COMP 3005: Database Management Systems **(Due: Apr. 10, 2024 (11:59 PM))**

COMP3005 Project (Winter 2024)

Instructor: Ahmed El-Roby and Abdelghny Orogat

1. Problem Statement

Design a database that stores a soccer events dataset spanning multiple competitions and seasons. The provided dataset is in JSON format and can be downloaded from `https://github.com/statsbomb/open-data/tree/0067cae166a56aa80b2ef18f61e16158d6a7359a`[1]. The documentation of the dataset is also available in the above URL. After designing the database, you need to import the data from the JSON files into your database. You will be required to use PostgreSQL[2] to store and query your database. This project is automatically-graded. The auto-grader is available for you to try out your solution[3]. You are strongly encouraged to read the documentation of the auto-grader and test your work prior to submitting it to ensure smooth and correct grading of your work. The auto-grader connects to the database and execute the queries in the queries.py file. You will be required to insert your queries in the right place in that file (read the documentation).

The queries you are required to execute are:

1. Q_1: In the La Liga season of 2020/2021, sort the players from highest to lowest based on their average xG scores. **Output both the player names and their average xG scores.** Consider only the players who made at least one shot (the xG scores are greater than 0).

2. Q_2: In the La Liga season of 2020/2021, find the players with the most shots. Sort them from highest to lowest. **Output both the player names and the number of shots.** Consider only the players who made at least one shot (the lowest number of shots should be 1, not 0).

3. Q_3: In the La Liga seasons of 2020/2021, 2019/2020, and 2018/2019 combined, find the players with the most first-time shots. Sort them from highest to lowest. **Output the player names and the number of first time shots.** Consider only the players who made at least one shot (the lowest number of shots should be 1, not 0).

4. Q_4: In the La Liga season of 2020/2021, find the teams with the most passes made. Sort them from highest to lowest. **Output the team names and the number of passes.** Consider the teams that make at least one pass (the lowest number of passes is 1, not 0).

5. Q_5: In the Premier League season of 2003/2004, find the players who were the most intended recipients of passes. Sort them from highest to lowest. **Output the player names and the number of times they were the intended recipients of passes.** Consider the players who received at least one pass (the lowest number of times they were the intended recipients is 1, not 0).

6. Q_6: In the Premier League season of 2003/2004, find the teams with the most shots made. Sort them from highest to lowest. **Output the team names and the number of shots.** Consider the teams that made at least one shot (the lowest number of shots is 1, not 0).

7. Q_7: In the La Liga season of 2020/2021, find the players who made the most through balls. Sort them from highest to lowest. **Output the player names and the number of through balls.** Consider the players who made at least one through ball pass (the lowest number of through balls is 1, not 0).

8. Q_8: In the La Liga season of 2020/2021, find the teams that made the most through balls. Sort them from highest to lowest. **Output the team names and the number of through balls.** Consider the teams with at least one through ball made in a match (the lowest total number of through balls is 1, not 0).

[1] Please use this exact URL to download the dataset as it will be the same dataset used to verify the correctness of your answers to the given queries.

[2] `https://www.postgresql.org/`

[3] `https://github.com/gabrielmartell/COMP3305-Project-Template`

9. Q_9: In the La Liga seasons of 2020/2021, 2019/2020, and 2018/2019 combined, find the players that were the most successful in completed dribbles. Sort them from highest to lowest. **Output the player names and the number of successful completed dribbles.** Consider the players that made at least one successful dribble (the smallest number of successful dribbles is 1, not 0).

10. Q_10: In the La Liga season of 2020/2021, find the players that were least dribbled past. Sort them from lowest to highest. **Output the player names and the number of dribbles.** Consider the players that were at least dribbled past once (the lowest number of occurrences of dribbling past the player is 1, not 0).

The auto-grader writes the output of the previous queries into CSV files (one for each query). These files will be compared to the gold standard output to verify the correctness of your answers. The grading will be based on the correctness and the efficiency[4] of your program. The grading rubric is discussed in Section **??**.

2. Deliverables

You will be required to submit the following:

1. A project report as one pdf file (on Brightspace).
2. Github repository URL[5] that includes:
   (a) Your exported database from PostgreSQL named **"dbexport.sql"**. **The command line to create your database dump should look like this:**

   ```
   C:\Program Files\PostgreSQL\13\bin>pg_dump.exe --file "C:\\Users\\user\\
   Desktop\\db_dump.sql" --host "localhost" --port "5433" --username "postgres" --verbose
    --format=p "postgres"
   ```

   This command will use plain text to create the dump for better readability. **You will need to import only the data from the seasons mentioned above such that your exported database is not excessively large.** The size limit on the Github repository is 2 GB. However, you are required to import all event types encountered in the dataset.

   (b) Your source code file(s) that maps and loads the existing JSON dataset from the JSON files into your database. This code must be stored in a directory named "json_loader".

   (c) The "queries.py" script found in the auto-grader with your SQL queries inserted into the right place in the file[6].

The Github repository will be submitted using this Google form: `https://forms.gle/VsWDwopfbjUX3L4YA`. You will be required to sign in to Google to verify your identity. If for any reason, you do not have a Gmail account, please send to me directly using ahmed.elroby@carleton.ca with the following information: Your name, id, Carleton email address, and the Github repository URL. If you implement the bonus queries (Section **??**), also send the URL of the video submission in the email message. Use "COMP 3005 - Project Submission" as the email subject. The form will be closed the next morning of the deadline. You can only make **one submission**. So, please fill out the form after submitting the report and make sure that the information in the form is correct before clicking on "Submit".

You are allowed to work on this project in teams of 3 students or less. But your team should make only one submission on behalf of the team members. But you must indicate the names, IDs, and emails of the team members in the project report. Teams made up of two students will get 10% bonus of their base grade. Teams made up of one student will get 20% bonus of their base grade.

3. Project Report

You need to submit one report file that contains the following sections. You can add other sections, but the following sections **must be** in the report:

[4] The efficiency will be directly impacted by the design of your database since all submissions will be using the same programming language and the code is executed on the same hardware.

[5] Important: Make sure your repository is public at the time of submission.

[6] Do NOT modify anything else in that file. You may risk incorrect grading of your submission, or flagged for suspicion of cheating.

3.1. Conceptual Design

This section should explain the conceptual design of the database. That is, the ER-diagram of the database and the explanation of all the assumptions made in the diagram regarding cardinalities and participation types. Make sure that the assumptions do not contradict with the problem statement in Section ??. Note that although the queries above target a limited number of event types, your design (and database instance) should reflect (store) as many events type as possible.

3.2. Reduction to Relation Schemas

Reduce your ER-diagram into relation schemas and list these in this section.

3.3. Database Schema Diagram

This section should show the final schema diagram of the database. This diagram should be similar to the schema diagram of the university database that we study in this course (Figure ??).

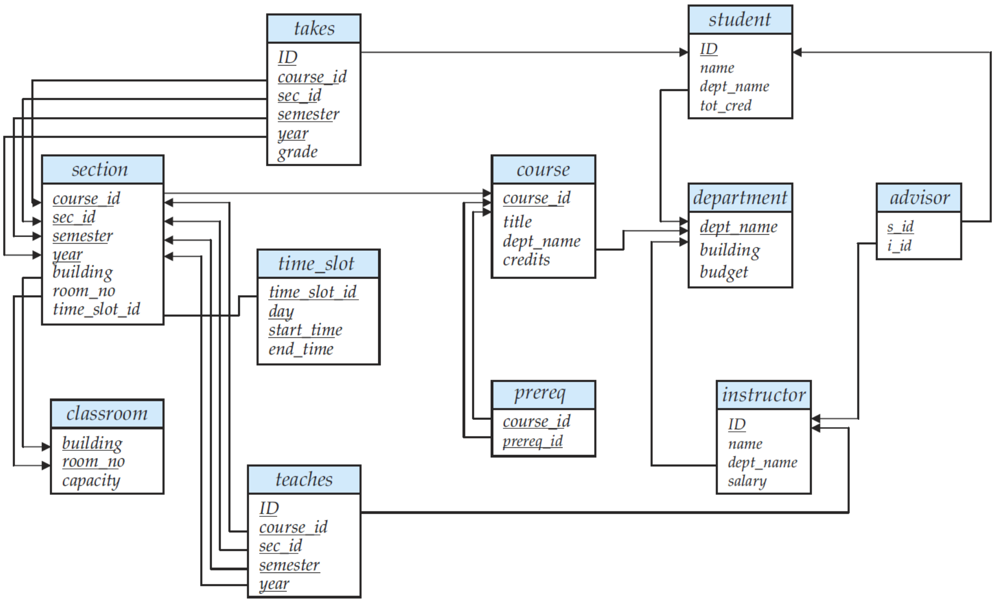

Figure 1: Database Schema Diagram

4. Grading Rubric

This project will be graded based on a total of 130 points. This includes a 9% bonus towards your final grade. The breakdown of the points is as follows:

- Project Report: 30 points.
  - Conceptual Design: 10 points.
  - Reduction to Relational Schemas: 10 points.
  - Schema Diagram: 10 points.
- Query Correctness: 50 points. Each query is worth 5 points.
- Query Efficiency: 50 points.
  - The baseline for efficiency is determined by the best-performing submission successfully processing 100% of the queries. The execution time of each successfully processed query for a submission will be compared to the execution time of the baseline, and the grade for this question will be assigned based on the following formula: $grade_{Q_1} = \frac{executionTime_{baseline}}{executionTime_{submission}} \times 100$. The average grade for the successfully executed queries will be the final efficiency grade.

- Example: One submission successfully process three queries Q_3, Q_5, and Q_{10} with the execution times t_3, t_5, and t_{10}, respectively. The corresponding execution times for the same queries from the baseline submission are t'_3, t'_5, and t'_{10}. The final grade of efficiency of this submission is $grade_{efficiency} = \frac{\left(\frac{t'_3}{t_3}+\frac{t'_5}{t_5}+\frac{t'_{10}}{t_{10}}\right)\times 100}{3}$. If the ratio of execution time $\frac{t'_i}{t_j}$ is greater than 1, it will be set to 1.
- Note that in order to receive a grade for the efficiency component, you need to successfully process at least 3 queries. Otherwise, you will get no points for efficiency.

5. Bonus Queries

The following two queries are more complicated than the aggregate queries required above. Successfully demonstrating any of them is worth 5 extra points. For a possible total of 10 points. The demonstration of these queries will be graded based on a video submission that you can submit using the same Google form you use to submit your Github repository.

1. Divide the goal into 6 equal-size areas (top-left, top-middle, top-right, bottom-left, bottom-middle, and bottom-right). In the La Liga seasons of 2020/2021, 2019/2020, and 2018/2019 combined, find the players who shot the most in either the top-left or top-right corners. Sort them from highest to lowest.
2. In the La Liga season of 2020/2021, find the teams with the most successful passes into the box. Sort them from the highest to lowest.

6. Instructions for Submission

Please follow these instructions in your submission:

- Submit your project report as one pdf file on Brightspace.
- Submit your GitHub repository using this Google form: `https://forms.gle/VsWDwopfbjUX3L4YA`. You can make only **one submission**. Make sure that your GitHub repository is public.
- The due date (April 10th) is a **hard deadline**. No submissions will be accepted after the deadline. It is your responsibility to guarantee there is a version of your report uploaded before the due date. If there is no submission after the due date, you will receive no marks for the project.