

# **Bootcamp Project**

## **Project 5 - Migrating pipelines from ADF to Synapse**

**Madhumitha Vijayakumar**

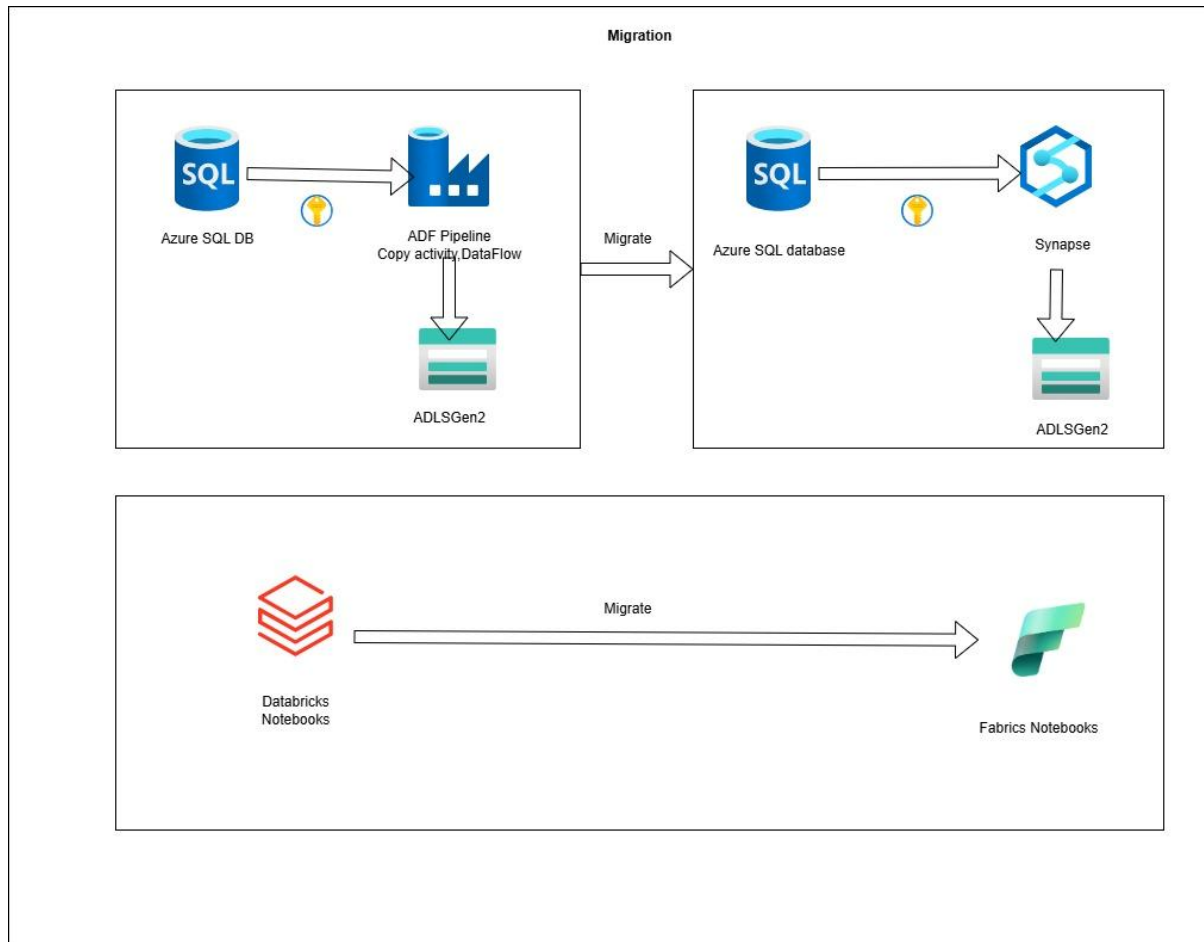
**11-05-2025**

## Project Overview

Develop ADF pipelines (copy activity, foreach loop, look up) and migrate the pipelines, datasets, linked services to Azure Synapse.

**Github link** – [https://github.com/DeepthiChethi/Azure\\_DE/tree/Madhumita/Project5](https://github.com/DeepthiChethi/Azure_DE/tree/Madhumita/Project5).

## Architecture



## Tools and Technologies

- Azure Data Factory
- Azure Synapse
- Azure SQL database
- ADLSGen2
- Azure Key vault
- Databricks
- Fabrics
- Draw.io

## Task -1: Migrating pipelines from ADF to Synapse

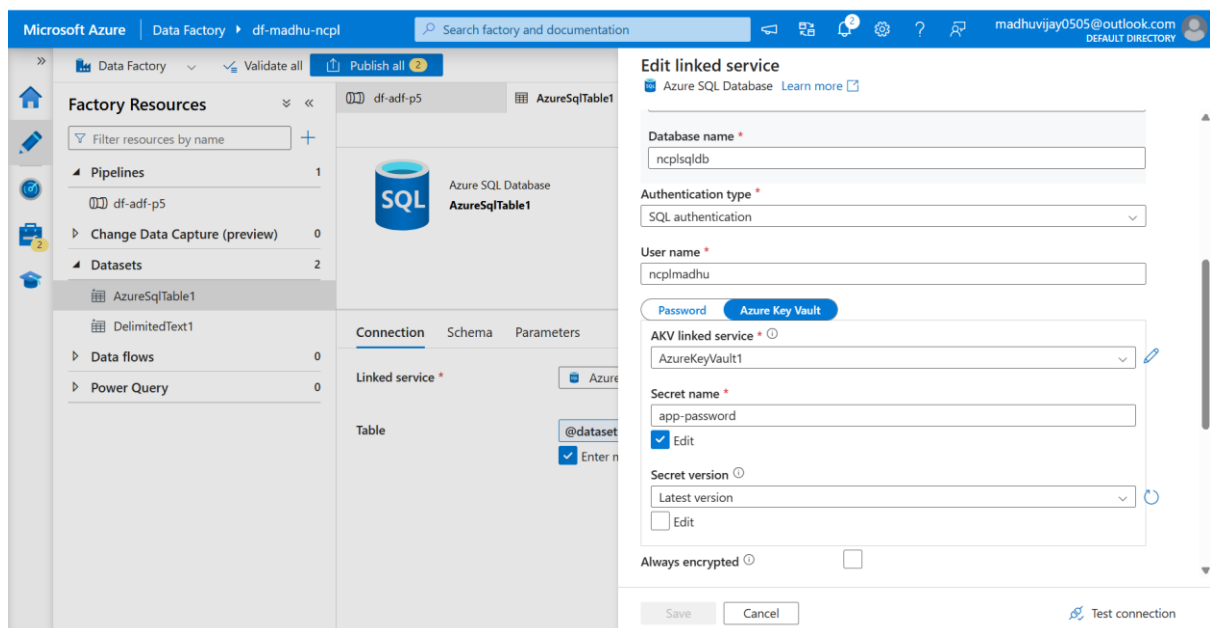
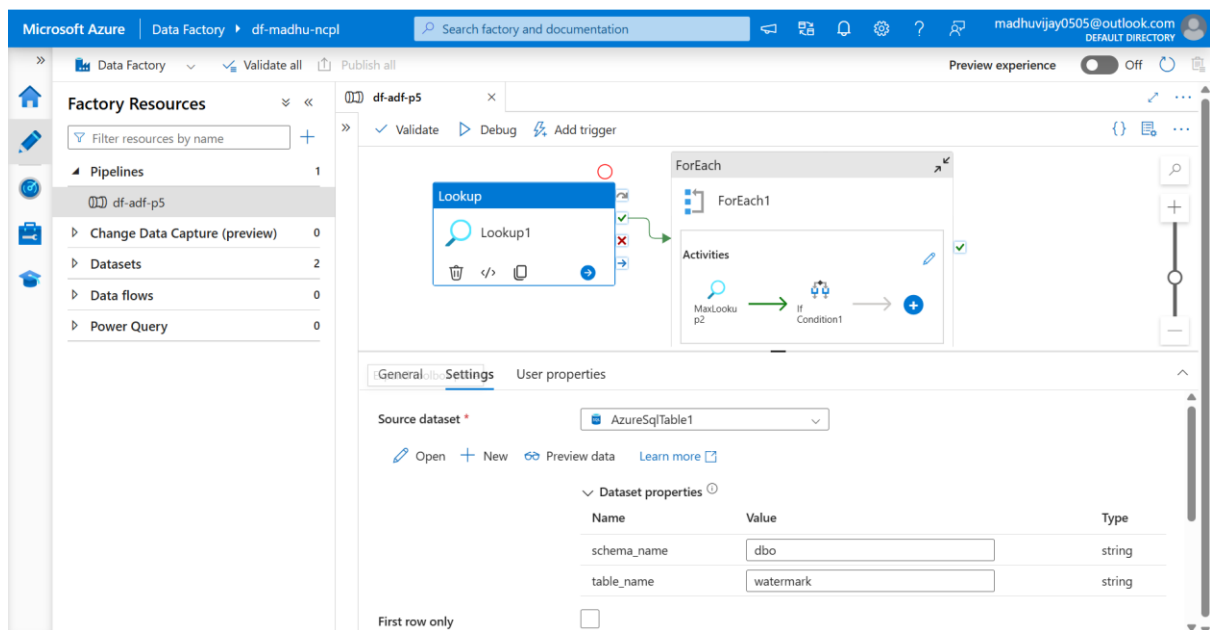
Create a watermark table in SSMS

Insert the values in the watermark table

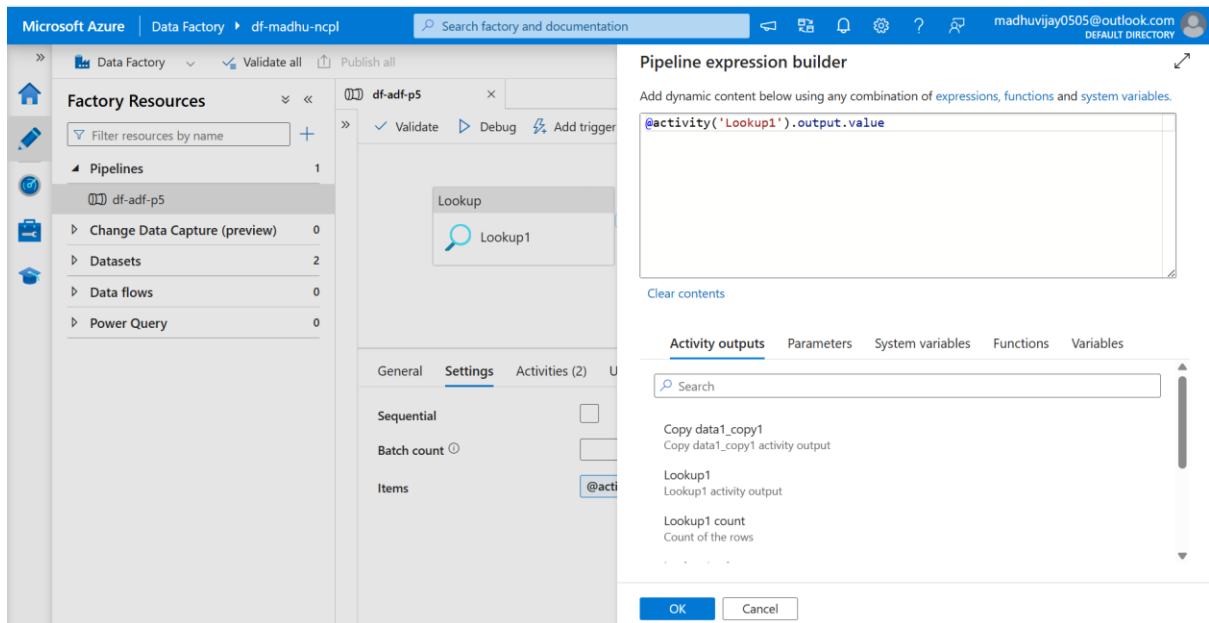
Create stored procedure for updating values in lpv column

Navigate to ADF workspace

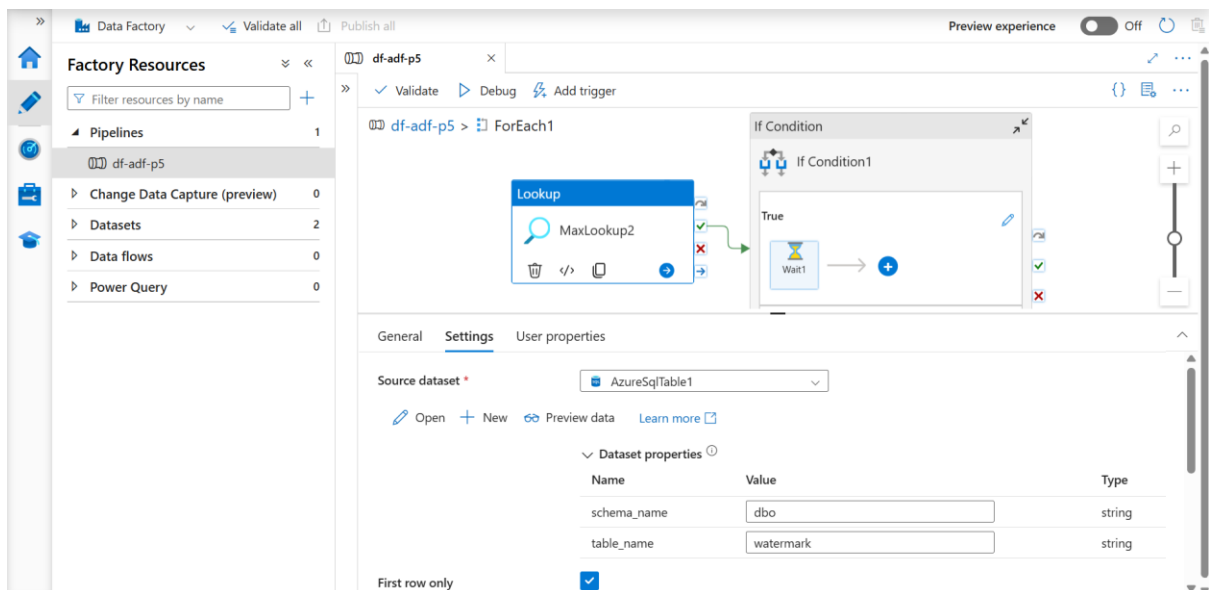
Create linked services for SQL database using key vault and connect lookup activity to foreach loop. Create parameters schema\_name and table\_name in lookup activity.



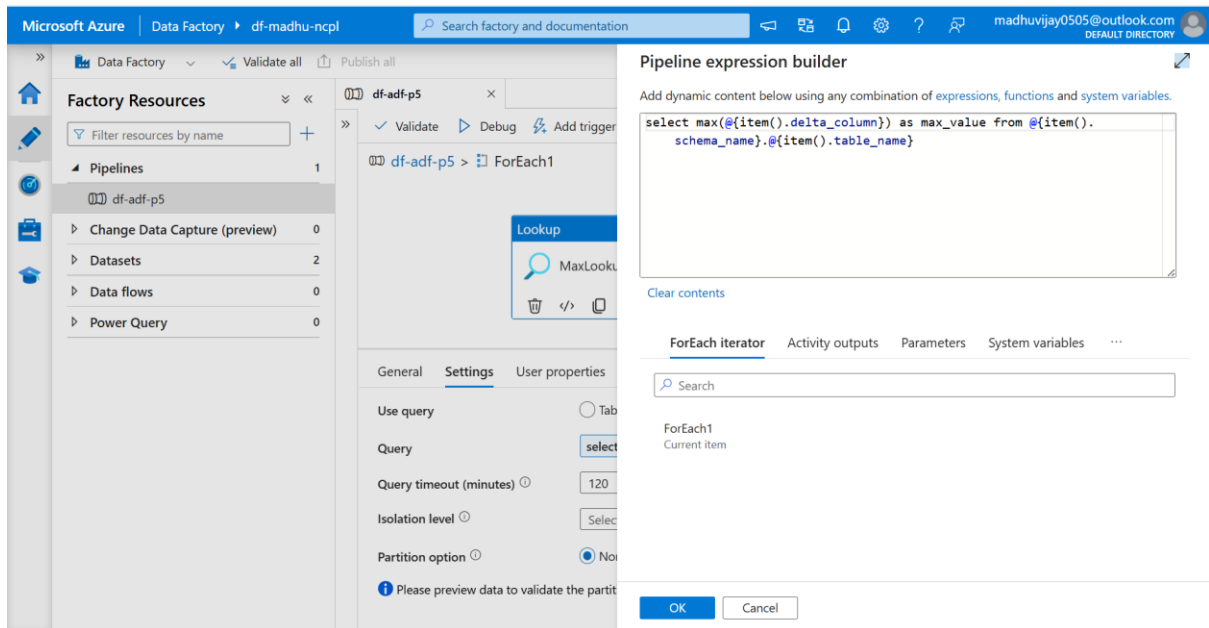
In foreach loop activity, click lookup value array for looping look up activity.



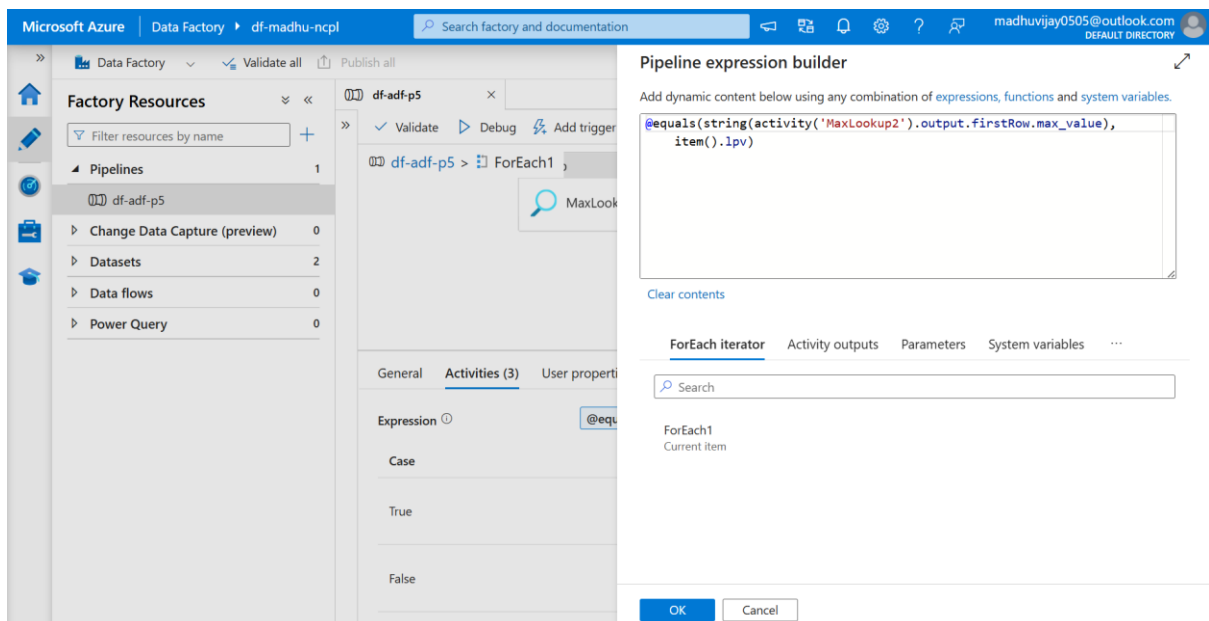
Drag and drop look up activity, enable first row only.



Use the query to select the maximum value of delta column.

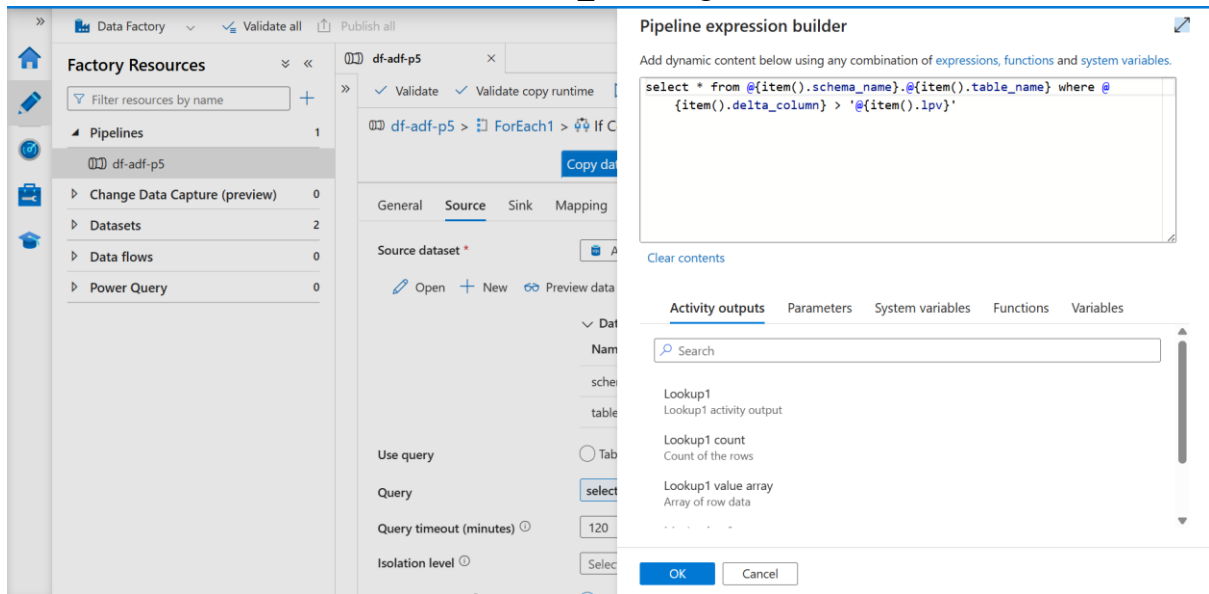


Drag and drop the if condition activity, if condition is true, wait activity will be executed and if condition is false, copy activity and Stored Procedure will be executed.



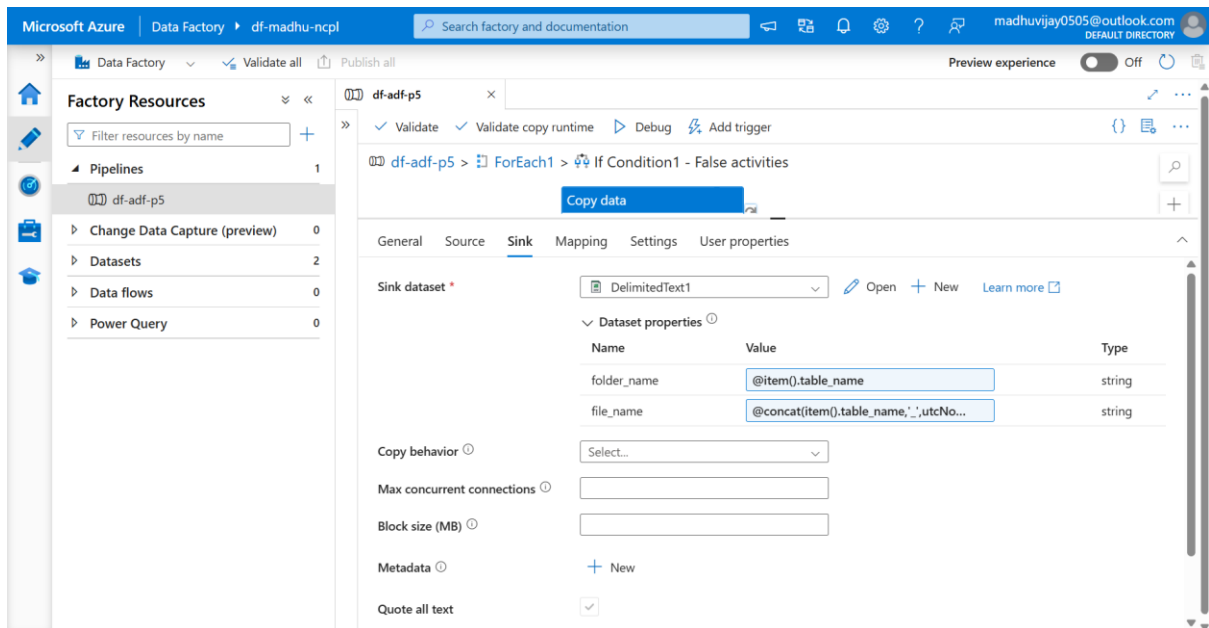
Copy activity-Source

Retrieve the records where the value in delta\_column greater the value in LPV.



Sink

Select delimited text as sink dataset, create folder\_name and file\_name parameters and add @item().table\_name and @concat(item().table\_name,'\_',utcNow(),'.csv') in the following parameters



Drag and drop stored procedure and on success connect copy activity with stored procedure

Microsoft Azure | Data Factory | df-madhu-ncpl | Search factory and documentation | madhuvijay0505@outlook.com | DEFAULT DIRECTORY

Factory Resources

- Pipelines: 1
  - df-adf-p5
- Change Data Capture (preview): 0
- Datasets: 2
- Data flows: 0
- Power Query: 0

df-adf-p5

Copy data

Stored procedure

General Settings User properties

Linked service \*  Test connection Edit + New

Stored procedure name \*  Enter manually

Stored procedure parameters

Import + New Delete

Name	Type *	Value
lpv	String	@activity('MaxLookup2').output.first...
tablename	String	@item().table_name

Microsoft Azure | Data Factory | df-madhu-ncpl | Search factory and documentation | madhuvijay0505@outlook.com | DEFAULT DIRECTORY

Factory Resources

- Pipelines: 1
  - df-adf-p5
- Change Data Capture (preview): 0
- Datasets: 2
- Data flows: 0
- Power Query: 0

df-adf-p5

Copy data

General Settings User properties

Linked service \*  Test connection Edit + New

Stored procedure name \*  Enter manually

Stored procedure parameters

Import + New Delete

Add dynamic content [Alt+Shift+D]

Pipeline expression builder

Add dynamic content below using any combination of expressions, functions and system variables.

@activity('MaxLookup2').output.firstRow.max\_value

Clear contents

Activity outputs Parameters System variables Functions Variables

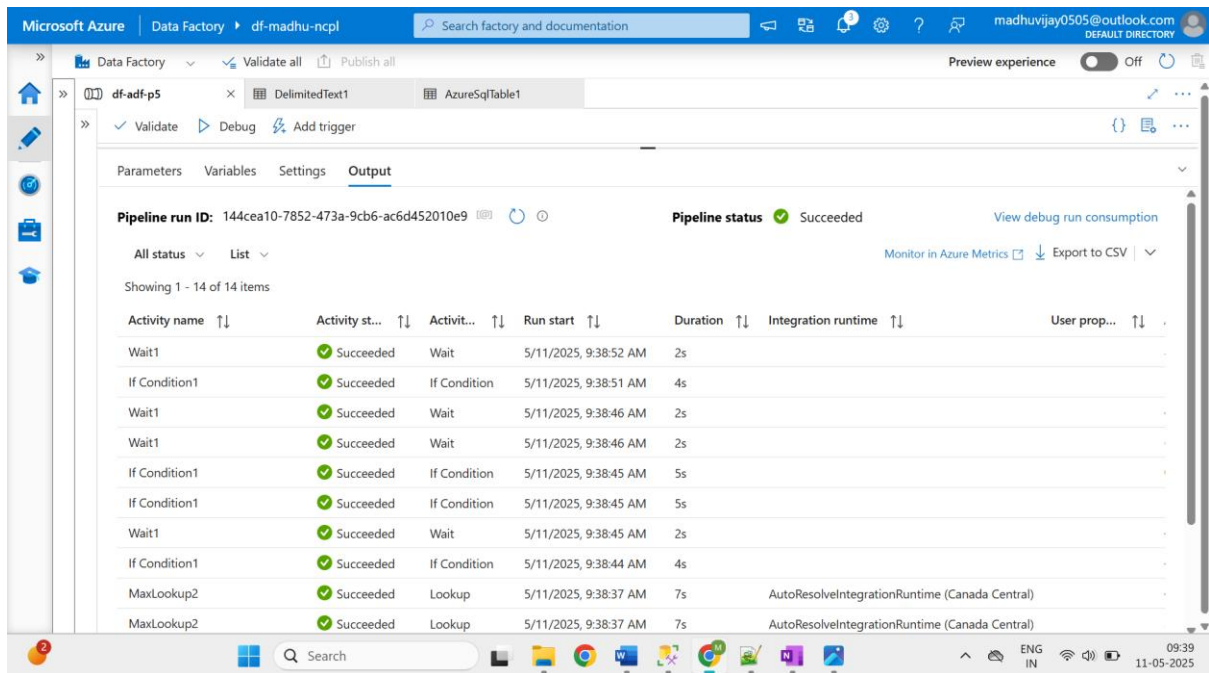
Search

Copy data1\_copy1  
Copy data1\_copy1 activity output

Lookup1  
Lookup1 activity output

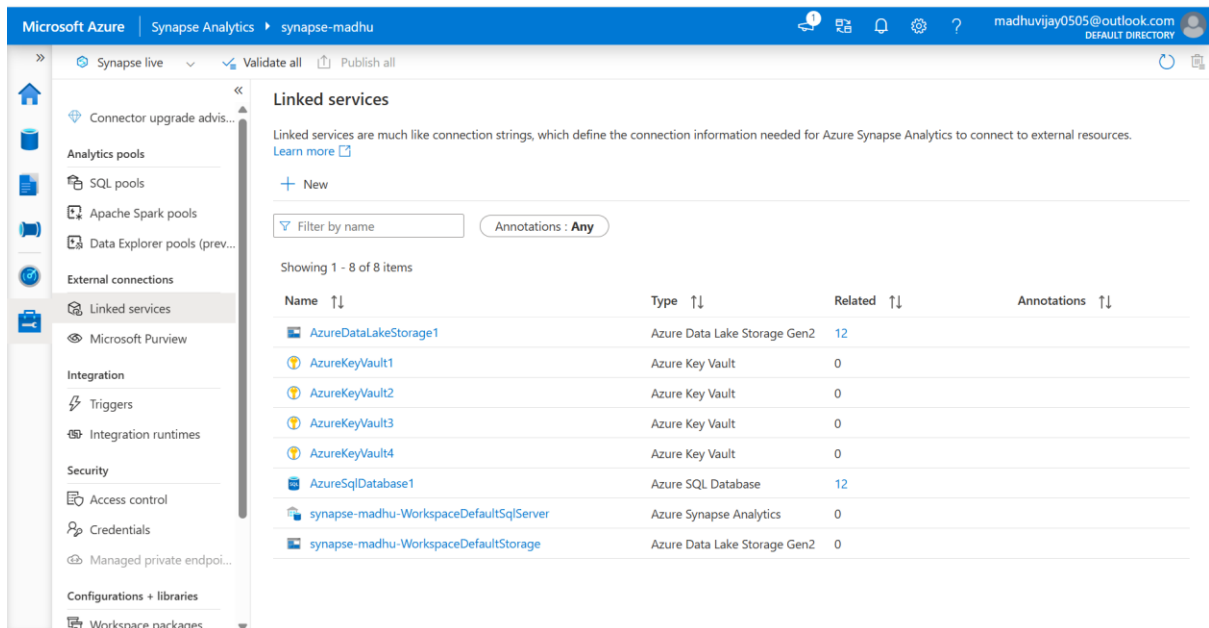
Lookup1 count  
Count of the rows

OK Cancel



## Synapse pipeline

At first, create linked service and datasets in synapse workspace.



Create a pipeline in synapse workspace with same as ADF pipeline

Copy JSON code from ADF to synapse



Microsoft Azure | Data Factory | df-madhu-ncpl | Search factory and documentation | madhuvijay0505@outlook.com | DEFAULT DIRECTORY

Pipeline name: df-adf-p5

Copy to clipboard

```
1 {
2   "name": "df-adf-p5",
3   "properties": {
4     "activities": [
5       {
6         "name": "Lookup1",
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": {
```

OK Cancel

Paste it in synapse workspace

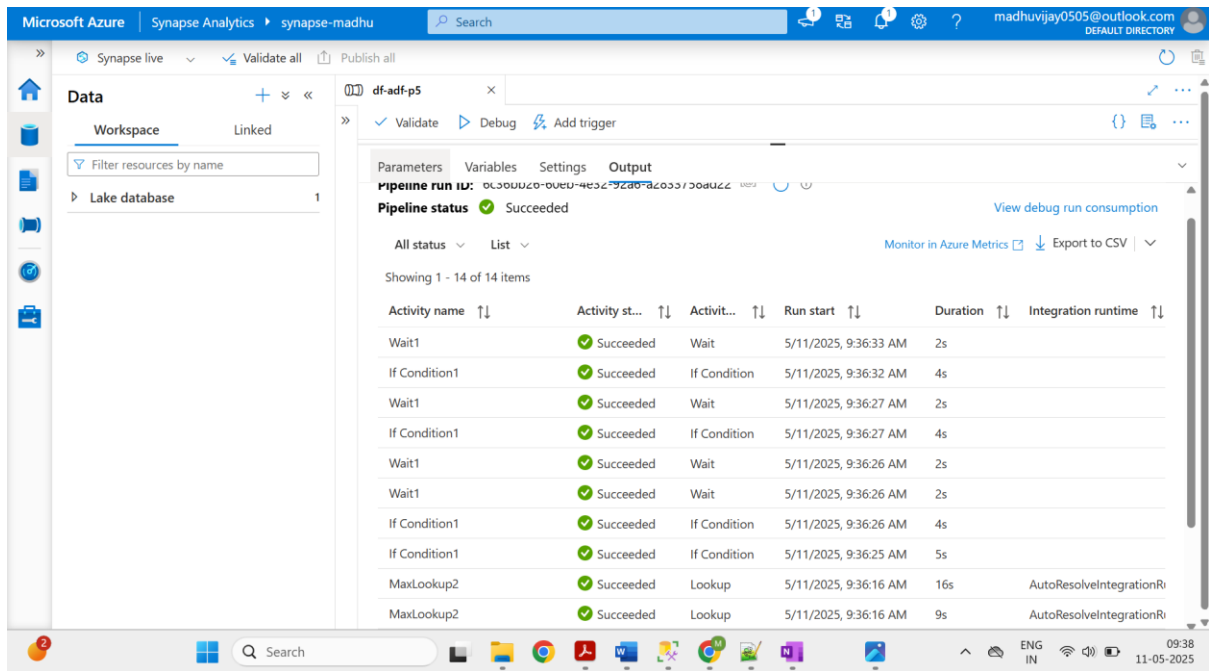
Microsoft Azure | Synapse Analytics | synapse-madhu | Search | madhuvijay0505@outlook.com | DEFAULT DIRECTORY

Pipeline name: df-adf-p5

Copy to clipboard

```
1 {
2   "name": "df-adf-p5",
3   "properties": {
4     "activities": [
5       {
6         "name": "Lookup1",
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": {
```

OK Cancel

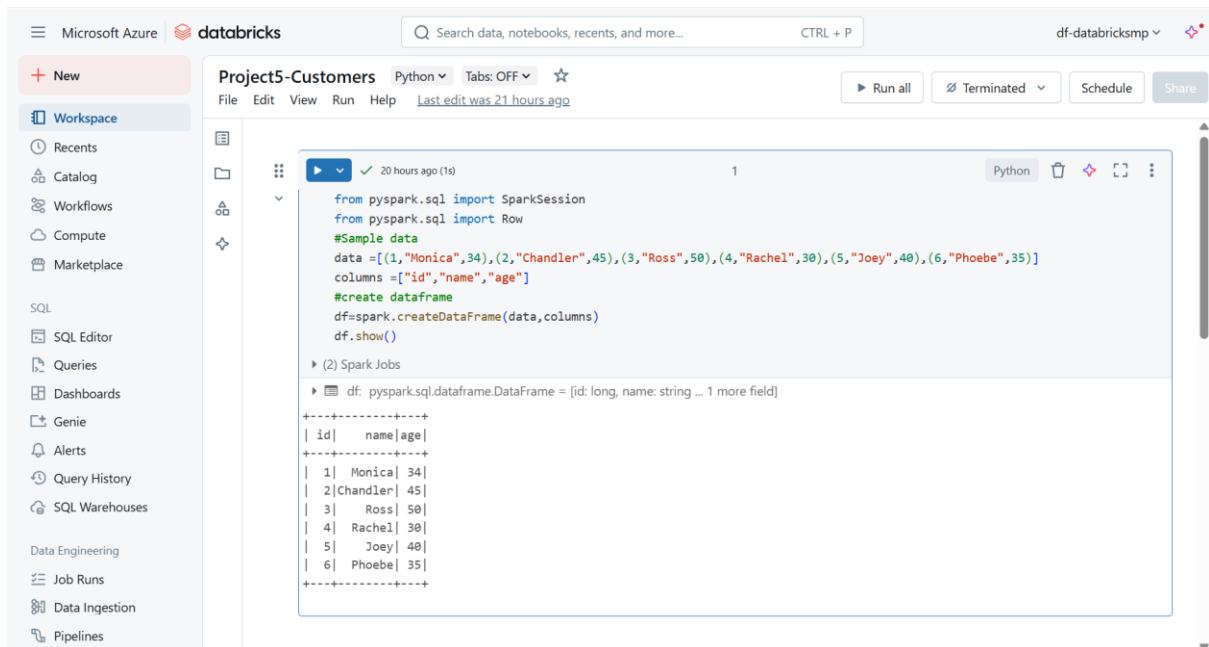


Run the pipeline and it successfully executed.

## Task 2

Migrate databricks notebook to fabric notebook.

Notebook 1 -Customers



Notebook 2- Joins

Microsoft Azure databricks Search data, notebooks, recent, and more... CTRL + P df-databricksmp

**Project5-Joins** Python Tabs: OFF ☆

File Edit View Run Help Last edit was 2 days ago

Run all Terminated Schedule Share

2 days ago (7s) 1 Python

```
#creating sample dataframe
from pyspark.sql import SparkSession
from pyspark.sql.functions import col
#sample data from Employees
emp_data=[(1,"Harry",1000),(2,"John",2000),(3,"Ravi",3000),(4,"Ravi",4000)]
emp_columns=["emp_id","emp_name","emp_salary"]
df_emp=spark.createDataFrame(emp_data,emp_column)

#sample data for dept
dept_data = [(1, "HR"), (2, "Finance"), (4, "IT")]
dept_columns = ["emp_id", "department"]
df_dept = spark.createDataFrame(dept_data, dept_columns)

df_emp.show()
df_dept.show()
```

4 Spark Jobs

df\_dept: pyspark.sql.dataframe.DataFrame = [emp\_id: long, department: string]

df\_emp: pyspark.sql.dataframe.DataFrame = [emp\_id: long, emp\_name: string ... 1 more field]

emp_id	emp_name	emp_salary
1	Harry	1000
2	John	2000
3	Ravi	3000
4	Ravi	4000

Microsoft Azure databricks Search data, notebooks, recent, and more... CTRL + P df-databricksmp

**Project5-Joins** Python Tabs: OFF ☆

File Edit View Run Help Last edit was 2 days ago

Run all Terminated Schedule Share

2 days ago (3s) 2

```
#inner Join
df_inner=df_emp.join(df_dept,on='emp_id',how="inner")
df_inner.show()
```

3 Spark Jobs

df\_inner: pyspark.sql.dataframe.DataFrame = [emp\_id: long, emp\_name: string ... 2 more fields]

emp_id	emp_name	emp_salary	department
1	Harry	1000	HR
2	John	2000	Finance
4	Ravi	4000	IT

2 days ago (1s) 3

```
#left Join
df_left=df_emp.join(df_dept,on='emp_id',how="left")
df_left.show()
```

4 Spark Jobs

df\_left: pyspark.sql.dataframe.DataFrame = [emp\_id: long, emp\_name: string ... 2 more fields]

Microsoft Azure databricks Search data, notebooks, recents, and more... CTRL + P df-databricksmp

**Project5-Joins** Python Tabs: OFF ☆

File Edit View Run Help Last edit was 2 days ago

Run all Terminated Schedule Share

```

emp_id|emp_name|emp_salary|department|
-----+-----+-----+-----+
| 1| Harry| 1000| HR|
| 2| John| 2000| Finance|
| 3| Ravi| 3000| NULL|
| 4| Ravi| 4000| IT|
-----+-----+-----+

```

2 days ago (1s) 4

```

#right Join
df_right=df_emp.join(df_dept,on='emp_id',how="right")
df_right.show()

```

(5) Spark Jobs

df\_right: pyspark.sql.dataframe.DataFrame = [emp\_id: long, emp\_name: string ... 2 more fields]

```

emp_id|emp_name|emp_salary|department|
-----+-----+-----+-----+
| 1| Harry| 1000| HR|
| 2| John| 2000| Finance|
| 4| Ravi| 4000| IT|
-----+-----+-----+

```

Microsoft Azure databricks Search data, notebooks, recents, and more... CTRL + P df-databricksmp

**Project5-Joins** Python Tabs: OFF ☆

File Edit View Run Help Last edit was 2 days ago

Run all Terminated Schedule Share

```

#full outer Join
df_outer=df_emp.join(df_dept,on='emp_id',how="outer")
df_outer.show()

```

(3) Spark Jobs

df\_outer: pyspark.sql.dataframe.DataFrame = [emp\_id: long, emp\_name: string ... 2 more fields]

```

emp_id|emp_name|emp_salary|department|
-----+-----+-----+-----+
| 1| Harry| 1000| HR|
| 2| John| 2000| Finance|
| 3| Ravi| 3000| NULL|
| 4| Ravi| 4000| IT|
-----+-----+-----+

```

[Shift+Enter] to run and move to next cell  
[Ctrl+Shift+P] to open the command palette  
[Esc H] to see all keyboard shortcuts

## Notebook 3-Sorted

Microsoft Azure databricks Search data, notebooks, recents, and more... CTRL + P df-databricksmp

**Project5-Sorted** Python Tabs: OFF

File Edit View Run Help Last edit was 2 days ago

Run all Terminated 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

```

from pyspark.sql import SparkSession
spark = SparkSession.builderappName("Sorting").getOrCreate()
#sample data
data = [(1, "Prakash", 32),
        (2, "Madhu", 26),
        (3, "Raghul", 25),
        (4, "Thilak", 35),
        (5, "Ranjana", 34),
        (6, "Sathvik", 5)]

columns = ["id", "name", "age"]
df = spark.createDataFrame(data, columns)
df_sorted_asc=df.orderBy("age")
df_sorted_desc = df.orderBy(df["age"].desc())

df_sorted_asc.show()
df_sorted_desc.show()

```

(2) Spark Jobs

- df: pyspark.sql.dataframe.DataFrame = [id: long, name: string ... 1 more field]
- df\_sorted\_asc: pyspark.sql.dataframe.DataFrame = [id: long, name: string ... 1 more field]
- df\_sorted\_desc: pyspark.sql.dataframe.DataFrame = [id: long, name: string ... 1 more field]

id	name	age
1	Prakash	32
2	Madhu	26
3	Raghul	25
4	Thilak	35
5	Ranjana	34
6	Sathvik	5

Microsoft Azure databricks Search data, notebooks, recents, and more... CTRL + P df-databricksmp

**Project5-Sorted** Python Tabs: OFF

File Edit View Run Help Last edit was 2 days ago

Run all Terminated 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

(2) Spark Jobs

- df: pyspark.sql.dataframe.DataFrame = [id: long, name: string ... 1 more field]
- df\_sorted\_asc: pyspark.sql.dataframe.DataFrame = [id: long, name: string ... 1 more field]
- df\_sorted\_desc: pyspark.sql.dataframe.DataFrame = [id: long, name: string ... 1 more field]

id	name	age
6	Sathvik	5
3	Raghul	25
2	Madhu	26
1	Prakash	32
5	Ranjana	34
4	Thilak	35

id	name	age
4	Thilak	35
5	Ranjana	34
1	Prakash	32
2	Madhu	26
3	Raghul	25
6	Sathvik	5

[Shift+Enter] to run and move to next cell

## Notebook 4-Students

The screenshot shows a Databricks workspace for 'Project5-Students'. The code defines sample data with 4 rows and creates a DataFrame. The table view below the code shows the following data:

id	name	age	lastname
1	Madhu	27	MadhuGeller
2	Teddy	26	TeddyGeller
3	Maithili	28	MaithiliGeller
4	Manishaa	27	ManishaaGeller

The screenshot shows the same Databricks workspace, but the code now includes a 5th row in the sample data. The table view shows the following data:

id	name	age	lastname
1	Madhu	27	MadhuGeller
2	Teddy	26	TeddyGeller
3	Maithili	28	MaithiliGeller
4	Manishaa	27	ManishaaGeller
5	Ponmala	27	PonmalaGeller

5 rows | 5.82s runtime

Export all the 4 notebooks from databricks in ipython notebook

Import the all the ipython notebooks in fabrics. Go to workspace->import-> import notebook.

Notebook 1- Customers

Project5-Customers(1) | Saved

Home Edit AI tools Run View

Run all Connect PySpark (Python) Environment Workspace default Data Wrangler

Explorer

```
1 from pyspark.sql import SparkSession
2 from pyspark.sql import Row
3 #Sample data
4 data = [(1,"Monica",34),(2,"Chandler",45),(3,"Ross",50),(4,"Rachel",30),(5,"Joey",40),(6,"Phoebe",35)]
5 columns = ["id","name","age"]
6 #create dataframe
7 df=spark.createDataFrame(data,columns)
8 df.show()
```

[1] ✓ - Session ready in 11 sec 702 ms. Command executed in 4 sec 252 ms by mfabricis on 12:28:37 AM, 5/11/25

id	name	age
1	Monica	34
2	Chandler	45
3	Ross	50
4	Rachel	30
5	Joey	40
6	Phoebe	35

Not connected AutoSave: On Selected Cell 1 of 2 cells

Project5-Customers(1) | Saved

Home Edit AI tools Run View

Run all Connect PySpark (Python) Environment Workspace default Data Wrangler

Explorer

```
1 df=df.withColumnRenamed("id","userid")\
2 .withColumnRenamed("name","username")\
3 .withColumnRenamed("age","userage")
4 df.show()
```

[2] ✓ - Command executed in 926 ms by mfabricis on 12:28:51 AM, 5/11/25

userid	username	userage
1	Monica	34
2	Chandler	45
3	Ross	50
4	Rachel	30
5	Joey	40
6	Phoebe	35

Not connected AutoSave: On Selected Cell 1 of 2 cells

Notebook 2- joins

Project5-Joins(1) | Saved

Home Edit AI tools Run View

Run All Connect PySpark (Python) Environment Workspace default Data Wrangler

Explorer

```

1 #creating sample dataframe
2 from pyspark.sql import SparkSession
3 from pyspark.sql.functions import col
4 #sample data from Employees
5 emp_data=[(1,"Harry",1000),(2,"John",2000),(3,"Ravi",3000),(4,"Ravi",4000)]
6 emp_columns=["emp_id","emp_name","emp_salary"]
7 df_emp=spark.createDataFrame(emp_data,emp_column)
8
9 #sample data for dept
10 dept_data=[(1,"HR"),(2,"Finance"),(4,"IT")]
11 dept_columns=["emp_id","department"]
12 df_dept=spark.createDataFrame(dept_data,dept_columns)
13
14 df_emp.show()
15 df_dept.show()

```

[1] ✓ - Session ready in 14 sec 258 ms. Command executed in 4 sec 948 ms by mfabric on 8:04:28 AM, 5/11/25

emp_id	emp_name	emp_salary
1	Harry	1000
2	John	2000
3	Ravi	3000
4	Ravi	4000

Not connected AutoSave: On Selected Cell 1 of 5 cells

Project5-Joins(1) | Saved

Home Edit AI tools Run View

Run All Connect PySpark (Python) Environment Workspace default Data Wrangler

Explorer

emp_id	department
1	HR
2	Finance
4	IT

```

1 #inner Join
2 df_inner=df_emp.join(df_dept,on='emp_id',how="inner")
3 df_inner.show()

```

[2] ✓ - Command executed in 2 sec 500 ms by mfabric on 8:04:34 AM, 5/11/25

emp_id	emp_name	emp_salary	department
1	Harry	1000	HR
2	John	2000	Finance
4	Ravi	4000	IT

Not connected AutoSave: On Selected Cell 1 of 5 cells

Notebook3- Sorted



Project5-Sorted(1) Search

Home Edit AI tools Run View Comments History Develop Share

Run all Connect PySpark (Python) Environment Workspace default Data Wrangler

Explorer

```

1 from pyspark.sql import SparkSession
2 spark = SparkSession.builder.appName("Sorting").getOrCreate()
3 #sample data
4 data = [(1, "Prakash", 32),
5         (2, "Madhu", 26),
6         (3, "Raghul", 25),
7         (4, "Thilak", 35),
8         (5, "Ranjana", 34),
9         (6, "Sathvik", 5)]
10
11 columns = ["id", "name", "age"]
12 df = spark.createDataFrame(data, columns)
13 df_sorted_asc = df.orderBy("age")
14 df_sorted_desc = df.orderBy(df["age"].desc())
15
16 df_sorted_asc.show()
17 df_sorted_desc.show()
18

```

[1] ✓ 15 sec - Session ready in 11 sec 385 ms. Command executed in 4 sec 102 ms by mfabrics on 8:14:46 AM, 5/11/25 PySpark (Python)

> Spark jobs (2 of 2 succeeded) Resources

```

...
+-----+
| id | name | age |
+-----+
| 6 | Sathvik | 5 |

```

Session timeout. Run the notebook to start a new session. AutoSave: On Selected Cell 1 of 1 cells

Project5-Sorted(1) Search

Home Edit AI tools Run View Comments History Develop Share

Run all Connect PySpark (Python) Environment Workspace default Data Wrangler

Explorer

```

...
+-----+
| id | name | age |
+-----+
| 6 | Sathvik | 5 |
| 3 | Raghul | 25 |
| 2 | Madhu | 26 |
| 1 | Prakash | 32 |
| 5 | Ranjana | 34 |
| 4 | Thilak | 35 |
+-----+

+-----+
| id | name | age |
+-----+
| 4 | Thilak | 35 |
| 5 | Ranjana | 34 |
| 1 | Prakash | 32 |
| 2 | Madhu | 26 |
| 3 | Raghul | 25 |
| 6 | Sathvik | 5 |
+-----+

```

Session timeout. Run the notebook to start a new session. AutoSave: On Selected Cell 1 of 1 cells

## Notebook 4-Students

The screenshot displays a Databricks workspace for 'Project5-Students(1)'. The notebook contains a PySpark script that creates a DataFrame with sample data and appends a 'Geller' column. The script is as follows:

```
1 from pyspark.sql import SparkSession
2 from pyspark.sql import Row
3 from pyspark.sql.functions import col, concat, lit
4
5 #sample data
6 data=[(1,"Madhu",27),(2,"Teddy",26),(3,"Maithili",28),(4,"Manishaa",27),(5,"Ponmala",27)]
7 Columns=["id","name","age"]
8
9 #createDataFrame
10 df=spark.createDataFrame(data,Columns)
11 df_add_one = df.withColumn("lastname",concat(df.name,lit("Geller")))
12 df_add_one.show()
```

The execution output shows the following table:

id	name	age	lastname
1	Madhu	27	MadhuGeller
2	Teddy	26	TeddyGeller
3	Maithili	28	MaithiliGeller
4	Manishaa	27	ManishaaGeller
5	Ponmala	27	PonmalaGeller

All notebooks ran successfully