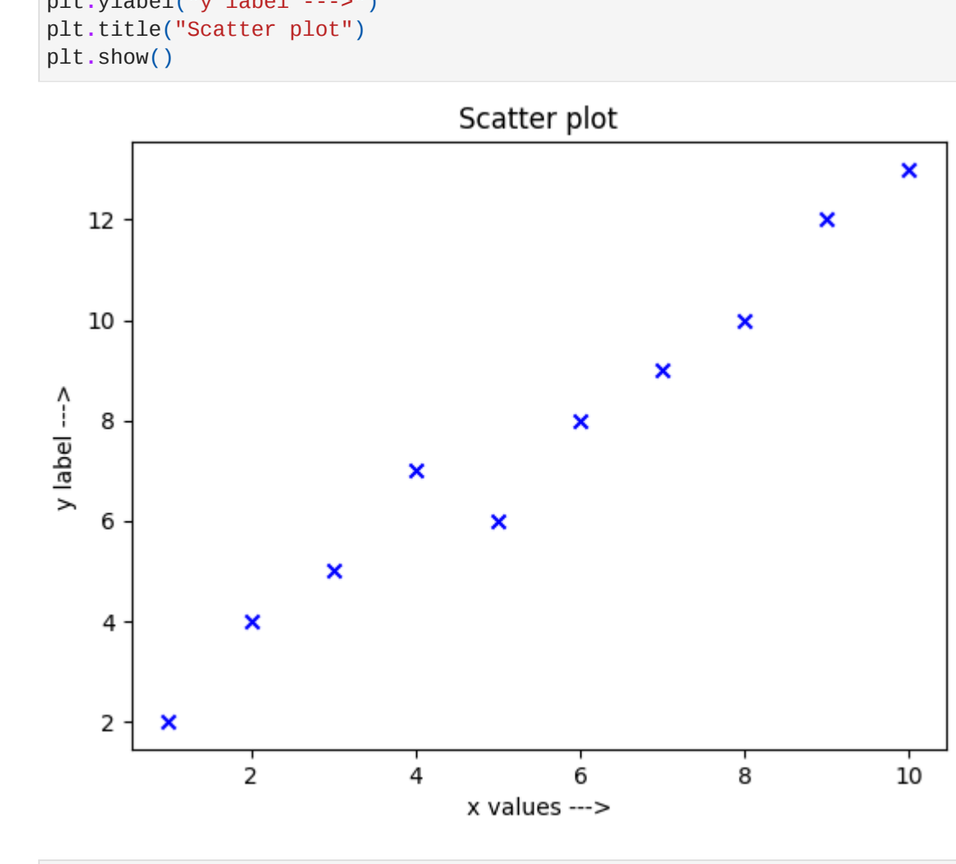
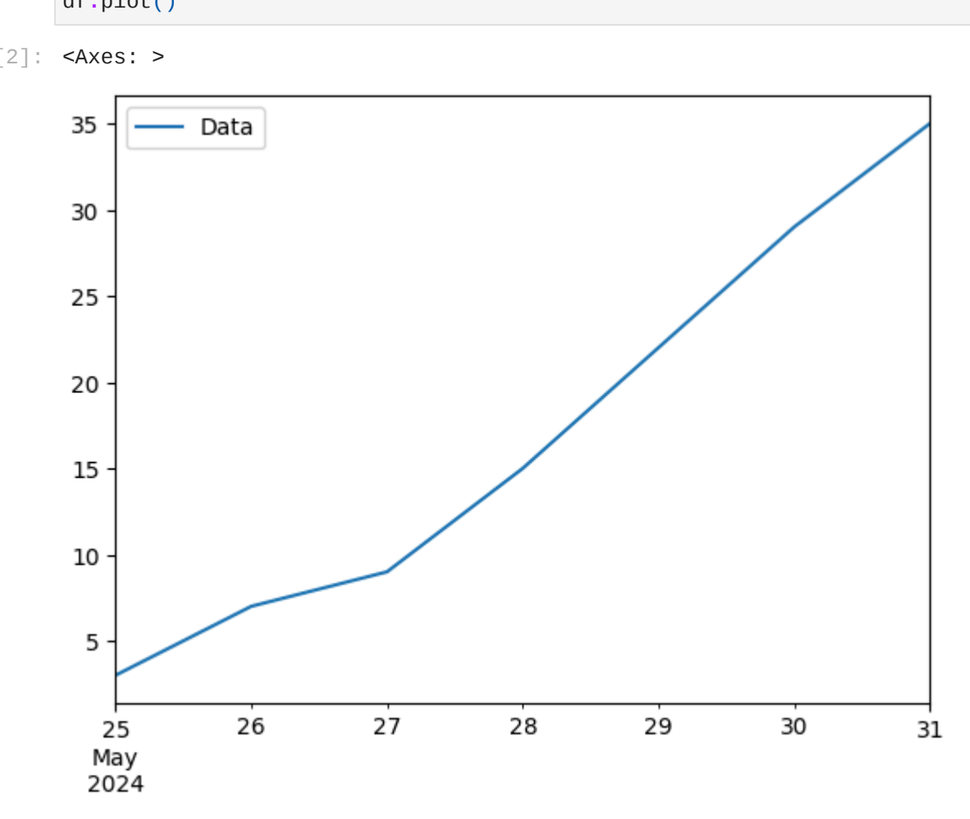
Matplotlib Assignment

In [1]:
```python
#Q1. Create a scatter plot using Matplotlib to visualize the relationship between two arrays, x and y for the given
# data.
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
y = [2, 4, 5, 7, 6, 8, 9, 10, 12, 13]
plt.scatter(x,y, c='b',marker='x')
plt.xlabel("x values --->")
plt.ylabel("y label --->")
plt.title("Scatter plot")
plt.show()
```

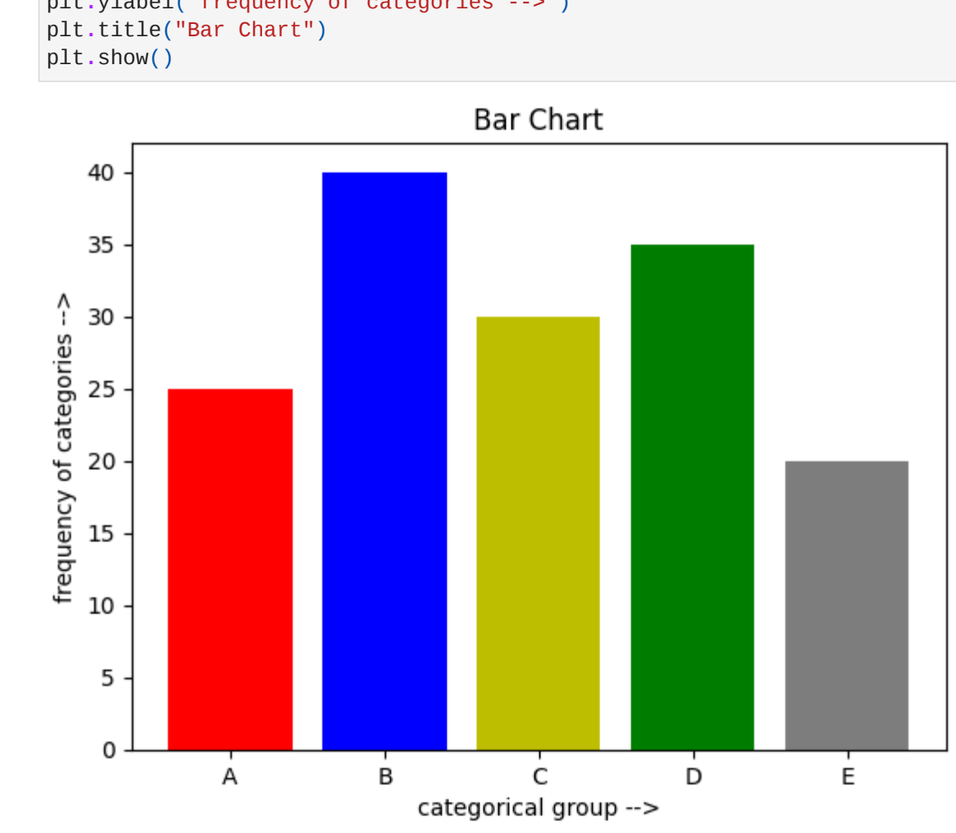


In [2]:
```python
#Q2.  Generate a line plot to visualize the trend of values for the given data.
data = np.array([3, 7, 9, 15, 22, 29, 35])
df=pd.DataFrame(data,columns=["Data"],index=pd.date_range('2024-05-25',periods=7))
df.plot()
```

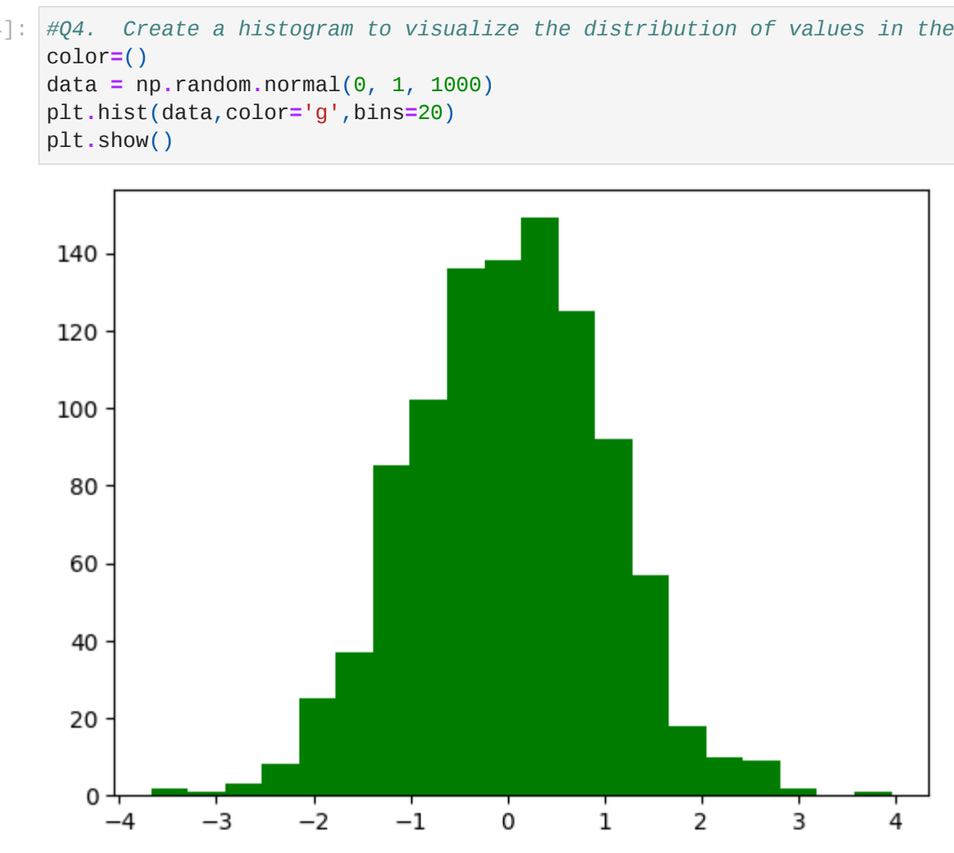
Out[2]: <Axes: >



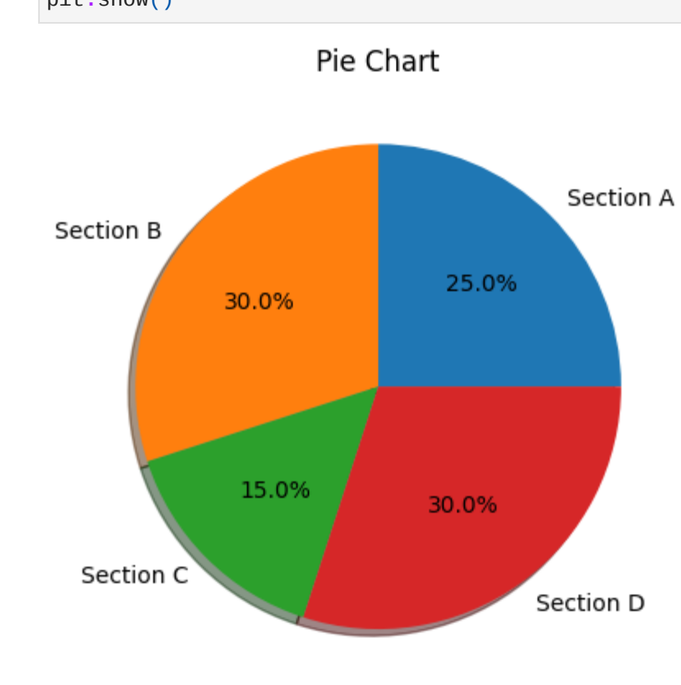
In [3]:
```python
#Q3.  Display a bar chart to represent the frequency of each item in the given array categories.
color=('r','b','y','g','grey')
categories = ['A', 'B', 'C', 'D', 'E']
values = [25, 40, 30, 35, 20]
plt.bar(categories,values,color=color)
plt.xlabel("categorical group -->")
plt.ylabel("frequency of categories -->")
plt.title("Bar Chart")
plt.show()
```



In [4]:
```python
#Q4.  Create a histogram to visualize the distribution of values in the array data.
color=()
data = np.random.normal(0, 1, 1000)
plt.hist(data,color='g',bins=20)
plt.show()
```



In [5]:
```python
#Q5.  Show a pie chart to represent the percentage distribution of different sections in the array `sections`.
sections = ['Section A', 'Section B', 'Section C', 'Section D']
sizes = [25, 30, 15, 30]
color=('r','b','y','g','grey')
plt.pie(sizes,labels=sections, shadow=True,autopct='%1.1f%%')
plt.title("Pie Chart")
plt.show()
```

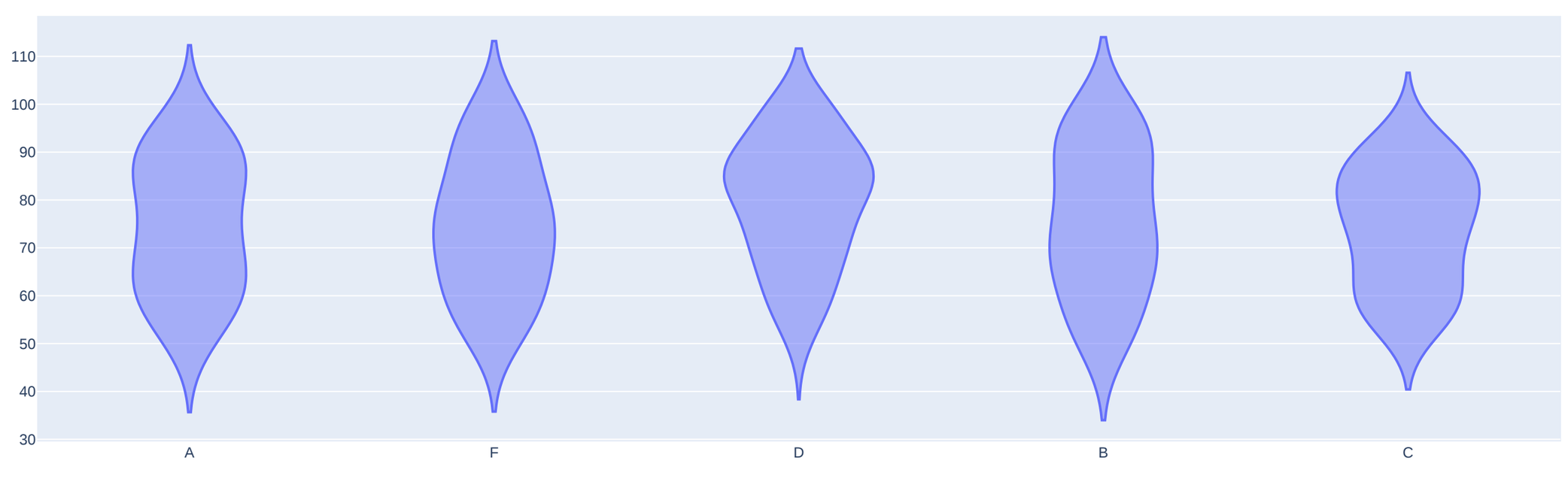


In [ ]:

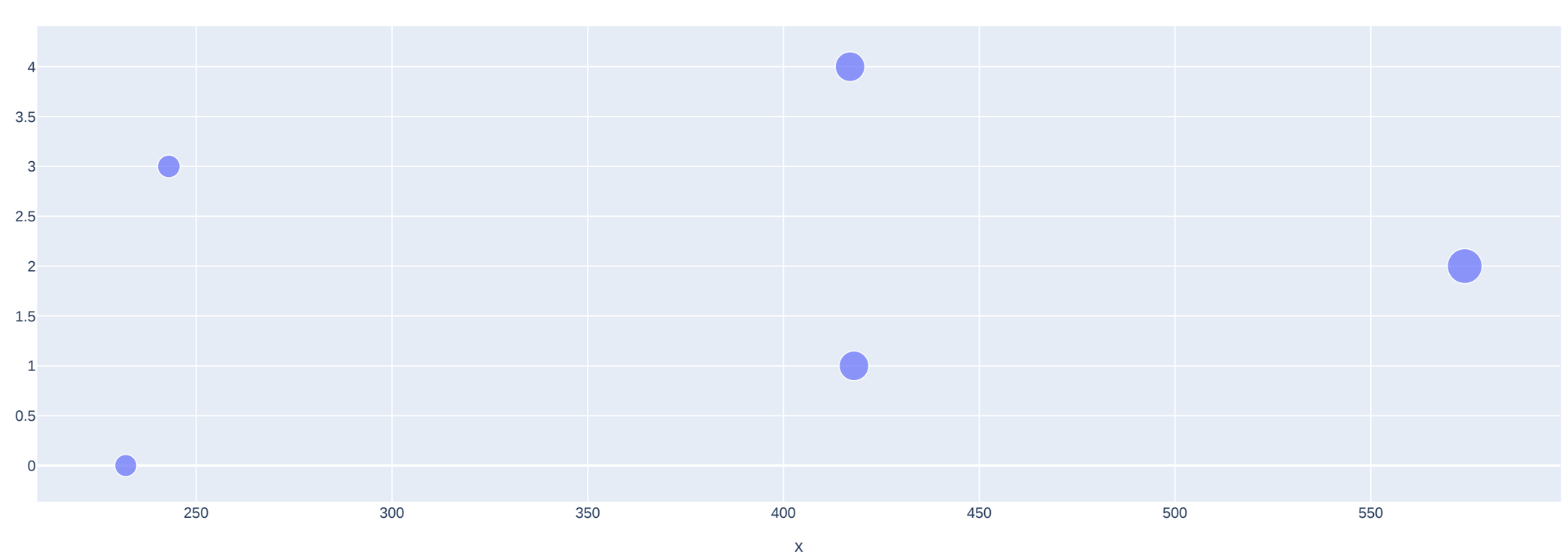
Plotly Assignment

In [6]:
```python
#Q1. Using the given dataset, to generate a 3D scatter plot to visualize the distribution of data points in a three
# dimensional space.
import plotly.graph_objects as go
import plotly.express as px
np.random.seed(30)
data = {
 'X': np.random.uniform(-10, 10, 300),
 'Y': np.random.uniform(-10, 10, 300),
 'Z': np.random.uniform(-10, 10, 300)
 }

df = pd.DataFrame(data)
df
fig=go.Figure()
fig.add_trace(go.Scatter3d(x=df['X'],y=df['Y'],z=df['Z'],mode='markers'))
fig.show()
```

In [7]:
```python
#Q2.  Using the Student Grades, create a violin plot to display the distribution of scores across different grade
# categories
np.random.seed(15)
data = {
 'Grade': np.random.choice(['A', 'B', 'C', 'D', 'F'], 200),
 'Score': np.random.randint(50, 100, 200)
 }
df=pd.DataFrame(data)
df
fig=go.Figure()
fig.add_trace(go.Violin(x=df['Grade'],y=df['Score']))
fig.show()
```

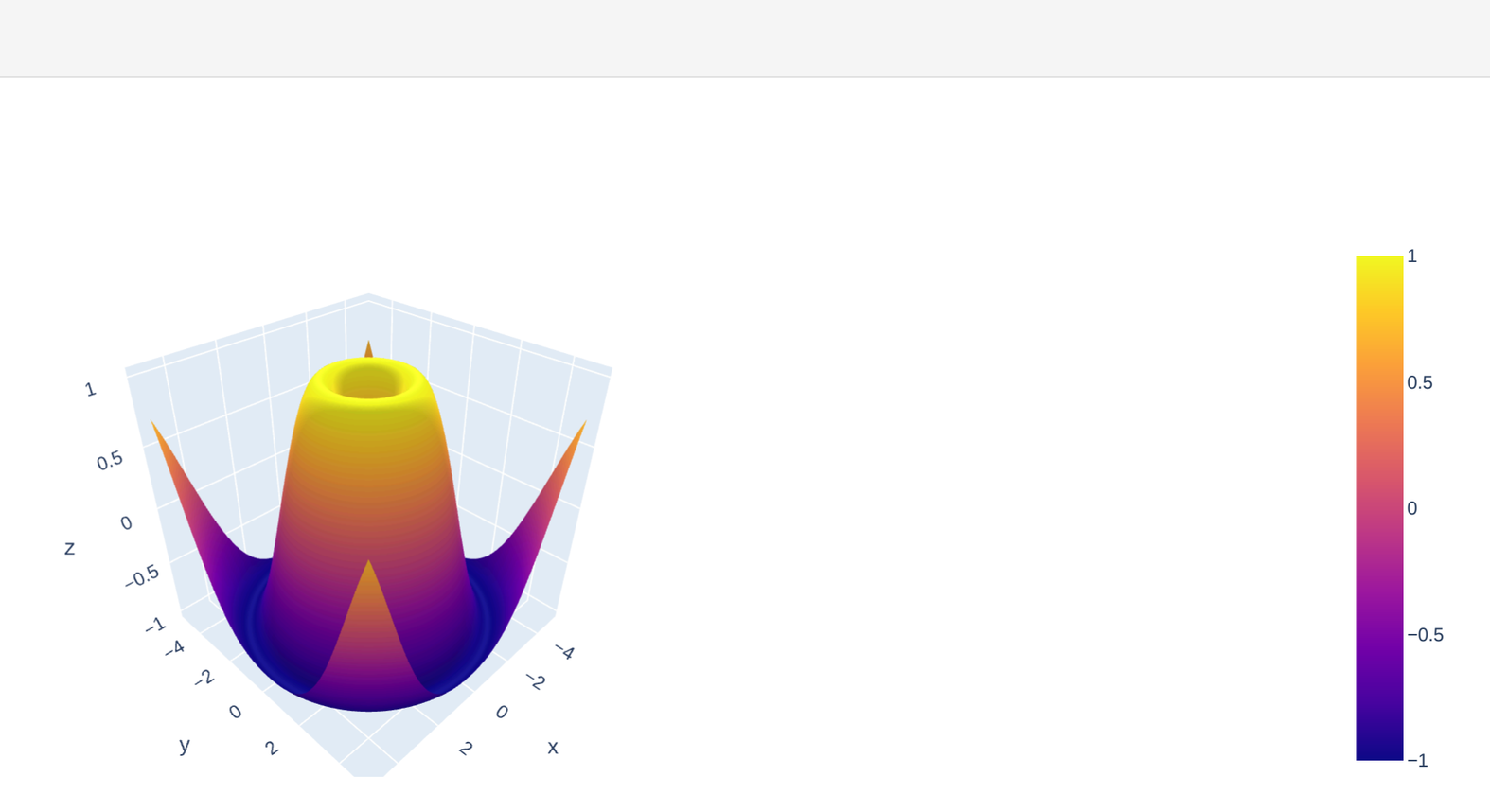


In [8]:
```python
#Q5.  Using the given dataset, create a bubble chart to represent each country's population (y-axis), GDP (x
# axis), and bubble size proportional to the population
np.random.seed(25)
data = {
 'Country': ['USA', 'Canada', 'UK',
'Germany', 'France'],
 'Population':
np.random.randint(100, 1000, 5),
 'GDP': np.random.randint(500, 2000,
5)
 }

df = pd.DataFrame(data)
df
fig=px.scatter(df['GDP'],df['Population'],size=df['Population'])
fig.show()
```

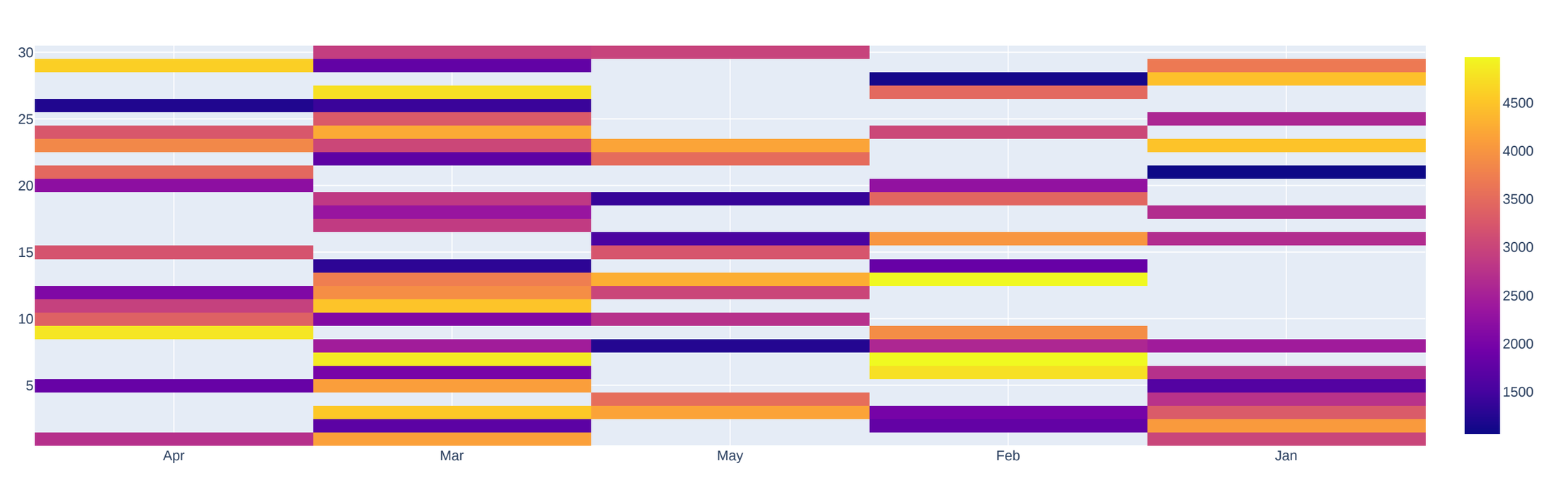


In [9]:
```python
# Using the sales data, generate a heatmap to visualize the variation in sales across
# different months and days.
np.random.seed(20)
data = {
 'Month': np.random.choice(['Jan', 'Feb', 'Mar', 'Apr', 'May'], 100),
 'Day': np.random.choice(range(1, 31), 100),
 'Sales': np.random.randint(1000, 5000, 100)
 }

df = pd.DataFrame(data)
```

In [10]:
```python
#Q4.   Using the given x and y data, generate a 3D surface plot to visualize the function
x= np.linspace(-5, 5, 100)
y = np.linspace(-5, 5, 100)
x, y = np.meshgrid(x, y)
z = np.sin(np.sqrt(x**2 + y**2))
data = {
 'X': x.flatten(),
 'Y': y.flatten(),
 'Z': z.flatten()
 }

df = pd.DataFrame(data)
fig=go.Figure()
fig.add_trace(go.Surface(x=x,y=y,z=z))
fig.show()
```



In [31]:
```python
#Q3.  Using the sales data, generate a heatmap to visualize the variation in sales across different months and
# days
np.random.seed(20)
data = {
 'Month': np.random.choice(['Jan', 'Feb', 'Mar', 'Apr', 'May'], 100),
 'Day': np.random.choice(range(1, 31), 100),
 'Sales': np.random.randint(1000, 5000, 100)
 }

df = pd.DataFrame(data)
df
fig=go.Figure()
fig.add_trace(go.Heatmap(x=df['Month'],y=df['Day'],z=df['Sales']))
fig.show()
```



In [ ]:

Bokeh Assignment

In [13]:
```python
#Q1. Create a Bokeh plot displaying a sine wave. Set x-values from 0 to 10 and y-values as the sine of x.
import math
ls=np.arange(0,10,0.09)
ls1=[math.sin(i) for i in ls]
x_value=np.array(ls)
sin_value=np.array(ls1)
df=pd.DataFrame({'X': x_value, 'Sin(X)': sin_value})
df
import bokeh.io
import bokeh.plotting
from bokeh.io import output_notebook, output_file, show
from bokeh.plotting import figure
output_file("test.html")
p=figure(title="Value of Sin Function")
p.xaxis.axis_label="X values -->"
p.yaxis.axis_label="Sin Values -->"
p.line(df['X'],df['Sin(X)'])
show(p)
```

In [14]:
```python
#Q2. Create a Bokeh scatter plot using randomly generated x and y values. Use different sizes and colors for the
# markers based on the 'sizes' and 'colors' columns
```

In [15]:
```python
x_value=np.random.randint(1,5,20)
y_value=np.random.randint(1,5,20)
colors = ['#%02x%02x%02x' % (r, g, b) for r, g, b in zip(np.random.randint(0, 256, 20), np.random.randint(0, 256, 20), np.random.randint(0, 256, 20))]
sizes = np.random.random(20) * 30 + 10
data={'X-Val':x_value,'Y-Val':y_value,'Colors':colors,'Sizes':sizes}
df=pd.DataFrame(data)
output_file('test1.html')
p=figure(title='Plot Between X and Y')
p.xaxis.axis_label='X-Values -->'
p.yaxis.axis_label='Y-Values -->'
p.scatter(df['X-Val'],df['Y-Val'], line_color=df['Colors'],size=df['Sizes'],fill_alpha=0.6)
show(p)
# fill_color and line_color both are used
# fill_alpha are used for transparency...
```

In [16]:
```python
#Q3. Generate a Bokeh bar chart representing the counts of different fruits using the following dataset.
fruits = ['Apples', 'Oranges', 'Bananas', 'Pears']
counts = [20, 25, 30, 35]
data={'Fruit_N':fruits,'Counts':counts}
df=pd.DataFrame(data)
output_file("test3.html")
df
p=figure(x_range=fruits,title='Plot Between Fruits and counts')
p.xaxis.axis_label='Fruits-Values -->'
p.yaxis.axis_label='Counts-Values -->'
p.vbar(x=df['Fruit_N'],top=df['Counts'],line_color="white")   # for horizontal->x and for vertical ->top
show(p)
```

In [17]:
```python
#Q4. Create a Bokeh histogram to visualize the distribution of the given data
data_hist = np.random.randn(1000)
hist, edges = np.histogram(data_hist, bins=30)
output_file("histogram.html")
p = figure(title="Histogram of Random Data", x_axis_label='Value', y_axis_label='Frequency')


p.quad(top=hist, bottom=0, left=edges[:-1], right=edges[1:], fill_color="navy", line_color="white", alpha=0.6)
p.xgrid.grid_line_color = None
p.y_range.start = 0
show(p)
```

In [32]:
```python
#Q5.  Create a Bokeh heatmap using the provided dataset.
import numpy as np
import pandas as pd
from bokeh.plotting import figure, show, output_notebook
from bokeh.models import ColumnDataSource, ColorBar
from bokeh.transform import linear_cmap
from bokeh.palettes import Viridis256


#
data_heatmap = np.random.rand(10, 10)
x = np.linspace(0, 1, 10)
y = np.linspace(0, 1, 10)
xx, yy = np.meshgrid(x, y)


data = {
    'x': xx.flatten(),
    'y': yy.flatten(),
    'values': data_heatmap.flatten()
}

df = pd.DataFrame(data)

source = ColumnDataSource(df)

p = figure(title="Heatmap Example", x_axis_label='X', y_axis_label='Y', tools="hover", tooltips=[('Value', '@values')])

mapper = linear_cmap(field_name='values', palette=Viridis256, low=min(df['values']), high=max(df['values']))

p.rect(x='x', y='y', width=1/10, height=1/10, source=source, line_color=None, fill_color=mapper)


color_bar = ColorBar(color_mapper=mapper['transform'], width=8, location=(0,0))
p.add_layout(color_bar, 'right')

show(p)
```

In [ ]:

Seaborn assignment

In [22]:
```python
#Q1. Create a scatter plot to visualize the relationship between two variables, by generating a synthetic
# dataset.
import seaborn as sns
sns.get_dataset_names()
```

Out[22]:
```
['anagrams',
 'anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'dowjones',
 'exercise',
 'flights',
 'fmri',
 'geyser',
 'glue',
 'healthexp',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'seaice',
 'taxis',
 'tips',
 'titanic',
 'anagrams',
 'anagrams',
 'anscombe',
 'anscombe',
 'attention',
 'attention',
 'brain_networks',
 'brain_networks',
 'car_crashes',
 'car_crashes',
 'diamonds',
 'diamonds',
 'dots',
 'dots',
 'dowjones',
 'dowjones',
 'exercise',
 'exercise',
 'flights',
 'flights',
 'fmri',
 'fmri',
 'geyser',
 'geyser',
 'glue',
 'glue',
 'healthexp',
 'healthexp',
 'iris',
 'iris',
 'mpg',
 'mpg',
 'penguins',
 'penguins',
 'planets',
 'planets',
 'seaice',
 'seaice',
 'taxis',
 'taxis',
 'tips',
 'tips',
 'titanic',
 'titanic',
 'anagrams',
 'anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'dowjones',
 'exercise',
 'flights',
 'fmri',
 'geyser',
 'glue',
 'healthexp',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'seaice',
 'taxis',
 'tips',
 'titanic']
```

In [23]:
```python
df=sns.load_dataset("diamonds")
df
```

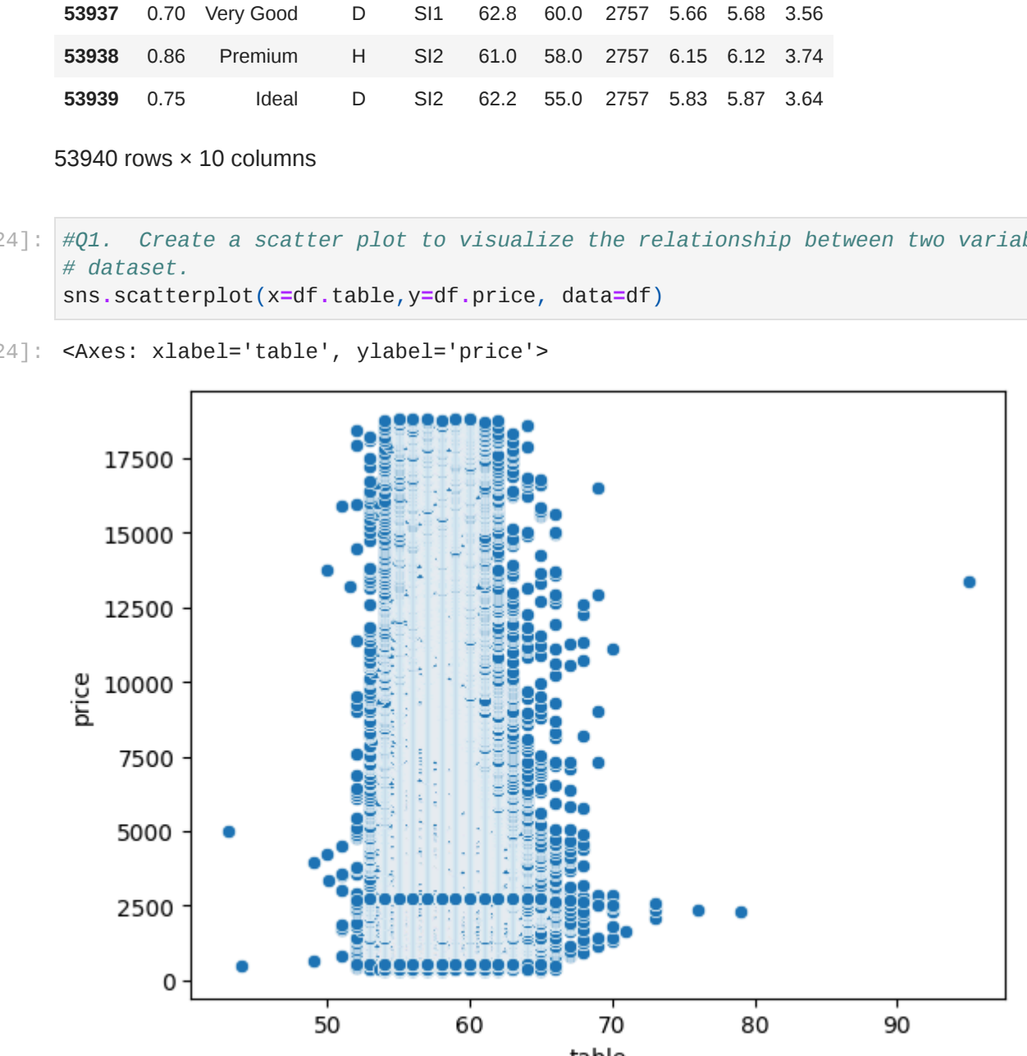
Out[23]:

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75
...	...	...	...	...	...	...	...	...	...	...
53935	0.72	Ideal	D	SI1	60.8	57.0	2757	5.75	5.76	3.50
53936	0.72	Good	D	SI1	63.1	55.0	2757	5.69	5.75	3.61
53937	0.70	Very Good	D	SI1	62.8	60.0	2757	5.66	5.68	3.56
53938	0.86	Premium	H	SI2	61.0	58.0	2757	6.15	6.12	3.74
53939	0.75	Ideal	D	SI2	62.2	55.0	2757	5.83	5.87	3.64

53940 rows × 10 columns

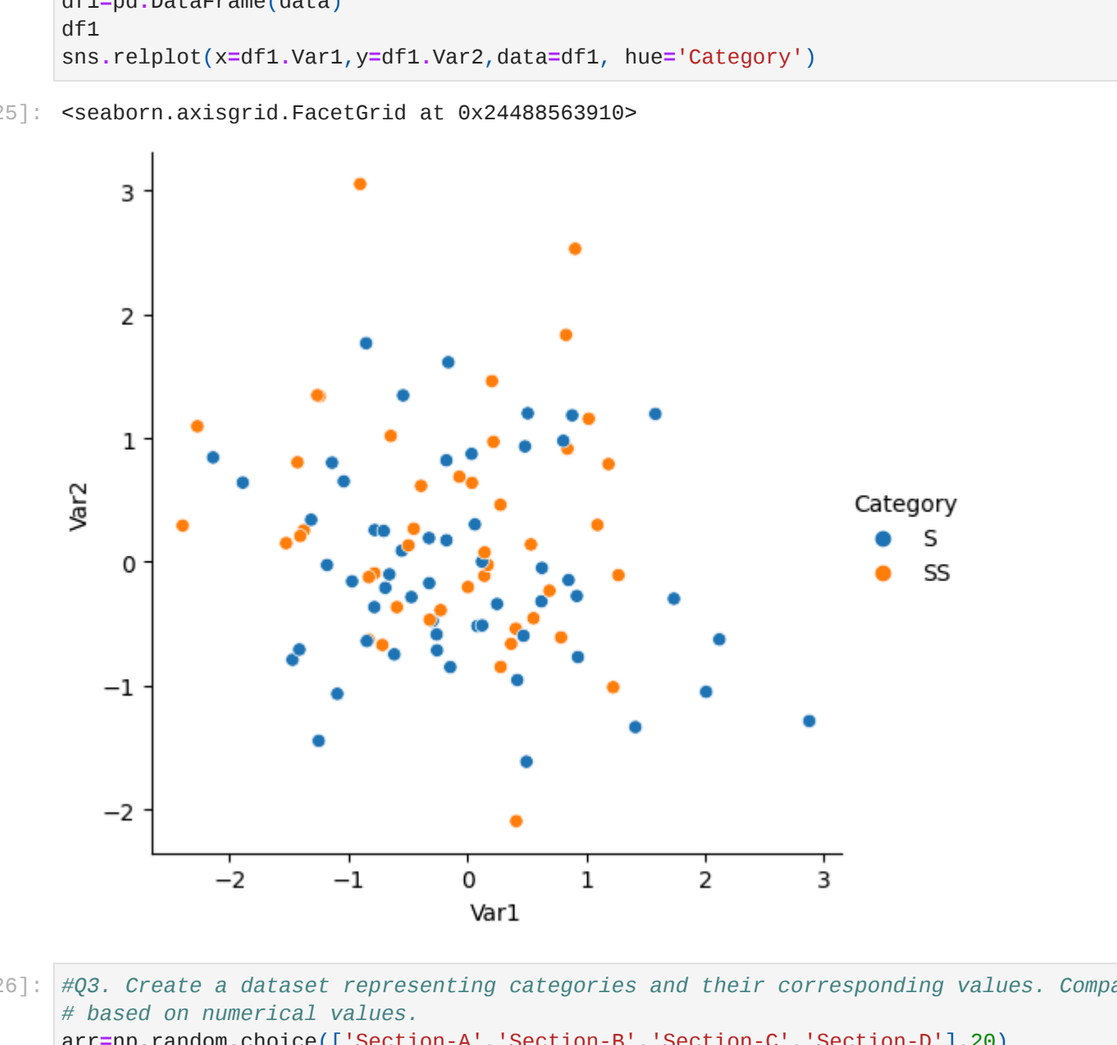
In [24]:
```python
#Q1.  Create a scatter plot to visualize the relationship between two variables, by generating a synthetic
# dataset.
sns.scatterplot(x=df.table,y=df.price, data=df)
```

Out[24]: <Axes: xlabel='table', ylabel='price'>



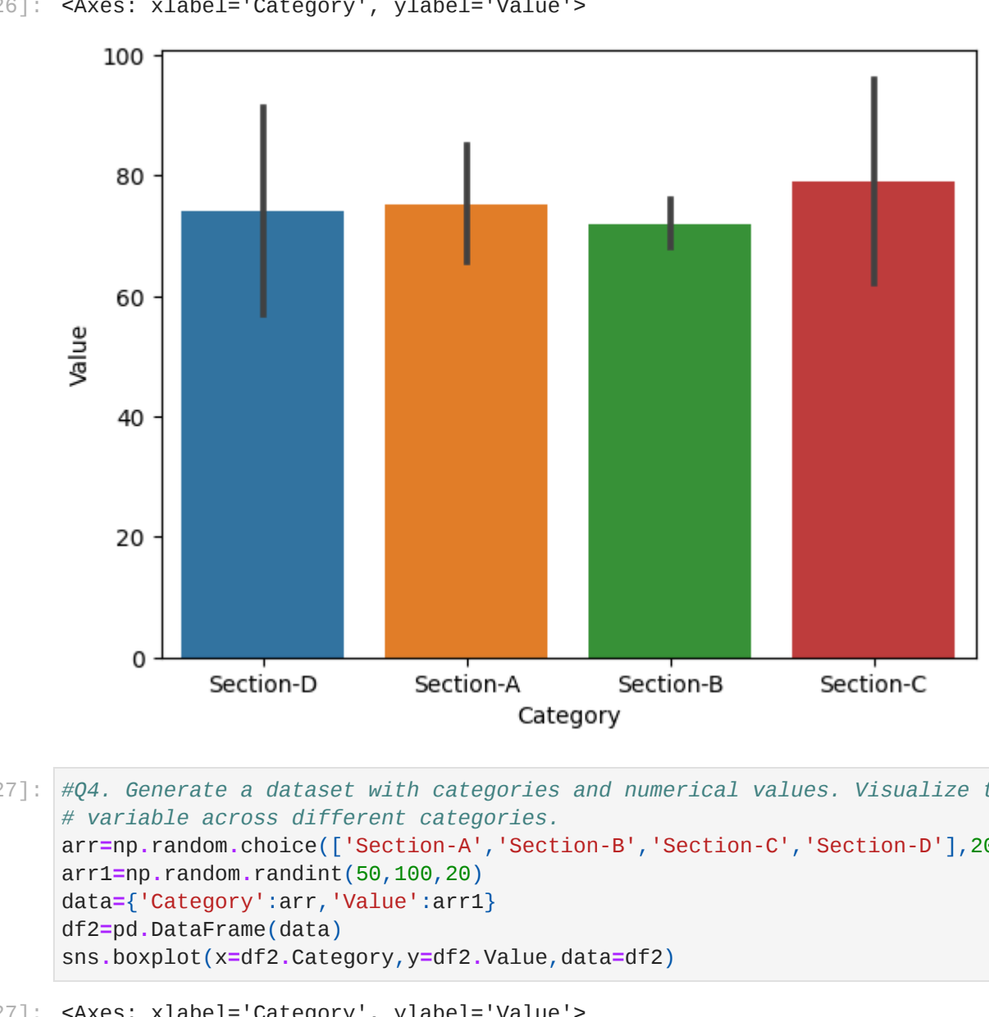
In [25]:
```python
#Q2.  Generate a dataset of random numbers. Visualize the distribution of a numerical variable
arr=np.random.randn(100)
arr2=np.random.randn(100)
arr3=np.random.choice(['S','SS'],100)
data={'Var1':arr,'Var2':arr2 ,'Category':arr3}
df1=pd.DataFrame(data)
df1
sns.relplot(x=df1.Var1,y=df1.Var2,data=df1, hue='Category')
```

Out[25]: <seaborn.axisgrid.FacetGrid at 0x24488563910>



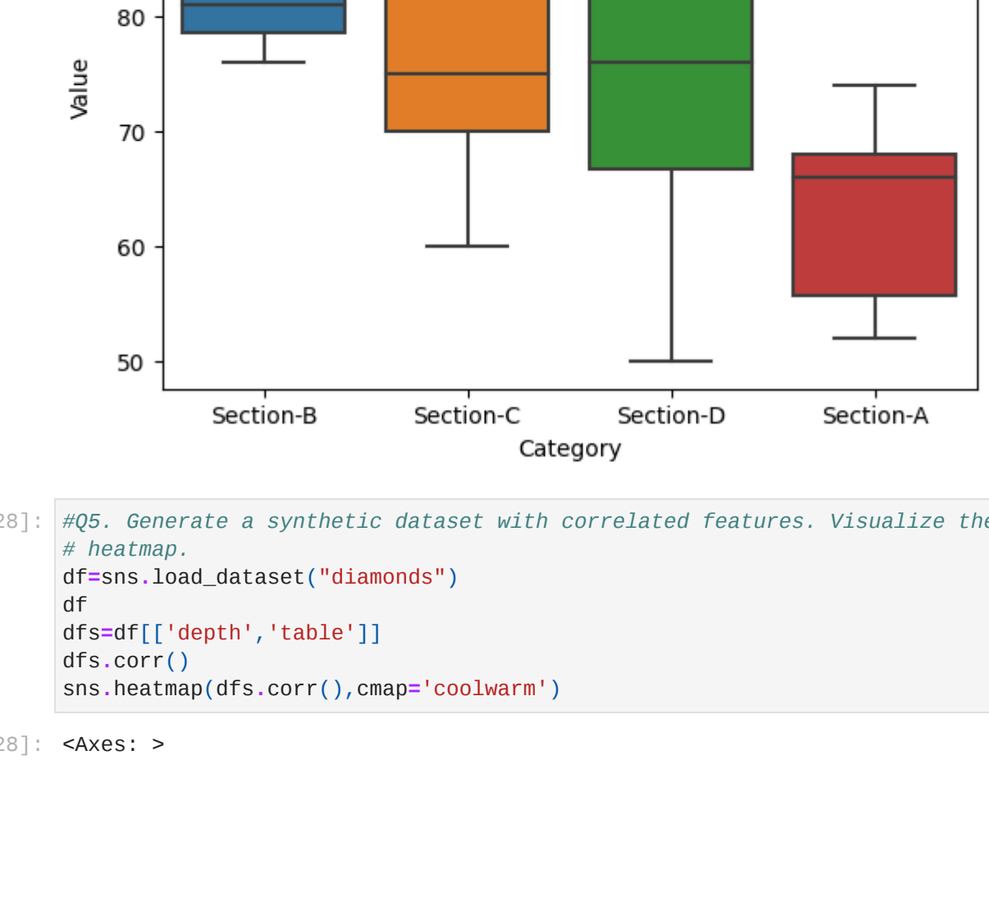
In [26]:
```python
#Q3. Create a dataset representing categories and their corresponding values. Compare different categories
# based on numerical values.
arr=np.random.choice(['Section-A','Section-B','Section-C','Section-D'],20)
arr1=np.random.randint(50,100,20)
data={'Category':arr,'Value':arr1}
df=pd.DataFrame(data)
sns.barplot(x=df.Category,y=df.Value,data=df)
```

Out[26]: <Axes: xlabel='Category', ylabel='Value'>



In [27]:
```python
#Q4. Generate a dataset with categories and numerical values. Visualize the distribution of a numerical
# variable across different categories.
arr=np.random.choice(['Section-A','Section-B','Section-C','Section-D'],20)
arr1=np.random.randint(50,100,20)
data={'Category':arr,'Value':arr1}
df2=pd.DataFrame(data)
sns.boxplot(x=df2.Category,y=df2.Value,data=df2)
```

Out[27]: <Axes: xlabel='Category', ylabel='Value'>



In [28]:
```python
#Q5. Generate a synthetic dataset with correlated features. Visualize the correlation matrix of a dataset using a
# heatmap.
df=sns.load_dataset("diamonds")
df
dfs=df[['depth','table']]
dfs.corr()
sns.heatmap(dfs.corr(),cmap='coolwarm')
```

Out[28]: <Axes: >


Math
Math
Math
Math
1.0
0.8
0.6
0.4
0.2
0.0
-0.2