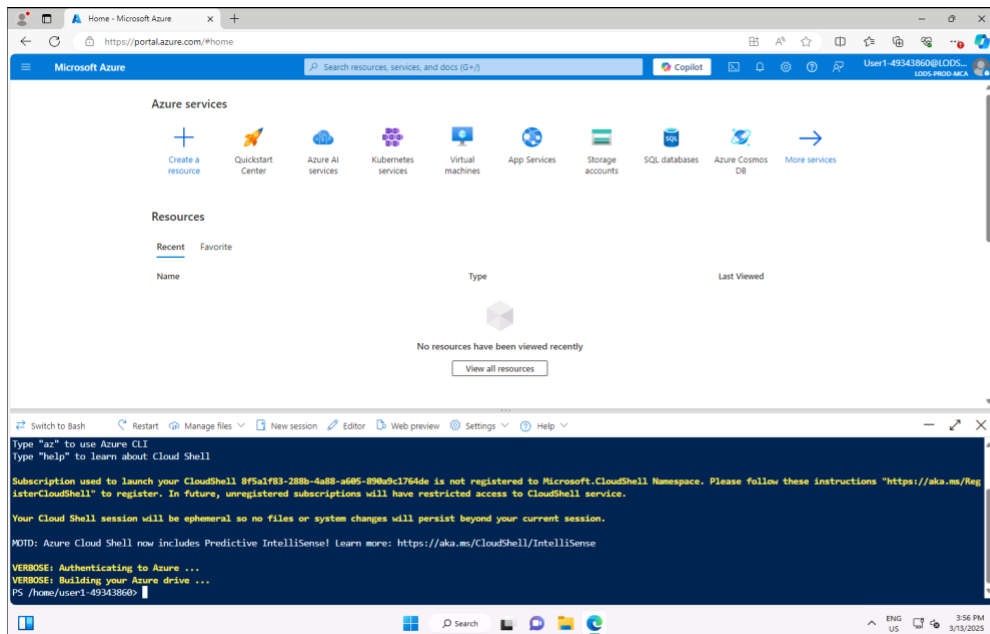


Lab 5 Synapse Spark

BUAN 6390.001 – Analytics Practicum

Name: Swastik Bhatnagar
Net ID: Sxb220210

Connecting to PowerShell:



Microsoft Azure

Azure services

Resources

Recent

Name

Type

Last Viewed

No resources have been viewed recently

View all resources

Switch to Bash Restart Manage files New session Editor Web preview Settings Help

Type "az" to use Azure CLI
Type "help" to learn about Cloud Shell

Subscription used to launch your CloudShell 8f5a1f83-288b-4a88-a605-890a3c1764de is not registered to Microsoft.CloudShell Namespace. Please follow these instructions "https://aka.ms/RegisterCloudShell" to register. In future, unregistered subscriptions will have restricted access to CloudShell service.

Your Cloud Shell session will be ephemeral so no files or system changes will persist beyond your current session.

MOHD: Azure Cloud Shell now includes Predictive IntelliSense! Learn more: https://aka.ms/CloudShell/IntelliSense

VERBOSE: Authenticating to Azure ...
VERBOSE: Building your Azure drive ...
PS /home/user1-49343860/

Synapse Spark
1 Hr 51 Min Remaining

Instructions Resources Help

Note: If you have previously created a cloud shell that uses a Bash environment, use the drop-down menu at the top left of the cloud shell pane to change it to **PowerShell**.

3. Note that you can resize the cloud shell by dragging the separator bar at the top of the pane, or by using the \rightarrow , \leftarrow , and \times icons at the top right of the pane. For more information about using the Azure Cloud Shell, see the [Azure Cloud Shell documentation](#).

4. In the PowerShell pane, enter the following commands to clone this repo:

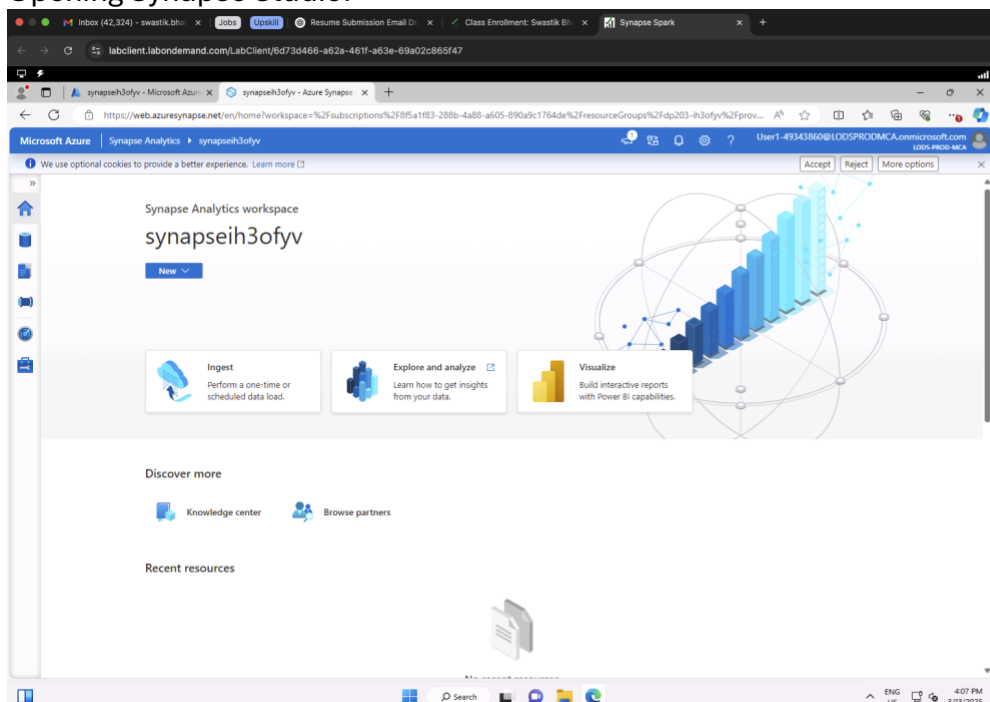
```
git clone https://github.com/Micro
```

5. After the repo has been cloned, enter the following commands to change to the folder for this lab and run the **setup.ps1** script it contains:

```
cd dp203/Allfiles/Labs/05  
./setup.ps1
```

6. If prompted, choose which subscription you want to

Opening Synapse Studio:



Microsoft Azure

Synapse Analytics workspace

synapseih3ofyv

New

Ingest

Perform a one-time or scheduled data load.

Explore and analyze

Learn how to get insights from your data.

Visualize

Build interactive reports with Power BI capabilities.

Discover more

Knowledge center

Browse partners

Recent resources

Synapse Spark
1 Hr 40 Min Remaining

Instructions Resources Help

Query data in files

The script provisions an Azure Synapse Analytics workspace and an Azure Storage account to host the data lake, then uploads some data files to the data lake.

View files in the data lake

1. After the script has completed, in the Azure portal, go to the **dp203-xxxxxx** resource group that it created, and select your Synapse workspace.

2. In the **Overview** page for your Synapse workspace, in the **Open Synapse Studio** card, select **Open** to open Synapse Studio in a new browser tab, signing in if prompted.

3. On the left side of Synapse Studio, use the \gg icon to expand the menu - this reveals the different pages within Synapse Studio that you'll use to manage resources and perform data analytics tasks.

4. On the **Manage** page, select the **Apache Spark pools** tab and note that a Spark pool with a name similar to **spark*xxxxxx** has been provisioned in the workspace. Later you will use this Spark pool to load and analyze data from files in the data lake storage for the workspace.

5. On the **Data** page, view the **Linked** tab and verify that your workspace includes a link to your Azure Data Lake Storage Gen2 storage account, which should have a name similar to **synapse*xxxxxx** (**Primary - datalake*xxxxxx**).

6. Expand your storage account and verify that it contains a file system container named **files**.

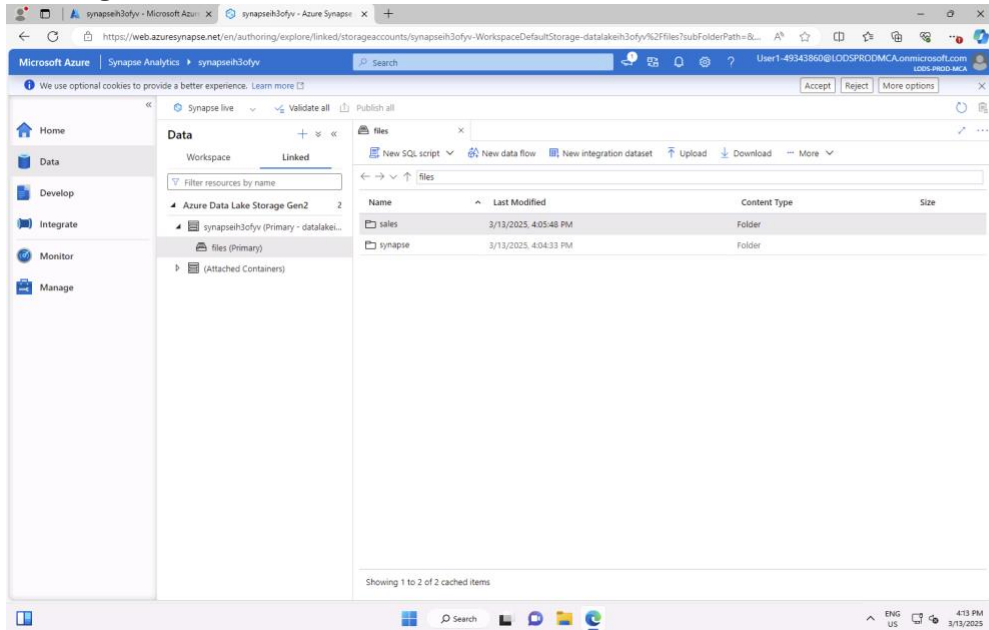
7. Select the **files** container, and note that it contains folders named **sales** and **synapse**. The **synapse** folder is used by Azure Synapse, and the **sales** folder contains the data files you are going to query.

8. Open the **sales** folder and the **orders** folder it contains, and observe that the **orders** folder contains .csv files for three years of sales data.

9. Right-click any of the files and select **Preview** to see the data it contains. Note that the files do not contain a header row, so you can unselect the option to display column headers.

Use Spark to analyze data

Viewing files in the data lake:



Microsoft Azure | Synapse Analytics | synapseh3ofyv

Workspace: Linked

Filter resources by name

Azure Data Lake Storage Gen2

synapseh3ofyv (Primary - datalake...)

files (Primary)

(Attached Containers)

Name	Last Modified	Content Type	Size
sales	3/13/2025, 4:05:48 PM	Folder	
synapse	3/13/2025, 4:04:33 PM	Folder	

Showing 1 to 2 of 2 cached items

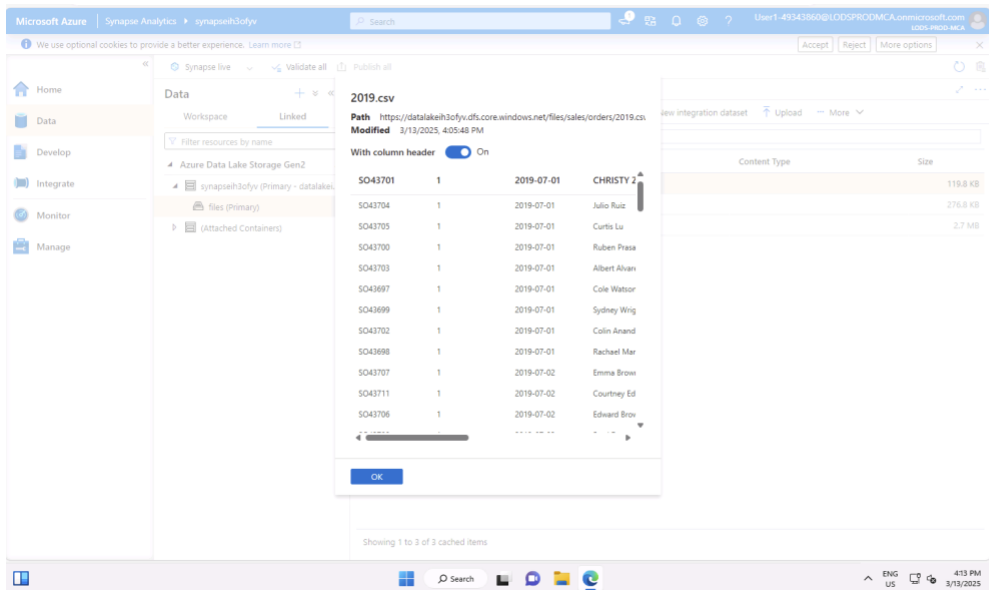
1 Hr 35 Min Remaining

Instructions Resources Help

- After the script has completed, in the Azure portal, go to the **dp203-xxxxxxx** resource group that it created, and select your Synapse workspace.
- In the **Overview** page for your Synapse workspace, in the **Open Synapse Studio** card, select **Open** to open Synapse Studio in a new browser tab, signing in if prompted.
- On the left side of Synapse Studio, use the **>>** icon to expand the menu - this reveals the different pages within Synapse Studio that you'll use to manage resources and perform data analytics tasks.
- On the **Manage** page, select the **Apache Spark pools** tab and note that a Spark pool with a name similar to **spark*xxxxxxx*** has been provisioned in the workspace. Later you will use this Spark pool to load and analyze data from files in the data lake storage for the workspace.
- On the **Data** page, view the **Linked** tab and verify that your workspace includes a link to your Azure Data Lake Storage Gen2 storage account, which should have a name similar to **synapse*xxxxxxx*** (**Primary - datalake*xxxxxxx***).
- Expand your storage account and verify that it contains a file system container named **files**.
- Select the **files** container, and note that it contains folders named **sales** and **synapse**. The **synapse** folder is used by Azure Synapse, and the **sales** folder contains the data files you are going to query.
- Open the **sales** folder and the **orders** folder it contains, and observe that the **orders** folder contains **.csv** files for three years of sales data.
- Right-click any of the files and select **Preview** to see the data it contains. Note that the files do not contain a header row, so you can unselect the option to display column headers.

Use Spark to explore data

- Select any of the files in the **orders** folder, and then in the **New notebook** list on the toolbar, select **Load to DataFrame**. A dataframe is a structure in Spark that represents a tabular dataset.
- In the new **Notebook 1** tab that opens, in the **Attach** to list, select your Spark pool



Microsoft Azure | Synapse Analytics | synapseh3ofyv

Workspace: Linked

Filter resources by name

Azure Data Lake Storage Gen2

synapseh3ofyv (Primary - datalake...)

files (Primary)

(Attached Containers)

Path	Modified	With column header	Content Type	Size
https://datakeih3ofyv.dfs.core.windows.net/files/sales/orders/2019.csv	3/13/2025, 4:05:48 PM	On		
2019.csv	3/13/2025, 4:05:48 PM	On		
2020.csv	3/13/2025, 4:05:48 PM	On		

Showing 1 to 3 of 3 cached items

1 Hr 35 Min Remaining

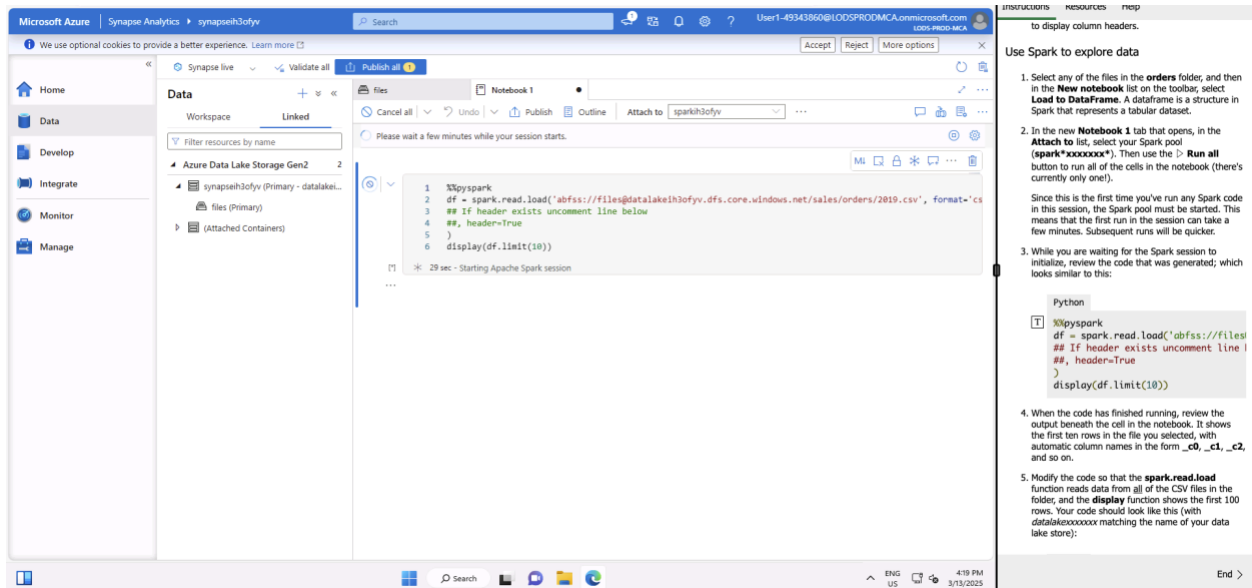
Instructions Resources Help

- After the script has completed, in the Azure portal, go to the **dp203-xxxxxxx** resource group that it created, and select your Synapse workspace.
- In the **Overview** page for your Synapse workspace, in the **Open Synapse Studio** card, select **Open** to open Synapse Studio in a new browser tab, signing in if prompted.
- On the left side of Synapse Studio, use the **>>** icon to expand the menu - this reveals the different pages within Synapse Studio that you'll use to manage resources and perform data analytics tasks.
- On the **Manage** page, select the **Apache Spark pools** tab and note that a Spark pool with a name similar to **spark*xxxxxxx*** has been provisioned in the workspace. Later you will use this Spark pool to load and analyze data from files in the data lake storage for the workspace.
- On the **Data** page, view the **Linked** tab and verify that your workspace includes a link to your Azure Data Lake Storage Gen2 storage account, which should have a name similar to **synapse*xxxxxxx*** (**Primary - datalake*xxxxxxx***).
- Expand your storage account and verify that it contains a file system container named **files**.
- Select the **files** container, and note that it contains folders named **sales** and **synapse**. The **synapse** folder is used by Azure Synapse, and the **sales** folder contains the data files you are going to query.
- Open the **sales** folder and the **orders** folder it contains, and observe that the **orders** folder contains **.csv** files for three years of sales data.
- Right-click any of the files and select **Preview** to see the data it contains. Note that the files do not contain a header row, so you can unselect the option to display column headers.

Use Spark to explore data

- Select any of the files in the **orders** folder, and then in the **New notebook** list on the toolbar, select **Load to DataFrame**. A dataframe is a structure in Spark that represents a tabular dataset.
- In the new **Notebook 1** tab that opens, in the **Attach** to list, select your Spark pool

Loading data to DataFrame and starting Spark pool:



Microsoft Azure | Synapse Analytics | synapseh3ofyv

Home Data Develop Integrate Monitor Manage

Workspace Linked

Filter resources by name

Azure Data Lake Storage Gen2

synapseh3ofyv (Primary - datalake...)

files (Primary)

(Attached Container)

Cancel all | Undo | Publish | Outline | Attach to | sparkh3ofyv

Please wait a few minutes while your session starts.

```
1 %%pyspark
2 df = spark.read.load('abfss://files@datalakeh3ofyv.dfs.core.windows.net/sales/orders/2019.csv', format='csv'
3
4 ## If header exists uncomment line below
5 ##, header=True
6 )
7 display(df.limit(10))
```

29 sec - Starting Apache Spark session

Python

```
1 %%pyspark
2 df = spark.read.load('abfss://files@datalakeh3ofyv.dfs.core.windows.net/sales/orders/2019.csv', format='csv'
3
4 ## If header exists uncomment line below
5 ##, header=True
6 )
7 display(df.limit(10))
```

1. Select any of the files in the **orders** folder, and then in the **New notebook** list on the toolbar, select **Load to DataFrame**. A dataframe is a structure in Spark that represents a tabular dataset.

2. In the new **Notebook 1** tab that opens, in the **Attach to** list, select your Spark pool (**sparkxxxxxx**). Then use the **Run all** button to run all of the cells in the notebook (there's currently only one!).

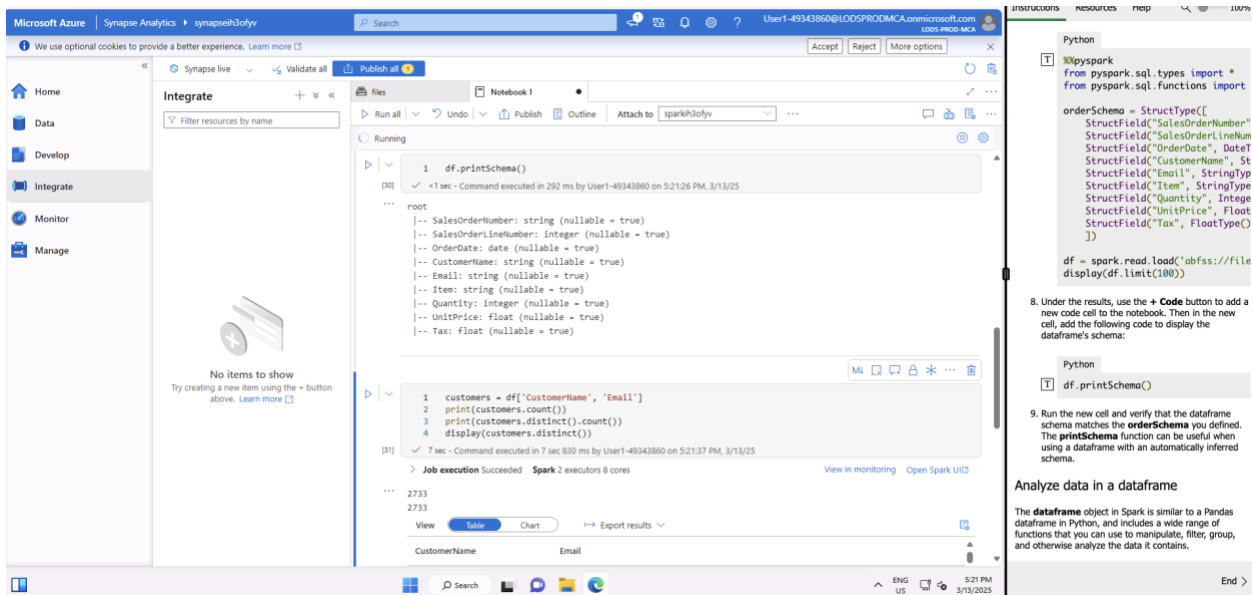
Since this is the first time you've run any Spark code in this session, the Spark pool must be started. This means that the first run in the session can take a few minutes. Subsequent runs will be quicker.

3. While you are waiting for the Spark session to initialize, review the code that was generated, which looks similar to this:

4. When the code has finished running, review the output beneath the cell in the notebook. It shows the first ten rows in the file you selected, with automatic column names in the form **_c0**, **_c1**, **_c2**, and so on.

5. Modify the code so that the **spark.read.load** function reads data from all of the CSV files in the folder, and the **display** function shows the first 100 rows. Your code should look like this (with **datalexxxxxxx** matching the name of your data lake store):

End >



Microsoft Azure | Synapse Analytics | synapseh3ofyv

Home Data Develop Integrate Monitor Manage

Integrate

Filter resources by name

No items to show
Try creating a new item using the + button above. [Learn more](#)

files

Run all | Undo | Publish | Outline | Attach to | sparkh3ofyv

```
1 df.printSchema()
2 print(customers.count())
3 print(customers.distinct().count())
4 display(customers.distinct())
```

Job execution Succeeded Spark 2 executors 8 cores

Python

```
1 df.printSchema()
```

8. Under the results, use the **+ Code** button to add a new code cell to the notebook. Then in the new cell, add the following code to display the dataframe's schema:

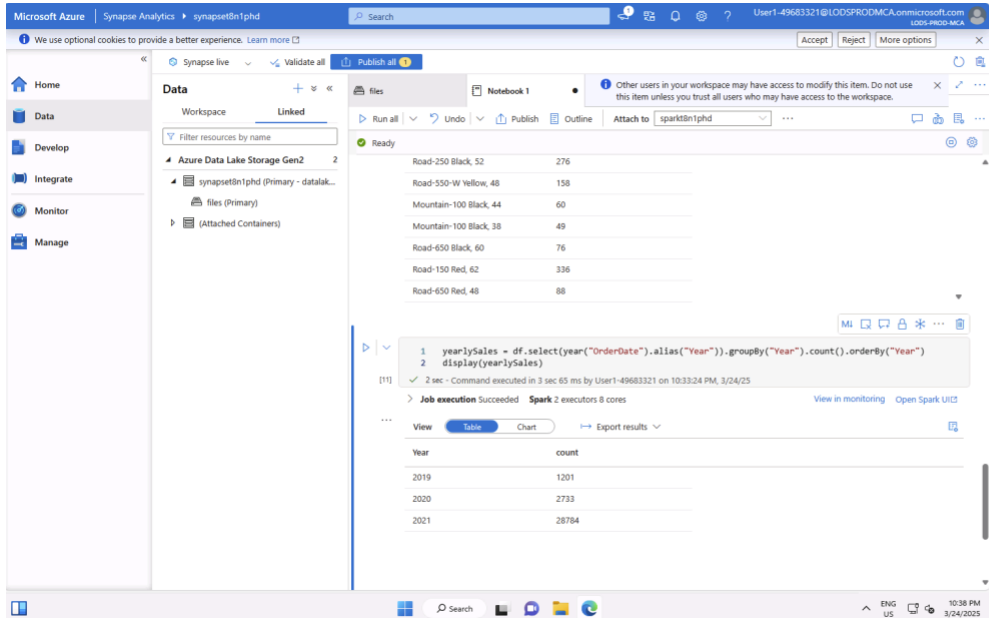
9. Run the new cell and verify that the dataframe schema matches the **orderSchema** you defined. The **printSchema** function can be useful when using a dataframe with an automatically inferred schema.

Analyze data in a dataframe

The **dataframe** object in Spark is similar to a Pandas dataframe in Python, and includes a wide range of functions that you can use to manipulate, filter, group, and otherwise analyze the data it contains.

End >

Aggregating and grouping data in a dataframe:



The screenshot shows the Microsoft Azure Synapse Analytics interface. On the left, the 'Data' section lists 'Azure Data Lake Storage Gen2' and 'synapsestn1phd'. The main area displays a notebook with two code cells. The first cell contains Python code to filter data for 'Road-250 Black, 52' and display it. The second cell contains Python code to group data by year and count the number of sales orders per year. The output of the second cell is a table showing the count of sales orders for each year from 2019 to 2021.

Year	count
2019	1201
2020	2733
2021	28784

On the right, the 'Aggregate and group data in a dataframe' section provides instructions for running the code and interpreting the results. It includes a note about the 'chain' method and a 'Query data using Spark SQL' section.

Aggregate and group data in a dataframe

1. Add a new code cell to the notebook, and enter the following code in it:

```
Python
productSales = df.select("Item", "Q")
display(productSales)
```

2. Run the code cell you added, and note that the results show the sum of order quantities grouped by product. The **groupBy** method groups the rows by Item, and the subsequent **sum** aggregate function is applied to all of the remaining numeric columns (in this case, Quantity).

3. Add another new code cell to the notebook, and enter the following code in it:

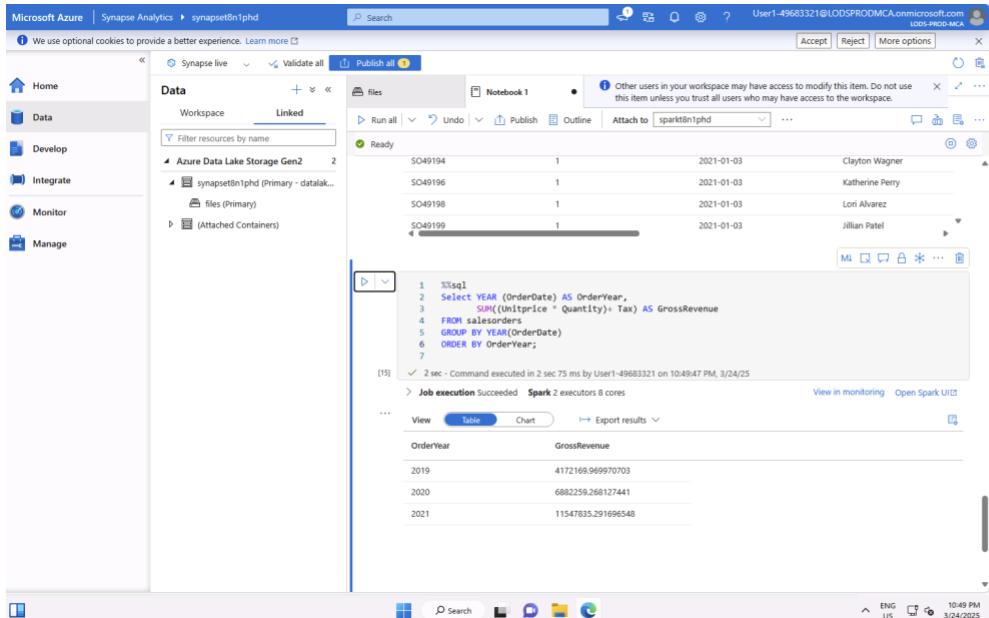
```
Python
yearlySales = df.select(year("OrderDate").alias("Year"), "Q")
display(yearlySales)
```

4. Run the code cell you added, and note that the results show the number of sales orders per year. Note that the **select** method includes a SQL **year** function to extract the year component of the **OrderDate** field, and then an **alias** method is used to assign a column name to the extracted year value. The data is then grouped by the derived **Year** column and the count of rows in each group is calculated before finally the **orderBy** method is used to sort the resulting dataframe.

Query data using Spark SQL

End >

Querying data using Spark SQL:



The screenshot shows the Microsoft Azure Synapse Analytics interface. On the left, the 'Data' section lists 'Azure Data Lake Storage Gen2' and 'synapsestn1phd'. The main area displays a notebook with two code cells. The first cell contains Spark SQL code to select the year of the order date, calculate the gross revenue, and group the results by year. The second cell contains Python code to display the results of the SQL query. The output of the second cell is a table showing the gross revenue for each year from 2019 to 2021.

OrderYear	GrossRevenue
2019	4172169.969970703
2020	6882259.268127441
2021	11547835.291696548

On the right, the 'Run SQL code in a cell' section provides instructions for running the code and interpreting the results. It includes a note about the '%%sql' magic and a 'Note' about Spark SQL documentation.

Run SQL code in a cell

While it's useful to be able to embed SQL statements into a cell containing PySpark code, data analysts often just want to work directly in SQL.

1. Add a new code cell to the notebook, and enter the following code in it:

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

2. Run the cell and review the results. Observe that:

- o The **%%sql** line at the beginning of the cell (called a magic) indicates that the Spark SQL language runtime should be used to run the code in this cell instead of PySpark.
- o The SQL code references the **salesorders** view that you created previously using PySpark.
- o The output from the SQL query is automatically displayed as the result under the cell.

Note: For more information about Spark SQL and dataframes, see the [Spark SQL documentation](#).

End >

Visualizing data with Spark:

The screenshot shows a Microsoft Azure Synapse Analytics notebook interface. The left sidebar contains navigation options: Home, Data, Develop, Integrate, Monitor, and Manage. The main workspace displays a code cell with the following SQL query:

```
1 %sql
2 SELECT * FROM salesorders
```

The query has been executed successfully, as indicated by the status "Job execution Succeeded" and "Spark 2 executors 8 cores". The results are displayed as a horizontal bar chart. The x-axis is labeled "Sum(Quantity)" and ranges from 0 to 100. The y-axis lists various product categories and their quantities. The chart type is set to "Bar chart" in the "View" pane on the right. The "Key" is set to "Item" and the "Values" are set to "Quantity". The "Aggregation" is set to "Sum".

On the right side of the notebook, there is a section titled "View results as a chart" with the following instructions:

1. Add a new code cell to the notebook, and enter the following code in it:
2. Run the code and observe that it returns the data from the **salesorders** view you created previously.
3. In the results section beneath the cell, change the **View** option from **Table** to **Chart**.
4. Use the **View options** button at the top right of the chart to display the options pane for the chart. Then set the options as follows and select **Apply**:
5. Verify that the chart looks similar to this:

Below the instructions, there is a small thumbnail image of the chart. At the bottom right, there is a section titled "Get started with matplotlib" with the following instructions:

1. Add a new code cell to the notebook, and enter the following code in it:

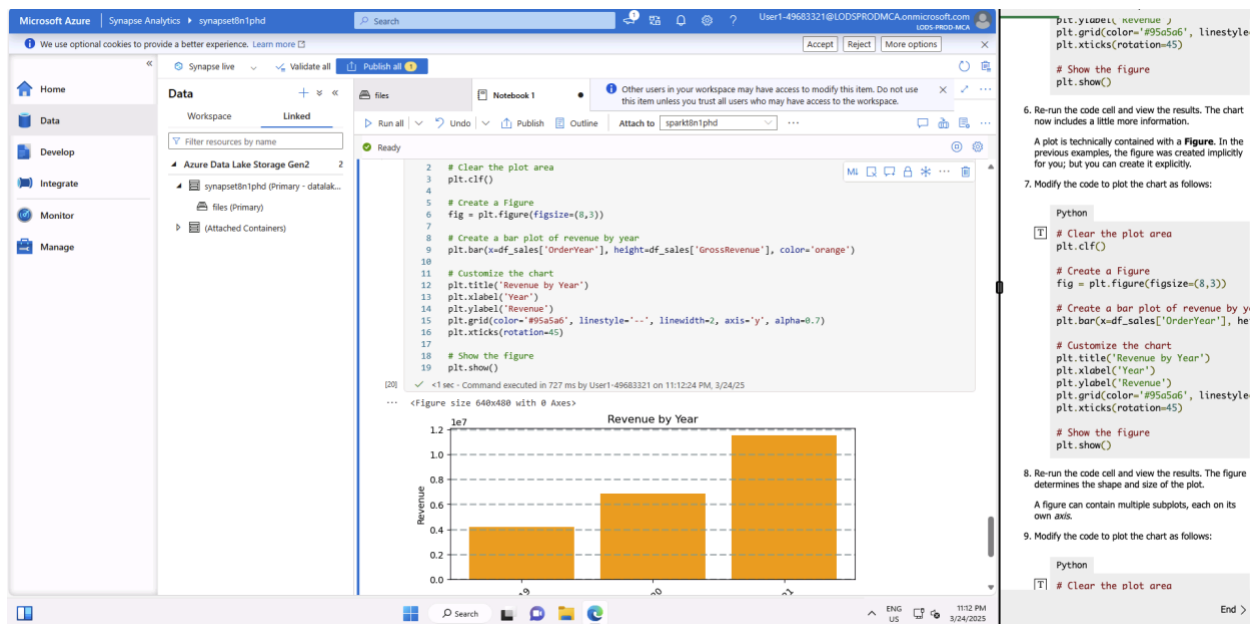
```
Python
1 sqlQuery = "SELECT CAST(YEAR(OrderDate) AS INT), SUM(UnitPrice * Quantity) AS GrossRevenue
2 FROM salesorders"
```

Using Matplotlib library:

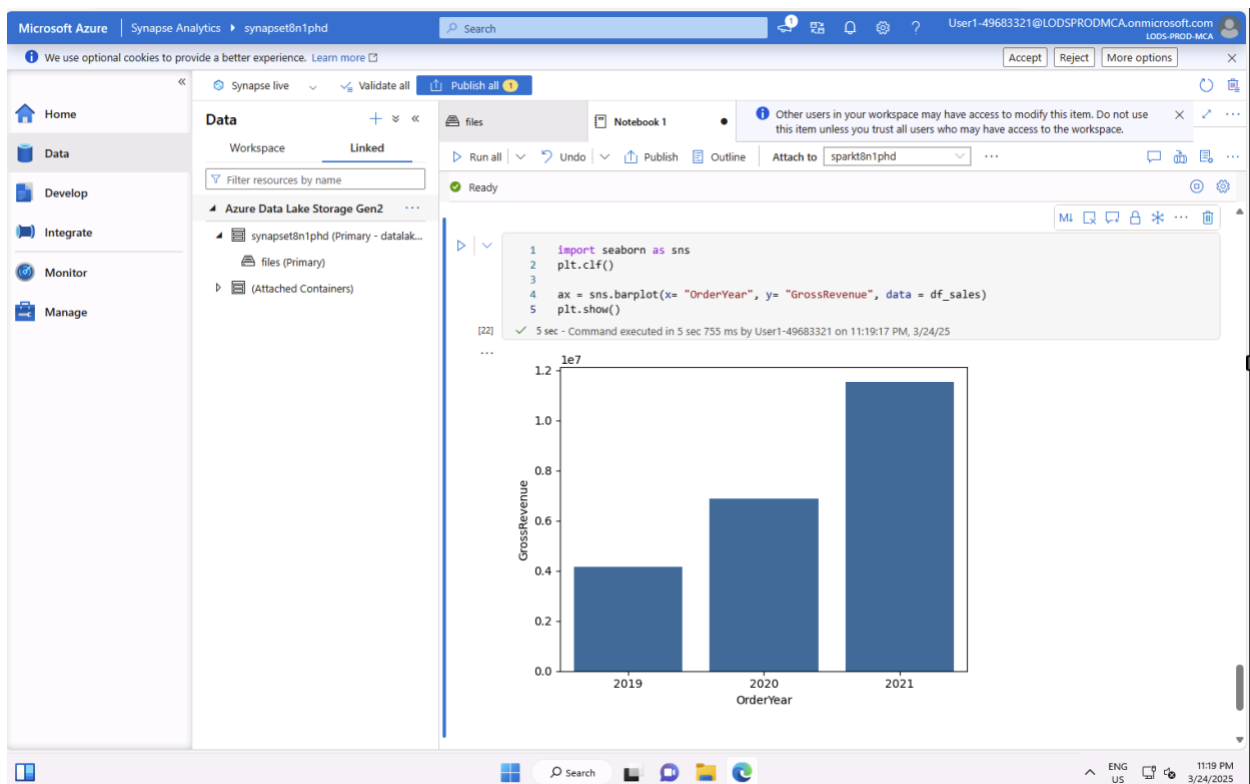
The screenshot shows a Microsoft Azure Synapse Analytics notebook interface. The left sidebar contains navigation options: Home, Data, Develop, Integrate, Monitor, and Manage. The main workspace displays a code cell with the following Python script:

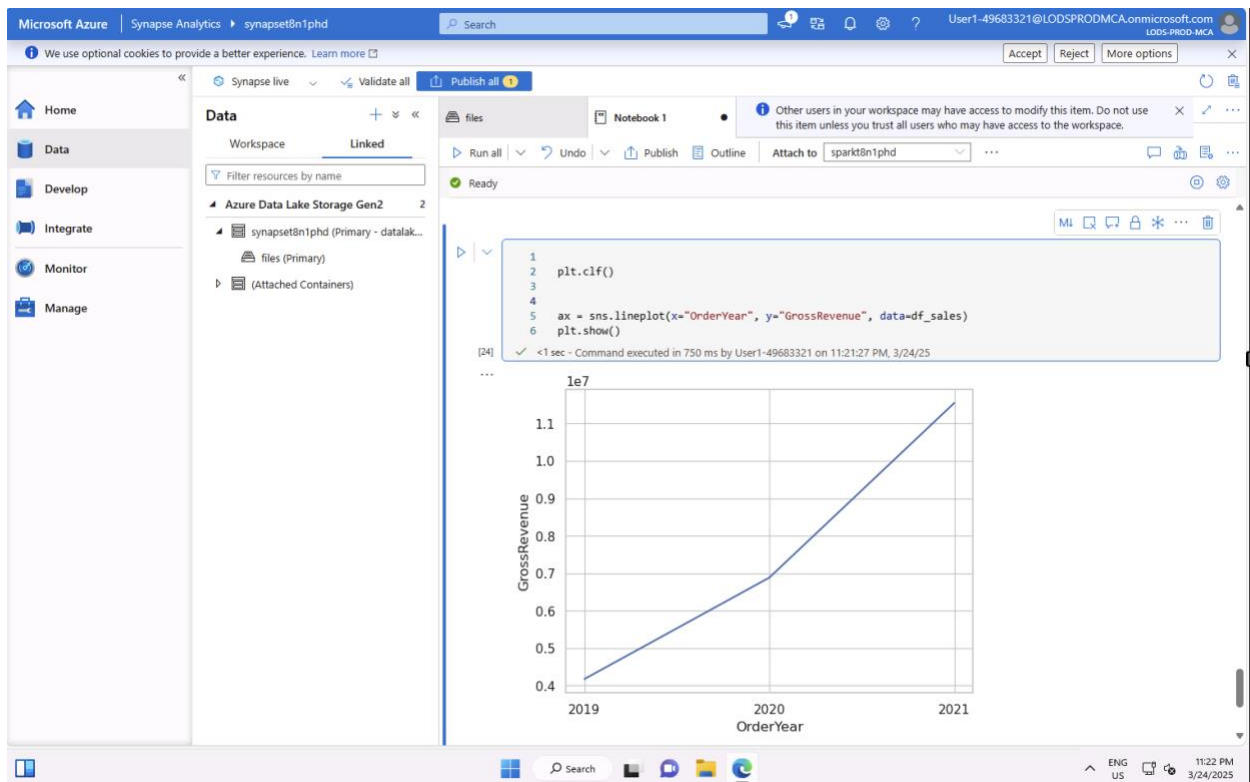
```
1 from matplotlib import pyplot as plt
2
3 # matplotlib requires a Pandas dataframe, not a Spark one
4 df_sales = df_spark.toPandas()
5
6 # Create a bar plot of revenue by year
7 plt.bar(x=df_sales['OrderYear'], height=df_sales['GrossRevenue'])
8
9 # Display the plot
10 plt.show()
```

The script has been executed successfully, as indicated by the status "Job execution Succeeded" and "Spark 2 executors 8 cores". The results are displayed as a vertical bar chart. The x-axis is labeled "OrderYear" and shows the years 2019, 2020, and 2021. The y-axis is labeled "GrossRevenue" and ranges from 0.0 to 1.2, with a multiplier of $1e7$ at the top. The chart shows that revenue increased significantly from 2019 to 2021.



Using the seaborn library:





Deleting Azure resources:

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes 'Home', 'Search resources, services, and docs (G+)', 'Copilot', and user information. The main content area is titled 'dp203-t8n1phd' and shows the 'Overview' tab. A table lists the resources within the group:

Name	Type	Location
datalakest8n1phd	Storage account	South Central US
sparkst8n1phd (synapsest8n1phd/sparkst8n1phd)	Apache Spark pool	South Central US
synapsest8n1phd	Synapse workspace	South Central US

A notification banner at the top right states: 'Deleting resource group dp203-t8n1phd' with a 'Dismiss all' button. Below the table, a 'Switch to Bash' button is visible. The bottom section shows a terminal window with the following output:

```
ContinuationToken
VersionId
LatestVersion
AccessTier
TagCount
Tags
LinklabProperties
Context
Name
Script completed at: 03/25/2025 09:06:38
PS /home/user1-49683321/dp203/Allfiles/Labs/05>
```

On the right side, a 'Notifications' panel shows a message: 'Deleting resource group dp203-t8n1phd' with a 'Dismiss all' button. Below this, a 'Delete Azure resources' section provides instructions:

1. Close the Synapse Studio browser tab and return to the Azure portal.
2. On the Azure portal, on the **Home** page, select **Resource groups**.
3. Select the **dp203-xxxxxxx** resource group for your Synapse Analytics workspace (not the managed resource group), and verify that it contains the Synapse workspace, storage account, and Spark pool for your workspace.
4. At the top of the **Overview** page for your resource group, select **Delete resource group**.
5. Enter the **dp203-xxxxxxx** resource group name to confirm you want to delete it, and select **Delete**.

After a few minutes, your Azure Synapse workspace resource group and the managed workspace resource group associated with it will be deleted.

End the lab
Please be sure to end the lab.

Conclusion:

In this lab, I got hands-on experience with Synapse Spark, learning how to connect to PowerShell and work within Synapse Studio. I explored how to access and manage files in a data lake, load data into a DataFrame, and start a Spark pool. It was interesting to see how data can be aggregated, grouped, and queried using Spark SQL. The best part was visualizing the results with Matplotlib and Seaborn, making complex data easier to understand. I also learned the importance of properly deleting Azure resources to keep things cost-efficient.