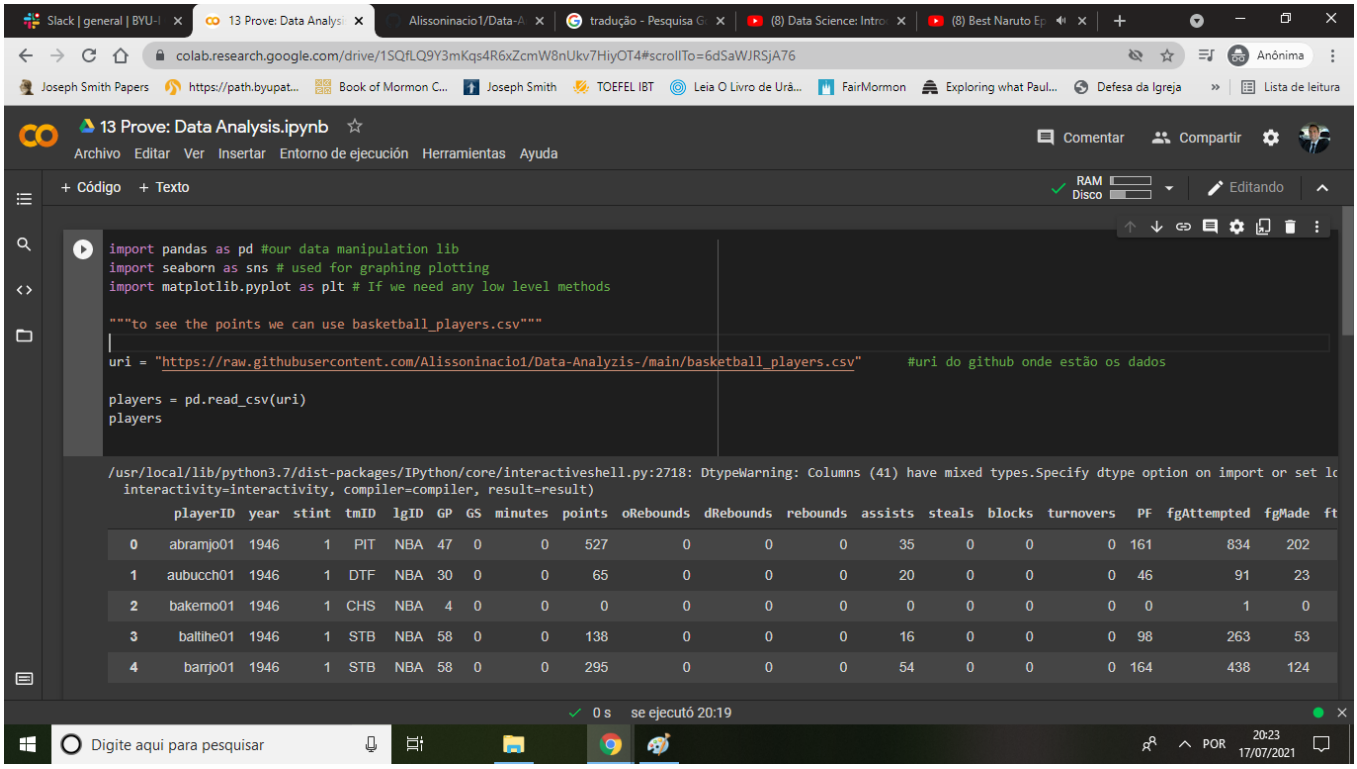
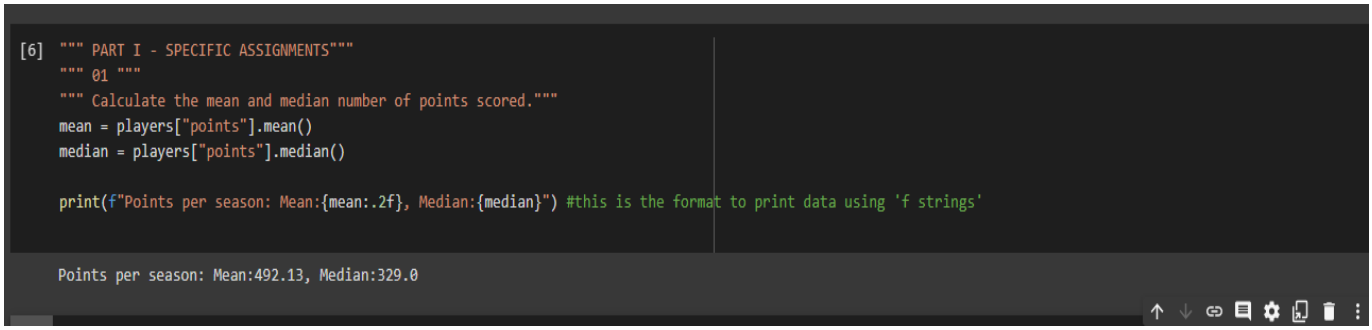
13 Prove: Data Analysis

Name: Alisson de Oliveira Inacio

PART I - SPECIFIC ASSIGNMENTS

1. Calculate the mean and median number of points scored. (In other words, each row is the amount of points a player scored during a particular season. Calculate the median of these values. The result of this is that we have the median number of points players score each season.)





The mean of points per season is 492.13 and the median is 329.0.

2. Determine the highest number of points recorded in a single season. Identify who scored those points and the year they did so.

13 Prove: Data Analysis.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda Se guardaron todos los cambios

Comentar Compartir

+ Código + Texto

RAM Disco Editando

```
[7] """ PART I - SPECIFIC ASSIGNMENTS"""
    """ 02 """
    """
    Determine the highest number of points recorded in a single season.
    Identify who scored those points and the year they did so.

    ******
    ABOUT THE LAST FILE =  we can see the points and the season, howver we can't see the names in order to identify who scored those points
    let's use and analyze another csv file called basketball_player_allstar.csv.
    Inside this file we can identify the point per season and who scored.
    ******
    """

    #I called a second uri and read it, after this, I used describe()
    uri_2 = "https://raw.githubusercontent.com/Alissoninacio1/Data-Analyzis-/main/basketball_player_allstar.csv"
    who_scored = pd.read_csv(uri_2)

    who_scored
```

```
player_id  last_name  first_name  season_id  conference  league_id  games_played  minutes  points  o_rebounds  d_rebounds  rebounds  assists  steals  blocks
```

+ Código + Texto

RAM Disco Editando

```
[15] #now, we could see our set of data, let's search for the max() of point in a column points
     maximo = who_scored["points"].max()
     print(maximo)
```

```
42.0
```

```
#This print is showing certain columns and making it easy to read
print(who_scored[["last_name", "first_name",  "season_id", "points"]].sort_values("points", ascending=False).head(10))
```

```
       last_name first_name  season_id  points
245  Chamberlain       Wilt       1961    42.0
794       Jordan    Michael       1987    40.0
87         Barry       Rick       1966    38.0
488      Garnett      Kevin       2002    37.0
977      McGrady      Tracy       2005    36.0
683      Iverson      Allen       2002    35.0
513       Gervin     George       1979    34.0
237     Chambers        Tom       1986    34.0
421       Erving     Julius       1983    34.0
100       Baylor      Elgin       1961    32.0
```

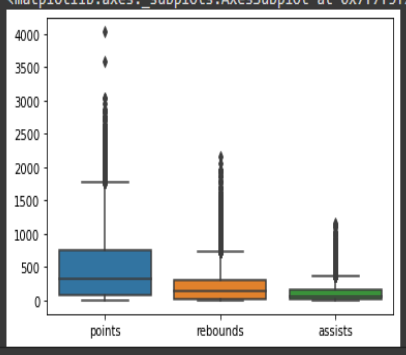
```
[27] #when see the dataFrame we confirm the  max points is 42 and we can see the name and the season
     print("Chamberlain Wilt, 1961, 42.0 points")
```

```
Chamberlain Wilt, 1961, 42.0 points
```

The highest points recorded in a season is 42.0 points.

3. Produce a boxplot that shows the distribution of total points, total assists, and total rebounds (each of these three is a separate box plot, but they can be on the same scale and in the same graphic).

```
[10] PART I - SPECIFIC ASSIGNMENTS
     03
     Produce a boxplot that shows the distribution of total points, total assists,
     and total rebounds (each of these three is a separate box plot, but they can be on the same scale and in the same graphic).
     """
     uri_3 = "https://raw.githubusercontent.com/Alissoninacio1/Data-Analyzis-/main/basketball_master.csv"
     master = pd.read_csv(uri_3)
     nba = pd.merge(players, master, how="left", left_on="playerID", right_on="bioID")

     sns.boxplot(data=nba[["points", "rebounds", "assists"]]) #boxplot of multiple columns using seaborn

<matplotlib.axes._subplots.AxesSubplot at 0x7f7f5f226bd0>
```



✓ 0 s se ejecutó 11:54

4. Produce a plot that shows how the number of points scored has changed over time by showing the median of points scored per year, over time. The x-axis is the year and the y-axis is the median number of points among all players for that year.

13 Prove: Data Analysis.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda Se guardaron todos los cambios

Comentar Compartir

+ Código + Texto

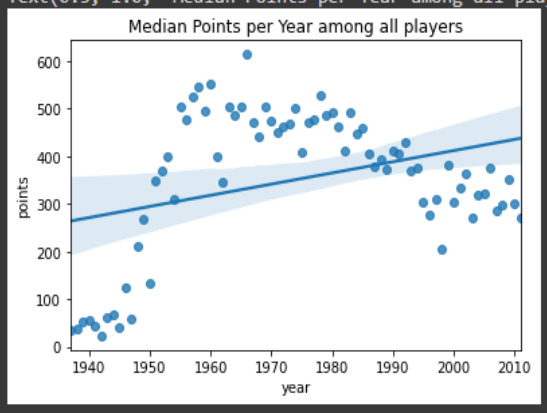
RAM Disco Editando

```
[30] """
     PART I - SPECIFIC ASSIGNMENTS
     04
     Produce a plot that shows how the number of points scored has changed over time by showing the median of points scored per year, over time.
     The x-axis is the year and the y-axis is the median number of points among all players for that year.
     """


     """
     To plot rebounds per year, we first need to group the statistics by year.
     When we do so, we need to specify how we want to aggregate the data of that year.
     In this case, we'll use the median of the reboundsPerGame
     """
     nba_year_set = nba[["points", "year"]].groupby("year").median()
     print(nba_year_set)

          points
     year
     1937    36.0
     1938    39.0
```

Below, we can see the plot using x-axis is the year and the y-axis is the median number of points among all players for that year.

```python
#let's remove any years where the median was 0
nba_year_set = nba_year_set[nba_year_set["points"] > 0]
sns.regplot(data=nba_year_set, x="year", y="points").set_title("Median Points per Year among all players")
```

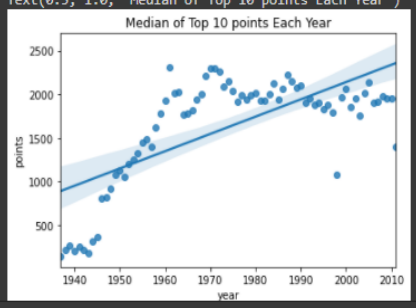
```
Text(0.5, 1.0, 'Median Points per Year among all players')
```




```python
# Another way to consider this is to find the top 10 rebounders of the year and look at their median.
```

```python
# Another way to consider this is to find the top 10 rebounders of the year and look at their median.


# Get the top 10 rebounders per year
nba_topPoints_perYear = nba[["points", "year"]].groupby("year")["points"].nlargest(10)

# Get the median of these 10
nba_topPoints_median_perYear = nba_topPoints_perYear.groupby("year").median()

# Put year back in as a column
nba_topPoints_median_perYear = nba_topPoints_median_perYear.reset_index()

# Again no zeros...
nba_topPoints_median_perYear_noZeros = nba_topPoints_median_perYear[nba_topPoints_median_perYear["points"] > 0]

# Now plot
sns.regplot(data=nba_topPoints_median_perYear_noZeros, x="year", y="points").set_title("Median of Top 10 points Each Year")
```

```
Text(0.5, 1.0, 'Median of Top 10 points Each Year')
```



0 s se ejecutó 12:38

PART II - COME UP WITH SUPPORTING EVIDENCE

1. Some players score a lot of points because they attempt a lot of shots. Among players that have scored a lot of points, are there some that are much more efficient (points per attempt) than others?

```
[35] PART 2 - A
     Some players score a lot of points because they attempt a lot of shots.
     Among players that have scored a lot of points, are there some that are much more efficient (points per attempt) than others?
     """
     """  FTA Percent of Team's Free Throws Attempted"""

     #This print is showing certain columns and making it easy to read
     print(players[["last_name", "first_name",'ft_attempted', 'ft_made']].sort_values("ft_attempted", ascending=False).head(10))

          last_name first_name  ft_attempted  ft_made
245   Chamberlain       Wilt          16.0      8.0
246   Chamberlain       Wilt          15.0      8.0
100        Baylor      Elgin          14.0     12.0
249   Chamberlain       Wilt          14.0     11.0
1251    Robertson      Oscar          14.0      8.0
789        Jordan    Michael          13.0      9.0
717       Johnson        Gus          13.0     11.0
151      Blackman    Rolando          13.0     11.0
110        Beatty      Zelmo          13.0     10.0
1430     Thompson      David          13.0     11.0

[33]

     print(players[["last_name", "first_name",'ft_made', 'ft_attempted']].sort_values("ft_made", ascending=False).head(10))

          last_name first_name  ft_made  ft_attempted
100        Baylor      Elgin     12.0          14.0
1248    Robertson      Oscar     12.0          13.0
1165       Pettit        Bob     11.0          12.0
249   Chamberlain       Wilt     11.0          14.0
94         Baylor      Elgin     11.0          12.0
151      Blackman    Rolando     11.0          13.0
1430     Thompson      David     11.0          13.0
717       Johnson        Gus     11.0          13.0
```

```
[32] """ FGA  Percent of Team's Field Goals Attempted"""

     print(players[["last_name", "first_name",'fg_attempted','fg_made']].sort_values("fg_attempted", ascending=False).head(10))

         last_name first_name  fg_attempted  fg_made
87        Barry       Rick          27.0     16.0
798      Jordan    Michael          27.0      9.0
977     McGrady      Tracy          26.0     15.0
513      Gervin     George          26.0     14.0
999       Mikan     George          26.0      9.0
196      Bryant       Kobe          25.0     12.0
237    Chambers        Tom          25.0     13.0
791      Jordan    Michael          25.0     10.0
1219       Rice       Glen          24.0     10.0
789      Jordan    Michael          24.0     10.0

[31] print(players[["last_name", "first_name",'fg_made', 'fg_attempted']].sort_values("fg_made", ascending=False).head(10))

          last_name first_name  fg_made  fg_attempted
488        Garnett      Kevin     17.0          24.0
245    Chamberlain       Wilt     17.0          23.0
794        Jordan    Michael     17.0          23.0
87          Barry       Rick     16.0          27.0
977       McGrady      Tracy     15.0          26.0
513        Gervin     George     14.0          26.0
1404    Stoudemire      Amare     14.0          22.0
421        Erving     Julius     14.0          22.0
683       Iverson      Allen     13.0          23.0
1069        Nater       Swen     13.0          24.0
```

We can see some columns that show us different players that are more efficient in different parameters in the different attempts.

2. It seems like some players may excel in one statistical category, but produce very little in other areas. Are there any players that are exceptional across many categories?

```
[55] print(players[["last_name", "first_name",'assists','points', 'three_attempted']].sort_values("three_attempted", ascending=False).head(10))
```

	last_name	first_name	assists	points	three_attempted
32	Allen	Ray	1.0	17.0	11.0
977	McGrady	Tracy	2.0	36.0	10.0
26	Allen	Ray	5.0	15.0	10.0
701	James	LeBron	2.0	29.0	10.0
202	Bryant	Kobe	6.0	31.0	9.0
31	Allen	Ray	1.0	28.0	9.0
122	Billups	Chauncey	5.0	17.0	8.0
193	Bryant	Kobe	4.0	27.0	8.0
703	James	LeBron	6.0	28.0	8.0
700	James	LeBron	9.0	27.0	7.0

```
[56] print(players[["last_name", "first_name",'assists','points', 'three_made']].sort_values("three_made", ascending=False).head(10))
```

	last_name	first_name	assists	points	three_made
1204	Price	Mark	4.0	19.0	6.0
122	Billups	Chauncey	5.0	17.0	5.0
31	Allen	Ray	1.0	28.0	5.0
32	Allen	Ray	1.0	17.0	5.0
1152	Payton	Gary	6.0	18.0	4.0
1218	Rice	Glen	0.0	16.0	4.0
977	McGrady	Tracy	2.0	36.0	4.0
703	James	LeBron	6.0	28.0	4.0
701	James	LeBron	2.0	29.0	4.0
1219	Rice	Glen	1.0	26.0	4.0

```
#Michael Jordan, Wilt Chamberlain and Magic johnson are names that are on the top of the most of sorted columns...
#they show some frecuency inside the sorted data presented.
```

As the comment described that are some players that are exceptional in many categories. Maybe I made this hard, but I analyze different sorted columns and verified when seeing, that some players are in almost top of. The columns showed me with frequency: Michael Jordan, Wilt Chamberlain, Magic Johnson and LeBron James. This was based on the columns data I analyzed.

3. Much has been said about the rise of the three-point shot in recent years. It seems that players are shooting and making more three-point shots than ever. Recognizing that this dataset doesn't contain the very most recent data, do you see a trend of more three-point shots either across the league or among certain groups of players? Is there a point at which popularity increased dramatically?

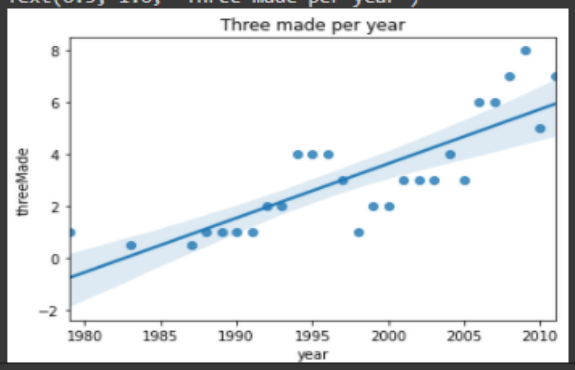
As we can see below, there was an increasing of three-points in recent years.

```
Recognizing that this dataset doesn't contain the very most recent data,
do you see a trend of more three-point shots either across the league or among certain groups of players?
Is there a point at which popularity increased dramatically?
"""

nba_year_set = players[["threeMade", "year"]].groupby("year").median()

print(nba_year_set)
```

```
nba_year_set = nba_year_set.reset_index()
sns.regplot(data=nba_year_set, x="year", y="threeMade")
```

```
[42] #let's remove any years where the median was 0
     nba_year_set = nba_year_set[nba_year_set["threeMade"] > 0]
     sns.regplot(data=nba_year_set, x="year", y="threeMade").set_title("Three made per year")
```

```
Text(0.5, 1.0, 'Three made per year')
```



0 s se ejecutó 16:14

PART III - SHOW CREATIVITY

1. The biographical data in this dataset contains information about home towns, home states, and home countries for these players. Can you find anything interesting about players who came from a similar location?

```
highschool , hscity , hsstate , hscountry , deathdate , race ],
dtype='object')
```

```
[33] """
     PART 3 - B
     The biographical data in this dataset contains information about home towns, home states,
     and home countries for these players. Can you find anything interesting about players who came from a similar location?
     """
     states = nba.birthState.value_counts()
     print(states.head(5))
```

```
CA    2048
NY    1956
IL    1465
PA    1136
OH    1075
Name: birthState, dtype: int64
```

```
college = nba.college.value_counts()
print(college.head(5))
```

```
none              1320
UCLA               522
North Carolina     447
Kentucky           421
Kansas             362
Name: college, dtype: int64
```

I chose to analyze two data of precedence, college and state. The interesting thing was that the most of players come from California and the second college of players precedence is from there (UCLA). This make me think that it is a very popular sport there.

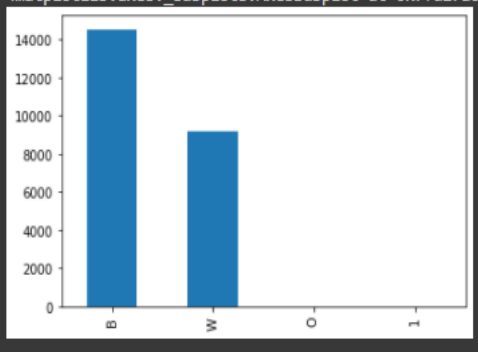
2. Find something else in this dataset that you consider interesting. Produce a graph to communicate your insight.

```
"""
3 PART - C

Find something else in this dataset that you consider interesting. Produce a graph to communicate your insigh
"""
race = nba.race.value_counts()
print(race)
```

```
B    14529
W     9164
O       29
1       21
Name: race, dtype: int64
```

```
race.plot(kind = 'bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd27a065d90>
```



This data was interesting. As a foreign I always saw (when I watched something about to basketball) that the most players were Afro-descendants. When counting the number per race I've found that it has some relevance the difference between black person or white person playing basketball. (Black and white is like the data was classified).