In [158]:

```
#1. Download the dataset

#2. Load the dataset into the tool.

import numpy as np
import pandas as pd

data = pd.read_csv(r"D:\ibm\ASS4\abalone.csv")
data.head()
#data.columns
```

Out[158]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

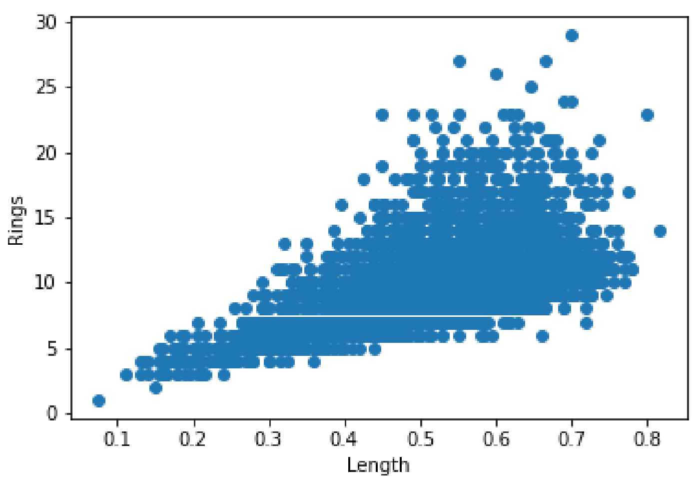
In [159]:

```
#3) Perform Below Visualizations.
# Univariate Analysis
# Bi-Variate Analysis
# Multi-Variate Analysis

import matplotlib.pyplot as plt
plt.scatter(data.Length,data.Rings)
plt.xlabel("Length")
plt.ylabel("Rings")
```

Out[159]:

```
Text(0, 0.5, 'Rings')
```



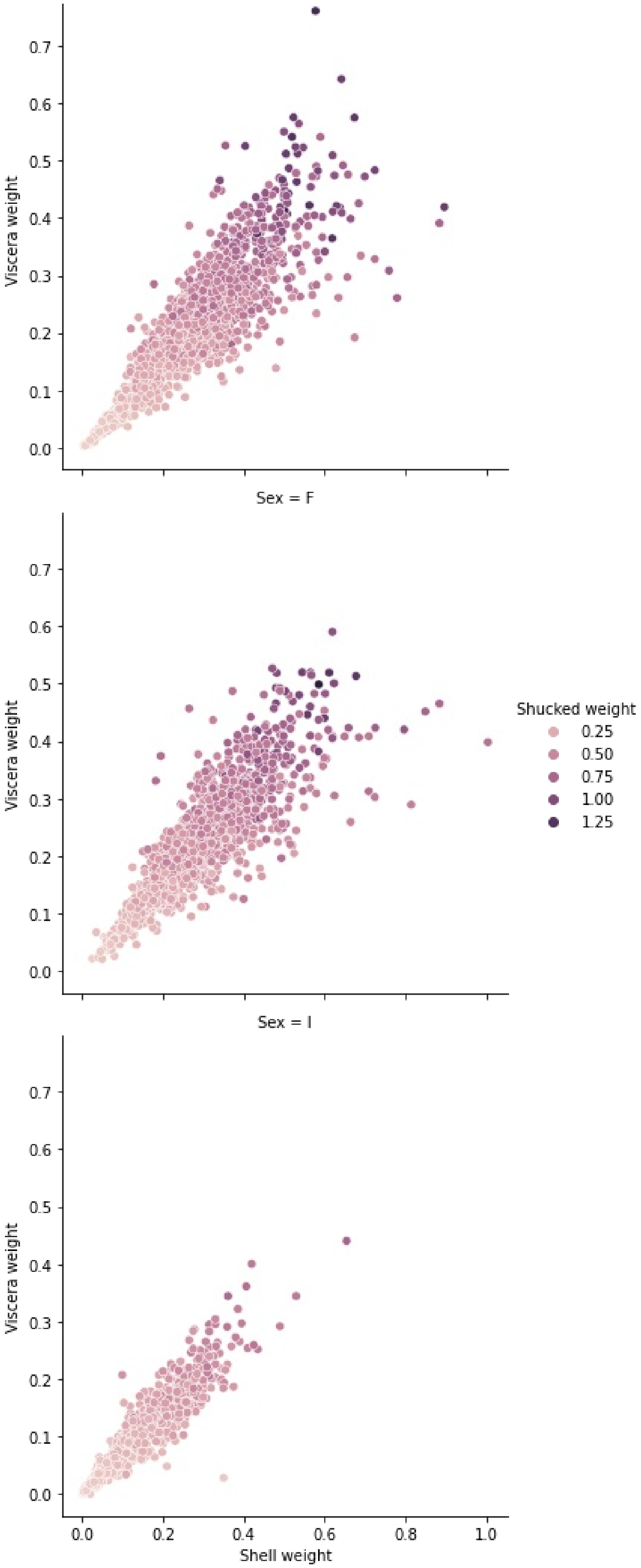
In [160]:

```
import seaborn as sns
sns.relplot(data=data,x="Shell weight",y='Viscera weight',hue='Shucked weight',col='Sex',
col_wrap=1)
```

Out[160]:

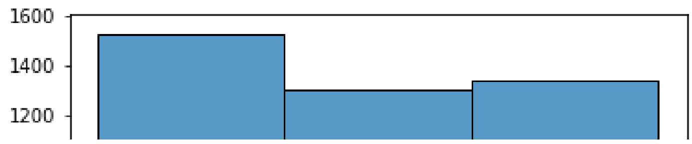
```
<seaborn.axisgrid.FacetGrid at 0x1f4b4c50b80>
```

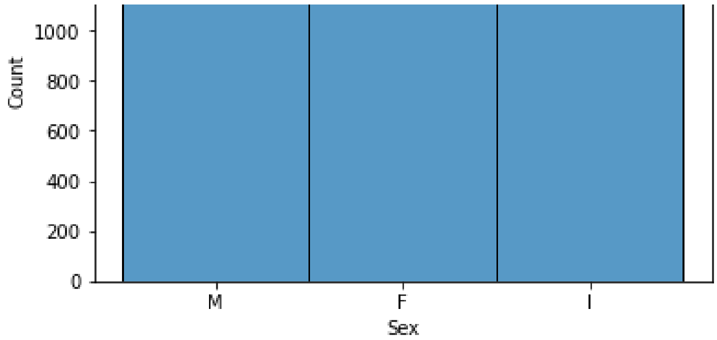
Sex = M



In [161]:

```
sns.histplot(x='Sex', data=data);
```





In [162]:

```
#4. Perform descriptive statistics on the dataset.
data.describe()
```

Out[162]:

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
count	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000
mean	0.523992	0.407881	0.139516	0.828742	0.359367	0.180594	0.238831	9.933684
std	0.120093	0.099240	0.041827	0.490389	0.221963	0.109614	0.139203	3.224169
min	0.075000	0.055000	0.000000	0.002000	0.001000	0.000500	0.001500	1.000000
25%	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	0.130000	8.000000
50%	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	0.234000	9.000000
75%	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	0.329000	11.000000
max	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	1.005000	29.000000

In [163]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4177 entries, 0 to 4176
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Sex             4177 non-null   object 
 1   Length          4177 non-null   float64
 2   Diameter        4177 non-null   float64
 3   Height          4177 non-null   float64
 4   Whole weight    4177 non-null   float64
 5   Shucked weight  4177 non-null   float64
 6   Viscera weight  4177 non-null   float64
 7   Shell weight    4177 non-null   float64
 8   Rings           4177 non-null   int64  
dtypes: float64(7), int64(1), object(1)
memory usage: 293.8+ KB
```

In [164]:

```
#5. Check for Missing values and deal with them.
data.isnull().sum()
```

Out[164]:

```
Sex               0
Length            0
Diameter          0
Height            0
Whole weight      0
Shucked weight    0
Viscera weight    0
Shell weight      0
Rings             0
dtype: int64
```

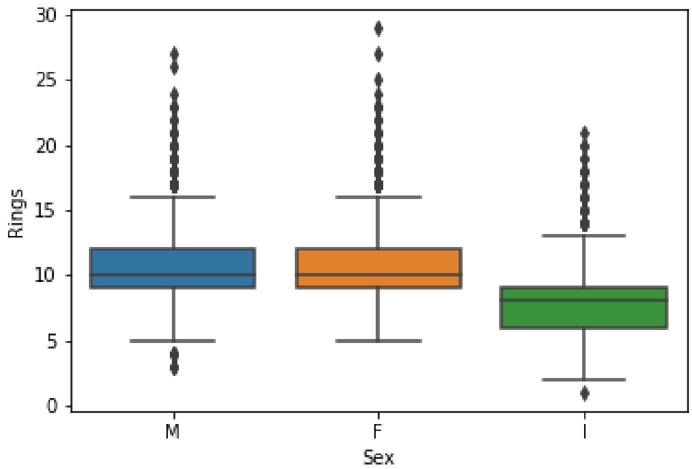
```
dtype: int64
```

In [165]:

```python
#6. Find the outliers and replace them outliers
sns.boxplot(x='Sex',y='Rings',data=data)
```

Out[165]:

```
<AxesSubplot:xlabel='Sex', ylabel='Rings'>
```



In [166]:

```python
#7. Check for Categorical columns and perform encoding
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
print(data.Sex.value_counts())

data.Sex=le.fit_transform(data.Sex)
print(data.Sex.value_counts())
```

```
M    1528
I    1342
F    1307
Name: Sex, dtype: int64
2    1528
1    1342
0    1307
Name: Sex, dtype: int64
```

In [167]:

```python
#8. Split the data into dependent and independent variables.
 #INDEPENDENT
x=data.iloc[:,[0,7]].values
x
```

Out[167]:

```
array([[2.   , 0.15 ],
       [2.   , 0.07 ],
       [0.   , 0.21 ],
       ...,
       [2.   , 0.308],
       [0.   , 0.296],
       [2.   , 0.495]])
```

In [168]:

```python
y = data.iloc[:,8].values
y
#dependent
```

Out[168]:

```
array([15,  7,  9, ...,  9, 10, 12], dtype=int64)
```

In [169]:

```
#9. Scale the independent variables
from sklearn.preprocessing import StandardScaler
ss = StandardScaler()
x = ss.fit_transform(x)
x
```

Out[169]:

```
array([[ 1.15198011, -0.63821689],
       [ 1.15198011, -1.21298732],
       [-1.28068972, -0.20713907],
       ...,
       [ 1.15198011,  0.49695471],
       [-1.28068972,  0.41073914],
       [ 1.15198011,  1.84048058]])
```

In [170]:

```
#10. Split the data into training and testing

from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

In [171]:

```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(3341, 2)
(836, 2)
(3341,)
(836,)
```

In [172]:

```
#11. Build the Model
#12. Train the Model

from sklearn.linear_model import LinearRegression
mlr=LinearRegression()
mlr.fit(x_train,y_train)
```

Out[172]:

```
LinearRegression()
```

In [173]:

```
#13. Test the Model

mlr.predict(x_test[0:5])
```

Out[173]:

```
array([11.2719321 ,  9.27390152, 11.02122356,  6.78830546, 11.88079569])
```

In [174]:

```
#14. Measure the performance using Metrics.

from sklearn.metrics import r2_score
r2_score(mlr.predict(x_test),y_test)
```

Out[174]:

```
-0.6963075808043906
```