# Data Engineering Task

The primary goal of this task is to assess your proficiency in both Python and SQL, particularly in data modeling and querying. Your task involves retrieving football (soccer) data from an API, storing it within a database of your preference, and crafting analytical queries for insights.

Kindly adhere to the following guidelines:

1. **API Access and Key Retrieval:**
    - Register an account on the apifootball site (opt for a temporary email for convenience) and secure the API key.
    - Familiarize yourself with the Events endpoint. For optimal results, make two separate requests to bypass API request limitations:
        - Request 1: Utilize dates from "2022-08-11" to "2022-11-11".
        - Request 2: Utilize dates from "2022-11-12" to "2023-05-29".
2. **Database Design:**
    - Create a compact dimensional model for the events data, comprising both dimensions and facts.
3. **SQL Queries.**
   Craft SQL queries that yield the following outputs:
    - **Final League Table:** Generate a query outputting the following columns: position, team_name, matches_played, won, draw, lost, goals_scored, goals_conceded, and points.
      Points allocation:
        - Victories: 3 points each
        - Draws: 1 point each
        - Defeats: 0 points
        - In case of points tie, utilize these tiebreakers:
            1. Superior goal difference
            2. Higher goals scored
            3. Fewer goals conceded
            4. More victories
    - **Display all teams sorted by Away goals scored**. The query must output the following columns: team_name, goals. Use team name in case of tie.
    - **Top 5 Referees with Most Cards:** Design a query listing referee_name and cards (sum of red and yellow cards). In case of a tie, arrange names in ascending order.

    - **Top 3 Goal Scorers by Match Round 14:** Develop a query providing player_name, team_name, and goals for the top three goal scorers up until match round 14. Sort by player name in case of a tie.
4. **Data Export:**
    - Export query results to CSV files: query_a.csv, query_b.csv, query_c.csv, and query_d.csv.
5. **Docker Container and App Deployment:**
    - Develop a Docker container to host your application.
6. **Readme File:**
    - Create a comprehensive readme file detailing your assumptions, choices, and instructions to execute your application.
7. **GitHub Repository:**
    - Upload your project to a GitHub repository and share the repository link with us for review.

By following these guidelines, you'll demonstrate your skills in working with APIs, databases, SQL querying, containerization, and documentation. This task provides an opportunity to showcase your technical prowess and problem-solving abilities. Feel free to reach out if you have any inquiries or require further assistance.
Good luck!