

Global Superstore Data Engineering Project Report

Team Contributions

This project was a collaborative effort by our team of three, with each member focusing on specific stages of the data pipeline, from ingestion to visualization, as well as documentation and presentation deliverables. Below are our individual contributions:

- **Nikhila Devi Maddala**

- Sourced the Global Superstore dataset from Kaggle and validated its suitability for the project.
- Configured Azure Data Factory for data ingestion into Azure Data Lake Storage Gen2 (Bronze layer).
- Developed transformation logic in Azure Databricks, including calculating Shipping Delay Days and standardizing fields, up to the Silver layer.
- Drafted the Data Ingestion and Data Transformation sections of the report and corresponding presentation slides.

- **Sai Pujitha Kamatam**

- Set up Azure Synapse Analytics to query transformed data in the Silver layer and created Gold layer views (e.g., `gold.sales`, `gold.customers`).
- Ensured seamless integration between Synapse and Power BI for visualization.
- Documented the Data Analytics section of the report and prepared related presentation visuals for the Gold layer.

- **Likhita Velagapudi**

- Built and published interactive Power BI dashboards to visualize sales trends, returns, and operational insights.
- Drafted the Data Visualization section of the report and prepared the dashboard walkthrough for presentation.
- Managed the GitHub repository for version control and documentation.

GitHub: https://github.com/nikhila272/DAMG7370-Azure-Project/blob/main/Global%20Superstore_Final.xls

Abstract

This project demonstrates the implementation of a cloud-based data pipeline using Microsoft Azure services to process and analyze the Global Superstore dataset, sourced from Kaggle. The pipeline ingests raw retail data, transforms it for analysis, and visualizes insights through interactive dashboards. Azure Data Factory handles ingestion, Azure Data Lake Storage Gen2 organizes data into Bronze, Silver, and Gold layers, Azure Databricks performs transformations, Azure Synapse Analytics enables querying, and Power BI delivers visualizations. The resulting dashboards provide actionable insights into sales performance, return patterns, and operational efficiency, showcasing the potential of cloud technologies for retail analytics.

Dataset Description

Dataset Overview

The Global Superstore dataset, sourced from Kaggle , comprises three CSV files—Orders, Returns, and People—totaling 12.09 MB. Key attributes include:

- **Orders:** Order ID, Order Date, Ship Date, Customer ID, Product ID, Sales, Quantity, Profit, Shipping Cost.
- **Returns:** Order ID, Return Reason, Market, Region.
- **People:** Regional managers mapped to sales regions.
- **Geographical Information:** Region, Market, Country, City.
- **Temporal Data:** Order and shipping dates spanning multiple years.

Justification for Selection

The dataset was selected for its:

1. **Manageable Scale:** At 12.09 MB, it offers a realistic challenge for cloud ingestion and processing.
 2. **Data Variety:** Includes numerical (sales, profit), categorical (region, category), and temporal (dates) data, supporting diverse transformations.
 3. **Retail Relevance:** Enables insights into sales trends, customer behavior, and operational metrics like shipping delays.
 4. **Accessibility:** Publicly available on Kaggle, ideal for automated ingestion via Azure Data Factory.
 5. **Visualization Potential:** Supports rich dashboards, such as sales by region or returns by category, for actionable business insights.
-

Potential Applications

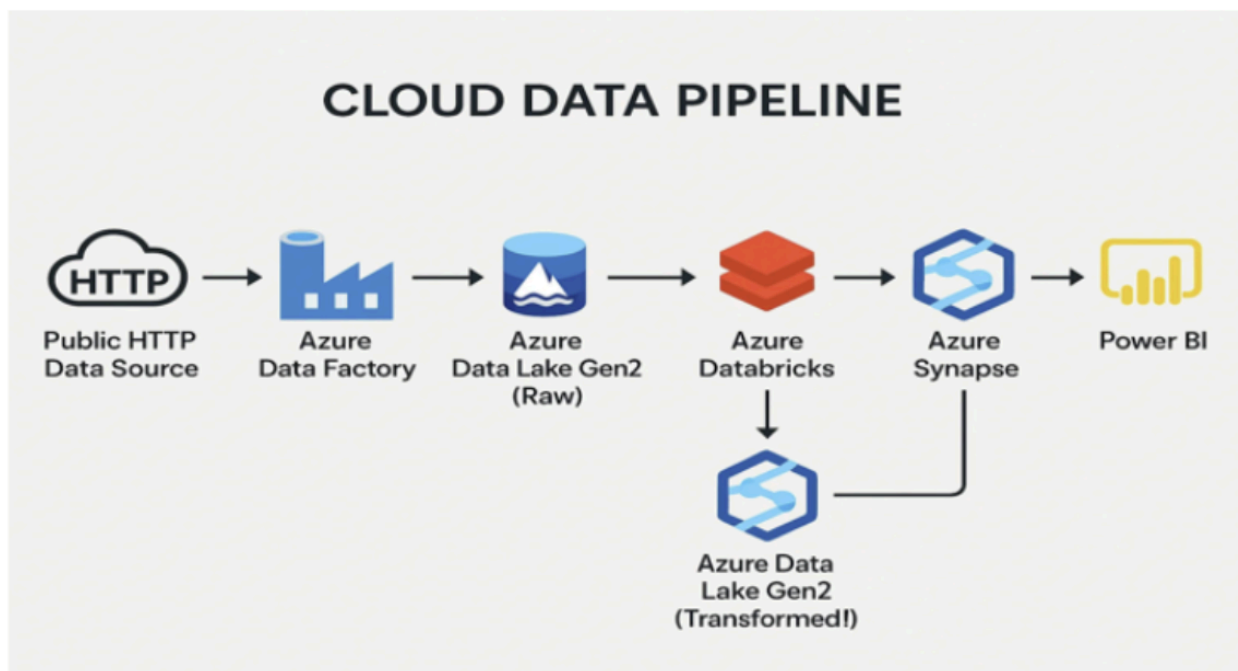
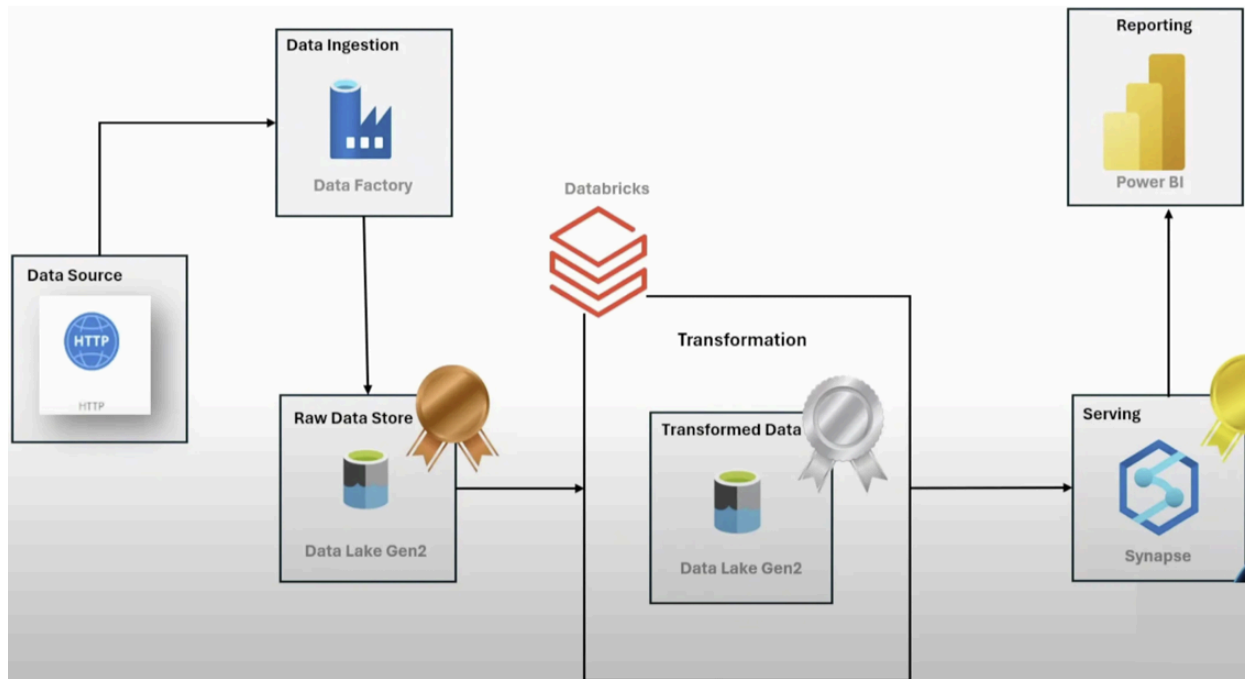
- Analyzing sales trends by region, market, or customer segment to optimize marketing strategies.
 - Identifying high-return products to improve quality or customer satisfaction.
 - Evaluating shipping efficiency through metrics like Shipping Delay Days for better logistics planning.
 - Supporting inventory management by understanding product category performance.
 - Facilitating research into global retail trends and consumer behavior.
-

Architecture and Design Overview

Our objective was to build a scalable, cloud-based data pipeline to process the Global Superstore dataset using Microsoft Azure services. The architecture follows a Bronze-Silver-Gold medallion pattern, ensuring structured data management across ingestion, transformation, analytics, and visualization stages.

Design Components and Flow

1. **Data Source:** Kaggle dataset (Orders, Returns, People CSVs).
2. **Data Ingestion:** Azure Data Factory moves raw data into the Bronze layer of Azure Data Lake Storage Gen2.
3. **Raw Data Storage:** Bronze layer stores unprocessed CSVs in neudamgdata1ake storage account.
4. **Transformation:** Azure Databricks cleans and enriches data, saving it to the Silver layer as Parquet files.
5. **Serving Layer:** Azure Synapse Analytics queries Silver data and creates Gold layer views for analytics.
6. **Reporting:** Power BI connects to Synapse to visualize insights through interactive dashboards.



Architectural Principles

- **Scalability:** Services like Databricks and Synapse scale to handle larger datasets.
- **Modularity:** Bronze, Silver, and Gold layers separate concerns for easier maintenance.
- **Security:** Data encryption in transit and at rest, with role-based access control.
- **Cost-Efficiency:** Serverless components (e.g., Synapse Serverless) minimize costs.
- **Maintainability:** Databricks notebooks and Data Factory pipelines are versioned via GitHub.

Cloud Implementation Using Microsoft Azure

Data Ingestion Using Azure Data Factory

Azure Data Factory (neu-aw-project, East US region) was used to ingest the Global Superstore dataset. The three CSV files were downloaded locally from Kaggle and uploaded to the Bronze container in Azure Data Lake Storage Gen2 (neudamgdatalake) using a static pipeline. A dynamic pipeline was also configured for scalability, enabling iterative file copying for future datasets.

The screenshot displays the Microsoft Azure Data Factory console for a project named 'neu-aw-g6-project'. The left sidebar shows 'Factory Resources' with a list of pipelines, including 'DynamicGitToRaw'. The main canvas shows the 'DynamicGitToRaw' pipeline with a 'Lookup' activity (LookupGit) and a 'ForEach' loop containing a 'DynamicCopy' activity. The 'Output' tab at the bottom shows the pipeline run ID '2f0d9c0d-5335-4d18-80a9-b38f9ee2f524' and a table of activity results.

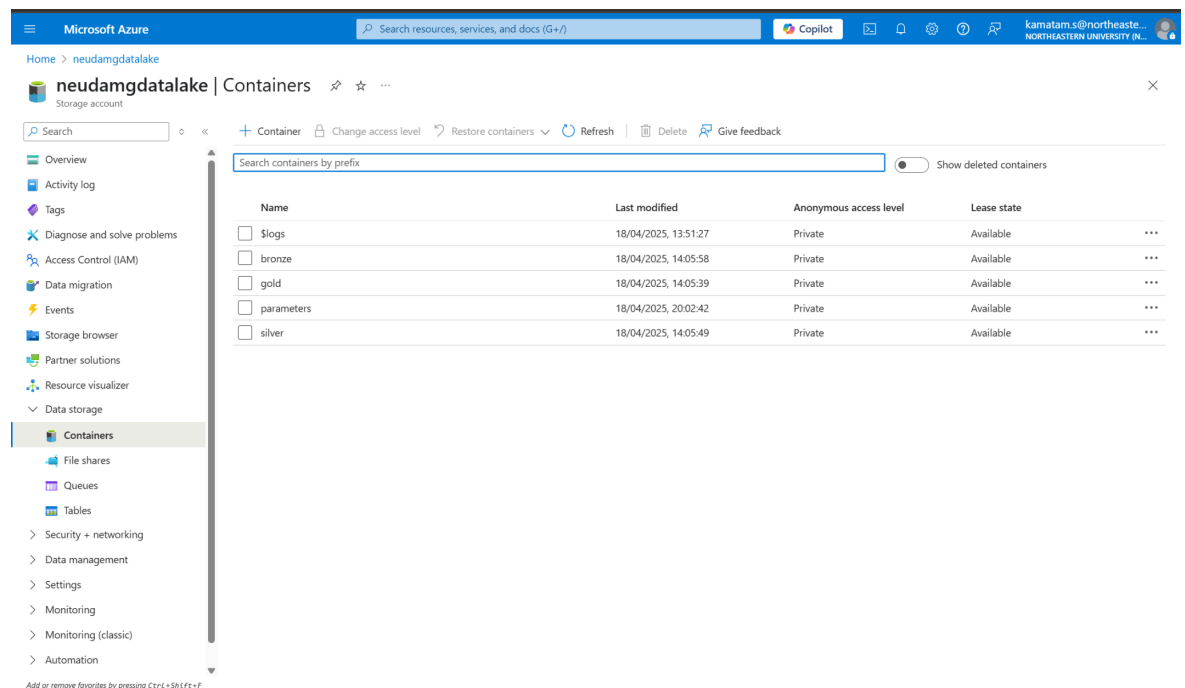
Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime
DynamicCopy	✓ Succeeded	Copy data	4/23/2025, 8:27:23 PM	13s	AutoResolveIntegration
DynamicCopy	✓ Succeeded	Copy data	4/23/2025, 8:27:08 PM	13s	AutoResolveIntegration

Storage in Azure Data Lake Gen2

The data lake is structured into three containers:

- **Bronze:** Raw CSVs (Orders, Returns, People).
- **Silver:** Transformed Parquet files after Databricks processing.
- **Gold:** Analytics-ready views created by Synapse.

This setup ensures progressive data refinement, with `neudamgdatalake` providing modular access at each stage.



Data Transformation Using Azure Databricks

Azure Databricks processed data from the Bronze layer using Spark DataFrames. Transformations included:

- **Orders:** Added Order Month, Order Year from Order Date; calculated Shipping Delay Days (Ship Date - Order Date).
 - **Returns:** Standardized Return Reason for consistency.
 - **People:** Unified Region names (e.g., "US-East" to "East US").
- The transformed data was saved as Parquet files in the Silver container. A `Py4JJavaError` during saving was resolved by updating the SAS token configuration for the Silver container.

Microsoft Azure databricks

silver_layer Python

File Edit View Run Help Last edit was 3 days ago

Run all Terminated Schedule Share

Workspace

3 days ago (1h)

```
from pyspark.sql.functions import *
from pyspark.sql.types import *
```

3 days ago (8h)

```
storage_account_name = "messagingdata4a"
sas_token = "spnresid=mskdk1202-86-21704-86-8128&sv=2025-04-21T12:06:8128&pr=https&sv=2024-11-04&sr=c&sig=FzEYDWHWTEt8uA3Cm8DhWu8k2u2C2Bh7Z0M85D0"
container_name = "bronze"

# Unmount the existing mount point if it exists
try:
    dbutils.fs.unmount(f"/mnt/{container_name}")
except:
    pass

# Set up configuration
dbutils.fs.mount(
    source=f"sas://{container_name}@{storage_account_name}.blob.core.windows.net",
    mount_point=f"/mnt/{container_name}",
    extra_configs={f"sas_auth": f"sas_token", f"storage_account_name": f"{storage_account_name}", f"blob.core.windows.net": f"sas_token"}
)

# Verify the mount
display(dbutils.fs.ls(f"/mnt/{container_name}"))
```

0 Spark Jobs

/mnt/bronze has been unmounted.

Microsoft Azure databricks

silver_layer Python

File Edit View Run Help Last edit was 3 days ago

Run all Terminated Schedule Share

Workspace

3 days ago (1h)

```
df_ord = spark.read.csv("/mnt/bronze/orders", header=True, inferSchema=True)
df_ppl = spark.read.csv("/mnt/bronze/people", header=True, inferSchema=True)
df_ret = spark.read.csv("/mnt/bronze/returns", header=True, inferSchema=True)
```

0 Spark Jobs

- df_ord: pyspark.sql.dataframe.DataFrame = [Row ID: Integer, Order ID: string ... 22 more fields]
- df_ppl: pyspark.sql.dataframe.DataFrame = [Person string, Region: string]
- df_ret: pyspark.sql.dataframe.DataFrame = [Returned: string, Order ID: string ... 1 more field]

3 days ago (8h)

```
storage_account_name = "messagingdata4a"
sas_token = "spnresid=mskdk1202-86-21704-12:2528&sv=2025-04-21T12:12:2528&pr=https&sv=2024-11-04&sr=c&sig=V8f011PhyQ0B9rykzWk7p4uVQC1Q1P1C38R3D0"
container_name = "silver"
```

Unmount the existing mount point if it exists

```
try:
    dbutils.fs.unmount(f"/mnt/{container_name}")
except:
    pass
```

Set up configuration

```
dbutils.fs.mount(
    source=f"sas://{container_name}@{storage_account_name}.blob.core.windows.net",
    mount_point=f"/mnt/{container_name}")
```

Microsoft Azure databricks

silver_layer Python

File Edit View Run Help Last edit was 3 days ago

Run all Terminated Schedule Share

Workspace

3 days ago (1h)

```
# Add Order Month and Order Year, clean customer name
from pyspark.sql.functions import month, year, col, split

df_ord = df_ord.withColumn("Order Month", month(col("order date"))).withColumn("Order Year", year(col("order date"))).withColumn("customer name", split(col("customer name"), ',')[0])
```

df_ord: pyspark.sql.dataframe.DataFrame = [Row ID: Integer, Order ID: string ... 24 more fields]

3 days ago (1h)

```
df_ord.display()
```

0 Spark Jobs

Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	customer name	Cons	
1	32208	CA-2012-124891	2012-07-31	2012-07-31	Same Day	BA-19495	Rick Hansen	Cons
2	20341	IN-2013-37878	2013-02-05	2013-02-07	Second Class	JB-16210	Justin Ritter	Corp
3	20390	IN-2013-12486	2013-03-07	2013-10-18	First Class	CR-12792	Craig Rafter	Cons
4	13324	ES-2013-157942	2013-01-28	2013-01-30	First Class	BM-16375	Katherine Murray	Ham
5	47221	SG-2013-4320	2013-11-05	2013-11-06	Same Day	BA-19495	Rick Hansen	Cons
6	22732	IN-2013-42360	2013-06-28	2013-07-01	Second Class	BM-15555	Jim Michum	Corp
7	30370	IN-2011-41826	2011-11-07	2011-11-09	First Class	TS-21340	Toby Swindell	Cons
8	31192	IN-2012-80369	2012-04-14	2012-04-16	Standard Class	MB-16085	Mike Brown	Cons

Microsoft Azure databricks

silver_layer Python

File Edit View Run Help Last edit was 3 days ago

Run all Terminated Schedule Share

Workspace

3 days ago (8h)

```
# Save to silver layer as Parquet
df_ord.write.format("parquet").mode("append").option("path", "/mnt/silver/GlobaL_Superstore_Orders").save()
```

0 Spark Jobs

3 days ago (1h)

```
df_ppl = df_ppl.withColumnRenamed("Person", "PersonName").withColumnRenamed("Region", "PersonRegion").withColumn("PersonRegion", upper(col("PersonRegion"))).withColumn("PersonID", monotonically_increasing_id())
```

df_ppl: pyspark.sql.dataframe.DataFrame = [PersonName: string, PersonRegion: string ... 1 more field]

3 days ago (1h)

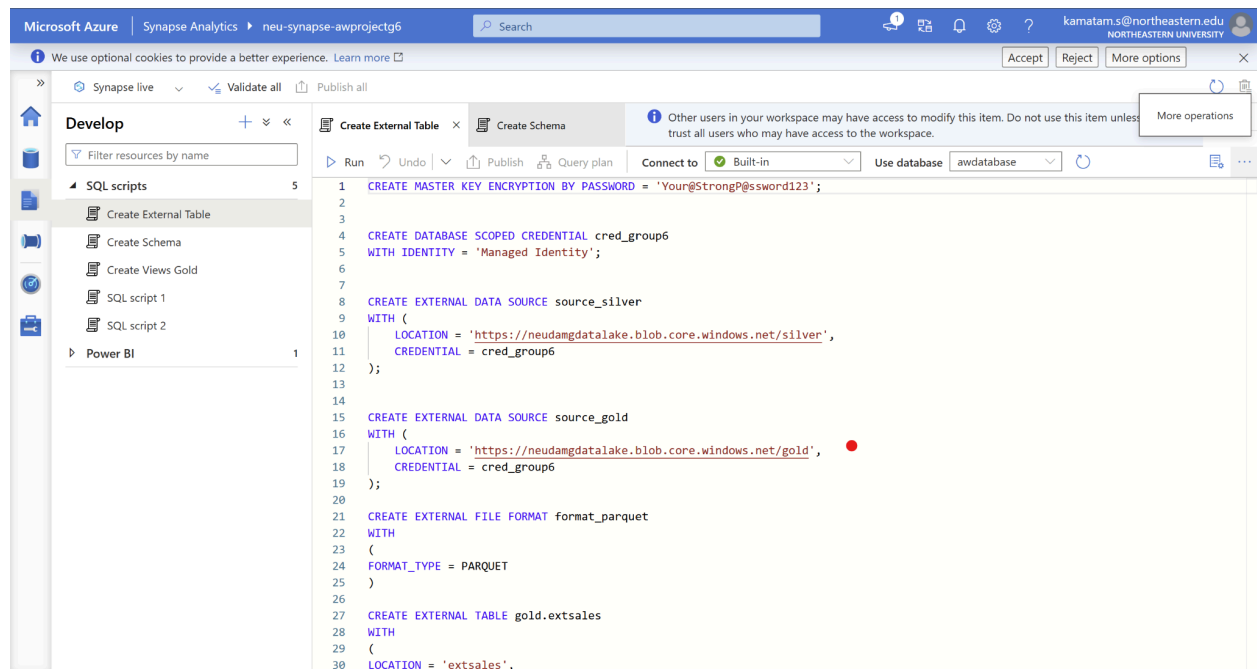
```
df_ppl.display()
```

0 Spark Jobs

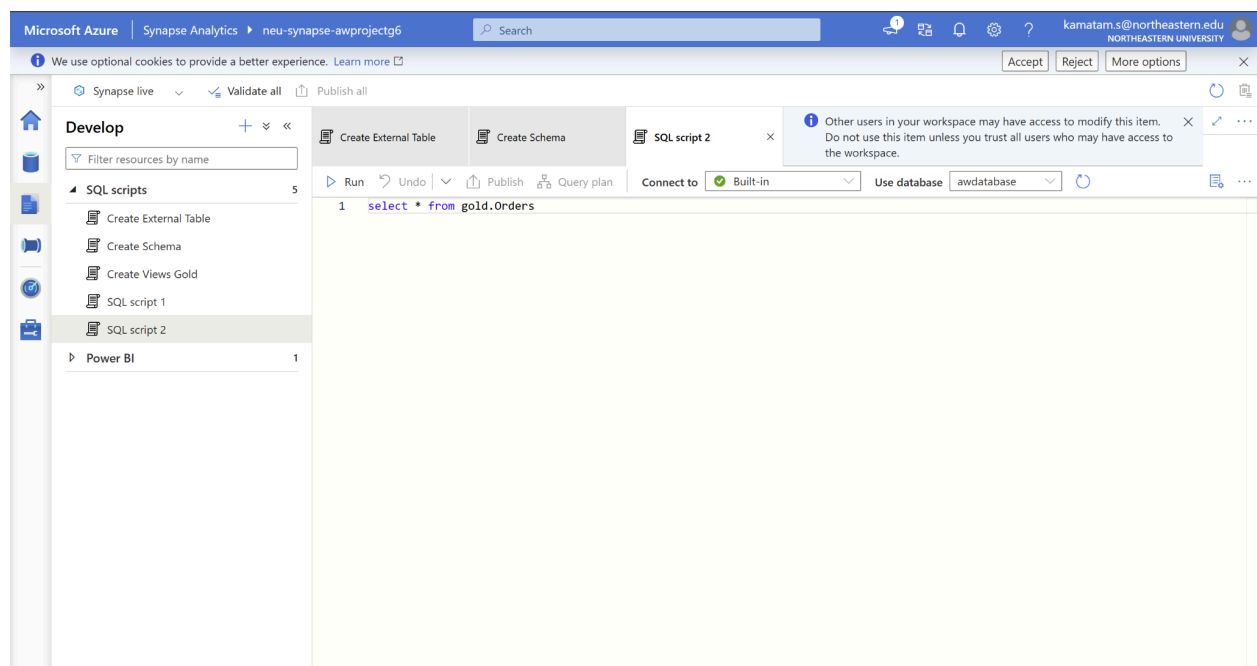
PersonName	PersonRegion	PersonID
Anna Anderson	CENTRAL	0
Chuck Miller	SOUTH	1

Serving Data with Azure Synapse Analytics

Azure Synapse Analytics queried the Silver layer to create Gold layer views, such as `gold.sales`, `gold.customers`, `gold.products`, and `gold.returns`. These views aggregated data for efficient querying, enabling questions like “Which regions have the highest sales?” or “What products have the most returns?” External tables over Parquet files ensured seamless integration with Power BI.



```
1 CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'Your@StrongP@ssword123';
2
3
4 CREATE DATABASE SCOPED CREDENTIAL cred_group6
5 WITH IDENTITY = 'Managed Identity';
6
7
8 CREATE EXTERNAL DATA SOURCE source_silver
9 WITH (
10     LOCATION = 'https://neudamdatalake.blob.core.windows.net/silver',
11     CREDENTIAL = cred_group6
12 );
13
14
15 CREATE EXTERNAL DATA SOURCE source_gold
16 WITH (
17     LOCATION = 'https://neudamdatalake.blob.core.windows.net/gold',
18     CREDENTIAL = cred_group6
19 );
20
21 CREATE EXTERNAL FILE FORMAT format_parquet
22 WITH
23 (
24     FORMAT_TYPE = PARQUET
25 )
26
27 CREATE EXTERNAL TABLE gold.extsales
28 WITH
29 (
30     LOCATION = 'extsales',
```



```
1 select * from gold.Orders
```

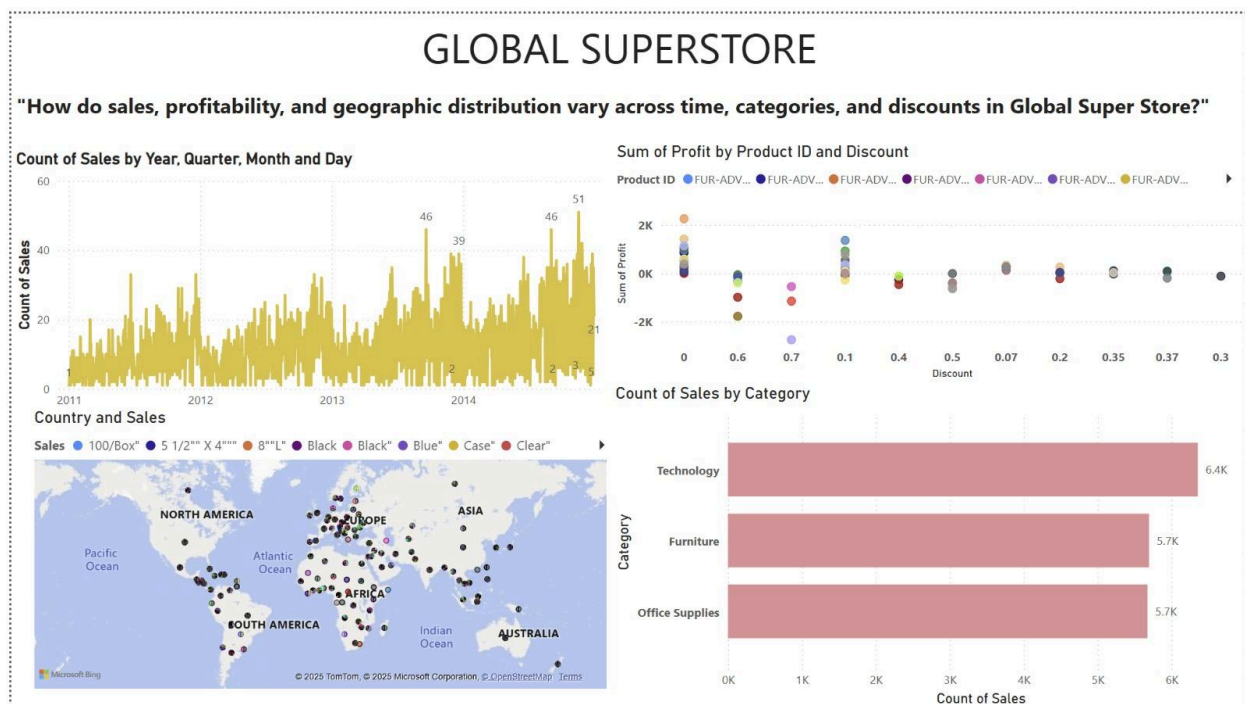

Power BI Visualizations and Insights

Power BI connected to Synapse via DirectQuery to create dashboards:

Dashboard

Global Superstore Sales Overview

- **KPIs:** Total Sales, Total Profit, Order Count.
- **Visuals:**
 - Bar Chart: Sales by Region (e.g., North America, Europe).
 - Line Chart: Sales Trends by Year (spikes in peak seasons).
 - Treemap: Sales by Product Category (e.g., Technology, Furniture).
- **Slicers:** Filter by Year, Region, Customer Segment.
- **Insights:** North America leads in sales; Technology products have the highest profit margins.
- **Returns and Operational Insights**
- **KPIs:** Total Returns, Average Shipping Delay Days.
- **Visuals:**
 - Bar Chart: Returns by Product Category (e.g., Office Supplies highest).
 - Line Chart: Shipping Delay Days by Month (seasonal delays in Q4).
 - Map: Returns by Region (e.g., high returns in Europe).
- **Slicers:** Filter by Market, Return Reason, Date Range.
- **Insights:** Office Supplies have the most returns; shipping delays peak in December.



Key Insights

1. **Sales Trends:** Sales peak in Q4 due to holiday seasons, with Technology products driving profits.
 2. **Returns Patterns:** High return rates in Office Supplies suggest quality or customer expectation issues.
 3. **Operational Efficiency:** Shipping delays in Q4 highlight the need for better logistics planning.
 4. **Regional Insights:** Europe shows high sales but also high returns, indicating potential market-specific challenges.
-

Conclusion

This project successfully implemented a scalable, cloud-based data pipeline for the Global Superstore dataset using Microsoft Azure services. From ingestion with Azure Data Factory to visualization in Power BI, each stage was optimized for efficiency and modularity. The medallion architecture ensured structured data management, while transformations like Shipping Delay Days and Gold layer views enabled deep retail analytics. The Power BI dashboards provided actionable insights into sales, returns, and operational efficiency, demonstrating the value of cloud technologies for retail decision-making. This pipeline serves as a practical example of leveraging Azure for data-driven business outcomes.

References

- Kaggle. (n.d.). Global Superstore Dataset.
<https://www.kaggle.com/datasets/apoorvaappz/global-super-store-dataset>
- Microsoft Azure. (n.d.). Azure Data Factory Documentation.
<https://learn.microsoft.com/en-us/azure/data-factory/>
- Microsoft Azure. (n.d.). Azure Databricks Documentation.
<https://learn.microsoft.com/en-us/azure/databricks/>
- Microsoft Azure. (n.d.). Azure Synapse Analytics Documentation.
<https://learn.microsoft.com/en-us/azure/synapse-analytics/>
- Microsoft Power BI. (n.d.). Power BI Documentation.
<https://learn.microsoft.com/en-us/power-bi/>