Database Management Systems

*Project 2*

|  |  |
| --- | --- |
| COURSE | Database Management Systems |
| SECTION | 001 |
| SEMESTER | Fall 2015 |
| INSTRUCTOR | Dr. Le Gruenwald |
| GROUP PROJECT NUMBER | 4 |
| GROUP MEMBERS | David Findley, Coleman Jackson, Swati Nemade, Stephen Smart |
| SCORE |  |

Problem 1)

**PROBLEM 1 SQL FILE**

-- As a refresher, here is the schema of our database.

/\*

create table Student(snum integer, sname varchar(20), major varchar(20), slevel varchar(10), age int, primary key(snum));

create table Class(cname varchar(50), meets\_at varchar(10), room varchar(10), fid int, primary key(cname));

create table Enrolled(snum int, cname varchar(50), primary key(snum, cname));

create table Faculty(fid int, fname varchar(20), deptid int, salary float, primary key(fid));

\*/

/\*

Upon reviewing our Project 2 Problem 2 database and related queries, we have

decided to implement a secondary index on the “fname” field in the Faculty

table. We did this after recognizing that we frequently reference faculty

members by their name, a field which is not a primary key and therefore not

indexed. By indexing faculty name we hope to increase the relative speed of

our queries looking for faculty by name. To this end, we reran queries 2 and

3 respectively, “Find the names of all Juniors who are enrolled in a class

taught by Johnson” and “Find the age of the oldest student who is either a

History major or is enrolled in a course taught by Johnson.” Both of these

queries should now be much more efficient at locating the tuple containing

Johnson’s data.

\*/

-- This statement creates the secondary index described.

create index facIndex on Faculty (fname);

-- Query 2: Find the names of all Juniors who are enrolled in a class taught by

-- Johnson

select sname

from Student

where snum in (

select snum

from Enrolled

where enrolled.cname in (

select cname

from Faculty, Class

-- In this line the index of "fname" will be used to find 'Johnson'

where Faculty.fid = Class.fid and Faculty.fname = 'Johnson'

)

)

and slevel = 'JR';

-- Query 3: Find the age of the oldest student who is either a History major or

-- is enrolled in a course taught by johnson

select max(age)

from Student

where snum in (

select snum

from Enrolled

where enrolled.cname in (

select cname

from Faculty, Class

-- Here as well we will make use of our index

where Faculty.fid = Class.fid and Faculty.fname = 'Johnson'

)

)

or major = 'History';

**PROBLEM 1 OUTPUT**

Index FACINDEX created.  
  
SNAME   
--------------------  
Daniels   
Daniels   
  
  
 MAX(AGE)  
----------  
 24

Problem 2)

**PLS FILES**

create or replace PROCEDURE PROBLEM2\_1 (fid INTEGER, fname VARCHAR, deptid INTEGER)

AS

-- These variables will store the average mean calc values

salary INTEGER;

dept\_total INTEGER;

dept\_num INTEGER;

dept\_avg FLOAT;

BEGIN

-- Initialize the values

dept\_total := 0;

dept\_num := 0;

-- for all faculty

FOR someone IN (select \* from Faculty)

LOOP

-- if someone is in that department then add them to the average

IF (someone.deptid = deptid)

THEN dept\_total := dept\_total + someone.salary;

dept\_num := dept\_num + 1;

END IF;

END LOOP;

-- calculate the average

dept\_avg := (dept\_total/dept\_num);

-- Set the actual value of the salary to 90% if over 50000

IF (dept\_avg > 50000)

THEN salary := (.9 \* dept\_avg);

-- to 100% if less than 30000

ELSIF (dept\_avg < 30000)

THEN salary := dept\_avg;

-- or to 80% if otherwise

ELSE salary := (.8 \* dept\_avg);

END IF;

-- now insert this faculty into the database

insert into FACULTY (fid, fname, deptid, salary) values (fid, fname, deptid, salary);

END PROBLEM2\_1;

create or replace PROCEDURE PROBLEM2\_2 (fid NUMBER, fname VARCHAR, deptid NUMBER, hidden\_dept NUMBER)

AS

-- variables need to compute average

salary NUMBER;

dept\_total NUMBER;

dept\_num NUMBER;

BEGIN

-- initialize values

dept\_total := 0;

dept\_num := 0;

-- for all faculty without the blocked dept

FOR someone IN (select \* from Faculty)

LOOP

-- add their salary to total and note their addition

IF (someone.deptid != hidden\_dept)

THEN dept\_total := dept\_total + someone.salary;

dept\_num := dept\_num + 1;

END IF;

END LOOP;

-- compute the average

salary := dept\_total/dept\_num;

-- insert the new faculty tuple

insert into FACULTY (fid, fname, deptid, salary) values (fid, fname, deptid, salary);

END PROBLEM2\_2;

**JAVA PROGRAM**

import java.sql.\*;

import java.util.Scanner;

// Driver class for problem 2 of project 3

public class GP3\_Problem2\_Group4

{

// This main method will begin the program.

public static void main(String[] args) throws SQLException

{

Scanner input = new Scanner(System.in);

System.out.println("AddFaculty 1.0\n");

ResultSet res;

CallableStatement call = null;

// Try to load the drive class for Oracle SQL

try

{

Class.forName("oracle.jdbc.OracleDriver");

}

// Catch an error in the loading of the driver

catch(Exception x)

{

System.out.println("Unable to load the driver class!");

}

// This string represents the url of the Oracle connection

String sourceURL = "jdbc:oracle:thin:@//oracle.cs.ou.edu:1521/pdborcl.cs.ou.edu";

// Here is our connection object

Connection dbConnection = null;

// Try to add the connection

try

{

// Get connection

dbConnection = DriverManager.getConnection(sourceURL,"jack0164","BNcy9Jv1");

}

// catch a failure in the connection

catch( SQLException x )

{

System.out.println("Couldn’t get connection!");

}

// create a statement object that is connected to our database

Statement stmt = dbConnection.createStatement();

String[] s;

// loop here until user selects close

while(true)

{

// string array to hold parameters queried for answers

s = queryUser(input);

// do certain things based on the size of the array

if(s == null)

{

// for an empty array close the connection and the program

dbConnection.close();

System.exit(0);

}

else if(s.length == 1)

// for length of 1 print all the faculty using the applicable method

printResults(stmt.executeQuery("select \* from Faculty order by fid"));

else if(s.length == 3)

{

// for a length of 3 we will call the PROBLEM2\_1 procedure with the array arguments

call = dbConnection.prepareCall("{call PROBLEM2\_1(?, ?, ?)}");

call.setString(1, s[0]);

call.setString(2, s[1]);

call.setString(3, s[2]);

call.execute();

}

else if(s.length == 4)

{

// for 4 arguments we do the same but for the PROBLEM2\_2 procedure

call = dbConnection.prepareCall("{call PROBLEM2\_2(?, ?, ?, ?)}");

call.setString(1, s[0]);

call.setString(2, s[1]);

call.setString(3, s[2]);

call.setString(4, s[3]);

call.execute();

}

}

}

// this method will print the results of our Faculty work

private static void printResults(ResultSet res) throws SQLException

{

// print a header line

System.out.println("FID FNAME DEPTID SALARY\n");

// while there is a result left

while(res.next())

{

// print the result as a line in the console

System.out.println(res.getString(1) + " " + res.getString(2) + " " + res.getString(3) + " " + res.getString(4) + "\n");

}

}

// method to query the user for input

private static String[] queryUser(Scanner in) {

// string array for results

String[] s = null;

// Give user instructions

System.out.println("Please select from the following\n" +

"1: Add Faculty using department average salary\n" +

"2: Add Faculty using overall average minus one dept\n" +

"3: Print All Faculty in the Database\n" +

"4: Exit the program\n");

// get rid of empty lines

int i = in.nextInt();

// set a switch for the user input options

switch (i)

{

// case one, populate the array for the three values input

case 1:

in.nextLine();

s = new String[3];

System.out.println("Please input fid\n");

s[0] = in.nextLine();

System.out.println("Please input fname\n");

s[1] = in.nextLine();

System.out.println("Please input deptid\n");

s[2] = in.nextLine();

break;

// case two, populate for the four values given

case 2:

in.nextLine();

s = new String[4];

System.out.println("Please input fid\n");

s[0] = in.nextLine();

System.out.println("Please input fname\n");

s[1] = in.nextLine();

System.out.println("Please input deptid\n");

s[2] = in.nextLine();

System.out.println("Please input deptid to ignore\n");

s[3] = in.nextLine();

break;

// case 3, create size 1 array

case 3:

s = new String[1];

break;

// exit case return null

case 4:

return null;

// default case will result in empty loop

default:

s = new String[2];

}

return s;

}

}

**PROGRAM OUTPUT**

**Add Faculty 1.0  
  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
3  
FID FNAME DEPTID SALARY  
  
101 Johnson 10 55000  
  
102 Lynn 10 65000  
  
103 Lynn 12 30000  
  
104 Black 11 32000  
  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
1  
Please input fid  
  
105  
Please input fname  
  
Smart  
Please input deptid  
  
10  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
3  
FID FNAME DEPTID SALARY  
  
101 Johnson 10 55000  
  
102 Lynn 10 65000  
  
103 Lynn 12 30000  
  
104 Black 11 32000  
  
105 Smart 10 54000  
  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
1  
Please input fid  
  
106  
Please input fname  
  
Nemade  
Please input deptid  
  
12  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
3  
FID FNAME DEPTID SALARY  
  
101 Johnson 10 55000  
  
102 Lynn 10 65000  
  
103 Lynn 12 30000  
  
104 Black 11 32000  
  
105 Smart 10 54000  
  
106 Nemade 12 24000  
  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
1  
Please input fid  
  
107  
Please input fname  
  
Findley  
Please input deptid  
  
11  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
3  
FID FNAME DEPTID SALARY  
  
101 Johnson 10 55000  
  
102 Lynn 10 65000  
  
103 Lynn 12 30000  
  
104 Black 11 32000  
  
107 Findley 11 25600  
  
105 Smart 10 54000  
  
106 Nemade 12 24000  
  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
2  
Please input fid  
  
108  
Please input fname  
  
Varahan  
Please input deptid  
  
10  
Please input deptid to ignore  
  
12  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
3  
FID FNAME DEPTID SALARY  
  
101 Johnson 10 55000  
  
102 Lynn 10 65000  
  
103 Lynn 12 30000  
  
104 Black 11 32000  
  
107 Findley 11 25600  
  
108 Varahan 10 46320  
  
105 Smart 10 54000  
  
106 Nemade 12 24000  
  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
2  
Please input fid  
  
109  
Please input fname  
  
Jackson  
Please input deptid  
  
12  
Please input deptid to ignore  
  
11  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
3  
FID FNAME DEPTID SALARY  
  
101 Johnson 10 55000  
  
102 Lynn 10 65000  
  
103 Lynn 12 30000  
  
104 Black 11 32000  
  
107 Findley 11 25600  
  
108 Varahan 10 46320  
  
109 Jackson 12 45720  
  
105 Smart 10 54000  
  
106 Nemade 12 24000  
  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
2  
Please input fid  
  
110  
Please input fname  
  
Greunwald  
Please input deptid  
  
11  
Please input deptid to ignore  
  
11  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program  
  
3  
FID FNAME DEPTID SALARY  
  
101 Johnson 10 55000  
  
102 Lynn 10 65000  
  
103 Lynn 12 30000  
  
104 Black 11 32000  
  
107 Findley 11 25600  
  
108 Varahan 10 46320  
  
109 Jackson 12 45720  
  
110 Greunwald 11 45720  
  
105 Smart 10 54000  
  
106 Nemade 12 24000  
  
Please select from the following  
1: Add Faculty using department average salary  
2: Add Faculty using overall average minus one dept  
3: Print All Faculty in the Database  
4: Exit the program**

Problem 3)

**PLS FILE:**

create or replace PROCEDURE PROBLEM3 AS

----------------- MODULE 1 -------------------------------------------------

-- Variables and procedure for changing major

id1 NUMBER;

id2 NUMBER;

major1 VARCHAR(20);

major2 VARCHAR(20);

-- Variable for totaling the employee bonuses in part 2

totalBonus NUMBER;

-- This procedure changes a students major or unenrolls a student

PROCEDURE change\_major(studentID IN NUMBER, majorName IN VARCHAR) AS

-- Declarations here

validSnum NUMBER;

BEGIN

-- save the count of student matching the snum, it should be one

select count(\*) into validSnum from Student S where S.snum = studentID;

-- Check for significant input

IF(studentID = 0 or majorName = '0')

THEN RETURN;

END IF;

-- IF namjor name passed in is Null

IF(majorName IS NULL)

-- then delete the Class Enrollments tuple

THEN delete from Enrolled E where E.snum = studentID;

-- and the Student tuple

delete from Student S where S.snum = studentID;

-- otherwise delete the classes associated with the student

ELSE delete from Enrolled E where E.snum = studentID;

-- and update the students major name

update Student S set S.major = majorName where S.snum = studentID;

END IF;

-- End of code for changing major

END change\_major;

----------------- MODULE 2 -------------------------------------------------

-- function for updating faculty bonus and returning the total bonus payout

FUNCTION total\_bonus RETURN NUMBER AS

-- count of classes for a professor

classCount NUMBER;

-- total of all bonuses

total NUMBER;

BEGIN

classCount := 0;

total := 0

-- for every faculty member

FOR someone IN (select \* from faculty)

LOOP

-- find the number of classes they teach and apply the bonus

select count(\*) into classCount from Class C where C.fid = someone.fid;

-- bonus .02 for every class taught

total := total + (someone.salary \* (classCount \* .02));

update Faculty F set F.salary = F.salary + (F.salary \* classCount \* .02) where someone.fid = F.fid;

classCount := 0;

END LOOP;

-- return the total amount

RETURN total;

END total\_bonus;

BEGIN

-- we will set values to test the procedures

id1 := 2;

id2 := 4;

major1 := NULL;

major2 := 'MIS';

-- THIS WILL DELETE SNUM 2 TUPLE

change\_major(id1,major1);

-- THIS WILL DELETE SNUM 4's ENROLLMENTS AND CHANGE MAJOR TO MIS

change\_major(id2,major2);

-- THIS WILL UPDATE FACULTY BONUSES TO THE VALUES PRESCRIBED

totalBonus := total\_bonus();

END PROBLEM3;

**SQL FILE:**

/\*

In this SQL file we will execute our procedure consisting of our

2 modules, please recall that this will delete snum 2 'BETHANY' and

all her enrollments from the database, update snum 4 'CODD' to an MIS

major and remove them from all enrolled classes and update faculty salaries

with bonus to be higher based on their class enrollments.

Please Note that we print relevant tables before and after the procedure.

Comparison will be neccessary, but I will bold the changes in the output

text file.

\*/

select \* from Student;

select \* from Faculty;

select \* from Enrolled;

EXECUTE PROBLEM3;

select \* from Student;

select \* from Faculty;

select \* from Enrolled;

**OUTPUT:**

SNUM SNAME MAJOR SLEVEL AGE  
---------- -------------------- -------------------- ---------- ----------  
 1 Adams History FR 18  
 2 Bethany History FR 20 — THIS WILL BE DELETED  
 3 Adams CS SF 20  
 4 Codd CS SF 22 — THIS WILL BE UPDATED  
 5 Daniels ECE JR 22  
 6 Daniels CS JR 24  
 7 Gordon ECE SR 24  
 8 Smith Physics SR 26  
  
 8 rows selected   
  
  
 FID FNAME DEPTID SALARY  
---------- -------------------- ---------- ----------  
 101 Johnson 10 55000 — ALL SALARIES WILL BE UPDATED  
 102 Lynn 10 65000  
 103 Lynn 12 30000  
 104 Black 11 32000  
  
  
 SNUM CNAME   
---------- --------------------------------------------------  
 1 Intro to Compilers   
 2 Intro to Compilers — THIS WILL BE REMOVED   
 3 Intro to Compilers   
 4 Advanced Java — AND THIS   
 4 Data Networks — AND THIS   
 4 Intro to Compilers — AND THIS   
 5 Data Networks   
 5 Intro to Compilers   
 6 Intro to Compilers   
  
 9 rows selected   
  
  
PL/SQL procedure successfully completed. ————————————————————— PROCEDURE COMPLETED  
  
  
  
 SNUM SNAME MAJOR SLEVEL AGE  
---------- -------------------- -------------------- ---------- ----------  
 1 Adams History FR 18  
 3 Adams CS SF 20  
 4 Codd MIS SF 22 — SNUM 2 is gone and CODD is an MIS major  
 5 Daniels ECE JR 22  
 6 Daniels CS JR 24  
 7 Gordon ECE SR 24  
 8 Smith Physics SR 26  
  
 7 rows selected   
  
  
 FID FNAME DEPTID SALARY  
---------- -------------------- ---------- ----------  
 101 Johnson 10 58300 — FACULTY Salaries have been updated  
 102 Lynn 10 68900  
 103 Lynn 12 31800  
 104 Black 11 32640  
  
  
 SNUM CNAME   
---------- --------------------------------------------------  
 1 Intro to Compilers   
 3 Intro to Compilers — ALL 2 and 4 sum entries have been deleted   
 5 Data Networks   
 5 Intro to Compilers   
 6 Intro to Compilers