Example#01

01. Importing Libraries

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

02. Creating arrays for x (Hours Studied) and y (Final Exam Score)

In [2]:
```
x = np.array([1,2,3,4,5])
y = np.array([3,4,2,4,5])

print(f"x or Independent variable: {x}\ny or Dependent variable: {y}")
```

```
x or Independent variable: [1 2 3 4 5]
y or Dependent variable: [3 4 2 4 5]
```

03. Finding Mean of x and y

Mean = (Sum of all values)/(Total Values)

In [3]:
```
xMean = np.mean(x)
yMean = np.mean(y)
print(f"xMean: {xMean}\nyMean: {yMean}")
```

```
xMean: 3.0
yMean: 3.6
```

04. Finding Slope ""m""

m= cov(x,y) / var(x) m= ∑ [(x - xMean) * (y - yMean)] / (x - xMean)^2

In [4]:
```
covXY = 0
varX = 0

n = len(x) # we need this as required to iterate n times

for i in range(n):
    covXY += (x[i] - xMean) * (y[i] - yMean)
    varX += (x[i] - xMean)**2

m = covXY / varX
print(covXY)
print(varX)
print(f"Slope: {m}")
```

```
4.0
10.0
Slope: 0.4
```

05. Finding c or y-Intercept

In [5]:
```
c = yMean - (m * xMean)
print(f"c:{c}")
```

```
c:2.4
```

06. Predicting the Value of y when x = 6, 7, 8

In [6]:
```
y1 = m*6 + c
print(y1)
```

```
4.800000000000001
```

In [7]:
```
y2 = m*7 + c
print(y2)
```

```
5.2
```

In [8]:
```
y3 = m*8 + c
print(y3)
```

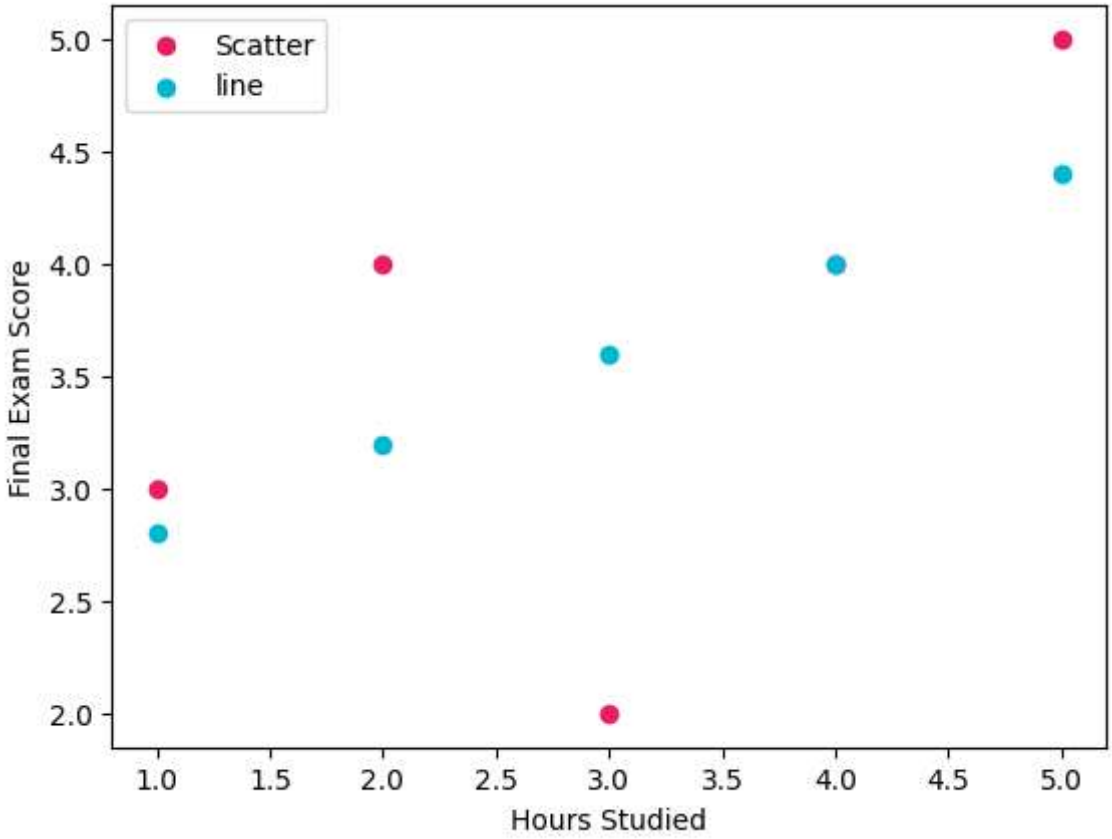
```
5.6
```

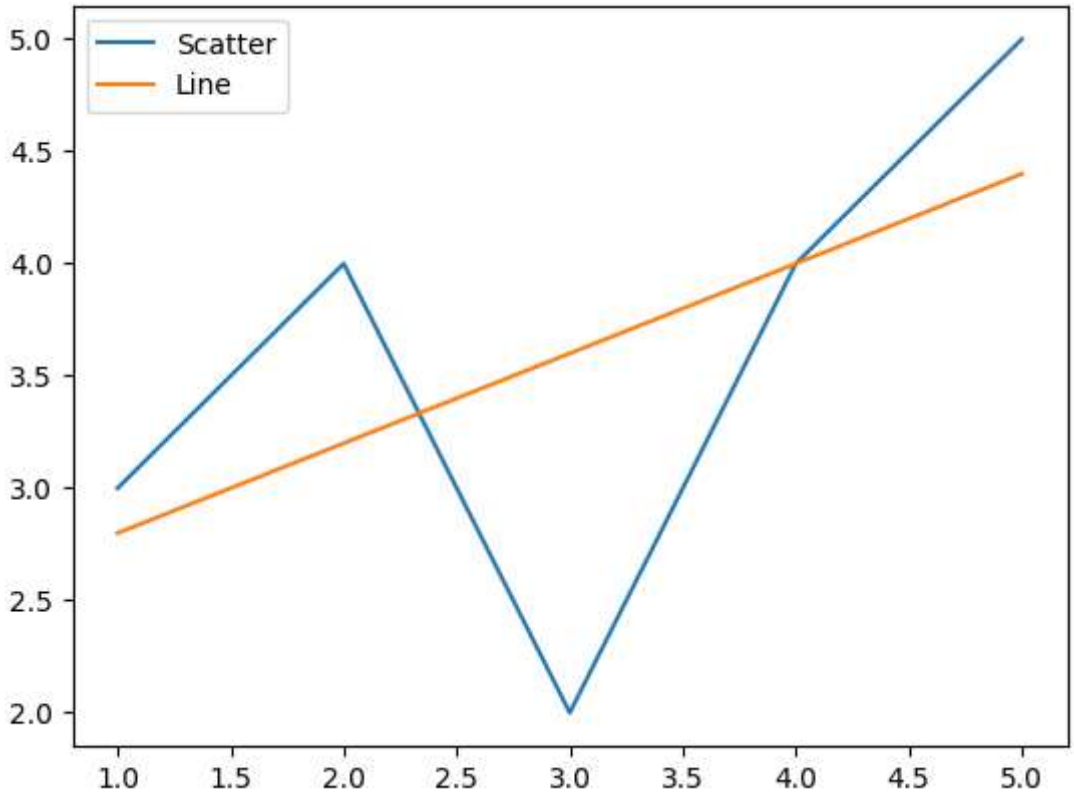
In [9]:
```
yPredict = m*x +c
yPredict
```

Out[9]:
```
array([2.8, 3.2, 3.6, 4. , 4.4])
```

07. Plotting the Original and Predicted Values

In [10]:
```
plt.scatter(x,y, color = "#e91e63", label = "Scatter")
plt.xlabel("Hours Studied")
plt.ylabel("Final Exam Score")
plt.scatter(x,yPredict, color = "#00b8d4", label = "line")
plt.legend()
plt.show()
plt.plot(x,y, label ="Scatter")
plt.plot(x,yPredict, label = "Line")
plt.legend()
plt.show()
```





08. Finding R Square

R^2= ∑ (yPredicted - yMean)^2 / (y - yMean)^2

In [11]:
```
rNum = 0
rDen = 0

for i in range(n):
    rNum += (yPredict[i] - yMean)**2
    rDen += (y[i] - yMean)**2

r = rNum / rDen

print(f"R Square: {r}")
print(f"Accuracy is: {round(r*100)}%")
```

```
R Square: 0.3076923076923078
Accuracy is: 31%
```

Example 02

1.importing libs

In [12]:
```
import numpy as np
import matplotlib.pyplot as plt
```

2. Creating arrays for x (Hours Studied) and y (Final Exam Score)

In [13]:
```
x = np.array([2,3,4,5,6,7])
y = np.array([60,70,80,85,90,95])

print(f"Hours Studied: {x}\nFinal Exam Score: {y}")
```

```
Hours Studied: [2 3 4 5 6 7]
Final Exam Score: [60 70 80 85 90 95]
```

03. Finding Mean of x and y

In [14]:
```
xMean = np.mean(x)
yMean = np.mean(y)
print(f"xMean: {xMean}\nyMean: {yMean}")
```

```
xMean: 4.5
yMean: 80.0
```

04. Finding Slope ""m""

In [15]:
```
covXY = 0
varX = 0

n = len(x) # we need this as required to iterate n times
```

```
for i in range(n):
    covXY += (x[i] - xMean) * (y[i] - yMean)
    varX += (x[i] - xMean)**2

m = covXY / varX
print(covXY)
print(varX)
print(f"Slope: {m}")
```

```
120.0
17.5
Slope: 6.857142857142857
```

05. Finding c or y-Intercept

In [16]:
```
c = yMean - (m * xMean)
print(f"c:{c}")
```

```
c:49.142857142857146
```

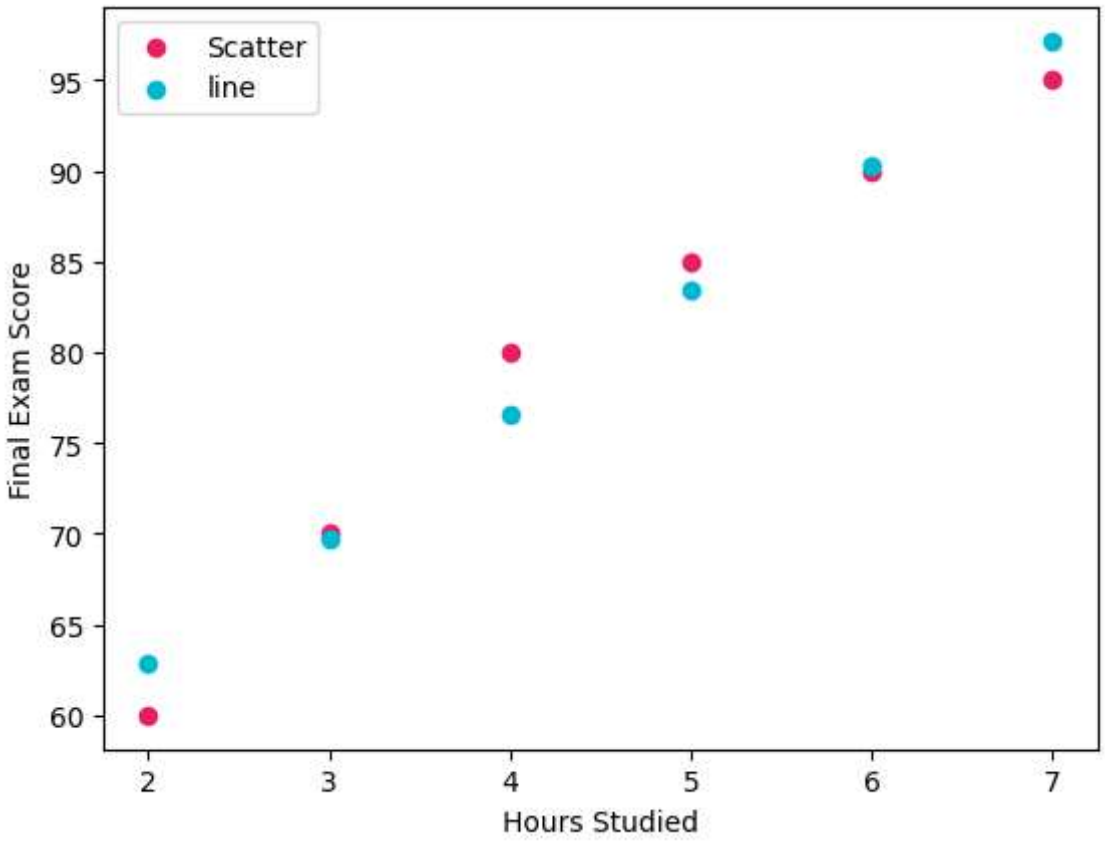
06. Predicting the Value of y (Final Exam Score)

In [17]:
```
yPredict = m*x + c
print(yPredict)
```

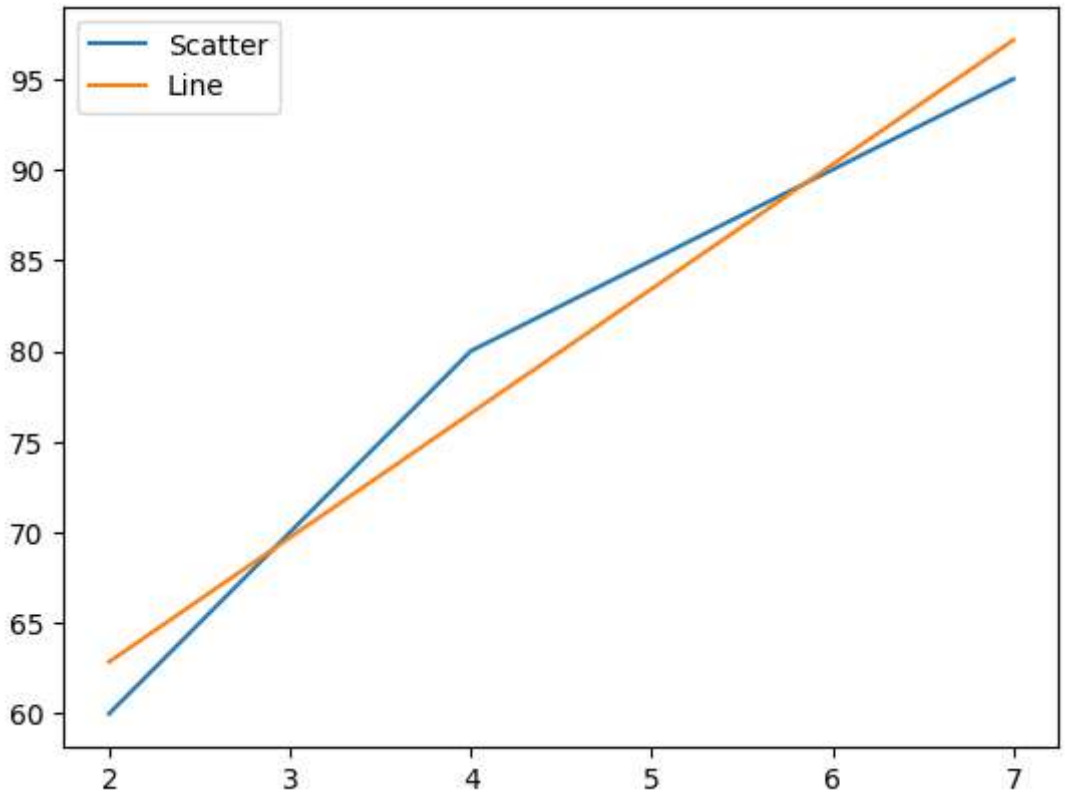
```
[62.85714286 69.71428571 76.57142857 83.42857143 90.28571429 97.14285714]
```

07. Plotting the Original and Predicted Values

In [18]:
```
plt.scatter(x,y, color = "#e91e63", label = "Scatter")
plt.xlabel("Hours Studied")
plt.ylabel("Final Exam Score")
plt.scatter(x,yPredict, color = "#00b8d4", label = "line")
plt.legend()
plt.show()
```



In [19]:
```
plt.plot(x,y, label ="Scatter")
plt.plot(x,yPredict, label = "Line")
plt.legend()
plt.show()
```



Exapmle 03

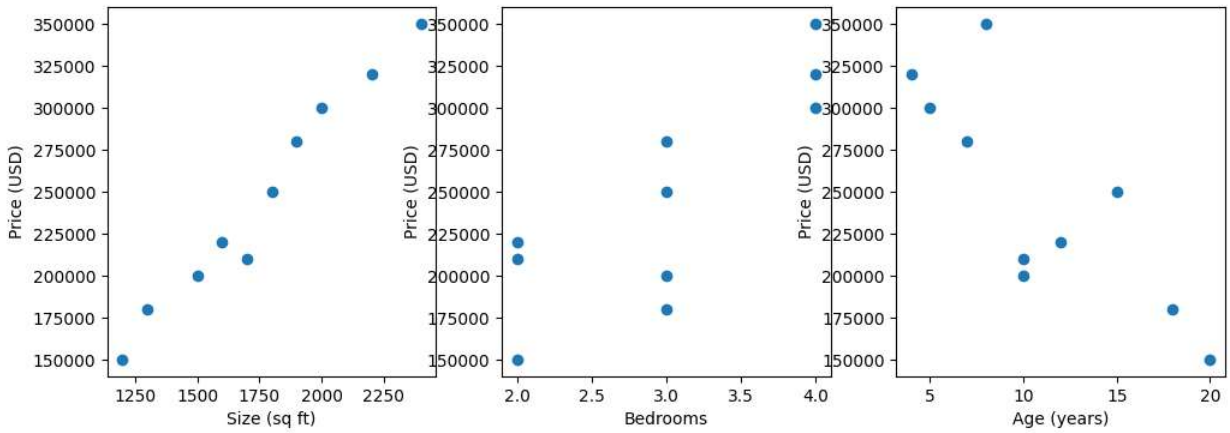
In [20]:
```
# Data
size = [1500, 2000, 1200, 1800, 1600, 2400, 1300, 1700, 1900, 2200]
bedrooms = [3, 4, 2, 3, 2, 4, 3, 2, 3, 4]
age = [10, 5, 20, 15, 12, 8, 18, 10, 7, 4]
price = [200000, 300000, 150000, 250000, 220000, 350000, 180000, 210000, 280000, 32000
```

2. Specify the Independent and Dependent variable

In [21]:
```
## Independent variables
x1 = size
x2 = bedrooms
x3 = age

# Dependent variable
y = price
```

3. Plot the Price against Each Feature

In [22]:
```
# Plotting
fig, axs = plt.subplots(1, 3, figsize=(12, 4))
axs[0].scatter(size, price)
axs[0].set_xlabel('Size (sq ft)')
axs[0].set_ylabel('Price (USD)')
axs[1].scatter(bedrooms, price)
axs[1].set_xlabel('Bedrooms')
axs[1].set_ylabel('Price (USD)')
axs[2].scatter(age, price)
axs[2].set_xlabel('Age (years)')
axs[2].set_ylabel('Price (USD)')

plt.show()
```



4. Find the Mean

In [23]:
```
x1Mean = np.mean(x1)
x2Mean = np.mean(x2)
x3Mean = np.mean(x3)

yMean = np.mean(y)
```

In [24]:
```
def findSlope(x,y):
    covXY = 0
    varX = 0

    xMean = np.mean(x)
    yMean = np.mean(y)

    n = len(x) # we need this as required to iterate n times

    for i in range(n):
        covXY += (x[i] - xMean) * (y[i] - yMean)
        varX += (x[i] - xMean)**2

    m = covXY / varX
    # print(covXY)
    # print(varX)
    # print(f"Slope: {m}")
    m = int(m)
    return m
```

m1

In [25]:
```
m1 = findSlope(x1,y)
print(f"Slope m1: {m1}")
```

```
Slope m1: 168
```

m2

In [26]:
```
m2 = findSlope(x2,y)
print(f"Slope m2: {m2}")
```

```
Slope m2: 65000
```

m3

In [27]:
```
m3 = findSlope(x3,y)
print(f"Slope m3: {m3}")
```

```
Slope m3: -9826
```

6. Finding c or y-Intercept

In [28]:
```
c = yMean - (m1 * x1Mean) - (m2 * x2Mean) - (m3 * x3Mean)
c
```

Out[28]:
```
-137576.59999999998
```

7. Predicting the Value of y

In [29]:
```
yPredict = m1*x1 + m2*x2 + m3*x3 + c
yPredict
```

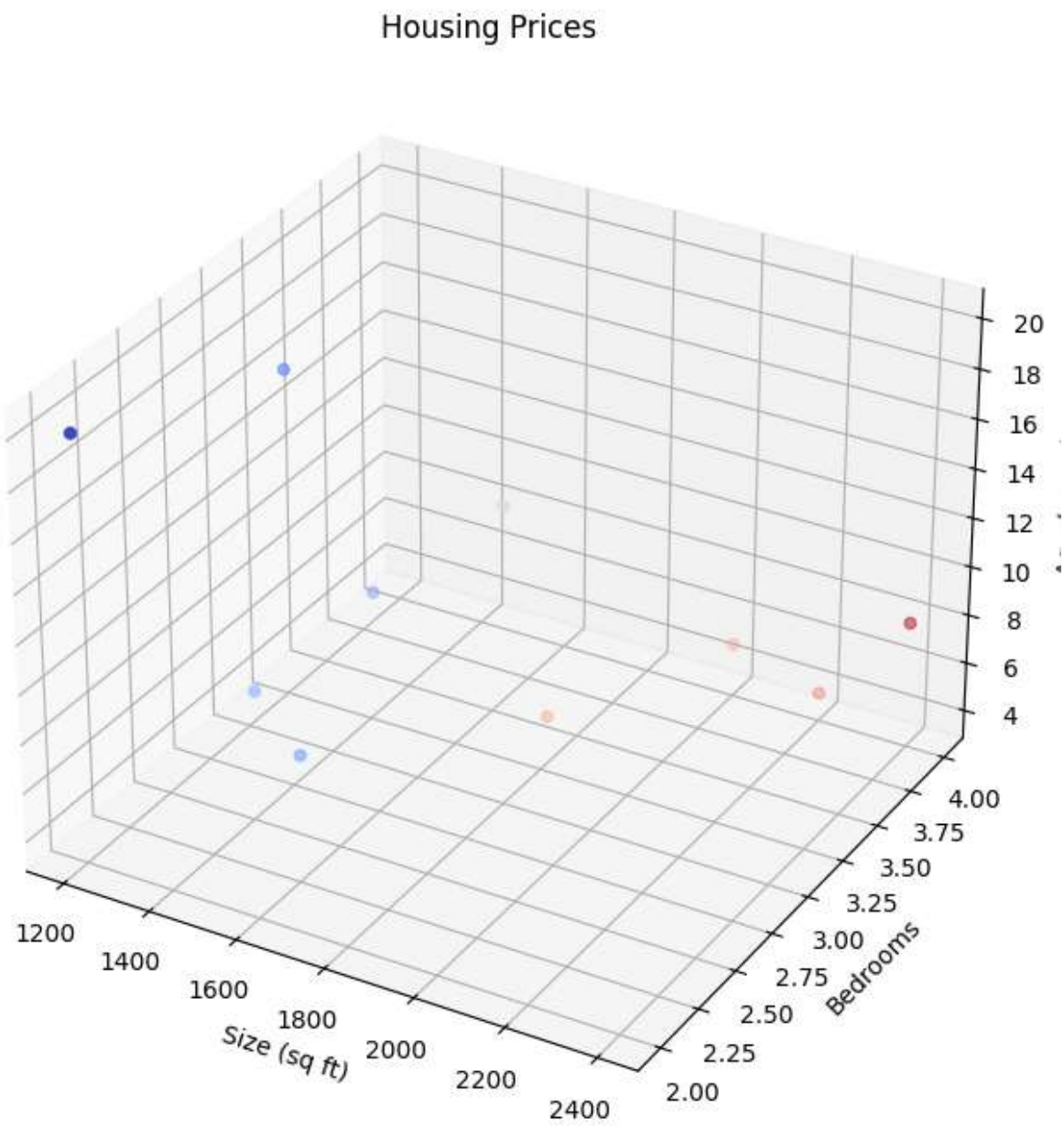
Out[29]:
```
array([-136076.6, -135576.6, -136376.6, ..., -137574.6, -137573.6,
       -137572.6])
```

In [30]:
```
rNum = 0
rDen = 0

for i in range(n):
    rNum += (yPredict[i] - yMean)**2
    rDen += (y[i] - yMean)**2

r = rNum / rDen

print(f"R Square: {r}")
print(f"Accuracy is: {round(r*100)}%")
print(f"Error: {round(100-r*100)}%")
```

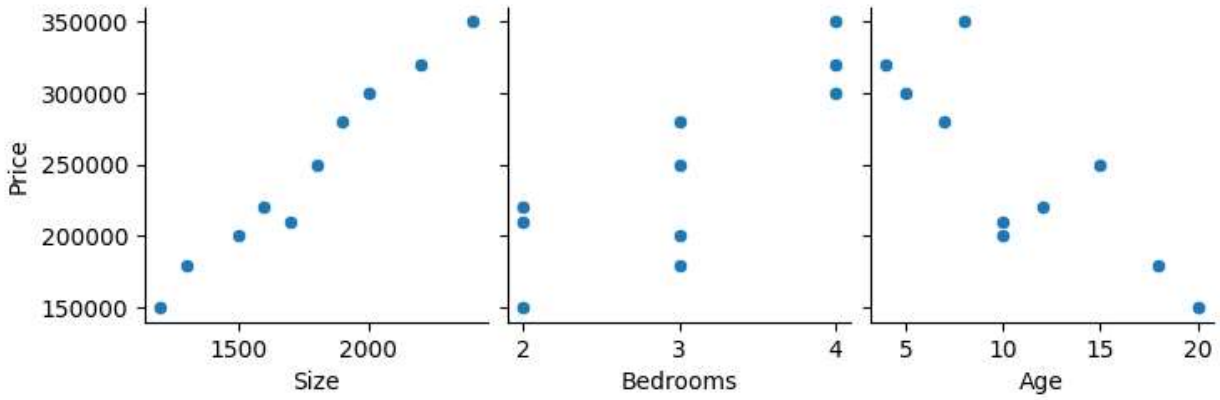
```
R Square: 33.96296745633484
Accuracy is: 3396%
Error: -3296%
```

In [31]:
```
from mpl_toolkits.mplot3d import Axes3D
```

In [32]:
```
fig = plt.figure(figsize=(8, 8))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(size, bedrooms, age, c=price, marker='o', cmap='coolwarm')
ax.set_xlabel('Size (sq ft)')
ax.set_ylabel('Bedrooms')
ax.set_zlabel('Age (years)')
ax.set_title('Housing Prices')
plt.show()
```



In [37]:
```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt

# Data
data = {'Size': [1500, 2000, 1200, 1800, 1600, 2400, 1300, 1700, 1900, 2200],
        'Bedrooms': [3, 4, 2, 3, 2, 4, 3, 2, 3, 4],
        'Age': [10, 5, 20, 15, 12, 8, 18, 10, 7, 4],
        'Price': [200000, 300000, 150000, 250000, 220000, 350000, 180000, 210000, 2800
df = pd.DataFrame(data)

# Plotting
sns.pairplot(df, x_vars=['Size', 'Bedrooms', 'Age'], y_vars=['Price'], kind='scatter')
plt.show()
```


Price
350000
300000
250000
200000
150000
1500
2000
Size
2
3
4
Bedrooms
5
10
15
20
Age