# BOOTCAMP PROJECT 4

Project Title: Incremental Data Loading and Automated Notifications using Microsoft Fabric

## Problem Statement:

In modern data ecosystems, organizations need to efficiently ingest, transform, and load data from various sources into centralized platforms for analytics, while also ensuring timely monitoring and notification upon successful data refreshes. This project addresses the challenge of incrementally loading data from on-premises sources to Microsoft Fabric Lakehouse, processing it through a structured transformation pipeline, and triggering automated notifications upon successful execution.

## Project Architecture Diagram:

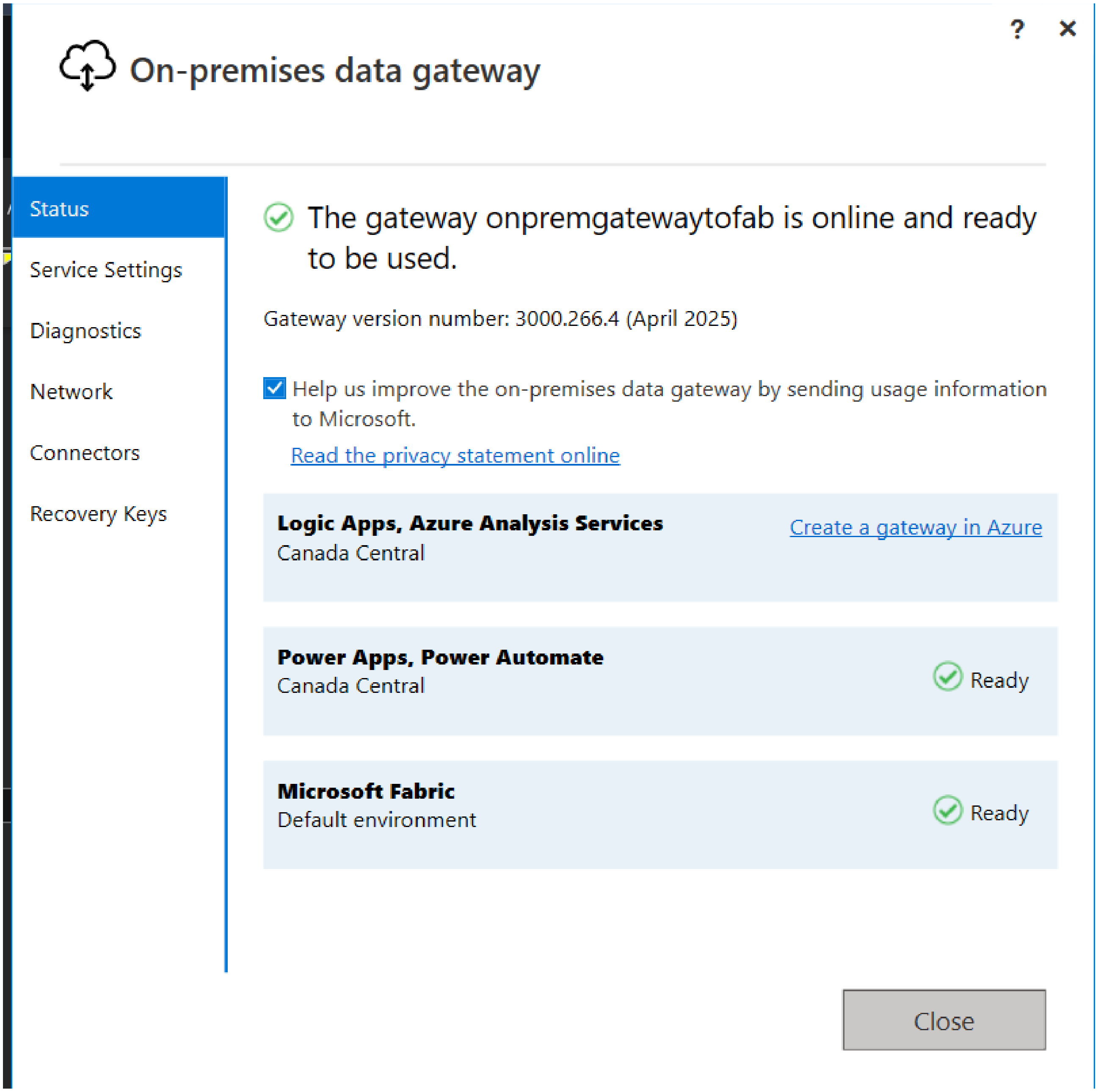
## Tools and Technology used :

- Microsoft Fabric
- On-Premises Data Gateway
- Fabric Lakehouse and Warehouse
- Fabric Dataflow Gen 1
- Fabric Notebook
- Email Notification Task (in-built)
- Draw.io / Visio for architecture diagram

# PROJECT FLOW

1. DATA INGESTION FROM LOCAL MACHINE TO FABRIC LAKEHOUSE USING ON-PREMISES GATEWAY

- In this phase, data gets transferred on local windows OS machine to fabric Lakehouse using On-premises gateway. Here are the steps I followed:



On-premises data gateway

✓ The gateway onpremgatewaytofab is online and ready to be used.

Gateway version number: 3000.266.4 (April 2025)

☑ Help us improve the on-premises data gateway by sending usage information to Microsoft.

Read the privacy statement online

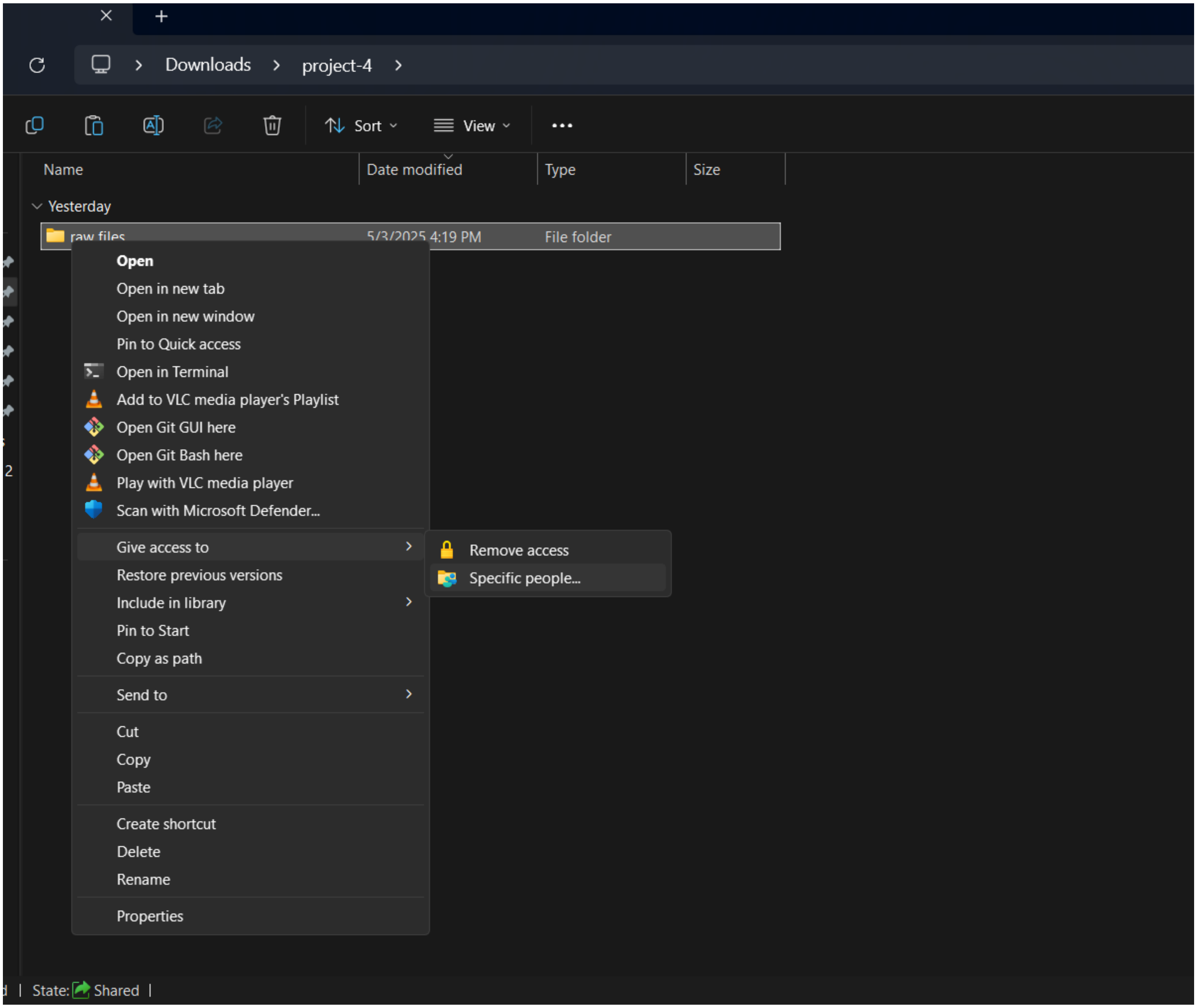
**Logic Apps, Azure Analysis Services**
Canada Central
Create a gateway in Azure

**Power Apps, Power Automate**
Canada Central
✓ Ready

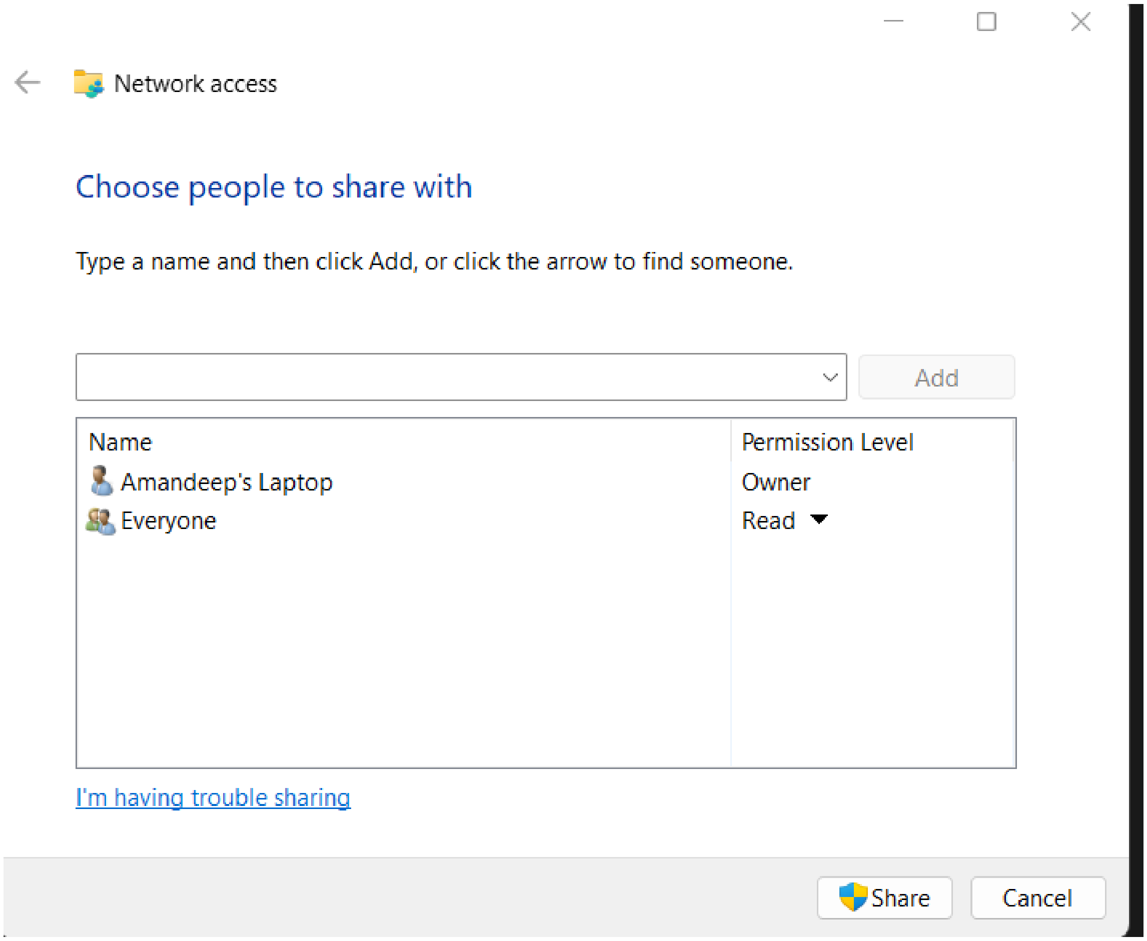
**Microsoft Fabric**
Default environment
✓ Ready

Install On-prem gateway and login with same credentials used in fabric. Setup up recouvery key in case of emergency.

Project-4  by Amandeep Singh



Go to the dir, which you want to share and give read only access to everyone



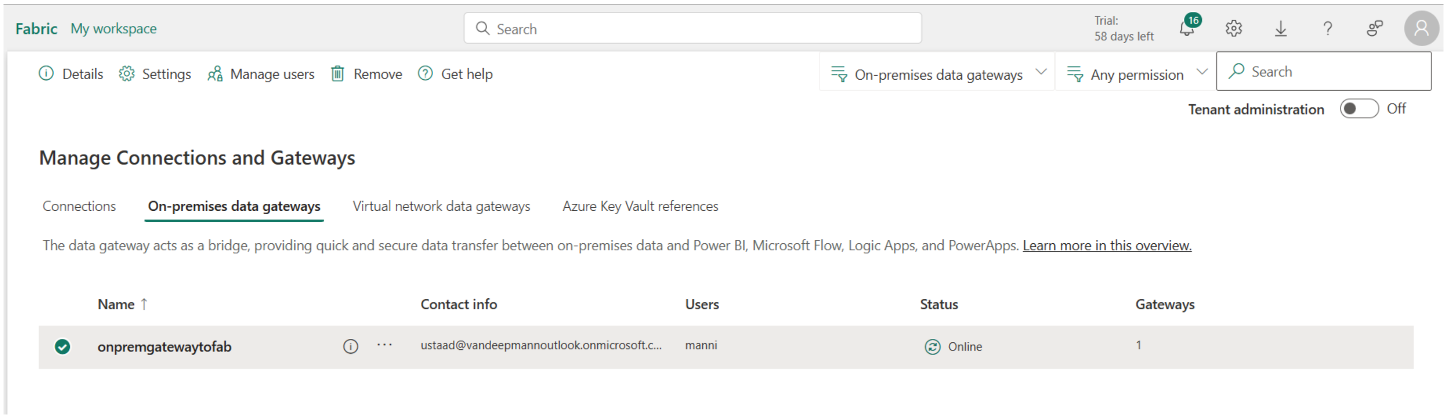
Choose people to share with

Type a name and then click Add, or click the arrow to find someone.

| Name | Permission Level |
|---|---|
| Amandeep's Laptop | Owner |
| Everyone | Read |

Now check it gateway status is online in the fabric.

Go to fabric>settings>manage connections and gateways>> gateways and check if it is online.



Manage Connections and Gateways

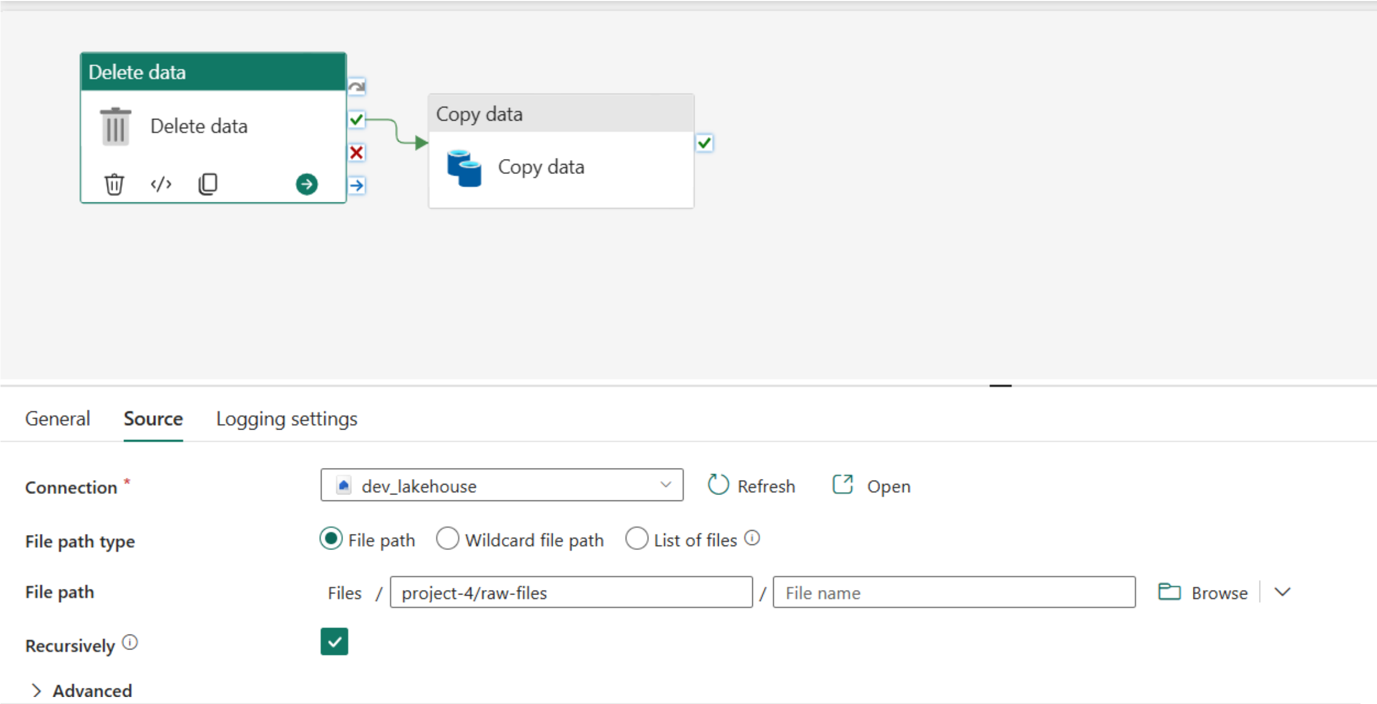
The data gateway acts as a bridge, providing quick and secure data transfer between on-premises data and Power BI, Microsoft Flow, Logic Apps, and PowerApps. Learn more in this overview.

| Name ↑ | Contact info | Users | Status | Gateways |
|---|---|---|---|---|
| onpremgatewaytofab | ustaad@vandeepmannoutlook.onmicrosoft.c... | manni | Online | 1 |

## Create new Data Pipeline for data ingestion

- Create delete activity and copy activity in the pipeline.
- Delete activity is to delete the previous csv files already present in lakehouse directory.

- In the copy activity, in source section, I have selected “folder” data source.
- In the destination section, I have used lakehouse as sink.



General | **Source** | Logging settings

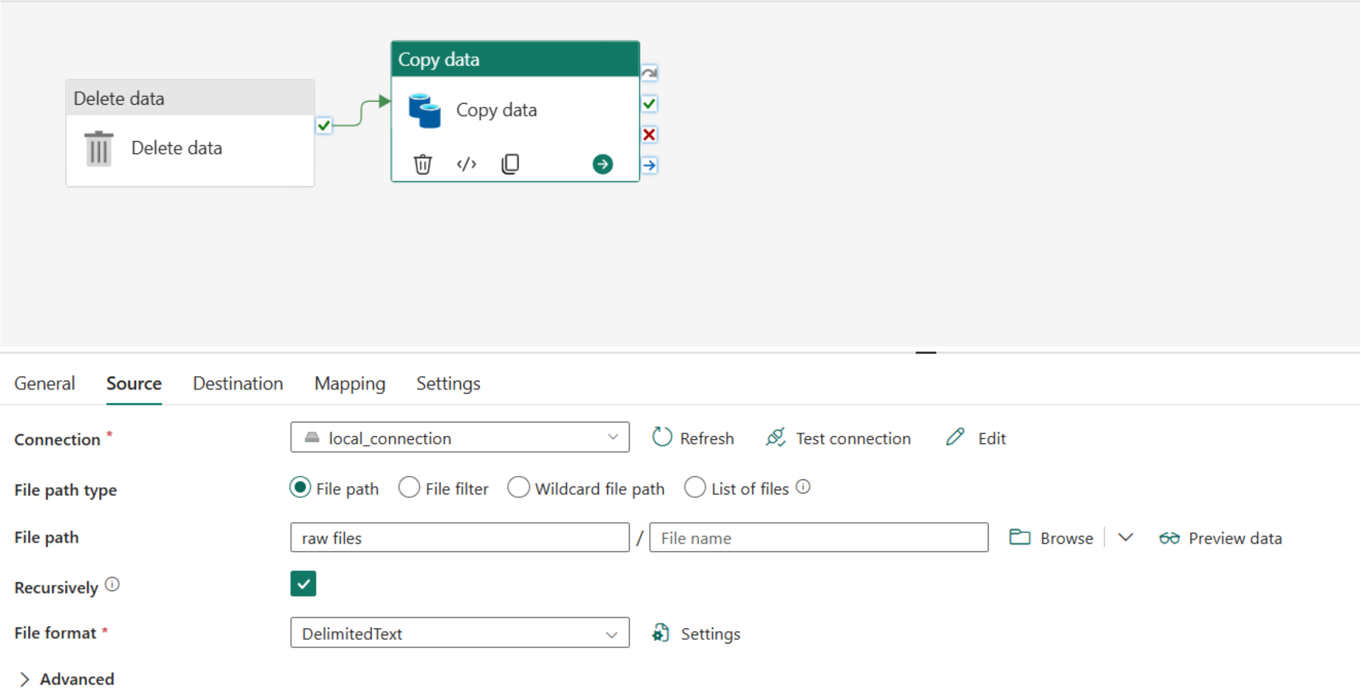
Connection * : dev_lakehouse ⌄ — Refresh — Open

File path type: ● File path ○ Wildcard file path ○ List of files ⓘ

File path: Files / project-4/raw-files / File name — Browse ⌄

Recursively ⓘ: ☑

› Advanced



General | **Source** | Destination | Mapping | Settings

Connection * : local_connection ⌄ — Refresh — Test connection — Edit

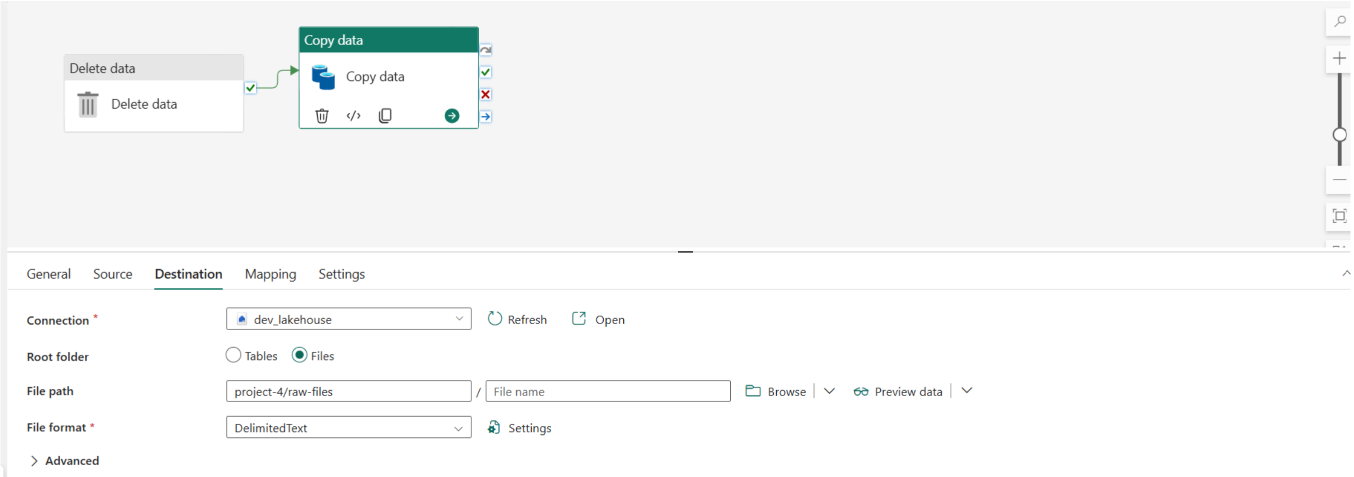
File path type: ● File path ○ File filter ○ Wildcard file path ○ List of files ⓘ

File path: raw files / File name — Browse ⌄ — Preview data

Recursively ⓘ: ☑

File format * : DelimitedText ⌄ — Settings

› Advanced



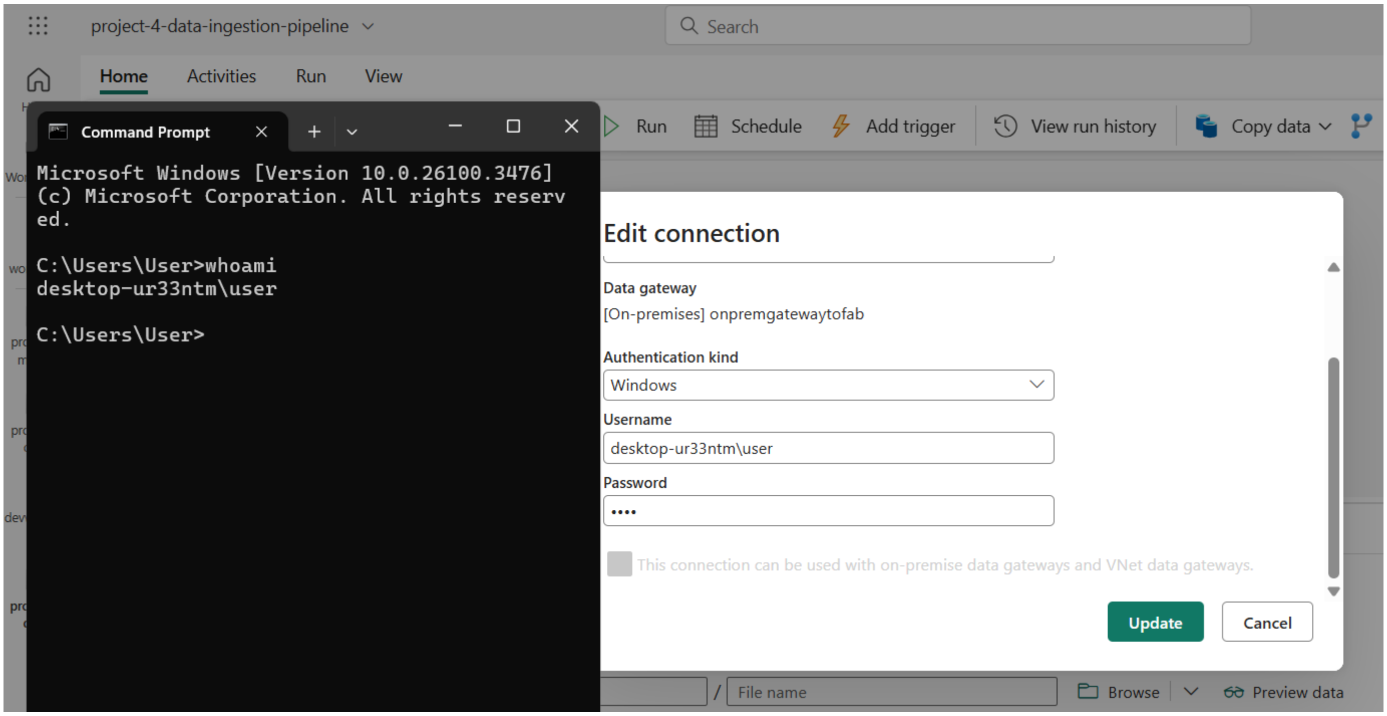
Connection * : dev_lakehouse

Root folder: ○ Tables ● Files

File path: project-4/raw-files / File name

File format * : DelimitedText

- In source section, I’ve used same gateway to connect with local PC.



```
Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserv
ed.

C:\Users\User>whoami
desktop-ur33ntm\user

C:\Users\User>
```

Edit connection

Data gateway
[On-premises] onpremgatewaytofab

Authentication kind
Windows

Username
desktop-ur33ntm\user

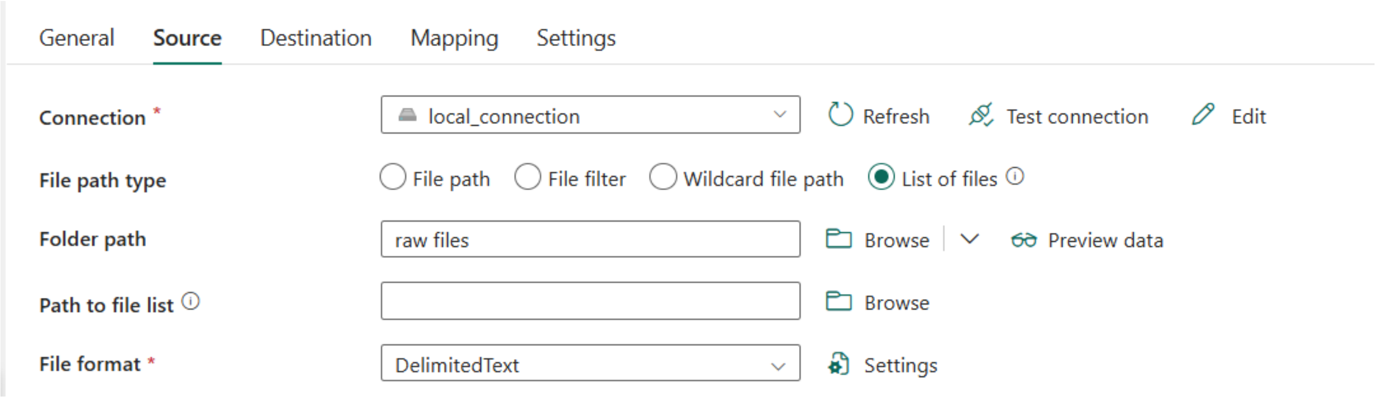
Password
••••

□ This connection can be used with on-premise data gateways and VNet data gateways.

Update   Cancel

Over here, we have to use windows credentials and Administrator’s account name.

- Select the list of files options to copy all csv files from  a dir



General | **Source** | Destination | Mapping | Settings

Connection * : local_connection — Refresh — Test connection — Edit

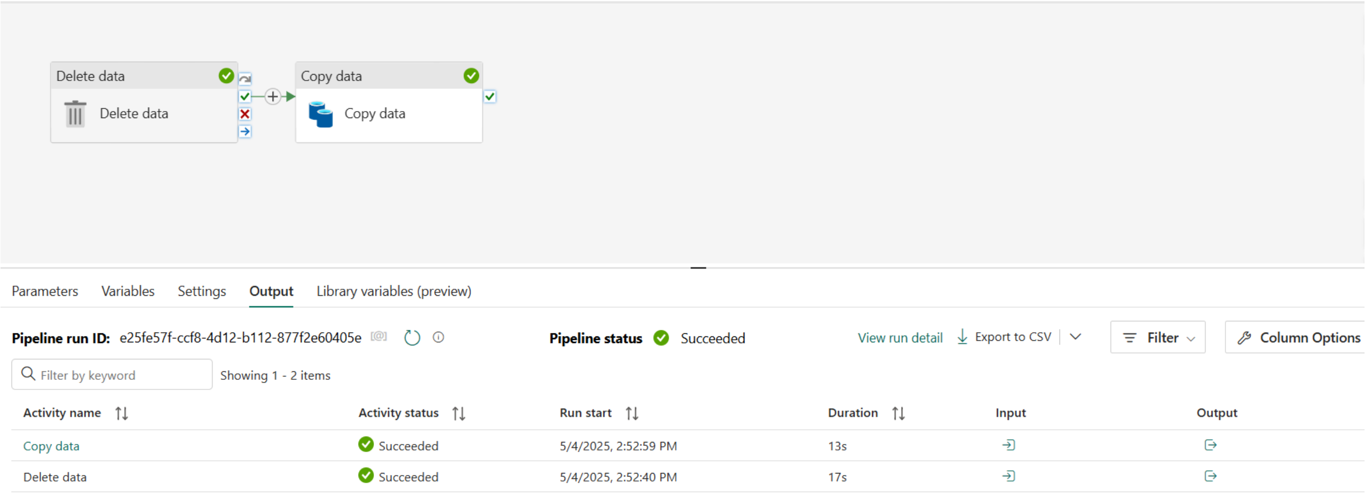
File path type: ○ File path ○ File filter ○ Wildcard file path ● List of files ⓘ

Folder path: raw files — Browse — Preview data

Path to file list ⓘ: — Browse

File format * : DelimitedText — Settings

Here is the completion status of data ingestion pipeline:



Parameters | Variables | Settings | **Output** | Library variables (preview)

**Pipeline run ID:** e25fe57f-ccf8-4d12-b112-877f2e60405e

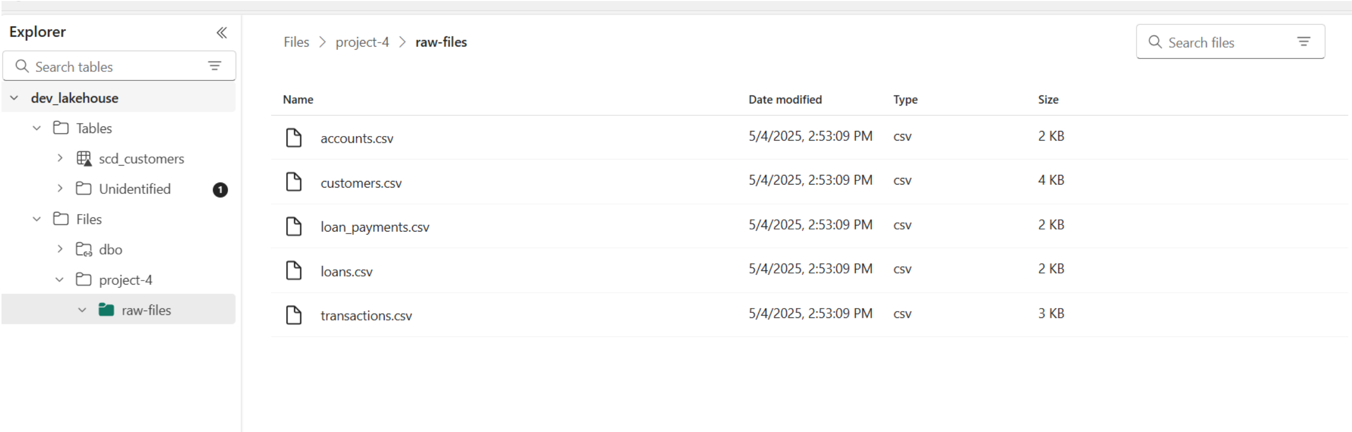
**Pipeline status** ✓ Succeeded

View run detail | Export to CSV | Filter | Column Options

Filter by keyword — Showing 1 - 2 items

| Activity name | Activity status | Run start | Duration | Input | Output |
|---|---|---|---|---|---|
| Copy data | ✓ Succeeded | 5/4/2025, 2:52:59 PM | 13s | | |
| Delete data | ✓ Succeeded | 5/4/2025, 2:52:40 PM | 17s | | |

The data get copied from LOCAL Machine to Fabric Lakehouse:



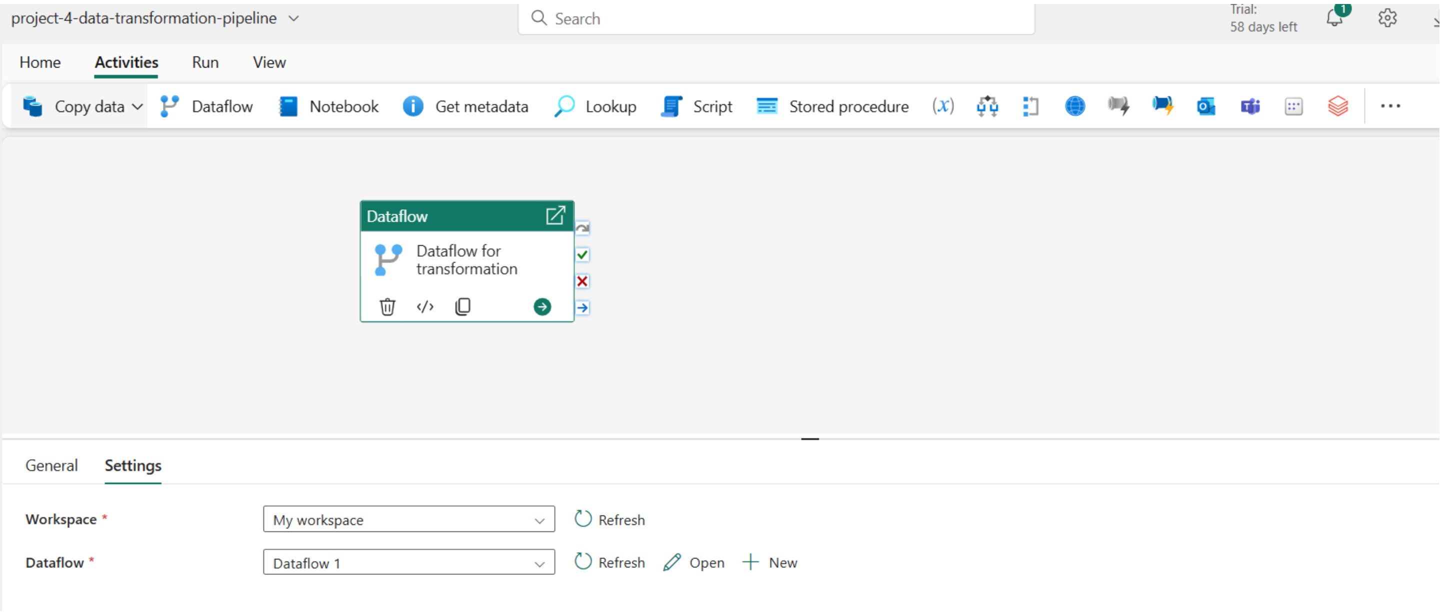
Files > project-4 > raw-files

Search files

| Name | Date modified | Type | Size |
|---|---|---|---|
| accounts.csv | 5/4/2025, 2:53:09 PM | csv | 2 KB |
| customers.csv | 5/4/2025, 2:53:09 PM | csv | 4 KB |
| loan_payments.csv | 5/4/2025, 2:53:09 PM | csv | 2 KB |
| loans.csv | 5/4/2025, 2:53:09 PM | csv | 2 KB |
| transactions.csv | 5/4/2025, 2:53:09 PM | csv | 3 KB |

## 2. DATA CLEANING AND DATA TRANSFORMATION USING DATAFLOW GEN1 AND STORING IN FABRIC DATA WAREHOUSE

- Now, data got ingested into fabric lakehouse in raw files ( csv files).
- It’s time to clean the data and transforming it according to requirements.
- DATAFLOW gen1 will be used to clean and transform the data and all csv files data will be saved into Tables in fabric- data warehouse.



Workspace * My workspace

Dataflow * Dataflow 1

Here the data pipleine is created where dataflow is invoked, which helps in data cleaning and data transformation.

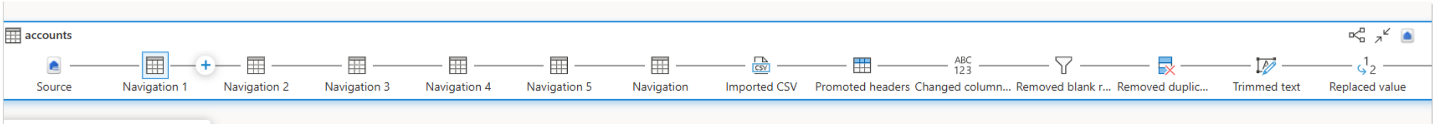
In the dataflow, I’m doing following tasks:

- Handle Missing Values
- Removing duplicate records
- Filtering the records
- Transforms date data type
- Trimming string column values

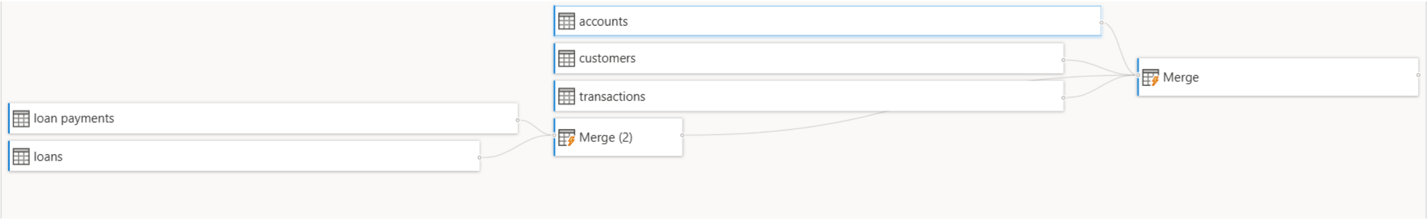
For each table. So there will 5 data sources and 5 sinks.

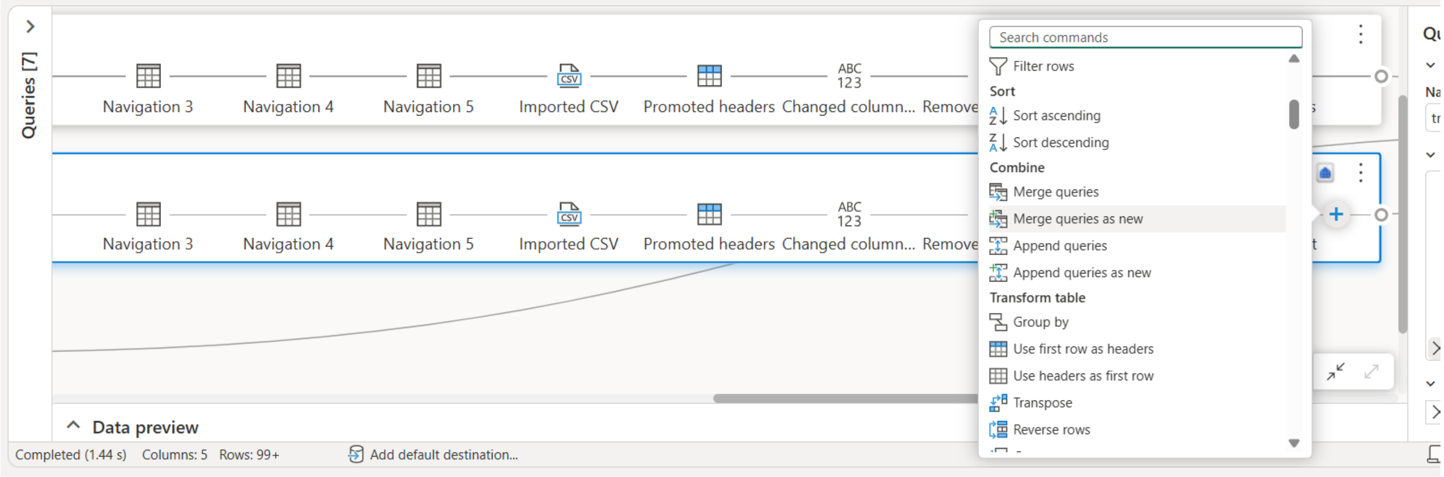
In addition to that, I'm creating new table, in which data from 5 tables get combined into single table named : "cleaned_common_table"

Here is the query view of 1 csv file :



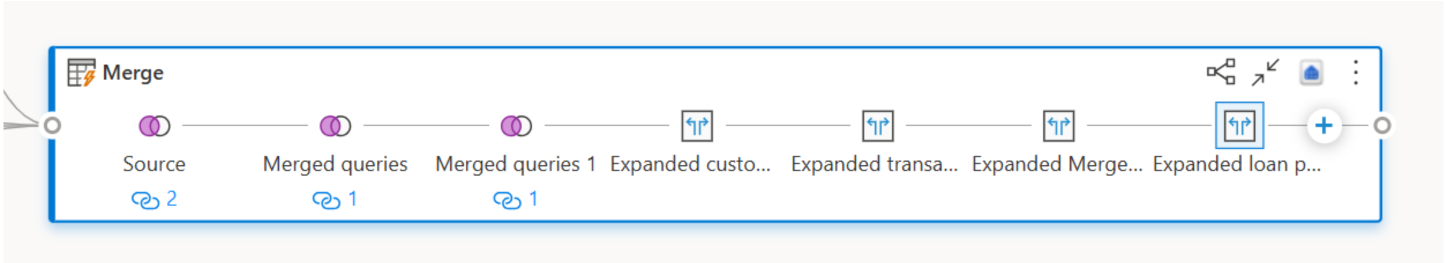
Here is the diagram view of those dataflow 1 :





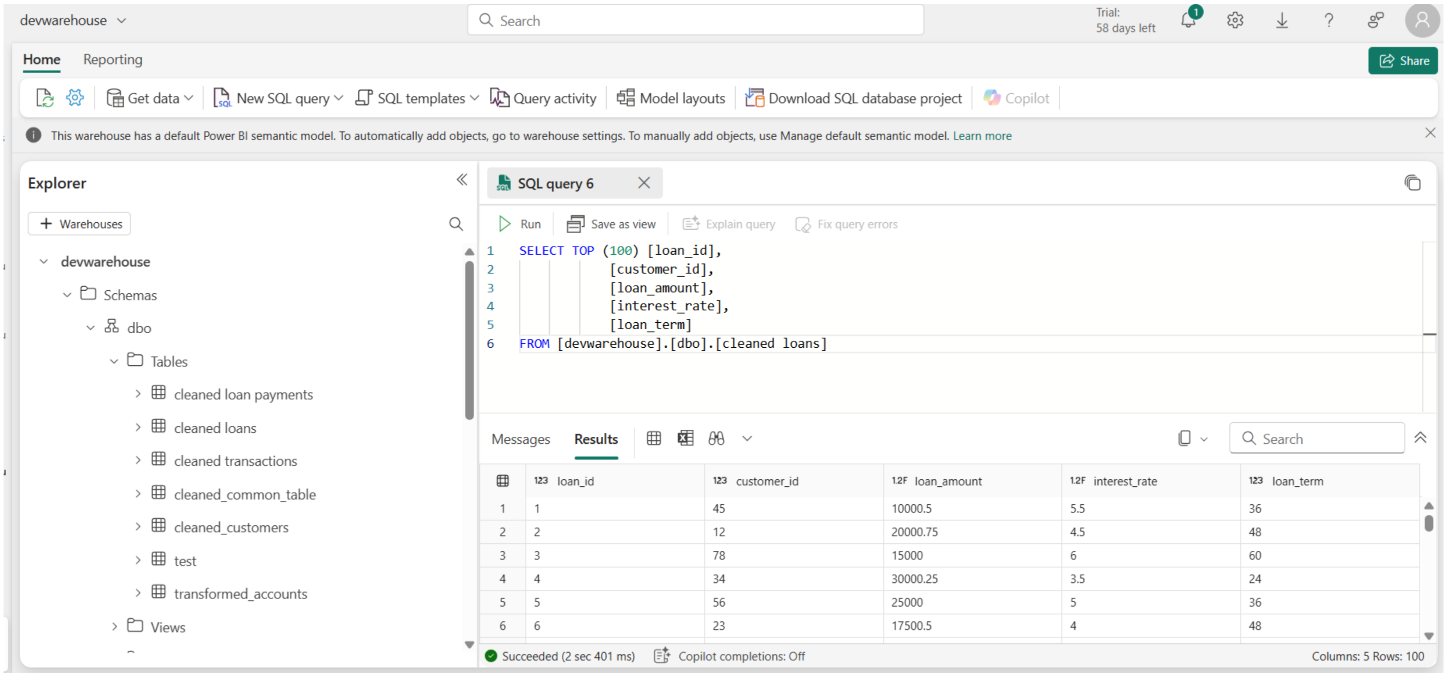
I'm using merge queries as new and merge queries activity to join two data stream into new stream.

In last, I've created destination table for that common stream too.



I'm using choose column activity and table expansion activity to explode the data into common stream.

Here the resulted data from dataflow 1 get stored as tables in devwarehouse:



This warehouse has a default Power BI semantic model. To automatically add objects, go to warehouse settings. To manually add objects, use Manage default semantic model. Learn more

- devwarehouse
  - Schemas
    - dbo
      - Tables
        - cleaned loan payments
        - cleaned loans
        - cleaned transactions
        - cleaned_common_table
        - cleaned_customers
        - test
        - transformed_accounts
      - Views

```
SELECT TOP (100) [loan_id],
            [customer_id],
            [loan_amount],
            [interest_rate],
            [loan_term]
FROM [devwarehouse].[dbo].[cleaned loans]
```

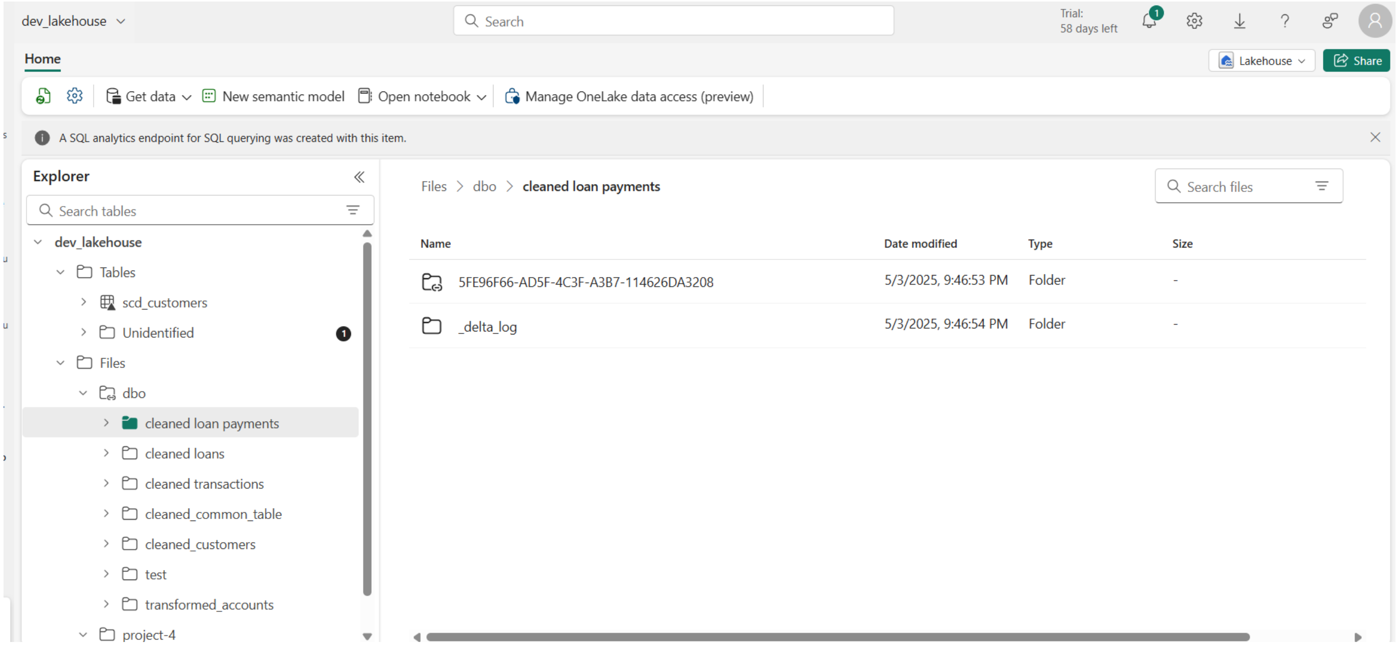
| | loan_id | customer_id | loan_amount | interest_rate | loan_term |
|---|---|---|---|---|---|
| 1 | 1 | 45 | 10000.5 | 5.5 | 36 |
| 2 | 2 | 12 | 20000.75 | 4.5 | 48 |
| 3 | 3 | 78 | 15000 | 6 | 60 |
| 4 | 4 | 34 | 30000.25 | 3.5 | 24 |
| 5 | 5 | 56 | 25000 | 5 | 36 |
| 6 | 6 | 23 | 17500.5 | 4 | 48 |

Succeeded (2 sec 401 ms)　Copilot completions: Off　Columns: 5 Rows: 100

## 3. TRANSFORMING THE DATA INTO SCD TYPE 1 TABLES:

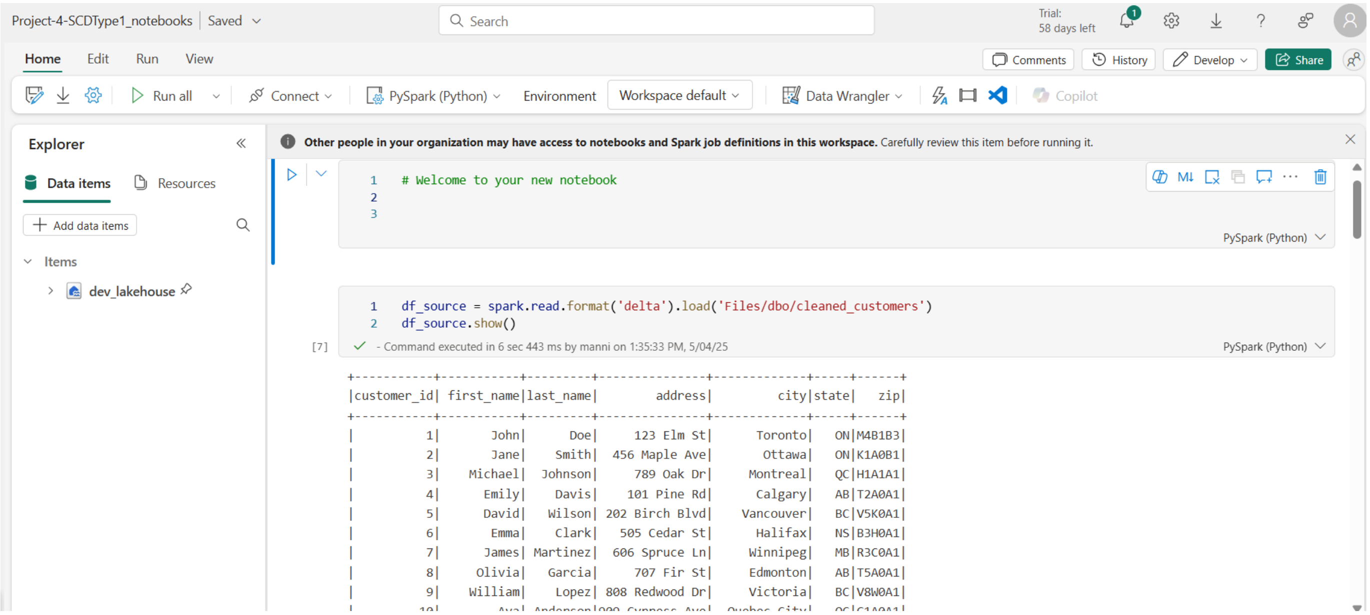
- For this, we are using jupyter notebooks in fabric.
- To access the tables from fabric warehouse, I'm using shortcut of lakehouse, which I've created in the lakehouse.

Project-4  by Amandeep Singh

- 

| Name | Date modified | Type | Size |
|---|---|---|---|
| 5FE96F66-AD5F-4C3F-A3B7-114626DA3208 | 5/3/2025, 9:46:53 PM | Folder | - |
| _delta_log | 5/3/2025, 9:46:54 PM | Folder | - |

- This is the notebook, where I was accessing warehouse’s tables using shortcut.



Other people in your organization may have access to notebooks and Spark job definitions in this workspace. Carefully review this item before running it.

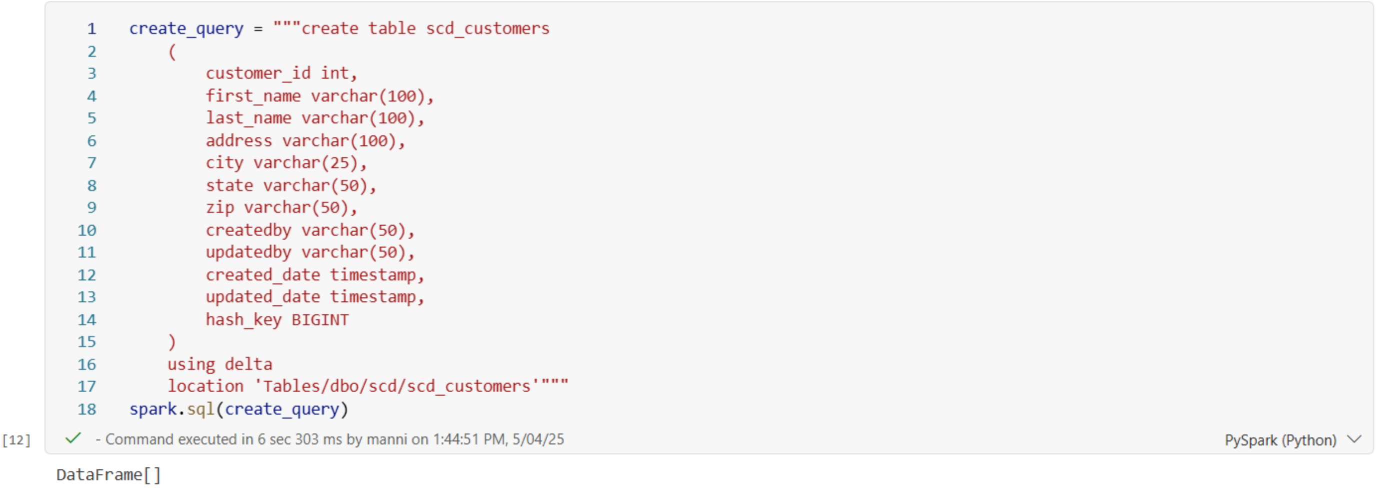
```
# Welcome to your new notebook


```

```
df_source = spark.read.format('delta').load('Files/dbo/cleaned_customers')
df_source.show()
```

[7] ✓ - Command executed in 6 sec 443 ms by manni on 1:35:33 PM, 5/04/25

```
+-----------+----------+---------+---------------+-------------+-----+------+
|customer_id| first_name|last_name|        address|         city|state|   zip|
+-----------+----------+---------+---------------+-------------+-----+------+
|          1|      John|      Doe|     123 Elm St|      Toronto|   ON|M4B1B3|
|          2|      Jane|    Smith|  456 Maple Ave|       Ottawa|   ON|K1A0B1|
|          3|   Michael|  Johnson|     789 Oak Dr|     Montreal|   QC|H1A1A1|
|          4|     Emily|    Davis|    101 Pine Rd|      Calgary|   AB|T2A0A1|
|          5|     David|   Wilson| 202 Birch Blvd|    Vancouver|   BC|V5K0A1|
|          6|      Emma|    Clark|   505 Cedar St|      Halifax|   NS|B3H0A1|
|          7|     James| Martinez|  606 Spruce Ln|     Winnipeg|   MB|R3C0A1|
|          8|    Olivia|   Garcia|     707 Fir St|     Edmonton|   AB|T5A0A1|
|          9|   William|    Lopez| 808 Redwood Dr|     Victoria|   BC|V8W0A1|
|         10|       Ava| Anderson|909 Cypress Ave|  Quebec City|   QC|G1A0A1|
```

Then, I’ve created SCD tables for each transformed table.



```
create_query = """create table scd_customers
    (
        customer_id int,
        first_name varchar(100),
        last_name varchar(100),
        address varchar(100),
        city varchar(25),
        state varchar(50),
        zip varchar(50),
        createdby varchar(50),
        updatedby varchar(50),
        created_date timestamp,
        updated_date timestamp,
        hash_key BIGINT
    )
    using delta
    location 'Tables/dbo/scd/scd_customers'"""
spark.sql(create_query)
```

[12] ✓ - Command executed in 6 sec 303 ms by manni on 1:44:51 PM, 5/04/25

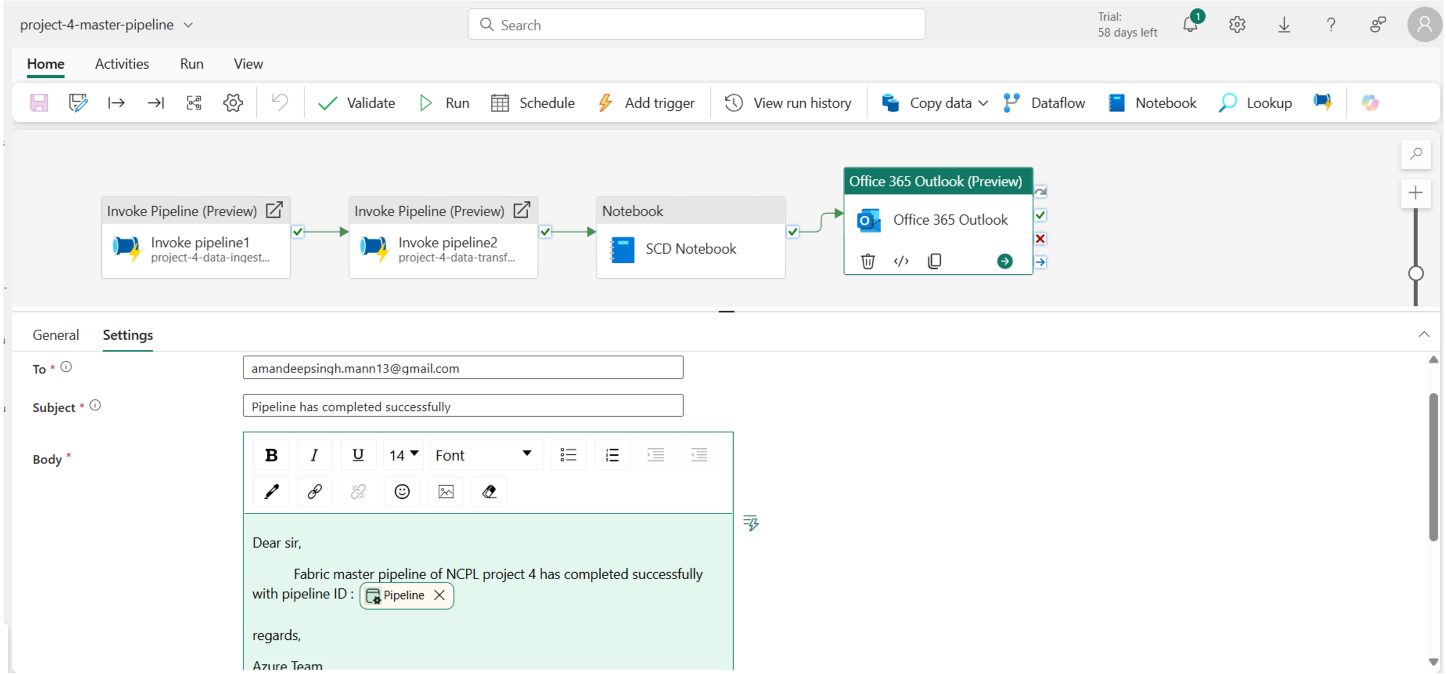
```
DataFrame[]
```

Following is the code to transform dataframe object into SCD Type 1 and storing into target table:

```
from delta.tables import DeltaTable

target_path = "Tables/dbo/scd/scd_customers"

delta_target = DeltaTable.forPath(spark, target_path)

 from pyspark.sql.functions import *

df_src1= df_source.withColumn("hash_key",crc32(concat(df_source.columns))) display(df_src1)
df_src1=df_src1.alias("src").join(delta_target.toDF().alias("tgt"),((col("src.customer_id")==col
("tgt.customer_id"))&(col("src.hash_key")==col("tgt.hash_key"))),"anti").select(col("src.")) df_src1.show()
from pyspark.sql.functions import col
delta_target.alias("tgt").merge(df_src1.alias("src"),"tgt.customer_id =
src.customer_id").whenMatchedUpdate(set={"tgt.customer_id":"src.customer_id","tgt.first_name":"src.
first_name","tgt.last_name":"src.last_name","tgt.address":"src.address","tgt.city":"src.city","tgt.state":"
src.state","tgt.zip":"src.zip","tgt.hash_key":"src.hash_key","tgt.updated_date":current_timestamp(),"tgt.
updatedby":lit("databricks_Updated") }).whenNotMatchedInsert(values={"tgt.customer_id":"src.custom
er_id","tgt.first_name":"src.first_name","tgt.last_name":"src.last_name","tgt.address":"src.address","tgt
.city":"src.city","tgt.state":"src.state","tgt.zip":"src.zip","tgt.hash_key":"src.hash_key","tgt.created_date"
:current_timestamp(),"tgt.createdby":lit("databricks"),"tgt.updated_date":current_timestamp(),"tgt.upda
tedby":lit("databricks")}).execute()
display(spark.read.format("delta").option("header","true").load(target_path))
```

## 4. SCHEDULING THE PROJECT BY MASTER PIPELINE AND ADDING ETL PROCESS COMPLETION NOTIFICATION:

- After completion of all phases, It's time to schedule the whole process by invoking pipelines and notebook using master pipeline.
- We are also adding e-mail notification that the pipeline has run successfully.



To * amandeepsingh.mann13@gmail.com

Subject * Pipeline has completed successfully

Body *

Dear sir,

Fabric master pipeline of NCPL project 4 has completed successfully with pipeline ID : Pipeline

regards,

Azure Team