**Database Application Development**

**Assignment (25%)**

**Submission:**

***Copy your milestone1 and milestone2 C++ program in to text file and submit your Database Application assignment as***

AS\_Group##.txt (replace ## with your group number. For example AS\_Group01.txt)

You can select any number to put for group number.

Make sure to add all students’ number and name in text file. Only one student from group need to submit the assignment file

Put appropriate comment in your code.

I created self-enroll group under Users and Group on Black Board. You can work up to 3 students in one group. You are allowed to work individually if you want to.

**Milestone 1 (10%)**

**Objective:**

In this assignment, you create a simple HR application using the C++ programming language and Oracle. This assignment helps students learn a basic understanding of application development using C++ programming and an Oracle database.

**Instruction:**

In this assignment, we use the same database that you use for the labs.

**Note**: For each query in your assignment, make sure you handle the errors and display the proper message including the error\_code and the error message.

try{

...

}

catch (SQLException& sqlExcp) {

cout << sqlExcp.getErrorCode() << ": " << sqlExcp.getMessage();

}

***Connecting to an Oracle database from a C++ Program***

In your function ***main()***, create a connection to your database.

First, declare the environment and the connection variables.

Environment\* env = nullptr;

Connection\* conn = nullptr;

Define and initialize the variable to store the username, password, and the host address.

string user = "username";

string pass = "password";

string constr = "myoracle12c.senecacollege.ca:1521/oracle12c";

Use the same Oracle username and password that you use for your labs and assignments.

Create the environment and the connection. Make sure you handle any errors may be thrown as you program is executed.

env = Environment::createEnvironment(Environment::DEFAULT);

conn = env->createConnection(user, pass, constr);

Remember to terminate and close the connection and the environment, when your program terminates.

env->terminateConnection(conn);

Environment::terminateEnvironment(env);

You will implement the following functions:

***int menu(void);***

The ***menu()***function returns an integer value which is the selected option by the user from the menu. This function displays the following menu options:

1. Find Employee
2. Employees Report
3. Add Employee
4. Update Employee
5. Remove Employee
6. Exit

Before printing the menu, display the following title on the screen

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HR Menu \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Prompt the user to enter an option (0 to 5). If the user enters an incorrect option, the user is asked to enter an option again. When the user enters a correct option (0 to 5), the function returns the selected value.

If the user selects 0 (Exit), the program terminates.

Every time you call this function, the menu is displayed and the user is prompted to enter a value between 0 and 5. The function keeps asking the user enter a value until the user enters a correct number.

int findEmployee(Connection \*conn, int employeeNumber, struct Employee \*emp);

This function receives an OCCI pointer (a reference variable to an Oracle database), an integer number as the employee number, and a pointer to a variable of type Employee. The function returns 0 if the employee does not exist. It returns 1 if the employee exits.

To store the employee data in the ***findEmployee()*** function, we use a pointer that refers to a variable of type structure called Employee. The Employee structure has the following members:

struct Employee{

int employeeNumber;

char lastName[50];

char firstName[50];

char extension[10];

char email[100];

char officecode[10];

int reportsTo[100];

char jobTitle[50];

};

The *ReportsTo* member stores the employee ID of the employee who is the manager of the given employee number.

If the employee exists, store the employee data into the members of an Employee variable using the third parameter in the ***findEmployee()*** function which references to that variable of type Employee.

void displayEmployee(Connection \*conn, struct Employee emp);

If the user selects option 1, prompt the user to enter a value for the employee number. Then, call function ***findEmployee()*** to check if the employee with the given employee number exists. If the returning value of function ***findEmployee()*** is 0, display a proper error message.

Sample error message:

Employee 1122 does not exist.

Otherwise, call the function ***displayEmployee()*** to display the employee information.

This function receives a Connection pointer (a reference variable to an Oracle database) and the members of a variable of type Employee and displays all members of the *emp* parameter.

See the following sample output:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HR Menu \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1) Find Employee

2) Employees Report

3) Add Employee

4) Update Employee

5) Remove Employee

0) Exit

Enter an option (0-5): 1

Enter Employee Number: 1002

-------------- Employee Information -------------

Employee Number: 1002

Last Name: Murphy

First Name: Diane

Extension: x5800

Email: dmurphy@classicmodelcars.com

Office Code: 1

Manager ID: 0

Job Title: President

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*HR Menu\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1) Find Employee

2) Employees Report

3) Add Employee

4) Update Employee

5) Remove Employee

0) Exit

Enter an option (0-5): 1

Enter Employee Number: 1078

Employee 1078 does not exist.

void displayAllEmployees(Connection \*conn);

If the user selects option 2 (Employees Report), call function *displayAllEmployees().*

This function receives a pointer of type OCCI Conection (a reference variable to an Oracle database) and displays all employees’ information if exist.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*HR Menu\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1) Find Employee

2) Employees Report

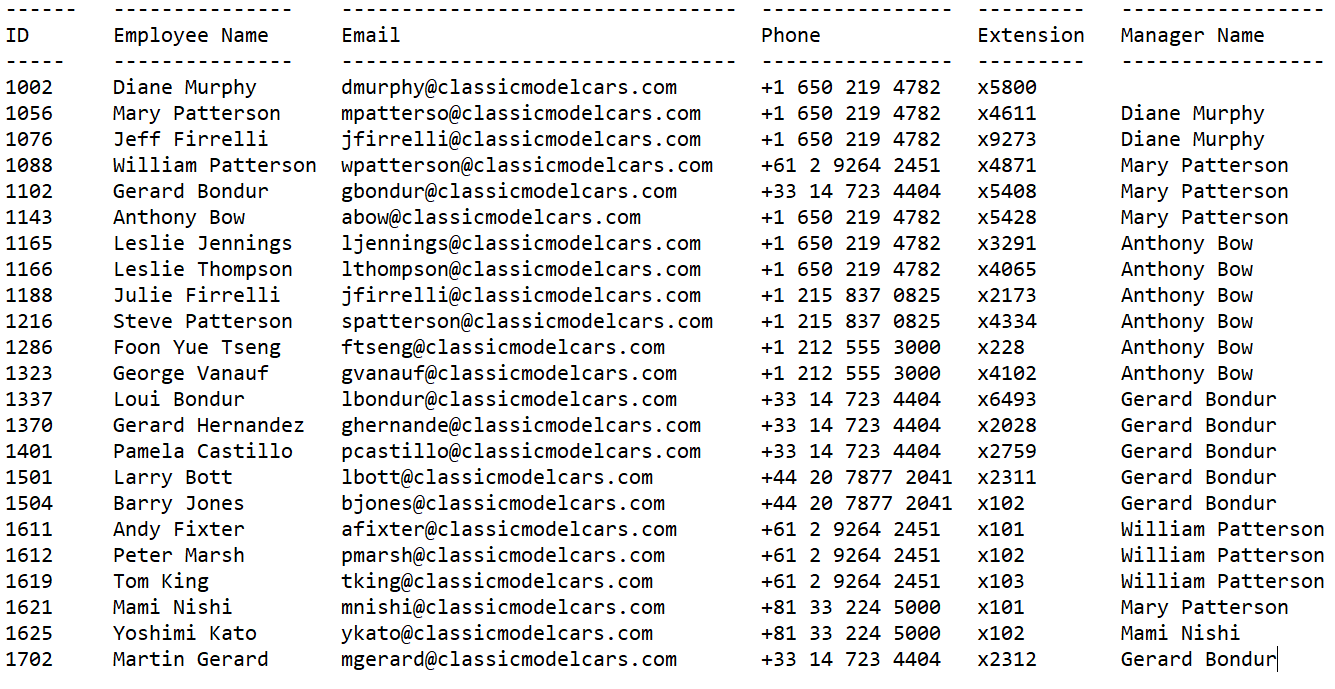
3) Add Employee

4) Update Employee

5) Remove Employee

0) Exit

Enter an option (0-5): 2



**Note**: For this report, you may need to query more than one table (join).

If the query does not return any rows, display a proper message:

There is no employees’ information to be displayed.

**Milestone 2 (15%)**

**Objective:**

In this assignment, you create a simple HR application using the C++ programming language and Oracle. This assignment helps students learn a basic understanding of application development using C++ programming and a Oracle database.

**Instruction:**

In this assignment, we complete the application from the first part to insert, update, and delete the employees’ information.

**Note**: For each query in your assignment, make sure you handle the errors and display the proper message including the error code and the error message.

try{

...

}

catch (SQLException& sqlExcp) {

cout << sqlExcp.getErrorCode() << ": " << sqlExcp.getMessage();

}

You will implement the following functions:

*void getEmployee(struct Employee \*emp);*

This function receives a pointer to a variable of type Employee. This function gets the employee’s information from the user and stores the values in the corresponding members of the structure pointed by the *emp* pointer.

See the following example:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HR Menu \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1) Find Employee

2) Employees Report

3) Add Employee

4) Update Employee

5) Remove Employee

0) Exit

Enter an option (0-5): 3

-------------- New Employee Information -------------

Employee Number: 1099

Last Name: Adam

First Name: Sarah

Extension: x2222

Email: s@email.com

Office Code: 1

Manager ID: 1002

Job Title: Cashier

The new employee is added successfully.

*void insertEmployee(*Connection \*conn, *struct Employee emp);*

This function receives an OCCI pointer (a reference variable to an Oracle database) and a variable of type Employee and inserts the given employee information stored in the parameter ***emp*** to the employees table.

Before you call the function ***insertEmployee()***, first call the function ***getEmployee()*** to get the employee’s information.

In the function ***insertEmployee()***, call ***findEmployee()*** to see whether the employee number of the given employee exists. If an employee with the same employee number exists display a proper message.

**Otherwise**, insert the employee information into the employee table and display a proper message.

For simplicity, for new employees the office code is 1 and the manager (reportsto) is 1002. When asking employee’s info, do not ask the user to enter the office code and the manager ID. Just display these values. The values in red are the values entered by the user.

After executing the statement make sure you execute commit to make all changes permanent and then terminate the statement.

conn->commit();

conn->terminateStatement(stmt);

See the following example:

If the employee number does not exist and the employee info inserted successfully:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HR Menu \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1) Find Employee

2) Employees Report

3) Add Employee

4) Update Employee

5) Remove Employee

0) Exit

Enter an option (0-5): 3

-------------- New Employee Information -------------

Employee Number: 1099

Last Name: Adam

First Name: Sarah

Extension: x2222

Email: s@email.com

Office Code: 1

Manager ID: 1002

Job Title: Cashier

The new employee is added successfully.

If the employee number exists:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HR Menu \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1) Find Employee

2) Employees Report

3) Add Employee

4) Update Employee

5) Remove Employee

0) Exit

Enter an option (0-5): 3

-------------- New Employee Information -------------

Employee Number: 1088

Last Name: Adam

First Name: Sarah

Extension: x2222

Email: s@email.com

Office Code: 1

Manager ID: 1002

Job Title: Cashier

An employee with the same employee number exists.

*void updateEmployee(*Connection \*conn, *int employeeNumber);*

Before you call ***updateEmployee()***, get the employee number from the user.

This function receives an OCCI pointer (a reference variable to an Oracle database) and an integer number as the employee number and updates the phone extension for the given employee. In function ***updateEmployee()***, call function ***findEmployee()*** to see if the employee exists in table employees. If employee does exist, display the employee’s last name and first name and then ask the user to enter the new phone extension. Store the new extension in table employees for the given employee number.

After executing the statement make sure you execute commit to make all changes permanent and then terminate the statement.

conn->commit();

conn->terminateStatement(stmt);

If the employee with the given ID exists:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HR Menu \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1) Find Employee

2) Employees Report

3) Add Employee

4) Update Employee

5) Remove Employee

0) Exit

Enter an option (0-5): 4

Employee Number: 1099

Last Name: Adam

First Name: Sarah

Extension: x6666

The employee's extension is updated successfully.

If the employee with the given ID does not exist:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HR Menu \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1) Find Employee

2) Employees Report

3) Add Employee

4) Update Employee

5) Remove Employee

0) Exit

Enter an option (0-5): 4

Employee Number: 1000

The employee with ID 1000 does not exist.

*void deleteEmployee(*Connection \*conn, *int employeeNumber);*

Before you call ***deleteEmployee()***, get the employee number from the user.

This function receives an OCCI pointer (a reference variable to an Oracle database) and an integer number as the employee number and deletes a row with the given employee number from table employees. In function ***deleteEmployee()***, call function ***findEmployee()*** to see if the employee with the given employee number exists. If the employee does not exist display a proper message.

If the employee exits, delete the row from table employees and display a proper message.

After executing the statement make sure you execute commit to make all changes permanent and then terminate the statement.

conn->commit();

conn->terminateStatement(stmt);

See the following example:

If the employee with the given ID exists:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HR Menu \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1) Find Employee

2) Employees Report

3) Add Employee

4) Update Employee

5) Remove Employee

0) Exit

Enter an option (0-5): 5

Employee Number: 1099

The employee with ID 1099 is deleted successfully.

If the employee with the given ID does not exist:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HR Menu \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1) Find Employee

2) Employees Report

3) Add Employee

4) Update Employee

5) Remove Employee

0) Exit

Enter an option (0-5): 5

Employee Number: 1000

The employee with ID 1000 does not exist.