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.

+ Code + Text

```
import pandas as pd
import numpy as np
data = {'birds': ['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spoonbills', 'Cranes', 'pl
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
birds=pd.DataFrame(data, index=labels)
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.

```
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: 720.0+ bytes
```

*3. Print the first 2 rows of the birds dataframe *

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

```
    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

```
print(birds[['birds','age']])
```

```
         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']

```
birds.iloc[[2,3,7],[birds.columns.get_loc('birds'),birds.columns.get_loc('age'),birds.colu
```

	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

```
birds[birds['visits']<4]
```

	birds	age	visits	priority
a	Cranes	3.5	2	yes
c	plovers	1.5	3	no

7. select the rows with columns ['birds', 'visits'] where the age is missing i.e NaN

```
birds.loc[birds['age'].isin(list([np.nan])),['birds', 'visits']]
```

	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

```
birds.loc[birds['age'].isin(list([np.nan])),['birds', 'visits']]
```

	birds	visits
d	spoonbills	4
h	Cranes	2

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

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

	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

```
g=birds.groupby('birds')
g.get_group('Cranes')['visits'].sum()
```

```
12
```

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

```
g['age'].mean()
```

```
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.

```
newbird=pd.DataFrame([('Fly',4,6,'yes')],columns=['birds','age','visits','priority'], inde
birds=birds.append(newbird)
birds=birds.drop(['k'])
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

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

```
g['birds'].count()
```

```
birds
Cranes        4
plovers       2
spoonbills    4
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.

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

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

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

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

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

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

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

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

✓ 0s completed at 2:21 PM