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 numpy as np
import pandas as pd
data = pd.DataFrame({'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbil
ls', 'spoonbills', 'Cranes', 'plovers', 'Cranes', 'spoonbills', 'spoonb
ills'], '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', 'ye
s', 'no', 'no', 'no', 'yes', 'no', 'no']})
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
data
```

Out[1]:

	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

	birds	age	visits	priority
9	spoonbills	4.0	2	no

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

In [2]:
```
birds=pd.DataFrame({'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbill
s', 'spoonbills', 'Cranes', 'plovers', 'Cranes', 'spoonbills', 'spoonbi
lls'], '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']},index=[labels])
birds
```

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
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 [3]:
```
g=birds.groupby('birds')
```

```
print(g)
```

```
<pandas.core.groupby.groupby.DataFrameGroupBy object at 0x00000137B9941
4A8>
```

In [4]:
```
for bird_name ,bird_detail in g:
    print(bird_name)
    print(bird_detail)
print(g.describe())
```

```
Cranes
    birds  age  visits priority
a  Cranes  3.5       2      yes
b  Cranes  4.0       4      yes
f  Cranes  3.0       4       no
h  Cranes  NaN       2      yes
plovers
     birds  age  visits priority
c  plovers  1.5       3       no
g  plovers  5.5       2       no
spoonbills
        birds  age  visits priority
d  spoonbills  NaN       4      yes
e  spoonbills  6.0       3       no
i  spoonbills  8.0       3       no
j  spoonbills  4.0       2       no
             age                                                visits
\
           count mean       std  min   25%  50%   75%  max  count mean

birds

Cranes       3.0  3.5  0.500000  3.0  3.25  3.5  3.75  4.0    4.0  3.0

plovers      2.0  3.5  2.828427  1.5  2.50  3.5  4.50  5.5    2.0  2.5

spoonbills   3.0  6.0  2.000000  4.0  5.00  6.0  7.00  8.0    4.0  3.0
```

```
                std  min   25%  50%   75%  max
birds
Cranes      1.154701  2.0  2.00  3.0  4.00  4.0
plovers     0.707107  2.0  2.25  2.5  2.75  3.0
spoonbills  0.816497  2.0  2.75  3.0  3.25  4.0
```

3. Print the first 2 rows of the birds dataframe

In [5]:
```
birds.iloc[:2]
```

Out[5]:

	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 [6]:
```
birds[['birds','age']]
```

Out[6]:

	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

	birds	age
i	spoonbills	8.0
j	spoonbills	4.0

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

In [7]:
```
m=birds.loc[:, ['birds', 'age', 'visits']]
m.iloc[[2,3,7],:]
```

Out[7]:

	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 [8]:
```
birds[birds.visits<4]
```

Out[8]:

	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

	birds	age	visits	priority
j	spoonbills	4.0	2	no

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

In [9]:
```
null_data = birds[birds.isnull().age]
null_data.loc[:,['birds', 'visits']]
```

Out[9]:

	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 [10]:
```
m=birds.loc[birds.birds=='Cranes',:]
m[m.age<4]
```

Out[10]:

	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 [11]:
```
m=birds.loc[birds.age>=2]
m.loc[m.age<=4]
```

Out[11]:

	birds	age	visits	priority

	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 [12]:
```
birds.loc[birds.birds == 'Cranes', 'visits'].sum()
```

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

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

In [13]:
```
g.mean().age
```

Out[13]:
```
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 [17]:
```
row=pd.DataFrame({'birds':'crow','age':2.0,'visits':4.0,'priority':'no'
},index=['k'])
birds_p=birds.append(row, ignore_index=False, sort=False)

birds_p
```

Out[17]:

	birds	age	visits	priority

	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	crow	2.0	4.0	no

In [18]:
```
birds_p = birds_p.drop('k')
birds_p
```

Out[18]:

	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

	birds	age	visits	priority
(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 [19]:
```
group=birds.groupby('birds')
group.count()
```

Out[19]:

	age	visits	priority
birds			
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 [20]:
```
birds.sort_values(by=['age'], ascending=[False])
```

Out[20]:

	birds	age	visits	priority
i	spoonbills	8.0	3	no
e	spoonbills	6.0	3	no
g	plovers	5.5	2	no
b	Cranes	4.0	4	yes

	birds	age	visits	priority
j	spoonbills	4.0	2	no
a	Cranes	3.5	2	yes
f	Cranes	3.0	4	no
c	plovers	1.5	3	no
d	spoonbills	NaN	4	yes
h	Cranes	NaN	2	yes

In [21]:
```
birds.sort_values(by=['visits'], ascending=[True])
```

Out[21]:

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

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

In [22]:
```
birds.loc[birds.priority == 'yes','priority'] = 1
birds.loc[birds.priority == 'no','priority'] = 0
birds
```

Out[22]:

	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 [23]:
```
birds.loc[birds.birds == 'Cranes','birds'] = 'trumpeters'
birds
```

Out[23]:

	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

	birds	age	visits	priority
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