Consider the following Python dictionary data and Python list labels:

data = {'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spoonbills', 'Cranes', 'plovers', 'Cranes', 'spoonbills', 'spoonbills'], 'age': [3.5, 4, 1.5, np.nan, 6, 3, 5.5, np.nan, 8, 4], 'visits': [2, 4, 3, 4, 3, 4, 2, 2, 3, 2], 'priority': ['yes', 'yes', 'no', 'yes', 'no', 'no', 'no', 'yes', 'no', 'no']}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

1. Create a DataFrame birds from this dictionary data which has the index labels.

In [112]:
```
import pandas as pd
import numpy as np
data = {'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spoonbills', 'Cra
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
df = pd.DataFrame(data=data,index=labels)
print(df)
```

```
        birds  age  visits priority
a      Cranes  3.5       2      yes
b      Cranes  4.0       4      yes
c     plovers  1.5       3       no
d  spoonbills  NaN       4      yes
e  spoonbills  6.0       3       no
f      Cranes  3.0       4       no
g     plovers  5.5       2       no
h      Cranes  NaN       2      yes
i  spoonbills  8.0       3       no
j  spoonbills  4.0       2       no
```

2. Display a summary of the basic information about birds DataFrame and its data.

In [113]:
```
df.describe()
```

Out[113]:

	age	visits
count	8.000000	10.000000
mean	4.437500	2.900000
std	2.007797	0.875595
min	1.500000	2.000000
25%	3.375000	2.000000
50%	4.000000	3.000000
75%	5.625000	3.750000
max	8.000000	4.000000

3. Print the first 2 rows of the birds dataframe

In [114]:
```
df.head(2)
```

Out[114]:

	birds	age	visits	priority
a	Cranes	3.5	2	yes
b	Cranes	4.0	4	yes

4. Print all the rows with only 'birds' and 'age' columns from the dataframe

In [115]:
```
df[['birds','age']]
```

Out[115]:

	birds	age
a	Cranes	3.5
b	Cranes	4.0
c	plovers	1.5
d	spoonbills	NaN
e	spoonbills	6.0
f	Cranes	3.0
g	plovers	5.5
h	Cranes	NaN
i	spoonbills	8.0
j	spoonbills	4.0

5. select [2, 3, 7] rows and in columns ['birds', 'age', 'visits']

In [116]:
```
df.iloc[[1,2,6], :3]
```

Out[116]:

	birds	age	visits
b	Cranes	4.0	4
c	plovers	1.5	3
g	plovers	5.5	2

6. select the rows where the number of visits is less than 4

In [117]:
```
df[df['visits'] == 4]
```

Out[117]:

	birds	age	visits	priority
b	Cranes	4.0	4	yes
d	spoonbills	NaN	4	yes
f	Cranes	3.0	4	no

7. select the rows with columns ['birds', 'visits'] where the age is missing i.e NaN

In [118]:
```
df.loc[df['age'].isnull(), ['birds','visits']]
```

Out[118]:

	birds	visits
d	spoonbills	4
h	Cranes	2

8. Select the rows where the birds is a Cranes and the age is less than 4

In [119]:
```
df.loc[(df['birds']=='Cranes') & (df['age'] < 4),:]
```

Out[119]:

	birds	age	visits	priority
a	Cranes	3.5	2	yes
f	Cranes	3.0	4	no

9. Select the rows the age is between 2 and 4(inclusive)

In [120]:
```
df.loc[(df['age'] > 2) & (df['age']<=4), :]
```

Out[120]:

	birds	age	visits	priority
a	Cranes	3.5	2	yes
b	Cranes	4.0	4	yes
f	Cranes	3.0	4	no
j	spoonbills	4.0	2	no

10. Find the total number of visits of the bird Cranes

In [121]:
```
df.loc[df['birds']=='Cranes', ['visits']].values.sum()
```

Out[121]:
```
12
```

11. Calculate the mean age for each different birds in dataframe.

In [122]:
```
df.groupby('birds').mean().age
```

Out[122]:
```
birds
Cranes        3.5
plovers       3.5
spoonbills    6.0
Name: age, dtype: float64
```

12. Append a new row 'k' to dataframe with your choice of values for each column. Then delete that row to return the original DataFrame.

In [123]:
```
df2 = pd.DataFrame([['flamingo',4,2,'yes']],columns=['birds','age','visits','prio
df = df.append(df2)
print(df)
df = df.drop(['k'])
print(df)
```

```
        birds  age  visits priority
a      Cranes  3.5       2      yes
b      Cranes  4.0       4      yes
c     plovers  1.5       3       no
d  spoonbills  NaN       4      yes
e  spoonbills  6.0       3       no
f      Cranes  3.0       4       no
g     plovers  5.5       2       no
h      Cranes  NaN       2      yes
i  spoonbills  8.0       3       no
j  spoonbills  4.0       2       no
k    flamingo  4.0       2      yes
        birds  age  visits priority
a      Cranes  3.5       2      yes
b      Cranes  4.0       4      yes
c     plovers  1.5       3       no
d  spoonbills  NaN       4      yes
e  spoonbills  6.0       3       no
f      Cranes  3.0       4       no
g     plovers  5.5       2       no
h      Cranes  NaN       2      yes
i  spoonbills  8.0       3       no
j  spoonbills  4.0       2       no
```

13. Find the number of each type of birds in dataframe (Counts)

In [124]:
```
df.birds.value_counts()
```

Out[124]:
```
Cranes        4
spoonbills    4
plovers       2
Name: birds, dtype: int64
```

14. Sort dataframe (birds) first by the values in the 'age' in decending order, then by the value in the 'visits' column in ascending order.

In [125]:
```
print(df.sort_values(['age'],ascending=False).drop(['visits','priority'],axis=1))
print(df.sort_values(['visits'],ascending=True).drop(['age','priority'],axis=1))
```

```
        birds  age
i  spoonbills  8.0
e  spoonbills  6.0
g     plovers  5.5
b      Cranes  4.0
j  spoonbills  4.0
a      Cranes  3.5
f      Cranes  3.0
c     plovers  1.5
d  spoonbills  NaN
h      Cranes  NaN
        birds  visits
a      Cranes       2
g     plovers       2
h      Cranes       2
j  spoonbills       2
c     plovers       3
e  spoonbills       3
i  spoonbills       3
b      Cranes       4
d  spoonbills       4
f      Cranes       4
```

15. Replace the priority column values with'yes' should be 1 and 'no' should be 0

In [126]:
```
df['priority'] = df['priority'].map({'yes':1,'no':0})
print(df)
```

```
        birds  age  visits  priority
a      Cranes  3.5       2         1
b      Cranes  4.0       4         1
c     plovers  1.5       3         0
d  spoonbills  NaN       4         1
e  spoonbills  6.0       3         0
f      Cranes  3.0       4         0
g     plovers  5.5       2         0
h      Cranes  NaN       2         1
i  spoonbills  8.0       3         0
j  spoonbills  4.0       2         0
```

16. In the 'birds' column, change the 'Cranes' entries to 'trumpeters'.

In [127]:
```
df.birds.replace('Cranes','trumpeters',inplace=True)
print(df)
```

```
        birds  age  visits  priority
a  trumpeters  3.5       2         1
b  trumpeters  4.0       4         1
c     plovers  1.5       3         0
d  spoonbills  NaN       4         1
e  spoonbills  6.0       3         0
f  trumpeters  3.0       4         0
g     plovers  5.5       2         0
h  trumpeters  NaN       2         1
i  spoonbills  8.0       3         0
j  spoonbills  4.0       2         0
```