pandas_basics_practice

September 26, 2021

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']

[85]:
```
import numpy as np
import pandas as pd
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']
data= pd.DataFrame.from_dict(data)
labels = pd.DataFrame.from_dict(labels)
data['labels'] = labels
```

1. Create a DataFrame birds from this dictionary data which has the index labels.

[90]:
```
birds = pd.DataFrame(data)
birds = birds.set_index('labels')
birds
```

[90]:

labels	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.

[16]:
```
birds.describe()
```

[16]:
```
             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

[91]:
```
birds.head(2)
```

[91]:
```
          birds  age  visits priority
labels
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

[18]:
```
print(birds[['birds', 'age']])
```

```
        birds  age
0      Cranes  3.5
1      Cranes  4.0
2     plovers  1.5
3  spoonbills  NaN
4  spoonbills  6.0
5      Cranes  3.0
6     plovers  5.5
7      Cranes  NaN
8  spoonbills  8.0
9  spoonbills  4.0
```

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

[23]:
```
birds[['birds','age','visits']].iloc[[2,3,7]]
```

[23]:
```
         birds  age  visits
2     plovers  1.5       3
3  spoonbills  NaN       4
7      Cranes  NaN       2
```

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

[21]:
```
birds[birds['visits']<4]
```

[21]:
```
         birds  age  visits priority labels
0       Cranes  3.5       2      yes      a
2      plovers  1.5       3       no      c
4  spoonbills  6.0       3       no      e
6      plovers  5.5       2       no      g
7       Cranes  NaN       2      yes      h
8  spoonbills  8.0       3       no      i
9  spoonbills  4.0       2       no      j
```

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

[24]:
```
x = birds[birds['age'].isnull()]
x[['birds','visits']]
```

[24]:
```
         birds  visits
3  spoonbills       4
7       Cranes       2
```

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

[26]:
```
t=birds[birds['age']<4]
t=t[t['birds']=='Cranes']
t
```

[26]:
```
    birds  age  visits priority labels
0  Cranes  3.5       2      yes      a
5  Cranes  3.0       4       no      f
```

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

[29]:
```
birds[(birds['age'] >= 2) & (birds['age']<=4)]
```

[29]:
```
         birds  age  visits priority labels
0       Cranes  3.5       2      yes      a
1       Cranes  4.0       4      yes      b
5       Cranes  3.0       4       no      f
9  spoonbills  4.0       2       no      j
```

10. Find the total number of visits of the bird Cranes

[31]:
```
birds[(birds['birds'] == 'Cranes') & (birds['visits'].notnull())].sum()
```

[31]:
```
birds       CranesCranesCranesCranes
age                             10.5
visits                            12
priority                  yesyesnoyes
labels                           abfh
dtype: object
```

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

[67]:
```
birds.groupby('birds')['age'].mean()
```

[67]:
```
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.

[95]:
```
birds.head()
birds.loc['k'] = ['plovers', 3.5, 3,'no']
print(birds)
birds = birds.drop('k')
print("_____________")
print(birds)
```

```
             birds  age  visits priority
labels
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  3.5       3       no

---------------
             birds  age  visits priority
labels
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)

[96]:
```
birds['birds'].value_counts()
```

[96]:
```
spoonbills    4
Cranes        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.

[98]:
```
t = birds.sort_values('age', ascending = False)
print(t)
t = t.sort_values('visits', ascending = True)
print("_________________________________")
print(t)
```

```
             birds  age  visits priority
labels
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
---------------------------------
             birds  age  visits priority
labels
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

[99]:
```
t['priority'] = t['priority'].replace({'yes': 1, 'no': 0})
t
```

[99]:
```
             birds  age  visits  priority
labels
g          plovers  5.5       2         0
j       spoonbills  4.0       2         0
```

```
a           Cranes  3.5       2         1
h           Cranes  NaN       2         1
i       spoonbills  8.0       3         0
e       spoonbills  6.0       3         0
c          plovers  1.5       3         0
b           Cranes  4.0       4         1
f           Cranes  3.0       4         0
d       spoonbills  NaN       4         1
```

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

[100]:
```
t['birds'] = t['birds'].replace({'Cranes' : 'trumpeters'})
t
```

[100]:
```
              birds  age  visits  priority
labels
g           plovers  5.5       2         0
j       spoonbills  4.0       2         0
a       trumpeters  3.5       2         1
h       trumpeters  NaN       2         1
i       spoonbills  8.0       3         0
e       spoonbills  6.0       3         0
c          plovers  1.5       3         0
b       trumpeters  4.0       4         1
f       trumpeters  3.0       4         0
d       spoonbills  NaN       4         1
```

[ ]:
```

```