

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 [166... 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']}
print(df)
```

	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 [167... df.info()

<class 'pandas.core.frame.DataFrame'>
Index: 10 entries, a to j
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype
---  -
 0   birds       10 non-null    object
 1   age         8 non-null     float64
 2   visits      10 non-null    int64
 3   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 [168... df.head(2)
```

```
Out[168]:
```

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

```
Out[169]:
```

	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 [170]: df.index
# [1, 2, 3] are indices of [birds, age, visits]
df.iloc[[2, 3, 7], [1, 2, 3]]
```

```
Out[170]:
```

	age	visits	priority
c	1.5	3	no
d	NaN	4	yes
h	NaN	2	yes

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

```
In [171]: df[df['visits'] < 4]
```

```
Out[171]:
```

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

```
Out[172]:
```

	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 [188...] df[df['birds'] == 'Cranes'][df['age'] < 4]  
df
```

C:\Users\Guruprasad Sajjan\AppData\Local\Temp\ipykernel_9728\2773224736.py:1: UserWarning: Boolean Series key will be reindexed to match DataFrame index.
df[df['birds'] == 'Cranes'][df['age'] < 4]

```
Out[188]:
```

	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
k	Penguin	3.2	4	1

9. Select the rows the age is between 2 and 4(inclusive)

```
In [189...] df[df['age']>= 2 ][df['age'] < 5]
```

C:\Users\Guruprasad Sajjan\AppData\Local\Temp\ipykernel_9728\1340239138.py:1: UserWarning: Boolean Series key will be reindexed to match DataFrame index.
df[df['age']>= 2][df['age'] < 5]

Out[189]:

	birds	age	visits	priority
a	trumpeters	3.5	2	1
b	trumpeters	4.0	4	1
f	trumpeters	3.0	4	0
j	spoonbills	4.0	2	0
k	Penguin	3.2	4	1

10. Find the total number of visits of the bird Cranes

```
In [175... g = df[df['birds'] == 'Cranes']
np.sum(g.loc[:, 'visits'])
```

Out[175]: 12

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

```
In [176... group_birds = df.groupby('birds')
group_birds.mean()['age']
```

Out[176]:

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 [177... df1 = pd.DataFrame({'birds': 'Penguin', 'age' : 3.2, 'visits' : 4, 'priority' : 'yes'}, index=[10])
df = pd.concat([df, df1])
df
```

Out[177]:

	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	Penguin	3.2	4	yes

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

```
In [178... group_birds = df.groupby('birds')
group_birds.count()
```

Out[178]:

	age	visits	priority
birds			
Cranes	3	4	4
Penguin	1	1	1
plovers	2	2	2
spoonbills	3	4	4

14. Sort dataframe (birds) first by the values in the 'age' in descending order, then by the value in the 'visits' column in ascending order.

```
In [185... df.sort_values(by = ['age', 'visits'], ascending = [False, True])
```

Out[185]:

	birds	age	visits	priority
i	spoonbills	8.0	3	0
e	spoonbills	6.0	3	0
g	plovers	5.5	2	0
j	spoonbills	4.0	2	0
b	trumpeters	4.0	4	1
a	trumpeters	3.5	2	1
k	Penguin	3.2	4	1
f	trumpeters	3.0	4	0
c	plovers	1.5	3	0
h	trumpeters	NaN	2	1
d	spoonbills	NaN	4	1

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

```
In [183... df.loc[df['priority'] == 'yes', 'priority'] = 1
df.loc[df['priority'] == 'no', 'priority'] = 0
df
```

Out[183]:

	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
k	Penguin	3.2	4	1

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

```
In [180... df.loc[df['birds'] == 'Cranes', 'birds'] = 'trumpeters'  
df
```

Out[180]:

	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
k	Penguin	3.2	4	yes