ETL Pipeline Development

Objective:

Design and implement an ETL pipeline that extracts data from multiple CSV files, applies transformations based on business rules, and loads the data into a PostgreSQL database. The pipeline should also include robust error handling and logging.

Process	Tech Stack Used
ETL Process	Python (including libraries such as pandas, numpy and psycopg2)
SQL Loading	PostgreSQL

Setup Library & Installation

Python Libraries

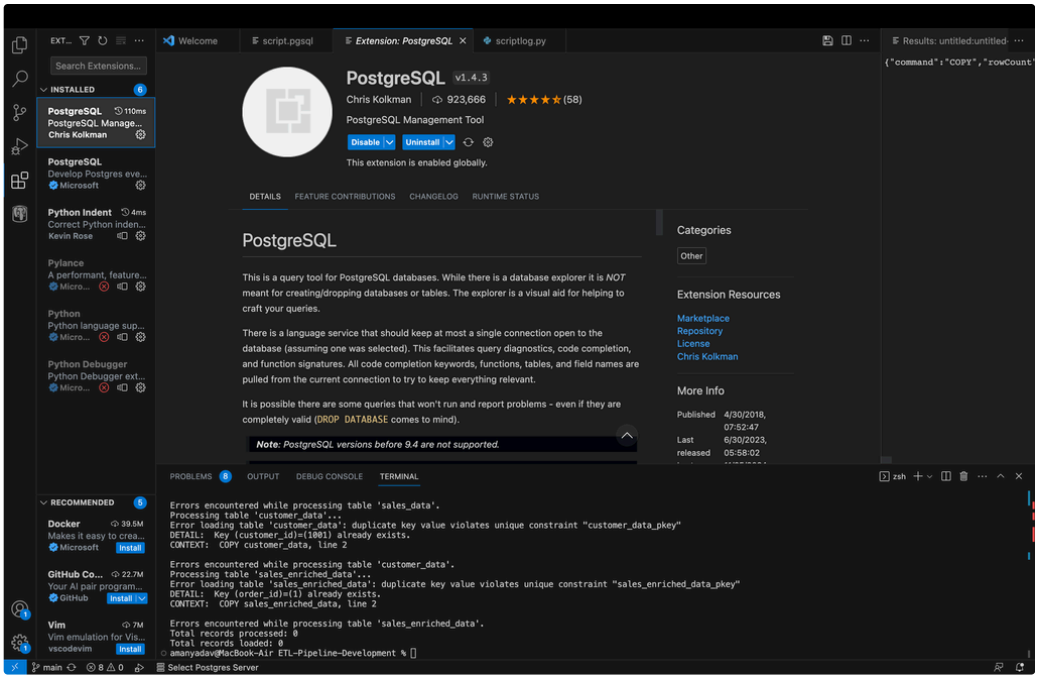
Start by installing python library like pandas, numpy and pycopg2 using the command line or terminal:

1. `pip3 install pandas`
2. `pip3 install numpy`
3. `pip3 install pycopg2`

PostgreSQL Installation

Download the PostgreSQL by clicking here and choosing your appropriate OS configuration.

[Optional] I have also installed the extension PostgreSQL by ChrisKolkman in VSCode since I'll be connecting the database using the VSCode



Steps

In the provided document after installing the necessary libraries open the **pipeline.py** file in your preferred code editor.

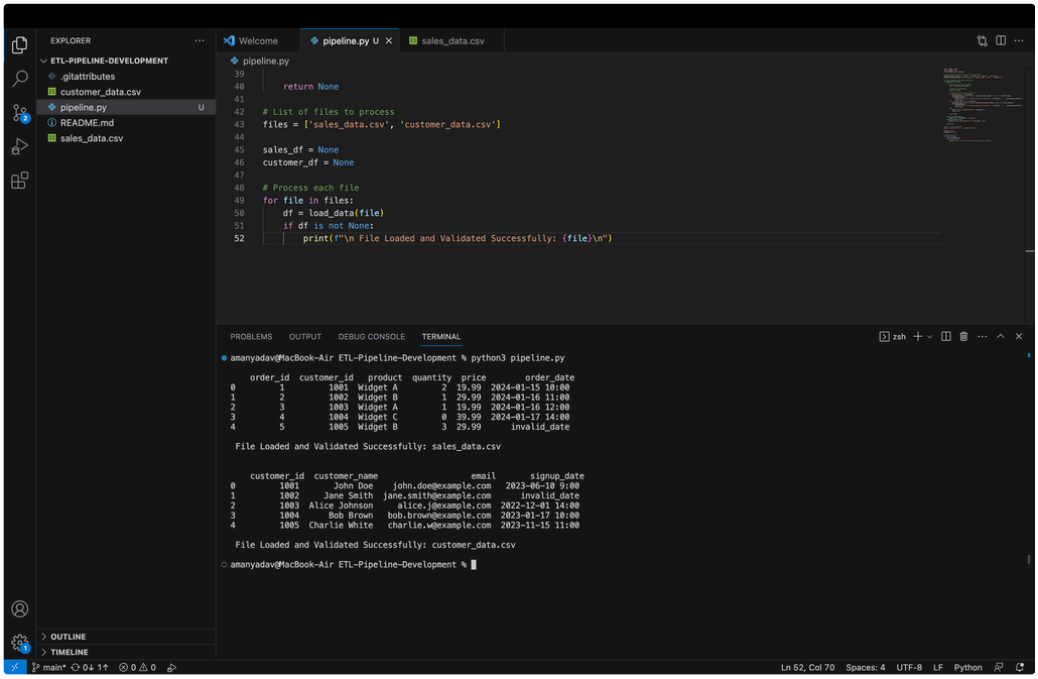
For demonstration purpose I have used Visual Studio Code. Once the pipeline.py file is open run it through command line/Terminal using `python3 pipeline.py` (Could be different depending on your file path I was already in the project fold so I have used the the provided terminal script)

In the first function in my python file I have extracted and validated the **sales_data.csv** and **customer_data.csv** file using the load_data() function

In order to store the dataset I have used **pandas dateframe**

```
def load_data(file_path):
    try:
        # Load the data into a DataFrame
        data = pd.read_csv(file_path)

        # Log data for debugging
        #print("\n",data)

        # Validate columns based on file type
        if 'sales_data.csv' in file_path:
            missing_columns = [col for col in required_sales_columns if col not in data.columns]
            if missing_columns:
                print(f"Error: The following columns are missing in '{file_path}': {',
'.join(missing_columns)}")
                return None
        elif 'customer_data.csv' in file_path:
            missing_columns = [col for col in required_customer_columns if col not in data.columns]
            if missing_columns:
                print(f"Error: The following columns are missing in '{file_path}': {',
'.join(missing_columns)}")
                return None
        else:
```

```
            print(f"Error: Unsupported file '{file_path}'")
            return None

        return data

    except FileNotFoundError:
        print(f"File {file_path} is missing")
    except Exception as e:
        print(f"Error loading data from '{file_path}': {e}")

    return None
```

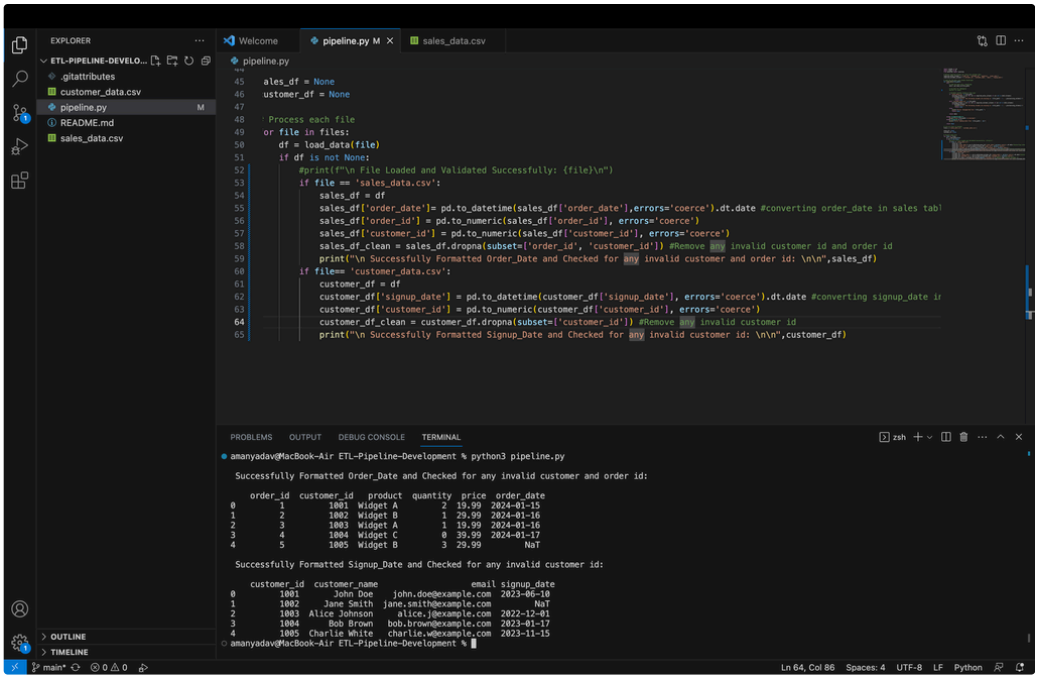


Extracting and validating the dataset

Once the data has been extracted I have performed transformation on the dataset using numpy and pandas.

Data Cleaning:

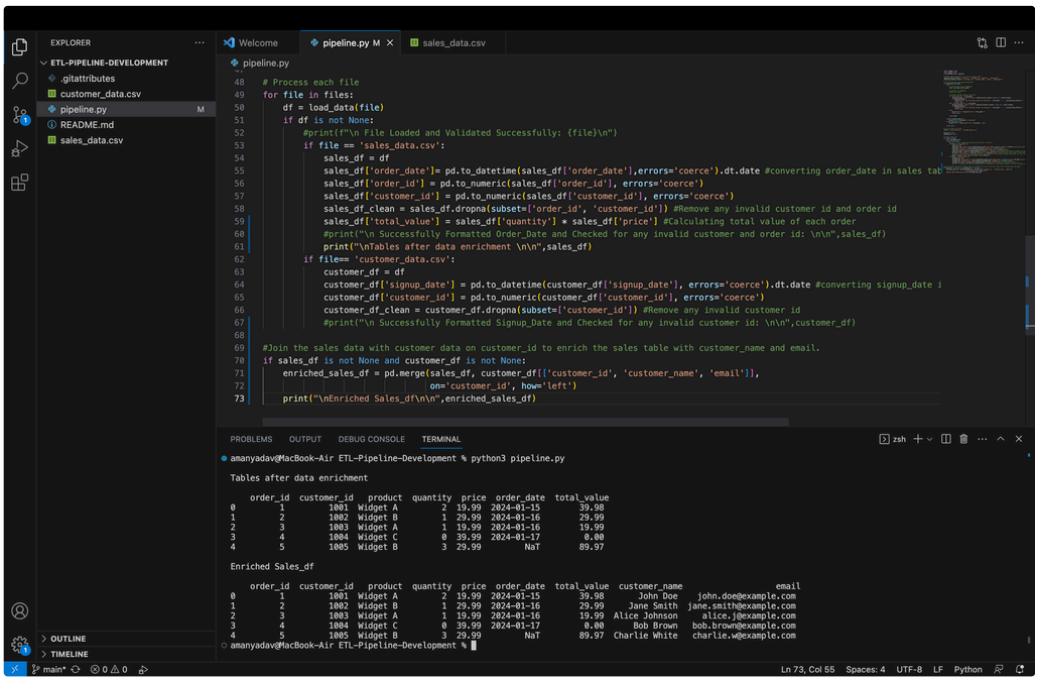
- Converted order_date and signup_date to a standard format (YYYY-MM-DD). Log errors for invalid dates and skip affected records.
- Remove any rows with missing or invalid customer_id or order_id.



Order_date and signup_date in YYYY-MM-DD format

Data Enrichment:

- Calculated the total value for each order (total_value = quantity * price).
- Joined the sales data with customer data on customer_id to enrich the sales table with customer_name and email.



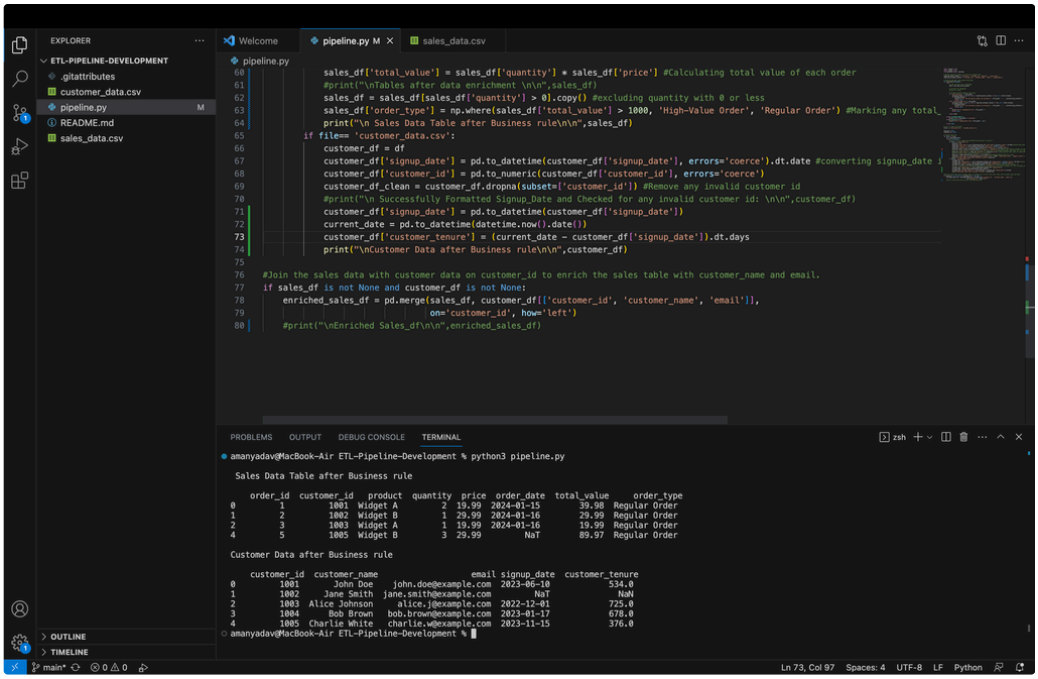
Data Enrichment

Business Logic:

- Excluded orders with a quantity of 0 or less.

-Marked orders with total values exceeding $1000 as "High-Value Orders" in a new column order_type.

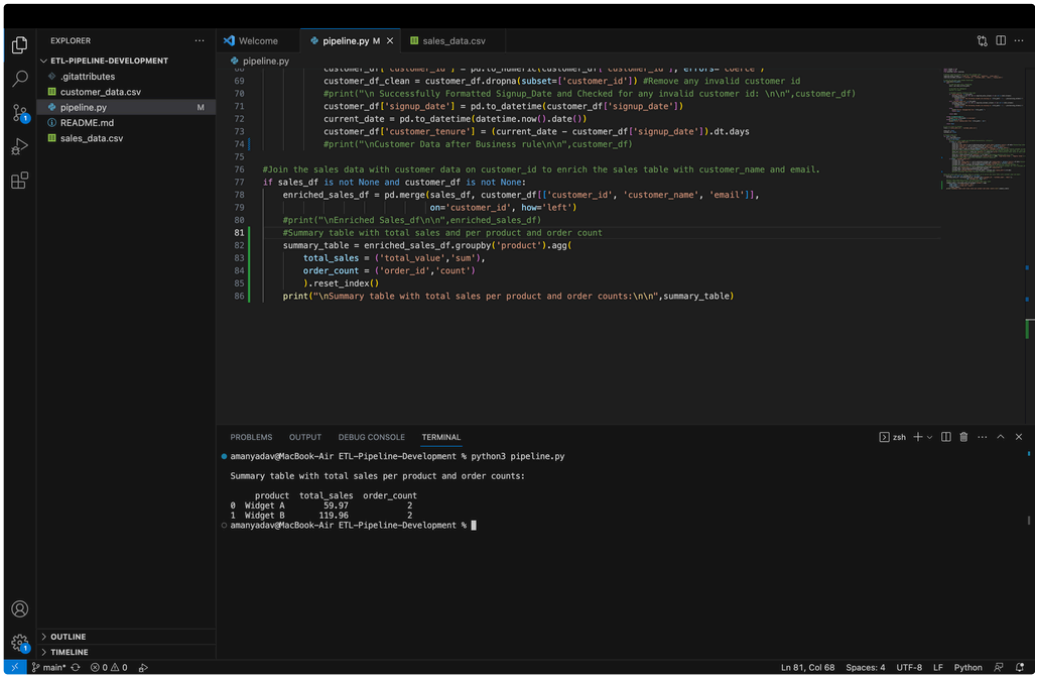
-Add a column customer_tenure representing the number of days since the customer's signup date.



Business Rule

Data Aggregation:

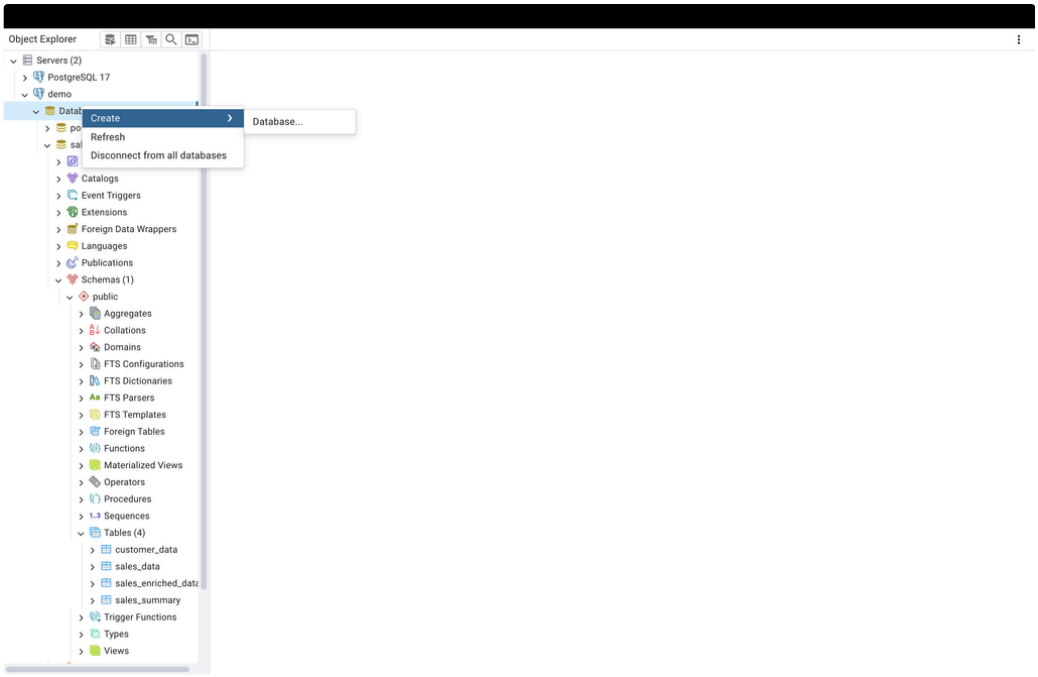
-Create a summary table with total sales (SUM(total_value)) per product and order counts (COUNT(order_id)).



Data Aggregation

Once the transformation has performed, as I already install PostgreSQL in my VSCode I will be using that to make the connection between my dataset.

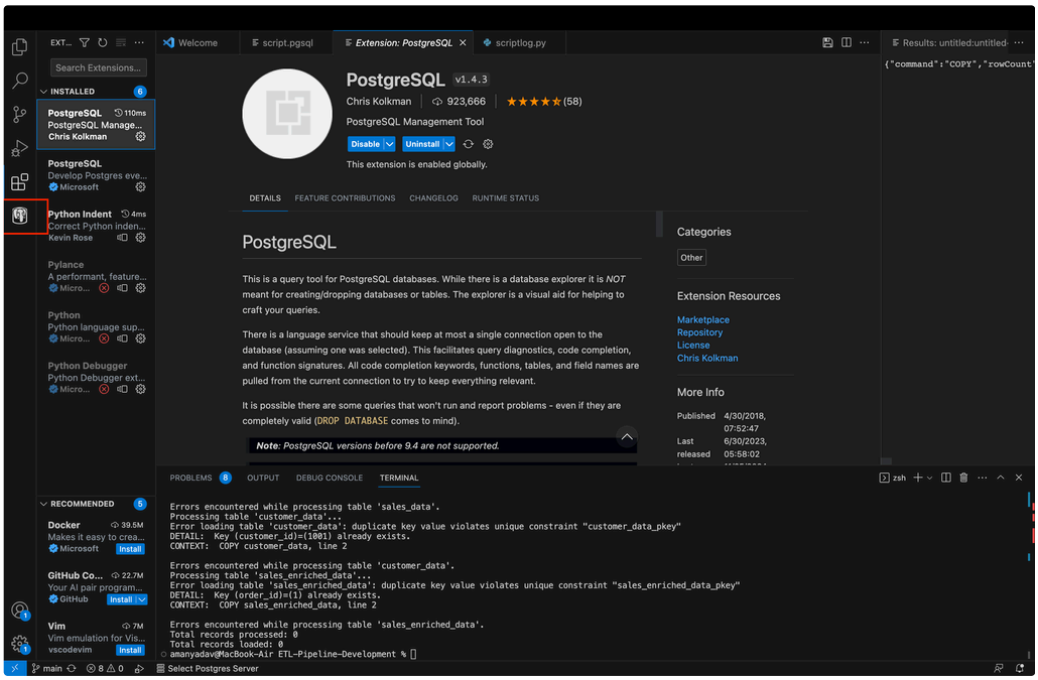
First we will create a database using PgAdmin 4.



Creating database

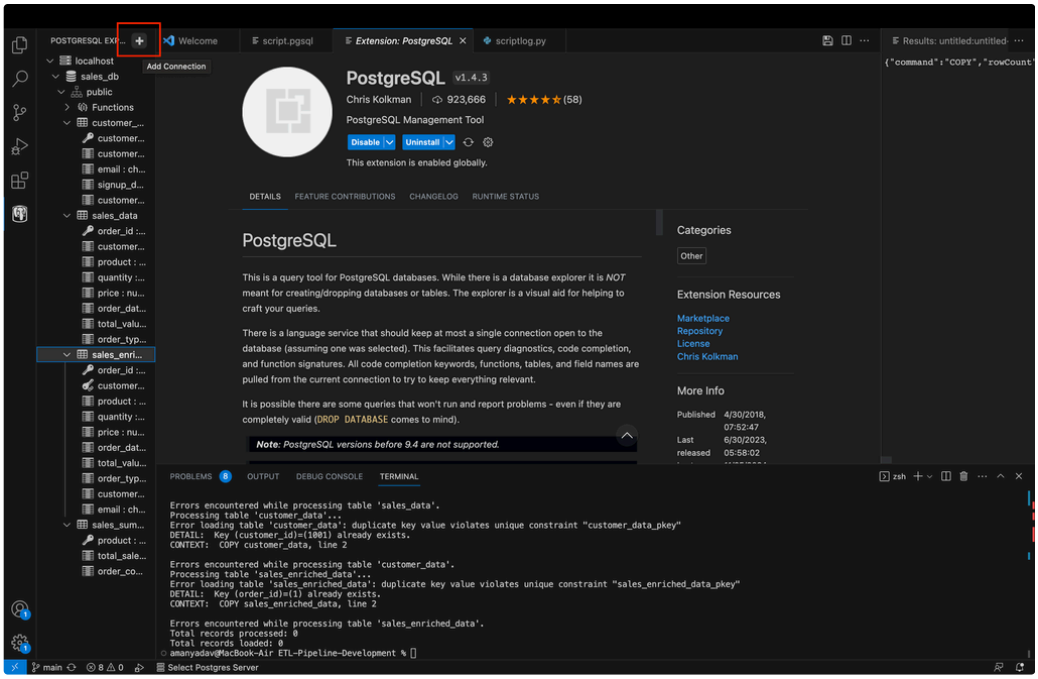
I already have created a database in name of sales_db.

Now back to VS Code by clicking on the PostgreSQL logo in the VSCode I will make the database connection



Creating connection

Once the PostgreSQL tab is open click on the “+” and choose your server name → enter your username → password → select database



After this I've created the PostgreSQL script to create table

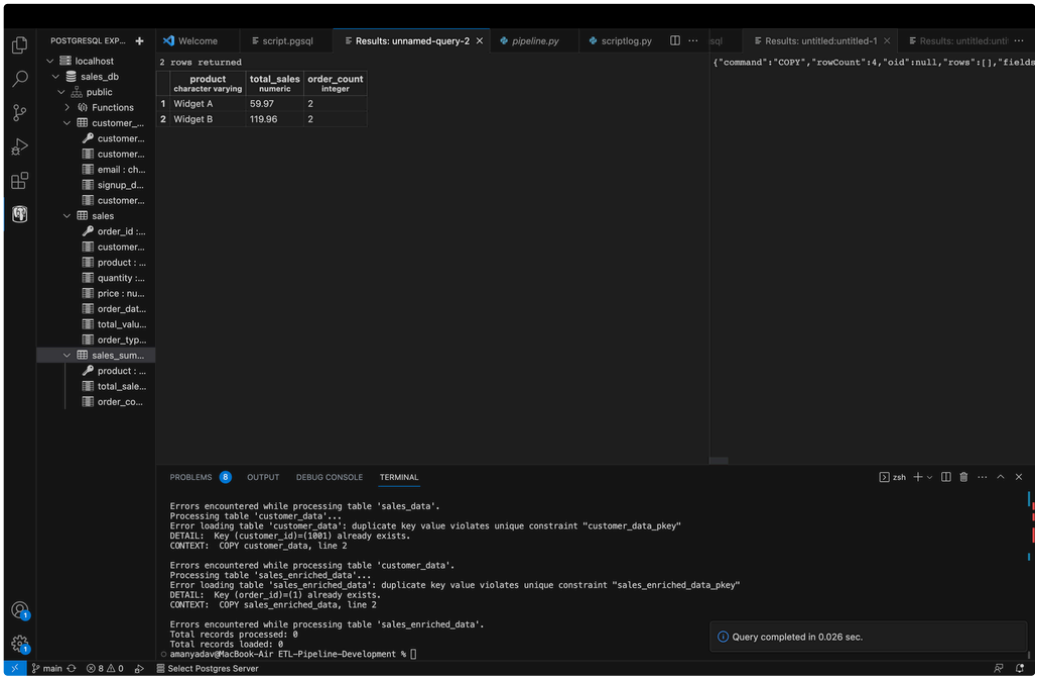
```sql
-- Create the sales table
CREATE TABLE sales_data (
    order_id SERIAL PRIMARY KEY,
    customer_id INT NOT NULL,
    product VARCHAR(255) NOT NULL,
    quantity INT NOT NULL,
    price DECIMAL(10, 2) NOT NULL,
    order_date DATE,
    total_value DECIMAL(10, 2) NOT NULL,
    order_type VARCHAR(50)

);

-- Create the sales_summary table
CREATE TABLE sales_summary (
    product VARCHAR(255) PRIMARY KEY,
    total_sales DECIMAL(10, 2) NOT NULL,
    order_count INT NOT NULL
);

--Create customer_data table
CREATE TABLE customer_data (
    customer_id SERIAL PRIMARY KEY,
    customer_name VARCHAR(255),
    email VARCHAR(255),
    signup_date DATE,
    customer_tenure VARCHAR(255) DEFAULT NULL
);

--Create the sales_enriched_data
CREATE TABLE sales_enriched_data (
    order_id SERIAL PRIMARY KEY,
    customer_id INT NOT NULL,
    product VARCHAR(255) NOT NULL,
    quantity INT NOT NULL,
```

```
    price DECIMAL(10, 2) NOT NULL,
    order_date DATE,
    total_value DECIMAL(10, 2) NOT NULL,
    order_type VARCHAR(50),
    customer_name VARCHAR(255),
    email VARCHAR(255),
    CONSTRAINT fk_customer FOREIGN KEY (customer_id) REFERENCES customer_data(customer_id) ON DELETE CASCADE
);
```

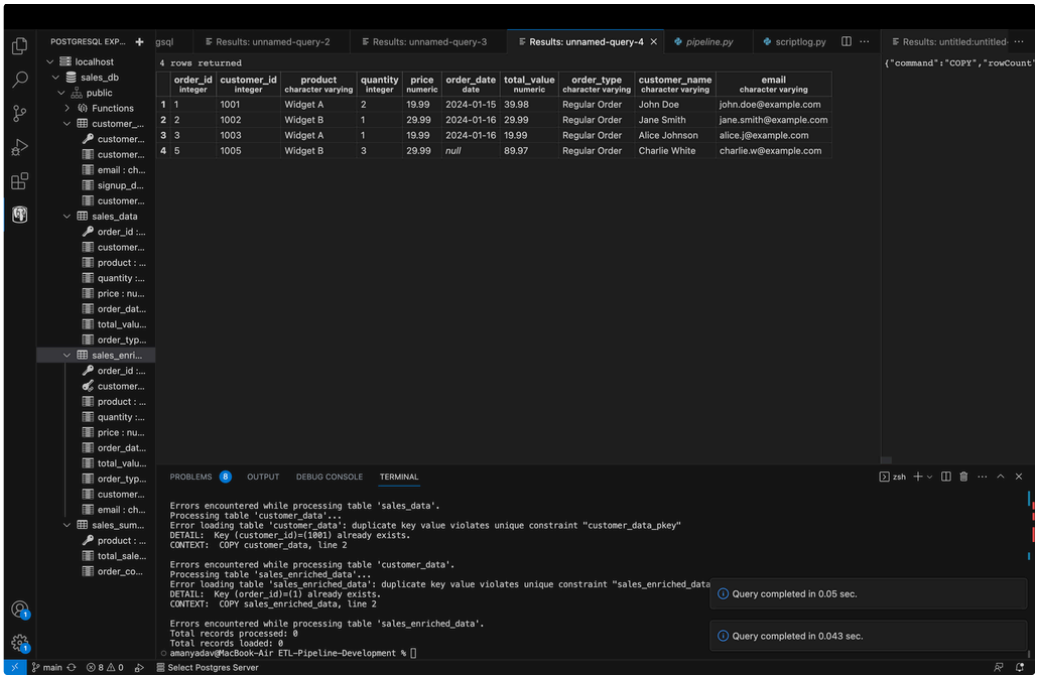
Once the table has created I have used copy command to copy the transformed dataset to load in the table which is created.

```
--Copied the data from the csv file to our table
COPY sales_summary FROM '/private/tmp/summary_table.csv'
DELIMITER ','
CSV HEADER
ENCODING 'UTF8';

COPY sales_data FROM '/private/tmp/transformed_sales_data.csv'
DELIMITER ','
CSV HEADER
ENCODING 'UTF8';

COPY customer_data FROM '/private/tmp/transformed_customer_data.csv'
DELIMITER ','
CSV HEADER
ENCODING 'UTF8';

COPY sales_enriched_data FROM '/private/tmp/enriched_sales_df.csv'
DELIMITER ','
CSV HEADER
ENCODING 'UTF8';
```

Once the transferred has performed I have run the query to view the columns



Summary Table



Sales Data Table

Challenges/Errors Faced	Resolution
Was having issue where column was missing/mismatch in sales_data.csv	Checked the error in the output terminal turned out to there was typo while creating the orginal dataset file
While validating the columns, columns were bring crossed check with both the dataset meaning load_data() function was also checking for missing sales_data columns in the customer_data and vice-versa	Implemented if condition to avoid that error.
While formatting date into YYYY-MM-DD format was unable to format the invalid_date.	Used `errors='coerce'` while converting date into YYYY-MM-DD Format so that invalid data will be set to NaN
While copying the dataset to PostgreSQL was having read access permission even though there was read permission given	1. moved the transformed file into tmp folder and then copied the data. 2. Directly exported from PgAdmin 4.