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 [107]:
```
import pandas as pd
import numpy as np
df =pd.DataFrame(data = {'birds': ['Cranes', 'Cranes', 'plovers', 'spoo
nbills', 'spoonbills', 'Cranes', 'plovers', 'Cranes', 'spoonbills', 'sp
oonbills'],
                         '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[107]:

	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

	birds	age	visits	priority
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 [108]: df.describe()
```

Out[108]:

	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 [109]: df.head(2)
```

Out[109]:

	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 [110]:
```
df[['birds','age']]
```

Out[110]:

	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 [115]:
```
first = df.loc[['c','d','h'], ['birds', 'age', 'visits']]
first
```

Out[115]:

	birds	age	visits
c	plovers	1.5	3
d	spoonbills	NaN	4
h	Cranes	NaN	2

In [116]:

```
#another way
print(df['birds'].iloc[2],df['age'].iloc[2],df['visits'].iloc[2])
print(df['birds'].iloc[3],df['age'].iloc[3],df['visits'].iloc[3])
print(df['birds'].iloc[7],df['age'].iloc[7],df['visits'].iloc[7])
```

```
plovers 1.5 3
spoonbills nan 4
Cranes nan 2
```

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

In [119]:

```
##df.loc[df.visits<4]
df.loc[df['visits']<4]
```

Out[119]:

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

Out[121]:

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

Out[38]:

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

Out[39]:

	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 [40]:
```
df[(df['birds']=='Cranes') & (df['visits']>0)].sum()
```

Out[40]:
```
birds       CranesCranesCranesCranes
age                             10.5
visits                            12
priority                yesyesnoyes
dtype: object
```

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

In [47]:
```
print('Cranes')
print(df['age'][df['birds'] =='Cranes'].mean())
print('spoonbills')
print(df['age'][df['birds'] =='spoonbills'].mean())
print('plovers')
print(df['age'][df['birds'] =='plovers'].mean())
```

```
Cranes
3.5
spoonbills
6.0
plovers
3.5
```

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 [14]:
```
import pandas as pd
import numpy as np
new_data = [({'birds':'parrots','age':2.5, 'visits':3,'priority':'yes'
})]
data = pd.DataFrame(new_data,index=['k'])
data
```

Out[14]:

	age	birds	priority	visits

	age	birds	priority	visits
k	2.5	parrots	yes	3

In [15]:
```
old_data = pd.DataFrame(data = {'birds': ['Cranes', 'Cranes', 'plovers'
, 'spoonbills', 'spoonbills', 'Cranes', 'plovers', 'Cranes', 'spoonbill
s', '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'])
```

In [20]:
```
data = pd.concat([data,old_data],ignore_index = False,sort = False)
data
```

Out[20]:

	age	birds	priority	visits
k	2.5	parrots	yes	3
a	3.5	Cranes	yes	2
b	4.0	Cranes	yes	4
c	1.5	plovers	no	3
d	NaN	spoonbills	yes	4
e	6.0	spoonbills	no	3
f	3.0	Cranes	no	4
g	5.5	plovers	no	2
h	NaN	Cranes	yes	2
i	8.0	spoonbills	no	3

	age	birds	priority	visits
j	4.0	spoonbills	no	2

In [22]:
```
data.drop('k')
```

Out[22]:

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

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

In [71]:
```
df.groupby('birds').count()
```

Out[71]:

	age	visits	priority
birds			
Cranes	3	4	4
plovers	2	2	2

	age	visits	priority
birds			
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 [87]:
```
print(df)
print("------------------------------------------------")

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

```
        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
------------------------------------------------
sorted data frame
```

Out[87]:

	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

	birds	age	visits	priority
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 [89]: df.replace(to_replace = ['yes','no'],value = [1,0])
```

Out[89]:

	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 [91]: df.replace( 'Cranes', 'trumpeters')
```

```
Out[91]:
```

	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