**Lab 06 – Database Application Development**

**Objective:**

In this lab students learn:

* How to connect to a MySQL server from a C++ program.
* How to write and execute SQL queries in a C++ program.

**Submission:**

***Your submission will be a single text-based .cpp file including your C++ program for the Database Application lab.***

L06\_ID\_LASTNAME.cpp

Your submission needs to be commented

**Lab Instruction:**

***Create MySQL server Connection:***

To connect to the MySQL server from your C++ program in Visual Studio, you need to follow these steps:

First, make sure that the MySQL server is installed on your computer. For this lab, we use MySQL server 8.0.

Open Visual Studio and create a new C++ project.

To connect and work with databases in MySQL server, we use the mysql.h header file or library. To set up the visual configuration, right clique on your project name in the navigation bar and go tap “Properties”:

Project/Properties

In “Configuration Management”, change “Active Solution Configuration” to “release” and “Active Solution Platforms” to “x64” (see Figure 1) and select “Apply”.

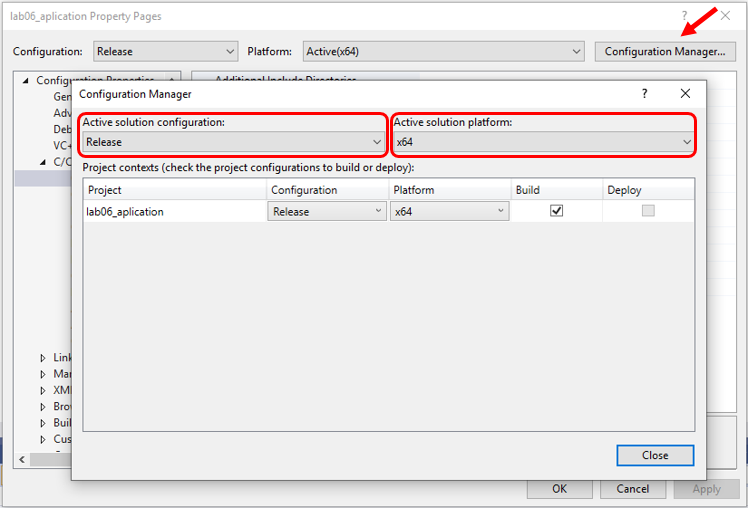


Figure 1

In property Page, Expand C/C++ in the navigation bar. Select “General”. In the general setting, choose “Additional Library Directories”. You need to add the following directories in the given order (Figure 2):

C:\Program Files\MySQL\Connector C++ 8.0\include

C:\Program Files\MySQL\MySQL Server 8.0\include

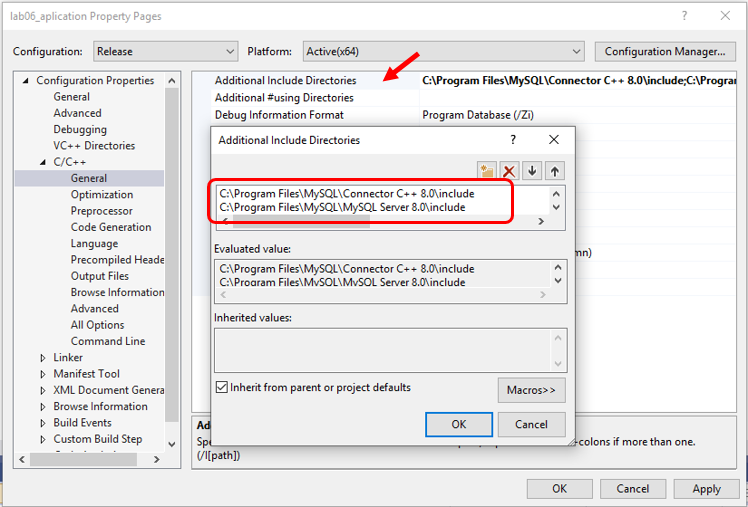


Figure 2

In the navigation bar, expand “Linker”. Select “Additional Library Directories”. Add the following directories in the given order (Figure 3):

C:\Program Files\MySQL\Connector C++ 8.0\lib64\vs14

C:\Program Files\MySQL\MySQL Server 8.0\lib

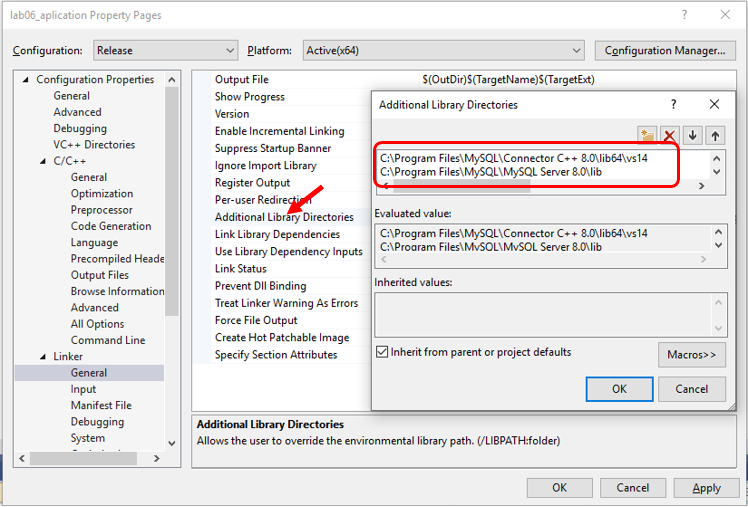


Figure 3

From “Linker”, select “Input”. In “Additional Dependencies”, add the following files (Figure 4):

mysqlcppconn.lib

libmysql.lib

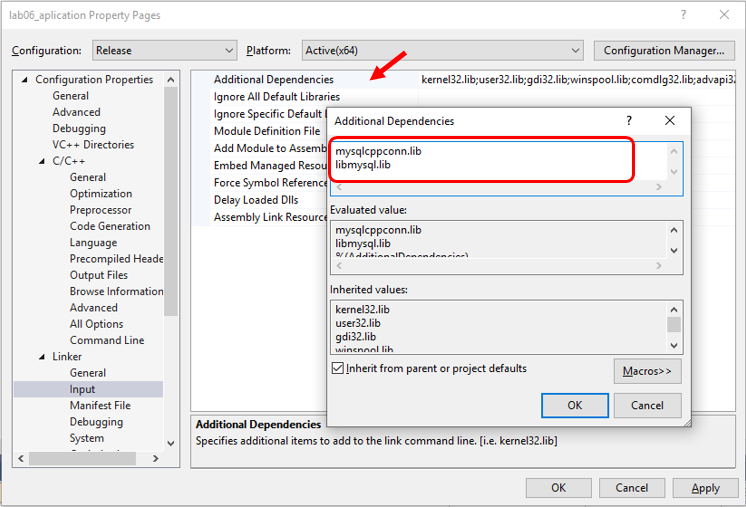


Figure 4

Select “OK” and close the property page.

You also need to copy libmysql.dll from C:\Program Files\MySQL\MySQL Server 8.0\lib to your project directory (folder).

***Connecting to a MySQL database from a C++ Program***

* Create a new C++ project in Visual Studio. Add a source code named databaseConnection.cpp to your project.
* At the beginning of the C++ program, add the mysql.h library.

**#include <mysql.h>**

* To create a connection to your MySQL database, you need to declare a pointer variable to your database.

To define a pointer (reference variable) to the MySQL database:

**MYSQL \*conn\_ptr;**

* To establish a connection to the MySQL server write the following command:

conn = mysql\_real\_connect(conn, "hostname", "username", "password", "database name", 3306, nullptr, 0);

**hostname:** mymysql.senecacollege.ca

**username:** your MySQL username

**password:** you MySQL password

* To check if the connection is successfully established, you need the following code:

if (conn) {

cout << "successful connection to database" << endl;

} else {

cout << "Connection Failed" << mysql\_error(conn) << endl;

}

See the following sample code for establishing a connection to a database:

|  |
| --- |
| #include <mysql.h>  #include <iostream>  using namespace std;  int main(void) {  MYSQL\* conn;  conn = mysql\_init(0);  conn = mysql\_real\_connect(conn, "hostname", "username", "password", "database  name", 3306, nullptr, 0);  if (conn) {  cout << "successful connection to database" << endl;  } else {  cout << "Connection Failed" << mysql\_error(conn) << endl;  }  return 0;  } |
|  |

* If the connection is successfully established, you can execute SQL queries in your C++ program. To execute a query, declare your SQL statement as follows:

string query = "select \* from offices;";

const char\* q = query.c\_str();

* After declaring your SQL query, you can execute it using the mysql\_query() function:

exequery = mysql\_query(conn, q);

This function returns an integer number, so you need to declare an integer variable to store the returning value:

int exequery;

* After calling function mysql\_query(conn,q), we need to make sure if the query has been executed successfully:

if (!exequery) {

//query execution is successful

cout << "The query executed successfully with no error." << endl;

}

else {

//query execution is not successful

cout << "Error message: " << mysql\_error(conn) << endl;

}

* Function mysql\_error(conn) returns the error message if the execution of the given SQL query failed.
* To see the error code, you also can call function mysql\_errorno(conn). This function returns the error code for the most recent function call.

if (!exequery) {

//query execution is successful

cout << "The query executed successfully with no error." << endl;

}

else {

//query execution is not successful

cout << "Error message: " << mysql\_error(conn) << ": "<< mysql\_errno(conn) << endl;

}

**Sample Output:**

successful connection to database

Error message: Table 'office' doesn't exist: 1146

* After successfully executing the query, the query result needs to be stored for fetching the rows:

res = mysql\_store\_result(conn);

* The returning result is data of type structure MYSQL\_RES. As a result, you need to declare a variable of type MYSQL\_RES to store the returning result in advanced before calling function mysql\_store\_result().

MYSQL\_RES\* res;

* After calling function mysql\_store\_result(), you can call mysql\_error() and mysql\_errorno for error handling and displaying proper message if the previous function execution returns an error.
* To fetch the rows, function mysql\_fetch\_row(res) is called. If the result of the query is empty, this function returns null. To check if the result set is empty you examine the returning value:

if (mysql\_fetch\_row(res) == nullptr){

cout << "The result is empty." << endl;

}

* Every time you execute function mysql\_fetch\_row(), if the result is not empty, the function returns the current row and moves the control to the next one. So, executing the above code fetches the current row which for the first time is the first row of the result set. To store the values in each row, you need to declare a variable of type MYSQL\_ROW:

MYSQL\_ROW row;

All rows in the result set can be fetch using a loop.

while (row = mysql\_fetch\_row(res)) {

printf("OfficeCode: %s, city: %s\n", row[0], row[1]);

}

The loop iterates until this function returns null. The null value means that there is no more rows to be fetched.

If the result set is empty, the loop will be never executed.

* The last function to be called is mysql\_close(conn). You call this function when the connection to the database is no longer required.

mysql\_close(conn);

**Lab Requirements:**

In this lab, you need to write a C++ program to execute the following queries and display the result returned by each query. For the output format, see the sample output.

* 1. Display Employee Number, First Name, Last Name, Phone Number, and Extension of all Employees who work in San Francisco. See the following Sample output.
  2. Display Employee Number, Last Name, Phone Number, and Extension for all managers. (You can use column reportsto to find the managers’ employee numbers)

Sample output:

