

Financial-Transaction-Data-Warehouse-Interactive-Dashboard

# *Financial-Transaction-Data- Warehouse-Interactive- Dashboard*

*Author:*  
*THOMAS ALFIO DRURY*

# Financial-Transaction-Data-Warehouse-Interactive-Dashboard

**Author:** Drury Thomas Alfio

## Summary

In this report, I will explain the process of data cleaning, transformation, and analysis using three raw data files to answer specific business questions.

### Datasets:

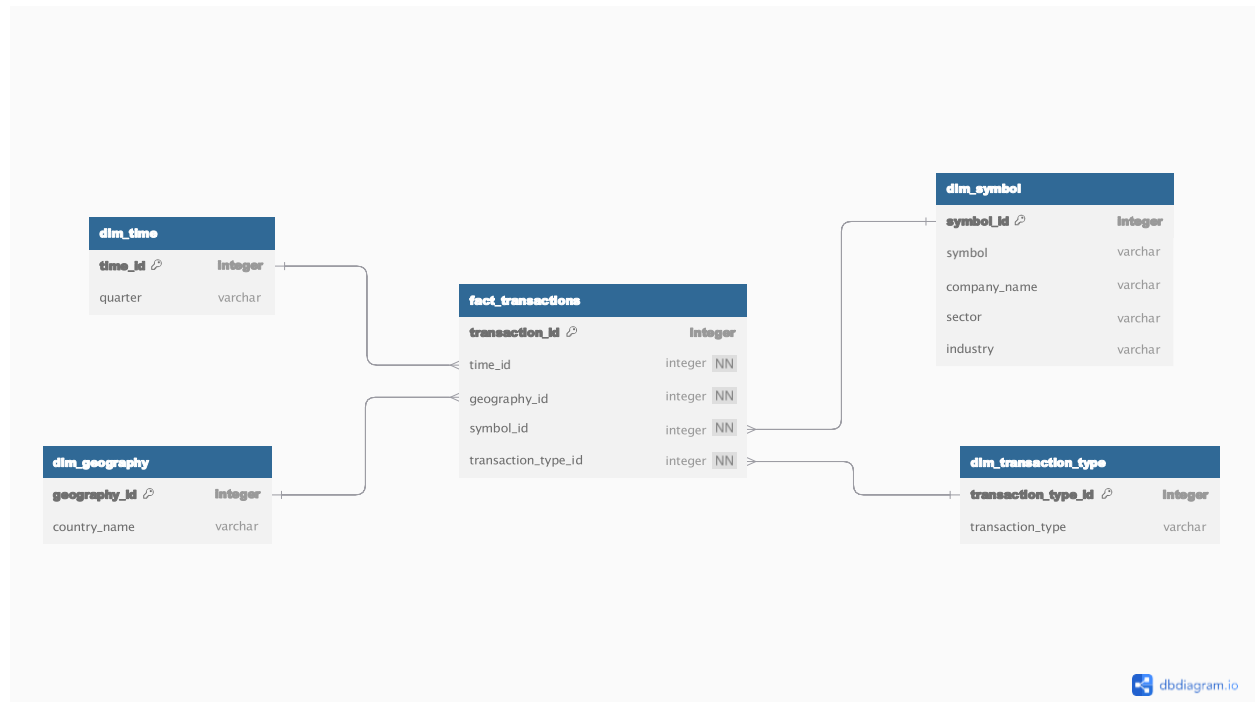
- **account-statement.csv:** Contains transaction data, including units traded, dates, and transaction types.
- **country.csv:** Provides general information about various countries.
- **symbols.csv:** Includes company-level details such as name, industry, and sector.

### Questions:

1. Top 5 sectors by number of SELL transactions in China during 2024
2. Top 5 industries by number of BUY transactions in Q4 2024
3. Quarterly ranking of 2024 based on total number of transactions (BUY + SELL).
4. A line chart showing the total number of transactions (BUY + SELL) over time.
5. Three bar charts showing (for the same date range and filtered data):
  - The top 3 traded symbols by transaction count.
  - The top 5 sectors by transaction count.
  - The top 5 industries by transaction count.

## Development Process

I began by analyzing the datasets to understand their structure and define the business and analytical requirements. Based on this analysis, I designed a star schema to support efficient querying and reporting. At the same time, I performed data cleaning to ensure data quality and integrity.






- **Country Name Variations:** Country naming issue (e.g. Taiwan in `symbols` file was 'Taiwan' while in `contry` file 'Taiwan, Province of China')
- **Symbol-Account Mismatches:** Identified and corrected mismatches between the `account` and `symbols` data. (Account transaction had info of companies not existing in the `symbols` dataframe)
- **Missing Data:** Removed null values and incomplete records to maintain data consistency.

## Streamlit Dashboard Implementation

The final solution features a fully interactive **Streamlit dashboard** with the following components:

- **Color-Coded Transaction Types:**

-  **BUY** transactions
-  **SELL** transactions
-  **DIVIDEND** transactions

- **Interactive Quarter Picker:**<sup>1</sup>

Users can select specific quarters using a slider to filter and analyze data dynamically. Company names and symbols, while not explicitly required for the business queries, were necessary elements to answer the question 'What are the top 3 traded companies by transaction count?' in the bar chart visualization. The dashboard also provides users the opportunity to display companies either by company name or by symbol.

- **A page that answers to the Pre-Built Query Analysis.**

- **Natural Language Query Page (Powered by Microsoft LIDA and OpenAI):**

This page allows users to ask questions in natural language. The system dynamically generates visualizations and answers, making advanced analytics accessible to non-technical users. **Note:** There is an example of use in the appendix of this document

## OLAP Operations Support

The system supports all major OLAP operations using the star schema design:

- **Roll-up:** Aggregate by higher-level dimensions (e.g., Individual companies → Sector)
- **Drill-down:** The opposite approach of Roll-up

---

<sup>1</sup> If you use an old version of Plotly, some graphs may look different than intended

- **Slice:** Filter data by one dimension (e.g., transactions in Q4 only)
- **Dice:** Filter by multiple dimensions (e.g., BUY transactions in China's technology sector, supported by LIDA)
- **Pivoting:** Rotate dimensions for alternative data perspectives (e.g., switching geography and time axes, or transaction type and sector views, supported by LIDA)

These capabilities enable users to explore data across the key dimensions: Time, Geography, Symbol, and Transaction Type.

## Final Deliverables

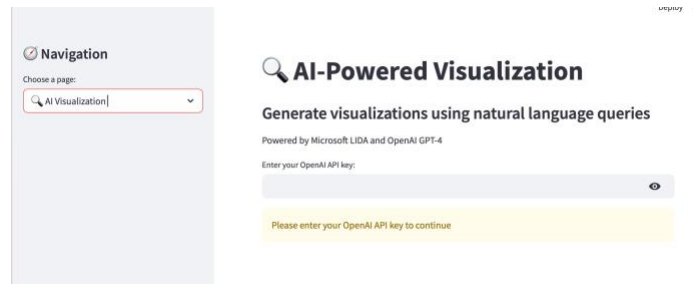
A zip file that includes the file used in particular:

- The raw df in the sub-folder '**raw\_file**' .
- '**requirements.txt**' - which contains the libraries used.
- A jupyter notebook - '**etl.ipynb**' -containing the ETL code process.
- A pdf of the star schema - '**Starschema.pdf**' .
- The final df filtered- '**transaction.merged**' -used in streamlit.
- The streamlit code - '**streamlit.py**'

## Appendix:

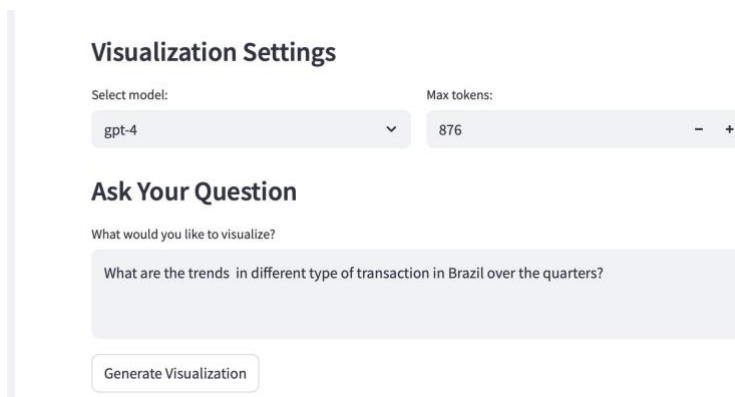
### Example of use of AI Visualization page:

Step 1: Write your Open Ai key in the box



The screenshot shows the 'AI-Powered Visualization' interface. On the left, a 'Navigation' sidebar has a dropdown menu with 'AI Visualization' selected. The main area is titled 'AI-Powered Visualization' and 'Generate visualizations using natural language queries'. It is powered by Microsoft LIDA and OpenAI GPT-4. There is a text input field for the 'Enter your OpenAI API key:' and a yellow warning box below it that says 'Please enter your OpenAI API key to continue'.

Step2: Choose the model, and ask any question you like in the box 'Ask Your Question' and then click Generate Visualization



The screenshot shows the 'Visualization Settings' section with a 'Select model:' dropdown set to 'gpt-4' and a 'Max tokens:' input set to '876'. Below this is the 'Ask Your Question' section with a text area containing the question: 'What are the trends in different type of transaction in Brazil over the quarters?'. A 'Generate Visualization' button is at the bottom.

Step 3: Wait approximately 5 second and you will get your graph automatically generated

