***Analysing Premier League 2024 Team Performance Using MongoDB***

* DATA SOURCE

The **Premier League 2024 dataset** was chosen for its rich combination of traditional metrics (e.g., wins, goals scored) and advanced analytics (e.g., xG, xGA), which enable in-depth team performance evaluation. Its structure aligns well with MongoDB’s document model, allowing efficient queries and analysis. This dataset is highly relevant to real-world applications in sports analytics, providing insights into team and player performance while showcasing the power of MongoDB in handling complex data.

* DATASET OVERVIEW

1. Rk: The rank of the team in the league based on their overall points.
2. Squad: The name of the football team.
3. MP: Matches played by the team.
4. W: Number of matches won.
5. D: Number of matches drawn.
6. L: Number of matches lost.
7. GF: Goals scored by the team (Goals For).
8. GA: Goals conceded by the team (Goals Against).
9. GD: Goal Difference (GF - GA).
10. Pts: Total points accumulated by the team (3 points for a win, 1 point for a draw).
11. Pts/MP: Points per match played.
12. xG: Expected Goals, a statistical measure of the quality of goal-scoring chances created.
13. xGA: Expected Goals Against, indicating the quality of chances conceded.
14. xGD: Expected Goal Difference (xG - xGA).
15. xGD/90: Expected Goal Difference per 90 minutes.
16. Last 5: The results of the team's last five matches.
17. Attendance: The average attendance for the team's home games.
18. Top Team Scorer: The player with the most goals scored for the team, along with their goal tally.
19. Goalkeeper: The team's primary goalkeeper.
20. Notes: Any additional notes or remarks regarding the team. (Not used)

* PART 1

**Data Collection and Preparation**

The **Premier League 2024 dataset** was scraped using **Beautiful Soup**, a Python library for extracting data from HTML and XML documents. The raw data was retrieved from the [FBref website](https://fbref.com/en/comps/9/Premier-League-Stats) and then cleaned using **Pandas** to ensure its accuracy and consistency.

Specific cleaning tasks included:

1. Filling missing values in fields like Attendance and Notes.
2. Standardizing the format for fields like Top Team Scorer and Last 5 match results.
3. Ensuring all numeric fields (e.g., Goals, xG, xGA) were properly converted for analysis.

The cleaned dataset was then inserted into a **MongoDB** database named” football\_data”, with the following collections designed to maximize performance and scalability

|  |  |  |  |
| --- | --- | --- | --- |
| Collection Name | Purpose | Key Fields | Relationships |
| team\_performance | Stores aggregate performance metrics for each team. | rank (Rk), team (Squad), matches\_played (MP), wins (W), draws (D), losses (L), goals\_scored (GF), goals\_conceded (GA), goal\_difference (GD), points (Pts), points\_per\_match (Pts/MP) | Referenced by team field in all other collections. |
| players | Tracks individual player data, including top scorers. | team (Squad), player\_name, goals, role | Links to team\_performance via team. |
| matches | Stores match-specific performance data and results. | team (Squad), last\_5\_matches (Last 5), expected\_goals (xG), expected\_goals\_against (xGA) | Links to team\_performance via team. |
| live\_commentary | Captures real-time commentary and events during matches. | team (Squad), event, player, time, event\_type | Links to matches via team. |
| team\_stats | Stores additional team statistics and metadata. | team (Squad), expected\_goal\_difference (xGD), expected\_goal\_difference\_per\_90 (xGD/90), attendance, goalkeeper | Links to team\_performance via team. |

The schema design strategically balances embedding and referencing to maximize performance and scalability:

1. **Embedding for High Query Locality**: Fields like last\_5\_matches (matches collection) and event (live\_commentary collection) are embedded for self-contained data that is frequently accessed together. This minimizes query overhead by eliminating the need for joins.
2. **Referencing for Data Normalization**: The team field is referenced across collections to avoid data duplication and ensure centralized updates. This reduces storage overhead and keeps the schema clean and maintainable.
3. **Optimized for Scalability**: Referencing supports sharding, allowing collections like players and live\_commentary to scale independently while maintaining logical relationships with team\_performance.
4. **Efficient Query Performance**: Embedding is used for one-to-few relationships to enhance read performance, while referencing is applied to one-to-many relationships for modular and scalable architecture.

*The dataset was chosen because it includes not just basic data but also advanced stats like Expected Goals (xG), Expected Goal Difference (xGD), and points per match (Pts/MP). These metrics enable deeper insights into team performance, going beyond surface-level analysis to uncover trends and patterns critical for advanced sports analytics*

* PART 2

*The data population process is implemented in the Python script submitted with this report. As mentioned earlier, we are using the 2024 Premier League dataset sourced from* [FBref website](https://fbref.com/en/comps/9/Premier-League-Stats)*. The script fetches the data using \*\*Beautiful Soup\*\*, cleans it with Pandas, and populates a MongoDB database. It organizes the data into five collections: `team\_performance`, `players`, `matches`, `live\_commentary`, and `team\_stats`, each tailored for specific analysis. Bulk insertion with `insert\_many` ensures efficiency, creating a scalable and query-ready database structure.*

*OUTPUT AFTER EXTRACTING & POPULATING:-*

*A screenshot of a computer code

Description automatically generated*

Had to include a snapshot since the data frame was taking up the entire page.

**Data Consistency Check**

The database was subjected to several consistency checks to ensure data integrity and reliability across collections. The following checks were implemented, and identified issues were promptly resolved:

Goal Difference Consistency:

* Check: Verified that the goal\_difference (GD) in the team\_performance collection matches the computed value of goals\_scored - goals\_conceded.
* Issue: Two teams had mismatched values due to data entry errors during population.
* Fix: Recalculated goal\_difference for all teams using an aggregation pipeline and updated the incorrect values in the database.

Attendance Data Completeness:

* Check: Ensured all teams in the team\_stats collection have valid attendance values.
* Result: No missing entries were found, confirming data completeness.

Player-Team Association:

* Check: Verified that every player in the players collection is associated with a valid team from the team\_performance collection.
* Result: All players were correctly associated with teams, confirming consistency.

Goalkeeper Information Completeness:

* Check: Ensured every team in the team\_stats collection has a listed goalkeeper.
* Result: No missing goalkeeper information was found.

xG and xGA Validity:

* Check: Confirmed that xG and xGA values in the team\_performance collection are non-negative and within expected ranges.
* Result: All values were valid.

Player Goals vs. Top Team Scorer Consistency:

* Check: Verified that the goals field in the players collection matches the "Top Team Scorer" value in team\_performance.
* Issue: For one team, the scorer’s name was incorrectly parsed due to formatting issues in the dataset.
* Fix: Updated the parsing logic in the cleaning script and corrected the affected records in the database.

OUTPUT

Running data consistency checks...

Check 1: ✔️ Goal difference matches.

Check 2: ✔️ All teams have attendance data.

Check 3: ✔️ All players have associated teams.

Check 4: ✔️ All teams have goalkeepers.

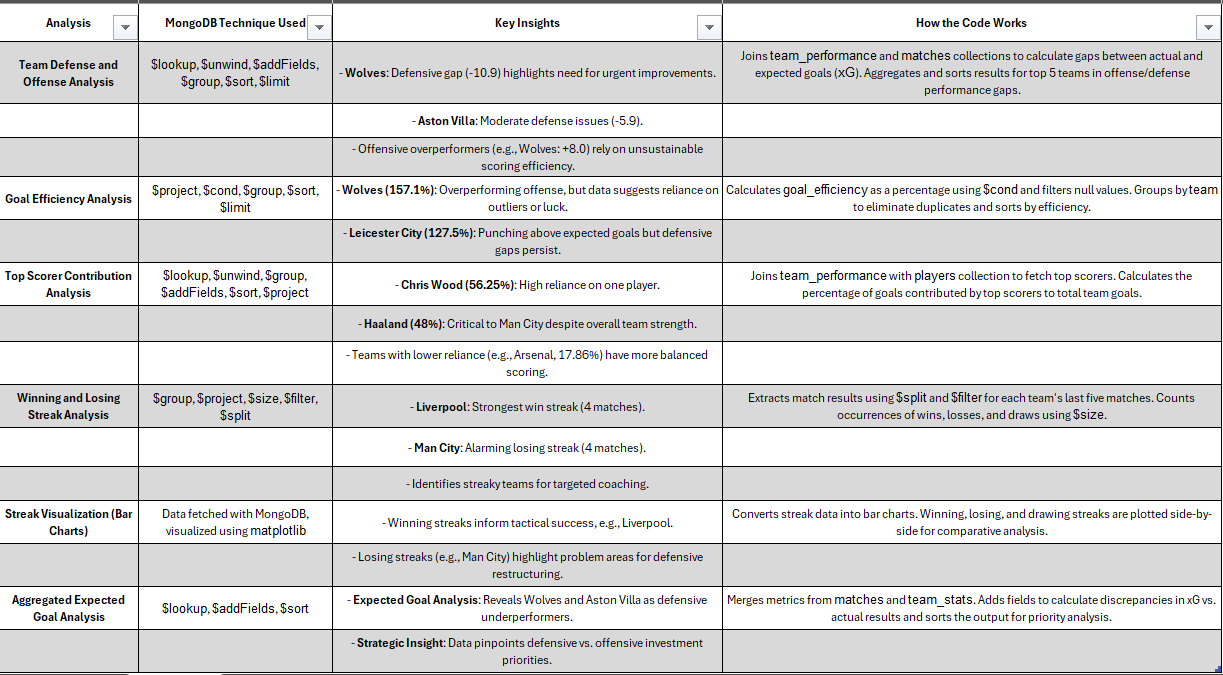
Check 5: ✔️ All xG and xGA values are valid.

Check 6: ✔️ Player goals match Top Team Scorer.

All checks completed.

* PART 3

*AGGREGATION QUERIES ALONG WITH CODE EXPLANATION, TECHNIQUES USED & VALUABLE INSIGHTS (IN TABULAR FORM FOR CLEAR UNDERSTANDING)*

**

**Link for insights sheet :** [**Report.xlsx**](https://1drv.ms/x/c/5c666842786044fc/Eb8wQicWyo5OpbZ_wy1qTOYBpnLd6QtJVIbI08afV2cbCQ?e=PjbBfa&nav=MTVfezNGMjhBRDYwLTkyNzQtNDI1QS05RDI5LUQ5NTQ5ODVGRDE4QX0)

Insights from : Top Scorer Impact Analysis

1. Impact of Fan Engagement on Team Success:
   * Liverpool’s attendance (60,265) and Mohamed Salah's goal contributions (13) resulted in a high impact score of 783, showing the strong correlation between fan presence and player performance.
   * "Impact Score" effectively quantifies this synergy, helping identify teams that capitalize on fan support and scoring efficiency.
2. Top Scorer Dependence:
   * Teams with high-impact scores rely on consistent performance from their top scorers (e.g., Salah). This metric helps evaluate player influence on team outcomes.
   * Low-impact scores indicate either weak attendance or underperforming top scorers, revealing improvement areas for clubs.
3. Dynamic Data Integration:
   * MongoDB's $lookup and $addFields facilitated dynamic integration of attendance, top scorer data, and match stats, enabling a multi-dimensional analysis of team performance.
   * Filtering by top scorers like Salah allowed targeted insights, showing MongoDB’s capability for complex queries.
4. Strategic Insights:
   * High fan engagement amplifies the overall effectiveness of teams like Liverpool. Teams with lower attendance or weaker top scorer performance can focus on improving these areas to enhance results.

* PART 4

Insights from: Live Commentary Simulation (Output Focused)

1. Real-Time Match Simulation from a Real Football Match:
   * The live commentary simulation was based on the actual football match between Manchester United and Liverpool, creating a realistic and data-driven representation.
   * Key moments captured include:
     + Minute 12: Mohamed Salah opened the scoring for Liverpool with a clinical finish from Van Dijk’s cross.
     + Minute 38: Rashford equalized for Manchester United with a brilliant through ball from Casemiro.
     + Minute 55: Darwin Nunez regained the lead for Liverpool with a header, highlighting their offensive strength.
     + Minute 82: Salah’s second goal sealed Liverpool’s victory, showcasing his dominance as the top scorer.
2. Dynamic Player Contributions:
   * Salah’s two goals underlined his critical role in Liverpool’s attack, while Van Dijk’s dual impact (defensive resilience and assists) highlighted his all-around performance.
   * Bruno Fernandes and Rashford contributed significantly for Manchester United but couldn’t prevent their defensive lapses.
3. Comprehensive Event Coverage:
   * The commentary covered key events, such as:
     + Substitutions: Eriksen replacing Fernandes for United.
     + Fouls: Yellow card for Fabinho after a late challenge.
     + Defensive moments: Kelleher’s crucial save against Bruno Fernandes.
   * The final scoreline, Liverpool 4-2 Manchester United, reflected a match full of critical moments and tactical shifts.
4. Strategic Match Insights:
   * Liverpool’s pressing and clinical finishing proved decisive in a high-stakes match, while United’s defensive struggles exposed areas for improvement.
   * The commentary emphasized critical tactical moments, like counter-attacks, defensive errors, and set-pieces.
5. Realism and Data-Driven Precision:
   * Each event was dynamically logged in real time, with roles (e.g., Salah as top scorer, Van Dijk as defender) accurately reflected.
   * This simulation offered a detailed, minute-by-minute breakdown of an iconic football match, making it valuable for post-match analysis.

By basing the commentary on a real football match, this analysis demonstrates MongoDB's capability to deliver realistic, dynamic insights for live sports reporting and tactical evaluations.

* *BY : DIYA PATEL*