Exploratory Data Analysis - Laptops Pricing dataset

Objectives

- Visualize individual feature patterns
- Run descriptive statistical analysis on the dataset
- Use groups and pivot tables to find the effect of categorical variables on price
- Use Pearson Correlation to measure the interdependence between variables

Setup

For this lab, we will be using the following libraries:

- `skillsnetwork` for downloading the data
- `pandas` (https://pandas.pydata.org/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMML0187ENSkillsNetwork31430127-2021-01-01) for managing the data.
- `numpy` (https://numpy.org/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMML0187ENSkillsNetwork31430127-2021-01-01) for mathematical operations.
- `scipy` (https://docs.scipy.org/doc/scipy/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMML0187ENSkillsNetwork31430127-2021-01-01) for statistical operations.
- `seaborn` (https://seaborn.pydata.org/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMML0187ENSkillsNetwork31430127-2021-01-01) for visualizing the data.
- `matplotlib` (https://matplotlib.org/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMML0187ENSkillsNetwork31430127-2021-01-01) for additional plotting tools.

Install Required Libraries

Importing Required Libraries

We import all required libraries in one place:

In [35]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
%matplotlib inline
```

Import the dataset

We will use the modified pre-processed version of the data set from the last lab : laptop_pricing_cleaned.csv.

In [36]:
```
file_name = "laptop_pricing_cleaned.csv"
```

Import the file to a pandas dataframe.

In [37]:
```
df = pd.read_csv(file_name, header=0)
```

Print the first 5 entries of the dataset to confirm loading.

In [38]:
```
df.head(10)
```

Out[38]:

	Manufacturer	Category	GPU	OS	CPU_core	Screen_Size_inch	CPU_frequency	RAM_GB	Storage_GB_SSD	Weight_pounds	Price	Pri binr
0	Acer	4	2	1	5	14.0	0.551724	8	256	3.52800	978	L
1	Dell	3	1	1	3	15.6	0.689655	4	256	4.85100	634	L
2	Dell	3	1	1	7	15.6	0.931034	8	256	4.85100	946	L
3	Dell	4	2	1	5	13.3	0.551724	8	128	2.69010	1244	L
4	HP	4	2	1	7	15.6	0.620690	8	256	4.21155	837	L
5	Dell	3	1	1	5	15.6	0.551724	8	256	4.85100	1016	L
6	HP	3	3	1	5	15.6	0.551724	8	256	4.63050	1117	L
7	Acer	3	2	1	5	15.0	0.551724	4	256	4.85100	866	L
8	Dell	3	1	1	5	15.6	0.862069	4	256	5.07150	812	L
9	Acer	3	3	1	7	15.0	0.620690	8	256	4.85100	1068	L

In [39]:
```
df.dtypes
```

Out[39]:
```
Manufacturer         object
Category              int64
GPU                   int64
OS                    int64
CPU_core              int64
Screen_Size_inch    float64
CPU_frequency       float64
RAM_GB                int64
Storage_GB_SSD        int64
Weight_pounds       float64
Price                 int64
Price-binned         object
Screen-Full_HD        int64
Screen-IPS_panel      int64
dtype: object
```

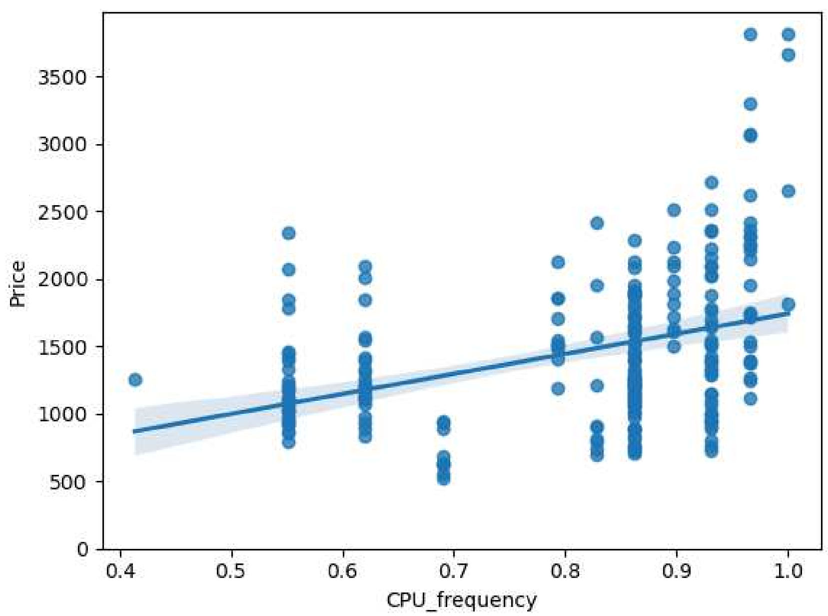
Task 1 - Visualize individual feature patterns

Continuous valued features

Generate regression plots for each of the parameters "CPU_frequency", "Screen_Size_inch" and "Weight_pounds" against "Price". Also, print the value of correlation of each feature with "Price".

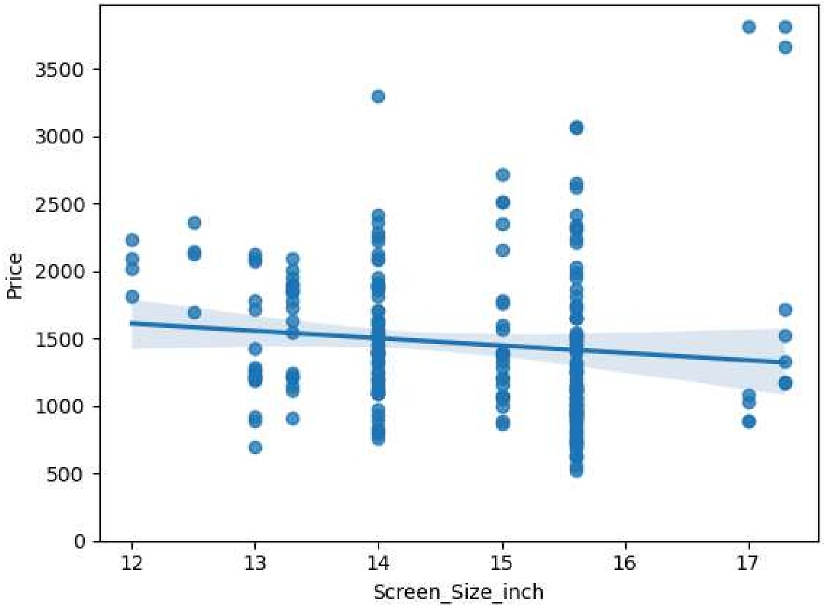
In [40]:
```
# CPU_frequency plot
sns.regplot(x="CPU_frequency", y="Price", data = df)
plt.ylim(0,)
```

Out[40]:
```
(0.0, 3974.15)
```



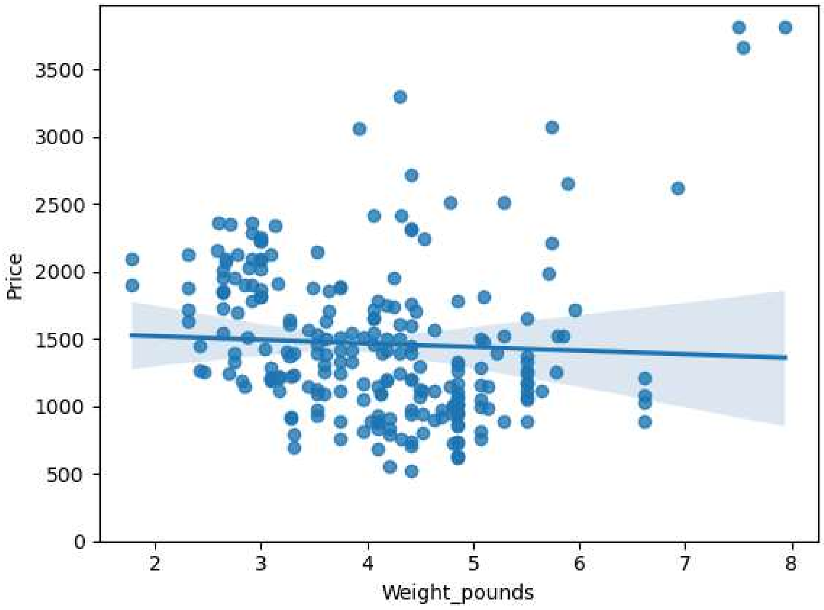
In [41]:
```
# Screen_Size_inch plot
sns.regplot( x= "Screen_Size_inch", y="Price", data=df)
plt.ylim(0,)
```

Out[41]: (0.0, 3974.15)



In [42]:
```
# Weight_pounds plot
sns.regplot(x="Weight_pounds", y="Price", data=df)
plt.ylim(0,)
```

Out[42]: (0.0, 3974.15)



In [43]:
```
# Correlation values of the three attributes with Price
df_corr = df[['CPU_frequency','Screen_Size_inch','Weight_pounds','Price']]
df_corr.corr()
```

Out[43]:

	CPU_frequency	Screen_Size_inch	Weight_pounds	Price
CPU_frequency	1.000000	-0.000948	0.066522	0.366666
Screen_Size_inch	-0.000948	1.000000	0.797534	-0.110644
Weight_pounds	0.066522	0.797534	1.000000	-0.050312
Price	0.366666	-0.110644	-0.050312	1.000000

In [44]:
```
# List of attributes of interest
attributes = ['CPU_frequency', 'Screen_Size_inch', 'Weight_pounds']

# Calculate correlations with 'Price'
correlations = df[attributes + ['Price']].corr()['Price'][attributes]
print(correlations)
```

```
CPU_frequency       0.366666
Screen_Size_inch   -0.110644
Weight_pounds      -0.050312
Name: Price, dtype: float64
```

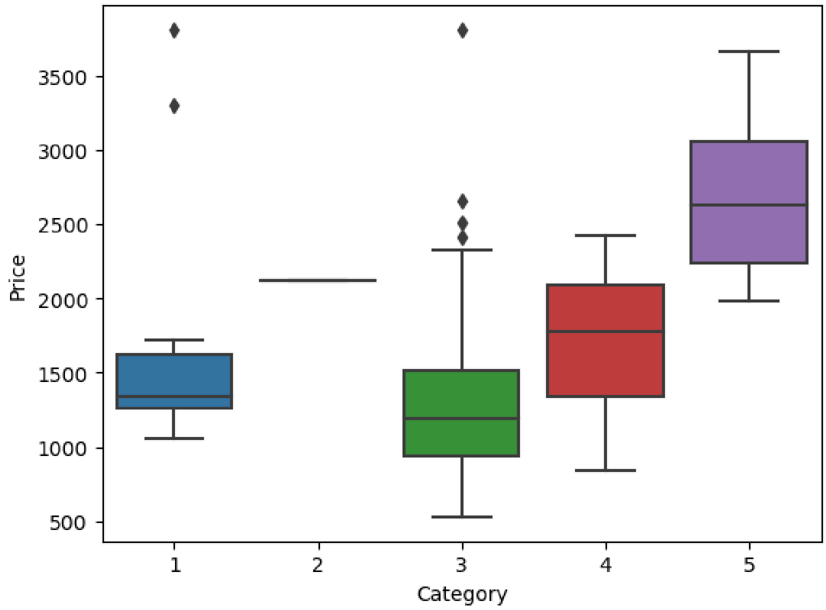
Interpretation: "CPU_frequency" has a 36% positive correlation with the price of the laptops. The other two parameters have weak correlation with price.

Categorical features

Generate Box plots for the different feature that hold categorical values. These features would be "Category", "GPU", "OS", "CPU_core", "RAM_GB", "Storage_GB_SSD"

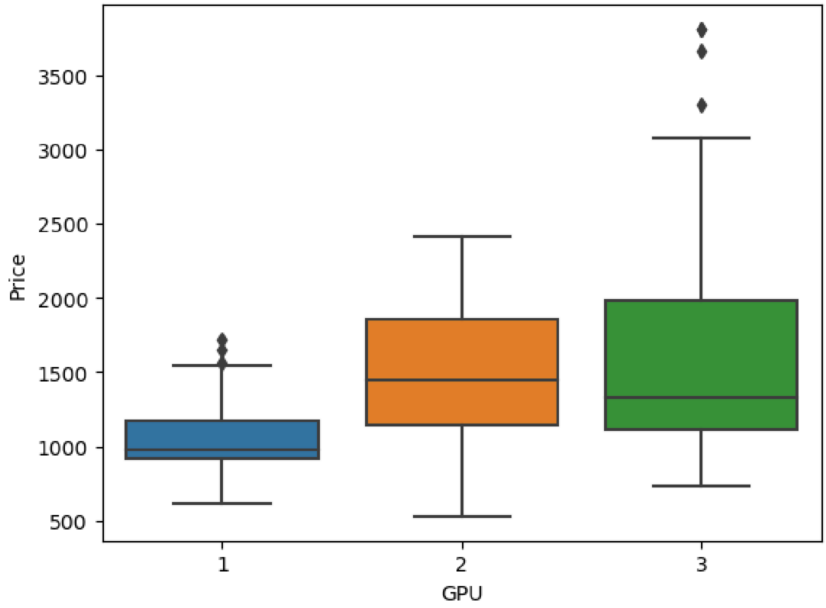
In [46]:
```
# Category Box plot
sns.boxplot(x="Category", y="Price", data=df)
```

Out[46]: <Axes: xlabel='Category', ylabel='Price'>



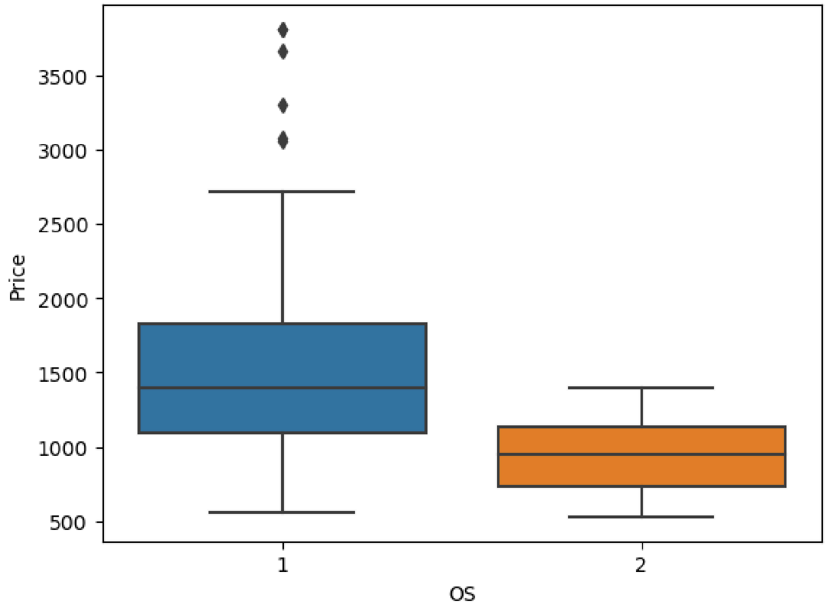
In [47]:
```
# GPU Box plot
sns.boxplot(x="GPU", y="Price", data=df)
```

Out[47]: <Axes: xlabel='GPU', ylabel='Price'>



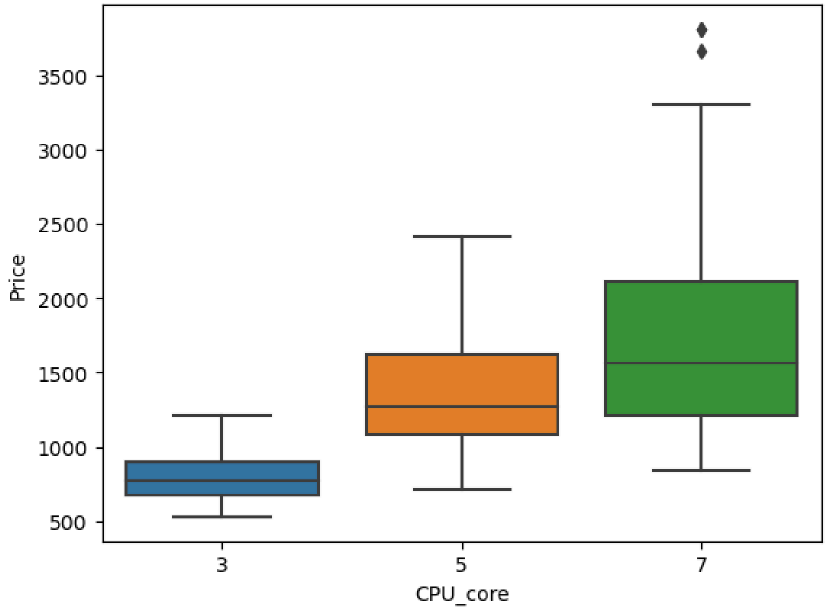
In [48]:
```
# OS Box plot
sns.boxplot(x="OS", y="Price", data=df)
```

Out[48]: <Axes: xlabel='OS', ylabel='Price'>



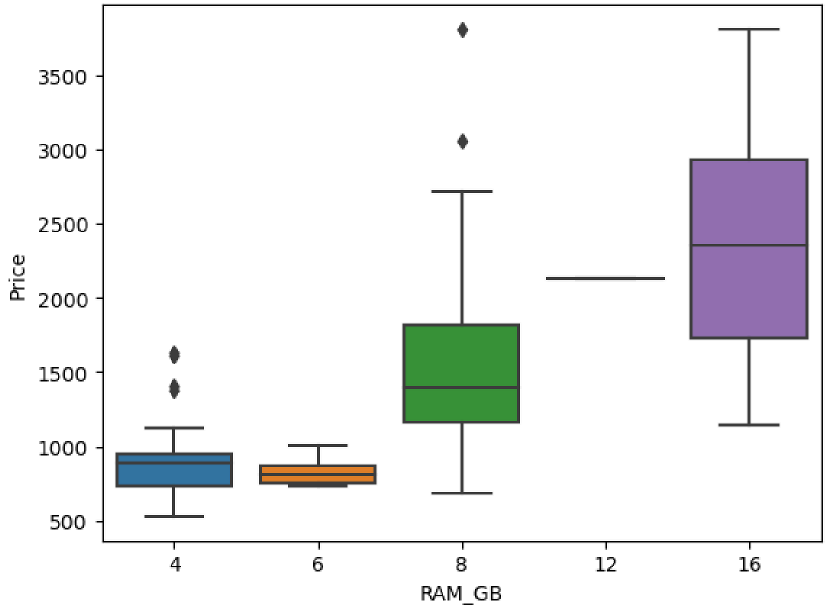
In [49]:
```
# CPU_core Box plot
sns.boxplot(x="CPU_core", y="Price", data=df)
```

Out[49]: `<Axes: xlabel='CPU_core', ylabel='Price'>`



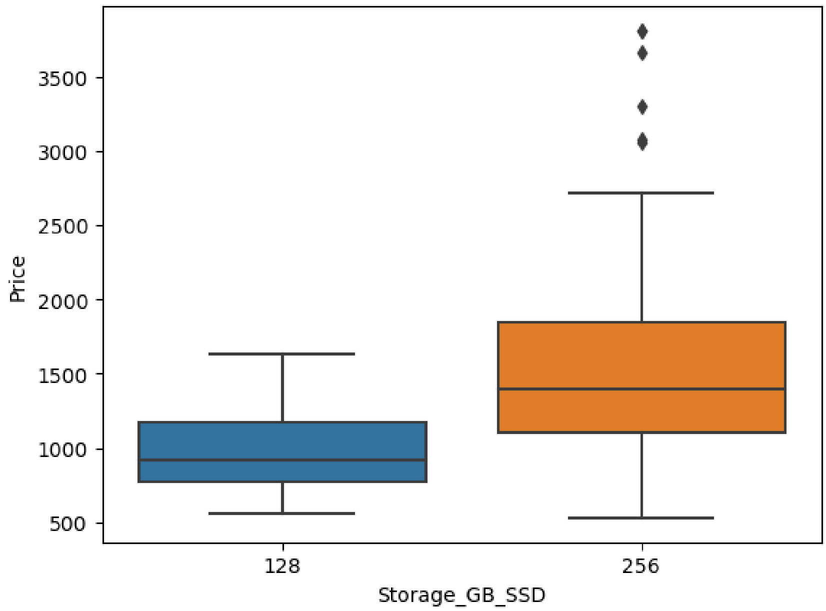
In [50]:
```
# RAM_GB Box plot
sns.boxplot(x="RAM_GB", y="Price", data=df)
```

Out[50]: `<Axes: xlabel='RAM_GB', ylabel='Price'>`



In [51]:
```python
# Storage_GB_SSD Box plot
sns.boxplot(x="Storage_GB_SSD", y="Price", data=df)
```

Out[51]: <Axes: xlabel='Storage_GB_SSD', ylabel='Price'>



Task 2 - Descriptive Statistical Analysis

Generate the statistical description of all the features being used in the data set. Include "object" data types as well.

In [53]:
```python
df.describe(include=['object'])
```

Out[53]:

	Manufacturer	Price-binned
count	238	238
unique	11	3
top	Dell	Low
freq	71	160

In [54]:
```python
df.describe()
```

Out[54]:

	Category	GPU	OS	CPU_core	Screen_Size_inch	CPU_frequency	RAM_GB	Storage_GB_SSD	Weight_pounds	
count	238.000000	238.000000	238.000000	238.000000	238.000000	238.000000	238.000000	238.000000	238.000000	2
mean	3.205882	2.151261	1.058824	5.630252	14.688655	0.813822	7.882353	245.781513	4.106221	14
std	0.776533	0.638282	0.235790	1.241787	1.166045	0.141860	2.482603	34.765316	1.078442	5
min	1.000000	1.000000	1.000000	3.000000	12.000000	0.413793	4.000000	128.000000	1.786050	5
25%	3.000000	2.000000	1.000000	5.000000	14.000000	0.689655	8.000000	256.000000	3.246863	10
50%	3.000000	2.000000	1.000000	5.000000	15.000000	0.862069	8.000000	256.000000	4.106221	13
75%	4.000000	3.000000	1.000000	7.000000	15.600000	0.931034	8.000000	256.000000	4.851000	17
max	5.000000	3.000000	2.000000	7.000000	17.300000	1.000000	16.000000	256.000000	7.938000	38

Task 3 - GroupBy and Pivot Tables

Group the parameters "GPU", "CPU_core" and "Price" to make a pivot table and visualize this connection using the pcolor plot.

In [56]:
```python
# Create the group
df_groupe = df[["GPU", "CPU_core", "Price"]]
df_grouped = df_groupe.groupby(["GPU", "CPU_core"], as_index=False).mean()
```

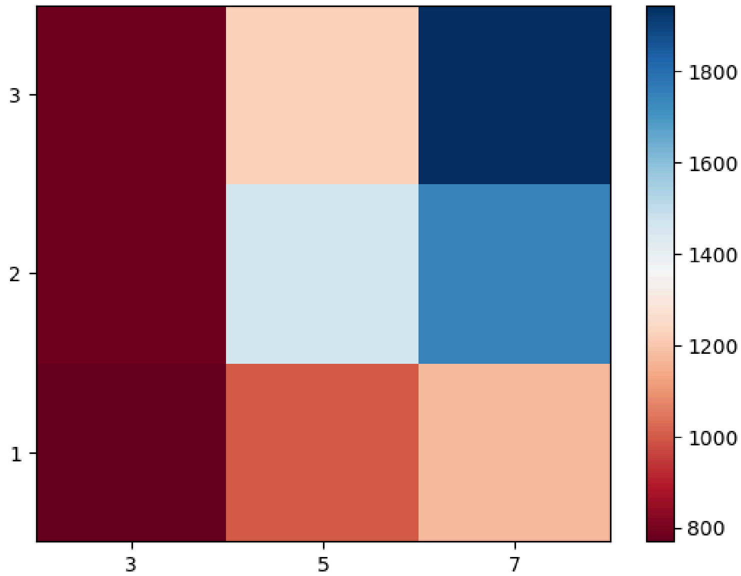
In [58]:
```
# Create the Pivot table
grouped_pivot = df_grouped.pivot(index="GPU", columns="CPU_core")
grouped_pivot
```

Out[58]:

	Price		
CPU_core	**3**	**5**	**7**
GPU			
1	769.250000	998.500000	1167.941176
2	785.076923	1462.197674	1744.621622
3	784.000000	1220.680000	1945.097561

In [59]:
```
# Create the Plot
fig, ax = plt.subplots()
im = ax.pcolor(grouped_pivot, cmap='RdBu')

#label names
row_labels = grouped_pivot.columns.levels[1]
col_labels = grouped_pivot.index

#move ticks and labels to the center
ax.set_xticks(np.arange(grouped_pivot.shape[1]) + 0.5, minor=False)
ax.set_yticks(np.arange(grouped_pivot.shape[0]) + 0.5, minor=False)

#insert labels
ax.set_xticklabels(row_labels, minor=False)
ax.set_yticklabels(col_labels, minor=False)

fig.colorbar(im)
```

Out[59]: <matplotlib.colorbar.Colorbar at 0x2d21925e510>



Task 4 - Pearson Correlation and p-values

Use the `scipy.stats.pearsonr()` function to evaluate the Pearson Coefficient and the p-values for each parameter tested above. This will help you determine the parameters most likely to have a strong effect on the price of the laptops.

In [61]:
```
for param in ['RAM_GB','CPU_frequency','Storage_GB_SSD','Screen_Size_inch','Weight_pounds','CPU_core','OS','GPU',
    pearson_coef, p_value = stats.pearsonr(df[param], df['Price'])
    print(param)
    print("The Pearson Correlation Coefficient for ",param," is", pearson_coef, " with a P-value of P =", p_value
```

```
RAM_GB
The Pearson Correlation Coefficient for  RAM_GB  is 0.5492972971857844  with a P-value of P = 3.681560628842868e
-20
CPU_frequency
The Pearson Correlation Coefficient for  CPU_frequency  is 0.36666555832636644  with a P-value of P = 5.50246368
9008642e-09
Storage_GB_SSD
The Pearson Correlation Coefficient for  Storage_GB_SSD  is 0.2434207552181029  with a P-value of P = 0.00014898
923191724174
Screen_Size_inch
The Pearson Correlation Coefficient for  Screen_Size_inch  is -0.11064420817118263  with a P-value of P = 0.0885
3397846830766
Weight_pounds
The Pearson Correlation Coefficient for  Weight_pounds  is -0.050312258377008784  with a P-value of P = 0.439769
3853479999
CPU_core
The Pearson Correlation Coefficient for  CPU_core  is 0.4593977773355115  with a P-value of P = 7.91295012700903
4e-14
OS
The Pearson Correlation Coefficient for  OS  is -0.22172980114827384  with a P-value of P = 0.000569664255924674
9
GPU
The Pearson Correlation Coefficient for  GPU  is 0.2882981988881428  with a P-value of P = 6.166949698364282e-06
Category
The Pearson Correlation Coefficient for  Category  is 0.28624275581264125  with a P-value of P = 7.2256962358067
33e-06
```