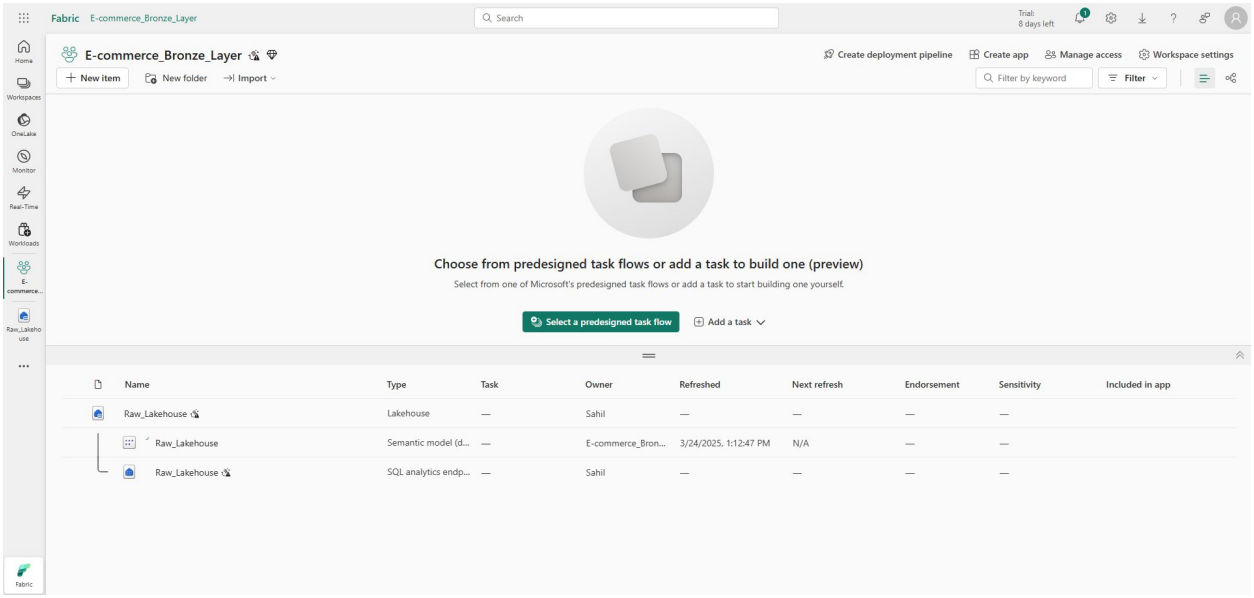
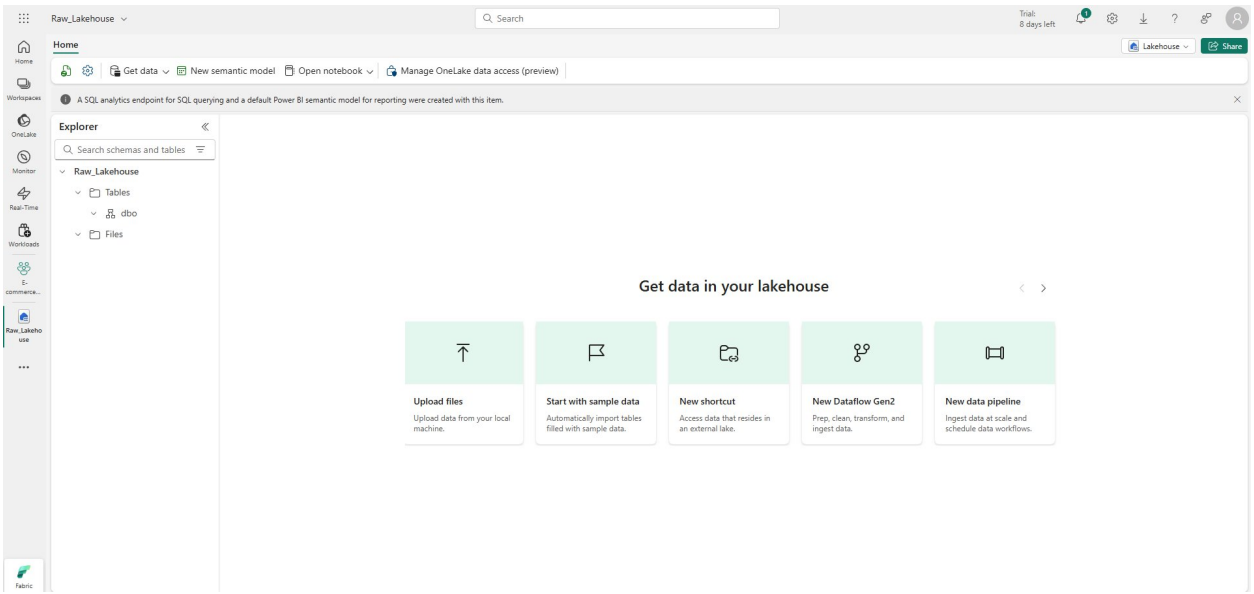
Microsoft Fabric with Medallion Architecture (Bronze-Silver-Gold)

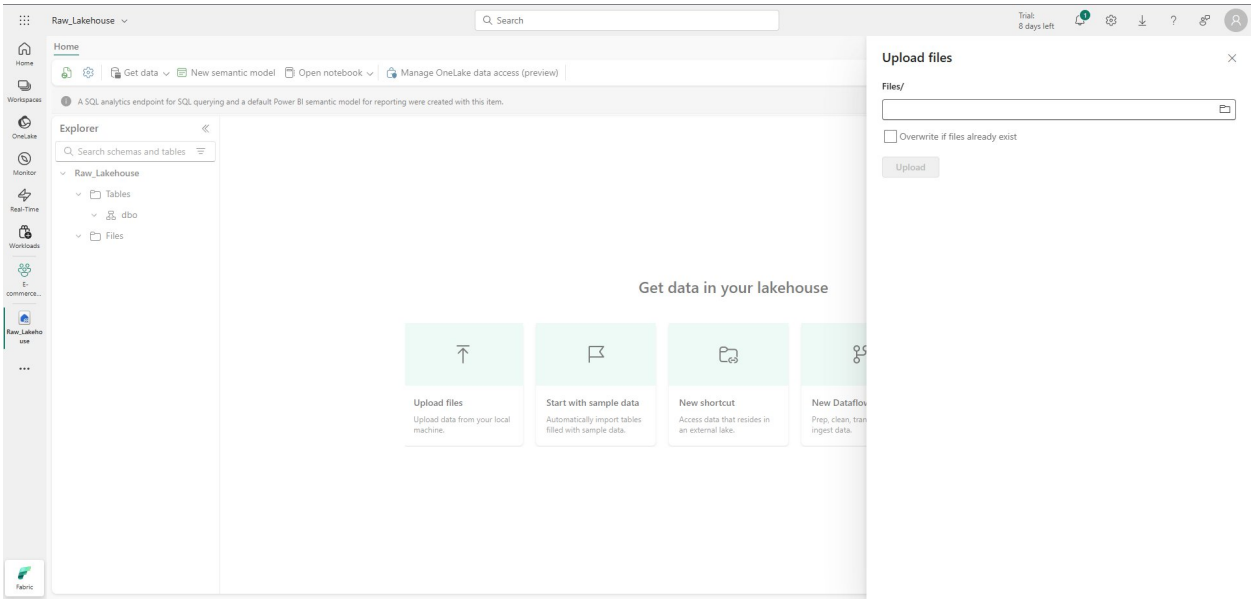
1. **Created e-commerce bronze layer**

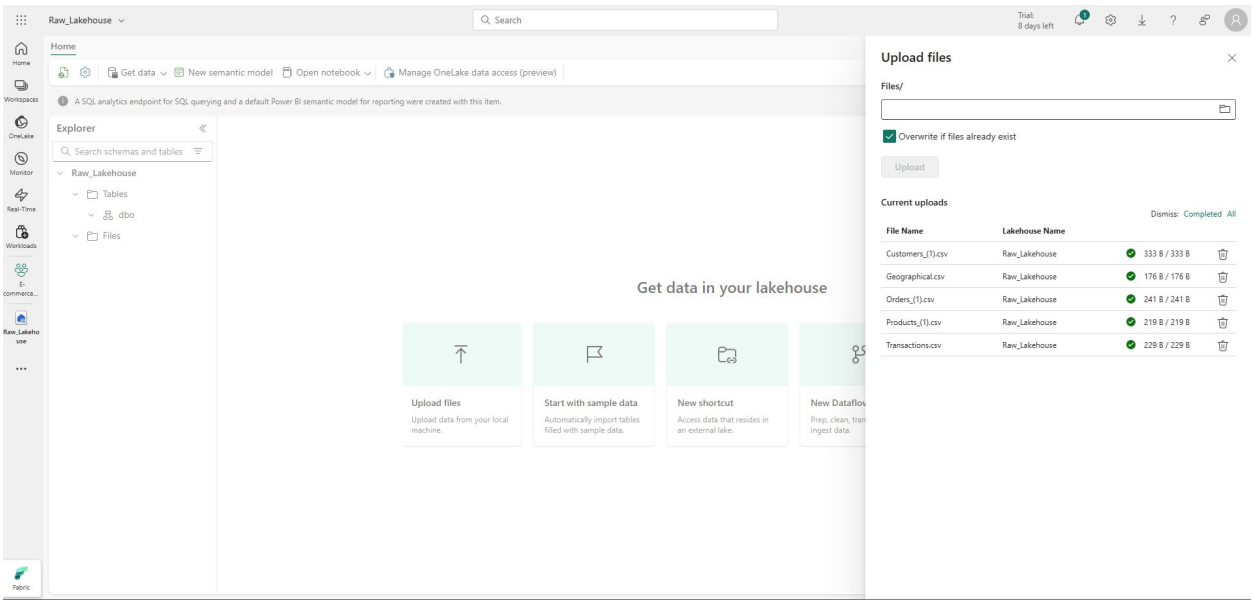


2. **Created Lakehouse in the bronze layer**

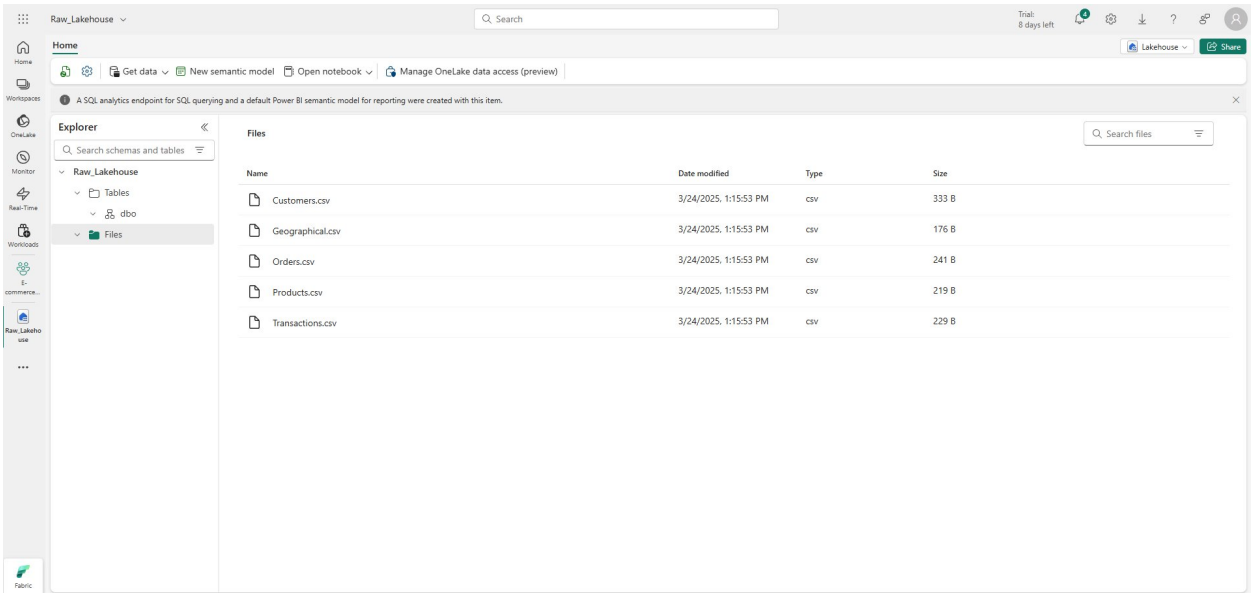


3. Uploading csv files to Lakehouse

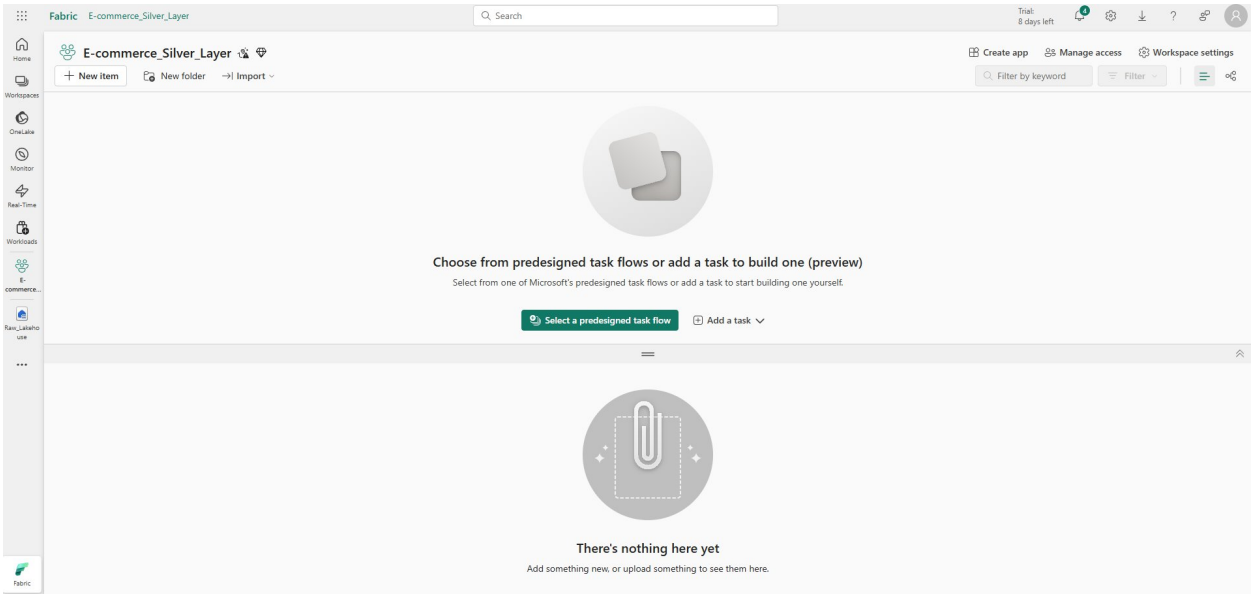


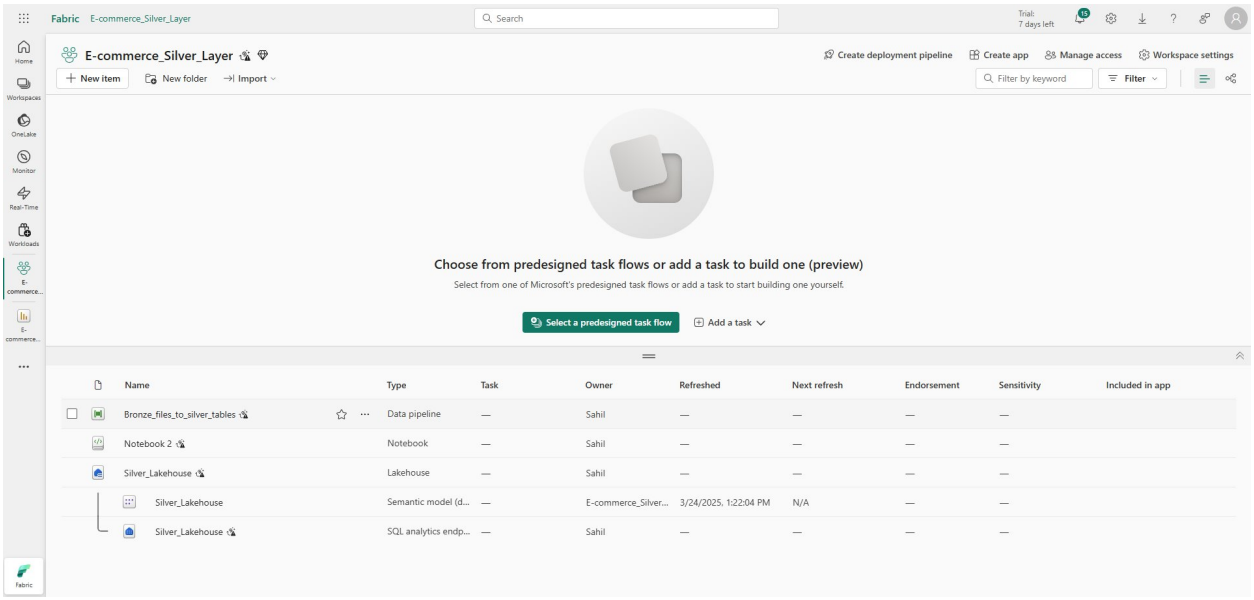


4. Files are present in Lakehouse

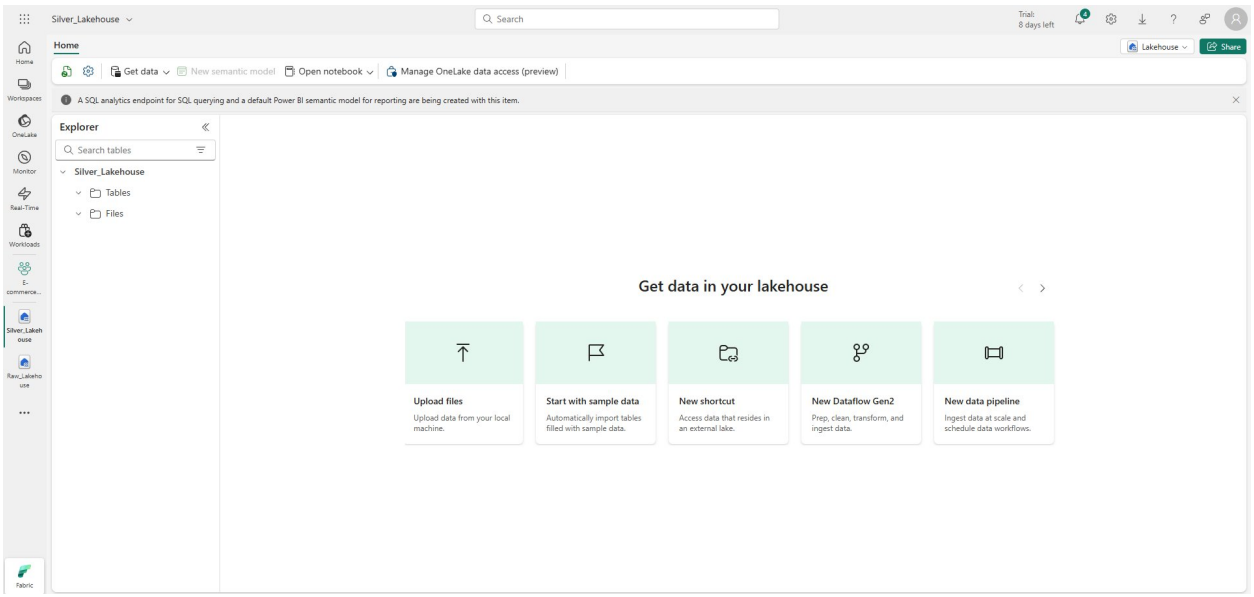


5. Separate workspace for e-commerce silver layer

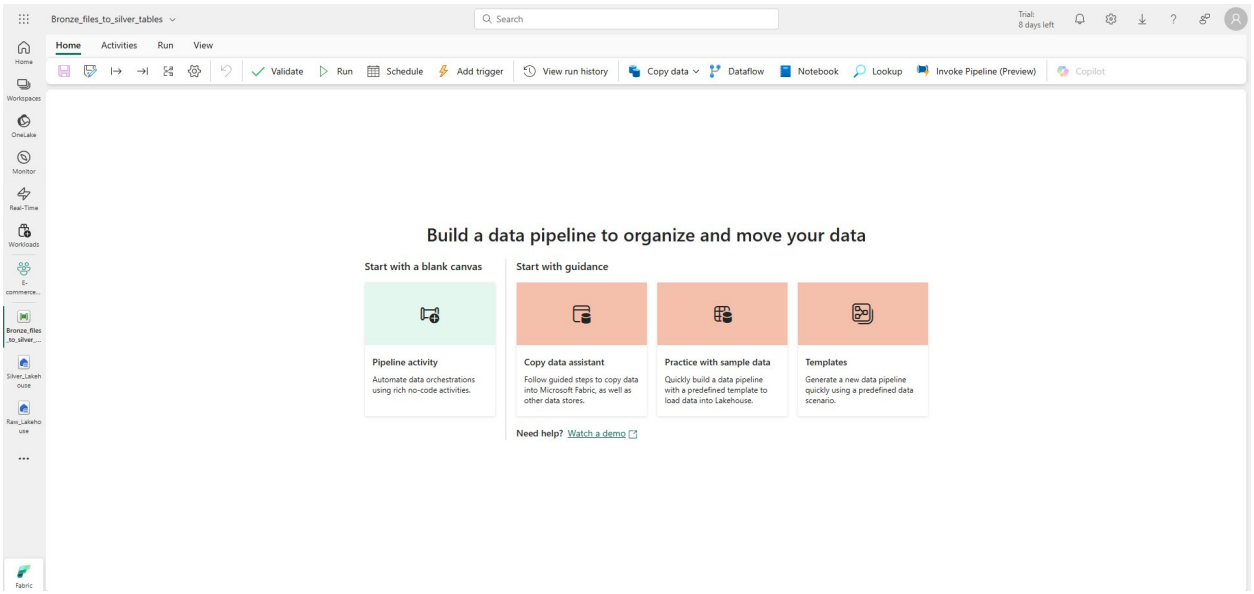




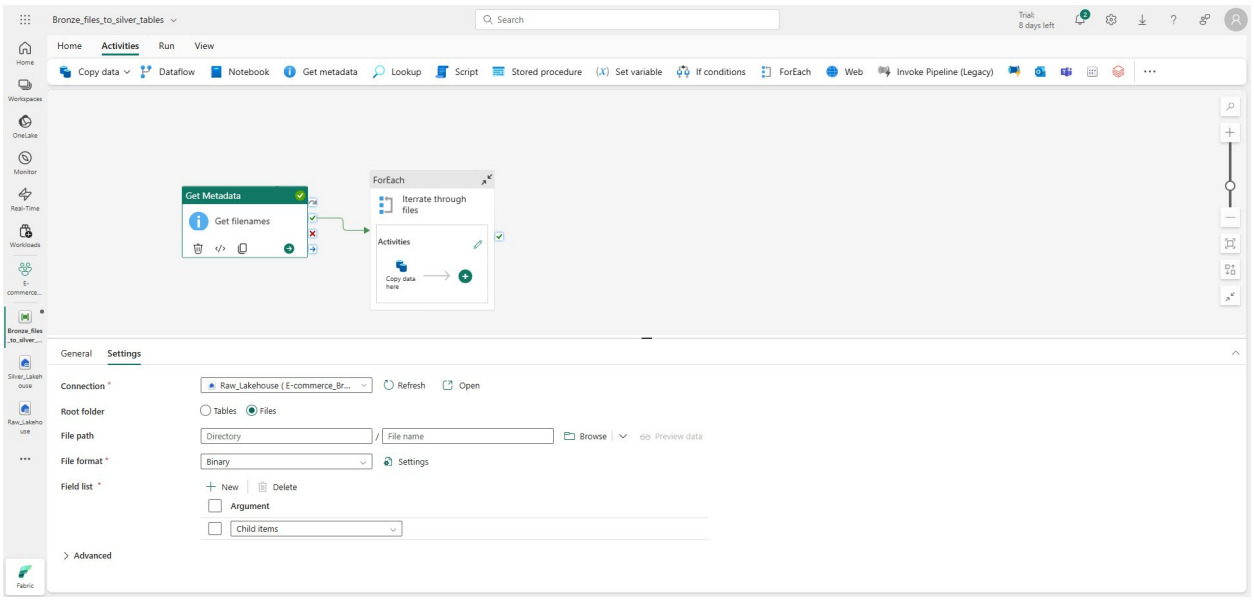
6. Load data from bronze to silver layer and want to load as tables not files Created a silver Lakehouse to store all the tables



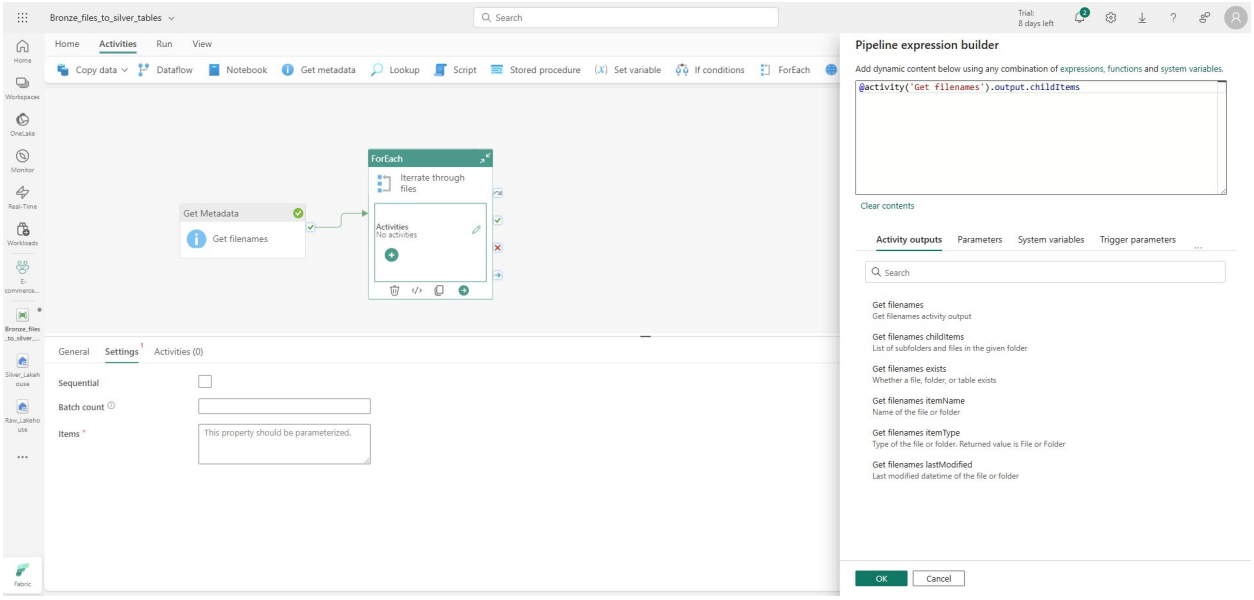
7. Created a data pipeline to convert all csv files into tables



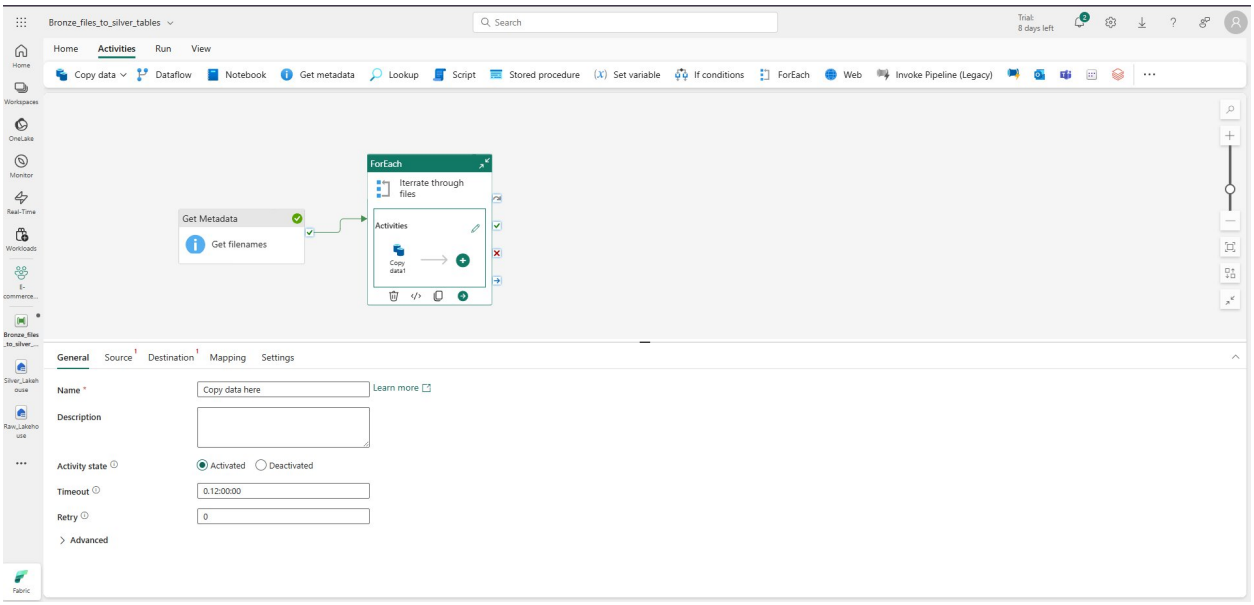
8. Get filenames from get meta data all files are present in array child items inside child items there is name property

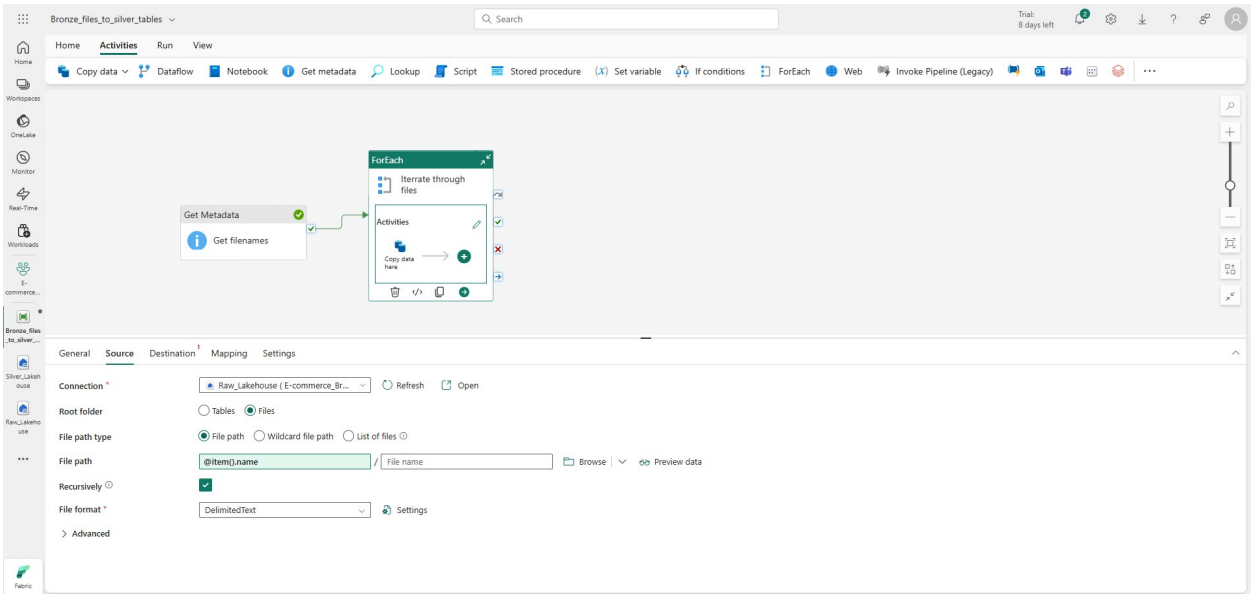


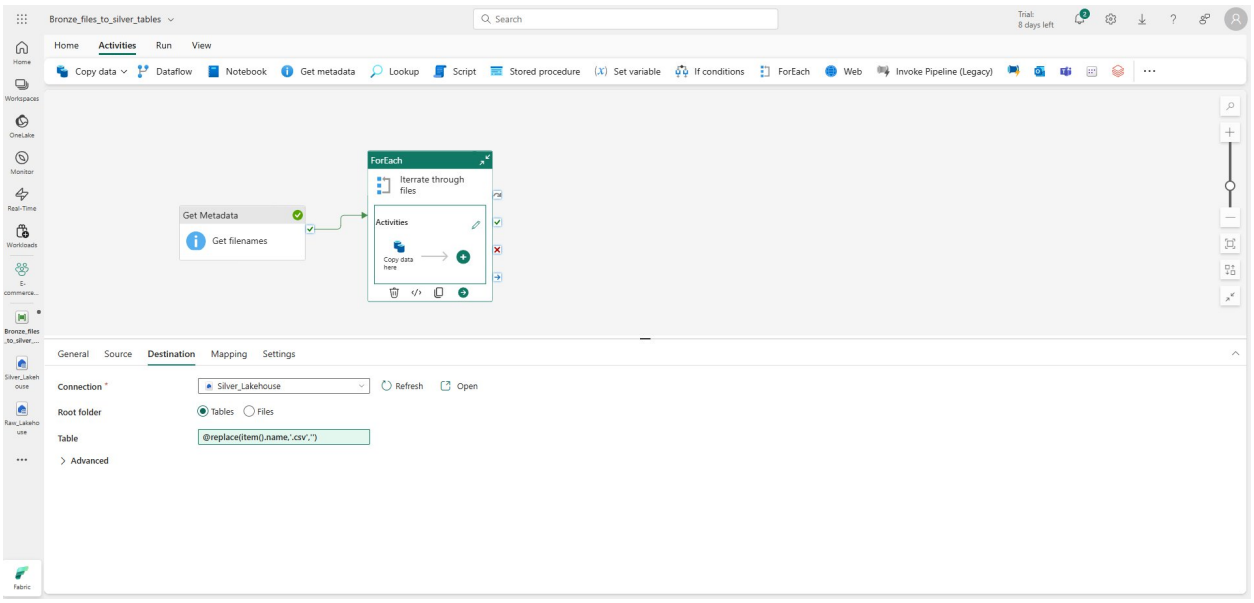
9. Iterate through child items used foreach

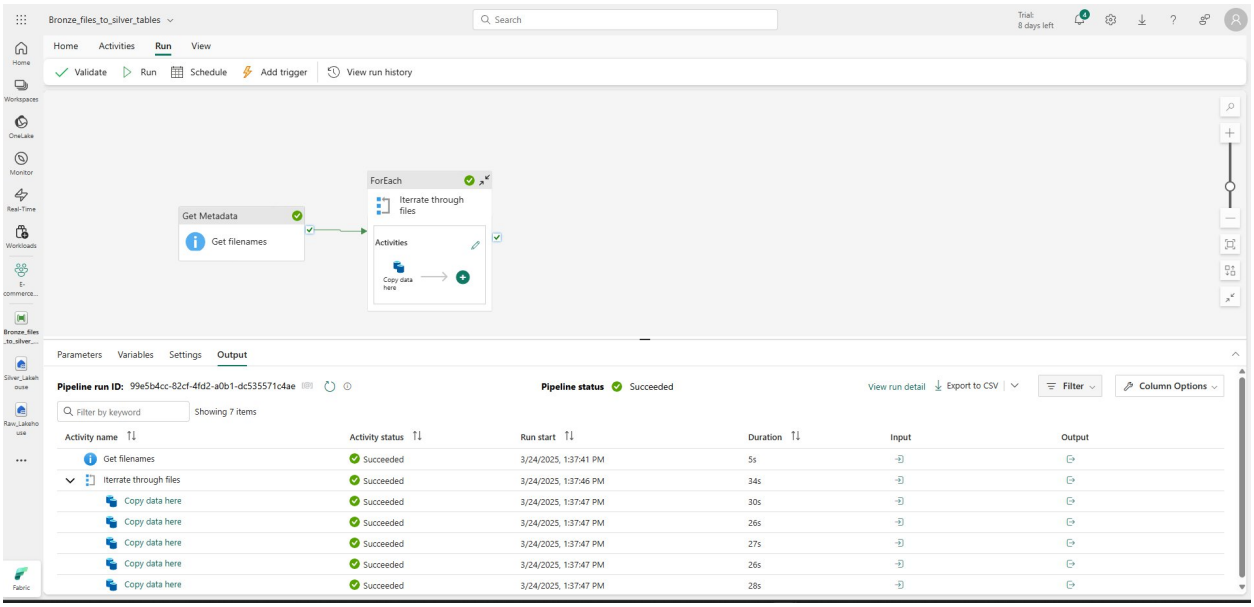


10. Copied entire data from source to destination

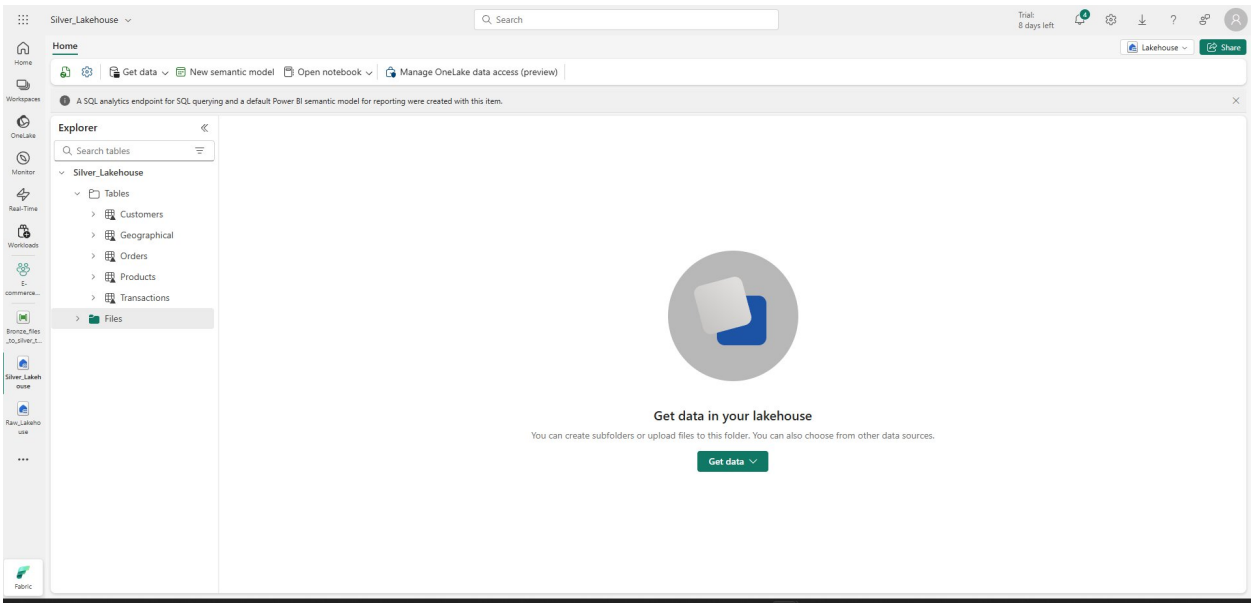

Bronze_files_to_silver_tables
Search
Trial: 8 days left
Home
Activities
Run
View
Copy data
Dataflow
Notebook
Get metadata
Lookup
Script
Stored procedure
Set variable
If conditions
ForEach
Web
Invoke Pipeline (Legacy)
Get Metadata
Get filenames
ForEach
Iterrate through files
Activities
Copy data1
General
Source
Destination
Mapping
Settings
Name *
Copy data here
Learn more
Description
Activity state
Activated
Deactivated
Timeout
0.12:00:00
Retry
0
Advanced


Bronze_files_to_silver_tables
Search
Trial: 8 days left
Home
Activities
Run
View
Copy data
Dataflow
Notebook
Get metadata
Lookup
Script
Stored procedure
Set variable
If conditions
ForEach
Web
Invoke Pipeline (Legacy)
Get Metadata
Get filenames
ForEach
Iterrate through files
Activities
Copy data here
General
Source
Destination
Mapping
Settings
Connection *
Raw_Lakehouse (E-commerce_Br...
Refresh
Open
Root folder
Tables
Files
File path type
File path
Wildcard file path
List of files
File path
@item().name
File name
Browse
Preview data
Recursively
File format *
DelimitedText
Settings
Advanced

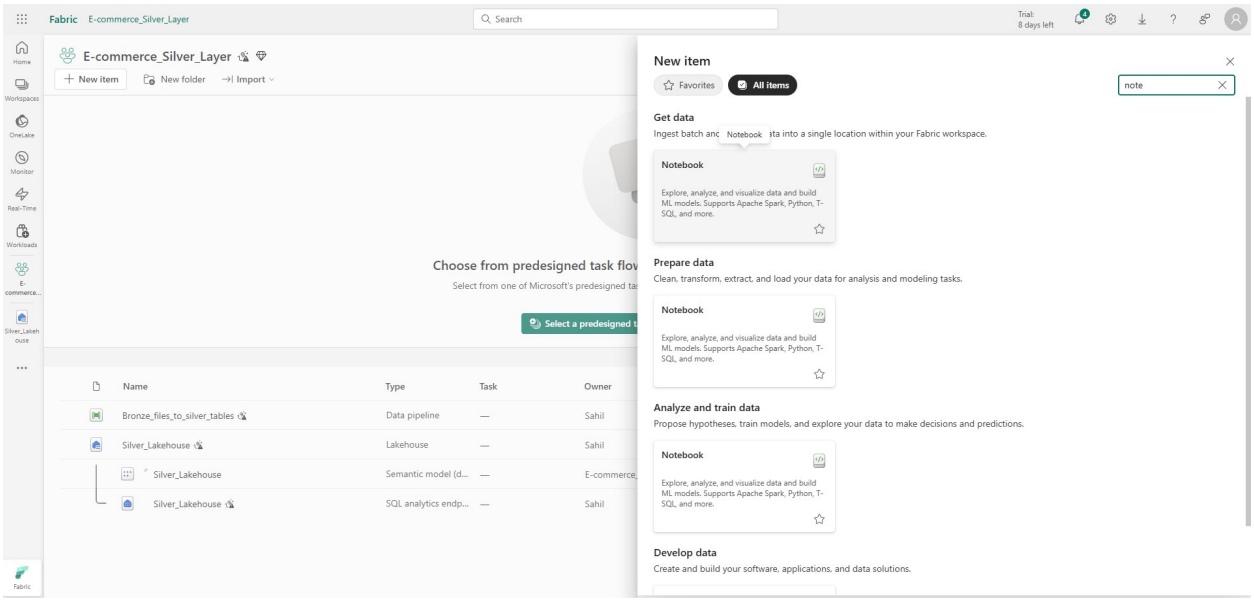




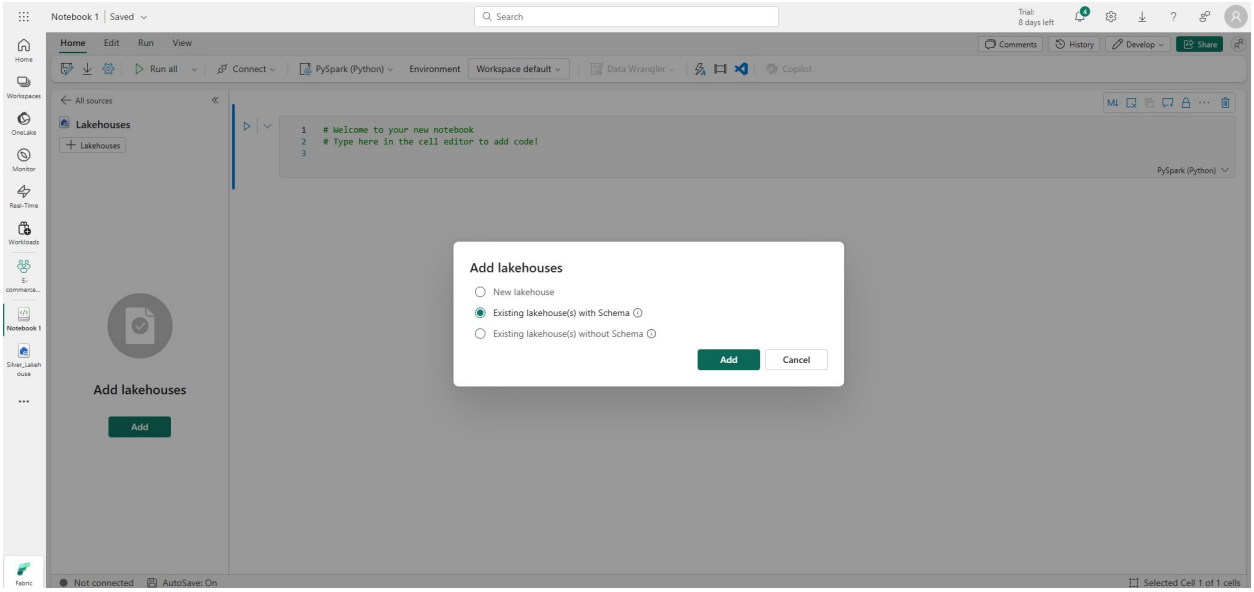
11. <u>The tables are present in silver Lakehouse</u>

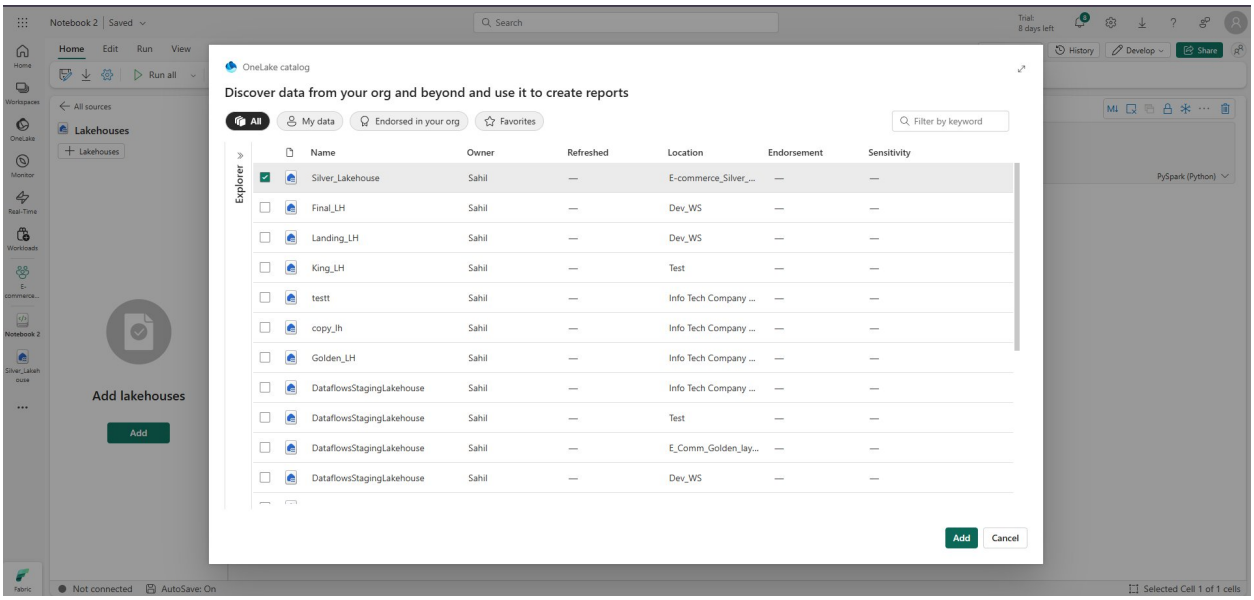


12. Now Need to do transformation on data can do this with dataflow gen 2 or Notebook all transformations happens on silver layer

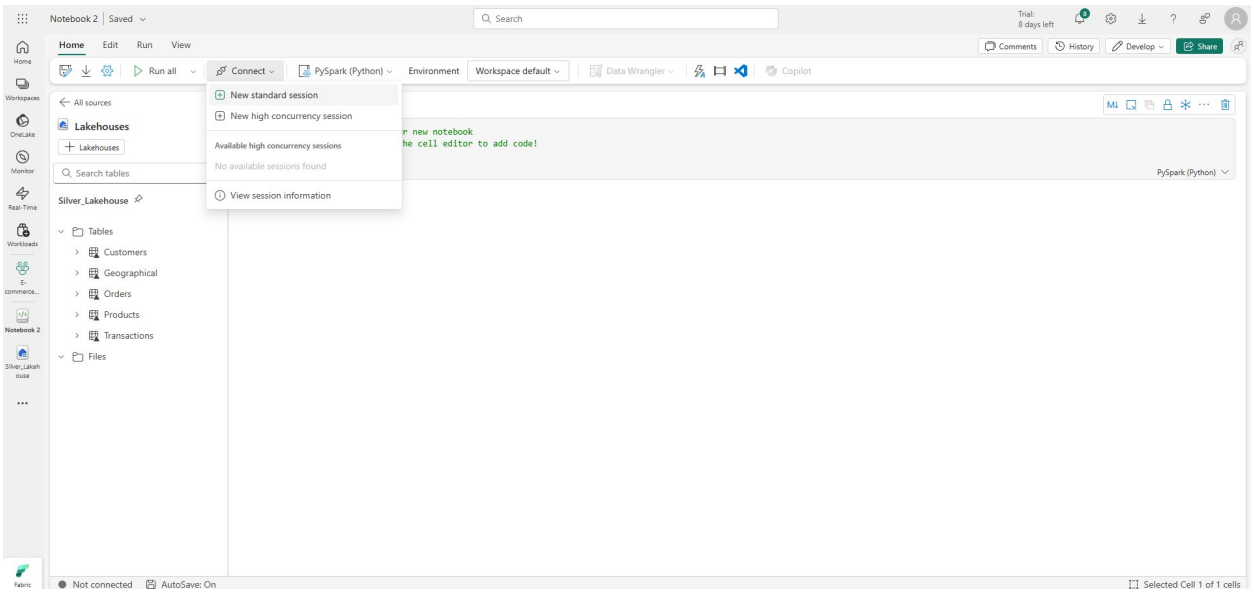


13. Connect your Lakehouse

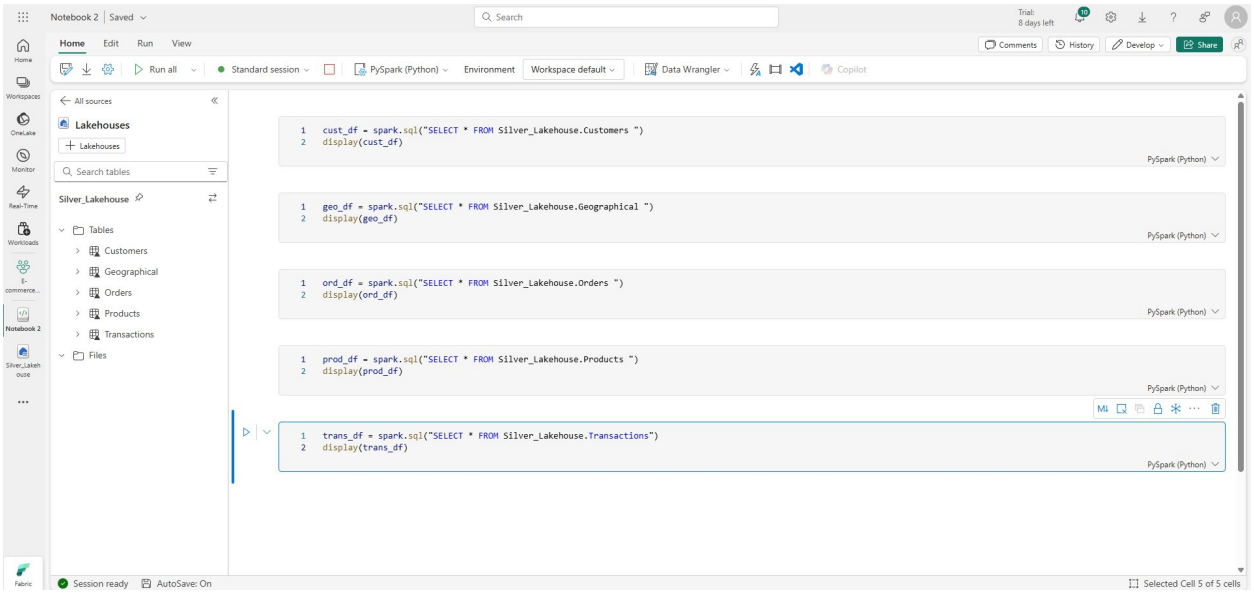


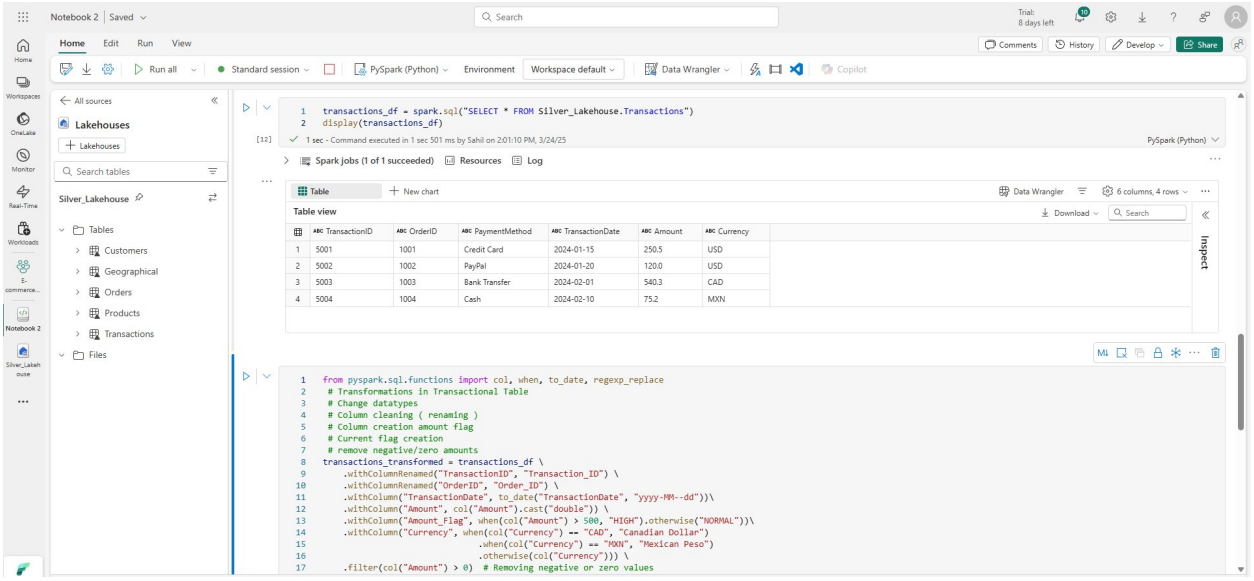


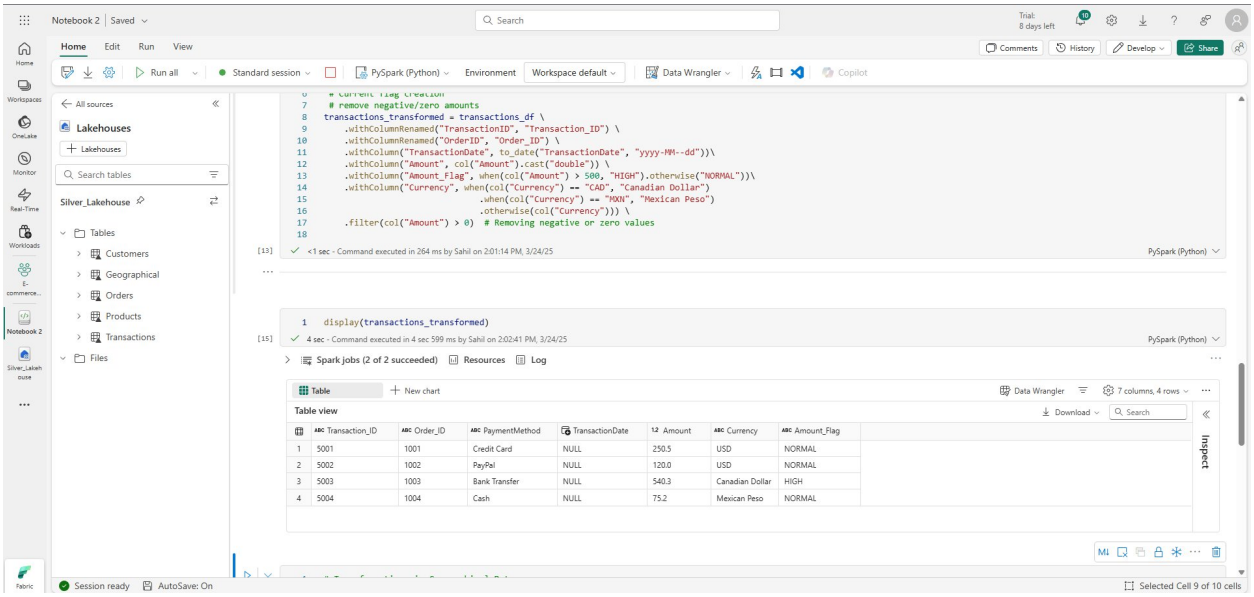
14. Connect spark session

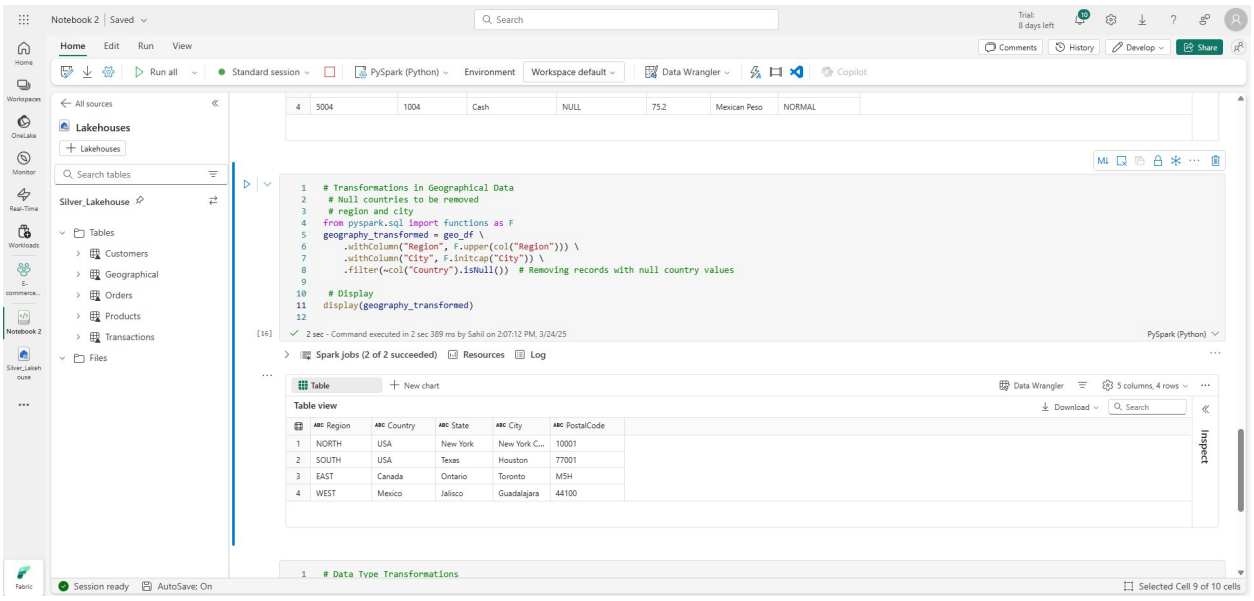


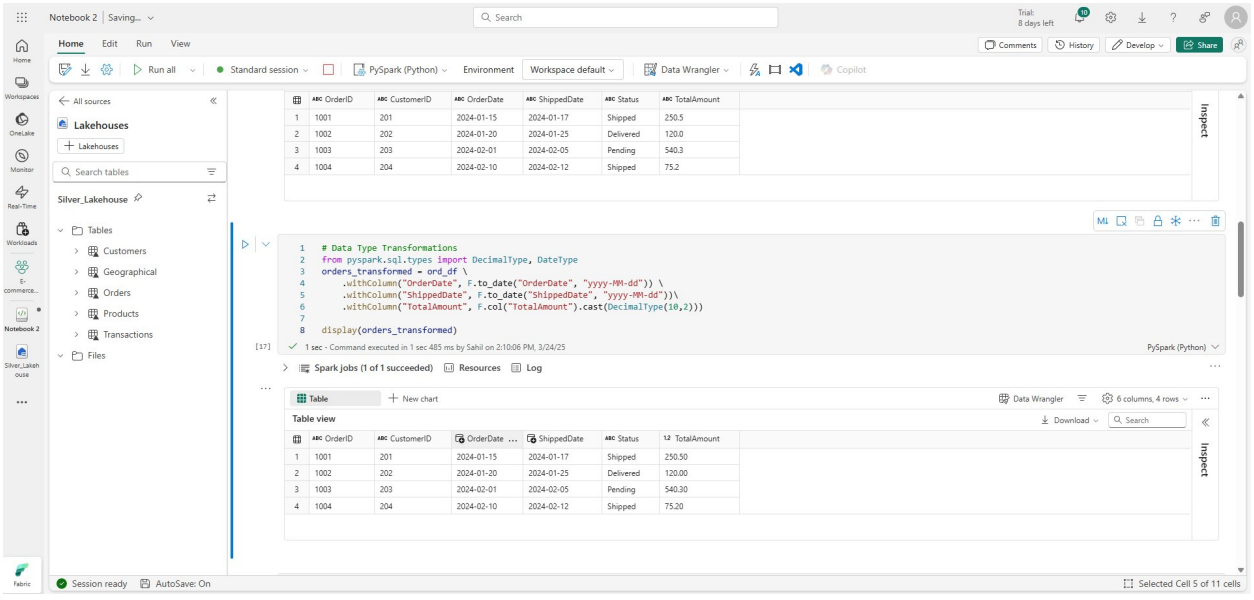
15. Now doing transformations

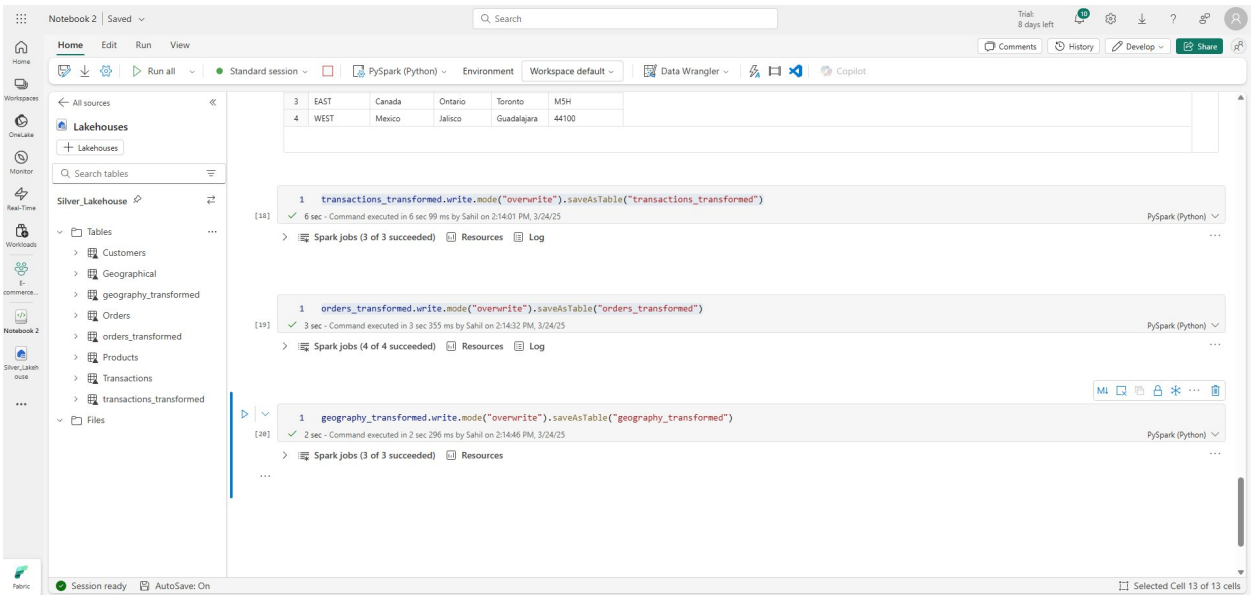



Notebook 2 | Saved
Search
Trial: 8 days left
Home Edit Run View
Comments History Develop Share
Run all Standard session PySpark (Python) Environment Workspace default Data Wrangler Copilot
All sources
Lakehouses
Lakehouses
Search tables
Silver_Lakehouse
Tables
Customers
Geographical
Orders
Products
Transactions
Files
1 transactions_df = spark.sql("SELECT * FROM Silver_Lakehouse.Transactions")
2 display(transactions_df)
[12] 1 sec - Command executed in 1 sec 501 ms by Sahil on 2:01:10 PM, 3/24/25
PySpark (Python)
Spark jobs (1 of 1 succeeded) Resources Log
Table New chart Data Wrangler 6 columns, 4 rows
Table view Download Search
TransactionID OrderID PaymentMethod TransactionDate Amount Currency
1 5001 1001 Credit Card 2024-01-15 250.5 USD
2 5002 1002 PayPal 2024-01-20 120.0 USD
3 5003 1003 Bank Transfer 2024-02-01 540.3 CAD
4 5004 1004 Cash 2024-02-10 75.2 MXN
Inspect
1 from pyspark.sql.functions import col, when, to_date, regexp_replace
2 # Transformations in Transactional Table
3 # Change datatypes
4 # Column cleaning (renaming)
5 # Column creation amount flag
6 # Current flag creation
7 # remove negative/zero amounts
8 transactions_transformed = transactions_df \
9 .withColumnRenamed("TransactionID", "Transaction_ID") \
10 .withColumnRenamed("OrderID", "Order_ID") \
11 .withColumn("TransactionDate", to_date("TransactionDate", "yyyy-MM-dd"))\
12 .withColumn("Amount", col("Amount").cast("double")) \
13 .withColumn("Amount_Flag", when(col("Amount") > 500, "HIGH").otherwise("NORMAL"))\
14 .withColumn("Currency", when(col("Currency") == "CAD", "Canadian Dollar")
15 .when(col("Currency") == "MXN", "Mexican Peso")
16 .otherwise(col("Currency"))) \
17 .filter(col("Amount") > 0) # Removing negative or zero values

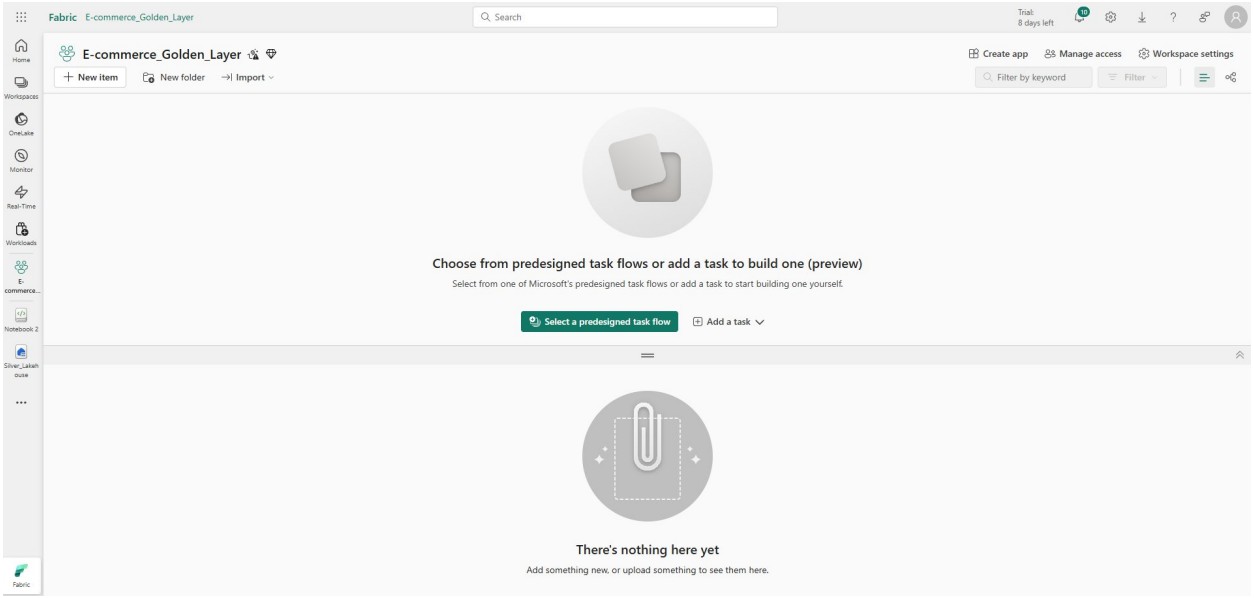

Notebook 2 | Saved
Search
Trial: 8 days left
Home Edit Run View
Comments History Develop Share
Run all Standard session PySpark (Python) Environment Workspace default Data Wrangler Copilot
All sources
Lakehouses
Lakehouses
Search tables
Silver_Lakehouse
Tables
Customers
Geographical
Orders
Products
Transactions
Files
# remove negative/zero amounts
transactions_transformed = transactions_df \
.withColumnRenamed("TransactionID", "Transaction_ID") \
.withColumnRenamed("OrderID", "Order_ID") \
.withColumn("TransactionDate", to_date("TransactionDate", "yyyy-MM-dd"))\
.withColumn("Amount", col("Amount").cast("double")) \
.withColumn("Amount_Flag", when(col("Amount") > 500, "HIGH").otherwise("NORMAL"))\
.withColumn("Currency", when(col("Currency") == "CAD", "Canadian Dollar")
.when(col("Currency") == "MXN", "Mexican Peso")
.otherwise(col("Currency"))) \
.filter(col("Amount") > 0) # Removing negative or zero values
[13] <1 sec - Command executed in 264 ms by Sahil on 2:01:14 PM, 3/24/25
PySpark (Python)
1 display(transactions_transformed)
[15] 4 sec - Command executed in 4 sec 599 ms by Sahil on 2:02:41 PM, 3/24/25
PySpark (Python)
Spark jobs (2 of 2 succeeded) Resources Log
Table New chart Data Wrangler 7 columns, 4 rows
Table view Download Search
Transaction_ID Order_ID PaymentMethod TransactionDate Amount Currency Amount_Flag
1 5001 1001 Credit Card NULL 250.5 USD NORMAL
2 5002 1002 PayPal NULL 120.0 USD NORMAL
3 5003 1003 Bank Transfer NULL 540.3 Canadian Dollar HIGH
4 5004 1004 Cash NULL 75.2 Mexican Peso NORMAL
Inspect
Session ready AutoSave: On
Selected Cell 9 of 10 cells

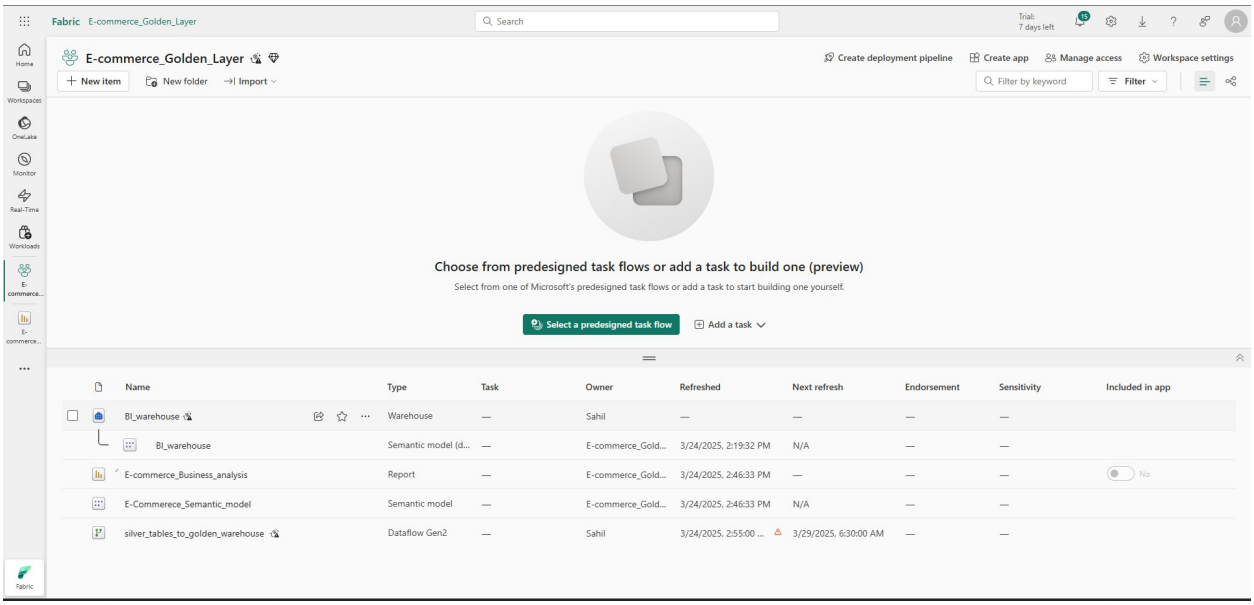

Notebook 2 | Saved
Home
Edit
Run
View
Run all
Standard session
PySpark (Python)
Environment
Workspace default
Data Wrangler
Lakehouses
Search tables
Silver_Lakehouse
Tables
Customers
Geographical
Orders
Products
Transactions
Files
# Transformations in Geographical Data
# Null countries to be removed
# region and city
from pyspark.sql import functions as F
geography_transformed = geo_df \
.withColumn("Region", F.upper(col("Region"))) \
.withColumn("City", F.initcap("City")) \
.filter(~col("Country").isNull()) # Removing records with null country values
# Display
display(geography_transformed)
2 sec - Command executed in 2 sec 389 ms by Sahil on 2:07:12 PM, 3/24/25
PySpark (Python)
Spark jobs (2 of 2 succeeded)
Resources
Log
Table
New chart
Table view
Region
Country
State
City
PostalCode
NORTH
USA
New York
New York C...
10001
SOUTH
USA
Texas
Houston
77001
EAST
Canada
Ontario
Toronto
M5H
WEST
Mexico
Jalisco
Guadalajara
44100
Inspect
Data Type Transformations
Session ready
AutoSave: On
Selected Cell 9 of 10 cells

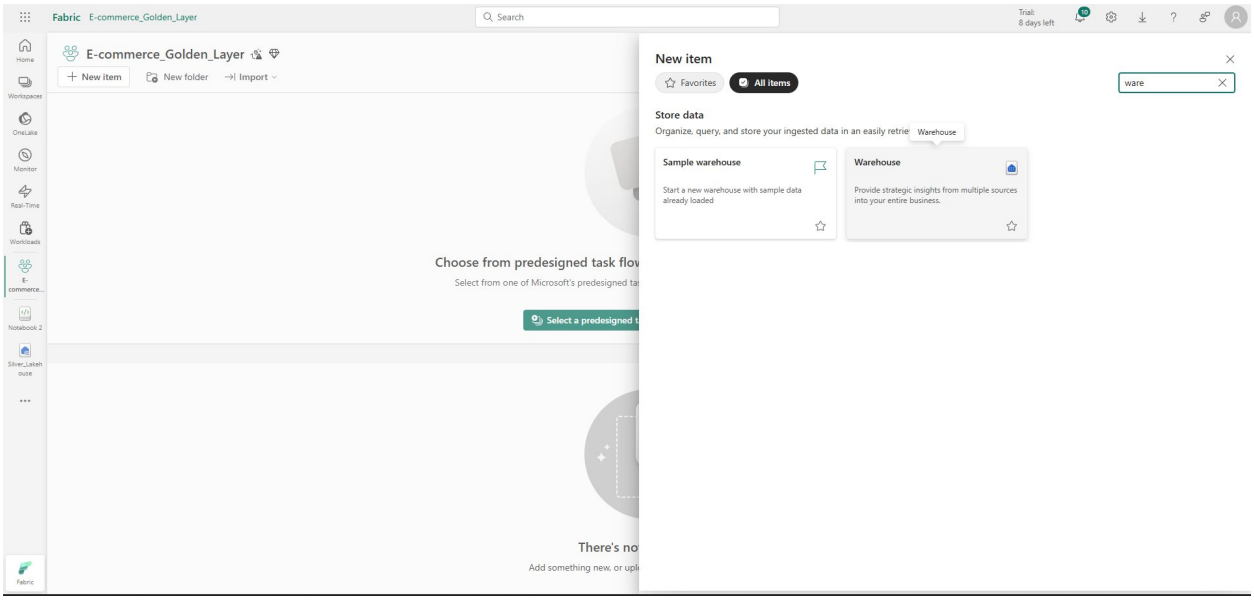

Notebook 2 | Saving...
Home
Edit
Run
View
Run all
Standard session
PySpark (Python)
Environment
Workspace default
Data Wrangler
Lakehouses
Search tables
Silver_Lakehouse
Tables
Customers
Geographical
Orders
Products
Transactions
Files
OrderID
CustomerID
OrderDate
ShippedDate
Status
TotalAmount
1001
201
2024-01-15
2024-01-17
Shipped
250.5
1002
202
2024-01-20
2024-01-25
Delivered
120.0
1003
203
2024-02-01
2024-02-05
Pending
540.3
1004
204
2024-02-10
2024-02-12
Shipped
75.2
# Data Type Transformations
from pyspark.sql.types import DecimalType, DateType
orders_transformed = ord_df \
.withColumn("OrderDate", F.to_date("OrderDate", "yyyy-MM-dd")) \
.withColumn("ShippedDate", F.to_date("ShippedDate", "yyyy-MM-dd"))\
.withColumn("TotalAmount", F.col("TotalAmount").cast(DecimalType(10,2)))
display(orders_transformed)
1 sec - Command executed in 1 sec 485 ms by Sahil on 2:10:06 PM, 3/24/25
PySpark (Python)
Spark jobs (1 of 1 succeeded)
Resources
Log
Table
New chart
Table view
Inspect
Session ready
AutoSave: On
Selected Cell 5 of 11 cells



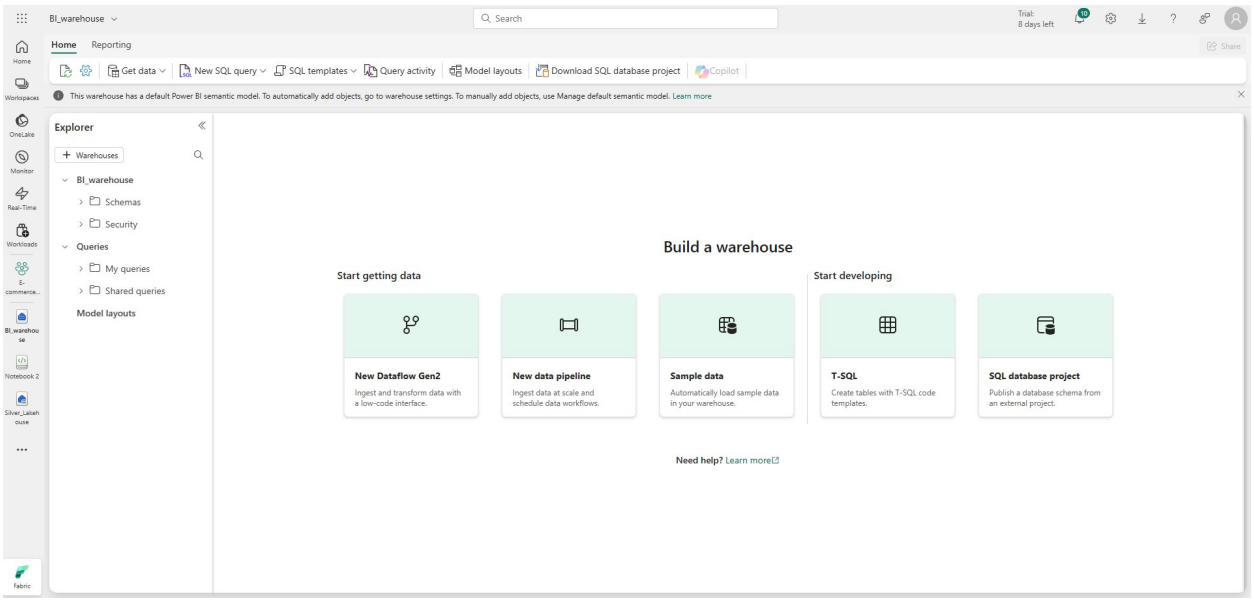
16. **<u>In Golder layer we load data in warehouses because warehouses are more efficient for reading data as they are OLAP Systems</u>**





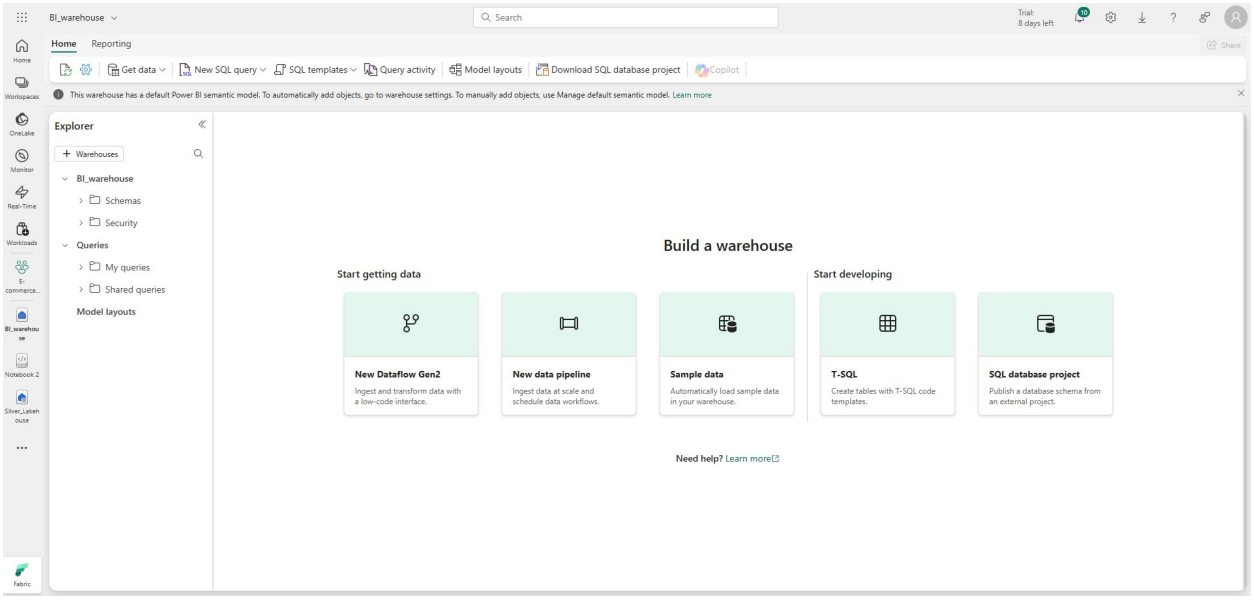


17. **<u>Now you want to load your cleaned and transformed data to this warehouse from silver layer</u>**

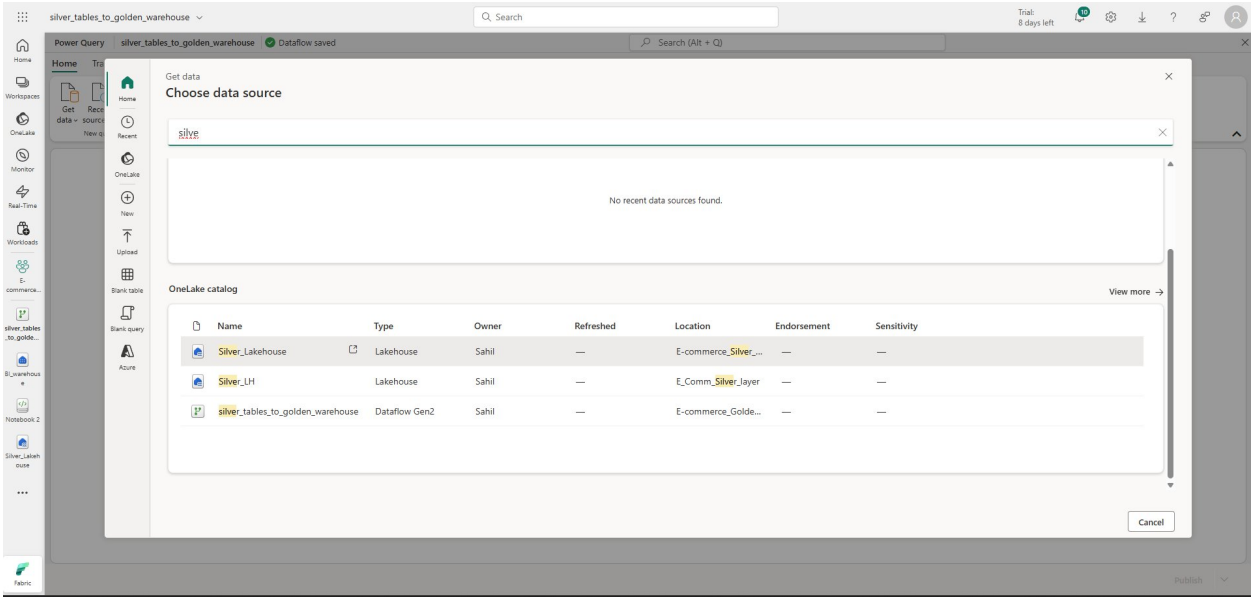


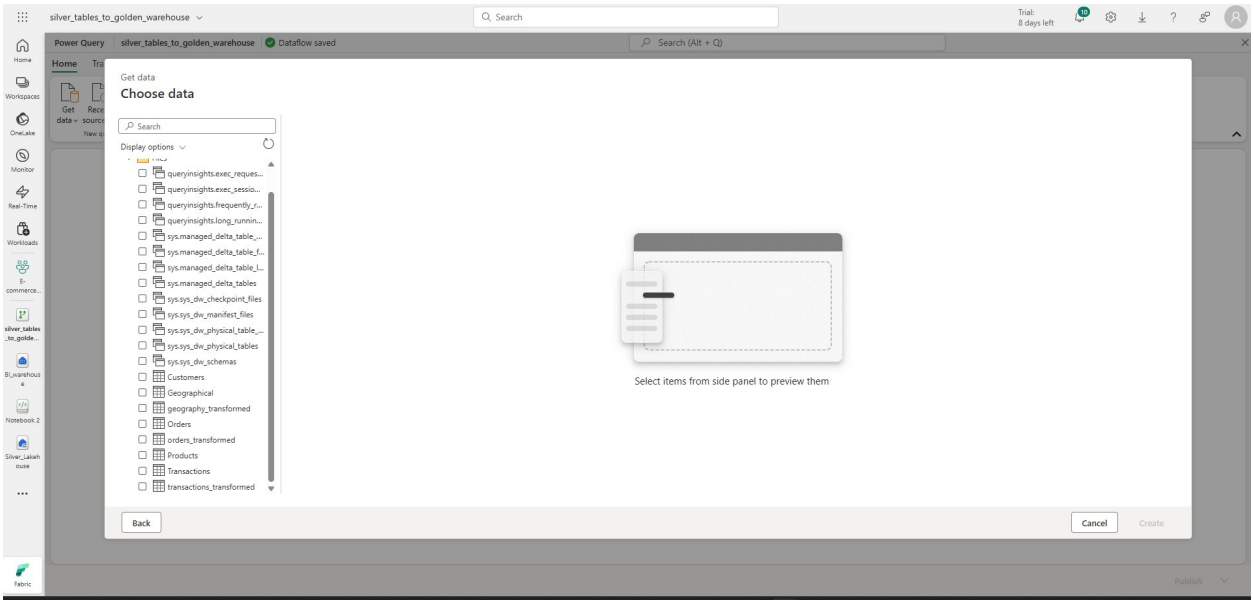
How will I bring my data?

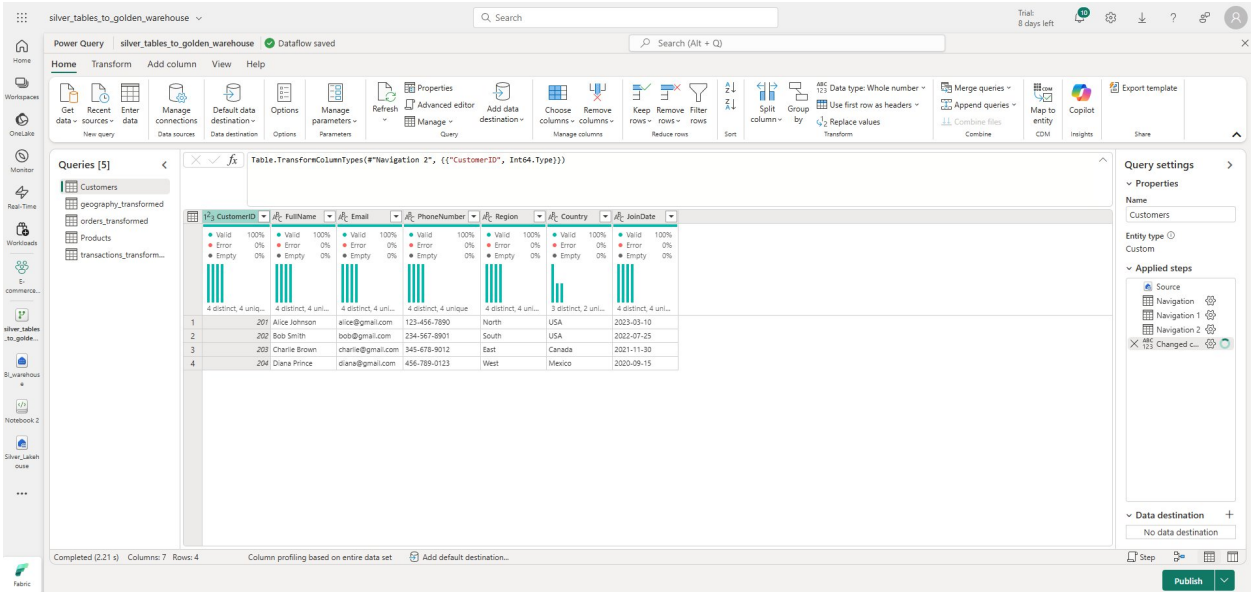
By dataflow gen2 why? Because of the golden layer and in this layer, we try to prefer things which are less technical so that noncoding users can also use this



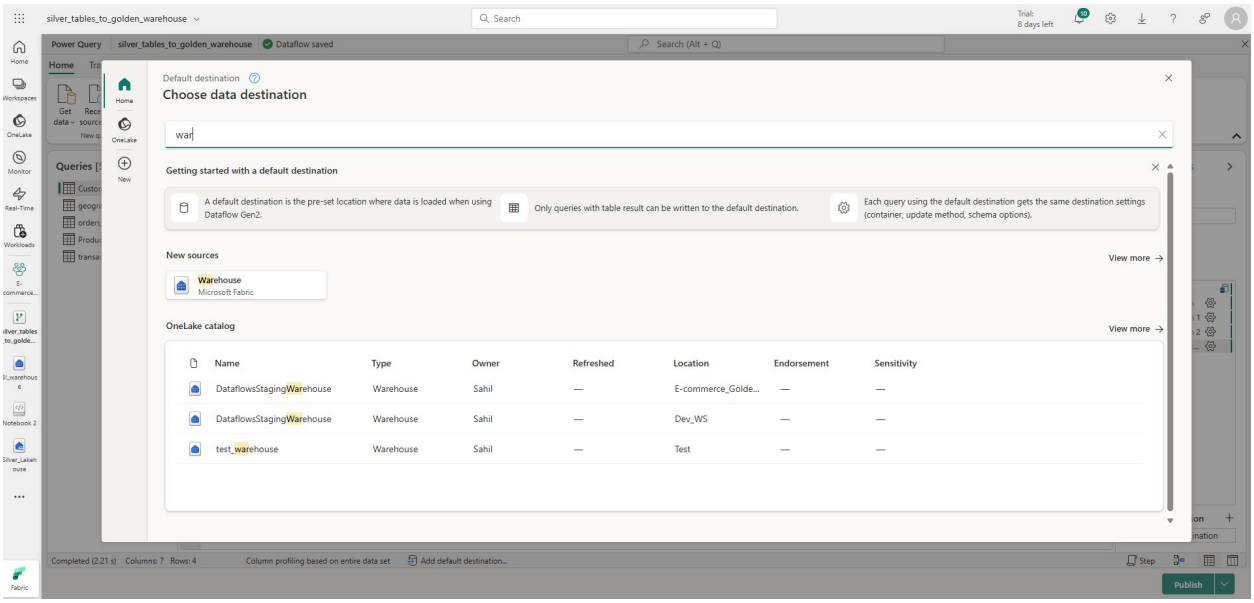
18. Load data from silver Lake house

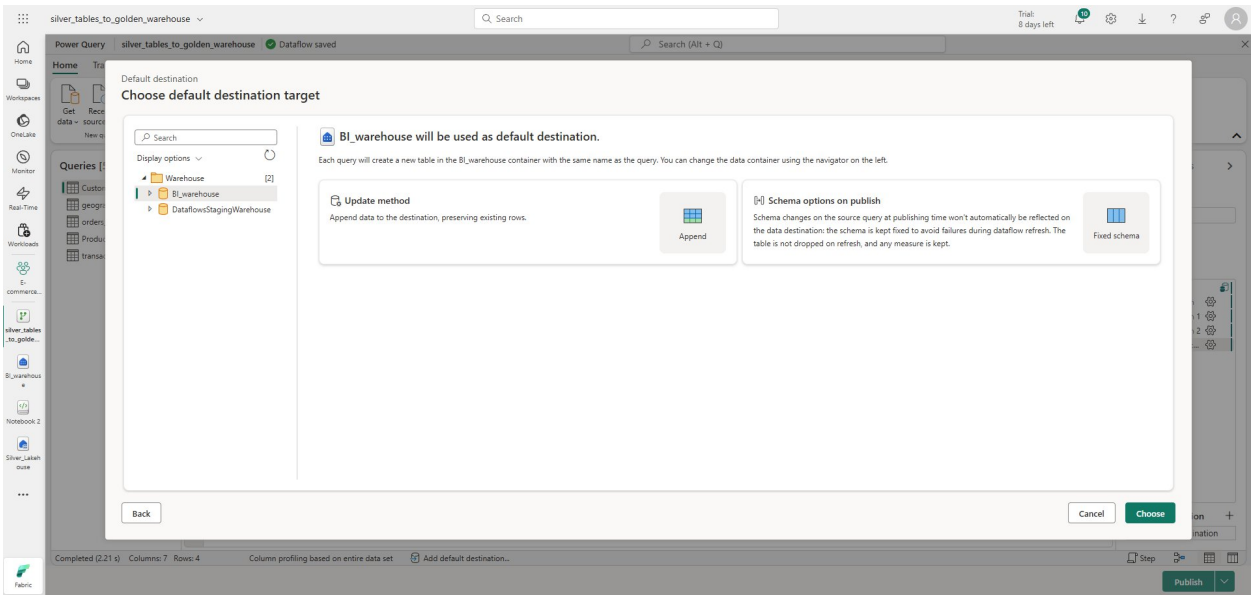

silver_tables_to_golden_warehouse
Search
Trial: 8 days left
Power Query
silver_tables_to_golden_warehouse
Dataflow saved
Search (Alt + Q)
Home
Get data
Choose data source
silve
No recent data sources found.
OneLake catalog
View more
Name
Type
Owner
Refreshed
Location
Endorsement
Sensitivity
Silver_Lakehouse
Lakehouse
Sahil
—
E-commerce_Silver_...
—
—
Silver_LH
Lakehouse
Sahil
—
E_Comm_Silver_layer
—
—
silver_tables_to_golden_warehouse
Dataflow Gen2
Sahil
—
E-commerce_Golde...
—
—
Cancel


silver_tables_to_golden_warehouse
Search
Trial: 8 days left
Power Query
silver_tables_to_golden_warehouse
Dataflow saved
Search (Alt + Q)
Home
Get data
Choose data
Search
Display options
queryinsights.exec_reques...
queryinsights.exec_sessio...
queryinsights.frequently_r...
queryinsights.long_runnin...
sys.managed_delta_table_...
sys.managed_delta_table_f...
sys.managed_delta_table_l...
sys.managed_delta_tables
sys.sys_dw_checkpoint_files
sys.sys_dw_manifest_files
sys.sys_dw_physical_table_...
sys.sys_dw_physical_tables
sys.sys_dw_schemas
Customers
Geographical
geography_transformed
Orders
orders_transformed
Products
Transactions
transactions_transformed
Select items from side panel to preview them
Back
Cancel
Create

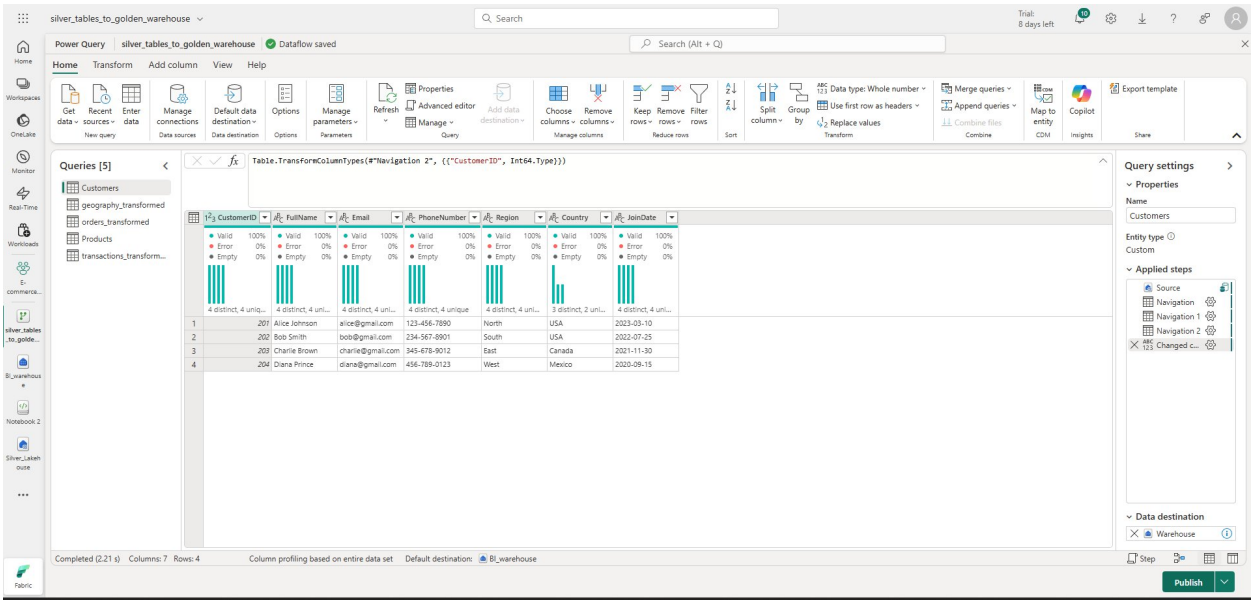


19. This is a silver layer and want to load into the golden layer's warehouse

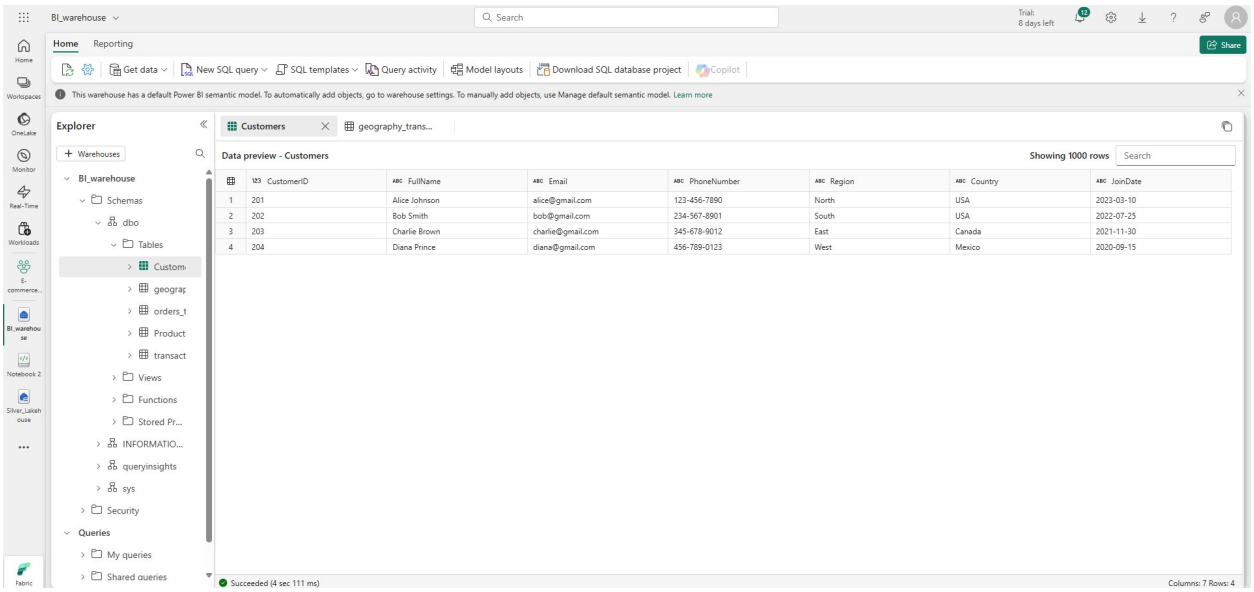




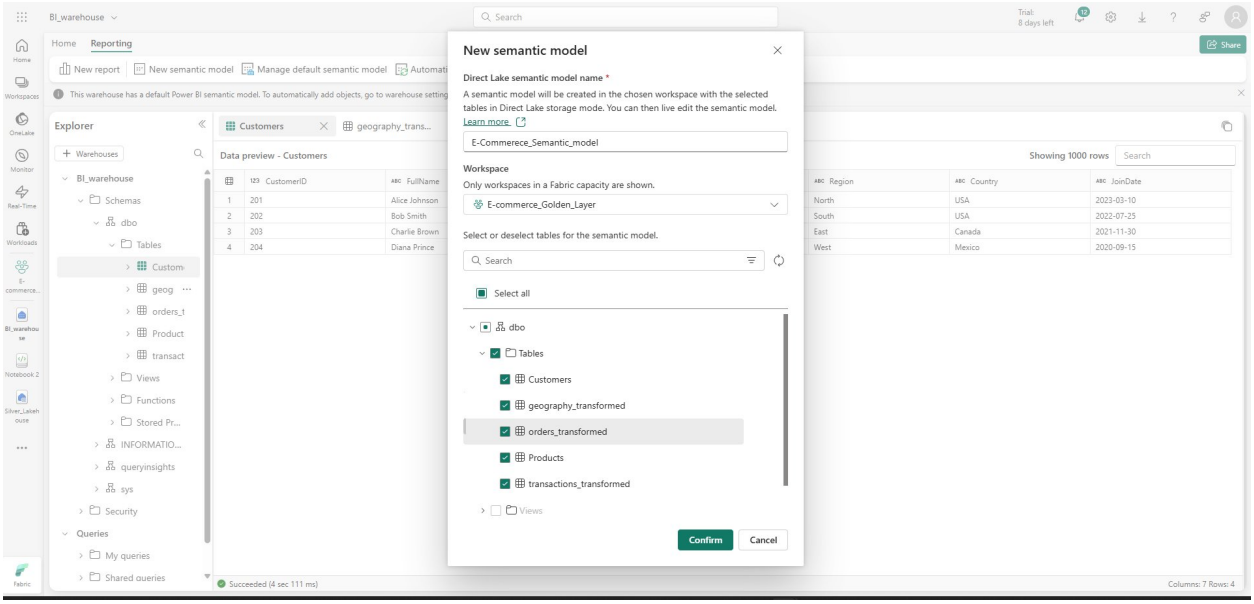
20. Click on publish

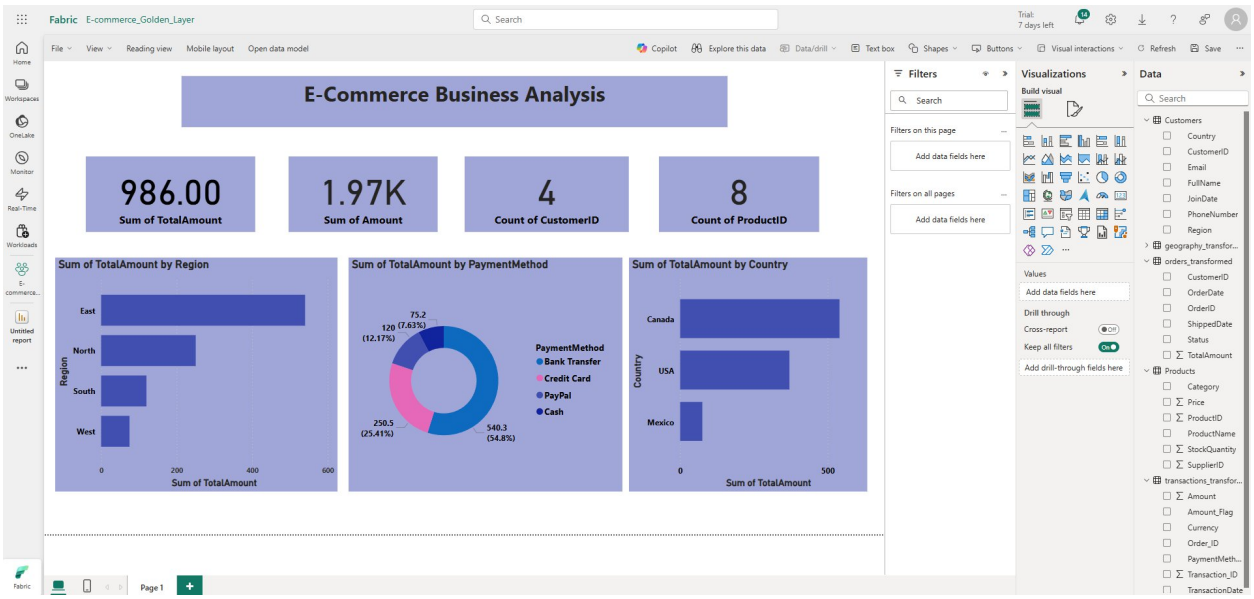


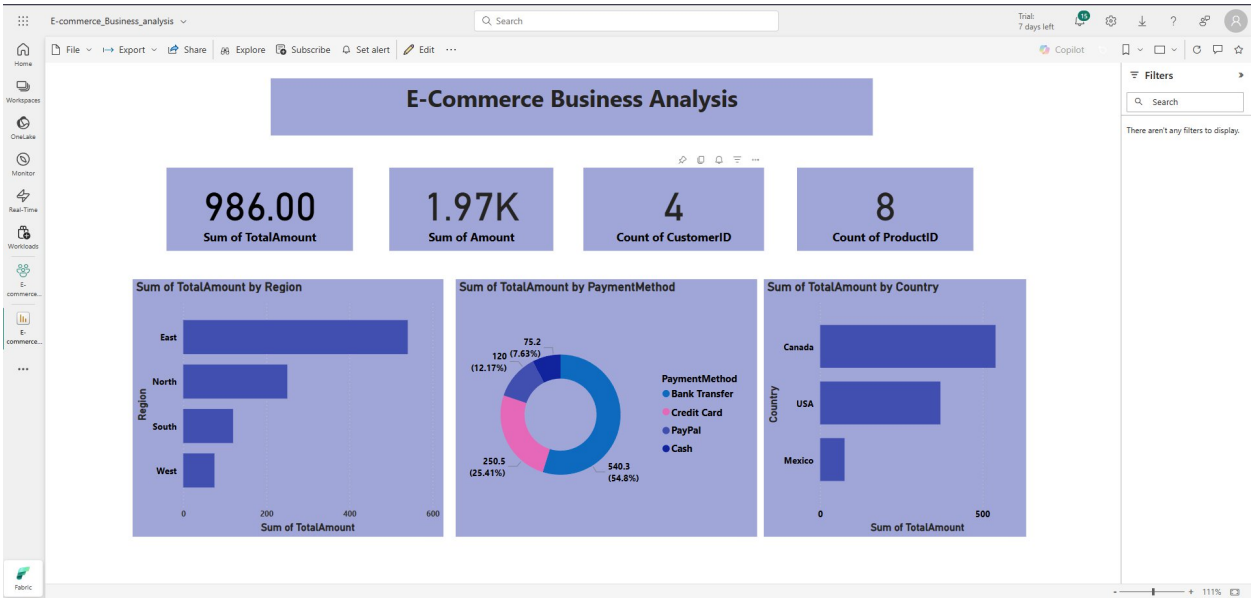
21. Your data finally moves to the warehouse where all BI tools connect



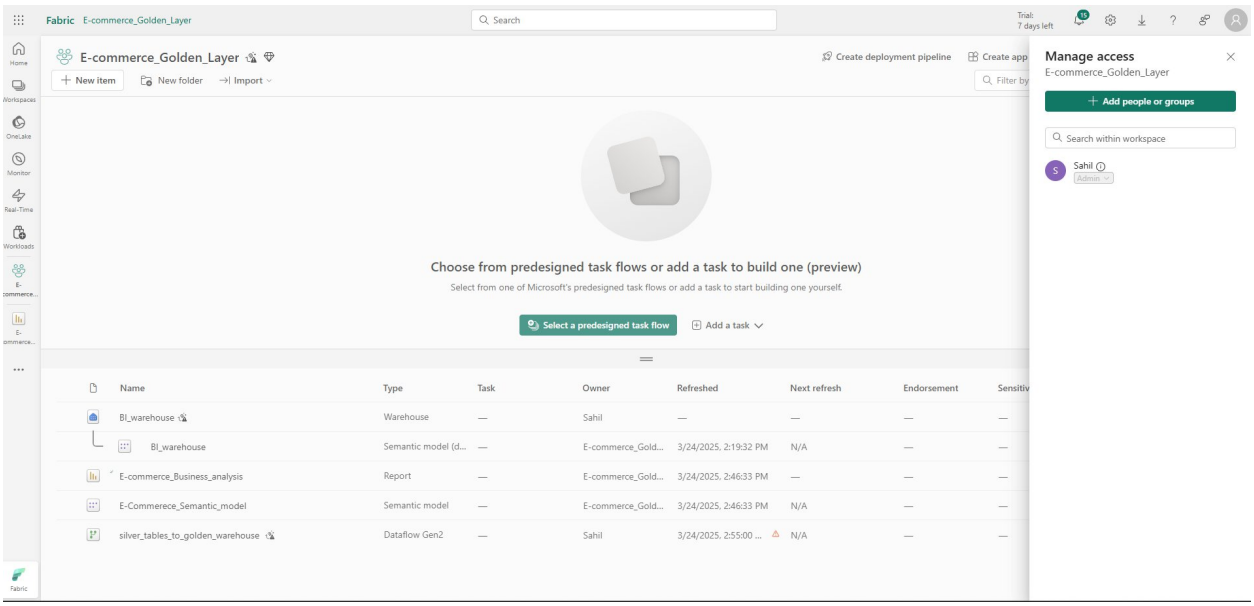
22. Finally, to end, users create report







23. To give access



24. To schedule the refresh for all the data pipeline, dataflows and notebooks