In [1]:
```
import pandas as pd
```

read a csv file

In [2]:
```
data = pd.read_csv('Automobile_data.csv')
data
```

Out[2]:

	company	body-style	horsepower	average-mileage	price
0	alfa-romero	convertible	111	21	13495
1	alfa-romero	convertible	111	21	16500
2	alfa-romero	hatchback	154	19	16500
3	audi	sedan	102	24	13950
4	audi	sedan	115	18	17450
5	audi	sedan	110	19	15250
6	audi	wagon	110	19	18920
7	bmw	sedan	101	23	16430

displays from top 5 rows

In [3]:
```
data.head(5)
```

Out[3]:

	company	body-style	horsepower	average-mileage	price
0	alfa-romero	convertible	111	21	13495
1	alfa-romero	convertible	111	21	16500
2	alfa-romero	hatchback	154	19	16500
3	audi	sedan	102	24	13950
4	audi	sedan	115	18	17450

displays from bottom 5 rows

In [4]:
```
data.tail(5)
```

Out[4]:

	company	body-style	horsepower	average-mileage	price
3	audi	sedan	102	24	13950
4	audi	sedan	115	18	17450
5	audi	sedan	110	19	15250
6	audi	wagon	110	19	18920
7	bmw	sedan	101	23	16430

shows any ramdom rows

In [5]:
```
data.sample(5)
```

Out[5]:

	company	body-style	horsepower	average-mileage	price

	company	body-style	horsepower	average-mileage	price
3	audi	sedan	102	24	13950
5	audi	sedan	110	19	15250
2	alfa-romero	hatchback	154	19	16500
7	bmw	sedan	101	23	16430
6	audi	wagon	110	19	18920

shows the data types

In [6]:
```
data.dtypes
```

Out[6]:
```
company            object
body-style         object
horsepower          int64
average-mileage     int64
price               int64
dtype: object
```

gives the index

In [7]:
```
data.index
```

Out[7]:
```
RangeIndex(start=0, stop=8, step=1)
```

gives all the column names

In [8]:
```
data.columns
```

Out[8]:
```
Index(['company', 'body-style', 'horsepower', 'average-mileage', 'price'], dtype='
ject')
```

gives all the values from the data table

In [9]:
```
data.values
```

Out[9]:
```
array([['alfa-romero', 'convertible', 111, 21, 13495],
       ['alfa-romero', 'convertible', 111, 21, 16500],
       ['alfa-romero', 'hatchback', 154, 19, 16500],
       ['audi', 'sedan', 102, 24, 13950],
       ['audi', 'sedan', 115, 18, 17450],
       ['audi', 'sedan', 110, 19, 15250],
       ['audi', 'wagon', 110, 19, 18920],
       ['bmw', 'sedan', 101, 23, 16430]], dtype=object)
```

calculates some statistical data

In [10]:
```
data.describe
```

Out[10]:
```
<bound method NDFrame.describe of        company   body-style  horsepower  average
ileage  price
0  alfa-romero  convertible         111               21  13495
1  alfa-romero  convertible         111               21  16500
2  alfa-romero    hatchback         154               19  16500
3         audi        sedan         102               24  13950
4         audi        sedan         115               18  17450
5         audi        sedan         110               19  15250
6         audi        wagon         110               19  18920
7          bmw        sedan         101               23  16430>
```

delete duplicate rows in which subset means column name and keep means keep the first row and remove duplicate

In [11]:
```
data_dup = data.drop_duplicates(subset='company', keep="first")
data_dup
```

Out[11]:

	company	body-style	horsepower	average-mileage	price
0	alfa-romero	convertible	111	21	13495
3	audi	sedan	102	24	13950
7	bmw	sedan	101	23	16430

Transposes whole table : rows as col and col as rows

In [12]:
```
T = data.T
T
```

Out[12]:

	0	1	2	3	4	5	6	7
company	alfa-romero	alfa-romero	alfa-romero	audi	audi	audi	audi	bmw
body-style	convertible	convertible	hatchback	sedan	sedan	sedan	wagon	sedan
horsepower	111	111	154	102	115	110	110	101
average-mileage	21	21	19	24	18	19	19	23
price	13495	16500	16500	13950	17450	15250	18920	16430

shows specific columns with data

In [13]:
```
data[['company','price']]
```

Out[13]:

	company	price
0	alfa-romero	13495
1	alfa-romero	16500
2	alfa-romero	16500
3	audi	13950
4	audi	17450
5	audi	15250
6	audi	18920
7	bmw	16430

shows in which row 'audi' is present

In [14]:
```
data.company=='audi'
```

Out[14]:
```
0    False
1    False
2    False
3     True
4     True
5     True
6     True
7    False
Name: company, dtype: bool
```

select rows and col with labels

In [15]:
```
data.loc[1:3,['company','price']]
```

Out[15]:

	company	price
1	alfa-romero	16500
2	alfa-romero	16500
3	audi	13950

select rows with indexes

In [16]:
```
data.iloc[[0,2]]
```

Out[16]:

	company	body-style	horsepower	average-mileage	price
0	alfa-romero	convertible	111	21	13495
2	alfa-romero	hatchback	154	19	16500

finding with condition : price less than 15000

In [17]:
```
data[data["price"] < 15000]
```

Out[17]:

	company	body-style	horsepower	average-mileage	price
0	alfa-romero	convertible	111	21	13495
3	audi	sedan	102	24	13950

dataset of variable data copied to other variable i.e data_1

In [18]:
```
data_1 = data.copy
data_1
```

Out[18]:
```
<bound method NDFrame.copy of         company   body-style  horsepower  average-mil
ge  price
0  alfa-romero  convertible        111                21  13495
1  alfa-romero  convertible        111                21  16500
2  alfa-romero    hatchback        154                19  16500
3         audi        sedan        102                24  13950
4         audi        sedan        115                18  17450
5         audi        sedan        110                19  15250
6         audi        wagon        110                19  18920
7          bmw        sedan        101                23  16430>
```

added a new column

In [19]:
```
data["color"] = ["Red","Yellow","Green","Black","Red","Yellow","Black","Green"]
data
```

Out[19]:

	company	body-style	horsepower	average-mileage	price	color
0	alfa-romero	convertible	111	21	13495	Red
1	alfa-romero	convertible	111	21	16500	Yellow
2	alfa-romero	hatchback	154	19	16500	Green
3	audi	sedan	102	24	13950	Black
4	audi	sedan	115	18	17450	Red

	company	body-style	horsepower	average-mileage	price	color
5	audi	sedan	110	19	15250	Yellow
6	audi	wagon	110	19	18920	Black
7	bmw	sedan	101	23	16430	Green

droped the 'color' column AXIS = 0 is row & AXIS = 1 is column

In [20]:
```
data.drop(['color'], axis = 1 , inplace=True)
```

finding first 5 rows having null values or not

In [21]:
```
pd.isna(data.head(5))
```

Out[21]:

	company	body-style	horsepower	average-mileage	price
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False

finds the mean value columnwise

In [22]:
```
data.mean()
```

Out[22]:
```
horsepower            114.250
average-mileage        20.500
price               16061.875
dtype: float64
```

finding Cumulative Sum

In [23]:
```
data.cumsum()
```

Out[23]:

	company	body-style	horsepower	average-mileage	p
0	alfa-romero	convertible	111	21	13
1	alfa-romeroalfa-romero	convertibleconvertible	222	42	29
2	alfa-romeroalfa-romeroalfa-romero	convertibleconvertiblehatchback	376	61	46
3	alfa-romeroalfa-romeroalfa-romeroaudi	convertibleconvertiblehatchbacksedan	478	85	60
4	alfa-romeroalfa-romeroalfa-romeroaudiaudi	convertibleconvertiblehatchbacksedansedan	593	103	77
5	alfa-romeroalfa-romeroalfa-romeroaudiaudiaudi	convertibleconvertiblehatchbacksedansedansedan	703	122	93
6	alfa-romeroalfa-romeroalfa-romeroaudiaudiaudiaudi	convertibleconvertiblehatchbacksedansedansedan...	813	141	112

	company	body-style	horsepower	average-mileage	p
7	alfa-romeroalfa-romeroalfa-romeroaudiaudiaudia...	convertibleconvertiblehatchbacksedansedansedan...	914	164	128

finding Cumulative Sum of particular column

In [24]:
```
to_be_cumsum = data[['horsepower','average-mileage','price']]
to_be_cumsum.cumsum()
```

Out[24]:

	horsepower	average-mileage	price
0	111	21	13495
1	222	42	29995
2	376	61	46495
3	478	85	60445
4	593	103	77895
5	703	122	93145
6	813	141	112065
7	914	164	128495

finding Cumulative product of particular column

In [25]:
```
to_be_cumsum.cumprod()
```

Out[25]:

	horsepower	average-mileage	price
0	111	21	13495
1	12321	441	222667500
2	1897434	8379	3674013750000
3	193538268	201096	51252491812500000
4	22256900820	3619728	8912266590066522432
5	2448259090200	68774832	-3544836577509218688
6	269308499922000	1306721808	4053405533512098816
7	27200158492122000	30054601584	4706809512302213120

converts the data into a stacked form

In [26]:
```
stacked = data.stack()
stacked
```

Out[26]:
```
0  company              alfa-romero
   body-style           convertible
   horsepower                   111
   average-mileage               21
   price                      13495
1  company              alfa-romero
   body-style           convertible
   horsepower                111
   average-mileage            21
   price                   16500
2  company           alfa-romero
   body-style          hatchback
   horsepower                154
   average-mileage            19
   price                   16500
3  company                  audi
   body-style              sedan
   horsepower                102
   average-mileage            24
   price                   13950
4  company                  audi
   body-style              sedan
   horsepower                115
   average-mileage            18
   price                   17450
5  company                  audi
   body-style              sedan
   horsepower                110
   average-mileage            19
   price                   15250
6  company                  audi
   body-style              wagon
   horsepower                110
   average-mileage            19
   price                   18920
7  company                   bmw
   body-style              sedan
   horsepower                101
   average-mileage            23
   price                   16430
dtype: object
```

Convert the price to a categorical data type

In [27]:
```
data["price"].astype("category")
```

Out[27]:
```
0    13495
1    16500
2    16500
3    13950
4    17450
5    15250
6    18920
7    16430
Name: price, dtype: category
Categories (7, int64): [13495, 13950, 15250, 16430, 16500, 17450, 18920]
```

sorting the data of column of "average-mileage"

In [28]:
```
data.sort_values(by="average-mileage")
```

Out[28]:

	company	body-style	horsepower	average-mileage	price
4	audi	sedan	115	18	17450
2	alfa-romero	hatchback	154	19	16500
5	audi	sedan	110	19	15250
6	audi	wagon	110	19	18920
0	alfa-romero	convertible	111	21	13495
1	alfa-romero	convertible	111	21	16500
7	bmw	sedan	101	23	16430

	company	body-style	horsepower	average-mileage	price
3	audi	sedan	102	24	13950

gives the size of the data, in this avg_milegae of 19 is 3 times

In [29]:
```
data.groupby("average-mileage").size()
```

Out[29]:
```
average-mileage
18    1
19    3
21    2
23    1
24    1
dtype: int64
```

save to csv file . index=False means there will be no first column

In [30]:
```
data.to_csv('data_to_csv.csv', index=False)
```

In [ ]:
```
```