About Dataset

Context

E-commerce has become a new channel to support businesses development. Through e-commerce, businesses can get access and establish a wider market presence by providing cheaper and more efficient distribution channels for their products or services. E-commerce has also changed the way people shop and consume products and services. Many people are turning to their computers or smart devices to order goods, which can easily be delivered to their homes.

Content

This is a sales transaction data set of UK-based e-commerce (online retail) for one year. This London-based shop has been selling gifts and homewares for adults and children through the website since 2007. Their customers come from all over the world and usually make direct purchases for themselves. There are also small businesses that buy in bulk and sell to other customers through retail outlet channels.

The data set contains 500K + rows and 8 columns. The following is the description of each column.

1.TransactionNo (categorical): a six-digit unique number that defines each transaction. The letter "C" in the code indicates a cancellation.

2.Date (numeric): the date when each transaction was generated.

3.ProductNo (categorical): a five or six-digit unique character used to identify a specific product.

4.Product (categorical): product/item name.

5.Price (numeric): the price of each product per unit in pound sterling (£).

6.Quantity (numeric): the quantity of each product per transaction. Negative values related to cancelled transactions.

7.CustomerNo (categorical): a five-digit unique number that defines each customer.

8.Country (categorical): name of the country where the customer resides.

Question

1.How was the sales trend over the months?

2.What are the most frequently purchased products?

3.How many products does the customer purchase in each transaction?

4.What are the most profitable segment customers?

5.Based on your findings, what strategy could you recommend to the business to gain more profit?

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
from matplotlib.ticker import FuncFormatter
%matplotlib inline
```

In [2]:
```
df = pd.read_csv('Sales Transaction v.4a.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	TransactionNo	Date	ProductNo	ProductName	Price	Quantity	CustomerNo	Country
0	581482	12/9/2019	22485	Set Of 2 Wooden Market Crates	21.47	12	17490.0	United Kingdom
1	581475	12/9/2019	22596	Christmas Star Wish List Chalkboard	10.65	36	13069.0	United Kingdom
2	581475	12/9/2019	23235	Storage Tin Vintage Leaf	11.53	12	13069.0	United Kingdom
3	581475	12/9/2019	23272	Tree T-Light Holder Willie Winkie	10.65	12	13069.0	United Kingdom
4	581475	12/9/2019	23239	Set Of 4 Knick Knack Tins Poppies	11.94	6	13069.0	United Kingdom

Preprocessing the Data

In [4]:
```
#checking the no of columns & rows of the data
df.shape
```

Out[4]:
```
(536350, 8)
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 536350 entries, 0 to 536349
Data columns (total 8 columns):
 #   Column         Non-Null Count   Dtype  
---  ------         --------------   -----  
 0   TransactionNo  536350 non-null  object 
 1   Date           536350 non-null  object 
 2   ProductNo      536350 non-null  object 
 3   ProductName    536350 non-null  object 
 4   Price          536350 non-null  float64
 5   Quantity       536350 non-null  int64  
 6   CustomerNo     536295 non-null  float64
 7   Country        536350 non-null  object 
dtypes: float64(2), int64(1), object(5)
memory usage: 32.7+ MB
```

In [6]:
```
#no of duplicates
df.duplicated().sum()
```

Out[6]:
```
5200
```

In [7]:
```
df[df.duplicated(subset=['TransactionNo','ProductName','ProductNo'])].head()
```

Out[7]:

	TransactionNo	Date	ProductNo	ProductName	Price	Quantity	CustomerNo	Country
905	581493	12/9/2019	79190B	Retro Plastic Polka Tray	7.24	12	12423.0	Belgium
985	581497	12/9/2019	21481	Fawn Blue Hot Water Bottle	7.24	1	17497.0	United Kingdom
1000	581497	12/9/2019	22356	Charlotte Bag Pink Polkadot	7.24	1	17497.0	United Kingdom
1033	581497	12/9/2019	23206	Lunch Bag Apple Design	6.04	1	17497.0	United Kingdom
1295	581502	12/9/2019	22153	Angel Decoration Stars On Dress	7.24	1	15910.0	United Kingdom

In [8]:
```
#removing duplicates
non_dup = df[~df.duplicated()]
```

In [9]:
```
non_dup.shape
```

Out[9]:
```
(531150, 8)
```

In [10]:
```
#converting the date object to datetime format for easy analysis
df['Date']=pd.to_datetime(df['Date'])
```

In [11]:
```
df['Month'] = df['Date'].dt.month
```

In [12]:
```
df.head()
```

Out[12]:

	TransactionNo	Date	ProductNo	ProductName	Price	Quantity	CustomerNo	Country	Month
0	581482	2019-12-09	22485	Set Of 2 Wooden Market Crates	21.47	12	17490.0	United Kingdom	12
1	581475	2019-12-09	22596	Christmas Star Wish List Chalkboard	10.65	36	13069.0	United Kingdom	12
2	581475	2019-12-09	23235	Storage Tin Vintage Leaf	11.53	12	13069.0	United Kingdom	12
3	581475	2019-12-09	23272	Tree T-Light Holder Willie Winkie	10.65	12	13069.0	United Kingdom	12
4	581475	2019-12-09	23239	Set Of 4 Knick Knack Tins Poppies	11.94	6	13069.0	United Kingdom	12

In [13]:
```
#Extracting year column from date column
df['Year'] = df['Date'].dt.year
```

In [14]:
```
df['Year'].nunique()
```

Out[14]:
```
2
```

In [15]:
```
df['Year'].value_counts()
```

Out[15]:
```
2019    494256
2018     42094
Name: Year, dtype: int64
```

In [16]:
```
df.tail()
```

Out[16]:

	TransactionNo	Date	ProductNo	ProductName	Price	Quantity	CustomerNo	Country	Mo
536345	C536548	2018-12-01	22168	Organiser Wood Antique White	18.96	-2	12472.0	Germany	
536346	C536548	2018-12-01	21218	Red Spotty Biscuit Tin	14.09	-3	12472.0	Germany	
536347	C536548	2018-12-01	20957	Porcelain Hanging Bell Small	11.74	-1	12472.0	Germany	
536348	C536548	2018-12-01	22580	Advent Calendar Gingham Sack	16.35	-4	12472.0	Germany	
536349	C536548	2018-12-01	22767	Triple Photo Frame Cornice	20.45	-2	12472.0	Germany	

In [17]:
```
#Checking for null values
df.isnull().sum()
```

Out[17]:
```
TransactionNo     0
Date              0
ProductNo         0
ProductName       0
Price             0
Quantity          0
CustomerNo       55
Country           0
Month             0
Year              0
dtype: int64
```

In [18]:
```
#dropping duplicates
df.dropna(inplace=True)
```

In [19]:
```
df.isnull().sum()
```

Out[19]:
```
TransactionNo    0
Date             0
ProductNo        0
ProductName      0
Price            0
Quantity         0
CustomerNo       0
Country          0
Month            0
Year             0
dtype: int64
```

In [20]:
```
#rename columns for better understanding
df=df.rename(columns={'TransactionNo':'Transaction_id','ProductNo':'Product_id','Custo
                      'Country':'Customer_Country','Date':'Transaction_Date','Year':'Tran
                      'Month':'Transaction_Month'})
```

In [21]:
```
df.head()
```

Out[21]:

	Transaction_id	Transaction_Date	Product_id	ProductName	Price	Quantity	Customer_id	Custome
0	581482	2019-12-09	22485	Set Of 2 Wooden Market Crates	21.47	12	17490.0	Unite
1	581475	2019-12-09	22596	Christmas Star Wish List Chalkboard	10.65	36	13069.0	Unite
2	581475	2019-12-09	23235	Storage Tin Vintage Leaf	11.53	12	13069.0	Unite
3	581475	2019-12-09	23272	Tree T-Light Holder Willie Winkie	10.65	12	13069.0	Unite
4	581475	2019-12-09	23239	Set Of 4 Knick Knack Tins Poppies	11.94	6	13069.0	Unite

In [22]:
```
df['Customer_Country'].unique()
```

Out[22]:
```
array(['United Kingdom', 'Norway', 'Belgium', 'Germany', 'France',
       'Austria', 'Netherlands', 'EIRE', 'USA', 'Channel Islands',
       'Iceland', 'Portugal', 'Spain', 'Finland', 'Italy', 'Greece',
       'Japan', 'Sweden', 'Denmark', 'Cyprus', 'Malta', 'Switzerland',
       'Australia', 'Czech Republic', 'Poland', 'Hong Kong', 'Singapore',
       'RSA', 'Israel', 'Unspecified', 'United Arab Emirates', 'Canada',
       'European Community', 'Bahrain', 'Brazil', 'Saudi Arabia',
       'Lebanon', 'Lithuania'], dtype=object)
```

In [23]:
```
df['Quantity'].min()
```

Out[23]:
```
-80995
```

In [24]:
```
#Dropping negative values
df=df[~df['Quantity']<0]
```

In [25]:
```
df['Quantity'].min()
```

Out[25]:
```
1
```

In [26]:
```
#Creating Sales Total column based on units of products bought and price
df['Sales_Total'] = df['Quantity'] * df['Price']
```

In [27]:
```
df.head()
```

Out[27]:

	Transaction_id	Transaction_Date	Product_id	ProductName	Price	Quantity	Customer_id	Custome
0	581482	2019-12-09	22485	Set Of 2 Wooden Market Crates	21.47	12	17490.0	Unite
1	581475	2019-12-09	22596	Christmas Star Wish List Chalkboard	10.65	36	13069.0	Unite
2	581475	2019-12-09	23235	Storage Tin Vintage Leaf	11.53	12	13069.0	Unite
3	581475	2019-12-09	23272	Tree T-Light Holder Willie Winkie	10.65	12	13069.0	Unite
4	581475	2019-12-09	23239	Set Of 4 Knick Knack Tins Poppies	11.94	6	13069.0	Unite

In [28]:
```
df.describe()
```

Out[28]:

	Price	Quantity	Customer_id	Transaction_Month	Transaction_Year	Sales_Tot
count	527764.000000	527764.000000	527764.000000	527764.000000	527764.000000	5.277640e+(
mean	12.629640	10.594679	15231.626733	7.562318	2018.921524	1.193069e+(
std	7.933224	156.786795	1716.522182	3.509039	0.268920	1.851192e+(
min	5.130000	1.000000	12004.000000	1.000000	2018.000000	5.130000e+(
25%	10.990000	1.000000	13813.000000	5.000000	2019.000000	1.717000e+(
50%	11.940000	3.000000	15159.000000	8.000000	2019.000000	4.383000e+(
75%	14.090000	11.000000	16729.000000	11.000000	2019.000000	1.194000e+(
max	660.620000	80995.000000	18287.000000	12.000000	2019.000000	1.002718e+(

Exploratory Data Analysis

The Most Expensive Product in the Store

In [29]:
```
most_expensive_product_name = df['ProductName'][df['Price'].idxmax()]
most_expensive_price = df['Price'].max()

print("The most expensive product is '{}' with a price of ${:.2f}".format(most_expensi
```

```
The most expensive product is 'Vintage Red Kitchen Cabinet' with a price of $660.62
```

The Chepeast Product in the store

In [30]:
```
The_cheapest_product_name = df['ProductName'][df['Price'].idxmin()]
The_chepeast_price = df['Price'].min()
print("The Chepeast product is '{}' with a price of ${:.2f}".format(The_cheapest_produ
```

```
The Chepeast product is 'Lunch Bag Dolly Girl Design' with a price of $5.13
```

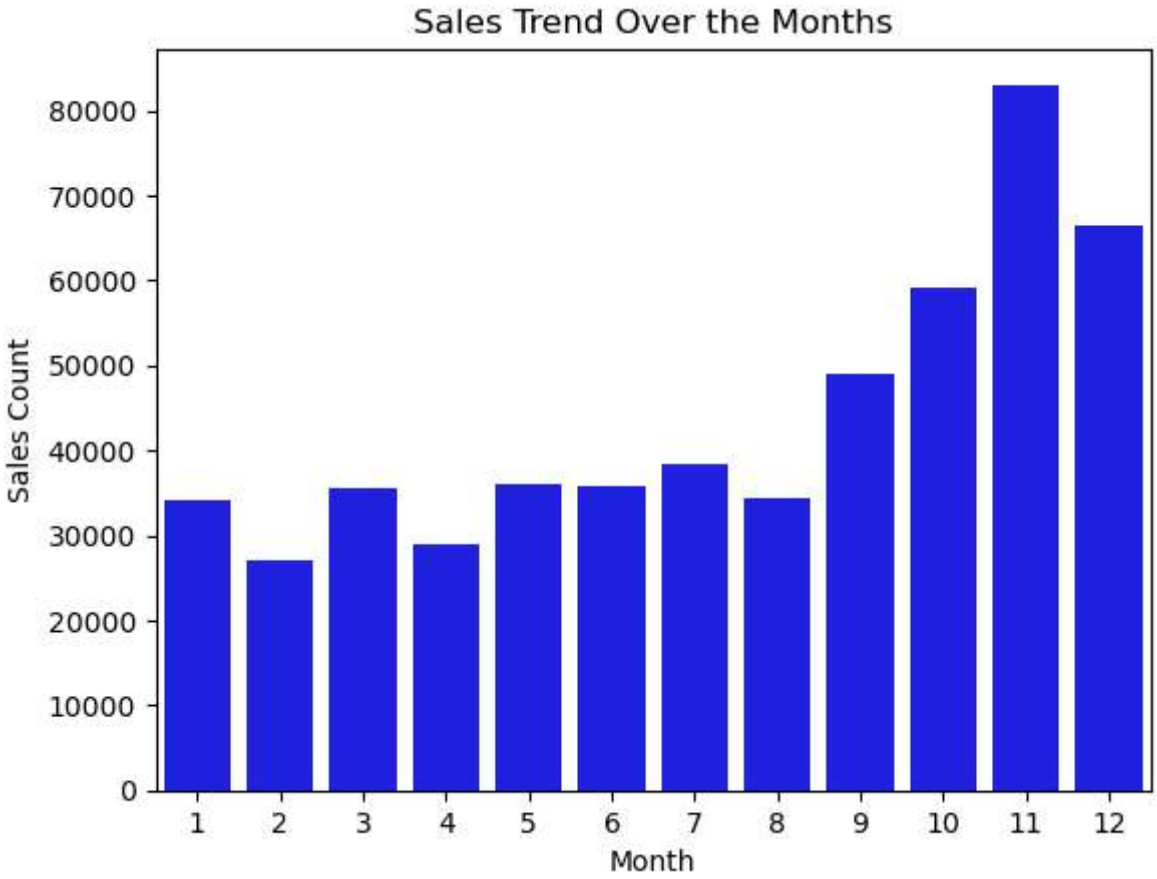
In [31]:
```
df.head()
```

Out[31]:

	Transaction_id	Transaction_Date	Product_id	ProductName	Price	Quantity	Customer_id	Custome
0	581482	2019-12-09	22485	Set Of 2 Wooden Market Crates	21.47	12	17490.0	Unite
1	581475	2019-12-09	22596	Christmas Star Wish List Chalkboard	10.65	36	13069.0	Unite
2	581475	2019-12-09	23235	Storage Tin Vintage Leaf	11.53	12	13069.0	Unite
3	581475	2019-12-09	23272	Tree T-Light Holder Willie Winkie	10.65	12	13069.0	Unite
4	581475	2019-12-09	23239	Set Of 4 Knick Knack Tins Poppies	11.94	6	13069.0	Unite

1.How was the sales trend over the months?

In [32]:
```
sns.countplot(x=df['Transaction_Month'],color='blue')
plt.title('Sales Trend Over the Months')
plt.ylabel('Sales Count')
plt.xlabel('Month')
plt.show()
```



2.What are the most frequently purchased products?

In [33]:
```
#selecting the products based on quantitiy and sorting them accordingly
top_purchased_products = df.groupby(['ProductName'])['Quantity'].sum().reset_index()
```

In [34]:
```
#then sorting them in descending order for the top 10 most frequently purchased
top_purchased_products = top_purchased_products.sort_values(by=['Quantity'],ascending=
```

Top 10 Highest Frequently Purchased Products

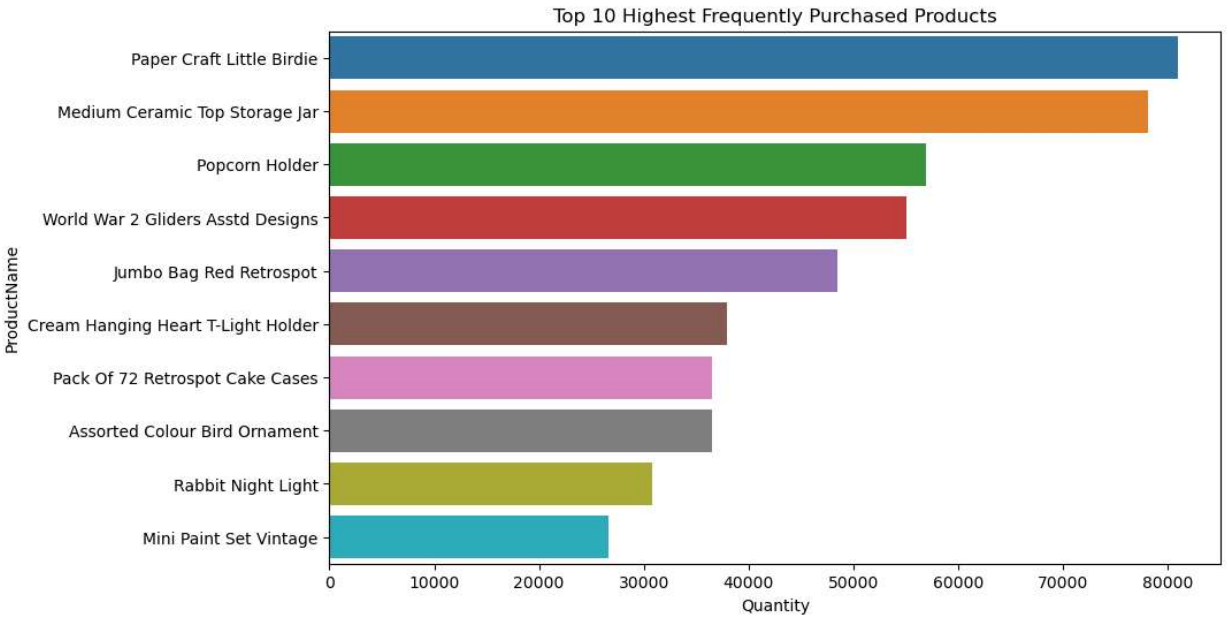
In [35]:
```
top_purchased_products.head(10)
```

Out[35]:

	ProductName	Quantity
2203	Paper Craft Little Birdie	80995
1887	Medium Ceramic Top Storage Jar	78033
2481	Popcorn Holder	56921
3670	World War 2 Gliders Asstd Designs	55047
1672	Jumbo Bag Red Retrospot	48478
825	Cream Hanging Heart T-Light Holder	37956
2157	Pack Of 72 Retrospot Cake Cases	36515
203	Assorted Colour Bird Ornament	36493
2540	Rabbit Night Light	30788
1938	Mini Paint Set Vintage	26633

A visual representation of Top 10 Highest Frequently Purchased Products

In [36]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x='Quantity', y='ProductName', data=top_purchased_products.head(10), palet
plt.xlabel('Quantity')
plt.ylabel('ProductName')
plt.title('Top 10 Highest Frequently Purchased Products')
plt.show()
```



From the result above, the most frequently purchased product is 'Paper Craft Little Birdie' with a Quantity of 80995

In [37]:
```
words=df['ProductName']
words = " ".join(df['ProductName'])
```

In [38]:
```
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(words)

# Display the word cloud using matplotlib
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```

3.How many products does the customer purchase in each transaction?

In [39]:
```
avg_products_per_trans = round(df['Quantity'].mean())
print(f"The average Number of Products Purchased Per Transaction : {avg_products_per_t
```

```
The average Number of Products Purchased Per Transaction : 11
```

In [40]:
```
df.groupby('Customer_id')['Sales_Total'].sum()
```

Out[40]:
```
Customer_id
12004.0     1509.60
12006.0       24.76
12008.0     5689.57
12013.0       69.96
12024.0      149.52
             ...
18280.0      623.26
18281.0      576.58
18282.0     1044.86
18283.0    12114.61
18287.0    18139.56
Name: Sales_Total, Length: 4718, dtype: float64
```

In [41]:
```
df.head()
```

Out[41]:

	Transaction_id	Transaction_Date	Product_id	ProductName	Price	Quantity	Customer_id	Custome
0	581482	2019-12-09	22485	Set Of 2 Wooden Market Crates	21.47	12	17490.0	Unite
1	581475	2019-12-09	22596	Christmas Star Wish List Chalkboard	10.65	36	13069.0	Unite
2	581475	2019-12-09	23235	Storage Tin Vintage Leaf	11.53	12	13069.0	Unite
3	581475	2019-12-09	23272	Tree T-Light Holder Willie Winkie	10.65	12	13069.0	Unite
4	581475	2019-12-09	23239	Set Of 4 Knick Knack Tins Poppies	11.94	6	13069.0	Unite

4 What are the most profitable segment customers?

In [42]:
```
profit_per_segment=df.groupby('Customer_Country')['Sales_Total'].sum().reset_index()
profit_per_segment = profit_per_segment.sort_values(by='Sales_Total',ascending=False).
```

In [43]:
```
profit_per_segment.head()
```

Out[43]:

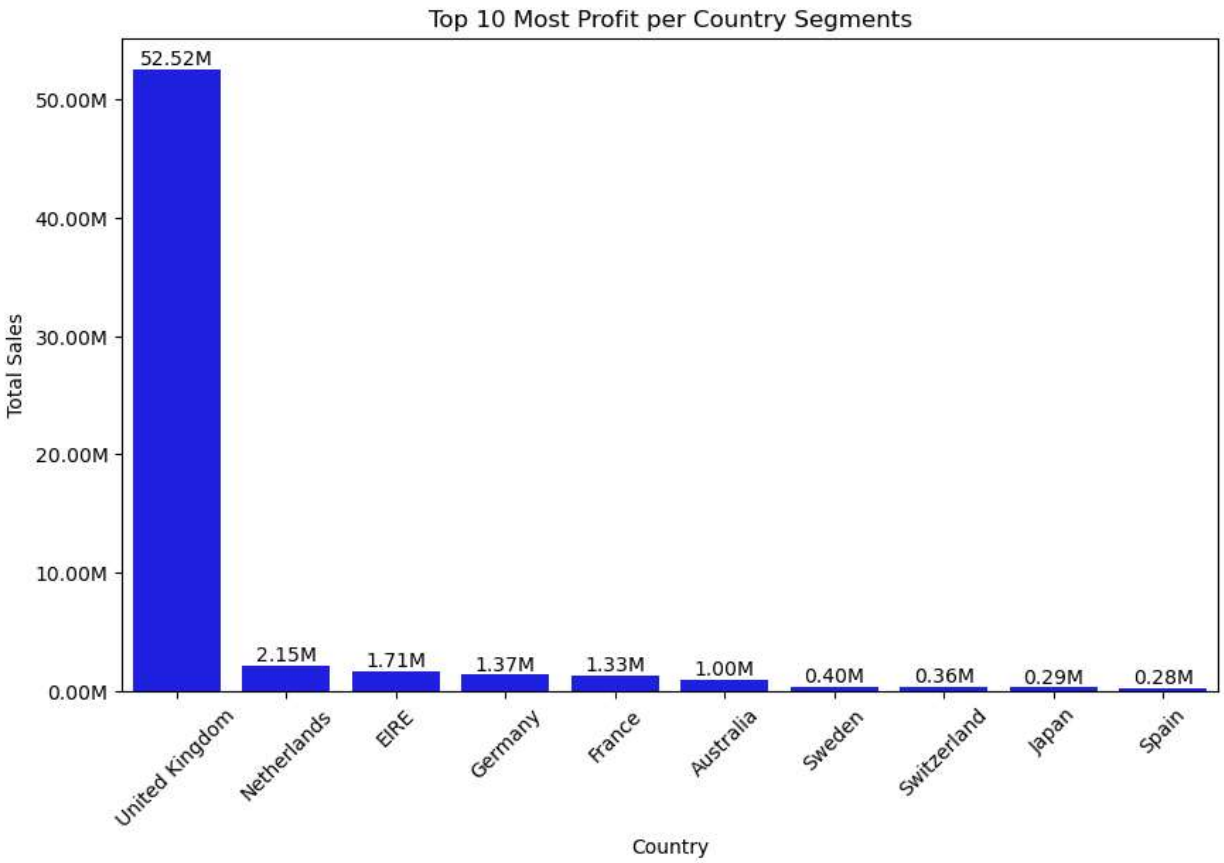
	Customer_Country	Sales_Total
36	United Kingdom	52524576.47
24	Netherlands	2151553.59
10	EIRE	1713410.95
14	Germany	1371543.27
13	France	1330652.89

In [44]:
```
def format_millions(value, _):
    return f'{value/1e6:.2f}M'
```

A visual Representation of top 10 most Profitable segments

In [45]:
```
# Visualization
plt.figure(figsize=(10, 6))
ax = sns.barplot(x='Customer_Country', y='Sales_Total', data=profit_per_segment, color
ax.yaxis.set_major_formatter(FuncFormatter(format_millions))
for p in ax.patches:
    height = p.get_height()
```

```
    ax.annotate(format_millions(height, None), (p.get_x() + p.get_width() / 2., height
plt.xlabel('Country')
plt.ylabel('Total Sales')
plt.title('Top 10 Most Profit per Country Segments')
plt.xticks(rotation=45)
plt.show()
```



United Kingdom stands out as the top perfoming country in terms of sales with a Total sales of Over 50M

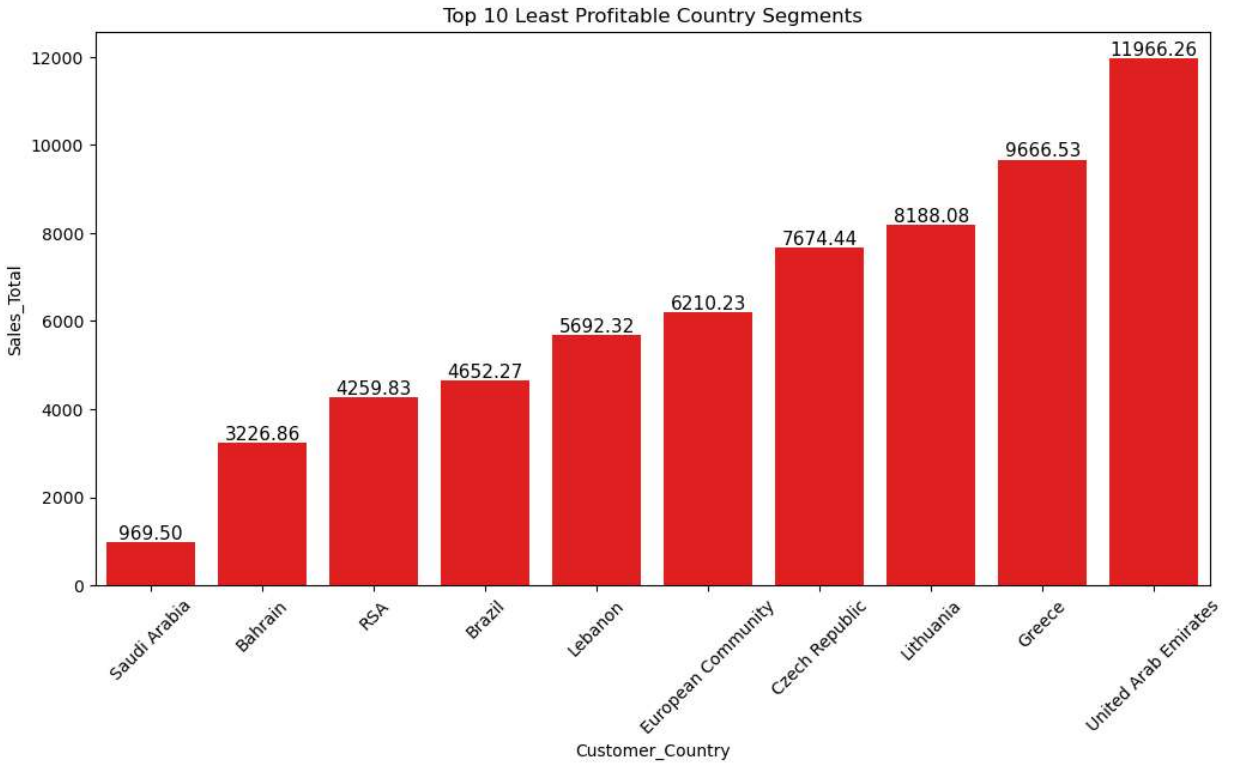
Top 10 Least Performing Segments

In [46]:
```
#Top 10 Least Profitable Segments
Bottom_10_selling_Regions = df.groupby('Customer_Country')['Sales_Total'].sum().reset_
Bottom_10_selling_Regions = Bottom_10_selling_Regions.sort_values(by='Sales_Total').he
Bottom_10_selling_Regions.head(10)
```

Out[46]:

	Customer_Country	Sales_Total
29	Saudi Arabia	969.50
2	Bahrain	3226.86
28	RSA	4259.83
4	Brazil	4652.27
21	Lebanon	5692.32
11	European Community	6210.23
8	Czech Republic	7674.44
22	Lithuania	8188.08
15	Greece	9666.53
35	United Arab Emirates	11966.26

In [47]:
```
# Visualization
plt.figure(figsize=(12, 6))
ax = sns.barplot(x='Customer_Country', y='Sales_Total', data=Bottom_10_selling_Regions
plt.xlabel('Customer_Country')
plt.ylabel('Sales_Total')
plt.title('Top 10 Least Profitable Country Segments')
plt.xticks(rotation=45)
for p in ax.patches:
    height = p.get_height()
    ax.annotate(f'{height:.2f}', (p.get_x() + p.get_width() / 2., height), ha='center'
plt.show()
```



From the result,Saudi Arabia is the least performing Country with sales of 969.50

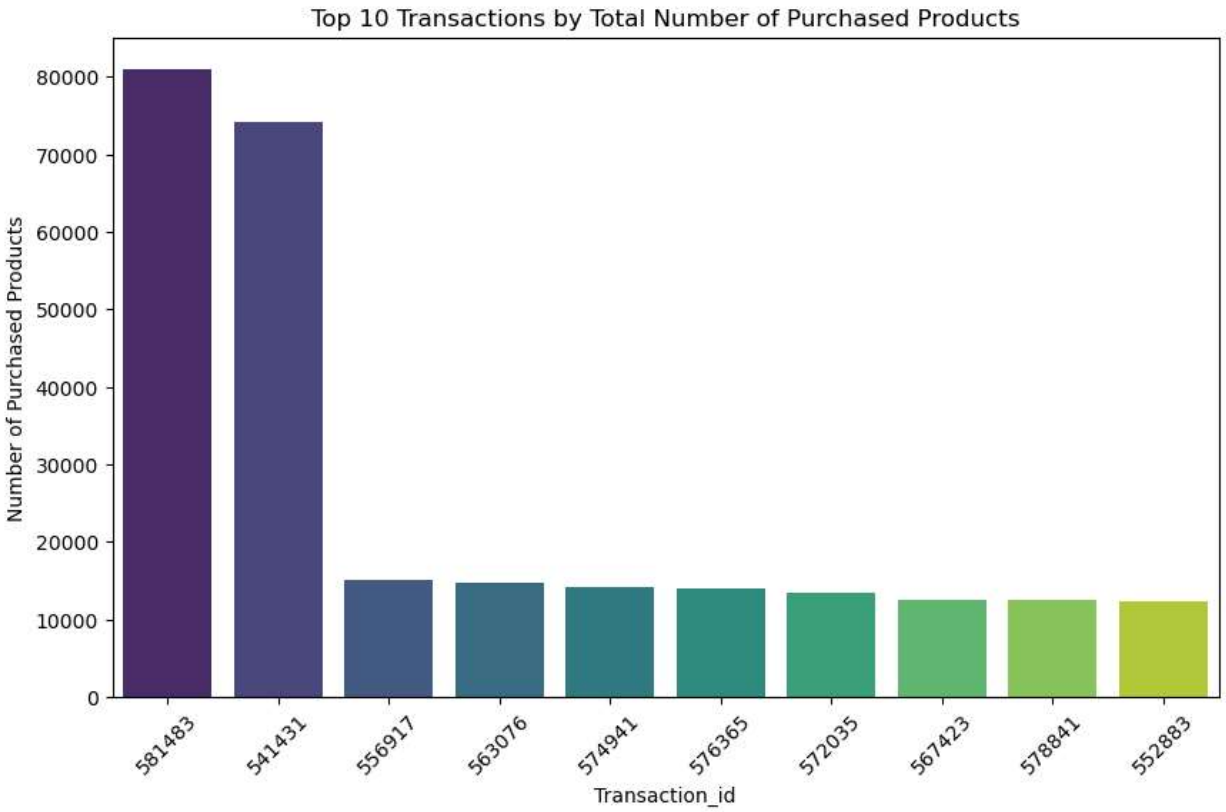
In [48]:
```
#Transaction ids with the highest purchased Products
total_purchase_Per_transation = df.groupby('Transaction_id')['Quantity'].sum().reset_i
```

In [49]:
```
top_10=total_purchase_Per_transation.sort_values(by='Quantity',ascending=False).head(1
top_10.head()
```

Out[49]:

	Transaction_id	Quantity
19753	581483	80995
2108	541431	74215
8847	556917	15049
11553	563076	14730
16767	574941	14149

In [50]:
```
# Visualization
plt.figure(figsize=(10, 6))
sns.barplot(data=top_10, x='Transaction_id', y='Quantity', palette='viridis')
plt.xlabel('Transaction_id')
plt.ylabel('Number of Purchased Products')
plt.title('Top 10 Transactions by Total Number of Purchased Products')
plt.xticks(rotation=45)
plt.show()
```



Customers with the highest number of transactions

In [51]:
```
Top_10_customers = df.groupby('Customer_id')['Transaction_id'].nunique().reset_index()
```

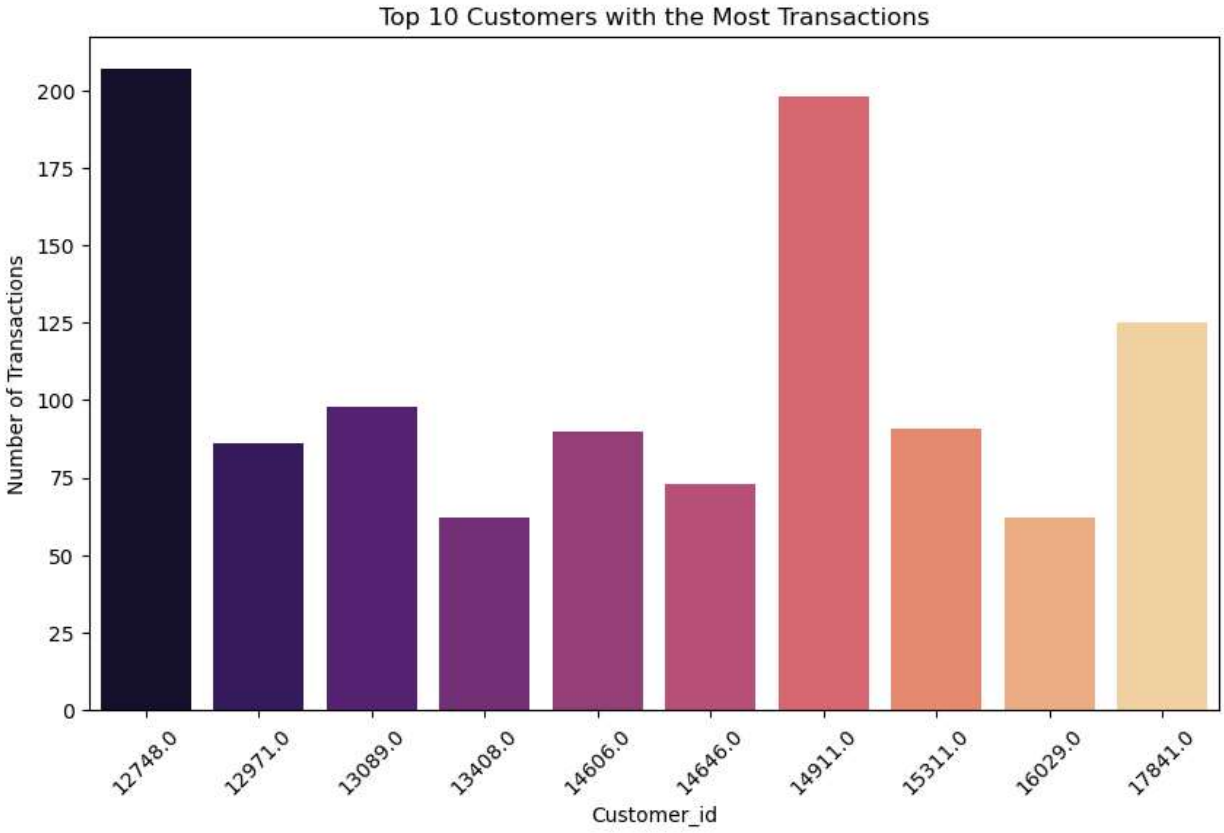
In [52]:
```
top_customers=Top_10_customers.sort_values(by='Transaction_id',ascending=False).head(1
```

In [53]:
```
top_customers.head(10)
```

Out[53]:

	Customer_id	Transaction_id
408	12748.0	207
2085	14911.0	198
4384	17841.0	125
667	13089.0	98
2406	15311.0	91
1851	14606.0	90
581	12971.0	86
1880	14646.0	73
2978	16029.0	62
918	13408.0	62

In [54]:
```
# Visualization
plt.figure(figsize=(10, 6))
sns.barplot(x='Customer_id', y='Transaction_id', data=top_customers, palette='magma')
plt.title('Top 10 Customers with the Most Transactions')
plt.xlabel('Customer_id')
plt.ylabel('Number of Transactions')
plt.xticks(rotation=45)
plt.show()
```



From the result above, customer with the highest number of Transactions is identified with id 12748.0 with a total of 207 transactions

5.Based on your findings, what strategy could you recommend to the business to gain more profit?

1.Data Driven Decisions : Utilise Data analytics in order to understand trends and customer preferences

2.Optimized sales Timing: Concetrate marketing efforts during the months of increased sales Do promotions on months with the least sales

3.Explore new Markets especially in less profitable regions

4.Discount and Promotions :To Stragically simulate demand, introduce sales discounts and promotions

5.Product Diversification : To encourage additional purchase,introduce related products

6.Cost control : Streamline Operations and negotiate supplier deals to reduce costs

7.Stock the unavailable products that make customers cancel their orders

In [ ]: