In [1]:

```
# Importing Libraries
from bs4 import BeautifulSoup as bs
import requests
import re
import pandas as pd
```

In [2]:

```
# Storing link
link = 'https://www.autocarindia.com/car-news/top-10-selling-cars-suvs-in-india-in-fy2022
-424413'
```

In [3]:

```
# Checking and storing link's response
page = requests.get(link)
page
```

Out[3]:

```
<Response [200]>
```

In [4]:

```
# Storing page content
soup=bs(page.content,"html.parser")
```

In [5]:

```
# Prining the content
# print(soup.prettify())
```

Title

In [6]:

```
# Scraping and storing the title
title = soup.title
print(title.get_text())
```

```
Top-10 best selling cars, SUVs in India in FY2022 | Autocar India
```

In [7]:

```
# Scraping name of the cars using find_all
car_name = soup.find_all("span", style = "color:#ff0000")
car_name
```

Out[7]:

```
[<span style="color:#ff0000">Maruti Suzuki Wagon R: 1,88,838 units sold</span>,
 <span style="color:#ff0000">Maruti Suzuki Swift: 1,67,827 units sold</span>,
 <span style="color:#ff0000">Maruti Suzuki Baleno: 1,48,187 units sold </span>,
 <span style="color:#ff0000">Maruti Suzuki Alto: 1,45,167 units sold </span>,
 <span style="color:#ff0000">Tata Nexon: 1,24,130 units sold </span>,
 <span style="color:#ff0000">Hyundai Creta: 1,18,092 units sold</span>,
 <span style="color:#ff0000">Maruti Suzuki Ertiga: 1,17,150 units sold </span>,
 <span style="color:#ff0000">Maruti Suzuki Vitara Brezza: 1,13,751 units sold </span>,
 <span style="color:#ff0000">Maruti Suzuki Dzire: 1,08,564 units sold</span>,
 <span style="color:#ff0000">Maruti Eeco: 1,08,345 units sold</span>]
```

In [8]:

```
# Extracting car names from scraped data
car_names = []
```

```
for i in range(len(car_name)):
    car_names.append(car_name[i].get_text().split(':')[0])
car_names
```

Out[8]:

```
['Maruti Suzuki Wagon R',
 'Maruti Suzuki Swift',
 'Maruti Suzuki Baleno',
 'Maruti Suzuki Alto',
 'Tata Nexon',
 'Hyundai Creta',
 'Maruti Suzuki Ertiga',
 'Maruti Suzuki Vitara Brezza',
 'Maruti Suzuki Dzire',
 'Maruti Eeco']
```

In [9]:

```
# Extracting scrapped units sold of each car
unit_sold = []
for i in range(len(car_name)):
    unit_sold.append(car_name[i].get_text().strip().split(':')[1])
unit_sold
```

Out[9]:

```
[' 1,88,838 units sold',
 ' 1,67,827 units sold',
 ' 1,48,187 units sold',
 ' 1,45,167 units sold',
 ' 1,24,130 units sold',
 ' 1,18,092 units sold',
 ' 1,17,150 units sold',
 ' 1,13,751 units sold',
 ' 1,08,564 units sold',
 ' 1,08,345 units sold']
```

In [10]:

```
# Extracting only digit values using regular expression
pattern = re.compile(r"\d,\d\d,\d\d\d")
units = []
for i in unit_sold:
    units.append(pattern.search(i))
unit_s = [units[k].group() for k in range(len(units))] #Using list comprehension to prin
t all the values
unit_s
```

Out[10]:

```
['1,88,838',
 '1,67,827',
 '1,48,187',
 '1,45,167',
 '1,24,130',
 '1,18,092',
 '1,17,150',
 '1,13,751',
 '1,08,564',
 '1,08,345']
```

In [11]:

```
# Removing commas and converting data-type to integer
int_unit = []
for i in range(len(units)):
    int_unit.append(int(units[i].group().replace(',','')))
print(int_unit)
```

```
[188838, 167827, 148187, 145167, 124130, 118092, 117150, 113751, 108564, 108345]
```

In [12]:

```
# Storing links of cars
link2 = 'https://www.zigwheels.com/newcars/Maruti-Suzuki/wagon-r#leadform'
link3 = 'https://www.zigwheels.com/newcars/Maruti-Suzuki/Swift'
link4 = 'https://www.zigwheels.com/newcars/Maruti-Suzuki/baleno'
link5 = 'https://www.zigwheels.com/newcars/Maruti-Suzuki/Alto-800'
link6 = 'https://www.zigwheels.com/newcars/Tata/nexon'
link7 = 'https://www.zigwheels.com/newcars/Hyundai/creta'
link8 = 'https://www.zigwheels.com/newcars/Maruti-Suzuki/ertiga'
link9 = 'https://www.zigwheels.com/newcars/Maruti-Suzuki/vitara-brezza'
link10 = 'https://www.zigwheels.com/newcars/Maruti-Suzuki/dzire'
link11 = 'https://www.zigwheels.com/newcars/Maruti-Suzuki/Eeco'
```

In [13]:

```
# Checking and storing link's responses
page2 = requests.get(link2)
page3 = requests.get(link3)
page4 = requests.get(link4)
page5 = requests.get(link5)
page6 = requests.get(link6)
page7 = requests.get(link7)
page8 = requests.get(link8)
page9 = requests.get(link9)
page10 = requests.get(link10)
page11 = requests.get(link11)
print("page2: " ,page2)
print("page3: " ,page3)
print("page4: " ,page4)
print("page5: " ,page5)
print("page6: " ,page6)
print("page7: " ,page7)
print("page8: " ,page8)
print("page9: " ,page9)
print("page10: " ,page10)
print("page11: " ,page11)
```

```
page2:  <Response [200]>
page3:  <Response [200]>
page4:  <Response [200]>
page5:  <Response [200]>
page6:  <Response [200]>
page7:  <Response [200]>
page8:  <Response [200]>
page9:  <Response [200]>
page10:  <Response [200]>
page11:  <Response [200]>
```

In [14]:

```
# Storing page content individually
soup2=bs(page2.content,"html.parser")
soup3=bs(page3.content,"html.parser")
soup4=bs(page4.content,"html.parser")
soup5=bs(page5.content,"html.parser")
soup6=bs(page6.content,"html.parser")
soup7=bs(page7.content,"html.parser")
soup8=bs(page8.content,"html.parser")
soup9=bs(page9.content,"html.parser")
soup10=bs(page10.content,"html.parser")
soup11=bs(page11.content,"html.parser")
# print(soup2.prettify())
```

In [15]:

```
# Creating list of all pages
soups = [soup2, soup3, soup4, soup5, soup6, soup7, soup8, soup9, soup10, soup11]
```

In [16]:

```
# Scraping description of all cars using find()
```

```
price1 = soup2.find("div", class_ = "clr-pry fnt-14 lh-22", itemprop = "description").get
_text()
price2 = soup3.find("div", class_ = "clr-pry fnt-14 lh-22", itemprop = "description").get
_text()
price3 = soup4.find("div", class_ = "clr-pry fnt-14 lh-22", itemprop = "description").get
_text()
price4 = soup5.find("div", class_ = "clr-pry fnt-14 lh-22", itemprop = "description").get
_text()
price5 = soup6.find("div", class_ = "pl-15 pr-15").get_text()
price6 = soup7.find("div", class_ = "pl-15 pr-15").get_text()
price7 = soup8.find("div", class_ = "clr-pry fnt-14 lh-22", itemprop = "description").get
_text()
price8 = soup9.find("div", class_ = "clr-pry fnt-14 lh-22", itemprop = "description").get
_text()
price9 = soup10.find("div", class_ = "clr-pry fnt-14 lh-22", itemprop = "description").g
et_text()
price10 = soup11.find("div", class_ = "clr-pry fnt-14 lh-22", itemprop = "description").
get_text()
pri_ces = [price1, price2, price3, price4, price7, price8, price9, price10]
pri_ces2 = [price5, price6]
# print(price6)
```

In [17]:

```
# Scraping prices of all cars
prices = []
pattern = re.compile(r"\d.\d\d Lakh to \d.\d\d Lakh")
pattern2 = re.compile(r"\d.\d\d Lakh to \d\d.\d\d Lakh")
def price_scrapper(pr):
    try:
        prices.append(pattern.search(pr).group().replace('to', '-'))
    except AttributeError:
        prices.append(pattern2.search(pr).group().replace('to', '-'))
for i in pri_ces:
    price_scrapper(i)
print(prices)
```

```
['5.47 Lakh - 7.20 Lakh', '5.91 Lakh - 8.85 Lakh', '6.49 Lakh - 9.71 Lakh', '3.39 Lakh -
5.03 Lakh', '8.41 Lakh - 12.79 Lakh', '7.99 Lakh - 13.96 Lakh', '6.24 Lakh - 9.17 Lakh',
'4.63 Lakh - 5.94 Lakh']
```

In [18]:

```
# Scraping prices of all cars having NoneType
prices2 = []
pat = re.compile(r"\d.\d\d Lakh to Rs. \d.\d\d Lakh")
pat2 = re.compile(r"\d.\d\d Lakh to Rs. \d\d.\d\d Lakh")
def price_scrapper2(pr):
    try:
        prices2.append(pat.search(pr).group().replace('to Rs.', '-'))
    except AttributeError:
        prices2.append(pat2.search(pr).group().replace('to Rs.', '-'))
for i in pri_ces2:
    price_scrapper2(i)
# print(prices2)
prices.insert(4,prices2[0])
prices.insert(5,prices2[1])
print(prices)
```

```
['5.47 Lakh - 7.20 Lakh', '5.91 Lakh - 8.85 Lakh', '6.49 Lakh - 9.71 Lakh', '3.39 Lakh -
5.03 Lakh', '7.59 Lakh - 12.77 Lakh', '0.44 Lakh - 18.15 Lakh', '8.41 Lakh - 12.79 Lakh',
'7.99 Lakh - 13.96 Lakh', '6.24 Lakh - 9.17 Lakh', '4.63 Lakh - 5.94 Lakh']
```

In [19]:

```
# Scraping fueltypes of all cars
fueltype = []
def ftype(sp):
    fueltype.append(sp.find("span", itemprop = "fuelType").get_text().strip())
for i in soups:
    ftype(i)
```

```
print(fueltype)
```

```
['Petrol & CNG', 'Petrol & CNG', 'Petrol Only', 'Petrol & CNG', 'Petrol & Diesel', 'Petro
l & Diesel', 'Petrol & CNG', 'Petrol Only', 'Petrol & CNG', 'Petrol & CNG']
```

In [20]:

```
# Scraping engine capacity of all cars
engines = []
def engine(engn):
    engines.append(engn.find("span", itemprop = "name").get_text().strip())
for i in soups:
    engine(i)
print(engines)
```

```
['998 cc & 1197 cc', 'CNG - 1198 cc Petrol - 1197 cc', '1197 cc', '796 cc', 'Diesel - 149
7 cc Petrol - 1199 cc', 'Petrol - 1497 cc & 1353 cc Diesel - 1493 cc', '1462 cc', '1462 c
c', '1197 cc', '1196 cc']
```

In [21]:

```
# Scraping transmission types of all cars
transmissions = []
def transmission(transm):
    transmissions.append(transm.find("span", itemprop = "vehicleTransmission").get_text(
).strip())
for i in soups:
    transmission(i)
print(transmissions)
```

```
['Manual & Automatic', 'Manual & Automatic', 'Manual & Automatic', 'Manual Only', 'Manual
& Automatic', 'Manual & Automatic', 'Manual & Automatic', 'Manual & Automatic', 'Manual &
Automatic', 'Manual Only']
```

In [22]:

```
# Scraping mileage of all cars
mileages = []
def mileage(mil):
    mileages.append(mil.find("span", itemprop = "fuelEfficiency").get_text().strip().spl
it("\t")[0])
for i in soups:
    mileage(i)
print(mileages)
```

```
['25.19 Kmpl - 34.05 Km/Kg', '23.76 Kmpl - 30.9 Km/Kg', '22.94 Kmpl', '22.05 Kmpl - 31.59
Km/Kg', '17.57 - 22.07 Kmpl', '16.8 Kmpl', '20.51 Kmpl - 26.11 Km/Kg', '20.15 Kmpl', '24.
12 Kmpl - 31.12 Km/Kg', '16.11 Kmpl - 20.88 Km/Kg']
```

In [23]:

```
# Creating a dataframe out of scrapped data
df = pd.DataFrame(data={'Name of Cars':car_names, 'Units Sold':int_unit, 'Mileage':milea
ges, 'Engine Capacity':engines, 'Transmission Type':transmissions, 'Fuel Type':fueltype,
'Price Range':prices})
df
```

Out[23]:

	Name of Cars	Units Sold	Mileage	Engine Capacity	Transmission Type	Fuel Type	Price Range
0	Maruti Suzuki Wagon R	188838	25.19 Kmpl - 34.05 Km/Kg	998 cc & 1197 cc	Manual & Automatic	Petrol & CNG	5.47 Lakh - 7.20 Lakh
1	Maruti Suzuki Swift	167827	23.76 Kmpl - 30.9 Km/Kg	CNG - 1198 cc Petrol - 1197 cc	Manual & Automatic	Petrol & CNG	5.91 Lakh - 8.85 Lakh
2	Maruti Suzuki Baleno	148187	22.94 Kmpl	1197 cc	Manual & Automatic	Petrol Only	6.49 Lakh - 9.71 Lakh
3	Maruti Suzuki Alto	145167	22.05 Kmpl - 31.59 Km/Kg	796 cc	Manual Only	Petrol & CNG	3.39 Lakh - 5.03 Lakh

	Name of Cars	Units Sold	Mileage	Engine Capacity	Transmission Type	Fuel Type	Price Range
4	Tata Nexon	124150	17.57 - 22.07 Kmpl	Diesel - 1497 cc Petrol - 1199 cc	Manual & Automatic	Petrol & Diesel	7.59 Lakh - 12.77 Lakh
5	Hyundai Creta	118092	16.8 Kmpl	Petrol - 1497 cc & 1353 cc Diesel - 1493 cc	Manual & Automatic	Petrol & Diesel	0.44 Lakh - 18.15 Lakh
6	Maruti Suzuki Ertiga	117150	20.51 Kmpl - 26.11 Km/Kg	1462 cc	Manual & Automatic	Petrol & CNG	8.41 Lakh - 12.79 Lakh
7	Maruti Suzuki Vitara Brezza	113751	20.15 Kmpl	1462 cc	Manual & Automatic	Petrol Only	7.99 Lakh - 13.96 Lakh
8	Maruti Suzuki Dzire	108564	24.12 Kmpl - 31.12 Km/Kg	1197 cc	Manual & Automatic	Petrol & CNG	6.24 Lakh - 9.17 Lakh
9	Maruti Eeco	108345	16.11 Kmpl - 20.88 Km/Kg	1196 cc	Manual Only	Petrol & CNG	4.63 Lakh - 5.94 Lakh