Pandas Exercises [Q1 – 16] 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 [4]:

```
import pandas as pd
import numpy as np

data = {
    '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', 'r
}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

birds = pd.DataFrame(data, index=labels)
print(birds)
```

```
        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 [7]:

```
birds.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 [41]:

```
birds.head(2)
```

Out[41]:

	birds	age	visits	priority
a	trumpeters	3.5	2	1
b	trumpeters	4.0	4	1

4. Print all the rows with only 'birds' and 'age' columns from the dataframe

In [9]:

```
birds[['birds','age']]
```

Out[9]:

	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]:

```
selected_rows = birds.loc[['c', 'd', 'h'], ['birds', 'age', 'visits']]
print(selected_rows)
```

```
        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 [18]:

```
selected_rows = birds[birds['visits'] < 4]
print(selected_rows)
```

```
        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]:

```
selected_rows = birds[birds['age'].isnull()][['birds', 'visits']]
print(selected_rows)
```

```
        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 [23]:

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

```
    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 [24]:

```
selected_rows = birds[(birds['age'] >= 2) & (birds['age'] <= 4)]
print(selected_rows)
```

```
        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 [25]:

```
cranes_visits = birds[birds['birds'] == 'Cranes']['visits'].sum()
print("Total number of visits of the bird Cranes:", cranes_visits)
```

```
Total number of visits of the bird Cranes: 12
```

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

In [26]:

```
mean_age_per_bird = birds.groupby('birds')['age'].mean()
print(mean_age_per_bird)
```

```
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 [34]:

```
# Append a new row 'k' to the DataFrame
new_row = pd.DataFrame({'birds': ['Parrot'], 'age': [2], 'visits': [3],
birds = pd.concat([birds,new_row])

print("DataFrame with the new row 'k':")
print(birds)

# Delete the row 'k' to return the original DataFrame
birds = birds.drop('k')

print("\nOriginal DataFrame after deleting the row 'k':")
print(birds)
```

```
DataFrame with the new row '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      Parrot  2.0       3       no

Original DataFrame after deleting the row '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
```

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

In [32]:

```
bird_counts = birds['birds'].value_counts()
bird_counts
```

Out[32]:

```
birds
Cranes        4
spoonbills    4
plovers       2
Name: count, 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 [35]:

```
sorted_birds = birds.sort_values(by=['age', 'visits'], ascending=[False
print(sorted_birds)
```

```
        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 [36]:

```
birds['priority'] = birds['priority'].replace({'yes': 1, 'no': 0})
print(birds)
```

```
        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 [39]:

```
birds['birds'] = birds['birds'].replace('Cranes', 'trumpeters')
print(birds)
```

```
        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
```