Quản lý chương trình đào tạo

# 1. Cơ sở dữ liệu

## 1.1. Tạo CSDL và bảng bảng

* CSDL: **CREATE DATABASE EduManager;**
* Bảng

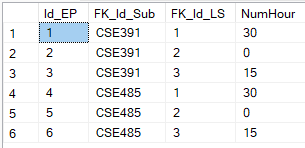
| **-- Bảng Subjects**  CREATE TABLE Subjects (  Id\_Sub VARCHAR(10) PRIMARY KEY,  Name\_Sub NVARCHAR(100)  );  **-- Bảng LearningStyle**  CREATE TABLE LearningStyle (  Id\_LS INT PRIMARY KEY,  Name\_LS NVARCHAR(50)  );  **-- Bảng EduProgram**  CREATE TABLE EduProgram (  Id\_EP INT PRIMARY KEY IDENTITY(1,1),  FK\_Id\_Sub VARCHAR(10),  FK\_Id\_LS INT,  NumHour INT,  FOREIGN KEY (FK\_Id\_Sub) REFERENCES Subjects(Id\_Sub),  FOREIGN KEY (FK\_Id\_LS) REFERENCES LearningStyle(Id\_LS)  ); |
| --- |

## 1.2. Chèn dữ liệu

| **-- Thêm dữ liệu vào bảng LearningStyle**  INSERT INTO LearningStyle (Id\_LS, Name\_LS) VALUES  (1, N'Lý thuyết / lý luận'),  (2, N'Bài tập / Thảo luận'),  (3, N'Thực hành'); |
| --- |

## 1.3. Tạo view

| CREATE VIEW EduProgramView AS  WITH SourceData AS (  SELECT  FK\_Id\_Sub,  Name\_Sub,  Name\_LS,  NumHour  FROM  EduProgram  JOIN  Subjects ON EduProgram.FK\_Id\_Sub = Subjects.Id\_Sub  JOIN  LearningStyle ON EduProgram.FK\_Id\_LS = LearningStyle.Id\_LS  )  SELECT  FK\_Id\_Sub AS "Mã môn học",  Name\_Sub AS "Tên môn học",  [Lý thuyết / lý luận] AS "Lý thuyết",  [Bài tập / Thảo luận] AS "Bài tập",  [Thực hành] AS "Thực hành"  FROM  SourceData  PIVOT (  MAX(NumHour)  FOR Name\_LS IN ([Lý thuyết / lý luận], [Bài tập / Thảo luận], [Thực hành])  ) AS PivotTable; |
| --- |



*Hình trên biểu thị dữ liệu ban đầu của bảng EduProgram khi chưa PIVOT*

Đoạn mã lệnh SQL trên tạo một khung nhìn (VIEW) trong cơ sở dữ liệu với tên là EduProgramView. Nó sử dụng cú pháp WITH...AS... để định nghĩa dữ liệu nguồn và sau đó áp dụng **PIVOT** để xoay dữ liệu từ định dạng hàng thành định dạng cột.



Cụ thể hơn:

* **Khung nhìn (VIEW)**: Tạo một dạng lưu trữ ảo của dữ liệu, cho phép truy xuất dễ dàng dưới dạng bảng mà không cần truy vấn lại cơ sở dữ liệu mỗi lần.
* **WITH SourceData AS ...**: Đây là một câu lệnh con (subquery) được gọi là SourceData. Nó thu thập dữ liệu từ bảng EduProgram, sau đó nối (JOIN) với bảng Subjects để lấy tên môn học (Name\_Sub) và với bảng LearningStyle để lấy kiểu học (Name\_LS). Nó cũng lấy thông tin số giờ học (NumHour).
* **SELECT ... FROM SourceData PIVOT ...**: Đây là phần chính của truy vấn. Nó sử dụng PIVOT để xoay dữ liệu:
* Phần **SELECT** xác định các cột sẽ xuất hiện trong kết quả. Nó cũng chuyển đổi tên cột thành tên thân thiện hơn với người dùng (ví dụ, "Mã môn học" thay vì FK\_Id\_Sub).
* Phần **PIVOT** định nghĩa cách xoay dữ liệu. Nó sử dụng hàm tổng hợp MAX để lấy số giờ lớn nhất cho từng môn học theo từng kiểu học. FOR Name\_LS IN ([Lý thuyết / lý luận], [Bài tập / Thảo luận], [Thực hành]) chỉ định rằng bạn muốn xoay dữ liệu trên cột Name\_LS và tạo các cột mới cho từng giá trị duy nhất (ở đây là "Lý thuyết / lý luận", "Bài tập / Thảo luận", "Thực hành").
* Tên "**PivotTable**" được đặt cho bảng kết quả sau khi áp dụng PIVOT.
* Kết quả cuối cùng là một bảng với các cột: "Mã môn học", "Tên môn học", "Lý thuyết", "Bài tập", "Thực hành", hiển thị số giờ cho từng kiểu học theo mỗi môn học.

Truy vấn này giúp bạn thấy rõ ràng số giờ học theo từng loại hoạt động (lý thuyết, bài tập, thực hành) cho mỗi môn học trong chương trình giáo dục.

# 2. Mã nguồn

## 2.1. Services

**ConnectDatabase.cs**

| using System;  using System.Collections.Generic;  using System.Data;  using System.Data.SqlClient;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace EduManager.Services  {  internal class ConnectDatabase  {  private static ConnectDatabase instance;  private string \_connectionString = @"Data Source=DUNGHAQUANG;Initial Catalog=EduManager;Integrated Security=True";  private ConnectDatabase() { }  **// Tạo kết nối**  public static ConnectDatabase getInstance()  {  if (instance == null)  {  instance = new ConnectDatabase();  }  return instance;  }  **// Sử dụng để truy vấn câu lệnh SELECT**  public DataTable ExecuteQuery(string query, SqlParameter[] parameter = null)  {  DataTable data = new DataTable();  using (SqlConnection sqlConnection = new SqlConnection(\_connectionString))  {  sqlConnection.Open();  SqlCommand cmd = new SqlCommand(query, sqlConnection);  if (parameter != null)  cmd.Parameters.AddRange(parameter);  SqlDataAdapter adapter = new SqlDataAdapter(cmd);  data.Clear();  adapter.Fill(data);  sqlConnection.Close();  }  return data;  }  **// Sử dụng để thêm sửa xóa**  public int ExecuteNonQuery(string query, SqlParameter[] parameter = null)  {  int rowsAffected = 0;  using (SqlConnection sqlConnection = new SqlConnection(\_connectionString))  {  sqlConnection.Open();  SqlCommand cmd = new SqlCommand(query, sqlConnection);  if (parameter != null)  cmd.Parameters.AddRange(parameter);  rowsAffected = cmd.ExecuteNonQuery();  sqlConnection.Close();  }  return rowsAffected;  }  public string GetValue(string query)  {  string value = null;  using (SqlConnection sqlConnection = new SqlConnection(\_connectionString))  {  sqlConnection.Open();  SqlCommand cmd = new SqlCommand(query, sqlConnection);  SqlDataReader dataReader = cmd.ExecuteReader();  if (dataReader.Read())  {  value = dataReader.GetValue(0).ToString();  }  sqlConnection.Close();  }  return value;  }  public SqlDataReader ExecuteReader(string query)  {  SqlConnection sqlConnection = new SqlConnection(\_connectionString);  sqlConnection.Open();  SqlCommand cmd = new SqlCommand(query, sqlConnection);  SqlDataReader reader = cmd.ExecuteReader();  return reader;  }  }  } |
| --- |

## 2.2. Models

### 2.2.1. Subject

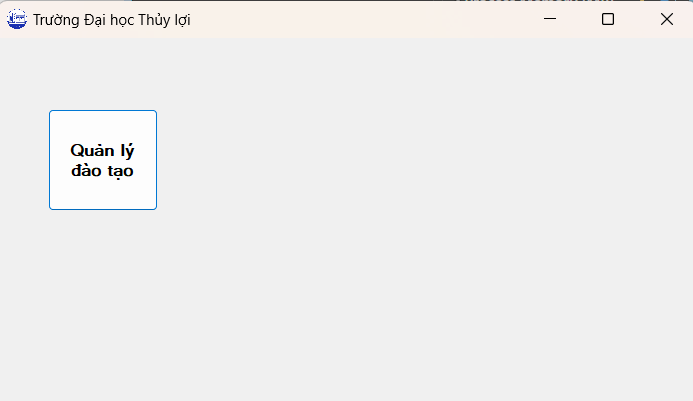
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace EduManager.Models  {  internal class Subject  {  public string Id\_Sub { get; set; }  public string Name\_Sub { get; set; }  public Subject() { }  public Subject(string id\_Sub)  {  Id\_Sub = id\_Sub;  }  public Subject(string id\_Sub, string name\_Sub)  {  Id\_Sub = id\_Sub;  Name\_Sub = name\_Sub;  }  }  } |
| --- |

### 2.2.2. EduProgram

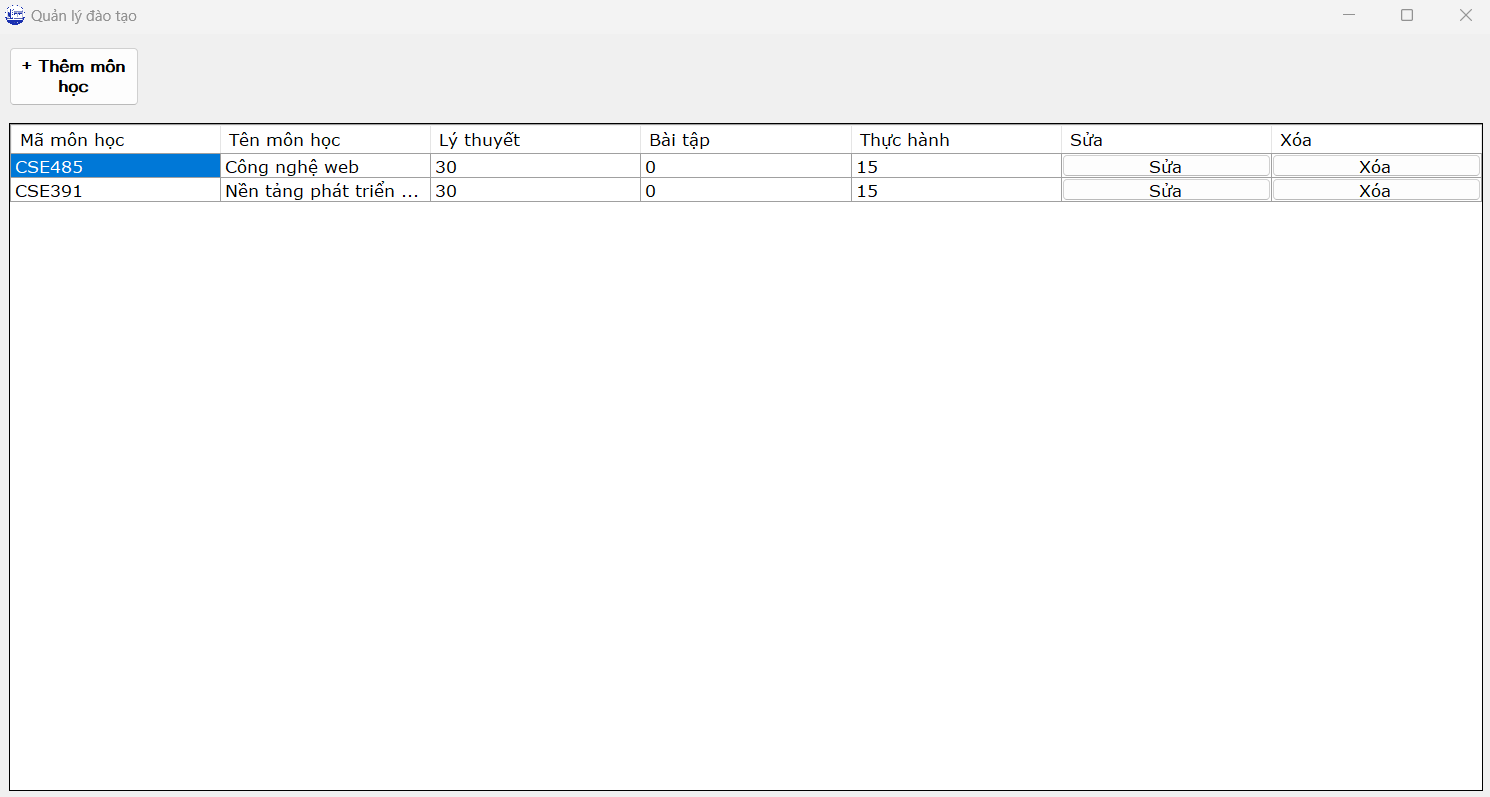
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace EduManager.Models  {  internal class EduProgram  {  public int Id\_EP { get; set; }  public string FK\_Id\_Sub { get; set; }  public int FK\_Id\_LS { get; set; }  public int NumHour { get; set; }  public EduProgram() { }  public EduProgram(string fK\_Id\_Sub)  {  FK\_Id\_Sub = fK\_Id\_Sub;  }  public EduProgram(string fK\_Id\_Sub, int fK\_Id\_LS, int numHour)  {  FK\_Id\_Sub = fK\_Id\_Sub;  FK\_Id\_LS = fK\_Id\_LS;  NumHour = numHour;  }  public EduProgram(int id\_EP, string fK\_Id\_Sub, int fK\_Id\_LS, int numHour)  {  Id\_EP = id\_EP;  FK\_Id\_Sub = fK\_Id\_Sub;  FK\_Id\_LS = fK\_Id\_LS;  NumHour = numHour;  }  }  } |
| --- |

## 2.3. Views

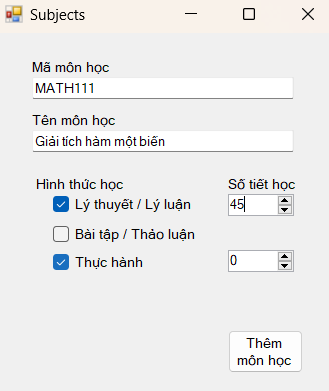
### 2.3.1. Form Main



### 2.3.2. Form EduProgram



### 2.3.3. Form Subjects



## 2.4. Controllers

### 2.4.1. EduProgramController

| using System;  using System.Collections.Generic;  using System.Data;  using System.Data.SqlClient;  using System.Linq;  using System.Runtime.InteropServices;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  using EduManager.Models;  using EduManager.Services;  namespace EduManager.Controllers  {  internal class EduProgramController  {  private static EduProgramController instance;  private ConnectDatabase connectDatabase = ConnectDatabase.getInstance();  private EduProgramController() { }  **// Khởi tạo**  public static EduProgramController Instance()  {  if (instance == null)  {  instance = new EduProgramController();  }  return instance;  }  **// Thiết lập việc cho phép chỉnh sửa cột**  public void ConfigureDataGridView(DataGridView dgv, string columnName)  {  DataGridViewColumn column = dgv.Columns[columnName]; // Lấy cột  if (column != null)  {  column.ReadOnly = true; // Đặt cột chỉ đọc  }  }  **// Hiển thị dữ liệu lên Datagridview**  public void showData(DataGridView dgv)  {  // Truy vấn dữ liệu  string query = "SELECT \* FROM EduProgramView";  dgv.Columns.Clear();  dgv.DataSource = ConnectDatabase.getInstance().ExecuteQuery(query);  // Thêm cột "Sửa"  DataGridViewButtonColumn editColumn = new DataGridViewButtonColumn();  editColumn.HeaderText = "Sửa";  editColumn.Text = "Sửa";  editColumn.UseColumnTextForButtonValue = true;  dgv.Columns.Add(editColumn);  // Thêm cột "Xóa"  DataGridViewButtonColumn deleteColumn = new DataGridViewButtonColumn();  deleteColumn.HeaderText = "Xóa";  deleteColumn.Text = "Xóa";  deleteColumn.UseColumnTextForButtonValue = true;  dgv.Columns.Add(deleteColumn);  ConfigureDataGridView(dgv, "Mã môn học");  }  **// Thêm mới dữ liệu**  public bool addData(EduProgram ep)  {  string query = "INSERT INTO EduProgram VALUES (@FK\_Id\_Sub, @FK\_Id\_LS, @NumHour)";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("FK\_Id\_Sub", ep.FK\_Id\_Sub),  new SqlParameter("FK\_Id\_LS", ep.FK\_Id\_LS),  new SqlParameter("NumHour", ep.NumHour)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);  return rowAffect > 0 ? true : false;  }  **// Sửa dữ liệu**  public bool editData(EduProgram ep)  {  string query = "UPDATE EduProgram " +  "SET NumHour = @NumHour " +  "WHERE FK\_Id\_Sub = @FK\_Id\_Sub AND FK\_Id\_LS = @FK\_Id\_LS;";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("FK\_Id\_Sub", ep.FK\_Id\_Sub),  new SqlParameter("FK\_Id\_LS", ep.FK\_Id\_LS),  new SqlParameter("NumHour", ep.NumHour)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);  return rowAffect > 0 ? true : false;  }  **// Xóa dữ liệu**  public bool removeData(EduProgram ed)  {  string query = "DELETE FROM EduProgram WHERE Id\_EP = @Id\_EP";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("Id\_EP", ed.Id\_EP)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);    return rowAffect > 0 ? true : false;  }  **// Xóa toàn bộ dữ liệu**  public bool removeAllData(EduProgram ed)  {  string query = "DELETE FROM EduProgram WHERE FK\_Id\_Sub = @FK\_Id\_Sub";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("FK\_Id\_Sub", ed.FK\_Id\_Sub)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);  return rowAffect > 0 ? true : false;  }  }  } |
| --- |

### 2.4.2. SubjectController

| using System;  using System.Collections.Generic;  using System.Data.SqlClient;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  using EduManager.Models;  using EduManager.Services;  namespace EduManager.Controllers  {  internal class SubjectController  {  private static SubjectController instance;  private ConnectDatabase connectDatabase = ConnectDatabase.getInstance();  private SubjectController() { }  **// Khởi tạo**  public static SubjectController Instance()  {  if (instance == null)  {  instance = new SubjectController();  }  return instance;  }  **// Thêm mới môn học**  public bool addSubject(Subject subject)  {  string query = "INSERT INTO Subjects VALUES (@Id\_Sub , @name\_Sub)";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("Id\_Sub", subject.Id\_Sub),  new SqlParameter("name\_Sub", subject.Name\_Sub)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);  // ExecuteNonQuery sẽ trả về số dòng được thay đổi tại Database. Ví dụ không có dữ liệu nào được thay đổi hay là thay đổi lỗi thì số dòng thay đổi sẽ = 0 => Lỗi  return rowAffect > 0 ? true : false;  }  **// Hiển thị môn học lên DataGridView**  public void showData(DataGridView dgv)  {  string query = "Select \r\n\tId\_Sub AS N'Mã môn học', \r\n\tName\_Sub AS N'Tên môn học'\r\nFROM Subjects";  dgv.DataSource = connectDatabase.ExecuteQuery(query);  }  **// Sửa môn học**  public bool editData(Subject s)  {  string query = "UPDATE Subjects SET Name\_Sub = @name\_Sub WHERE Id\_Sub = @Id\_Sub";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("Id\_Sub", s.Id\_Sub),  new SqlParameter("name\_Sub", s.Name\_Sub)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);  return (rowAffect > 0 ? true : false);  }  **// Xóa môn học**  public bool removeData(Subject s)  {  string query = "DELETE FROM Subjects WHERE Id\_Sub = @Id\_Sub";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("Id\_Sub", s.Id\_Sub)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);  return rowAffect > 0 ? true : false;  }  }  } |
| --- |

### 