

# **DELHI TECHNOLOGICAL UNIVERSITY**



## **DATABASE MANAGEMENT SYSTEM LAB PROJECT**

### **INDIAN PREMIER LEAGUE (IPL) DATABASE**

**SUBMITTED TO:**

Mr. MANOJ SETHI

**SUBMITTED BY:**

VAIBHAV RAJ SAHNI

2K19/CO/419

# ACKNOWLEDGEMENT

I would like to express a deep sense of thanks and gratitude to my teacher Mr. Manoj Sethi for giving me the golden opportunity to do this project on the topic “INDIAN PREMIER LEAGUE (IPL) Database” and guiding me immensely through the course of the project. His constructive advice and constant motivation have been responsible for the successful completion of the project.

I would also like to thank Mr. Yogesh Singh, our Vice-Chancellor, for his co-ordination in extending every possible support for completion of this project.

Last but not the least, I would like to thank all those who had helped me directly or indirectly towards the completion of the project.

# CERTIFICATE

This is to certify that Vaibhav Raj Sahni (2K19/CO/419) has successfully completed the Database Management Lab Project entitled “INDIAN PREMIER LEAGUE (IPL) Database”. The project report is a result of his efforts and endeavors. The project is found worthy of acceptance as final project for Database Management Systems Lab. The project has been prepared under my knowledge consistently.

Mr. Manoj Sethi

(DBMS Teacher)

# CONTENTS

- Introduction
- Proposed work
- Implementation
  - Entity Relationship Diagram
  - Normalisation
  - Schema
  - Data Pre-processing
  - Creating database
  - Importing data
  - Adding Constraints
- Results
- Conclusion
- References

# INTRODUCTION

The Indian Premier League (IPL) is a professional Twenty20 Cricket league in India usually contested between March and May of every year by eight teams representing eight different cities or states in India. The league was founded by the Board of Control for Cricket in India (BCCI) in 2007 and its first season was played in 2008. Ever since then, 13 seasons of IPL have been played and its elaborate ball-by-ball data amounts to a whole database system. In this project, I will build a MySQL database using dataset taken from Kaggle.com. The dataset consists of 6 compressed comma-separated-value (.csv) files that are downloaded from the same.

## Dataset Used

Each dataset is contained in a comma-separated-values (csv) formatted file in the UTF-8 character set. The first line in each file contains headings for that column. A “ ” (blank space) is used to denote that a particular field is missing or has a NULL value for varchar inputs while integer inputs which were earlier also stored in similar format were changed to an integer 0 to rectify the anomaly. It should be noted that the data available for download from Kaggle.com is from 2008 to 2016 and serves our purpose. The available IPL data files are as follows:

### Ball\_by\_Ball.csv

Contains the following information for ball-by-ball data:

- Match\_Id(integer)- identifies which match is the data from
- Innings\_Id (integer) – identifies which innings of the match is the data from
- Over\_Id(integer) - identifies which innings of the match is the data from
- Ball\_Id (integer) - identifies which ball of the over is the data from
- Team\_Batting\_Id (integer) - identifies team\_id of the team currently batting
- Team\_Bowling\_Id (integer) - identifies team\_id of the team currently bowling
- Striker\_Id (integer) - gives player\_id of the player who is on the striker's end
- Striker\_Batting\_Position (integer) - gives the batting position of the striker

- Non\_Striker\_Id (integer) - gives player\_id of the player who is on the non-striker's end
- Bowler\_Id (integer) - gives player\_id of the player who is bowling currently
- Batsman\_Scored (integer) - number of runs scored on the current ball
- Extra\_Type (varchar) - states the type of extra run scored on the current ball if any
- Extra\_Runs (integer) - number of extra runs on the current ball if any
- Player\_Dismissal\_Id (integer) - gives player\_id of the player who is dismissed, else gives 0 as value.
- Dismissal\_Type (varchar) - identifies the type of dismissal
- Fielder\_Id (integer) - identifies player\_id of the fielder involved

### **Matches.csv**

Contains the following information for matches:

- Match\_Id (integer) - a number to uniquely identify each match
- Team\_Name\_Id (integer) – gives team\_id of one of the teams
- Opponent\_Team\_Id (integer) – gives team\_id of the opponent team
- Season\_Id (integer) - identifies which season of the IPL is the data from
- Venue\_Name (varchar) – states the name of the venue
- Toss\_Winner\_Id (integer) - gives team\_id of the team who won the toss
- Toss\_Decision (varchar) - states if the toss winning team chose to bat or field
- Is\_Superover (boolean) - identifies if the match was decided by a superover
- Is\_Result (boolean) - identifies if the match was completed or ended in “No Result”
- Is\_DuckWorthLewis (boolean) - identifies if the match was decided

### **Duckworth Lewis System (DLS)**

- Win\_Type (varchar) - states if the match was won “by runs”, “by wickets” or resulted in a “tie”
- Won\_By (integer) - gives number of wickets or runs by which match decision was made
- Match\_Winner\_Id (integer) - gives team\_id of the team that won the match
- Man\_Of\_The\_Match\_Id (integer) - gives player\_id of the player who won the man of the match award
- First\_Umpire\_Id (integer) - gives player\_id of the First Umpire
- Second\_Umpire\_Id (integer) - gives player\_id of the Second Umpire
- City\_Name (varchar) – states the city in which the match is being played
- Host\_Country (varchar) - states the country in which the match is being played

### **Player.csv**

Contains the following information for players:

- Player\_Id (integer) - a number to uniquely identify each player
- Player\_Name (varchar) - states the name of the player
- Batting\_Hand (varchar) – states if the player bats right-handed or left-handed
- Bowling\_Skill (varchar) – states the bowling arm and bowling technique
- Country (varchar) - states the country in which the match is being played
- Is\_Umpire (boolean) - identifies if the current player is an umpire

### **Season.csv**

Contains the following information for seasons:

- Season\_Id (integer) - a number to uniquely identify each season
- Season\_Year (integer) - identifies which year is the data from
- Orange\_Cap\_Id (integer) - gives player\_id of the player who scored the highest runs in the season
- Purple\_Cap\_Id (integer) - gives player\_id of the player who took the highest number of wickets in the season
- Man\_Of\_The\_Series\_Id (integer) - gives player\_id of the player who won the man of the series award

### **Team.csv**

Contains the following information for teams:

- Team\_Id (integer) - a number to uniquely identify each team
- Team\_Name (varchar) - states the name of the team
- Team\_Short\_Code (varchar) - states the short hand name used for the team

### **Player\_Match.csv**

Contains the following information:

- Match\_Id(integer)- identifies which match is the data from
- Player\_Id (integer) - gives player\_id of the concerned player
- Team\_Id (integer) - gives team\_id of the concerned player
- Is\_Keeper (boolean) - identifies if the current player is a wicket-keeper
- Is\_Captain (boolean) - identifies if the current player is the captain of the team

# PROPOSED WORK

The purpose of this project is to do the following:

- Learn about and use the database management system MySQL.
- Learn the essentials of database design, e.g., Entity-Relationship diagrams, logical schema, etc.
- Debug dataset anomalies using python script to replace blank spaces by 0 for integer values.
- Practice database queries by posing basic and more advanced queries using MySQL.

The project objectives include:

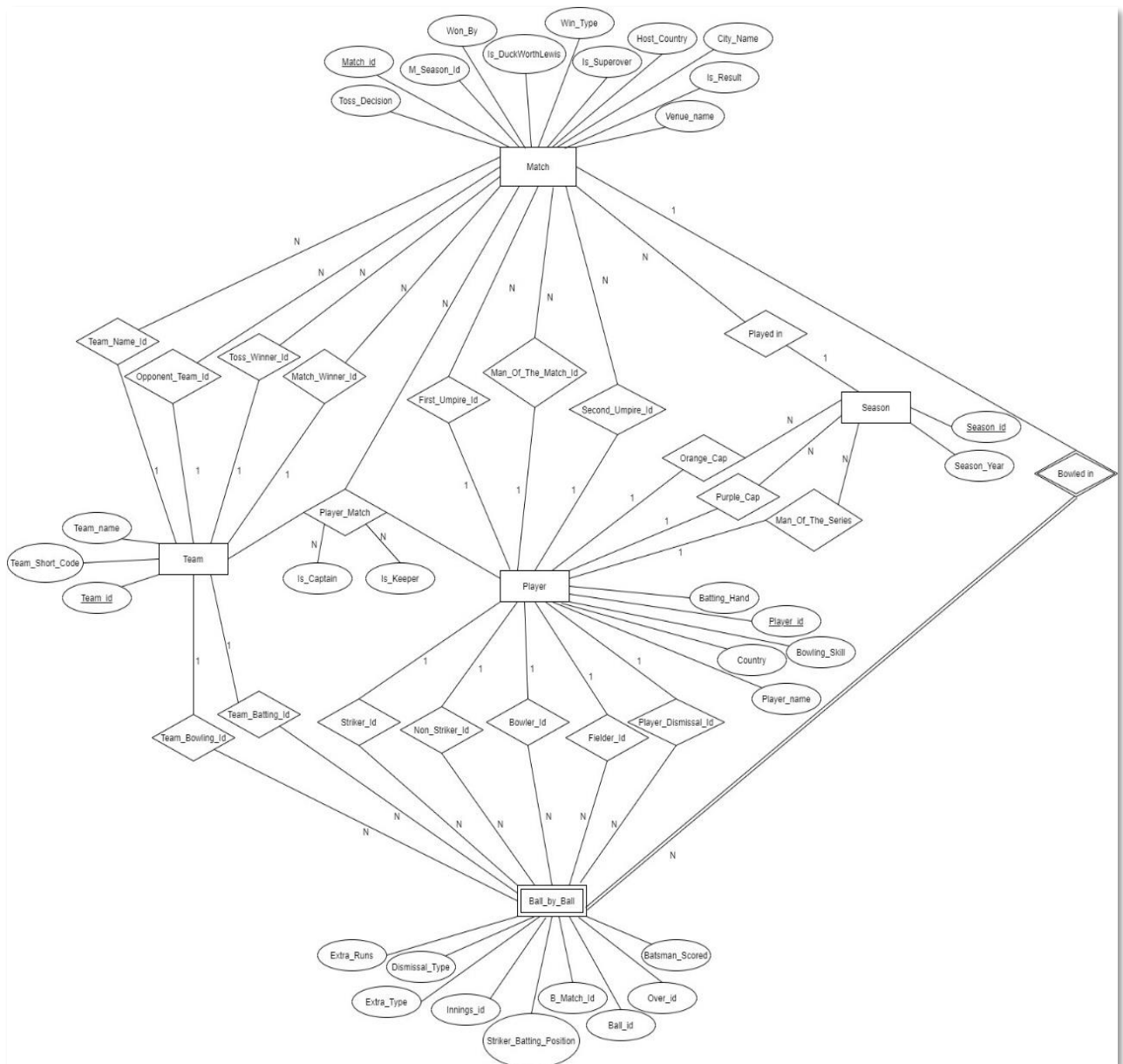
- Understand the data in the IPL dataset.
- Design a relational database and store the IPL data in it.
  - Model the database using an Entity-Relationship (ER) diagram
  - Perform normalisation
  - Create a logical schema
  - Create MySQL database
  - Load data into the database
  - Add primary and foreign key constraints
- Write SQL queries using different constraints, clauses and rules of the Structured Query Language and display their results as well



# IMPLEMENTATION

## ENTITY RELATIONSHIP DIAGRAM

Entity Relationship Diagram, also known as ERD is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. In other words, ER diagrams help to explain the logical structure of databases. They are created based on three basic concepts: entities, attributes and relationships. they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the inter-connectedness of entities, relationships and their attributes. The ER diagram of my database consists looks like this:



## **NORMALISATION**

Normalization is the process of organizing the data in the database. Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies. Normalization divides the larger table into the smaller table and links them using relationship. The normal form is used to reduce redundancy from the database table.

The conditions for 1NF, 2NF and 3NF are checked. This ER diagram is in 1 NF form due to the absence of multivalued attributes in any of the relations.

Following the second normal form, every non-prime attribute should be fully functionally dependent on prime key attribute. That is, if  $X \rightarrow A$  holds, then there should not be any proper subset  $Y$  of  $X$ , for which  $Y \rightarrow A$  also holds true. This means there should be no partial dependency. This condition is satisfied by all the relations and hence they are in second normal form.

For a relation to be in Third Normal Form, it must be in Second Normal form and no non-prime attribute must be transitively dependent on prime key attribute and for any non-trivial functional dependency,  $X \rightarrow A$ , then either –

- $X$  is a super key or,
- $A$  is prime attribute.

This condition is also satisfied by all the relations, that means, they are in the third normal form.

## **SCHEMA**

A database schema represents the logical configuration of all or part of a relational database. It can exist both as a visual representation and as a set of formulas known as integrity constraints that govern a database. These formulas are expressed in a data definition language, such as SQL. A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams.

The logical schema of my database consists of 6 tables or relations:

- Ball\_by\_Ball
- Matches
- Player
- Team
- Season
- Player\_Match

## BALL\_BY\_BALL

B_Match_Id	Innings_Id	Over_Id	Batsman_Scored	Team_Batting_Id
Striker_Id	Extra_Runs	Striker_Batting_Position	Ball_Id	Team_Bowling_Id
Bowler_Id	Non_Striker_Id	Extra_Type	Dismissal_Type	Fielder_Id
Player_Dismissal_Id				

## MATCHES

Match_Id	Win_Type	Venue_Name	M_Season_Id	Opponent_Team_Id
Toss_Winner_Id	Toss_Decision	Is_Superover	Is_Result	Match_Winner_Id
Man_Of_The_Match_Id	Won_By	Is_DuckWorthLewis	First_Umpire_Id	Team_Name_Id
Second_Umpire_Id	City_Name	Host_Country		

## PLAYER

Player_Name	Player_Id	Batting_Hand	Bowling_Skill	Country
Is_Umpire				

## PLAYER\_MATCH

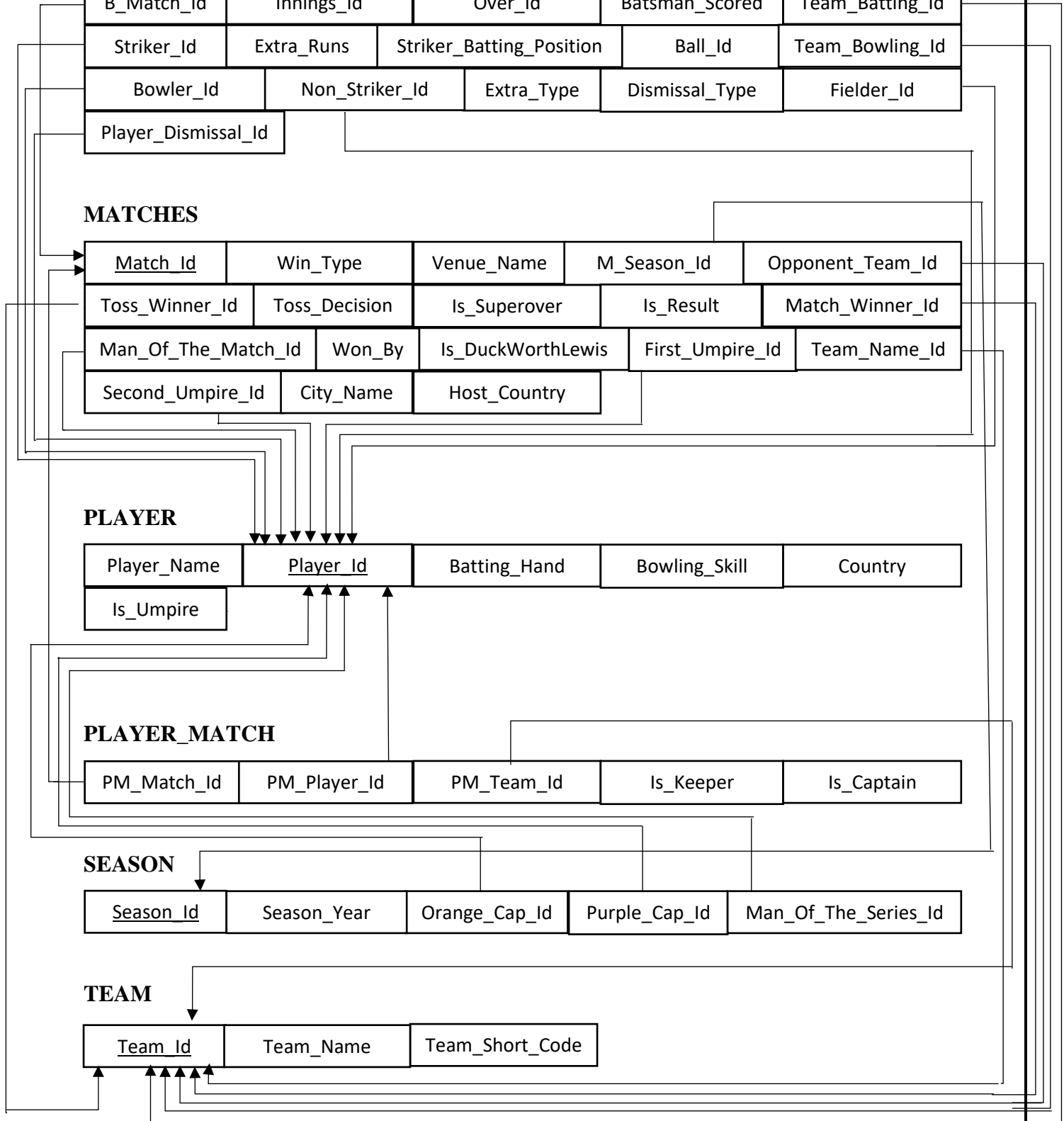
PM_Match_Id	PM_Player_Id	PM_Team_Id	Is_Keeper	Is_Captain
-------------	--------------	------------	-----------	------------

## SEASON

Season_Id	Season_Year	Orange_Cap_Id	Purple_Cap_Id	Man_Of_The_Series_Id
-----------	-------------	---------------	---------------	----------------------

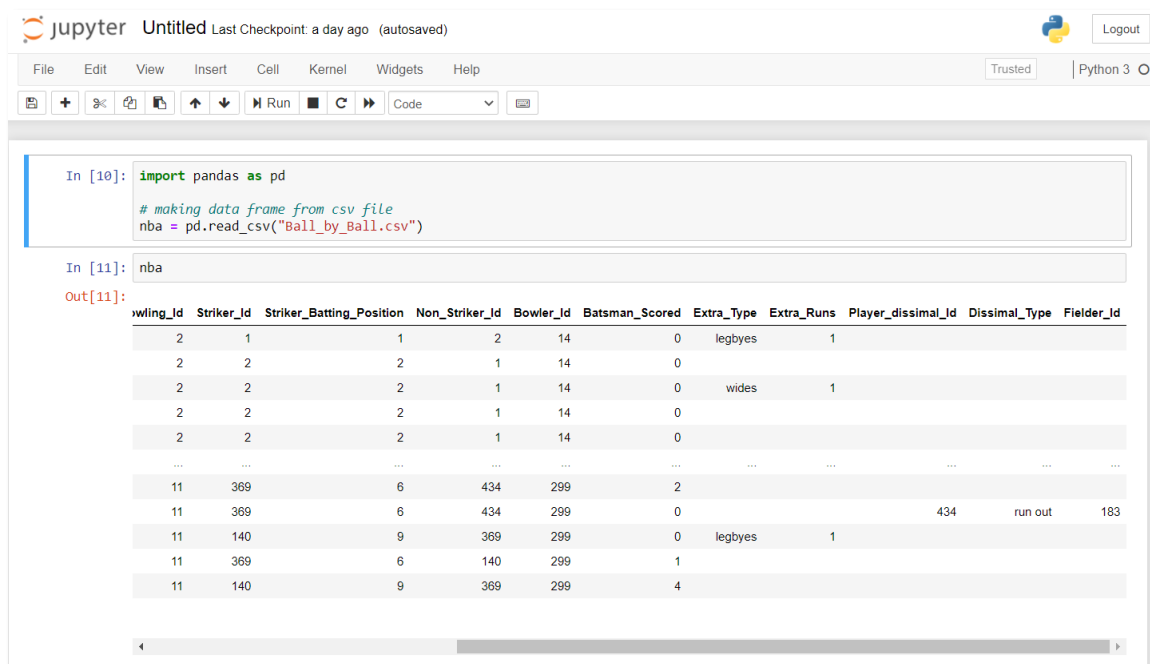
## TEAM

Team_Id	Team_Name	Team_Short_Code
---------	-----------	-----------------



## PRE-PROCESSING DATA

The data collected had some inconsistencies and thus, the data could not be imported properly. For example, in an integer field where the answer was supposed to be zero, instead in the csv file, it was stored as “ ” (blank space), which is a character value. Therefore, python scripts were written for this data pre-processing to convert these blank spaces to the value 0.



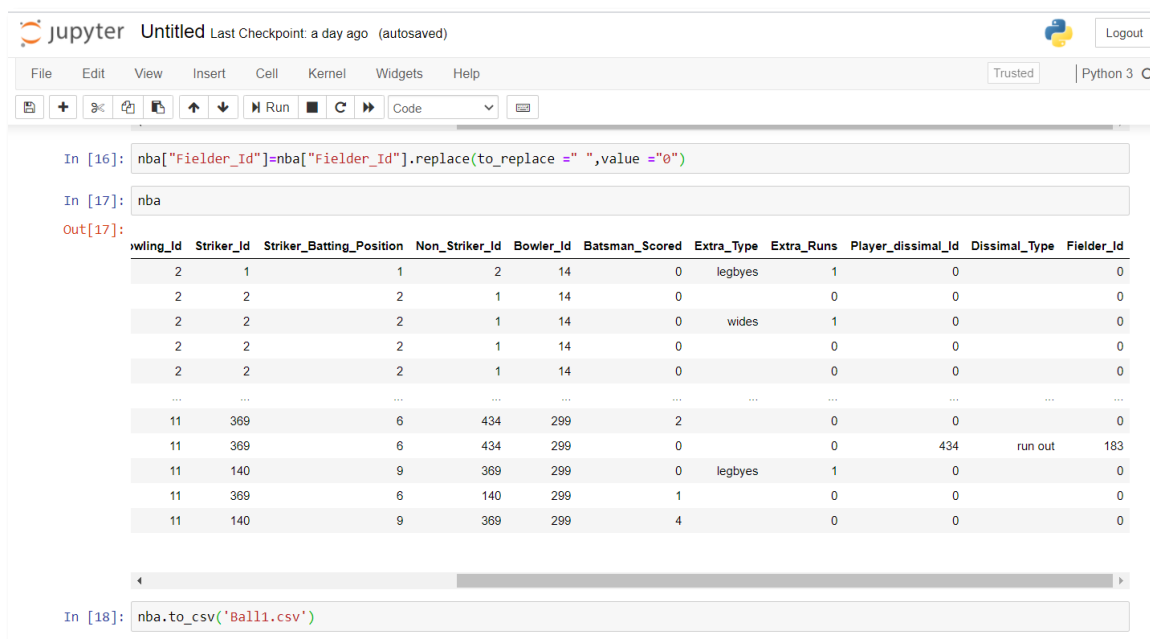
The screenshot shows a Jupyter Notebook interface with the following code and output:

```
In [10]: import pandas as pd
# making data frame from csv file
nba = pd.read_csv("Ball_by_Ball.csv")

In [11]: nba
```

Out[11]:

bowling_Id	Striker_Id	Striker_Batting_Position	Non_Striker_Id	Bowler_Id	Batsman_Scored	Extra_Type	Extra_Runs	Player_dissimal_Id	Dissimal_Type	Fielder_Id
2	1	1	2	14	0	legbyes	1			
2	2	2	1	14	0					
2	2	2	1	14	0	wides	1			
2	2	2	1	14	0					
2	2	2	1	14	0					
...	...	...	...	...	...	...	...	...	...	...
11	369	6	434	299	2					
11	369	6	434	299	0			434	run out	183
11	140	9	369	299	0	legbyes	1			
11	369	6	140	299	1					
11	140	9	369	299	4					



The screenshot shows the same Jupyter Notebook interface with additional code and output:

```
In [16]: nba["Fielder_Id"] = nba["Fielder_Id"].replace(to_replace = " ", value = "0")

In [17]: nba
```

Out[17]:

bowling_Id	Striker_Id	Striker_Batting_Position	Non_Striker_Id	Bowler_Id	Batsman_Scored	Extra_Type	Extra_Runs	Player_dissimal_Id	Dissimal_Type	Fielder_Id
2	1	1	2	14	0	legbyes	1	0		0
2	2	2	1	14	0		0	0		0
2	2	2	1	14	0	wides	1	0		0
2	2	2	1	14	0		0	0		0
2	2	2	1	14	0		0	0		0
...	...	...	...	...	...	...	...	...	...	...
11	369	6	434	299	2		0	0		0
11	369	6	434	299	0		0	434	run out	183
11	140	9	369	299	0	legbyes	1	0		0
11	369	6	140	299	1		0	0		0
11	140	9	369	299	4		0	0		0

```
In [18]: nba.to_csv('Ball1.csv')
```

## CREATING THE DATABASE

After creating the ER diagram and logical schema, the database was created in MySQL using SQL. Structured Query Language (SQL) is a database language by the use of which we can create a database and also perform certain operations on the existing database. SQL can be used as a Data Definition Language (DDL) by using the SQL commands that can be used to define the database schema. It simply deals with descriptions of the database schema and is used to create and modify the structure of database objects in the database.

The database was created using the CREATE DATABASE command

```
Create Database IPL;  
Use IPL;
```

The tables were created using the CREATE TABLE command

```
Create Table Ball_by_Ball  
(  
    Match_Id integer,  
    Innings_Id integer,  
    Over_Id integer,  
    Ball_Id integer,  
    Team_Batting_Id integer,  
    Team_Bowling_Id integer,  
    Striker_Id integer,  
    Striker_Batting_Position integer,  
    Non_Striker_Id integer,  
    Bowler_Id integer,  
    Batsman_Scored integer,  
    Extra_Type varchar(30),  
    Extra_Runs integer,  
    Player_Dismissal_Id integer,  
    Dismissal_Type varchar(30),  
    Fielder_Id integer  
);
```

```
Create Table Matches  
(  
    Match_Id integer PRIMARY KEY,  
    Team_Name_Id integer,  
    Opponent_Team_Id integer,  
    Season_Id integer,  
    Venue_Name varchar(200),  
    Toss_Winner_Id integer,  
    Toss_Decision varchar(15),  
    Is_Superover integer,  
    Is_Result integer,  
    Is_DuckworthLewis integer,  
    Win_Type varchar(30),  
    Won_By integer,  
    Match_Winner_Id integer,  
    Man_Of_The_Match_Id integer,  
    First_Umpire_Id integer,  
    Second_Umpire_Id integer,  
    City_Name varchar(200),  
    Host_Country varchar(50)  
);
```

```
Create Table Player
(
    Player_Id integer PRIMARY KEY,
    Player_Name varchar(100),
    Batting_Hand varchar(30),
    Bowling_Skill varchar(50),
    Country varchar(50),
    Is_Umpire integer
);
```

```
Create Table Team
(
    Team_Id integer PRIMARY KEY,
    Team_Name varchar(60),
    Team_Short_Code varchar(4)
);
```

```
Create Table Season
(
    Season_Id integer PRIMARY KEY,
    Season_Year integer,
    Orange_Cap_Id integer,
    Purple_Cap_Id integer,
    Man_Of_The_Series_Id integer
);
```

```
Create Table Player_Match
(
    Match_Id integer,
    Player_Id integer,
    Team_Id integer,
    Is_Keeper integer,
    Is_Captain integer
);
```

## IMPORTING THE DATA

After creating the database schema in mySQL, the next step is to import the data into the created tables. The data has already been pre-processed and ready to be imported. The files to be imported are of very large size, with some of them having records as much as 1.3 Lakhs. The csv files were then directly imported using the Table Data Import Wizard feature of MySQL WorkBench.

## ADDING CONSTRAINTS

Next, all the foreign key constraints were added in the respective tables:

- Ball\_by\_Ball

```
ALTER TABLE Ball_by_Ball
ADD FOREIGN KEY (Match_Id) REFERENCES Matches(Match_Id);
ALTER TABLE Ball_by_Ball
ADD FOREIGN KEY (Striker_Id) REFERENCES Player(Player_Id);
ALTER TABLE Ball_by_Ball
ADD FOREIGN KEY (Non_Striker_Id) REFERENCES Player(Player_Id);
ALTER TABLE Ball_by_Ball
ADD FOREIGN KEY (Bowler_Id) REFERENCES Player(Player_Id);
ALTER TABLE Ball_by_Ball
ADD FOREIGN KEY (Fieldler_Id) REFERENCES Player(Player_Id);
ALTER TABLE Ball_by_Ball
ADD FOREIGN KEY (Player_Dismissal_Id) REFERENCES Player(Player_Id);
ALTER TABLE Ball_by_Ball
ADD FOREIGN KEY (Team_Batting_Id) REFERENCES Team(Team_Id);
ALTER TABLE Ball_by_Ball
ADD FOREIGN KEY (Team_Bowling_Id) REFERENCES Team(Team_Id);
```

- Matches

```
ALTER TABLE Matches
ADD FOREIGN KEY (Team_Name_Id) REFERENCES Team(Team_Id);
ALTER TABLE Matches
ADD FOREIGN KEY (Opponent_team_Id) REFERENCES Team(Team_Id);
ALTER TABLE Matches
ADD FOREIGN KEY (Toss_Winner_Id) REFERENCES Team(Team_Id);
ALTER TABLE Matches
ADD FOREIGN KEY (Match_Winner_Id) REFERENCES Team(Team_Id);
ALTER TABLE Matches
ADD FOREIGN KEY (Man_Of_The_Match_Id) REFERENCES Player(Player_Id);
ALTER TABLE Matches
ADD FOREIGN KEY (First_Umpire_Id) REFERENCES Player(Player_Id);
ALTER TABLE Matches
ADD FOREIGN KEY (Second_Umpire_Id) REFERENCES Player(Player_Id);
ALTER TABLE Matches
ADD FOREIGN KEY (Season_Id) REFERENCES Season(Season_Id);
```



- Season

```
ALTER TABLE Season
ADD FOREIGN KEY (Orange_Cap_Id) REFERENCES Player(Player_Id);
ALTER TABLE Season
ADD FOREIGN KEY (Purple_Cap_Id) REFERENCES Player(Player_Id);
ALTER TABLE Season
ADD FOREIGN KEY (Man_Of_The_Series_Id) REFERENCES Player(Player_Id);
```

- Player\_Match

```
ALTER TABLE Player_Match
ADD FOREIGN KEY (Match_Id) REFERENCES Matches(Match_Id);
ALTER TABLE Player_Match
ADD FOREIGN KEY (Player_Id) REFERENCES Player(Player_Id);
ALTER TABLE Player_Match
ADD FOREIGN KEY (Team_Id) REFERENCES Team(Team_Id);
```

# RESULTS

## SQL QUERIES

1. How many matches of the IPL have been hosted by which countries?

### QUERY

```
Select Host_Country, count(*) as Matches_Hosted  
from matches  
group by Host_country;
```

### OUTPUT

Host_Country	Matches_Hosted
India	497
South Africa	57
U.A.E	20

2. How many players from each country have played in the IPL?

### QUERY

```
Select Country, count(*) as No_of_Players  
from player  
group by Country;
```

### OUTPUT

Country	No_of_Players
India	291
New Zealand	26
Australia	78
Pakistan	15
South Africa	44
Sri Lanka	22
West Indies	20
Zimbabwea	3
England	18
Bangladesh	5
Netherlands	1

3. Name all the players who have opened the batting for the “Chennai Super Kings”.

## QUERY

```
Select distinct player_name as Opening_Batsmen
from player, ball_by_ball, team
where over_id=1 and ball_id=1 and team.team_id=ball_by_ball.Team_Batting_Id and
(player.player_id=ball_by_ball.striker_id or player.player_id=ball_by_ball.non_striker_id)
and team_name="Chennai Super Kings";
```

## OUTPUT

Opening_Batsman
PA Patel
ML Hayden
SP Fleming
S Vidyut
GC Smith
RJ Quiney
M Vijay
S Badrinath
GJ Bailey
JA Morkel
S Anirudha
MEK Hussey
F du Plessis
R Ashwin
WP Saha
BB McCullum
DR Smith

4. What is the total number of players that each team has recruited?

## QUERY

```
Select Team_Name, count(distinct player_id) as No_of_Players
from player_match, team
where team.team_id=player_match.team_id
group by player_match.team_id;
```

## OUTPUT

Team_Name	No_of_Players
Kolkata Knight Riders	83
Royal Challengers Bangalore	100
Chennai Super Kings	58
Kings XI Punjab	87
Rajasthan Royals	86
Delhi Daredevils	88
Mumbai Indians	89
Deccan Chargers	62
Kochi Tuskers Kerala	20
Pune Warriors	44
Sunrisers Hyderabad	43
Rising Pune Supergiants	23
Gujarat Lions	19

5. List the total number of matches won by runs, by wickets and those which were tied.

## QUERY

```
Select Win_Type, count(*) as No_of_Wins  
from matches  
group by Win_Type;
```

## OUTPUT

Win_Type	No_of_Wins
by runs	261
by wickets	307
Tie	6

6. List the top 50 players with the greatest number of Man of the Match Award wins.

## QUERY

```
Select player_name as Player, count(man_of_the_match_id) as MOTM
from player, matches
where player.player_id=matches.man_of_the_match_id
group by man_of_the_match_id
having MOTM>1
order by MOTM desc;
```

## OUTPUT

Player	MOTM
DA Warner	14
SK Raina	13
RG Sharma	13
MEK Hussey	12
MS Dhoni	12
G Gambhir	12
AM Rahane	12
V Kohli	11
V Sehwag	11
JH Kallis	10
SR Watson	10
DR Smith	10
SE Marsh	9
A Mishra	9
SR Tendulkar	8
KA Pollard	8
RA Jadeja	7
AC Gilchrist	7
Harbhajan ...	6
A Nehra	6
BJ Hodge	6
M Vijay	6
AT Rayudu	6
UT Yadav	6
AD Russell	6
SC Ganguly	5
BB McCullum	5
KC Sangakk...	5
RV Uthappa	5
DPMD Jaya...	5
DW Steyn	5
SL Malinga	5
AJ Finch	5
JP Faulkner	5
PA Patel	4
ML Hayden	4
Yuvraj Singh	4
SK Warne	4
KD Karthik	4
MK Pandey	4
JP Duminy	4
B Kumar	4
GJ Maxwell	4
SP Narine	4

7. What is the total number of left-handed and right-handed batsmen that have played in the IPL?

### QUERY

```
Select Batting_Hand, count(*) as No_of_Players
from player where batting_hand is not null
group by Batting_Hand;
```

### OUTPUT

Batting_Hand	No_of_Players
Left_Hand	126
Right_Hand	345

8. Which team has won a game with the highest margin of runs?

### QUERY

```
Select T1.team_name as Winning_Team, max(won_by) as Max_victory_by_runs, "Against", T2.team_name as Losing_Team
from matches, team T1, team T2
where win_type="by runs" and T1.team_id=matches.team_name_id and T2.team_id=matches.Opponent_team_id;
```

### OUTPUT

Winning_Team	Max_victory_by_runs	Against	Losing_Team
Royal Challengers Bangalore	144	Against	Kolkata Knight Riders



9. Name all the players and their score who have scored a century.

## QUERY

```
Select player_name as Player, sum(batsman_scored) as Runs
from player, ball_by_ball
where ball_by_ball.striker_id=player.player_id
group by striker_id, match_id
having sum(batsman_scored)>=100 ;
```

## OUTPUT

Player	Runs	Player	Runs
BB McCullum	158	AM Rahane	103
BB McCullum	100	MK Pandey	114
V Kohli	100	SE Marsh	115
V Kohli	108	AB de Villiers	105
V Kohli	105	AB de Villiers	133
V Kohli	113	AB de Villiers	129
MEK Hussey	116	SR Tendulkar	100
SK Raina	100	KP Pietersen	103
YK Pathan	100	CH Gayle	102
SR Watson	101	CH Gayle	107
SR Watson	104	CH Gayle	128
V Sehwag	119	CH Gayle	175
V Sehwag	122	CH Gayle	117
ST Jayasuriya	114	PC Valthathy	120
AC Gilchrist	109	M Vijay	127
AC Gilchrist	106	M Vijay	113
A Symonds	117	DA Warner	107
RG Sharma	109	DA Warner	109
WP Saha	115	SPD Smith	101
DPMD Jaya...	110	DA Miller	101
		Q de Kock	108
		LMP Simmons	100

10. Create a view to answer the following:

- Name the player who has played the maximum number of matches along with the count.
- What is the number of matches played by Chris Gayle?

```
Create view maxmatch as
Select player_name as Player, count(player_match.player_id) as No_of_Matches
from player, player_match
where player_match.player_id=player.player_id
group by player_match.player_id;
```

#### QUERY 1

```
Select Player , No_of_Matches
from maxmatch
where No_of_Matches=(SELECT max(No_of_Matches) FROM maxmatch);
```

#### OUTPUT 1

Player	No_of_Matches
SK Raina	146

#### QUERY 2

```
Select * from maxmatch where Player like "%gayle";
```

#### OUTPUT 2

Player	No_of_Matches
CH Gayle	89

11. List the top 50 bowlers in the descending order of the number of wickets they have taken.

## QUERY

```
Select player_name as Player,count(player_dismissal_id) as Wickets
from ball_by_ball, player
where ball_by_ball.bowler_id=player.player_id and player_dismissal_id>0
group by ball_by_ball.bowler_id
order by wickets desc
Limit 50;
```

## OUTPUT

Player	Wickets
SL Malinga	133
Harbhajan Singh	112
A Mishra	107
DJ Bravo	107
PP Chawla	106
A Nehra	92
R Vinay Kumar	90
Z Khan	86
SR Watson	86
IK Pathan	84
PP Ojha	84
JA Morkel	82
R Ashwin	82
RP Singh	82
P Kumar	76
MM Sharma	74
DW Steyn	73
B Kumar	72
DS Kulkarni	71
L Balaji	69
AB Dinda	64
RA Jadeja	64
MM Patel	61
R Bhatia	59
SK Warne	59

Player	Wickets
Sandeep Sharma	59
UT Yadav	57
SP Narine	56
M Morkel	56
YS Chahal	54
SK Trivedi	53
JP Faulkner	52
M Muralitharan	51
I Sharma	50
JH Kallis	50
MG Johnson	48
A Kumble	48
AR Patel	46
KA Pollard	46
S Aravind	43
SB Jakati	40
CH Morris	39
YK Pathan	39
MA Starc	39
Iqbal Abdulla	39
S Sreesanth	38
RJ Harris	38
KV Sharma	37
Shakib Al Hasan	36
MC Henriques	36

12. Create a view to answer the following:

- List the highest scores of all the batsmen.
- What is Mahendra Singh Dhoni's highest score?

```
Create view maxscore as
Select player_name as Player, sum(batsman_scored) as Runs
from player, ball_by_ball
where ball_by_ball.striker_id=player.player_id
group by striker_id, match_id;
```

## QUERY 1

```
Select Player, max(Runs) as Highest_Score from maxscore group by Player;
```

## OUTPUT 1

Player	Highest_Score
SC Ganguly	91
BB McCullum	158
RT Ponting	28
DJ Hussey	71
Mohammad Hafeez	16
R Dravid	75
W Jaffer	50
V Kohli	113
JH Kallis	89
CL White	78
MV Boucher	50
B Akhil	27
AA Noffke	9
P Kumar	34
Z Khan	23
SB Joshi	3
PA Patel	81

## QUERY 2

```
Select Player, max(Runs) as Highest_Score from maxscore group by Player having Player like "%dhoni";
```

## OUTPUT 2

Player	Highest_Score
MS Dhoni	70

# CONCLUSION

In this project, I built a MySQL database using the Indian Premier League (IPL) dataset. The dataset consisted of 6 comma-separated-value (.csv) files that were downloaded from the Kaggle.com. During my course of database management system, I learned about the basics of database design. I designed a database that can be used by cricket enthusiasts as well as data analysts to efficiently enlist data comparisons between players and teams. This can also be very helpful to people who are involved in games like Dream11, My11Circle, etc. This project gave me the opportunity to implement my knowledge of database and database management systems. While doing this project, I also gained deeper understanding on database design and how it can be implemented in real life situations.

# REFERENCES

- <https://www.kaggle.com/harsha547/indian-premier-league-csv-dataset> for dataset
- <https://app.diagrams.net/> for making an Entity-Relation Diagram
- <https://docs.python.org/3/> for pre-processing data
- [www.stackoverflow.com](http://www.stackoverflow.com) for queries