2. Data Source: Titanic dataset from Kaggle

Description: This dataset contains information about Titanic passengers, including features such as age, class, sex, and survival status.

3. Explanation for Choosing This Data Set:

Reason: The Titanic dataset is commonly used for practice in data analysis. It has both numerical and categorical data, making it suitable for various analyses and insights.

In [6]:

```
# import libraries
import pandas as pd
import numpy as np
import os
```

In [8]:

```
# create path
path = r'C:\Users\Asus\Music\achievement 6 project'
```

In [10]:

```
# import Mallorca listings dataset
data = pd.read_csv(os.path.join(path, 'Data', 'tested.csv'), index_col = False)
data.head(
```

Out[10]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

3. Clean the Data

Check for Missing Values:

In [16]:

```
# Check for missing values
print(data.isnull().sum())
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

Handle Missing Values:

In [25]:

```
# Fill missing values in 'Age' with the median age
data['Age'] = data['Age'].fillna(data['Age'].median())

# Check if 'Ticket' and 'Cabin' columns exist before dropping them
columns_to_drop = ['Ticket', 'Cabin']
existing_columns_to_drop = [col for col in columns_to_drop if col in data.columns]

# Drop columns with too many missing values or irrelevant
data = data.drop(existing_columns_to_drop, axis=1)

# Drop rows with missing values in essential columns
data = data.dropna(subset=['Embarked'])
```

In [31]:

```
data.head()
```

Out[31]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Fare	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	7.8292	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	7.0000	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	9.6875	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	8.6625	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	12.2875	S

Check for Duplicates

In [34]:

```
# Check for duplicate rows
print(data.duplicated().sum())
```

```
0
```

Check for Consistency

In [37]:

```
# Check the unique values in 'Sex' and 'Embarked'
print(data['Sex'].value_counts())
print(data['Embarked'].value_counts())
```

```
Sex
male      266
female    152
Name: count, dtype: int64
Embarked
S    270
C    102
Q     46
Name: count, dtype: int64
```

4. Understand the Data

In [40]:

```
# Review Variables
# Display the first few rows and data types
print(data.head())
print(data.info())
```

```
   PassengerId  Survived  Pclass  \
0          892         0       3
1          893         1       3
2          894         0       2
3          895         0       3
4          896         1       3

                                           Name     Sex   Age  SibSp  Parch  \
0                              Kelly, Mr. James    male  34.5      0      0
1              Wilkes, Mrs. James (Ellen Needs)  female  47.0      1      0
2                     Myles, Mr. Thomas Francis    male  62.0      0      0
3                              Wirz, Mr. Albert    male  27.0      0      0
4  Hirvonen, Mrs. Alexander (Helga E Lindqvist)  female  22.0      1      1

      Fare Embarked
0   7.8292        Q
1   7.0000        S
2   9.6875        Q
3   8.6625        S
4  12.2875        S
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 10 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Survived     418 non-null    int64
 2   Pclass       418 non-null    int64
 3   Name         418 non-null    object
 4   Sex          418 non-null    object
 5   Age          418 non-null    float64
 6   SibSp        418 non-null    int64
 7   Parch        418 non-null    int64
 8   Fare         417 non-null    float64
 9   Embarked     418 non-null    object
dtypes: float64(2), int64(5), object(3)
memory usage: 32.8+ KB
None
```

Perform Descriptive Statistical Analysis:

In [43]:

```
# Display descriptive statistics
print(data.describe(include='all'))
```

```
        PassengerId    Survived      Pclass              Name   Sex  \
count    418.000000  418.000000  418.000000               418   418
unique          NaN         NaN         NaN               418     2
top             NaN         NaN         NaN  Kelly, Mr. James  male
freq            NaN         NaN         NaN                 1   266
mean    1100.500000    0.363636    2.265550               NaN   NaN
std      120.810458    0.481622    0.841838               NaN   NaN
min      892.000000    0.000000    1.000000               NaN   NaN
25%      996.250000    0.000000    1.000000               NaN   NaN
50%     1100.500000    0.000000    3.000000               NaN   NaN
75%     1204.750000    1.000000    3.000000               NaN   NaN
max     1309.000000    1.000000    3.000000               NaN   NaN

               Age       SibSp       Parch        Fare Embarked
count   418.000000  418.000000  418.000000  417.000000      418
unique         NaN         NaN         NaN         NaN        3
top            NaN         NaN         NaN         NaN        S
freq           NaN         NaN         NaN         NaN      270
mean     29.599282    0.447368    0.392344   35.627188      NaN
std      12.703770    0.896760    0.981429   55.907576      NaN
min       0.170000    0.000000    0.000000    0.000000      NaN
25%      23.000000    0.000000    0.000000    7.895800      NaN
50%      27.000000    0.000000    0.000000   14.454200      NaN
```

```
75%      35.750000    1.000000    0.000000   31.500000      NaN
max      76.000000    8.000000    9.000000  512.329200      NaN
```

```
In [ ]:
```