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 [1]:
```
import numpy as np
data = {'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spoonbills', '

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

import pandas as pd
birds = pd.DataFrame(data = data, index=labels)

birds.head(10)
```

Out[1]:

	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 [2]:
```
birds.info()
```

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

3. Print the first 2 rows of the birds dataframe

In [3]:
```
print(birds[:2])
```

```
    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 [4]:
```
print(birds[['birds', 'age']])
```

```
         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 [5]:
```
birds[['birds', 'age', 'visits']].iloc[[2,3,7]]
```

Out[5]:

	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 [6]:
```
visit_less_than_4 = birds[birds['visits'] < 4]
print(visit_less_than_4)
```

```
        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 [7]:
```
age_NaN = birds[birds['age'].isna()].filter(items=['birds', 'age'])
print(age_NaN)
```

```
        birds  age
d  spoonbills  NaN
h      Cranes  NaN
```

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

In [8]:
```
birds[(birds['birds'] == 'Cranes') & (birds['age'] < 4)]
```

Out[8]:

	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 [9]:
```
birds[birds.age.between(2,4, inclusive=True)]
```

Out[9]:

	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 [10]:
```
birds[birds['birds']=='Cranes']['visits'].sum()
```

Out[10]: 12

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

In [11]:
```
birds[['birds','age']].groupby(['birds']).mean()
```

Out[11]:

birds	age
Cranes	3.5
plovers	3.5
spoonbills	6.0

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 [12]:
```
new_row = pd.DataFrame({'birds':'Cranes', 'age':3.0, 'visits':2, 'priority': '
birds = birds.append(new_row)
birds
```

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
k	Cranes	3.0	2	yes

In [14]:
```
birds.drop(index='k', inplace=True)
birds
```

Out[14]:

	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 [15]:
```
birds.groupby(['birds']).size()
```

Out[15]:
```
birds
Cranes        4
plovers       2
spoonbills    4
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 [16]:
```
birds.sort_values(['age', 'visits'], ascending=[False, True], axis=0)
```

Out[16]:

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

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

In [17]:
```
# clone so that original one doesn't get affected
birds_clone = birds.copy(deep=True)
birds_clone['priority'] = birds_clone.apply(lambda x: 1 if x['priority'] == 'y
birds_clone
```

Out[17]:

	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 [18]:
```
# clone so that original one doesn't get affected
birds_clone = birds.copy(deep=True)

birds_clone['birds'] = birds.apply(lambda x: 'trumpeters' if x['birds']=='Cran
birds_clone
```

Out[18]:

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