Cricket Players and Teams Database Management System

Devansh Kukadia - 202303030
Frinad Kandoriya - 202303044
Yash Shah - 202303004

➢ Functional Depenedencies:

1. Sponsorships Table:

- `SponsorID → SponsorName, Amount, StartDate, EndDate`
- The sponsor's ID uniquely determines the sponsor's name, amount, start date, and end date.

2. Coaches Table:

- `CoachID → HeadCoach, BowlingCoach, BattingCoach, FieldingCoach`
- The coach's ID uniquely determines the coaches for different roles.

3. GoverningBody Table:

- `GoverningBodyName → President, VicePresident, Secretary, Treasurer, Revenue, Headquarters`
- The governing body name uniquely determines its organizational structure, revenue, and headquarters.

4. Team Table:

- `TeamName → CoachID, GoverningBodyName, YearOfFoundation, NumberOfChampionships, TeamODIRank, TeamT20Rank, TeamTestRank`
- The team name uniquely determines the associated coach, governing body, foundation year, championships, and team rankings.

5. **Player Table:**
   - `PlayerID → FullName, DateOfBirth, Nationality, Role, BattingStyle, BowlingStyle, MatchesPlayed, Runs, Wickets, Average, StrikeRate, Economy, PlayerODIRank, PlayerT20Rank, PlayerTestRank, TeamName`
   - The player ID uniquely determines all other player attributes and their associated team.

6. **Venue Table:**
   - `StadiumName → City, Capacity`
   - The stadium name uniquely determines the city and capacity.

7. **Match Table:**
   a. `MatchID → Team1Name, Team2Name, StadiumName, InChampID, WinnerTeamName, POTMPlayerID, Date, Scorecard, Result`
   - The match ID uniquely determines the participating teams, venue, championship, winner, player of the match, date, scorecard, and result.

   b. `Team1Name, Team2Name → MatchID`
   - A combination of Team1 and Team2 uniquely identifies the match.

   c. `WinnerTeamName → MatchID`
   - The winner of the match determines the match.

8. **Umpire Table:**
   - `UmpireID → FullName, DateOfBirth, Nationality, Matches`
   - The umpire ID uniquely determines the umpire's personal information and number of matches officiated.

9. **Performance Table:**
   - `PlayerID, MatchID → RunsScored, WicketsTaken, StrikeRate, Economy, CatchesTaken, RunOuts`
   - The combination of a player and match ID uniquely determines the player's performance in the match.

10. **International Championship Table:**
   - `InChampID → StartDate, EndDate, PastWinners, Format`
   - The international championship ID uniquely determines the start date, end date, past winners, and the format of the championship.

➢ List of INSERT, UPDATE and DELETE Anomalies:

1. UPDATE anomalies:

- Team Name Change:
    - If a player’s team changes its name, it must be updated in all occurrences (Player, Match, Team). Failing to do so may result in inconsistent team names across the database.
- Coach Information:
    - If a coach's information (e.g., name or nationality) changes, it must be updated in multiple places if coach data is stored redundantly. Any missed update leads to inconsistent data.
- Player Information:
    - If a player’s personal information (e.g., mobile number or email) changes, it needs to be updated wherever the player is referenced. Redundant storage of player data increases the chances of inconsistency.
- Sponsor Change:
    - Changing a sponsor’s name or details requires updating all rows related to that sponsor (Team, Match, Sponsorship tables). Missing an update in one place can lead to inconsistent sponsor information.

2. DELETE anomalies:

- Player Deletion:
    - Deleting a player might unintentionally delete information related to their team or matches they played, leading to loss of vital historical data (e.g., match performance or team membership).
- Team Deletion:
    - If a team is deleted, the matches and other records that are associated with the team might also be deleted inadvertently, leading to loss of match data or performance statistics.

- Sponsor Deletion:
    - Deleting a sponsor might cause the removal of teams or other data linked to the sponsor, creating a problem when you want to retain team information but lose sponsorship details.
- Match Deletion:
    - Deleting a match might also remove information about players’ performances, leading to the loss of critical performance history.

3. INSERT anomalies:

- Adding a Player:
  - If team information (e.g., TeamName) is required to insert a player, it might not be possible to add a player until the team information is available, leading to dependency issues.
- Adding a Match:
  - Inserting a match might require that both Team1 and Team2 exist in the database, forcing you to insert teams prematurely or with incomplete information just to create a match record.
- Adding a Coach:
  - If a team cannot exist in the database without having a coach, adding a team without coach information would be impossible, creating an insertion anomaly. This could lead to the need for inserting placeholder or null values.
- Adding a Sponsor:
  - Inserting a sponsor might require team and player details to already exist, creating dependencies where sponsor details can't be added independently.

➢ List of Redundancies:

1. Sponsorship Redundancies:

- **Multiple sponsorship records** could exist for the same sponsor across various teams or players. For example, if a sponsor supports multiple teams or players, details like `SponsorName`, `Amount`, `StartDate`, and `EndDate` might be repeated for each sponsorship. This redundancy can be avoided by normalizing the sponsorship table or using junction tables that link sponsors to multiple entities (teams/players) without repeating the sponsor details.

2. Coach Redundancies:

- **Coach information (CoachID, names)** is related to multiple teams. If a coach is associated with several teams, their details may be duplicated in those teams. For example, the same `HeadCoach`, `BowlingCoach`, `BattingCoach`, or `FieldingCoach` could be associated with multiple teams, leading to redundancy.

3. Player Information Redundancy in Performance Table:

- The `PlayerID` **in the Performance table** refers to the player's identity. If certain performance-related information (such as `FullName`, `Role`, or `Nationality`) is being stored repeatedly for each performance, then it's redundant, as player details are already stored in the **Player** table.
- Instead, the performance should only capture the **specific statistics** for that match (e.g., `RunsScored`, `WicketsTaken`), with player details referred to via foreign keys.

4. Team Information in the Match Table:

- **Team information (TeamName)** is stored in both the **Match** and **Team** tables. For each match, you need to reference two teams, but storing full team information in both places would be redundant. If the same information (like `YearOfFoundation`, `Rankings`) is repeatedly stored in the Match table or somewhere else, it should only reside in the Team table.

5. Umpire Information:

- If **umpire details** (like their name, nationality, or date of birth) are repeated across multiple matches in the **Match-Umpire relationship**, it would result in redundancy. Since umpires are often associated with multiple matches, these details should only exist in the **Umpire** table and be referenced using foreign keys.

6. Venue Redundancy:

- Venue information like the stadium name, city, or capacity is tied to multiple matches. If this information is repeated for each match (i.e., if match records store venue details instead of a reference to the venue), it would lead to redundancy. Instead, match records should only store a reference (foreign key) to the Venue table, which holds the actual venue details.

7. Governing Body Redundancy:

- **Governing body details** like the name, president, and headquarters are related to multiple teams. If this information is repeated for each team or across other entities, it results in redundancy. The **GoverningBodyName** field should reference only unique governing body records.

8. Player Rankings:

- **Player rankings (ODI Rank, T20 Rank, Test Rank)** are stored in the Player table. If ranking information is stored multiple times across other entities (e.g., in Performance or Match tables), it would be redundant. Rankings should only be updated in the **Player** table and referenced elsewhere.

9. Match Results Redundancy:

- **Winner information** (which team won a match) and the **Player of the Match** (`POTM`) details could be redundant if stored in more than one place. If this information is repeated elsewhere, it should instead just reference the relevant **Match** record where the result and Player of the Match are already captured.

10. International Championship Data:

- If details about the **teams participating** in a championship are repeated across multiple match records, it can result in redundancy. The International Championship entity should have references to the teams participating without storing redundant details like team names or rankings in every associated match.

➢ Normal Forms:

1. Sponsorships Table:

- **Attributes:** `SponsorID (PK), SponsorName, Amount, StartDate, EndDate`
- **Primary Key:** `SponsorID`
- **1NF:** Yes, all values are atomic.
- **2NF:** Yes, as the table has a single primary key (`SponsorID`), there can be no partial dependencies.
- **3NF:** Yes, no transitive dependencies exist. All non-prime attributes depend directly on the primary key.

- **BCNF:** Yes, `SponsorID` is a superkey, and no other attribute determines anything else.

2. Coaches Table:

- **Attributes:** `CoachID (PK), HeadCoach, BowlingCoach, BattingCoach, FieldingCoach`
- **Primary Key:** `CoachID`
- **1NF:** Yes, all values are atomic.
- **2NF:** Yes, there is a single key (`CoachID`), so no partial dependencies.
- **3NF:** Yes, all attributes are dependent on the primary key, no transitive dependencies.
- **BCNF:** Yes, `CoachID` is a superkey, and no other attribute determines anything else.

3. GoverningBody Table:

- **Attributes:** `GoverningBodyName (PK), President, VicePresident, Secretary, Treasurer, Revenue, Headquarters`
- **Primary Key:** `GoverningBodyName`
- **1NF:** Yes, all values are atomic.
- **2NF:** Yes, as there is a single primary key (`GoverningBodyName`), there can be no partial dependencies.
- **3NF:** Yes, no transitive dependencies are present.
- **BCNF:** Yes, `GoverningBodyName` is a superkey, and no other attributes functionally determine any other attributes.

4. Team Table:

- **Attributes:** `TeamName (PK), CoachID (FK), GoverningBodyName (FK), YearOfFoundation, NumberOfChampionships, TeamODIRank, TeamT20Rank, TeamTestRank`
- **Primary Key:** `TeamName`
- **1NF:** Yes, all values are atomic.
- **2NF:** Yes, the table has a single primary key, so no partial dependencies.
- **3NF:** Yes, all attributes are fully dependent on the primary key.
- **BCNF:** Yes, `TeamName` is a superkey, and no other attributes (e.g., `CoachID`, `GoverningBodyName`) functionally determine other attributes.

5. Player Table:

- **Attributes:** `PlayerID (PK), FullName, DateOfBirth, Nationality, Role, BattingStyle, BowlingStyle, MatchesPlayed, Runs, Wickets, Average, StrikeRate, Economy, PlayerODIRank, PlayerT20Rank, PlayerTestRank, TeamName (FK)`
- **Primary Key:** `PlayerID`
- **1NF:** Yes, all values are atomic.
- **2NF:** Yes, there is no partial dependency (since there is only one key, `PlayerID`).
- **3NF:** Yes, all attributes depend directly on the primary key `PlayerID`. `TeamName` is a foreign key but does not introduce transitive dependencies since it is related to a different entity.
- **BCNF:** Yes, `PlayerID` is a superkey, and no other attributes (e.g., `TeamName`) functionally determine other attributes.

6. Venue Table:

- **Attributes:** `StadiumName (PK), City, Capacity`
- **Primary Key:** `StadiumName`
- **1NF:** Yes, all values are atomic.
- **2NF:** Yes, since `StadiumName` is the primary key, there are no partial dependencies.
- **3NF:** Yes, all attributes are fully dependent on the primary key, and there are no transitive dependencies.
- **BCNF:** Yes, `StadiumName` is a superkey, and no other attributes determine other attributes.

7. Match Table:

- **Attributes:** `MatchID (PK), Team1Name (FK), Team2Name (FK), StadiumName (FK), InChampID (FK), WinnerTeamName (FK), POTMPlayerID (FK), Date, Scorecard, Result`
- **Primary Key:** `MatchID`
- **1NF:** Yes, all values are atomic.
- **2NF:** Yes, since the table has a single primary key (`MatchID`), there can be no partial dependencies.
- **3NF:** Yes, all non-key attributes are dependent on the primary key, and there are no transitive dependencies.
- **BCNF:** Yes, `MatchID` is a superkey, and no other attributes determine anything else.

8. Umpire Table:

- **Attributes:** `UmpireID (PK), FullName, DateOfBirth, Nationality, Matches`
- **Primary Key:** `UmpireID`
- **1NF:** Yes, all values are atomic.
- **2NF:** Yes, there is no partial dependency.
- **3NF:** Yes, no transitive dependencies exist.
- **BCNF:** Yes, `UmpireID` is a superkey, and no other attributes determine other attributes.

9. Performance Table:

- **Attributes:** `PlayerID (PK, FK), MatchID (PK, FK), RunsScored, WicketsTaken, StrikeRate, Economy, CatchesTaken, RunOuts`
- **Primary Key:** Composite key (`PlayerID`, `MatchID`)
- **1NF:** Yes, all values are atomic.
- **2NF:** Yes, since the composite key (`PlayerID`, `MatchID`) determines all non-prime attributes, there are no partial dependencies.
- **3NF:** Yes, all attributes are fully dependent on the composite key, and no transitive dependencies exist.
- **BCNF:** Yes, the composite key (`PlayerID`, `MatchID`) is the superkey, and no other functional dependencies violate BCNF.

10. International Championship Table:

- **Attributes:** `InChampID (PK), StartDate, EndDate, PastWinners, Format, Teams`
- **Primary Key:** `InChampID`
- **1NF:** Yes, all values are atomic.
- **2NF:** Yes, the single primary key `InChampID` determines all non-prime attributes.
- **3NF:** Yes, no transitive dependencies exist.
- **BCNF:** Yes, `InChampID` is a superkey, and no other attributes determine other attributes.

Conclusion: All Relations Are in BCNF

- **Reason:**
  - Every table satisfies 1NF, 2NF, and 3NF.
  - For each functional dependency `X → Y` in the tables, the left-hand side (X) is a **superkey**.
  - There are no cases of partial dependencies or transitive dependencies in any of the relations.
  - All attributes are fully dependent on their primary or composite keys, and no non-prime attributes are functionally determined by other non-prime attributes.

Why No Further Normalization Is Needed:

1. **No anomalies exist**: Since all relations are in **BCNF**, they do not suffer from any insert, update, or delete anomalies that arise from functional dependencies.
2. **Functional Dependencies are respected**: All non-key attributes are functionally dependent on the whole key, meaning there are no unnecessary redundancies or dependencies to break down further.
3. **BCNF is sufficient**: As BCNF is a stricter form of 3NF, achieving BCNF ensures the highest level of data integrity without needing further normalization. All functional dependencies have superkeys as their determinants, which means there are no violation cases that would require further refinement or decomposition.