

A complete End-to-End Azure Data Engineering project

Overview:

This project demonstrates a complete end-to-end data engineering pipeline using Azure cloud services with e-commerce data. The solution focuses on data ingestion, transformation, storage, and analytics using modern cloud-based tools.

Data used

E-Commerce Data Engineering Pipeline using Azure

Azure services used

1. Azure Data Lake Gen2 - For storing raw and processed data
2. Azure Databricks (Spark) - For data transformation and processing using PySpark
3. Azure Synapse Analytics - For querying and analysing the processed data
4. Azure Data factory – For ETL process and pipelines
5. Azure key vault - to securely store and manage secrets and certificates
6. Azure Entra ID - for securing and managing user identities and access to resources
7. MySQL and MongoDB – Databases

Project Objectives:

- Ingest raw e-commerce data into a data lake.
- Clean and transform data using PySpark in Azure Databricks.
- Store the cleaned data in structured format (Parquet/Delta) in ADLS.
- Analyze the processed data using Synapse SQL.

Pre-requisites:

Azure free-trial/paid subscription

<https://azure.microsoft.com/en-us/pricing/purchase-options/azure-account>

Service deployment documents:

<https://learn.microsoft.com/en-us/azure/data-factory/quickstart-create-data-factory>

<https://learn.microsoft.com/en-us/azure/databricks/getting-started/>

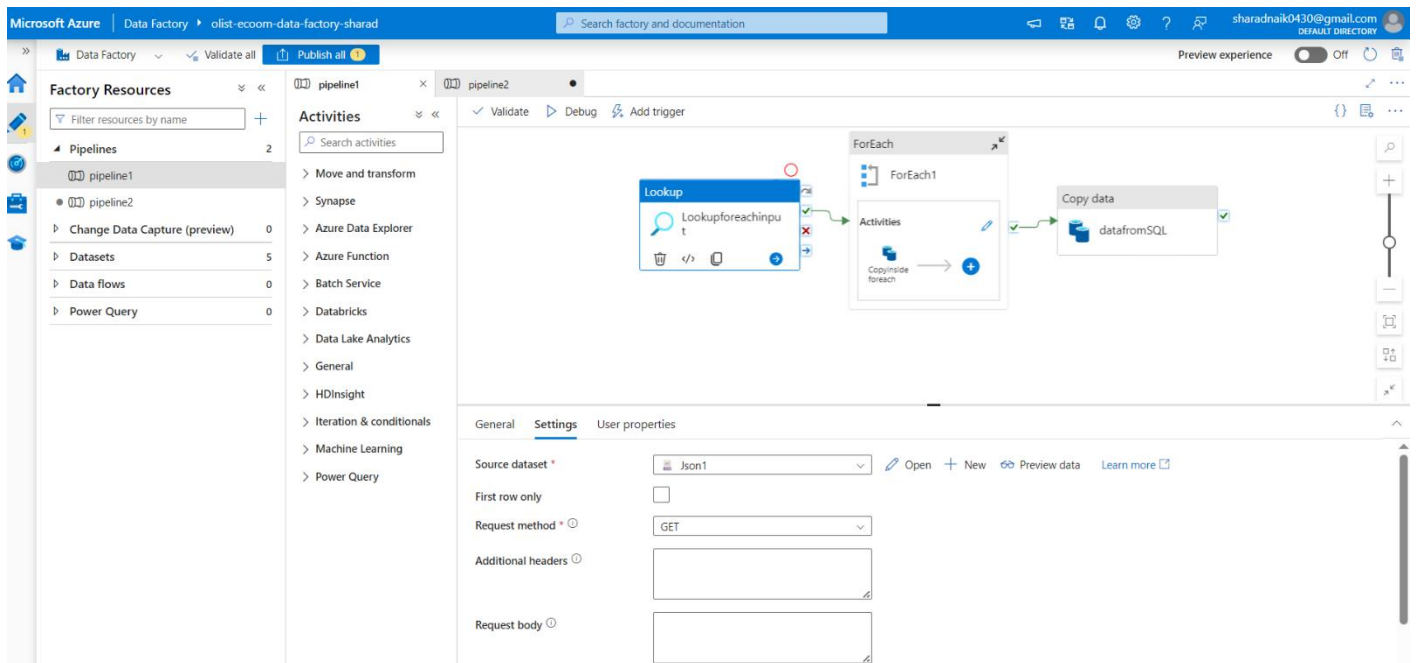
<https://learn.microsoft.com/en-us/azure/storage/blobs/create-data-lake-storage-account>

<https://learn.microsoft.com/en-us/azure/synapse-analytics/get-started-create-workspace>

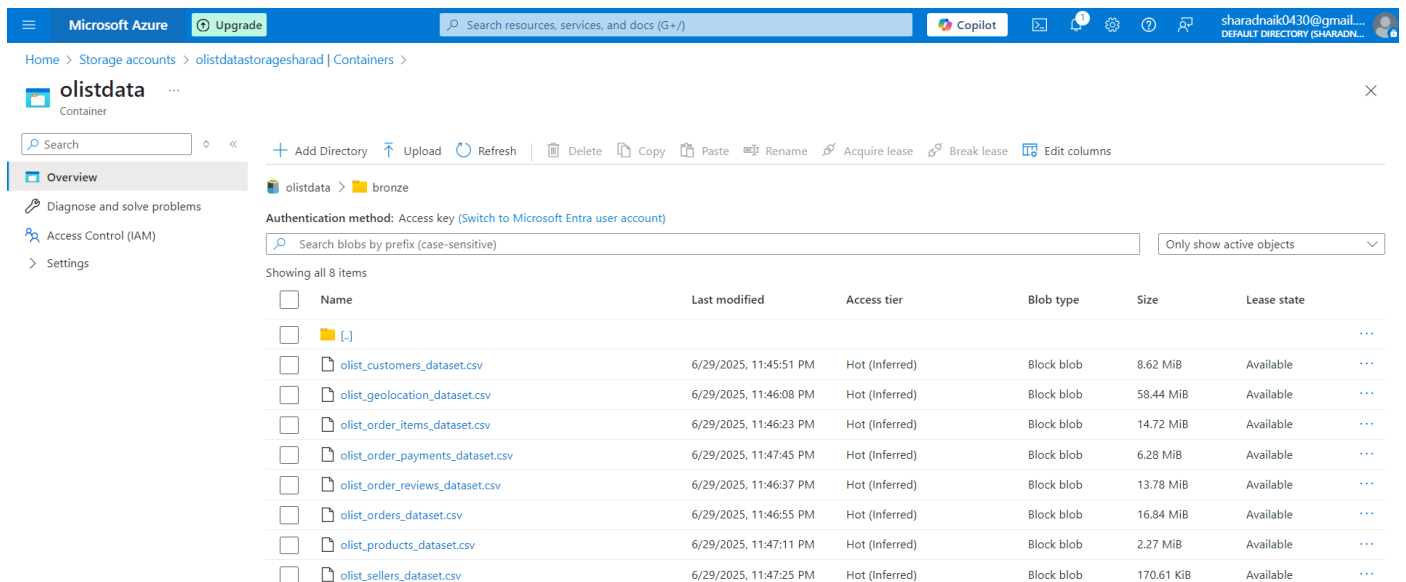
Workflow Overview:

Step 1: Data Extraction (Bronze Layer)

- Extracted raw data from a MySQL database using Azure Data Factory.
- Implemented parameterized pipelines to support dynamic table extraction.

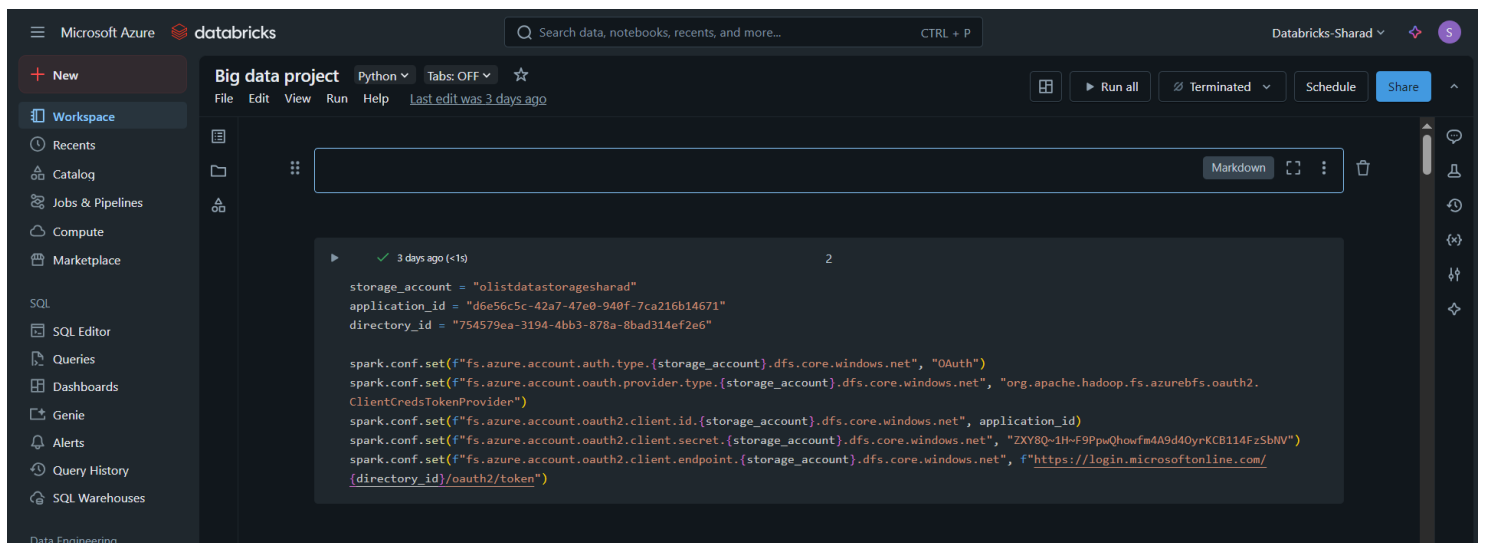


- Stored the raw data in the Bronze layer of ADLS (/bronze folder).

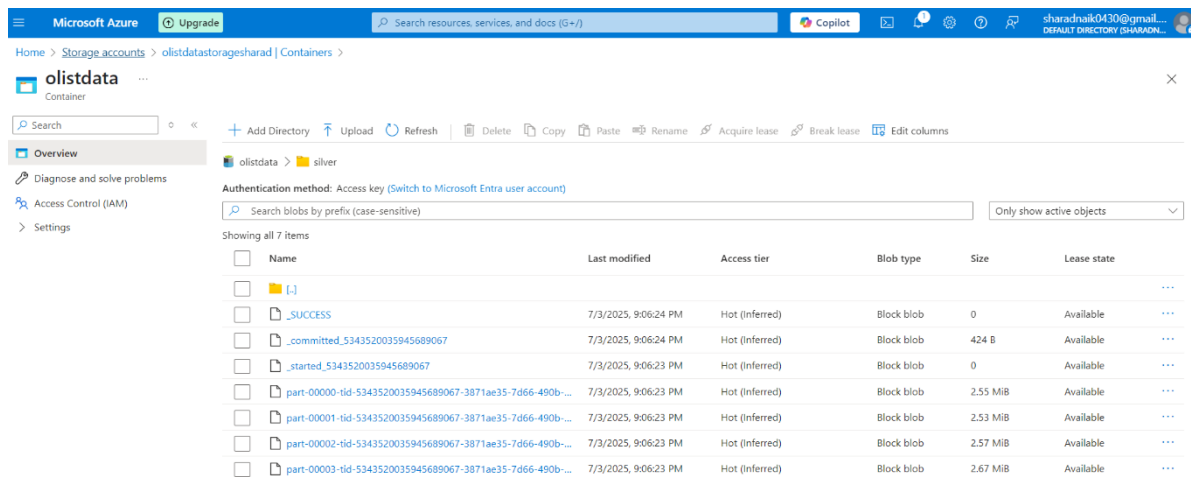


Step 2: Data Cleaning & Transformation (Silver Layer)

- Connected Azure Databricks to ADLS Gen2 using service principal.



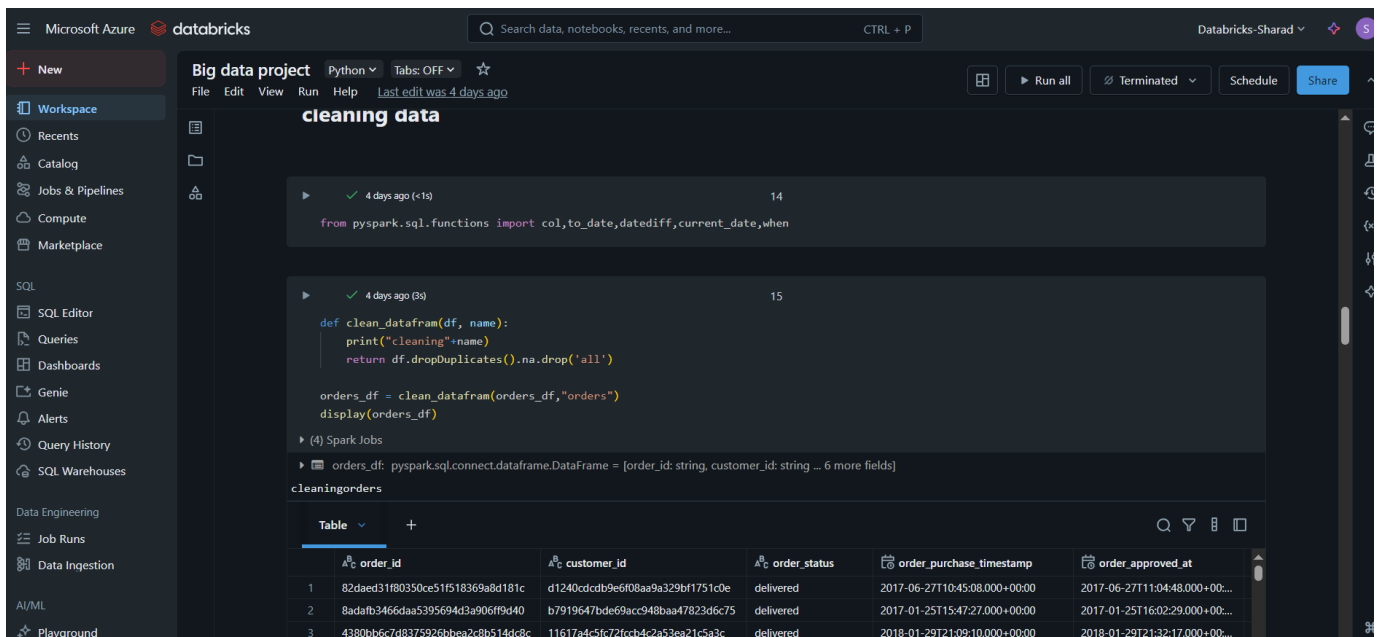
- Loaded data from the bronze folder for cleaning and transformation using PySpark.



- Stored the processed data into the silver layer of ADLS (/silver folder).

Step 3: Enriching Data with MongoDB

- Imported additional customer and product metadata from MongoDB into Databricks.
- Performed joins and enrichments with MySQL data.
- Saved the cleaned and enriched data into the silver folder.



The screenshot shows the Databricks interface for a notebook titled "cleaning data". The notebook is written in Python and includes the following code:

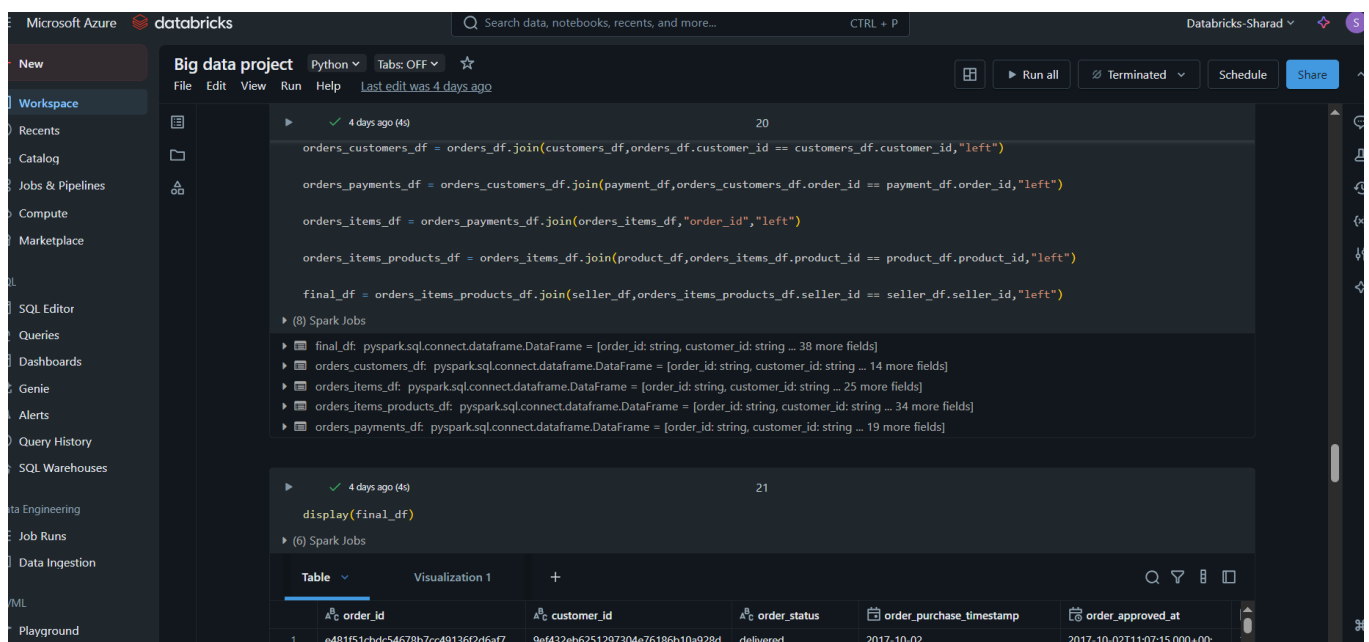
```
from pyspark.sql.functions import col, to_date, datediff, current_date, when
```

```
def clean_dataframe(df, name):  
    print("cleaning "+name)  
    return df.dropDuplicates().na.drop('all')
```

```
orders_df = clean_dataframe(orders_df, "orders")  
display(orders_df)
```

The output of the notebook shows a table with 3 rows of order data:

order_id	customer_id	order_status	order_purchase_timestamp	order_approved_at
82daed31f80350ce51f518369a8d181c	d1240cdcd9e6f08aa9a329bf1751c0e	delivered	2017-06-27T10:45:08.000+00:00	2017-06-27T11:04:48.000+00:00
8adafb3466daa5395694d3a906f9d40	b7919647bde69acc948baa47823d6c75	delivered	2017-01-25T15:47:27.000+00:00	2017-01-25T16:02:29.000+00:00
4380bb6c7d8375926bba2c8b514dc8c	11617a4c5fc72fccb4c2a53ea21c5a3c	delivered	2018-01-29T21:09:10.000+00:00	2018-01-29T21:32:17.000+00:00



The screenshot shows the continuation of the Databricks notebook. The notebook contains the following code:

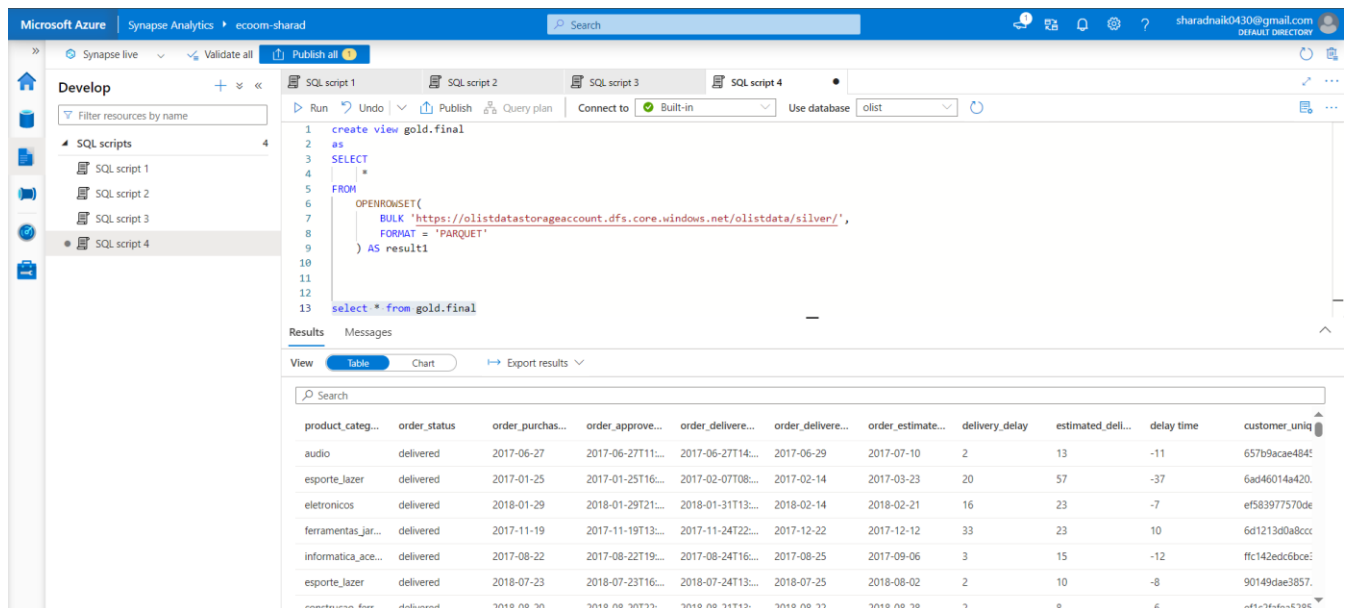
```
orders_customers_df = orders_df.join(customers_df, orders_df.customer_id == customers_df.customer_id, "left")  
orders_payments_df = orders_customers_df.join(payment_df, orders_customers_df.order_id == payment_df.order_id, "left")  
orders_items_df = orders_payments_df.join(orders_items_df, "order_id", "left")  
orders_items_products_df = orders_items_df.join(product_df, orders_items_df.product_id == product_df.product_id, "left")  
final_df = orders_items_products_df.join(seller_df, orders_items_products_df.seller_id == seller_df.seller_id, "left")  
display(final_df)
```

The output of the notebook shows a table with 1 row of order data:

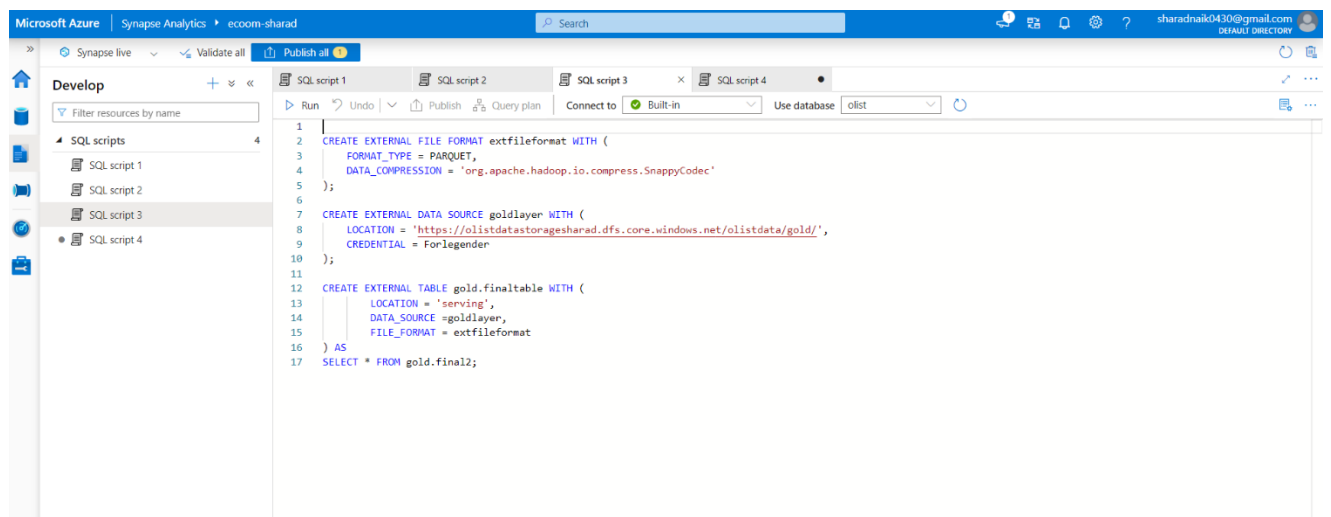
order_id	customer_id	order_status	order_purchase_timestamp	order_approved_at
e481f51cbdc54678b7cc49136f2d6af7	9ef432eb6251297304e76186b10a928d	delivered	2017-10-02	2017-10-02T11:07:15.000+00:00

Step 4: Data Curation & Reporting (Gold Layer)

- Ingested the silver layer data into Azure Synapse Analytics for final curation.
- Created Synapse views and SQL scripts to generate metrics.
- Used CETAS to stored curated datasets back into the Gold layer in ADLS (/gold folder).



- CETAS script to store data to ADLS



Outcomes:

- Demonstrated a real-time batch data processing pipeline using industry-standard practices.
- Implemented secure and scalable connections between services.
- Achieved structured data layers that support reporting and analytics.