## IMG_256

### Colleague of computing

### Department of software engineering

Course name: Fundamental of Big Data analysis and Business intelligence

Course code: SEng5112

Title: Building an End-to-End Data Pipeline

**GitHub Repository**: https://github.com/Birhanubb/big-data-ETL-AND-BI

Name: Birhanu Bashaw

Id: 0130/13

### Submitted to :Derbew F.

**Submission date:13/02/2025**

**Introduction**

This assignment shows the full cycle of a data pipeline: **Extraction, Transformation, and Aggregation (ETL)** followed by **Visualization**. The process starts by extracting data from a CSV file, transforming it using PySpark for efficient data manipulation, performing aggregation using DuckDB for fast queries, and finally, visualizing the results through a bar chart , line chart and donut chart.

The e-commerce dataset used here contains transaction details, including customer information and purchase amounts. The goal of this project is to identify the top 50 customers based on their total purchases.

### Technologies Used

* **Python**: Programming language used for implementing the data pipeline.
* **Pandas**: A Python library for data manipulation and analysis, used for initial data extraction and inspection.
* **PySpark**: A framework for distributed data processing, used for data transformation and handling large datasets.
* **DuckDB**: An in-memory SQL database for fast aggregation and querying.
* **Matplotlib**: A library for creating visualizations in Python, used for generating a bar chart.
* **VS Code**: Integrated Development Environment (IDE) used to write and execute the Python code.

### Data Sources

The data source for this project is a **CSV file** that contains transaction details of customers from kaggle Data sets. The file has various fields, including customer IDs, transaction details, product prices, and transaction dates.

### Data Extraction Process

In this step, the data is extracted from the CSV file using **Pandas** to load it into a DataFrame. The extracted data is then previewed to inspect its structure.

The file path to the dataset is specified, and the file is read into a Pandas DataFrame. The first 5 rows of the dataset are displayed to get a sense of the data structure. Additionally, the shape of the dataset (number of rows and columns) is displayed for an initial overview. Missing values in each column are checked and printed to understand the completeness of the dataset.

### Data Transformation Using PySpark

After the data is extracted, the **PySpark** framework is used for scalable data transformations. PySpark is ideal for handling large datasets efficiently and supports distributed computing.

A PySpark session is initialized, and the data is loaded into a PySpark Data Frame. The first 5 rows of the PySpark Data Frame are displayed to verify the contents. Several transformations are applied to clean and structure the data, including filling missing values in the CustomerDOB and CustAccountBalance columns, removing invalid transactions (where price is less than or equal to 0), converting the TransactionDate column to the proper date type, and removing duplicates based on the TransactionID.

Once all transformations are completed, the PySpark DataFrame is converted into a Pandas DataFrame for further analysis.

### Aggregation Using DuckDB

For fast aggregation, the data is loaded into **DuckDB**, an in-memory database that supports SQL queries. DuckDB is optimized for high-performance analytical queries on large datasets.

An in-memory connection to DuckDB is established, and the Pandas DataFrame is registered as a table in DuckDB. A SQL query is written to calculate the total purchases per customer (CustomerID), and the top 50 customers are selected based on total sales. The results of the query are stored in a Pandas DataFrame for easy analysis and further processing.

### Data Visualization

In this final step, we visualize the aggregated data of **top 50 customers by total purchases** using different types of Diagrams. These visualizations allow us to better understand the sales distribution and trends among the top customers.

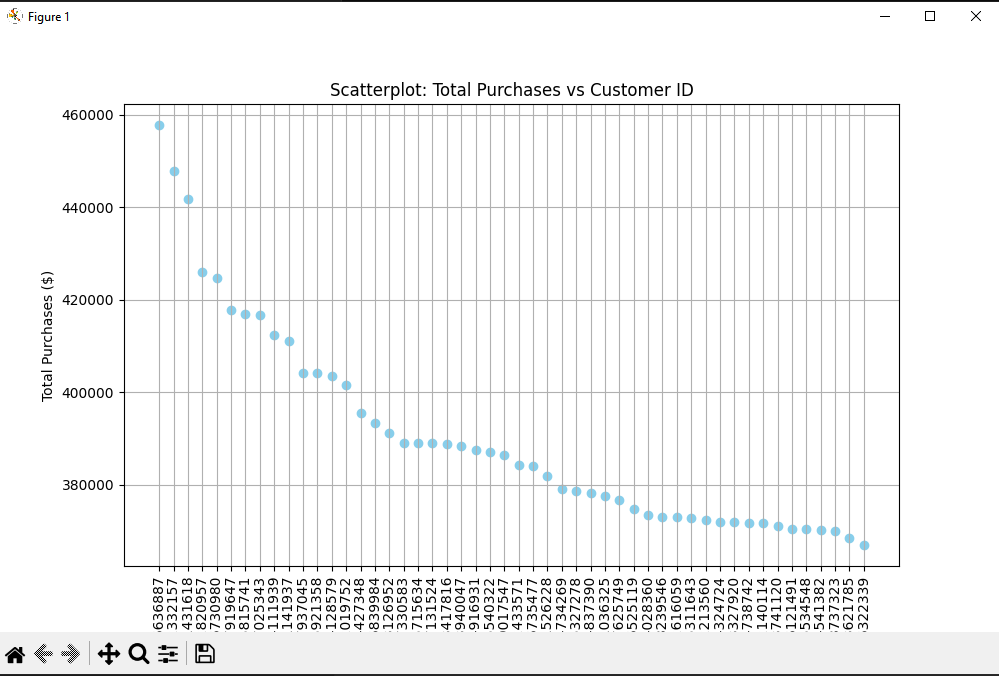
#### ****1. Bar Chart: Top 50 Customers by Total Purchases****

A bar chart is created to display the total purchases made by the top 50 customers. The chart is the x-axis labels (customer IDs) are rotated for better legibility.

#### **1**

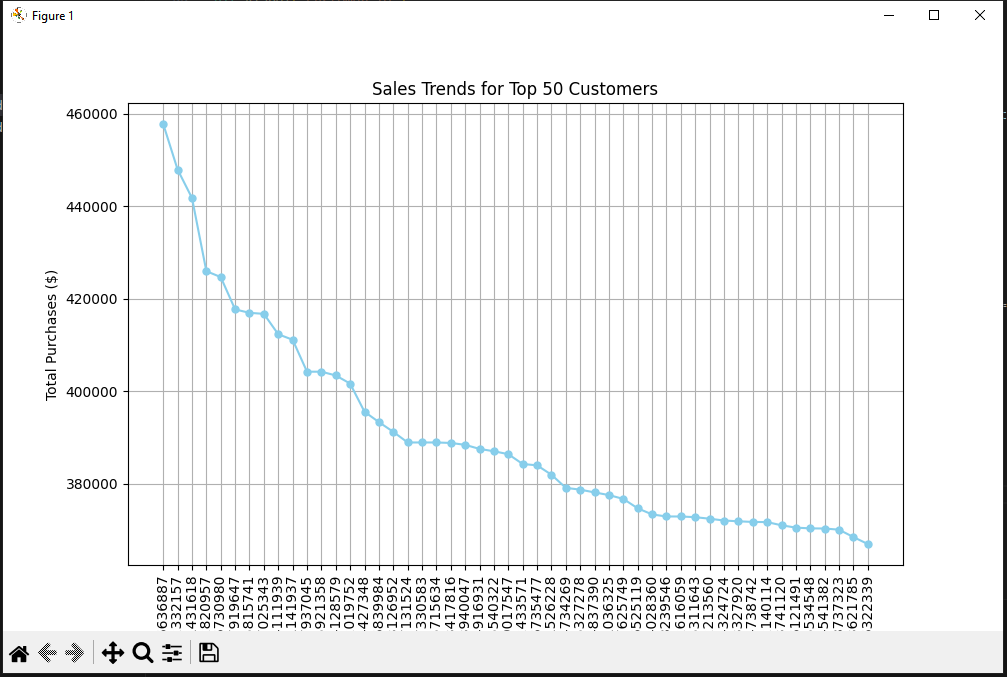
#### ****2. Scatterplot: Total Purchases vs Customer ID****

A scatterplot is used to visualize the relationship between **customer ID** and **total purchases**. This helps in identifying any outliers or patterns in the data.



#### ****3. Line Chart: Sales Trends for Top Customers****

A line chart is used to show the **sales trends** for the top 50 customers. It visually represents how the total sales evolve across customers.



#### ****4. Donut Chart: Total Sales Distribution by Customer****

A donut chart is created to show the distribution of total sales across the top 50 customers. It is a variation of the pie chart with a hole in the center to create the donut effect.

### 3

### Results and Insights

Data transformations were successfully applied to handle missing values, invalid entries, and duplicate records. The aggregation step enabled the identification of the top 50 customers based on their total purchases. The bar chart visualization provided a clear and easy-to-understand representation of the most valuable customers.

This project demonstrates how powerful data processing tools like **PySpark**, **DuckDB**, and **Matplotlib** can be used to clean, aggregate, and visualize large datasets efficiently.

**GitHub Repository Link**

**<https://github.com/Birhanubb/big-data-ETL-AND-BI>**

**File location**

**https://www.kaggle.com/datasets/thealpha22/e-commerce-1m**