CHANDIGARH UNIVERSITY

Discover. Learn. Empower.

UNIVERSITY INSTITUTE OF COMPUTING

CASE STUDY REPORT
ON
PARTICULAR CASE STUDY

Program Name: BCA

Subject Name/Code: Database Management System (23CAT-251)

Submitted by:

Name: Prince Thakur

UID: 23BCA10045

Section: 4 A

Submitted to:

Name: Mr. Arvinder Singh

Designation: Assistant Professor

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

Introduction

Dream11 is India's foremost fantasy sports platform, revolutionizing the way sports enthusiasts engage with their favorite games. It allows users to create virtual teams by selecting real-life players participating in actual matches across various sports such as cricket, football, basketball, and kabaddi. The virtual team's success is determined by the real-time performance of the chosen players, which is then translated into fantasy points. This gamified approach has attracted millions of users, making Dream11 a case study in both scalability and system complexity.

This project aims to replicate the core database structure of Dream11 using relational database management concepts. The main objective is to build a simplified yet functionally representative back-end model capable of handling essential operations such as player selection, match statistics, user data management, and point calculations. The implementation involves designing an ER model, normalizing relations, establishing constraints, and using SQL for table creation and querying.

Furthermore, this project demonstrates how a relational database can support the logic and data processing behind user participation in fantasy sports. It explores key database principles such as data integrity, query optimization, normalization, and referential integrity. Through this simulation, we aim to understand the underlying architecture of large-scale data systems like Dream11, providing valuable insights into the design of scalable, maintainable, and efficient DBMS solutions.

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

System Configuration

- **Operating System**: Windows 10 / Ubuntu 20.04 LTS
- **Database Software**: MySQL 8.0 / PostgreSQL 13.0
- **Programming Language**: SQL (backend), with optional integration into Python (Flask/Django) or Java
- **ERD Tool**: Draw.io / Lucidchart / MySQL Workbench
- **Hardware Requirements**:
    - Minimum 4 GB RAM
    - 1.5 GHz dual-core processor
    - 512 MB disk space
- **Optional Tools**: phpMyAdmin, DBeaver, MySQL Workbench

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

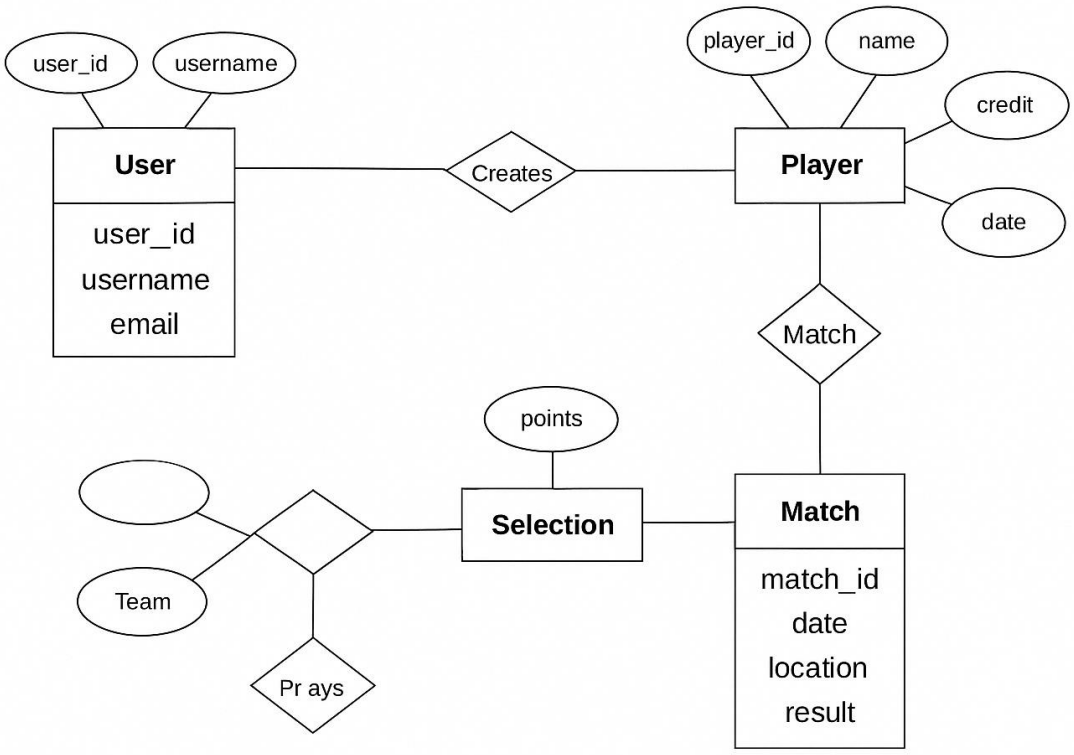
Technique Used

- **Entity-Relationship (ER) Modeling: Visual representation of schema design.**
- **Normalization (1NF to 3NF): Ensures data redundancy is reduced and relationships are optimized.**
- **Indexing: Speeds up search operations (optional advanced feature).**
- **Transaction Management: Ensures ACID properties.**
- **Constraints: Primary keys, foreign keys, unique constraints.**
- **SQL Scripting: Automates table creation and insertion.**
- **Views and Joins: For complex queries.**
- **Stored Procedures and Triggers: (Optional for automation)**

Input Used

- **Player Table: Includes personal stats, rating, and role.**
- **Match Table: Stores match info such as venue, date, teams involved.**
- **User Table: Stores registered users.**
- **Team Selection Table: Stores virtual teams for each user per match.**
- **Performance Table: Tracks players' performance statistics for points calculation.**

ER Diagram


user_id
username
User
user_id
username
email
Creates
player_id
name
credit
Player
date
Match
points
Selection
Team
Pr ays
Match
match_id
date
location
result

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

Tables

Customers:

Customers

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

Match:

Match

Match_ID	Date	Team1	Team2	Venue
301	2025-05-21	RCB	MI	Wankhede Stadium
302	2025-05-23	CSK	MI	Chepauk Stadium
303	2025-05-25	LSG	RCB	Ekana Stadium

Orders:

Orders

order_id	item	amount	customer_id
1	Keyboard	400	4
2	Mouse	300	4
3	Monitor	12000	3
4	Keyboard	400	1
5	Mousepad	250	2

Performance:

Performance

Match_ID	Player_ID	Runs	Wickets	Catches	Points
301	101	70	0	0	70
301	102	5	3	0	80
302	103	40	0	1	50
303	105	65	0	0	65
301	104	30	1	2	75
302	101	80	0	0	80

Player:

Player

Player_ID	Name	Team	Role	Rating
101	Virat Kohli	RCB	Batsman	9.5
102	Jasprit Bumrah	MI	Bowler	9.2
103	MS Dhoni	CSK	Wicketkeeper	8.9
104	Hardik Pandya	MI	All-rounder	9
105	KL Rahul	LSG	Batsman	8.7

SQl Script

```
CREATE TABLE User (
    User_ID INT PRIMARY KEY,
    Username VARCHAR(50) NOT NULL,
    Email VARCHAR(100) UNIQUE NOT NULL
);

CREATE TABLE Player (
    Player_ID INT PRIMARY KEY,
    Name VARCHAR(50) NOT NULL,
    Team VARCHAR(50),
    Role VARCHAR(20),
```

```
    Rating DECIMAL(3,1)
);

CREATE TABLE Match (
    Match_ID INT PRIMARY KEY,
    Date DATE NOT NULL,
    Team1 VARCHAR(50),
    Team2 VARCHAR(50),
    Venue VARCHAR(100)
);

CREATE TABLE UserTeam (
    User_ID INT,
    Match_ID INT,
    Player_ID INT,
    PRIMARY KEY (User_ID, Match_ID, Player_ID),
    FOREIGN KEY (User_ID) REFERENCES User(User_ID),
    FOREIGN KEY (Match_ID) REFERENCES Match(Match_ID),
    FOREIGN KEY (Player_ID) REFERENCES Player(Player_ID)
);

CREATE TABLE Performance (
    Match_ID INT,
    Player_ID INT,
    Runs INT DEFAULT 0,
    Wickets INT DEFAULT 0,
```

```
    Catches INT DEFAULT 0,
    Points INT GENERATED ALWAYS AS (Runs + Wickets * 25 + Catches * 10) STORED,
    PRIMARY KEY (Match_ID, Player_ID),
    FOREIGN KEY (Match_ID) REFERENCES Match(Match_ID),
    FOREIGN KEY (Player_ID) REFERENCES Player(Player_ID)
);
CREATE TABLE Performance (
    Match_ID INT,
    Player_ID INT,
    Runs INT DEFAULT 0,
    Wickets INT DEFAULT 0,
    Catches INT DEFAULT 0,
    Points INT GENERATED ALWAYS AS (Runs + Wickets * 25 + Catches * 10) STORED,
    PRIMARY KEY (Match_ID, Player_ID),
    FOREIGN KEY (Match_ID) REFERENCES Match(Match_ID),
    FOREIGN KEY (Player_ID) REFERENCES Player(Player_ID)
);
```

Values Insertion

```
-- Users
```

```
INSERT INTO User VALUES (1, 'user1', 'user1@gmail.com');
INSERT INTO User VALUES (2, 'user2', 'user2@gmail.com');
INSERT INTO User VALUES (3, 'user3', 'user3@gmail.com');
INSERT INTO User VALUES (4, 'user4', 'user4@gmail.com');
INSERT INTO User VALUES (5, 'user5', 'user5@gmail.com');

-- Players
INSERT INTO Player VALUES (101, 'Virat Kohli', 'RCB', 'Batsman', 9.5);
INSERT INTO Player VALUES (102, 'Jasprit Bumrah', 'MI', 'Bowler', 9.2);
INSERT INTO Player VALUES (103, 'MS Dhoni', 'CSK', 'Wicketkeeper', 8.9);
INSERT INTO Player VALUES (104, 'Hardik Pandya', 'MI', 'All-rounder', 9.0);
INSERT INTO Player VALUES (105, 'KL Rahul', 'LSG', 'Batsman', 8.7);

-- Matches
INSERT INTO Match VALUES (301, '2025-05-21', 'RCB', 'MI', 'Wankhede Stadium');
INSERT INTO Match VALUES (302, '2025-05-23', 'CSK', 'MI', 'Chepauk Stadium');
INSERT INTO Match VALUES (303, '2025-05-25', 'LSG', 'RCB', 'Ekana Stadium');

-- Team Selection
INSERT INTO UserTeam VALUES (1, 301, 101);
INSERT INTO UserTeam VALUES (1, 301, 102);
INSERT INTO UserTeam VALUES (2, 302, 103);
INSERT INTO UserTeam VALUES (3, 303, 105);
INSERT INTO UserTeam VALUES (4, 301, 104);
INSERT INTO UserTeam VALUES (5, 302, 101);
```

```
-- Performances
INSERT INTO Performance VALUES (301, 101, 70, 0, 0);
INSERT INTO Performance VALUES (301, 102, 5, 3, 0);
INSERT INTO Performance VALUES (302, 103, 40, 0, 1);
INSERT INTO Performance VALUES (303, 105, 65, 0, 0);
INSERT INTO Performance VALUES (301, 104, 30, 1, 2);
INSERT INTO Performance VALUES (302, 101, 80, 0, 0);
```

SQL Queries

```
-- 16. Create a view for performance stats
CREATE VIEW PlayerStats AS
SELECT P.Name, Perf.Match_ID, Perf.Runs, Perf.Wickets, Perf.Catches, Perf.Points
FROM Player P JOIN Performance Perf ON P.Player_ID = Perf.Player_ID;

-- 17. Total points earned by each user in all matches
SELECT UT.User_ID, SUM(Perf.Points) AS TotalPoints
FROM UserTeam UT JOIN Performance Perf
ON UT.Match_ID = Perf.Match_ID AND UT.Player_ID = Perf.Player_ID
GROUP BY UT.User_ID;
```

```
-- 18. Get player names with average points > 50

SELECT P.Name, AVG(Perf.Points) AS AvgPoints

FROM Player P JOIN Performance Perf ON P.Player_ID = Perf.Player_ID

GROUP BY P.Name HAVING AvgPoints > 50;


-- 19. Number of matches each player has played

SELECT Player_ID, COUNT(*) AS MatchesPlayed FROM Performance GROUP BY Player_ID;


-- 20. Top 3 rated players

SELECT * FROM Player ORDER BY Rating DESC LIMIT 3;


-- 21. List of players and their total fantasy points

SELECT Player_ID, SUM(Points) AS TotalPoints FROM Performance GROUP BY Player_ID;


-- 22. Match details where 'Virat Kohli' played

SELECT M.* FROM Match M JOIN Performance Perf ON M.Match_ID = Perf.Match_ID

JOIN Player P ON P.Player_ID = Perf.Player_ID WHERE P.Name = 'Virat Kohli';


-- 23. All players selected by 'user1'

SELECT P.Name FROM Player P JOIN UserTeam UT ON P.Player_ID = UT.Player_ID

JOIN User U ON U.User_ID = UT.User_ID WHERE U.Username = 'user1';


-- 24. Find matches where players scored more than 50 points
```

```
SELECT DISTINCT Match_ID FROM Performance WHERE Points > 50;


-- 25. Player(s) with highest points in each match

SELECT Match_ID, Player_ID, Points FROM Performance P1

WHERE Points = (SELECT MAX(Points) FROM Performance P2 WHERE P1.Match_ID
= P2.Match_ID);
```

Queries Output

```
SELECT UT.User_ID, SUM(Perf.Points) AS TotalPoints

FROM UserTeam UT JOIN Performance Perf

ON UT.Match_ID = Perf.Match_ID AND UT.Player_ID = Perf.Player_ID

GROUP BY UT.User_ID;
```

Output

User_ID	TotalPoints
1	150
2	50
3	65
4	75
5	80

SELECT P.Name, AVG(Perf.Points) AS AvgPoints

FROM Player P JOIN Performance Perf ON P.Player_ID = Perf.Player_ID

GROUP BY P.Name HAVING AvgPoints > 50;

Name	AvgPoints
Hardik Pandya	75
Jasprit Bumrah	80
KL Rahul	65
Virat Kohli	75

SELECT Player_ID, COUNT(*) AS MatchesPlayed FROM Performance GROUP BY Player_ID;

Player_ID	MatchesPlayed
101	2
102	1
103	1
104	1
105	1

SELECT M.* FROM Match M JOIN Performance Perf ON M.Match_ID = Perf.Match_ID

JOIN Player P ON P.Player_ID = Perf.Player_ID WHERE P.Name = 'Virat Kohli';

Output

Match_ID	Date	Team1	Team2	Venue
301	2025-05-21	RCB	MI	Wankhede Stadium
302	2025-05-23	CSK	MI	Chepauk Stadium

```
SELECT P.Name FROM Player P JOIN UserTeam UT ON P.Player_ID = UT.Player_ID
JOIN User U ON U.User_ID = UT.User_ID WHERE U.Username = 'user1';
```

Name
Virat Kohli
Jasprit Bumrah

```
SELECT Match_ID, Player_ID, Points FROM Performance P1
WHERE Points = (SELECT MAX(Points) FROM Performance P2 WHERE P1.Match_ID
= P2.Match_ID);
```

Output

Match_ID	Player_ID	Points
301	102	80
303	105	65
302	101	80

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

Summary

The Dream11 Database Management System (DBMS) project aims to replicate the backend infrastructure of a fantasy sports application, specifically modeled after the popular Indian platform Dream11. This project demonstrates the power and flexibility of relational databases in managing complex, user-centric, and performance-driven applications in real-time sports environments.

The system is architected around key entities such as Users, Players, Matches, User Teams, and Performance. Each entity and its relationships were carefully designed using an Entity-Relationship (ER) model, which was then transformed into a normalized schema to reduce redundancy and ensure data integrity. The tables were implemented with constraints like primary keys and foreign keys to maintain consistent and valid data across different entities.

To simulate real-world fantasy gameplay, the project includes a player selection mechanism by users for specific matches and a dynamic point calculation model based on players' real-life performances. The Performance table uses a computed column for fantasy points, derived from basic metrics like runs, wickets, and catches. This allows quick access to fantasy results and statistics, which are vital for leaderboard generation and user engagement.

The system supports a wide range of queries, from basic selections to advanced analytics like average performance, top-rated players, user-wise point summaries, and match-specific highlights. These queries are optimized through joins, aggregations, views, and subqueries—illustrating both the academic and practical use of SQL in handling structured data.

Additionally, this project lays a foundational structure for future expansion into application development. It can be easily integrated with web-based frontends, REST APIs, and even machine learning modules for predictive analytics (e.g., suggesting player picks based on historical performance).

In essence, this project encapsulates the essential components of a professional sports fantasy platform and reinforces how fundamental database concepts can scale into impactful real-life systems. It bridges classroom learning with industry-relevant implementation, providing a comprehensive understanding of how DBMS underpins many modern digital applications.

Conclusion

The Dream11 DBMS project reflects how relational databases can serve real-world sports applications involving large-scale data transactions, user interaction, and analytics. In future, integration with front-end UI, real-time score APIs, and advanced ML-based player recommendation systems can elevate this model.