	# Run from top to bottom to load all libraries import pandas as pd
In [2]: In [3]:	<pre>df = pd.read_csv('car_data.csv') # Answer 1 print("From", df["Year"].min(), "To", df.max()["Year"], ", vehicles are present in this data")</pre>
In [4]:	From 2003 To 2018 , vehicles are present in this data # Answer 2 print(df.min()["Selling_Price"], "is the lowest price to which a vehicle is sold")
In [5]:	0.1 is the lowest price to which a vehicle is sold # Answer 3
In [6]:	<pre>print(df.max()["Selling_Price"], "is the highest price to which a vehicle is sold") 35.0 is the highest price to which a vehicle is sold # Answer 4</pre>
In [7]:	<pre>print("There are", df.shape[0], "enteries with", df.shape[1], "columns") There are 301 enteries with 9 columns # Answer 5</pre>
	<pre>print("Are there any missing entries in this data ?") print("Answer:", df.isnull().values.any()) Are there any missing entries in this data ? Answer: False</pre>
In [8]:	<pre># Answer 6 print("There are", len(set(df['Car_Name'].tolist())), "types of different vehicles in the data") There are 98 types of different vehicles in the data</pre>
In [9]:	<pre># Answer 7 if ("CNG" in df["Fuel_Type"].unique()): print("There are", df["Fuel_Type"].isin(["CNG"]).values.sum(), "CNG vehicles in this data")</pre>
In [10]:	<pre>else: print("No CNG vehicles in this data") There are 2 CNG vehicles in this data # Answer 8</pre>
In [11]:	<pre>print("There are", df["Seller_Type"].isin(["Individual"]).values.sum(), "vehicles here are for sell from Individuals directly</pre> There are 106 vehicles here are for sell from Individuals directly
111 [11].	<pre>if ("Automatic" in df["Transmission"].unique()): print("There are", df["Transmission"].isin(["Automatic"]).values.sum(), "automatic vehicles in this data") else: print("No Automatic vehicles in this data")</pre>
In [12]:	There are 40 automatic vehicles in this data # Answer 10 Car Numbers = []
	<pre>Years = list(range(2014, df["Year"].max()+1)) for i in range(2014, df["Year"].max()+1):</pre>
	<pre>import matplotlib.pyplot as plt plt.xticks(Years) plt.title("Graph of number of vehicles manufactured after 2014") plt.xlabel("Years")</pre>
	<pre>plt.ylabel("Car Numbers") plt.plot(Years, Car_Numbers) plt.show()</pre>
	Graph of number of vehicles manufactured after 2014 60 - 50 -
	y 40 - y 30 - y 20 -
	10 - 0 - 2014 2015 2016 2017 2018
In [36]:	# Answer 11 # From here to the end of this notebook
In [15]:	<pre>import seaborn as sns df = pd.read_csv('car_data.csv') # ScatterPlot Graph</pre>
	<pre>res = sns.scatterplot(x="Selling_Price", y="Year", data=df) plt.xlim(-2, 36) plt.show()</pre>
	2016 -
	2012 - 2010 - 2010 - 2008 - 2006 - 20
	2004 - 0 5 10 15 20 25 30 35 Selling_Price
In [16]:	<pre># LinePlot Graph res = sns.lineplot(x="Selling_Price", y="Year", data=df) plt.xlim(-2, 10) plt.show()</pre>
	2018 - 2016 - 2014 -
	2012 - 2010 - 2008 -
	2006 - 2004 - -2 0 2 4 6 8 10 Selling_Price
In [18]:	<pre># BoxenPlot Graph res = sns.boxenplot(x=df['Selling_Price']) plt.show()</pre>
	0 5 10 15 20 25 30 35 Selling_Price
In [19]:	<pre># BoxPlot Graph res = sns.boxplot(x=df['Selling_Price']) plt.show()</pre>

	0 5 10 15 20 25 30 35 Selling_Price
In [20]:	<pre># ViolinPlot Graph res = sns.violinplot(x=df['Selling_Price']) plt.show()</pre>
	0 10 20 30 40 Selling_Price
In [22]:	<pre># CountPlot Graph res = sns.countplot(x=df['Year']) plt.show()</pre>
	60 - 50 -
	40 - 10 30 - 20 -
	10 - 200320042005200620072008201920192019201920192018
In [23]:	
	10 -
	Selling Price 4 -
	2 - 20032004200520062007200820090102017201320142015201620172018 Year
In [24]:	<pre># PointPlot Graph res = sns.pointplot(x=df["Year"], y=df["Selling_Price"]) plt.show()</pre>
	10 - 8 -
	Selling Price of Pric
	20032004200520062007200820090102017201320142015201620172018 Year
In [26]:	<pre># DistPlot Graph res = sns.distplot(x=df['Selling_Price']) plt.show()</pre>
	C:\Users\karti\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn\distributions.py:2619: FutureW arning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function f or histograms). warnings.warn(msg, FutureWarning)
	0.20 -
	0.15
	0.00 0 10 20 30 40
In [27]:	<pre># DisPlot Graph res = sns.displot(x=df['Selling_Price']) plt.show()</pre>
	100 - 80 -
	60 - 40 - 40 - 40 - 40 - 40 - 40 - 40 -
	40 - 20 -
	0 5 10 15 20 25 30 35 Selling_Price
In [28]:	<pre># HistPlot Graph res = sns.histplot(x=df['Selling_Price']) plt.show()</pre>
	100 - 80 - 60 -
	40 -
	0 5 10 15 20 25 30 35 Selling_Price
In [30]:	<pre># JointPlot Graph res = sns.jointplot(x=df['Selling_Price'], y=df["Selling_Price"]) plt.show()</pre>
	35 -
	30
	20 - 20 -
	10 - 5 -
In []:	# And there are many other graphs that can be made using Seaborn
- 1.	