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 [2]:
```
import pandas as pd,numpy as np
birds=pd.DataFrame({"labels":['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j'
],
                    "birds":['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spo
onbills', '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']
                   })
birds.set_index('labels',inplace=True)
birds
```

Out[2]:

labels	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 [15]:
```
birds=pd.DataFrame({"labels":['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j'
],
                    "birds":['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spo
onbills', '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']
                   })
birds.describe()
```

Out[15]:

	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 [16]:
```
birds.head(2)
```

Out[16]:

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

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

In [22]:
```
birds[['birds','age']]
```

Out[22]:

	birds	age
0	Cranes	3.5
1	Cranes	4.0
2	plovers	1.5
3	spoonbills	NaN
4	spoonbills	6.0
5	Cranes	3.0
6	plovers	5.5
7	Cranes	NaN
8	spoonbills	8.0
9	spoonbills	4.0

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

In [43]:
```
birds[['birds','age','visits']].iloc[[2,3,7]]
```

Out[43]:

	birds	age	visits
2	plovers	1.5	3
3	spoonbills	NaN	4
7	Cranes	NaN	2

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

In [26]:
```
birds[birds['visits']<4]
```

Out[26]:

	labels	birds	age	visits	priority
0	a	Cranes	3.5	2	yes
2	c	plovers	1.5	3	no
4	e	spoonbills	6.0	3	no
6	g	plovers	5.5	2	no
7	h	Cranes	NaN	2	yes
8	i	spoonbills	8.0	3	no
9	j	spoonbills	4.0	2	no

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

In [29]:
```
birds[['birds','visits']][birds.age=='NaN']
```

Out[29]:

	birds	visits

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

In [57]:
```
birds[(birds.age<4)&(birds['birds']=='Cranes')]
```

Out[57]:

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

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

In [62]:
```
birds[(birds.age>=2)&(birds.age<=4)]
```

Out[62]:

	labels	birds	age	visits	priority
0	a	Cranes	3.5	2	yes
1	b	Cranes	4.0	4	yes
5	f	Cranes	3.0	4	no
9	j	spoonbills	4.0	2	no

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

In [10]:
```
g=birds.groupby('birds')
Cranes_df=g.get_group('Cranes')
Cranes_df['visits'].sum()
```

Out[10]:
```
12
```

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

In [12]:
```
g['age'].mean()
```

Out[12]:
```
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 [18]:
```
birds1=birds.append({'labels':'k','birds':'plovers','age':4.5,'visits':3,'priority':'no'},ignore_index=True)
birds1.drop(birds1.index[-1])
```

Out[18]:

	labels	birds	age	visits	priority
0	a	Cranes	3.5	2	yes
1	b	Cranes	4.0	4	yes
2	c	plovers	1.5	3	no
3	d	spoonbills	NaN	4	yes
4	e	spoonbills	6.0	3	no
5	f	Cranes	3.0	4	no
6	g	plovers	5.5	2	no
7	h	Cranes	NaN	2	yes
8	i	spoonbills	8.0	3	no
9	j	spoonbills	4.0	2	no

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

In [13]:
```
birds['birds'].value_counts()
```

Out[13]:
```
spoonbills    4
Cranes        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 [24]:
```
k=birds.sort_values(by='age',ascending=False)
k.sort_values(by='visits',ascending=True)
```

Out[24]:

	labels	birds	age	visits	priority
6	g	plovers	5.5	2	no
9	j	spoonbills	4.0	2	no
0	a	Cranes	3.5	2	yes
7	h	Cranes	NaN	2	yes
8	i	spoonbills	8.0	3	no
4	e	spoonbills	6.0	3	no
2	c	plovers	1.5	3	no
1	b	Cranes	4.0	4	yes
5	f	Cranes	3.0	4	no
3	d	spoonbills	NaN	4	yes

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

In [25]:
```
birds.replace({'yes':1,'no':0})
```

Out[25]:

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

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

In [26]:
```
birds.replace({'birds':'Cranes'},'trumpeters')
```

Out[26]:

	labels	birds	age	visits	priority
0	a	trumpeters	3.5	2	yes
1	b	trumpeters	4.0	4	yes
2	c	plovers	1.5	3	no
3	d	spoonbills	NaN	4	yes
4	e	spoonbills	6.0	3	no
5	f	trumpeters	3.0	4	no
6	g	plovers	5.5	2	no
7	h	trumpeters	NaN	2	yes
8	i	spoonbills	8.0	3	no
9	j	spoonbills	4.0	2	no

In [ ]:
```

```