# Portfolio Project:

# Analyzing data with Apache Spark

## Project Overview

This project focuses on analyzing data using Apache Spark within Microsoft Fabric, which serves as a powerful platform for data processing and analysis. The objective is to ingest data into a Fabric lakehouse and utilize PySpark for efficient data reading and analysis

## Objectives

1. **Creating a Workspace**
   - Set up a workspace in a tenant with Fabric capacity enabled.
2. **Creating a Lakehouse and Upload Files**
   - Establish a data lakehouse and upload necessary data files.
3. **Creating a Notebook**
   - Develop a notebook for interactive coding and data analysis.
4. **Data Analysis Steps**
   - Create a DataFrame
   - Explore data in a DataFrame
   - Aggregate and group data in a DataFrame
   - Use Spark to transform data files
   - Save the transformed data
   - Save data in partitioned files
   - Create a table
   - Run SQL code in a cell
   - View results as a chart
   - Get started with matplotlib
   - Use the seaborn library

# Experience

## Create a Workspace

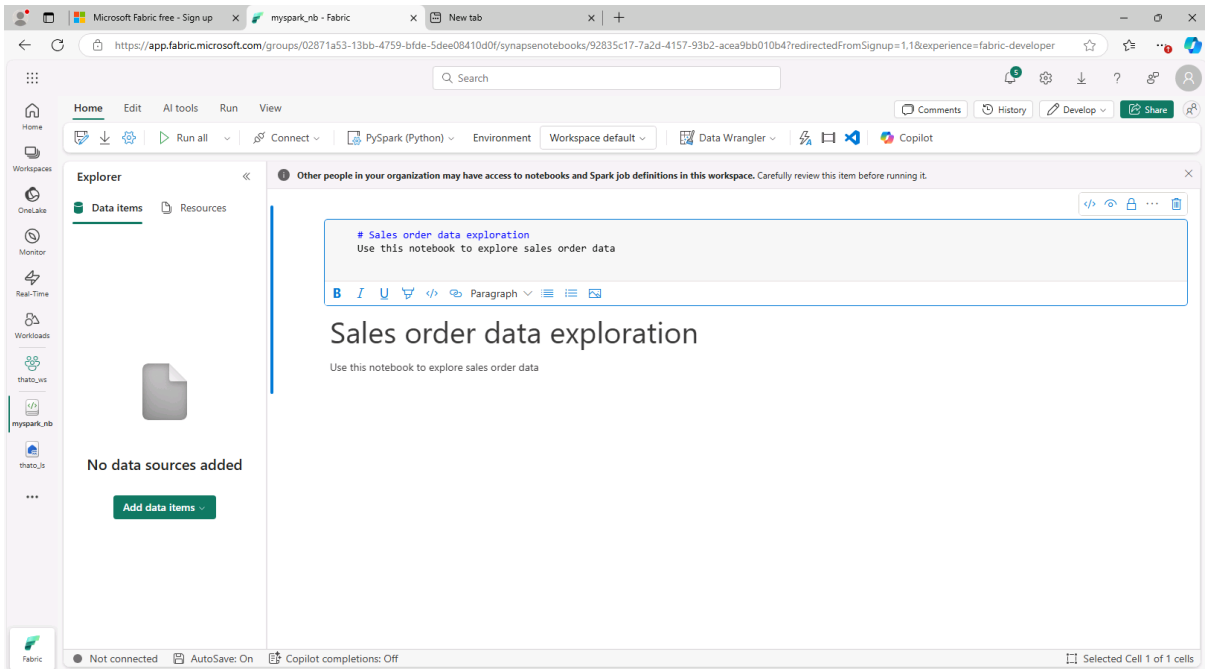
- Navigated to the Microsoft Fabric home page and signed in with credentials.
- Selected Workspaces from the menu bar and created.

## Create a Lakehouse and Upload Files

- In the menu bar, selected Create, then chose Lakehouse under the Data Engineering section, assigning it a unique name.
- Download the data files from GitHub and extract them to verify the presence of three CSV files: 2019.csv, 2020.csv, and 2021.csv.
- Uploaded the extracted orders folder to the lakehouse.

## Create a Notebook

- Selected Create from the menu bar, then chose Notebook under the Data Engineering section.
- Renamed the notebook to something descriptive and converted the first cell to a markdown cell with the title "Sales order data exploration".



## Create a DataFrame

- Loaded the 2019.csv file into a Spark DataFrame.

```python
df = spark.read.format("csv").option("header","false").load("Files/orders/2019.csv")
# df now is a Spark DataFrame containing CSV data from "Files/orders/2019.csv".
display(df)
```

✓ 2 sec - Command executed in 2 sec 521 ms by User1-53367231 on 12:30:30 AM, 7/29/25 PySpark (Python)

```python
orderSchema = StructType([
    StructField("SalesOrderNumber", StringType()),
    StructField("SalesOrderLineNumber", IntegerType()),
    StructField("OrderDate", DateType()),
    StructField("CustomerName", StringType()),
    StructField("Email", StringType()),
    StructField("Item", StringType()),
    StructField("Quantity", IntegerType()),
    StructField("UnitPrice", FloatType()),
    StructField("Tax", FloatType())
])

df = spark.read.format("csv").schema(orderSchema).load("Files/orders/2019.csv")

display(df)
```

✓ 6 sec - Command executed in 6 sec 253 ms by User1-53367231 on 12:32:11 AM, 7/29/25 PySpark (Python)

Spark jobs (1 of 1 succeeded) Resources Log

Table New chart Data Wrangler 9 columns, 1000 rows

Table view Download Filter by keyword

| | SalesOrderNumber | SalesOrderLineNumber | OrderDate | CustomerName | Email | Item | Quantity | UnitPrice | Ta |
|---|---|---|---|---|---|---|---|---|---|
| 1 | SO43701 | 1 | 2019-07-01 | Christy Zhu | christy12@... | Mountain-... | 1 | 3399.99 | 27 |
| 2 | SO43704 | 1 | 2019-07-01 | Julio Ruiz | julio1@adv... | Mountain-... | 1 | 3374.99 | 26 |
| 3 | SO43705 | 1 | 2019-07-01 | Curtis Lu | curtis9@ad... | Mountain-... | 1 | 3399.99 | 27 |
| 4 | SO43700 | 1 | 2019-07-01 | Ruben Prasad | ruben10@... | Road-650 ... | 1 | 699.0982 | 55 |
| 5 | SO43703 | 1 | 2019-07-01 | Albert Alvarez | albert7@a... | Road-150 ... | 1 | 3578.27 | 28 |
| 6 | SO43697 | 1 | 2019-07-01 | Cole Watson | cole1@adv... | Road-150 ... | 1 | 3578.27 | 28 |

Inspect

## Explore Data in a DataFrame

- Filtered the DataFrame to return only the CustomerName and Email columns.

```python
customers = df.select("CustomerName", "Email").where(df['Item']=='Road-250 Red, 52')
print(customers.count())
print(customers.distinct().count())

display(customers.distinct())
```

✓ 1 sec - Command executed in 1 sec 595 ms by User1-53367231 on 12:38:57 AM, 7/29/25

Table New chart

Table view

| | CustomerName | Email |
|---|---|---|
| 1 | Bridget Andersen | bridget15... |
| 2 | Mya Butler | mya14@ad... |
| 3 | Deanna Hernandez | deanna29... |
| 4 | Ricky Navarro | ricky10@a... |
| 5 | Omar Ye | omar9@ad... |
| 6 | Kellie Gutierrez | kellie9@ad... |
| 7 | Raymond Rana | raymond13... |
| 8 | Derrick Moreno | derrick6@a... |
| 9 | Megan Walker | megan25... |
| 10 | Edward Jackson | edward34... |
| 11 | Wayne She | wayne1@a... |
| 12 | Willie Shan | willie29@a... |
| 13 | Jake Guo | jake15@ad... |
| 14 | Sandra Lu | sandra18@... |
| 15 | Adam Kumar | adam24@a... |
| 16 | Lawrence Ramos | lawrence16... |
| 17 | Abigail Garcia | abigail57@... |

## Aggregate and Group Data in a DataFrame

Grouped the data by Item and sum the quantities.

```
productSales = df.select("Item", "Quantity").groupBy("Item").sum()

display(productSales)
```

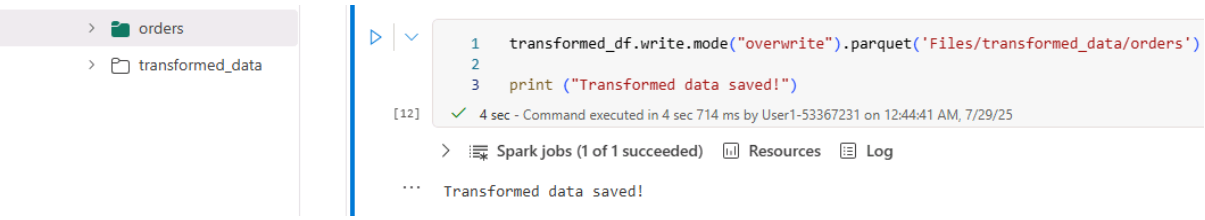
## Use Spark to Transform Data Files

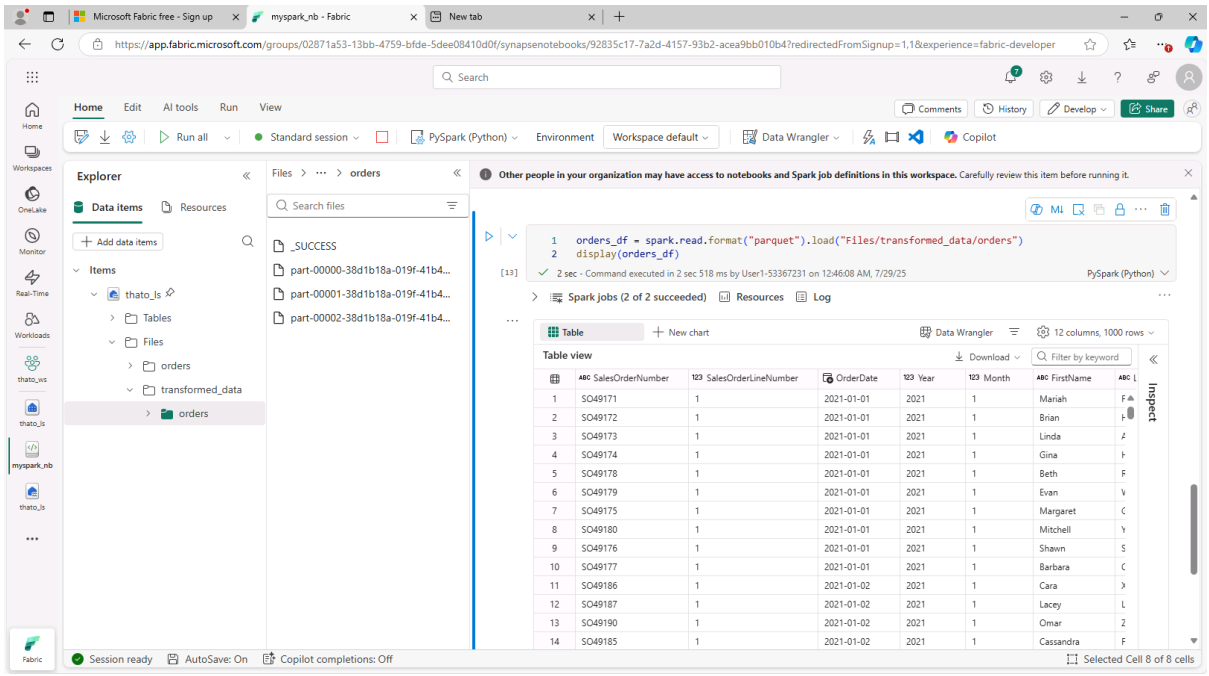
Transformed the DataFrame by adding Year and Month columns.

```
from pyspark.sql.functions import *

yearlySales = df.select(year(col("OrderDate")).alias("Year")).groupBy("Year").count().orderBy("Year")

display(yearlySales)
```

## Save the Transformed Data

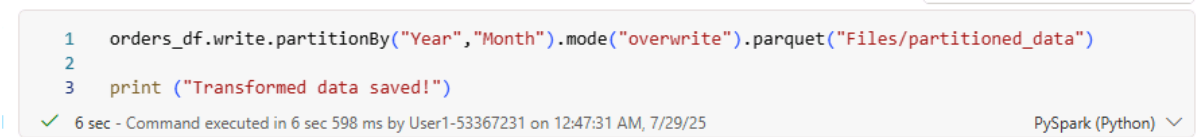
Saved the transformed DataFrame in Parquet format.



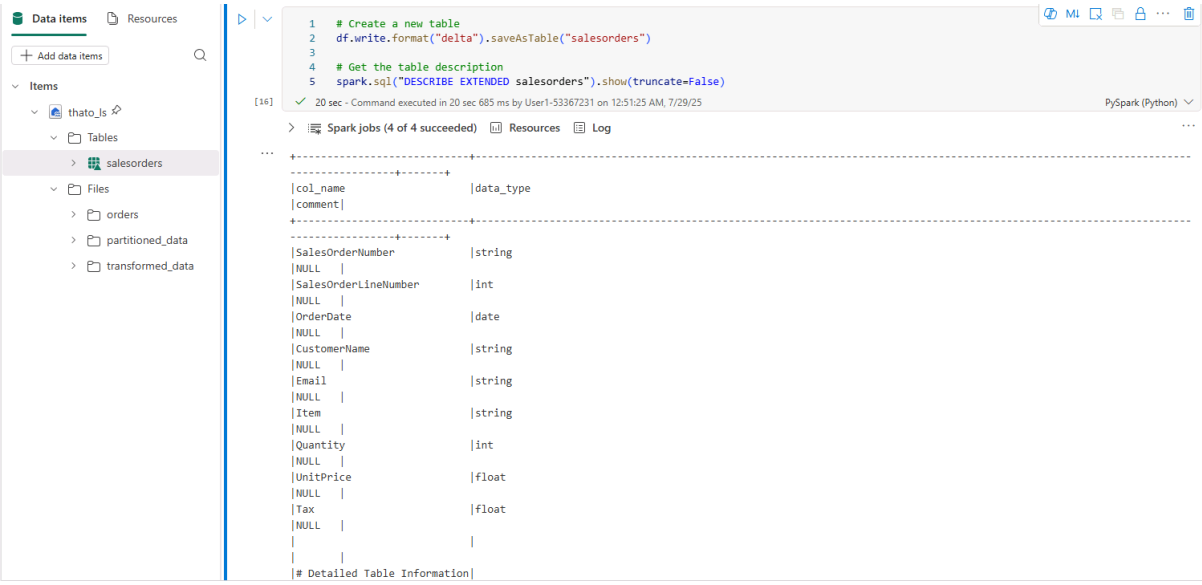

## Save Data in Partitioned Files

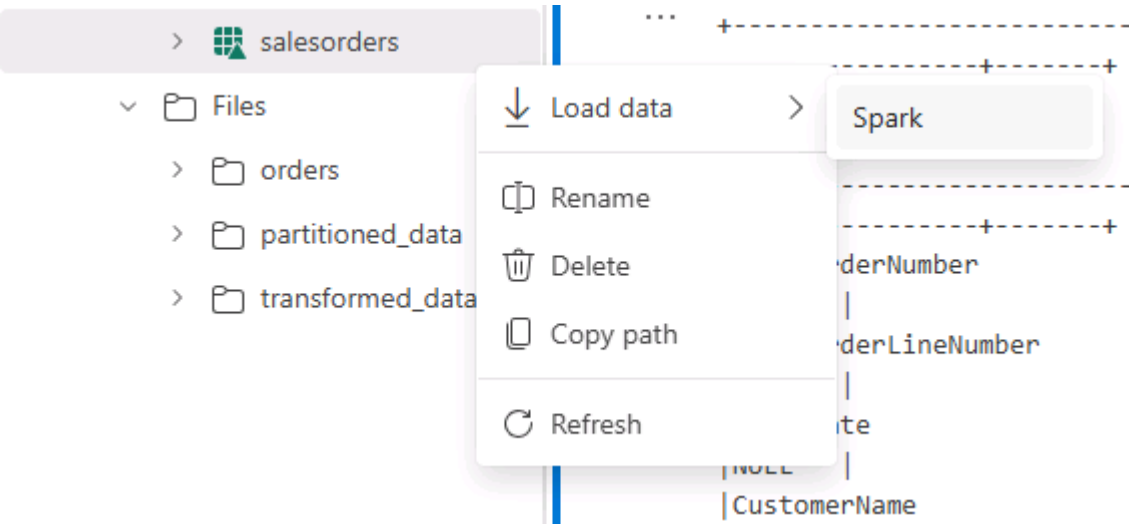
Saved the DataFrame partitioned by Year and Month.




## Create a Table

Created a new table from the DataFrame.




## Run SQL Code in a Cell

Executed SQL queries directly in a cell.

```
%%sql
SELECT YEAR(OrderDate) AS OrderYear,
       SUM((UnitPrice * Quantity) + Tax) AS GrossRevenue
FROM salesorders
GROUP BY YEAR(OrderDate)
ORDER BY OrderYear;
```

✓ 2 sec - Command executed in 2 sec 387 ms by User1-53367231 on 12:55:09 AM, 7/29/25

Spark jobs (3 of 3 succeeded) Resources

Table New chart

Table view

| | 123 OrderYear | 1.2 GrossRevenue |
|---|---|---|
| 1 | 2019 | 4172169.969970703 |
| 2 | 2020 | 6882259.268127441 |
| 3 | 2021 | 11547835.2916965... |

## View Results as a Chart

Displayed data from the salesorders view and created a chart.

```
%%sql
SELECT * FROM salesorders
```

✓ 1 sec - Command executed in 1 sec 453 ms by User1-53367231 on 12:56:06 AM, 7/29/25

Spark jobs (1 of 1 succeeded) Resources Log

Table New chart

Table view

| | ABC SalesOrderNumber | 123 SalesOrderLineNumber | OrderDate | ABC CustomerName | ABC Email | ABC Item | 123 Quantity | 1.2F UnitPrice | 1.2F Tax |
|---|---|---|---|---|---|---|---|---|---|
| 49 | SO43752 | 1 | 2019-07-07 | Luke Adams | luke46@ad... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 50 | SO43750 | 1 | 2019-07-07 | Stacey Lu | stacey12@... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 51 | SO43751 | 1 | 2019-07-07 | Darrell Raji | darrell9@a... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 52 | SO43746 | 1 | 2019-07-07 | Kelvin Liang | kelvin35@... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 53 | SO43753 | 1 | 2019-07-07 | Tabitha Arthur | tabitha5@... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 54 | SO43745 | 1 | 2019-07-07 | Manuel Fernandez | manuel14... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 55 | SO43747 | 1 | 2019-07-07 | Christian Jackson | christian47... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 56 | SO43748 | 1 | 2019-07-07 | Alexandria Hughes | alexandria1... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 57 | SO43749 | 1 | 2019-07-07 | Paige Murphy | paige39@a... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 58 | SO43756 | 1 | 2019-07-08 | Cedric Liang | cedric16@... | Road-650 ... | 1 | 699.09820556... | 55.927898... |
| 59 | SO43755 | 1 | 2019-07-08 | Kayla Long | kayla33@a... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 60 | SO43754 | 1 | 2019-07-08 | Morgan Phillips | morgan4@... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 61 | SO43758 | 1 | 2019-07-08 | Jackson Li | jackson28... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 62 | SO43761 | 1 | 2019-07-08 | Brenda Fernandez | brenda19... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |

# Explore Maplotlib

Visualized the data using matplotlib.

```
%%sql
SELECT * FROM salesorders
```

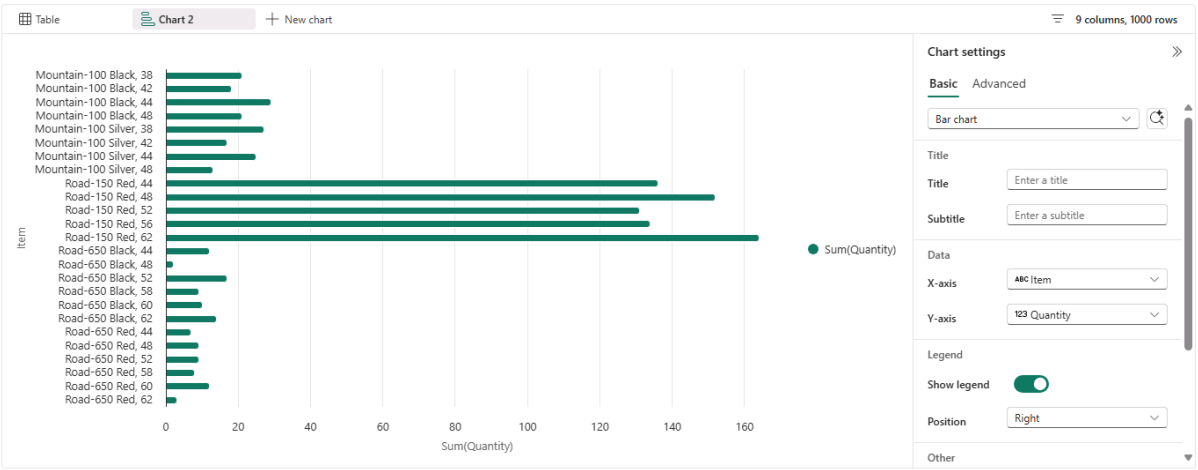
✓ 1 sec - Command executed in 1 sec 453 ms by User1-53367231 on 12:56:06 AM, 7/29/25

Spark jobs (1 of 1 succeeded) Resources Log

Table New chart

Table view

| | SalesOrderNumber | SalesOrderLineNumber | OrderDate | CustomerName | Email | Item | Quantity | UnitPrice | Tax |
|---|---|---|---|---|---|---|---|---|---|
| 49 | SO43752 | 1 | 2019-07-07 | Luke Adams | luke46@ad... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 50 | SO43750 | 1 | 2019-07-07 | Stacey Lu | stacey12@... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 51 | SO43751 | 1 | 2019-07-07 | Darrell Raji | darrell9@a... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 52 | SO43746 | 1 | 2019-07-07 | Kelvin Liang | kelvin35@... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 53 | SO43753 | 1 | 2019-07-07 | Tabitha Arthur | tabitha5@... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 54 | SO43745 | 1 | 2019-07-07 | Manuel Fernandez | manuel14... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 55 | SO43747 | 1 | 2019-07-07 | Christian Jackson | christian47... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 56 | SO43748 | 1 | 2019-07-07 | Alexandria Hughes | alexandria1... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 57 | SO43749 | 1 | 2019-07-07 | Paige Murphy | paige39@a... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 58 | SO43756 | 1 | 2019-07-08 | Cedric Liang | cedric16@... | Road-650 ... | 1 | 699.09820556... | 55.927898... |
| 59 | SO43755 | 1 | 2019-07-08 | Kayla Long | kayla33@a... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 60 | SO43754 | 1 | 2019-07-08 | Morgan Phillips | morgan4@... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 61 | SO43758 | 1 | 2019-07-08 | Jackson Li | jackson28... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |
| 62 | SO43761 | 1 | 2019-07-08 | Brenda Fernandez | brenda19... | Road-150 ... | 1 | 3578.2700195... | 286.26159... |



```python
sqlQuery = "SELECT CAST(YEAR(OrderDate) AS CHAR(4)) AS OrderYear, \
                SUM((UnitPrice * Quantity) + Tax) AS GrossRevenue, \
                COUNT(DISTINCT SalesOrderNumber) AS YearlyCounts \
            FROM salesorders \
            GROUP BY CAST(YEAR(OrderDate) AS CHAR(4)) \
            ORDER BY OrderYear"
df_spark = spark.sql(sqlQuery)
df_spark.show()
```

✓ 3 sec - Command executed in 3 sec 449 ms by User1-53367231 on 12:59:45 AM, 7/29/25

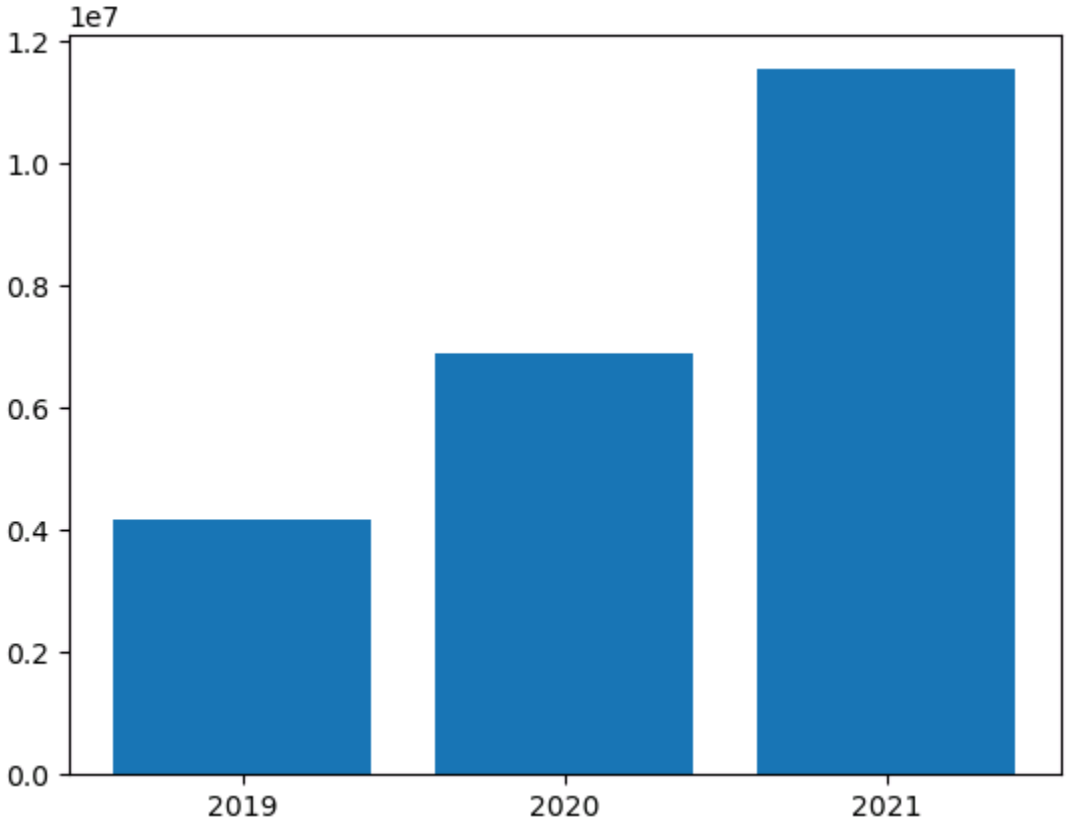
> Spark jobs (3 of 3 succeeded) Resources Log

```
+---------+--------------------+------------+
|OrderYear|        GrossRevenue|YearlyCounts|
+---------+--------------------+------------+
|     2019|   4172169.969970703|        1201|
|     2020|   6882259.268127441|        2733|
|     2021|1.1547835291696548E7|       12525|
+---------+--------------------+------------+
```

```python
from matplotlib import pyplot as plt

# matplotlib requires a Pandas dataframe, not a Spark one
df_sales = df_spark.toPandas()

# Create a bar plot of revenue by year
plt.bar(x=df_sales['OrderYear'], height=df_sales['GrossRevenue'])

# Display the plot
plt.show()
```

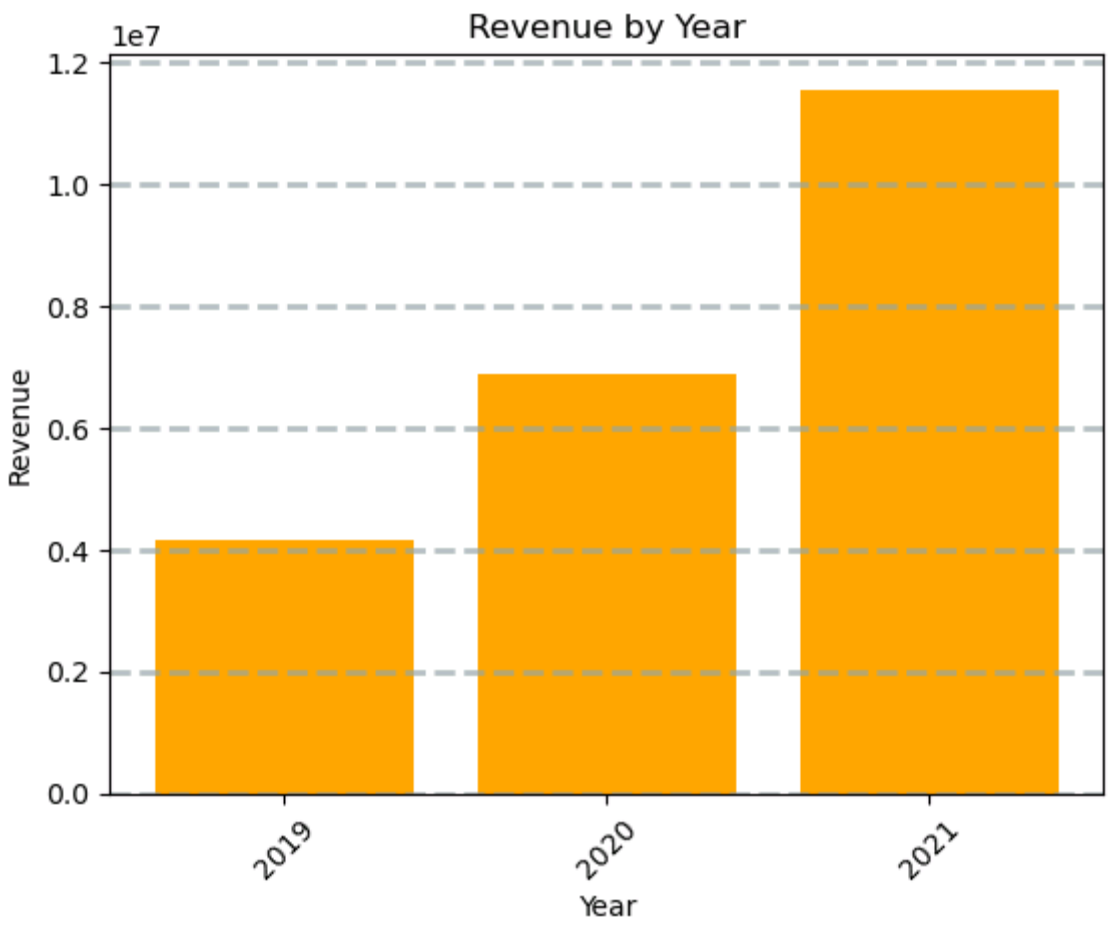
✓ 2 sec - Command executed in 2 sec 440 ms by User1-53367231 on 1:00:47 AM, 7/29/25

> Spark jobs (5 of 5 succeeded) Resources Log



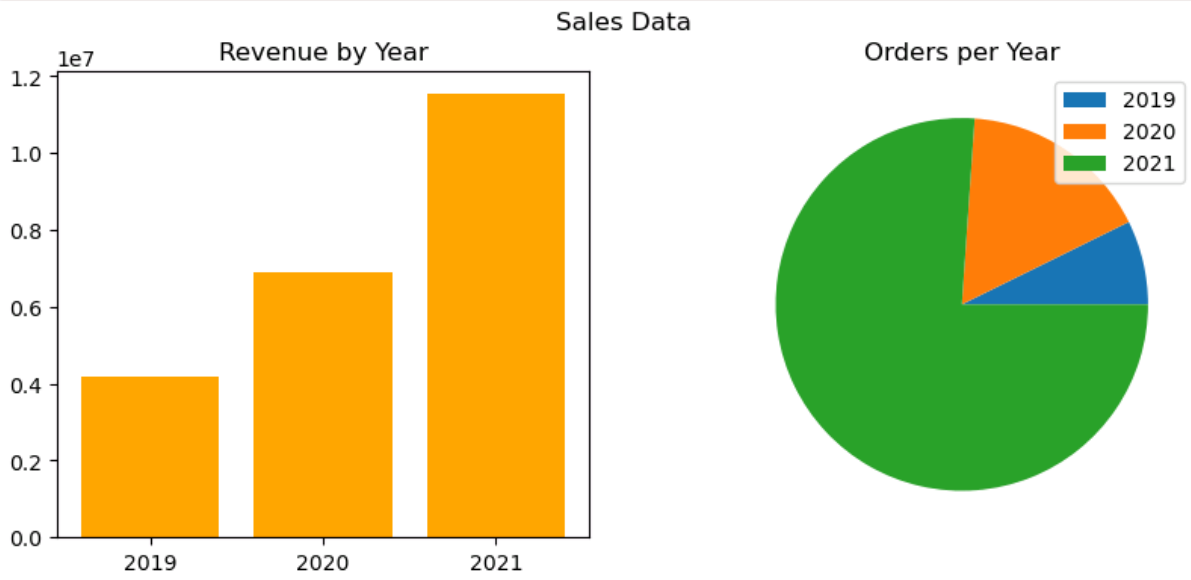
```python
from matplotlib import pyplot as plt

# Clear the plot area
plt.clf()

# Create a bar plot of revenue by year
plt.bar(x=df_sales['OrderYear'], height=df_sales['GrossRevenue'], color='orange')

# Customize the chart
plt.title('Revenue by Year')
plt.xlabel('Year')
plt.ylabel('Revenue')
plt.grid(color='#95a5a6', linestyle='--', linewidth=2, axis='y', alpha=0.7)
plt.xticks(rotation=45)

# Show the figure
plt.show()
```

✓ <1 sec - Command executed in 341 ms by User1-53367231 on 1:02:26 AM, 7/29/25



```python
from matplotlib import pyplot as plt

# Clear the plot area
plt.clf()

# Create a figure for 2 subplots (1 row, 2 columns)
fig, ax = plt.subplots(1, 2, figsize = (10,4))

# Create a bar plot of revenue by year on the first axis
ax[0].bar(x=df_sales['OrderYear'], height=df_sales['GrossRevenue'], color='orange')
ax[0].set_title('Revenue by Year')

# Create a pie chart of yearly order counts on the second axis
ax[1].pie(df_sales['YearlyCounts'])
ax[1].set_title('Orders per Year')
ax[1].legend(df_sales['OrderYear'])

# Add a title to the Figure
fig.suptitle('Sales Data')

# Show the figure
plt.show()
```

✓ <1 sec - Command executed in 869 ms by User1-53367231 on 1:08:58 AM, 7/29/25

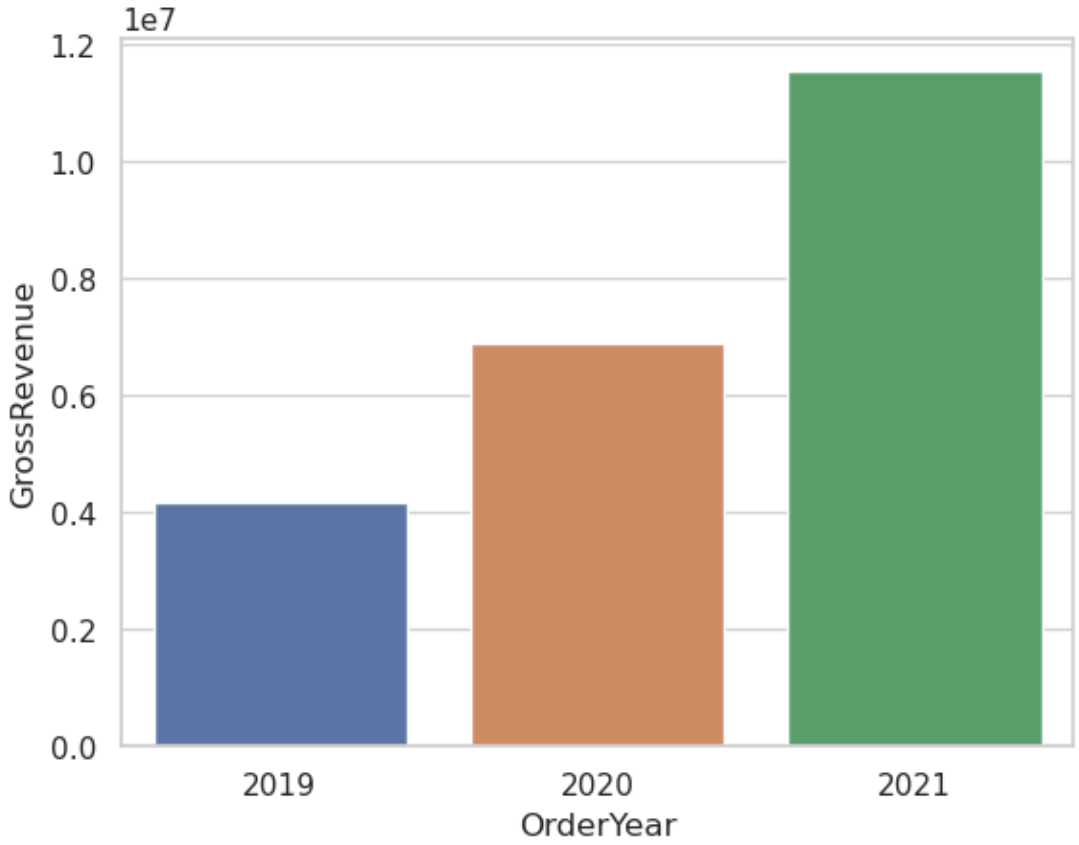



## Use the Seaborn Library

Created a bar chart using seaborn.

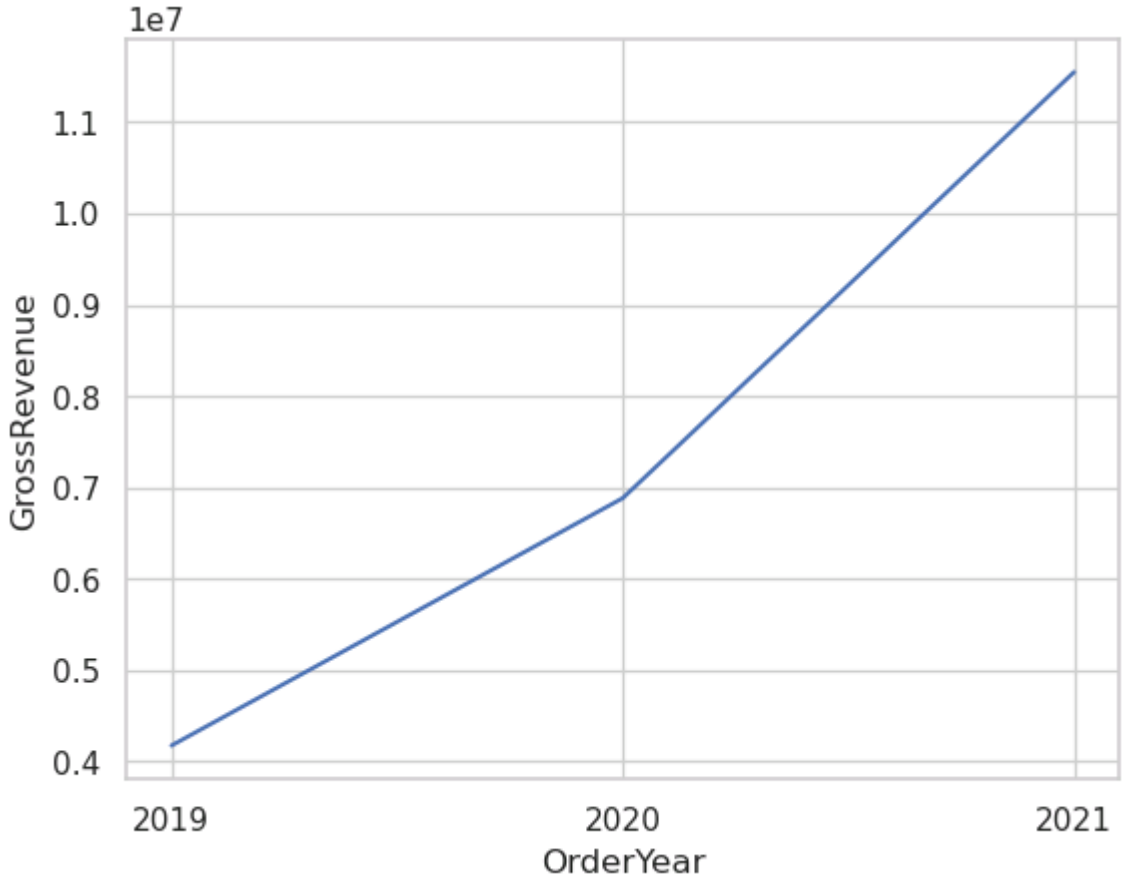
```python
import seaborn as sns

# Clear the plot area
plt.clf()

# Set the visual theme for seaborn
sns.set_theme(style="whitegrid")

# Create a bar chart
ax = sns.barplot(x="OrderYear", y="GrossRevenue", data=df_sales)

plt.show()
```

✓ <1 sec - Command executed in 304 ms by User1-53367231 on 1:18:14 AM, 7/29/25



```python
import seaborn as sns

# Clear the plot area
plt.clf()

# Create a line chart
ax = sns.lineplot(x="OrderYear", y="GrossRevenue", data=df_sales)

plt.show()
```

✓ <1 sec - Command executed in 323 ms by User1-53367231 on 1:20:09 AM, 7/29/25




## Results

- ✓ A workspace and lakehouse were successfully created, allowing for data ingestion.
- ✓ Data files were uploaded and processed into a Spark DataFrame.
- ✓ Various data analysis techniques were applied, including filtering, aggregation, and transformation of the data.
- ✓ The transformed data was saved in both Parquet format and partitioned files for efficient storage and retrieval.

- ✓ A table was created in the Spark metastore, and SQL queries were executed to analyze the data.
- ✓ Visualizations were generated using both matplotlib and seaborn, providing insights into the sales data.

## Conclusion

This lab provided a practical introduction to analyzing data with Apache Spark in Microsoft Fabric, encompassing workspace setup, data ingestion, and various data analysis techniques. Valuable insights were gained into the capabilities of Apache Spark for data processing and visualization, enabling effective manipulation of data using Spark and Python libraries.

## Resources

Source folder: https://github.com/MicrosoftLearning/dp-data/raw/main/orders.zip

GitHub: https://github.com/ThatoMTNG/Microsoft-Fabric-Analytics-Engineer-DP-600-

## Mentions

Project Author: Thato Metsing (https://www.linkedin.com/in/thatometsing/)

Project Mentor: Maureen Direro (https://www.linkedin.com/in/maureen-direro-46a6b220/)