Phase - 1, Data and Applications Project

Kunal Jain	**2019111037**
Samay Kothari	**2019113017**
Tejasvi Chebrolu	**2019114005**

Mini World:

Our mini-world is IPL where we are planning to implement a database for the IPL matches for the ongoing season of the tournament along with legacy information about all the teams, matches and players. We are storing the information about the matches, players, teams, etc. for the same. There will be different types of stats and information stored about the tournament which will be very useful for its users.

The users of the database will be the cricket fans across the world who love the cricket stats and also the management of the whole tournament. For example a fan may want to know the scorecard and schedule of the matches of his/her favourite team or a manager of the tournament may want to know the maximum run scorer for the season and various other stats for smooth functioning of the tournament.

Assumptions

We assume that all teams have 11 players during a season (i.e. no reserves), all of whom play all the matches for that team during the season. We also assume there are no injuries or cancelled matches.
We assume the database is going to be used for practical purposes only and since there can only be one season running at any given point of time, we reset all the information pertaining to the current season in our database whenever a new season is entered.

We assume the tournament to have only the round robin stage.

Database

Inserts/Updates to Seasons:
At the start of the season, it is required to declare a year and non mutable list of teams.
Whenever a new entry is made to this entity, all the attributes in other tables pertaining to the current season are reset to default.
Whenever there is an update to ‘Result’, update ‘Orange Cap’ and ‘Purple. cap’ by querying max ‘Current runs’ and ‘Current wickets’ in ‘Player’ table.

Inserts/Updates to Teams:
Initialized with ‘Name’ and ‘Home Ground’.
Team must have a unique ‘Name’.
‘Player’, ‘Match’ and ‘Season results’ list initially empty. ‘Current captain’ and ‘Current Management’ set to null.
Update players as they are added or removed from the team.
Match can be updated only if management, captain are not null.
Season results can be updated only if the boolean ‘Finished’ of the season entity is false.

Inserts/Updates to Match:
First insert all the details of the match
Update ‘Matches’ attribute of both the teams
Update points in ‘Result’ attribute of season
Update ‘Matches played’, ‘Runs’, ‘Wickets’, ‘MoM’, ‘Current runs’ and ‘Current wickets’ of each player in the match.

Inserts/Updates to Player:
Initialized with ‘Name’ and ‘Age’.

‘Matches played’, ‘Runs’, ‘Wickets’, ‘MoM’, ‘Current runs’ and ‘Current wickets’ are initialized with 0. ‘Current team ID’ is initialized with null. Whenever ‘Current team ID’ is updated, if the previous value wasn’t null,

Entities:

1. Teams
    a. Current management team (Weak type entity)
2. Players
3. Seasons
4. Matches
    a. Scorecard of the match(Weak type entity)
5. Stadium

Attributes:

Teams			
Attribute	**Type**	**Constraints**	**Note**
ID	str	Required	Surrogate primary key
Name	str	Required	Secondary key
Season results	[int, int]	Default value: [] Season year must be in ‘Seasons’	[Season year, position]
Home ground	str	Required	Ground ID from ‘Stadium’

Players			
Attribute	**Type**	**Constraints**	**Note**

ID	str	Required	Surrogate primary key
Name	str	Required	
Age	int	Required Age >= 0	
Speciality	str	“Batsman”/”Bowler”/”All rounder”	(Subclass)
Matches played	int	Default value: 0	Updated when a new match in which the player played is inserted.
Runs	int	Default value: 0	
Wickets	int	Default value: 0	
MoM’s	int	Default value: 0	
Current runs	int	Default value: 0 Current runs <= Runs	
Current wickets	int	Default value: 0 Current wickets <= Wickets	

Seasons			
Attribute	**Type**	**Constraints**	**Note**
Year	int	Required	Primary key
Results	[str, int]	Default value: [] Mutable only if Finished is false	[Team ID, points] Points derived from match results
Orange cap	str	Default value: null Mutable only if Finished is false	Derived from ‘Players’

Purple cap	str	Default value: null Mutable only if Finished is false	Derived from ‘Players’
Finished	bool	Default value: false No other entry must have Finished == true	Derived as Finished = (Results.length == (Teams.length) * (Teams.length - 1))

Matches			
Attribute	**Type**	**Constraints**	**Note**
ID	str	Required	Surrogate primary key
Winner ID	str	Winner ID == Team ID 1 or Winner ID == Team ID 2	Referential Integrity, team ID should be valid team ID
Umpires	[str]	Umpires.length == 3	[Umpire Name]
MoM ID	str	MoM ID must exist in player list of one of the teams	
Stadium ID	str		
Scorecard	[str, int, int]	Must have length 22 All players must be in one of the teams Wickets >= 0 Runs >= 0	[Player ID, Runs, Wickets]

Season	int	The season must be ongoing	

Stadium			
Attribute	**Type**	**Constraints**	**Note**
Stadium name	str		Composite key (Name, City)
Stadium city	str		
Max capacity	int	>= 0	

Relationships:

- Player for:
  - Entities: Team and players
  - Cardinality is T:P = 1:11
  - This relationship gives us the information about the players that play for the team.
- Captain:
  - Entities: Team and player
  - Cardinality is T:C = 1:1
  - This relationship gives us the captain of the team and is also the example of multiple relationships between two entities.
- Management(Weak Entity):
  - Entities: Team and coaches and team doctors.
  - Cardinality is T:M = 1:N
  - Gives us the information about the management staff of the team.
- Played:
  - Entities: Teams and Matches
  - Cardinality is T:M = 1:N(considering that the team plays n number of matches in a season)

    - Gives us the information about the matches played by a team,
- Plays:
    - Relation attribute: Stadium
    - Entities: Teams, Match, Players, Season.
    - Gives us the information about the match played and how it is related to the given entities.
- Scorecard(Weak entity):
    - Entities: Match and Scorecard
    - Cardinality: M:S = 1:1
    - Gives us the information about the runs scored and wickets taken by the players of the teams in a match.

Functionality

1. All information about players, teams, matches and seasons can be retrieved
2. Application users will be able to retrieve team players and their management.
3. Users will be able to retrieve the overall top scorers list in terms of runs and wickets. They can do this for the ongoing tournament as well. They can also set thresholds for their queries, ie, getting all players above a certain score
4. Users can retrieve all teams and their points and positions in any of the seasons. They can also get the total points any team has won till now across all seasons.
5. The average total scores and average total wickets of all batsmen, bowlers or all rounders of a team can also be calculated and reported.
6. Users can search for teams, players and stadiums by their names.
7. The total number of matches that have been played in a particular stadium can be reported.

8. Matches can be added, given that they satisfy all the constraints (Exactly 2 teams having 11 players each, stadium should exist, result should be decisive, etc)
9. Player information can be updated, eg, their age.
10. If players or matches have been wrongly inserted, they can be removed

Requirements

Database Requirements:

- ✓ *At least five entity types*: Teams, Players, Seasons, Matches, Stadium
- ✓ *At least one entity with two key attributes*: Stadium(Composite key of stadium name, stadium city)
- ✓ *At least two weak-entity types*: Team Management, Scorecard of the match
- ✓ *At least five relationship types (which should include cardinality ratios and participation constraints):* Stadium to Teams and Matches. Players to Teams. Teams to Season, Player and Matches. Matches to Teams, Seasons.
- ✓ *At least one (n > 3) degree relationship type:* The relation between the season, match, team and players.(n=4)
- ✓ *At least one subclass:* Players can be divided into the subclasses of batsman, bowler and all-rounder.
- ✓ *Few composite, multi-valued, derived attributes:*
    - ✓ Composite: Names of the players can be divided into first-name and last-name.
    - ✓ Multi-Valued: The Umpires attribute to the match entity, since there are 3 Umpires for a match.
    - ✓ Derived: The runs and wickets of a player is a derived attribute from the scorecard of the matches played by the player.

- ✓ *Relationship type with the same participating entity type in distinct roles:* **Team captain** can be considered as an example since it is related to a team as both a player and a captain.
- ✓ *Relationship with cardinality constraints constraint (specifies a minimum or maximum participation):* The relation between a **team and players**(a team can have at max 11 players).

Functional Requirements:

- ✓ *Selection retrieval:* Players, teams, matches and seasons can be retrieved
- ✓ *Projection retrieval:* Players scoring above a particular threshold are retrievable
- ✓ *Aggregate retrieval:* Total matches in a particular stadium are retrievable
- ✓ *Search retrieval:* All entities can be searched
- ✓ *Analysis:* Number of players with a batting average(number of runs/matches in a season) greater than a number across all seasons and a number of players with a bowling average(number of wickets/match) lesser than a number across all seasons
- ✓ *Insert modification:* All entities support insertion operation with constraints
- ✓ *Update modification:* All entities support valid updates
- ✓ *Delete modification:* Players in a season, matches in a season, seasons.