Import the Libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
```

Importing the dataset

In [2]:
```
ds = pd.read_csv('titanic_dataset.csv')
```

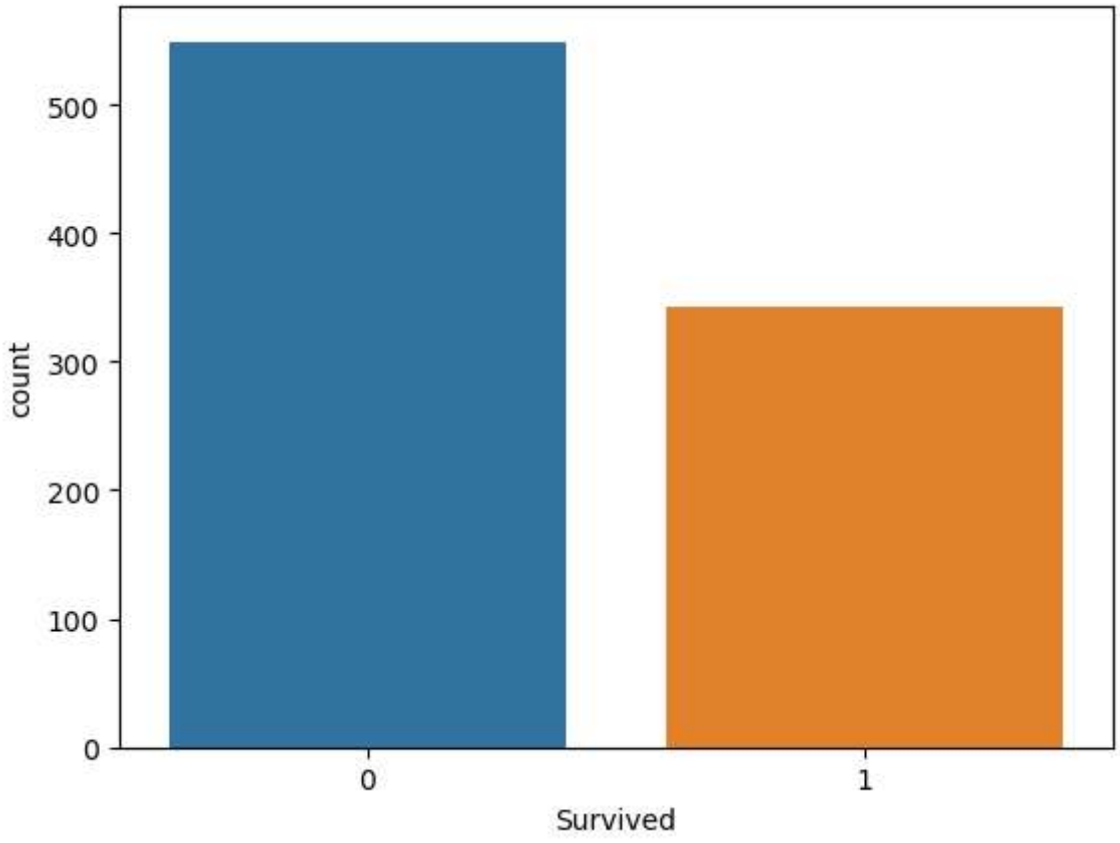
Checking for Null Values

In [3]:
```
null_values = ds.isnull().sum()
print(null_values)
```

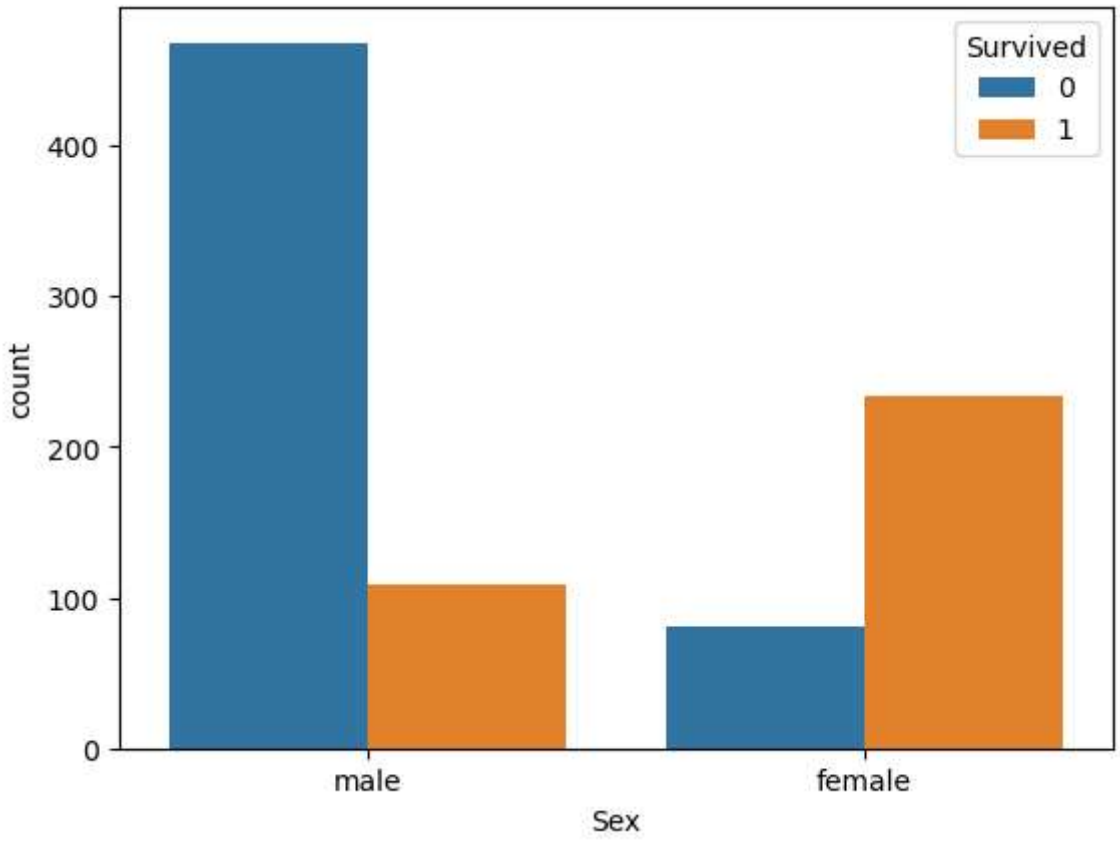
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

Data Visualization

In [4]:
```
sns.countplot(x='Survived', data=ds)
plt.show()
```

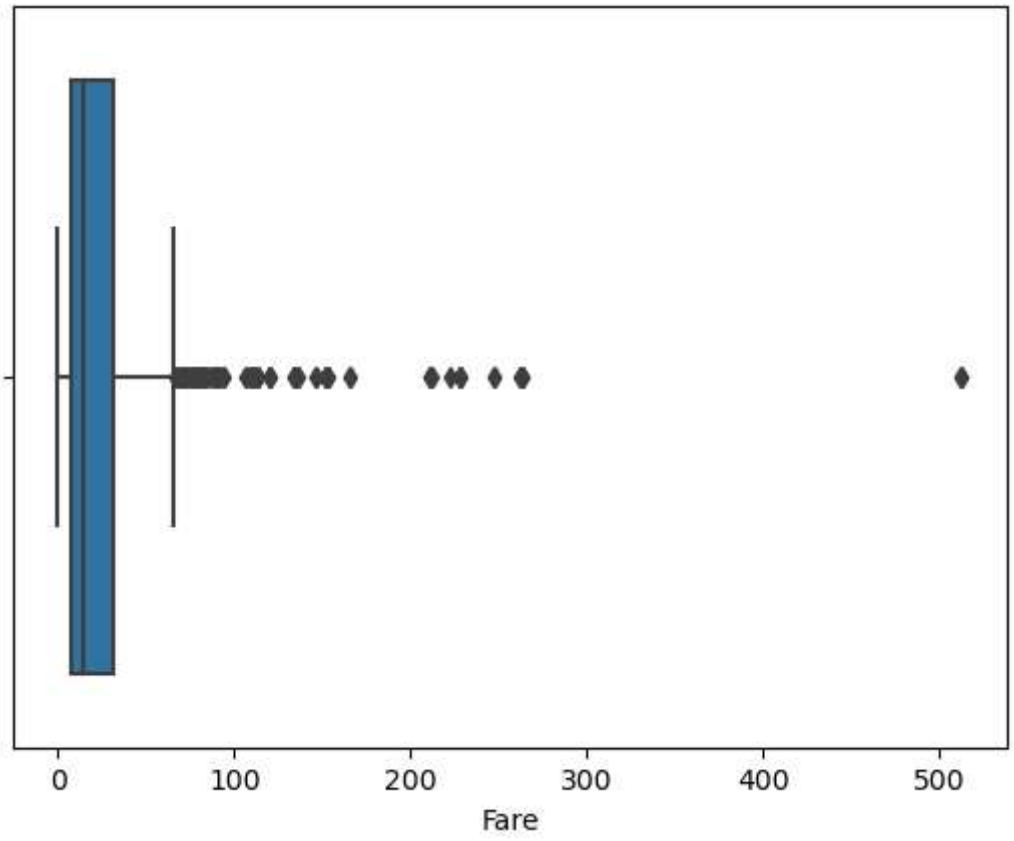


In [5]:
```
sns.countplot(x='Sex', hue='Survived', data=ds)
plt.show()
```



Outlier Detection

In [6]:
```
sns.boxplot(x='Fare', data=ds)
plt.show()
```



Splitting Dependent and Independent variables

In [7]:
```
x = ds.drop(['Survived', 'Name', 'Ticket', 'Cabin'], axis=1)
y = ds['Survived']
```

Perform Encoding

In [8]:
```
encoder = LabelEncoder()
x['Sex'] = encoder.fit_transform(x['Sex'])
x['Embarked'] = encoder.fit_transform(x['Embarked'])
```

Feature Scaling

In [9]:
```
scaler = StandardScaler()
x_scaled = scaler.fit_transform(x)
```

Splitting Data into Train and Test

In [10]:
```
x_train, x_test, y_train, y_test = train_test_split(x_scaled, y, test_size=0.2
```

Now i am going to do simple modeling using the train and test data.Simple linear regression model using the binary classification

Importing required

In [11]:
```
from sklearn.impute import SimpleImputer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import precision_score, recall_score, f1_score, confusior
from sklearn.metrics import accuracy_score, classification_report
```

Impute missing values with mean

In [12]:
```
imputer = SimpleImputer(strategy='mean')
x_train_imputed = imputer.fit_transform(x_train)
x_test_imputed = imputer.transform(x_test)
```

Train the logistic regression model on imputed data

In [13]:
```
lr_model = LogisticRegression(random_state=42)
lr_model.fit(x_train_imputed, y_train)
```

Out[13]:
```
LogisticRegression(random_state=42)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Predict on the test data

In [14]:
```
y_pred = lr_model.predict(x_test_imputed)
```

Accuracy Score

In [15]:
```
accuracy = accuracy_score(y_test, y_pred)
print('Accuracy:', accuracy)
```

```
Accuracy: 0.8100558659217877
```

Confusion Matrix

In [16]:
```
lr_confusion = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", lr_confusion)
```

```
Confusion Matrix:
 [[90 15]
 [19 55]]
```

Print a classification report for more detailed evaluation

In [17]:
```
classification_rep = classification_report(y_test, y_pred)
print("Classification Report:\n", classification_rep)
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.86      0.84       105
           1       0.79      0.74      0.76        74

    accuracy                           0.81       179
   macro avg       0.81      0.80      0.80       179
weighted avg       0.81      0.81      0.81       179
```

In [ ]:
```

```