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 [31]:

```
data = {'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spoonbills', 'Cranes', 'plo

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
import numpy as np
import pandas as pd
from pandas import DataFrame

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

Out[31]:

	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 [32]:

```
df.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 [44]:

```
df.iloc[:2, 0:4]
```

Out[44]:

	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 [45]:

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

Out[45]:

	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 [62]:

```
df.iloc[[2,3,7],[0,1,2]]
```

Out[62]:

	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 [66]:

```
print(df[df['visits']<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 [71]:

```
df[df['age'].isnull()]
```

Out[71]:

	birds	age	visits	priority
d	spoonbills	NaN	4	yes
h	Cranes	NaN	2	yes

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

In [89]:

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

Out[89]:

	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 [90]:

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

Out[90]:

	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 [100]:

```
df[(df['birds'] =='Cranes') & (df['visits'] > 0)].sum()
```

Out[100]:

```
birds       CranesCranesCranesCranes
age                             10.5
visits                            12
priority                  yesyesnoyes
dtype: object
```

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

In [102]:

```
df['age'].mean()
```

Out[102]:

```
4.4375
```

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 [134]:

```
df.loc['k'] = ['eagle',7.0,9,'yes']
df = df.drop('k')
df
```

Out[134]:

	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

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

In [128]:

```
print(df.groupby(df["birds"]).count())
```

```
            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 [133]:

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

Out[133]:

	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 [140]:

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

Out[140]:

```
i    0
e    0
g    0
j    0
b    1
a    1
f    0
c    0
h    1
d    1
Name: priority, dtype: int64
```

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

In [141]:

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

Out[141]:

```
i    spoonbills
e    spoonbills
g       plovers
j    spoonbills
b    trumpeters
a    trumpeters
f    trumpeters
c       plovers
h    trumpeters
d    spoonbills
Name: birds, dtype: object
```

In [ ]: