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 [1]:
```
# importing the required packages
import numpy as np
import pandas as pd


data = {'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spoonbills', 'Cra
        'age': [3.5, 4, 1.5, np.nan, 6, 3, 5.5, np.nan, 8, 4], 'visits': [2, 4, 3
        'priority': ['yes', 'yes', 'no', 'yes', 'no', 'no', 'no', 'yes', 'no', 'n

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

# Creating dataframe
pd.DataFrame(data, index = labels)
```

Out[1]:

	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 [2]:
```
df = pd.DataFrame(data, index = labels)

print("Summary of the basic information about birds DataFrame and its data: ")
df.describe()
```

```
Summary of the basic information about birds DataFrame and its data:
```

Out[2]:

	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 [3]:
```
#displaying 2 rows of birds dataframe
df.head(2)
```

Out[3]:

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

Out[4]:

	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 [5]:
```
# Declaring the data set by other variable name
data1 = {'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spoonbills', 'Cr
        'age': [3.5, 4, 1.5, np.nan, 6, 3, 5.5, np.nan, 8, 4], 'visits': [2, 4, 3
        'priority': ['yes', 'yes', 'no', 'yes', 'no', 'no', 'no', 'yes', 'no', 'n

df1 = pd.DataFrame(data1,index=["1","2","3","4","5","6","7","8","9","10"] )

df1.loc[["2","3","7"],("birds","age","visits")]
```

Out[5]:

	birds	age	visits
2	Cranes	4.0	4
3	plovers	1.5	3
7	plovers	5.5	2

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

In [6]:
```
#No. ofless than 4
df = pd.DataFrame(data, index=labels)
df[df["visits"] < 4]
```

Out[6]:

	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 [7]:
```
df.loc[df["age"].isnull()]
```

Out[7]:

	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 [8]:
```
df[(df["birds"] == "Cranes") & (df["age"] < 4)]
```

Out[8]:

	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 [9]:
```
df[(df["age"]>2) & (df["age"]<=4)]
```

Out[9]:

	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 [10]:
```
df[df["birds"] == "Cranes"].sum() ["visits"]
```

Out[10]:
```
12
```

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

In [11]:
```
df.groupby(["birds"]).mean() ["age"]
```

Out[11]:
```
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 [12]:
```
print("Appending kth row  into dataframe:")

data2 = {"birds":"Cuckoo",
         "age":3,"visits":2,
         "priority":"yes"}

x = pd.DataFrame(data2,index=["k"],columns=["birds","age","visits","priority"])

df.append(x)
```

```
Appending kth row  into dataframe:
```

Out[12]:

	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	Cuckoo	3.0	2	yes

In [13]:
```
print("Deleting kth row from the dataframe:")
df.drop(df.tail(0).index)
```

```
Deleting kth row from the dataframe:
```

Out[13]:

	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 [14]:
```
df["birds"].value_counts()
```

Out[14]:
```
Cranes        4
spoonbills    4
plovers       2
Name: birds, 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 [15]:
```
print("Sorting the age column in descending order : \n")
sort_birds = df.sort_values("age", ascending=False)
print(sort_birds,"\n")

print("Sorting the visit column in ascending order : \n")
sort_visits = df.sort_values("visits", ascending=True)
print(sort_visits)
```

```
Sorting the age column in descending order :

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

Sorting the visit column in ascending order :

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

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

In [16]:
```
df.replace(["yes","no"],[1,0])
```

Out[16]:

	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 [17]:
```
df.replace(["Cranes"],["trumpeters"])
```

Out[17]:

	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