

Project Phase 3 – Relational Database of Soccer/Football ER Diagram

Team: Stormers

By:

Nirmit Dagli

Srujan Sai Vemula

Nicholas Darko Brown

1. DESCRIPTION OF USE CASES

1. Comprehensive Player Performance Analysis Use Case: Technical Analysis for Player Evaluation

- Primary Users: Technical Directors, Scouts, Managers
- Purpose: Evaluate overall player performance across multiple metrics
- Business Value:
 - Identify top performers for potential recruitment
 - Assess current squad performance
 - Make informed decisions about contract renewals
 - Support team selection decisions
- Key Metrics:
 - Overall contribution (goals + assists)
 - Match participation
 - Passing ability
 - Physical performance (distance covered)
- Filtering Logic:
 - Minimum 3 matches played (ensures statistical significance)
 - Minimum 3 goal contributions (focuses on impactful players)

2. Young Talent Analysis Use Case: Youth Talent Scouting and Development

- Primary Users: Scouts, Youth Development Teams, Recruitment Teams
- Purpose: Identify and track promising young players
- Business Value:
 - Early identification of talent
 - Investment opportunities in young players
 - Long-term squad planning
 - Academy performance assessment
- Key Metrics:
 - Age verification
 - Playing time analysis
 - Production metrics (goals/assists)
 - Technical ability (passing stats)
- Filtering Logic:
 - Born after 2001 (under 23)
 - Minimum 2 matches and 60 minutes per game (regular players)

3. Team Performance Analysis Use Case: Team Performance Monitoring and Reporting

- Primary Users: Club Management, Coaches, Media Teams
- Purpose: Analyze team's overall performance and trends
- Business Value:
 - Track team progress
 - Compare performance against targets
 - Stadium utilization analysis
 - Financial planning (attendance trends)
- Key Metrics:
 - Win/Draw/Loss record
 - Goal difference
 - Average attendance
 - Points calculation (3 for win, 1 for draw)
- Filtering Logic:
 - Minimum 3 matches played (ensures meaningful analysis)
 - Complete game records (wins + draws + losses \geq 3)

4. Injury Impact Analysis Use Case: Medical and Performance Impact Assessment

- Primary Users: Medical Staff, Fitness Coaches, Team Managers
- Purpose: Analyze how injuries affect player performance
- Business Value:
 - Improve injury prevention strategies
 - Optimize return-to-play protocols
 - Monitor player recovery effectiveness
 - Risk assessment for player availability
- Key Metrics:
 - Injury frequency
 - Recovery time analysis
 - Post-injury performance
 - Playing time management
- Filtering Logic:
 - Has recorded injuries
 - Minimum 2 matches after return (performance assessment)

5. Playmaker Performance Analysis Use Case: Creative Player Assessment

- Primary Users: Tactical Analysts, Coaches, Recruitment Team
- Purpose: Evaluate players' creative and assistive contributions
- Business Value:
 - Tactical planning
 - Identify key playmakers
 - Recruitment planning
 - Team style analysis
- Key Metrics:
 - Assist numbers
 - Passing efficiency
 - Creative contribution
 - Work rate (distance covered)
- Filtering Logic:
 - Only midfielders and forwards
 - Minimum 3 matches played
 - Minimum 2 assists
 - Ordered by creative contribution

2. SQL QUERIES FOR USE CASES

1. SELECT

```
p.name AS player_name,  
t.team_name,  
pos.position_name,  
COUNT(DISTINCT pms.match_id) as matches_played,  
SUM(pms.goals_scored) as total_goals,  
SUM(pms.assists) as total_assists,  
ROUND(AVG(pms.passes_completed), 1) as avg_passes,  
ROUND(AVG(pms.distance_covered), 1) as avg_distance,  
ROUND((SUM(pms.goals_scored) + SUM(pms.assists)) / COUNT(DISTINCT  
pms.match_id), 2) as goal_contributions_per_game  
FROM player p  
INNER JOIN player_match_stats pms ON p.player_id = pms.player_id  
INNER JOIN contract c ON p.player_id = c.player_id  
INNER JOIN team t ON c.team_id = t.team_id  
INNER JOIN position pos ON p.position_id = pos.position_id  
GROUP BY p.player_id, p.name, t.team_name, pos.position_name  
HAVING  
    matches_played >= 3 AND  
    (total_goals + total_assists) >= 3  
ORDER BY goal_contributions_per_game DESC;
```

Player Performance Analysis

The screenshot displays a database management interface with a sidebar on the left showing a tree view of databases and tables. The main area contains a SQL query editor with a query for player performance analysis. Below the query editor is a 'Result Grid' showing the results of the query. The query filters for players with at least 3 matches played and a combined total of goals and assists of at least 3, ordered by goal contributions per game in descending order.

```
1 SELECT
2     p.name AS player_name,
3     t.team_name,
4     pos.position_name,
5     COUNT(DISTINCT pms.match_id) as matches_played,
6     SUM(pms.goals_scored) as total_goals,
7     SUM(pms.assists) as total_assists,
8     ROUND(AVG(pms.passes_completed), 1) as avg_passes,
9     ROUND(AVG(pms.distance_covered), 1) as avg_distance,
10    ROUND((SUM(pms.goals_scored) + SUM(pms.assists)) / COUNT(DISTINCT pms.match_id), 2) as goal_contributions_per_game
11 FROM player p
12 INNER JOIN player_match_stats pms ON p.player_id = pms.player_id
13 INNER JOIN contract c ON p.player_id = c.player_id
14 INNER JOIN team t ON c.team_id = t.team_id
15 INNER JOIN position pos ON p.position_id = pos.position_id
16 GROUP BY p.player_id, p.name, t.team_name, pos.position_name
17 HAVING
18     matches_played >= 3 AND
19     (total_goals + total_assists) >= 3
20 ORDER BY goal_contributions_per_game DESC;
```

player_name	team_name	position_name	matches_played	total_goals	total_assists	avg_passes	avg_distance	goal_contributions_per_game
Bruno Fernandes	Manchester United	Midfielder	3	3	6	75.0	11.5	3.00
Bukayo Saka	Arsenal	Forward	3	4	5	41.7	11.3	3.00
Kylian Mbappé	Real Madrid	Forward	3	6	2	38.3	10.5	2.67
Erling Haaland	Barcelona	Striker	3	7	1	27.7	10.2	2.67
Mohamed Salah	Liverpool FC	Forward	3	5	3	38.9	9.8	2.67
Phil Foden	Manchester City	Forward	3	4	4	48.3	10.5	2.67
Jude Bellingham	Real Madrid	Midfielder	3	5	2	68.3	12.0	2.33
Harry Kane	Arsenal	Striker	5	8	2	42.0	10.1	2.00
Kevin De Bruyne	Manchester United	Midfielder	5	2	8	77.6	11.0	2.00
Luka Modric	Real Madrid	Midfielder	3	1	5	92.0	10.8	2.00

Result 38

Time	Action	Response	Duration / Fetch
103 15:35:16	SELECT p.name AS player_name, t.team_name, pos.position_name, COUNT(...	10 row(s) returned	0.0028 sec / 0.00
104 15:39:30	SELECT p.name AS player_name, t.team_name, pos.position_name, COUNT(...	10 row(s) returned	0.0026 sec / 0.00

2. SELECT

```
p.name AS player_name,
t.team_name,
TIMESTAMPDIFF(YEAR, p.birthday, CURDATE()) as age,
COUNT(DISTINCT pms.match_id) as matches_played,
ROUND(AVG(pms.minutes_played), 1) as avg_minutes,
SUM(pms.goals_scored) as total_goals,
SUM(pms.assists) as total_assists,
ROUND(AVG(pms.passes_completed), 1) as avg_passes
FROM player p
INNER JOIN player_match_stats pms ON p.player_id = pms.player_id
INNER JOIN contract c ON p.player_id = c.player_id
INNER JOIN team t ON c.team_id = t.team_id
WHERE p.birthday >= '2001-01-01'
GROUP BY p.player_id, p.name, t.team_name, p.birthday
HAVING
    matches_played >= 2 AND
    avg_minutes >= 60
ORDER BY total_goals + total_assists DESC;
```

book_sales

csc325

lms_db

myDB

soccer

Tables

award_type

city

contract

contract_status

country

football_match

injury

injury_type

language

league

match_official

official_assignment

personnel

player

player_match_stats

position

role

severity

specialization

stadium

team

tournament

tournament_type

transfer

transfer_type

Views

Stored Procedures

Functions

sys

test_company

```

1 SELECT
2     p.name AS player_name,
3     t.team_name,
4     TIMESTAMPDIFF(YEAR, p.birthday, CURDATE()) as age,
5     COUNT(DISTINCT pms.match_id) as matches_played,
6     ROUND(AVG(pms.minutes_played), 1) as avg_minutes,
7     SUM(pms.goals_scored) as total_goals,
8     SUM(pms.assists) as total_assists,
9     ROUND(AVG(pms.passes_completed), 1) as avg_passes
10 FROM player p
11 INNER JOIN player_match_stats pms ON p.player_id = pms.player_id
12 INNER JOIN contract c ON p.player_id = c.player_id
13 INNER JOIN team t ON c.team_id = t.team_id
14 WHERE p.birthday >= '2001-01-01'
15 GROUP BY p.player_id, p.name, t.team_name, p.birthday
16 HAVING
17     matches_played >= 2 AND
18     avg_minutes >= 60
19 ORDER BY total_goals + total_assists DESC;

```

100%

7:13

Result Grid

Filter Rows:

Search

Export:

player_name	team_name	age	matches_played	avg_minutes	total_goals	total_assists	avg_passes
Bukayo Saka	Arsenal	23	3	90.0	4	5	41.7
Jude Bellingham	Real Madrid	21	3	90.0	5	2	68.3

Result 40

Action Output

	Time	Action	Response	Duration / Fetch
105	15:46:12	SELECT p.name AS player_name, t.team_name, TIMESTAMPDIFF(YEAR, p.birthday...	2 row(s) returned	0.0021 sec / 0.00
106	15:46:31	SELECT p.name AS player_name, t.team_name, TIMESTAMPDIFF(YEAR, p.birthday...	2 row(s) returned	0.0022 sec / 0.00

Young Talent Analysis (Players under 23)

3. SELECT

```
t.team_name,  
COUNT(DISTINCT m.match_id) as matches_played,  
SUM(CASE WHEN m.home_score > m.away_score THEN 1 ELSE 0 END) as wins,  
SUM(CASE WHEN m.home_score = m.away_score THEN 1 ELSE 0 END) as draws,  
SUM(CASE WHEN m.home_score < m.away_score THEN 1 ELSE 0 END) as losses,  
SUM(m.home_score) as goals_for,  
SUM(m.away_score) as goals_against,  
ROUND(AVG(m.attendance), 0) as avg_attendance  
FROM team t  
INNER JOIN stadium s ON t.home_stadium_id = s.stadium_id  
INNER JOIN football_match m ON s.stadium_id = m.stadium_id  
GROUP BY t.team_id, t.team_name  
HAVING  
    matches_played >= 3 AND  
    wins + draws + losses >= 3  
ORDER BY (wins * 3 + draws) DESC;
```

book_sales

csc325

lms_db

myDB

soccer

Tables
award_type
city
contract
contract_status
country
football_match
injury
injury_type
language
league
match_official
official_assignment
personnel
player
player_match_stats
position
role
severity
specialization
stadium
team
tournament
tournament_type
transfer
transfer_type
Views
Stored Procedures
Functions
sys
test_company

```

1 SELECT
2     t.team_name,
3     COUNT(DISTINCT m.match_id) as matches_played,
4     SUM(CASE WHEN m.home_score > m.away_score THEN 1 ELSE 0 END) as wins,
5     SUM(CASE WHEN m.home_score = m.away_score THEN 1 ELSE 0 END) as draws,
6     SUM(CASE WHEN m.home_score < m.away_score THEN 1 ELSE 0 END) as losses,
7     SUM(m.home_score) as goals_for,
8     SUM(m.away_score) as goals_against,
9     ROUND(AVG(m.attendance), 0) as avg_attendance
10 FROM team t
11 INNER JOIN stadium s ON t.home_stadium_id = s.stadium_id
12 INNER JOIN football_match m ON s.stadium_id = m.stadium_id
13 GROUP BY t.team_id, t.team_name
14 HAVING
15     matches_played >= 3 AND
16     wins + draws + losses >= 3
17 ORDER BY (wins * 3 + draws) DESC;

```

100%

34:17

Result Grid

Filter Rows: Search

Export:

team_name	matches_played	wins	draws	losses	goals_for	goals_against	avg_attendan...
Arsenal	3	3	0	0	7	3	58333
Liverpool FC	3	3	0	0	9	2	51333
Chelsea	3	3	0	0	7	3	58333
Tottenham	3	3	0	0	9	2	51333
Real Madrid	3	2	0	1	7	4	77000
Bayern Munich	3	2	0	1	7	4	77000
Manchester United	3	0	3	0	8	9	70333
Barcelona	3	0	3	0	6	6	85000
Manchester City	3	0	3	0	8	8	70333
PSG	3	0	3	0	6	6	85000

Result 42

Action Output

	Time	Action	Response	Duration / Fetch
107	15:49:10	SELECT t.team_name, COUNT(DISTINCT m.match_id) as matches_played, SUM...	SUM... 10 row(s) returned	0.0023 sec / 0.00
108	15:49:24	SELECT t.team_name, COUNT(DISTINCT m.match_id) as matches_played, SUM...	SUM... 10 row(s) returned	0.0024 sec / 0.00

Team Performance Analysis

4. SELECT

p.name AS player_name,

t.team_name,

COUNT(DISTINCT i.injury_id) as injury_count,

ROUND(AVG(DATEDIFF(i.expected_return, i.injury_date)), 0) as avg_recovery_days,

COUNT(DISTINCT pms.match_id) as matches_played_after_injury,

ROUND(AVG(CASE

 WHEN m.match_date > i.expected_return

 THEN pms.minutes_played

END), 0) as avg_minutes_after_return

FROM player p

INNER JOIN contract c ON p.player_id = c.player_id

INNER JOIN team t ON c.team_id = t.team_id

INNER JOIN injury i ON p.player_id = i.player_id

INNER JOIN player_match_stats pms ON p.player_id = pms.player_id

INNER JOIN football_match m ON pms.match_id = m.match_id

GROUP BY p.player_id, p.name, t.team_name

HAVING

 injury_count > 0 AND

 matches_played_after_injury >= 2

ORDER BY avg_recovery_days DESC;

book_sales

csc325

lms_db

myDB

soccer

Tables

award_type

city

contract

contract_status

country

football_match

injury

injury_type

language

league

match_official

official_assignment

personnel

player

player_match_stats

position

role

severity

specialization

stadium

team

tournament

tournament_type

transfer

transfer_type

Views

Stored Procedures

Functions

sys

test_company

```

1 SELECT
2     p.name AS player_name,
3     t.team_name,
4     COUNT(DISTINCT i.injury_id) as injury_count,
5     ROUND(AVG(DATEDIFF(i.expected_return, i.injury_date)), 0) as avg_recovery_days,
6     COUNT(DISTINCT pms.match_id) as matches_played_after_injury,
7     ROUND(AVG(CASE
8         WHEN m.match_date > i.expected_return
9             THEN pms.minutes_played
10        END), 0) as avg_minutes_after_return
11 FROM player p
12 INNER JOIN contract c ON p.player_id = c.player_id
13 INNER JOIN team t ON c.team_id = t.team_id
14 INNER JOIN injury i ON p.player_id = i.player_id
15 INNER JOIN player_match_stats pms ON p.player_id = pms.player_id
16 INNER JOIN football_match m ON pms.match_id = m.match_id
17 GROUP BY p.player_id, p.name, t.team_name
18 HAVING
19     injury_count > 0 AND
20     matches_played_after_injury >= 2
21 ORDER BY avg_recovery_days DESC;

```

100% 33:21

Result Grid

Filter Rows: Search Export

player_name	team_name	injury_count	avg_recovery_days	matches_played_after_inj...	avg_minutes_after_ret...
Bruno Fernandes	Manchester United	1	56	3	NULL
Kevin De Bruyne	Manchester United	2	41	5	86
Virgil van Dijk	Liverpool FC	1	20	3	90
Mohamed Salah	Liverpool FC	1	20	3	88
Jude Bellingham	Real Madrid	1	20	3	90
Harry Kane	Arsenal	1	16	5	88
Luka Modric	Real Madrid	1	15	3	82
Bukayo Saka	Arsenal	1	10	3	NULL
Kylian Mbappé	Real Madrid	1	9	3	84

Result 44

Action Output

	Time	Action	Response	Duration / Fetch
✖	115 15:58:48	SELECT p.name AS player_name, t.team_na...	Error Code: 1054. Unknown column 'player_match_stats.match_date'...	0.00096 sec
✖	116 15:57:48	SELECT p.name AS player_name, t.team_na...	Error Code: 1054. Unknown column 'pms.match_date' in 'field list'	0.00075 sec
✔	117 15:59:04	SELECT p.name AS player_name, t.team_na...	9 row(s) returned	0.0020 sec / 0.00
✖	118 16:00:34	SELECT p.name AS player_name, t.team_na...	Error Code: 1064. You have an error in your SQL syntax; check the m...	0.00060 sec
✔	119 16:00:39	SELECT p.name AS player_name, t.team_na...	9 row(s) returned	0.0022 sec / 0.00

Injury Impact on Performance

```
5. SELECT
p.name AS player_name,
t.team_name,
COUNT(DISTINCT pms.match_id) as matches_played,
SUM(pms.assists) as total_assists,
ROUND(AVG(pms.passes_completed), 1) as avg_passes,
ROUND(SUM(pms.assists) * 100.0 / SUM(pms.passes_completed), 2) as assist_efficiency,
ROUND(AVG(pms.distance_covered), 1) as avg_distance
FROM player p
INNER JOIN player_match_stats pms ON p.player_id = pms.player_id
INNER JOIN contract c ON p.player_id = c.player_id
INNER JOIN team t ON c.team_id = t.team_id
INNER JOIN position pos ON p.position_id = pos.position_id
WHERE pos.position_name IN ('Midfielder', 'Forward')
GROUP BY p.player_id, p.name, t.team_name
HAVING
    matches_played >= 3 AND
    total_assists >= 2
ORDER BY total_assists DESC, avg_passes DESC;
```

Filter Objects

book_sales

csc325

lms_db

myDB

soccer

Tables

award_type

city

contract

contract_status

country

football_match

injury

injury_type

language

league

match_official

official_assignment

personnel

player

player_match_stats

position

role

severity

specialization

stadium

team

tournament

tournament_type

transfer

transfer_type

Views

Stored Procedures

Functions

sys

test_company

```

1 SELECT
2     p.name AS player_name,
3     t.team_name,
4     COUNT(DISTINCT pms.match_id) as matches_played,
5     SUM(pms.assists) as total_assists,
6     ROUND(AVG(pms.passes_completed), 1) as avg_passes,
7     ROUND(SUM(pms.assists) * 100.0 / SUM(pms.passes_completed), 2) as assist_efficiency,
8     ROUND(AVG(pms.distance_covered), 1) as avg_distance
9 FROM player p
10 INNER JOIN player_match_stats pms ON p.player_id = pms.player_id
11 INNER JOIN contract c ON p.player_id = c.player_id
12 INNER JOIN team t ON c.team_id = t.team_id
13 INNER JOIN position pos ON p.position_id = pos.position_id
14 WHERE pos.position_name IN ('Midfielder', 'Forward')
15 GROUP BY p.player_id, p.name, t.team_name
16 HAVING
17     matches_played >= 3 AND
18     total_assists >= 2
19 ORDER BY total_assists DESC, avg_passes DESC;

```

100%

46:19

Result Grid

Filter Rows: Search

Export:

player_name	team_name	matches_played	total_assists	avg_passes	assist_efficien...	avg_distance
Kevin De Bruyne	Manchester United	5	8	77.6	2.06	11.0
Bruno Fernandes	Manchester United	3	6	75.0	2.67	11.5
Luka Modric	Real Madrid	3	5	92.0	1.81	10.8
Bukayo Saka	Arsenal	3	5	41.7	4.00	11.3
Phil Foden	Manchester City	3	4	48.3	2.76	10.5
Mohamed Salah	Liverpool FC	3	3	38.3	2.61	9.8
Jude Bellingham	Real Madrid	3	2	68.3	0.98	12.0
Kylian Mbappé	Real Madrid	3	2	38.3	1.74	10.5

Result 46

Action Output

	Time	Action	Response	Duration / Fetch
117	15:59:04	SELECT p.name AS player_name, t.team_na...	9 row(s) returned	0.0020 sec / 0.00
118	16:00:34	SELECT p.name AS player_name, t.team_na...	Error Code: 1064. You have an error in your SQL syntax; check the m...	0.00060 sec
119	16:00:39	SELECT p.name AS player_name, t.team_na...	9 row(s) returned	0.0022 sec / 0.00
120	16:03:32	SELECT p.name AS player_name, t.team_na...	8 row(s) returned	0.0024 sec / 0.00
121	16:03:48	SELECT p.name AS player_name, t.team_na...	8 row(s) returned	0.0023 sec / 0.00

Playmaker Performance Analysis

MongoDB Design Reasoning

The updated MongoDB design uses a unified document structure that brings all related information about a player into a single document. This means that details like personal information, team affiliation, match performance, contracts, injuries, and transfer history are all stored together in one place. For example, a player's match statistics—such as goals scored, assists, and minutes played—are embedded directly within their document, along with team details like the team's name, jersey color, and market value.

By integrating everything into one document, we can retrieve all the data about a player with just one query. This is especially useful for scenarios like scouting reports, performance evaluations, or team management, where all this information is often needed at once. Instead of making multiple queries or joining collections, we get everything in one go, which saves time and improves performance.

This design works well for our use case because MongoDB's schema-less nature allows us to easily adapt and add new fields as needed. For instance, if we wanted to track advanced analytics or player-specific metrics in the future, we could do so without having to redesign the database. While storing all this data in one document might create some redundancy—such as repeating team details for multiple players—it keeps the queries simpler and faster, which is more important for the kind of read-heavy operations we anticipate.

In summary, the unified document approach makes the database efficient, flexible, and easy to work with, especially for real-time access to interrelated data. It's a practical choice for managing the complex and hierarchical nature of sports data.

MongoDB Schema Design

- **Key Entity:** Player
 - Reason: Players are the central unit of analysis and the most dynamic entity, with attributes that frequently change, such as match statistics, injuries, and transfers.
- **Collections:**
 - **Player:** Contains player details such as personal info, current team, contract, match stats, injuries, and tournaments.
 - **Team (embedded in Player):** Stores team-related details like name, stadium, league, and market value.
 - **Match Stats (array in Player):** Includes per-match data like goals scored, minutes played, assists, and results.

- **Tournaments (array in Player):** Captures tournament details such as type (league, cup), start/end dates, and prizes.
- **Injuries (array in Player):** Tracks injury severity, recovery plan, and injury date.
- **Transfers (array in Player):** Records transfer details like type (permanent/loan), teams involved, and transfer fees.

Design Choices and Reasoning

- **Player-Centric Design:**
 - Players are the most dynamic entities, and their data is accessed more frequently than teams.
 - A player-centric approach avoids fragmentation caused by frequent transfers and injuries.
- **Embedded Documents:**
 - Match stats, injuries, and contracts are embedded within the player document for faster read operations.
- **Use of Arrays:**
 - Arrays are used to store repeating attributes like match stats, injuries, transfers, and tournaments, ensuring data is logically grouped.
- **Normalization and Referencing:**
 - Key data like current team details are embedded for quick access.
 - Static team attributes like stadiums and leagues are referenced where possible to avoid redundancy.

Each Team Member's Contribution

1. Nirmitt Dagli:

- MongoDB Schema Design: Created a player-centric database design with embedded documents (implementation of MongoDB Atlas).
- Implemented match stats and injury tracking.
- Coordinated team tasks and ensured timely completion.
- Generated outputs and captured screenshots.

2. Nicholas Brown:

- SQL Queries: Designed and executed SQL queries for retrieving player stats, injuries, and team details.
- Generated outputs and captured screenshots for use case demonstrations.
- Ensured query optimization for large datasets.

3. Sai Srujan Vemula:

- Project and PPT Documentation: Compiled the report
- Tested queries for performance and accuracy
- MongoDB and SQL Schema Design
- Analyzed obstacles and documented the learning experience.