Working on Real Project with Python

(A part of Big Data Analysis)

Cars Dataset

Here, The data of different cars is given with their specifictions This data is available as a CSV file. We are going to analyze this data set using the Pandas Data Frame

1) Introduction (for Data cleaning) . Find all null value in the dataset.If there is any null value in my column, then fill it with mean of that column

In [2]:
```
import pandas as pd
```

In [3]:
```
car=pd.read_csv(r"E:\python\2. Cars Data1.csv")
```

In [4]:
```
car
```

Out[4]:

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Hors
0	Acura	MDX	SUV	Asia	All	$36,945	$33,337	3.5	6.0	
1	Acura	RSX Type S 2dr	Sedan	Asia	Front	$23,820	$21,761	2.0	4.0	
2	Acura	TSX 4dr	Sedan	Asia	Front	$26,990	$24,647	2.4	4.0	
3	Acura	TL 4dr	Sedan	Asia	Front	$33,195	$30,299	3.2	6.0	
4	Acura	3.5 RL 4dr	Sedan	Asia	Front	$43,755	$39,014	3.5	6.0	
...	...	...	...	...	...	...	...	...	...	...
427	Volvo	C70 LPT convertible 2dr	Sedan	Europe	Front	$40,565	$38,203	2.4	5.0	
428	Volvo	C70 HPT convertible 2dr	Sedan	Europe	Front	$42,565	$40,083	2.3	5.0	
429	Volvo	S80 T6 4dr	Sedan	Europe	Front	$45,210	$42,573	2.9	6.0	
430	Volvo	V40	Wagon	Europe	Front	$26,135	$24,641	1.9	4.0	
431	Volvo	XC70	Wagon	Europe	All	$35,145	$33,112	2.5	5.0	

432 rows × 15 columns

In [5]:
```
1 car.head()
```

Out[5]:

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower
0	Acura	MDX	SUV	Asia	All	$36,945	$33,337	3.5	6.0	265.0
1	Acura	RSX Type S 2dr	Sedan	Asia	Front	$23,820	$21,761	2.0	4.0	200.0
2	Acura	TSX 4dr	Sedan	Asia	Front	$26,990	$24,647	2.4	4.0	200.0
3	Acura	TL 4dr	Sedan	Asia	Front	$33,195	$30,299	3.2	6.0	270.0
4	Acura	3.5 RL 4dr	Sedan	Asia	Front	$43,755	$39,014	3.5	6.0	225.0

In [11]:
```
1 car.shape
```

Out[11]: (432, 15)

In [14]:
```
1 df=car.dropna(how='all')
```

In [15]:
```
1 df.shape
```

Out[15]: (428, 15)

In [16]:
```
1 df.isnull()
```

Out[16]:

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower
0	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...
427	False	False	False	False	False	False	False	False	False	False
428	False	False	False	False	False	False	False	False	False	False
429	False	False	False	False	False	False	False	False	False	False
430	False	False	False	False	False	False	False	False	False	False
431	False	False	False	False	False	False	False	False	False	False

428 rows × 15 columns

In [24]:
```
1 df.isnull().sum()
```

Out[24]:
```
Make          0
Model         0
Type          0
Origin        0
DriveTrain    0
MSRP          0
Invoice       0
EngineSize    0
Cylinders     0
Horsepower    0
MPG_City      0
MPG_Highway   0
Weight        0
Wheelbase     0
Length        0
dtype: int64
```

In [25]:
```
1 df['Cylinders'].fillna(df['Cylinders'].mean(),inplace = True)  ## for chan
```

```
C:\Users\Desh Deepak Verma\AppData\Local\Temp\ipykernel_12476\1103984115.py:
1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  df['Cylinders'].fillna(df['Cylinders'].mean(),inplace = True)
```

In [23]:
```
1 df.isnull().sum()
```

Out[23]:
```
Make          0
Model         0
Type          0
Origin        0
DriveTrain    0
MSRP          0
Invoice       0
EngineSize    0
Cylinders     0
Horsepower    0
MPG_City      0
MPG_Highway   0
Weight        0
Wheelbase     0
Length        0
dtype: int64
```

2) Question(Based on value count funtion) -check what are the different types of make are there in our dataset. And what is the count (occurrence) of each make in data?

In [26]:
```
1 df.head(2)
```

Out[26]:

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower
0	Acura	MDX	SUV	Asia	All	$36,945	$33,337	3.5	6.0	265.0
1	Acura	RSX Type S 2dr	Sedan	Asia	Front	$23,820	$21,761	2.0	4.0	200.0

In [28]:
```
1 df['Make'].value_counts()
```

Out[28]:
```
Toyota           28
Chevrolet        27
Mercedes-Benz    26
Ford             23
BMW              20
Audi             19
Honda            17
Nissan           17
Volkswagen       15
Chrysler         15
Dodge            13
Mitsubishi       13
Volvo            12
Jaguar           12
Hyundai          12
Subaru           11
Pontiac          11
Mazda            11
Lexus            11
Kia              11
Buick             9
Mercury           9
Lincoln           9
Saturn            8
Cadillac          8
Suzuki            8
Infiniti          8
GMC               8
Acura             7
Porsche           7
Saab              7
Land Rover        3
Oldsmobile        3
Jeep              3
Scion             2
Isuzu             2
MINI              2
Hummer            1
Name: Make, dtype: int64
```

3) Instruction (Filtering)

-Show all the records where Origin is Asia or Europe

In [29]:
```
1 df.head(2)
```

Out[29]:

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower
0	Acura	MDX	SUV	Asia	All	$36,945	$33,337	3.5	6.0	265.0
1	Acura	RSX Type S 2dr	Sedan	Asia	Front	$23,820	$21,761	2.0	4.0	200.0

In [31]:
```
1 df[df['Origin'].isin(['Asia','Europe'])]
```

Out[31]:

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Hors
0	Acura	MDX	SUV	Asia	All	$36,945	$33,337	3.5	6.0	
1	Acura	RSX Type S 2dr	Sedan	Asia	Front	$23,820	$21,761	2.0	4.0	
2	Acura	TSX 4dr	Sedan	Asia	Front	$26,990	$24,647	2.4	4.0	
3	Acura	TL 4dr	Sedan	Asia	Front	$33,195	$30,299	3.2	6.0	
4	Acura	3.5 RL 4dr	Sedan	Asia	Front	$43,755	$39,014	3.5	6.0	
...	...	...	...	...	...	...	...	...	...	...
427	Volvo	C70 LPT convertible 2dr	Sedan	Europe	Front	$40,565	$38,203	2.4	5.0	
428	Volvo	C70 HPT convertible 2dr	Sedan	Europe	Front	$42,565	$40,083	2.3	5.0	
429	Volvo	S80 T6 4dr	Sedan	Europe	Front	$45,210	$42,573	2.9	6.0	
430	Volvo	V40	Wagon	Europe	Front	$26,135	$24,641	1.9	4.0	
431	Volvo	XC70	Wagon	Europe	All	$35,145	$33,112	2.5	5.0	

281 rows × 15 columns

4) Instruction (Removing unwanted records)

- Remove all the records(rows)where Weight is above 4000.

In [32]:
```
1 df.head(2)
```

Out[32]:

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower
0	Acura	MDX	SUV	Asia	All	$36,945	$33,337	3.5	6.0	265.0
1	Acura	RSX Type S 2dr	Sedan	Asia	Front	$23,820	$21,761	2.0	4.0	200.0

In [40]:
```
df[~(df['Weight']>4000)]
```

Out[40]:

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Ho
1	Acura	RSX Type S 2dr	Sedan	Asia	Front	$23,820	$21,761	2.0	4.0	
2	Acura	TSX 4dr	Sedan	Asia	Front	$26,990	$24,647	2.4	4.0	
3	Acura	TL 4dr	Sedan	Asia	Front	$33,195	$30,299	3.2	6.0	
4	Acura	3.5 RL 4dr	Sedan	Asia	Front	$43,755	$39,014	3.5	6.0	
5	Acura	3.5 RL w/Navigation 4dr	Sedan	Asia	Front	$46,100	$41,100	3.5	6.0	
...	...	...	...	...	...	...	...	...	...	...
427	Volvo	C70 LPT convertible 2dr	Sedan	Europe	Front	$40,565	$38,203	2.4	5.0	
428	Volvo	C70 HPT convertible 2dr	Sedan	Europe	Front	$42,565	$40,083	2.3	5.0	
429	Volvo	S80 T6 4dr	Sedan	Europe	Front	$45,210	$42,573	2.9	6.0	
430	Volvo	V40	Wagon	Europe	Front	$26,135	$24,641	1.9	4.0	
431	Volvo	XC70	Wagon	Europe	All	$35,145	$33,112	2.5	5.0	

325 rows × 15 columns

In [42]:
```
df.shape
```

Out[42]:
```
(428, 15)
```

In [43]:
```
428-103
```

Out[43]:
```
325
```

5) Instruction (Aplying Function on a column)

-Increase all the values of MPG_city column by 3.

In [44]:
```
df.head(2)
```

Out[44]:

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower
0	Acura	MDX	SUV	Asia	All	$36,945	$33,337	3.5	6.0	265.0
1	Acura	RSX Type S 2dr	Sedan	Asia	Front	$23,820	$21,761	2.0	4.0	200.0

In [46]:
```
1 df['MPG_City']=df['MPG_City'].apply(lambda x:x+3)
```

```
C:\Users\Desh Deepak Verma\AppData\Local\Temp\ipykernel_12476\1173862363.py:
1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df['MPG_City']=df['MPG_City'].apply(lambda x:x+3)
```

In [47]:
```
1 df
```

Out[47]:

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Hors
0	Acura	MDX	SUV	Asia	All	$36,945	$33,337	3.5	6.0	
1	Acura	RSX Type S 2dr	Sedan	Asia	Front	$23,820	$21,761	2.0	4.0	
2	Acura	TSX 4dr	Sedan	Asia	Front	$26,990	$24,647	2.4	4.0	
3	Acura	TL 4dr	Sedan	Asia	Front	$33,195	$30,299	3.2	6.0	
4	Acura	3.5 RL 4dr	Sedan	Asia	Front	$43,755	$39,014	3.5	6.0	
...	...	...	...	...	...	...	...	...	...	...
427	Volvo	C70 LPT convertible 2dr	Sedan	Europe	Front	$40,565	$38,203	2.4	5.0	
428	Volvo	C70 HPT convertible 2dr	Sedan	Europe	Front	$42,565	$40,083	2.3	5.0	
429	Volvo	S80 T6 4dr	Sedan	Europe	Front	$45,210	$42,573	2.9	6.0	
430	Volvo	V40	Wagon	Europe	Front	$26,135	$24,641	1.9	4.0	
431	Volvo	XC70	Wagon	Europe	All	$35,145	$33,112	2.5	5.0	

428 rows × 15 columns

In [ ]:
```
1
```

In [ ]:
```
1
```