HOME PRIJECT AZURE DATABRICKS DATA ENGINEERING PROJECT REPORT

OCTOBER 20, 2025

RUSHIRAJ PATHAK

Contents

1. Introduction 2
2. Objectives 2
3. Tools and Technologies 2
4. Implementation Steps 3
   Step 1 – Azure Setup 3
   Step 2 – Storage Account Creation 5
   Step 3 – Databricks Workspace and Cluster 9
   Step 4 – Connecting Databricks to Azure Data Lake 13
   Step 5 – Data Ingestion and Cleaning 16
   Step 6 – Data Transformation 18
   Step 7 – Writing Data to Data Lake 20
   Step 8 – Creating Database and Delta Table 21
5. Results and Observations 24
6. Conclusion 24

1. Introduction

This project demonstrates the creation of an end-to-end cloud-based data pipeline using **Microsoft Azure Databricks** integrated with **Azure Data Lake Storage Gen2**. The goal is to establish a secure data flow from ingestion to transformation and storage, while performing scalable data analytics using **PySpark** and **Delta Lake**.

The project walks through all critical stages of setup and implementation, including Azure resource provisioning, Databricks workspace configuration, data transformation using Spark, and Delta table creation for structured querying.

2. Objectives

- The main objectives of this project are:
- To create and configure a secure Azure environment for data engineering.
- To connect **Azure Databricks** with **Azure Data Lake Storage** using secure access keys.
- To ingest and clean raw data using **PySpark**.
- To transform, aggregate, and store data in **Delta format**.
- To register and query the processed data as a **Hive Metastore table**.

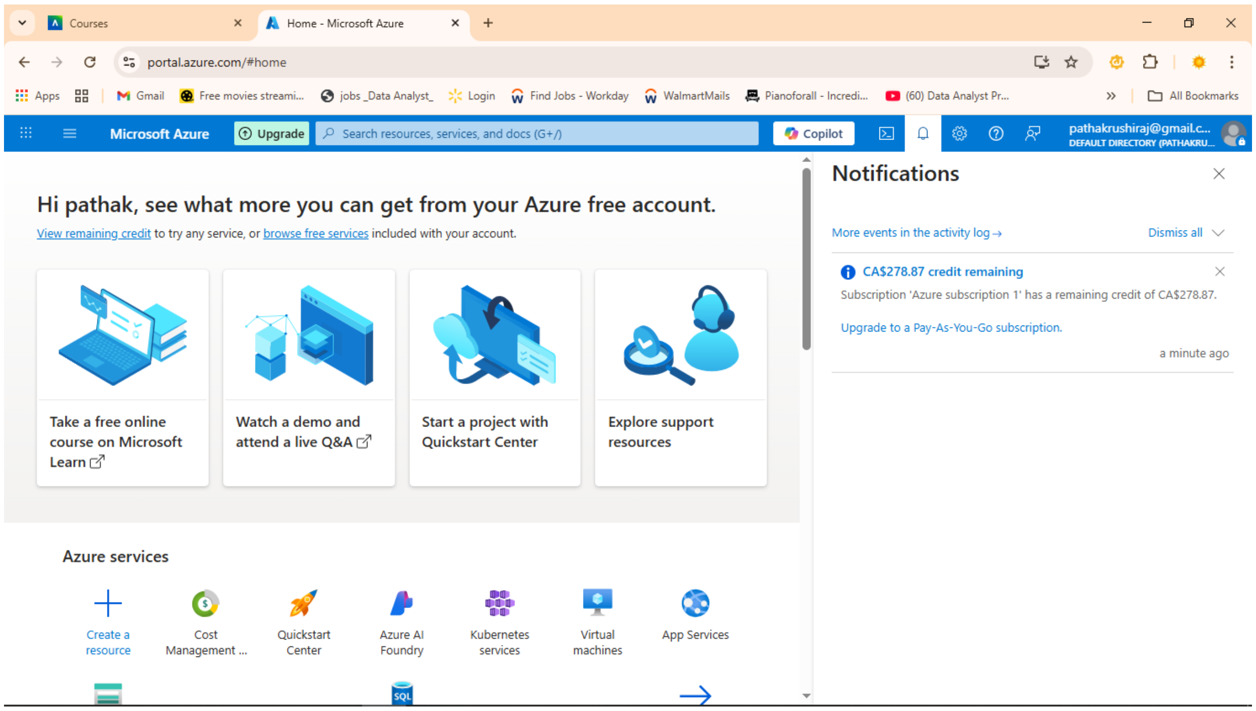
3. Tools and Technologies

Tool / Service	**Purpose**
Microsoft Azure Portal	Resource and billing management
Azure Storage Account (ADLS Gen2)	Cloud-based data storage
Azure Databricks	Data transformation and analysis using PySpark
PySpark	Distributed data processing framework
Delta Lake	Reliable and ACID-compliant data storage layer
Hive Metastore	Metadata management for SQL querying

4. Implementation Steps

Step 1 – Azure Setup

- Created a free Azure account with CA**$300 free credit**.
- Configured **billing profiles** and verified **active subscription**.


Courses
Home - Microsoft Azure
portal.azure.com/#home
Microsoft Azure
Upgrade
Search resources, services, and docs (G+/)
Copilot
pathakrushiraj@gmail.c...
DEFAULT DIRECTORY (PATHAKRU...
Take a free online course on Microsoft Learn
Watch a demo and attend a live Q&A
Start a project with Quickstart Center
Explore support resources
Azure services
Create a resource
Cost Management ...
Quickstart Center
Azure AI Foundry
Kubernetes services
Virtual machines
App Services
Notifications
More events in the activity log →
Dismiss all

Hi pathak, see what more you can get from your Azure free account.

View remaining credit to try any service, or browse free services included with your account.

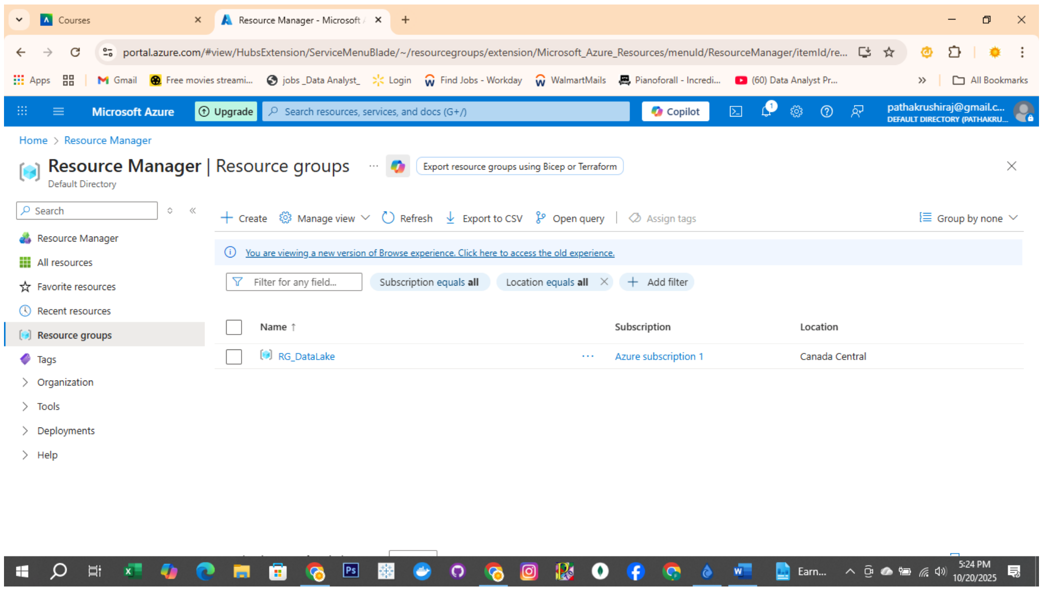
CA$278.87 credit remaining

Subscription 'Azure subscription 1' has a remaining credit of CA$278.87.

Upgrade to a Pay-As-You-Go subscription.

a minute ago

Figure 1Azure Portal home page showing "$300 free credit"

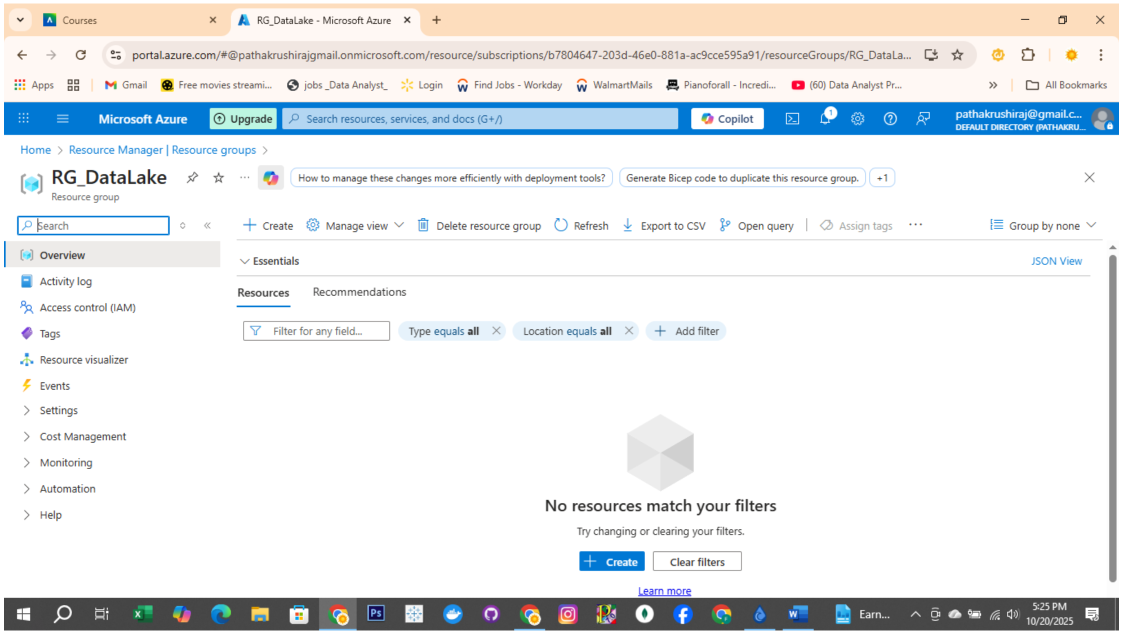


Resource Manager | Resource groups

Default Directory

Name	Subscription	Location
RG_DataLake	Azure subscription 1	Canada Central

Figure 2– Resource Group Creation Form



RG_DataLake

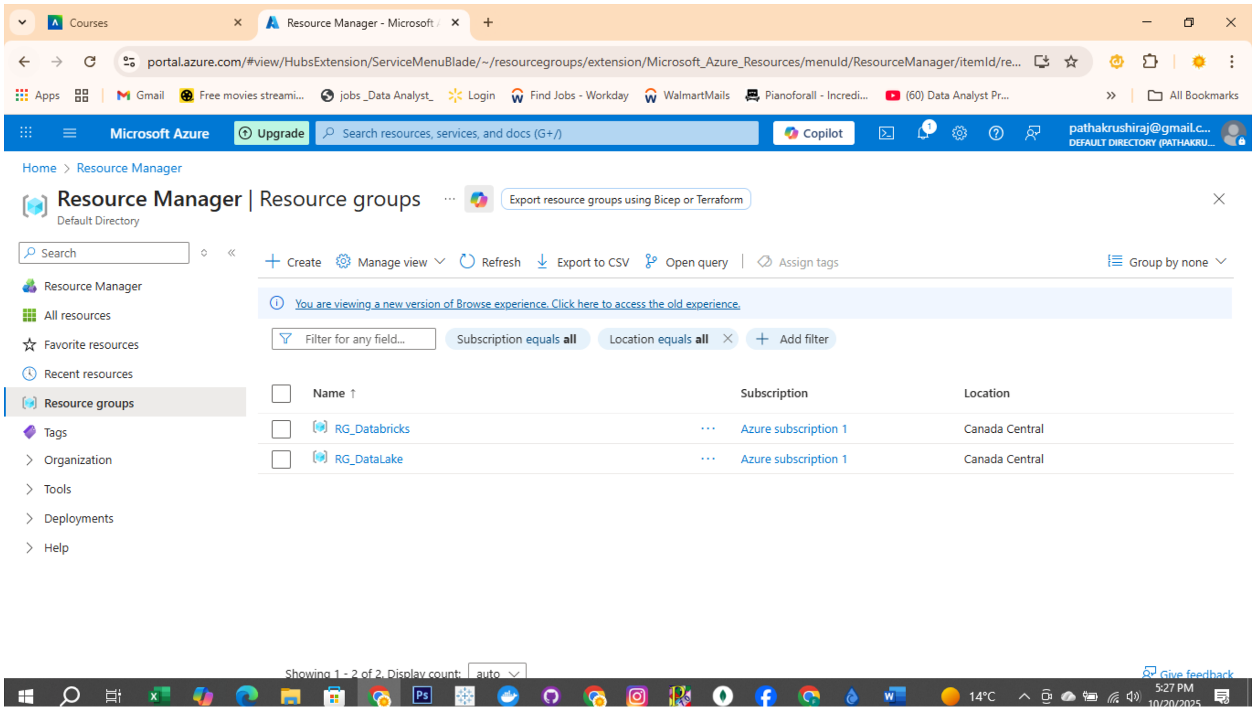
Resource group

No resources match your filters

Try changing or clearing your filters.

Figure 3RG_DataLake Overview Page

Step 2 – Storage Account Creation



Home > Resource Manager

Resource Manager | Resource groups

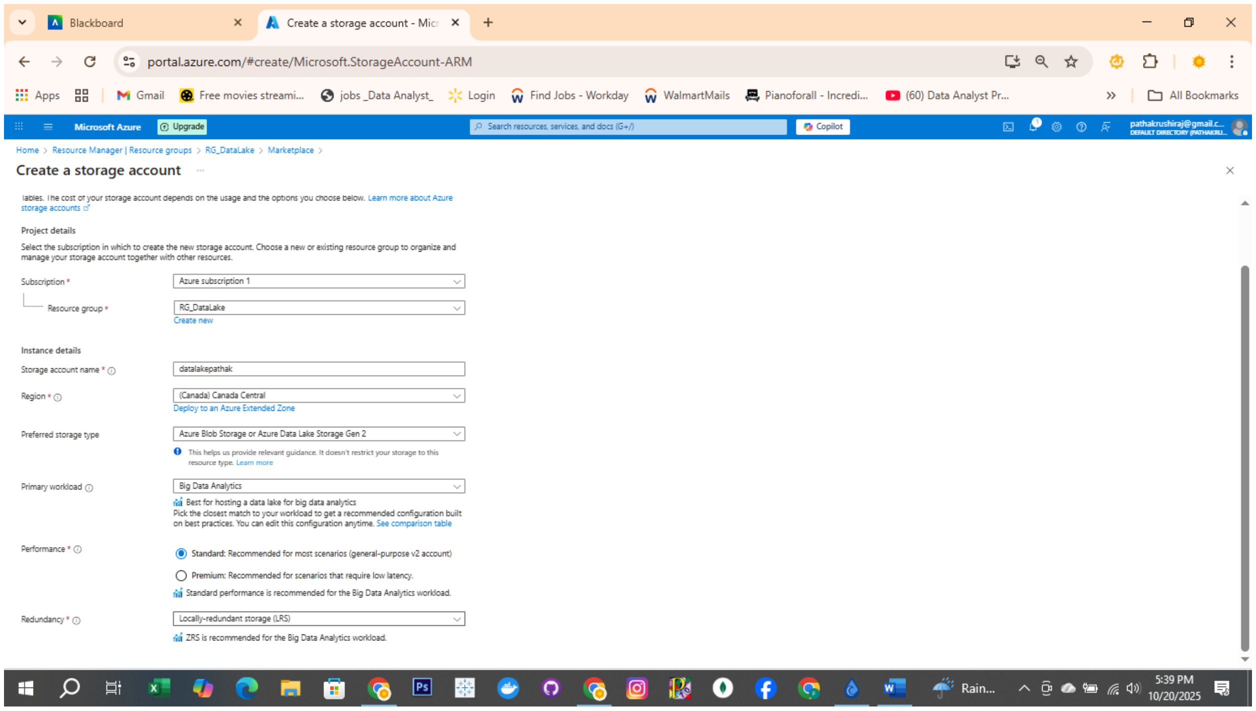
Default Directory

You are viewing a new version of Browse experience. Click here to access the old experience.

Name	Subscription	Location
RG_Databricks	Azure subscription 1	Canada Central
RG_DataLake	Azure subscription 1	Canada Central

Showing 1 - 2 of 2. Display count: auto

Figure 4Both Resource Groups Listed



Create a storage account

Tables. The cost of your storage account depends on the usage and the options you choose below. Learn more about Azure storage accounts

Project details

Select the subscription in which to create the new storage account. Choose a new or existing resource group to organize and manage your storage account together with other resources.

Subscription *	Azure subscription 1
Resource group *	RG_DataLake

Instance details

Storage account name *	datalakepathak
Region *	(Canada) Canada Central
Preferred storage type	Azure Blob Storage or Azure Data Lake Storage Gen 2
Primary workload	Big Data Analytics
Performance *	[✓ Standard: Recommended for most scenarios (general-purpose v2 account)] Premium: Recommended for scenarios that require low latency.
Redundancy *	Locally-redundant storage (LRS)

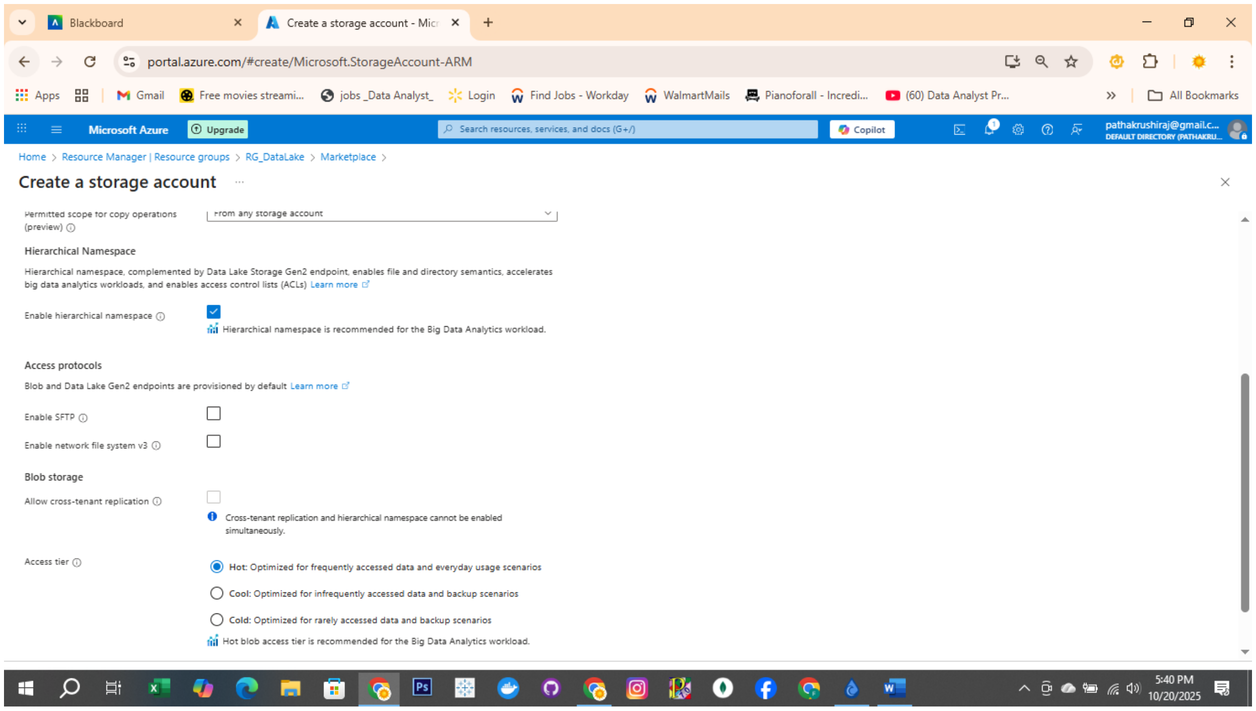
This helps us provide relevant guidance. It doesn't restrict your storage to this resource type. Learn more

Best for hosting a data lake for big data analytics
Pick the closest match to your workload to get a recommended configuration built on best practices. You can edit this configuration anytime. See comparison table

Standard performance is recommended for the Big Data Analytics workload.

ZRS is recommended for the Big Data Analytics workload.

Figure 5Storage Account Basics Tab Filled



Create a storage account

Permitted scope for copy operations (preview): From any storage account

Hierarchical Namespace

Hierarchical namespace, complemented by Data Lake Storage Gen2 endpoint, enables file and directory semantics, accelerates big data analytics workloads, and enables access control lists (ACLs) Learn more

Enable hierarchical namespace: ✓
Hierarchical namespace is recommended for the Big Data Analytics workload.

Access protocols

Blob and Data Lake Gen2 endpoints are provisioned by default Learn more

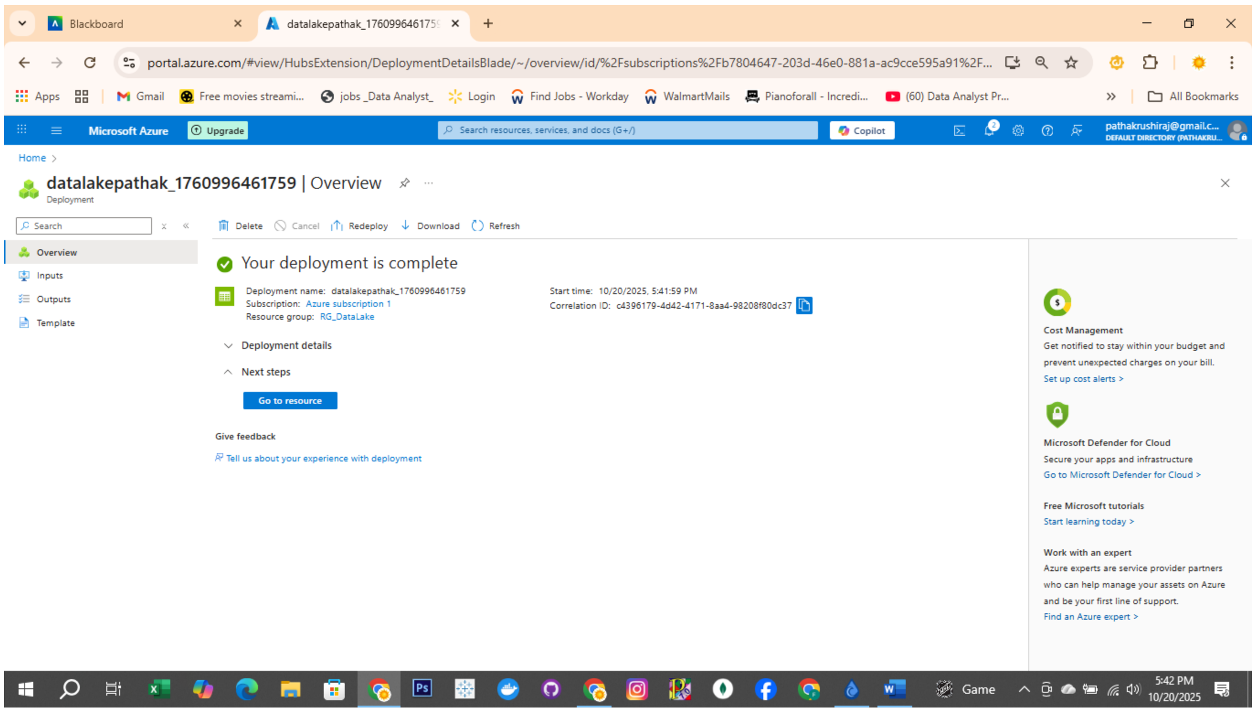
Enable SFTP: ☐
Enable network file system v3: ☐

Blob storage

Allow cross-tenant replication: ☐
Cross-tenant replication and hierarchical namespace cannot be enabled simultaneously.

Access tier: [✓ Hot: Optimized for frequently accessed data and everyday usage scenarios] Cool: Optimized for infrequently accessed data and backup scenarios; Cold: Optimized for rarely accessed data and backup scenarios
Hot blob access tier is recommended for the Big Data Analytics workload.

Figure 6Advanced Tab with Hierarchical Namespace Enabled



datalakepathak_1760996461759 | Overview

Deployment

Your deployment is complete

Deployment name: datalakepathak_1760996461759
Subscription: Azure subscription 1
Resource group: RG_DataLake

Start time: 10/20/2025, 5:41:59 PM
Correlation ID: c4396179-4d42-4171-8aa4-98208f80dc37

Deployment details

Next steps

Go to resource

Give feedback

Tell us about your experience with deployment

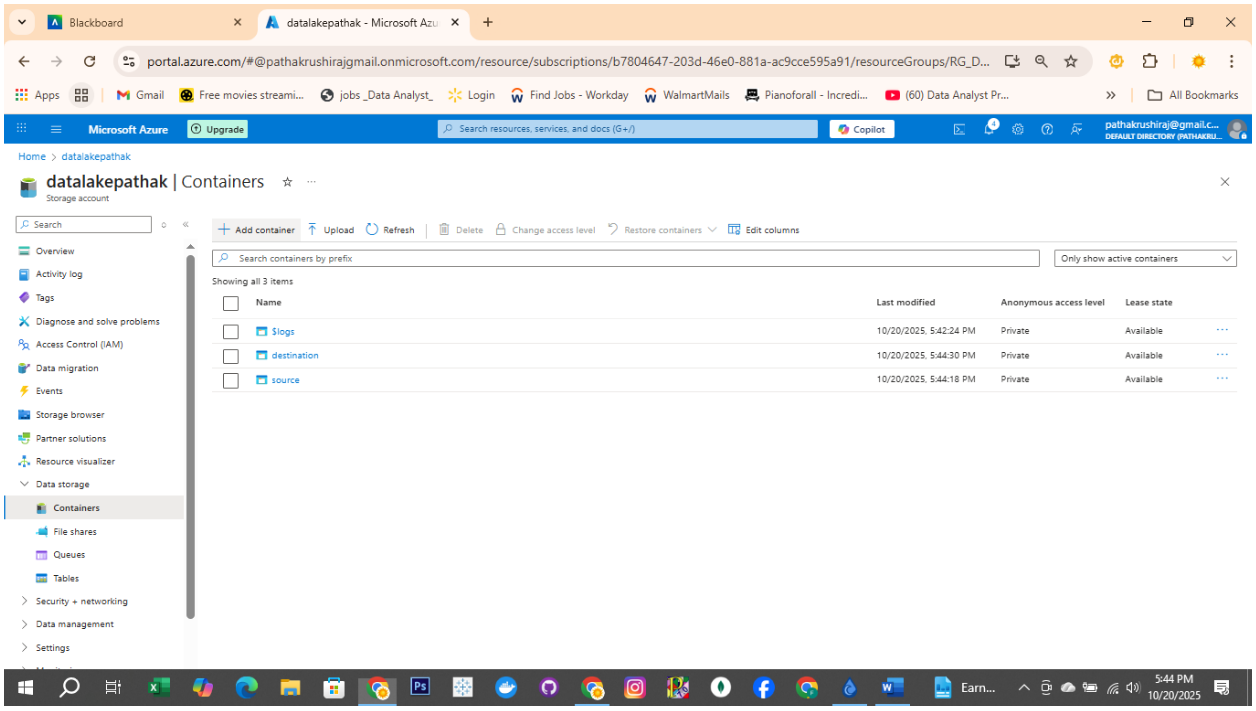
Cost Management
Get notified to stay within your budget and prevent unexpected charges on your bill.
Set up cost alerts >

Microsoft Defender for Cloud
Secure your apps and infrastructure
Go to Microsoft Defender for Cloud >

Free Microsoft tutorials
Start learning today >

Work with an expert
Azure experts are service provider partners who can manage your assets on Azure and be your first line of support.
Find an Azure expert >

Figure 7Storage Account Overview Page



datalakepathak | Containers

Storage account

Showing all 3 items

Name	Last modified	Anonymous access level	Lease state
$logs	10/20/2025, 5:42:24 PM	Private	Available
destination	10/20/2025, 5:44:30 PM	Private	Available
source	10/20/2025, 5:44:18 PM	Private	Available

Figure 8Containers List Page

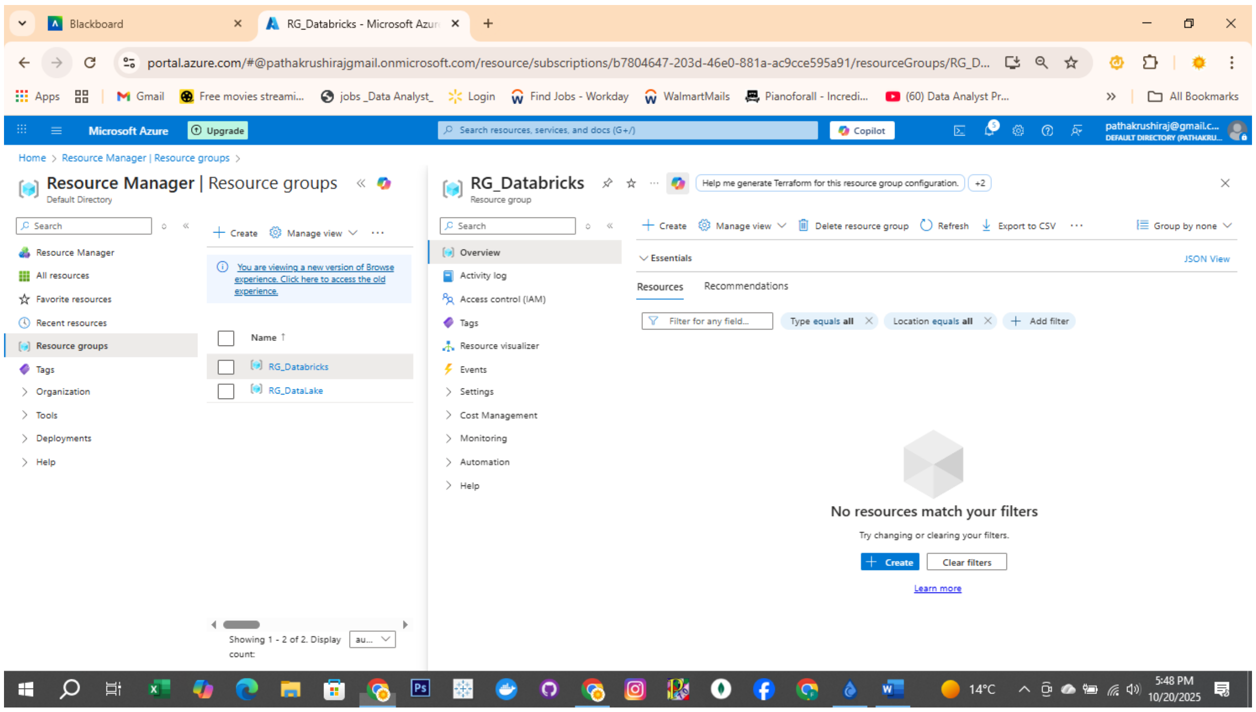
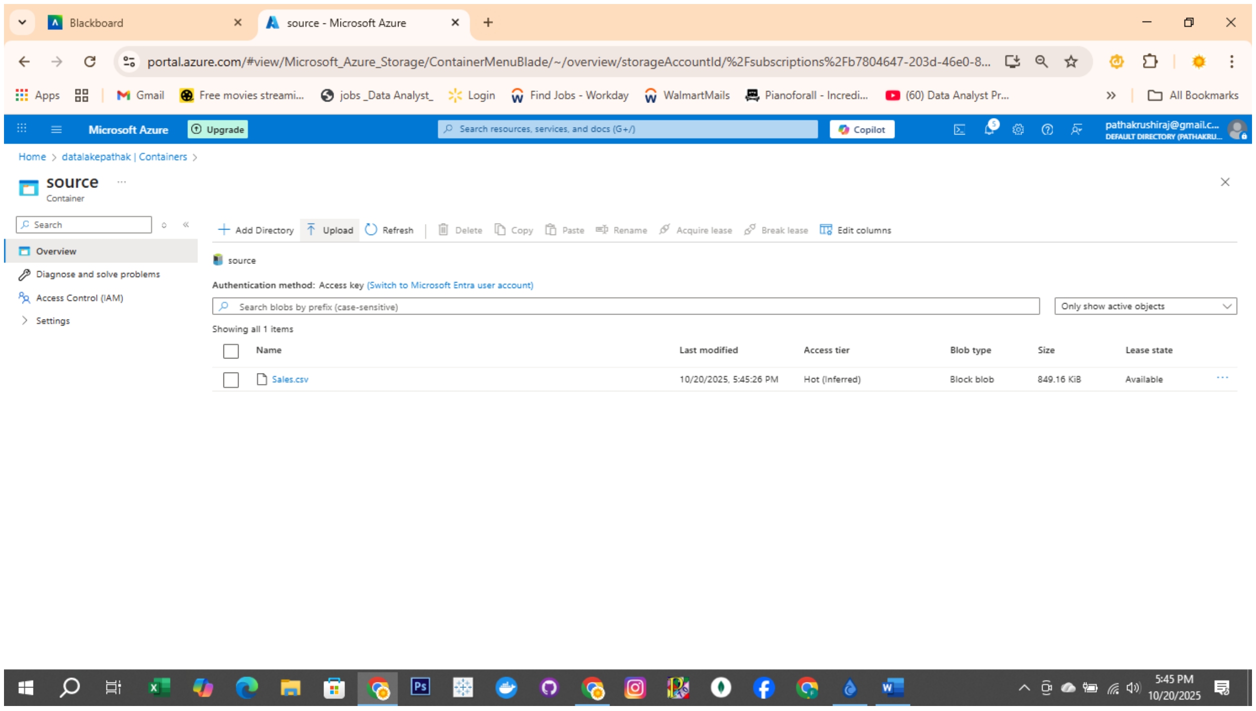


Figure 9RG_Databricks Overview Page



Blackboard
source - Microsoft Azure
portal.azure.com/#view/Microsoft_Azure_Storage/ContainerMenuBlade/~/overview/storageAccountId/%2Fsubscriptions%2Fb7804647-203d-46e0-8...
Microsoft Azure
Upgrade
Search resources, services, and docs (G+/)
Copilot
Home > datalakepathak | Containers
source
Container
Overview
Diagnose and solve problems
Access Control (IAM)
Settings
Add Directory
Upload
Refresh
Delete
Copy
Paste
Rename
Acquire lease
Break lease
Edit columns
source
Search blobs by prefix (case-sensitive)
Only show active objects
5:45 PM
10/20/2025

Authentication method: Access key (Switch to Microsoft Entra user account)

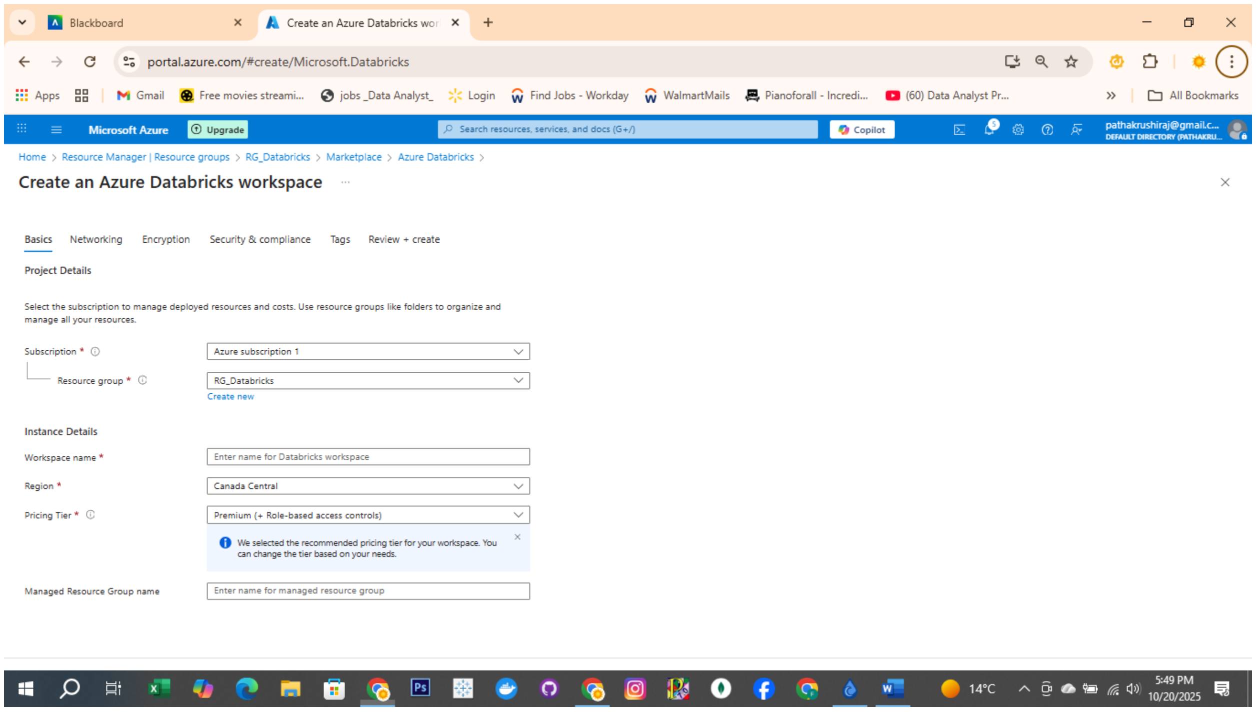
Showing all 1 items

Name	Last modified	Access tier	Blob type	Size	Lease state
Sales.csv	10/20/2025, 5:45:26 PM	Hot (Inferred)	Block blob	849.16 KiB	Available

Figure 10Uploaded sales.csv in Source Container

Step 3 – Databricks Workspace and Cluster

- Created a **Databricks workspace**.
- Configured a single-node cluster named **Cluster_SingleNode**.
- Attached the cluster to the notebook.


Blackboard
Create an Azure Databricks wor
portal.azure.com/#create/Microsoft.Databricks
Apps
Gmail
Free movies streami...
jobs _Data Analyst_
Login
Find Jobs - Workday
WalmartMails
Pianoforall - Incredi...
(60) Data Analyst Pr...
All Bookmarks
Microsoft Azure
Upgrade
Search resources, services, and docs (G+/)
Copilot
pathakrushiraj@gmail.c...
DEFAULT DIRECTORY (PATHAKRU...
Home > Resource Manager | Resource groups > RG_Databricks > Marketplace > Azure Databricks >

Create an Azure Databricks workspace

Basics | Networking | Encryption | Security & compliance | Tags | Review + create

Project Details

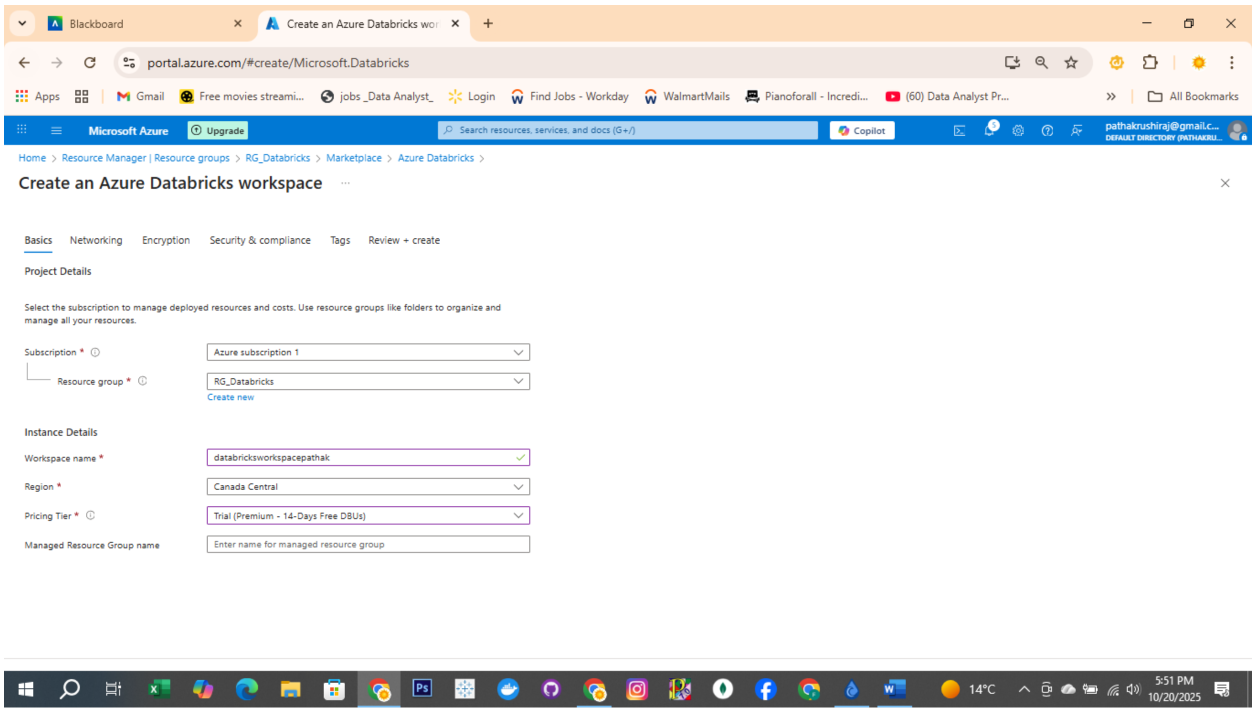
Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *	Azure subscription 1
Resource group *	RG_Databricks
	Create new

Instance Details

Workspace name *	Enter name for Databricks workspace
Region *	Canada Central
Pricing Tier *	Premium (+ Role-based access controls)
	We selected the recommended pricing tier for your workspace. You can change the tier based on your needs.
Managed Resource Group name	Enter name for managed resource group

Figure 11Azure Databricks Selection Screen



Create an Azure Databricks workspace

Basics | Networking | Encryption | Security & compliance | Tags | Review + create

Project Details

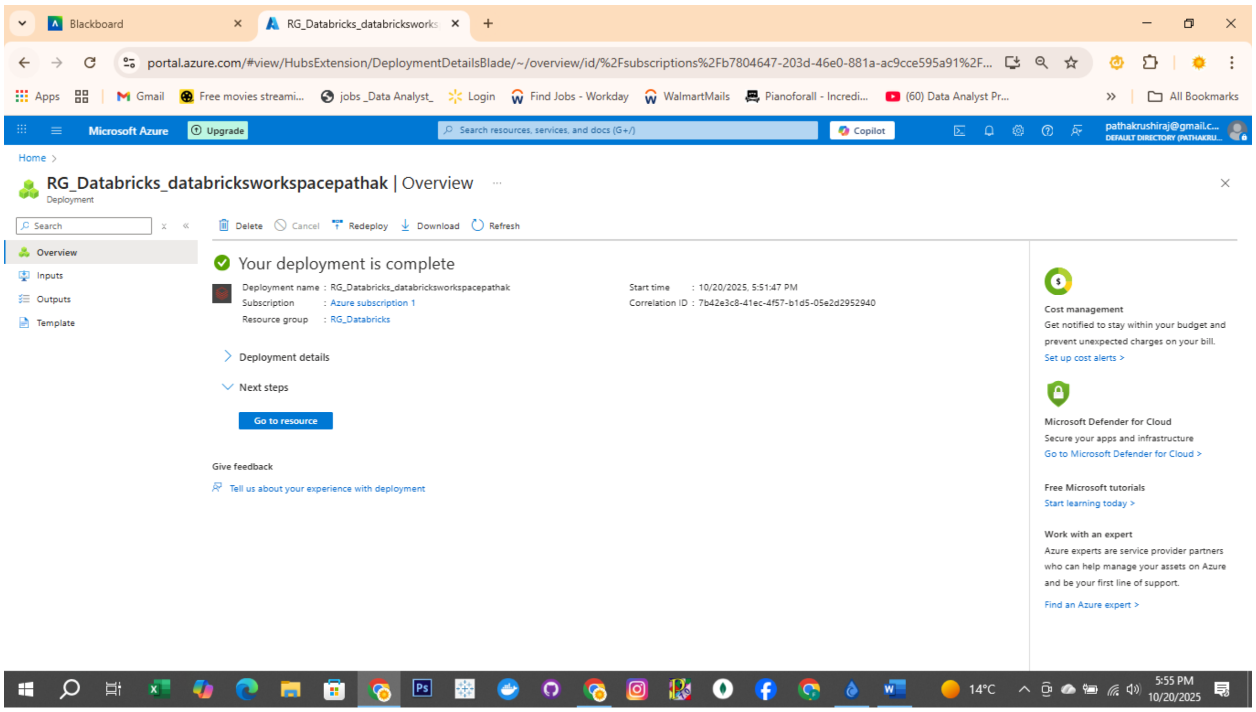
Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *	Azure subscription 1
Resource group *	RG_Databricks

Instance Details

Workspace name *	databricksworkspacepathak
Region *	Canada Central
Pricing Tier *	Trial (Premium - 14-Days Free DBUs)
Managed Resource Group name	Enter name for managed resource group

Figure 12Databricks Basics Tab Filled



RG_Databricks_databricksworkspacepathak | Overview

Deployment

Your deployment is complete

Deployment name	RG_Databricks_databricksworkspacepathak
Subscription	Azure subscription 1
Resource group	RG_Databricks
Start time	10/20/2025, 5:51:47 PM
Correlation ID	7b42e3c8-41ec-4f57-b1d5-05e2d2952940

Figure 13 Deployment Complete Page

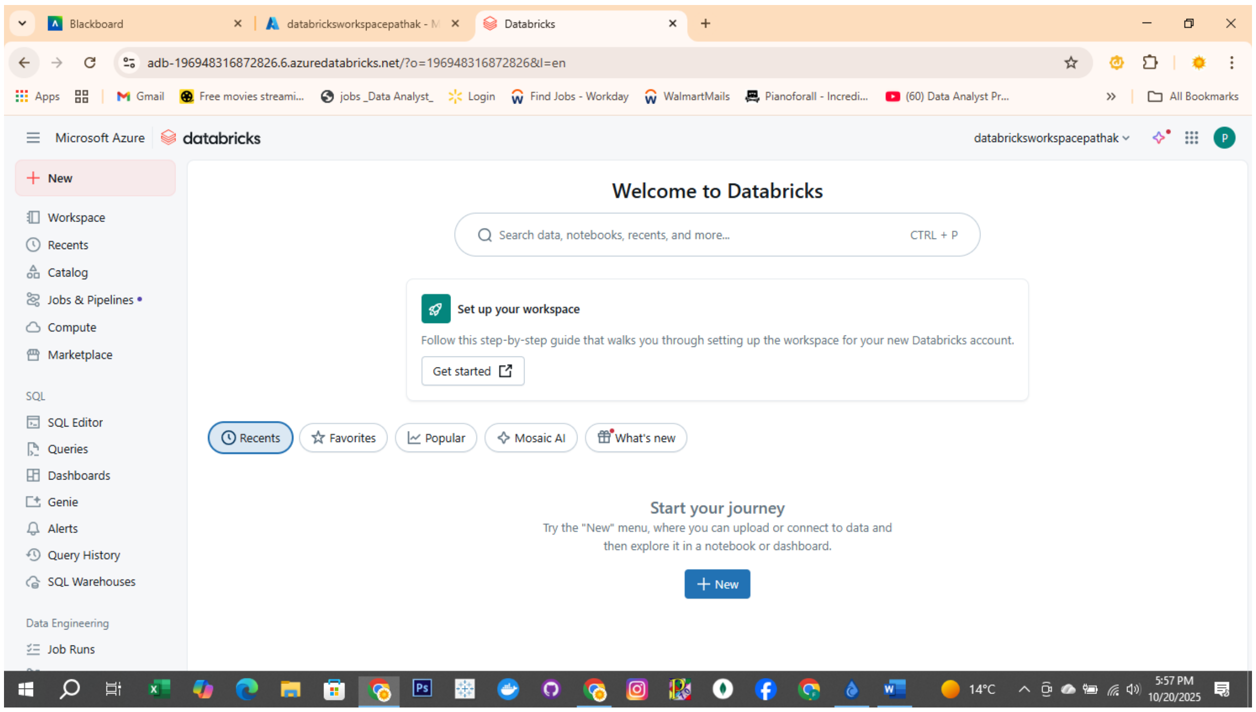

Welcome to Databricks
Search data, notebooks, recents, and more...
Set up your workspace
Follow this step-by-step guide that walks you through setting up the workspace for your new Databricks account.
Get started
Recents
Favorites
Popular
Mosaic AI
What's new
Start your journey
Try the "New" menu, where you can upload or connect to data and then explore it in a notebook or dashboard.
New

Figure 14Databricks Home Page (After Launch)

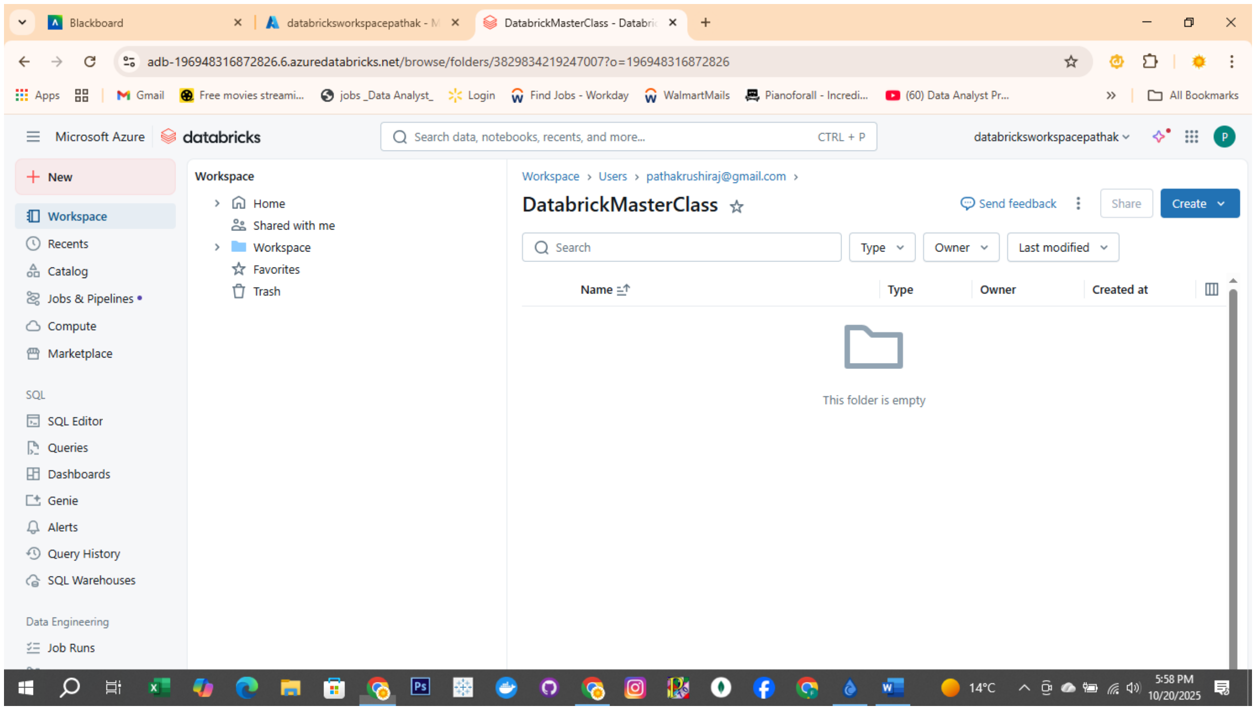
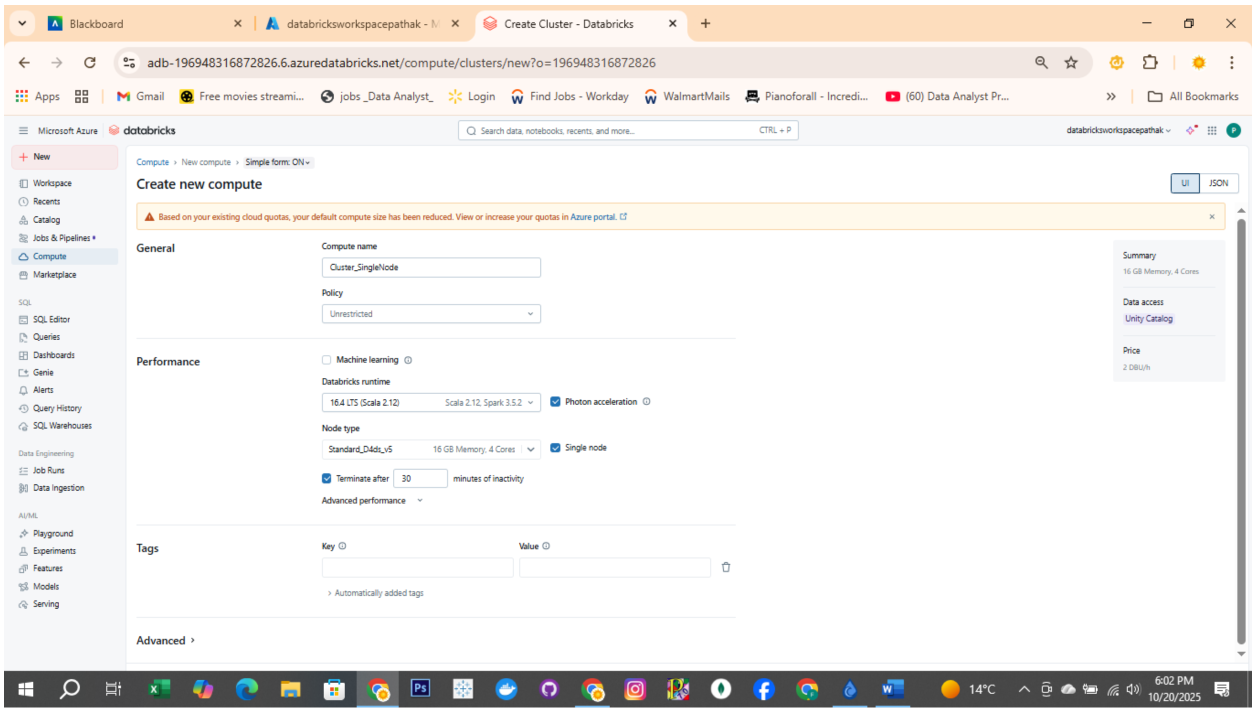


Figure 15Workspace Folder Created


Create new compute

Based on your existing cloud quotas, your default compute size has been reduced. View or increase your quotas in Azure portal.

General

Compute name: Cluster_SingleNode

Policy: Unrestricted

Performance

Machine learning

Databricks runtime: 16.4 LTS (Scala 2.12) Scala 2.12, Spark 3.5.2

Photon acceleration

Node type: Standard_D4ds_v5 16 GB Memory, 4 Cores

Single node

Terminate after 30 minutes of inactivity

Advanced performance

Tags

Key | Value

Automatically added tags

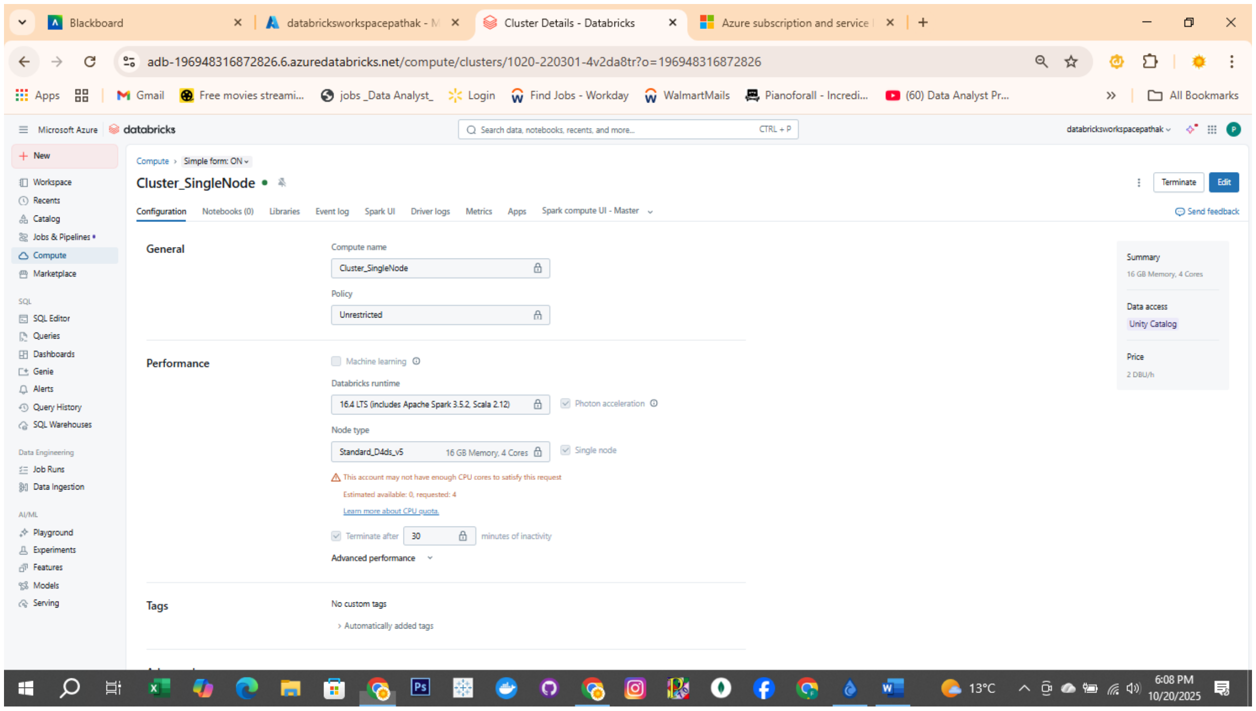
Advanced

Summary: 16 GB Memory, 4 Cores

Data access: Unity Catalog

Price: 2 DBU/h

Figure 16Cluster Configuration Filled


Cluster_SingleNode

Configuration | Notebooks (0) | Libraries | Event log | Spark UI | Driver logs | Metrics | Apps | Spark compute UI - Master

General

Compute name: Cluster_SingleNode

Policy: Unrestricted

Performance

Machine learning

Databricks runtime: 16.4 LTS (includes Apache Spark 3.5.2, Scala 2.12)

Photon acceleration

Node type: Standard_D4ds_v5 16 GB Memory, 4 Cores

Single node

This account may not have enough CPU cores to satisfy this request

Estimated available: 0, requested: 4

Learn more about CPU quota.

Terminate after 30 minutes of inactivity

Advanced performance

Tags

No custom tags

Automatically added tags

Summary: 16 GB Memory, 4 Cores

Data access: Unity Catalog

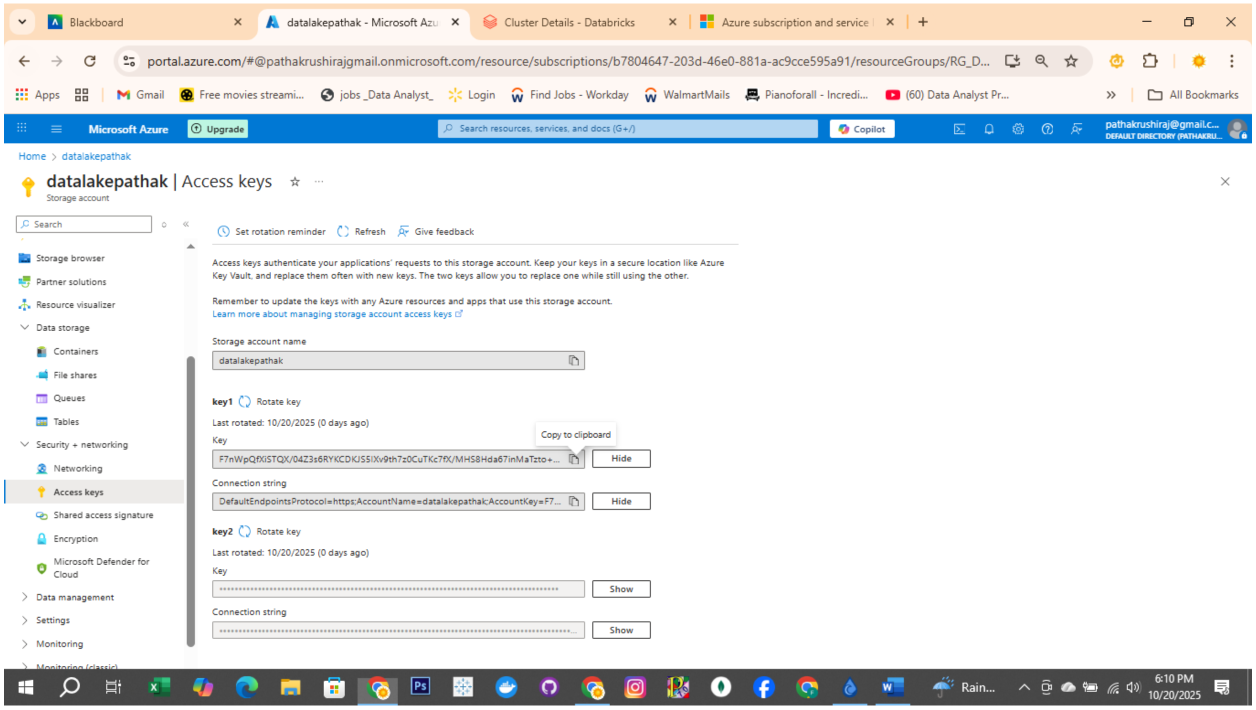
Price: 2 DBU/h

Figure 17Active Cluster (Green Indicator)

Step 4 – Connecting Databricks to Azure Data Lake

- Used **access keys** for secure ABFSS (Azure Blob File System) connection.

```
spark.conf.set(
  "fs.azure.account.key.datalakepathak.dfs.core.windows.net",
  "<your-storage-access-key>"
)
```



datalakepathak | Access keys

Storage account

Access keys authenticate your applications' requests to this storage account. Keep your keys in a secure location like Azure Key Vault, and replace them often with new keys. The two keys allow you to replace one while still using the other.

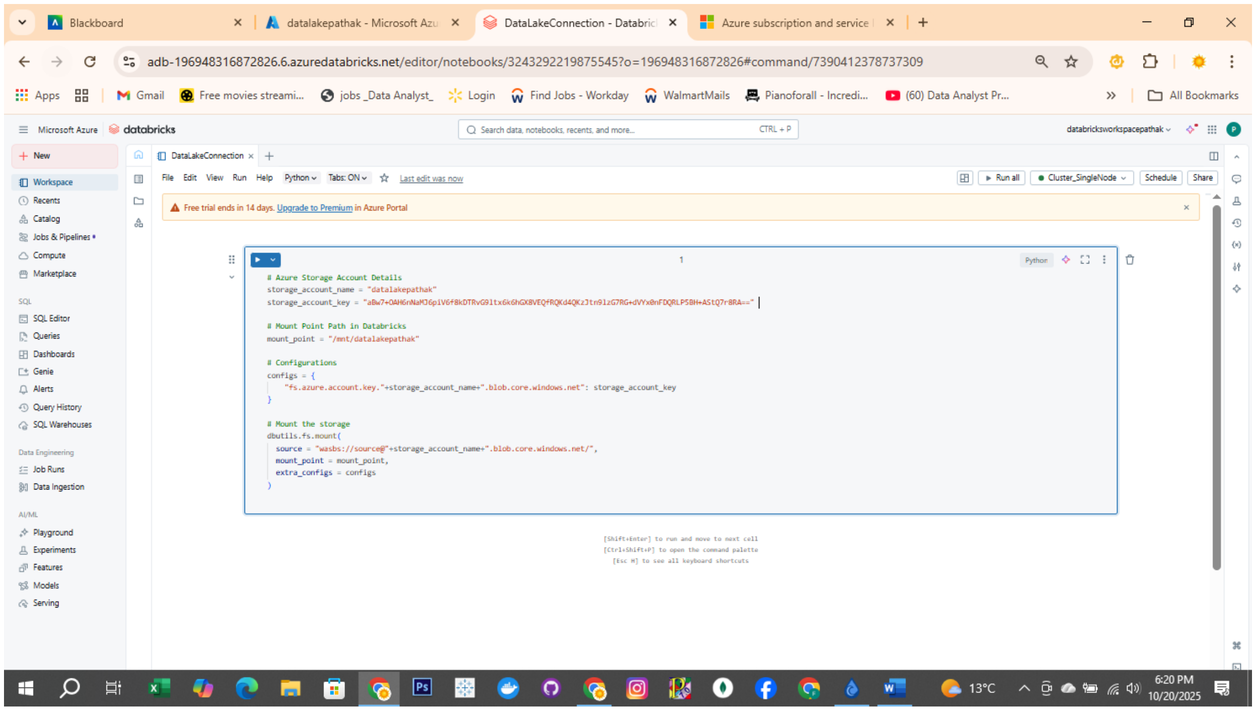
Remember to update the keys with any Azure resources and apps that use this storage account.
Learn more about managing storage account access keys

Storage account name: datalakepathak

key1 Rotate key
Last rotated: 10/20/2025 (0 days ago)
Key: F7nWpQfXiSTQX/04Z3s6RYKCDKJS5IXv9th7z0CuTKc7fX/MHS8Hda67inMaTzto+...
Connection string: DefaultEndpointsProtocol=https;AccountName=datalakepathak;AccountKey=F7...

key2 Rotate key
Last rotated: 10/20/2025 (0 days ago)
Key: ••
Connection string: ••...

Figure 18Access Keys Page



```python
# Azure Storage Account Details
storage_account_name = "datalakepathak"
storage_account_key = "aBw7+OAH6nNaMJ6piV6f8kDTRvG9ltx6k6hGX8VEQfRQKd4QKzJtn9lzG7RG+dVYx0nFDQRLP5BH+AStQ7r8RA=="

# Mount Point Path in Databricks
mount_point = "/mnt/datalakepathak"

# Configurations
configs = {
    "fs.azure.account.key."+storage_account_name+".blob.core.windows.net": storage_account_key
}

# Mount the storage
dbutils.fs.mount(
  source = "wasbs://source@"+storage_account_name+".blob.core.windows.net/",
  mount_point = mount_point,
  extra_configs = configs
)
```

Figure 19Code in Notebook

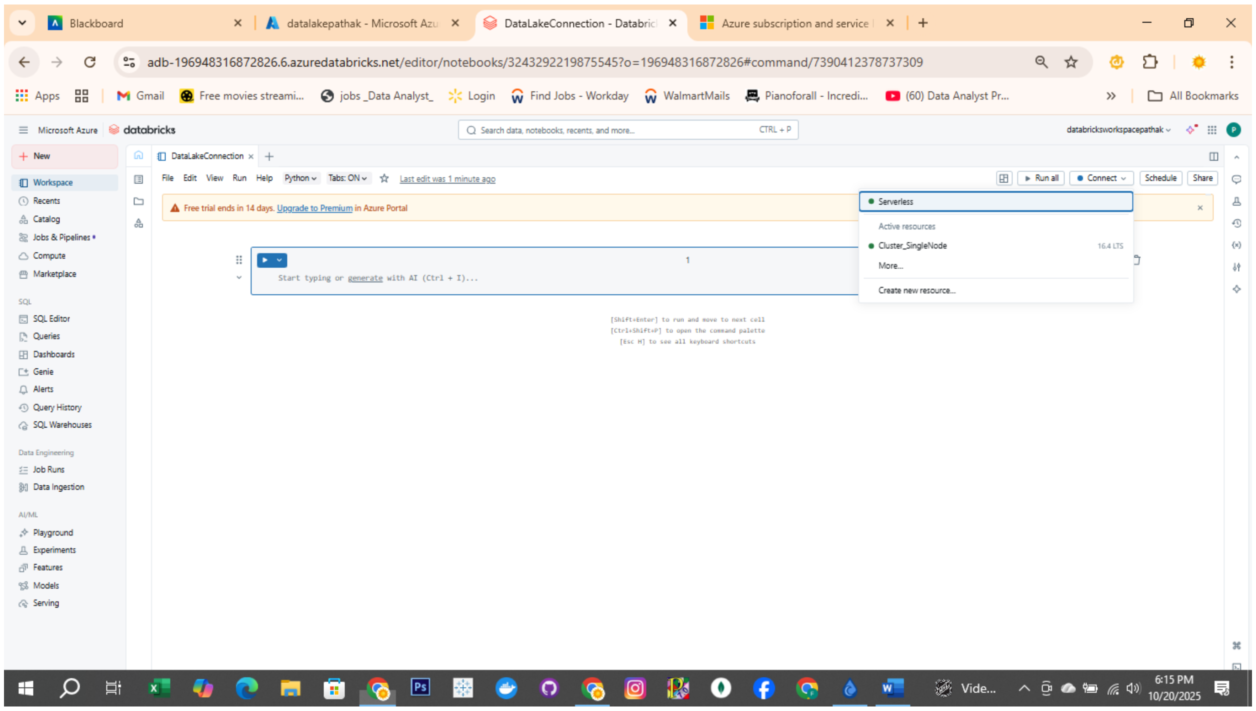
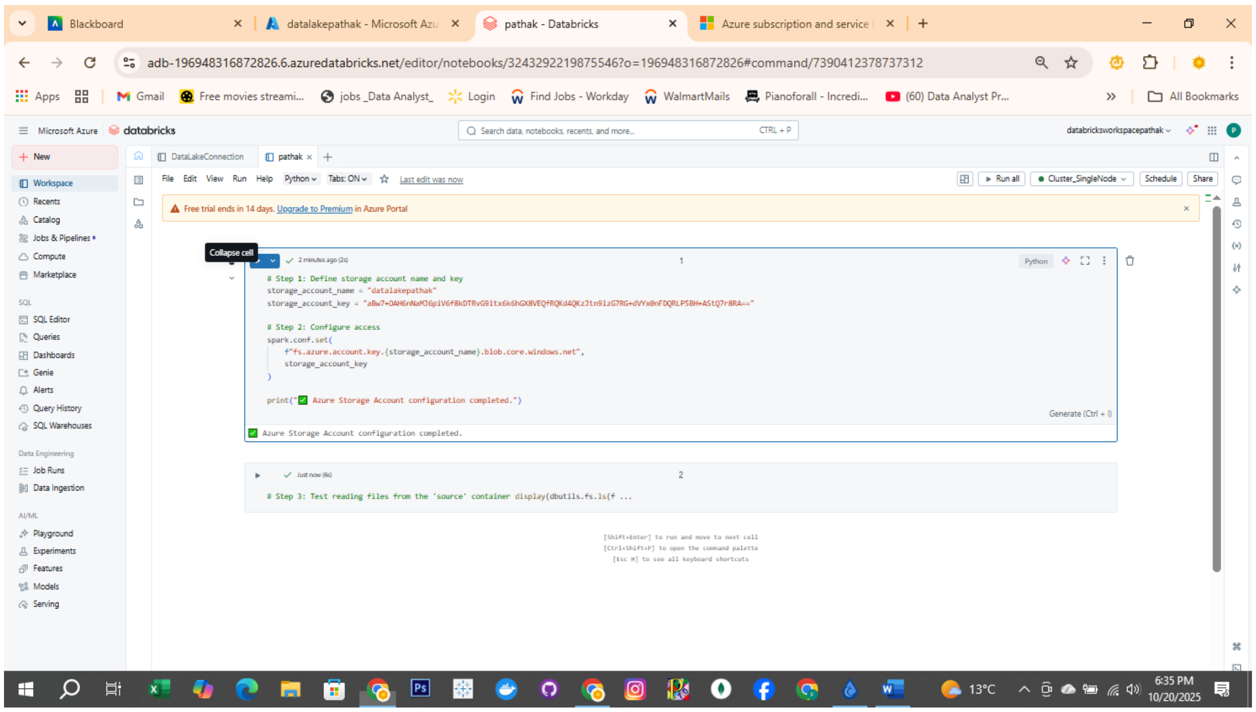


Figure 20Notebook Creation Popup

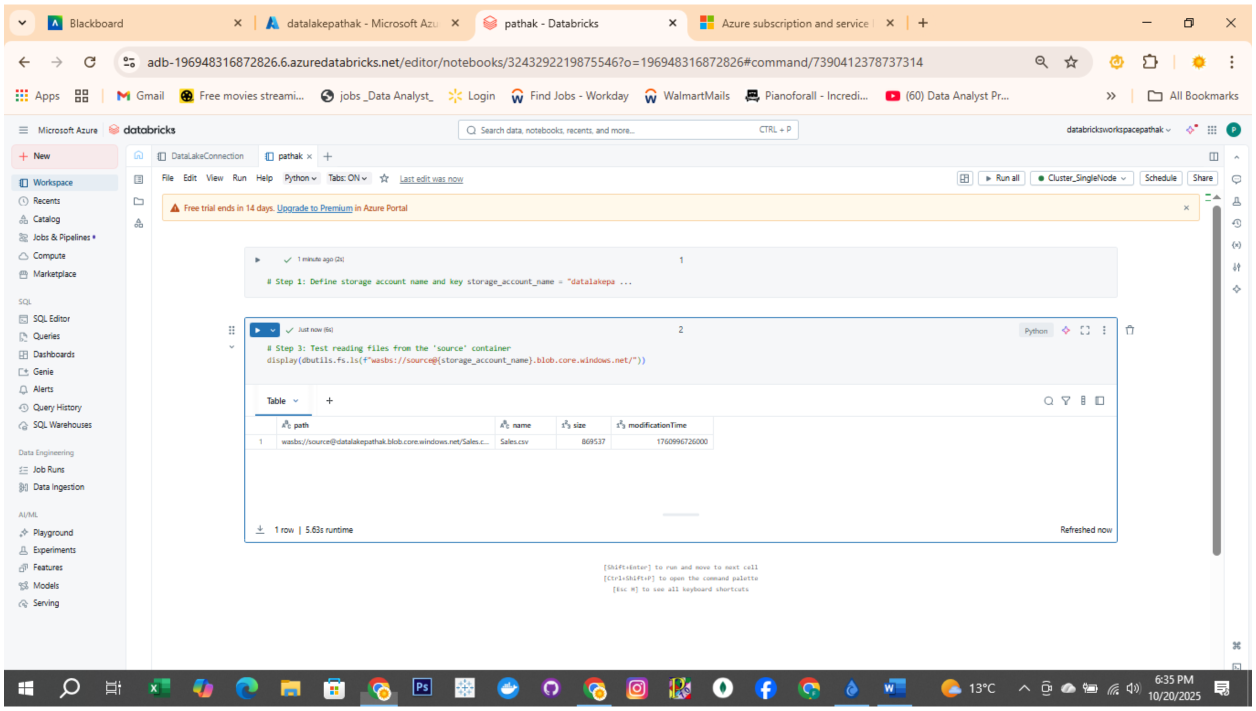


```
# Step 1: Define storage account name and key
storage_account_name = "datalakepathak"
storage_account_key = "aBw7+OAH6nNaMJ6piV6f8kDTRvG91tx6k6hGX8VEQfRQKd4QKzJtn91zG7RG+dVYx0nFDQRLP5BH+AStQ7r8RA=="

# Step 2: Configure access
spark.conf.set(
    f"fs.azure.account.key.{storage_account_name}.blob.core.windows.net",
    storage_account_key
)

print("✅ Azure Storage Account configuration completed.")
```

```
✅ Azure Storage Account configuration completed.
```

```
# Step 3: Test reading files from the 'source' container display(dbutils.fs.ls(f ...
```

Figure 21Notebook Code Cell with Configuration



```
# Step 1: Define storage account name and key storage_account_name = "datalakepa ...
```

```
# Step 3: Test reading files from the 'source' container
display(dbutils.fs.ls(f"wasbs://source@{storage_account_name}.blob.core.windows.net/"))
```

	path	name	size	modificationTime
1	wasbs://source@datalakepathak.blob.core.windows.net/Sales.c...	Sales.csv	869537	1760996726000

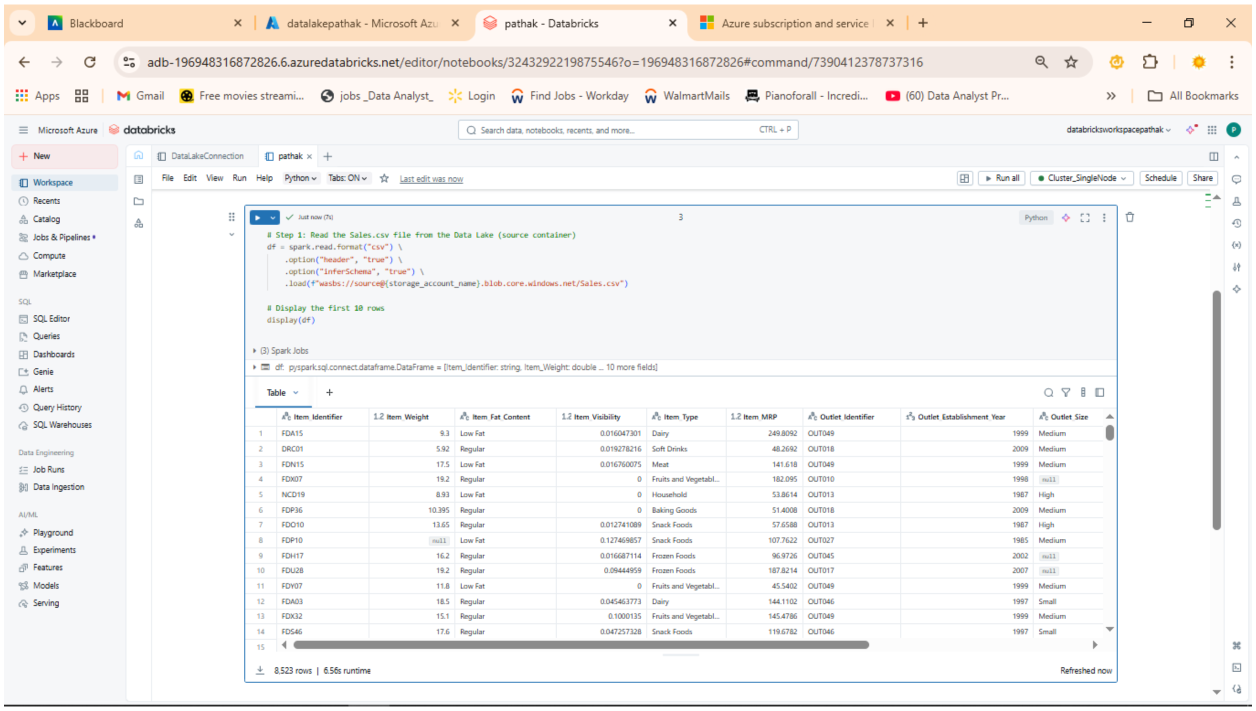
1 row | 5.63s runtime

Figure 22Output showing sales.csv inside source container

Step 5 – Data Ingestion and Cleaning

- Read the Sales.csv file from the **source** container.
- Removed null values and duplicates using PySpark functions.

df_cleaned = df.dropna().dropDuplicates()



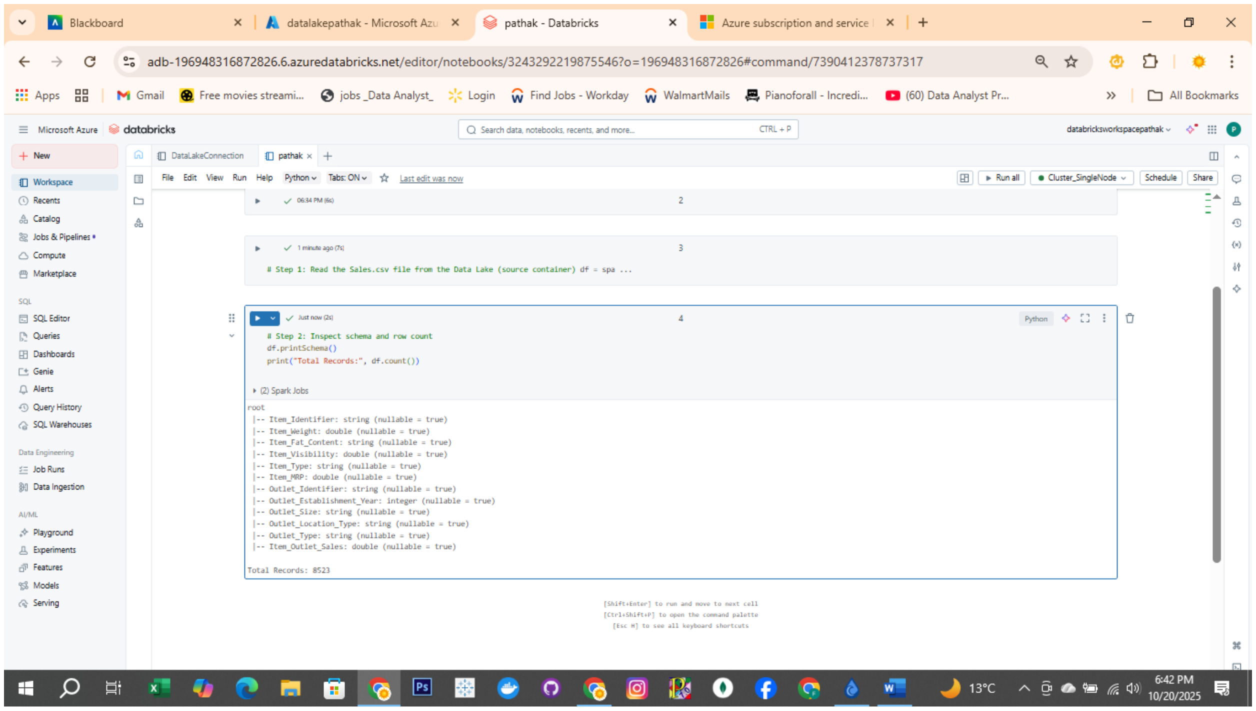
```python
# Step 1: Read the Sales.csv file from the Data Lake (source container)
df = spark.read.format("csv") \
    .option("header", "true") \
    .option("inferSchema", "true") \
    .load(f"wasbs://source@{storage_account_name}.blob.core.windows.net/Sales.csv")

# Display the first 10 rows
display(df)
```

```
(3) Spark Jobs
df: pyspark.sql.connect.dataframe.DataFrame = [Item_Identifier: string, Item_Weight: double ... 10 more fields]
```

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size
1	FDA15	9.3	Low Fat	0.016047301	Dairy	249.8092	OUT049	1999	Medium
2	DRC01	5.92	Regular	0.019278216	Soft Drinks	48.2692	OUT018	2009	Medium
3	FDN15	17.5	Low Fat	0.016760075	Meat	141.618	OUT049	1999	Medium
4	FDX07	19.2	Regular	0	Fruits and Vegetabl...	182.095	OUT010	1998	null
5	NCD19	8.93	Low Fat	0	Household	53.8614	OUT013	1987	High
6	FDP36	10.395	Regular	0	Baking Goods	51.4008	OUT018	2009	Medium
7	FDO10	13.65	Regular	0.012741089	Snack Foods	57.6588	OUT013	1987	High
8	FDP10	null	Low Fat	0.127469857	Snack Foods	107.7622	OUT027	1985	Medium
9	FDH17	16.2	Regular	0.016687114	Frozen Foods	96.9726	OUT045	2002	null
10	FDU28	19.2	Regular	0.09444959	Frozen Foods	187.8214	OUT017	2007	null
11	FDY07	11.8	Low Fat	0	Fruits and Vegetabl...	45.5402	OUT049	1999	Medium
12	FDA03	18.5	Regular	0.045463773	Dairy	144.1102	OUT046	1997	Small
13	FDX32	15.1	Regular	0.1000135	Fruits and Vegetabl...	145.4786	OUT049	1999	Medium
14	FDS46	17.6	Regular	0.047257328	Snack Foods	119.6782	OUT046	1997	Small
15									

8,523 rows | 6.56s runtime Refreshed now

Figure 23Raw Data Loaded into Spark DataFrame

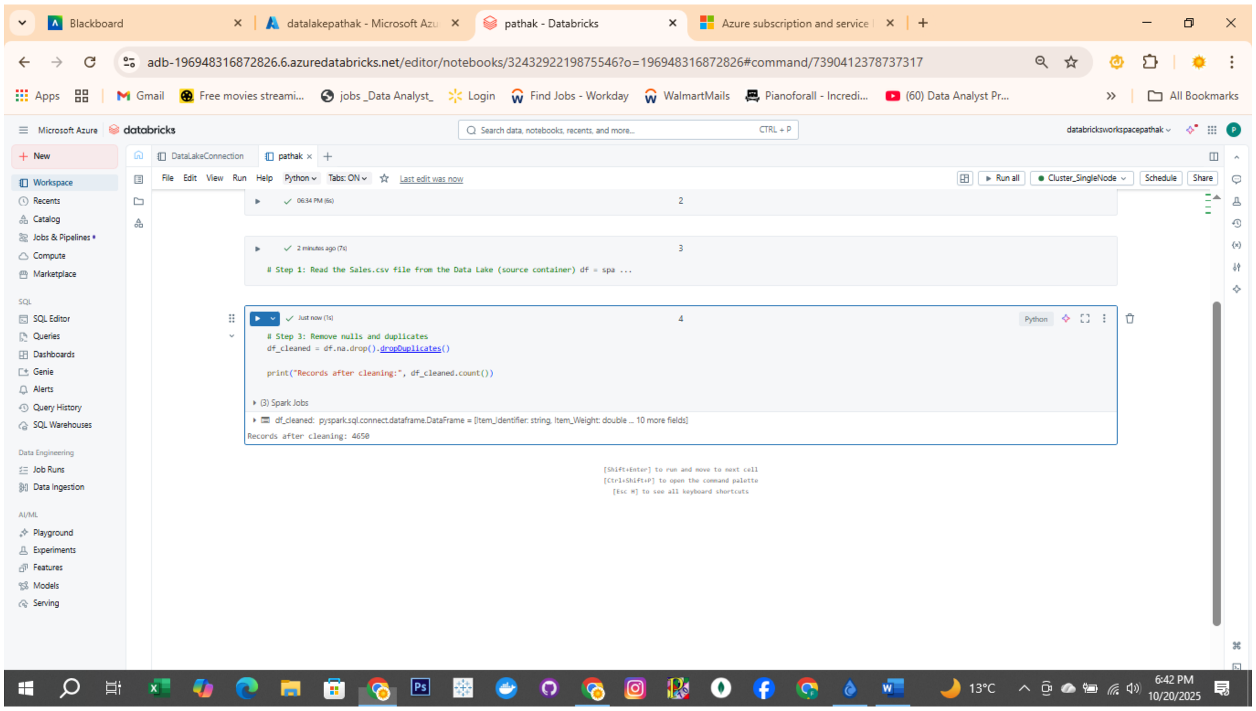


```
# Step 1: Read the Sales.csv file from the Data Lake (source container) df = spa ...
```

```
# Step 2: Inspect schema and row count
df.printSchema()
print("Total Records:", df.count())
```

```
root
 |-- Item_Identifier: string (nullable = true)
 |-- Item_Weight: double (nullable = true)
 |-- Item_Fat_Content: string (nullable = true)
 |-- Item_Visibility: double (nullable = true)
 |-- Item_Type: string (nullable = true)
 |-- Item_MRP: double (nullable = true)
 |-- Outlet_Identifier: string (nullable = true)
 |-- Outlet_Establishment_Year: integer (nullable = true)
 |-- Outlet_Size: string (nullable = true)
 |-- Outlet_Location_Type: string (nullable = true)
 |-- Outlet_Type: string (nullable = true)
 |-- Item_Outlet_Sales: double (nullable = true)

Total Records: 8523
```

Figure 24Schema and Record Count



```
# Step 1: Read the Sales.csv file from the Data Lake (source container) df = spa ...
```

```
# Step 3: Remove nulls and duplicates
df_cleaned = df.na.drop().dropDuplicates()

print("Records after cleaning:", df_cleaned.count())
```

```
df_cleaned:  pyspark.sql.connect.dataframe.DataFrame = [Item_Identifier: string, Item_Weight: double ... 10 more fields]
Records after cleaning: 4650
```

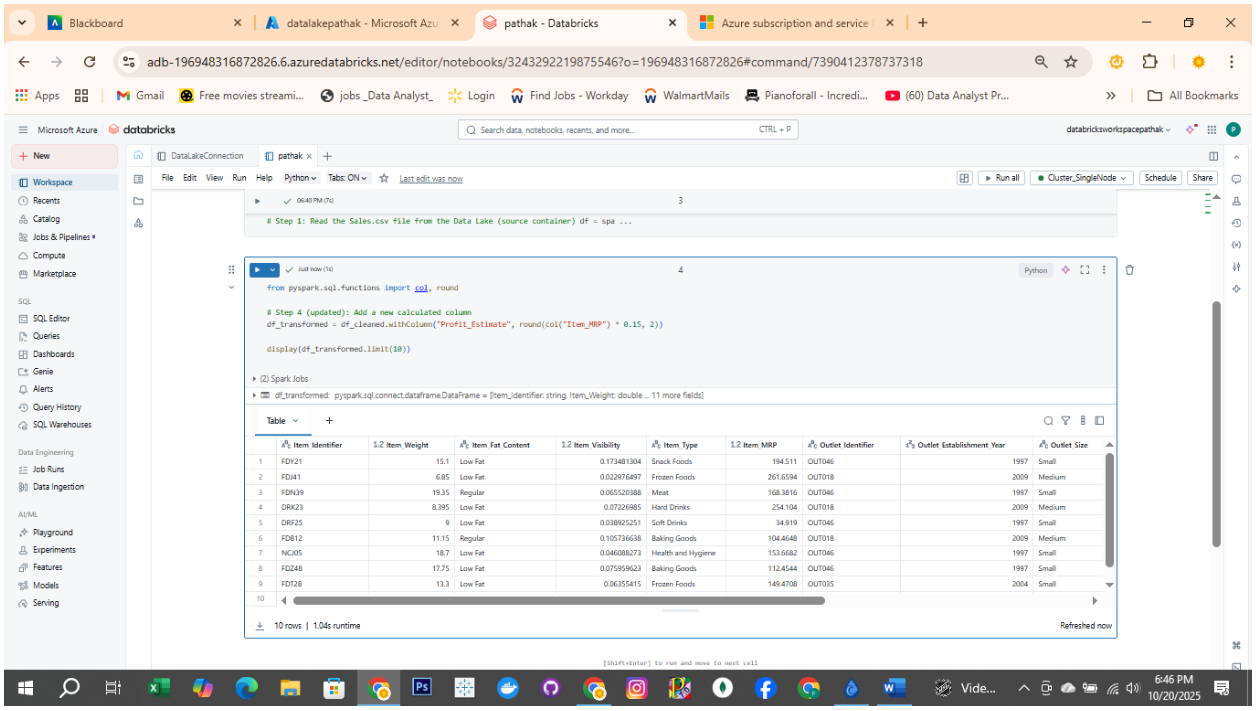
Figure 25Cleaning Step Results

Step 6 – Data Transformation

- Grouped data by Outlet_Type and calculated total sales.

from pyspark.sql.functions import sum

df_transformed = df_cleaned.groupBy("Outlet_Type").agg(sum("Item_Outlet_Sales").alias("Total_Sales"))



```
# Step 1: Read the Sales.csv file from the Data Lake (source container) df = spa ...
```

```
from pyspark.sql.functions import col, round

# Step 4 (updated): Add a new calculated column
df_transformed = df_cleaned.withColumn("Profit_Estimate", round(col("Item_MRP") * 0.15, 2))

display(df_transformed.limit(10))
```

(2) Spark Jobs

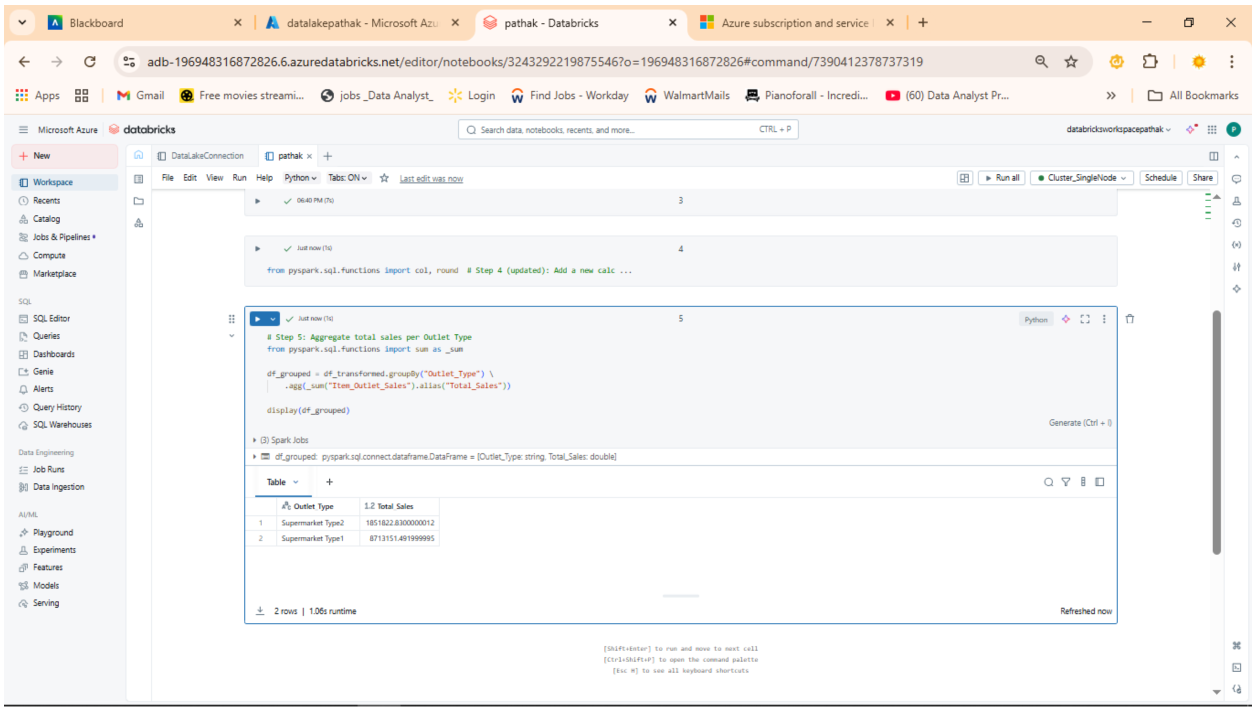
df_transformed: pyspark.sql.connect.dataframe.DataFrame = [Item_Identifier: string, Item_Weight: double ... 11 more fields]

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size
1	FDY21	15.1	Low Fat	0.173481304	Snack Foods	194.511	OUT046	1997	Small
2	FDJ41	6.85	Low Fat	0.022976497	Frozen Foods	261.6594	OUT018	2009	Medium
3	FDN39	19.35	Regular	0.065520388	Meat	168.3816	OUT046	1997	Small
4	DRK23	8.395	Low Fat	0.07226985	Hard Drinks	254.104	OUT018	2009	Medium
5	DRF25	9	Low Fat	0.038925251	Soft Drinks	34.919	OUT046	1997	Small
6	FDB12	11.15	Regular	0.105736638	Baking Goods	104.4648	OUT018	2009	Medium
7	NCJ05	18.7	Low Fat	0.046088273	Health and Hygiene	153.6682	OUT046	1997	Small
8	FDZ48	17.75	Low Fat	0.075959623	Baking Goods	112.4544	OUT046	1997	Small
9	FDT28	13.3	Low Fat	0.06355415	Frozen Foods	149.4708	OUT035	2004	Small
10									

10 rows | 1.04s runtime

Refreshed now

Figure 26DataFrame with Profit_Estimate Column



```
from pyspark.sql.functions import col, round  # Step 4 (updated): Add a new calc ...
```

```
# Step 5: Aggregate total sales per Outlet Type
from pyspark.sql.functions import sum as _sum

df_grouped = df_transformed.groupBy("Outlet_Type") \
    .agg(_sum("Item_Outlet_Sales").alias("Total_Sales"))

display(df_grouped)
```

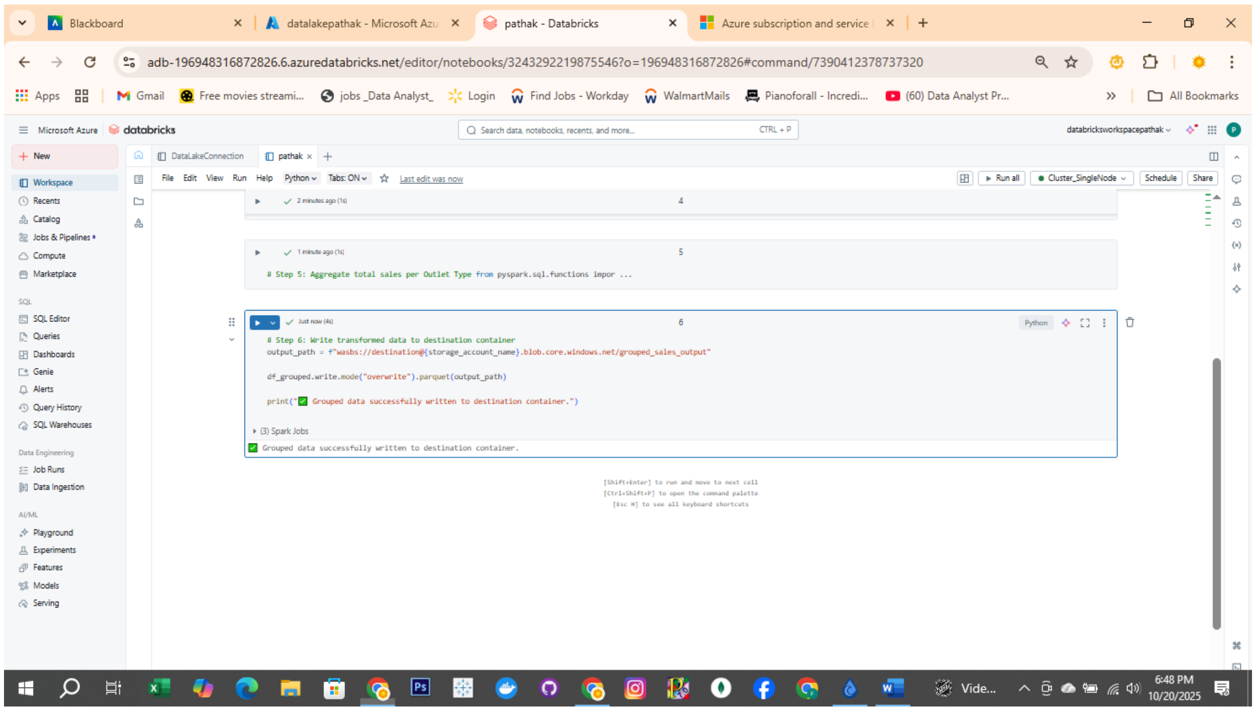
(3) Spark Jobs

df_grouped: pyspark.sql.connect.dataframe.DataFrame = [Outlet_Type: string, Total_Sales: double]

	Outlet_Type	Total_Sales
1	Supermarket Type2	1851822.8300000012
2	Supermarket Type1	8713151.491999995

2 rows | 1.06s runtime

Figure 27Grouped Total Sales by Outlet Type



```
# Step 5: Aggregate total sales per Outlet Type from pyspark.sql.functions impor ...
```

```
# Step 6: Write transformed data to destination container
output_path = f"wasbs://destination@{storage_account_name}.blob.core.windows.net/grouped_sales_output"

df_grouped.write.mode("overwrite").parquet(output_path)

print("✅ Grouped data successfully written to destination container.")
```

(3) Spark Jobs

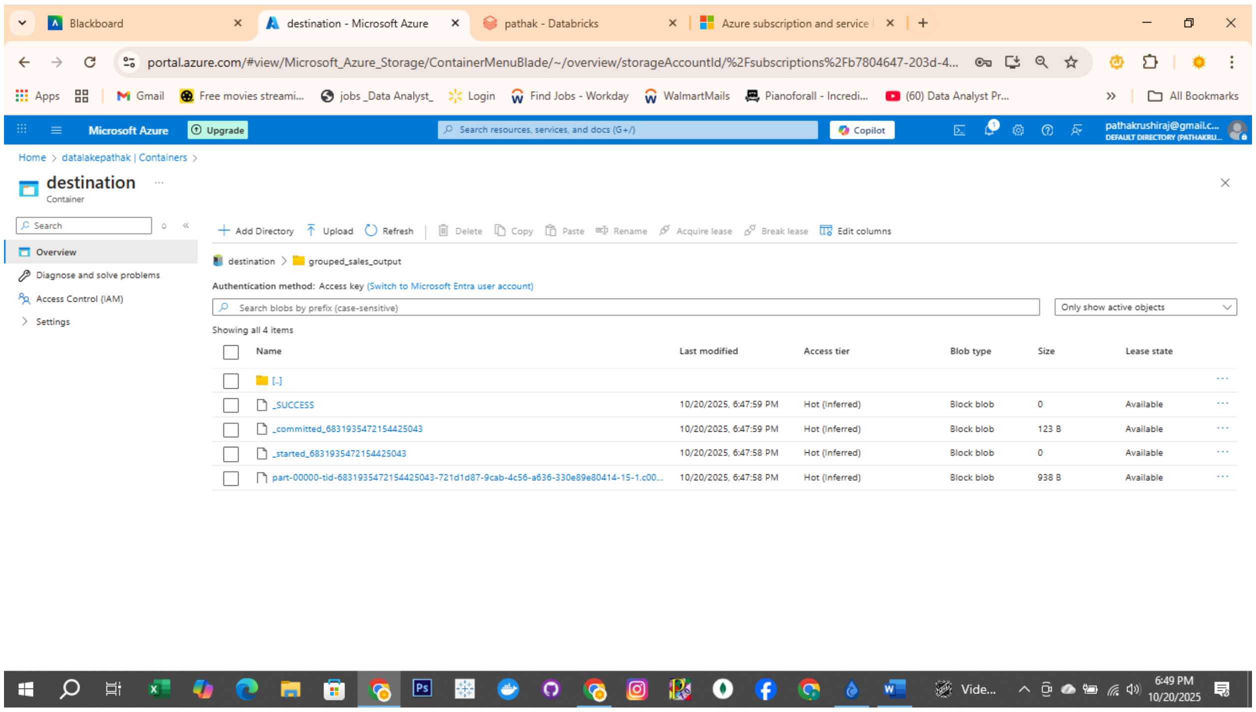
```
✅ Grouped data successfully written to destination container.
```

Figure 28Output Written Confirmation

Step 7 – Writing Data to Data Lake

- Saved the transformed data in Parquet and Delta formats.

df_transformed.write.format("delta").mode("overwrite").save("abfss://destination@datalakepathak.dfs.core.windows.net/grouped_sales_output")


Home > datalakepathak | Containers >

destination
Container

destination > grouped_sales_output

Authentication method: Access key (Switch to Microsoft Entra user account)

Showing all 4 items

Name	Last modified	Access tier	Blob type	Size	Lease state
[..]					
_SUCCESS	10/20/2025, 6:47:59 PM	Hot (Inferred)	Block blob	0	Available
_committed_6831935472154425043	10/20/2025, 6:47:59 PM	Hot (Inferred)	Block blob	123 B	Available
_started_6831935472154425043	10/20/2025, 6:47:58 PM	Hot (Inferred)	Block blob	0	Available
part-00000-tid-6831935472154425043-721d1d87-9cab-4c56-a636-330e89e80414-15-1.c00...	10/20/2025, 6:47:58 PM	Hot (Inferred)	Block blob	938 B	Available

Figure 29Destination Container Showing group_output Folder

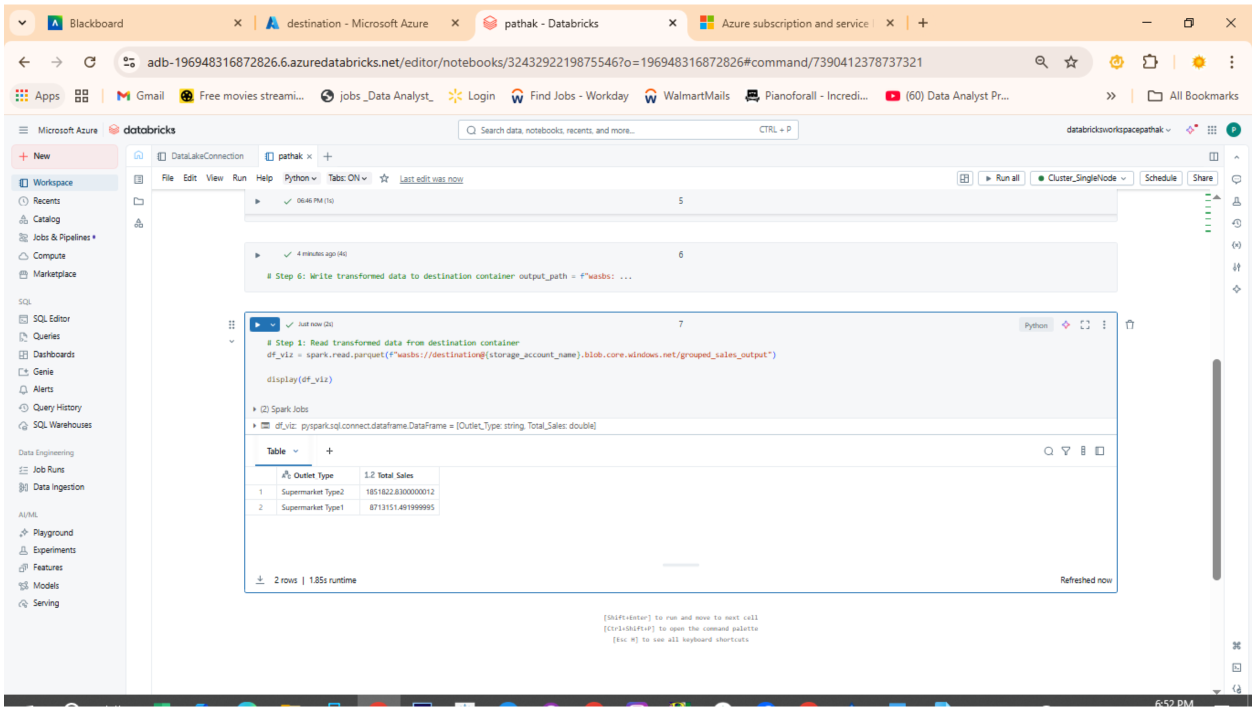
Step 8 – Creating Database and Delta Table

- Created a Hive Metastore database salesdb and registered a Delta table.

Spark.sql(“CREATE DATABASE IF NOT EXISTS salesdb”)

spark.sql(“USE salesdb”)

df_transformed.write.format(“delta”).mode(“overwrite”).saveAsTable(“salesdb.external_sales”)



```
# Step 6: Write transformed data to destination container output_path = f"wasbs: ...
```

```
# Step 1: Read transformed data from destination container
df_viz = spark.read.parquet(f"wasbs://destination@{storage_account_name}.blob.core.windows.net/grouped_sales_output")

display(df_viz)
```

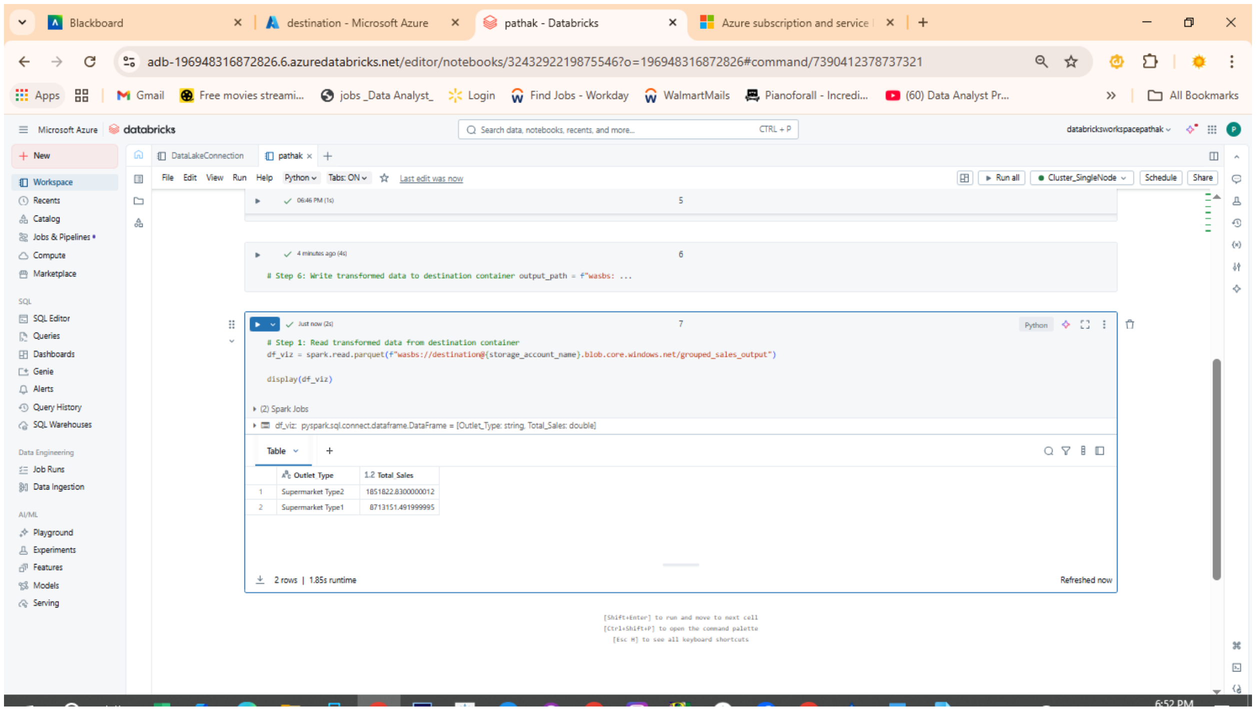
(2) Spark Jobs

df_viz: pyspark.sql.connect.dataframe.DataFrame = [Outlet_Type: string, Total_Sales: double]

	Outlet_Type	Total_Sales
1	Supermarket Type2	1851822.8300000012
2	Supermarket Type1	8713151.491999995

2 rows | 1.85s runtime

Figure 30Aggregated Data Displayed in Databricks



```
# Step 6: Write transformed data to destination container output_path = f"wasbs: ...
```

```
# Step 1: Read transformed data from destination container
df_viz = spark.read.parquet(f"wasbs://destination@{storage_account_name}.blob.core.windows.net/grouped_sales_output")

display(df_viz)
```

```
(2) Spark Jobs
df_viz: pyspark.sql.connect.dataframe.DataFrame = [Outlet_Type: string, Total_Sales: double]
```

	Outlet_Type	Total_Sales
1	Supermarket Type2	1851822.8300000012
2	Supermarket Type1	8713151.491999995

2 rows | 1.85s runtime

Figure 31Aggregated Data Displayed in Databricks

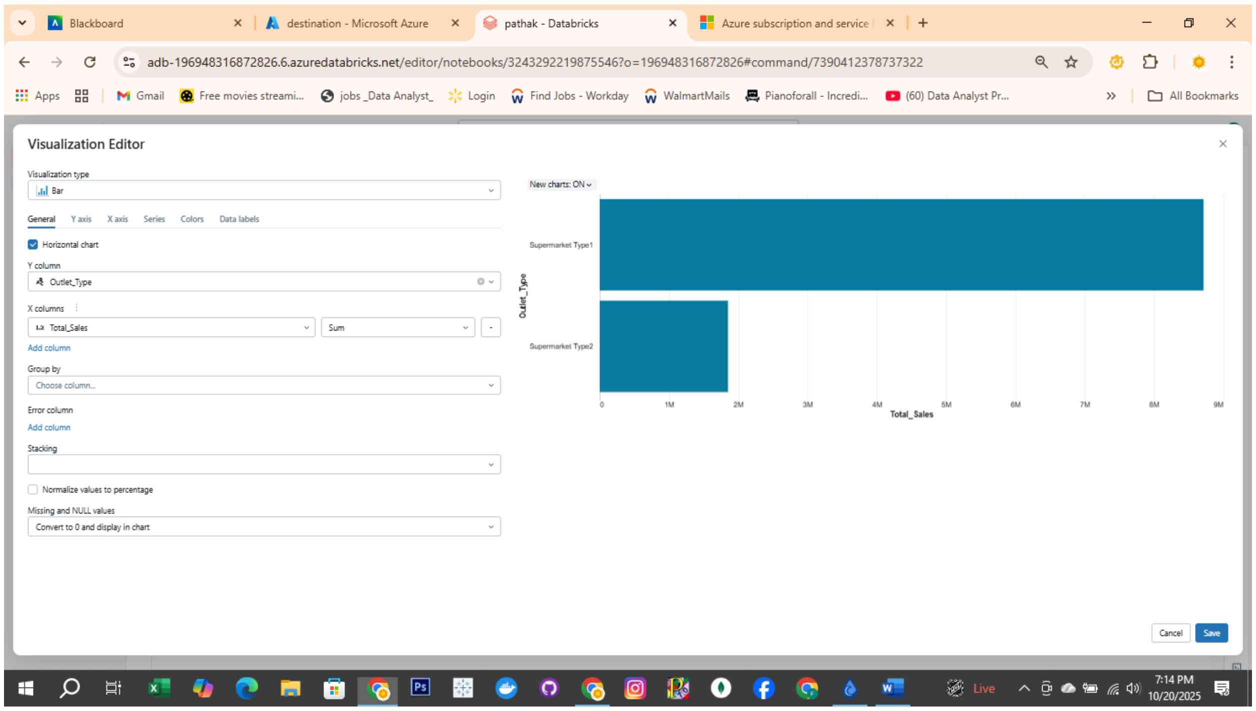
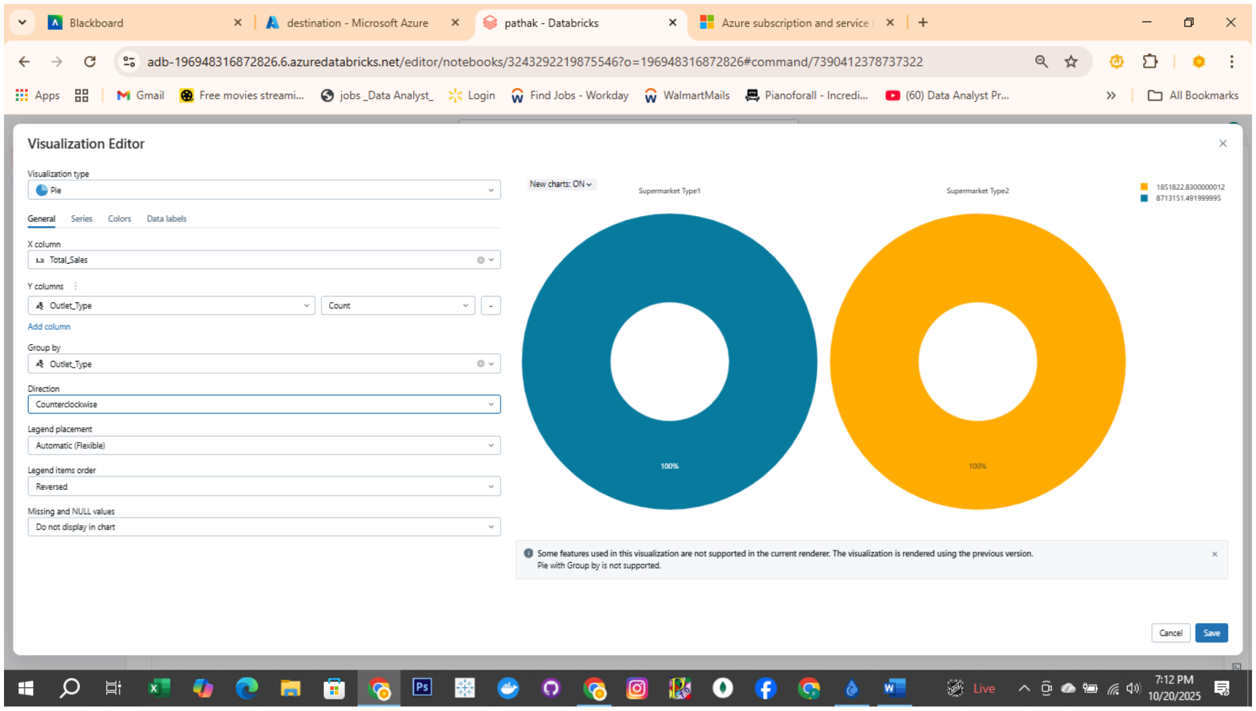


Figure 32Bar Chart Total Sales by Outlet Type



Visualization type: Pie

General | Series | Colors | Data labels

X column: Total_Sales

Y columns: Outlet_Type | Count

Add column

Group by: Outlet_Type

Direction: Counterclockwise

Legend placement: Automatic (Flexible)

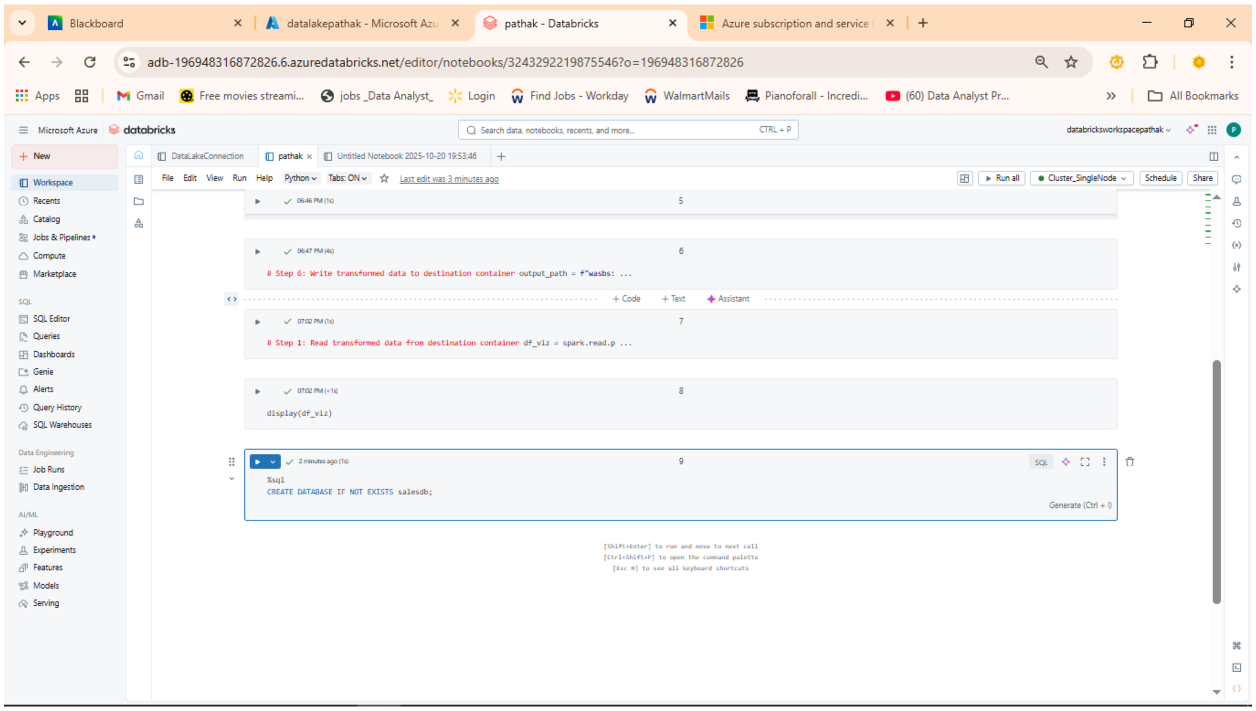
Legend items order: Reversed

Missing and NULL values: Do not display in chart

New charts: ON

Some features used in this visualization are not supported in the current renderer. The visualization is rendered using the previous version.
Pie with Group by is not supported.

Figure 33Pie Chart Sales Share by Outlet Type



```
# Step 6: Write transformed data to destination container output_path = f"wasbs: ...
```

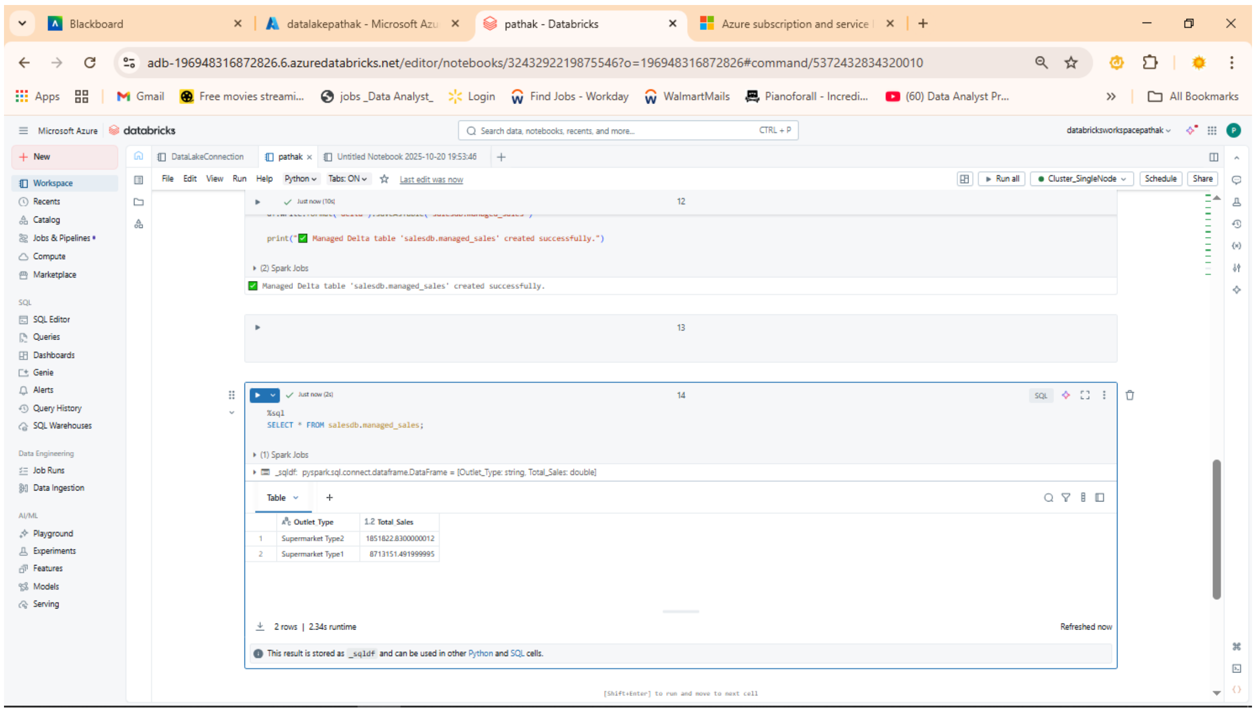
```
# Step 1: Read transformed data from destination container df_viz = spark.read.p ...
```

```
display(df_viz)
```

```
%sql
CREATE DATABASE IF NOT EXISTS salesdb;
```

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

Figure 34Create a Database



```
print("✅ Managed Delta table 'salesdb.managed_sales' created successfully.")
```

```
✅ Managed Delta table 'salesdb.managed_sales' created successfully.
```

```
%sql
SELECT * FROM salesdb.managed_sales;
```

_sqldf: pyspark.sql.connect.dataframe.DataFrame = [Outlet_Type: string, Total_Sales: double]

	Outlet_Type	Total_Sales
1	Supermarket Type2	1851822.8300000012
2	Supermarket Type1	8713151.491999995

2 rows | 2.34s runtime

This result is stored as `_sqldf` and can be used in other Python and SQL cells.

Figure 35Verify the Table

5. Results and Observations

- Successfully established **secure integration** between Azure Databricks and ADLS Gen2.
- Verified data ingestion, transformation, and output in Delta format.
- Registered the transformed data as a queryable **Hive Metastore table**.
- Generated meaningful insights such as **total sales by outlet type**.
- *(Optional)* Visualization confirmed higher sales in Supermarket Type2.

6. Conclusion

This project highlights the power of **Azure Databricks** for building scalable and automated data engineering pipelines. Using PySpark, large-scale data can be processed efficiently and stored in Delta format for reliable analytics. Integrating Databricks with Azure Data Lake provides a secure, cloud-native architecture for enterprise-grade data processing.

By leveraging Databricks' flexibility and Azure's managed services, this project successfully demonstrates an end-to-end workflow — from ingestion to transformation, database registration, and querying — establishing a strong foundation for real-world big data analytics projects.