Sports Tournament Tracker: Project Report

Project: Sports Tournament Tracker
Date: July 23, 2025

1. Executive summary

This report details the design, implementation, and functionality of the Sports Tournament Tracker database, a comprehensive system for managing and analyzing sports tournament data. Built using MySQL, this database provides a robust framework for tracking teams, players, match results, and individual player statistics. Key features include the ability to generate dynamic leaderboards, view detailed match histories, calculate player performance metrics, and export summary reports. This system offers a centralized and efficient solution for tournament organizers, analysts, and fans to monitor and engage with the competition.

2. Database Schema Design

The database is built upon a relational schema designed to ensure data integrity and efficient querying. The schema consists of four primary tables: Teams, Players, Matches, and Stats.

2.1. Teams Table
Stores information about each participating team.

- CREATE TABLE Teams (
-     team_id INT AUTO_INCREMENT PRIMARY KEY,
-     team_name VARCHAR(100) NOT NULL UNIQUE
-);

2.2. Players Table
Stores information about each individual player, linked to their respective team.

- player_id (INT, PRIMARY KEY, AUTO_INCREMENT): Unique identifier for each player.
- team_id (INT, FOREIGN KEY): Links the player to a team in the Teams table.
- player_name (VARCHAR(100), NOT NULL): The full name of the player.
- position (VARCHAR(100)): The player's primary position (e.g., Forward, Goalkeeper.

2.3. Matches Table
Records the details of each match played in the tournament.

- match_id (INT, PRIMARY KEY, AUTO_INCREMENT): Unique identifier for each match.

- match_date DATE NOT NULL,
- team1_id INT,
- team2_id INT,
- team1_score INT DEFAULT 0,
- team2_score INT DEFAULT 0,
- venue VARCHAR(100),
- FOREIGN KEY (team1_id) REFERENCES Teams(team_id),
- FOREIGN KEY (team2_id) REFERENCES Teams(team_id)
-

2.4. Stats Table

Captures individual player statistics for each match.

- stat_id (INT, PRIMARY KEY, AUTO_INCREMENT): Unique identifier for each stat entry.
- match_id (INT, FOREIGN KEY): Links the stat to a specific match.
- player_id (INT, FOREIGN KEY): Links the stat to a specific player.
- Points scored (INT, DEFAULT 0): Number of goals scored by the player.
- assists (INT, DEFAULT 0): Number of assists made by the player.

rebounds INT DEFAULT 0,

fouls INT DEFAULT 0,

FOREIGN KEY (match_id) REFERENCES Matches(match_id),

FOREIGN KEY (player_id) REFERENCES Players(player_id)

3. Sample Data Insertion

To populate the database for testing and demonstration, sample data was inserted into each table. This included creating several teams, populating them with players, scheduling matches, and recording player stats for those matches.

Example Insert Statements:

INSERT INTO Teams (team_name) VALUES

('Eagles'), ('Tigers'), ('Sharks'), ('Wolves');

-- Insert Players

INSERT INTO Players (player_name, team_id, position) VALUES

```sql
('Alice Johnson', 1, 'Forward'),
('Bob Smith', 1, 'Guard'),
('Charlie Lee', 2, 'Center'),
('Diana Prince', 2, 'Forward'),
('Evan Davis', 3, 'Guard'),
('Fiona White', 3, 'Forward'),
('George King', 4, 'Center'),
('Hannah Scott', 4, 'Guard');

INSERT INTO Matches (match_date, team1_id, team2_id, team1_score, team2_score,
venue) VALUES
('2024-06-01', 1, 2, 78, 65, 'Stadium A'),
('2024-06-02', 3, 4, 55, 60, 'Stadium B'),
('2024-06-05', 1, 3, 82, 75, 'Stadium A'),
('2024-06-06', 2, 4, 70, 69, 'Stadium C');

INSERT INTO Stats (match_id, player_id, points_scored, assists, rebounds, fouls)
VALUES
(1, 1, 25, 5, 8, 2),
(1, 2, 20, 7, 3, 1),
(1, 3, 30, 4, 10, 3),
(1, 4, 15, 2, 5, 4),

(2, 5, 22, 6, 7, 2),
(2, 6, 18, 7, 4, 2),
(2, 7, 28, 3, 11, 3),
(2, 8, 20, 5, 6, 1),

(3, 1, 35, 4, 9, 2),
(3, 2, 28, 8, 3, 0),
(3, 5, 25, 6, 5, 1),
(3, 6, 20, 3, 7, 1),

(4, 3, 24, 7, 9, 2),
(4, 4, 18, 6, 6, 3),
(4, 7, 30, 2, 12, 4),
(4, 8, 19, 5, 4, 2);
```

4. Core Queries and Views

A suite of SQL queries and views was developed to extract meaningful information from the database.

4.1. Match Results Query

This query retrieves a formatted list of all match results, including team names and scores.

```
SELECT
   m.match_id,

  m.match_date,

  t1.team_name AS team1,

  m.team1_score,

  t2.team_name AS team2,

  m.team2_score,

  m.venue

FROM Matches m
JOIN Teams t1 ON m.team1_id = t1.team_id
JOIN Teams t2 ON m.team2_id = t2.team_id

ORDER BY m.match_date;
```

4.2. Player Scores Leaderboard View
A view was created to generate a leaderboard of top-scoring players based on goals.

```
CREATE VIEW V_PlayerLeaderboard AS
SELECT
   SELECT

  p.player_name,

  t.team_name,

  SUM(s.points_scored) AS total_points,

  SUM(s.assists) AS total_assists,

  SUM(s.rebounds) AS total_rebounds

FROM Stats s

JOIN Players p ON s.player_id = p.player_id

JOIN Teams t ON p.team_id = t.team_id

GROUP BY p.player_id
```

- **Team standing by point (win =3 ,draw=1, loss=0)**

```
CREATE OR REPLACE VIEW TeamStandings AS

SELECT

    team_id,

    team_name,

    COUNT(*) AS games_played,

    SUM(CASE
        WHEN (team_id = Matches.team1_id AND team1_score > team2_score) OR
(team_id = Matches.team2_id AND team2_score > team1_score) THEN 1

        ELSE 0

    END) AS wins,

    SUM(CASE

        WHEN team1_score = team2_score THEN 1

        ELSE 0

    END) AS draws,

    SUM(CASE

        WHEN (team_id = Matches.team1_id AND team1_score < team2_score) OR
(team_id = Matches.team2_id AND team2_score < team1_score) THEN 1

        ELSE 0

    END) AS losses,

    SUM(CASE

        WHEN team_id = Matches.team1_id THEN team1_score

        WHEN team_id = Matches.team2_id THEN team2_score

        ELSE 0

    END) AS goals_for,

    SUM(CASE

        WHEN team_id = Matches.team1_id THEN team2_score

        WHEN team_id = Matches.team2_id THEN team1_score

        ELSE 0

    END) AS goals_against
```

```
FROM Teams
JOIN Matches ON (team_id = Matches.team1_id OR team_id = Matches.team2_id)
GROUP BY team_id, team_name
ORDER BY (wins * 3 + draws) DESC, (goals_for - goals_against) DESC;
```

- **View for top scoring players**

```
CREATE OR REPLACE VIEW PlayerLeaderboard AS
SELECT
  p.player_id,
  p.player_name,
  t.team_name,
  SUM(points_scored) AS total_points,
  SUM(assists) AS total_assists,
  SUM(rebounds) AS total_rebounds
FROM Stats s
JOIN Players p ON s.player_id = p.player_id
JOIN Teams t ON p.team_id = t.team_id
GROUP BY p.player_id
ORDER BY total_points DESC;
```

-- **CTE to calculate average stats per player**

```
WITH AvgPlayerPerformance AS (
  SELECT
    p.player_id,
    p.player_name,
    t.team_name,
    COUNT(s.match_id) AS matches_played,
    AVG(s.points_scored) AS avg_points,
    AVG(s.assists) AS avg_assists,
    AVG(s.rebounds) AS avg_rebounds
  FROM Stats s
  JOIN Players p ON s.player_id = p.player_id
  JOIN Teams t ON p.team_id = t.team_id
  GROUP BY p.player_id
)
```

Team Performance Report Query

This query provides a detailed breakdown of a single team's performance, including match results and top player contributors.

```
SELECT * FROM AvgPlayerPerformance
ORDER BY avg_points DESC;
SELECT
  ts.team_name,
  ts.games_played,
  ts.wins,
  ts.draws,
  ts.losses,

  ts.goals_for,

  ts.goals_against,

  (ts.wins * 3 + ts.draws) AS points

FROM TeamStandings ts

ORDER BY points DESC, (ts.goals_for - ts.goals_against) DESC;
```

6. Reporting and Data Export

The system is capable of generating summary reports that can be easily exported. For example, to get a comprehensive performance report for a specific team, a query joining multiple tables can be executed, and the results can be exported to formats like CSV.

7. Conclusion and Summary View

The Sports Tournament Tracker provides a powerful and flexible tool for managing sports data. The relational schema is scalable, and the combination of direct queries and summary views allows for both granular and high-level analysis. The V_PointsTable and V_PlayerLeaderboard views serve as live, dynamic summaries of the tournament's progress, offering immediate insights to all stakeholders. This project successfully meets the objective of creating a centralized system for tracking match results and player statistics efficiently.