In [1]:
```
import pandas as pd
```

In [2]:
```
text=pd.read_csv('dirtydata.csv')
pd.set_option('display.max_rows', 15)
```

In [3]:
```
text
```

Out[3]:

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
...	...	...	...	...	...
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	-280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

32 rows × 5 columns

In [4]:
```
text.isnull()
```

Out[4]:

	Duration	Date	Pulse	Maxpulse	Calories
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
...	...	...	...	...	...
27	False	False	False	False	False
28	False	False	False	False	True
29	False	False	False	False	False
30	False	False	False	False	False
31	False	False	False	False	False

32 rows × 5 columns

In [5]:
```
text['Calories'].isnull().sum()/text['Calories'].sum()*100
```

Out[5]:
```
0.025061400431056088
```

In [6]:
```
x=text['Calories'].mean()
```

In [7]:
```
text['Calories'].fillna(x,inplace=True)
```

In [8]:
```
text
```

Out[8]:

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.100000
1	60	'2020/12/02'	117	145	479.000000
2	60	'2020/12/03'	103	135	340.000000
3	45	'2020/12/04'	109	175	282.400000
4	45	'2020/12/05'	117	148	406.000000
...	...	...	...	...	...
27	60	'2020/12/27'	92	118	241.000000
28	60	'2020/12/28'	103	132	266.013333
29	60	'2020/12/29'	100	132	-280.000000
30	60	'2020/12/30'	102	129	380.300000
31	60	'2020/12/31'	92	115	243.000000

32 rows × 5 columns

In [9]:
```
text.shape  #function to check dimension of dataset
```

Out[9]: (32, 5)

In [10]:
```
text.dtypes #check type
```

Out[10]:
```
Duration      int64
Date         object
Pulse         int64
Maxpulse      int64
Calories    float64
dtype: object
```

In [11]:
```
text.isnull().sum()
```

Out[11]:
```
Duration    0
Date        1
Pulse       0
Maxpulse    0
Calories    0
dtype: int64
```

In [12]:
```
text.describe()
```

Out[12]:

	Duration	Pulse	Maxpulse	Calories
count	32.000000	32.000000	32.000000	32.000000
mean	68.437500	103.500000	128.500000	266.013333
std	70.039591	7.832933	12.998759	159.469153
min	30.000000	90.000000	101.000000	-300.000000
25%	60.000000	100.000000	120.000000	249.000000
50%	60.000000	102.500000	127.500000	278.500000
75%	60.000000	106.500000	132.250000	341.325000
max	450.000000	130.000000	175.000000	479.000000

In [13]:
```
text['Calories']=text['Calories'].abs() #replace negative with absolute i.e. positive
```

In [14]:
```
text
```

Out[14]:

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.100000
1	60	'2020/12/02'	117	145	479.000000
2	60	'2020/12/03'	103	135	340.000000
3	45	'2020/12/04'	109	175	282.400000
4	45	'2020/12/05'	117	148	406.000000
...	...	...	...	...	...
27	60	'2020/12/27'	92	118	241.000000
28	60	'2020/12/28'	103	132	266.013333
29	60	'2020/12/29'	100	132	280.000000
30	60	'2020/12/30'	102	129	380.300000
31	60	'2020/12/31'	92	115	243.000000

32 rows × 5 columns

In [15]:
```
text.round(2) #rounding 2 decimal
```

Out[15]:

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.10
1	60	'2020/12/02'	117	145	479.00
2	60	'2020/12/03'	103	135	340.00
3	45	'2020/12/04'	109	175	282.40
4	45	'2020/12/05'	117	148	406.00
...	...	...	...	...	...
27	60	'2020/12/27'	92	118	241.00
28	60	'2020/12/28'	103	132	266.01
29	60	'2020/12/29'	100	132	280.00
30	60	'2020/12/30'	102	129	380.30
31	60	'2020/12/31'	92	115	243.00

32 rows × 5 columns

In [16]:
```
text1=text
```

In [17]:
```
text1['Calories']=text1['Calories'].astype(int)
```

In [18]:
```
text1
```

Out[18]:

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409
1	60	'2020/12/02'	117	145	479
2	60	'2020/12/03'	103	135	340
3	45	'2020/12/04'	109	175	282
4	45	'2020/12/05'	117	148	406
...	...	...	...	...	...
27	60	'2020/12/27'	92	118	241
28	60	'2020/12/28'	103	132	266
29	60	'2020/12/29'	100	132	280
30	60	'2020/12/30'	102	129	380
31	60	'2020/12/31'	92	115	243

32 rows × 5 columns

In [19]:
```
text.dropna(subset='Date',inplace=True)
```

In [20]:
```
text
```

Out[20]:

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409
1	60	'2020/12/02'	117	145	479
2	60	'2020/12/03'	103	135	340
3	45	'2020/12/04'	109	175	282
4	45	'2020/12/05'	117	148	406
...	...	...	...	...	...
27	60	'2020/12/27'	92	118	241
28	60	'2020/12/28'	103	132	266
29	60	'2020/12/29'	100	132	280
30	60	'2020/12/30'	102	129	380
31	60	'2020/12/31'	92	115	243

31 rows × 5 columns

In [ ]:
```

```

In [21]:
```
text['Date']=pd.to_datetime(text['Date'])
```

In [22]:
```
text
```

Out[22]:

	Duration	Date	Pulse	Maxpulse	Calories
0	60	2020-12-01	110	130	409
1	60	2020-12-02	117	145	479
2	60	2020-12-03	103	135	340
3	45	2020-12-04	109	175	282
4	45	2020-12-05	117	148	406
...	...	...	...	...	...
27	60	2020-12-27	92	118	241
28	60	2020-12-28	103	132	266
29	60	2020-12-29	100	132	280
30	60	2020-12-30	102	129	380
31	60	2020-12-31	92	115	243

31 rows × 5 columns

In [23]:
```
text.loc[7,'Duration']=45
```

In [24]:
```
text.duplicated()
```

Out[24]:
```
0     False
1     False
2     False
3     False
4     False
      ...
27    False
28    False
29    False
30    False
31    False
Length: 31, dtype: bool
```

In [25]:
```
text.drop_duplicates(inplace=True)
```

In [26]:
```
text.duplicated().sum()
```

Out[26]:
```
0
```

In [27]:
```
newtext=pd.read_csv('nba.csv')
```

In [28]:
```
newtext
```

Out[28]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0	PG	25	2-Jun	180	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99	SF	25	6-Jun	235	Marquette	6796117.0
2	John Holland	Boston Celtics	30	SG	27	5-Jun	205	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28	SG	22	5-Jun	185	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8	PF	29	10-Jun	231	NaN	5000000.0
...	...	...	...	...	...	...	...	...	...
452	Trey Lyles	Utah Jazz	41	PF	20	10-Jun	234	Kentucky	2239800.0
453	Shelvin Mack	Utah Jazz	8	PG	26	3-Jun	203	Butler	2433333.0
454	Raul Neto	Utah Jazz	25	PG	24	1-Jun	179	NaN	900000.0
455	Tibor Pleiss	Utah Jazz	21	C	26	3-Jul	256	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24	C	26	Jul-00	231	Kansas	947276.0

457 rows × 9 columns

In [29]:
```
newtext.isnull().sum()
```

Out[29]:
```
Name         0
Team         0
Number       0
Position     0
Age          0
Height       0
Weight       0
College     84
Salary      11
dtype: int64
```

In [30]:
```
newtext['Position'].unique()
```

Out[30]:
```
array(['PG', 'SF', 'SG', 'PF', 'C'], dtype=object)
```

In [31]:
```
newtext['Position'].value_counts()
```

Out[31]:
```
SG    102
PF    100
PG     92
SF     85
C      78
Name: Position, dtype: int64
```

In [34]:
```
newtext['pos']=newtext['Position'].replace(['PG', 'SF', 'SG', 'PF', 'C'],[1,2,3,4,5]) //Category to numerical
```

```
  Cell In[34], line 1
    newtext['pos']=newtext['Position'].replace(['PG', 'SF', 'SG', 'PF', 'C'],[1,2,3,4,5]) //Category to numerical
                                                                                          ^
SyntaxError: invalid syntax
```

In [ ]:
```
newtext
```

Category to Quantitative

In [ ]:
```
from sklearn import preprocessing as pp
```

In [ ]:
```
l_e=pp.LabelEncoder()
```

In [ ]:
```
newtext['Position']=l_e.fit_transform(newtext['Position'])
```

In [ ]:
```
newtext
```

Quantitative to Category

In [ ]:
```
category=pd.cut(newtext.Age,bins=[19,25,30,35,45],labels=['A','B','C','D'])
```

In [ ]:
```
newtext.insert(5,'Age_Group',category)
```

In [ ]:
```
newtext
```

In [ ]:
```
newtext.to_csv('preprocessnewnba.csv')
```

In [ ]:
```
data=pd.read_csv('A1_ALCHOHOL.csv')
```

In [ ]:
```
data
```

In [ ]:
```
data.columns=[c.strip() for c in data.columns]
```

In [ ]:
```
data.columns
```

In [ ]:
```
data.isnull().sum()
```

In [ ]:
```
data['Deaths']=data['Deaths'].abs()
```

In [ ]:
```
data
```

In [ ]:
```
data['Alcohol']=data['Alcohol'].abs()
```

In [ ]:
```
data
```

In [ ]:
```
data['Liver'].fillna(data['Liver'].mean(),inplace=True)
```

In [ ]:
```
data
```

In [ ]:
```
data['Liver']=data['Liver'].round(2)
```

In [ ]:
```
data
```

In [ ]:
```
data['Heart'].fillna(data['Heart'].median(),inplace=True)
```

In [ ]:
```
data
```

In [ ]:
```
data.loc[10,'Alcohol']=7.90
```

In [ ]:
```
data
```

In [ ]:
```
data.loc[6,'Heart']=136.0
```

In [ ]:
```
data
```

In [ ]:
```

```

In [ ]:
```

```