Expploratory Data Analysis

This will shows us how wee can do EDA using python Three important steps to keep in mind are:
1- Understand data 2- Clean the data 3- Find a relationship between data

In [ ]:
```
# import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
kashti =sns.load_dataset('titanic')
```

In [ ]:
```
kashti.to_csv('kashti.csv')
```

In [ ]:
```
kashti.info()
```

In [ ]:
```
ks = kashti
```

In [ ]:
```
ks.head
```

In [ ]:
```
ks.tail
```

In [ ]:
```
# rows and columns
ks.shape
```

In [ ]:
```
ks.describe()
```

In [ ]:
```
# unique values
ks.nunique()
```

In [24]:
```
# column name
ks.columns
```

Out[24]:
```
Index(['survived', 'pclass', 'sex', 'age', 'sibsp', 'parch', 'fare',
       'embarked', 'class', 'who', 'adult_male', 'deck', 'embark_town',
       'alive', 'alone'],
      dtype='object')
```

In [ ]:
```
ks['sex'].unique()
```

In [ ]:
```
print(' : ')
print(ks[' : '].unique())
print('\n')
```

In [27]:
```
# Unique values for all the coloumns
column_values = ks[["survived", "pclass", "sex", "age", "sibsp", "parch", "fare",
"embarked", "class", "who", "adult_male", "deck", "embark_town",
"alive", "alone"]].values
unique_values =  np.unique(column_values)
print(unique_values)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
C:\Users\MUYYAS~1\AppData\Local\Temp/ipykernel_912/3873173251.py in <module>
      2 "embarked", "class", "who", "adult_male", "deck", "embark_town",
      3 "alive", "alone"]].values
----> 4 unique_values =  np.unique(column_values)
      5 print(unique_values)

~\AppData\Local\Programs\Python\Python310\lib\site-packages\numpy\core\override
s.py in unique(*args, **kwargs)

~\AppData\Local\Programs\Python\Python310\lib\site-packages\numpy\lib\arrayseto
ps.py in unique(ar, return_index, return_inverse, return_counts, axis)
    270     ar = np.asanyarray(ar)
    271     if axis is None:
--> 272         ret = _unique1d(ar, return_index, return_inverse, return_counts
)
    273         return _unpack_tuple(ret)
    274

~\AppData\Local\Programs\Python\Python310\lib\site-packages\numpy\lib\arrayseto
ps.py in _unique1d(ar, return_index, return_inverse, return_counts)
    331         aux = ar[perm]
    332     else:
--> 333         ar.sort()
    334         aux = ar
    335     mask = np.empty(aux.shape, dtype=np.bool_)

TypeError: '<' not supported between instances of 'str' and 'int'
```

Cleaning and filtering the data

In [29]:
```
# find missing value inside
ks.isnull().sum()
```

Out[29]:
```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

In [32]:
```
# removing missing data
ks_clean = ks.drop(['deck'], axis=1)
ks_clean.head()
```

Out[32]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	em
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	S
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	S
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	S
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	S

In [33]:
```
ks_clean.isnull().sum()
```

Out[33]:
```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
embark_town      2
alive            0
alone            0
dtype: int64
```

In [34]:
```
ks_clean.dropna().shape
```

Out[34]:
```
(712, 14)
```

In [35]:
```
ks_clean.isnull().sum()
```

Out[35]:
```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
embark_town      2
alive            0
alone            0
dtype: int64
```

In [36]:
```
ks_clean = ks_clean.dropna()
```

In [37]:
```
ks_clean.isnull().sum()
```

Out[37]:
```
survived       0
pclass         0
sex            0
age            0
sibsp          0
parch          0
fare           0
embarked       0
class          0
who            0
adult_male     0
embark_town    0
alive          0
alone          0
dtype: int64
```

In [38]:
```
ks_clean.shape
```

Out[38]:
```
(712, 14)
```

In [39]:
```
ks.shape
```

Out[39]:
```
(891, 15)
```

In [41]:
```
ks_clean["sex"].value_counts()
```

Out[41]:
```
male      453
female    259
Name: sex, dtype: int64
```

In [42]:
```
ks.describe()
```

Out[42]:

	survived	pclass	age	sibsp	parch	fare
count	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [43]:
```
ks_clean.describe()
```

Out[43]:

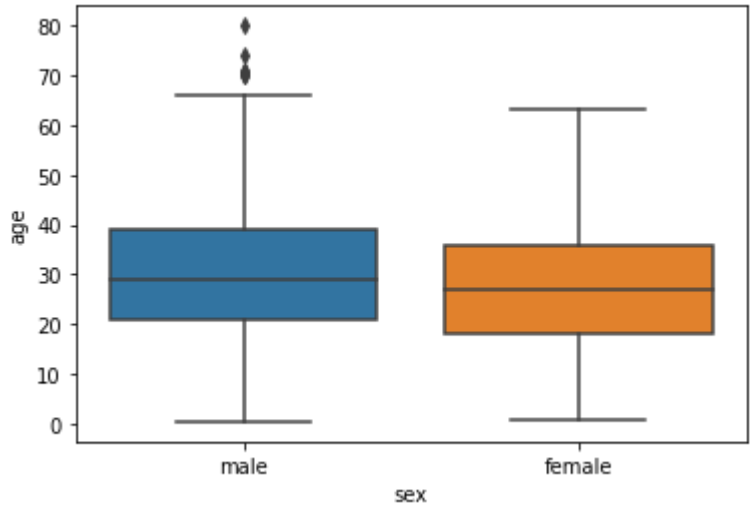
	survived	pclass	age	sibsp	parch	fare
count	712.000000	712.000000	712.000000	712.000000	712.000000	712.000000
mean	0.404494	2.240169	29.642093	0.514045	0.432584	34.567251
std	0.491139	0.836854	14.492933	0.930692	0.854181	52.938648
min	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000	1.000000	20.000000	0.000000	0.000000	8.050000
50%	0.000000	2.000000	28.000000	0.000000	0.000000	15.645850
75%	1.000000	3.000000	38.000000	1.000000	1.000000	33.000000
max	1.000000	3.000000	80.000000	5.000000	6.000000	512.329200

In [44]:
```
#sns.boxplot()
ks_clean.columns
```

Out[44]:
```
Index(['survived', 'pclass', 'sex', 'age', 'sibsp', 'parch', 'fare',
       'embarked', 'class', 'who', 'adult_male', 'embark_town', 'alive',
       'alone'],
      dtype='object')
```

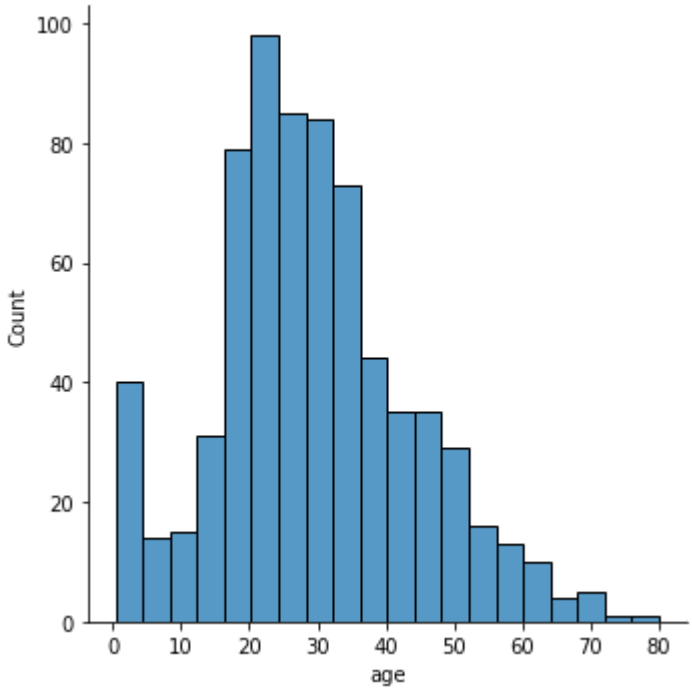
In [45]:
```
sns.boxplot(x= 'sex', y= 'age', data=ks_clean)
```

Out[45]:
```
<AxesSubplot:xlabel='sex', ylabel='age'>
```



In [46]:
```
sns.displot(data=ks_clean['age'])
```

Out[46]:
```
<seaborn.axisgrid.FacetGrid at 0x2e1a738cfd0>
```



In [47]:
```
# out liers removal
ks_clean['age'].mean()
```

Out[47]:
```
29.64209269662921
```

In [48]:
```
ks_clean['age'] < 68
```

Out[48]:
```
0      True
1      True
2      True
3      True
4      True
       ...
885    True
886    True
887    True
889    True
890    True
Name: age, Length: 712, dtype: bool
```

In [49]:
```
ks_clean = ks_clean[ks_clean['age']<68]
ks_clean.head()
```

Out[49]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	em
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	S
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	S
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	S
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	S

In [50]:
```
ks_clean.shape
```

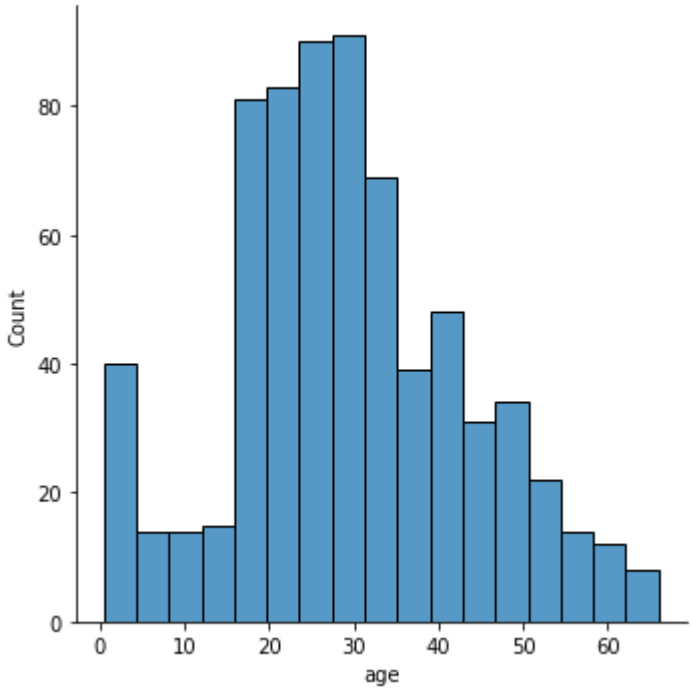
Out[50]:
```
(705, 14)
```

In [51]:
```
ks_clean['age'].mean()
```

Out[51]:
```
29.21797163120567
```

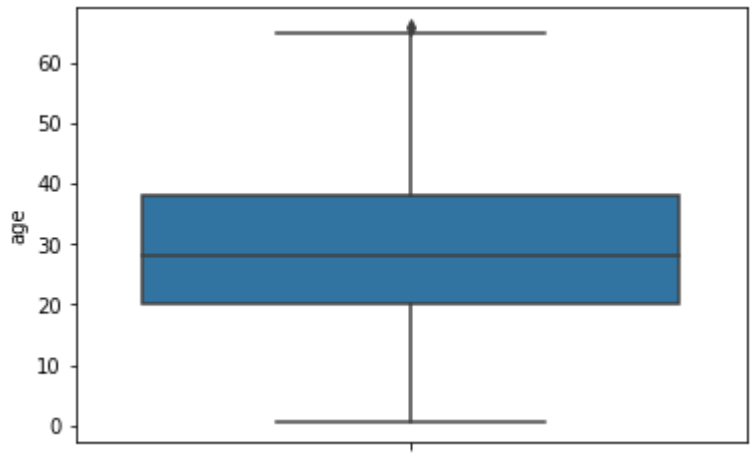
In [52]:
```
sns.displot(ks_clean['age'])
```

Out[52]:
```
<seaborn.axisgrid.FacetGrid at 0x2e1a738cf10>
```



In [53]:
```
sns.boxplot(y='age', data=ks_clean)
```

Out[53]:
```
<AxesSubplot:ylabel='age'>
```



In [54]:
```
ks_clean.head
```

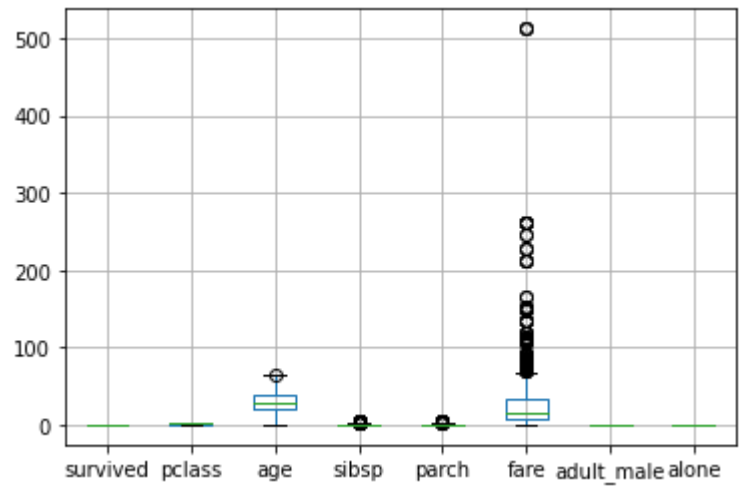
Out[54]:
```
<bound method NDFrame.head of      survived  pclass     sex   age  sibsp  parch
fare embarked   class  \
0           0       3    male  22.0      1      0   7.2500        S   Third
1           1       1  female  38.0      1      0  71.2833        C   First
2           1       3  female  26.0      0      0   7.9250        S   Third
3           1       1  female  35.0      1      0  53.1000        S   First
4           0       3    male  35.0      0      0   8.0500        S   Third
..        ...     ...     ...   ...    ...    ...      ...      ...     ...
885         0       3  female  39.0      0      5  29.1250        Q   Third
886         0       2    male  27.0      0      0  13.0000        S  Second
887         1       1  female  19.0      0      0  30.0000        S   First
889         1       1    male  26.0      0      0  30.0000        C   First
890         0       3    male  32.0      0      0   7.7500        Q   Third

       who  adult_male  embark_town alive  alone
0      man        True  Southampton    no  False
1    woman       False    Cherbourg   yes  False
2    woman       False  Southampton   yes   True
3    woman       False  Southampton   yes  False
4      man        True  Southampton    no   True
..     ...         ...          ...   ...    ...
885  woman       False   Queenstown    no  False
886    man        True  Southampton    no   True
887  woman       False  Southampton   yes   True
889    man        True    Cherbourg   yes   True
890    man        True   Queenstown    no   True

[705 rows x 14 columns]>
```

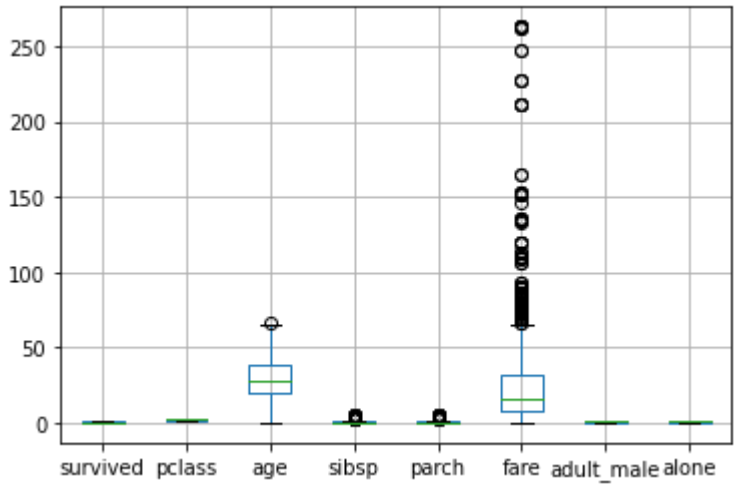
In [56]:
```
ks_clean.boxplot()
```

Out[56]:
```
<AxesSubplot:>
```



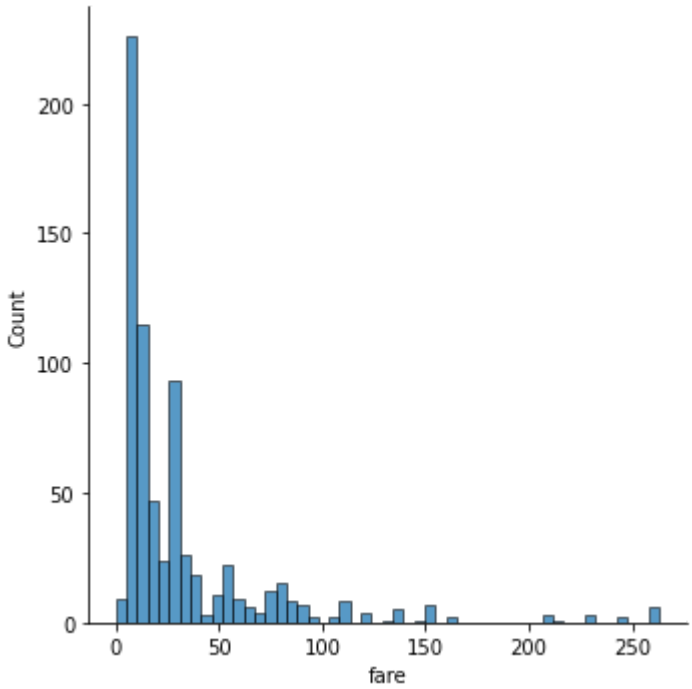
In [57]:
```
ks_clean = ks_clean[ks_clean['fare']<300]
ks_clean.boxplot()
```

Out[57]: <AxesSubplot:>



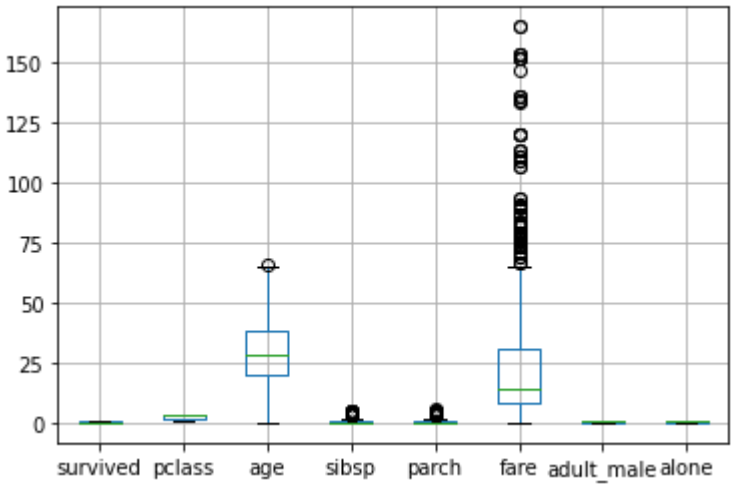
In [58]:
```
sns.displot(ks_clean['fare'])
```

Out[58]: <seaborn.axisgrid.FacetGrid at 0x2e1a97cc4c0>



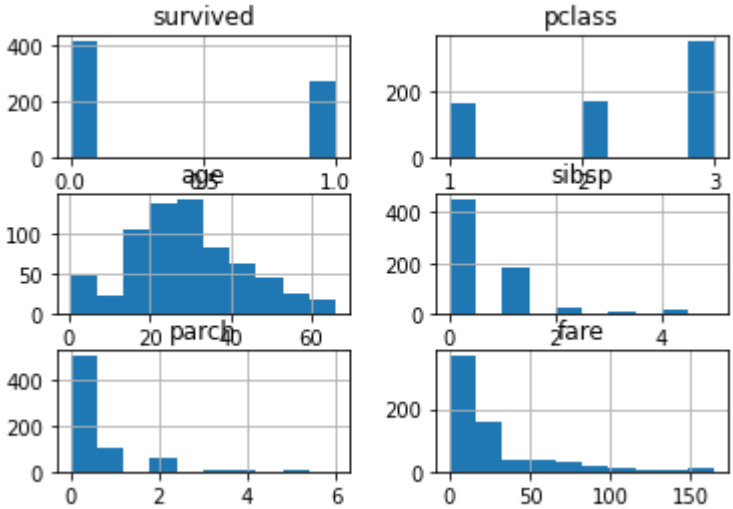
In [59]:
```
ks_clean = ks_clean[ks_clean['fare']<200]
ks_clean.boxplot()
```

Out[59]:
```
<AxesSubplot:>
```



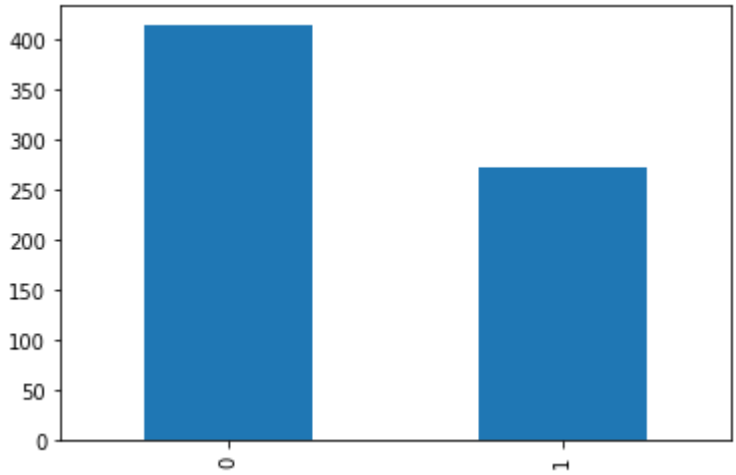
In [60]:
```
ks_clean.hist()
```

Out[60]:
```
array([[<AxesSubplot:title={'center':'survived'}>,
        <AxesSubplot:title={'center':'pclass'}>],
       [<AxesSubplot:title={'center':'age'}>,
        <AxesSubplot:title={'center':'sibsp'}>],
       [<AxesSubplot:title={'center':'parch'}>,
        <AxesSubplot:title={'center':'fare'}>]], dtype=object)
```



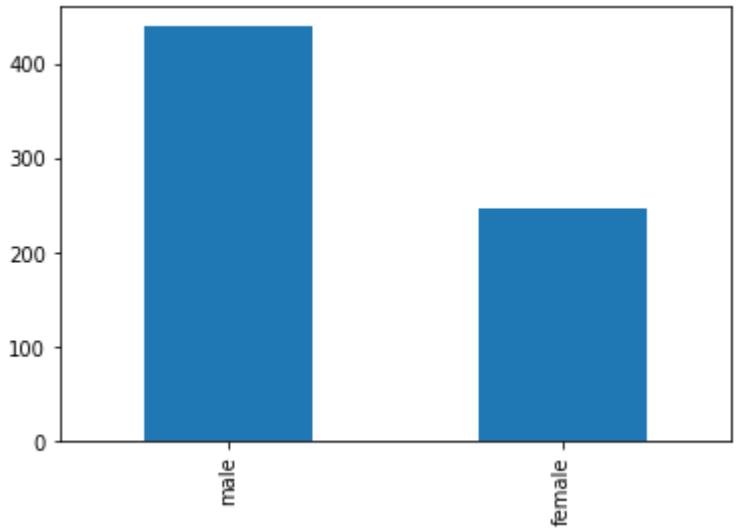
In [61]:
```
pd.value_counts(ks_clean['survived']).plot.bar()
```

Out[61]: <AxesSubplot:>



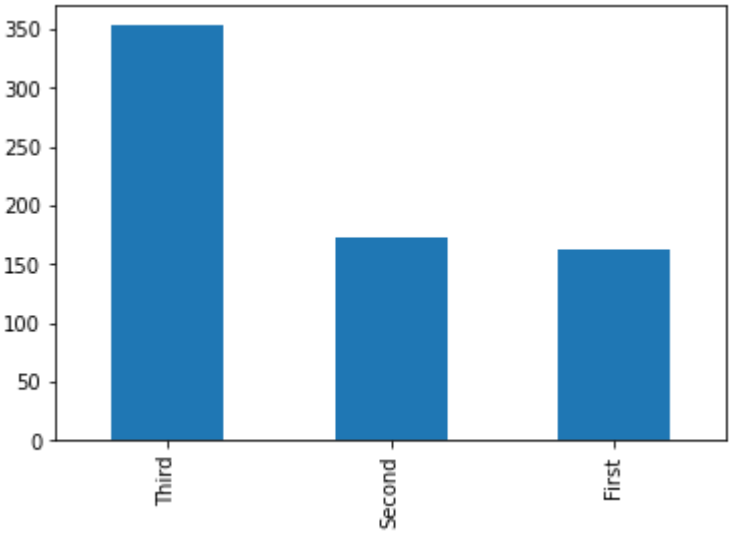
In [62]:
```
pd.value_counts(ks_clean['sex']).plot.bar()
```

Out[62]: <AxesSubplot:>



In [64]:
```
pd.value_counts(ks_clean['class']).plot.bar()
```

Out[64]: <AxesSubplot:>



In [65]:
```
ks_clean.groupby(['sex']).mean()
```

Out[65]:

sex	survived	pclass	age	sibsp	parch	fare	adult_male	alone
female	0.740891	2.125506	27.651822	0.631579	0.704453	36.971306	0.000000	0.380567
male	0.204545	2.363636	30.017432	0.440909	0.254545	23.027091	0.909091	0.675000

In [68]:
```
ks_clean.groupby(['sex', 'class']).mean()
```

Out[68]:

sex	class	survived	pclass	age	sibsp	parch	fare	adult_male	alone
female	First	0.957746	1.0	35.014085	0.492958	0.436620	82.933041	0.000000	0.366197
	Second	0.918919	2.0	28.722973	0.500000	0.621622	21.951070	0.000000	0.405405
	Third	0.460784	3.0	21.750000	0.823529	0.950980	15.875369	0.000000	0.372549
male	First	0.406593	1.0	40.356264	0.362637	0.252747	54.841575	0.967033	0.549451
	Second	0.153061	2.0	30.340102	0.377551	0.244898	21.221429	0.908163	0.632653
	Third	0.151394	3.0	26.143108	0.494024	0.258964	12.197757	0.888446	0.737052

In [69]:
```
ks_clean.groupby(['sex', 'class','who']).mean()
```

Out[69]:

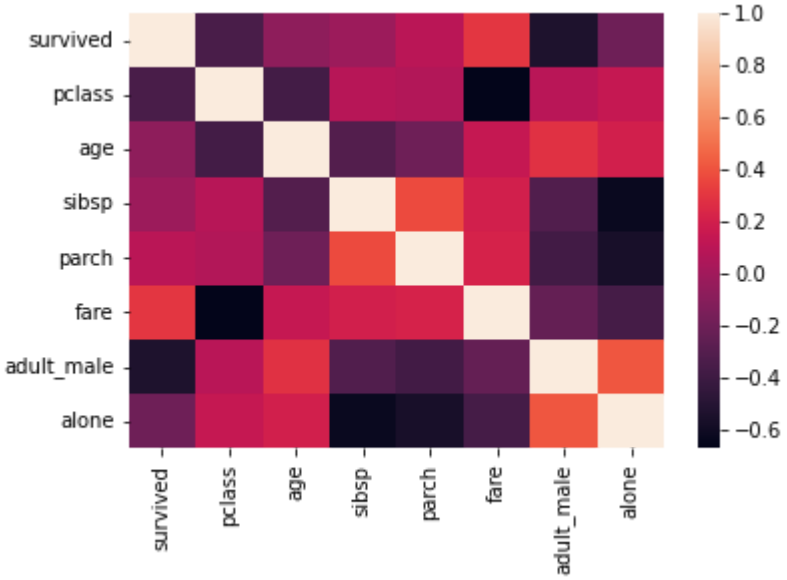
sex	class	who	survived	pclass	age	sibsp	parch	fare	adult_male
		child	0.500000	1.0	8.000000	1.000000	2.000000	135.775000	0.0
	First	man	NaN	NaN	NaN	NaN	NaN	NaN	NaN
		woman	0.971014	1.0	35.797101	0.478261	0.391304	81.401390	0.0
		child	1.000000	2.0	6.600000	0.700000	1.300000	29.240000	0.0
female	Second	man	NaN	NaN	NaN	NaN	NaN	NaN	NaN
		woman	0.906250	2.0	32.179688	0.468750	0.515625	20.812175	0.0
		child	0.533333	3.0	7.100000	1.533333	1.100000	19.023753	0.0
	Third	man	NaN	NaN	NaN	NaN	NaN	NaN	NaN
		woman	0.430556	3.0	27.854167	0.527778	0.888889	14.563542	0.0
		child	1.000000	1.0	5.306667	0.666667	2.000000	117.802767	0.0
	First	man	0.386364	1.0	41.551136	0.352273	0.193182	52.695170	1.0
		woman	NaN	NaN	NaN	NaN	NaN	NaN	NaN
		child	1.000000	2.0	2.258889	0.888889	1.222222	27.306022	0.0
male	Second	man	0.067416	2.0	33.179775	0.325843	0.146067	20.606133	1.0
		woman	NaN	NaN	NaN	NaN	NaN	NaN	NaN
		child	0.321429	3.0	6.515000	2.821429	1.321429	27.716371	0.0
	Third	man	0.130045	3.0	28.607623	0.201794	0.125561	10.249231	1.0
		woman	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Relationship

In [71]:
```
cor_ks_clean = ks_clean.corr()
```

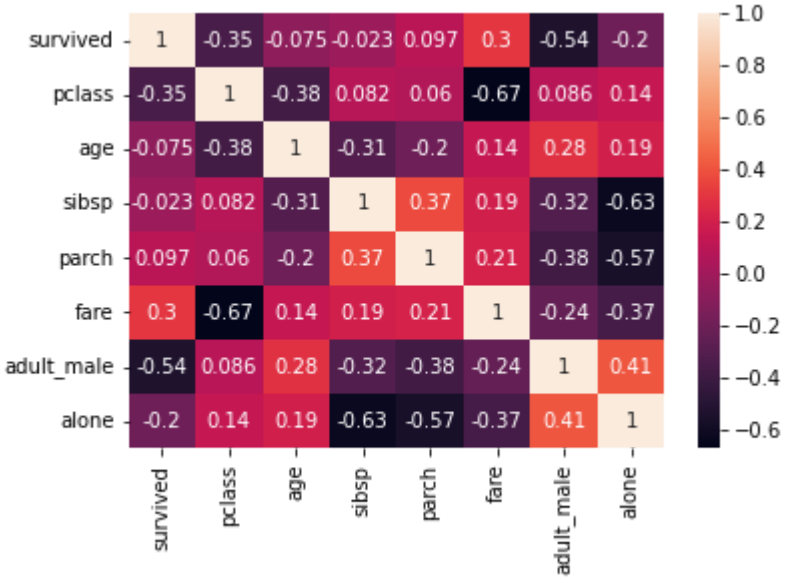
In [72]:
```
sns.heatmap(cor_ks_clean)
```

Out[72]: <AxesSubplot:>



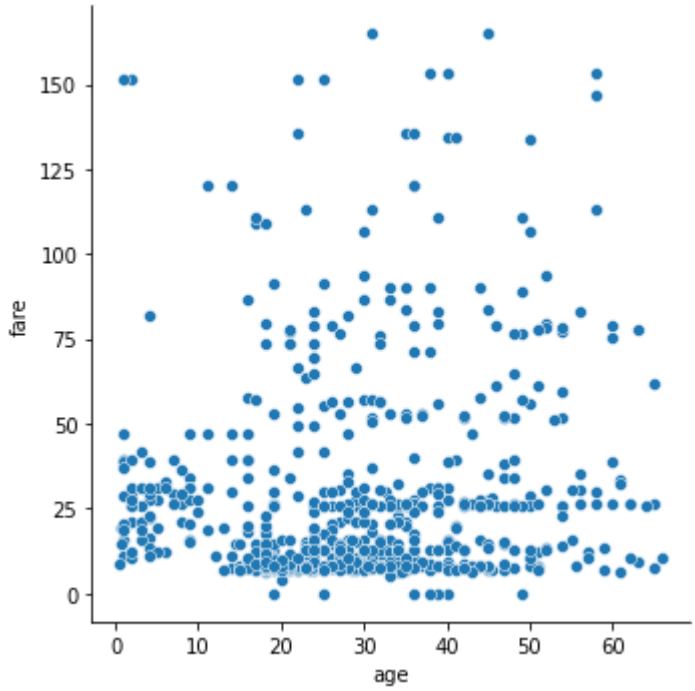
In [75]:
```
sns.heatmap(cor_ks_clean, annot=True)
```

Out[75]: <AxesSubplot:>



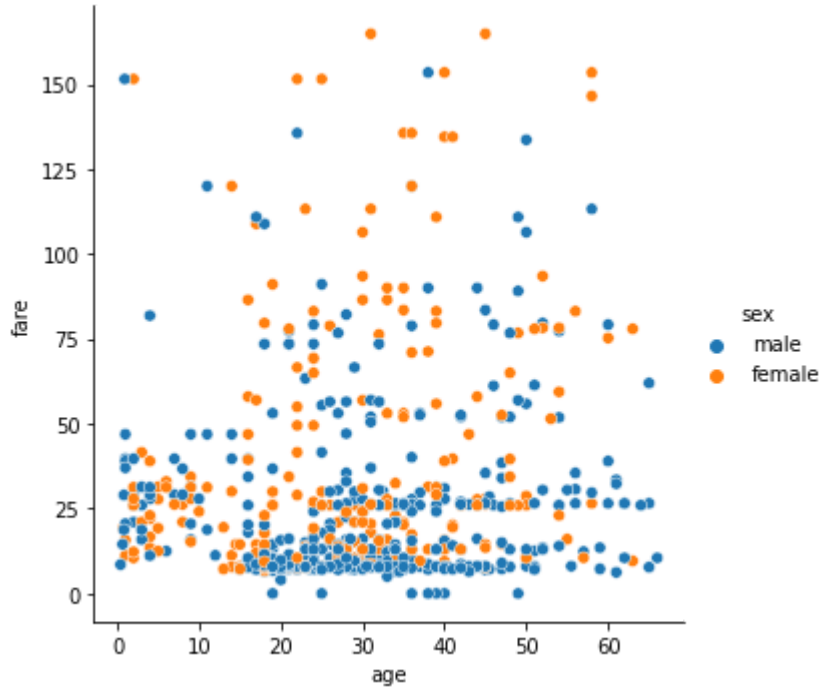
In [76]:
```
sns.relplot(x='age', y='fare', data=ks_clean)
```

Out[76]:
```
<seaborn.axisgrid.FacetGrid at 0x2e1aad044c0>
```



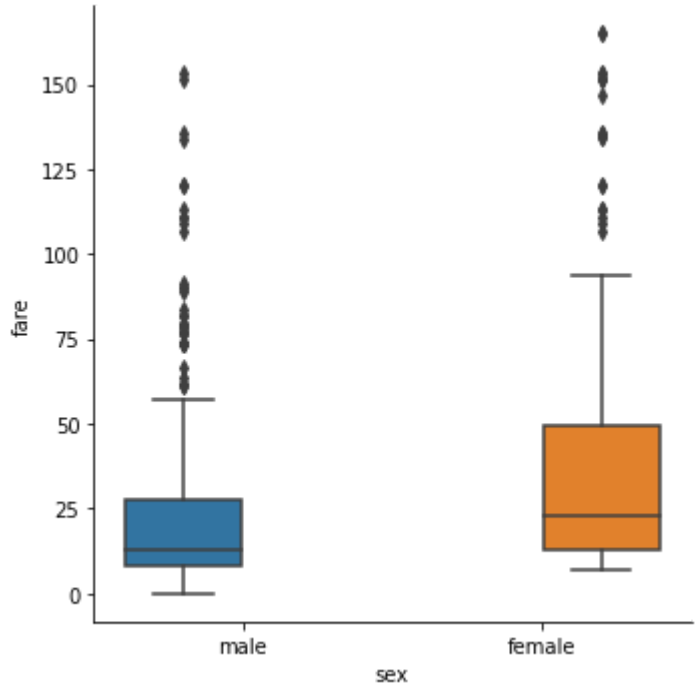
In [78]:
```
sns.relplot(x='age', y='fare', hue='sex', data=ks_clean)
```

Out[78]:
```
<seaborn.axisgrid.FacetGrid at 0x2e1a970c670>
```



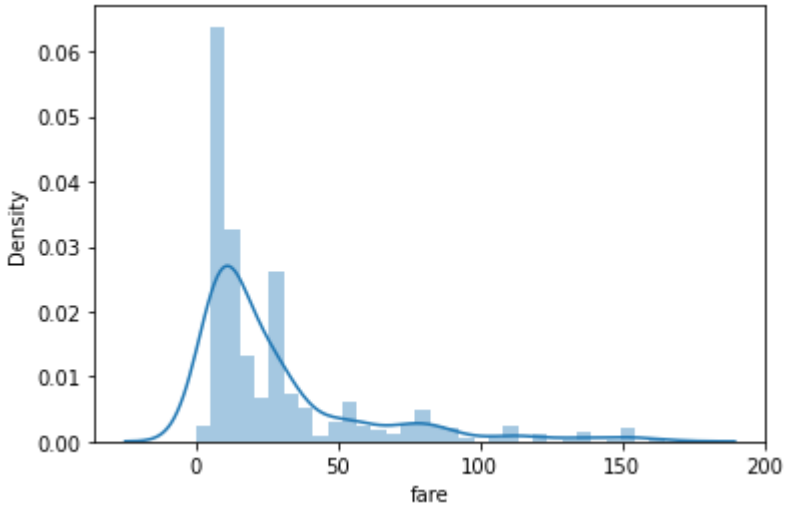
In [81]:
```
sns.catplot(x='sex', y='fare', hue='sex', data=ks_clean, kind='box')
```

Out[81]:
```
<seaborn.axisgrid.FacetGrid at 0x2e1aac913f0>
```



In [83]:
```
sns.distplot(ks_clean['fare'])
ks_clean['fare_log'] =np.log(ks_clean['fare'])
```

```
C:\Users\muyyassarhussain\AppData\Local\Programs\Python\Python310\lib\site-pack
ages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated f
unction and will be removed in a future version. Please adapt your code to use
either `displot` (a figure-level function with similar flexibility) or `histplo
t` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\muyyassarhussain\AppData\Local\Programs\Python\Python310\lib\site-pack
ages\pandas\core\arraylike.py:364: RuntimeWarning: divide by zero encountered i
n log
  result = getattr(ufunc, method)(*inputs, **kwargs)
```



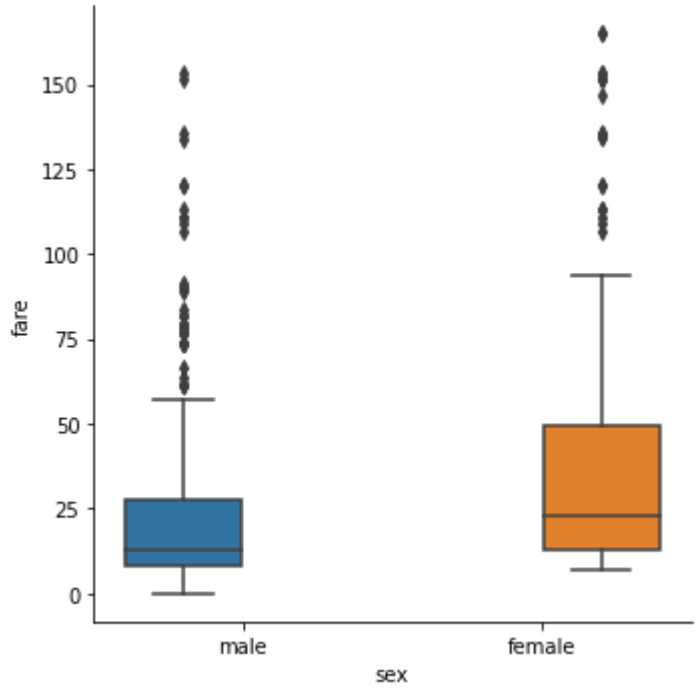
In [84]:
```
ks_clean.head()
```

Out[84]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	em
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	S
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	S
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	S
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	S

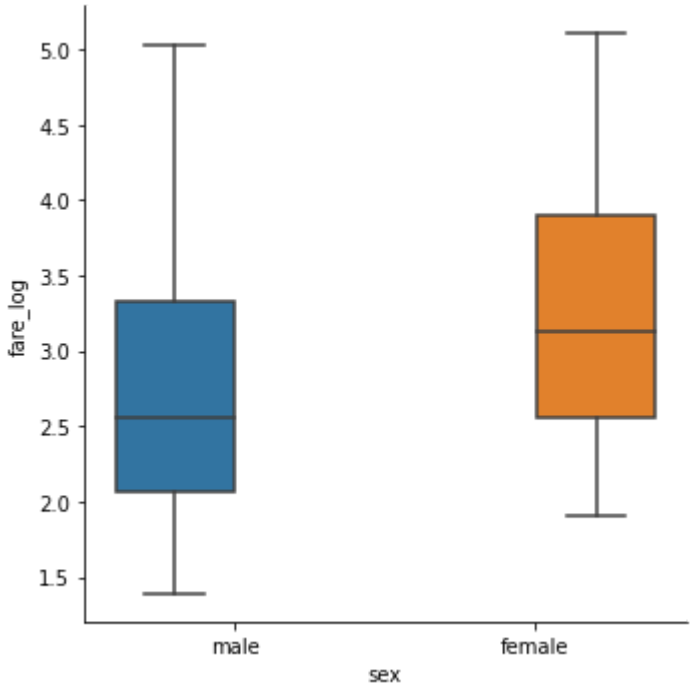
In [85]:
```
sns.catplot(x='sex', y='fare', hue='sex', data=ks_clean, kind='box')
```

Out[85]:
```
<seaborn.axisgrid.FacetGrid at 0x2e1ab324f40>
```



In [86]:
```
sns.catplot(x='sex', y='fare_log', hue='sex', data=ks_clean, kind='box')
```

Out[86]:
```
<seaborn.axisgrid.FacetGrid at 0x2e1ad445150>
```



In [ ]:
```

```