Data Analysis using SparkSQL

Title of Project:

Implement Data analysis using Spark SQL on Azure Databricks & Process data for errors, seasonality, and anomalies.

Project Overview:

The goal of this project is to implement data analysis using Spark SQL on Azure Databricks and process the data for errors, seasonality, and anomalies. The project involves leveraging Azure Databricks for its Spark-based analytics capabilities and Spark SQL for querying and processing large datasets efficiently.

About the project:

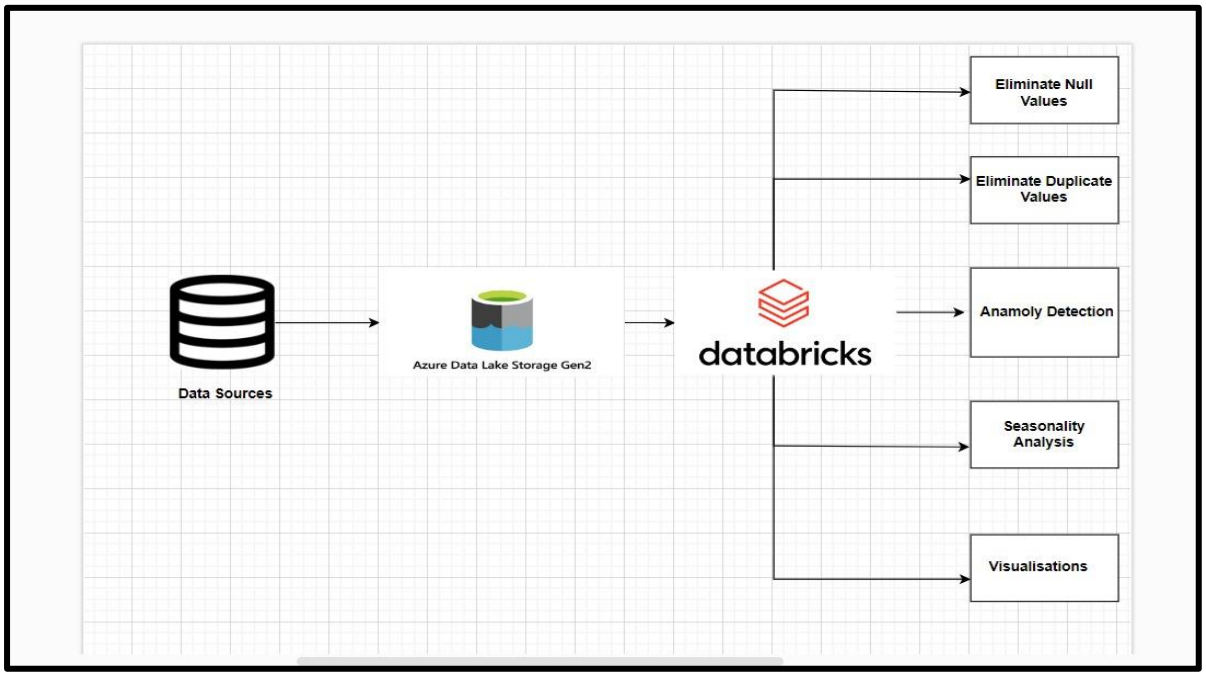
The project involves implementing data analysis using Spark SQL on Azure Databricks and processing the data for errors, seasonality, and anomalies. This initiative aims to leverage the scalability and performance of Spark on the Azure Databricks platform for efficient data processing and analysis.

Key Components of this project:

1) Data Ingestion
2) Data Exploration
3) Data Cleaning
4) Data Transformation
5) Data Analysis with Spark SQL
6) Error Detection
7) Seasonality Analysis
8) Anomaly Detection
9) Visualization and Reporting

Architecture Diagram:

The Architecture diagram shows the flow of execution of project and the operations that are performed on the dataset.



How it works

1) Setup Azure Databricks Environment:

- Create an Azure Databricks workspace in the Azure portal.
- Choose the appropriate subscription, resource group, and workspace name.
- Configure the pricing tier based on your requirements. Options include Standard and Premium tiers with different capabilities and performance levels.
- Once the deployment is complete, click on Go to Resource.
- Create a new notebook in the workspace.
- Create a compute by providing the required details and configure it with data bricks notebook.

2) Data Ingestion:

- Create an Azure Data Lake Storage Gen2 account in Azure Portal.
- Enable the Hierarchical Namespace in the Advanced Settings of ADLS account.
- Create a container in the ADLS Gen2 account.Upload the source file into container.
- Configure the Azure Data Lake Storage Gen2 account with Data Bricks to perform all the required operations.

3) Data Exploration and Cleaning:

- Explore the ingested data using Databricks notebooks, which provide an interactive environment for data exploration and analysis.
- Use Spark DataFrame APIs or Spark SQL within the notebooks to examine the data's schema, sample rows.
- Identify data quality issues such as missing values, duplicate records, inconsistent formats, and outliers.
- Apply data cleaning techniques such as removing null values, deduplication to ensure data quality and consistency.

4) Data Transformation:

- Transform the cleaned data into a format suitable for analysis using Spark SQL.
- Perform transformations such as filtering, aggregating to prepare the data for analysis.
- Leverage Spark's distributed processing capabilities to handle large-scale data transformations efficiently.

5) Data Analysis with Spark SQL:

- Write Spark SQL queries within Databricks notebooks to perform advanced analytics on the transformed data.
- Utilize SQL syntax to extract insights, perform aggregations, and apply statistical functions.
- Leverage Spark SQL's built-in functions and windowing functions for complex analytics tasks such as time series analysis, Seasonality detection, and anomaly detection.

6) Error Detection:

- Implement algorithms or techniques within Spark SQL to detect errors or anomalies in the data.
- Utilize SQL queries and functions to identify inconsistencies or patterns deviating from expected behaviour.

7) Seasonality Analysis:

- Analyse the data for seasonality patterns using Spark SQL.
- Apply time series analysis techniques such as moving averages, or Fourier transforms to identify recurring trends or fluctuations over time.
- Use SQL queries and functions to aggregate data over time intervals and calculate seasonal metrics such as seasonal averages.

  Steps:

- Apply moving average to smooth out seasonal variations.
- Analyse Moving Averages
- Visualize Moving Averages in Azure Data Bricks.

8) Anomaly Detection:

- Develop models or algorithms within Spark SQL to detect anomalies in the data.
- Utilize machine learning algorithms, statistical methods, or rule-based approaches for anomaly detection.

Steps:

- Calculate Z-scores for each value.
- Define anomaly threshold (e.g., Z-score greater than 1•Analyze the identified anomalies
- Visualization of Anomalies (e.g., scatter plot of ORDERDATE vs. QUANTITYORDERED)

9) Visualization and Reporting:

- Visualize the analyzed data and insights using Databricks notebooks or external visualization tools.
- Use visualization in Databricks' built-in visualization tools to create compelling visualizations that highlight key insights and trends.

Azure Resources / Tools used for this Project

1) Azure Data Lake Storage Gen2 (ADLS Gen2):

- ADLS Gen2 is the storage solution where the Olympics data will be ingested.
- ADLS Gen2 provides a scalable and cost-effective storage solution for big data analytics workloads.

2) Azure Databricks:

- Azure Databricks is a powerful analytics platform that enables organizations to use the capabilities of Apache Spark for processing big data in a collaborative and scalable environment.
- PySpark notebooks are be used in Azure Databricks to perform data transformation tasks like filtering ,group by and aggregations, selecting distinct data, renaming columns etc., on the ingested Olympics data.

3) Spark SQL:

- Spark SQL is a component of Apache Spark that introduces a structured data processing model along with a SQL-like query language.
- It allows users to run SQL queries programmatically and seamlessly integrate SQL queries with existing Spark programs.

4) Azure Portal:

- The Azure Portal is a web-based interface for managing Azure services. It would be used for provisioning and configuring resources such as Azure Data Lake Storage Gen2, Azure Data Factory, and Azure Databricks.

Project Requirements

1) **Data Ingestion:**
   - Ingest raw data from various sources such as databases, files, or streaming sources into Azure Databricks storage.
2) **Data Exploration:**
   - Explore the ingested data to understand its structure, schema, and quality.
3) **Data Cleaning:**
   - Clean the data to handle inconsistencies, missing values, and outliers.
   - Ensure data consistency and reliability for further analysis.

4) Data Transformation:

- Transform the cleaned data into a suitable format for analysis.
- Reshape, aggregate, or derive new features as needed for analysis.

5) Data Analysis with Spark SQL:

- Utilize Spark SQL to perform advanced analytics on the transformed data.
- Write SQL queries to extract insights, perform aggregations, and apply statistical functions.

6) Error Detection:

- Implement algorithms or techniques to detect errors in the data.
- Identify inconsistencies, anomalies, or patterns deviating from expected behavior.

7) Seasonality Analysis:

- Analyse the data for seasonality patterns to understand recurring trends or fluctuations over time.

8) Anomaly Detection:

- Develop models or algorithms to detect anomalies in the data.
- Use outlier detection, anomaly scoring (using z-score) , or machine learning-based techniques for anomaly detection.

9) Visualization and Reporting:

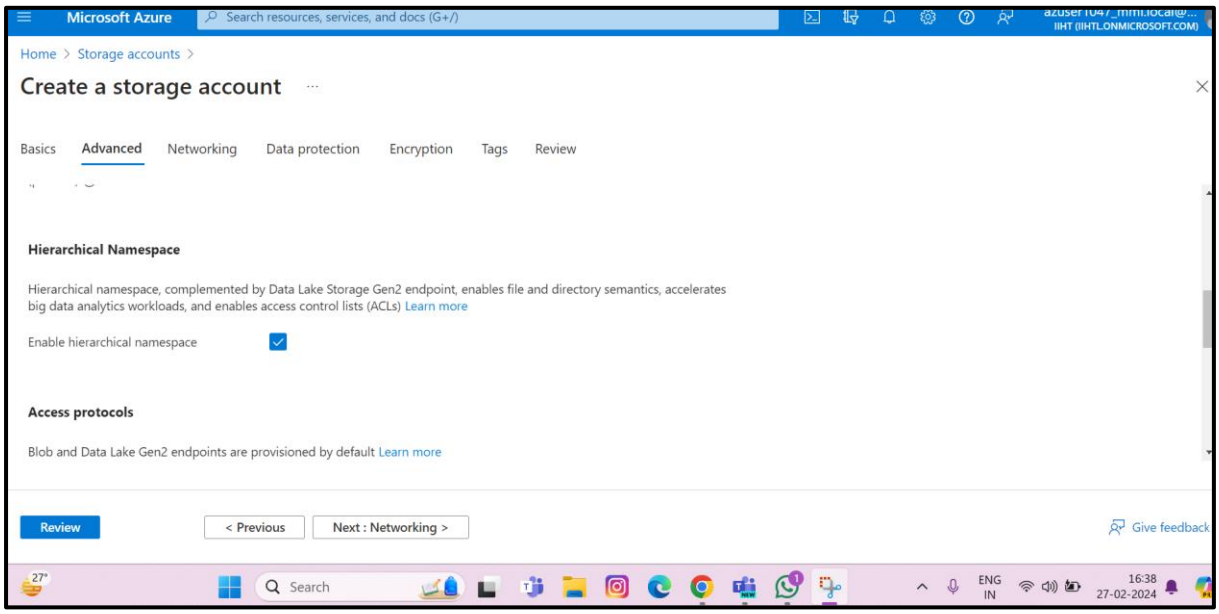
- Visualize analysed data and insights using interactive visualizations or reports.
- Communicate findings to stakeholders for data-driven decision-making.

Tasks Performed

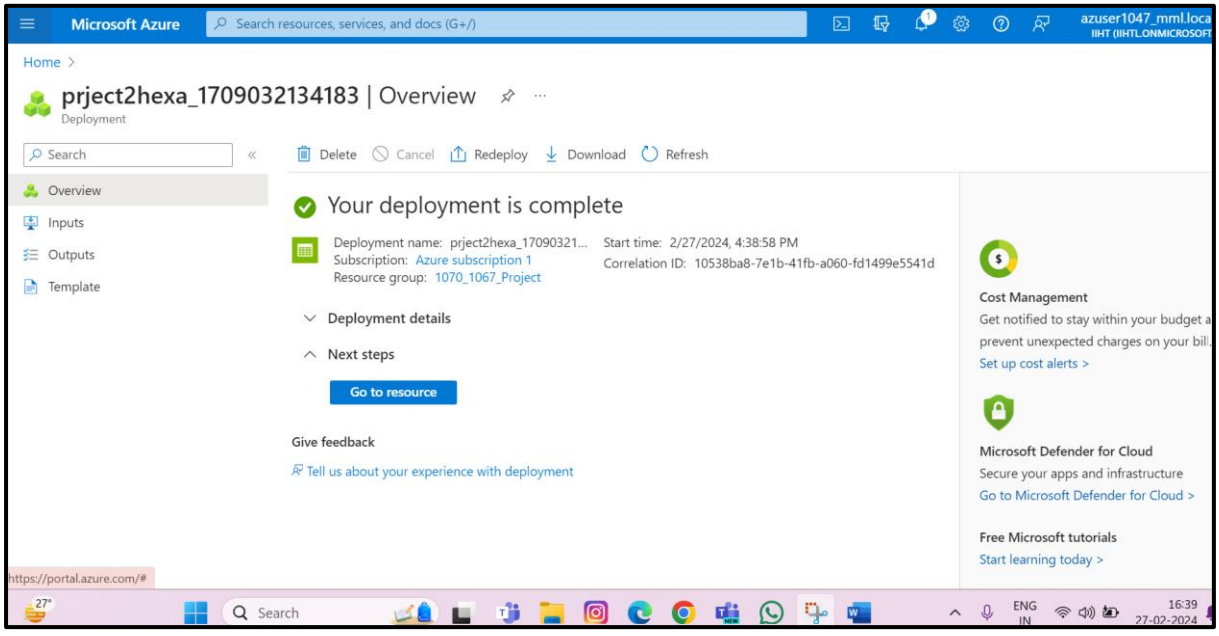
1) Provision Azure Databricks workspace in Azure portal.
2) Ingest raw data from sources into Azure Databricks.
3) Explore data using Databricks notebooks for structure and quality.
4) Clean data by handling missing values, duplicates, and inconsistencies.
5) Transform cleaned data into suitable format for analysis.
6) Apply transformations such as reshaping and aggregating using Spark SQL.
7) Write Spark SQL queries in Databricks notebooks for advanced analytics.
8) Analyze data for errors, seasonality patterns, and anomalies using SQL queries.
9) Implement algorithms in Spark SQL for error detection and anomaly detection.
10) Analyze data for seasonality patterns using SQL queries and functions.

Execution of Project

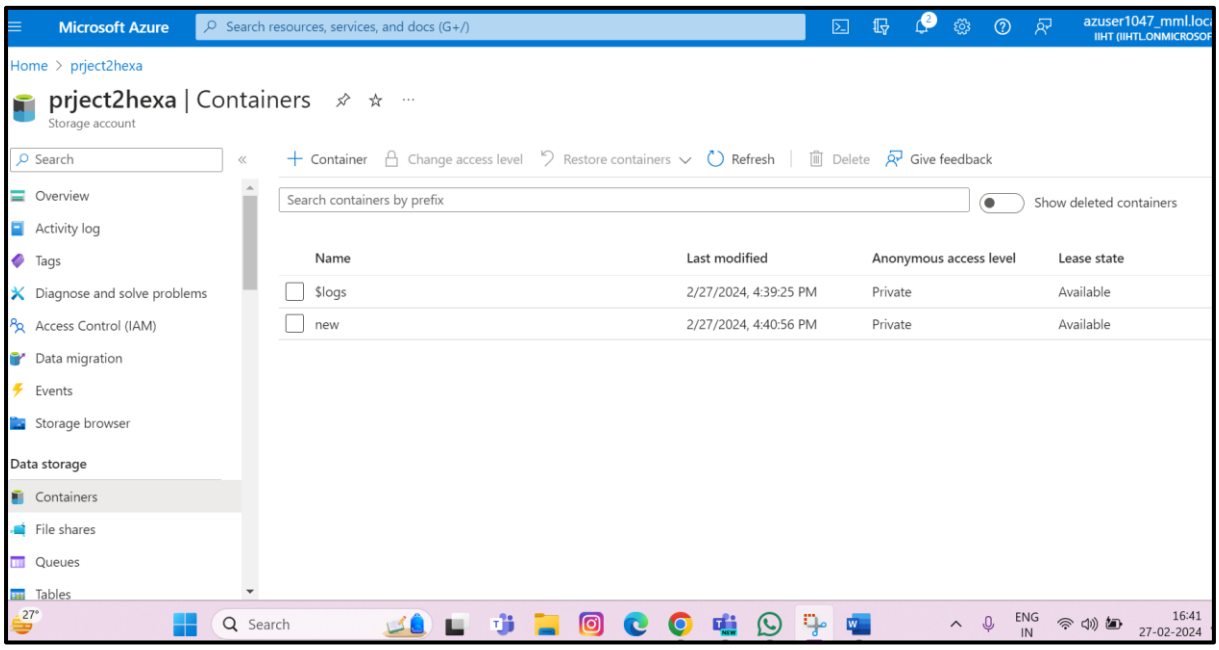
Creating an Azure Data Lake Storage GEN2 Account by enabling Hierarchical Namespace



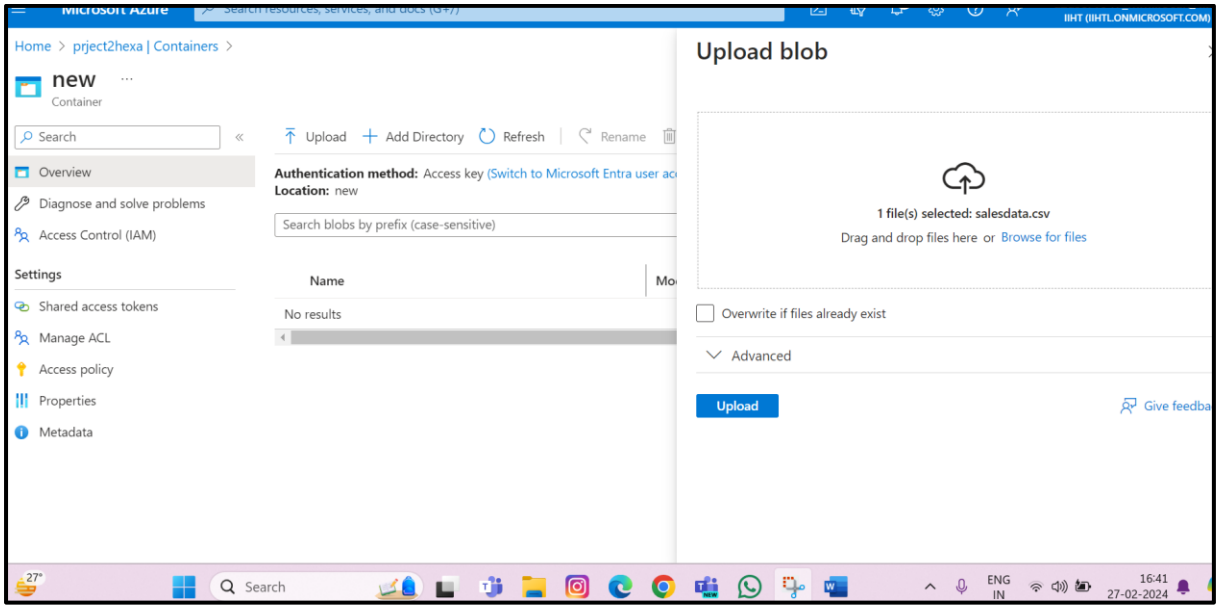
Account Created successfully:



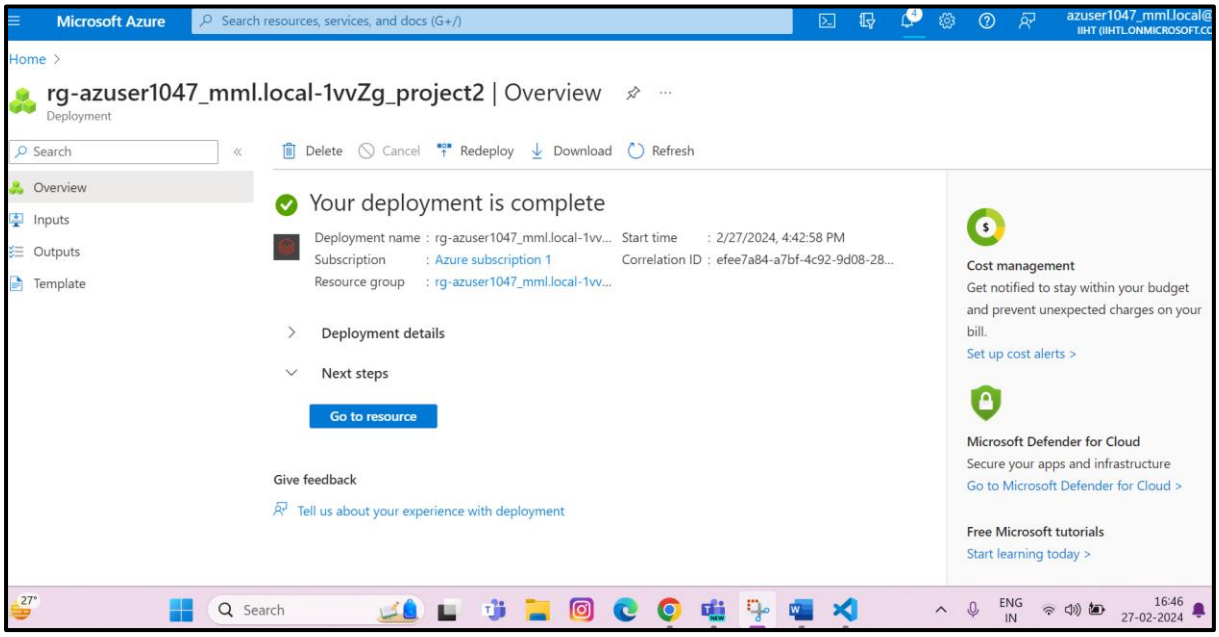
Inside the ADLS account create a container to upload file:



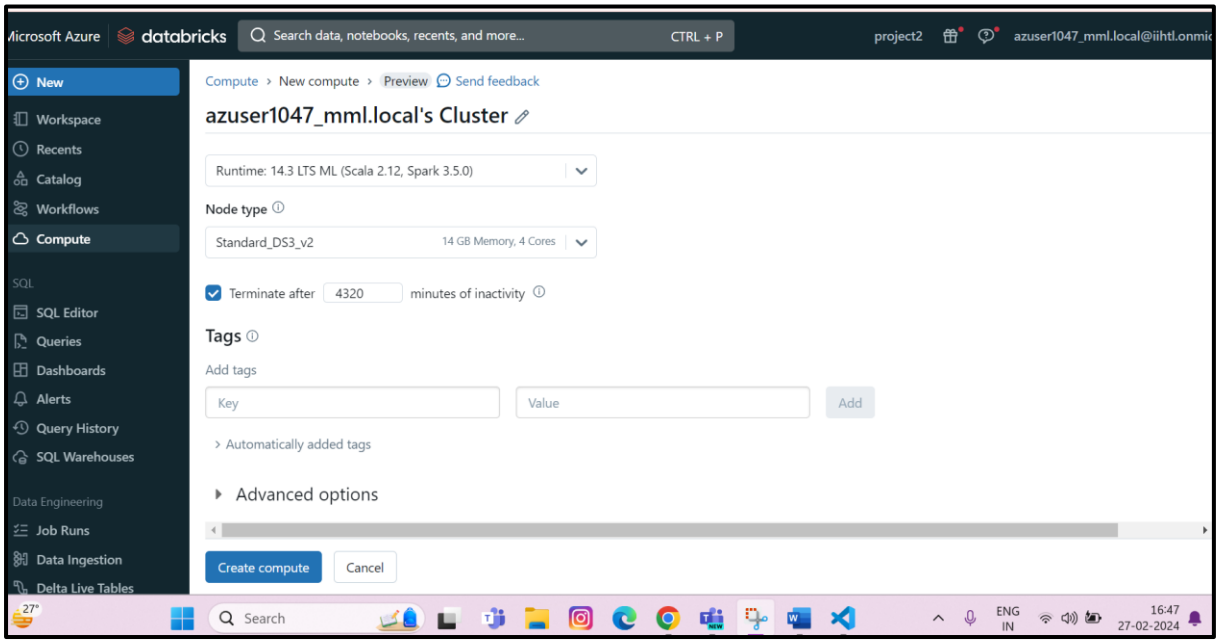
Uploading file in Container:



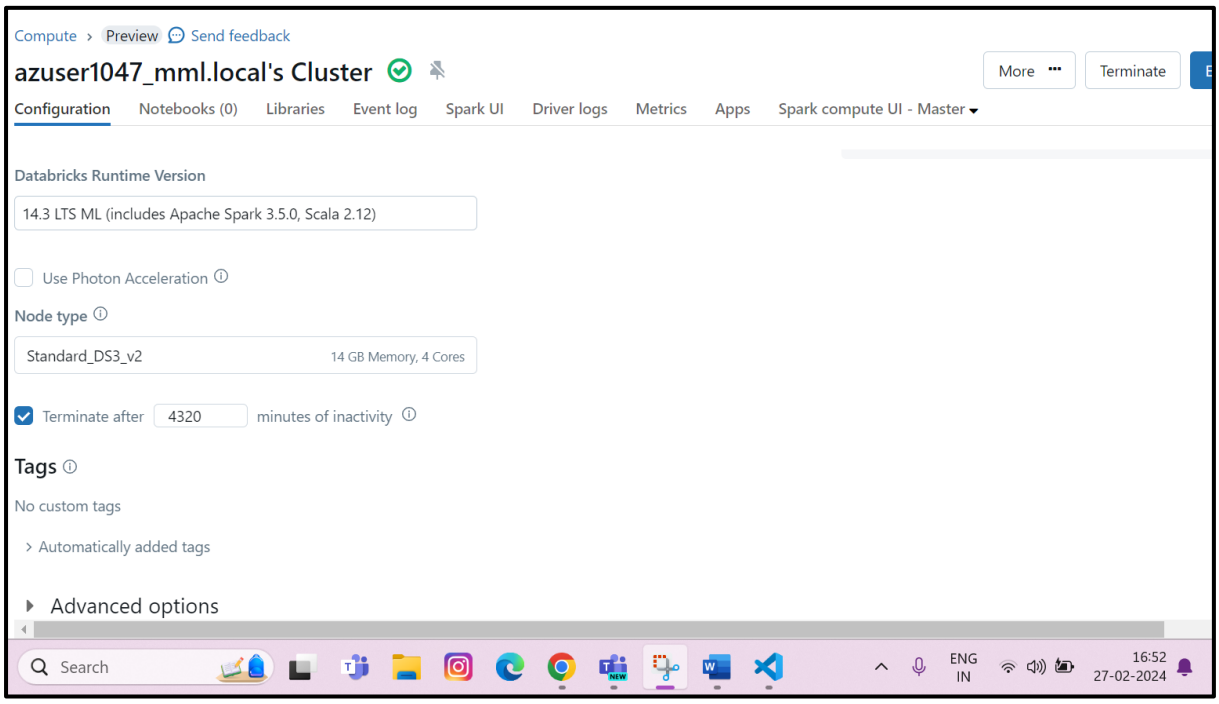
Creating Azure Databricks Workspace



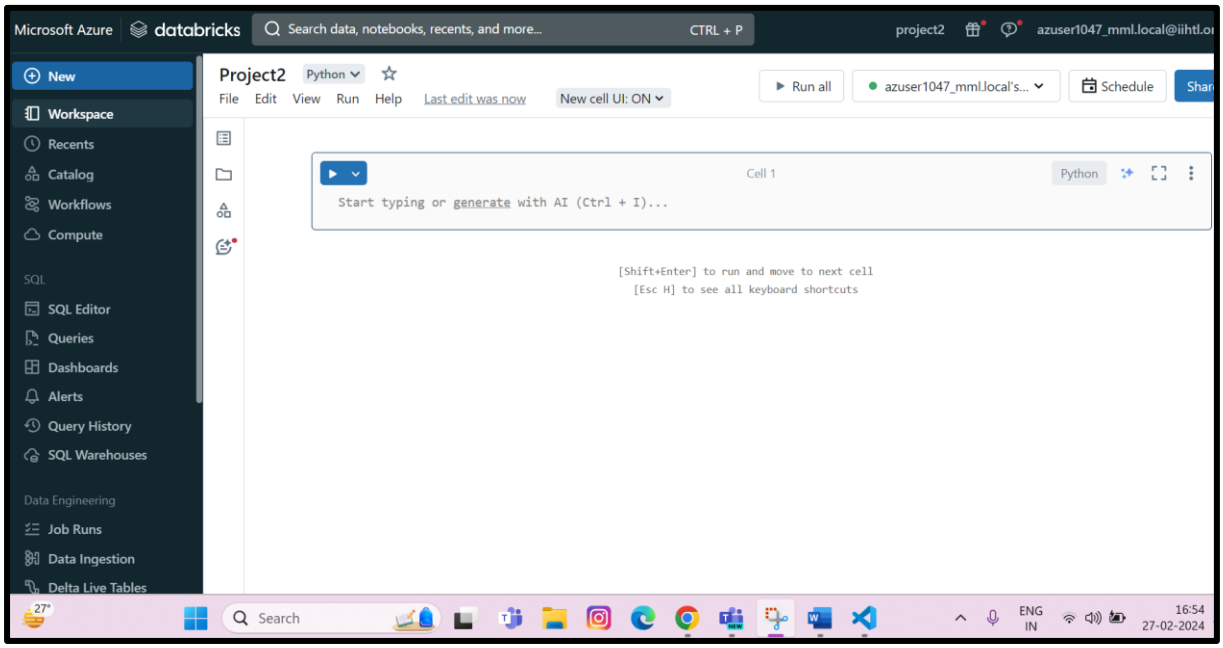
Creating a cluster after launching Databricks notebook:



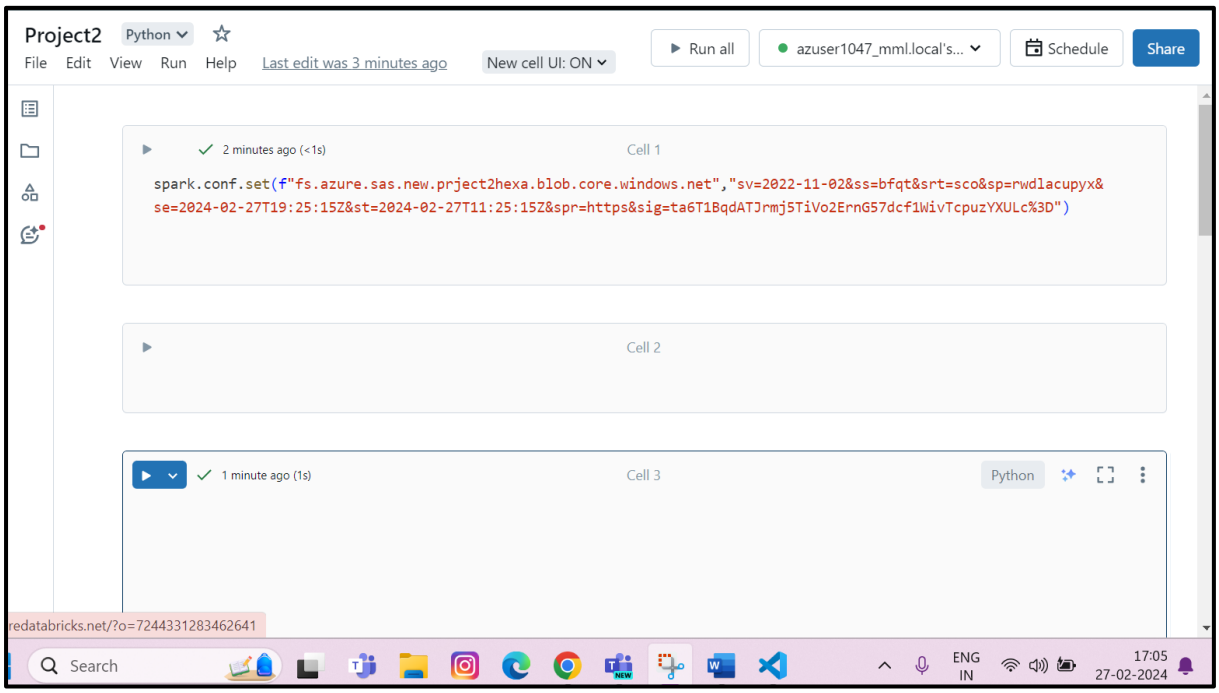
Cluster created successfully:



Creating a Databricks workspace:

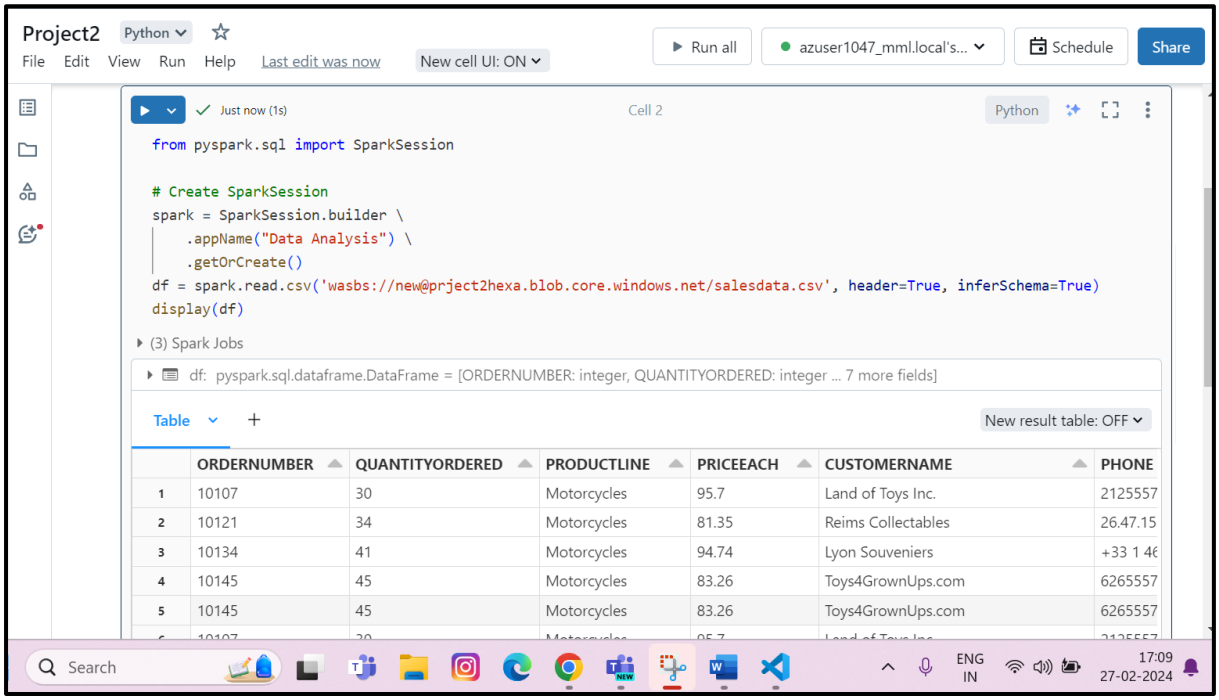


Connecting Databricks to source file through SAS token:

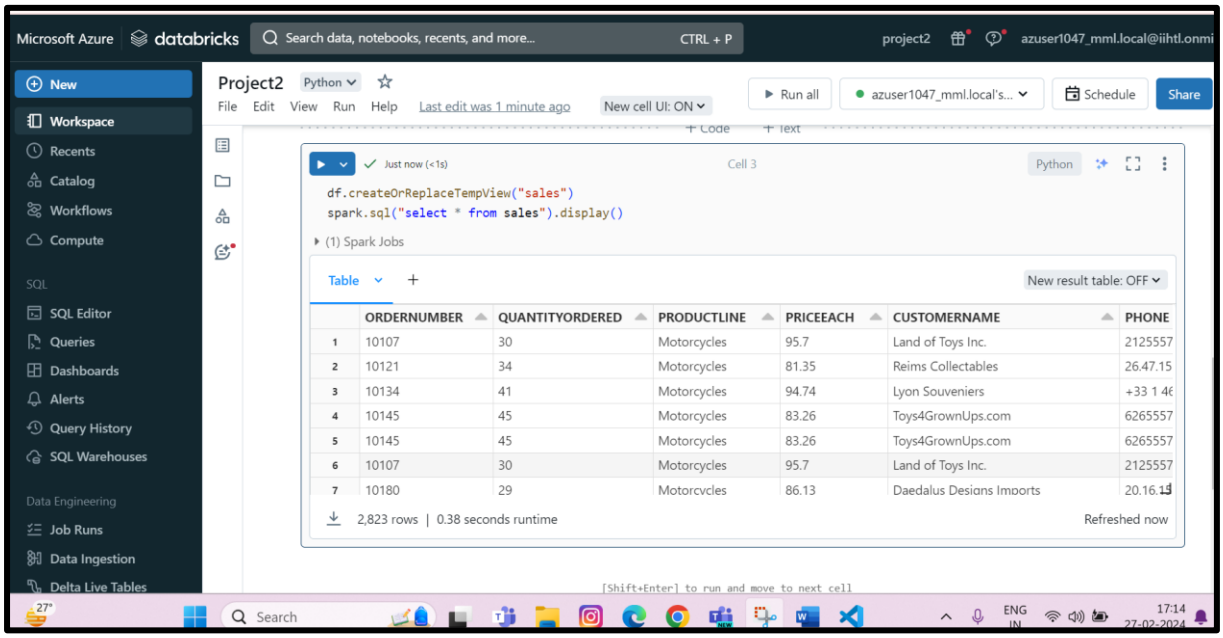


Data Exploration

Displaying Schema:



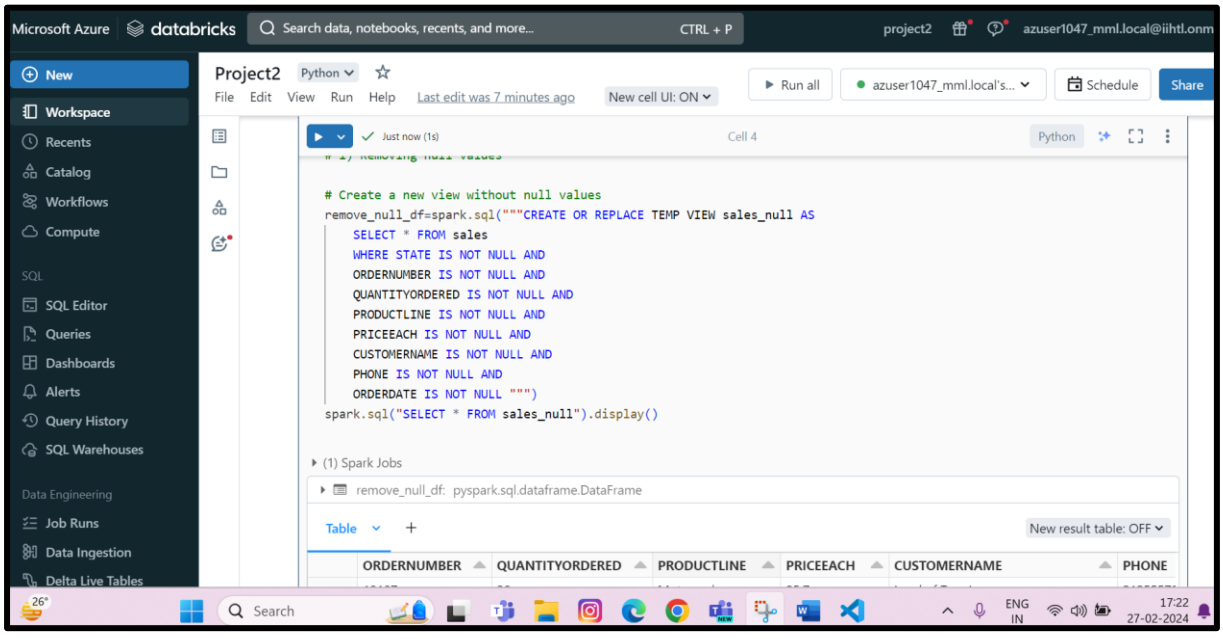
Creating a temp view and selecting all the rows:



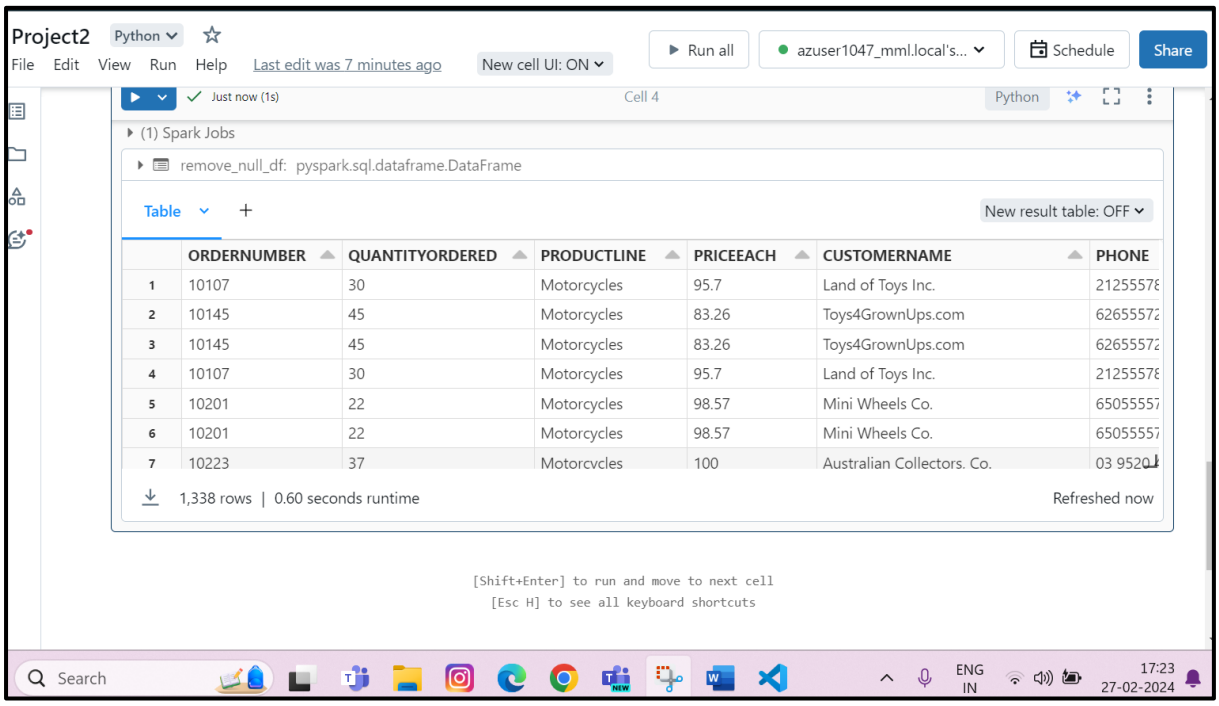
Data Cleaning

Removing null values:

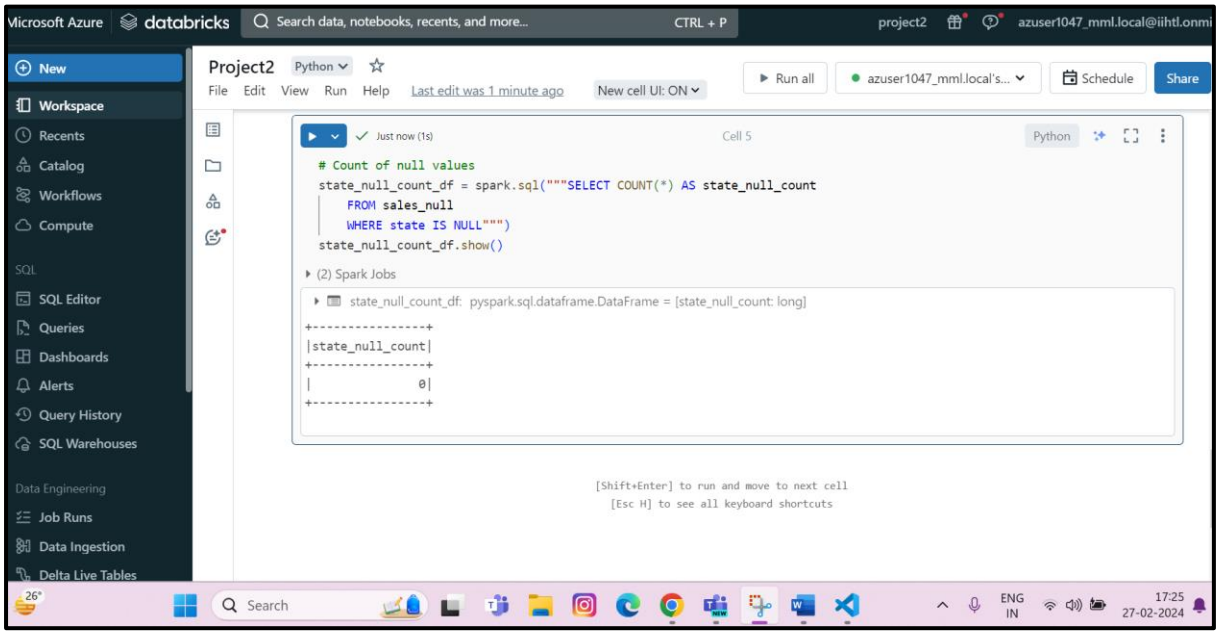
Code:



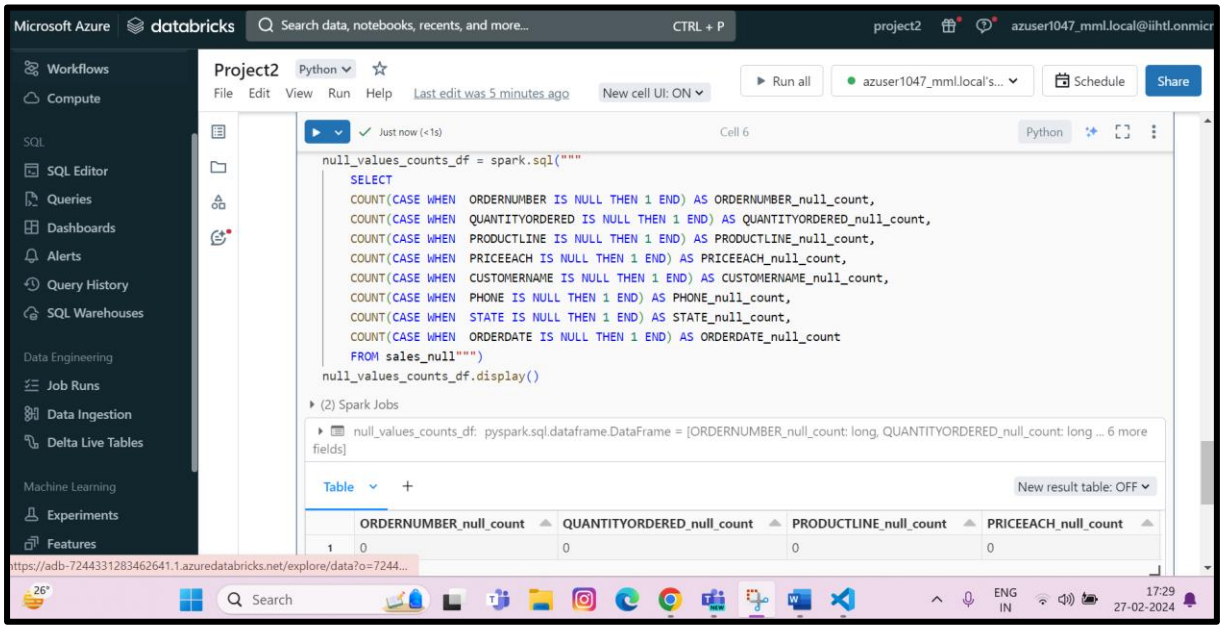
Output:



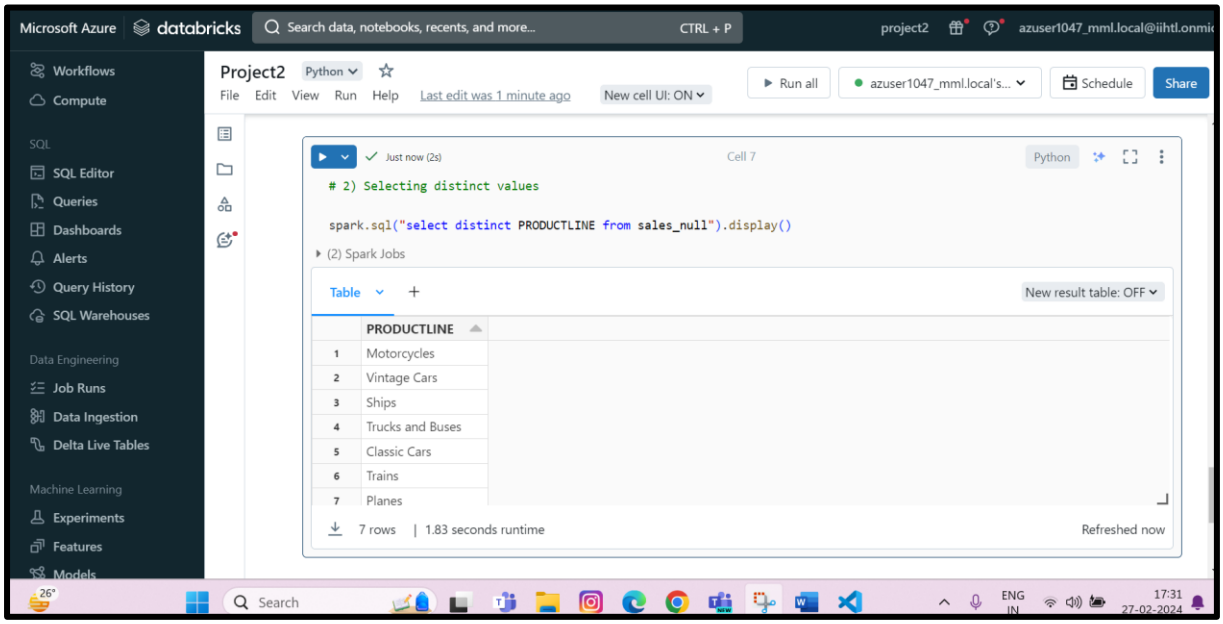
Count of null Values:



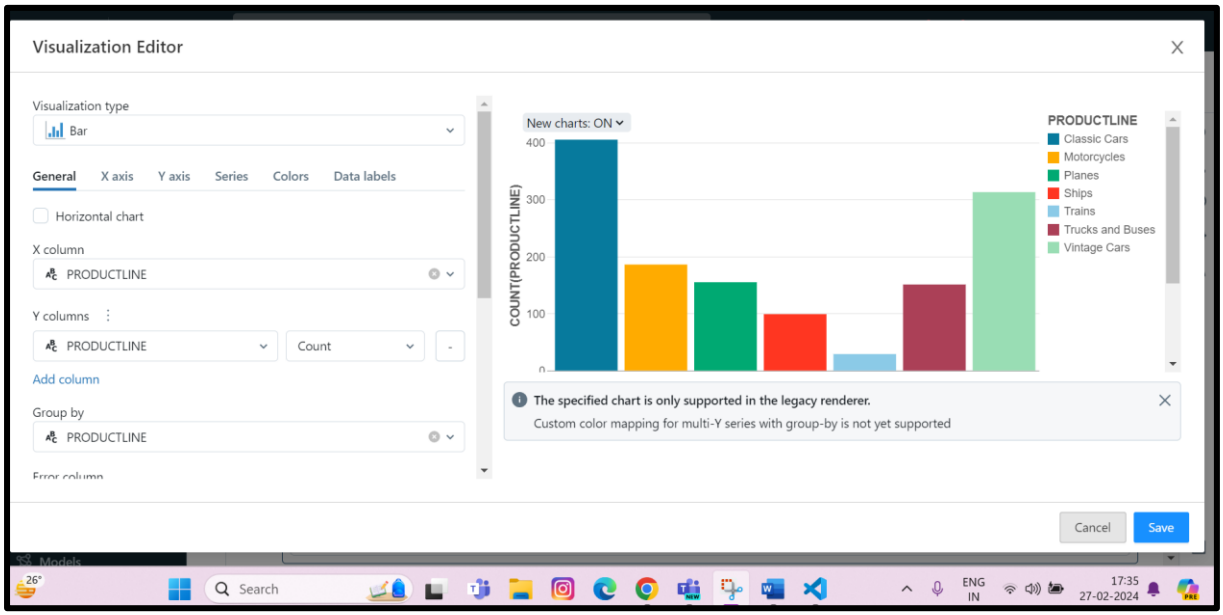
Count of null values in all columns



Distinct Values :

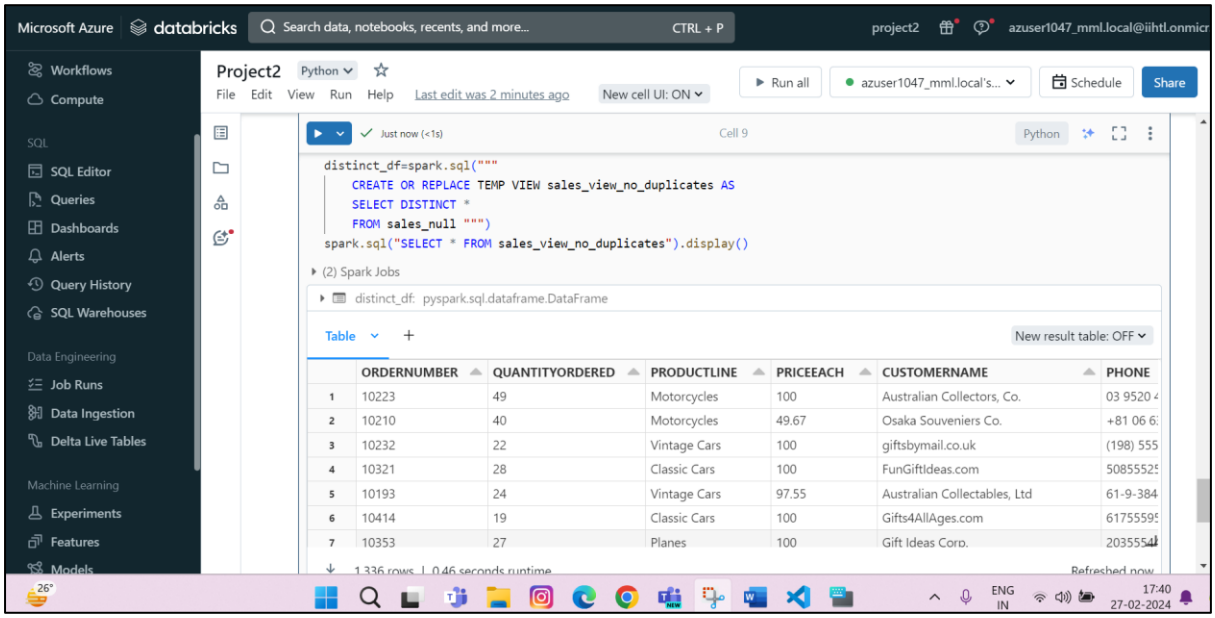


Visualisation that shows count of each product grouped by the product names:

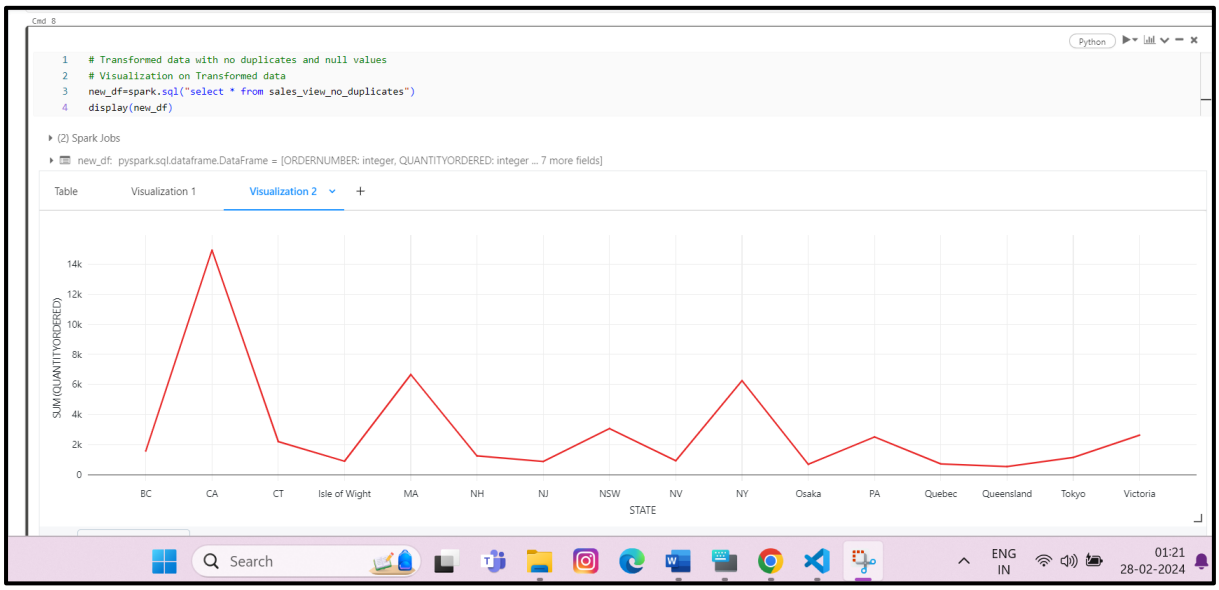


Error Detection

Removing duplicate values:

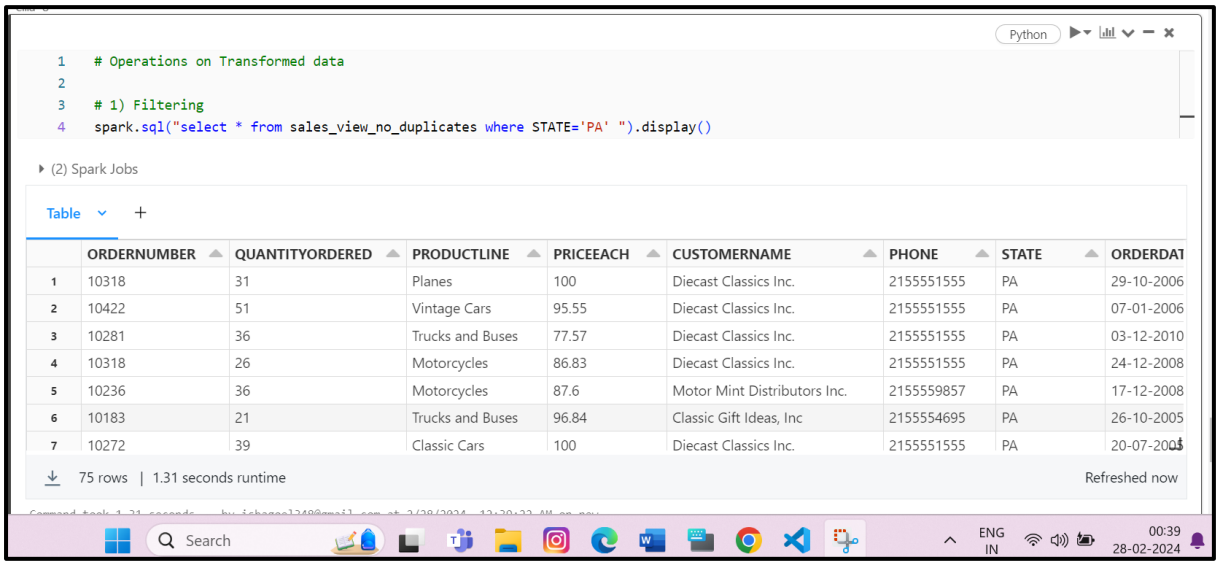


Visualise Transformed Data with no null and no duplicate values:



Operations On Transformed Data

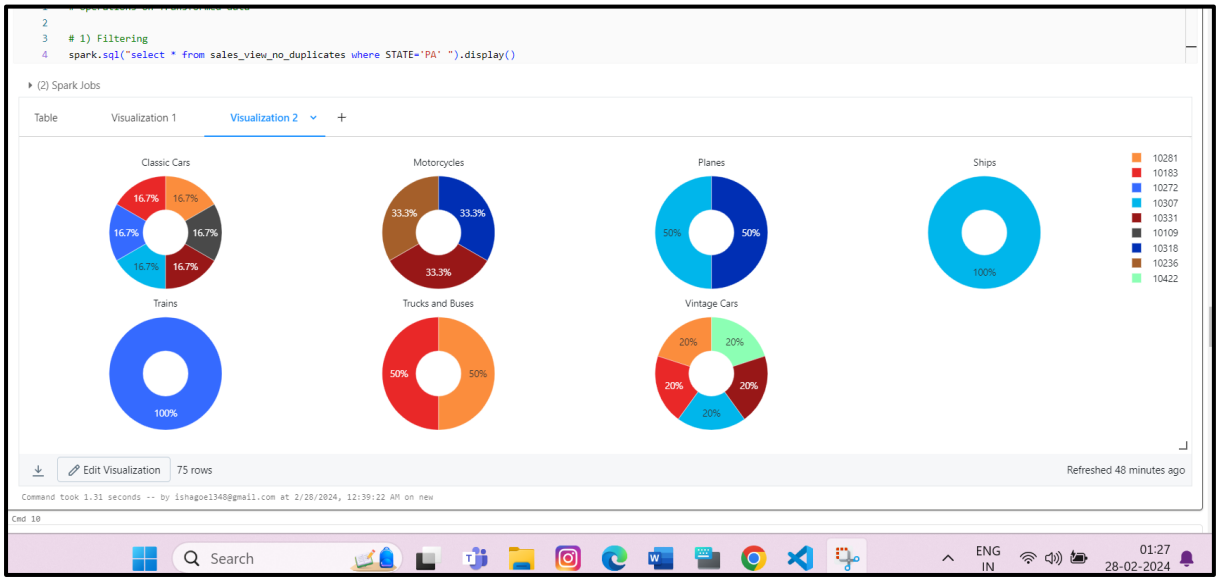
Filtering:



Visualization for filter:

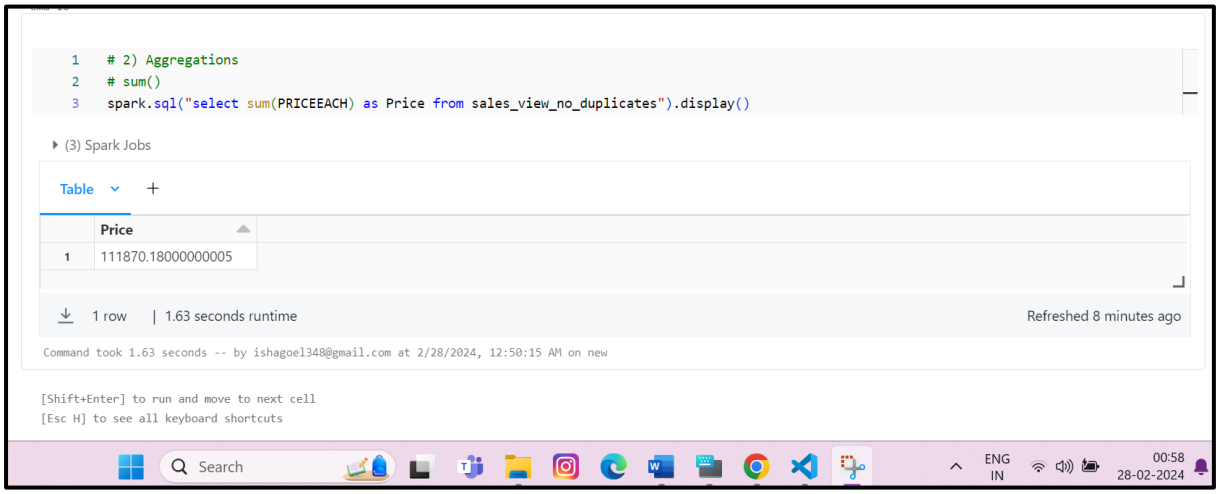
Scatter:

Pie Chart:

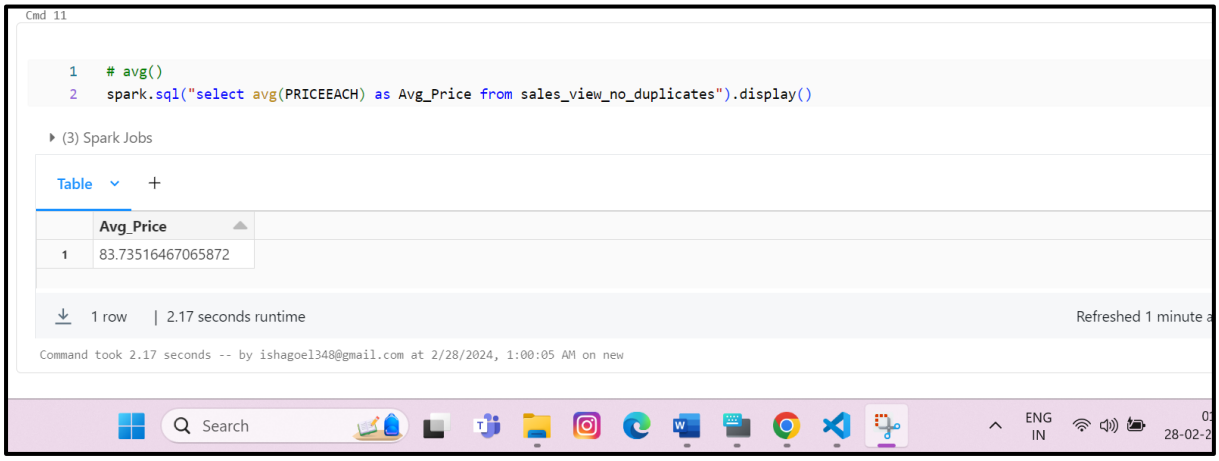


Aggregations

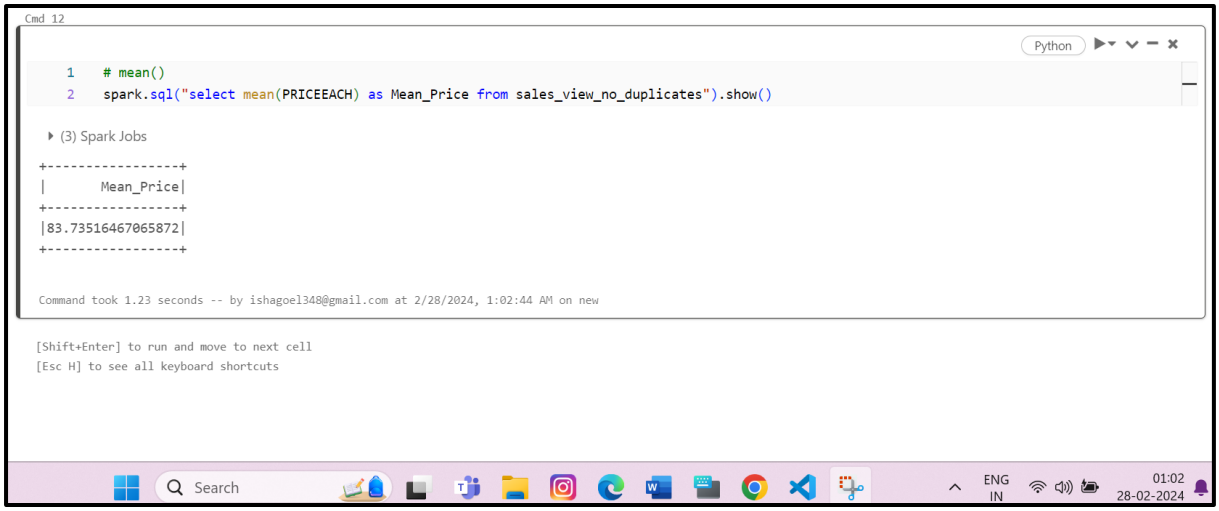
Sum():



Avg():



Mean():



Anomaly Detection

Calculating z-scores for each value:

The Z-score is a statistical measure that indicates how many standard deviations a data point is from the mean.

$Z = (X-\mu)/\sigma$ where

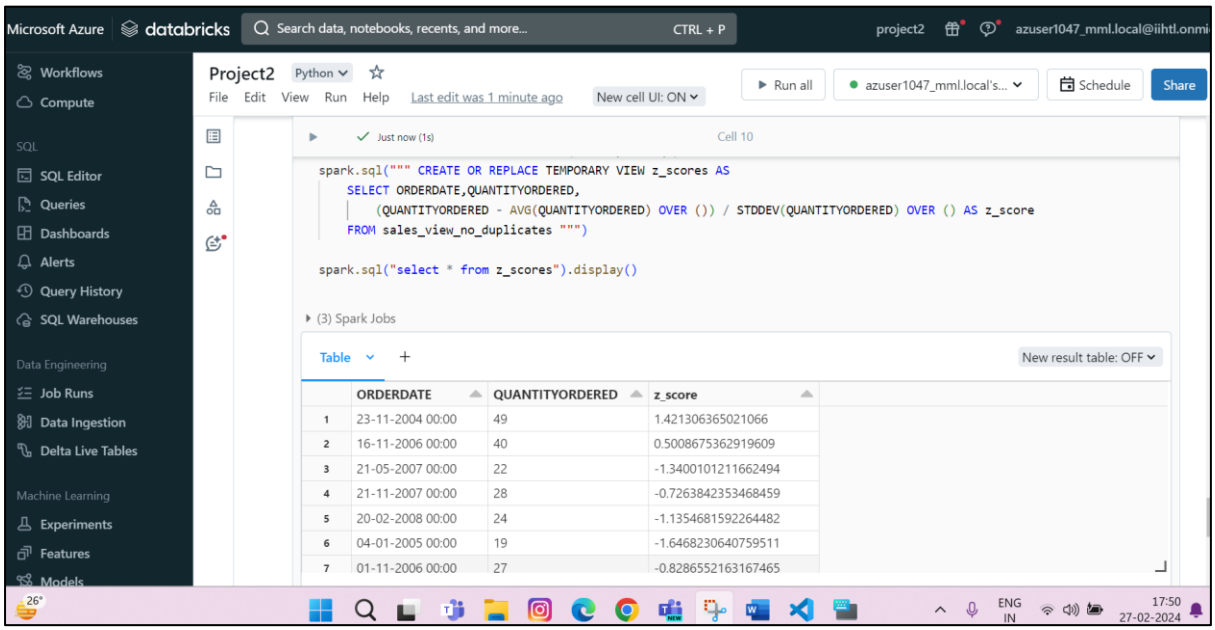
X is the data point,

μ is the mean, and

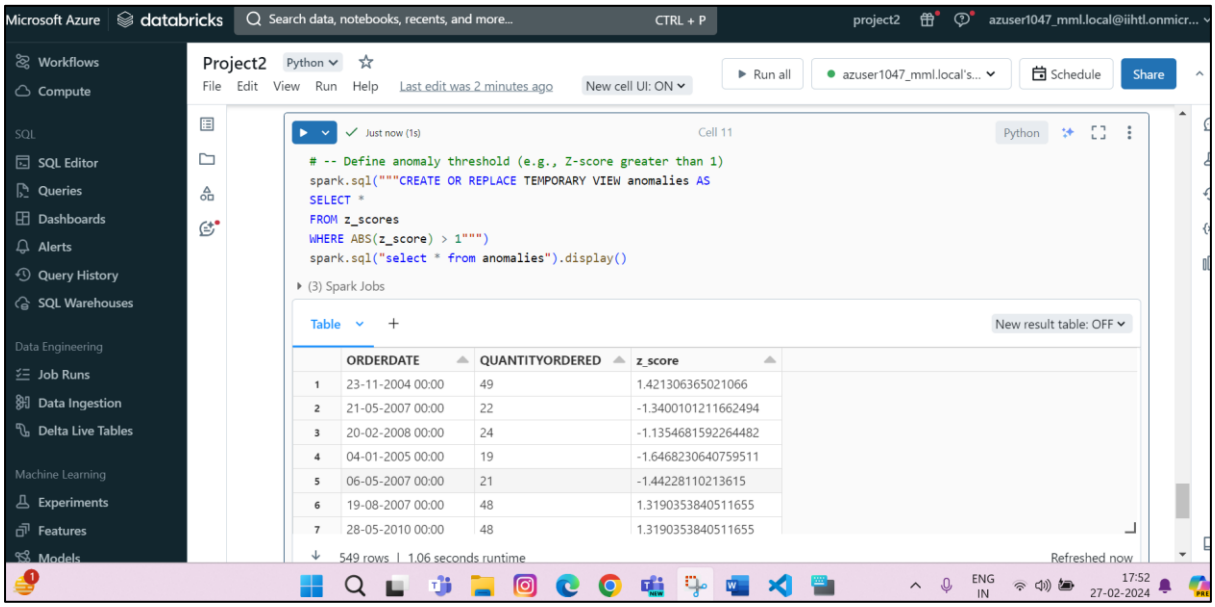
σ is the standard deviation.

AVG(value) OVER () calculates the mean value of the entire dataset.

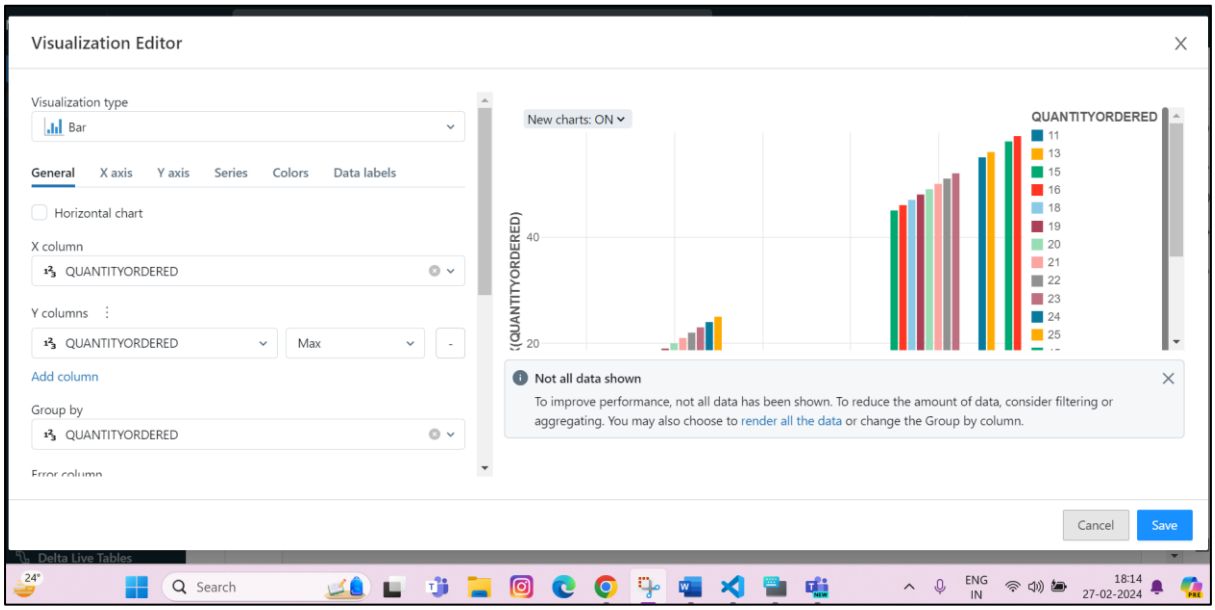
STDDEV(value) OVER () calculates the standard deviation of the entire dataset.



Display of z-scores



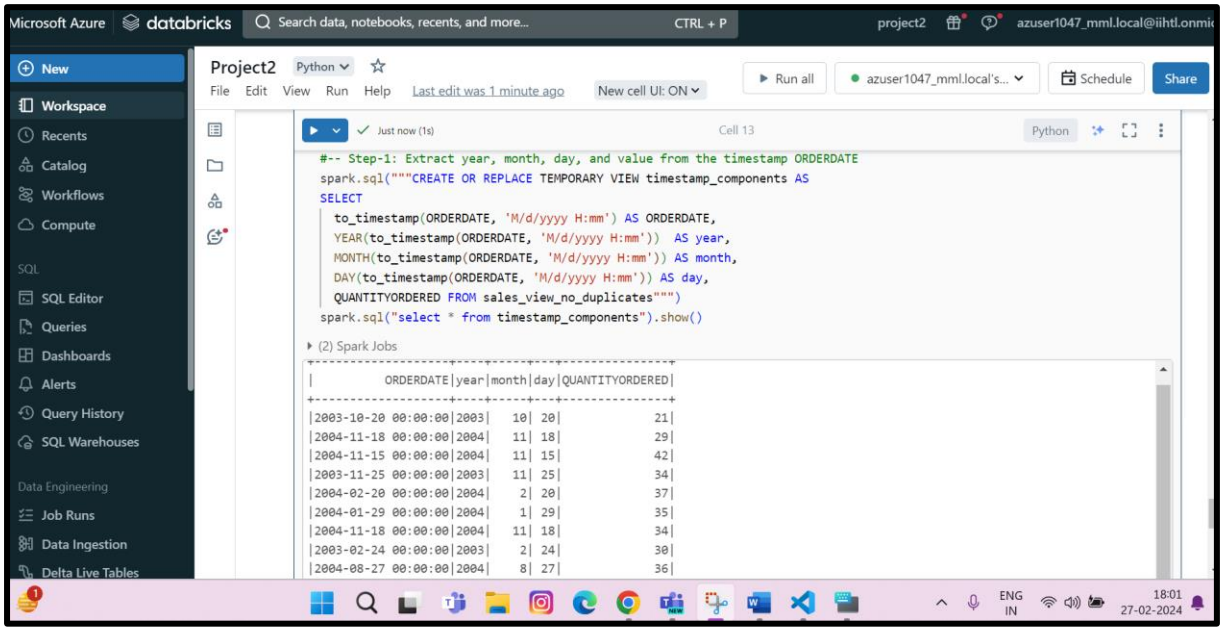
Visualise anomaly:



Seasonality Detection

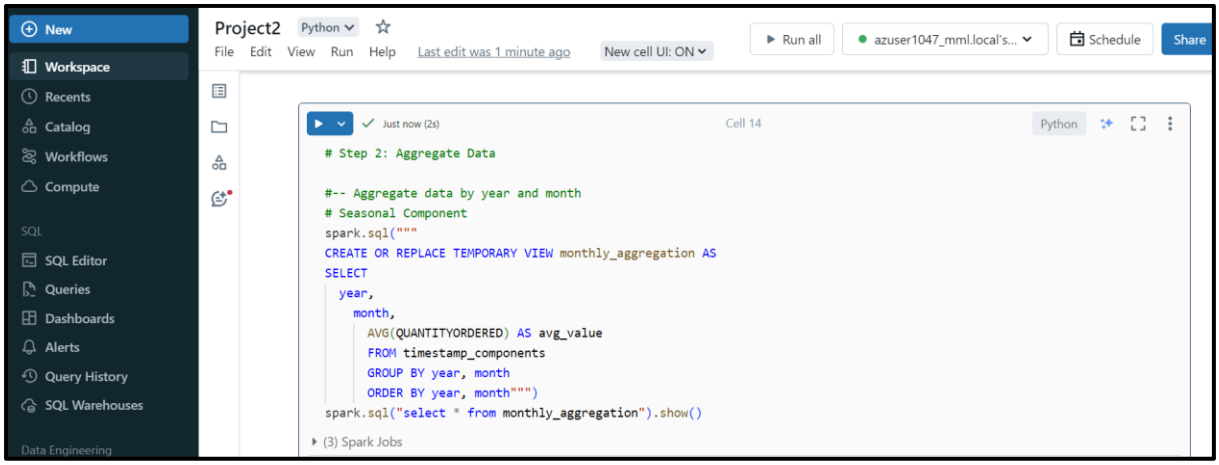
Seasonality refers to the repetitive and predictable patterns or fluctuations in a time series dataset that occur at regular intervals over time. These patterns typically repeat over specific periods, such as days, weeks, months, or years, and are often influenced by external factors such as weather, holidays, or economic cycles.

Extract Timestamp Components: Extract year, month, day, and value from the timestamp ORDERDATE.

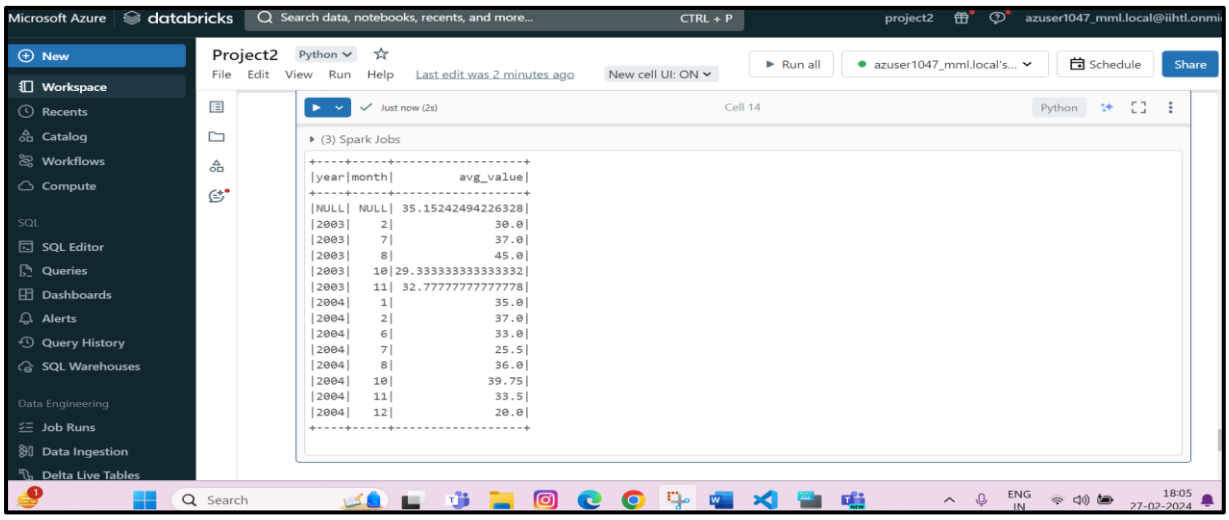


Aggregate Data: Aggregate data by year and month

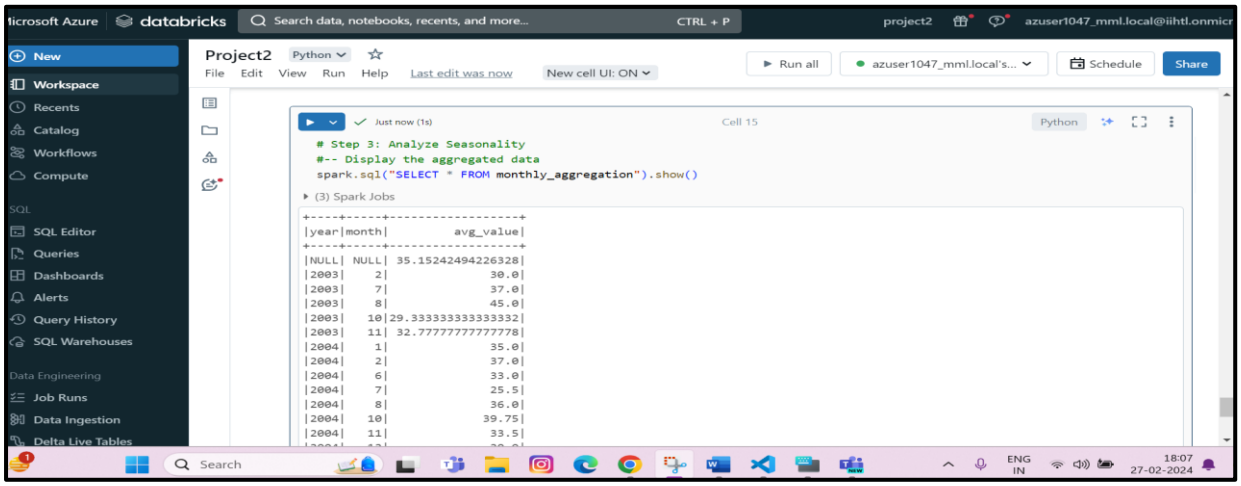
Code:



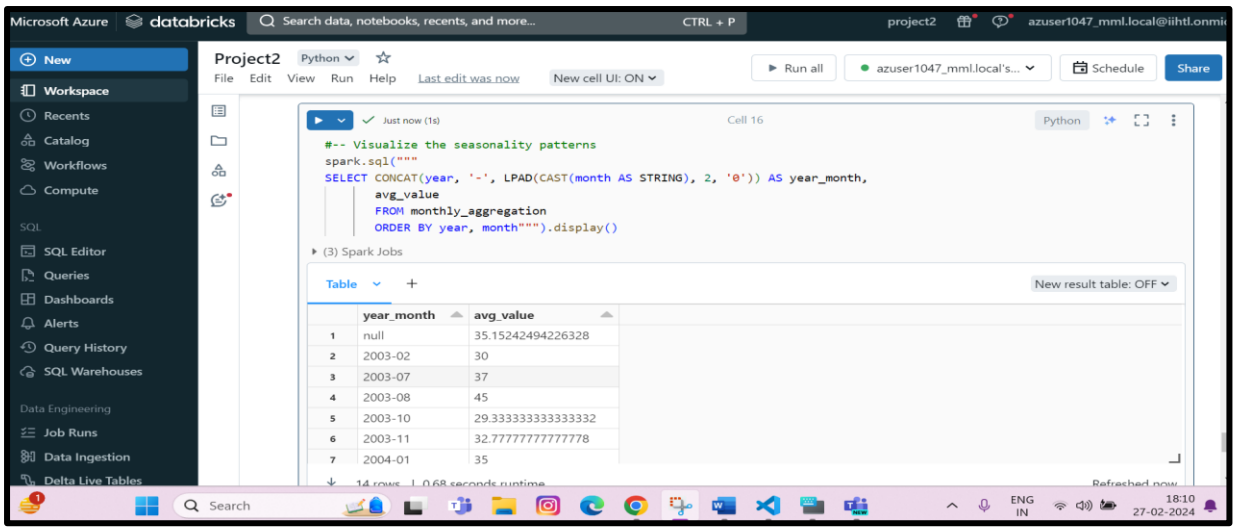
Output:



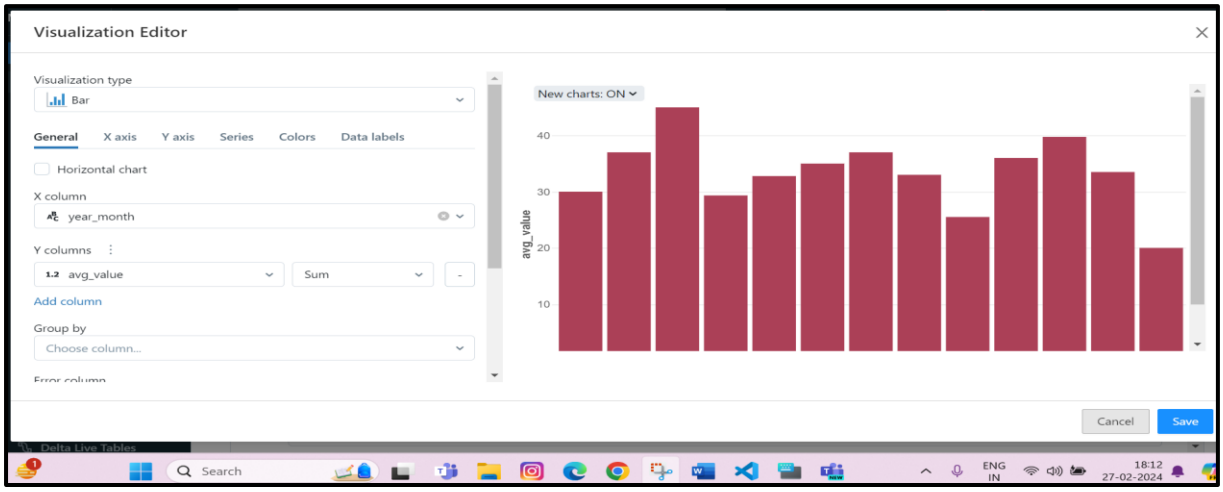
Analyse Seasonality:



Visualize Seasonality:



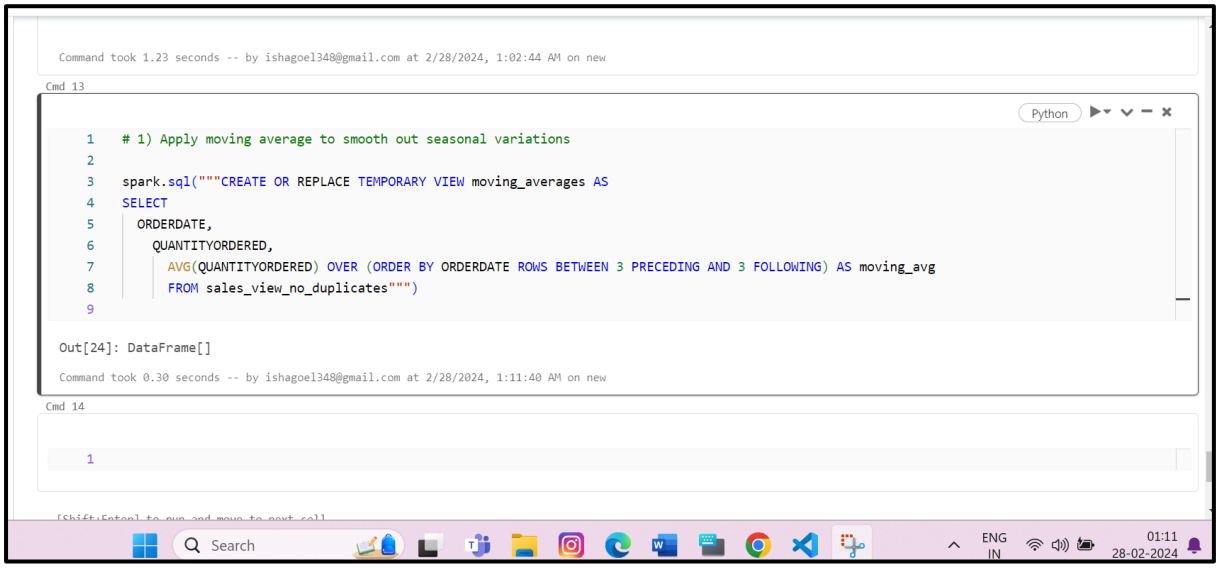
Through Bar Chart:



Moving Averages

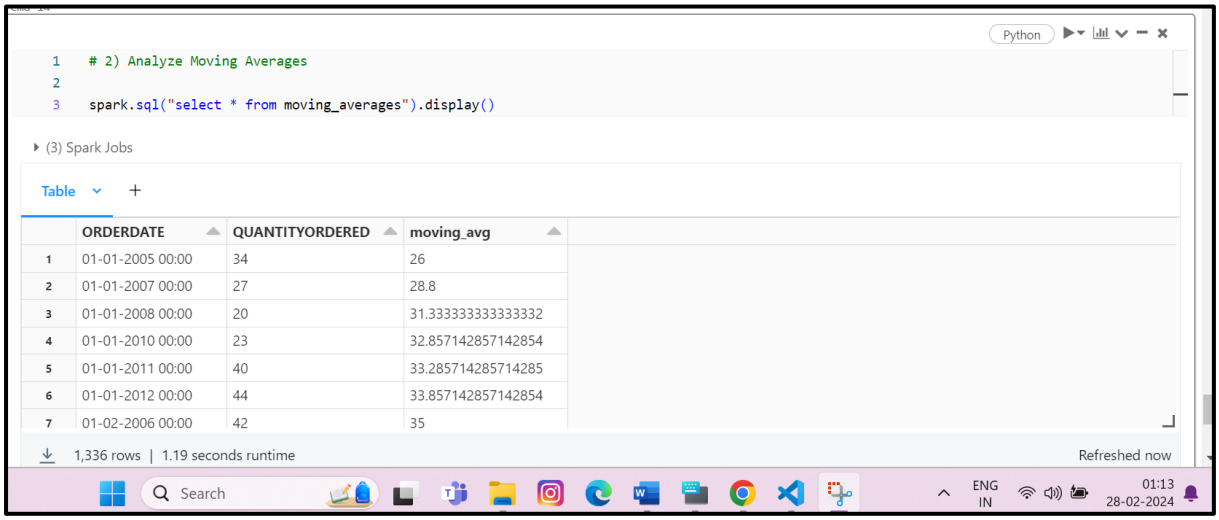
Moving averages are commonly used to smooth out fluctuations in time-series data and identify trends or patterns over a specific period. In the context of detecting seasonality using Spark SQL on Azure Databricks, we can apply moving averages to analyze the trend and smooth out the seasonal variations in the data.

Calculate moving averages



Analyse Moving Averages

Once you have calculated the moving averages, you can analyze them to identify trends and smooth out seasonal variations in the data.



```
# 2) Analyze Moving Averages

spark.sql("select * from moving_averages").display()
```

(3) Spark Jobs

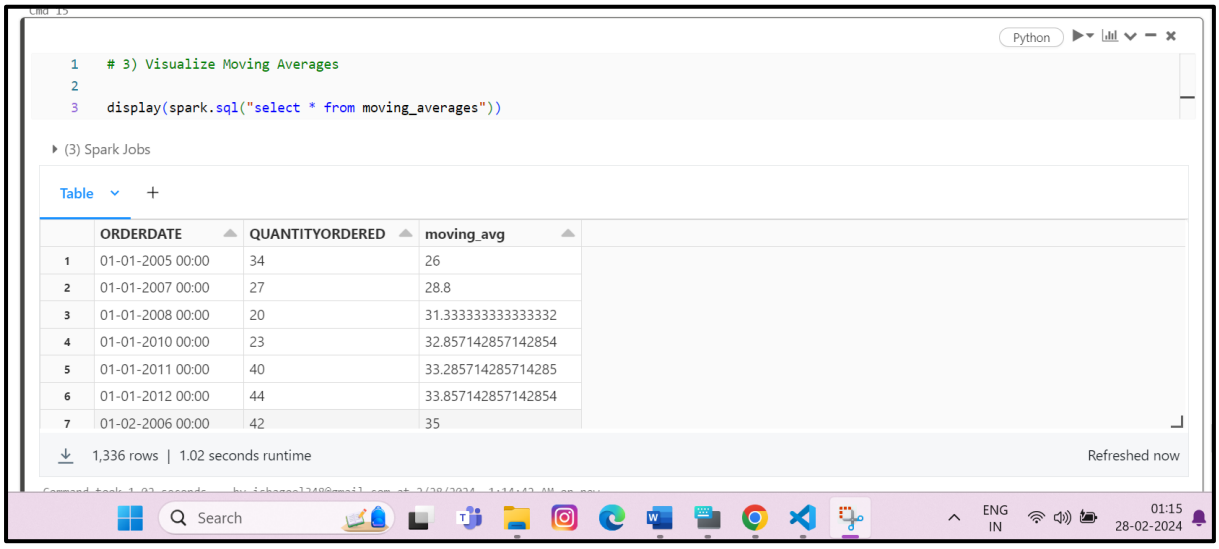
Table

	ORDERDATE	QUANTITYORDERED	moving_avg
1	01-01-2005 00:00	34	26
2	01-01-2007 00:00	27	28.8
3	01-01-2008 00:00	20	31.333333333333332
4	01-01-2010 00:00	23	32.857142857142854
5	01-01-2011 00:00	40	33.285714285714285
6	01-01-2012 00:00	44	33.857142857142854
7	01-02-2006 00:00	42	35

1,336 rows | 1.19 seconds runtime — Refreshed now

Visualize moving averages

Visualization can provide a clearer understanding of the trend and seasonal variations in the data after applying moving averages.



```
# 3) Visualize Moving Averages

display(spark.sql("select * from moving_averages"))
```

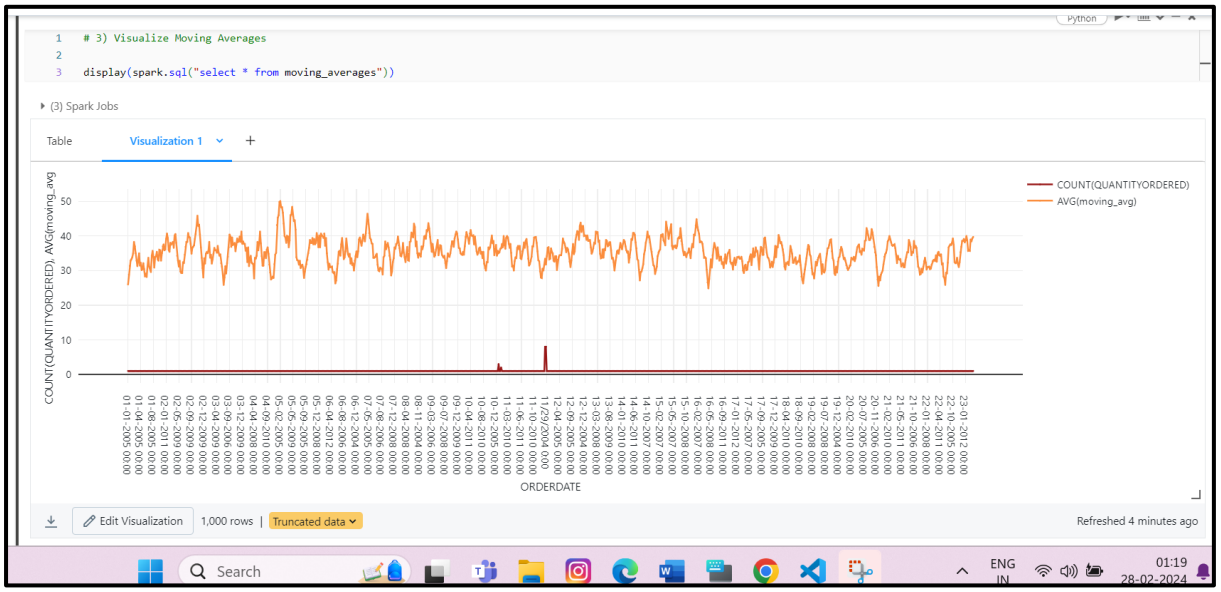
(3) Spark Jobs

Table

	ORDERDATE	QUANTITYORDERED	moving_avg
1	01-01-2005 00:00	34	26
2	01-01-2007 00:00	27	28.8
3	01-01-2008 00:00	20	31.333333333333332
4	01-01-2010 00:00	23	32.857142857142854
5	01-01-2011 00:00	40	33.285714285714285
6	01-01-2012 00:00	44	33.857142857142854
7	01-02-2006 00:00	42	35

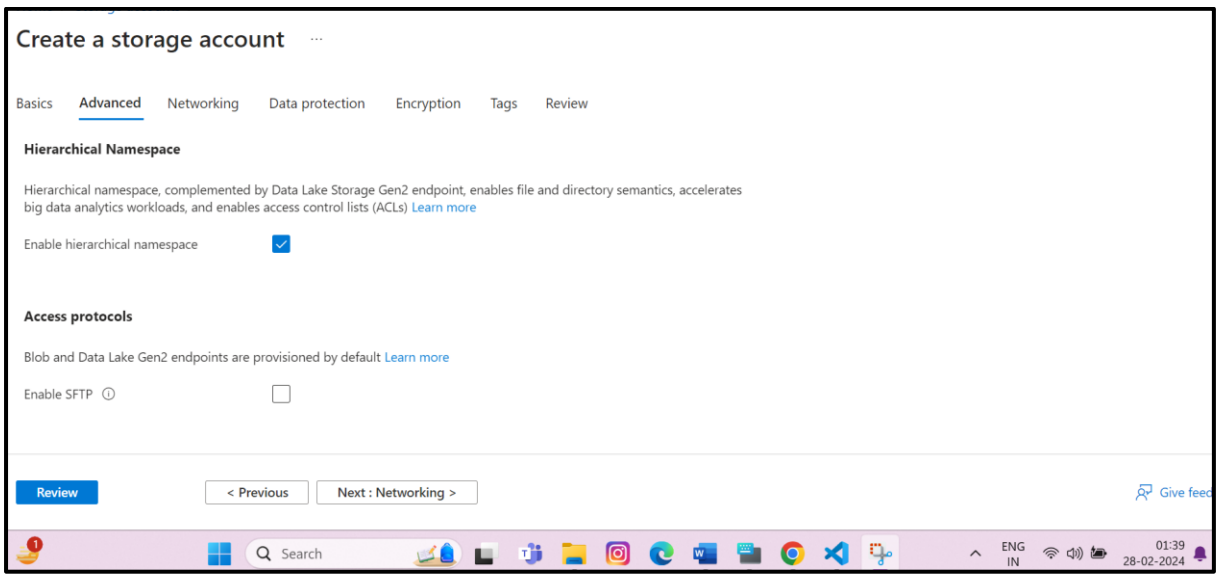
1,336 rows | 1.02 seconds runtime — Refreshed now

Through Line Chart:

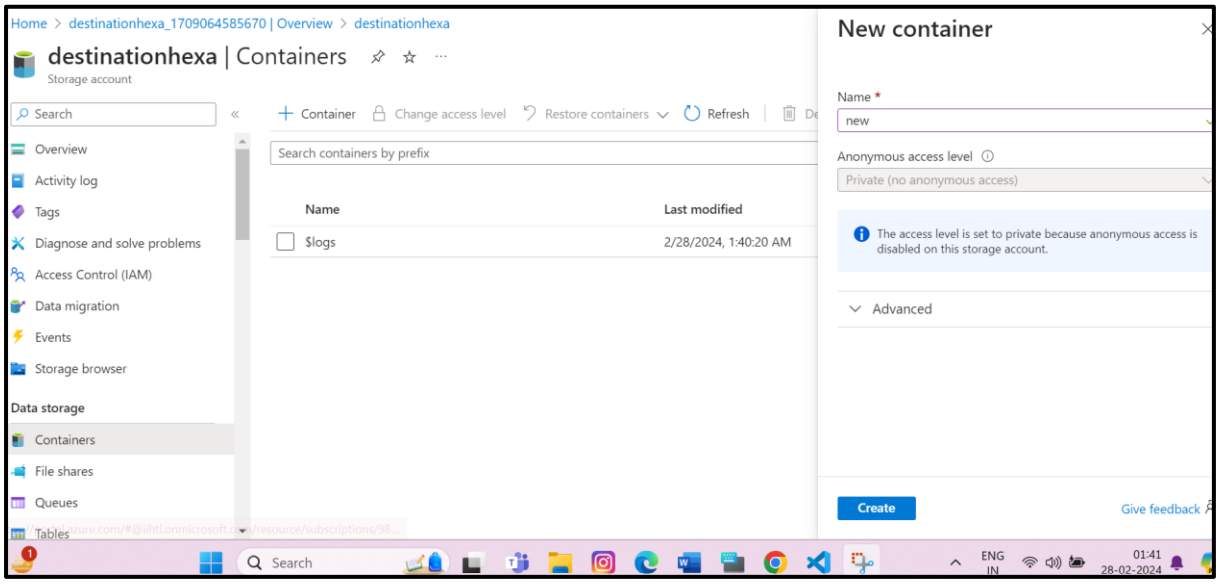


Writing the transformed file to destination

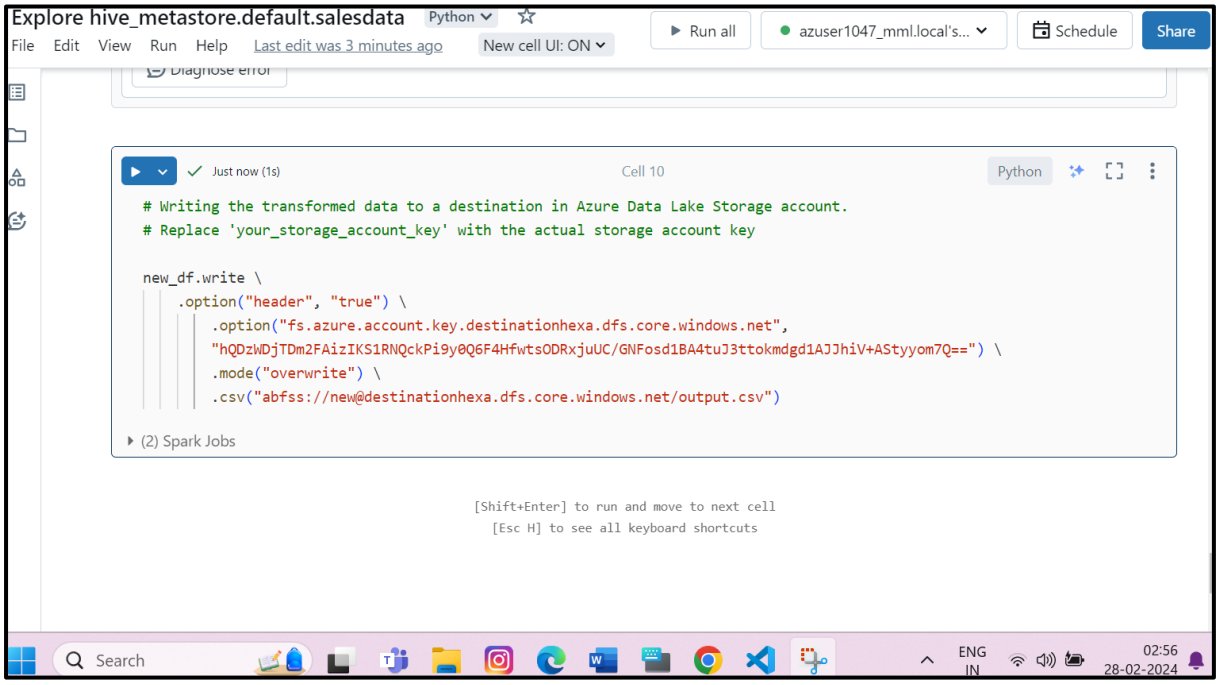
Create a destination storage account to write the transformed data.



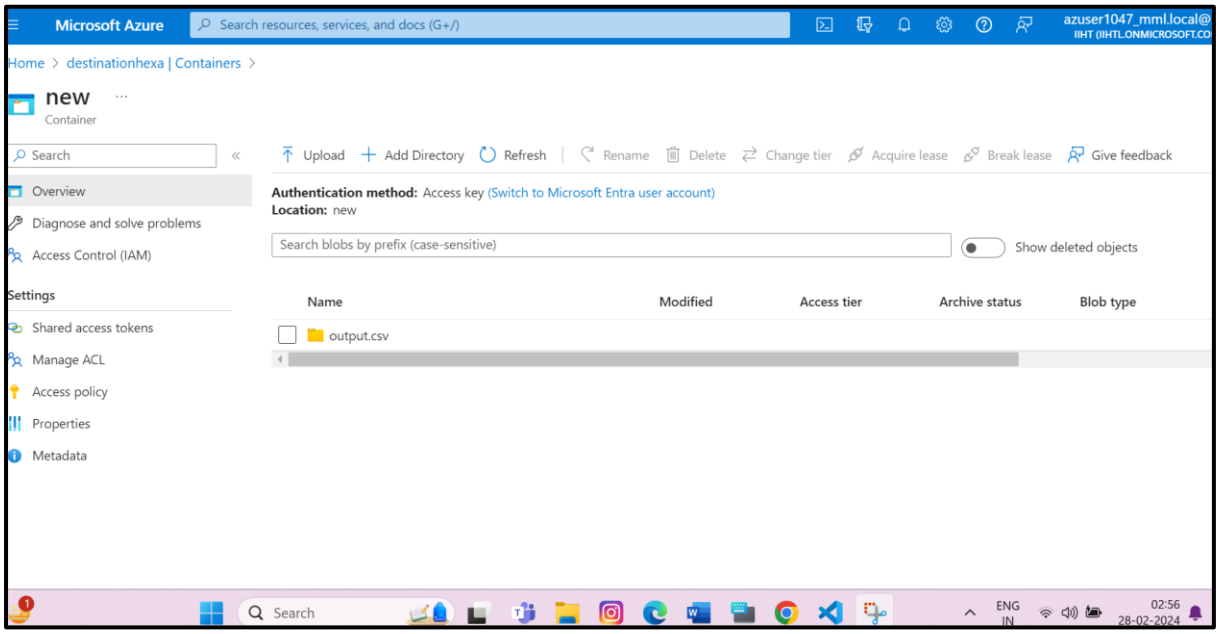
In the destination storage account create a container to write the transformed data:



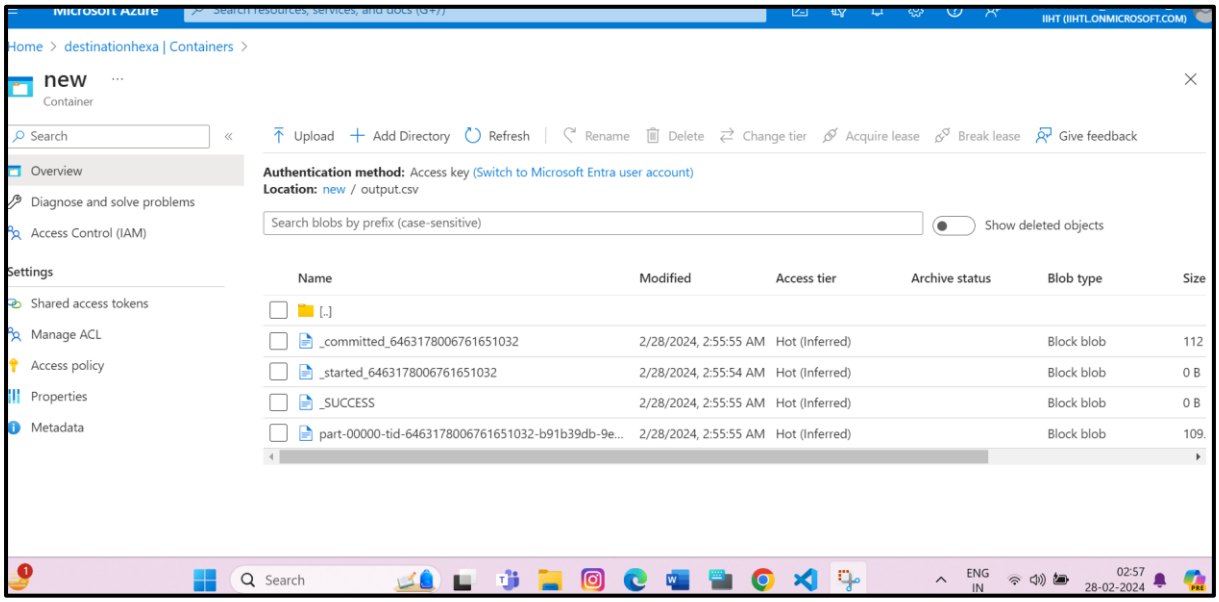
Write the transformed data to “new” in “destination” Azure Data Lake Storage:

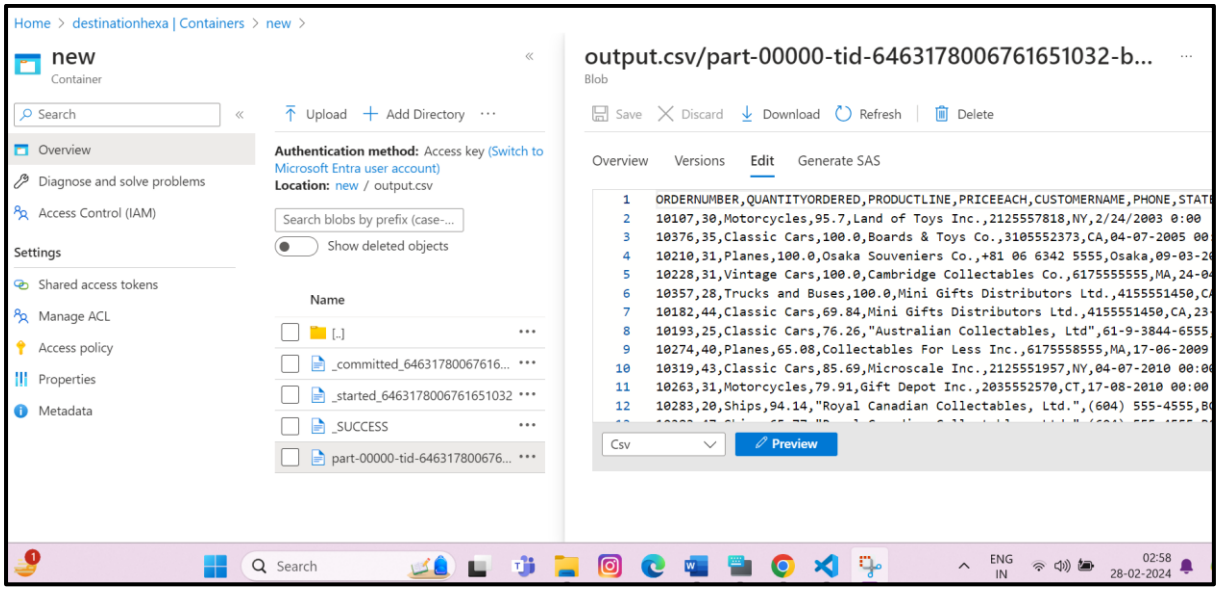


Writing into output.csv file inside the “new” container:



Files have been written to output.csv:





Conclusion

In conclusion, the project focused on leveraging Spark SQL on Azure Databricks for data analysis. The project demonstrated the effectiveness of using Spark SQL on Azure Databricks for data analysis tasks, particularly in detecting seasonality and identifying anomalies in time series data. The insights gained from this project can inform strategic decision-making and operational planning for various domains, including retail, finance, healthcare, and more**.**