Import libraries

In [1]:
```
# Import the Pandas library, which is used for data manipulation and analy

import pandas as pd
```

In [2]:
```
# Import the NumPy library, which provides support for large, multi-dimens

import numpy as np
```

In [3]:
```
# Import the Seaborn library, which is built on top of Matplotlib and prov

import seaborn as sns
```

In [4]:
```
# Import the Matplotlib library, which is a popular plotting library for c

import matplotlib.pyplot as plt
```

Load The Data

In [5]:
```
# Load the Titanic dataset from the "tested.csv" file into a Pandas DataFr

titanic_data = pd.read_csv("tested.csv")
```

In [6]:
```
# Preview of data

titanic_data
```

Out[6]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298
...	...	...	...	...	...	...	...	...	...
413	1305	0	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236
414	1306	1	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758
415	1307	0	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262
416	1308	0	3	Ware, Mr. Frederick	male	NaN	0	0	359309
417	1309	0	3	Peter, Master. Michael J	male	NaN	1	1	2668

418 rows × 12 columns

In [7]:
```
# Length of Data

len(titanic_data)
```

Out[7]: 418

In [8]:
```
# Displaying the first few rows of the 'titanic_data' dataset using the .h

titanic_data.head()
```

Out[8]:

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

In [9]:
```
#  Get information about the index of the 'titanic_data' dataset.

titanic_data.index
```

Out[9]:
```
RangeIndex(start=0, stop=418, step=1)
```

In [10]:
```
# Retrieve the column names of the 'titanic_data' dataset.

titanic_data.columns
```

Out[10]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibS
p',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

Summary of DataFrame

In [11]:
```
# Display information about the 'titanic_data' dataset, including column r

titanic_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 12 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          332 non-null    float64
 6   SibSp        418 non-null    int64  
 7   Parch        418 non-null    int64  
 8   Ticket       418 non-null    object 
 9   Fare         417 non-null    float64
 10  Cabin        91 non-null     object 
 11  Embarked     418 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 39.3+ KB
```

In [12]:
```
# Retrieve the data types of each column in the 'titanic_data' dataset.

titanic_data.dtypes
```

Out[12]:
```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

In [13]:
```
# Generate a summary statistical description of numerical columns in the

titanic_data.describe()
```

Out[13]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fa
count	418.000000	418.000000	418.000000	332.000000	418.000000	418.000000	417.00000
mean	1100.500000	0.363636	2.265550	30.272590	0.447368	0.392344	35.62718
std	120.810458	0.481622	0.841838	14.181209	0.896760	0.981429	55.90757
min	892.000000	0.000000	1.000000	0.170000	0.000000	0.000000	0.00000
25%	996.250000	0.000000	1.000000	21.000000	0.000000	0.000000	7.89580
50%	1100.500000	0.000000	3.000000	27.000000	0.000000	0.000000	14.45420
75%	1204.750000	1.000000	3.000000	39.000000	1.000000	0.000000	31.50000
max	1309.000000	1.000000	3.000000	76.000000	8.000000	9.000000	512.32920

In [14]:
```
# Generate a summary statistical description of categorical (object) colur

titanic_data.describe(include = 'O')
```

Out[14]:

	Name	Sex	Ticket	Cabin	Embarked
count	418	418	418	91	418
unique	418	2	363	76	3
top	Kelly, Mr. James	male	PC 17608	B57 B59 B63 B66	S
freq	1	266	5	3	270

In [15]:
```
# Calculate the number of duplicated rows in the 'titanic_data' dataset ar

dup = titanic_data.duplicated().sum()

# Print the result to the console.

print("The number of duplicated values in the dataset are: ", dup)
```

```
The number of duplicated values in the dataset are:  0
```

In [16]:

```
# Iterate through categorical (object) columns in the 'titanic_data' datas

for col in titanic_data.select_dtypes(include = "object"):

    # Print the name of the column.
    print(f"Name of Column: {col}")

    # Print the unique values in the column.
    print(titanic_data[col].unique())

    # Print a separator for clarity.
    print('\n', '-'*60, '\n')
```

```
Name of Column: Name
['Kelly, Mr. James' 'Wilkes, Mrs. James (Ellen Needs)'
 'Myles, Mr. Thomas Francis' 'Wirz, Mr. Albert'
 'Hirvonen, Mrs. Alexander (Helga E Lindqvist)'
 'Svensson, Mr. Johan Cervin' 'Connolly, Miss. Kate'
 'Caldwell, Mr. Albert Francis'
 'Abrahim, Mrs. Joseph (Sophie Halaut Easu)' 'Davies, Mr. John Samuel'
 'Ilieff, Mr. Ylio' 'Jones, Mr. Charles Cresson'
 'Snyder, Mrs. John Pillsbury (Nelle Stevenson)' 'Howard, Mr. Benjamin'
 'Chaffee, Mrs. Herbert Fuller (Carrie Constance Toogood)'
 'del Carlo, Mrs. Sebastiano (Argenia Genovesi)' 'Keane, Mr. Daniel'
 'Assaf, Mr. Gerios' 'Ilmakangas, Miss. Ida Livija'
 'Assaf Khalil, Mrs. Mariana (Miriam")"' 'Rothschild, Mr. Martin'
 'Olsen, Master. Artur Karl' 'Flegenheim, Mrs. Alfred (Antoinette)'
 'Williams, Mr. Richard Norris II'
 'Ryerson, Mrs. Arthur Larned (Emily Maria Borie)'
 'Robins, Mr. Alexander A' 'Ostby, Miss. Helene Ragnhild'
 'Daher, Mr. Shedid' 'Brady, Mr. John Bertram' 'Samaan, Mr. Elias'
 'Louch, Mr. Charles Alexander' 'Jefferys, Mr. Clifford Thomas'
 'Dean, Mrs. Bertram (Eva Georgetta Light)'
 'Johnston, Mrs. Andrew G (Elizabeth Lily" Watson)"'
 'Mock, Mr. Philipp Edmund'
 'Katavelas, Mr. Vassilios (Catavelas Vassilios")"' 'Roth, Miss. Sarah
A'
 'Cacic, Miss. Manda' 'Sap, Mr. Julius' 'Hee, Mr. Ling' 'Karun, Mr. Fran
z'
 'Franklin, Mr. Thomas Parham' 'Goldsmith, Mr. Nathan'
 'Corbett, Mrs. Walter H (Irene Colvin)'
 'Kimball, Mrs. Edwin Nelson Jr (Gertrude Parsons)'
 'Peltomaki, Mr. Nikolai Johannes' 'Chevre, Mr. Paul Romaine'
 'Shaughnessy, Mr. Patrick'
 'Bucknell, Mrs. William Robert (Emma Eliza Ward)'
 'Coutts, Mrs. William (Winnie Minnie" Treanor)"'
 'Smith, Mr. Lucien Philip' 'Pulbaum, Mr. Franz'
 'Hocking, Miss. Ellen Nellie""' 'Fortune, Miss. Ethel Flora'
 'Mangiavacchi, Mr. Serafino Emilio' 'Rice, Master. Albert'
 'Cor, Mr. Bartol' 'Abelseth, Mr. Olaus Jorgensen'
 'Davison, Mr. Thomas Henry' 'Chaudanson, Miss. Victorine'
 'Dika, Mr. Mirko' 'McCrae, Mr. Arthur Gordon'
 'Bjorklund, Mr. Ernst Herbert' 'Bradley, Miss. Bridget Delia'
 'Ryerson, Master. John Borie'
 'Corey, Mrs. Percy C (Mary Phyllis Elizabeth Miller)'
 'Burns, Miss. Mary Delia' 'Moore, Mr. Clarence Bloomfield'
 'Tucker, Mr. Gilbert Milligan Jr' 'Fortune, Mrs. Mark (Mary McDougald)'
 'Mulvihill, Miss. Bertha E' 'Minkoff, Mr. Lazar'
 'Nieminen, Miss. Manta Josefina' 'Ovies y Rodriguez, Mr. Servando'
 'Geiger, Miss. Amalie' 'Keeping, Mr. Edwin' 'Miles, Mr. Frank'
 'Cornell, Mrs. Robert Clifford (Malvina Helen Lamson)'
 'Aldworth, Mr. Charles Augustus' 'Doyle, Miss. Elizabeth'
 'Boulos, Master. Akar' 'Straus, Mr. Isidor' 'Case, Mr. Howard Brown'
 'Demetri, Mr. Marinko' 'Lamb, Mr. John Joseph' 'Khalil, Mr. Betros'
 'Barry, Miss. Julia' 'Badman, Miss. Emily Louisa'
 "O'Donoghue, Ms. Bridget" 'Wells, Master. Ralph Lester'
 'Dyker, Mrs. Adolf Fredrik (Anna Elisabeth Judith Andersson)'
 'Pedersen, Mr. Olaf' 'Davidson, Mrs. Thornton (Orian Hays)'
 'Guest, Mr. Robert' 'Birnbaum, Mr. Jakob' 'Tenglin, Mr. Gunnar Isidor'
 'Cavendish, Mrs. Tyrell William (Julia Florence Siegel)'
```

```
 'Makinen, Mr. Kalle Edvard' 'Braf, Miss. Elin Ester Maria'
 'Nancarrow, Mr. William Henry'
 'Stengel, Mrs. Charles Emil Henry (Annie May Morris)'
 'Weisz, Mr. Leopold' 'Foley, Mr. William'
 'Johansson Palmquist, Mr. Oskar Leander'
 'Thomas, Mrs. Alexander (Thamine Thelma")"' 'Holthen, Mr. Johan Martin'
 'Buckley, Mr. Daniel' 'Ryan, Mr. Edward'
 'Willer, Mr. Aaron (Abi Weller")"' 'Swane, Mr. George'
 'Stanton, Mr. Samuel Ward' 'Shine, Miss. Ellen Natalia'
 'Evans, Miss. Edith Corse' 'Buckley, Miss. Katherine'
 'Straus, Mrs. Isidor (Rosalie Ida Blun)' 'Chronopoulos, Mr. Demetrios'
 'Thomas, Mr. John' 'Sandstrom, Miss. Beatrice Irene'
 'Beattie, Mr. Thomson' 'Chapman, Mrs. John Henry (Sara Elizabeth Lawr
y)'
 'Watt, Miss. Bertha J' 'Kiernan, Mr. John'
 'Schabert, Mrs. Paul (Emma Mock)' 'Carver, Mr. Alfred John'
 'Kennedy, Mr. John' 'Cribb, Miss. Laura Alice' 'Brobeck, Mr. Karl Rudol
f'
 'McCoy, Miss. Alicia' 'Bowenur, Mr. Solomon' 'Petersen, Mr. Marius'
 'Spinner, Mr. Henry John' 'Gracie, Col. Archibald IV'
 'Lefebre, Mrs. Frank (Frances)' 'Thomas, Mr. Charles P'
 'Dintcheff, Mr. Valtcho' 'Carlsson, Mr. Carl Robert'
 'Zakarian, Mr. Mapriededer' 'Schmidt, Mr. August' 'Drapkin, Miss. Jenni
e'
 'Goodwin, Mr. Charles Frederick' 'Goodwin, Miss. Jessie Allis'
 'Daniels, Miss. Sarah' 'Ryerson, Mr. Arthur Larned'
 'Beauchamp, Mr. Henry James'
 'Lindeberg-Lind, Mr. Erik Gustaf (Mr Edward Lingrey")"'
 'Vander Planke, Mr. Julius' 'Hilliard, Mr. Herbert Henry'
 'Davies, Mr. Evan' 'Crafton, Mr. John Bertram' 'Lahtinen, Rev. William'
 'Earnshaw, Mrs. Boulton (Olive Potter)' 'Matinoff, Mr. Nicola'
 'Storey, Mr. Thomas' 'Klasen, Mrs. (Hulda Kristina Eugenia Lofqvist)'
 'Asplund, Master. Filip Oscar' 'Duquemin, Mr. Joseph' 'Bird, Miss. Elle
n'
 'Lundin, Miss. Olga Elida' 'Borebank, Mr. John James'
 'Peacock, Mrs. Benjamin (Edith Nile)' 'Smyth, Miss. Julia'
 'Touma, Master. Georges Youssef' 'Wright, Miss. Marion'
 'Pearce, Mr. Ernest' 'Peruschitz, Rev. Joseph Maria'
 'Kink-Heilmann, Mrs. Anton (Luise Heilmann)' 'Brandeis, Mr. Emil'
 'Ford, Mr. Edward Watson'
 'Cassebeer, Mrs. Henry Arthur Jr (Eleanor Genevieve Fosdick)'
 'Hellstrom, Miss. Hilda Maria' 'Lithman, Mr. Simon' 'Zakarian, Mr. Orti
n'
 'Dyker, Mr. Adolf Fredrik' 'Torfa, Mr. Assad'
 'Asplund, Mr. Carl Oscar Vilhelm Gustafsson' 'Brown, Miss. Edith Eilee
n'
 'Sincock, Miss. Maude' 'Stengel, Mr. Charles Emil Henry'
 'Becker, Mrs. Allen Oliver (Nellie E Baumgardner)'
 'Compton, Mrs. Alexander Taylor (Mary Eliza Ingersoll)'
 'McCrie, Mr. James Matthew' 'Compton, Mr. Alexander Taylor Jr'
 'Marvin, Mrs. Daniel Warner (Mary Graham Carmichael Farquarson)'
 'Lane, Mr. Patrick'
 'Douglas, Mrs. Frederick Charles (Mary Helene Baxter)'
 'Maybery, Mr. Frank Hubert' 'Phillips, Miss. Alice Frances Louisa'
 'Davies, Mr. Joseph' 'Sage, Miss. Ada' 'Veal, Mr. James'
 'Angle, Mr. William A' 'Salomon, Mr. Abraham L'
 'van Billiard, Master. Walter John' 'Lingane, Mr. John'
```

```
 'Drew, Master. Marshall Brines' 'Karlsson, Mr. Julius Konrad Eugen'
 'Spedden, Master. Robert Douglas' 'Nilsson, Miss. Berta Olivia'
 'Baimbrigge, Mr. Charles Robert'
 'Rasmussen, Mrs. (Lena Jacobsen Solvang)' 'Murphy, Miss. Nora'
 'Danbom, Master. Gilbert Sigvard Emanuel' 'Astor, Col. John Jacob'
 'Quick, Miss. Winifred Vera' 'Andrew, Mr. Frank Thomas'
 'Omont, Mr. Alfred Fernand' 'McGowan, Miss. Katherine'
 'Collett, Mr. Sidney C Stuart' 'Rosenbaum, Miss. Edith Louise'
 'Delalic, Mr. Redjo' 'Andersen, Mr. Albert Karvin' 'Finoli, Mr. Luigi'
 'Deacon, Mr. Percy William'
 'Howard, Mrs. Benjamin (Ellen Truelove Arman)'
 'Andersson, Miss. Ida Augusta Margareta' 'Head, Mr. Christopher'
 'Mahon, Miss. Bridget Delia' 'Wick, Mr. George Dennick'
 'Widener, Mrs. George Dunton (Eleanor Elkins)'
 'Thomson, Mr. Alexander Morrison' 'Duran y More, Miss. Florentina'
 'Reynolds, Mr. Harold J' 'Cook, Mrs. (Selena Rogers)'
 'Karlsson, Mr. Einar Gervasius'
 'Candee, Mrs. Edward (Helen Churchill Hungerford)'
 'Moubarek, Mrs. George (Omine Amenia" Alexander)"'
 'Asplund, Mr. Johan Charles' 'McNeill, Miss. Bridget'
 'Everett, Mr. Thomas James' 'Hocking, Mr. Samuel James Metcalfe'
 'Sweet, Mr. George Frederick' 'Willard, Miss. Constance'
 'Wiklund, Mr. Karl Johan' 'Linehan, Mr. Michael'
 'Cumings, Mr. John Bradley' 'Vendel, Mr. Olof Edvin'
 'Warren, Mr. Frank Manley' 'Baccos, Mr. Raffull' 'Hiltunen, Miss. Mart
a'
 'Douglas, Mrs. Walter Donald (Mahala Dutton)'
 'Lindstrom, Mrs. Carl Johan (Sigrid Posse)'
 'Christy, Mrs. (Alice Frances)' 'Spedden, Mr. Frederic Oakley'
 'Hyman, Mr. Abraham' 'Johnston, Master. William Arthur Willie""'
 'Kenyon, Mr. Frederick R' 'Karnes, Mrs. J Frank (Claire Bennett)'
 'Drew, Mr. James Vivian' 'Hold, Mrs. Stephen (Annie Margaret Hill)'
 'Khalil, Mrs. Betros (Zahie Maria" Elias)"' 'West, Miss. Barbara J'
 'Abrahamsson, Mr. Abraham August Johannes' 'Clark, Mr. Walter Miller'
 'Salander, Mr. Karl Johan' 'Wenzel, Mr. Linhart'
 'MacKay, Mr. George William' 'Mahon, Mr. John' 'Niklasson, Mr. Samuel'
 'Bentham, Miss. Lilian W' 'Midtsjo, Mr. Karl Albert'
 'de Messemaeker, Mr. Guillaume Joseph' 'Nilsson, Mr. August Ferdinand'
 'Wells, Mrs. Arthur Henry (Addie" Dart Trevaskis)"'
 'Klasen, Miss. Gertrud Emilia' 'Portaluppi, Mr. Emilio Ilario Giuseppe'
 'Lyntakoff, Mr. Stanko' 'Chisholm, Mr. Roderick Robert Crispin'
 'Warren, Mr. Charles William' 'Howard, Miss. May Elizabeth'
 'Pokrnic, Mr. Mate' 'McCaffry, Mr. Thomas Francis' 'Fox, Mr. Patrick'
 'Clark, Mrs. Walter Miller (Virginia McDowell)' 'Lennon, Miss. Mary'
 'Saade, Mr. Jean Nassr' 'Bryhl, Miss. Dagmar Jenny Ingeborg '
 'Parker, Mr. Clifford Richard' 'Faunthorpe, Mr. Harry'
 'Ware, Mr. John James' 'Oxenham, Mr. Percy Thomas'
 'Oreskovic, Miss. Jelka' 'Peacock, Master. Alfred Edward'
 'Fleming, Miss. Honora' 'Touma, Miss. Maria Youssef'
 'Rosblom, Miss. Salli Helena' 'Dennis, Mr. William'
 'Franklin, Mr. Charles (Charles Fardon)' 'Snyder, Mr. John Pillsbury'
 'Mardirosian, Mr. Sarkis' 'Ford, Mr. Arthur'
 'Rheims, Mr. George Alexander Lucien'
 'Daly, Miss. Margaret Marcella Maggie""' 'Nasr, Mr. Mustafa'
 'Dodge, Dr. Washington' 'Wittevrongel, Mr. Camille'
 'Angheloff, Mr. Minko' 'Laroche, Miss. Louise' 'Samaan, Mr. Hanna'
 'Loring, Mr. Joseph Holland' 'Johansson, Mr. Nils'
```

```
 'Olsson, Mr. Oscar Wilhelm' 'Malachard, Mr. Noel'
 'Phillips, Mr. Escott Robert' 'Pokrnic, Mr. Tome'
 'McCarthy, Miss. Catherine Katie""'
 'Crosby, Mrs. Edward Gifford (Catherine Elizabeth Halstead)'
 'Allison, Mr. Hudson Joshua Creighton' 'Aks, Master. Philip Frank'
 'Hays, Mr. Charles Melville' 'Hansen, Mrs. Claus Peter (Jennie L Howar
d)'
 'Cacic, Mr. Jego Grga' 'Vartanian, Mr. David' 'Sadowitz, Mr. Harry'
 'Carr, Miss. Jeannie' 'White, Mrs. John Stuart (Ella Holmes)'
 'Hagardon, Miss. Kate' 'Spencer, Mr. William Augustus'
 'Rogers, Mr. Reginald Harry' 'Jonsson, Mr. Nils Hilding'
 'Jefferys, Mr. Ernest Wilfred' 'Andersson, Mr. Johan Samuel'
 'Krekorian, Mr. Neshan' 'Nesson, Mr. Israel' 'Rowe, Mr. Alfred G'
 'Kreuchen, Miss. Emilie' 'Assam, Mr. Ali' 'Becker, Miss. Ruth Elizabet
h'
 'Rosenshine, Mr. George (Mr George Thorne")"'
 'Clarke, Mr. Charles Valentine' 'Enander, Mr. Ingvar'
 'Davies, Mrs. John Morgan (Elizabeth Agnes Mary White) '
 'Dulles, Mr. William Crothers' 'Thomas, Mr. Tannous'
 'Nakid, Mrs. Said (Waika Mary" Mowad)"' 'Cor, Mr. Ivan'
 'Maguire, Mr. John Edward' 'de Brito, Mr. Jose Joaquim'
 'Elias, Mr. Joseph' 'Denbury, Mr. Herbert' 'Betros, Master. Seman'
 'Fillbrook, Mr. Joseph Charles' 'Lundstrom, Mr. Thure Edvin'
 'Sage, Mr. John George'
 'Cardeza, Mrs. James Warburton Martinez (Charlotte Wardle Drake)'
 'van Billiard, Master. James William' 'Abelseth, Miss. Karen Marie'
 'Botsford, Mr. William Hull'
 'Whabee, Mrs. George Joseph (Shawneene Abi-Saab)' 'Giles, Mr. Ralph'
 'Walcroft, Miss. Nellie' 'Greenfield, Mrs. Leo David (Blanche Strouse)'
 'Stokes, Mr. Philip Joseph' 'Dibden, Mr. William' 'Herman, Mr. Samuel'
 'Dean, Miss. Elizabeth Gladys Millvina""' 'Julian, Mr. Henry Forbes'
 'Brown, Mrs. John Murray (Caroline Lane Lamson)' 'Lockyer, Mr. Edward'
 "O'Keefe, Mr. Patrick"
 'Lindell, Mrs. Edvard Bengtsson (Elin Gerda Persson)'
 'Sage, Master. William Henry' 'Mallet, Mrs. Albert (Antoinette Magnin)'
 'Ware, Mrs. John James (Florence Louise Long)' 'Strilic, Mr. Ivan'
 'Harder, Mrs. George Achilles (Dorothy Annan)'
 'Sage, Mrs. John (Annie Bullen)' 'Caram, Mr. Joseph'
 'Riihivouri, Miss. Susanna Juhantytar Sanni""'
 'Gibson, Mrs. Leonard (Pauline C Boeson)' 'Pallas y Castello, Mr. Emili
o'
 'Giles, Mr. Edgar' 'Wilson, Miss. Helen Alice' 'Ismay, Mr. Joseph Bruc
e'
 'Harbeck, Mr. William H' 'Dodge, Mrs. Washington (Ruth Vidaver)'
 'Bowen, Miss. Grace Scott' 'Kink, Miss. Maria'
 'Cotterill, Mr. Henry Harry""' 'Hipkins, Mr. William Edward'
 'Asplund, Master. Carl Edgar' "O'Connor, Mr. Patrick" 'Foley, Mr. Josep
h'
 'Risien, Mrs. Samuel (Emma)' "McNamee, Mrs. Neal (Eileen O'Leary)"
 'Wheeler, Mr. Edwin Frederick""' 'Herman, Miss. Kate'
 'Aronsson, Mr. Ernst Axel Algot' 'Ashby, Mr. John' 'Canavan, Mr. Patric
k'
 'Palsson, Master. Paul Folke' 'Payne, Mr. Vivian Ponsonby'
 'Lines, Mrs. Ernest H (Elizabeth Lindsey James)'
 'Abbott, Master. Eugene Joseph' 'Gilbert, Mr. William'
 'Kink-Heilmann, Mr. Anton'
 'Smith, Mrs. Lucien Philip (Mary Eloise Hughes)' 'Colbert, Mr. Patrick'
```

```
 'Frolicher-Stehli, Mrs. Maxmillian (Margaretha Emerentia Stehli)'
 'Larsson-Rondberg, Mr. Edvard A' 'Conlon, Mr. Thomas Henry'
 'Bonnell, Miss. Caroline' 'Gale, Mr. Harry'
 'Gibson, Miss. Dorothy Winifred' 'Carrau, Mr. Jose Pedro'
 'Frauenthal, Mr. Isaac Gerald'
 'Nourney, Mr. Alfred (Baron von Drachstedt")"'
 'Ware, Mr. William Jeffery' 'Widener, Mr. George Dunton'
 'Riordan, Miss. Johanna Hannah""' 'Peacock, Miss. Treasteall'
 'Naughton, Miss. Hannah'
 'Minahan, Mrs. William Edward (Lillian E Thorpe)'
 'Henriksson, Miss. Jenny Lovisa' 'Spector, Mr. Woolf'
 'Oliva y Ocana, Dona. Fermina' 'Saether, Mr. Simon Sivertsen'
 'Ware, Mr. Frederick' 'Peter, Master. Michael J']

 ------------------------------------------------------------

Name of Column: Sex
['male' 'female']

 ------------------------------------------------------------

Name of Column: Ticket
['330911' '363272' '240276' '315154' '3101298' '7538' '330972' '248738'
 '2657' 'A/4 48871' '349220' '694' '21228' '24065' 'W.E.P. 5734'
 'SC/PARIS 2167' '233734' '2692' 'STON/O2. 3101270' '2696' 'PC 17603'
 'C 17368' 'PC 17598' 'PC 17597' 'PC 17608' 'A/5. 3337' '113509' '2698'
 '113054' '2662' 'SC/AH 3085' 'C.A. 31029' 'C.A. 2315' 'W./C. 6607'
 '13236' '2682' '342712' '315087' '345768' '1601' '349256' '113778'
 'SOTON/O.Q. 3101263' '237249' '11753' 'STON/O 2. 3101291' 'PC 17594'
 '370374' '11813' 'C.A. 37671' '13695' 'SC/PARIS 2168' '29105' '19950'
 'SC/A.3 2861' '382652' '349230' '348122' '386525' '349232' '237216'
 '347090' '334914' 'F.C.C. 13534' '330963' '113796' '2543' '382653'
 '349211' '3101297' 'PC 17562' '113503' '359306' '11770' '248744' '36870
2'
 '2678' 'PC 17483' '19924' '349238' '240261' '2660' '330844' 'A/4 31416'
 '364856' '29103' '347072' '345498' 'F.C. 12750' '376563' '13905' '35003
3'
 '19877' 'STON/O 2. 3101268' '347471' 'A./5. 3338' '11778' '228414'
 '365235' '347070' '2625' 'C 4001' '330920' '383162' '3410' '248734'
 '237734' '330968' 'PC 17531' '329944' '2680' '2681' 'PP 9549' '13050'
 'SC/AH 29037' 'C.A. 33595' '367227' '392095' '368783' '371362' '350045'
 '367226' '211535' '342441' 'STON/OQ. 369943' '113780' '4133' '2621'
 '349226' '350409' '2656' '248659' 'SOTON/OQ 392083' 'CA 2144' '113781'
 '244358' '17475' '345763' '17463' 'SC/A4 23568' '113791' '250651' '1176
7'
 '349255' '3701' '350405' '347077' 'S.O./P.P. 752' '347469' '110489'
 'SOTON/O.Q. 3101315' '335432' '2650' '220844' '343271' '237393' '31515
3'
 'PC 17591' 'W./C. 6608' '17770' '7548' 'S.O./P.P. 251' '2670' '2673'
 '29750' 'C.A. 33112' '230136' 'PC 17756' '233478' '113773' '7935'
 'PC 17558' '239059' 'S.O./P.P. 2' 'A/4 48873' 'CA. 2343' '28221' '22687
5'
 '111163' 'A/5. 851' '235509' '28220' '347465' '16966' '347066'
 'C.A. 31030' '65305' '36568' '347080' 'PC 17757' '26360' 'C.A. 34050'
 'F.C. 12998' '9232' '28034' 'PC 17613' '349250' 'SOTON/O.Q. 3101308'
 'S.O.C. 14879' '347091' '113038' '330924' '36928' '32302' 'SC/PARIS 214
8'
```

```
 '342684' 'W./C. 14266' '350053' 'PC 17606' '2661' '350054' '370368'
 'C.A. 6212' '242963' '220845' '113795' '3101266' '330971' 'PC 17599'
 '350416' '110813' '2679' '250650' 'PC 17761' '112377' '237789' '3470'
 '17464' '26707' 'C.A. 34651' 'SOTON/O2 3101284' '13508' '7266' '345775'
 'C.A. 42795' 'AQ/4 3130' '363611' '28404' '345501' '345572' '350410'
 'C.A. 34644' '349235' '112051' 'C.A. 49867' 'A. 2. 39186' '315095'
 '368573' '370371' '2676' '236853' 'SC 14888' '2926' 'CA 31352'
 'W./C. 14260' '315085' '364859' '370129' 'A/5 21175' 'SOTON/O.Q. 310131
4'
 '2655' 'A/5 1478' 'PC 17607' '382650' '2652' '33638' '345771' '349202'
 'SC/Paris 2123' '113801' '347467' '347079' '237735' '315092' '383123'
 '112901' '392091' '12749' '350026' '315091' '2658' 'LP 1588' '368364'
 'PC 17760' 'AQ/3. 30631' 'PC 17569' '28004' '350408' '347075' '2654'
 '244368' '113790' '24160' 'SOTON/O.Q. 3101309' 'PC 17585' '2003' '23685
4'
 'PC 17580' '2684' '2653' '349229' '110469' '244360' '2675' '2622'
 'C.A. 15185' '350403' 'PC 17755' '348125' '237670' '2688' '248726'
 'F.C.C. 13528' 'PC 17759' 'F.C.C. 13540' '113044' '11769' '1222' '36840
2'
 '349910' 'S.C./PARIS 2079' '315083' '11765' '2689' '3101295' '112378'
 'SC/PARIS 2147' '28133' '112058' '248746' '315152' '29107' '680' '36671
3'
 '330910' '364498' '376566' 'SC/PARIS 2159' '349911' '244346' '364858'
 '349909' 'PC 17592' 'C.A. 2673' 'C.A. 30769' '371109' '13567' '347065'
 '21332' '28664' '113059' '17765' 'SC/PARIS 2166' '28666' '334915'
 '365237' '19928' '347086' 'A.5. 3236' 'PC 17758' 'SOTON/O.Q. 3101262'
 '359309' '2668']

 ------------------------------------------------------------

Name of Column: Cabin
[nan 'B45' 'E31' 'B57 B59 B63 B66' 'B36' 'A21' 'C78' 'D34' 'D19' 'A9'
 'D15' 'C31' 'C23 C25 C27' 'F G63' 'B61' 'C53' 'D43' 'C130' 'C132' 'C10
1'
 'C55 C57' 'B71' 'C46' 'C116' 'F' 'A29' 'G6' 'C6' 'C28' 'C51' 'E46' 'C5
4'
 'C97' 'D22' 'B10' 'F4' 'E45' 'E52' 'D30' 'B58 B60' 'E34' 'C62 C64' 'A1
1'
 'B11' 'C80' 'F33' 'C85' 'D37' 'C86' 'D21' 'C89' 'F E46' 'A34' 'D' 'B26'
 'C22 C26' 'B69' 'C32' 'B78' 'F E57' 'F2' 'A18' 'C106' 'B51 B53 B55'
 'D10 D12' 'E60' 'E50' 'E39 E41' 'B52 B54 B56' 'C39' 'B24' 'D28' 'B41'
 'C7' 'D40' 'D38' 'C105']

 ------------------------------------------------------------

Name of Column: Embarked
['Q' 'S' 'C']

 ------------------------------------------------------------
```

Explaining Datasets

```
survival : Survival 0 = No, 1 = Yes
pclass : Ticket class 1 = 1st, 2 = 2nd, 3 = 3rd
sex : Sex
Age : Age in years
sibsp : Number of siblings / spouses aboard the Titanic
parch # of parents / children aboard the Titanic
ticket : Ticket number fare Passenger fare cabin Cabin number
```

```
Data Analysis
```

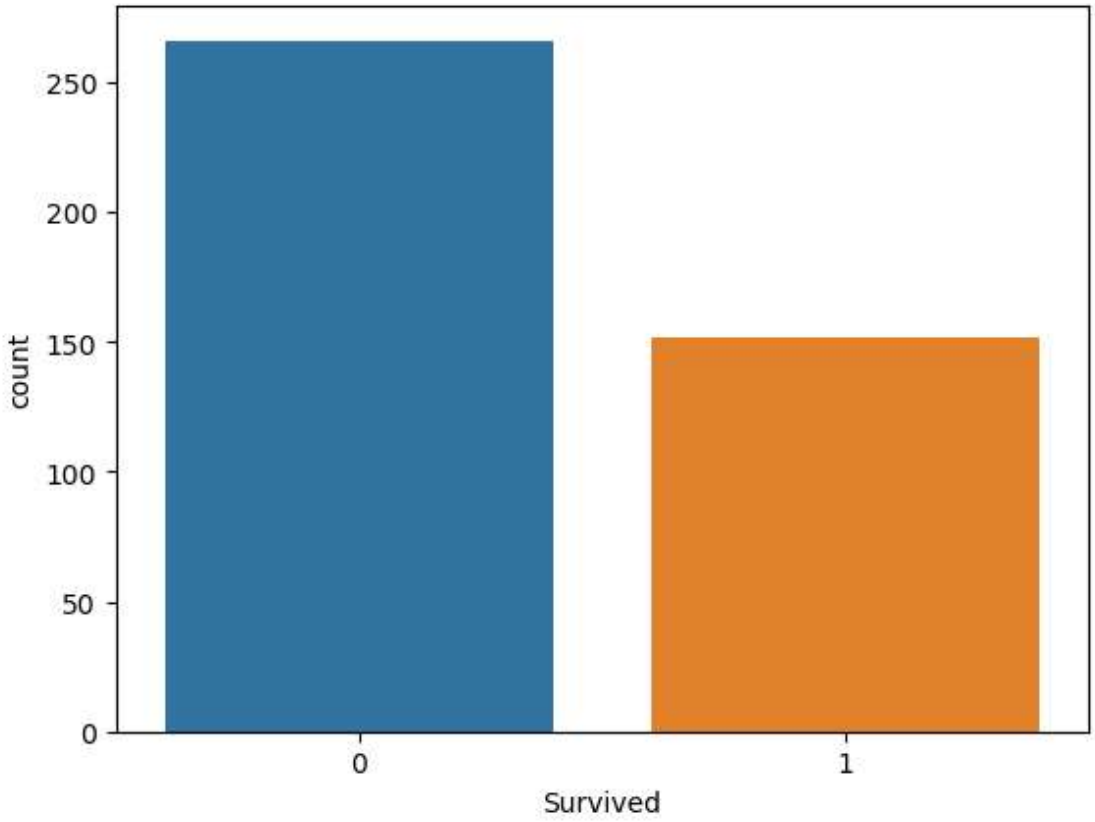
Find out how many survived vs Died using countplot method of seaborn

In [17]:
```
#countplot of survived vs not survived
```

In [18]:
```
# Create a countplot to visualize the distribution of 'Survived' values i

sns.countplot(x='Survived', data=titanic_data)
```

Out[18]: <Axes: xlabel='Survived', ylabel='count'>

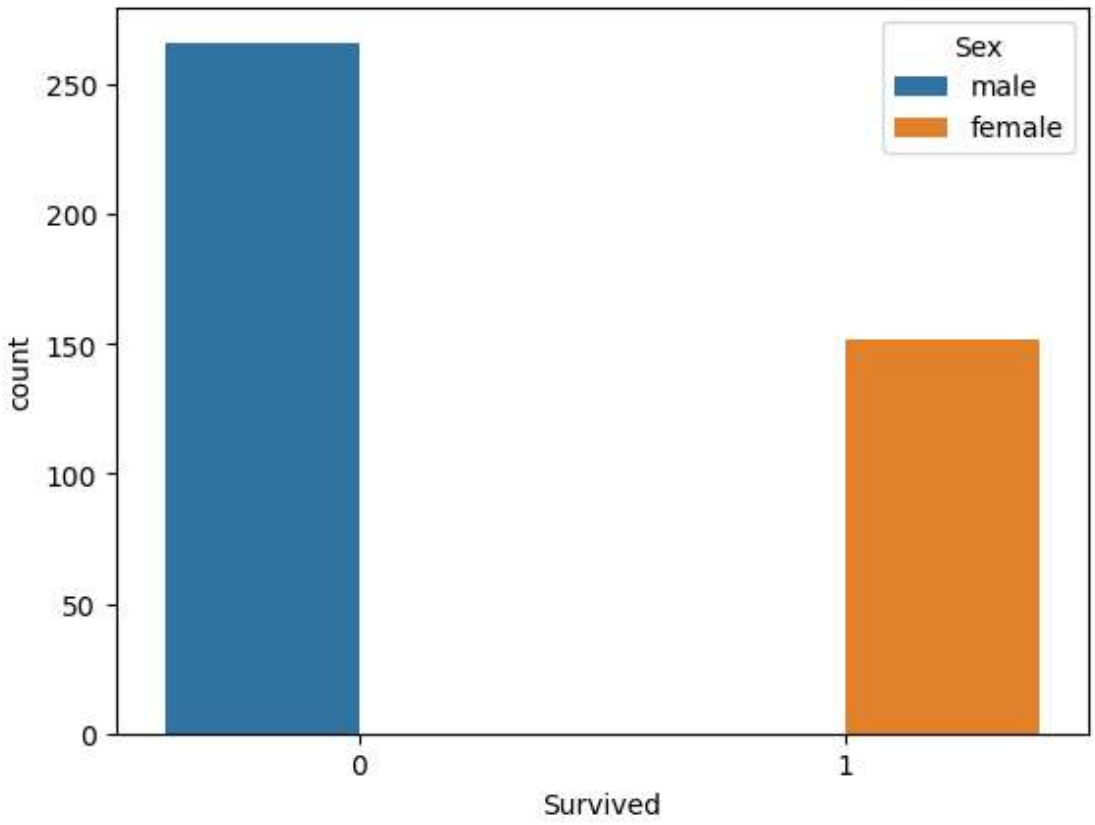


Male vs Female Survival

In [19]:
```
# Create a countplot to visualize the distribution of 'Survived' values ir

sns.countplot(x='Survived', data=titanic_data, hue='Sex')
```

Out[19]: <Axes: xlabel='Survived', ylabel='count'>



Check for NULL

In [20]:
```
# Check for missing values in the dataset and create a DataFrame of Boole

titanic_data.isna()
```

Out[20]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabi
0	False	False	False	False	False	False	False	False	False	False	Tru
1	False	False	False	False	False	False	False	False	False	False	Tru
2	False	False	False	False	False	False	False	False	False	False	Tru
3	False	False	False	False	False	False	False	False	False	False	Tru
4	False	False	False	False	False	False	False	False	False	False	Tru
...	...	...	...	...	...	...	...	...	...	...	
413	False	False	False	False	False	True	False	False	False	False	Tru
414	False	False	False	False	False	False	False	False	False	False	Fals
415	False	False	False	False	False	False	False	False	False	False	Tru
416	False	False	False	False	False	True	False	False	False	False	Tru
417	False	False	False	False	False	True	False	False	False	False	Tru

418 rows × 12 columns

Check how many values are NULL

In [21]:
```
# Calculate the number of missing values for each column in the 'titanic_

titanic_data.isna().sum()
```

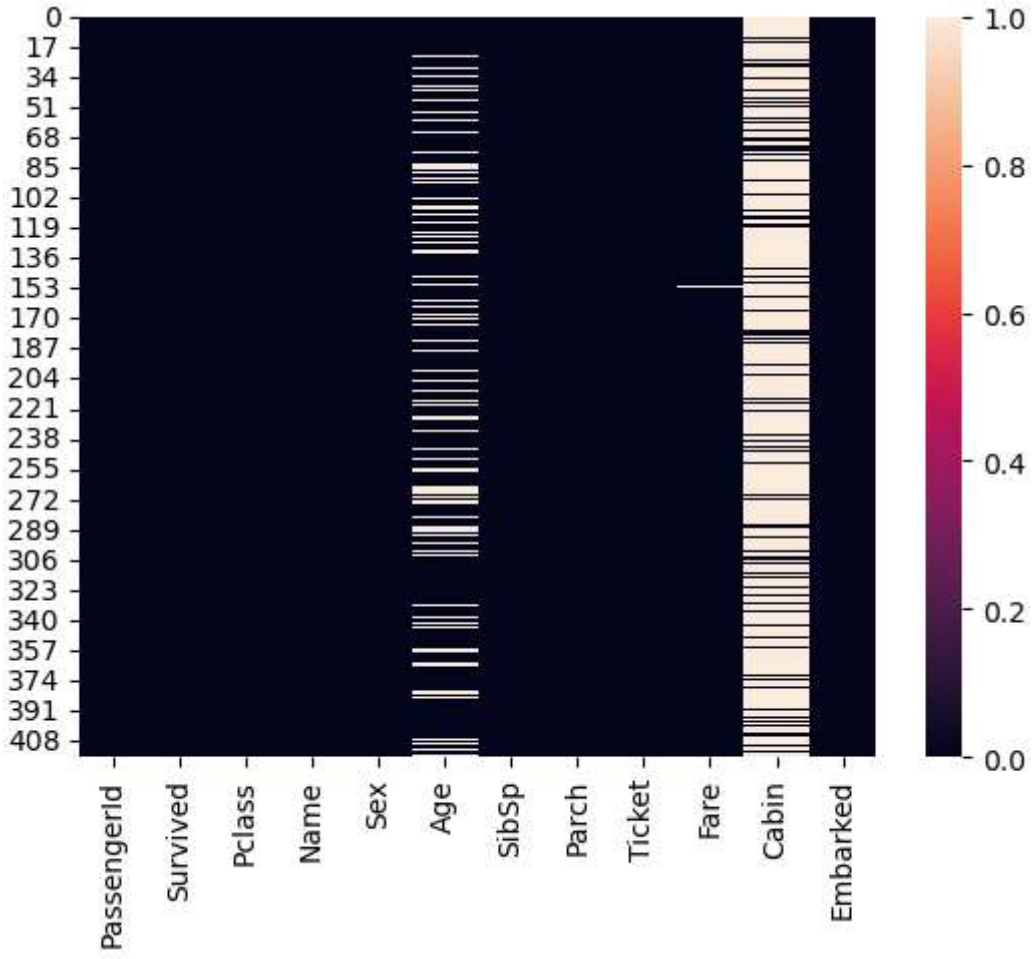
Out[21]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

Visualize NULL values with the help of HeatMap

In [22]:
```
# Create a heatmap to visualize the missing values in the 'titanic_data' c

sns.heatmap(titanic_data.isna())
```

Out[22]: <Axes: >



Find the percentage (%) of NULL values in age column

In [23]:
```
# Calculate the percentage of missing values in the 'Age' column of the 't

(titanic_data['Age'].isna().sum()/len(titanic_data['Age']))*100
```

Out[23]: 20.574162679425836

Find the percentage (%) of NULL values in cabin column

In [24]:
```
# Calculate the percentage of missing values in the 'Cabin' column of the

(titanic_data['Cabin'].isna().sum()/len(titanic_data['Cabin']))*100
```

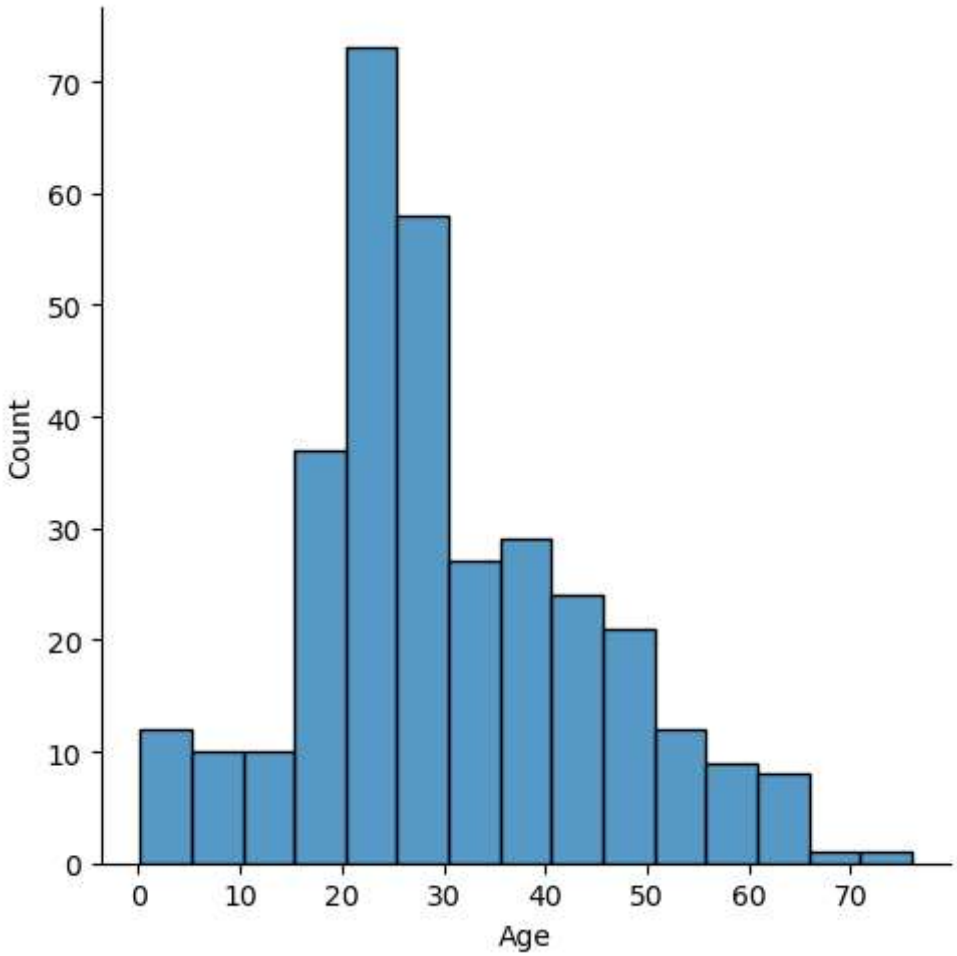
Out[24]: 78.22966507177034

Find the distribution for the age column

In [25]:
```
# Create a distribution plot (displot) to visualize the distribution of 'A

sns.displot(x='Age' , data = titanic_data)
```

Out[25]:
```
<seaborn.axisgrid.FacetGrid at 0x23c04f0f3d0>
```



Data Cleaning

Fill the missing values we will find the missing values for age. In order to fill missing values we use fillna method. For now we will fill the missing age by taking average of all age.

Fill Age Column

In [26]:
```
# Impute missing values in the 'Age' column with the mean value of the col

titanic_data['Age'].fillna(titanic_data['Age'].mean(), inplace=True)
```

Verify NuLL Values

In [27]:
```
# Check the number of remaining missing values in the 'Age' column.

titanic_data['Age'].isna().sum()
```

Out[27]: 0

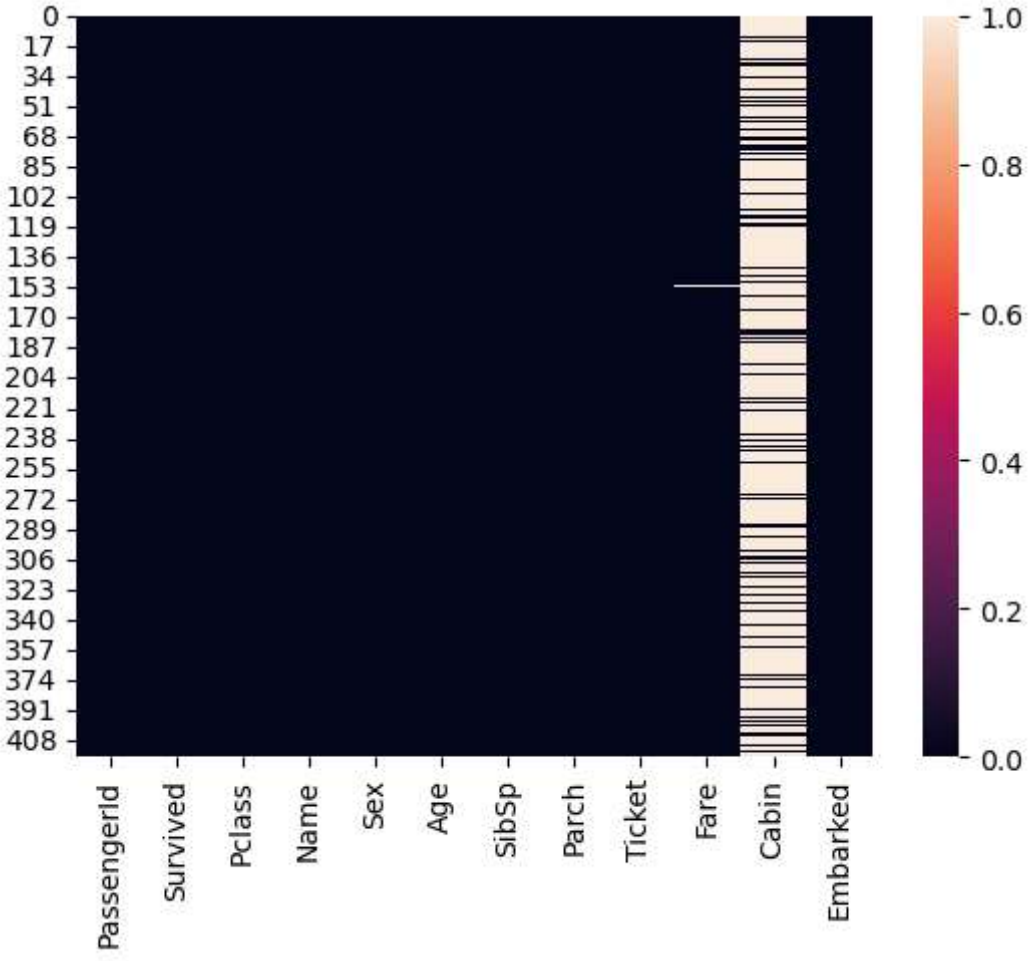
Alternatively we will visualise the null value using heatmap we will use heatmap method by passing only records which are null

In [28]:
```
#Visualize NuLL values again
```

In [29]:
```
# Create a heatmap to visualize the remaining missing values in the datase

sns.heatmap(titanic_data.isna())
```

Out[29]: <Axes: >



we can see the cabin column has a number of null values, as such we can not use it for prediction.

Hence, we will drop it.

In [30]:
```
# Drop the 'Cabin' column from the dataset.

titanic_data.drop('Cabin',axis=1,inplace=True)
```

In [31]:
```
# Display the first few rows of the modified 'titanic_data' dataset.

titanic_data.head()
```

Out[31]:

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

Preaparing Data for Model

```
No we will require to convert all non-numerical columns to numeric. Please
note this is required for feeding data
into model. Lets see which columns are non numeric info describe method
```

In [32]:
```
# Display information about the 'titanic_data' dataset, including column n

titanic_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 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   Ticket       418 non-null    object 
 9   Fare         417 non-null    float64
 10  Embarked     418 non-null    object 
dtypes: float64(2), int64(5), object(4)
memory usage: 36.0+ KB
```

In [33]:
```
# Get the data types of each column in the 'titanic_data' dataset.

titanic_data.dtypes
```

Out[33]:
```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Embarked        object
dtype: object
```

We can see, Name, Sex, Ticket and Embarked are non-numerical.It seems Name,Embarked and Ticket number are not useful for Machine Learning Prediction hence we will eventually drop it. For Now we would convert Sex Column to dummies numerical values

In [34]:
```
#Convert sex column to numeric columns
```

In [35]:
```
# Create dummy variables for the 'Sex' column and store them in the 'Gende

gender = pd.get_dummies(titanic_data['Sex'],drop_first=True)
```

In [36]:
```
# Create a new column 'Gender' and assign the dummy variable values for 'S

titanic_data['Gender']= gender
```

In [37]:
```
# Display the first few rows of the modified 'titanic_data' dataset.

titanic_data.head()

# Note: In the 'Gender' column, 'male' is encoded as 1, and 'female' is en
```

Out[37]:

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

Drop the column which are not require

In [38]:
```
# Drop the specified columns ('Name', 'Sex', 'Ticket', 'Embarked') from th

titanic_data.drop(['Name','Sex','Ticket','Embarked'], axis=1, inplace=Tru
```

In [39]:
```
# Display the first few rows of the modified 'titanic_data' DataFrame.

titanic_data.head()
```

Out[39]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	Gender
0	892	0	3	34.5	0	0	7.8292	1
1	893	1	3	47.0	1	0	7.0000	0
2	894	0	2	62.0	0	0	9.6875	1
3	895	0	3	27.0	0	0	8.6625	1
4	896	1	3	22.0	1	1	12.2875	0

In [40]:
```
# Check for missing values in the dataset and calculate the number of miss

titanic_data.isna().sum()
```

Out[40]:
```
PassengerId    0
Survived       0
Pclass         0
Age            0
SibSp          0
Parch          0
Fare           1
Gender         0
dtype: int64
```

In [41]:
```
# Impute missing values in the 'Fare' column with the mean value of the co

titanic_data['Fare'].fillna(titanic_data['Fare'].mean(), inplace=True)
```

In [42]:
```
#Seperate Dependent and Independent variables
```

In [43]:
```
# Select the features ('x') and the target ('y') for modeling.

x=titanic_data[['PassengerId','Pclass','Age','SibSp','Parch','Fare','Gende
y=titanic_data['Survived']
```

In [44]:
```
# Displaying the dependent varialbles

x
```

Out[44]:

	PassengerId	Pclass	Age	SibSp	Parch	Fare	Gender
0	892	3	34.50000	0	0	7.8292	1
1	893	3	47.00000	1	0	7.0000	0
2	894	2	62.00000	0	0	9.6875	1
3	895	3	27.00000	0	0	8.6625	1
4	896	3	22.00000	1	1	12.2875	0
...	...	...	...	...	...	...	...
413	1305	3	30.27259	0	0	8.0500	1
414	1306	1	39.00000	0	0	108.9000	0
415	1307	3	38.50000	0	0	7.2500	1
416	1308	3	30.27259	0	0	8.0500	1
417	1309	3	30.27259	1	1	22.3583	1

418 rows × 7 columns

In [45]:
```
# Displaying the independent varialbles

y
```

Out[45]:
```
0      0
1      1
2      0
3      0
4      1
      ..
413    0
414    1
415    0
416    0
417    0
Name: Survived, Length: 418, dtype: int64
```

Data Modelling

```
Building Model using Logistic Regression

Build the model
```

In [46]:
```
# Import the train_test_split function to split the dataset into training
```

In [47]:
```
from sklearn.model_selection import train_test_split
```

In [48]:
```
#train test split
```

In [49]:
```
# Split the feature set 'x' and the target 'y' into training and testing s
# Here, 33% of the data is used for testing, and a random seed (random_sta

X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.33,
```

In [50]:
```
# Import the LogisticRegression model from scikit-learn.
```

In [51]:
```
from sklearn.linear_model import LogisticRegression
```

In [52]:
```
# Create a logistic regression model.

lr = LogisticRegression()
```

In [57]:
```
# Fit the logistic regression model to the training data.

lr.fit(X_train,y_train)
# # Ignoring errors
# import warnings
# warnings.filterwarnings('ignore')
```

Out[57]:
```
LogisticRegression

LogisticRegression()
```

In [58]:
```
# Prediction
```

In [59]:
```
# Use the trained model to make predictions on the test data.

predict=lr.predict(X_test)
```

Testing

```
See how our model is performing
```

In [60]:
```
# Import the confusion_matrix function from scikit-learn.
```

In [61]:
```
from sklearn.metrics import confusion_matrix
```

In [62]:
```
# Calculate the confusion matrix for the test data and format it into a Da
# The rows and columns are labeled to indicate actual and predicted values

pd.DataFrame(confusion_matrix(y_test,predict),columns=['Predicted No','Pre
 index=['Actual No','Actual Yes'])
```

Out[62]:

	Predicted No	Predicted Yes
Actual No	92	0
Actual Yes	0	46

In [63]:
```
# Import the classification_report function from scikit-learn.
```

In [64]:
```
from sklearn.metrics import classification_report
```

In [65]:
```
# Generate a classification report for the model's performance on the test
# This report includes metrics like precision, recall, F1-score, and suppo

report = classification_report(y_test, predict)
```

In [66]:
```
# Print the classification report to the console.

print(report)
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        92
           1       1.00      1.00      1.00        46

    accuracy                           1.00       138
   macro avg       1.00      1.00      1.00       138
weighted avg       1.00      1.00      1.00       138
```

Precision is fine considering Model Selected and Available Data. Accuracy can be increased by further using more features (which we dropped earlier) and/or by using other model

```
Note:
Precision : Precision is the ratio of correctly predicted positive
observations to the total predicted positive
observations Recall : Recall is the ratio of correctly predicted positive
observations to the all observations in
```

```
actual class F1 score - F1 Score is the weighted average of Precision and
Recall.
```

In [ ]: