FINAL ASSIGNMENT REPORT

SCENARIO A

NOVEMBER 5, 2022

PRAMODA GUNARATHNE
20531218

Table Of Contents

1. Introduction

1.1. Purpose

Cricket is an excellent sport for improving general fitness, stamina, and hand-eye coordination. It is a popular sport in both Sri Lanka and other cricket-playing countries. With a database for management and storing large data with efficiency, persistence, and security the cricket database management system can be used to keep track of the data over time and keep logs of the data. Cricket sport is one of the most popular sports, hence preserving the data overtime is important and must be done to avoid future conflicts.

The purpose of this project is to analyse and establish a database management system(DBMS) for cricket tournaments. This cricket database management system can be used to maintain the records in a simpler way and can be used in further analytics in the sport for either management or in analysing the current tournament results. So, the primary goal of the project is to learn the fundamentals of database management systems and incorporate such a system for the sport that is cricket. This cricket database management system is a general-purpose system that allows the definition, creation, querying, update, alter, stored procedures and many more which will be discussed in the following topics.

1.2. Project Scope

An Internet-based information system with a database as the back end for information storage adds a new dimension, as the database can be used to maintain information consistency and to generate web pages. This cricket central database can be used as the back end for information storage, and a suitable user interface can be created later as the front-end for information browsing.

For this database management system, at international level there are 4 tournaments considered but the three mains for this management system will be T20, ODI and test match. With T20 matches are played for 20 overs per country. In T20 World Cup matches, each bowler is limited to four overs. At the start of each inning, a six-over powerplay is required. The powerplay will have field restrictions. In the first six overs, a maximum of two fielders may field outside the 30-yard circle *(Times of SportsA Leading Sports News Channel For All The Exclusive Sports Updates Across Globe. et al., 2022).*

In ODI (One Day International) matches are played for a quota of 50 overs per country and are either played during the day or during night. ODI cricket is also known as instant cricket because a match result is always available within seven to eight hours, unless weather conditions prevent it. The test match series is also considered in this database. The number of overs changes with every day and the minimum is 90 overs. There are two main websites used to get datasets for the database which is Kaggle.com *(Narayanan, 2020)* and data.world (*data.world.* 2021). These two websites provide data which is inserted for the tables created, most of the data used is not consistent with the datasets in the websites and sometimes used without context to different tables in the cricket database management system.

1.3. Overview

The system receives most of its input in the form of cricket scorecards, previously stored data analytics and current statues of the matches. Data management, retrieval, the creation of static or dynamic web pages, and the response to user requests are all aspects of information processing. The database query language, MySQL is used as the backend. The database system's output is the ability to answer frequently asked questions about cricket and to present statistical information for series, country, match, team statistics, batting records, bowling records, fielding, wicket keeping, player profile, and overall performance to users.

The ER diagram drawn is at a normalised standard. The logical design is more of like a formal design that is used for reference when creating the database as the relations and the names of the tables are used in the actual database implementation. The normalized relations aim to avoid redundant data and maintain data integrity when updating/deleting data (i.e., reducing anomalies) in related relations. The business rules are produced for more thorough understanding of the datasets gathered for the creation of the database or any rules that needs to be followed.

The implementation of the database includes several tables to get the best intended cricket database management system. Some of the tables are not that important but they are included for completion. This is explained in the Database Implementation section which thoroughly explains the concept of the design, assumptions and how tables are created. For each table, variety sets of data are inserted enough to get results of the queries written. This section also explains how to obtain results from the data inserted by using queries (25 queries in total). There are few stored procedures, triggers and views created and tested accordingly to provide an efficient amount of information regarding the advanced features. The language used to connect the database and obtain the same results from the queries are python3. Connecting the database with python3 language required the lecture 10 slides for referencing. All the sample results for the queries can be obtained by using the code. The last section reflects on my work with a summary of what I achieved and challenges I faced with ways to improve for further usage of the database system.

2. Design the Database

We must make the design of the database precise and normalized to the standard. Following the steps of creating the entity tables, cardinality constraint table for the relationship sets, participation constraint table for the relationship sets, entity diagram for the database, normalized relational database schema and data dictionary for attributes and business rules establish a firm outline to implement the database later.

2.1. Entity Relationship Diagram (ER Model)

An entity-relationship diagram, or Entity Relationship diagram, is required for modelling database data. It is the foundation on which a database is built. Entity Relationship diagrams define what data will be stored: entities and their attributes. They also demonstrate how entities are related to one another. To draw a precise entity diagram, first we must identify the entities in the database by gathering information by analysing the information gathered. After gathering the data, the relationship between the entities can be shown through the cardinality and participation tables. Finally, the complete ER Diagram is drawn.

The information to be taken as entities are gathered by reviewing the past records and current records. This also provides the relevant information about the data attributes too. A requirement gathering process is carried out and the information collect from this is presented below which helps to obtain a standardized ER diagram.

2.1.1. Entity table and participating sets

This table is used to identify the entities, primary key and their attributes accordingly so it makes it easier for us to draw the ER model. The entities and their respective attributes are gathered as follows and its assumptions are mentioned below:

Entity Sets	**Primary key/s**	**Other Attributes**
Tournament	TID	series, year, overs
Team [Parent/owner entity]	teamName	no_wins, no_loses, no_bowlers, no_batsmen, wicketkeeper, no_allRounders, tot_matches
Player [Weak entity]	playerNum [partial key]	pName, dob
Ranks	playerType	rating
Batsmen [Sub-type entity]		no_sixes, no_fours, tot_runs, batting_strike_rate
Bowler [Sub-type entity]		tot_wickets, no_balls, bowling_economy
Coach	CID	coachName
Umpire	UID	umpireName, no_of_standings
Inning	inningID	tot_score
Match	MID	first_team, second_team
Stadium	stadiumID	sName, city, country, capacity
Scoreboard	scoreID	scoreID, tot_wickets, tot_sixes, tot_fours, tot_runs, winner, runnerUp, man_of_match, highestScore

- Assume that the teamName is the country of the team.
- Assume that the wicketkeeper is a multi-valued attribute. Because in a team there can be more than 1 wicket keeper. This issue is solved during relational schema.
- Assume that batting_strike_rate and bowling_economy are derived attributes.
- Assume that the tot_score in Inning table is a derived attribute where the tot_score is calculated by the number of hits and runs the player achieved during the match.
- Here the player is a weak entity as it cannot exist without the team. Hence the primary key is a composite key.
- Assuming that batsmen and bowler is a sub type entity of super type player entity.
- Assuming that Batting strike rate is calculated as runs scored divided by balls faced into 100.
- Assuming that Bowling economy is calculated with runs conceded divided by overs bowled
- Assuming tot_matches are calculated from no_wins + no_loses. (no_draws does not count in this case)

The next step is to create the participating table to identify the relationship sets, between which entity sets they have a relationship and attributes of the relationship set if any. The relationship sets can also be included as tables in the database moving on, hence meaningful relationships are established.

Relationship sets	Between which entity sets	Attributes of relationship sets(if any)
Have	Tournament, Match	
Plays	Team, Match	date_time, scheduleID
Has A	Team, Player	
Mentored By	Team, Coach	
Scores In	Player, Inning	
Ranking	Player, Rank	
Umpired By	Match, Umpire	
At	Match, Stadium	
Rank In	Team, Tournament	ranking
Scored	Match, Scoreboard	
Manage	Player, captain_playerNum	

- Assume that the manage relationship is between player and captain where captain is a role of the player.

2.1.2. Cardinality Table

Relationships can include more information as constraints to provide a more accurate representation of the relationship between entity sets. A restriction that limits the number of relationship instances in which an entity can participate is known as cardinality constraint.

The following table gives the cardinality relationship between the relationship sets and the reasons:

Relationship set	Cardinality Constraint	Reason
Have	One-Many	One tournament can have many matches and many matches can be played in one tournament at a time.
Plays	Many-Many	Many matches can have many teams playing (basically 2)and many teams can play in more than one match in a series.
Has A	Many-One	The team has many players, but each player can only have one team.
Mentored By	One-Many	One team can be mentored by many coaches and many coaches can mentor one team.
Scores In	Many-One	Many players (typically 2) can score in one inning at a time. In one inning, many players can score.
Umpired by	One-Many	A match has many umpires but each umpire judges one match at a time.

At	One-One	One match can take place at one stadium at a time. A stadium can have one match at a time.
Rank In	Many-Many	Many teams can rank in many different tournaments. Many tournaments can have many teams ranked in the ranking board.
Scored	One-One	One match can have scores of one scoreboards at a given match. Each scoreboard is present in one match at a time.
Manage	Many-One	Many players can be managed by one captain. One captain from the player list can manage many players at a time.

- Assume that a team can have one coach at a time.
- Assume that the players that score in innings are the batsmen.(striker or non-striker)
- Assume that many umpires judge a single match at a time.
- Manages relationship is a urinary relationship.
- Here, we place the relationships by using "Look across"

2.1.3. Participation Table

A constraint defining the smallest number of entities that can be associated with another entity via a meaningful relationship is known as participation constraint.

The following table gives the participation relationship between the relationship sets and the reasons:

Relationship set	**Participation Constraint**	**Reason**
Have	Tournament – partial, match – partial	A match can exist without a tournament present, and a tournament may exist without having a match.
Plays	Match - total, Team – partial	A match cannot exist without at least one team. A team can exist without a match.
Has A	Team – partial, Player- total	A player cannot exist without a team and a team can exist without players.
Mentored By	Team – total, coach - partial	At least one team is mentored by one coach, but a coach can exist without a team.
Scores In	Player- partial, Inning- total	A player can exist without having to score in an inning, but an inning cannot exist without at least one player scoring in an inning.

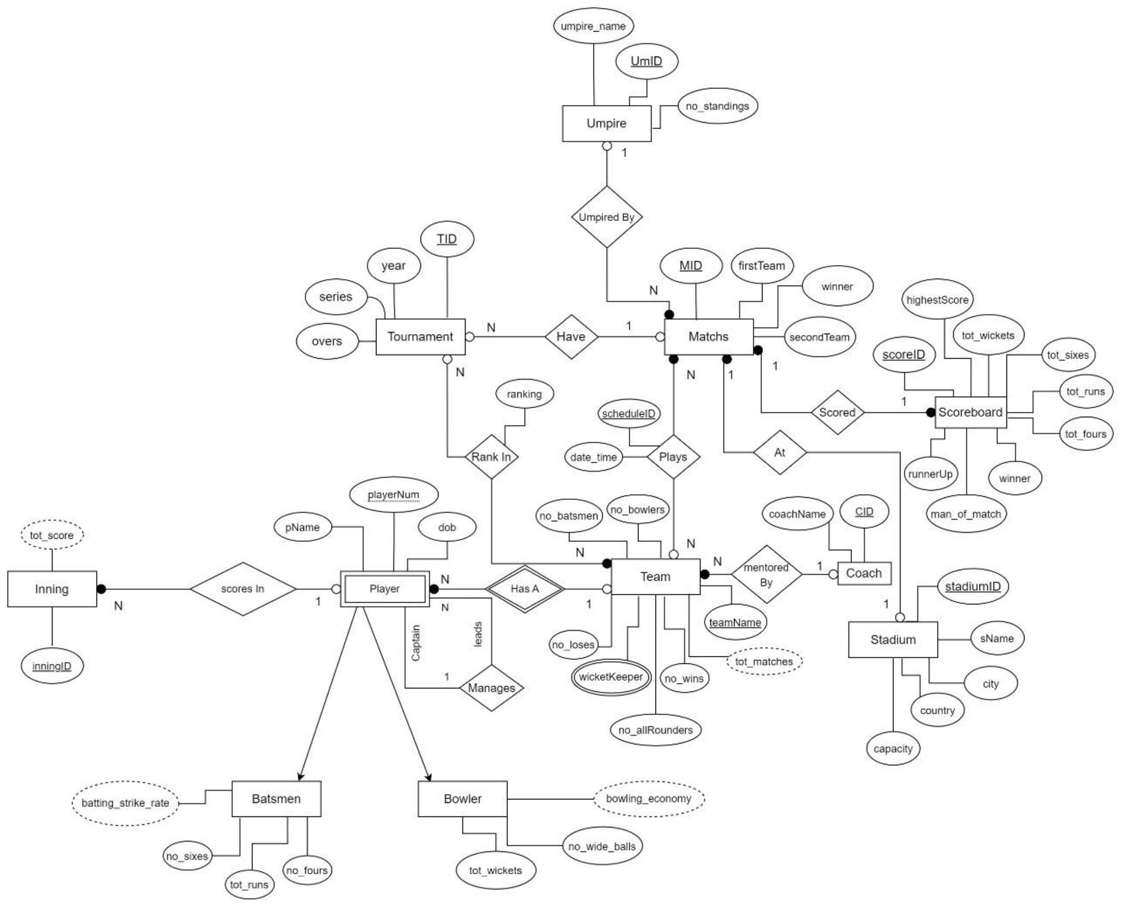
Umpired By	Match- total, umpire - partial	A match must be umpired by an umpire, but an umpire can exist without umpiring a match.
At	Match – total, stadium- partial	A match cannot exist without a stadium to play at, but a stadium can exist without a match.
Rank In	Team – total, Tournament- partial	A participating team cannot exist without a rank in tournament series, but a tournament series can exist without at least one team.
Scored	Match – total, scoreboard- total	A match cannot exist without at least one scoreboard, but a scoreboard cannot exist without a match to score on.

- Previous assumptions are considered here as well.
- Assume that urinary relationships cannot have participation constraint. But if it were to have a team can exit without a captain but a captain cannot exist without a team to manage.
- Here we place the constraints using "Look here"

2.1.4. Entity Relationship diagram

The entity relationship diagram is drawn using Chen's notation. The ER model employs graphical representation to depict its constructs and their relationships. The below diagram shows the ER diagram of the cricket database management system. The ER diagram needs some twerking to be done during relational schema.

The below ER diagram shows all the entities, data attributes, primary keys, relationships. Cardinality constraint and participation constraint.



The multivalued attribute in the ER diagram states that the ER diagram is in 1St normal form. This is refined during the relational schema.

2.2. Relational Database Schema

The process of converting an ER model to a relational schema is known as mapping the ER model to relational schema, where the goal is to avoid anomalies and redundancy. Using well-established guidelines, we convert an ER model to a proper relational schema. The ER diagram above is matched to its relational schema. The initial relational schema that is arrived at, required normalization as the relational schema doesn’t support multivalued attributes where the final relational schema is at least in third normal form (3NF). All the tables have passed 1st normal form.

With relational schema we consider primary keys, composite keys (a key that has more than one key as the primary key), and foreign keys. With foreign keys both the relations must belong to the same domain where the primary key should have the same datatype. Also, the relations should be able to talk with the existing primary keys between the 2 tables. The foreign keys are used to create links/ relationships between tables to avoid inconsistency and maintain integrity of data.

The domain constraints are considered where the datatypes are selected carefully, the entity integrity states that the primary keys cannot be null and referential integrity constraint are also stated properly to connect certain tables.

The following steps shows how the final relational schema is arrived at:

NOTE: Highlighted by yellow means foreign key and underlined means primary key.

- Tournament(TID, year, series, overs)
- Matchs(MID, firstTeam, secondTeam, UmID, stadiumID, TID)

  FK UmID REF Umpire(UmID)

  FK stadiumID REF Stadium(stadiumID)

  FK TID REF Tournament(TID)

- Umpire(UmID, umpire_name, no_standings)
- Stadium(stadiumID, sName, city, country, capacity)

- In scoreboard the winner, runnerUp and man_of_match attributes are functionally dependent on highestScore and MID. Two non-prime attributes determine the non-prime winner attribute and man_of_match attribute.

Scoreboard(scoreId, tot_wickets, tot_sixes, tot_runs, tot_fours, winner, runnerUp, man_of_match, highestScore, MID)

- So, the winner, runnerUp, MID and highestScore can be taken into a separate table known as Achievement table where highestScore and MID attributes become the primary key (composite key)to winner and man_of_match attributes. And the scoreboard table contains the rest of the attributes. Now, it is in 3NF.

  - Scoreboard(scoreID, tot_wickets, tot_sixes, tot_runs, tot_fours, highestScore, MID)

    FK MID REF Match(MID)

  - Achievement(highestScore, MID , winner, man_of_match, runnerUp)

- Initially the Team table is in 1NF as there is a multi-valued attribute that is wicketKeeper. This is taken into a separate table with player number and teamName both as primary keys forming a composite key.
  - Team(teamName, no_wins, no_loses, no_bowlers, no_batsmen, no_allRounders, tot_matches, CID)

    FK CID REF Coach(CID)

- Assume that tot_matches is written to the team table using an alert and update query later after the Team table is created
  - wicketkeeper(keeper_playerNum, teamName)

    FK keeper_playerNum REF Player(playerNum)

- Plays(scheduleID, MID, winTeam, date_time)

  FK MID REF Match(MID)

  FK winTeam REF Team(teamName)

- Coach(CID, coachName)
- RankIn(TID, teamName, ranking)

  FK TID REF Tournament(TID)

  FK teamName REF Team(teamName)

- Player(playerNum,teamName, pName, dob, captain)

  FK teamName REF Team(teamName)

- Inning(inningID, tot_score, playerNum, teamName)

  FK playerNum REF Player(playerNum)

  FK teamName REF Team(teamName)

- Batsmen(batting_strike_rate, no_sixes, no_fours, no_runs, playerNum, teamName)

  FK playerNum REF Player(playerNum)

  FK teamName REF Team(teamName)

- Bowler(bowling_economy, tot_wickets, no_balls, playerNum, teamName)

  FK playerNum REF Player(playerNum)

  FK teamName REF Team(teamName)

Assumptions

- Since the relationship between match and stadium is 1:1 and the most suitable table to add the foreign key was Match as the match needs to know the stadiumID to keep a match.
- Since the participation constraint between the 1:1 relationship in match and scoreboard, both the entities are written.
- Wicketkeeper is a multivalued attribute therefore it's in first normal form and hence this attribute is taken as a separate table from Teams table where the composite key becomes the primary key.
- The N:N relationship between match and team leads to plays as a new table with 2 foreign keys.
- Player is a weak entity therefore the primary key is a composite key which is the partial key + owner entity's primary key.
- In unary relationship, the role is written inside the entity.
- In sub-types, the super type entity's primary key becomes the sub-type's foreign key and primary key.
- There are 15 tables in total after refinement and normalization.

The above relational schema is in 3rd normal form and can be used as a reference to implement the database system using MySQL in Linux environment. The mapping of conceptual database design to logical database design is complete.

2.3. Data description

To properly implement the database, more information such as data types of attributes and constraints (mostly business rules) are required. Such data is typically derived during the ER modelling stage (for example – the data collected in part 2.1. can be used to further describe the attributes). These data descriptions are called data dictionary or data catalogues.

2.3.1. Data Dictionary

Below diagrams shows the data descriptions of each database table in tabular format for more further easiness of implementation of the database system. For each valid table in the database, we have data dictionary tables.

Primary keys and foreign keys are indicated as not null.

1. The Tournament table:

Used to store tournament data with different series

Attribute	**Type**	**Size**	**Null**	**Primary Key**	**Description**	**Other constraints**
TID	char	4	N	Y	It is a digit used to identify a tournament	
series	varchar	12	N		Name of the series	
year	char	4	Y		Year of the series started	
overs	char	3	Y		Number of overs in the series	

- Assuming the starting date and end date of any tournament starts at the beginning of month January and ends in the beginning of month November.
- The year of the series is ranging from 1990s to 2022.
- Assuming that only one series can take place for one year except T10. Therefore, years that T10 matches were held other series were also held.

2. The Matchs table:

Keeps the data of the relevant matches in a series

Attribute	**Type**	**Size**	**Null**	**Primary Key**	**Description**	**Other constraints**

MID	char	4	N	Y	Match ID	
firstTeam	varchar	24	N		Name the participating team	
secondTeam	varchar	24	N		Name of the other participating team	
UmID	Char	6	N		Umpire ID	Foreign key
stadiumID	Char	6	N		Stadium ID	Foreign key
TID	Char	6	N	Y	It is a digit used to identify a tournament	Foreign key

- Assuming that multiple matches are played in a tournament series before finals.

3. The Umpire table:

Keeps the data of the umpires involved in each match

Attribute	**Type**	**Size**	**Null**	**Primary Key**	**Description**	**Other constraints**
UmID	char	2	N	Y	Umpire ID	
umpire_name	varchar	20	N		Name of the umpire	
no_standings	Int	4	Y			

- Assuming that no of standings means how many matches the umpire has judged in total.

4. The stadium table:

Contains the information about the stadiums that matches are held in.

Attribute	**Type**	**Size**	**Null**	**Primary Key**	**Description**	**Other constraints**
stadiumID	char	6	N	Y	stadium ID	
sName	varchar	24	N		Name of the stadium	
city	Varchar	18	Y		Name of the city	
country	Varchar	18	Y		Country the stadium is located	

capacity	Int	20	Y		Limit	

5. The Scoreboard table:
   Keeps the information about the scores the winning team scored after a match is played.

Attribute	**Type**	**Size**	**Null**	**Primary Key**	**Description**	**Other constraints**
scoreID	char	6	N	Y	score ID	
tot_wickets	Int	4	Y		Total wickets obtained	
tot_runs	Int	4	Y		Total runs obtained	
tot_sixes	Int	4	Y		Total number of sixes	
tot_fours	Int	4	Y		Total number of fours	
highScore	Int	12	N		Highest score	
MID	char	4	N		Match ID	Foreign key

6. Achievement table

Keeps the details of the team that won the match.

Attribute	**Type**	**Size**	**Null**	**Primary Key**	**Description**	**Other constraints**
highestScore	Int	12	N	Y	Highest score	
winner	varchar	24	Y		Name of the team	
runnerUp	Varchar	24	N		Runner up team name	
man_of_match	Varchar	24	Y		Name of the player	

7. The team table:

Keeps the data of the current teams playing in a match

Attribute	**Type**	**Size**	**Null**	**Primary Key**	**Description**	**Other constraints**
teamName	Varchar	24	N	Y	Team Name	
no_wins	Int	4	Y		Total wins	
no_loses	Int	4	Y		Total loses	
no_bowlers	Int	4	Y		number of bowlers	

no_batsmen	Int	4	Y		Number of batsmen	
no_allRounders	Int	4	Y		Number of all rounders	
tot_matches	Int	8	N		Total matches	
CID	char	4	N		Coach ID	Foreign key

- Team Name is unique as I have considered the team's name to be the country of the team therefore only one country can exist in the world hence team ID is not mandatory.
- Assume that no of losses, no of wins are the records obtained as a total value throughout the years.

8. The Wicketkeeper table:

This table keeps information about the wicket keepers.

Attribute	**Type**	**Size**	**Null**	**Primary Key**	**Description**	**Other constraints**
Keeper_playerNum	Int	6	N	Y	Wicket Keeper name	
teamName	varchar	24	N	Y	Name of the team	

- Assuming there are 2 wicket keepers per team.

9. The plays table:

Keeps the data about the participating teams in a current match.

Attribute	**Type**	**Size**	**Null**	**Primary Key**	**Description**	**Other constraints**
scheduleID	Int	3	N	Y	Schedule ID	
MID	char	4	N	Y	Match ID	Foreign key
winTeam	varchar	24	N	Y	Name of the winning team	Foreign key
date_time	Datetime		Y		Date and time of whe the match is to start	

10. The coach table:

Keeps information about the coaches in each team.

Attribute	Type	Size	Null	Primary Key	Description	Other constraints
CID	char	4	N	Y	Coach ID	
coachName	varchar	24	N		Name of the Coach	

- Assume that one coach can mentor one team.

11. The RankIn table:

Keeps the ranks of the teams in each tournament.

Attribute	Type	Size	Null	Primary Key	Description	Other constraints
TID	char	6	N	Y	Tournament ID	Foreign key
teamName	varchar	24	N	Y	Name of the team	Foreign key
ranking	Int	5	Y		Rank of the team	

12. The player table:

Keeps information about each player in each team.

Attribute	Type	Size	Null	Primary Key	Description	Other constraints
playerNum	Int	5	N	Y	Player number	
teamName	varchar	24	N	Y	Name of the team	Foreign key
pName	Varchar	24	N		Name of the player	
dob	Date		Y		Date of birth	
Captain	Char	3	Y		Captain of the team	

- Assuming that player number and team name, both used to identify a player uniquely.
- Assuming that the captain is an attribute that states whether the cricket player is captain or not using YES or NULL.

13. The Inning table:

Holds the data of player’s stats in each inning during a match

Attribute	**Type**	**Size**	**Null**	**Primary Key**	**Description**	**Other constraints**
inningID	Char	5	N	Y	Inning ID	
playerNum	Int	5	N		Player number	Foreign key
teamName	Varchar	24	N		Team name	Foreign Key
tot_score	Int	6	Y		Highest score	

- The score is the score obtained by the player during each inning.

14. The Batsmen table:

Keeps the batsmen data after a match is played.

Attribute	**Type**	**Size**	**Null**	**Primary Key**	**Description**	**Other constraints**
playerNum	Int	16	N	Y	Player number	Foreign key
teamName	Varchar	24	N		Team name	Foreign Key
batting_strike_rate	Decimal	(5,2)	Y		Batting rate	
no_sixes	Int	4	Y		Number of sixes	
no_fours	Int	4	Y		Number of fours	
no_runs	Int	4	Y		Number of runs	

15. The bowler table:

Keeps the bowler's data after a match is played.

Attribute	**Type**	**Size**	**Null**	**Primary Key**	**Description**	**Other constraints**
playerNum	Int	16	N	Y	Player number	Foreign key
teamName	Varchar	24	N		Team name	Foreign Key
bowling_economy	Decimal	(5,2)	Y		Bowling rate	
tot_wickets	Int	4	Y		Number of wickets	
no_balls	Int	4	Y		Number of no balls	

- The bowler and the batsmen table records are obtained from the current match being played.

2.3.2. Business rules

A business rule is a statement that imposes some kind of constraint on a specific aspect of the database, such as the elements of a field specification or the characteristics of a given relationship. The business rules found in the cricket database management system will be given as below.

Business rules	**Description**
BR1	A team must have 16 players before playing in a match
BR2	T20 can only have 20 overs at a time
BR3	ODI can have 50 overs in a match
BR4	Test match can have a minimal of 90 overs
BR5	At least 4 bowlers must be present in a team
BR6	At least 5 batsmen must be present in the team
BR7	At least 2 wicket keepers must be there in a team
BR8	There are 6 balls in an over
BR9	There can be more than one umpire
BR10	There can be only two teams playing in one match
BR11	The cricket matches cannot last more than 6 hours except if weather affects the match
BR12	Only one series take place during the period of that year except in years where T10 series is played.
BR13	All the teams completing must have a certain rank in the RankIn table.
BR14	At least one batsman must play in an inning.
BR15	All the players must have 2 + years of experience in the field.
BR16	There must be at least 9 fielders expect the wicket keeper and bowler.

3. Database Implementation

Implementation of the database is seemly easy as the logical database design is obtained correctly in section 2. The final schema and the data descriptions are used to create the physical database design. In this section creation of the database and insertion of the valid data are done accurately. All the query results and commands are logged into a file called CricketDB_20531218.out so anyone can view the results and use for comparison with the outputs they received.

The name of the database is CricketDB_20531218, and it is created using:

CREATE DATABASE CricketDB_20531218;

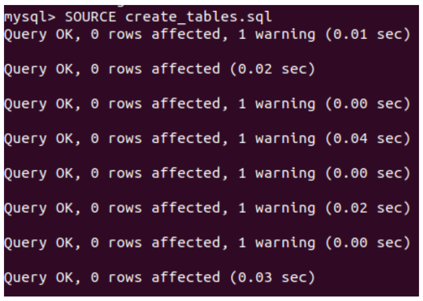
And before performing data definition language and data manipulation language we must use the database:

USE CricketDB_20531218;

NOTE: Every command from create table to advanced queries are all written in SQL files.

3.1. Creation Of Database

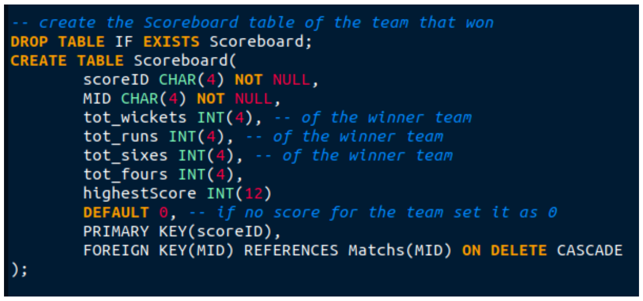
The create_tables.sql file contains the data definition language queries which means it contains the queries to create the 15 tables. The tables are created in order considering the referential integrity constraints. To write the tables into the database we can source the SQL file.



```
mysql> SOURCE create_tables.sql
Query OK, 0 rows affected, 1 warning (0.01 sec)

Query OK, 0 rows affected (0.02 sec)

Query OK, 0 rows affected, 1 warning (0.00 sec)

Query OK, 0 rows affected, 1 warning (0.04 sec)

Query OK, 0 rows affected, 1 warning (0.00 sec)

Query OK, 0 rows affected, 1 warning (0.02 sec)

Query OK, 0 rows affected, 1 warning (0.00 sec)

Query OK, 0 rows affected (0.03 sec)
```

Here, it shows the file being SOURCED and the queries being entered.

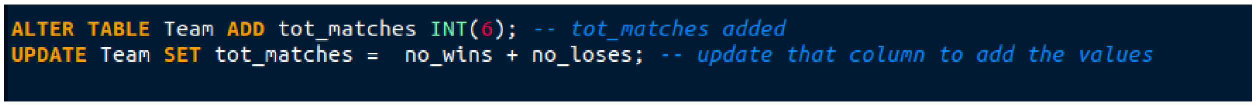
A look at the create_tables.sql file:



```
-- create the Scoreboard table of the team that won
DROP TABLE IF EXISTS Scoreboard;
CREATE TABLE Scoreboard(
        scoreID CHAR(4) NOT NULL,
        MID CHAR(4) NOT NULL,
        tot_wickets INT(4), -- of the winner team
        tot_runs INT(4), -- of the winner team
        tot_sixes INT(4), -- of the winner team
        tot_fours INT(4),
        highestScore INT(12)
        DEFAULT 0, -- if no score for the team set it as 0
        PRIMARY KEY(scoreID),
        FOREIGN KEY(MID) REFERENCES Matchs(MID) ON DELETE CASCADE
);
```

Here it shows the scoreboard table where the highestScore is set for a default value of 0 if it is not inserted manually. To look further into the table, please look at the create_tables.sql file. Inside the file the order is maintained considering the foreign key constraints.

NOTE: The create query written for Team table doesn't include tot_matches column as that would be added as a data manipulation query to the database as follows in a more advanced approach. These two queries can be found in the Queries.sql file, make sure to run them before querying from the team table.



```
ALTER TABLE Team ADD tot_matches INT(6); -- tot_matches added
UPDATE Team SET tot_matches =  no_wins + no_loses; -- update that column to add the values
```

3.2. Insertion To the Database

A variety and different sets of data were inserted to each table. For some of the important tables like player, team, match, scoreboard, tournament a reasonable amount of data was inserted such as more than 20 data sets for player table. This is useful when retrieving data using different types of queries in section 4. For the tables that include foreign constraints the data are inserted accordingly and meaningfully where the foreign key constraints are unharmed. The references used for insertion are under reference list.

The order to run these files and insert the tables properly with consideration to the foreign keys is as follows:

insertOne.sql → insertTwo.sql → insertThree.sql.

The following section briefly describes the sample data inserted.

- For tournament table the data inserted ranges from the year 1984-2022. It contains 20 data sets with different aspects being covered from T10, T20, ODI, Test Match.
- For the umpire table, there are 15 different data sets entered where umpires can judge different matches in different tournaments. The same umpire is used to judge a match in a different tournament series.
- For stadium table there are 11 stadiums entered that gives a variety of information about the cricket stadiums available.
- In coach table, enough data is entered so one team can have one coach. All the data entered are valid.
- For team table, there are 16 main teams in the cricket world and their team details are entered accordingly.
- In player table, suffice data is entered with at least 2 (in most cases 3) players from each team. The captain column shows only few captains, and the rest are NULL.
- All the wicket keepers (generally 2) are entered to the WicketKeeper table from each team. These wicket Keepers are players that already exist in the player table or players that are not in Player table.
- There are 21 matches entered in total for tournaments ranging from T1-T20.
- Plays tables contain the schedule the team that won the match with date and time.
- The scoreboard gives details of each match that is entered. Some data rows contain NULL values for tot_sixes.
- The achievement table gives the sample data where which team has won the match and which team is the runner up.
- The batsmen and the bowler tables have the calculated attributes, which are batting_strike_rate and bowling_economy. The players with high batting_strike _rate is relatively a good player where has a player who has a lower bowling_economy means they are a good bowler.
- The inning table contain a variety of data showing each player's total score in each team.
- The RankIn table shows all the ranks of some of the teams in each tournament series.

4. Usage Of the Database

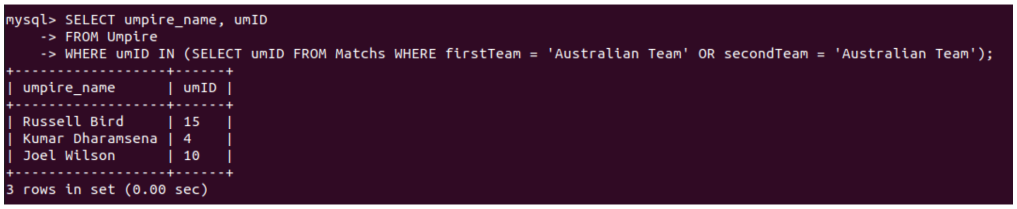
There are a variety of queries written to retrieve enough data from the Cricket database. Different types of queries are written to retrieve data from the tables, such as retrieve information by performing string manipulation, date-time functions, joins, sub queries and more. Retrieval of information from the database can be used for analysing old data with current data and come into conclusions. The administrative management can retrieve information from tables to perform audits and for management purposes.

4.1. Design and Implementing of Queries

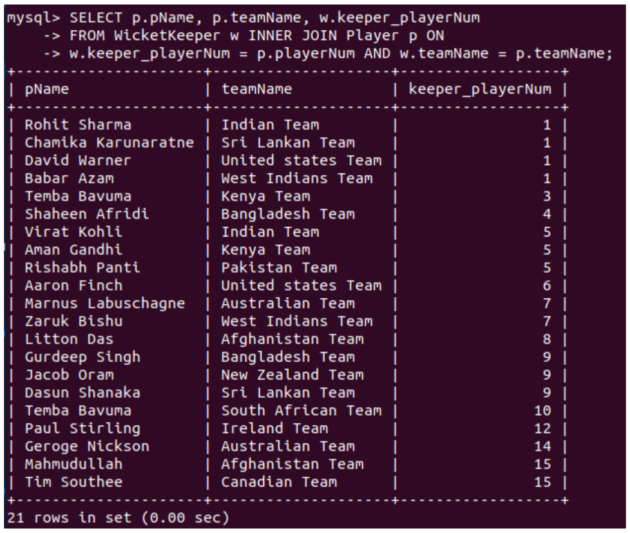
This section contains few queries that are written to retrieve results from the database. The queries are written in queries.sql file and it can be scored to the database to see the results. There are 25 queries written in total but in this section only 11 of the queries are explained.

1. Display umpire details who have umpired matches where one of the teams is the Australian team by using a sub query.

This query is used to get umpire details to compare them with the other umpire details in the umpire table who have not judged a match where Australian team was playing.


```
mysql> SELECT umpire_name, umID
    -> FROM Umpire
    -> WHERE umID IN (SELECT umID FROM Matchs WHERE firstTeam = 'Australian Team' OR secondTeam = 'Australian Team');
+-----------------+------+
| umpire_name     | umID |
+-----------------+------+
| Russell Bird    | 15   |
| Kumar Dharamsena | 4   |
| Joel Wilson     | 10   |
+-----------------+------+
3 rows in set (0.00 sec)
```

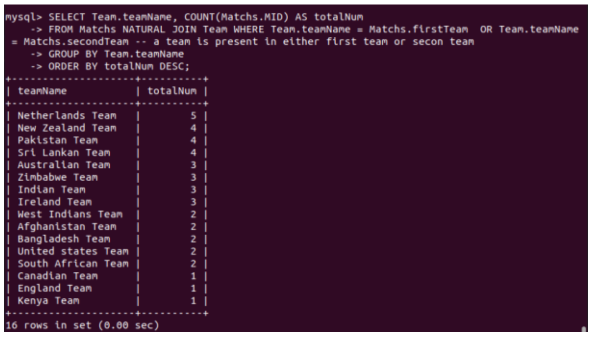
2. Display Wicket keeper details of each team with their player details using Inner join query.


```
mysql> SELECT p.pName, p.teamName, w.keeper_playerNum
    -> FROM WicketKeeper w INNER JOIN Player p ON
    -> w.keeper_playerNum = p.playerNum AND w.teamName = p.teamName;
+--------------------+--------------------+------------------+
| pName              | teamName           | keeper_playerNum |
+--------------------+--------------------+------------------+
| Rohit Sharma       | Indian Team        |                1 |
| Chamika Karunaratne | Sri Lankan Team   |                1 |
| David Warner       | United states Team |                1 |
| Babar Azam         | West Indians Team  |                1 |
| Temba Bavuma       | Kenya Team         |                3 |
| Shaheen Afridi     | Bangladesh Team    |                4 |
| Virat Kohli        | Indian Team        |                5 |
| Aman Gandhi        | Kenya Team         |                5 |
| Rishabh Panti      | Pakistan Team      |                5 |
| Aaron Finch        | United states Team |                6 |
| Marnus Labuschagne | Australian Team    |                7 |
| Zaruk Bishu        | West Indians Team  |                7 |
| Litton Das         | Afghanistan Team   |                8 |
| Gurdeep Singh      | Bangladesh Team    |                9 |
| Jacob Oram         | New Zealand Team   |                9 |
| Dasun Shanaka      | Sri Lankan Team    |                9 |
| Temba Bavuma       | South African Team |               10 |
| Paul Stirling      | Ireland Team       |               12 |
| Geroge Nickson     | Australian Team    |               14 |
| Mahmudullah        | Afghanistan Team   |               15 |
| Tim Southee        | Canadian Team      |               15 |
+--------------------+--------------------+------------------+
21 rows in set (0.00 sec)
```

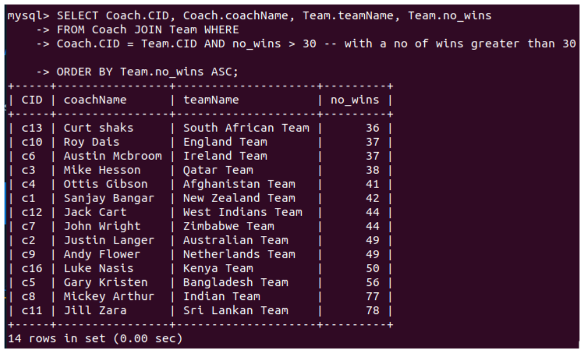
The query can be used to compare with the players who are not wicket keepers in each team as every team have two wicket keepers.

Can be used by administrators to identify who are wicket the keepers in each team from the database.

3. Display which teams have played in matches by counting the match ID and display their team names using a Natural join and sort the total Number from descending order.

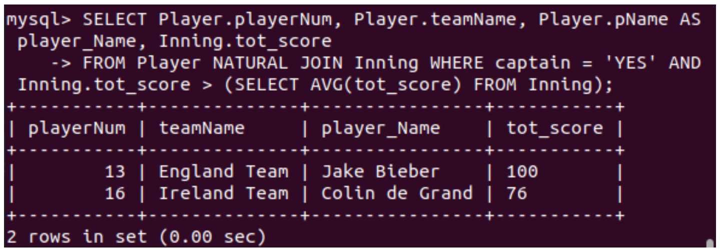

```
mysql> SELECT Team.teamName, COUNT(Matchs.MID) AS totalNum
    -> FROM Matchs NATURAL JOIN Team WHERE Team.teamName = Matchs.firstTeam  OR Team.teamName
 = Matchs.secondTeam -- a team is present in either first team or secon team
    -> GROUP BY Team.teamName
    -> ORDER BY totalNum DESC;
+--------------------+----------+
| teamName           | totalNum |
+--------------------+----------+
| Netherlands Team   |        5 |
| New Zealand Team   |        4 |
| Pakistan Team      |        4 |
| Sri Lankan Team    |        4 |
| Australian Team    |        3 |
| Zimbabwe Team      |        3 |
| Indian Team        |        3 |
| Ireland Team       |        3 |
| West Indians Team  |        2 |
| Afghanistan Team   |        2 |
| Bangladesh Team    |        2 |
| United states Team |        2 |
| South African Team |        2 |
| Canadian Team      |        1 |
| England Team       |        1 |
| Kenya Team         |        1 |
+--------------------+----------+
16 rows in set (0.00 sec)
```

This table can be used to verify how many teams have participated in matches. Only the Nepal team hasn't participated in any match yet.


```
mysql> SELECT Coach.CID, Coach.coachName, Team.teamName, Team.no_wins
    -> FROM Coach JOIN Team WHERE
    -> Coach.CID = Team.CID AND no_wins > 30 -- with a no of wins greater than 30

    -> ORDER BY Team.no_wins ASC;
+-----+----------------+--------------------+---------+
| CID | coachName      | teamName           | no_wins |
+-----+----------------+--------------------+---------+
| c13 | Curt shaks     | South African Team |      36 |
| c10 | Roy Dais       | England Team       |      37 |
| c6  | Austin Mcbroom | Ireland Team       |      37 |
| c3  | Mike Hesson    | Qatar Team         |      38 |
| c4  | Ottis Gibson   | Afghanistan Team   |      41 |
| c1  | Sanjay Bangar  | New Zealand Team   |      42 |
| c12 | Jack Cart      | West Indians Team  |      44 |
| c7  | John Wright    | Zimbabwe Team      |      44 |
| c2  | Justin Langer  | Australian Team    |      49 |
| c9  | Andy Flower    | Netherlands Team   |      49 |
| c16 | Luke Nasis     | Kenya Team         |      50 |
| c5  | Gary Kristen   | Bangladesh Team    |      56 |
| c8  | Mickey Arthur  | Indian Team        |      77 |
| c11 | Jill Zara      | Sri Lankan Team    |      78 |
+-----+----------------+--------------------+---------+
14 rows in set (0.00 sec)
```

4. Retrieve coach details who have coached teams where no of wins is greater than 30. Number of wins are sorted in ascending order as shown on the left.

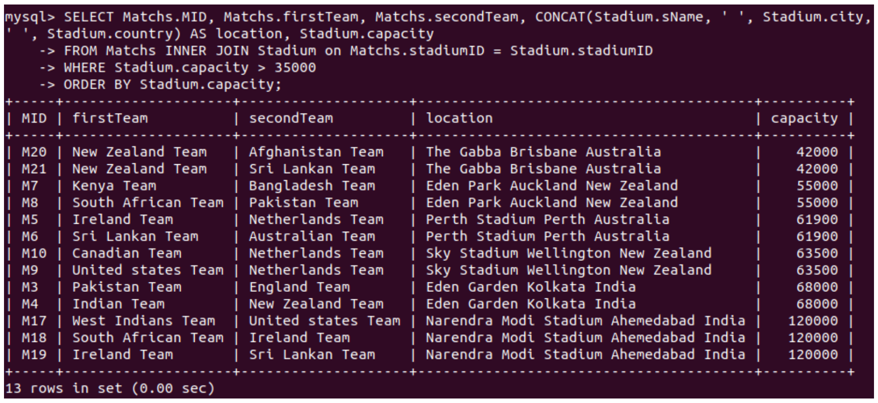
Only 14 rows are retrieved as the rest of the 6 teams have number of wins less than 30 so they are not taken into the result query.



```
mysql> SELECT Player.playerNum, Player.teamName, Player.pName AS
 player_Name, Inning.tot_score
    -> FROM Player NATURAL JOIN Inning WHERE captain = 'YES' AND
 Inning.tot_score > (SELECT AVG(tot_score) FROM Inning);
+-----------+-------------+----------------+-----------+
| playerNum | teamName    | player_Name    | tot_score |
+-----------+-------------+----------------+-----------+
|        13 | England Team | Jake Bieber   | 100       |
|        16 | Ireland Team | Colin de Grand | 76       |
+-----------+-------------+----------------+-----------+
2 rows in set (0.00 sec)
```

5. List down the players who are captains in the teams with a total score less than the average total score in innings played using natural join and a subquery to obtain the average total score.

These results can be used by the administrators to get captain details where they have obtained a total score greater than the average score.

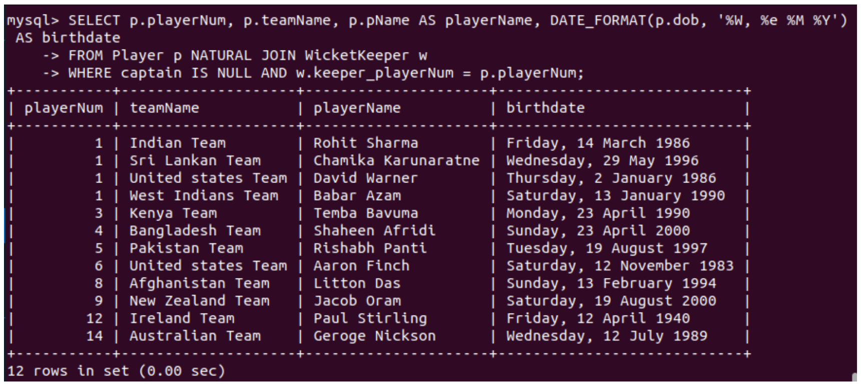


```
mysql> SELECT Matchs.MID, Matchs.firstTeam, Matchs.secondTeam, CONCAT(Stadium.sName, ' ', Stadium.city,
' ', Stadium.country) AS location, Stadium.capacity
    -> FROM Matchs INNER JOIN Stadium on Matchs.stadiumID = Stadium.stadiumID
    -> WHERE Stadium.capacity > 35000
    -> ORDER BY Stadium.capacity;
+-----+--------------------+--------------------+-----------------------------------------+----------+
| MID | firstTeam          | secondTeam         | location                                | capacity |
+-----+--------------------+--------------------+-----------------------------------------+----------+
| M20 | New Zealand Team   | Afghanistan Team   | The Gabba Brisbane Australia            |    42000 |
| M21 | New Zealand Team   | Sri Lankan Team    | The Gabba Brisbane Australia            |    42000 |
| M7  | Kenya Team         | Bangladesh Team    | Eden Park Auckland New Zealand          |    55000 |
| M8  | South African Team | Pakistan Team      | Eden Park Auckland New Zealand          |    55000 |
| M5  | Ireland Team       | Netherlands Team   | Perth Stadium Perth Australia           |    61900 |
| M6  | Sri Lankan Team    | Australian Team    | Perth Stadium Perth Australia           |    61900 |
| M10 | Canadian Team      | Netherlands Team   | Sky Stadium Wellington New Zealand      |    63500 |
| M9  | United states Team | Netherlands Team   | Sky Stadium Wellington New Zealand      |    63500 |
| M3  | Pakistan Team      | England Team       | Eden Garden Kolkata India               |    68000 |
| M4  | Indian Team        | New Zealand Team   | Eden Garden Kolkata India               |    68000 |
| M17 | West Indians Team  | United states Team | Narendra Modi Stadium Ahemedabad India  |   120000 |
| M18 | South African Team | Ireland Team       | Narendra Modi Stadium Ahemedabad India  |   120000 |
| M19 | Ireland Team       | Sri Lankan Team    | Narendra Modi Stadium Ahemedabad India  |   120000 |
+-----+--------------------+--------------------+-----------------------------------------+----------+
13 rows in set (0.00 sec)
```

6. Display matches details and stadium details where a match is played in a stadium capacity greater than 35000.

This query results in 13 rows where the rest of the data rows are not taken into account as their capacity is less than 35000. The location attribute taken into the table is concatenation of the stadium name, city, and country attributes.

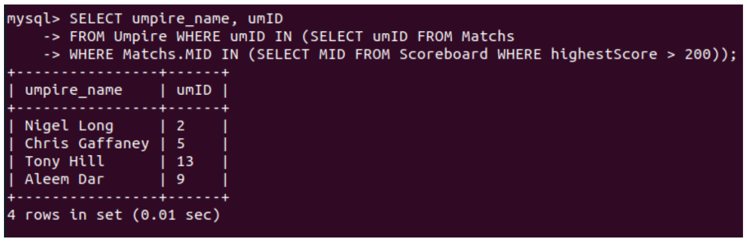
7. Produce a list of players who work is a wicket player but not a captain, showing player number, team name, player name and birth date. Show only the date and month of the birth date in the form of Friday, 6 August 2022.



```
mysql> SELECT p.playerNum, p.teamName, p.pName AS playerName, DATE_FORMAT(p.dob, '%W, %e %M %Y')
 AS birthdate
    -> FROM Player p NATURAL JOIN WicketKeeper w
    -> WHERE captain IS NULL AND w.keeper_playerNum = p.playerNum;
+-----------+--------------------+---------------------+--------------------------+
| playerNum | teamName           | playerName          | birthdate                |
+-----------+--------------------+---------------------+--------------------------+
|         1 | Indian Team        | Rohit Sharma        | Friday, 14 March 1986    |
|         1 | Sri Lankan Team    | Chamika Karunaratne | Wednesday, 29 May 1996   |
|         1 | United states Team | David Warner        | Thursday, 2 January 1986 |
|         1 | West Indians Team  | Babar Azam          | Saturday, 13 January 1990 |
|         3 | Kenya Team         | Temba Bavuma        | Monday, 23 April 1990    |
|         4 | Bangladesh Team    | Shaheen Afridi      | Sunday, 23 April 2000    |
|         5 | Pakistan Team      | Rishabh Panti       | Tuesday, 19 August 1997  |
|         6 | United states Team | Aaron Finch         | Saturday, 12 November 1983 |
|         8 | Afghanistan Team   | Litton Das          | Sunday, 13 February 1994 |
|         9 | New Zealand Team   | Jacob Oram          | Saturday, 19 August 2000 |
|        12 | Ireland Team       | Paul Stirling       | Friday, 12 April 1940    |
|        14 | Australian Team    | Geroge Nickson      | Wednesday, 12 July 1989  |
+-----------+--------------------+---------------------+--------------------------+
12 rows in set (0.00 sec)
```

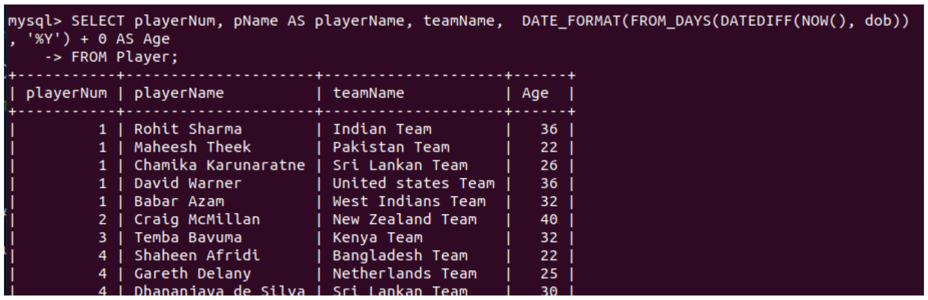
This query can be used to obtain wicket keeper player details where their date of birth is presented in the format of Friday, 6 August 2022. This result from the query can be used to compare with the query results obtained in query 2.

8. Display the umpire details who have umpired matches that have a high score of more than 200 by using two sub queries. The first query is used to get match IDs with the high score greater than 200 where the second query is written to get the umpire IDs from the match table and display the umpire details. Used to compare with the results where highest score in a match is less than 200.



```
mysql> SELECT umpire_name, umID
    -> FROM Umpire WHERE umID IN (SELECT umID FROM Matchs
    -> WHERE Matchs.MID IN (SELECT MID FROM Scoreboard WHERE highestScore > 200));
+----------------+------+
| umpire_name    | umID |
+----------------+------+
| Nigel Long     | 2    |
| Chris Gaffaney | 5    |
| Tony Hill      | 13   |
| Aleem Dar      | 9    |
+----------------+------+
4 rows in set (0.01 sec)
```

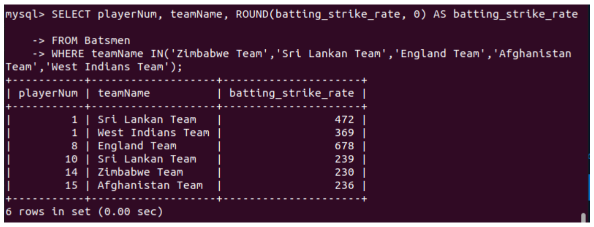
9. Data manipulation query is written to get the age of all the players from the player table.



```
mysql> SELECT playerNum, pName AS playerName, teamName,  DATE_FORMAT(FROM_DAYS(DATEDIFF(NOW(), dob))
, '%Y') + 0 AS Age
    -> FROM Player;
+-----------+---------------------+--------------------+------+
| playerNum | playerName          | teamName           | Age  |
+-----------+---------------------+--------------------+------+
|         1 | Rohit Sharma        | Indian Team        |   36 |
|         1 | Maheesh Theek       | Pakistan Team      |   22 |
|         1 | Chamika Karunaratne | Sri Lankan Team    |   26 |
|         1 | David Warner        | United states Team |   36 |
|         1 | Babar Azam          | West Indians Team  |   32 |
|         2 | Craig McMillan      | New Zealand Team   |   40 |
|         3 | Temba Bavuma        | Kenya Team         |   32 |
|         4 | Shaheen Afridi      | Bangladesh Team    |   22 |
|         4 | Gareth Delany       | Netherlands Team   |   25 |
|         4 | Dhananjaya de Silva | Sri Lankan Team    |   30 |
```

This query produces 36 rows where it gives player details along with their calculated age.

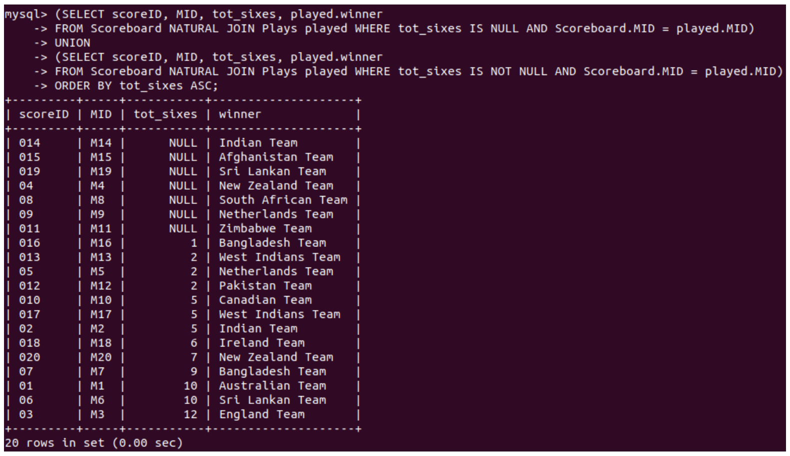
10. Show the batting strike rate of every batsman player in Zimbabwe, Sri Lankan, Afghanistan and West Indians teams and round it off to a whole number, display the player details as well with player number and their relative team's name.



```
mysql> SELECT playerNum, teamName, ROUND(batting_strike_rate, 0) AS batting_strike_rate

    -> FROM Batsmen
    -> WHERE teamName IN('Zimbabwe Team','Sri Lankan Team','England Team','Afghanistan
Team','West Indians Team');
+-----------+-------------------+---------------------+
| playerNum | teamName          | batting_strike_rate |
+-----------+-------------------+---------------------+
|         1 | Sri Lankan Team   |                 472 |
|         1 | West Indians Team |                 369 |
|         8 | England Team      |                 678 |
|        10 | Sri Lankan Team   |                 239 |
|        14 | Zimbabwe Team     |                 230 |
|        15 | Afghanistan Team  |                 236 |
+-----------+-------------------+---------------------+
6 rows in set (0.00 sec)
```

The batting strike rate of each player is original given to 2dp, but this query gives the batting strike rate of each player in the 4 teams to a whole number. This query results can be used in further calculation involving strike rate such as yearly strike rate from all the matches played.

11.   Get the scoreboard details where the winner of the match scored NULL sixes and where the winner scored sixes in the match and order the total sixes obtained by sorting in ascending order.



```
mysql> (SELECT scoreID, MID, tot_sixes, played.winner
    -> FROM Scoreboard NATURAL JOIN Plays played WHERE tot_sixes IS NULL AND Scoreboard.MID = played.MID)
    -> UNION
    -> (SELECT scoreID, MID, tot_sixes, played.winner
    -> FROM Scoreboard NATURAL JOIN Plays played WHERE tot_sixes IS NOT NULL AND Scoreboard.MID = played.MID)
    -> ORDER BY tot_sixes ASC;
+---------+-----+-----------+--------------------+
| scoreID | MID | tot_sixes | winner             |
+---------+-----+-----------+--------------------+
| 014     | M14 |      NULL | Indian Team        |
| 015     | M15 |      NULL | Afghanistan Team   |
| 019     | M19 |      NULL | Sri Lankan Team    |
| 04      | M4  |      NULL | New Zealand Team   |
| 08      | M8  |      NULL | South African Team |
| 09      | M9  |      NULL | Netherlands Team   |
| 011     | M11 |      NULL | Zimbabwe Team      |
| 016     | M16 |         1 | Bangladesh Team    |
| 013     | M13 |         2 | West Indians Team  |
| 05      | M5  |         2 | Netherlands Team   |
| 012     | M12 |         2 | Pakistan Team      |
| 010     | M10 |         5 | Canadian Team      |
| 017     | M17 |         5 | West Indians Team  |
| 02      | M2  |         5 | Indian Team        |
| 018     | M18 |         6 | Ireland Team       |
| 020     | M20 |         7 | New Zealand Team   |
| 07      | M7  |         9 | Bangladesh Team    |
| 01      | M1  |        10 | Australian Team    |
| 06      | M6  |        10 | Sri Lankan Team    |
| 03      | M3  |        12 | England Team       |
+---------+-----+-----------+--------------------+
20 rows in set (0.00 sec)
```

From the first query, I have obtained the matchID, scored and winning team where total sixes is null value. The second query outputs the same column results where total sixes are not null. These two queries are joined by using a union. Since its union, it adheres to the two rules where both the sub queries have the same number of columns and same datatypes. This query can be used to determine total sixes achieved by each winning team.

There are 14 more queries written in Queries.sql file where I have used different techniques for queries to obtain variety of results.

4.2. Design and implementation of advanced features

A sufficient number of advanced features are implemented to acquire results in more advanced manner for specific analytic purposes. The 4 main advanced features are written in three different sql files.

NOTE: Before each advanced query such as stored procedures, views, indexes, or triggers, first I have written a query to drop/ delete the query if it already exists to avoid errors. An example is shown below:

```
DROP PROCEDURE IF EXISTS get_schedules; -- first drop the procedure if it exists
```

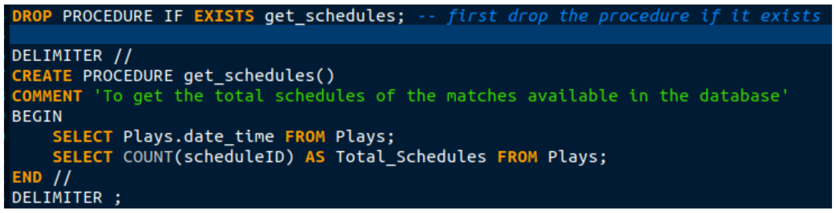
4.2.1. Stored Procedures

A stored routine (procedure or function) is a set of SQL statements that can be stored in the server.

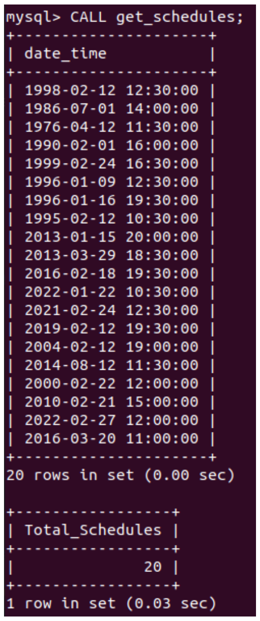
So now clients can refer to the stored procedure by calling it instead of affecting individually. Stored procedures can be especially useful when multiple client applications written in different languages or running on different platforms require the same database operations.

There are three stored procedures implemented that are used to do the necessary function so that clients can easily manage the outputs obtained.

1. A stored procedure to get all the schedules of the matches played where the user can get the schedule details with the data time and the schedule ID by calling the procedure. It also gives the number of scheduled matches as shown on the right. This query can be used for the client to easily get the schedule details without having to write select queries

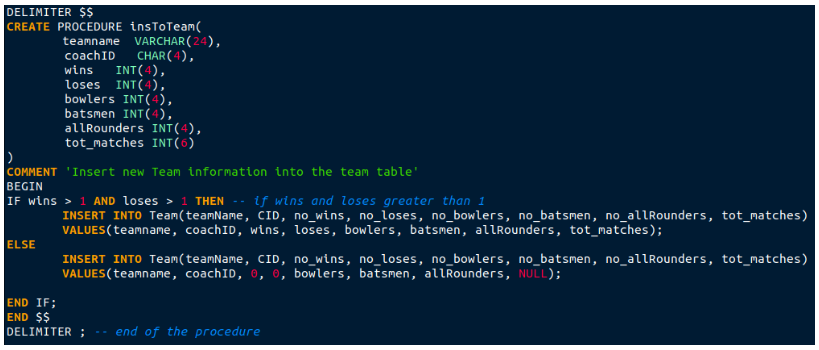


```
DROP PROCEDURE IF EXISTS get_schedules; -- first drop the procedure if it exists

DELIMITER //
CREATE PROCEDURE get_schedules()
COMMENT 'To get the total schedules of the matches available in the database'
BEGIN
    SELECT Plays.date_time FROM Plays;
    SELECT COUNT(scheduleID) AS Total_Schedules FROM Plays;
END //
DELIMITER ;
```



```
mysql> CALL get_schedules;
+---------------------+
| date_time           |
+---------------------+
| 1998-02-12 12:30:00 |
| 1986-07-01 14:00:00 |
| 1976-04-12 11:30:00 |
| 1990-02-01 16:00:00 |
| 1999-02-24 16:30:00 |
| 1996-01-09 12:30:00 |
| 1996-01-16 19:30:00 |
| 1995-02-12 10:30:00 |
| 2013-01-15 20:00:00 |
| 2013-03-29 18:30:00 |
| 2016-02-18 19:30:00 |
| 2022-01-22 10:30:00 |
| 2021-02-24 12:30:00 |
| 2019-02-12 19:30:00 |
| 2004-02-12 19:00:00 |
| 2014-08-12 11:30:00 |
| 2000-02-22 12:00:00 |
| 2010-02-21 15:00:00 |
| 2022-02-27 12:00:00 |
| 2016-03-20 11:00:00 |
+---------------------+
20 rows in set (0.00 sec)

+-----------------+
| Total_Schedules |
+-----------------+
|              20 |
+-----------------+
1 row in set (0.03 sec)
```

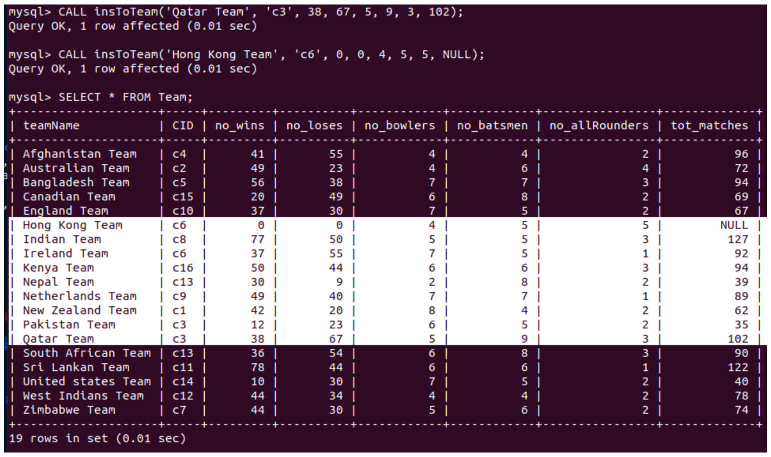
2. To insert to the team table where the user can insert to the table if the no of wins and no of loses is greater than 1 then enter the values with else make tot_score null.



```
DELIMITER $$
CREATE PROCEDURE insToTeam(
        teamname  VARCHAR(24),
        coachID   CHAR(4),
        wins   INT(4),
        loses  INT(4),
        bowlers INT(4),
        batsmen INT(4),
        allRounders INT(4),
        tot_matches INT(6)
)
COMMENT 'Insert new Team information into the team table'
BEGIN
IF wins > 1 AND loses > 1 THEN -- if wins and loses greater than 1
        INSERT INTO Team(teamName, CID, no_wins, no_loses, no_bowlers, no_batsmen, no_allRounders, tot_matches)
        VALUES(teamname, coachID, wins, loses, bowlers, batsmen, allRounders, tot_matches);
ELSE
        INSERT INTO Team(teamName, CID, no_wins, no_loses, no_bowlers, no_batsmen, no_allRounders, tot_matches)
        VALUES(teamname, coachID, 0, 0, bowlers, batsmen, allRounders, NULL);

END IF;
END $$
DELIMITER ; -- end of the procedure
```

The query is used so a client can insert to the team table easily without having to write insert queries hence this saves time. If the number of wins and losses are less than 1 then make tot_matches NULL automatically.

The output of the query can be achieved by calling the procedure and inserting to the team table in both scenarios as shown below.

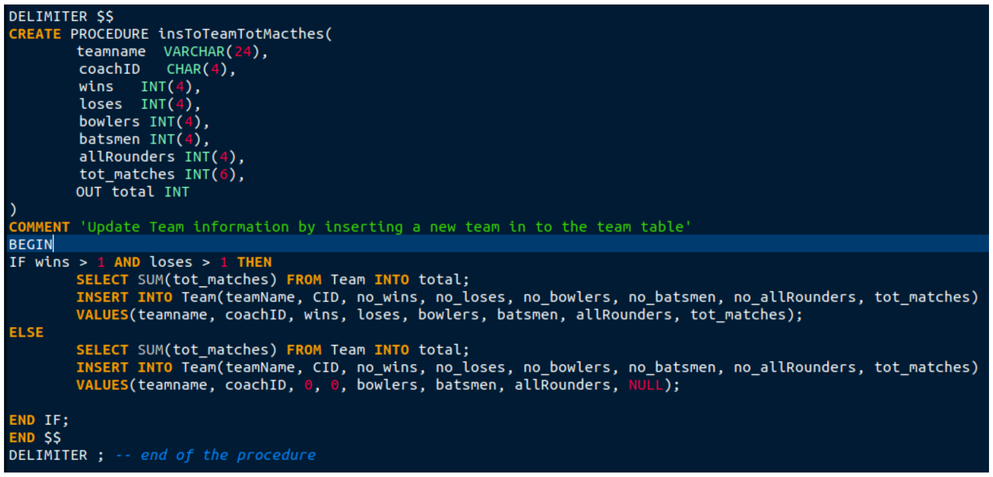


```
mysql> CALL insToTeam('Qatar Team', 'c3', 38, 67, 5, 9, 3, 102);
Query OK, 1 row affected (0.01 sec)

mysql> CALL insToTeam('Hong Kong Team', 'c6', 0, 0, 4, 5, 5, NULL);
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM Team;
+--------------------+-----+---------+----------+------------+------------+----------------+-------------+
| teamName           | CID | no_wins | no_loses | no_bowlers | no_batsmen | no_allRounders | tot_matches |
+--------------------+-----+---------+----------+------------+------------+----------------+-------------+
| Afghanistan Team   | c4  |      41 |       55 |          4 |          4 |              2 |          96 |
| Australian Team    | c2  |      49 |       23 |          4 |          6 |              4 |          72 |
| Bangladesh Team    | c5  |      56 |       38 |          7 |          7 |              3 |          94 |
| Canadian Team      | c15 |      20 |       49 |          6 |          8 |              2 |          69 |
| England Team       | c10 |      37 |       30 |          7 |          5 |              2 |          67 |
| Hong Kong Team     | c6  |       0 |        0 |          4 |          5 |              5 |        NULL |
| Indian Team        | c8  |      77 |       50 |          5 |          5 |              3 |         127 |
| Ireland Team       | c6  |      37 |       55 |          7 |          5 |              1 |          92 |
| Kenya Team         | c16 |      50 |       44 |          6 |          6 |              3 |          94 |
| Nepal Team         | c13 |      30 |        9 |          2 |          8 |              2 |          39 |
| Netherlands Team   | c9  |      49 |       40 |          7 |          7 |              1 |          89 |
| New Zealand Team   | c1  |      42 |       20 |          8 |          4 |              2 |          62 |
| Pakistan Team      | c3  |      12 |       23 |          6 |          5 |              2 |          35 |
| Qatar Team         | c3  |      38 |       67 |          5 |          9 |              3 |         102 |
| South African Team | c13 |      36 |       54 |          6 |          8 |              3 |          90 |
| Sri Lankan Team    | c11 |      78 |       44 |          6 |          6 |              1 |         122 |
| United states Team | c14 |      10 |       30 |          7 |          5 |              2 |          40 |
| West Indians Team  | c12 |      44 |       34 |          4 |          4 |              2 |          78 |
| Zimbabwe Team      | c7  |      44 |       30 |          5 |          6 |              2 |          74 |
+--------------------+-----+---------+----------+------------+------------+----------------+-------------+
19 rows in set (0.01 sec)
```

The first call procedure inserts a new team where number of wins and loses are greater than 1 where the second call procedure inserts a team where the number of wins and loses are less than 1 hence its inserted as NULL.

Where the highlighted area starts we can see procedure 2 entered values and end of the highlighted area we can see the procedure 1 values correctly entered.

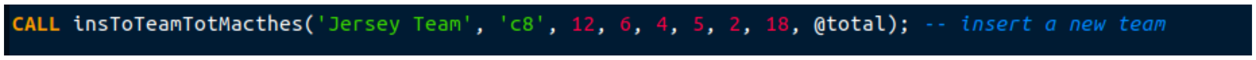
3. Calculates the total matches played by every match in the database and takes it out as an out parameter when a user tries to insert to the Team table.



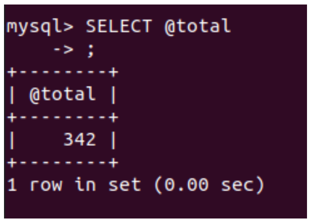
```
DELIMITER $$
CREATE PROCEDURE insToTeamTotMacthes(
         teamname  VARCHAR(24),
         coachID   CHAR(4),
         wins   INT(4),
         loses  INT(4),
         bowlers INT(4),
         batsmen INT(4),
         allRounders INT(4),
         tot_matches INT(6),
         OUT total INT
)
COMMENT 'Update Team information by inserting a new team in to the team table'
BEGIN
IF wins > 1 AND loses > 1 THEN
         SELECT SUM(tot_matches) FROM Team INTO total;
         INSERT INTO Team(teamName, CID, no_wins, no_loses, no_bowlers, no_batsmen, no_allRounders, tot_matches)
         VALUES(teamname, coachID, wins, loses, bowlers, batsmen, allRounders, tot_matches);
ELSE
         SELECT SUM(tot_matches) FROM Team INTO total;
         INSERT INTO Team(teamName, CID, no_wins, no_loses, no_bowlers, no_batsmen, no_allRounders, tot_matches)
         VALUES(teamname, coachID, 0, 0, bowlers, batsmen, allRounders, NULL);

END IF;
END $$
DELIMITER ; -- end of the procedure
```

it's the same as the earlier query but has an out parameter where the sum of the total matches can be retrieved outside the procedure as its OUT parameter.

The below query is used to call the procedure and get the sum of the matches outside by using the @total parameter.



```
CALL insToTeamTotMacthes('Jersey Team', 'c8', 12, 6, 4, 5, 2, 18, @total); -- insert a new team
```

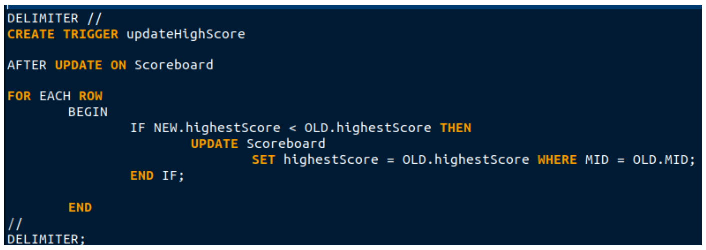


```
mysql> SELECT @total
    -> ;
+--------+
| @total |
+--------+
|    342 |
+--------+
1 row in set (0.00 sec)
```

The returned result will be stored in the variable @total afterwards and can be displayed with a SELECT statement as shown on the left.

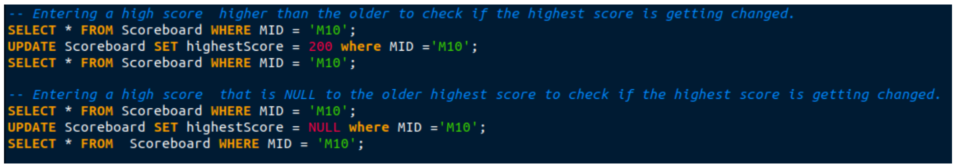
4.2.2. Triggers

A trigger is defined to activate when a statement inserts, updates, or deletes rows in the associated table. It triggers certain events before each row that is inserted into a table or after each row that is updated. There are two triggers written which will be activated if a user tries to update or insert to the existing tables:



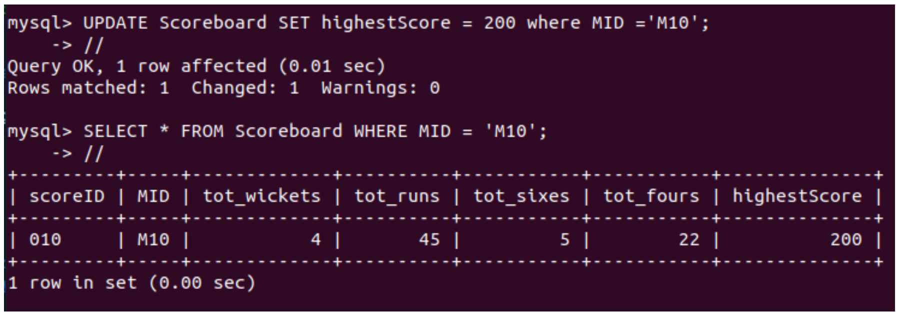
```
DELIMITER //
CREATE TRIGGER updateHighScore

AFTER UPDATE ON Scoreboard

FOR EACH ROW
        BEGIN
                IF NEW.highestScore < OLD.highestScore THEN
                        UPDATE Scoreboard
                                SET highestScore = OLD.highestScore WHERE MID = OLD.MID;
                END IF;

        END
//
DELIMITER;
```

1. If the user tries to update the existing highest score in the scoreboard and if it is less than the existing highest score create a trigger to keep the existing highest score on the table.



```
-- Entering a high score  higher than the older to check if the highest score is getting changed.
SELECT * FROM Scoreboard WHERE MID = 'M10';
UPDATE Scoreboard SET highestScore = 200 where MID ='M10';
SELECT * FROM Scoreboard WHERE MID = 'M10';

-- Entering a high score  that is NULL to the older highest score to check if the highest score is getting changed.
SELECT * FROM Scoreboard WHERE MID = 'M10';
UPDATE Scoreboard SET highestScore = NULL where MID ='M10';
SELECT * FROM  Scoreboard WHERE MID = 'M10';
```

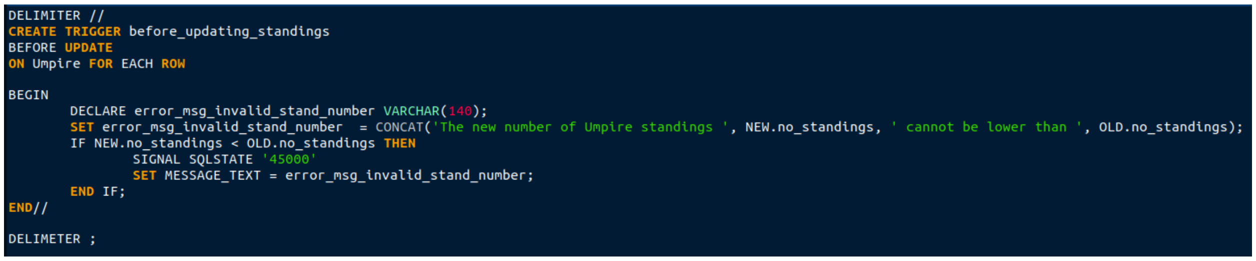
The trigger is activated after update on scoreboard table for each row. This trigger is called when the user wants to update an existing highest score and if it less than the existing one. To test this trigger, we could update an existing score in the scoreboard table by first entering a higher score and then making it NULL.



```
mysql> UPDATE Scoreboard SET highestScore = 200 where MID ='M10';
    -> //
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM Scoreboard WHERE MID = 'M10';
    -> //
+---------+-----+-------------+----------+-----------+-----------+--------------+
| scoreID | MID | tot_wickets | tot_runs | tot_sixes | tot_fours | highestScore |
+---------+-----+-------------+----------+-----------+-----------+--------------+
| 010     | M10 |           4 |       45 |         5 |        22 |          200 |
+---------+-----+-------------+----------+-----------+-----------+--------------+
1 row in set (0.00 sec)
```

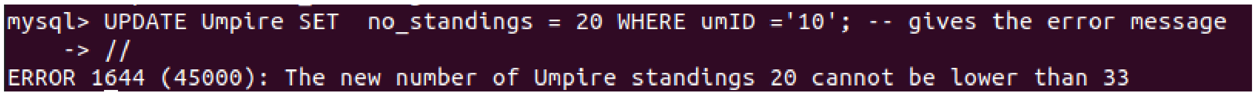
The output shows that the trigger entered is activated when a user tries to update the highest score in the scoreboard table.

2. If the user tries to update the number of umpire standings in the umpire table of a particular umpire and the new number that the user tries to insert is lower than the old number of umpire standings create a trigger to give the user a warning saying that the new number of umpire standings should be greater than the old number of umpire standings.



```
DELIMITER //
CREATE TRIGGER before_updating_standings
BEFORE UPDATE
ON Umpire FOR EACH ROW

BEGIN
        DECLARE error_msg_invalid_stand_number VARCHAR(140);
        SET error_msg_invalid_stand_number  = CONCAT('The new number of Umpire standings ', NEW.no_standings, ' cannot be lower than ', OLD.no_standings);
        IF NEW.no_standings < OLD.no_standings THEN
                SIGNAL SQLSTATE '45000'
                SET MESSAGE_TEXT = error_msg_invalid_stand_number;
        END IF;
END//

DELIMETER ;
```

The trigger entered displays an error message if the user tries to update an existing no_standings in the umpire table to a lower value than the older value else it is entered without any errors as shown below.

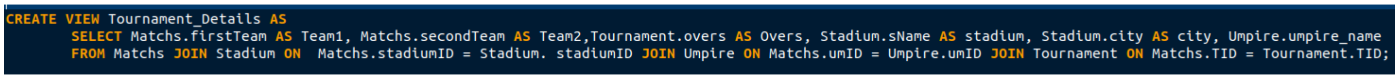


```
mysql> UPDATE Umpire SET  no_standings = 20 WHERE umID ='10'; -- gives the error message
    -> //
ERROR 1644 (45000): The new number of Umpire standings 20 cannot be lower than 33
```

4.2.3. Views

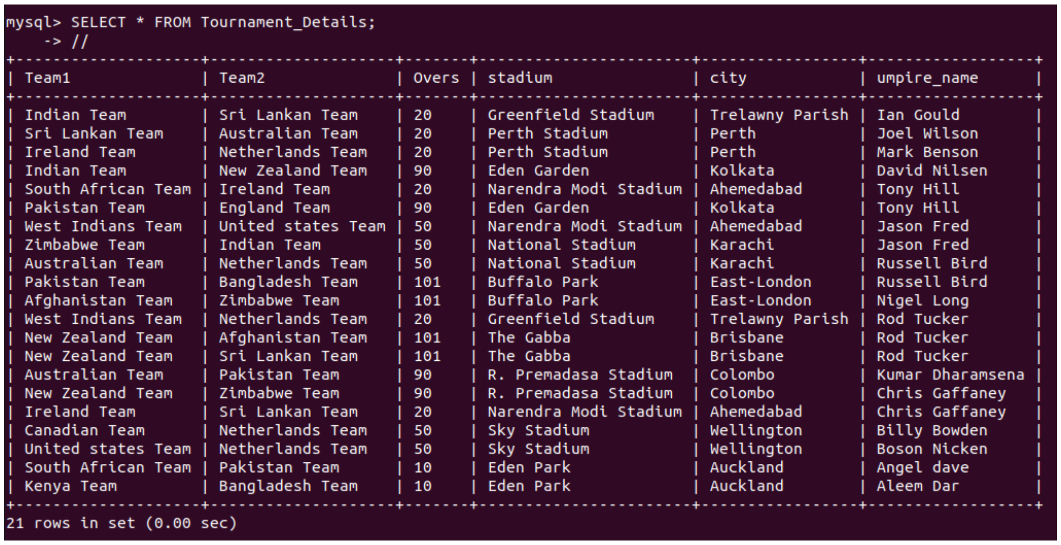
A view is a table which is derived from other stored tables which are called base tables or defining tables. There are two views written to get detailed information by combining existing tables.

1. Create a detailed view containing all the information regarding a tournament, getting match, stadium, and umpire details.



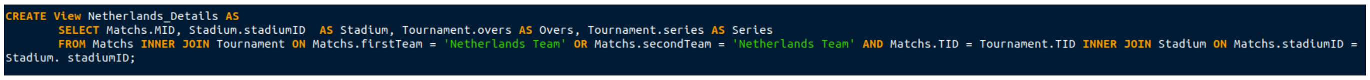
```sql
CREATE VIEW Tournament_Details AS
        SELECT Matchs.firstTeam AS Team1, Matchs.secondTeam AS Team2,Tournament.overs AS Overs, Stadium.sName AS stadium, Stadium.city AS city, Umpire.umpire_name
        FROM Matchs JOIN Stadium ON  Matchs.stadiumID = Stadium. stadiumID JOIN Umpire ON Matchs.umID = Umpire.umID JOIN Tournament ON Matchs.TID = Tournament.TID;
```

By using a select query we can check the view for the tournament view created and the data will be fetched from the existing tables and produce the output as follows:



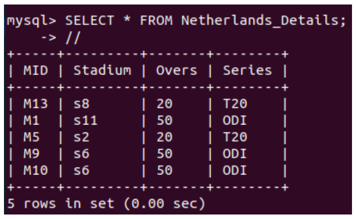
```
mysql> SELECT * FROM Tournament_Details;
    -> //
+--------------------+--------------------+-------+-----------------------+-----------------+------------------+
| Team1              | Team2              | Overs | stadium               | city            | umpire_name      |
+--------------------+--------------------+-------+-----------------------+-----------------+------------------+
| Indian Team        | Sri Lankan Team    | 20    | Greenfield Stadium    | Trelawny Parish | Ian Gould        |
| Sri Lankan Team    | Australian Team    | 20    | Perth Stadium         | Perth           | Joel Wilson      |
| Ireland Team       | Netherlands Team   | 20    | Perth Stadium         | Perth           | Mark Benson      |
| Indian Team        | New Zealand Team   | 90    | Eden Garden           | Kolkata         | David Nilsen     |
| South African Team | Ireland Team       | 20    | Narendra Modi Stadium | Ahemedabad      | Tony Hill        |
| Pakistan Team      | England Team       | 90    | Eden Garden           | Kolkata         | Tony Hill        |
| West Indians Team  | United states Team | 50    | Narendra Modi Stadium | Ahemedabad      | Jason Fred       |
| Zimbabwe Team      | Indian Team        | 50    | National Stadium      | Karachi         | Jason Fred       |
| Australian Team    | Netherlands Team   | 50    | National Stadium      | Karachi         | Russell Bird     |
| Pakistan Team      | Bangladesh Team    | 101   | Buffalo Park          | East-London     | Russell Bird     |
| Afghanistan Team   | Zimbabwe Team      | 101   | Buffalo Park          | East-London     | Nigel Long       |
| West Indians Team  | Netherlands Team   | 20    | Greenfield Stadium    | Trelawny Parish | Rod Tucker       |
| New Zealand Team   | Afghanistan Team   | 101   | The Gabba             | Brisbane        | Rod Tucker       |
| New Zealand Team   | Sri Lankan Team    | 101   | The Gabba             | Brisbane        | Rod Tucker       |
| Australian Team    | Pakistan Team      | 90    | R. Premadasa Stadium  | Colombo         | Kumar Dharamsena |
| New Zealand Team   | Zimbabwe Team      | 90    | R. Premadasa Stadium  | Colombo         | Chris Gaffaney   |
| Ireland Team       | Sri Lankan Team    | 20    | Narendra Modi Stadium | Ahemedabad      | Chris Gaffaney   |
| Canadian Team      | Netherlands Team   | 50    | Sky Stadium           | Wellington      | Billy Bowden     |
| United states Team | Netherlands Team   | 50    | Sky Stadium           | Wellington      | Boson Nicken     |
| South African Team | Pakistan Team      | 10    | Eden Park             | Auckland        | Angel dave       |
| Kenya Team         | Bangladesh Team    | 10    | Eden Park             | Auckland        | Aleem Dar        |
+--------------------+--------------------+-------+-----------------------+-----------------+------------------+
21 rows in set (0.00 sec)
```

2. Create a view to get all Match details where Netherlands team is Playing by fetching the stadium and tournament details from the relevant tables.



```sql
CREATE View Netherlands_Details AS
        SELECT Matchs.MID, Stadium.stadiumID  AS Stadium, Tournament.overs AS Overs, Tournament.series AS Series
        FROM Matchs INNER JOIN Tournament ON Matchs.firstTeam = 'Netherlands Team' OR Matchs.secondTeam = 'Netherlands Team' AND Matchs.TID = Tournament.TID INNER JOIN Stadium ON Matchs.stadiumID =
Stadium. stadiumID;
```

The output of this view can be obtained as follows where the view shows the match details, stadium, and tournament details:



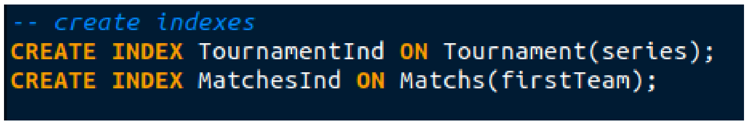
```
mysql> SELECT * FROM Netherlands_Details;
    -> //
+-----+---------+-------+--------+
| MID | Stadium | Overs | Series |
+-----+---------+-------+--------+
| M13 | s8      | 20    | T20    |
| M1  | s11     | 50    | ODI    |
| M5  | s2      | 20    | T20    |
| M9  | s6      | 50    | ODI    |
| M10 | s6      | 50    | ODI    |
+-----+---------+-------+--------+
5 rows in set (0.00 sec)
```

Shows the output from the view where the user can see what the associated details to Netherlands team are.

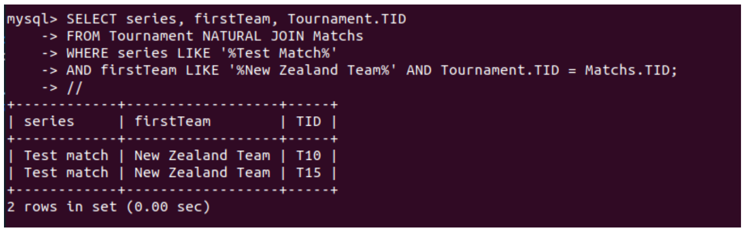
4.2.4. Indexes

An index is a data structure that is used to speed up access to tuples in a relation based on the values of one or more attributes. Indexes are used to quickly find rows with specific column values. There are two indexes written to speed up the process of retrieving information:

1. Make two indexes called TournamentInd and MacthesInd to find the series and all the matches played in a tournament that is played in a particular series where first participating team includes specific team name.



```
-- create indexes
CREATE INDEX TournamentInd ON Tournament(series);
CREATE INDEX MatchesInd ON Matchs(firstTeam);
```

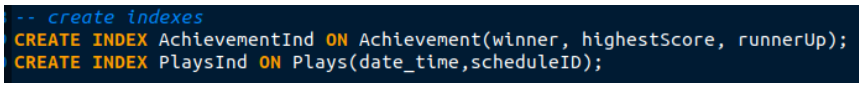


```
mysql> SELECT series, firstTeam, Tournament.TID
    -> FROM Tournament NATURAL JOIN Matchs
    -> WHERE series LIKE '%Test Match%'
    -> AND firstTeam LIKE '%New Zealand Team%' AND Tournament.TID = Matchs.TID;
    -> //
+------------+------------------+-----+
| series     | firstTeam        | TID |
+------------+------------------+-----+
| Test match | New Zealand Team | T10 |
| Test match | New Zealand Team | T15 |
+------------+------------------+-----+
2 rows in set (0.00 sec)
```

These two indexes can be used to retrieve tournament details where series equals to ‘test match’ and first team equals to ‘New Zealand Team’.

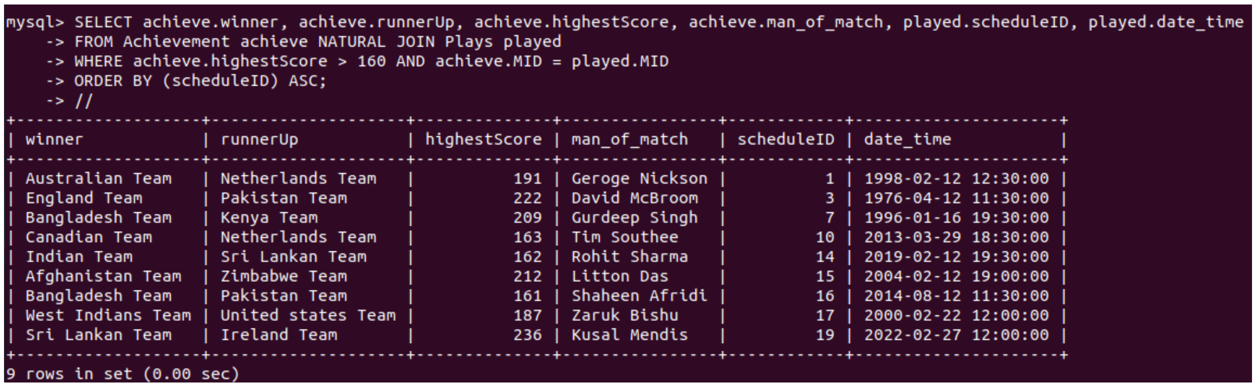
The query shows the results obtained using the two indexes created.

2. Use two indexes called AchievementInd and PlaysInd to find the winner, runner up, highestscore in a match, the date, time, and the schedule ID of that match where the highestScore is greater than 160. These two indexes can be used to speed up the process of finding the winner and its schedule details.



```
-- create indexes
CREATE INDEX AchievementInd ON Achievement(winner, highestScore, runnerUp);
CREATE INDEX PlaysInd ON Plays(date_time,scheduleID);
```

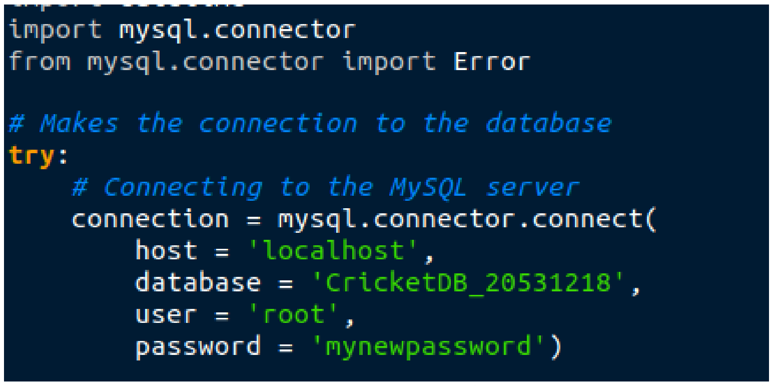
To test this index, we can use a select query to retrieve winner details in a match played where the high score is greater than 160 as shown below.



```
mysql> SELECT achieve.winner, achieve.runnerUp, achieve.highestScore, achieve.man_of_match, played.scheduleID, played.date_time
    -> FROM Achievement achieve NATURAL JOIN Plays played
    -> WHERE achieve.highestScore > 160 AND achieve.MID = played.MID
    -> ORDER BY (scheduleID) ASC;
    -> //
+-------------------+--------------------+--------------+----------------+------------+---------------------+
| winner            | runnerUp           | highestScore | man_of_match   | scheduleID | date_time           |
+-------------------+--------------------+--------------+----------------+------------+---------------------+
| Australian Team   | Netherlands Team   |          191 | Geroge Nickson |          1 | 1998-02-12 12:30:00 |
| England Team      | Pakistan Team      |          222 | David McBroom  |          3 | 1976-04-12 11:30:00 |
| Bangladesh Team   | Kenya Team         |          209 | Gurdeep Singh  |          7 | 1996-01-16 19:30:00 |
| Canadian Team     | Netherlands Team   |          163 | Tim Southee    |         10 | 2013-03-29 18:30:00 |
| Indian Team       | Sri Lankan Team    |          162 | Rohit Sharma   |         14 | 2019-02-12 19:30:00 |
| Afghanistan Team  | Zimbabwe Team      |          212 | Litton Das     |         15 | 2004-02-12 19:00:00 |
| Bangladesh Team   | Pakistan Team      |          161 | Shaheen Afridi |         16 | 2014-08-12 11:30:00 |
| West Indians Team | United states Team |          187 | Zaruk Bishu    |         17 | 2000-02-22 12:00:00 |
| Sri Lankan Team   | Ireland Team       |          236 | Kusal Mendis   |         19 | 2022-02-27 12:00:00 |
+-------------------+--------------------+--------------+----------------+------------+---------------------+
9 rows in set (0.00 sec)
```

4.3. Database Connectivity and Python Implementation

The database connectivity is done using python3 where I first, installed mysql-connector-python and then made the connection to the CricketDB_20531218 database in the program.py source code.

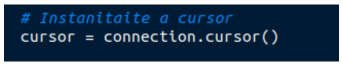


```python
import mysql.connector
from mysql.connector import Error

# Makes the connection to the database
try:
    # Connecting to the MySQL server
    connection = mysql.connector.connect(
        host = 'localhost',
        database = 'CricketDB_20531218',
        user = 'root',
        password = 'mynewpassword')
```

By importing mysql.connector we can connect the database with python3.

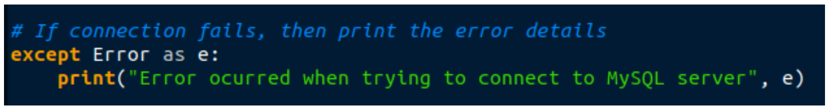
To make the connection, I have hard coded the host as localhost, database name, username as root and my password as shown on the left.

On successful, the cursor is instantiated as shown →



```python
# Instanitaite a cursor
cursor = connection.cursor()
```

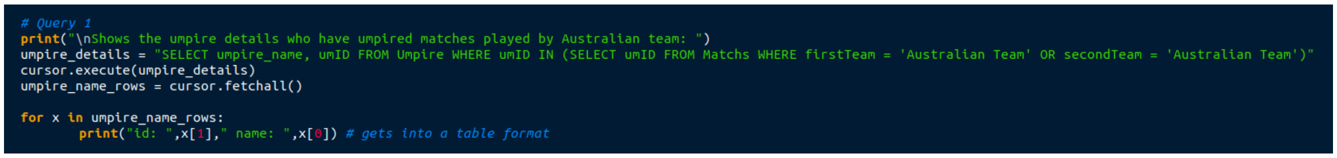
The cursor is used to retrieve data row by row. On error, it results in an error message as shown below.



```python
# If connection fails, then print the error details
except Error as e:
    print("Error ocurred when trying to connect to MySQL server", e)
```

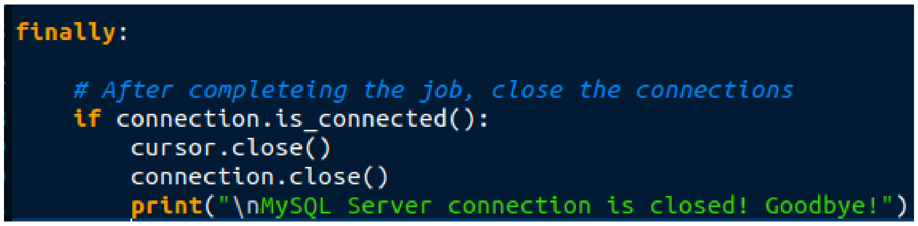
It prints the error message and returns to the terminal, so the users have to run the file again.

After the connection is made securely, the queries can be written to fetch the results and print them to the screen using a for loop as shown below:



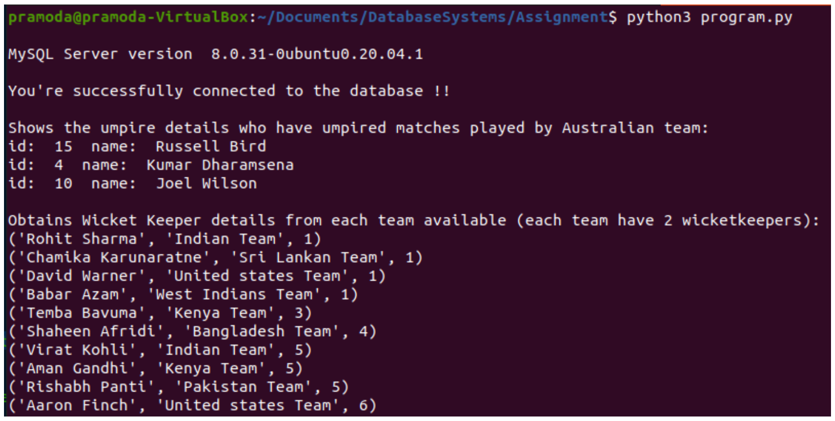
```python
# Query 1
print("\nShows the umpire details who have umpired matches played by Australian team: ")
umpire_details = "SELECT umpire_name, umID FROM Umpire WHERE umID IN (SELECT umID FROM Matchs WHERE firstTeam = 'Australian Team' OR secondTeam = 'Australian Team')"
cursor.execute(umpire_details)
umpire_name_rows = cursor.fetchall()

for x in umpire_name_rows:
       print("id: ",x[1]," name: ",x[0]) # gets into a table format
```

First, we execute the query with cursor.execute() and fetch all the rows from the database and print using a for loop. There are 14 more queries that are written in this pattern to obtain a variety of results. When the retrieval of query results is over, we can close the cursor and disconnect the connection using an if as shown below. It will automatically starts executing the program where the query results will be printed to the screen and disconnect after displaying is done.



```python
finally:

    # After completeing the job, close the connections
    if connection.is_connected():
        cursor.close()
        connection.close()
        print("\nMySQL Server connection is closed! Goodbye!")
```

The results after running the program.py file can be obtained as follows:



```
pramoda@pramoda-VirtualBox:~/Documents/DatabaseSystems/Assignment$ python3 program.py

MySQL Server version  8.0.31-0ubuntu0.20.04.1

You're successfully connected to the database !!

Shows the umpire details who have umpired matches played by Australian team:
id:  15  name:  Russell Bird
id:  4  name:  Kumar Dharamsena
id:  10  name:  Joel Wilson

Obtains Wicket Keeper details from each team available (each team have 2 wicketkeepers):
('Rohit Sharma', 'Indian Team', 1)
('Chamika Karunaratne', 'Sri Lankan Team', 1)
('David Warner', 'United states Team', 1)
('Babar Azam', 'West Indians Team', 1)
('Temba Bavuma', 'Kenya Team', 3)
('Shaheen Afridi', 'Bangladesh Team', 4)
('Virat Kohli', 'Indian Team', 5)
('Aman Gandhi', 'Kenya Team', 5)
('Rishabh Panti', 'Pakistan Team', 5)
('Aaron Finch', 'United states Team', 6)
```

First run the python program using the command python3 then, we can see that it first displays the MySQL version and then continues to output the query results.

5. Discussion

This Cricket Database management system is used to achieve past and current information about a cricket event. The ER diagram is in 1st normal form due to multivalued attribute present, this is refined in the relational schema by performing decomposition to get the logical design to normalization form. The implementation of the database was done by sourcing sql files to the database created. After insertion, different types of queries were written to obtain a wide range of results from the tables by using joins, sub queries, string and data time manipulation, and aggregation using group by and order by clauses. There were at most 2 advanced features covered in each category of stored procedures, triggers, views, and indexes. Significantly, the connection of the database to python3 was done accurately as mentioned in section 4.3.

When trying to insert to the tables with the referential integrity constraints it was time consuming and had to be entered with care as a lot of the tables referenced to each other in foreign keys. During the implementation of the queries, many of the insertions to the tables were modified and it was very time consuming to obtain results to some queries as my database is complex. The triggers and views designed were not mandatory but for easiness I have created simple triggers and views.

To improve and reflect on the implemented database we can use the following mentioned points. The captain attribute in the Player table can be taken as a separate entity rather than a role of Player to give more precision. In Matchs table, we can make firstTeam and secondTeam attributes as primary key with MID therefore Matchs table have a composite key so when data is queried, we can obtain more understandable data. We can implement more triggers where a trigger event is activated when the user tries to update a table after batsmen details are added to the table, where a trigger event can be activated after a user updates the RankIn table. Few views can be implemented to obtain more specific results from the tables using joins and subqueries. The limitations could be that only one user can employee the database at a time as there is no multithreading involved to obtain query results from the python3 program. The cricket database was complete where any user who wants to access it can do so by following the user guide to get the initial start-up.

In summary, every aspect of the assignment's scope is covered to the fullest and since, my project is a complex implementation of the cricket database, the report extended for more than 20 pages. However, it is assured that this Cricket database can be used for advanced analytic purposes by management or administrators and to produce various forms of cricket books, magazines, and souvenirs. As lessons learned, the progress could have been continuous as it was time consuming where new innovative ideas for entities and attributes were added during some of the stages. So, create table queries and insertion queries had to be refined a few times. Other than that, the database was implemented without any issues. Finally, this cricket database management system can be used for future in reference to the sport in gathering information or for comparison with current tournaments for analytical purposes.

6. References

Times of SportsA Leading Sports News Channel For All The Exclusive Sports Updates Across Globe., Sports, T.of and A Leading Sports News Channel For All The Exclusive Sports Updates Across Globe. (2022) *T20 World Cup 2022 new rules: Ban of use of saliva, Mankading becomes legal*, *Times of Sports News Today, Latest Headlines Updates, Live Match Score*. Available at https://www.timesofsports.com/cricket/t20i-world-cup/2022-rules/ (Accessed: October 20, 2022).

Narayanan, M. (2020) *Cricket Data*, *Kaggle*. Available at: https://www.kaggle.com/datasets/mahendran1/icc-cricket (Accessed: October 15, 2022).

There are 13 cricket datasets available on data.world. (2021) *data.world*. Available at: https://data.world/datasets/cricket (Accessed: October 15, 2022).

List of cricket grounds by capacity (2022) *Wikipedia*. Wikimedia Foundation. Available at: https://en.wikipedia.org/wiki/List_of_cricket_grounds_by_capacity (Accessed: October 25, 2022).