Project – 1 Documentation

**Scraping and Analysis of Statistics of Sports Players from Real-Time Webpage**

**OVERVIEW: A** project that gives you a better understanding of scraping data from websites and how to analyse them. Usage of various libraries as NumPy, Mat Plot, Pandas.

In the course of completing the project, you use the web scraping function, converting the extracted data into a pandas data Frame, and Storing the analysed data.

**Problem Statement**

Web scrape basketball statistics from Wikipedia of some of the greatest basketball players and export it as a CSV file format.

**Software Requirements**

1. Programming Language: Python

2. Environment: Jupyter Notebooks / Google Collab

3. Database: CSV (export type)

4. Operation System: Windows XP or above

5. Libraries Used: Beautiful Soup, requests, Pandas, NumPy, boto3 ,Matplotlib, display

1. **Open a New Notebook and import the required libraires**

|  |  |
| --- | --- |
|  | import bs4  import requests  import pandas as pd  import numpy as np  !pip install boto3  import boto3  import matplotlib.pyplot as plt  from IPython.display import display |

Description: The required libraries are imported. Libraries can also be imported using the keyword ‘as’ which allows us to rename the name of the library with a shorter name.

1. **Reading the webpage**

def get\_basketball\_stats(link):

response = requests.get(link)

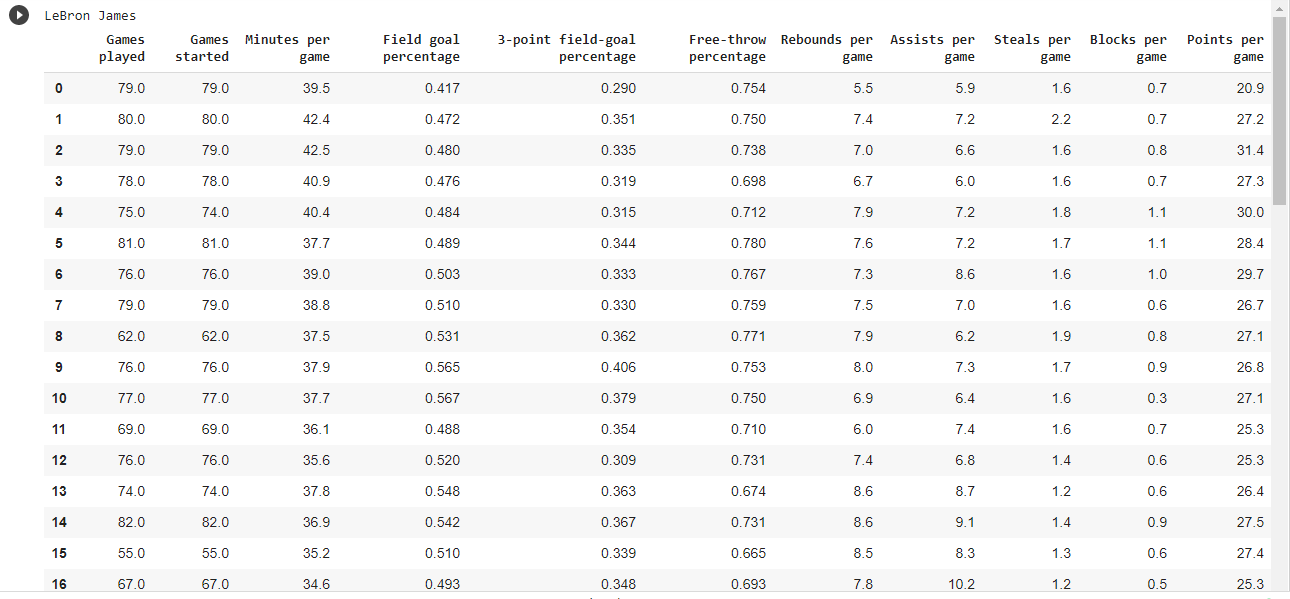
soup = bs4.BeautifulSoup(response.text, 'html.parser')

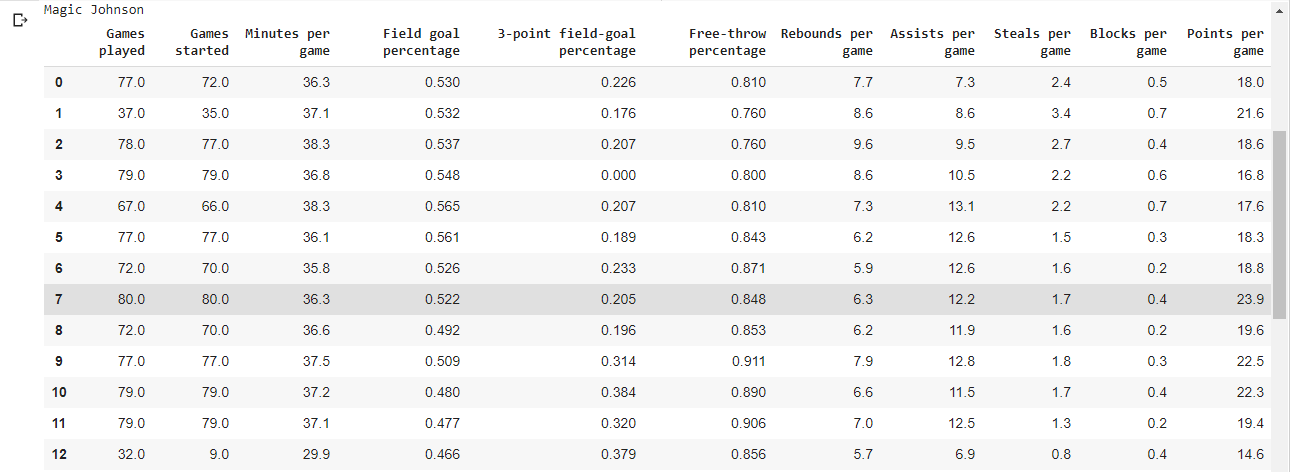
Description: The above function is used to read webpage using get() method present in requests library and the result is stored in variable named ‘response’. The data fetched is converted into text format using BeautifulSoup() method present in bs4 library and is stored in a variable named soup.

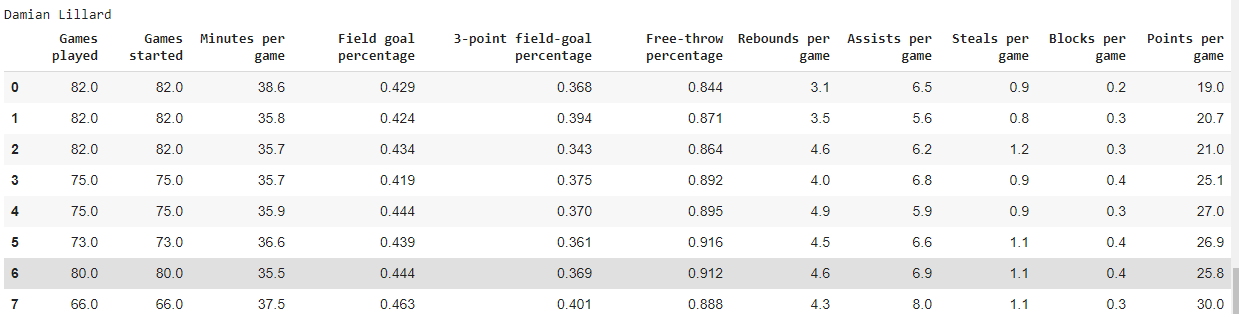
1. **Main Function Process**

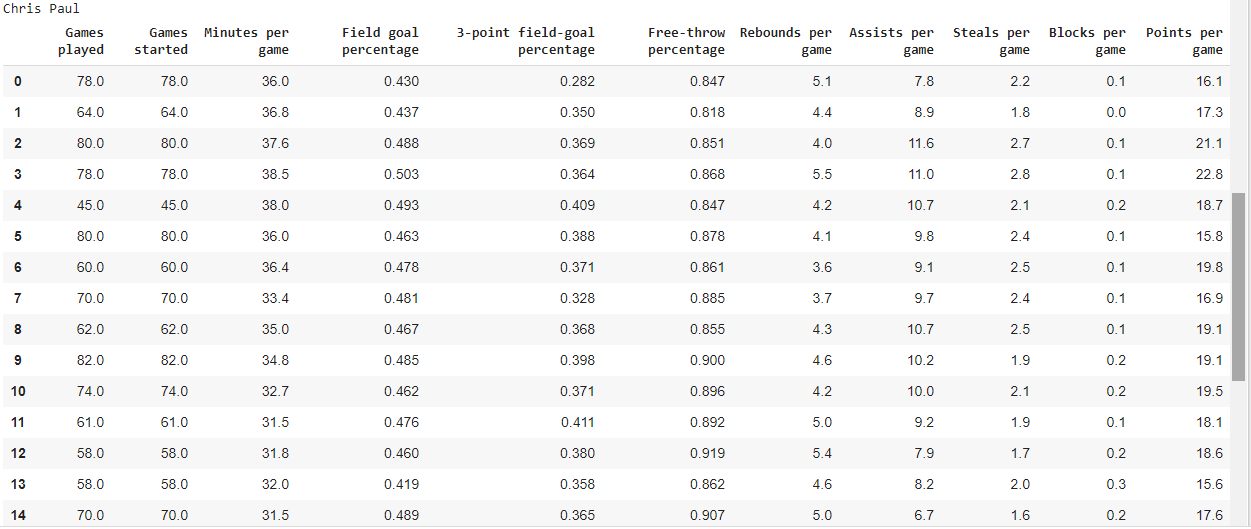
|  |  |
| --- | --- |
| table = soup.find(class\_='wikitable sortable')  headers = table.tr  titles = headers.find\_all('abbr')  data = {title['title']: [] for title in titles}  for row in table.find\_all('tr')[1:]:    for key, a in zip(data.keys(),row.find\_all('td')[2:]):  data[key].append(''.join(c for c in a.text if (c.isdigit() or c == '.')))  Min = min([len(x) for x in data.values()])  for key in data.keys():  data[key] = list(map(lambda x: float(x), data[key][:Min]))  return data  Description: The tabular data present in the webpage is associated with a CSS class named ‘wikitable sortable’. The rows of the table are accessed using tr tag and are stored in variable named headers.  Titles of the table can be differentiated using abbr tag are stored in titles variable which are obtained by using find\_all() method. A dictionary named data is created whose keys are titles of each column and values are empty lists initially.  As the first row contains the titles it is excluded and for loop is initialized to second row. We iterate over each table row by finding each table tag tr and assign it to the object.  In the second for loop we iterate over each cell in the table, as each cell corresponds to a different column we obtain the corresponding key corresponding to column using zip() method the column containing years and team names is removed by slicing.  We take the data which is valid ignoring all the special symbols or characters except the ‘.’ Symbol which is used for decimal representation and the extra rows are removed by finding the list with minimum elements. The dictionary data is updated with required rows by using a lambda function (an anonymous function in python) and the dictionary with name data is returned.   1. **Declaring links and names of the personals to scrap the data**   links=['https://en.wikipedia.org/wiki/LeBron\_James'\  ,'https://en.wikipedia.org/wiki/Magic\_Johnson'\  ,'https://en.wikipedia.org/wiki/Chris\_Paul'\  ,'https://en.wikipedia.org/wiki/Charles\_Barkley'\  ,'https://en.wikipedia.org/wiki/David\_Robinson'\  ,'https://en.wikipedia.org/wiki/Stephen\_Curry'\  ,'https://en.wikipedia.org/wiki/John\_Stockton'\  ,'https://en.wikipedia.org/wiki/Russell\_Westbrook'\  ,'https://en.wikipedia.org/wiki/Damian\_Lillard'\  ,'https://en.wikipedia.org/wiki/Ray\_Allen']  names=['LeBron James','Magic Johnson','Chris Paul','Charles Barkley','David Robinson','Stephen Curry','John Stockton','Russell Westbrook','Damian Lillard','Ray Allen']  leborn\_james\_dict = get\_basketball\_stats(links[0])  maigc\_johnson\_dict = get\_basketball\_stats(links[1])  chris\_paul\_dict = get\_basketball\_stats(links[2])  charles\_barkley\_dict = get\_basketball\_stats(links[3])  david\_robinson\_dict = get\_basketball\_stats(links[4])  stephen\_curry\_dict = get\_basketball\_stats(links[5])  jhon\_stockton\_dict = get\_basketball\_stats(links[6])  russell\_westbrook\_dict = get\_basketball\_stats(links[7])  damian\_lillard\_dict = get\_basketball\_stats(links[8])  ray\_allen\_dict = get\_basketball\_stats(links[9])  lj\_table = pd.DataFrame(leborn\_james\_dict)  mj\_table = pd.DataFrame(maigc\_johnson\_dict)  cp\_table = pd.DataFrame(chris\_paul\_dict)  cb\_table = pd.DataFrame(charles\_barkley\_dict)  dr\_table = pd.DataFrame(david\_robinson\_dict)  sc\_table = pd.DataFrame(stephen\_curry\_dict)  js\_table = pd.DataFrame(jhon\_stockton\_dict)  rw\_table = pd.DataFrame(russell\_westbrook\_dict)  dl\_table = pd.DataFrame(damian\_lillard\_dict)  ra\_table = pd.DataFrame(ray\_allen\_dict)  list\_table =[lj\_table, mj\_table,cp\_table,cb\_table,dr\_table,sc\_table,js\_table,rw\_table,dl\_table,ra\_table]  i = 0  for name in names:  print(name)  display(list\_table[i])  i += 1 |  |
|  |  |

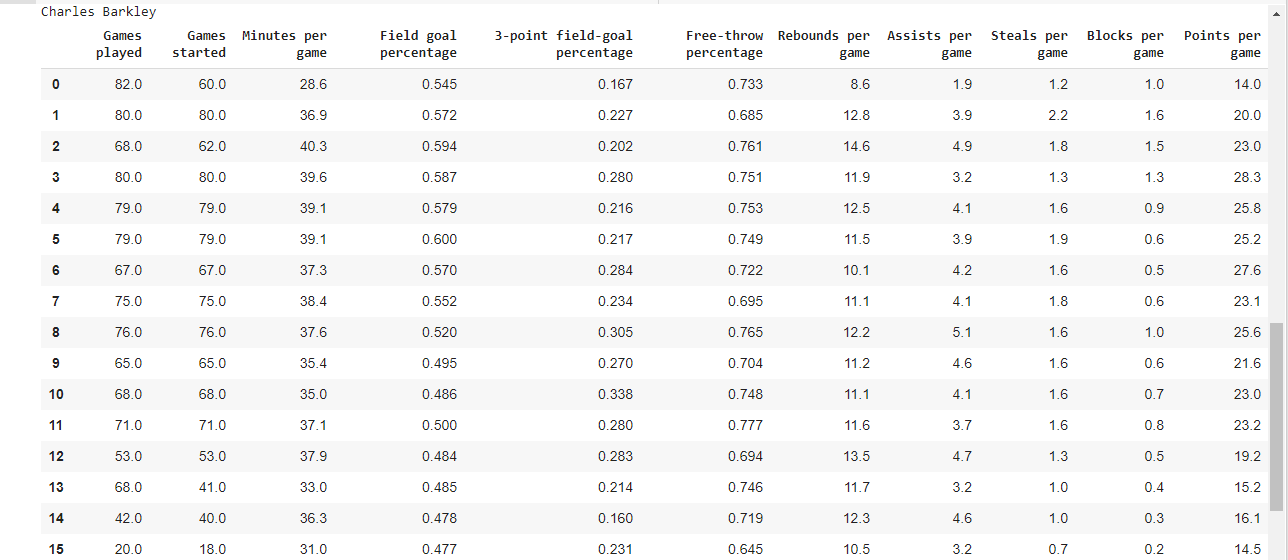
**Output:**

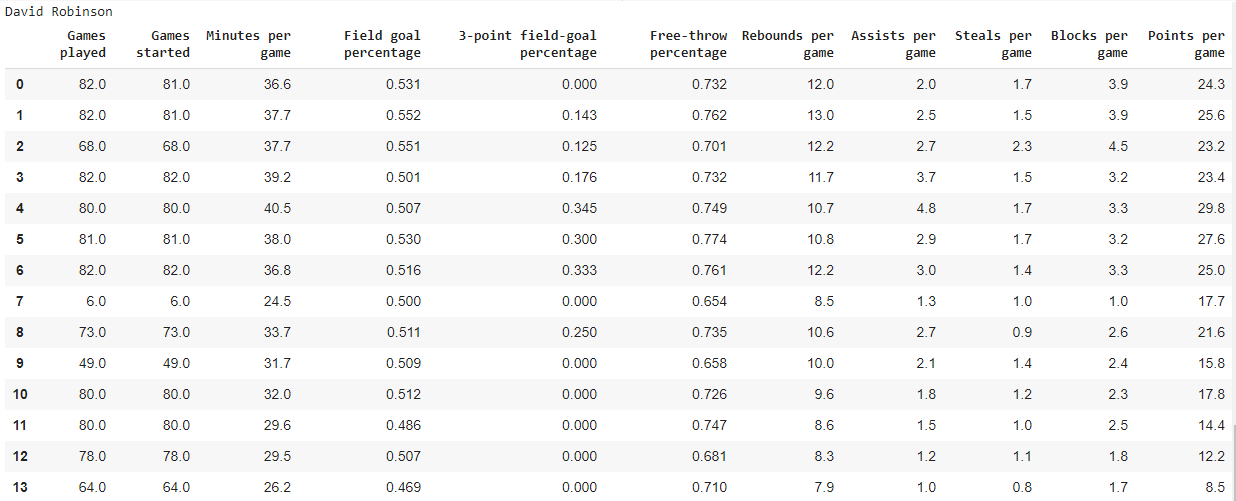
****

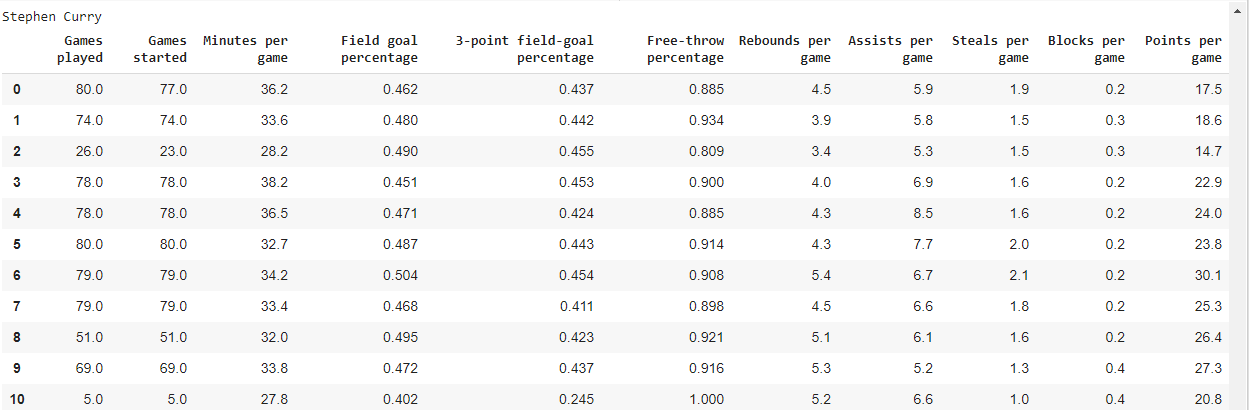
****

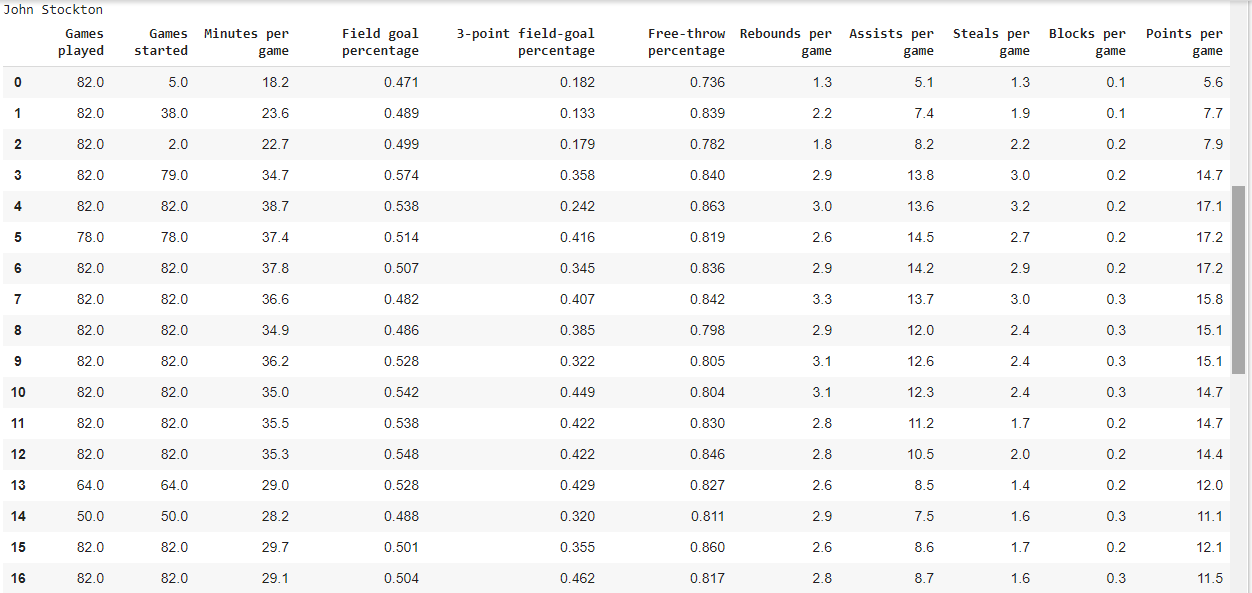
****

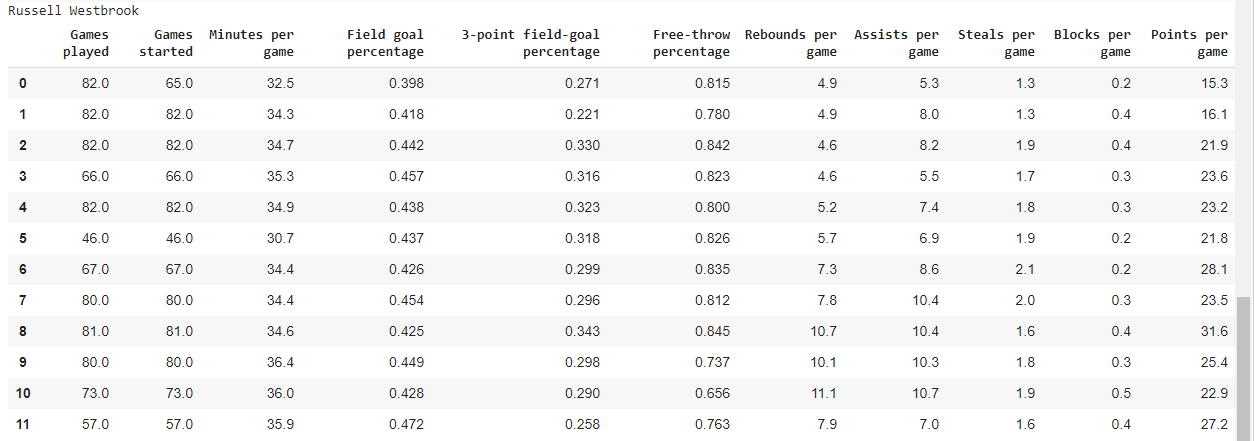
****

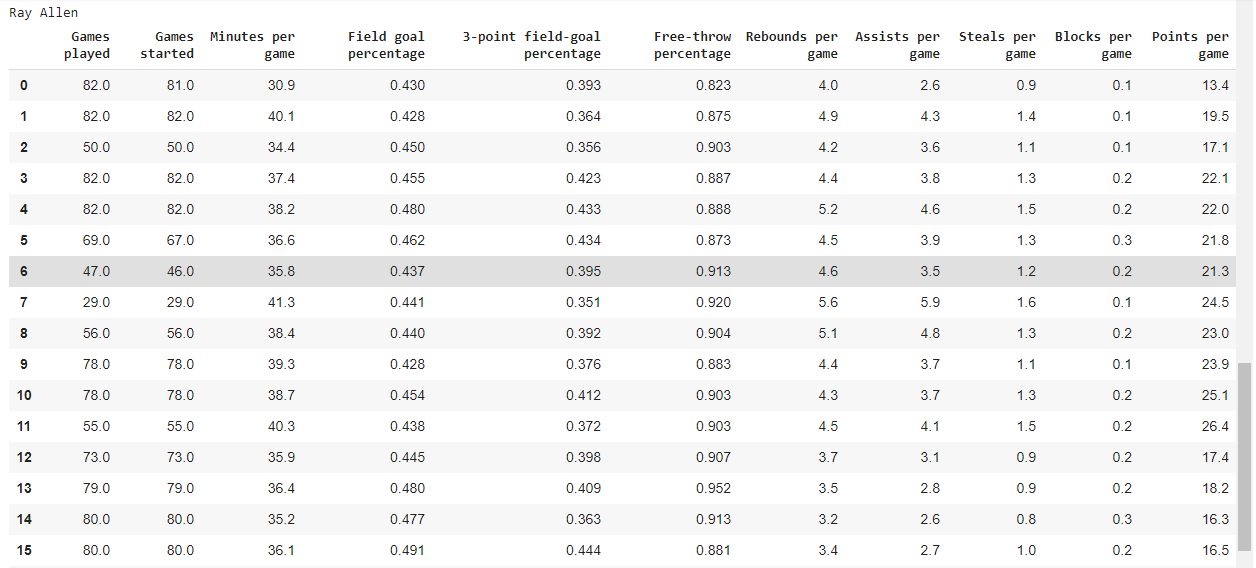
****

****

****

****

****

****

Description: Links of the webpages of each player is stored in a list named links and names of the players in other list.

For each player in the list the table is displayed.

**6. Making a plot using matplotlib**

j = 0

for name in names:

    plt.plot(list\_table[j][['Points per game']],label=name)

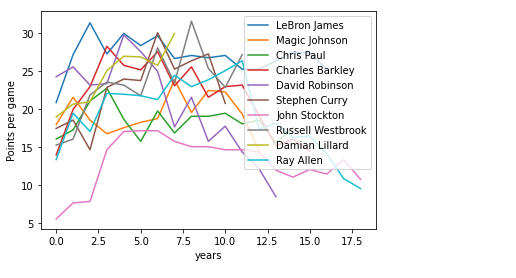
    plt.legend()

    plt.xlabel('years')

    plt.ylabel('Points per game')

    j += 1

**Output:**

****

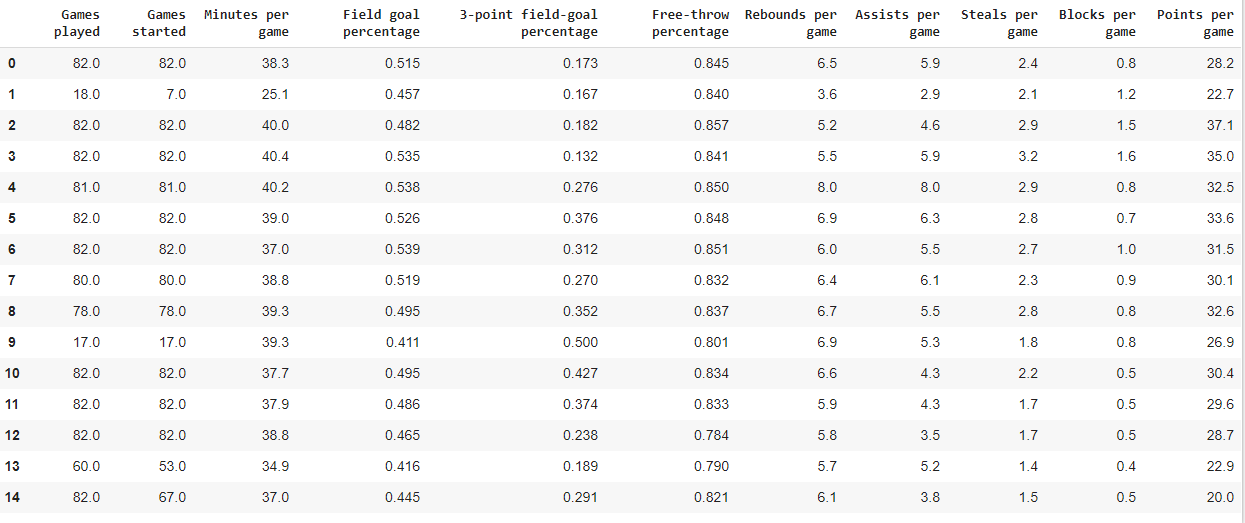
Description: The performance of each player is plotted by taking points in each game on y- axis and number of years on x – axis, plt.legend() method is used to mention an area describing the elements of the graph. plt.xlabel() method is used to label x-axis and similarly plt.ylabel() for labeling y-axis.

**7. Storing the Player Statistics in Object Storage**

csv\_name = 'MJ1.csv'

mj\_table.to\_csv(csv\_name)

mj\_table



Description: The tabular data of one of the player is converted into csv format and stored in a csv file name MJ1 with a .csv extension

|  |
| --- |
|  |
|  |

**Conclusion**

This project aims at collecting and scraping the data from various websites available on internet, converting it into tabular form thus making it convenient for analyzing the data in an appropriate way.

Plotting of graph using matplotlib which gives the performance of all the players on a single graph

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  |  |