WORLD EXPORTS & GLOBAL TRADE PERFORMANCE ANALYTICS PLATFORM

Technologies Used:- Azure Databricks | Apache Airflow | Spark | Power BI

- **PROJECT OVERVIEW**

This project implements an end-to-end data engineering pipeline to analyze Global Trade Export Performance using the Medallion Architecture (Bronze → Silver → Gold) on Azure Databricks, orchestrated using Apache Airflow, and visualized in Power BI.

The pipeline ingests export datasets, applies validation and cleaning, builds curated analytical datasets, and generates insights such as:

-> Top performing exporting countries
-> Product-wise export performance
-> Regional trade analytics
-> Year-over-year growth trends
-> Emerging market identification

🌍 Business Problem

International trade organizations and policy bodies collect massive amounts of export data across:

- Countries
- Regions
- Product categories

However, they face major challenges:

- Raw export data scattered across multiple files
- Manual consolidation causing inconsistencies & errors
- No centralized system for analyzing global trade trends
- Difficulty identifying high-performing & declining markets
- Limited visibility for policy makers to make data-driven decisions

- **OBJECTIVES**

The objective of this project is to build an End-to-End Global Trade Analytics Platform that can support the following:

-> Automates ingestion and processing of global export datasets

->Cleans, standardizes, and validates trade data

-> Performs advanced analytics using Pandas & PySpark

-> Orchestrates ETL workflows using Apache Airflow

-> Generates meaningful insights & visuals using Power BI Dashboards

-> Supports policy makers, economists, and analysts in decision making

- **TECHNOLOGY STACK**

1. Data Processing

Python- Used as the primary programming language for implementing data processing logic, handling datasets, and building ETL workflows due to its simplicity, flexibility, and strong ecosystem.

Pandas & NumPy- Pandas is used for data cleaning, transformation, and analysis, while NumPy provides high-performance numerical computing support. Together, they help in handling structured datasets efficiently.

2. Big Data Analytics

Databricks- A cloud-based unified analytics platform used to process large-scale datasets, build pipelines, and run distributed computing workloads. It enables collaboration and simplifies big data processing.

PySpark- The Python API for Apache Spark, used to perform large-scale data processing, transformations, and analytics on massive datasets in a distributed environment.

3. Workflow Orchestration

Apache Airflow- Used to schedule, automate, and manage data pipelines. It ensures tasks run in sequence, handles retries, monitors workflows, and maintains pipeline reliability.

4. Data Visualization

Power BI- Used to build interactive dashboards and visual analytics that help stakeholders explore trade insights, monitor KPIs, and make data-driven decisions.

5. Others

Git & GitHub- Used for version control and collaboration, ensuring code management, tracking changes, and maintaining project history.

Databricks Notebooks- Interactive notebook environments used for data exploration, development, testing, and visualization of intermediate outputs during the data pipeline development process.

- **ARCHITECTURE OVERVIEW**

This project follows Medallion Architecture:

Raw Source Data

⬇️

BRONZE ➡️ Raw Ingestion Layer

⬇️

SILVER ➡️ Cleaned & Standardized Layer

⬇️

GOLD ➡️ Business & Analytics Layer

⬇️

POWER BI Dashboards

Workflow is scheduled and controlled using Apache Airflow, and Unity Catalog lineage tracks data flow from Bronze → Silver → Gold.

- **DATASET DESCRIPTION**

1️⃣Global Export Fact Dataset

The schema of this dataset is:

Country_Name
Country_Code
Year
Month
Product_Code
Product_Name
Product_Category
Region
Export_Value_USD
Export_Units

2️⃣Country Reference Dataset

The schema of this dataset is:

Country_Code
Country_Name
Region

3️⃣Product Reference Dataset

The schema of this dataset is:

Product_Code
Product_Name
Product_Category

The two reference datasets are used so that they act as trusted lookup that ensure the fact data is consistent, accurate, and standardized.
The datasets are initially stored in the unity catalog of databricks. It uses built-in storage "Volumes" to store the datasets. The URL path looks something like "/Volumes/capstone/default/datasets/".

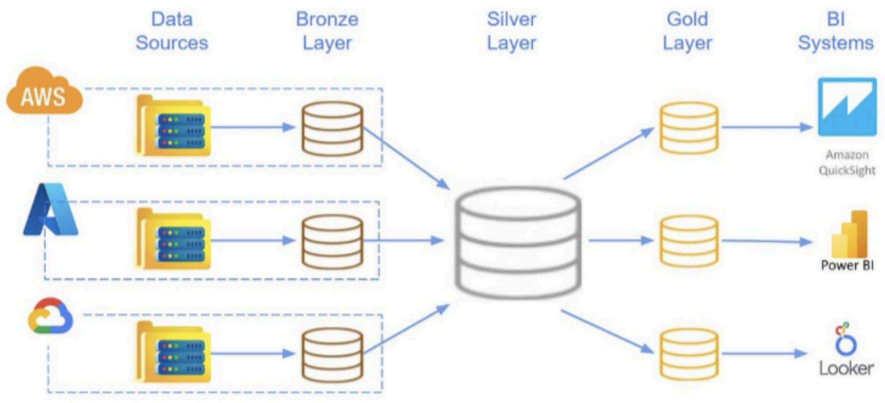
- **OVERALL ARCHITECTURE**

The project follows the Medallion Architecture, a layered data design pattern widely used in modern data engineering to progressively improve data quality and analytical value. It consists of three structured layers: Bronze, Silver, and Gold.

-> Bronze Layer: Stores raw ingested data exactly as received from source systems, preserving original integrity.

-> Silver Layer: Cleans, standardizes, and enriches data to make it structured, reliable, and analytics-ready.

-> Gold Layer: Delivers highly curated, business-focused datasets optimized for reporting and advanced analytics.



- **DATABRICKS PLATFORM METHODOLOGY**

Databricks serves as the core data engineering and analytics platform for this project. Built on Apache Spark, it provides powerful distributed computing capabilities to handle large datasets efficiently. Databricks allows collaborative notebook development, scalable compute environments, and seamless data transformation workflows. In this project, Databricks was used to ingest raw global export data, perform large-scale transformations, enrich data using reference datasets, and structure it into layered storage zones.

Step 1: Storing Raw Data in Catalog (Volumes)

We first ingested all global trade datasets into Databricks Unity Catalog Volumes, which serve as secure and scalable storage locations within Databricks.

-> A Volume is a storage container inside Unity Catalog used to store raw files such as CSV, JSON, or Parquet.

-> It ensures centralized governance, access control, and easy integration with Delta pipelines.

Step 2: Modular Design Using Three Notebooks

To maintain a clean and maintainable workflow, we followed a modular notebook approach:

- Bronze Notebook – Reads raw data from Volumes
- Silver Notebook – Cleans, standardizes, and transforms the data
- Gold Notebook – Creates final analytics-ready datasets

This modular separation improves readability, debugging, scalability, and reusability.

- **DATA PROCESSING**

🔷 Bronze Layer (Raw Ingestion)

Platform: Azure Databricks
Technology: PySpark + Delta Tables

✔️ Responsibilities

-> Read raw CSV files
-> Validate schema
-> Validate country and product references
-> Load into Bronze Delta Tables

Bronze tables are created in

capstone.default.bronze_fact
capstone.default.bronze_country
capstone.default.bronze_product

🔶 Silver Layer (Data Cleaning & Standardization)

Platform: Azure Databricks
Technology: PySpark Transformations

✔️ Responsibilities

-> Remove null critical fields

-> Remove negative values
-> Standardize country / product attributes using references
-> Remove duplicates
-> Cast to correct data types

The silver is created in

capstone.default.silver_fact

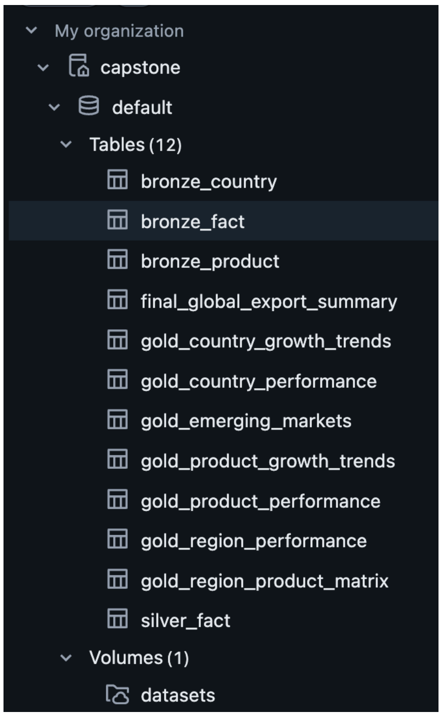
Result: trusted standardized dataset

🟡 Gold Layer (Analytics & Business Layer)

Platform: Azure Databricks
Technology: Grouping, Filtering, and Aggregation

✔️ Gold Tables Created

TABLE	PURPOSE
gold_country_performance	Top exporting countries with ranking
gold_product_performance	Product-wise performance
gold_region_performance	Region-level analytics
gold_country_growth_trends	YoY growth %
gold_product_growth_trends	Product YoY growth %
gold_region_product_matrix	Region vs Product export trends
gold_emerging_markets	Countries growing consistently


My organization
capstone
default
Tables (12)
bronze_country
bronze_fact
bronze_product
final_global_export_summary
gold_country_growth_trends
gold_country_performance
gold_emerging_markets
gold_product_growth_trends
gold_product_performance
gold_region_performance
gold_region_product_matrix
silver_fact
Volumes (1)
datasets

These datasets are analytics ready and directly consumed by Power BI.

🌀 AIRFLOW ORCHESTRATION

Apache Airflow acts as the orchestration engine for automating and managing the data pipeline. It enables scheduled execution of Databricks jobs, tracks pipeline success or failure, and ensures reliable workflow execution. Both manual and scheduled executions were supported, ensuring reusability and flexibility. Airflow helped establish a production-like automated ETL pipeline environment instead of running notebooks manually.

Platform: Apache Airflow
Purpose: Execute pipeline in order with dependencies

DAG Structure:

bronze_layer_ingestion
↓
silver_layer_processing
↓
gold_layer_processing

Setup of Airflow:

✔️ Prerequisites

-> Python 3.10
-> pip
-> Virtual Environment (recommended)

✔️ Stable setup of Airflow

-> Airflow 2.9.3
-> Databricks provider 4.0.0
-> Python 3.10

1️⃣ Create and Activate Virtual Environment

`python3 -m venv airflow310`
`source airflow310/bin/activate`

2️⃣ Install Airflow & Databricks Provider

`pip install "apache-airflow==2.9.3"`
`pip install apache-airflow-providers-databricks`

3️⃣ Initialize Airflow

`airflow db init`

4️⃣ Create Airflow Admin User

```
airflow users create \
--username admin \
--password admin \
--firstname Admin \
--lastname User \
--role Admin \
--email admin@example.com
```

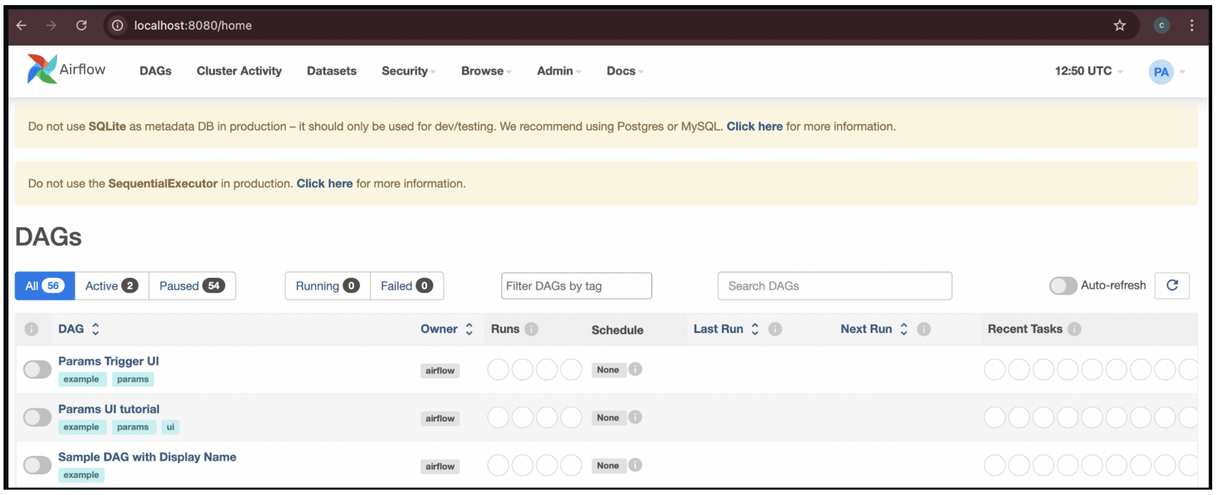
5️⃣ Start Airflow Services

On one terminal / cmd, run
`airflow weserver`

On another terminal / cmd, run

`airflow scheduler`

Then access airflow UI on "http://localhost:8080" or on any other port that was mentioned.



Do not use **SQLite** as metadata DB in production – it should only be used for dev/testing. We recommend using Postgres or MySQL. **Click here** for more information.

Do not use the **SequentialExecutor** in production. **Click here** for more information.

DAGs

DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks
Params Trigger UI example params	airflow		None			
Params UI tutorial example params ui	airflow		None			
Sample DAG with Display Name example	airflow		None			

Connect Databricks to Airflow:

1️⃣ Create Job

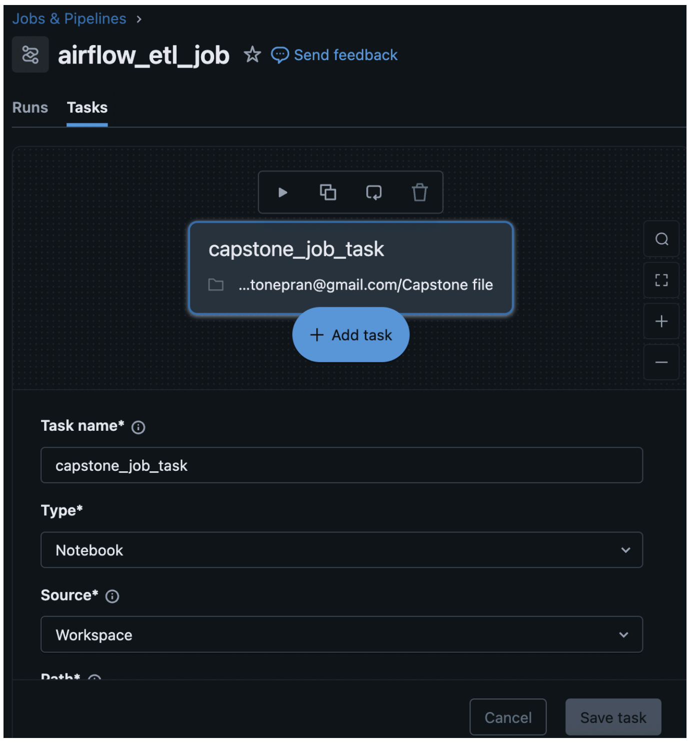
-> Login to Azure portal
-> Open your Databricks workspace
-> Click on "Launch Workspace"
-> From the left navigation, click "Jobs & Pipelines"

-> Click "Create Job"
-> Fill out the configuartion details and choose the notebook
-> Add notebook task

Task Name: bronze_task / silver_task / gold_task
Type: Notebook
Source: Workspace

-> Select Cluster
-> Save and run the job to verify. Copy the JobID from the URL

https://adb-xxxxxxxx/jobs/931572949178308
This number is the JobID.



capstone_job_task
…tonepran@gmail.com/Capstone file

Task name*: capstone_job_task

Type*: Notebook

Source*: Workspace

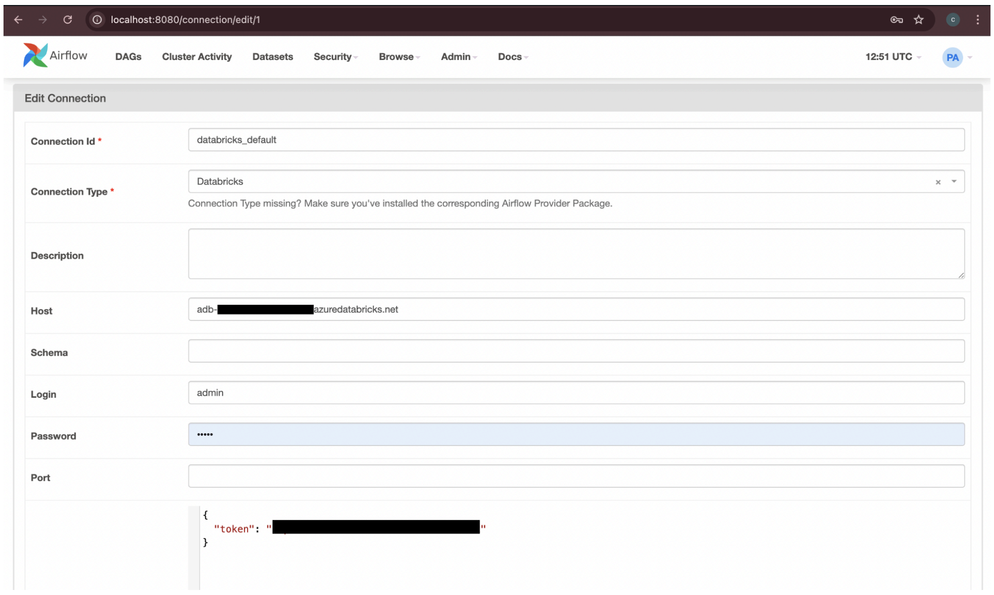
Path*

2 Create Connection on Airflow UI

Follow : Airflow UI → Admin → Connections → Add Connection

Fill the details:

-> Conn Id - databricks_capstone
-> Conn Type - Databricks
-> Host - adb-XXXXXXXXXXXX.azuredatabricks.net (It is the workspace url path)
-> Extras - { "token": "YOUR_DATABRICKS_PAT_TOKEN" }



Edit Connection

Connection Id *	databricks_default
Connection Type *	Databricks Connection Type missing? Make sure you've installed the corresponding Airflow Provider Package.
Description	
Host	adb-[redacted]azuredatabricks.net
Schema	
Login	admin
Password	•••••
Port	

```
{
  "token": "[redacted]"
}
```

3️⃣ Add DAG in Airflow

-> Place the DAG file into your Airflow DAGs folder:
`/Users/<username>/airflow/dags`

Example: cd ~/airflow/dags
nano global_trade_pipeline.py

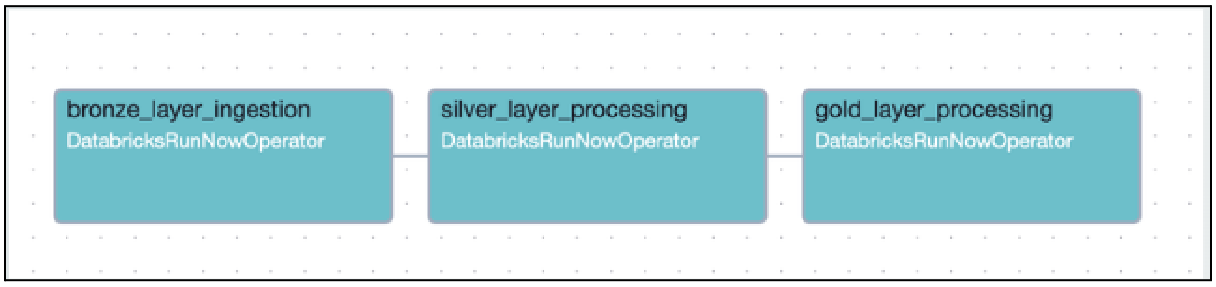
Write the DAG code in the nano editor or can directly access the DAG file from the folder and write the code there.

4️⃣ Run the DAG code

-> Restart the scheduler and web server
-> Identify the DAG in the Airflow UI, turn it ON, and trigger DAG

DAG Execution order:

Bronze Layer → Silver Layer → Gold Layer



Monitor execution via:

-> Graph View
-> Gantt View
-> Task Logs

5️⃣ Verify the workflow orchestration

-> Trigger the DAG and look at its status to check if it is a SUCCESS / FAIL
-> Open the job in Databricks workspace and verify if the job has run or not
-> Open Catalog -> Select a table -> Lineage. You can verify if the table has been updated or not

✔️ Benefits

-> Full automation
-> Dependency control
-> Logs & monitoring
-> Retries & scheduling capability

📊 **Power BI Dashboard**

Power BI was used to transform processed Gold-layer data into meaningful visual dashboards. Interactive charts, time-series visualizations, country performance comparisons, and product category distributions were created to support strategic decision-making. Power BI enables users to drill down, filter insights, and analyze trade performance dynamically.

The final Gold tables are visualized using Power BI.
We connect Power BI to Databricks to read analytical tables directly from Unity Catalog.

✔️ Prerequisites

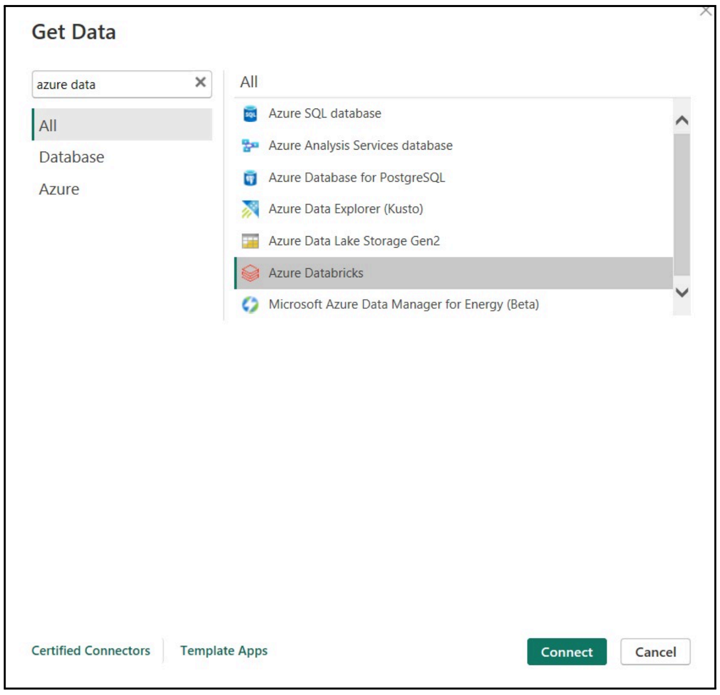
-> Power BI Desktop installed
-> Access to Azure Databricks Workspace
-> Unity Catalog / Delta tables created
-> Personal Access Token (PAT)

How to generate PAT in Databricks??

User Settings → Developer → Access Tokens → Generate Token

🔗 Connect Power BI to Databricks Cluster

-> Open Power BI → Get Data
-> Search Azure Databricks
-> Go to SQL Warehouses in Databricks and select the current compute
-> Go to "Connection Details" and copy paste the Server Hostname and HTTP Path in powerBI
-> Add the PAT in the following steps and click Connect
-> Select the tables that you would like to load. Click "Load Data"

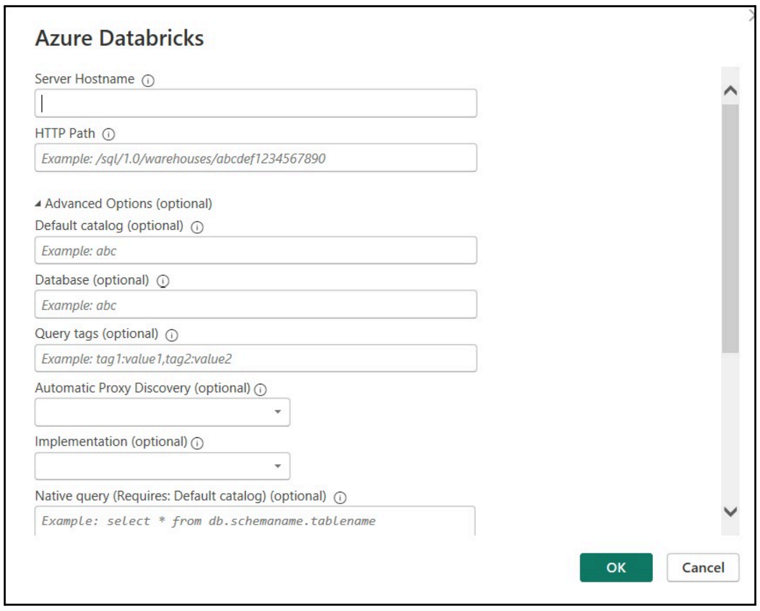

Get Data

azure data

All
Database
Azure

All

- Azure SQL database
- Azure Analysis Services database
- Azure Database for PostgreSQL
- Azure Data Explorer (Kusto)
- Azure Data Lake Storage Gen2
- Azure Databricks
- Microsoft Azure Data Manager for Energy (Beta)


Azure Databricks

Server Hostname:

HTTP Path: *Example: /sql/1.0/warehouses/abcdef1234567890*

Advanced Options (optional)

Default catalog (optional): *Example: abc*

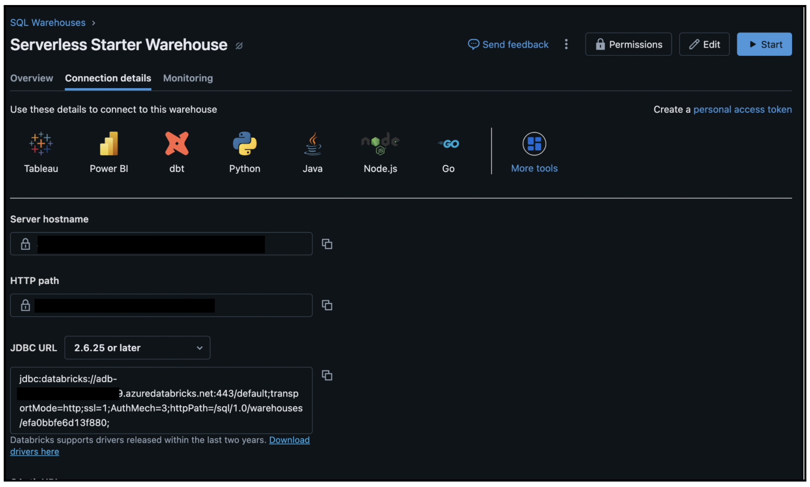
Database (optional): *Example: abc*

Query tags (optional): *Example: tag1:value1,tag2:value2*

Automatic Proxy Discovery (optional):

Implementation (optional):

Native query (Requires: Default catalog) (optional): `Example: select * from db.schemaname.tablename`



Serverless Starter Warehouse

Use these details to connect to this warehouse

Create a personal access token

Server hostname: [redacted]

HTTP path: [redacted]

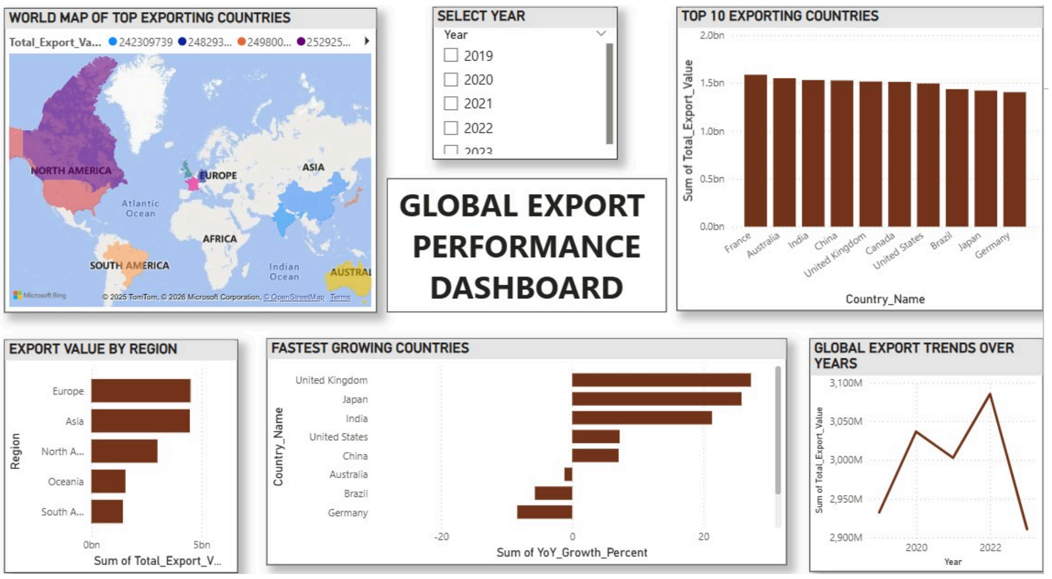
JDBC URL: 2.6.25 or later

```
jdbc:databricks://adb-
[redacted]9.azuredatabricks.net:443/default;transp
ortMode=http;ssl=1;AuthMech=3;httpPath=/sql/1.0/warehouses
/efa0bbfe6d13f880;
```

Databricks supports drivers released within the last two years. Download drivers here

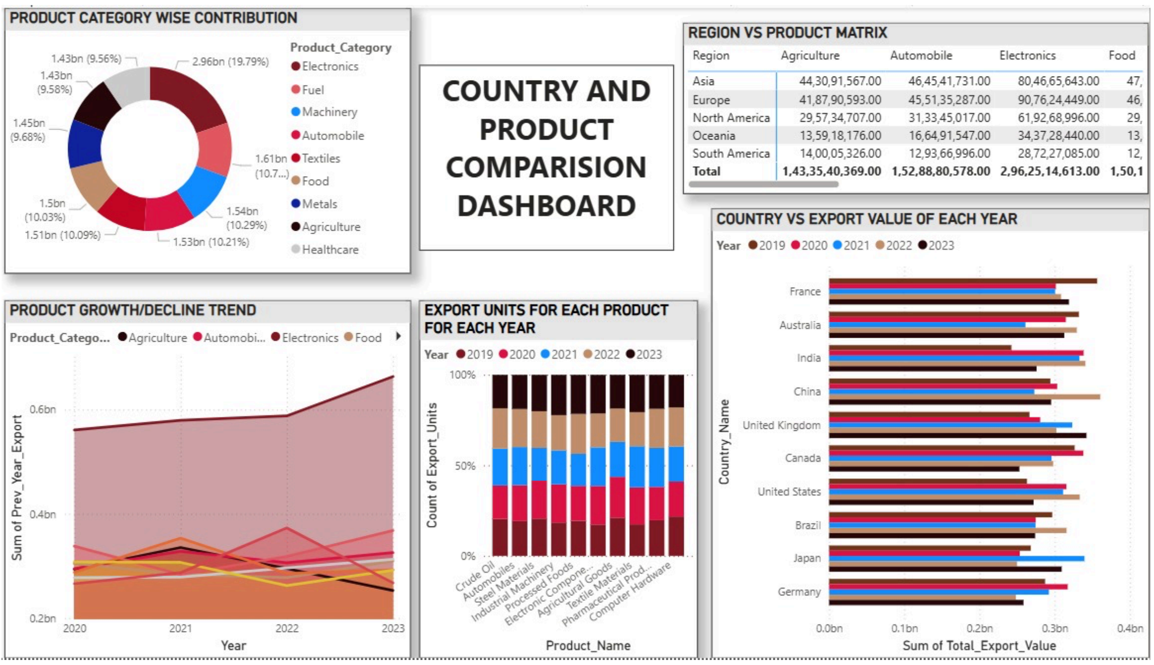
✔️ Key Dashboards

- Country Export Leaderboard
- Product Performance Trends
- Region Export Comparison
- Growth & Decline Analysis
- Emerging Markets Report



WORLD MAP OF TOP EXPORTING COUNTRIES
SELECT YEAR
Year
2019
2020
2021
2022
2023
GLOBAL EXPORT PERFORMANCE DASHBOARD
TOP 10 EXPORTING COUNTRIES
Sum of Total_Export_Value
Country_Name
EXPORT VALUE BY REGION
Region
Sum of Total_Export_V...
FASTEST GROWING COUNTRIES
Country_Name
Sum of YoY_Growth_Percent
GLOBAL EXPORT TRENDS OVER YEARS
Sum of Total_Export_Value
Year

Global Export Performance Dashboard - This dashboard provides a comprehensive analytical overview of global export performance, helping policymakers and analysts understand how different countries and regions are contributing to international trade.

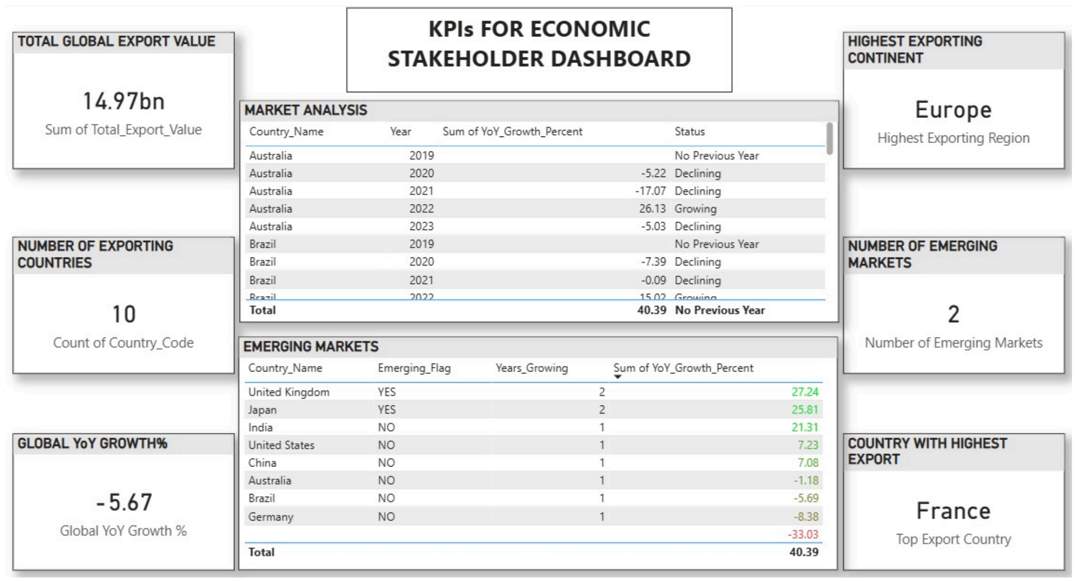


PRODUCT CATEGORY WISE CONTRIBUTION
Product_Category
Electronics
Fuel
Machinery
Automobile
Textiles
Food
Metals
Agriculture
Healthcare
COUNTRY AND PRODUCT COMPARISION DASHBOARD
COUNTRY VS EXPORT VALUE OF EACH YEAR
Year
2019
2020
2021
2022
2023
Country_Name
Sum of Total_Export_Value
PRODUCT GROWTH/DECLINE TREND
Sum of Prev_Year_Export
Year
EXPORT UNITS FOR EACH PRODUCT FOR EACH YEAR
Count of Export_Units
Product_Name

REGION VS PRODUCT MATRIX

Region	Agriculture	Automobile	Electronics	Food
Asia	44,30,91,567.00	46,45,41,731.00	80,46,65,643.00	47,
Europe	41,87,90,593.00	45,51,35,287.00	90,76,24,449.00	46,
North America	29,57,34,707.00	31,33,45,017.00	61,92,68,996.00	29,
Oceania	13,59,18,176.00	16,64,91,547.00	34,37,28,440.00	13,
South America	14,00,05,326.00	12,93,66,996.00	28,72,27,085.00	12,
Total	**1,43,35,40,369.00**	**1,52,88,80,578.00**	**2,96,25,14,613.00**	**1,50,1**

Country & Product Comparison Dashboard - This dashboard provides a detailed view of how different product categories and countries contribute to global exports, helping analysts understand trade composition, performance variation, and growth trends.



KPIs FOR ECONOMIC STAKEHOLDER DASHBOARD

TOTAL GLOBAL EXPORT VALUE
14.97bn
Sum of Total_Export_Value

NUMBER OF EXPORTING COUNTRIES
10
Count of Country_Code

GLOBAL YoY GROWTH%
-5.67
Global YoY Growth %

HIGHEST EXPORTING CONTINENT
Europe
Highest Exporting Region

NUMBER OF EMERGING MARKETS
2
Number of Emerging Markets

COUNTRY WITH HIGHEST EXPORT
France
Top Export Country

MARKET ANALYSIS

Country_Name	Year	Sum of YoY_Growth_Percent	Status
Australia	2019		No Previous Year
Australia	2020	-5.22	Declining
Australia	2021	-17.07	Declining
Australia	2022	26.13	Growing
Australia	2023	-5.03	Declining
Brazil	2019		No Previous Year
Brazil	2020	-7.39	Declining
Brazil	2021	-0.09	Declining
Brazil	2022	15.02	Growing
Total		**40.39**	**No Previous Year**

EMERGING MARKETS

Country_Name	Emerging_Flag	Years_Growing	Sum of YoY_Growth_Percent
United Kingdom	YES	2	27.24
Japan	YES	2	25.81
India	NO	1	21.31
United States	NO	1	7.23
China	NO	1	7.08
Australia	NO	1	-1.18
Brazil	NO	1	-5.69
Germany	NO	1	-8.38
			-33.03
Total			**40.39**

KPI Dashboard for Economic Stakeholders - This dashboard provides a strategic overview of global export performance through key KPIs, growth indicators, and emerging market insights. It helps policymakers and trade authorities evaluate economic strength, identify risk, and recognize emerging opportunities in global trade.

✔️ Insights Delivered

- Which countries dominate exports?
- Which products drive maximum revenue?
- Which regions perform best?
- Who is growing consistently?

🧠 Data Model — Star Schema

This project follows a Dimensional Model (Star Schema)

✔️ Fact Table

-> silver_fact

✔️ Dimension Tables

-> Country Dimension

-> Product Dimension

This improves:

-> Query performance
-> BI friendliness
-> Simplicity

🕒 Scheduling Strategy

Currently: Manually triggered via Airflow

Supports:
-> Daily
-> Weekly
-> Batch / Incremental
-> Event-triggered execution
The scheduling strategy can be defined in the DAG code.

```
with DAG(
  dag_id="global_trade_pipeline",
    start_date=datetime(2023, 1, 1),
    schedule_interval=None,
    catchup=False,
    description="Bronze -> Silver -> Gold Databricks Pipeline",
    tags=["capstone", "databricks"]
)
```

The schedule_interval can be changed to @daily, @weekly for scheduled execution.
None represents manual triggering.
<hr>

🚀 Run the project end-to-end

1️⃣Upload datasets to Databricks Volumes

2️⃣Run Bronze Notebook Manually

3️⃣Create Databricks Jobs

4️⃣Configure Airflow Connection

5️⃣Deploy DAG

6️⃣ Trigger Pipeline

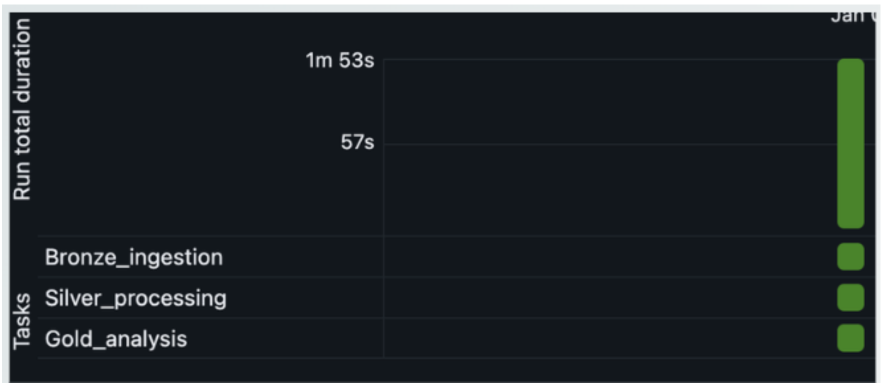
7️⃣ View Gold Tables in Databricks

8️⃣ Connect Power BI and Build Dashboard

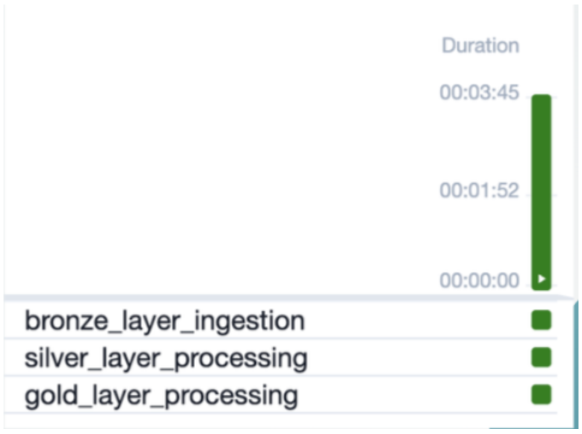
- **RESULTS & CONCLUSION**

When the DAG is triggered in the Apache Airflow, the workflow starts running by accessing the notebooks from Databricks. Automatically, the jobs also runs in Databricks when airflow job is triggered without manual triggering. Hence, the workflow is automated.

Through this project, we have succesfully automated an enterprise level ETL process that efficiently ingests, cleans, and transforms global trade data using Databricks and orchestrates the workflow throughApache Airflow. The processed Gold-layer data was then utilized to build insightful and interactive Power BI dashboards to represent data visually. Overall, this project demonstrates how modern tools can support data-driven business decisions.



Databricks UI



Airflow UI