Milestone 5 | NBA Statistics

INTRODUCTION: Stats for players and teams have long been a part of professional sports, but since the 2000s, data analytics has become an increasingly important part of developing and running a successful sports team. This revolution in data has also resulted in new ways of measuring what it means for a player or team to be effective.

If you're feeling a little rusty on the details of professional basketball, here's how it's played:

A game of basketball is played between two teams, each with five players. The objective is to score more points than the opposing team by shooting a ball through a hoop/basket. Players can score for their team in a variety of ways – point values are assigned to the location of the shot.

A basket made from inside the "three-point line" is worth two points, while a shot made from beyond the line is worth three points. "Free throws" can also be awarded to a player or a team when the opposing team commits a foul or breaks a rule. These are worth one point each.

HOW IT WORKS: Follow the prompts in the questions below to investigate your data. Post your answers in the provided boxes: the **yellow boxes** for the queries you write and **blue boxes** for text-based answers. When you're done, export your document as a pdf file and submit it on the Milestone page – see instructions for creating a PDF at the end of the Milestone.

RESOURCES: If you need hints on the Milestone or are feeling stuck, there are multiple ways of getting help. Attend Drop-In Hours to work on these problems with your peers, or reach out to the HelpHub if you have questions. Good luck!

PROMPT: In this Milestone, you'll be looking at the way that professional basketball in the NBA has changed over seventeen recent seasons. If you were a coach in the league, what could you say about how the game is being played, and what are the most successful teams doing to be successful?

SQL App: Here's that link to our specialized SQL app, where you'll write your SQL queries and interact with the data.

— Data Set **Description**

The NBA games dataset (`nba.games`) contains information about 23 335 games played from the 2004 season through the 2020 season. There are eighteen columns in the dataset, of which the following will be used in the Milestone:

- **`season`** - Starting year for the season the game was played. For example, games that are part of the 2010–11 season will have a season value of 2010, even if they are played in 2021.
- **`team_home`**, **`team_away`** - Full name of the home and visiting teams, respectively. Names will always reflect their current franchise names, even if they were known by a different name in prior years.
- **`pts_home`**, **`pts_away`** - Number of points scored by the home and visiting teams, respectively, in each game.
- **`home_team_win`** - Flag indicating whether the home team won (1) or the visiting team won (0).
- **`pct_3p_home, pct_3p_away`** - Percentage of 3 point shots made by the home team and away team, respectively.

– **Task 1:** Game Statistics Trends Over Time

A. Start by calculating the total number of rows and the first & last seasons in the dataset. This should be done in one query. If done correctly, the number of games is 23 335, the first season represented is 2004, and the last season represented is 2020.

```
SELECT
  --season, team_away,team_home, pts_away, pts_home,
home_team_win, pct_3p_away, pct_3p_home
  count(*)
FROM
  nba.games;
SELECT
  DISTINCT season
FROM
  nba.games
order by
  season DESC
```

B. Write a query that returns the average score from the home team, away team, and the average of the home_team_win column. The average of the home_team_win column can be interpreted as the win rate for the home team. What do these values tell you about what you can expect from the result of a random NBA game?

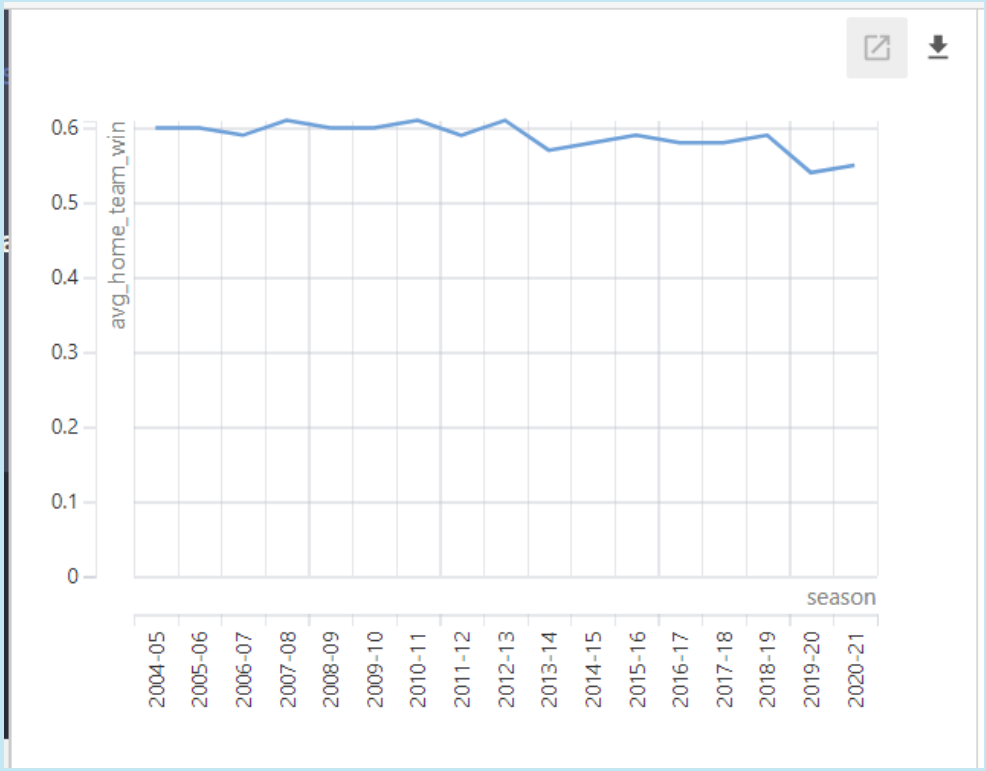
```
SELECT
  --season, team_away,team_home, pts_away, pts_home,
home_team_win, pct_3p_away, pct_3p_home
  DISTINCT team_home,
  ROUND(AVG(pts_home), 0) AS avg_points_home,
  ROUND(AVG(pts_away), 0) AS avg_points_away,
  ROUND(AVG(home_team_win), 2) AS avg_home_team_win
FROM
  nba.games
```

```
group by
  team_home
```

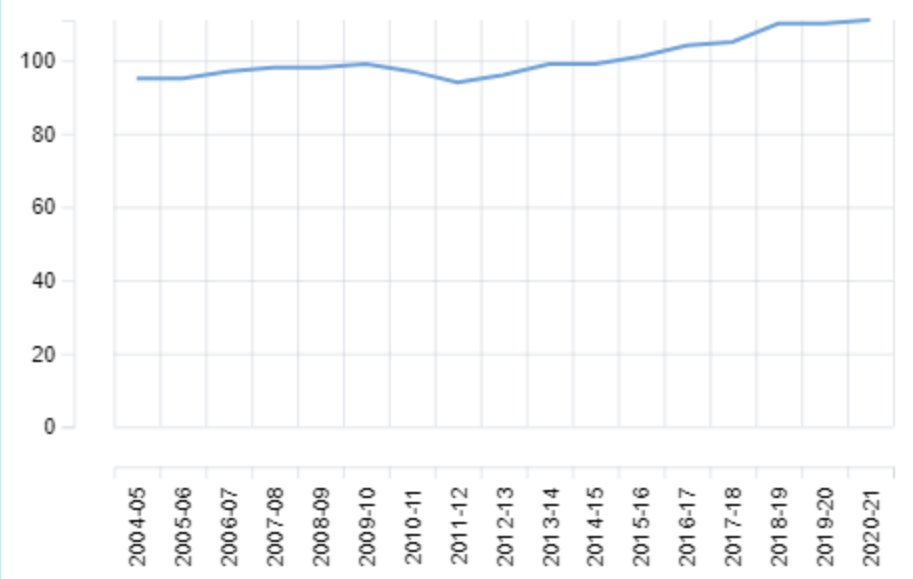
The results of this query suggest that if a team is playing home, from past results, they have a higher likelihood of winning on average.

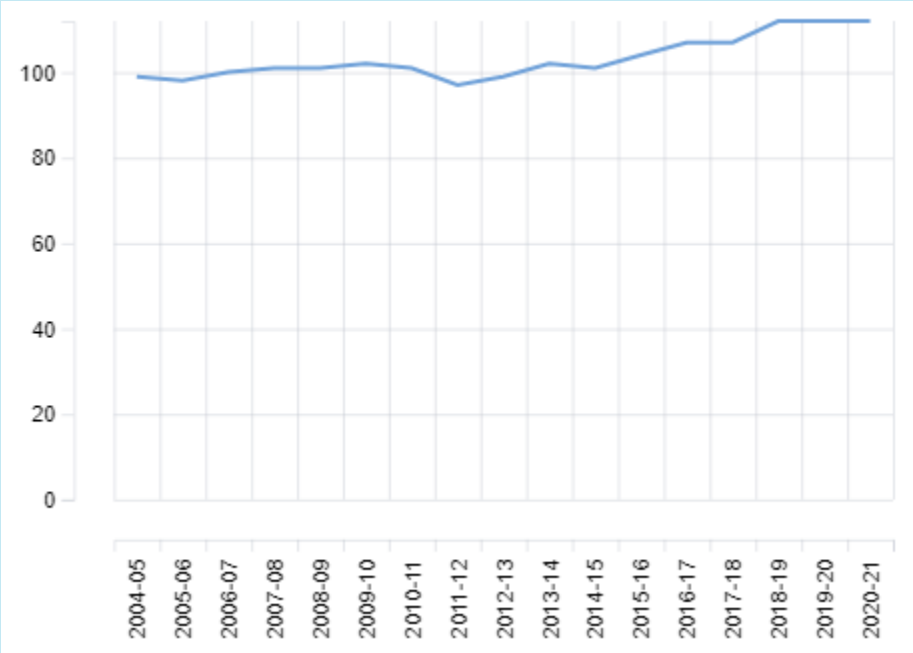
C. Modify your query from part B, so that the average scores from the home team, away team, and the home team win rate are grouped by each NBA season. Sort your output so that the seasons are ordered chronologically. What can you say about the trend in these values over the years?

```
SELECT
  --season, team_away,team_home, pts_away, pts_home,
home_team_win, pct_3p_away, pct_3p_home
  DISTINCT season,
  ROUND(AVG(pts_home), 0) AS avg_points_home,
  ROUND(AVG(pts_away), 0) AS avg_points_away,
  ROUND(AVG(home_team_win), 2) AS avg_home_team_win
FROM
  nba.games
group by
  season
ORDER BY
  season ASC
```



The average win rate of a home game win has gone down by 5% from 2004 to 2020.





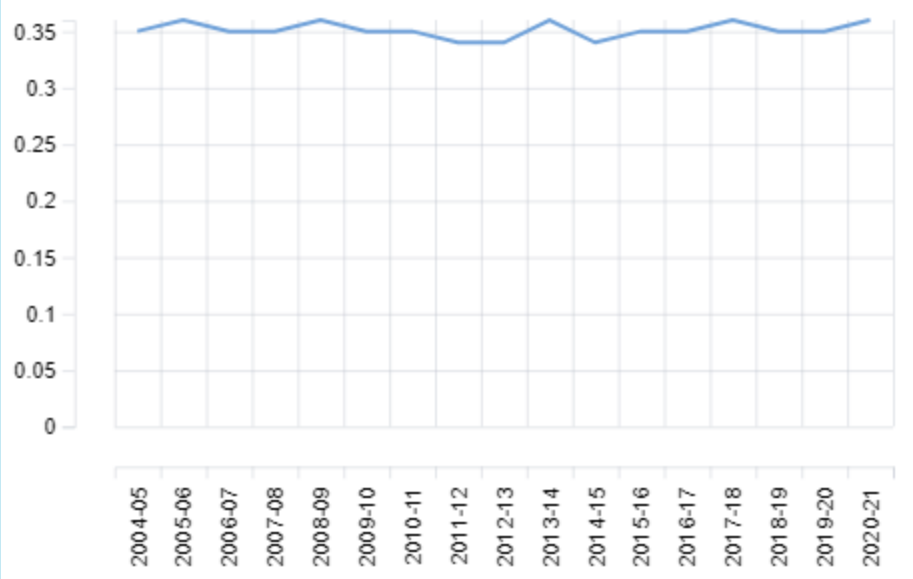
Both points avg points home and away are seeing slow and steady increases in percentages.

D. Add two more summaries to your query from part C, to get the average 3–point shot rate for both away and home teams. Do these values change over time?

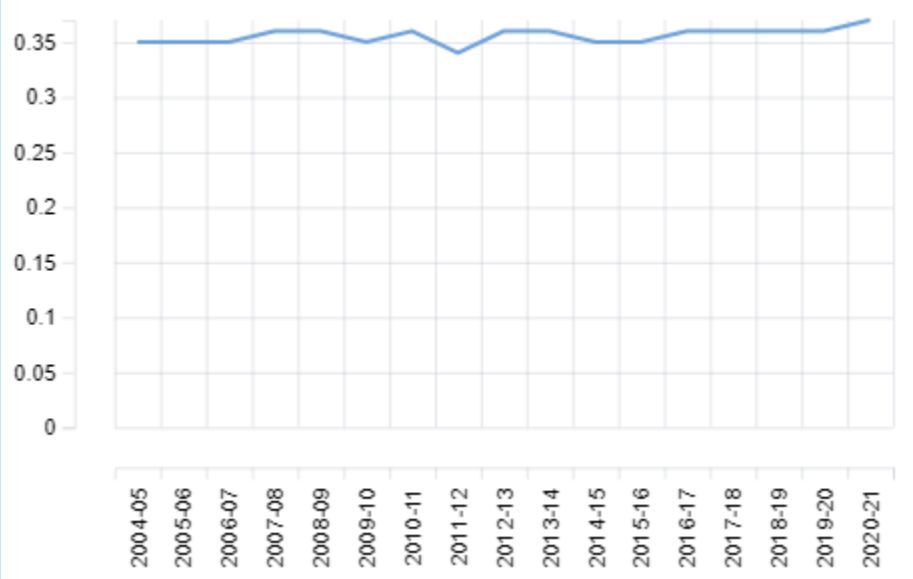
```
SELECT
  --season, team_away,team_home, pts_away, pts_home,
home_team_win, pct_3p_away, pct_3p_home
  DISTINCT season,
  ROUND(AVG(pts_home), 0) AS avg_points_home,
  ROUND(AVG(pts_away), 0) AS avg_points_away,
  ROUND(AVG(home_team_win), 2) AS avg_home_team_win,
  ROUND(AVG(pct_3p_away), 2) AS avg_3p_away,
  ROUND(AVG(pct_3p_home), 2) AS avg_3p_home
FROM
  nba.games
group by
  season
ORDER BY
  season ASC
```

So while the values change over time, the difference is almost unnoticeable unless seen through a graph.

The first graph is away results with a 1% average increase from 2004 to 2020 in 3pt shots made from away teams.



The second graph show results with a 2% average increase from 2004 to 2020 in 3pt shots made from home teams.



These percentages seem relatively small; however, this factor in the sport should not be ignored when games can be won by just a 1 point difference in the final scores of both competing teams.

— **Task 2:** Investigating 3-point Shooting

The average three-point shot rate is about 35.4% over the entire dataset. Let's write some queries to investigate just how important a high three-point shot rate is in terms of winning games.

A. Write a query that returns the average home team win rate and average three-point percentage at home grouped by home team name and season. (You should get a table with 510 rows.)

```
SELECT
  --season, team_away,team_home, pts_away, pts_home,
home_team_win, pct_3p_away, pct_3p_home
  DISTINCT season,
  team_home,
  ROUND(AVG(home_team_win), 2) AS avg_home_team_win,
  ROUND(AVG(pct_3p_home), 2) AS avg_3p_home
FROM
  nba.games
group by
  season,
  team_home
ORDER BY
  season ASC,
  avg_home_team_win DESC,
  avg_3p_home DESC
```

B. Modify your query so we are only looking at results from 2018 or later. Remember, the season column is a text field - don't forget your quotes! (This should reduce your results down to 90 rows.)

```
SELECT
  --season, team_away,team_home, pts_away, pts_home,
home_team_win, pct_3p_away, pct_3p_home
  season,
```

```
  team_home,
  ROUND(AVG(home_team_win), 2) AS avg_home_team_win,
  ROUND(AVG(pct_3p_home), 2) AS avg_3p_home
FROM
  nba.games
WHERE
  season LIKE '%2018%'
  or season LIKE '%2019%'
  or season LIKE '%2020%'
GROUP BY
  season,
  team_home
ORDER BY
  season ASC,
  avg_home_team_win DESC,
  avg_3p_home DESC

--I'm not a fan of how this code turned out. It feels messy
and unreadable, but I don't know a better way.
```

C. Add another expression to your query to answer the following question: How many teams had a three-point shot rate of at least 37% (i.e. 0.37)? (You'll get this from the output of the SQL app interface, rather than directly from the query.)

```
SELECT
  --season, team_away,team_home, pts_away, pts_home,
home_team_win, pct_3p_away, pct_3p_home
  season,
  team_home,
  ROUND(AVG(home_team_win), 2) AS avg_home_team_win,
  ROUND(AVG(pct_3p_home), 2) AS avg_3p_home
FROM
  nba.games
WHERE
  season LIKE '%2018%'
```

```
  or season LIKE '%2020%'
  or season LIKE '%2019%'
GROUP BY
  season,
  team_home
HAVING ROUND(AVG(pct_3p_home),2) >= .37
ORDER BY
  season ASC,
  avg_3p_home ASC
```

35 teams had a three point shot rate of 37% or higher.

D. Add an additional condition to your query to filter to teams with a losing record (win rate < 0.5), in addition to having a high three-point shot rate. (As with the previous part, you'll read this from the SQL app interface instead of directly from the query.)

```
SELECT
  --season, team_away,team_home, pts_away, pts_home,
home_team_win, pct_3p_away, pct_3p_home
  season,
  team_home,
  ROUND(AVG(home_team_win), 2) AS avg_home_team_win,
  ROUND(AVG(pct_3p_home), 2) AS avg_3p_home
FROM
  nba.games
WHERE
  season LIKE '%2018%'
  or season LIKE '%2020%'
  or season LIKE '%2019%'
GROUP BY
  season,
  team_home
```

```
HAVING ROUND(AVG(pct_3p_home),2) >= .37 AND
ROUND(AVG(home_team_win),2) < 0.50
ORDER BY
  season ASC,
  avg_3p_home ASC
```

There are only 7 teams that have a loss rate of less than .5 AND who also have a 3pt average of 37% or higher which have played in the year 2018 or so forth.

E. Repeat parts C and D, but this time filtering to teams that had a low 3-point shooting rate of 34% (0.34) or less. How many teams had this low of a 3-point accuracy, and how many of these teams had a losing record? (Paste only the query that answers the last question into the query box.)

```
SELECT
  --season, team_away,team_home, pts_away, pts_home,
home_team_win, pct_3p_away, pct_3p_home
  season,
  team_home,
  ROUND(AVG(home_team_win), 2) AS avg_home_team_win,
  ROUND(AVG(pct_3p_home), 2) AS avg_3p_home
FROM
  nba.games
WHERE
  season LIKE '%2018%'
  or season LIKE '%2020%'
  or season LIKE '%2019%'
GROUP BY
  season,
  team_home
HAVING ROUND(AVG(pct_3p_home),2) <= .34
ORDER BY
```

```
  season ASC,
  avg_3p_home ASC

---------------

SELECT
  --season, team_away,team_home, pts_away, pts_home,
home_team_win, pct_3p_away, pct_3p_home
  season,
  team_home,
  ROUND(AVG(home_team_win), 2) AS avg_home_team_win,
  ROUND(AVG(pct_3p_home), 2) AS avg_3p_home
FROM
  nba.games
WHERE
  season LIKE '%2018%'
  or season LIKE '%2020%'
  or season LIKE '%2019%'
GROUP BY
  season,
  team_home
HAVING ROUND(AVG(pct_3p_home),2) <= .34 AND
ROUND(AVG(home_team_win), 2) < 0.50
ORDER BY
  season ASC,
  avg_3p_home ASC
```

22 teams had a “low” 3 pt average of 34% or lower, and of those 22 teams, 13 teams had an average losing record.

— **Level Up:** Building the Team Stats table

Most of the time when working with data, you will have to build summary tables yourself. This is done to alleviate storage costs, especially since these tables have

to constantly be updated via a SQL query. In this Level Up you'll create two summary tables, one for the home team and one for the away team. Although you don't have the ability to join these tables (yet!), you can come back to this LevelUp and create the full table once you have learned how to join data.

A. Write a query that returns the average number of home points scored, average 3 point percentage for the home team, and the number of wins for each team and season combination in the `nba.games` table. (You should get a table with 510 rows.)

```
SELECT
  --season, team_away,team_home, pts_away, pts_home,
home_team_win, pct_3p_away, pct_3p_home
  DISTINCT team_home,
  season,
  ROUND(AVG(pts_home), 2) AS avg_pts_home,
  ROUND(AVG(pct_3p_home), 2) AS avg_3p_home,
  count(home_team_win) AS n_home_team_wins
FROM
  nba.games
GROUP BY
  team_home,
  season
order by
  season,
  team_home
```

B. Repeat part A but now do the same thing for the **away** team. Note that you will have to get a little creative to calculate the number of away wins since the table only tells you whether or not the home team won the game. (You should get a table with 510 rows.)

```
SELECT
```

```
  --season, team_away,team_home, pts_away, pts_home,
home_team_win, pct_3p_away, pct_3p_home
  DISTINCT team_away,
  season,
  ROUND(AVG(pts_away), 2) AS avg_pts_away,
  ROUND(AVG(pct_3p_away), 2) AS avg_3p_away,
  count(home_team_win) AS n_home_wins
FROM
  nba.games
GROUP BY
  team_away,
  season
order by
  season,
  team_away;
SELECT
  team,
  season,
  wins
FROM
  nba.team_stats
GROUP BY
  team,
  season,
  wins
ORDER BY
  season,
  team

--This answer I'm not proud of because while I do get 510
rows for the first query, the second query only prints out
300 rows and I'm not sure why. I will go to study hours to
review this question because I want to know how to do this.
```

— Submission

Great work completing this Milestone! To submit your completed Milestone, you will need to download / export this document as a PDF and then upload it to the Milestone submission page. You can find the option to download as a PDF from the File menu in the upper-left corner of the Google Doc interface.