CHAPTER 4

IMPLEMENTATION

System implementation is the important stage of project when the theoretical design is tuned into practical system.

4.1 Creating Database Using MySQL

Queries used for creating database and different tables used in GMS are given below.

To create new database

mysql> create database dbmsproject;

To use newly created database

mysql> use dbmsproject;
Database changed

To create table BATCH

mysql> CREATE TABLE BATCH(Batch_id varchar(10) PRIMARY KEY, Start_time varchar(20), Finish_time varchar(20), Batch_type varchar(30));

To create table MEMBER

mysql> CREATE TABLE MEMBER(Full_name varchar(20), Member_id varchar(10) PRIMARY KEY, Batch_id varchar(10), Gender varchar(10), Age int(11), Contact_no varchar(15), Email_id varchar(30), Address varchar(50), Weight decimal(10,2), Height decimal(10,2), Constraint fk_bid FOREIGN KEY(Batch_id) references BATCH(Batch_id) on delete cascade);

To create table TRAINER

mysql> CREATE TABLE TRAINER(Trainer_name varchar(20), Trainer_id varchar(10) PRIMARY KEY, Batch_id varchar(10), Gender varchar(10), Age int(11), Contact_no varchar(15), Email_id varchar(30), Address varchar(50), Constraint fk_batchid FOREIGN KEY(Batch_id) references BATCH(Batch_id) on delete cascade);

To create table PAYMENT

mysql> CREATE TABLE PAYMENT(Member_id varchar(10), Fees decimal(10,2), Date_of_Payment date, Due_Date date, Constraint fk_mid FOREIGN KEY(Member_id) references MEMBER(Member_id) on delete cascade);

To create table STAFF

mysql> CREATE TABLE STAFF(Staff_id varchar(10), Staff_name varchar(20), Password varchar(20));

To create view table GYM

mysql> CREATE VIEW GYM AS select M.Member_id, M.Full_name, B.Batch_id, T.Trainer_id, T.Trainer_name from MEMBER M, BATCH B, TRAINER T where M.Batch_id=B.Batch_id and B.Batch_id=T.Batch_id;

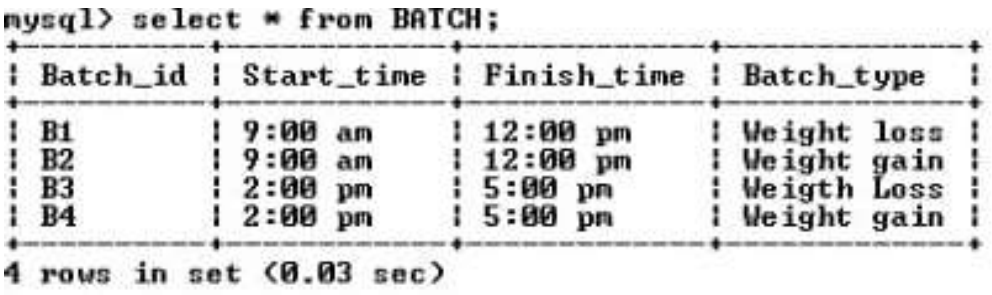
To insert values to BATCH table

mysql> insert into BATCH values(‘B1’,’9:00am’,’12:00pm’,’Weight loss’);

Query OK, 1 row affected (0.18 sec)

mysql> insert into BATCH values(‘B2’,’9:00am’,’12:00pm’,’Weight gain’);

To view values in table BATCH



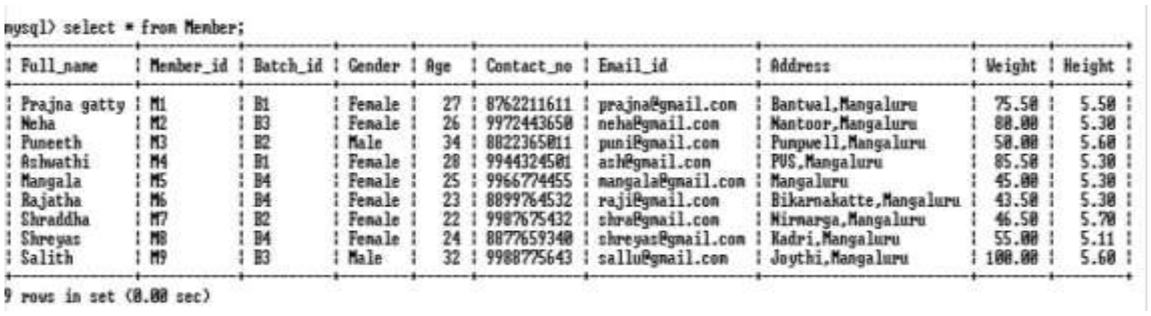
```
mysql> select * from BATCH;
+----------+------------+-------------+-------------+
| Batch_id | Start_time | Finish_time | Batch_type  |
+----------+------------+-------------+-------------+
| B1       | 9:00 am    | 12:00 pm    | Weight loss |
| B2       | 9:00 am    | 12:00 pm    | Weight gain |
| B3       | 2:00 pm    | 5:00 pm     | Weigth Loss |
| B4       | 2:00 pm    | 5:00 pm     | Weight gain |
+----------+------------+-------------+-------------+
4 rows in set (0.03 sec)
```

To insert values to MEMBER table

mysql>insert into MEMBER values (‘Prajnagatty’, ’M1’, ’B1’, ’Female’, 27,’8762211611’, ’prajna@gmail.com’, ’Bantwal,Mangaluru’ ,75.50, 5.50);

mysql>insert into MEMBER values (‘Neha’, ’M2’, ’B3’, ’Female’, 26,’9972443650’, ’neha@gmail.com’,’Nantoor,Mangaluru’,80.00,5.30);

To view values in table MEMBER



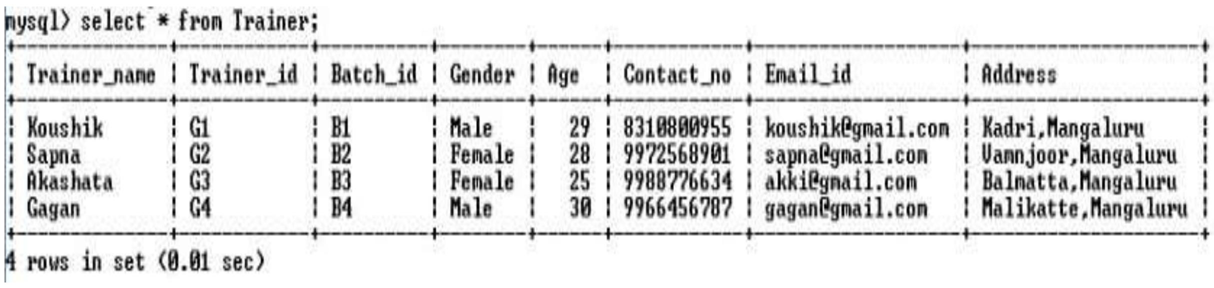
```
mysql> select * from Member;
+--------------+-----------+----------+--------+-----+------------+--------------------+-------------------------+--------+--------+
| Full_name    | Member_id | Batch_id | Gender | Age | Contact_no | Email_id           | Address                 | Weight | Height |
+--------------+-----------+----------+--------+-----+------------+--------------------+-------------------------+--------+--------+
| Prajna gatty | M1        | B1       | Female |  27 | 8762211611 | prajna@gmail.com   | Bantwal,Mangaluru       |  75.50 |   5.50 |
| Neha         | M2        | B3       | Female |  26 | 9972443650 | neha@gmail.com     | Nantoor,Mangaluru       |  80.00 |   5.30 |
| Puneeth      | M3        | B2       | Male   |  34 | 8822365011 | puni@gmail.com     | Pumpwell,Mangaluru      |  50.00 |   5.60 |
| Ashwathi     | M4        | B1       | Female |  28 | 9944324501 | ash@gmail.com      | PVS,Mangaluru           |  85.50 |   5.30 |
| Mangala      | M5        | B4       | Female |  25 | 9966774455 | mangala@gmail.com  | Mangaluru               |  45.00 |   5.30 |
| Rajatha      | M6        | B4       | Female |  23 | 8899764532 | raji@gmail.com     | Bikarnakatte,Mangaluru  |  43.50 |   5.30 |
| Shraddha     | M7        | B2       | Female |  22 | 9987675432 | shra@gmail.com     | Nirmarga,Mangaluru      |  46.50 |   5.70 |
| Shreyas      | M8        | B4       | Female |  24 | 8877659340 | shreyas@gmail.com  | Kadri,Mangaluru         |  55.00 |   5.11 |
| Salith       | M9        | B3       | Male   |  32 | 9988775643 | sallu@gmail.com    | Joythi,Mangaluru        | 100.00 |   5.60 |
+--------------+-----------+----------+--------+-----+------------+--------------------+-------------------------+--------+--------+
9 rows in set (0.00 sec)
```

To insert values to TRAINER table

mysql> insert into TRAINER values (‘Koushik’, ’G1’, ’B1’, ’Male’, 29,’8310800955’, ’koushik@gmail.com’,’Kadri,Mangaluru’);

mysql>insert into TRAINER values (‘Sapna’, ’G2’, ’B2’, ’Female’, 28,’9972568901’, ’sapna@gmail.com’, ’Vamanjoor,Mangaluru’);

To view values in table TRAINER



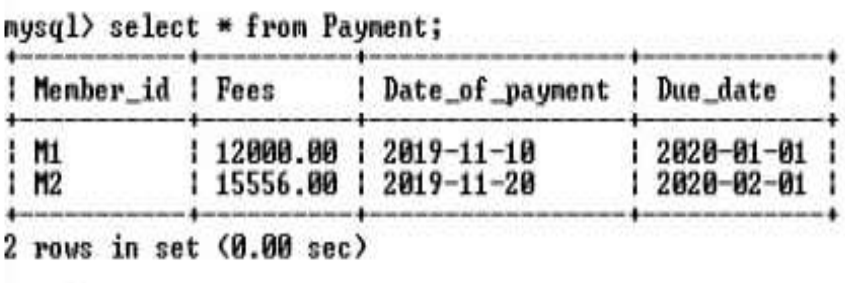
```
mysql> select * from Trainer;
+--------------+------------+----------+--------+-----+------------+-------------------+---------------------+
| Trainer_name | Trainer_id | Batch_id | Gender | Age | Contact_no | Email_id          | Address             |
+--------------+------------+----------+--------+-----+------------+-------------------+---------------------+
| Koushik      | G1         | B1       | Male   |  29 | 8310800955 | koushik@gmail.com | Kadri,Mangaluru     |
| Sapna        | G2         | B2       | Female |  28 | 9972568901 | sapna@gmail.com   | Vamnjoor,Mangaluru  |
| Akashata     | G3         | B3       | Female |  25 | 9988776634 | akki@gmail.com    | Balmatta,Mangaluru  |
| Gagan        | G4         | B4       | Male   |  30 | 9966456787 | gagan@gmail.com   | Malikatte,Mangaluru |
+--------------+------------+----------+--------+-----+------------+-------------------+---------------------+
4 rows in set (0.01 sec)
```

To insert values to PAYMENT table

mysql> insert into PAYMENT values(‘M1’,12000.00,’2019-11-10’,’2020-01-01’);

mysql> insert into PAYMENT values(‘M2’,15556.00,’2019-11-20’,’2020-02-01’);

To view values in table PAYMENT



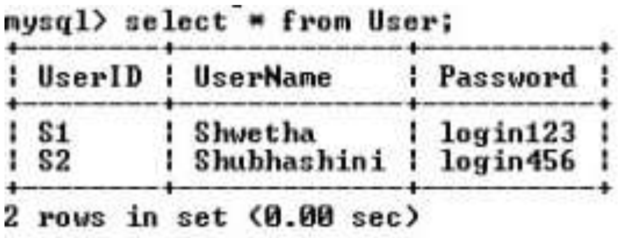
```
mysql> select * from Payment;
+-----------+----------+-----------------+------------+
| Member_id | Fees     | Date_of_payment | Due_date   |
+-----------+----------+-----------------+------------+
| M1        | 12000.00 | 2019-11-10      | 2020-01-01 |
| M2        | 15556.00 | 2019-11-20      | 2020-02-01 |
+-----------+----------+-----------------+------------+
2 rows in set (0.00 sec)
```

To insert values to STAFF table

mysql> insert into STAFF values(‘S1’,’Shwetha’,’login123’);

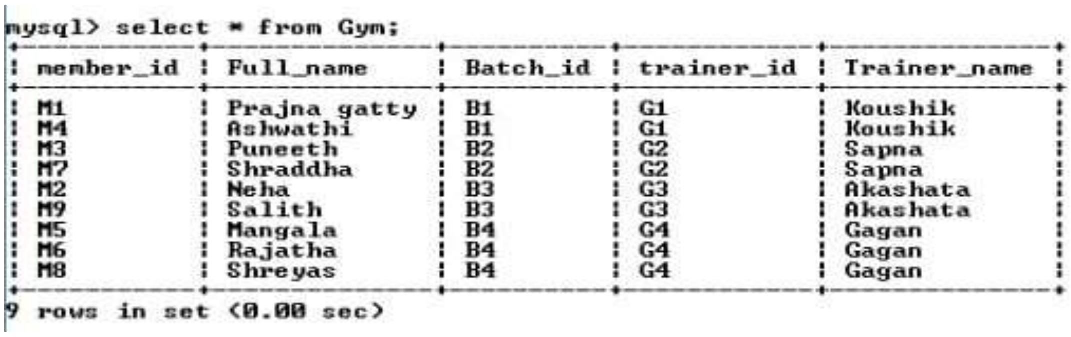
mysql> insert into STAFF values(‘S2’,’Shubhashini’,’login456’);

To view values in table STAFF



```
mysql> select * from User;
+--------+-------------+----------+
| UserID | UserName    | Password |
+--------+-------------+----------+
| S1     | Shwetha     | login123 |
| S2     | Shubhashini | login456 |
+--------+-------------+----------+
2 rows in set (0.00 sec)
```

To view values in table GYM



```
mysql> select * from Gym;
+-----------+--------------+----------+------------+--------------+
| member_id | Full_name    | Batch_id | trainer_id | Trainer_name |
+-----------+--------------+----------+------------+--------------+
| M1        | Prajna gatty | B1       | G1         | Koushik      |
| M4        | Ashwathi     | B1       | G1         | Koushik      |
| M3        | Puneeth      | B2       | G2         | Sapna        |
| M7        | Shraddha     | B2       | G2         | Sapna        |
| M2        | Neha         | B3       | G3         | Akashata     |
| M9        | Salith       | B3       | G3         | Akashata     |
| M5        | Mangala      | B4       | G4         | Gagan        |
| M6        | Rajatha      | B4       | G4         | Gagan        |
| M8        | Shreyas      | B4       | G4         | Gagan        |
+-----------+--------------+----------+------------+--------------+
9 rows in set (0.00 sec)
```

4.2 Stored Procedure

A stored procedure is a prepared SQL code that can be reused over and over again. So if an SQL query needs to be written over and over again, save it as a stored procedure, and then just call it to execute it. It is also possible to pass parameters to a stored procedure, so that the stored procedure can act based on the parameter value(s) that is passed.

Stored procedure used in GMS is as follows

use dbmsproject;

DELIMITER //

create procedure AmountPaid()

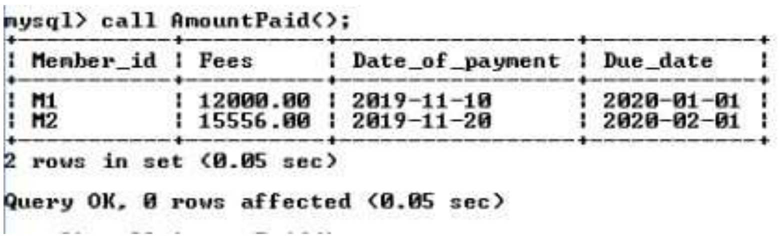
begin

select * from Payment;

end //

DELIMITER ;

To view the values in the stored procedure AMOUNTPAID



```
mysql> call AmountPaid();
+-----------+----------+-----------------+------------+
| Member_id | Fees     | Date_of_payment | Due_date   |
+-----------+----------+-----------------+------------+
| M1        | 12000.00 | 2019-11-10      | 2020-01-01 |
| M2        | 15556.00 | 2019-11-20      | 2020-02-01 |
+-----------+----------+-----------------+------------+
2 rows in set (0.05 sec)

Query OK, 0 rows affected (0.05 sec)
```

4.3 Triggers

A trigger is a special type of stored procedure that automatically executes when an event occurs in the database server. DML triggers execute when a user tries to modify data through a data manipulation language (DML) event. DML events are INSERT, UPDATE, or DELETE statements on a table or view. These triggers fire when any valid event is fired, regardless of whether or not any table rows are affected.

Trigger used in GMS is as follows

DELIMITER //

create trigger deltrig after delete on BATCH

for each row

begin

delete from MEMBER.Batch_id, TRAINER.Batch_id where Batch_id=MEMBER.Batch_id and Batch_id=TRAINER.Batch_id;

end //

DELIMITER ;

Trigger to automatically delete member and trainer information in MEMBER and TRAINER table when a row from BATCH table is deleted and it also deletes the member from the payments.