PROJECT1: TITANIC DATASET CLEANING & VISUALIZATION

STEP1 : Import libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

STEP2 : Load a dataset

```
df = pd.read_csv("/content/titanic.csv")
```

STEP3 : Data cleaning

1. Finding Statistical information

```
df.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

2. Finding all type information

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

3. Find missing values sum

```
df.isna().sum()
```

	0
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

4. handling/filling null values

```
df["Age"].fillna(df["Age"].mean(), inplace=True)
df["Age"].astype("int64") #--> convert datatype from float to int
df["Embarked"].fillna(df["Embarked"].mode()[0], inplace=True)
df["Cabin"].fillna("Unknown", inplace=True)
```

```
<ipython-input-7-4d327177a7f3>:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the op


  df["Age"].fillna(df["Age"].mean(), inplace=True)
<ipython-input-7-4d327177a7f3>:3: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the op
```

```
  df["Embarked"].fillna(df["Embarked"].mode()[0], inplace=True)
<ipython-input-7-4d327177a7f3>:4: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the op


  df["Cabin"].fillna("Unknown", inplace=True)
```

5. Find duplicate value sum

```
df.duplicated().sum()
```

```
np.int64(0)
```

6. Find duplicates value from passenger id col

```
df["PassengerId"].duplicated().sum()
```

```
np.int64(0)
```

6. detect Null values

```
df.isna().sum()
```

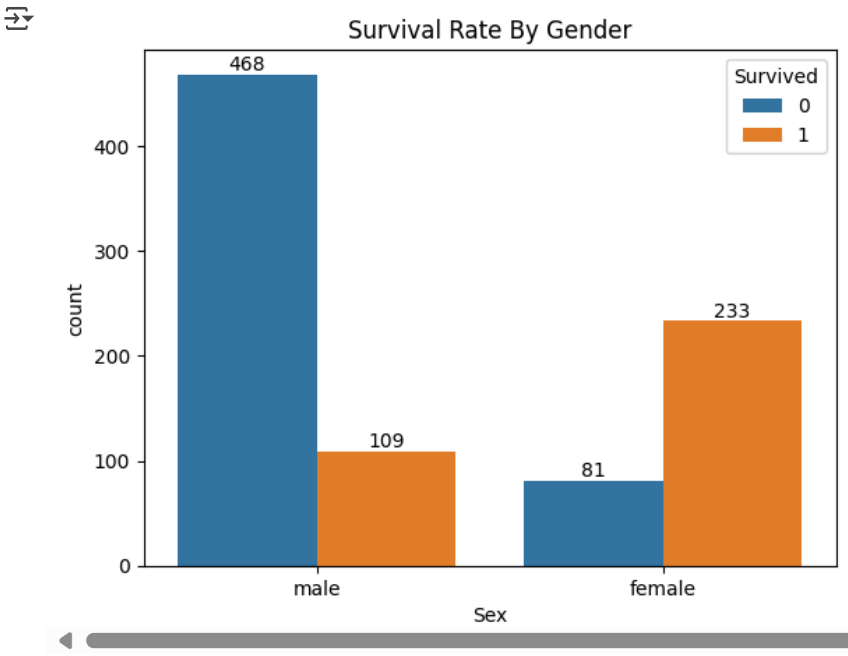
	0
PassengerId	0
Survived	0
Pclass	0
Name	0
Sex	0
Age	0
SibSp	0
Parch	0
Ticket	0
Fare	0
Cabin	0
Embarked	0

dtype: int64

STEP4 : Data visualization

QUE1 : How does the survival rate differ between males and females according to the chart?

```
sex = sns.countplot(data=df, x="Sex", hue="Survived")
sex.bar_label(sex.containers[0])
sex.bar_label(sex.containers[1])
plt.title("Survival Rate By Gender")
plt.show()
```

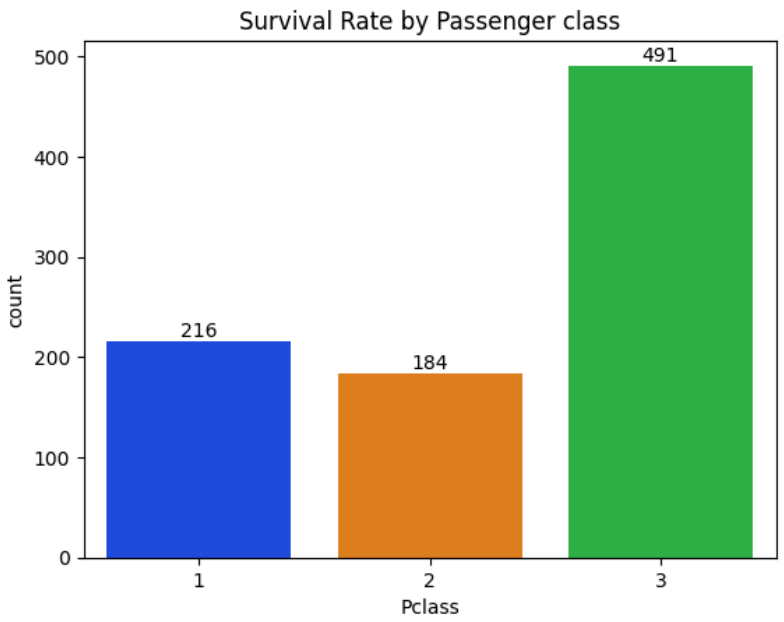


CONCLUSION : The chart highlights that females had significantly higher survival rates compared to males, with 233 females surviving and only 109 males.

QUE2 : What is the relationship between passenger class and survival rates as shown in the chart?"

```
pclass = sns.countplot(data=df, x="Pclass", palette="bright")
pclass.bar_label(pclass.containers[0])
pclass.bar_label(pclass.containers[1])
pclass.bar_label(pclass.containers[2])
plt.title("Survival Rate by Passenger class")
plt.show()
```

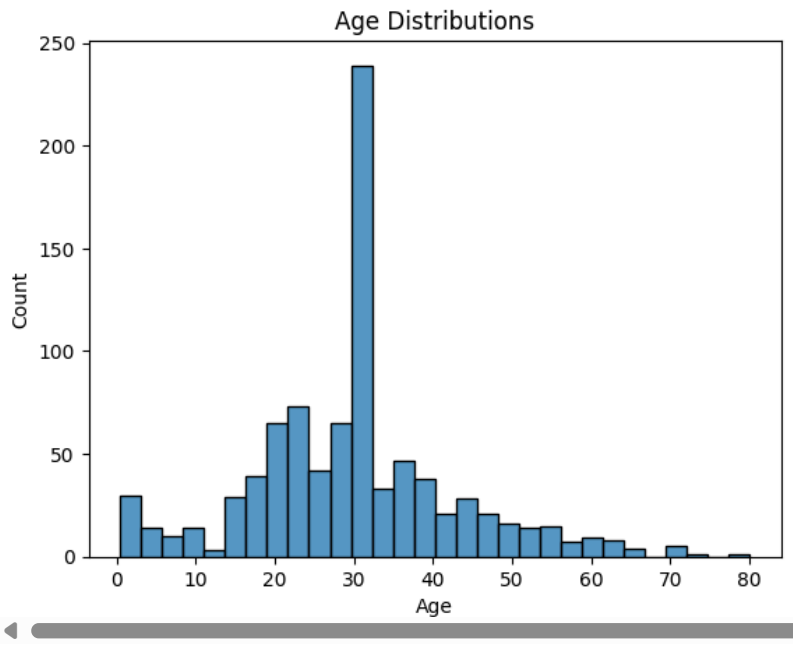
```
<ipython-input-11-d092ee8644d7>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  pclass = sns.countplot(data=df, x="Pclass", palette="bright")
```



CONCLUSION : The chart illustrates that the survival rate was highest for passengers in Class 3 (491 survivors), followed by Class 1 (216 survivors), and lowest in Class 2 (184 survivors).

QUE3 : What does the age distribution reveal about the concentration of passengers at different ages?

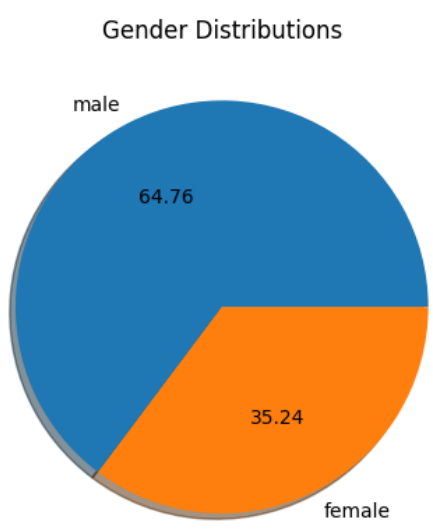
```
sns.histplot(data=df, x="Age")
plt.title("Age Distributions")
plt.show()
```



CONCLUSION : The chart highlights a notable peak in the population's age distribution at 30 years, indicating a higher concentration of individuals at this age compared to others.

QUE4 : What does the pie chart suggest about the proportion of males and females in the population?

```
gender_counts = df["Sex"].value_counts()
plt.pie(gender_counts.values, labels=gender_counts.index, autopct="%.2f", shadow=True, startangle=0)
plt.title("Gender Distributions")
plt.show()
```

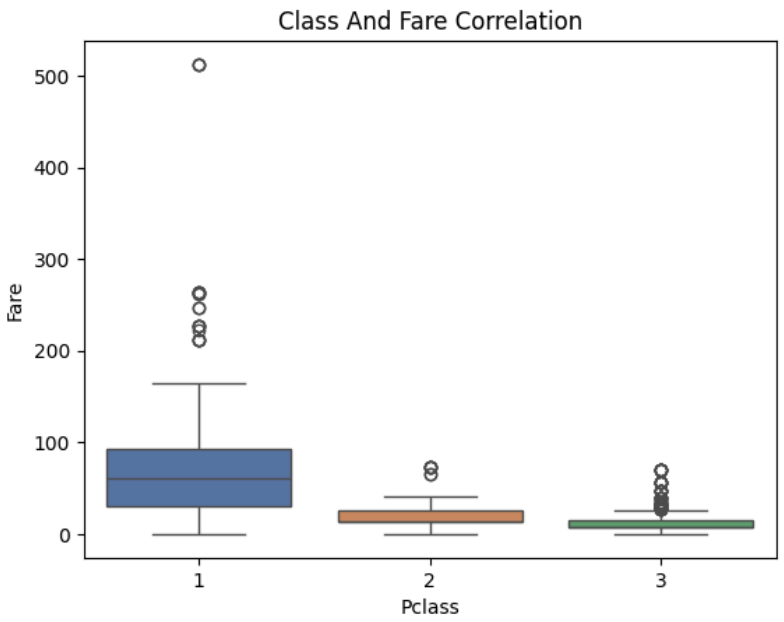


CONCLUSION : The pie chart reveals that the population consists of 64.76% males and 35.24% females, highlighting a male majority.

QUE5 : What relationship does the chart reveal between passenger class and fare variability, including the presence of outliers in higher classes?

```
sns.boxplot(data=df, x="Pclass", y="Fare", palette="deep")
plt.title("Class And Fare Correlation")
plt.show()
```

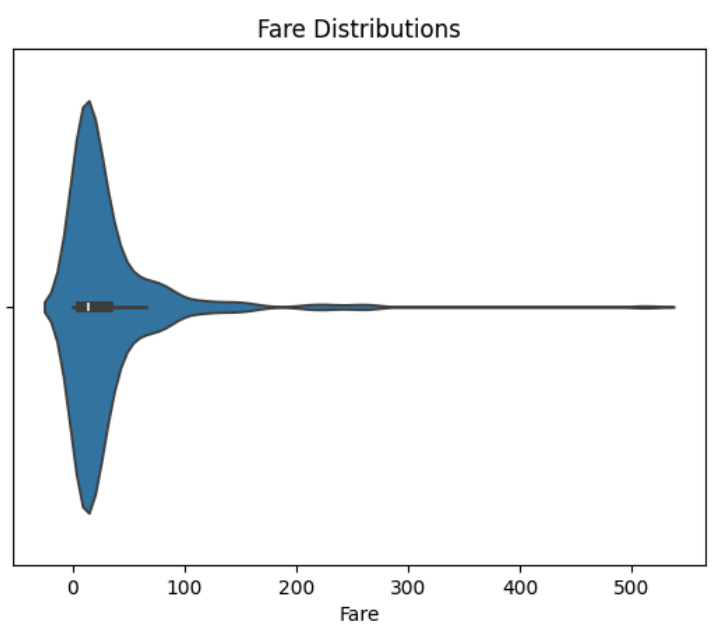
```
<ipython-input-14-c42d4372642f>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(data=df, x="Pclass", y="Fare", palette="deep")
```



CONCLUSION : The chart demonstrates that higher passenger classes are associated with higher fares, showing greater variability and outliers in first class compared to other classes.

QUE6 : How does the distribution of fare prices reflect the socio-economic diversity among Titanic passengers?

```
sns.violinplot(data=df, x="Fare")
plt.title("Fare Distributions")
plt.show()
```

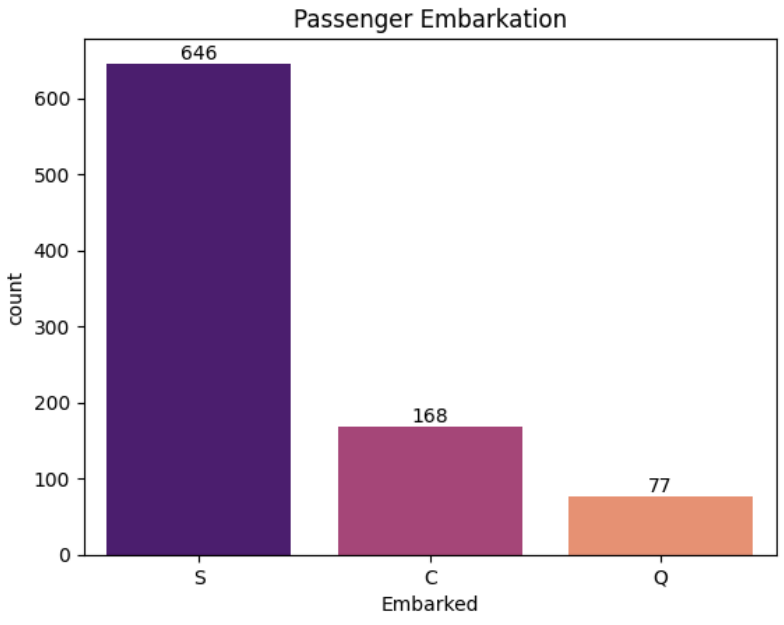


CONCLUSION : The chart shows that fare prices are heavily concentrated at lower values, with fewer high-value fares creating a long tail distribution.

QUE7 : How does the distribution of passengers by embarkation location reflect travel trends aboard the Titanic?

```
gb = df.groupby("Embarked")["PassengerId"].count()
print(gb)
embark = sns.countplot(data=df, x="Embarked", palette="magma")
embark.bar_label(embark.containers[0])
embark.bar_label(embark.containers[1])
embark.bar_label(embark.containers[2])
plt.title("Passenger Embarkation")
plt.show()
```

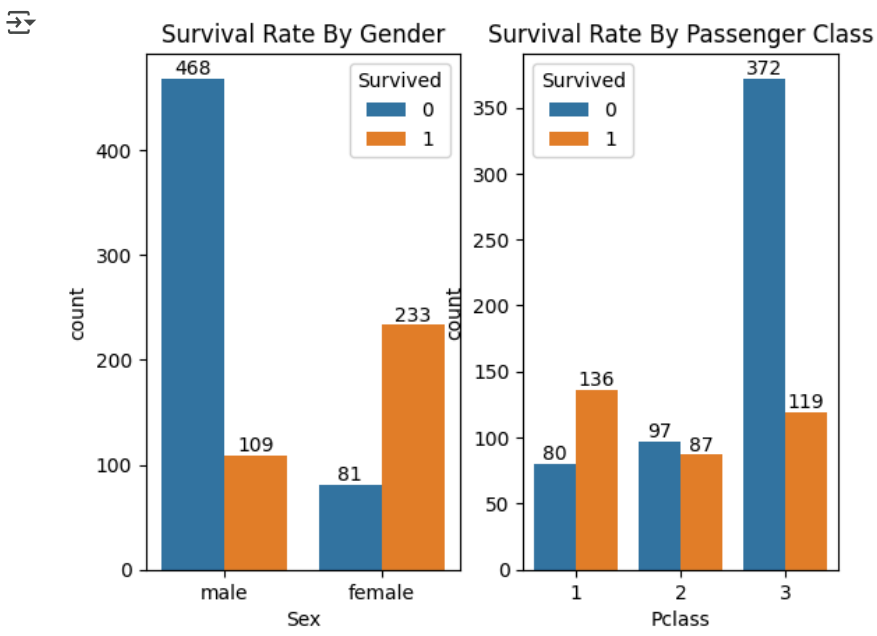
```
Embarked
C    168
Q     77
S    646
Name: PassengerId, dtype: int64
<ipython-input-16-264f14c56804>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  embark = sns.countplot(data=df, x="Embarked", palette="magma")
```



*CONCLUSION : The chart demonstrates that the majority of passengers, 646, embarked from location S, followed by 168 from location C, and 77 from location Q. *

QUE8 : How do gender and passenger class influence survival rates, as highlighted by the chart?

```
plt.subplot(1, 2, 1)
sex = sns.countplot(data=df, x="Sex", hue="Survived")
sex.bar_label(sex.containers[0])
sex.bar_label(sex.containers[1])
plt.title("Survival Rate By Gender")

plt.subplot(1, 2, 2)
pclass = sns.countplot(data=df, x="Pclass", hue="Survived")
pclass.bar_label(pclass.containers[0])
pclass.bar_label(pclass.containers[1])
# pclass.bar_label(pclass.containers[2])
plt.title("Survival Rate By Passenger Class")

plt.show()
```

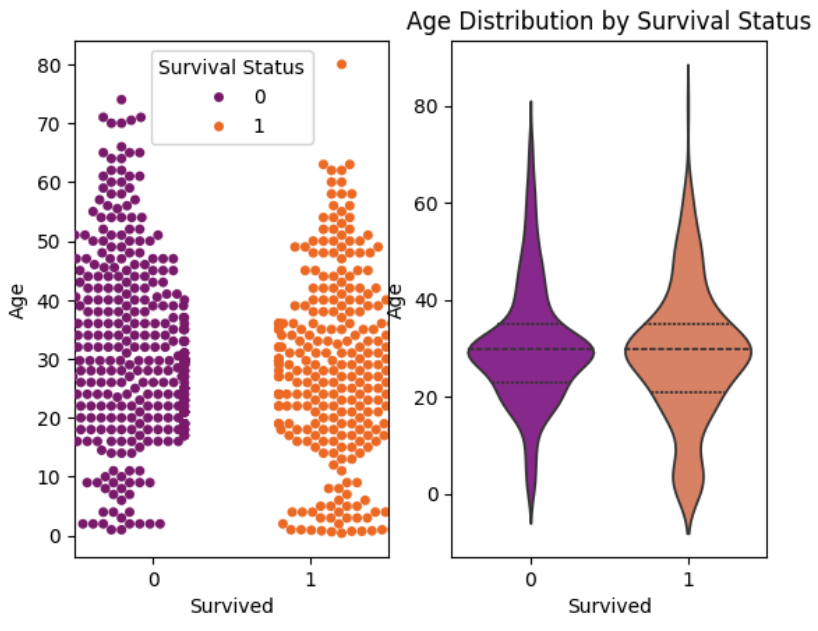


CONCLUSION : The chart reveals that females and first-class passengers had significantly higher survival rates compared to males and lower-class passengers.

QUE9 : What does the chart reveal about the relationship between age and survival rates on the Titanic?

```
# Swarm plot for Age distribution by Survival status
plt.subplot(1, 2, 1)
sns.swarmplot(x='Survived', y='Age', data=df, hue='Survived', dodge=True, palette='inferno')
plt.legend(title='Survival Status')

# Violin plot for Age distribution by Survival status
plt.subplot(1, 2, 2)
sns.violinplot(x='Survived', y='Age', data=df, palette='plasma', inner='quartile')

plt.title('Age Distribution by Survival Status')
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/seaborn/categorical.py:3399: UserWarning: 39.7% of the points cannot be placed; you may want to decrease the size of the markers or us
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.11/dist-packages/seaborn/categorical.py:3399: UserWarning: 19.3% of the points cannot be placed; you may want to decrease the size of the markers or us
  warnings.warn(msg, UserWarning)
<ipython-input-18-29d09d292824>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.violinplot(x='Survived', y='Age', data=df, palette='plasma', inner='quartile')
/usr/local/lib/python3.11/dist-packages/seaborn/categorical.py:3399: UserWarning: 47.5% of the points cannot be placed; you may want to decrease the size of the markers or us
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.11/dist-packages/seaborn/categorical.py:3399: UserWarning: 25.7% of the points cannot be placed; you may want to decrease the size of the markers or us
  warnings.warn(msg, UserWarning)
```

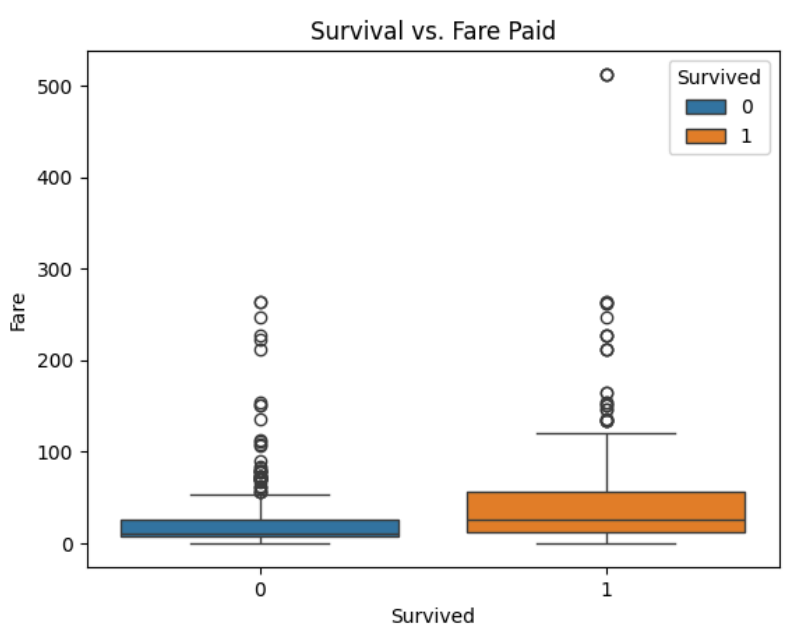


CONCLUSION : The chart highlights distinct age patterns, showing younger passengers had higher survival rates compared to older passengers.

QUE10 : What does the chart reveal about the relationship between fare amounts paid by passengers and their chances of survival?

```
sns.boxplot(data=df, x="Survived", y="Fare", hue="Survived")
plt.title("Survival vs. Fare Paid")
```

```
plt.show()
```



CONCLUSION : : The chart indicates that passengers who survived generally paid higher fares, as shown by the wider spread of fare values for survivors compared to non-survivors.

STEP5 : Save clean data

```
df.to_csv("Titanic Cleaned Data.csv", index=False)
print("Data Cleaning & Visualized Completed...")
print("Titanic data Cleaning & Visualized Project Done!...")
```

```
Data Cleaning & Visualized Completed...
Titanic data Cleaning & Visualized Project Done!...
```

 # Titanic Data Cleaning & Visualization Project Completed

```
Start coding or generate with AI.
```