

## CHAPTER 1

# INTRODUCTION

### 1.1. Introduction to Database Management System (DBMS)

A database management system (DBMS) is system software for creating and managing databases. The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage data.

A DBMS makes it possible for end users to create, read, update and delete data in a database. The DBMS essentially serves as an interface between the database and end users or application programs, ensuring that data is consistently organized and remains easily accessible.

The DBMS manages three important things: the data, the database engine that allows data to be accessed, locked and modified -- and the database schema, which defines the database's logical structure. These three foundational elements help provide concurrency, security, data integrity and uniform administration procedures. Typical database administration tasks supported by the DBMS include change management, performance monitoring/tuning and backup and recovery. Many database management systems are also responsible for automated rollbacks, restarts and recovery as well as the logging and auditing of activity.

The DBMS is perhaps most useful for providing a centralized view of data that can be accessed by multiple users, from multiple locations, in a controlled manner. A DBMS can limit what data the end user sees, as well as how that end user can view the data, providing many views of a single database schema. End users and software programs are free from having to understand where the data is physically located or on what type of storage media it resides because the DBMS handles all requests.

The DBMS can offer both logical and physical data independence. That means it can protect users and applications from needing to know where data is stored or having to be concerned about changes to the physical structure of data (storage and hardware).

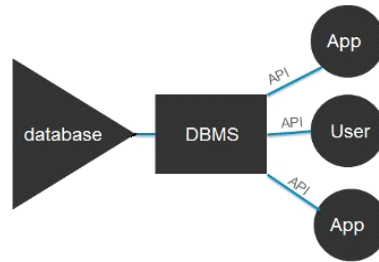


Fig. 1.1: SQL as API

## 1.2. DBMS -Architecture

The design of a DBMS depends on its architecture. It can be centralized or decentralized or hierarchical. The architecture of a DBMS can be seen as either single tier or multi-tier. An n-tier architecture divides the whole system into related but independent **n** modules, which can be independently modified, altered, changed, or replaced.

In 1-tier architecture, the DBMS is the only entity where the user directly sits on the DBMS and uses it. Any changes done here will directly be done on the DBMS itself. It does not provide handy tools for end-users. Database designers and programmers normally prefer to use single-tier architecture.

If the architecture of DBMS is 2-tier, then it must have an application through which the DBMS can be accessed. Programmers use 2-tier architecture where they access the DBMS by means of an application. Here the application tier is entirely independent of the database in terms of operation, design, and programming.

### 1.2.1. 3-tierArchitecture

3-tier architecture separates its tiers from each other based on the complexity of the users and how they use the data present in the database. It is the most widely used architecture to design a DBMS.

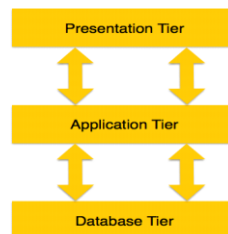


Fig. 1.2: DBMS 3-tier architecture

- **Database (Data) Tier** — At this tier, the database resides along with its query processing languages. We also have the relations that define the data and their constraints at this level.

- **Application (Middle) Tier** — At this tier reside the application server and the programs that access the database. For a user, this application tier presents an abstracted view of the database. End-users are unaware of any existence of the database beyond the application. At the other end, the database tier is not aware of any other user beyond the application tier. Hence, the application layer sits in the middle and acts as a mediator between the end-user and the database.
- **User (Presentation) Tier** — End-users operate on this tier and they know nothing about any existence of the database beyond this layer. At this layer, multiple views of the database can be provided by the application. All views are generated by applications that reside in the application-tier.

Multiple-tier database architecture is highly modifiable, as almost all its components are independent and can be changed independently. Data models define how the logical structure of a database is modeled. Data Models are fundamental entities to introduce abstraction in a DBMS. Data models define how data is connected to each other and how they are processed and stored inside the system.

The very first data model could be flat data-models, where all the data used are to be kept in the same plane. Earlier data models were not so scientific; hence they were prone to introduce lots of duplication and update anomalies.

### 1.2.2. Table(database)

A table is a collection of related data held in a structured format within a database. It consists of columns, and rows.

In relational databases, and flat file databases, a table is a set of data elements (values) using a model of vertical columns (identifiable by name) and horizontal rows, the cell being the unit where a row and column intersect. A table has a specified number of columns, but can have any number of rows. Each row is identified by one or more values appearing in a particular column subset. The columns subset which uniquely identifies a row is called the primary key.

"Table" is another term for "relation"; although there is the difference in that a table is usually a multi set (bag) of rows where a relation is a set and does not allow duplicates. Besides the actual data rows, tables generally have associated with them some metadata, such as constraints on the table or on the values within particular columns.

The data in a table does not have to be physically stored in the database. Views also function as relational tables, but their data are calculated at query time.

### **1.2.3. Field content**

Every table is broken up into smaller entities called fields. The fields in the BILLING table consists of ID, UNAME, BILL & DATE.

A field is a column in a table that is designed to maintain specific information about every record in the table.

### **1.2.4. Record & Row**

A record is also called as a row of data is each individual entry that exists in a table. For example, there are 4 records in the above BILLING table. Following is a single row of data or record in the BILLING table.

### **1.2.5. Column**

A column is a vertical entity in a table that contains all information associated with a specific field in a table. For example, a column in the BILLING table is PRODUCT\_ID, which represents product description.

### **1.2.6. NULL value**

A NULL value in a table is a value in a field that appears to be blank, which means a field with a NULL value is a field with no value.

It is very important to understand that a NULL value is different than a zero value or a field that contains spaces. A field with a NULL value is the one that has been left blank during a record creation. SQL supports a special value known as NULL which is used to represent the values of attributes that may be unknown or not apply to a tuple. For example, the Apartment number attribute of an address applies only to address that are in apartment buildings and not to other types of residences.

## **1.3. SQL**

SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in a relational database.

SQL is the standard language for Relational Database System. All the Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

Also, they are using different dialects, such as –

- MS SQL Server using T-SQL,
- Oracle using PL/SQL,
- MS Access version of SQL is called JET SQL (native format) etc.

### **1.3.1. Advantages of SQL**

SQL is widely popular because it offers the following advantages –

- Allows users to access data in the relational database management systems.
- Allows users to describe the data.
- Allows users to define the data in a database and manipulate that data.
- Allows to embed within other languages using SQL modules, libraries & pre-compilers.
- Allows users to create and drop databases and tables.
- Allows users to create view, stored procedure, functions in a database.
- Allows users to set permissions on tables, procedures and views.

### **1.3.2. A Brief History of SQL**

- 1970 – Dr. Edgar F. "Ted" Codd of IBM is known as the father of relational databases. He described a relational model for databases.
- 1974 – Structured Query Language appeared.
- 1978 – IBM worked to develop Codd's ideas and released a product named System/R.
- 1986 – IBM developed the first prototype of relational database and standardized by ANSI. The first relational database was released by Relational Software which later came to be known as Oracle.

### **1.3.3. SQL Process**

When you are executing an SQL command for any RDBMS, the system determines the best way to carry out your request and SQL engine figures out how to interpret the task.

There are various components included in this process. These components are –

- Query Dispatcher
- Optimization Engines

- Classic Query Engine
- SQL Query Engine, etc.

A classic query engine handles all the non-SQL queries, but a SQL query engine won't handle logical files.

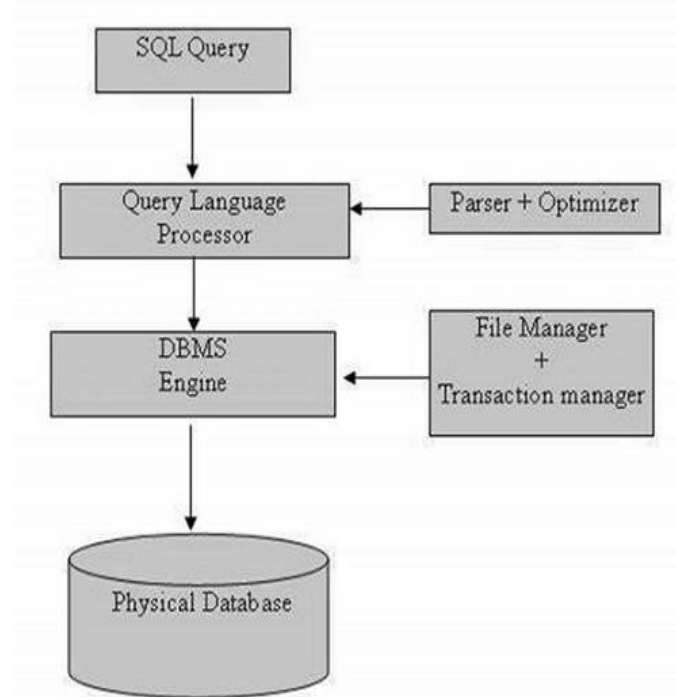


Fig. 1.3: Simple diagram of SQL Architecture

#### 1.3.4. SQL Commands

The standard SQL commands to interact with relational databases are CREATE, SELECT, INSERT, UPDATE, DELETE and DROP. These commands can be classified into the following groups based on their nature—

- **DDL - Data Definition Language**

Sr.No.	Command & Description
1	<b>CREATE</b> Creates a new table, a view of a table, or other object in the database.
2	<b>ALTER</b> Modifies an existing database object, such as a table.
3	<b>DROP</b> Deletes an entire table, a view of a table or other objects in the database.

- **DML - Data Manipulation Language**

Sr.No.	Command & Description
1	<b>SELECT</b>

	Retrieves certain records from one or more tables.
2	<b>INSERT</b> Creates a record.
3	<b>UPDATE</b> Modifies records.
4	<b>DELETE</b> Deletes records.

- **DCL - Data Control Language**

Sr.No.	Command & Description
1	<b>GRANT</b> Gives a privilege to user.
2	<b>REVOKE</b> Takes back privileges granted from user.

## 1.4. Advantages of DBMS

Using a DBMS to store and manage data comes with advantages, but also overhead. One of the biggest advantages of using a DBMS is that it lets end users and application programmers' access and use the same data while managing data integrity. Data is better protected and maintained when it can be shared using a DBMS instead of creating new iterations of the same data stored in new files for every new application. The DBMS provides a central store of data that can be accessed by multiple users in a controlled manner.

Central storage and management of data within the DBMS provides:

- Data abstraction and independence
- Data security
- A locking mechanism for concurrent access
- An efficient handler to balance the needs of multiple applications using the same data
- The ability to swiftly recover from crashes and errors, including restart ability and recoverability
- Robust data integrity capabilities
- Logging and auditing of activity
- Simple access using a standard application programming interface(API)
- Uniform administration procedures for data

Another advantage of a DBMS is that it can be used to impose a logical, structured organization on the data. A DBMS delivers economy of scale for processing large amounts of data because it is optimized for such operations.

A DBMS can also provide many views of a single database schema. A view defines what data the user sees and how that user sees the data. The DBMS provides a level of abstraction between the conceptual schema that defines the logical structure of the database and the physical schema that describes the files, indexes and other physical mechanisms used by the database. When a DBMS is used, systems can be modified much more easily when business requirements change. New categories of data can be added to the database without disrupting the existing system and applications can be insulated from how data is structured and stored.

## **1.5. Applications and Uses of Database Management System**

Due the evolution of Database management system, companies are getting more from their work because they can keep records of everything. Also, it makes them faster to search information and records about any people or product that makes them more effective in work. So here we are sharing some of the applications and uses of database management system (DBMS).

### **1.5.1. Online Shopping**

Online shopping has become a big trend of these days. No one wants to go to shops and waste his time. Everyone wants to shop from home. So, all these products are added and sold only with the help of DBMS. Purchase information, invoice bills and payment, all of these are done with the help of DBMS.

### **1.5.2. Library Management System**

There are thousands of books in the library so it is very difficult to keep record of all the books in a copy or register. So, DBMS used to maintain all the information relate to book issue dates, name of the book, author and availability of the book.

### **1.5.3. Banking**

We make thousands of transactions through banks daily and we can do this without going to the bank. So how banking has become so easy that by sitting at home we can send or get money through banks. That is all possible just because of DBMS that manages all the bank transactions.



### **1.5.4. Universities and colleges**

Examinations are done online today and universities and colleges maintain all these records through DBMS. Student's registrations details, results, courses and grades all the information are stored in database.

### **1.5.5. Credit card transactions**

For purchase of credit cards and all the other transactions are made possible only by DBMS. A credit card holder knows the importance of their information that all are secured through DBMS.

### **1.5.6. Social Media Sites**

We all are on social media websites to share our views and connect with our friends. Daily millions of users signed up for these social media accounts like Facebook, twitter, Pinterest and Google plus. But how all the information of users is stored and how we become able to connect to other people, yes this all because DBMS.

### **1.5.7. Telecommunications**

Any telecommunication company cannot even think about their business without DBMS. DBMS is must for these companies to store the call details and monthly postpaid bills.

### **1.5.8. Finance**

Those days have gone far when information related to money was stored in registers and files. Today the time has totally changed because there are lots of thing to do with finance like storing sales, holding information and finance statement management etc.

### **1.5.9. Military**

Military keeps records of millions of soldiers and it has millions of files that should be keep secured and safe. As DBMS provides a big security assurance to the military information so it is widely used in militaries. One can easily search for all the information about anyone within seconds with the help of DBMS.

### **1.5.10. Railway Reservation System**

Database is required to keep record of ticket booking, train's departure and arrival status. Also, if trains get late then people get to know it through database update.

## CHAPTER 2

# REQUIREMENT ANALYSIS

### 2.1. Input requirements:

User must be able to maintain the record of the valuables of the customer kept in the locker. He could also be able to view the information of the customer, and he can view the details of the locker.

### 2.2. Output requirements:

The main outcome of the project “Bank Locker Management System” is offering better solution for bank security system. It manages all the information about bank locker. Bank offers locker facilities to individuals at very small annual fee. These lockers are maintained in a secure facility that is under constant surveillance and security.

### 2.3. Hardware requirements:

- Processor: Intel Core i5
- Processor Speed: 1 GHz
- Solid State Drive (SSD): 512 GB
- Random Access Memory (RAM): 1GB
- System Type: 64 Bit

### 2.4. Software requirements:

- Operating System (OS): Windows 10
- Front End Tool: PHP
- Back End Tool: My SQL
- Database: MySQL

## CHAPTER 3

# DESIGN

Database design is the process of producing a detailed data model of database. This data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

The term database design can be used to describe many different parts of the design of an overall database system. Principally, and most correctly, it can be thought of as the logical design of the base data structures used to store the data. In the relational model these are the tables and views. In an object database the entities and relationships map directly to object classes and named relationships. However, the term database design could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application within the database management system.

### 3.1. Entity-Relationship Diagram(Model)

Entity-Relationship (ER) Model is based on the notion of real-world entities and relationships among them. While formulating real-world scenario into the database model, the ER Model creates entity set, relationship set, general attributes and constraints.

ER Model is best used for the conceptual design of a database. ER Model is based on,

- Entities and their attributes.

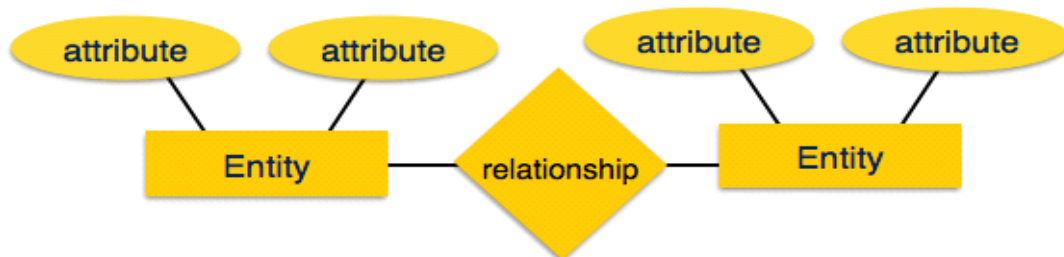


Fig. 3.1: Relationships among entities.

- **Entity** –An entity in an ER Model is a real-world entity having properties called attributes. Every attribute is defined by its set of values called domain. For example, in

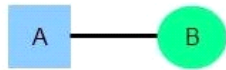
a school database, a student is considered as an entity. Student has various attributes like name, age, class, etc.

- **Relationship** –The logical association among entities is called relationship. Relationships are mapped with entities in various ways. Mapping cardinalities define the number of associations between two entities.

- **Mapping cardinalities**

- **one to one**

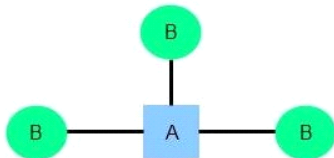
One instance of an entity (A) is associated with one other instance of another entity (B). For example, in a database of customer, each customer name (A) is associated with only one user number (B).



- **one to many**

One instance of an entity (A) is associated with zero, one or many instances of another entity (B), but for one instance of entity B there is only one instance of entity.

A The administrator has all the information about all the users and about all products. Only admin have access into this admin page. Admin may be the owner of the shop.



- **many to many**

One instance of an entity (A) is associated with one, zero or many instances of another entity (B), and one instance of entity B is associated with one, zero or many instances of entity A. The application was designed into two modules first is for the customers who wish to buy the articles.

Once the authorized personnel feed the relevant data into the system, several reports could be generated as per the security.

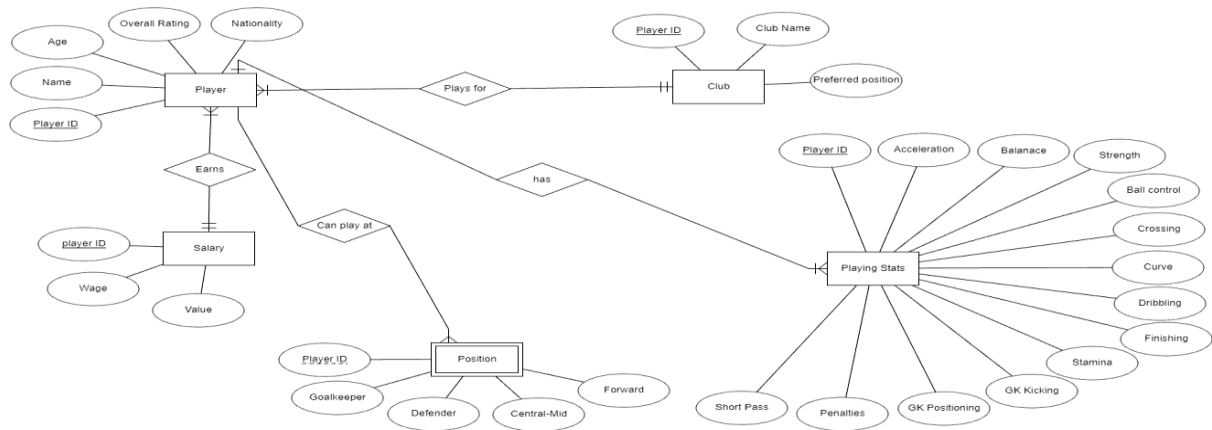


Fig. 3.2: E-R Diagram for FIFA 18 Player Management System

### 3.2. Relational Model

The most popular data model in DBMS is the Relational Model. It is more scientific a model than others. This model is based on first-order predicate logic and defines a table as an n-array relation. For an ex:

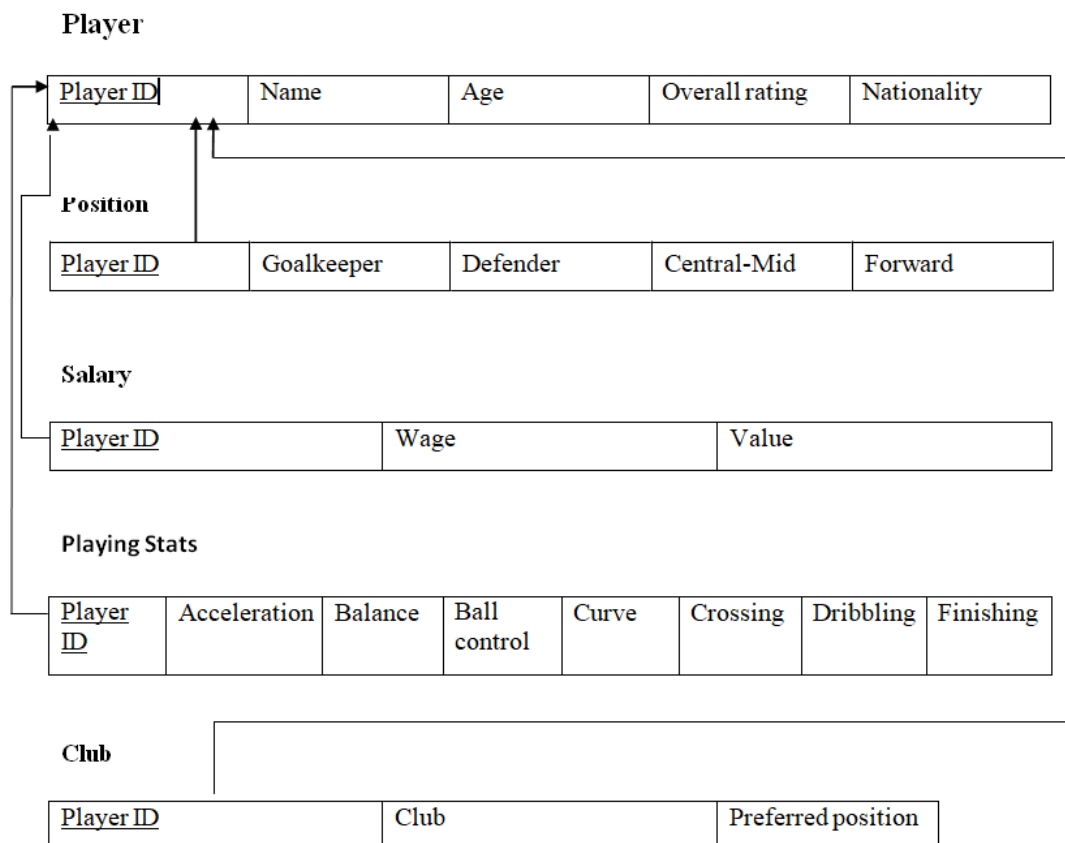


Fig 3.2 Relational schema

The main highlights of this model are

- Data is stored in tables called relations.
- Relations can be normalized.
- In normalized relations, values saved are atomic values.
- Each row in a relation contains a unique value.
- Each column in a relation contains values from a same domain.

### 3.3. Database Schema

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.

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. It's the database designers who design the schema to help programmers understand the database and make it useful. For an ex:

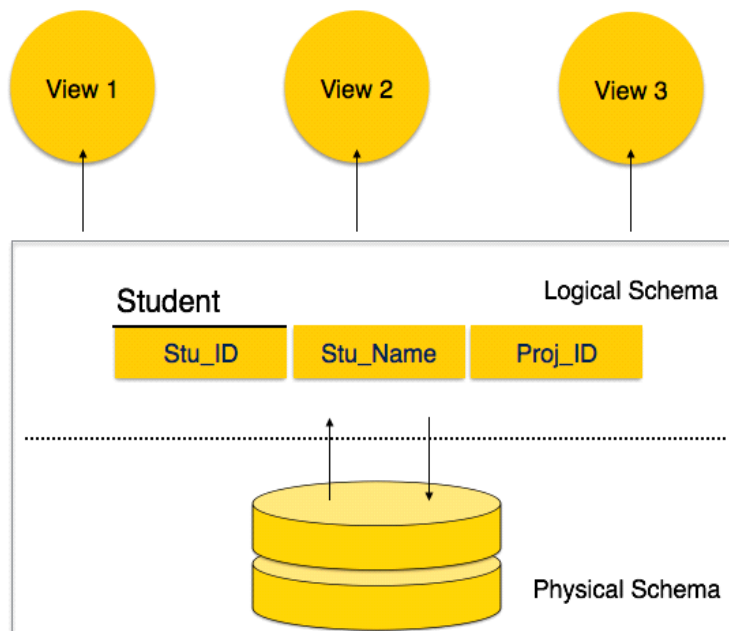
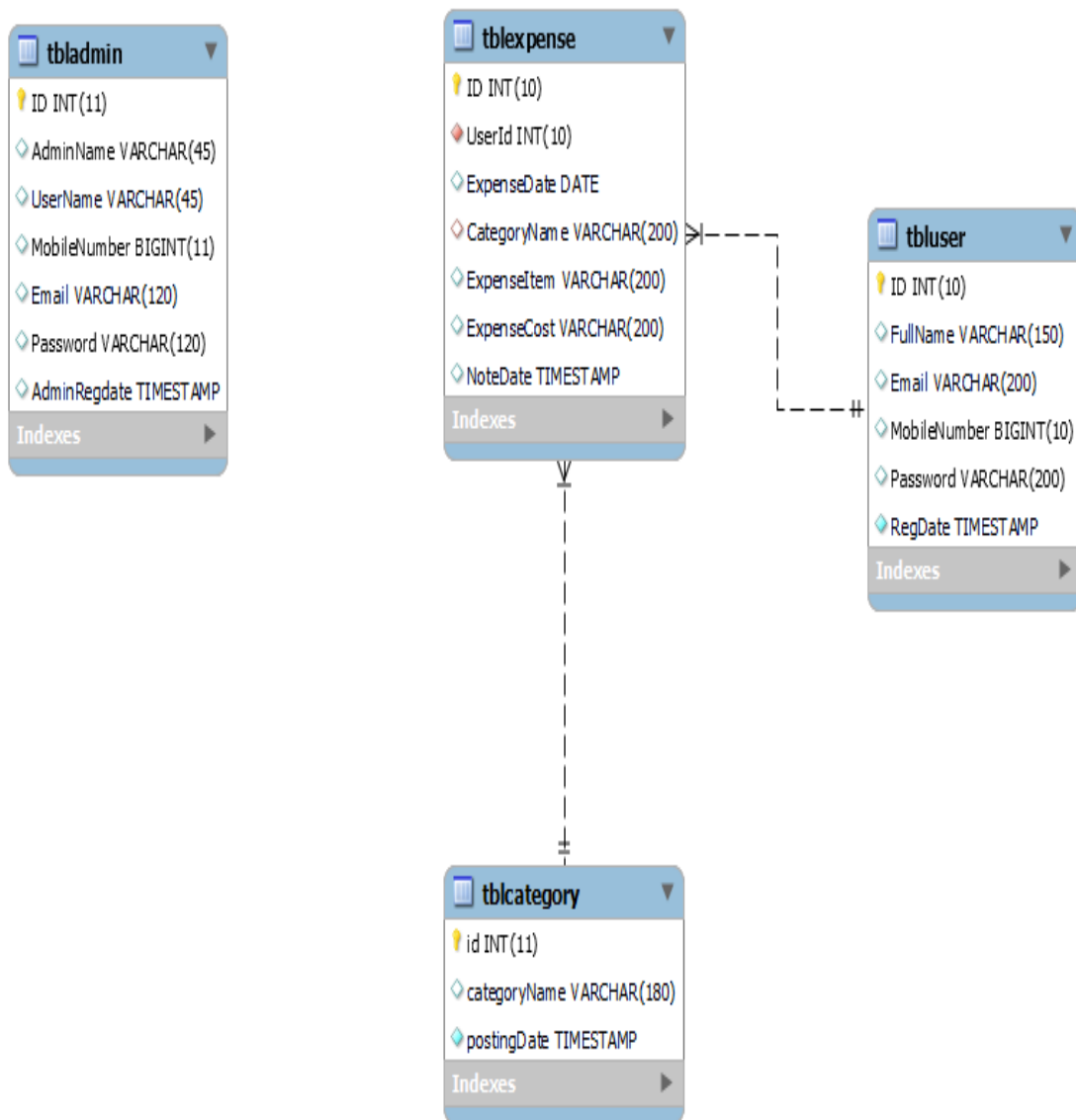


Fig. 3.4: Database Schema

A database schema can be divided broadly into two categories

- **Physical Database Schema** – This schema pertains to the actual storage of data and its form of storage like files, indices, etc. It defines how the data will be stored in a secondary storage.
- **Logical Database Schema** – This schema defines all the logical constraints that need to be applied on the data stored. It defines tables, views, and integrity constraints.

Fig 3.5: Schema Design



**FIFA 18 Player Management System contains 9 MySQL tables:**

- **Table Structures:**
- **Player details**

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	id	int(11)			No	None		AUTO_INCREMENT	Change  Drop  More
<input type="checkbox"/> 2	player_id	int(7)			No	None			Change  Drop  More
<input type="checkbox"/> 3	player_name	char(30)	latin1_swedish_ci		No	None			Change  Drop  More
<input type="checkbox"/> 4	age	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 5	overall_rating	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 6	nationality	char(30)	latin1_swedish_ci		Yes	NULL			Change  Drop  More

Fig 3.3 Player table structure

The player table consists of 6 columns. Player\_id and player\_name are primary keys and player\_id have references of other tables as well. To insert data, player\_id should exist in this table before inserting into other tables.

- **Stats details**

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	id	int(3)			No	None		AUTO_INCREMENT	Change  Drop  More
<input type="checkbox"/> 2	player_id	int(7)			No	None			Change  Drop  More
<input type="checkbox"/> 3	acceleration	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 4	balance	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 5	ball_control	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 6	crossing	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 7	curve	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 8	dribbling	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 9	finishing	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 10	gk_kicking	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 11	gk_positioning	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 12	penalties	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 13	short_pass	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 14	stamina	int(2)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 15	strength	int(2)			Yes	NULL			Change  Drop  More

Fig 3.4 Player stats table structure

Player stats table consists of 14 attributes, among which, player\_id is primary key and also has a foreign key reference to “player” table. It is designed to contain all the football technicalities of a player.

- **Salary details**



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	id	int(11)			No	None		AUTO_INCREMENT	Change  Drop  More
<input type="checkbox"/> 2	player_id	int(11)			No	None			Change  Drop  More
<input type="checkbox"/> 3	wage	int(11)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 4	value	int(11)			Yes	NULL			Change  Drop  More

Fig 3.5 Salary table structure

The salary table consists player\_id as primary key and also have a foreign key reference to “player” table. It is designed to store player weekly wage and his current value in the market.

#### • Position details

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	id	int(11)			No	None		AUTO_INCREMENT	Change  Drop  More
<input type="checkbox"/> 2	player_id	int(11)			No	None			Change  Drop  More
<input type="checkbox"/> 3	gk	int(11)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 4	df	int(11)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 5	cm	int(11)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 6	fr	int(11)			Yes	NULL			Change  Drop  More

Fig 3.6 Position table structure

The position table also have player\_id as primary key and a foreign key reference to “player” table. It is designed to store the positions a player can play, if so, then how well he does on a scale of rating from 0 to 99. It helps user to access player based on positional play and decide the best position for a player.

#### • Club details

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	id	int(11)			No	None		AUTO_INCREMENT	Change  Drop  More
<input type="checkbox"/> 2	player_id	int(11)			No	None			Change  Drop  More
<input type="checkbox"/> 3	club	char(30)	latin1_swedish_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/> 4	preferred_position	char(20)	latin1_swedish_ci		Yes	NULL			Change  Drop  More

Fig 3.7 Club table structure

The club details table has club information and the preferred position of a player at that club. It also have player\_id as primary key and also a foreign key reference on “player” table.

#### • Delete logs

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	<b>id</b>	int(10)			No	None		AUTO_INCREMENT	Change  Drop  More
<input type="checkbox"/> 2	<b>action</b>	varchar(50)	utf8mb4_unicode_ci		No	None			Change  Drop  More
<input type="checkbox"/> 3	<b>time</b>	timestamp			No	None			Change  Drop  More

Fig 3.8 Delete logs trigger structure

The delete logs table consists of 3 columns. ID column is unique and set to auto increment. Action column contain the action along with table name. Time column contains the time at which the trigger was automatically invoked based on the action.

#### • Update logs

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	<b>id</b>	int(10)			No	None		AUTO_INCREMENT	Change  Drop  More
<input type="checkbox"/> 2	<b>action</b>	varchar(50)	utf8mb4_unicode_ci		No	None			Change  Drop  More
<input type="checkbox"/> 3	<b>time</b>	timestamp			No	None			Change  Drop  More

Fig 3.9 Update logs trigger structure

The Update logs table consists of 3 columns. ID column is unique and set to auto increment. Action column contain the action along with table name. Time column contains the time at which the trigger was automatically invoked based on the action.

#### • Insert logs

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	<b>id</b>	int(10)			No	None		AUTO_INCREMENT	Change  Drop  More
<input type="checkbox"/> 2	<b>action</b>	varchar(50)	utf8mb4_unicode_ci		No	None			Change  Drop  More
<input type="checkbox"/> 3	<b>time</b>	timestamp			No	None			Change  Drop  More

Fig 3.10 Insert logs trigger structure

The Insert logs table consists of 3 columns. ID column is unique and set to auto increment. Action column contain the action along with table name. Time column contains the time at which the trigger was automatically invoked based on the action.

There are 7 stored procedures present inside search page on the web application. These are called whenever any search instance occur on the web page. The results of the stored procedures are then displayed on the UI in a tabular structure.

## CHAPTER 4

### IMPLEMENTATION

#### 4.1. FRONT END PHP

PHP started out as a small open-source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

- PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
- PHP is a server-side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites.
- It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.
- PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time.
- PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures (COM and CORBA), making n-tier development a possibility for the first time.
- PHP is forgiving: PHP language tries to be as forgiving as possible.
- PHP Syntax is C-Like.

#### 4.2. BACKEND TOOLMYSQL

**MySQL** is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter My, and "SQL", the acronym for Structured Query Language. A relational database organizes data into one or more data tables in which data may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to

implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.

MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation). In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create MariaDB.

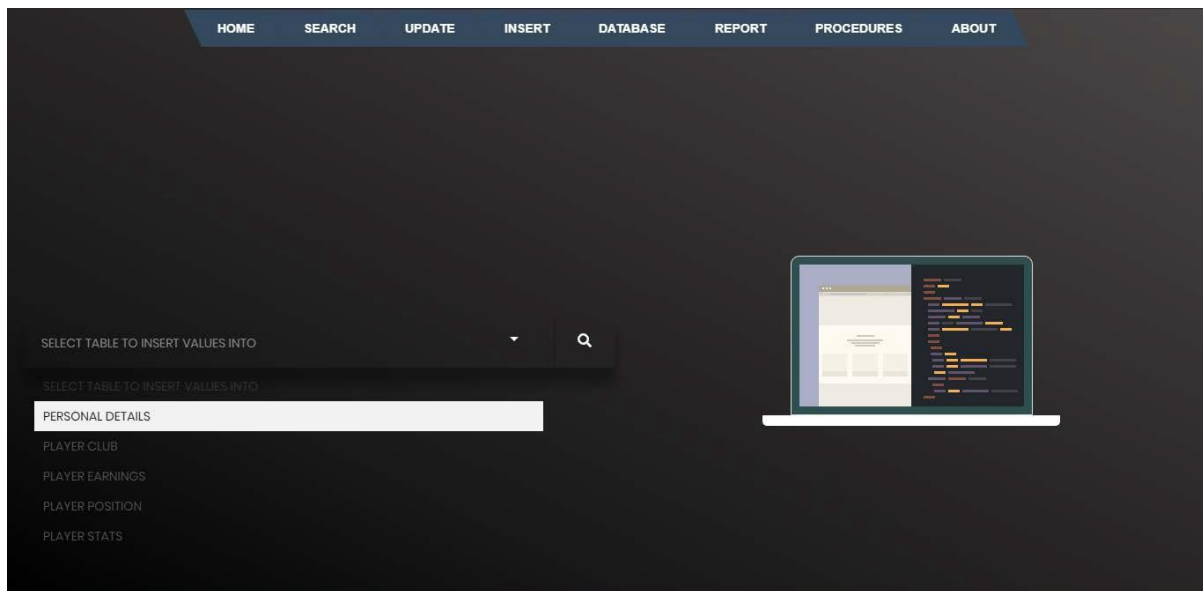
MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often, MySQL is used with other programs to implement applications that need relational database capability. MySQL is a component of the LAMP web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and WordPress. MySQL is also used by many popular websites, including Facebook, Flickr, MediaWiki, Twitter, and YouTube.

MySQL was created by a Swedish company, MySQL AB, founded by Swedes David Axmark, Allan Larsson and the API consistent with the mSQL system, many developers were able to Finland Swede Michael "Monty" Widenius. Original development of MySQL by Widenius and Axmark began in 1994.<sup>[22]</sup> The first version of MySQL appeared on 23 May 1995. It was initially created for personal usage from mSQL based on the low-level language ISAM, which the creators considered too slow and inflexible. They created a new SQL interface, while keeping the same API as mSQL. By keeping use MySQL instead of the (proprietary licensed) mSQL antecedent.

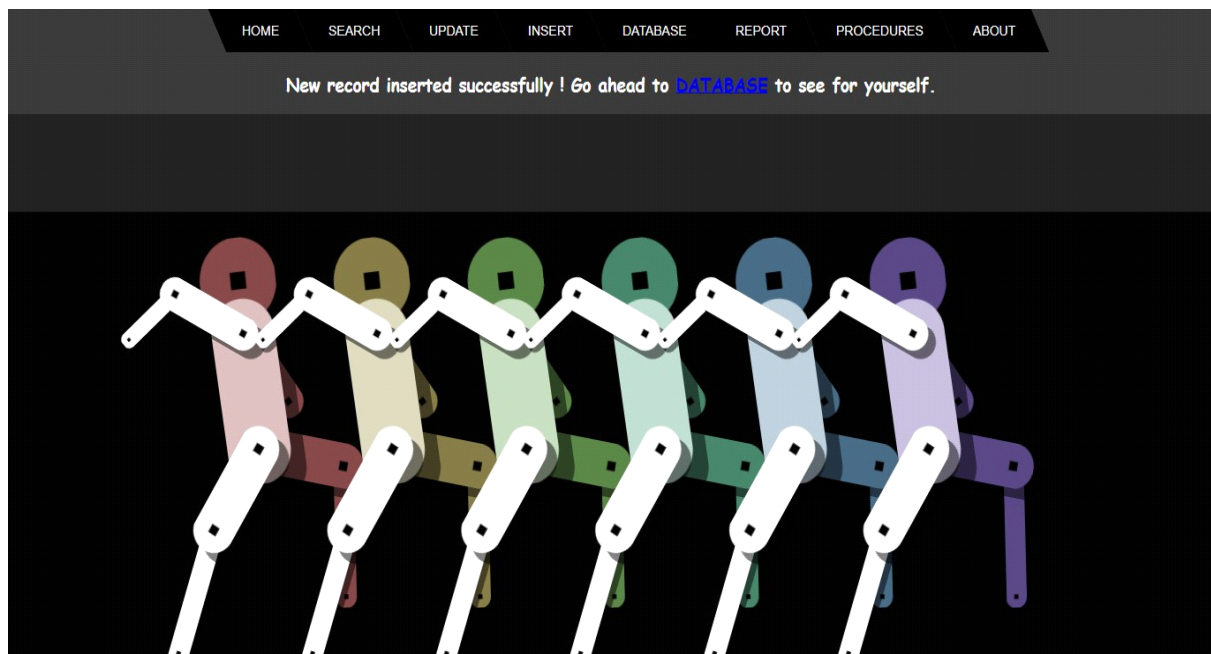
## CHAPTER 5

# RESULT

**Inserting new records:**



















**Fig 5.1** Selecting table for insertion



**Fig 5.2** Successful insert instance page

## Update existing records:

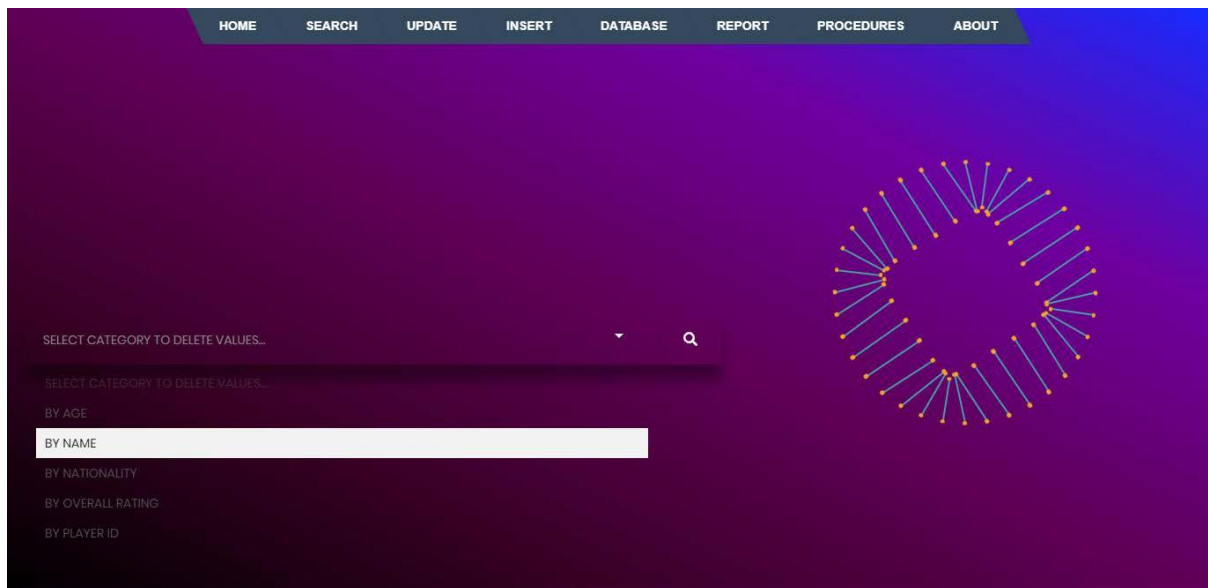
**Fig 5.3 Selecting table to modify record**

HOME SEARCH UPDATE INSERT DATABASE REPORT PROCEDURES ABOUT Refresh						
ID	NAME	AGE	OVERALL RATING	NATIONALITY		
20801	CRISTIANO RONALDO	32	94	Portugal	 	Save
138956	G CHIellini	32	89	Italy	 	
153079	S AGUERO	29	89	Argentina	 	
155862	SERGIO RAMOS	31	90	Spain	 	
158023	LIONEL MESSI	30	94	Argentina	 	
167495	M NEUER	31	92	Germany	 	
167864	G HIGUAIN	29	90	Argentina	 	
173731	G BALE	27	89	Wales	 	

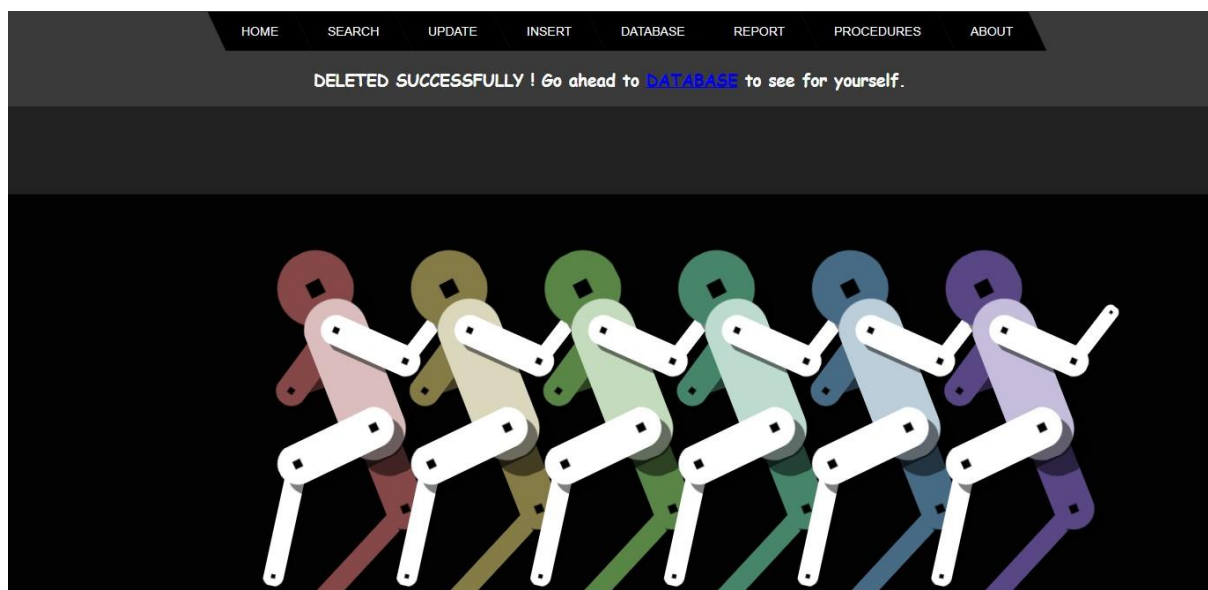
**Fig 5.4 Modifying records in real time**



## Deleting records:



**Fig 5.5** Selecting categories for deleting record



**Fig 5.6** Successful deletion instance



**Triggers:**

DELETE TRIGGERS		
ID	ACTION	TIME
15	Deleted Successfully in PERSONAL DETAILS Table	2018-12-09 13:14:33
16	Deleted Successfully in PLAYER CLUB'S Table	2018-12-09 13:43:36
17	Deleted Successfully in PLAYER CLUB'S Table	2018-12-09 13:50:21
18	Deleted Successfully in PLAYER'S POSITION Table	2018-12-09 13:56:29
19	Deleted Successfully in PLAYER'S SALARY Table	2018-12-09 15:33:03
20	Deleted Successfully in PLAYER'S POSITION Table	2018-12-09 15:33:42

**Fig 5.7 Delete triggers table**

INSERT TRIGGERS		
ID	ACTION	TIME
13	Inserted Successfully in PERSONAL DETAILS Table	2018-12-09 13:07:05
14	Inserted Successfully in PERSONAL DETAILS Table	2018-12-09 13:39:09
15	Inserted Successfully in PLAYER CLUB'S Table	2018-12-09 13:39:31
22	Inserted Successfully in PLAYER STATS Table	2018-12-09 13:42:05
23	Inserted Successfully in PLAYER'S POSITION Table	2018-12-09 13:42:18
24	Inserted Successfully in PLAYER SALARY Table	2018-12-09 13:42:29

**Fig 5.8 Insert triggers table**

UPDATE TRIGGERS		
ID	ACTION	TIME
10	Updated Successfully in PERSONAL DETAILS Table	2018-12-09 13:17:32
11	Updated Successfully in PERSONAL DETAILS Table	2018-12-09 13:17:50
12	Updated Successfully in PERSONAL DETAILS Table	2018-12-09 13:44:47
13	Updated Successfully in CLUB'S Table	2018-12-09 13:49:48
14	Updated Successfully in CLUB'S Table	2018-12-09 13:50:00
15	Updated Successfully in CLUB'S Table	2018-12-09 13:50:12
16	Updated Successfully in PLAYER SALARY Table	2018-12-09 15:31:00

**Fig 5.9 Update triggers table**

**Database tables:**

PLAYER CLUB		
PLAYER ID	CLUB	PREFERRED POSITION
20801	Real Madrid CF	LW
158023	FC Barcelona	RW
190871	Paris Saint-Germain	LW
176580	FC Barcelona	ST
167495	FC Bayern Munich	GK
188545	FC Bayern Munich	ST

**Fig 5.10 Player club table**

PLAYER EARNINGS		
PLAYER ID	WAGE	VALUE
20801	565	9550000
158023	565	10500000
190871	280	12300000
176580	510	9700000
167495	230	6100000
188545	335	9200000

**Fig 5.11 Player salary table**

PLAYER POSITIONS				
PLAYER ID	GOALKEEPER	DEFENDER	CENTRAL-MID	FORWARD
20801	13	26	82	94
158023	6	45	82	94
190871	10	46	79	93
176580	12	50	80	92
167495	92	10	8	4
188545	12	57	78	91

**Fig 5.12 Player position table**

PLAYER STATS													
PLAYER ID	ACCELERATION	BALANCE	BALL CONTROL	CROSSING	CURVE	DRIBBLING	FINISHING	GK KICKING	GK POSITIONING	PENALTIES	SHORT PASS	STAMINA	STRENGTH
20801	89	63	93	85	81	91	94	15	14	85	83	92	80
158023	92	95	95	77	89	97	95	15	14	74	88	73	59
190871	94	82	95	75	81	96	89	15	15	81	81	78	53
176580	88	60	91	77	86	86	94	31	33	85	83	89	80
167495	58	35	48	15	14	30	13	95	91	47	55	44	83
188545	79	80	89	62	77	85	91	12	8	81	83	79	84

Fig 5.13 Player stats table

PERSONAL DETAILS				
PLAYER ID	NAME	AGE	OVERALL RATING	NATIONALITY
20801	CRISTIANO RONALDO	32	94	Portugal
158023	LIONEL MESSI	30	94	Argentina
190871	NEYMAR	25	92	Brazil
176580	LUIS SUAREZ	30	92	Uruguay
167495	M NEUER	31	92	Germany
188545	R LEWANDOWSKI	28	91	Poland

Fig 5.14 Player's table

## CONCLUSION

This project is developed to nurture the needs of a user/scouting agent to monitor players and inspect their technicalities from every aspect on a football field. This is a computerized version of player management system which will benefit the players as well as the staff of a club. In this entire process one can search player details, add new skilled players, update ratings and view all the player statistics. The software takes care of data and carefully stores all the player information. It provides security and encapsulation by the use of stored procedures.

## REFERENCES

- [1] Ramez Elmasri and Shamkant B. Navathe, *Fundamentals of Database Systems*. Addison Wesley Publishing Company. 57-90
- [2] MySQLQueries[Retrievedon29/10/2019from<http://dev.mysql.com/doc/refman/5.7/en/>]
- [3] Ramakrishnan, and Gehrke, *Database Management Systems*, 3<sup>rd</sup> edition, 2014, Mc- GrawHill.

