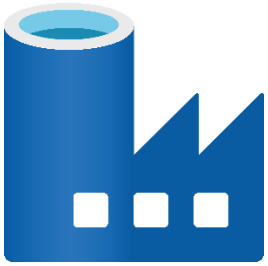


AZURE DATA ENGINEERING REAL-TIME END-TO-END PROJECT

NGUYEN CONG LUC
nguyencongluc.82@gmail.com
Luc Nguyen | LinkedIn
0329206845

INDEX

Introduction

As a passionate and dedicated individual eager to start my journey in Data Engineering, I am excited to present a detailed Azure Data Engineering project that I have worked on to build foundational skills in this field.

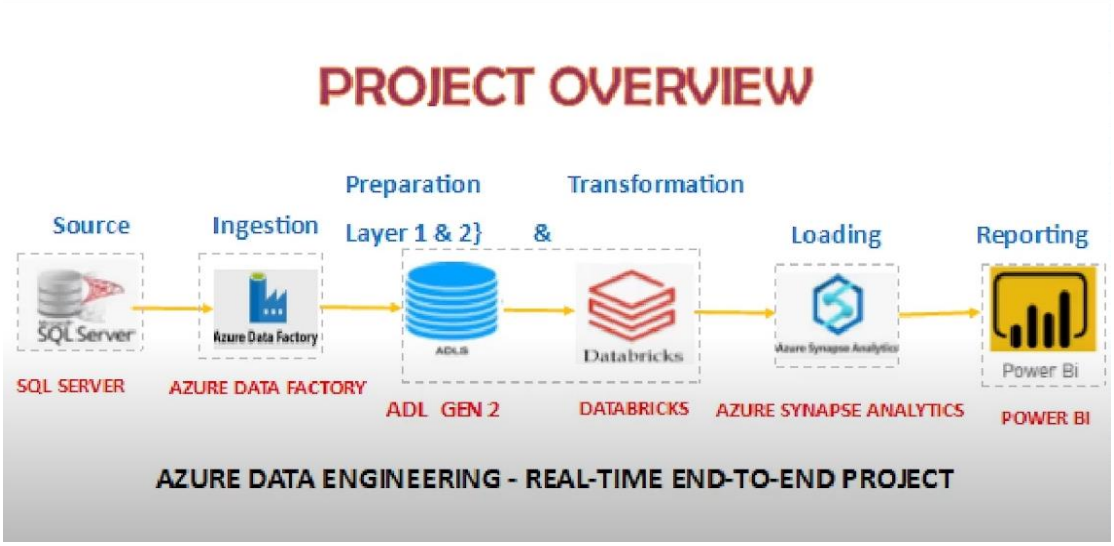
This project showcases my ability to learn and apply various Azure services, including SQL Server, Azure Data Factory (ADF), Azure Data Lake Storage (ADLS Gen 2), Databricks, Azure Synapse Analytics, and Power BI. Through this hands-on experience, I gained insights into data pipeline orchestration, data transformation, storage, and visualization processes.

While I may be at the beginning of my professional journey, I am confident that my enthusiasm for learning and my commitment to building scalable data solutions will make me a valuable addition to any team.

CHAPTER 1. Project Overview

1.1 Overview

A comprehensive real-time, end-to-end Azure Data Engineering project. It showcases how to build a complete data pipeline using various Azure services.



Data used is sales data of Adventure Works Cycles

Adventure Works Cycles is a fictitious company that manufactures and sells bicycles, bicycle accessories, and related products, used as the setting in AdventureWorks, a sample database developed by Microsoft.

1.2 The technologies involved

Here's a breakdown of the technologies involved:

1. SQL Server: Acts as the source database for the initial data.

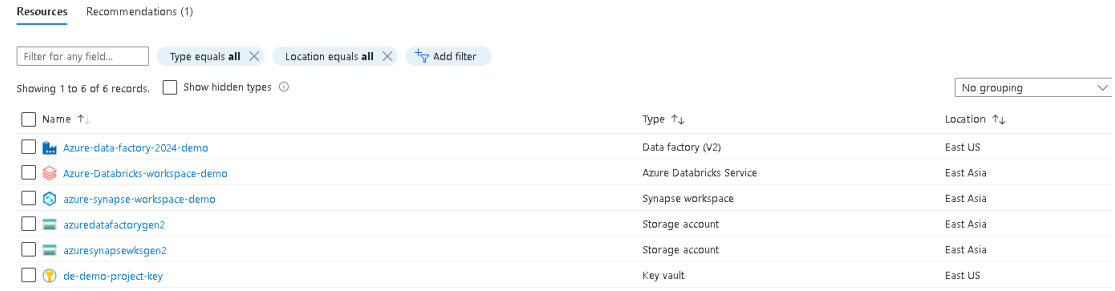
2. Azure Data Factory (ADF): Used for orchestrating data movement and transformations.

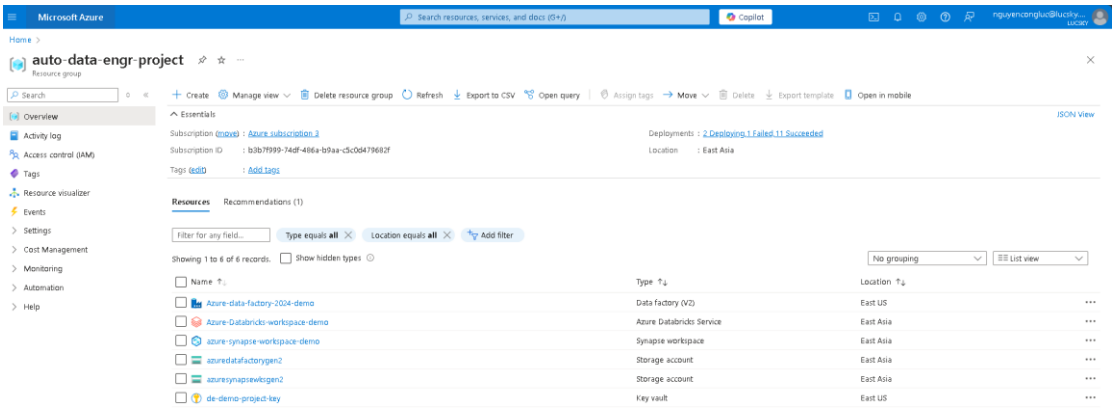
3. ADLS Gen 2 (Azure Data Lake Storage): Stores data in two layers. Layer 1 for raw data from SQL Server via Azure Data Factory and Layer 2 for processed data from Databricks.

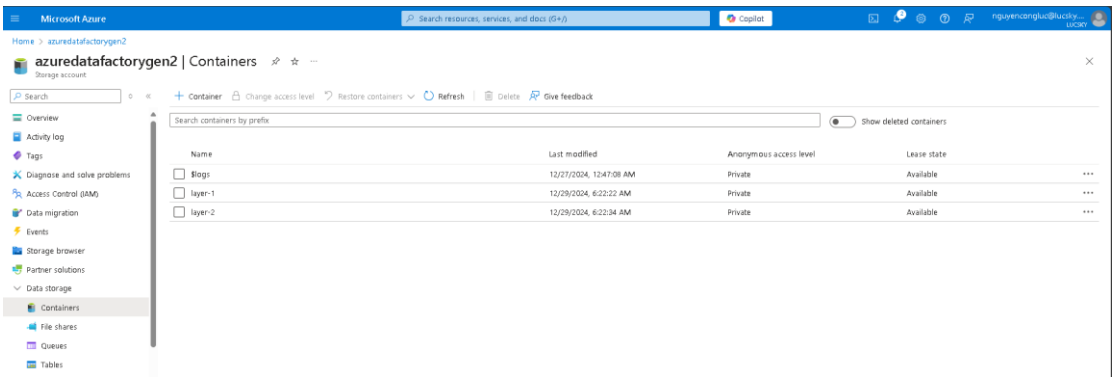
4. Databricks: Implements advanced data processing and transformation, using Spark for big data workloads.

5. Azure Synapse Analytics: Powers data analysis and serves as the data warehouse for reporting and insights.

6. Power Bl: Provides the visualization and reporting layer to deliver business insights







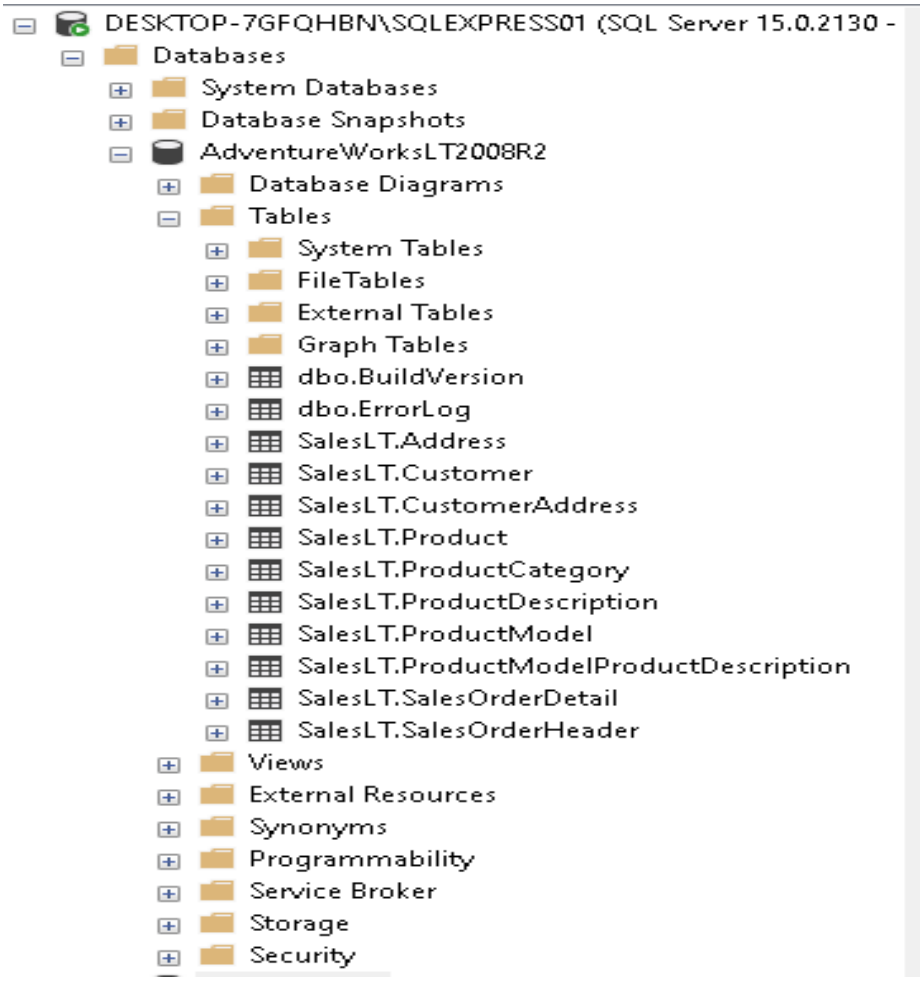
CHAPTER 2. On-Prem & Cloud Setup

2.1 Setup sql server

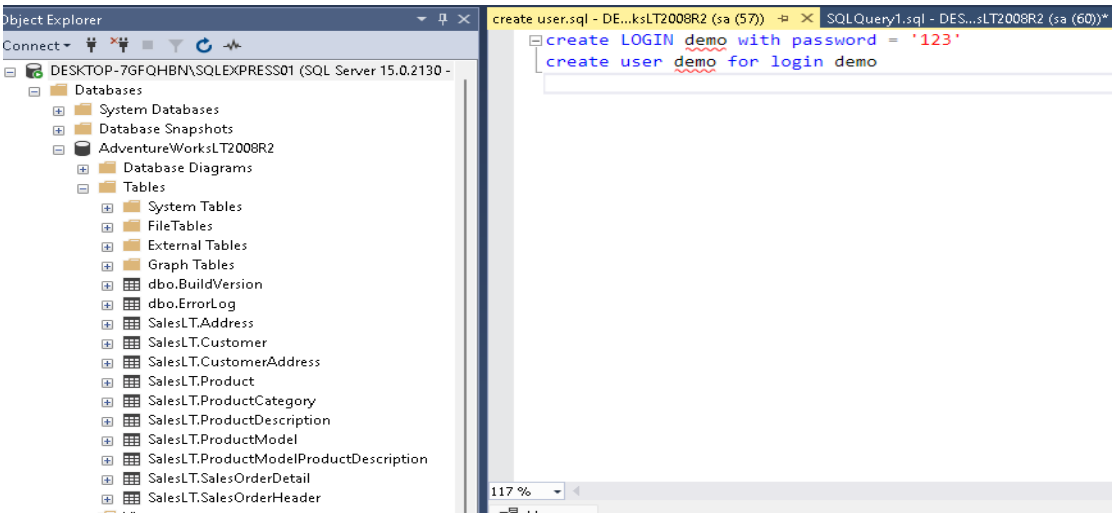
In this project, SQL Server acts as the source database, storing the initial raw data that is extracted, processed, and transformed through the Azure data pipeline.

Create Database SQL Server from A full database backup of AdventureWorksLT2008R2.

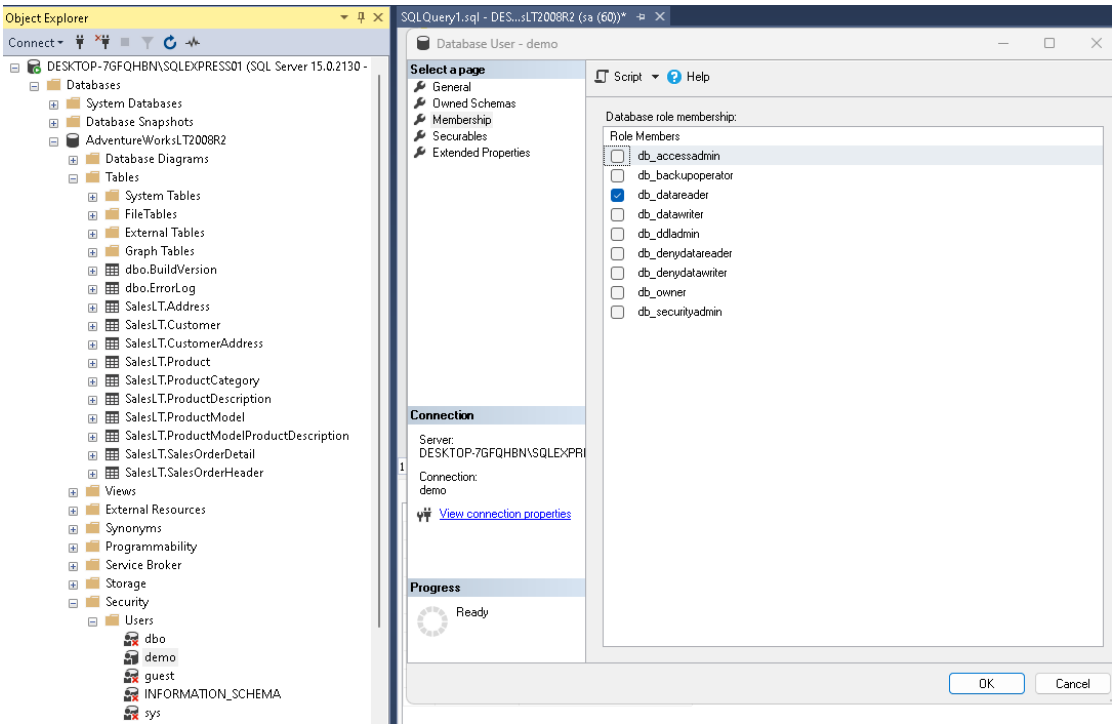
Resource: https://github.com/Microsoft/sql-server-samples/releases/download/adventureworks2008r2/adventure-works-2008r2-lt.bak



Create user for database:



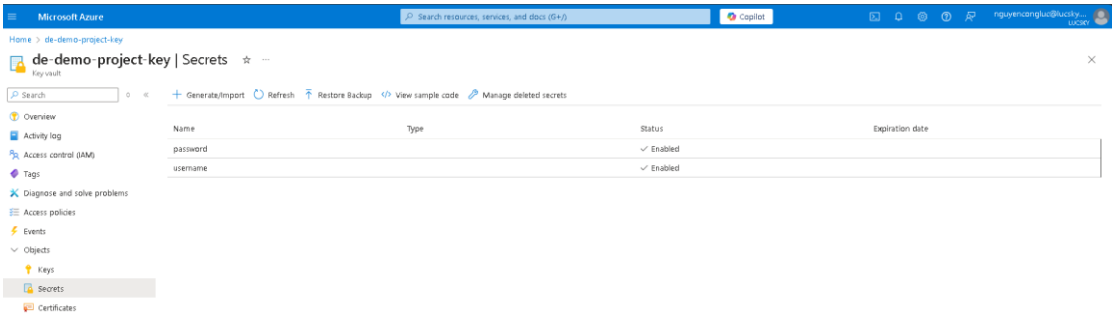
Set read permissons for user



2.2 Setup Key vault

In this project, Azure Key Vault securely stores the SQL Server credentials (username and password) and other sensitive data. It enables authorized services like Azure Data Factory and Databricks to access these secrets securely using Managed Identity, ensuring encryption, controlled access, and seamless integration.

Create secrets with pass and username of account “demo”

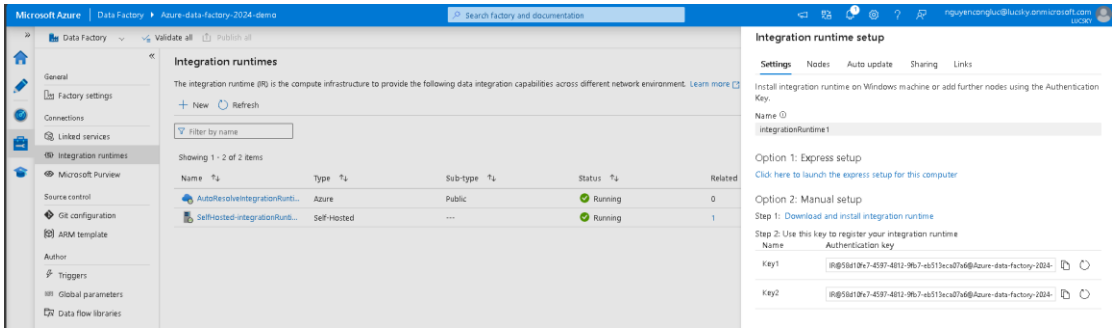


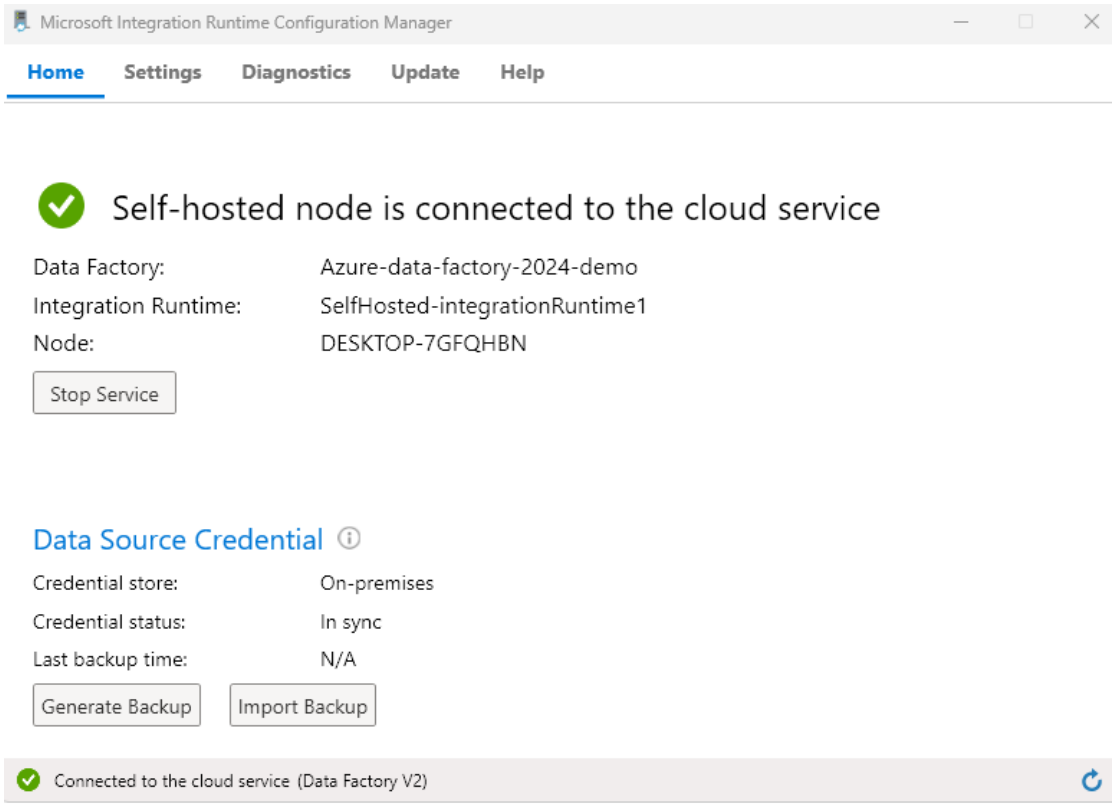
CHAPTER 3. Data Ingestion (E)

In this project, Azure Data Factory (ADF) orchestrates the data pipeline by extracting, transforming, and loading data between services like SQL Server, Azure Data Lake, and Databricks

3.1 Set up Microsoft Integration Runtime

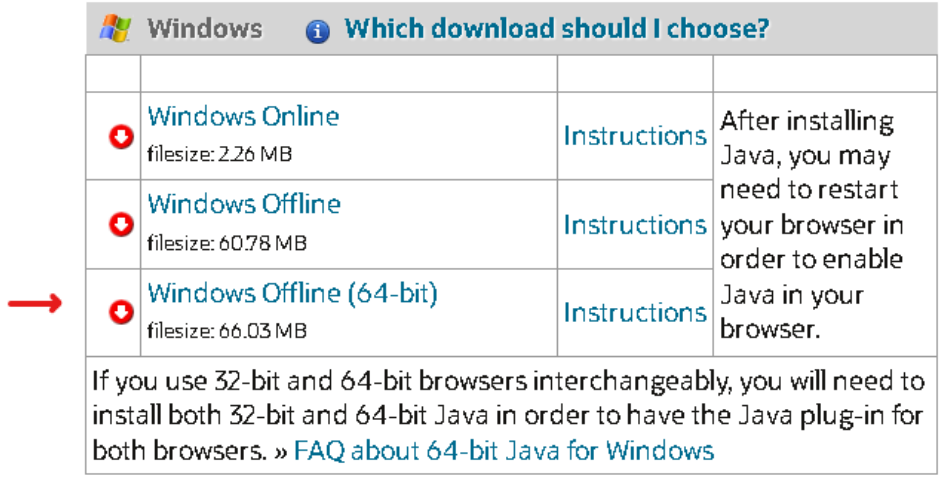
In this project, Microsoft Integration Runtime enables secure and efficient data movement between on-premises SQL Server and Azure services like Data Factory.





Setup JRE to running

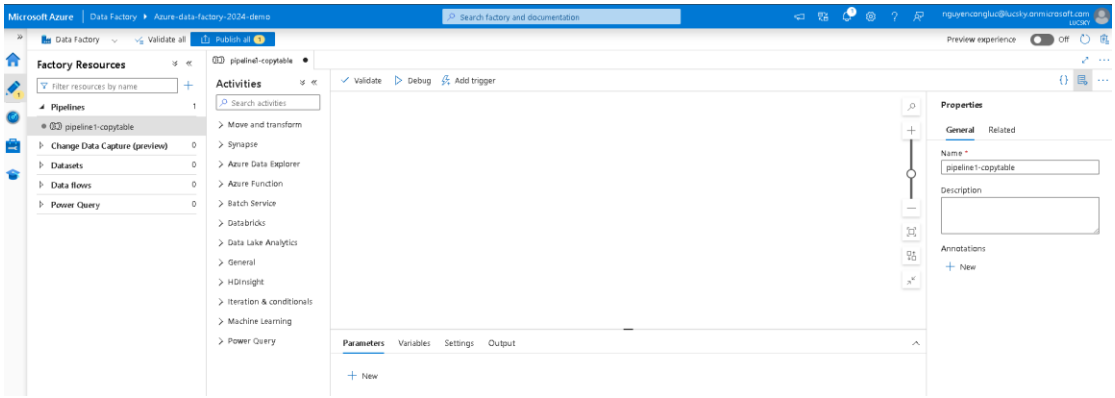
[Java Downloads for All Operating Systems]



Windows	Which download should I choose?		
	Windows Online filesize: 2.26 MB	Instructions	After installing Java, you may need to restart your browser in order to enable Java in your browser.
	Windows Offline filesize: 60.78 MB	Instructions	
→	Windows Offline (64-bit) filesize: 66.03 MB	Instructions	
If you use 32-bit and 64-bit browsers interchangeably, you will need to install both 32-bit and 64-bit Java in order to have the Java plug-in for both browsers. » FAQ about 64-bit Java for Windows			

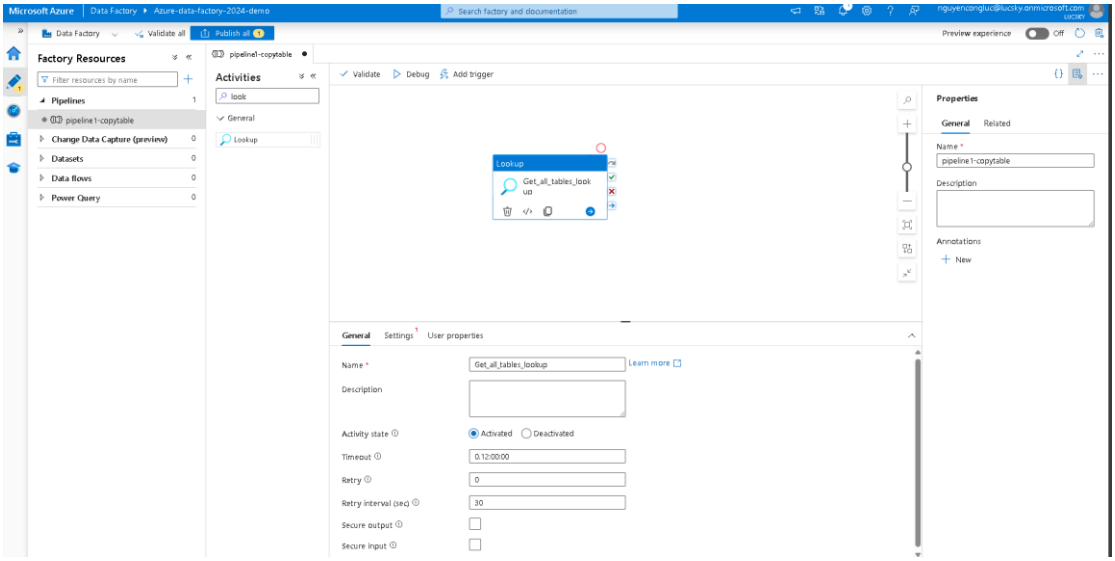
3.2 Create Azure Data Factory pipeline

In this project, the Azure Data Factory pipeline automates the movement and transformation of data from SQL Server to Azure Data Lake, Databricks, and other services, ensuring seamless integration and processing within the data pipeline.

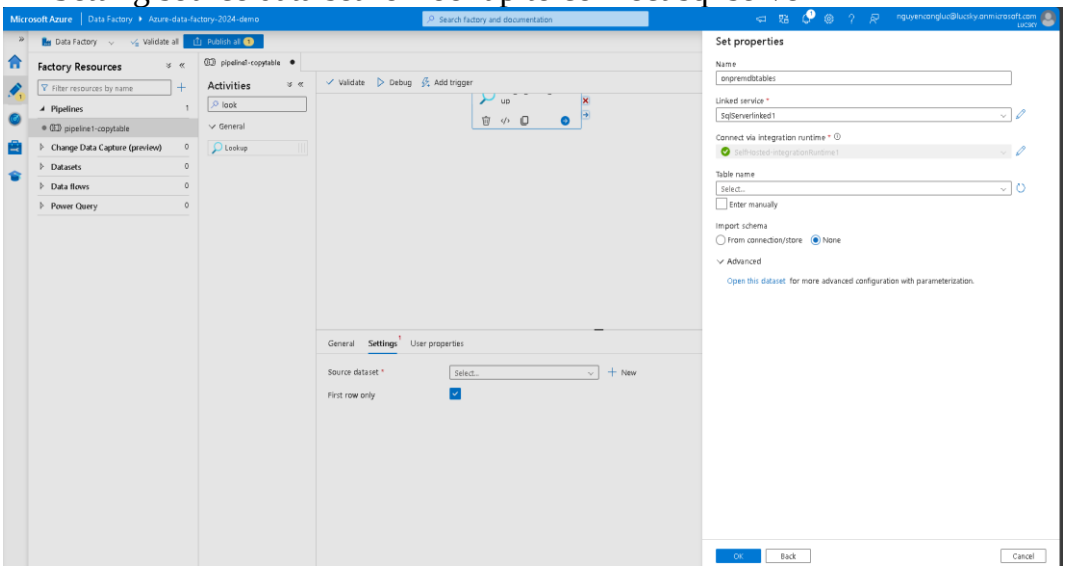


3.2.1 Create lookup in pipeline

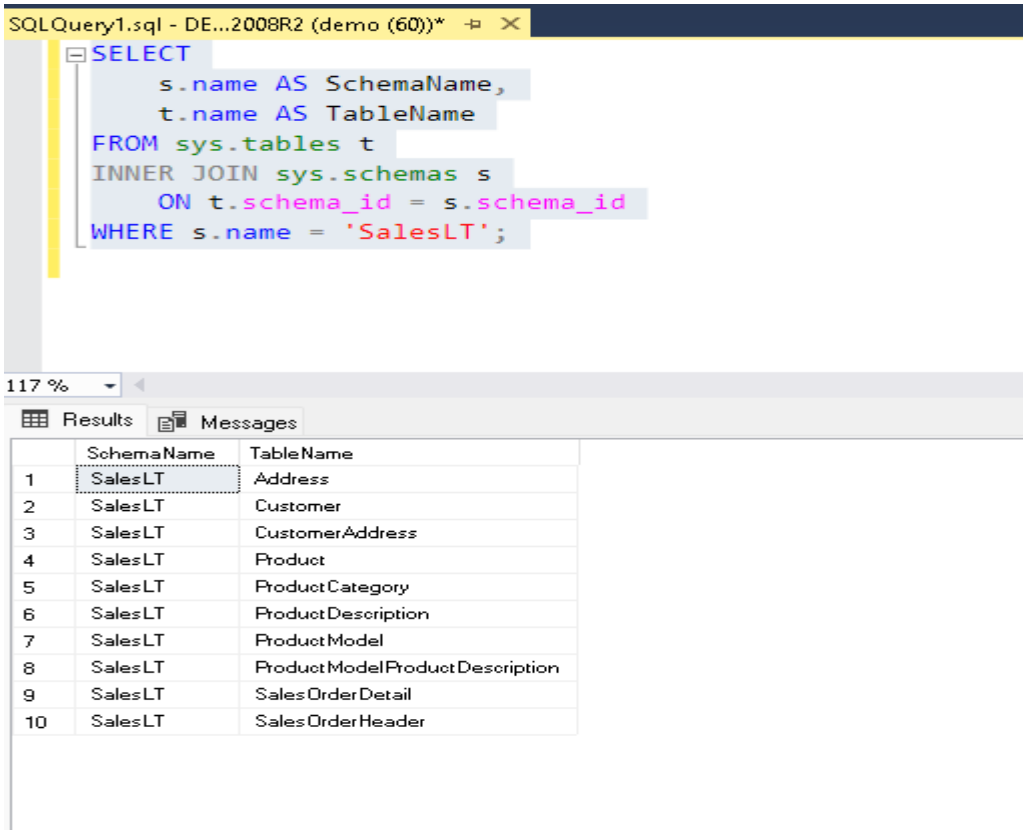
Lookup activity can retrieve a dataset from any of the data sources supported like sql server



Setting source data set for lookup to connect sql server

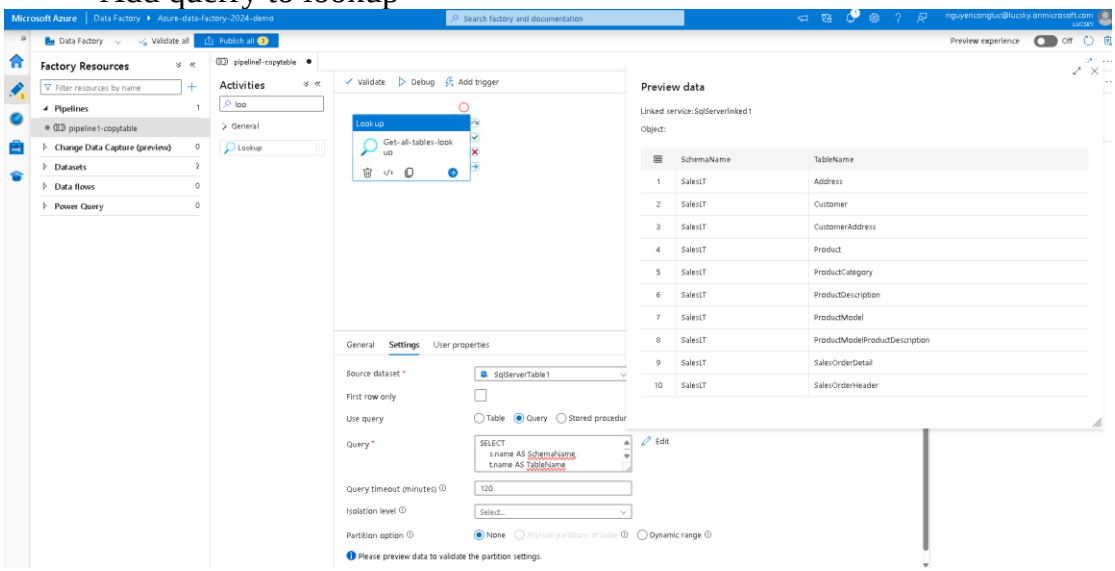


Create querry copy all table and schema

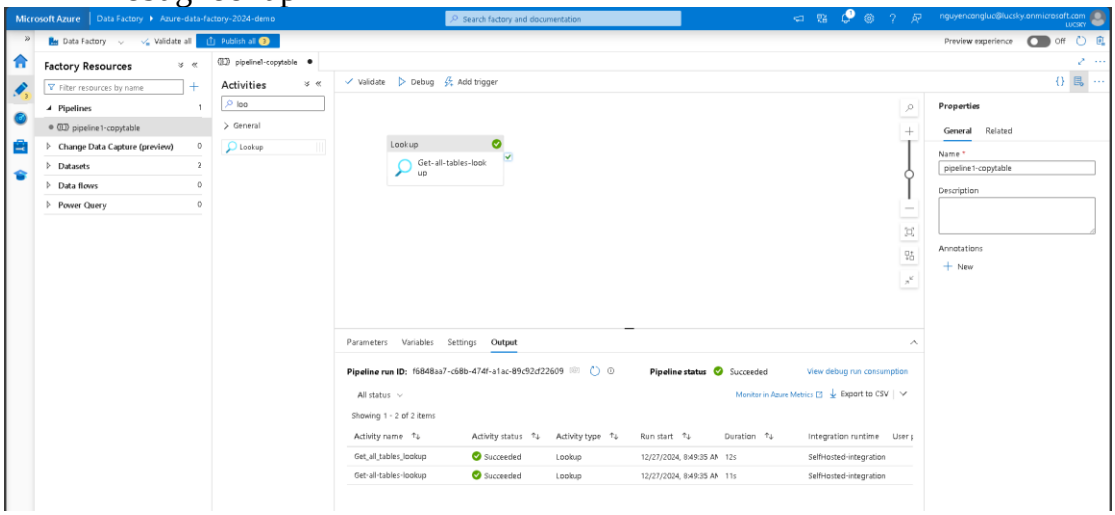


	SchemaName	TableName
1	SalesLT	Address
2	SalesLT	Customer
3	SalesLT	CustomerAddress
4	SalesLT	Product
5	SalesLT	ProductCategory
6	SalesLT	ProductDescription
7	SalesLT	ProductModel
8	SalesLT	ProductModelProductDescription
9	SalesLT	SalesOrderDetail
10	SalesLT	SalesOrderHeader

Add querry to lookup

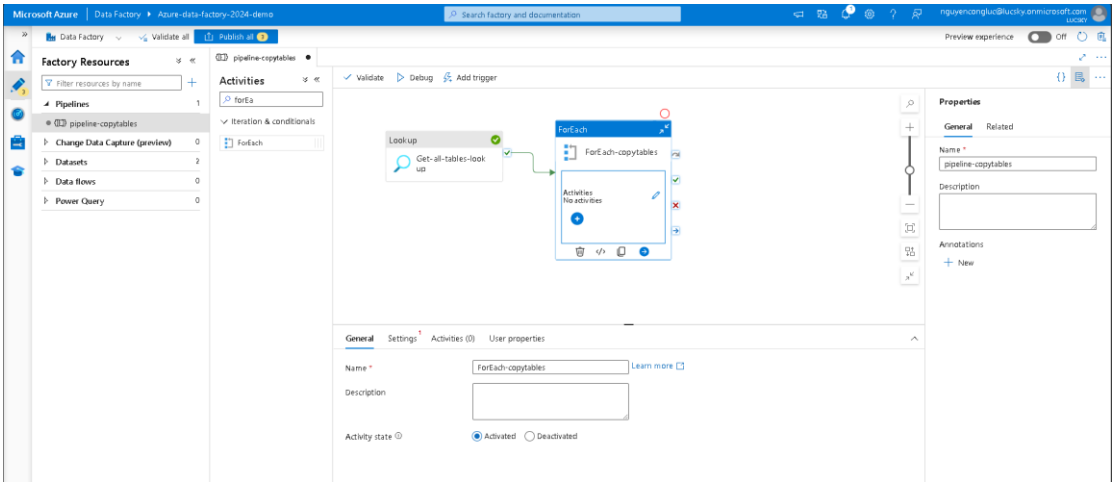


Debug lookup



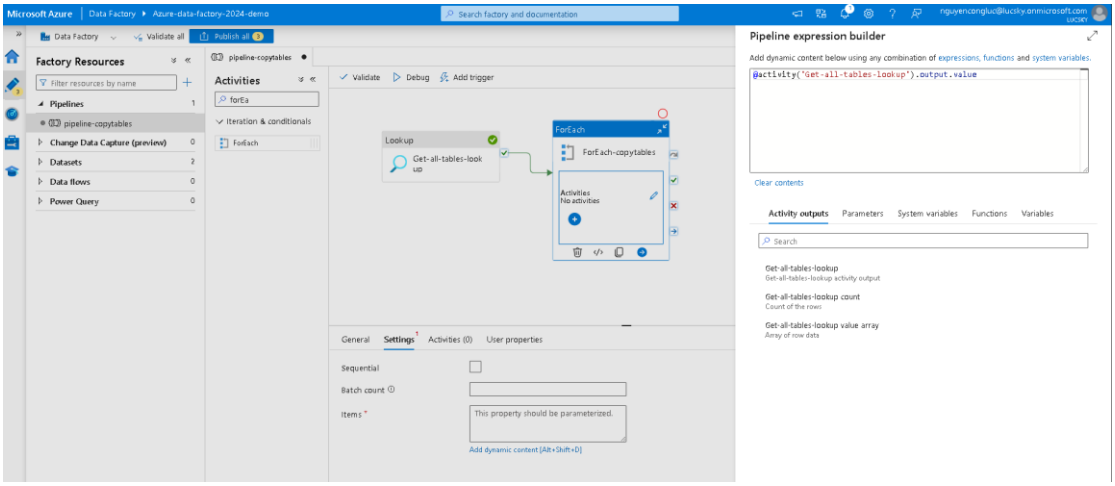
3.2.2 Create forEach

In this project, the ForEach activity allows you to iterate over the list of table names from the Lookup and perform actions (like data extraction or transformation) on each table



Add Pipeline expression builder to forEach.

The Pipeline Expression Builder in the ForEach activity's Items setting allows you to create dynamic expressions that reference outputs or parameters, enabling the pipeline to iterate over data (e.g., table names) and perform actions dynamically during the loop.

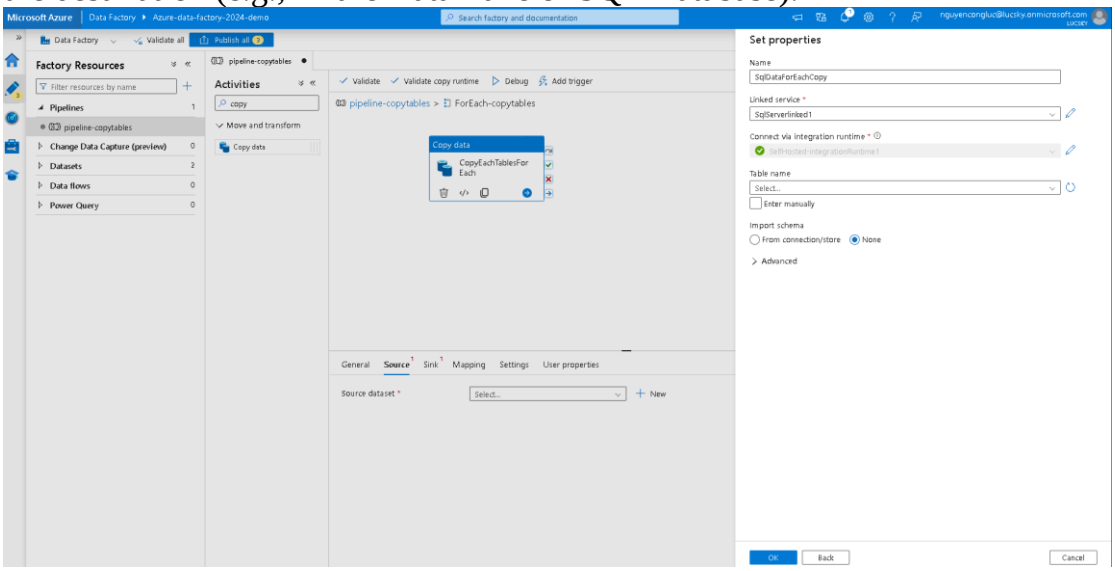


Add Copy Data Activity inside forEach

In the project, adding a Copy Data activity inside ForEach allows you to dynamically copy data from each table in the schema to a target destination (e.g., Azure Data Lake or SQL Server) during the pipeline execution.

Add source data to Copy Data

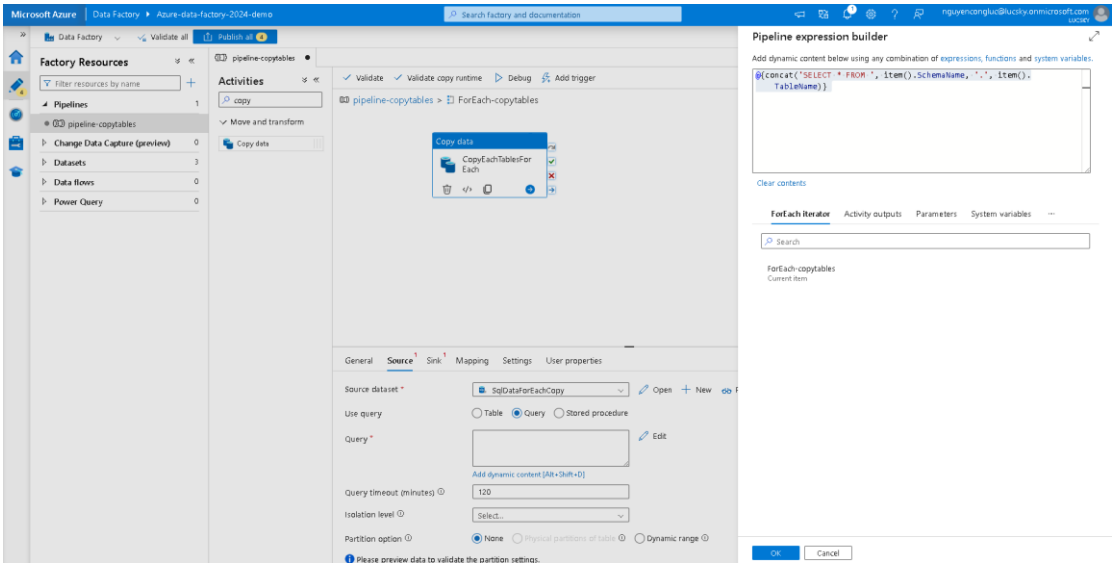
In the Copy Data activity, the Source dataset defines the structure and location of the data to be copied. In your project, it specifies the source from which data will be extracted, such as SQL Server tables in the SalesLT schema. It tells the pipeline where to pull the data from before it is copied to the destination (e.g., Azure Data Lake or SQL Database).



Add query to Copy Data

The query dynamically generates and executes a query to select all data from each table in the SalesLT schema. The Copy Data activity then extracts this data and copies it to a specified destination (e.g., Azure Data Lake or another database).

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

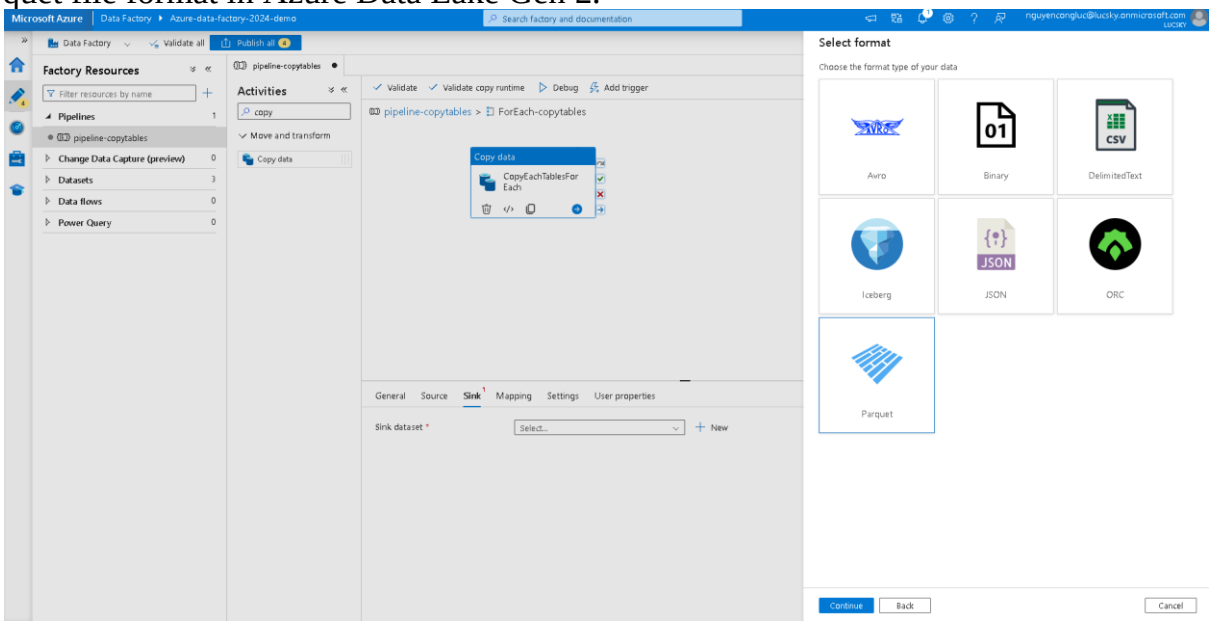


Add query to Copy Data

The query dynamically generates and executes a query to select all data from each table in the SalesLT schema. The Copy Data activity then extracts this data and copies it to a specified destination (e.g., Azure Data Lake or another database)

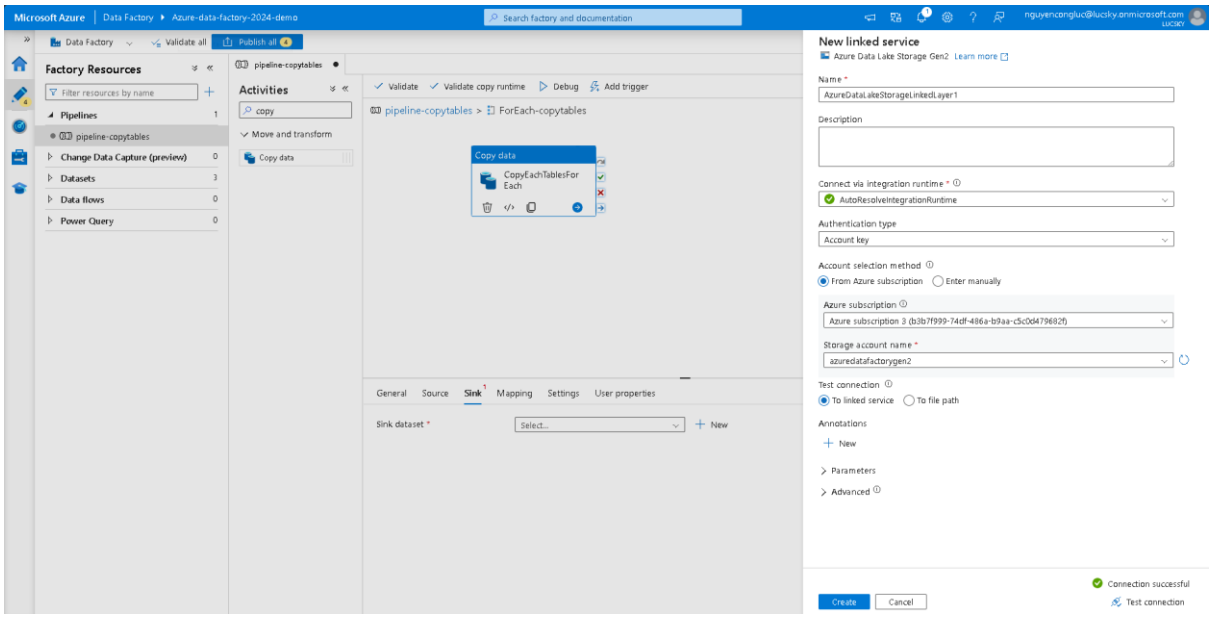
Adding a Sink dataset

Set up file in Sink dataset: with the Parquet format in Azure Data Lake Gen 2 in the Copy Data activity allows you to store the copied data in a Parquet file format in Azure Data Lake Gen 2.

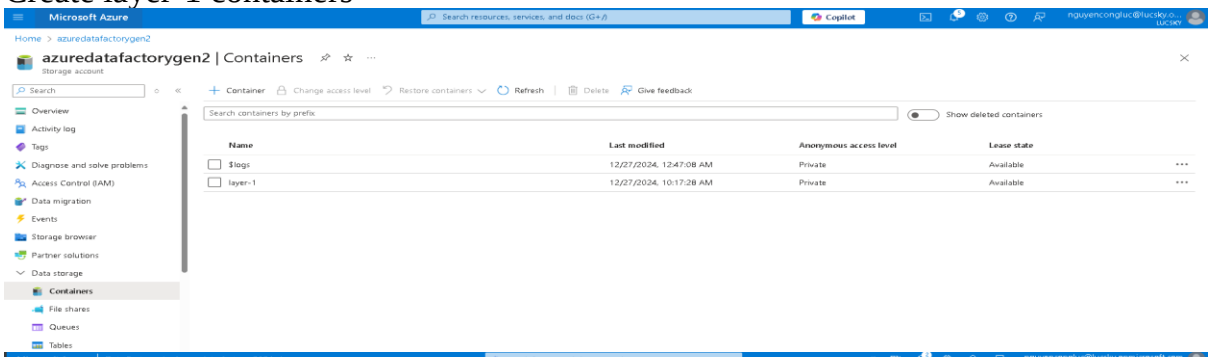


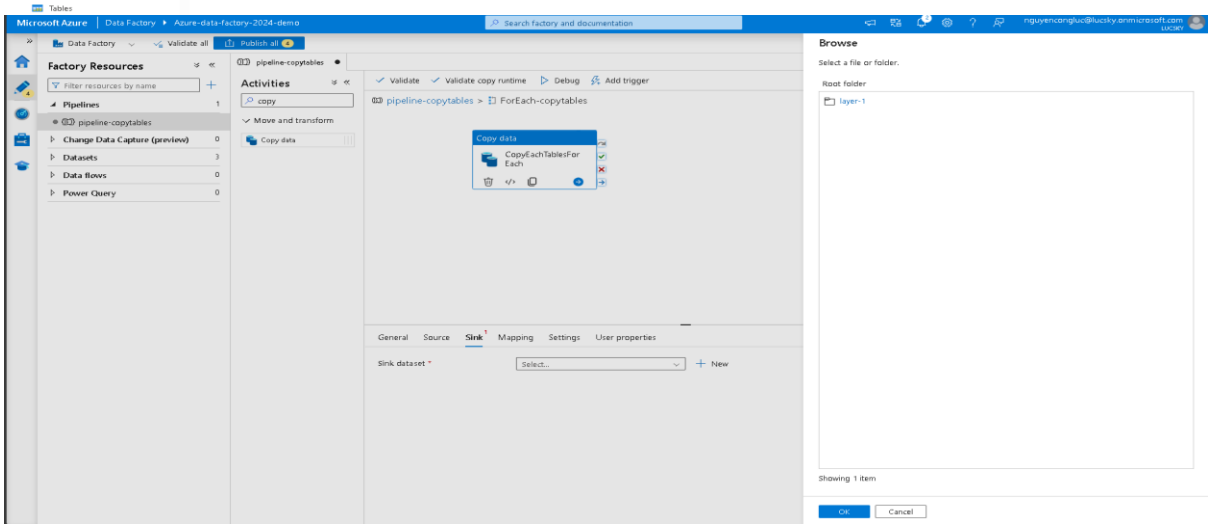
Set up linked service in Sink dataset:

It defines how ADF connects to external data stores like Azure Data Lake Storage Gen 2. The linked service provides authentication details, endpoints, and other necessary configurations for connecting to the storage account.

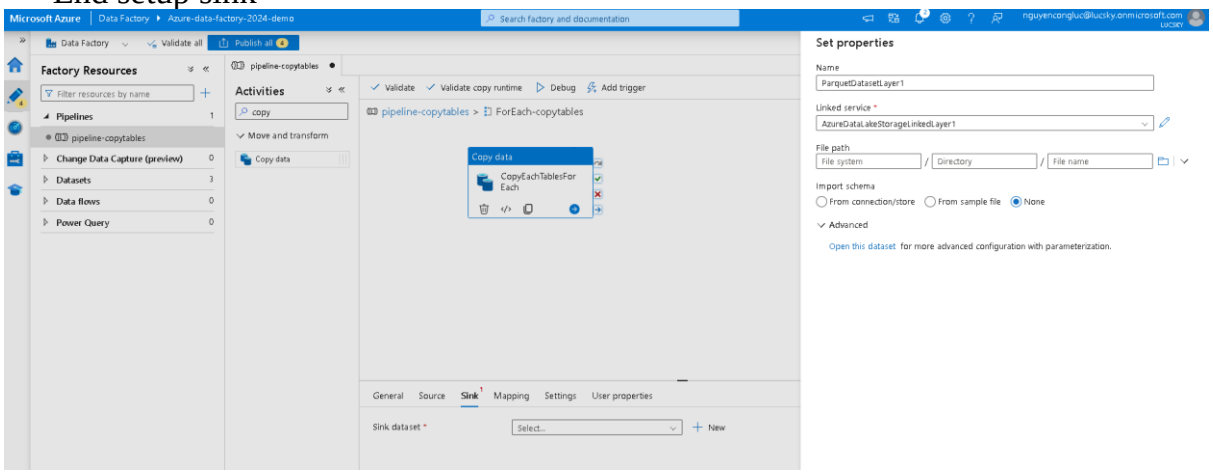


Create layer-1 containers



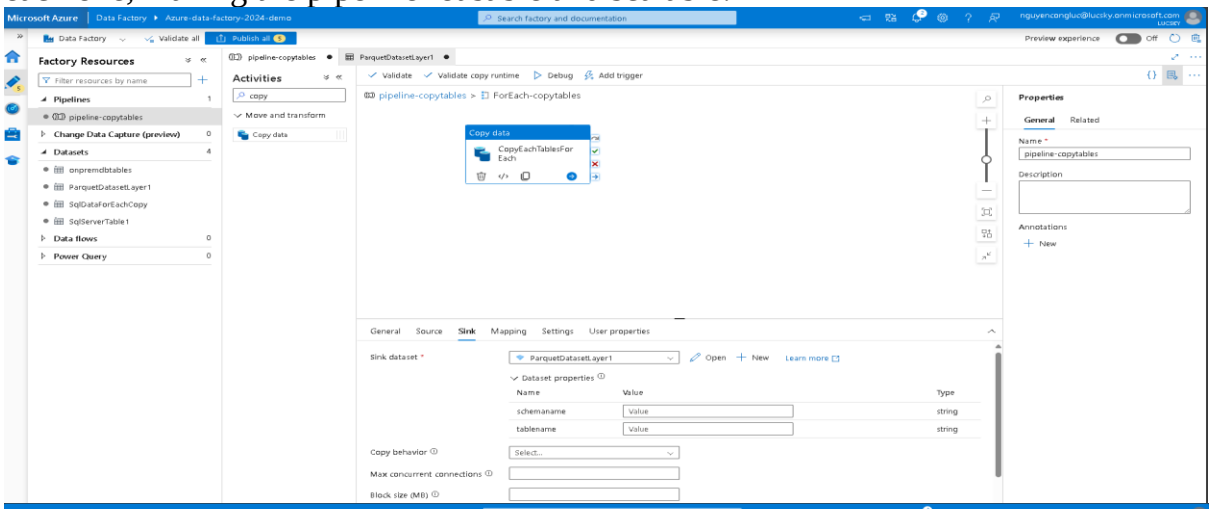


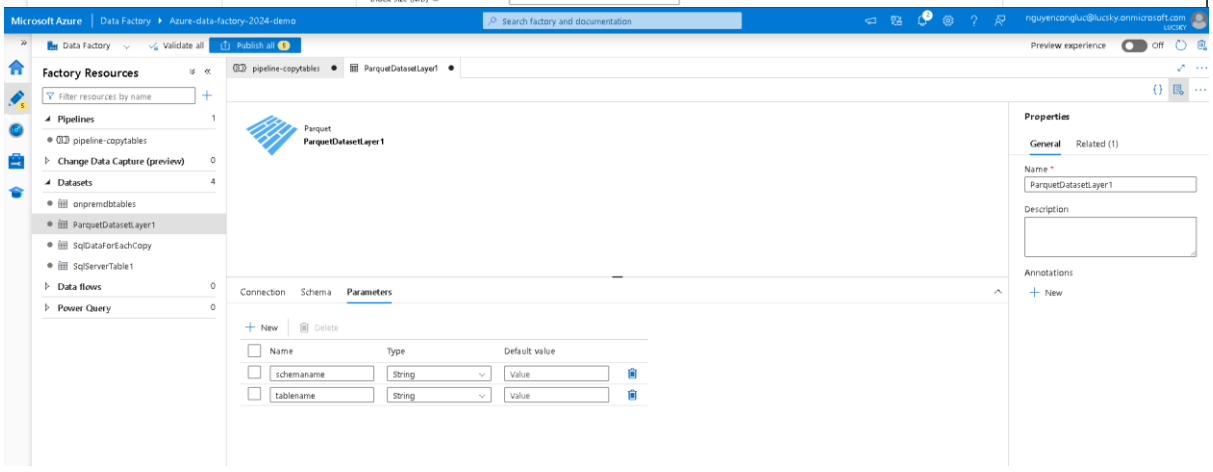
End setup sink



Set parameters for ParquetDatasetLayer1 in open of sink dataset

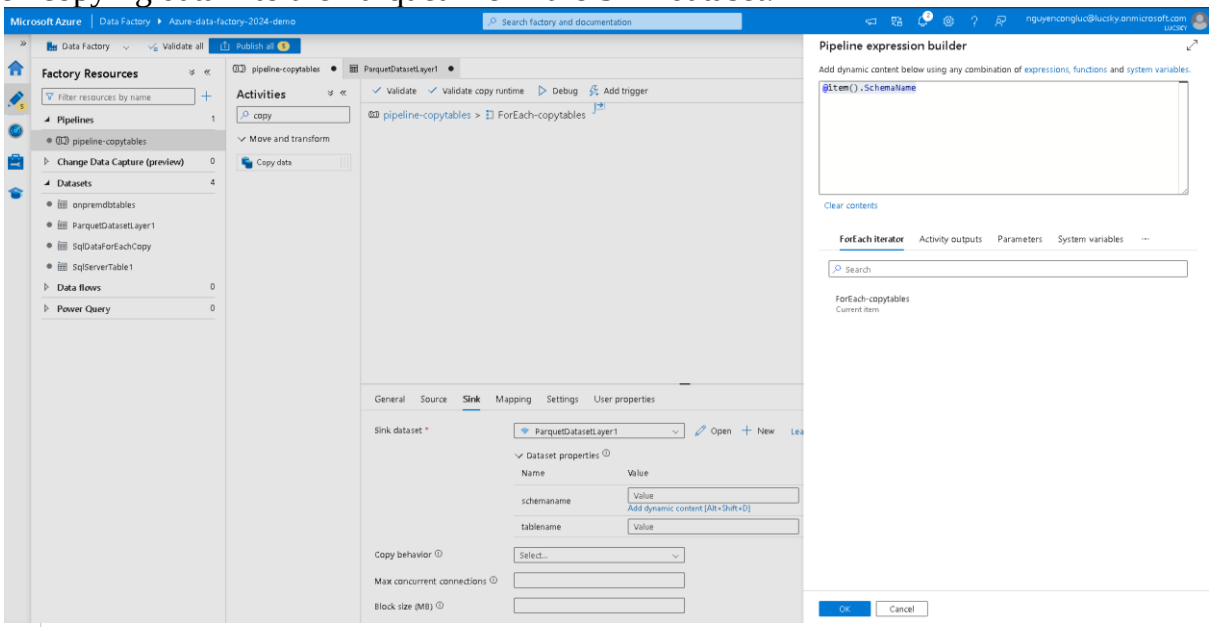
The schemaname and tablename parameters allow dynamic handling of data in the Sink dataset. They enable flexibility by letting you pass different schema and table names at runtime, which can be used to create dynamic file paths or names. This helps store data from various tables into different Parquet files without creating separate datasets for each one, making the pipeline reusable and scalable.

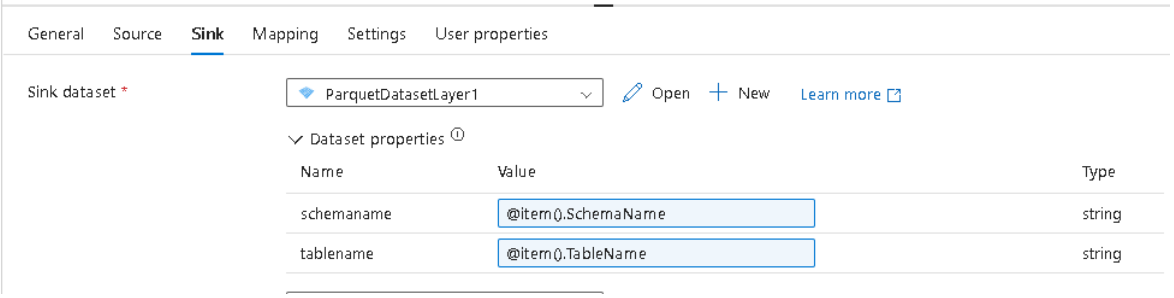




Setup value for names

In this case, @item().SchemaName, @item().TableName refers to the schema name and table name that are dynamically passed through the pipeline (likely from a previous activity like Lookup or ForEach) for each table. This allows you to dynamically use the schema name, table name when copying data into the Parquet file in the Sink dataset.



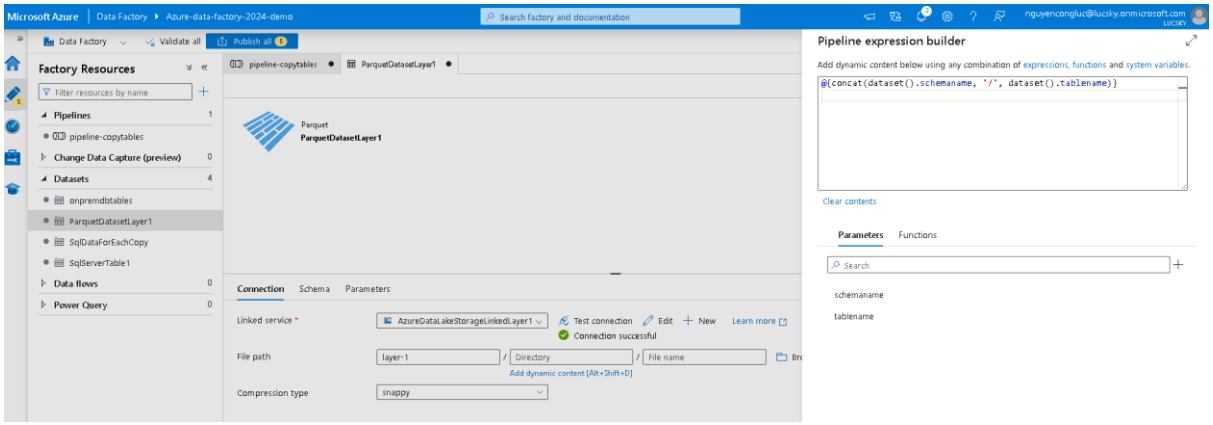


Setting the directory in the Parquet Sink dataset

Setting the directory as

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

in the Parquet Sink dataset is used to dynamically create a folder structure in your Azure Data Lake Storage. This concatenates the schema name and table name to define the folder where the Parquet file will be stored.

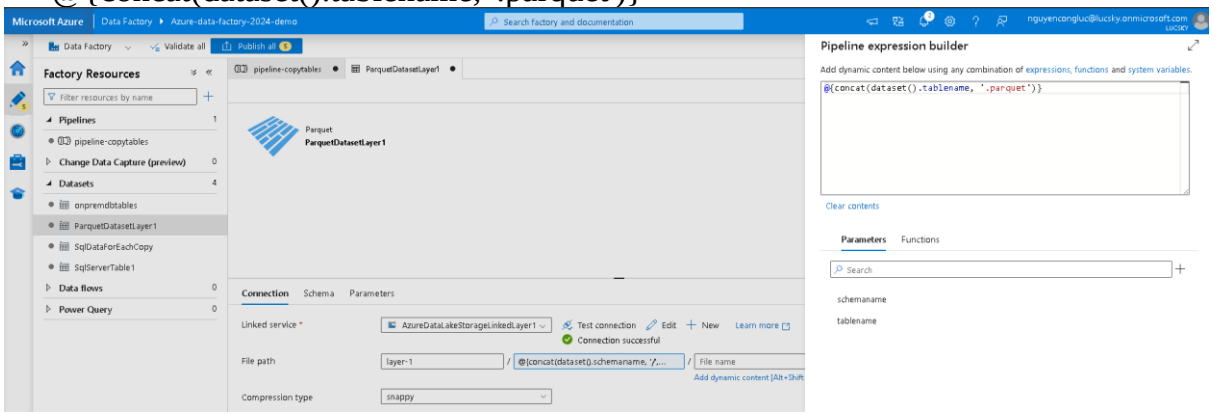


Setting the file name as

@{concat(dataset().tablename, '.parquet')}

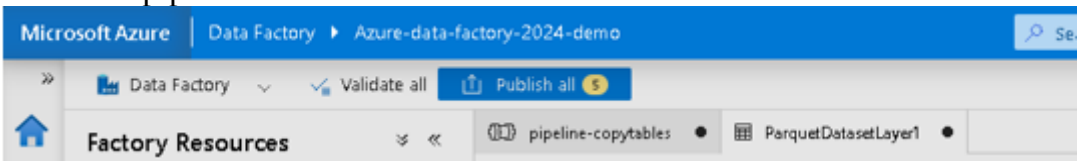
in the Sink dataset for Parquet means the file will be named dynamically based on the tablename.

@{concat(dataset().tablename, '.parquet')}

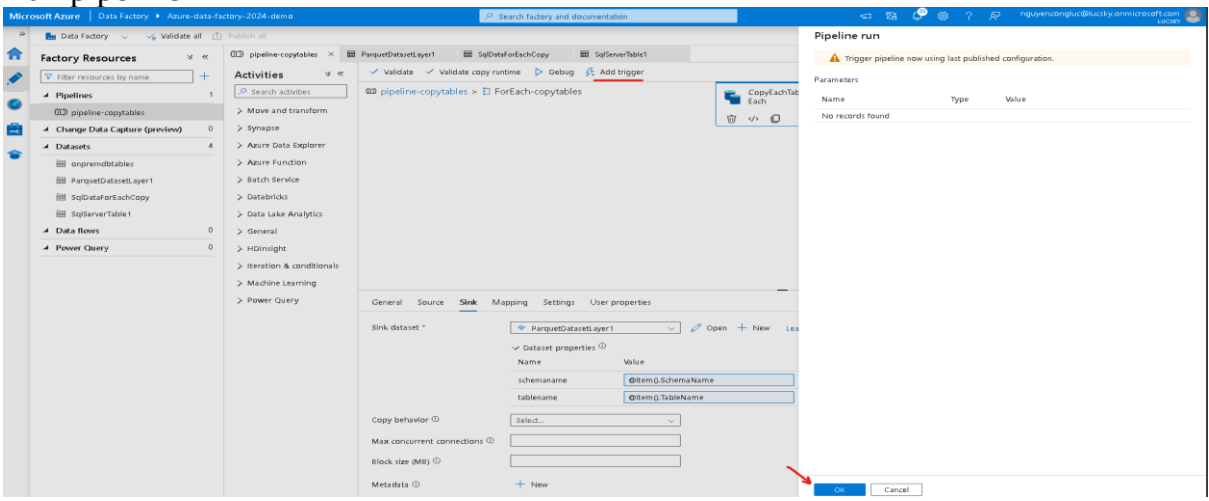


3.2.3 Run pipeline

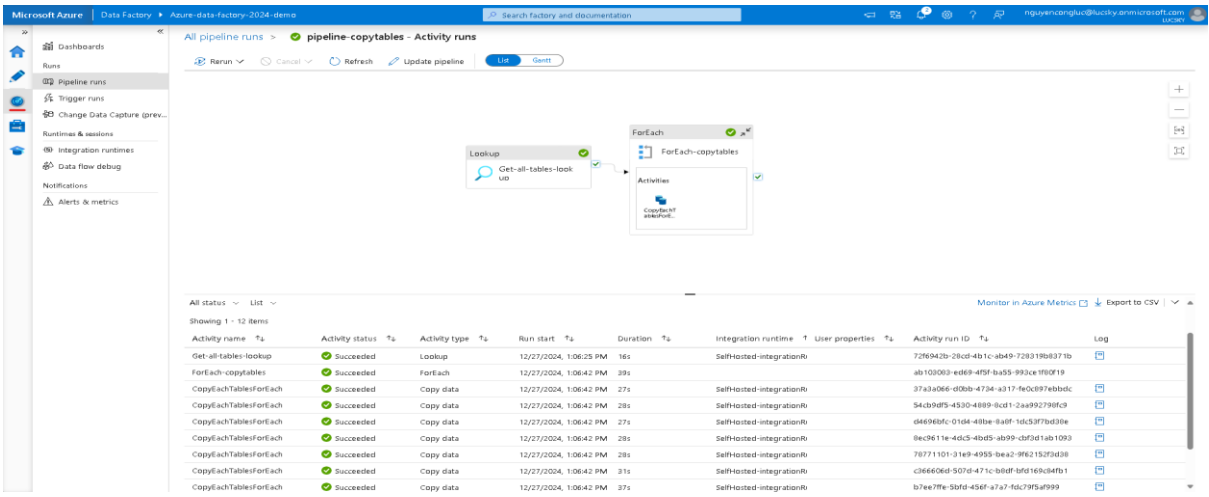
Public all pipelines



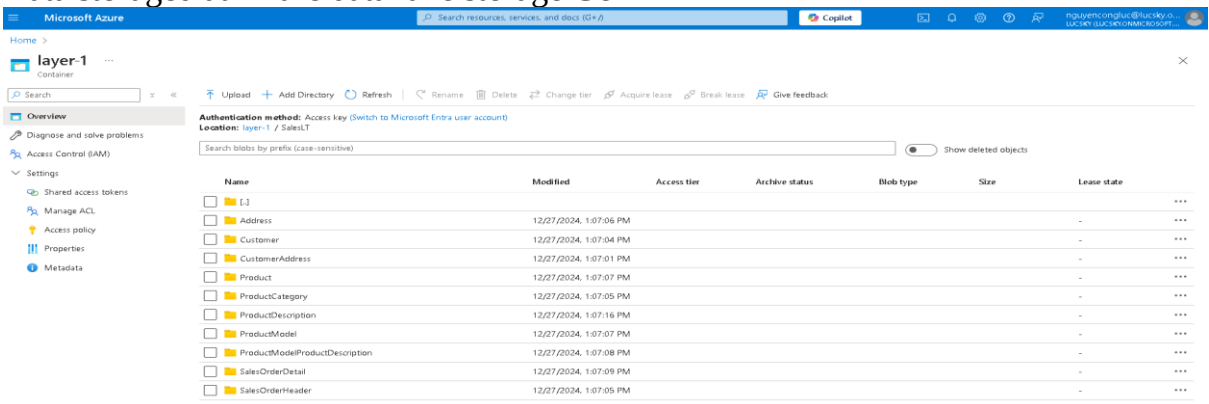
Run pipeline



Monitor



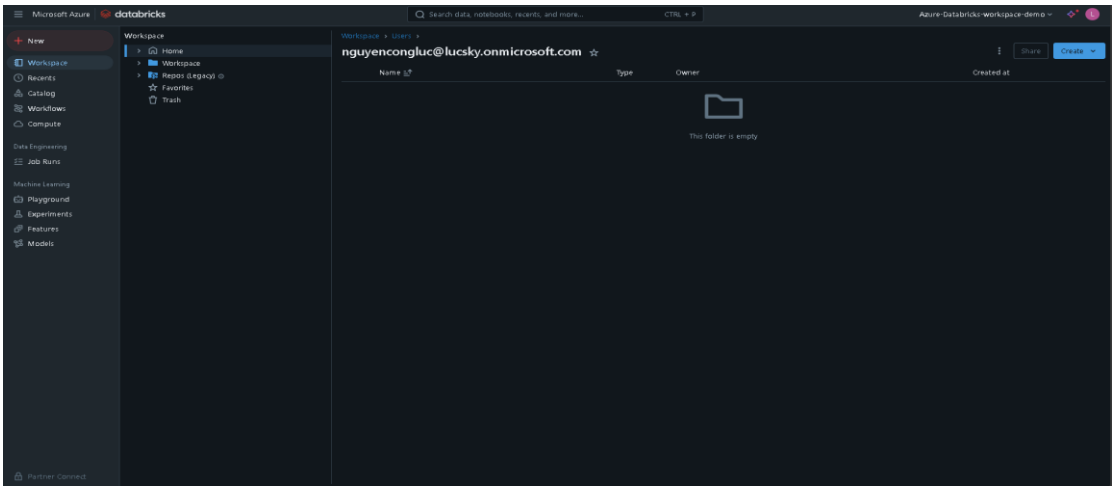
Data storaged at Azure data lake storage Gen 2



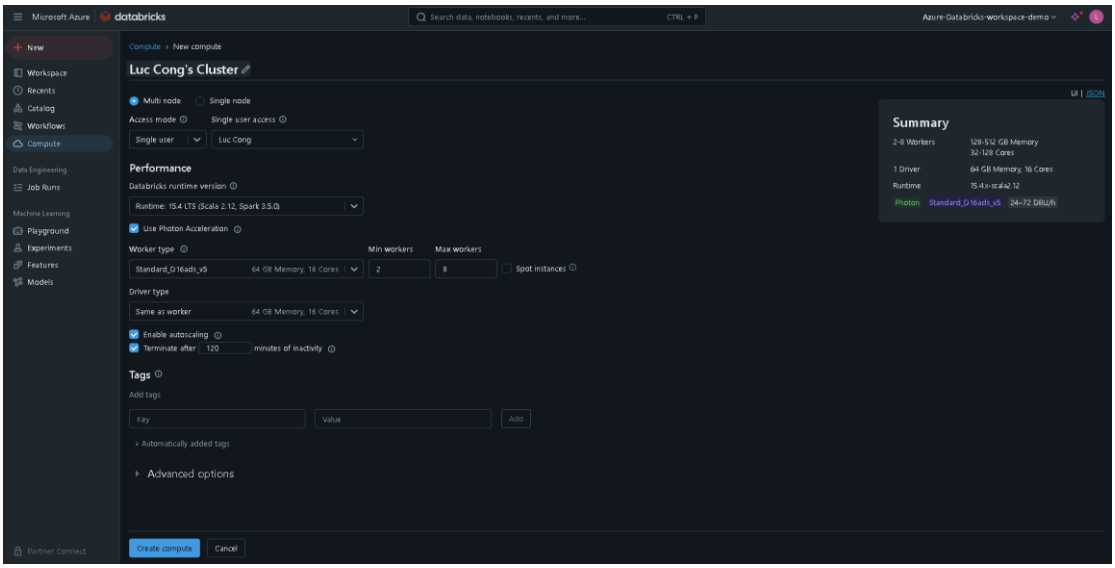
CHAPTER 4. Data Transformation(T)

4.1 Create notebook to transformation data in databricks

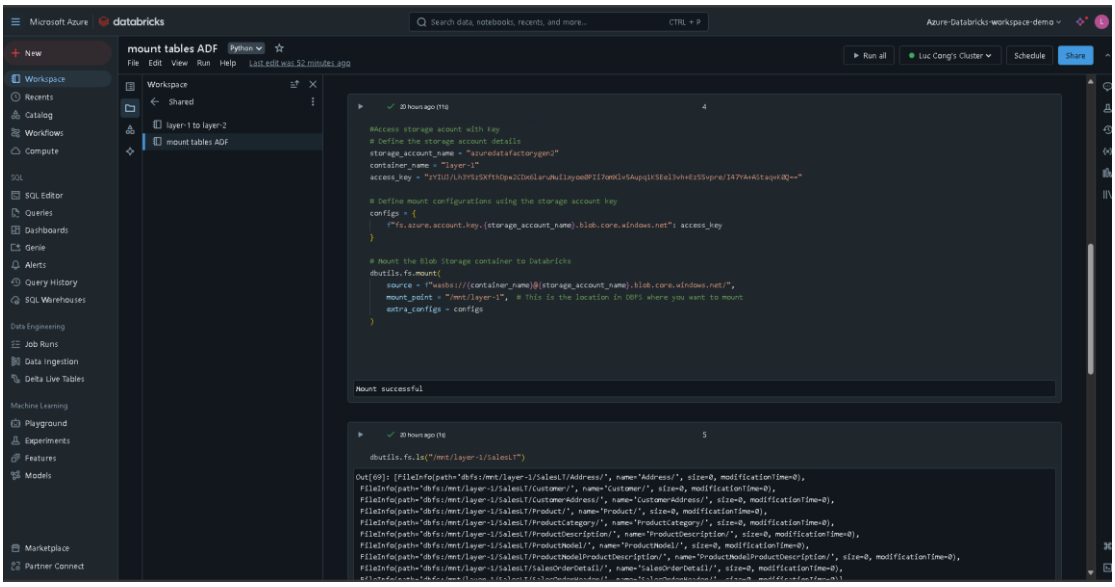
Using Databricks for tranformation data



Create cluster

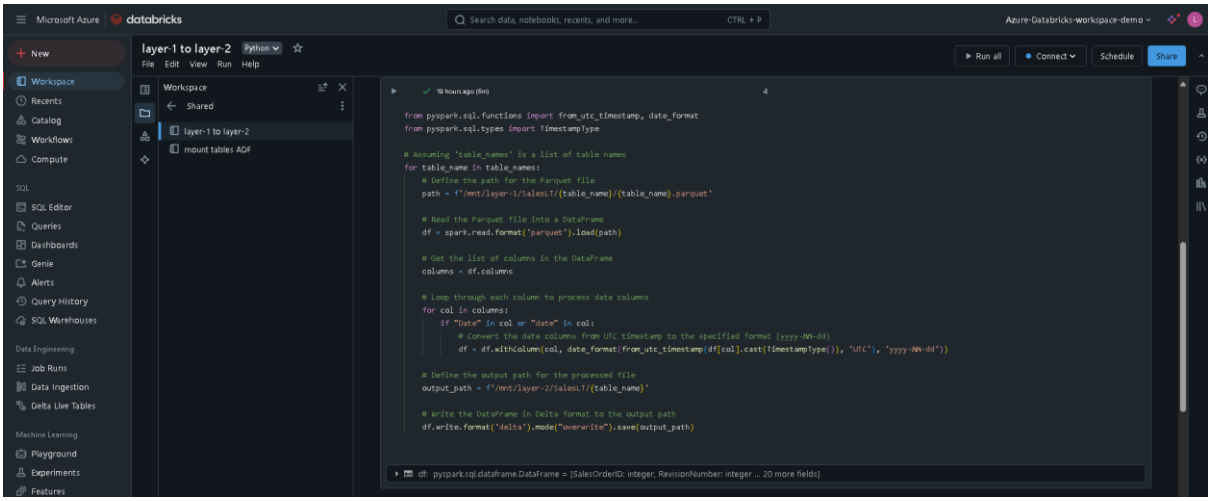


Connect Databricks Azure data lake storage gen 2
This code allows Databricks to interact with Azure data lake storage gen 2as if it's part of the local file system by mounting the specified container. Once mounted, you can access the files stored in that container directly within Databricks

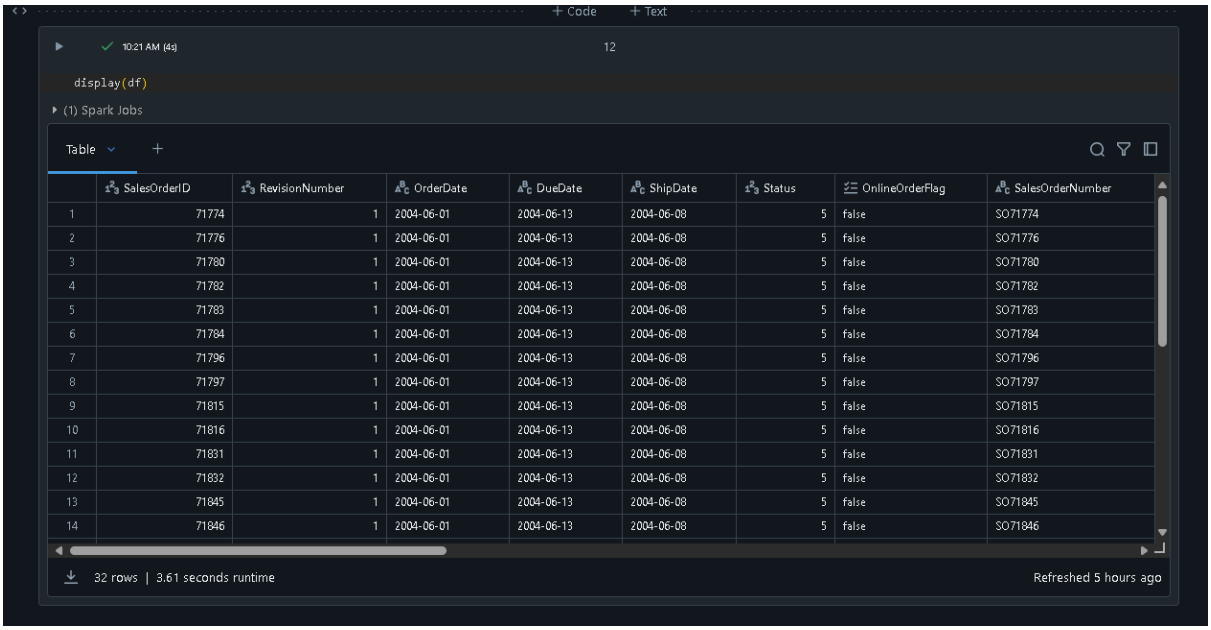


Data transformation to all the SalesLT tables

Address collum need convert to other data type, from date time to timestamp

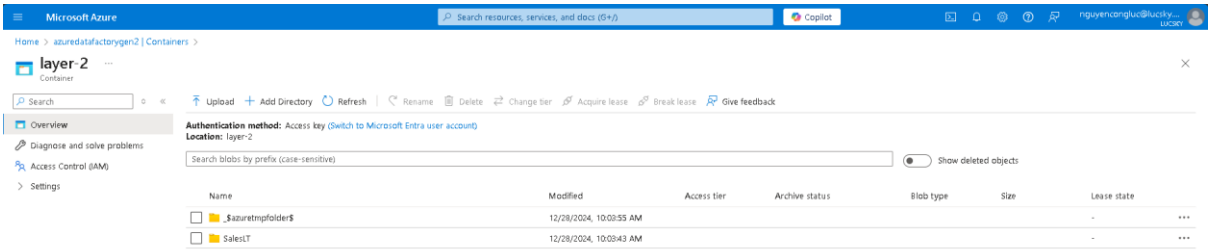


Result



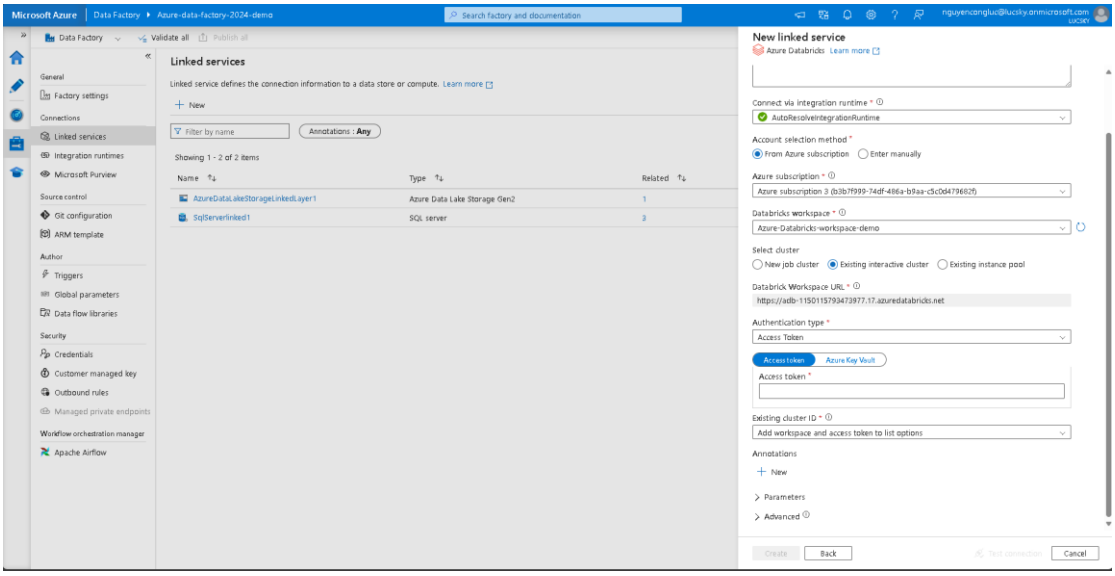
	SalesOrderID	RevisionNumber	OrderDate	DueDate	ShipDate	Status	OnlineOrderFlag	SalesOrderNumber
1	71774	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71774
2	71776	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71776
3	71780	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71780
4	71782	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71782
5	71783	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71783
6	71784	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71784
7	71796	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71796
8	71797	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71797
9	71815	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71815
10	71816	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71816
11	71831	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71831
12	71832	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71832
13	71845	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71845
14	71846	1	2004-06-01	2004-06-13	2004-06-08	5	false	SO71846

Layer 2- Clean data created by Databricks is created at Azure data lake gen 2



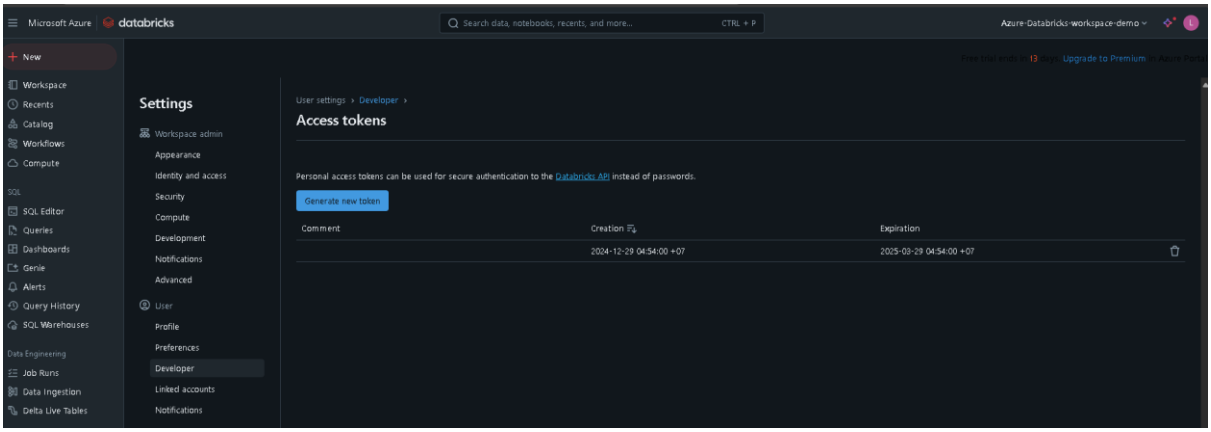
4.2 Link Azure databricks to pipeline in Azure data factory

Adding a Databricks linked service in Azure Data Factory (ADF) allows ADF to connect and interact with Databricks. This enables ADF to trigger Databricks notebooks or jobs within its pipelines for data transformation, automate workflows, and monitor execution. It integrates Databricks into the ETL process, ensuring seamless collaboration between ingestion (ADF) and transformation (Databricks).

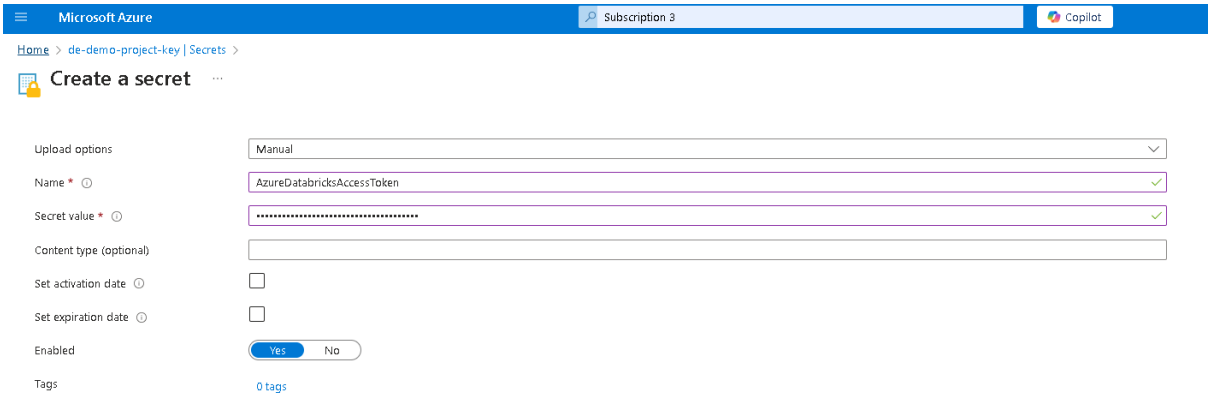


Create access token to access databricks on azure data factory

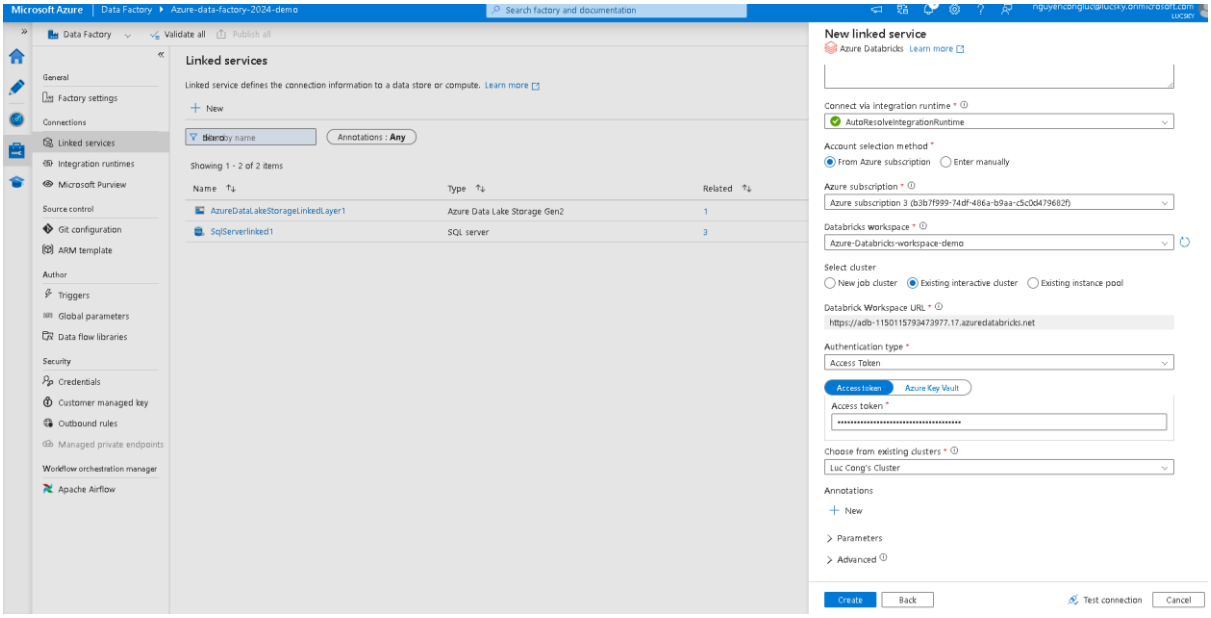
dapi632317582e85def72eee6294afff37fb-2



Save access token at secret of Key Vault to security

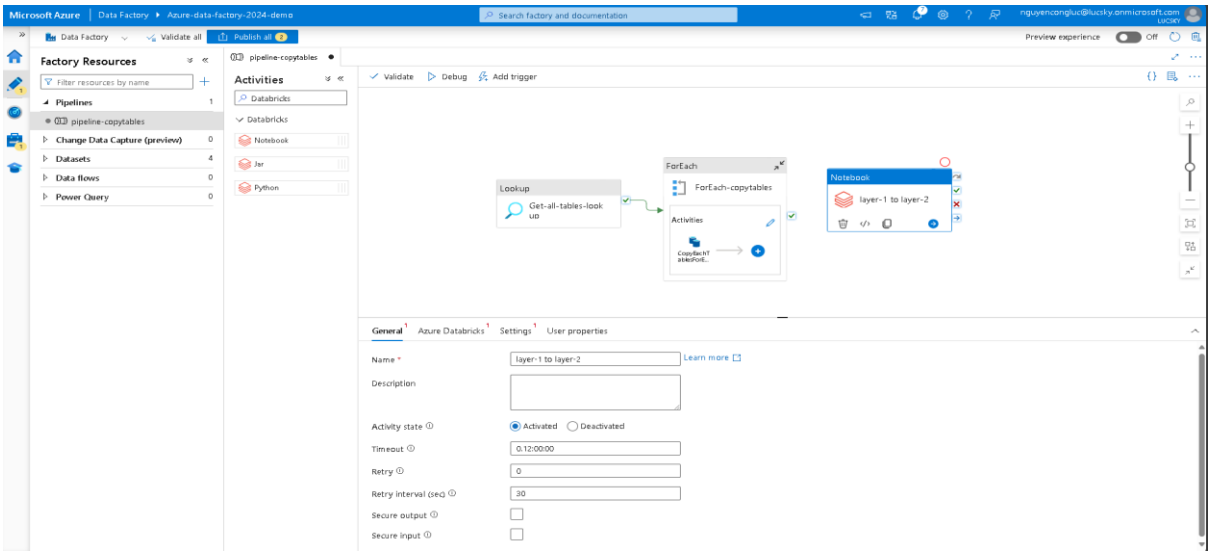


Create databricks linked service successfull

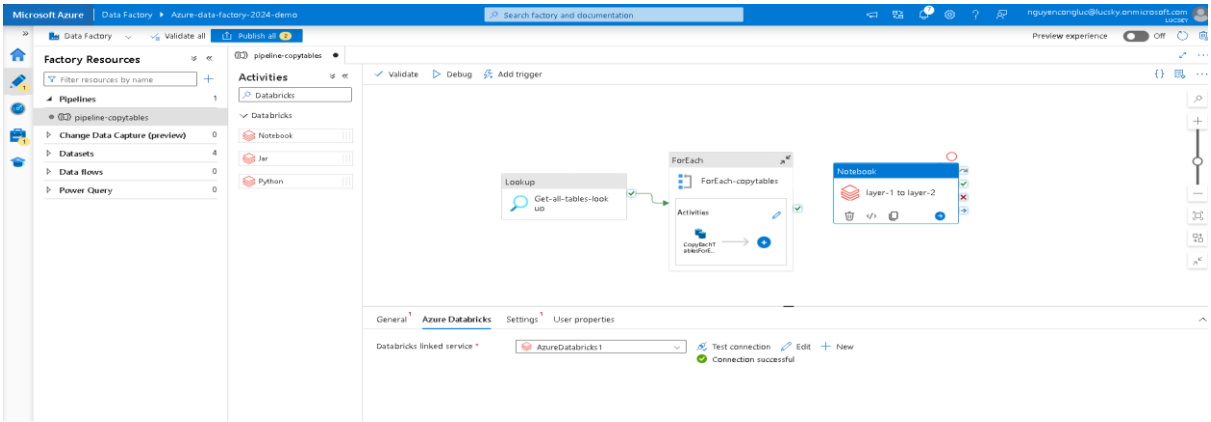

Microsoft Azure
Data Factory
Azure-data-factory-2024-demo
Search factory and documentation
Data Factory
Validate all
Publish all
General
Factory settings
Connections
Linked services
Integration runtimes
Microsoft Purview
Source control
Git configuration
ARM template
Author
Triggers
Global parameters
Data flow libraries
Security
Credentials
Customer managed key
Outbound rules
Managed private endpoints
Workflow orchestration manager
Apache Airflow
Linked services
Linked service defines the connection information to a data store or compute. Learn more
New
Filter by name
Annotations : Any
Showing 1 - 2 of 2 items
Name
Type
Related
AzureDataLakeStorageLinkedLayer1
Azure Data Lake Storage Gen2
1
SqlServerlinked1
SQL server
3
New linked service
Azure Databricks Learn more
Connect via integration runtime *
AutoResolveIntegrationRuntime
Account selection method *
From Azure subscription
Enter manually
Azure subscription *
Azure subscription 3 (b3b7f999-74df-486a-b9aa-c5c0d479682f)
Databricks workspace *
Azure-Databricks-workspace-demo
Select cluster
New job cluster
Existing interactive cluster
Existing instance pool
Databrick Workspace URL *
https://adb-1150115793473977.17.azuredatabricks.net
Authentication type *
Access Token
Access token
Azure Key Vault
Access token *
Choose from existing clusters *
Luc Cong's Cluster
Annotations
New
Parameters
Advanced
Create
Back
Test connection
Cancel

Adding a **Databricks Notebook Activity in pipeline**

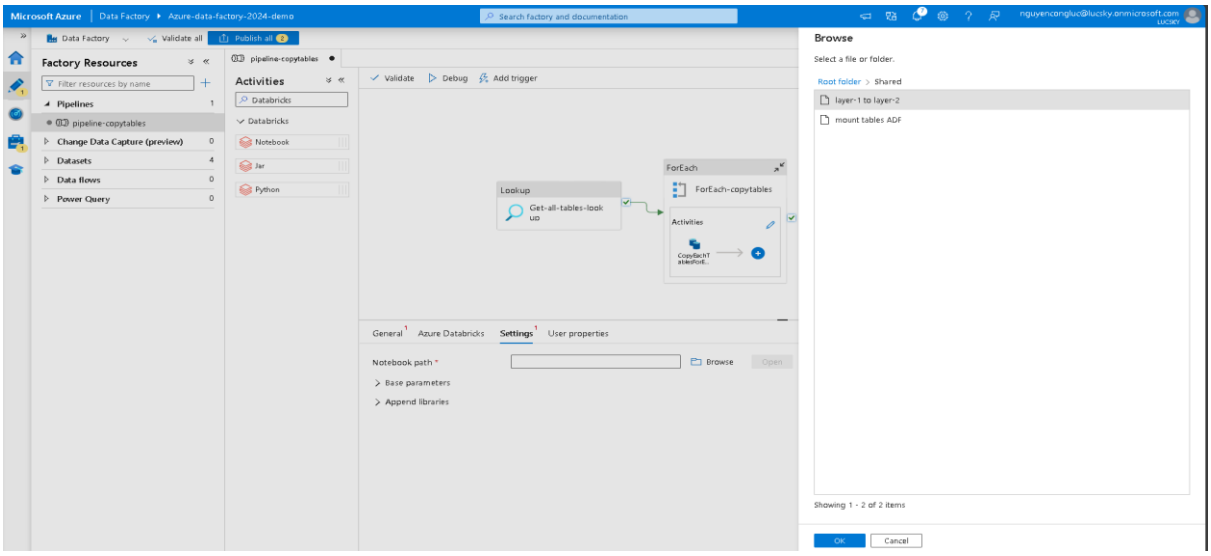
Adding a **Databricks Notebook Activity** named **"layer-1 to layer-2"** into an Azure Data Factory (ADF) pipeline is used for **data transformation**. Specifically, this activity processes and refines data from **Layer-1 (raw or ingested data)** to **Layer-2 (cleaned or structured data)** in the data lake.



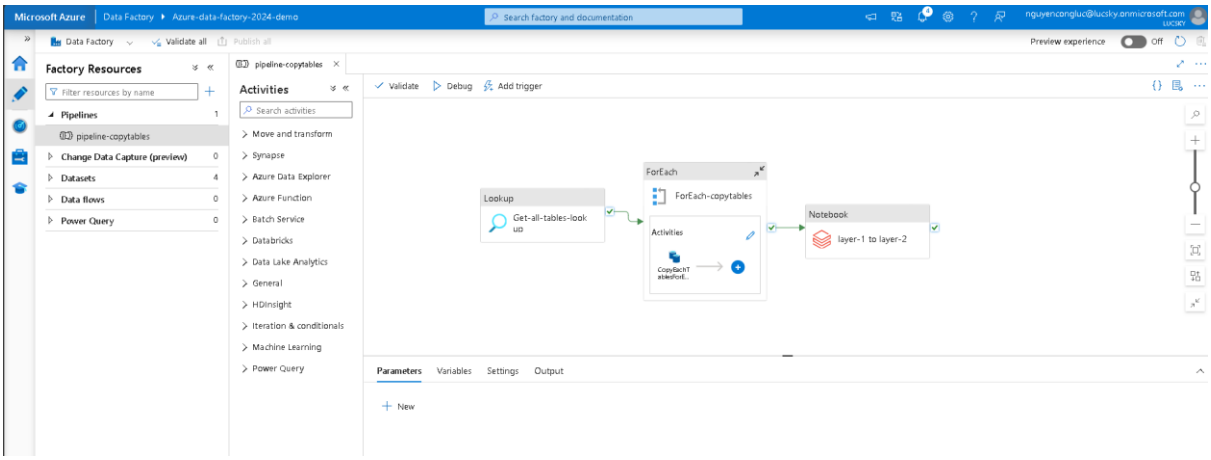
Set Databricks linked service



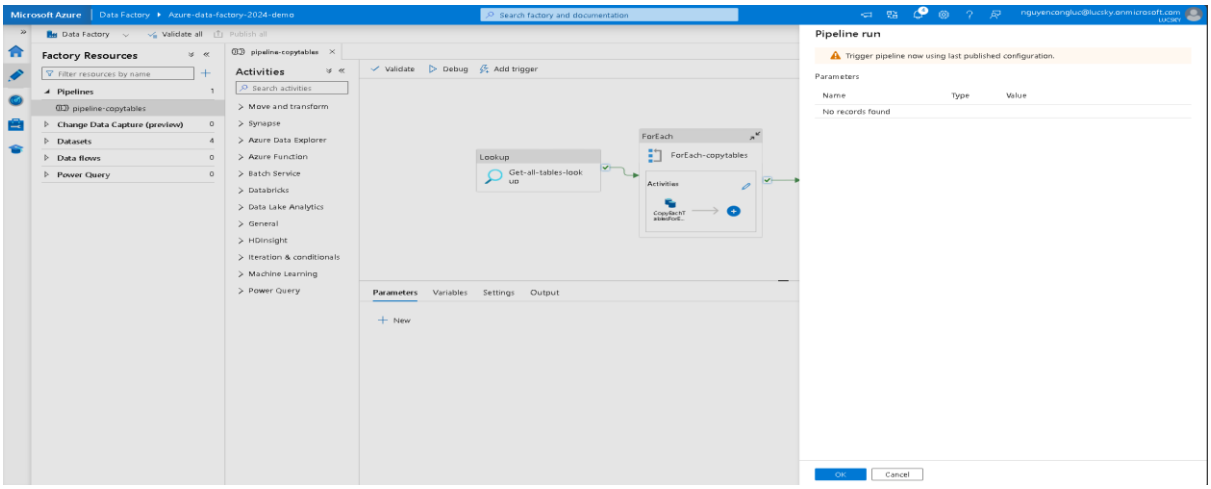
Set notebook path



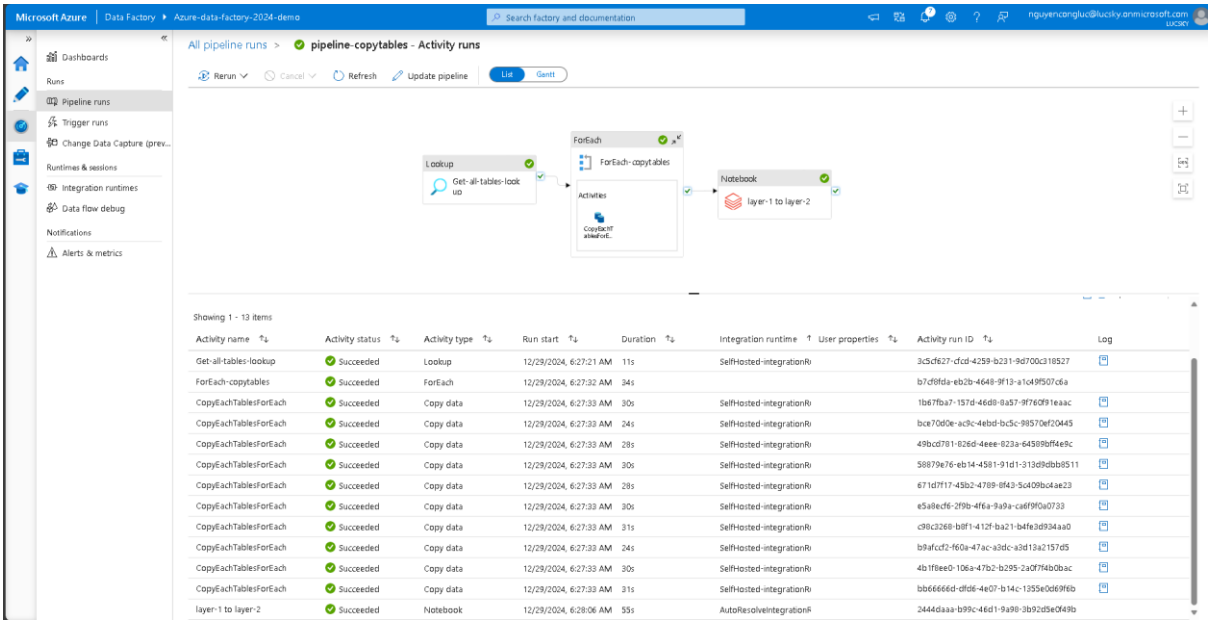
Complete pipeline for data



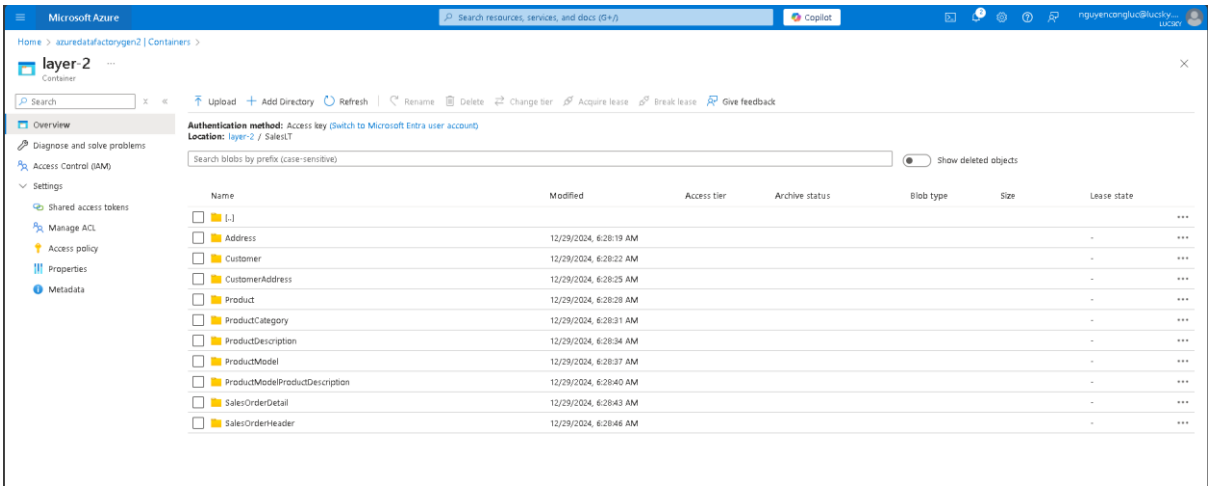
Running pipeline



successful result



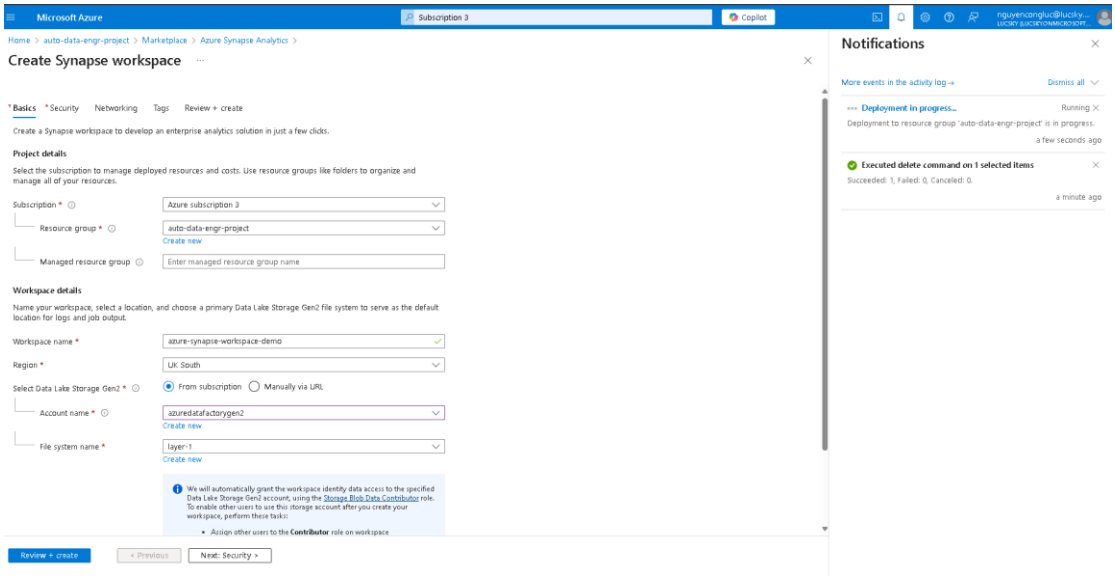
Processed data



CHAPTER 5. Data loading (L)

5.1 Create Synapse Workspace resource

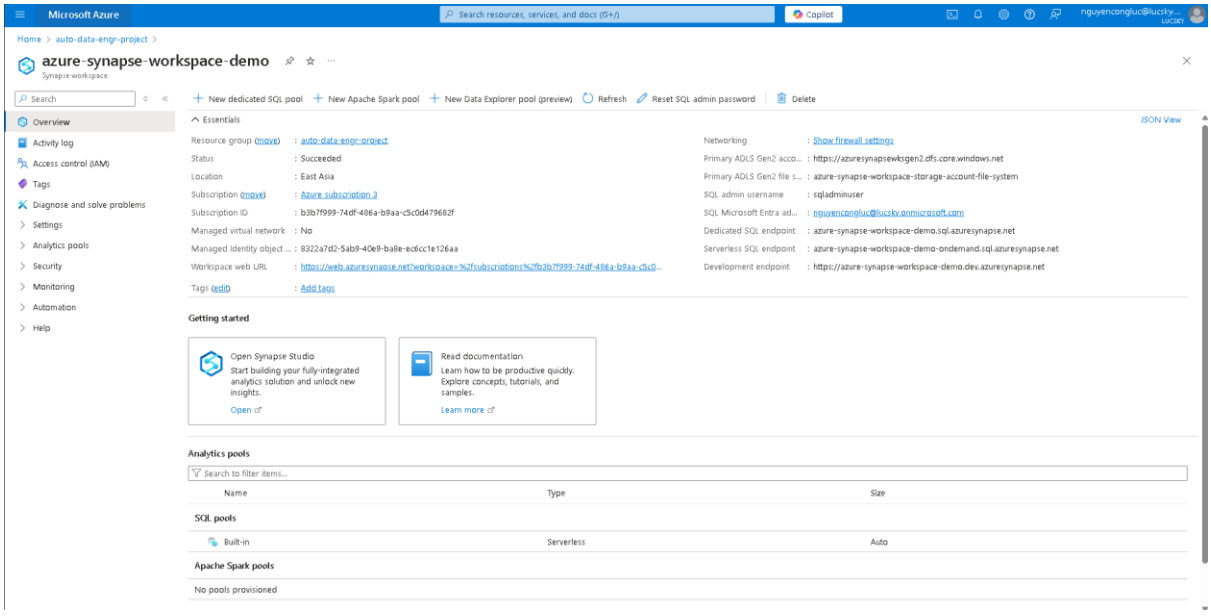
Synapse Workspace is used to load and prepare data for enterprise-level analytics and reporting..



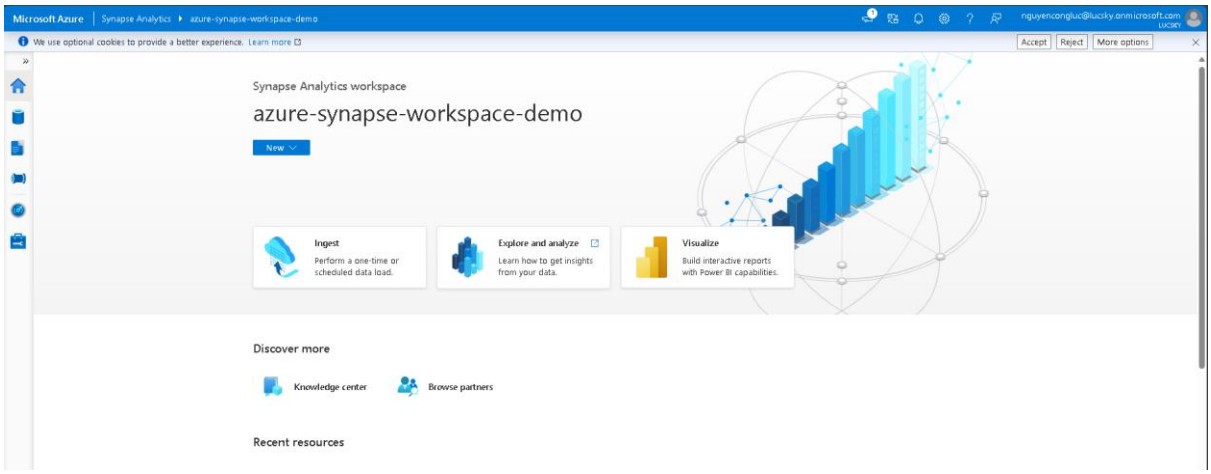
Account:

sqladminuser

abcd@1234



Dashboard of Azure Synapse

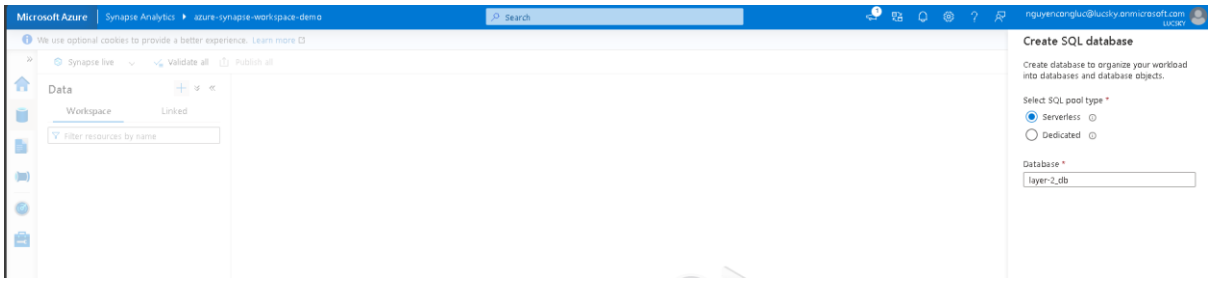


5.2 Create View from data of ADL Gen 2(Layer 2)

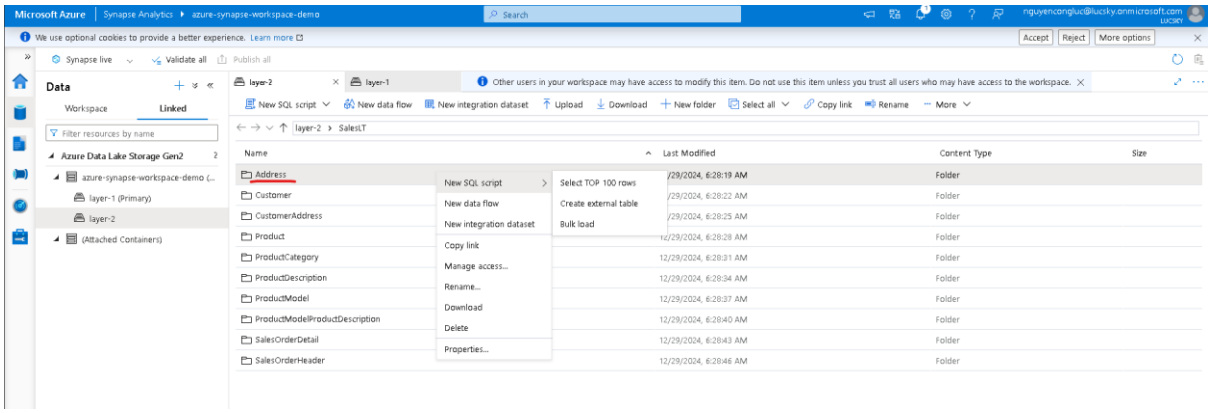
Dashboard of Azure Synapse

Dedicated SQL pool: Create a real database in Azure Synapse with distributed tables and fixed resource requirements.

Serverless SQL pool: Does not create a real database but only queries data from external sources (like Azure Data Lake Gen 2), does not require fixed resources.



We can query at all table in ADL Gen 2 when linked



Select TOP 100 rows

Address

Source folder format settings

Specify the format and layout of your data.

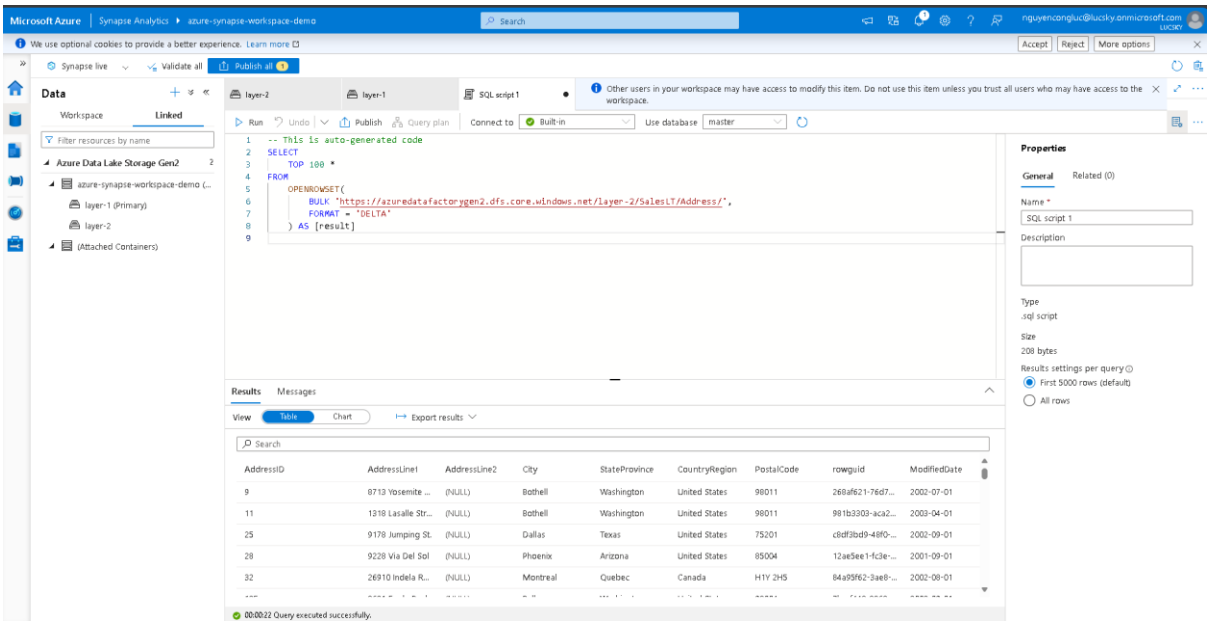
Folder path

https://azuredatafactorygen2.dfs.core.windows.net/layer-2/SalesLT/Address/

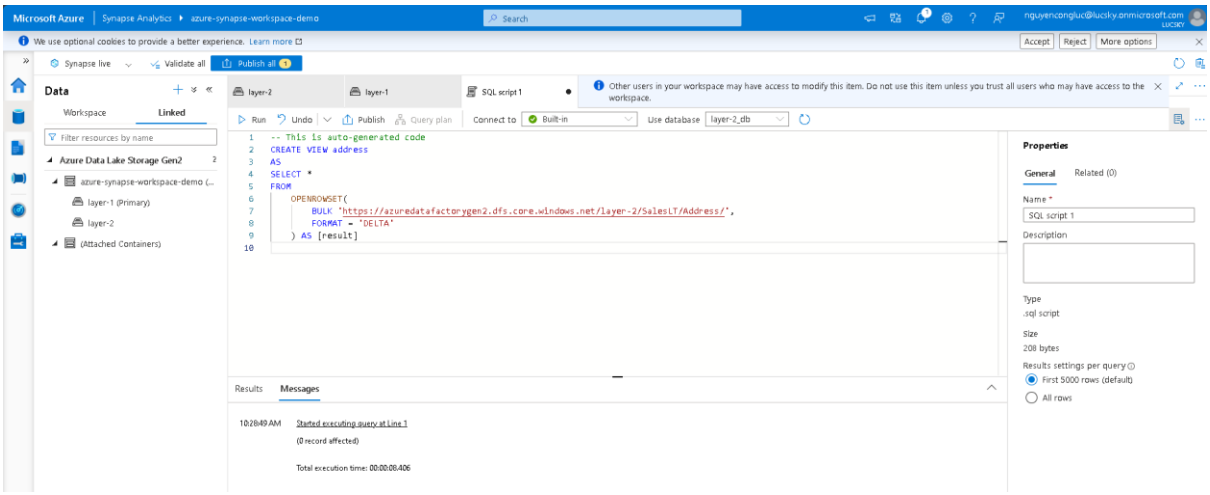
File type

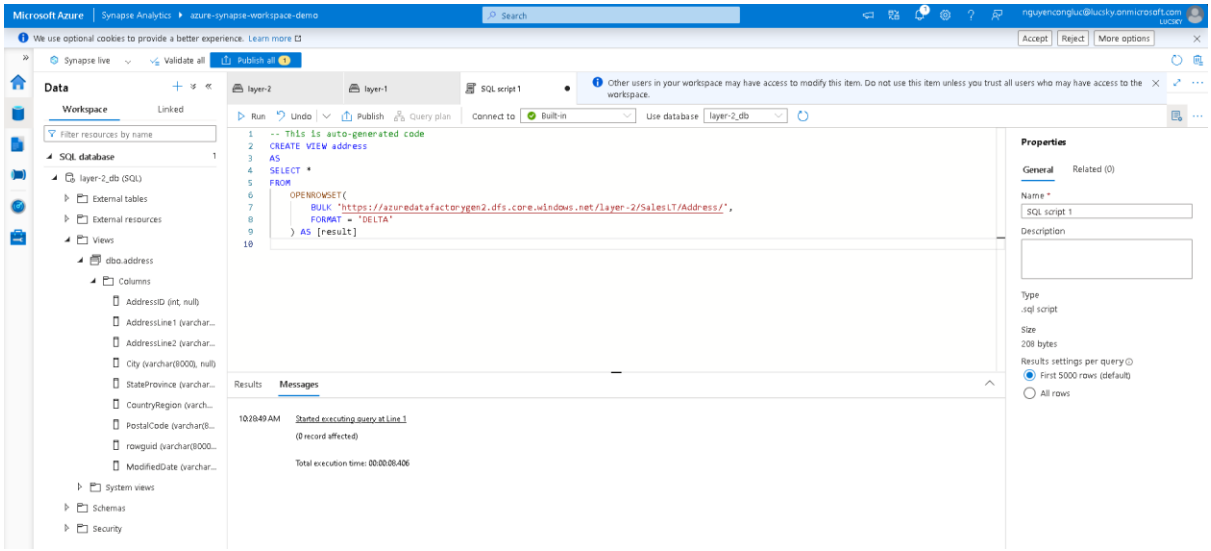
Delta format

Result of query

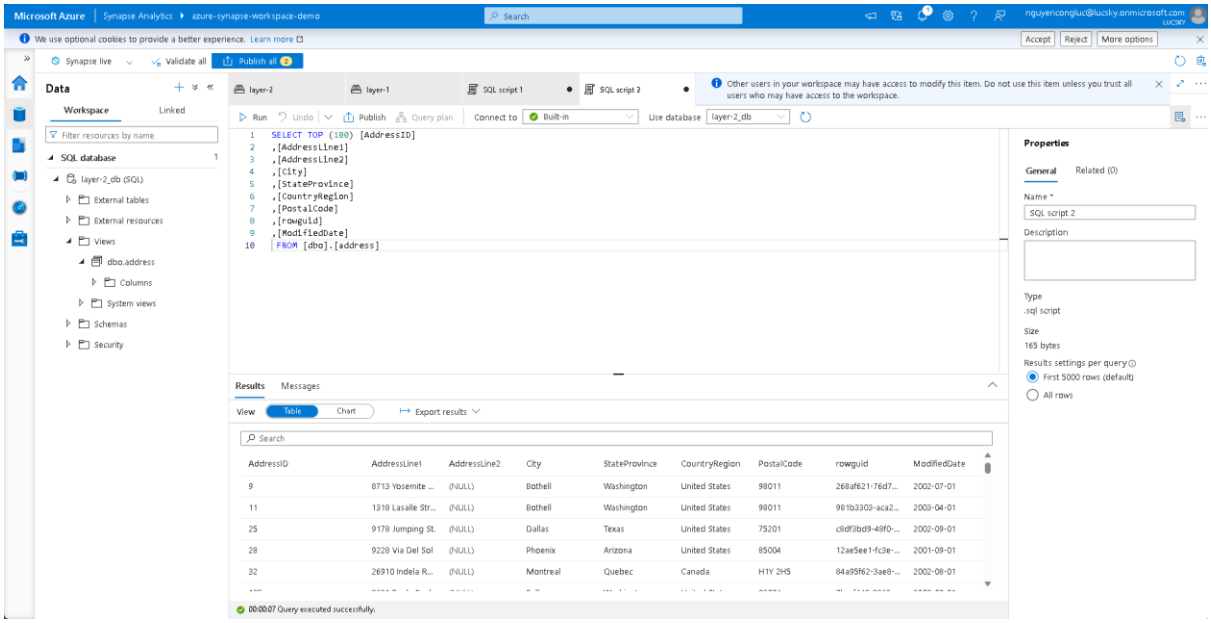


Create a view at temp Database in Synapse Analytics layer-2_db



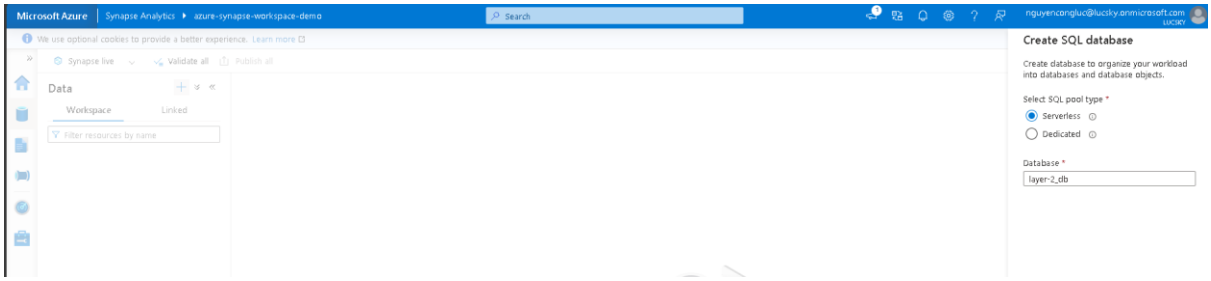


Select top 100 at view

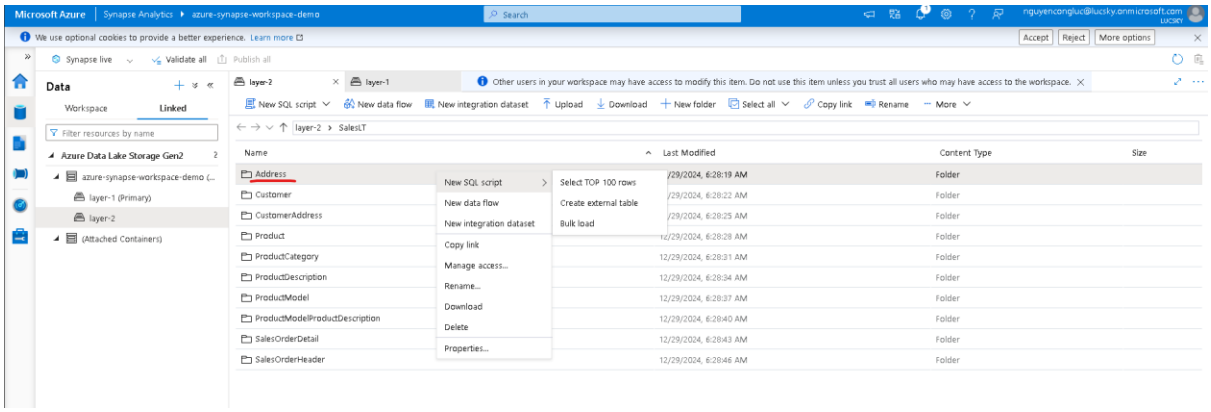


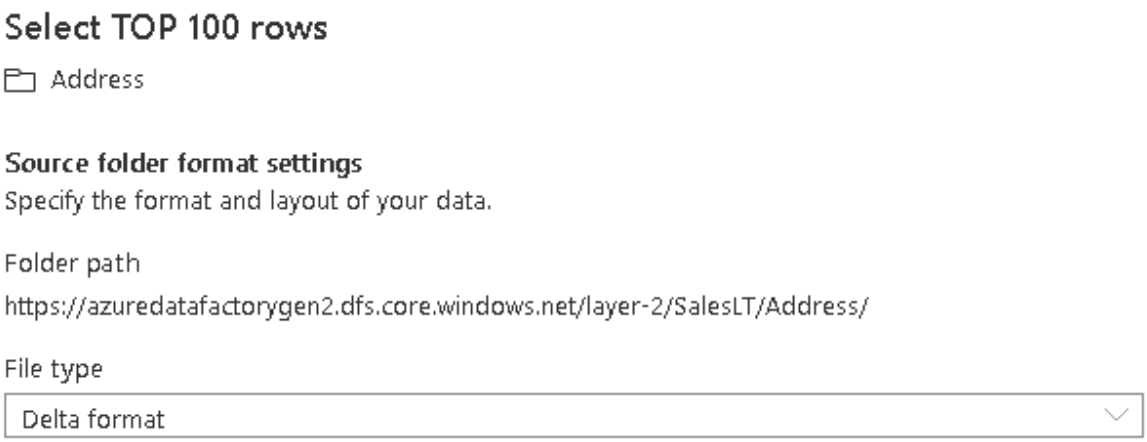
Dedicated SQL pool: Create a real database in Azure Synapse with distributed tables and fixed resource requirements.

Serverless SQL pool: Does not create a real database but only queries data from external sources (like Azure Data Lake Gen 2), does not require fixed resources.

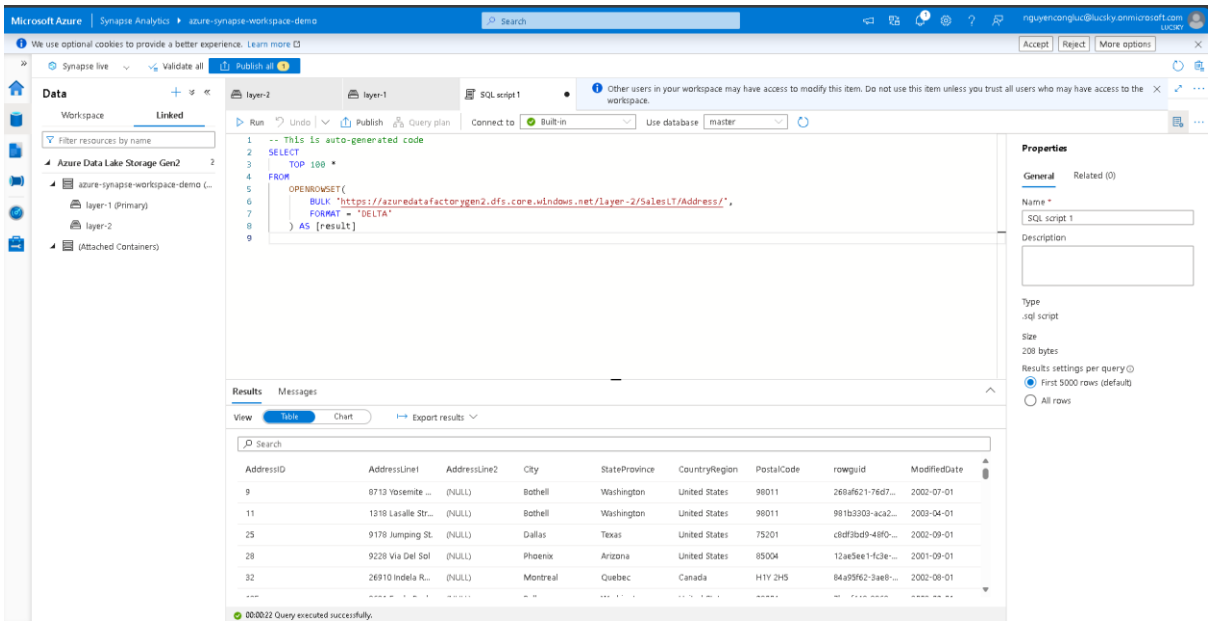


We can query at all table in ADL Gen 2 when linked

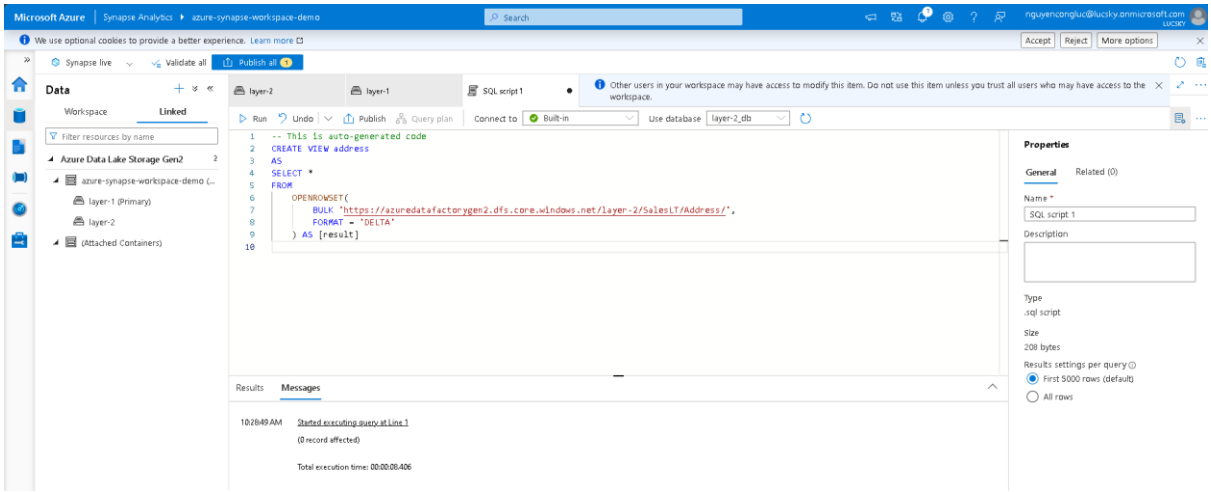




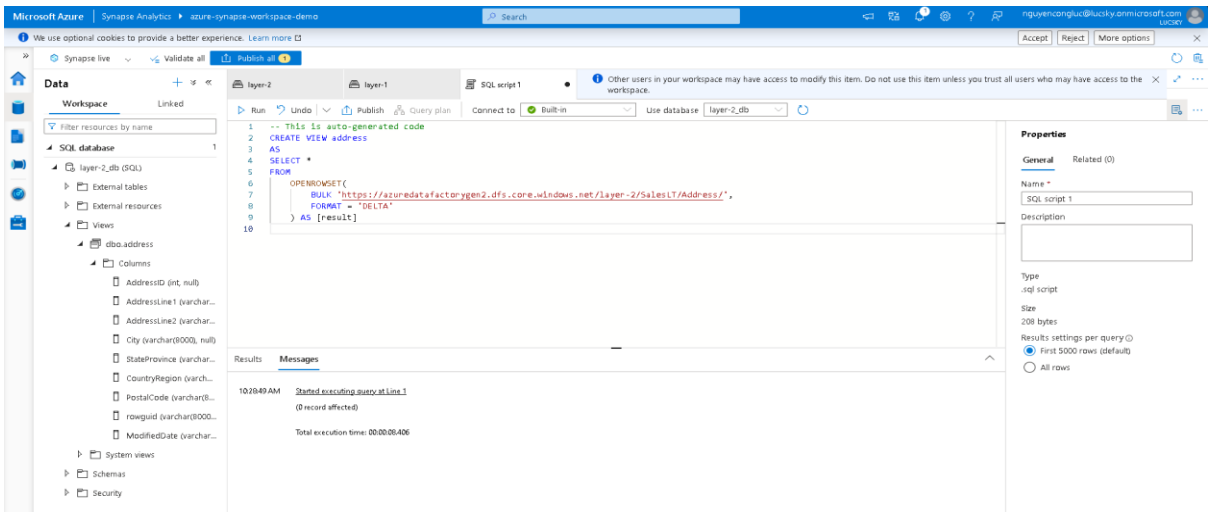
Result of query



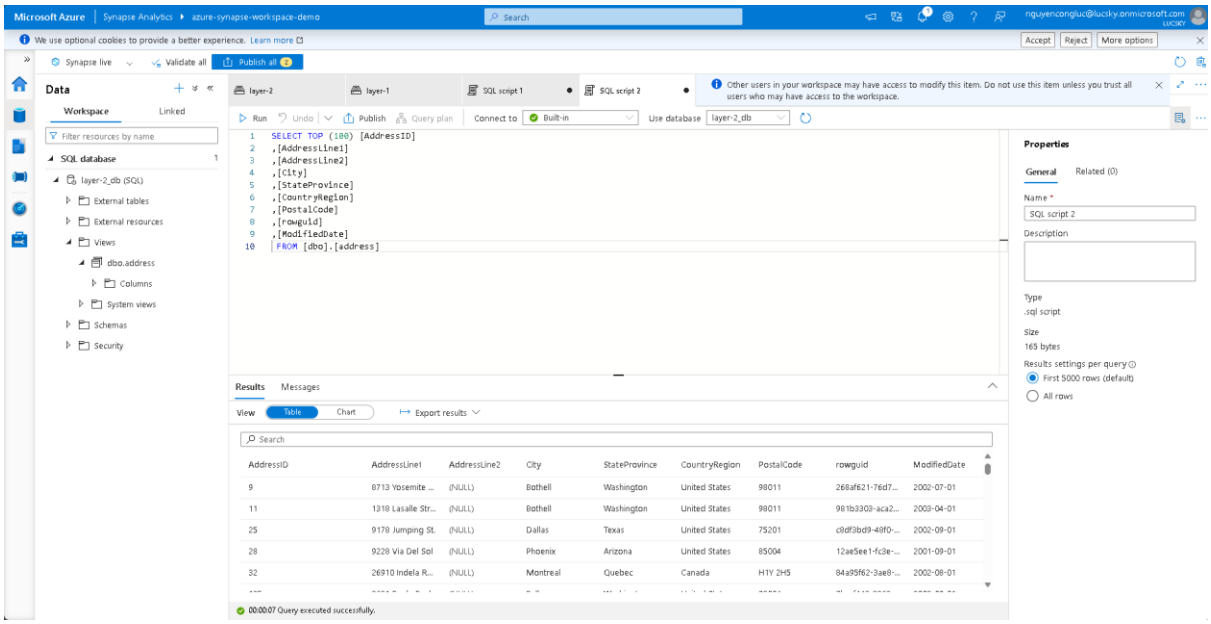
Create a view at temp Database in Synapse Analytics layer-2_db



Views created

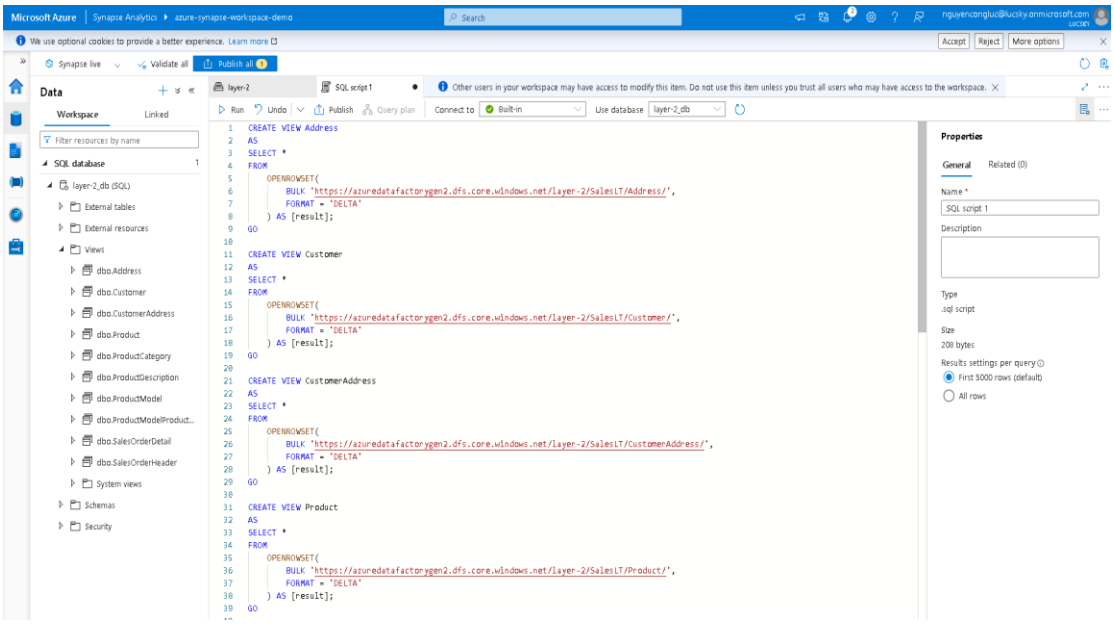


Select top 100 at view to test



Continue create other views from all tables

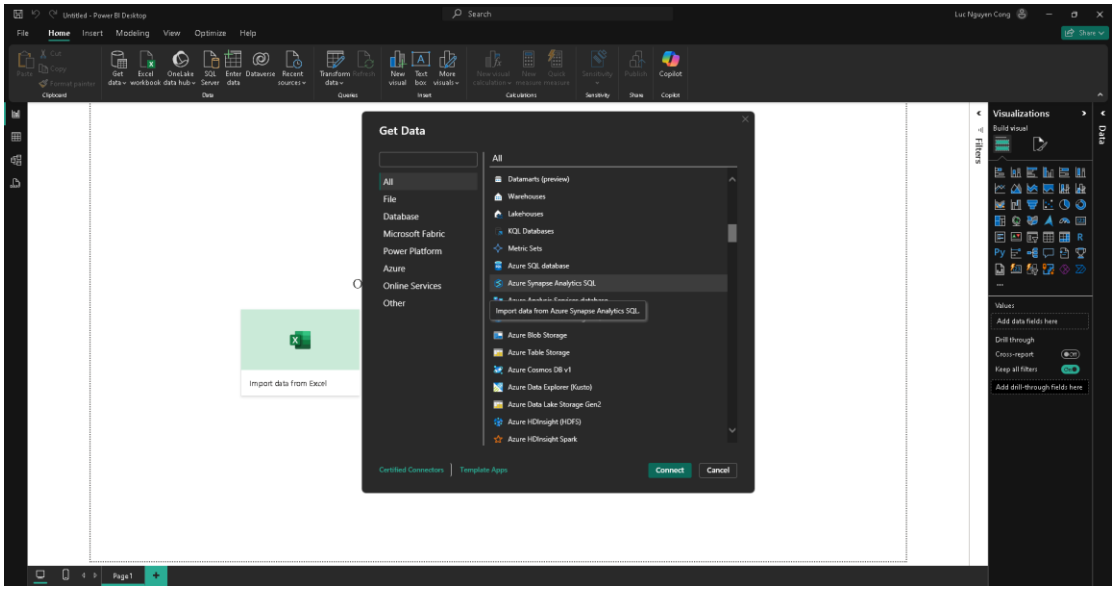
```
CREATE VIEW Address
AS
SELECT *
FROM
    OPENROWSET(
        BULK 'https://azuredatafactorygen2.dfs.core.windows.net/layer-2/SalesLT/Address/',
        FORMAT = 'DELTA'
    ) AS [result];
GO
CREATE VIEW Product
AS
SELECT *
FROM
    OPENROWSET(
        BULK 'https://azuredatafactorygen2.dfs.core.windows.net/layer-2/SalesLT/Product/',
        FORMAT = 'DELTA'
    ) AS [result];
GO
CREATE VIEW ProductCategory
AS
SELECT *
FROM
    OPENROWSET(
        BULK 'https://azuredatafactorygen2.dfs.core.windows.net/layer-2/SalesLT/ProductCategory/',
        FORMAT = 'DELTA'
    ) AS [result];
GO
-- Continue with other views...
```



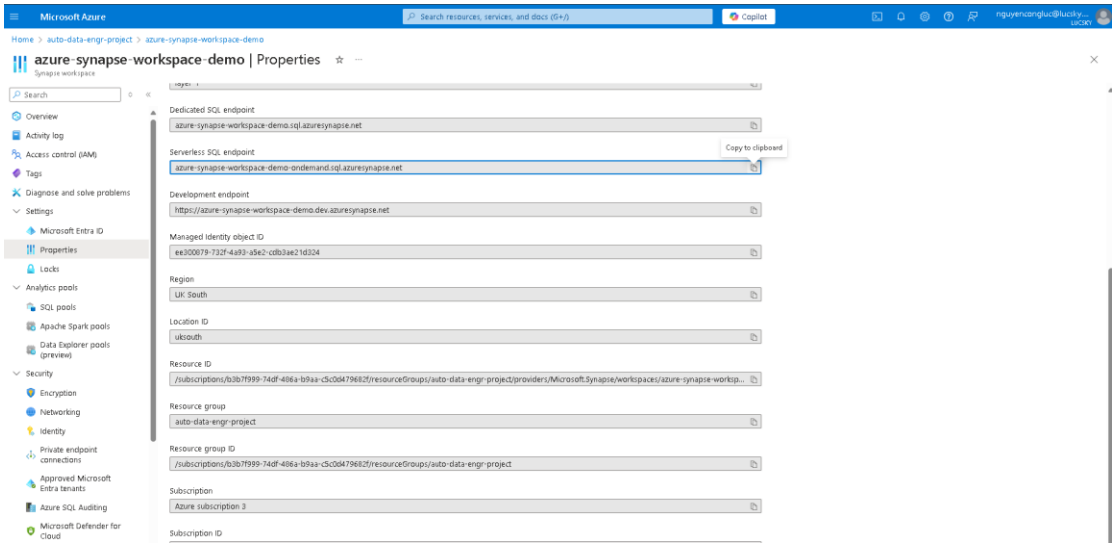
CHAPTER 6. Data Reporting

6.1 Connect data from Azure synapse analyst

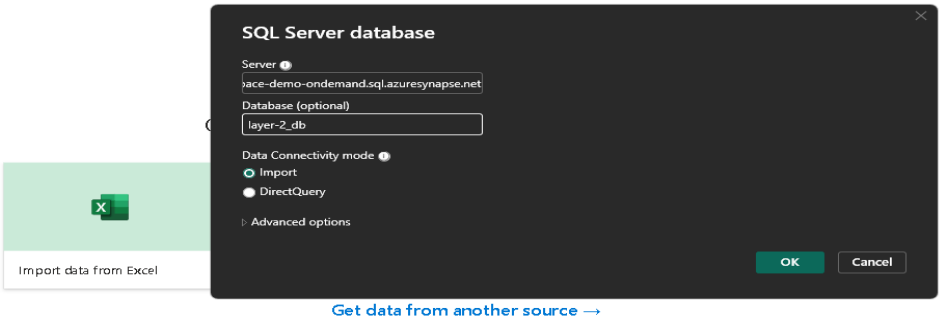
Select resouce



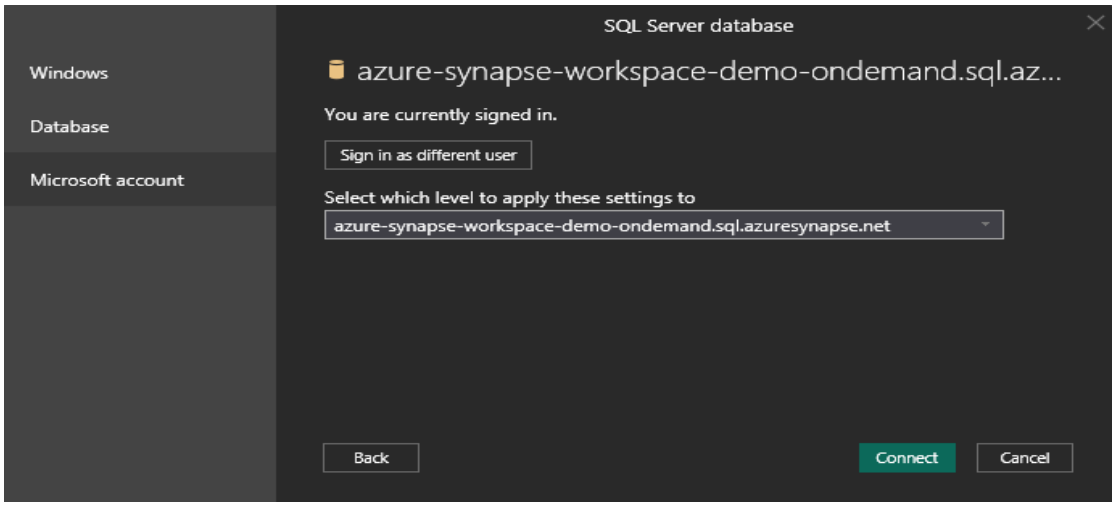
Get server name from Azure synapse analyst



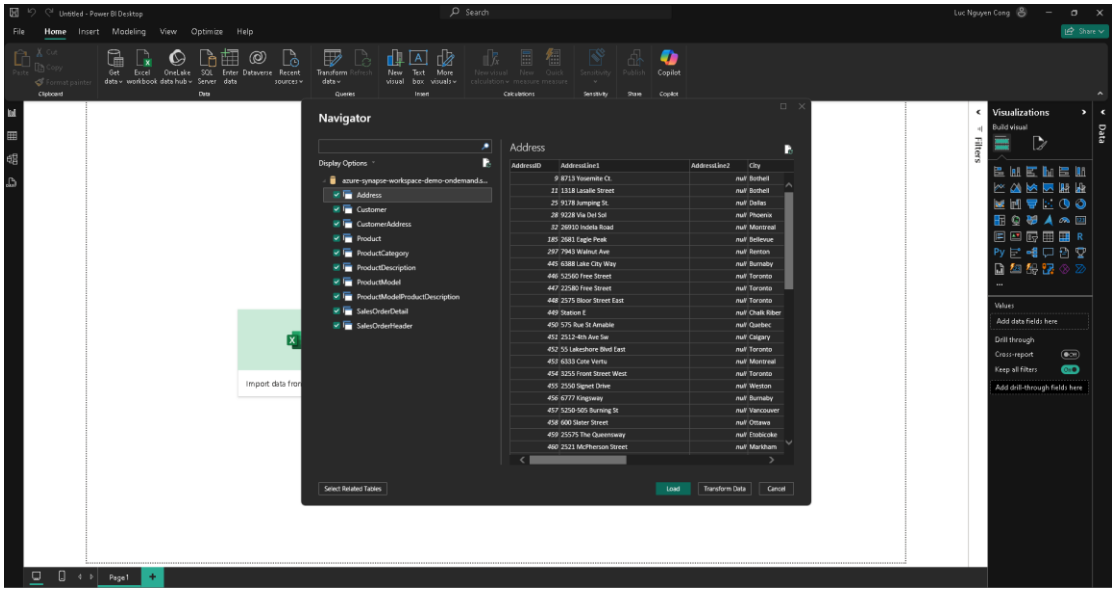
Setup connection



Action



Load data to Power BI successfully



6.2 Design dashboard

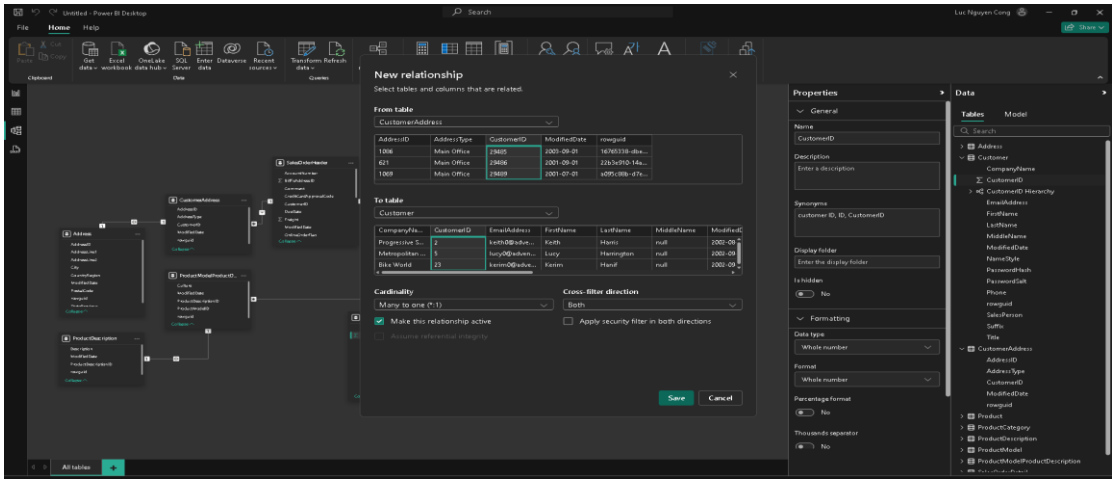
Report View: Used for designing and visualizing reports with charts, tables, and images.

Data View: Displays raw data from tables, allowing users to inspect and work with the data.

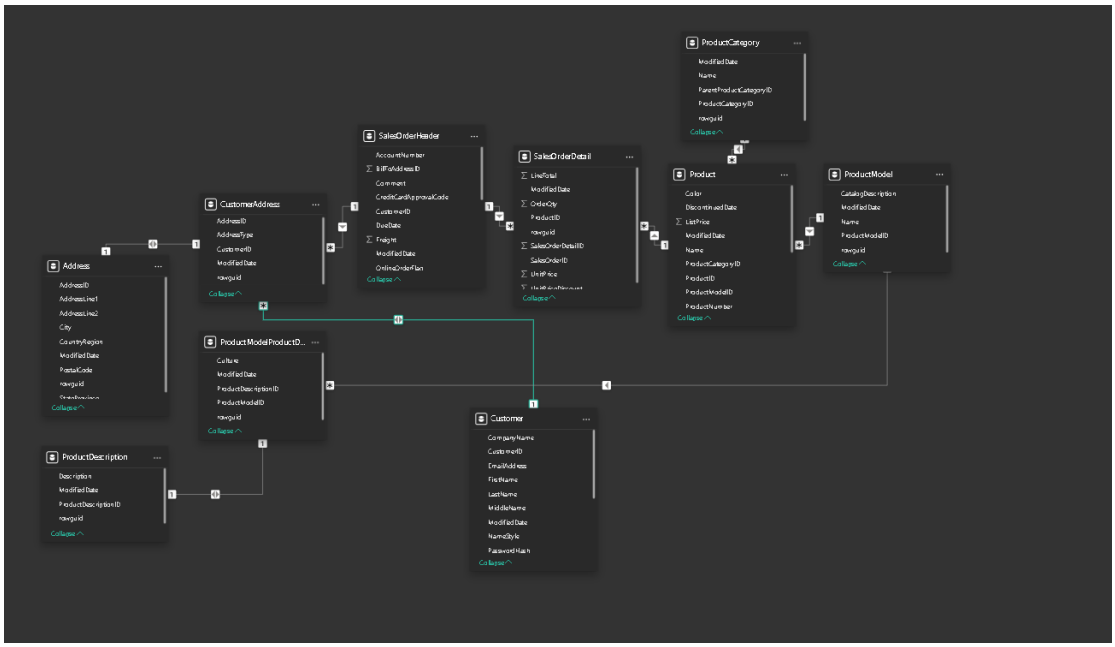
Model View: Manages relationships between tables in the data model.

DAX Query View: Analyzes and optimizes DAX queries, helping to understand and improve report performance.

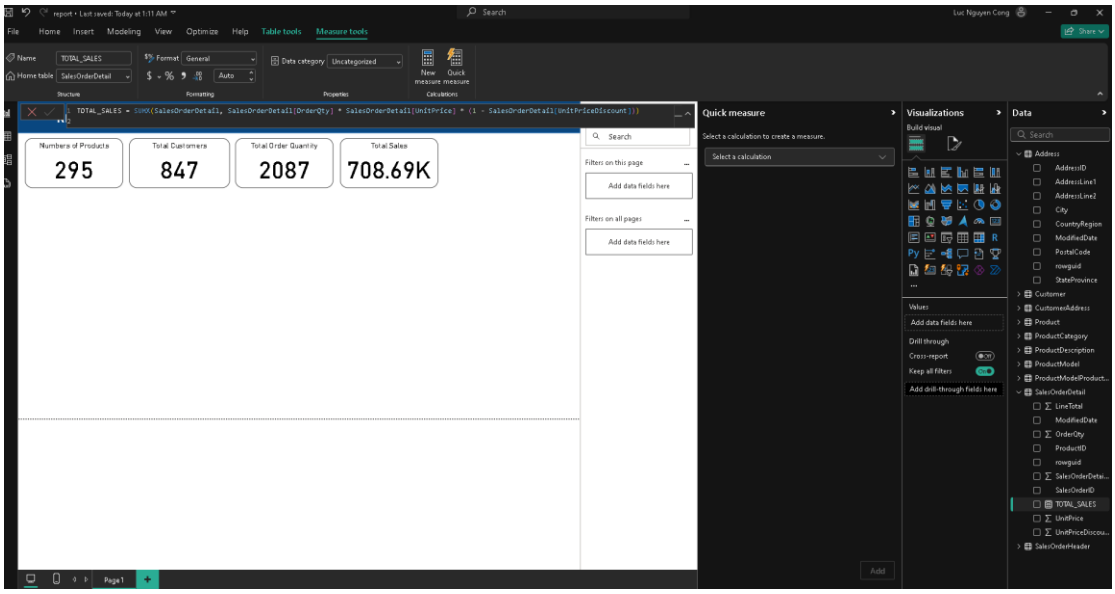
Create relationships for a table CustomerAddress and Customer



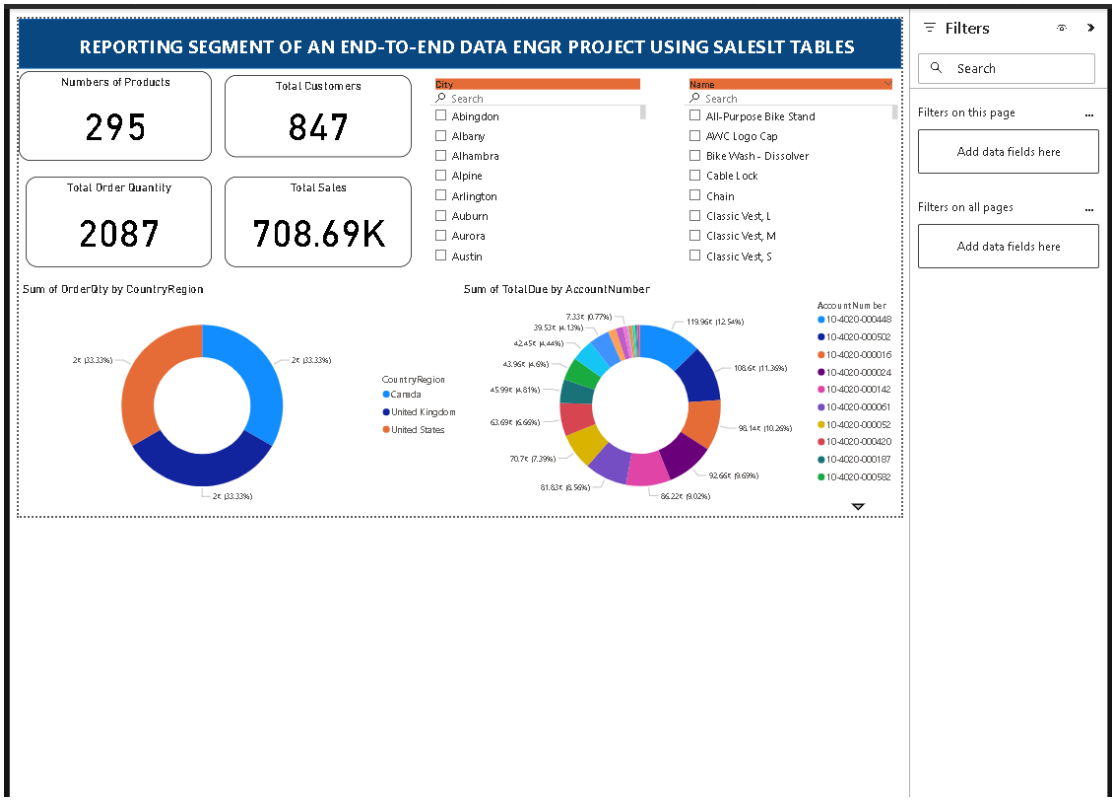
Model View



Design with Card and measure

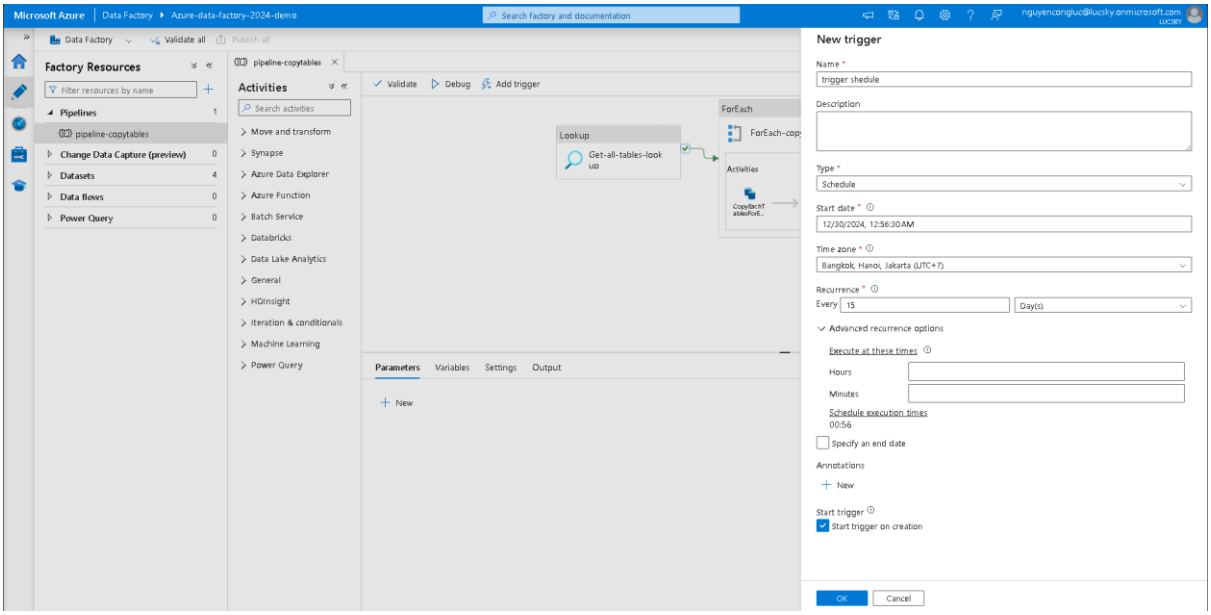


Result



CHAPTER 7. Pipelines Testing/Automation

Create trigger to automation update data. Any changes in data (sql server) are also reflected on Power BI



CHAPTER 8. SUMMARY

In this end-to-end Data Engineering project, I successfully built and tested the entire pipeline, starting from collecting source data, performing ETL (Extract, Transform, Load), and finally Report data via Power BI. The process includes setting up and activating Azure Data Factory, using Azure Data Bricks to transform data, and checking integrity through automatic updates.

The final result demonstrated that the pipeline worked correctly, meeting the requirements for real-time data processing and updating. This is a solid foundation for more complex data engineering projects in the future.

This project not only provides a comprehensive view of the data processing process but also illustrates how to apply modern tools to achieve the highest efficiency.