2025/2026

REPORT

Business Intelligence And Database Management Systems

TUNIS BUSINESS SCHOOL
UNIVERSITY OF TUNIS

SALES PERFORMANCE & FORECASTING BI SOLUTION FOR A RETAIL SUPERSTORE

Professor:
PROF. Manel Abdelkader

Presented by :
Mariem Soltani
Rihem Abassi

Table of contents

Tools and Software used

Introduction

This Business Intelligence project transforms raw sales data from a nationwide retail superstore into actionable business insights. We're building an interactive analytics platform that helps stakeholders understand sales performance, customer behavior, and market trends across different regions and product categories.

Industry and Organization Description

- **IIndustry**: Retail/E-commerce
- **Organization**: A medium-sized retail superstore operating
- **Business Model**: B2C sales of office supplies, furniture, and technology products
- **Customer Segments**: Consumer, Corporate, and Home Office
- **Geographic Coverage**: Nationwide coverage across 4 regions (East, West, South, Central)

Use Case Understanding

The Problem

The company has years of sales data but lacks a unified view of performance. Managers spend hours extracting insights from spreadsheets, making it difficult to spot trends, identify opportunities, or respond quickly to market changes.

Our Solution

We're creating a comprehensive BI dashboard that:

- Tracks key sales metrics in real-time
- Identifies top-performing products and regions
- Analyzes customer buying patterns
- Optimizes shipping and operations
- Supports data-driven decision making

This will help increase sales, reduce costs, and improve customer satisfaction by providing clear, visual insights that anyone in the organization can understand and use to make better decisions.

Analytical Questions

1. **Sales Performance**: What are the monthly and yearly sales trends?
2. **Top Products**: Which products and categories generate the highest revenue?
3. **Customer Value**: Which customer segments are most valuable in terms of average order value?
4. **Regional Analysis**: How do sales vary across different regions and states?
5. **Shipping Efficiency:** What is the relationship between shipping mode and sales/profitability?
6. **Seasonal Patterns**: Are there specific months or seasons with consistently higher sales?
7. **Customer Loyalty**: Which customers make repeat purchases and what is their lifetime value?
8. **Product Category Mix**: What is the revenue distribution across furniture, office supplies, and technology?
9. **Sales by Day of Week**: Are certain days of the week more profitable?
10. **Corporate vs Consumer**: How do buying patterns differ between corporate and consumer segments?
11. **Growth Opportunities**: Which underperforming products/categories have growth potential?

KPIs

1. **Total Sales Revenue**: Sum of all sales
2. **Number of orders**: Sum of all orders
3. **Average Order Value (AOV)**: Total sales / Number of orders
4. **Customer Lifetime Value (CLV)**: Average revenue per customer
5. **Sales Growth Rate**: Month-over-month or year-over-year sales growth
6. **Regional Contribution**: Percentage of sales by region
7. **Product Category Mix**: Revenue distribution across categories
8. **Customer Segment Contribution**: Revenue by customer segment
9. **Shipping Mode Efficiency**: Sales by shipping mode with associated costs

Implementation

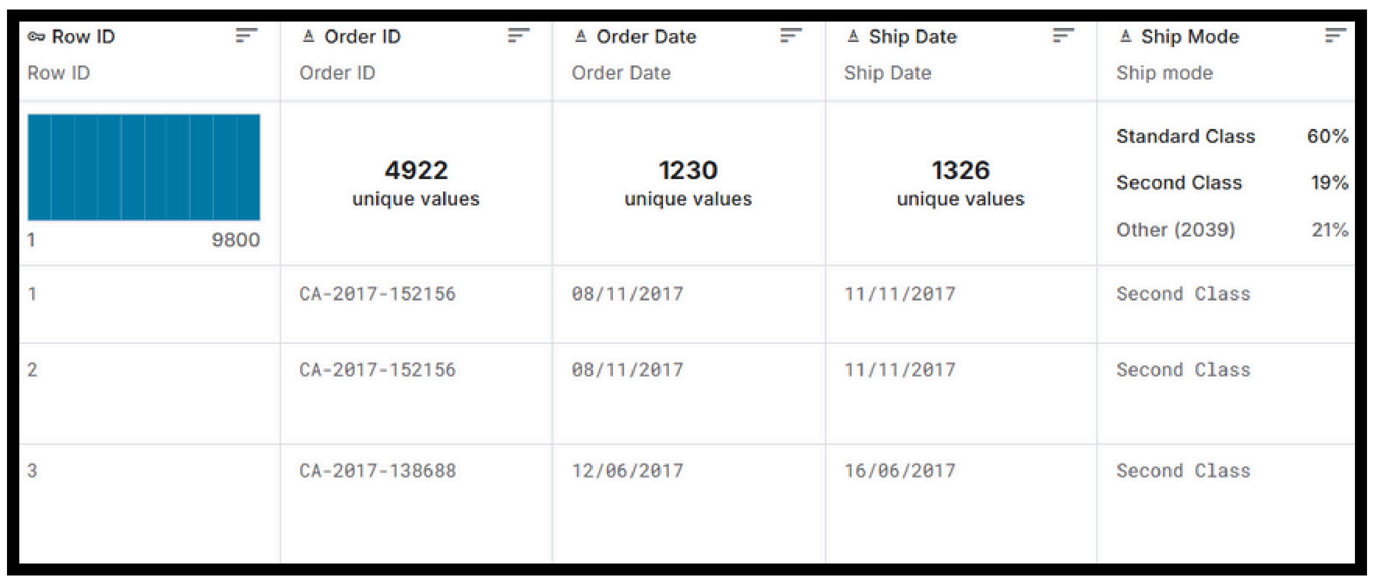
Phase 1 : Data gathering

Step 1 :Data collection

Source : Kaggle “Superstore Sales Dataset” by Rohit Sahoo

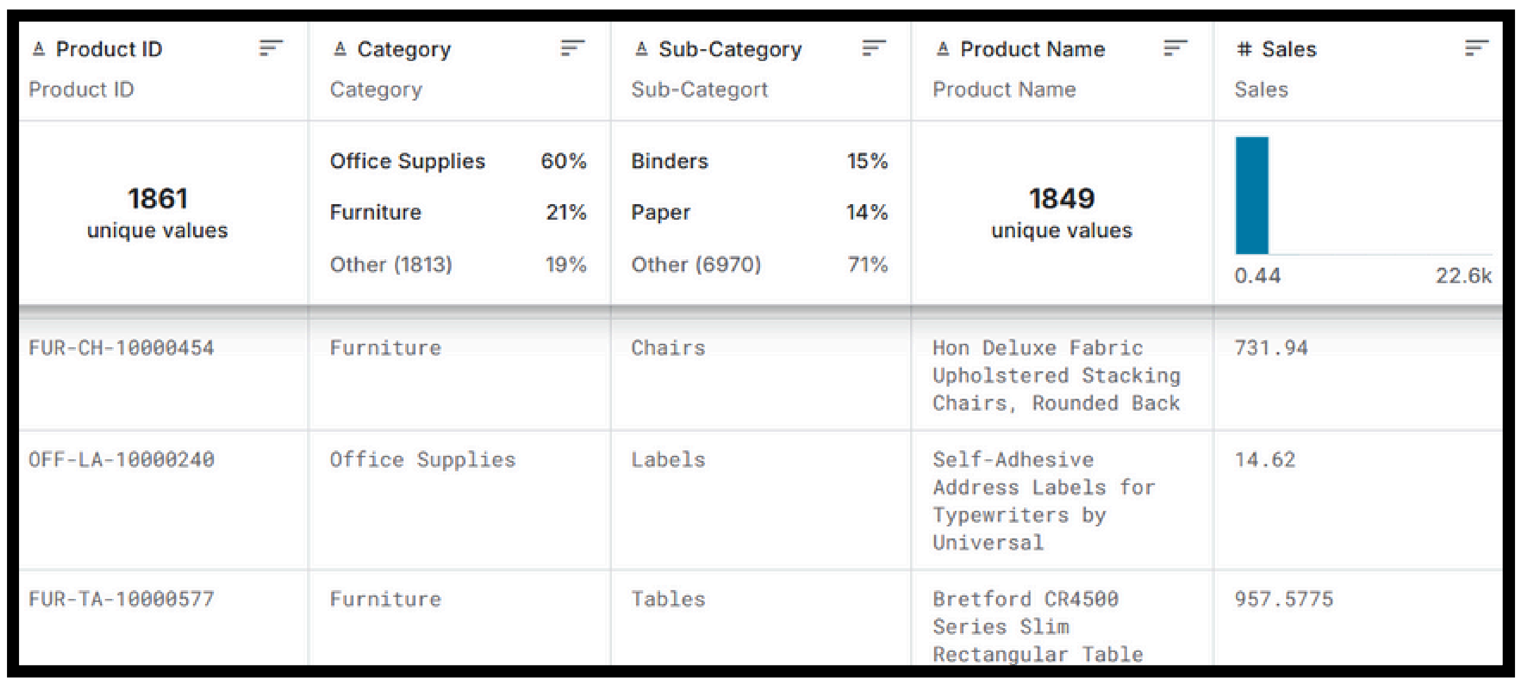
Link : https://www.kaggle.com/datasets/rohitsahoo/sales-forecasting

Step 2 : Initial Data Review



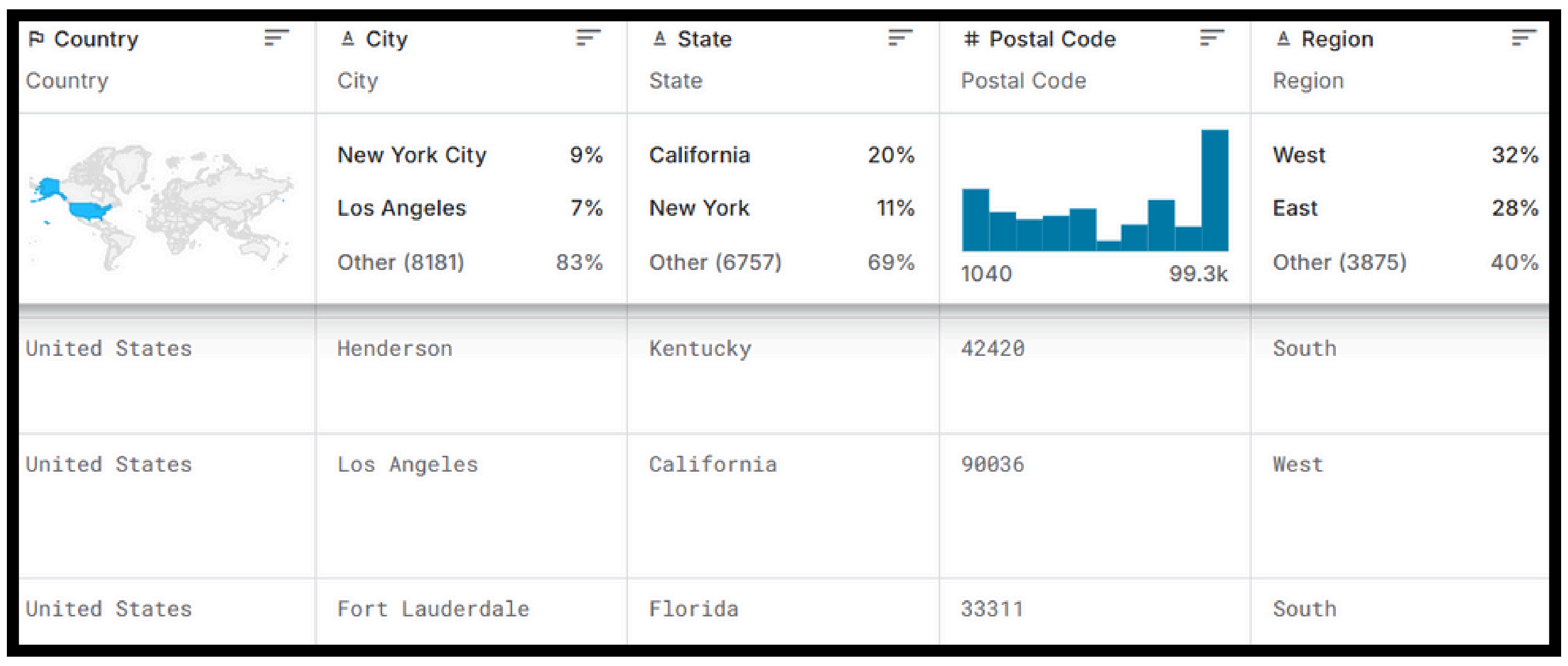
Row ID Row ID	Order ID Order ID	Order Date Order Date	Ship Date Ship Date	Ship Mode Ship mode
1 – 9800	4922 unique values	1230 unique values	1326 unique values	Standard Class 60% Second Class 19% Other (2039) 21%
1	CA-2017-152156	08/11/2017	11/11/2017	Second Class
2	CA-2017-152156	08/11/2017	11/11/2017	Second Class
3	CA-2017-138688	12/06/2017	16/06/2017	Second Class

Order’s date database overview



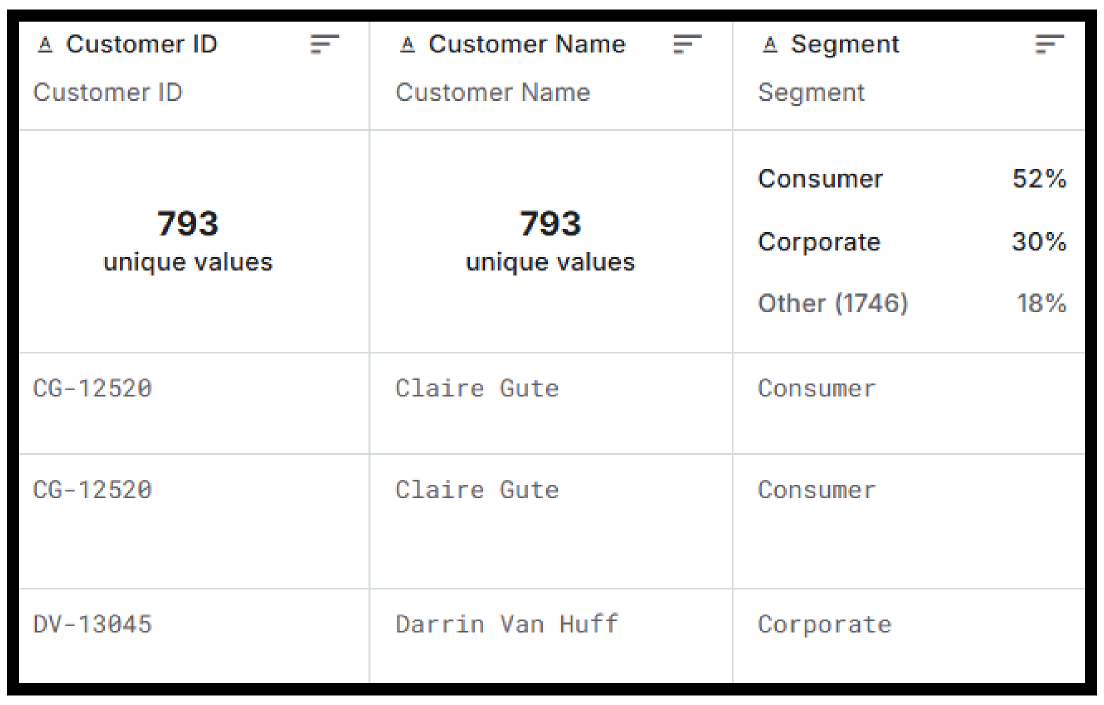
Product ID Product ID	Category Category	Sub-Category Sub-Categort	Product Name Product Name	Sales Sales
1861 unique values	Office Supplies 60% Furniture 21% Other (1813) 19%	Binders 15% Paper 14% Other (6970) 71%	1849 unique values	0.44 – 22.6k
FUR-CH-10000454	Furniture	Chairs	Hon Deluxe Fabric Upholstered Stacking Chairs, Rounded Back	731.94
OFF-LA-10000240	Office Supplies	Labels	Self-Adhesive Address Labels for Typewriters by Universal	14.62
FUR-TA-10000577	Furniture	Tables	Bretford CR4500 Series Slim Rectangular Table	957.5775

Products database overview



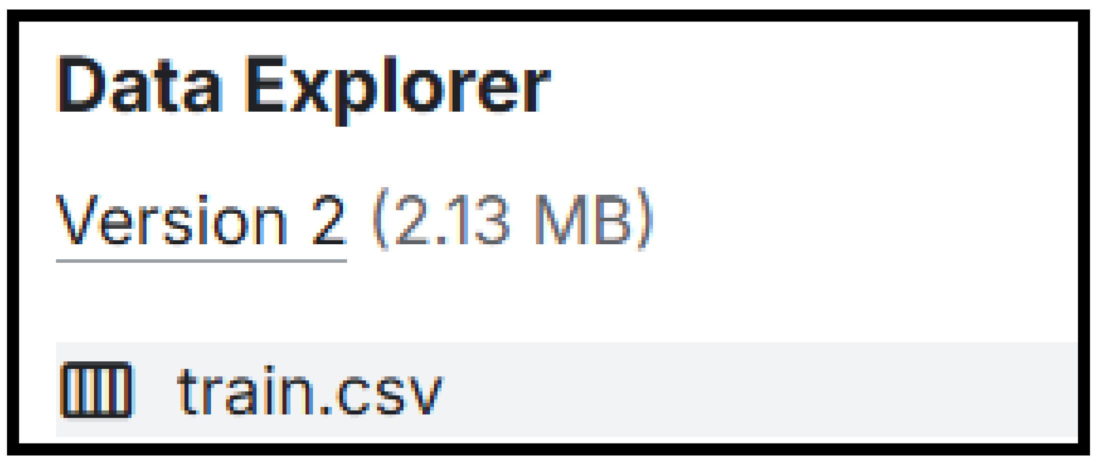
Country Country	City City	State State	Postal Code Postal Code	Region Region
	New York City 9% Los Angeles 7% Other (8181) 83%	California 20% New York 11% Other (6757) 69%	1040 – 99.3k	West 32% East 28% Other (3875) 40%
United States	Henderson	Kentucky	42420	South
United States	Los Angeles	California	90036	West
United States	Fort Lauderdale	Florida	33311	South

geographic database overview



Customer ID Customer ID	Customer Name Customer Name	Segment Segment
793 unique values	793 unique values	Consumer 52% Corporate 30% Other (1746) 18%
CG-12520	Claire Gute	Consumer
CG-12520	Claire Gute	Consumer
DV-13045	Darrin Van Huff	Corporate

Customers database overview



Data Explorer

Version 2 (2.13 MB)

train.csv

The collected dataset

kaggle

Phase 2 : Data preparation

```python
import pandas as pd
import numpy as np
from datetime import datetime
import os
import sqlite3
```

Importing necessary Python libraries (pandas, numpy, datetime, os, sqlite3)

Step 1 :Data Extraction

```python
df = pd.read_csv('train.csv')
print(f"✅ Data loaded: {df.shape[0]} rows, {df.shape[1]} columns")
```

Loading retail sales dataset from 'train.csv' file into pandas DataFrame and Confirming successful data load with row and column count verification

Step 2 :Data Cleaning

```python
print("Converting dates...")
df['Order Date'] = pd.to_datetime(df['Order Date'], format='%d/%m/%Y')
df['Ship Date'] = pd.to_datetime(df['Ship Date'], format='%d/%m/%Y')
```

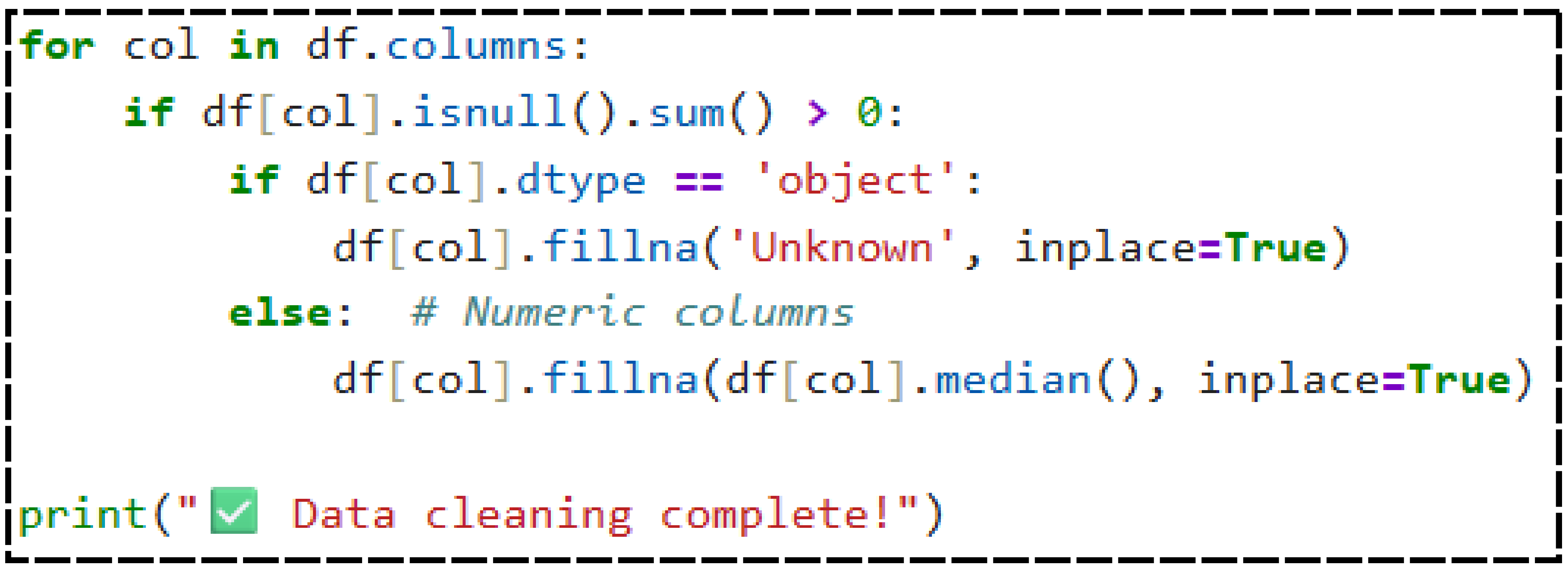
Fix date columns

```python
if duplicates > 0:
    df = df.drop_duplicates()
    print(f"Removed {duplicates} duplicate rows")
```

Remove duplicates if any

```python
text_columns = ['Customer Name', 'Product Name', 'City', 'State']
for col in text_columns:
    if col in df.columns:
        df[col] = df[col].str.strip()
```

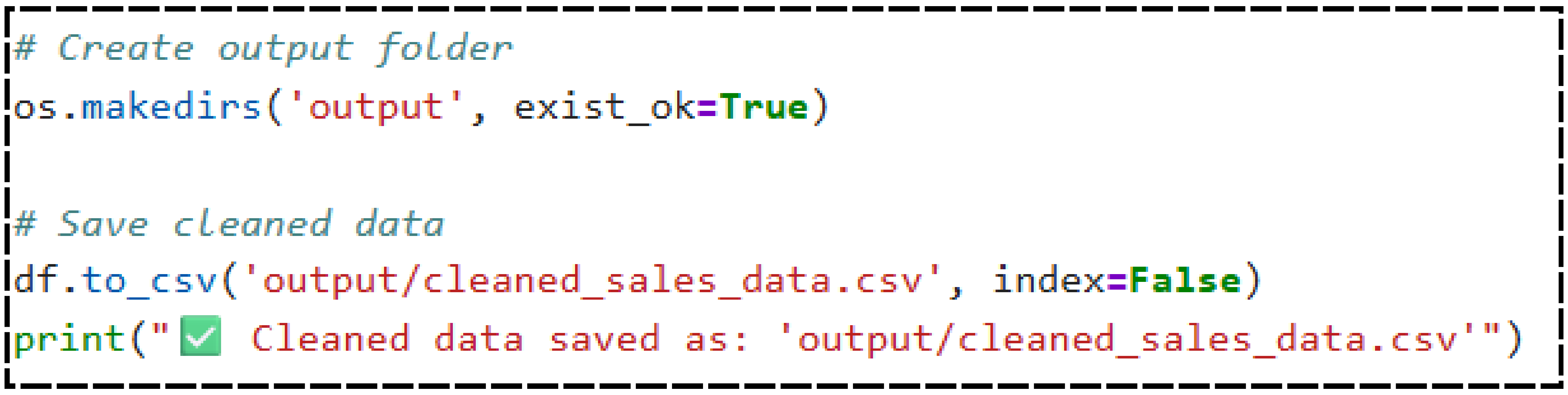
Fix text columns (remove extra spaces)



```python
for col in df.columns:
    if df[col].isnull().sum() > 0:
        if df[col].dtype == 'object':
            df[col].fillna('Unknown', inplace=True)
        else:  # Numeric columns
            df[col].fillna(df[col].median(), inplace=True)

print("✅ Data cleaning complete!")
```

Handle missing values (simple approach)

Step 3 :Data Saving



```python
# Create output folder
os.makedirs('output', exist_ok=True)

# Save cleaned data
df.to_csv('output/cleaned_sales_data.csv', index=False)
print("✅ Cleaned data saved as: 'output/cleaned_sales_data.csv'")
```

Saving cleaned data to CSV files

Phase 3 : Data modeling / loading

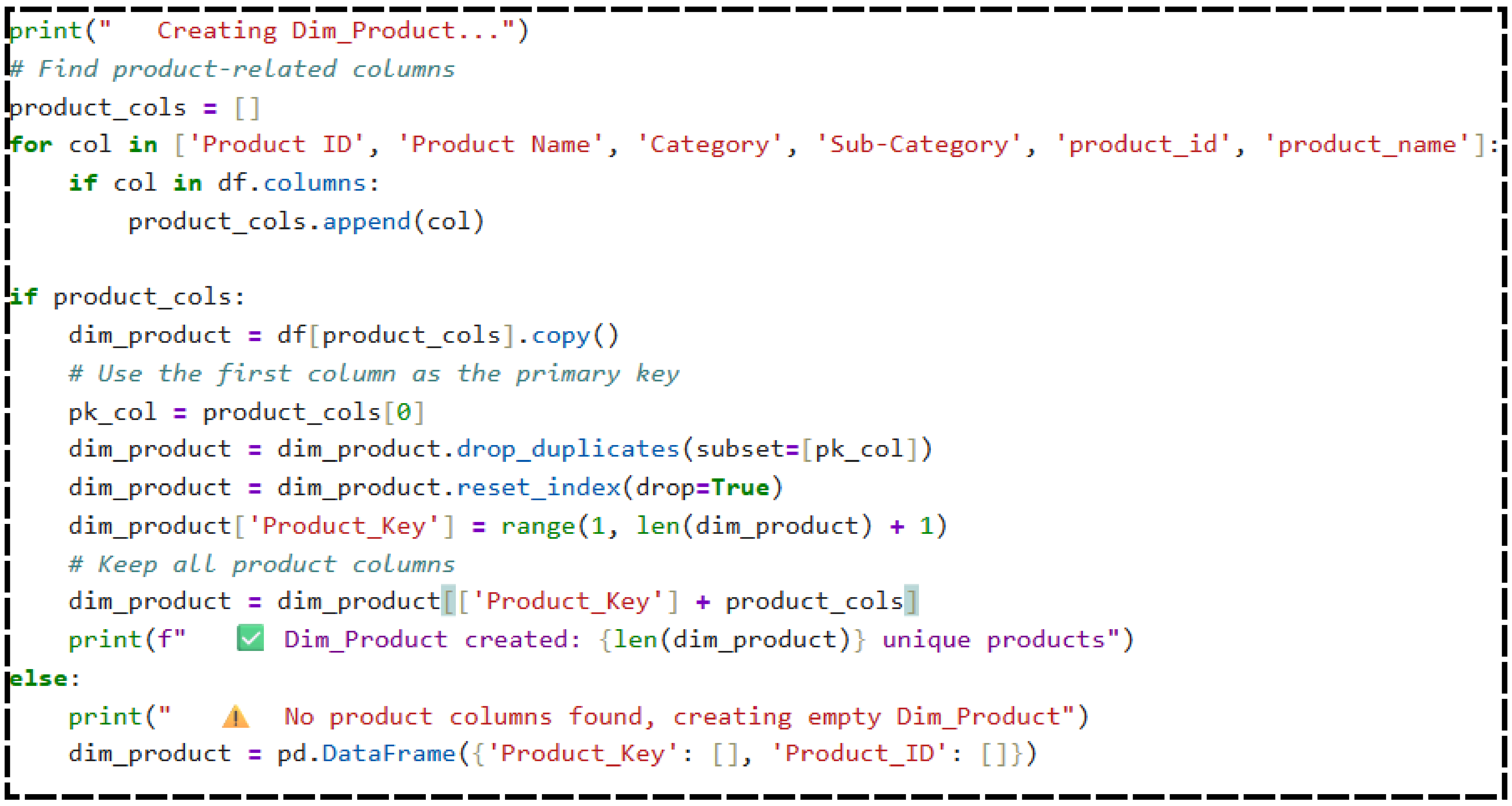
Fact Table

- **FactSales**:
  - Sale-id (PK)
  - Product-key (FK)
  - Customer-key (FK)
  - date-key (FK)
  - GeographyID (FK)
  - SalesAmount
  - Quantity

Dimension Tables

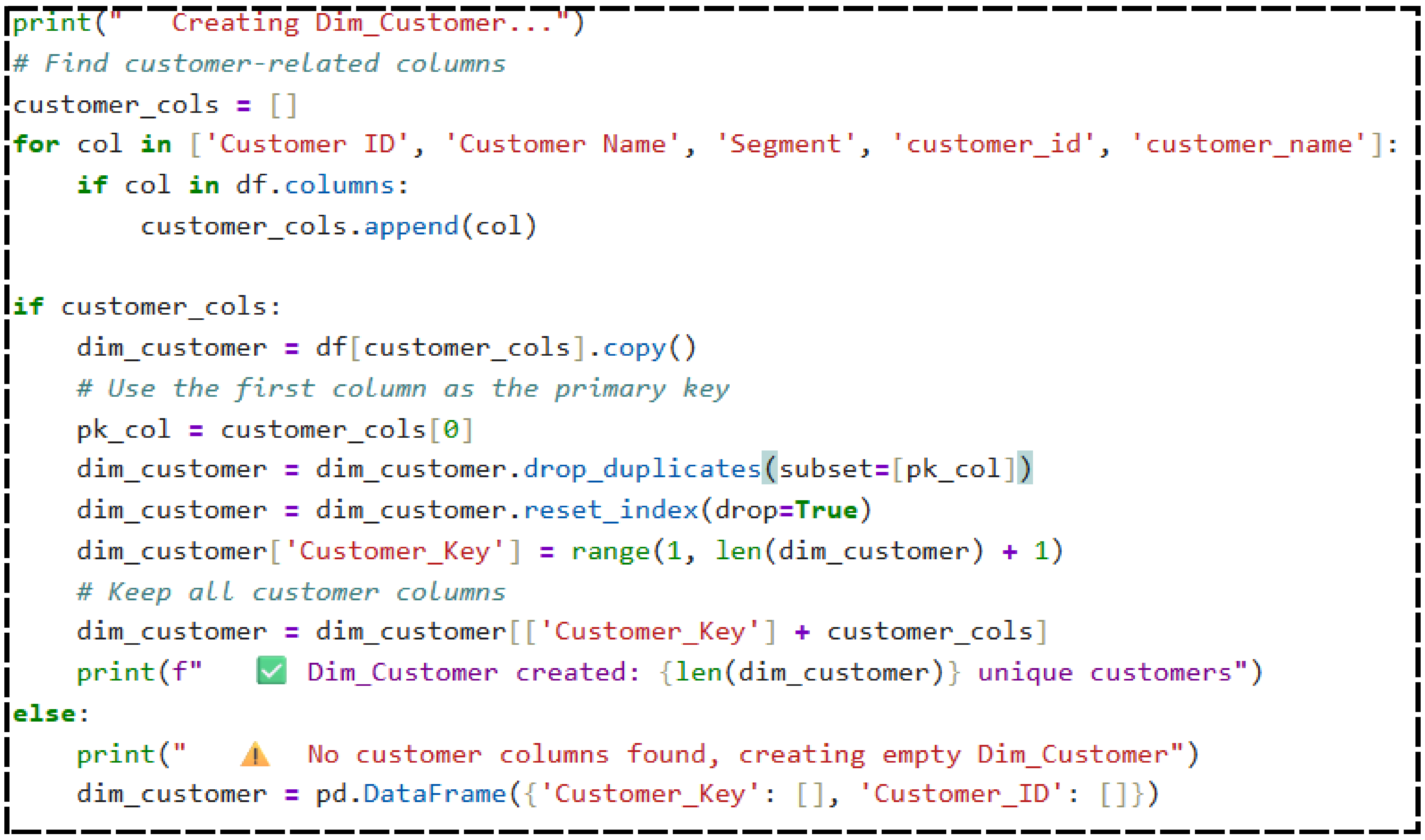
1. **DimProduct**
   - Product-id (PK)
   - ProductName
   - Category
   - SubCategory
2. **DimCustomer**
   - CustomerID (PK)
   - CustomerName
   - Segment
3. **DimGeography**
   - GeographyID (PK)
   - City
   - State
   - Region
   - Country
4. **Dimdate**
   - date-key (PK)
   - Date
   - Month
   - Quarter
   - Year
   - DayOfWeek
   - IsHoliday
   - IsWeekend

Step 1 :Creation of dimension/fact table



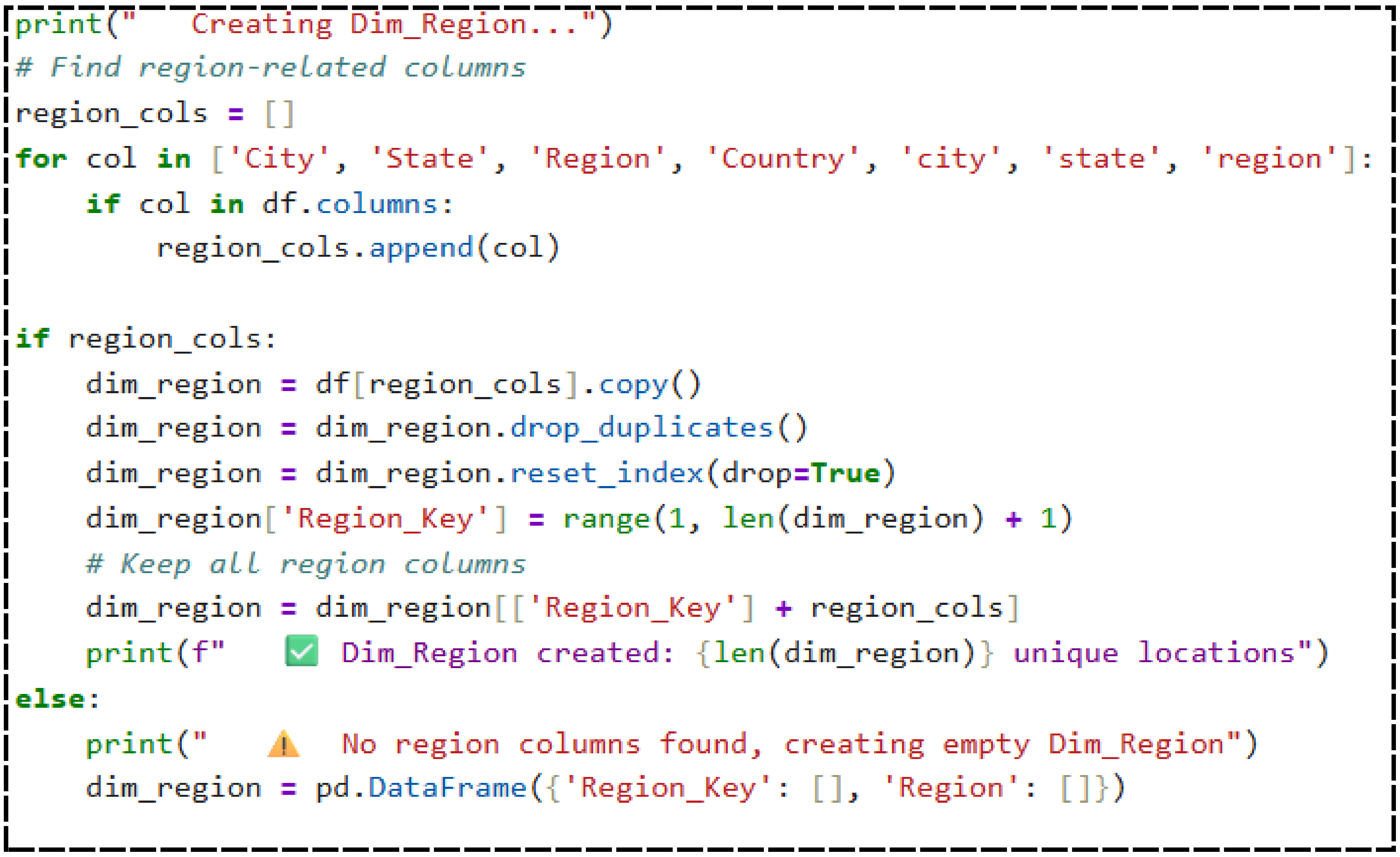
```python
print("   Creating Dim_Product...")
# Find product-related columns
product_cols = []
for col in ['Product ID', 'Product Name', 'Category', 'Sub-Category', 'product_id', 'product_name']:
    if col in df.columns:
        product_cols.append(col)

if product_cols:
    dim_product = df[product_cols].copy()
    # Use the first column as the primary key
    pk_col = product_cols[0]
    dim_product = dim_product.drop_duplicates(subset=[pk_col])
    dim_product = dim_product.reset_index(drop=True)
    dim_product['Product_Key'] = range(1, len(dim_product) + 1)
    # Keep all product columns
    dim_product = dim_product[['Product_Key'] + product_cols]
    print(f"   ✅ Dim_Product created: {len(dim_product)} unique products")
else:
    print("   ⚠️ No product columns found, creating empty Dim_Product")
    dim_product = pd.DataFrame({'Product_Key': [], 'Product_ID': []})
```

Creation of the product table



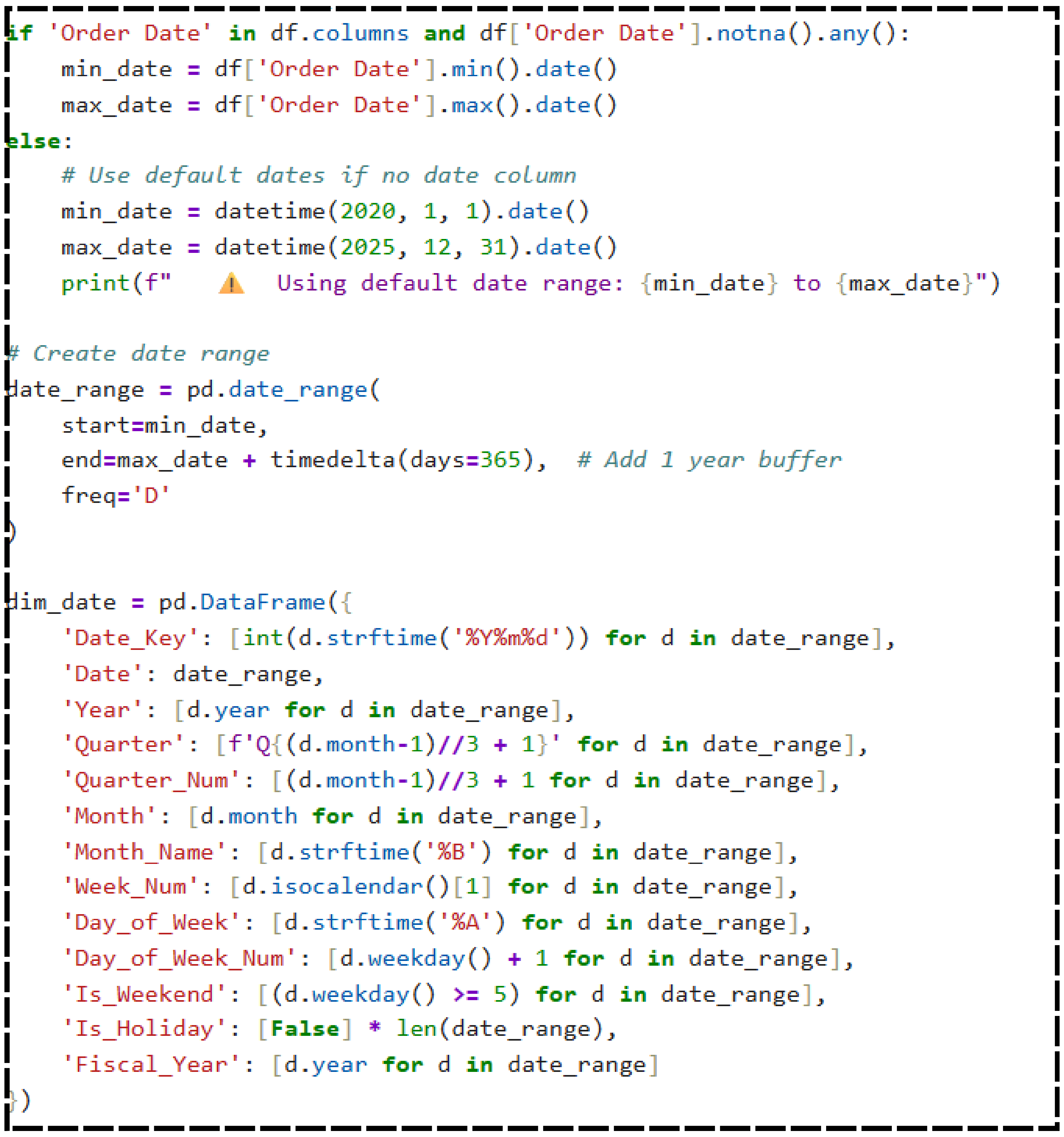
```python
print("   Creating Dim_Customer...")
# Find customer-related columns
customer_cols = []
for col in ['Customer ID', 'Customer Name', 'Segment', 'customer_id', 'customer_name']:
    if col in df.columns:
        customer_cols.append(col)

if customer_cols:
    dim_customer = df[customer_cols].copy()
    # Use the first column as the primary key
    pk_col = customer_cols[0]
    dim_customer = dim_customer.drop_duplicates(subset=[pk_col])
    dim_customer = dim_customer.reset_index(drop=True)
    dim_customer['Customer_Key'] = range(1, len(dim_customer) + 1)
    # Keep all customer columns
    dim_customer = dim_customer[['Customer_Key'] + customer_cols]
    print(f"   ✅ Dim_Customer created: {len(dim_customer)} unique customers")
else:
    print("   ⚠️ No customer columns found, creating empty Dim_Customer")
    dim_customer = pd.DataFrame({'Customer_Key': [], 'Customer_ID': []})
```

Creation of the customer table



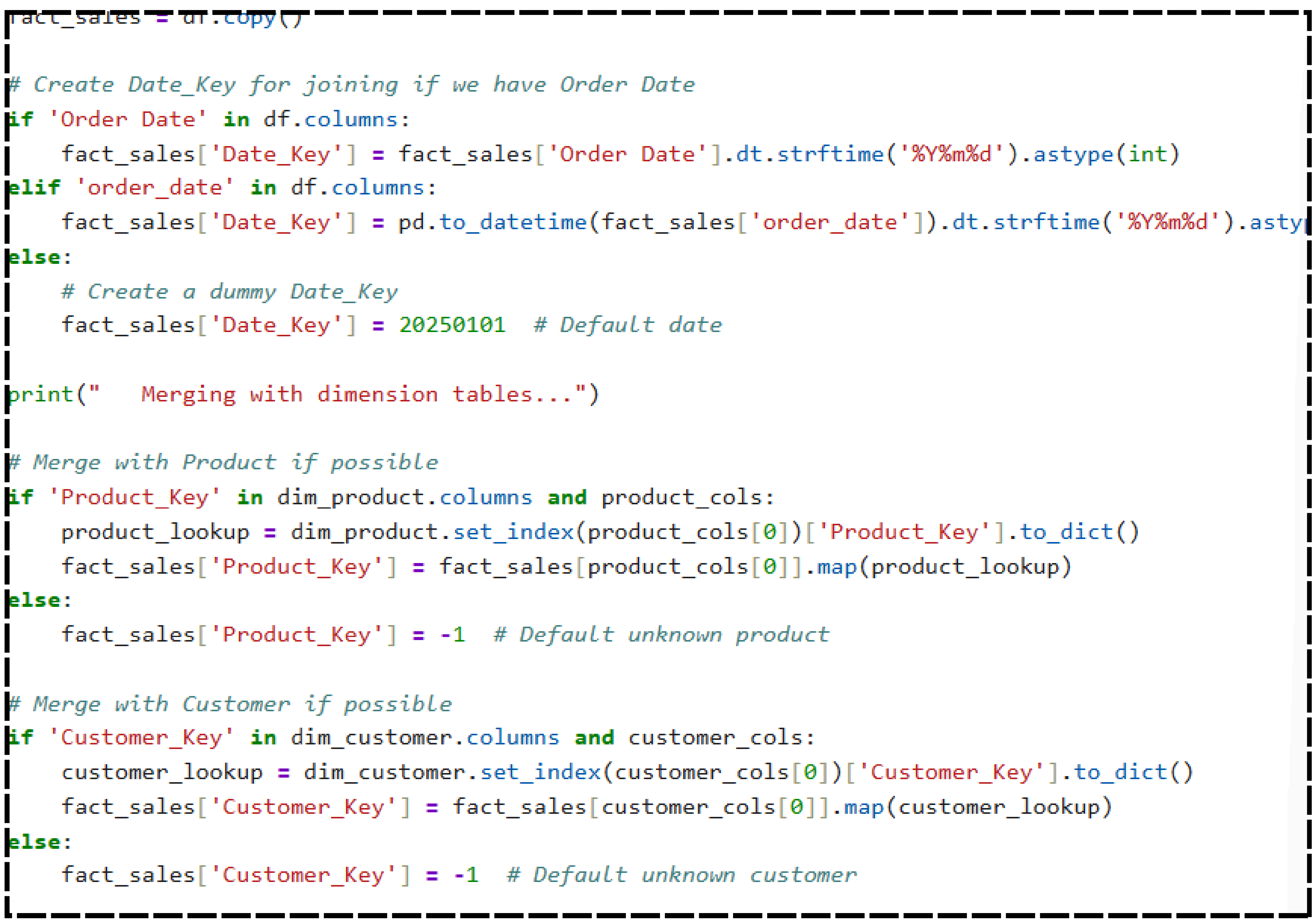
```python
print("   Creating Dim_Region...")
# Find region-related columns
region_cols = []
for col in ['City', 'State', 'Region', 'Country', 'city', 'state', 'region']:
    if col in df.columns:
        region_cols.append(col)

if region_cols:
    dim_region = df[region_cols].copy()
    dim_region = dim_region.drop_duplicates()
    dim_region = dim_region.reset_index(drop=True)
    dim_region['Region_Key'] = range(1, len(dim_region) + 1)
    # Keep all region columns
    dim_region = dim_region[['Region_Key'] + region_cols]
    print(f"   ✅ Dim_Region created: {len(dim_region)} unique locations")
else:
    print("   ⚠️ No region columns found, creating empty Dim_Region")
    dim_region = pd.DataFrame({'Region_Key': [], 'Region': []})
```

Creation of the geographic table



```python
if 'Order Date' in df.columns and df['Order Date'].notna().any():
    min_date = df['Order Date'].min().date()
    max_date = df['Order Date'].max().date()
else:
    # Use default dates if no date column
    min_date = datetime(2020, 1, 1).date()
    max_date = datetime(2025, 12, 31).date()
    print(f"   ⚠️ Using default date range: {min_date} to {max_date}")

# Create date range
date_range = pd.date_range(
    start=min_date,
    end=max_date + timedelta(days=365),  # Add 1 year buffer
    freq='D'
)

dim_date = pd.DataFrame({
    'Date_Key': [int(d.strftime('%Y%m%d')) for d in date_range],
    'Date': date_range,
    'Year': [d.year for d in date_range],
    'Quarter': [f'Q{(d.month-1)//3 + 1}' for d in date_range],
    'Quarter_Num': [(d.month-1)//3 + 1 for d in date_range],
    'Month': [d.month for d in date_range],
    'Month_Name': [d.strftime('%B') for d in date_range],
    'Week_Num': [d.isocalendar()[1] for d in date_range],
    'Day_of_Week': [d.strftime('%A') for d in date_range],
    'Day_of_Week_Num': [d.weekday() + 1 for d in date_range],
    'Is_Weekend': [(d.weekday() >= 5) for d in date_range],
    'Is_Holiday': [False] * len(date_range),
    'Fiscal_Year': [d.year for d in date_range]
})
```

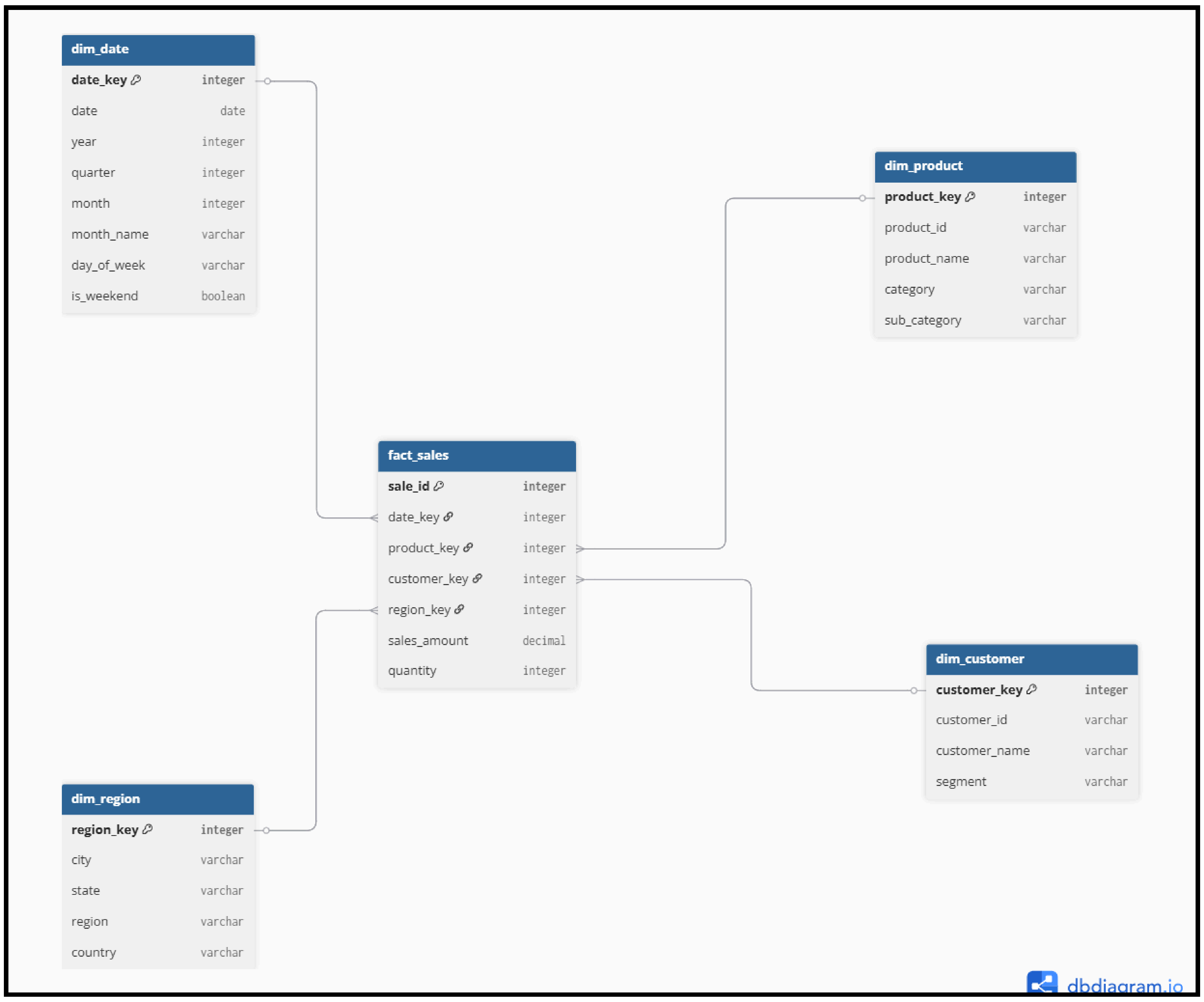
Creation of the date table

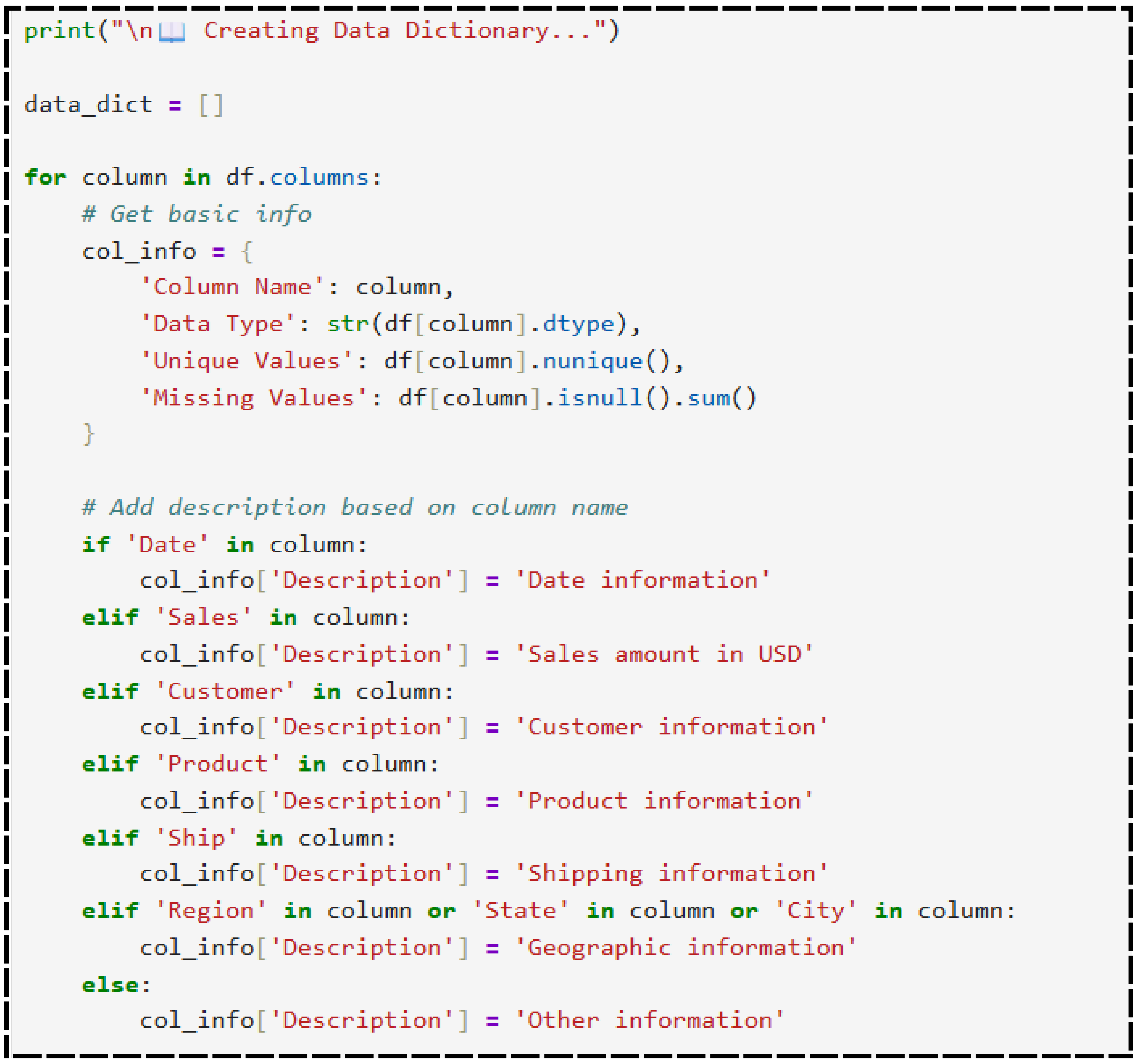


```
fact_sales = df.copy()

# Create Date_Key for joining if we have Order Date
if 'Order Date' in df.columns:
    fact_sales['Date_Key'] = fact_sales['Order Date'].dt.strftime('%Y%m%d').astype(int)
elif 'order_date' in df.columns:
    fact_sales['Date_Key'] = pd.to_datetime(fact_sales['order_date']).dt.strftime('%Y%m%d').asty
else:
    # Create a dummy Date_Key
    fact_sales['Date_Key'] = 20250101  # Default date

print("   Merging with dimension tables...")

# Merge with Product if possible
if 'Product_Key' in dim_product.columns and product_cols:
    product_lookup = dim_product.set_index(product_cols[0])['Product_Key'].to_dict()
    fact_sales['Product_Key'] = fact_sales[product_cols[0]].map(product_lookup)
else:
    fact_sales['Product_Key'] = -1  # Default unknown product

# Merge with Customer if possible
if 'Customer_Key' in dim_customer.columns and customer_cols:
    customer_lookup = dim_customer.set_index(customer_cols[0])['Customer_Key'].to_dict()
    fact_sales['Customer_Key'] = fact_sales[customer_cols[0]].map(customer_lookup)
else:
    fact_sales['Customer_Key'] = -1  # Default unknown customer
```

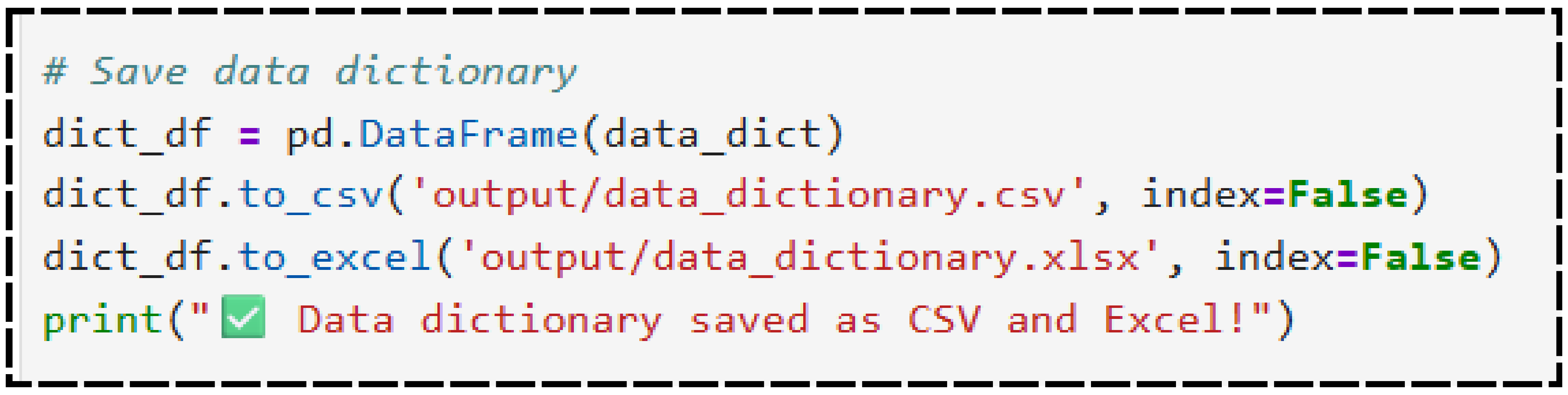
Creation of the fact table

Step 2 :Star schema





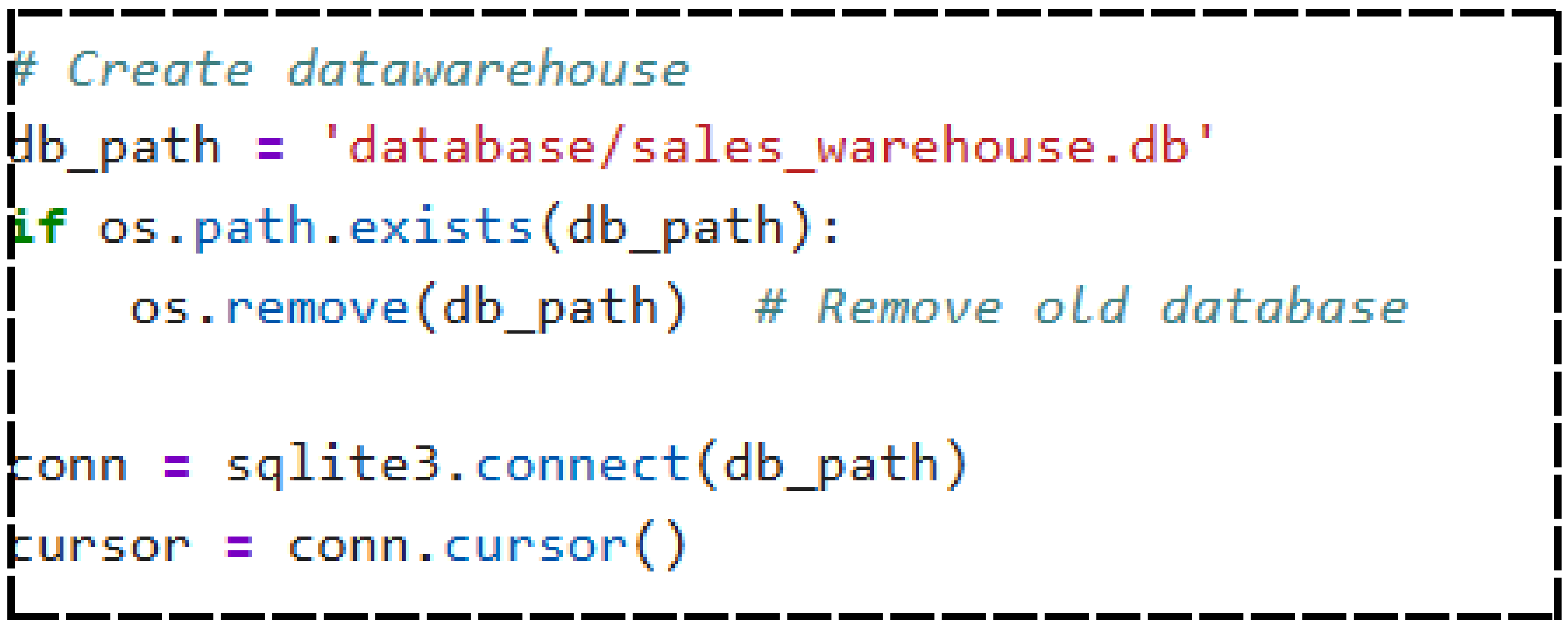
```python
print("\n📖 Creating Data Dictionary...")

data_dict = []

for column in df.columns:
    # Get basic info
    col_info = {
        'Column Name': column,
        'Data Type': str(df[column].dtype),
        'Unique Values': df[column].nunique(),
        'Missing Values': df[column].isnull().sum()
    }

    # Add description based on column name
    if 'Date' in column:
        col_info['Description'] = 'Date information'
    elif 'Sales' in column:
        col_info['Description'] = 'Sales amount in USD'
    elif 'Customer' in column:
        col_info['Description'] = 'Customer information'
    elif 'Product' in column:
        col_info['Description'] = 'Product information'
    elif 'Ship' in column:
        col_info['Description'] = 'Shipping information'
    elif 'Region' in column or 'State' in column or 'City' in column:
        col_info['Description'] = 'Geographic information'
    else:
        col_info['Description'] = 'Other information'
```



```python
# Save data dictionary
dict_df = pd.DataFrame(data_dict)
dict_df.to_csv('output/data_dictionary.csv', index=False)
dict_df.to_excel('output/data_dictionary.xlsx', index=False)
print("✅ Data dictionary saved as CSV and Excel!")
```

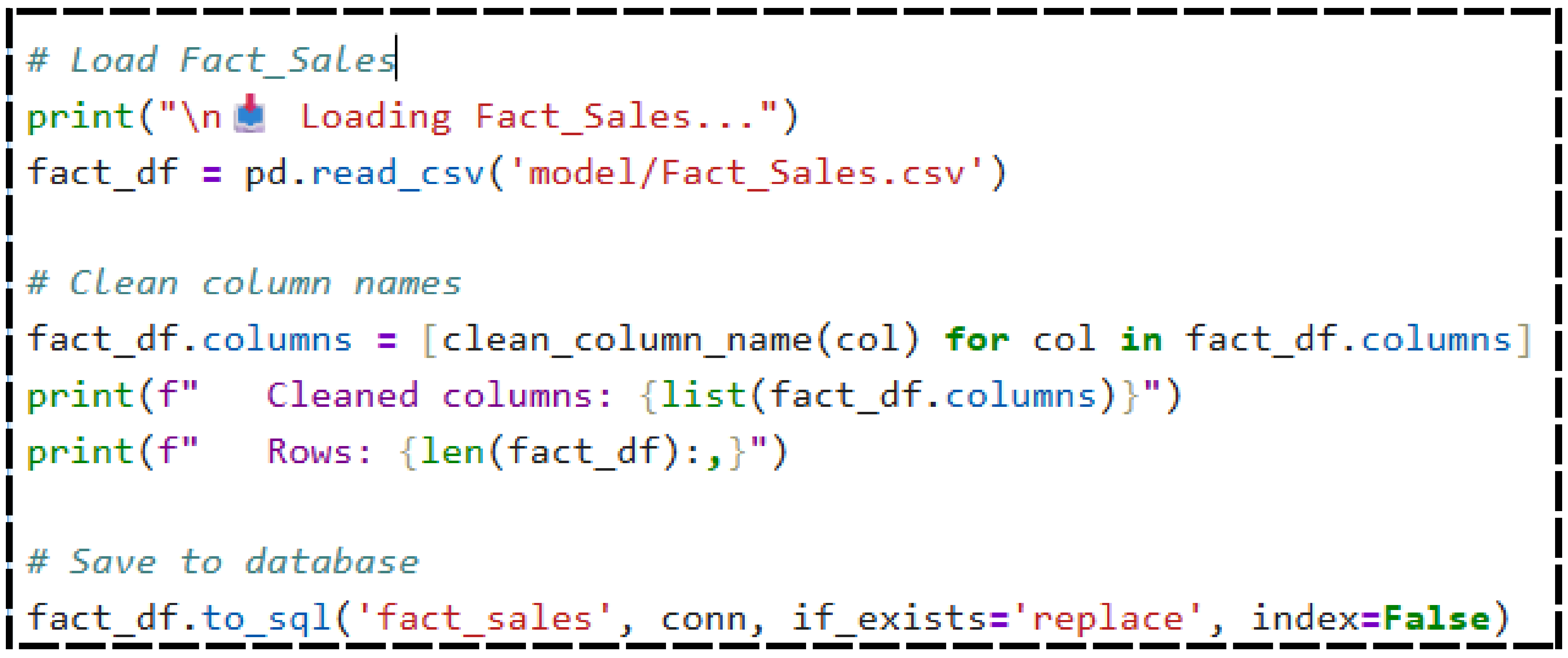
Creation of the data dictionary

Step 3 :Creation of the data warehouse

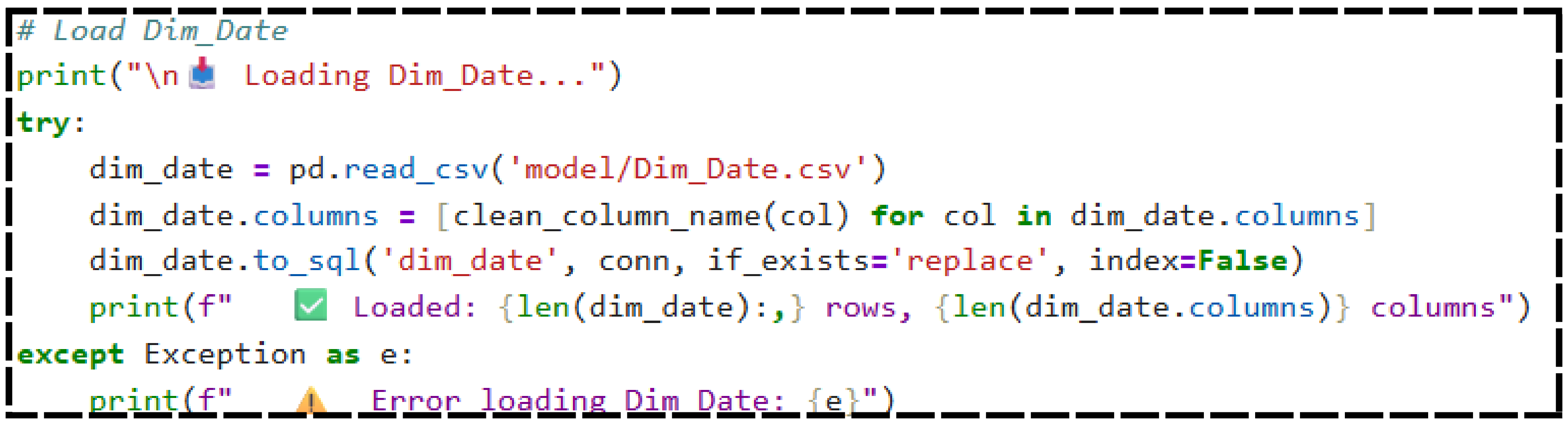


```python
# Create datawarehouse
db_path = 'database/sales_warehouse.db'
if os.path.exists(db_path):
    os.remove(db_path)  # Remove old database

conn = sqlite3.connect(db_path)
cursor = conn.cursor()
```

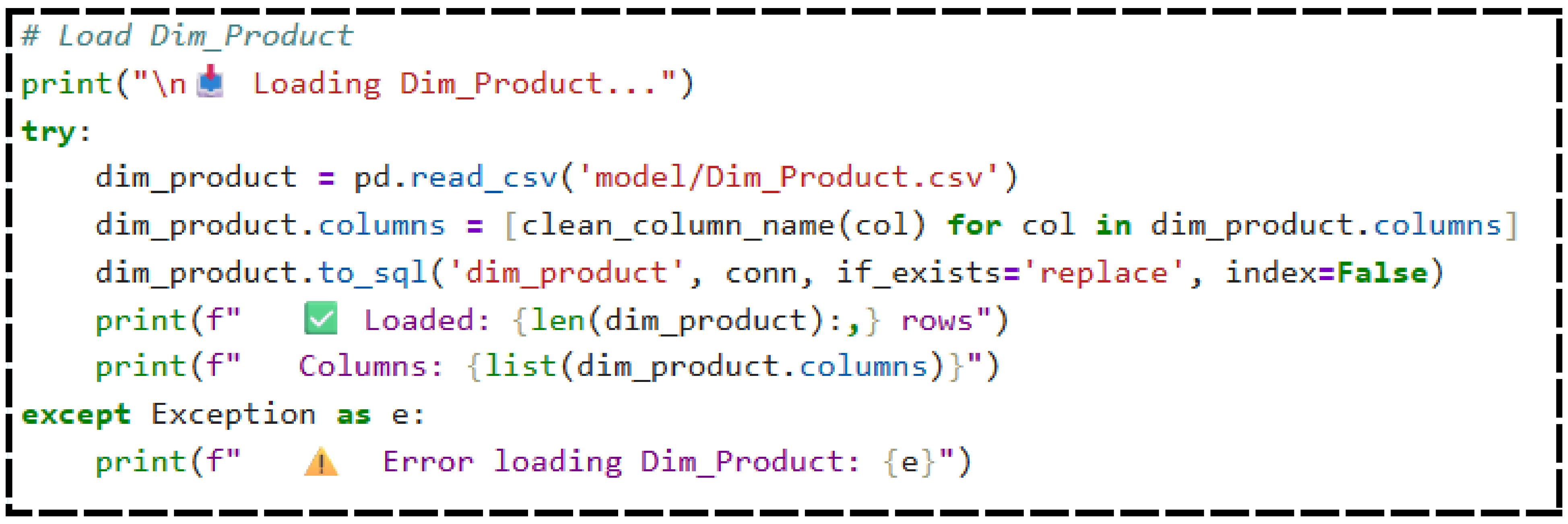
Step 4 :Loading tables into the data warehouse



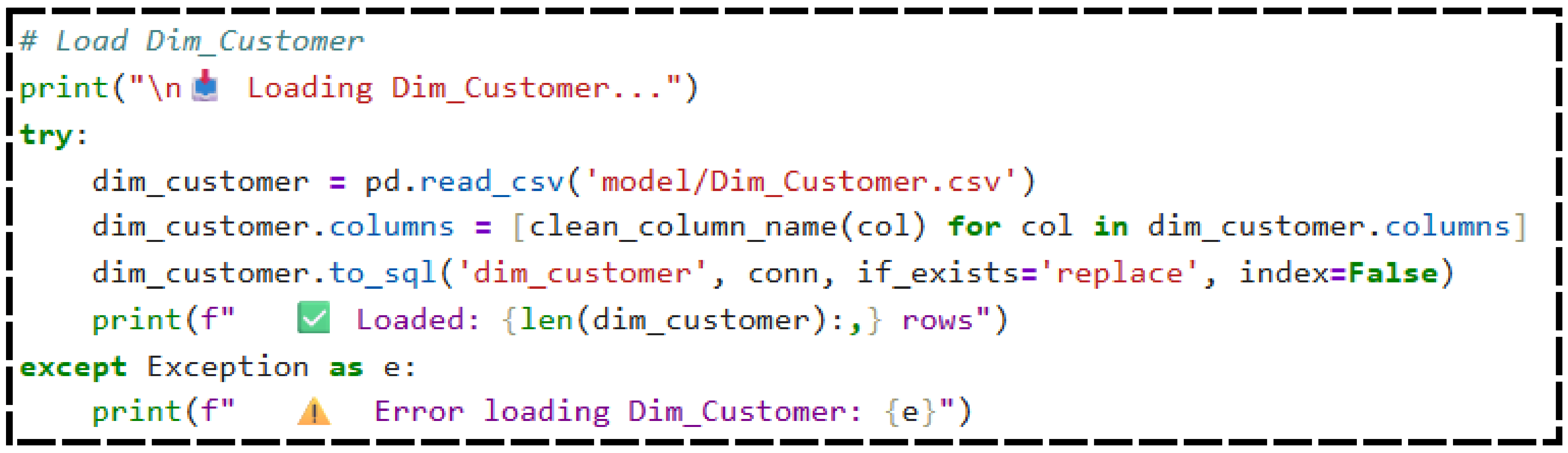
```python
# Load Fact_Sales
print("\n📥  Loading Fact_Sales...")
fact_df = pd.read_csv('model/Fact_Sales.csv')

# Clean column names
fact_df.columns = [clean_column_name(col) for col in fact_df.columns]
print(f"   Cleaned columns: {list(fact_df.columns)}")
print(f"   Rows: {len(fact_df):,}")

# Save to database
fact_df.to_sql('fact_sales', conn, if_exists='replace', index=False)
```



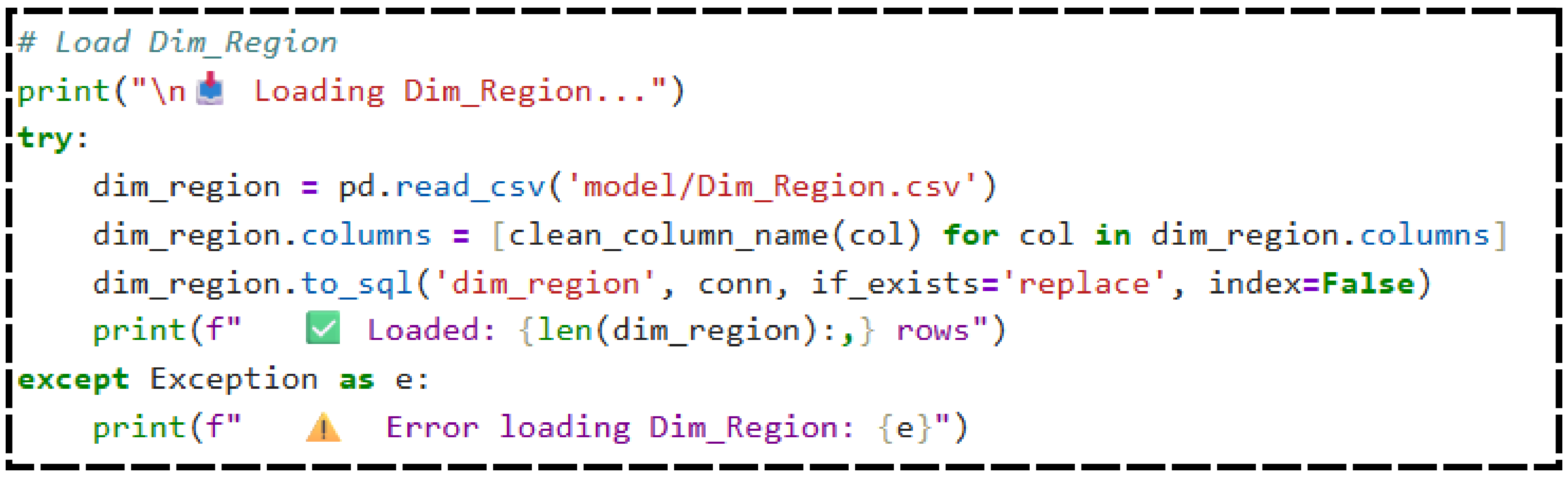
```python
# Load Dim_Date
print("\n📥  Loading Dim_Date...")
try:
    dim_date = pd.read_csv('model/Dim_Date.csv')
    dim_date.columns = [clean_column_name(col) for col in dim_date.columns]
    dim_date.to_sql('dim_date', conn, if_exists='replace', index=False)
    print(f"   ✅  Loaded: {len(dim_date):,} rows, {len(dim_date.columns)} columns")
except Exception as e:
    print(f"   ⚠️   Error loading Dim_Date: {e}")
```



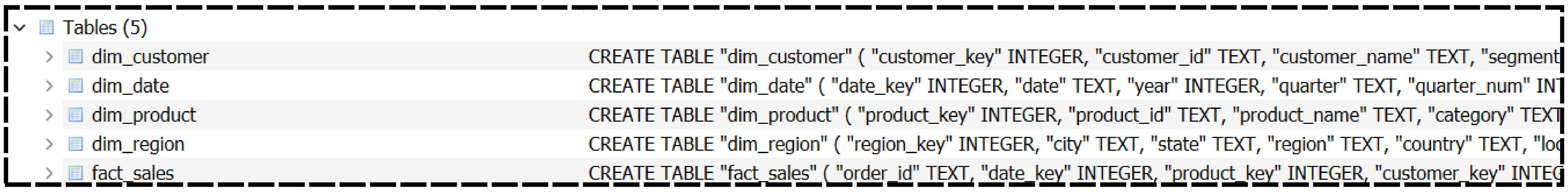
```python
# Load Dim_Product
print("\n📥  Loading Dim_Product...")
try:
    dim_product = pd.read_csv('model/Dim_Product.csv')
    dim_product.columns = [clean_column_name(col) for col in dim_product.columns]
    dim_product.to_sql('dim_product', conn, if_exists='replace', index=False)
    print(f"   ✅  Loaded: {len(dim_product):,} rows")
    print(f"   Columns: {list(dim_product.columns)}")
except Exception as e:
    print(f"   ⚠️   Error loading Dim_Product: {e}")
```



```python
# Load Dim_Customer
print("\n📥  Loading Dim_Customer...")
try:
    dim_customer = pd.read_csv('model/Dim_Customer.csv')
    dim_customer.columns = [clean_column_name(col) for col in dim_customer.columns]
    dim_customer.to_sql('dim_customer', conn, if_exists='replace', index=False)
    print(f"   ✅  Loaded: {len(dim_customer):,} rows")
except Exception as e:
    print(f"   ⚠️   Error loading Dim_Customer: {e}")
```



```python
# Load Dim_Region
print("\n📥  Loading Dim_Region...")
try:
    dim_region = pd.read_csv('model/Dim_Region.csv')
    dim_region.columns = [clean_column_name(col) for col in dim_region.columns]
    dim_region.to_sql('dim_region', conn, if_exists='replace', index=False)
    print(f"   ✅  Loaded: {len(dim_region):,} rows")
except Exception as e:
    print(f"   ⚠️   Error loading Dim_Region: {e}")
```

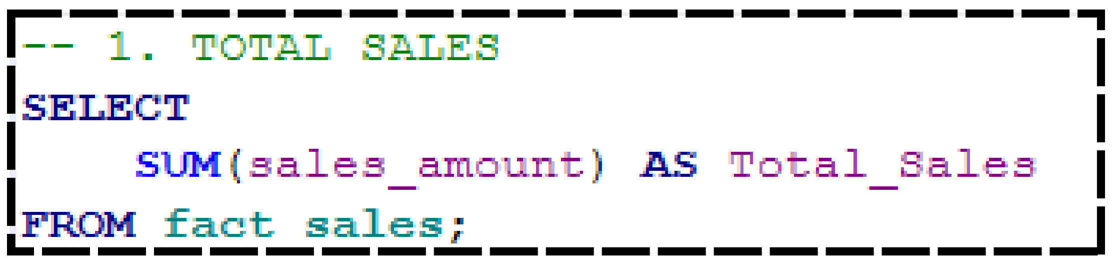


```
Tables (5)
  dim_customer    CREATE TABLE "dim_customer" ( "customer_key" INTEGER, "customer_id" TEXT, "customer_name" TEXT, "segment
  dim_date        CREATE TABLE "dim_date" ( "date_key" INTEGER, "date" TEXT, "year" INTEGER, "quarter" TEXT, "quarter_num" IN
  dim_product     CREATE TABLE "dim_product" ( "product_key" INTEGER, "product_id" TEXT, "product_name" TEXT, "category" TEXT
  dim_region      CREATE TABLE "dim_region" ( "region_key" INTEGER, "city" TEXT, "state" TEXT, "region" TEXT, "country" TEXT, "lo
  fact_sales      CREATE TABLE "fact_sales" ( "order_id" TEXT, "date_key" INTEGER, "product_key" INTEGER, "customer_key" INTEG
```

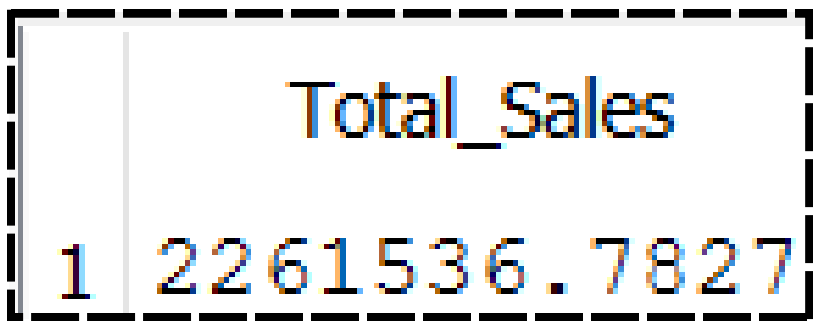
the tables created in sqlite database

Step 5 :ROLAP Queries

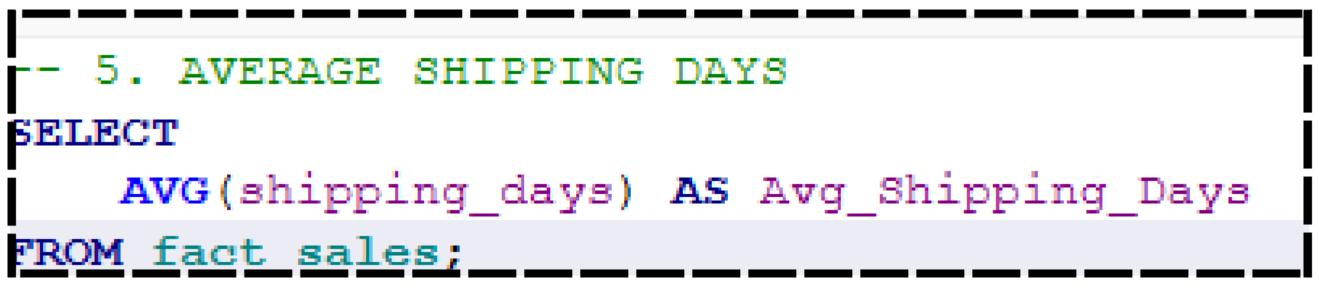
1. SLICING Operations



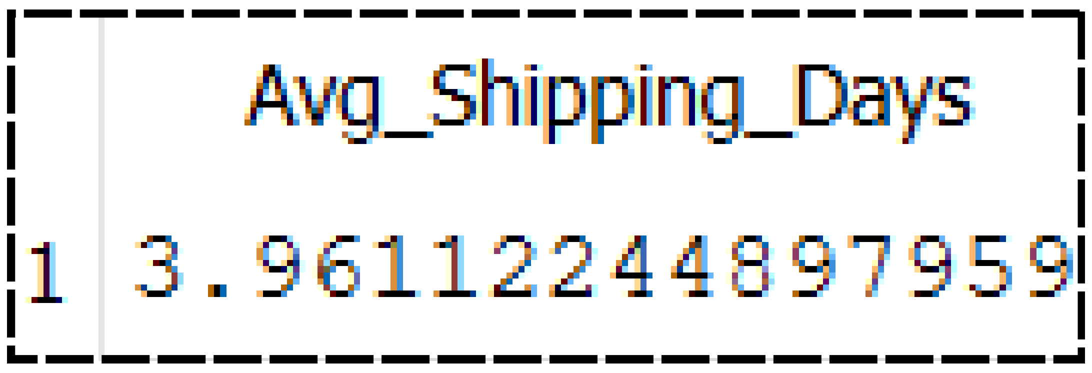
```sql
-- 1. TOTAL SALES
SELECT
    SUM(sales_amount) AS Total_Sales
FROM fact_sales;
```



	Total_Sales
1	2261536.7827

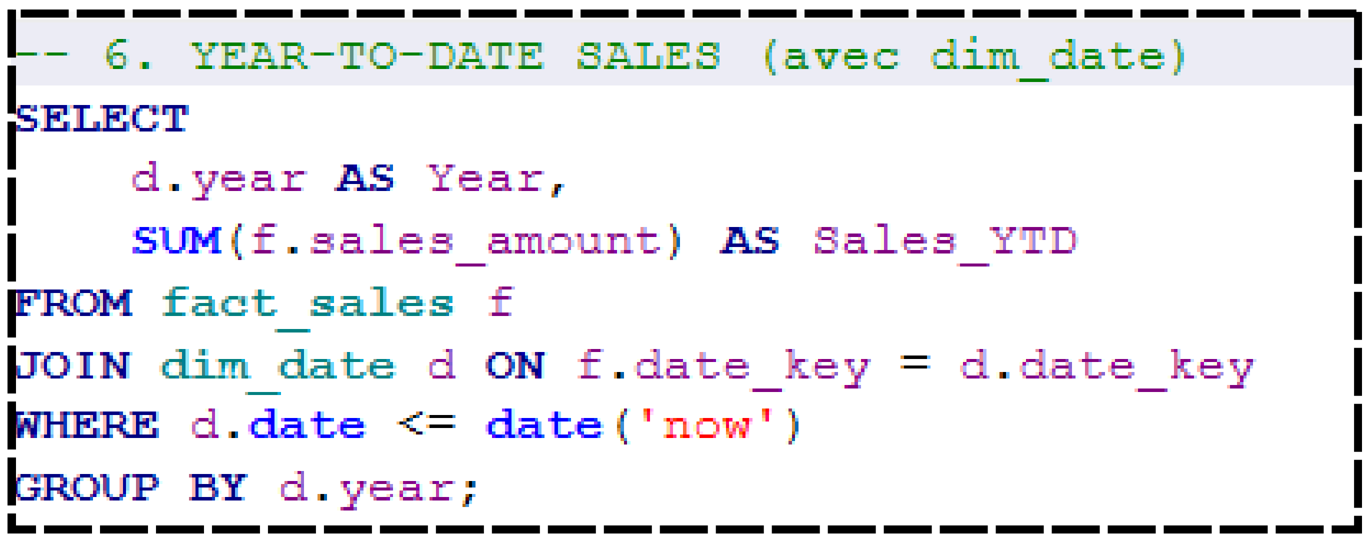


```sql
-- 5. AVERAGE SHIPPING DAYS
SELECT
    AVG(shipping_days) AS Avg_Shipping_Days
FROM fact_sales;
```

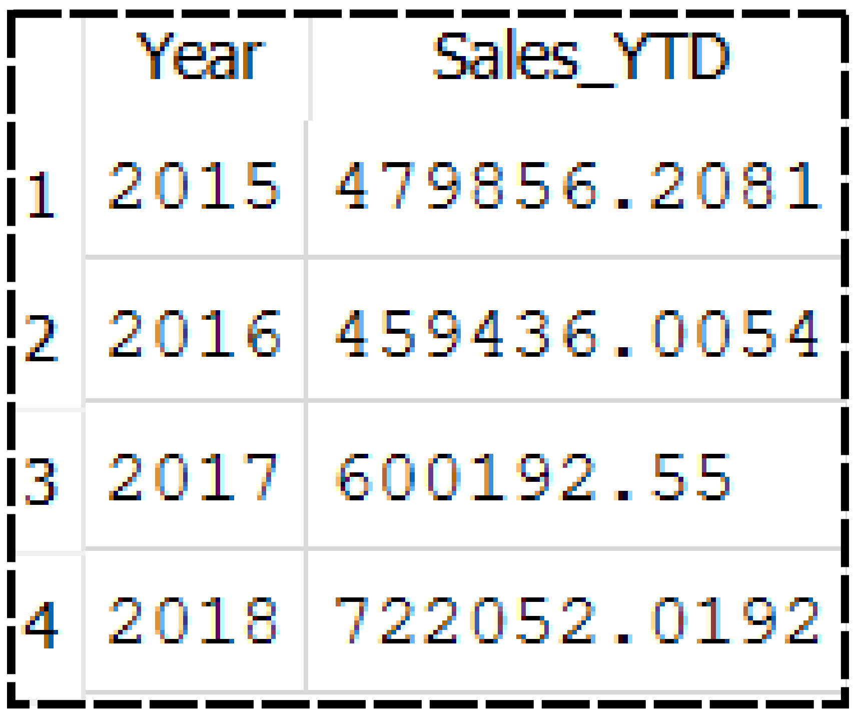


	Avg_Shipping_Days
1	3.96112244897959

2. DICING Operations

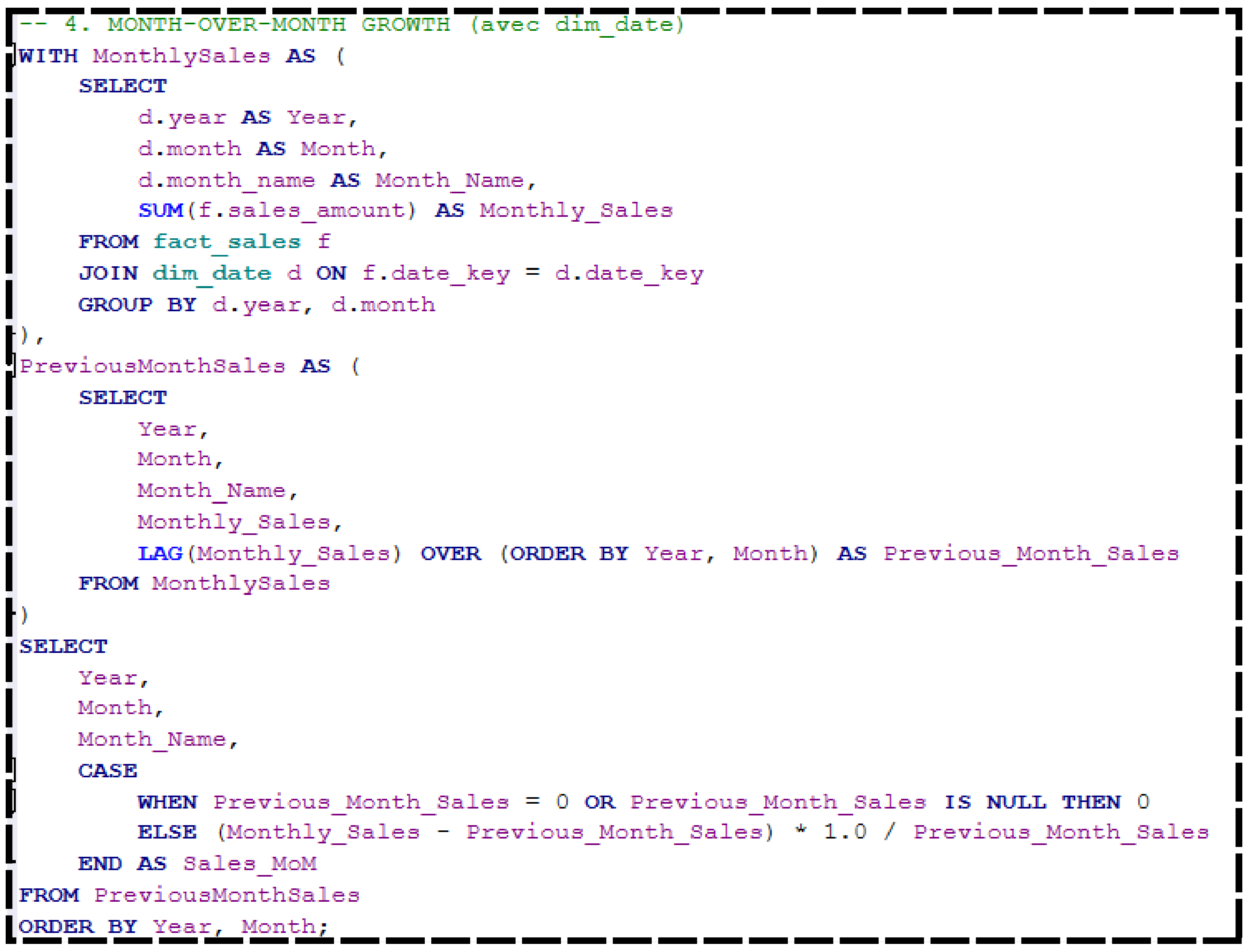


```sql
-- 6. YEAR-TO-DATE SALES (avec dim_date)
SELECT
    d.year AS Year,
    SUM(f.sales_amount) AS Sales_YTD
FROM fact_sales f
JOIN dim_date d ON f.date_key = d.date_key
WHERE d.date <= date('now')
GROUP BY d.year;
```

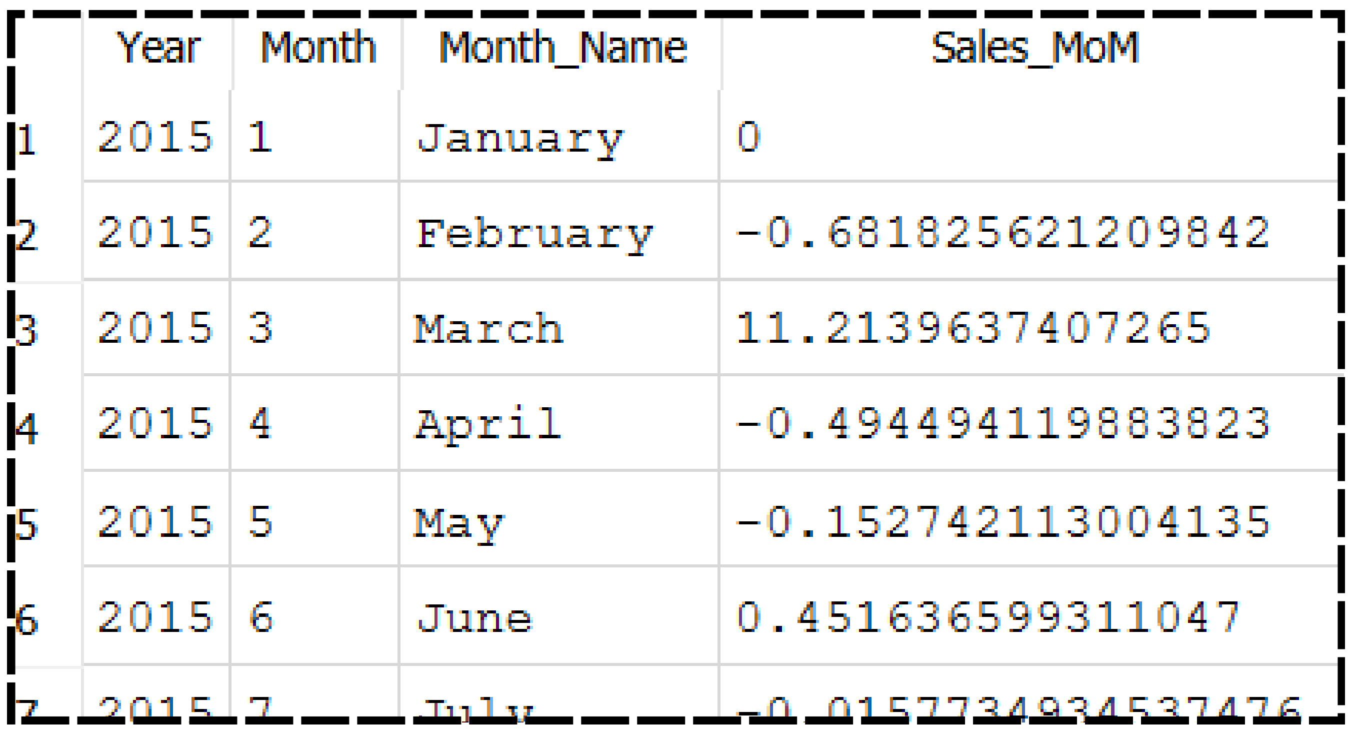


	Year	Sales_YTD
1	2015	479856.2081
2	2016	459436.0054
3	2017	600192.55
4	2018	722052.0192

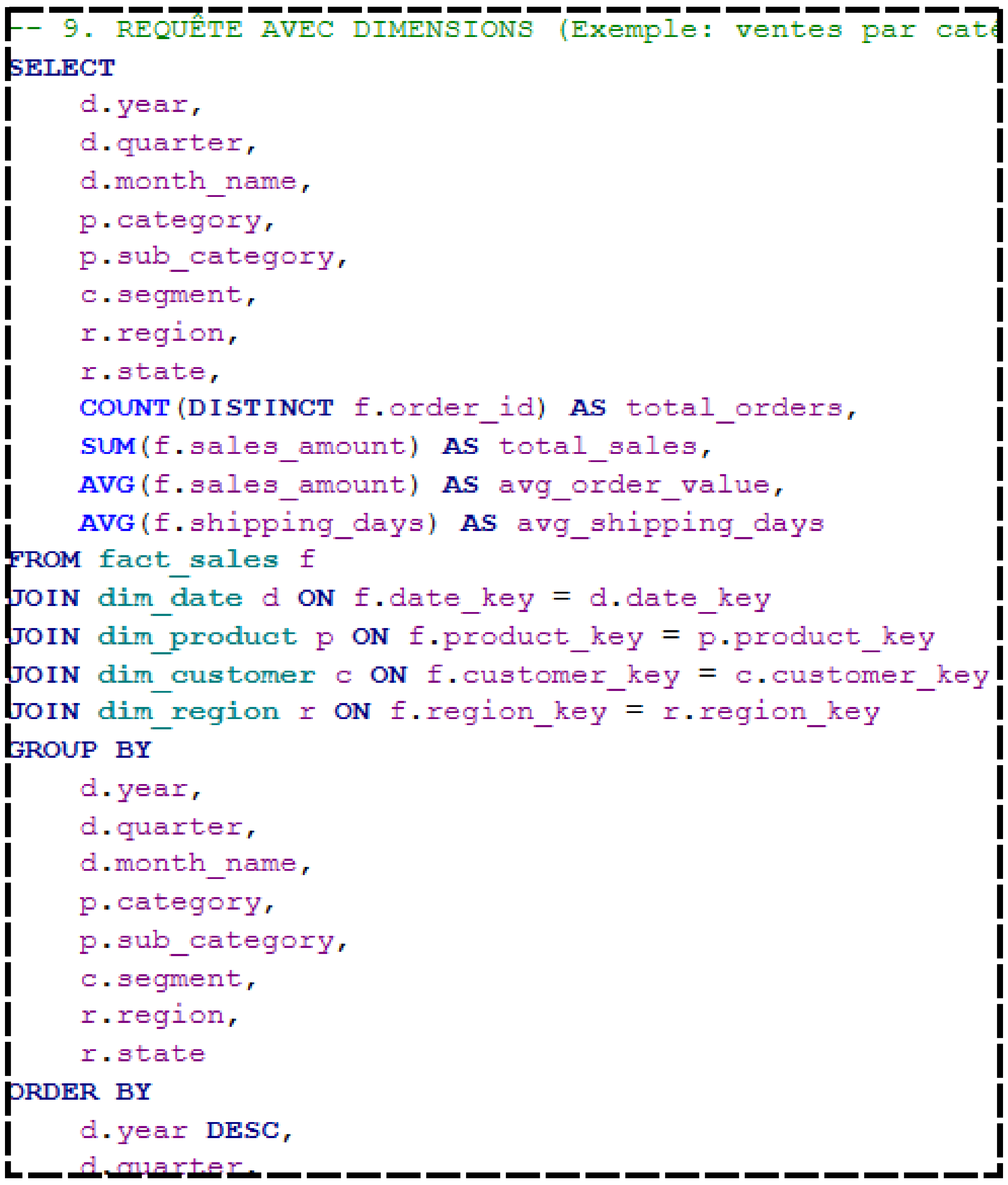
3. DRILL DOWN Operations



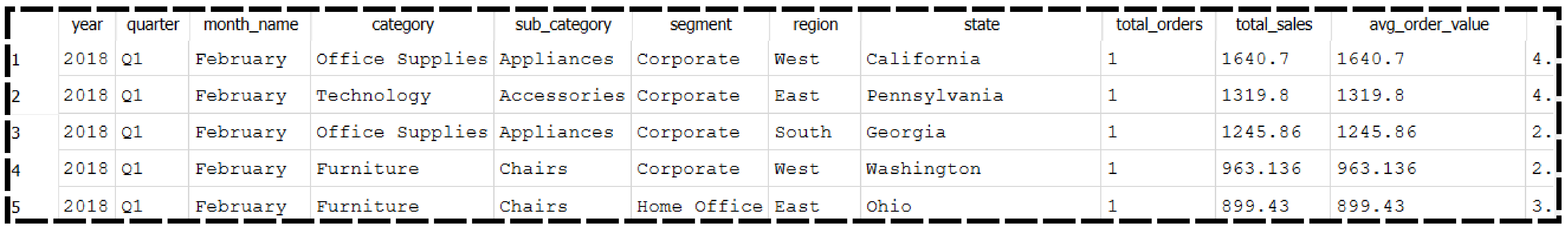
```sql
-- 4. MONTH-OVER-MONTH GROWTH (avec dim_date)
WITH MonthlySales AS (
    SELECT
        d.year AS Year,
        d.month AS Month,
        d.month_name AS Month_Name,
        SUM(f.sales_amount) AS Monthly_Sales
    FROM fact_sales f
    JOIN dim_date d ON f.date_key = d.date_key
    GROUP BY d.year, d.month
),
PreviousMonthSales AS (
    SELECT
        Year,
        Month,
        Month_Name,
        Monthly_Sales,
        LAG(Monthly_Sales) OVER (ORDER BY Year, Month) AS Previous_Month_Sales
    FROM MonthlySales
)
SELECT
    Year,
    Month,
    Month_Name,
    CASE
        WHEN Previous_Month_Sales = 0 OR Previous_Month_Sales IS NULL THEN 0
        ELSE (Monthly_Sales - Previous_Month_Sales) * 1.0 / Previous_Month_Sales
    END AS Sales_MoM
FROM PreviousMonthSales
ORDER BY Year, Month;
```



	Year	Month	Month_Name	Sales_MoM
1	2015	1	January	0
2	2015	2	February	-0.681825621209842
3	2015	3	March	11.2139637407265
4	2015	4	April	-0.494494119883823
5	2015	5	May	-0.152742113004135
6	2015	6	June	0.451636599311047
7	2015	7	July	-0.0157734934537476

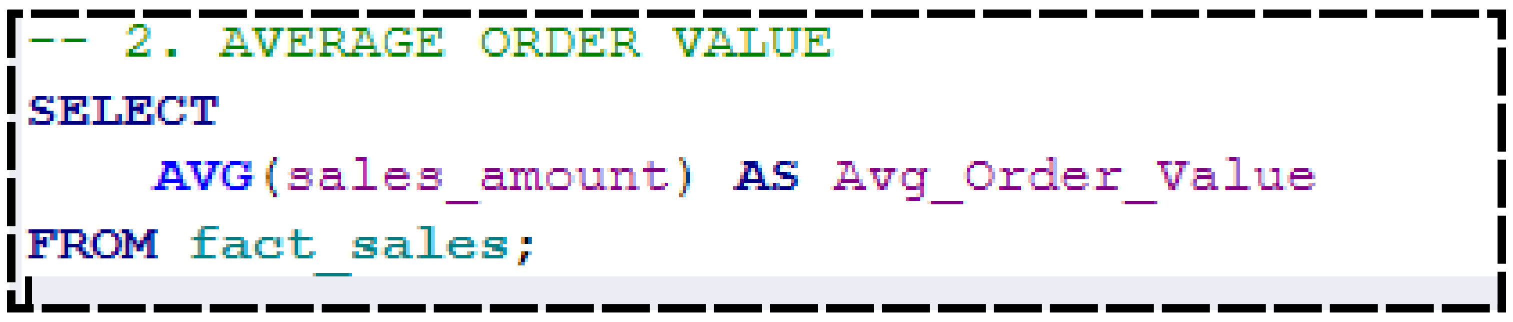


```sql
-- 9. REQUÊTE AVEC DIMENSIONS (Exemple: ventes par caté
SELECT
    d.year,
    d.quarter,
    d.month_name,
    p.category,
    p.sub_category,
    c.segment,
    r.region,
    r.state,
    COUNT(DISTINCT f.order_id) AS total_orders,
    SUM(f.sales_amount) AS total_sales,
    AVG(f.sales_amount) AS avg_order_value,
    AVG(f.shipping_days) AS avg_shipping_days
FROM fact_sales f
JOIN dim_date d ON f.date_key = d.date_key
JOIN dim_product p ON f.product_key = p.product_key
JOIN dim_customer c ON f.customer_key = c.customer_key
JOIN dim_region r ON f.region_key = r.region_key
GROUP BY
    d.year,
    d.quarter,
    d.month_name,
    p.category,
    p.sub_category,
    c.segment,
    r.region,
    r.state
ORDER BY
    d.year DESC,
    d.quarter,
```

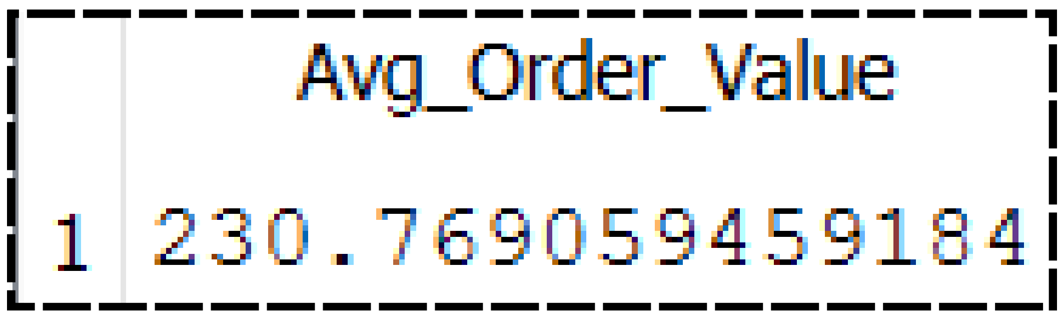


	year	quarter	month_name	category	sub_category	segment	region	state	total_orders	total_sales	avg_order_value	
1	2018	Q1	February	Office Supplies	Appliances	Corporate	West	California	1	1640.7	1640.7	4.
2	2018	Q1	February	Technology	Accessories	Corporate	East	Pennsylvania	1	1319.8	1319.8	4.
3	2018	Q1	February	Office Supplies	Appliances	Corporate	South	Georgia	1	1245.86	1245.86	2.
4	2018	Q1	February	Furniture	Chairs	Corporate	West	Washington	1	963.136	963.136	2.
5	2018	Q1	February	Furniture	Chairs	Home Office	East	Ohio	1	899.43	899.43	3.

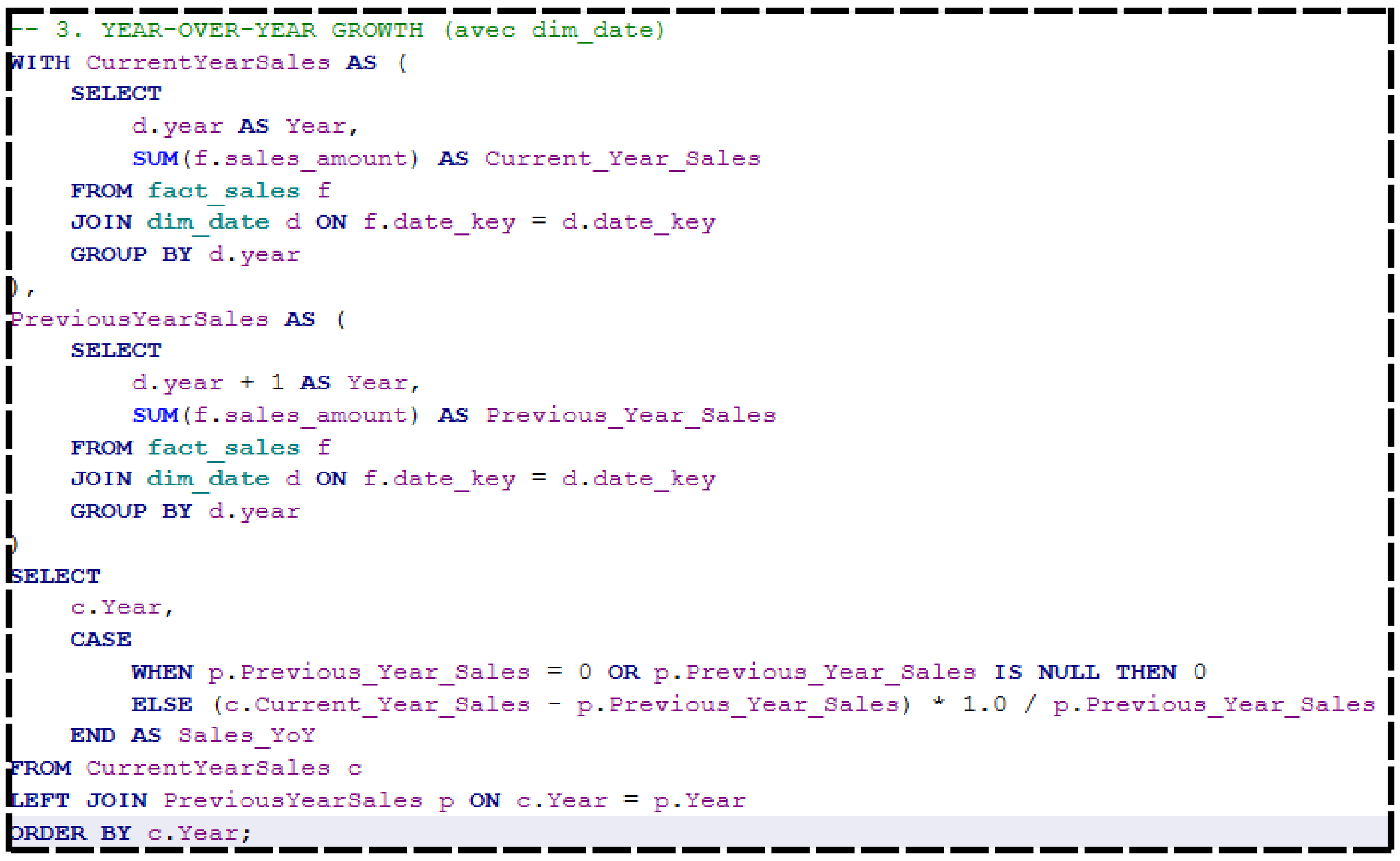
4. ROLL UP Operations



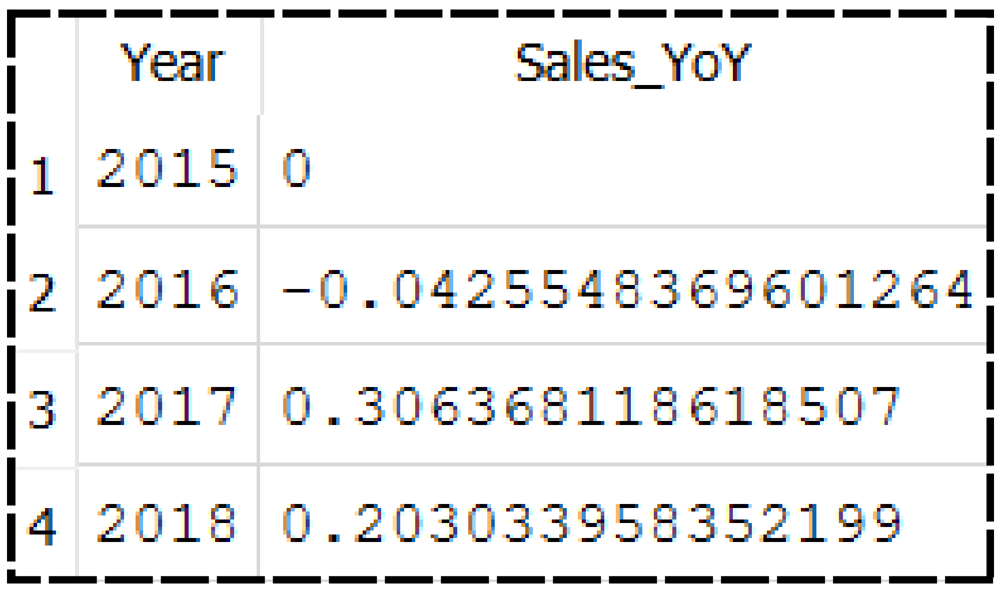
```sql
-- 2. AVERAGE ORDER VALUE
SELECT
    AVG(sales_amount) AS Avg_Order_Value
FROM fact_sales;
```



	Avg_Order_Value
1	230.769059459184



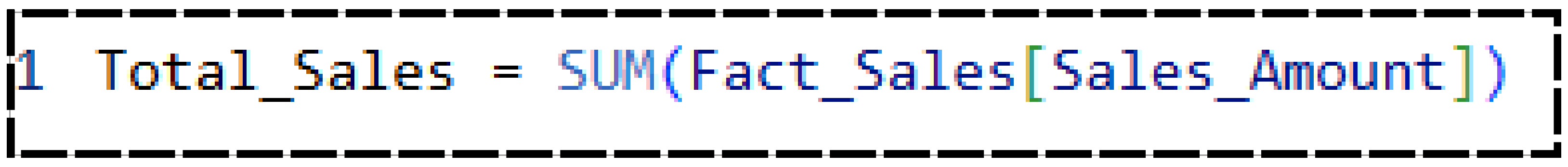
```sql
-- 3. YEAR-OVER-YEAR GROWTH (avec dim_date)
WITH CurrentYearSales AS (
    SELECT
        d.year AS Year,
        SUM(f.sales_amount) AS Current_Year_Sales
    FROM fact_sales f
    JOIN dim_date d ON f.date_key = d.date_key
    GROUP BY d.year
),
PreviousYearSales AS (
    SELECT
        d.year + 1 AS Year,
        SUM(f.sales_amount) AS Previous_Year_Sales
    FROM fact_sales f
    JOIN dim_date d ON f.date_key = d.date_key
    GROUP BY d.year
)
SELECT
    c.Year,
    CASE
        WHEN p.Previous_Year_Sales = 0 OR p.Previous_Year_Sales IS NULL THEN 0
        ELSE (c.Current_Year_Sales - p.Previous_Year_Sales) * 1.0 / p.Previous_Year_Sales
    END AS Sales_YoY
FROM CurrentYearSales c
LEFT JOIN PreviousYearSales p ON c.Year = p.Year
ORDER BY c.Year;
```



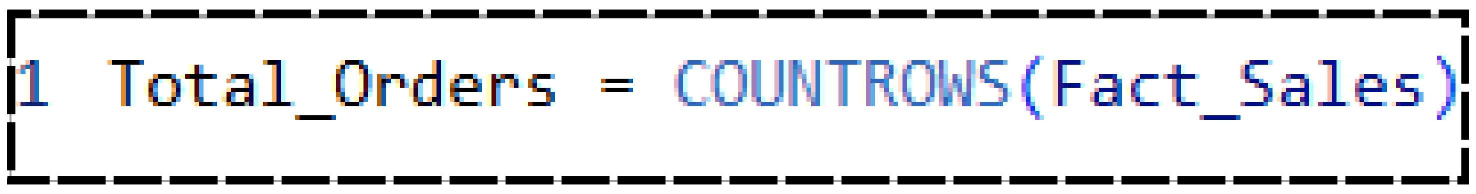
	Year	Sales_YoY
1	2015	0
2	2016	-0.0425548369601264
3	2017	0.306368118618507
4	2018	0.203033958352199

Phase 4 : Data visualization

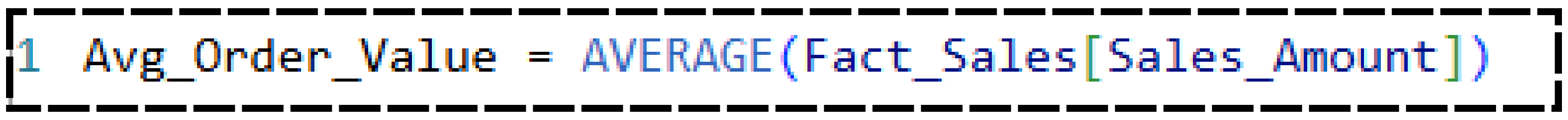
Step 1 :Dax Measures



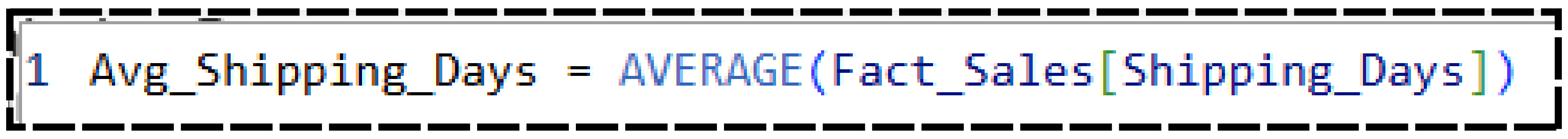
```
Total_Sales = SUM(Fact_Sales[Sales_Amount])
```



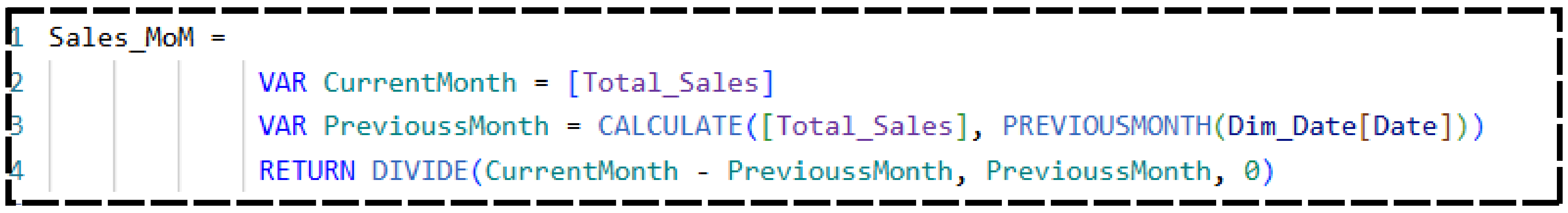
```
Total_Orders = COUNTROWS(Fact_Sales)
```



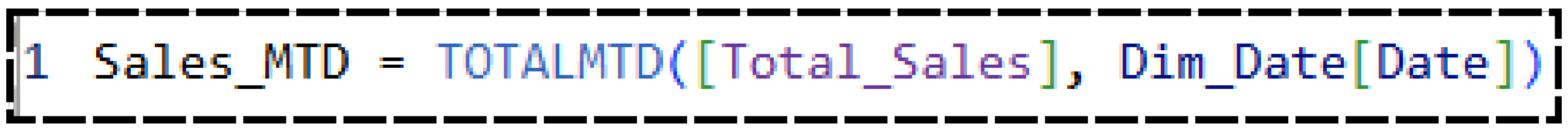
```
Avg_Order_Value = AVERAGE(Fact_Sales[Sales_Amount])
```



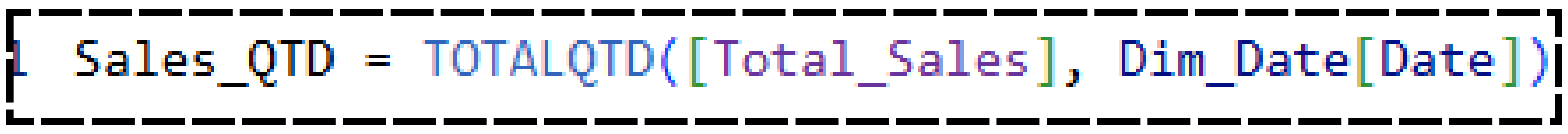
```
Avg_Shipping_Days = AVERAGE(Fact_Sales[Shipping_Days])
```



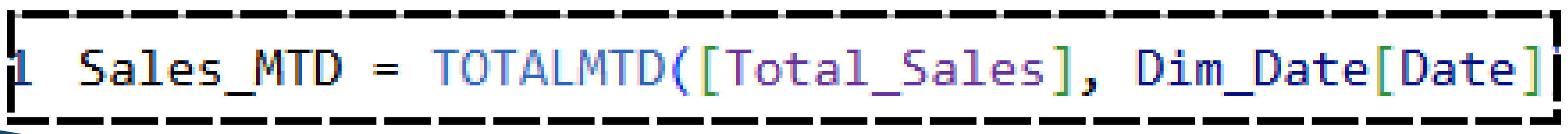
```
Sales_MoM =
            VAR CurrentMonth = [Total_Sales]
            VAR PreviousMonth = CALCULATE([Total_Sales], PREVIOUSMONTH(Dim_Date[Date]))
            RETURN DIVIDE(CurrentMonth - PreviousMonth, PreviousMonth, 0)
```



```
Sales_MTD = TOTALMTD([Total_Sales], Dim_Date[Date])
```



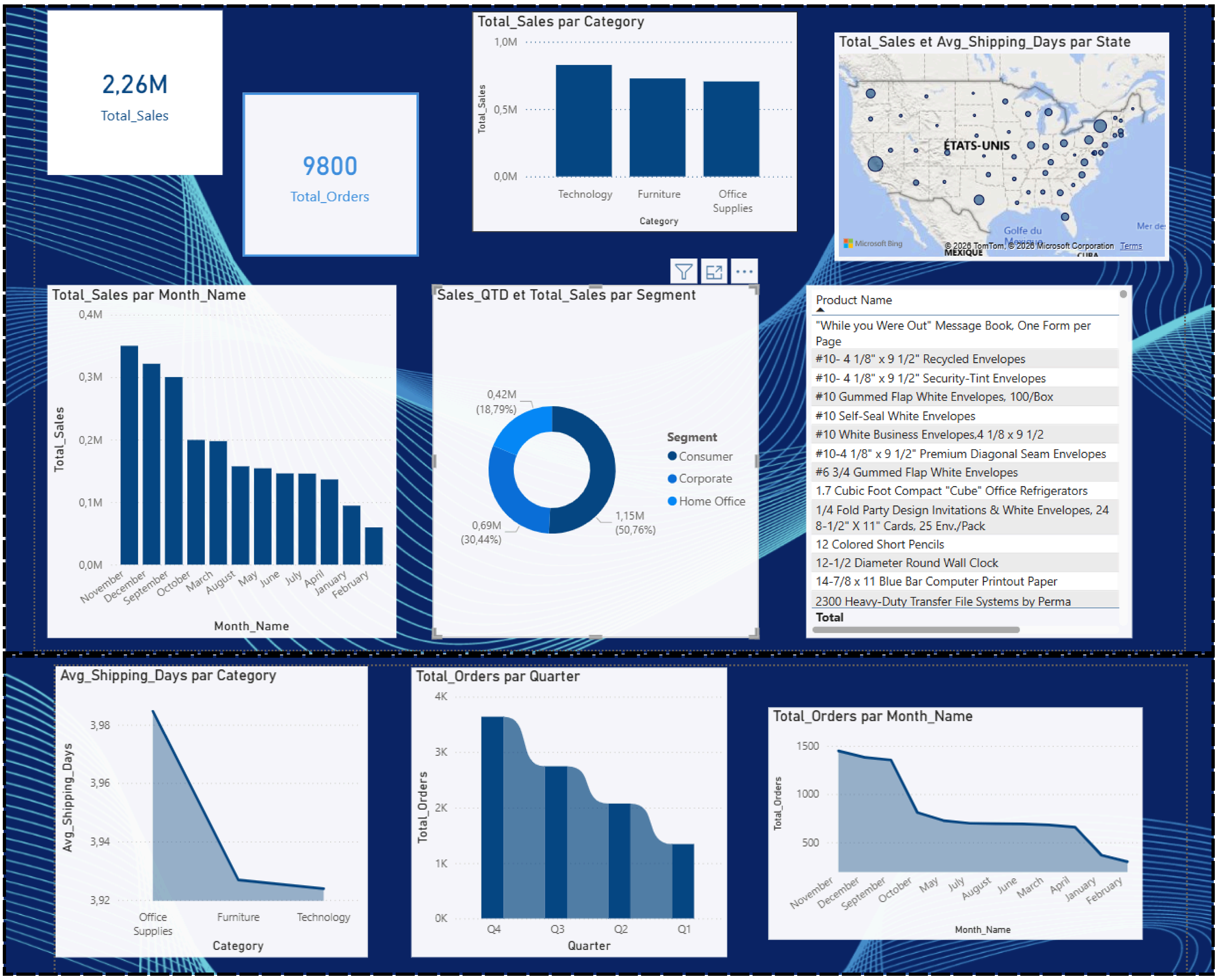
```
Sales_QTD = TOTALQTD([Total_Sales], Dim_Date[Date])
```



```
Sales_MTD = TOTALMTD([Total_Sales], Dim_Date[Date]
```

Step 2 : visualization

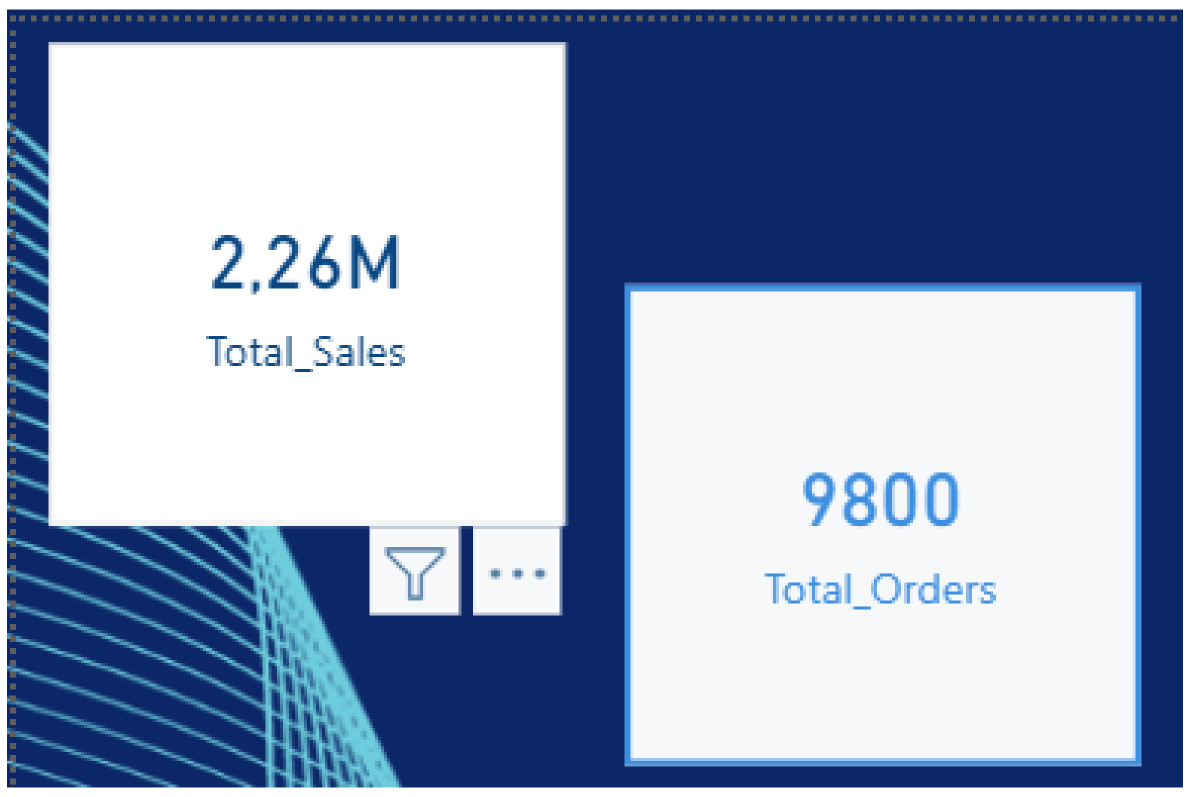
Dashboarding :



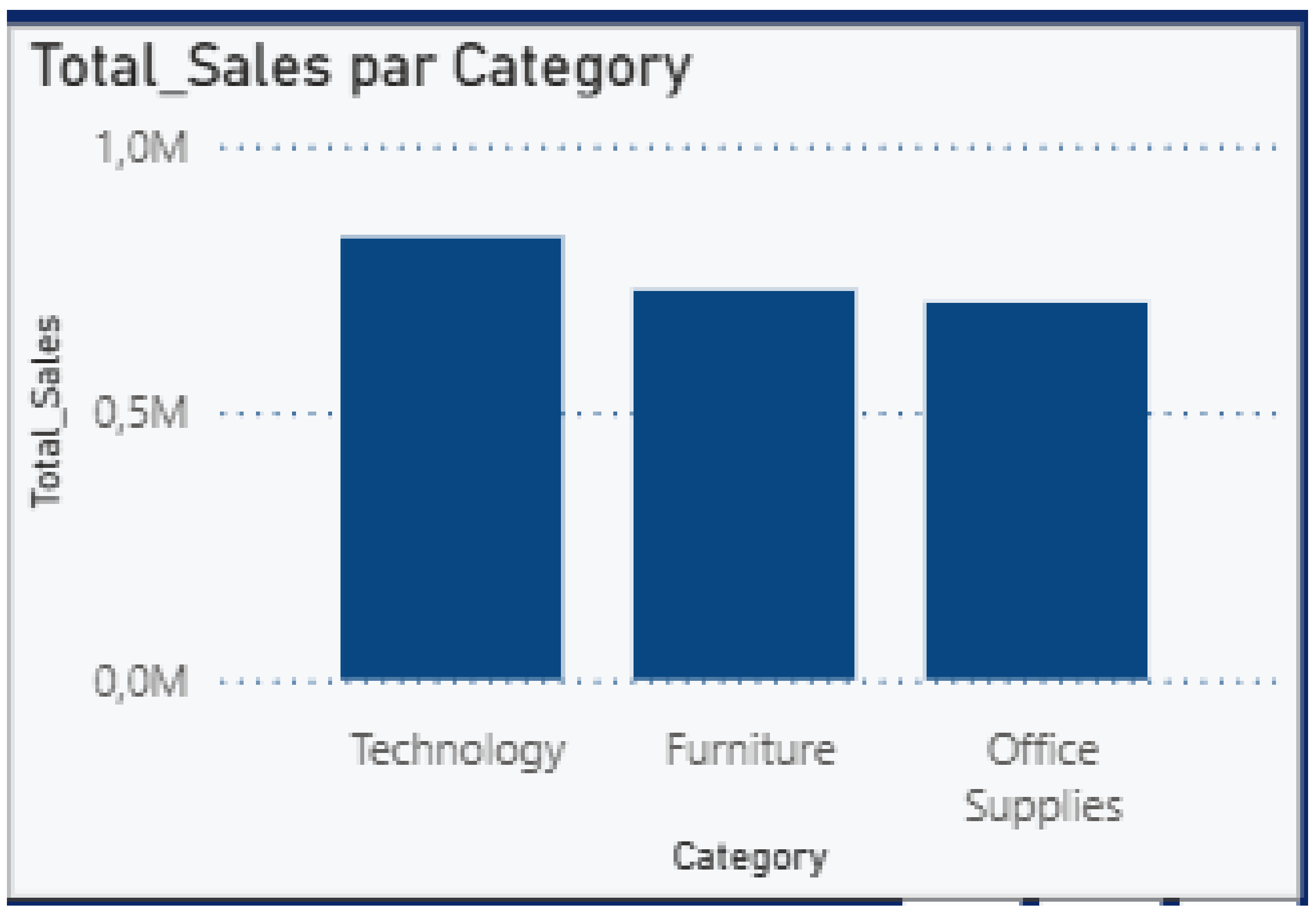
Product Name
"While you Were Out" Message Book, One Form per Page
#10- 4 1/8" x 9 1/2" Recycled Envelopes
#10- 4 1/8" x 9 1/2" Security-Tint Envelopes
#10 Gummed Flap White Envelopes, 100/Box
#10 Self-Seal White Envelopes
#10 White Business Envelopes,4 1/8 x 9 1/2
#10-4 1/8" x 9 1/2" Premium Diagonal Seam Envelopes
#6 3/4 Gummed Flap White Envelopes
1.7 Cubic Foot Compact "Cube" Office Refrigerators
1/4 Fold Party Design Invitations & White Envelopes, 24 8-1/2" X 11" Cards, 25 Env./Pack
12 Colored Short Pencils
12-1/2 Diameter Round Wall Clock
14-7/8 x 11 Blue Bar Computer Printout Paper
2300 Heavy-Duty Transfer File Systems by Perma
Total

INSIGHTS AND BUSINESS INTERPRETATION

This section presents the key insights derived from the Business Intelligence dashboards developed using the Sales Forecasting dataset. The analysis focuses on sales performance, customer segmentation, product categories, temporal trends, and geographical distribution in order to support data-driven decision-making

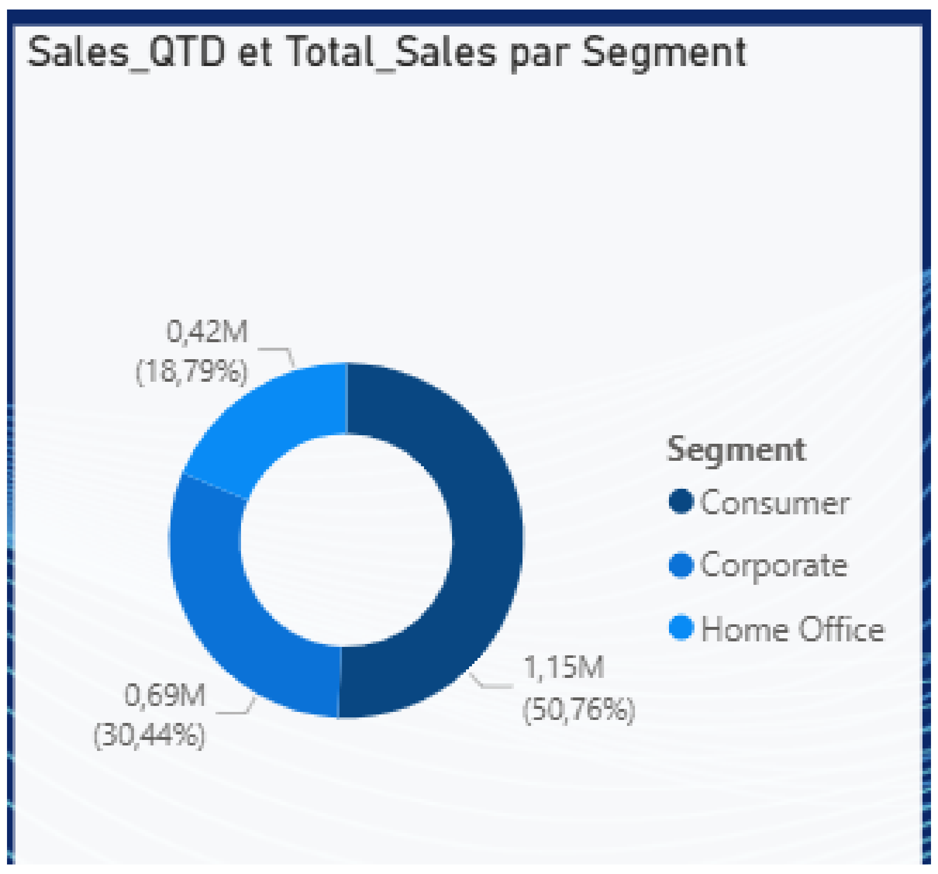


The analysis reveals a strong overall sales performance, with total sales reaching approximately 2.26 million across 9,800 orders. This volume indicates a healthy level of business activity and validates the relevance of the dataset for analytical and strategic purposes.



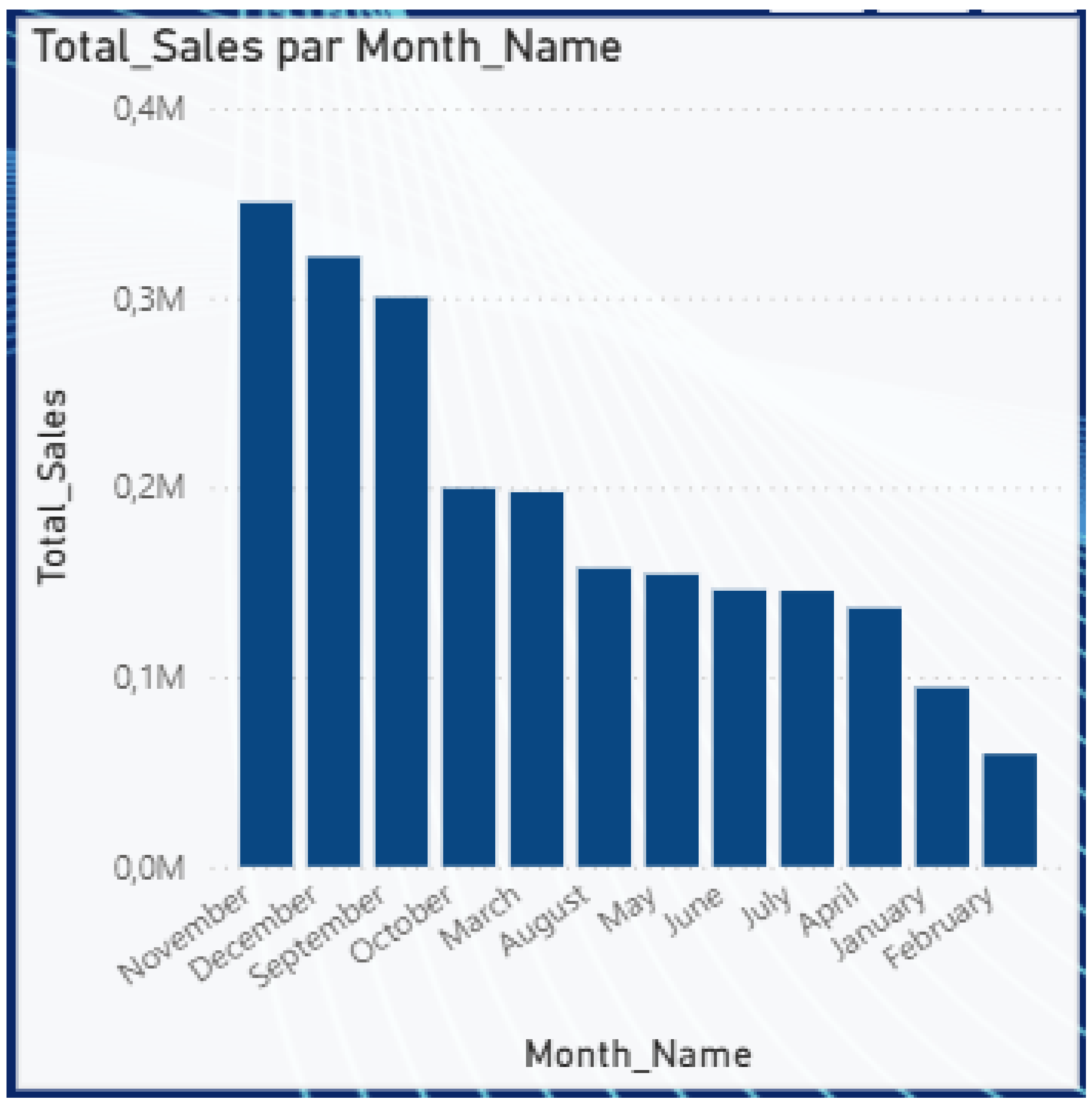
Among the three main product categories, Technology generates the highest total sales, followed by Furniture and Office Supplies. Technology products significantly outperform other categories, highlighting their importance as a key revenue driver.

Insights :This result suggests that customer demand is concentrated in high-value and innovation-driven products, emphasizing the strategic role of technology offerings in sustaining revenue growth.



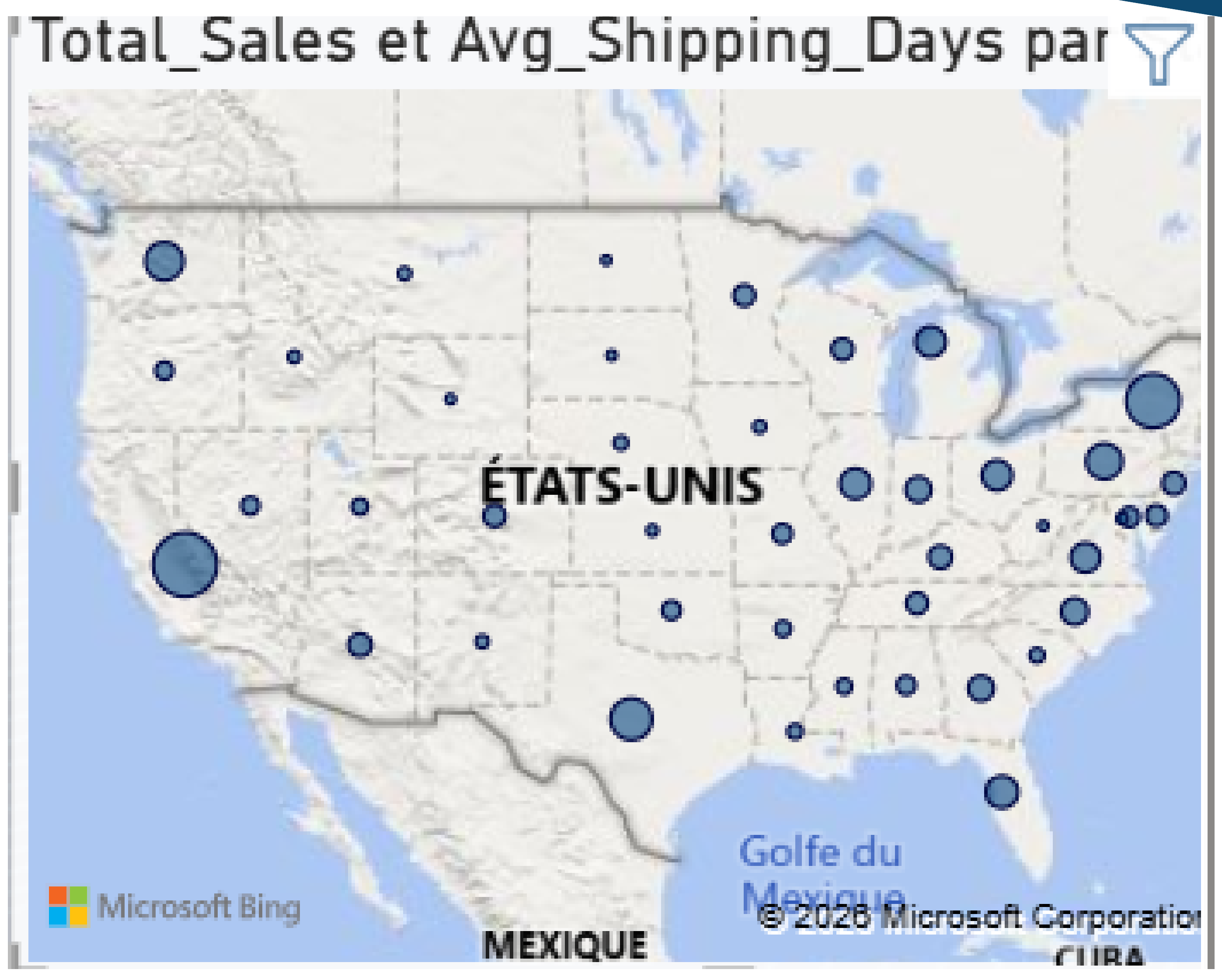
The Consumer segment contributes approximately 50.8% of total sales, making it the dominant customer group. The Corporate segment accounts for about 30.4%, while the Home Office segment contributes the remaining 18.8%.

Insights : The heavy reliance on consumer customers implies that the company’s revenue is sensitive to changes in individual purchasing behavior, economic conditions, and seasonal demand patterns



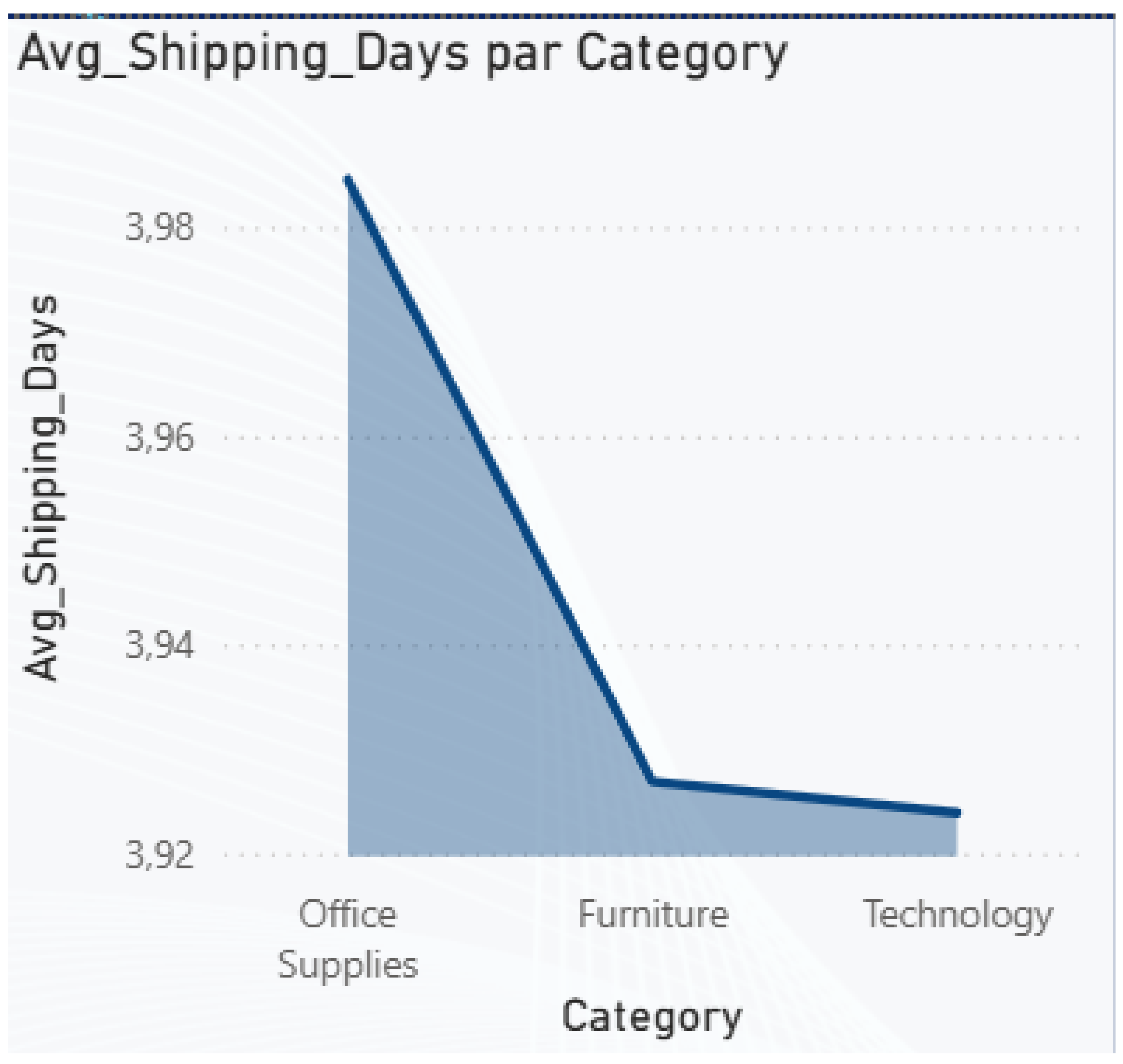
Monthly sales analysis reveals clear seasonality. Sales peak during November, December, and September, while January and February record the lowest sales levels.

Insights : This pattern indicates that year-end demand, likely driven by promotions and holiday shopping, plays a major role in revenue generation. Conversely, early-year periods represent a slowdown in customer purchasing activity


Total_Sales et Avg_Shipping_Days par
ÉTATS-UNIS
Golfe du Mexique
MEXIQUE
CUBA
Microsoft Bing
© 2026 Microsoft Corporation

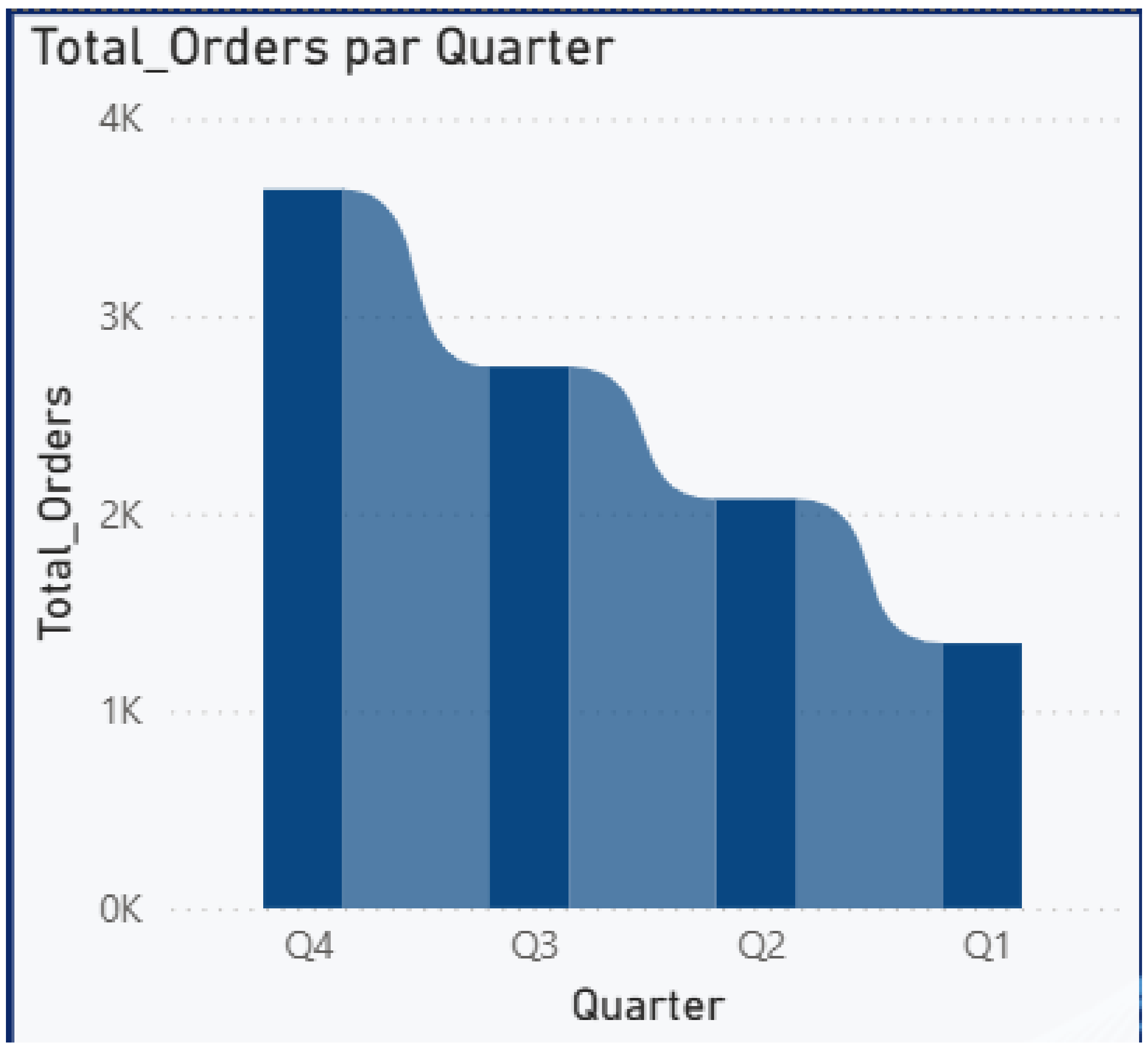
Sales are concentrated in highly populated states like California, Texas, New York, and Illinois. Some states in the Midwest and Northeast show lower sales volumes.

Insights : Consider targeted marketing or sales campaigns in underperforming regions. Investigate whether logistics, awareness, or distribution is limiting sales in those areas.
This finding highlights potential inefficiencies in logistics and distribution networks, as well as opportunities for regional market expansion.



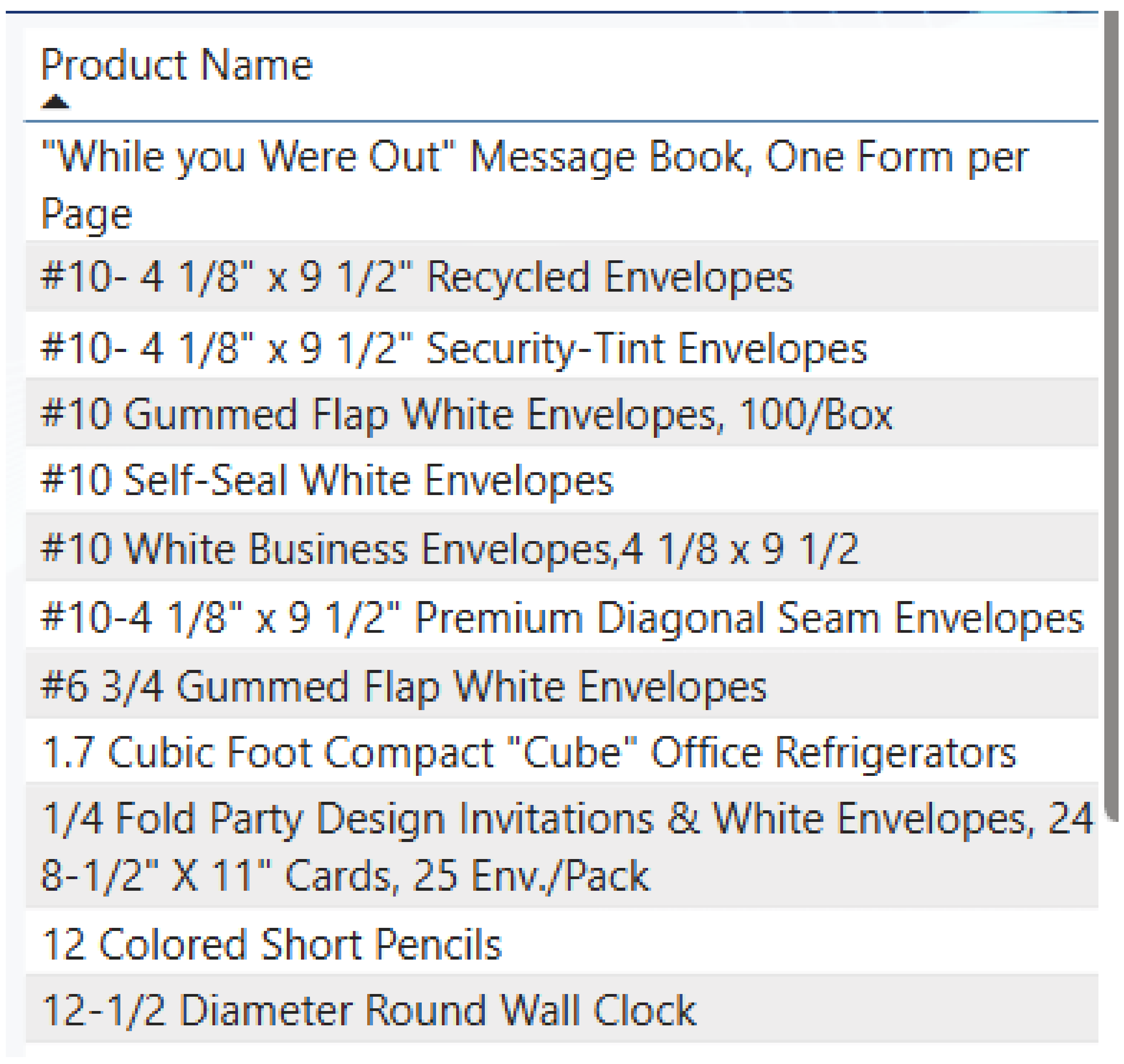
The analysis of average shipping days by product category indicates observable differences in delivery performance across categories. Office Supplies exhibit the longest average shipping duration, followed by Furniture, while Technology products have the shortest average shipping time.

Insights : This variation can be explained by differences in product characteristics and logistics complexity. Office Supplies often include low-value, high-volume items that may be deprioritized in shipping operations, while Furniture products typically involve larger sizes and handling constraints. In contrast, Technology products benefit from standardized packaging and higher operational priority, leading to faster delivery times.
Shipping performance directly influences customer satisfaction and repeat purchasing behavior. Longer delivery times for Office Supplies may negatively impact customer perception, particularly for Consumer and Corporate segments that value fast fulfillment. This finding highlights the need for differentiated logistics strategies based on product category



Quarterly analysis of total orders reveals a clear declining trend from **Q4** to **Q1**. The highest number of orders is observed in **Q4**, followed by **Q3**, **Q2**, and finally **Q1**, which records the lowest order volume.

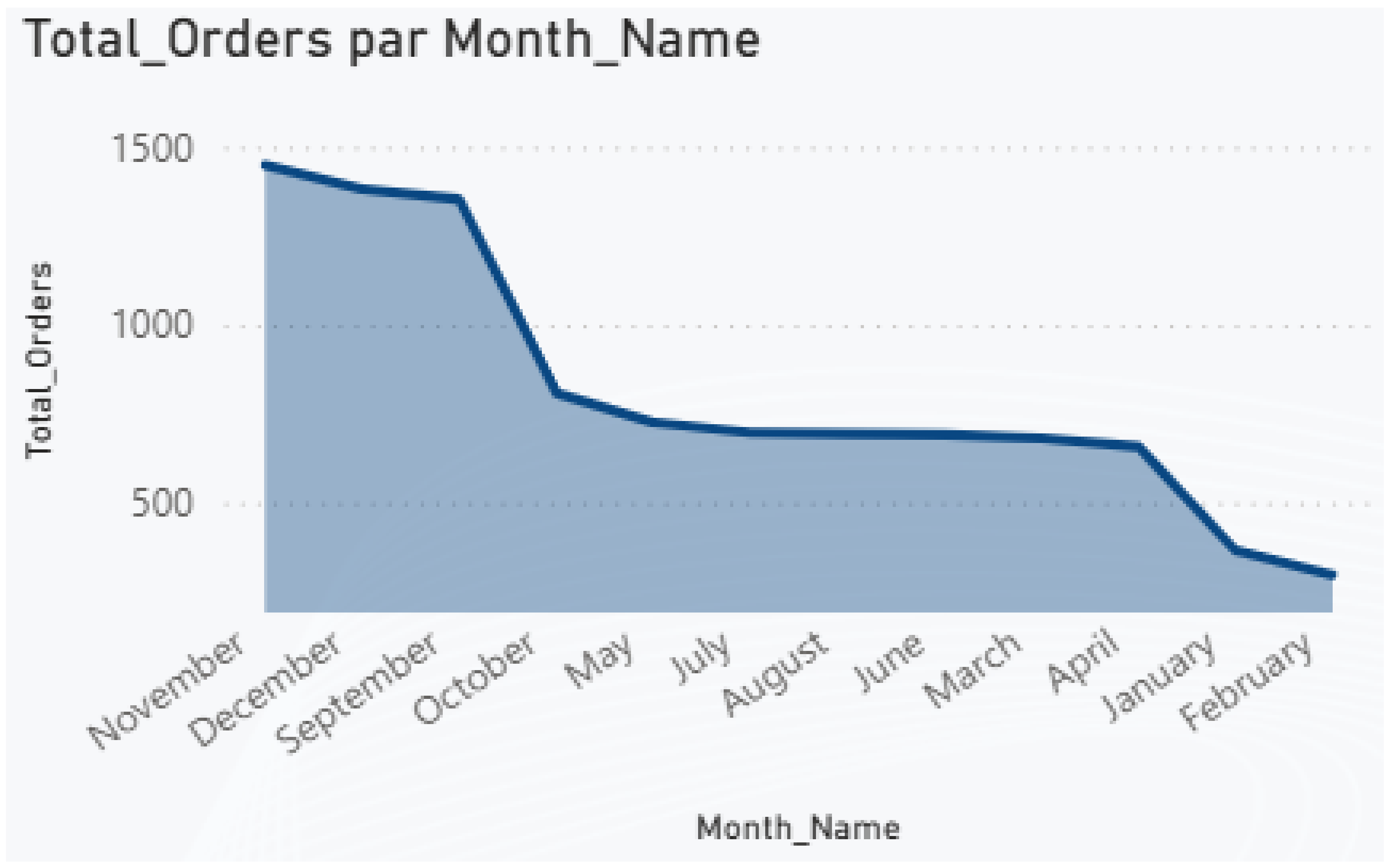
Insights :This pattern confirms the presence of strong seasonality in customer purchasing behavior. The peak in Q4 is likely driven by end-of-year demand, promotional campaigns, and budget consumption by corporate customers. Conversely, the lower performance in Q1 may reflect post-holiday spending reductions and conservative purchasing behavior.
Understanding quarterly demand cycles is essential for effective resource planning. Failure to anticipate peak periods may result in inventory shortages and operational bottlenecks, while overestimating demand in low-activity quarters can increase holding and operational costs.



Product Name
"While you Were Out" Message Book, One Form per Page
#10- 4 1/8" x 9 1/2" Recycled Envelopes
#10- 4 1/8" x 9 1/2" Security-Tint Envelopes
#10 Gummed Flap White Envelopes, 100/Box
#10 Self-Seal White Envelopes
#10 White Business Envelopes,4 1/8 x 9 1/2
#10-4 1/8" x 9 1/2" Premium Diagonal Seam Envelopes
#6 3/4 Gummed Flap White Envelopes
1.7 Cubic Foot Compact "Cube" Office Refrigerators
1/4 Fold Party Design Invitations & White Envelopes, 24 8-1/2" X 11" Cards, 25 Env./Pack
12 Colored Short Pencils
12-1/2 Diameter Round Wall Clock

The product-level analysis reveals a strong concentration of sales among a limited number of products. High-value items, particularly within the Technology category (such as office equipment and 3D printers), contribute disproportionately to total sales. In contrast, most Office Supplies products generate relatively low individual sales despite their frequent presence in orders.

Insights : This distribution reflects a long-tail sales pattern, where a small subset of products drives the majority of revenue while the remaining products play a supporting role.
The concentration of sales in a few high-performing products increases revenue efficiency but also exposes the organization to dependency risk. Product-level monitoring is therefore essential for balancing revenue maximization with portfolio stability



The chart shows that orders are highest at the end of the year (November–December), then drop after that. During the middle of the year, orders stay fairly stable, and the lowest orders are in January and February. This means demand is strongest at year-end and weakest at the beginning of the year.

Insights : This pattern highlights the need to scale inventory, staffing, and marketing from September onward, implement post-holiday promotions in early Q1 to stimulate demand, and align operational and financial planning with these predictable fluctuations to maximize revenue during peaks and maintain stability during slower periods.

Business Recommendations

Based on the analytical findings, the following recommendations are proposed:

1. Strengthen Investment in Technology Products: The organization should prioritize Technology products through targeted marketing campaigns and inventory optimization, as they represent the highest revenue contribution.
2. Enhance Consumer Segment Retention Strategies: Given the dominance of the Consumer segment, loyalty programs and personalized promotions should be implemented to improve customer retention and lifetime value.
3. Optimize Seasonal Planning and Inventory Management: Inventory and logistics capacity should be increased ahead of peak months, particularly during the fourth quarter, to avoid stock shortages and missed sales opportunities.
4. Improve Logistics and Distribution Efficiency: Regions with longer shipping times should be analyzed in detail to identify bottlenecks and improve delivery performance through alternative carriers or regional distribution centers.
5. Integrate Sales Forecasting into Decision-Making: Forecasting models based on historical sales trends should be used to support demand planning, budgeting, and strategic decisionmaking.
6. Improve Shipping Efficiency for Office Supplies: The organization should review fulfillment processes for Office Supplies, including warehouse allocation and carrier selection, to reduce delivery lead times.
7. Adopt Category-Based Logistics Strategies: Different shipping models should be applied depending on product category, prioritizing high-demand and high-value items such as Technology products.
8. Scale Logistics Capacity During Q4: Additional logistics resources should be allocated during Q4 to accommodate the surge in order volume and maintain service quality.
9. Introduce Demand-Stimulation Strategies in Q1: Promotional campaigns, discounts, or bundled offers could be used to stimulate demand during low-activity quarters.
10. Align Workforce and Inventory Planning with Quarterly Trends: Staffing levels and inventory replenishment schedules should reflect observed seasonal order patterns to optimize operational efficiency.

Limitations of the Study

Despite the valuable insights generated, this study has several limitations:

- The analysis focuses primarily on sales volume, without incorporating profitability or margin indicators.
- Customer-level demographic and behavioral attributes are not available, limiting deeper segmentation.
- The dataset contains historical data only, without real-time updates or external economic factors.
- Shipping analysis is limited to average delivery time and does not include logistics costs or carrier-level performance.
- Marketing and promotional data are not included, preventing evaluation of campaign effectiveness.

Future Research and Improvements

Future enhancements to this BI solution could include:

- Integration of profit and cost metrics to support profitability analysis.
- Application of advanced forecasting techniques such as ARIMA or Prophet models.
- Implementation of customer segmentation models (e.g., RFM or clustering).
- Development of drill-through dashboards for detailed regional and product analysis.
- Inclusion of operational KPIs such as delivery cost, return rates, and service-level performance

Conclusion

This study demonstrates the effective application of Business Intelligence techniques in transforming retail sales data into actionable analytical insights. Through systematic data preparation, dimensional modeling, and data visualization, the project provides a comprehensive analysis of sales performance, customer behavior, product categories, and seasonal trends.
The proposed BI solution enables stakeholders to monitor key performance indicators in real time, identify high-performing products and regions, analyze seasonal demand patterns, and optimize shipping strategies.

The findings highlight the strong revenue contribution of Technology products, the dominant role of the Consumer segment, clear sales peaks during the fourth quarter, and opportunities to improve logistics performance, particularly for Office Supplies.

Overall, the BI solution enhances analytical transparency and supports evidence-based decision-making, while establishing a solid foundation for future improvements in sales forecasting, performance evaluation, and strategic planning.

MADE ON 21-01-2026

MARIEM SOLTANI

RIHEM ABASSI