Portfolio Introduction

M. Nafees Khawar

Data Analyst

Welcome to my portfolio, where I invite you to embark on a data-driven journey through the fascinating realm of supply chain analytics. I am M. Nafees Khawar, a dedicated Data Analyst, and I am excited to share with you the insights, discoveries, and experiences I've gathered in the world of data.

Supply chain analytics is not just about numbers; it's about understanding the intricate dance of products and services as they move from suppliers to customers. It's the art of uncovering patterns, making informed decisions, and optimizing operations in industries as diverse as manufacturing, retail, healthcare, and logistics.

For this project, I have had the privilege of working with a dynamic Fashion and Beauty startup, focusing on the vibrant world of Makeup products. The dataset I've gathered is a treasure trove of information, with each feature holding a unique story to tell. From product types to manufacturing costs, customer demographics to transportation modes, it's all here, waiting to be explored.

In this portfolio, I will take you on a journey through the data. We'll unveil the stories behind the numbers, explore the trends, patterns, and opportunities that lie within, and demonstrate the power of data-driven decision-making. It's a testament to the value of analytics in a rapidly evolving world, where informed decisions drive success.

Join me as we dive into this captivating journey of data exploration and analysis. I look forward to sharing my passion for data with you, and I hope this portfolio inspires you as much as it has inspired me.

About Dataset

Supply chain analytics serves as a cornerstone of data-driven decision-making, empowering diverse industries like manufacturing, retail, healthcare, and logistics to optimize their operations. It entails the art of collecting, dissecting, and interpreting data that traces the intricate journey of products and services from suppliers to customers.

In my pursuit of data-driven excellence, I've harnessed a trove of invaluable insights from a thriving Fashion and Beauty startup. My dataset, the heartbeat of this supply chain analysis, centers on the dynamic world of Makeup products. Behold, a treasure chest of information:

```
>>>Product Type: The essence of variety.
>>>SKU: Uniqueness distilled into code.
>>>Price: The art of value.
>>>Availability: The heartbeat of supply.
>>>Number of products sold: A testament to demand.
>>>Revenue generated: The lifeblood of success.
>>>Customer demographics: Unveiling the faces of our audience.
>>>Stock levels: The pulse of inventory.
>>>Lead times: Predicting the future.
>>>Order quantities: The building blocks of transactions.
>>>Shipping times: Navigating time and space.
>>>Shipping carriers: The vehicles of delivery.
>>>Shipping costs: The price of connectivity.
>>>Supplier name: The roots of our products.
>>>Location: Where worlds meet.
>>>Lead time: A countdown to delivery.
>>>Production volumes: The output of industry.
>>>Manufacturing lead time: Crafting perfection.
>>>Manufacturing costs: The price of creation.
>>>Inspection results: Quality at a glance.
>>>Defect rates: Imperfections in numbers.
>>>Transportation modes: The journey choices.
>>>Routes: Mapping the path.
>>>Costs: The thread tying it all together.
```

In this portfolio, I unveil the insights and revelations drawn from this rich dataset, showcasing the power of data-driven decision-making in the realm of Makeup products and the intricate world of supply chains. Explore with me the stories behind the numbers, the patterns within the data, and the opportunities for optimization and growth.

Pre-Processing Data

Worked by: M.Nafees Khawar

Import Libraries & Dataset

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
data=pd.read_csv('supply_chain_data.csv')
data.head()
```

Out[2]:

	Product type	SKU	Price	Availability	Number of products sold	Revenue generated	Customer demographics	Stock levels	Lead times	q
0	haircare	SKU0	69.808006	55	802	8661.996792	Non-binary	58	7	
1	skincare	SKU1	14.843523	95	736	7460.900065	Female	53	30	
2	haircare	SKU2	11.319683	34	8	9577.749626	Unknown	1	10	
3	skincare	SKU3	61.163343	68	83	7766.836426	Non-binary	23	13	
4	skincare	SKU4	4.805496	26	871	2686.505152	Non-binary	5	3	

5 rows × 24 columns

In [3]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 24 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Product type             100 non-null    object 
 1   SKU                      100 non-null    object 
 2   Price                    100 non-null    float64
 3   Availability             100 non-null    int64  
 4   Number of products sold  100 non-null    int64  
 5   Revenue generated        100 non-null    float64
 6   Customer demographics    100 non-null    object 
 7   Stock levels             100 non-null    int64  
 8   Lead times               100 non-null    int64  
 9   Order quantities         100 non-null    int64  
 10  Shipping times           100 non-null    int64  
 11  Shipping carriers        100 non-null    object 
 12  Shipping costs           100 non-null    float64
 13  Supplier name            100 non-null    object 
 14  Location                 100 non-null    object 
 15  Lead time                100 non-null    int64  
 16  Production volumes       100 non-null    int64  
 17  Manufacturing lead time  100 non-null    int64  
 18  Manufacturing costs      100 non-null    float64
 19  Inspection results       100 non-null    object 
 20  Defect rates             100 non-null    float64
 21  Transportation modes     100 non-null    object 
 22  Routes                   100 non-null    object 
 23  Costs                    100 non-null    float64
dtypes: float64(6), int64(9), object(9)
memory usage: 18.9+ KB
```

In [4]:
```
print('The number of rows and columns in the dataset',data.shape)
```

```
The number of rows and columns in the dataset (100, 24)
```

Summary Statistics with Custom Background Gradient

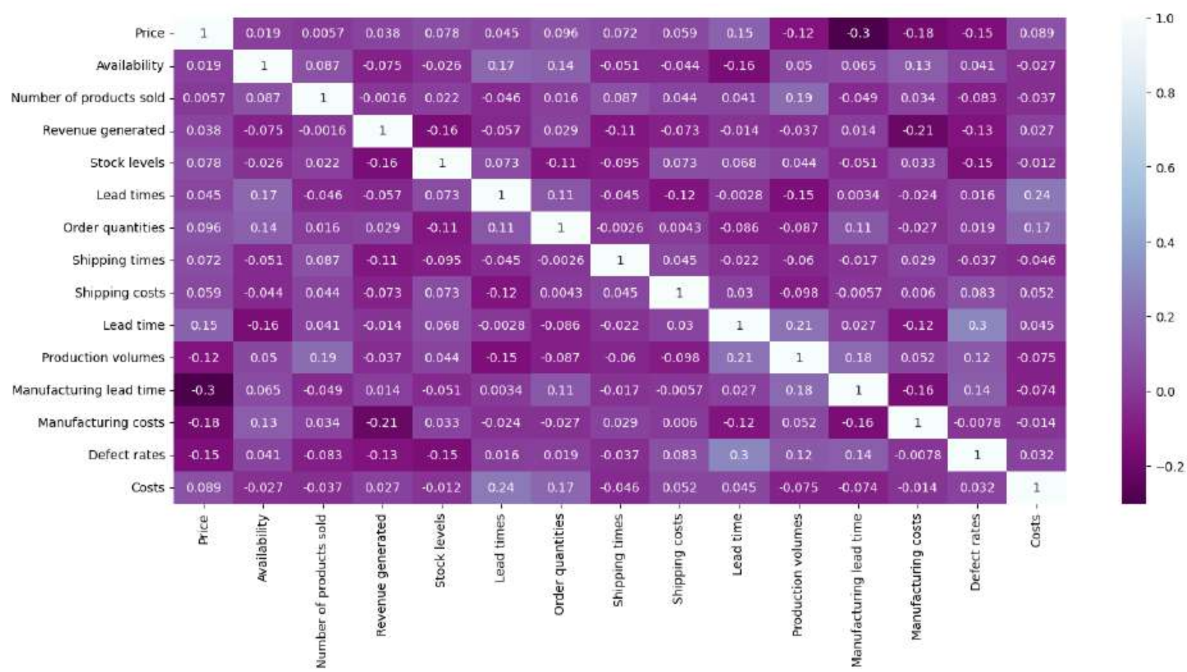
In [5]:
```python
data.describe().style.background_gradient(cmap='winter_r')
```

Out[5]:

	Price	Availability	Number of products sold	Revenue generated	Stock levels	Lead times	Order quantities	S
count	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	10(
mean	49.462461	48.400000	460.990000	5776.048187	47.770000	15.960000	49.220000	5
std	31.168193	30.743317	303.780074	2732.841744	31.369372	8.785801	26.784429	2
min	1.699976	1.000000	8.000000	1061.618523	0.000000	1.000000	1.000000	1
25%	19.597823	22.750000	184.250000	2812.847151	16.750000	8.000000	26.000000	3
50%	51.239831	43.500000	392.500000	6006.352023	47.500000	17.000000	52.000000	6
75%	77.198228	75.000000	704.250000	8253.976921	73.000000	24.000000	71.250000	8
max	99.171329	100.000000	996.000000	9866.465458	100.000000	30.000000	96.000000	1(

Correlation Heatmap of Data

In [6]:
```python
plt.figure(figsize=(16, 7))
sns.heatmap(data.corr(),annot=True,cmap='BuPu_r')
plt.show()
```



Percentage of Missing Values in the DataFrame

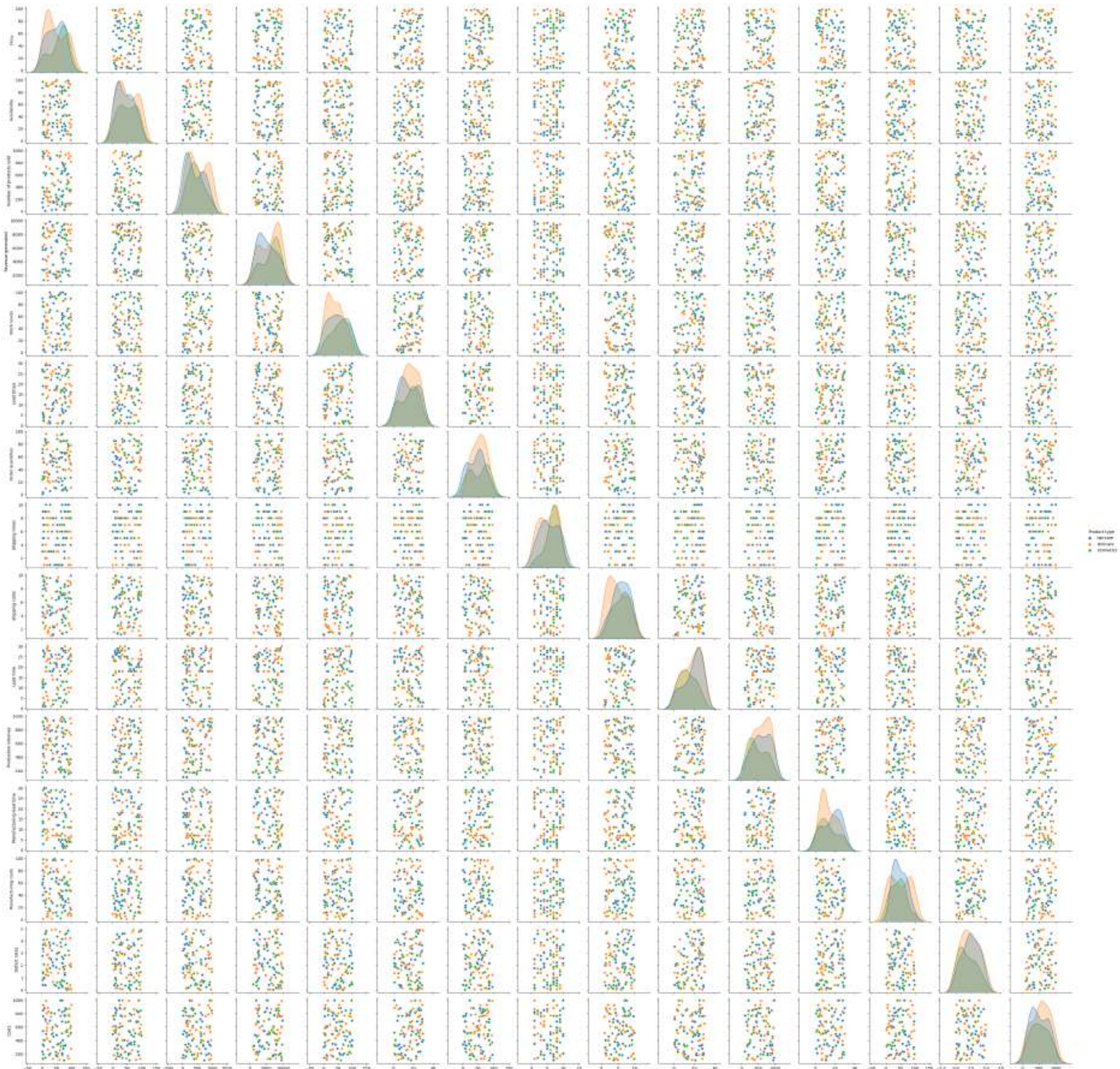
In [7]:
```
data.isna().sum()/len(data)
```

Out[7]:
```
Product type               0.0
SKU                        0.0
Price                      0.0
Availability               0.0
Number of products sold    0.0
Revenue generated          0.0
Customer demographics      0.0
Stock levels               0.0
Lead times                 0.0
Order quantities           0.0
Shipping times             0.0
Shipping carriers          0.0
Shipping costs             0.0
Supplier name              0.0
Location                   0.0
Lead time                  0.0
Production volumes         0.0
Manufacturing lead time    0.0
Manufacturing costs        0.0
Inspection results         0.0
Defect rates               0.0
Transportation modes       0.0
Routes                     0.0
Costs                      0.0
dtype: float64
```

Pairplot of Data Features

In [8]:
```python
sns.pairplot(data,hue='Product type')
```

Out[8]:
```
<seaborn.axisgrid.PairGrid at 0x245cd16e290>
```

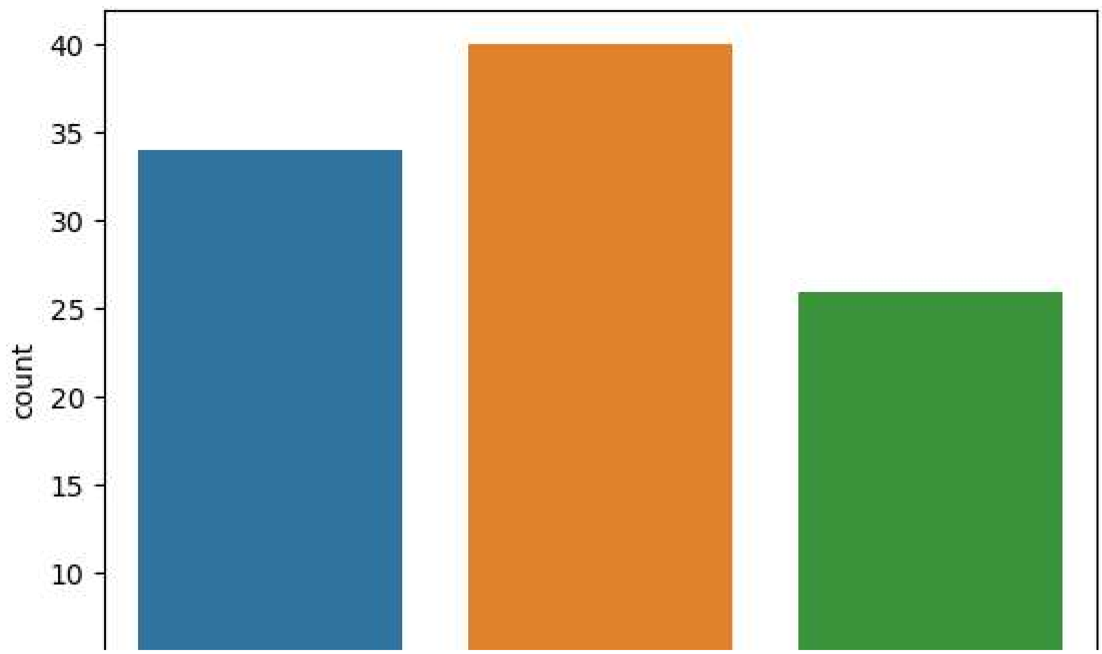


Categorizing Data Variables

In [9]:
```python
categorial= [i for i in data.columns if data [i].dtypes=='object']
numerical= [i for i in data.columns if data [i].dtypes=='object']
```

Countplots for Categorical Variables

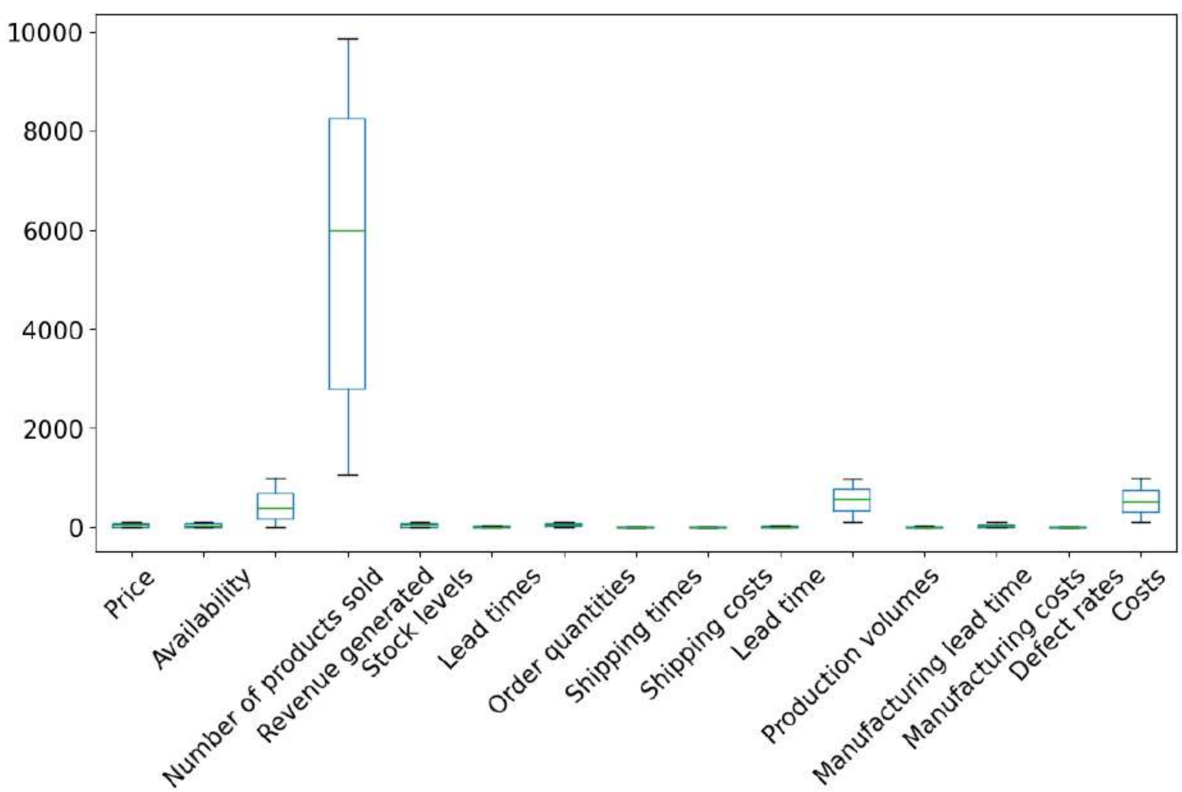
In [10]:
```python
for i in data.select_dtypes(include='object'):
    sns.countplot(data,x=data[i])
    plt.xticks(rotation=90)
    plt.show()
```



Boxplots for Data Variables

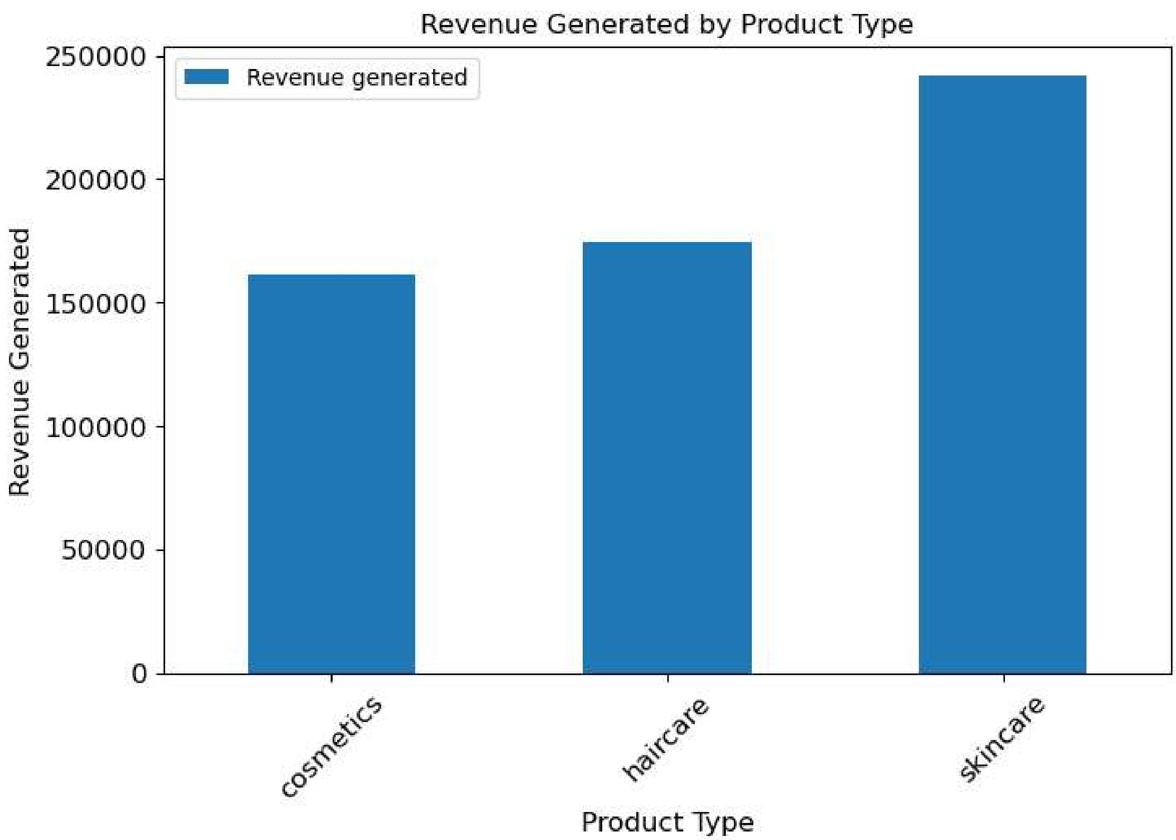
In [11]:
```python
plt.figure(figsize=(12, 6))
data.boxplot(grid=False, rot=45, fontsize=15)
```

Out[11]: <Axes: >



Revenue Analysis by Product Type

In [12]:
```
revenue_by_product = data.groupby(['Product type'])[['Revenue generated']].sum
revenue_by_product.plot(kind='bar', figsize=(8, 5), title="Revenue Generated by
plt.xlabel("Product Type", fontsize=12)
plt.ylabel("Revenue Generated", fontsize=12)
plt.xticks(rotation=45)


plt.show()
```



Top Locations by Revenue

In [13]:
```
revenue_by_location = data.groupby(['Location'])[['Revenue generated']].sum()
revenue_by_location = revenue_by_location.sort_values(by='Revenue generated', a

print(revenue_by_location.head())
```

```
           Revenue generated
Location
Mumbai         137755.026877
Kolkata        137077.551005
Chennai        119142.815748
Bangalore      102601.723882
Delhi           81027.701225
```

Revenue Analysis by Product Type and Location

In [14]:
```
data.groupby(['Product type', 'Location'])[['Revenue generated']].sum()\
.sort_index()\
.sort_values(by='Product type', ascending=False)\
.unstack()\
.style.background_gradient(cmap='winter_r')
```

Out[14]:

	Revenue generated				
Location	**Bangalore**	**Chennai**	**Delhi**	**Kolkata**	**Mumbai**
Product type					
cosmetics	19309.562880	31461.947457	37429.677331	24163.571855	49156.506477
haircare	51654.345696	28723.448932	14625.900767	35027.713247	44423.981964
skincare	31637.815307	58957.419359	28972.123128	77886.265903	44174.538437

Total Revenue by Product Type

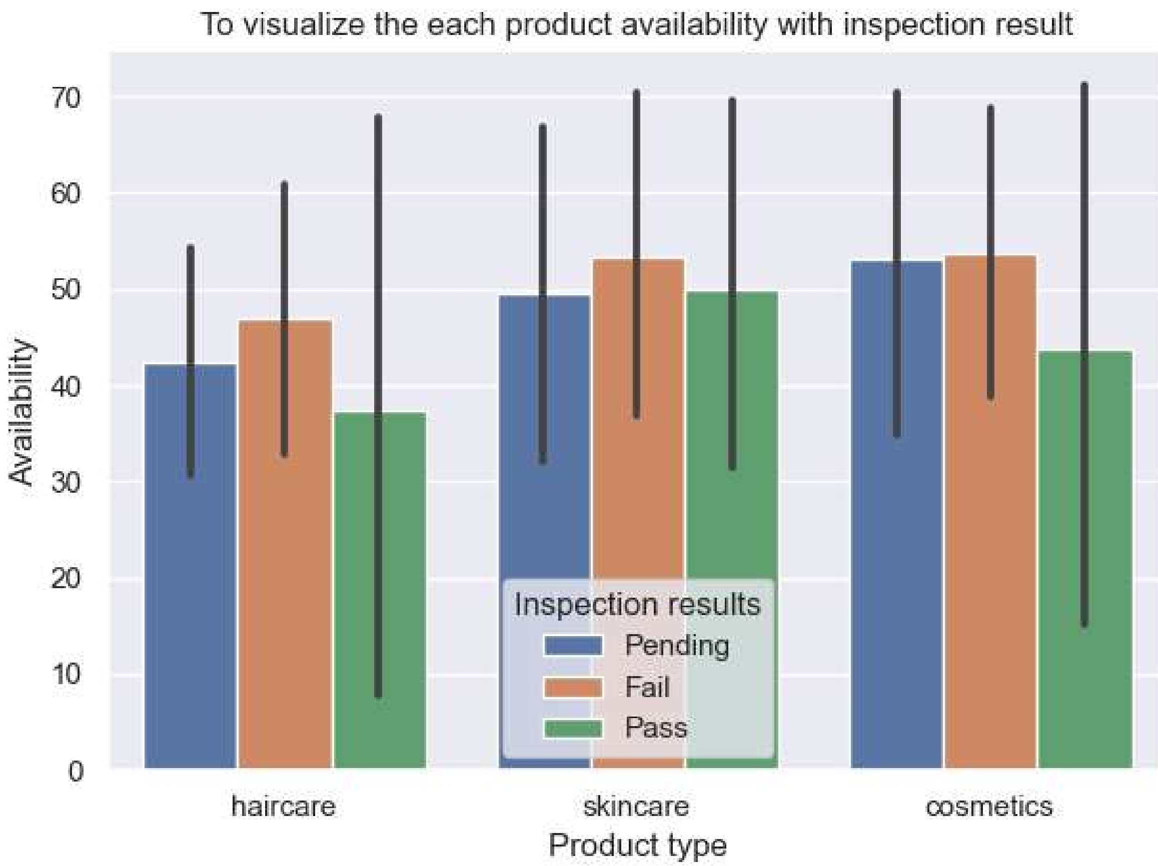
In [15]:
```
data.groupby(['Product type']).sum()\
.sort_index()\
.style.background_gradient(cmap='Dark2_r')
```

Out[15]:

	Price	Availability	Number of products sold	Revenue generated	Stock levels	Lead times	Order quantities	Shipping times
Product type								
cosmetics	1491.387498	1332	11757	161521.265999	1525	400	1343	171
haircare	1564.485482	1471	13611	174455.390605	1644	528	1480	191
skincare	1890.373155	2037	20731	241628.162133	1608	668	2099	213

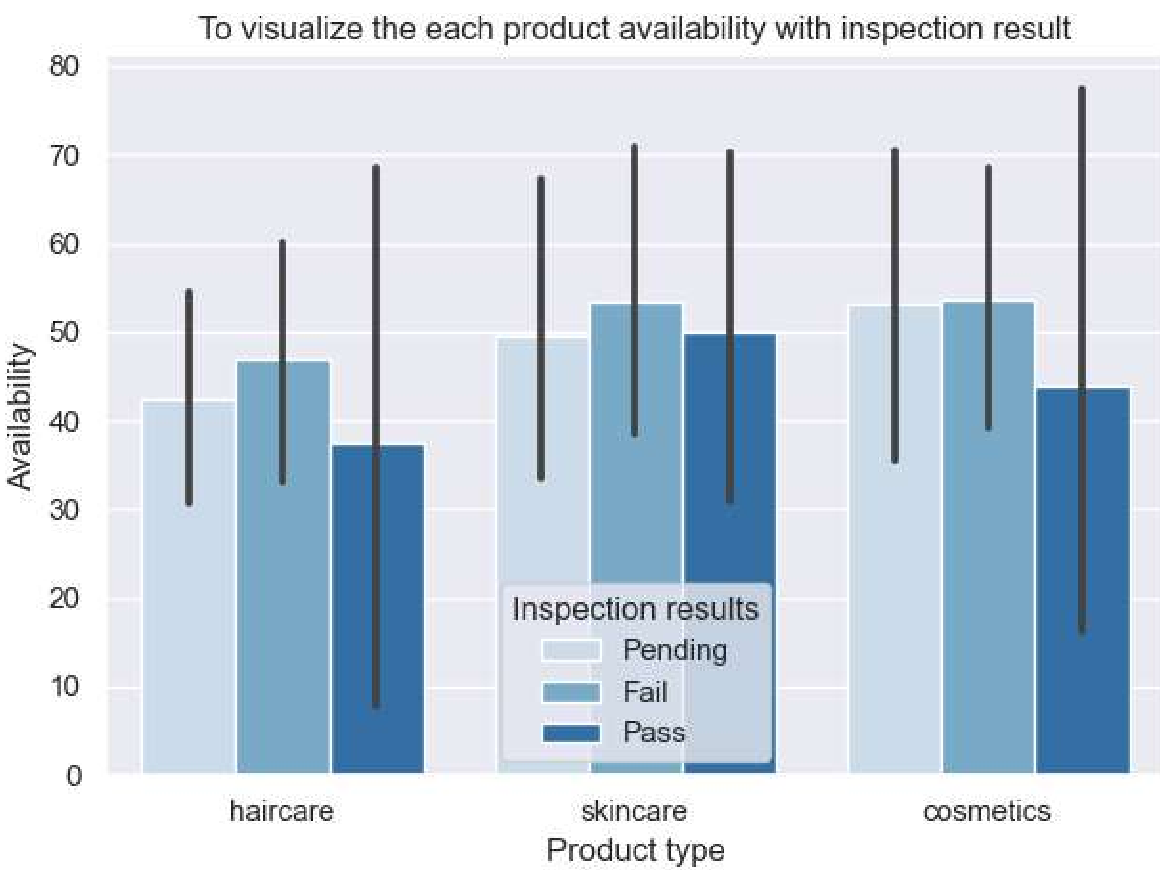
Product Availability by Inspection Results

In [16]:
```
sns.set_theme(context='notebook', style='darkgrid')
sns.barplot(data=data, x='Product type', y='Availability', hue='Inspection resu
plt.title("To visualize the each product availability with inspection result")
plt.tight_layout()
plt.show()
```



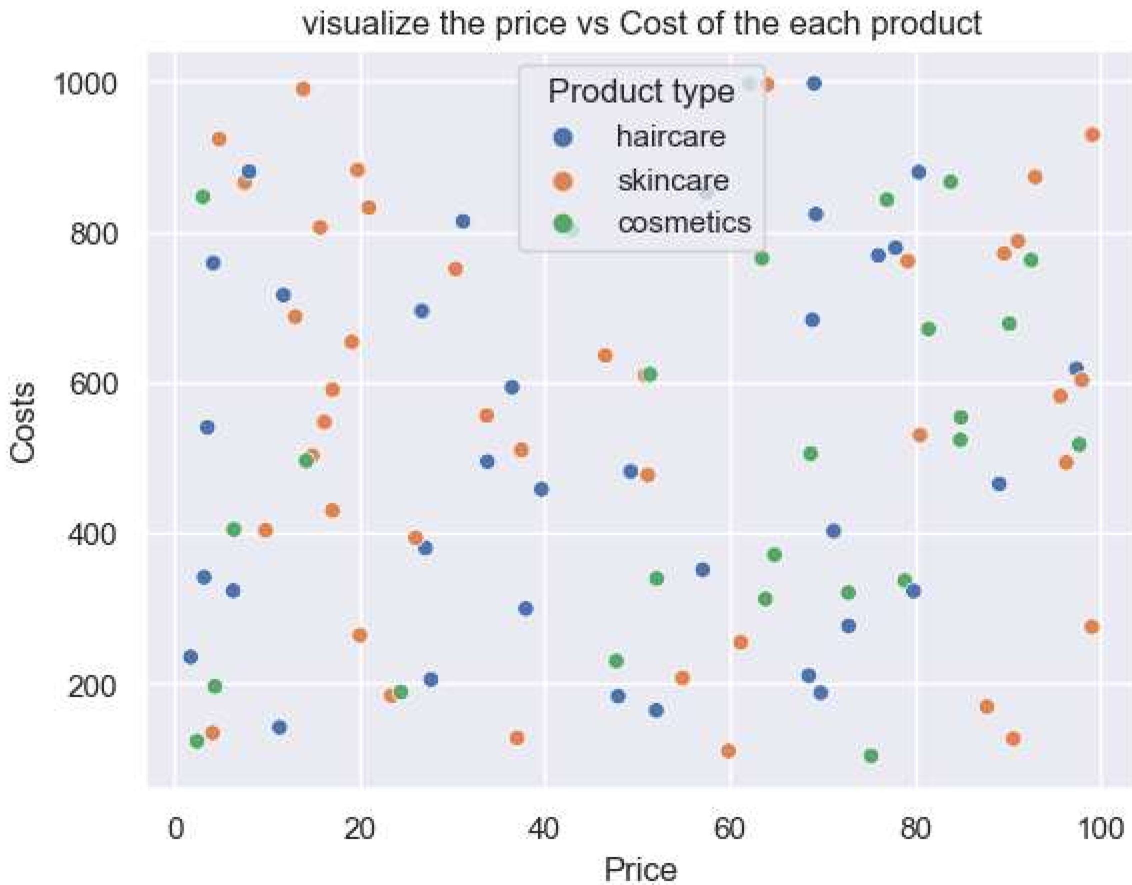
Product Availability by Inspection Results with Custom Palette

In [17]:
```
custom_palette = ['blue'] * len(data['Product type'].unique())

sns.set_theme(context='notebook', style='darkgrid')
sns.barplot(data=data, x='Product type', y='Availability', hue='Inspection resu
plt.title("To visualize the each product availability with inspection result")
plt.tight_layout()
plt.show()
```



Price vs. Costs for Each Product

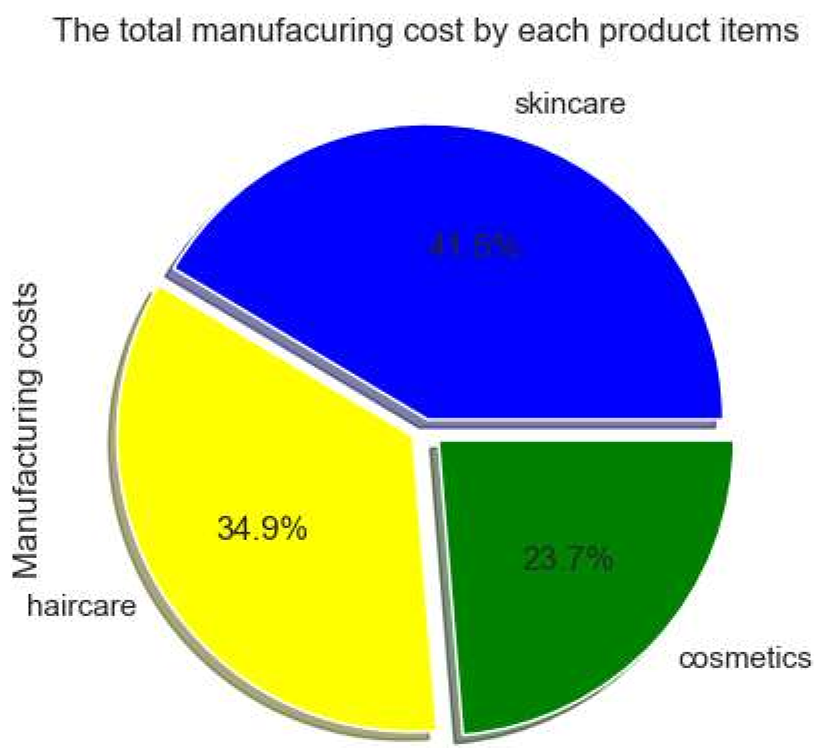
In [18]:
```
sns.set_theme(style='darkgrid')
sns.scatterplot(data=data,x='Price',hue='Product type',y='Costs')
plt.title("visualize the price vs Cost of the each product")
plt.show()
```



Total Manufacturing Costs by Product Type

In [19]:
```
data.groupby(['Product type'])['Manufacturing costs'].sum()\
.sort_values(ascending=False)\
.plot(kind='pie',labels=['skincare','haircare','cosmetics'],autopct='%1.1f%%',
```

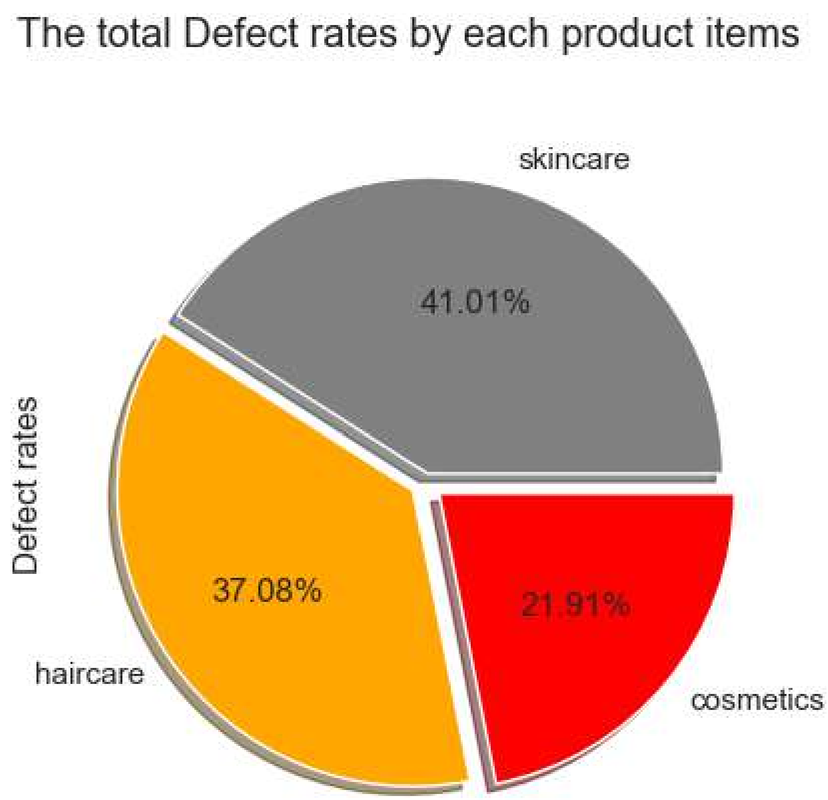
Out[19]: <Axes: title={'center': 'The total manufacuring cost by each product items'}, ylabel='Manufacturing costs'>



Total Defect Rates by Product Type

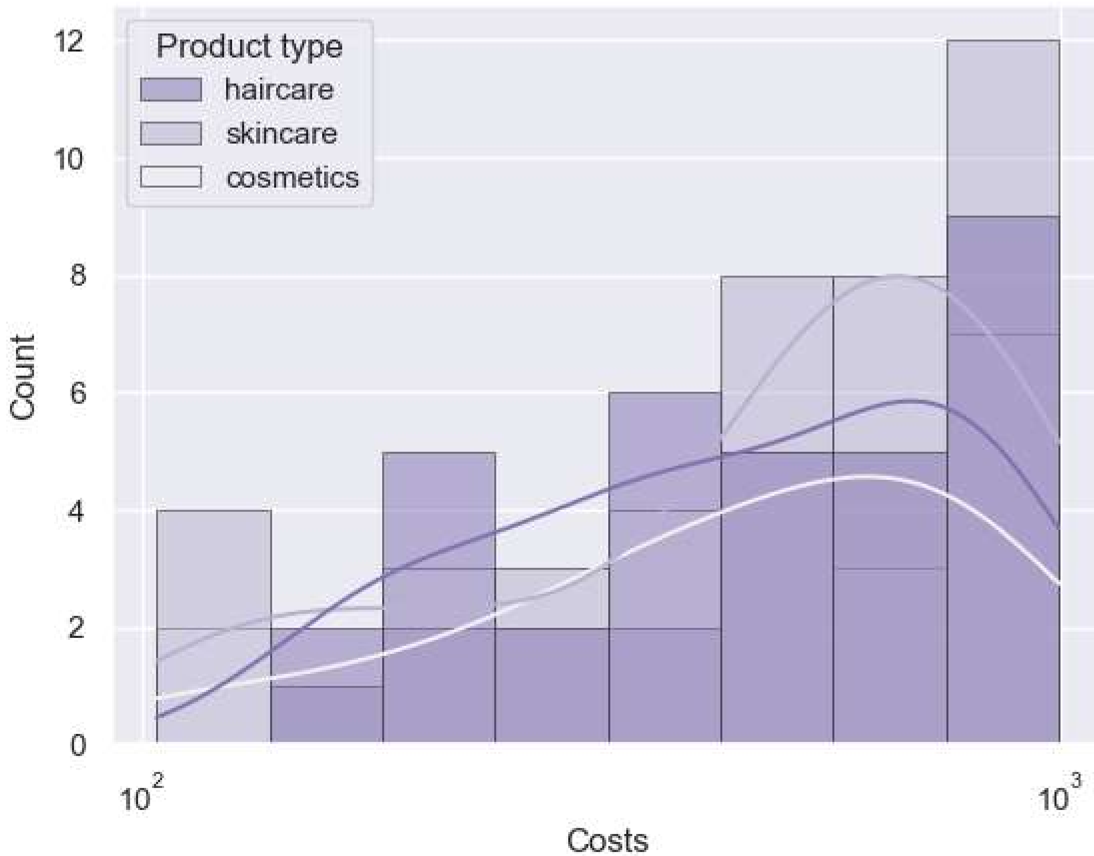
In [20]:
```
data.groupby(['Product type'])['Defect rates'].sum()\
.sort_values(ascending=False)\
.plot(kind='pie',labels=['skincare','haircare','cosmetics'],autopct='%1.2f%%',
```

Out[20]:
```
array([<Axes: ylabel='Defect rates'>], dtype=object)
```



Distribution of Costs by Product Type

In [21]:
```
sns.set_theme(style='darkgrid')
sns.histplot(data=data,x='Costs',hue='Product type',palette="light:m_r",edgecol
plt.show()
```



Pivot Table: Number of Products Sold by Product Type and Location

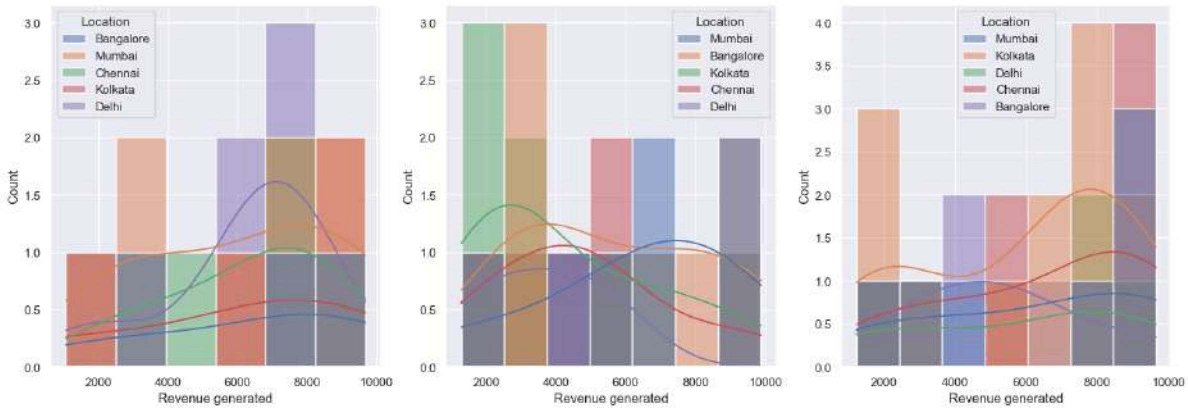
In [22]:
```
pd.pivot_table(data,index='Product type',columns=['Location'],values='Number of
.style.background_gradient(cmap='twilight_shifted_r')
```

Out[22]:

Location Product type	Bangalore	Chennai	Delhi	Kolkata	Mumbai
cosmetics	513.666667	348.600000	667.166667	315.500000	401.000000
haircare	240.000000	386.833333	651.500000	425.875000	445.285714
skincare	286.500000	522.666667	621.200000	623.153846	443.000000

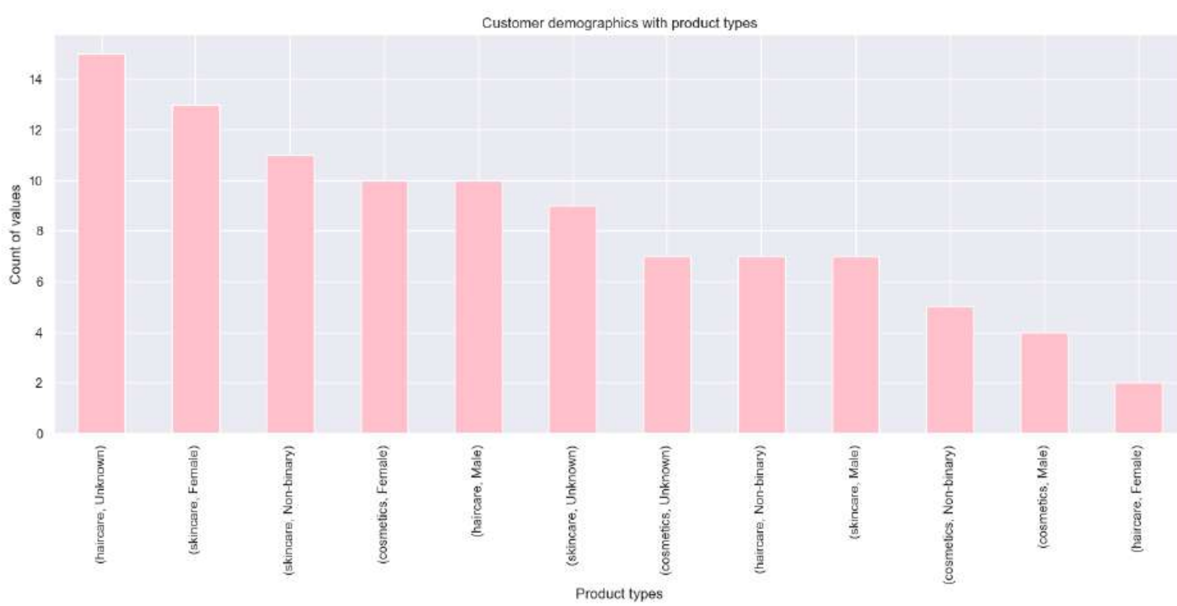
Revenue Distribution by Product Type and Location

In [23]:
```
plt.subplots(1,3,figsize=(25,6))
plt.subplot(141)
ax =sns.histplot(data=data[data['Product type']=='cosmetics'],x='Revenue gener
plt.subplot(142)
ax =sns.histplot(data=data[data['Product type']=='haircare'],x='Revenue genera
plt.subplot(143)
ax =sns.histplot(data=data[data['Product type']=='skincare'],x='Revenue genera
plt.show()
```



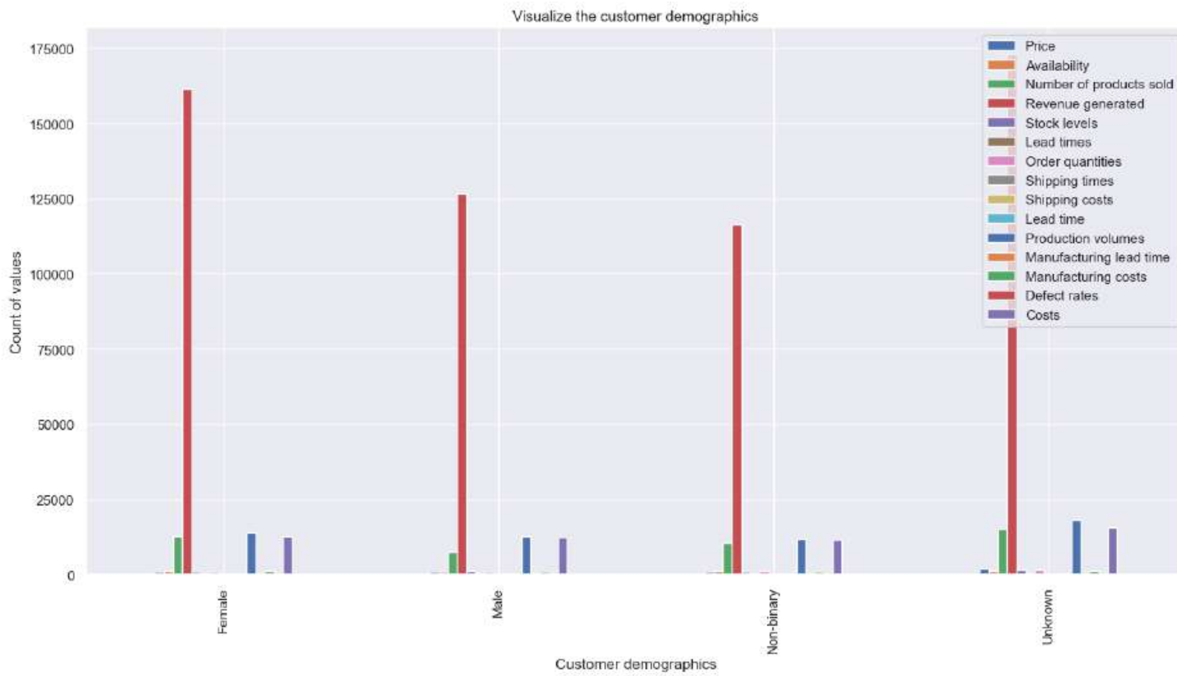
Customer Demographics by Product Type

In [24]:
```
data.groupby(['Product type'])['Customer demographics'].value_counts()\
.sort_index()\
.sort_values(ascending=False)\
.plot(kind='bar',title="Customer demographics with product types",figsize=(17,
plt.xlabel("Product types")
plt.ylabel("Count of values")
plt.show()
```



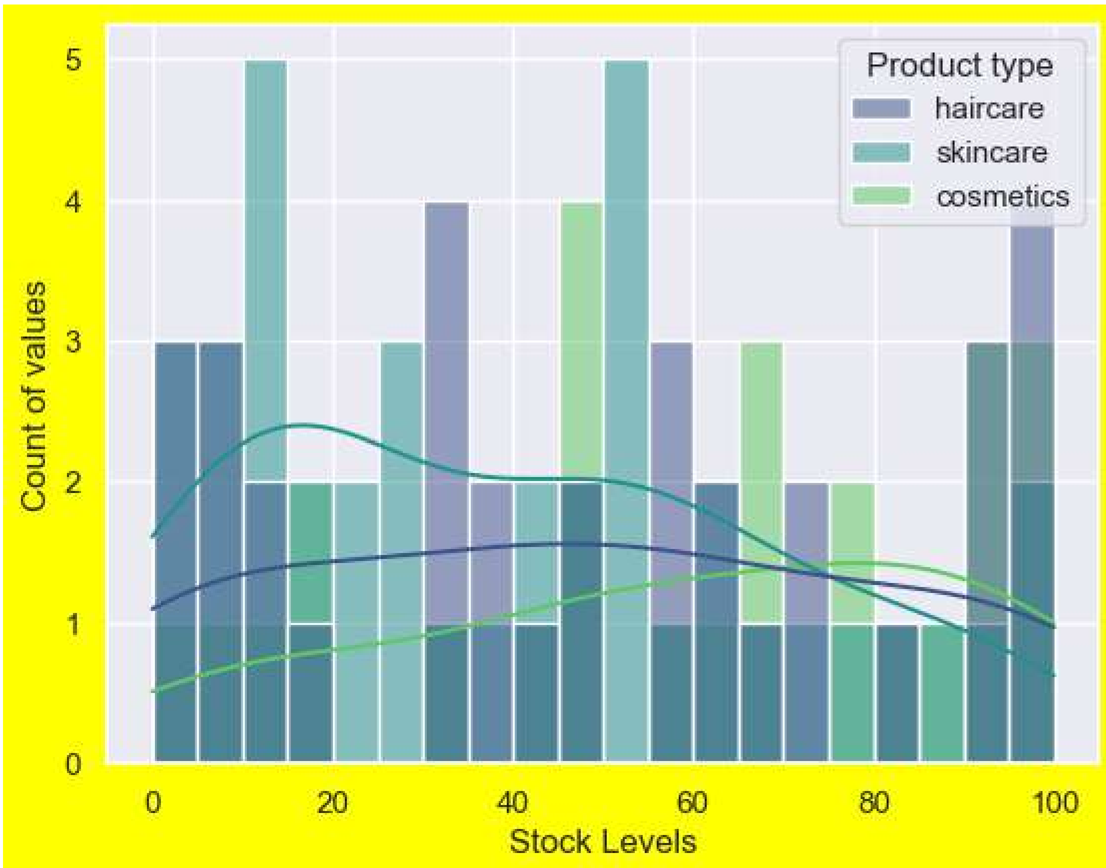
Sum of Data by Customer Demographics

In [25]:
```
data.groupby(['Customer demographics']).sum()\
.sort_index()\
.plot(kind='bar',figsize=(16,8))
plt.title("Visualize the customer demographics")
plt.xlabel("Customer demographics",fontweight=20)
plt.ylabel("Count of values")
plt.show()
```



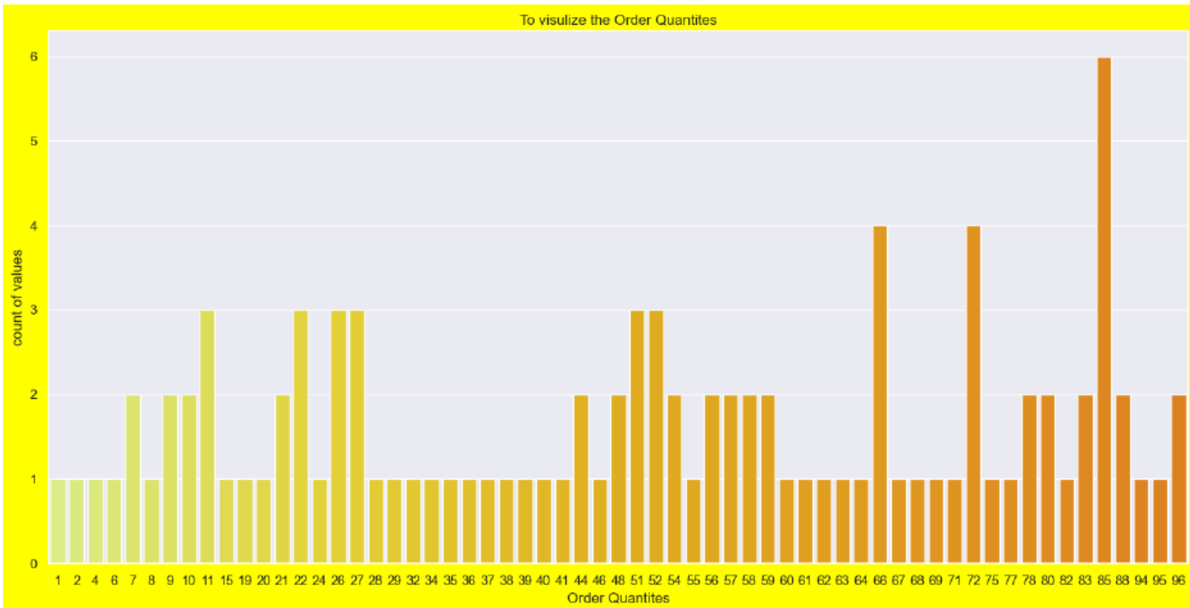
Stock Levels Distribution by Product Type

In [26]:
```
sns.set_theme(style='darkgrid')
plt.rcParams['figure.facecolor']='yellow'
sns.histplot(data=data,x='Stock levels',hue='Product type',bins=20,palette='vi
plt.xlabel("Stock Levels")
plt.ylabel("Count of values")
plt.show()
```



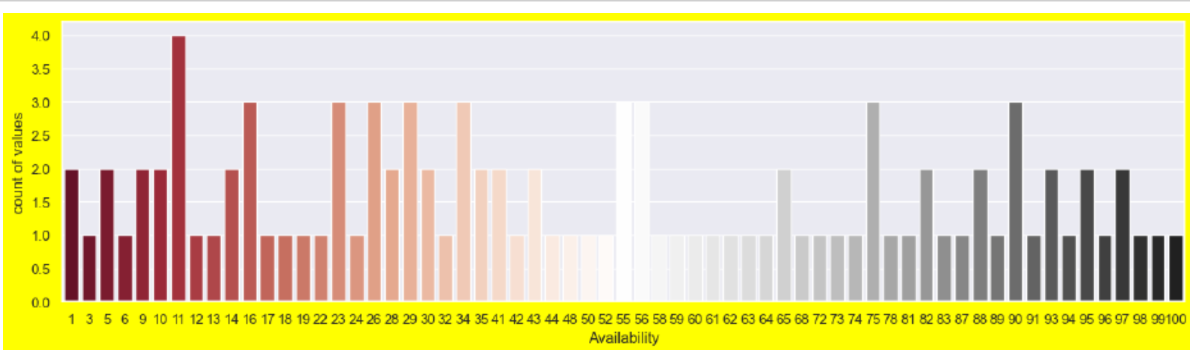
Distribution of Order Quantities

In [27]:
```
plt.figure(figsize=(17,8))
sns.countplot(data=data,x='Order quantities',palette='Wistia')
plt.title("To visulize the Order Quantites")
plt.xlabel("Order Quantites")
plt.ylabel("count of values")
plt.show()
```



Distribution of Availability

In [28]:
```
plt.figure(figsize=(16,4))
sns.countplot(data=data,x='Availability',palette='RdGy')
plt.xlabel("Availability")
plt.ylabel("count of values")
plt.show()
```



In [29]:
```
data.columns
```

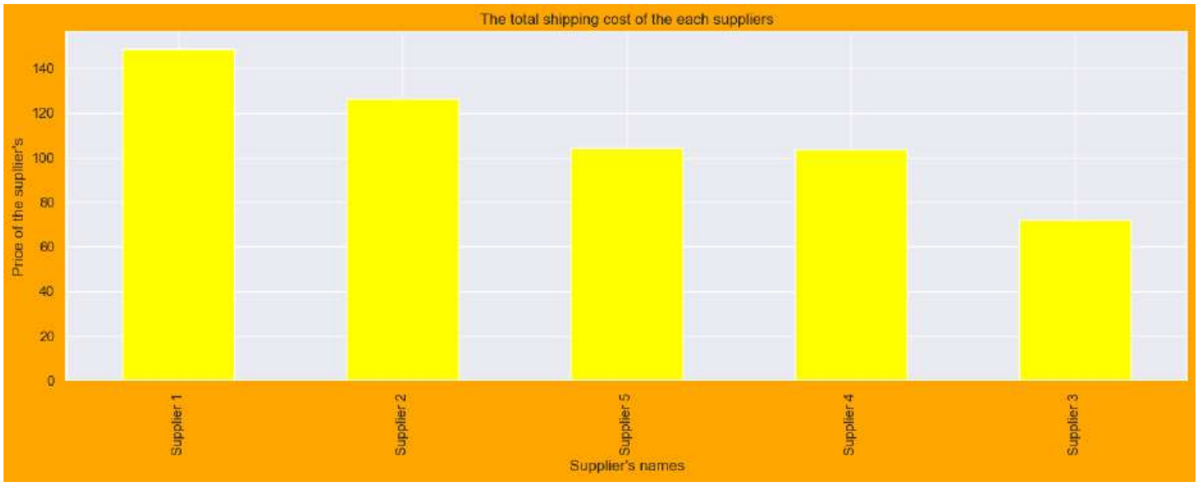
Out[29]:
```
Index(['Product type', 'SKU', 'Price', 'Availability',
       'Number of products sold', 'Revenue generated', 'Customer demographic
s',
       'Stock levels', 'Lead times', 'Order quantities', 'Shipping times',
       'Shipping carriers', 'Shipping costs', 'Supplier name', 'Location',
       'Lead time', 'Production volumes', 'Manufacturing lead time',
       'Manufacturing costs', 'Inspection results', 'Defect rates',
       'Transportation modes', 'Routes', 'Costs'],
      dtype='object')
```

Distribution of Shipping Times

In [30]:
```
plt.rcParams['figure.facecolor']='orange'
data['Shipping times'].value_counts()\
.sort_values(ascending=False)\
.plot(kind='bar',title="Understanding the shipping times",figsize=(16,5),color
plt.xlabel("Time")
plt.ylabel("Count of values")
plt.show()
```

Total Shipping Costs by Supplier

In [31]:
```
data.groupby(['Supplier name'])['Shipping costs'].sum()\
.sort_values(ascending=False)\
.plot(kind='bar',title="The total shipping cost of the each suppliers",figsize=
plt.xlabel("Supplier's names")
plt.ylabel("Price of the supllier's")
plt.show()
```



Total Shipping Costs by Supplier and Location

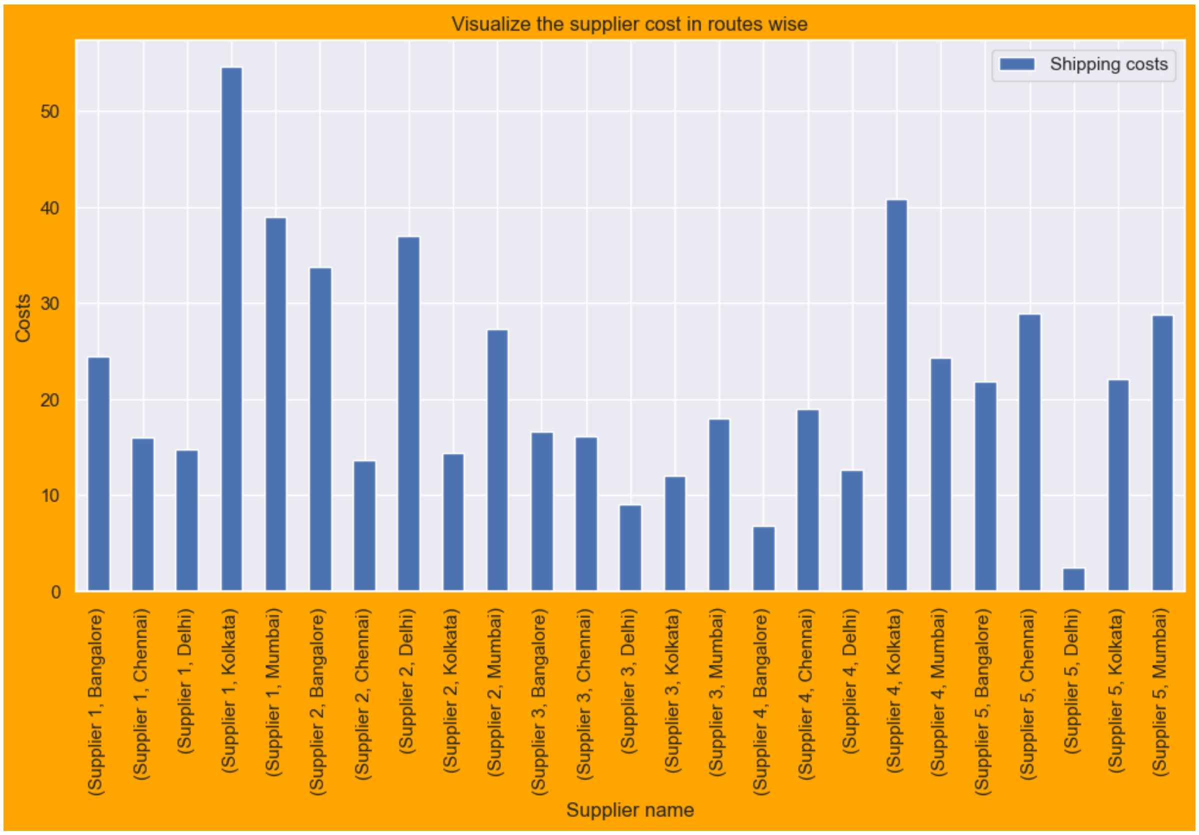
In [32]:
```
data.groupby(['Supplier name','Location'])[['Shipping costs']].sum()\
.sort_index()\
.unstack()\
.style.background_gradient(cmap='Reds')
```

Out[32]:

	Shipping costs				
Location	**Bangalore**	**Chennai**	**Delhi**	**Kolkata**	**Mumbai**
Supplier name					
Supplier 1	24.478536	15.957139	14.777578	54.658678	38.960219
Supplier 2	33.725821	13.692824	37.051131	14.425672	27.366470
Supplier 3	16.628690	16.159605	9.004716	12.012523	18.026025
Supplier 4	6.792438	19.035269	12.709169	40.814152	24.321286
Supplier 5	21.846532	28.936740	2.505621	22.124321	28.803753

Supplier Costs by Location or Routes

In [33]:
```
supplier=data.groupby(['Supplier name','Location'])[['Shipping costs']].sum()
supplier.plot(kind='bar',title="Visualize the supplier cost in routes wise",fi
plt.xlabel("Supplier name")
plt.ylabel("Costs")
plt.show()
```



Shipping Costs by Carriers and Location

In [34]:
```
'''
We use a groupby function with shipping and location and cost columns
\ used for filter and unstack function converted to rows
and finally visualize with background color
'''
data.groupby(['Shipping carriers','Location'])['Costs'].sum()\
.unstack()\
.style.background_gradient(cmap='nipy_spectral')
```

Out[34]:

Location	Bangalore	Chennai	Delhi	Kolkata	Mumbai
Shipping carriers					
Carrier A	4010.034588	3822.282532	2194.619202	2163.042576	1737.092806
Carrier B	5156.304957	4243.420799	3756.959961	5860.229558	3708.528990
Carrier C	1394.381893	4369.309861	2271.988994	4258.472904	3977.908594

Supplier Costs by Routes

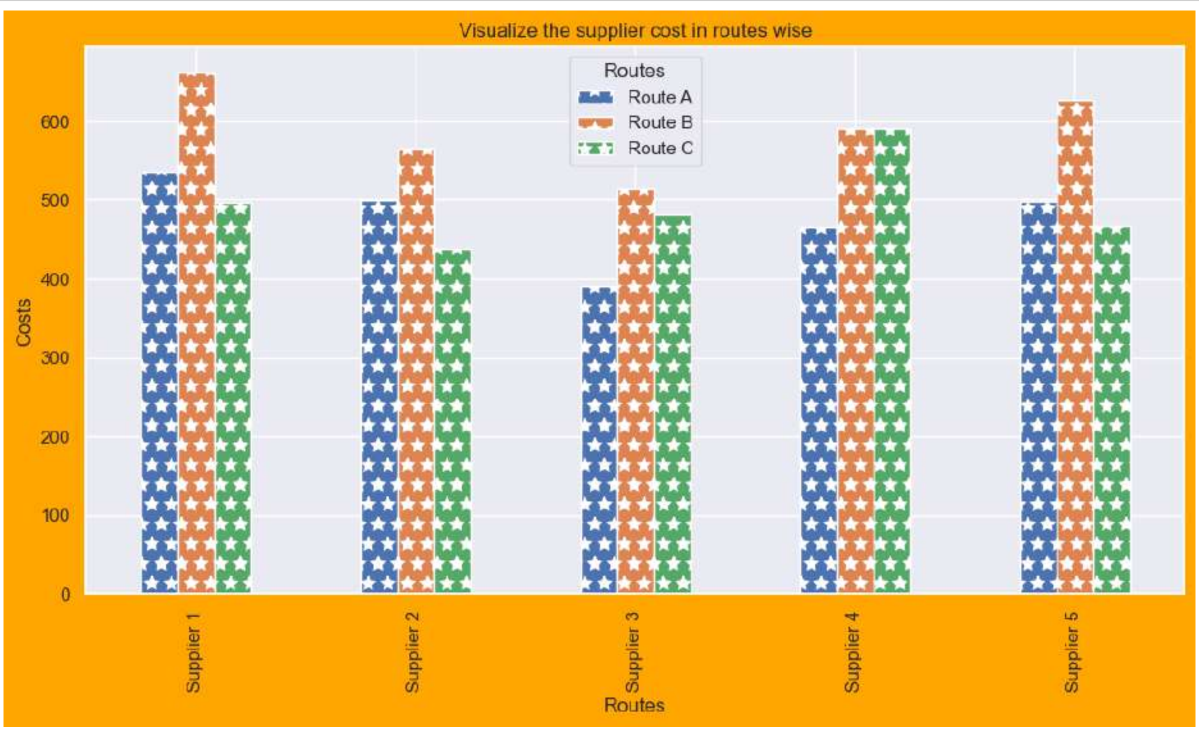
In [35]:
```
'''
We create a pivot tabel for each supliers spend most cost with routes
wise, we take index as supplier name columns routes and
values are costs once we done with visualize with background color

'''
pd.pivot_table(data,index='Supplier name',columns=['Routes'],values='Costs')\
.style.background_gradient(cmap='YlOrBr')
```

Out[35]:

Routes	Route A	Route B	Route C
Supplier name			
Supplier 1	535.451837	661.026634	495.759134
Supplier 2	499.177114	565.627125	438.211349
Supplier 3	391.100859	514.451725	480.441713
Supplier 4	466.373041	590.960046	591.254232
Supplier 5	497.997427	627.270012	467.604074

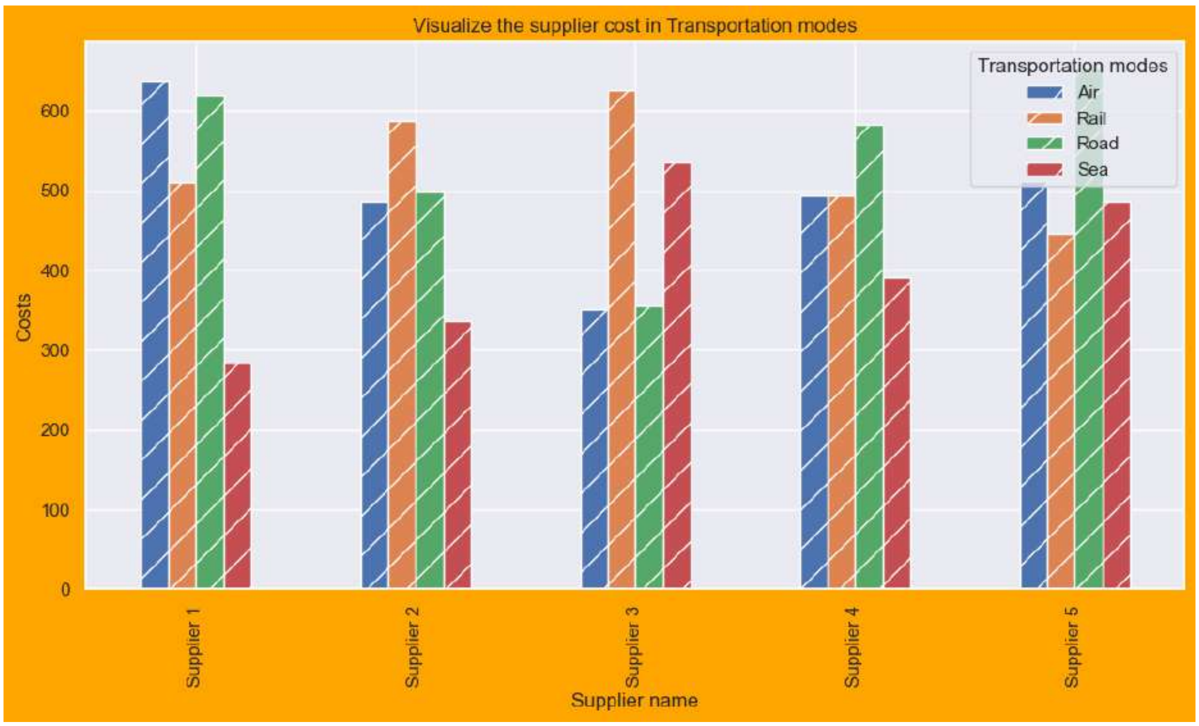
Supplier Costs by Routes Visualization

In [36]:
```
supp=pd.pivot_table(data,index='Supplier name',columns=['Routes'],values='Cost
supp.plot(kind='bar',title="Visualize the supplier cost in routes wise",figsiz
plt.xlabel("Routes")
plt.ylabel("Costs")
plt.show()
```



Supplier Costs by Transportation Modes

In [37]:
```
'''Same as above pivot table but this time we find the cost of each supplies wi
which mean each suplier which transpotaion spend more cost.
'''

trans=pd.pivot_table(data,index='Supplier name',columns=['Transportation modes
trans.plot(kind='bar',title="Visualize the supplier cost in Transportation mode
plt.xlabel("Supplier name")
plt.ylabel("Costs")
plt.show()
```



Project Conclusion and Key Insights

In the course of this project, I have worked extensively on various aspects of our dataset, covering areas such as revenue, costs, product types, suppliers, transportation modes, and customer demographics. Through data visualization and in-depth analysis, I have unearthed invaluable insights that can guide strategic decision-making and operational improvements.

Key Insights

I have gained a deeper understanding of our supply chain, identifying which suppliers, products, and transportation modes have the most significant impact on our costs and revenue.

An examination of customer demographics and product preferences has provided a clearer picture of our target audience.

The visualization of availability, order quantities, and defect rates has enriched my understanding of product performance.

Route-specific and transportation mode-specific cost analyses have highlighted areas for potential cost savings.

The distribution of stock levels, shipping times, and order quantities has offered insights into inventory management and order fulfillment.

As I bring this project to a close, it is essential to underscore the central role of data analysis in informed decision-making. The findings and insights generated in this project can serve as a foundation for future strategies and improvements in various aspects of our operations. After wholeheartedly dedicating my time and expertise to this project I am delighted to unveil the concluding description and headings for my portfolio.