Client Report - Finding Relationships in Baseball

See code

Course DS 250

AUTHOR
Wil Jones

▼ Show the code

```
import pandas as pd
import numpy as np
import sqlite3
from lets_plot import *

LetsPlot.setup_html(isolated_frame=True)
```

▼ Show the code

```
sqlite_file = 'lahmansbaseballdb.sqlite'
con = sqlite3.connect(sqlite_file)
```

QUESTION – TASK 1

Write an SQL query to create a new dataframe about baseball players who attended BYU-Idaho. The new table should contain five columns: playerID, schoolID, salary, and the yearID/teamID associated with each salary. Order the table by salary (highest to lowest) and print out the table in your report.

▼ Show the code

```
query = '''
SELECT s.playerID, MAX(sp.schoolID) AS schoolID, MAX(s.salary) AS salary, s.yearID, s.teamID
FROM CollegePlaying sp
JOIN Salaries s ON sp.playerID = s.playerID
WHERE sp.schoolID = 'idbyuid'
GROUP BY s.playerID, s.yearID, s.teamID
ORDER BY salary DESC;
'''
df_byu = pd.read_sql_query(query, con)
df_byu
```

	playerID	schoolID	salary	yearID	teamID
0	lindsma01	idbyuid	4000000.0	2014	CHA
1	lindsma01	idbyuid	3600000.0	2012	BAL

	playerID	schoolID	salary	yearID	teamID
2	lindsma01	idbyuid	2800000.0	2011	COL
3	lindsma01	idbyuid	2300000.0	2013	CHA
4	lindsma01	idbyuid	1625000.0	2010	HOU
5	stephga01	idbyuid	1025000.0	2001	SLN
6	stephga01	idbyuid	900000.0	2002	SLN
7	stephga01	idbyuid	800000.0	2003	SLN
8	stephga01	idbyuid	550000.0	2000	SLN
9	lindsma01	idbyuid	410000.0	2009	FLO
10	lindsma01	idbyuid	395000.0	2008	FLO
11	lindsma01	idbyuid	380000.0	2007	FLO
12	stephga01	idbyuid	215000.0	1999	SLN
13	stephga01	idbyuid	185000.0	1998	PHI
14	stephga01	idbyuid	150000.0	1997	PHI

QUESTION – TASK 2

This three-part question requires you to calculate batting average (number of hits divided by the number of at-bats)

a. At least 1 at-bat

▼ Show the code

```
query = '''
SELECT playerID, yearID, CAST(H AS FLOAT)/AB AS batting_avg
FROM Batting
WHERE AB >= 1
ORDER BY batting_avg DESC, playerID
LIMIT 5;
'''
df_avg_1ab = pd.read_sql_query(query, con)
df_avg_1ab
```

	playerID	yearID	batting_avg
0	aberal01	1957	1.0
1	abernte02	1960	1.0
2	abramge01	1923	1.0
3	acklefr01	1964	1.0
4	alanirj01	2019	1.0

b. At least 10 at-bats

▼ Show the code

```
query = '''
SELECT playerID, yearID, CAST(H AS FLOAT)/AB AS batting_avg
FROM Batting
WHERE AB >= 10
ORDER BY batting_avg DESC, playerID
LIMIT 5;
'''
df_avg_10ab = pd.read_sql_query(query, con)
df_avg_10ab
```

	playerID	yearID	batting_avg
0	nymanny01	1974	0.642857
1	carsoma01	2013	0.636364
2	altizda01	1910	0.600000
3	johnsde01	1975	0.600000
4	silvech01	1948	0.571429

c. Career batting average (100+ AB total)

▶ Show the code

	playerID	career_avg
0	cobbty01	0.366299
1	barnero01	0.359682
2	hornsro01	0.358497
3	jacksjo01	0.355752
4	meyerle01	0.355509

QUESTION – TASK 3

Pick any two baseball teams and compare them using a metric of your choice (average salary, home runs, number of wins, etc). Write an SQL query to get the data you need, then make a graph using Lets-Plot to visualize the comparison. What do you learn?

▶ Show the code

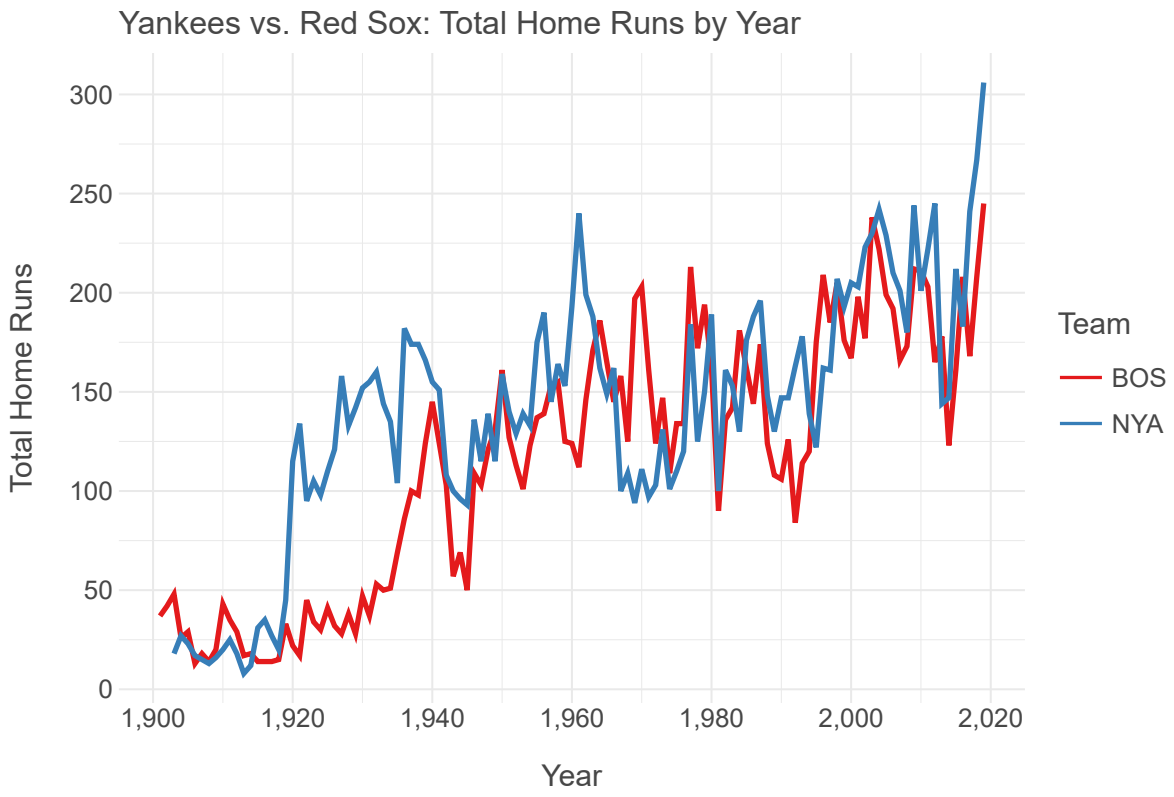
	yearID	teamID	total_HR
0	1901	BOS	37

	yearID	teamID	total_HR
1	1902	BOS	42
2	1903	BOS	48
3	1903	NYA	18
4	1904	BOS	26
...	...	...	...
231	2017	NYA	241
232	2018	BOS	208
233	2018	NYA	267
234	2019	BOS	245
235	2019	NYA	306

236 rows × 3 columns

▼ Show the code

```
# Visualization of Yankees vs. Red Sox Home Runs Over Time
ggplot(df_hr, aes(x='yearID', y='total_HR', color='teamID')) + \
    geom_line(size=1.2) + \
    ggtitle("Yankees vs. Red Sox: Total Home Runs by Year") + \
    labs(x="Year", y="Total Home Runs", color="Team") + \
    theme_minimal()
```



Based on the graph, we observe fluctuations in home run totals over time for both teams. In recent years, the Yankees have consistently hit more home runs compared to the Red Sox, especially post-2010. This could

indicate a shift in hitting strategy, roster strength, or ballpark influence.

STRETCH QUESTION – TASK 1

Advanced Salary Distribution by Position (with Case Statement):

▼ Show the code

```
query = '''
WITH main_pos AS (
  SELECT playerID, yearID, Pos, COUNT(*) AS games
  FROM Fielding
  GROUP BY playerID, yearID, Pos
),
most_played AS (
  SELECT playerID, yearID, Pos
  FROM (
    SELECT *, ROW_NUMBER() OVER (PARTITION BY playerID, yearID ORDER BY games DESC) AS rn
    FROM main_pos
  )
  WHERE rn = 1
)
SELECT mp.Pos AS position,
       ROUND(AVG(s.salary), 2) AS average_salary,
       COUNT(DISTINCT s.playerID) AS total_players,
       MAX(s.salary) AS highest_salary,
       CASE
         WHEN AVG(s.salary) > 3000000 THEN 'High Salary'
         WHEN AVG(s.salary) BETWEEN 2000000 AND 3000000 THEN 'Medium Salary'
         ELSE 'Low Salary'
       END AS salary_category
FROM most_played mp
JOIN Salaries s ON mp.playerID = s.playerID AND mp.yearID = s.yearID
GROUP BY mp.Pos
ORDER BY average_salary DESC;
'''
df_salary_pos = pd.read_sql_query(query, con)
df_salary_pos
```

	position	average_salary	total_players	highest_salary	salary_category
0	3B	2954624.15	359	33000000.0	Medium Salary
1	SS	2932867.67	181	22600000.0	Medium Salary
2	OF	2595167.98	933	27328046.0	Medium Salary
3	1B	2392855.86	1049	28000000.0	Medium Salary
4	P	1939697.14	2558	33000000.0	Low Salary

	position	average_salary	total_players	highest_salary	salary_category
5	2B	1463198.38	672	24000000.0	Low Salary
6	C	1340954.00	364	16071429.0	Low Salary

STRETCH QUESTION – TASK 2

Advanced Career Longevity and Performance (with Subqueries):

▼ Show the code

```
query = '''
WITH player_years AS (
  SELECT playerID, MIN(yearID) AS start_year, MAX(yearID) AS end_year, COUNT(DISTINCT yearID) AS
         career_length
  FROM Batting
  GROUP BY playerID
  HAVING SUM(G) >= 10
)
SELECT py.playerID, p.nameFirst, p.nameLast, py.career_length
FROM player_years py
JOIN people p ON py.playerID = p.playerID
ORDER BY py.career_length DESC
LIMIT 10;
'''
df_longevity = pd.read_sql_query(query, con)
df_longevity
```

	playerID	nameFirst	nameLast	career_length
0	ansonca01	Cap	Anson	27
1	ryanno01	Nolan	Ryan	27
2	johnto01	Tommy	John	26
3	mcguide01	Deacon	McGuire	26
4	collied01	Eddie	Collins	25
5	henderi01	Rickey	Henderson	25
6	houghch01	Charlie	Hough	25
7	kaatji01	Jim	Kaat	25
8	moyerja01	Jamie	Moyer	25
9	wallabo01	Bobby	Wallace	25