```
import pandas as pd
import numpy as np

df = pd.read_csv("Academic_Performance.csv")
```

Checking for missing values and inconsistencies.

```
missed = df.isnull().sum()
print(missed)
```

```
STUDENT_ID              0
GENDER                 22
PLACEMENT              15
HONOR_OPTED_OR_NOT     14
EDUCATION_TYPE         15
ACADEMIC_PROGRAM       34
COURSE 1 MARKS         11
COURSE 2 MARKS          8
COURSE 3 MARKS         14
COURSE 4 MARKS         14
COURSE 5 MARKS         22
PERCENTILE              0
OVEARLL_GRADE           0
dtype: int64
```

```
df.dtypes
```

```
STUDENT_ID              object
GENDER                  object
PLACEMENT               object
HONOR_OPTED_OR_NOT      object
EDUCATION_TYPE          object
ACADEMIC_PROGRAM        object
COURSE 1 MARKS         float64
COURSE 2 MARKS         float64
COURSE 3 MARKS         float64
COURSE 4 MARKS         float64
COURSE 5 MARKS         float64
PERCENTILE               int64
OVEARLL_GRADE           object
dtype: object
```

for missing values and inconsistencies.

```
df["COURSE 1 MARKS"]=df["COURSE 1 MARKS"].replace(np.NaN,df["COURSE 1
MARKS"].median())
df["COURSE 1 MARKS"].isnull().sum()
```

```
0
```

```
df["COURSE 2 MARKS"]=df["COURSE 2 MARKS"].replace(np.NaN,df["COURSE 2
MARKS"].mean())
df["COURSE 2 MARKS"].isnull().sum()
df["COURSE 3 MARKS"]=df["COURSE 3 MARKS"].replace(np.NaN,df["COURSE 3
MARKS"].mean())
df["COURSE 3 MARKS"].isnull().sum()
df["COURSE 4 MARKS"]=df["COURSE 4 MARKS"].replace(np.NaN,df["COURSE 4
MARKS"].mean())
df["COURSE 4 MARKS"].isnull().sum()
df["COURSE 5 MARKS"]=df["COURSE 5 MARKS"].replace(np.NaN,df["COURSE 5
MARKS"].mean())
df["COURSE 5 MARKS"].isnull().sum()
```

```
0
```

```
df.isnull().sum()
```

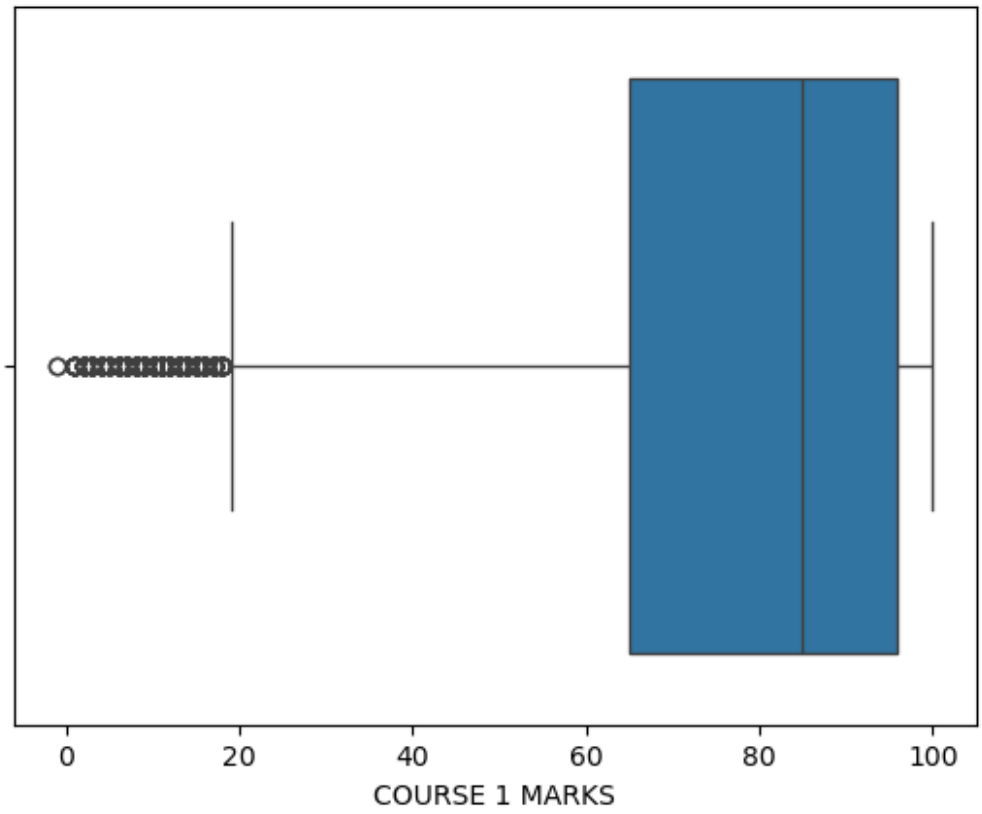
```
STUDENT_ID             0
GENDER                 0
PLACEMENT              0
HONOR_OPTED_OR_NOT     0
EDUCATION_TYPE         0
ACADEMIC_PROGRAM       0
COURSE 1 MARKS         0
COURSE 2 MARKS         0
COURSE 3 MARKS         0
COURSE 4 MARKS         0
COURSE 5 MARKS         0
PERCENTILE             0
OVEARLL_GRADE          0
dtype: int64
```

checking for Outliers

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

```
import seaborn as sns
sns.boxplot(x=df['COURSE 1 MARKS'])
```

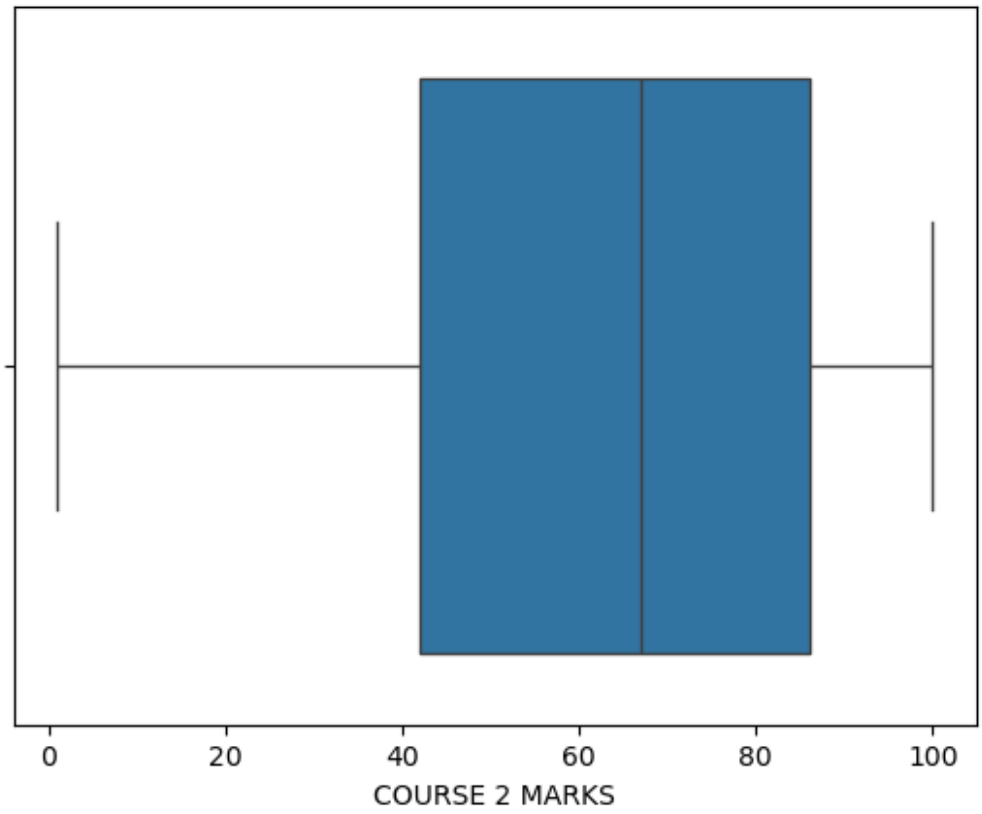
```
<Axes: xlabel='COURSE 1 MARKS'>
```



Many low-score outliers (below 20) are present and no significant high-score outliers can be seen.

```
#similarly check for all courses
import seaborn as sns
sns.boxplot(x=df['COURSE 2 MARKS'])
```

```
<Axes: xlabel='COURSE 2 MARKS'>
```



Handle the outliers and get count

```
Q1 = df["COURSE 1 MARKS"].quantile(0.25)
Q3 = df["COURSE 1 MARKS"].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# finding outliers--->
outliers = df[(df["COURSE 1 MARKS"] < lower_bound) | (df["COURSE 1
MARKS"] > upper_bound)]
print(f"Outliers detected: {outliers.shape[0]}")
```

```
Outliers detected: 294
```

```
pf = df
```

C). data transformations on categorical variables to convert it into numerical variables.

```
from sklearn.preprocessing import LabelEncoder

label = LabelEncoder()
```

```
pf['GENDER'] = label.fit_transform(pf['GENDER'])
pf['GENDER'].isnull().sum()
```

```
0
```

```
pf['PLACEMENT'] = label.fit_transform(pf['PLACEMENT'])
pf['HONOR_OPTED_OR_NOT'] =
label.fit_transform(pf['HONOR_OPTED_OR_NOT'])
pf['EDUCATION_TYPE'] = label.fit_transform(pf['EDUCATION_TYPE'])
pf['ACADEMIC_PROGRAM'] = label.fit_transform(pf['ACADEMIC_PROGRAM'])
pf.isnull().sum()
```

```
STUDENT_ID             0
GENDER                 0
PLACEMENT              0
HONOR_OPTED_OR_NOT     0
EDUCATION_TYPE         0
ACADEMIC_PROGRAM       0
COURSE 1 MARKS         0
COURSE 2 MARKS         0
COURSE 3 MARKS         0
COURSE 4 MARKS         0
COURSE 5 MARKS         0
PERCENTILE             0
OVEARLL_GRADE          0
dtype: int64
```

D). Create a rightly skewed synthetic dataset of 1000 data points. Apply Log Transformation, Square Root Transformation and MinMax Scaling on this data. Demonstrate the impact of these transformations on the data using histogram plots.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler

np.random.seed(42)
right_skewed_data = np.random.exponential(scale=2.0, size=1000)

# Apply transformations
log_transformed = np.log1p(right_skewed_data)
sqrt_transformed = np.sqrt(right_skewed_data)
scaler = MinMaxScaler()
minmax_transformed = scaler.fit_transform(right_skewed_data.reshape(-
1, 1)).flatten()

fig, axes = plt.subplots(2, 2, figsize=(12, 8))

axes[0, 0].hist(right_skewed_data, bins=30, color='blue', alpha=0.7)
axes[0, 0].set_title("Original Right-Skewed Data")
```

```
axes[0, 1].hist(log_transformed, bins=30, color='coral', alpha=0.9)
axes[0, 1].set_title("Log Transformation")

axes[1, 0].hist(sqrt_transformed, bins=30, color='crimson', alpha=0.8)
axes[1, 0].set_title("Square Root Transformation")

axes[1, 1].hist(minmax_transformed, bins=30, color='red', alpha=0.7)
axes[1, 1].set_title("Min-Max Scaling")
plt.show()
```