Cumulative_Analysis_Visualization

September 21, 2025

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sqlalchemy import create_engine
from urllib.parse import quote_plus

# Set a clean style for the plots
sns.set_style("whitegrid")
plt.style.use("seaborn-v0_8-deep")

# ========================================
# === Database Connection and Data Loading
# ========================================

# Database credentials
user = "root"
password = "Root7878"
host = "localhost"
port = 3306
database = "DataWarehouse"

# Encode password safely
password = quote_plus(password)

# Create SQLAlchemy engine
try:
    engine = create_engine(f"mysql+pymysql://{user}:{password}@{host}:{port}/
 ↪{database}")

    # SQL query to get raw data
    raw_sql_query = """
    SELECT
        order_date,
        sales_amount
    FROM fact_sales
    WHERE order_date IS NOT NULL;
    """
```

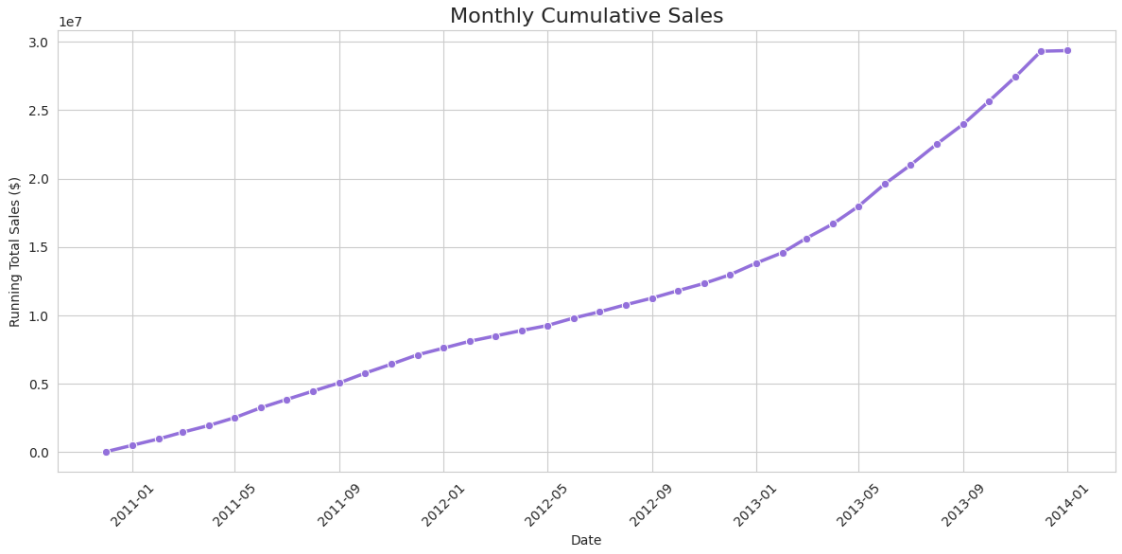
```
    df_raw = pd.read_sql(raw_sql_query, engine)

    # ==================================
    # === Data Transformations in Pandas
    # ==================================
    df_raw['order_date'] = pd.to_datetime(df_raw['order_date'])

    # Monthly Cumulative Sales
    df_monthly = df_raw.groupby(pd.Grouper(key='order_date',␣
 ↪freq='MS'))['sales_amount'].sum().reset_index()
    df_monthly.rename(columns={'order_date': 'month_start', 'sales_amount':␣
 ↪'total_sales'}, inplace=True)
    df_monthly['running_total_sales'] = df_monthly['total_sales'].cumsum()

    # Yearly Cumulative Sales
    df_yearly = df_raw.groupby(pd.Grouper(key='order_date',␣
 ↪freq='YS'))['sales_amount'].sum().reset_index()
    df_yearly.rename(columns={'order_date': 'year_start', 'sales_amount':␣
 ↪'total_sales'}, inplace=True)
    df_yearly['running_total_sales'] = df_yearly['total_sales'].cumsum()

    print("Monthly Cumulative DataFrame Head:")
    print(df_monthly.head())
    print("-" * 50)
    print("Yearly Cumulative DataFrame Head:")
    print(df_yearly.head())
    print("-" * 50)

except Exception as e:
    print(f"Error connecting to the database or loading data: {e}")
    print("Please ensure your database credentials are correct and the database␣
 ↪is running.")
    df_monthly = pd.DataFrame()
    df_yearly = pd.DataFrame()
```

```
Monthly Cumulative DataFrame Head:
  month_start  total_sales  running_total_sales
0  2010-12-01        43419                43419
1  2011-01-01       469795               513214
2  2011-02-01       466307               979521
3  2011-03-01       485165              1464686
4  2011-04-01       502042              1966728
--------------------------------------------------
Yearly Cumulative DataFrame Head:
  year_start  total_sales  running_total_sales
0 2010-01-01        43419                43419
1 2011-01-01      7075088              7118507
```

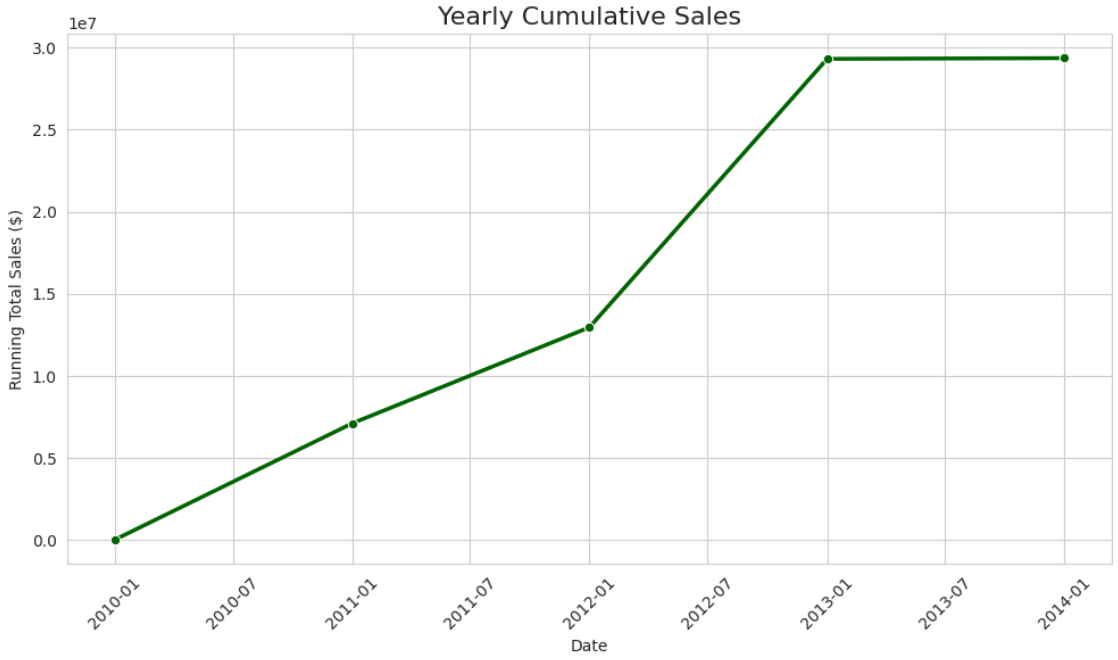
```
2 2012-01-01       5842231              12960738
3 2013-01-01      16344878              29305616
4 2014-01-01         45642              29351258
--------------------------------------------------
```

[ ]:
```
# ================================
# === Data Visualizations
# ================================
```

[2]:
```
if not df_monthly.empty and not df_yearly.empty:

    # --- 1. Monthly Cumulative Sales ---
    plt.figure(figsize=(12, 6))
    sns.lineplot(
        data=df_monthly,
        x='month_start',
        y='running_total_sales',
        marker='o',
        linewidth=2.5,
        color='mediumpurple'
    )
    plt.title('Monthly Cumulative Sales', fontsize=16)
    plt.xlabel('Date')
    plt.ylabel('Running Total Sales ($)')
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()

else:
    print("One or both DataFrames are empty. No visualizations will be␣
 ↪generated.")
```



[3]:
```
if not df_monthly.empty and not df_yearly.empty:

    # --- 2. Yearly Cumulative Sales ---
    plt.figure(figsize=(10, 6))
    sns.lineplot(
        data=df_yearly,
        x='year_start',
        y='running_total_sales',
        marker='o',
        linewidth=2.5,
        color='darkgreen'
    )
    plt.title('Yearly Cumulative Sales', fontsize=16)
    plt.xlabel('Date')
    plt.ylabel('Running Total Sales ($)')
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()

else:
    print("One or both DataFrames are empty. No visualizations will be␣
 ↪generated.")
```



[ ]: