**Prepared-By: Firomsa Fekadu**

**ID No: DBU/041/14**

**E-Commerce Data ETL Pipeline Documentation**

This document provides an overview of the ETL (Extract, Transform, Load) pipeline used to process and analyze e-commerce data, along with the design choices, data schema, and assumptions made during the process.

# Data Source Identification & Understanding

**Dataset Selection:**

I downloaded the **“Olist”** dataset, which contains over **1 million rows**, from Kaggle. The dataset includes several tables related to an e-commerce business, such as **Customers**, **Orders**, **Products**, **Sellers**, and **Geolocation**. These tables are interlinked by keys like **customer\_id**, **product\_id**, and **seller\_id**, enabling analysis of customer behaviors, order trends, and seller performance. The data provides rich insights into product sales, customer locations, and order statuses, which will be useful for understanding business performance and making data-driven decisions.

**1. ETL Pipeline Overview**

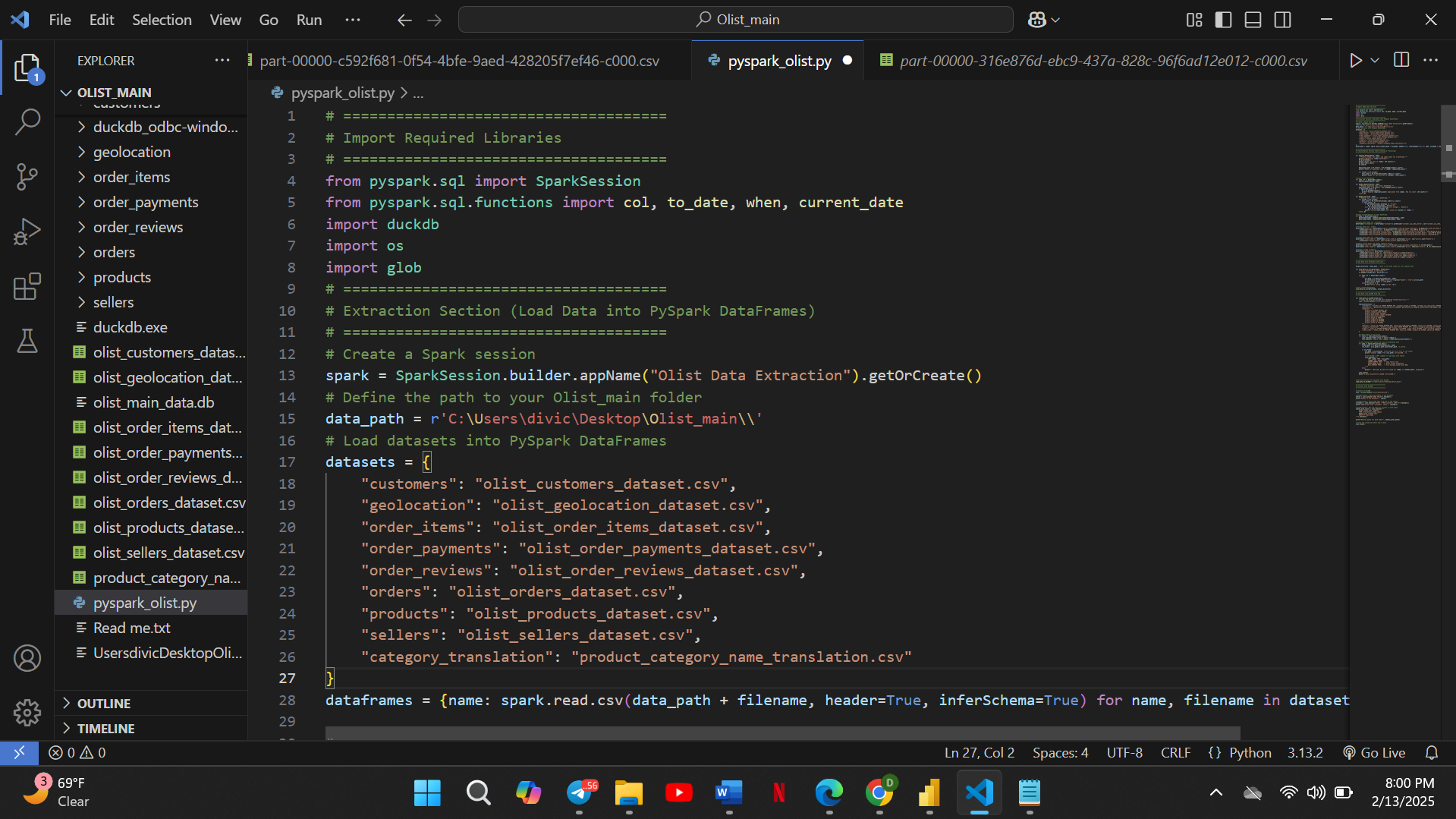
The ETL pipeline extracts raw data from multiple sources (databases or flat files), transforms the data to fit analytical requirements, and loads the processed data into a destination system (e.g., a data warehouse or Power BI).

**Pipeline Steps:**

1. **Extract:** Retrieve data from the raw data sources (e.g., databases, CSV files).
2. **Transform:** Clean and preprocess the data, including handling missing values, correcting data types, and aggregating information.
3. **Load:** Load the cleaned and transformed data into a destination system for analysis and reporting.

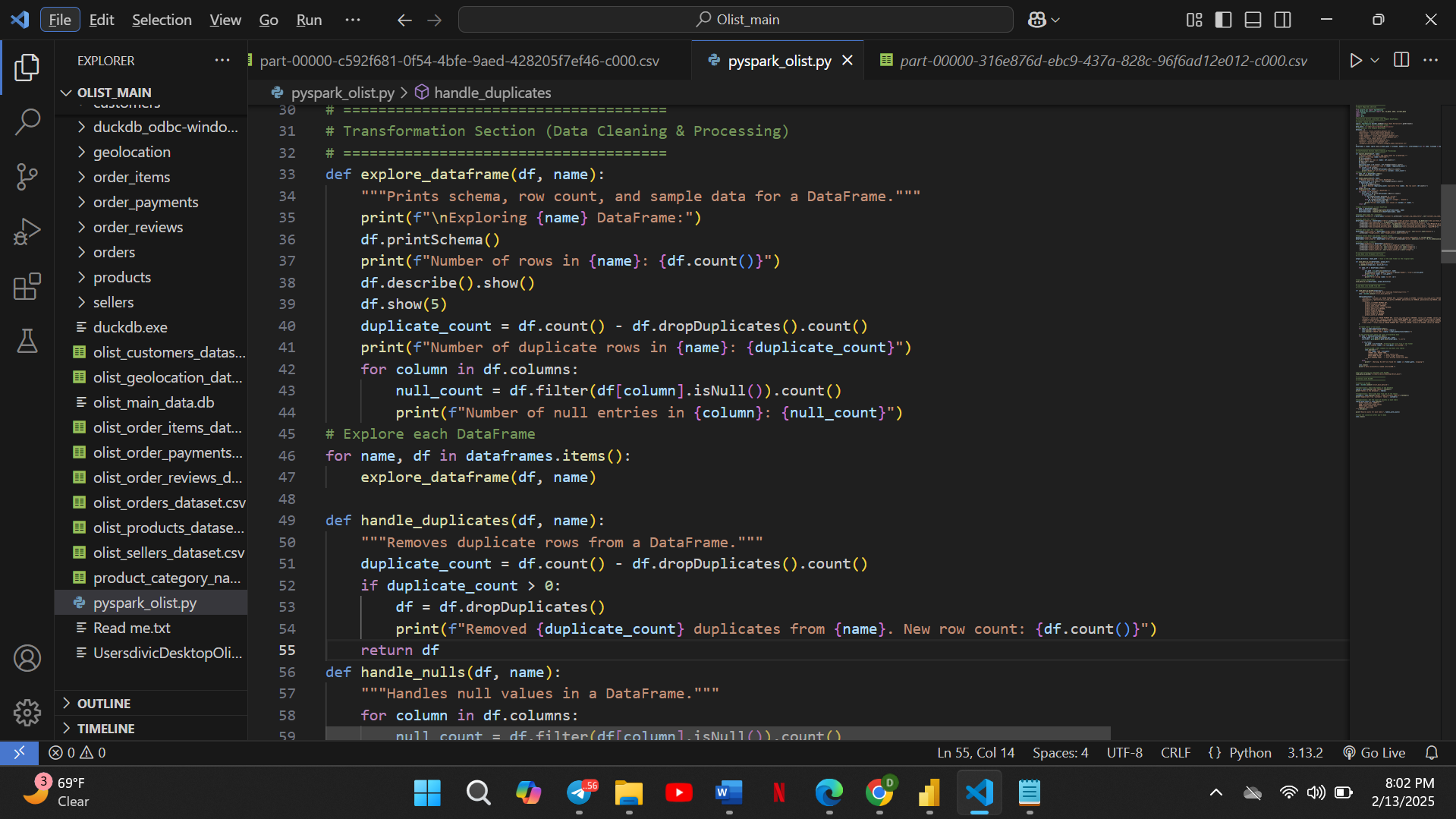
# 2. Data Extraction:

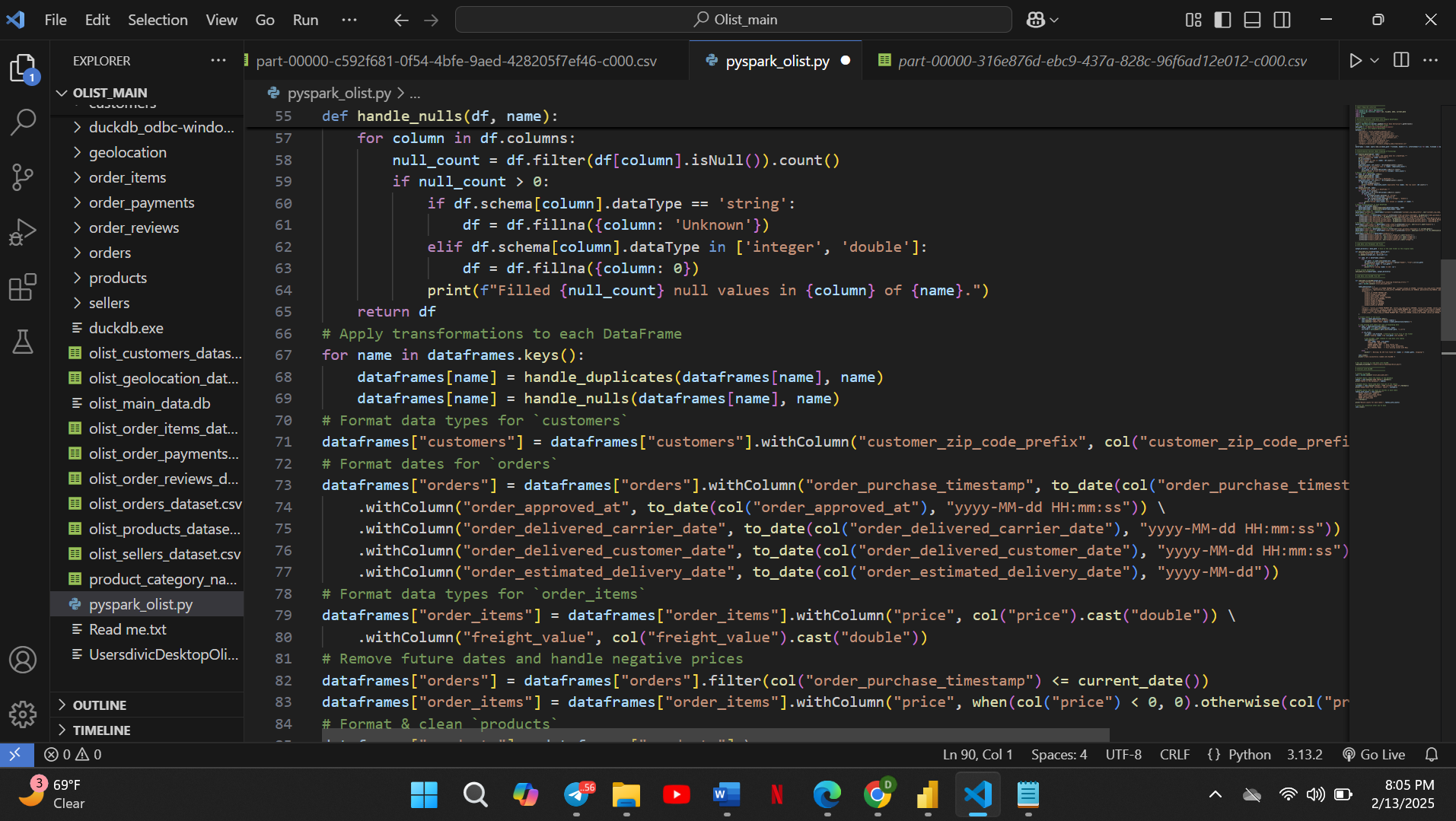
For the **data extraction** process, I used **PySpark** to load the **Olist** dataset, which contains over 1 million rows, into a DataFrame. PySpark's distributed processing capabilities allowed for efficient handling of this large dataset. After loading the CSV file into the DataFrame, I performed an initial check to ensure the data was correctly loaded by displaying the first few rows. This step allowed for a seamless transition from raw data to an analyzable format suitable for further processing and analysis.



# 3. Data Transformation:

For the **data transformation and cleaning** process, I applied several techniques to ensure the datasets were ready for analysis. First, I handled missing values by filling them with appropriate defaults or removing rows with excessive missing data. I also checked for and removed any **duplicates** to ensure the accuracy of the data. In terms of **data types**, I formatted columns like dates and numerical values to align with the appropriate formats, making them suitable for analysis. Additionally, I identified and corrected any **inconsistencies or errors**, such as invalid entries or misformatted data, to ensure data integrity throughout the pipeline. These steps were crucial for preparing the dataset for accurate and efficient analysis.

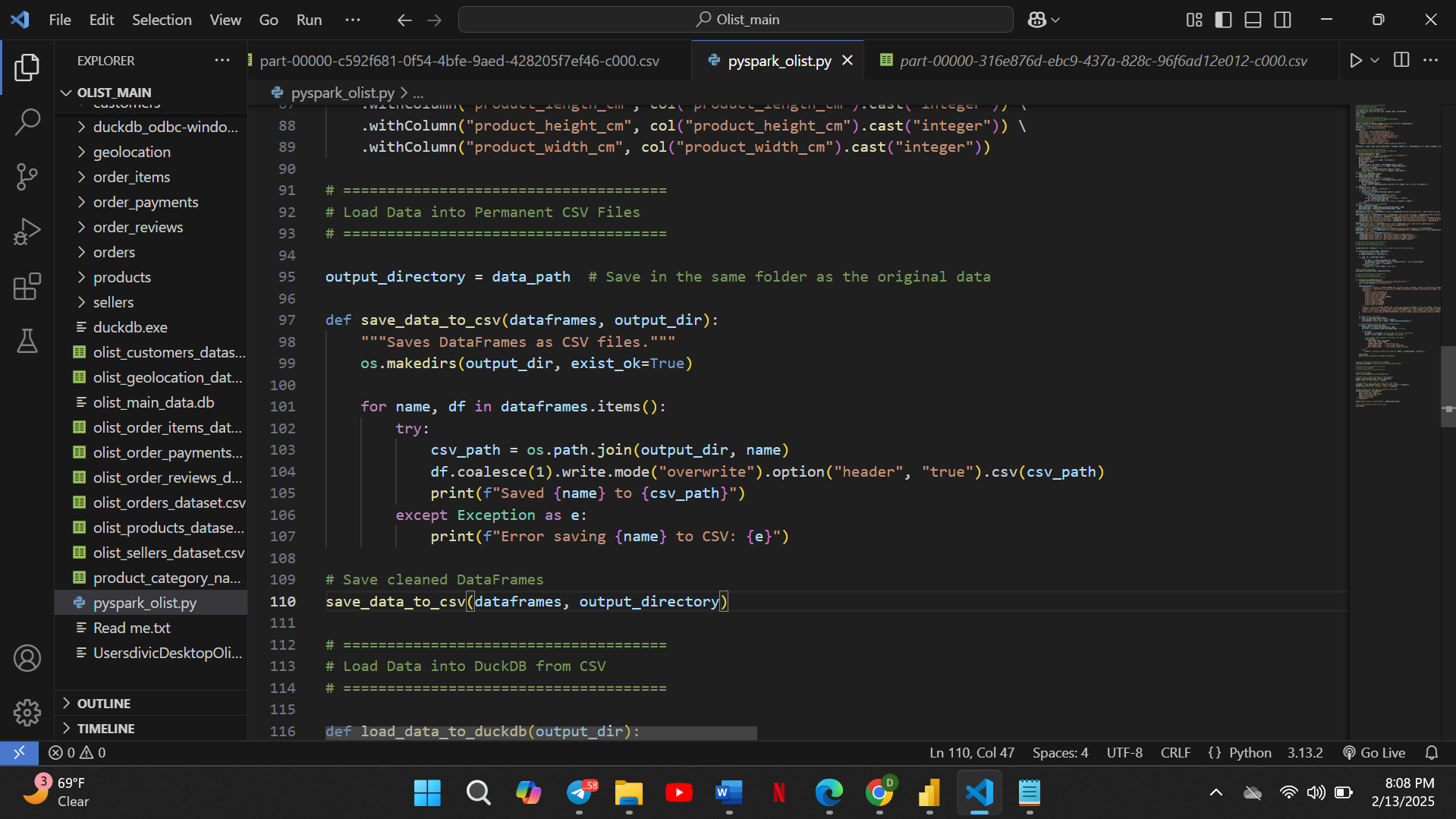
****

****

# Data Loading:

1. **Load Data into Permanent CSV Files**

In the **Data Loading** phase, I first saved the cleaned datasets into **permanent CSV files** for archival purposes and future reference. This ensured that the original data was safely stored and could be accessed if needed for future operations or verification.

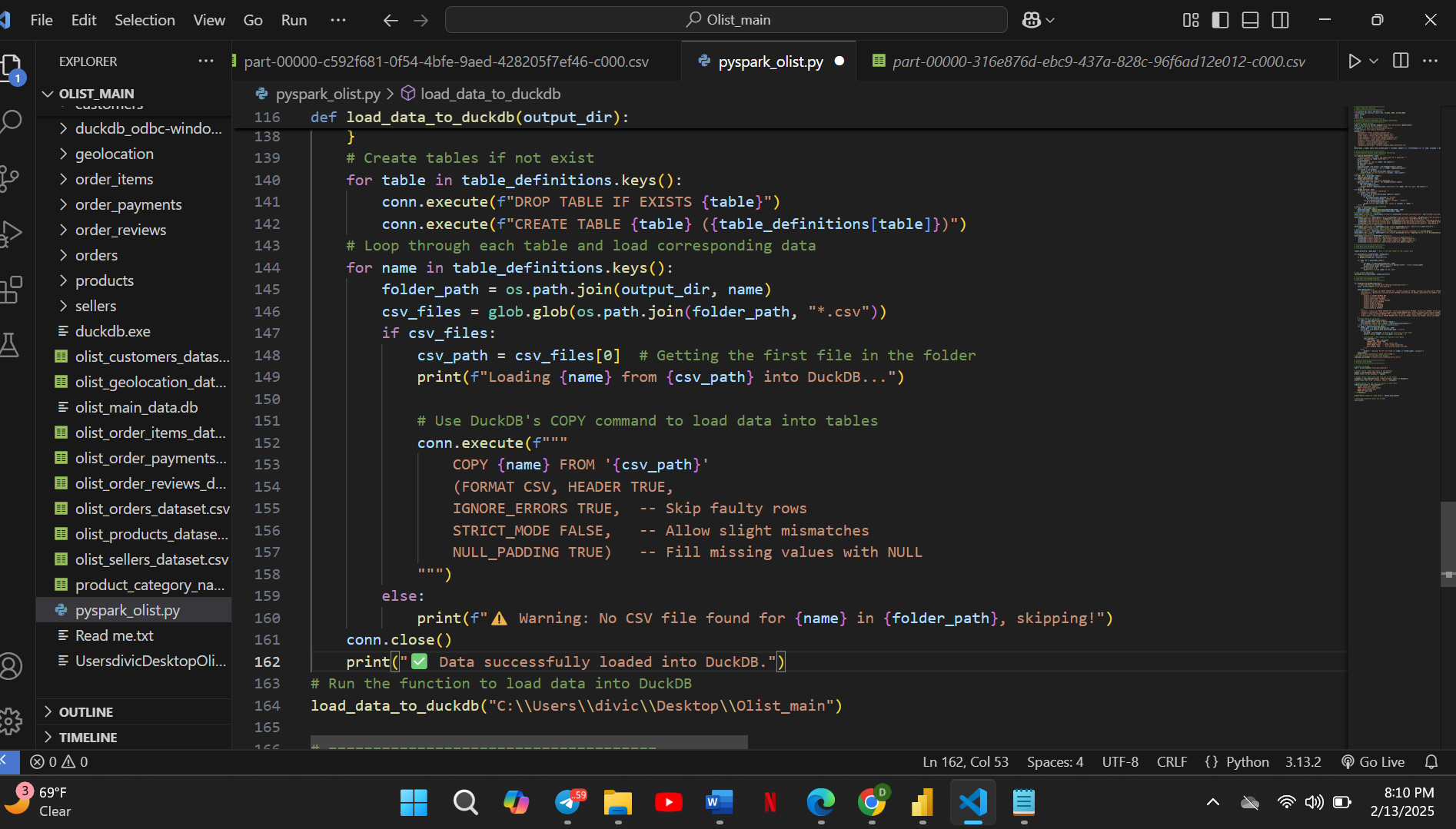
****

1. **Load Data into DuckDB from CSV**

I proceeded to load the data into **DuckDB**, a high-performance database, from the previously saved CSV files. DuckDB was chosen for its efficient handling of large datasets and its ability to process data without the need for a separate database server. I used appropriate SQL commands to import the data, and after loading, I verified the success of the operation by querying the data directly in DuckDB to ensure it had been stored correctly and was accessible for further analysis.

**A screenshot of a computer

AI-generated content may be incorrect.**

****

**Database Schema Overview with DuckDB**

The e-commerce data is stored in **DuckDB**, a high-performance, in-process SQL database management system. DuckDB was chosen due to its ability to efficiently handle analytical workloads and large datasets in an easy-to-use format, making it ideal for the project.

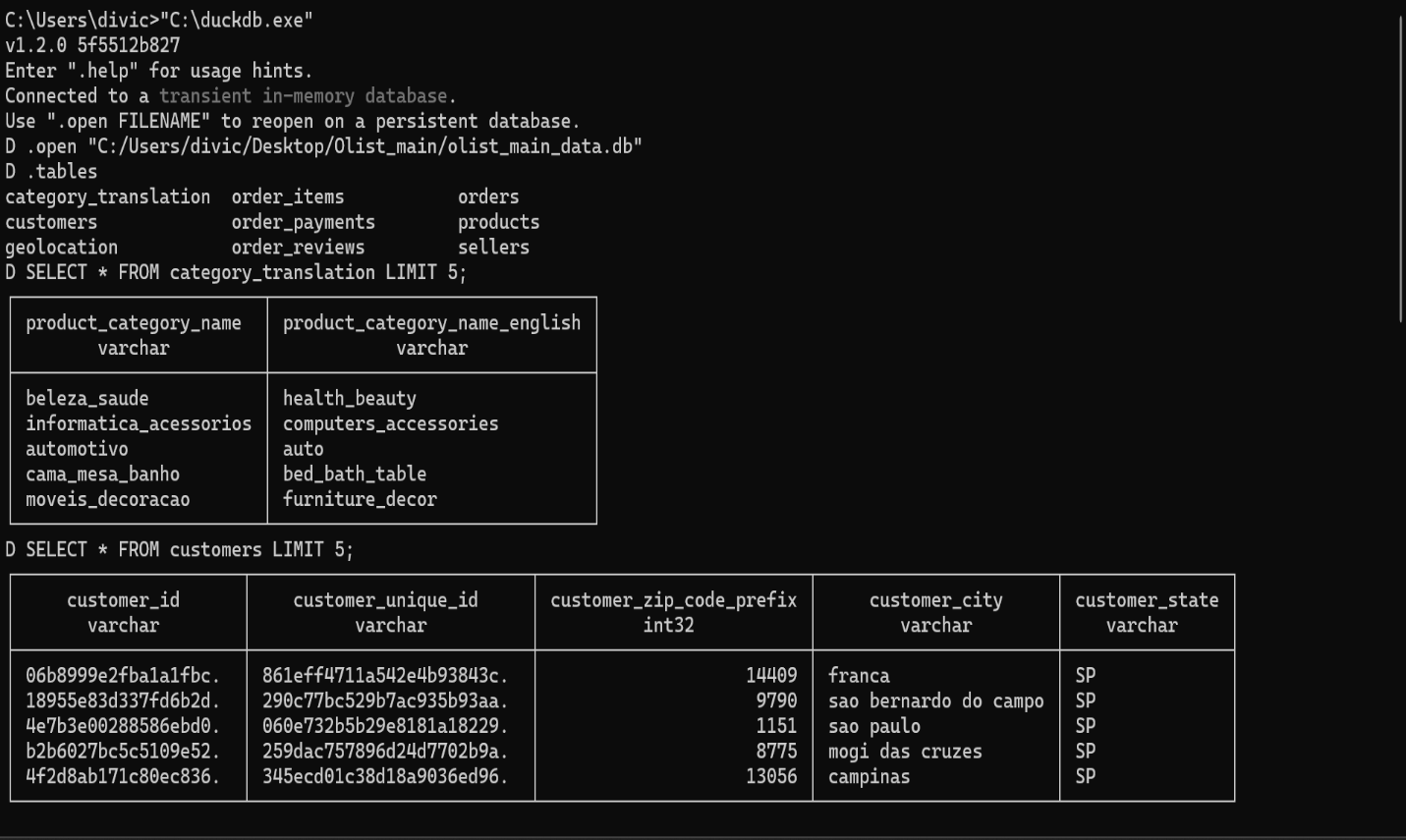
**Steps Taken for Data Storage and Verification:**

1. **Loading Data into DuckDB:**
   * Initially, the data was loaded from various sources (such as CSV or Excel files) into DuckDB tables. The data was organized into multiple tables: **Customers**, **Orders**, **Products**, **Sellers**, and **Geolocation**.
2. **Verifying Data Storage:**
   * After loading the data, I verified that it had been stored correctly by running several SQL queries to check for consistency and completeness. For each table, I executed SQL commands to ensure that the data matched the expected schema and that no records were missing or corrupted.
3. **SQL Commands for Verification:**
   * Here is an example of the SQL command used to check the contents of the **Customers** table:

**SELECT \* FROM customers LIMIT 5;**

* + This command retrieves the first 5 records from the customers table to verify that the data has been correctly loaded.

1. **Verification Output:**
   * The output of the SQL query was displayed in the DuckDB console, confirming that the records were correctly stored. To provide additional clarity, I have included an image of the SQL query result below, showing the retrieved data from the table:



**Key Database Schema Components:**

* **Customers Table:** Stores information about customers including personal details and contact info.
* **Orders Table:** Contains order details, referencing both customers and products.
* **Products Table:** Stores product information such as names, categories, and prices.
* **Sellers Table:** Holds details about sellers, their locations, and contact info.
* **Geolocation Table:** Stores geolocation data, including the city, state, country, and geographic coordinates.

The use of **DuckDB** ensures that the data can be queried efficiently, allowing for deep analytical insights. The data validation through SQL commands guarantees the accuracy and integrity of the dataset, which is critical for reporting and analysis.

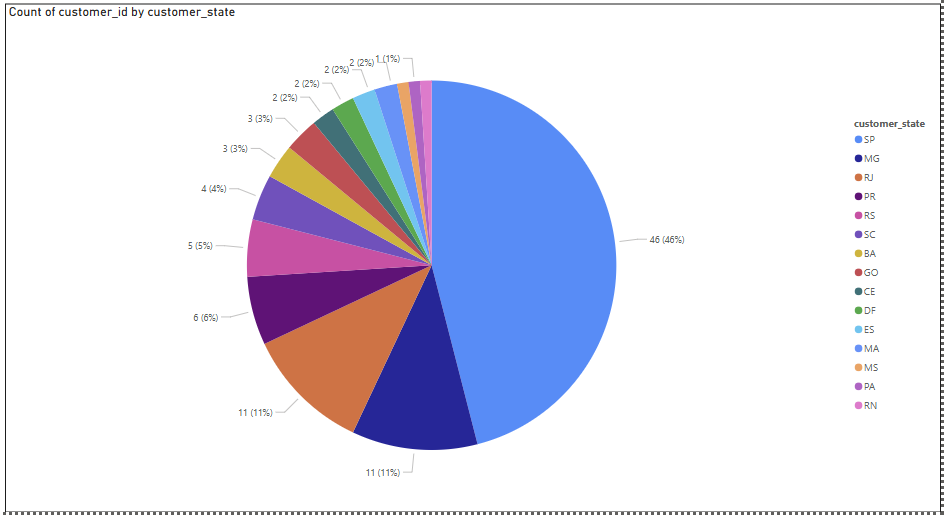
# Data Visualization and Insights:

In the **Data Visualization and Insights** phase, I utilized **Power BI** to visualize the data stored in DuckDB, allowing me to extract valuable insights from the large e-commerce dataset. To establish a connection between Power BI and DuckDB, I used a **Python script** that facilitated access to the tables stored in the DuckDB database. This integration enabled me to load the data directly into Power BI for visualization purposes.

The key visualizations created included various charts such as bar charts, pie charts, and line graphs to represent essential business metrics such as sales trends, customer segmentation, and product performance.

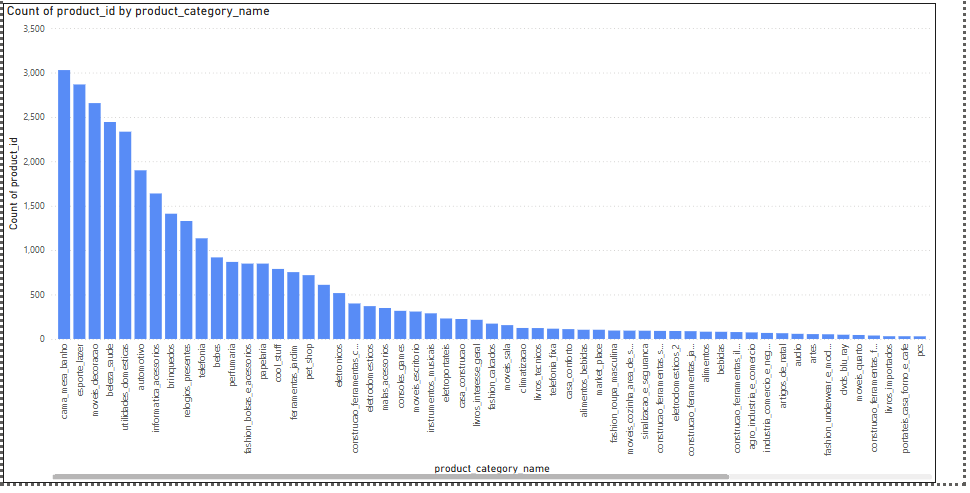
1. **Customer Distribution by State (Pie Chart)**

* **Goal:** To analyze the geographical distribution of customers based on their state.
* **Legend:** customer\_state
* **Values:** Count of customer\_id

****

1. **Product Distribution by Category (Stacked Column Chart)**

* **Goal:** To visualize the proportion of products across different categories, highlighting category dominance.
* **X-axis:** Product\_category\_name
* **Y-axis:** Count of product\_id



1. **Number of Sellers by State (Donut Chart)**

* **Goal:** To visualize the distribution of sellers across different states.
* **Legend:** seller\_state
* **Values:** Count of seller\_id

**A colorful circle with numbers

AI-generated content may be incorrect.**

**6. Design Choices:**

* **Geographic Insights:** Seller location data is crucial for understanding regional sales performance. By grouping sales by seller\_city or seller\_state, we can identify areas of high demand and tailor marketing efforts to those regions.
* **Revenue Analysis:** Revenue breakdown by product and payment method allows for identifying high-value products and preferred payment channels, which can help optimize product offerings and customer experience.
* **Top-N Product Analysis:** For performance analysis, the top N products by order count can be showcased to identify best-sellers and to ensure inventory management aligns with demand.

# Conclusion

**Optimization:** The pipeline provides a structured process for cleaning, transforming, and loading data into an analytical system. Regular maintenance and updates to data quality checks will ensure consistent results.