ASSIGNMENT 2

Data Wrangling II

Create an "Academic performance" dataset of students and perform the following operations using Python.

1.Scan all variables for missing values and inconsistencies. If there are missing values and/or inconsistencies, use any of the suitable techniques to deal with them.

2. Scan all numeric variables for outliers. If there are outliers, use any of the suitable techniques to deal with them.

3. Apply data transformations on at least one of the variables. The purpose of this transformation should be one of the following reasons: to change the scale for better understanding of the variable, to convert a non-linear relation into a linear one, or to decrease the skewness and convert the distribution into a normal distribution.

Reason and document your approach properly.

In [97]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import math
```

Creating the dataset

In [65]:
```
rollno = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
name = ["a", "b", "c", "d", "e", "f", "g", "h", "i", "j", np.nan, np.nan, "k", "l",
marks = [40, 23, 50, 78, 48, 89, 90, 67, 84, 96, 76, np.nan, 97, np.nan, 65]
grade = ["F", "F", "P", "P", "P", "P", "P", "P", "P", "P", "P", "F", "P", np.nan, n
```

In [66]:
```
df = pd.DataFrame({"rollno" : rollno, "name" : name, "marks" : marks, "grade" : gra
```

In [67]:
```
df
```

Out[67]:

	rollno	name	marks	grade
0	1	a	40.0	F
1	2	b	23.0	F
2	3	c	50.0	P
3	4	d	78.0	P
4	5	e	48.0	P
5	6	f	89.0	P
6	7	g	90.0	P
7	8	h	67.0	P
8	9	i	84.0	P
9	10	j	96.0	P
10	11	NaN	76.0	P
11	12	NaN	NaN	F
12	13	k	97.0	P
13	14	l	NaN	NaN
14	15	m	65.0	NaN

Dataset Statistics

In [68]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   rollno  15 non-null     int64  
 1   name    13 non-null     object 
 2   marks   13 non-null     float64
 3   grade   13 non-null     object 
dtypes: float64(1), int64(1), object(2)
memory usage: 612.0+ bytes
```

In [69]:
```
df.describe()
```

Out[69]:

	rollno	marks
count	15.000000	13.000000
mean	8.000000	69.461538
std	4.472136	23.247277
min	1.000000	23.000000
25%	4.500000	50.000000
50%	8.000000	76.000000
75%	11.500000	89.000000
max	15.000000	97.000000

In [70]:
```
df.dtypes
```

Out[70]:
```
rollno      int64
name       object
marks     float64
grade      object
dtype: object
```

In [71]:
```
df.columns
```

Out[71]:
```
Index(['rollno', 'name', 'marks', 'grade'], dtype='object')
```

In [72]:
```
df.isna().sum()
```

Out[72]:
```
rollno    0
name      2
marks     2
grade     2
dtype: int64
```

In [73]:
```
df.to_csv("academic_performance.csv")
```

Null values

In [74]:
```
df.isna().sum()
```

Out[74]:
```
rollno    0
name      2
marks     2
grade     2
dtype: int64
```

In [75]:
```
df["marks"] = df["marks"].fillna(df["marks"].mean())
```

In [76]:
```
df
```

Out[76]:

	rollno	name	marks	grade
0	1	a	40.000000	F
1	2	b	23.000000	F
2	3	c	50.000000	P
3	4	d	78.000000	P
4	5	e	48.000000	P
5	6	f	89.000000	P
6	7	g	90.000000	P
7	8	h	67.000000	P
8	9	i	84.000000	P
9	10	j	96.000000	P
10	11	NaN	76.000000	P
11	12	NaN	69.461538	F
12	13	k	97.000000	P
13	14	l	69.461538	NaN
14	15	m	65.000000	NaN

In [77]:
```
def fun1(value):
    return int(math.floor(value))
```

In [78]:
```
df["marks"] = df["marks"].apply(fun1)
```

In [79]:
```
df
```

Out[79]:

	rollno	name	marks	grade
0	1	a	40	F
1	2	b	23	F
2	3	c	50	P
3	4	d	78	P
4	5	e	48	P
5	6	f	89	P
6	7	g	90	P
7	8	h	67	P
8	9	i	84	P
9	10	j	96	P
10	11	NaN	76	P
11	12	NaN	69	F
12	13	k	97	P
13	14	l	69	NaN
14	15	m	65	NaN

In [80]:
```
df = df[df['name'].notna()]
```

In [81]:
```
df
```

Out[81]:

	rollno	name	marks	grade
0	1	a	40	F
1	2	b	23	F
2	3	c	50	P
3	4	d	78	P
4	5	e	48	P
5	6	f	89	P
6	7	g	90	P
7	8	h	67	P
8	9	i	84	P
9	10	j	96	P
12	13	k	97	P
13	14	l	69	NaN
14	15	m	65	NaN

In [82]:
```
for index, row in df.iterrows():
    # print(row['marks'], row['grade'])
    if (row['marks'] > 40):
        df.loc[index, 'grade'] = 'P'
    else:
        df.loc[index, 'grade'] = 'F'
```

In [83]:
```
df
```

Out[83]:

	rollno	name	marks	grade
0	1	a	40	F
1	2	b	23	F
2	3	c	50	P
3	4	d	78	P
4	5	e	48	P
5	6	f	89	P
6	7	g	90	P
7	8	h	67	P
8	9	i	84	P
9	10	j	96	P
12	13	k	97	P
13	14	l	69	P
14	15	m	65	P

Outliers

In [84]:
```
first_outlier = [16, 'n', 200, 'P']
second_outlier = [17, 'o', -100, 'F']
```

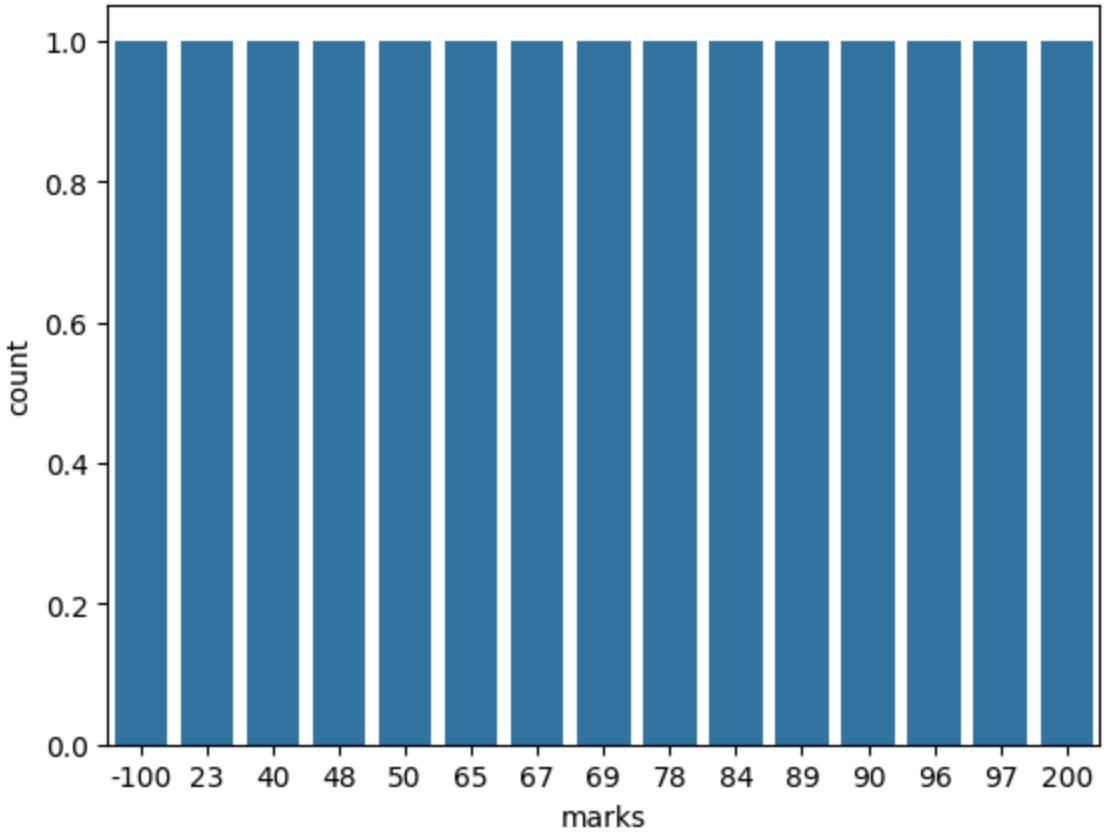
In [85]:
```
df.loc[15] = first_outlier
df.loc[16] = second_outlier
```

In [86]:
```
df
```

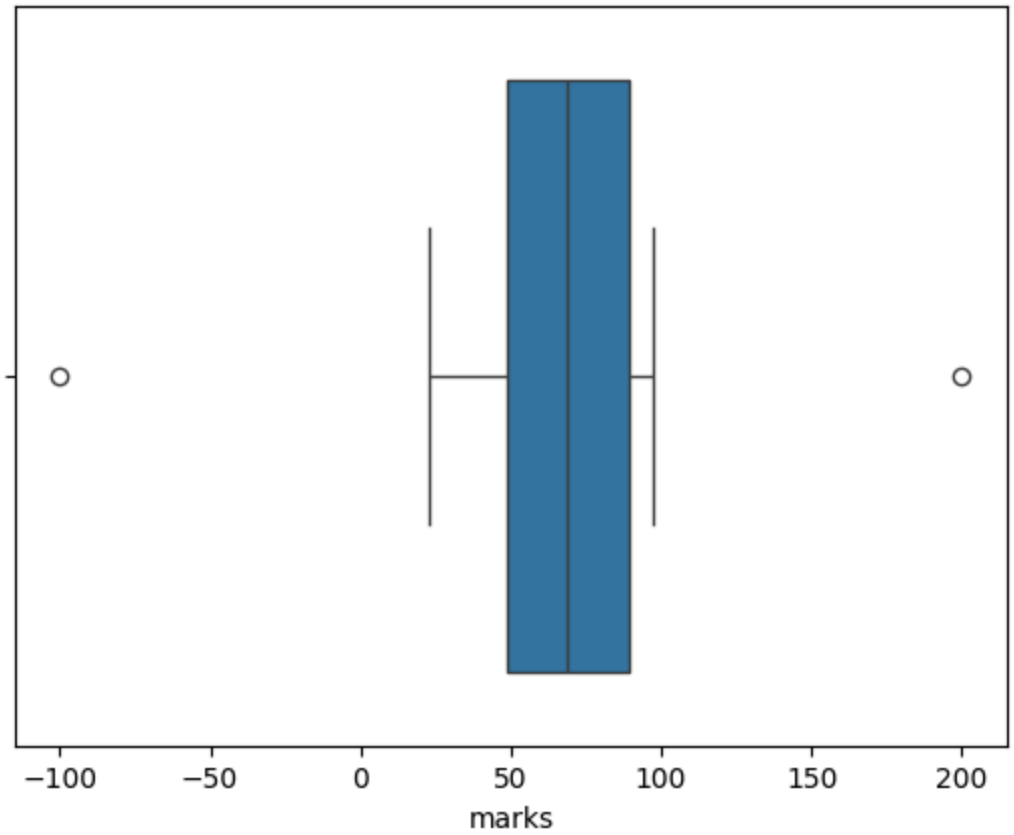
Out[86]:

	rollno	name	marks	grade
0	1	a	40	F
1	2	b	23	F
2	3	c	50	P
3	4	d	78	P
4	5	e	48	P
5	6	f	89	P
6	7	g	90	P
7	8	h	67	P
8	9	i	84	P
9	10	j	96	P
12	13	k	97	P
13	14	l	69	P
14	15	m	65	P
15	16	n	200	P
16	17	o	-100	F

In [87]:
```
sns.countplot(data=df, x=df['marks']);
```



In [88]:
```
sns.boxplot(data=df, x='marks');
```



In [89]:
```
from matplotlib.cbook import boxplot_stats
outliers = boxplot_stats(df['marks']).pop(0)['fliers']
outliers
```

Out[89]:
```
array([-100,  200])
```

In [90]:
```
df
```

Out[90]:

	rollno	name	marks	grade
0	1	a	40	F
1	2	b	23	F
2	3	c	50	P
3	4	d	78	P
4	5	e	48	P
5	6	f	89	P
6	7	g	90	P
7	8	h	67	P
8	9	i	84	P
9	10	j	96	P
12	13	k	97	P
13	14	l	69	P
14	15	m	65	P
15	16	n	200	P
16	17	o	-100	F

In [91]:
```
df = df.drop([15,16], axis=0)
```

In [92]:
```
df
```

Out[92]:

	rollno	name	marks	grade
0	1	a	40	F
1	2	b	23	F
2	3	c	50	P
3	4	d	78	P
4	5	e	48	P
5	6	f	89	P
6	7	g	90	P
7	8	h	67	P
8	9	i	84	P
9	10	j	96	P
12	13	k	97	P
13	14	l	69	P
14	15	m	65	P

Scaling the marks column

In [93]:
```
from sklearn.preprocessing import MinMaxScaler
```

In [94]:
```
scaler = MinMaxScaler()
```

In [95]:
```
df[['marks']] = scaler.fit_transform(df[['marks']])
```

In [96]:
```
df
```

Out[96]:

	rollno	name	marks	grade
0	1	a	0.229730	F
1	2	b	0.000000	F
2	3	c	0.364865	P
3	4	d	0.743243	P
4	5	e	0.337838	P
5	6	f	0.891892	P
6	7	g	0.905405	P
7	8	h	0.594595	P
8	9	i	0.824324	P
9	10	j	0.986486	P
12	13	k	1.000000	P
13	14	l	0.621622	P
14	15	m	0.567568	P