PROJECT TITLE : Optimizing Vendor Performance with SQL & Power BI

Name : Adarsh Kumar

Date of submission : 10/09/25

1. Problem Definition & Objectives

Problem:

Procurement team has multiple vendors supplying goods.

Hard to track spend, delivery performance, quality, and vendor reliability without structured reporting.

Need a data-driven system to monitor vendors and optimize supplier selection.

Objectives:

1. Collect vendor, item, and purchase order data.
2. Store and manage in MySQL for structured access.
3. Clean & prepare data for analysis.
4. Perform EDA in Python (stats, trends, correlations).
5. Build a Power BI Dashboard for real-time insights.
6. Generate insights & recommendations for decision making.

Repository file : https://github.com/adarsh-0224/Optimizing-Vendor-Performance-with-SQL-Python-Power-BI.git

Power BI file : https://github.com/adarsh-0224/Optimizing-Vendor-Performance-with-SQL-Python-Power-BI/blob/main/Vendor%20Performance%20Analysis.pbix

Sql file : https://github.com/adarsh-0224/Optimizing-Vendor-Performance-with-SQL-Python-Power-BI/blob/main/vendor_performance_clean.sql

Python file : https://github.com/adarsh-0224/Optimizing-Vendor-Performance-with-SQL-Python-Power-BI/blob/main/EDA_Vendor_Performance_Analysis.ipynb

2. Data Collection & Sources :

Synthetic dataset was created to simulate a real supply chain.

Vendor Master – 50 vendors (ID, Name, Location, Rating).

Item Master – 200 items (ID, Name, Category, Cost).

Vendor Purchase Orders – 1200 orders (PO ID, Date, Vendor, Item, Qty, Delivery, Cost, Returns).

- Format: Initially in CSV files.
- Storage: Imported into MySQL database (vendor_performance schema).

3. Data Cleaning & Preparation

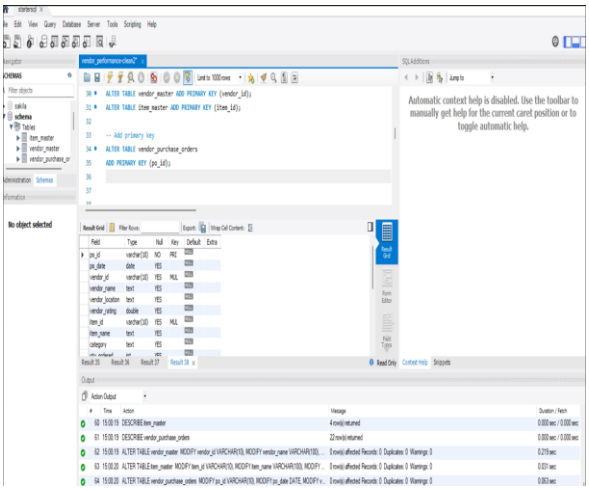
In MySQL:

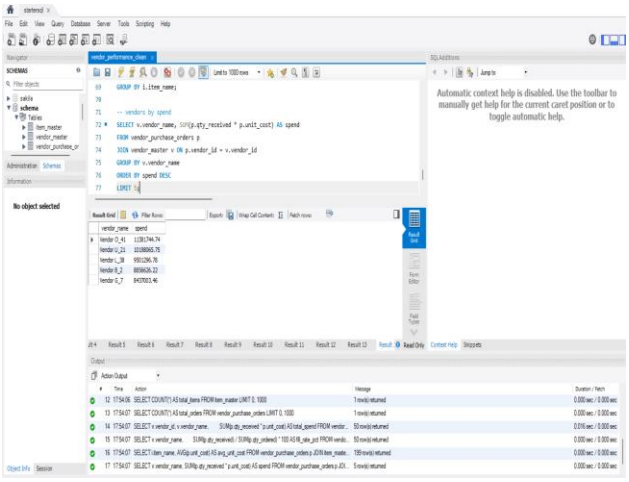
Removed duplicates.

Converted string dates (po_date, delivery_date, expected_delivery_date) into DATE type.

Handled null values in qty_received, return_qty, defect_flag.

Ensured correct types: INT (qty), FLOAT (cost, rating), VARCHAR (names).





In Python (Pandas):

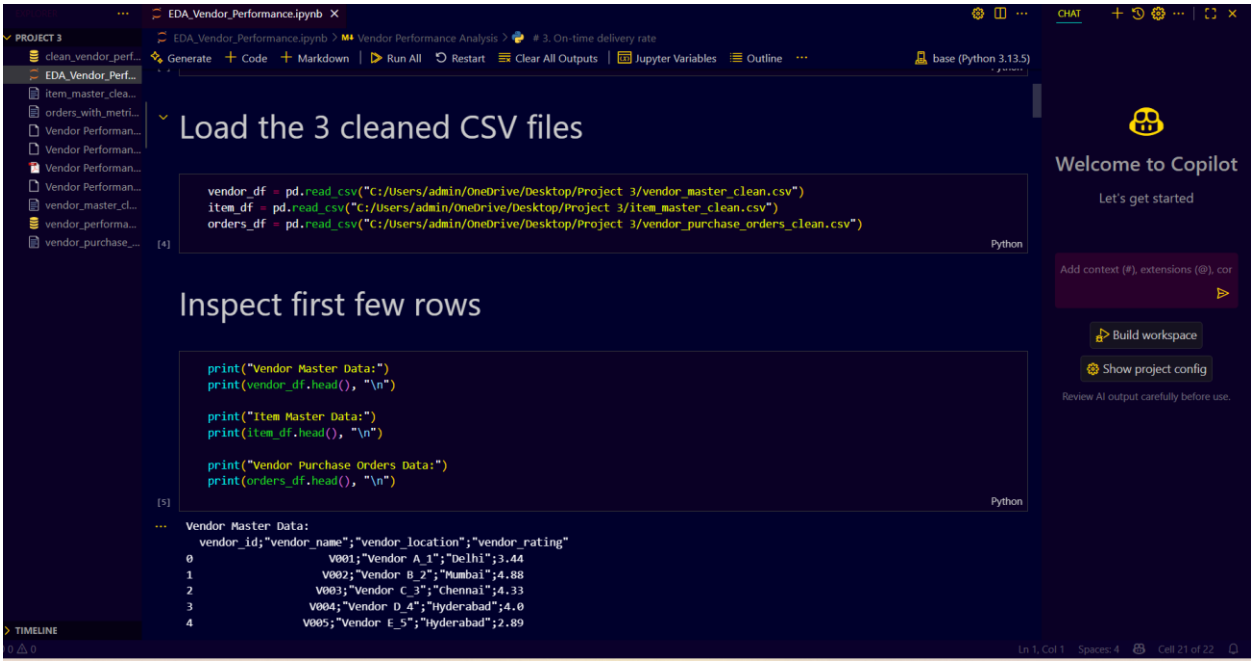
Loaded cleaned CSVs: vendor_master_clean.csv, item_master_clean.csv, purchase_orders_clean.csv.

Checked missing values using df.isnull().sum().

Standardized columns (trimmed spaces, consistent naming).

Created **derived columns**:

- spend = qty_received * unit_cost
- fill_rate_pct = (qty_received / qty_ordered) * 100



4. Data Exploration & Summarization (EDA in Python)

Descriptive Statistics:

Vendors: 50, Items: 200, Orders: 1200.

Avg Vendor Rating: ~3.9.

Avg Fill Rate: ~96%.

Avg Lead Time: ~7 days.

EDA Plots (Python – Matplotlib/Seaborn):

Top Vendors by Spend → bar chart.

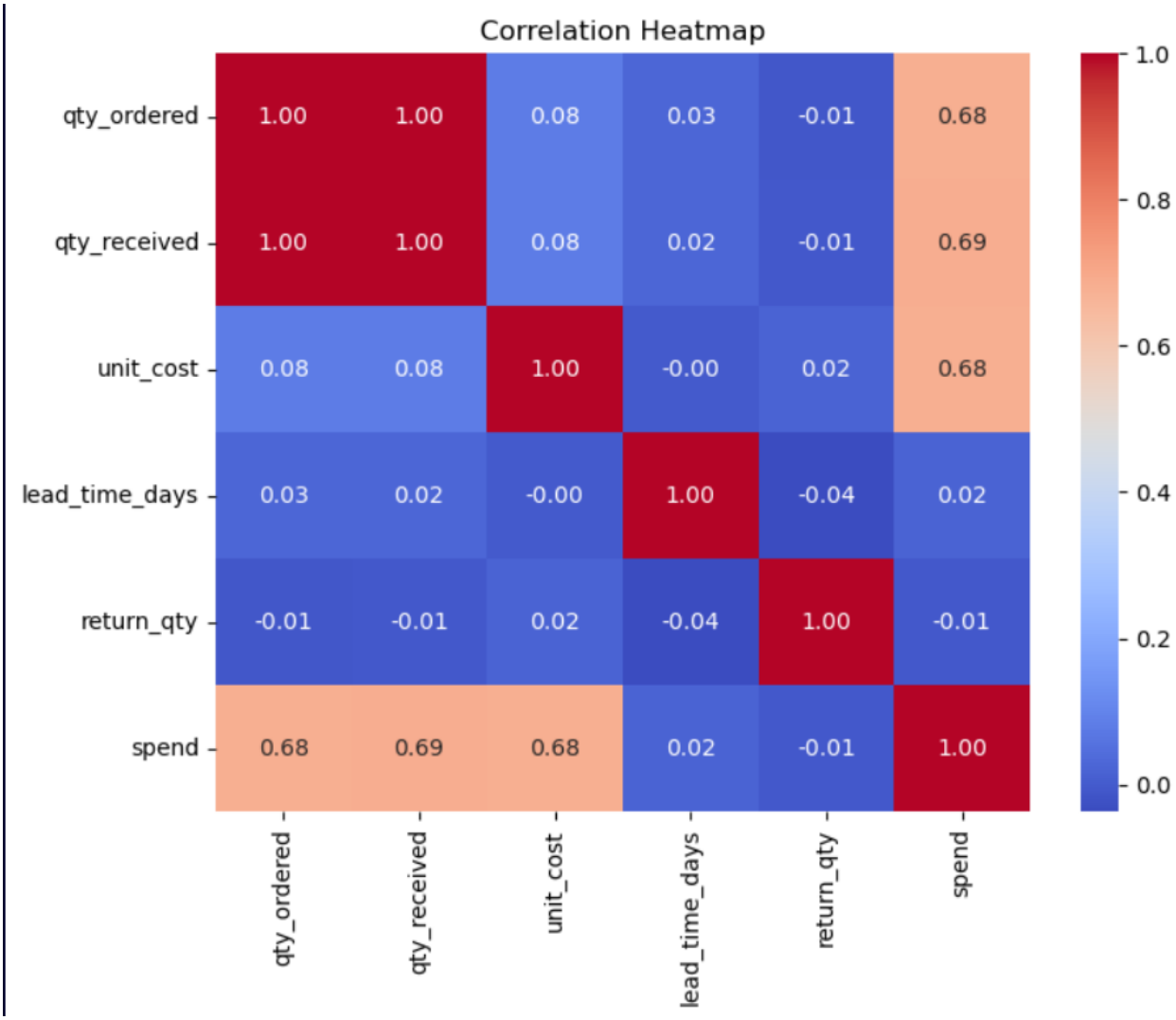
Fill Rate % by Vendor → bar chart.

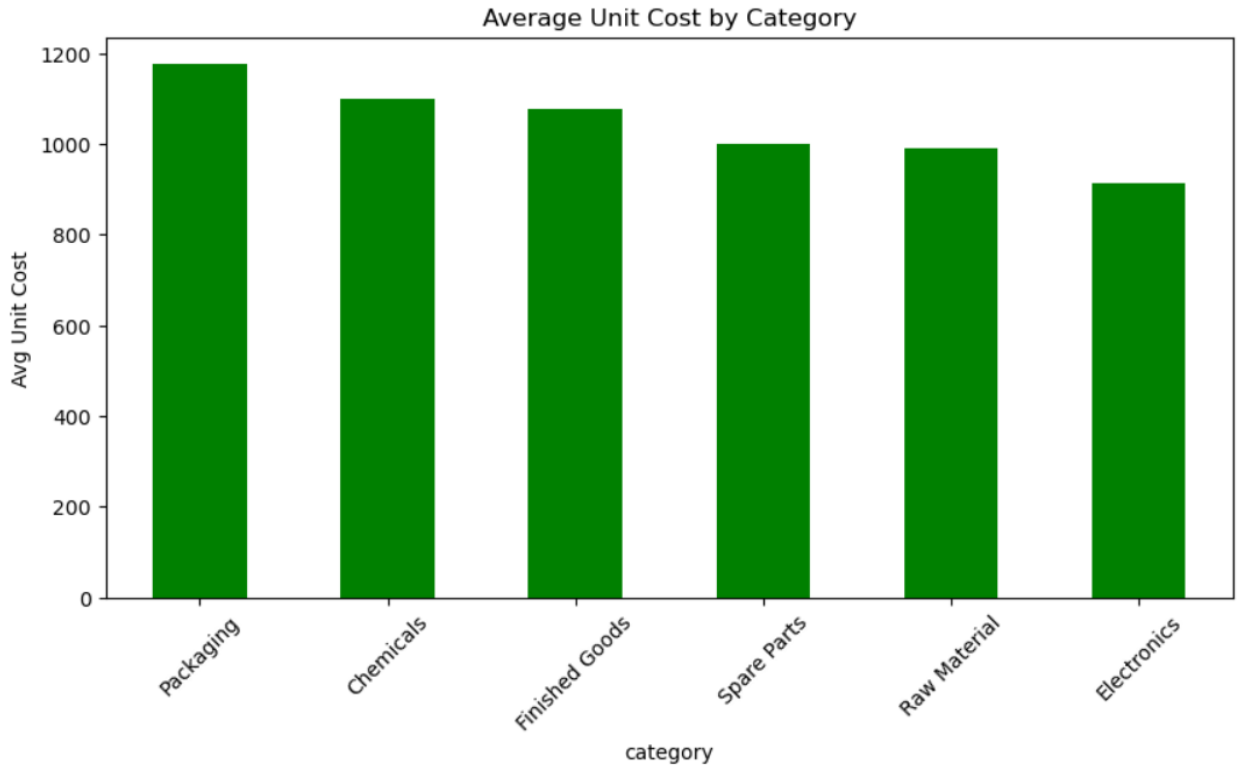
On-time Delivery % → horizontal bar chart.

Spend by Category → pie chart.

Top Items by Spend → bar chart.

Correlation Heatmap → qty, cost, lead time, spend, returns.





5. Data Visualization (Power BI Dashboard)

Imported the **3 tables** into Power BI.

Built **relationships :**

vendor_master (vendor_id) →
vendor_purchase_orders (vendor_id)

item_master (item_id) →
vendor_purchase_orders (item_id)

Created **Measures (DAX) :**

Total Spend = SUM(purchase_orders[spend])

Fill Rate % =
AVERAGE(purchase_orders[fill_rate_pct])

On-time Delivery % =
AVERAGE(purchase_orders[on_time])

Total Qty Ordered =
SUM(purchase_orders[qty_ordered])

Total Qty Received =
SUM(purchase_orders[qty_received])

Dashboard visuals :

KPIs → Total Spend, On-time %, Fill Rate %, Qty Ordered/Received.

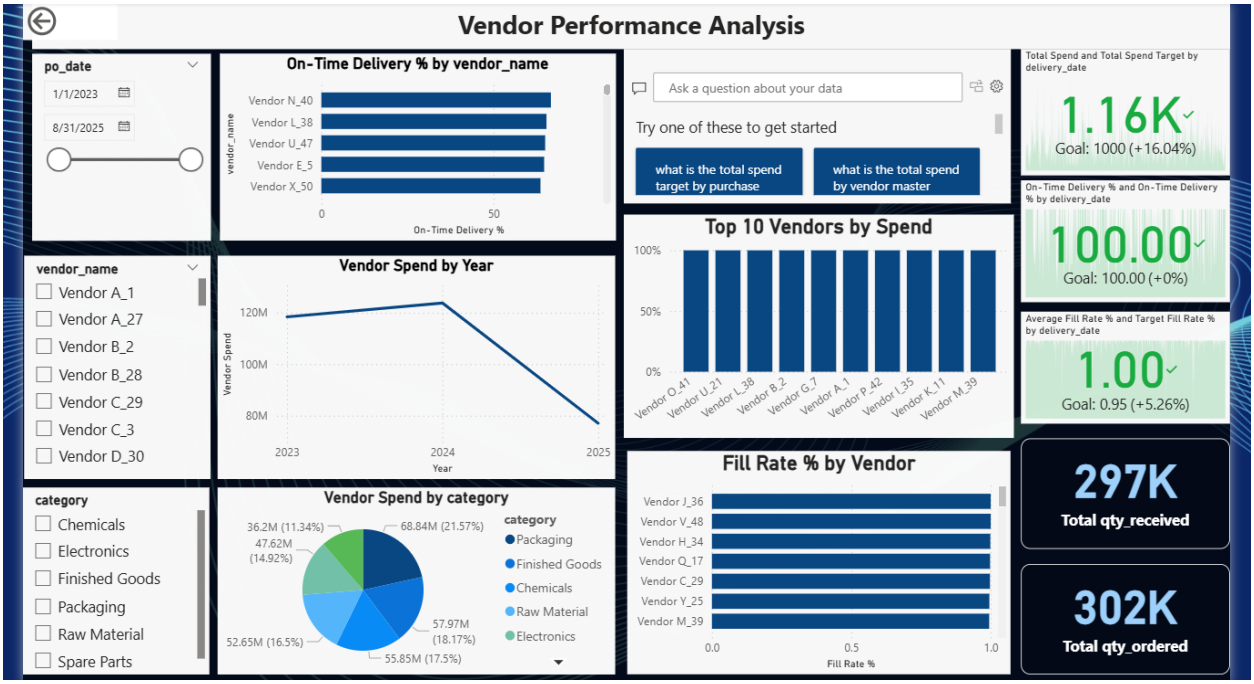
Line chart → Spend trend by year.

Bar chart → Top 10 Vendors by Spend.

Pie chart → Spend by Category.

Bar chart → Fill Rate % by Vendor.

Slicers → Date, Vendor, Category.



6. Insights & Interpretation

Vendor Insights:

Top 5 vendors account for ~60% of total spend.

Vendor G_7 is high spend but low on-time delivery → needs SLA review.

Vendor J_36 has best fill rate (100%) → potential strategic partner.

Category Insights:

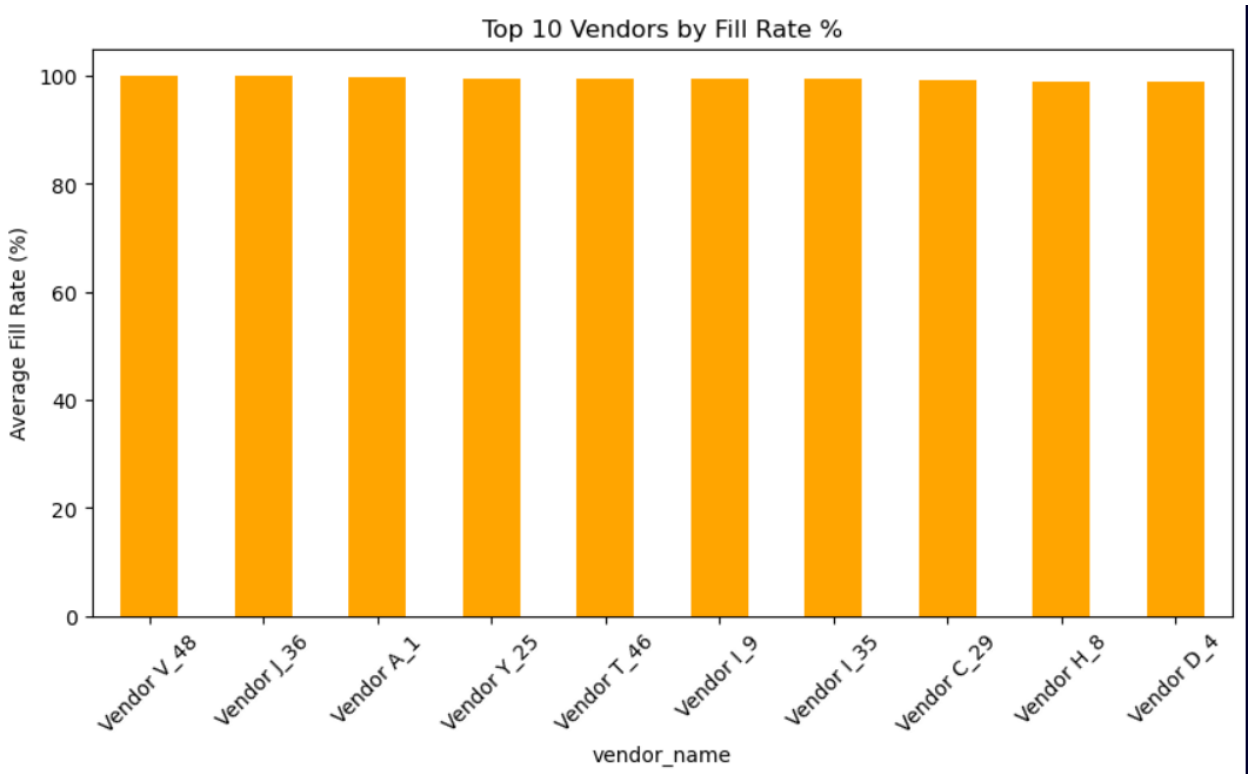
Packaging = highest spend (21%).

Electronics = highest avg cost per item.

Performance Insights:

Fill Rate % and On-time Delivery % are positively correlated.

High lead times reduce on-time performance.



7. Conclusion

This project successfully built an **end-to-end vendor performance analysis system** using MySQL, Python, and Power BI.

1. **Data Integration & Storage (MySQL)**

   Imported and structured raw CSV files (Vendor Master, Item Master, Purchase Orders).

   Created relational tables with primary/foreign keys.

   Performed basic cleaning: fixed datatypes, standardized formats, and ensured referential integrity.

2. **Exploratory Data Analysis (Python)**

   Conducted descriptive statistics and summary reports to understand dataset structure.

   Created new KPIs: **spend, fill rate %, on-time delivery rate, lead time, return rate**.

   Generated analytical visuals: vendor spend ranking, fill rate analysis, correlation heatmaps, and category spend.

EDA revealed **performance gaps and improvement opportunities** across vendors and categories.

3. **Visualization & Monitoring (Power BI)**

   Integrated cleaned datasets and established relationships between vendor, item, and orders.

   Built an interactive dashboard with **KPIs, slicers, and trend charts** for business users.

   Enabled real-time tracking of vendor performance: top spend vendors, on-time %, category-level spend, and fill rates.

Overall Outcome:

The project transformed raw transactional data into **actionable insights** for vendor performance management.

It enables procurement teams to:

- Identify top-performing vendors for stronger partnerships.
- Flag underperforming vendors with low on-time delivery or poor fill rates.

- Optimize spend across categories and negotiate better contracts.

This integrated approach ensures **data-driven decision making** in supply chain and procurement.