Brief

The SCGS Esports Club is organising a school-wide Pubg tournament. The president of the club, Edward Simons (nickname “Eddy the Eagle”), has requested the assistance of the year 12 Digital Solutions class at SCGS to create a database driven app to track the results of the tournament matches.

The secretary of the club, Tom Wright (nickname “Tom the Target”), has provided the following details and rules for the tournament.

- Each team nominated for the tournament must have exactly 4 members.
- Each team must have a unique team name.
- Each player may only be a member of one team.
- Each player must have a valid Pubg account with a nickname and a squad’s rank.
- All tournament matches are “squads” matches where all teams have 4 members. There are no solo or duo matches.
- Up to 25 teams (100 players in total) can participate in any given match.
- At the end of the tournament, the team with the greatest number of wins is the overall tournament winner.
- In the event of a tie, the teams on the same number of wins will play a final match to determine the overall winner.
- The date and time of each match is recorded as well as the duration of the match, the map it was played on, the MVP of the match and the winning team.
- The total number of kills for each team in each match is also recorded.
- Upon registering, each player's real name, nickname and current Pubg squad tier is recorded.
- At the conclusion of the tournament, as well as the winning team, awards are also presented to:
    - the team with the most kills in total, and
    - the individual player with the greatest number of MVPs throughout the tournament.

The Vice President of the club, Jack Cross (nickname “Skull Crusher”), has requested the Digital Solutions class to also design some interfaces for the club to use for an app that would allow the tournament organisers Lachlan Holliday (aka “Lunatic Lachy”) and Edmund Roche (aka ”Radioactive Man”) to input:

- registered player details,
- team details,
- match results and
- match stats, as described above.

The club’s data analysts Jackson Hoffman (aka “Hannibal Hoff”) and Tristan Barry-Smith (aka “Mr. T”) require the proposed app to also output the following reports (utilising SQL queries) to aid them in determining award recipients:

- A list of all teams and their total number of wins for the tournament, sorted in descending order
- A list of all teams and their total number of kills for the tournament, sorted in descending order
- A list of all players and their total number of MVPs for the tournament, sorted in descending order
- A list of all teams and the number of matches they’ve played, sorted in ascending order

The SCGS Esports club thank you for your assistance.

Regards

SCGS Esports secretary, Tom Wright (aka “Tom the Target”)

Your Task:

1. Analyse the data requirements of the proposed app and create a schema diagram which outlines a valid database structure for the scenario.
2. Draw a Data Flow Diagram (DFD) to describe the major external entities, processes and data stores in the proposed system and how data might flow between them.
3. Draw interface wireframes for the following app interfaces required for the app. Annotate the wireframes to show examples of where the ***elements and principles of visual communication*** have been utilised. Also include annotations to show which of the ***useability principles*** have been utilised.
    a. Add new player
    b. Add new team
    c. Enter match result
    d. Enter players kills for match
4. Based on your database schema from part 1, write four SQL queries to satisfy the data analysts requirements from the brief.
5. Write pseudocode algorithms that would add functionality to the interfaces you created in part 3.

Extension:

6. Create a database matching our schema in your preferred database software.
7. Insert dummy data into your database for testing purposes.
8. Run and verify your SQL queries to ensure they provide the required output.
9. Use your preferred programming language and the algorithms you wrote in part 5 to create functional interfaces matching your wireframes from part 3.
10. Using your preferred programming language, create output interfaces for displaying the results of the SQL queries you wrote in part 4 and tested in part 8.