In [4]:

```
import pandas as pd
```

In [7]:

```
# two main datatypes
series = pd.Series (["BMW", "Toyoto","Honda"])
```

In [9]:

```
series
```

Out[9]:

```
0       BMW
1    Toyoto
2     Honda
dtype: object
```

In [10]:

```
# Series = 1 dimensional
```

In [11]:

```
colours =pd.Series(["red", "Blue","White"])
```

In [12]:

```
colours
```

Out[12]:

```
0      red
1     Blue
2    White
dtype: object
```

In [13]:

```
 car_data = pd.DataFrame({"Car Make" : series, "Colour": colours})
 car_data
```

Out[13]:

	Car Make	Colour
0	BMW	red
1	Toyoto	Blue
2	Honda	White

In [15]:

```
#Import data
car_sales =pd.read_csv("car-sales.csv")
```

In [16]:

```
car_sales
```

Out[16]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

In [17]:

```
# Exporting a dataframe
car_sales.to_csv("exported-car-sales.csv ")
```

Describe data

In [20]:

```
#Attributes
car_sales.dtypes
#funtion
#which have ()
```

Out[20]:

```
Make                object
Colour              object
Odometer (KM)        int64
Doors                int64
Price               object
dtype: object
```

In [22]:

```
car_sales.columns
```

Out[22]:

```
Index(['Make', 'Colour', 'Odometer (KM)', 'Doors', 'Price'], dtype='objec
t')
```

In [23]:

```
car_sales.index
```

Out[23]:

```
RangeIndex(start=0, stop=10, step=1)
```

In [24]:

```
 car_sales.describe()
    #it show value of numeric value
```

Out[24]:

	Odometer (KM)	Doors
count	10.000000	10.000000
mean	78601.400000	4.000000
std	61983.471735	0.471405
min	11179.000000	3.000000
25%	35836.250000	4.000000
50%	57369.000000	4.000000
75%	96384.500000	4.000000
max	213095.000000	5.000000

In [25]:

```
 car_sales.info
```

Out[25]:

```
<bound method DataFrame.info of      Make Colour  Odometer (KM)  Doors
Price
0  Toyota  White         150043      4   $4,000.00
1   Honda    Red          87899      4   $5,000.00
2  Toyota   Blue          32549      3   $7,000.00
3     BMW  Black          11179      5  $22,000.00
4  Nissan  White         213095      4   $3,500.00
5  Toyota  Green          99213      4   $4,500.00
6   Honda   Blue          45698      4   $7,500.00
7   Honda   Blue          54738      4   $7,000.00
8  Toyota  White          60000      4   $6,250.00
9  Nissan  White          31600      4   $9,700.00>
```

In [26]:

```
car_sales.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   Make           10 non-null     object
 1   Colour         10 non-null     object
 2   Odometer (KM)  10 non-null     int64 
 3   Doors          10 non-null     int64 
 4   Price          10 non-null     object
dtypes: int64(2), object(3)
memory usage: 528.0+ bytes
```

In [29]:

```
car_sales.mean()
```

```
C:\Users\alokr\AppData\Local\Temp\ipykernel_13924\4073448239.py:1: FutureW
arning: The default value of numeric_only in DataFrame.mean is deprecated.
In a future version, it will default to False. In addition, specifying 'nu
meric_only=None' is deprecated. Select only valid columns or specify the v
alue of numeric_only to silence this warning.
  car_sales.mean()
```

Out[29]:

```
Odometer (KM)    78601.4
Doors                4.0
dtype: float64
```

In [30]:

```
car_sales.sum()
```

Out[30]:

```
Make             ToyotaHondaToyotaBMWNissanToyotaHondaHondaToyo...
Colour               WhiteRedBlueBlackWhiteGreenBlueBlueWhiteWhite
Odometer (KM)                                               786014
Doors                                                           40
Price            $4,000.00$5,000.00$7,000.00$22,000.00$3,500.00...
dtype: object
```

In [31]:

```
#to get particular vale
car_sales["Doors"].sum()
```

Out[31]:

```
40
```

In [33]:

```
len(car_sales)
```

Out[33]:

```
10
```

In [34]:

```
car_sales
```

Out[34]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

Viewing and Selecting data

In [35]:

```
car_sales.head()
#it return top 5
```

Out[35]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00

In [36]:

```
car_sales.head(6)
```

Out[36]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00

In [37]:

```
car_sales.tail()
#bottom  5
```

Out[37]:

	Make	Colour	Odometer (KM)	Doors	Price
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

In [38]:

```
#loc and iloc
#loc referas to index but ilock refers to position
```

In [41]:

```
animals =pd.Series(["cat","Dog","bird","panda","snakke"],
                   index=[0, 3 ,6 ,8 ,3])
animals
```

Out[41]:

```
0       cat
3       Dog
6      bird
8     panda
3    snakke
dtype: object
```

In [42]:

```
animals.loc[3]
```

Out[42]:

```
3       Dog
3    snakke
dtype: object
```

In [43]:

```
animals.iloc[3]
```

Out[43]:

```
'panda'
```

In [44]:

```
car_sales
```

Out[44]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

In [45]:

```
car_sales.loc[3]
```

Out[45]:

```
Make                     BMW
Colour                 Black
Odometer (KM)          11179
Doors                      5
Price             $22,000.00
Name: 3, dtype: object
```

In [46]:

```
car_sales.iloc[3]
```

Out[46]:

```
Make                      BMW
Colour                  Black
Odometer (KM)           11179
Doors                       5
Price              $22,000.00
Name: 3, dtype: object
```

In [47]:

```
  animals.iloc[:3]
```

Out[47]:

```
0     cat
3     Dog
6    bird
dtype: object
```

In [48]:

```
car_sales["Make"]
```

Out[48]:

```
0    Toyota
1     Honda
2    Toyota
3       BMW
4    Nissan
5    Toyota
6     Honda
7     Honda
8    Toyota
9    Nissan
Name: Make, dtype: object
```

In [49]:

```
car_sales.Make
#it dont work when name have space
```

Out[49]:

```
0    Toyota
1     Honda
2    Toyota
3       BMW
4    Nissan
5    Toyota
6     Honda
7     Honda
8    Toyota
9    Nissan
Name: Make, dtype: object
```

In [51]:

```
#doing boolean conditiion
car_sales[car_sales["Make"] =="Toyota"]
```

Out[51]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
2	Toyota	Blue	32549	3	$7,000.00
5	Toyota	Green	99213	4	$4,500.00
8	Toyota	White	60000	4	$6,250.00

In [53]:

```
car_sales[car_sales["Odometer (KM)"] > 100000]
```

Out[53]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
4	Nissan	White	213095	4	$3,500.00

In [54]:

```
pd.crosstab(car_sales["Make"],car_sales["Doors"])
#compering two coloumns
```

Out[54]:

Doors	**3**	**4**	**5**
Make			
BMW	0	0	1
Honda	0	3	0
Nissan	0	2	0
Toyota	1	3	0

In [55]:

```
#groupby
#compering two or more colomns
car_sales.groupby(["Make"]).mean()
```

```
C:\Users\alokr\AppData\Local\Temp\ipykernel_13924\269331477.py:3: FutureWa
rning: The default value of numeric_only in DataFrameGroupBy.mean is depre
cated. In a future version, numeric_only will default to False. Either spe
cify numeric_only or select only columns which should be valid for the fun
ction.
  car_sales.groupby(["Make"]).mean()
```

Out[55]:

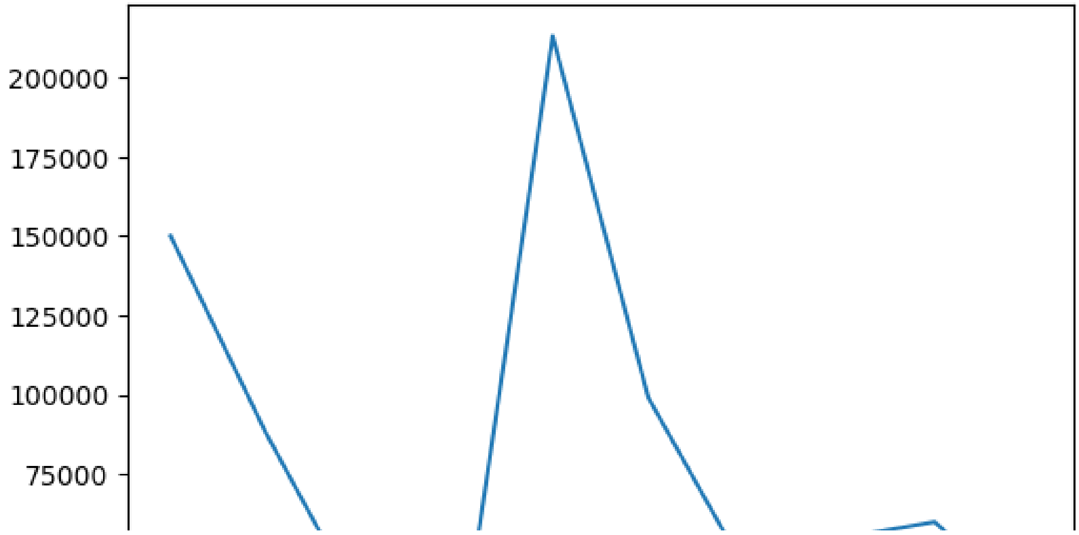
	Odometer (KM)	Doors
Make		
BMW	11179.000000	5.00
Honda	62778.333333	4.00
Nissan	122347.500000	4.00
Toyota	85451.250000	3.75

In [56]:

```
car_sales["Odometer (KM)"].plot()
```

Out[56]:

```
<AxesSubplot: >
```

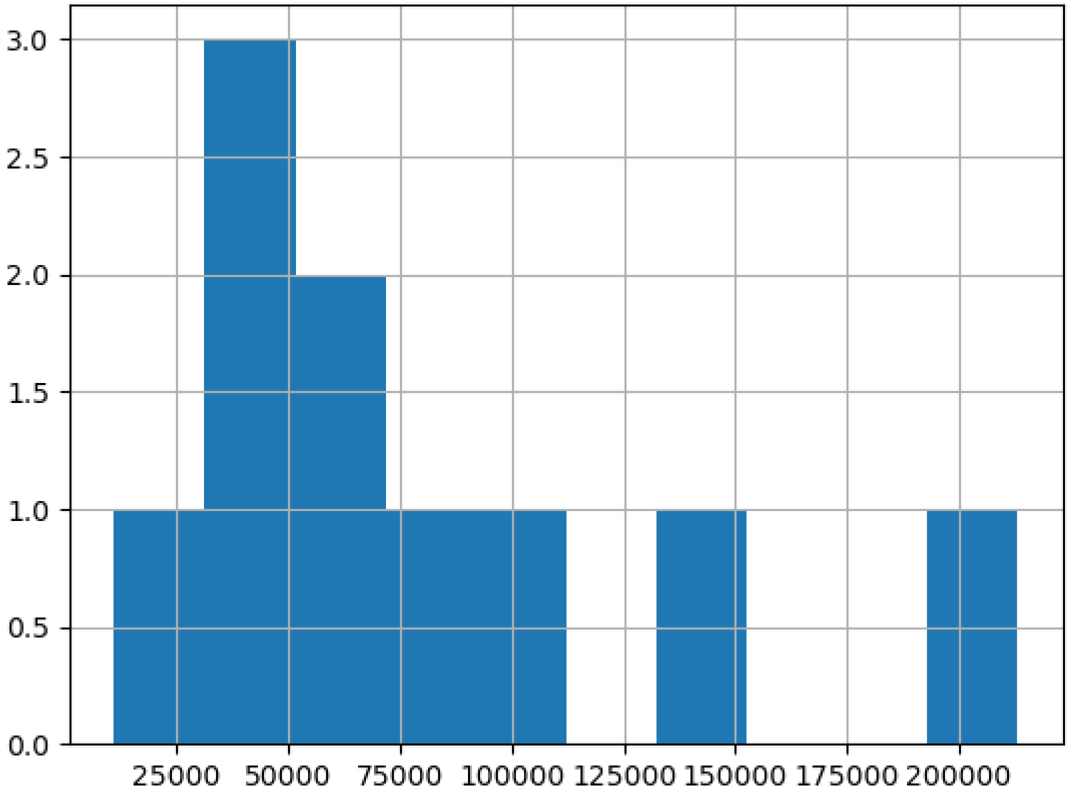


In [57]:

```
car_sales["Odometer (KM)"].hist()
```

Out[57]:

```
<AxesSubplot: >
```



In [59]:

```
car_sales["Price"].plot()
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[59], line 1
----> 1 car_sales["Price"].plot()

File ~\Machine_learning\project1\env\lib\site-packages\pandas\plotting\_co
re.py:1000, in PlotAccessor.__call__(self, *args, **kwargs)
    997             label_name = label_kw or data.columns
    998             data.columns = label_name
-> 1000 return plot_backend.plot(data, kind=kind, **kwargs)

File ~\Machine_learning\project1\env\lib\site-packages\pandas\plotting\_ma
tplotlib\__init__.py:71, in plot(data, kind, **kwargs)
     69         kwargs["ax"] = getattr(ax, "left_ax", ax)
     70 plot_obj = PLOT_CLASSES[kind](data, **kwargs)
---> 71 plot_obj.generate()
     72 plot_obj.draw()
     73 return plot_obj.result

File ~\Machine_learning\project1\env\lib\site-packages\pandas\plotting\_ma
tplotlib\core.py:450, in MPLPlot.generate(self)
    448 def generate(self) -> None:
    449     self._args_adjust()
--> 450     self._compute_plot_data()
    451     self._setup_subplots()
    452     self._make_plot()

File ~\Machine_learning\project1\env\lib\site-packages\pandas\plotting\_ma
tplotlib\core.py:635, in MPLPlot._compute_plot_data(self)
    633 # no non-numeric frames or series allowed
    634 if is_empty:
--> 635     raise TypeError("no numeric data to plot")
    637 self.data = numeric_data.apply(self._convert_to_ndarray)

TypeError: no numeric data to plot
```

In [60]:

```
#to plot it change into int
```

In [61]:

```
car_sales["Price"] =car_sales["Price"].str.replace('[\$\,\.]','' ).astype(int)
```

```
C:\Users\alokr\AppData\Local\Temp\ipykernel_13924\452675535.py:1: FutureWa
rning: The default value of regex will change from True to False in a futu
re version.
  car_sales["Price"] =car_sales["Price"].str.replace('[\$\,\.]','' ).astyp
e(int)
```

In [63]:

```
car_sales
```

Out[63]:

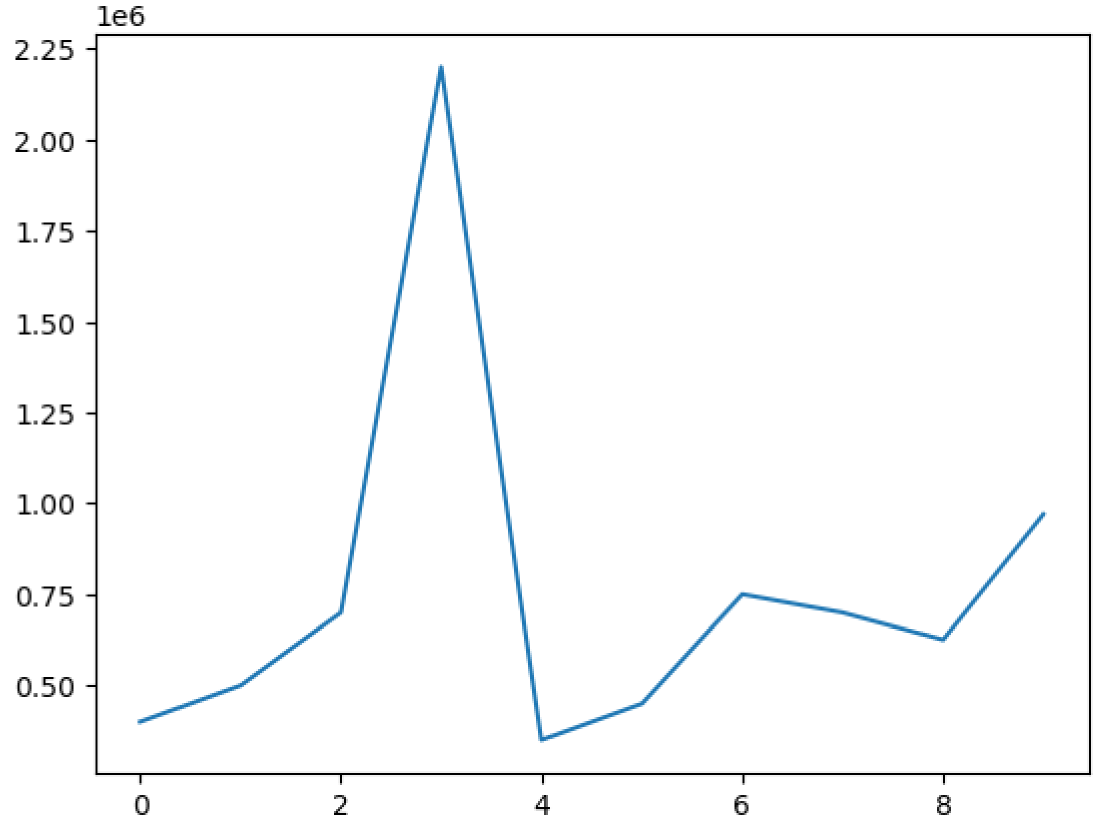
	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	400000
1	Honda	Red	87899	4	500000
2	Toyota	Blue	32549	3	700000
3	BMW	Black	11179	5	2200000
4	Nissan	White	213095	4	350000
5	Toyota	Green	99213	4	450000
6	Honda	Blue	45698	4	750000
7	Honda	Blue	54738	4	700000
8	Toyota	White	60000	4	625000
9	Nissan	White	31600	4	970000

In [64]:

```
car_sales["Price"].plot()
```

Out[64]:

```
<AxesSubplot: >
```



Mainipulating Data

In [66]:

```
car_sales["Make"].str.lower()
```

Out[66]:

```
0    toyota
1     honda
2    toyota
3       bmw
4    nissan
5    toyota
6     honda
7     honda
8    toyota
9    nissan
Name: Make, dtype: object
```

In [67]:

```
car_sales
```

Out[67]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	400000
1	Honda	Red	87899	4	500000
2	Toyota	Blue	32549	3	700000
3	BMW	Black	11179	5	2200000
4	Nissan	White	213095	4	350000
5	Toyota	Green	99213	4	450000
6	Honda	Blue	45698	4	750000
7	Honda	Blue	54738	4	700000
8	Toyota	White	60000	4	625000
9	Nissan	White	31600	4	970000

In [68]:

```
#it not save to to save lower in table use code
car_sales["Make"]=car_sales["Make"].str.lower()
```

In [69]:

```
car_sales
```

Out[69]:

	Make	Colour	Odometer (KM)	Doors	Price
0	toyota	White	150043	4	400000
1	honda	Red	87899	4	500000
2	toyota	Blue	32549	3	700000
3	bmw	Black	11179	5	2200000
4	nissan	White	213095	4	350000
5	toyota	Green	99213	4	450000
6	honda	Blue	45698	4	750000
7	honda	Blue	54738	4	700000
8	toyota	White	60000	4	625000
9	nissan	White	31600	4	970000

In [70]:

```
car_sales_missing=pd.read_csv("car-sales-missing-data.csv")
```

In [71]:

```
car_sales_missing
```

Out[71]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.0	4.0	$4,000
1	Honda	Red	87899.0	4.0	$5,000
2	Toyota	Blue	NaN	3.0	$7,000
3	BMW	Black	11179.0	5.0	$22,000
4	Nissan	White	213095.0	4.0	$3,500
5	Toyota	Green	NaN	4.0	$4,500
6	Honda	NaN	NaN	4.0	$7,500
7	Honda	Blue	NaN	4.0	NaN
8	Toyota	White	60000.0	NaN	NaN
9	NaN	White	31600.0	4.0	$9,700

In [73]:

```
#to fill na value use .fillna()
car_sales_missing["Odometer"].fillna(car_sales_missing["Odometer"].mean())
```

Out[73]:

```
0    150043.000000
1     87899.000000
2     92302.666667
3     11179.000000
4    213095.000000
5     92302.666667
6     92302.666667
7     92302.666667
8     60000.000000
9     31600.000000
Name: Odometer, dtype: float64
```

In [74]:

```
car_sales_missing
#we see it not fill in main file
```

Out[74]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.0	4.0	$4,000
1	Honda	Red	87899.0	4.0	$5,000
2	Toyota	Blue	NaN	3.0	$7,000
3	BMW	Black	11179.0	5.0	$22,000
4	Nissan	White	213095.0	4.0	$3,500
5	Toyota	Green	NaN	4.0	$4,500
6	Honda	NaN	NaN	4.0	$7,500
7	Honda	Blue	NaN	4.0	NaN
8	Toyota	White	60000.0	NaN	NaN
9	NaN	White	31600.0	4.0	$9,700

In [75]:

```
car_sales_missing["Odometer"].fillna(car_sales_missing["Odometer"].mean(),
                                    inplace=True)
```

In [76]:

```
car_sales_missing
```

Out[76]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.000000	4.0	$4,000
1	Honda	Red	87899.000000	4.0	$5,000
2	Toyota	Blue	92302.666667	3.0	$7,000
3	BMW	Black	11179.000000	5.0	$22,000
4	Nissan	White	213095.000000	4.0	$3,500
5	Toyota	Green	92302.666667	4.0	$4,500
6	Honda	NaN	92302.666667	4.0	$7,500
7	Honda	Blue	92302.666667	4.0	NaN
8	Toyota	White	60000.000000	NaN	NaN
9	NaN	White	31600.000000	4.0	$9,700

In [79]:

```
car_sales_missing.dropna()
#it remove
```

Out[79]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.000000	4.0	$4,000
1	Honda	Red	87899.000000	4.0	$5,000
2	Toyota	Blue	92302.666667	3.0	$7,000
3	BMW	Black	11179.000000	5.0	$22,000
4	Nissan	White	213095.000000	4.0	$3,500
5	Toyota	Green	92302.666667	4.0	$4,500

In [78]:

```
car_sales_missing
```

Out[78]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.000000	4.0	$4,000
1	Honda	Red	87899.000000	4.0	$5,000
2	Toyota	Blue	92302.666667	3.0	$7,000
3	BMW	Black	11179.000000	5.0	$22,000
4	Nissan	White	213095.000000	4.0	$3,500
5	Toyota	Green	92302.666667	4.0	$4,500
6	Honda	NaN	92302.666667	4.0	$7,500
7	Honda	Blue	92302.666667	4.0	NaN
8	Toyota	White	60000.000000	NaN	NaN
9	NaN	White	31600.000000	4.0	$9,700

In [80]:

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

In [81]:

```
car_sales_missing
```

Out[81]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.000000	4.0	$4,000
1	Honda	Red	87899.000000	4.0	$5,000
2	Toyota	Blue	92302.666667	3.0	$7,000
3	BMW	Black	11179.000000	5.0	$22,000
4	Nissan	White	213095.000000	4.0	$3,500
5	Toyota	Green	92302.666667	4.0	$4,500

In [83]:

```
#adding new colomns
seats_column =pd.Series([5,5,5,5,5])
#new coloumn called seats
car_sales["Seats"] = seats_column
car_sales
```

Out[83]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats
0	toyota	White	150043	4	400000	5.0
1	honda	Red	87899	4	500000	5.0
2	toyota	Blue	32549	3	700000	5.0
3	bmw	Black	11179	5	2200000	5.0
4	nissan	White	213095	4	350000	5.0
5	toyota	Green	99213	4	450000	NaN
6	honda	Blue	45698	4	750000	NaN
7	honda	Blue	54738	4	700000	NaN
8	toyota	White	60000	4	625000	NaN
9	nissan	White	31600	4	970000	NaN

In [86]:

```
car_sales["Seats"].fillna(5, inplace =True)
```

In [87]:

```
car_sales
```

Out[87]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats
0	toyota	White	150043	4	400000	5.0
1	honda	Red	87899	4	500000	5.0
2	toyota	Blue	32549	3	700000	5.0
3	bmw	Black	11179	5	2200000	5.0
4	nissan	White	213095	4	350000	5.0
5	toyota	Green	99213	4	450000	5.0
6	honda	Blue	45698	4	750000	5.0
7	honda	Blue	54738	4	700000	5.0
8	toyota	White	60000	4	625000	5.0
9	nissan	White	31600	4	970000	5.0

In [90]:

```
#colomn from list but in this we give all value that means to total row
fuel =[7.5,9.2,5.0,6.5,5.5,7.5,9.1,8.1,7.2,8.8,]
car_sales["fuel per 100KM"] =fuel
car_sales
```

Out[90]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM
0	toyota	White	150043	4	400000	5.0	7.5
1	honda	Red	87899	4	500000	5.0	9.2
2	toyota	Blue	32549	3	700000	5.0	5.0
3	bmw	Black	11179	5	2200000	5.0	6.5
4	nissan	White	213095	4	350000	5.0	5.5
5	toyota	Green	99213	4	450000	5.0	7.5
6	honda	Blue	45698	4	750000	5.0	9.1
7	honda	Blue	54738	4	700000	5.0	8.1
8	toyota	White	60000	4	625000	5.0	7.2
9	nissan	White	31600	4	970000	5.0	8.8

In [91]:

```
car_sales["Fuel used"] =car_sales["Odometer (KM)"]/100 * car_sales["fuel per 100KM"]
```

In [92]:

```
car_sales
```

Out[92]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	Fuel used
0	toyota	White	150043	4	400000	5.0	7.5	11253.225
1	honda	Red	87899	4	500000	5.0	9.2	8086.708
2	toyota	Blue	32549	3	700000	5.0	5.0	1627.450
3	bmw	Black	11179	5	2200000	5.0	6.5	726.635
4	nissan	White	213095	4	350000	5.0	5.5	11720.225
5	toyota	Green	99213	4	450000	5.0	7.5	7440.975
6	honda	Blue	45698	4	750000	5.0	9.1	4158.518
7	honda	Blue	54738	4	700000	5.0	8.1	4433.778
8	toyota	White	60000	4	625000	5.0	7.2	4320.000
9	nissan	White	31600	4	970000	5.0	8.8	2780.800

In [93]:

```
#creating column using single value
car_sales["Number of wheels"] =4
```

In [94]:

```
car_sales
```

Out[94]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	Fuel used	Number of wheels
0	toyota	White	150043	4	400000	5.0	7.5	11253.225	4
1	honda	Red	87899	4	500000	5.0	9.2	8086.708	4
2	toyota	Blue	32549	3	700000	5.0	5.0	1627.450	4
3	bmw	Black	11179	5	2200000	5.0	6.5	726.635	4
4	nissan	White	213095	4	350000	5.0	5.5	11720.225	4
5	toyota	Green	99213	4	450000	5.0	7.5	7440.975	4
6	honda	Blue	45698	4	750000	5.0	9.1	4158.518	4
7	honda	Blue	54738	4	700000	5.0	8.1	4433.778	4
8	toyota	White	60000	4	625000	5.0	7.2	4320.000	4
9	nissan	White	31600	4	970000	5.0	8.8	2780.800	4

In [96]:

```
#to remove coloumn use drop
car_sales.drop("Fuel used", axis=1, inplace=True)
```

In [97]:

```
car_sales
```

Out[97]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	Number of wheels
0	toyota	White	150043	4	400000	5.0	7.5	4
1	honda	Red	87899	4	500000	5.0	9.2	4
2	toyota	Blue	32549	3	700000	5.0	5.0	4
3	bmw	Black	11179	5	2200000	5.0	6.5	4
4	nissan	White	213095	4	350000	5.0	5.5	4
5	toyota	Green	99213	4	450000	5.0	7.5	4
6	honda	Blue	45698	4	750000	5.0	9.1	4
7	honda	Blue	54738	4	700000	5.0	8.1	4
8	toyota	White	60000	4	625000	5.0	7.2	4
9	nissan	White	31600	4	970000	5.0	8.8	4

In [100]:

```
car_sale_suffled=car_sales.sample(frac=1)
#sample mean sample of data frac mean how many perctage we want 1 mean 100per
```

In [101]:

```
car_sale_suffled
```

Out[101]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	Number of wheels
2	toyota	Blue	32549	3	700000	5.0	5.0	4
3	bmw	Black	11179	5	2200000	5.0	6.5	4
1	honda	Red	87899	4	500000	5.0	9.2	4
5	toyota	Green	99213	4	450000	5.0	7.5	4
0	toyota	White	150043	4	400000	5.0	7.5	4
7	honda	Blue	54738	4	700000	5.0	8.1	4
9	nissan	White	31600	4	970000	5.0	8.8	4
4	nissan	White	213095	4	350000	5.0	5.5	4
8	toyota	White	60000	4	625000	5.0	7.2	4
6	honda	Blue	45698	4	750000	5.0	9.1	4

In [103]:

```
car_sale_suffled.reset_index()
```

Out[103]:

	index	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	Number of wheels
0	2	toyota	Blue	32549	3	700000	5.0	5.0	4
1	3	bmw	Black	11179	5	2200000	5.0	6.5	4
2	1	honda	Red	87899	4	500000	5.0	9.2	4
3	5	toyota	Green	99213	4	450000	5.0	7.5	4
4	0	toyota	White	150043	4	400000	5.0	7.5	4
5	7	honda	Blue	54738	4	700000	5.0	8.1	4
6	9	nissan	White	31600	4	970000	5.0	8.8	4
7	4	nissan	White	213095	4	350000	5.0	5.5	4
8	8	toyota	White	60000	4	625000	5.0	7.2	4
9	6	honda	Blue	45698	4	750000	5.0	9.1	4

In [104]:

```
car_sale_suffled.reset_index(drop=True, inplace=True)
#drop remove suffle index
car_sale_suffled
```

Out[104]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	Number of wheels
0	toyota	Blue	32549	3	700000	5.0	5.0	4
1	bmw	Black	11179	5	2200000	5.0	6.5	4
2	honda	Red	87899	4	500000	5.0	9.2	4
3	toyota	Green	99213	4	450000	5.0	7.5	4
4	toyota	White	150043	4	400000	5.0	7.5	4
5	honda	Blue	54738	4	700000	5.0	8.1	4
6	nissan	White	31600	4	970000	5.0	8.8	4
7	nissan	White	213095	4	350000	5.0	5.5	4
8	toyota	White	60000	4	625000	5.0	7.2	4
9	honda	Blue	45698	4	750000	5.0	9.1	4

In [105]:

```
 #apply use to cange value
car_sales["Odometer (KM)"] = car_sales["Odometer (KM)"].apply(lambda x: x/1.6)
```

In [106]:

```
car_sales
```

Out[106]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	fuel per 100KM	Number of wheels
0	toyota	White	93776.875	4	400000	5.0	7.5	4
1	honda	Red	54936.875	4	500000	5.0	9.2	4
2	toyota	Blue	20343.125	3	700000	5.0	5.0	4
3	bmw	Black	6986.875	5	2200000	5.0	6.5	4
4	nissan	White	133184.375	4	350000	5.0	5.5	4
5	toyota	Green	62008.125	4	450000	5.0	7.5	4
6	honda	Blue	28561.250	4	750000	5.0	9.1	4
7	honda	Blue	34211.250	4	700000	5.0	8.1	4
8	toyota	White	37500.000	4	625000	5.0	7.2	4
9	nissan	White	19750.000	4	970000	5.0	8.8	4

In [ ]:

```

```