**NATIONAL INSTITUTE OF TECHNOLOGY, TIRUCHIRAPPALLI-15**

****

**Department of Computer Applications**

***College Management System***

**PROJECT WORK**

*Submitted By*

**Kushbinder singh – 205118033**

*Under the guidance of*

**Dr. R Eswari**

*Submitted in fulfillment of the project in PL/SQL*

**NATIONAL INSTITUTE OF TECHNOLOGY, TIRUCHIRAPPALLI-15**

****

**CERTIFICATE**

*This is to certify that* ***Mr. Kushbinder singh,*** *student of 2nd semester MCA (batch 2018-2021) of National Institute of Technology, Tiruchirappalli has successfully completed the project* ***College Management System*** *in PL/SQL**under the guidance of* ***Ms. R Eswari.***

Signature

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Ser. No** | **Description** | **Page No.** |
| 1 | Problem Description and Analysis |  |
| 2 | Schema and Tables Descriptions |  |
| 3 | Creation of Tables using SQL |  |
| 4 | Defining Triggers For Tables |  |
| 5 | Procedures |  |
| 6 | Sample Data and Results |  |
| 7 | Enhancement and conclusion |  |
| 8 | References & bibliography | 25 |

**Problem Description and Analysis**

***School Management System***

In existing system, professor takes the attendance manually and records the details of absentees or presenters. Using this records professor stores the student’s attendance in each day in the attendance list and calculates the attendance percentage manually. In this system professor must consider each student separately and keep this list safely. It takes lot of time. Professor must spend more time to record the each student’s attendance in the attendance list.

The project aims in implementing software that helps professor to take the attendance using mobile phone. For this mobile must support java application. This project is based on J2ME technology. Professor enters the attendance to the mobile using keypad. It is a manual process. After saving the absentees or presenters in to the mobile, teacher can edit the attendance list in the mobile. And this attendance details can send to a computer using GPRS. After sending the files to a computer, attendance list must be updated automatically.

**Modules**

**1.User Module:**

The main purpose of the user module is to provide security. This module is specially designed for staffs, which use mobile phone to take attendance. Each staff enter username and password before enter in to attendance list. If username and password cannot match, he/she can’t enter in to attendance page.

**2. Attendance Entry Module:**

The purpose of Attendance EntryModule is to mark the attendance using cell phone. In this module Lecturer takes the attendance using the cell phone. Lecturers select the branch, semester and year. After this session he enters in to attendance page. Here staff makes a mark.

**Schema and Tables Description**

Following tables are used in the project :-

Table **STUD**

|  |  |
| --- | --- |
| Roll  Name  Dob  Address  Phone  Doj  Classid | Primary key  Foreign key |

Table **CLASs**

|  |  |
| --- | --- |
| Classid  Clas  Sec  Teacherid  Syllabusid  Timetableid | Primary key  Foreign key  Foreign key  Foreign key |

Table **TEACHER**

|  |  |
| --- | --- |
| Teacherid  Name  Subject1  Subject2  Phone | Primary key |

Table **ATTENDANCE**

|  |  |
| --- | --- |
| Roll  Present  Absent  Leaves | Primary key |

Table **fee**

|  |  |
| --- | --- |
| Roll  mon  fine  amount | Primary key |

Table **marks**

|  |  |
| --- | --- |
| Roll  maths  english  science  hindi  sst  tot  pass | Primary key |

**Creating Tables**

1. **TABLE STUD:**

create table stud

( roll number(4) PRIMARY KEY ,

name varchar2(25),

dob date,

address varchar2(30),

phone number(10),

doj date,

classid number(3)

);

Table created.

1. **TABLE CLAS**

create table clas

( classid number(3) PRIMARY KEY,

clas number(2),

sec char(1),

teacherid number(3),

syllabusid number(3),

timetableid number(3)

);

Table created.

1. **TABLE TEACHER:**

create table teacher

( teacherid number(3) PRIMARY KEY,

name varchar2(25),

subject1 varchar(15),

subject2 varchar(15),

phone number(10)

);

Table Created.

**4.TAblE ATTENDENCE**

customerid number not null ,

loanid number not null,

branchid number not null,

operation varchar2(20),

userid varchar2(20),

opdate date)

Table created.

**5.TABLE MARKS**

create table marks

(

roll number(4) PRIMARY KEY,

maths number(3),

english number(3),

science number(3),

hindi number(3),

sst number(3),

tot number(3),

pass varchar2(4)

);

Table created.

**6. TABLE FEE:**

create table fee

(

roll number(4),

mon char(3),

fine number(6,2),

amount number(8,2)

);

Table Created.

**CREATING TRIGGERS**

**1. VALIDATING ROLL NUMBER**

create or replace trigger chk\_rollno before insert on student

for each row

begin

if :new.roll<0101 then

raise\_application\_error(-20011,'INVALID ROLL NUMBER !!');

end if;

end;

/

create or replace trigger chk\_rollno1 before insert on student

for each row

begin

if :new.roll>1299 then

raise\_application\_error(-20011,'INVALID ROLL NUMBER !!');

end if;

end;

/

**Query to check trigger-**

insert into student values(0001,'hari','08-mar-1998','indore',7415761326,'01-jun-2018',001);

insert into student values(1300,'hari','08-mar-1998','indore',7415761326,'01-jun-2018',001);

**2. DELETE NOT ALLOWED ON CLASS TABLE**

create or replace trigger classs\_del before delete on class for each row

begin

raise\_application\_error(-20002,'CANT DELETE');

end;

/

**Query to check trigger-**

delete from class where classsid=001;

**3. FOR UPDATION NOT ALLOWED**

CREATE or replace TRIGGER result\_alter before update on marks for each row

declare

begin

raise\_application\_error(-20002,'UPDATION NOT ALLOWED');

end;

/

**Query to check trigger-**

update marks set pass='no' where roll=0101;

**4. UPDATION OR DELETION NOT ALLOWED ON FEE TABLE**

create or replace trigger fee\_alter before delete or update on fee for each row

declare

begin

raise\_application\_error(-20002,'updation or deletion not allowed');

end;

/

**Query to check trigger-**

delete from fee where roll=0101;

**5. VALIDATION SECTION ENTERED**

create or replace trigger chk\_sec before insert on class

for each row

WHEN(new.sec!=’a’ and new.sec!=’b’)

begin

raise\_application\_error(-20011,'INVALID SECTION ENTERED!!');

end;

/

**Query to check trigger-**

insert into class values(002,2,'c',202,302,402);

**PROCEDURE CREATION**

1. **PROCEDURE FOR ADDING MORE STUDENT IN STUDENT TABLE**

create or replace procedure addstd(r number,sname varchar2,dob date,addr varchar2,ph number,dj date,cid number)

is

begin

insert into student values(r,sname,dob,addr,ph,dj,cid);

end;

/

**EXECUTING PROCEDURE-**

exec addstd(0101,'hari','09-mar-2013','ashok nagar','7415681326','09-jun-2019',001);

**2. RESULT GENERATION**

Create or replace procedure all\_result is

temp marks%rowtype;

rec student%rowtype;

cursor c1 is select roll from marks;

begin

dbms\_output.put\_line('Roll | Name ||Eng ||Maths ||Science||SSt ||Hindi ||Total ||Result ');

for i in c1

loop

select \* into temp from marks where marks.roll=i.roll;

select \* into rec from student where student.roll=i.roll;

dbms\_output.put\_line(temp.roll || ' '||rec.name||' '||temp.english ||' ' || temp.maths

||' ' ||temp.science||' '|| temp.sst||' ' || temp.hindi||' ' ||temp.tot||' '||temp.pass);

end loop;

exception when no\_data\_found then

dbms\_output.put\_line('Incomplete entry of either studentent details or marks');

end;

**EXECUTING PROCEDURE-**

exec all\_result;

**3.FOR ADDING CLASSES IN CLASS TABLE**

create or replace procedure addcls(cid number,c number,s char,tid number,sid number,ttid number)

is

begin

insert into class values(cid,c,s,tid,sid,ttid);

end;

/

**EXECUTING PROCEDURE-**

exec addcls(002,2,'b',202,302,402);

**4.RESULT FOR EACH CLASSS**

create or replace procedure classs\_result(ClassId number) is

student\_rec student%rowtype;

classs\_rec class%rowtype;

marks\_rec marks%rowtype;

cursor student\_cur

is select \* from student where classsid=ClassId;

begin

open student\_cur;

loop

fetch student\_cur into student\_rec;

exit when student\_cur %notfound;

select \* into classs\_rec from class where student\_rec.classsid=class.classsid;

select \* into marks\_rec from marks where marks.roll=student\_rec.roll;

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

dbms\_output.put\_line('Classs Result for: '|| classs\_rec.class);

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

dbms\_output.put\_line('Roll number : '||student\_rec.roll );

dbms\_output.put\_line('Name : '||student\_rec.name );

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

dbms\_output.put\_line('English : '||marks\_rec.english );

dbms\_output.put\_line('Mathematics : '||marks\_rec.maths );

dbms\_output.put\_line('Hindi : '||marks\_rec.hindi );

dbms\_output.put\_line('Science : '||marks\_rec.science );

dbms\_output.put\_line('Social Sc. : '||marks\_rec.sst );

dbms\_output.put\_line('Total : '||marks\_rec.tot );

dbms\_output.put\_line('Result : '||marks\_rec.pass );

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

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

end loop;

exception when no\_data\_found then

dbms\_output.put\_line('Result has not been calculated...');

end;

/

**EXECUTING PROCEDURE-**

exec classs\_result(001);

**5. REPORT CARD FOR EACH STUDENTENT**

create or replace procedure reportcard(rol number) is

student\_rec student%rowtype;

classs\_rec class%rowtype;

marks\_rec marks%rowtype;

att\_rec attendance%rowtype;

begin

select \* into student\_rec from student where student.roll=rol;

select \* into classs\_rec from class where student\_rec.classsid=class.classsid;

select \* into marks\_rec from marks where marks.roll=student\_rec.roll;

select \* into att\_rec from attendance where attendance.roll=rol;

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

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

dbms\_output.put\_line('Roll number : '||student\_rec.roll );

dbms\_output.put\_line('Name : '||student\_rec.name );

dbms\_output.put\_line('Classs : '||classs\_rec.class );

dbms\_output.put\_line('Section : '||classs\_rec.sec );

dbms\_output.put\_line('SUBJECTS MARKS');

dbms\_output.put\_line('English : '||marks\_rec.english );

dbms\_output.put\_line('Mathematics : '||marks\_rec.maths );

dbms\_output.put\_line('Hindi : '||marks\_rec.hindi );

dbms\_output.put\_line('Science : '||marks\_rec.science );

dbms\_output.put\_line('Social Sc. : '||marks\_rec.sst );

dbms\_output.put\_line('Total : '||marks\_rec.tot );

dbms\_output.put\_line('Result : '||marks\_rec.pass );

dbms\_output.put\_line('ATTENDANCE : '||att\_rec.present ||' Days');

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

exception when no\_data\_found then

dbms\_output.put\_line('Result has not been calculated ...');

end;

/

**EXECUTING PROCEDURE-**

exec reportcard(0101);

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Roll number : 101

Name : hari

Classs : 1

Section : a

SUBJECTS MARKS

English : 40

Mathematics : 40

Hindi : 40

Science : 40

Social Sc. : 40

Total : 200

Result : yes

ATTENDANCE : 200 Days

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**6. ATTENDANCE DETAILS**

create or replace procedure attendan(rol number) is

rec student%rowtype;

att attendance%rowtype;

classs\_rec class%rowtype;

begin

dbms\_output.put\_line('\*\*\*\*\*\*\*\*\*\*\* ATTENDANCE DETAILS \*\*\*\*\*\*\*\*\*\*\*\*');

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

select \* into rec from student where student.roll=rol;

select \* into att from attendance where attendance.roll=rec.roll;

select \* into classs\_rec from class where rec.classsid=class.classsid;

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

dbms\_output.put\_line('Roll number : '||rec.roll );

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

dbms\_output.put\_line('Classs : '||classs\_rec.class );

dbms\_output.put\_line('Section : '||classs\_rec.sec );

dbms\_output.put\_line('ATTENDANCE NO OF DAYS');

dbms\_output.put\_line('PRESENT : '||att.present );

dbms\_output.put\_line('ABSENT : '||att.absent );

dbms\_output.put\_line('PRIOR LEAVES : '||att.leaves );

if att.absent=0 then

dbms\_output.put\_line('\*~~~~~ PERFECT ATTENDANCE ~~~~~\*');

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

elsif att.absent>45 then

dbms\_output.put\_line('\*LACKS ATTENDANCE -NOT ELIGIBLE FOR APPEARING IN EXAMINATION');

else

dbms\_output.put\_line('\*ELIGIBLE FOR APPEARING IN EXAMINATION');

end if;

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

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

exception when no\_data\_found then

dbms\_output.put\_line('Data has not been issued...');

end;

/

**EXECUTING PROCEDURE-**

SQL>exec attendan(0101)

\*\*\*\*\*\*\*\*\*\*\* ATTENDANCE DETAILS \*\*\*\*\*\*\*\*\*\*\*\*

----------------------------------------------

Roll number : 101

Name : hari

Classs : 1

Section : a

ATTENDANCE NO OF DAYS

PRESENT : 200

ABSENT : 100

PRIOR LEAVES : 0

**Enhancement and conclusion**

Enhancement:

This is a very basic effort to computerize the mobile based attendance management. There are yet a lot of features that can be added to the system. Some of them are as follows:

* We can provide the facility of generating time-table with teachers’ allotment to each period depending upon their subjects etc.
* We can provide the features of saving the files for each month of the year for fee calculation.

Conclusion:

This project helped in realizing the concepts of analyzing databases and programming in SQL, PL/SQL.

As it is not an easy task to start with a project, least to mention an database oriented one. On this regard I owe my whole hearted thanks to our faculty

**Ms. R Eswari** for lending a helping hand during the course of my project.

**References and Bibliography**

1. Ivan Bayross, SQL, PL/SQL The programming language of Oracle
2. Korth, Sudershan, Databse System Concepts
3. Complete reference for oracle