In [1]:
```
# Web scrapping :- It is used for fetching the data from various web page and convert into .csv file to do
# further analyzation.
# Parcing :- Data comes in format and we convert into required format. e.g. When we do scrapping all the
# sites data comes in string format and we convert into .csv format
```

In [2]:
```
import requests
```

In [3]:
```
# It sends http requests to web page for scrapping if it's response comes 200 or above then it gives the data
# and we can scrap the page otherwise we are not allowed .
```

In [4]:
```
from bs4 import BeautifulSoup
```

In [5]:
```
# bs4 :- It is a package.
# BeautifulSoup :- It is a python liabrary. It is use for extracts content from url
```

In [6]:
```
import pandas as pd
```

In [7]:
```
# Pandas :- It is a python liabrary. It is 2D liabrary means it gives data in rows and columns.
```

In [8]:
```
url = 'https://www.worldometers.info/world-population/population-by-country/'
```

In [9]:
```
page = requests.get(url)
page
```

Out[9]:
```
<Response [200]>
```

In [10]:
```
# requests.get():- It is afunction. requests send the request and .get will fetch the html content
# that we have save in variable name 'page'.
# In the output we get the rsponse 403. It means we can fetch the data from this particular site.
```

In [11]:
```
soup = BeautifulSoup(page.text)
soup
```

Out[11]:
```
<!DOCTYPE html>
<!--[if IE 8]> <html lang="en" class="ie8"> <![endif]--><!--[if IE 9]> <html lang="en" class="ie9"> <![endif]--><!--[if !IE]>
<!--><html lang="en"> <!--<![endif]--> <head> <meta charset="utf-8"/> <meta content="IE=edge" http-equiv="X-UA-Compatible"/>
<meta content="width=device-width, initial-scale=1" name="viewport"/> <title>Population by Country (2022) - Worldometer</titl
e><meta content="List of countries and dependencies in the world ranked by population, from the most populated. Growth rate,
median age, fertility rate, area, density, population density, urbanization, urban population, share of world population." na
me="description"/><!-- Favicon --><link href="/favicon/favicon.ico" rel="shortcut icon" type="image/x-icon"/><link href="/fav
icon/apple-icon-57x57.png" rel="apple-touch-icon" sizes="57x57"/><link href="/favicon/apple-icon-60x60.png" rel="apple-touch-
icon" sizes="60x60"/><link href="/favicon/apple-icon-72x72.png" rel="apple-touch-icon" sizes="72x72"/><link href="/favicon/ap
ple-icon-76x76.png" rel="apple-touch-icon" sizes="76x76"/><link href="/favicon/apple-icon-114x114.png" rel="apple-touch-icon"
sizes="114x114"/><link href="/favicon/apple-icon-120x120.png" rel="apple-touch-icon" sizes="120x120"/><link href="/favicon/ap
ple-icon-144x144.png" rel="apple-touch-icon" sizes="144x144"/><link href="/favicon/apple-icon-152x152.png" rel="apple-touch-i
con" sizes="152x152"/><link href="/favicon/apple-icon-180x180.png" rel="apple-touch-icon" sizes="180x180"/><link href="/favic
on/android-icon-192x192.png" rel="icon" sizes="192x192" type="image/png"/><link href="/favicon/favicon-32x32.png" rel="icon"
sizes="32x32" type="image/png"/><link href="/favicon/favicon-96x96.png" rel="icon" sizes="96x96" type="image/png"/><link href
="/favicon/favicon-16x16.png" rel="icon" sizes="16x16" type="image/png"/><link href="/favicon/manifest.json" rel="manifest"/>
<meta content="#ffffff" name="msapplication-TileColor"/><meta content="/favicon/ms-icon-144x144.png" name="msapplication-Tile
Image"/><meta content="#ffffff" name="theme-color"/><!-- og image --><meta content="http://www.worldometers.info/img/worldome
ters-fb.jpg" property="og:image"/> <!-- Bootstrap --> <link href="/css/bootstrap.min.css" rel="stylesheet"/><link href="/wm1
```

In [12]:
```
# Here we are using BeautifulSoup to convert all html content into text format storing in variable name 'soup'.
# If our data is not in readable format then we have to use soup.prettify() function. Then it gibes into readable or
# more arranged format. Here our data is already in arranged format we need not have to use prettify() function.
```

In [13]:
```
table = soup.find('table', id= "example2")
table = table.find('tbody')
table
```

Out[13]:
```
<tbody> <tr> <td>1</td> <td style="font-weight: bold; font-size:15px; text-align:left"><a href="/world-population/china-popul
ation/">China</a></td> <td style="font-weight: bold;">1,439,323,776</td> <td>0.39 %</td> <td>5,540,090</td> <td>153</td> <td>
9,388,211</td> <td>-348,399</td> <td>1.7</td> <td>38</td> <td>61 %</td> <td>18.47 %</td> </tr> <tr> <td>2</td> <td style="fon
t-weight: bold; font-size:15px; text-align:left"><a href="/world-population/india-population/">India</a></td> <td style="font
-weight: bold;">1,380,004,385</td> <td>0.99 %</td> <td>13,586,631</td> <td>464</td> <td>2,973,190</td> <td>-532,687</td> <td>
2.2</td> <td>28</td> <td>35 %</td> <td>17.70 %</td> </tr> <tr> <td>3</td> <td style="font-weight: bold; font-size:15px; text-
align:left"><a href="/world-population/us-population/">United States</a></td> <td style="font-weight: bold;">331,002,651</td>
<td>0.59 %</td> <td>1,937,734</td> <td>36</td> <td>9,147,420</td> <td>954,806</td> <td>1.8</td> <td>38</td> <td>83 %</td> <td
>4.25 %</td> </tr> <tr> <td>4</td> <td style="font-weight: bold; font-size:15px; text-align:left"><a href="/world-population/
indonesia-population/">Indonesia</a></td> <td style="font-weight: bold;">273,523,615</td> <td>1.07 %</td> <td>2,898,047</td>
<td>151</td> <td>1,811,570</td> <td>-98,955</td> <td>2.3</td> <td>30</td> <td>56 %</td> <td>3.51 %</td> </tr> <tr> <td>5</td>
<td style="font-weight: bold; font-size:15px; text-align:left"><a href="/world-population/pakistan-population/">Pakistan</a>
</td> <td style="font-weight: bold;">220,892,340</td> <td>2.00 %</td> <td>4,327,022</td> <td>287</td> <td>770,880</td> <td>-2
33,379</td> <td>3.6</td> <td>23</td> <td>35 %</td> <td>2.83 %</td> </tr> <tr> <td>6</td> <td style="font-weight: bold; font-s
ize:15px; text-align:left"><a href="/world-population/brazil-population/">Brazil</a></td> <td style="font-weight: bold;">212,
559,417</td> <td>0.72 %</td> <td>1,509,890</td> <td>25</td> <td>8,358,140</td> <td>21,200</td> <td>1.7</td> <td>33</td> <td>8
8 %</td> <td>2.73 %</td> </tr> <tr> <td>7</td> <td style="font-weight: bold; font-size:15px; text-align:left"><a href="/world
-population/nigeria-population/">Nigeria</a></td> <td style="font-weight: bold;">206,139,589</td> <td>2.58 %</td> <td>5,175,9
90</td> <td>226</td> <td>910,770</td> <td>-60,000</td> <td>5.4</td> <td>18</td> <td>52 %</td> <td>2.64 %</td> </tr> <tr> <td>
```

In [14]:
```
# In this we need to find the table tag which we required. In total page there is only one table So we use .find() method.
#if there are many table tag then we use .find_all method.
# We further narrow it down to tbody by using .find() method by using on table tags.
```

In [15]:
```
rows = table.find_all('tr')
rows
```

Out[15]:
```
[<tr> <td>1</td> <td style="font-weight: bold; font-size:15px; text-align:left"><a href="/world-population/china-populatio
n/">China</a></td> <td style="font-weight: bold;">1,439,323,776</td> <td>0.39 %</td> <td>5,540,090</td> <td>153</td> <td>9,38
8,211</td> <td>-348,399</td> <td>1.7</td> <td>38</td> <td>61 %</td> <td>18.47 %</td> </tr>,
 <tr> <td>2</td> <td style="font-weight: bold; font-size:15px; text-align:left"><a href="/world-population/india-populatio
n/">India</a></td> <td style="font-weight: bold;">1,380,004,385</td> <td>0.99 %</td> <td>13,586,631</td> <td>464</td> <td>2,9
73,190</td> <td>-532,687</td> <td>2.2</td> <td>28</td> <td>35 %</td> <td>17.70 %</td> </tr>,
 <tr> <td>3</td> <td style="font-weight: bold; font-size:15px; text-align:left"><a href="/world-population/us-population/">Un
ited States</a></td> <td style="font-weight: bold;">331,002,651</td> <td>0.59 %</td> <td>1,937,734</td> <td>36</td> <td>9,14
7,420</td> <td>954,806</td> <td>1.8</td> <td>38</td> <td>83 %</td> <td>4.25 %</td> </tr>,
 <tr> <td>4</td> <td style="font-weight: bold; font-size:15px; text-align:left"><a href="/world-population/indonesia-populati
on/">Indonesia</a></td> <td style="font-weight: bold;">273,523,615</td> <td>1.07 %</td> <td>2,898,047</td> <td>151</td> <td>
1,811,570</td> <td>-98,955</td> <td>2.3</td> <td>30</td> <td>56 %</td> <td>3.51 %</td> </tr>,
 <tr> <td>5</td> <td style="font-weight: bold; font-size:15px; text-align:left"><a href="/world-population/pakistan-populatio
n/">Pakistan</a></td> <td style="font-weight: bold;">220,892,340</td> <td>2.00 %</td> <td>4,327,022</td> <td>287</td> <td>77
0,880</td> <td>-233,379</td> <td>3.6</td> <td>23</td> <td>35 %</td> <td>2.83 %</td> </tr>,
 <tr> <td>6</td> <td style="font-weight: bold; font-size:15px; text-align:left"><a href="/world-population/brazil-populatio
n/">Brazil</a></td> <td style="font-weight: bold;">212,559,417</td> <td>0.72 %</td> <td>1,509,890</td> <td>25</td> <td>8,358,
140</td> <td>21,200</td> <td>1.7</td> <td>33</td> <td>88 %</td> <td>2.73 %</td> </tr>,
 <tr> <td>7</td> <td style="font-weight: bold; font-size:15px; text-align:left"><a href="/world-population/nigeria-populatio
```

In [16]:
```
pop_info = []
for i in rows:
    info = i.find_all("td")
    text = [j.text for j in info]
    pop_info.append(text)
```

In [17]:
```
pop_info
```

Out[17]:
```
[['1',
  'China',
  '1,439,323,776',
  '0.39 %',
  '5,540,090',
  '153',
  '9,388,211',
  '-348,399',
  '1.7',
  '38',
  '61 %',
  '18.47 %'],
 ['2',
  'India',
  '1,380,004,385',
  '0.99 %',
  '13,586,631',
  '464',
  '2,973,190',
```

In [18]:
```
# In previous step we get all html data. Now we want remove the html tags from the data and separate it.
# For this we use for loop and first we need to seperate td tags where all the tables data is saved. So we seperate it
# by using .find_all() method after this we remove the tags by using .text method and
# the each and every data append in pop_info list as we can see in output.
```

In [19]:
```
index = []
country = []
population_2020 = []
Yearly_Change = []
Net_change = []
Density_perkm2 = []
Land_area_km2 = []
Migrant_net = []

for i in pop_info:
    index.append(i[0])
    country.append(i[1])
    population_2020.append(i[2])
    Yearly_Change.append(i[3])
    Net_change.append(i[4])
    Density_perkm2.append(i[5])
    Land_area_km2.append(i[6])
    Migrant_net.append(i[7])
```

In [20]:
```
# As we see before all data is in one list. Now we seperate data each column wise so we again use for loop and append in
# each list column wise.
```

In [21]:
```
dict1 = {'index':index, 'country':country, 'population(2020)':population_2020, 'Yearly Change' : Yearly_Change, 'Net change': Net
```

In [22]:
```
# Now we want to assign every columns data to its column name.
# So we create dictionary and assign columns data to its column's name.
```

In [23]:
```
population_data1 = pd.DataFrame(dict1)
population_data1
```

Out[23]:

| | index | country | population(2020) | Yearly Change | Net change | Density p/km2 | Land_area_km2 | Migrant_net |
|---|---|---|---|---|---|---|---|---|
| **0** | 1 | China | 1,439,323,776 | 0.39 % | 5,540,090 | 153 | 9,388,211 | -348,399 |
| **1** | 2 | India | 1,380,004,385 | 0.99 % | 13,586,631 | 464 | 2,973,190 | -532,687 |
| **2** | 3 | United States | 331,002,651 | 0.59 % | 1,937,734 | 36 | 9,147,420 | 954,806 |
| **3** | 4 | Indonesia | 273,523,615 | 1.07 % | 2,898,047 | 151 | 1,811,570 | -98,955 |
| **4** | 5 | Pakistan | 220,892,340 | 2.00 % | 4,327,022 | 287 | 770,880 | -233,379 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **230** | 231 | Montserrat | 4,992 | 0.06 % | 3 | 50 | 100 | |
| **231** | 232 | Falkland Islands | 3,480 | 3.05 % | 103 | 0 | 12,170 | |
| **232** | 233 | Niue | 1,626 | 0.68 % | 11 | 6 | 260 | |
| **233** | 234 | Tokelau | 1,357 | 1.27 % | 17 | 136 | 10 | |
| **234** | 235 | Holy See | 801 | 0.25 % | 2 | 2,003 | 0 | |

235 rows × 8 columns

In [ ]:
```
# By using pandas we create table.
```

In [25]:
```
population_data1.to_csv('population_data1.csv', index=False)
```

In [ ]:
```
# .to_csv we convert data frame into the excel file and save as population_data1
# and by writing index=False delete the if there is any extra indexing.
# Now we can use the data in excel file do the analysys and further process.
```