

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 [3]: import numpy as np  
import pandas as pd
```

```
In [ ]: data = {'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spoonbi  
lls', '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 [88]: df= pd.DataFrame(data,index=labels)  
df
```

Out[88]:

	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

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

Out[6]:

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

Out[9]:

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

Out[13]:

	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 [15]: `df[['birds', 'age', 'visits']].iloc[[2,3,7]]`

Out[15]:

	birds	age	visits
--	-------	-----	--------

	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 [17]: `df[df.visits <4]`

Out[17]:

	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 [37]: `df[df.age.isnull()][['birds', 'visits']]`

Out[37]:

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

Out[42]:

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

Out[43]:

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

Out[50]: 12

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

```
In [54]: g=df.groupby('birds')
g.mean().age
```

```
Out[54]: 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 [72]: df2=pd.DataFrame({'birds':'Peacock','age':2,'visits':4,'priority':'yes'},index=['k'])
df=df.append(df2)
print(df)
df=df.drop('k')
```

```
      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   Peacock  2.0      4      yes
k   Peacock  2.0      4      yes
```

Out[72]:

	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

	birds	age	visits	priority
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 [78]: g=df.groupby('birds')
g.size()
```

```
Out[78]: birds
Cranes      4
plovers     2
spoonbills  4
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 [83]: df.sort_values(by=['age', 'visits'], ascending=[False, True])
```

```
Out[83]:
```

	birds	age	visits	priority
i	spoonbills	8.0	3	no
e	spoonbills	6.0	3	no
g	plovers	5.5	2	no

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

```
In [92]: df
```

```
Out[92]:
```

	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

	birds	age	visits	priority
j	spoonbills	4.0	2	0

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

In [94]: `df.birds=df.birds.replace('Cranes','trumpeters')`

In [95]: `df`

Out[95]:

	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
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