In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from ydata_profiling import ProfileReport
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import LabelEncoder
from imblearn.over_sampling import SMOTE
from collections import Counter
import seaborn as sns
import warnings
warnings.simplefilter(action = 'ignore')
```

In [2]:
```
df = pd.read_csv(r"C:\Users\HP\Downloads\Superstore 20.csv")
df.head(10)
```

Out[2]:

	Row_ID	Order_ID	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	
0	1	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	C
1	2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	C
2	3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	C
3	4	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	C
4	5	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	C
5	6	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	C
6	7	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	C
7	8	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	C
8	9	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	C
9	10	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	C

10 rows × 21 columns

In [3]:
```
pip install ydata-profiling
```

```
Requirement already satisfied: ydata-profiling in c:\users\hp\anaconda3\lib\site-pac
kages (4.12.2)Note: you may need to restart the kernel to use updated packages.

Requirement already satisfied: scipy<1.16,>=1.4.1 in c:\users\hp\anaconda3\lib\site-
packages (from ydata-profiling) (1.13.1)
Requirement already satisfied: pandas!=1.4.0,<3,>1.1 in c:\users\hp\anaconda3\lib\si
te-packages (from ydata-profiling) (2.2.2)
Requirement already satisfied: matplotlib>=3.5 in c:\users\hp\anaconda3\lib\site-pac
kages (from ydata-profiling) (3.9.2)
Requirement already satisfied: pydantic>=2 in c:\users\hp\anaconda3\lib\site-package
s (from ydata-profiling) (2.8.2)
Requirement already satisfied: PyYAML<6.1,>=5.0.0 in c:\users\hp\anaconda3\lib\site-
packages (from ydata-profiling) (6.0.1)
Requirement already satisfied: jinja2<3.2,>=2.11.1 in c:\users\hp\anaconda3\lib\site
-packages (from ydata-profiling) (3.1.4)
Requirement already satisfied: visions<0.8.0,>=0.7.5 in c:\users\hp\anaconda3\lib\si
te-packages (from visions[type_image_path]<0.8.0,>=0.7.5->ydata-profiling) (0.7.6)
Requirement already satisfied: numpy<2.2,>=1.16.0 in c:\users\hp\anaconda3\lib\site-
packages (from ydata-profiling) (1.26.4)
Requirement already satisfied: htmlmin==0.1.12 in c:\users\hp\anaconda3\lib\site-pac
kages (from ydata-profiling) (0.1.12)
Requirement already satisfied: phik<0.13,>=0.11.1 in c:\users\hp\anaconda3\lib\site-
packages (from ydata-profiling) (0.12.4)
Requirement already satisfied: requests<3,>=2.24.0 in c:\users\hp\anaconda3\lib\site
-packages (from ydata-profiling) (2.32.3)
Requirement already satisfied: tqdm<5,>=4.48.2 in c:\users\hp\anaconda3\lib\site-pac
kages (from ydata-profiling) (4.66.5)
Requirement already satisfied: seaborn<0.14,>=0.10.1 in c:\users\hp\anaconda3\lib\si
te-packages (from ydata-profiling) (0.13.2)
Requirement already satisfied: multimethod<2,>=1.4 in c:\users\hp\anaconda3\lib\site
-packages (from ydata-profiling) (1.12)
Requirement already satisfied: statsmodels<1,>=0.13.2 in c:\users\hp\anaconda3\lib\s
ite-packages (from ydata-profiling) (0.14.2)
Requirement already satisfied: typeguard<5,>=3 in c:\users\hp\anaconda3\lib\site-pac
kages (from ydata-profiling) (4.4.1)
Requirement already satisfied: imagehash==4.3.1 in c:\users\hp\anaconda3\lib\site-pa
ckages (from ydata-profiling) (4.3.1)
Requirement already satisfied: wordcloud>=1.9.3 in c:\users\hp\anaconda3\lib\site-pa
ckages (from ydata-profiling) (1.9.4)
Requirement already satisfied: dacite>=1.8 in c:\users\hp\anaconda3\lib\site-package
s (from ydata-profiling) (1.8.1)
Requirement already satisfied: PyWavelets in c:\users\hp\anaconda3\lib\site-packages
(from imagehash==4.3.1->ydata-profiling) (1.7.0)
Requirement already satisfied: pillow in c:\users\hp\anaconda3\lib\site-packages (fr
om imagehash==4.3.1->ydata-profiling) (10.4.0)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\hp\anaconda3\lib\site-pac
kages (from jinja2<3.2,>=2.11.1->ydata-profiling) (2.1.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\hp\anaconda3\lib\site-pa
ckages (from matplotlib>=3.5->ydata-profiling) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\hp\anaconda3\lib\site-packag
es (from matplotlib>=3.5->ydata-profiling) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\hp\anaconda3\lib\site-p
ackages (from matplotlib>=3.5->ydata-profiling) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\hp\anaconda3\lib\site-p
ackages (from matplotlib>=3.5->ydata-profiling) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\hp\anaconda3\lib\site-pac
```

```
kages (from matplotlib>=3.5->ydata-profiling) (24.1)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\hp\anaconda3\lib\site-pa
ckages (from matplotlib>=3.5->ydata-profiling) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\hp\anaconda3\lib\sit
e-packages (from matplotlib>=3.5->ydata-profiling) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\hp\anaconda3\lib\site-packag
es (from pandas!=1.4.0,<3,>1.1->ydata-profiling) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\hp\anaconda3\lib\site-pack
ages (from pandas!=1.4.0,<3,>1.1->ydata-profiling) (2023.3)
Requirement already satisfied: joblib>=0.14.1 in c:\users\hp\anaconda3\lib\site-pack
ages (from phik<0.13,>=0.11.1->ydata-profiling) (1.4.2)
Requirement already satisfied: annotated-types>=0.4.0 in c:\users\hp\anaconda3\lib\s
ite-packages (from pydantic>=2->ydata-profiling) (0.6.0)
Requirement already satisfied: pydantic-core==2.20.1 in c:\users\hp\anaconda3\lib\si
te-packages (from pydantic>=2->ydata-profiling) (2.20.1)
Requirement already satisfied: typing-extensions>=4.6.1 in c:\users\hp\anaconda3\lib
\site-packages (from pydantic>=2->ydata-profiling) (4.11.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\hp\anaconda3\lib
\site-packages (from requests<3,>=2.24.0->ydata-profiling) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\hp\anaconda3\lib\site-packag
es (from requests<3,>=2.24.0->ydata-profiling) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\hp\anaconda3\lib\site-
packages (from requests<3,>=2.24.0->ydata-profiling) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\hp\anaconda3\lib\site-
packages (from requests<3,>=2.24.0->ydata-profiling) (2024.12.14)
Requirement already satisfied: patsy>=0.5.6 in c:\users\hp\anaconda3\lib\site-packag
es (from statsmodels<1,>=0.13.2->ydata-profiling) (0.5.6)
Requirement already satisfied: colorama in c:\users\hp\anaconda3\lib\site-packages
(from tqdm<5,>=4.48.2->ydata-profiling) (0.4.6)
Requirement already satisfied: attrs>=19.3.0 in c:\users\hp\anaconda3\lib\site-packa
ges (from visions<0.8.0,>=0.7.5->visions[type_image_path]<0.8.0,>=0.7.5->ydata-profi
ling) (23.1.0)
Requirement already satisfied: networkx>=2.4 in c:\users\hp\anaconda3\lib\site-packa
ges (from visions<0.8.0,>=0.7.5->visions[type_image_path]<0.8.0,>=0.7.5->ydata-profi
ling) (3.3)
Requirement already satisfied: six in c:\users\hp\anaconda3\lib\site-packages (from
patsy>=0.5.6->statsmodels<1,>=0.13.2->ydata-profiling) (1.16.0)
```

In [4]:
```
profile = ProfileReport(df, explorative=True)
profile.to_notebook_iframe()
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

Pandas Profiling Report

Overview

Brought to you by YData

Overview | Alerts 12 | Reproduction

Dataset statistics

Number of variables	21
Number of observations	9994
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	9.2 MiB
Average record size in memory	969.6 B

Variable types

Numeric	6
Text	6
DateTime	2
Categorical	7

Variables

Select Columns

In [5]:
```
df.shape
```

Out[5]: (9994, 21)

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Row_ID         9994 non-null   int64  
 1   Order_ID       9994 non-null   object 
 2   Order_Date     9994 non-null   object 
 3   Ship_Date      9994 non-null   object 
 4   Ship_Mode      9994 non-null   object 
 5   Customer_ID    9994 non-null   object 
 6   Customer_Name  9994 non-null   object 
 7   Segment        9994 non-null   object 
 8   Country        9994 non-null   object 
 9   City           9994 non-null   object 
 10  State          9994 non-null   object 
 11  Postal_Code    9994 non-null   int64  
 12  Region         9994 non-null   object 
 13  Product_ID     9994 non-null   object 
 14  Category       9994 non-null   object 
 15  Sub_Category   9994 non-null   object 
 16  Product_Name   9994 non-null   object 
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64  
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
dtypes: float64(3), int64(3), object(15)
memory usage: 1.6+ MB
```

In [7]:
```
print(df.columns)
```

```
Index(['Row_ID', 'Order_ID', 'Order_Date', 'Ship_Date', 'Ship_Mode',
       'Customer_ID', 'Customer_Name', 'Segment', 'Country', 'City', 'State',
       'Postal_Code', 'Region', 'Product_ID', 'Category', 'Sub_Category',
       'Product_Name', 'Sales', 'Quantity', 'Discount', 'Profit'],
      dtype='object')
```

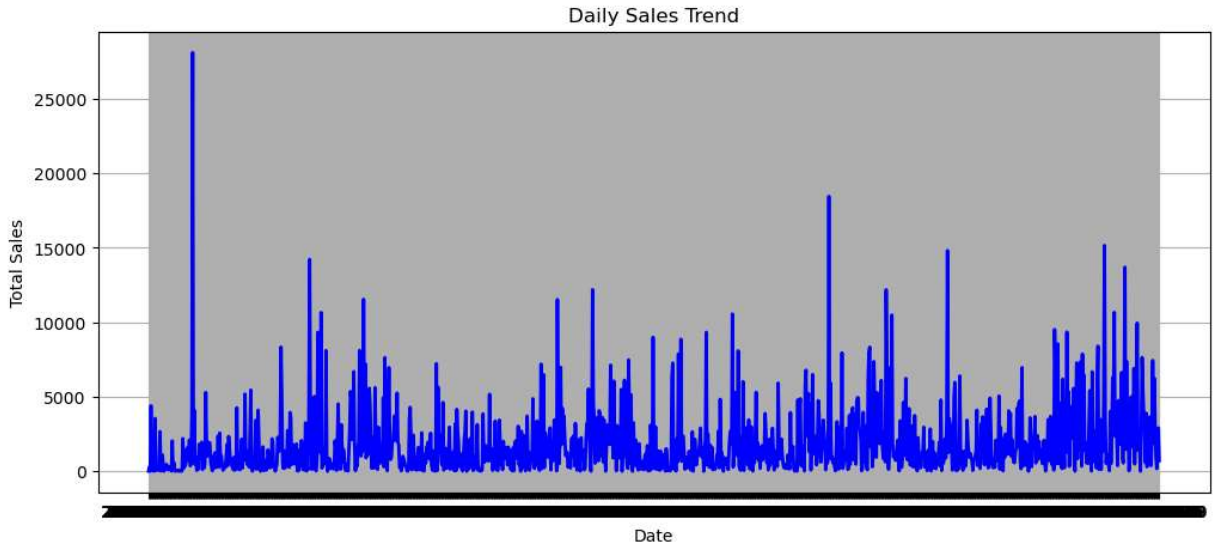
In [8]:
```
# Group by order date to get daily sales
daily_sales = df.groupby('Order_Date')['Sales'].sum()

# Display the first few rows
daily_sales.head()
```

Out[8]:
```
Order_Date
2014-01-03      16.448
2014-01-04     288.060
2014-01-05      19.536
2014-01-06    4407.100
2014-01-07      87.158
Name: Sales, dtype: float64
```

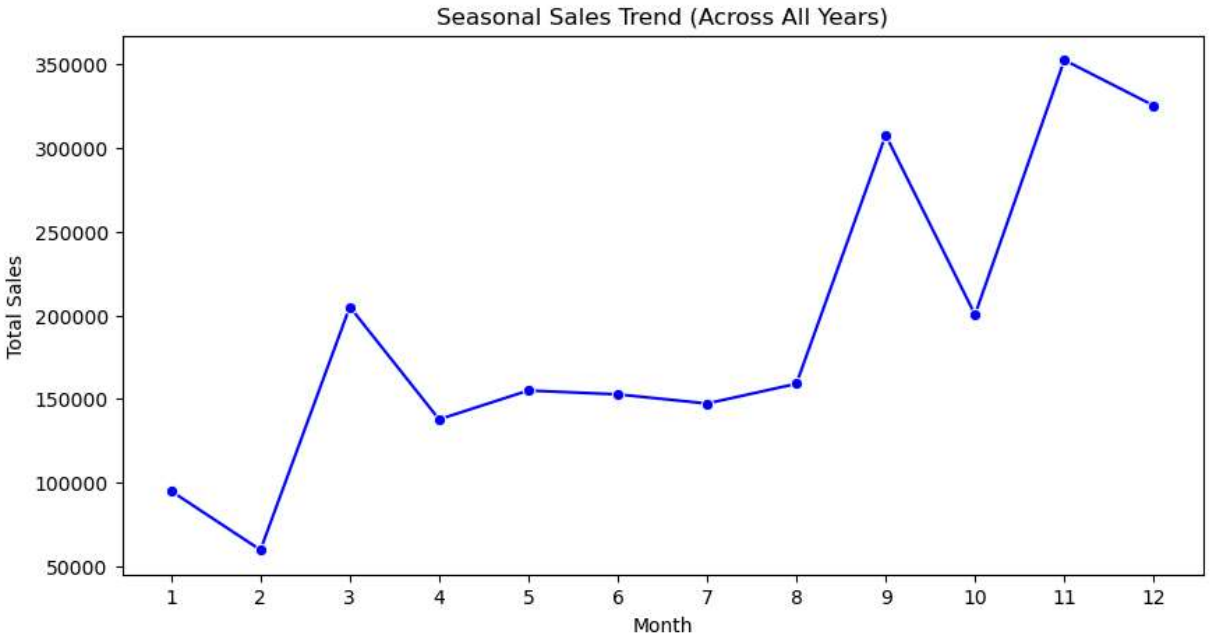
In [9]:
```
# Find the date with the highest sales
highest_sales_date = daily_sales.idxmax()
highest_sales_value = daily_sales.max()

print(f"📌 The date with the highest sales is {highest_sales_date} with a total sa
```

📌 The date with the highest sales is 2014-03-18 with a total sales of $28106.72

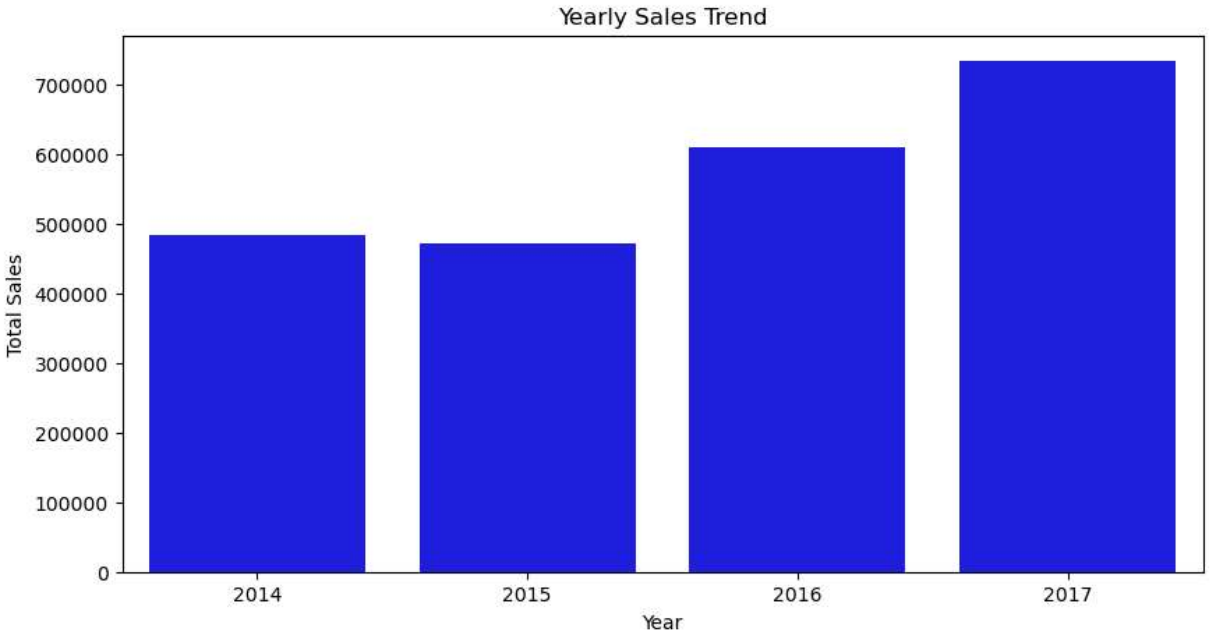
In [10]:
```
%matplotlib inline

plt.figure(figsize=(12, 5))
plt.plot(daily_sales.index, daily_sales.values, color='blue', linewidth=2)
plt.xlabel("Date")
plt.ylabel("Total Sales")
plt.title("Daily Sales Trend")
plt.grid(True)
plt.show()
```



In [11]:
```
df['Order_Date'] = pd.to_datetime(df['Order_Date'], errors='coerce')


# Extract month and year
df['Year'] = df['Order_Date'].dt.year
df['Month'] = df['Order_Date'].dt.month

# Group by month to see seasonality
monthly_sales = df.groupby('Month')['Sales'].sum()

# Plot Monthly Trends
plt.figure(figsize=(10, 5))
sns.lineplot(x=monthly_sales.index, y=monthly_sales.values, marker='o', color='blue
plt.xticks(range(1, 13))
plt.xlabel("Month")
plt.ylabel("Total Sales")
plt.title("Seasonal Sales Trend (Across All Years)")
plt.show()
```

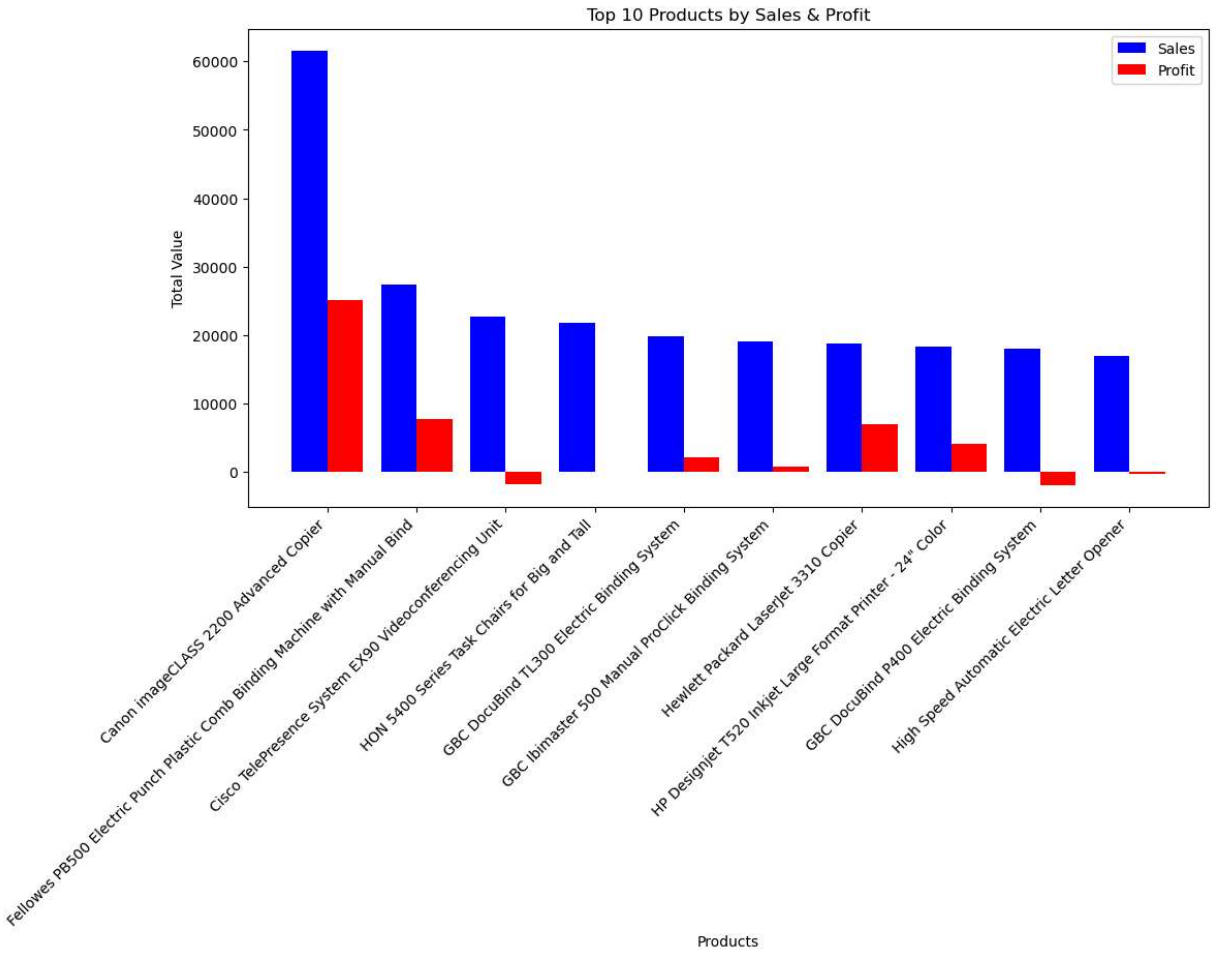


In [12]:
```
yearly_sales = df.groupby('Year')['Sales'].sum()

# Plot Yearly Trends
plt.figure(figsize=(10, 5))
sns.barplot(x=yearly_sales.index, y=yearly_sales.values, color='blue')
plt.xlabel("Year")
plt.ylabel("Total Sales")
plt.title("Yearly Sales Trend")
plt.show()
```



In [13]:
```
# Group by Product and calculate total Sales and Profit
product_performance = df.groupby('Product_Name')[['Sales', 'Profit']].sum().reset_i

# Sort by Sales for better visualization (Top 10 Products)
top_products = product_performance.sort_values(by='Sales', ascending=False).head(10
```

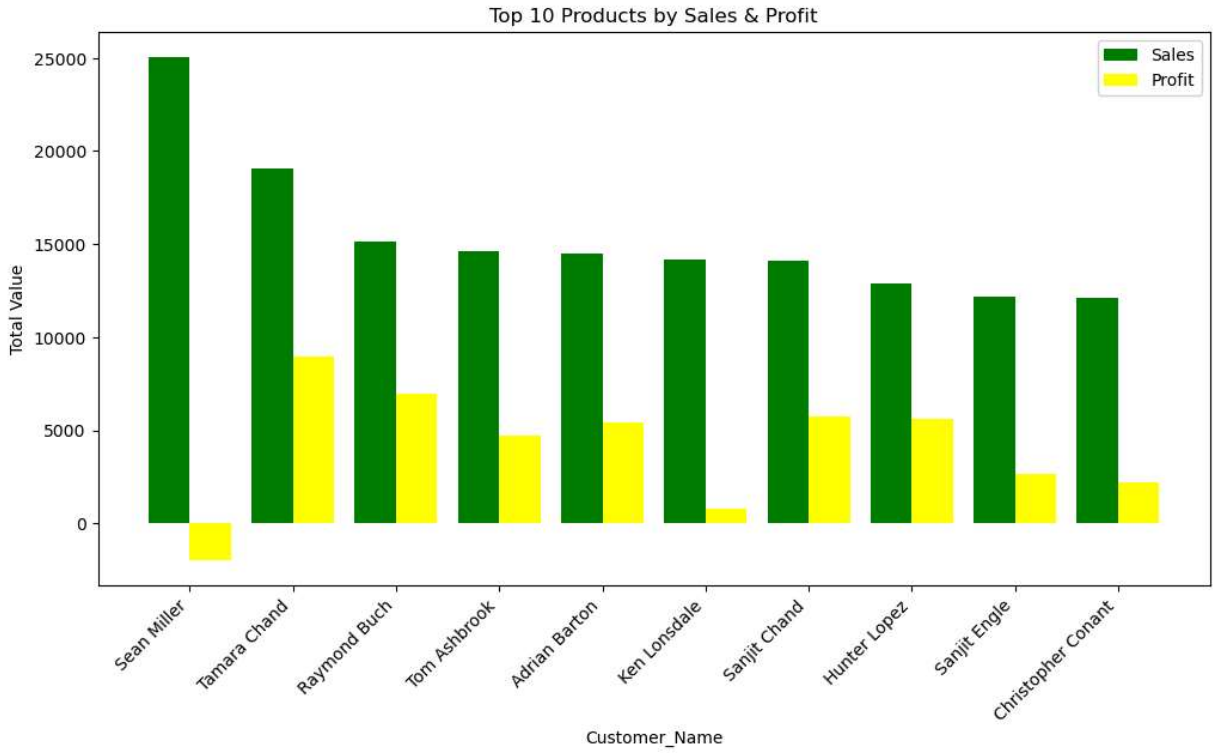
```
# Set width of bars
bar_width = 0.4
x_indexes = np.arange(len(top_products))  # Create indexes for bars

# Plot the bars
fig, ax = plt.subplots(figsize=(12, 6))
ax.bar(x_indexes, top_products["Sales"], width=bar_width, label="Sales", color="blu
ax.bar(x_indexes + bar_width, top_products["Profit"], width=bar_width, label="Profi

# Formatting the chart
ax.set_xlabel("Products")
ax.set_ylabel("Total Value")
ax.set_title("Top 10 Products by Sales & Profit")
ax.set_xticks(x_indexes + bar_width / 2)  # Position labels in the center
ax.set_xticklabels(top_products["Product_Name"], rotation=45, ha="right")
ax.legend()

# Show plot
plt.show()
```



In [14]:

```
# Group by Product and calculate total Sales and Profit
product_performance = df.groupby('Customer_Name')[['Sales', 'Profit']].sum().reset_

# Sort by Sales for better visualization (Top 10 Products)
top_products = product_performance.sort_values(by='Sales', ascending=False).head(10

# Set width of bars
```

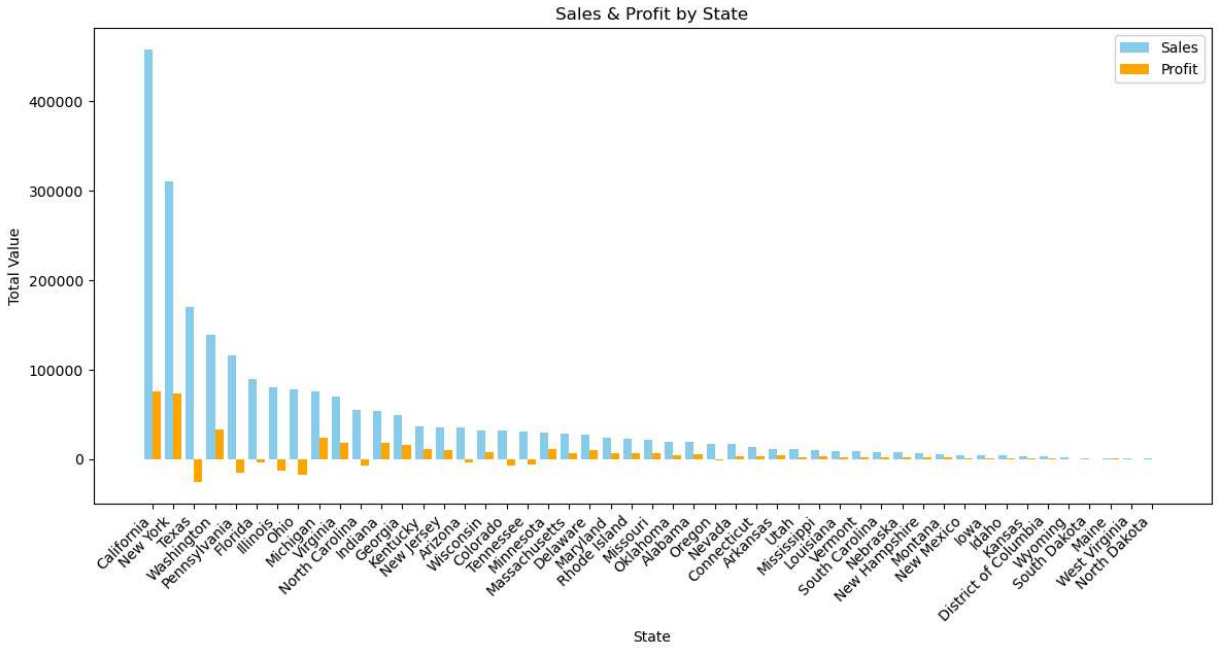
```
bar_width = 0.4
x_indexes = np.arange(len(top_products))  # Create indexes for bars

# Plot the bars
fig, ax = plt.subplots(figsize=(12, 6))
ax.bar(x_indexes, top_products["Sales"], width=bar_width, label="Sales", color="gre
ax.bar(x_indexes + bar_width, top_products["Profit"], width=bar_width, label="Profi

# Formatting the chart
ax.set_xlabel("Customer_Name")
ax.set_ylabel("Total Value")
ax.set_title("Top 10 Products by Sales & Profit")
ax.set_xticks(x_indexes + bar_width / 2)  # Position labels in the center
ax.set_xticklabels(top_products["Customer_Name"], rotation=45, ha="right")
ax.legend()

# Show plot
plt.show()
```



In [15]:

```
# Group by State and calculate total Sales and Profit
state_performance = df.groupby("State")[["Sales", "Profit"]].sum().reset_index()

# Sort by Sales for better visualization
state_performance = state_performance.sort_values(by="Sales", ascending=False)

# Set bar width and create index positions for bars
bar_width = 0.4
x_indexes = np.arange(len(state_performance))

# Create figure
fig, ax = plt.subplots(figsize=(14, 6))

# Plot Sales & Profit side by side
```

```
ax.bar(x_indexes, state_performance["Sales"], width=bar_width, label="Sales", color
ax.bar(x_indexes + bar_width, state_performance["Profit"], width=bar_width, label="

# Formatting the chart
ax.set_xlabel("State")
ax.set_ylabel("Total Value")
ax.set_title("Sales & Profit by State")
ax.set_xticks(x_indexes + bar_width / 2)  # Center labels
ax.set_xticklabels(state_performance["State"], rotation=45, ha="right")  # Rotate f
ax.legend()

# Show plot
plt.show()
```



In [16]:
```
pip install prophet
```

```
Requirement already satisfied: prophet in c:\users\hp\anaconda3\lib\site-packages
(1.1.6)
Requirement already satisfied: cmdstanpy>=1.0.4 in c:\users\hp\anaconda3\lib\site-pa
ckages (from prophet) (1.2.5)
Requirement already satisfied: numpy>=1.15.4 in c:\users\hp\anaconda3\lib\site-packa
ges (from prophet) (1.26.4)
Requirement already satisfied: matplotlib>=2.0.0 in c:\users\hp\anaconda3\lib\site-p
ackages (from prophet) (3.9.2)
Requirement already satisfied: pandas>=1.0.4 in c:\users\hp\anaconda3\lib\site-packa
ges (from prophet) (2.2.2)
Requirement already satisfied: holidays<1,>=0.25 in c:\users\hp\anaconda3\lib\site-p
ackages (from prophet) (0.69)
Requirement already satisfied: tqdm>=4.36.1 in c:\users\hp\anaconda3\lib\site-packag
es (from prophet) (4.66.5)
Requirement already satisfied: importlib-resources in c:\users\hp\anaconda3\lib\site
-packages (from prophet) (6.5.2)
Requirement already satisfied: stanio<2.0.0,>=0.4.0 in c:\users\hp\anaconda3\lib\sit
e-packages (from cmdstanpy>=1.0.4->prophet) (0.5.1)
Requirement already satisfied: python-dateutil in c:\users\hp\anaconda3\lib\site-pac
kages (from holidays<1,>=0.25->prophet) (2.9.0.post0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\hp\anaconda3\lib\site-pa
ckages (from matplotlib>=2.0.0->prophet) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\hp\anaconda3\lib\site-packag
es (from matplotlib>=2.0.0->prophet) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\hp\anaconda3\lib\site-p
ackages (from matplotlib>=2.0.0->prophet) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\hp\anaconda3\lib\site-p
ackages (from matplotlib>=2.0.0->prophet) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\hp\anaconda3\lib\site-pac
kages (from matplotlib>=2.0.0->prophet) (24.1)
Requirement already satisfied: pillow>=8 in c:\users\hp\anaconda3\lib\site-packages
(from matplotlib>=2.0.0->prophet) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\hp\anaconda3\lib\site-pa
ckages (from matplotlib>=2.0.0->prophet) (3.1.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\hp\anaconda3\lib\site-packag
es (from pandas>=1.0.4->prophet) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\hp\anaconda3\lib\site-pack
ages (from pandas>=1.0.4->prophet) (2023.3)
Requirement already satisfied: colorama in c:\users\hp\anaconda3\lib\site-packages
(from tqdm>=4.36.1->prophet) (0.4.6)
Requirement already satisfied: six>=1.5 in c:\users\hp\anaconda3\lib\site-packages
(from python-dateutil->holidays<1,>=0.25->prophet) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [17]:
```
from prophet import Prophet
import pandas as pd
import matplotlib.pyplot as plt
```

In [18]:
```
print(df.columns)
```

```
Index(['Row_ID', 'Order_ID', 'Order_Date', 'Ship_Date', 'Ship_Mode',
       'Customer_ID', 'Customer_Name', 'Segment', 'Country', 'City', 'State',
       'Postal_Code', 'Region', 'Product_ID', 'Category', 'Sub_Category',
       'Product_Name', 'Sales', 'Quantity', 'Discount', 'Profit', 'Year',
       'Month'],
      dtype='object')
```

In [19]:
```
# Rename columns for Prophet
sales_data = df[['Order_Date', 'Sales']].copy()
sales_data.columns = ['ds', 'y']

# Ensure 'ds' is a datetime format
sales_data['ds'] = pd.to_datetime(sales_data['ds'])

# Check the dataset
print(sales_data.head())
```

```
          ds         y
0 2016-11-08  261.9600
1 2016-11-08  731.9400
2 2016-06-12   14.6200
3 2015-10-11  957.5775
4 2015-10-11   22.3680
```

In [20]:
```
# Initialize Prophet model
model = Prophet()

# Fit the model
model.fit(sales_data)
```

```
07:32:24 - cmdstanpy - INFO - Chain [1] start processing
07:32:27 - cmdstanpy - INFO - Chain [1] done processing
```

Out[20]: <prophet.forecaster.Prophet at 0x1de1e331cd0>

In [21]:
```
# yhat: Predicted sales

# yhat_lower: Lower bound of prediction

# yhat_upper: Upper bound of prediction

# Create a future dataframe for prediction (next 12 months)
future = model.make_future_dataframe(periods=365)  # Forecast for 1 year
forecast = model.predict(future)

# Check the forecast output
print(forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail())
```

```
            ds        yhat  yhat_lower   yhat_upper
1597 2018-12-26  222.747540 -591.739029  1018.968314
1598 2018-12-27  207.893068 -617.009155  1019.548463
1599 2018-12-28  218.951181 -620.334157  1042.914213
1600 2018-12-29  199.237047 -607.219038   962.735119
1601 2018-12-30  205.507106 -581.660858   985.580612
```

In [22]:
```
print(forecast.tail())
print(sales_data.tail())
```

```
            ds       trend  yhat_lower   yhat_upper  trend_lower  \
1597 2018-12-26  200.886215 -591.739029  1018.968314   192.387474
1598 2018-12-27  200.842213 -617.009155  1019.548463   192.312407
1599 2018-12-28  200.798211 -620.334157  1042.914213   192.241143
1600 2018-12-29  200.754208 -607.219038   962.735119   192.167737
1601 2018-12-30  200.710206 -581.660858   985.580612   192.097189

      trend_upper  additive_terms  additive_terms_lower  additive_terms_upper  \
1597   209.722029       21.861325             21.861325             21.861325
1598   209.762256        7.050855              7.050855              7.050855
1599   209.813533       18.152971             18.152971             18.152971
1600   209.834255       -1.517162             -1.517162             -1.517162
1601   209.829014        4.796900              4.796900              4.796900

         weekly  weekly_lower  weekly_upper     yearly  yearly_lower  \
1597   3.411013      3.411013      3.411013  18.450312     18.450312
1598  -9.784300     -9.784300     -9.784300  16.835155     16.835155
1599   3.235217      3.235217      3.235217  14.917753     14.917753
1600 -14.265179    -14.265179    -14.265179  12.748017     12.748017
1601  -5.588185     -5.588185     -5.588185  10.385084     10.385084

      yearly_upper  multiplicative_terms  multiplicative_terms_lower  \
1597     18.450312                   0.0                         0.0
1598     16.835155                   0.0                         0.0
1599     14.917753                   0.0                         0.0
1600     12.748017                   0.0                         0.0
1601     10.385084                   0.0                         0.0

      multiplicative_terms_upper        yhat
1597                         0.0  222.747540
1598                         0.0  207.893068
1599                         0.0  218.951181
1600                         0.0  199.237047
1601                         0.0  205.507106
            ds        y
9989 2014-01-21   25.248
9990 2017-02-26   91.960
9991 2017-02-26  258.576
9992 2017-02-26   29.600
9993 2017-05-04  243.160
```

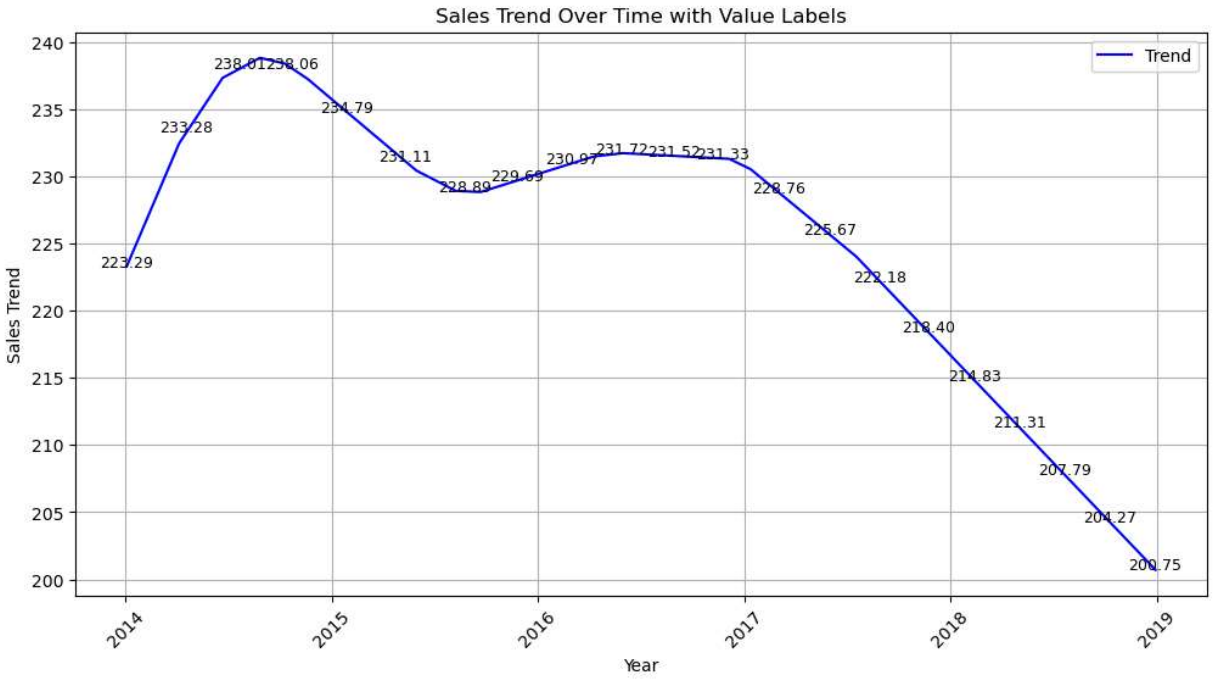
In [23]:
```
sales_data['ds'] = pd.to_datetime(sales_data['ds'])
forecast['ds'] = pd.to_datetime(forecast['ds'])
```

In [24]:
```
print("\nData Types:")
print(forecast.dtypes)
print(sales_data.dtypes)
```

```
Data Types:
ds                             datetime64[ns]
trend                                 float64
yhat_lower                            float64
yhat_upper                            float64
trend_lower                           float64
trend_upper                           float64
additive_terms                        float64
additive_terms_lower                  float64
additive_terms_upper                  float64
weekly                                float64
weekly_lower                          float64
weekly_upper                          float64
yearly                                float64
yearly_lower                          float64
yearly_upper                          float64
multiplicative_terms                  float64
multiplicative_terms_lower            float64
multiplicative_terms_upper            float64
yhat                                  float64
dtype: object
ds    datetime64[ns]
y            float64
dtype: object
```

In [73]:

```python
import matplotlib.pyplot as plt

# Extract only the trend component
trend_data = forecast[['ds', 'trend']]

# Create figure
fig, ax = plt.subplots(figsize=(12, 6))

# Plot trend line
ax.plot(trend_data['ds'], trend_data['trend'], label="Trend", color='blue', linesty

# Add value labels at well-spaced intervals (e.g., every 50th point)
step = max(1, len(trend_data) // 20)  # Adjust this to control label spacing
for i in range(0, len(trend_data), step):
    ax.text(trend_data['ds'].iloc[i], trend_data['trend'].iloc[i],
            f"{trend_data['trend'].iloc[i]:.2f}", fontsize=9, color='black', ha='ce

# Formatting
ax.set_xlabel("Year")
ax.set_ylabel("Sales Trend")
ax.set_title("Sales Trend Over Time with Value Labels")
ax.legend()
plt.xticks(rotation=45)
plt.grid(True)

plt.show()
```


Sales Trend Over Time with Value Labels
Trend
Sales Trend
Year
240
235
230
225
220
215
210
205
200
2014
2015
2016
2017
2018
2019
223.29
233.28
238.01
238.06
234.79
231.11
228.89
229.69
230.97
231.72
231.52
231.33
228.76
225.67
222.18
218.40
214.83
211.31
207.79
204.27
200.75