(INT-27)

Statement of Work (SoW)for Retail Sales Data Processing & Analysis

By Abhishek Kumar

Intern

(INT-27)

(2025-26)

Table of Contents

Topic: Retail Data Processing & Analysis – Project Introduction

1: Introduction→

This project is a comprehensive implementation of a modern data pipeline designed to process and analyse retail sales data using the Medallion Architecture.

The Retail Data Processing & Analysis project is a complete, end-to-end data engineering solution developed to transform raw retail transaction data into meaningful business insights. It simulates how real-world retail companies manage, analyse, and visualize their sales data using a modern data pipeline architecture.

The core objective of this project is to take raw sales data—coming from multiple CSV files such as sales, customers, products, and store locations—and systematically process it through three logical layers using the Medallion Architecture: Bronze, Silver, and Gold.

How the project works:

1. Bronze Layer – Raw Data Ingestion
This stage is responsible for collecting raw CSV files and loading them into a structured storage format using PySpark. Minimal transformation is done at this point. The key focus is to capture the original data with metadata like ingestion date and file source to maintain data traceability and auditing.

2. Silver Layer – Data Cleaning & Enrichment
In the silver layer, the data undergoes cleaning (handling nulls, fixing data types), and multiple datasets (like product info, customer data, and store data) are joined with the sales data to create a consolidated and enriched view. Calculated fields like total_sales (price × quantity) are also introduced. The clean and enriched dataset is stored in a MySQL database for further analysis.

3. Gold Layer – Business-Level Aggregation
This final stage focuses on creating analytical tables that summarize the data to support business intelligence. It includes aggregations such as:

- Total revenue by product category or store
- Monthly and weekly sales trends
- Top-performing products and stores These summary tables are stored again in MySQL and serve as the foundation for dashboard reporting.

4. Power BI Dashboard – Data Visualization
To bring the analysis to life, the gold-layer data is connected to Power BI, where a dynamic dashboard is built. It includes visualizations like:

- KPIs: Total revenue, average order value, number of orders
- Bar charts: Top 5 selling products
- Line graphs: Sales trends over time

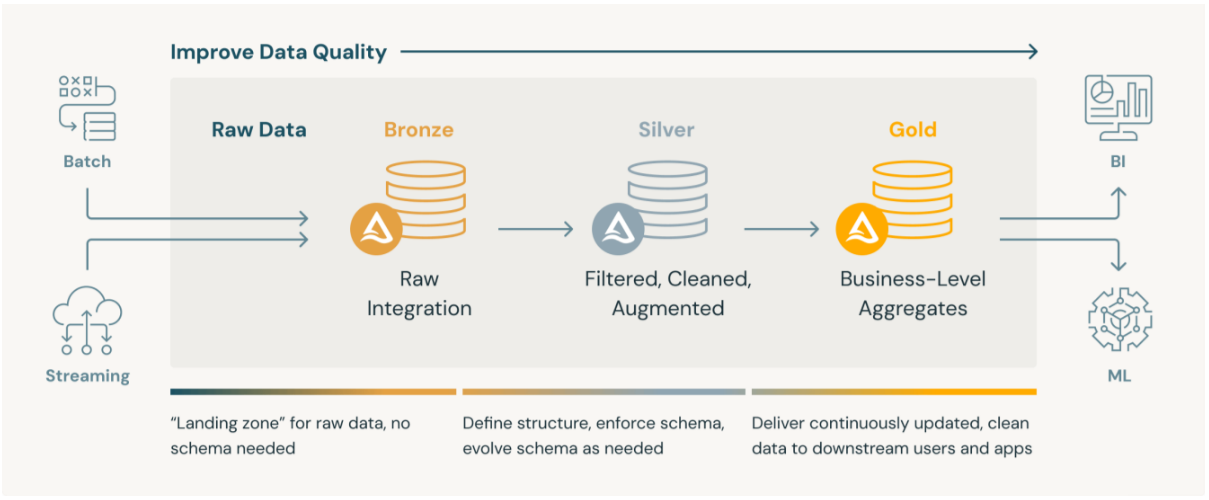
- Pie charts: Category-wise contribution
- Maps: Region/store-level performance

5. Automation, Auditing & Best Practices
The entire pipeline is automated using Python scripts to ensure data flows smoothly from ingestion to visualization. Audit fields are included at each layer to track when and how the data was processed. Clean code and proper documentation were maintained throughout the project.

1.1 Basic Concepts:

1.Medallion Architecture:

Medallion Architecture is a data design pattern with its core idea being to logically organize data into layers. Classic Medallion Architecture consists of three layers: bronze, silver, gold. Data is processed, cleaned, and moved between layers using data pipelines. The quality and structure of data increases from layer to layer.


Improve Data Quality
Batch
Streaming
Raw Data
Bronze
Silver
Gold
Raw Integration
Filtered, Cleaned, Augmented
Business-Level Aggregates
BI
ML
"Landing zone" for raw data, no schema needed
Define structure, enforce schema, evolve schema as needed
Deliver continuously updated, clean data to downstream users and apps

→**Bronze Layer:** Raw data as-is, ingested directly from source files.

→**Silver Layer:** Cleaned and enriched data after removing duplicates, fixing nulls, and joining related tables.

→**Gold Layer:** Aggregated, business-level data used for reporting and visualization.

2. PySpark:

PySpark is the Python API for Apache Spark, one of the most powerful open-source engines for large-scale data processing and analytics. It allows you to write Spark applications using Python, combining the simplicity of Python with the speed and scalability of Apache Spark.

→Reading and writing large datasets (CSV, Parquet, etc.)

→Cleaning and transforming data efficiently

→Performing joins and aggregations at scale

3.Parquet Files:

Apache Parquet is a modern, open-source, columnar storage file format specifically optimized for large-scale data processing. It is widely used in big data environments because of its efficiency in storage, speed, and compatibility with distributed processing systems.

4. MySQL:
A relational database used to store the cleaned (Silver Layer) and final analytical (Gold Layer) data. MySQL enables structured querying of the processed data and acts as a backend for the Power BI dashboard.

5. ETL Process (Extract, Transform, Load):

→Extract: Load raw data from CSV files

→Transform: Clean, join, enrich, and aggregate the data

→Load: Store the data in MySQL tables for further analysis

6. Data Cleaning:
Involves handling missing values, fixing data types, removing duplicates, and correcting inconsistent formats.

7. Power BI:
A Business Intelligence (BI) tool used to create interactive dashboards. It connects to MySQL and shows: KPIs ,Charts(line, bar, column) ,Maps (for location-based analytics)

8.ADF Automation: Azure Data Factory (ADF) automation refers to the ability to orchestrate, schedule, and manage end-to-end data workflows in a fully automated manner — without manual intervention. This means ADF can automate every stage of a data pipeline, from data ingestion to transformation and loading, at scheduled times or in response to specific triggers.

9. Databricks:

Databricks is a unified data analytics platform built for big data and AI workloads. It provides a collaborative environment to process large datasets using Apache Spark.

10. Microsoft Azure:

Azure is Microsoft's cloud computing platform offering over 200 services for compute, storage, networking, machine learning, and data analytics.

11.Azure Blob Storage:

Azure Blob Storage is Microsoft Azure's cloud-based object storage solution for storing large amounts of unstructured data — such as text, images, videos, logs, or big data files (CSV, JSON, Parquet, etc.).

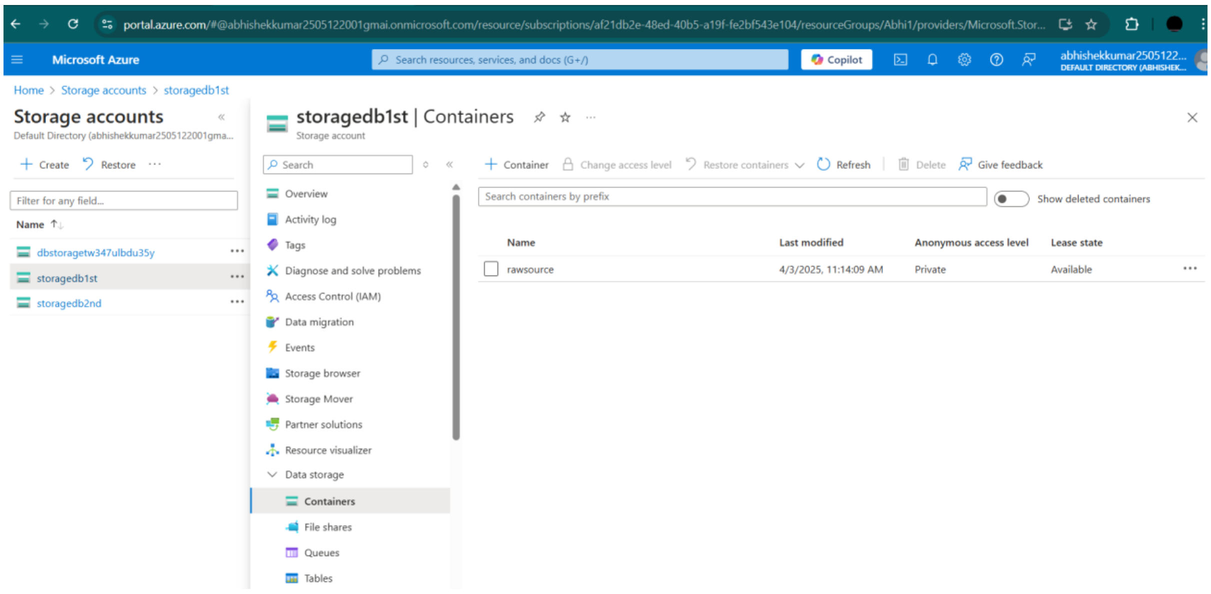
12. ADLS gen2:

Azure Data Lake Storage Gen2 is a highly scalable and secure data lake platform built on top of Azure Blob Storage. It combines the capabilities of hierarchical file systems (like Hadoop) with the durability, scalability, and performance of Azure Blob Storage.

2. Architecture Overview:

In this project, we follow the Medallion Architecture, which breaks the data pipeline into three main layers: Bronze, Silver, and Gold. Each layer represents a step in cleaning and refining the data, making the entire process more scalable, easier to manage, and easier to track.

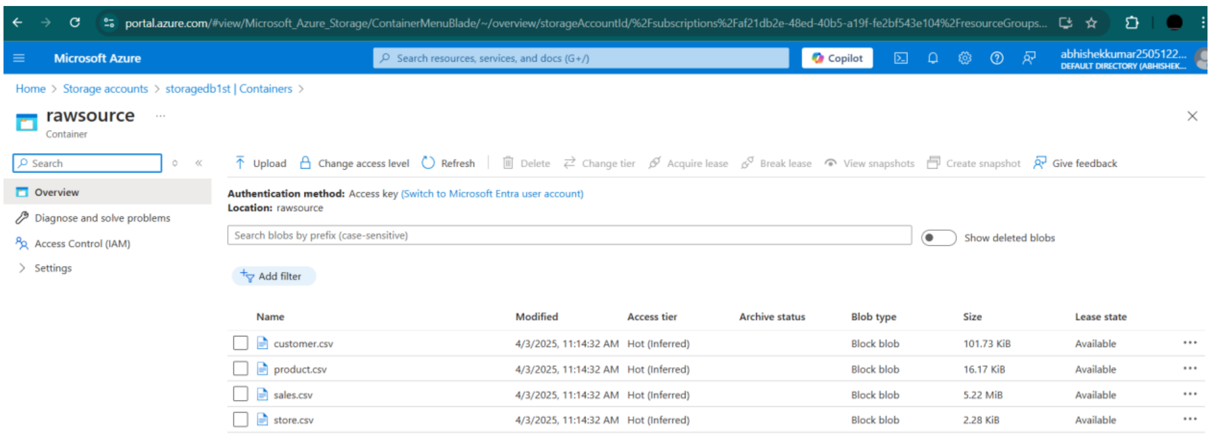
In this project, the Medallion architecture is implemented using Azure Data Lake Storage Gen2. However, since the original source file was in CSV format and the raw data needed to be in Parquet format for more efficient processing, the file was first stored in Azure Blob Storage. It was then transformed and brought into ADLS Gen2 for further processing.



Name
dbstoragetw347ulbdu35y
storagedb1st
storagedb2nd

Name	Last modified	Anonymous access level	Lease state
rawsource	4/3/2025, 11:14:09 AM	Private	Available

Fig: 2.1 Blob Storage having the container rawsource



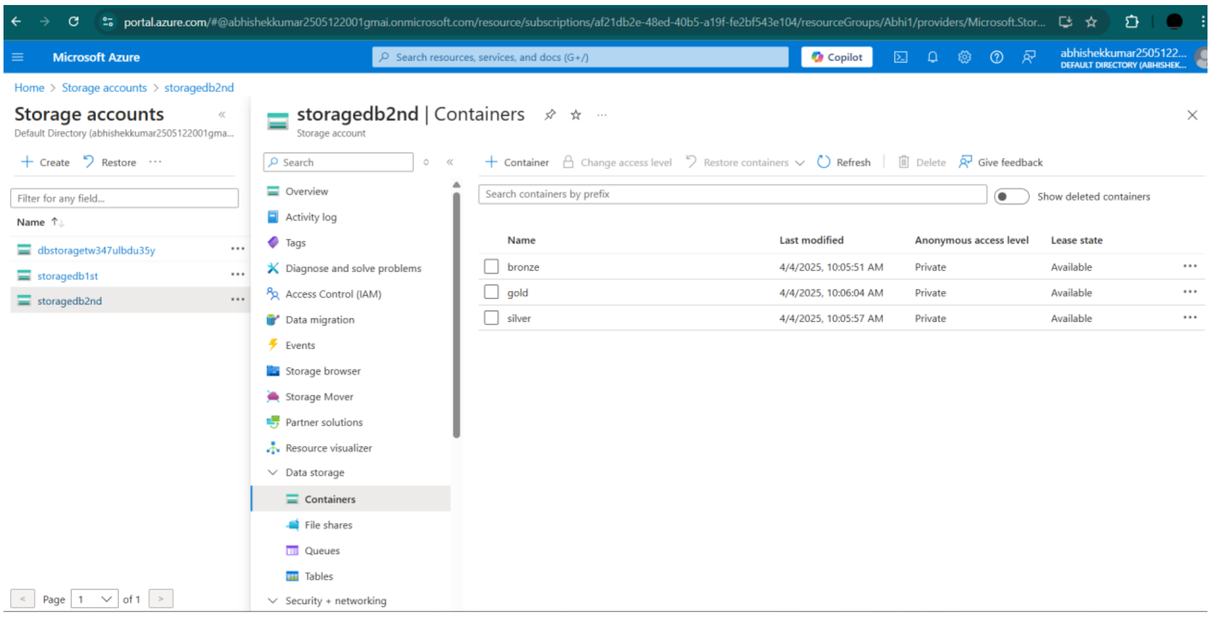
Authentication method: Access key (Switch to Microsoft Entra user account)
Location: rawsource

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
customer.csv	4/3/2025, 11:14:32 AM	Hot (Inferred)		Block blob	101.73 KiB	Available
product.csv	4/3/2025, 11:14:32 AM	Hot (Inferred)		Block blob	16.17 KiB	Available
sales.csv	4/3/2025, 11:14:32 AM	Hot (Inferred)		Block blob	5.22 MiB	Available
store.csv	4/3/2025, 11:14:32 AM	Hot (Inferred)		Block blob	2.28 KiB	Available

Fig: 2.2 The rawsource container containing csv's

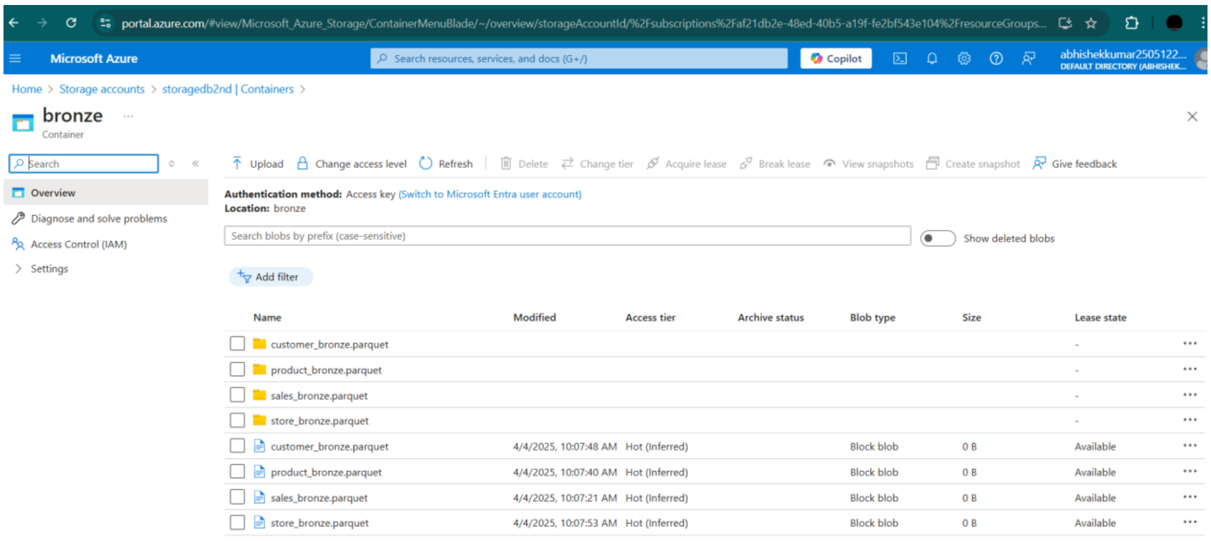
2.1.Bronze Layer:

The Bronze layer holds the raw, untouched data just as it came from the source system — the only change is that it's stored in Parquet format for better performance. No data types are defined at this stage. This layer is important because it acts as a backup — all the later steps (Silver and Gold) can be recreated from it if needed. It usually also includes useful details like when the data was added, the original file name, or the source it came from. The Bronze layer can have multiple tables, each representing different stages of raw data collection.



Name	Last modified	Anonymous access level	Lease state
bronze	4/4/2025, 10:05:51 AM	Private	Available
gold	4/4/2025, 10:06:04 AM	Private	Available
silver	4/4/2025, 10:05:57 AM	Private	Available

Fig:2.1.1: ADLS gen 2 storage containing all three layers bronze, silver, gold


portal.azure.com/#view/Microsoft_Azure_Storage/ContainerMenuBlade/~/overview/storageAccountId/%2Fsubscriptions%2Faf21db2e-48ed-40b5-a19f-fe2bf543e104%2FresourceGroups...
Microsoft Azure
Home > Storage accounts > storagedb2nd | Containers >
bronze
Container
Overview
Diagnose and solve problems
Access Control (IAM)
Settings
Upload
Change access level
Refresh
Add filter

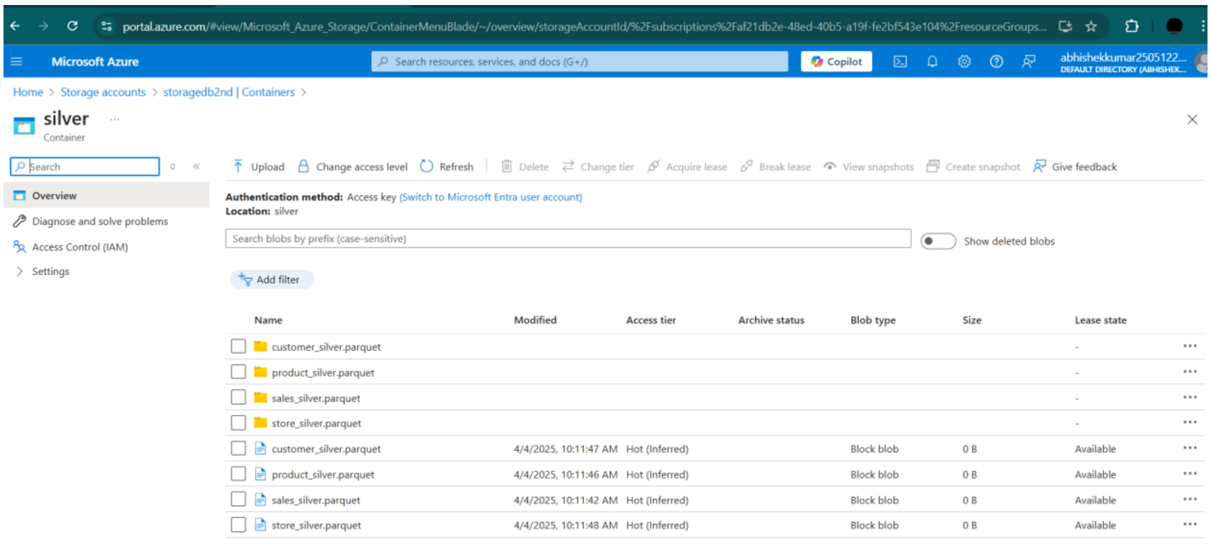
Authentication method: Access key (Switch to Microsoft Entra user account)
Location: bronze

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
customer_bronze.parquet						-
product_bronze.parquet						-
sales_bronze.parquet						-
store_bronze.parquet						-
customer_bronze.parquet	4/4/2025, 10:07:48 AM	Hot (Inferred)		Block blob	0 B	Available
product_bronze.parquet	4/4/2025, 10:07:40 AM	Hot (Inferred)		Block blob	0 B	Available
sales_bronze.parquet	4/4/2025, 10:07:21 AM	Hot (Inferred)		Block blob	0 B	Available
store_bronze.parquet	4/4/2025, 10:07:53 AM	Hot (Inferred)		Block blob	0 B	Available

fig:2.1.2: inside the bronze layer→ contains the converted parquet files from rawsource

2.2 Silver Layer

The Silver Layer consists of cleaned, standardized, and transformed data derived from the raw inputs in the Bronze Layer. At this stage, key data quality operations are performed, such as removing duplicate records and addressing corrupted or inaccurate data entries. Complex fields may be broken down into separate columns, and appropriate data types are assigned to ensure consistency across the dataset. Null values from the critical columns are also removed. And then it is moved to the gold layer for further process. Silver layer can be stored in two formats parquet as well as SQL format. As we are writing the silver layer in both formats shown in figure 2.2.1,2.2.2



Authentication method: Access key (Switch to Microsoft Entra user account)
Location: silver

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
customer_silver.parquet						-
product_silver.parquet						-
sales_silver.parquet						-
store_silver.parquet						-
customer_silver.parquet	4/4/2025, 10:11:47 AM	Hot (Inferred)		Block blob	0 B	Available
product_silver.parquet	4/4/2025, 10:11:46 AM	Hot (Inferred)		Block blob	0 B	Available
sales_silver.parquet	4/4/2025, 10:11:42 AM	Hot (Inferred)		Block blob	0 B	Available
store_silver.parquet	4/4/2025, 10:11:48 AM	Hot (Inferred)		Block blob	0 B	Available

Fig2.2.1 Silver layer contains all the cleaned, transformed/processed data

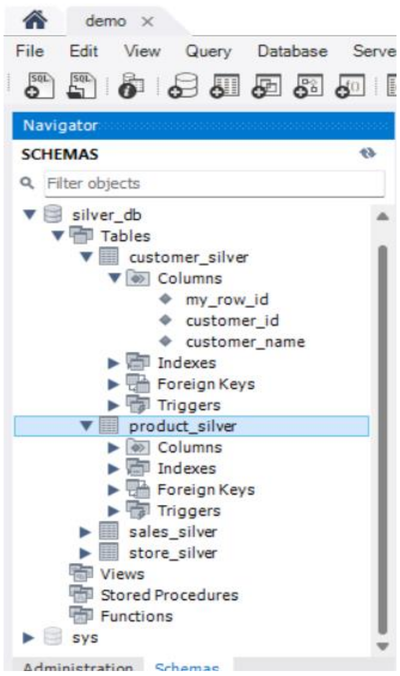


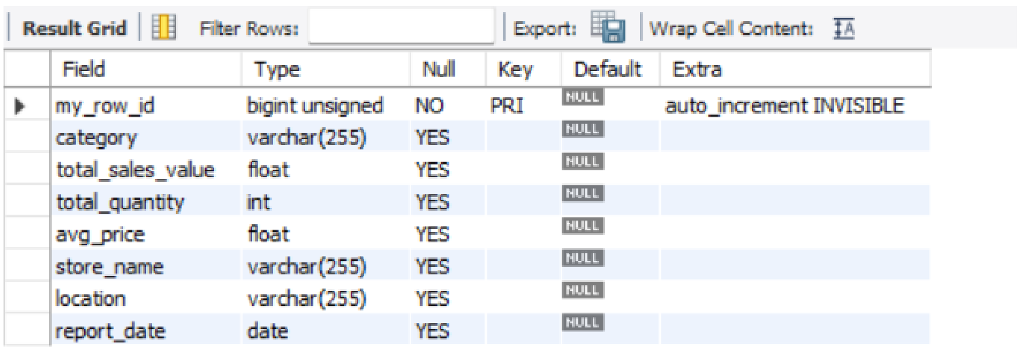
Fig:2.2.2 Silver_db

2.3 Gold Layer:

The Gold Layer contains final, ready-to-use data tailored for specific business use cases. It pulls relevant information from multiple Silver Layer tables (and sometimes external sources) to create meaningful reports and dashboards.

Operations Performed in MySQL (Using Silver Layer as Source)

- 1. **Sales Aggregation by Product Category** : SUM(total_sales_value) ,SUM(quantity) ,AVG(price)
- 2. **Store-wise Sales Analysis** :SUM(total_sales_value), grouped by store location.
- 3. **Time-based Sales Trends** : Aggregating sales data for trend analysis.

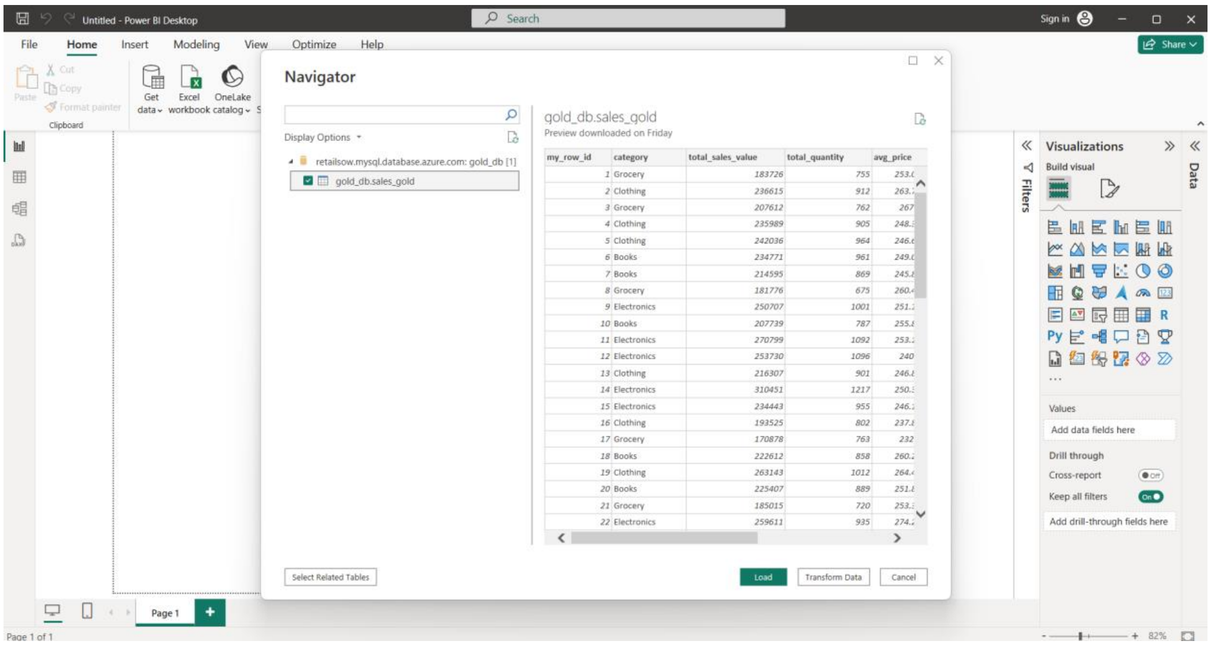


Field	Type	Null	Key	Default	Extra
my_row_id	bigint unsigned	NO	PRI	NULL	auto_increment INVISIBLE
category	varchar(255)	YES		NULL	
total_sales_value	float	YES		NULL	
total_quantity	int	YES		NULL	
avg_price	float	YES		NULL	
store_name	varchar(255)	YES		NULL	
location	varchar(255)	YES		NULL	
report_date	date	YES		NULL	

Fig:2.3.1:Gold_db schema

2.4. Power Bi:

Finally, the gold layer data is ingested to the power bi for generating insights and visualization



gold_db.sales_gold
Preview downloaded on Friday

my_row_id	category	total_sales_value	total_quantity	avg_price
1	Grocery	183726	755	253.
2	Clothing	236615	912	263.
3	Grocery	207612	762	267
4	Clothing	235989	905	248.
5	Clothing	242036	964	246.
6	Books	234771	961	249.
7	Books	214595	869	245.
8	Grocery	181776	675	260.
9	Electronics	250707	1001	251.
10	Books	207739	787	255.
11	Electronics	270799	1092	253.
12	Electronics	253730	1096	240
13	Clothing	216307	901	246.
14	Electronics	310451	1217	250.
15	Electronics	234443	955	246.
16	Clothing	193525	802	237.
17	Grocery	170878	763	232
18	Books	222612	858	260.
19	Clothing	263143	1012	264.
20	Books	225407	889	251.
21	Grocery	185015	720	253.
22	Electronics	259611	935	274.

fig2.4.1:Gold_db data

3. Dataset Schema:

3.1. product.csv

Column Name	Type	Description
product_id	int	Unique Product ID (Primary Key)
product_name	varchar	Name of the product
category	varchar	Product category

3.2. sales.csv:

Column Name	Type	Description
sales_id	int	Unique Sales ID (Primary Key)
product_id	int	Foreign Key to Product
customer_id	int	Foreign Key to Customer
store_id	int	Foreign Key to Store
quantity	int	Quantity of product sold

3.3. customer.csv

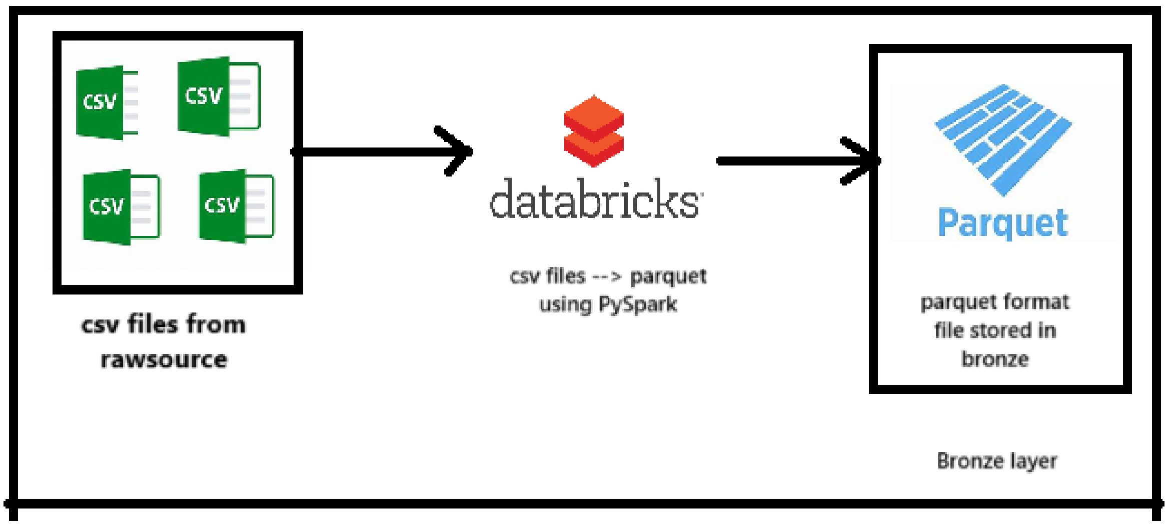
Column Name	Type	Description
customer_id	int	Unique Customer ID (Primary Key)
customer_name	varchar	Customer's full name
customer_type	varchar	Type of customer (e.g., Regular, Premium)

4. store.csv

Column Name	Type	Description
store_id	int	Unique Store ID (Primary Key)
store_name	varchar	Store name
location	varchar	Store location

4.Solution Implementation:

4.1 Data ingestion layer:


CSV
CSV
CSV
CSV
csv files from rawsource
databricks
csv files --> parquet using PySpark
Parquet
parquet format file stored in bronze
Bronze layer

Rawsource→Bronze

The data ingestion process begins by uploading the source CSV files to Azure Blob Storage, specifically within a container named rawsource, as illustrated in Figures 2.1 and 2.2. These files represent the initial raw data collected from the source systems.

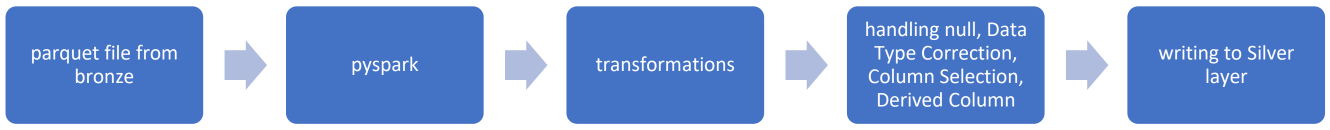
Using PySpark, the raw CSV files are then read from the rawsource container, processed, and converted into Parquet format—a more efficient and optimized file type for analytical workloads. The converted Parquet files are subsequently stored in Azure Data Lake Gen2, within the bronze container, as shown in Figure 2.1.2.

This Bronze Layer serves as the foundational stage of the Medallion Architecture, where the raw data is securely stored in a structured format, ready for downstream cleaning and transformation in the Silver Layer.

4.2 Transformation and cleaning:

Bronze→Silver

Once the raw data is ingested into the Bronze Layer in Parquet format, the next phase involves refining the data within the Silver Layer. Using PySpark, various transformation and data cleaning steps are performed to ensure the data is accurate, consistent, and ready for downstream analysis.


parquet file from bronze
pyspark
transformations
handling null, Data Type Correction, Column Selection, Derived Column
writing to Silver layer

The key steps carried out in this process include:

1. Data Loading from Bronze: The Parquet files stored in the Bronze container are read using PySpark for further processing. This ensures efficient loading and compatibility with large datasets.
2. Handling Null and Empty Values:

   →Unwanted null values and empty strings are identified and removed to maintain data integrity.

   →Columns containing only empty or whitespace entries are filtered out.

   →Records with missing essential fields are excluded to ensure completeness.
3. Dropping Duplicates:

→Duplicate records are detected and removed based on relevant identifiers to avoid skewed analysis.

4. Data Type Corrections:

   →Schema inference is validated, and columns are explicitly cast to their appropriate data types such as String Type, Integer Type, or Date Type, ensuring consistent formatting across the dataset.

5. Column Selection & Renaming:

   →Only necessary columns relevant to downstream use cases are selected.

   → Columns are renamed to follow standardized naming conventions, enhancing readability and consistency throughout the pipeline.

6. Derived Columns & Transformations:

   →New columns (e.g., Total_Price = Quantity × Price) are introduced to add business value.

   →Categorical fields may be mapped or cleaned (e.g., standardizing product categories or region names).

7. Saving the Silver Output:

   →The cleaned and transformed dataset is saved in Parquet format and stored back into the Silver container within Azure Data Lake Gen2.

   →This version of the data is now structured, reliable, and ready for analytical modelling and business intelligence.

4.3 Aggregation and Business logic → Gold Layer

The Gold Layer focused on generating business-focused aggregations that drive key performance indicators (KPIs) for the final Power BI dashboard. These aggregations were created using SQL queries executed on the cleaned Silver Layer tables in MySQL. The final output was stored in a dedicated fact table named sales_gold within the gold_db schema in MySQL, making it ready for reporting and analytical use.

4.4 Dashboarding with Power Bi:

The final stage of the project involved visualizing the curated data from the Gold Layer using Power BI. The sales_gold fact table, stored in the gold_db schema in MySQL, served as the primary data source for the dashboard.

KPI's**:**

1. Total Sales Revenue (Sum of total_amount from gold_sales)

2. Average Order Value (Total Sales / Number of Transactions)

3. Top 5 Best-Selling Products (Based on total_quantity from gold_sales)

4. Sales Trend Over Time (Monthly/Weekly sales visualization)

Dashboard:

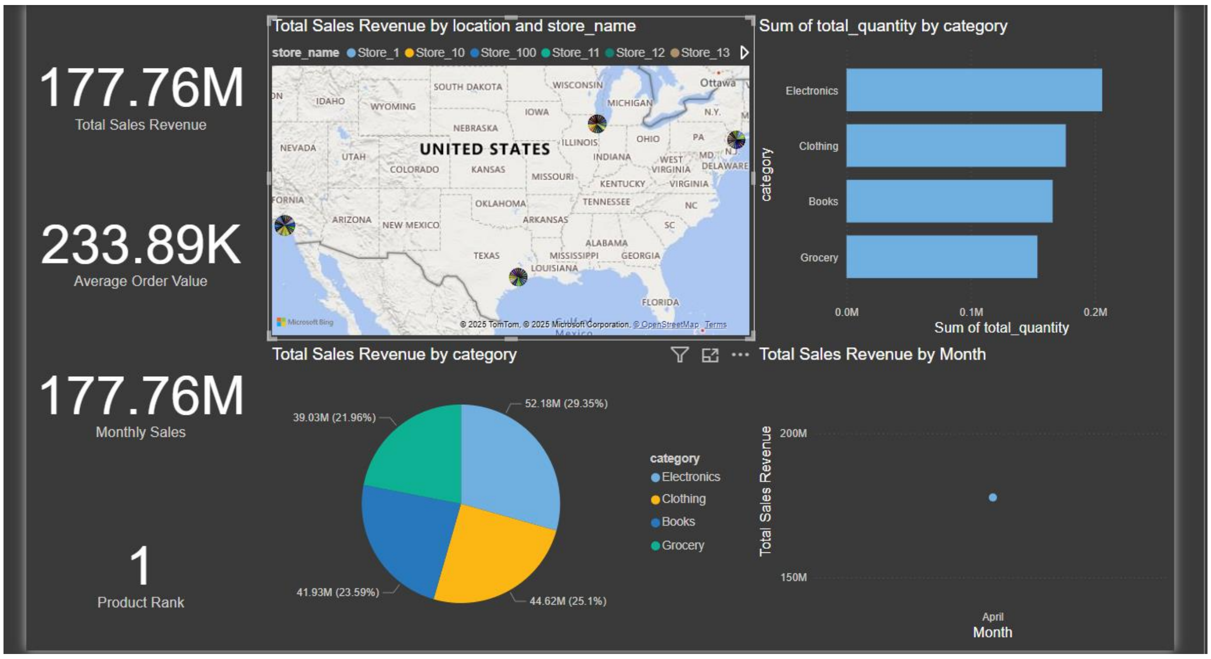


Fig4.4.1 Dashboard for KPI's and visualization

4.5 Automation using Azure Data Factory:

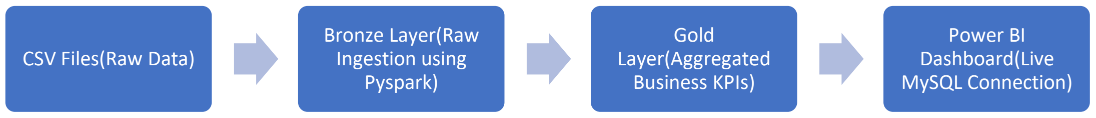
To ensure end-to-end automation, Azure Data Factory pipelines were created to orchestrate:

1.Trigger data ingestion from CSVs into Bronze.

2.Launch Databricks notebooks for silver and gold processing.

3.Load final gold data into MySQL.

This made the pipeline fully automated, requiring no manual intervention once deployed.


CSV Files(Raw Data)
Bronze Layer(Raw Ingestion using Pyspark)
Gold Layer(Aggregated Business KPIs)
Power BI Dashboard(Live MySQL Connection)

5. Git Repository:

All code, SQL scripts, transformation logic, and supporting files used in this project have been version-controlled and maintained in a Git repository. This ensures collaboration, traceability, and ease of access for future reference or enhancements.

Link → https://github.com/AbhishekK25122001/Retail_Data_Processing_SOW

6. Challenges & Solutions:

Throughout the implementation of this project, I encountered various challenges while working with big data technologies, cloud services, and dashboard integration. These challenges provided valuable learning experiences and helped me develop troubleshooting and problem-solving skills.

Challenges:

Finding apt location that provides MySQL flexible server service in Azure.

Connecting MySQL flexible server with workbench for querying.

Library unavailability caused error while writing to MySQL.

Connecting MySQL to Power BI prompted Error.

Solutions:

Going through latest blogs and updates to find locations that provide MySQL flexible server service in Azure.

Understanding and rectifying mistakes through documentations available online.

Fixing library unavailability by directly installing library into cluster.

Downloaded and installed missing .NET essentials for MySQL connectivity to Power BI.

7.Learnings

As a fresher working on my first industrial internship project, this experience was a significant milestone in my professional journey. It gave me the opportunity to work on real-world data, apply modern tools, and follow industry best practices throughout the end-to-end data engineering lifecycle.

Understood the importance of a layered architecture (Bronze, Silver, Gold) for organizing raw, clean, and business-ready data.

Gained practical knowledge of designing data pipelines, managing dependencies, and handling schema evolution.

developed PySpark notebooks for data ingestion, cleaning, and transformation.

Performed complex joins, aggregations, and business rule logic to generate KPIs using MySQL and Spark SQL.

Designed and published an interactive Power BI dashboard connected to the gold layer.

8 Conclusion

This project successfully delivered a complete, cloud-based data pipeline and analytics solution for retail_data_processing use case. Through the integration of Azure Databricks, Data Lake Storage, MySQL, Power BI, and Azure Data Factory, I implemented an end-to-end ETL architecture following the Medallion pattern (Bronze, Silver, Gold).

8.1 Achievements

Ingested raw logistics data from CSV into a structured data lake.

Built PySpark pipelines for data cleaning, enrichment, and transformation.

Designed MySQL-based aggregation logic to compute key business metrics.

Developed a dynamic Power BI dashboard showcasing real-time KPIs.

Automated the entire workflow using Azure Data Factory.

8.2 Impact on Logistics Stakeholders

The solution enables:

Operations Managers to monitor delivery performance, delays, and fuel usage.

Route Planners to identify optimization opportunities using historical trends.

Executives to view key metrics through an interactive dashboard for data-driven decision-making.

8.3 My Takeaways

Learned modern data engineering tools.

Handled real-world datasets and solved practical technical challenges.

Gained confidence in working with cloud platforms and automation tools.

Developed a deeper understanding of both backend data architecture and visualization.

This project enhanced my technical abilities and also taught me the value of structured workflows, documentation, and stakeholder communication. It has been a rewarding first step into the data industry, and a strong foundation for my future career in data engineering and analytics.