In [103]:
```
import pandas as pd
data=pd.read_csv("F:\\Desktop\\nba.csv")
df=pd.DataFrame(data)
print(df)
```

```
                Name            Team  Number Position   Age Height  Weight  \
0     Avery Bradley  Boston Celtics     0.0       PG  25.0    6-2   180.0
1       Jae Crowder  Boston Celtics    99.0       SF  25.0    6-6   235.0
2      John Holland  Boston Celtics    30.0       SG  27.0    6-5   205.0
3       R.J. Hunter  Boston Celtics    28.0       SG  22.0    6-5   185.0
4     Jonas Jerebko  Boston Celtics     8.0       PF  29.0   6-10   231.0
..              ...             ...     ...      ...   ...    ...     ...
453   Shelvin Mack       Utah Jazz     8.0       PG  26.0    6-3   203.0
454      Raul Neto       Utah Jazz    25.0       PG  24.0    6-1   179.0
455   Tibor Pleiss       Utah Jazz    21.0        C  26.0    7-3   256.0
456    Jeff Withey       Utah Jazz    24.0        C  26.0    7-0   231.0
457            NaN             NaN     NaN      NaN   NaN    NaN     NaN

                College     Salary
0                 Texas  7730337.0
1             Marquette  6796117.0
2    Boston University        NaN
3        Georgia State  1148640.0
4                   NaN  5000000.0
..                  ...        ...
453              Butler  2433333.0
454                 NaN   900000.0
455                 NaN  2900000.0
456              Kansas   947276.0
457                 NaN        NaN

[458 rows x 9 columns]
```

In [107]:
```
k=df.groupby('Team')
print(k)
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x000001D99A35DAF0>
```

In [108]:
```
k.count()
```

Out[108]:

Team	Name	Number	Position	Age	Height	Weight	College	Salary
Atlanta Hawks	15	15	15	15	15	15	11	15
Boston Celtics	15	15	15	15	15	15	13	14
Brooklyn Nets	15	15	15	15	15	15	13	15
Charlotte Hornets	15	15	15	15	15	15	13	15
Chicago Bulls	15	15	15	15	15	15	12	15
Cleveland Cavaliers	15	15	15	15	15	15	12	14
Dallas Mavericks	15	15	15	15	15	15	12	15
Denver Nuggets	15	15	15	15	15	15	9	14
Detroit Pistons	15	15	15	15	15	15	15	15
Golden State Warriors	15	15	15	15	15	15	12	15
Houston Rockets	15	15	15	15	15	15	11	15
Indiana Pacers	15	15	15	15	15	15	12	15
Los Angeles Clippers	15	15	15	15	15	15	14	15
Los Angeles Lakers	15	15	15	15	15	15	12	15
Memphis Grizzlies	18	18	18	18	18	18	17	14
Miami Heat	15	15	15	15	15	15	11	13
Milwaukee Bucks	16	16	16	16	16	16	14	16
Minnesota Timberwolves	14	14	14	14	14	14	9	13
New Orleans Pelicans	19	19	19	19	19	19	16	19
New York Knicks	16	16	16	16	16	16	11	16
Oklahoma City Thunder	15	15	15	15	15	15	14	15
Orlando Magic	14	14	14	14	14	14	10	14
Philadelphia 76ers	15	15	15	15	15	15	15	14
Phoenix Suns	15	15	15	15	15	15	13	15
Portland Trail Blazers	15	15	15	15	15	15	15	15
Sacramento Kings	15	15	15	15	15	15	13	15
San Antonio Spurs	15	15	15	15	15	15	11	15
Toronto Raptors	15	15	15	15	15	15	10	15
Utah Jazz	15	15	15	15	15	15	10	15
Washington Wizards	15	15	15	15	15	15	13	15

In [99]:
```
l=[1,2,3]
k=l.append(4)
print(l)

```

```
[1, 2, 3, 4]
```

In [ ]:
```

```

In [29]:
```
# k.get_group('Boston Celtics')
```

Out[29]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
5	Amir Johnson	Boston Celtics	90.0	PF	29.0	6-9	240.0	NaN	12000000.0
6	Jordan Mickey	Boston Celtics	55.0	PF	21.0	6-8	235.0	LSU	1170960.0
7	Kelly	Boston	41.0	C	25.0	7-0	238.0	Gonzaga	2165160.0

In [ ]:
```
k.get_group('Philadelphia 76ers')
```

In [31]:
```
# k.get_group(7730337.0  )
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_20036\272345494.py in <module>
----> 1 k.get_group(7730337.0  )

~\anaconda3\lib\site-packages\pandas\core\groupby\groupby.py in get_group(self,
name, obj)
    745         inds = self._get_index(name)
    746         if not len(inds):
--> 747             raise KeyError(name)
    748
    749         return obj._take_with_is_copy(inds, axis=self.axis)

KeyError: 7730337.0
```

In [29]:
```
1
```

Out[29]:

Team	Name	Number	Position	Age	Height	Weight	College	Salary
Atlanta Hawks	15	15	15	15	15	15	11	15
Boston Celtics	15	15	15	15	15	15	13	14
Brooklyn Nets	15	15	15	15	15	15	13	15
Charlotte Hornets	15	15	15	15	15	15	13	15
Chicago Bulls	15	15	15	15	15	15	12	15
Cleveland Cavaliers	15	15	15	15	15	15	12	14
Dallas Mavericks	15	15	15	15	15	15	12	15
Denver Nuggets	15	15	15	15	15	15	9	14
Detroit Pistons	15	15	15	15	15	15	15	15
Golden State Warriors	15	15	15	15	15	15	12	15
Houston Rockets	15	15	15	15	15	15	11	15
Indiana Pacers	15	15	15	15	15	15	12	15
Los Angeles Clippers	15	15	15	15	15	15	14	15
Los Angeles Lakers	15	15	15	15	15	15	12	15
Memphis Grizzlies	18	18	18	18	18	18	17	14
Miami Heat	15	15	15	15	15	15	11	13
Milwaukee Bucks	16	16	16	16	16	16	14	16
Minnesota Timberwolves	14	14	14	14	14	14	9	13
New Orleans Pelicans	19	19	19	19	19	19	16	19
New York Knicks	16	16	16	16	16	16	11	16
Oklahoma City Thunder	15	15	15	15	15	15	14	15
Orlando Magic	14	14	14	14	14	14	10	14
Philadelphia 76ers	15	15	15	15	15	15	15	14
Phoenix Suns	15	15	15	15	15	15	13	15
Portland Trail Blazers	15	15	15	15	15	15	15	15
Sacramento Kings	15	15	15	15	15	15	13	15
San Antonio Spurs	15	15	15	15	15	15	11	15
Toronto Raptors	15	15	15	15	15	15	10	15
Utah Jazz	15	15	15	15	15	15	10	15
Washington Wizards	15	15	15	15	15	15	13	15

In [44]:
```
1 k=df.groupby('Position')
```

Out[44]: <pandas.core.groupby.generic.DataFrameGroupBy object at 0x000001F966B415B0>

In [45]:
```
1 k.count()
```

Out[45]:

Position	Name	Team	Number	Age	Height	Weight	College	Salary
C	78	78	78	78	78	78	49	78
PF	100	100	100	100	100	100	84	97
PG	92	92	92	92	92	92	80	88
SF	85	85	85	85	85	85	71	84
SG	102	102	102	102	102	102	89	99

In [45]:
```
1 k.count()
```

Out[45]:

Position	Name	Team	Number	Age	Height	Weight	College	Salary
C	78	78	78	78	78	78	49	78
PF	100	100	100	100	100	100	84	97
PG	92	92	92	92	92	92	80	88
SF	85	85	85	85	85	85	71	84
SG	102	102	102	102	102	102	89	99

In [46]:
```
# k.get_group('PF')
```

Out[46]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
5	Amir Johnson	Boston Celtics	90.0	PF	29.0	6-9	240.0	NaN	12000000.0
6	Jordan Mickey	Boston Celtics	55.0	PF	21.0	6-8	235.0	LSU	1170960.0
24	Chris McCullough	Brooklyn Nets	1.0	PF	21.0	6-11	200.0	Syracuse	1140240.0
25	Willie Reed	Brooklyn Nets	33.0	PF	26.0	6-10	220.0	Saint Louis	947276.0
...	...	...	...	...	...	...	...	...	...
435	Meyers Leonard	Portland Trail Blazers	11.0	PF	24.0	7-1	245.0	Illinois	3075880.0
441	Noah Vonleh	Portland Trail Blazers	21.0	PF	20.0	6-9	240.0	Indiana	2637720.0
442	Trevor Booker	Utah Jazz	33.0	PF	28.0	6-8	228.0	Clemson	4775000.0
446	Derrick Favors	Utah Jazz	15.0	PF	24.0	6-10	265.0	Georgia Tech	12000000.0
452	Trey Lyles	Utah Jazz	41.0	PF	20.0	6-10	234.0	Kentucky	2239800.0

100 rows × 9 columns

In [52]:
```
data={'name':[1,2,3,4],'age':[14,15,16,17]}
df=pd.DataFrame(data)
print(df)
```

```
   name  age
0     1   14
1     2   15
2     3   16
3     4   17
```

In [60]:
```
# df.rename(columns={'name':'harsha','age':'nani'},inplace=True)
```

In [61]:
```
# print(df)
```

```
   harsha  nani
0       1    14
1       2    15
2       3    16
3       4    17
```

In [62]:
```
#Write a Pandas program to create and display a one-dimensional array-like o
import numpy as np
import pandas as pd
ds=pd.Series([2,4,6,8,10])
print(ds)
```

```
0     2
1     4
2     6
3     8
4    10
dtype: int64
```

In [68]:
```
#Write a Pandas program to convert a Panda module Series to Python list and
ds=pd.Series([10,20,30,40,50])
print(ds)
print(type(ds))
print("convert pandas to python list")
print(ds.tolist())
print(type(ds.tolist()))
```

```
0    10
1    20
2    30
3    40
4    50
dtype: int64
<class 'pandas.core.series.Series'>
convert pandas to python list
[10, 20, 30, 40, 50]
<class 'list'>
```

In [76]:

```
#  Write a Pandas program to add, subtract, multiple and divide two Pandas S
# Sample Series: [2, 4, 6, 8, 10], [1, 3, 5, 7, 9]
ds1=pd.Series([2,4,6,8,10])
ds2=pd.Series([1,3,5,7,9])
# add pandas series arrays
ds=ds1+ds2
print(ds)
#substract two arrays
ds=ds1+ds2
print(ds)
# multiplication of two arrays
ds=ds1*ds2
print(ds)
# divide the series 1 by series 2
ds=ds1/ds2
print(ds)


```

```
0     3
1     7
2    11
3    15
4    19
dtype: int64
0     3
1     7
2    11
3    15
4    19
dtype: int64
0     2
1    12
2    30
3    56
4    90
dtype: int64
0    2.000000
1    1.333333
2    1.200000
3    1.142857
4    1.111111
dtype: float64
```

In [77]:
```
#  Write a Pandas program to compare the elements of the two Pandas Series.
# Sample Series: [2, 4, 6, 8, 10], [1, 3, 5, 7, 10]
ds1=pd.Series([2,4,6,8,10])
ds2=pd.Series([1,3,5,7,10])
print('series1')
print(ds1)
print('series2')
print(ds2)
print("compare the element of series")
print("Equals:")
print(ds1 == ds2)
print("Greater than:")
print(ds1 > ds2)
print("Less than:")
print(ds1 < ds2)

```

```
series1
0     2
1     4
2     6
3     8
4    10
dtype: int64
series2
0     1
1     3
2     5
3     7
4    10
dtype: int64
compare the element of series
Equals:
0    False
1    False
2    False
3    False
4     True
dtype: bool
Greater than:
0     True
1     True
2     True
3     True
4    False
dtype: bool
Less than:
0    False
1    False
2    False
3    False
4    False
dtype: bool
```

In [78]:
```
# Write a Pandas program to convert a dictionary to a Pandas series.
dictpd=pd.Series({'harsha':25,'nani':20,'himanth':100})
print(dictpd)

```

```
harsha      25
nani        20
himanth    100
dtype: int64
```

In [91]:
```
#  Write a Pandas program to convert a NumPy array to a Pandas series. Go to
# Sample Series:
# NumPy array:
import numpy as np
# [10 20 30 40 50]
# d=np.array([10,20,30,40,50])
# print(d)
# ds=pd.Series(d)
# print(dS)
np_array = np.array([10, 20, 30, 40, 50])
print("NumPy array:")
print(np_array)
new_series = pd.Series(np_array)
print("Converted Pandas series:")
print(new_series)
```

```
NumPy array:
[10 20 30 40 50]
Converted Pandas series:
0    10
1    20
2    30
3    40
4    50
dtype: int32
```

In [113]:

```
import pandas as pd
# Write a Pandas program to change the data type of given a column or a Seri

# Sample Series:
# Original Data Series:
# 0 100
# 1 200
# 2 python
# 3 300.12
# 4 400
# dtype: object
# Change the said data type to numeric:
# 0 100.00
# 1 200.00
# 2 NaN
# 3 300.12
# 4 400.00
# dtype: float64
import pandas as pd
s1 = pd.Series(['100', '200', 'python', '300.12', '400'])
print(s1)
s2=pd.to_numeric(s1,errors='coerce' )
print(s2)

```

```
0       100
1       200
2    python
3    300.12
4       400
dtype: object
0    100.00
1    200.00
2       NaN
3    300.12
4    400.00
dtype: float64
```

In [141]:

```
# Write a Pandas program to convert the first column of a DataFrame as a Ser
import pandas as pd
d = {'col1': [1, 2, 3, 4, 7, 11], 'col2': [4, 5, 6, 9, 5, 0], 'col3': [7, 5,
df=pd.DataFrame(data=d)
print(df)
s1 = df.iloc[:,0]
print(s1)

```

```
   col1  col2  col3
0     1     4     7
1     2     5     5
2     3     6     8
3     4     9    12
4     7     5     1
5    11     0    11
0     1
1     2
2     3
3     4
4     7
5    11
Name: col1, dtype: int64
```

In [142]:
```
# Write a Pandas program to create a subset of a given series based on value
import pandas as pd
s = pd.Series([0, 1,2,3,4,5,6,7,8,9,10])
print("Original Data Series:")
print(s)
print("\nSubset of the above Data Series:")
n = 6
new_s = s[s < n]
print(new_s)
```

```
Original Data Series:
0      0
1      1
2      2
3      3
4      4
5      5
6      6
7      7
8      8
9      9
10    10
dtype: int64

Subset of the above Data Series:
0    0
1    1
2    2
3    3
4    4
5    5
dtype: int64
```

In [1]:
```
#PROGRAMS ON PANDAS DATA FRAME
# 1. Write a Pandas program to get the powers of an array values element-wis
# Sample data: {'X':[78,85,96,80,86], 'Y':[84,94,89,83,86],'Z':[86,97,96,72,
import pandas as pd
df = pd.DataFrame({'X':[78,85,96,80,86], 'Y':[84,94,89,83,86],'Z':[86,97,96,
print(df)

```

```
    X   Y   Z
0  78  84  86
1  85  94  97
2  96  89  96
3  80  83  72
4  86  86  83
```

In [44]:

```
# 2. Write a Pandas program to create and display a DataFrame from a specifi
# Sample Python dictionary data and list labels:
import numpy as np
import pandas as pd
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'M
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
df = pd.DataFrame(exam_data , index=labels)
print(df)
```

	name	score	attempts	qualify
a	Anastasia	12.5	1	yes
b	Dima	9.0	3	no
c	Katherine	16.5	2	yes
d	James	NaN	3	no
e	Emily	9.0	2	no
f	Michael	20.0	3	yes
g	Matthew	14.5	1	yes
h	Laura	NaN	1	no
i	Kevin	8.0	2	no
j	Jonas	19.0	1	yes

In [53]:
```
# Write a Pandas program to get the first 3 rows of a given DataFrame.
import pandas as pd
import numpy as np
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'M
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
df=pd.DataFrame(exam_data,index=labels)
print(df)
df.iloc[0:4]
```

```
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
d      James    NaN         3      no
e      Emily    9.0         2      no
f    Michael   20.0         3     yes
g    Matthew   14.5         1     yes
h      Laura    NaN         1      no
i      Kevin    8.0         2      no
j      Jonas   19.0         1     yes
```

Out[53]:

	name	score	attempts	qualify
a	Anastasia	12.5	1	yes
b	Dima	9.0	3	no
c	Katherine	16.5	2	yes
d	James	NaN	3	no

In [54]:
```
# Write a Pandas program to select the 'name' and 'score' columns from the f
import pandas as pd
import numpy as np

exam_data  = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', '
        'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
        'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
        'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no'
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam_data , index=labels)
print("Select specific columns:")
print(df[['name', 'score']])
```

```
Select specific columns:
        name  score
a  Anastasia   12.5
b       Dima    9.0
c  Katherine   16.5
d      James    NaN
e      Emily    9.0
f    Michael   20.0
g    Matthew   14.5
h      Laura    NaN
i      Kevin    8.0
j      Jonas   19.0
```

In [57]:
```
# Write a Pandas program to select the specified columns and rows from a giv
# Select 'name' and 'score' columns in rows 1, 3, 5, 6 from the following da

# Sample DataFrame:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'M
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
df = pd.DataFrame(exam_data , index=labels)
print("Select specific columns and rows:")
print(df.iloc[[1, 3, 5, 6], [1, 3]])
```

```
Select specific columns and rows:
   score qualify
b    9.0      no
d    NaN      no
f   20.0     yes
g   14.5     yes
```

In [59]:
```
# 7. Write a Pandas program to select the rows where the number of attempts
# Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'M
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
df=pd.DataFrame(exam_data,index=labels)
print(df)
```

```
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
d      James    NaN         3      no
e      Emily    9.0         2      no
f    Michael   20.0         3     yes
g    Matthew   14.5         1     yes
h      Laura    NaN         1      no
i      Kevin    8.0         2      no
j      Jonas   19.0         1     yes
```

In [61]:
```
x=df[df['attempts']>2]
print(x)
```

```
      name  score  attempts qualify
b     Dima    9.0         3      no
d    James    NaN         3      no
f  Michael   20.0         3     yes
```

In [62]:
```
x.dropna()
```

Out[62]:

	name	score	attempts	qualify
b	Dima	9.0	3	no
f	Michael	20.0	3	yes

In [68]:
```
# . Write a Pandas program to count the number of rows and columns of a Data

import pandas as pd
import numpy as np# Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'M
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
df=pd.DataFrame(exam_data,index=labels)
print(df)
```

```
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
d      James    NaN         3      no
e      Emily    9.0         2      no
f    Michael   20.0         3     yes
g    Matthew   14.5         1     yes
h      Laura    NaN         1      no
i      Kevin    8.0         2      no
j      Jonas   19.0         1     yes
```

In [79]:
```
total_rows=len(df.axes[0])
print('no of rows'+str(total_rows))
total_columns=len(df.axes[1])
print('no of columns'+str(total_columns))
```

```
no of rows10
no of columns4
```

In [81]:
```
df[df['score'].isna()]
```

Out[81]:

	name	score	attempts	qualify
d	James	NaN	3	no
h	Laura	NaN	1	no

In [85]:
```
# 10. Write a Pandas program to select the rows the score is between 15 and
df[df['score'].between(15,20)]
```

Out[85]:

	name	score	attempts	qualify
c	Katherine	16.5	2	yes
f	Michael	20.0	3	yes
j	Jonas	19.0	1	yes

In [6]:
```
1
2  # Write a Pandas program to change the score in row 'd' to 11.5
3  import pandas as pd
4  import numpy as np# Sample Python dictionary data and list labels:
5  exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'M
6  'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
7  'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
8  'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
9  labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
10 df=pd.DataFrame(exam_data,index=labels)
11 # print(df)
12 df.loc['d','score']=100
13 print(df)
```

```
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
d      James  100.0         3      no
e      Emily    9.0         2      no
f    Michael   20.0         3     yes
g    Matthew   14.5         1     yes
h      Laura    NaN         1      no
i      Kevin    8.0         2      no
j      Jonas   19.0         1     yes
```

In [11]:
```
1  # Write a Pandas program to select the rows where number of attempts in the
2  print(df[(df['attempts']<2)&(df['score']>15)])
```

```
    name  score  attempts qualify
j  Jonas   19.0         1     yes
```

In [18]:
```
1  print(df['attempts'].sum())
```

```
19
```

In [19]:
```
1  # Write a Pandas program to calculate the mean score for each different stud
2  print(df['score'].mean())
3
```

```
23.166666666666668
```

In [37]:
```
# Write a Pandas program to append a new row 'k' to DataFrame with given val
df.loc['k']=[1,'suresh','yes',15.5]
# address = ['Delhi', 'Bangalore', 'Chennai', 'Patna',1,2,3,4,5,6]
# df['ADDRESS']=address
print(df)
```

```
        name   score attempts qualify
a  Anastasia    12.5        1     yes
b       Dima     9.0        3      no
c  Katherine    16.5        2     yes
d      James   100.0        3      no
e      Emily     9.0        2      no
f    Michael    20.0        3     yes
g    Matthew    14.5        1     yes
h      Laura     NaN        1      no
i      Kevin     8.0        2      no
j      Jonas    19.0        1     yes
k          1  suresh      yes    15.5
```

In [38]:
```
import pandas as pd
import numpy as np# Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'M
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
df=pd.DataFrame(exam_data,index=labels)
print(df)
```

```
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
d      James    NaN         3      no
e      Emily    9.0         2      no
f    Michael   20.0         3     yes
g    Matthew   14.5         1     yes
h      Laura    NaN         1      no
i      Kevin    8.0         2      no
j      Jonas   19.0         1     yes
```

In [42]:
```
# adress=[1,2,3,4,5,6,7,8,9,10]
# df['ADDRESS']=[1,2,3,4,5,6,7,8,9,10]
df.insert(2, "Age", [21, 23, 24, 21,1,2,3,4,5,6], True)
print(df)
```

```
        name  score  Age  attempts qualify  ADDRESS
a  Anastasia   12.5   21         1     yes        1
b       Dima    9.0   23         3      no        2
c  Katherine   16.5   24         2     yes        3
d      James    NaN   21         3      no        4
e      Emily    9.0    1         2      no        5
f    Michael   20.0    2         3     yes        6
g    Matthew   14.5    3         1     yes        7
h      Laura    NaN    4         1      no        8
i      Kevin    8.0    5         2      no        9
j      Jonas   19.0    6         1     yes       10
```

In [71]:
```
# Write a Pandas program to sort the DataFrame first by 'name' in descending
import pandas as pd
import numpy as np
exam_data  = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', '
        'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
        'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
        'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no'
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
df = pd.DataFrame(exam_data , index=labels)
print("Orginal rows:")
print(df)
```

```
Orginal rows:
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
d      James    NaN         3      no
e      Emily    9.0         2      no
f    Michael   20.0         3     yes
g    Matthew   14.5         1     yes
h      Laura    NaN         1      no
i      Kevin    8.0         2      no
j      Jonas   19.0         1     yes
```

In [87]:
```
f = df.sort_values(['score','name'],ascending=[False,True])
f
```

Out[87]:

	name	score	attempts	qualify
f	Michael	20.0	3	yes
j	Jonas	19.0	1	yes
c	Katherine	16.5	2	yes
g	Matthew	14.5	1	yes
a	Anastasia	12.5	1	yes
b	Dima	9.0	3	no
e	Emily	9.0	2	no
i	Kevin	8.0	2	no
d	James	NaN	3	no
h	Laura	NaN	1	no

In [97]:
```
df['qualify'].value_counts()
df['qualify'].nunique()
```

Out[97]: 2

In [172]:

```
# Write a Pandas program to replace the 'qualify' column contains the values
import pandas as pd
import numpy as np
exam_data  = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', '
        'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
        'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
        'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no'
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
df = pd.DataFrame(exam_data , index=labels)
print("Original rows:")
# print(df)
print("\nReplace the 'qualify' column contains the values 'yes' and 'no'  wi
df['qualify'] = df['qualify'].map({'yes':True ,'no':False})
print(df)
```

```
Original rows:

Replace the 'qualify' column contains the values 'yes' and 'no'  with True and
False:
        name  score  attempts  qualify
a  Anastasia   12.5         1     True
b       Dima    9.0         3    False
c  Katherine   16.5         2     True
d      James    NaN         3    False
e      Emily    9.0         2    False
f    Michael   20.0         3     True
g    Matthew   14.5         1     True
h      Laura    NaN         1    False
i      Kevin    8.0         2    False
j      Jonas   19.0         1     True
```

In [206]:
```
# Write a Pandas program to change the name 'James' to 'Suresh' in name colu
import pandas as pd
import numpy as np
exam_data  = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', '
        'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
        'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
        'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no'
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
df = pd.DataFrame(exam_data , index=labels)
df['name']=df['name'].replace('James','suresh')
print(df)

```

```
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
d     suresh    NaN         3      no
e      Emily    9.0         2      no
f    Michael   20.0         3     yes
g    Matthew   14.5         1     yes
h      Laura    NaN         1      no
i      Kevin    8.0         2      no
j      Jonas   19.0         1     yes
```

In [209]:
```
# 19. Write a Pandas program to delete the 'attempts' column from the DataFr
df.pop('score')
print(df)

```

```
        name qualify
a  Anastasia     yes
b       Dima      no
c  Katherine     yes
d     suresh      no
e      Emily      no
f    Michael     yes
g    Matthew     yes
h      Laura      no
i      Kevin      no
j      Jonas     yes
```

In [ ]:
```

```