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']

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

data = {'birds': ['Cranes', 'Cranes', 'plovers', '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'
       }
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]:
```
df = pd.DataFrame(data, index=labels)
df.head()
```

Out[2]:

	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

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

In [3]:
```
df.info()
print('<<<------------------------>>>')
df.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 10 entries, a to j
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   birds     10 non-null     object
 1   age       8 non-null      float64
 2   visits    10 non-null     int64
 3   priority  10 non-null     object
dtypes: float64(1), int64(1), object(2)
memory usage: 400.0+ bytes
<<<------------------------>>>
```

Out[3]:

	age	visits
count	8.000000	10.000000
mean	4.437500	2.900000
std	2.007797	0.875595

	age	visits
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 [4]:
```
df.head(2)
```

Out[4]:

	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 [5]:
```
df[['birds', 'age']]
```

Out[5]:

	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 [6]:
```
df[['birds', 'age', 'visits']].iloc[[2,3,7]]
```

Out[6]:

	birds	age	visits
c	plovers	1.5	3
d	spoonbills	NaN	4
h	Cranes	NaN	2

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

In [7]:
```
df[df['visits'] < 4]
```

Out[7]:

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

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

In [8]:
```
df[pd.isna(df['age'])][['birds', 'visits']]
```

Out[8]:

	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 [9]:
```
df[(df['age'] < 4) & (df['birds'] == 'Cranes')]
```

Out[9]:

	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 [10]:
```
df[df['age'].between(2,4)]
```

Out[10]:

	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 [11]:
```
df[df['birds']=='Cranes']['visits'].sum()
```

Out[11]:
```
12
```

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

In [12]:
```
print(df.groupby(by='birds')['age'].mean())
```

```
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 [13]:
```
df = df.append({'birds': 'Flutter', 'age': 4.5, 'visits':2, 'priority':'yes'}
df = df[:-1]
df
```

Out[13]:

	birds	age	visits	priority
0	Cranes	3.5	2	yes
1	Cranes	4.0	4	yes
2	plovers	1.5	3	no
3	spoonbills	NaN	4	yes
4	spoonbills	6.0	3	no
5	Cranes	3.0	4	no
6	plovers	5.5	2	no
7	Cranes	NaN	2	yes
8	spoonbills	8.0	3	no
9	spoonbills	4.0	2	no

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

In [14]:
```
df.groupby('birds')['birds'].count()
```

Out[14]:
```
birds
Cranes         4
plovers        2
spoonbills     4
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 [15]:
```
df.sort_values(['age', 'visits'], ascending=[False, True])['birds']
```

Out[15]:
```
8    spoonbills
4    spoonbills
6       plovers
9    spoonbills
1        Cranes
0        Cranes
5        Cranes
2       plovers
7        Cranes
3    spoonbills
Name: birds, dtype: object
```

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

In [16]:
```
mask = df['priority'] == 'yes'
```

```
df['priority'] = np.where(mask, 1, 0)
df
```

Out[16]:

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

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

In [17]:

```
df.loc[df['birds'] == 'Cranes', 'birds'] = 'trumpeters'
df
```

Out[17]:

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