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 pandas as pd
import numpy as np

df = pd.DataFrame({
    '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']
}, index=['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j'])

df
```

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]:

```
df.describe()
```

Out[2]:

	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 [3]:

```
df.iloc[:2]
```

Out[3]:

	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]:

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

Out[4]:

	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]:

```
df[['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]:

```
df[df['visits'] < 4]
```

Out[6]:

	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]:

```
df[['birds','visits']][np.isnan(df['age'])]
```

Out[7]:

	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 [8]:

```
df[(df.birds == 'Cranes') & (df.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]:

```
df[(df.age >= 2) & (df.age <= 4)]
```

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]:

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

Out[10]:

12

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

In [11]:

```
d_birds = df['birds'].unique()
g = df.groupby(df.birds)
for i in d_birds:
    print('Bird Type:{0}, Average age:{1}'.format(i,g.get_group(i).mean()['age']))
```

```
Bird Type:Cranes, Average age:3.5
Bird Type:plovers, Average age:3.5
Bird Type:spoonbills, Average age: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 dataframe
new_df = pd.DataFrame({
    'birds': ['spoonbills'],
    'age': [3.5],
    'visits': [4],
    'priority': ['yes']
}, index=['k'])

new_df
```

Out[12]:

	birds	age	visits	priority
k	spoonbills	3.5	4	yes

In [13]:

```
# appending df
df = pd.concat([df,new_df])
df
```

Out[13]:

	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	spoonbills	3.5	4	yes

In [14]:

In [14]:

```
# remove row that create
df = df.drop(index='k')
df
```

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]:

```
d_birds = df['birds'].unique()
g = df.groupby(df.birds)
for i in d_birds:
    print('Bird Type:{0}, Count:{1}'.format(i,g.get_group(i).count()['birds']))
```

```
Bird Type:Cranes, Count:4
Bird Type:plovers, Count:2
Bird Type:spoonbills, Count: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 [16]:

```
# Sort df by age in descending order
df.sort_values(by='age', ascending=False)
```

Out[16]:

	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
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 [17]:

In [17]:

```
# Sort df by visits in ascending order
df.sort_values(by='visits')
```

Out[17]:

	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 [18]:

```
df = df.replace({'priority':{'yes':1,'no':0}})
df
```

Out[18]:

	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 [19]:

```
df = df.replace({'birds':'Cranes'}, 'trumpeters')
df
```

Out[19]:

	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

In [ ]: