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

In [89]:
```
import pandas as pd
import numpy as np

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']

birds=pd.DataFrame(data,index=labels)

birds
```

Out[89]:

	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 [90]:
```
birds.describe()
```

Out[90]:

	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 [91]:
```
birds[:2]
```

Out[91]:

	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 [92]:
```
#print(birds[birds.columns[[0, 1]]])
birds[['birds','age']]
```

Out[92]:

	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 [93]:
```
birds[['birds','age','visits']].iloc[[1,2,6]]
#birds[['birds','age','visits']].filter(items=['b','c','g'], axis=0)
```

Out[93]:

	birds	age	visits
b	Cranes	4.0	4
c	plovers	1.5	3
g	plovers	5.5	2

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

In [94]:
```
birds.where(birds['visits']<4).dropna()
```

Out[94]:

	birds	age	visits	priority
a	Cranes	3.5	2.0	yes
c	plovers	1.5	3.0	no
e	spoonbills	6.0	3.0	no
g	plovers	5.5	2.0	no
i	spoonbills	8.0	3.0	no
j	spoonbills	4.0	2.0	no

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

In [95]:
```
birds[['birds','visits']][birds.age.isnull()]
```

Out[95]:

	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

In [96]:
```
birds.loc[(birds['birds'] == 'Cranes') & (birds['age'] < 4)]

#birds[(birds['birds'] == 'Cranes') & (birds['age'] < 4)]
```

Out[96]:

	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 [98]:
```
birds[(birds.age>2) & (birds.age<=4)]
```

Out[98]:

	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 [99]:
```
a=birds.where(birds['birds']=='Cranes').dropna()
print("Total number of visits of Crane :",a['visits'].sum())
```

```
Total number of visits of Crane : 10.0
```

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

In [101]:
```
g=birds.groupby('birds')
print(g.mean())
```

```
             age  visits
birds
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.

In [103]:
```
birds.loc['k'] = ['Hen','100','5','yes']
birds
birds=birds.drop(['k'])
birds
```

Out[103]:

	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

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

In [104]:
```
g=birds.groupby('birds')
g['birds'].count()
```

Out[104]:
```
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.

In [105]:
```
birds=birds.sort_values('age',ascending=False)
birds
birds=birds.sort_values('visits',ascending=True)
birds
```

Out[105]:

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

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

In [106]:
```
birds.sort_index(inplace=True)
birds['priority']=birds['priority'].replace({
    'yes':'1',
    'no':'0'})
birds
```

Out[106]:

	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

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

In [107]:
```
birds['birds']=birds['birds'].replace('Cranes','trumpeters')
birds
```

Out[107]:

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

In [ ]:

In [ ]:

In [ ]: