

```
# Author: Jay Jitesh Chavan
# Data Analyst
```

```
pip install pandas numpy matplotlib seaborn
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (1.26.4)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: seaborn in /usr/local/lib/python3.11/dist-packages (0.13.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2024.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.3.0)
Requirement already satisfied: cyclor>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (4.53.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.4.7)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (24.1)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (3.1.4)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
dataset= pd.read_csv("UberDataset.csv")
```

```
dataset
```

	START_DATE	END_DATE	CATEGORY	START	STOP	MILES	PURPOSE
0	01-01-2016 21:11	01-01-2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	01-02-2016 01:25	01-02-2016 01:37	Business	Fort Pierce	Fort Pierce	5.0	NaN
2	01-02-2016 20:25	01-02-2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	01-05-2016 17:31	01-05-2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	01-06-2016 14:42	01-06-2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit
...	...	...	...	...	...	...	...
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	Unknown Location	3.9	Temporary Site
1152	12/31/2016 15:03	12/31/2016 15:38	Business	Unknown Location	Unknown Location	16.2	Meeting
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	Gampaha	6.4	Temporary Site
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	Ilukwatta	48.2	Temporary Site
1155	Totals	NaN	NaN	NaN	NaN	12204.7	NaN

1156 rows x 7 columns

Next steps:

Generate code with dataset

View recommended plots

New interactive sheet

```
dataset.shape
```


```
(1156, 7)
```

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1156 entries, 0 to 1155
Data columns (total 7 columns):
#   Column      Non-Null Count  Dtype
---  -
0   START_DATE  1156 non-null   object
1   END_DATE    1155 non-null   object
2   CATEGORY    1155 non-null   object
3   START       1155 non-null   object
4   STOP        1155 non-null   object
5   MILES       1156 non-null   float64
6   PURPOSE     653 non-null    object
dtypes: float64(1), object(6)
memory usage: 63.3+ KB
```

## ▼ Data Preprocessing

```
dataset['PURPOSE'].fillna("NOT",inplace = True)
```

 <ipython-input-51-395aacddabd5>:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series. The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which it is performed is a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)'

```
dataset['PURPOSE'].fillna("NOT",inplace = True)
```

```
dataset.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1156 entries, 0 to 1155
Data columns (total 7 columns):
#   Column      Non-Null Count  Dtype
---  -
0   START_DATE  1156 non-null   object
1   END_DATE    1155 non-null   object
2   CATEGORY    1155 non-null   object
3   START       1155 non-null   object
4   STOP        1155 non-null   object
5   MILES       1156 non-null   float64
6   PURPOSE     653 non-null    object
dtypes: float64(1), object(6)
memory usage: 63.3+ KB
```

	START_DATE	END_DATE	CATEGORY	START	STOP	MILES	PURPOSE
0	01-01-2016 21:11	01-01-2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	01-02-2016 01:25	01-02-2016 01:37	Business	Fort Pierce	Fort Pierce	5.0	NOT
2	01-02-2016 20:25	01-02-2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	01-05-2016 17:31	01-05-2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	01-06-2016 14:42	01-06-2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit

Next steps: [Generate code with dataset](#) [View recommended plots](#) [New interactive sheet](#)

```
dataset ['START_DATE'] = pd.to_datetime(dataset['START_DATE'], errors = 'coerce' )
```

```
dataset ['END_DATE'] = pd.to_datetime(dataset['END_DATE'], errors = 'coerce' )
```

```
dataset.info()
```


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1156 entries, 0 to 1155
Data columns (total 7 columns):
#   Column      Non-Null Count  Dtype
---  -
0   START_DATE  1156 non-null   datetime64[ns]
1   END_DATE    1155 non-null   datetime64[ns]
2   CATEGORY    1155 non-null   object
3   START       1155 non-null   object
4   STOP        1155 non-null   object
dtypes: datetime64[ns](2), object(5)
memory usage: 63.3+ KB
```



```
5 MILES      1156 non-null  float64
6 PURPOSE    1156 non-null  object
dtypes: datetime64[ns](2), float64(1), object(4)
memory usage: 63.3+ KB
```


```
from datetime import datetime

dataset ['DATE'] = pd.DatetimeIndex(dataset['START_DATE']).date
dataset ['TIME'] = pd.DatetimeIndex(dataset['START_DATE']).hour
```

```
dataset.head()
```



	START_DATE	END_DATE	CATEGORY	START	STOP	MILES	PURPOSE	DATE	TIME	
0	2016-01-01 21:11:00	2016-01-01 21:17:00	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain	2016-01-01	21.0	
1	2016-01-02 01:25:00	2016-01-02 01:37:00	Business	Fort Pierce	Fort Pierce	5.0	NOT	2016-01-02	1.0	
2	2016-01-02 20:25:00	2016-01-02 20:38:00	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies	2016-01-02	20.0	



Next steps:

[Generate code with dataset](#)

[View recommended plots](#)

[New interactive sheet](#)


```
dataset ['DAY/NIGHT'] = pd.cut(x=dataset['TIME'],bins = [0,10,15,19,24], labels = ['Morning','Afternoon','Evening'],'N
```



```
dataset ['MONTH'] = pd.DatetimeIndex(dataset['START_DATE']).month
month_label = { 1.0: 'Jan', 2.0: 'Feb', 3.0: 'Mar', 4.0: 'Apr',
               5.0: 'May', 6.0: 'Jun', 7.0: 'Jul', 8.0: 'Aug',
               9.0: 'Sep', 10.0: 'Oct', 11.0: 'Nov', 12.0: 'Dec' }
```

```
dataset ['MONTH'] = dataset.MONTH.map (month_label)
Mon = dataset.MONTH.value_counts (sort=False)
```

```
dataset['DAY'] = dataset.START_DATE.dt.weekday
day_label = {
    0: 'Mon', 1:'Tues', 2:'Wed', 3:'Thun',4:'Fri', 5:'Sat', 6:'Sun'}
dataset['DAY'] = dataset['DAY'].map(day_label)
```

```
dataset.head()
```



	START_DATE	END_DATE	CATEGORY	START	STOP	MILES	PURPOSE	DATE	TIME	DAY/NIGHT	MONTH	DAY	
0	2016-01-01 21:11:00	2016-01-01 21:17:00	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain	2016-01-01	21.0	Night	Jan	Fri	
1	2016-01-02 01:25:00	2016-01-02 01:37:00	Business	Fort Pierce	Fort Pierce	5.0	NOT	2016-01-02	1.0	Morning	Jan	Sat	
2	2016-01-02 20:25:00	2016-01-02 20:38:00	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies	2016-01-02	20.0	Night	Jan	Sat	
3	2016-01-05 17:31:00	2016-01-05 17:45:00	Business	Fort Pierce	Fort Pierce	4.7	Meeting	2016-01-05	17.0	Evening	Jan	Tues	

Next steps:

[Generate code with dataset](#)

[View recommended plots](#)

[New interactive sheet](#)

```
dataset.dropna(inplace = True)
```

```
dataset.shape
```

```
(413, 12)
```

## ✓ Data Visualization

```
plt.figure(figsize=(20, 5))
```

```
# First subplot with different colors
```

```
plt.subplot(1, 2, 1)
```

```
sns.countplot(x=dataset['CATEGORY'], palette='husl') # 'husl' gives distinct colors
```

```
plt.xticks(rotation=90)
```

```
# Second subplot with different colors
```

```
plt.subplot(1, 2, 2)
```

```
sns.countplot(x=dataset['PURPOSE'], palette='Set2') # 'Set2' also has distinct colors
```

```
plt.show()
```

```
<ipython-input-63-cd9295bb19bc>:5: FutureWarning:
```

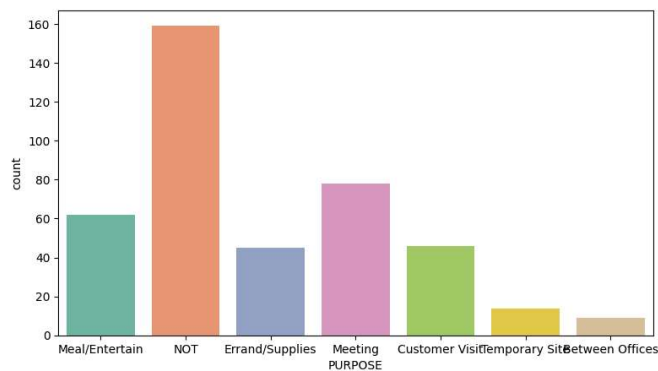
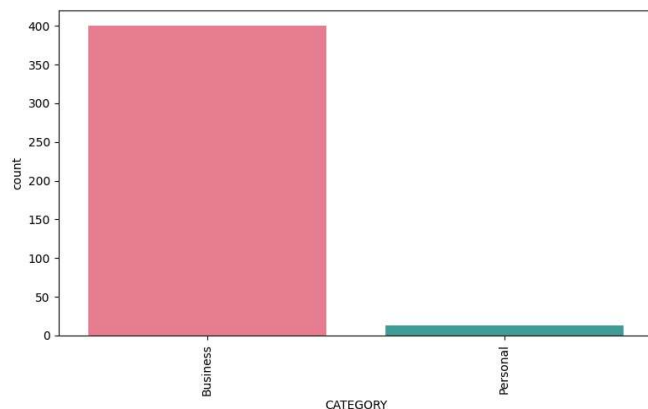
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to

```
sns.countplot(x=dataset['CATEGORY'], palette='husl') # 'husl' gives distinct colors
```

```
<ipython-input-63-cd9295bb19bc>:10: FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to

```
sns.countplot(x=dataset['PURPOSE'], palette='Set2') # 'Set2' also has distinct colors
```

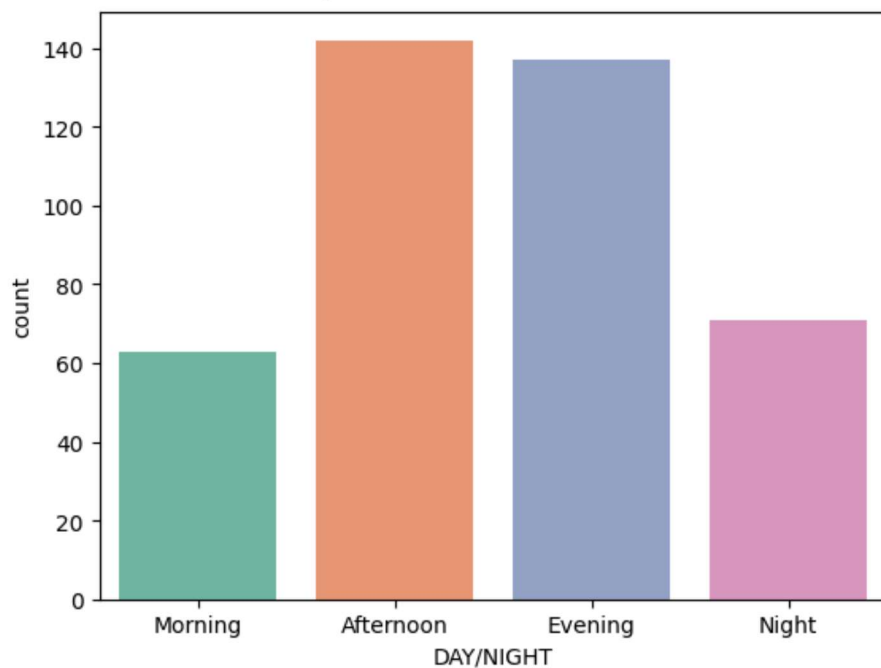


```
sns.countplot(x=dataset['DAY/NIGHT'], palette="Set2") # 'Set2' provides distinct colors
```

```
<ipython-input-64-2099a6459c02>:1: FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to

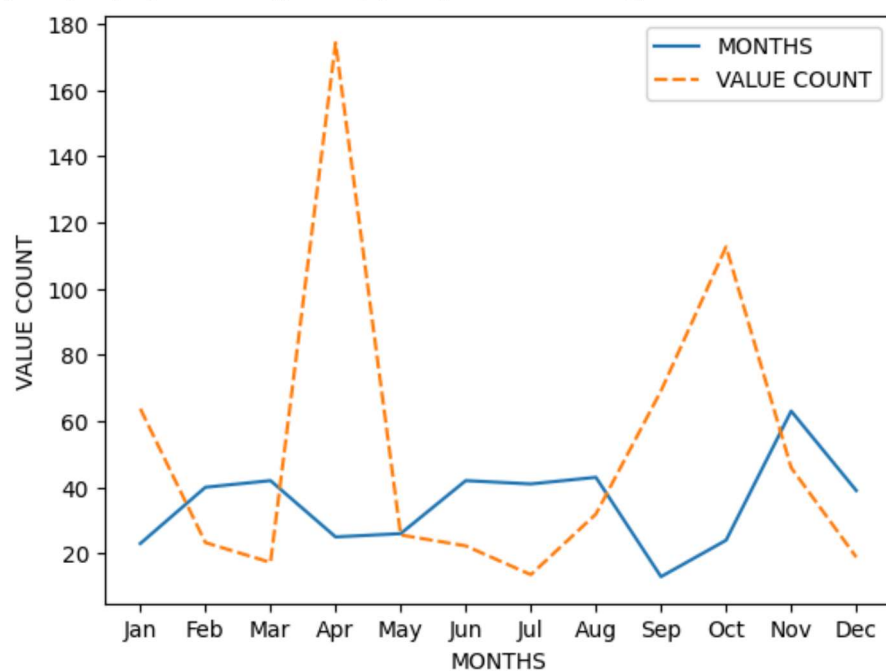
```
sns.countplot(x=dataset['DAY/NIGHT'], palette="Set2") # 'Set2' provides distinct colors
<Axes: xlabel='DAY/NIGHT', ylabel='count'>
```



```
df = pd.DataFrame({
    'MONTHS' : Mon.values,
    'VALUE COUNT': dataset.groupby ('MONTH', sort=False) ['MILES'].max()
})
```

```
p = sns.lineplot(data=df)
p.set(xlabel="MONTHS", ylabel="VALUE COUNT")
```

```
[Text(0.5, 0, 'MONTHS'), Text(0, 0.5, 'VALUE COUNT')]
```



```
# Count occurrences of each category in DAY column
day_label = dataset.DAY.value_counts()
```

```
# Create the bar plot with different solid colors
```

```
plt.figure(figsize=(10, 5))
```

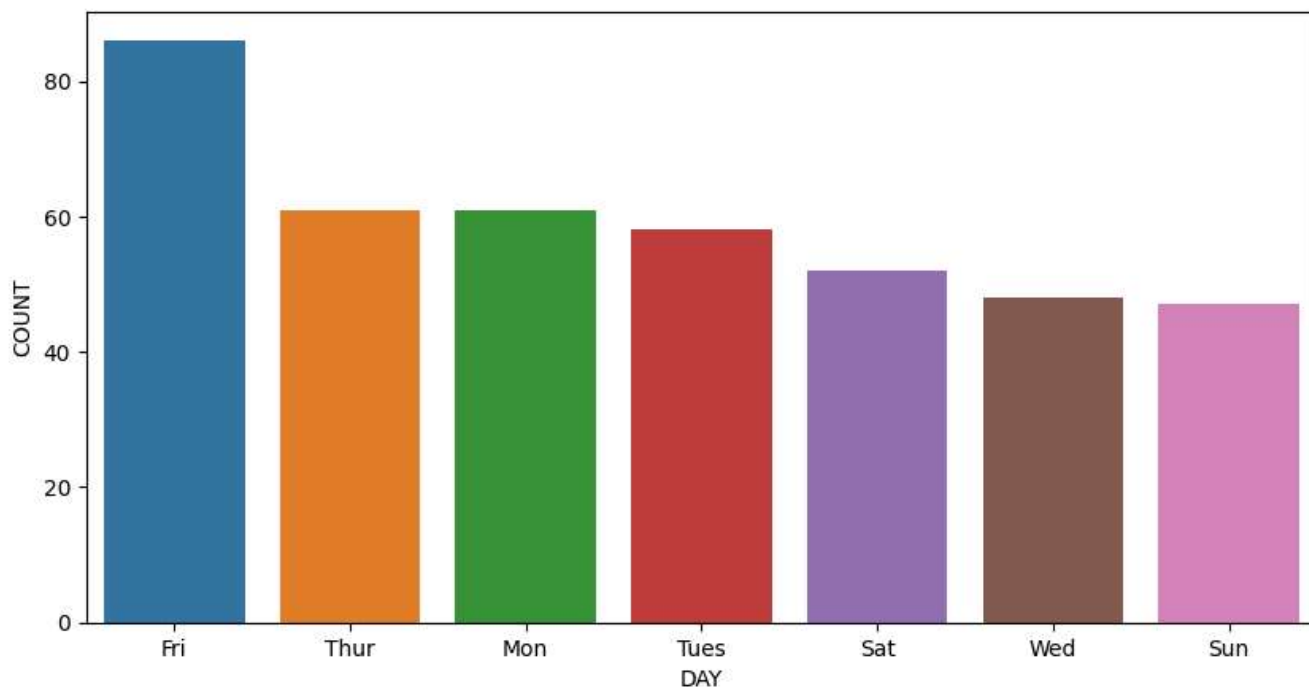
```
sns.barplot(x=day_label.index, y=day_label.values, palette="tab10") # 'tab10' provides distinct solid colors
```

```
# Labeling the axes
```

```
plt.xlabel('DAY')
```

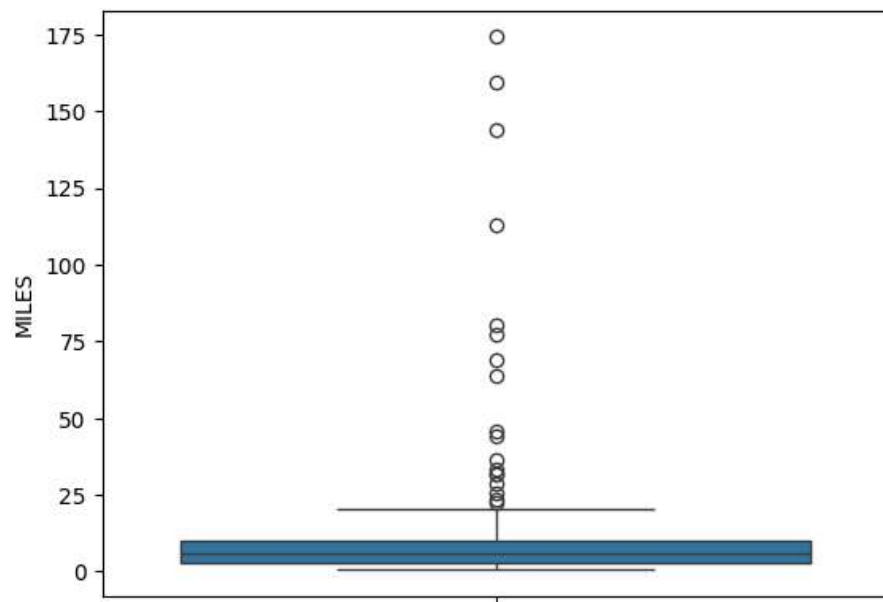
```
plt.ylabel('COUNT')
```

```
↗ <ipython-input-66-961d86d21b6f>:6: FutureWarning:
    Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to
    sns.barplot(x=day_label.index, y=day_label.values, palette="tab10") # 'tab10' provides distinct solid colors
    Text(0, 0.5, 'COUNT')
```



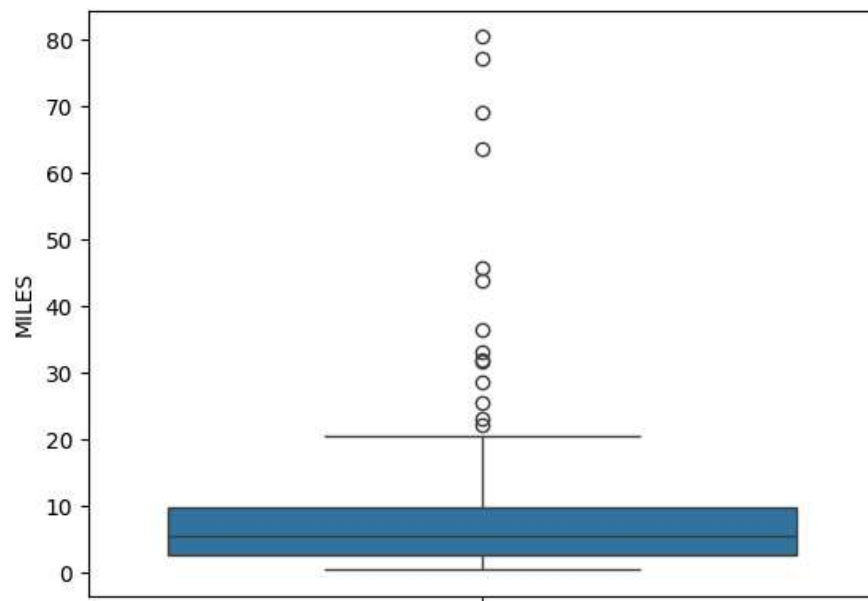
```
sns.boxplot(dataset['MILES'])
```

↵ <Axes: ylabel='MILES'>




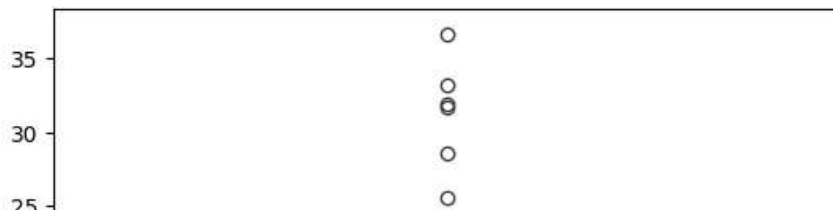
```
sns.boxplot(dataset[dataset['MILES'] < 100]['MILES'])
```

↵ <Axes: ylabel='MILES'>



```
sns.boxplot(dataset[dataset['MILES'] < 40]['MILES'])
```

 <Axes: ylabel='MILES'>



```
plt.figure(figsize=(10, 5))
```

```
sns.histplot(dataset[dataset['MILES'] < 40]['MILES'], bins=20, kde=True, color="purple")
```

```
plt.xlabel("MILES")
```

```
plt.ylabel("Density")
```

```
plt.title("Distribution of MILES (MILES < 40)")
```

 Text(0.5, 1.0, 'Distribution of MILES (MILES < 40)')

