Product Performance Analysis

This notebook presents an analysis of product performance using a Sales dataset focusing on Bike products. The objective is to assess the performance of various products, identifying best-sellers and underperformers, while also understanding the underlying reasons. Finally, measures for improvement will be suggested based on these insights.

Data Import and Cleaning

Importing libraries

In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
data=pd.read_csv('E:/Sales_data.csv')
```

In [4]:
```
data.head(5)
```

Out[4]:

	index	Date	Year	Month	Customer Age	Customer Gender	Country	State	Product Category	Sub Category	Quantity	Unit Cost	Unit Price	Cost	Revenue	C
0	0	2/19/2016	2016.0	February	29.0	F	United States	Washington	Accessories	Tires and Tubes	1.0	80.00	109.000000	80.0	109.0	
1	1	2/20/2016	2016.0	February	29.0	F	United States	Washington	Clothing	Gloves	2.0	24.50	28.500000	49.0	57.0	
2	2	2/27/2016	2016.0	February	29.0	F	United States	Washington	Accessories	Tires and Tubes	3.0	3.67	5.000000	11.0	15.0	
3	3	3/12/2016	2016.0	March	29.0	F	United States	Washington	Accessories	Tires and Tubes	2.0	87.50	116.500000	175.0	233.0	
4	4	3/12/2016	2016.0	March	29.0	F	United States	Washington	Accessories	Tires and Tubes	3.0	35.00	41.666667	105.0	125.0	

Data Inspection and Cleaning

In [5]:
```
data.isna().sum()
```

Out[5]:
```
index                 0
Date                  1
Year                  1
Month                 1
Customer Age          1
Customer Gender       1
Country               1
State                 1
Product Category      1
Sub Category          1
Quantity              1
Unit Cost             1
Unit Price            1
Cost                  1
Revenue               0
Column1           32293
dtype: int64
```

In [6]:
```
data.drop(['Column1'], axis=1, inplace=True)
```

In [7]:
```
data.drop(['index'], axis=1,inplace=True)
```

In [8]:
```
data.dropna(inplace=True)
```

In [9]:
```
data.dtypes
```

Out[9]:
```
Date                 object
Year                float64
Month                object
Customer Age        float64
Customer Gender      object
Country              object
State                object
Product Category     object
Sub Category         object
Quantity            float64
Unit Cost           float64
Unit Price          float64
Cost                float64
Revenue             float64
dtype: object
```

In [10]:
```
data['Date'] = pd.to_datetime(data['Date'])
data['Year'] = data['Year'].astype(int)
data['Customer Gender'] = data['Customer Gender'].astype('category')
data['Country'] = data['Country'].astype('category')
data['State'] = data['State'].astype('category')
data['Product Category'] = data['Product Category'].astype('category')
data['Sub Category'] = data['Sub Category'].astype('category')
```

In [11]:
```
data.dtypes
```

Out[11]:
```
Date                datetime64[ns]
Year                         int32
Month                       object
Customer Age               float64
Customer Gender           category
Country                   category
State                     category
Product Category          category
Sub Category              category
Quantity                   float64
Unit Cost                  float64
Unit Price                 float64
Cost                       float64
Revenue                    float64
dtype: object
```

In [12]:
```
data.duplicated().sum()
```

Out[12]:
```
1
```

In [13]:
```
data.drop_duplicates(inplace=True)
```

In [14]:
```
data.duplicated().sum()
```

Out[14]:
```
0
```

Adding a new column with profit deatils

In [15]:
```
data['Profit'] = data['Revenue'] - data['Cost']
```

Cleaned Data

In [16]:
```
data.head(5)
```

Out[16]:

	Date	Year	Month	Customer Age	Customer Gender	Country	State	Product Category	Sub Category	Quantity	Unit Cost	Unit Price	Cost	Revenue	Profit
0	2016-02-19	2016	February	29.0	F	United States	Washington	Accessories	Tires and Tubes	1.0	80.00	109.000000	80.0	109.0	29.0
1	2016-02-20	2016	February	29.0	F	United States	Washington	Clothing	Gloves	2.0	24.50	28.500000	49.0	57.0	8.0
2	2016-02-27	2016	February	29.0	F	United States	Washington	Accessories	Tires and Tubes	3.0	3.67	5.000000	11.0	15.0	4.0
3	2016-03-12	2016	March	29.0	F	United States	Washington	Accessories	Tires and Tubes	2.0	87.50	116.500000	175.0	233.0	58.0
4	2016-03-12	2016	March	29.0	F	United States	Washington	Accessories	Tires and Tubes	3.0	35.00	41.666667	105.0	125.0	20.0

Exploratory Data Analysis (EDA)

In [17]:
```
data.describe()
```

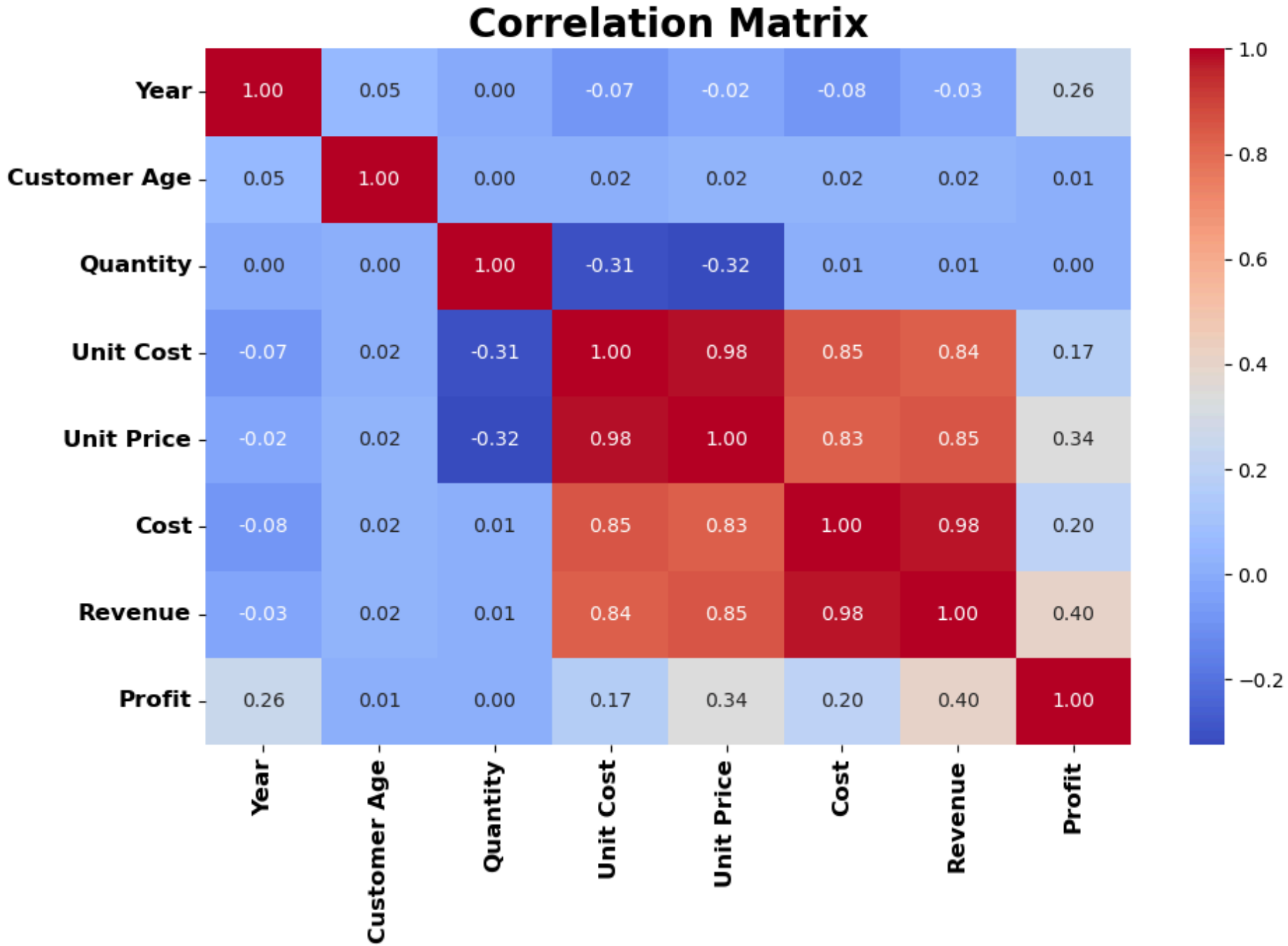
Out[17]:

	Year	Customer Age	Quantity	Unit Cost	Unit Price	Cost	Revenue	Profit
count	34865.000000	34865.000000	34865.000000	34865.000000	34865.000000	34865.000000	34865.000000	34865.000000
mean	2015.569253	36.382705	2.002524	349.890315	389.243248	576.020479	640.887652	64.867173
std	0.495188	11.113005	0.813948	490.019492	525.322781	690.503877	736.653849	152.881797
min	2015.000000	17.000000	1.000000	0.670000	0.666667	2.000000	2.000000	-937.000000
25%	2015.000000	28.000000	1.000000	45.000000	53.666667	85.000000	102.000000	5.000000
50%	2016.000000	35.000000	2.000000	150.000000	179.000000	261.000000	319.000000	27.000000
75%	2016.000000	44.000000	3.000000	455.000000	521.000000	769.000000	902.000000	96.000000
max	2016.000000	87.000000	3.000000	3240.000000	5082.000000	3600.000000	5082.000000	1842.000000

In [18]:
```
plt.figure(figsize=(10, 7))
sns.heatmap(data.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix',fontsize=20,weight='bold')
```

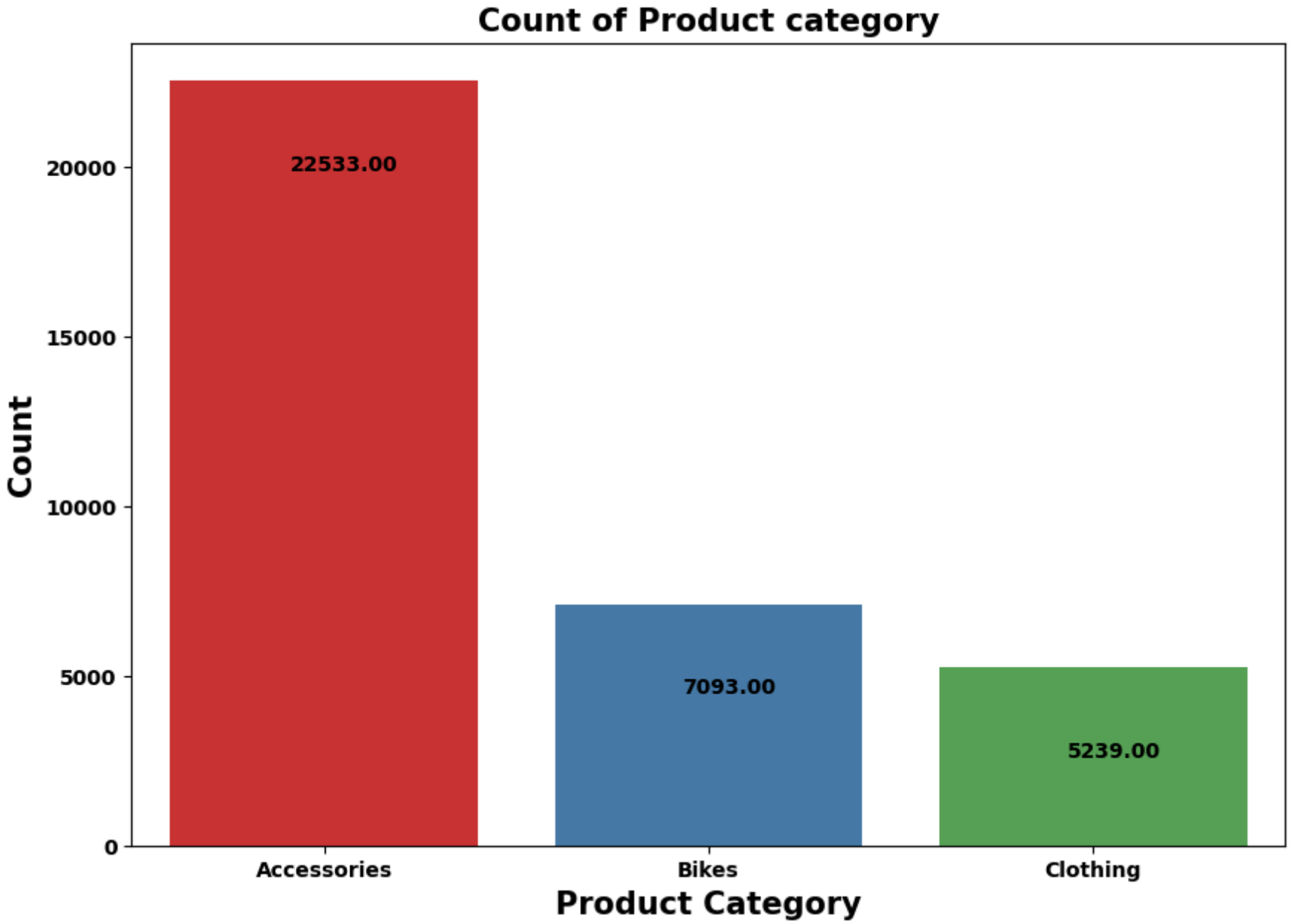
```
plt.xticks(fontsize=12,weight='bold')
plt.yticks(fontsize=12,weight='bold')
plt.tight_layout()
plt.show()
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_17308\2399596412.py:2: FutureWarning: The default value of numeric_only in DataFrame.co
rr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  sns.heatmap(data.corr(), annot=True, cmap='coolwarm', fmt=".2f")
```


Correlation Matrix
Year
Customer Age
Quantity
Unit Cost
Unit Price
Cost
Revenue
Profit
1.00 0.05 0.00 -0.07 -0.02 -0.08 -0.03 0.26
0.05 1.00 0.00 0.02 0.02 0.02 0.02 0.01
0.00 0.00 1.00 -0.31 -0.32 0.01 0.01 0.00
-0.07 0.02 -0.31 1.00 0.98 0.85 0.84 0.17
-0.02 0.02 -0.32 0.98 1.00 0.83 0.85 0.34
-0.08 0.02 0.01 0.85 0.83 1.00 0.98 0.20
-0.03 0.02 0.01 0.84 0.85 0.98 1.00 0.40
0.26 0.01 0.00 0.17 0.34 0.20 0.40 1.00
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

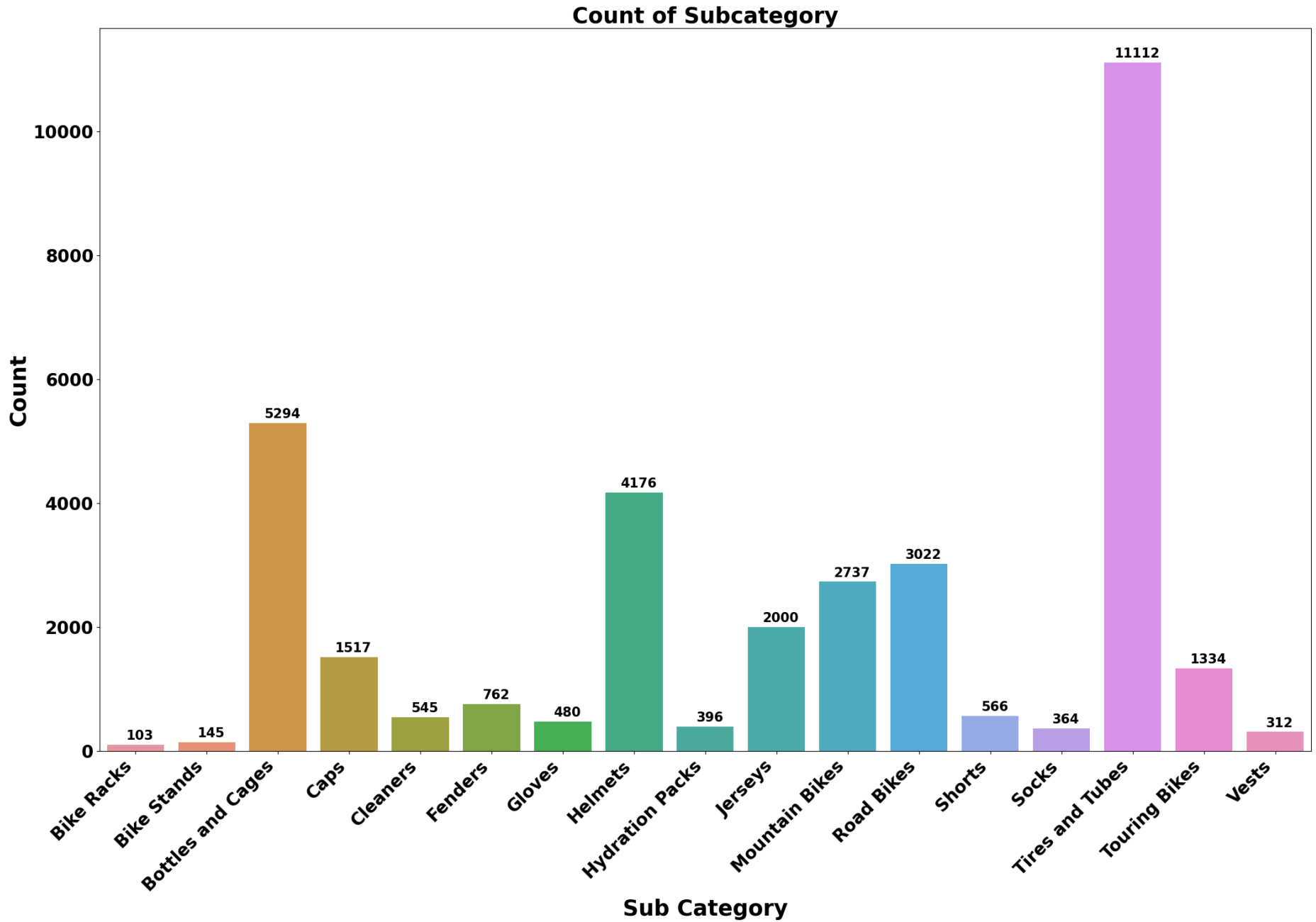
Univariate Analysis

In [19]:
```
plt.figure(figsize=(10, 7))
df1=sns.countplot(x=data['Product Category'],palette='Set1')
for p in df1.patches:
    df1.annotate(format(p.get_height(), '.2f'),
                 (p.get_x() + p.get_width() / 2., p.get_height()),
                 ha = 'center', va = 'center',
                 xytext = (10, -40),
                 textcoords = 'offset points',fontsize=10,weight='bold')
plt.xticks(fontsize=10,weight='bold')
plt.yticks(fontsize=10,weight='bold')
plt.title('Count of Product category',fontsize=15,weight='bold')
plt.xlabel('Product Category',fontsize=15,weight='bold')
plt.ylabel('Count',fontsize=15,weight='bold')
plt.show()
```



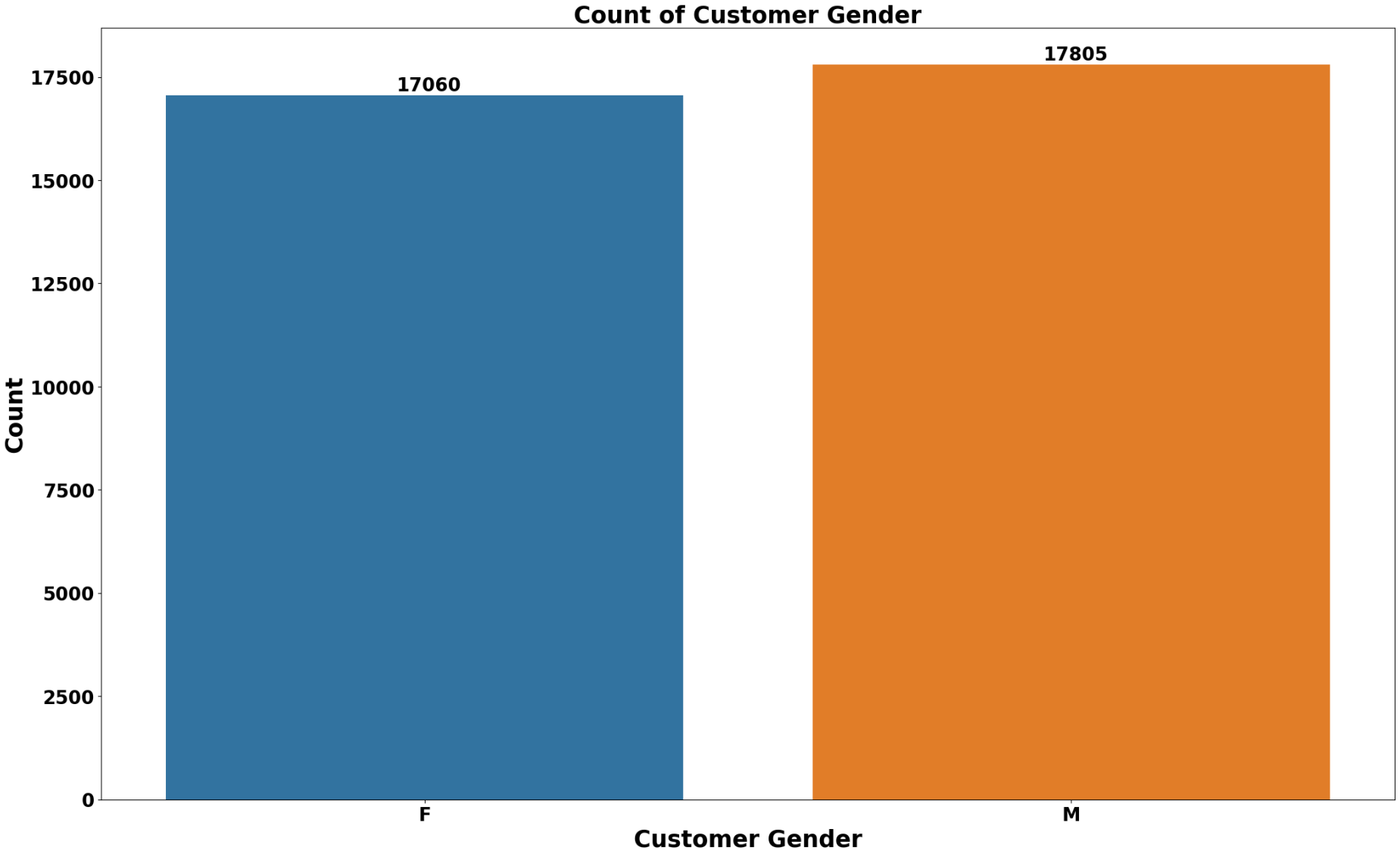
In [20]:
```
plt.figure(figsize=(25, 15))
df2=sns.countplot(x=data['Sub Category'])
for p in df2.patches:
    df2.annotate(format(p.get_height(),'.0f'),
```

```
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha = 'center', va = 'center',
                xytext = (5, 10),
                textcoords = 'offset points',
                fontsize=15,
                weight='bold')
plt.xticks(fontsize=20,weight='bold',rotation=45, ha='right')
plt.yticks(fontsize=20,weight='bold')
plt.title('Count of Subcategory',fontsize=25,weight='bold')
plt.xlabel('Sub Category',fontsize=25,weight='bold')
plt.ylabel('Count',fontsize=25,weight='bold')
plt.show()
```



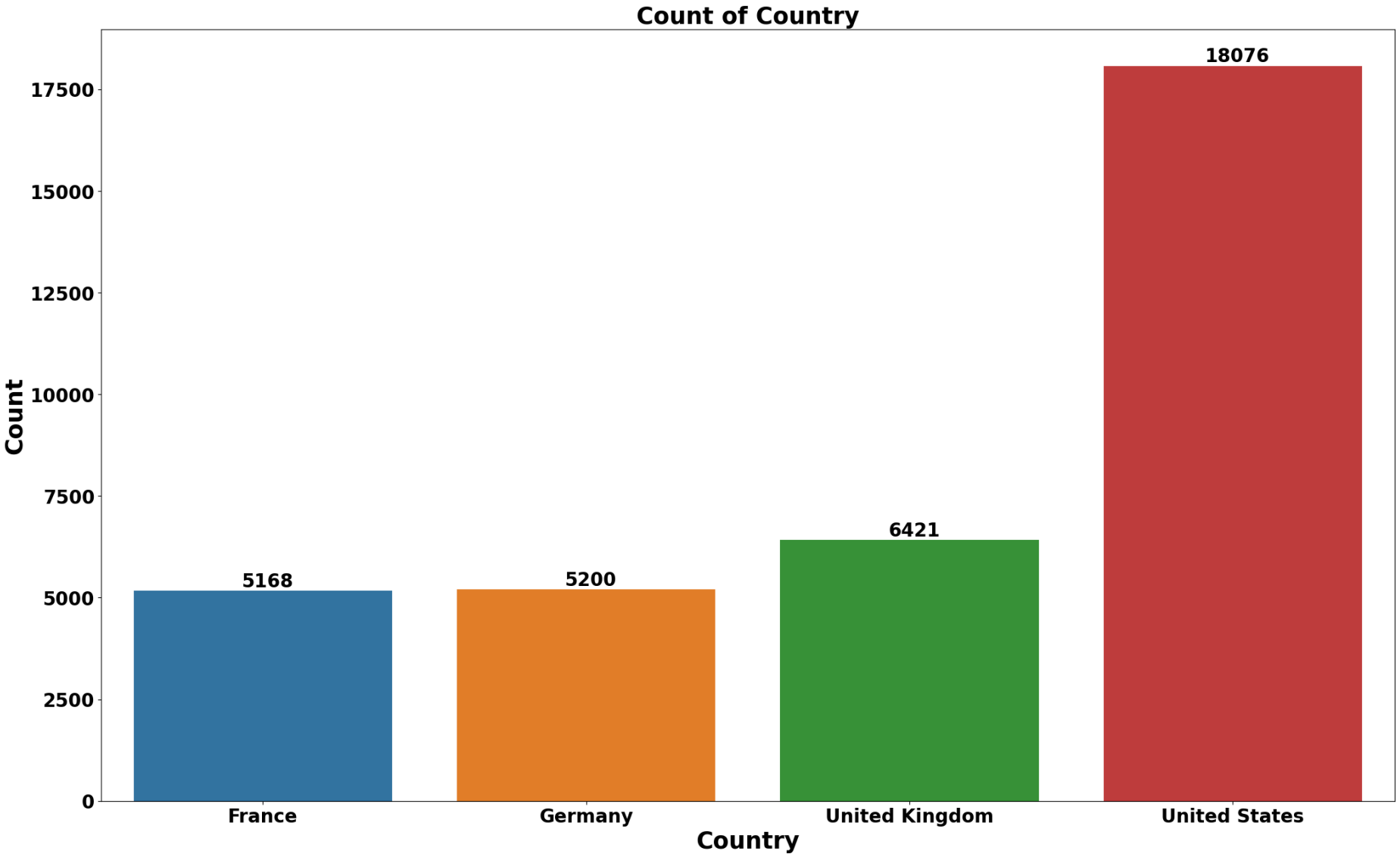
In [21]:
```
plt.figure(figsize=(25, 15))
df3=sns.countplot(x=data['Customer Gender'])
```

```
for p in df3.patches:
    df3.annotate(format(p.get_height(),'.0f'),
                 (p.get_x() + p.get_width() / 2., p.get_height()),
                 ha = 'center', va = 'center',
                 xytext = (5, 10),
                 textcoords = 'offset points',
                 fontsize=20,
                 weight='bold')
plt.xticks(fontsize=20,weight='bold')
plt.yticks(fontsize=20,weight='bold')
plt.title('Count of Customer Gender',fontsize=25,weight='bold')
plt.xlabel('Customer Gender',fontsize=25,weight='bold')
plt.ylabel('Count',fontsize=25,weight='bold')
plt.show()
```



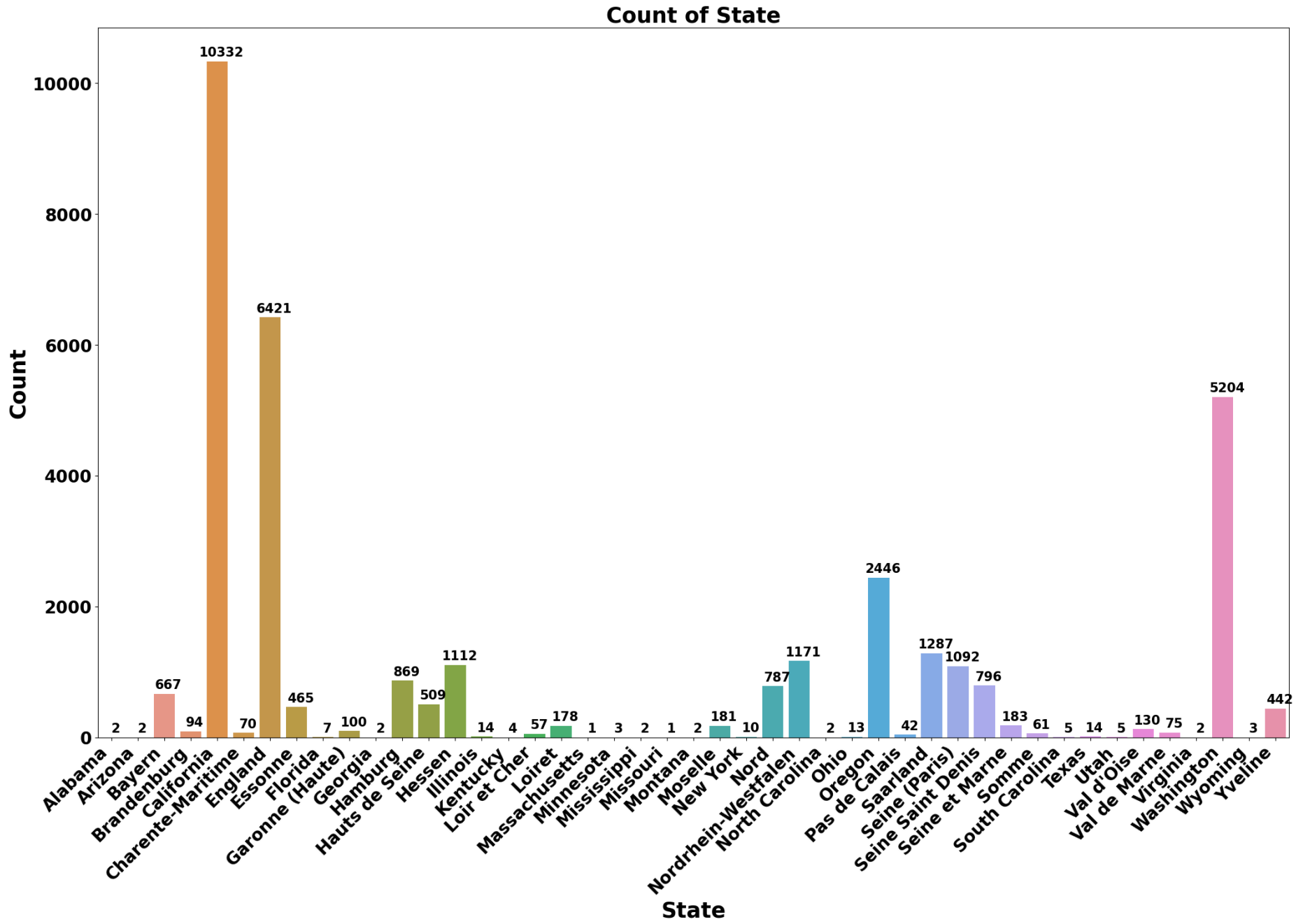
In [22]:
```
plt.figure(figsize=(25, 15))
df4=sns.countplot(x=data['Country'])
for p in df4.patches:
    df4.annotate(format(p.get_height(),'.0f'),
                 (p.get_x() + p.get_width() / 2., p.get_height()),
                 ha = 'center', va = 'center',
                 xytext = (5, 10),
                 textcoords = 'offset points',
```

```
                fontsize=20,
                weight='bold')
plt.xticks(fontsize=20,weight='bold')
plt.yticks(fontsize=20,weight='bold')
plt.title('Count of Country',fontsize=25,weight='bold')
plt.xlabel('Country',fontsize=25,weight='bold')
plt.ylabel('Count',fontsize=25,weight='bold')
plt.show()
```



In [23]:

```
plt.figure(figsize=(25, 15))
df5=sns.countplot(x=data['State'])
for p in df5.patches:
    df5.annotate(format(p.get_height(),'.0f'),
                 (p.get_x() + p.get_width() / 2., p.get_height()),
                 ha = 'center', va = 'center',
                 xytext = (5, 10),
                 textcoords = 'offset points',
                 fontsize=15,
                 weight='bold')
plt.xticks(fontsize=20,weight='bold',rotation=45, ha='right')
plt.yticks(fontsize=20,weight='bold')
plt.title('Count of State',fontsize=25,weight='bold')
plt.xlabel('State',fontsize=25,weight='bold')
plt.ylabel('Count',fontsize=25,weight='bold')
plt.show()
```



In [24]:
```
data['State'].value_counts()
```

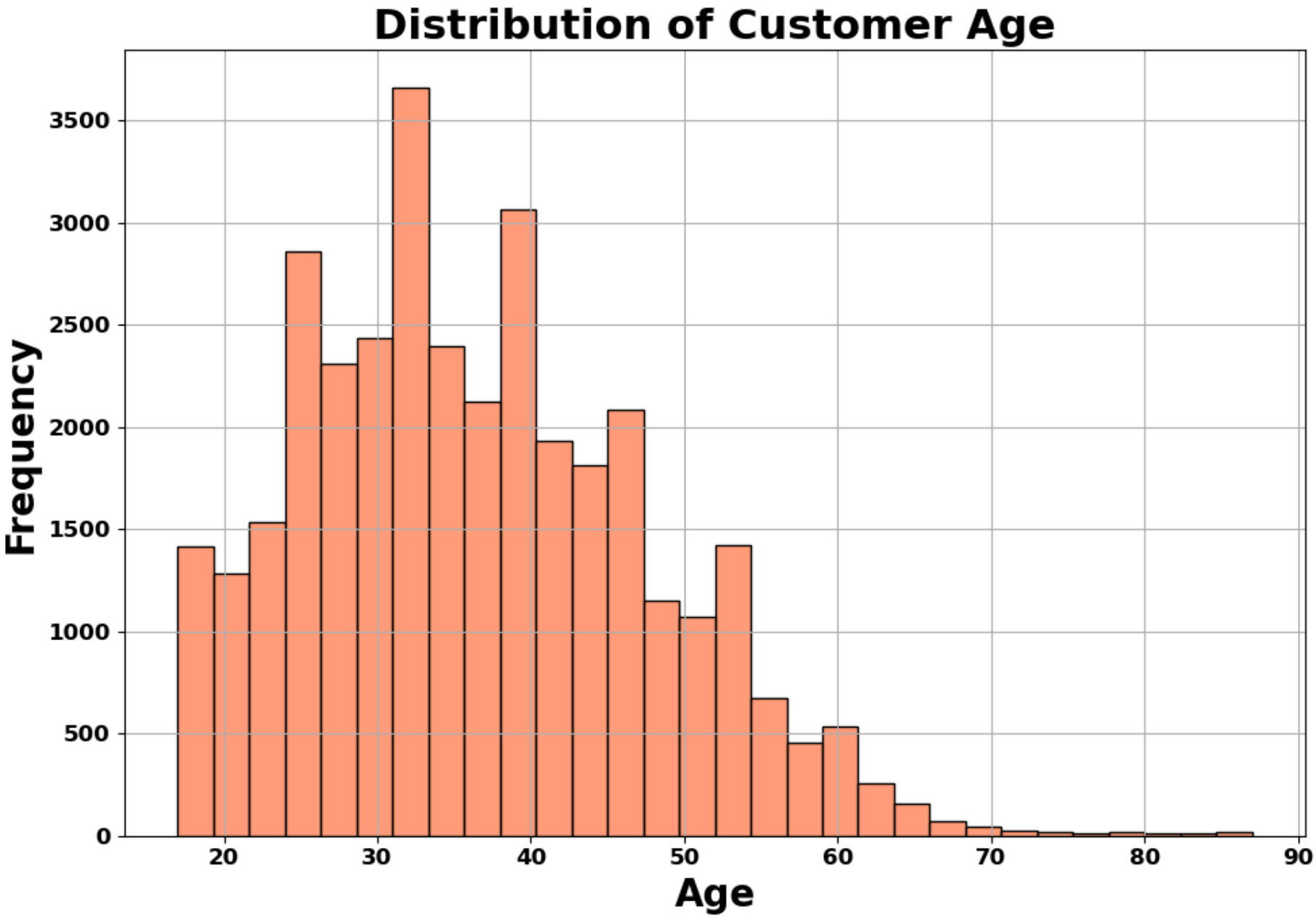
Out[24]:

California	10332
England	6421
Washington	5204
Oregon	2446
Saarland	1287
Nordrhein-Westfalen	1171
Hessen	1112
Seine (Paris)	1092
Hamburg	869
Seine Saint Denis	796
Nord	787
Bayern	667
Hauts de Seine	509
Essonne	465
Yveline	442
Seine et Marne	183
Moselle	181
Loiret	178
Val d'Oise	130
Garonne (Haute)	100
Brandenburg	94
Val de Marne	75
Charente-Maritime	70
Somme	61
Loir et Cher	57
Pas de Calais	42
Texas	14
Illinois	14
Ohio	13
New York	10
Florida	7
Utah	5
South Carolina	5
Kentucky	4
Minnesota	3
Wyoming	3
Virginia	2
Alabama	2
North Carolina	2
Arizona	2
Mississippi	2
Georgia	2
Montana	2
Missouri	1

```
Massachusetts              1
Name: State, dtype: int64
```

In [25]:

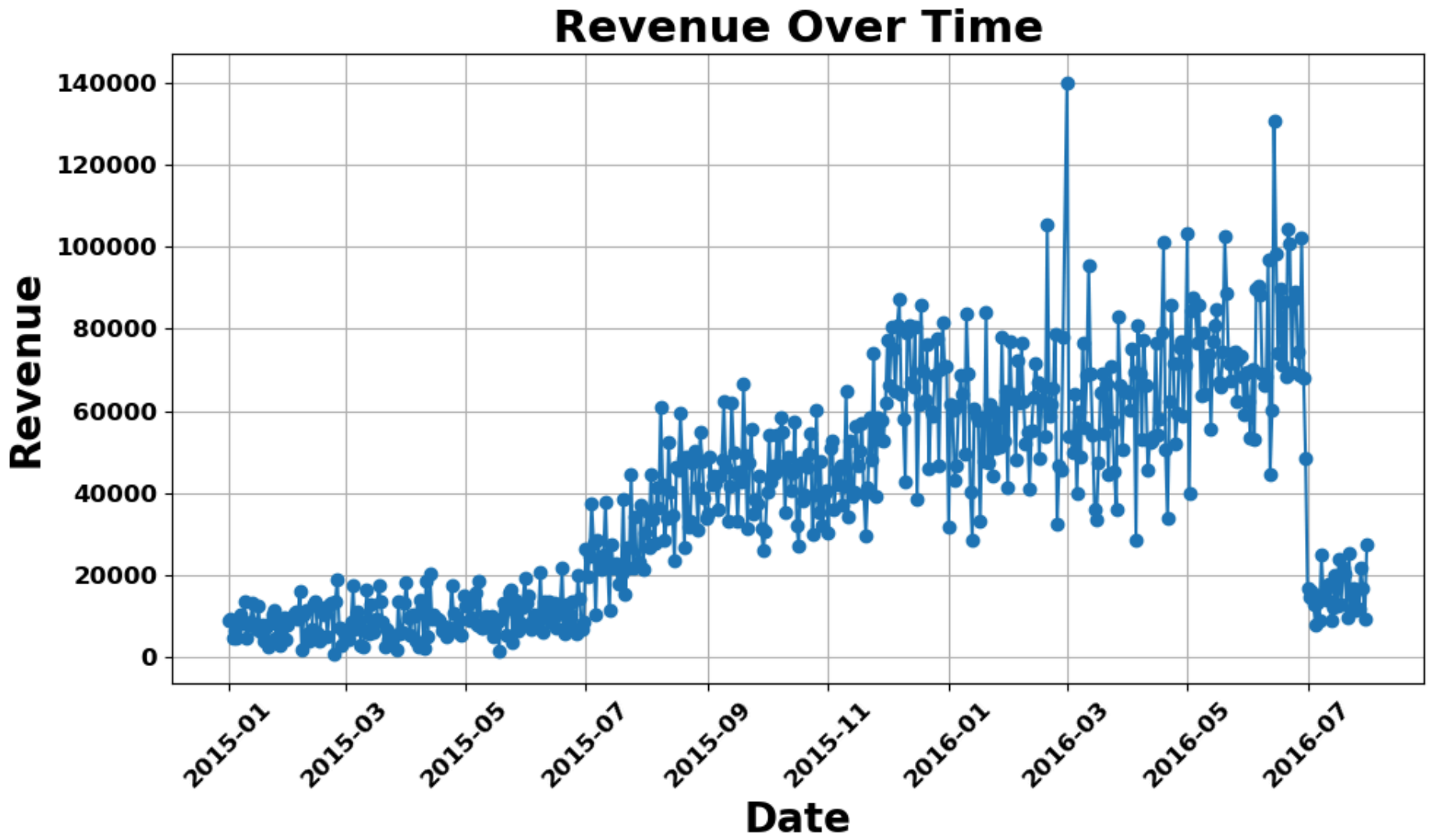
```
plt.figure(figsize=(10,7))
sns.histplot(data['Customer Age'], bins=30, color='coral', edgecolor='black')
plt.title('Distribution of Customer Age', fontsize=22, weight='bold')
plt.xlabel('Age', fontsize=20, weight='bold')
plt.ylabel('Frequency', fontsize=20, weight='bold')
plt.xticks(fontsize=12, weight='bold')
plt.yticks(fontsize=12, weight='bold')
plt.grid(True)
plt.tight_layout()
plt.show()
```



Bivariate Analysis

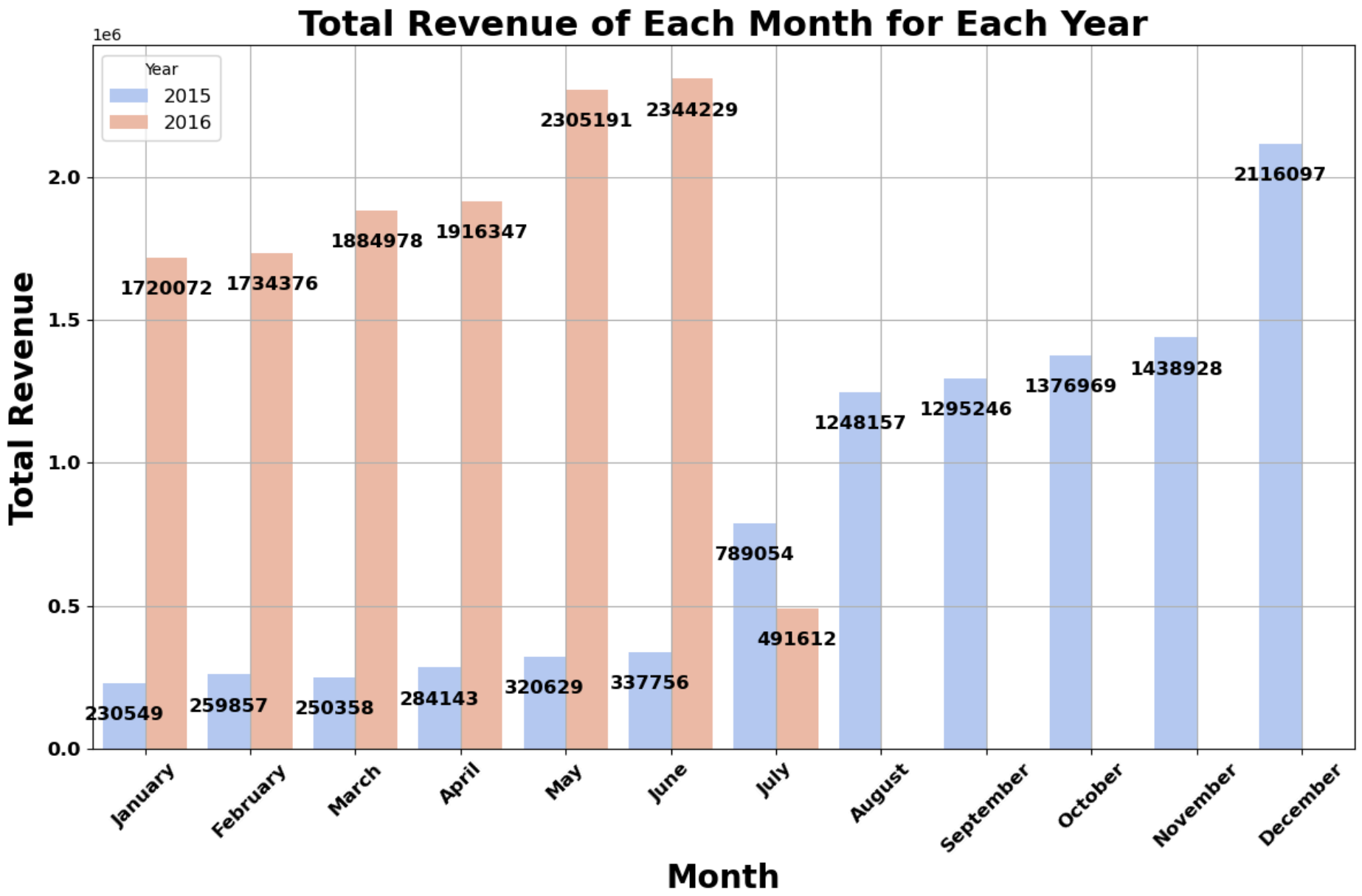
In [26]:

```
revenue_over_time = data.groupby('Date')['Revenue'].sum()

plt.figure(figsize=(10, 6))
plt.plot(revenue_over_time.index, revenue_over_time.values, marker='o', linestyle='-')

plt.title('Revenue Over Time', fontsize=22, weight='bold')
plt.xlabel('Date', fontsize=20, weight='bold')
plt.ylabel('Revenue', fontsize=20, weight='bold')

plt.xticks(fontsize=12, weight='bold', rotation=45)

plt.yticks(fontsize=12, weight='bold')

plt.grid(True)

plt.tight_layout()

plt.show()
```



In [27]:
```
monthly_revenue = data.groupby(['Year', 'Month'])['Revenue'].sum().reset_index()
months_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July',
                'August', 'September', 'October', 'November', 'December']

plt.figure(figsize=(12, 8))
df6=sns.barplot(x='Month', y='Revenue', hue='Year', data=monthly_revenue, order=months_order, palette='coolwarm')
plt.title('Total Revenue of Each Month for Each Year',fontsize=22,
               weight='bold')
plt.xlabel('Month',fontsize=20,
```

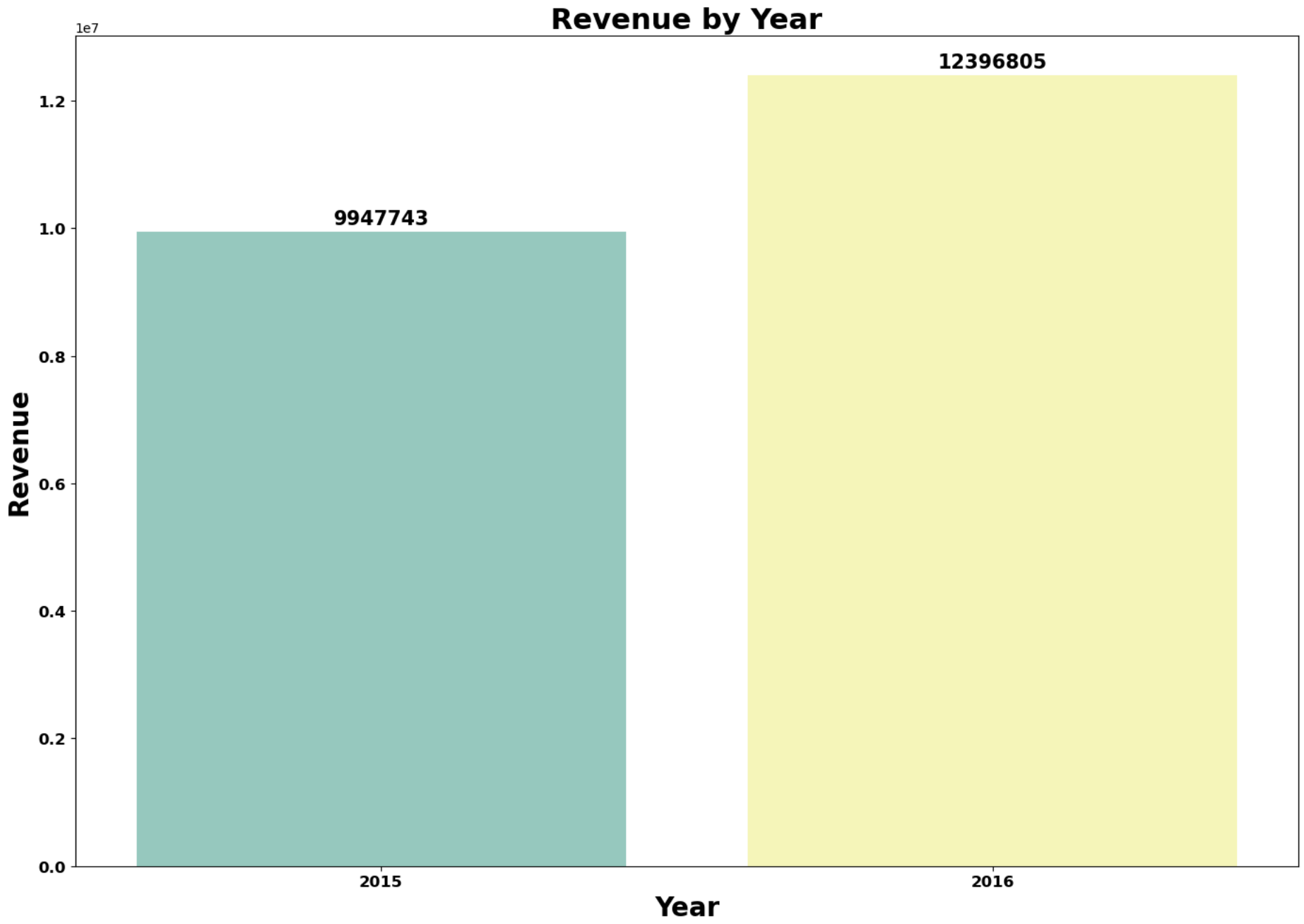
```
                weight='bold')
plt.ylabel('Total Revenue',fontsize=20,
                weight='bold')
plt.xticks(rotation=45,fontsize=12,
                weight='bold')
plt.yticks(fontsize=12,
                weight='bold')
plt.legend(title='Year', loc='upper left',fontsize=12)
plt.grid(True)
for p in df6.patches:
    df6.annotate(format(p.get_height(), '.0f'),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha = 'center', va = 'center',
                xytext = (0, -20),
                textcoords = 'offset points',
                fontsize=12,
                weight='bold')
plt.tight_layout()
plt.show()
```



Total Revenue for Each Year

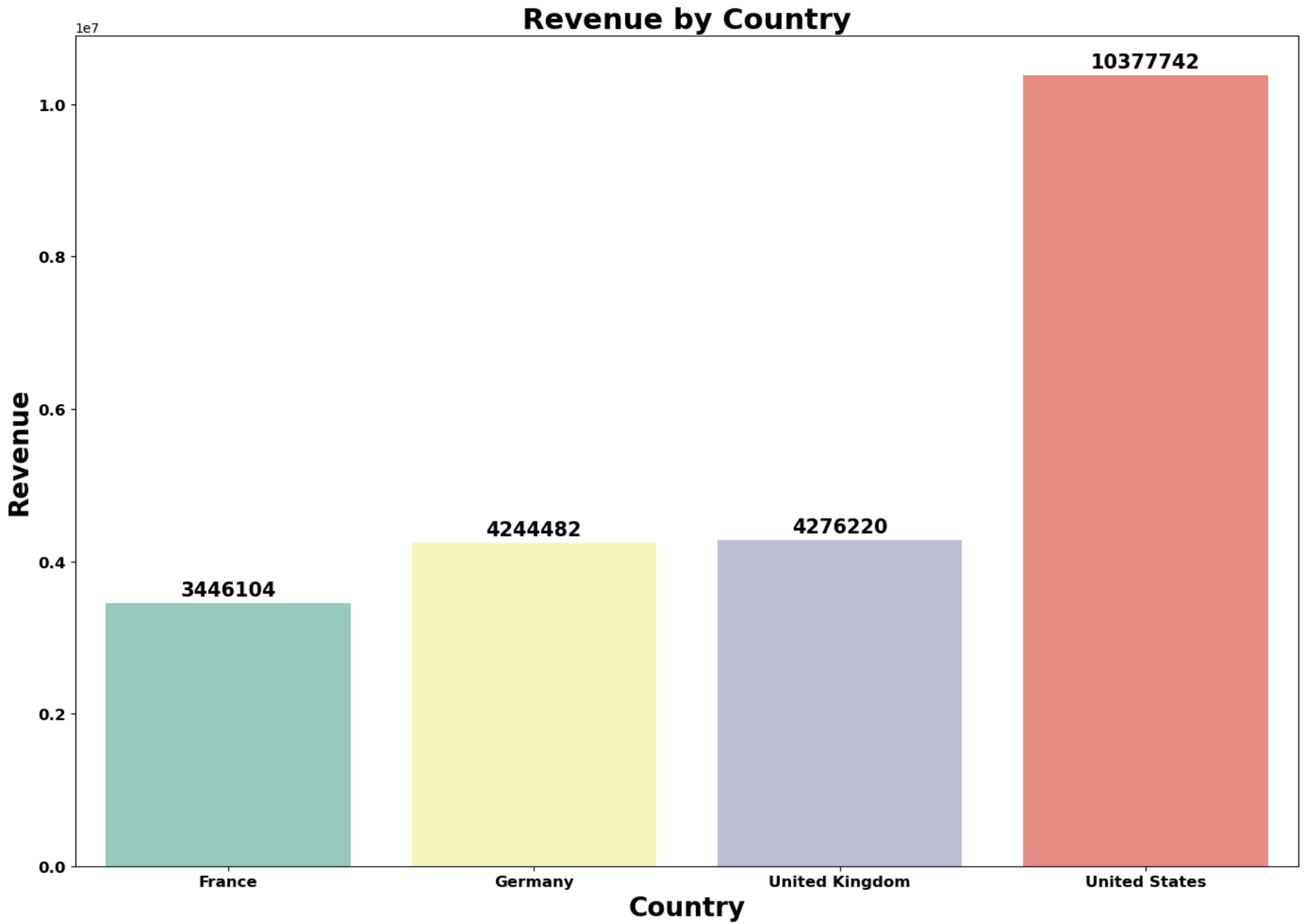
In [28]:
```
revenue_by_year = data.groupby(by='Year')[['Revenue']].sum().reset_index()
plt.figure(figsize=(14, 10))
```

```
df7=sns.barplot(x='Year', y='Revenue', data=revenue_by_year, palette='Set3')
plt.title('Revenue by Year', fontsize=22, weight='bold')
plt.xlabel('Year', fontsize=20, weight='bold')
plt.ylabel('Revenue', fontsize=20, weight='bold')
plt.xticks(fontsize=12, weight='bold')
plt.yticks(fontsize=12, weight='bold')
for p in df7.patches:
    df7.annotate(format(p.get_height(), '.0f'),
                 (p.get_x() + p.get_width() / 2., p.get_height()),
                 ha = 'center', va = 'center',
                 xytext = (0, 10),
                 textcoords = 'offset points',
                 fontsize=15,
                 weight='bold')
plt.tight_layout()
plt.show()
```



In [29]:
```
revenue_by_country = data.groupby('Country')['Revenue'].sum().reset_index()
plt.figure(figsize=(14, 10))
```

```
df8=sns.barplot(x='Country', y='Revenue', data=revenue_by_country, palette='Set3')
plt.title('Revenue by Country', fontsize=22, weight='bold')
plt.xlabel('Country', fontsize=20, weight='bold')
plt.ylabel('Revenue', fontsize=20, weight='bold')
plt.xticks(fontsize=12, weight='bold')
plt.yticks(fontsize=12, weight='bold')
for p in df8.patches:
    df8.annotate(format(p.get_height(), '.0f'),
                 (p.get_x() + p.get_width() / 2., p.get_height()),
                 ha = 'center', va = 'center',
                 xytext = (0, 10),
                 textcoords = 'offset points',
                 fontsize=15,
                 weight='bold')
plt.tight_layout()
plt.show()
```



Top 10 States Ranked by Revenue

In [30]:
```
data.groupby(['Country','State'])['Revenue'].sum().reset_index().sort_values('Revenue',ascending=False).head(10)
```

Out[30]:

	Country	State	Revenue
139	United States	California	6076916.0
96	United Kingdom	England	4276220.0
177	United States	Washington	2873511.0
164	United States	Oregon	1383186.0
76	Germany	Saarland	1055844.0
71	Germany	Nordrhein-Westfalen	931677.0
58	Germany	Hessen	917107.0
32	France	Seine (Paris)	719148.0
56	Germany	Hamburg	714036.0
47	Germany	Bayern	537380.0

Bottom 10 States Ranked by Revenue

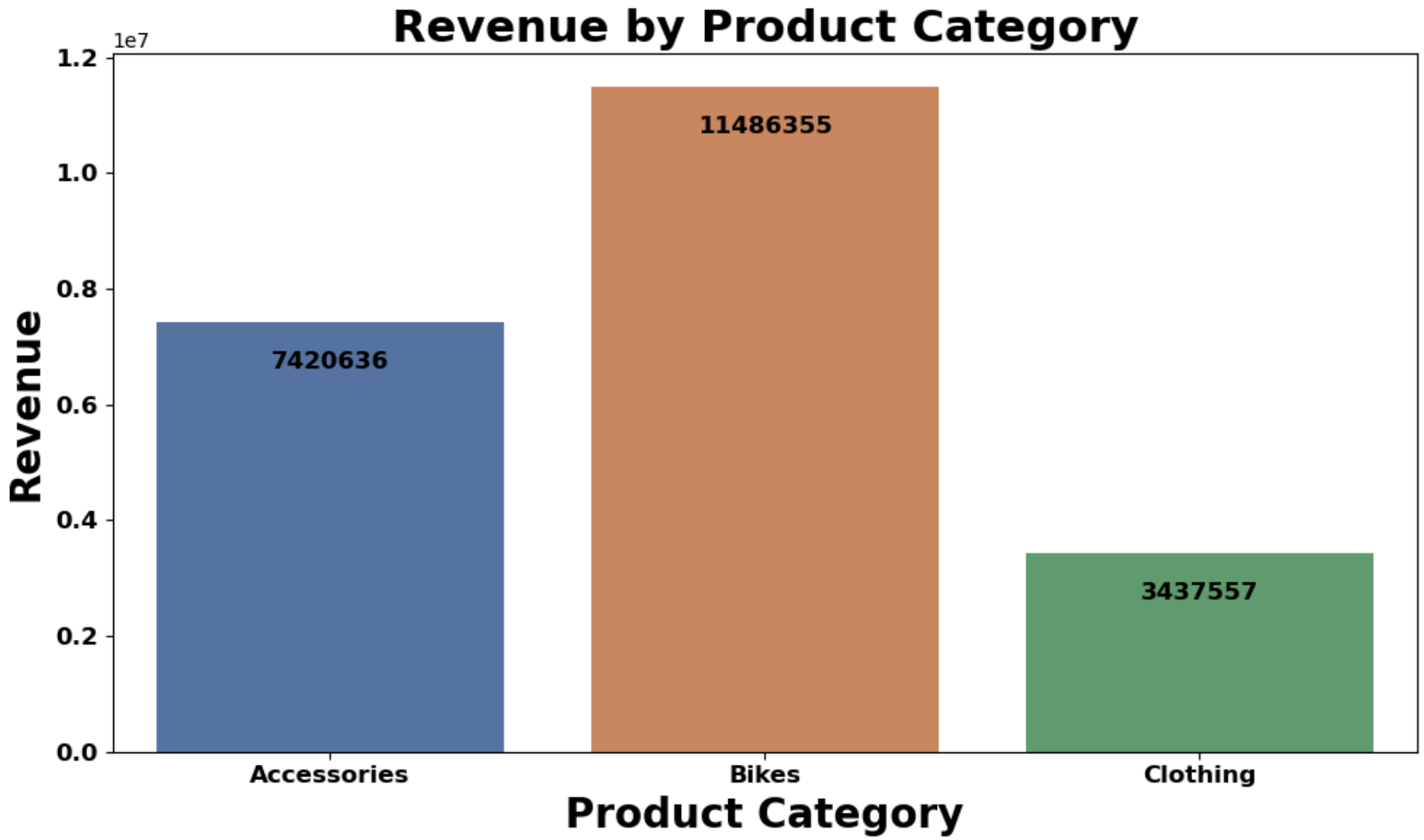
In [31]:
```
data.groupby(['Country','State'])['Revenue'].sum().reset_index().sort_values('Revenue',ascending=False).tail(10)
```

Out[31]:

	Country	State	Revenue
68	Germany	Moselle	0.0
69	Germany	New York	0.0
70	Germany	Nord	0.0
72	Germany	North Carolina	0.0
73	Germany	Ohio	0.0
74	Germany	Oregon	0.0
75	Germany	Pas de Calais	0.0
77	Germany	Seine (Paris)	0.0
78	Germany	Seine Saint Denis	0.0
179	United States	Yveline	0.0

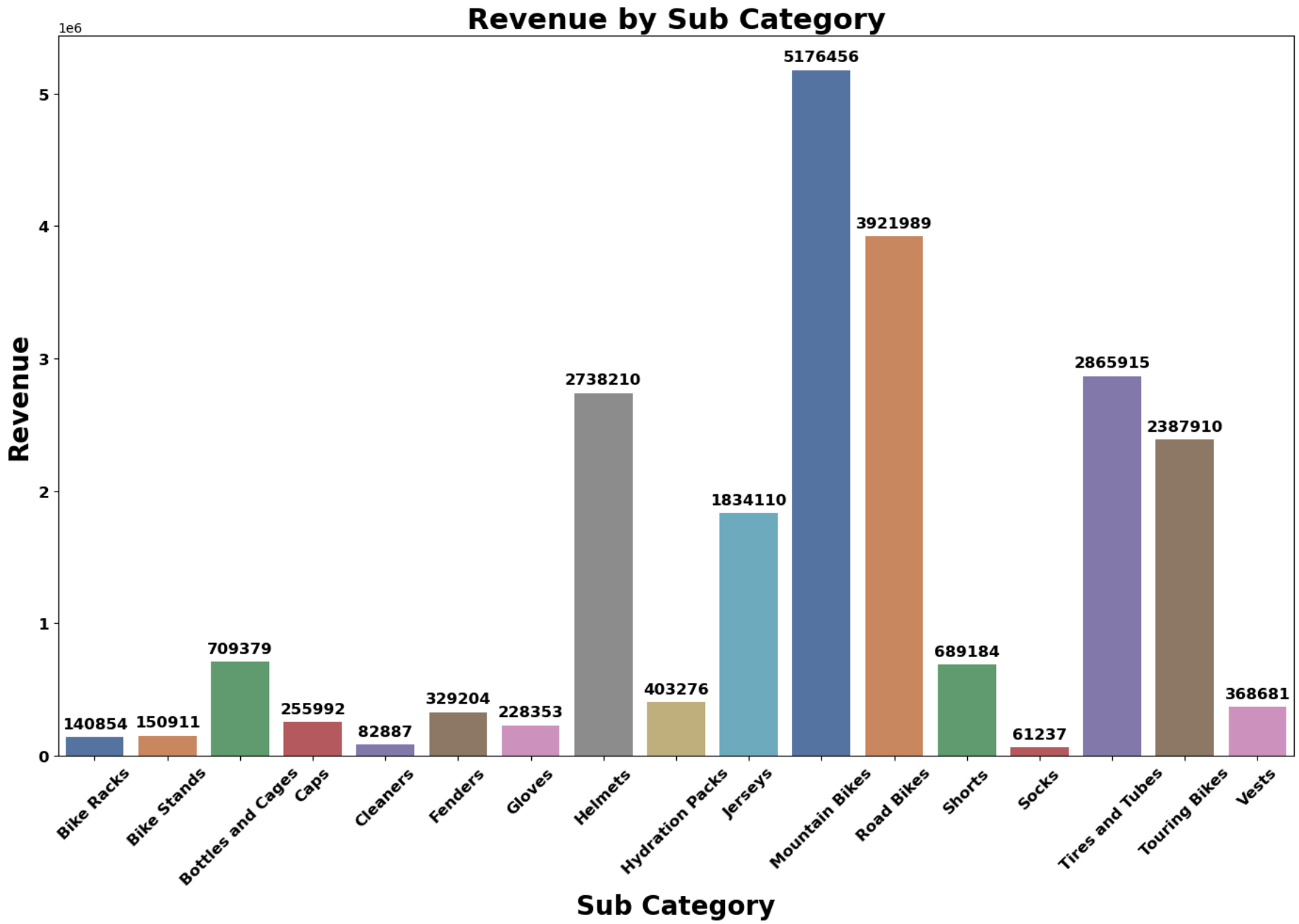
In [32]:

```
revenue_by_category = data.groupby('Product Category')['Revenue'].sum().reset_index()
plt.figure(figsize=(10, 6))
df9=sns.barplot(x='Product Category', y='Revenue', data=revenue_by_category, palette='deep')
plt.title('Revenue by Product Category', fontsize=22, weight='bold')
plt.xlabel('Product Category', fontsize=20, weight='bold')
plt.ylabel('Revenue', fontsize=20, weight='bold')
plt.xticks(fontsize=12, weight='bold')
plt.yticks(fontsize=12, weight='bold')
for p in df9.patches:
    df9.annotate(format(p.get_height(), '.0f'),
                 (p.get_x() + p.get_width() / 2., p.get_height()),
                 ha = 'center', va = 'center',
                 xytext = (0, -20),
                 textcoords = 'offset points',
                 fontsize=12,
                 weight='bold')
plt.tight_layout()
plt.show()
```



In [33]:
```
revenue_by_subcategory = data.groupby('Sub Category')['Revenue'].sum().reset_index()
plt.figure(figsize=(14, 10))
df10=sns.barplot(x='Sub Category', y='Revenue', data=revenue_by_subcategory, palette='deep')
plt.title('Revenue by Sub Category', fontsize=22, weight='bold')
plt.xlabel('Sub Category', fontsize=20, weight='bold')
plt.ylabel('Revenue', fontsize=20, weight='bold')
plt.xticks(fontsize=12, weight='bold',rotation=45)
plt.yticks(fontsize=12, weight='bold')
for p in df10.patches:
```

```
    df10.annotate(format(p.get_height(), '.0f'),
                  (p.get_x() + p.get_width() / 2., p.get_height()),
                  ha = 'center', va = 'center',
                  xytext = (0, 10),
                  textcoords = 'offset points',
                  fontsize=12,
                  weight='bold')
plt.tight_layout()
plt.show()
```

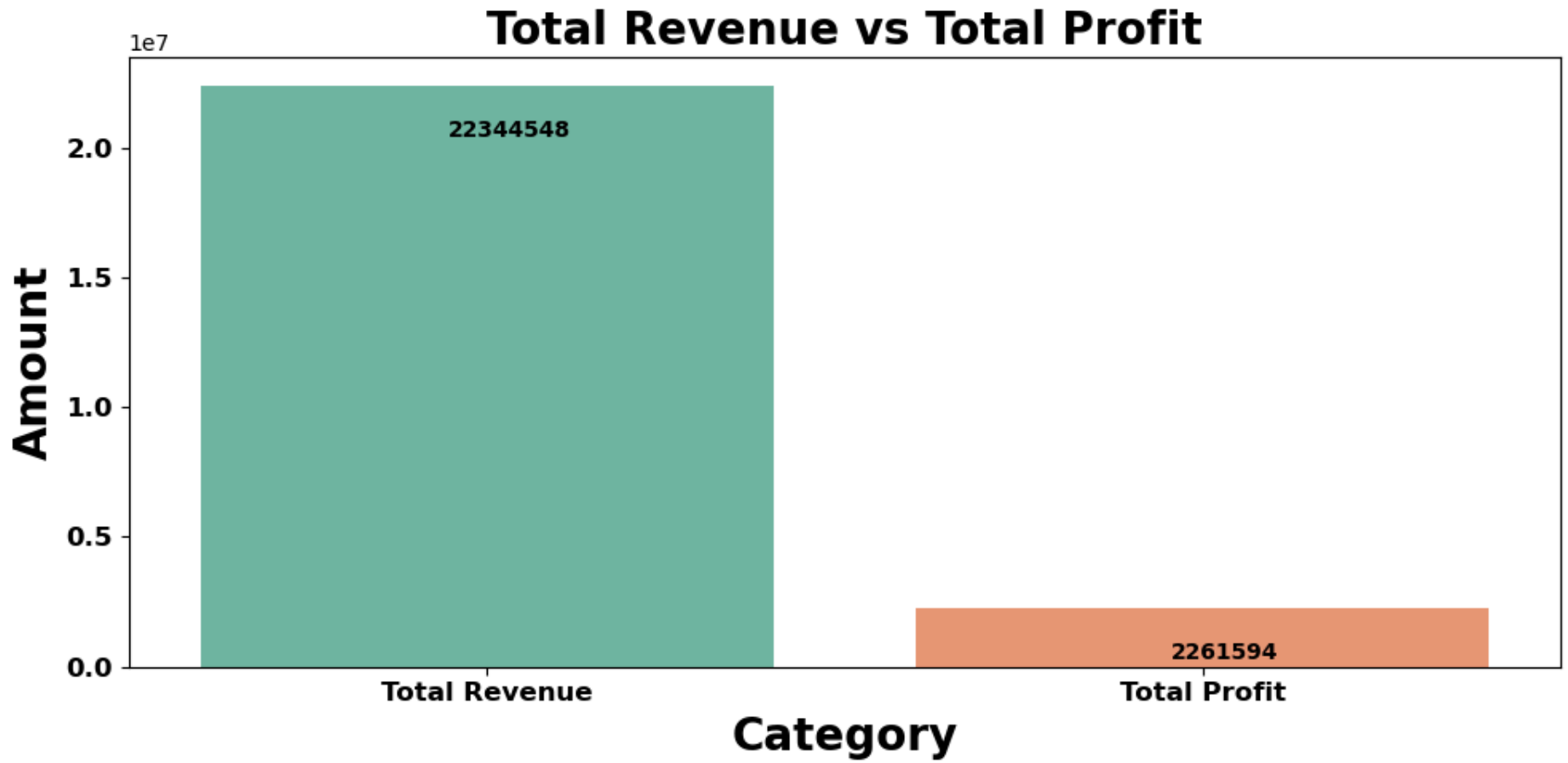


Revenue Analysis Highlights:

- **A significant portion of customers falls within the *30 to 40* age group.**
- **There is a noticeable month-on-month *increase* in revenue.**
- **Revenue experienced a commendable *24% growth* in 2016 compared to 2015.**
- **The *United States* emerges as the top-performing country.**
- ***Bikes* are identified as the leading product category.**
- ***Mountain Bikes* stand out as the top subcategory.**

To ensure robustness, these findings will undergo cross-verification against profitability metrics to identify the true top performers.

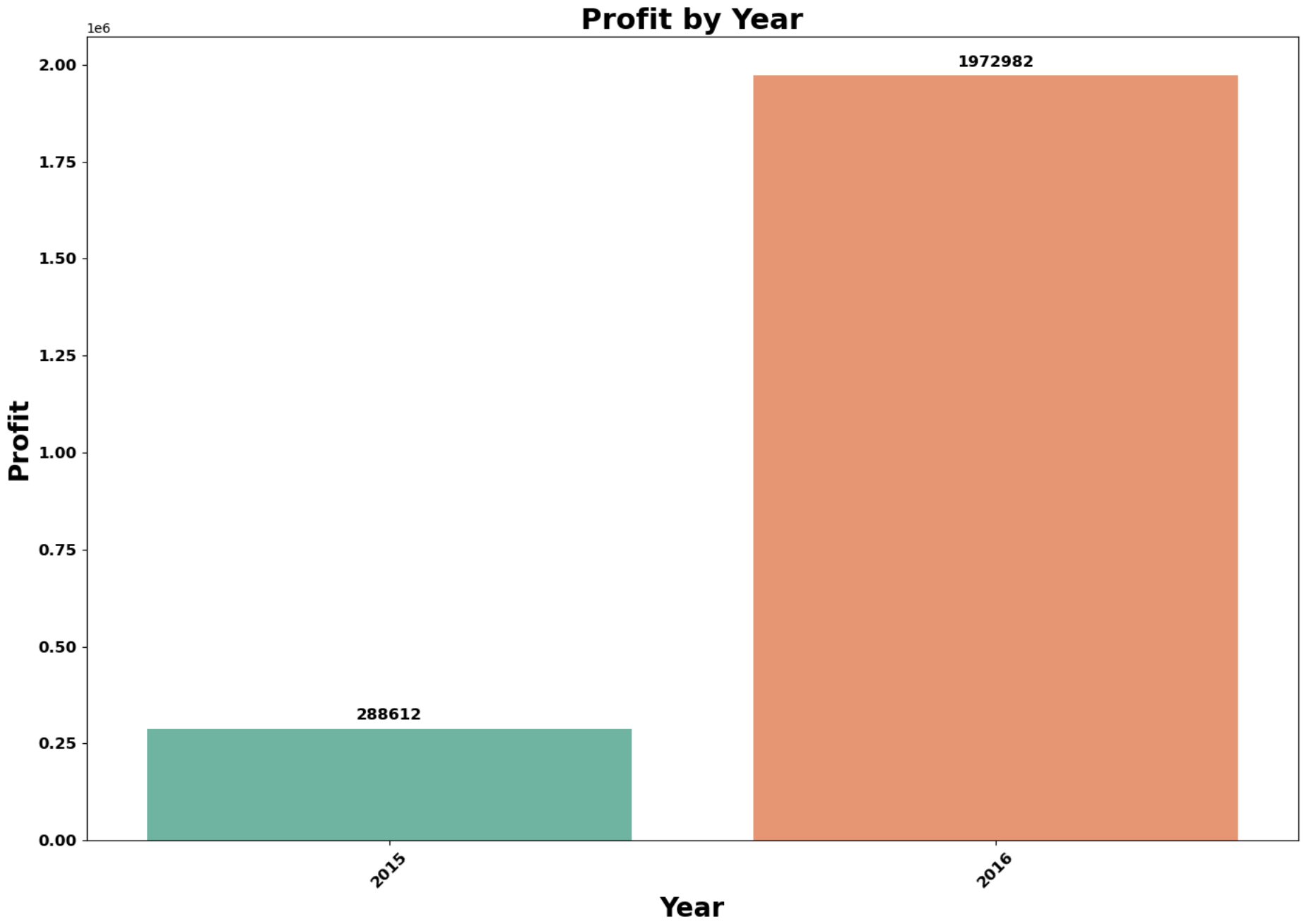
In [34]:
```
plt.figure(figsize=(10,5))
total_revenue = data['Revenue'].sum()
total_profit = data['Profit'].sum()
df=pd.DataFrame({'Category': ['Total Revenue', 'Total Profit'],
                           'Amount': [total_revenue, total_profit]})
df11=sns.barplot(x='Category',y='Amount',data=df, palette='Set2')
plt.xlabel('Category',fontsize=20,weight='bold')
plt.ylabel('Amount',fontsize=20,weight='bold')
plt.title('Total Revenue vs Total Profit',fontsize=20,weight='bold')
plt.xticks(fontsize=12,weight='bold')
plt.yticks(fontsize=12,weight='bold')
for p in df11.patches:
    df11.annotate(format(p.get_height(), '.0f'),
                  (p.get_x() + p.get_width() / 2., p.get_height()),
                  ha = 'center', va = 'center',
                  xytext = (10, -20),
                  textcoords = 'offset points',
                  fontsize=10,weight='bold')
plt.tight_layout()
plt.show()
```



Total Profit for Each Year

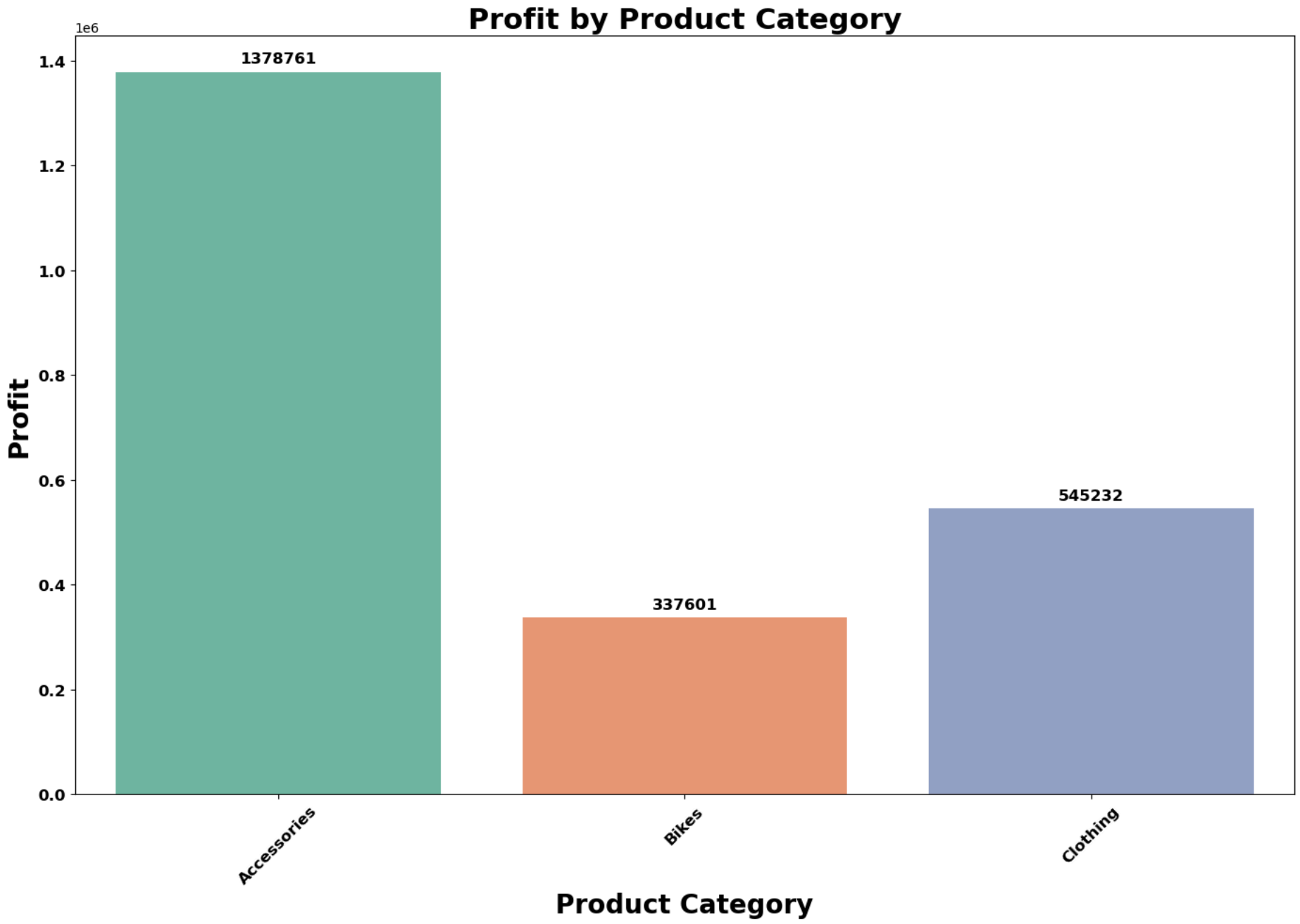
In [35]:
```
profit_by_year = data.groupby('Year')['Profit'].sum().reset_index()
plt.figure(figsize=(14, 10))
df11=sns.barplot(x='Year', y='Profit', data=profit_by_year, palette='Set2')
plt.title('Profit by Year', fontsize=22, weight='bold')
plt.xlabel('Year', fontsize=20, weight='bold')
plt.ylabel('Profit', fontsize=20, weight='bold')
plt.xticks(fontsize=12, weight='bold',rotation=45)
plt.yticks(fontsize=12, weight='bold')
for p in df11.patches:
    df11.annotate(format(p.get_height(), '.0f'),
                  (p.get_x() + p.get_width() / 2., p.get_height()),
                  ha = 'center', va = 'center',
```

```
                xytext = (0, 10),
                textcoords = 'offset points',
                fontsize=12,
                weight='bold')
plt.tight_layout()
plt.show()
```



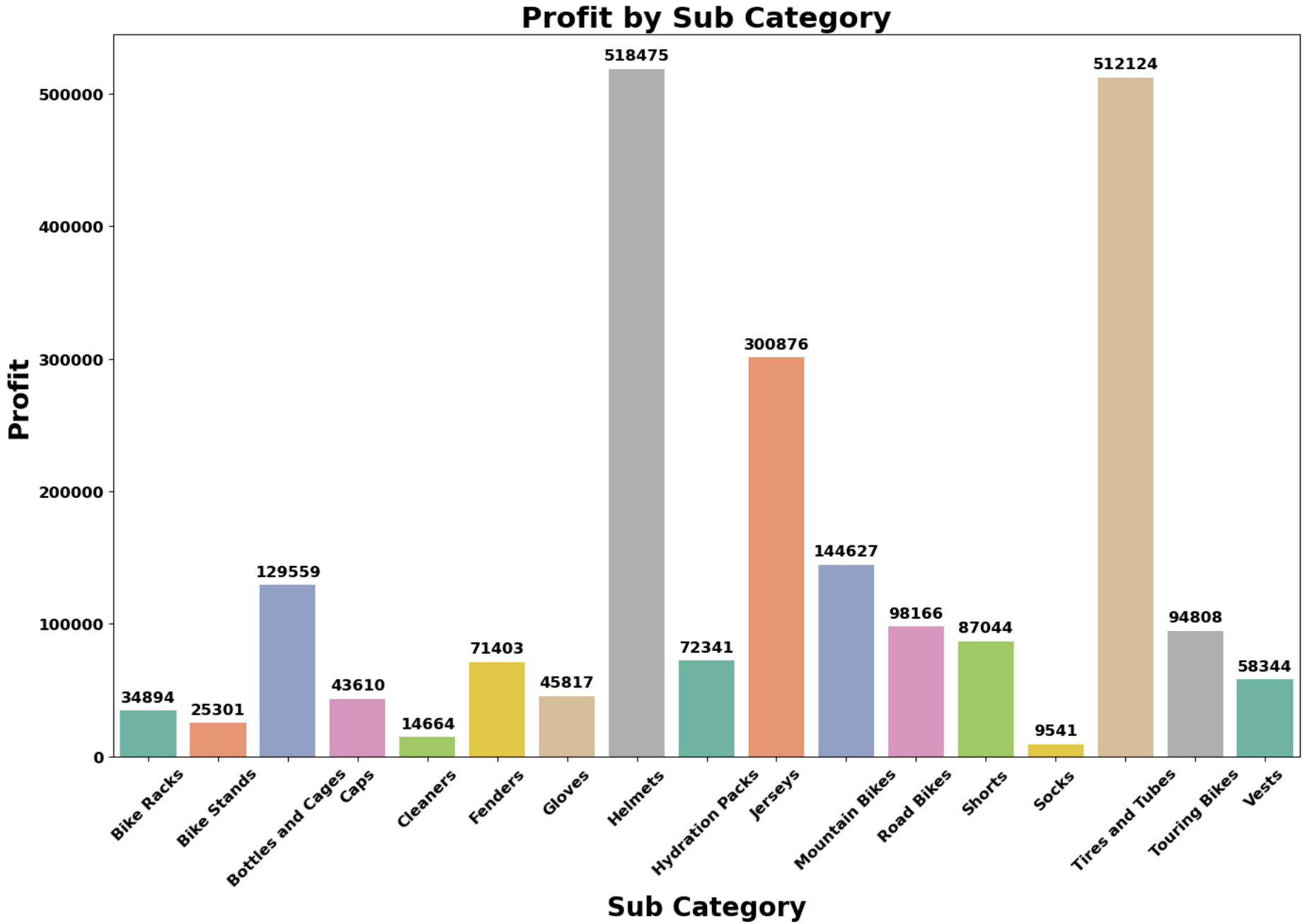
In [54]:
```
profit_by_productcategory = data.groupby('Product Category')['Profit'].sum().reset_index()
plt.figure(figsize=(14, 10))
```

```
df20=sns.barplot(x='Product Category', y='Profit', data=profit_by_productcategory, palette='Set2')
plt.title('Profit by Product Category', fontsize=22, weight='bold')
plt.xlabel('Product Category', fontsize=20, weight='bold')
plt.ylabel('Profit', fontsize=20, weight='bold')
plt.xticks(fontsize=12, weight='bold',rotation=45)
plt.yticks(fontsize=12, weight='bold')
for p in df20.patches:
    df20.annotate(format(p.get_height(), '.0f'),
                  (p.get_x() + p.get_width() / 2., p.get_height()),
                  ha = 'center', va = 'center',
                  xytext = (0, 10),
                  textcoords = 'offset points',
                  fontsize=12,
                  weight='bold')
plt.tight_layout()
plt.show()
```



In [36]:
```
profit_by_subcategory = data.groupby('Sub Category')['Profit'].sum().reset_index()
plt.figure(figsize=(14, 10))
```

```
df12=sns.barplot(x='Sub Category', y='Profit', data=profit_by_subcategory, palette='Set2')
plt.title('Profit by Sub Category', fontsize=22, weight='bold')
plt.xlabel('Sub Category', fontsize=20, weight='bold')
plt.ylabel('Profit', fontsize=20, weight='bold')
plt.xticks(fontsize=12, weight='bold',rotation=45)
plt.yticks(fontsize=12, weight='bold')
for p in df12.patches:
    df12.annotate(format(p.get_height(), '.0f'),
                  (p.get_x() + p.get_width() / 2., p.get_height()),
                  ha = 'center', va = 'center',
                  xytext = (0, 10),
                  textcoords = 'offset points',
                  fontsize=12,
                  weight='bold')
plt.tight_layout()
plt.show()
```



Top Five Subcategories Ranked by Profit

In [37]:
```
data.groupby('Sub Category')['Profit'].sum().reset_index().sort_values('Profit',ascending=False).head(5).reset_index()
```

Out[37]:

	index	Sub Category	Profit
0	7	Helmets	518475.0
1	14	Tires and Tubes	512124.0
2	9	Jerseys	300876.0
3	10	Mountain Bikes	144627.0
4	2	Bottles and Cages	129559.0

Bottom Five Subcategories Ranked by Profit

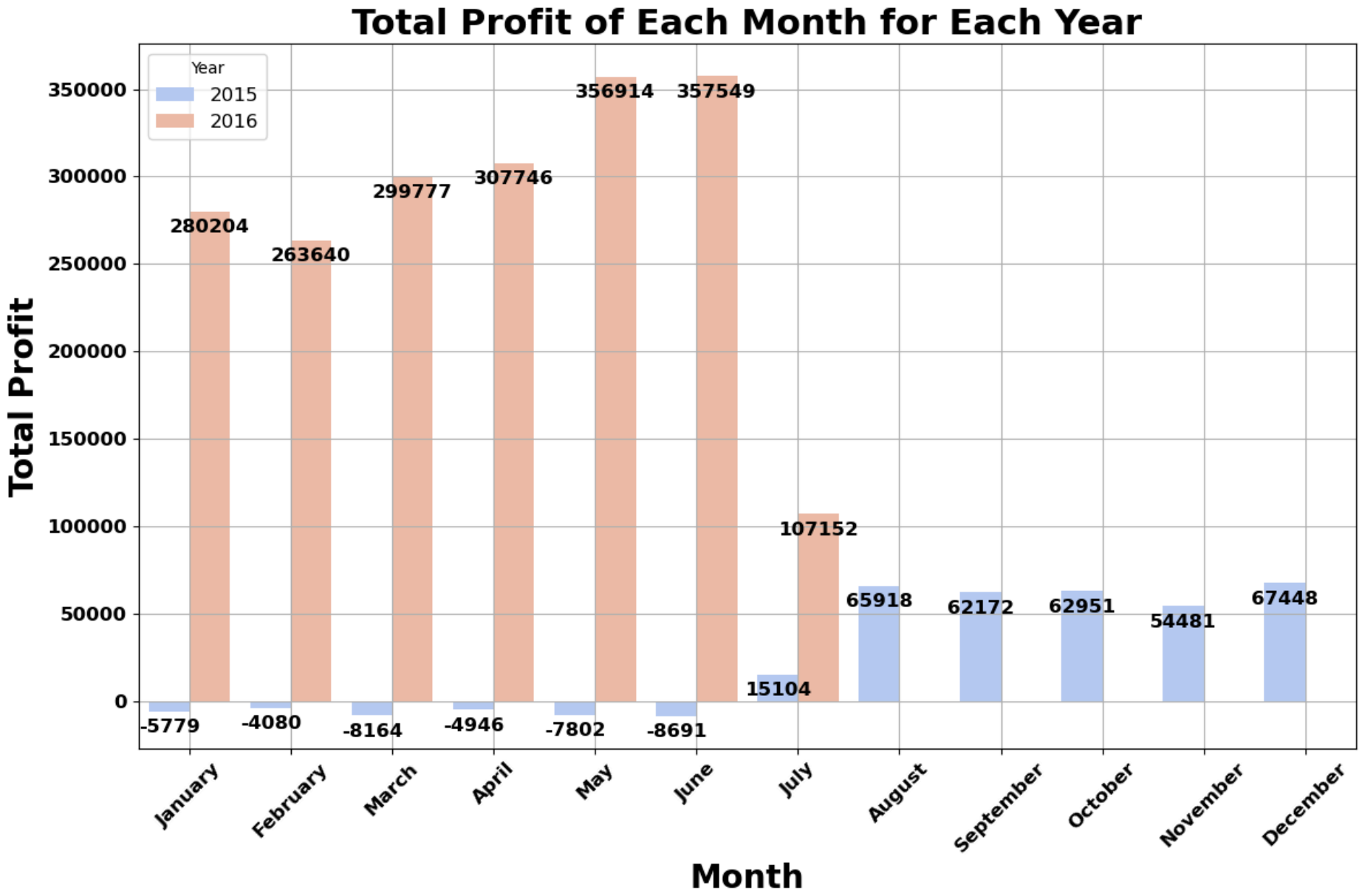
In [38]:
```
data.groupby('Sub Category')['Profit'].sum().reset_index().sort_values('Profit',ascending=False).tail(5).reset_index()
```

Out[38]:

	index	Sub Category	Profit
0	3	Caps	43610.0
1	0	Bike Racks	34894.0
2	1	Bike Stands	25301.0
3	4	Cleaners	14664.0
4	13	Socks	9541.0

In [39]:
```
monthly_profit = data.groupby(['Year', 'Month'])['Profit'].sum().reset_index()
months_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July',
                'August', 'September', 'October', 'November', 'December']

plt.figure(figsize=(12, 8))
df13=sns.barplot(x='Month', y='Profit', hue='Year', data=monthly_profit, order=months_order, palette='coolwarm')
plt.title('Total Profit of Each Month for Each Year',fontsize=22,
                weight='bold')
plt.xlabel('Month',fontsize=20,
                weight='bold')
plt.ylabel('Total Profit',fontsize=20,
                weight='bold')
plt.xticks(rotation=45,fontsize=12,
```

```
                weight='bold')
plt.yticks(fontsize=12,
                weight='bold')
plt.legend(title='Year', loc='upper left',fontsize=12)
plt.grid(True)
for p in df13.patches:
    df13.annotate(format(p.get_height(), '.0f'),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha = 'center', va = 'center',
                xytext = (0, -10),
                textcoords = 'offset points',
                fontsize=12,
                weight='bold')
plt.tight_layout()
plt.show()
```



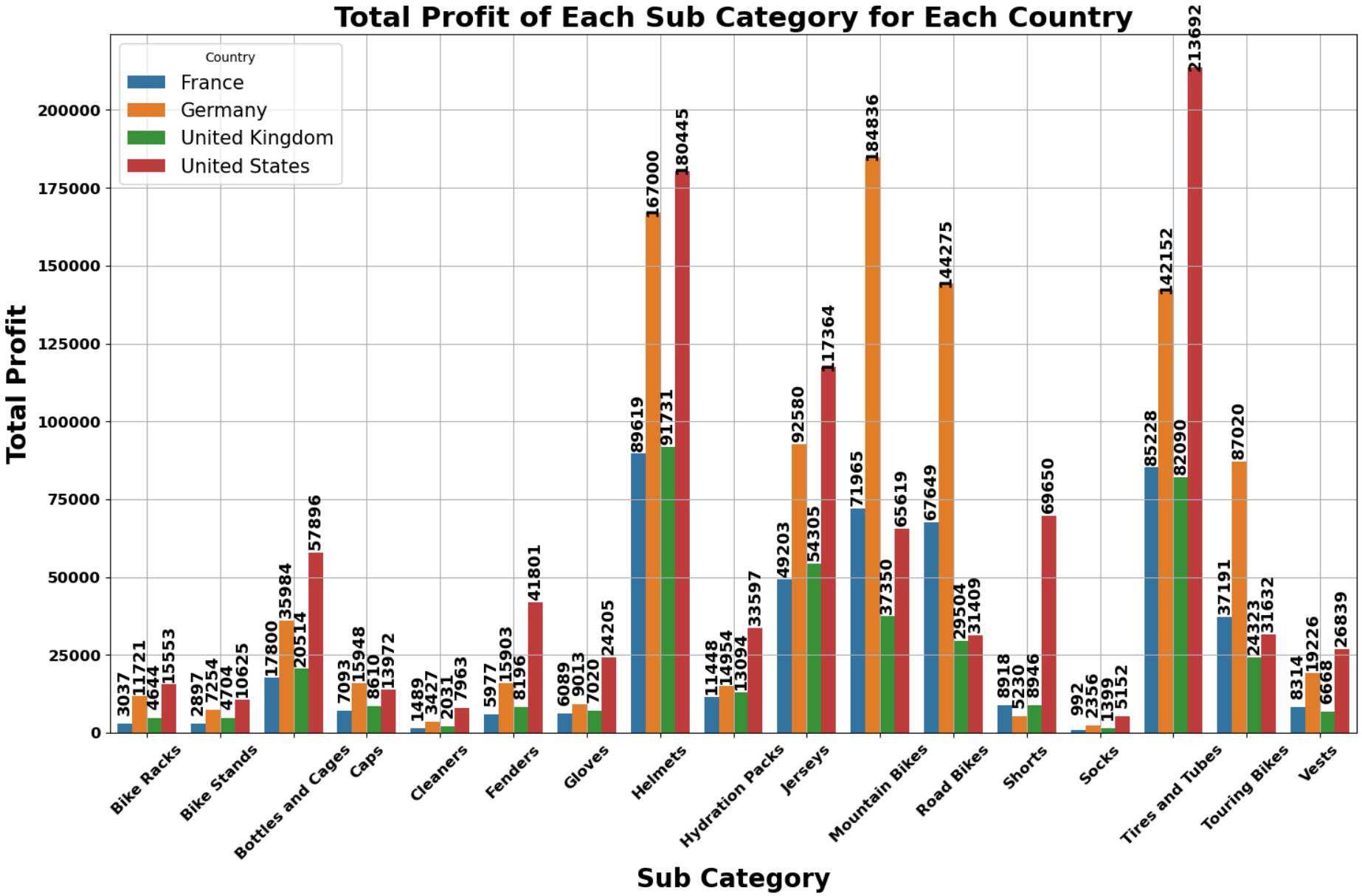
Total Loss

In [40]:
```
-1*(data[data['Profit']<0]['Profit'].sum())
```

Out[40]:
```
524737.0
```

In [41]:
```
m=data[data['Profit']>0].reset_index()
```

In [46]:
```
profit_by_subcategory_country=m.groupby(['Sub Category','Country'])['Profit'].sum().reset_index()
plt.figure(figsize=(15, 10))

df14=sns.barplot(x='Sub Category',y='Profit',hue='Country',data=profit_by_subcategory_country)
plt.title('Total Profit of Each Sub Category for Each Country',fontsize=22,
                weight='bold')
plt.xlabel('Sub Category',fontsize=20,
                weight='bold')
plt.ylabel('Total Profit',fontsize=20,
                weight='bold')
plt.xticks(rotation=45,fontsize=12,
                weight='bold')
plt.yticks(fontsize=12,
                weight='bold')
plt.legend(title='Country', loc='upper left',fontsize=15)
plt.grid(True)
for p in df14.patches:
    df14.annotate(format(p.get_height(), '.0f'),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha = 'center', va = 'center',
                xytext = (0, 25),
                textcoords = 'offset points',
                fontsize=13,
                weight='bold',rotation=90)
plt.tight_layout()
plt.show()
```



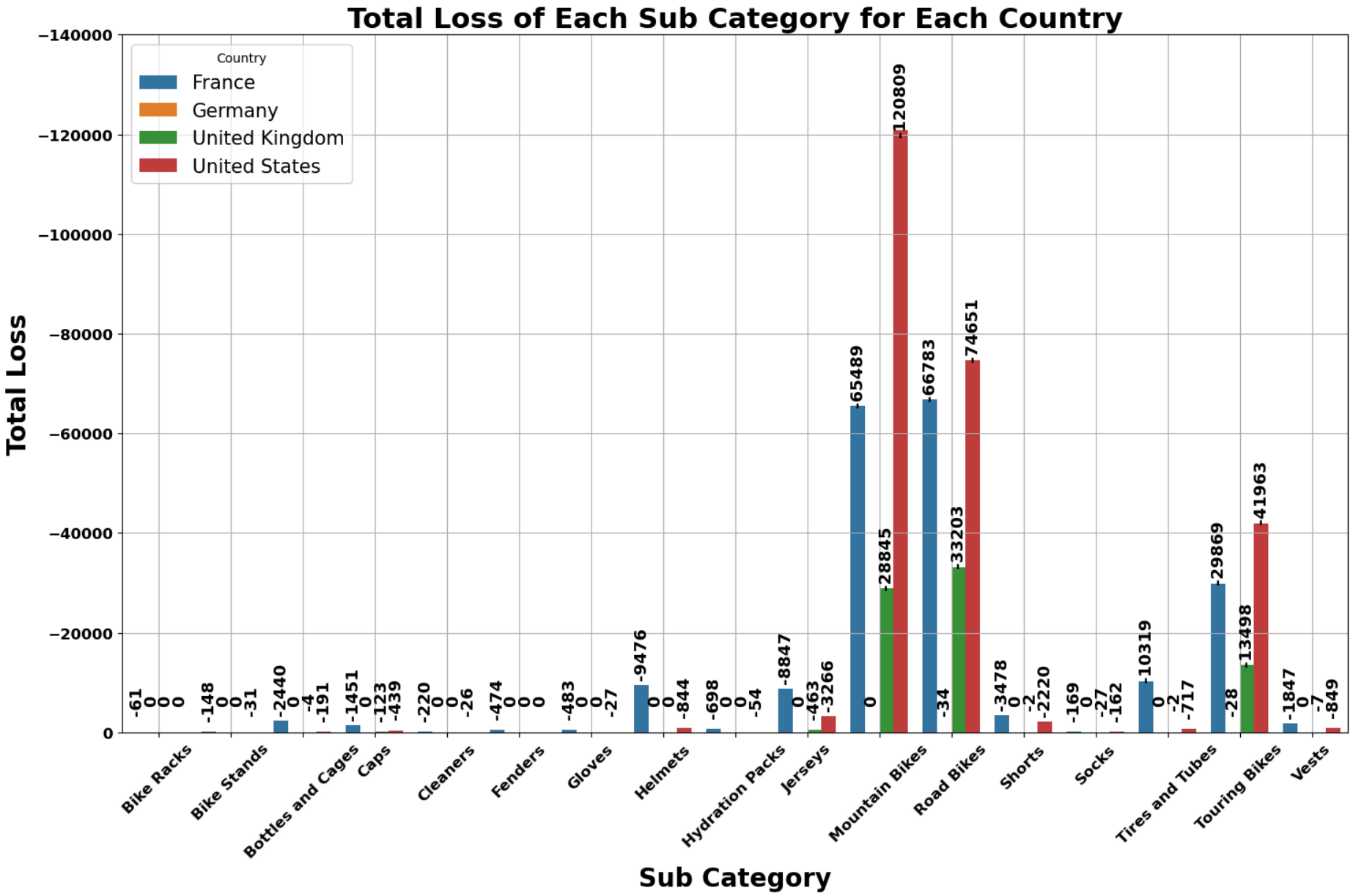
In [51]:
```
m.groupby('Country')['Profit'].sum().reset_index().sort_values('Profit',ascending=False)
```

Out[51]:

	Country	Profit
1	Germany	958879.0
3	United States	947414.0
0	France	474909.0
2	United Kingdom	405129.0

In [44]:
```
n=data[data['Profit']<0].reset_index()
```

In [45]:
```
loss_by_subcategory_country=n.groupby(['Sub Category','Country'])['Profit'].sum().reset_index()
plt.figure(figsize=(15, 10))

df15=sns.barplot(x='Sub Category',y='Profit',hue='Country',data=loss_by_subcategory_country)
plt.title('Total Loss of Each Sub Category for Each Country',fontsize=22,
                weight='bold')
plt.xlabel('Sub Category',fontsize=20,
                weight='bold')
plt.ylabel('Total Loss',fontsize=20,
                weight='bold')
plt.xticks(rotation=45,fontsize=12,
                weight='bold')
plt.yticks(fontsize=12,
                weight='bold')
plt.ylim(0,-140000)
plt.legend(title='Country', loc='upper left',fontsize=15)
plt.grid(True)
for p in df15.patches:
    df15.annotate(format(p.get_height(), '.0f'),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha = 'center', va = 'center',
                xytext = (0, 25),
                textcoords = 'offset points',
                fontsize=13,
                weight='bold',rotation=90)
plt.tight_layout()
plt.show()
```



In [52]:
```
n.groupby('Country')['Profit'].sum().reset_index().sort_values('Profit',ascending=False)
```

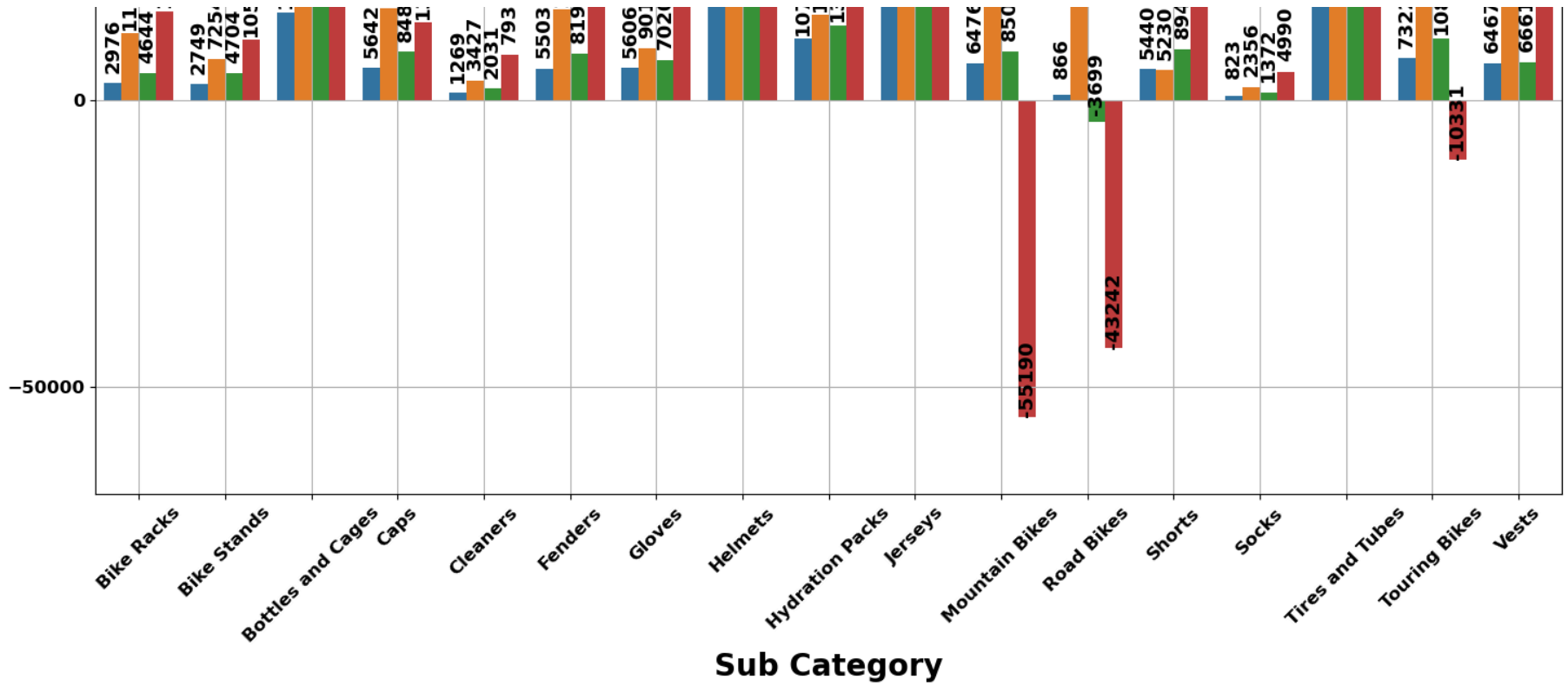
Out[52]:

	Country	Profit
1	Germany	-62.0
2	United Kingdom	-76174.0
0	France	-202252.0
3	United States	-246249.0

In [48]:

```
finalprofit_by_subcategory_country=data.groupby(['Sub Category','Country'])['Profit'].sum().reset_index()
plt.figure(figsize=(15, 18))

df16=sns.barplot(x='Sub Category',y='Profit',hue='Country',data=finalprofit_by_subcategory_country)
plt.title('Total Profit and Loss of Each Sub Category for Each Country',fontsize=22,
                weight='bold')
plt.xlabel('Sub Category',fontsize=20,
                weight='bold')
plt.ylabel('Total Profit and Loss',fontsize=20,
                weight='bold')
plt.xticks(rotation=45,fontsize=12,
                weight='bold')
plt.yticks(fontsize=12,
                weight='bold')
plt.legend(title='Country', loc='upper left',fontsize=15)
plt.grid(True)
for p in df16.patches:
    df16.annotate(format(p.get_height(), '.0f'),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha = 'center', va = 'center',
                xytext = (0, 25),
                textcoords = 'offset points',
                fontsize=13,
                weight='bold',rotation=90)
plt.tight_layout()
plt.show()
```

Total Profit and Loss of Each Sub Category for Each Country
Country
France
Germany
United Kingdom
United States
Total Profit and Loss
200000
150000
100000
50000
721
15553
594
15360
35984
20510
57705
15948
3533
7
15903
41801
24178
80143
167000
91731
179601
4954
3094
33543
40356
92580
53842
114098
184836
144241
67430
74909
142152
82088
212975
86992
825
19226
25990



In [53]:
```
data.groupby('Country')['Profit'].sum().reset_index().sort_values('Profit',ascending=False)
```

Out[53]:

	Country	Profit
1	Germany	958817.0
3	United States	701165.0
2	United Kingdom	328955.0
0	France	272657.0

Analysis Report: Profit Analysis by Product Category and Country

Insights:

- **Top Product Category: Accessories**:
  - Allocate resources towards promoting and expanding the accessories category.
- **Top Subcategory by Profit: Helmets**:
  - Consider targeted marketing campaigns or promotions to further boost helmet sales.
- **Runner-up Subcategory by Profit: Tires and Tubes**:
  - Explore opportunities to enhance sales of tires and tubes through strategic pricing or bundling strategies.
- **Bottom Subcategory by Profit: Socks**:
  - Evaluate the performance of socks subcategory and consider adjustments to pricing or marketing strategies.
- **Top Performing Country: Germany**:
- **Lowest Performing Country: France**:

Analysis Report: Profit Analysis and Strategy Recommendations

Insights:

- **Top Revenue Performers: Bikes, Mountain Bikes, United States**:
  - Despite being top in revenue, these categories are not performing well in terms of profit due to significant losses in the United States market.
- **Strategy Recommendation: Pricing Optimization**:
  - Adjust the pricing strategy for bikes, in the United States market to ensure profitability.
  - Conduct market research to determine the optimal price points that balance revenue and profit margins.
- **Competitive Analysis**:
  - Evaluate competitors' pricing strategies and positioning to identify opportunities for pricing adjustments while remaining competitive.
- **Promotional Strategies**:

- Implement targeted promotional campaigns or discounts to stimulate demand without compromising profitability.

In [ ]:

In [ ]: