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.

```
#importing pandas

import pandas as pd

#creating birds dataframe
#missing values np.nan are created as empty strings


birds_df = pd.DataFrame ({
    "birds": ['Cranes', 'Cranes', 'plovers', 'spoonbills', 'spoonbills', 'Cranes', 'plovers',
    "age" : [3.5, 4, 1.5, "", 6, 3, 5.5,"" , 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'])

#printing birds_df
birds_df
```

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

2. Display a summary of the basic information about birds DataFrame and its data.

```
#basic information on birds_df DataFrame
birds_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       10 non-null     object
 2   visits    10 non-null     int64 
 3   priority  10 non-null     object
dtypes: int64(1), object(3)
memory usage: 400.0+ bytes
```

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

```
#using iloc to get first 2 rows
birds_df.iloc[:2,]
```

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

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

```
#using iloc to get only 'birds' and 'age' columns
birds_df.iloc[:,:2]
```

	birds	age

5. select [2, 3, 7] rows and in columns ['birds', 'age', 'visits']

```
#using .iloc to select [2, 3, 7] rows and in columns ['birds', 'age', 'visits']
birds_df.iloc[[1,2,6],[0,1,2]]
```

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

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

```
#selecting rows with number of visits less than 4 using .loc
birds_df1 = birds_df.loc[birds_df.visits <4]

#rows with number of visits less than 4 are,
birds_df1
```

	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		2	yes
i	spoonbills	8	3	no
j	spoonbills	4	2	no

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

```
#selecting rows with age missing using .loc
birds_df2 = birds_df.loc[birds_df.age =="",['birds', 'visits']]

#Birds and visits with age missing are,
birds_df2
```

	birds	visits
d	spoonbills	4

8. Select the rows where the birds is a Cranes and the age is less than 4

```
#converting the age column to numeric as missing values were input as string
#the code is reffered from https://stackoverflow.com/questions/46227170/not-supported-between
birds_df["age"] = pd.to_numeric(birds_df["age"])

# selecting rows where birds is a Cranes and the age is less than 4 using loc
birds_df3 = birds_df.loc[(birds_df.birds == "Cranes") & (birds_df.age < 4)]

birds_df3
```

	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)

```
#selecting rows where age is between 2 and 4 using loc
birds_df4 = birds_df.loc[(birds_df.age >= 2) & (birds_df.age <= 4)]
birds_df4
```

	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

```
#creating datframe with birds as "Cranes"
birds_df5 = birds_df.loc[birds_df.birds =="Cranes"]


#total number of visits using sum() function
birds_df5["visits"].sum()
```

```
12
```

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

```
#using Groupby function and mean()
birds_df.groupby('birds').mean()
```

birds	age	visits
Cranes	3.5	3.0
plovers	3.5	2.5
spoonbills	6.0	3.0

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.

```
#creating the new row k
#below code to create new row and append is taken from https://cmdlinetips.com/2021/03/how-to
row_k = pd.Series(data={"birds" : "plovers", "age" : 5, "visits" : 2, "priority" : "yes"},nam
birds_df = birds_df.append(row_k,ignore_index=False)

birds_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
k	plovers	5.0	2	yes

```
#removing the new column added
#code taken from https://cmdlinetips.com/2021/03/how-to-delete-rows-in-pandas-dataframe/
```

```
birds_df.drop("k",axis = "index")
```

	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)

```
#using Groupby function and count()
#applying the count on birds column itself as its not having any missing values
birds_df.groupby('birds')['birds'].count()
```

```
birds
Cranes        4
plovers       3
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.

```
#sorting the dataframe
birds_df.sort_values(['age','visits'],ascending =[False,True])
```

	birds	age	visits	priority
i	spoonbills	8.0	3	no
e	spoonbills	6.0	3	no
g	plovers	5.5	2	no
k	plovers	5.0	2	yes
j	spoonbills	4.0	2	no

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

```
#using·.replace·function·on·whole·dataframe·as·yes·and·no·are·only·on·priority·column
birds_df = birds_df.replace({"yes" : 1, "no" : 0})
birds_df
```

	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
k	plovers	5.0	2	1

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

```
#using .replace function on whole dataframe as Cranes are only in birds column
birds_df = birds_df.replace({"Cranes" : "trumpeters"})
birds_df
```

	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
j	spoonbills	4.0	2	0
k	plovers	5.0	2	1

✓ 0s completed at 11:38 PM