In [6]:
```
pip install seaborn
```

```
Requirement already satisfied: seaborn in c:\users\neha\appdata\local\programs\py
thon\python313\lib\site-packages (0.13.2)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in c:\users\neha\appdata\loca
l\programs\python\python313\lib\site-packages (from seaborn) (2.2.4)
Requirement already satisfied: pandas>=1.2 in c:\users\neha\appdata\local\program
s\python\python313\lib\site-packages (from seaborn) (2.2.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in c:\users\neha\appdata\l
ocal\programs\python\python313\lib\site-packages (from seaborn) (3.10.1)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\neha\appdata\local\pr
ograms\python\python313\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn)
(1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\users\neha\appdata\local\progra
ms\python\python313\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (0.
12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\neha\appdata\local\p
rograms\python\python313\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seabor
n) (4.56.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\neha\appdata\local\p
rograms\python\python313\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seabor
n) (1.4.8)
Requirement already satisfied: packaging>=20.0 in c:\users\neha\appdata\local\pro
grams\python\python313\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn)
(24.2)
Requirement already satisfied: pillow>=8 in c:\users\neha\appdata\local\programs
\python\python313\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (11.
1.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\neha\appdata\local\pr
ograms\python\python313\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn)
(3.2.3)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\neha\appdata\loca
l\programs\python\python313\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seab
orn) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\neha\appdata\local\progra
ms\python\python313\lib\site-packages (from pandas>=1.2->seaborn) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in c:\users\neha\appdata\local\prog
rams\python\python313\lib\site-packages (from pandas>=1.2->seaborn) (2025.2)
Requirement already satisfied: six>=1.5 in c:\users\neha\appdata\local\programs\p
ython\python313\lib\site-packages (from python-dateutil>=2.7->matplotlib!=3.6.1,>
=3.4->seaborn) (1.17.0)
Note: you may need to restart the kernel to use updated packages.
```

In [9]:
```
import pandas as pd
import mysql.connector
import os

# List of CSV files and their corresponding table names
csv_files = [
    ('customers.csv', 'customers'),
    ('orders.csv', 'orders'),
    ('sellers.csv', 'sales'),
    ('products.csv', 'products'),
    ('order_items.csv', 'delivery'),
    ('geolocation.csv', 'geolocation'),
    ('payments.csv', 'payments')  # Added payments.csv for specific handling
]

# Connect to the MySQL database
```

```
conn = mysql.connector.connect(
    host='127.0.0.1',
    user='root',
    password='Bittu_N@098',
    database='ecommerce'
)
cursor = conn.cursor()

# Folder containing the CSV files
folder_path = 'C:/Users/Neha/Desktop/portfolio projects done yet/Ecommerce'

def get_sql_type(dtype):
    if pd.api.types.is_integer_dtype(dtype):
        return 'INT'
    elif pd.api.types.is_float_dtype(dtype):
        return 'FLOAT'
    elif pd.api.types.is_bool_dtype(dtype):
        return 'BOOLEAN'
    elif pd.api.types.is_datetime64_any_dtype(dtype):
        return 'DATETIME'
    else:
        return 'TEXT'

for csv_file, table_name in csv_files:
    file_path = os.path.join(folder_path, csv_file)

    # Read the CSV file into a pandas DataFrame
    df = pd.read_csv(file_path)

    # Replace NaN with None to handle SQL NULL
    df = df.where(pd.notnull(df), None)

    # Debugging: Check for NaN values
    print(f"Processing {csv_file}")
    print(f"NaN values before replacement:\n{df.isnull().sum()}\n")

    # Clean column names
    df.columns = [col.replace(' ', '_').replace('-', '_').replace('.', '_') for

    # Generate the CREATE TABLE statement with appropriate data types
    columns = ', '.join([f'`{col}` {get_sql_type(df[col].dtype)}' for col in df.
    create_table_query = f'CREATE TABLE IF NOT EXISTS `{table_name}` ({columns})
    cursor.execute(create_table_query)

    # Insert DataFrame data into the MySQL table
    for _, row in df.iterrows():
        # Convert row to tuple and handle NaN/None explicitly
        values = tuple(None if pd.isna(x) else x for x in row)
        sql = f"INSERT INTO `{table_name}` ({', '.join(['`' + col + '`' for col
        cursor.execute(sql, values)

    # Commit the transaction for the current CSV file
    conn.commit()

# Close the connection
conn.close()
```

```
Processing customers.csv
NaN values before replacement:
customer_id                 0
customer_unique_id          0
customer_zip_code_prefix    0
customer_city               0
customer_state              0
dtype: int64

Processing orders.csv
NaN values before replacement:
order_id                            0
customer_id                         0
order_status                        0
order_purchase_timestamp            0
order_approved_at                 160
order_delivered_carrier_date     1783
order_delivered_customer_date    2965
order_estimated_delivery_date       0
dtype: int64

Processing sellers.csv
NaN values before replacement:
seller_id                 0
seller_zip_code_prefix    0
seller_city               0
seller_state              0
dtype: int64

Processing products.csv
NaN values before replacement:
product_id                      0
product category              610
product_name_length           610
product_description_length    610
product_photos_qty            610
product_weight_g                2
product_length_cm               2
product_height_cm               2
product_width_cm                2
dtype: int64

Processing order_items.csv
NaN values before replacement:
order_id               0
order_item_id          0
product_id             0
seller_id              0
shipping_limit_date    0
price                  0
freight_value          0
dtype: int64

Processing geolocation.csv
NaN values before replacement:
geolocation_zip_code_prefix    0
geolocation_lat                0
geolocation_lng                0
geolocation_city               0
geolocation_state              0
```

```
dtype: int64

Processing payments.csv
NaN values before replacement:
order_id                0
payment_sequential      0
payment_type            0
payment_installments    0
payment_value           0
dtype: int64
```

In [ ]:

In [ ]: