ex 2 set 1

January 12, 2024

[ ]:
```
EX 2 WORKING WITH DATA USING PANDAS
```

[ ]:
```
DATE 18/12/23
```

[ ]:
```
URK22AI1048
HARI HARAN K
```

[ ]:
```
'''
#AIM
    #DATA MANIPULATION WITH PANDAS
#DESCRIPTION
    Python Pandas is defined as an open-source library that provides␣
 ↪high-performance data
manipulation in Python. Started by Wes McKinney in 2008 out of a need for a␣
 ↪powerful and
flexible quantitative analysis tool, panda has grown into one of the most␣
 ↪popular Python
libraries. Pandas is built on top of the Numpy package, means Numpy is required␣
 ↪for
operating the Pandas

Create a DataFrame
data0 = pd.DataFrame()

Read data from .csv
df = pd.read_csv('filename.csv')

Selection of data from Data Frame
df['col_name'] #Select one column
df[['col_name1','col_name2']] # Select more than one column

Filtering data from Data Frame
df[df['col_name']&gt;value].head()

Filtering missing values
df[ df[ 'col_name' ].isnull() ]
```

```
Manipulating data in Data Frame
 The result of all these aggregation functions applied to a row or column is␣
 ↪always a
number
 Pandas excludes NaN values

 The commonly used aggregate functions are min (), max (), count (), sum (),␣
 ↪mean (),
median (), prod(), std()
# To get aggregate function of all columns
df. aggregate function
# To get aggregate function of spedified columns
df[‘col_name’]. aggregate function

 Apply any binary arithmetical operation (+,-,*,/) to an entire row

s = df[‘col_name’]/100

Sorting the data in Data Frame
 Sort by column name

df.sort_values(by=‘col_name’, ascending = False, inplace = True)

 Sort by index value

df.sort_index(axis = 0, ascending = False, inplace = True)
#inplace=True (overwriting the original DataFrame)
#inplace=False (return a copy of the modi ed DataFrame)

Grouping data in Data Frame
 Group the data according to some criteria
 It is necessary to apply an aggregation function
df.groupby(‘col_name&#39;).count()
df[[‘col_name1’,’col_name2’]].groupby(‘col_name2&#39;).count()

Ranking data in Data Frame
 The rank is returned on the basis of position after sorting.
 Method takes a string input (‘average’, ‘min’, ‘max’, ‘first’, ‘dense’) which
tells pandas what to do with same values.
 Default is average which means assign average of ranks to the similar values.

df.rank()
df[‘col_name&#39;]. rank(ascending = False, method = &#39;first&#39;)
```

[10]:
```
#URK22AI1048 7Q
import pandas as pd
```

```
df = pd.read_csv('Toyota.csv')

max_km = df['KM'].max()

min_weight = df['Weight'].min()

print("Maximum value in the 'KM' column:", max_km)
print("Minimum value in the 'weight' column:", min_weight)
```

```
Maximum value in the 'KM' column: 243000
Minimum value in the 'weight' column: 1015
```

[11]:
```
#URK22AI1048 8Q
import pandas as pd

df = pd.read_csv('Toyota.csv')

mean_cc = df['CC'].mean()

median_cc = df['CC'].median()

print("Mean of the 'CC' column:", mean_cc)
print("Median of the 'CC' column:", median_cc)
```

```
Mean of the 'CC' column: 1610.3587174348697
Median of the 'CC' column: 1600.0
```

[13]:
```
#URK22AI1048 9Q
import pandas as pd

df = pd.read_csv('Toyota.csv')

df.loc[df['CC'] == 2000, 'Horse Power'] = df.loc[df['CC'] == 2000, 'HP'] * 2

print(df)

df.to_csv('updated_data.csv', index=False)
```

```
     Price  Age     KM FuelType   HP  MetColor  Automatic    CC  Doors  \
0    13500   23  46986   Diesel   90         1          0  2000      3
1    13750   23  72937   Diesel   90         1          0  2000      3
2    13950   24  41711   Diesel   90         1          0  2000      3
3    14950   26  48000   Diesel   90         0          0  2000      3
4    13750   30  38500   Diesel   90         0          0  2000      3
..     …   …    …       …   …        …         …   …     …
494  11950   54  58745   Petrol  110         1          0  1600      4
495  11250   52  58596   Petrol  110         1          0  1600      3
496  11750   54  58530   Petrol  110         0          0  1600      5
497  10950   55  58377   Petrol  110         1          0  1600      3
```

```
498  11250   56  58142   Petrol  110         1           0  1600      5

     Weight  Horse Power
0      1165        180.0
1      1165        180.0
2      1165        180.0
3      1165        180.0
4      1170        180.0
..      …           …
494    1035          NaN
495    1045          NaN
496    1075          NaN
497    1050          NaN
498    1080          NaN

[499 rows x 11 columns]
```

[28]:
```
#URK22AI1048 10Q
import pandas as pd

df = pd.read_csv('Toyota.csv')

filtered_df = df[df['MetColor'] == 1]

print(filtered_df)
```

```
     Price  Age     KM FuelType   HP  MetColor  Automatic    CC  Doors  Weight
0    13500   23  46986   Diesel   90         1          0  2000      3    1165
1    13750   23  72937   Diesel   90         1          0  2000      3    1165
2    13950   24  41711   Diesel   90         1          0  2000      3    1165
6    16900   27  94612   Diesel   90         1          0  2000      3    1245
7    18600   30  75889   Diesel   90         1          0  2000      3    1245
..     …   …    …       …   …        …          …   …    …     …
492   9799   51  59000   Petrol   97         1          0  1400      3    1025
494  11950   54  58745   Petrol  110         1          0  1600      4    1035
495  11250   52  58596   Petrol  110         1          0  1600      3    1045
497  10950   55  58377   Petrol  110         1          0  1600      3    1050
498  11250   56  58142   Petrol  110         1          0  1600      5    1080

[372 rows x 10 columns]
```

[29]:
```
#URK22AI1048 11Q
import pandas as pd
df = pd.read_csv('Toyota.csv')
result_df = df.sort_values(by='Age').head(50)[['Price', 'Age', 'CC']]
print(result_df)
```

```
     Price  Age    CC
185  18245    1  1600
```

184	17795	1	1400
183	21500	2	1600
182	21125	2	1400
110	31000	4	2000
111	31275	4	2000
109	32500	4	2000
179	22500	6	1600
177	19950	7	1600
114	22950	7	2000
117	17900	7	1600
181	18700	7	1400
180	18500	7	1600
113	24950	8	2000
115	24990	8	2000
116	21950	8	2000
172	19500	8	1400
112	24950	8	2000
173	18950	8	1400
174	21950	8	1600
175	19950	8	1600
176	18950	8	1600
178	21950	8	1600
171	23750	8	1600
162	19600	9	1600
170	18245	9	1600
169	17795	9	1400
152	18450	10	1400
157	18900	11	1600
98	18750	11	1600
138	23000	11	2000
164	17650	11	1400
104	19450	11	1600
103	18500	11	1600
168	20500	12	1600
153	19500	12	1600
142	19950	13	1600
137	16250	13	1600
133	15950	13	1600
129	15850	13	1600
120	18950	13	1600
102	18500	13	1400
147	24500	13	1600
154	21750	13	1600
122	16350	14	1600
106	18800	14	1600
166	19950	14	1600
149	20950	14	1600
165	19950	14	1600

```
163  19500   14  1600
```

[39]:
```python
#URK22AI1048 12Q
import pandas as pd

df = pd.read_csv('Toyota.csv')

grouped_df = df.groupby(['FuelType', 'MetColor'])

for group_name, group_data in grouped_df:
    print(f"Group: {group_name}")
    print(group_data)
    print("\n")
```

```
Group: ('CNG', 0)
     Price  Age      KM FuelType   HP  MetColor  Automatic    CC  Doors  \
189   7750   43  178858      CNG  110         0          0  1600      3
387   9250   48  142130      CNG  110         0          0  1600      5

     Weight
189    1084
387    1119


Group: ('CNG', 1)
     Price  Age      KM FuelType   HP  MetColor  Automatic    CC  Doors  \
199  11950   39   98823      CNG  110         1          0  1600      5
209  11950   37   82743      CNG  110         1          0  1600      5
223  14950   44   71793      CNG  110         1          0  1600      4
296  12950   44   41499      CNG  110         1          0  1600      5
383   8500   55  150000      CNG  110         1          0  1600      3
436  11500   47   78785      CNG  110         1          0  1600      5

     Weight
199    1119
209    1121
223    1067
296    1103
383    1075
436    1119


Group: ('Diesel', 0)
     Price  Age      KM FuelType   HP  MetColor  Automatic    CC  Doors  \
3    14950   26   48000   Diesel   90         0          0  2000      3
4    13750   30   38500   Diesel   90         0          0  2000      3
5    12950   32   61000   Diesel   90         0          0  2000      3
9    12950   23   71138   Diesel   69         0          0  1900      3
```

```
43   16950   27  110404   Diesel   90         0          0  2000      5
44   16950   22  100250   Diesel   90         0          0  2000      5
45   19000   23   84000   Diesel   90         0          0  2000      5
109  32500    4       1   Diesel  116         0          0  2000      5
186   6950   43  243000   Diesel   69         0          0  1900      3
187   9500   38  180638   Diesel   90         0          0  2000      4
191   4350   44  158320   Diesel   69         0          0  1800      5
216  13500   33   75699   Diesel   69         0          0  1900      3
378   6500   53  216000   Diesel   69         0          0  1900      3
379   6400   51  198167   Diesel   69         0          0  1900      4
380   7000   53  176000   Diesel   69         0          0  1900      3
386  10250   53  143513   Diesel   69         0          0  1900      5
393   4450   56  129155   Diesel   69         0          0  1800      5
406  10950   51  103018   Diesel   69         0          0  1900      5
416   9950   51   96135   Diesel   72         0          0  2000      3
418   8950   55   94401   Diesel   72         0          0  2000      3
447  10995   49   74656   Diesel   69         0          0  1900      3
487   8950   54   61000   Diesel   69         0          0  2000      5

     Weight
3      1165
4      1170
5      1170
9      1105
43     1255
44     1255
45     1270
109    1480
186    1110
187    1160
191    1110
216    1105
378    1110
379    1095
380    1105
386    1140
393    1110
406    1140
416    1115
418    1115
447    1105
487    1140


Group: ('Diesel', 1)
     Price  Age      KM FuelType   HP  MetColor  Automatic    CC  Doors  \
0    13500   23   46986   Diesel   90         1          0  2000      3
1    13750   23   72937   Diesel   90         1          0  2000      3
```

2	13950	24	41711	Diesel	90	1	0	2000	3
6	16900	27	94612	Diesel	90	1	0	2000	3
7	18600	30	75889	Diesel	90	1	0	2000	3
46	17950	27	79375	Diesel	90	1	0	2000	5
48	17950	22	72215	Diesel	90	1	0	2000	5
50	17950	22	62636	Diesel	90	1	0	2000	5
68	22250	22	30000	Diesel	110	1	0	2000	5
87	17950	20	66966	Diesel	90	1	0	2000	3
89	21950	19	50005	Diesel	110	1	0	2000	3
91	22250	20	37500	Diesel	90	1	0	2000	3
92	19950	16	34472	Diesel	90	1	0	1995	3
95	19950	17	30351	Diesel	90	1	0	1995	3
110	31000	4	4000	Diesel	116	1	0	2000	5
111	31275	4	1500	Diesel	116	1	0	2000	5
112	24950	8	13253	Diesel	116	1	0	2000	5
113	24950	8	13253	Diesel	116	1	0	2000	5
114	22950	7	10000	Diesel	116	1	0	2000	5
115	24990	8	6000	Diesel	90	1	0	2000	5
116	21950	8	10841	Diesel	90	1	0	2000	5
118	19250	20	63000	Diesel	90	1	0	2000	5
119	22250	17	57313	Diesel	110	1	0	2000	5
121	19950	19	51099	Diesel	90	1	0	2000	5
138	23000	11	25000	Diesel	116	1	0	2000	5
188	11950	40	179860	Diesel	90	1	0	2000	5
190	11950	40	161000	Diesel	69	1	0	1900	3
192	4750	44	131273	Diesel	69	1	0	1800	5
193	11750	40	130062	Diesel	69	1	0	1900	5
194	13250	41	123425	Diesel	69	1	0	1900	5
196	11900	44	110000	Diesel	69	1	0	1900	5
197	14750	39	108847	Diesel	90	1	0	2000	5
203	10450	35	91456	Diesel	69	1	0	1900	3
204	12950	43	89968	Diesel	69	1	0	1900	5
210	13250	41	81106	Diesel	69	1	0	1900	5
211	14750	40	80425	Diesel	90	1	0	2000	5
214	13500	33	78108	Diesel	90	1	0	2000	3
237	13950	35	59500	Diesel	69	1	0	1900	3
243	13500	33	57711	Diesel	90	1	0	2000	3
268	14750	40	48952	Diesel	90	1	0	2000	5
270	13500	33	48928	Diesel	69	1	0	1900	3
272	13500	35	48052	Diesel	69	1	0	1900	3
381	7750	54	174139	Diesel	72	1	0	2000	4
382	8900	45	174000	Diesel	69	1	0	1900	5
384	8950	54	149329	Diesel	72	1	0	2000	5
388	7750	48	140700	Diesel	69	1	0	1900	5
389	9450	54	138394	Diesel	69	1	0	1900	4
390	7750	55	137000	Diesel	72	1	0	2000	5
391	8250	52	135258	Diesel	69	1	0	1900	5
396	9950	53	117913	Diesel	69	1	0	1900	5

397	12450	47	117430	Diesel	90	1	0	2000	3
401	10500	54	115046	Diesel	69	1	0	1900	5
402	5150	56	113997	Diesel	72	1	0	2000	5
412	8950	48	98100	Diesel	69	1	0	1900	5
422	9250	53	90097	Diesel	69	1	0	1900	5
458	8695	50	70440	Diesel	69	1	0	1900	3
463	8750	47	69000	Diesel	69	1	0	1900	5
465	11450	55	68520	Diesel	72	1	0	2000	3
480	11500	48	63000	Diesel	69	1	0	1900	5

	Weight
0	1165
1	1165
2	1165
6	1245
7	1245
46	1255
48	1255
50	1255
68	1275
87	1245
89	1265
91	1260
92	1260
95	1260
110	1480
111	1480
112	1320
113	1320
114	1270
115	1280
116	1270
118	1255
119	1275
121	1255
138	1320
188	1205
190	1105
192	1110
193	1140
194	1140
196	1095
197	1205
203	1110
204	1140
210	1140
211	1205
214	1170

```
237    1110
243    1165
268    1205
270    1105
272    1105
381    1100
382    1095
384    1135
388    1110
389    1095
390    1135
391    1140
396    1110
397    1165
401    1140
402    1135
412    1140
422    1140
458    1105
463    1140
465    1115
480    1140
```

```
Group: ('Petrol', 0)
     Price  Age     KM FuelType   HP  MetColor  Automatic    CC  Doors  Weight
8    21500   27  19700   Petrol  192         0          0  1800      3    1185
10   20950   25  31461   Petrol  192         0          0  1800      3    1185
11   19950   22  43610   Petrol  192         0          0  1800      3    1185
12   19600   25  32189   Petrol  192         0          0  1800      3    1185
15   22000   28  18739   Petrol  192         0          0  1800      3    1185
..     …  …    …       …  …        …         …   …    …      …
484   9500   54  62519   Petrol   97         0          0  1400      3    1025
486  10750   50  61672   Petrol  110         0          0  1600      5    1075
490  10950   47  60348   Petrol  110         0          0  1600      4    1030
493  11750   51  58761   Petrol   97         0          0  1300      5    1060
496  11750   54  58530   Petrol  110         0          0  1600      5    1075

[103 rows x 10 columns]
```

```
Group: ('Petrol', 1)
     Price  Age     KM FuelType   HP  MetColor  Automatic    CC  Doors  Weight
13   21500   31  23000   Petrol  192         1          0  1800      3    1185
14   22500   32  34131   Petrol  192         1          0  1800      3    1185
16   22750   30  34000   Petrol  192         1          0  1800      3    1185
17   17950   24  21716   Petrol  110         1          0  1600      3    1105
19   16950   30  64359   Petrol  110         1          0  1600      3    1105
```

```
..      …  …    …       …  …        …         …   …    …     …
492   9799   51  59000   Petrol   97         1          0  1400      3    1025
494  11950   54  58745   Petrol  110         1          0  1600      4    1035
495  11250   52  58596   Petrol  110         1          0  1600      3    1045
497  10950   55  58377   Petrol  110         1          0  1600      3    1050
498  11250   56  58142   Petrol  110         1          0  1600      5    1080

[307 rows x 10 columns]
```

[40]:
```
#URK22AI1048 13Q
import pandas as pd

df = pd.read_csv('Toyota.csv')

null_counts = df.isnull().sum()

print("Null counts:")
print(null_counts)

df['CC'].fillna('25', inplace=True)

print("\nDataFrame after filling null values:")
print(df)
```

```
Null counts:
Price        0
Age          0
KM           0
FuelType     0
HP           0
MetColor     0
Automatic    0
CC           0
Doors        0
Weight       0
dtype: int64

DataFrame after filling null values:
     Price  Age     KM FuelType   HP  MetColor  Automatic    CC  Doors  Weight
0    13500   23  46986   Diesel   90         1          0  2000      3    1165
1    13750   23  72937   Diesel   90         1          0  2000      3    1165
2    13950   24  41711   Diesel   90         1          0  2000      3    1165
3    14950   26  48000   Diesel   90         0          0  2000      3    1165
4    13750   30  38500   Diesel   90         0          0  2000      3    1170
..     …  …    …       …  …        …         …   …    …     …
494  11950   54  58745   Petrol  110         1          0  1600      4    1035
```

```
495  11250   52  58596   Petrol  110         1          0  1600      3    1045
496  11750   54  58530   Petrol  110         0          0  1600      5    1075
497  10950   55  58377   Petrol  110         1          0  1600      3    1050
498  11250   56  58142   Petrol  110         1          0  1600      5    1080

[499 rows x 10 columns]
```

[45]:
```
#URK22AI1048 14Q
import pandas as pd

df = pd.read_csv('Toyota.csv')

new_rows = {'Price': [25000, 30000, 18000],
            'Age': [3, 5, 2],
            'CC': [1800, 2200, 1600],
            'Fuel Type': ['Petrol', 'Diesel', 'Petrol'],
            'Metcolor': [1, 0, 1],
            'HP': [120, 150, 100]}


df = df.append(pd.DataFrame(new_rows), ignore_index=True)

print(df)
```

```
     Price  Age       KM FuelType   HP  MetColor  Automatic    CC  Doors  \
0    13500   23  46986.0   Diesel   90       1.0        0.0  2000    3.0
1    13750   23  72937.0   Diesel   90       1.0        0.0  2000    3.0
2    13950   24  41711.0   Diesel   90       1.0        0.0  2000    3.0
3    14950   26  48000.0   Diesel   90       0.0        0.0  2000    3.0
4    13750   30  38500.0   Diesel   90       0.0        0.0  2000    3.0
..     …   …       …       …  …       …         …    …     …
497  10950   55  58377.0   Petrol  110       1.0        0.0  1600    3.0
498  11250   56  58142.0   Petrol  110       1.0        0.0  1600    5.0
499  25000    3      NaN      NaN  120       NaN        NaN  1800    NaN
500  30000    5      NaN      NaN  150       NaN        NaN  2200    NaN
501  18000    2      NaN      NaN  100       NaN        NaN  1600    NaN

     Weight Fuel Type  Metcolor
0    1165.0       NaN       NaN
1    1165.0       NaN       NaN
2    1165.0       NaN       NaN
3    1165.0       NaN       NaN
4    1170.0       NaN       NaN
..      …       …       …
497  1050.0       NaN       NaN
498  1080.0       NaN       NaN
499     NaN    Petrol       1.0
500     NaN    Diesel       0.0
```

```
501     NaN    Petrol       1.0

[502 rows x 12 columns]

/tmp/ipykernel_1340718/2691036192.py:14: FutureWarning: The frame.append method
is deprecated and will be removed from pandas in a future version. Use
pandas.concat instead.
  df = df.append(pd.DataFrame(new_rows), ignore_index=True)
```

[1]:
```
#URK22AI1048 15Q
import pandas as pd

df = pd.read_csv('Toyota.csv')

df = df.drop(columns=['HP'])

print(df)
```

```
     Price  Age     KM FuelType  MetColor  Automatic    CC  Doors  Weight
0    13500   23  46986   Diesel         1          0  2000      3    1165
1    13750   23  72937   Diesel         1          0  2000      3    1165
2    13950   24  41711   Diesel         1          0  2000      3    1165
3    14950   26  48000   Diesel         0          0  2000      3    1165
4    13750   30  38500   Diesel         0          0  2000      3    1170
..     …   …    …       …        …         …   …     …      …
494  11950   54  58745   Petrol         1          0  1600      4    1035
495  11250   52  58596   Petrol         1          0  1600      3    1045
496  11750   54  58530   Petrol         0          0  1600      5    1075
497  10950   55  58377   Petrol         1          0  1600      3    1050
498  11250   56  58142   Petrol         1          0  1600      5    1080

[499 rows x 9 columns]
```

[2]:
```
#URK22AI1048 16Q
import pandas as pd

df = pd.read_csv('Toyota.csv')

df = df.drop([49, 99])

df.reset_index(drop=True, inplace=True)

print(df)
```

```
     Price  Age     KM FuelType   HP  MetColor  Automatic    CC  Doors  Weight
0    13500   23  46986   Diesel   90         1          0  2000      3    1165
1    13750   23  72937   Diesel   90         1          0  2000      3    1165
2    13950   24  41711   Diesel   90         1          0  2000      3    1165
3    14950   26  48000   Diesel   90         0          0  2000      3    1165
```

```
4    13750   30  38500   Diesel   90          0           0  2000       3    1170
..     …   …    …       …   …        …          …   …     …      …
492  11950   54  58745   Petrol  110          1           0  1600       4    1035
493  11250   52  58596   Petrol  110          1           0  1600       3    1045
494  11750   54  58530   Petrol  110          0           0  1600       5    1075
495  10950   55  58377   Petrol  110          1           0  1600       3    1050
496  11250   56  58142   Petrol  110          1           0  1600       5    1080

[497 rows x 10 columns]
```

[3]:
```
#URK22AI1048 17Q
import pandas as pd

df = pd.read_csv('Toyota.csv')


filtered_df = df[df['CC'] > 1600]

result_df = filtered_df[['Price', 'KM', 'HP']].head(100)

print(result_df)
```

```
     Price     KM  HP
0    13500  46986  90
1    13750  72937  90
2    13950  41711  90
3    14950  48000  90
4    13750  38500  90
..     …    …  ..
458   8695  70440  69
463   8750  69000  69
465  11450  68520  72
480  11500  63000  69
487   8950  61000  69

[92 rows x 3 columns]
```

[4]:
```
#URK22AI1048 18Q
import pandas as pd

df = pd.read_csv('Toyota.csv')

row_index = 5
col_name = 'Price'

data_loc = df.loc[row_index, col_name]
print(f"Using loc: {data_loc}")
```

```
row_index = 5
col_index = 2

data_iloc = df.iloc[row_index, col_index]
print(f"Using iloc: {data_iloc}")
```

```
Using loc: 12950
Using iloc: 61000
```

[6]:
```
#URK22AI1048 19Q
import pandas as pd
df = pd.read_csv('Toyota.csv')
print("Index:")
print(df.index)

print("\nColumn Names:")
print(df.columns)

print("\nDescriptive Statistics:")
print(df.describe())

print("\nSize:")
print(df.size)

print("\nNumber of Dimensions (ndim):")
print(df.ndim)

print("\nShape:")
print(df.shape)

print("\nInfo:")
print(df.info())
```

```
Index:
RangeIndex(start=0, stop=499, step=1)

Column Names:
Index(['Price', 'Age', 'KM', 'FuelType', 'HP', 'MetColor', 'Automatic', 'CC',
       'Doors', 'Weight'],
      dtype='object')

Descriptive Statistics:
              Price         Age             KM          HP    MetColor  \
count    499.000000  499.000000     499.000000  499.000000  499.000000
mean   14072.026052   35.124248   53237.234469  103.951904    0.745491
std     4059.047783   13.815937   36963.132169   17.882192    0.436022
min     4350.000000    1.000000       1.000000   69.000000    0.000000
25%    11450.000000   25.000000   27910.500000   97.000000    0.000000
```

```
50%    12950.000000   39.000000   45725.000000  110.000000    1.000000
75%    16700.000000   44.000000   71759.000000  110.000000    1.000000
max    32500.000000   56.000000  243000.000000  192.000000    1.000000

         Automatic           CC       Doors       Weight
count  499.000000   499.000000  499.000000   499.000000
mean     0.044088  1610.358717    4.236473  1099.643287
std      0.205497   180.135440    0.933638    66.825080
min      0.000000  1300.000000    3.000000  1015.000000
25%      0.000000  1400.000000    3.000000  1060.000000
50%      0.000000  1600.000000    5.000000  1080.000000
75%      0.000000  1600.000000    5.000000  1119.000000
max      1.000000  2000.000000    5.000000  1615.000000

Size:
4990

Number of Dimensions (ndim):
2

Shape:
(499, 10)

Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 499 entries, 0 to 498
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Price       499 non-null    int64
 1   Age         499 non-null    int64
 2   KM          499 non-null    int64
 3   FuelType    499 non-null    object
 4   HP          499 non-null    int64
 5   MetColor    499 non-null    int64
 6   Automatic   499 non-null    int64
 7   CC          499 non-null    int64
 8   Doors       499 non-null    int64
 9   Weight      499 non-null    int64
dtypes: int64(9), object(1)
memory usage: 39.1+ KB
None
```

[ ]:
```
'''
RESULT
The above programs were created and executed successfully.
'''
```