MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
A Constituent Institution of Manipal University

Recreational Activity Registration System (R.A.R.S.)

Presented by,

Pradyot S Nadig
(190948012)
MTech CSIS

Mohammad Shagil Siddiqui
(190948019)
MTech CSIS

Course Title: Information Systems Lab II

Course Code: CSE 5262

TABLE OF CONTENTS

Chapter 1

INTRODUCTION:

Like culture and art, recreation, leisure and sports activities play an important role in communities. Their many benefits include improving the health and well-being of individuals, contributing to the empowerment of individuals, and promoting the development of inclusive communities. Recreation, leisure and sports activities may involve individuals, small groups, teams or whole communities and are relevant to people of all different ages, abilities and levels of skill. The types of recreation, leisure and sports activities people participate in vary greatly depending on local context, and tend to reflect the social systems and cultural values.

The aim is to create a backend of a database application which is capable of organizing and registering faculties of the university for such activities.

1.1 Problem Statement:

The Every Organization, whether big or small, has challenges to overcome and managing the every event. The Event registration Management System has different event needs, so we design a relational database schema which helps in developing interface for easy registering participants for an even. This is designed to assist in strategic planning and it will help to ensure that the organization is equipped with the right level of information and details for future goals.

1.2 Integrity Constraints:

In this project the following integrity constraints were used while creating tables of the database.

1) Users Table:
   a) userid: Primary key
   b) Username: unique
   c) Email: unique

2) Events Type Table:
   a) typeid: Primary Key
   b) Type_name: unique

3) Events Table:
   a) Eventid: Primary Key
   b) typeid: Foreign Key to Events Type(typeid).
   c) userid: Foreign Key to Users (userid)

4) Registration Table:
   a) Reg_id: Primary Key
   b) Eventid: Foreign Key to Events(eventid)
   c) Userid: Foreign Key to Users(userid)

5) Payment Table:
   a) Payment_id: Primary Key,
   b) Reg_id: Foreign Key to Register(reg_id)

Chapter 2
DATABASE DESIGN AND IMPLEMENTATION

2.1 Database Definition Language Queries:

```
create database project;
use project;

create table users(
userid varchar(20) primary key,
password varchar(20) not null,
username varchar(30) unique not null,
contact_num bigint not null,
email varchar(40) unique not null,
department varchar(20) not null,
family_members int not null
);

create table eventtypes(
typeid int primary key auto_increment,
type_name varchar(20) unique not null
);

create table events(
eventid varchar(20) primary key,
event_name varchar(30) not null,
typeid int not null ,
event_date_time datetime not null,
userid varchar(20) not null,
max_attendees int not null,
foreign key(typeid) references eventtypes(typeid),
foreign key(userid) references users(userid)
);

create table registration(
reg_id varchar(20) primary key,
eventid varchar(20) not null,
userid varchar(20) not null,
members_count int not null,
reg_time timestamp default current_timestamp,
foreign key (eventid) references events(eventid),
foreign key(userid) references users(userid)
);
create table payment(
payment_id int primary key auto_increment,
reg_id varchar(20)     not null,
fees numeric not null,
foreign key(reg_id) references registration(reg_id)
);
```

2.2 Relational Database Schema:

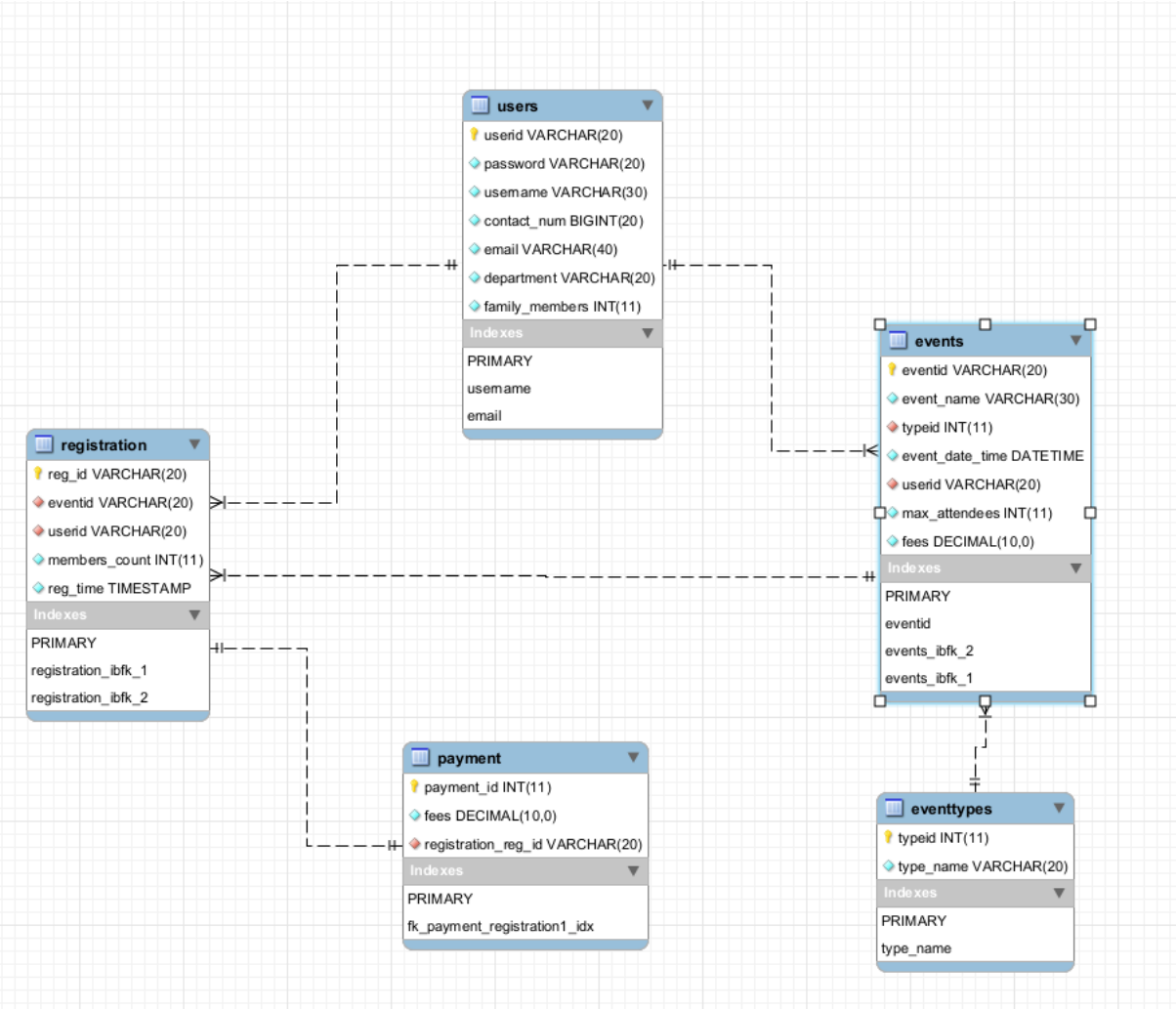

Fig. 1 Relational schema for Recreational Activity Registration System

2.3 Implementation:

Screenshots of the various tables present in the database-

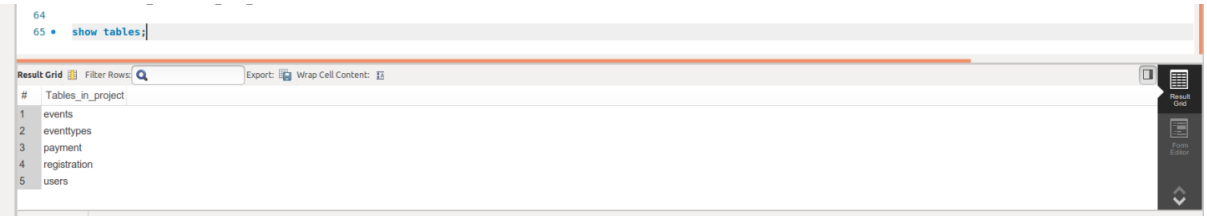

Fig 2. Tables present in the Database

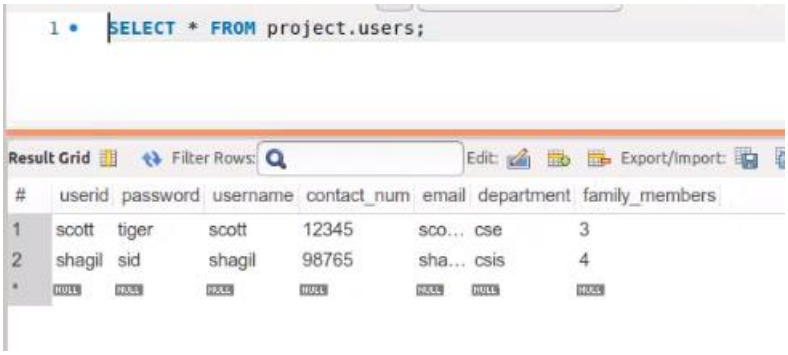

Fig 3. Users table

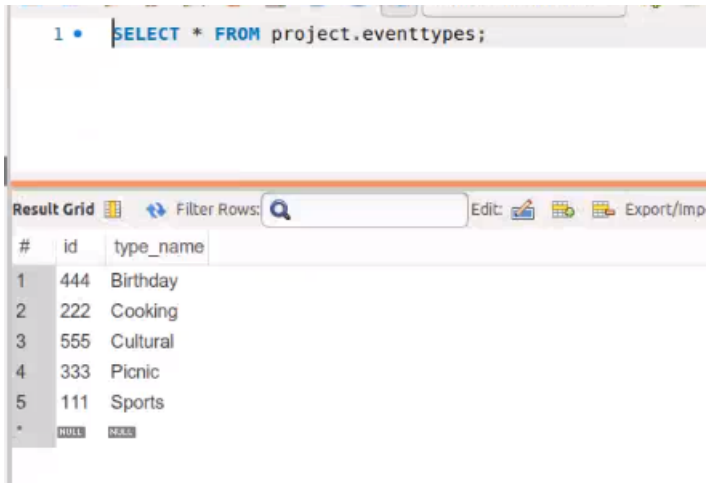

Fig 4. Event Types table

```
1 •  SELECT * FROM project.events;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

eventid	event_name	event_type	event_date_time	organiser_id	max_attendees	fees
01	Singing	555	2020-05-18 10...	scott	50	100
02	ShagilBirt...	444	2020-05-28 10...	shagil	30	200
03	Dancing	444	2020-05-29 12...	scott	40	100
04	Cricket	111	2020-05-30 08...	shagil	200	100
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Fig 5. Events table

```
1 •  SELECT * FROM project.registration;
```

Result Grid | Filter Rows: | Edit: | Export/In

#	reg_id	event_id	user_id	members_count	reg_time
1	90	01	scott	5	2020-05-18 04:40:10
2	91	02	shagil	7	2020-05-18 04:40:10
3	92	03	scott	4	2020-05-28 10:00:10
4	93	04	shagil	5	2020-05-28 10:00:10
*	NULL	NULL	NULL	NULL	NULL

Fig 6. Registration table

```
1 •  SELECT * FROM project.payment;
```

Result Grid | Filter Rows: | Edit:

#	payment_id	registration_id	fees
1	1	91	100
*	NULL	NULL	NULL

Fig 7. Payment table

2.4 Implementation of Basic and Complex Queries:

2.4.1 Basic Queries:

1) Display all the events in descending order of time.

select * from events order by event_date_time desc;

```
61 •  select * from events order by event_date_time desc;
62
```

Result Grid Filter Rows: Edit: Export/Import:

#	eventid	event_name	event_type	event_date_time	organiser_id	max_attendees	fees
1	04	Cricket	111	2020-05-30 08...	shagil	200	100
2	03	Dancing	444	2020-05-29 12...	scott	40	100
3	02	ShagilBirt...	444	2020-05-28 10...	shagil	30	200
4	01	Singing	555	2020-05-18 10...	scott	50	100
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

2) Display a user where username is scott and password is tiger.

select * from users where userid='scott' and password='tiger' ;

```
78 •  select * from users where userid='scott' and password='tiger' ;
79
80
```

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Cont

#	userid	password	username	contact_num	email	department	family_members
1	scott	tiger	scott	12345	sco...	cse	3
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

3) Display events where event fees is less than 100.

select event_name,event_date_time from events where fees <=100;

```
89 •  select event_name,event_date_time from events where fees <=100;
90
91
```

Result Grid Filter Rows: Export: Wrap Cell Content:

#	event_name	event_date_time
1	Singing	2020-05-18 10...
2	Dancing	2020-05-29 12...
3	Cricket	2020-05-30 08...

4) Display events where scott has registered into the events

select * from registration where user_id='scott';

```
103 •  select * from registration where user_id='scott';
104
105
106
107
108
109
110
```

Result Grid Filter Rows: Edit: Export/Import:

#	reg_id	event_id	user_id	members_count	reg_time
1	90	01	scott	5	2020-05-18 04:40:10
2	92	03	scott	4	2020-05-28 10:00:10
*	NULL	NULL	NULL	NULL	NULL

5) Delete an entry from registration table where registration id is 92

delete from registration where reg_id='92';
select * from registration;

```
68 •  delete from registration where reg_id='92';
69 •  select * from registration;
```

Result Grid Filter Rows: Edit: Export/Import:

#	reg_id	event_id	user_id	members_count	reg_time
1	90	01	scott	5	2020-05-18 04:40:10
2	91	02	shagil	7	2020-05-18 04:40:10
3	93	04	shagil	5	2020-05-28 10:00:10
*	NULL	NULL	NULL	NULL	NULL

2.4.2 Complex Queries:

1) Natural Join of Users table with events table to check who the organisers for which event are.

select * from events natural join users;

```
71 •  select * from events natural join users;
72
73
74
75
76
77
78
79
```

Result Grid Filter Rows: Export: Wrap Cell Content:

#	userid	eventid	event_name	typeid	event_date_time	max_attendees	fees	password	username	contact_num	email	department	family_members
1	scott	01	Singing	555	2020-05-18 10:10:10	50	100	tiger	scott	12345	sco…	cse	3
2	shagil	02	ShagilBirt…	444	2020-05-28 10:10:10	30	200	sid	shagil	98765	sha…	csis	4
3	scott	03	Dancing	444	2020-05-29 12:10:10	40	100	tiger	scott	12345	sco…	cse	3
4	shagil	04	Cricket	111	2020-05-30 08:00:00	200	100	sid	shagil	98765	sha…	csis	4

2) Natural join of events table with registration table to check what and how many registration have been done for particular event.

select * from events natural join registration;

```
90 •  select * from events natural join registration;
91
92
93
```

Result Grid Filter Rows: Export: Wrap Cell Content:

#	eventid	userid	event_name	typeid	event_date_time	max_attendees	fees	reg_id	members_count	reg_time
1	01	scott	Singing	555	2020-05-18 10:10:10	50	100	90	5	2020-05-18 04:40:10
2	02	shagil	ShagilBirt…	444	2020-05-28 10:10:10	30	200	91	7	2020-05-18 04:40:10
3	04	shagil	Cricket	111	2020-05-30 08:00:00	200	100	93	5	2020-05-28 10:00:10

3) Inner Join of events table with registration table

select * from events as e inner join registration as r on e.userid=r.userid;

```
100
101
102 •  select * from events as e inner join registration as r on e.userid=r.userid;
103
104
```

Result Grid Filter Rows: Export: Wrap Cell Content:

#	eventid	event_name	typeid	event_date_time	userid	max_attendees	fees	reg_id	eventid	userid	members_count	reg_time
1	01	Singing	555	2020-05-18 10:10:10	scott	50	100	90	01	scott	5	2020-05-18 04:40:10
2	02	ShagilBirt…	444	2020-05-28 10:10:10	shagil	30	200	91	02	shagil	7	2020-05-18 04:40:10
3	02	ShagilBirt…	444	2020-05-28 10:10:10	shagil	30	200	93	04	shagil	5	2020-05-28 10:00:10
4	03	Dancing	444	2020-05-29 12:10:10	scott	40	100	90	01	scott	5	2020-05-18 04:40:10
5	04	Cricket	111	2020-05-30 08:00:00	shagil	200	100	91	02	shagil	7	2020-05-18 04:40:10
6	04	Cricket	111	2020-05-30 08:00:00	shagil	200	100	93	04	shagil	5	2020-05-28 10:00:10

4) Left join of registration with payment to check who have paid.

select * from registration as r left join payment as p on r.reg_id=p.reg_id;

```
107 •  select * from registration as r left join payment as p on r.reg_id=p.reg_id;
108
109
```

Result Grid Filter Rows: Export: Wrap Cell Content:

#	reg_id	eventid	userid	members_count	reg_time	payment_id	reg_id	fees
1	91	02	shagil	7	2020-05-18 04:40:10	1	91	100
2	90	01	scott	5	2020-05-18 04:40:10	NULL	NULL	NULL
3	93	04	shagil	5	2020-05-28 10:00:10	NULL	NULL	NULL

5) Select events that start with 's'

select * from events where event_name like 'S%' ;

```
111
112 •  select * from events where event_name like 'S%';
113
114
115
```

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell

#	eventid	event_name	typeid	event_date_time	userid	max_attendees	fees
1	01	Singing	555	2020-05-18 10:10:10	scott	50	100
2	02	ShagilBirt...	444	2020-05-28 10:10:10	shagil	30	200
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

---------X----X----X-------