In [ ]:
```
#Consider the following Python dictionary data and Python list labels
data = {'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spoonbi
lls', 'Cranes', 'plovers', 'Cranes', 'spoonbills', 'spoonbills']
        , 'age': [3.5, 4, 1.5,'nan', 6, 3, 5.5,'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 in
dex labels.
#2. Display a summary of the basic information about birds DataFrame an
d its data.
#3. Print the first 2 rows of the birds dataframe
#4. Print all the rows with only 'birds' and 'age' columns from the dat
aframe
#5. select [2, 3, 7] rows and in columns ['birds', 'age', 'visits']
#6. select the rows where the number of visits is less than 4
#7. select the rows with columns ['birds', 'visits'] where the age is m
issing i.e NaN
#8. Select the rows where the birds is a Cranes and the age is less tha
n 4
#9. Select the rows the age is between 2 and 4(inclusive)
#10. Find the total number of visits of the bird Cranes
#11. Calculate the mean age for each different birds in dataframe.
#12. Append a new row 'k' to dataframe with your choice of values for e
ach column. Then delete that row to return the original DataFrame.
#13. Find the number of each type of birds in dataframe (Counts)
#14. Sort dataframe (birds) first by the values in the 'age' in decendi
ng order, then by the value in the 'visits' column in ascending order.
#15. Replace the priority column values with'yes' should be 1 and 'no'
 should be 0
#16. In the 'birds' column, change the 'Cranes' entries to 'trumpeter
s'.
```

In [4]:
```
#1. Create a DataFrame birds from this dictionary data which has the in
```

```
dex labels.
import pandas as pd
birds_df=pd.DataFrame({'birds': ['Cranes', 'Cranes', 'plovers', 'spoonb
ills', 'spoonbills', 'Cranes', 'plovers', 'Cranes', 'spoonbills', 'spoo
nbills'],
                       'age': [3.5, 4, 1.5,'nan', 6, 3, 5.5,'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'] )
birds_df
```

Out[4]:

	birds	age	visits	priority
a	Cranes	3.5	2	yes
b	Cranes	4	4	yes
c	plovers	1.5	3	no
d	spoonbills	nan	4	yes
e	spoonbills	6	3	no
f	Cranes	3	4	no
g	plovers	5.5	2	no
h	Cranes	nan	2	yes
i	spoonbills	8	3	no
j	spoonbills	4	2	no

In [7]:
```
#2. Display a summary of the basic information about birds DataFrame an
d its data.

g=birds_df.groupby('birds')
g
for i,j in g:
```

```
    print(i)
    print(j)
```

```
Cranes
    birds  age  visits priority
a  Cranes  3.5       2      yes
b  Cranes    4       4      yes
f  Cranes    3       4       no
h  Cranes  nan       2      yes
plovers
     birds  age  visits priority
c  plovers  1.5       3       no
g  plovers  5.5       2       no
spoonbills
        birds  age  visits priority
d  spoonbills  nan       4      yes
e  spoonbills    6       3       no
i  spoonbills    8       3       no
j  spoonbills    4       2       no
```

In [13]:
```
#3. Print the first 2 rows of the birds dataframe
s=birds_df.iloc[0:2]
s
```

Out[13]:

	birds	age	visits	priority
a	Cranes	3.5	2	yes
b	Cranes	4	4	yes

In [14]:
```
#4. Print all the rows with only 'birds' and 'age' columns from the dat
aframe
df=birds_df.iloc[:,0:2]
df
```

Out[14]:

	birds	age

	birds	age
a	Cranes	3.5
b	Cranes	4
c	plovers	1.5
d	spoonbills	nan
e	spoonbills	6
f	Cranes	3
g	plovers	5.5
h	Cranes	nan
i	spoonbills	8
j	spoonbills	4

In [26]:
```
#5. select [2, 3, 7] rows and in columns ['birds', 'age', 'visits']
df1=birds_df.iloc[1:3,0:3]
df1
df2=birds_df.iloc[6:7,0:3]
df2
c=pd.concat([df1,df2])
c
```

Out[26]:

	birds	age	visits
b	Cranes	4	4
c	plovers	1.5	3
g	plovers	5.5	2

In [27]:
```
#6. select the rows where the number of visits is less than 4
birds_df[birds_df['visits']<4]
```

Out[27]:

	birds	age	visits	priority
a	Cranes	3.5	2	yes
c	plovers	1.5	3	no
e	spoonbills	6	3	no
g	plovers	5.5	2	no
h	Cranes	nan	2	yes
i	spoonbills	8	3	no
j	spoonbills	4	2	no

In [30]:
```
#7. select the rows with columns ['birds', 'visits'] where the age is m
issing i.e NaN
df=birds_df[birds_df['age']=='nan']
df
df[['birds', 'visits']]
```

Out[30]:

	birds	visits
d	spoonbills	4
h	Cranes	2

In [39]:
```
#8. Select the rows where the birds is a Cranes and the age is less tha
n 4
df1=birds_df[birds_df['birds']=='Cranes']
df1
df2=df1.iloc[0:3]
df2
df3=df2[df2['age']<4]
df3
```

Out[39]:

	birds	age	visits	priority

	birds	age	visits	priority
a	Cranes	3.5	2	yes
f	Cranes	3	4	no

In [44]:
```
#9. Select the rows the age is between 2 and 4(inclusive)
df=birds_df[birds_df['age']!='nan']
df
df1=df[df['age']<=4]
df1
```

Out[44]:

	birds	age	visits	priority
a	Cranes	3.5	2	yes
b	Cranes	4	4	yes
c	plovers	1.5	3	no
f	Cranes	3	4	no
j	spoonbills	4	2	no

In [46]:
```
#10. Find the total number of visits of the bird Cranes
df1=birds_df[birds_df['birds']=='Cranes']
df1
df2=df1['visits'].sum()
df2
```

Out[46]: 12

In [4]:
```
#11. Calculate the mean age for each different birds in dataframe.
import pandas as pd
birds_df=pd.DataFrame({'birds': ['Cranes', 'Cranes', 'plovers', 'spoonb
ills', 'spoonbills', 'Cranes', 'plovers', 'Cranes', 'spoonbills', 'spoo
nbills'],
                       'age': [3.5, 4, 1.5,'nan', 6, 3, 5.5,'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=birds_df[birds_df['age']!='nan']
df
df1=df.iloc[:,0:2]
df1
df_cranes=df1[df1['birds']=='Cranes']
a=df_cranes['age'].mean()
print("Mean age for cranes=",a)
df_plovers=df1[df1['birds']=='plovers']
b=df_plovers['age'].mean()
print("Mean age for plovers=",b)
df_spoonbills=df1[df1['birds']=='spoonbills']
c=df_spoonbills['age'].mean()
print("Mean age for spoonbills=",c)

# I was getting "No numeric types to aggregate" error when i used g.mea
n() after grouping birds. Hence calculated mean by extracting data fram
es for each bird.
```

```
Mean age for cranes= 3.5
Mean age for plovers= 3.5
Mean age for spoonbills= 6.0
```

In [79]:

```
#12.Append a new row 'k' to dataframe with your choice of values for ea
ch column. Then delete that row to return the original DataFrame.
newrow_df=pd.DataFrame({'birds': ['Cranes', 'Cranes', 'plovers', 'spoon
bills', 'spoonbills', 'Cranes', 'plovers', 'Cranes', 'spoonbills', 'spo
onbills','Crows'],
                        'age': [3.5, 4, 1.5,'nan', 6, 3, 5.5,'nan', 8, 4,
5],'visits': [2, 4, 3, 4, 3, 4, 2, 2, 3, 2,5],'priority': ['yes', 'yes'
, 'no', 'yes', 'no', 'no', 'no', 'yes', 'no', 'no','yes']},
                        index=['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h',
'i', 'j','k'] )
newrow_df
original_df=newrow_df.loc['a':'j']
original_df
```

Out[79]:

	birds	age	visits	priority
a	Cranes	3.5	2	yes
b	Cranes	4	4	yes
c	plovers	1.5	3	no
d	spoonbills	nan	4	yes
e	spoonbills	6	3	no
f	Cranes	3	4	no
g	plovers	5.5	2	no
h	Cranes	nan	2	yes
i	spoonbills	8	3	no
j	spoonbills	4	2	no

In [83]:
```
#13. Find the number of each type of birds in dataframe (Counts)

g=birds_df.groupby('birds')
g.describe()
```

Out[83]:

	visits							
	count	mean	std	min	25%	50%	75%	max
birds								
Cranes	4.0	3.0	1.154701	2.0	2.00	3.0	4.00	4.0
plovers	2.0	2.5	0.707107	2.0	2.25	2.5	2.75	3.0
spoonbills	4.0	3.0	0.816497	2.0	2.75	3.0	3.25	4.0

In [109]:
```
#14. Sort dataframe (birds) first by the values in the 'age' in decendi
```

```
ng order, then by the value in the 'visits' column in ascending order.

df=birds_df[birds_df['age']!='nan']
df
df1=df.sort_values(by=['age'], ascending=[False])
df1
print("Dataframe for age values in descending order:",df1)
df2=df.sort_values(by=['visits'], ascending=[True])
print("Dataframe for visits values in ascending order:",df2)

#reference link:https://stackoverflow.com/questions/17618981/how-to-sor
t-pandas-data-frame-using-values-from-several-columns
```

```
Dataframe for age values in descending order:         birds  age  visit
s priority
i  spoonbills    8       3       no
e  spoonbills    6       3       no
g     plovers  5.5       2       no
b      Cranes    4       4      yes
j  spoonbills    4       2       no
a      Cranes  3.5       2      yes
f      Cranes    3       4       no
c     plovers  1.5       3       no
Dataframe for visits values in ascending order:         birds  age  vis
its priority
a      Cranes  3.5       2      yes
g     plovers  5.5       2       no
j  spoonbills    4       2       no
c     plovers  1.5       3       no
e  spoonbills    6       3       no
i  spoonbills    8       3       no
b      Cranes    4       4      yes
f      Cranes    3       4       no
```

In [93]:
```
#15. Replace the priority column values with'yes' should be 1 and 'no'
 should be 0
df1=birds_df.replace('yes','1')
df1
df2=df1.replace('no','0')
```

```
df2

#reference link: https://pandas.pydata.org/pandas-docs/stable/referenc
e/api/pandas.DataFrame.replace.html
```

Out[93]:

	birds	age	visits	priority
a	Cranes	3.5	2	1
b	Cranes	4	4	1
c	plovers	1.5	3	0
d	spoonbills	nan	4	1
e	spoonbills	6	3	0
f	Cranes	3	4	0
g	plovers	5.5	2	0
h	Cranes	nan	2	1
i	spoonbills	8	3	0
j	spoonbills	4	2	0

In [94]:
```
#16. In the 'birds' column, change the 'Cranes' entries to 'trumpeter
s'.
df1=birds_df.replace('Cranes','trumpeters')
df1
```

Out[94]:

	birds	age	visits	priority
a	trumpeters	3.5	2	yes
b	trumpeters	4	4	yes
c	plovers	1.5	3	no
d	spoonbills	nan	4	yes

	birds	age	visits	priority
e	spoonbills	6	3	no
f	trumpeters	3	4	no
g	plovers	5.5	2	no
h	trumpeters	nan	2	yes
i	spoonbills	8	3	no
j	spoonbills	4	2	no