# (1)Compute total records in the dataset

In [50]:

```
import pandas as pd
import numpy as np
```

In [51]:

```
#loading dataset
df = pd.read_csv("/content/drive/MyDrive/DataVisualization_4th_sem/LabPractica
df.head(3)
```

Out[51]:

| | symboling | normalized-losses | make | fuel-type | aspiration | num-of-doors | body-style | drive-wheels |
|---|---|---|---|---|---|---|---|---|
| **0** | 3 | ? | alfa-romero | gas | std | two | convertible | rwd |
| **1** | 3 | ? | alfa-romero | gas | std | two | convertible | rwd |
| **2** | 1 | ? | alfa-romero | gas | std | two | hatchback | rwd |

3 rows × 26 columns

In [52]:

```
print("Total records in the dataset : ",len(df))
```

```
Total records in the dataset :  205
```

# (2)Compute number of attributes and its naming list in the dataset

In [53]:

```
print("Number of attributes in the dataset : ",len(df.columns))
```

```
Number of attributes in the dataset :  26
```

In [54]:

```
print("attributes naming list in the dataset ",df.columns)
```

```
attributes naming list in the dataset  Index(['symboling', 'normalized-losses',
'make', 'fuel-type', 'aspiration',
       'num-of-doors', 'body-style', 'drive-wheels', 'engine-location',
       'wheel-base', 'length', 'width', 'height', 'curb-weight', 'engine-type',
       'num-of-cylinders', 'engine-size', 'fuel-system', 'bore', 'stroke',
       'compression-ratio', 'horsepower', 'peak-rpm', 'city-mpg',
       'highway-mpg', 'price'],
      dtype='object')
```

## (3)Display top 25 and last 25 records in the dataset.

```
In [55]: print("Top 25 records in the dataset.")
         df.head(25)
```

```
Top 25 records in the dataset.
```

Out[55]:

| | symboling | normalized-losses | make | fuel-type | aspiration | num-of-doors | body-style | drive wheel |
|---|---|---|---|---|---|---|---|---|
| **0** | 3 | ? | alfa-romero | gas | std | two | convertible | rw |
| **1** | 3 | ? | alfa-romero | gas | std | two | convertible | rw |
| **2** | 1 | ? | alfa-romero | gas | std | two | hatchback | rw |
| **3** | 2 | 164 | audi | gas | std | four | sedan | fw |
| **4** | 2 | 164 | audi | gas | std | four | sedan | 4w |
| **5** | 2 | ? | audi | gas | std | two | sedan | fw |
| **6** | 1 | 158 | audi | gas | std | four | sedan | fw |
| **7** | 1 | ? | audi | gas | std | four | wagon | fw |
| **8** | 1 | 158 | audi | gas | turbo | four | sedan | fw |
| **9** | 0 | ? | audi | gas | turbo | two | hatchback | 4w |
| **10** | 2 | 192 | bmw | gas | std | two | sedan | rw |
| **11** | 0 | 192 | bmw | gas | std | four | sedan | rw |
| **12** | 0 | 188 | bmw | gas | std | two | sedan | rw |
| **13** | 0 | 188 | bmw | gas | std | four | sedan | rw |
| **14** | 1 | ? | bmw | gas | std | four | sedan | rw |
| **15** | 0 | ? | bmw | gas | std | four | sedan | rw |
| **16** | 0 | ? | bmw | gas | std | two | sedan | rw |
| **17** | 0 | ? | bmw | gas | std | four | sedan | rw |
| **18** | 2 | 121 | chevrolet | gas | std | two | hatchback | fw |
| **19** | 1 | 98 | chevrolet | gas | std | two | hatchback | fw |
| **20** | 0 | 81 | chevrolet | gas | std | four | sedan | fw |
| **21** | 1 | 118 | dodge | gas | std | two | hatchback | fw |
| **22** | 1 | 118 | dodge | gas | std | two | hatchback | fw |
| **23** | 1 | 118 | dodge | gas | turbo | two | hatchback | fw |
| **24** | 1 | 148 | dodge | gas | std | four | hatchback | fw |

25 rows × 26 columns

In [56]:

```
print("Last 25 records in the dataset.")
df.tail(25)
```

```
Last 25 records in the dataset.
```

Out[56]:

| | symboling | normalized-losses | make | fuel-type | aspiration | num-of-doors | body-style |
|---|---|---|---|---|---|---|---|
| **180** | -1 | 90 | toyota | gas | std | four | sedan |
| **181** | -1 | ? | toyota | gas | std | four | wagon |
| **182** | 2 | 122 | volkswagen | diesel | std | two | sedan |
| **183** | 2 | 122 | volkswagen | gas | std | two | sedan |
| **184** | 2 | 94 | volkswagen | diesel | std | four | sedan |
| **185** | 2 | 94 | volkswagen | gas | std | four | sedan |
| **186** | 2 | 94 | volkswagen | gas | std | four | sedan |
| **187** | 2 | 94 | volkswagen | diesel | turbo | four | sedan |
| **188** | 2 | 94 | volkswagen | gas | std | four | sedan |
| **189** | 3 | ? | volkswagen | gas | std | two | convertible |
| **190** | 3 | 256 | volkswagen | gas | std | two | hatchback |
| **191** | 0 | ? | volkswagen | gas | std | four | sedan |
| **192** | 0 | ? | volkswagen | diesel | turbo | four | sedan |
| **193** | 0 | ? | volkswagen | gas | std | four | wagon |
| **194** | -2 | 103 | volvo | gas | std | four | sedan |
| **195** | -1 | 74 | volvo | gas | std | four | wagon |
| **196** | -2 | 103 | volvo | gas | std | four | sedan |
| **197** | -1 | 74 | volvo | gas | std | four | wagon |
| **198** | -2 | 103 | volvo | gas | turbo | four | sedan |
| **199** | -1 | 74 | volvo | gas | turbo | four | wagon |
| **200** | -1 | 95 | volvo | gas | std | four | sedan |
| **201** | -1 | 95 | volvo | gas | turbo | four | sedan |
| **202** | -1 | 95 | volvo | gas | std | four | sedan |
| **203** | -1 | 95 | volvo | diesel | turbo | four | sedan |
| **204** | -1 | 95 | volvo | gas | turbo | four | sedan |

25 rows × 26 columns

## (4)Display total number of missing values in each attribute in the dataset. Find and display invalid(missing, zeros etc) records in the dataset.

---

In [57]:
```
print("Total number of missing values in each attribute in the dataset.")
df.isnull().sum()
```

```
Total number of missing values in each attribute in the dataset.
```

Out[57]:

| | 0 |
|---|---|
| **symboling** | 0 |
| **normalized-losses** | 0 |
| **make** | 0 |
| **fuel-type** | 0 |
| **aspiration** | 0 |
| **num-of-doors** | 0 |
| **body-style** | 0 |
| **drive-wheels** | 0 |
| **engine-location** | 0 |
| **wheel-base** | 0 |
| **length** | 0 |
| **width** | 0 |
| **height** | 0 |
| **curb-weight** | 0 |
| **engine-type** | 0 |
| **num-of-cylinders** | 0 |
| **engine-size** | 0 |
| **fuel-system** | 0 |
| **bore** | 0 |
| **stroke** | 0 |
| **compression-ratio** | 0 |
| **horsepower** | 0 |
| **peak-rpm** | 0 |
| **city-mpg** | 0 |
| **highway-mpg** | 0 |
| **price** | 0 |

**dtype:** int64

In [58]:

```
print("Find and display invalid(missing, zeros etc) records in the dataset.")
df[df.isnull().any(axis=1)]
```

```
Find and display invalid(missing, zeros etc) records in the dataset.
```

Out[58]:

| symboling | normalized-losses | make | fuel-type | aspiration | num-of-doors | body-style | drive-wheels | engin locatio |
|---|---|---|---|---|---|---|---|---|

0 rows × 26 columns

---

## (5)Display all numeric and non-numeric attribute in the dataset. Identify the data-types of every attribute.

---

In [59]:

```python
print("All numeric attribute in the dataset")
df.select_dtypes(include=["int64","float64"]).columns
```

```
All numeric attribute in the dataset
```

Out[59]:

```
Index(['symboling', 'wheel-base', 'length', 'width', 'height', 'curb-weight',
       'engine-size', 'compression-ratio', 'city-mpg', 'highway-mpg'],
      dtype='object')
```

In [60]:

```python
print("All non-numeric attribute in the dataset")
df.select_dtypes(exclude=["int64","float64"]).columns
```

```
All non-numeric attribute in the dataset
```

Out[60]:

```
Index(['normalized-losses', 'make', 'fuel-type', 'aspiration', 'num-of-door
s',
       'body-style', 'drive-wheels', 'engine-location', 'engine-type',
       'num-of-cylinders', 'fuel-system', 'bore', 'stroke', 'horsepower',
       'peak-rpm', 'price'],
      dtype='object')
```

In [61]:

```python
print("The data-types of every attribute.")
df.dtypes
```

```
The data-types of every attribute.
```

Out[61]:

| | 0 |
|---:|---:|
| **symboling** | int64 |
| **normalized-losses** | object |
| **make** | object |
| **fuel-type** | object |
| **aspiration** | object |
| **num-of-doors** | object |
| **body-style** | object |
| **drive-wheels** | object |
| **engine-location** | object |
| **wheel-base** | float64 |
| **length** | float64 |
| **width** | float64 |
| **height** | float64 |
| **curb-weight** | int64 |
| **engine-type** | object |
| **num-of-cylinders** | object |
| **engine-size** | int64 |
| **fuel-system** | object |
| **bore** | object |
| **stroke** | object |
| **compression-ratio** | float64 |
| **horsepower** | object |
| **peak-rpm** | object |
| **city-mpg** | int64 |
| **highway-mpg** | int64 |
| **price** | object |

**dtype:** object

## (6)Create a new data frame of numeric attributes. Delete the all invalid records from the new data frame

---

In [62]:
```
print("Create a new data frame of numeric attributes")
new_df=df.loc[:,df.select_dtypes(include=["int64","float64"]).columns]
new_df.head(3)
```

```
Create a new data frame of numeric attributes
```

Out[62]:

| | symboling | wheel-base | length | width | height | curb-weight | engine-size | compression-ratio | cit mp |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 3 | 88.6 | 168.8 | 64.1 | 48.8 | 2548 | 130 | 9.0 | 2 |
| **1** | 3 | 88.6 | 168.8 | 64.1 | 48.8 | 2548 | 130 | 9.0 | 2 |
| **2** | 1 | 94.5 | 171.2 | 65.5 | 52.4 | 2823 | 152 | 9.0 | 1 |

---

## (7)Compute the average height width and length of all automobiles.

---

In [63]:
```
print("the average height width and length of all automobiles.")
df[["height","width","length"]].mean()
```

```
the average height width and length of all automobiles.
```

Out[63]:

| | 0 |
|---|---|
| **height** | 53.724878 |
| **width** | 65.907805 |
| **length** | 174.049268 |

**dtype:** float64

---

## (8)Compute the standard deviation height width and length of all automobiles.

---

In [64]:
```
print("the standatrd deviation of  height width and length of all automobiles.
df[["height","width","length"]].std()
```

```
the standatrd deviation of  height width and length of all automobiles.
```

Out[64]:

|  | 0 |
|---|---|
| **height** | 2.443522 |
| **width** | 2.145204 |
| **length** | 12.337289 |

**dtype:** float64

---

# (9)Describe each attribute in dataset by min, max, mean, median, mode and std.

---

In [65]:

```
print("Min : ")
df.loc[:,df.select_dtypes(include=["int64","float64"]).columns].min()
```

```
Min :
```

Out[65]:

|  | 0 |
|---|---|
| **symboling** | -2.0 |
| **wheel-base** | 86.6 |
| **length** | 141.1 |
| **width** | 60.3 |
| **height** | 47.8 |
| **curb-weight** | 1488.0 |
| **engine-size** | 61.0 |
| **compression-ratio** | 7.0 |
| **city-mpg** | 13.0 |
| **highway-mpg** | 16.0 |

**dtype:** float64

In [66]:

```
print("Max : ")
df.loc[:,df.select_dtypes(include=["int64","float64"]).columns].max()
```

```
Max :
```

Out[66]:

|  | 0 |
|---|---|
| **symboling** | 3.0 |
| **wheel-base** | 120.9 |
| **length** | 208.1 |
| **width** | 72.3 |
| **height** | 59.8 |
| **curb-weight** | 4066.0 |
| **engine-size** | 326.0 |
| **compression-ratio** | 23.0 |
| **city-mpg** | 49.0 |
| **highway-mpg** | 54.0 |

**dtype:** float64

In [67]:

```
print("Mean: ")
df.loc[:,df.select_dtypes(include=["int64","float64"]).columns].mean()
```

```
Mean:
```

Out[67]:

|  | 0 |
|---|---|
| **symboling** | 0.834146 |
| **wheel-base** | 98.756585 |
| **length** | 174.049268 |
| **width** | 65.907805 |
| **height** | 53.724878 |
| **curb-weight** | 2555.565854 |
| **engine-size** | 126.907317 |
| **compression-ratio** | 10.142537 |
| **city-mpg** | 25.219512 |
| **highway-mpg** | 30.751220 |

**dtype:** float64

In [68]:

```
print("Median : ")
df.loc[:,df.select_dtypes(include=["int64","float64"]).columns].median()
```

```
Median :
```

Out[68]:

|  | 0 |
|---|---|
| **symboling** | 1.0 |
| **wheel-base** | 97.0 |
| **length** | 173.2 |
| **width** | 65.5 |
| **height** | 54.1 |
| **curb-weight** | 2414.0 |
| **engine-size** | 120.0 |
| **compression-ratio** | 9.0 |
| **city-mpg** | 24.0 |
| **highway-mpg** | 30.0 |

**dtype:** float64

In [69]:

```
print("Mode : ")
df.loc[:,df.select_dtypes(exclude=["int64","float64"]).columns].mode()
```

```
Mode :
```

Out[69]:

|  | normalized-losses | make | fuel-type | aspiration | num-of-doors | body-style | drive-wheels | engine-location | engine-type |
|---|---|---|---|---|---|---|---|---|---|
| **0** | ? | toyota | gas | std | four | sedan | fwd | front | ohc |

In [70]:

```
print("Standard Deviation : ")
df.loc[:,df.select_dtypes(include=["int64","float64"]).columns].std()
```

```
Standard Deviation :
```

Out[70]:

| | 0 |
|---:|---:|
| **symboling** | 1.245307 |
| **wheel-base** | 6.021776 |
| **length** | 12.337289 |
| **width** | 2.145204 |
| **height** | 2.443522 |
| **curb-weight** | 520.680204 |
| **engine-size** | 41.642693 |
| **compression-ratio** | 3.972040 |
| **city-mpg** | 6.542142 |
| **highway-mpg** | 6.886443 |

**dtype:** float64

---

# Learning Outcome

---

- **Dataset Understanding**

  ```
  Learned how to load a dataset and perform basic
  exploration such as finding total records, attributes,
  and viewing top/bottom entries.
  ```
- **Data Cleaning**

  ```
  Understood how to detect missing and invalid
  values, separate numeric and non-numeric attributes,
  and create a cleaned numeric dataframe by removing invalid
  records.
  ```
- **Analysis of Data**

  ```
  Gained the ability to compute descriptive statistics
  like mean, median, mode, min, max, and standard
  deviation for numeric features.
  ```

---