In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

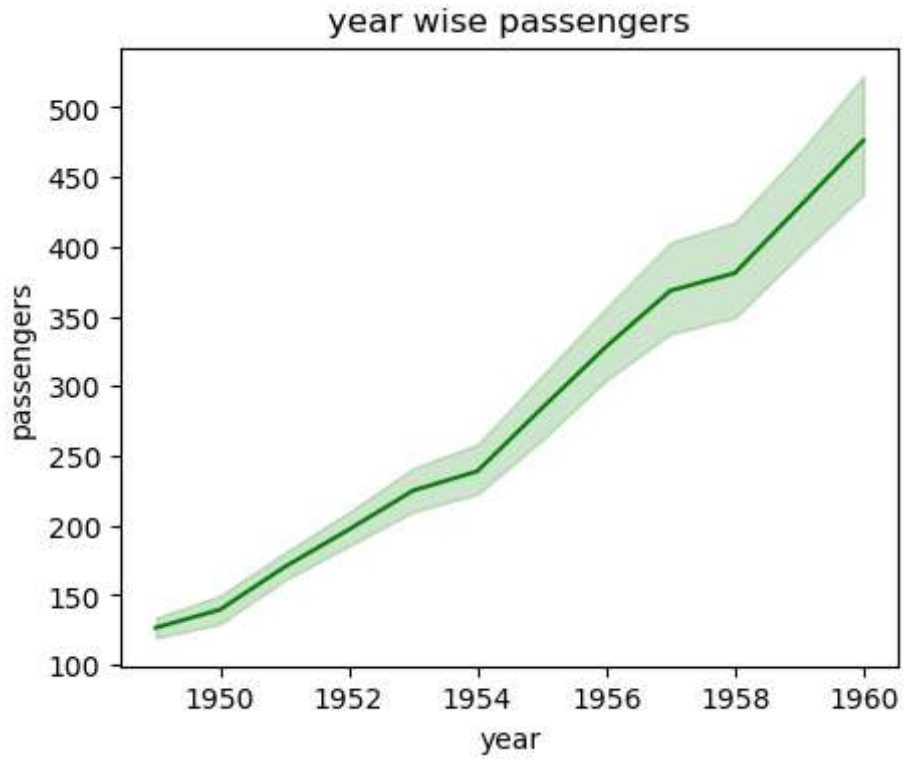
In [2]:
```
df=pd.read_csv("flights.csv")
df
```

Out[2]:

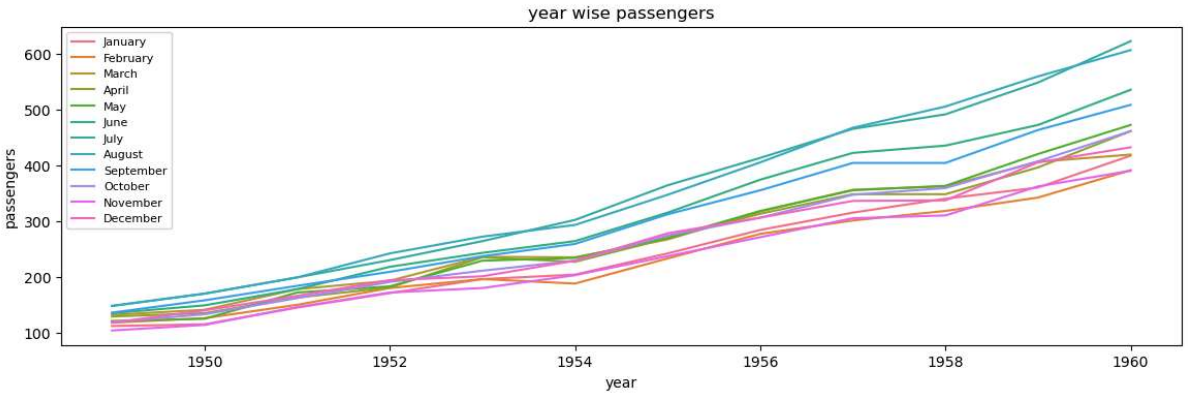
	year	month	passengers
0	1949	January	112
1	1949	February	118
2	1949	March	132
3	1949	April	129
4	1949	May	121
...	...	...	...
139	1960	August	606
140	1960	September	508
141	1960	October	461
142	1960	November	390
143	1960	December	432

144 rows × 3 columns

In [7]:
```
# line plot
plt.figure(figsize=(5,4))
sns.lineplot(x="year",y="passengers",data=df,color="g")
plt.title("year wise passengers")
plt.show()
```



In [13]:
```
plt.figure(figsize=(14,4))
sns.lineplot(x="year",y="passengers",data=df,color="g" ,hue="month")
plt.title("year wise passengers")
plt.legend(fontsize=8)
plt.show()
```

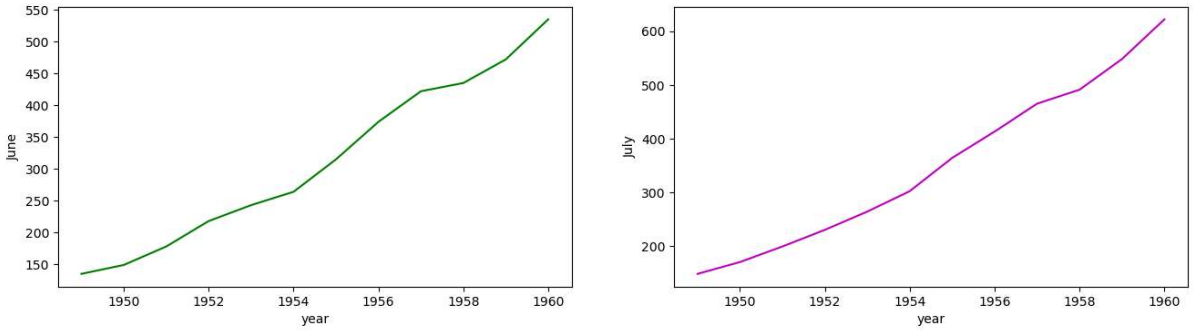


In [14]:
```
df=df.pivot(index="year",columns="month",values="passengers")
df
```

Out[14]:

month year	April	August	December	February	January	July	June	March	May	November	Octob
1949	129	148	118	118	112	148	135	132	121	104	1
1950	135	170	140	126	115	170	149	141	125	114	1
1951	163	199	166	150	145	199	178	178	172	146	1
1952	181	242	194	180	171	230	218	193	183	172	1
1953	235	272	201	196	196	264	243	236	229	180	2
1954	227	293	229	188	204	302	264	235	234	203	2
1955	269	347	278	233	242	364	315	267	270	237	2
1956	313	405	306	277	284	413	374	317	318	271	3
1957	348	467	336	301	315	465	422	356	355	305	3
1958	348	505	337	318	340	491	435	362	363	310	3
1959	396	559	405	342	360	548	472	406	420	362	4
1960	461	606	432	391	417	622	535	419	472	390	4

In [20]:
```
plt.figure(figsize=(16,4))
plt.subplot(1,2,1)
sns.lineplot(x=df.index,y="June",data=df,color="g")
plt.subplot(1,2,2)
sns.lineplot(x=df.index,y="July",data=df,color="m")
plt.show()
```



In [25]:
```
# scatterplot
df=pd.read_csv("tips.csv")
df
```

Out[25]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

In [27]:
```
len(sns.get_dataset_names())
```

Out[27]:
```
22
```

In [28]:
```
sns.get_dataset_names()
```

Out[28]:
```
['anagrams',
 'anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'dowjones',
 'exercise',
 'flights',
 'fmri',
 'geyser',
 'glue',
 'healthexp',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'seaice',
 'taxis',
 'tips',
 'titanic']
```

In [29]:
```
df=sns.load_dataset("titanic")
df
```

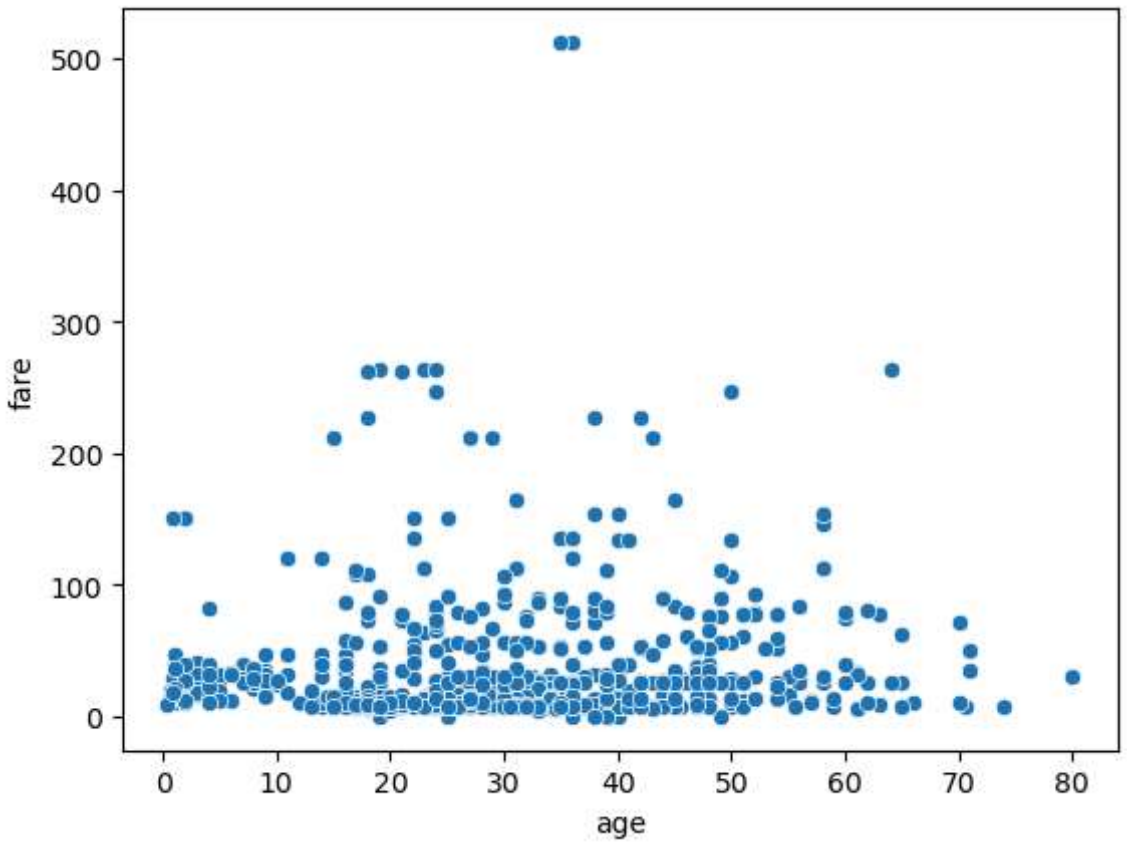
Out[29]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
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
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True
...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	Second	man	True
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True

891 rows × 15 columns

In [30]:
```
sns.scatterplot(x="age",y="fare",data=df)
```

Out[30]:
```
<Axes: xlabel='age', ylabel='fare'>
```



In [32]:
```
df.rename(columns={"adult_male":"adultmale"})
```

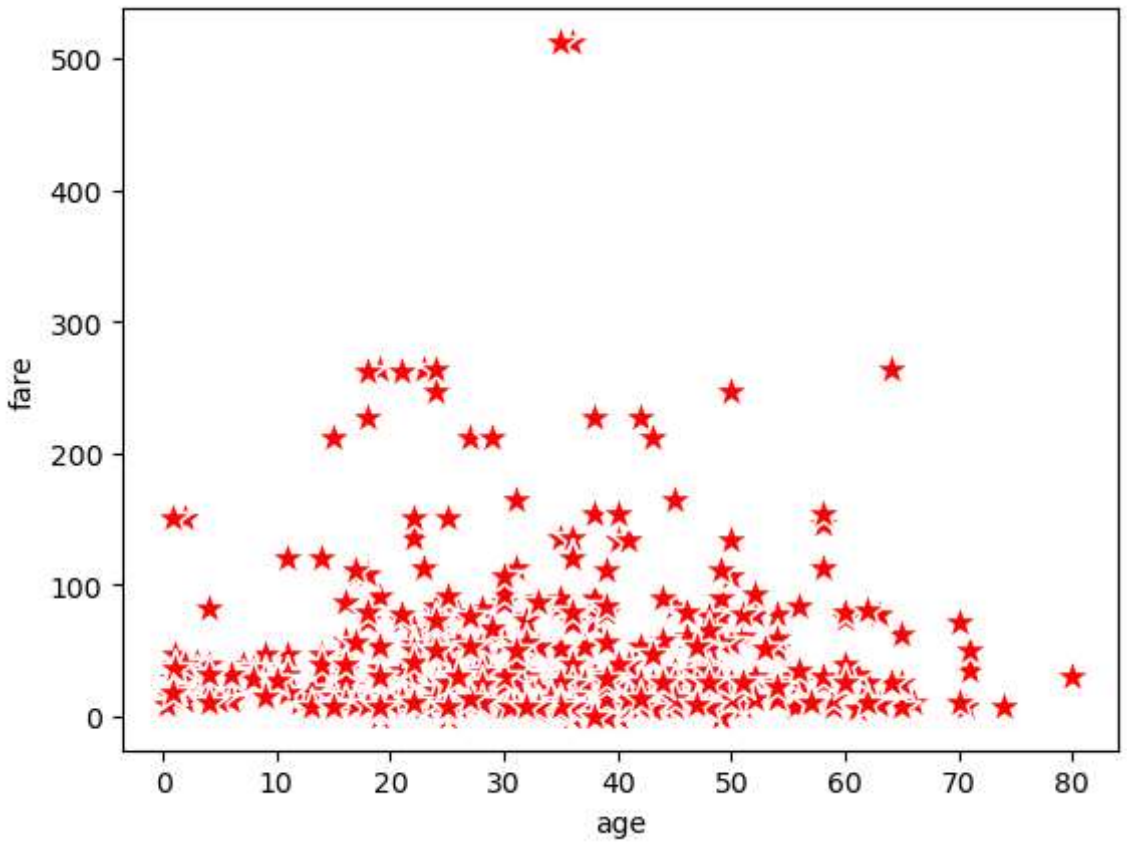
Out[32]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adultmale
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
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
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True
...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	Second	man	True
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True

891 rows × 15 columns

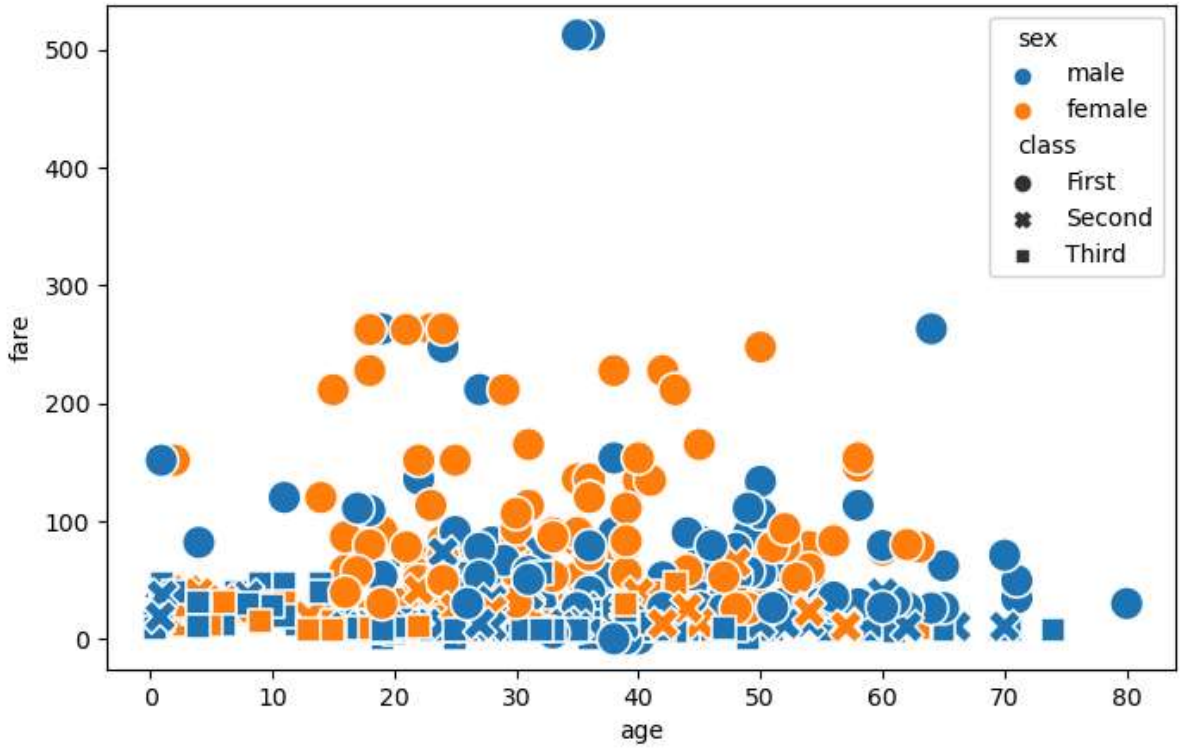
In [38]:
```
sns.scatterplot(x="age",y="fare",data=df,color="r",marker="*",s=200)
```

Out[38]:
```
<Axes: xlabel='age', ylabel='fare'>
```



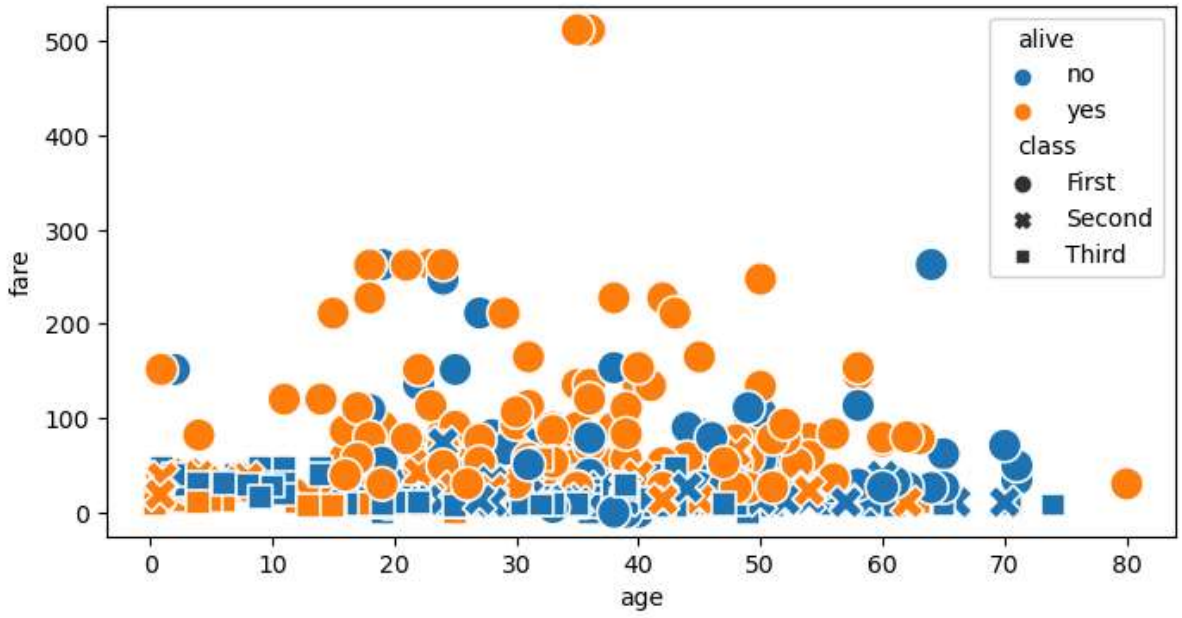
In [44]:
```
plt.figure(figsize=(8,5))
sns.scatterplot(x="age",y="fare",data=df,color="r",marker="*",s=200,hue="sex",style
```

Out[44]:
```
<Axes: xlabel='age', ylabel='fare'>
```



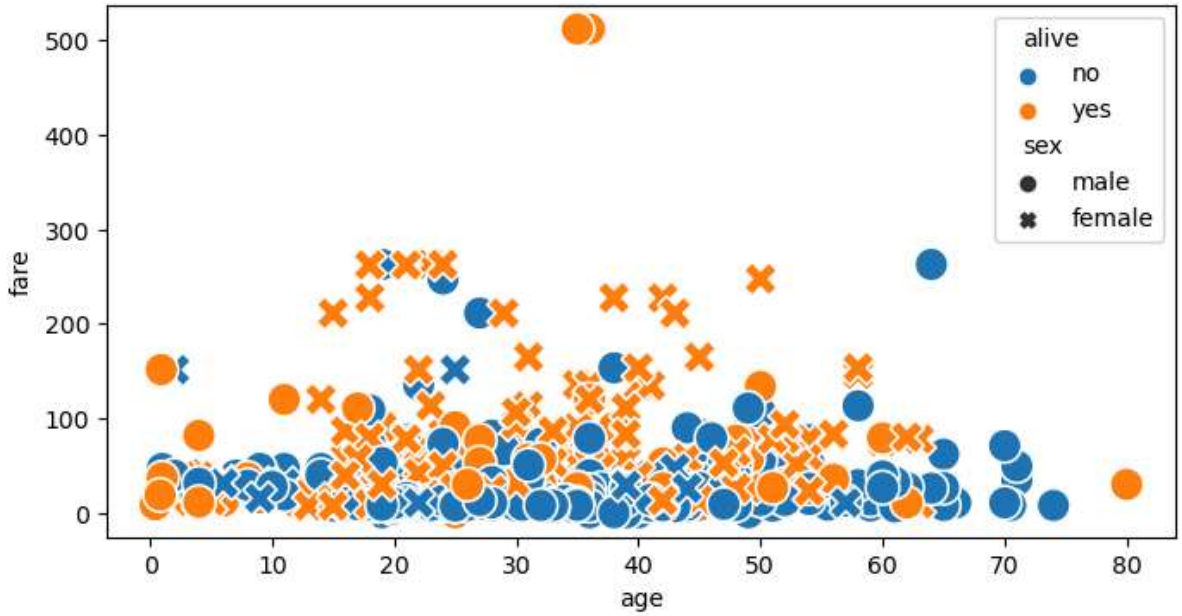
In [47]:
```
plt.figure(figsize=(8,4))
sns.scatterplot(x="age",y="fare",data=df,color="r",marker="*",s=200,hue="alive",sty
```

Out[47]:
```
<Axes: xlabel='age', ylabel='fare'>
```



In [48]:
```
plt.figure(figsize=(8,4))
sns.scatterplot(x="age",y="fare",data=df,color="r",marker="*",s=200,hue="alive",sty
```

Out[48]:
```
<Axes: xlabel='age', ylabel='fare'>
```



In [56]:
```
# pair plot
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df=sns.load_dataset("tips")
df
```

Out[56]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

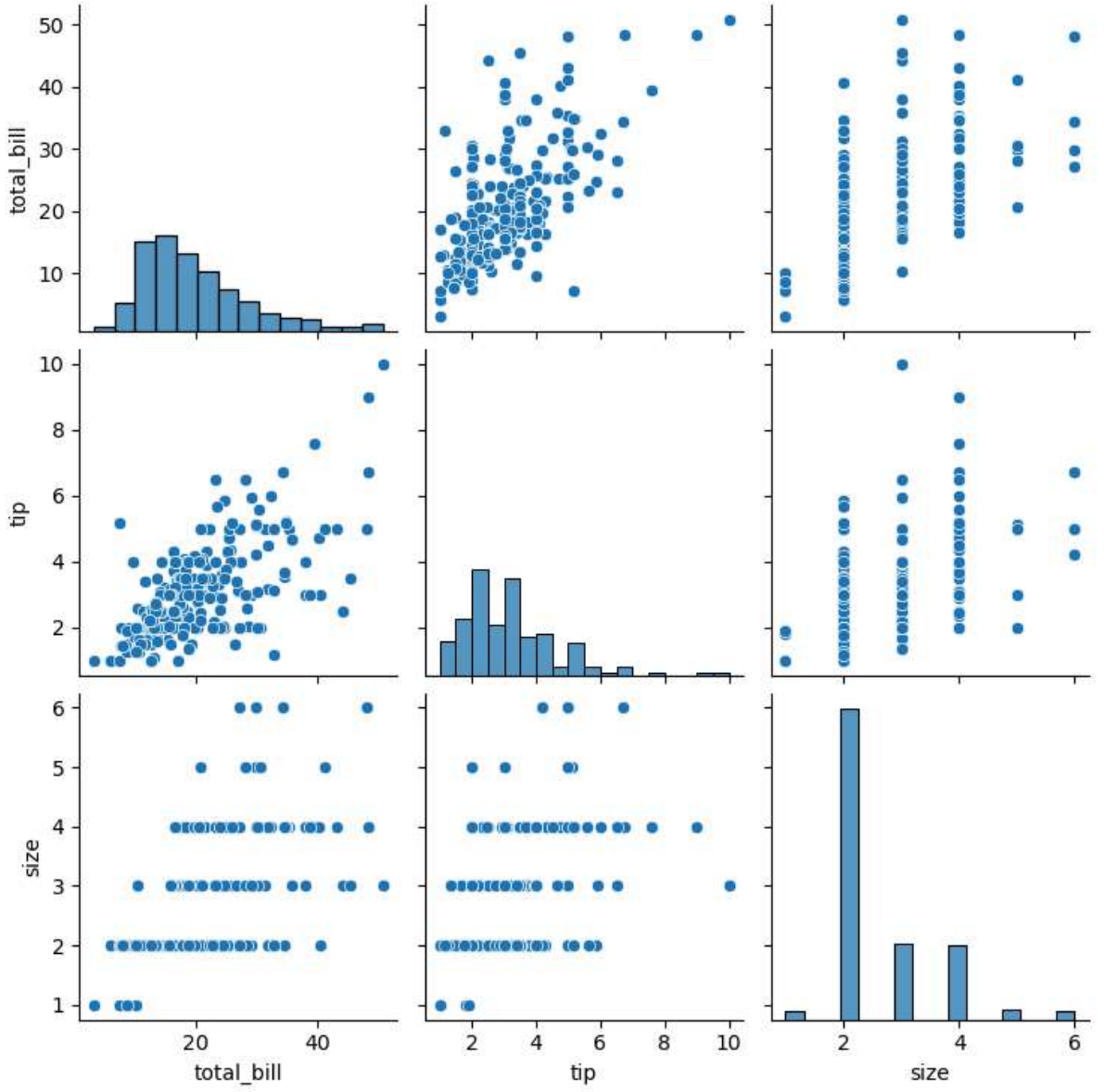
244 rows × 7 columns

In [57]:
```
df.describe()
```

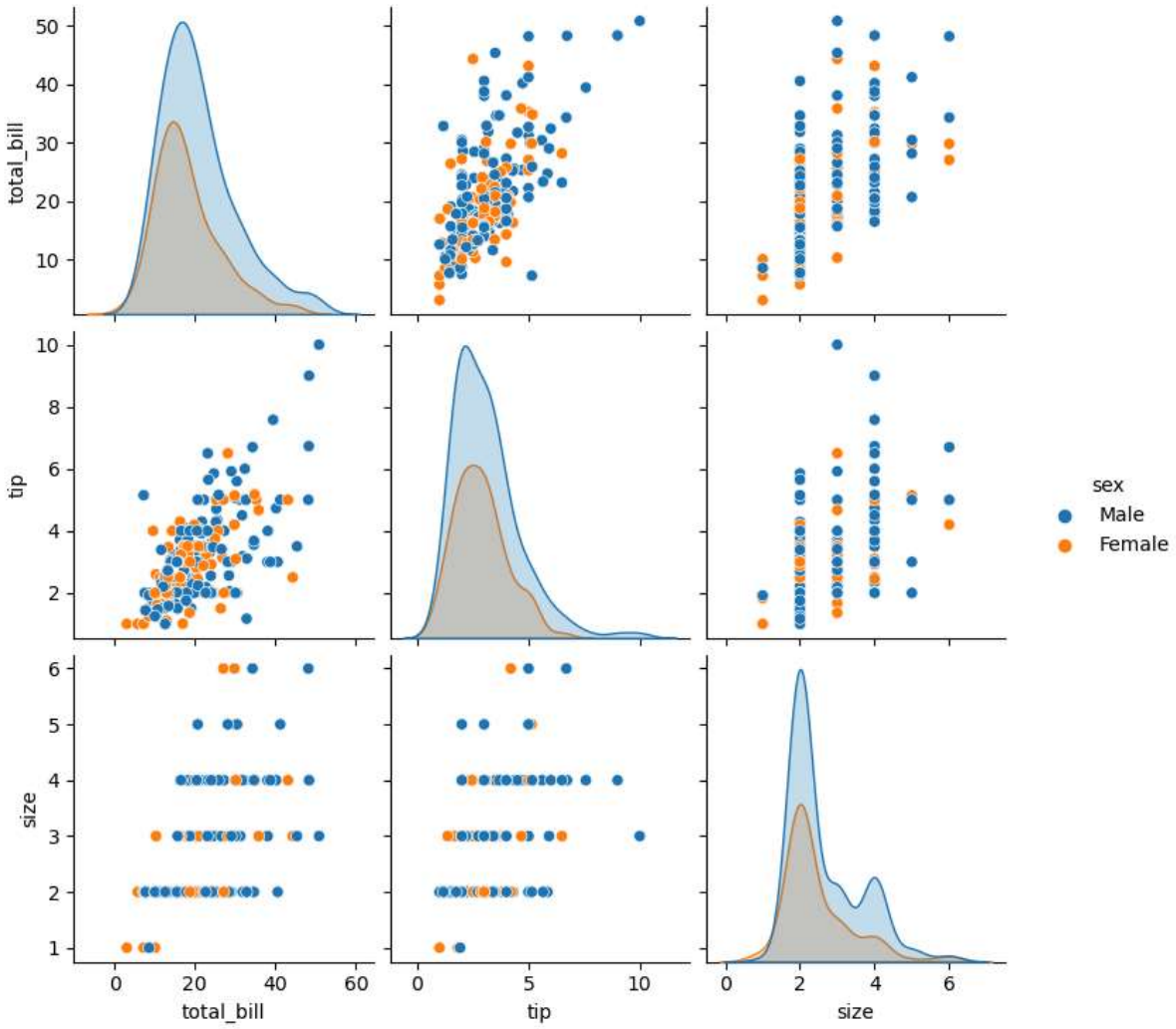
Out[57]:

	total_bill	tip	size
count	244.000000	244.000000	244.000000
mean	19.785943	2.998279	2.569672
std	8.902412	1.383638	0.951100
min	3.070000	1.000000	1.000000
25%	13.347500	2.000000	2.000000
50%	17.795000	2.900000	2.000000
75%	24.127500	3.562500	3.000000
max	50.810000	10.000000	6.000000

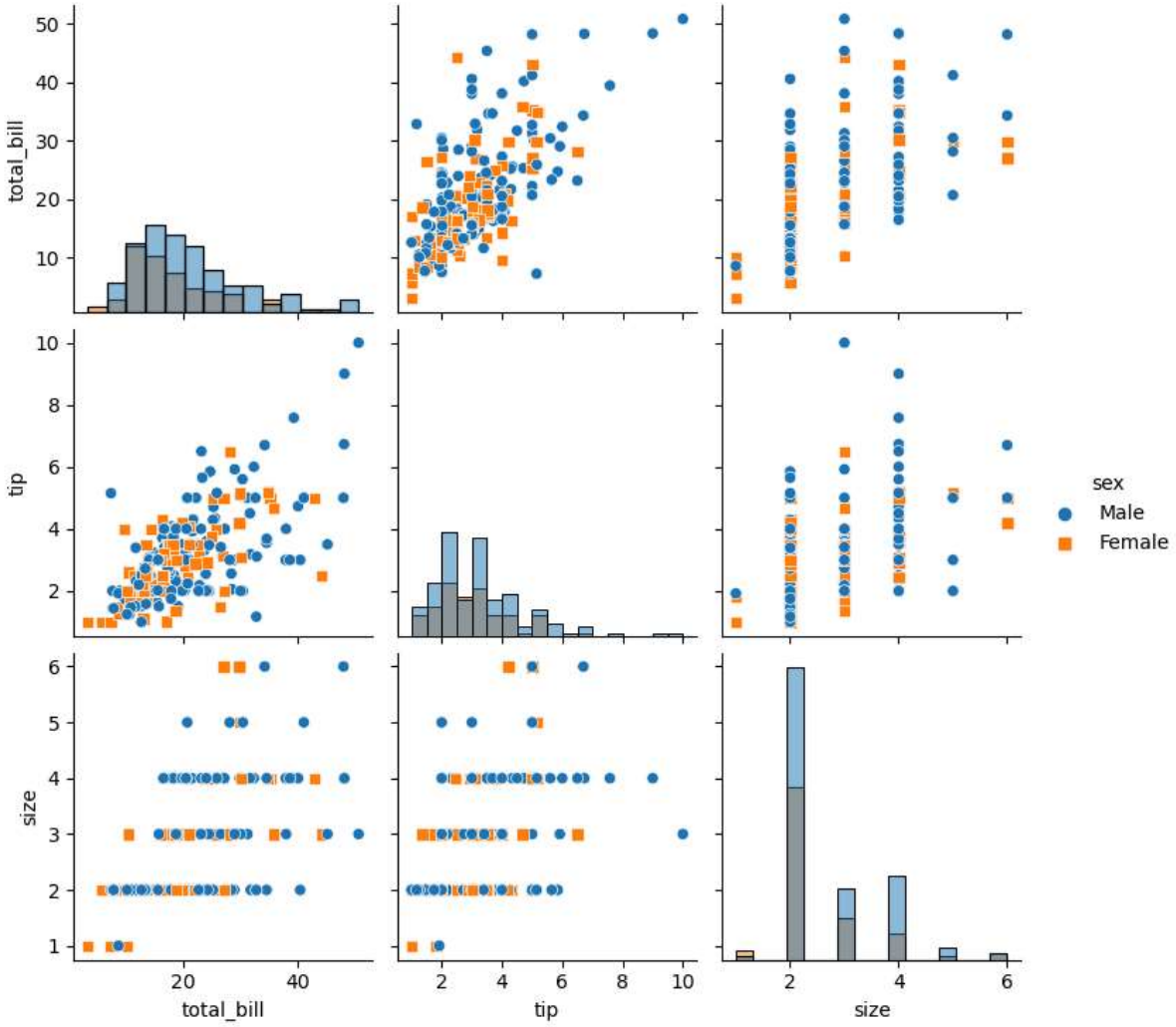
In [58]:
```
sns.pairplot(df)
plt.show()
```



In [59]:
```
sns.pairplot(df,hue="sex")
plt.show()
```

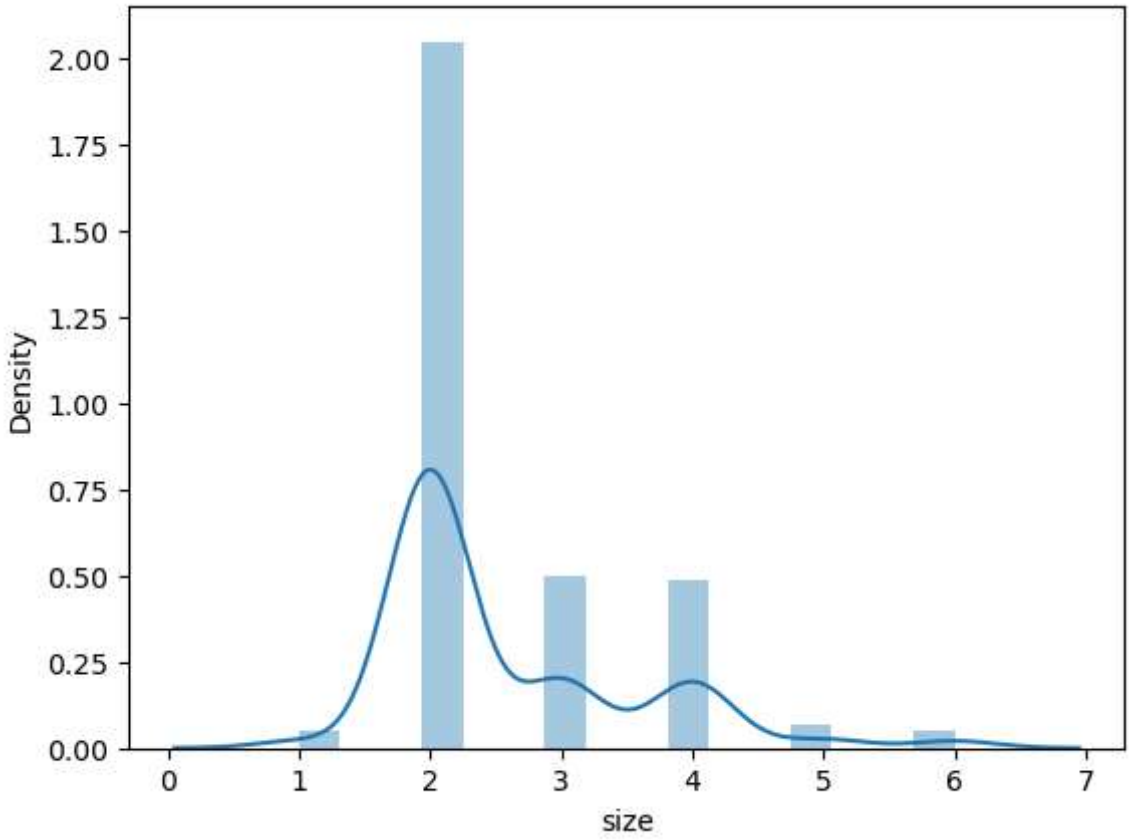


In [63]:
```
sns.pairplot(df,hue="sex",diag_kind="hist", markers=["o", "s"])
plt.show()
```



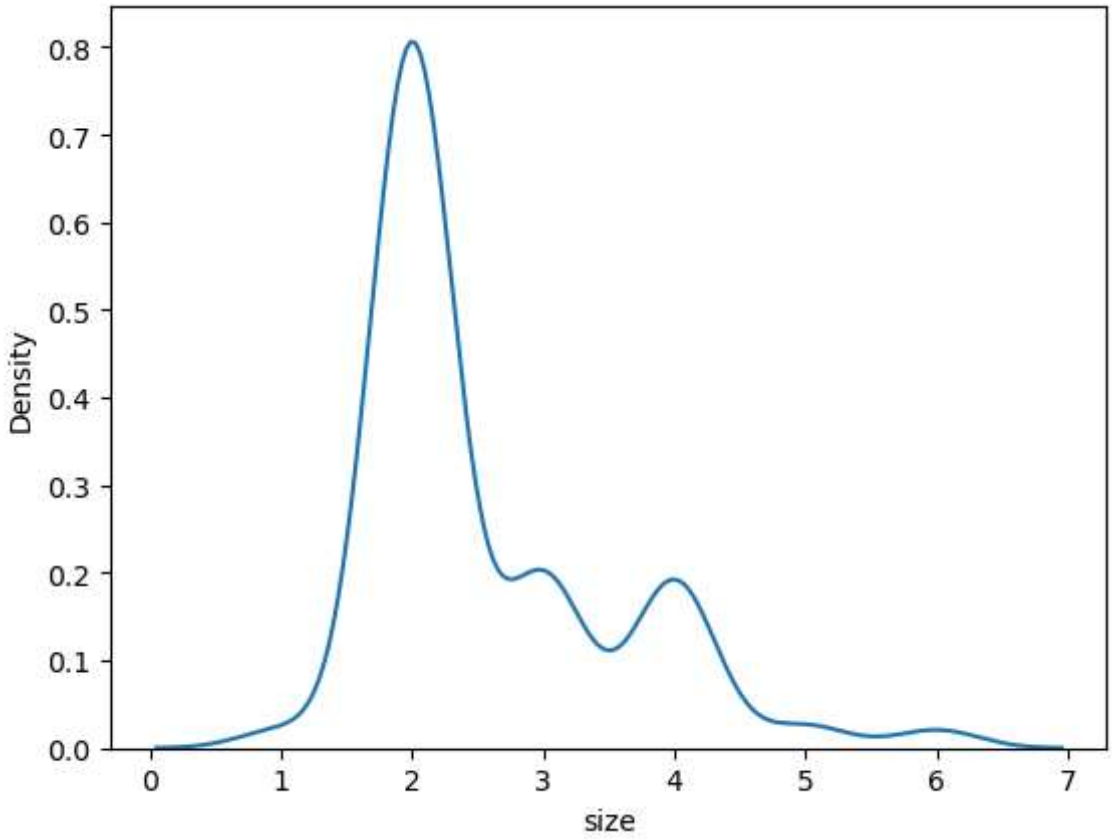
In [67]:
```
# distplot
import warnings
warnings.filterwarnings("ignore")
sns.distplot(df["size"])
```

Out[67]:
```
<Axes: xlabel='size', ylabel='Density'>
```



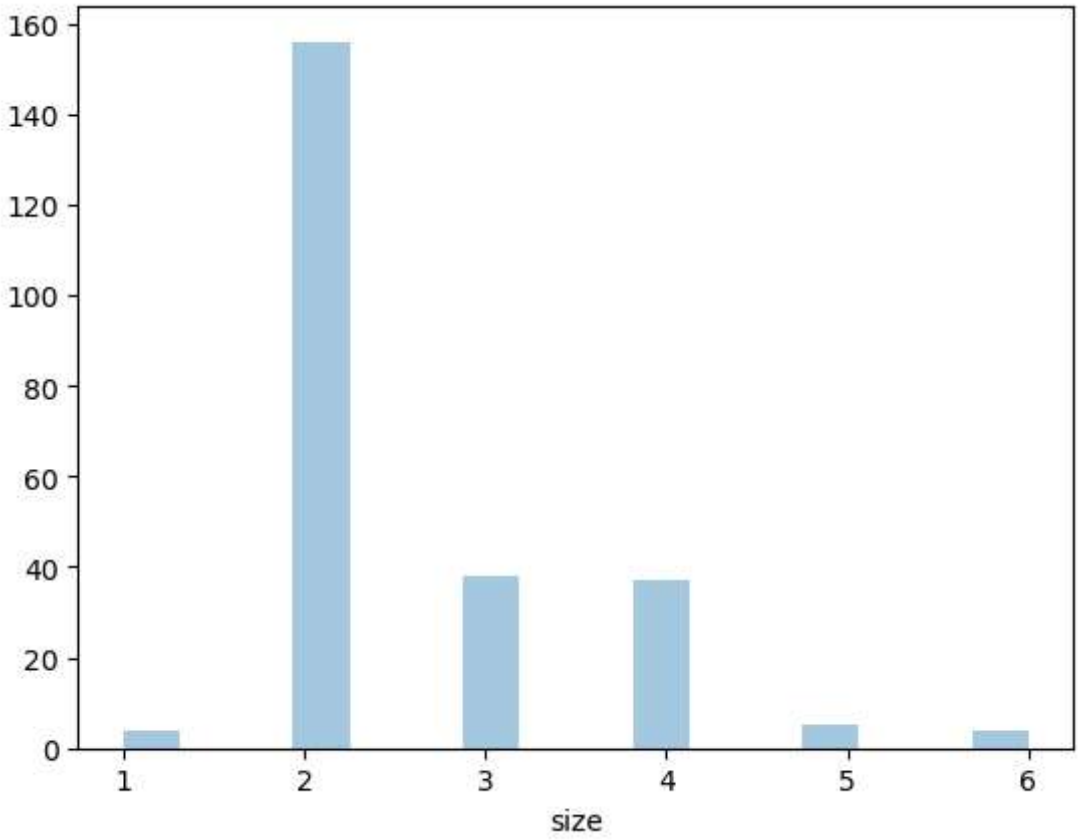
In [68]:
```
sns.distplot(df["size"],hist=False)
```

Out[68]:
```
<Axes: xlabel='size', ylabel='Density'>
```



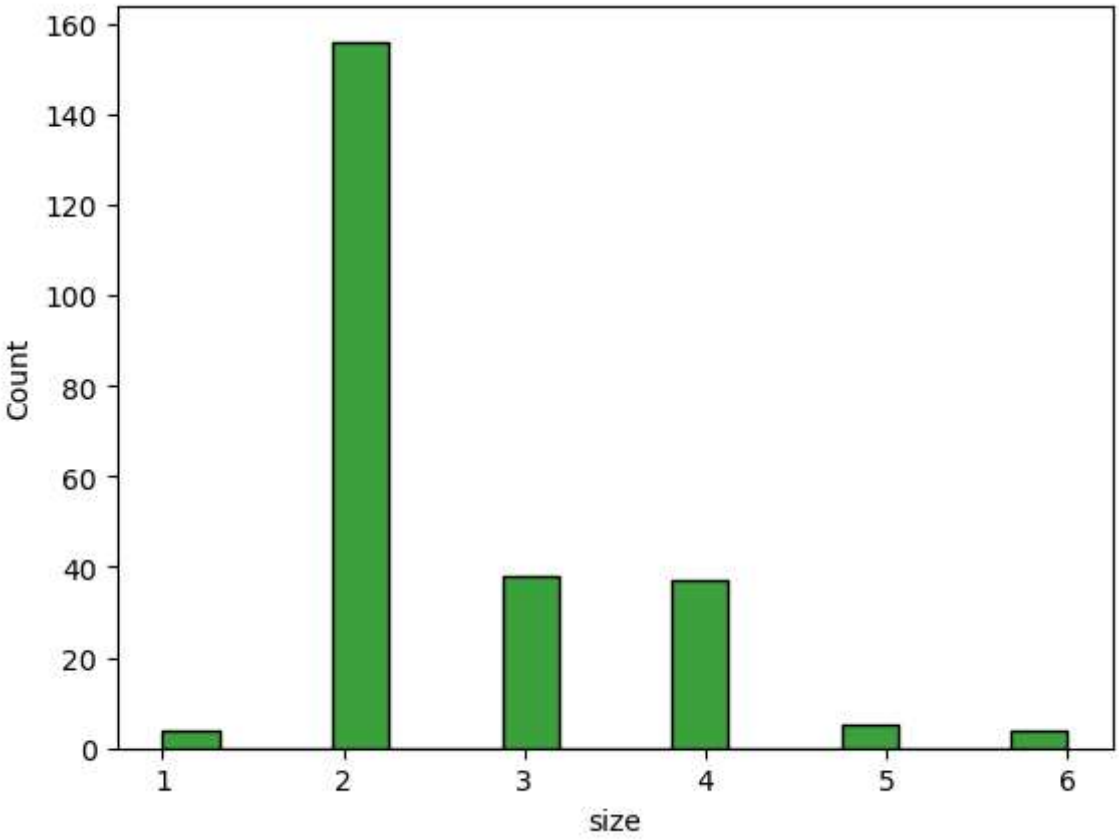
In [70]:
```
sns.distplot(df["size"],kde=False)
```

Out[70]:
```
<Axes: xlabel='size'>
```



In [72]:
```
sns.histplot(df["size"],color="g")
```

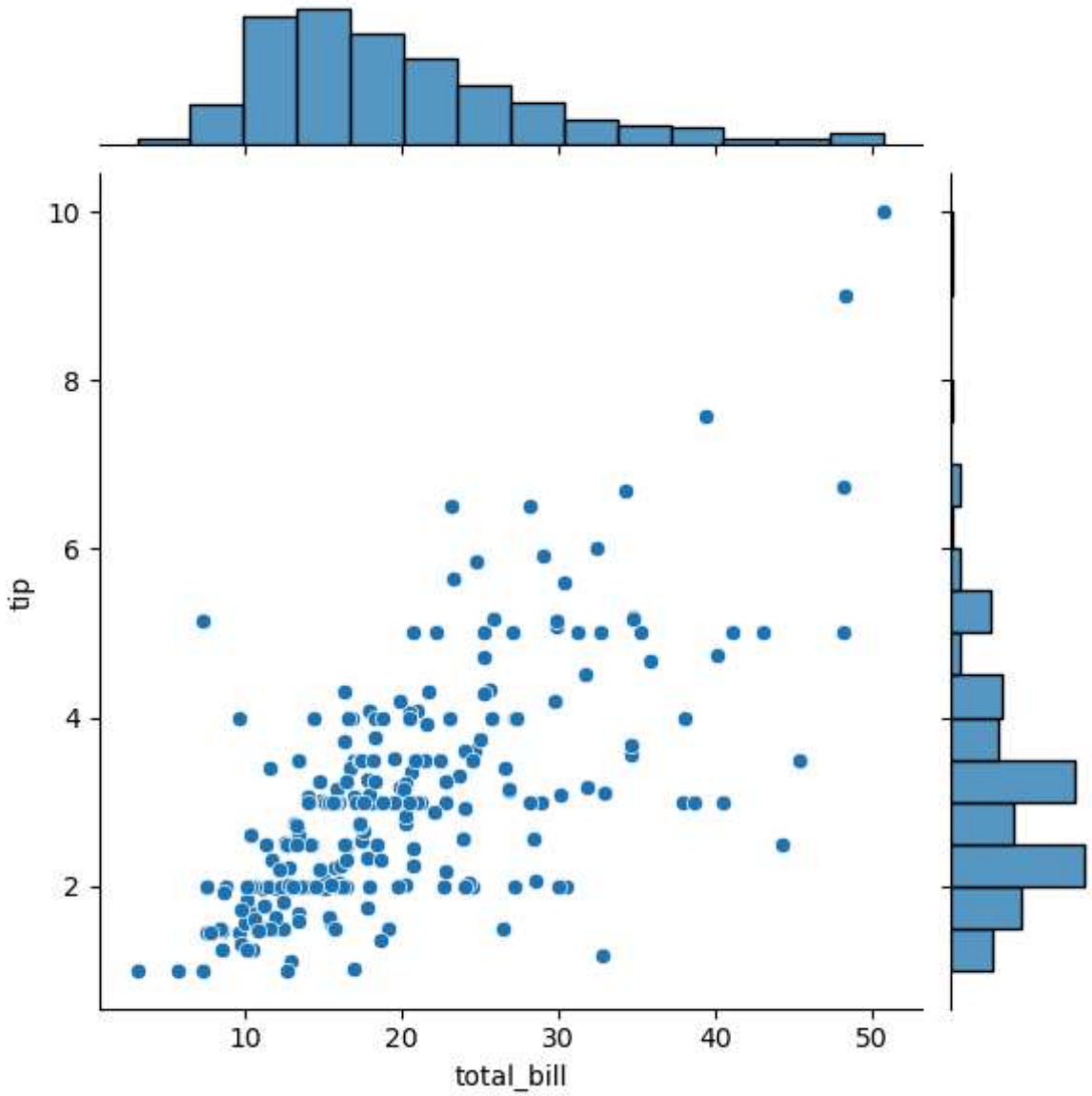
Out[72]:
```
<Axes: xlabel='size', ylabel='Count'>
```



In [73]:
```
# joint plot
```

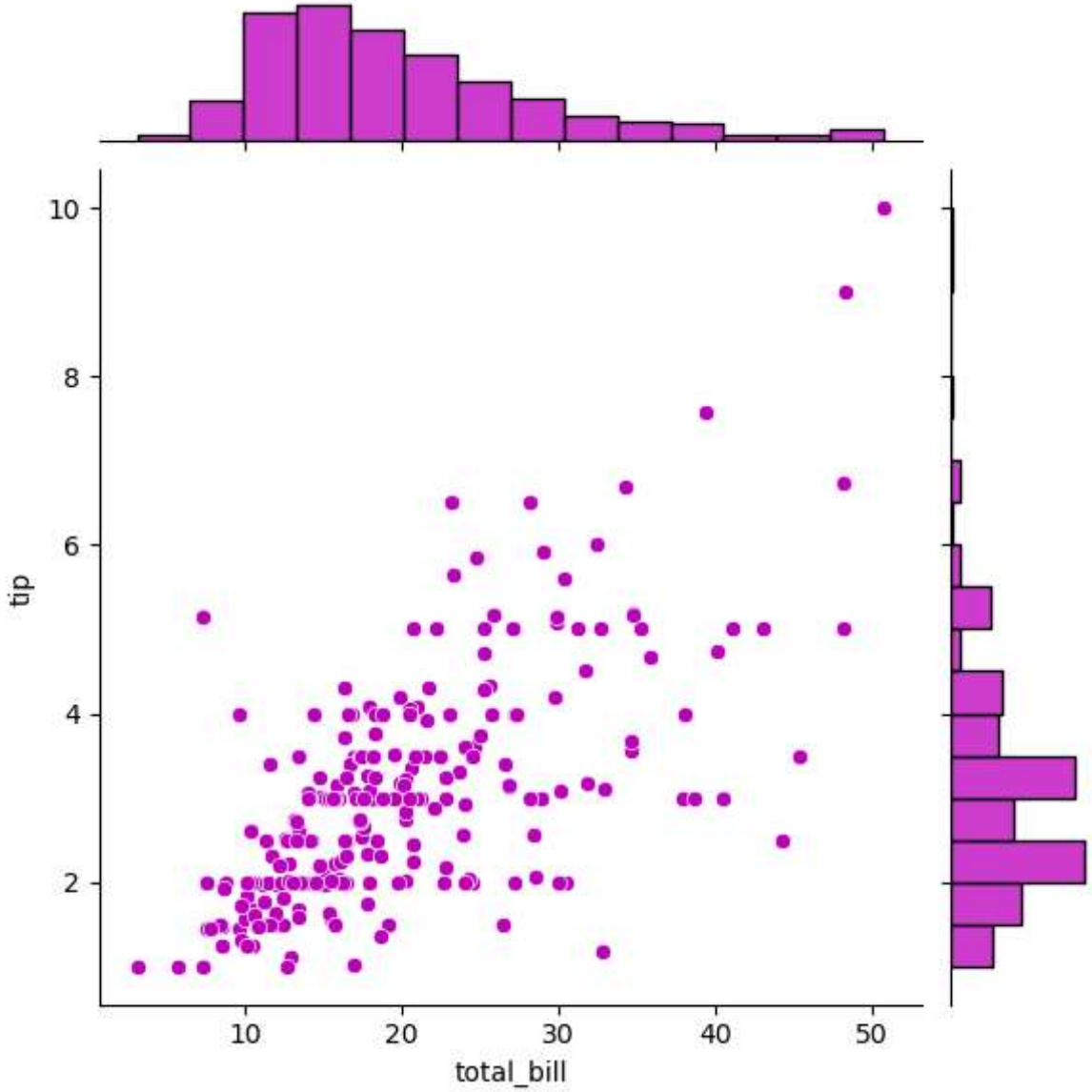
In [74]:
```
sns.jointplot(x="total_bill",y="tip",data=df)
```

Out[74]: <seaborn.axisgrid.JointGrid at 0x1360f1925d0>



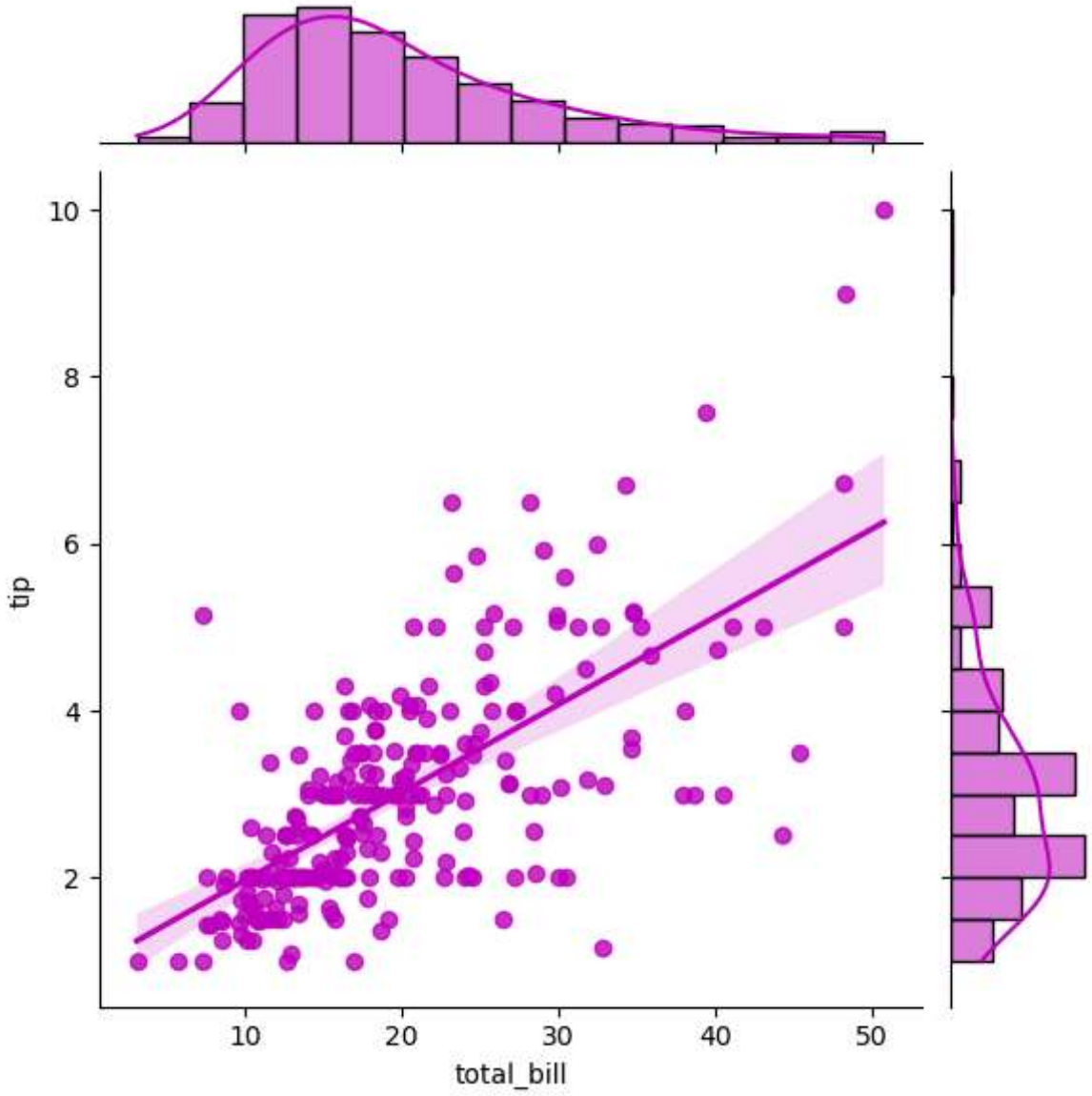
In [75]:
```
sns.jointplot(x="total_bill",y="tip",data=df,color="m")
```

Out[75]: <seaborn.axisgrid.JointGrid at 0x1360c6ff9d0>



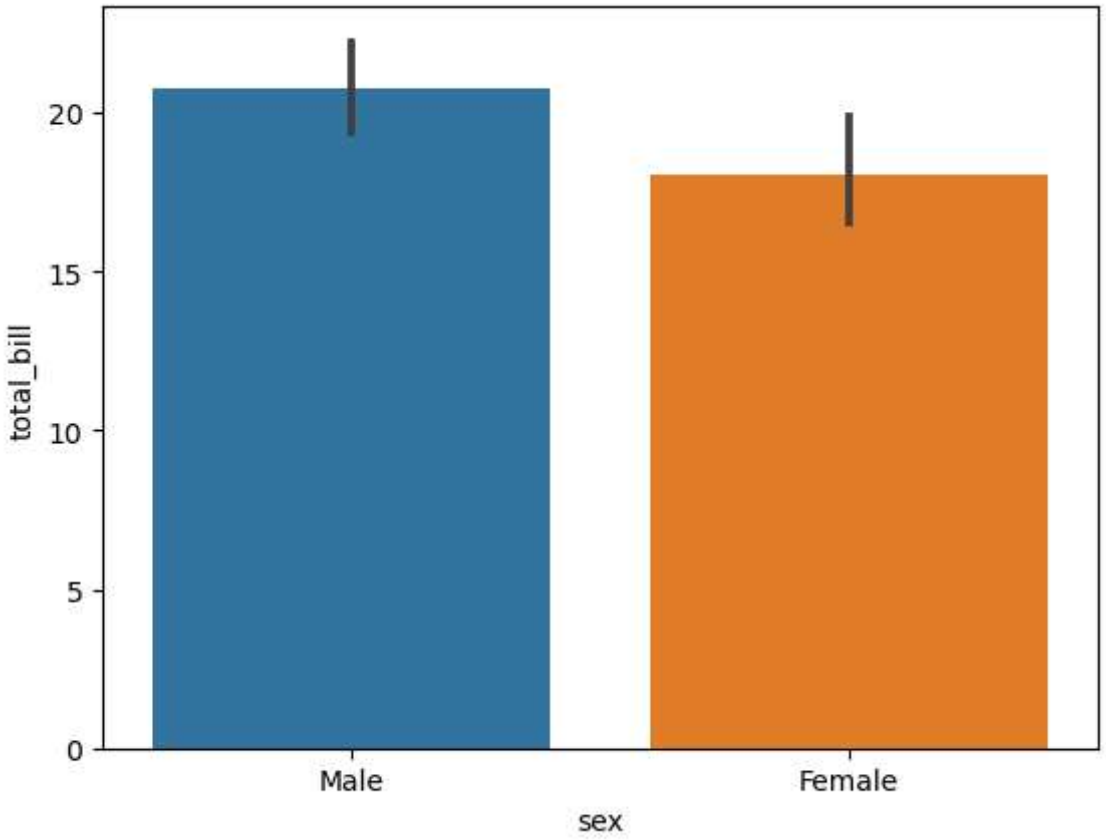
In [78]:
```
sns.jointplot(x="total_bill",y="tip",data=df,color="m",kind="reg")
```

Out[78]:
```
<seaborn.axisgrid.JointGrid at 0x13608432450>
```



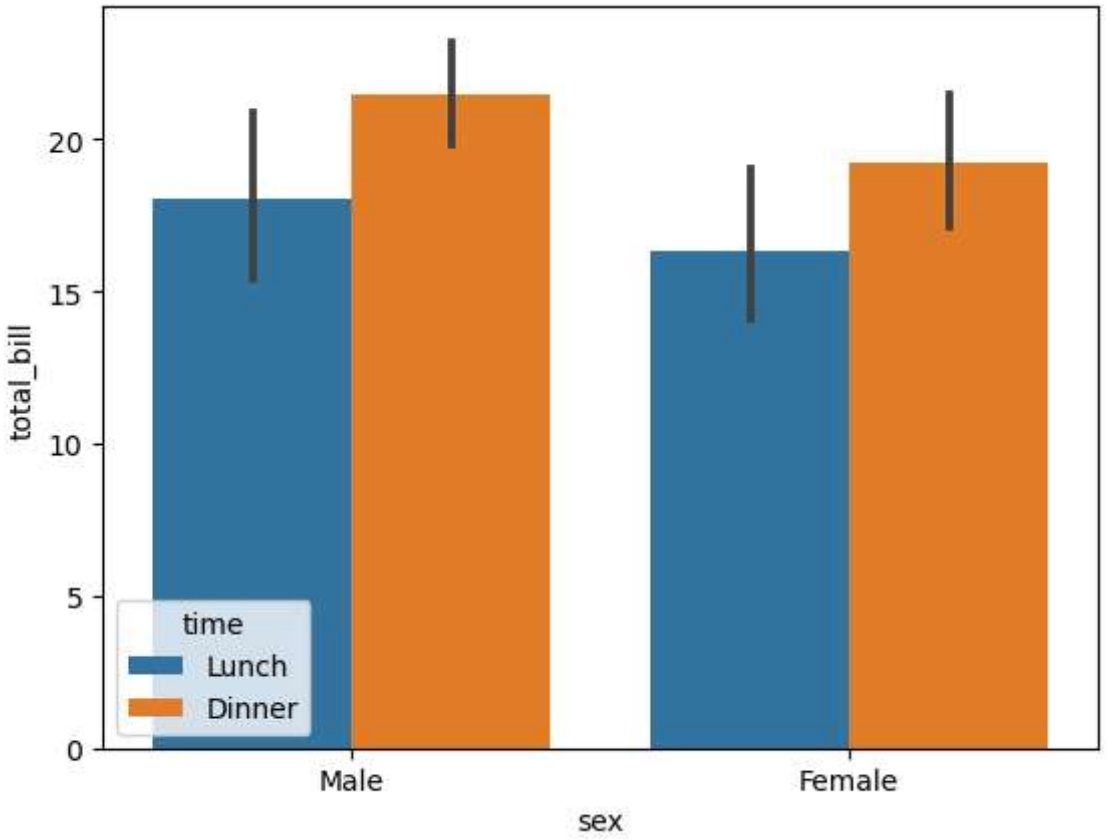
In [81]:
```
# barplot
sns.barplot(x="sex",y="total_bill",data=df)
```

Out[81]:
```
<Axes: xlabel='sex', ylabel='total_bill'>
```



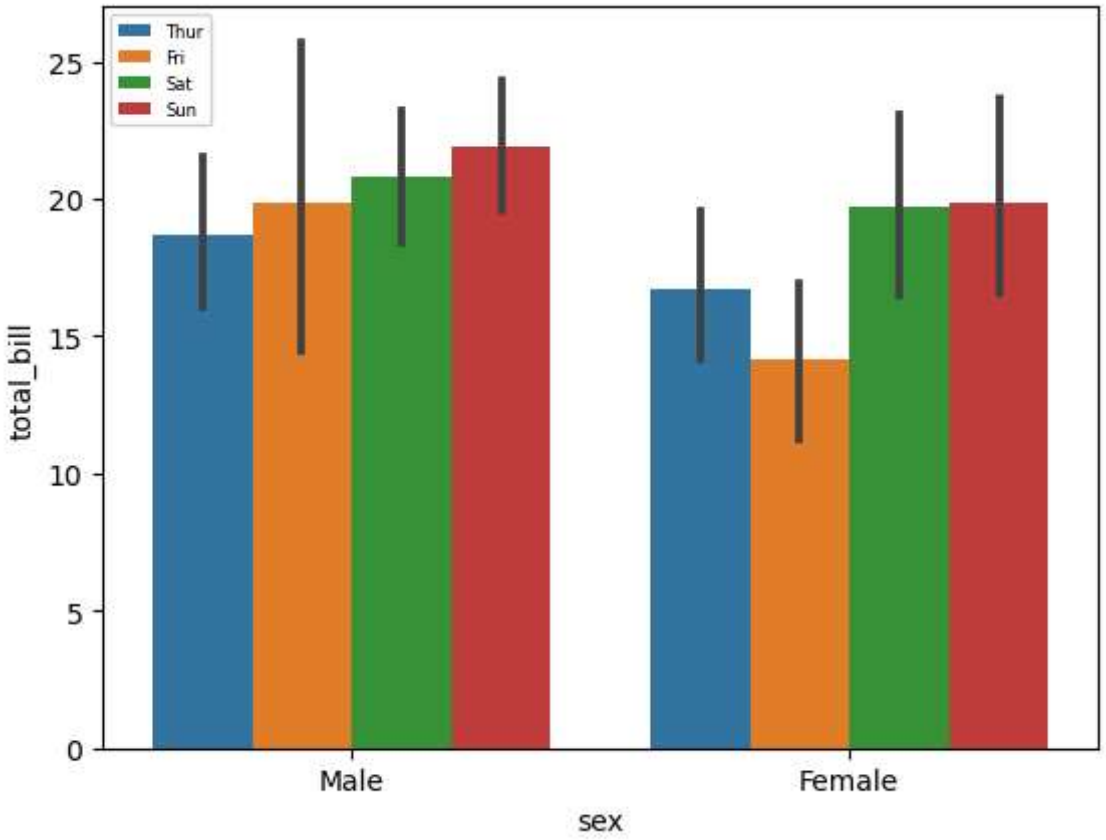
In [82]:
```
sns.barplot(x="sex",y="total_bill",data=df,hue="time")
```

Out[82]:
```
<Axes: xlabel='sex', ylabel='total_bill'>
```



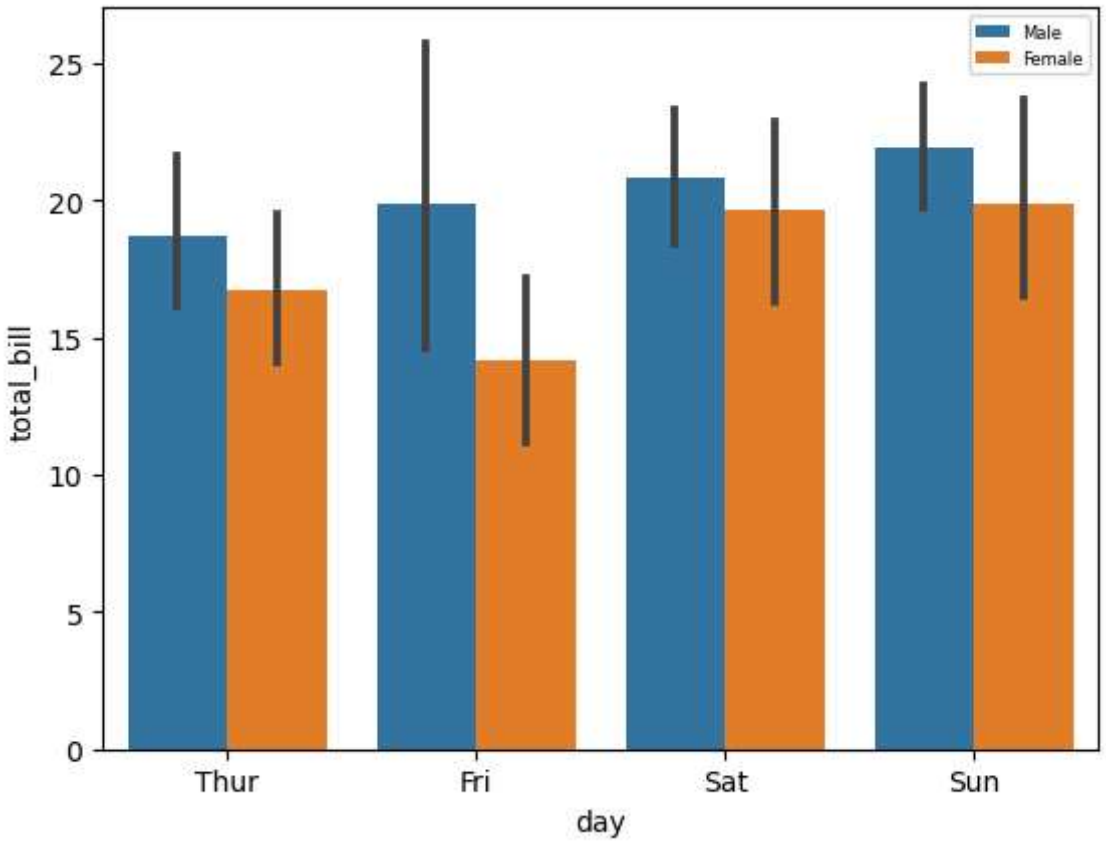
In [85]:
```
sns.barplot(x="sex",y="total_bill",data=df,hue="day")
plt.legend(fontsize=6)
```

Out[85]:
```
<matplotlib.legend.Legend at 0x1360bb890d0>
```

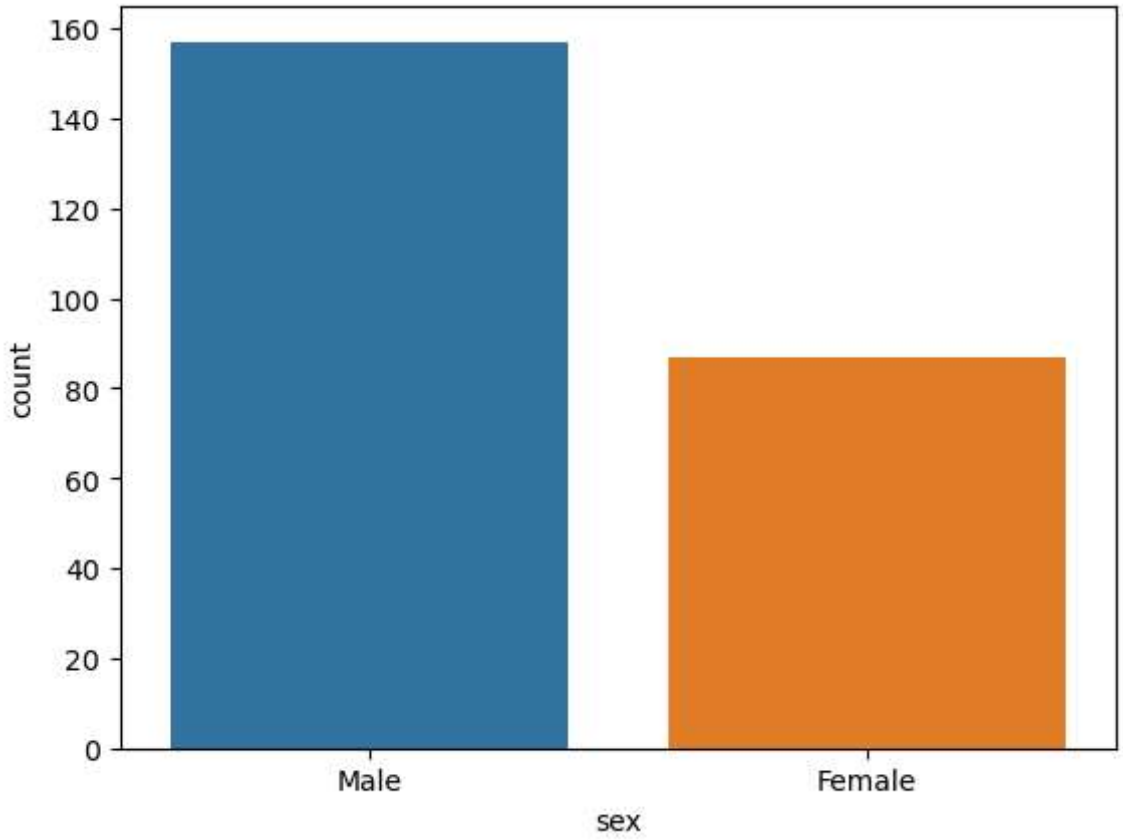


In [89]:
```
sns.barplot(x="day",y="total_bill",data=df,hue="sex")
plt.legend(fontsize=6)
```

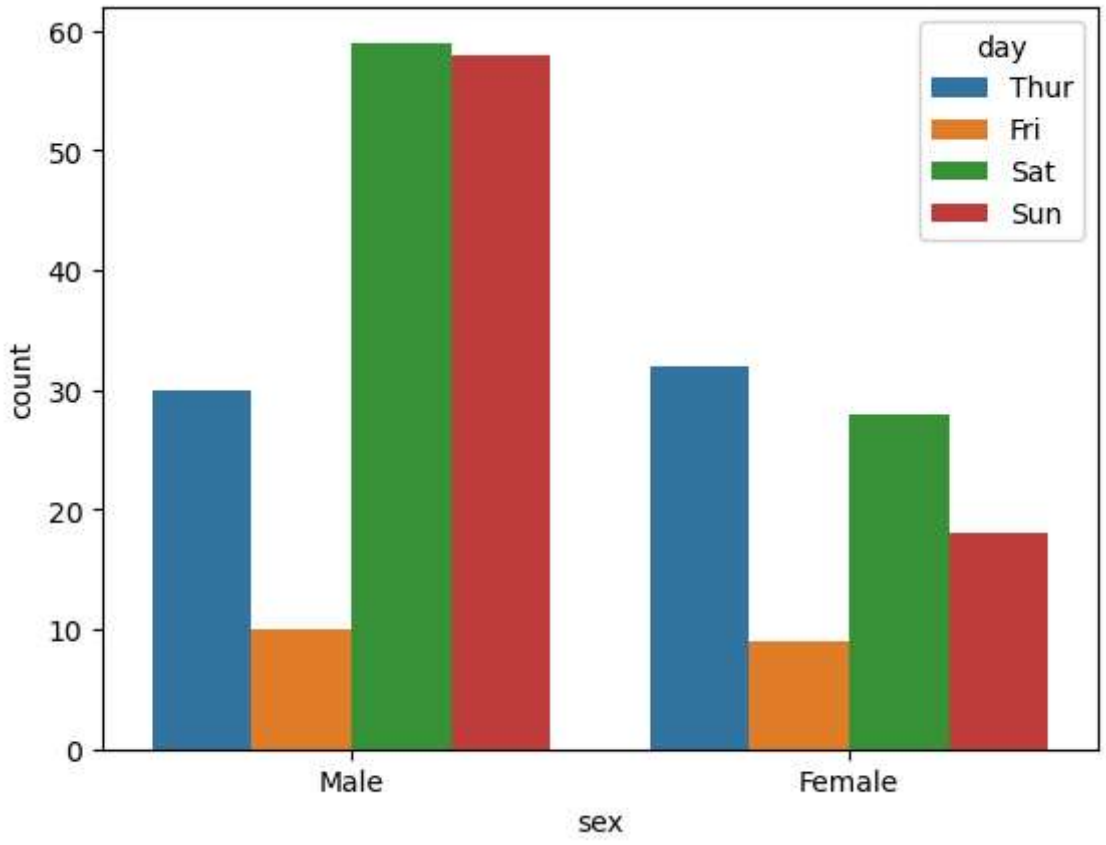
Out[89]:
```
<matplotlib.legend.Legend at 0x1360ab5eb90>
```



In [91]:
```
# countplot
sns.countplot(x="sex",data=df)
plt.show()
```



In [94]:
```
sns.countplot(x="sex",data=df,hue="day")
plt.show()
```



In [101…
```
df1=df[["sex"]]
df1
```

Out[101]:

	sex
0	Female
1	Male
2	Male
3	Male
4	Female
...	...
239	Male
240	Female
241	Male
242	Male
243	Female

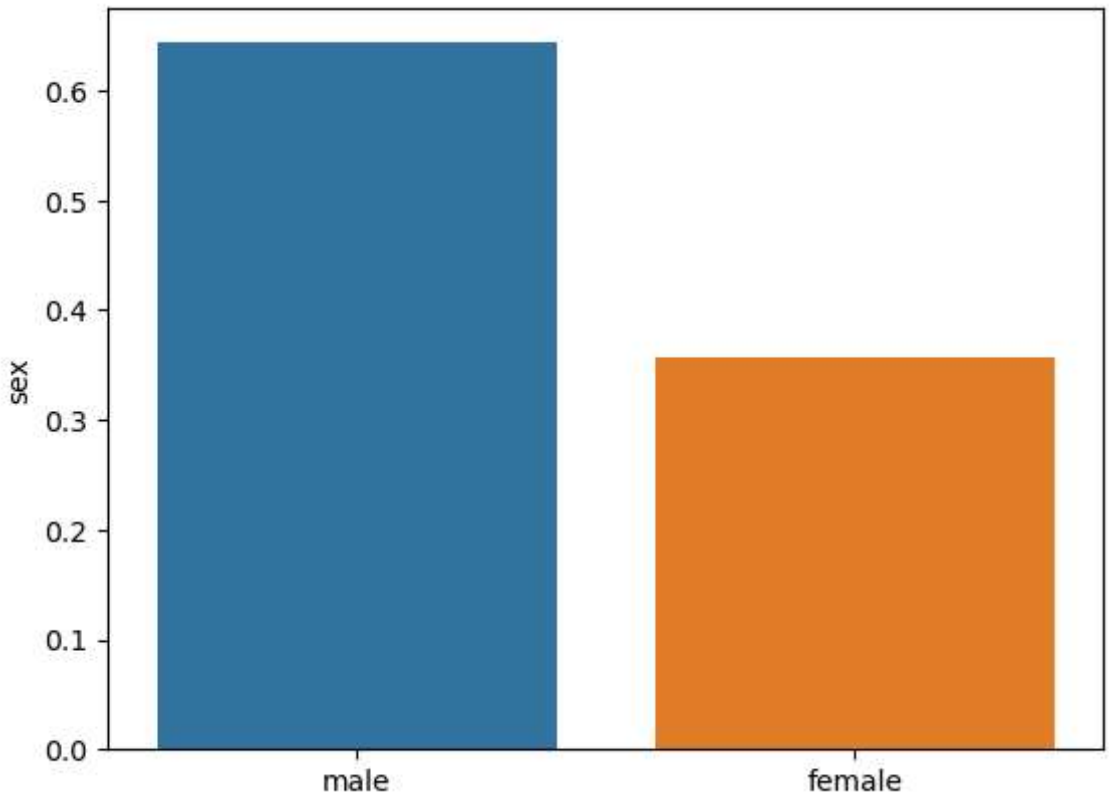
244 rows × 1 columns

In [135…
```
y=df1["sex"].value_counts(normalize=True)
y
```

Out[135]:
```
Male      0.643443
Female    0.356557
Name: sex, dtype: float64
```

In [141…
```
sns.barplot(x=["male","female"],y=df1["sex"].value_counts(normalize=True))
```

Out[141]:
```
<Axes: ylabel='sex'>
```



In [1]:
```
import numpy as np
import pandas as pd
```

```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df=sns.load_dataset("titanic")
df
```

Out[2]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
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
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True
...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	Second	man	True
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True

891 rows × 15 columns

In [3]:
```
df.describe()
```

Out[3]:

	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

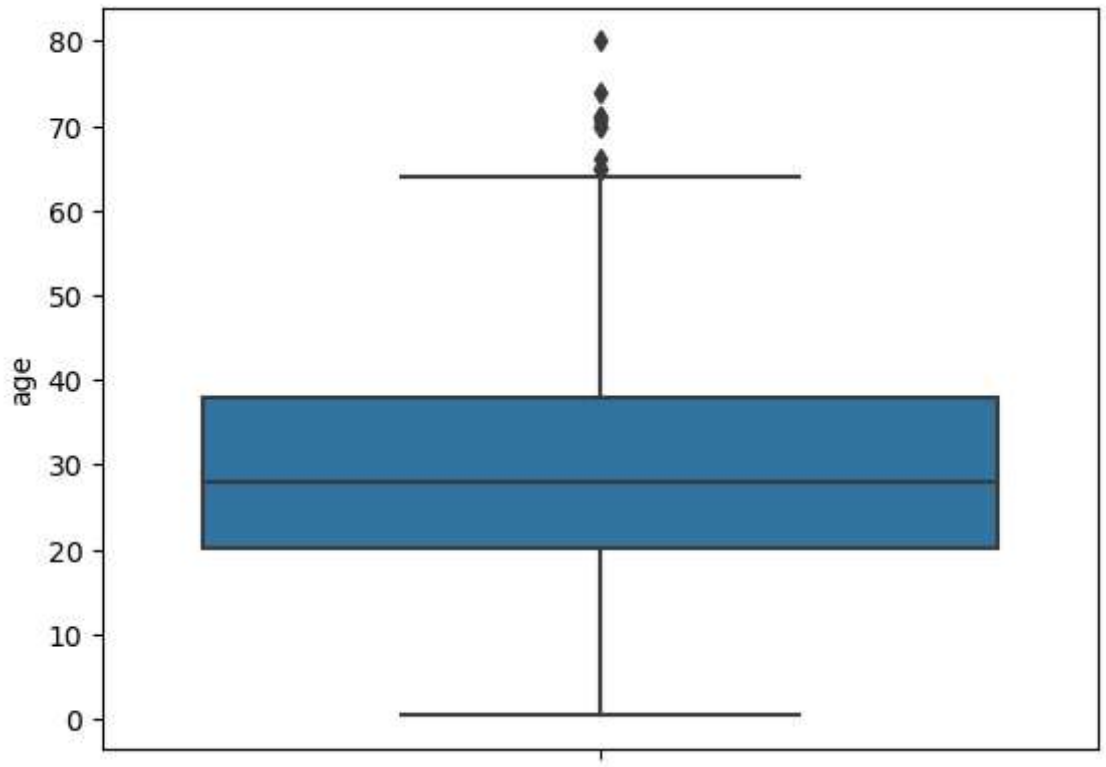
In [4]:
```
df.describe(include="object")
```

Out[4]:

	sex	embarked	who	embark_town	alive
count	891	889	891	889	891
unique	2	3	3	3	2
top	male	S	man	Southampton	no
freq	577	644	537	644	549

In [6]:
```
sns.boxplot(y="age",data=df)
```

Out[6]: <Axes: ylabel='age'>



In [7]:
```
q1=df.age.quantile(0.25)
q1
```

Out[7]: 20.125

In [8]:
```
q3=df.age.quantile(0.75)
q3
```

Out[8]: 38.0

In [9]:
```
iqr=q3-q1
iqr
```

Out[9]: 17.875

In [10]:
```
max=q3+1.5*iqr
max
```

Out[10]: 64.8125

In [11]:
```
min=q1-1.5*iqr
min
```

Out[11]: -6.6875

In [13]:
```
df=df[df.age<=max]
df
```

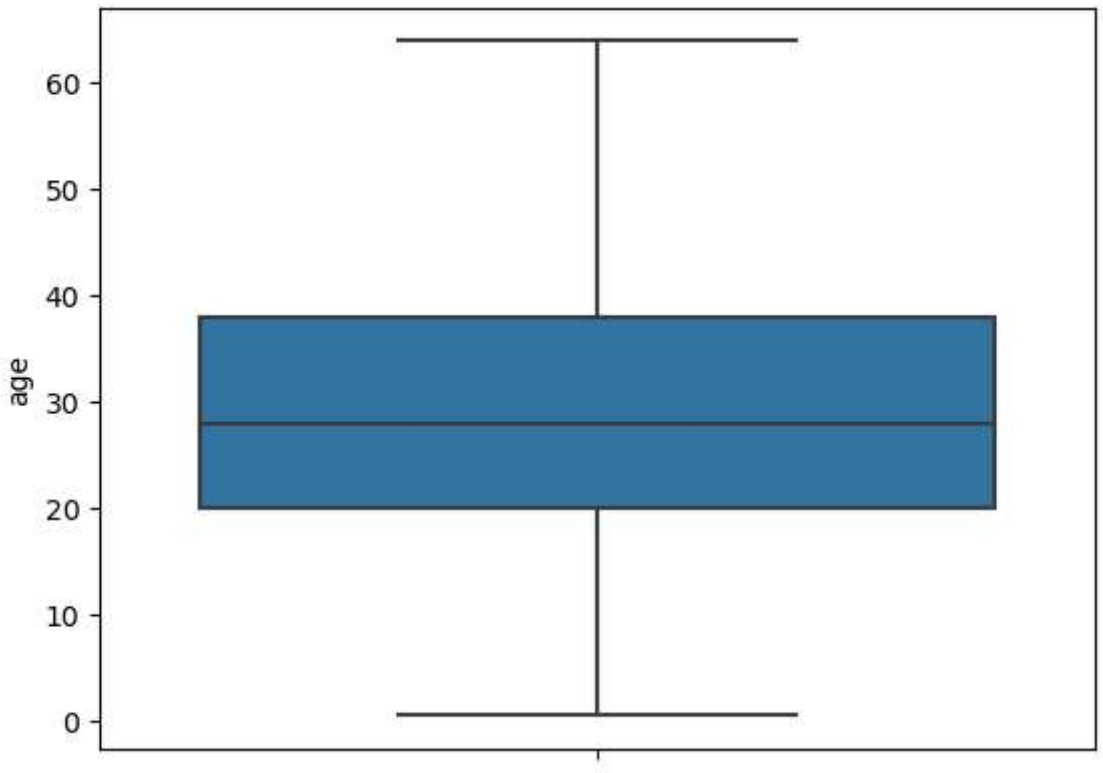
Out[13]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
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
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True
...	...	...	...	...	...	...	...	...	...	...	...
885	0	3	female	39.0	0	5	29.1250	Q	Third	woman	False
886	0	2	male	27.0	0	0	13.0000	S	Second	man	True
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True

703 rows × 15 columns

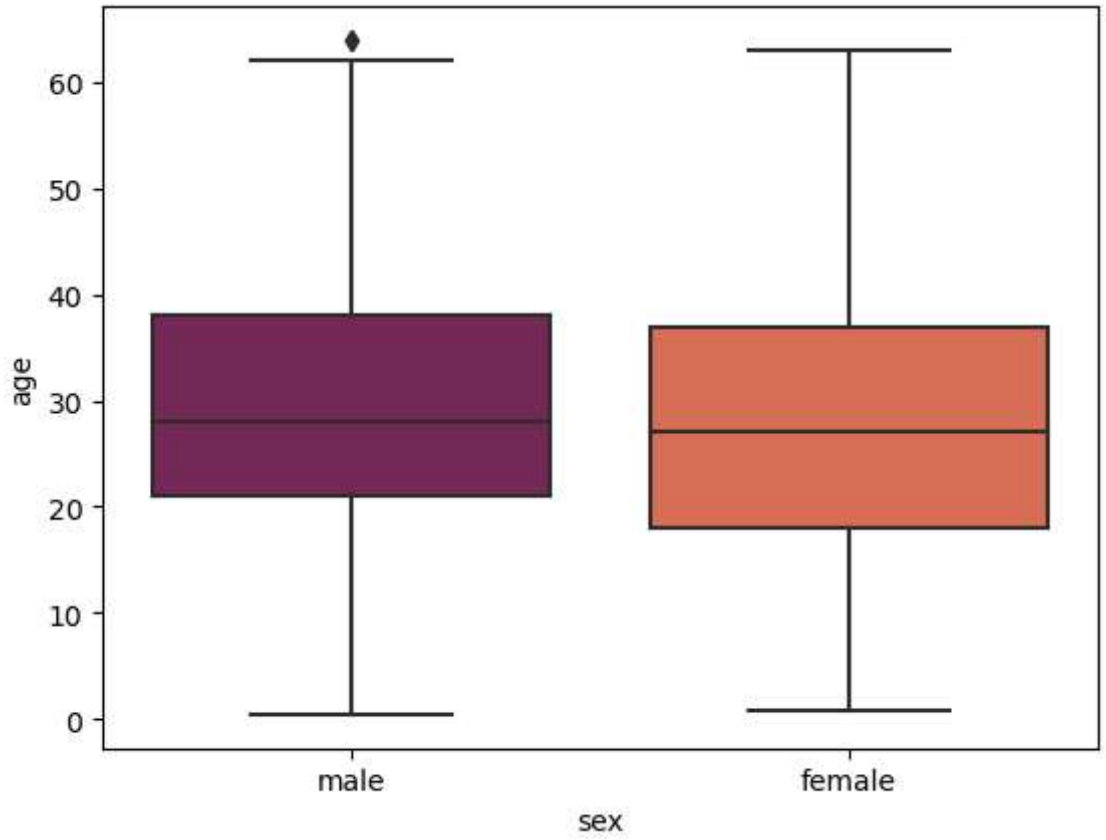
In [14]:
```
sns.boxplot(y="age",data=df)
```

Out[14]:
```
<Axes: ylabel='age'>
```



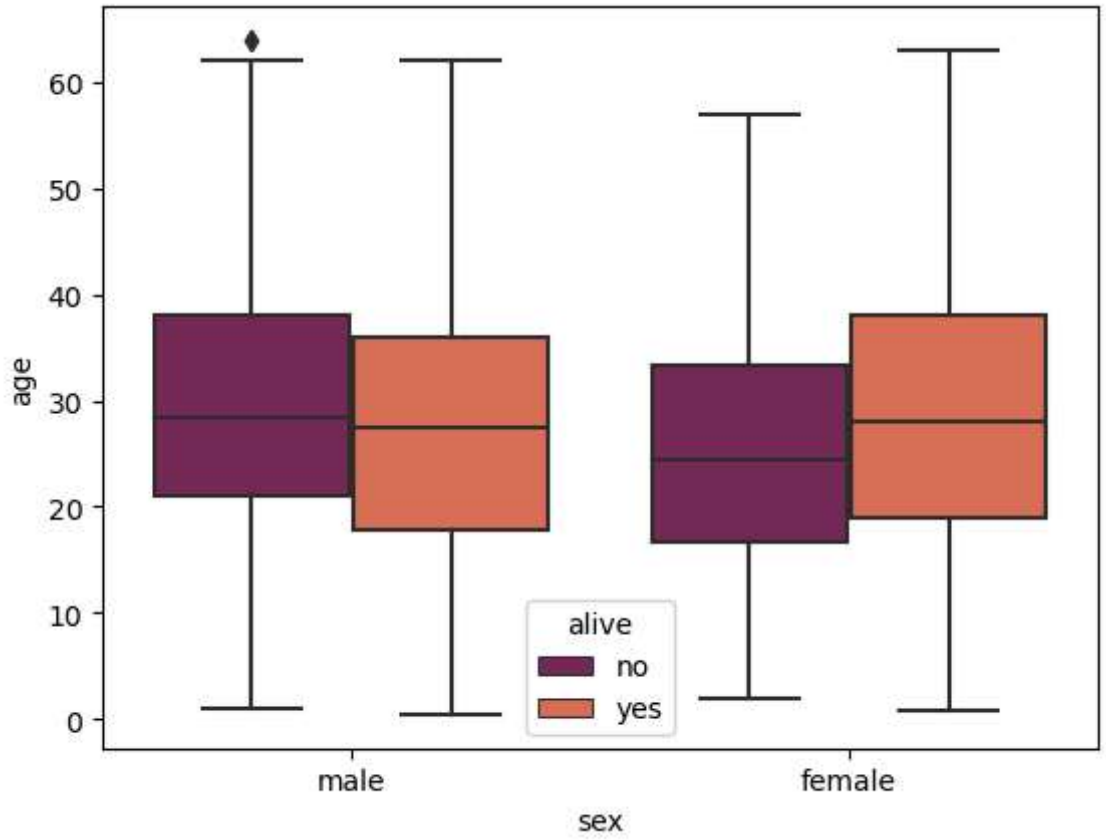
In [16]:
```
sns.boxplot(y="age",x="sex",data=df,palette="rocket")
```

Out[16]:
```
<Axes: xlabel='sex', ylabel='age'>
```



In [17]:
```
sns.boxplot(y="age",x="sex",data=df,palette="rocket",hue="alive")
```

Out[17]:
```
<Axes: xlabel='sex', ylabel='age'>
```



In [18]:
```
# violin plot
df=sns.load_dataset("tips")
df
```

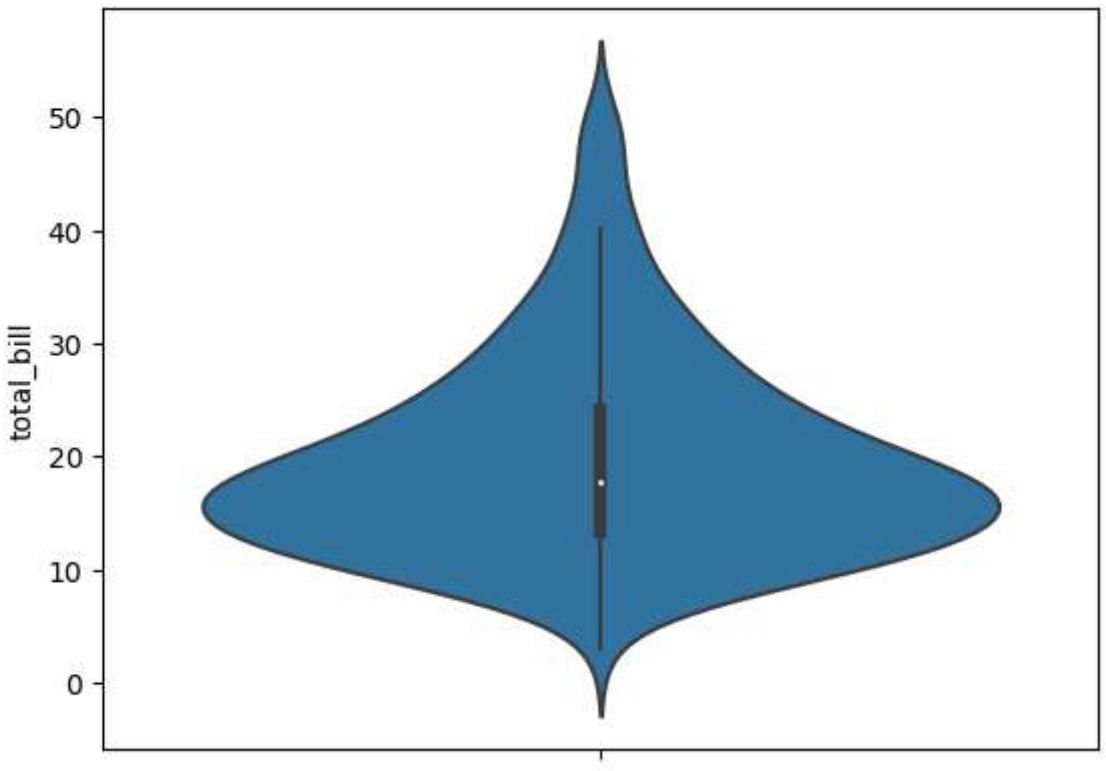
Out[18]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

In [19]:
```
sns.violinplot(y="total_bill",data=df)
```

Out[19]:
```
<Axes: ylabel='total_bill'>
```



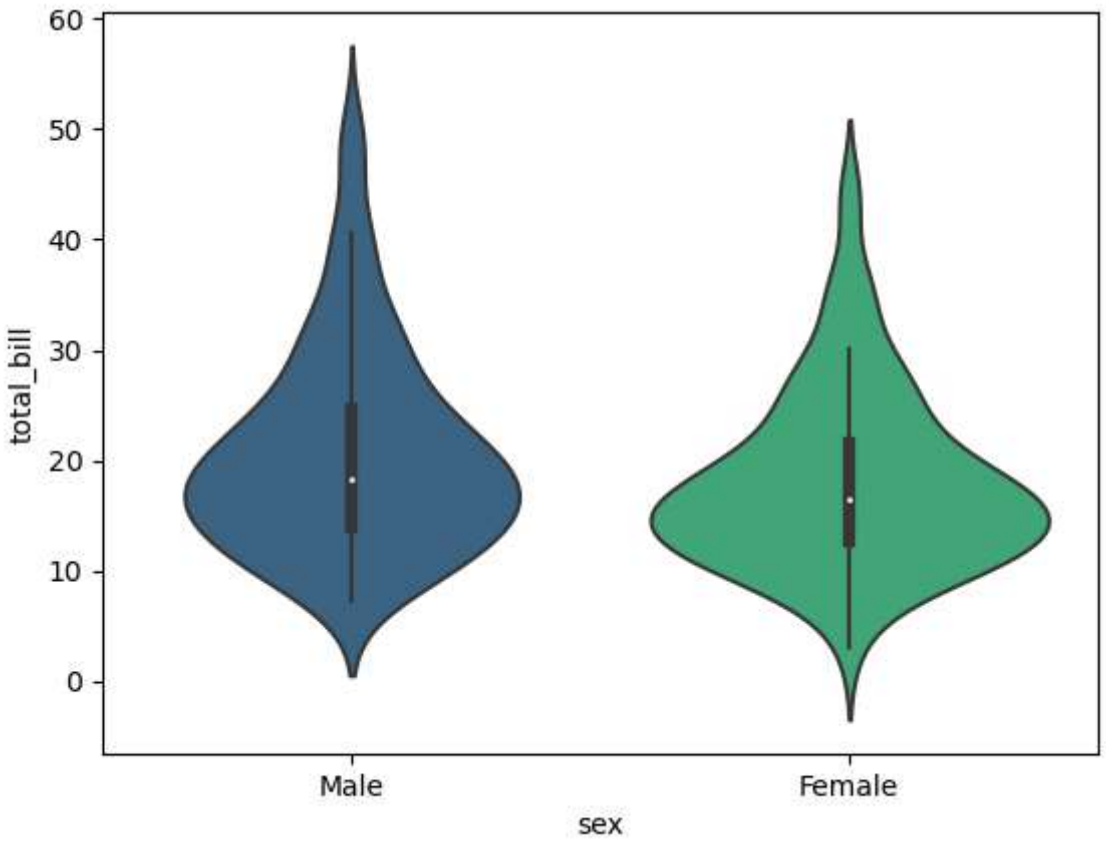
In [20]:
```
df.describe()
```

Out[20]:

	total_bill	tip	size
count	244.000000	244.000000	244.000000
mean	19.785943	2.998279	2.569672
std	8.902412	1.383638	0.951100
min	3.070000	1.000000	1.000000
25%	13.347500	2.000000	2.000000
50%	17.795000	2.900000	2.000000
75%	24.127500	3.562500	3.000000
max	50.810000	10.000000	6.000000

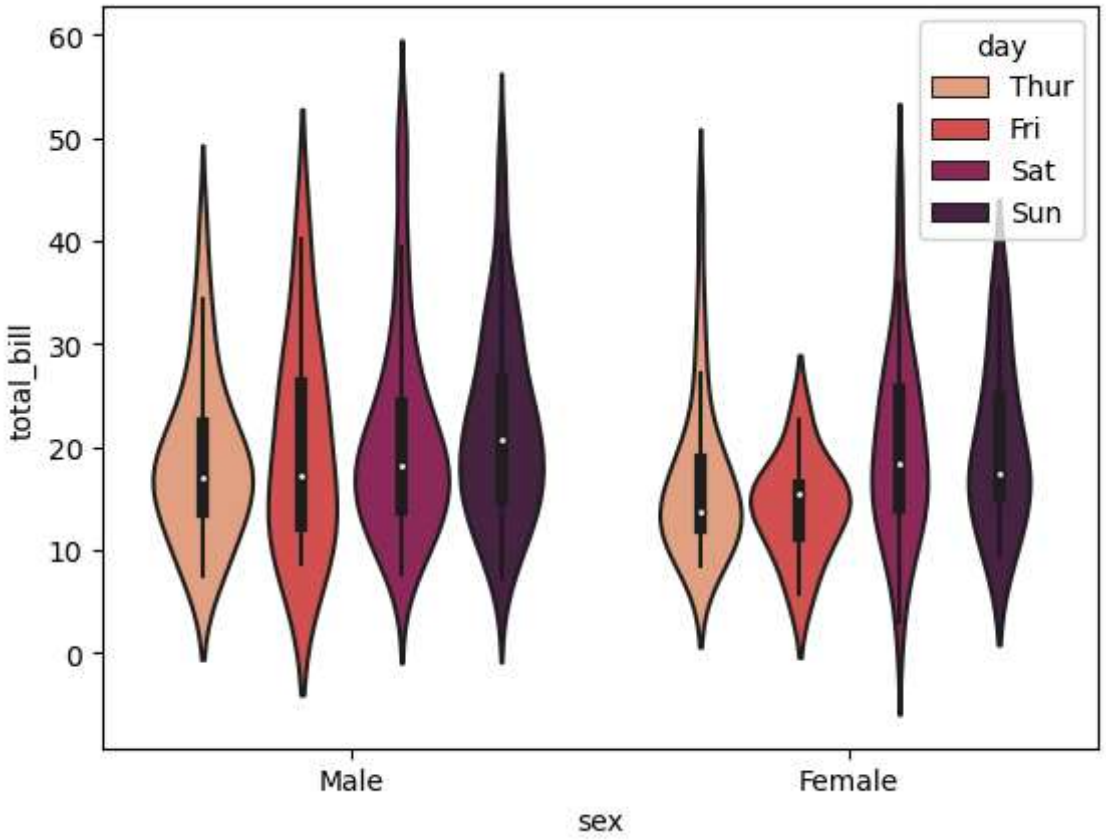
In [22]:
```
sns.violinplot(y="total_bill",x="sex",data=df,palette="viridis")
```

Out[22]:
```
<Axes: xlabel='sex', ylabel='total_bill'>
```



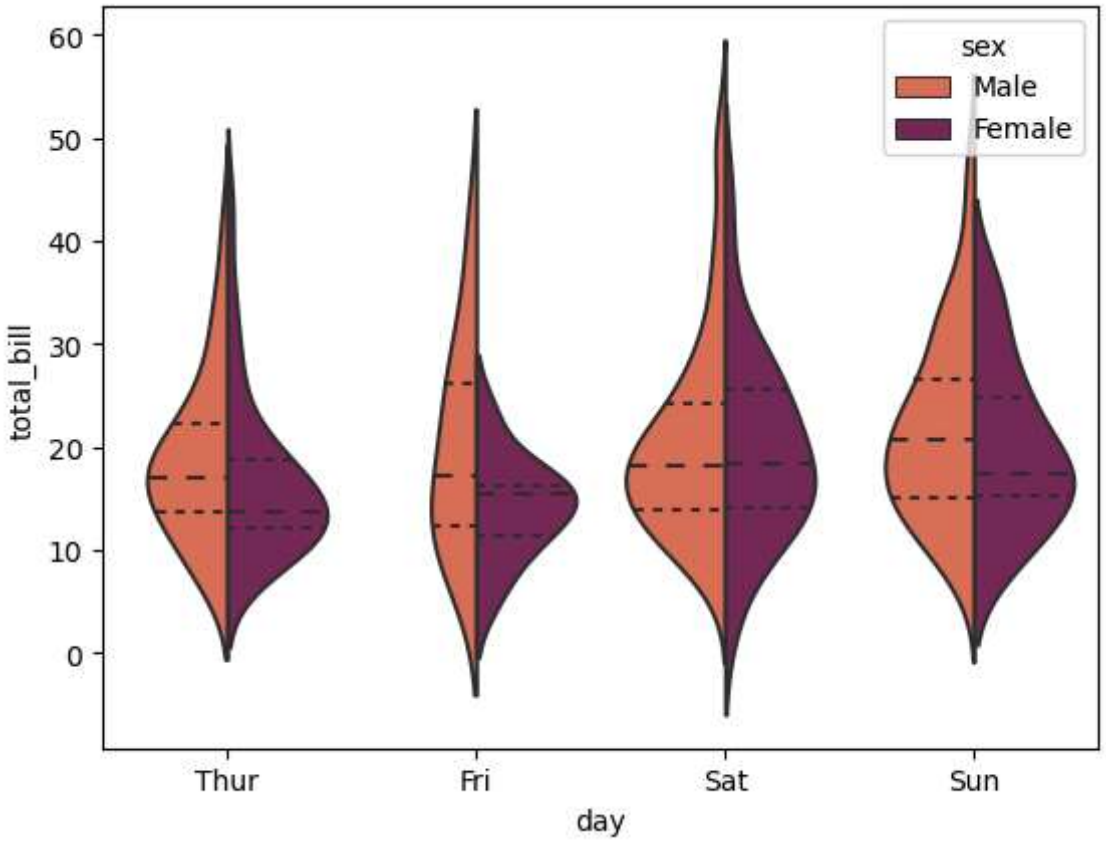
In [24]:
```
sns.violinplot(y="total_bill",x="sex",data=df,palette="rocket_r",hue="day")
```

Out[24]:
```
<Axes: xlabel='sex', ylabel='total_bill'>
```



In [30]:
```
sns.violinplot(y="total_bill",x="day",data=df,palette="rocket_r",split=True,inner='
```

Out[30]:
```
<Axes: xlabel='day', ylabel='total_bill'>
```

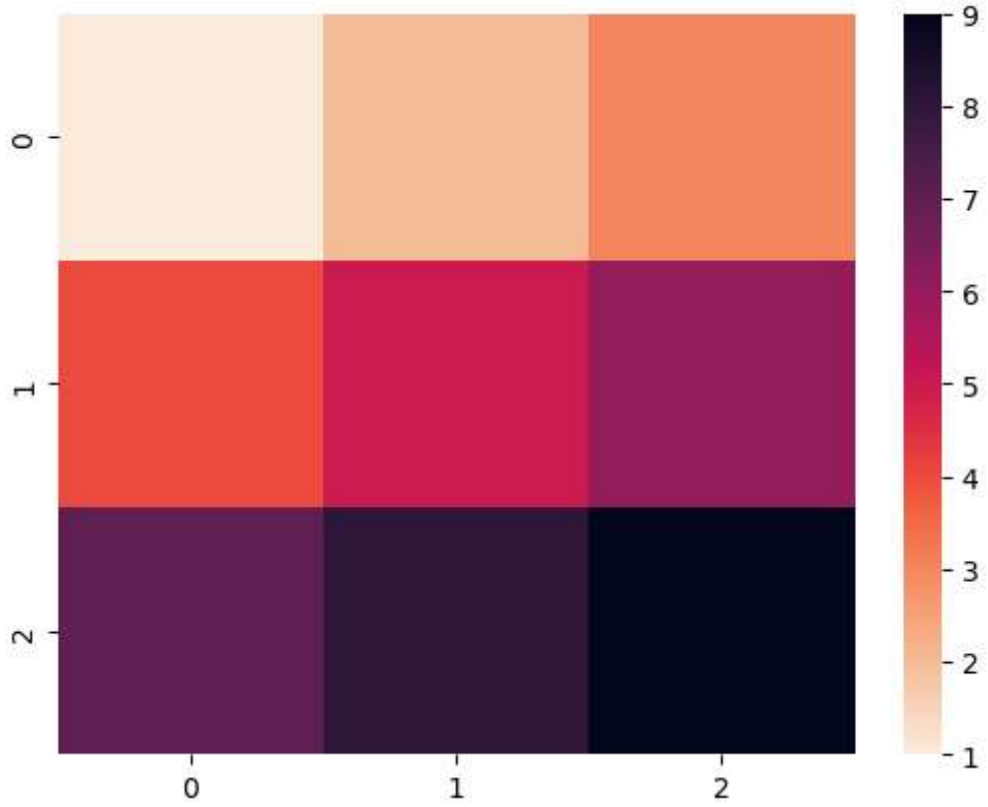


In [32]:
```
# heatmap
arr=np.arange(1,10).reshape(3,3)
arr
```

Out[32]:
```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

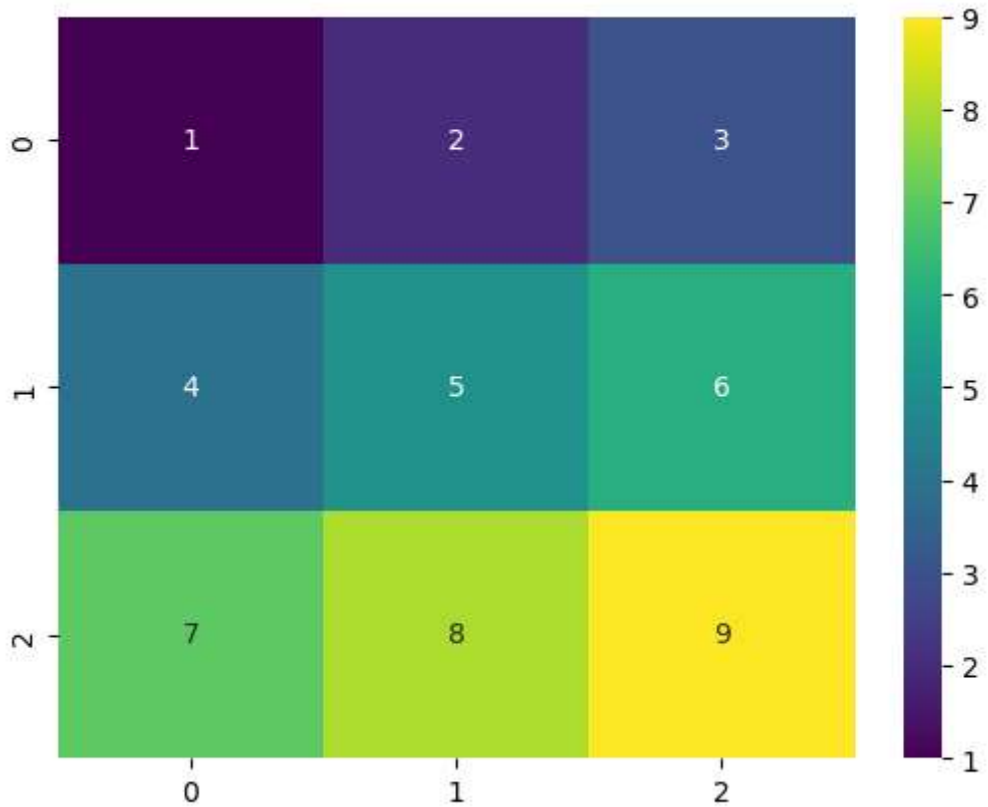
In [34]:
```
sns.heatmap(arr,cmap="rocket_r")
```

Out[34]:
```
<Axes: >
```



In [36]:
```
sns.heatmap(arr,cmap="viridis",annot=True)
```

Out[36]:
```
<Axes: >
```

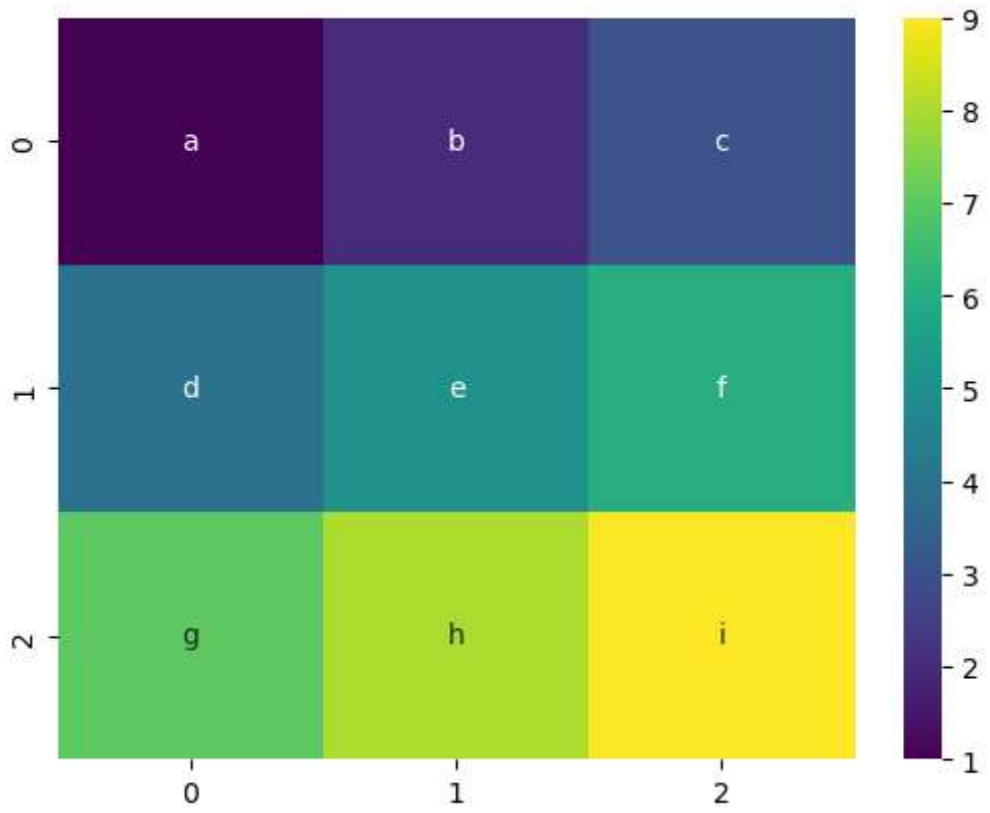


In [37]:
```
val=np.array([["a","b","c"],["d","e","f"],["g","h","i"]])
val
```

Out[37]:
```
array([['a', 'b', 'c'],
       ['d', 'e', 'f'],
       ['g', 'h', 'i']], dtype='<U1')
```

In [40]:
```
sns.heatmap(arr,cmap="viridis",annot=val,fmt="s")
```

Out[40]:
```
<Axes: >
```



In [41]:
```
# project using heatmap
```

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
df=pd.read_csv("globalwarming.csv",index_col=0)
df
```

Out[1]:

Country Name	Country Code	Indicator Name	Indicator Code	2000	2001	2002	2003	
United States	USA	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	20.178751	19.636505	19.613404	19.564105	19.65
United Kingdom	GBR	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	9.199549	9.233175	8.904123	9.053278	8.98
India	IND	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	0.979870	0.971698	0.967381	0.992392	1.02
China	CHN	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	2.696862	2.742121	3.007083	3.524074	4.03
Russian Federation	RUS	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	10.627121	10.669603	10.715901	11.090647	11.12
Australia	AUS	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	17.200610	16.733367	17.370452	16.901959	17.02
France	FRA	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	5.946665	6.153061	6.068664	6.115998	6.12
Germany	DEU	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	10.095640	10.366287	10.058673	9.969355	9.89
Canada	CAN	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	17.367115	16.985030	16.559378	17.461199	17.25
Brazil	BRA	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	1.871118	1.898354	1.844380	1.762482	1.82
Argentina	ARG	CO2 emissions	EN.ATM.CO2E.PC	3.835574	3.568600	3.291548	3.525584	4.06

Country Name	Country Code	Indicator Name	Indicator Code	2000	2001	2002	2003	
		(metric tons per capita)						
Pakistan	PAK	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	0.768458	0.764702	0.788668	0.804959	0.87
Nepal	NPL	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	0.129282	0.135226	0.106877	0.113902	0.1(
Bangladesh	BGD	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	0.211802	0.242020	0.246756	0.256602	0.26
Japan	JPN	CO2 emissions (metric tons per capita)	EN.ATM.CO2E.PC	9.622352	9.464309	9.573130	9.725282	9.9(

In [2]:
```
df.drop(["Country Code","Indicator Name","Indicator Code"],axis=1,inplace=True)
```

In [3]:
```
pd.set_option("display.precision",2)
```

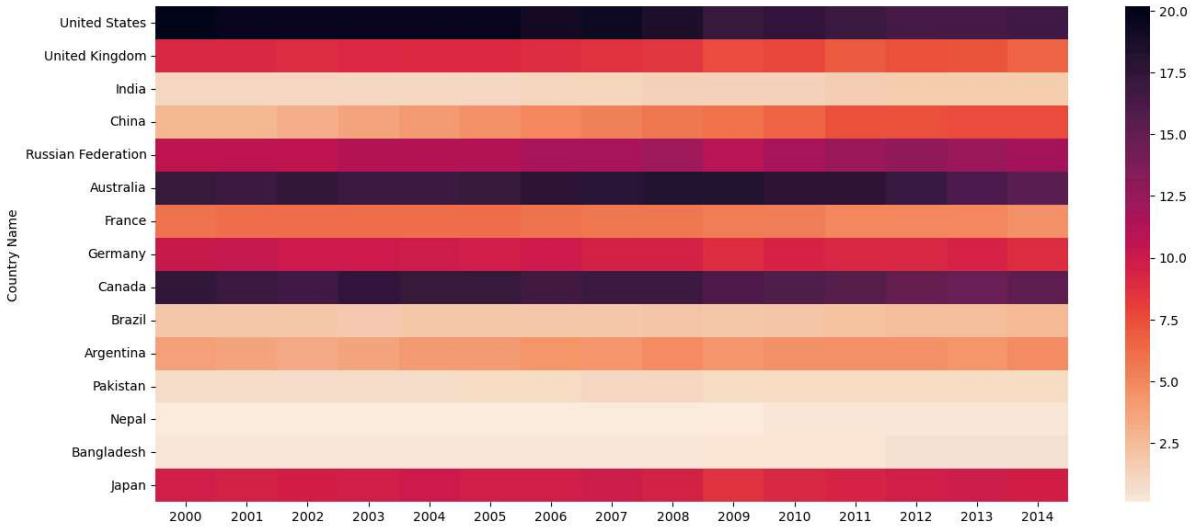
In [4]:
```
df
```

Out[4]:

Country Name	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012
United States	20.18	19.64	19.61	19.56	19.66	19.59	19.09	19.22	18.46	17.16	17.44	16.98	16.31
United Kingdom	9.20	9.23	8.90	9.05	8.99	8.98	8.90	8.62	8.42	7.57	7.86	7.08	7.36
India	0.98	0.97	0.97	0.99	1.03	1.07	1.12	1.19	1.31	1.43	1.40	1.48	1.60
China	2.70	2.74	3.01	3.52	4.04	4.52	4.98	5.33	5.70	6.01	6.56	7.24	7.42
Russian Federation	10.63	10.67	10.72	11.09	11.12	11.25	11.67	11.67	12.01	11.02	11.69	12.33	12.78
Australia	17.20	16.73	17.37	16.90	17.03	17.17	17.65	17.87	18.16	18.20	17.74	17.54	17.07
France	5.95	6.15	6.07	6.12	6.12	6.10	5.91	5.77	5.69	5.44	5.43	5.08	5.08
Germany	10.10	10.37	10.06	9.97	9.90	9.67	9.91	9.49	9.51	8.82	9.28	9.12	9.20
Canada	17.37	16.99	16.56	17.46	17.26	17.25	16.70	16.86	16.88	15.96	15.72	15.64	14.89
Brazil	1.87	1.90	1.84	1.76	1.83	1.86	1.84	1.90	2.01	1.88	2.13	2.21	2.34
Argentina	3.84	3.57	3.29	3.53	4.07	4.14	4.43	4.38	4.68	4.41	4.56	4.60	4.57
Pakistan	0.77	0.76	0.79	0.80	0.87	0.89	0.93	0.99	0.97	0.95	0.95	0.93	0.92
Nepal	0.13	0.14	0.11	0.11	0.11	0.12	0.10	0.10	0.13	0.16	0.19	0.20	0.21
Bangladesh	0.21	0.24	0.25	0.26	0.27	0.28	0.30	0.30	0.33	0.36	0.39	0.41	0.43
Japan	9.62	9.46	9.57	9.73	9.91	9.70	9.63	9.78	9.45	8.62	9.15	9.32	9.64

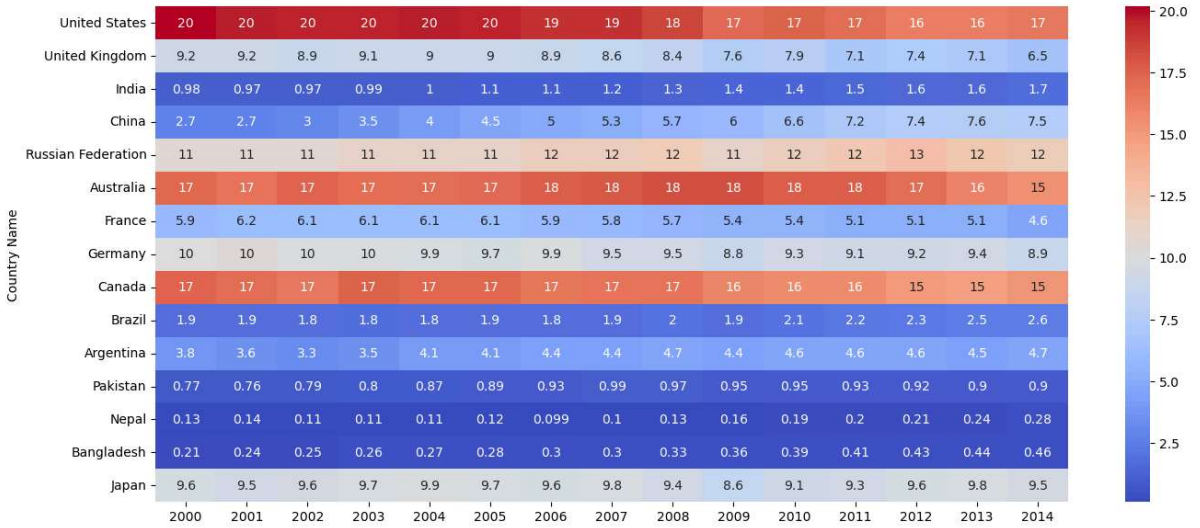
In [5]:
```
plt.figure(figsize=(16,7))
sns.heatmap(df,cmap="rocket_r")
```

Out[5]:
```
<Axes: ylabel='Country Name'>
```



In [6]:
```
plt.figure(figsize=(16,7))
sns.heatmap(df,cmap="coolwarm",annot=True)
```

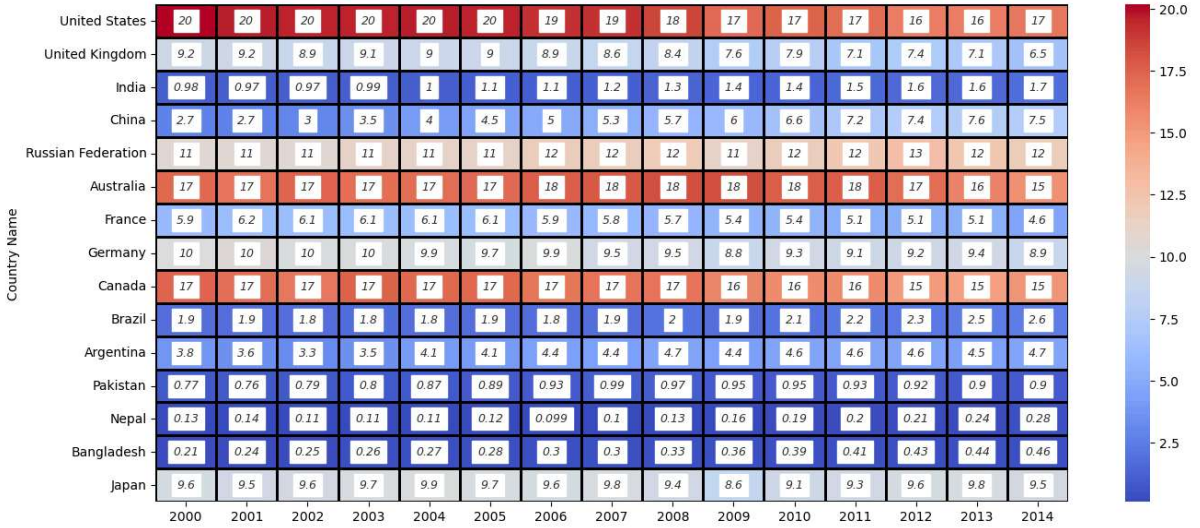
Out[6]:
```
<Axes: ylabel='Country Name'>
```



In [7]:
```
plt.figure(figsize=(16,7))
val={
    "color":"k",
    "fontsize":9,
    "fontstyle":"italic",
    "backgroundcolor":"w",
    "alpha":0.8

}
sns.heatmap(df,cmap="coolwarm",annot=True,annot_kws=val,linewidth=1,linecolor="k")
```

Out[7]: <Axes: ylabel='Country Name'>



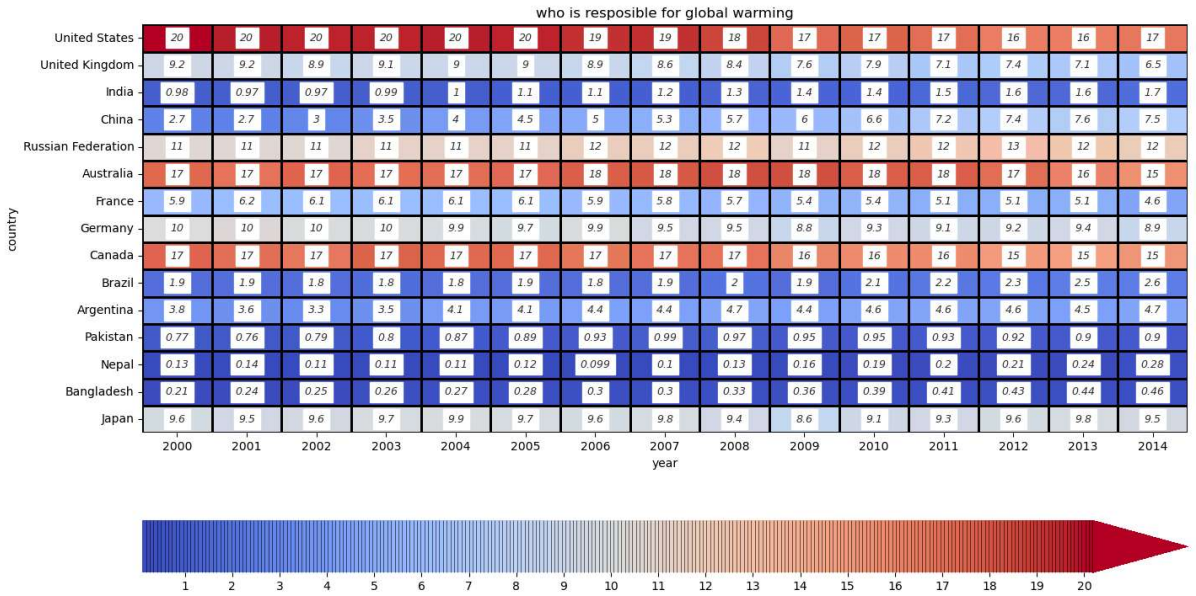
In [18]:
```
plt.figure(figsize=(16,9))
val={
    "color":"k",
    "fontsize":9,
    "fontstyle":"italic",
    "backgroundcolor":"w",
    "alpha":0.8

}

cbar={'orientation':"horizontal",
       "shrink":1,
#         'extend':"both",
#         'extend':None,
       'extend':"max",
#         'extend':None,
```

```
        'extendfrac':0.1,
        'ticks':np.arange(0,22),
        'drawedges':True,


}
ob=sns.heatmap(df,cmap="coolwarm",annot=True,annot_kws=val,linewidth=1,linecolor="k
ob.set(title="who is resposible for global warming",
       xlabel="year",
       ylabel="country")
plt.show()
```



In [19]:
```
# correlation
# 1- perfect +ve correlation
# 0- no correlation
# -1- perfect -ve correlation
```

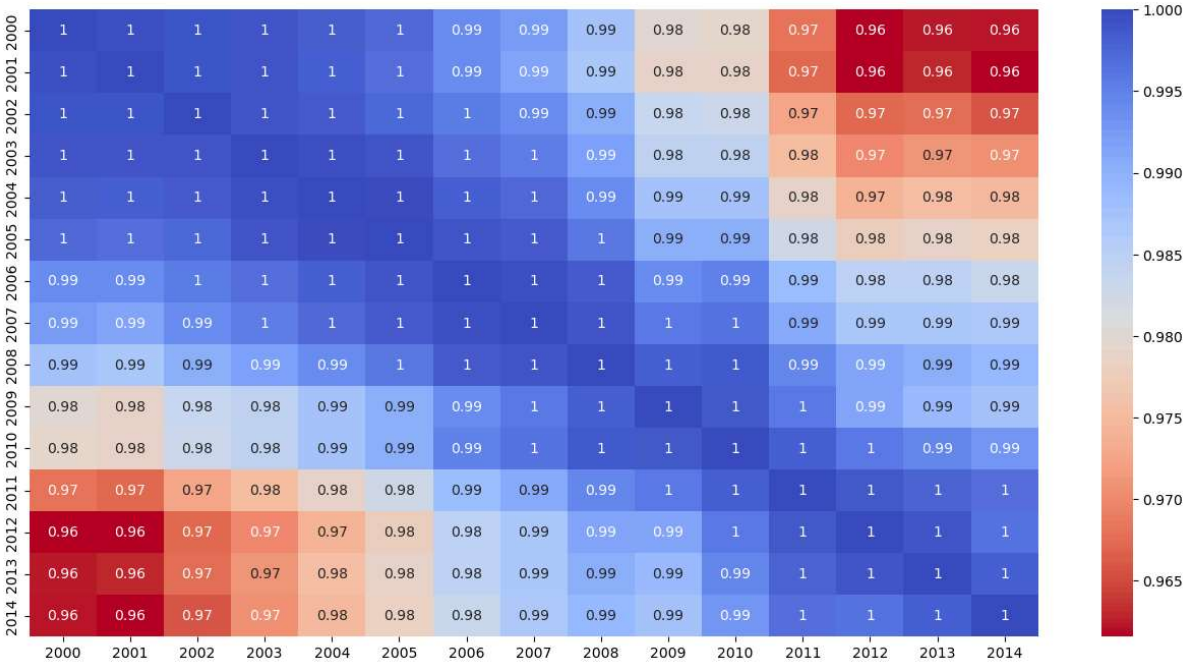
In [20]:
```
df.corr()
```

Out[20]:

	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
2000	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.98	0.98	0.97	0.96	0.96
2001	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.98	0.98	0.97	0.96	0.96
2002	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.98	0.98	0.97	0.97	0.97
2003	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.98	0.98	0.98	0.97	0.97
2004	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.98	0.97	0.98
2005	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.98	0.98	0.98
2006	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.98	0.98
2007	0.99	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99
2008	0.99	0.99	0.99	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.99
2009	0.98	0.98	0.98	0.98	0.99	0.99	0.99	1.00	1.00	1.00	1.00	1.00	0.99	0.99
2010	0.98	0.98	0.98	0.98	0.99	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00	0.99
2011	0.97	0.97	0.97	0.98	0.98	0.98	0.99	0.99	0.99	1.00	1.00	1.00	1.00	1.00
2012	0.96	0.96	0.97	0.97	0.97	0.98	0.98	0.99	0.99	0.99	1.00	1.00	1.00	1.00
2013	0.96	0.96	0.97	0.97	0.98	0.98	0.98	0.99	0.99	0.99	0.99	1.00	1.00	1.00
2014	0.96	0.96	0.97	0.97	0.98	0.98	0.98	0.99	0.99	0.99	0.99	1.00	1.00	1.00

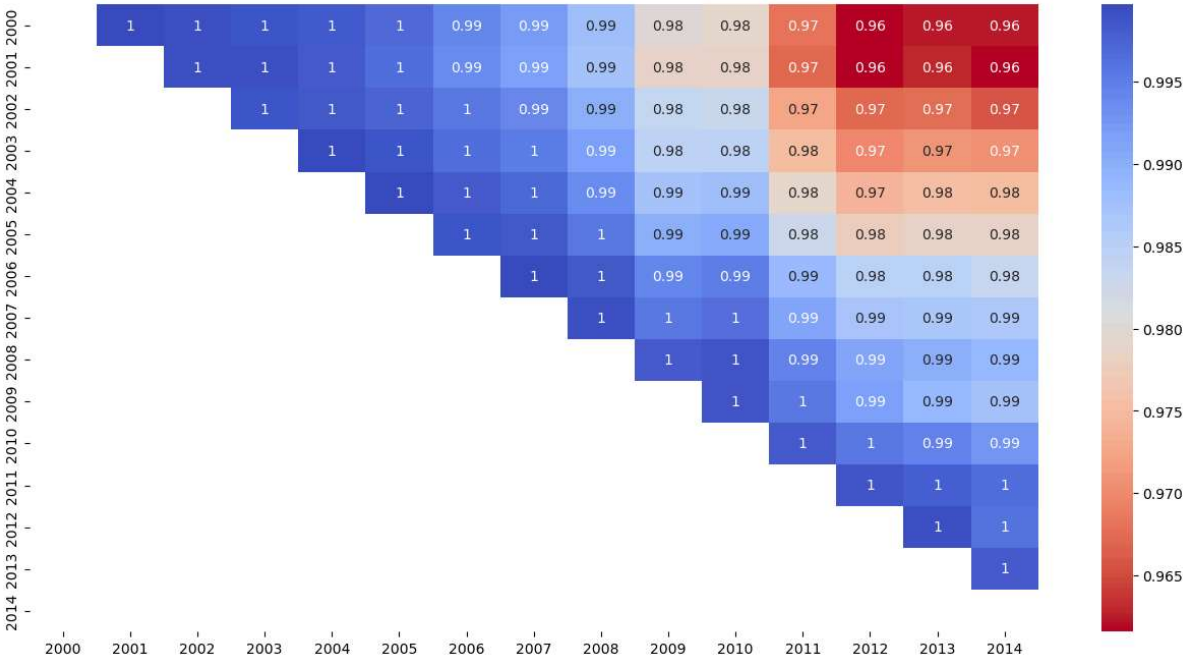
In [25]:
```
plt.figure(figsize=(16,8))
sns.heatmap(df.corr(),cmap="coolwarm_r",annot=True)
```

Out[25]:
```
<Axes: >
```



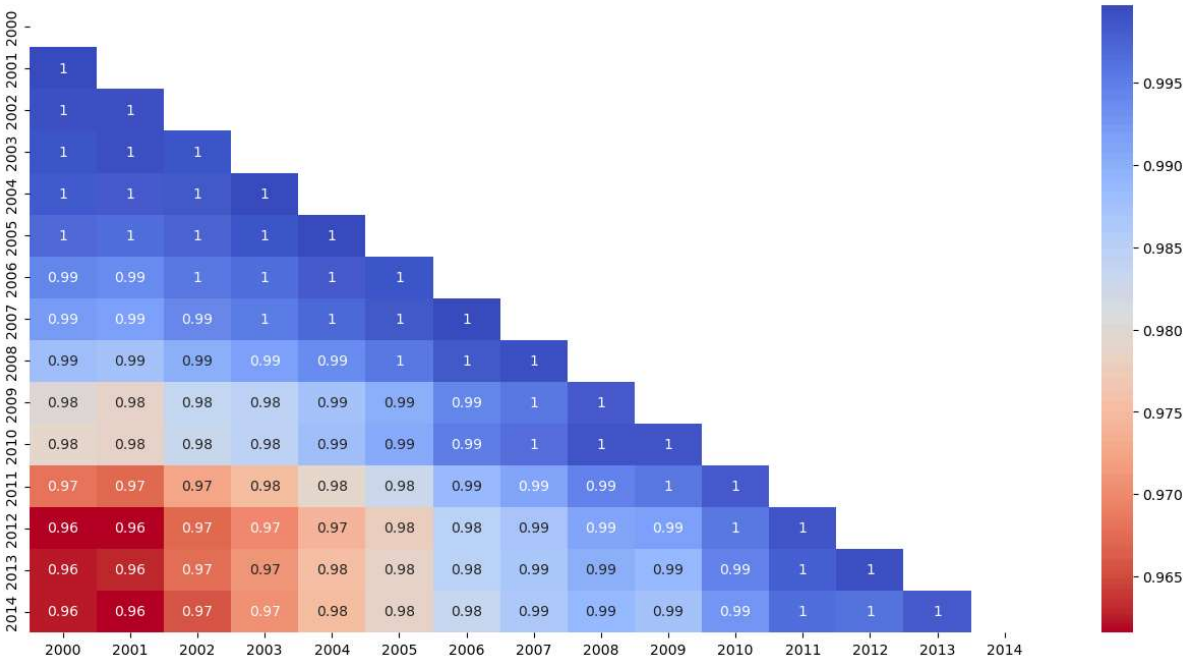
In [26]:
```
plt.figure(figsize=(16,8))
corr_mx=df.corr()
matrix=np.tril(corr_mx)
sns.heatmap(corr_mx,mask=matrix,annot=True,cmap="coolwarm_r")
```

Out[26]:
```
<Axes: >
```



In [27]:
```
plt.figure(figsize=(16,8))
corr_mx=df.corr()
matrix=np.triu(corr_mx)
sns.heatmap(corr_mx,mask=matrix,annot=True,cmap="coolwarm_r")
```

Out[27]: <Axes: >



In [28]:
```
# stats
df=pd.DataFrame()
df
```

Out[28]: —

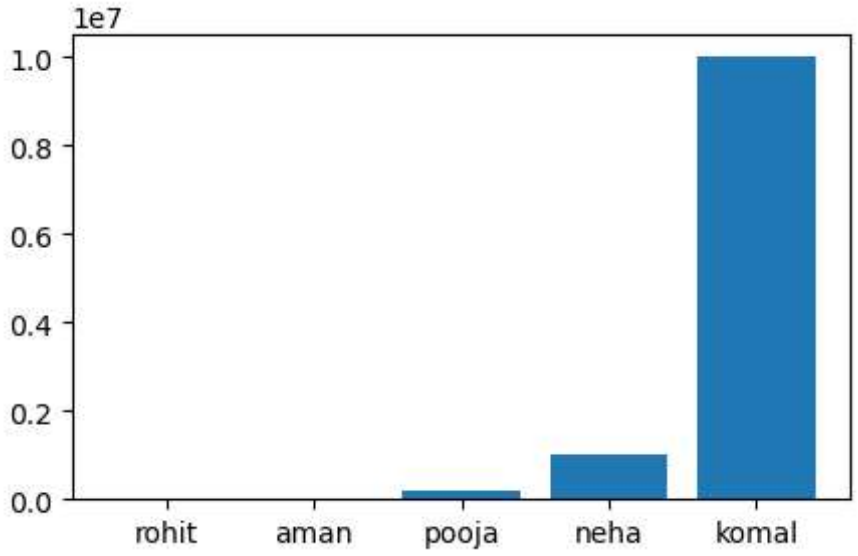
In [29]:
```
df["name"]=["rohit","aman","pooja","neha","komal"]
df["sales"]=[1000,10000,200000,1000000,10000000]
df
```

Out[29]:

	name	sales
0	rohit	1000
1	aman	10000
2	pooja	200000
3	neha	1000000
4	komal	10000000

In [34]:
```
plt.figure(figsize=(5,3))
plt.bar(df.name,df.sales)
```

Out[34]:
```
<BarContainer object of 5 artists>
```



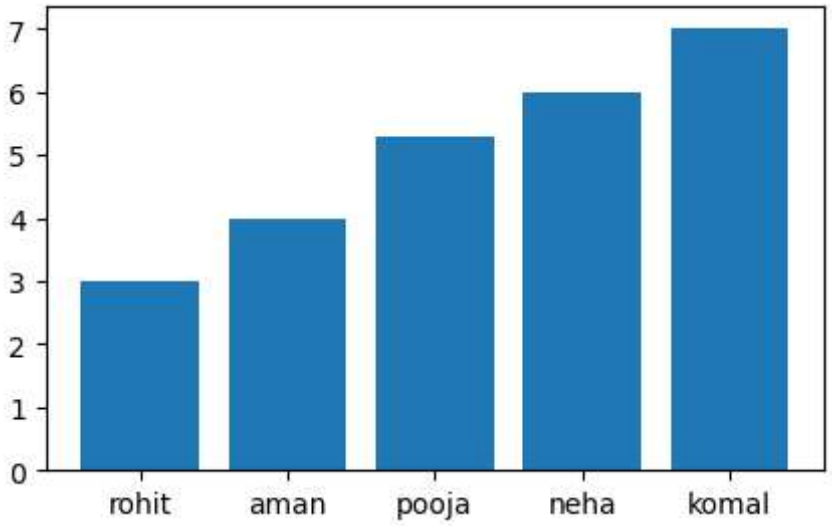
In [35]:
```
# Log function
from math import log10
df["logsales"]=df.sales.apply(lambda x:log10(x))
df
```

Out[35]:

	name	sales	logsales
0	rohit	1000	3.0
1	aman	10000	4.0
2	pooja	200000	5.3
3	neha	1000000	6.0
4	komal	10000000	7.0

In [36]:
```
plt.figure(figsize=(5,3))
plt.bar(df.name,df.logsales)
```

Out[36]:
```
<BarContainer object of 5 artists>
```



In [37]:
```
# std- standard deviation
```

In [38]:
```
df=sns.load_dataset("titanic")
df
```

Out[38]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.25	S	Third	man	True
1	1	1	female	38.0	1	0	71.28	C	First	woman	False
2	1	3	female	26.0	0	0	7.92	S	Third	woman	False
3	1	1	female	35.0	1	0	53.10	S	First	woman	False
4	0	3	male	35.0	0	0	8.05	S	Third	man	True
...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.00	S	Second	man	True
887	1	1	female	19.0	0	0	30.00	S	First	woman	False
888	0	3	female	NaN	1	2	23.45	S	Third	woman	False
889	1	1	male	26.0	0	0	30.00	C	First	man	True
890	0	3	male	32.0	0	0	7.75	Q	Third	man	True

891 rows × 15 columns

In [39]:
```
df=df[["sex","age"]]
df
```

Out[39]:

	sex	age
0	male	22.0
1	female	38.0
2	female	26.0
3	female	35.0
4	male	35.0
...	...	...
886	male	27.0
887	female	19.0
888	female	NaN
889	male	26.0
890	male	32.0

891 rows × 2 columns

In [40]:
```
df.isnull().sum()
```

Out[40]:
```
sex      0
age    177
dtype: int64
```

In [41]:
```
mean=df.age.mean()
mean
```

Out[41]:
```
29.69911764705882
```

In [42]:
```
df.age.fillna(mean,inplace=True)
```

```
C:\Users\Admin\AppData\Local\Temp\ipykernel_9340\2869447227.py:1: SettingWithCopyW
arning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df.age.fillna(mean,inplace=True)
```

In [43]:
```
df
```

Out[43]:

	sex	age
0	male	22.0
1	female	38.0
2	female	26.0
3	female	35.0
4	male	35.0
...	...	...
886	male	27.0
887	female	19.0
888	female	29.7
889	male	26.0
890	male	32.0

891 rows × 2 columns

In [44]:
```
df.isnull().sum()
```

Out[44]:
```
sex    0
age    0
dtype: int64
```

In [45]:
```
std=df.age.std()
std
```

Out[45]:
```
13.002015226002884
```

In [46]:
```
mean=df.age.mean()
mean
```

Out[46]:
```
29.69911764705882
```

In [47]:
```
max=mean+3*std
max
```

Out[47]:
```
68.70516332506747
```

In [48]:
```
min=mean-3*std
min
```

Out[48]:
```
-9.306928030949834
```

In [50]:
```
df[~((df.age<=max) & (df.age>=min))]
```

Out[50]:

	sex	age
96	male	71.0
116	male	70.5
493	male	71.0
630	male	80.0
672	male	70.0
745	male	70.0
851	male	74.0

In [51]:
```
df1=df[((df.age<=max) & (df.age>=min))]
df1
```

Out[51]:

	sex	age
0	male	22.0
1	female	38.0
2	female	26.0
3	female	35.0
4	male	35.0
...	...	...
886	male	27.0
887	female	19.0
888	female	29.7
889	male	26.0
890	male	32.0

884 rows × 2 columns

In [ ]: