BOOTCAMP PROJECT – 5

MIGRATING PIPELINES FROM ADF TO SYNAPSE

AND

MIGRATING AZURE DATABRICKS NOTEBOOKS TO FABRIC NOTEBOOKS

SUBMITTED BY – DRUSYA SURESH

PROJECT OBJECTIVE:

TASK 1:

To build an end-to-end data pipeline on Azure Data Factory that:

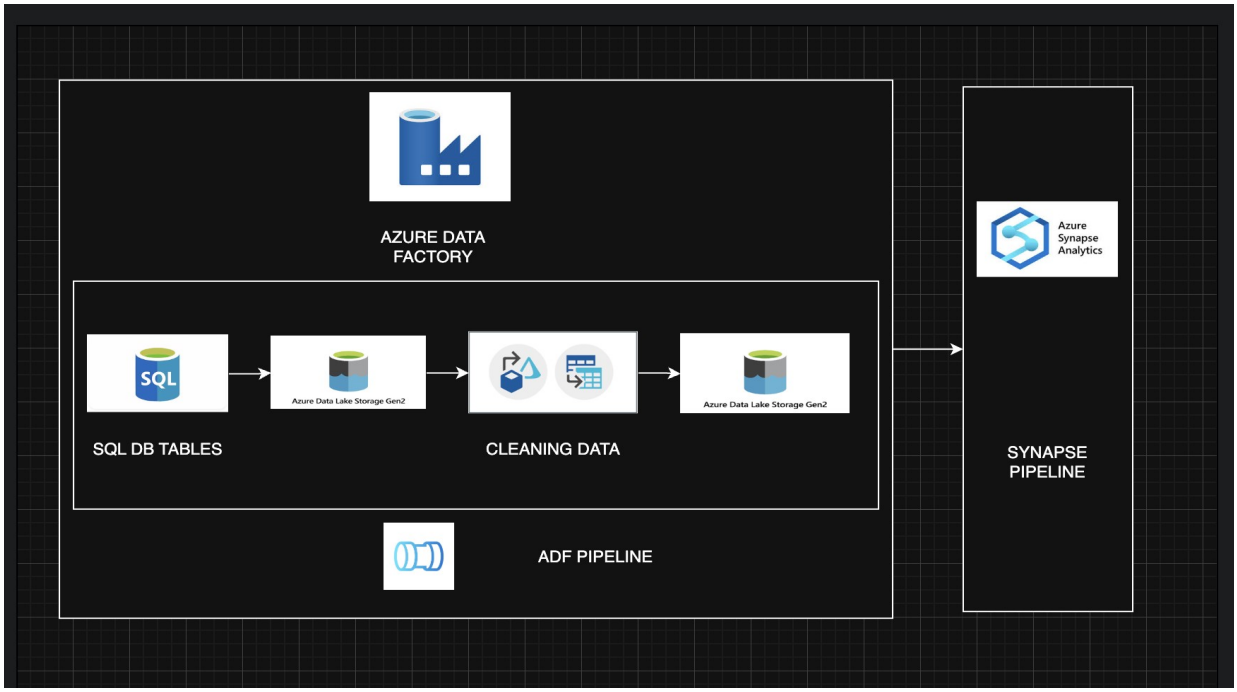
1. Create a pipeline to bring multiple tables from Azure SQL to ADLS gen2.
2. Create a pipeline to clean the data using Dataflows.
3. Migrate the above developed pipelines from ADF to synapse.
4. Test the pipelines after changing the connections.

TASK 2:

1. Create four Notebooks in Databricks.
2. Migrate these Notebooks from Databricks to Fabric.
3. Change connections and Test the Notebooks

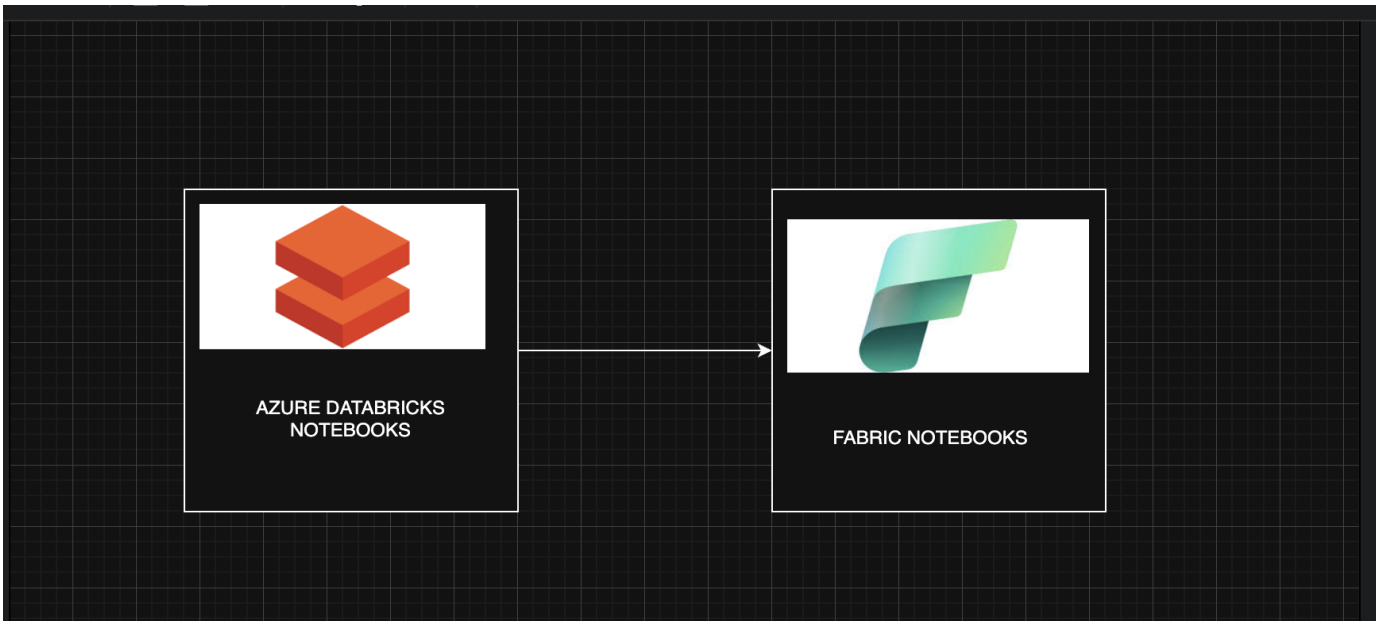
ARCHITECTURE DIAGRAM:

TASK 1:



-

TASK 2:



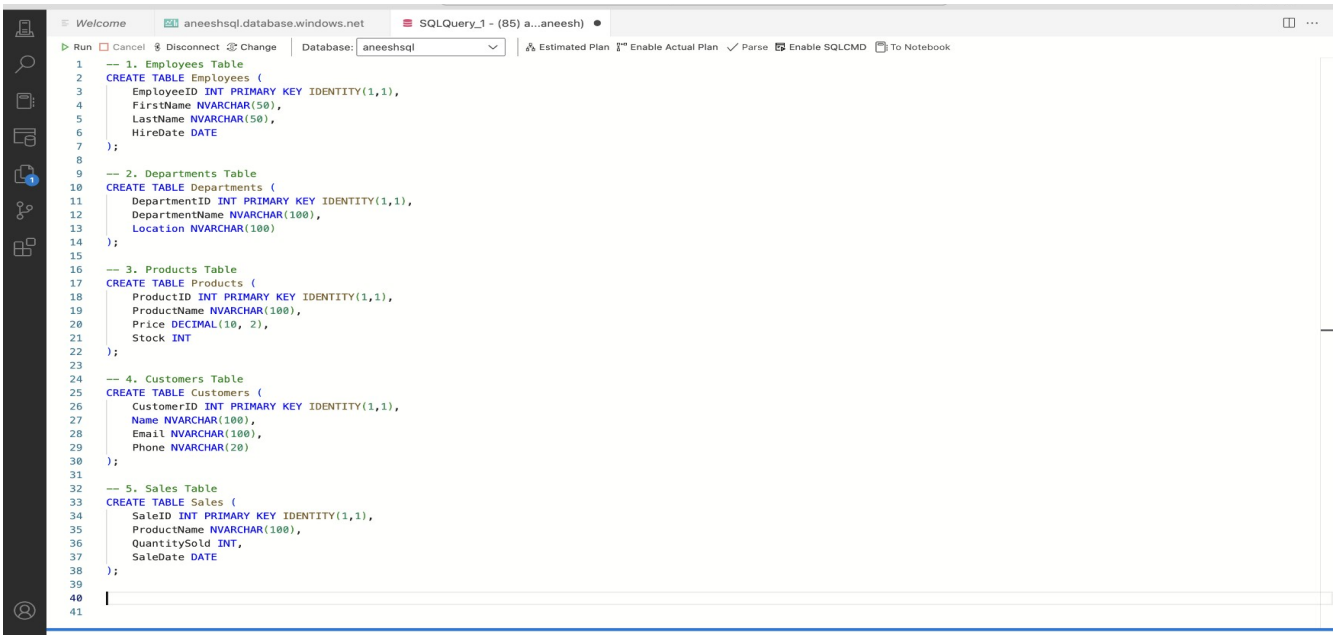
TOOLS AND TECHNOLOGIES USED:

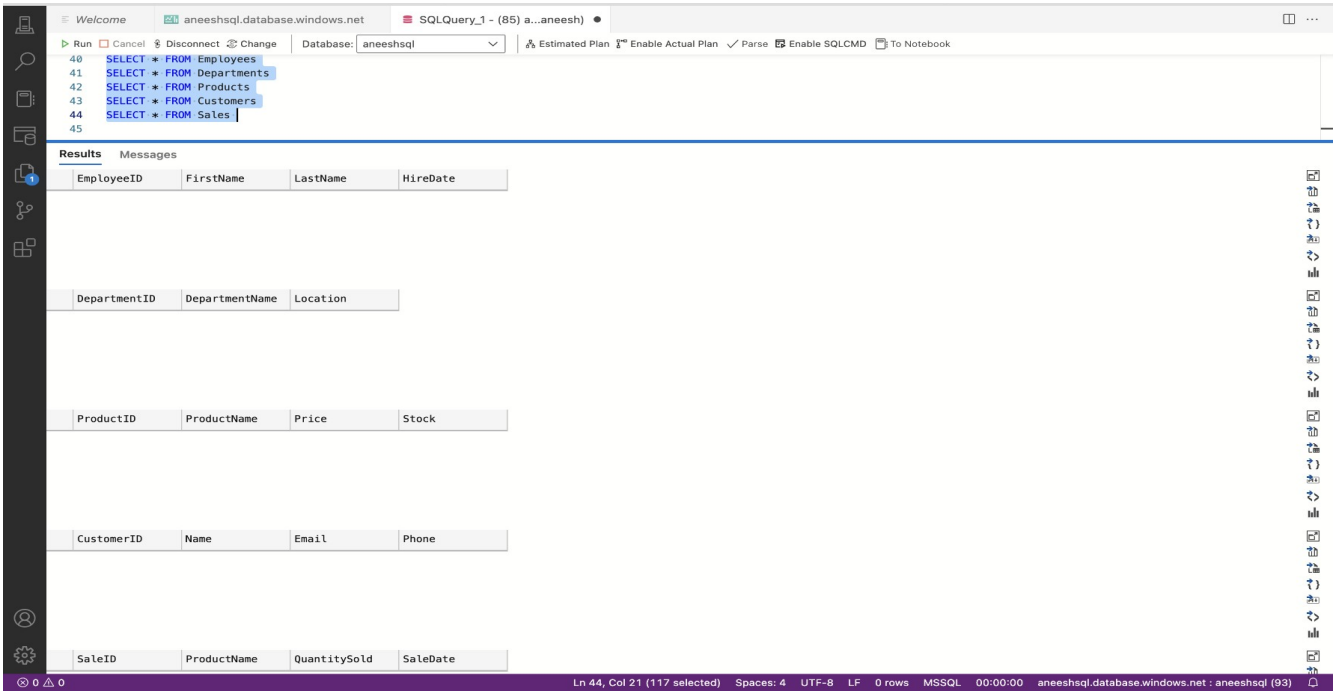
- Azure Data factory
- Azure Synapse
- Azure SQL
- Azure data lake Gen2
- Azure key vault
- Databricks
- Fabric

TASK 1: MIGRATING PIPELINE FROM ADF TO SYNAPSE

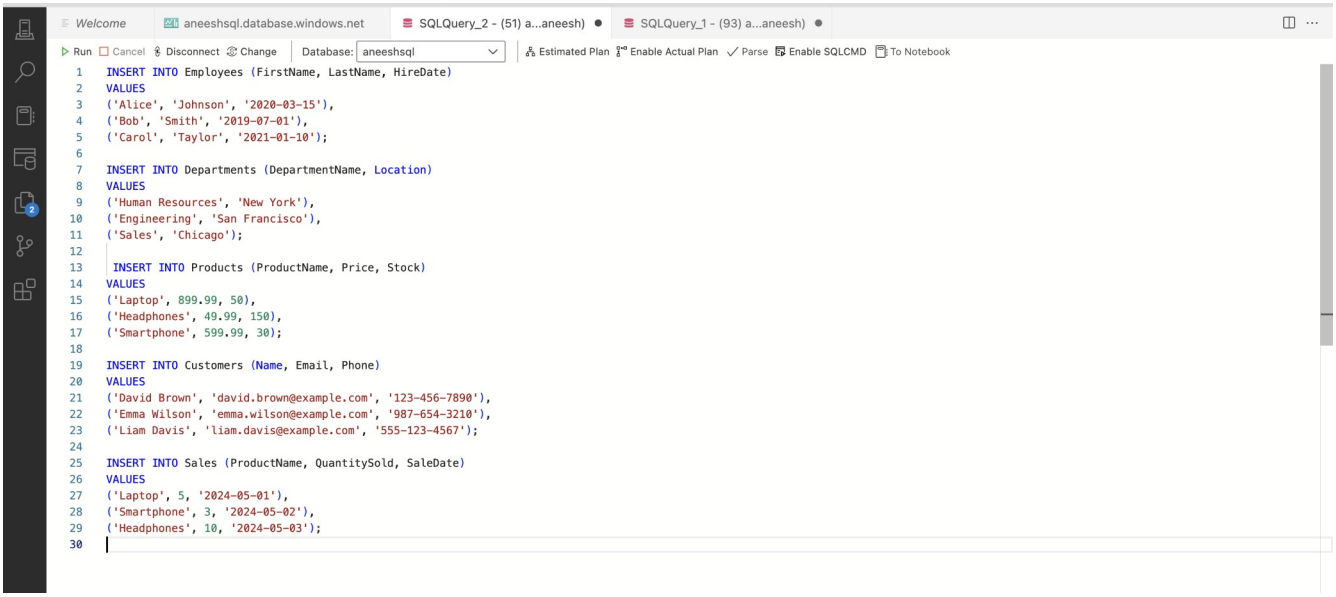
Created a pipeline in Azure data factory for copying multiple tables from SQL DB to ADLS gen 2 bronze container as csv files. Then using data flows cleaned the data and loaded it back to ADLS silver container. For connecting SQL DB and ADLS I use the key vault integration for security of credentials. The step by step creation of the pipeline is shown below and after creating the pipeline in data factory I migrate the pipeline into Synapse and run it there using the migrating methods.

1. Create tables in SQL DB.



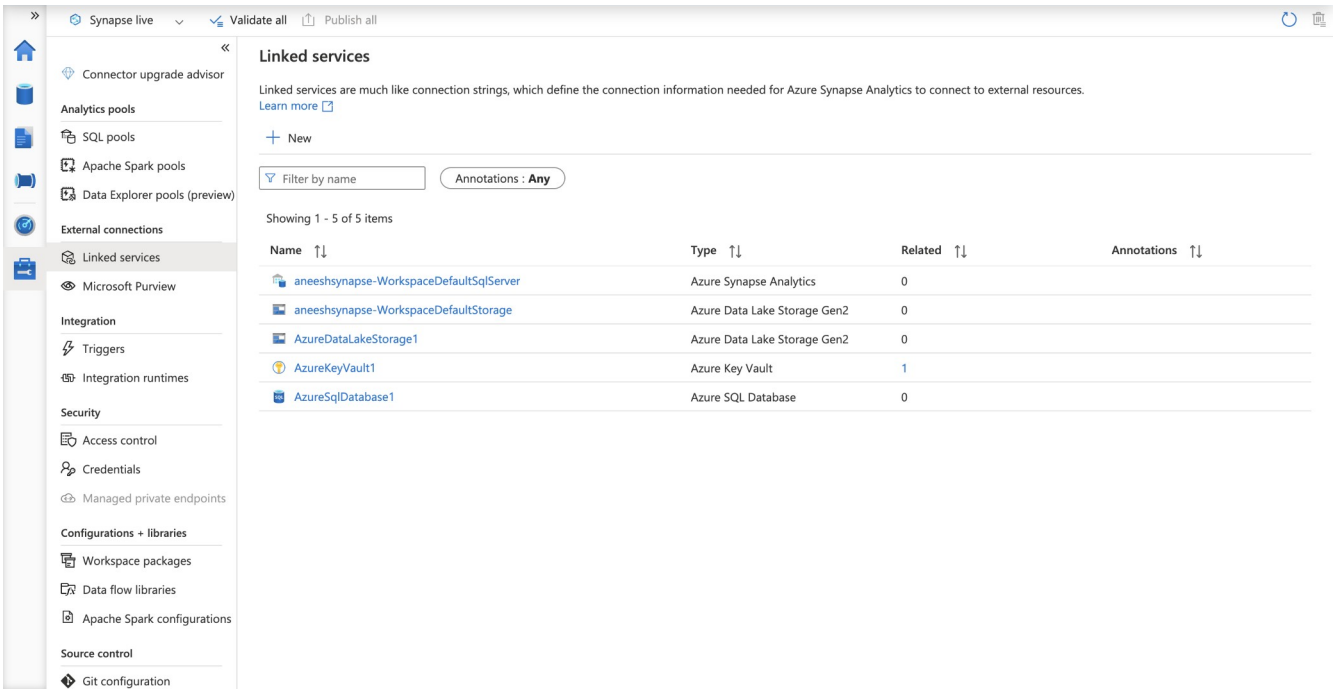


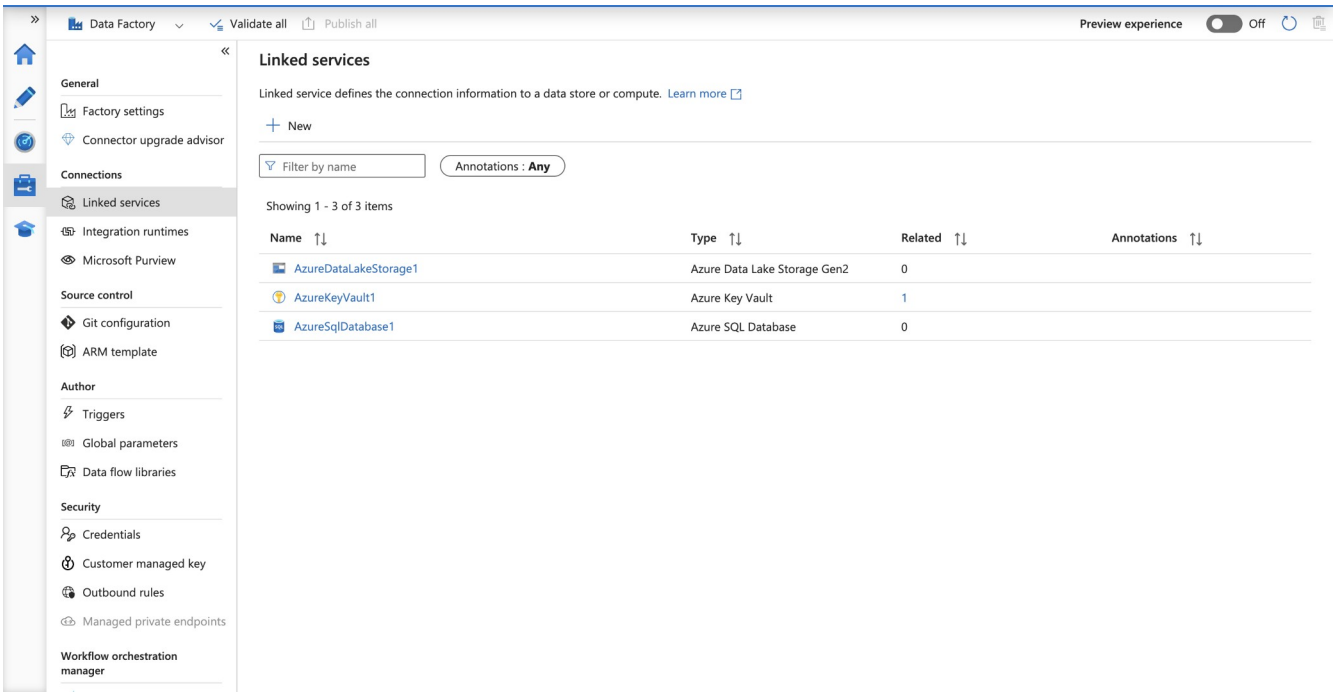
Entered some data into the tables.



2. Created linked service:

Created linked services needed for the pipeline in data factory and the same linked service and datasets in synapse analytics workspace also because while migrating the linked services should be the same otherwise will throw error. Integrated key vault while linking storage accounts and sql database.

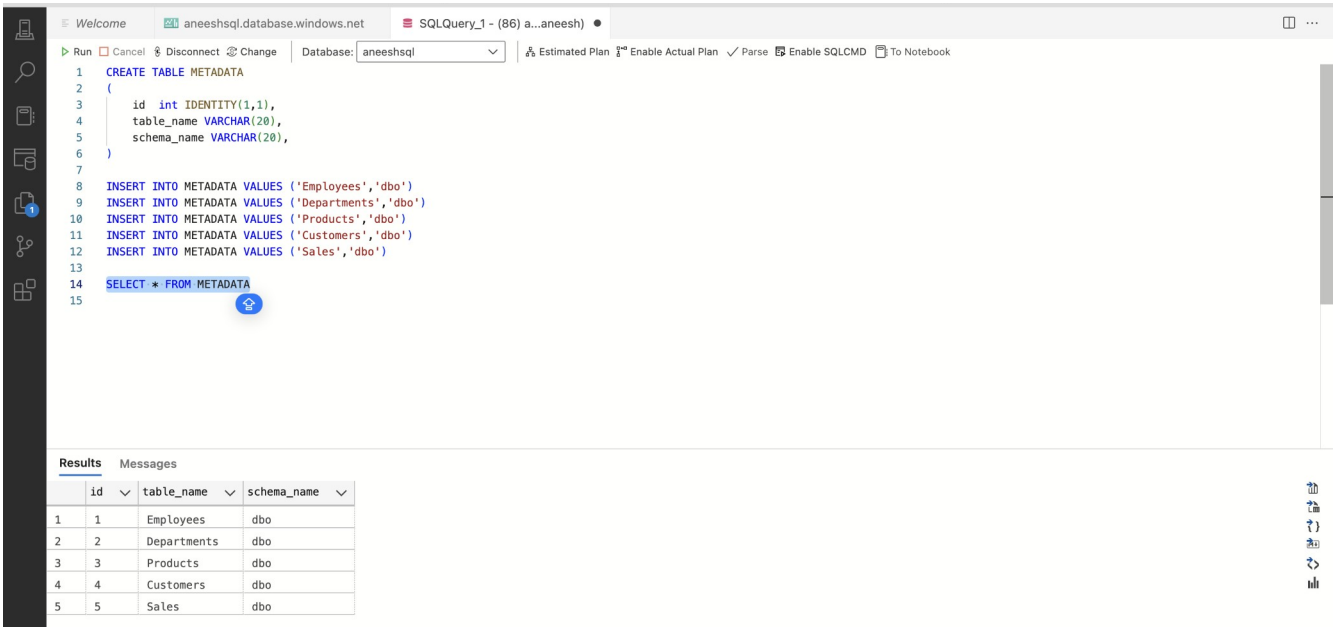




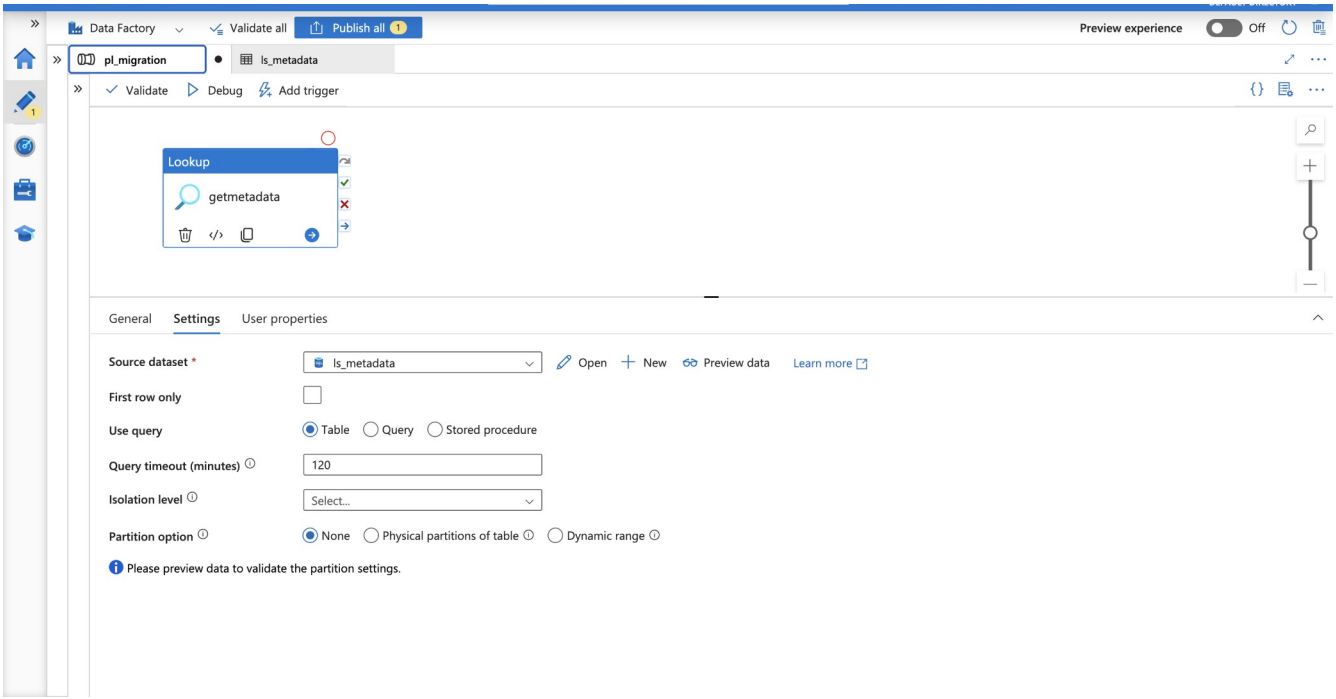
3. Creating a pipeline:

I created a pipeline where I am copying multiple tables I have created in my database to storage account as csv files. For this I use a lookup, for each and copy data activity and after copying I clean the data using the data flow and load it back to storage account silver container. Since I am using lookup activity to incrementally load my tables I also created a metadata table.

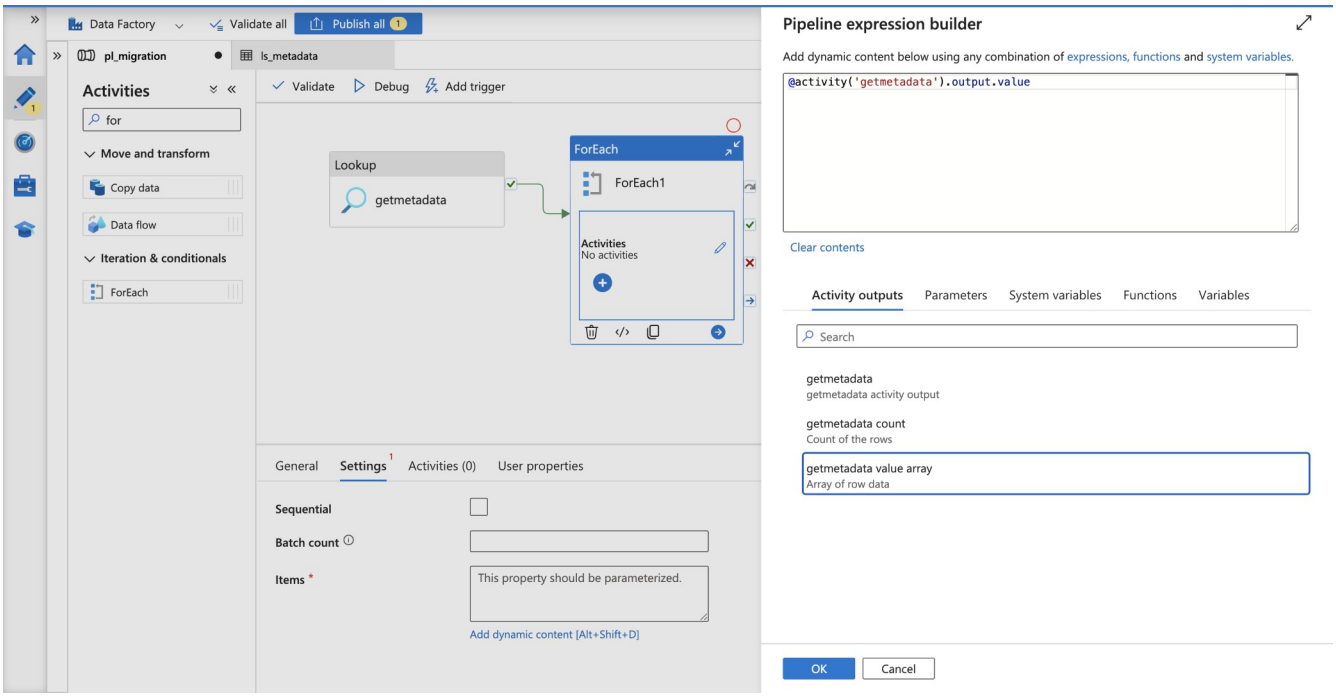
Metadata table creation.

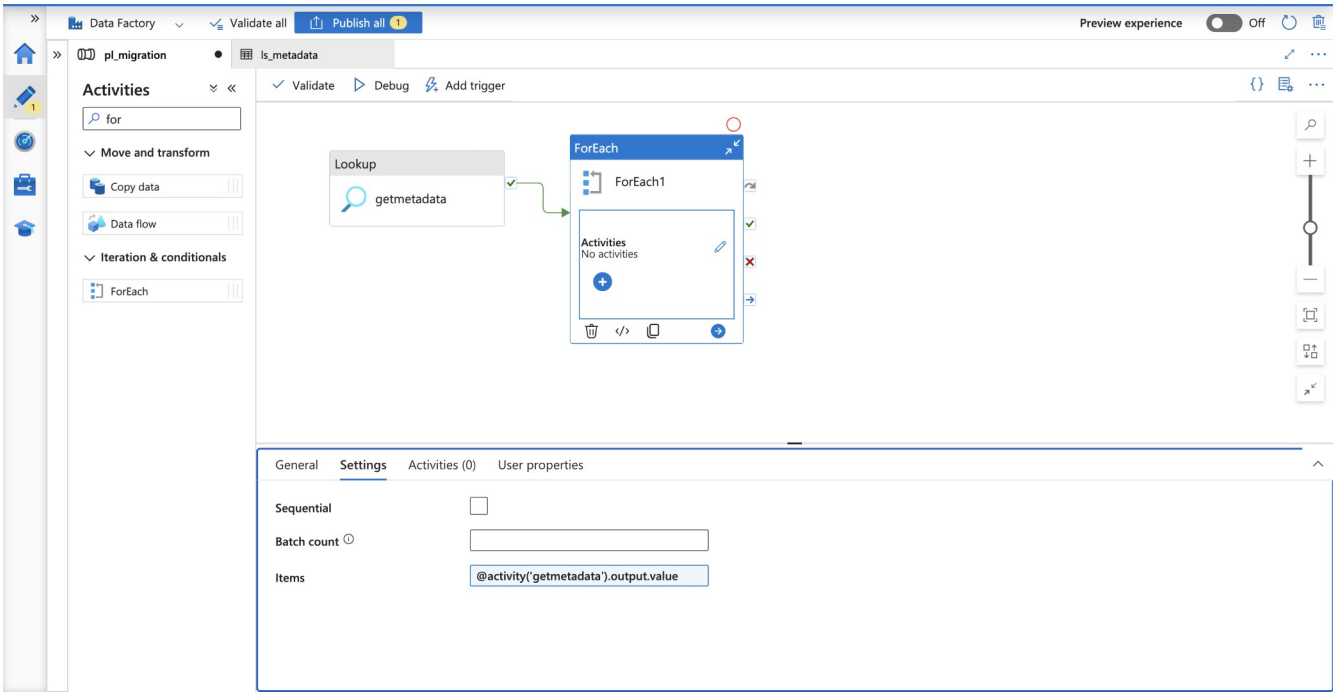


Added lookup activity in and loaded the metadata table into the lookup activity to load the tables incrementally.

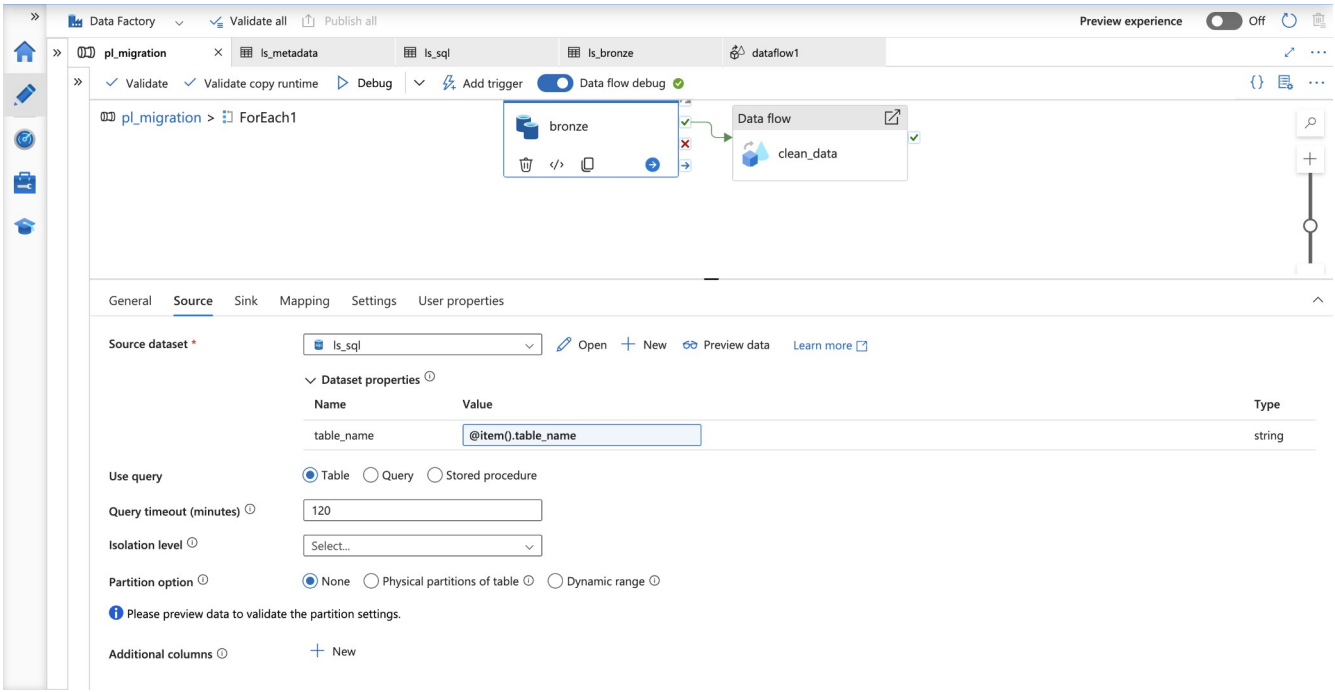


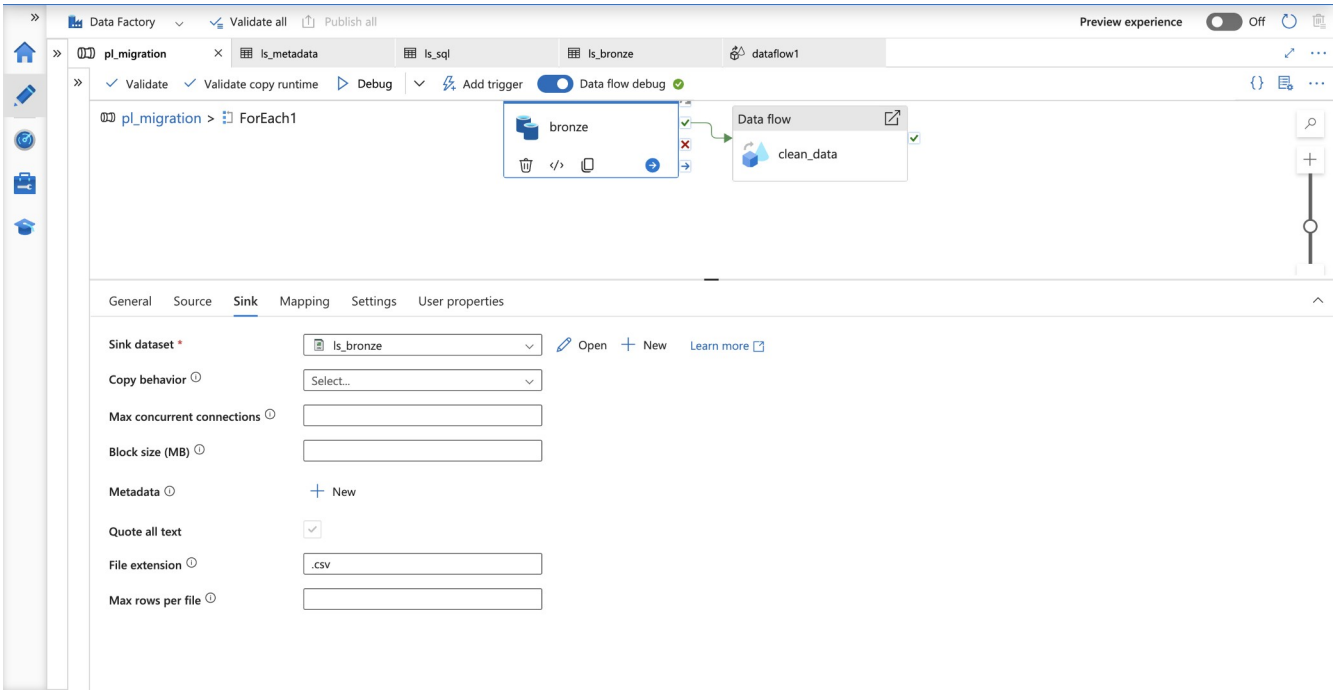
Next I added a for each to iterate through the tables and inside that I added a copy data activity to copy the tables and load them as csv files in the bronze container.



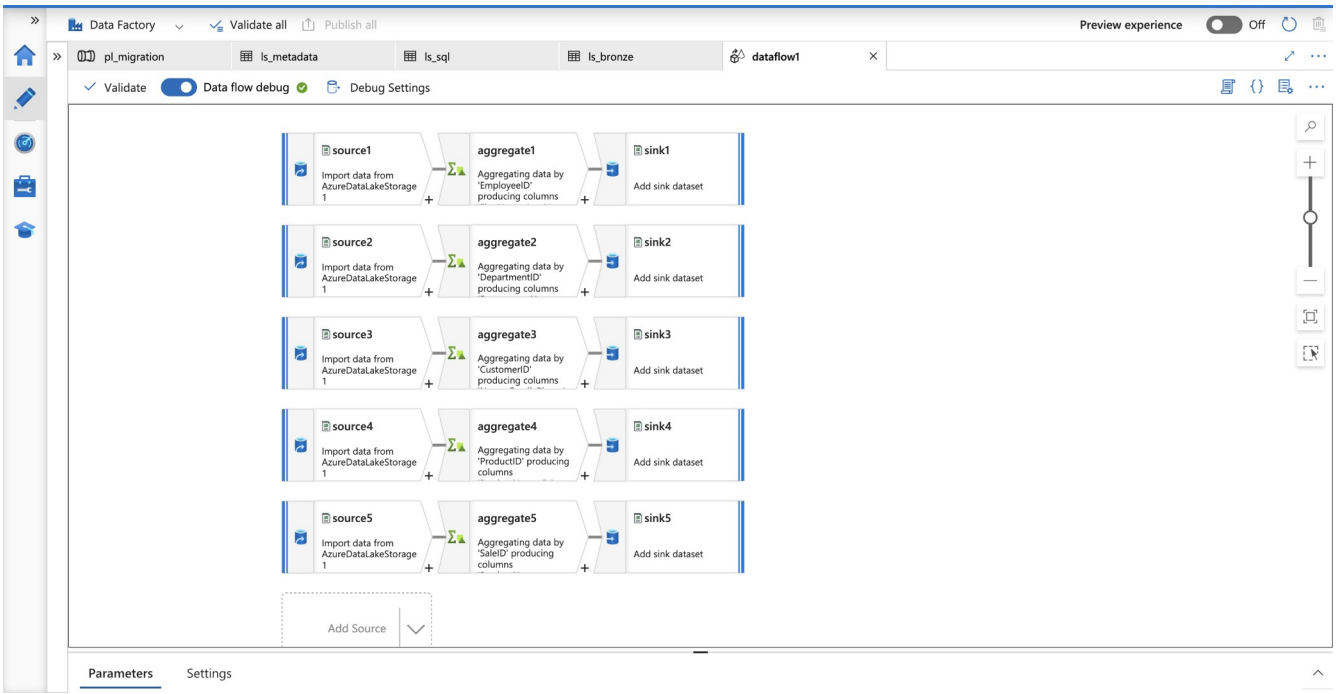


Now inside the for each I added a copy data activity to copy the tables into csv files in storage account bronze layer.

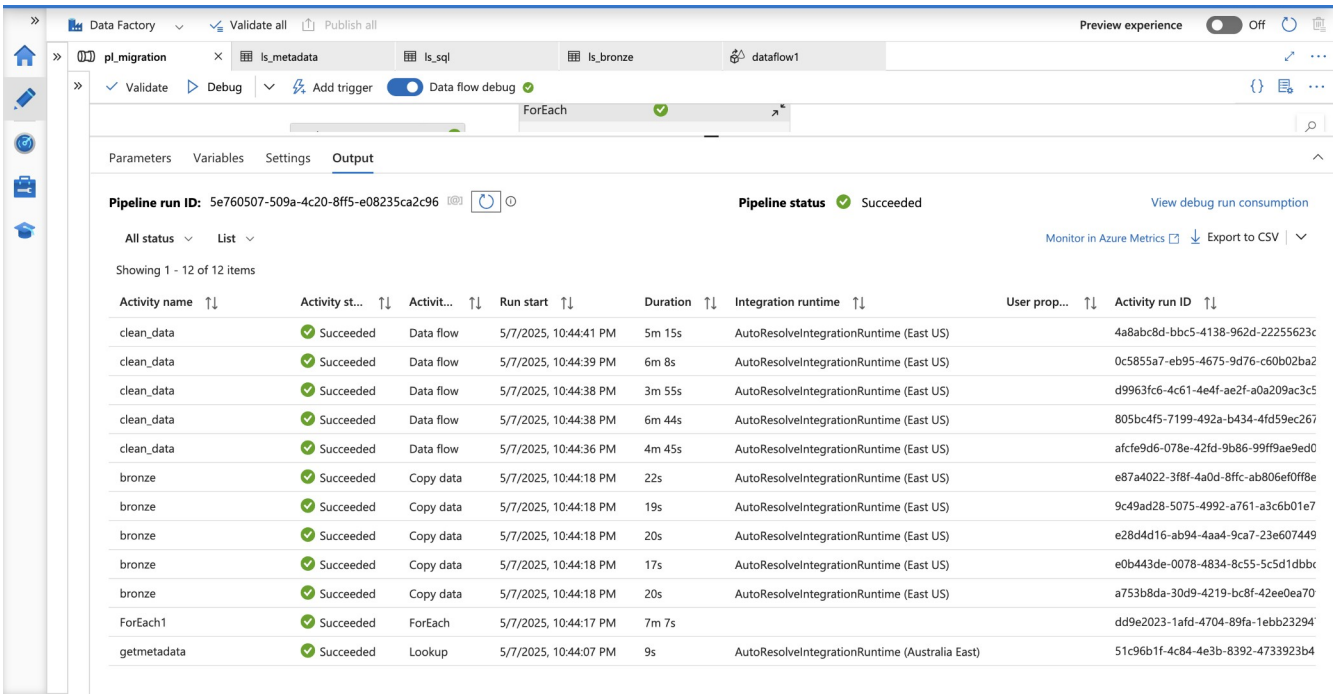




Then I added a data flow activity to clean my data and load it into silver container in my storage account.

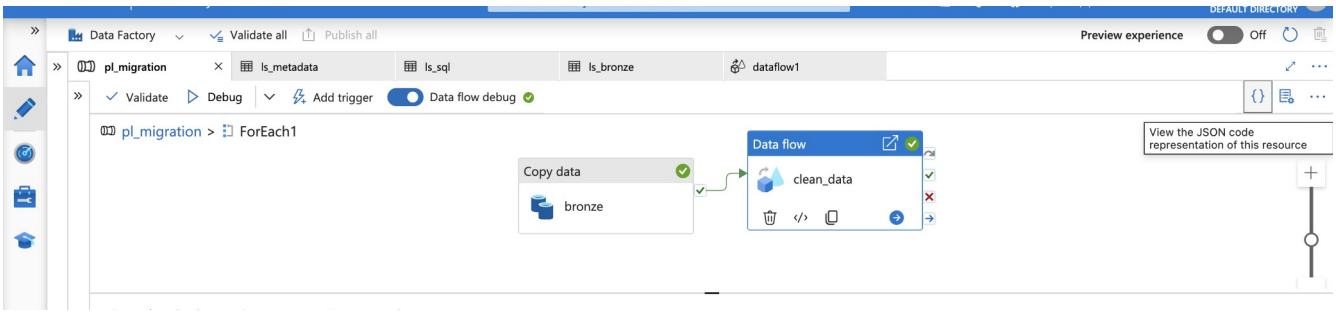


Then I run my pipeline and it was successful.

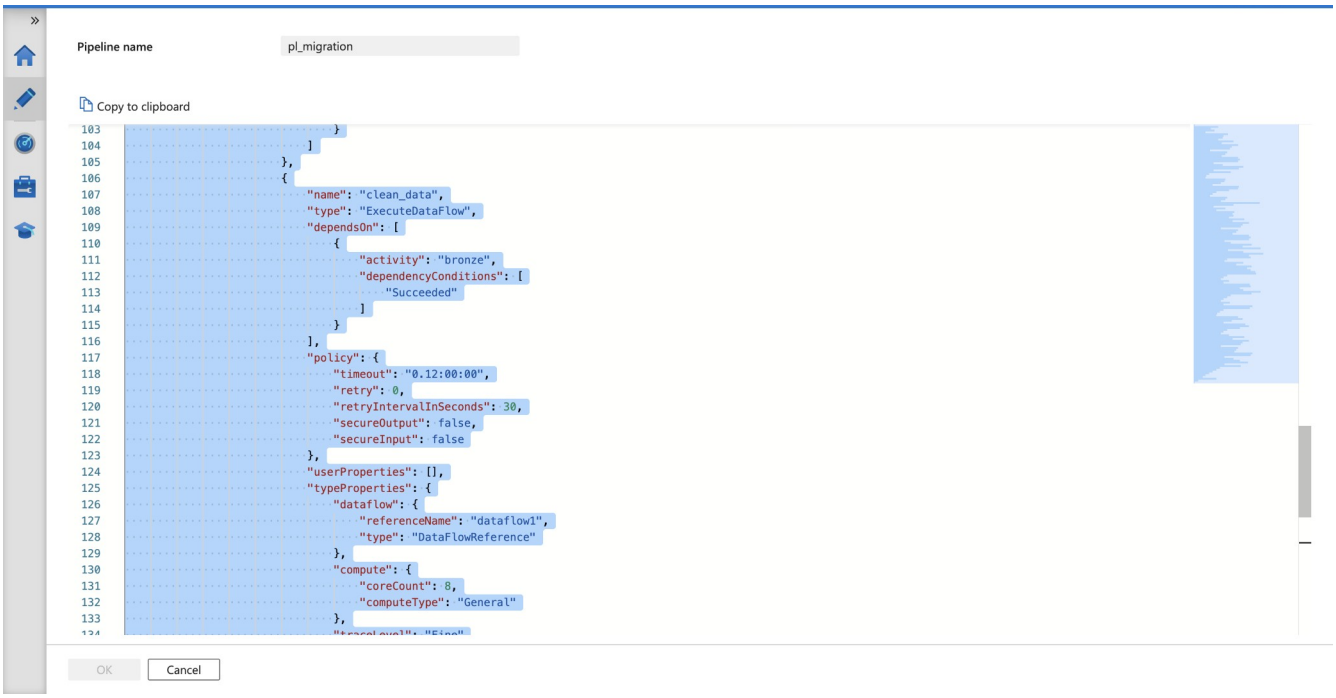


4. Migrating the pipeline to synapse

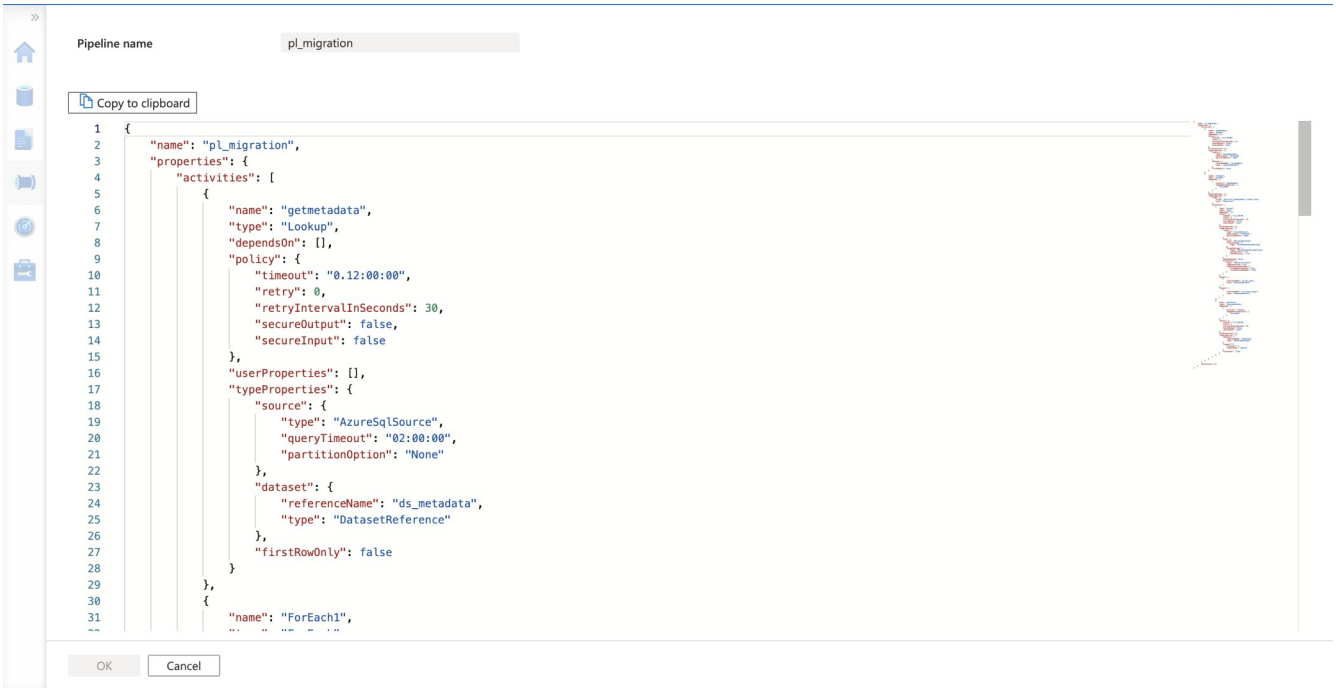
Now I want to migrate my pipeline into synapse and run it in synapse for that I copied the json code of the pipeline from data factory and pasted it in synapse pipeline as follows:

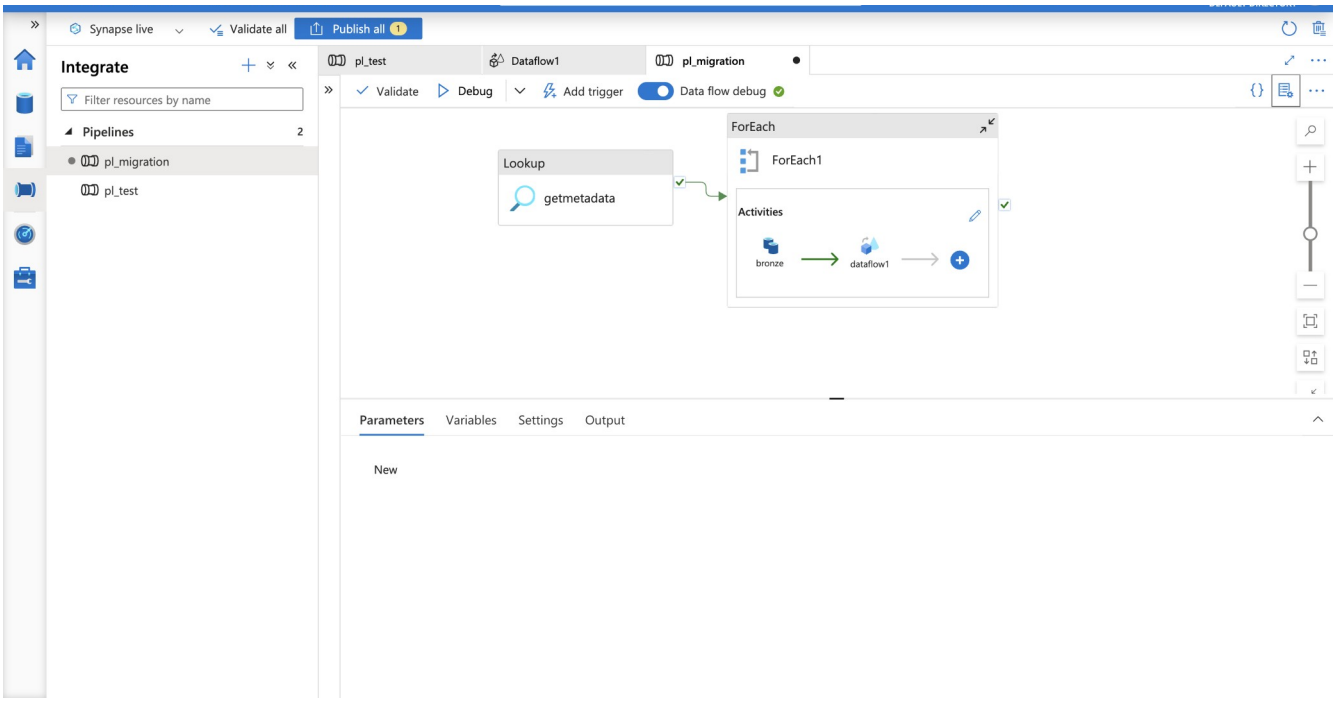


Click on the java bracket option on the right top and copy the code.

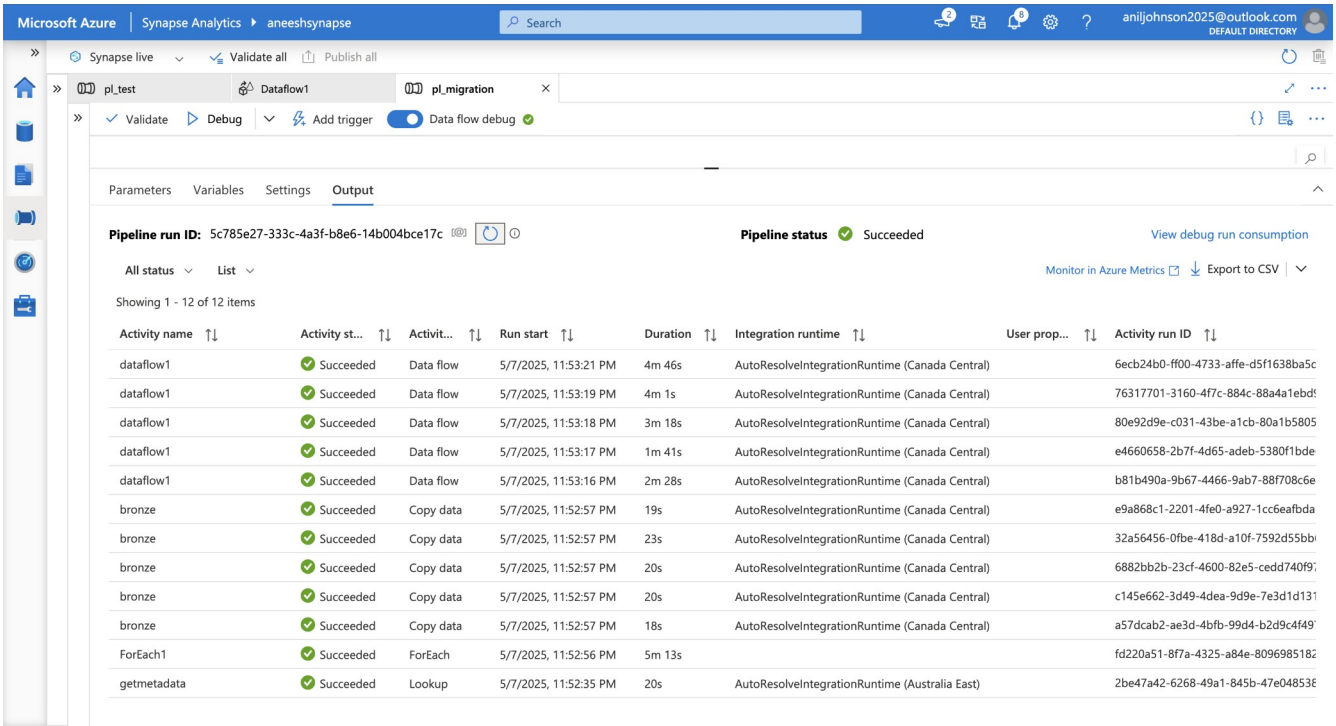


Paste the code in the new pipeline in synapse.





Published the pipeline and run the pipeline successfully.



Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime	User prop...	Activity run ID
dataflow1	Succeeded	Data flow	5/7/2025, 11:53:21 PM	4m 46s	AutoResolveIntegrationRuntime (Canada Central)		6ecb24b0-ff00-4733-affe-d5f1638ba5c
dataflow1	Succeeded	Data flow	5/7/2025, 11:53:19 PM	4m 1s	AutoResolveIntegrationRuntime (Canada Central)		76317701-3160-4f7c-884c-88a4a1ebd
dataflow1	Succeeded	Data flow	5/7/2025, 11:53:18 PM	3m 18s	AutoResolveIntegrationRuntime (Canada Central)		80e92d9e-c031-43be-a1cb-80a1b5805
dataflow1	Succeeded	Data flow	5/7/2025, 11:53:17 PM	1m 41s	AutoResolveIntegrationRuntime (Canada Central)		e4660658-2b7f-4d65-adeb-5380f1bde
dataflow1	Succeeded	Data flow	5/7/2025, 11:53:16 PM	2m 28s	AutoResolveIntegrationRuntime (Canada Central)		b81b490a-9b67-4466-9ab7-88f708c6e
bronze	Succeeded	Copy data	5/7/2025, 11:52:57 PM	19s	AutoResolveIntegrationRuntime (Canada Central)		e9a868c1-2201-4fe0-a927-1cc6eafbda
bronze	Succeeded	Copy data	5/7/2025, 11:52:57 PM	23s	AutoResolveIntegrationRuntime (Canada Central)		32a56456-0fbe-418d-a10f-7592d55bb
bronze	Succeeded	Copy data	5/7/2025, 11:52:57 PM	20s	AutoResolveIntegrationRuntime (Canada Central)		6882bb2b-23cf-4600-82e5-cedd740f9
bronze	Succeeded	Copy data	5/7/2025, 11:52:57 PM	20s	AutoResolveIntegrationRuntime (Canada Central)		c145e662-3d49-4dea-9d9e-7e3d1d131
bronze	Succeeded	Copy data	5/7/2025, 11:52:57 PM	18s	AutoResolveIntegrationRuntime (Canada Central)		a57dcab2-ae3d-4bfb-99d4-b2d9c4f49
ForEach1	Succeeded	ForEach	5/7/2025, 11:52:56 PM	5m 13s			fd220a51-8f7a-4325-a84e-8096985182
getmetadata	Succeeded	Lookup	5/7/2025, 11:52:35 PM	20s	AutoResolveIntegrationRuntime (Australia East)		2be47a42-6268-49a1-845b-47e048538

Thus successfully migrated pipeline created in data factory to synapse.
Steps taken to troubleshoot the errors faced:

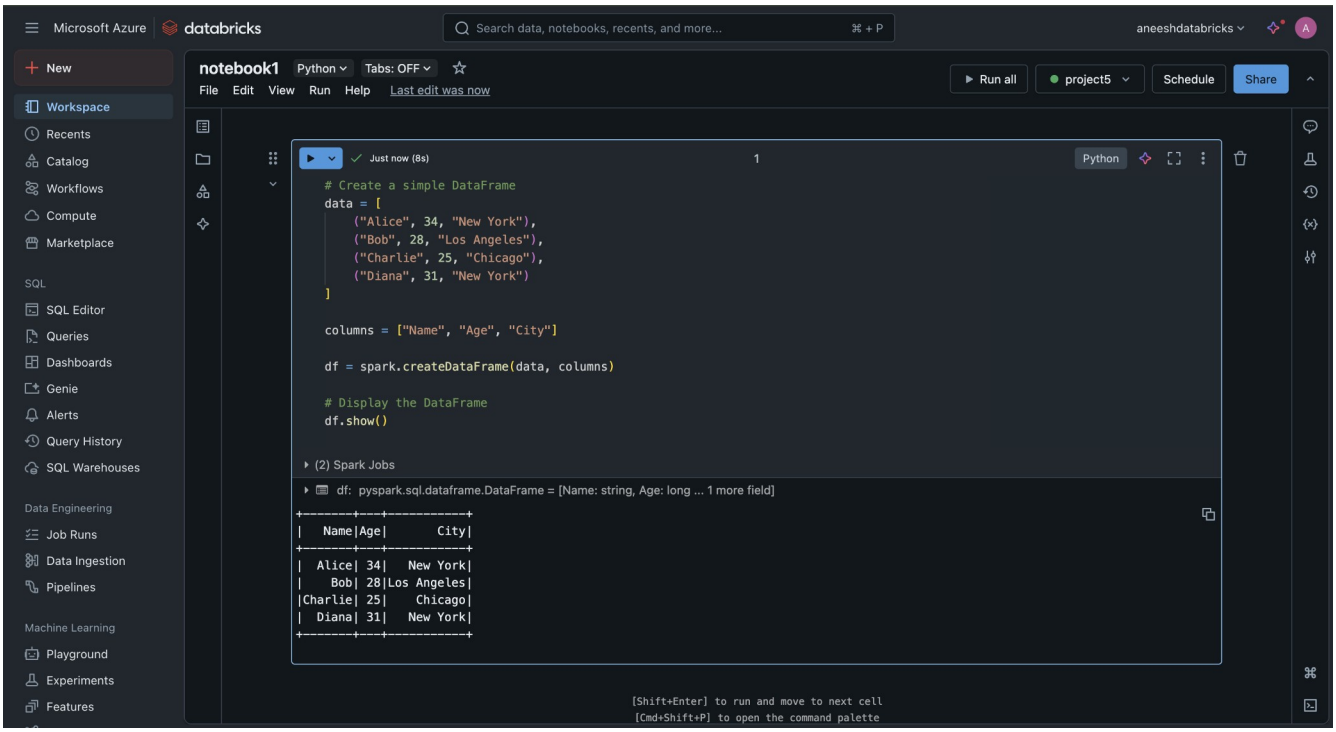
- Fixed broken resource references in the JSON
- Recreated or renamed datasets and the Data Flow

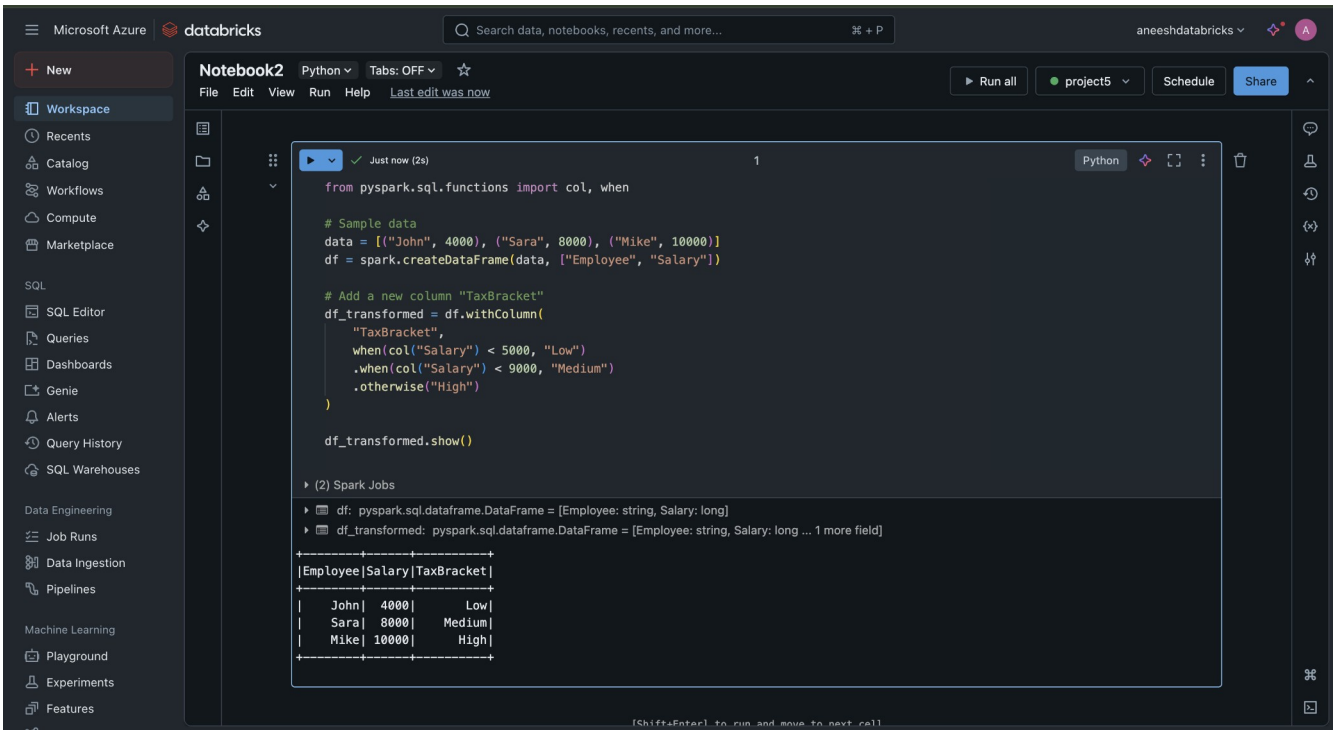
- Enabled the Debug session for design-time testing
- Verified the pipeline executes successfully

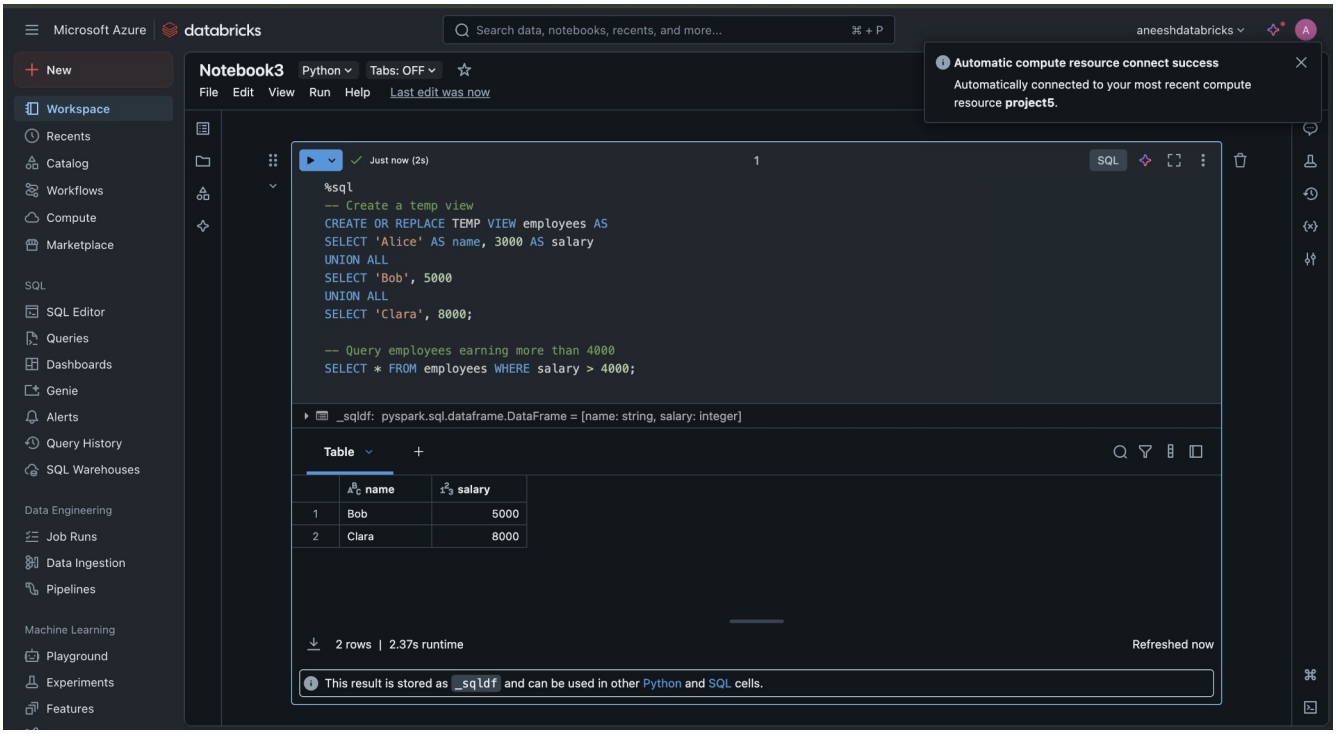
TASK 2:MIGRATING DATABRICKS NOTEBOOKS TO FABRIC NOTEBOOKS

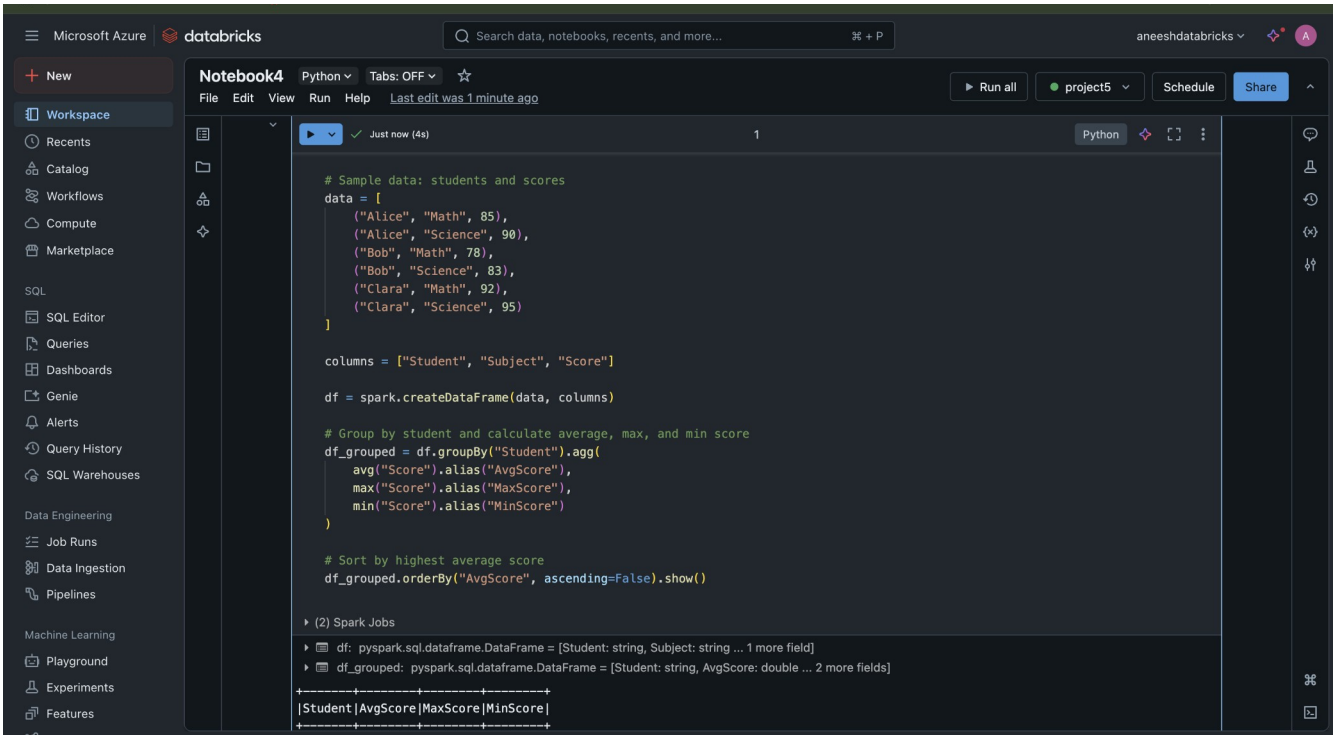
1. Create notebooks in databricks.

I created 4 notebooks in databricks with some random activities in it.



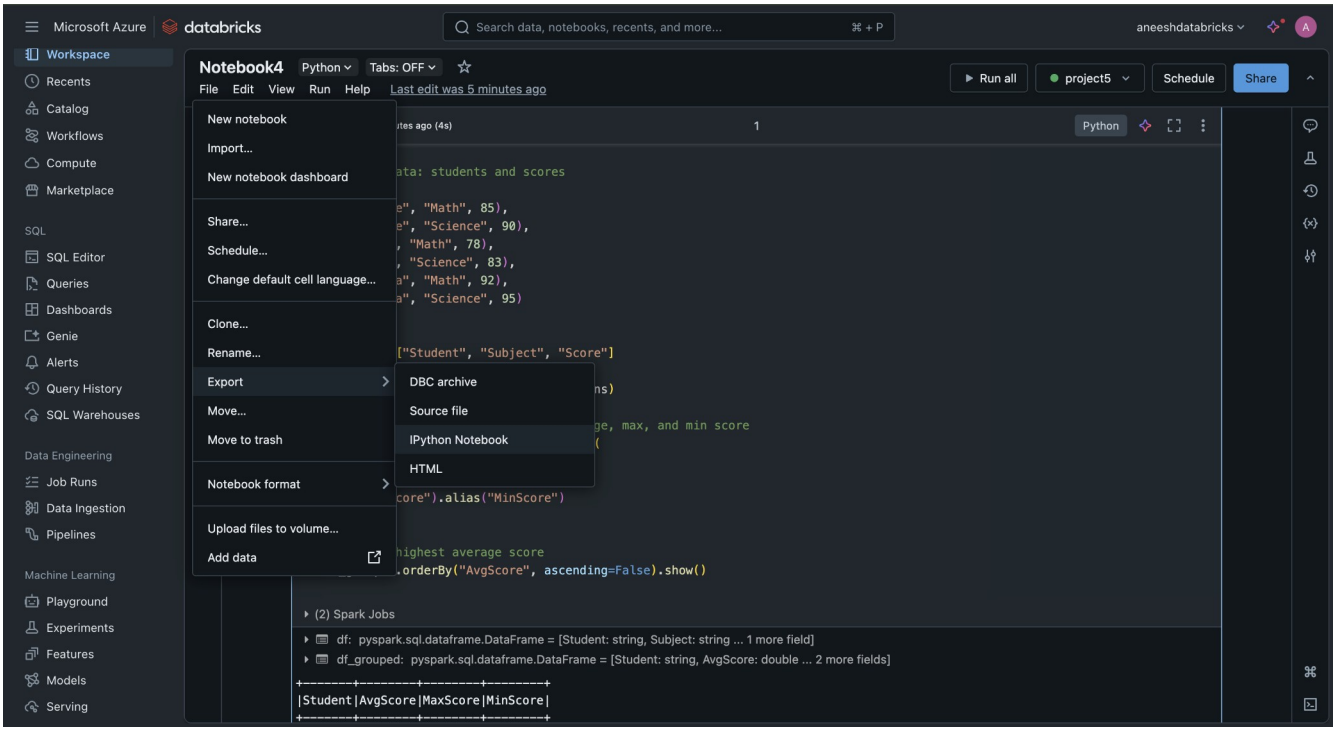

Microsoft Azure
databricks
Search data, notebooks, recents, and more...
aneeshdatabricks
New
Notebook2
Python
Tabs: OFF
File Edit View Run Help
Last edit was now
Run all
project5
Schedule
Share
Workspace
Recents
Catalog
Workflows
Compute
Marketplace
SQL
SQL Editor
Queries
Dashboards
Genie
Alerts
Query History
SQL Warehouses
Data Engineering
Job Runs
Data Ingestion
Pipelines
Machine Learning
Playground
Experiments
Features
Just now (2s)
1
Python
from pyspark.sql.functions import col, when
# Sample data
data = [("John", 4000), ("Sara", 8000), ("Mike", 10000)]
df = spark.createDataFrame(data, ["Employee", "Salary"])
# Add a new column "TaxBracket"
df_transformed = df.withColumn(
"TaxBracket",
when(col("Salary") < 5000, "Low")
.when(col("Salary") < 9000, "Medium")
.otherwise("High")
)
df_transformed.show()
(2) Spark Jobs
df: pyspark.sql.dataframe.DataFrame = [Employee: string, Salary: long]
df_transformed: pyspark.sql.dataframe.DataFrame = [Employee: string, Salary: long ... 1 more field]
+--------+------+----------+
|Employee|Salary|TaxBracket|
+--------+------+----------+
John	4000	Low
Sara	8000	Medium
Mike	10000	High
+--------+------+----------+


Microsoft Azure
databricks
Search data, notebooks, recents, and more...
aneeshdatabricks
Automatic compute resource connect success
Automatically connected to your most recent compute resource project5.
New
Notebook3
Python
Tabs: OFF
File Edit View Run Help
Last edit was now
Workspace
Recents
Catalog
Workflows
Compute
Marketplace
SQL
SQL Editor
Queries
Dashboards
Genie
Alerts
Query History
SQL Warehouses
Data Engineering
Job Runs
Data Ingestion
Pipelines
Machine Learning
Playground
Experiments
Features
Just now (2s)
1
SQL
%sql
-- Create a temp view
CREATE OR REPLACE TEMP VIEW employees AS
SELECT 'Alice' AS name, 3000 AS salary
UNION ALL
SELECT 'Bob', 5000
UNION ALL
SELECT 'Clara', 8000;
-- Query employees earning more than 4000
SELECT * FROM employees WHERE salary > 4000;
_sqldf: pyspark.sql.dataframe.DataFrame = [name: string, salary: integer]
Table
name
salary
1 Bob 5000
2 Clara 8000
2 rows | 2.37s runtime
Refreshed now
This result is stored as _sqldf and can be used in other Python and SQL cells.



2, Export the notebooks

Goto File → Export → iPython notebook

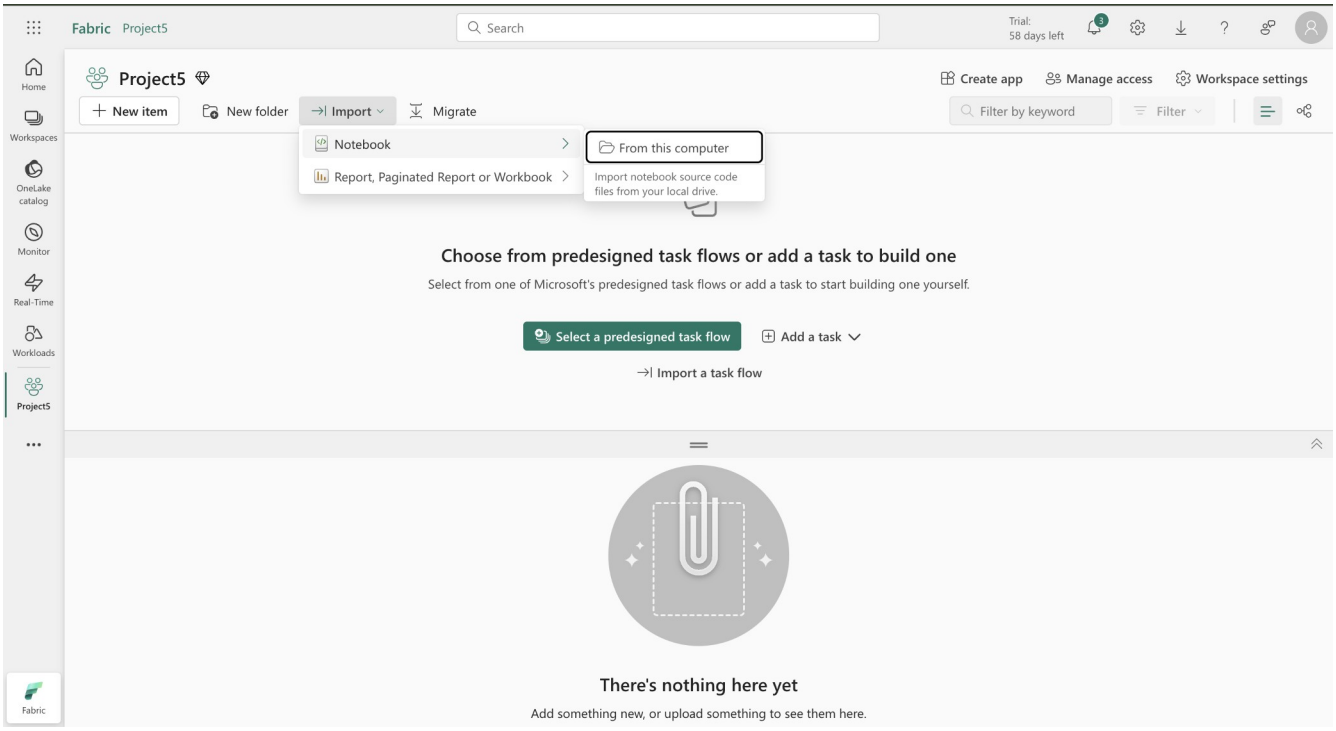


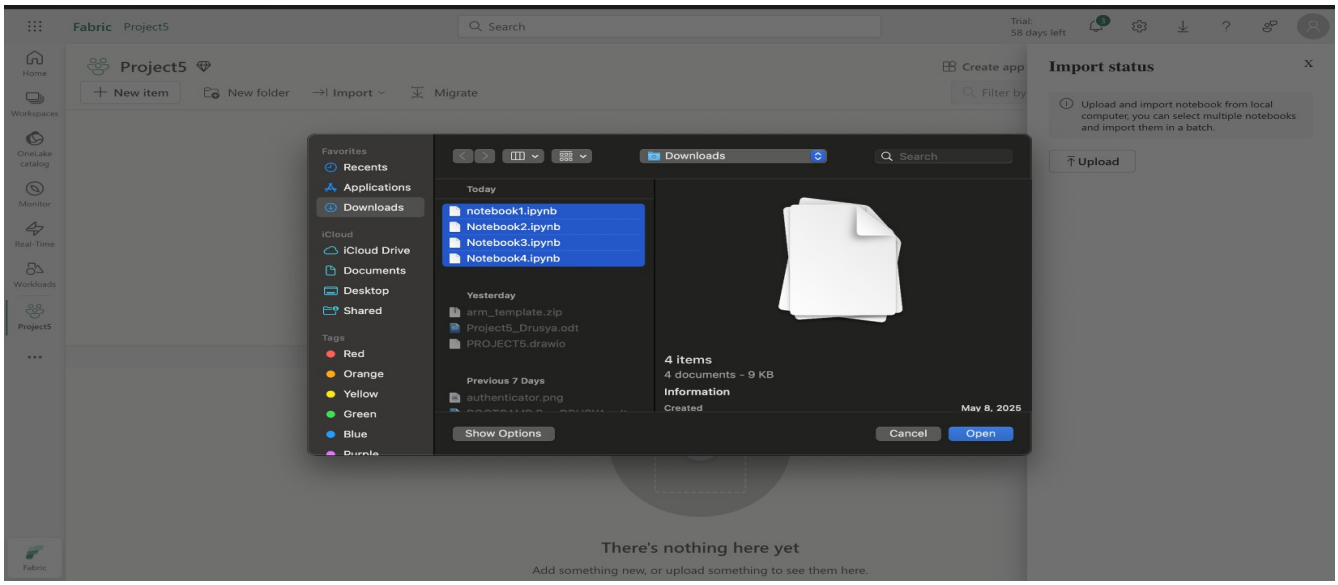
Similarly done for all the notebooks.

3. Import the notebooks in Fabric.

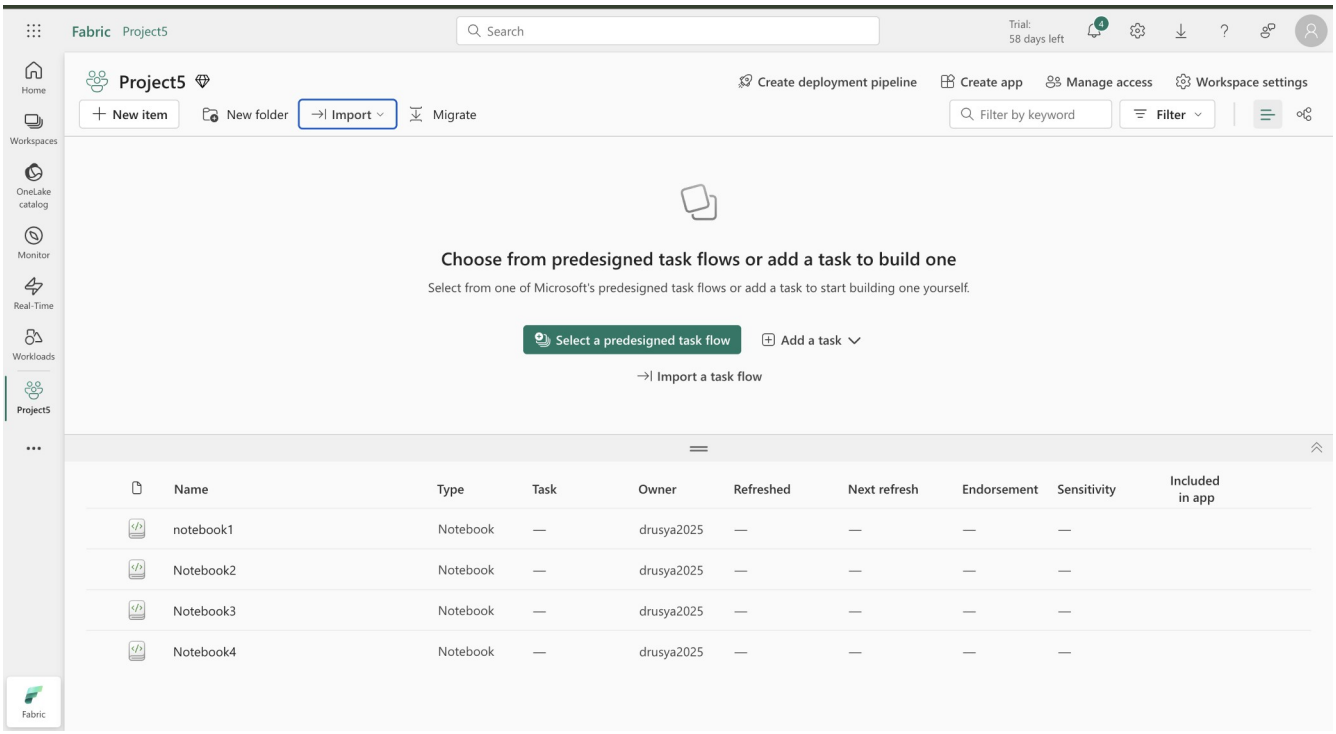
For importing the notebooks to fabric I logged into fabric and created a new fabric workspace. Then imported the notebooks.

Goto Import → Import notebook → From this computer

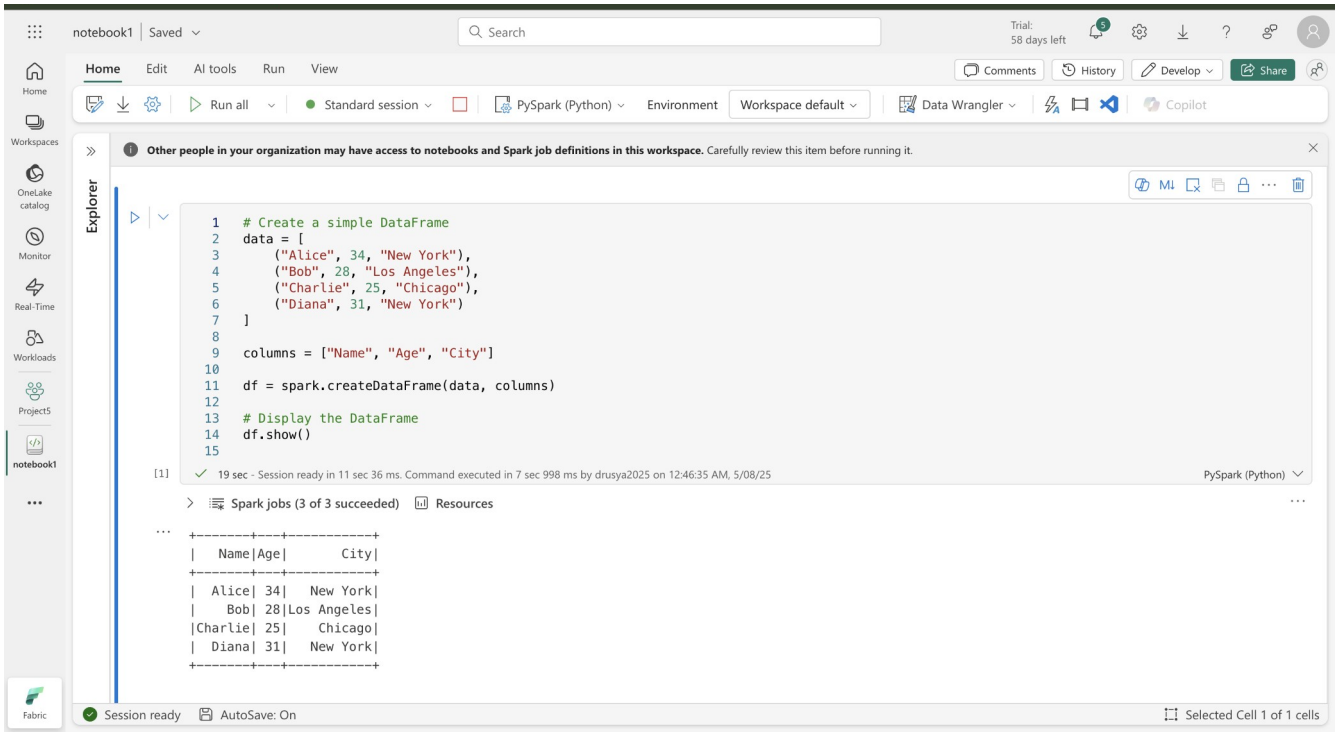


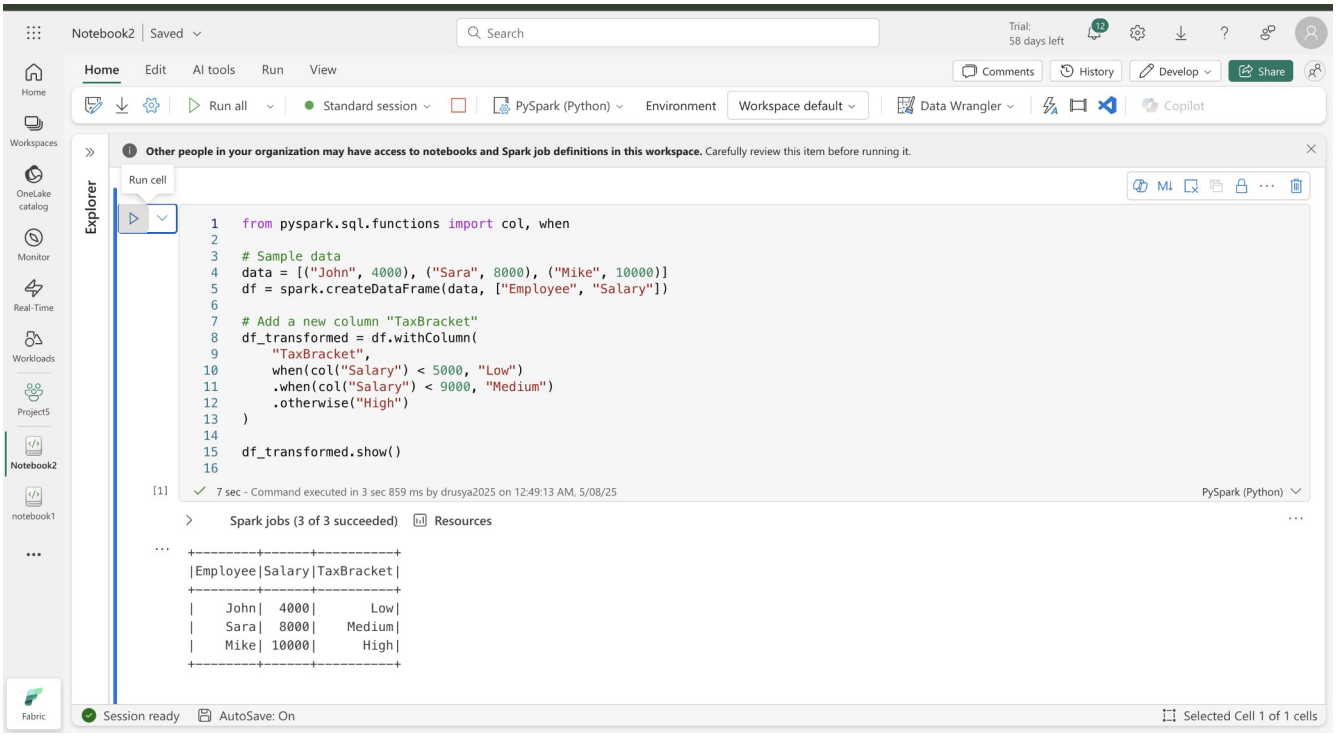


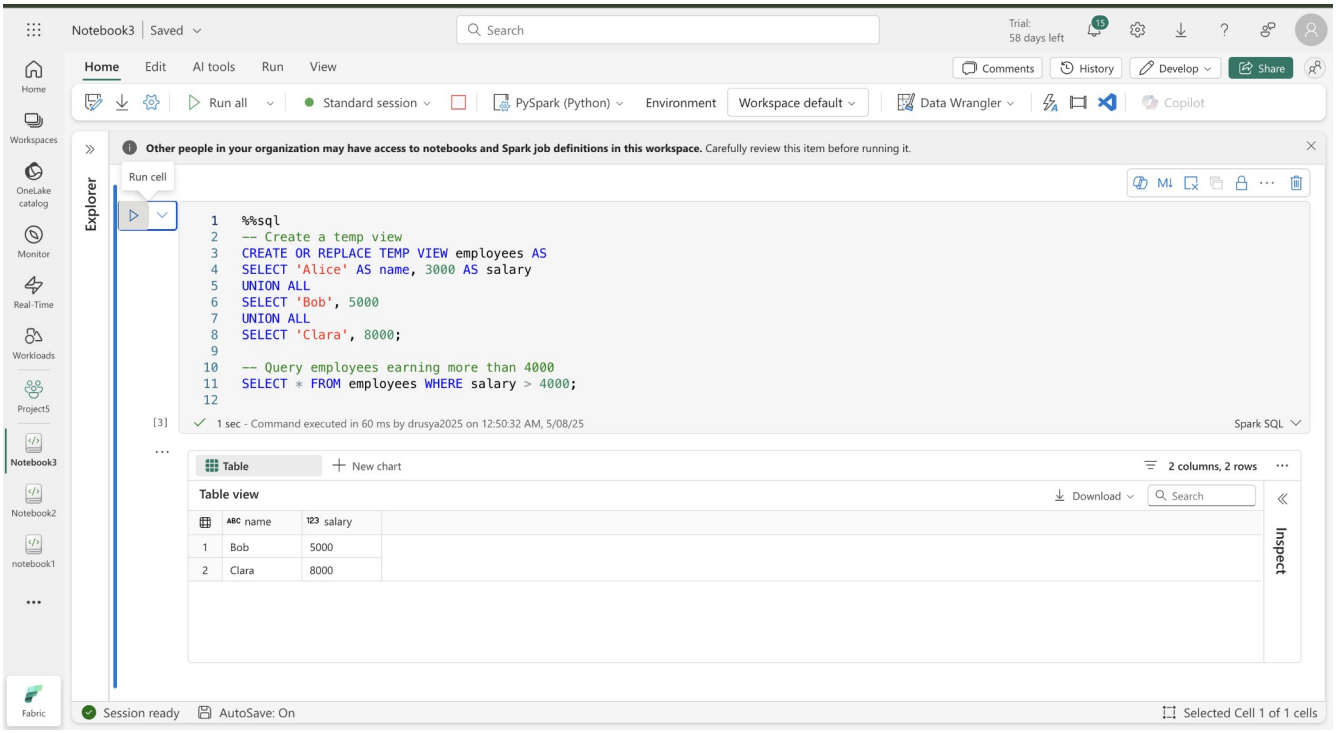
Now all notebooks have been imported to fabric workspace.

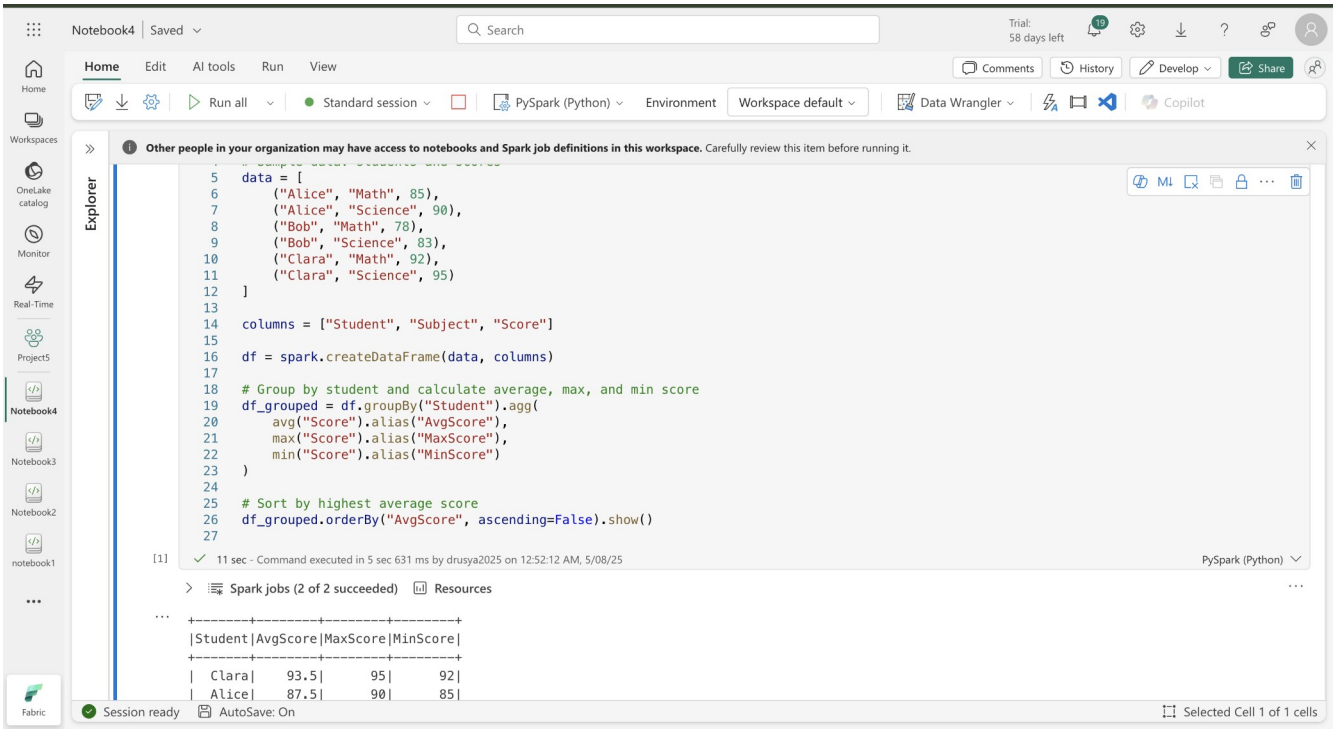


Now opened the notebooks and run the commands it has run successfully.




Notebook2 | Saved
Search
Trial: 58 days left
Home Edit AI tools Run View
Comments History Develop Share
Run all Standard session PySpark (Python) Environment Workspace default Data Wrangler Copilot
Other people in your organization may have access to notebooks and Spark job definitions in this workspace. Carefully review this item before running it.
Explorer
Run cell
1 from pyspark.sql.functions import col, when
2
3 # Sample data
4 data = [("John", 4000), ("Sara", 8000), ("Mike", 10000)]
5 df = spark.createDataFrame(data, ["Employee", "Salary"])
6
7 # Add a new column "TaxBracket"
8 df_transformed = df.withColumn(
9 "TaxBracket",
10 when(col("Salary") < 5000, "Low")
11 .when(col("Salary") < 9000, "Medium")
12 .otherwise("High")
13)
14
15 df_transformed.show()
16
[1] 7 sec - Command executed in 3 sec 859 ms by drusya2025 on 12:49:13 AM, 5/08/25
PySpark (Python)
Spark jobs (3 of 3 succeeded) Resources
+--------+------+----------+
|Employee|Salary|TaxBracket|
+--------+------+----------+
John	4000	Low
Sara	8000	Medium
Mike	10000	High
+--------+------+----------+
Session ready AutoSave: On
Selected Cell 1 of 1 cells


Notebook3 | Saved
Search
Trial: 58 days left
Home Edit AI tools Run View
Comments History Develop Share
Run all Standard session PySpark (Python) Environment Workspace default Data Wrangler Copilot
Other people in your organization may have access to notebooks and Spark job definitions in this workspace. Carefully review this item before running it.
Explorer
Run cell
1 %%sql
2 -- Create a temp view
3 CREATE OR REPLACE TEMP VIEW employees AS
4 SELECT 'Alice' AS name, 3000 AS salary
5 UNION ALL
6 SELECT 'Bob', 5000
7 UNION ALL
8 SELECT 'Clara', 8000;
9
10 -- Query employees earning more than 4000
11 SELECT * FROM employees WHERE salary > 4000;
12
[3] 1 sec - Command executed in 60 ms by drusya2025 on 12:50:32 AM, 5/08/25
Spark SQL
Table New chart
2 columns, 2 rows
Table view
Download Search
name salary
1 Bob 5000
2 Clara 8000
Inspect
Session ready AutoSave: On
Selected Cell 1 of 1 cells



CONCLUSION

Thus the project successfully migrated pipeline from data factory to synapse anlytics using the json script. The project also achieved successful migration of data bricks notebooks to fabric notebooks using the export import options in the notebooks.