

UNIVERSIDAD MARIANO GÁLVEZ DE GUATEMALA

CENTRO UNIVERSITARIO EL NARANJO

FACULTAD DE INGENIERIA DE SISTEMAS DE
INFORMACIÓN

Programación #1

ING. Alan G. Ucelo Morán



Laboratorio 6#

NOMBRES:KEVIN ALEXANDER MAZARIEGOS PINEDA

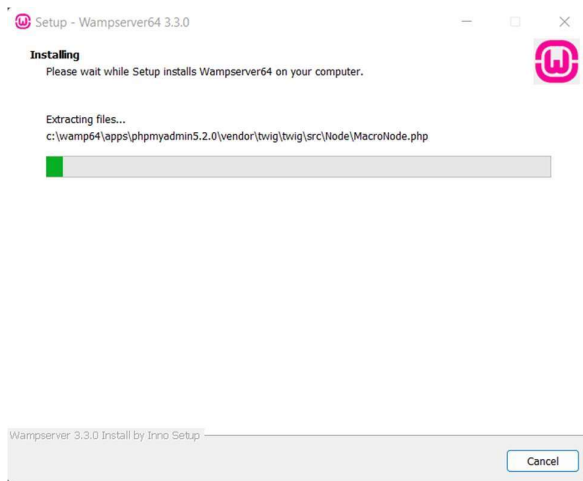
CARNÉ: 9390-22-5048

27 de Mayo. de 23

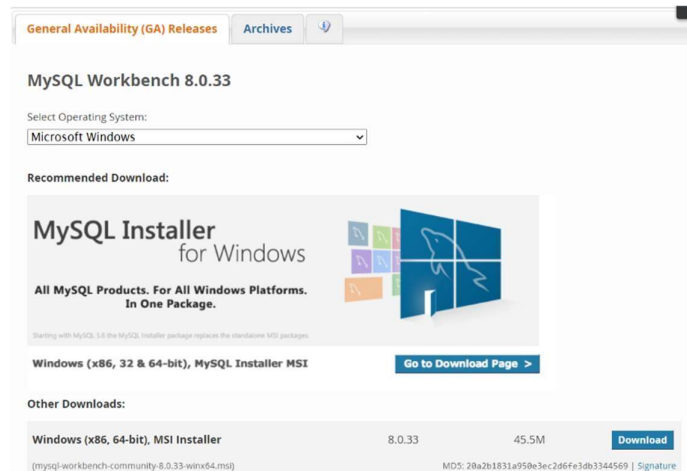
C++ y MYSQL conexión

Para realizar una correcta conexión primero tenemos que descargar e instalar los paquetes a utilizar los cuales son los siguientes

- Wampserver: con este servidor tendremos acceso al gestor de base de datos “MYSQL” entre otras herramientas


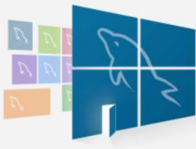


- Workbench: este será el gestor que utilizaremos para la creación, edición y eliminación de base de datos



- **MYSQL conexión c/c++:** esta es una librería "drivers" que se utilizara para conexión de MYSQL con c++

Recommended Download:

MySQL Installer for Windows

All MySQL Products. For All Windows Platforms. In One Package.

Starting with MySQL 5.6 the MySQL Installer package replaces the standalone MSI packages.

Windows (x86, 32 & 64-bit), MySQL Installer MSI [Go to Download Page >](#)

Other Downloads:

Download	Version	Size	Action
Windows (x86, 32-bit), MSI Installer <small>(mysql-connector-c++-8.0.33-win32.msi)</small>	8.0.33	29.7M	Download
<small>MD5: 81636abd24d8da2bd3cc6086c5f7c187 Signature</small>			
Windows (x86, 64-bit), MSI Installer <small>(mysql-connector-c++-8.0.33-win64.msi)</small>	8.0.33	34.2M	Download
<small>MD5: ca3928090fc1009788ef5a93b87069e8 Signature</small>			

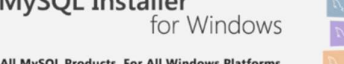
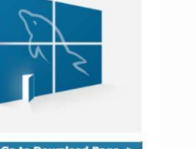
- **MSQL server:** esta es otra librería que se usara como el motor principal de la conexión

MySQL Community Server 8.0.33

Select Operating System:

[Looking for previous GA versions?](#)

Recommended Download:

MySQL Installer for Windows

All MySQL Products. For All Windows Platforms. In One Package.

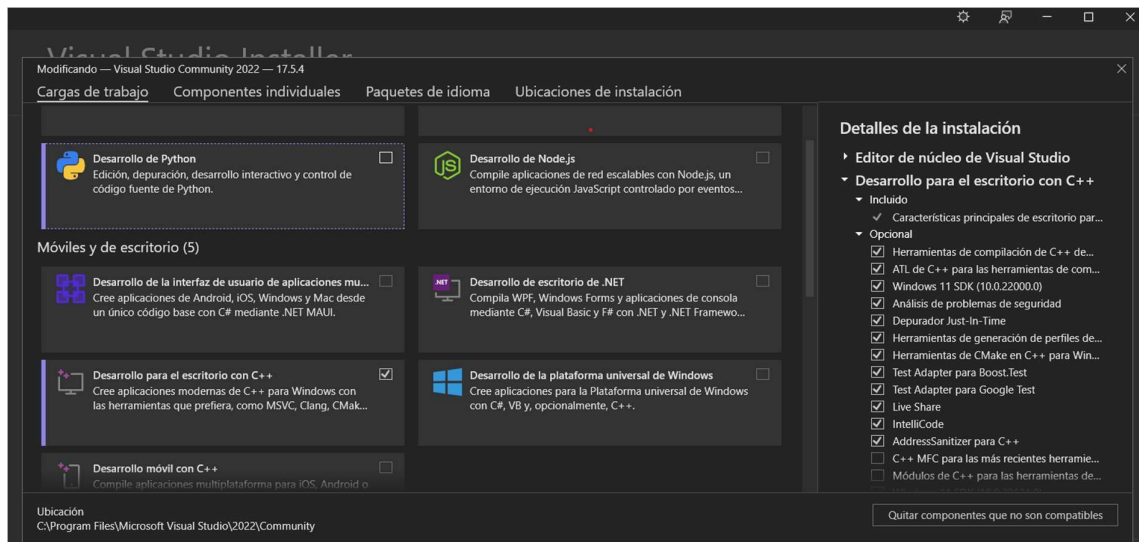
Starting with MySQL 5.6 the MySQL Installer package replaces the standalone MSI packages.

Windows (x86, 32 & 64-bit), MySQL Installer MSI [Go to Download Page >](#)

Other Downloads:

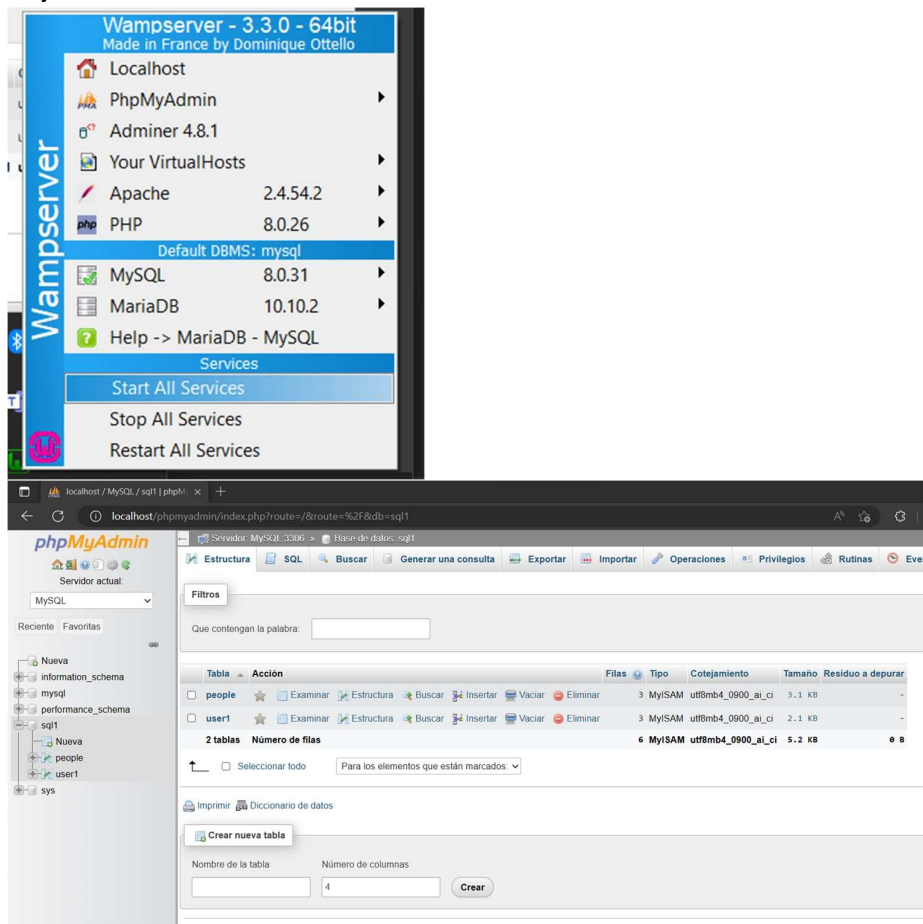
Download	Version	Size	Action
Windows (x86, 64-bit), ZIP Archive <small>(mysql-8.0.33-winx64.zip)</small>	8.0.33	230.1M	Download
<small>MD5: 582a256f77da9846d88ef9fae86c8790 Signature</small>			
Windows (x86, 64-bit), ZIP Archive Debug Binaries & Test Suite <small>(mysql-8.0.33-winx64-debug-test.zip)</small>	8.0.33	663.7M	Download
<small>MD5: fddff737ba0a00a32ae730573f24a9702 Signature</small>			

- **Visual studio comunity:** este será el IDE donde realizaremos la programación requerida y las configuraciones de conexión

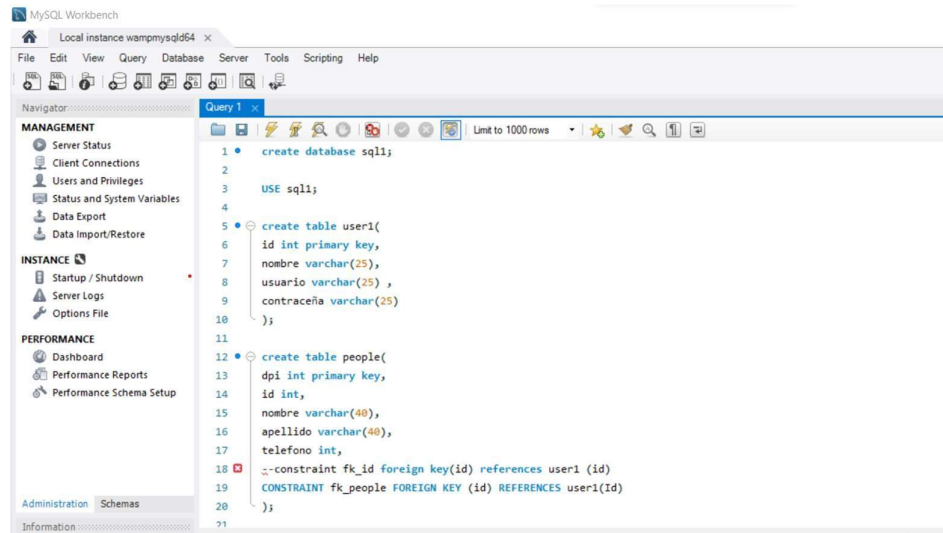


Con estas herramientas instalados en nuestra computadora podemos comenzar con la realización de la conexión de MYSQL con c++.

1. Lo primero sería encender el wampserver para poder tener acceso al servidor donde estará alojado la base de datos



2. Ya con esta parte activada podemos elegir entre usar workbench o el mismo servidor de MYSQL para crear la base de datos



3. Dentro de workbench o el administrado ejecutamos el siguiente código para crear la base de datos:

create database Asignacion2; -- con esto creamos la base de datos

use Asignacion2; -- aqui enpesamos a usar la base de datos para crear eliminar o editar tables

```
create table Profesor(
idP int primary key not null auto_increment,
nombreP varchar(50),
edadP int (10),
telP int(10),
correoP varchar(50),
titulo varchar(50)
);
```

```
create table Clase(
idC int primary key not null auto_increment,
idP int,
nombreC varchar(50),
horarion varchar(25),
salon varchar(25),
constraint fk_idP foreign key(idP) references Profesor (idP)
);
```

```
create table Estudiante(
idE int primary key not null auto_increment,
nombreE varchar(50),
edadE int (10),
telE int(10),
correoE varchar(50)
);
```

```
create table Asignacion(
idA int primary key not null auto_increment,
idC int,
```

```
idE int,  
constraint fk_idC foreign key(idC) references Clase (idC),  
constraint fk_idE foreign key(idE) references Estudiante (idE)  
);
```

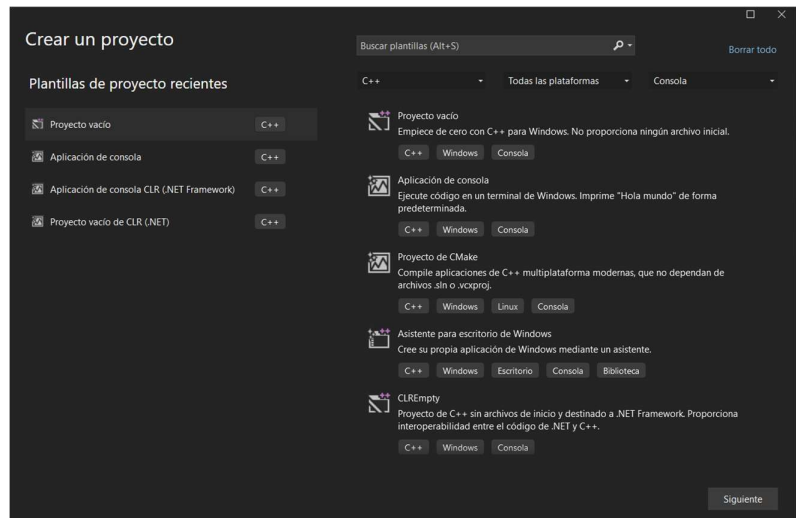
```
insert into Profesor(nombreP, edadP, telP, correoP, titulo) value('Kevin  
Mazariegos',27,42478258,'ElbuenTioben@gmail.com','Ingenieria en sistemas');  
insert into Profesor(nombreP, edadP, telP, correoP, titulo) value('Edgar  
Paz',25,56982341,'YosoyTupadre@gmail.com','Ingenieria en sistemas');  
insert into Profesor(nombreP, edadP, telP, correoP, titulo) value('Mynor  
Montejo',30,23289632,'TengoAnciedad@gmail.com','Ingenieria en Mecatronica');  
insert into Profesor(nombreP, edadP, telP, correoP, titulo) value('Wilson  
Romero',29,6660666,'vivapeten@gmail.com','Ingenieria en sistemas');  
insert into Profesor(nombreP, edadP, telP, correoP, titulo) value('Mauro  
Can',40,82584742,'nosemeOcurenada@gmail.com','Ingenieria en Mecanica');
```

```
insert into Clase(idP, nombreC, horarion, salon) value(4,'programacion 8','8:00 - 10:00','salon  
201');  
insert into Clase(idP, nombreC, horarion, salon) value(1,'Calculo 3','8:00 - 10:00','salon 202');  
insert into Clase(idP, nombreC, horarion, salon) value(3,'fisica 10','8:00 - 10:00','salon 203');  
insert into Clase(idP, nombreC, horarion, salon) value(5,'base de datos','8:00 - 10:00','salon  
205');  
insert into Clase(idP, nombreC, horarion, salon) value(2,'Calculo 1','8:00 - 10:00','salon 200');
```

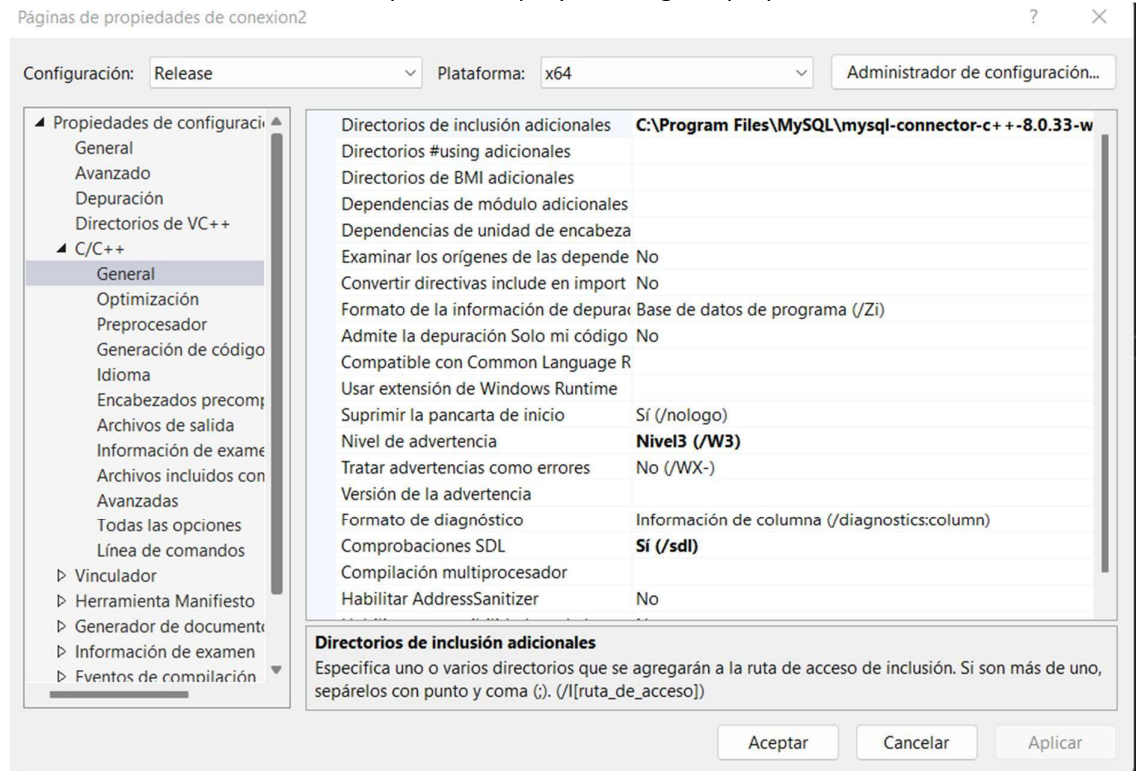
```
insert into Estudiante(nombreE, edadE, telE, correoE) value('Mayerli  
Pineda',20,42478258,'MegustaBadBuny@gmail.com');  
insert into Estudiante(nombreE, edadE, telE, correoE) value('Brayan  
Garcia',26,48572878,'DineroMony@gmail.com');  
insert into Estudiante(nombreE, edadE, telE, correoE) value('Guiyermo  
Mendes',24,42478258,'BadDestiny@gmail.com');  
insert into Estudiante(nombreE, edadE, telE, correoE) value('Jimmy  
Medina',28,12345678,'Drugs@gmail.com');
```

```
insert into Asignacion(idC,idE) value(1,1);  
insert into Asignacion(idC,idE) value(1,2);  
insert into Asignacion(idC,idE) value(2,1);  
insert into Asignacion(idC,idE) value(3,1);con este código ya tenemos una base de datos con  
datos dentro para poder testear otrabajar
```

4. Ahora continuamos con el IDE para trabajar en c++, lo primero será elegir un proyecto en blanco en el cual colocar el código y realizar ciertas modificaciones para que la conexión sea posible



5. Para las modificantes abrimos el apartado de proyecto luego de propiedades



6. Dentro del apartado c/c++->general-> directorios de inclusión adicionales agregamos la librería que se encuentran en la imagen

```
C:\Program Files\MySQL\Connector C++ 8.0\include  
C:\Program Files\MySQL\MySQL Server 8.0\include  
C:\Program Files\MySQL\Connector C++ 8.0\lib64  
C:\Program Files\MySQL\MySQL Server 8.0\lib
```

7. Luego agregamos dentro del apartado de vinculo->entradas ingresamos las dependencias que aparecen en la imagen
vincular

```
C:\Program Files\MySQL\Connector C++ 8.0\lib64\vs14  
mysqlcppconn.lib  
C:\Program Files\MySQL\MySQL Server 8.0\lib  
libmysql.lib
```

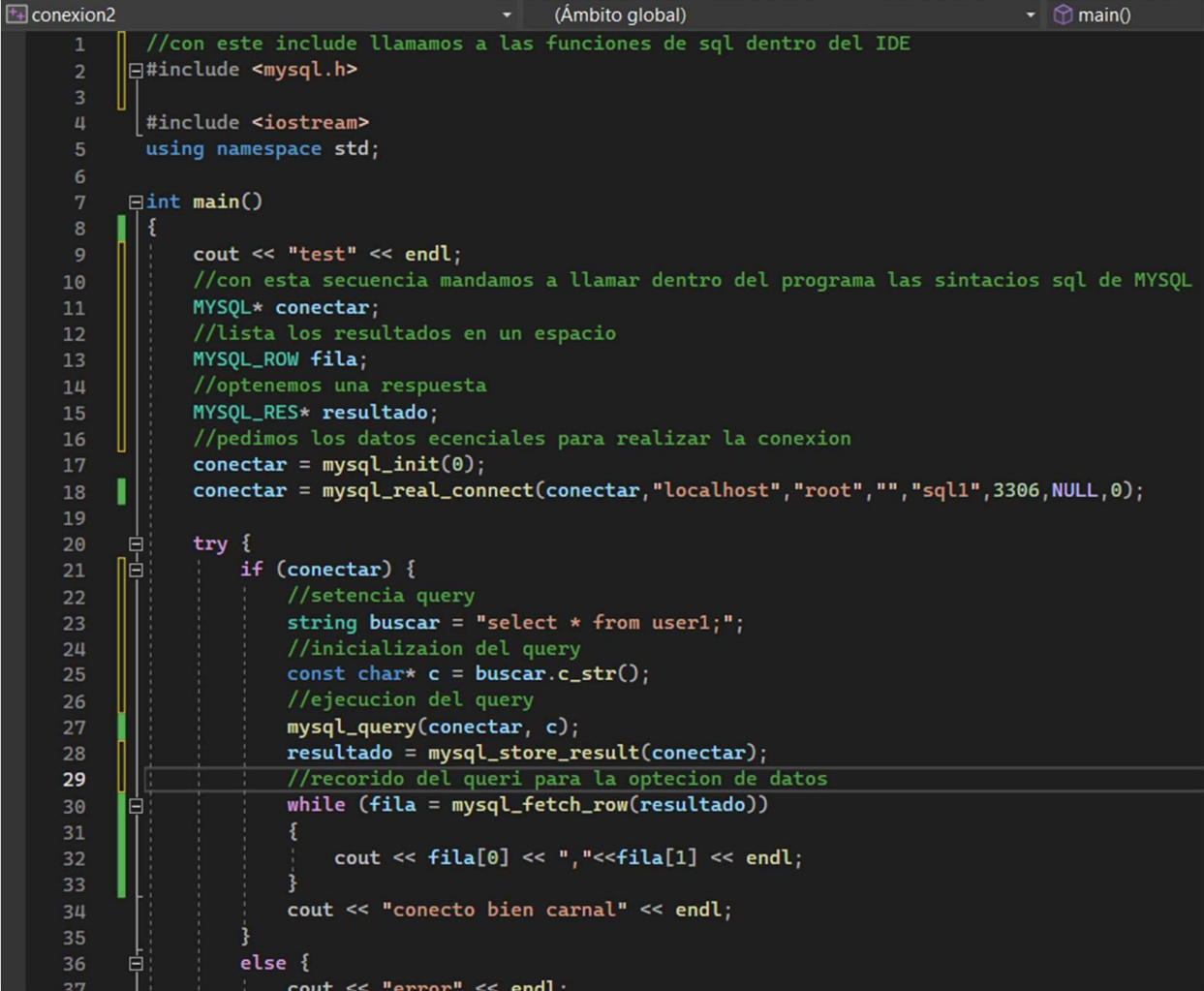
8. Ahora copiamos los siguientes archivos dentro de la carpeta del programa esto para que la ejecución de la conexión no presente errores

```
libmysql.dll  
libmysql.lib
```

9. Si esto nos llegara a dar un error del tipo dll tendremos que copiar los siguientes archivos en la carpeta del programa

```
libcrypto-1_1-x64.dll  
libssl-1_1-x64.dll
```


Con estos pasos tendremos una conexión estable entre el servidor de MYSQL y c++ ahora lo que



```
conexion2 (Ámbito global) main()
1 //con este include llamamos a las funciones de sql dentro del IDE
2 #include <mysql.h>
3
4 #include <iostream>
5 using namespace std;
6
7 int main()
8 {
9     cout << "test" << endl;
10    //con esta secuencia mandamos a llamar dentro del programa las sintaxis sql de MYSQL
11    MYSQL* conectar;
12    //lista los resultados en un espacio
13    MYSQL_ROW fila;
14    //obtenemos una respuesta
15    MYSQL_RES* resultado;
16    //pedimos los datos esenciales para realizar la conexión
17    conectar = mysql_init(0);
18    conectar = mysql_real_connect(conectar, "localhost", "root", "", "sql1", 3306, NULL, 0);
19
20    try {
21        if (conectar) {
22            //sentencia query
23            string buscar = "select * from user1;";
24            //inicialización del query
25            const char* c = buscar.c_str();
26            //ejecución del query
27            mysql_query(conectar, c);
28            resultado = mysql_store_result(conectar);
29            //recorrido del query para la obtención de datos
30            while (fila = mysql_fetch_row(resultado))
31            {
32                cout << fila[0] << ", " << fila[1] << endl;
33            }
34            cout << "conecto bien carnal" << endl;
35        }
36        else {
37            cout << "error" << endl;
```

necesitamos en la inserción de código en el IDE para realizar la correcta conexión y prueba para comprobar que el código se encuentra en buenas condiciones

Con esto tenemos una conexión y parte de query de una base de datos conectado a c++

Crud:

1. Creamos menus y submenús para la ejecución de el crud

```
try {
    do
    {
        cout << "asignacion de cursos  " << endl;
        cout << "ver datos .....1  " << endl;
        cout << "ingresar .....2  " << endl;
        cout << "modificar .....3  " << endl;
        cout << "eliminar .....4" << endl;
        cout << "finalizar .....5" << endl;

        cin >> i;
        switch (i)
        {
            case 1:
                do { ... } while (a != 5);
                break;
            default:
                break;
            case 2:
                do { ... } while (a != 3);
                break;
            case 3:
                do { ... } while (a != 3);
                break;
            case 4:
                string opcion;
                do { ... } while (a != 3);

                break;
        }

    } while (i != 5);
}
```

2. Mostrar: con el siguiente código mostramos los datos existentes

```
case 1:
do
{
    cout << "profesor .....1 " << endl;
    cout << "estudiante .....2 " << endl;
    cout << "clase .....3 " << endl;
    cout << "asignacion .....4" << endl;
    cout << "cancel .....5" << endl;
    cin >> a;
    switch (a)
    {
        case 1:
            if (conectar) {
                //setencia query
                string buscar = "select * from Profesor;";
                //inicializaion del query
                const char* c = buscar.c_str();
                //ejecucion del query
                mysql_query(conectar, c);
                resultado = mysql_store_result(conectar);
                //recorido del queri para la optecion de datos
                while (fila = mysql_fetch_row(resultado))
                {
                    cout << fila[0] << "," << fila[1] << "," << fila[2] << "," << fila[3] << "," << fila[4] << endl;
                }
                cout << "conexion correctoa" << endl;
            }
            else {
                cout << "error" << endl;
            }
            break;
    }
}
```

3. Insertar: con esta parte del código insertamos los datos que necesitamos en sus respectivas tablas

```
case 2:
do
{
    cout << "estudiante .....1 " << endl;
    cout << "asignacion .....2" << endl;
    cout << "cancel .....3" << endl;
    cin >> a;
    switch (a)
    {
        case 1:
            if (conectar) {
                cout << "Ingrese el nombre del estudiante: ";
                string nombre;
                getline(cin.ignore(), nombre);
                es.setNombreE(nombre);

                cout << "Ingrese la edad del estudiante: ";
                int edad;
                cin >> edad;
                cin.ignore();
                es.setEdadE(edad);

                cout << "Ingrese el teléfono del estudiante: ";
                int tel;
                cin >> tel;
                cin.ignore();
                es.setTelE(tel);

                cout << "Ingrese el correo del estudiante: ";
                string correo;
                getline(cin.ignore(), correo);
                es.setCorreoE(correo);

                //setencia query
                string insert = "insert into Estudiante(nombreE, edadE, telE, correoE) value('" + es.getNombreE() + "', "
                    + to_string(es.getEdadE()) + ", " + to_string(es.getTelE()) + ", '" + es.getCorreoE() + "')";
                //inicializaion del query
                const char* c = insert.c_str();
                //ejecucion del query
                q_estado = mysql_query(conectar, c);
                if (!q_estado)
                {
                    cout << "ingresado con exito" << endl;
                }
                else {
                    cout << "no ingresado" << endl;
                }
            }
            else {
                cout << "error" << endl;
            }
            break;
    }
}
```

4. Modificar: con esta siguiente parte podemos modificar los datos que busquemos

```
switch (a)
{
case 1:
    if (conectar) {
        // Obtener el ID del estudiante que deseas modificar
        int id;
        cout << "Ingrese el ID del estudiante a modificar: ";
        cin >> id;
        cin.ignore();

        string buscarM = "SELECT * FROM Estudiante WHERE idE = " + to_string(id) + ";";

        // Ejecutar la consulta
        const char* c = buscarM.c_str();
        q_estado = mysql_query(conectar, c);
        if (!q_estado)
        {
            resultado = mysql_store_result(conectar);
            if (fila = mysql_fetch_row(resultado))
            {
                // El estudiante fue encontrado, puedes mostrar los datos existentes antes de la modificación
                cout << "Datos del estudiante a modificar:" << endl;
                cout << "ID: " << fila[0] << endl;
                cout << "Nombre: " << fila[1] << endl;
                cout << "Edad: " << fila[2] << endl;
                cout << "Teléfono: " << fila[3] << endl;
                cout << "Correo: " << fila[4] << endl;

                // Solicitar los nuevos valores al usuario
                cout << "Ingrese el nuevo nombre del estudiante: ";
                string nombre;
                getline(cin, nombre);
                es.setNombreE(nombre);

                cout << "Ingrese la nueva edad del estudiante: ";
                int edad;
                cin >> edad;
                cin.ignore();
                es.setEdadE(edad);

                cout << "Ingrese el nuevo teléfono del estudiante: ";
                int tel;
                cin >> tel;
                cin.ignore();
                es.setTelE(tel);

                cout << "Ingrese el nuevo correo del estudiante: ";
                string correo;
                getline(cin, correo);
                es.setCorreoE(correo);

                // Construir la sentencia de actualización
                string update = "UPDATE Estudiante SET nombreE = '" + es.getNombreE() + "', edadE = "
                    + to_string(es.getEdadE()) + ", telE = " + to_string(es.getTelE()) + ", correoE = '"
                    + es.getCorreoE() + "' WHERE idE = " + to_string(id) + ";";

                // Ejecutar la sentencia de actualización
                const char* c = update.c_str();
                q_estado = mysql_query(conectar, c);
                if (!q_estado)
                {
                    cout << "Dato modificado con éxito" << endl;
                }
                else {
                    cout << "Error al modificar el dato" << endl;
                }
            }
        }
    }
}
```

5. Eliminar: con este ultimo paso podemos eliminar

```
case 1:
if (conectar) {
    // Obtener el ID del estudiante que deseas modificar
    int id;
    cout << "Ingrese el ID del estudiante a eliminar: ";
    cin >> id;
    cin.ignore();

    string buscarM = "SELECT * FROM Estudiante WHERE idE = " + to_string(id) + ";";

    // Ejecutar la consulta
    const char* c = buscarM.c_str();
    q_estado = mysql_query(conectar, c);
    if (!q_estado)
    {
        resultado = mysql_store_result(conectar);
        if (fila = mysql_fetch_row(resultado))
        {
            // El estudiante fue encontrado, puedes mostrar los datos existentes antes de la modificación
            cout << "Datos del estudiante a eliminar:" << endl;
            cout << "ID: " << fila[0] << endl;
            cout << "Nombre: " << fila[1] << endl;
            cout << "Edad: " << fila[2] << endl;
            cout << "Teléfono: " << fila[3] << endl;
            cout << "Correo: " << fila[4] << endl;

            // Mostrar confirmación antes de eliminar
            char opcion;
            cout << "¿Está seguro de eliminar el estudiante? (y/n): ";
            cin >> opcion;

            if (opcion == 'y' || opcion == 'Y') {
                // Construir la sentencia de eliminación
                string eliminar = "DELETE FROM Estudiante WHERE idE = " + to_string(id) + ";";

                // Ejecutar la sentencia de eliminación
                const char* c = eliminar.c_str();
                q_estado = mysql_query(conectar, c);
                if (!q_estado)
                {
                    cout << "Dato eliminado con éxito" << endl;
                }
                else {
                    cout << "Error al eliminar el dato" << endl;
                }
            }
            else {
                cout << "No se eliminó el estudiante" << endl;
            }
        }
    }
    else {
        // El estudiante no fue encontrado
        cout << "No se encontró un estudiante con el ID especificado" << endl;
    }
}
```