Aggregate Functions Lab

Introduction

In this lab we will query data from a table populated with Babe Ruth's career hitting statistics. We will use aggregate functions to pull interesting information from the table that basic queries cannot track. We will discover many interesting facts about Babe Ruth, like his total career homeruns and his most homeruns in one season.

Objectives

- Write queries with aggregate functions like `COUNT`, `MAX`, `MIN`, and `SUM`
- Create an alias for the return value of an aggregate function
- Use `GROUP BY` to sort the data sets returned by aggregate functions
- Compare aggregates using the `HAVING` clause

Babe Ruth -- Career Hitting Statistics

We will query from the `babe_ruth_stats` table featured below.

year	team	league	doubles	triples	hits	HR	games	runs	RBI	at_bats	BB	SB	SO	AVG
1914	"BOS"	"AL"	1	0	2	0	5	1	2	10	0	0	4	0.2
1915	"BOS"	"AL"	10	1	29	4	42	16	21	92	9	0	23	0.315
1916	"BOS"	"AL"	5	3	37	3	67	18	15	136	10	0	23	0.272
1917	"BOS"	"AL"	6	3	40	2	52	14	12	123	12	0	18	0.325
1918	"BOS"	"AL"	26	11	95	11	95	50	66	317	58	6	58	0.3
1919	"BOS"	"AL"	34	12	139	29	130	103	114	432	101	7	58	0.322
1920	"NY"	"AL"	36	9	172	54	142	158	137	458	150	14	80	0.376
1921	"NY"	"AL"	44	16	204	59	152	177	171	540	145	17	81	0.378
1922	"NY"	"AL"	24	8	128	35	110	94	99	406	84	2	80	0.315
1923	"NY"	"AL"	45	13	205	41	152	151	131	522	170	17	93	0.393
1924	"NY"	"AL"	39	7	200	46	153	143	121	529	142	9	81	0.378
1925	"NY"	"AL"	12	2	104	25	98	61	66	359	59	2	68	0.29
1926	"NY"	"AL"	30	5	184	47	152	139	146	495	144	11	76	0.372
1927	"NY"	"AL"	29	8	192	60	151	158	164	540	137	7	89	0.356
1928	"NY"	"AL"	29	8	173	54	154	163	142	536	137	4	87	0.323
1929	"NY"	"AL"	26	6	172	46	135	121	154	499	72	5	60	0.345
1930	"NY"	"AL"	28	9	186	49	145	150	153	518	136	10	61	0.359
1931	"NY"	"AL"	31	3	199	46	145	149	163	534	128	5	51	0.373
1932	"NY"	"AL"	13	5	156	41	133	120	137	457	130	2	62	0.341
1933	"NY"	"AL"	21	3	138	34	137	97	103	459	114	4	90	0.301
1934	"NY"	"AL"	17	4	105	22	125	78	84	365	104	1	63	0.288
1935	"BOS"	"NL"	0	0	13	6	28	13	12	72	20	0	24	0.181

Connect to the Database

In [1]:
```
import sqlite3
import pandas as pd
```

In [3]:
```
conn = sqlite3.connect('babe_ruth.db')
cur = conn.cursor()
```

Queries

total_seasons

Counts the total number of `year`s that Babe Ruth played professional baseball

In [9]:
```
cur.execute("""select count(year) as num_years
                        from babe_ruth_stats;""")
df = pd.DataFrame(cur.fetchall())
df.columns = [i[0] for i in cur.description]
df
```

Out[9]:

	num_years
0	22

total_seasons_with_ny

Counts the total number of `year`s played with the `NY` Yankees

In [10]:
```
cur.execute("""select count(year) as num_years
                        from babe_ruth_stats
                        where team ="NY";""")
df = pd.DataFrame(cur.fetchall())
df.columns = [i[0] for i in cur.description]
df
```

Out[10]:

	num_years
0	15

most_hr

Selects the most `HR` that Babe Ruth hit in one season

In [12]:
```
cur.execute("""select *
                        from babe_ruth_stats
                        where HR = (select max(HR) from babe_ruth_stats);""")
df = pd.DataFrame(cur.fetchall())
df.columns = [i[0] for i in cur.description]
df
```

Out[12]:

	id	year	team	league	doubles	triples	hits	HR	games	runs	RBI	at_bats	BB	SB	SO	AVG
0	14	1927	NY	AL	29	8	192	60	151	158	164	540	137	7	89	0.356

least_hr

Select the least number of `HR` hit in one season

In [13]:
```
cur.execute("""select *
                        from babe_ruth_stats
                        where HR = (select min(HR) from babe_ruth_stats);""")
df = pd.DataFrame(cur.fetchall())
df.columns = [i[0] for i in cur.description]
df
```

Out[13]:

	id	year	team	league	doubles	triples	hits	HR	games	runs	RBI	at_bats	BB	SB	SO	AVG
0	1	1914	BOS	AL	1	0	2	0	5	1	2	10	0	0	4	0.2

total_hr

Returns the total number of `HR` hit by Babe Ruth during his career

In [14]:
```
cur.execute("""select sum(HR)
                    from babe_ruth_stats;""")
df = pd.DataFrame(cur.fetchall())
df.columns = [i[0] for i in cur.description]
df
```

Out[14]:

	sum(HR)
0	714

year_and_games_with_least_hr

Babe Ruth hit 0 `HR` one year. That statistic might not be indicative of a typical Babe Ruth season if he played in only a handful of games that year. Let's figure out how many games he played that season. Select the `year` and `games` from the season in which Ruth hit 0 `HR`.

In [16]:
```
cur.execute("""select year, games
                    from babe_ruth_stats
                    where HR = 0;""")
df = pd.DataFrame(cur.fetchall())
df.columns = [i[0] for i in cur.description]
df
```

Out[16]:

	year	games
0	1914	5

select_yr_and_min_hr_with_at_least_100_games

We determined that Babe Ruth hit 0 homeruns in his first year, when he played only five games. Let's avoid the outliers by looking at years in which Ruth played in at least 100 games. Select the `year` with the least number of `HR` from only those seasons with over 100 `games` played.

In [17]:
```
cur.execute("""select year, hr
                    from babe_ruth_stats
                    where games > 100
                    group by 1
                    order by hr
                    limit 1;""")
df = pd.DataFrame(cur.fetchall())
df.columns = [i[0] for i in cur.description]
df
```

Out[17]:

	year	HR
0	1934	22

avg_batting_avg_aliased_as_career_average

Select the average, `AVG`, of Ruth's batting averages. The header of the result would be `AVG(AVG)` which is quite confusing, so provide an alias of `career_average`.

In [18]:
```
cur.execute("""select avg(AVG) as career_average
                    from babe_ruth_stats;""")
df = pd.DataFrame(cur.fetchall())
df.columns = [i[0] for i in cur.description]
df
```

Out[18]:

	career_average
0	0.322864

total_years_and_hits_per_team

Select the `team` and the total number of `year`s and `hits`, but represent the results on a per team basis. (**Hint**: you will need to sort the result with a certain clause...)

In [20]:
```
cur.execute("""select team,
                    count(year) as num_years,
                    sum(hits) as total_hits
                    from babe_ruth_stats
                    group by 1;""")
df = pd.DataFrame(cur.fetchall())
df.columns = [i[0] for i in cur.description]
df
```

Out[20]:

	team	num_years	total_hits
0	BOS	7	355
1	NY	15	2518

total_years_and_hr_per_team_ordered_by_hr

The previous query returns Babe Ruth's Boston stats first. However, the overwhelming majority of Ruth's career statistics came when he played for the `NY` Yankees. Shouldn't we list Ruth's `NY` stats first? Write the previous query again, but this time we want Babe Ruth's total `HR`s instead of his total `hits`. Make sure that the resulting data set lists Babe Ruth's stats as a Yankee first.

> **Hint**: You will need to chain another sorting clause after `GROUP BY`.

In [21]:
```
cur.execute("""select team,
                    count(year) as num_years,
                    sum(HR) as total_hr
                    from babe_ruth_stats
                    group by 1
                    order by sum(HR) desc;""")
df = pd.DataFrame(cur.fetchall())
df.columns = [i[0] for i in cur.description]
df
```

Out[21]:

	team	num_years	total_hr
0	NY	15	659
1	BOS	7	55

years_with_on_base_over_300

We want to know the years in which Ruth successfully reached base over 300 times. We need to add `hits` and `BB` to calculate how many times Ruth reached base. Simply add the two columns together (ie: `SELECT hits + BB FROM ...`) and give this value an alias of `on_base`. Select the `year` and `on_base` for only those years with an `on_base` over 300.

> **Hint**: `WHERE` won't work here!

In [22]:
```
cur.execute("""select year,
                    hits + BB as on_base
                    from babe_ruth_stats
                    group by 1
                    having hits + BB > 300
                    order by hits + BB desc;""")
df = pd.DataFrame(cur.fetchall())
df.columns = [i[0] for i in cur.description]
df
```

Out[22]:

	year	on_base
0	1923	375
1	1921	349

2	1924	342
3	1927	329
4	1926	328
5	1931	327
6	1920	322
7	1930	322
8	1928	310

Summary

Well done! In this lab we continued adding complexity to our SQL statements and wrote aggregate functions. We were able to build our queries from giving us totals and averages to showing us the total years and homeruns earned by team as well as calculating Babe Ruth's total on base and then selecting only years that met a minimum value of our calculated on base attribute.