In [2]:

```python
import pandas as pd
import numpy as np

# Load Titanic dataset
df = pd.read_csv("train.csv")

# Display the first few rows and info
display(df.head())
df.info()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Far
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.250
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.283
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.925
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.100
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.050

```
<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
```

In [10]:

```python
# 1. Handling missing values

# Convert to numeric first (results in float64 due to NaNs and decimals)
df['Age'] = pd.to_numeric(df['Age'], errors='coerce')

# To convert to 'Int64', we must first handle the decimals.
# We'll round them to the nearest whole number, then cast.
df['Age'] = df['Age'].round(0).astype('Int64')

# Fill missing 'Embarked' with the most frequent value
```

```
df['Embarked'] = df['Embarked'].fillna(df['Embarked'].mode()[0])

# Fill missing 'Fare' with the median value
df['Fare'] = df['Fare'].fillna(df['Fare'].median())
```

In [12]:

```
# 2. Convert non-numeric columns to category types to save memory
df['Pclass'] = df['Pclass'].astype('category')
df['Survived'] = df['Survived'].astype('category')
df['Sex'] = df['Sex'].astype('category')
df['Embarked'] = df['Embarked'].astype('category')
```

In [28]:

```
# 3. Check the data types after the conversion and filling missing values
print("Data types after conversion:")
df.dtypes
```

```
Data types after conversion:
```

Out[28]:

```
PassengerId          int64
Survived          category
Pclass            category
Name                object
Sex               category
Age                  Int64
SibSp                int64
Parch                int64
Ticket              object
Fare               float64
Cabin               object
Embarked          category
Norm_Fare          float64
Standardized_Age   Float64
dtype: object
```

In [30]:

```
# 4. Apply Aggregation: Example for 'Age' and 'Fare'
age_stats = df['Age'].agg(['mean', 'std', 'min', 'max'])
fare_stats = df['Fare'].agg(['mean', 'std', 'min', 'max'])

# Display the summary statistics
print("\nAge Stats:")
age_stats
print("\nFare Stats:")
fare_stats
```

```
Age Stats:

Fare Stats:
```

Out[30]:

```
mean     32.204208
std      49.693429
min       0.000000
max     512.329200
Name: Fare, dtype: float64
```

In [32]:

```
# 5. Apply Normalization (Scaling)
# Normalize 'Fare' column: scale the 'Fare' to [0, 1] using Min-Max normalization
min_fare = df['Fare'].min()
max_fare = df['Fare'].max()
df['Norm_Fare'] = (df['Fare'] - min_fare) / (max_fare - min_fare)
```

In [34]:

```
# 6. Apply Standardization (Z-score scaling)
# Standardize the 'Age' column: mean = 0, std = 1
age_mean = df['Age'].mean()
age_std = df['Age'].std()
df['Standardized_Age'] = (df['Age'] - age_mean) / age_std
```

In [36]:

```
# 7. Display first few rows of the transformed dataframe
```

```
print("\nFirst few rows of the transformed dataframe:")
df.head()
```

```
First few rows of the transformed dataframe:
```

Out[36]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fa
0	1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.25
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38	1	0	PC 17599	71.28
2	3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.92
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.10
4	5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.05

In [39]:

```
# 8. Optional: Save the transformed dataframe to a new CSV file
df.to_csv('titanic_transformed.csv', index=False)
```