cse250_project_3_Jacob_Padgett.md

Project 3 - Jacob Padgett

Elevator pitch

This project comes as a result our client seeking data from the history of baseball. I connect to the provided database and answer questions presented. Finally, there is a chart requested, which I provide a comparison of the Arizona Diamondbacks vs the San Die Padres and their runs scored from doubles.

TECHNICAL DETAILS

Grand Question 1

	playerid	schoolid	salary	yearid	teamid
0	lindsma01	idbyuid	4000000	2014	CHA
1	lindsma01	idbyuid	4000000	2014	CHA
2	lindsma01	idbyuid	3600000	2012	BAL

Grand Question 2

Part 1

	playerid	yearid	batting_average
0	mccafsp01	1889	1
1	snowch01	1874	1
2	oconnfr01	1893	1

Part 2

	playerid	yearid	batting_average
0	snowch01	1874	1
1	baldwki01	1884	1
2	oconnfr01	1893	1

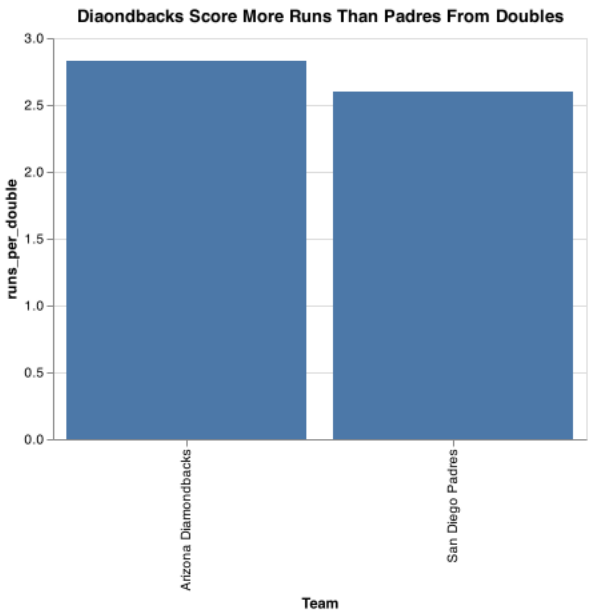
Part 3

	namefirst	namelast	hits	at_bat	bat_avg
0	Bob	Hazle	54	134	0.402985
1	Curt	Davis	40	105	0.380952
2	Showboat	Fisher	95	254	0.374016

Grand Question 3

	Team	teamid	at_bat	doubles	runs_scored	doubles_by_at_bat	runs_per_double
0	Arizona Diamondbacks	ARI	5491	235	665	0.0427973	2.82979
1	San Diego Padres	SDN	5357	180	468	0.0336009	2.6

Chart



Code Example

This is an example of how I did the work

```
result = dw.query('byuidss/cse-250-baseball-database',
    '''
    SELECT
    p.playerid
    , c.schoolid
    , s.salary
    , s.yearid
    , s.teamid
    FROM people p
    JOIN collegeplaying c ON p.playerid = c.playerid
    JOIN salaries s on c.playerid = s.playerid
    WHERE c.schoolid = 'idbyuid'
    ORDER BY s.salary DESC
    ;
    ''')

gq1 = result.dataframe
print(gq1.head(3).to_markdown())
gq1
```

GRAND QUESTION 1

Write an SQL query to create a new dataframe about baseball players who attended BYU-Idaho. The new table should contai 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.

```
result = dw.query('byuidss/cse-250-baseball-database',
    '''
    SELECT
    p.playerid
    , c.schoolid
    , s.salary
    , s.yearid
    , s.teamid
    FROM people p
    JOIN collegeplaying c ON p.playerid = c.playerid
    JOIN salaries s on c.playerid = s.playerid
    WHERE c.schoolid = 'idbyuid'
    ORDER BY s.salary DESC
    ;
    ''')
```

As can be seen, here is my result:

	playerid	schoolid	salary	yearid	teamid
0	lindsma01	idbyuid	4000000	2014	CHA
1	lindsma01	idbyuid	4000000	2014	CHA
2	lindsma01	idbyuid	3600000	2012	BAL

GRAND QUESTION 2

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

Part 1

- Write an SQL query that provides playerID, yearID, and batting average for players with at least one at bat. Sort the table from highest batting average to lowest, and show the top 5 results in your report.

```
result = dw.query('byuidss/cse-250-baseball-database',
    '''
    SELECT
    playerid
    , yearid
    , h/ab AS batting_average
    FROM batting
    WHERE h >= 1
    ORDER BY batting_average DESC
    LIMIT 5
    ;
    ''')

gq2a = result.dataframe
print(gq2a.head(3).to_markdown())
gq2a
```

	playerid	yearid	batting_average
0	mccafsp01	1889	1
1	snowch01	1874	1

	playerid	yearid	batting_average
2	oconnfr01	1893	1

Part 2

- Write an SQL query that provides playerID, yearID, and batting average for players with at least one at bat. Sort the table from highest batting average to lowest, and show the top 5 results in your report.

```
result = dw.query('byuidss/cse-250-baseball-database',
    '''
    SELECT
    playerid
    , yearid
    , h/ab AS batting_average
    FROM batting
    WHERE ab >= 1
    ORDER BY batting_average DESC
    LIMIT 5
    ;
    ''')

gq2b = result.dataframe
print(gq2b.head(3).to_markdown())
gq2b
```

	playerid	yearid	batting_average
0	snowch01	1874	1
1	baldwki01	1884	1
2	oconnfr01	1893	1

Part 3

- Now calculate the batting average for players over their entire careers (all years combined). Only include players with more th 100 at bats, and print the top 5 results.

```
result = dw.query('byuidss/cse-250-baseball-database',
    '''
    SELECT p.namefirst, p.namelast, SUM(b.h) as hits, SUM(b.ab) AS at_bat, SUM(b.h)/SUM(b.ab) AS bat_avg
    FROM people p
    JOIN batting b ON p.playerid = b.playerid
    WHERE b.ab > 100
    GROUP BY p.playerid
    ORDER BY bat_avg DESC
    LIMIT 5
    ;
   ''')

gq2c = result.dataframe
print(gq2c.head(3).to_markdown())
gq2c
```

	namefirst	namelast	hits	at_bat	bat_avg
0	Bob	Hazle	54	134	0.402985
1	Curt	Davis	40	105	0.380952

	namefirst	namelast	hits	at_bat	bat_avg
2	Showboat	Fisher	95	254	0.374016

GRAND QUESTION 3

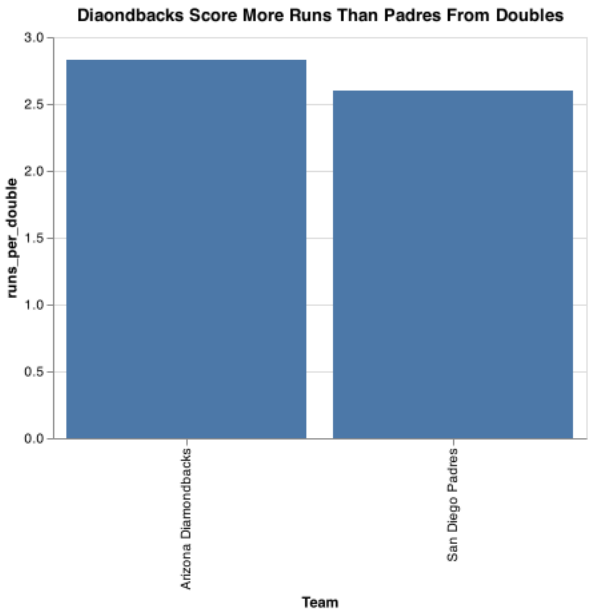
Pick any two baseball teams and compare them using a metric of your choice (average salary, home runs, number of wins, et Write an SQL query to get the data you need. Use Python if additional data wrangling is needed, then make a graph in Altair t visualize the comparison. Provide the visualization and its description.

```
    '''
    -- At Bat vs Triples with Runs Scored

    SELECT    f.franchname AS Team
            , t.teamid
            , t.ab AS at_bat
            , t.2b AS doubles
            , t.r AS runs_scored
            , t.2b/t.ab AS doubles_by_at_bat
            , t.r/t.2b AS runs_per_double
    FROM teams t
    JOIN teamsfranchises f ON t.franchid = f.franchid
    WHERE f.active = "Y" AND t.teamid = "ARI" OR t.teamid = "SDN"
    GROUP BY f.franchname
    ORDER BY runs_scored DESC
    ;
   ''')

gq3 = result.dataframe
print(gq3.head(3).to_markdown())
gq3
```

	Team	teamid	at_bat	doubles	runs_scored	doubles_by_at_bat	runs_per_double
0	Arizona Diamondbacks	ARI	5491	235	665	0.0427973	2.82979
1	San Diego Padres	SDN	5357	180	468	0.0336009	2.6

Chart For GQ3

Arizona scores more than San Diego per each double hit.



APPENDIX A (PYTHON SCRIPT)

```
# %%
import datadotworld as dw
import pandas as pd
import altair as alt

# import sys
# !{sys.executable} -m pip install tabulate
# %%
# Grand Question 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 ea
salary. Order the table by salary (highest to lowest) and print out the table in your report.

result = dw.query('byuidss/cse-250-baseball-database',
    '''
    SELECT
    p.playerid
    , c.schoolid
    , s.salary
    , s.yearid
    , s.teamid
    FROM people p
    JOIN collegeplaying c ON p.playerid = c.playerid
    JOIN salaries s on c.playerid = s.playerid
    WHERE c.schoolid = 'idbyuid'
    ORDER BY s.salary DESC
    ;
    ''')

gq1 = result.dataframe
print(gq1.head(3).to_markdown())
gq1

# %%
# Grand Question 2
#
# This three-part question requires you to calculate batting average (number of hits divided by the number of at-
```

```
bats)
#
# Write an SQL query that provides playerID, yearID, and batting average for players with at least one at bat. So
the table from highest batting average to lowest, and show the top 5 results in your report.
#
# Write an SQL query that provides playerID, yearID, and batting average for players with at least one at bat. So
the table from highest batting average to lowest, and show the top 5 results in your report.
#
# Now calculate the batting average for players over their entire careers (all years combined). Only include
players with more than 100 at bats, and print the top 5 results.

# %%
# Part 1
# Write an SQL query that provides playerID, yearID, and batting average for players with at least one at bat. So
the table from highest batting average to lowest, and show the top 5 results in your report.

result = dw.query('byuidss/cse-250-baseball-database',
    '''
    SELECT
    playerid
    , yearid
    , h/ab AS batting_average
    FROM batting
    WHERE h >= 1
    ORDER BY batting_average DESC
    LIMIT 5
    ;
    ''')

gq2a = result.dataframe
print(gq2a.head(3).to_markdown())
gq2a

# %%
# Part 2
# Write an SQL query that provides playerID, yearID, and batting average for players with at least one at bat. So
the table from highest batting average to lowest, and show the top 5 results in your report.

result = dw.query('byuidss/cse-250-baseball-database',
    '''
    SELECT
    playerid
    , yearid
    , h/ab AS batting_average
    FROM batting
    WHERE ab >= 1
    ORDER BY batting_average DESC
    LIMIT 5
    ;
    ''')

gq2b = result.dataframe
print(gq2b.head(3).to_markdown())
gq2b

# %%
# Part 3
# Now calculate the batting average for players over their entire careers (all years combined). Only include
players with more than 100 at bats, and print the top 5 results.

result = dw.query('byuidss/cse-250-baseball-database',
    '''
    SELECT p.namefirst, p.namelast, SUM(b.h) as hits, SUM(b.ab) AS at_bat, SUM(b.h)/SUM(b.ab) AS bat_avg
    FROM people p
    JOIN batting b ON p.playerid = b.playerid
    WHERE b.ab > 100
    GROUP BY p.playerid
    ORDER BY bat_avg DESC
```

```
    LIMIT 5
    ;
   ''')

gq2c = result.dataframe
print(gq2c.head(3).to_markdown())
gq2c

# %%
# Grand Question 3
#
# Pick any two baseball teams and compare them using a metric of your choice (average salary, home runs, number o
wins, etc.). Write an SQL query to get the data you need. Use Python if additional data wrangling is needed, then
make a graph in Altair to visualize the comparison. Provide the visualization and its description.

result = dw.query('byuidss/cse-250-baseball-database',
    '''
    -- At Bat vs Triples with Runs Scored

    SELECT    f.franchname AS Team
            , t.teamid
            , t.ab AS at_bat
            , t.2b AS doubles
            , t.r AS runs_scored
            , t.2b/t.ab AS doubles_by_at_bat
            , t.r/t.2b AS runs_per_double
    FROM teams t
    JOIN teamsfranchises f ON t.franchid = f.franchid
    WHERE f.active = "Y" AND t.teamid = "ARI" OR t.teamid = "SDN"
    GROUP BY f.franchname
    ORDER BY runs_scored DESC
    ;
   ''')

gq3 = result.dataframe
print(gq3.head(3).to_markdown())
gq3

# %%
#
# Diamondbacks score more runs from doubles than Padres

# Chart
gq3_chart = (
    alt.Chart(gq3)
    .encode(x="Team", y="runs_per_double")
    .mark_bar()
    .properties(width=400, title="Diaondbacks Score More Runs Than Padres From Doubles")
)


gq3_chart.save('gq3_chart.png')
gq3_chart
```