**Online Hostel-Mess**

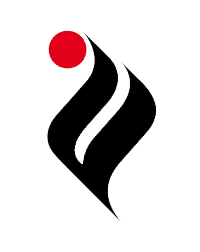
**Management System**

**DBMS Project Report**

***Submitted By:***

**Keshav Gupta (101603160)**

**Kartik Arora (401603016)**



**Computer Science Department**

**Thapar University,Patiala**

**CERTIFICATE**

**This is to certify that the project entitled “*Online Hostel-Mess Management*” embodies the original work done by Keshav Gupta & Kartik Arora under my supervision.**

**Thapar University, Patiala Mr. Parteek Bhatia**

**ACKNOWLEDGEMENT**

**Here we gladly present this project report on “ONLINE HOSTEL-MESS MANAGEMENT SYSTEM” as part of the 3rd semester DBMS project. At the time of submitting this report we use this opportunity to mention those people who were with us during the work. We take this occasion to thank God for blessing us with his grace and taking our endeavor to a successful culmination. We extend our sincere and heartfelt thanks to our esteemed guide, Mr. Parteek Bhatia for providing us the right guidance and advice at the crucial junctures and for showing us the right way.**

**Last but not the least; we would like to thank our friends for the support and encouragement they gave us during the course of our work.**

**Submitted by:**

**Keshav Gupta**

**Kartik Arora**

**(ii)**

**ABSTRACT**

**“ONLINE HOSTEL-MESS MANAGEMENT SYSTEM” is basically a Web-App developed for managing various activities in the hostel and mess. For the past few years the number of educational institutions is increasing rapidly. Thereby the number of hostels is also increasing for the accommodation of the students studying in these institutions. And hence there is a lot of strain on the persons who are running the hostel and softwares are not usually used in this context. This particular project deals with the problems on managing a hostel as well as mess and avoids the problems which occur when carried manually. Identification of the drawbacks of the existing system leads to the designing of such a computerized system that will be compatible to the existing system with the system which is more user friendly and more GUI oriented. We can improve the efficiency of the system, thus overcome the drawbacks of the existing system.**

**(iii)**

**TABLE OF CONTENTS**

**ACKNOWLEDGEMENT ii**

**ABSTRACT iii**

1. **INTRODUCTION 01**
   1. PROJECT OVERVIEW 01
   2. PROJECT OBJECTIVES 02
2. **SYSTEM ANALYSIS 03**
   1. EXISTING SYSTEM 03
   2. PROPOSED SYSTEM 04
3. **CASE STUDY OF PROJECT 05**
4. **FEASIBILITY STUDY 06**
   1. TECHNICAL FEASIBILITY 06
   2. ECONOMICAL FEASIBILITY 07
   3. OPERATIONAL FEASIBILITY 08
5. **REQUIREMENT ANALYSIS AND SPECIFICATIONS 09**
   1. HARDWARE REQUIREMENTS 09
   2. SOFTWARE REQUIREMENTS 10
6. **ER DIAGRAM 11**
7. **NORMALISATION OF TABLES 12**
8. **PL/SQL QUERIES FOR TABLES 19**
9. **CONCLUSION 45**
10. **REFERENCES 46**

***INTRODUCTION***

**PROJECT OVERVIEW**

**The online hostel-mess management system is web based application to provide hostel students certain online facilities which they can avail more efficiently. This project also keeps details of the hostellers. It is headed by Warden. He will be the administrator.**

**This document is intended to minimize human work and make hostel-mess management an easier job including mess calculation, complaint registration and notice board etc. Hostellers can view notice board, register complaints, make their leave entry for hostel as well as mess, can see their mess menu, monthly mess bill, total leaves and many other facilities simply by login into their respective online system.**

**(1)**

**PROJECT OBJECTIVES**

* **Students can make their leave entry in the leave form while going to their home.**
* **Students can also register complaints.**
* **Hostellers can check the status of their complaints.**
* **Admin can edit notice board and each student can view it.**
* **Mess secretary can calculate total monthly mess fee by considering mess fee and total leaves and can also edit mess menu.**
* **Hostellers can check the status of every month’s mess fee**
* **Hostellers can check their total no. of leaves in hostel.**

**(2)**

***SYSTEM ANALYSIS***

**EXISTING SYSTEM**

**The existing system is manual based and need lot of efforts and consume enough time. It may also lead to corruptions in the mess fee calculation process. The existing system does not deals with mess calculation and complaint registration more efficiently. Here hostellers have to make their leave entries on pen and paper which is further difficult to maintain such records.**

**DISADVANTAGES:**

**• More human power**

**• More strength and strain of manual labour needed**

**• Repetition of same procedure.**

**• Low Data security.**

**• Data redundancy is not there.**

**• Difficulty to handle.**

**• Difficulty to update data. Record keeping is difficult.**

**(3)**

**PROPOSED SYSTEM**

**The proposed system is having many advantages over the existing system. It require less overhead and very efficient. The proposed system deals with the mess fee calculation and complaint registration process and other processes more efficiently.**

**ADVANTAGES:**

* **Less manual labour required.**
* **More fast process than existing system.**
* **More transparency is present.**
* **Integrity of data is there.**
* **Data redundancy is more.**
* **More data security.**
* **Backup of data can be easily generated.**

**(4)**

**CASE STUDY OF THE PROJECT**

**In a university there are many hostels. Every student is associated with a particular hostel.**

**For every student there is an individual mess bill amount, complaint registration system, leave entry system. Every student can access these facilities by logging in into their respective system. Every student can make the leave entry while going home, register for their complaints, check status of their compliants, check their monthly mess bill and leaves etc.**

**(5)**

***FEASIBILITY STUDY***

**TECHNICAL STUDY**

**The technical feasibility in the proposed system deals with the technology used in the system. It deals with the hardware and software used in the system whether they are of latest technology or not. It happens that after a system is prepared a new technology arises and the user wants the system based on that technology.**

**This system use WINDOWS platform, PHP as front end technology and PL/SQL as backend technology. Thus ONLINE HOSTEL-MESS MANAGEMENT SYSTEM is technically feasible.**

**(6)**

**ECONOMICAL FEASIBILITY**

**Economic analysis is the most frequently used method for evaluating the effectiveness of a new system. More commonly known as cost/benefit analysis. PHP software and PL/SQL database is easily available on the internet.**

**OPERATIONAL FEASIBILITY**

**The project has been developed in such a way that it becomes very easy even for a person with little computer knowledge to operate it. This software is very user friendly and does not require any technical person to operate .Thus the project is even operationally feasible.**

**(7)**

**REQUIREMENT ANALYSIS AND SPECIFICATIONS**

**Functions and features delivered to the end users**

**The end users of the proposed system are:**

**Administrator module:**

**In administrator module administrator manages the master data’s like server details and student details. Accept the leave entry application form of students, view the application forms, view the complaints of the students in the hostel and delete from the database.**

**Student Module:**

**In student module, they can submit application form regarding leaves, change password, can check status,view monthly mess fee and submit the leave form when going home, register complaints.**

**Mess-Secretary Module:**

**In secretary module, the secretary can calculate the mess bill, and edit the mess menu, view the notice board and also inform the students about any event that is going to organized in mess say Hostel night, etc.**

**(8)**

**HARDWARE CONFIGURATION**

**The section of hardware configuration is an important task related to the software development insufficient random access memory may affect adversely on the speed and efficiency of the entire system. The process should be powerful to handle the entire operations. The hard disk should have sufficient capacity to store the file and application.**

**Processor: Pentium IV and above**

**Processor speed: 1.4 GHz Onwards**

**System memory: 128 Mb minimum 256 Mb recommended**

**Cache size: 512 KB**

**RAM: 512 MB (Minimum)**

**(9)**

**SOFTWARE CONFIGURATION**

**A major element in building a system is the section of compatible software since the software in the market is experiencing in geometric progression. Selected software should be acceptable by the firm and one user as well as it should be feasible for the system.**

**This document gives a detailed description of the software requirement specification. The study of requirement specification is focused specially on the functioning of the system. It allows the developer or analyst to understand the system, function to be carried out the performance level to be obtained and corresponding interfaces to be established.**

**Front end tool: PHP as scripting language**

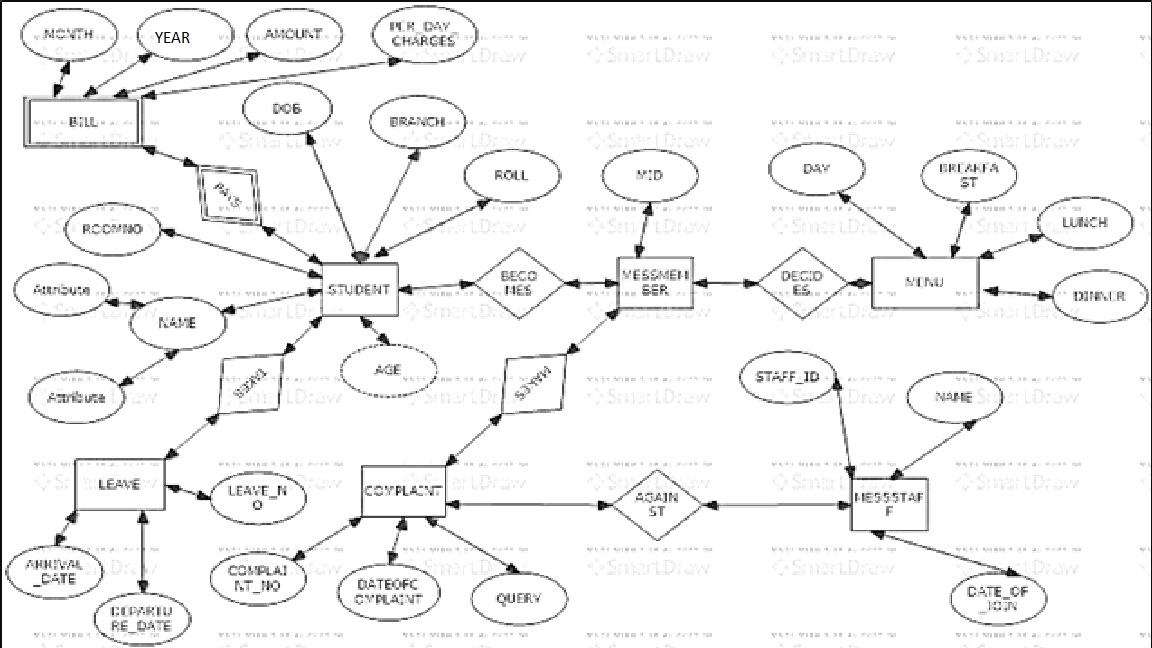
**Backend tool: ORACLE SQL 11g version for database**

**Operating system: Windows 2007/2010**

**Client Side: HTML, CSS**

**(10)**

**ER-DIAGRAM**

****

**(11)**

***NORMALISATION OF DATABASE TABLES***

* **STUDENT-TABLE**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ROLL** | | **DOB** | **F\_NAME** | **L\_NAME** | **BRANCH** | **ROOM\_NO** | **CITY** | **STATE** | **MOBILE** |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

**1ST Normal Form:**

Restricting the user to enter only single values for mobile number. It is found that it is already in Ist Normal form as for every entry of the above relation there exists atmost a single value.

**2nd Normal Form:**

Again by the definition of second normal form it is found that it is in second normal form as every non-key attribute is fully functional dependent over the primary key.

**3rd Normal Form:**

It is also in 3rd normal form.Hence it is now a normalized table.

**(12)**

* **Bill Table:**

ROLL

AMOUNT

MONTH

YEAR

PER DAY CHARGE

**From above FD diagram it is clear that it is already normalized according to 1st normal form, 2nd normal form as well as 3rd normal forms.**

**(13)**

* **Complaint Table:**

roll

dcomplaint

Complaint\_no

cquery

**From above diagram,**

**It is in 1st normal form as for every relation there exists atmost single value. And for 2nd normal form ,every non key attribute is fully dependent over the primary key. Again it is in 3rd normal due to the absence of transitive dependence.**

**(14)**

* **Leave Table:**

roll

ddate

l\_no

adate

**From above diagram,**

**Similarily ,It is in 1st normal form as for every relation there exists atmost single value. And for 2nd normal form ,every non key attribute is fully dependent over the primary key. Again it is in 3rd normal due to the absence of transitive dependence.**

**(15)**

* **Menu Table:**

**It is already in Normalized form here according to below FD Diagram:**

Dinner

Breakfast

Day

Lunch

**(16)**

* **Mess Staff Table:**

Name

Staffed

Doj(Date of Join)

**So from above diagram it is in 1st as well as 2nd and 3rd normal form already .Hence we can say that this is already a normalized table.**

**(17)**

* **Mess member Table:**

Roll

Mess id

* **Mess ID Table:**

Year

Mess id

**So from above diagrams it is in 1st as well as 2nd and 3rd normal form already .Hence we can say that this is already a normalized table.**

**(18)**

**PL/SQL QUERIES**

**Student Table:**

**Create table for student:**

create table student(roll varchar(20) primary key, dob date, fname varchar(20) not null, lname varchar(20) not null, branch varchar(20) not null, room\_no varchar(20) not null, city varchar(20) not null, state varchar(20) not null,mobile varchar(20) not null);

# Queries Regarding INSERT, UPDATE, DELETE, RETRIEVE On STUDENT TABLE

**Insert data into student table:**

declare

PROCEDURE insertstudent( rollno varchar,dob date ,fname varchar,lname varchar,branch varchar,room varchar,city varchar,state varchar,mobile varchar) AS

BEGIN

insert into student values(rollno,dob,fname,lname,branch,room,city,state,mobile);

dbms\_output.put\_line('RECORD INSERTED SUCCESSFULLY!!');

END;

begin

insertstudent(:Enter\_Roll\_No,:Enter\_Your\_Dob,:Father\_Name,:Mother\_Name,:Enter\_Branch,:Enter\_Room\_No,:Enter\_City,:Enter\_State,:Enter\_MobileNo);

END;

**Update data into student table:**

declare

PROCEDURE updatestudent( rollno varchar,dob1 date ,fname1 varchar,lname1 varchar,branch1 varchar, room varchar,city1 varchar,state1 varchar) AS

BEGIN

update student set dob=dob1,fname=fname1,lname=lname1,branch=branch1,room\_no=room, city=city1,state=state1 where roll=rollno ;

if sql%found then

dbms\_output.put\_line('successfully updated!');

else

dbms\_output.put\_line('Not updated Successfully! Please reenter your values..');

end if;

END;

Begin

updatestudent(:Enter\_Roll\_No,:Enter\_Your\_Dob,:Father\_Name,:Mother\_Name,:Enter\_Branch,:Enter\_Room\_No,:Enter\_City,:Enter\_State);

END;

**Delete a record :**

declare

PROCEDURE deletestudent( rollno varchar) AS

BEGIN

delete from student where roll=rollno;

if sql%found

then

dbms\_output.put\_line('One Row deleted!');

else

dbms\_output.put\_line('No row deleted..Please check your values...');

end if;

END;

begin

deletestudent(:Enter\_Roll\_No);

END;

**Retrieve data from the student table:**

declare

procedure retrieve(rollno varchar) is

cursor c1 is select \* from student where roll= rollno;

rec c1%rowtype;

begin

open c1;

loop

fetch c1 into rec;

exit when c1%notfound;

dbms\_output.put\_line('Roll No is '||rec.roll);

dbms\_output.put\_line('Date of Birth is '||rec.dob);

dbms\_output.put\_line('Father name is '||rec.fname);

dbms\_output.put\_line('Mother name is '||rec.lname);

dbms\_output.put\_line('branch is '||rec.branch);

dbms\_output.put\_line('Room No is '||rec.room\_no);

dbms\_output.put\_line('Your City is '||rec.city);

dbms\_output.put\_line('Your State is '||rec.state);

dbms\_output.put\_line('Contact No is '||rec.mobile);

end loop;

close c1;

end;

begin

retrieve(:Enter\_roll\_no);

end

**BILL Table:**

**Create table bill:**

create table bill (roll varchar(20) references student(roll),month varchar(4) not null ,year varchar(20) not null,amount number(37) not null,perdaycharge number(20) not null ,primary key(roll,month,year));

# Queries Regarding the INSERT, UPDATE, DELETE, RETRIEVE On BILL Table

**Insert data into bill:**

declare

parent\_key EXCEPTION;

PRAGMA EXCEPTION\_INIT(Parent\_key,-02291);

PROCEDURE insertbill( rollno varchar, month varchar, year varchar, amount number, perday number) AS

BEGIN

insert into bill values(rollno,month,year,amount,perday);

EXCEPTION

when Parent\_key then

DBMS\_OUTPUT.PUT\_LINE('Entered Roll No is not found here!!');

END;

begin

insertbill(:Enter\_Roll\_No,:Enter\_Month,:Enter\_year,:Enter\_TotalAmount,:Enter\_PerDay\_Charges);

END;

**Update data in Bill Table:**

declare

PROCEDURE updatebill( rollno varchar, month1 varchar, year1 varchar, amount1 number, perday number) AS

BEGIN

update bill set month=month1,year=year1,amount=amount1,perdaycharge=perday where roll=rollno;

if sql%found then

dbms\_output.put\_line('successfully updated!');

else

dbms\_output.put\_line('Not updated Successfully! Please reenter your values..');

end if;

END;

begin

updatebill(:Enter\_Roll\_No,:Enter\_Month\_Again,:Enter\_Year\_Again,:Enter\_TotalAmount,:Enter\_PerDay\_Charges);

END;

**Delete Data in Bill Table:**

declare

PROCEDURE deletebill( rollno varchar,month1 varchar) AS

BEGIN

delete from bill where roll=rollno and month=month1;

if sql%found

then

dbms\_output.put\_line('One Row deleted!');

else

dbms\_output.put\_line('No row deleted..Please check your values...');

end if;

END;

begin

deletebill(:Enter\_Roll\_No,:Enter\_Month);

END;

**Retrieval Of Data from Bill Table:**

declare

procedure retrievebill(rollno varchar,month1 varchar) is

cursor c1 is select \* from bill where roll= rollno and month=month1;

rec c1%rowtype;

begin

open c1;

loop

fetch c1 into rec;

exit when c1%notfound;

dbms\_output.put\_line('Roll No is '||rec.roll);

dbms\_output.put\_line('Bill Month is '||rec.month);

dbms\_output.put\_line('Corresponding Year is '||rec.year);

dbms\_output.put\_line('Total Amount is '||rec.amount);

dbms\_output.put\_line('Per Day Charge is '||rec.perdaycharge);

end loop;

close c1;

end;

begin

retrievebill(:Enter\_roll\_no,:Enter\_Month);

end

**Complaint Table:**

**Create table complaint:**

create table complaint( roll references student(roll) not null,complaint\_no varchar(20) primary key ,dcomplaint date not null, cquery varchar(37) not null);

# Queries Regarding the INSERT, DELETE, RETRIEVAL on Complaint Table

**Insertion of complaint by student**

declare

parent\_key EXCEPTION;

PRAGMA EXCEPTION\_INIT(Parent\_key,-02291);

PROCEDURE insertcomplaint( rollno varchar,cdate date,cquery1 varchar) AS

BEGIN

insert into complaint values(rollno,incre1.nextval,cdate,cquery1);

EXCEPTION

when Parent\_key then

DBMS\_OUTPUT.PUT\_LINE('Entered Roll No is not found here!!');

END;

begin

insertcomplaint(:Enter\_Roll\_No,:Enter\_Complaint\_date,:Enter\_Your\_Complaint);

END;

**Deletion of Complaint By Admin:**

declare

PROCEDURE delcomplaint( complaintno number) AS

BEGIN

delete from complaint where complaint\_no=complaintno;

if sql%found

then

dbms\_output.put\_line('One Row deleted!');

else

dbms\_output.put\_line('No row deleted..Please check your values...');

end if;

END;

begin

delcomplaint(:Enter\_complaint\_date);

END;

**Retrieval Of complaint By Admin:**

declare

procedure retrievecomplaint(cno varchar) is

cursor c1 is select \* from complaint where complaint\_no= cno;

rec c1%rowtype;

begin

open c1;

loop

fetch c1 into rec;

exit when c1%notfound;

dbms\_output.put\_line('Roll No is '||rec.roll);

dbms\_output.put\_line('Complaint Number is '||rec.complaint\_no);

dbms\_output.put\_line('Date of Complaint is '||rec.dcomplaint);

dbms\_output.put\_line('Your Query is '||rec.cquery);

end loop;

close c1;

end;

begin

retrievecomplaint(:Enter\_complaint\_no);

end

**Leave Table:**

**Create table leave:**

create table leave( roll references student(roll) not null, l\_no number(10) primary key, ddate date not null, adate date not null);

# Queries Regarding the INSERT, RETRIEVAL on Leave Table

**Insertion Of Leave Entry By the Student:**

declare

parent\_key EXCEPTION;

PRAGMA EXCEPTION\_INIT(Parent\_key,-02291);

PROCEDURE insertleave( rollno varchar,ddate date,adate date) AS

BEGIN

insert into leave values(rollno,incre.nextval,ddate,adate);

EXCEPTION

when Parent\_key then

DBMS\_OUTPUT.PUT\_LINE('Entered Roll No is not found here!!');

END;

begin

insertleave(:Enter\_Roll\_No,:Enter\_Departure\_date,:Enter\_Arrival\_date);

END;

**Retrieval of leaves By the Admin as Well as Mess Manager:**

declare

procedure retrieveleave(rno varchar) is

cursor c1 is select \* from leave where roll= rno;

rec c1%rowtype;

begin

open c1;

loop

fetch c1 into rec;

exit when c1%notfound;

dbms\_output.put\_line('-------------------------------');

dbms\_output.put\_line('Roll No is '||rec.roll);

dbms\_output.put\_line('Leave Number is '||rec.l\_no);

dbms\_output.put\_line('Date of departure is '||rec.ddate);

dbms\_output.put\_line('Your Arrival date is '||rec.adate);

dbms\_output.put\_line('-------------------------------');

end loop;

close c1;

end;

begin

retrieveleave(:Enter\_roll\_no);

end

**Menu Table:**

**Create table menu:**

create table menu(day varchar(20) primary key, breakfast varchar(20) not null, lunch varchar(20) not null, dinner varchar(20) not null)

# Queries Regarding the INSERT, UPDATE, DELETE, RETRIEVAL on Menu Table

**Insertion of menu By Mess Manager:**

declare

PROCEDURE insertmenu( day varchar, breakfast varchar, lunch varchar, dinner varchar) AS

BEGIN

insert into menu values(day,breakfast,lunch,dinner);

END;

begin

insertmenu(:Enter\_Day,:Enter\_Breakfast,:Enter\_lunch,:Enter\_Dinner);

END;

**Updation of menu By Mess Manager:**

declare

PROCEDURE updatemenu( day1 varchar, breakfast1 varchar, lunch1 varchar, dinner1 varchar) AS

BEGIN

update menu set breakfast=breakfast1,lunch=lunch1,dinner= dinner1 where day=day1;

if sql%found then

dbms\_output.put\_line('successfully updated!');

else

dbms\_output.put\_line('Not updated Successfully! Please reenter your values..');

end if;

END;

begin

updatemenu(:Enter\_Day,:Enter\_Breakfast,:Enter\_lunch,:Enter\_Dinner);

END;

**Deletion of menu By Mess Manager:**

declare

PROCEDURE deletemenu(day1 varchar) AS

BEGIN

delete from menu where day=day1;

if sql%found

then

dbms\_output.put\_line('One Row deleted!');

else

dbms\_output.put\_line('No row deleted..Please check your values...');

end if;

END;

begin

deletemenu(:Enter\_Day);

END;

**Retrieval Of Menu:**

declare

procedure retrievemenu(day1 varchar) is

cursor c1 is select \* from menu where day= day1;

rec c1%rowtype;

begin

open c1;

loop

fetch c1 into rec;

exit when c1%notfound;

dbms\_output.put\_line('-------------------------------');

dbms\_output.put\_line('Corresponding Day is '||rec.day);

dbms\_output.put\_line('Breakfast Menu '||rec.breakfast);

dbms\_output.put\_line('Afternoon Menu '||rec.lunch);

dbms\_output.put\_line('Dinner menu is '||rec.dinner);

dbms\_output.put\_line('-------------------------------');

end loop;

close c1;

end;

begin

retrievemenu(:Enter\_day);

end

**Mess\_Staff Table:**

**Create table mess\_staff:**

create table mess\_staff (staffed varchar(20) primary key,name varchar(20) not null, doj date not null)

# Queries Regarding the INSERT, UPDATE, DELETE, RETRIEVAL on Mess Staff Table

**Insertion of mess staff:**

declare

PROCEDURE insertstaff( id varchar, name varchar, doj1 date) AS

BEGIN

insert into mess\_staff values(id,name,doj1);

END;

begin

insertstaff(:Enter\_Staff\_ID,:Enter\_Name,:Enter\_DateOfJoin);

END;

**Update of Mess Staff:**

declare

PROCEDURE Updatestaff( id varchar, name1 varchar, doj1 date) AS

BEGIN

update mess\_staff set name=name1,doj=doj1 where staffed=id;

if sql%found then

dbms\_output.put\_line('successfully updated!');

else

dbms\_output.put\_line('Not updated Successfully! Please reenter your values..');

end if;

END;

begin

Updatestaff(:Enter\_Staff\_ID,:Enter\_Name,:Enter\_DateOfJoin);

END;

**Deletion of Mess Staff**

declare

PROCEDURE deletestaff( id varchar) AS

BEGIN

delete from mess\_staff where staffed=id;

if sql%found

then

dbms\_output.put\_line('One Row deleted!');

else

dbms\_output.put\_line('No row deleted..Please check your values...');

end if;

END;

begin

deletestaff(:Enter\_Staff\_ID);

END;

**Retrieve Data from Mess Staff Table:**

declare

procedure retrievestaff(id varchar) is

cursor c1 is select \* from mess\_staff where staffed= id;

rec c1%rowtype;

begin

open c1;

loop

fetch c1 into rec;

exit when c1%notfound;

dbms\_output.put\_line('-------------------------------');

dbms\_output.put\_line('Corresponding ID is '||rec.staffed);

dbms\_output.put\_line('Name is '||rec.name);

dbms\_output.put\_line('Date of Joining is '||rec.doj);

dbms\_output.put\_line('-------------------------------');

end loop;

close c1;

end;

begin

retrievestaff(:Enter\_id);

end

**Mess Member Table:**

**Create table messmem:**

create table messmem(roll references student(roll),mid varchar(20) primary key);

# Queries Regarding the INSERT, UPDATE, DELETE, RETRIEVAL on Mess Member Table

**Insert data into mess member Table:**

declare

parent\_key EXCEPTION;

PRAGMA EXCEPTION\_INIT(Parent\_key,-02291);

PROCEDURE insertmessmem( rollno varchar, mid varchar) AS

BEGIN

insert into messmem values(rollno,mid);

EXCEPTION

when Parent\_key then

DBMS\_OUTPUT.PUT\_LINE('Entered Roll No is not found here!!');

END;

begin

insertmessmem(:Enter\_Roll\_No,:Enter\_Mess\_ID);

END;

**Update Data in Table :**

declare

PROCEDURE updatemessmem( rollno varchar, mid1 varchar) AS

BEGIN

update messmem set roll=rollno where mid=mid1;

if sql%found then

dbms\_output.put\_line('successfully updated!');

else

dbms\_output.put\_line('Not updated Successfully! Please reenter your values..');

end if;

END;

begin

updatemessmem(:Enter\_Roll\_No,:Enter\_Mess\_ID);

END;

**Delete data :**

declare

PROCEDURE deletemessmem( mid1 varchar) AS

BEGIN

delete from messmem where mid=mid1;

if sql%found

then

dbms\_output.put\_line('One Row deleted!');

else

dbms\_output.put\_line('No row deleted..Please check your values...');

end if;

END;

begin

deletemessmem(:Enter\_Mess\_ID);

END;

**Retrieve data:**

declare

procedure retrieveid(id varchar) is

cursor c1 is select \* from messmem where mid= id;

rec c1%rowtype;

begin

open c1;

loop

fetch c1 into rec;

exit when c1%notfound;

dbms\_output.put\_line('-------------------------------');

dbms\_output.put\_line('Corresponding Roll No is '||rec.roll);

dbms\_output.put\_line('Mess ID is '||rec.mid);

dbms\_output.put\_line('-------------------------------');

end loop;

close c1;

end;

begin

retrieveid(:Enter\_id);

end

**Mess ID Table:**

**Create table messid:**

create table messid(mid references messmem(mid),year varchar(20));

# Queries Regarding the INSERT, UPDATE, DELETE, RETRIEVAL on Mess ID Table

**Insert into messid:**

declare

parent\_key EXCEPTION;

PRAGMA EXCEPTION\_INIT(Parent\_key,-02291);

PROCEDURE insertmid( mid1 varchar,year1 varchar) AS

BEGIN

insert into messid values(mid1,year1);

EXCEPTION

when Parent\_key then

DBMS\_OUTPUT.PUT\_LINE('Entered Roll No is not found here!!');

END;

begin

insertmid(:Enter\_Mess\_ID,:Enter\_Year);

END;

**Updation Of Data:**

declare

PROCEDURE updatemid( mid1 varchar,year1 varchar) AS

BEGIN

update messid set year=year1 where mid=mid1;

if sql%found then

dbms\_output.put\_line('successfully updated!');

else

dbms\_output.put\_line('Not updated Successfully! Please reenter your values..');

end if;

END;

begin

updatemid(:Enter\_Mess\_ID,:Enter\_Year);

END;

**Deletion Of Data:**

declare

PROCEDURE deletemid( mid1 varchar) AS

BEGIN

delete from messid where mid=mid1;

if sql%found

then

dbms\_output.put\_line('One Row deleted!');

else

dbms\_output.put\_line('No row deleted..Please check your values...');

end if;

END;

begin

deletemid(:Enter\_Mess\_ID);

END;

**Retrieve Data:**

declare

procedure retrieveyear(id varchar) is

cursor c1 is select \* from messid where mid= id;

rec c1%rowtype;

begin

open c1;

loop

fetch c1 into rec;

exit when c1%notfound;

dbms\_output.put\_line('-------------------------------');

dbms\_output.put\_line('Corresponding Mess ID is '||rec.mid);

dbms\_output.put\_line('Year as a Mess Member is '||rec.year);

dbms\_output.put\_line('-------------------------------');

end loop;

close c1;

end;

begin

retrieveyear(:Enter\_id);

end

**Triggers for complaint Table and Leave Table**

**Complaint Table:**

CREATE OR REPLACE TRIGGER complaintauto

BEFORE INSERT ON complaint

FOR EACH ROW

BEGIN

SELECT incre1.NEXTVAL

INTO :new.complaint\_no

FROM dual;

END;

//Use Of Sequences

CREATE SEQUENCE incre1

START WITH 1

INCREMENT BY 1

CACHE 100;

**Leave Table:**

//Use Of Trigger

CREATE OR REPLACE TRIGGER leaveauto

BEFORE INSERT ON leave

FOR EACH ROW

BEGIN

SELECT incre.NEXTVAL

INTO :new.l\_no

FROM dual;

END;

//Use Of Sequence

CREATE SEQUENCE incre

START WITH 1

INCREMENT BY 1

CACHE 100;

**CONCLUSION**

**To conclude the description about the project, developed using PHP and ORACLE SQL 11g is based on the requirement specification of the user and the analysis of the existing system, with flexibility for future enhancement. ONLINE HOSTEL-MESS MANAGEMENT SYSTEM is very useful for mess fee calculation, register complaints & leave entries. This hostel-mess management software is designed for people who want to manage various activities in the hostel. For the past few years the numbers of educational institutions are increasing rapidly. And hence there is a lot of strain on the person who is running the hostel and softwares are not usually used in this context. This particular project deals with the problems on managing a hostel and avoids the problems which occur when carried manually. Identification of the drawbacks of the existing system leads to the designing of computerized system that will be compatible to the existing system with the system which is more user friendly and more GUI oriented.**

**(45)**

**REFERENCES**

* **Parteek Bhatia and Gurvinder Singh, “Simplified approach to DBMS”.**
* **Leave form from J-hostel.**
* **Wikipedia**
* **www.stackoverflow.com**

**(46)**