

# **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:

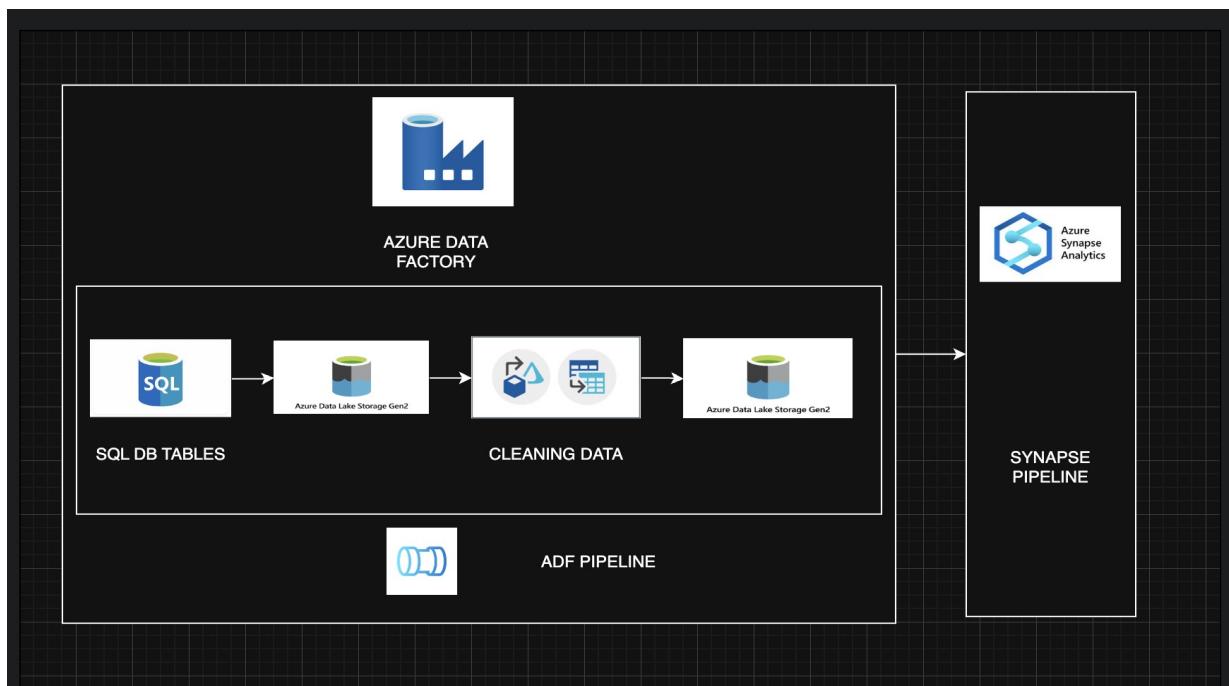
1. Create a pipeline to bring multiple tables from Azure SQL to ADLS gen2.
2. Create a pipeline to clean the data using Dataflows.
3. Migrate the above developed pipelines from ADF to synapse.
4. Test the pipelines after changing the connections.

## **TASK 2:**

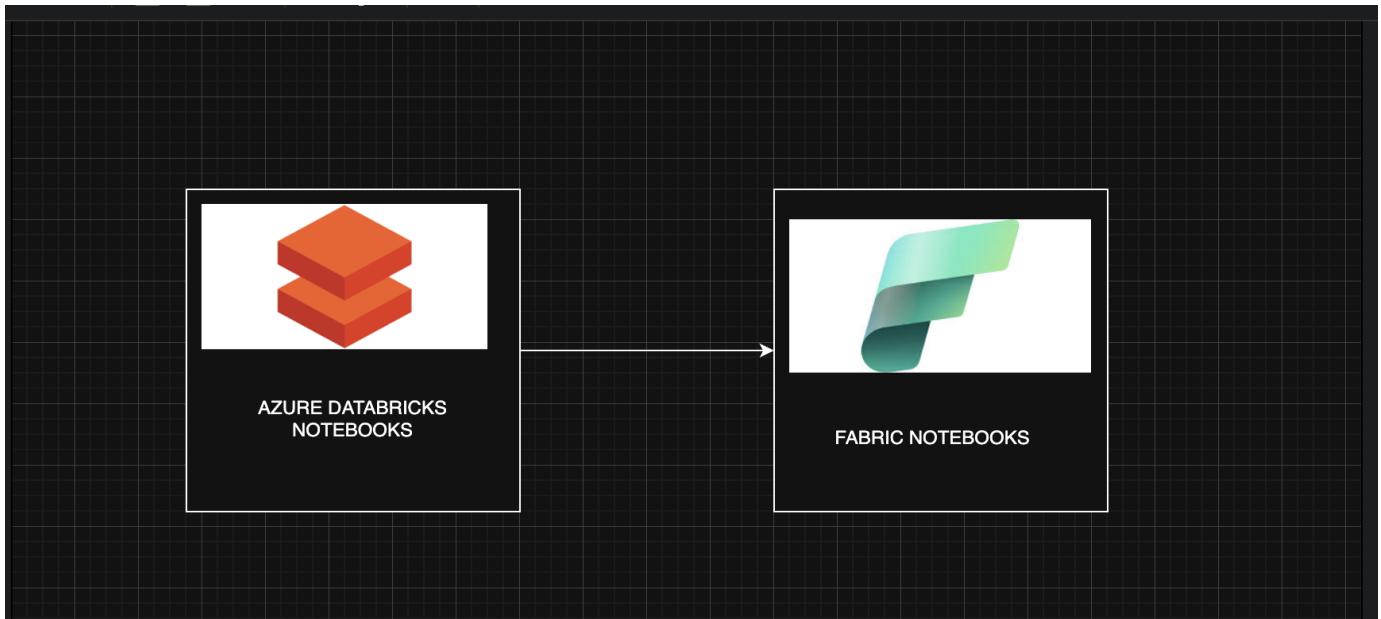
1. Create four Notebooks in Databricks.
2. Migrate these Notebooks from Databricks to Fabric.
3. Change connections and Test the Notebooks

# **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.

```
1 -- 1. Employees Table
2 CREATE TABLE Employees (
3     EmployeeID INT PRIMARY KEY IDENTITY(1,1),
4     FirstName NVARCHAR(50),
5     LastName NVARCHAR(50),
6     HireDate DATE
7 );
8
9 -- 2. Departments Table
10 CREATE TABLE Departments (
11     DepartmentID INT PRIMARY KEY IDENTITY(1,1),
12     DepartmentName NVARCHAR(100),
13     Location NVARCHAR(100)
14 );
15
16 -- 3. Products Table
17 CREATE TABLE Products (
18     ProductID INT PRIMARY KEY IDENTITY(1,1),
19     ProductName NVARCHAR(100),
20     Price DECIMAL(10, 2),
21     Stock INT
22 );
23
24 -- 4. Customers Table
25 CREATE TABLE Customers (
26     CustomerID INT PRIMARY KEY IDENTITY(1,1),
27     Name NVARCHAR(100),
28     Email NVARCHAR(100),
29     Phone NVARCHAR(20)
30 );
31
32 -- 5. Sales Table
33 CREATE TABLE Sales (
34     SaleID INT PRIMARY KEY IDENTITY(1,1),
35     ProductName NVARCHAR(100),
36     QuantitySold INT,
37     SaleDate DATE
38 );
39
40
41
```

```
40 SELECT * FROM Employees
41 SELECT * FROM Departments
42 SELECT * FROM Products
43 SELECT * FROM Customers
44 SELECT * FROM Sales
45
```

Results Messages

EmployeeID	FirstName	LastName	HireDate
------------	-----------	----------	----------

DepartmentID	DepartmentName	Location
--------------	----------------	----------

ProductID	ProductName	Price	Stock
-----------	-------------	-------	-------

CustomerID	Name	Email	Phone
------------	------	-------	-------

SaleID	ProductName	QuantitySold	SaleDate
--------	-------------	--------------	----------

Ln 44, Col 21 (117 selected) Spaces: 4 UTF-8 LF 0 rows MSSQL 00:00:00 aneeshsql.database.windows.net : aneeshsql (93)

Entered some data into the tables.

```
1  INSERT INTO Employees (FirstName, LastName, HireDate)
2  VALUES
3  ('Alice', 'Johnson', '2020-03-15'),
4  ('Bob', 'Smith', '2019-07-01'),
5  ('Carol', 'Taylor', '2021-01-10');
6
7  INSERT INTO Departments (DepartmentName, Location)
8  VALUES
9  ('Human Resources', 'New York'),
10 ('Engineering', 'San Francisco'),
11 ('Sales', 'Chicago');
12
13  INSERT INTO Products (ProductName, Price, Stock)
14  VALUES
15  ('Laptop', 899.99, 50),
16  ('Headphones', 49.99, 150),
17  ('Smartphone', 599.99, 30);
18
19  INSERT INTO Customers (Name, Email, Phone)
20  VALUES
21  ('David Brown', 'david.brown@example.com', '123-456-7890'),
22  ('Emma Wilson', 'emma.wilson@example.com', '987-654-3210'),
23  ('Liam Davis', 'liam.davis@example.com', '555-123-4567');
24
25  INSERT INTO Sales (ProductName, QuantitySold, SaleDate)
26  VALUES
27  ('Laptop', 5, '2024-05-01'),
28  ('Smartphone', 3, '2024-05-02'),
29  ('Headphones', 10, '2024-05-03');
```

## 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.

The screenshot shows the Azure Synapse Analytics workspace settings page. On the left, there's a navigation sidebar with various options like 'Synapse live', 'Analytics pools', 'External connections', 'Microsoft Purview', 'Integration', 'Triggers', 'Integration runtimes', 'Security', 'Access control', 'Credentials', 'Managed private endpoints', 'Configurations + libraries', 'Workspace packages', 'Data flow libraries', 'Apache Spark configurations', 'Source control', and 'Git configuration'. The 'External connections' section is currently selected. On the right, under 'Linked services', it says 'Linked services are much like connection strings, which define the connection information needed for Azure Synapse Analytics to connect to external resources.' Below this, there's a 'New' button and a search/filter bar. A table lists five existing linked services:

Name	Type	Related	Annotations
aneeshsynapse-WorkspaceDefaultSqlServer	Azure Synapse Analytics	0	
aneeshsynapse-WorkspaceDefaultStorage	Azure Data Lake Storage Gen2	0	
AzureDataLakeStorage1	Azure Data Lake Storage Gen2	0	
AzureKeyVault1	Azure Key Vault	1	
AzureSqlDatabase1	Azure SQL Database	0	

### 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.

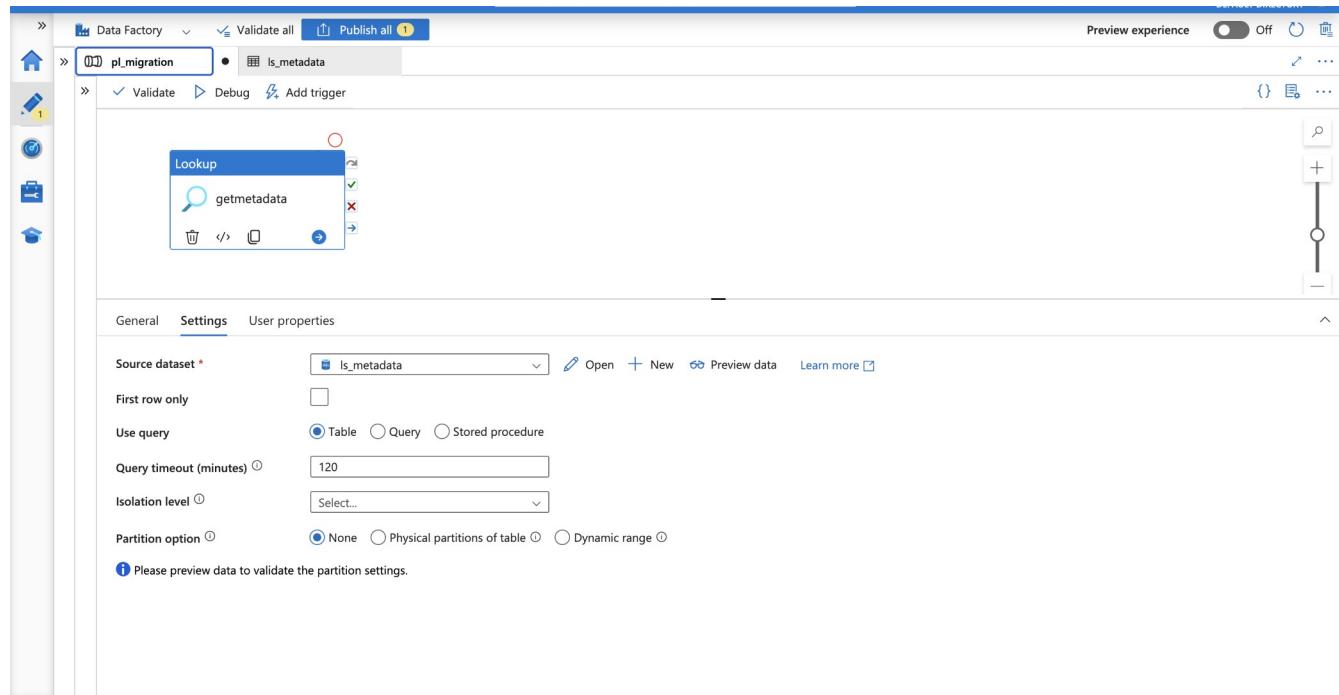
```

CREATE TABLE METADATA
(
    id int IDENTITY(1,1),
    table_name VARCHAR(20),
    schema_name VARCHAR(20),
)
INSERT INTO METADATA VALUES ('Employees','dbo')
INSERT INTO METADATA VALUES ('Departments','dbo')
INSERT INTO METADATA VALUES ('Products','dbo')
INSERT INTO METADATA VALUES ('Customers','dbo')
INSERT INTO METADATA VALUES ('Sales','dbo')
SELECT * FROM METADATA

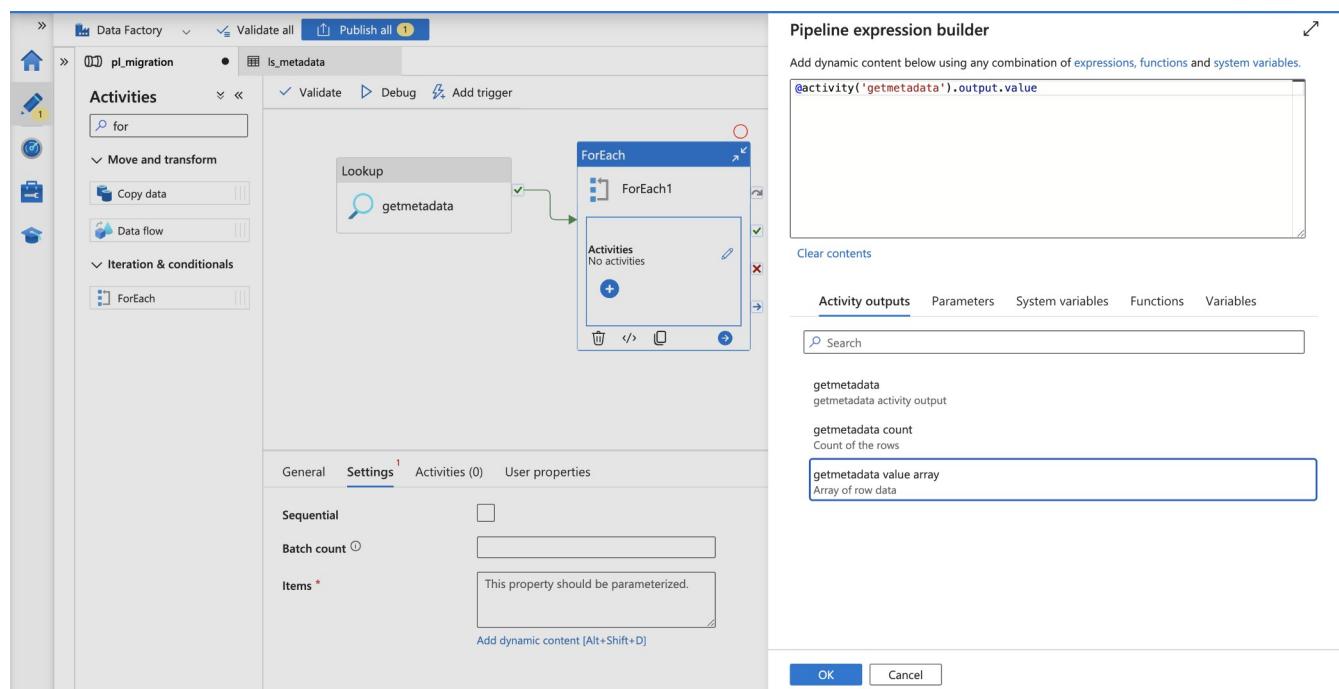
```

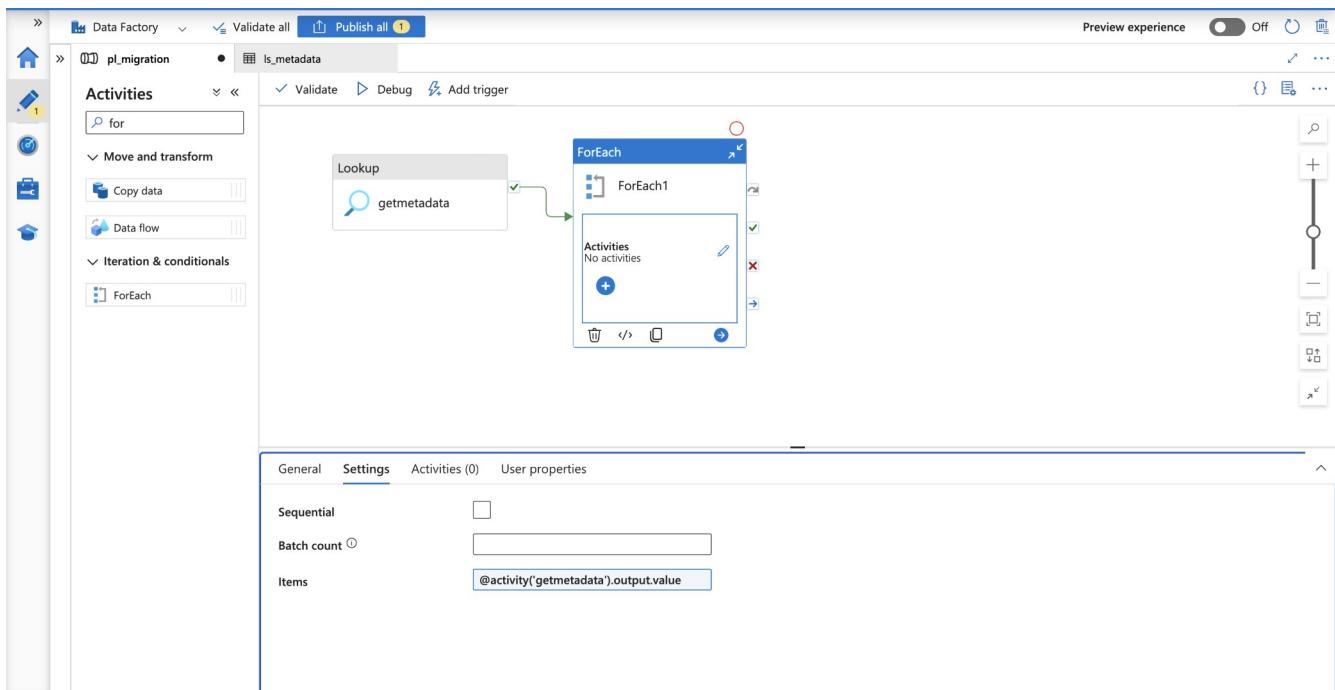
	id	table_name	schema_name
1	1	Employees	dbo
2	2	Departments	dbo
3	3	Products	dbo
4	4	Customers	dbo
5	5	Sales	dbo

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

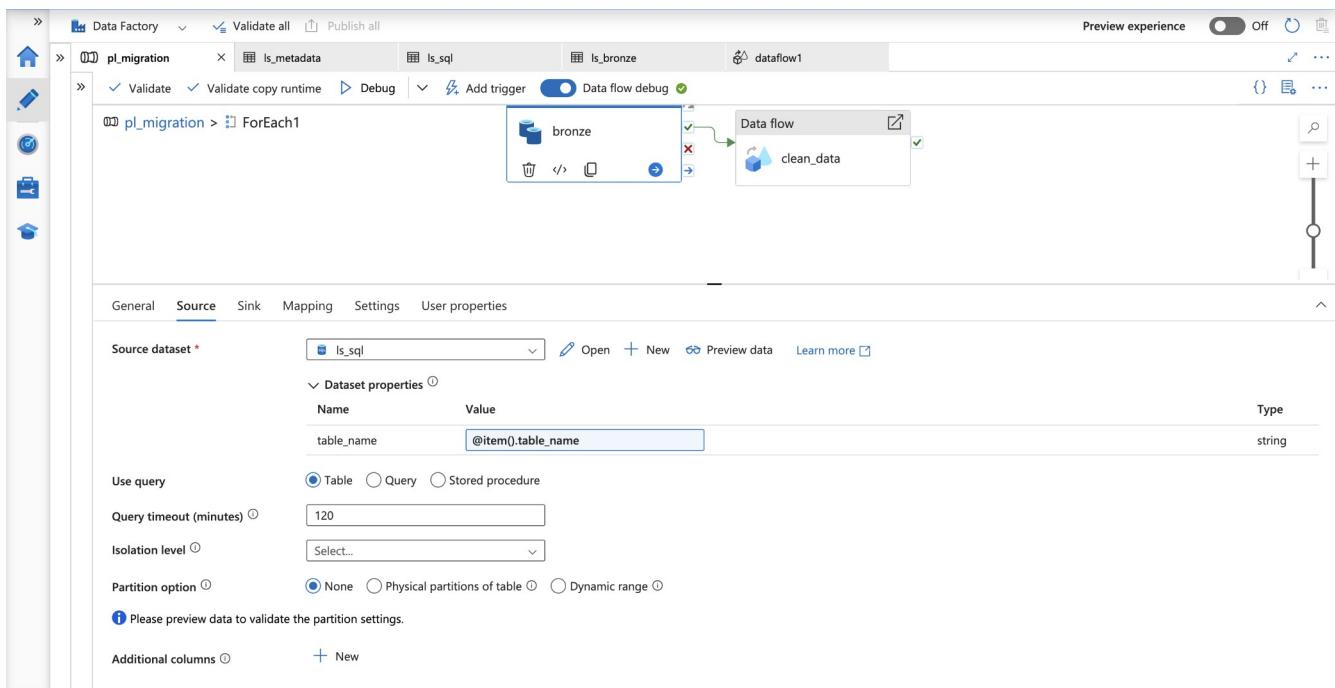


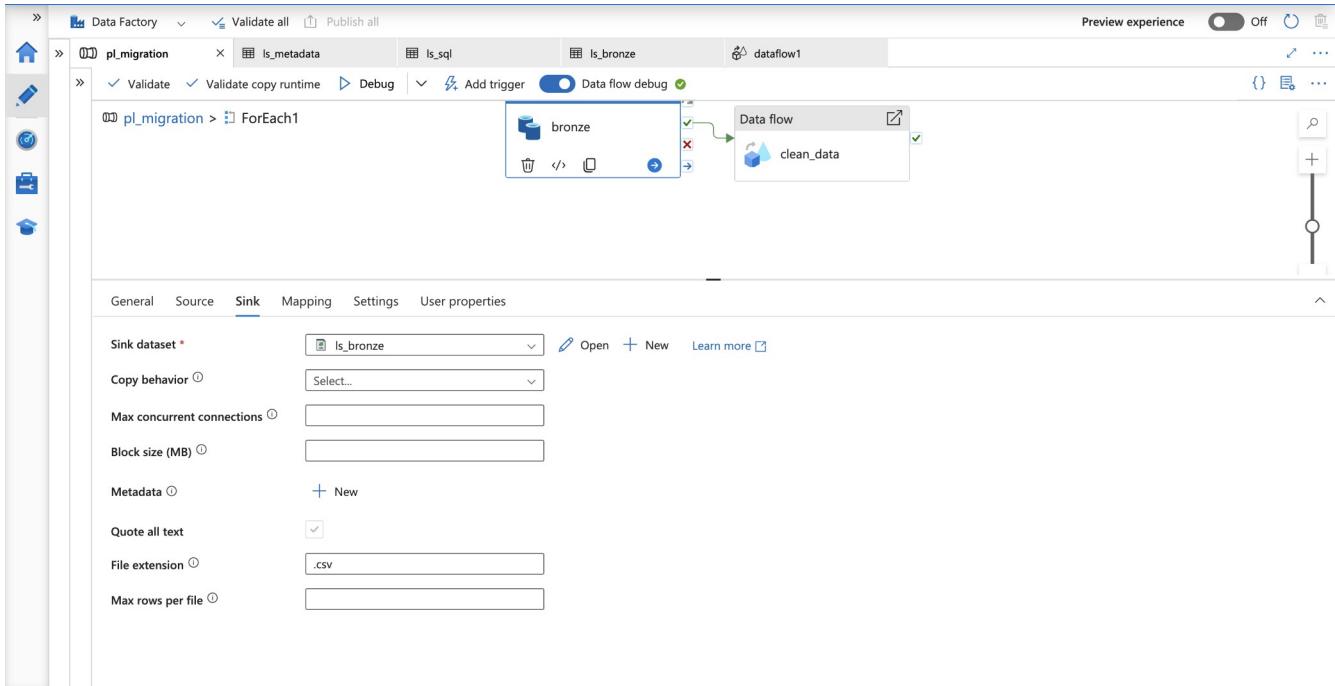
Next I added a for each to iterate through the tables and inside that I added a copy data activity to copy the tables and load them as csv files in the bronze container.



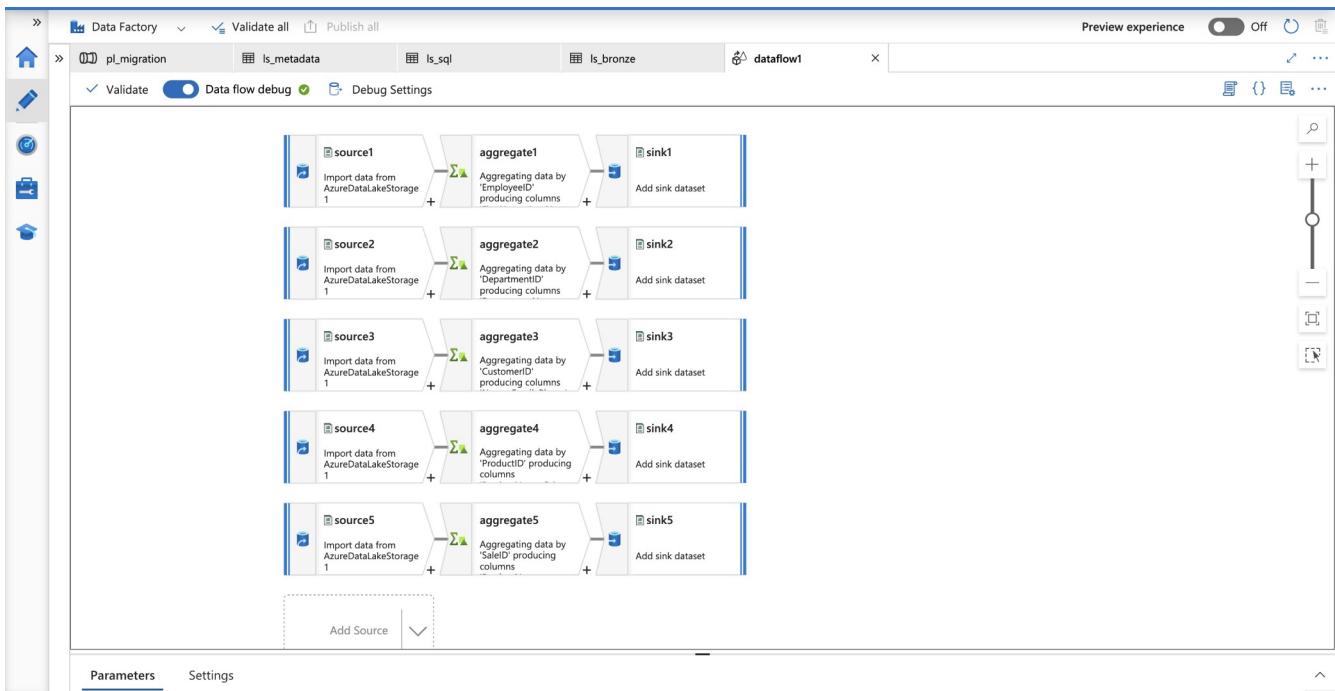


Now inside the for each I added a copy data activity to copy the tables into csv files in storage account bronze layer.





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.

The screenshot shows the Azure Data Factory pipeline run history. The pipeline ID is 5e760507-509a-4c20-8ff5-e08235ca2c96. The status is Succeeded. The table below lists 12 activities, each with a green checkmark indicating success. The activities include various data flow and copy operations.

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime	User prop...	Activity run ID
clean_data	Succeeded	Data flow	5/7/2025, 10:44:41 PM	5m 15s	AutoResolveIntegrationRuntime (East US)		4a8abc8d-bbc5-4138-962d-22255623c
clean_data	Succeeded	Data flow	5/7/2025, 10:44:39 PM	6m 8s	AutoResolveIntegrationRuntime (East US)		0c5855a7-eb95-4675-9d76-c60b02ba2
clean_data	Succeeded	Data flow	5/7/2025, 10:44:38 PM	3m 55s	AutoResolveIntegrationRuntime (East US)		d9963fc6-4c61-4e4f-ae2f-a0a209ac3c5
clean_data	Succeeded	Data flow	5/7/2025, 10:44:38 PM	6m 44s	AutoResolveIntegrationRuntime (East US)		805bc4f5-7199-492a-b434-4fd59ec26
clean_data	Succeeded	Data flow	5/7/2025, 10:44:36 PM	4m 45s	AutoResolveIntegrationRuntime (East US)		afcfe9d6-078e-42fd-9b86-9ff9ae9ed0
bronze	Succeeded	Copy data	5/7/2025, 10:44:18 PM	22s	AutoResolveIntegrationRuntime (East US)		e87a4022-3f8f-4a0d-8ffc-ab806ef0f0f8e
bronze	Succeeded	Copy data	5/7/2025, 10:44:18 PM	19s	AutoResolveIntegrationRuntime (East US)		9c49ad28-5075-4992-a761-a3c6b01e7
bronze	Succeeded	Copy data	5/7/2025, 10:44:18 PM	20s	AutoResolveIntegrationRuntime (East US)		e28d4d16-ab94-4aa4-9ca7-23e607449
bronze	Succeeded	Copy data	5/7/2025, 10:44:18 PM	17s	AutoResolveIntegrationRuntime (East US)		e0b443de-0078-4834-8c55-5c5d1dbbc
bronze	Succeeded	Copy data	5/7/2025, 10:44:18 PM	20s	AutoResolveIntegrationRuntime (East US)		a753b8da-30d9-4219-bc8f-42ee0ea70
ForEach1	Succeeded	ForEach	5/7/2025, 10:44:17 PM	7m 7s			dd9e2023-1af1-4704-89fa-1eb23294
getmetadata	Succeeded	Lookup	5/7/2025, 10:44:07 PM	9s	AutoResolveIntegrationRuntime (Australia East)		51c96b1f-4c84-4e3b-8392-4733923b4

#### 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:

The screenshot shows the Azure Data Factory pipeline editor. A 'ForEach' loop is selected, which contains a 'Copy data' activity and a 'Data flow' activity named 'clean\_data'. The 'clean\_data' data flow has a green checkmark indicating it has been successfully migrated.

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

Pipeline name: pl\_migration

```

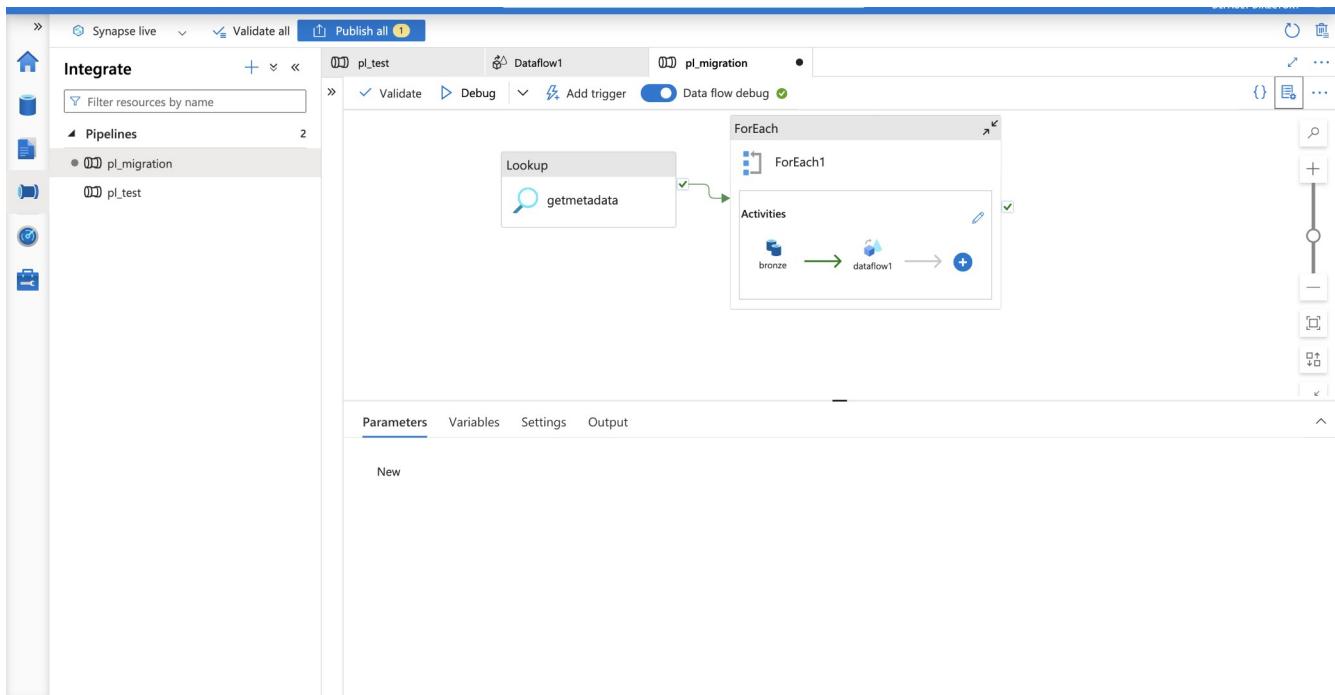
103     },
104     ],
105   },
106   {
107     "name": "clean_data",
108     "type": "ExecuteDataFlow",
109     "dependsOn": [
110       {
111         "activity": "bronze",
112         "dependencyConditions": [
113           "Succeeded"
114         ]
115       }
116     ],
117     "policy": {
118       "timeout": "0.12:00:00",
119       "retry": 0,
120       "retryIntervalInSeconds": 30,
121       "secureOutput": false,
122       "secureInput": false
123     },
124     "userProperties": [],
125     "typeProperties": {
126       "dataflow": {
127         "referenceName": "dataflow1",
128         "type": "DataFlowReference"
129       },
130       "compute": {
131         "coreCount": 8,
132         "computeType": "General"
133       }
134     }
135   }
136 }
```

Paste the code in the new pipeline in synapse.

Pipeline name: pl\_migration

```

1  {
2    "name": "pl_migration",
3    "properties": {
4      "activities": [
5        {
6          "name": "getmetadata",
7          "type": "Lookup",
8          "dependsOn": [],
9          "policy": {
10            "timeout": "0.12:00:00",
11            "retry": 0,
12            "retryIntervalInSeconds": 30,
13            "secureOutput": false,
14            "secureInput": false
15          },
16          "userProperties": [],
17          "typeProperties": {
18            "source": {
19              "type": "AzureSqlSource",
20              "queryTimeout": "02:00:00",
21              "partitionOption": "None"
22            },
23            "dataset": {
24              "referenceName": "ds_metadata",
25              "type": "DatasetReference"
26            },
27            "firstRowOnly": false
28          }
29        },
30        {
31          "name": "ForEach1",
32          "...": "... = ..."
33        }
34      ]
35    }
36  }
```



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-88a4a1ebd1
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-a927-1cc6eafbd
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-4fbf-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:

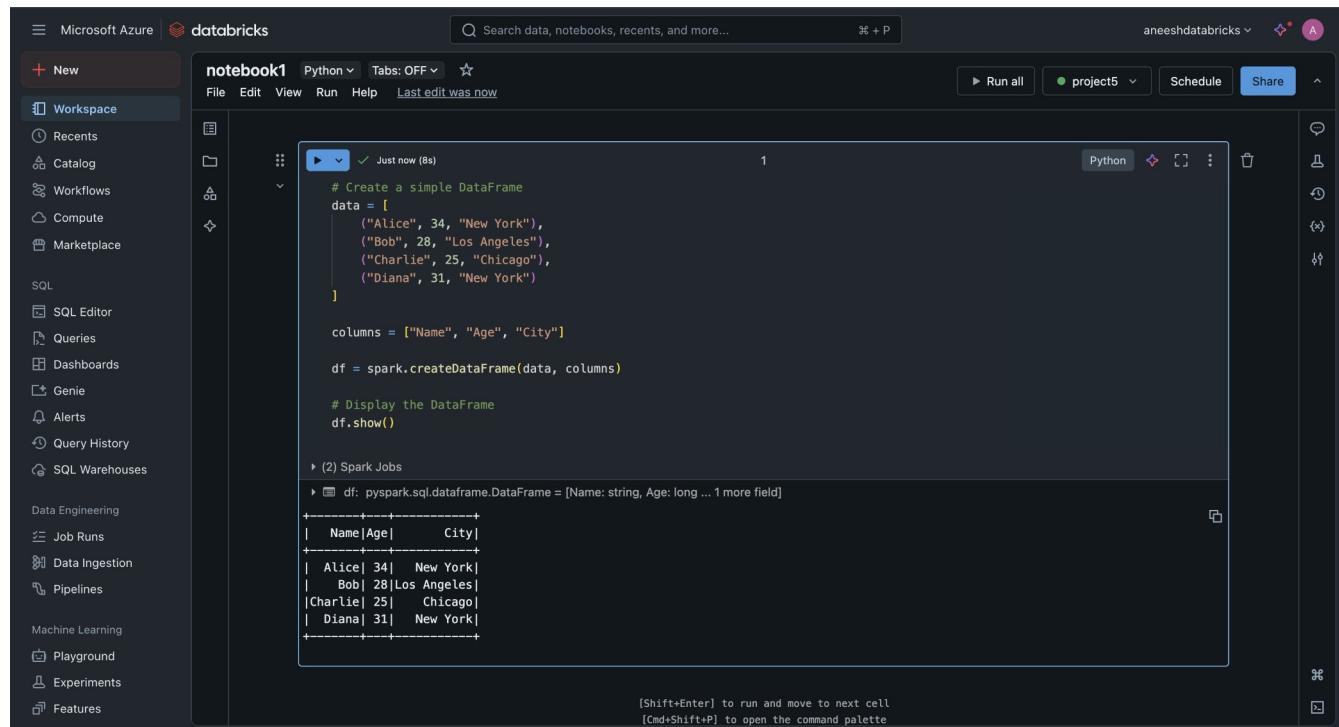
- Fixed broken resource references in the JSON
- Recreated or renamed datasets and the Data Flow

- 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.



The screenshot shows a Databricks notebook interface. On the left is a sidebar with various workspace options like Recents, Catalog, Workflows, Compute, Marketplace, SQL, and Data Engineering. The main area is titled 'notebook1' and shows a single code cell. The code creates a DataFrame from a list of tuples and displays it:

```

# Create a simple DataFrame
data = [
    ("Alice", 34, "New York"),
    ("Bob", 28, "Los Angeles"),
    ("Charlie", 25, "Chicago"),
    ("Diana", 31, "New York")
]

columns = ["Name", "Age", "City"]

df = spark.createDataFrame(data, columns)

# Display the DataFrame
df.show()

```

The output of the code is a table:

Name	Age	City
Alice	34	New York
Bob	28	Los Angeles
Charlie	25	Chicago
Diana	31	New York

At the bottom of the notebook area, there are instructions: '[Shift+Enter] to run and move to next cell [Cmd+Shift+P] to open the command palette'.

Microsoft Azure | databricks

Search data, notebooks, recents, and more... ⌘ + P

aneeshdatabricks A

Notebook2 Python Tabs: OFF ⚡

File Edit View Run Help Last edit was now

Run all project5 Schedule Share

**Code Cell 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()
```

**Output 1**

```
(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 |
+---+
```

Shift+Enter to run and move to next cell

Microsoft Azure | databricks

Search data, notebooks, recents, and more... ⌘ + P

aneeshdatabricks A

Notebook3 Python Tabs: OFF ⚡

File Edit View Run Help Last edit was now

Automatic compute resource connect success  
Automatically connected to your most recent compute resource **project5**.

**Code Cell 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;
```

**Output 1**

name	salary
Bob	5000
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.

```

# Sample data: students and scores
data = [
    ("Alice", "Math", 85),
    ("Alice", "Science", 90),
    ("Bob", "Math", 78),
    ("Bob", "Science", 83),
    ("Clara", "Math", 92),
    ("Clara", "Science", 95)
]

columns = ["Student", "Subject", "Score"]

df = spark.createDataFrame(data, columns)

# Group by student and calculate average, max, and min score
df_grouped = df.groupBy("Student").agg(
    avg("Score").alias("AvgScore"),
    max("Score").alias("MaxScore"),
    min("Score").alias("MinScore"
)

# Sort by highest average score
df_grouped.orderBy("AvgScore", ascending=False).show()

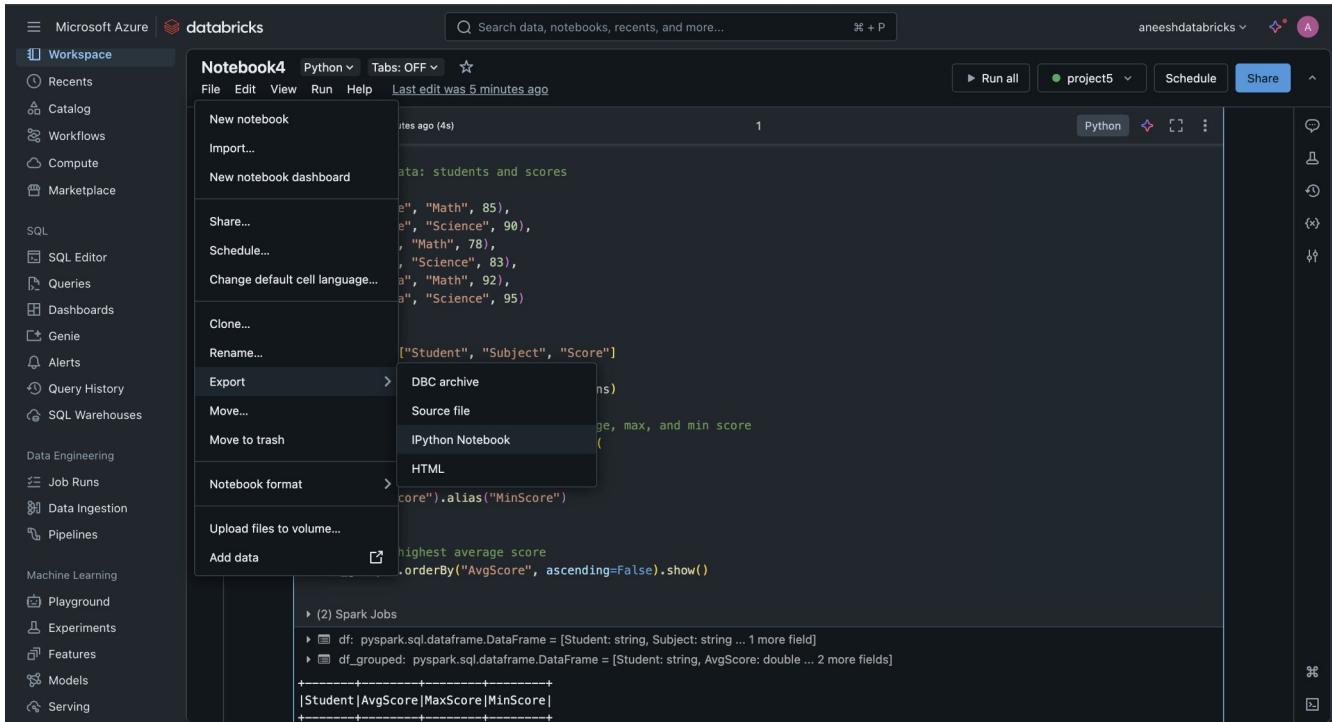
```

(2) Spark Jobs

Student	AvgScore	MaxScore	MinScore
Alice	87.5	90	85
Bob	80.5	83	78
Clara	92	95	92

## 2, Export the notebooks

Goto File → Export → iPython notebook

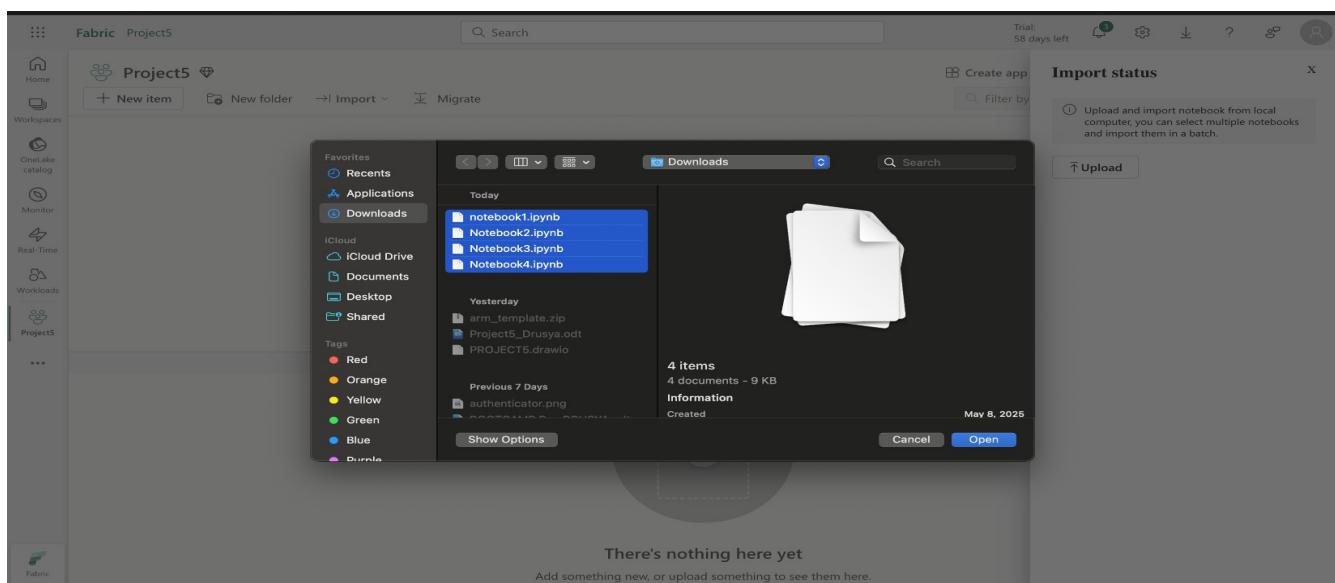
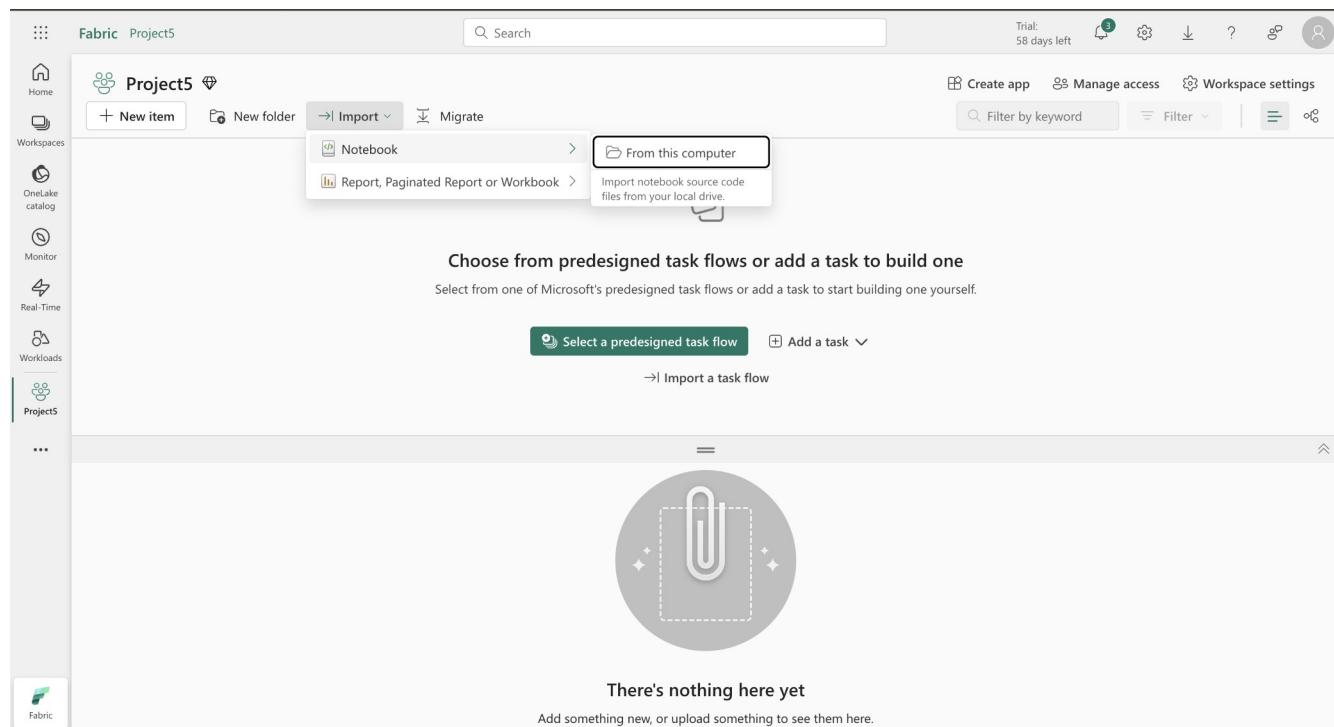


Similarly done for all the notebooks.

### 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.

The screenshot shows the Microsoft Fabric Project5 workspace. The left sidebar includes links for Home, Workspaces, OneLake catalog, Monitor, Real-Time, Workloads, and Project5. The main area displays a list of four notebooks: notebook1, Notebook2, Notebook3, and Notebook4. Each entry includes columns for Name, Type, Task, Owner, Refreshed, Next refresh, Endorsement, Sensitivity, and Included in app. A central message says "Choose from predesigned task flows or add a task to build one". Buttons for "Select a predesigned task flow" and "Add a task" are present, along with an "Import a task flow" link.

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

The screenshot shows a Jupyter Notebook session titled "notebook1". The code cell contains Python code to create a DataFrame and display its contents. The output cell shows the DataFrame with four rows of data: Alice (34, New York), Bob (28, Los Angeles), Charlie (25, Chicago), and Diana (31, New York). The status bar at the bottom indicates "Session ready" and "AutoSave: On".

```
1 # Create a simple DataFrame
2 data = [
3     ("Alice", 34, "New York"),
4     ("Bob", 28, "Los Angeles"),
5     ("Charlie", 25, "Chicago"),
6     ("Diana", 31, "New York")
7 ]
8
9 columns = ["Name", "Age", "City"]
10 df = spark.createDataFrame(data, columns)
11
12 # Display the DataFrame
13 df.show()
14
15
```

[1] ✓ 19 sec - Session ready in 11 sec 36 ms. Command executed in 7 sec 998 ms by drusya2025 on 12:46:35 AM, 5/08/25

> ⚡ Spark jobs (3 of 3 succeeded) [Resources]

Name	Age	City
Alice	34	New York
Bob	28	Los Angeles
Charlie	25	Chicago
Diana	31	New York

Notebook2 | Saved

Home Edit AI tools Run View

Search Trial: 58 days left

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.

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()
```

[1] ✓ 7 sec - Command executed in 3 sec 859 ms by drusya2025 on 12:49:13 AM, 5/08/25

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

Home Edit AI tools Run View

Search Trial: 58 days left

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.

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;
```

[3] ✓ 1 sec - Command executed in 60 ms by drusya2025 on 12:50:32 AM, 5/08/25

Table view

	name	salary
1	Bob	5000
2	Clara	8000

Download Search Inspect Selected Cell 1 of 1 cells

The screenshot shows the Microsoft Fabric Notebook interface. The top navigation bar includes 'Home', 'Edit', 'AI tools', 'Run', 'View', 'Comments', 'History', 'Develop', 'Share', and user profile icons. The left sidebar lists workspaces: OneLake catalog, Monitor, Real-Time, Project5, Notebook4 (selected), Notebook3, Notebook2, and notebook1. The main area is titled 'Explorer' and contains a code cell with PySpark Python code. The code creates a DataFrame from a list of student scores and groups them by student to calculate average, max, and min scores. The output shows two rows: Clara with scores 93.5, 95, 92 and Alice with scores 87.5, 90, 85.

```

5  data = [
6      ("Alice", "Math", 85),
7      ("Alice", "Science", 90),
8      ("Bob", "Math", 78),
9      ("Bob", "Science", 83),
10     ("Clara", "Math", 92),
11     ("Clara", "Science", 95)
12 ]
13
14 columns = ["Student", "Subject", "Score"]
15
16 df = spark.createDataFrame(data, columns)
17
18 # Group by student and calculate average, max, and min score
19 df_grouped = df.groupBy("Student").agg(
20     avg("Score").alias("AvgScore"),
21     max("Score").alias("MaxScore"),
22     min("Score").alias("MinScore")
23 )
24
25 # Sort by highest average score
26 df_grouped.orderBy("AvgScore", ascending=False).show()
27

```

[1] ✓ 11 sec - Command executed in 5 sec 631 ms by drusya2025 on 12:52:12 AM, 5/08/25

> Spark jobs (2 of 2 succeeded) Resources

Student	AvgScore	MaxScore	MinScore
Clara	93.5	95	92
Alice	87.5	90	85

Selected Cell 1 of 1 cells

## CONCLUSION

Thus the project successfully migrated pipeline from data factory to synapse analytics 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.