BUỔI 3 - ĐỒ ÁN MÔN HỌC

3.1. Mục tiêu

- Tìm hiểu yêu cầu của đồ án.
- Giới thiệu một số cơ sở lý thuyết để thực hiện đồ án (nối kết đến mysql bằng Java/gcc/MS Visual C)

3.2. Giới thiệu về đồ án

3.2.1. Yêu cầu của đồ án

Mục tiêu của đồ án là nhằm minh họa vai trò của DBMS trong một hệ thống thông tin. Cho sinh viên thấy được một số ưu điểm của việc sử dụng hệ quản trị CSDL trong quản lý và truy xuất dữ liệu so với việc sử dụng hệ thống tập tin thông thường. Ngoài ra, còn nhằm kiểm tra việc vận dụng một số tính năng nâng cao của các hệ quản trị CSDL như stored procedure, function, trigger, transaction,…

Để đạt được các mục tiêu trên, các yêu cầu cơ bản đối với đồ án như sau:

- Viết một chương trình ứng dụng (bằng một ngôn ngữ bất kỳ như Java, C, PHP,…) có nối kết vào MySQL để quản lý dữ liệu.
- Chương trình có sử dụng các tính năng của MySQL như stored procedure, function, trigger, transaction,…
- Giao diện chương trình có thể là giao diện đồ họa hoặc giao diện text (console) hoặc di động,…

3.2.2. Một số đề tài (nhưng không giới hạn)

1) Viết chương trình cài đặt một số nghiệp vụ ngân hàng như: mở tài khoản, chuyển tiền (ATM), rút tiền,…
2) Viết ứng dụng bán hàng trực tuyến với các chức năng như đặt hàng online, quản lý hàng hóa,…
3) Viết ứng dụng book vé online: máy bay, khách sạn, xe.
4) Viết ứng dụng quản lý đơn giản như quản lý sinh viên, quản lý kho hàng,…

3.2.3. Qui trình thực hiện đề tài

1) Phân tích yêu cầu, xác định và mô tả chức năng của hệ thống.
2) Thiết kế CSDL (mô hình E-R, mô hình vật lý).
3) Thiết kế giải thuật, sơ đồ luồng xử lý (workflow) cho các chức năng của hệ thống.
4) Cài đặt và kiểm thử.

3.3. Nối kết và truy xuất MySQL bằng Java

Phương pháp nối kết MySQL trong Java được sử dụng rộng rãi nhất là sử dụng **JDBC** (Java Database Connectivity), một giao diện lập trình ứng dụng viết bằng ngôn ngữ Java

để nối kết và truy xuất các hệ quản trị CSDL, hoặc các thư viện thuần Java như MySQL Connector/J driver.

Trong phần này, sẽ giới thiệu phương pháp nối kết MySQL từ Java sử dụng MySQL **Connector/J**. Để sử dụng phương pháp nối kết này, đầu tiên ta phải download gói này tại địa chỉ sau: http://dev.mysql.com/downloads/connector/j

3.3.1. Tạo nối kết

Việc tạo nối kết đến MySQL bao gồm 4 bước:

1) Import gói **java.sql.***
2) Đăng ký driver (khởi tạo một đối tượng mới của lớp "com.mysql.jdbc.Driver", dùng lớp **Class** của Java)
3) Tạo chuỗi nối kết
4) Tạo đối tượng nối kết **Connection** (hàm **getConnection()** của lớp **DriverManager**)

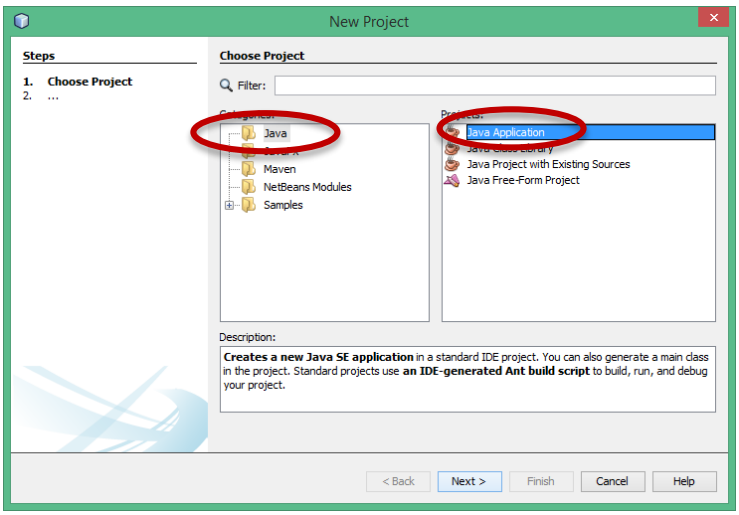
Ví dụ:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

Connection conn = null;
try {
    Class.forName("com.mysql.jdbc.Driver").newInstance();
    conn = DriverManager.getConnection("jdbc:mysql://<address>/<db_name>?" +
                                       "user=<username>&password=<pass>");

    //thực hiện các truy vấn dữ liệu...
    ...
}
catch (SQLException ex) {    //xử lý ngoại lệ nếu có
    ex.printStackTrace();
}
```

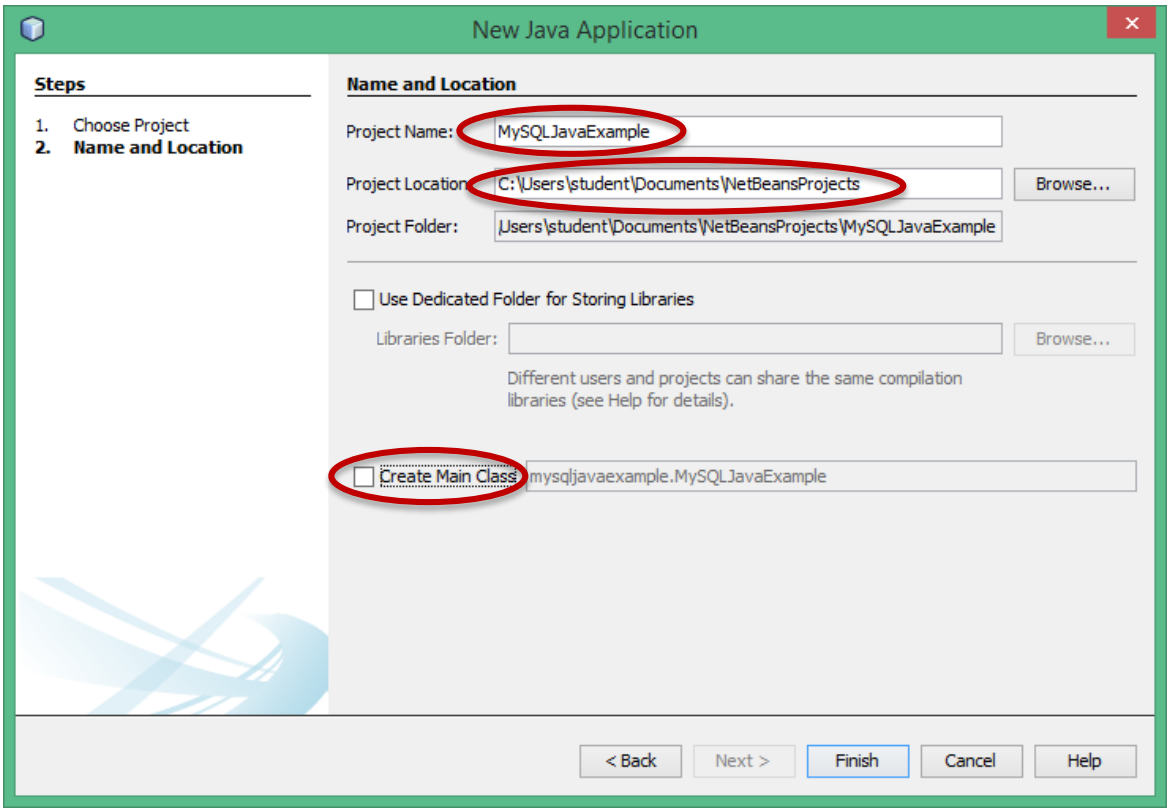
Trong đó, **<address>** là địa chỉ của MySQL server, **<db_name>** là tên CSDL cần nối kết, **<username>** là tên người dùng và **<pass>** là mật khẩu của người dùng.

Sau đây sẽ hướng dẫn chi tiết các bước viết một chương trình Java bằng Netbeans để nối kết đến MySQL:

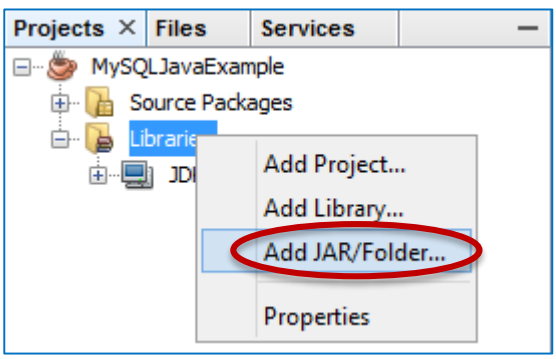
1) Tạo 1 project kiểu "Java Application" trong Netbeans:



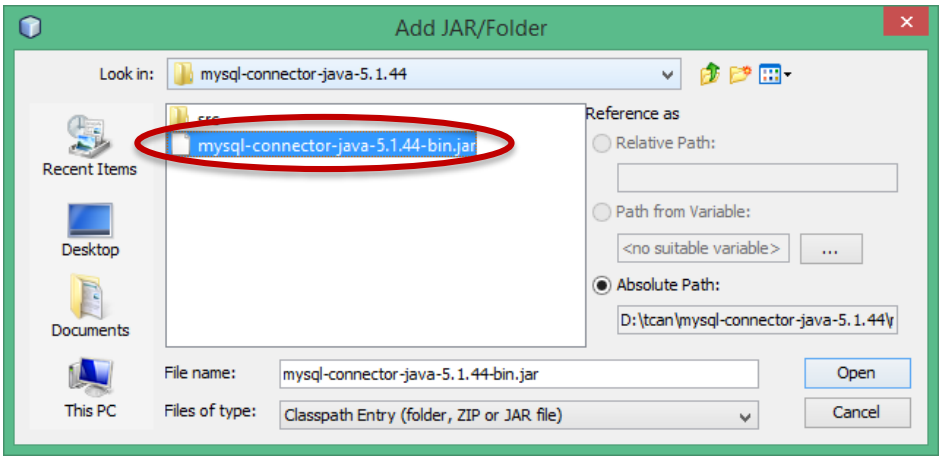
2) Đặt tên project và nơi lưu trữ project. Lưu ý, bỏ chọn "Create Main Class":



3) Khai báo sử dụng thư viện Connector/J của MySQL:



Sau khi click "Add JAR/Folder...", chọn tập tin "mysql-connector-java-5.1.44-bin.jar" trong thư mục của thư viện Connector/J (download từ trang web của MySQL)



4) Tạo 1 lớp mới có tên là "MySQLConnect" và nhập vào nội dung cho lớp trên như sau:

```
import java.sql.Connection;
import java.sql.DriverManager;

public class MySQLConnect {
  public static void main(String args[]) {
    Connection conn = null;

    try {
      Class.forName("com.mysql.cj.jdbc.Driver").newInstance();

      conn = DriverManager.getConnection("jdbc:mysql://localhost/test?"
                                         + "user=root");

      System.out.println("Noi ket thanh cong");
    } catch (Exception ex) { //xử lý ngoại lệ nếu có
      System.out.println("Noi ket khong thanh cong");
      ex.printStackTrace();
    }
  }
}
```

5) Thực thi chương trình trên. Lưu ý, trước khi thực thi phải khởi động MySQL Server trước.

3.3.2. Thực thi câu lệnh SQL

Có 3 cách để thực thi câu lệnh SQL: i) sử dụng lớp **`java.sql.Statement`**, ii) sử dụng lớp **`java.sql.PreparedStatement`**, và iii) sử dụng lớp **`java.sql.CallableSatement`**. Để nhận kết quả trả về từ câu truy vấn, ta dùng lớp **`java.sql.ResultSet`**.

Ví dụ sau minh họa cách sử dụng lớp **`Statement`** và **`ResultSet`** để thực thi và lấy kết quả trả về từ câu truy vấn:

```
import java.sql.Connection;
```

```java
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;

//giả sử nối kết đến MySQL đã được thực hiện (xem VD trên)

Statement stmt = null;
ResultSet rs = null;
try {
    stmt = conn.createStatement();

    //dùng phương thức executeQuery để yêu cầu thực hiện lệnh SQL
    rs = stmt.executeQuery("SELECT foo FROM bar");

    //hoặc kết hợp phương thức execute và getResultSet như sau
    if (stmt.execute("SELECT foo FROM bar")) {
        rs = stmt.getResultSet();
    }

    //thao tác trên tập kết quả trả về rs....
}
catch (SQLException ex){    //xử lý ngoại lệ
    System.out.println("SQLException: " + ex.getMessage());
}
finally {
    //giải phóng tài nguyên khi không sử dụng nữa
    if (rs != null) {
        try {
            rs.close();
        } catch (SQLException sqlEx) { } //đoạn mã xử lý ng/lệ

        rs = null;
    }

    if (stmt != null) {
        try {
            stmt.close();
        } catch (SQLException sqlEx) { } //đoạn mã xử lý ng/lệ

        stmt = null;
    }
}
```

Ví dụ sau sử dụng lớp **`PreparedStatement`** để thực hiện các câu lệnh SQL:

```java
PreparedStatement pStmt = null;
ResultSet rs = null;

//thực hiện câu lệnh truy vấn
pStmt = conn.prepareStatement("SELECT * FROM Accounts WHERE username=?");
pStmt.setString(1, "jones");  //thiết đặt giá trị cho tham số thứ 1
rs = pStmt.executeQuery();

//thực hiện câu lệnh INSERT
pStmt = conn.prepareStstement("INSERT INTO Accounts VALUE (?, ?, ?)");
```

```
pStmt.setString(1, "tcan");
pStmt.setString(2, "Tran Cong An");
pStmt.setInt(3, 500);
pStmt.executeUpdate();
```

3.3.3. Gọi hàm và thủ tục với `CallableStatement`

Để gọi hàm hoặc thủ tục, ta thường sử dụng lớp **`CallableStatement`**. Ví dụ sau minh họa cách gọi thủ tục và lấy giá trị tham số OUT từ thủ tục:

```
CallableStatement cStmt = null;

//call stored procedure "getBalance(IN username varchar(8), OUT balance int)"
cStmt = conn.prepareCall("{call getBalance(?, ?)}");
cStmt.setString(1, "jones");
cStmt.registerOutParameter(2, Types.Int);  //đăng ký tham số OUT

cStmt.executeQuery();  //gọi thủ tục

Int balance = cStmt.getInt(2);  //lấy giá trị tham số OUT balance
```

Nếu thủ tục có trả kết quả trả về cho lời gọi thủ tục, ta nhận kết quả trả về như sau:

```
CallableStatement cStmt = null;

//gọi hàm SV_LOAI(IN mssv char(10)),
//hàm trả về DS các SV tốt nghiệp loại được truyền vào cho đối số
cStmt = con.prepareCall("{call SV_LOAI(?)}");
cStmt.setString(1, "gioi");

ResultSet rs = cStmt.executeQuery();

System.out.println("Danh sách sinh viên loại giỏi:");
while (rs.next()) {
    String mssv = rs.getString("mssv");        //resultset trả về gồm 2 trường là
    String hoten = rs.getString("hoTen");      //MSSV và họ tên
    System.out.println(mssv + "\t " + hoten);
}
```

3.3.4. Tài liệu tham khảo

Một số nguồn tài liệu tham khảo có liên quan đến việc truy xuất MySQL từ Java:

5) JDBC basics (Oracle Java documentation): http://docs.oracle.com/javase/tutorial/jdbc/basics/index.html
6) JDBC concepts (MySQL Documentation): https://dev.mysql.com/doc/connector-j/5.1/en/connector-j-usagenotes-basic.html

7) MySQL and Java JDBC Tutorial (Vogella): http://www.vogella.com/tutorials/MySQLJava/article.html
8) Using JDBC with GUI API (Oracle Java Documentation): https://docs.oracle.com/javase/tutorial/jdbc/basics/jdbcswing.html

3.4. Nối kết và truy xuất MySQL bằng ngôn ngữ C

Một trong những phương pháp phổ biến nhất để nối kết vào MySQL bằng ngôn ngữ C/C++ là sử dụng thư viện **MySQL Connector/C** (cho ngôn ngữ C) hoặc **MySQL Connector/C++** (cho ngôn ngữ C++).

3.4.1. Nối kết đến MySQL với gcc/Linux

Sau đây là một ví dụ nối kết đến MySQL server bằng ngôn ngữ C với trình biên dịch **gcc**:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <mysql.h>

int main() {

    printf("MySQL client version: %s\n", mysql_get_client_info());

    MYSQL mysql;
    MYSQL_RES *result;
    MYSQL_ROW row;

    char query_str[1000];
    my_ulonglong numRows;
    unsigned int numFields;

    //init mysql connection
    if (mysql_init(&mysql) == NULL) {
        printf("Failed to init mysql\n");
        exit(1);
    }
    else
        printf("Init mysql successfully\n");

    //connect to mysql server
    if (!mysql_real_connect(&mysql, "localhost", "root", "", NULL, 0, NULL, 0)) {
        printf("Cannot connect to mysql server: %s\n", mysql_error(&mysql));
        exit(1);
    }
    else
        printf("Connected\n");

    //select database
    if (mysql_select_db(&mysql, "dbms") == 0)
        printf("Database 'DBMS' selected\n");
    else
        printf("Failed to select database 'DBMS': %s\n", mysql_error(&mysql));

    //make a query
```

```
    strcpy(query_str, "SELECT * FROM sinhvien");
    if (mysql_real_query(&mysql, query_str, strlen(query_str)) == 0) {  //success

        //get the result
        result = mysql_store_result(&mysql);


        if (result) {
            numRows = mysql_num_rows(result);
            numFields = mysql_num_fields(result);
            printf("%llu records found\n", numRows);
            printf("Number of fields: %u\n", numFields);

            //fetch every rows in the result and display to the screen
            while ((row = mysql_fetch_row(result))) {
                for (int i=0; i<numFields; i++) {
                    printf("%s\t", row[i] ? row[i] : "NULL");
                }
                printf("\n");
            }
            mysql_free_result(result);
        }
        else {
            printf("Error in getting records: %s\n", mysql_error(&mysql));
        }
    }
    else {
        printf("Query failed: %s\n", mysql_error(&mysql));
    }
    mysql_close(&mysql);

    return 0;
}
```

Cú pháp để biên dịch chương trình trên bằng **gcc** (giả sử chương trình trên được lưu với tên là **"mysql_c.c"**):

gcc mysql_c.c -o mysql_c `mysql_config --cflags --libs`

Điều kiện để có thể biên dịch được chương trình trên là:

1) Đã cài **MySQL Connector/C** (connector này được cài kèm theo MySQL server) (download link: https://dev.mysql.com/downloads/connector/c/6.0.html)
2) Đường dẫn đến thư mục chứa chương trình **mysql_config** được thêm vào biến môi trường **PATH** (thường là thư mục **bin** trong thư mục cài đặt MySQL)

Nếu dòng lệnh trên biên dịch không thành công thì thay **`mysql_config --cflags --libs`** bằng **-I<đường dẫn đến thư mục chứa mysql.h> -L<đường dẫn đến thư mục lib của MySQL> -lmysqlclient**.

3.4.2. Nối kết đến MySQL với VC/Windows

Chương trình sau đây sẽ nối kết đến **mysql** và hiển thị version của MySQL server ra màn hình:

```c
#include <stdio.h>
#include <windows.h>
#include <mysql.h>

MYSQL *conn;
int version = 1;

int main ( int argc, char *argv[] )
{
    conn = mysql_init ( NULL );
    mysql_real_connect ( conn, "localhost", "root",
            "password", "test", 3308, NULL, 0 );
       version = mysql_get_server_version( conn );
       printf("\nMySQL Version = %d\n",version);
    mysql_close ( conn );
    return 0;
}
```

Để biên dịch chương trình trên với Visual C bằng dòng lệnh, thực hiện các bước sau (giả sử chương trình trên được lưu với tên **mysql_vc.c** trong thư mục **c:\dbms**)

1) Copy hai tập tin **libmysql.dll** và **libmysql.lib** vào thư mục **c:\dbms**
2) Chuyển vào thư mục **c:\dbms** (**cd c:\dbms**)
3) Biên dịch chương trình bằng lệnh **cl**:

   ```
   cl /I "C:\Program Files (x86)\Microsoft Visual Studio
   10.0\VC\include" /I "C:\Program Files (x86)\MySQL\mysql-5.5.30-
   win32\include" /I "C:\Program Files (x86)\Microsoft
   SDKs\Windows\v7.0A\Include" /MD "C:\Program Files (x86)\MySQL\mysql-
   5.5.30-win32\lib\libmysql.lib" mysql_vc.c
   ```

   Lưu ý, thay các đường dẫn trong lệnh trên bằng các đường dẫn thích hợp của hệ thống. Nếu thành công thì kết quả sẽ như sau:

   ```
   Microsoft (R) Incremental Linker Version 10.00.30319.01
   Copyright (C) Microsoft Corporation.  All rights reserved.

   /out:mysql_vc.exe
   "C:\Program Files (x86)\MySQL\mysql-5.5.30-win32\lib\libmysql.lib"
   mysql1.obj
   ```

4) Thực thi chương trình **mysql_vc.exe** để xem kết quả:

   ```
   c:\dbms\mysql_vc.exe
   MySQL Version = 50532
   ```

Một số lỗi thường gặp khi biên dịch chương trình trên với VC:

1) Không tìm thấy tập tin **mysql.h**:

   ```
   mysql1.c
   ```

```
mysql1.c(2) : fatal error C1083: Cannot open include file: 'mysql.h':
No such file or directory
```

Cách giải quyết: Thêm vào thông số sau trong câu lệnh biên dịch (thay đổi đường dẫn cho phù hợp):

```
/I "C:\Program Files\MySQL\MySQL Server 5.5\include" mysql1.c
```

2) Lỗi cú pháp C2061: syntax error : identifier 'SOCKET'

Nguyên nhân: thiếu lệnh **`#include <windows.h>`**

Hiện tượng: thông báo lỗi tương tự như sau

```
mysql1.c
c:\..\mysql_com.h(291) : error C2061: syntax error : identifier 'SOCKET'
c:\..\mysql_com.h(337) : error C2059: syntax error : '}'
c:\..\mysql_com.h(451) : error C2143: syntax error : missing ')' before '*'
c:\..\mysql_com.h(451) : error C2143: syntax error : missing '{' before '*'
c:\..\mysql_com.h(451) : error C2371: 'Vio' : redefinition; different basic types
...
C:\..\mysql.h(374) : error C2143: syntax error : missing ')' before '*'
C:\..\mysql.h(374) : error C2143: syntax error : missing '{' before '*'
C:\..\mysql.h(374) : error C2059: syntax error : ')'
C:\..\mysql.h(375) : error C2143: syntax error : missing ')' before '*'
C:\..\mysql.h(375) : error C2143: syntax error : missing '{' before '*'
C:\..\mysql.h(375) : fatal error C1003: error count exceeds 100; stopping
compilation
```

Cách giải quyết: thêm lệnh **`#include <windows.h>`** vào giống như chương trình demo

3) Lỗi không tìm thấy tập tin **`windows.h`**

```
mysql1.c
mysql1.c(2) : fatal error C1083: Cannot open include file: 'windows.h':
No such file or directory
```

Cách giải quyết: bổ sung đường dẫn đến tập tin **`windows.h`** trong câu lệnh biên dịch chương trình. Ví dụ:

```
cl /I "C:\Program Files (x86)\Microsoft Visual Studio 10.0\VC\include"
/I "C:\Program Files\MySQL\MySQL Server 5.5\include"
/I "C:\Program Files (x86)\Microsoft SDKs\Windows\v7.0A\Include"
/MD "C:\Program Files\MySQL\MySQL Server 5.5\lib\libmysql.lib" mysql1.c
```

4) Bị lỗi trong bước liên kết được các thư viện:

Hiện tượng: thông báo lỗi tương tự như sau

```
mysql1.obj : error LNK2019: unresolved external symbol
_mysql_close@4 referenced in function _main
mysql1.obj : error LNK2019: unresolved external symbol
_mysql_real_connect@32 referenced in function _main
mysql1.obj : error LNK2019: unresolved external symbol
_mysql_init@4 referenced in function _main
mysql1.exe : fatal error LNK1120: 3 unresolved externals
```

Nguyên nhân: không tương thích giữa phiên bản MySQL (32bit hay 64bit) với các tập tin header của VC.

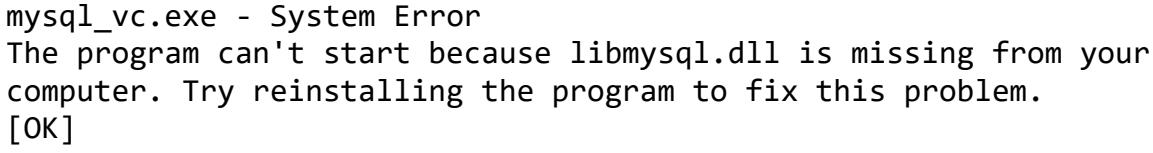
Cách giải quyết: sửa đường dẫn đến các tập tin header của MySQL cho phù hợp với phiên bãn 3 của VC.

```
cl /I "C:\Program Files (x86)\Microsoft Visual Studio 10.0\VC\include"
/I "C:\Program Files (x86)\MySQL\mysql-5.5.30-win32\include"
/I "C:\Program Files (x86)\Microsoft SDKs\Windows\v7.0A\Include"
/MD "C:\Program Files (x86)\MySQL\mysql-5.5.30-win32\lib\libmysql.lib"
mysql1.c
```

5) Chương trình không thể thực thi do thiếu **`libmysql.dll`**:

   Hiện tượng: khi chạy chương trình sẽ hiện thông báo lỗi như sau

   ```
   mysql_vc.exe
   ```

   Cách giải quyết: copy hai tập tin **`libmysql.dll`** và **`libmysql.lib`** vào thư mục chứa chương trình demo

Một số lỗi khác có thể tham khảo thêm tại: http://www.chriscalender.com/common-errors-and-resolutions-for-building-your-own-mysql-or-mariadb-cc-program-on-windows/

3.4.3. Tài liệu tham khảo

Một số nguồn tài liệu tham khảo có liên quan đến việc truy xuất MySQL từ C:

1) MySQL Connector/C++ Developer Guide (MySQL Documentation): https://dev.mysql.com/doc/connector-cpp/en/connector-cpp-examples-complete-example-1.html
2) A simple c++ application that connects to MySQL database (Windows): https://www.youtube.com/watch?v=rTY0mW7IvgI
3) MySQL 5th edition, Chapter 5 – Writing MySQL programs using C (gcc/Linux): http://www.kitebird.com/mysql-book/
4) C sample transaction for mysql (Stackoverflow): https://stackoverflow.com/questions/11526369/c-sample-transaction-for-mysql

5) Giải quyết lỗi không thể load **`libmysqlclient.18.dylib`**: https://stackoverflow.com/questions/10557507/rails-mysql-on-osx-library-not-loaded-libmysqlclient-18-dylib
6) MySQL C API programming tutorial (zetcode): http://zetcode.com/db/mysqlc/
7) MySQL Connector/C developer guide (Windows/Linux): https://downloads.mysql.com/docs/connector-c-en.pdf
8) Building C API client program (MySQL Documentation): https://dev.mysql.com/doc/refman/5.6/en/c-api-building-clients.html
9) Creating a basic C/C++ Program to Interact with MySQL and MariaDB: http://www.chriscalender.com/creating-a-basic-cc-program-to-interact-with-mysql-and-mariadb/
10) Common Errors and Resolutions for Building your own MySQL or MariaDB C/C++ Program on Windows: http://www.chriscalender.com/common-errors-and-resolutions-for-building-your-own-mysql-or-mariadb-cc-program-on-windows/

Phụ lục – Một số hàm cơ bản của MySQL Connector/C

Function	**Description**
my_init()	Initialize global variables, and thread handler in thread-safe programs
mysql_affected_rows()	Returns the number of rows changed, deleted, or inserted by the last UPDATE, DELETE, or INSERT query
mysql_autocommit()	Toggles autocommit mode on/off
mysql_change_user()	Changes user and database on an open connection
mysql_character_set_name()	Return default character set name for current connection
mysql_close()	Closes a server connection
mysql_commit()	Commits the transaction
mysql_connect()	Connects to a MySQL server (this function is deprecated; use mysql_real_connect() instead)
mysql_create_db()	Creates a database (this function is deprecated; use the SQL statement CREATE DATABASE instead)
mysql_data_seek()	Seeks to an arbitrary row number in a query result set

Function	Description
mysql_drop_db()	Drops a database (this function is deprecated; use the SQL statement DROP DATABASE instead)
mysql_errno()	Returns the error number for the most recently invoked MySQL function
mysql_error()	Returns the error message for the most recently invoked MySQL function
mysql_escape_string()	Escapes special characters in a string for use in an SQL statement
mysql_fetch_field()	Returns the type of the next table field
mysql_fetch_field_direct()	Returns the type of a table field, given a field number
mysql_fetch_fields()	Returns an array of all field structures
mysql_fetch_lengths()	Returns the lengths of all columns in the current row
mysql_fetch_row()	Fetches the next row from the result set
mysql_field_count()	Returns the number of result columns for the most recent statement
mysql_free_result()	Frees memory used by a result set
mysql_get_character_set_info()	Return information about default character set
mysql_get_client_info()	Returns client version information as a string
mysql_get_client_version()	Returns client version information as an integer
mysql_get_host_info()	Returns a string describing the connection
mysql_get_option()	Returns the value of a mysql_options() option
mysql_get_server_info()	Returns the server version number
mysql_get_server_version()	Returns version number of server as an integer
mysql_info()	Returns information about the most recently executed query
mysql_init()	Gets or initializes a MYSQL structure

Function	Description
mysql_list_dbs()	Returns database names matching a simple regular expression
mysql_list_fields()	Returns field names matching a simple regular expression
mysql_list_tables()	Returns table names matching a simple regular expression
mysql_more_results()	Checks whether any more results exist
mysql_next_result()	Returns/initiates the next result in multiple-result executions
mysql_num_fields()	Returns the number of columns in a result set
mysql_num_rows()	Returns the number of rows in a result set
mysql_options()	Sets connect options for mysql_real_connect()
mysql_query()	Executes an SQL query specified as a null-terminated string
mysql_real_connect()	Connects to a MySQL server
mysql_rollback()	Rolls back the transaction
mysql_row_seek()	Seeks to a row offset in a result set, using value returned from mysql_row_tell()
mysql_row_tell()	Returns the row cursor position
mysql_select_db()	Selects a database
mysql_server_end()	Finalize the MySQL C API library
mysql_server_init()	Initialize the MySQL C API library
mysql_set_character_set()	Set default character set for current connection
mysql_set_server_option()	Sets an option for the connection (like multi-statements)
mysql_sqlstate()	Returns the SQLSTATE error code for the last error
mysql_shutdown()	Shuts down the database server

Trình tự xem thực hành

Buổi 1 – Giới thiệu MySQL

- Giới thiệu một số đăng điểm của MySQL: xem trong tài liệu
- Cài đặt MySQL: giới thiệu 2 cách
    - Sử dụng không cần cài đặt (minh họa chạy mySQL trong XAMPP)
    - Sử dụng bộ cài đặt: chỉ cần giới thiệu nơi download (search trên Google) và nói một số đặc điểm của bộ cài đặt chứ không cần cài đặt thử, nếu SV muốn thì tự cài).
- Giới thiệu các cách nối kết vào MySQL:
    - Từ công cụ dòng lệnh (mySQL)
    - Từ công cụ đồ họa: chỉ SV nơi download và cài đặt, sử dụng MySQL Workbench (tạo nối kết, nối kết, thực hiện câu truy vấn theo nhiều chế độ,…)
    - Lưu ý với SV:
        - Các máy tính trong phòng thực hành chỉ cài được version **5.x** trở về trước.
        - Chương trình này thỉnh thoảng bị lỗi (nhất là đang mở chương trình mà không sử dụng một khoảng thời gian) và cách tốt nhất là cài đặt lại (không cần uninstall).
- Giới thiệu sơ các lệnh cơ bản của MySQL (xem trong tài liệu). Chỉ cần giới thiệu sơ qua vì SV đã học các lệnh này rồi. Tập trung giải thích các kiểu dữ liệu vì mỗi hệ quản trị có kiểu dữ liệu khác nhau.
- Giải thích các loại ràng buộc cho SV. Lưu ý với sinh viên là MySQL cho phép tạo ràng buộc CHECK nhưng không kiểm tra ràng buộc này.
- Cho SV tự coi hướng dẫn và làm các bài tập
- Hướng dẫn giải một số bài tập

Buổi 2 – MySQL nâng cao

Buổi 3 – Project kickoff