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.

Importing pandas and numpy

In [3]:

```
import pandas as pd
import numpy as np
```

In [4]:

```
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 [5]:

```
birds=pd.DataFrame(data, index=labels)
```

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

In [6]:

```
birds.describe()
```

Out[6]:

	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 [7]:

```
birds.head(2)
```

Out[7]:

	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 [8]:

```
birds[['birds','age']]
```

Out[8]:

	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 [9]:

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

Out[9]:

	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 [10]:

```
birds[birds['visits']<4]
```

Out[10]:

	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 [11]:

```
birds[birds['visits']<4]
```

Out[11]:

	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

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

In [12]:

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

Out[12]:

	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

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

In [13]:

```
birds[birds['age'].between(2,4)]
```

Out[13]:

	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 [14]:

```
birds[(birds['birds']=='Cranes') & birds['visits']].sum()
```

Out[14]:

```
birds       0.0
age         0.0
visits      0.0
priority    0.0
dtype: float64
```

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

In [15]:

```
birds.groupby('birds')['age'].mean()
```

Out[15]:

```
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 [19]:

```
birds.loc['k']=['new',10.0,10.1,'yes']
birds
```

Out[19]:

	birds	age	visits	priority
a	Cranes	3.5	2.0	yes
b	Cranes	4.0	4.0	yes
c	plovers	1.5	3.0	no
d	spoonbills	NaN	4.0	yes
e	spoonbills	6.0	3.0	no
f	Cranes	3.0	4.0	no
g	plovers	5.5	2.0	no
h	Cranes	NaN	2.0	yes
i	spoonbills	8.0	3.0	no
j	spoonbills	4.0	2.0	no
k	new	10.0	10.1	yes

In [20]:

```
birds=birds.drop('k')
birds
```

Out[20]:

	birds	age	visits	priority
a	Cranes	3.5	2.0	yes
b	Cranes	4.0	4.0	yes
c	plovers	1.5	3.0	no
d	spoonbills	NaN	4.0	yes
e	spoonbills	6.0	3.0	no
f	Cranes	3.0	4.0	no
g	plovers	5.5	2.0	no
h	Cranes	NaN	2.0	yes
i	spoonbills	8.0	3.0	no
j	spoonbills	4.0	2.0	no

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

In [21]:

```
birds.groupby('birds').count()
```

Out[21]:

birds	age	visits	priority
Cranes	3	4	4
plovers	2	2	2
spoonbills	3	4	4

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 [22]:

```
birds.sort_values(by=['age','visits'], ascending=[False,True])
```

Out[22]:

	birds	age	visits	priority
i	spoonbills	8.0	3.0	no
e	spoonbills	6.0	3.0	no
g	plovers	5.5	2.0	no
j	spoonbills	4.0	2.0	no
b	Cranes	4.0	4.0	yes
a	Cranes	3.5	2.0	yes
f	Cranes	3.0	4.0	no
c	plovers	1.5	3.0	no
h	Cranes	NaN	2.0	yes
d	spoonbills	NaN	4.0	yes

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

In [23]:

```
birds['priority'] = birds['priority'].map({'yes':1,'no':0})
birds
```

Out[23]:

	birds	age	visits	priority
a	Cranes	3.5	2.0	1
b	Cranes	4.0	4.0	1
c	plovers	1.5	3.0	0
d	spoonbills	NaN	4.0	1
e	spoonbills	6.0	3.0	0
f	Cranes	3.0	4.0	0
g	plovers	5.5	2.0	0
h	Cranes	NaN	2.0	1
i	spoonbills	8.0	3.0	0
j	spoonbills	4.0	2.0	0

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

In [24]:

```
birds['birds'] = birds['birds'].replace('Cranes','trumpeters')
birds
```

Out[24]:

	birds	age	visits	priority
a	trumpeters	3.5	2.0	1
b	trumpeters	4.0	4.0	1
c	plovers	1.5	3.0	0
d	spoonbills	NaN	4.0	1
e	spoonbills	6.0	3.0	0
f	trumpeters	3.0	4.0	0
g	plovers	5.5	2.0	0
h	trumpeters	NaN	2.0	1
i	spoonbills	8.0	3.0	0
j	spoonbills	4.0	2.0	0

In [24]:

```

```