# supply chain analysis 3rd attempt 26-2-2025

February 26, 2025

[1]:
```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
# Load the dataset
df = pd.read_excel('supply_chain_data.xls')
# Display the first 5 rows of the dataset
print(df.head())
```

```
  Product type   SKU      Price  Availability  Number of products sold  \
0     haircare  SKU0  69.808006            55                      802
1     skincare  SKU1  14.843523            95                      736
2     haircare  SKU2  11.319683            34                        8
3     skincare  SKU3  61.163343            68                       83
4     skincare  SKU4   4.805496            26                      871

   Revenue generated Customer demographics  Stock levels  Lead times  \
0        8661.996792            Non-binary            58           7
1        7460.900065                Female            53          30
2        9577.749626               Unknown             1          10
3        7766.836426            Non-binary            23          13
4        2686.505152            Non-binary             5           3

   Order quantities  …  Location Lead time  Production volumes  \
0                96  …    Mumbai        29                 215
1                37  …    Mumbai        23                 517
2                88  …    Mumbai        12                 971
3                59  …   Kolkata        24                 937
4                56  …     Delhi         5                 414

  Manufacturing lead time Manufacturing costs  Inspection results  \
0                      29           46.279879             Pending
1                      30           33.616769             Pending
2                      27           30.688019             Pending
3                      18           35.624741                Fail
4                       3           92.065161                Fail
```

```
   Defect rates  Transportation modes   Routes       Costs
0      0.226410                  Road  Route B  187.752075
1      4.854068                  Road  Route B  503.065579
2      4.580593                   Air  Route C  141.920282
3      4.746649                  Rail  Route A  254.776159
4      3.145580                   Air  Route A  923.440632

[5 rows x 24 columns]
```

[3]:
```
# Check the shape of the dataset (rows, columns)
print("Shape of the dataset:", df.shape)
```

```
Shape of the dataset: (100, 24)
```

[4]:
```
# Check for missing values
print("Missing values in each column:")
print(df.isnull().sum())
```

```
Missing values in each column:
Product type                0
SKU                         0
Price                       0
Availability                0
Number of products sold     0
Revenue generated           0
Customer demographics       0
Stock levels                0
Lead times                  0
Order quantities            0
Shipping times              0
Shipping carriers           0
Shipping costs              0
Supplier name               0
Location                    0
Lead time                   0
Production volumes          0
Manufacturing lead time     0
Manufacturing costs         0
Inspection results          0
Defect rates                0
Transportation modes        0
Routes                      0
Costs                       0
dtype: int64
```

[5]:
```
# Get basic statistics of numerical columns
print("Basic statistics:")
```

```
print(df.describe())
```

```
Basic statistics:
            Price  Availability  Number of products sold  Revenue generated  \
count  100.000000    100.000000               100.000000         100.000000
mean    49.462461     48.400000               460.990000        5776.048187
std     31.168193     30.743317               303.780074        2732.841744
min      1.699976      1.000000                 8.000000        1061.618523
25%     19.597823     22.750000               184.250000        2812.847151
50%     51.239830     43.500000               392.500000        6006.352023
75%     77.198228     75.000000               704.250000        8253.976920
max     99.171329    100.000000               996.000000        9866.465458

       Stock levels  Lead times  Order quantities  Shipping times  \
count    100.000000  100.000000        100.000000      100.000000
mean      47.770000   15.960000         49.220000        5.750000
std       31.369372    8.785801         26.784429        2.724283
min        0.000000    1.000000          1.000000        1.000000
25%       16.750000    8.000000         26.000000        3.750000
50%       47.500000   17.000000         52.000000        6.000000
75%       73.000000   24.000000         71.250000        8.000000
max      100.000000   30.000000         96.000000       10.000000

       Shipping costs   Lead time  Production volumes  \
count      100.000000  100.000000          100.000000
mean         5.548149   17.080000          567.840000
std          2.651376    8.846251          263.046861
min          1.013487    1.000000          104.000000
25%          3.540248   10.000000          352.000000
50%          5.320534   18.000000          568.500000
75%          7.601695   25.000000          797.000000
max          9.929816   30.000000          985.000000

       Manufacturing lead time  Manufacturing costs  Defect rates       Costs
count                100.00000           100.000000    100.000000  100.000000
mean                  14.77000            47.266693      2.277158  529.245782
std                    8.91243            28.982841      1.461366  258.301696
min                    1.00000             1.085069      0.018608  103.916248
25%                    7.00000            22.983299      1.009650  318.778455
50%                   14.00000            45.905622      2.141863  520.430444
75%                   23.00000            68.621026      3.563995  763.078231
max                   30.00000            99.466109      4.939255  997.413450
```

[6]:
```
# Check data types of each column
print("Data types:")
print(df.dtypes)
```

```
Data types:
```

```
Product type                 object
SKU                          object
Price                       float64
Availability                  int64
Number of products sold       int64
Revenue generated           float64
Customer demographics        object
Stock levels                  int64
Lead times                    int64
Order quantities              int64
Shipping times                int64
Shipping carriers            object
Shipping costs              float64
Supplier name                object
Location                     object
Lead time                     int64
Production volumes            int64
Manufacturing lead time       int64
Manufacturing costs         float64
Inspection results           object
Defect rates                float64
Transportation modes         object
Routes                       object
Costs                       float64
dtype: object
```

[7]:
```
df = df.dropna()
```

[8]:
```
# Check for duplicates
print("Number of duplicate rows:", df.duplicated().sum())
```

```
Number of duplicate rows: 0
```

[9]:
```
df['Product type'] = df['Product type'].astype(str)
df['SKU'] = df['SKU'].astype(str)
df['Price'] = df['Price'].astype(float)
df['Availability'] = df['Availability']. astype(int)
df['Number of products sold'] = df['Number of products sold'].astype(int)
df['Revenue generated'] = df['Revenue generated'].astype(float)
df['Customer demographics'] = df['Customer demographics'].astype(str)
df['Stock levels'] = df['Stock levels'].astype(int)
df['Lead times'] = df['Lead times'].astype(int)
df['Order quantities'] = df['Order quantities'].astype(int)
df['Shipping times'] = df['Shipping times'].astype(int)
df['Shipping carriers'] = df['Shipping carriers'].astype(str)
df['Shipping costs'] = df['Shipping costs'].astype(float)
df['Supplier name'] = df['Supplier name'].astype(str)
df['Location'] = df['Location'].astype(str)
```

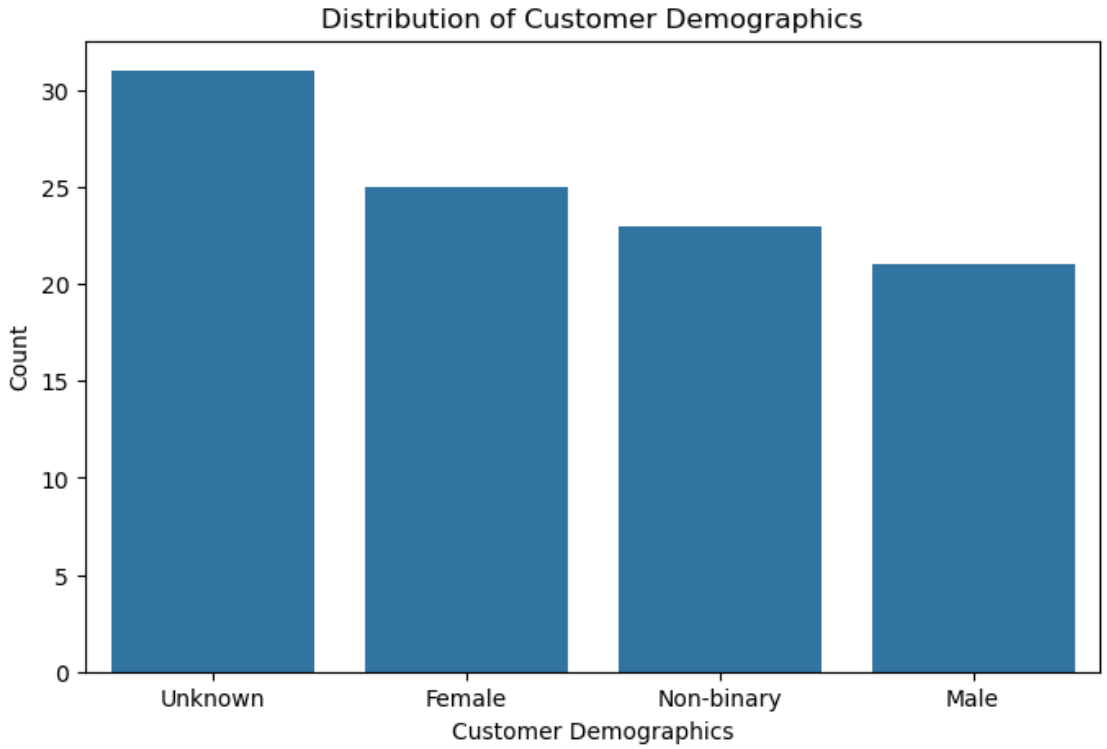
```
df['Lead time'] = df['Lead time'].astype(int)
df['Production volumes'] = df['Production volumes'].astype(int)
df['Manufacturing lead time'] = df['Manufacturing lead time'].astype(int)
df['Manufacturing costs'] = df['Manufacturing costs'].astype(float)
df['Inspection results'] = df['Inspection results'].astype(str)
df['Defect rates'] = df['Defect rates'].astype(float)
df['Transportation modes'] = df['Transportation modes'].astype(str)
df['Routes'] = df['Routes'].astype(str)
df['Costs'] = df['Costs'].astype(float)

# Display the updated dataframe
print(df.head())
```

```
  Product type   SKU      Price  Availability  Number of products sold  \
0     haircare  SKU0  69.808006            55                      802
1     skincare  SKU1  14.843523            95                      736
2     haircare  SKU2  11.319683            34                        8
3     skincare  SKU3  61.163343            68                       83
4     skincare  SKU4   4.805496            26                      871

   Revenue generated Customer demographics  Stock levels  Lead times  \
0        8661.996792            Non-binary            58           7
1        7460.900065                Female            53          30
2        9577.749626               Unknown             1          10
3        7766.836426            Non-binary            23          13
4        2686.505152            Non-binary             5           3

   Order quantities  …  Location Lead time  Production volumes  \
0                96  …    Mumbai        29                 215
1                37  …    Mumbai        23                 517
2                88  …    Mumbai        12                 971
3                59  …   Kolkata        24                 937
4                56  …     Delhi         5                 414

  Manufacturing lead time Manufacturing costs  Inspection results  \
0                      29           46.279879             Pending
1                      30           33.616769             Pending
2                      27           30.688019             Pending
3                      18           35.624741                Fail
4                       3           92.065161                Fail

   Defect rates  Transportation modes   Routes       Costs
0      0.226410                  Road  Route B  187.752075
1      4.854068                  Road  Route B  503.065579
2      4.580593                   Air  Route C  141.920282
3      4.746649                  Rail  Route A  254.776159
4      3.145580                   Air  Route A  923.440632
```

```
[5 rows x 24 columns]
```

[10]:
```
# Check the distribution of values in the 'Customer demographics' column
print(df['Customer demographics'].value_counts())

# Visualize the distribution
plt.figure(figsize=(8, 5))
sns.countplot(data=df, x='Customer demographics', order=df['Customer␣
 ↪demographics'].value_counts().index)
plt.title('Distribution of Customer Demographics')
plt.xlabel('Customer Demographics')
plt.ylabel('Count')
plt.show()
```

```
Customer demographics
Unknown       31
Female        25
Non-binary    23
Male          21
Name: count, dtype: int64
```

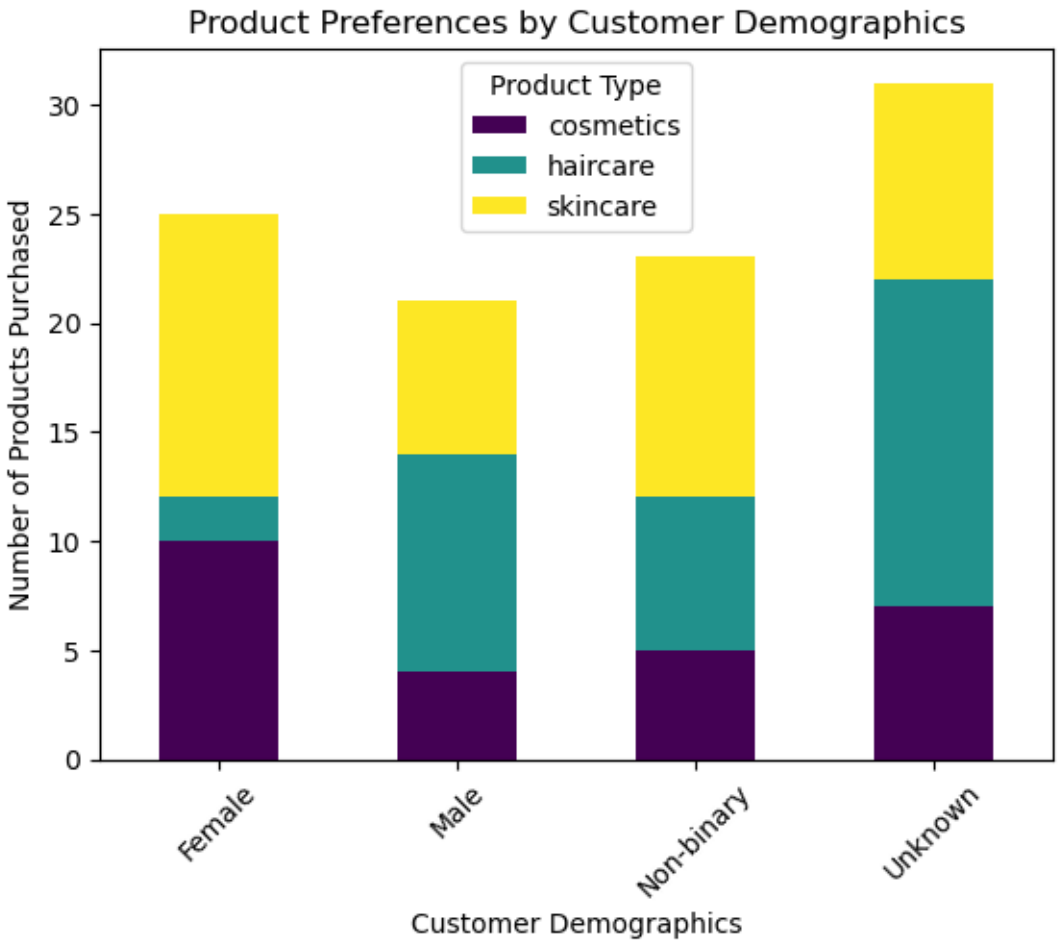



[11]:
```
# Group by 'Customer demographics' and 'Product type' to see product preferences
```

```
product_preferences = df.groupby(['Customer demographics', 'Product type']).
 ↪size().unstack(fill_value=0)

# Display the product preferences
print("Product Preferences by Customer Demographics:")
print(product_preferences)

# Visualize the product preferences
plt.figure(figsize=(12, 6))
product_preferences.plot(kind='bar', stacked=True, colormap='viridis')
plt.title('Product Preferences by Customer Demographics')
plt.xlabel('Customer Demographics')
plt.ylabel('Number of Products Purchased')
plt.xticks(rotation=45)
plt.legend(title='Product Type')
plt.show()
```

```
Product Preferences by Customer Demographics:
Product type           cosmetics  haircare  skincare
Customer demographics
Female                        10         2        13
Male                           4        10         7
Non-binary                     5         7        11
Unknown                        7        15         9
```

```
<Figure size 1200x600 with 0 Axes>
```




[12]:
```
# Replace "Unknown" with "Male"
df['Customer demographics'] = df['Customer demographics'].replace('Unknown',␣
  ↪'Male')

# Verify the updated distribution
print("Updated Customer Demographics Distribution:")
print(df['Customer demographics'].value_counts())

# Verify the updated product preferences
updated_preferences = df.groupby(['Customer demographics', 'Product type']).
  ↪size().unstack(fill_value=0)
print("\nUpdated Product Preferences by Customer Demographics:")
print(updated_preferences)
```

```
Updated Customer Demographics Distribution:
Customer demographics
Male          52
Female        25
```

```
Non-binary    23
Name: count, dtype: int64

Updated Product Preferences by Customer Demographics:
Product type           cosmetics  haircare  skincare
Customer demographics
Female                        10         2        13
Male                          11        25        16
Non-binary                     5         7        11
```

[13]:
```
# Replace "Non-binary" with "Female"
df['Customer demographics'] = df['Customer demographics'].replace('Non-binary',␣
 ↪'Female')

# Verify the updated distribution
print("Updated Customer Demographics Distribution:")
print(df['Customer demographics'].value_counts())

# Verify the updated product preferences
updated_preferences = df.groupby(['Customer demographics', 'Product type']).
 ↪size().unstack(fill_value=0)
print("\nUpdated Product Preferences by Customer Demographics:")
print(updated_preferences)
```

```
Updated Customer Demographics Distribution:
Customer demographics
Male      52
Female    48
Name: count, dtype: int64

Updated Product Preferences by Customer Demographics:
Product type           cosmetics  haircare  skincare
Customer demographics
Female                        15         9        24
Male                          11        25        16
```

[15]:
```
# Save the cleaned dataset to a new .xlsx file
df.to_excel('cleaned_supply_chain_data.xlsx', index=False)
```

[16]:
```
# Display the first 5 rows of the cleaned dataset
print(df.head())

# Check for missing values again
print("Missing values after cleaning:")
print(df.isnull().sum())
```

```
  Product type   SKU      Price  Availability  Number of products sold  \
0     haircare  SKU0  69.808006            55                      802
1     skincare  SKU1  14.843523            95                      736
```

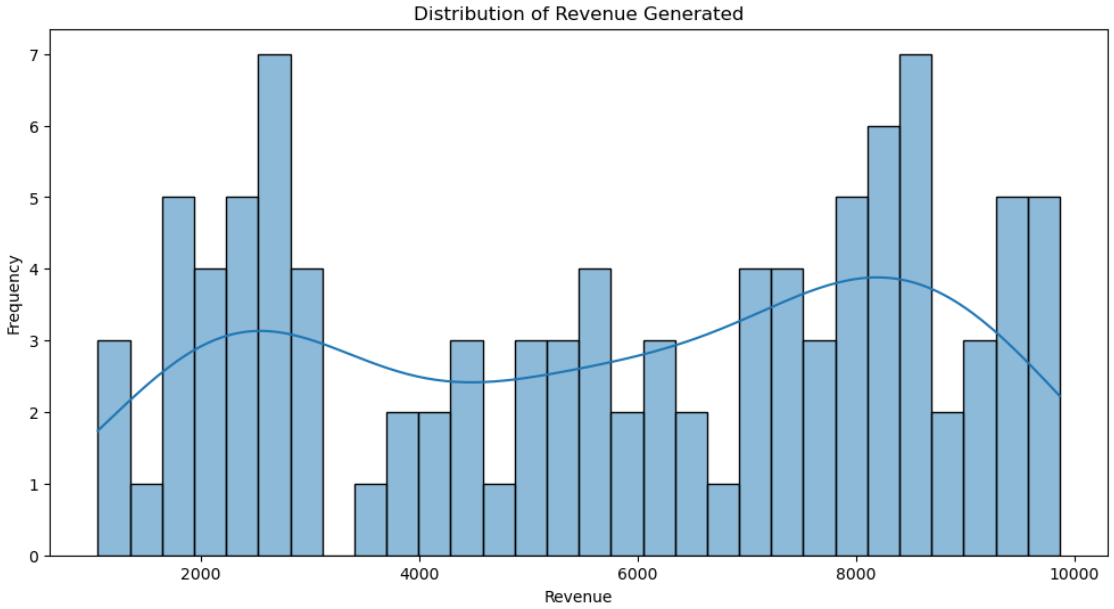
```
2     haircare  SKU2  11.319683           34                       8
3     skincare  SKU3  61.163343           68                      83
4     skincare  SKU4   4.805496           26                     871

   Revenue generated Customer demographics  Stock levels  Lead times  \
0        8661.996792                Female            58           7
1        7460.900065                Female            53          30
2        9577.749626                  Male             1          10
3        7766.836426                Female            23          13
4        2686.505152                Female             5           3

   Order quantities  …  Location Lead time  Production volumes  \
0                96  …    Mumbai        29                 215
1                37  …    Mumbai        23                 517
2                88  …    Mumbai        12                 971
3                59  …   Kolkata        24                 937
4                56  …     Delhi         5                 414

  Manufacturing lead time Manufacturing costs  Inspection results  \
0                      29           46.279879             Pending
1                      30           33.616769             Pending
2                      27           30.688019             Pending
3                      18           35.624741                Fail
4                       3           92.065161                Fail

   Defect rates  Transportation modes   Routes       Costs
0      0.226410                  Road  Route B  187.752075
1      4.854068                  Road  Route B  503.065579
2      4.580593                   Air  Route C  141.920282
3      4.746649                  Rail  Route A  254.776159
4      3.145580                   Air  Route A  923.440632

[5 rows x 24 columns]
Missing values after cleaning:
Product type               0
SKU                        0
Price                      0
Availability               0
Number of products sold    0
Revenue generated          0
Customer demographics      0
Stock levels               0
Lead times                 0
Order quantities           0
Shipping times             0
Shipping carriers          0
Shipping costs             0
Supplier name              0
```

```
Location                    0
Lead time                   0
Production volumes          0
Manufacturing lead time     0
Manufacturing costs         0
Inspection results          0
Defect rates                0
Transportation modes        0
Routes                      0
Costs                       0
dtype: int64
```

[17]:
```python
# Summary statistics for numerical columns
print(df.describe())

# Distribution of categorical columns
print(df['Product type'].value_counts())
print(df['Shipping carriers'].value_counts())
print(df['Customer demographics'].value_counts())

# Visualize distributions
plt.figure(figsize=(12, 6))
sns.histplot(df['Revenue generated'], bins=30, kde=True)
plt.title('Distribution of Revenue Generated')
plt.xlabel('Revenue')
plt.ylabel('Frequency')
plt.show()
```

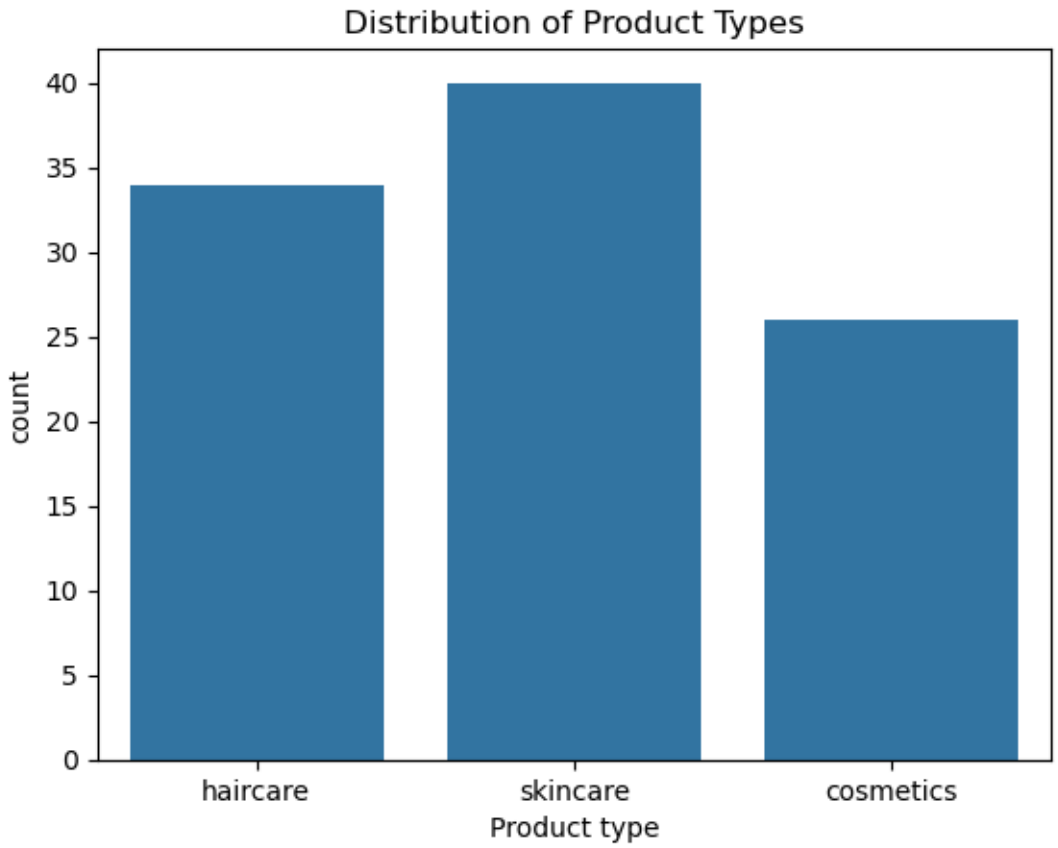
```
            Price  Availability  Number of products sold  Revenue generated  \
count  100.000000    100.000000               100.000000         100.000000
mean    49.462461     48.400000               460.990000        5776.048187
std     31.168193     30.743317               303.780074        2732.841744
min      1.699976      1.000000                 8.000000        1061.618523
25%     19.597823     22.750000               184.250000        2812.847151
50%     51.239830     43.500000               392.500000        6006.352023
75%     77.198228     75.000000               704.250000        8253.976920
max     99.171329    100.000000               996.000000        9866.465458

       Stock levels  Lead times  Order quantities  Shipping times  \
count    100.000000  100.000000        100.000000      100.000000
mean      47.770000   15.960000         49.220000        5.750000
std       31.369372    8.785801         26.784429        2.724283
min        0.000000    1.000000          1.000000        1.000000
25%       16.750000    8.000000         26.000000        3.750000
50%       47.500000   17.000000         52.000000        6.000000
75%       73.000000   24.000000         71.250000        8.000000
max      100.000000   30.000000         96.000000       10.000000
```

```
       Shipping costs   Lead time  Production volumes  \
count      100.000000  100.000000          100.000000
mean         5.548149   17.080000          567.840000
std          2.651376    8.846251          263.046861
min          1.013487    1.000000          104.000000
25%          3.540248   10.000000          352.000000
50%          5.320534   18.000000          568.500000
75%          7.601695   25.000000          797.000000
max          9.929816   30.000000          985.000000

       Manufacturing lead time  Manufacturing costs  Defect rates       Costs
count                100.00000           100.000000    100.000000  100.000000
mean                  14.77000            47.266693      2.277158  529.245782
std                    8.91243            28.982841      1.461366  258.301696
min                    1.00000             1.085069      0.018608  103.916248
25%                    7.00000            22.983299      1.009650  318.778455
50%                   14.00000            45.905622      2.141863  520.430444
75%                   23.00000            68.621026      3.563995  763.078231
max                   30.00000            99.466109      4.939255  997.413450
Product type
skincare     40
haircare     34
cosmetics    26
Name: count, dtype: int64
Shipping carriers
Carrier B    43
Carrier C    29
Carrier A    28
Name: count, dtype: int64
Customer demographics
Male      52
Female    48
Name: count, dtype: int64
```

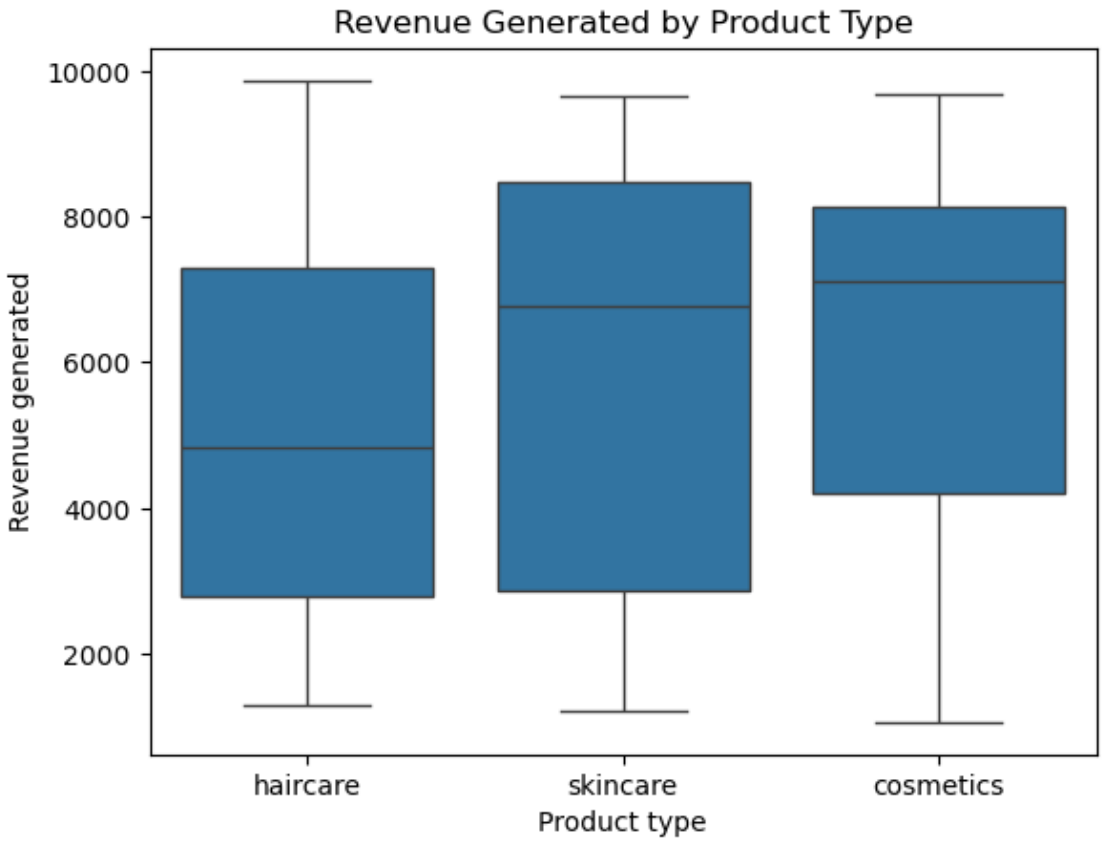



[18]:
```
sns.scatterplot(x='Price', y='Revenue generated', data=df)
plt.title('Price vs. Revenue Generated')
plt.show()
```

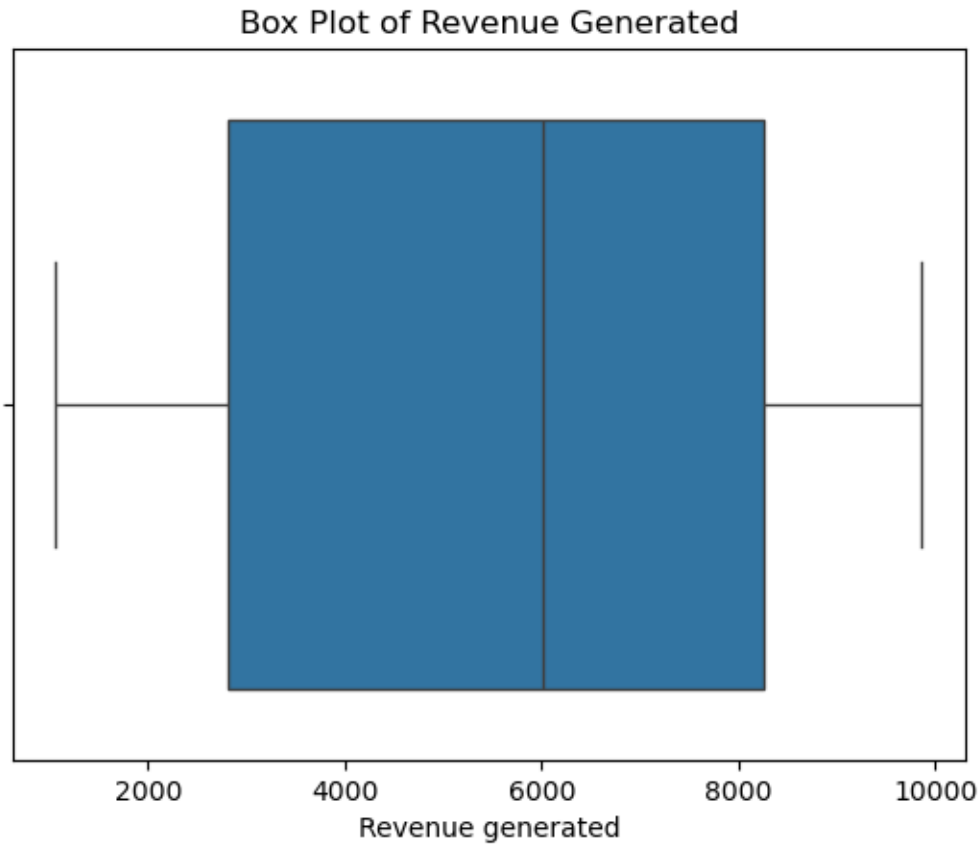

[19]:
```
sns.countplot(x='Product type', data=df)
plt.title('Distribution of Product Types')
plt.show()
```



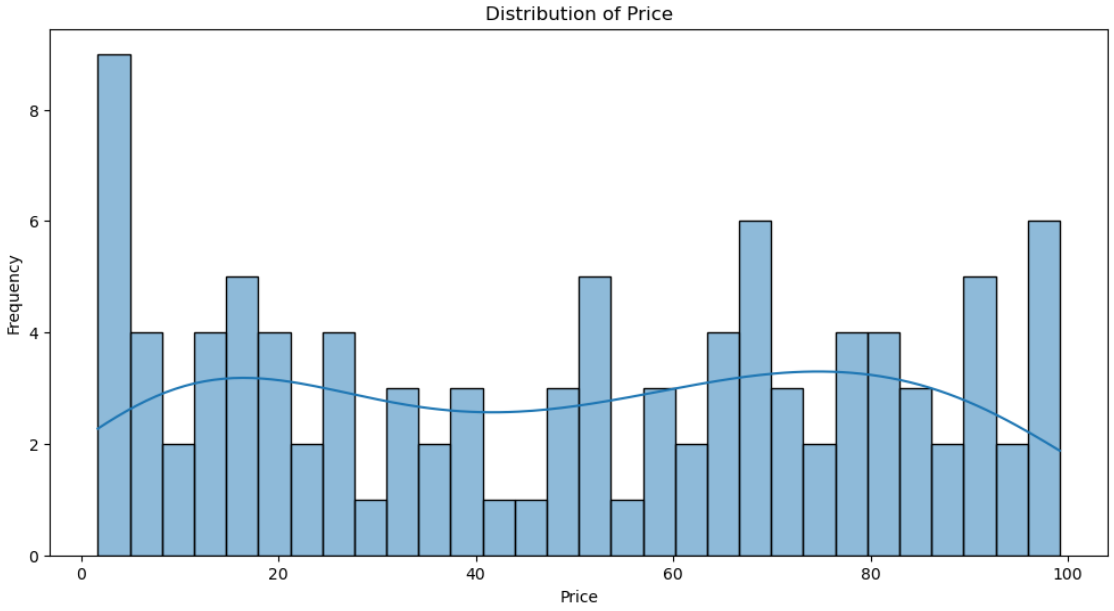

[20]:
```
sns.boxplot(x='Product type', y='Revenue generated', data=df)
plt.title('Revenue Generated by Product Type')
plt.show()
```

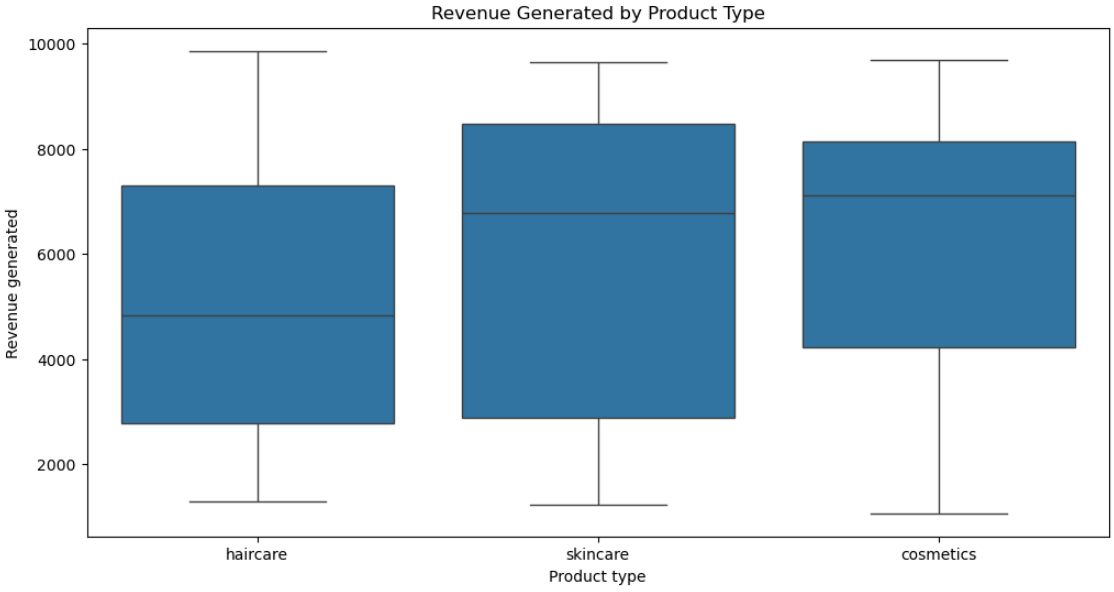



[21]:
```
sns.boxplot(x=df['Revenue generated'])
plt.title('Box Plot of Revenue Generated')
plt.show()
```

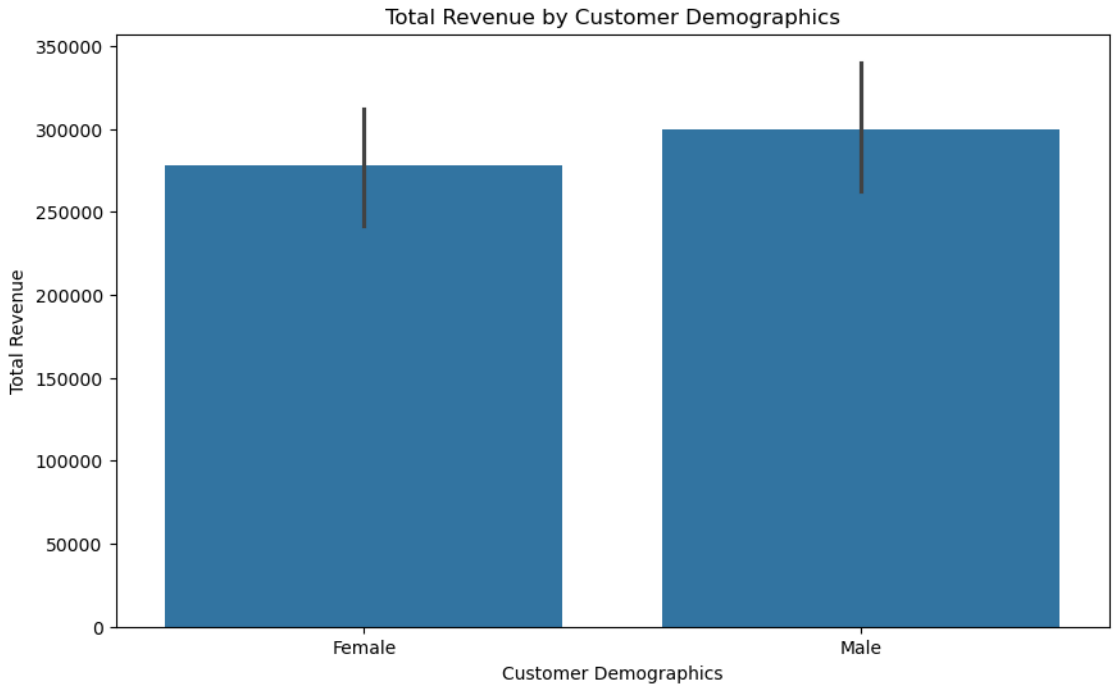



[23]:
```
# 1. Analyze Price Distribution
plt.figure(figsize=(12, 6))
sns.histplot(df['Price'], bins=30, kde=True)
plt.title('Distribution of Price')
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.show()

# 2. Compare Revenue by Product Type
plt.figure(figsize=(12, 6))
sns.boxplot(x='Product type', y='Revenue generated', data=df)
plt.title('Revenue Generated by Product Type')
plt.show()
```

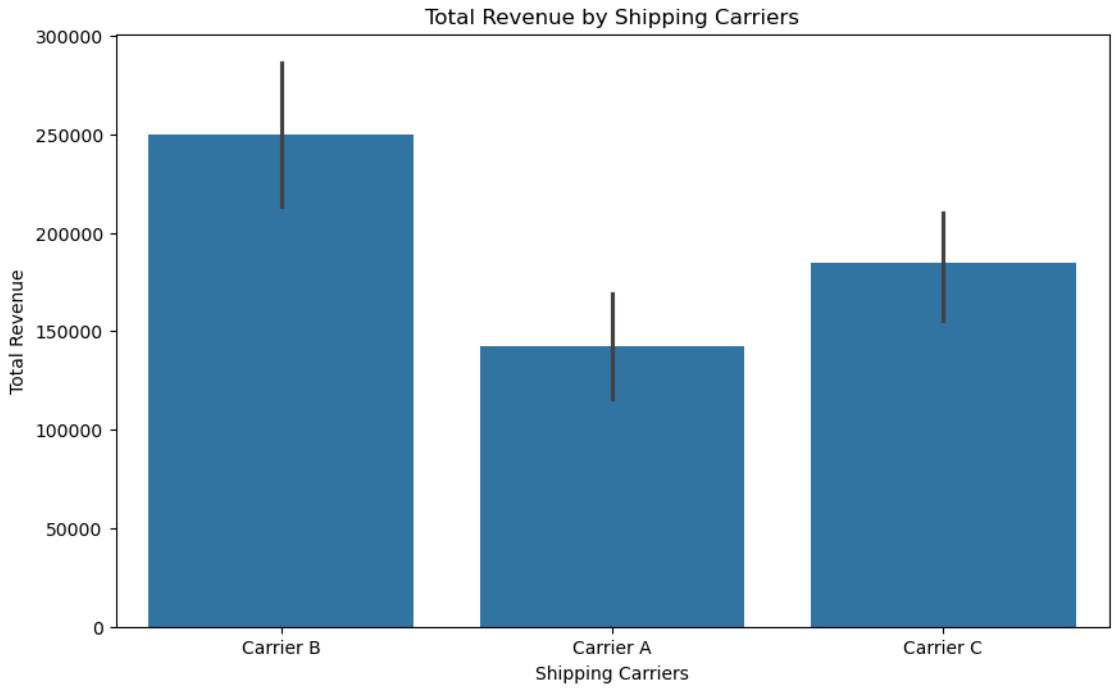


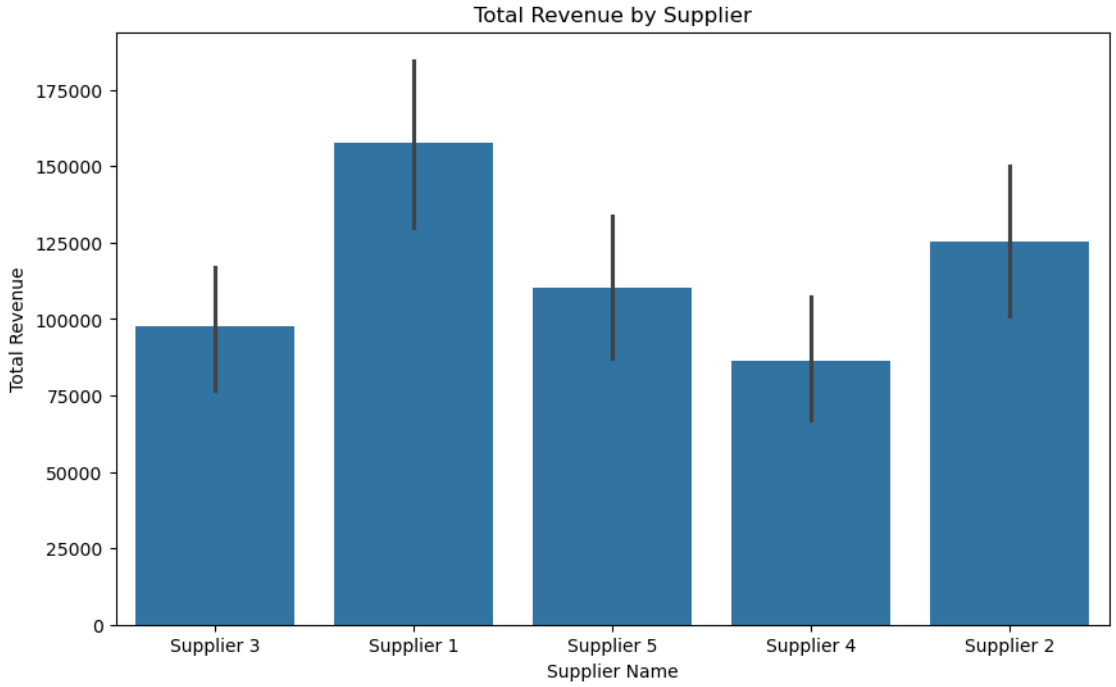

[26]:
```
plt.figure(figsize=(10, 6))
sns.barplot(data=df, x='Customer demographics', y='Revenue generated',␣
 ↪estimator=sum)
plt.title('Total Revenue by Customer Demographics')
plt.xlabel('Customer Demographics')
plt.ylabel('Total Revenue')
plt.show()
```



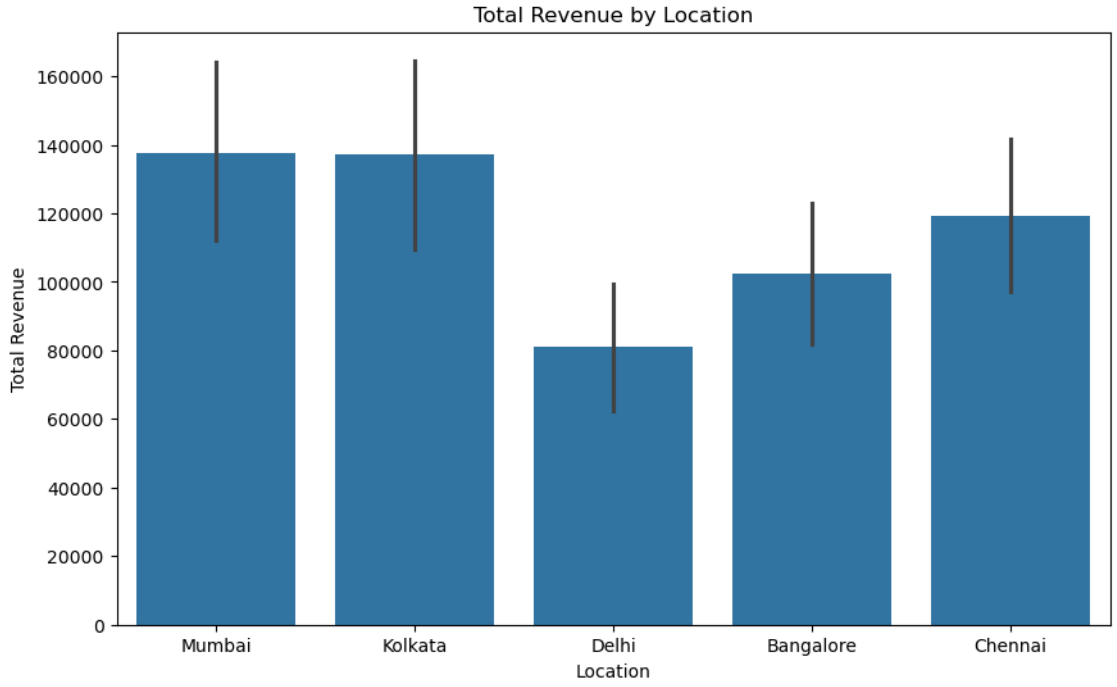

[27]:
```
plt.figure(figsize=(10, 6))
sns.barplot(data=df, x='Shipping carriers', y='Revenue generated',␣
 ↪estimator=sum)
plt.title('Total Revenue by Shipping Carriers')
plt.xlabel('Shipping Carriers')
plt.ylabel('Total Revenue')
plt.show()
```



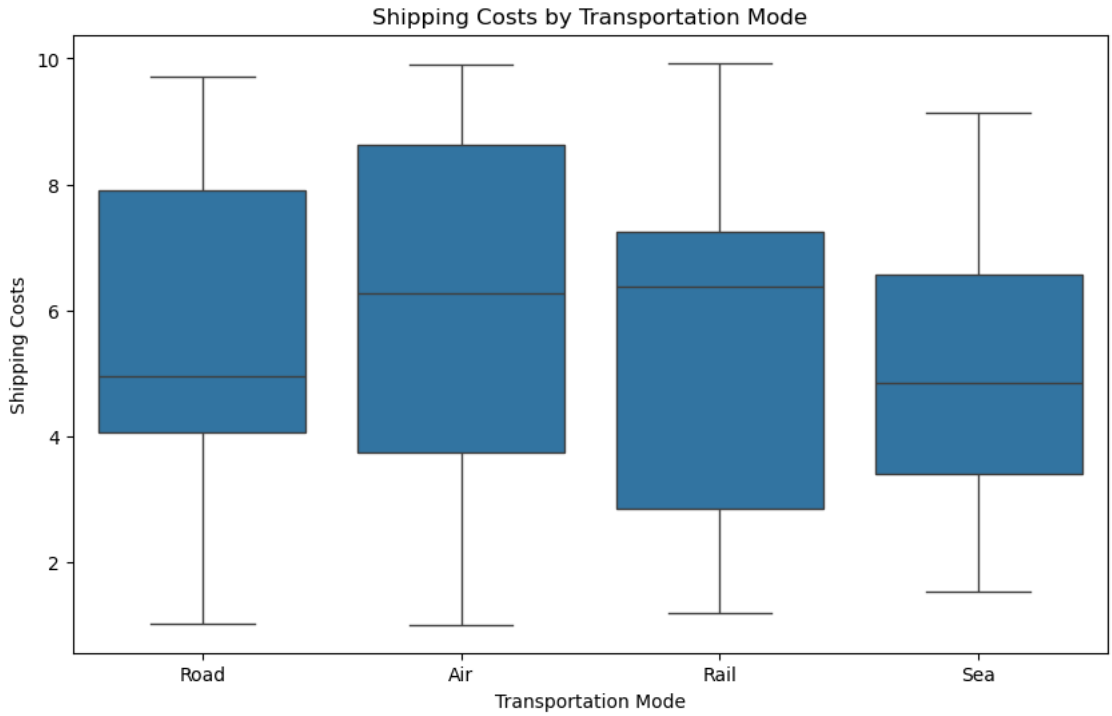

[28]:
```
plt.figure(figsize=(10, 6))
sns.barplot(data=df, x='Supplier name', y='Revenue generated', estimator=sum)
plt.title('Total Revenue by Supplier')
plt.xlabel('Supplier Name')
plt.ylabel('Total Revenue')
plt.show()
```



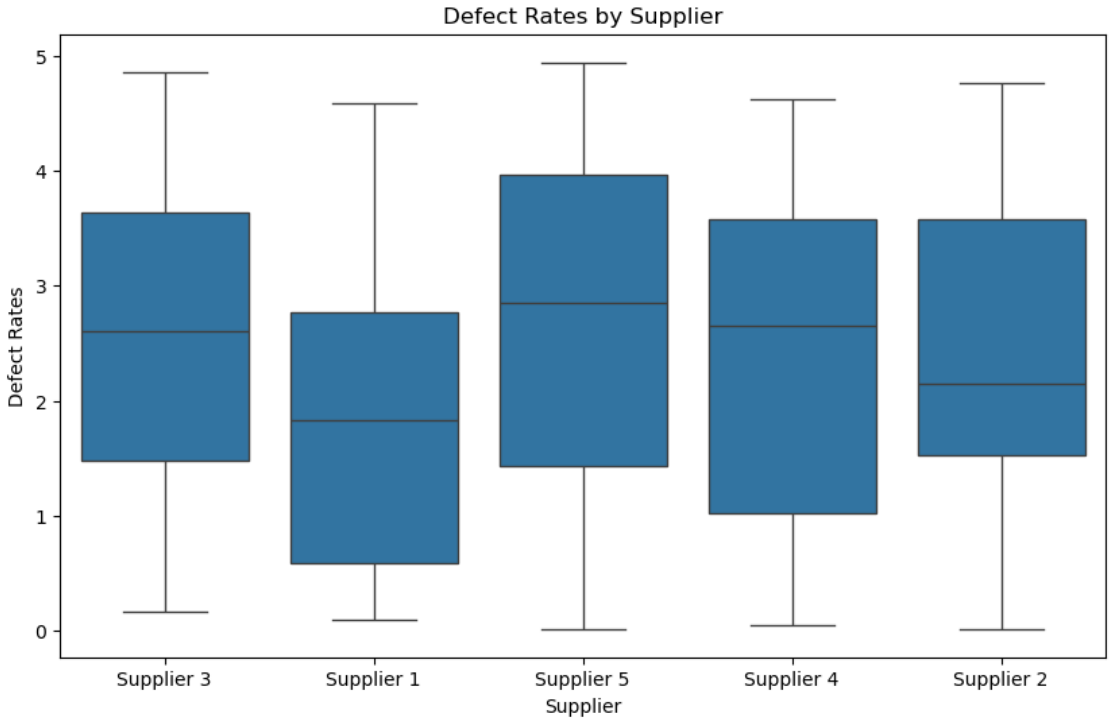

[29]:
```
plt.figure(figsize=(10, 6))
sns.barplot(data=df, x='Location', y='Revenue generated', estimator=sum)
plt.title('Total Revenue by Location')
plt.xlabel('Location')
plt.ylabel('Total Revenue')
plt.show()
```



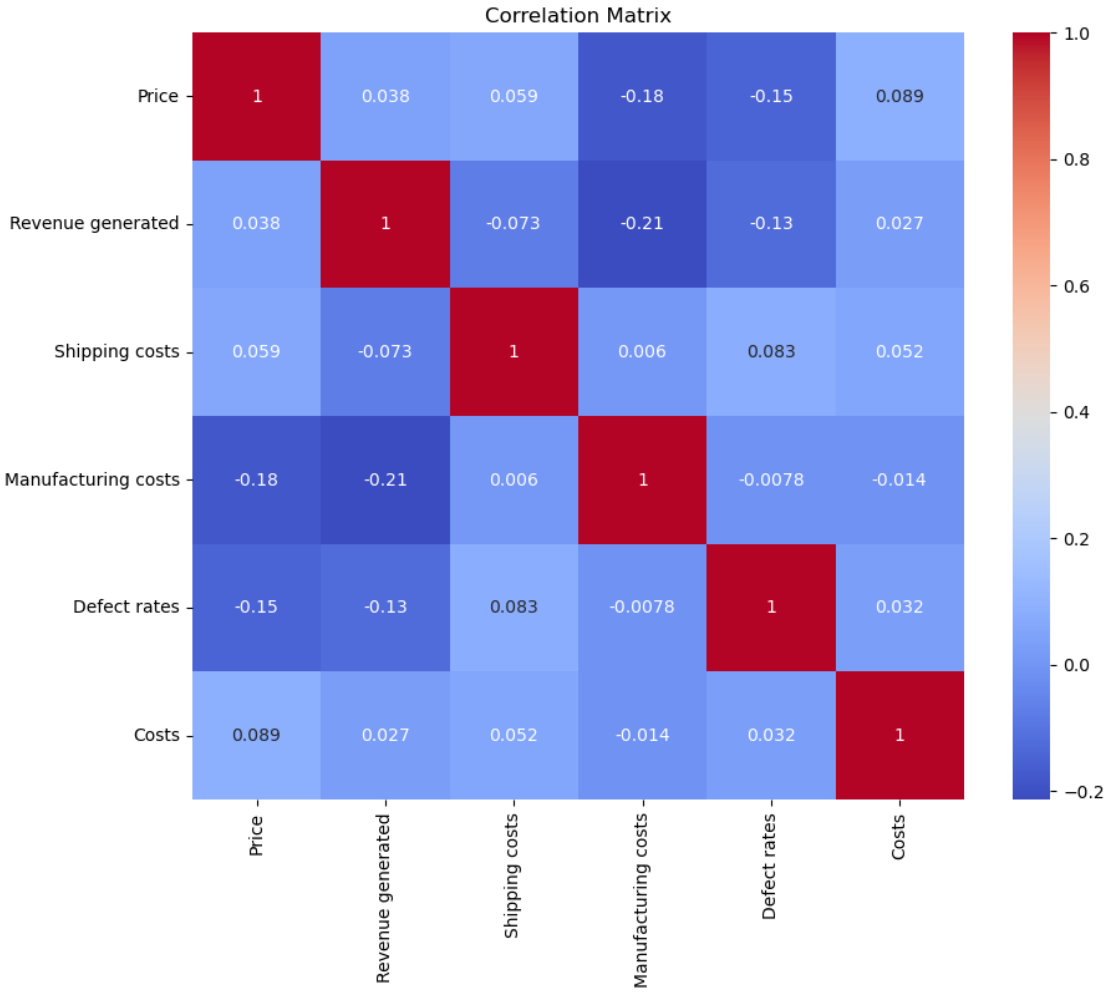

[30]:
```
# Shipping Costs by Transportation Mode
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, x='Transportation modes', y='Shipping costs')
plt.title('Shipping Costs by Transportation Mode')
plt.xlabel('Transportation Mode')
plt.ylabel('Shipping Costs')
plt.show()
```




[31]:
```
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, x='Supplier name', y='Defect rates')
plt.title('Defect Rates by Supplier')
plt.xlabel('Supplier')
plt.ylabel('Defect Rates')
plt.show()
```

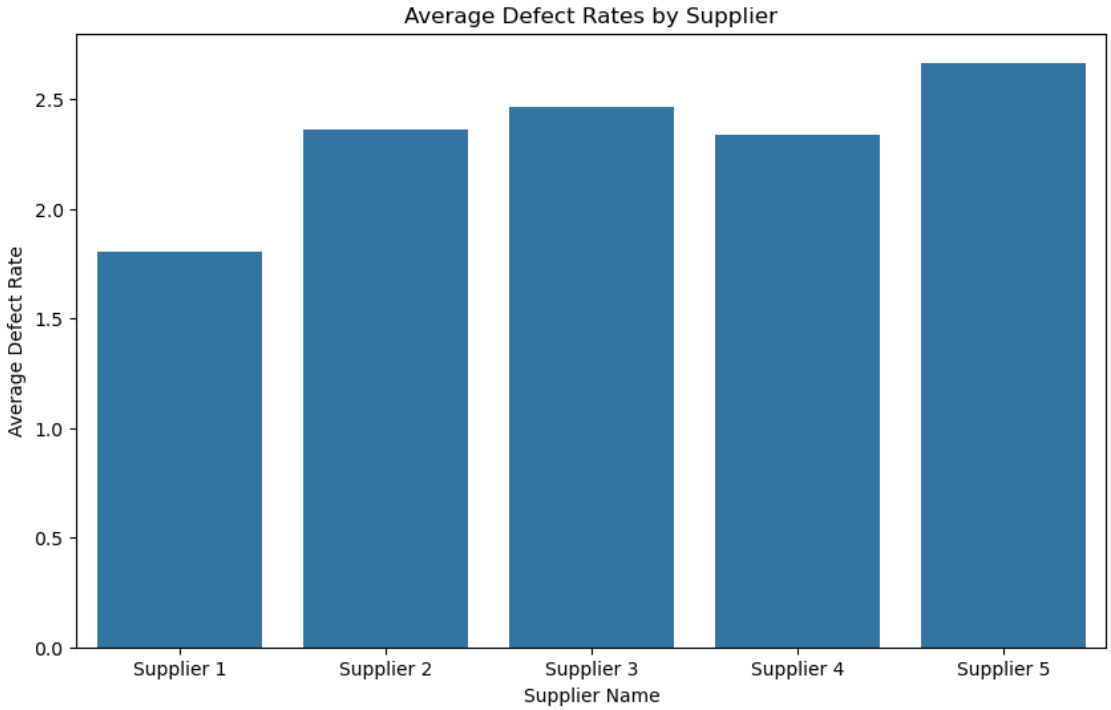



[25]:
```
# Select only numerical columns
numerical_df = df.select_dtypes(include=['float64', 'int64'])

# Generate the correlation matrix
plt.figure(figsize=(10, 8))
sns.heatmap(numerical_df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

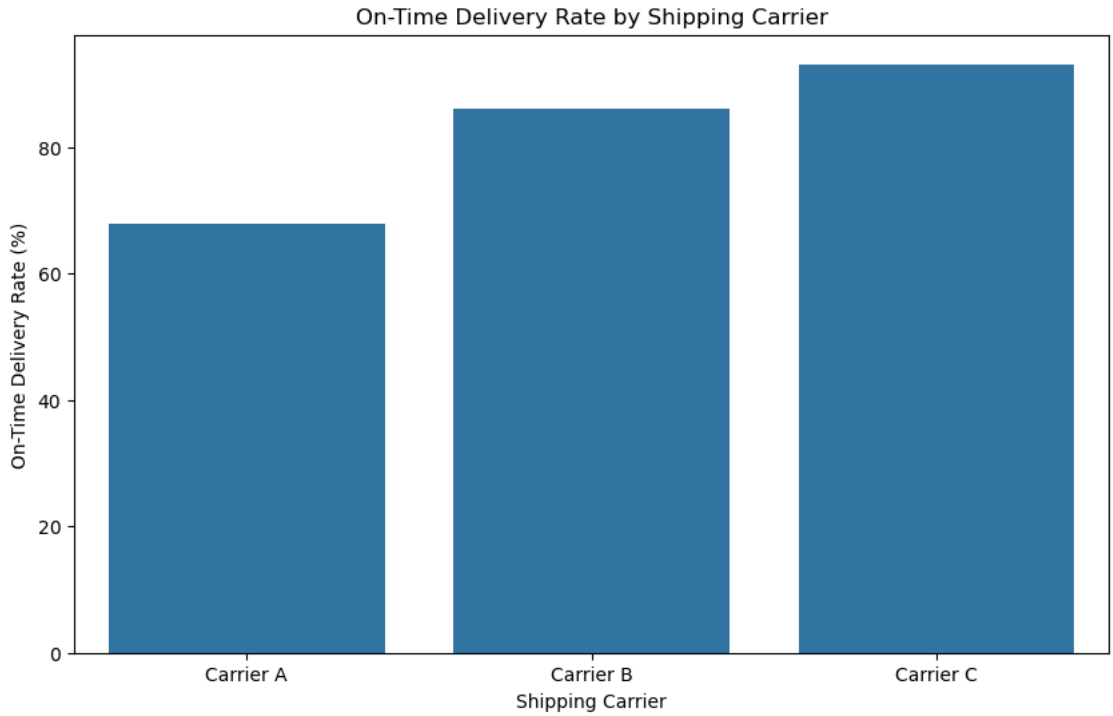



[32]:
```
# Group by 'Supplier name' and calculate average defect rates
defect_rates_by_supplier = df.groupby('Supplier name')['Defect rates'].mean()

# Visualize defect rates by supplier
plt.figure(figsize=(10, 6))
sns.barplot(x=defect_rates_by_supplier.index, y=defect_rates_by_supplier.values)
plt.title('Average Defect Rates by Supplier')
plt.xlabel('Supplier Name')
plt.ylabel('Average Defect Rate')
```

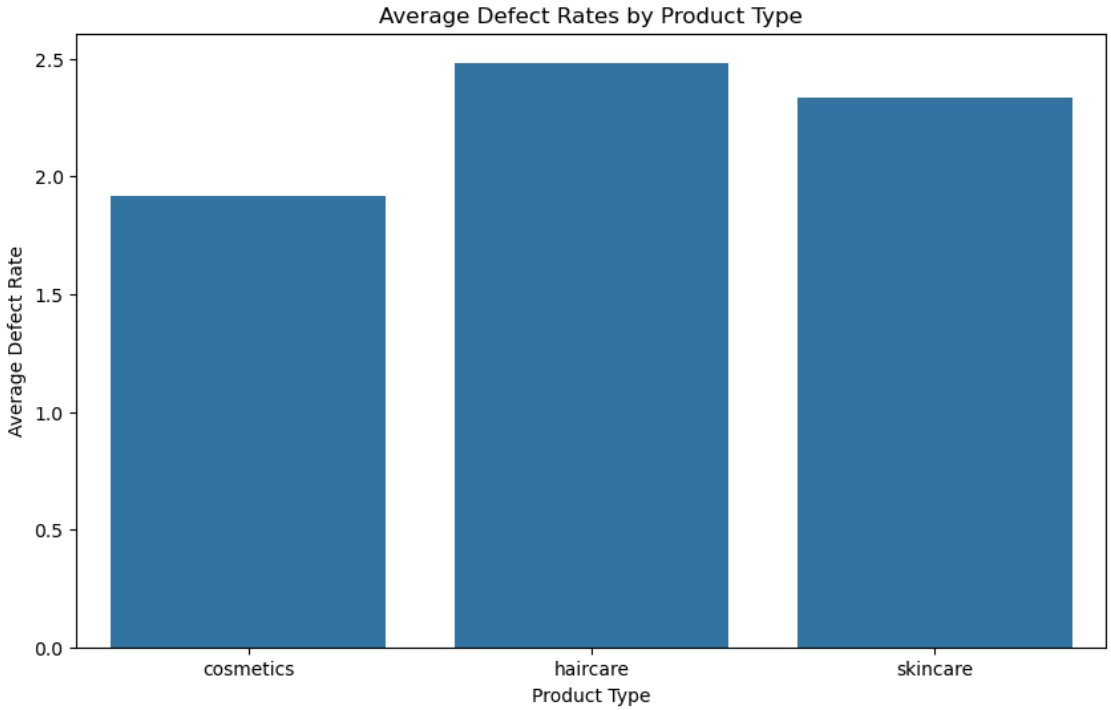
[32]:
```
Text(0, 0.5, 'Average Defect Rate')
```



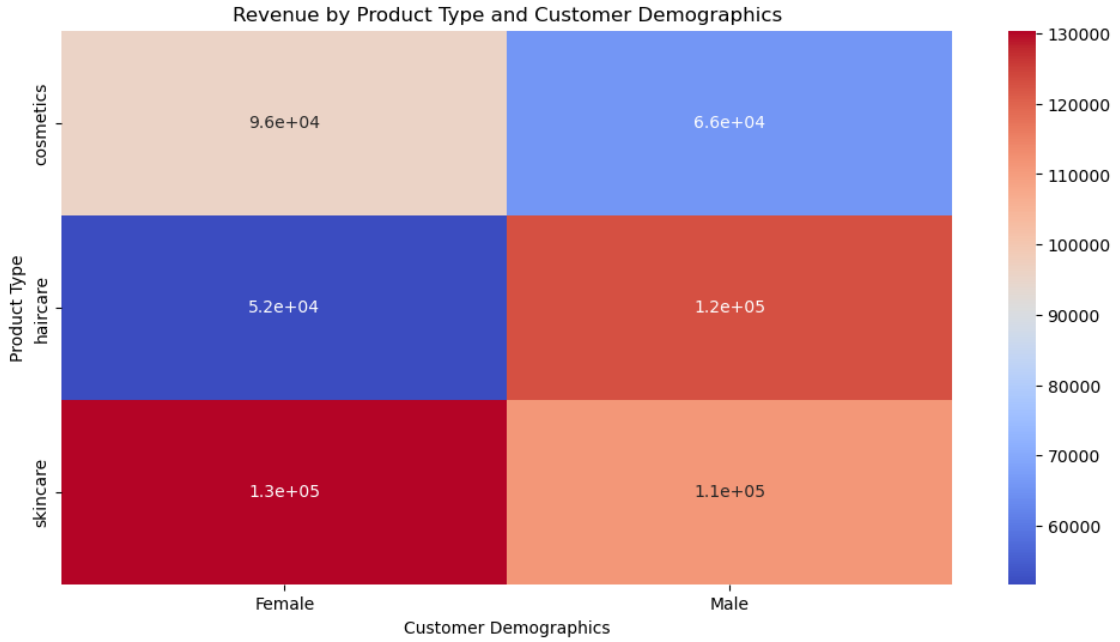

[33]:
```
# Calculate on-time delivery rate by shipping carrier
df['On_Time_Delivery'] = df['Shipping times'] <= df['Lead times']
on_time_delivery_by_carrier = df.groupby('Shipping␣
 ↪carriers')['On_Time_Delivery'].mean() * 100

# Visualize on-time delivery rates by carrier
plt.figure(figsize=(10, 6))
sns.barplot(x=on_time_delivery_by_carrier.index, y=on_time_delivery_by_carrier.
 ↪values)
plt.title('On-Time Delivery Rate by Shipping Carrier')
plt.xlabel('Shipping Carrier')
plt.ylabel('On-Time Delivery Rate (%)')
plt.show()
```



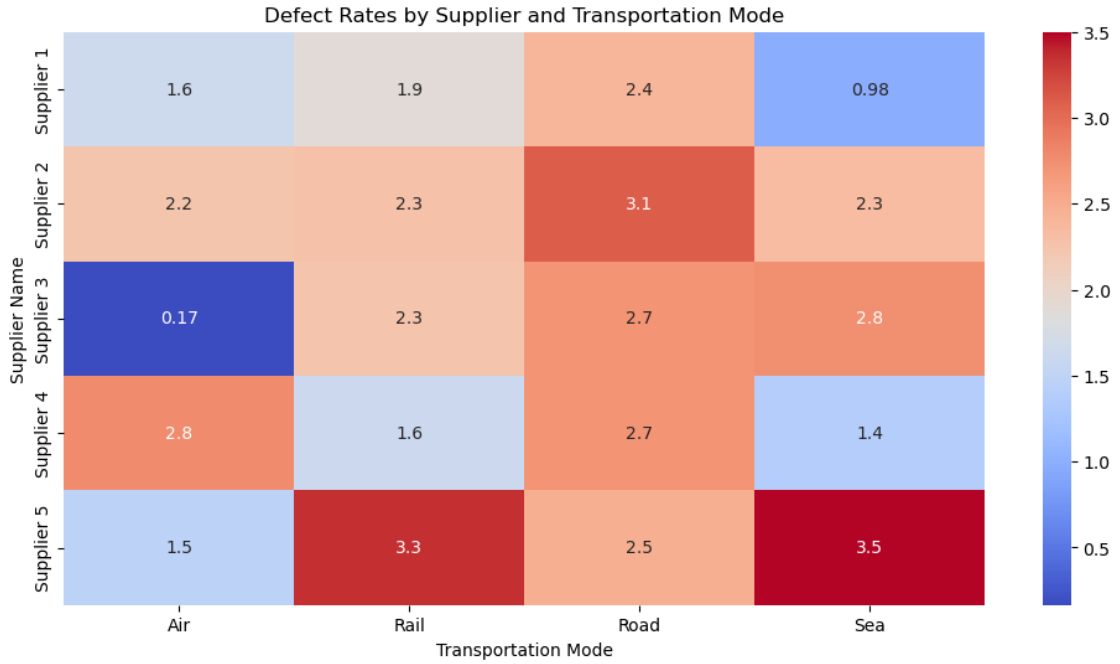

[34]:
```
# Group by 'Product type' and calculate average defect rates
defect_rates_by_product = df.groupby('Product type')['Defect rates'].mean()

# Visualize defect rates by product type
plt.figure(figsize=(10, 6))
sns.barplot(x=defect_rates_by_product.index, y=defect_rates_by_product.values)
plt.title('Average Defect Rates by Product Type')
plt.xlabel('Product Type')
plt.ylabel('Average Defect Rate')
plt.show()
```



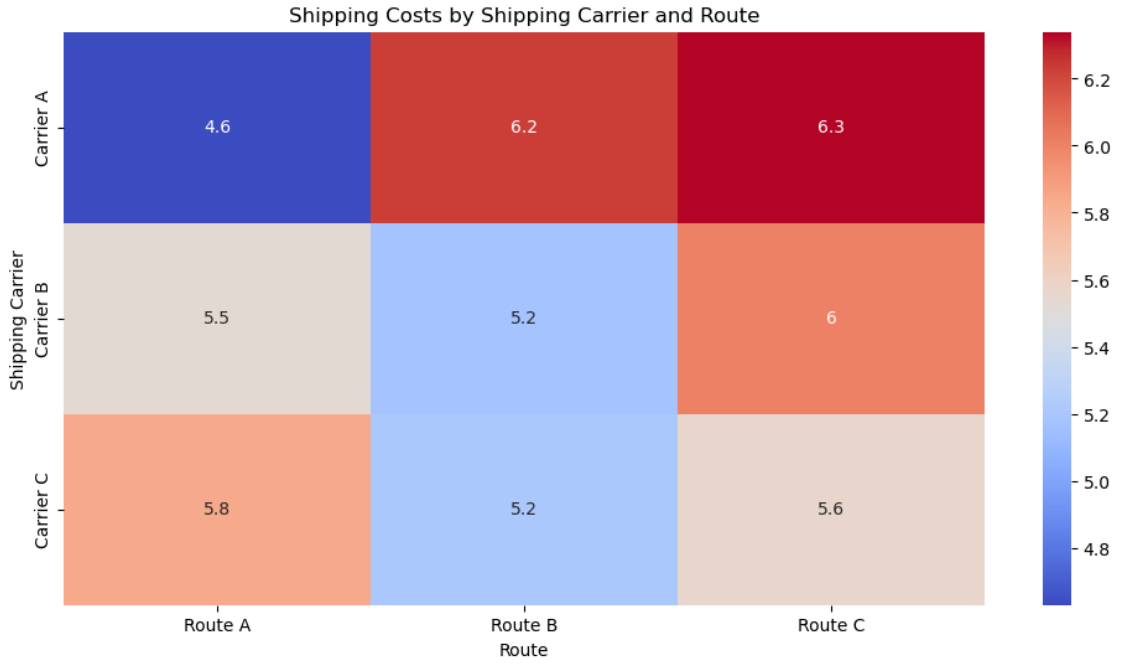

[35]:
```
# Group by 'Product type' and 'Customer demographics' and calculate total revenue
revenue_by_product_and_demographics = df.groupby(['Product type', 'Customer demographics'])['Revenue generated'].sum().unstack()

# Visualize revenue by product type and customer demographics
plt.figure(figsize=(12, 6))
sns.heatmap(revenue_by_product_and_demographics, annot=True, cmap='coolwarm')
plt.title('Revenue by Product Type and Customer Demographics')
plt.xlabel('Customer Demographics')
plt.ylabel('Product Type')
plt.show()
```




[36]:
```
# Group by 'Supplier name' and 'Transportation modes' and calculate average␣
 ↪defect rates
defect_rates_by_supplier_and_transport = df.groupby(['Supplier name',␣
 ↪'Transportation modes'])['Defect rates'].mean().unstack()

# Visualize defect rates by supplier and transportation mode
plt.figure(figsize=(12, 6))
sns.heatmap(defect_rates_by_supplier_and_transport, annot=True, cmap='coolwarm')
plt.title('Defect Rates by Supplier and Transportation Mode')
plt.xlabel('Transportation Mode')
plt.ylabel('Supplier Name')
plt.show()
```




[37]:
```
# Group by 'Shipping carriers' and 'Routes' and calculate average shipping costs
shipping_costs_by_carrier_and_route = df.groupby(['Shipping carriers',␣
 ↪'Routes'])['Shipping costs'].mean().unstack()

# Visualize shipping costs by carrier and route
plt.figure(figsize=(12, 6))
sns.heatmap(shipping_costs_by_carrier_and_route, annot=True, cmap='coolwarm')
plt.title('Shipping Costs by Shipping Carrier and Route')
plt.xlabel('Route')
plt.ylabel('Shipping Carrier')
plt.show()
```


Shipping Costs by Shipping Carrier and Route
Shipping Carrier
Carrier A
Carrier B
Carrier C
4.6
6.2
6.3
5.5
5.2
6
5.8
5.2
5.6
Route A
Route B
Route C
Route
6.2
6.0
5.8
5.6
5.4
5.2
5.0
4.8

[ ]: