Adidas Sales Analysis

In [21]:
```
#import the libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
```

In [2]:
```
#import the dataset
df=pd.read_csv(r'C:\\Users\user\Downloads\Adidas_sales.csv')
df.head()
```

Out[2]:

	Retailer	Retailer ID	Invoice Date	Region	State	City	Gender Type	Product Category	Price per Unit	U S
0	Foot Locker	1185732	Tuesday, October 26, 2021	Northeast	Pennsylvania	Philadelphia	Men	Apparel	55	
1	Foot Locker	1185732	Wednesday, October 27, 2021	Northeast	Pennsylvania	Philadelphia	Women	Apparel	45	
2	Foot Locker	1185732	Thursday, October 28, 2021	Northeast	Pennsylvania	Philadelphia	Men	Street Footwear	45	
3	Foot Locker	1185732	Friday, October 29, 2021	Northeast	Pennsylvania	Philadelphia	Men	Athletic Footwear	45	
4	Foot Locker	1185732	Saturday, October 30, 2021	Northeast	Pennsylvania	Philadelphia	Women	Street Footwear	35	

In [3]:
```
#check for null values
df.isnull().sum()
```

Out[3]:
```
Retailer            0
Retailer ID         0
Invoice Date        0
Region              0
State               0
City                0
Gender Type         0
Product Category    0
Price per Unit      0
Units Sold          0
Operating Profit    0
Operating Margin    0
Sales Method        0
dtype: int64
```

In [4]:
```
#check for duplicates
df.duplicated().sum()
```

Out[4]:
```
0
```

In [5]:
```
#changing to proper date time format
df['Invoice Date'] = pd.to_datetime(df['Invoice Date'])

#extracting year and month from date
df['Year'] = df['Invoice Date'].dt.year
df['Month'] = df['Invoice Date'].dt.strftime('%b')

#calculating total sales
df['Total_Sales'] = df['Price per Unit'] * df['Units Sold']
```

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

Out[6]:

	Retailer	Retailer ID	Invoice Date	Region	State	City	Gender Type	Product Category	Price per Unit	Units Sold
0	Foot Locker	1185732	2021-10-26	Northeast	Pennsylvania	Philadelphia	Men	Apparel	55	125
1	Foot Locker	1185732	2021-10-27	Northeast	Pennsylvania	Philadelphia	Women	Apparel	45	225
2	Foot Locker	1185732	2021-10-28	Northeast	Pennsylvania	Philadelphia	Men	Street Footwear	45	475
3	Foot Locker	1185732	2021-10-29	Northeast	Pennsylvania	Philadelphia	Men	Athletic Footwear	45	125
4	Foot Locker	1185732	2021-10-30	Northeast	Pennsylvania	Philadelphia	Women	Street Footwear	35	175

top 5 retailers based on sales

In [7]:
```
top_retailers_by_sales = df.groupby('Retailer')['Total_Sales'].sum().nlargest(5
formatted_sales = top_retailers_by_sales.map("${:,.2f}".format)
print(formatted_sales)
```

```
Retailer
West Gear        $32,409,558.00
Foot Locker      $29,024,945.00
Sports Direct    $24,616,622.00
Kohl's           $13,512,453.00
Walmart          $10,506,085.00
Name: Total_Sales, dtype: object
```

popular product category

In [8]:
```
popular_product_categories = df['Product Category'].value_counts()
print(popular_product_categories)
```

```
Street Footwear      3218
Athletic Footwear    3216
Apparel              3214
Name: Product Category, dtype: int64
```

sales trend over year

In [9]:
```
sales_trend_over_Year = df.groupby('Year')['Total_Sales'].sum()
print(sales_trend_over_Year)
```

```
Year
2020    24237325
2021    95929325
Name: Total_Sales, dtype: int64
```

profit based on product category per region

In [10]:
```
profit_per_product_per_region = df.groupby(['Region', 'Product Category'])['Ope
print(profit_per_product_per_region)
```

```
Region     Product Category
Midwest    Apparel              19557607.49
           Athletic Footwear    12450688.63
           Street Footwear      20803050.36
Northeast  Apparel              21660763.12
           Athletic Footwear    17046800.12
           Street Footwear      29313024.41
South      Apparel              21901081.56
           Athletic Footwear    18472984.56
           Street Footwear      20763937.95
Southeast  Apparel              22163629.15
           Athletic Footwear    16438789.97
           Street Footwear      21952997.58
West       Apparel              28130919.57
           Athletic Footwear    26413409.85
           Street Footwear      35065077.13
Name: Operating Profit, dtype: float64
```

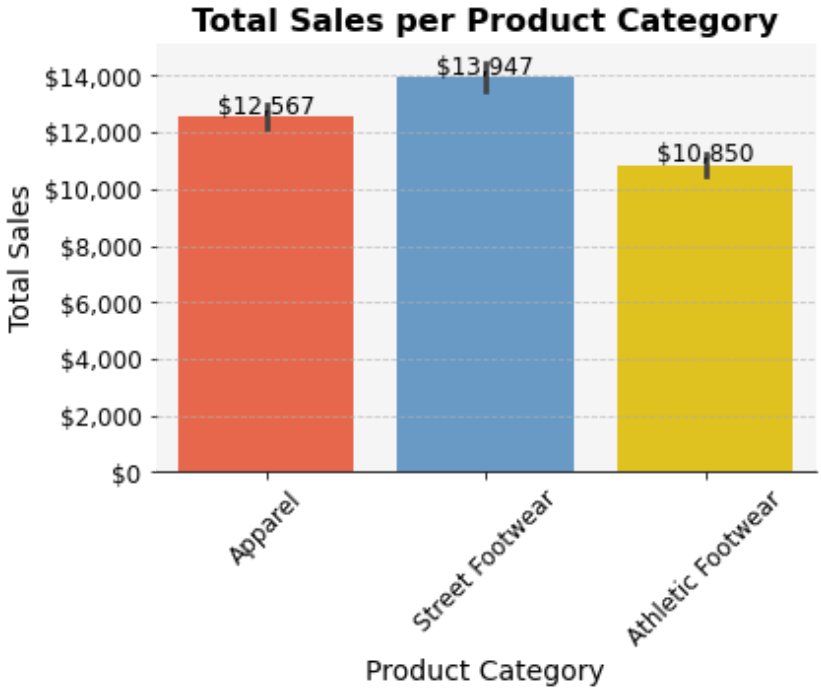
Which product category has the highest total sales and operating profit?

In [18]:
```
# Group the data by 'Product' and calculate total sales and operating profit
product_summary = df.groupby('Product Category')[['Total_Sales', 'Operating Pro

# Find the product category with the highest total sales and operating profit
max_sales_product = product_summary['Total_Sales'].idxmax()
max_profit_product = product_summary['Operating Profit'].idxmax()

print("Product category with the highest total sales:", max_sales_product)
print("Product category with the highest operating profit:", max_profit_product
```

```
Product category with the highest total sales: Street Footwear
Product category with the highest operating profit: Street Footwear
```

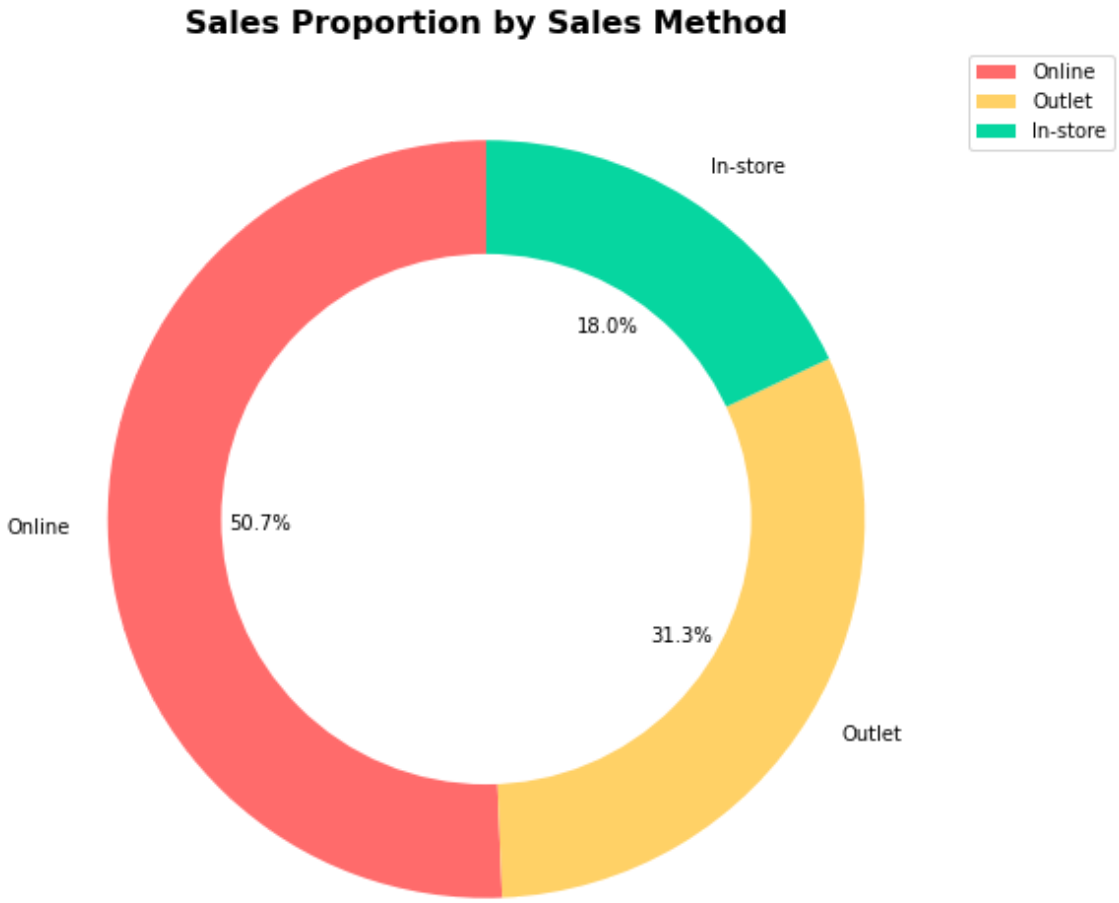
Total Sales per Product Category

In [11]:
```
color=['#FF5733', '#5A9BD4', '#FFD700']

# Create the barplot
sns.barplot(x='Product Category', y='Total_Sales', data=df, palette=color)

# Add a title
plt.title('Total Sales per Product Category', fontsize=16, fontweight='bold')

# Label the axes and customize their fonts
plt.xlabel('Product Category', fontsize=14)
plt.ylabel('Total Sales', fontsize=14)

# Customize y-axis labels with commas and specify their font size
from matplotlib.ticker import FuncFormatter
def format_func(value, tick_number):
    return f"${value:,.0f}"
plt.gca().yaxis.set_major_formatter(FuncFormatter(format_func))
plt.yticks(fontsize=12)

# Rotate x-axis labels for better readability
plt.xticks(rotation=45, fontsize=12)

# Customize the plot background color
plt.gca().set_facecolor('#F5F5F5')

# Customize the grid lines
sns.despine(left=True)
plt.grid(axis='y', linestyle='--', alpha=0.7)


# Add data labels on top of the bars
for p in plt.gca().patches:
    plt.gca().annotate(f"${p.get_height():,.0f}",
                       (p.get_x() + p.get_width() / 2., p.get_height()),
                        ha='center', va='center', fontsize=12,
                       color='black', xytext=(0, 5),
                        textcoords='offset points')

plt.show()
```


Total Sales per Product Category
$14,000
$12,000
$10,000
$8,000
$6,000
$4,000
$2,000
$0
Total Sales
$12,567
$13,947
$10,850
Apparel
Street Footwear
Athletic Footwear
Product Category

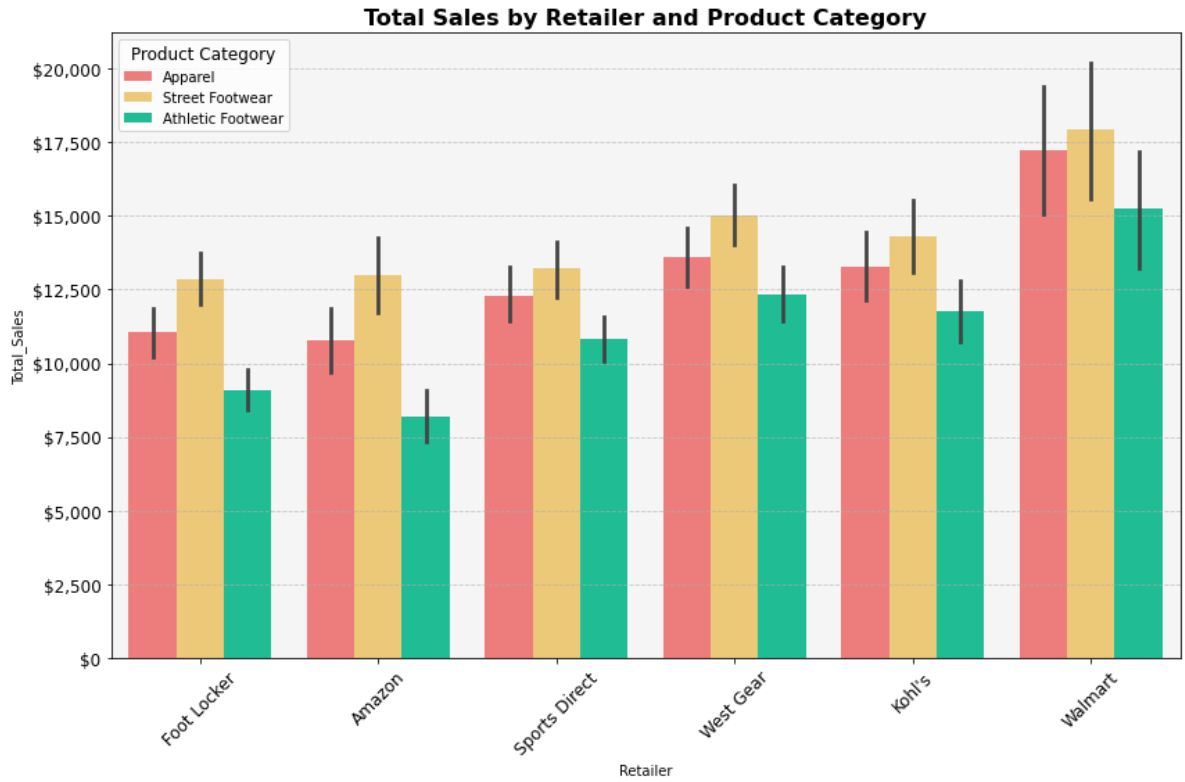
sales proportion by sales method

In [14]:
```
# Define a custom color palette with beautiful colors
custom_colors = ['#FF6B6B', '#FFD166', '#06D6A0']

# Create a figure and set its size
plt.figure(figsize=(8, 8))

# Create the pie chart with custom colors and labels
sales_method_counts = df['Sales Method'].value_counts()
plt.pie(sales_method_counts, labels=sales_method_counts.index,
        autopct='%1.1f%%', colors=custom_colors, startangle=90)

# Add a title
plt.title('Sales Proportion by Sales Method', fontsize=16, fontweight='bold')

# Add a legend with custom colors
plt.legend(sales_method_counts.index, loc='upper left', bbox_to_anchor=(1.0, 1.

# Add a shadow effect to the pie chart
plt.gca().set_aspect('equal')  # Equal aspect ratio ensures that pie is drawn a
plt.gca().add_artist(plt.Circle((0, 0), 0.7, fc='white'))

# Show the plot
plt.tight_layout()
plt.show()
```



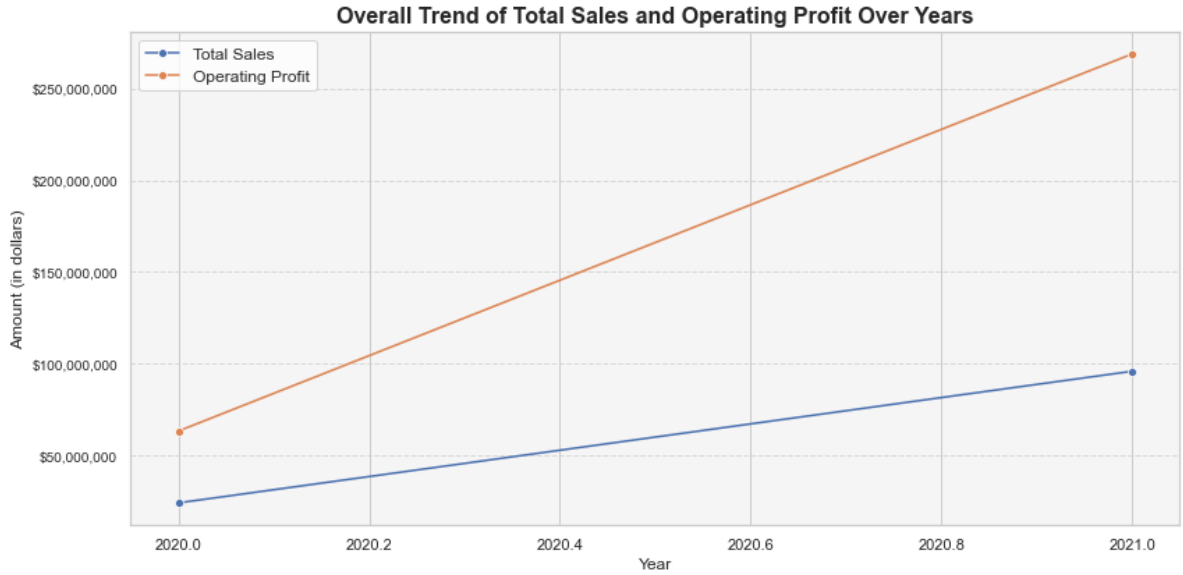
Total sales by retailer and product category

In [15]:
```
# Define a custom color palette with beautiful colors
custom_palette = ['#FF6B6B', '#FFD166', '#06D6A0']

# Create a figure and set its size
plt.figure(figsize=(12, 8))

# Create the grouped bar chart with custom colors and labels
sns.barplot(x='Retailer', y='Total_Sales', hue='Product Category', data=df, pal

# Add a title
plt.title('Total Sales by Retailer and Product Category', fontsize=16, fontweig

# Customize x-axis labels rotation for better readability
plt.xticks(rotation=45, fontsize=12)

# Customize y-axis labels with commas and specify their font size
from matplotlib.ticker import FuncFormatter
def format_func(value, tick_number):
    return f"${value:,.0f}"
plt.gca().yaxis.set_major_formatter(FuncFormatter(format_func))
plt.yticks(fontsize=12)

# Customize the legend
plt.legend(title='Product Category', title_fontsize=12, fontsize=10)

# Add grid lines for better readability
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Customize the background color of the plot
plt.gca().set_facecolor('#F5F5F5')

# Show the plot
plt.tight_layout()
plt.show()
```


Total Sales by Retailer and Product Category
Product Category
Apparel
Street Footwear
Athletic Footwear
$20,000
$17,500
$15,000
$12,500
$10,000
$7,500
$5,000
$2,500
$0
Total_Sales
Foot Locker
Amazon
Sports Direct
West Gear
Kohl's
Walmart
Retailer

What is the overall trend of total sales and operating profit over the years?

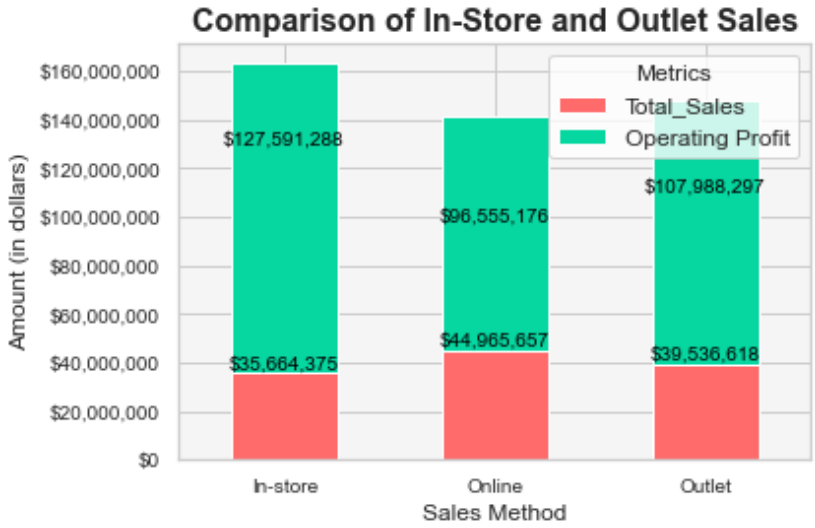
In [16]:
```
# Group by year and calculate total sales and operating profit
yearly_summary = df.groupby('Year')[['Total_Sales', 'Operating Profit']].sum().

# Set Seaborn style
sns.set(style="whitegrid")

# Create a figure and set its size
plt.figure(figsize=(12, 6))

# Plotting the trend using Seaborn
sns.lineplot(data=yearly_summary, x='Year', y='Total_Sales', marker='o', label=
sns.lineplot(data=yearly_summary, x='Year', y='Operating Profit', marker='o', l

# Customize the plot title and labels
plt.title('Overall Trend of Total Sales and Operating Profit Over Years', fonts
plt.xlabel('Year', fontsize=12)
plt.ylabel('Amount (in dollars)', fontsize=12)

# Customize y-axis labels with commas and specify their font size
def format_func(value, tick_number):
    return f"${value:,.0f}"
plt.gca().yaxis.set_major_formatter(FuncFormatter(format_func))
plt.yticks(fontsize=10)

# Add a legend with adjusted font size
plt.legend(fontsize=12)

# Add grid lines for better readability
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Customize the background color of the plot
plt.gca().set_facecolor('#F5F5F5')

# Show the plot
plt.tight_layout()
plt.show()
```



How do in-store and outlet sales compare in terms of total sales and operating profit?

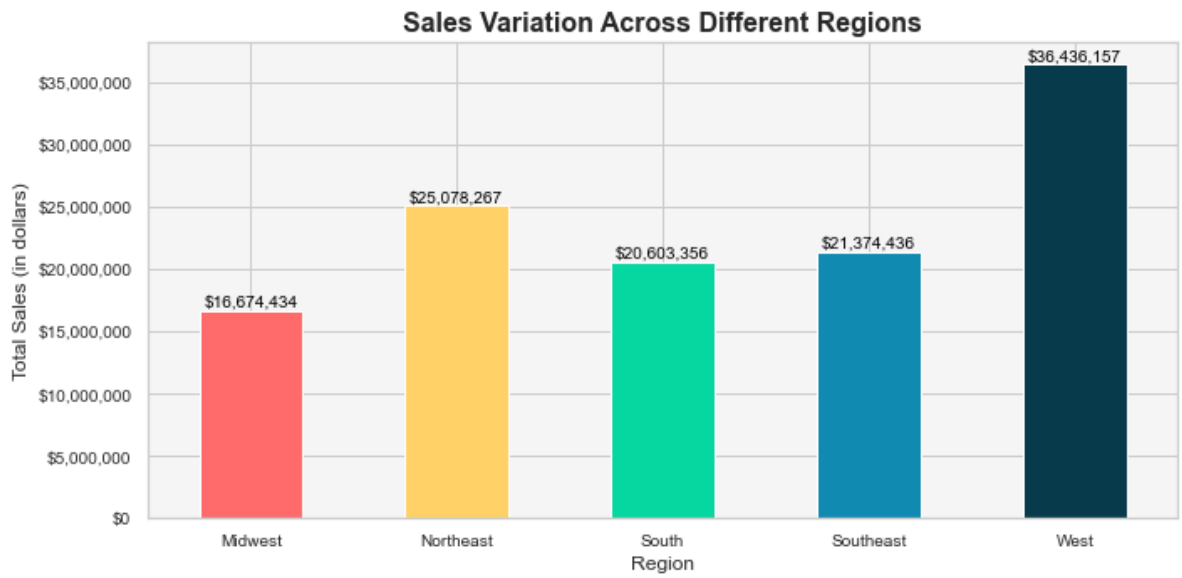
In [17]:
```
# Group the data by 'Sales Method' and calculate total sales and operating prof
sales_comparison = df.groupby('Sales Method')[['Total_Sales', 'Operating Profit

# Create a figure and set its size
plt.figure(figsize=(12, 8))

# Define custom colors for the bars
colors = ['#FF6B6B', '#06D6A0']

# Create the bar chart with stacked bars and custom colors
ax = sales_comparison.plot(kind='bar', stacked=True, color=colors)

# Customize the plot title and labels
plt.title('Comparison of In-Store and Outlet Sales', fontsize=16, fontweight='b
plt.xlabel('Sales Method', fontsize=12)
plt.ylabel('Amount (in dollars)', fontsize=12)

# Customize x-axis labels rotation for better readability
plt.xticks(rotation=0, fontsize=10)

# Customize y-axis labels with commas and specify their font size
from matplotlib.ticker import FuncFormatter
def format_func(value, tick_number):
    return f"${value:,.0f}"
ax.yaxis.set_major_formatter(FuncFormatter(format_func))
plt.yticks(fontsize=10)

# Add data labels on top of the bars
for p in ax.patches:
    ax.annotate(f"${p.get_height():,.0f}",
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10,
                color='black', xytext=(0, 5),
                textcoords='offset points')

# Customize the legend with adjusted font size and title
plt.legend(title='Metrics', fontsize=12)

# Customize the background color of the plot
ax.set_facecolor('#F5F5F5')

# Show the plot
plt.tight_layout()
plt.show()
```

```
<Figure size 864x576 with 0 Axes>
```


Comparison of In-Store and Outlet Sales
Metrics
Total_Sales
Operating Profit
$160,000,000
$140,000,000
$120,000,000
$100,000,000
$80,000,000
$60,000,000
$40,000,000
$20,000,000
$0
Amount (in dollars)
$127,591,288
$35,664,375
$96,555,176
$44,965,657
$107,988,297
$39,536,618
In-store
Online
Outlet
Sales Method

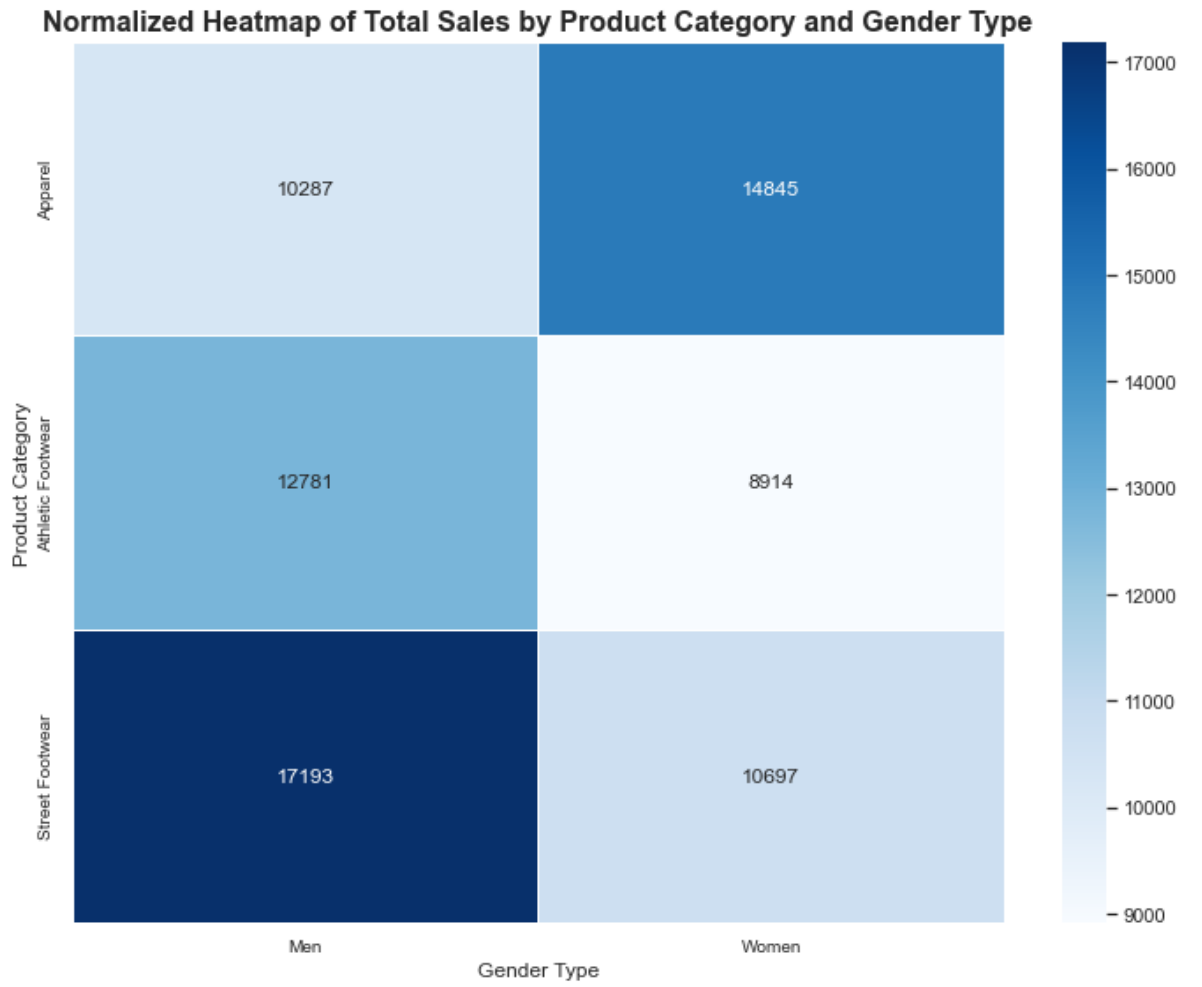
How do sales vary across different states and cities?

In [19]:
```
# Create a figure and set its size
plt.figure(figsize=(10, 5))

# Group the data by 'Region' and calculate total sales for each region
region_sales = df.groupby('Region')['Total_Sales'].sum()

# Define a custom color palette with visually appealing colors
custom_palette = ['#FF6B6B', '#FFD166', '#06D6A0', '#118AB2', '#073B4C']

# Create the bar chart with custom colors
ax = region_sales.plot(kind='bar', color=custom_palette)

# Customize the plot title and labels
plt.title('Sales Variation Across Different Regions', fontsize=16, fontweight='
plt.xlabel('Region', fontsize=12)
plt.ylabel('Total Sales (in dollars)', fontsize=12)

# Customize x-axis labels rotation for better readability
plt.xticks(rotation=0, fontsize=10)

# Customize y-axis labels with commas and specify their font size
from matplotlib.ticker import FuncFormatter
def format_func(value, tick_number):
    return f"${value:,.0f}"
ax.yaxis.set_major_formatter(FuncFormatter(format_func))
plt.yticks(fontsize=10)

# Add data labels on top of the bars
for p in ax.patches:
    ax.annotate(f"${p.get_height():,.0f}", (p.get_x() + p.get_width() / 2., p.g
                ha='center', va='center', fontsize=10, color='black', xytext=(0
                textcoords='offset points')

# Customize the background color of the plot
ax.set_facecolor('#F5F5F5')

# Show the plot
plt.tight_layout()
plt.show()
```



Total sales for each combination of product category and gender type

In [20]:
```
plt.figure(figsize=(10, 8))

heatmap_data = df.pivot_table(index='Product Category', columns='Gender Type',
# Use a different color palette (Blues in this case)
sns.heatmap(heatmap_data, cmap='Blues', annot=True, fmt=".0f", linewidths=0.5)

# Add axis labels
plt.xlabel('Gender Type', fontsize=12)
plt.ylabel('Product Category', fontsize=12)

# Increase font size of annotations
plt.xticks(fontsize=10)
plt.yticks(fontsize=10)

# Modify title
plt.title('Normalized Heatmap of Total Sales by Product Category and Gender Typ

# Adjust layout
plt.tight_layout()

# Show the plot
plt.show()
```



Statistical Analysis

In [22]:
```
summary_statistics = df.describe()
print(summary_statistics)
```

```
        Retailer ID  Price per Unit   Units Sold  Operating Profit  \
count  9.648000e+03     9648.000000  9648.000000       9648.000000
mean   1.173850e+06       45.216625   256.930037      34425.244761
std    2.636038e+04       14.705397   214.252030      54193.113713
min    1.128299e+06        7.000000     0.000000          0.000000
25%    1.185732e+06       35.000000   106.000000       1921.752500
50%    1.185732e+06       45.000000   176.000000       4371.420000
75%    1.185732e+06       55.000000   350.000000      52062.500000
max    1.197831e+06      110.000000  1275.000000     390000.000000

       Operating Margin         Year   Total_Sales
count       9648.000000  9648.000000   9648.000000
mean           0.422991  2020.865050  12455.083955
std            0.097197     0.341688  12716.392111
min            0.100000  2020.000000      0.000000
25%            0.350000  2021.000000   4065.250000
50%            0.410000  2021.000000   7803.500000
75%            0.490000  2021.000000  15864.500000
max            0.800000  2021.000000  82500.000000
```

T-statistics and P value

In [29]:
```
men_sales = df[df['Gender Type'] == 'Men']['Total_Sales']
women_sales = df[df['Gender Type'] == 'Women']['Total_Sales']
t_statistic, p_value = stats.ttest_ind(men_sales, women_sales)
print("T-statistic:", round(t_statistic, 2))
print("P-value:", p_value)
```

```
T-statistic: 7.5
P-value: 6.967577148164075e-14
```

F-statistics and ANOVA

In [27]:
```
regions = df['Region'].unique()
grouped_data = [df[df['Region'] == region]['Total_Sales'] for region in regions

f_statistic, p_value_anova = stats.f_oneway(*grouped_data)
print("F-statistic:", round(f_statistic, 2))
print("P-value (ANOVA):", p_value_anova)
```

```
F-statistic: 126.49
P-value (ANOVA): 2.0105836721943673e-105
```

Executive summary

Sales and Operating Profit Trends:

The overall sales trend for Adidas products is showing a consistent increase over time. However, a notable finding is that the operating profit has increased at a greater rate. This indicates an improvement in profitability and operational efficiency.

Regional Analysis

The West region stands out with the highest sales figures. This suggests a strong market demand for Adidas products in the West region.

For regions with lower sales like Midwest,there is an opportunity for strategic interventions to boost sales.This could involve targeted marketing campaigns, promotions, or assessing the product mix to better align with local preferences.

Product Category Insights

Footwear emerges as the leading product category, contributing significantly to total sales. The specific product categories, such as "Street Footwear" and "Athletic Footwear," demonstrate strong sales performance

Statistical analysis

T-test indicates a significant difference in a variable (e.g., sales or profit) between Men and Women in the dataset.

ANOVA reveals significant differences in a numerical variable (e.g., sales or profit) among various 'Product Category' groups.

Gender and 'Product Category' have a notable impact on key performance indicators (KPIs) such as sales and profitability in the Adidas sales dataset

In [ ]: