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

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_birds.info()

df_birds.describe()
```

```
<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
```

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 [7]:
```
print(df_birds[:2],'\n')

print(df_birds.iloc[:2])
```

```
    birds  age  visits priority
a  Cranes  3.5       2      yes
b  Cranes  4.0       4      yes

    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 [9]:
```
print(df_birds.iloc[:, 0:2])

# print(df_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
          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 [17]:
```
print(df_birds.iloc[[2,3,7], [0,1,2]])

print(df_birds.iloc[[2,3,7]])
```

```
          birds  age  visits
c     plovers  1.5       3
d  spoonbills  NaN       4
h      Cranes  NaN       2
          birds  age  visits priority
c     plovers  1.5       3       no
d  spoonbills  NaN       4      yes
h      Cranes  NaN       2      yes
```

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

In [12]:
```
df_birds[df_birds["visits"] < 4]
```

Out[12]:

	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 [21]:
```
df_birds.loc[df_birds["age"].isnull(),["birds", "visits"]]
```

Out[21]:

	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 [108]:
```
df_birds[(df_birds["birds"] == "Cranes") & (df_birds["age"] < 4)]
```

Out[108]:

	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 [23]:
```
df_birds[df_birds["age"].between(2,4)]
```

Out[23]:

	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]:
```
data = df_birds[df_birds["birds"] == "Cranes"]
data["visits"].sum()
```

Out[12]: 12

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

In [111]:
```
df_birds.groupby("birds").mean()
```

Out[111]:

birds	age	visits
Cranes	3.5	3.0
plovers	3.5	2.5
spoonbills	6.0	3.0

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 [112]:
```
df_birds.loc["k"] = ["Peacock", "10","5","yes"]
df_birds
```

Out[112]:

	birds	age	visits	priority
a	Cranes	3.5	2	yes
b	Cranes	4	4	yes
c	plovers	1.5	3	no
d	spoonbills	NaN	4	yes
e	spoonbills	6	3	no
f	Cranes	3	4	no
g	plovers	5.5	2	no
h	Cranes	NaN	2	yes
i	spoonbills	8	3	no
j	spoonbills	4	2	no
k	Peacock	10	5	yes

In [113]:
```
df_birds = df_birds.drop(['k'])
df_birds
```

Out[113]:

	birds	age	visits	priority
a	Cranes	3.5	2	yes
b	Cranes	4	4	yes
c	plovers	1.5	3	no
d	spoonbills	NaN	4	yes
e	spoonbills	6	3	no
f	Cranes	3	4	no
g	plovers	5.5	2	no
h	Cranes	NaN	2	yes
i	spoonbills	8	3	no
j	spoonbills	4	2	no

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

In [15]:
```
#pd.value_counts(df_birds["birds"].values)
df_birds["birds"].value_counts()
```

Out[15]:
```
spoonbills    4
Cranes        4
plovers       2
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 [25]:
```
df_birds.sort_values('age', ascending=False).sort_values('visits', ascending=True)
```

Out[25]:

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

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

In [116]:
```
df_birds["priority"] = df_birds["priority"].map({"yes":"1", "no":"0"})
df_birds
```

Out[116]:

	birds	age	visits	priority
a	Cranes	3.5	2	1
b	Cranes	4	4	1
c	plovers	1.5	3	0
d	spoonbills	NaN	4	1
e	spoonbills	6	3	0
f	Cranes	3	4	0
g	plovers	5.5	2	0
h	Cranes	NaN	2	1
i	spoonbills	8	3	0
j	spoonbills	4	2	0

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

In [117]:
```
df_birds["birds"] = df_birds["birds"].replace({"Cranes":"trumpeters"})
df_birds
```

Out[117]:

	birds	age	visits	priority
a	trumpeters	3.5	2	1
b	trumpeters	4	4	1
c	plovers	1.5	3	0
d	spoonbills	NaN	4	1
e	spoonbills	6	3	0
f	trumpeters	3	4	0
g	plovers	5.5	2	0
h	trumpeters	NaN	2	1
i	spoonbills	8	3	0
j	spoonbills	4	2	0