w14 CS241, Carlos W Mercado

In [1]:
```
import pandas as pd
import os
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```
pd.options.display.max_columns = None
```

In [3]:
```
players_original = pd.read_csv('basketball_players.csv', low_memory=False)
```

In [4]:
```
players = pd.read_csv('basketball_players.csv', low_memory=False)
players
```

Out[4]:

	playerID	year	stint	tmID	lgID	GP	GS	minutes	points	oRebounds	dRebounds	rebounds	assists	steals	blocks	turn
0	abramjo01	1946	1	PIT	NBA	47	0	0	527	0	0	0	35	0	0	
1	aubucch01	1946	1	DTF	NBA	30	0	0	65	0	0	0	20	0	0	
2	bakerno01	1946	1	CHS	NBA	4	0	0	0	0	0	0	0	0	0	
3	baltihe01	1946	1	STB	NBA	58	0	0	138	0	0	0	16	0	0	
4	barrjo01	1946	1	STB	NBA	58	0	0	295	0	0	0	54	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
23746	kaisero01	1962	0	PHT	ABL1	27	0	978	467	0	0	140	75	0	0	
23747	spragbr01	1962	0	PHT	ABL1	27	0	746	356	0	0	216	21	0	0	
23748	tayloro02	1962	0	PHT	ABL1	28	0	1007	355	0	0	107	134	0	0	
23749	wellsra01	1962	0	PHT	ABL1	2	0	36	4	0	0	6	3	0	0	
23750	wrighle01	1962	0	PHT	ABL1	28	0	813	195	0	0	257	32	0	0	

23751 rows × 42 columns

In [5]:
```
players.columns
```

Out[5]:
```
Index(['playerID', 'year', 'stint', 'tmID', 'lgID', 'GP', 'GS', 'minutes',
       'points', 'oRebounds', 'dRebounds', 'rebounds', 'assists', 'steals',
       'blocks', 'turnovers', 'PF', 'fgAttempted', 'fgMade', 'ftAttempted',
       'ftMade', 'threeAttempted', 'threeMade', 'PostGP', 'PostGS',
       'PostMinutes', 'PostPoints', 'PostoRebounds', 'PostdRebounds',
       'PostRebounds', 'PostAssists', 'PostSteals', 'PostBlocks',
       'PostTurnovers', 'PostPF', 'PostfgAttempted', 'PostfgMade',
       'PostftAttempted', 'PostftMade', 'PostthreeAttempted', 'PostthreeMade',
       'note'],
      dtype='object')
```

(1) Many sports analysts argue about which player is the GOAT (the Greatest Of All Time). Based on this data, who would you say is the GOAT? Provide evidence to back up your decision.

In [6]:
```
# Let's just get the colums we are interested in
selection = ['playerID', 'GP', 'minutes', 'points', 'rebounds', 'assists', 'steals', 'blocks', 'turnovers',
'PF', 'fgMade', 'ftMade', 'threeMade']
players_interesting_data = players_original[selection]
players_interesting_data
```

Out[6]:

	playerID	GP	minutes	points	rebounds	assists	steals	blocks	turnovers	PF	fgMade	ftMade	threeMade
0	abramjo01	47	0	527	0	35	0	0	0	161	202	123	0
1	aubucch01	30	0	65	0	20	0	0	0	46	23	19	0
2	bakerno01	4	0	0	0	0	0	0	0	0	0	0	0
3	baltihe01	58	0	138	0	16	0	0	0	98	53	32	0
4	barrjo01	58	0	295	0	54	0	0	0	164	124	47	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
23746	kaisero01	27	978	467	140	75	0	0	0	53	159	124	25
23747	spragbr01	27	746	356	216	21	0	0	0	77	147	61	1
23748	tayloro02	28	1007	355	107	134	0	0	0	78	134	82	5
23749	wellsra01	2	36	4	6	3	0	0	0	2	1	2	0
23750	wrighle01	28	813	195	257	32	0	0	0	78	75	44	1

23751 rows × 13 columns

In [7]:
```
# Let's combine all rows by playerID
players_grouped_byName = players_interesting_data.groupby('playerID').sum()
players_grouped_byName
```

Out[7]:

playerID	GP	minutes	points	rebounds	assists	steals	blocks	turnovers	PF	fgMade	ftMade	threeMade
abdelal01	256	3200	1465	846	85	71	70	247	484	620	225	0
abdulka01	1560	57446	38387	17440	5660	1160	3189	2527	4657	15837	6712	1
abdulma01	724	19202	9087	2146	3555	26	6	0	2027	3597	1893	0
abdulma02	586	15628	8553	1087	2079	487	46	963	1106	3514	1051	474
abdulta01	236	4807	1830	776	266	184	83	309	485	720	372	18
...	...	...	...	...	...	...	...	...	...	...	...	...
ziegeba01	22	0	45	0	0	0	0	0	0	19	7	0
zimmede01	2	32	4	4	7	0	0	4	4	2	0	0
zoetji01	7	30	2	8	1	1	3	4	9	1	0	0
zopfbi01	53	398	118	46	73	0	0	0	34	49	20	0
zunicma01	113	0	604	0	50	0	0	0	391	221	162	0

4903 rows × 12 columns

In [8]:
```
# The GOAT player should excel in all of these areas, so lets drop all those rows with at least one 0
players_noZeroStats = players_grouped_byName[players_grouped_byName != 0].dropna()
players_noZeroStats
```

Out[8]:

playerID	GP	minutes	points	rebounds	assists	steals	blocks	turnovers	PF	fgMade	ftMade	threeMade
abdulka01	1560.0	57446.0	38387.0	17440.0	5660.0	1160.0	3189.0	2527.0	4657.0	15837.0	6712.0	1.0
abdulma02	586.0	15628.0	8553.0	1087.0	2079.0	487.0	46.0	963.0	1106.0	3514.0	1051.0	474.0
abdulta01	236.0	4807.0	1830.0	776.0	266.0	184.0	83.0	309.0	485.0	720.0	372.0	18.0
abdursh01	830.0	28878.0	15028.0	6239.0	2109.0	820.0	638.0	2134.0	2324.0	5434.0	4006.0	154.0
ackeral01	30.0	234.0	81.0	29.0	16.0	6.0	4.0	11.0	13.0	34.0	5.0	8.0
...	...	...	...	...	...	...	...	...	...	...	...	...
youngni01	357.0	8178.0	4086.0	674.0	337.0	191.0	69.0	416.0	702.0	1507.0	699.0	373.0
youngsa01	193.0	3273.0	1281.0	450.0	138.0	119.0	49.0	171.0	249.0	502.0	244.0	33.0
youngth01	361.0	10171.0	4534.0	1800.0	397.0	408.0	113.0	462.0	721.0	1927.0	563.0	117.0
zhizhwa01	137.0	1254.0	604.0	231.0	39.0	25.0	35.0	68.0	155.0	213.0	108.0	70.0
zidekge01	135.0	1328.0	453.0	286.0	32.0	14.0	12.0	69.0	241.0	161.0	130.0	1.0

1704 rows × 12 columns

In [9]:
```
# Let's find a 0.9 quantile for this group, which will represent all the stats above the 90%
selection = ['GP', 'minutes', 'points', 'rebounds', 'assists', 'steals', 'blocks', 'turnovers', 'PF', 'fgMad
e', 'ftMade', 'threeMade']
GOATplayer_90 = players_noZeroStats[selection].quantile(0.9)
GOATplayer_90
```

Out[9]:
```
GP            892.4
minutes     26151.2
points      12403.6
rebounds     4657.0
assists      2957.6
steals        939.0
blocks        510.0
turnovers    1651.0
PF           2295.1
fgMade       4753.7
ftMade       2457.0
threeMade     550.5
Name: 0.9, dtype: float64
```

In [10]:
```
# Now let's compare the list of players with the GOATplayer >90% stats
GOAT = players_noZeroStats[
    (players_noZeroStats['GP'] > GOATplayer_90['GP']) &
    (players_noZeroStats['minutes'] > GOATplayer_90['minutes']) &
    (players_noZeroStats['points'] > GOATplayer_90['points']) &
    (players_noZeroStats['rebounds'] > GOATplayer_90['rebounds']) &
    (players_noZeroStats['assists'] > GOATplayer_90['assists']) &
    (players_noZeroStats['steals'] > GOATplayer_90['steals']) &
    (players_noZeroStats['blocks'] > GOATplayer_90['blocks']) &
    (players_noZeroStats['turnovers'] > GOATplayer_90['turnovers']) &
    (players_noZeroStats['PF'] > GOATplayer_90['PF']) &
    (players_noZeroStats['fgMade'] > GOATplayer_90['fgMade']) &
    (players_noZeroStats['ftMade'] > GOATplayer_90['ftMade']) &
    (players_noZeroStats['threeMade'] > GOATplayer_90['threeMade'])
]
GOAT
# We'll get a preliminar list of the best players
```

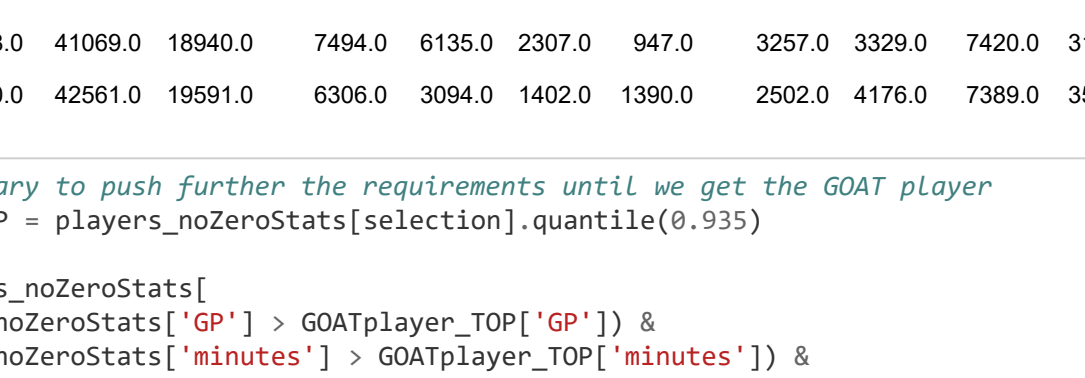
Out[10]:

playerID	GP	minutes	points	rebounds	assists	steals	blocks	turnovers	PF	fgMade	ftMade	threeMade
bryanko01	1161.0	42384.0	29484.0	6141.0	5419.0	1722.0	594.0	3432.0	2991.0	10286.0	7407.0	1505.0
cartevi01	986.0	35173.0	21135.0	5018.0	3836.0	1145.0	665.0	2131.0	2867.0	7713.0	4208.0	1501.0
drexlcl01	1086.0	37537.0	22195.0	6677.0	6125.0	2207.0	719.0	2977.0	3285.0	8335.0	4698.0	827.0
jordami01	1072.0	41013.0	32292.0	6672.0	5633.0	2514.0	893.0	2924.0	2783.0	12192.0	7327.0	581.0
piercpa01	1025.0	37783.0	22591.0	6160.0	3935.0	1501.0	637.0	2998.0	2914.0	7406.0	6101.0	1678.0
pippesc01	1178.0	41069.0	18940.0	7494.0	6135.0	2307.0	947.0	3257.0	3329.0	7420.0	3122.0	978.0
robincl02	1380.0	42561.0	19591.0	6306.0	3094.0	1402.0	1390.0	2502.0	4176.0	7389.0	3560.0	1253.0

In [11]:
```
# It's necessary to push further the requirements until we get the GOAT player
GOATplayer_TOP = players_noZeroStats[selection].quantile(0.935)
# Now
GOAT = players_noZeroStats[
    (players_noZeroStats['GP'] > GOATplayer_TOP['GP']) &
    (players_noZeroStats['minutes'] > GOATplayer_TOP['minutes']) &
    (players_noZeroStats['points'] > GOATplayer_TOP['points']) &
    (players_noZeroStats['rebounds'] > GOATplayer_TOP['rebounds']) &
    (players_noZeroStats['assists'] > GOATplayer_TOP['assists']) &
    (players_noZeroStats['steals'] > GOATplayer_TOP['steals']) &
    (players_noZeroStats['blocks'] > GOATplayer_TOP['blocks']) &
    (players_noZeroStats['turnovers'] > GOATplayer_TOP['turnovers']) &
    (players_noZeroStats['PF'] > GOATplayer_TOP['PF']) &
    (players_noZeroStats['fgMade'] > GOATplayer_TOP['fgMade']) &
    (players_noZeroStats['ftMade'] > GOATplayer_TOP['ftMade']) &
    (players_noZeroStats['threeMade'] > GOATplayer_TOP['threeMade'])
]
# Here's the man, accordingly with my stats selection
GOAT
```

Out[11]:

playerID	GP	minutes	points	rebounds	assists	steals	blocks	turnovers	PF	fgMade	ftMade	threeMade
pippesc01	1178.0	41069.0	18940.0	7494.0	6135.0	2307.0	947.0	3257.0	3329.0	7420.0	3122.0	978.0



(2) 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?

In [12]:
```
master = pd.read_csv('basketball_master.csv', low_memory=False)
# There's an extra useless row at index row 0. Let's drop it.
master = master.drop(0)
master
```

Out[12]:

	bioID	useFirst	firstName	middleName	lastName	nameGiven	fullGivenName	nameSuffix	nameNick	pos	firstseason	l
1	abdelal01	Alaa	Alaa	NaN	Abdelnaby	NaN	NaN	NaN	NaN	F-C	0.0	
2	abdulka01	Kareem	Kareem	NaN	Abdul-Jabbar	NaN	Ferdinand Lewis Alcindor, Jr.	NaN	Lew, Cap	C	0.0	
3	abdulma01	Mahdi	Mahdi	NaN	Abdul-Rahman	NaN	Walter Raphael Hazzard, Jr.	NaN	Walt	G	0.0	
4	abdulma02	Mahmoud	Mahmoud	NaN	Abdul-Rauf	NaN	Chris Wayne Jackson	NaN	NaN	G	0.0	
5	abdulta01	Tariq	Tariq	NaN	Abdul-Wahad	NaN	Olivier Michael Saint-Jean	NaN	NaN	G-F	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	
5057	rosegl01	Glen	Glen	NaN	Rose	NaN	NaN	NaN	NaN	NaN	NaN	
5058	shephle01	Len	Len	NaN	Shepherd	NaN	NaN	NaN	NaN	NaN	NaN	
5059	glammge01	George	George	NaN	Glammack	NaN	NaN	NaN	NaN	NaN	NaN	
5060	eurasge01	Gene	Gene	NaN	Eurash	NaN	NaN	NaN	NaN	NaN	NaN	
5061	koliswa01	Walt	Walt	NaN	Kolish	NaN	NaN	NaN	NaN	NaN	NaN	

5061 rows × 26 columns

In [13]:
```
# Let's reduce the original datagrame
players_reduced = players_original.groupby('playerID').sum()
players_reduced = players_reduced.reset_index()
players_reduced
```

Out[13]:

	playerID	year	stint	GP	GS	minutes	points	oRebounds	dRebounds	rebounds	assists	steals	blocks	turnovers	Pl
0	abdelal01	13946	9	256	0	3200	1465	283	563	846	85	71	70	247	484
1	abdulka01	39570	20	1560	0	57446	38387	2975	9394	17440	5660	1160	3189	2527	465
2	abdulma01	21657	12	724	0	19202	9087	18	39	2146	3555	26	6	0	202
3	abdulma02	17948	9	586	0	15628	8553	219	868	1087	2079	487	46	963	110
4	abdulta01	15997	10	236	0	4807	1830	286	490	776	266	184	83	309	485
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	..
4898	ziegeba01	3886	0	22	0	0	45	0	0	0	0	0	0	0	(
4899	zimmede01	2005	1	2	0	32	4	1	3	4	7	0	0	4	4
4900	zoetji01	1982	1	7	0	30	2	3	5	8	1	1	3	4	9
4901	zopfbi01	1970	1	53	0	398	118	0	0	46	73	0	0	0	34
4902	zunicma01	3895	1	113	0	0	604	0	0	0	50	0	0	0	39

4903 rows × 39 columns

In [14]:
```
# Now let's merge this master dataframe with the players_reduced dataframe
nba = pd.merge(players_reduced, master, how='left', left_on='playerID', right_on='bioID')
# Where there's no matches between 'playerID' and 'bioID', rows are dropped
```

In [15]:
```
nba.columns
```

Out[15]:
```
Index(['playerID', 'year', 'stint', 'GP', 'GS', 'minutes', 'points',
       'oRebounds', 'dRebounds', 'rebounds', 'assists', 'steals', 'blocks',
       'turnovers', 'PF', 'fgAttempted', 'fgMade', 'ftAttempted', 'ftMade',
       'threeAttempted', 'threeMade', 'PostGP', 'PostGS', 'PostMinutes',
       'PostPoints', 'PostoRebounds', 'PostdRebounds', 'PostRebounds',
       'PostAssists', 'PostSteals', 'PostBlocks', 'PostTurnovers', 'PostPF',
       'PostfgAttempted', 'PostfgMade', 'PostftAttempted', 'PostftMade',
       'PostthreeAttempted', 'PostthreeMade', 'bioID', 'useFirst', 'firstName',
       'middleName', 'lastName', 'nameGiven', 'fullGivenName', 'nameSuffix',
       'nameNick', 'pos', 'firstseason', 'lastseason', 'height', 'weight',
       'college', 'collegeOther', 'birthDate', 'birthCity', 'birthState',
       'birthCountry', 'highSchool', 'hsCity', 'hsState', 'hsCountry',
       'deathDate', 'race'],
      dtype='object')
```

In [16]:
```
# Now let's make a selection over this data
selection = ['GP', 'minutes', 'points', 'rebounds', 'assists', 'steals', 'blocks', 'turnovers', 'PF', 'fgMad
e', 'ftMade', 'threeMade', 'birthCity', 'birthState', 'birthCountry']
nba = nba[selection]
nba
```

Out[16]:

	GP	minutes	points	rebounds	assists	steals	blocks	turnovers	PF	fgMade	ftMade	threeMade	birthCity	birthState	bi
0	256	3200	1465	846	85	71	70	247	484	620	225	0	Cairo	NaN	
1	1560	57446	38387	17440	5660	1160	3189	2527	4657	15837	6712	1	New York	NY	
2	724	19202	9087	2146	3555	26	6	0	2027	3597	1893	0	Wilmington	DE	
3	586	15628	8553	1087	2079	487	46	963	1106	3514	1051	474	Gulfport	MS	
4	236	4807	1830	776	266	184	83	309	485	720	372	18	Maisons Alfort	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4898	22	0	45	0	0	0	0	0	0	19	7	0	Chicago	IL	
4899	2	32	4	4	7	0	0	4	4	2	0	0	Monroe	LA	
4900	7	30	2	8	1	1	3	4	9	1	0	0	Uxbridge	ON	
4901	53	398	118	46	73	0	0	0	34	49	20	0	NaN	NaN	
4902	113	0	604	0	50	0	0	0	391	221	162	0	Renton	PA	

4903 rows × 15 columns

In [17]:
```
nba_countries = nba.groupby('birthCountry').sum()
nba_countries.reset_index()
```

Out[17]:

	birthCountry	GP	minutes	points	rebounds	assists	steals	blocks	turnovers	PF	fgMade	ftMade	threeMade
0	ARG	2577	59490	26553	11076	5126	2008	762	3549	5719	9565	5191	2232
1	AUS	1561	32794	12451	7963	2486	777	1325	2289	3968	5116	2000	219
2	BAH	989	28651	13184	7118	2199	725	1079	2167	2820	5343	2497	1
3	BEL	910	27984	14004	2616	4812	822	74	2095	1593	5556	2523	369
4	BOS	184	1766	617	231	171	61	8	139	263	218	138	43
...	...	...	...	...	...	...	...	...	...	...	...	...	...
75	USA	1018692	23937005	10657827	4516062	2370715	647695	369884	1201665	2419731	4089590	2193833	269159
76	VEN	218	3748	1280	380	553	111	23	275	282	471	226	112
77	VIN	733	13061	2989	3461	344	264	1193	597	1575	1318	353	0
78	YUG	4924	109449	42007	20761	9312	3644	2983	7026	11636	16091	7487	2338
79	ZAI	1197	36789	11729	12359	1240	494	3289	2173	3383	4169	3391	0

80 rows × 13 columns

Now let's find something interestin about these people

In [18]:
```
# Countries with more points
nba_countries[['points']].sort_values(by=['points'], ascending=False).head(10).reset_index()
```

Out[18]:

	birthCountry	points
0	USA	10657827
1	GER	81819
2	FRA	50890
3	YUG	42007
4	ESP	39142
5	CRO	32815
6	NGR	32023
7	ISV	29558
8	LTU	29039
9	JAM	28081

In [19]:
```
# Top 5 USA states with more minutes played
nba_states = nba.groupby('birthState').sum()
nba_states.reset_index()
nba_states_sorted = nba_states[['minutes']].sort_values(by=['minutes'], ascending=False)
nba_states_sorted.head()
```

Out[19]:

birthState	minutes
CA	2374086
NY	2322038
IL	1677843
PA	1388634
OH	1287481

In [20]:
```
# Bottom 5 USA states with least minutes played
nba_states_sorted.tail()
```

Out[20]:

birthState	minutes
BC	3429
(Fr Guiana)	1811
FK	1804
KS	779
NSW	431

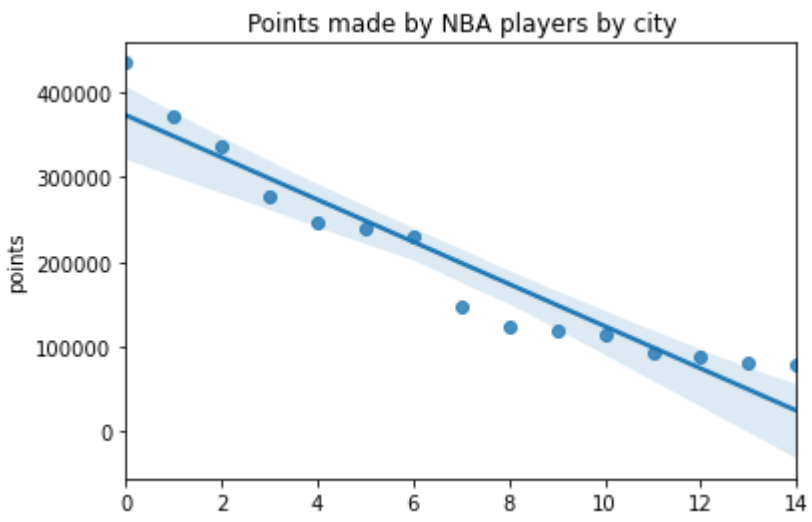
In [21]:
```
# And now, worldwide cities that excel producing great players (just the top 15)
nba_cities = nba.groupby('birthCity').sum()
nba_cities.reset_index()
# The >90% required for compative purposes
stats_excel = nba_cities[['GP', 'minutes', 'points', 'rebounds', 'assists', 'steals', 'blocks', 'turnovers',
'PF', 'fgMade', 'ftMade', 'threeMade']].quantile(0.9)
cities_excel = nba_cities[
    (nba_cities['GP'] > stats_excel['GP']) &
    (nba_cities['minutes'] > stats_excel['minutes']) &
    (nba_cities['points'] > stats_excel['points']) &
    (nba_cities['rebounds'] > stats_excel['rebounds']) &
    (nba_cities['assists'] > stats_excel['assists']) &
    (nba_cities['steals'] > stats_excel['steals']) &
    (nba_cities['blocks'] > stats_excel['blocks']) &
    (nba_cities['turnovers'] > stats_excel['turnovers']) &
    (nba_cities['PF'] > stats_excel['PF']) &
    (nba_cities['fgMade'] > stats_excel['fgMade']) &
    (nba_cities['ftMade'] > stats_excel['ftMade']) &
    (nba_cities['threeMade'] > stats_excel['threeMade'])
]
# And let's sort this findings by points
cities_excel = cities_excel.sort_values(by=['points'], ascending=False).reset_index().head(15)
cities_excel
```

Out[21]:

	birthCity	GP	minutes	points	rebounds	assists	steals	blocks	turnovers	PF	fgMade	ftMade	threeMade
0	Chicago	39767	951715	434684	167298	107547	33082	11836	53796	92398	169042	83266	12402
1	Brooklyn	29624	728137	371644	129513	94516	15695	6315	34711	77200	139035	87715	5859
2	Philadelphia	29671	757259	337000	147162	79767	16333	7758	30233	70779	130044	69216	7696
3	New York	23996	566747	276597	105633	67402	11648	8729	24112	57658	104195	64120	4087
4	Los Angeles	25121	575480	245981	98489	58919	18437	8417	30487	57258	94186	48999	8610
5	Washington	22907	534601	239874	99469	50897	14270	6277	26285	52735	91079	52106	5118
6	Detroit	20545	502851	229986	99887	42505	13205	10043	30125	50785	91010	43596	4370
7	Atlanta	13182	333592	147474	62090	29212	10357	5750	17082	32073	57960	29163	2391
8	Dallas	12294	287692	122717	59222	23979	7613	6998	15102	30077	46798	27528	1593
9	Indianapolis	9736	236669	119172	34642	23853	6566	2165	11272	22474	45025	24963	3245
10	Oakland	9765	249953	113623	46936	26919	7763	3683	14932	22868	42934	23072	4683
11	Seattle	8116	206160	93691	28368	23280	6754	2658	13081	19300	35090	17857	5654
12	Baltimore	8550	204697	87441	27520	25308	7655	2959	11972	18492	33736	17379	2590
13	Columbus	8030	187708	81842	33947	14322	4384	3553	10078	17767	31353	17274	1862
14	San Francisco	7223	169036	78329	33028	23754	4107	914	7853	16315	30306	15756	1961

In [22]:
```
# Now let's plot the points made by all the players by city. Chicago goes first
sns.regplot(data=cities_excel, x=cities_excel.index, y='points').set_title('Points made by NBA players by cit
y')
```

Out[22]: Text(0.5, 1.0, 'Points made by NBA players by city')



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

In [23]:
```
# Let's get a graph that displays the relationship between height and weight for ALL nba players
master.columns
```

Out[23]:
```
Index(['bioID', 'useFirst', 'firstName', 'middleName', 'lastName', 'nameGiven',
       'fullGivenName', 'nameSuffix', 'nameNick', 'pos', 'firstseason',
       'lastseason', 'height', 'weight', 'college', 'collegeOther',
       'birthDate', 'birthCity', 'birthState', 'birthCountry', 'highSchool',
       'hsCity', 'hsState', 'hsCountry', 'deathDate', 'race'],
      dtype='object')
```

In [24]:
```
selection = ['height', 'weight']
sizes = master[selection]
sizes
```

Out[24]:

	height	weight
1	82.0	240.0
2	85.0	225.0
3	74.0	185.0
4	73.0	162.0
5	78.0	223.0
...	...	...
5057	NaN	NaN
5058	NaN	NaN
5059	NaN	NaN
5060	NaN	NaN
5061	NaN	NaN

5061 rows × 2 columns

In [25]:
```
# Drop those rows with 0 and NaN values
sizes = sizes[sizes['height'].notna() & sizes['weight'].notna()]
```

In [26]:
```
sizes = sizes[sizes['height'] > 0]
```

In [27]:
```
sizes = sizes[sizes['weight'] > 0]
sizes
```

Out[27]:

	height	weight
1	82.0	240.0
2	85.0	225.0
3	74.0	185.0
4	73.0	162.0
5	78.0	223.0
...	...	...
4988	72.0	170.0
4998	70.0	170.0
5010	73.0	180.0
5020	71.0	175.0
5034	69.0	150.0

4817 rows × 2 columns

In [28]:
```
sns.regplot(data=sizes, x='weight', y='height').set_title('Points made by NBA players by city')
```

Out[28]:
```
Text(0.5, 1.0, 'Points made by NBA players by city')
```