In [1]:

```
import os
import pandas as pd
os.chdir("e:\working folder")
os.getcwd()

sns = pd.read_csv('snsdata.csv')
sns.head(6)
```

Out[1]:

	gradyear	gender	age	friends	basketball	football	soccer	softball	volleyball	swimming	..
0	2006	M	18.982	7	0	0	0	0	0	0	..
1	2006	F	18.801	0	0	1	0	0	0	0	..
2	2006	M	18.335	69	0	1	0	0	0	0	..
3	2006	F	18.875	0	0	0	0	0	0	0	..
4	2006	NaN	18.995	10	0	0	0	0	0	0	..
5	2006	F	NaN	142	0	0	0	0	0	0	..

6 rows × 40 columns

In [2]:

```
print("Average age is %d"%sns['age'].mean(skipna=True))
print("Is there any null value in age data?", pd.isnull(sns['age']).sum()>0)

```

```
Average age is 17
Is there any null value in age data? True
```

In [3]:

```python
#how to fill NAs with zero


sns.fillna(0, inplace=True)
averageval=sns['age'].mean(skipna=True)
averageval=round(averageval,2)
print(averageval)
print("Now the average age is %d" %averageval)
sns.head(6)
```

```
14.94
Now the average age is 14
```

Out[3]:

	gradyear	gender	age	friends	basketball	football	soccer	softball	volleyball	swimming	..
0	2006	M	18.982	7	0	0	0	0	0	0	..
1	2006	F	18.801	0	0	1	0	0	0	0	..
2	2006	M	18.335	69	0	1	0	0	0	0	..
3	2006	F	18.875	0	0	0	0	0	0	0	..
4	2006	0	18.995	10	0	0	0	0	0	0	..
5	2006	F	0.000	142	0	0	0	0	0	0	..

6 rows × 40 columns

In [4]:

```python
#Replace missing values and NAs with some numbers

filename = "snsdata.csv"
sns1 = pd.read_csv("snsdata.csv")
sns1.fillna({'age': 19, 'gender': 'F'}, inplace=True)
sns1.head(6)
```

Out[4]:

	gradyear	gender	age	friends	basketball	football	soccer	softball	volleyball	swimming	..
0	2006	M	18.982	7	0	0	0	0	0	0	..
1	2006	F	18.801	0	0	1	0	0	0	0	..
2	2006	M	18.335	69	0	1	0	0	0	0	..
3	2006	F	18.875	0	0	0	0	0	0	0	..
4	2006	F	18.995	10	0	0	0	0	0	0	..
5	2006	F	19.000	142	0	0	0	0	0	0	..

6 rows × 40 columns

In [5]:

```python
#Replace NAs with mean

filename = "snsdata.csv"
sns2 = pd.read_csv("snsdata.csv")
sns2.fillna({'age': sns2['age'].mean(skipna=True)}, inplace=True)
sns2['age'].head(6)
```

Out[5]:

```
0    18.98200
1    18.80100
2    18.33500
3    18.87500
4    18.99500
5    17.99395
Name: age, dtype: float64
```

In [6]:

```python
#replace NAs with the median in age

filename = "snsdata.csv"
sns3 = pd.read_csv("snsdata.csv")
sns3.fillna({'age': sns3['age'].median(skipna=True)}, inplace=True)
sns2['age'].head(6)
```

Out[6]:

```
0    18.98200
1    18.80100
2    18.33500
3    18.87500
4    18.99500
5    17.99395
Name: age, dtype: float64
```

In [7]:

```python
#to check the number of rows and columns

sns.shape
```

Out[7]:

```
(30000, 40)
```

In [8]:

```
#To check the data type of the data frame

sns.dtypes
```

Out[8]:

```
gradyear          int64
gender           object
age             float64
friends           int64
basketball        int64
football          int64
soccer            int64
softball          int64
volleyball        int64
swimming          int64
cheerleading      int64
baseball          int64
tennis            int64
sports            int64
cute              int64
sex               int64
sexy              int64
hot               int64
kissed            int64
dance             int64
band              int64
marching          int64
music             int64
rock              int64
god               int64
church            int64
jesus             int64
bible             int64
hair              int64
dress             int64
blonde            int64
mall              int64
shopping          int64
clothes           int64
hollister         int64
abercrombie       int64
die               int64
death             int64
drunk             int64
drugs             int64
dtype: object
```

In [9]:

```python
#to write on the numeric columns of the data frame

num_cols=['int16', 'int32', 'int64', 'float16', 'float32', 'float64']
sns.select_dtypes(include=num_cols).head(6)
```

Out[9]:

	gradyear	age	friends	basketball	football	soccer	softball	volleyball	swimming	cheerleadi
0	2006	18.982	7	0	0	0	0	0	0	
1	2006	18.801	0	0	1	0	0	0	0	
2	2006	18.335	69	0	1	0	0	0	0	
3	2006	18.875	0	0	0	0	0	0	0	
4	2006	18.995	10	0	0	0	0	0	0	
5	2006	0.000	142	0	0	0	0	0	0	

6 rows × 39 columns

In [10]:

```
#who to return only categorical columns from the given data frame
#dataframe.select_dtypes(include=none, exclude=none)

sns.select_dtypes(include=['object'])
```

Out[10]:

	gender
0	M
1	F
2	M
3	F
4	0
5	F
6	F
7	M
8	F
9	F
10	F
11	F
12	F
13	0
14	F
15	0
16	0
17	F
18	F
19	F
20	F
21	M
22	F
23	F
24	F
25	M
26	F
27	M
28	F
29	F
...	...
29970	0

	gender
29971	0
29972	F
29973	M
29974	F
29975	F
29976	F
29977	F
29978	F
29979	F
29980	F
29981	F
29982	F
29983	M
29984	F
29985	M
29986	M
29987	M
29988	F
29989	F
29990	M
29991	F
29992	M
29993	F
29994	M
29995	M
29996	M
29997	M
29998	M
29999	F

30000 rows × 1 columns

In [11]:

```
#How to check the name of numeric columns

sns._get_numeric_data().columns
```

Out[11]:

```
Index(['gradyear', 'age', 'friends', 'basketball', 'football', 'soccer',
       'softball', 'volleyball', 'swimming', 'cheerleading', 'baseball',
       'tennis', 'sports', 'cute', 'sex', 'sexy', 'hot', 'kissed', 'dance',
       'band', 'marching', 'music', 'rock', 'god', 'church', 'jesus', 'bible',
       'hair', 'dress', 'blonde', 'mall', 'shopping', 'clothes', 'hollister',
       'abercrombie', 'die', 'death', 'drunk', 'drugs'],
      dtype='object')
```

In [12]:

```
#How to check the name of numeric columns

sns.select_dtypes(exclude=num_cols).columns
```

Out[12]:

```
Index(['gender'], dtype='object')
```

In [13]:

```
#find the SD of the numeric columns


sns._get_numeric_data().std()
```

Out[13]:

```
gradyear         1.118053
age              9.842131
friends         36.530877
basketball       0.804708
football         0.705357
soccer           0.917226
softball         0.739707
volleyball       0.639943
swimming         0.516990
cheerleading     0.514333
baseball         0.521726
tennis           0.516961
sports           0.471080
cute             0.802441
sex              1.123504
sexy             0.528209
hot              0.479145
kissed           0.509338
dance            1.162574
band             1.118786
marching         0.287091
music            1.252366
rock             0.720375
god              1.343226
church           0.834028
jesus            0.581709
bible            0.204645
hair             1.097958
dress            0.449436
blonde           1.942319
mall             0.695758
shopping         0.724391
clothes          0.472640
hollister        0.346779
abercrombie      0.279555
die              0.624516
death            0.436796
drunk            0.399125
drugs            0.345522
dtype: float64
```

In [14]:

```
sns[sns._get_numeric_data().columns].std()
```

Out[14]:

```
gradyear         1.118053
age              9.842131
friends         36.530877
basketball       0.804708
football         0.705357
soccer           0.917226
softball         0.739707
volleyball       0.639943
swimming         0.516990
cheerleading     0.514333
baseball         0.521726
tennis           0.516961
sports           0.471080
cute             0.802441
sex              1.123504
sexy             0.528209
hot              0.479145
kissed           0.509338
dance            1.162574
band             1.118786
marching         0.287091
music            1.252366
rock             0.720375
god              1.343226
church           0.834028
jesus            0.581709
bible            0.204645
hair             1.097958
dress            0.449436
blonde           1.942319
mall             0.695758
shopping         0.724391
clothes          0.472640
hollister        0.346779
abercrombie      0.279555
die              0.624516
death            0.436796
drunk            0.399125
drugs            0.345522
dtype: float64
```

In [15]:

```
#Describing data
sns.describe()
```

Out[15]:

	gradyear	age	friends	basketball	football	soccer	
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.
mean	2007.500000	14.943375	30.179467	0.267333	0.252300	0.222767	0.
std	1.118053	9.842131	36.530877	0.804708	0.705357	0.917226	0.
min	2006.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.
25%	2006.750000	15.647000	3.000000	0.000000	0.000000	0.000000	0.
50%	2007.500000	16.890000	20.000000	0.000000	0.000000	0.000000	0.
75%	2008.250000	18.067000	44.000000	0.000000	0.000000	0.000000	0.
max	2009.000000	106.927000	830.000000	24.000000	15.000000	27.000000	17.

8 rows × 39 columns

In [122]:

```
#Displaying data with filter

sns[sns.gradyear == 2007].describe()
```

Out[122]:

	gradyear	age	friends	basketball	football	soccer	softball	
count	7500.0	7500.000000	7500.000000	7500.000000	7500.000000	7500.000000	7500.000000	7
mean	2007.0	15.485609	30.738133	0.232800	0.239867	0.207333	0.141867	
std	0.0	9.601350	38.151804	0.763336	0.690889	0.864202	0.683278	
min	2007.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	2007.0	17.256250	4.000000	0.000000	0.000000	0.000000	0.000000	
50%	2007.0	17.591000	20.000000	0.000000	0.000000	0.000000	0.000000	
75%	2007.0	17.936000	44.000000	0.000000	0.000000	0.000000	0.000000	
max	2007.0	106.927000	830.000000	24.000000	11.000000	15.000000	13.000000	

8 rows × 39 columns

In [123]:

```
sns[sns.gradyear == sns['gradyear'].max()].describe()
```

Out[123]:

	gradyear	age	friends	basketball	football	soccer	softball	
count	7500.0	7500.000000	7500.000000	7500.000000	7500.000000	7500.000000	7500.000000	7
mean	2009.0	13.683081	33.023200	0.351467	0.263600	0.300667	0.197333	
std	0.0	10.767207	38.721648	0.881689	0.724282	1.112728	0.818830	
min	2009.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	2009.0	15.272000	5.000000	0.000000	0.000000	0.000000	0.000000	
50%	2009.0	15.655000	22.000000	0.000000	0.000000	0.000000	0.000000	
75%	2009.0	16.005000	48.000000	0.000000	0.000000	0.000000	0.000000	
max	2009.0	106.927000	792.000000	10.000000	15.000000	27.000000	15.000000	

8 rows × 39 columns

In [127]:

```
#Display stats of selected Rows & Columns

sns[['gradyear', 'age','friends']][sns.gradyear==sns['gradyear'].max()]
```

Out[127]:

	gradyear	age	friends
22500	2009	0.000	103
22501	2009	15.877	0
22502	2009	0.000	53
22503	2009	16.175	11
22504	2009	0.000	24
22505	2009	16.301	27
22506	2009	16.145	16
22507	2009	15.792	3
22508	2009	16.550	121
22509	2009	16.014	0
22510	2009	15.474	30
22511	2009	15.737	0
22512	2009	15.340	0
22513	2009	0.000	50
22514	2009	15.277	0
22515	2009	0.000	0
22516	2009	16.315	38
22517	2009	15.562	27
22518	2009	15.066	1
22519	2009	0.000	39
22520	2009	15.546	24
22521	2009	16.104	23
22522	2009	16.129	28
22523	2009	16.203	184
22524	2009	15.420	1
22525	2009	15.307	79
22526	2009	0.000	38
22527	2009	16.178	23
22528	2009	15.266	8
22529	2009	15.316	27
...	...	...	...
29970	2009	0.000	0
29971	2009	15.811	13

	gradyear	age	friends
29972	2009	15.885	73
29973	2009	16.148	8
29974	2009	16.063	2
29975	2009	15.647	0
29976	2009	16.172	16
29977	2009	16.238	39
29978	2009	15.929	28
29979	2009	15.387	13
29980	2009	0.000	3
29981	2009	15.674	0
29982	2009	16.227	13
29983	2009	16.835	0
29984	2009	15.644	11
29985	2009	16.249	0
29986	2009	16.214	51
29987	2009	16.400	13
29988	2009	16.230	2
29989	2009	0.000	0
29990	2009	15.699	0
29991	2009	0.000	229
29992	2009	0.000	7
29993	2009	0.000	0
29994	2009	15.195	33
29995	2009	16.115	0
29996	2009	15.792	1
29997	2009	15.784	0
29998	2009	16.378	0
29999	2009	18.724	3

7500 rows × 3 columns

```
In [ ]:
```