Lab 04:

LẬP TRÌNH WINDOWS FORM KẾT NỐI CSDL SQL SERVER

A. MỤC TIÊU:

- ✓ Hướng dẫn sinh viên làm quen với việc xây dựng ứng dụng Windows App có kết nối với CSDL SQL Server bằng Entity FrameWork của .NET.
- ✓ Sử dụng mô hình *Code First* trong *EntityFrameWork và* Sử dụng ADO.NET
- ✓ Thiết kế các Form nhập liệu cho các bảng trong cơ sở dữ liệu (hiện thị, thêm, xóa, sửa)
- ✓ Nâng cao: Tìm hiểu vấn đề transaction trong thao tác với cơ sở dữ liệu.

B. BÀI TẬP:

Bài Tập 1 Sử dụng EntityFrameWork với mô hình Code First

- ✓ Sử dụng SQL Server tạo cơ sở dữ liệu "QuanLySinhVien" đơn giản với 2 bảng

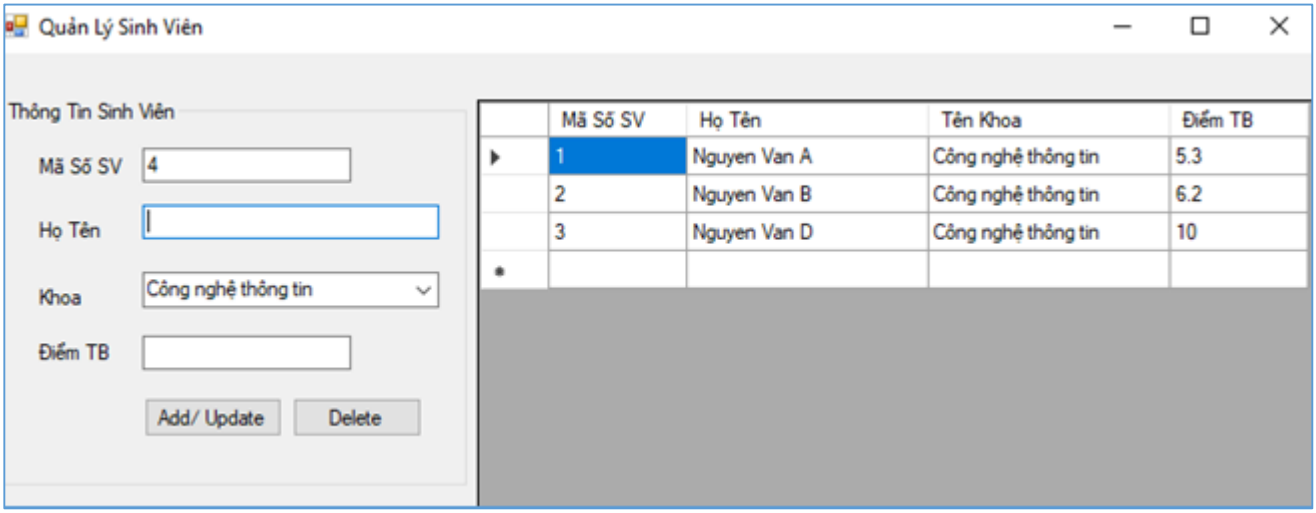
  Student (***StudentID***, *FullName*, *AverageScore*, *FalcultyID*)

  và **Falculty**(***FalcultyID***, *FacultyName*)

  để thể hiện thông tin Sinh viên và Khoa

```
USE [QuanLySinhVien]
GO
/****** Object:  Table [dbo].[Student]    Script Date: 08/13/2019 23:16:37
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Student](
	[StudentID] [nvarchar](20) NOT NULL,
	[FullName] [nvarchar](200) NULL,
	[AverageScore] [double] NULL,
	[FacultyID] [int] NULL,
 CONSTRAINT [PK_Student] PRIMARY KEY CLUSTERED
(
	[StudentID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Falculty]    Script Date: 08/13/2019
23:16:37 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Falculty](
	[FacultyID] [int] NOT NULL,
	[FacultyName] [nvarchar](200) NULL,
 CONSTRAINT [PK_Falculty] PRIMARY KEY CLUSTERED
(
	[FacultyID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

✓ Viết chương trình quản lý sinh viên có giao diện sau đây



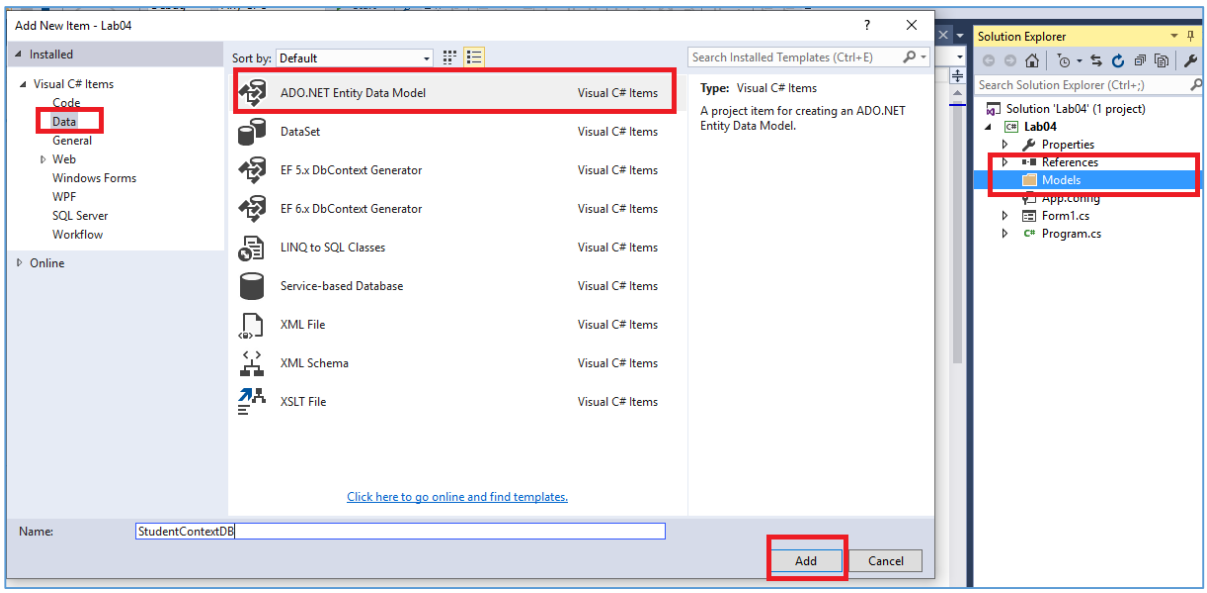
Yêu Cầu Xử Lý

- Sự kiện Form_load: Hiển thị danh sách các sinh viên đang có trong CSDL
  Combobox Khoa lấy từ bảng **Falculty** và hiện thị tên khoa
- Add/ Update: Nếu Sinh Viên chưa có mã số trong cơ sở dữ liệu thì Thêm mới
  Nếu đã tồn tại thì cập nhật
  Thông báo thành công khi insert/update hoặc khi gặp lỗi.
- Delete: Nếu mã Sinh Viên không tồn tại thông báo thể hiện thông báo
  Nếu mã Sinh viên tồn tại. Cảnh báo Yes/No trước khi thực hiện, xóa dòng dữ liệu có mã số StudentID trong cơ sở dữ liệu khi chọn Yes.

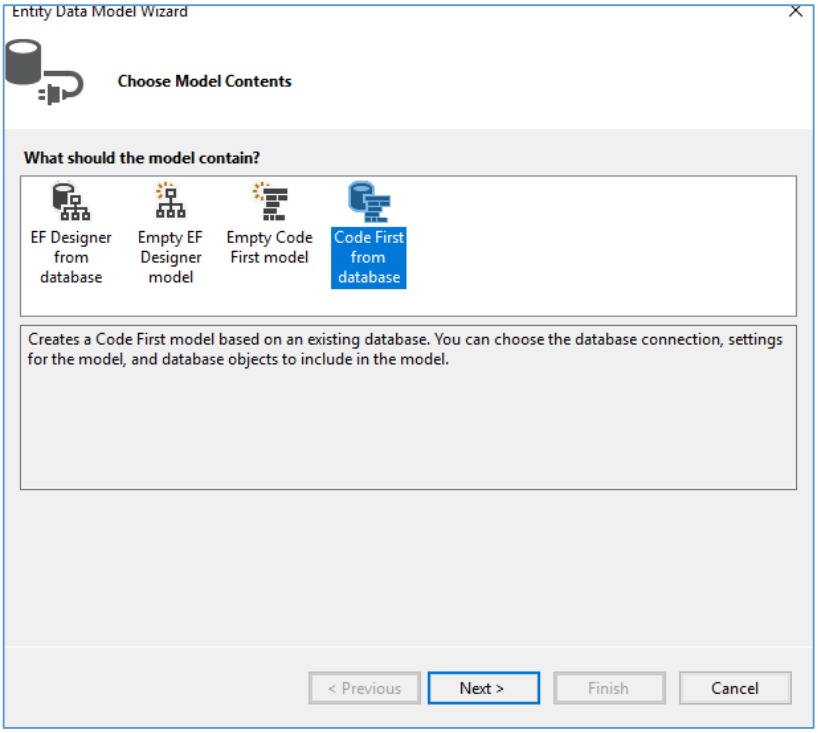
Hướng Dẫn

Bước 1: Entity FrameWork sinh ra các class chúng ta nên gom vào 1 thư mục (Models) để dễ dàng quản lý.

Click chuột phải vào Models chọn ***New Item***. Chọn Loại ***Data/ ADO.NET Entity Data Model***

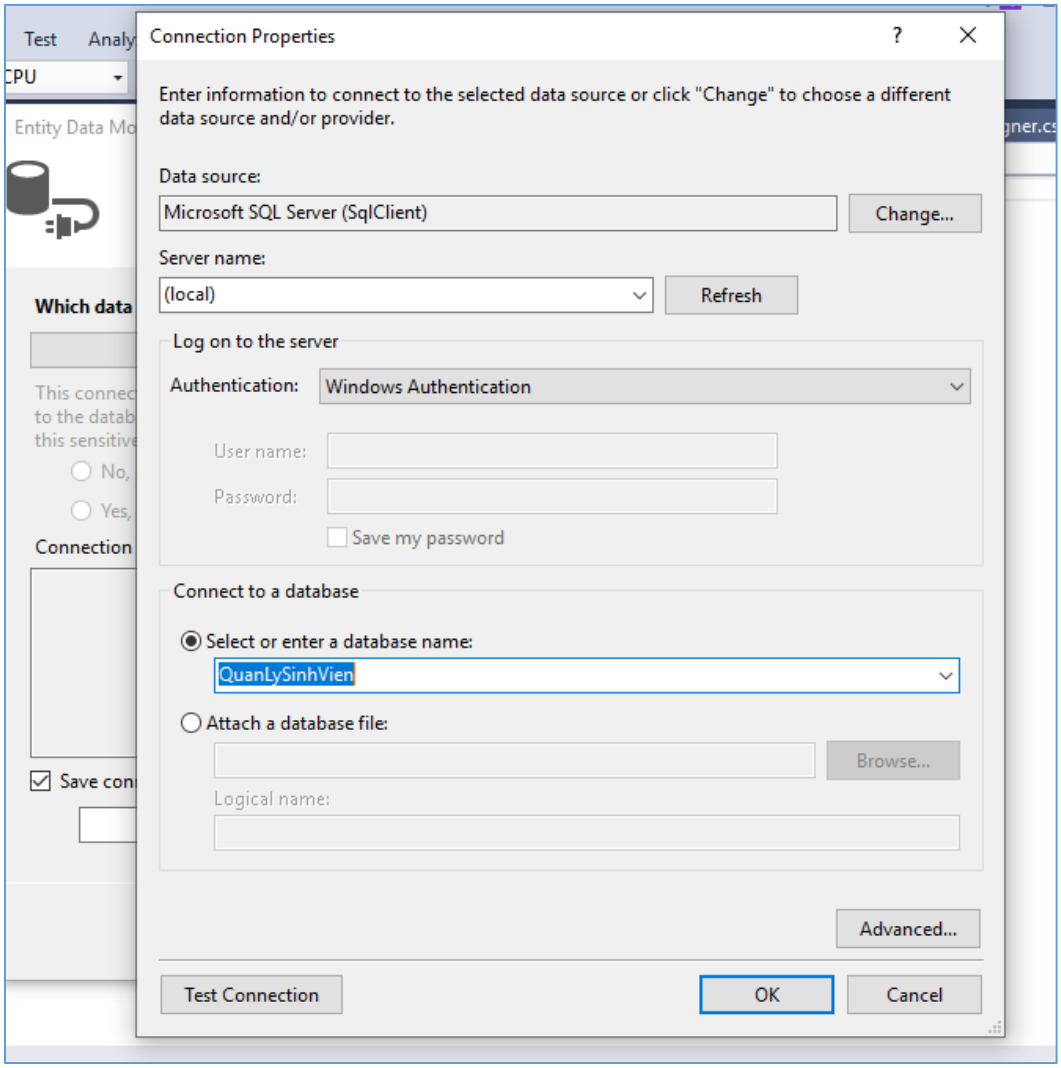


Đặt tên context là "***StudentContextDB***" (mặc định là Model1). Và chọn Add

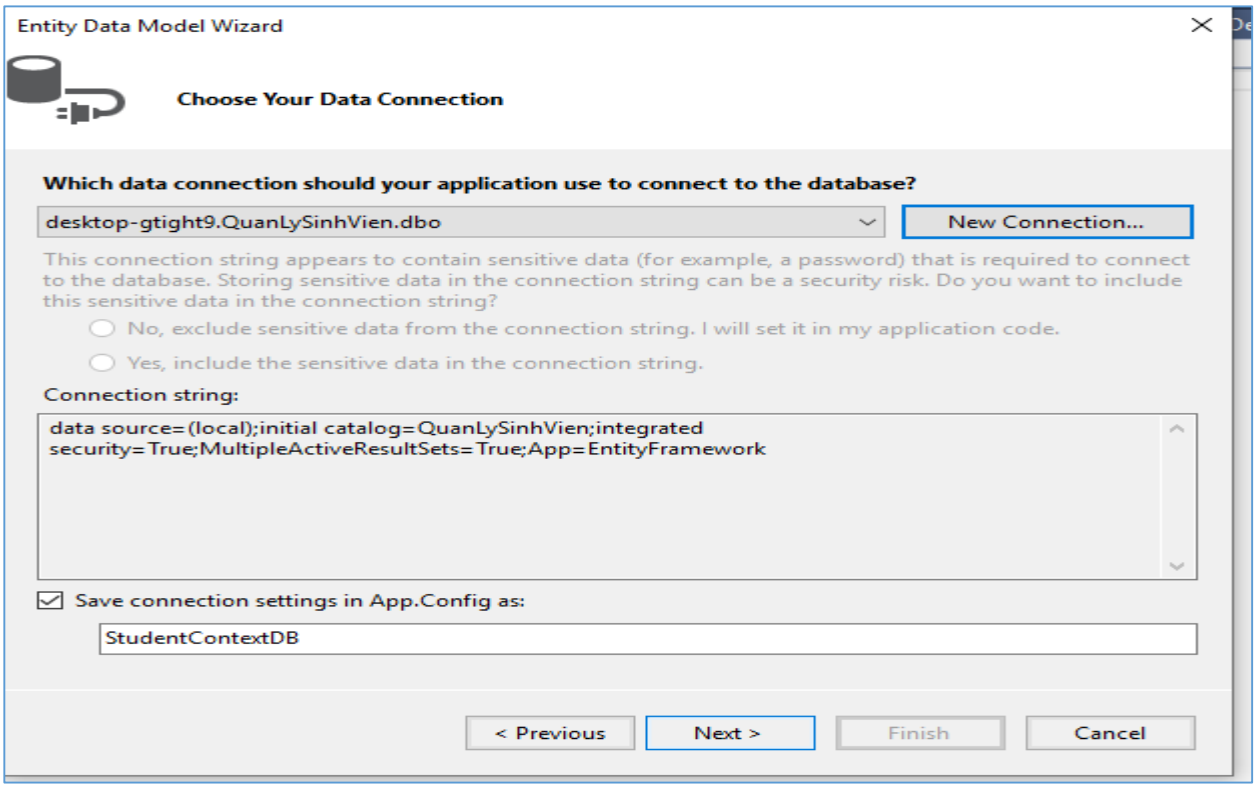


Ta chọn loại model là "***Code first from database***". Chọn Next

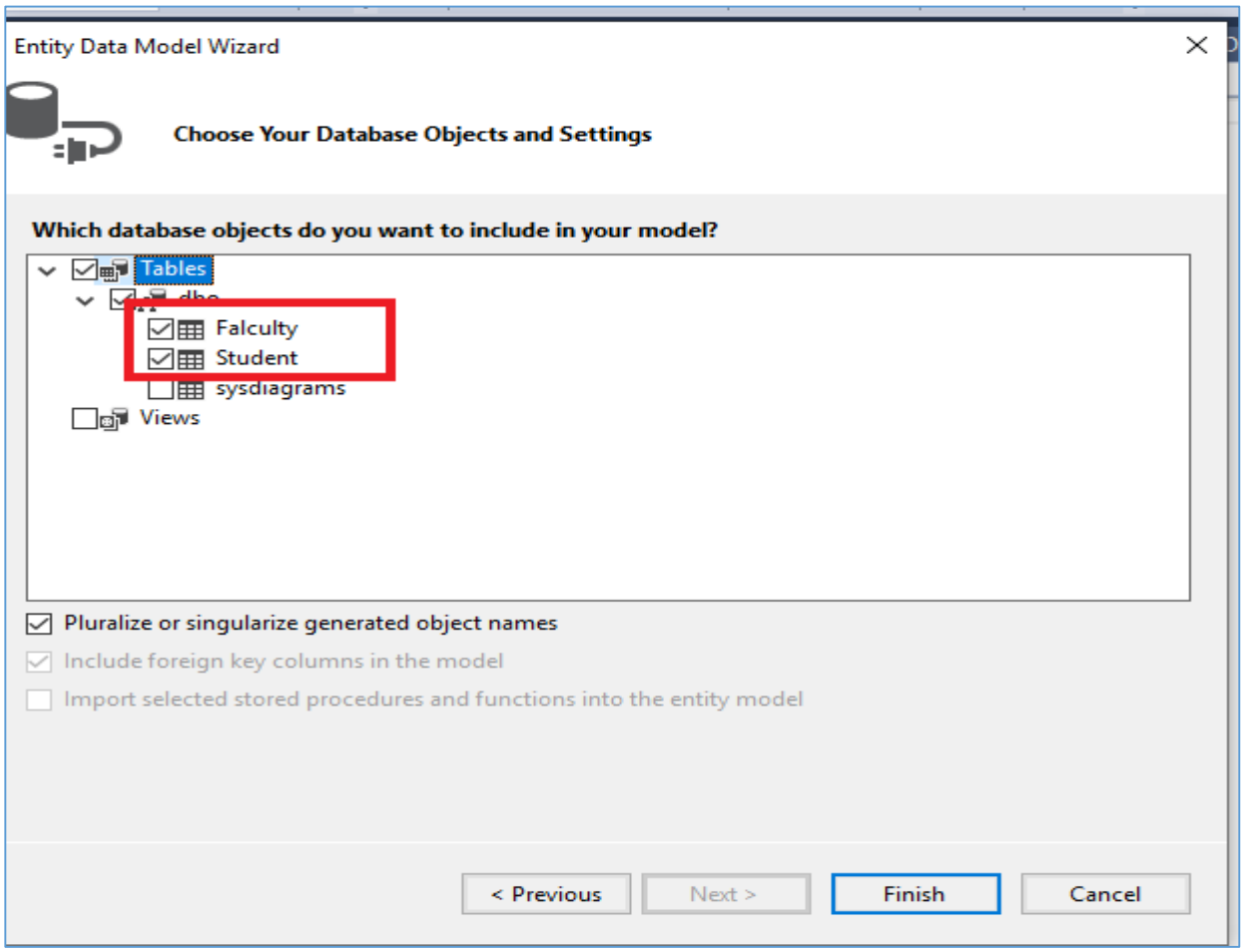
Tìm cơ sở dữ liệu Student ở SQL để trỏ database name vào



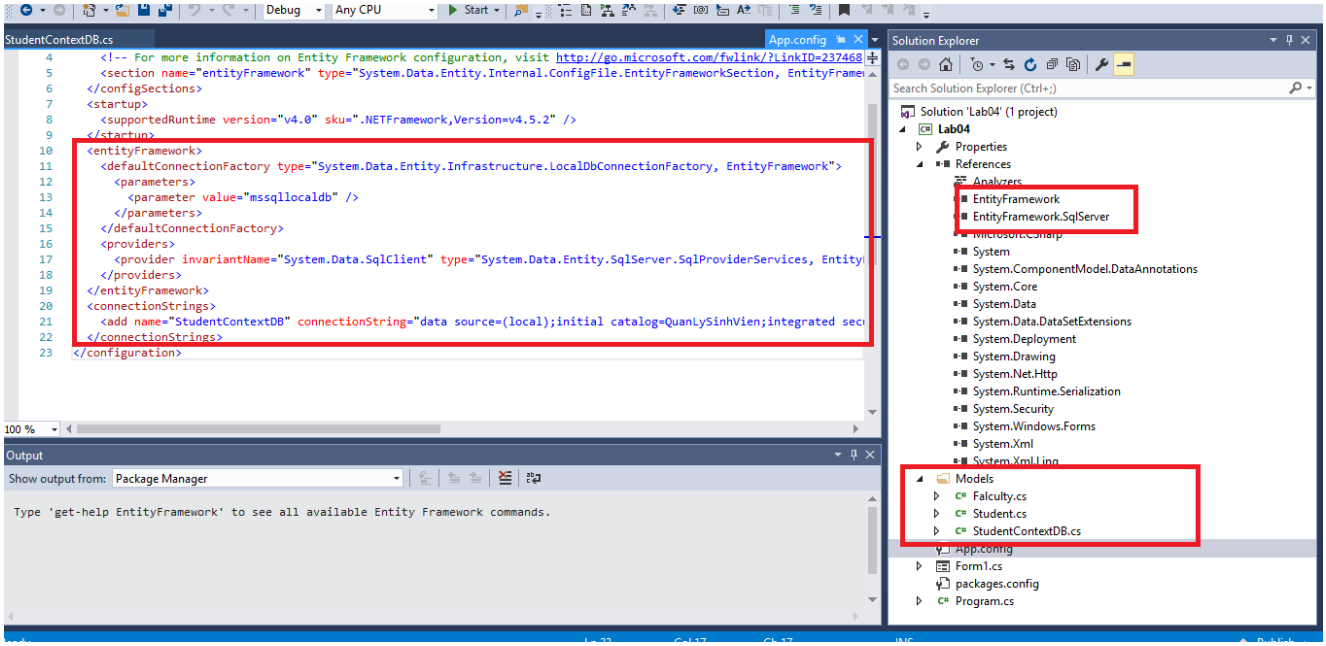
Chọn Next để tiếp tục tạo



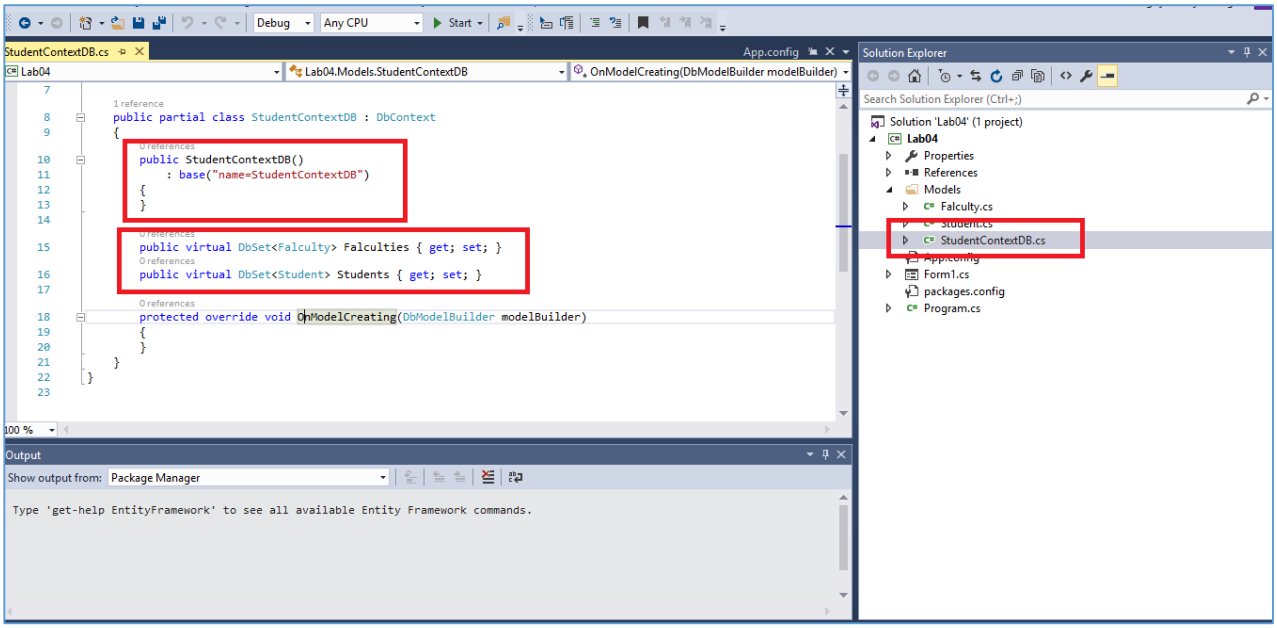
Sau đó chọn các bảng muốn tạo object



Sau khi Finish Entity FrameWork đã tạo cho chúng ta các class tương ứng như trong cơ sở dữ liệu



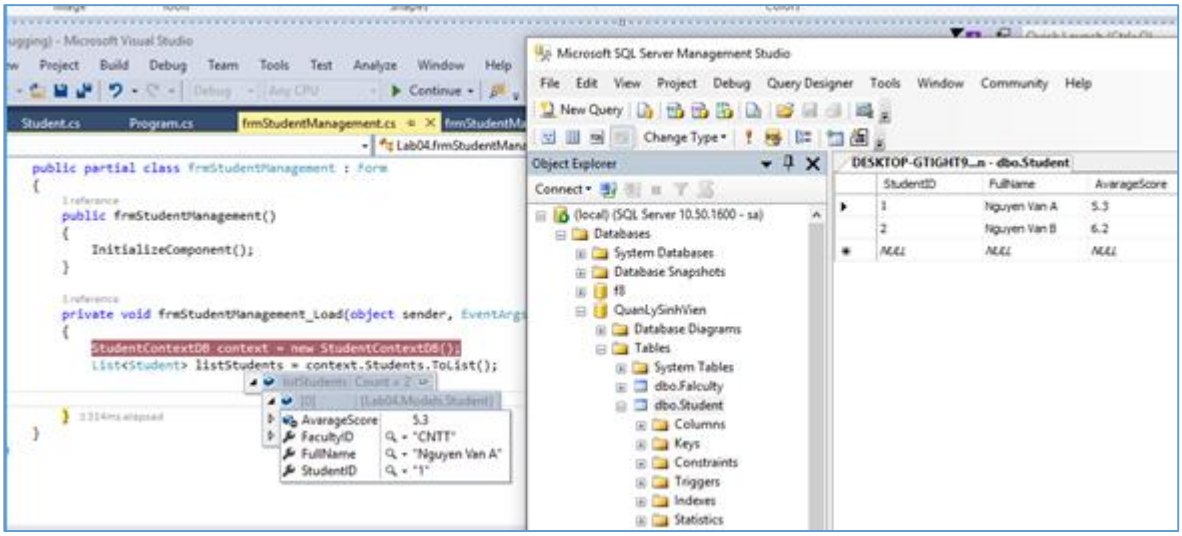
Ở File StudentContextDB.cs chứa tập hợp DataSet các table



✓ Sử dụng

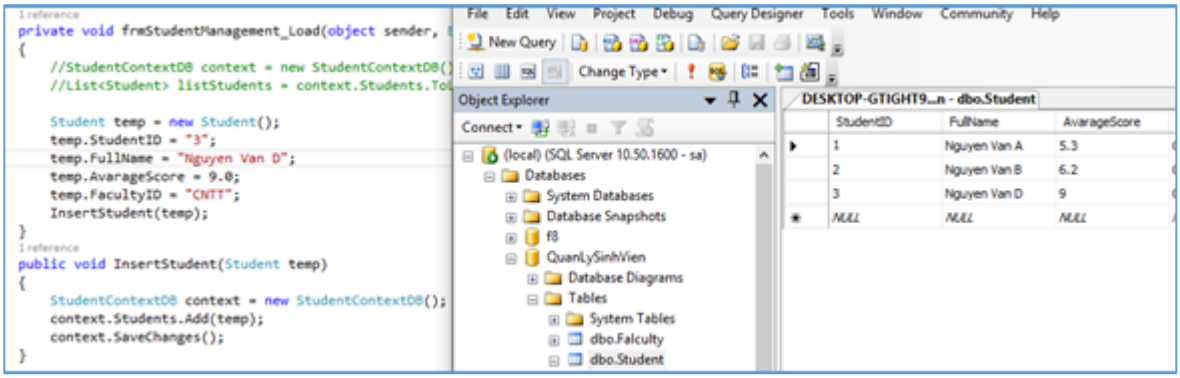
- Lấy tất cả các dữ liệu từ bảng sinh viên (Thử chạy ở form_load)

```
StudentContextDB context = new StudentContextDB();
List<Student> listStudents = context.Students.ToList();
```



- Viết hàm Insert 1 Student: Đưa đối tượng vào DbSet

```
public void InsertStudent(Student temp)
{
        StudentContextDB context = new StudentContextDB();
        context.Students.Add(temp);  // đưa đối tượng temp vào DBSet sinh Viên
        context.SaveChanges();       // Lưu thay đổi
 }
```



- Viết hàm Update: Cần lấy lại thông tin update

```
public void UpdateStudent(Student temp)
            {
                StudentContextDB context = new StudentContextDB();
                Student item = context.Students.FirstOrDefault(p => p.StudentID ==
    temp.StudentID);  //lay ra lai thong tin cũ
               if (item != null)
               {
                   //cập nhật
                    item.FullName = temp.FullName;  //muon thay doi Name
                    item.AverageScore = temp.AverageScore; // update score
                    //...
                    context.SaveChanges();       // Lưu thay đổi
                }
        }
```

- Viết hàm Delete: Cần remove đối tượng khỏi DbSet

```
    public void DeleteStudent(Student temp)
    {
         StudentContextDB context = new StudentContextDB();
         Student item = context.Students.FirstOrDefault(p => p.StudentID ==
        temp.StudentID);  //lay ra lai thong tin cũ

         if (item != null)
         {
               context.Students.Remove(temp);
               context.SaveChanges();       // Lưu thay đổi
         }
    }
```

Bước 2: Thiết kế và lập trình

Bài Tập 2: Sử dụng công nghệ **ADO.NET** để làm bài tập 1

Hướng Dẫn Sử dụng công nghệ ADO.NET

- ✓ Sử dụng chuỗi connectString trong config App.config để dễ dàng chỉnh sửa thay đổi.

```
<connectionStrings>
          <add name="DSStudentConnectString" connectionString="data
      source=(local);initial catalog=QuanLySinhVien;integrated
      security=True;MultipleActiveResultSets=True;"
      providerName="System.Data.SqlClient"/>
 </connectionStrings>
```

Để đọc được chuỗi này ta add thêm thư viện ***System.Configuration*** có sẵn của FrameWork

✓ Sử dụng **SqlConnection**, **SqlCommand**, **SqlDataReader**… để thao tác với cơ sở dữ liệu trong thư viện `using System.Data.SqlClient;`

- Ví dụ hàm Lấy tất cả danh sách sinh viên trong bảng Student được viết như sau

```
public static List<Student> GetAllStudent()
{
                    List<Student> listStudent = new List<Student>();
                    string connectionString =
       ConfigurationManager.ConnectionStrings["DSStudentConnectString"].ConnectionString;
                    string queryString = "Select * from Student";
                    using (SqlConnection connection = new SqlConnection(connectionString))
                    {
                        SqlCommand command = new SqlCommand(queryString, connection);
                        command.CommandType = CommandType.Text;
                        connection.Open();

                        using (SqlDataReader objReader = command.ExecuteReader())
                        {
                            if (objReader.HasRows)
                            {
                               while (objReader.Read())
                                {
                                   Student temp = new Student();
                                    string studentID =
       objReader.GetString(objReader.GetOrdinal("StudentID"));
                                   if (studentID != null)
                                   {
                                       temp.StudentID = studentID;
                                       temp.FullName = objReader["FullName"].ToString();
                                       string averageScore =
       objReader["AverageScore"].ToString();
                                       if (averageScore != null)
                                           temp.AverageScore =
       Convert.ToDouble(averageScore);

                                  string falcutyID = objReader["FacultyID"].ToString();
                                       if (falcutyID != null)
                                           temp.FacultyID = Convert.ToInt32(falcutyID);
                                   }

                                   listStudent.Add(temp);
                               }
                           }
                        }

                    }
                    return listStudent;
}
```

- Viết hàm ExcuteNonQuery để thực hiện 1 lệnh câu lệnh SQL (commandText)

```
public void ExcuteNonQuery(string commandText)
  {
```

```
 string connectionString =
ConfigurationManager.ConnectionStrings["DSStudentConnectString"].ConnectionString;

    using (SqlConnection connection = new SqlConnection(connectionString))
     {
                        SqlCommand command = new SqlCommand(commandText, connection);
                        command.CommandType = CommandType.Text;
                        connection.Open();
                        command.ExecuteNonQuery();
     }

}
```

- Viết hàm Insert, Update, Delete cho Sinh viên

```
public void InsertStudent()
{
  string queryString = @"INSERT INTO Student
           ([StudentID]
           ,[FullName]
           ,[AverageScore]
           ,[FacultyID])
     VALUES('{0}' ,'{1}',{2},{3})";
  queryString = string.Format(queryString, this.StudentID, this.FullName,
this.AverageScore, this.FacultyID);
 ExcuteNonQuery(queryString);
}
public void UpdateStudent(string studentUpdateID)
{
            string queryString = @"UPDATE Student
          Set [FullName] = '{1}'
             ,[AverageScore] = {2}
             ,[FacultyID] = {3}
          WHERE StudentID= {0}";
            queryString = string.Format(queryString, this.StudentID, this.FullName,
this.AverageScore, this.FacultyID);
            ExcuteNonQuery(queryString);
 }
public void DeleteStudent()
{
            string queryString = @"DELETE FROM Student
          WHERE StudentID= {0}";
            queryString = string.Format(queryString, this.StudentID);
            ExcuteNonQuery(queryString);
}
```

Bài tập 3: Viết tiếp ở bài tập 1 hoặc bài tập 2

- Thêm 1 form mới là ***frmFalculty*** có đủ các chức năng thêm, xóa, sửa, hiện thị thông tin khoa tương tự.
- Viết sự kiện ở DataGridView khi người dùng chọn vào dòng nào thì thông tin sẽ được hiện thị lại ngay bên cạnh ở phần thông tin (cho cả 2 form Khoa và Sinh Viên)
- Khi người dùng nhập vào 1 tên khoa chưa tồn tại trong cơ sở dữ liệu. Khi Insert/Update có cảnh báo sẽ thêm thông tin khoa

  Click Yes sẽ insert thêm khoa vào cơ sở dữ liệu, với FacultyID = max(FacultyID) + 1
- Sử dụng ***Transaction*** để làm yêu cầu trên (đảm bảo toàn vẹn dữ liệu, không có trường hợp insert thông tin sinh viên với mã khoa không tồn tại trong cơ sở dữ liệu)

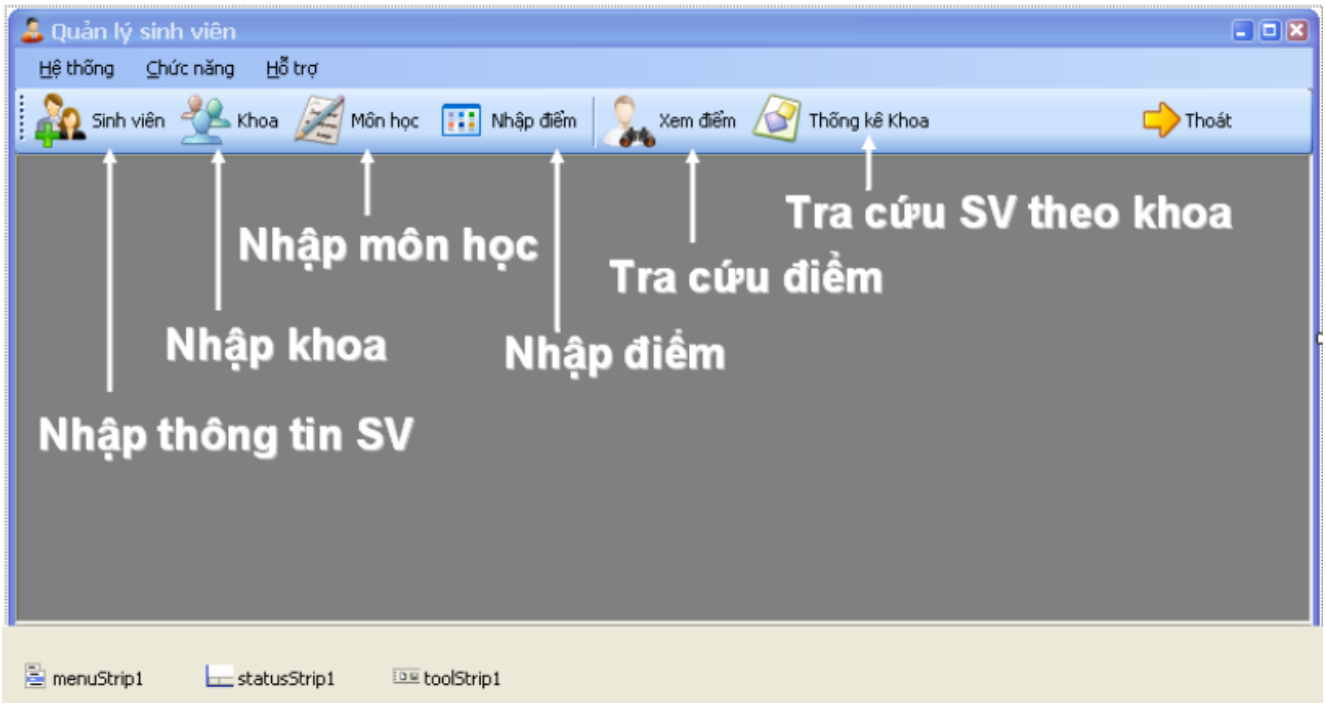
✓ Hướng dẫn sử dụng Transaction trên EntityFrameWork

```
StudentContextDB context = new StudentContextDB();
        using (DbContextTransaction tran = context.Database.BeginTransaction())
        {
            try
            {
                //insert Khoa
                context.Falculties.Add(tempFalculty);
                //insert sinh vien
                context.Students.Add(tempStudent);
                //saves all
                context.SaveChanges();
                //commit transaction
                tran.Commit();
            }
            catch (Exception ex)
            {
                //Rollback transaction if exception occurs
                tran.Rollback();
                throw ex;
            }
        }
```

✓ Hướng dẫn sử dụng Transaction trên ADO.NET cho 2 câu `queryString1, queryString2`

```
  string connectionString =
ConfigurationManager.ConnectionStrings["DSStudentConnectString"].ConnectionString;
            using (SqlConnection conn = new SqlConnection(connectionString))
            {
                SqlTransaction transaction = null;
                try
                {
                    conn.Open();
                    transaction = conn.BeginTransaction();
                    using (SqlCommand cmd = new SqlCommand(queryString1, conn, transaction))
                    {
                        cmd.ExecuteNonQuery();
                    }
                    using (SqlCommand cmd = new SqlCommand(queryString2, conn, transaction))
                    {
                        cmd.ExecuteNonQuery();
                    }
                    transaction.Commit();
                }
                catch (Exception ex)
                {
                    transaction.Rollback();
                    throw ex;
                }
}
```

Bài tập 4: Tạo project Lab04-04 Xây dựng chương trình quản lý sinh viên Windows Application MDI



Với Database có chỉnh sửa so với bài tập 1 như sau

Student (***StudentID***, *FullName, Birthday, Address, Mobile, FalcultyID*)

Birthday: DATETIME: Ngày tháng năm sinh của sinh viên

Address: NVARCHAR (250): Địa chỉ của sinh Viên

Mobile: NVARCHAR(50): Điện thoại liên lạc

Falculty(***FalcultyID***, *FacultyName)*

Course *(**CourseID**, CourseName)*

CourseID: NVARCHAR(20): Mã môn học

CourseName: NVARCHAR(250): Tên Môn học

CourseScore(***StudentID, CourseID***, *Score)*

Score**:** Double**:** Điểm môn học (0.0 -> 10.0)

- ✓ Chức năng nhập liệu:Thông tin sinh viên, Danh mục các khoa, Danh mục các môn học, Điểm của các sinh viên
- ✓ Chức năng thống kê/báo cáo: Xem danh sách sinh viên với điểm tương ứng, Xem sinh viên trong từng khoa tương ứng

----------Hết Lab 04----------