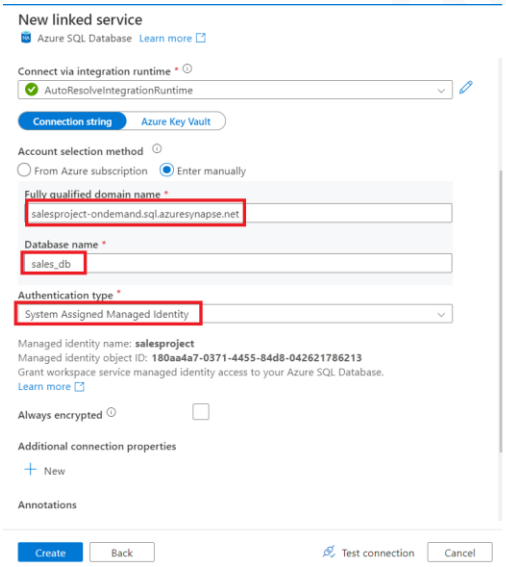


To obtain the Fully Qualified Domain Name (FQDN) in the Synapse account, follow these steps:

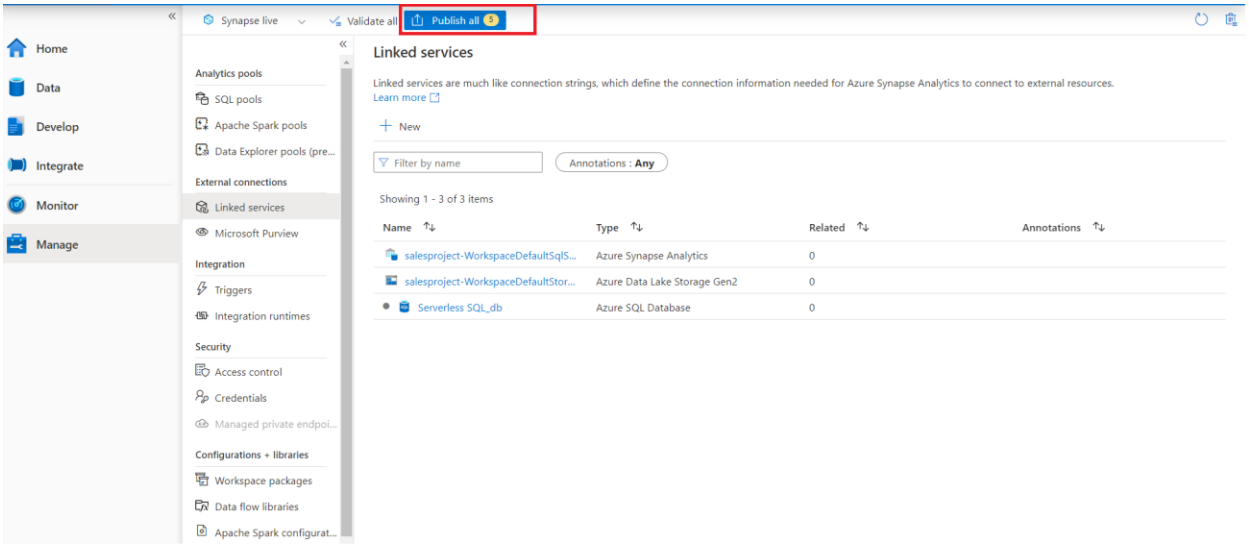
(1) Go to your Synapse account.
(2) Navigate to the "Properties" section.
(3) Copy the Serverless SQL endpoint from there.



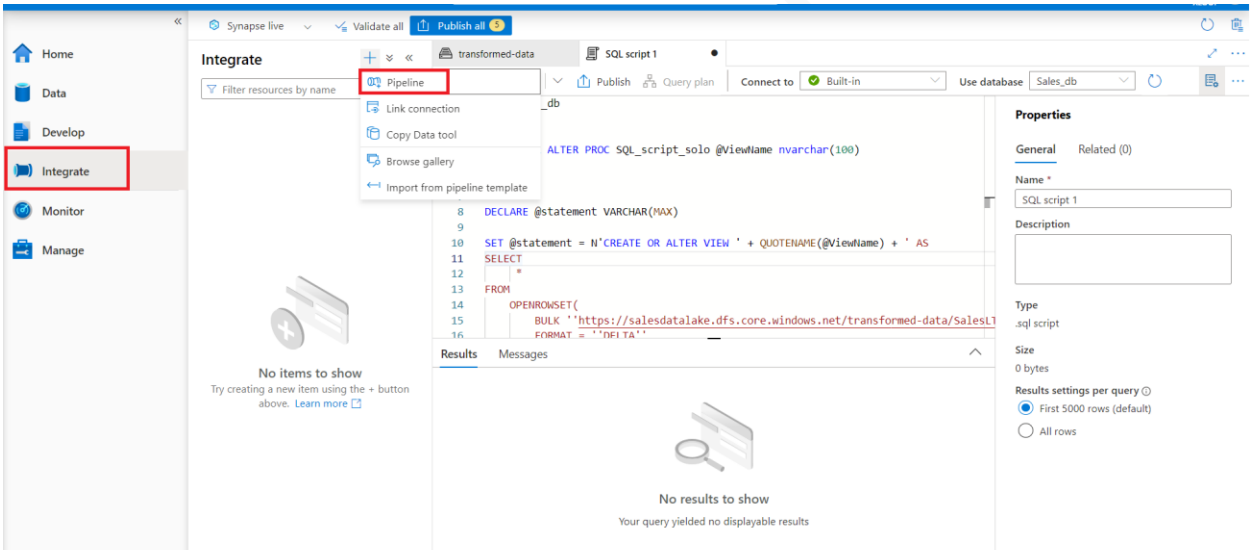
Paste the Serverless SQL endpoint in the linked service, specify the database name, and select the authentication type as **"System Assigned Managed Identity."** Finally, click on **"Create."**



After making your changes, remember to publish all the modifications.

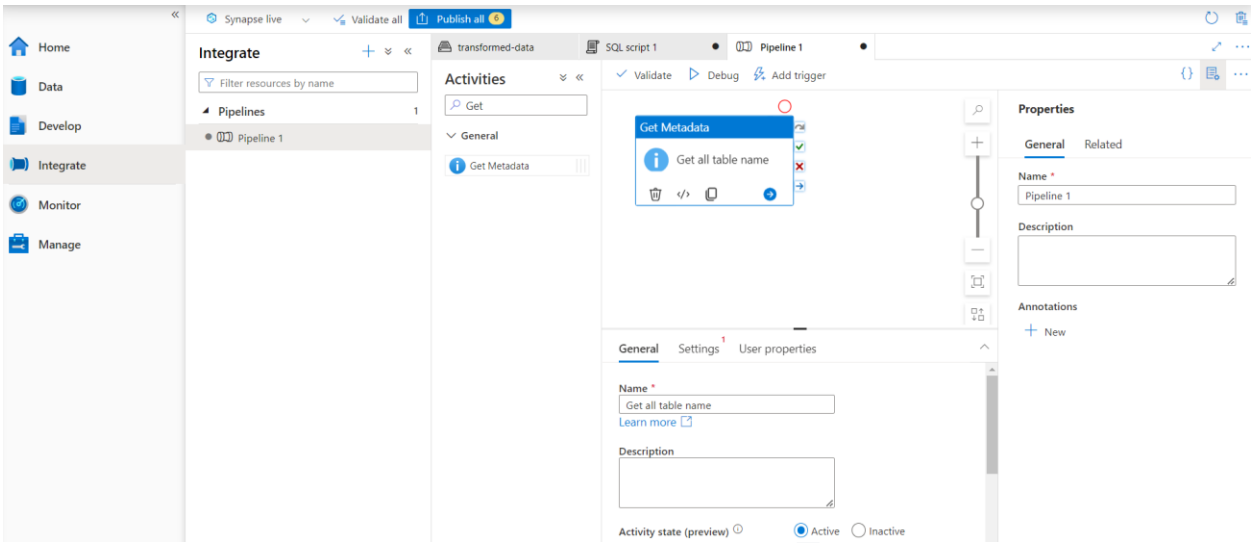


Now, go to the **"Integrate"** tab and click on the **"+"** icon to create a new pipeline.

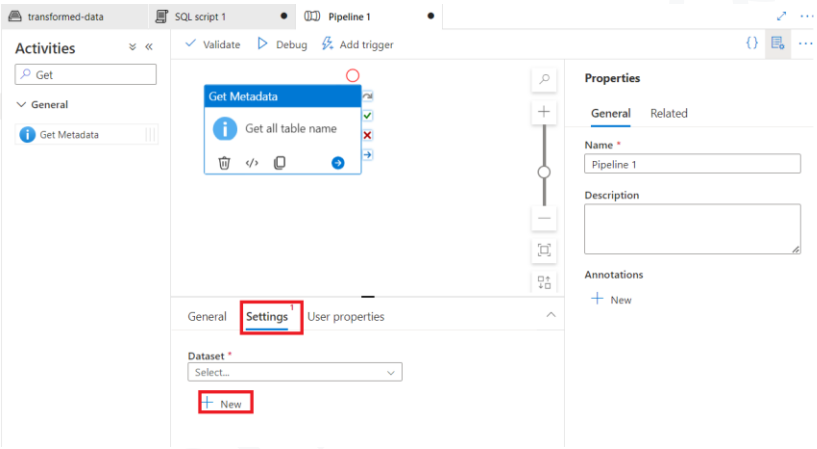


In the pipeline, search for the **"Get metadata**" activity and drag it onto the blank canvas. This activity will retrieve all the names of tables from your Data Lake Gen2 in the transformed data container.

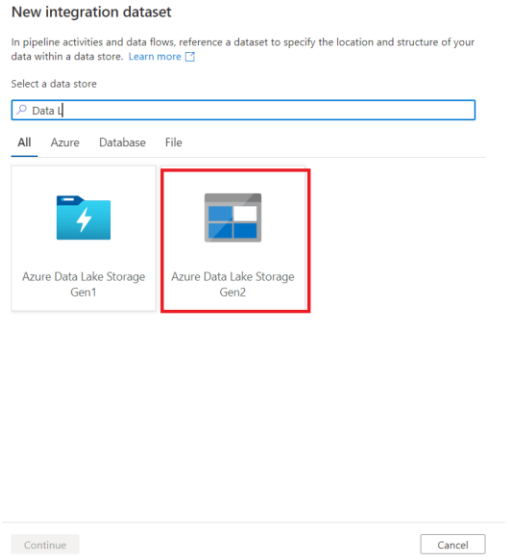
Documentation:
https://learn.microsoft.com/en-us/azure/data-factory/control-flow-get-metadata-activity



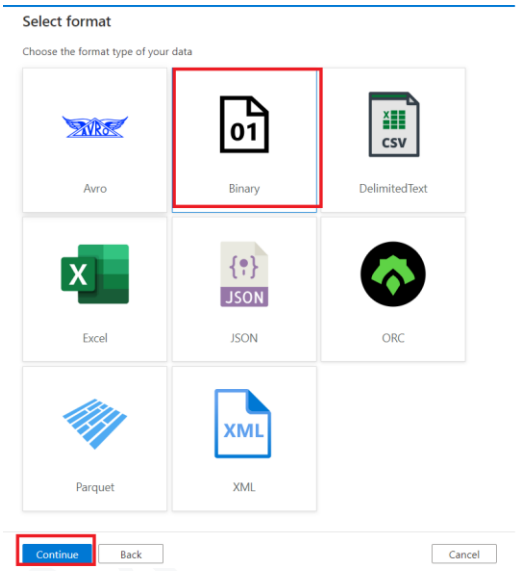
Now, navigate to the "**Settings**" tab and click on **"New**."



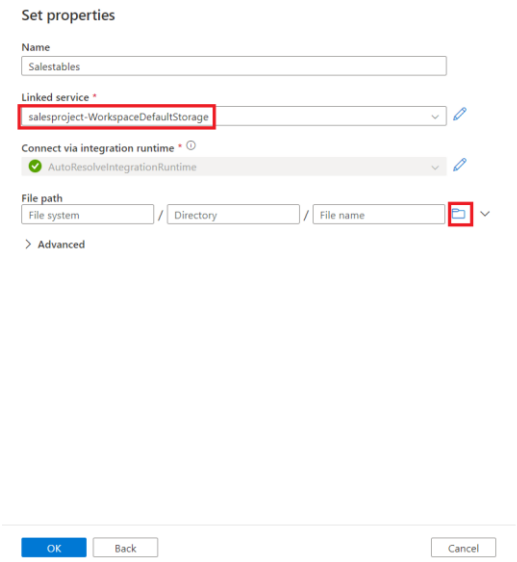
Next, search for "**Data Lake Gen2**."



So far, you have obtained only the table name. Now, select the Binary format and click on "**Continue**."

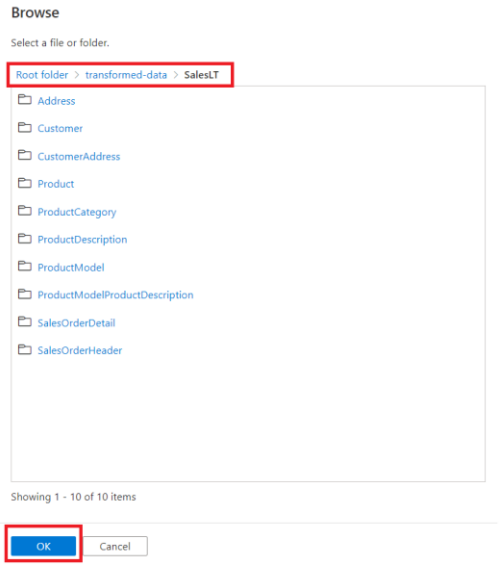


Choose the default linked service, and then click on the browse icon.



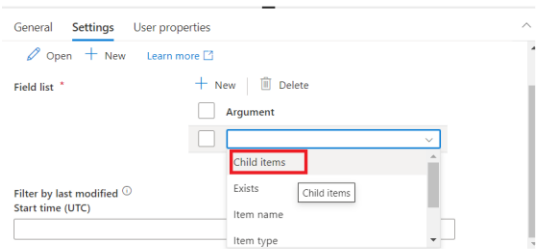
Select the folder where your transformed data is saved.

By clicking the "**OK**" button, all the tables will be displayed

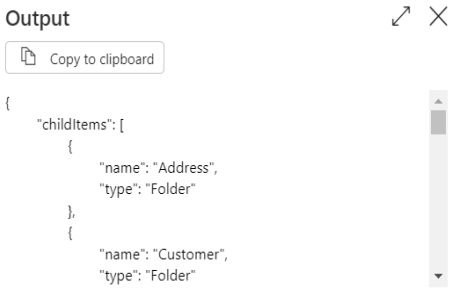


Then click in the Field list and click on new.

In the dropdown menu, select "**Child items**" to retrieve all child items from the SalesLT folder in the transformed-data container.

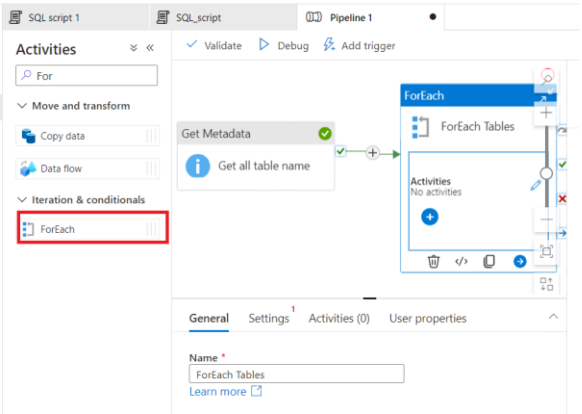


After completing the steps and debugging the activity, you should see that you're able to retrieve all the child items from the SalesLT folder in the transformed-data container in the output.

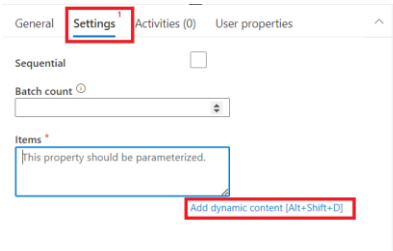


Now, search for the "**ForEach**" activity, drag it onto the blank canvas within the existing "**ForEach**" activity, and link it with the "**Get metadata**" activity upon success.

Documentation:
https://learn.microsoft.com/en-us/azure/data-factory/control-flow-for-each-activity



In the "**ForEach**" activity, navigate to the "**Settings**" section, and in the **"Items**" section, click on "**Add Dynamic content**."

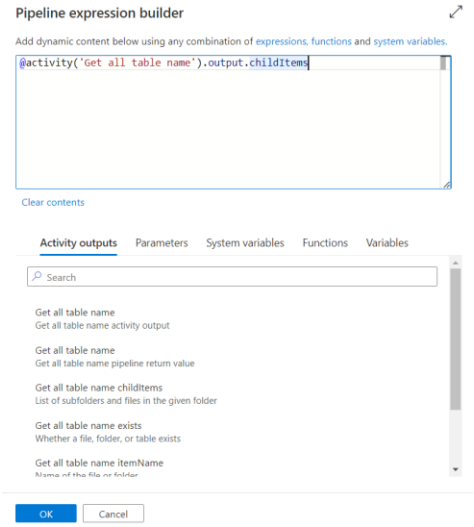


Mehrdad & Huzaifa

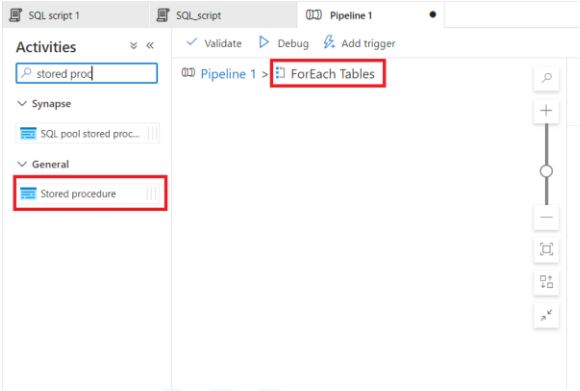
In the pipeline Expression Builder, input the following script:

Script:

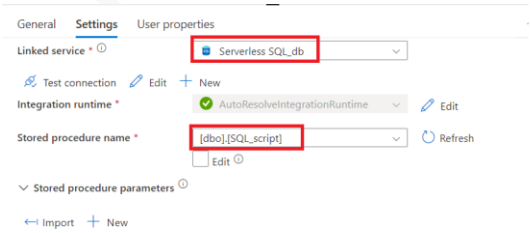
```
@activity('Get all table name').output.childItems
```



Next, in the "**ForEach**" activity, search for the "**Stored procedure**" and drag it onto a blank canvas.



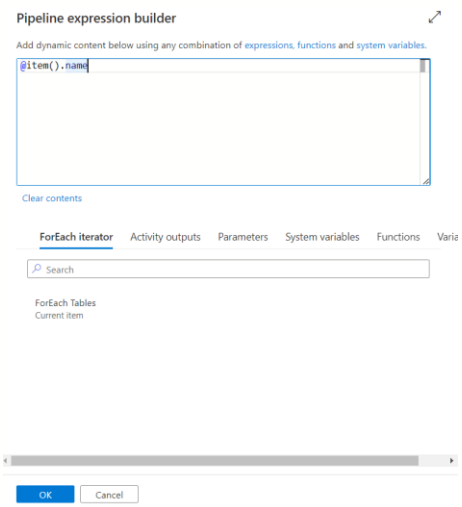
Select the Linked service "**Serverless SQL_db**" and choose the script in which you created the stored procedure.



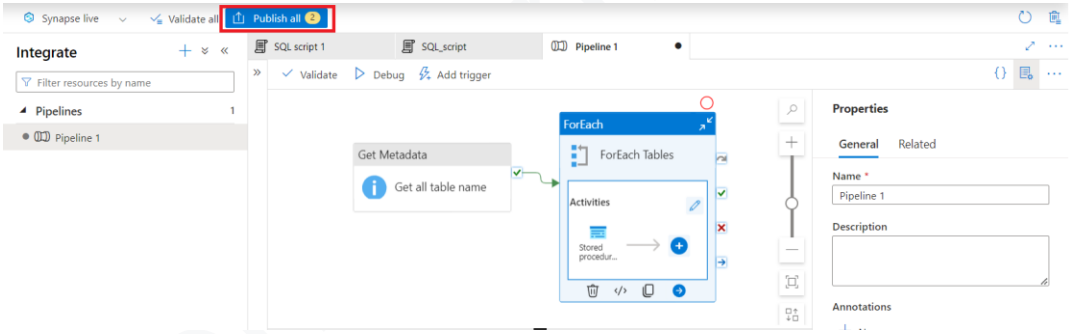
In the "**Stored procedure parameters,**" provide the name as **"ViewName"** with the type set to "**String**." To set the value, click on "**Add Dynamic content**," then in the pipeline expression builder, input the following script and click "**OK**."

Script:

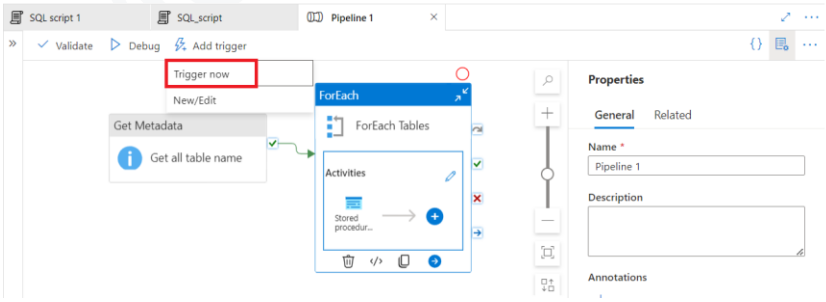
```
@item().name
```



Next, navigate to the outer pipeline and publish all the changes.



Then Trigger the Pipeline as Trigger Now.



As you can see, our pipeline runs successfully.

Activity runs

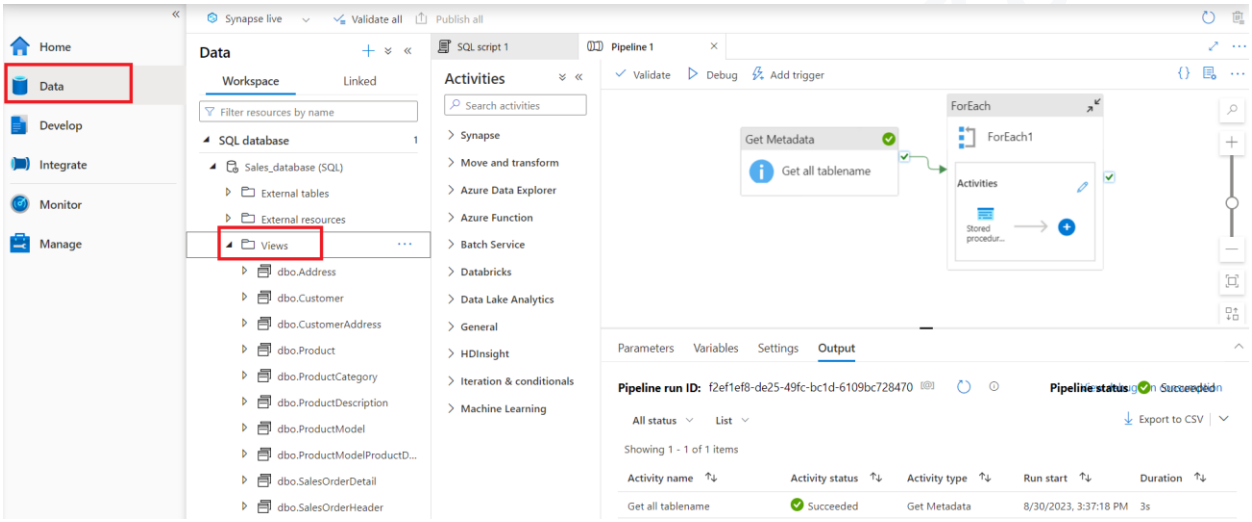
Pipeline run ID 0c5ac0dd-6ccb-4f37-850a-016ca56dcf89

All status List Export to CSV

Showing 1 - 12 items

Activity name	Activity status	Activity type	Run start	Duration	Log	Integration runtir
Stored procedure1	Succeeded	Stored procedure	8/30/2023, 3:48:22 PM	26s		AutoResolveInteg
Stored procedure1	Succeeded	Stored procedure	8/30/2023, 3:48:22 PM	28s		AutoResolveInteg
Stored procedure1	Succeeded	Stored procedure	8/30/2023, 3:48:22 PM	26s		AutoResolveInteg
Stored procedure1	Succeeded	Stored procedure	8/30/2023, 3:48:22 PM	26s		AutoResolveInteg

In the **"Develop"** section, under the **"Views"** tab, you can see all the tables. Finally, we load all these tables into Azure Synapse Warehouse.



So Congratulations our loading part is Completed. 👏

Contacts

Syed Muhammad Mehmaam

I am a Data Engineer with a couple of years of experience. I have worked on various projects related to Cloud Data Engineering and Data Scientist.

If you'd like to get in touch:

Linkedin : https://www.linkedin.com/in/muhammad-mehmaam/
Github : https://github.com/Mehmaam99
Mail : muhammadmehmaam@gmail.com

Muhammad Huzaifa Imtiaz

I'm an Azure Cloud Data Engineer with a few years of hands-on experience. I've had my fair share of projects in the exciting world of cloud data engineering.

If you ever want to chat or connect, don't hesitate to reach out!

Linkedin : https://www.linkedin.com/in/huzaifai/
Github : https://github.com/Huzaifa116
Mail : datas.huzaifa@gmail.com

Mehmaam & Huzaifa

Thank You

We are writing this thank-you note to acknowledge your outstanding achievement in completing the project. We must express our sincere appreciation for the effort you put into the project, your dedication and hard work are truly commendable. We look forward to seeing more exceptional performances from you in the future.

Once again, congratulations on your remarkable effort, and we hope to establish a long-term association with you.

Mehmaam & Huzaifa