Symbiosis Institute of Technology

A DBMS Project Report on

IPL 2020

Submitted by

SIDDHARTH VYAS - 18070124067

SRIJAN CHADHA - 18070124069

VIBHOR BHARGAVA – 18070124076

Under the Guidance of

Mrs. SHRUTI PATIL

Department of Information Technology

SYMBIOSIS INSTITUTE OF TECHNOLOGY, PUNE (2018-2022)

Index

1) Problem statement:

Being an ongoing event and a hot topic, we decided to understand how the data is organised for IPL2020 tournament (for example: teams, players, maximum runs, maximum wickets taken, etc), design and simulate it using the database management system, picking up the live data from daily matches and eventually showcasing it in our project.

2) Proposed Solution:

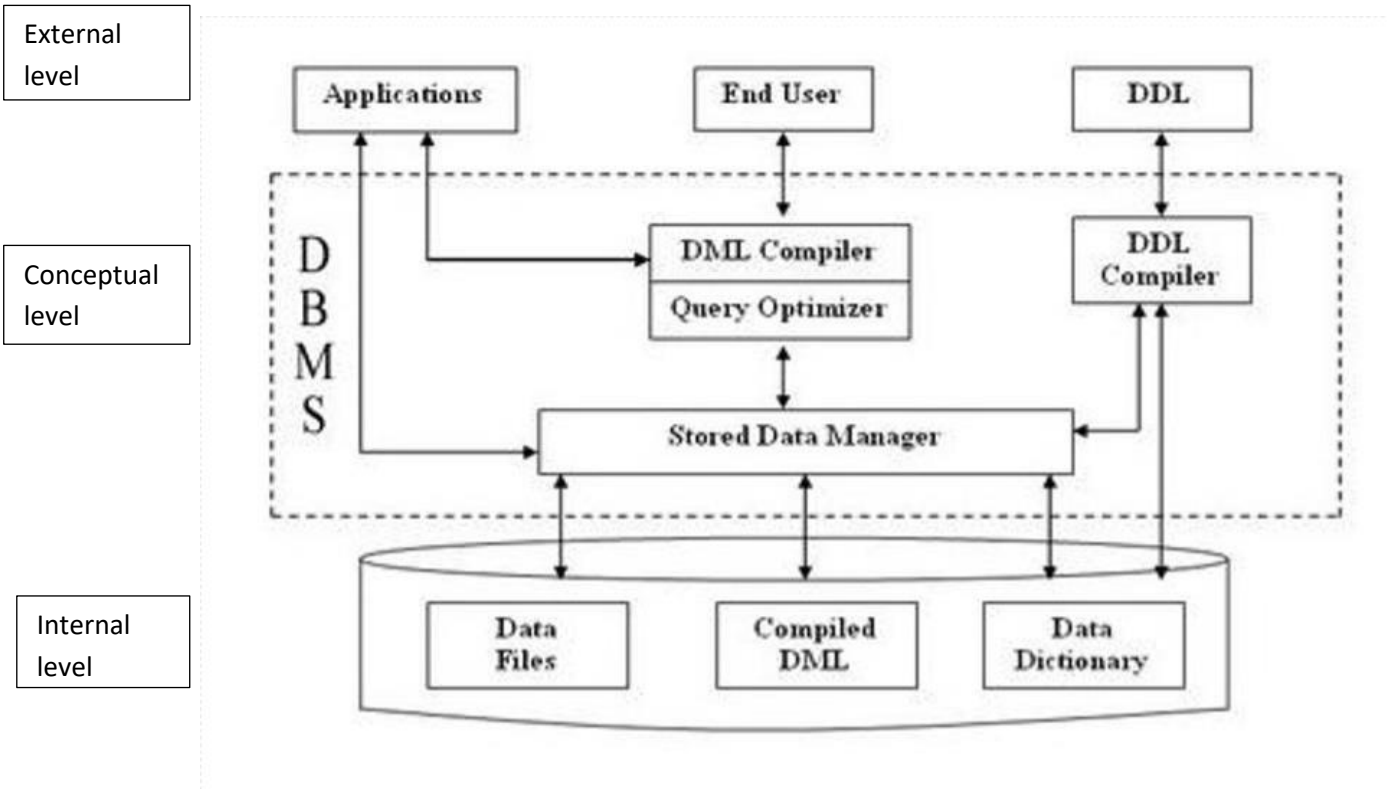
We would be working on a well organized and structured language, i.e., SQL and would opt for MySQL database software for our project as guided by our faculty.

We would target on gathering data of daily matches being held live and would be subsequently updating the data on a daily basis.
A glimpse showing points table of our proposed solution:

Teams	M	W	L	T/NR	Pts	NRR
Super Kings	13	9	4	0	18	+0.544
Royal Challengers	13	8	4	1	17	+0.283
Knight Riders	13	8	5	0	16	+0.476
Mumbai Indians	13	8	5	0	16	+0.033
Kings XI Punjab	14	7	7	0	14	-0.051
Rajasthan Royals	14	6	7	1	13	-0.691
Deccan Chargers	14	6	8	0	12	+0.222
Kochi Tuskers	14	6	8	0	12	-0.214
Pune Warriors	14	4	9	1	9	-0.134
Delhi Daredevils	14	4	9	1	9	-0.448

Updated till Delhi Daredevils vs Pune Warriors match

3) System Architecture:



Internal Level: The internal level has an internal schema which describes the physical storage structure of the database. The internal schema is also known as a physical schema. It uses the physical data model. It is used to define that how the data will be stored in a block. The physical level is used to describe complex low-level data structures in detail. This would provide most data abstraction from the external level thus a Naïve user won't be able to acces it.

- In the IPL Database, the internal level is managed by the backend staff who actually designs the database and tables of various matches, players, points table and other details and store them in proper Data Files in their systems.
- The database would be Distributed Database as the matches are held at various venues thus it allows the data administrators to access the database via network from anywhere.

- Data Dictionary would basically contain the meta data or simply the data of data of various defined tables like their datatypes etc. For ex: What would be the datatype of player name, runs, strike rate etc.
- The IPL database will consist of several tables including the "TEAMS, PLAYERS, ORANGE CAP, PURPLE CAP, MAXIMUM SIXES, FASTEST FIFTY, POINTS TABLE, BEST CATCH, LONGEST SIX, MATCH RESULTS"
- These data will be interconnected with each other through ids. Teams will be connected to the players. Each player will have his unique id. Through this unique id players will be listed in the tables of orange cap, purple cap, maximum sixes, etc. and by the use of these unique ids the information about the player and his team can be gathered.

Conceptual Level: The conceptual schema describes the design of a database at the conceptual level. Conceptual level is also known as logical level. The conceptual schema describes the structure of the whole database. The conceptual level describes what data are to be stored in the database and also describes what relationship exists among those data. In the conceptual level, internal details such as an implementation of the data structure are hidden. Programmers and database administrators work at this level. Basically, this would contain the E-R Diagram/EER-Diagram of the database.

- The IPL database will consist of several tables including as mentioned in internal level and thus a suitable conceptual level diagram or an ER Diagram would be proposed and designed accordingly by the members. These members would only design the blueprint of database or a logical view of it.

External Level: At the external level, a database contains several schemas that sometimes called as subschema. The subschema is used to describe the different views of the database. An external schema is also known as View schema. Each view schema describes the database part that a particular user group is interested and hides the remaining database from that user group. The view schema describes the end user interaction with database systems. DQL is generally used in this level.

This level is basically managed by the Front-End members who integrates the backend database to various platforms like a website or mobile application.

- At the external level in our IPL Database, users will be able to extract the data which they want. If a Naive user wants to know which player is in which team, who has scored the most runs, who is the most valuable player, what were the match results and several queries which won't manipulate the database and thus data abstraction is maintained.
- An example of this level type can be seen in apps like CRICBUZZ, DREAM 11.

Functional Requirements:

USE CASE: QUERY IPL DATABASE
ID: UC 1
ACTOR: Front-End developer/user.
Pre-condition: The user cannot make changes on the database.
Flow Of Events: 1) The user will enter a particular query like fetching the points table of teams. 2) Upon request, the application would check for correct syntax of query according to DQL and which table's data the user wan to access. 3) Query would be sent to database and required information would be fetched and given to user.
Secondary Scenario: ● User enters wrong syntax of query.
Post-condition: ● The response would be given to user.

USE CASE: DATA MANIPULATION
ID: UC 2
ACTOR: Data Admin
Pre-condition: The database and tables should be created with values.
Flow Of Events: 1) Admin generates a data manipulation language like insert, delete. 2) Upon query request, syntax would be checked. 3) Particular table would be manipulated with changes made by admin.
Secondary Scenario: ● The table is already deleted or syntax is wrong.
Post-condition: ● The requested changes would be done to the database.

USE CASE: DATA DEFINITION
ID: UC 3
ACTOR: Backend Developer
Pre-condition: There should be a proper E-R/EER model defined with relational schema, no. of tables, information about columns and rows must be known before-hand along.
Flow Of Events: 1) The backend team would create several tables with well defined datatypes, keys(primary,foreign), null value conditions etc. 2) Data or the values would be filled inside tables for each record in the database. 3) Stores the database in Data Files
Secondary Scenario: ● Syntax is wrong or out-of-bounds condition for datatype.
Post-condition: ● The database and its tables would be created.

4) Entities, Attributes and relationships:

- Teams:
    1) Team ID
    2) Team Name
    3) Team Captain
    4) Team Owner
    5) Home Ground

6) No of titles won

- Players:
    1) Team ID
    2) Player ID
    3) Player Name
    4) Role
    5) Runs Scored
    6) Wickets Taken

- Orange Cap
    1) Player ID
    2) Team ID
    3) Player's team
    4) Runs Scored
    5) Highest Score

- Purple Cap
    1) Player ID
    2) Team ID
    3) Player's team
    4) Wickets Taken
    5) Best Figures

- Maximum Sixes
    1) Player ID
    2) Team ID
    3) Player's team
    4) No of sixes

- Fastest Fifty
    1) Player ID
    2) Team ID
    3) Player's team
    4) Balls taken to reach 50

- Points Table
    1) Team ID

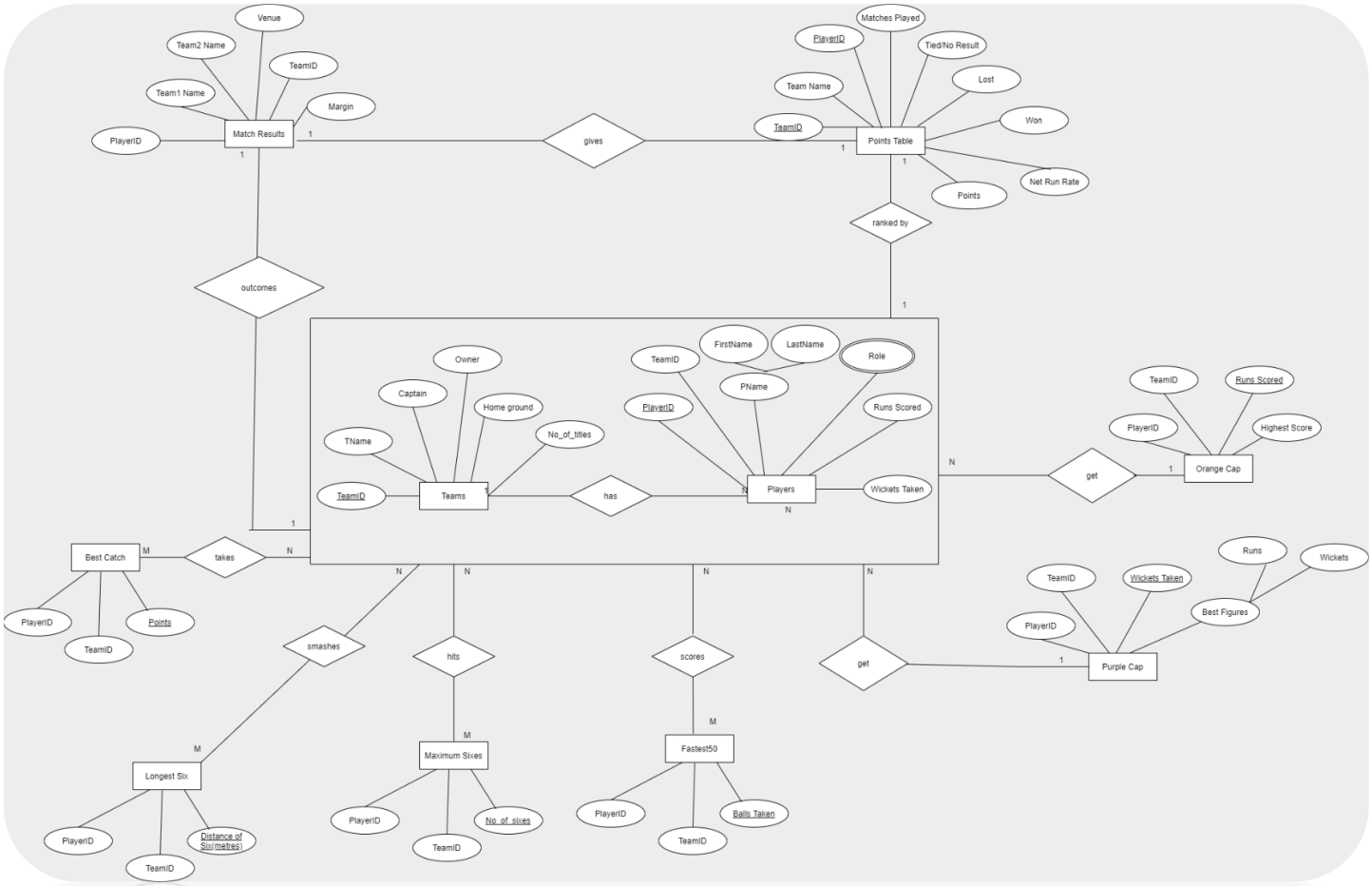
2) Player ID
3) Team Name
4) Matches Played
5) WON
6) LOSS
7) Tied/no result
8) NRR

- Best CATCH
    1) Player ID
    2) Team ID
    3) Player's team
    4) Points taken

- Longest six
    1) Player ID
    2) Team ID
    3) Player 's team
    4) Length of the six (in meters)

- Match Results
    1) Team name 1
    2) Team name 2
    3) Venue
    4) Player ID
    5) Team ID
    6) Result
    7) Margin

5) Enhanced Entity-Relationship (EER) Diagram:



6) Relational Schema with defined keys:

Teams (TeamID, TName, Captain, Owner, HomeGround, No_of_titles)

- Primary Key: TeamID

Players (PlayerID, TeamID, FirstName, LastName, RunsScored, WicketsTaken)

- Primary Key: PlayerID
- Foreign Key: TeamID

Players_Role (PlayerID, Role)

- Primary Keys(Composite Primary Key): PlayerID, Role

BestCatch (Points, PlayerID, TeamID)

- Primary Key: Points
- Foreign Keys: PlayerID, TeamID

Takes (PlayerID, Points)

- Primary Keys(Composite Primary Key): PlayerID, Points

LongestSix (Distance in metres, PlayerID, TeamID)

- Primary Key: Distance in metres
- Foreign Keys: PlayerID, TeamID

Smashes (PlayerID, Distance in Metres)

- Primary Keys(Composite Primary Key): PlayerID, Distance in Metres

Maximum Sixes (No of sixes, PlayerID, TeamID)

- Primary Key: No of sixes
- Foreign Keys: PlayerID, TeamID

Hits (PlayerID, No of sixes)

- Primary Keys(Composite Primary Key): PlayerID, No of sixes

Fastest50 (Balls Taken, PlayerID, TeamID)

- Primary Key: Balls Taken
- Foreign Keys: PlayerID, TeamID

Scores (PlayerID, Balls Taken)

- Primary Keys(Composite Primary Key): PlayerID, Balls Taken

Purple Cap (Wickets Taken, PlayerID, TeamID, BestFigures_Runs Given, BestFigures_Wickets)

- Primary Key: Wickets Taken
- Foreign Keys: PlayerID, TeamID

Orange Cap (Runs Scored, PlayerID, TeamID, BestFigures)

- Primary Key: Runs Scored
- Foreign Keys: PlayerID, TeamID

Points Table (TeamID, PlayerID, TName, MatchesPlayed, Won, Lost, Tied/NoResult, NetRunRate, Points)

- Primary Keys(Composite Primary Key): PlayerID, TeamID

Match Result (Team1 Name, Team2 Name, Venue, TeamID, PlayerID, Margin)

- Foreign Keys: TeamID, PlayerID

7) Codd's Rule:

These are the rules that every database must follow and apply. Rule 0 specifies that the system must qualify as relational as database and as a management system A database in which all the 12 codd's rule are applied is termed as a Relational Database Management System.

In our project, some of the Codd's 12 Rules are applied as follows:

1. **Information Rule:** The detailed information or the data of our project entitled IPL 2018 is stored in the database. All the data is stored in the form of tables. Tables will be created which store the data about teams of the Ipl , players of each team, points table, etc. Also, metadata(information about datatypes etc) would be stored in tabular form.Also, rows and columns are unordered(changing positions won't affect the database).
   - ✓ Hence, Rule 1 is applied in our project.

2. **Guaranteed Access Rule:** Each and every single data element value is logically accessible through a combination of table name, primary key(row value) and Attribute(column value). Pointers are not used to access the data value.
   - Captain's name of each team can be found out using the table name "Teams" , "TeamID" as the primary key and column name "Captains".
   - Number of batsmen in a team can be find using "Players" table , "PlayerID" as the primary key and column name "Role".
   - Example: SELECT Captain FROM Teams WHERE TeamID=1 will give a unique record of a team member.

   - ✓ Hence, Rule 2 is applied in our project.

3. **Systematic Treatment of the NULL Values**: Majority of the tables in our database do not have NULL values. But in case of the "Match Results" table if a particular match did not happen due to rain or some other reasons "Margin" column will have NULL value. Also, no value of primary key would be null.

- ✓ Hence, Rule 3 is applied in our project.

4. **Active Online Catalog:**

The active online catalog would basically contain the meta-data of various tables and columns like datatypes of each column, relations they belong to etc.
Data Dictionary for our project is as follows:

RELATIONS

Relation_name	No_of_columns
Teams	6
Players	6
Player_Role	2
BestCatch	3
Takes	2
LongestSix	3
Smashes	2
MaximumSixes	3
Hits	2
Fastest50	3
Scores	2
Purple Cap	5
Orange Cap	4
Points Table	9
Match Result	6

COLUMNS

Column_Name	Data_Type	Belongs_to_relation
TeamID	Int	Teams
PlayerID	Int	Teams
TName	Varchar(20)	Teams
Captain	int	Teams
Owner	Varchar(20)	Teams
HomeGround	Varchar (20)	Teams
No_of_titles	Int	Teams
PlayerID	Int	Players
TeamID	Int	Players
FirstName	Varchar(20)	Players
LastName	Varchar(20)	Players
RunsScored	int	Players
WicketsTaken	int	Players
PlayerID	int	Players_Role
TeamID	Int	Players Role
Role	Varchar(20)	Players_Role
Points	Int	BestCatch
PlayerID	Int	BestCatch
TeamID	int	BestCatch
PlayerID	Int	Takes
TeamID	Int	Takes
Points	int	Takes
Distance(in m)	Int	LongestSix
PlayerID	Int	LongestSix
TeamID	Int	LongestSix
PlayerID	int	Smashes
PlayerID	Int	Smashes
Distance(in m)	int	Smashes
No. of Sixes	Int	Maximum Sixes
PlayerID	Int	Maximum Sixes
Team ID	Int	Maximum Sixes
PlayerID	Int	Hits
TeamID	Int	Hits
No. of Sixes	Int	Hits
Balls Taken	Int	Fastest 50
PlayerID	Int	Fastest 50
TeamID	Int	Fastest 50
PlayerID	Int	Scores

TeamID	Int	Scores
Balls Taken	Int	Scores
Wickets Taken	Int	Purple Cap
PlayerID	Int	Purple Cap
TeamID	Int	Purple Cap
BestFigures_RunsGiven	Int	Purple Cap
BestFigures_Wickets	Int	Purple Cap
Runs Scored	Int	Orange Cap
PlayerID	Int	Orange Cap
TeamID	Int	Orange Cap
Player's Team	Varchar(20)	Orange Cap
Highest Score	Int	Orange Cap
PlayerID	Int	Points Table
TeamID	Int	Points Table
Team Name	Varchar(20)	Points Table
Matches Played	Int	Points Table
Won	char	Points Table
Loss	char	Points Table
Tied/no Result	char	Points Table
NRR	float	Points Table
PlayerID	Int	Match Results
TeamID	Int	Match Results
Team Name 1	Varchar(20)	Match Results
Team Name 2	Varchar(20)	Match Results
Result	Varchar(50)	Match Results
Margin	Varchar(50)	Match Results
Tied/no Result	char	Match Results

✓ Hence, Rule 4 is applied in our project.

5. **Comprehensive data sub language rule**: Our database has been created using the "SQL" language. This language supports data definition, data manipulation and transaction management operations. It can be directly accessed by using the MYSQL workbench.

✓ Hence, Rule 5 is applied in our project.

6. **View Updating Rule:** View is basically a virtual table which can be achieved by fetching only particular columns. Ex: From “Match Results” table, if we want to see just the columns “Team Name 1” and “Team Name 2” we can achieve it using selection and projections. Also, views, the sub parts of a table in a database can be updated using the “insert”, “delete” and “update” commands in the system itself.

✓ Hence, Rule 6 is applied in our project.

7. **Relational Level Operations(High level Insert, update and delete) rule**: Our database supports the high-level insertion, deletion and updation using the command “insert”, “delete” and “update” respectively. Also, various operations such as union, intersection and minus operations are applicable in our database.

- If we want bring a new player (say) for the team KKR after the auctions we can insert the details of that player in the “Players” table using the “insert” operation.
- If some players withdraw from the IPL due to injuries or some personal reasons, we can delete their details from the “Players” table of that particular team.
- If for some reason one or two matches are rescheduled or home grounds of the team may be changed due to some political reasons, we can update these changes using the “update” command.
- If we want to know the Player who has hit maximum sixes in the tournament and also hit longest six, we can use intersection from Longest Sixes and Maximum Sixes tables.
- If we want best performer or an all-rounder of tournament, we can use union operation of orange and purple cap tables.

✓ Hence, Rule 7 is applied in our project.

8. **Physical data independence**: Due to Physical independence, any of the below change will not affect the conceptual layer.

- Using a new storage device like Hard Drive or Magnetic Tapes
- Modifying the file organization technique in the Database
- Switching to different data structures.
- Changing the access method.
- Modifying indexes.
- Changes to compression techniques or hashing algorithms.
- Change of Location of Database from say C drive to D Drive

- ✓ Hence, Rule 8 is applied in our project.

9. **Logical Data Independence**: This rule is very difficult to follow in case of IPL database as it would be a Distributed Database. So, any changes made in the database like renaming a table at a particular venue where a match is held would change it on that particular user's logical view but that table would still remain the same in some other's view at some other venue. Due to this, any of the below change will affect the external layer.

- Add/Modify/Delete a new attribute, entity or relationship is possible without a rewrite of existing application programs
- Merging two records into one
- Breaking an existing record into two or more records

- Hence, Rule 9 cannot be applied in our project.

10. **Integrity Independence:** This database will be completely independent of the front-end application and its interface. All its integrity constraints can be independently modified without the need of any change in the application.

✓ Hence, Rule 10 can be applied in our project.

11. **Distribution Independence:** IPL database would be common at each venue and as our database is Distributed Database, any user accessing the IPL database from any venue where a match is being held, will get an impression that the data is centered at one site only.

✓ Hence, Rule 11 can be applied in our project.

12. **Non subversion Rule:** The interface that will be used for this project's database will not be able to bypass the security and integrity constraints as the IPL database would follow strong level of data encryption. So, even if a user gets to work on the Physical Schema like the Data Files, the user would not be able to make changes to any constraints imposed on Teams, Players etc.

✓ Hence, Rule 12 can be applied in our project.