Project Overview

Based on the actual Premier League historical data available, we propose developing a comprehensive football analytics platform focused on team performance, match prediction, and betting market analysis. This system will leverage match-level statistics and extensive betting odds data to create valuable insights for football analysts, fans, and potential bookmakers.

Database Architecture

Operational Database (MySQL on school's remote server)

Core Tables:

1. **Teams**
   - TeamID (PK)
   - TeamName
   - HomeGroundName
   - City
   - IsTop6Club (Boolean)
   - Region
   - YearFounded
2. **Seasons**
   - SeasonID (PK)
   - SeasonName (e.g., "2023-2024")
   - StartDate
   - EndDate
   - NumberOfTeams
   - VARIntroduced (Boolean)
   - COVIDSeason (Boolean)
3. **Matches**
   - MatchID (PK)
   - SeasonID (FK)
   - MatchDate
   - MatchTime
   - HomeTeamID (FK)
   - AwayTeamID (FK)
   - FTHG (Full Time Home Goals)

- FTAG (Full Time Away Goals)
- FTR (Full Time Result)
- HTHG (Half Time Home Goals)
- HTAG (Half Time Away Goals)
- HTR (Half Time Result)
- RefereeID (FK)
- Attendance
- IsNeutralVenue (Boolean)

4. **MatchStatistics**
   - StatID (PK)
   - MatchID (FK)
   - HomeShots
   - AwayShots
   - HomeShotsTarget
   - AwayShotsTarget
   - HomeCorners
   - AwayCorners
   - HomeFouls
   - AwayFouls
   - HomeYellowCards
   - AwayYellowCards
   - HomeRedCards
   - AwayRedCards
5. **Referees**
   - RefereeID (PK)
   - RefereeName
   - YearsExperience
   - MatchesOfficiated
   - AverageCardsPerMatch
6. **Bookmakers**
   - BookmakerID (PK)
   - BookmakerName
   - BookmakerCode (e.g., "B365", "BW", "PS")
   - Website
7. **BettingOdds**
   - OddsID (PK)
   - MatchID (FK)

- BookmakerID (FK)
- MarketType (ENUM: 'MatchResult', 'OverUnder25', 'AsianHandicap')
- HomeWinOdds
- DrawOdds
- AwayWinOdds
- OverOdds
- UnderOdds
- AsianHandicap
- AHHomeOdds
- AHAwayOdds
- OddsTimestamp
- IsClosingOdds (Boolean)

Analytical Database (Star Schema)

Fact Tables:

1. **TeamMatchStatsFact**
   - FactID (PK)
   - MatchID (FK)
   - TeamID (FK)
   - TimeID (FK)
   - RefereeID (FK)
   - IsHome (Boolean)
   - Goals
   - GoalsConceded
   - Shots
   - ShotsOnTarget
   - Corners
   - Fouls
   - YellowCards
   - RedCards
   - Result (ENUM: 'W', 'D', 'L')
   - PointsEarned
   - ExpectedGoals (calculated)
   - ShotAccuracy
   - ShotConversion
2. **BettingMarketFact**

- FactID (PK)
- MatchID (FK)
- BookmakerID (FK)
- TimeID (FK)
- MarketType (FK)
- ActualResult (ENUM: 'H', 'D', 'A')
- HomeWinOdds
- DrawOdds
- AwayWinOdds
- ImpliedProbabilityHome
- ImpliedProbabilityDraw
- ImpliedProbabilityAway
- BookmakerMargin
- OddsWon (Boolean)
- ReturnOnInvestment
- ValueBetIndex

Dimension Tables:

1. **TeamDimension** (expanded from Teams table)
2. **TimeDimension** (date hierarchies: Year, Season, Month, Week, Day)
3. **RefereeDimension** (expanded from Referees table)
4. **BookmakerDimension** (different betting companies)
5. **MarketTypeDimension** (different betting markets)

Key SQL Analyses & Features

1. Team Performance Analysis

```sql
-- Team-specific home advantage analysis with proper aggregation
WITH TeamMatches AS (
    SELECT
        m.MatchID,
        m.HomeTeamID AS TeamID,
        'Home' AS VenueType,
        m.FTHG AS GoalsScored,
        m.FTAG AS GoalsConceded,
        CASE
            WHEN m.FTR = 'H' THEN 3
            WHEN m.FTR = 'D' THEN 1
```

```sql
            ELSE 0
        END AS Points,
        ms.HomeShots AS Shots,
        ms.HomeShotsTarget AS ShotsOnTarget
    FROM Matches m
    JOIN MatchStatistics ms ON m.MatchID = ms.MatchID

    UNION ALL

    SELECT
        m.MatchID,
        m.AwayTeamID AS TeamID,
        'Away' AS VenueType,
        m.FTAG AS GoalsScored,
        m.FTHG AS GoalsConceded,
        CASE
            WHEN m.FTR = 'A' THEN 3
            WHEN m.FTR = 'D' THEN 1
            ELSE 0
        END AS Points,
        ms.AwayShots AS Shots,
        ms.AwayShotsTarget AS ShotsOnTarget
    FROM Matches m
    JOIN MatchStatistics ms ON m.MatchID = ms.MatchID
)
SELECT
    t.TeamName,
    COUNT(CASE WHEN tm.VenueType = 'Home' THEN 1 END) AS HomeMatches,
    COUNT(CASE WHEN tm.VenueType = 'Away' THEN 1 END) AS AwayMatches,
    ROUND(AVG(CASE WHEN tm.VenueType = 'Home' THEN tm.GoalsScored END), 2)
AS AvgHomeGoalsScored,
    ROUND(AVG(CASE WHEN tm.VenueType = 'Away' THEN tm.GoalsScored END), 2)
AS AvgAwayGoalsScored,
    ROUND(AVG(CASE WHEN tm.VenueType = 'Home' THEN tm.Points END), 2) AS
AvgHomePoints,
    ROUND(AVG(CASE WHEN tm.VenueType = 'Away' THEN tm.Points END), 2) AS
AvgAwayPoints,
    ROUND(AVG(CASE WHEN tm.VenueType = 'Home' THEN tm.Points END) -
          AVG(CASE WHEN tm.VenueType = 'Away' THEN tm.Points END), 2) AS
HomeAdvantagePoints,
    ROUND(AVG(CASE WHEN tm.VenueType = 'Home' THEN tm.GoalsScored END) -
          AVG(CASE WHEN tm.VenueType = 'Away' THEN tm.GoalsScored END), 2)
AS HomeAdvantageGoals,
    ROUND(SUM(CASE WHEN tm.VenueType = 'Home' THEN tm.Points ELSE 0 END) *
100.0 /
          (COUNT(CASE WHEN tm.VenueType = 'Home' THEN 1 END) * 3), 2) AS
```

```
HomePointsEfficiency,
    ROUND(SUM(CASE WHEN tm.VenueType = 'Away' THEN tm.Points ELSE 0 END) *
100.0 /
          (COUNT(CASE WHEN tm.VenueType = 'Away' THEN 1 END) * 3), 2) AS
AwayPointsEfficiency
FROM TeamMatches tm
JOIN Teams t ON tm.TeamID = t.TeamID
GROUP BY t.TeamName
HAVING COUNT(CASE WHEN tm.VenueType = 'Home' THEN 1 END) > 5
   AND COUNT(CASE WHEN tm.VenueType = 'Away' THEN 1 END) > 5
ORDER BY HomeAdvantagePoints DESC;
```

2. Enhanced Expected Goals Model

```
-- Team-specific expected goals model with shot quality consideration
WITH TeamShotEfficiency AS (
    SELECT
        m.MatchID,
        tm.TeamID,
        t.TeamName,
        CASE WHEN tm.TeamID = m.HomeTeamID THEN 'Home' ELSE 'Away' END AS
VenueType,
        CASE
            WHEN tm.TeamID = m.HomeTeamID THEN ms.HomeShotsTarget
            ELSE ms.AwayShotsTarget
        END AS ShotsOnTarget,
        CASE
            WHEN tm.TeamID = m.HomeTeamID THEN ms.HomeShots
            ELSE ms.AwayShots
        END AS TotalShots,
        CASE
            WHEN tm.TeamID = m.HomeTeamID THEN m.FTHG
            ELSE m.FTAG
        END AS GoalsScored,
        CASE
            WHEN tm.TeamID = m.HomeTeamID THEN 'H'
            WHEN m.FTR = 'D' THEN 'D'
            ELSE 'A'
        END AS Result
    FROM Matches m
    CROSS JOIN (
        SELECT m.HomeTeamID AS TeamID FROM Matches m
        UNION
        SELECT m.AwayTeamID AS TeamID FROM Matches m
    ) tm
```

```
    JOIN Teams t ON tm.TeamID = t.TeamID
    JOIN MatchStatistics ms ON m.MatchID = ms.MatchID
    WHERE (tm.TeamID = m.HomeTeamID OR tm.TeamID = m.AwayTeamID)
),
TeamAverages AS (
    SELECT
        TeamID,
        TeamName,
        VenueType,
        AVG(CASE WHEN ShotsOnTarget > 0 THEN GoalsScored * 1.0 /
ShotsOnTarget ELSE 0 END) AS AvgShotConversion,
        COUNT(*) AS MatchesPlayed
    FROM TeamShotEfficiency
    GROUP BY TeamID, TeamName, VenueType
)
SELECT
    tse.MatchID,
    tse.TeamName,
    tse.VenueType,
    tse.ShotsOnTarget,
    tse.TotalShots,
    tse.GoalsScored,
    ROUND(tse.ShotsOnTarget * ta.AvgShotConversion, 2) AS ExpectedGoals,
    CASE
        WHEN tse.GoalsScored > ROUND(tse.ShotsOnTarget *
ta.AvgShotConversion, 2) THEN 'Overperformed'
        WHEN tse.GoalsScored < ROUND(tse.ShotsOnTarget *
ta.AvgShotConversion, 2) THEN 'Underperformed'
        ELSE 'As Expected'
    END AS PerformanceVsExpected,
    ROUND(tse.GoalsScored - (tse.ShotsOnTarget * ta.AvgShotConversion), 2)
AS GoalDifferentialVsExpected
FROM TeamShotEfficiency tse
JOIN TeamAverages ta ON tse.TeamID = ta.TeamID AND tse.VenueType =
ta.VenueType
WHERE ta.MatchesPlayed >= 5
ORDER BY ABS(GoalDifferentialVsExpected) DESC;
```

3. Refined Referee Analysis

```
-- Enhanced referee tendencies with league average comparison
WITH RefereeSummary AS (
    SELECT
        r.RefereeID,
        r.RefereeName,
```

```
        COUNT(m.MatchID) AS MatchesRefereed,
        AVG(ms.HomeYellowCards) AS AvgHomeYellows,
        AVG(ms.AwayYellowCards) AS AvgAwayYellows,
        AVG(ms.HomeRedCards) AS AvgHomeReds,
        AVG(ms.AwayRedCards) AS AvgAwayReds,
        AVG(ms.HomeFouls) AS AvgHomeFouls,
        AVG(ms.AwayFouls) AS AvgAwayFouls,
        COUNT(CASE WHEN m.FTR = 'H' THEN 1 END) * 100.0 / COUNT(*) AS
HomeWinPct,
        COUNT(CASE WHEN m.FTR = 'A' THEN 1 END) * 100.0 / COUNT(*) AS
AwayWinPct,
        COUNT(CASE WHEN m.FTR = 'D' THEN 1 END) * 100.0 / COUNT(*) AS
DrawPct,
        AVG(ms.HomeYellowCards) - AVG(ms.AwayYellowCards) AS YellowCardBias,
        AVG(ms.HomeFouls) - AVG(ms.AwayFouls) AS FoulBias
    FROM Referees r
    JOIN Matches m ON r.RefereeID = m.RefereeID
    JOIN MatchStatistics ms ON m.MatchID = ms.MatchID
    GROUP BY r.RefereeID, r.RefereeName
    HAVING COUNT(m.MatchID) >= 10
),
LeagueAverages AS (
    SELECT
        AVG(ms.HomeYellowCards) AS LeagueAvgHomeYellows,
        AVG(ms.AwayYellowCards) AS LeagueAvgAwayYellows,
        AVG(ms.HomeRedCards) AS LeagueAvgHomeReds,
        AVG(ms.AwayRedCards) AS LeagueAvgAwayReds,
        AVG(ms.HomeFouls) AS LeagueAvgHomeFouls,
        AVG(ms.AwayFouls) AS LeagueAvgAwayFouls,
        COUNT(CASE WHEN m.FTR = 'H' THEN 1 END) * 100.0 / COUNT(*) AS
LeagueHomeWinPct,
        COUNT(CASE WHEN m.FTR = 'A' THEN 1 END) * 100.0 / COUNT(*) AS
LeagueAwayWinPct,
        COUNT(CASE WHEN m.FTR = 'D' THEN 1 END) * 100.0 / COUNT(*) AS
LeagueDrawPct
    FROM Matches m
    JOIN MatchStatistics ms ON m.MatchID = ms.MatchID
)
SELECT
    rs.RefereeName,
    rs.MatchesRefereed,
    ROUND(rs.AvgHomeYellows, 2) AS AvgHomeYellows,
    ROUND(la.LeagueAvgHomeYellows, 2) AS LeagueAvgHomeYellows,
    ROUND(rs.AvgHomeYellows - la.LeagueAvgHomeYellows, 2) AS HomeYellowDiff,
    ROUND(rs.AvgAwayYellows, 2) AS AvgAwayYellows,
    ROUND(la.LeagueAvgAwayYellows, 2) AS LeagueAvgAwayYellows,
```

```
    ROUND(rs.AvgAwayYellows - la.LeagueAvgAwayYellows, 2) AS AwayYellowDiff,
    ROUND(rs.HomeWinPct, 1) AS HomeWinPct,
    ROUND(la.LeagueHomeWinPct, 1) AS LeagueHomeWinPct,
    ROUND(rs.HomeWinPct - la.LeagueHomeWinPct, 1) AS HomeWinPctDiff,
    ROUND(rs.YellowCardBias, 2) AS YellowCardHomeBias,
    ROUND(rs.FoulBias, 2) AS FoulHomeBias,
    CASE
        WHEN ABS(rs.HomeWinPct - la.LeagueHomeWinPct) > 10 THEN
'Significant'
        WHEN ABS(rs.HomeWinPct - la.LeagueHomeWinPct) > 5 THEN 'Moderate'
        ELSE 'Normal'
    END AS HomeAdvantageEffect,
    CASE
        WHEN rs.YellowCardBias > 0.5 THEN 'Favors Away Team'
        WHEN rs.YellowCardBias < -0.5 THEN 'Favors Home Team'
        ELSE 'Neutral'
    END AS DisciplinaryBias
FROM RefereeSummary rs
CROSS JOIN LeagueAverages la
ORDER BY ABS(rs.HomeWinPct - la.LeagueHomeWinPct) DESC;
```

4. Normalized Bookmaker Analysis

```
-- Bookmaker comparison with improved odds analysis
WITH OddsAnalysis AS (
    SELECT
        m.MatchID,
        b.BookmakerID,
        b.BookmakerName,
        m.FTR AS ActualResult,
        bo.HomeWinOdds,
        bo.DrawOdds,
        bo.AwayWinOdds,
        CASE
            WHEN m.FTR = 'H' THEN bo.HomeWinOdds
            WHEN m.FTR = 'D' THEN bo.DrawOdds
            WHEN m.FTR = 'A' THEN bo.AwayWinOdds
        END AS WinningOdds,
        (1/bo.HomeWinOdds) AS ImpliedProbHome,
        (1/bo.DrawOdds) AS ImpliedProbDraw,
        (1/bo.AwayWinOdds) AS ImpliedProbAway,
        (1/bo.HomeWinOdds + 1/bo.DrawOdds + 1/bo.AwayWinOdds) AS
TotalImpliedProb,
        1 - (1/bo.HomeWinOdds + 1/bo.DrawOdds + 1/bo.AwayWinOdds) AS
BookmakerMargin
```

```
    FROM Matches m
    JOIN BettingOdds bo ON m.MatchID = bo.MatchID AND bo.MarketType =
'MatchResult' AND bo.IsClosingOdds = 1
    JOIN Bookmakers b ON bo.BookmakerID = b.BookmakerID
)
SELECT
    BookmakerName,
    COUNT(*) AS NumberOfMatches,
    ROUND(AVG(WinningOdds), 2) AS AvgWinningOdds,
    ROUND(AVG(BookmakerMargin) * 100, 2) AS AvgMarginPercent,
    ROUND(AVG(CASE WHEN ActualResult = 'H' THEN ImpliedProbHome ELSE 0 END)
/
          AVG(CASE WHEN ActualResult = 'H' THEN 1 ELSE 0 END), 4) AS
HomeWinProbAccuracy,
    ROUND(AVG(CASE WHEN ActualResult = 'D' THEN ImpliedProbDraw ELSE 0 END)
/
          AVG(CASE WHEN ActualResult = 'D' THEN 1 ELSE 0 END), 4) AS
DrawProbAccuracy,
    ROUND(AVG(CASE WHEN ActualResult = 'A' THEN ImpliedProbAway ELSE 0 END)
/
          AVG(CASE WHEN ActualResult = 'A' THEN 1 ELSE 0 END), 4) AS
AwayWinProbAccuracy,
    ROUND(SUM(CASE
        WHEN (ActualResult = 'H' AND ImpliedProbHome < 0.5) OR
             (ActualResult = 'D' AND ImpliedProbDraw < 0.3) OR
             (ActualResult = 'A' AND ImpliedProbAway < 0.5)
        THEN 1 ELSE 0 END) * 100.0 / COUNT(*), 2) AS UnderdogSuccessRate,
    ROUND(AVG(CASE
        WHEN (ActualResult = 'H' AND ImpliedProbHome < 0.5) OR
             (ActualResult = 'D' AND ImpliedProbDraw < 0.3) OR
             (ActualResult = 'A' AND ImpliedProbAway < 0.5)
        THEN WinningOdds ELSE NULL END), 2) AS AvgUnderdogWinningOdds
FROM OddsAnalysis
GROUP BY BookmakerName
HAVING COUNT(*) >= 50
ORDER BY AvgMarginPercent;
```

5. Enhanced Form and Momentum Analysis

```
-- Team form analysis with improved window functions
WITH TeamMatches AS (
    SELECT
        m.MatchID,
        m.MatchDate,
        m.HomeTeamID AS TeamID,
```

```
        'Home' AS VenueType,
        ht.TeamName,
        at.TeamName AS OpponentName,
        CASE
            WHEN m.FTR = 'H' THEN 'W'
            WHEN m.FTR = 'D' THEN 'D'
            ELSE 'L'
        END AS Result,
        CASE
            WHEN m.FTR = 'H' THEN 3
            WHEN m.FTR = 'D' THEN 1
            ELSE 0
        END AS Points,
        m.FTHG AS GoalsScored,
        m.FTAG AS GoalsConceded
    FROM Matches m
    JOIN Teams ht ON m.HomeTeamID = ht.TeamID
    JOIN Teams at ON m.AwayTeamID = at.TeamID

    UNION ALL

    SELECT
        m.MatchID,
        m.MatchDate,
        m.AwayTeamID AS TeamID,
        'Away' AS VenueType,
        at.TeamName,
        ht.TeamName AS OpponentName,
        CASE
            WHEN m.FTR = 'A' THEN 'W'
            WHEN m.FTR = 'D' THEN 'D'
            ELSE 'L'
        END AS Result,
        CASE
            WHEN m.FTR = 'A' THEN 3
            WHEN m.FTR = 'D' THEN 1
            ELSE 0
        END AS Points,
        m.FTAG AS GoalsScored,
        m.FTHG AS GoalsConceded
    FROM Matches m
    JOIN Teams ht ON m.HomeTeamID = ht.TeamID
    JOIN Teams at ON m.AwayTeamID = at.TeamID
),
TeamFormAnalysis AS (
    SELECT
```

```sql
        tm.MatchID,
        tm.TeamName,
        tm.OpponentName,
        tm.MatchDate,
        tm.VenueType,
        tm.Result,
        tm.Points,
        tm.GoalsScored,
        tm.GoalsConceded,
        -- Last 5 matches form
        SUM(tm.Points) OVER (
            PARTITION BY tm.TeamID
            ORDER BY tm.MatchDate
            ROWS BETWEEN 5 PRECEDING AND 1 PRECEDING
        ) AS Last5MatchesPoints,
        -- Last 5 matches goal difference
        SUM(tm.GoalsScored - tm.GoalsConceded) OVER (
            PARTITION BY tm.TeamID
            ORDER BY tm.MatchDate
            ROWS BETWEEN 5 PRECEDING AND 1 PRECEDING
        ) AS Last5MatchesGoalDiff,
        -- Last 5 matches consecutive results (simplified version)
        STRING_AGG(tm.Result) OVER (
            PARTITION BY tm.TeamID
            ORDER BY tm.MatchDate
            ROWS BETWEEN 5 PRECEDING AND 1 PRECEDING
        ) AS Last5Results,
        -- Home form (last 3 home matches)
        SUM(CASE WHEN tm.VenueType = 'Home' THEN tm.Points ELSE NULL END)
OVER (
            PARTITION BY tm.TeamID, tm.VenueType = 'Home'
            ORDER BY tm.MatchDate
            ROWS BETWEEN 3 PRECEDING AND 1 PRECEDING
        ) AS Last3HomeMatchesPoints,
        -- Away form (last 3 away matches)
        SUM(CASE WHEN tm.VenueType = 'Away' THEN tm.Points ELSE NULL END)
OVER (
            PARTITION BY tm.TeamID, tm.VenueType = 'Away'
            ORDER BY tm.MatchDate
            ROWS BETWEEN 3 PRECEDING AND 1 PRECEDING
        ) AS Last3AwayMatchesPoints
    FROM TeamMatches tm
)
SELECT
    tfa.MatchID,
    tfa.TeamName,
```

```
    tfa.OpponentName,
    tfa.MatchDate,
    tfa.VenueType,
    tfa.Result,
    tfa.Points,
    tfa.GoalsScored,
    tfa.GoalsConceded,
    tfa.Last5MatchesPoints,
    tfa.Last5MatchesGoalDiff,
    tfa.Last5Results,
    CASE
        WHEN tfa.Last5MatchesPoints >= 13 THEN 'Excellent'
        WHEN tfa.Last5MatchesPoints >= 10 THEN 'Very Good'
        WHEN tfa.Last5MatchesPoints >= 7 THEN 'Good'
        WHEN tfa.Last5MatchesPoints >= 4 THEN 'Average'
        WHEN tfa.Last5MatchesPoints >= 1 THEN 'Poor'
        ELSE 'Very Poor'
    END AS FormRating,
    CASE
        WHEN tfa.VenueType = 'Home' AND tfa.Last3HomeMatchesPoints IS NOT
NULL THEN tfa.Last3HomeMatchesPoints
        WHEN tfa.VenueType = 'Away' AND tfa.Last3AwayMatchesPoints IS NOT
NULL THEN tfa.Last3AwayMatchesPoints
        ELSE NULL
    END AS RecentVenueFormPoints,
    CASE
        WHEN tfa.Result = 'W' AND tfa.Last5MatchesPoints <= 4 THEN 'Upset
Win'
        WHEN tfa.Result = 'L' AND tfa.Last5MatchesPoints >= 10 THEN 'Upset
Loss'
        WHEN tfa.Last5MatchesPoints >= 10 AND tfa.Result = 'W' THEN
'Expected Win'
        WHEN tfa.Last5MatchesPoints <= 4 AND tfa.Result = 'L' THEN 'Expected
Loss'
        ELSE 'Regular Result'
    END AS ResultContext
FROM TeamFormAnalysis tfa
WHERE tfa.Last5MatchesPoints IS NOT NULL
ORDER BY tfa.MatchDate DESC, tfa.TeamName;
```

6. Dynamic League Table Generator (Stored Procedure)

```
-- Dynamic league table generator with improved metrics
CREATE PROCEDURE GenerateLeagueTable(IN seasonId INT, IN cutoffDate DATE)
BEGIN
```

```
    WITH MatchResults AS (
        -- Home team results
        SELECT
            m.HomeTeamID AS TeamID,
            1 AS MatchesPlayed,
            CASE WHEN m.FTR = 'H' THEN 1 ELSE 0 END AS Wins,
            CASE WHEN m.FTR = 'D' THEN 1 ELSE 0 END AS Draws,
            CASE WHEN m.FTR = 'A' THEN 1 ELSE 0 END AS Losses,
            m.FTHG AS GoalsFor,
            m.FTAG AS GoalsAgainst,
            CASE WHEN m.FTR = 'H' THEN 3 WHEN m.FTR = 'D' THEN 1 ELSE 0 END
AS Points,
            ms.HomeShots AS Shots,
            ms.HomeShotsTarget AS ShotsOnTarget,
            ms.HomeCorners AS Corners,
            ms.HomeYellowCards AS YellowCards,
            ms.HomeRedCards AS RedCards
        FROM Matches m
        JOIN MatchStatistics ms ON m.MatchID = ms.MatchID
        WHERE m.SeasonID = seasonId AND m.MatchDate <= cutoffDate

        UNION ALL

        -- Away team results
        SELECT
            m.AwayTeamID AS TeamID,
            1 AS MatchesPlayed,
            CASE WHEN m.FTR = 'A' THEN 1 ELSE 0 END AS Wins,
            CASE WHEN m.FTR = 'D' THEN 1 ELSE 0 END AS Draws,
            CASE WHEN m.FTR = 'H' THEN 1 ELSE 0 END AS Losses,
            m.FTAG AS GoalsFor,
            m.FTHG AS GoalsAgainst,
            CASE WHEN m.FTR = 'A' THEN 3 WHEN m.FTR = 'D' THEN 1 ELSE 0 END
AS Points,
            ms.AwayShots AS Shots,
            ms.AwayShotsTarget AS ShotsOnTarget,
            ms.AwayCorners AS Corners,
            ms.AwayYellowCards AS YellowCards,
            ms.AwayRedCards AS RedCards
        FROM Matches m
        JOIN MatchStatistics ms ON m.MatchID = ms.MatchID
        WHERE m.SeasonID = seasonId AND m.MatchDate <= cutoffDate
    ),
    TeamStats AS (
        SELECT
            t.TeamID,
```

```
            t.TeamName,
            SUM(mr.MatchesPlayed) AS Played,
            SUM(mr.Wins) AS Won,
            SUM(mr.Draws) AS Drawn,
            SUM(mr.Losses) AS Lost,
            SUM(mr.GoalsFor) AS GoalsFor,
            SUM(mr.GoalsAgainst) AS GoalsAgainst,
            SUM(mr.Points) AS Points,
            SUM(mr.Shots) AS TotalShots,
            SUM(mr.ShotsOnTarget) AS TotalShotsOnTarget,
            SUM(mr.Corners) AS TotalCorners,
            SUM(mr.YellowCards) AS TotalYellowCards,
            SUM(mr.RedCards) AS TotalRedCards,
            ROUND(SUM(mr.ShotsOnTarget) * 100.0 / NULLIF(SUM(mr.Shots), 0),
1) AS ShotAccuracy,
            ROUND(SUM(mr.GoalsFor) * 100.0 / NULLIF(SUM(mr.ShotsOnTarget),
0), 1) AS ConversionRate,
            ROUND(SUM(mr.Points) * 100.0 / (SUM(mr.MatchesPlayed) * 3), 1)
AS PointsPercentage
        FROM Teams t
        JOIN MatchResults mr ON t.TeamID = mr.TeamID
        GROUP BY t.TeamID, t.TeamName
    )
    SELECT
        ROW_NUMBER() OVER (ORDER BY Points DESC, (GoalsFor - GoalsAgainst)
DESC, GoalsFor DESC) AS Position,
        TeamName,
        Played,
        Won,
        Drawn,
        Lost,
        GoalsFor,
        GoalsAgainst,
        (GoalsFor - GoalsAgainst) AS GoalDifference,
        Points,
        ROUND(Points * 1.0 / Played, 2) AS PointsPerGame,
        PointsPercentage AS PointsEfficiency,
        ShotAccuracy,
        ConversionRate,
        ROUND(GoalsFor * 1.0 / Played, 2) AS GoalsScoredPerGame,
        ROUND(GoalsAgainst * 1.0 / Played, 2) AS GoalsConcededPerGame,
        CASE
            WHEN Played >= 10 AND PointsPercentage > 70 THEN 'Title
Contender'
            WHEN Played >= 10 AND PointsPercentage > 60 THEN 'Top 4
Contender'
```

```
            WHEN Played >= 10 AND PointsPercentage > 50 THEN 'European
Spots'
            WHEN Played >= 10 AND PointsPercentage > 40 THEN 'Mid-table'
            WHEN Played >= 10 AND PointsPercentage > 30 THEN 'Relegation
Battle'
            WHEN Played >= 10 THEN 'Relegation Candidate'
            ELSE 'Too Early'
        END AS CurrentStatus
    FROM TeamStats
    ORDER BY Points DESC, (GoalsFor - GoalsAgainst) DESC, GoalsFor DESC;
END;
```

7. Match Result Prediction Model

```
-- Match prediction model based on team form and head-to-head history
CREATE PROCEDURE PredictMatchResult(IN homeTeamId INT, IN awayTeamId INT, IN
matchDate DATE)
BEGIN
    -- Get team form (last 5 matches)
    WITH TeamForm AS (
        SELECT
            TeamID,
            AVG(PointsEarned) AS AvgPoints,
            AVG(GoalsScored) AS AvgGoalsScored,
            AVG(GoalsConceded) AS AvgGoalsConceded,
            SUM(CASE WHEN Result = 'W' THEN 1 ELSE 0 END) AS WinsLast5,
            SUM(CASE WHEN Result = 'D' THEN 1 ELSE 0 END) AS DrawsLast5,
            SUM(CASE WHEN Result = 'L' THEN 1 ELSE 0 END) AS LossesLast5
        FROM TeamMatchStatsFact
        WHERE MatchDate < matchDate
        GROUP BY TeamID
        HAVING COUNT(*) >= 5
    ),
    -- Home team specific form
    HomeForm AS (
        SELECT
            TeamID,
            AVG(PointsEarned) AS HomeAvgPoints,
            AVG(GoalsScored) AS HomeAvgGoalsScored,
            AVG(GoalsConceded) AS HomeAvgGoalsConceded
        FROM TeamMatchStatsFact
        WHERE MatchDate < matchDate AND IsHome = 1
        GROUP BY TeamID
        HAVING COUNT(*) >= 3
    ),
```

```
    -- Away team specific form
    AwayForm AS (
        SELECT
            TeamID,
            AVG(PointsEarned) AS AwayAvgPoints,
            AVG(GoalsScored) AS AwayAvgGoalsScored,
            AVG(GoalsConceded) AS AwayAvgGoalsConceded
        FROM TeamMatchStatsFact
        WHERE MatchDate < matchDate AND IsHome = 0
        GROUP BY TeamID
        HAVING COUNT(*) >= 3
    ),
    -- Head-to-head results
    HeadToHead AS (
        SELECT
            COUNT(*) AS TotalMatches,
            SUM(CASE WHEN (HomeTeamID = homeTeamId AND Result = 'W') OR
(AwayTeamID = homeTeamId AND Result = 'L') THEN 1 ELSE 0 END) AS
HomeTeamWins,
            SUM(CASE WHEN Result = 'D' THEN 1 ELSE 0 END) AS Draws,
            SUM(CASE WHEN (HomeTeamID = awayTeamId AND Result = 'W') OR
(AwayTeamID = awayTeamId AND Result = 'L') THEN 1 ELSE 0 END) AS
AwayTeamWins,
            AVG(CASE WHEN HomeTeamID = homeTeamId THEN HomeGoals ELSE
AwayGoals END) AS AvgHomeTeamGoals,
            AVG(CASE WHEN HomeTeamID = awayTeamId THEN HomeGoals ELSE
AwayGoals END) AS AvgAwayTeamGoals
        FROM Matches
        WHERE (HomeTeamID = homeTeamId AND AwayTeamID = awayTeamId) OR
(HomeTeamID = awayTeamId AND AwayTeamID = homeTeamId)
        AND MatchDate < matchDate
    )
    -- Calculate prediction
    SELECT
        ht.TeamName AS HomeTeam,
        at.TeamName AS AwayTeam,
        tf_home.AvgPoints AS HomeTeamFormPoints,
        tf_away.AvgPoints AS AwayTeamFormPoints,
        hf.HomeAvgPoints AS HomeTeamHomeFormPoints,
        af.AwayAvgPoints AS AwayTeamAwayFormPoints,
        hf.HomeAvgGoalsScored AS HomeTeamAvgHomeGoalsScored,
        af.AwayAvgGoalsScored AS AwayTeamAvgAwayGoalsScored,
        h2h.HomeTeamWins AS HomeTeamH2HWins,
        h2h.Draws AS H2HDraws,
        h2h.AwayTeamWins AS AwayTeamH2HWins,
```

```
        -- Simple prediction model components
        0.4 * (hf.HomeAvgPoints / 3) + -- Home team home form (40% weight)
        0.3 * (1 - (af.AwayAvgPoints / 3)) + -- Away team away form inverted
(30% weight)
        0.15 * (tf_home.AvgPoints - tf_away.AvgPoints) / 3 + -- General form
difference (15% weight)
        0.15 * CASE -- Head-to-head history (15% weight)
            WHEN h2h.TotalMatches > 0 THEN h2h.HomeTeamWins /
h2h.TotalMatches
            ELSE 0.5 -- Default to 50% if no history
        END AS HomeWinProbability,

        -- Expected goals prediction
        ROUND(hf.HomeAvgGoalsScored * 0.7 + af.AwayAvgGoalsConceded * 0.3,
1) AS PredictedHomeGoals,
        ROUND(af.AwayAvgGoalsScored * 0.7 + hf.HomeAvgGoalsConceded * 0.3,
1) AS PredictedAwayGoals,

        -- Most likely result
        CASE
            WHEN (0.4 * (hf.HomeAvgPoints / 3) + 0.3 * (1 -
(af.AwayAvgPoints / 3)) + 0.15 * (tf_home.AvgPoints - tf_away.AvgPoints) / 3
+ 0.15 * CASE WHEN h2h.TotalMatches > 0 THEN h2h.HomeTeamWins /
h2h.TotalMatches ELSE 0.5 END) > 0.55 THEN 'Home Win'
            WHEN (0.4 * (hf.HomeAvgPoints / 3) + 0.3 * (1 -
(af.AwayAvgPoints / 3)) + 0.15 * (tf_home.AvgPoints - tf_away.AvgPoints) / 3
+ 0.15 * CASE WHEN h2h.TotalMatches > 0 THEN h2h.HomeTeamWins /
h2h.TotalMatches ELSE 0.5 END) < 0.45 THEN 'Away Win'
            ELSE 'Draw'
        END AS PredictedResult
    FROM Teams ht
    JOIN Teams at ON ht.TeamID = homeTeamId AND at.TeamID = awayTeamId
    LEFT JOIN TeamForm tf_home ON tf_home.TeamID = homeTeamId
    LEFT JOIN TeamForm tf_away ON tf_away.TeamID = awayTeamId
    LEFT JOIN HomeForm hf ON hf.TeamID = homeTeamId
    LEFT JOIN AwayForm af ON af.TeamID = awayTeamId
    LEFT JOIN HeadToHead h2h ON 1=1;
END;
```

Database Indexing Strategy

To ensure optimal query performance, we'll implement a comprehensive indexing strategy:

```
-- Primary Keys (automatically indexed)
-- Foreign Keys (need explicit indexing)
```

```
CREATE INDEX idx_matches_season ON Matches(SeasonID);
CREATE INDEX idx_matches_teams ON Matches(HomeTeamID, AwayTeamID);
CREATE INDEX idx_matches_referee ON Matches(RefereeID);
CREATE INDEX idx_matches_date ON Matches(MatchDate);

-- Match Statistics Table
CREATE INDEX idx_matchstats_match ON MatchStatistics(MatchID);

-- Betting Odds Table
CREATE INDEX idx_bettingodds_match ON BettingOdds(MatchID);
CREATE INDEX idx_bettingodds_bookmaker ON BettingOdds(BookmakerID);
CREATE INDEX idx_bettingodds_market ON BettingOdds(MarketType);
CREATE INDEX idx_bettingodds_match_bookmaker ON BettingOdds(MatchID,
BookmakerID);

-- Analytical Database Fact Tables
CREATE INDEX idx_teammatchstats_match ON TeamMatchStatsFact(MatchID);
CREATE INDEX idx_teammatchstats_team ON TeamMatchStatsFact(TeamID);
CREATE INDEX idx_teammatchstats_team_ishome ON TeamMatchStatsFact(TeamID,
IsHome);
CREATE INDEX idx_teammatchstats_referee ON TeamMatchStatsFact(RefereeID);
CREATE INDEX idx_teammatchstats_time ON TeamMatchStatsFact(TimeID);

CREATE INDEX idx_bettingmarket_match ON BettingMarketFact(MatchID);
CREATE INDEX idx_bettingmarket_bookmaker ON BettingMarketFact(BookmakerID);
CREATE INDEX idx_bettingmarket_market ON BettingMarketFact(MarketType);
```

ETL Process

1. **Data Import and Cleaning**
   - Import historical CSV files
   - Handle missing values (especially in betting odds)
   - Standardize team names across seasons
   - Validate and correct anomalous data points
2. **Operational Database Loading**
   - Create and populate Teams table with team metadata
   - Load Seasons table with distinct season data
   - Populate Matches table with match results and metadata
   - Extract and store MatchStatistics separately
   - Normalize bookmaker data into Bookmakers and BettingOdds tables
3. **Analytical Database Transformation**
   - Create denormalized TeamMatchStatsFact records (one per team per match)

- Calculate derived metrics (shot accuracy, expected goals, etc.)
- Transform betting odds into analytical metrics (implied probabilities, margins)
- Build time dimension with appropriate hierarchies

4. **Incremental Update Strategy**
   - Design process for adding new match data each week
   - Implement recalculation of derived metrics
   - Enable historical data corrections

Use Case Summary Table

Use Case	SQL Objects Involved	Business Value
Team Performance Analysis	TeamMatchStatsFact, Teams	Evaluate and compare team performance at home vs. away
Expected Goals Model	TeamMatchStatsFact, MatchStatistics	Identify over/underperforming teams based on shot quality
Referee Analysis	Referees, Matches, MatchStatistics	Detect bias patterns and impact on match outcomes
Bookmaker Comparison	BettingOdds, Bookmakers, Matches	Identify value bets and market inefficiencies
Form and Momentum Tracking	TeamMatchStatsFact, Matches	Analyze team form patterns and forecast performance trends
League Table Generator	Matches, Teams, MatchStatistics	Create dynamic league tables for any point in a season
Match Prediction	TeamMatchStatsFact, Matches	Predict outcomes of future matches based on form and history

Application Design

Key Interface Components

1. **Team Dashboard**
   - Performance metrics (home/away)
   - Form tracker
   - Upcoming fixtures with predictions
2. **Match Analysis Screen**
   - Head-to-head history
   - Current form comparison

- Expected goals visualization
- Betting odds comparison

3. **League Table View**
   - Standard table with points, GD
   - Enhanced metrics (shot accuracy, efficiency)
   - Form-based coloring
   - Historical position tracking
4. **Prediction Engine**
   - Match outcome prediction tool
   - Confidence indicators
   - Key factors influencing prediction
5. **Admin Interface**
   - Data import/validation
   - Correction tools
   - Database maintenance functions

Project Timeline and Phases

1. **Database Design** (Week 1-2)
   - Normalized schema for operational DB
   - Star schema for analytical DB
   - Indexing strategy
2. **ETL Development** (Week 3-4)
   - Data cleaning scripts
   - Transformation logic
   - Historical data loading
3. **Core Analytics** (Week 5-6)
   - Key SQL queries
   - Stored procedures
   - Performance optimization
4. **Frontend Development** (Week 7-8)
   - GUI design
   - Data visualization
   - Report generation
5. **Testing & Documentation** (Week 9-10)
   - Query validation
   - Performance testing

- User guide creation

This Premier League Analytics project combines robust database design with sophisticated SQL analytics while providing meaningful football insights through statistical models and betting market analysis.