CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION TO SQL

SQL which is an abbreviation for **Structured Query Language** is a language to request data from a database, to add, update, or remove data within a database, or to manipulate the metadata of the database.

Sometimes SQL is characterized as non-procedural because procedural languages generally require the details of the operations to be specified, such as opening and closing tables, loading and searching indexes, or flushing buffers and writing data to file systems. Therefore, SQL is designed at a higher conceptual level of operation than procedural languages.

Commonly used statements are grouped into the following categories

Data Query Language (DQL)

SELECT-Used to retrieve certain records from one or more tables.

Data Manipulation Language (DML)

INSERT - Used to create a record

UPDATE - Used to change certain records.

DELETE - Used to delete certain records.

Data Definition Language (DDL)

CREATE - Used to create a new table, a view of a table, or other object in database.

ALTER - Used to modify an existing database object, such as a table.

DROP - Used to delete an entire table, a view of a table or other object in the database.

Data Control Language (DCL)

GRANT - Used to give a privilege to someone

REVOKE - Used to take back privileges granted to someone.

1.2 INTRODUCTION TO FRONT END SOFTWARE

The “front end languages” live in the browser. After you type in an address in the address bar at the top and hit Enter, your browser will receive an at least an HTML file from the web server.

Each of these languages performs a separate but very important function but the work harmoniously together to determine how the web page is STRUCTURED(HTML), how it LOOKS(CSS), and how its FUNCTIONS (JavaScript).

Front end web development is NOT design (You won't be playing around in Photoshop or anything), but a *front-end developer* does apply the work of designers to the web page by translating their well-designed layouts into real code. The front-end developer stands between the designer on one end and the back-end developer on the other, translating the design into code and plugging the data from the back-end developer into the right spots.

PHP is a server-side scripting language designed primarily for web development but also used as a general-purpose programming language. Originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Development Team.

PHP code may be embedded into HTML or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server software combines the result of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and it can be used to implement stand-alone graphical applications.

The standard PHP interpreter, powered by the Zend Engine, is free to use software released under the PHP License. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as Common Gateway Interface(CGI) executable. PHP has been widely ported on web servers on almost every operating system and platform, free of charge.

CHAPTER 2

REQUIREMENTS SPECIFICATION

2.1 SOFTWARE REQUIREMENTS

Operating System	:	64bit WINDOWS Operating System, X64-based processor
Database	:	MYSQL
Scripting Language	:	HTML5, PHP
Server	:	WAMP

2.2 HARDWARE REQUIREMENTS

Processor	:	Intel Celeron CPU N3060 @1.60GHz or Above
RAM	:	4.00 GB or Above
Hard Disk	:	1 TB
Compact Disk	:	CD-ROM, CD-R, CD-RW
Input device	:	Keyboard

CHAPTER 3

OBJECTIVE OF THE PROJECT

The main objective of creating a Sports Academy Management System database project is

1. Improvement in control and performance

   The system is developed to cope up with the current issues and problems of sports academy. The system can add item, search item, display items, delete the items and also bug free.

2. Maintaining and managing well

   This database helps the sports academy to manage and maintains the information regarding members, courses, assigned instructors and information regarding the games and tournaments

3. Save time

   This system allows the staffs of sports academy to maintain correct records of all members of the academy.

CHAPTER 4

IMPLEMENTATION

4.1 ER DIAGRAM

An **entity relationship diagram (ERD)** shows the relationships of entity sets stored in a database.

An entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases.

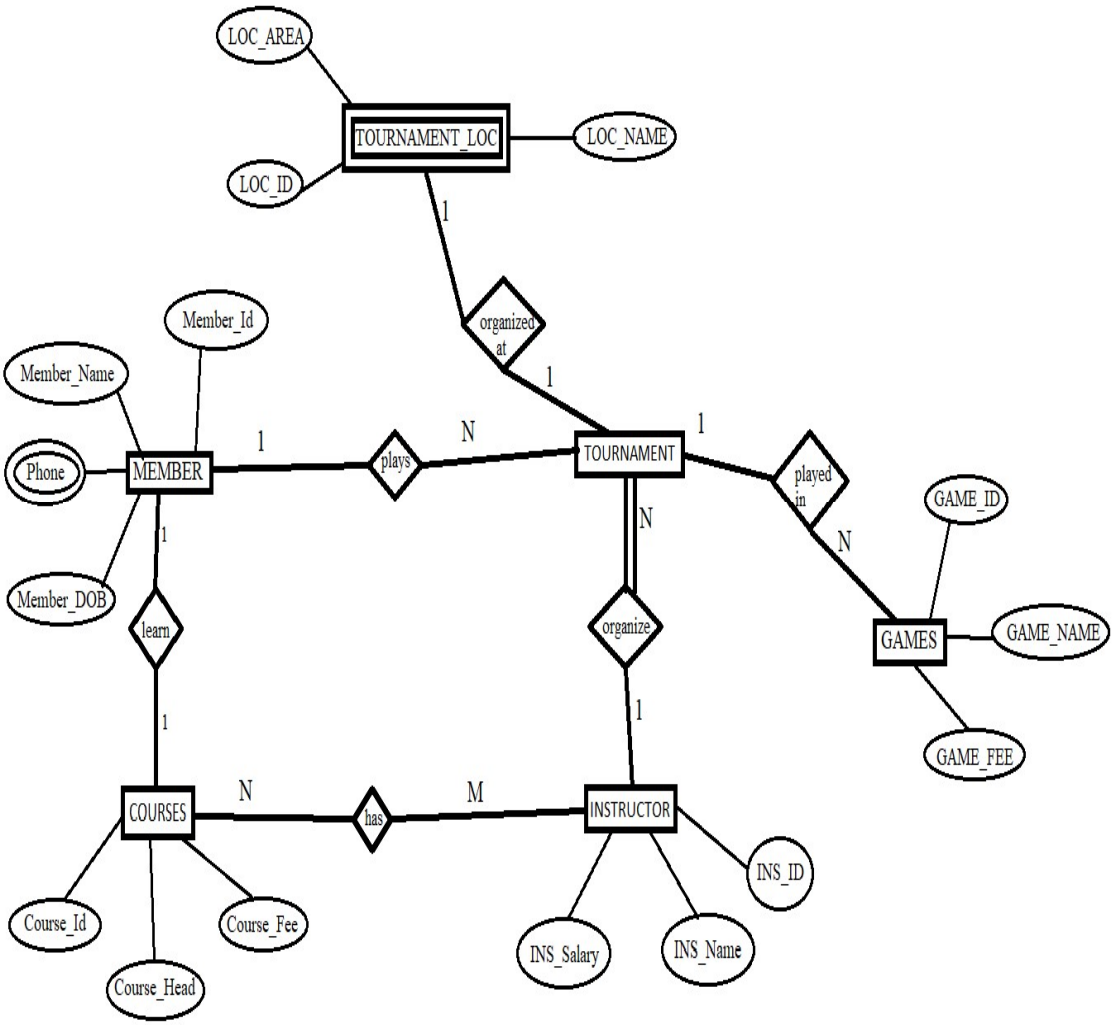


FIGURE 4.1: ER DIAGRAM

4.2 MAPPING OF ER DIAGRAM TO RELATIONS

STEP 1: Mapping of Regular Entities

For each regular entity type E in the ER schema, create relation R that includes all simple attributes of E.

MEMBER

MEMBER_ID	MEMBER_NAME	MEMBER_DOB

COURSE

COURSE_ID	COURSE_HEAD	COURSE_FEE

INSTRUCTOR

INS_ID	INS_NAME	TILE_NAME	**COURSE_ID**	INS_SALARY

GAMES

GAMES_ID	GAME_NAME	GAME_FEE

TOURNAMENT

TOUR_ID	PRIZE	**MEMBER_ID**

STEP 2: Mapping of Weak Entity Types

For each weak entity, create a table that includes all of its simple attributes.

TOURNAMENT_LOC

LOC_ID	LOC_NAME	LOC_AREA	TOUR_ID

STEP 3: Mapping of 1-1 Relationship

Identify the relation S that represents the participating entity type at the 1-side of the relationship type.

Include as foreign key in S the primary key of the relations T that represents the other entity type participating in R.

For each binary 1:1 relationship type R in ER schema, identify the relations S and T that correspond to the entity types participating in R if any.

There are no 1:1 relationship.

COURSE

COURSE ID	COURSE_HEAD	COURSE_FEE

MEMBER

MEMBER_ID	MEMBER_NAME	PHONE	MEMBER_DOB

STEP 4: Mapping of 1-N Relationship

To map 1:N relationships, the primary key in the 'one side' of the relationship is added to the 'many side' as a foreign key.

MEMBER

MEMBER_ID	MEMBER_NAME	PHONE	MEMBER_DOB

TOURNAMENT

TOUR_ID	PRIZE	**MEMBER_ID**

INSTRUCTOR

INS_ID	INS_NAME	**COURSE_ID**	INS_SALARY

GAMES

GAME_ID	GAME_NAME	GAME_FEE

STEP 5: Mapping of M-N Relationship

Create a new relation S to represent R.

Include as foreign key attributes in S the primary key of the relations that represents the participating entity types their combination will form the primary key of S.

Also, include any simple attributes of the M:N relationship type as attributes of S.

COURSE

COURSE_ID	COURSE_HEAD	COURSE_FEE

INSTRUCTOR

INS_ID	INS_NAME	**COURSE_ID**	INS_SALARY

STEP 6: Mapping of multi-valued attributes

For each multivalued attributes A, create a new relation R. This relation R will include an attribute corresponding to A. plus the primary key attribute K-as a foreign key in R-of the relation that represents the entity type of relationship type that has A as an attribute.

The Primary Key of R is the combination of A and K. If the multivalued attribute is composite, we include its simple components.

CONTACTS

PHONE	**MEMBER_ID**

FIGURE 4.2: MAPPING OF ER DIAGRAM TO RELATIONS

STEP 7.: Mapping of N-Ary Relationship Types

For each n-ary relationship type R, where n>2 create a new relationship S to represent R. λ include as foreign key attributes in S the primary keys of the relations that represent the participating entity types.

λ also includes any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S

There are **no** n-ary relationship types.

4.3 SCHEMA DIAGRAM

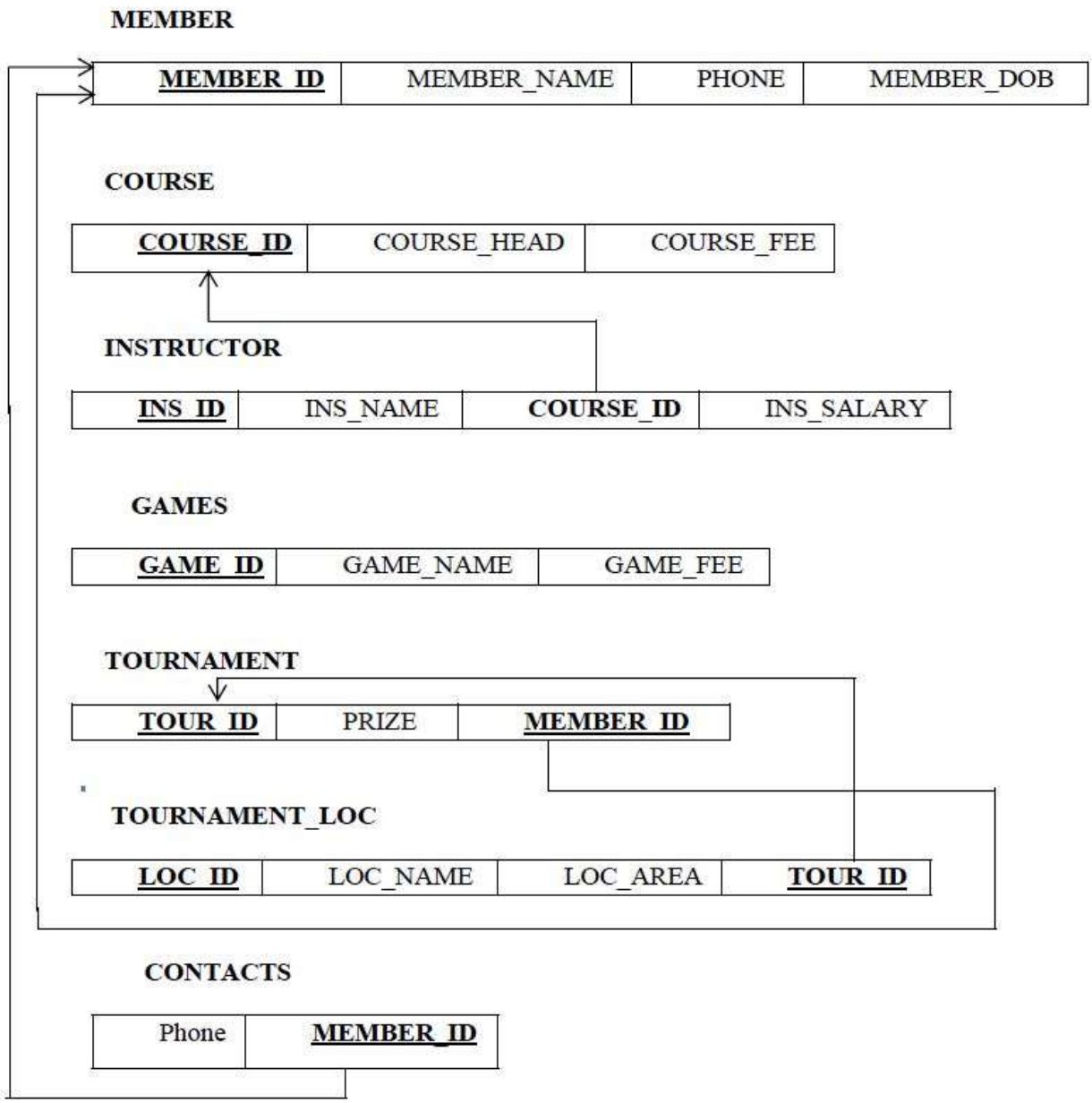


FIGURE 4.3: SCHEMA DIAGRAM

4.4 NORMALIZE THE RELATIONS

Database normalization, or simply normalization, is the process of organizing the columns(attributes) and tables(relations) of a relational database to reduce data redundancy and improve data integrity. Normalization involves arranging attributes in relations based on dependencies between attributes.

1. **First Normal Form**

As per First normal form, no two rows of data must contain repeating group of information. Each set of columns must have a unique value, such that multiple columns cannot be used to fetch the same row. Each table should be organized into rows, and each row should have a primary key that will distinguishes it as unique.

Example:

GAMES

GAME_ID	GAME_NAME	GAME_FEE

All the tables in the database are normalized to 1NF as all the attributes are atomic.

2. **Second Normal Form (2NF)**

A table is in 2NF if it is in 1NF and if all non-key attributes are fully functionally dependent on all of the key.

Example

MEMBER

MEMBER_ID	MEMBER_NAME	MEMBER_DOB

3. **Third Normal Form(3NF):**

A table is in 3NF if it is in 2NF and if it has no transitive dependency. X->Y, Y->Z, X>Z According to CODD's definition a relation schema R is in 3NF. It satisfies 2NF and no non-prime attribute of R is transitively dependent on the primary key. Since the database is relationally-schema mapped the relations are normalized and do not have any 3NF. All tables in database satisfies up to 3NF.

4.5 CREATION OF TABLES

CREATE TABLE MEMBER

```
CREATE TABLE MEMBER
(
MEMBER_ID VARCHAR(10) PRIMARY KEY,
MEMBER_NAME VARCHAR(20),
PHONE VARCHAR(15),
MEMBER_DOB VARCHAR(10)
);
```

CREATE TABLE COURSE

```
CREATE TABLE COURSE
(
COURSE_ID VARCHAR(20) PRIMARY KEY
COURSE_HEAD VARCHAR(20),
COURSE_FEE INT(5)
);
```

CREATE TABLE GAMES

```
CREATE TABLE GAMES
(
GAME_ID VARCHAR(10) PRIMARY KEY,
GAME_NAME VARCHAR(10),
GAME_FEE INT(5)
);
```

CREATE TABLE INSTRUCTOR

```
CREATE TABLE INSTRUCTOR
(
INS_ID VARCHAR(10) PRIMARY KEY,
```

```
INS_NAME VARCHAR(50),
COURSE_ID VARCHAR(20),
COURSE_ID REFERENCES COURSE(COURSE_ID) ON DELETE CASCADE,
INS_SALARY INT(10)
);
```

CREATE TABLE TOURNAMENT

```
CREATE TABLE TOURNAMENT
(
TOUR_ID VARCHAR(20) PRIMARY KEY,
PRIZE INT(5),
MEMBER_ID VARCHAR(20),
MEMBER_ID REFERENCES MEMBER(MEMBER_ID) ON DELETE CASCADE
);
```

CREATE TABLE TOURNAMENT LOC

```
CREATE TABLE TOURNAMENT_LOC
(
LOC_ID VARCHAR(20) PRIMARY KEY,
LOC_NAME VARCHAR(50),
LOC_AREA VARCHAR(20),
TOUR_ID REFERENCES TOURNAMENT(TOUR_ID) ON DELETE CASCADE
);
```

CREATE TABLE CONTACT

```
CREATE TABLE CONTACT
(
PHONE VARCHAR(15),
MEMBER_ID VARCHAR(20),
MEMBER_ID REFERENCES MEMBER(MEMBER_ID) ON DELETE CASCADE
);
```

4.6 INSERTION OF TUPLES

INSERT VALUES TO MEMBER

INSERT INTO MEMBER VALUES(M101,”ABHINAV SINGH”,9656896542,1974-03-27);

INSERT INTO MEMBER VALUES(M102,”HARSHVARDHAN RAJ”,9959965632,1975-11-12);

INSERT INTO MEMBER VALUES(M103,”RAJESH RANJAN”,9886785489,1972-05-19);

INSERT INTO MEMBER VALUES(M104,”NAVEEN BHARTI”,7895648973,1981-01-21);

INSERT INTO MEMBER VALUES(M105,”AJAY CHOUDHARY”,8974623148,1980-04-22);

```
mysql> select * from member;
+-----------+------------------+------------+------------+
| MEMBER_ID | MEMBER_NAME      | PHONE      | MEMBER_DOB |
+-----------+------------------+------------+------------+
| M101      | ABHINAV SINGH    | 9656896542 | 1974-03-27 |
| M102      | HARSHVARDHAN RAJ | 9959965632 | 1975-11-12 |
| M103      | RAJESH RANJAN    | 9886785489 | 1972-05-19 |
| M104      | NAVEEN BHARTI    | 7895648973 | 1981-01-21 |
| M105      | AJAY CHOUDHARY   | 8974623148 | 1980-04-22 |
+-----------+------------------+------------+------------+
5 rows in set (0.05 sec)
```

INSERT VALUES TO COURSE

INSERT INTO COURSE VALUES(“01”,”SHYAM SINGH”,10000);

INSERT INTO COURSE VALUES(“02”,”BHUSHAN KUMAR”,15000);

INSERT INTO COURSE VALUES(“03”,”ASHUTOSH RAINA”,14000);

INSERT INTO COURSE VALUES(“04”,”SURESH PRASAD”,18000);

INSERT INTO COURSE VALUES(“05”,”ANIL KUMAR”,11000);

```
mysql> select * from course;
+-----------+----------------+-----------+
| COURSE_ID | COURSE_HEAD    | COURSE_FEE |
+-----------+----------------+-----------+
| 01        | SHYAM SINGH    |      10000 |
| 02        | BHUSHAN KUMAR  |      15000 |
| 03        | ASHUTOSH RAINA |      14000 |
| 04        | SURESH PRASAD  |      18000 |
| 05        | ANIL KUMAR     |      11000 |
+-----------+----------------+-----------+
5 rows in set (0.03 sec)
```

INSERT VALUES TO INSTRUCTOR

INSERT INTO INSTRUCTOR VALUES(“INS001”,”SUNIL THAKUR”,”01”,45000);

INSERT INTO INSTRUCTOR VALUES(“INS002”,”RAMESH VERMA”,”02”,45000);

INSERT INTO INSTRUCTOR VALUES(“INS003”,”ARUN LAL”,”03”,45000);

INSERT INTO INSTRUCTOR VALUES(“INS004”,”ASHOK KUMAR”,”04”,45000);

INSERT INTO INSTRUCTOR VALUES(“INS005”,”MANISH SINGH”,”05”,45000);

```
mysql> select * from instructor;
+-------+-------------+----------+-----------+---------------------+
| INS_ID | INS_NAME    | COURSE_ID | INS_SALARY | Entry_Time          |
+-------+-------------+----------+-----------+---------------------+
| INS003 | ARUN LAL    | 03       |     32000 | 2019-11-10 17:30:06 |
| INS002 | RAMESH VERMA | 02       |     40000 | 2019-11-10 17:29:09 |
| INS001 | SUNIL THAKUR | 01       |     45000 | 2019-11-10 17:27:43 |
| INS004 | ASHOK KUMAR | 04       |     38000 | 2019-11-10 17:30:06 |
| INS005 | MANISH SINGH | 05       |     42000 | 2019-11-10 17:30:38 |
+-------+-------------+----------+-----------+---------------------+
5 rows in set (0.00 sec)
```

INSERT VALUES TO TABLE GAMES

INSERT INTO GAMES VALUES(“GM001”,”CRICKET”,10000);

INSERT INTO GAMES VALUES(“GM002”,”TENNIS”,9000);

INSERT INTO GAMES VALUES(“GM003”,”BASKETBALL”,6000);

INSERT INTO GAMES VALUES(“GM004”,”FOOTBALL”,9000);

INSERT INTO GAMES VALUES(“GM005”,”BADMINTON”,8000);

```
mysql> select * from GAMES;
+----------+-------------+----------+
| GAME_ID | GAME_NAME   | GAME_FEE |
+----------+-------------+----------+
| GM001    | CRICKET     |    10000 |
| GM002    | TENNIS      |     9000 |
| GM003    | BASKETBALL  |     6000 |
| GM004    | FOOTBALL    |     9000 |
| GM005    | BADMINTON   |     8000 |
+----------+-------------+----------+
5 rows in set (0.00 sec)
```

INSERT VALUES TO TOURNAMENT

INSERT INTO TOURNAMENT VALUES(“T501”,”1500”,”M101”);

INSERT INTO TOURNAMENT VALUES(“T502”,”1000”,”M102”);

INSERT INTO TOURNAMENT VALUES(“T503”,”2500”,”M103”);

INSERT INTO TOURNAMENT VALUES(“T504”,”2200”,”M104”);

INSERT INTO TOURNAMENT VALUES(“T505”,”3000”,”M105”);

```
mysql> select * from tournament;
+---------+-------+----------+
| TOUR_ID | PRIZE | MEMBER_ID |
+---------+-------+----------+
| T501    |  1500 | M101     |
| T502    |  1000 | M102     |
| T503    |  2500 | M103     |
| T504    |  2200 | M104     |
| T505    |  3000 | M105     |
+---------+-------+----------+
5 rows in set (0.00 sec)
```

INSERT VALUES TO TOURNAMENT_LOC

INSERT INTO TOURNAMENT_LOC VALUES(“LC01”,”COMPLEX A”,”BANGALORE”,”T501”);

INSERT INTO TOURNAMENT_LOC VALUES(“LC02”,”COMPLEX B”,”MANGALORE”,”T502”);

INSERT INTO TOURNAMENT_LOC VALUES(“LC03”,”COMPLEX C”,”DELHI”,”T503”);

INSERT INTO TOURNAMENT_LOC VALUES(“LC04”,”COMPLEX D”,”MUMBAI”,”T504”);

INSERT INTO TOURNAMENT_LOC VALUES(“LC05”,”COMPLEX E”,”CHENNAI”,”T505”);

```
mysql> select * from tournament_loc;
+--------+-----------+-----------+---------+
| LOC_ID | LOC_NAME  | LOC_AREA  | TOUR_ID |
+--------+-----------+-----------+---------+
| LC01   | COMPLEX A | BANGALORE | T501    |
| LC02   | COMPLEX B | MANGALORE | T502    |
| LC03   | COMPLEX C | DELHI     | T503    |
| LC04   | COMPLEX D | MUMBAI    | T504    |
| LC05   | COMPLEX E | CHENNAI   | T505    |
+--------+-----------+-----------+---------+
5 rows in set (0.00 sec)
```

INSERT VALUES TO CONTACT

INSERT INTO CONTACT VALUES(“9656896542”,“M101”);

INSERT INTO CONTACT VALUES(“9959965632”,“M101”);

INSERT INTO CONTACT VALUES(“9886785489”,“M101”);

INSERT INTO CONTACT VALUES(“7895648973”,“M101”);

INSERT INTO CONTACT VALUES(“8974623148”,“M101”);

```
mysql> select * from contact;
+------------+-----------+
| PHONE      | MEMBER_ID |
+------------+-----------+
| 9656896542 | M101      |
| 9959965632 | M101      |
| 9886785489 | M101      |
| 7895648973 | M101      |
| 8974623148 | M101      |
+------------+-----------+
5 rows in set (0.00 sec)
```

4.7 CREATION OF TRIGGERS

The trigger is made such that when a new record is inserted into a CD table, it automatically inserts the current date and time when the record is inserted.

CREATE TRIGGER ‘timestamp’

BEFORE INSERT ON ‘instructor’

FOR EACH ROW set new.ENTRY_TIME=now()

```
mysql> create trigger time_stamp before insert on instructor for each row set new.ENTRY_TIME=now();
Query OK, 0 rows affected (0.12 sec)
```

```
mysql> select * from instructor;
+--------+--------------+----------+------------+---------------------+
| INS_ID | INS_NAME     | COURSE_ID | INS_SALARY | Entry_Time          |
+--------+--------------+----------+------------+---------------------+
| INS003 | ARUN LAL     | 03       |      32000 | 2019-11-10 17:30:06 |
| INS002 | RAMESH VERMA | 02       |      40000 | 2019-11-10 17:29:09 |
| INS001 | SUNIL THAKUR | 01       |      45000 | 2019-11-10 17:27:43 |
| INS004 | ASHOK KUMAR  | 04       |      38000 | 2019-11-10 17:30:06 |
| INS005 | MANISH SINGH | 05       |      42000 | 2019-11-10 17:30:38 |
+--------+--------------+----------+------------+---------------------+
5 rows in set (0.00 sec)
```

4.8 CREATION OF STORED PROCEDURES

STORED PROCEDURE ON MEMBER TABLE TO FIND THE AGE

```
DELIMITER $$
CREATE PROCEDURE GetAge()
BEGIN
Select *,year(curdate())-year(Member_DOB)as AGE from MEMBER;
END$$
```

```
mysql> DELIMITER $$
mysql> CREATE PROCEDURE GetAge()
    -> BEGIN
    -> Select *,year(curdate())-year(Member_DOB)as AGE from MEMBER;
    -> END $$
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> call GetAge();
+-----------+------------------+------------+------------+------+
| MEMBER_ID | MEMBER_NAME      | PHONE      | MEMBER_DOB | AGE  |
+-----------+------------------+------------+------------+------+
| M101      | ABHINAV SINGH    | 9656896542 | 1974-03-27 |   45 |
| M102      | HARSHVARDHAN RAJ | 9959965632 | 1975-11-12 |   44 |
| M103      | RAJESH RANJAN    | 9886785489 | 1972-05-19 |   47 |
| M104      | NAVEEN BHARTI    | 7895648973 | 1981-01-21 |   38 |
| M105      | AJAY CHOUDHARY   | 8974623148 | 1980-04-22 |   39 |
+-----------+------------------+------------+------------+------+
5 rows in set (0.00 sec)

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

CHAPTER 5

FRONT END DESIGN

5.1 SYSTEM DESIGN

System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. System design could see it as the application of systems theory to product development. There is some overlap with the disciplines of system analysis, system architecture and systems engineering. If the border topic of product development "blends the perspective of marketing, design, and manufacturing into a single approach to product development," then design is the act of taking the marketing information and creating the design of the product to be manufactured. System design is therefore the process of defining and developing systems to satisfy specified requirements of the user.

Until the 1990's systems design had a crucial and respected role in the data processing industry. In 1990's standardization of hardware and software resulted in the ability to build modular systems. The increasing importance of software running on generic platforms has enhanced the discipline of software engineering.

Object-oriented analysis and design methods are becoming the most widely used methods for computer system design. The UML has become the standard language in object-oriented analysis and design. It is widely used for modelling software systems and organizations.

System design is one of the most important phases of software development process. The purpose of the design is to plan the solution of a problem specified by the requirement documentation. In other words, the first step in the solution to the problem is the design of the project.

5.2 FRONT END CODE

5.2.1 CREATING FRONT END PAGE TO LINK ALL TABLES

```
<!DOCTYPE html>
<html>
<head>
  <title>
```

```
            SPORTS ACADEMY
  </title>
</head>
<body>  <h1>
     <link href='https://fontsgoogleapis.com/css?family=Cinzel Decorative'
rel='stylesheet'>
   <h2><font style="text-anchor:" color="white" face="Cinzel Decorative" size="55">
  <u>SPORTS ACADEMY DATABASE MANAGEMENT SYSTEM</font></h2>

<style> body{ background:
url("https://ak9.picdn.net/shutterstock/videos/473089/thumb/1.jpg") no-repeat;
background-size:cover;
font-family:Cinzel Decorative;
color:white;
}
ul{
     margin:3pc;
     padding:0px;
     List-style:none;
}

ul l li{
float:left;
width:200px;
height:60px;
 background-color:mediumslateblue;
opacity:.8;
line-height:60px;
text-align:center;
font-size:25px;
```

```
margin-right:40px;
}

ul li a{
    text-decoration:underline;
    text-align: center;
    color:white;
   display:block;
}

ul  li a:hover{
     background-color:mediumseagreen;
}

ul   li ul li{
      display:none;
}
ul li:hover ul li{
   display:block;
  }

</style><br>
<ul>
   <li><a href="member.html">MEMBER</a></li>
   <li><a href="courses.html">COURSES</a></li>
   <li><a href="games.html">GAMES</a></li>
   <li><a href="tournament.html">TOURNAMENT</a></li>
</ul>
<br>
</h1>
<style>
```

```
*{
   box-sizing:border-box;
}
.column{
float:left;
width:25%;
Padding: 5px;
}

/* Clearfix(clear floats)*/
.row:: after{

      content: “”;
      Clear:both;
      Display:table;
}
</style>
</head>
<body>

<div class=”row”>
<div class=”column”>
 <img src=”https://encrypted-
tbn0.gstatic.com/images?q=tbn:ANd9GcQBZtEVBKDfihnxpNwwu6W1yQWT0197g
7IbDqERtLoQt4BtFRuh0g” alt=”Snow” style=”width:100% height:100%”>

-
</div>
<div class=”column”>
 <img src=”https://images.all-free-
download.com/images/graphivlarge/basketball_02_hd_pictures_168213.jpg”
alt=”Forest” style=”width:80% height:100%”>
</div>
```

```
 <div class="column">
 <img src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQV-
zZsmBRzkBMfCI4OW-t7B3i9TkuzGW7z7_F1KaALJA4hPzgjPQ" alt="Mountains"
style="width:100% height:100%>
</div>
 <div class="row">
<div class="column">
 <img src="https://i1.wp.com/www.healthfitnessrevolution.com/wp-
content/uploads/2015/04/ThinkstockPhotos-513920615.jpg?fit=725%2C483"
alt="Snow" style="width:80% height:100%">
</div>
</div>
 </body>
 </html>
```

FRONT END CODE FOR INSERTION

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
Body {
 font-family: Arial, Helvetica, sans-serif;
 background-color: black;
}

*{
 box-sizing: border-box;
}
/*Add padding to container*/
.container {
        padding: 16px;
        background-color: white;
```

```
}
/*Full-width input fields*/

input[type=text],input[type=password] {
 width: 100%;
 padding: 15px;
 margin: 5px 0 22px 0;
 display: inline-block;
 border: none;
 background: #f1f1f0;
}

input[type=text]:focus, input[type=password]:focus {
 background-color: #ffffff;
 outline: none;
}
/*Overwrite default style of hr*/

hr{
 border: 1px solid #f1f1f1;
 margin-bottom: 25px;
}
/*Set a style for the submit button*/
.registerbtn {
 background-color: #4CAF50;
 color: white;
 padding: 16px 20px;
          margin: 8px 0;
border: none;
cursor: pointer;
width: 100%;
```

```
opacity: 0.6;
}

.registerbtn:hover {
 opacity: 1;
}
/*Add a blue text color to links*/
a {
        color: dodgerblue;
}

/*Set a grey background color and center the text of the “sign in” section*/

.signin {
 background-color: #ffffff;
 text-align: center;
}
</style>
</head>
<body>

<from ACTION=”minsert.php”METHOD=”POST”>
<div class=”container”>
<center><h1 style=”color:#3366cc” size=”30”;>ENTER THE MEMBER
DETAILS</h1></center>

<hr>

<label for=”M_ID”><b>Member ID</b></label>
<input type=”text” placeholder=”Enter MEMBER_ID” name=”M_ID” required>
```

```
<label for="M_NAME"><b>Member NAME</b></label>
<input type="text" placeholder="Enter Member Name" name="M_NAME" required>
<label for="M_PHONE"><b>Member Phone</b></label>
<input type="text" placeholder="Enter Member Phone" name="M_PHONE"
required>
<label for="M_DOB"><b>Member DOB</b></label>
<input type="data" placeholder="Enter Member DOB" name="M_DOB" required>
<hr>
<button type="submit" class="registerbtn">SUBMIT</button>
<button type="reset" class="registerbtn">RESET</button>
</div>
</form>
</body>
</html>
```

FRONT END CODE FOR SEARCH

```
<!DOCTYPE html>
<html>
 <head>
        <title>member search</title>
        <style>
body {
 background-image: url("http://letip.com/wp-
content/uploads/2018/09/FindMember.png");
 background-size:cover;
 background-repeat: no-repeat;}

input[type=text]{
        width: 150px;
        box-sizing: border-box;
        font-size: 40px;
}
```

```
</style>
 <meta charset=”UTF-8”>
 <meta name=”viewport” content=”width=device-width, initial-scale=1.0”>

</head>
<body>
<center><u><font face=”Georgia Bold” Color=”reddishbrown” size=20>ENTER
THE MEMBER ID TO BE SEARCHED</font></h1><br><br><br><br><br><br>

</center><br><br><br><font face =”Times New Roman” size=10 color=”orange”>

<form action=”msearch1.php” method=”post”>
 <center><b>MEMBER_ID:<input type=”text” name=”MEMBER_ID”><br><br>
 <input type=”submit” name=”search” value=”Find”>
</form>
</body>
</html>
```

FRONT END CODE FOR DELETE

```
<html>
<title> member delete </TITLE>
<style>
body {
   background-image:url(“        -------------link-----------------”);
background -size :contain;
    Background-repeat: no-repeat;}
</style>
<body>
<center><u><font face=”Georgia Bold” Color=”reddishbrown” size=20> ENTER
THE MEMBER ID TO BE DELETED </font></h1><br><br><br>
</center><br><br><br><font face=”Times New Roman” size=5 color=”black”>
```

```
<Form Action=”mdel.php” METHOD=”POST”>
<center><b>Member ID:<INPUT TYPE=”TEXT” NAME=”t1”><br<br>
<input type=”Submit” name=”submit”/><br></FONT></form>
</body>
</html>
```

5.3 CONNECTIVITY TO DATABASE

CONNECTING FRONT END OF INSERT TO BACK END

```
<?php

if(isset($_POST[‘M_ID’])&&isset($_POST[‘M_NAME’])&&
isset($_POST[‘M_PHONE’])&& isset($_POST[’M_DOB’]));

$MEMBER_ID = $_POST[‘M_ID’];
$MEMBER_NAME = $_POST[‘M_NAME’];
$MEMBER_PHONE = $_POST[‘M_PHONE’];
$MEMBER_DOB = $_POST[‘M_DOB’];

$link = new mysqli(‘localhost’,’root’,’’,’sports_academy’);

if($link->connect_error)

die(‘connection error: ’.$link->connect_error);

$sql3 = “INSERT INTO MEMBER(MEMBER_ID, MEMBER_NAME,
        MEMBER_PHONE, MEMBER_DOB)
VALUES(‘’’.$MEMBER_ID.’’’,’’’.$MEMBER_NAME’’’,’’’.$MEMBER_PHONE’’
’,’’’.$MEMBER_DOB’’’))”;

        $result = $link->query($sql3);
```

```
            if($result > 0):
             echo 'Successfully posted';
            else:
             echo 'Unable to post';
 endif;
 $link->close();

die();
endif;

?>
```

CONNECTING FRONT END OF SEARCH TO BACK END

```
<?php
$host="localhost";
$user="root";
$password="";
$con=new mysqli($host,$password,"sports_academy");
if($con->connect_error) {
                    die("Connection failed:".$con->connect_error);
                     }
if($_SERVER["REQUEST_METHOD"] == "POST")
{
 $mid=$_POST['MEMBER_ID'];
echo"
                     <h1>User details of $mid is:<h1><br/><br/>";
$sql="select* from member where MEMBER_ID like '%$mid%';
                     $result = $con->query($sql);
                 if($result->num_rows>0){
            echo
"<table><tr><th>MEMBER_ID</th><th>MEMBER_NAME</th><th>MEMBER_P
HONE</th><th>MEMBER_DOB</th><tr>";
```

```
                    while($row=$result->fetch_assoc()){
  echo"<tr><td>"
.$row["MEMBER_ID"]."</td><td>".$row["Member_DOB"]."</td></tr>
<br>";
      }
      echo "</table>";
      echo "<button class=btn_Continue><a href='sign_in.html'>Try again</a>";
      }
      }
      $con->close();
?>
```

CONNECTING FRONT END OF DELETE TO BACK END

```
<?php
$host="localhost";
$user="root";
$password="";
$con=new mysqli($host.$user,$password,"sports_academy");
Name
if($_SERVER["REQUEST_METHOD"] == "POST")
{
 $a=$_POST['t1'];
 if($a!"")
 {
        $sql1 = "select * from MEMBER where MEMBER_ID='$a'";
        $result = mysqli_query($con,$sql1);
        if(mysqli_num_rows($result)>0){
       $sql3 = "delete from MEMBER where MEMBER_ID = '$a'";
       mysqli_query($con,$sql3);
       echo "MEMBER Deleted Successfully";
```

```
        }else{
        echo "$a does not Exist!";
        }
        }else{
        echo "MEMBER_ID Field is Empty";
        }

        $con->close();

}

?>
```

CHAPTER 6

TESTING

This chapter gives the outline of all testing methods that are carried out to get a bug free system. Quality can be achieved by testing the product using different techniques at different phases of the project development. The purpose of testing is to discover error. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components sub-assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.1 TESTING PROCESS: -

Testing is an integral part of software development. Testing process certifies whether the product that is developed complies with the standards that I was designed to. Testing process involves building of test cases against which the product has to be used.

6.2 TESTING OBJECTIVE: -

The main objectives of testing process are as follows.

1. Testing is a process of executing a program with the intent of finding an error.
2. A good test case is one that has high probability of finding undiscovered error.
3. A successful test is one that uncovers the undiscovered error.

6.3 TEST CASE

The test cases provided here test the most important features of the project.

Test cases for the project

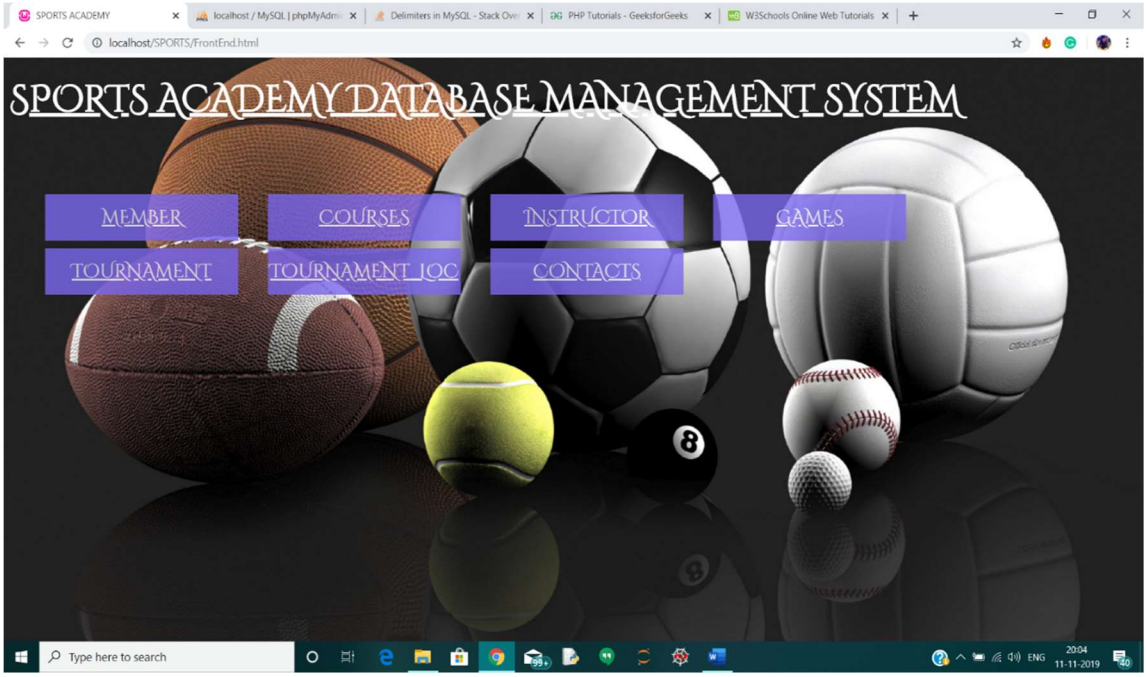
SNO	TEST INPUT	EXPECTED RESULT	OBSERVED RESULT	REMARKS
1	INSERT A RECORD	New tuple should be inserted	Query OK 1 row effected or inserted	PASS
2	SEARCH A RECORD	Display the record	Required record displayed	PASS
3	DISPLAY RECORD	Display the record	record displayed	PASS
4	DELETE A RECORD	Delete the record	Query OK 1 row affected or Row Deleted	PASS
5	CREATE TRIGGER	Trigger Created	Query OK Trigger Created	PASS
6	CREATE STORED PROCEDURES	Stored procedures created	Query OK Stored Procedures Created	PASS
7.	INSERT A RECORD	New tuple should be inserted	Failed to insert	FAIL

CHAPTER 7

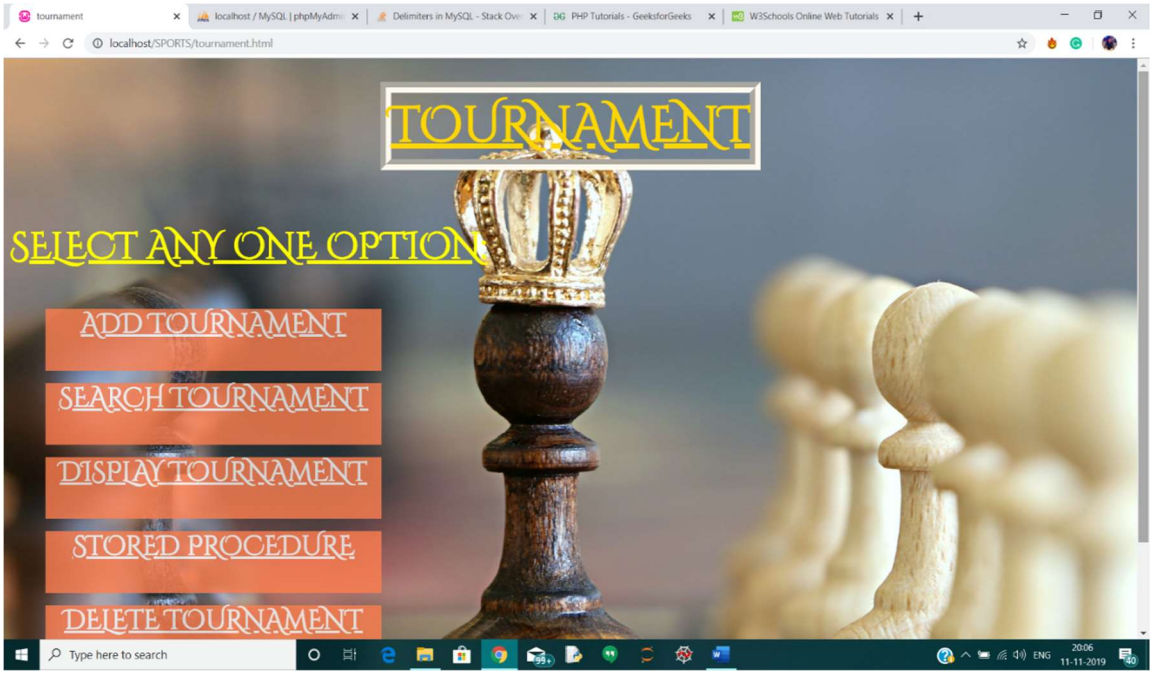
RESULTS

This section describes the screens of "Sports Academy Database". The snapshots are shown below for each module.

SNAPSHOTS

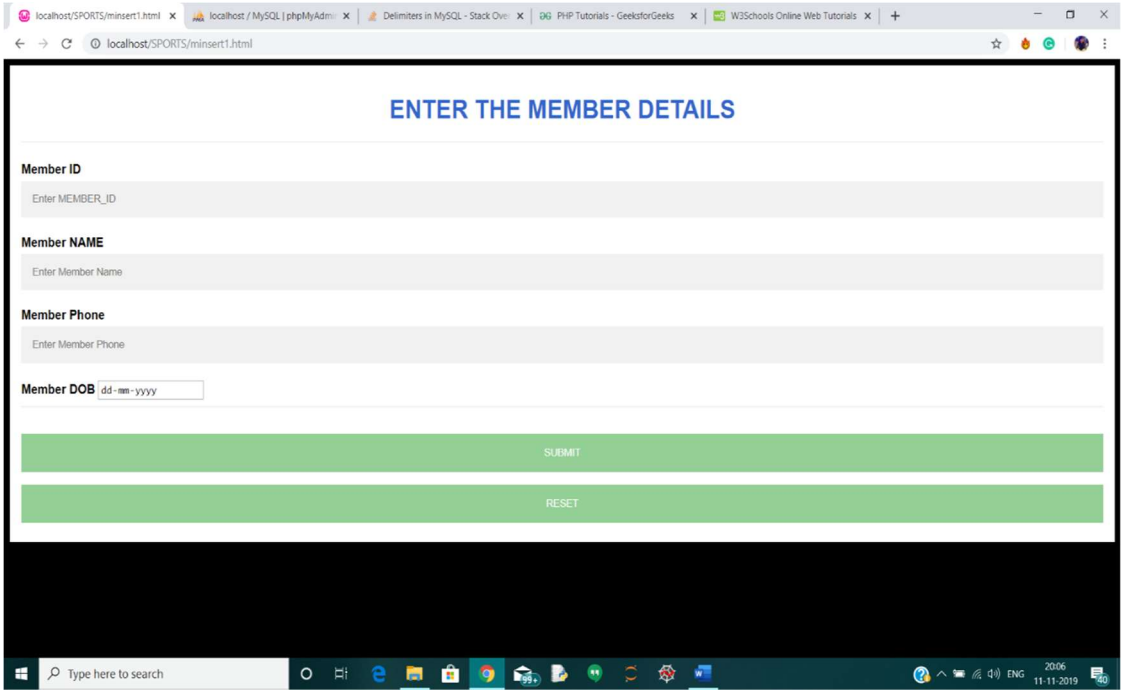


SNAPSHOT 7.1 SPORTS ACADEMY DATABASE MAIN PAGE



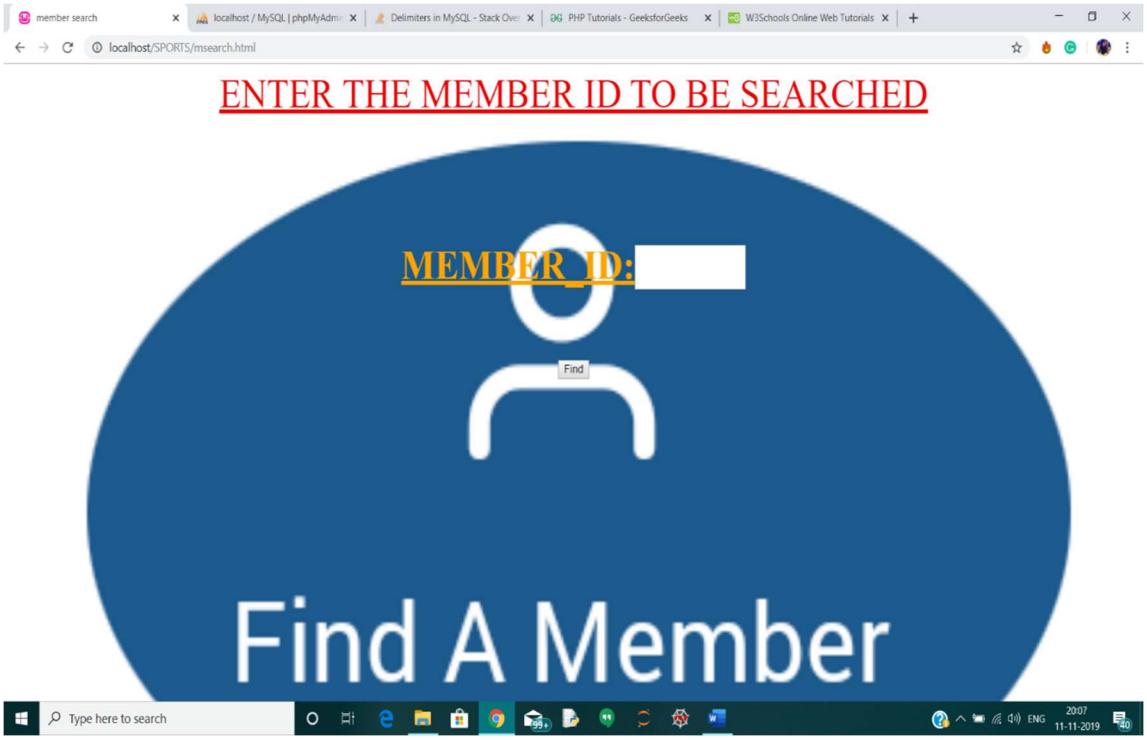
SNAPSHOT 7.2: TOURNAMENT ENTITY MAIN PAGE

This is the insertion page where we can insert in the front end which gets stored in back-end.



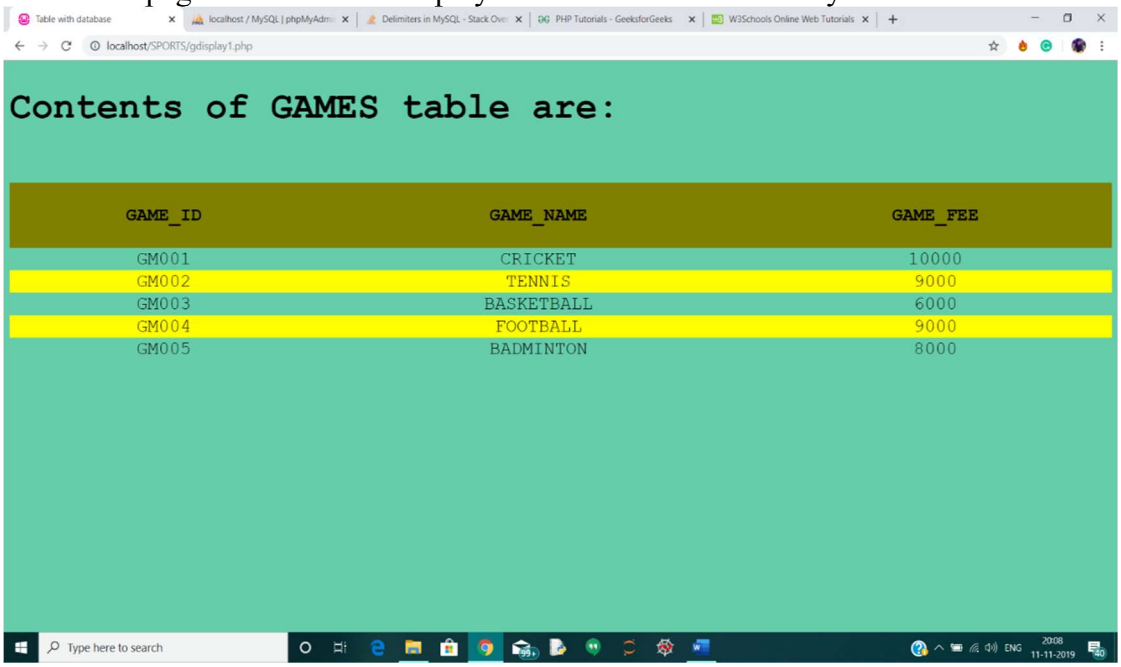
SNAPSHOT 7.3: INSERTION PAGE

This is the page where we can search for the details from front-end.



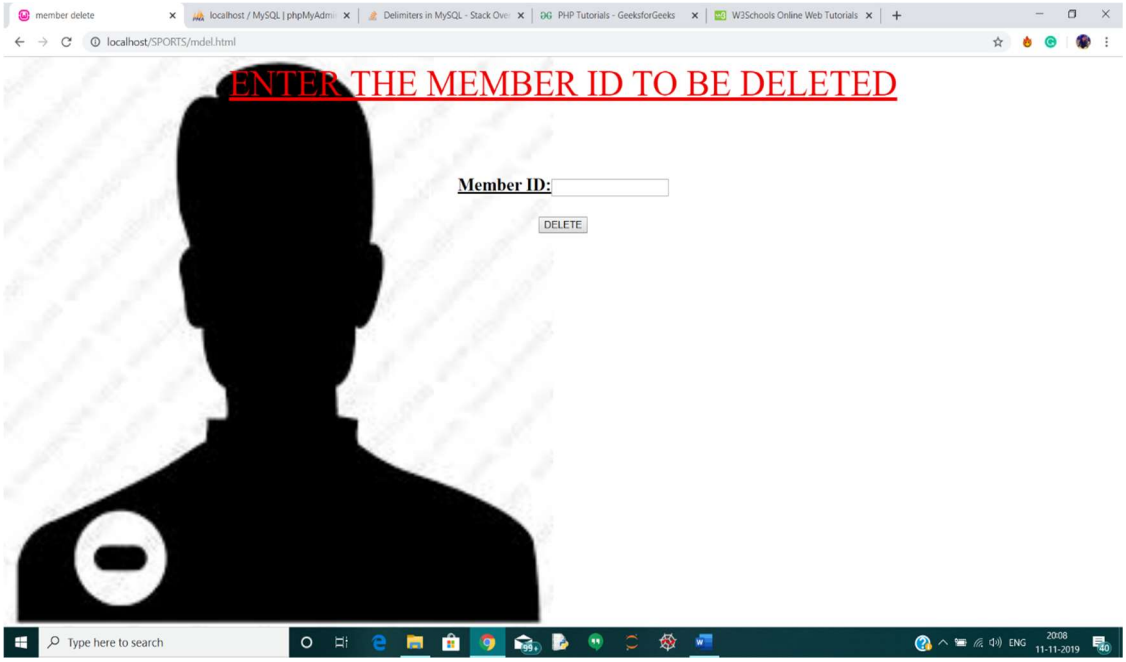
SNAPSHOT 7.4: SEARCH PAGE

This is the page where we can display all the records of the entity.



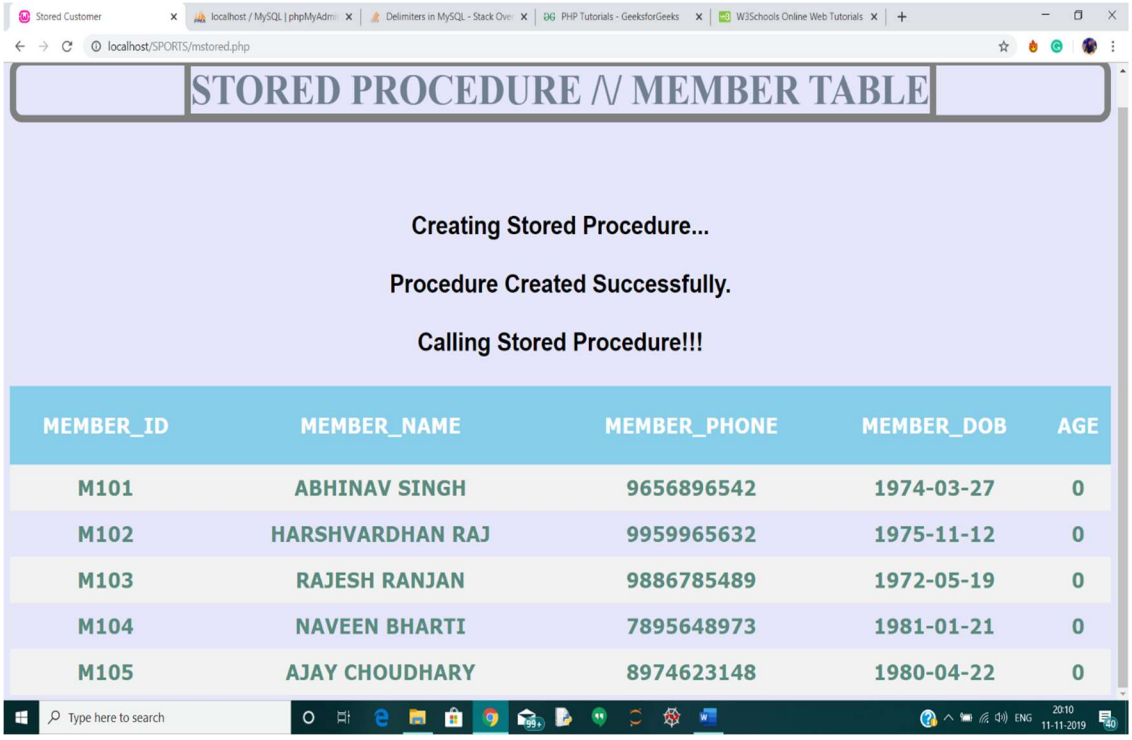
SNAPSHOT 7.5: DISPLAY PAGE

This is the page where we can delete the record from front-end which gets reflected in the back-end.



SNAPSHOT 7.6: DELETION PAGE

This is the page which displays the stored procedure at the front-end



SNAPSHOT 7.7: STORED PROCEDURE FRONT END PAGE

CONCLUSION

This database is a far more efficient mechanism to store and organize data than spreadsheets; it allows for a centralized facility that can easily be modified and quickly shared among multiple users. Having a web based front end removes the requirement of users having to understand and use a database directly, and allows users to connect from anywhere with an internet connection and a basic web browser. It also allows the possibility of queries to obtain information for various surveys

This website provides a computerized version of sports academy management system which will benefit the staff of the sports academy.

REFERENCES

I. Fundamentals of Database System, 7th Edition
-By Elmasri Ramez and Navathe Shamkanth

II. PHP and MySQL Web Development
-By Luke Welling and Laura Thompson

III. HTML & CSS: Design and Build Web Sites
-By John Duckett

IV. For Front End Code and CSS styling
a. https://www.w3schools.com/html
b. https://www.stackoverflow.com
c. https://www.tutorialspoint.com

V. For MySQL references
a. https://www.youtube.com
b. https://www.udemy.com