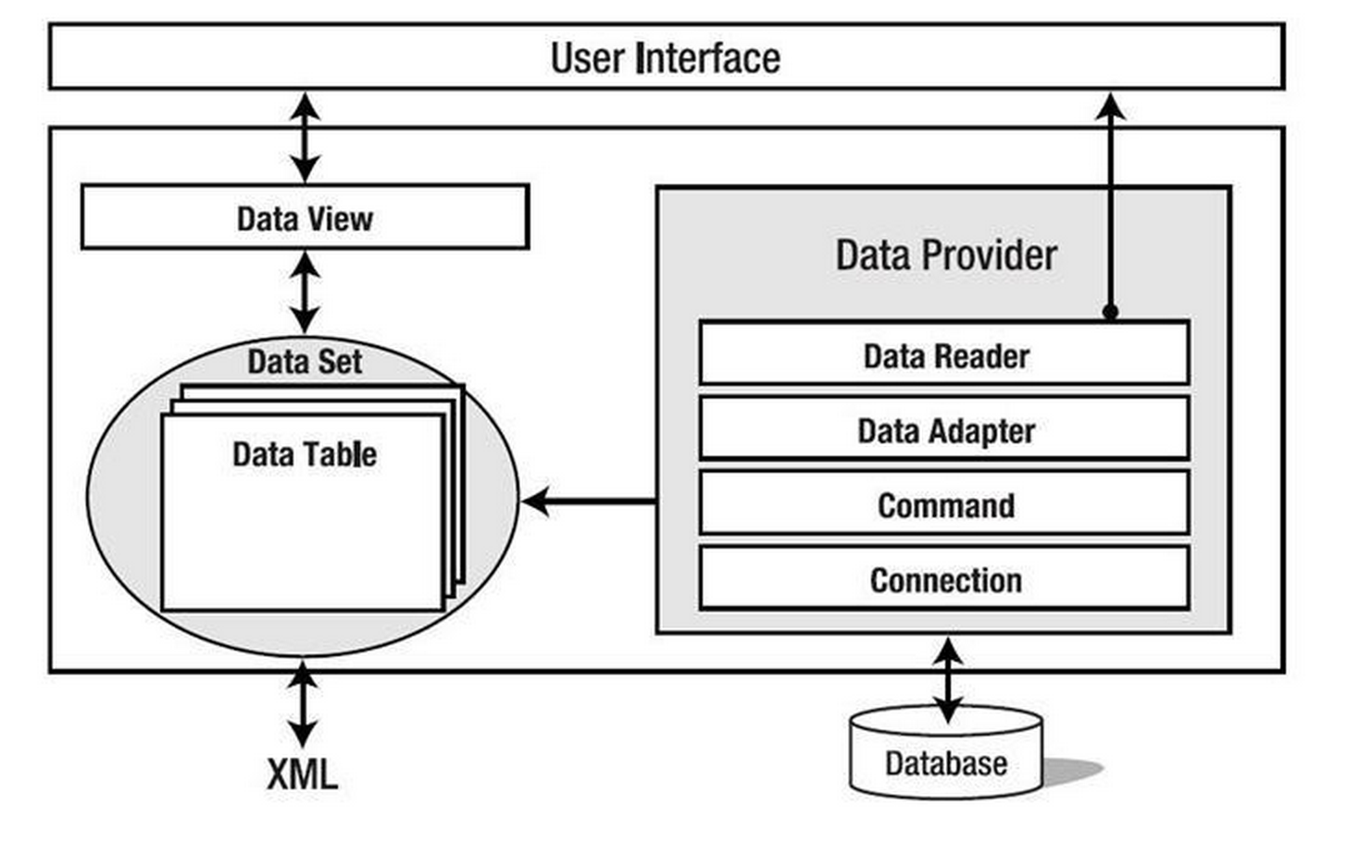
**ADO.NET**

****

ADO.NET là một tập hợp hướng đối tượng các thư viện cho phép bạn tương tác với nguồn dữ liệu (data sources). Thường thì nguồn dữ liệu là một cơ sở dữ liệu, nhưng có thể cũng là 1 tập tin Text, bảng tính Excel hoặc 1 tập tin XML.

**Data Providers**

Ứng với mỗi cơ sở dữ liệu khác nhau thì ADO.NET cho phép chúng ta tương tác cũng khác nhau. Một Data Provider được sử dụng cho việc kết nối đến cơ sở dữ liệu, đọc, nhận, cập nhật,…trong DataSet và trong cơ sở dữ liệu. Tuy nhiên do có khá nhiều nguồn dữ liệu nên nảy sinh ra các giao thức cũng khác nhau; vì thế để giao tiếp được với cơ sở dữ liệu chúng ta phải sử dụng giao thức phù hợp với nguồn dữ liệu đó.

Các giao thức như ODBC, OleDb, hay khác hơn là các giao thức khác thông qua thư viện .NET của lớp ADO.NET

Một số giao thức thường dùng như:

|  |  |  |
| --- | --- | --- |
| **Provider Name** | **Tiền tố API** | **Data Source** |
| ODBC Data Provider | Odbc | Dùng cho cơ sở dữ liệu cũ |
| OleDb Data Provider | OleDb | Dùng cho cơ sở dữ liệu như Access, Excel |
| Oracle Data Provider | Oracle | Dùng cho cơ sở dữ liệu Oracle |
| SQL Data Provider | Sql | Dùng cho cơ sở dữ liệu SQL Server |
| Borland Data Provider | Bdp | Dùng cho cơ sở dữ liệu Interbase, SQL Server, IBM DB2, Oracle |

**ADO.NET Objects**

ADO.NET bao gồm nhiều đối tượng giúp chúng ta làm việc với dữ liệu. Phần này sẽ giới thiệu các đối tượng chính thường sử dụng. Một số đối tượng đó là:

**SqlConnection Object**

là đối tượng kết nối với cơ sở dữ liệu. Kết nối này nhằm xác định máy chủ database server, tên cơ sở dữ liệu Database name, tên người dùng user name, mật khẩu password, và các thông số khác được yêu cầu để kết nối với cơ sở dữ liệu. Bảng sau đây hiển thị những thuộc tính và phương thức hay sử dụng nhất của một đối tượng kết nối

|  |  |
| --- | --- |
| Tên | Mô tả |
| ConnectionString | Cung cấp thông tin như datasource, tên cơ sở dữ liệu, được sử dụng để thiết lập kết nối với một cơ sở dữ liệu |
| Open() | Mở một kết nói với datasource được khai báo tại ConnectionString |
| Close() | Được sử dụng để đóng kết nối với data source |
| State | Được sử dụng để kiểm tra trạng thái của một kết nối. 0: kết nối đang đóng, 1: kết nối đang mở. |
|  |  |

**SqlCommand Object**

Quá trình tương tác với cơ sở dữ liệu nghĩa là bạn phải có các hành động cho nó, tương ứng với thực hiện một đối tượng lệnh. Đối tượng này thực thi các câu truy vấn SQL , Store Procedure, hoặc chỉ định tham chiếu đến một đối tượng SqlDataAdapter

**SqlDataReader Object**

Nhiều thao tác dữ liệu đòi hỏi bạn chỉ lấy một luồng dữ liệu để đọc. Đối tượng Data Reader cho phép bạn lấy được kết quả của một câu lệnh SELECT từ một đối tượng command. Để tăng hiệu suất, dữ liệu trả về từ một data reader là một luồng dữ liệu fast forward-only. Có nghĩa là bạn chỉ có thể lấy dữ liệu từ luồng theo một thứ tự nhất định. Mặc dù điều này có lợi về mặt tốc độ, nhưng nếu bạn cần phải thao tác dữ liệu, thì một DataSet sẽ là một đối tượng tốt hơn để làm việc.

Các thuộc tính của đối tượng DataReader:

|  |  |
| --- | --- |
| Tên | Mô tả |
| Read() | Phương thức được sử dụng để đọc mọt dòng. |
| Close() | Được sử dụng để đóng đối tượng DataReader. |
| NextResult | Được sử dụng để di chuyển đến  hàng dữ liệu tiếp theo trong trường hợp câu lệnh truy vấn trả lại nhiều dòng kết quả |
|  |  |

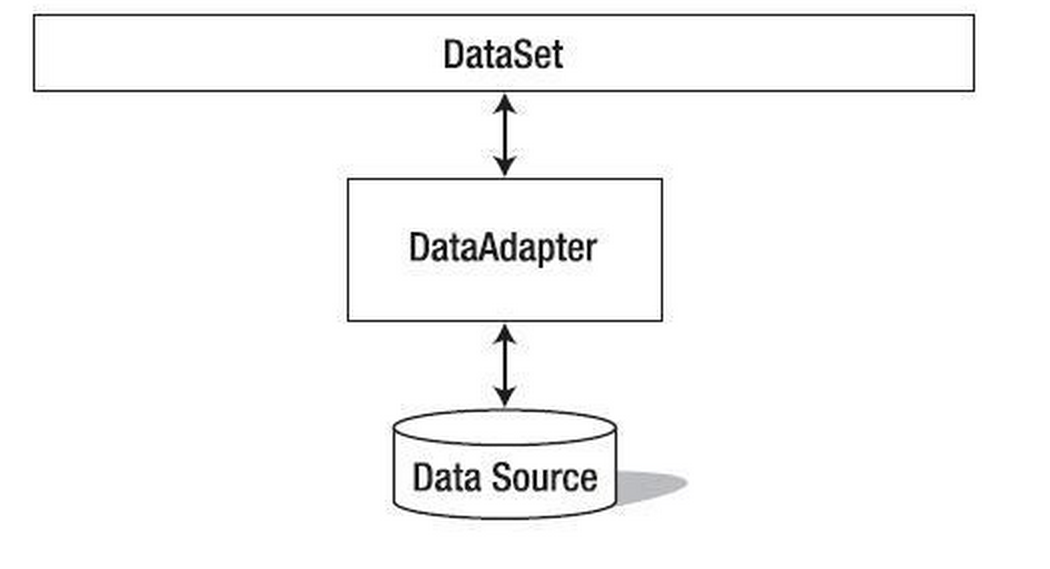
**DataSet Object**

Đối tượng DataSet là một thể hiện của dữ liệu trong bộ nhớ. Chúng chứa nhiều đối tượng DataTable, bên trong DataTable lại có nhiều column và row, giống như các database table thông thường. Thậm chí bạn có thể định nghĩa dữ liệu giữa các table để tạo các quan hệ parent-child. DataSet được thiết kế đặc biệt để giúp quản lý dữ liệu trong bộ nhớ và để hỗ trợ các thao tác **không cần kết nối** trên dữ liệu. DataSet là một đối tượng được dùng bởi tất cả Data Provider, đó là lý do tại sao nó không có một Data Provider prefix trong tên gọi.

**SqlDataAdapter**

Đôi lúc dữ liệu mà bạn làm việc là read-only và bạn ít khi cần thay đổi dữ liệu nguồn. Vài trường hợp cần lưu trữ tạm dữ liệu trong bộ nhớ để hạn chế truy xuất đến cơ sở dữ liệu. DataAdapter giúp bạn quản lý dữ liệu trong chế độ ngắt kết nối disconnect. DataAdapter sẽ đẩy vào DataSet khi đọc dữ liệu và thực hiện thay đổi tất cả dữ liệu vào Database.

DataAdapter chứa một tham chiếu đến đối tượng connection và mở/đóng kết nối tự động khi đọc và ghi dữ liệu vào database. Hơn nữa, DataAdapter chứa đối tượng command cho những thao tác SELECT, INSERT, UPDATE và DELETE. STORE PROCEDURE trên dữ liệu. Bạn sẽ có một DataAdapter được định nghĩa cho mỗi table trong một DataSet và nó sẽ quản lý các giao tiếp với database cho bạn. Tất cả những gì bạn cần làm là chỉ cho DataAdapter khi nào nạp hoặc ghi vào cơ sở dữ liệu.



**DATA SET**

DataSet trong ADO.NET là một bước phát triển lớn trong việc phát triển ứng dụng cơ sở dữ liệu đa hệ. Khi lấy và chỉnh sửa dữ liệu, duy trì liên tục kết nối tới Data Source trong khi chờ user yêu cầu thì rõ ràng là tốn tài nguyên máy rất nhiều.  
  
DataSet giúp ích ở đây rất lớn. Vì DataSet cho phép lưu trữ dữ liệu và chỉnh sửa tại ‘local cache’, hay gọi là offline mode. Có thể xem xét và **xử lý thông tin trong khi ngắt kết nối**. Sau khi chỉnh sửa và xem xong thì tạo một kết nối và update dữ liệu từ local vào Data Source.   
Dữ liệu trong DataSet được lưu trữ dưới dạng một Collection các Tables và bạn cần phải xử lý thông qua các lớp DataTable -> DataRow và DataColumn.

**DATA ADAPTER**  
Bạn chỉ cần tưởng tượng rằng : bạn có một cái bể nước (DataSource) , một cái máy bơm (DataAdapter) và một cái thùng đựng nước (DataSet). Thì khi lấy nước dùng cái bơm lấy nước từ bể, kiểm tra và lọc nước sau đó lại dùng cái bơm hút lại về cái bể nước. Đó chính là vai trò của cái bơm và DataAdapter tương tự như vậy.

**DATA SOURCE**

Là nguồn dữ liệu hay còn gọi là Cơ Sở Dữ Liệu.

**TẠO CƠ SỞ DỮ LIỆU NHƯ SAU**

**Mô tả:**

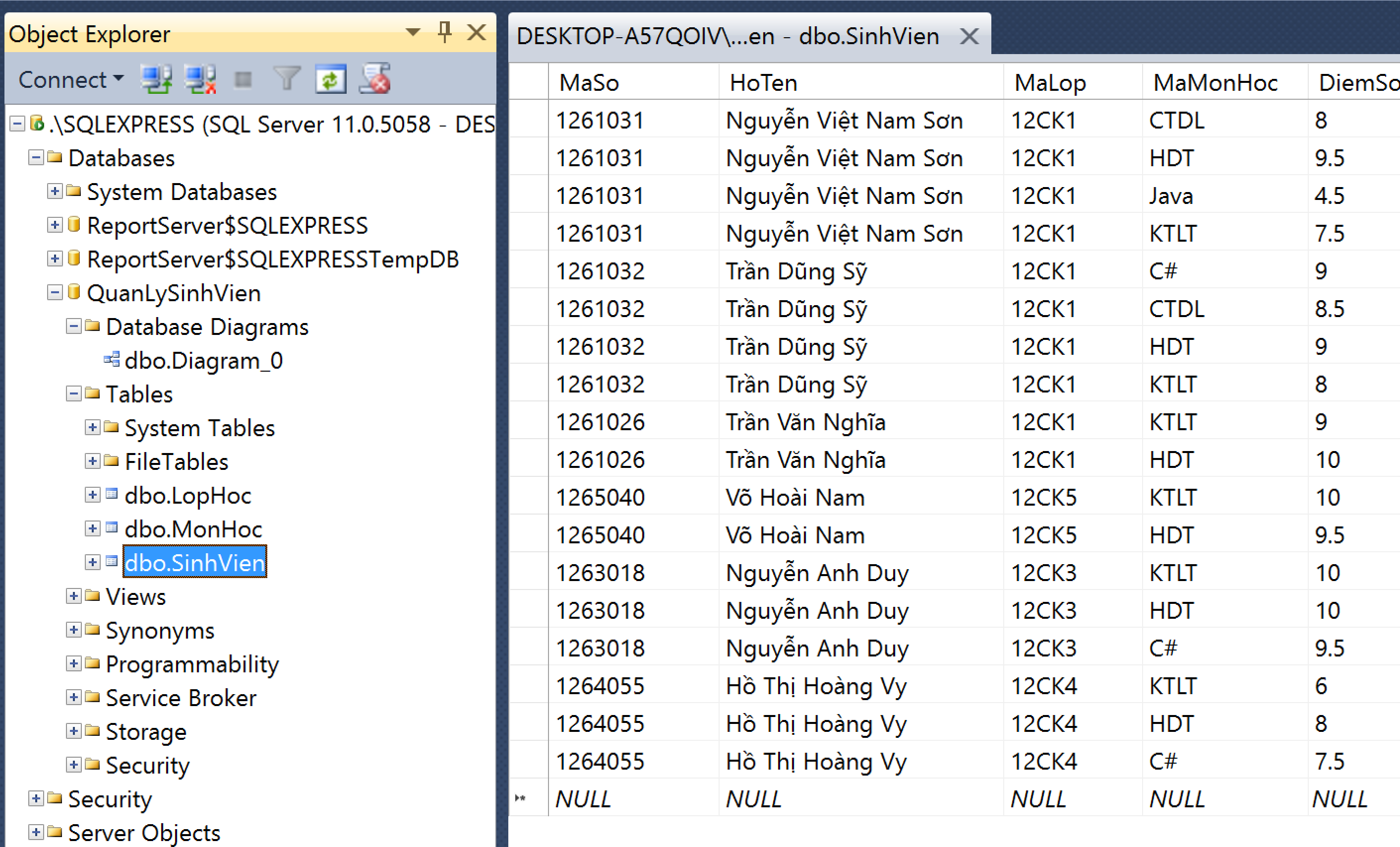
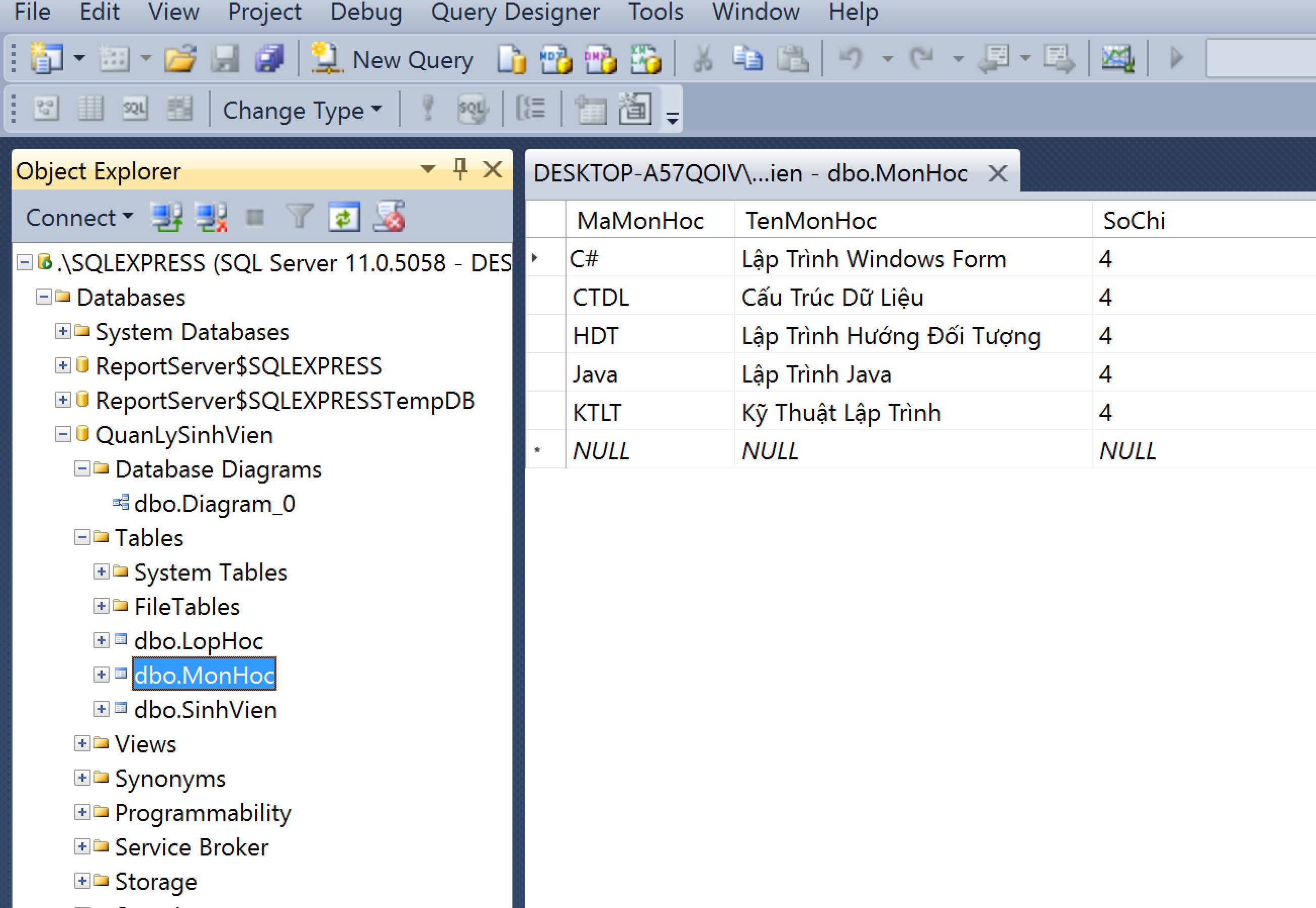
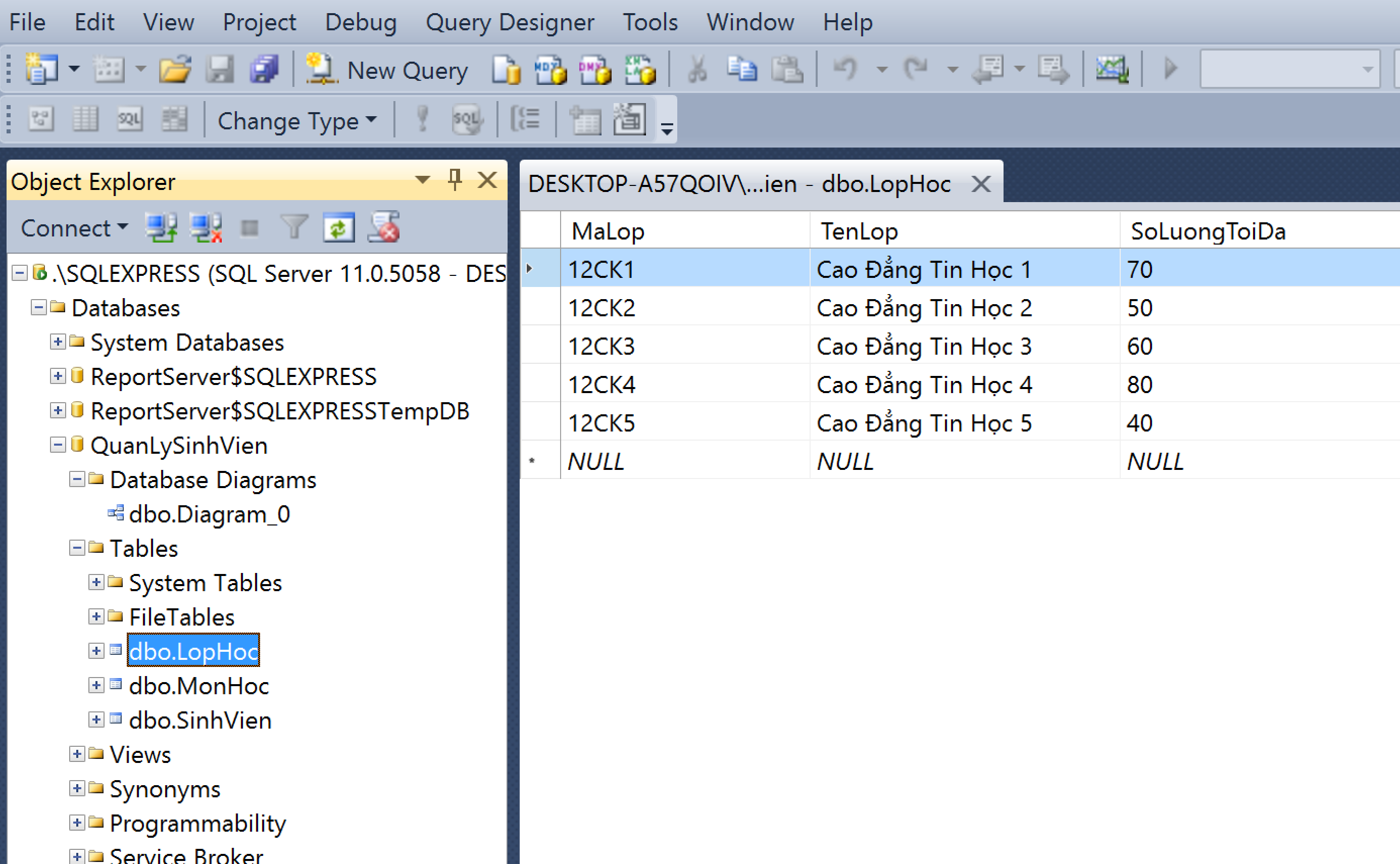
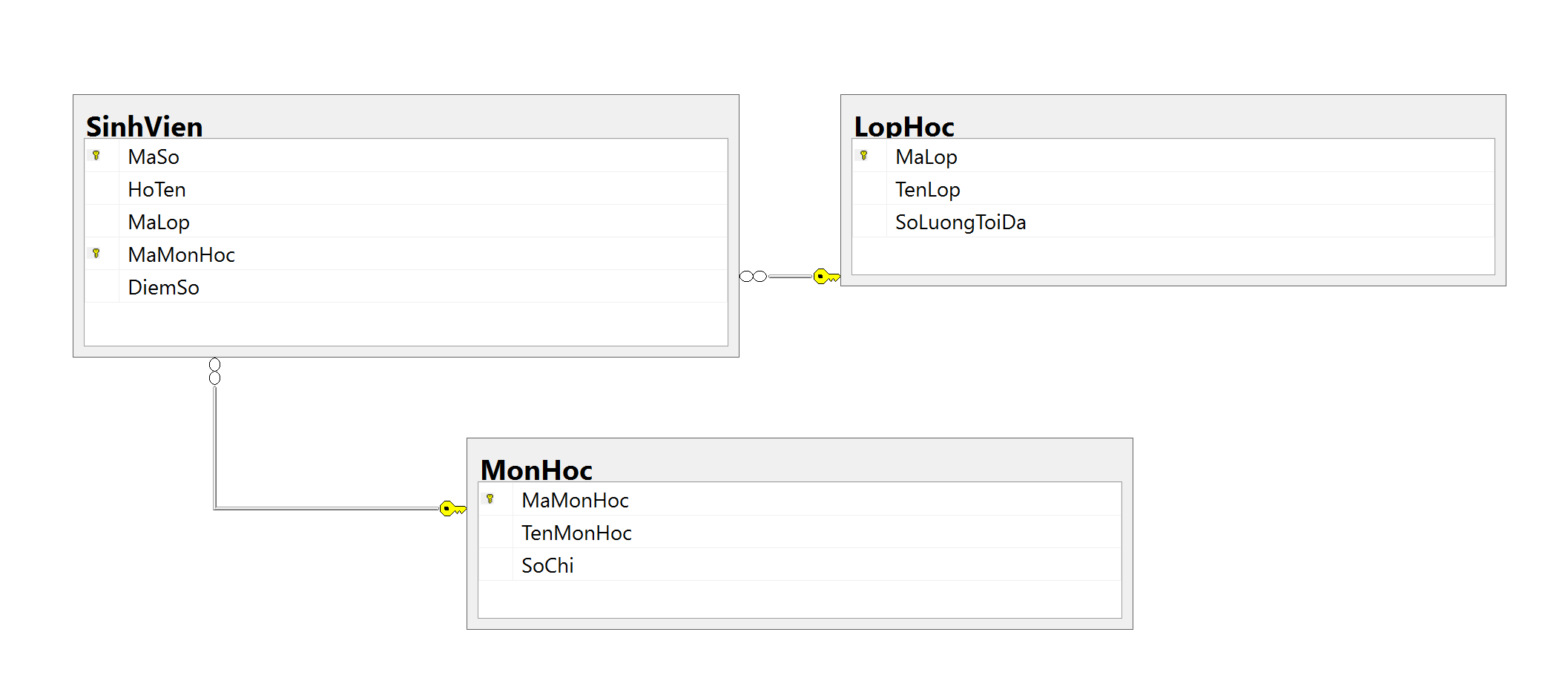
|  |
| --- |
| **LopHoc** |
| **MaLop (nchar(10)) => Khóa chính** |
| **TenLop (nvarchar(50))** |
| **SoLuongToiDa (int) // số lượng học sinh tối đa mà lớp có thể chứa** |

|  |
| --- |
| **MonHoc** |
| **MaMonHoc (nchar(10)) => Khóa chính** |
| **TenMonHoc (nvarchar(50))** |
| **SoChi (int) // số tín chỉ tương ứng của môn học** |

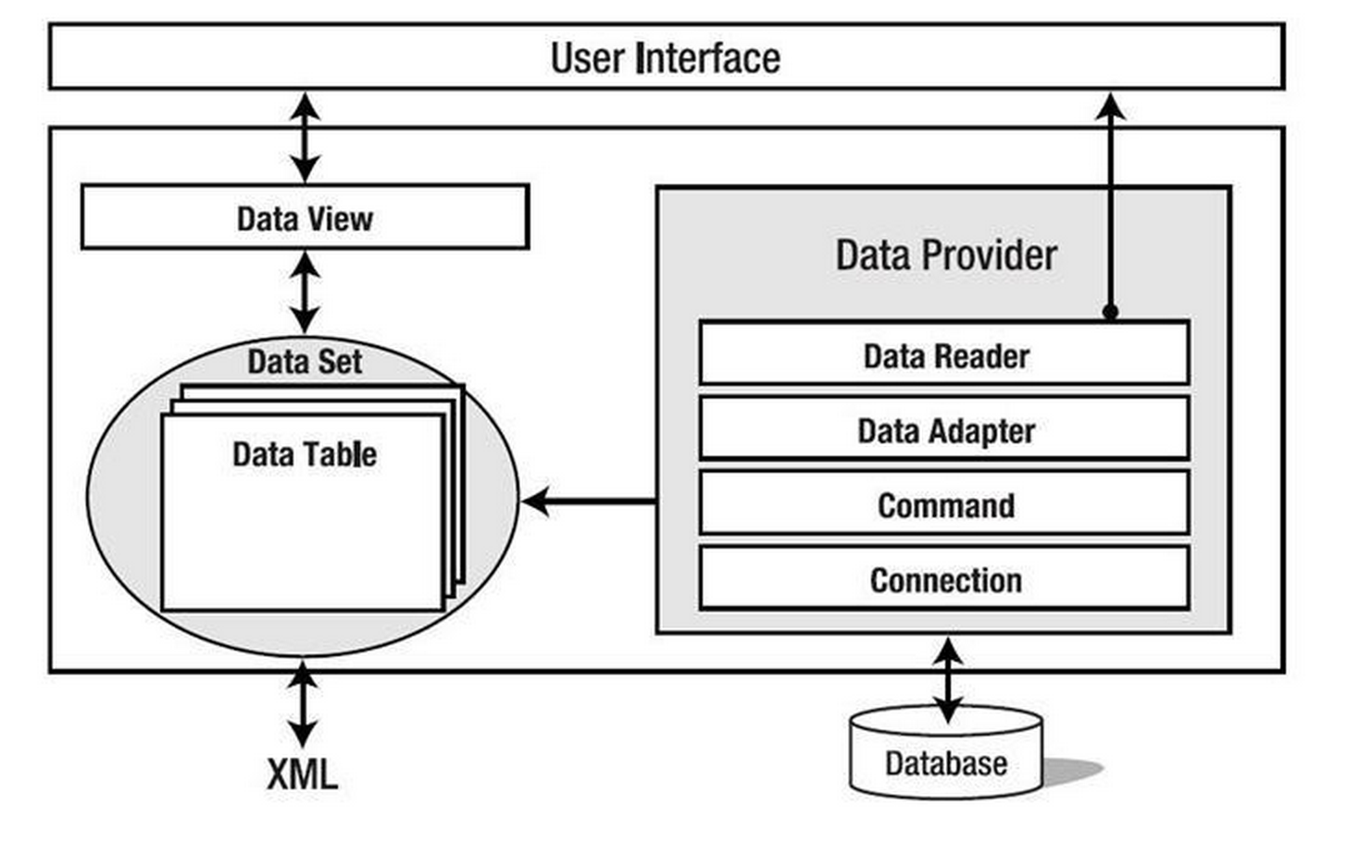
|  |
| --- |
| **SinhVien** |
| **MaSo (nchar(10)) => Khóa chính** |
| **HoTen (nvarchar(50))** |
| **MaLop (nchar(10)) => Khóa ngoại liên kết tới khóa chính của bảng LopHoc** |
| **MaMonHoc (nchar(10)) => Khóa chính cũng đồng thời là khóa ngoại liên kết tới khóa chính của bảng MonHoc** |
| **DiemSo (int) => Điểm tổng kết môn của sinh viên ứng với môn học đó.** |

**Giải thích tại sao bảng SinhVien lại cho MaMonHoc là khóa chính cũng đồng thời là khóa ngoại ? Đáng lý ra phải là khóa ngoại thôi chứ sao lại là khóa chính nữa ?**

**HÌNH ẢNH MÔ PHỎNG**

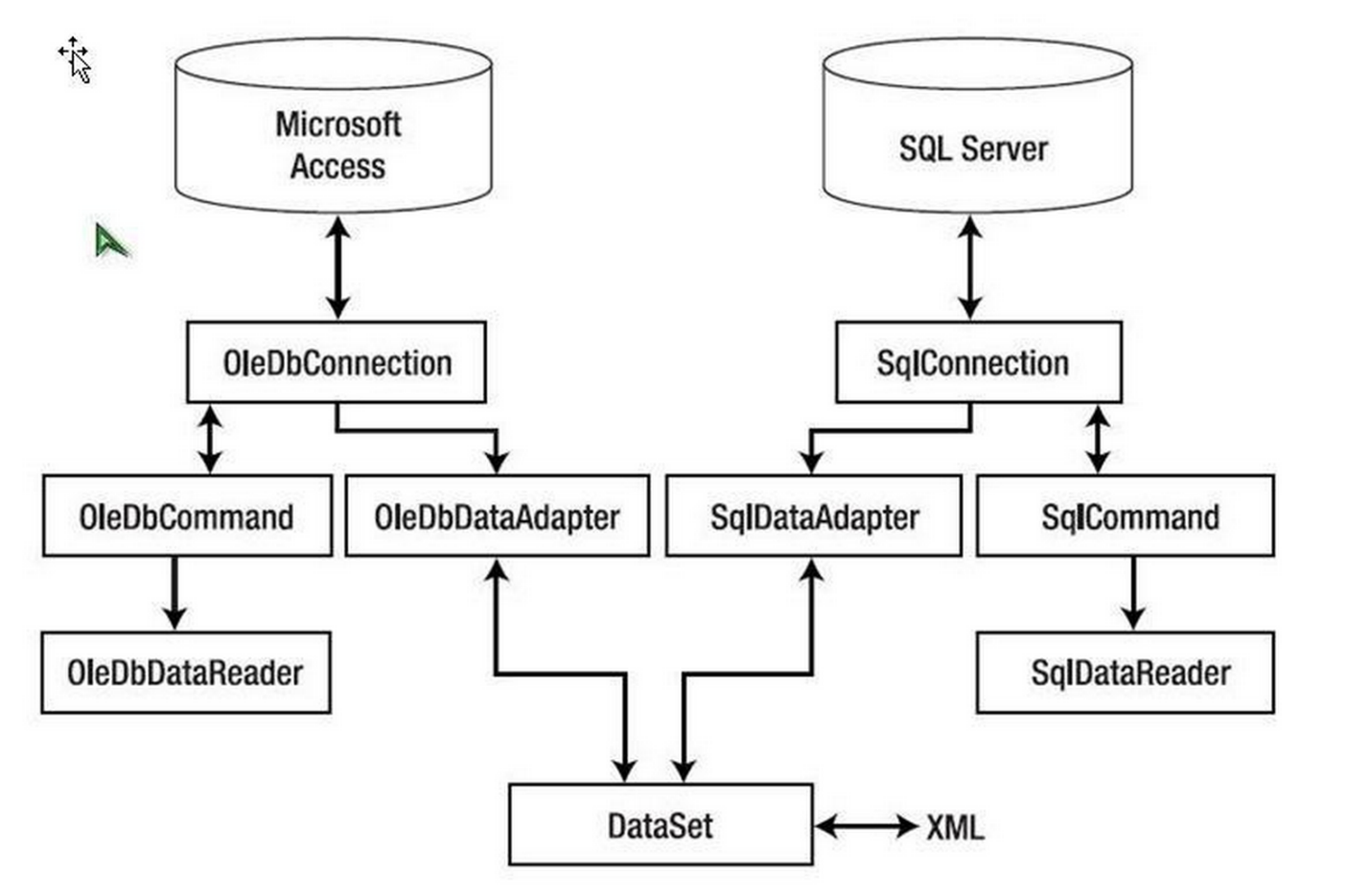


**CẤU TRÚC ADO.NET**

****

ADO.NET có 2 thành phần chính : data providers và datasets  
Có thể thấy mối quan hệ của chúng trên hình. Chú ý ‘DataView class’ không phải là một thành phần ‘data provider’. DataView được sử dụng phần lớn để gắn (bind) dữ liệu vào winform và webform.

DataSet là cơ sở dữ liệu được lưu trữ trong bộ nhớ chính (in-memory database).



Giải thích:

1. SqlCommand : thực thi SQL queries, câu lệnh hoặc lưu trữ thủ tục  
2. SqlConnection : tạo kết nối tới SQL Server  
3. SqlDataAdapter : cầu nối trung gian giữa dataset và data source  
4. SqlReader : cung cấp một data stream tới kết quả  
5. SqlError : lưu trữ thông tin về lỗi và cảnh cáo (warning)  
6. SqlException : các ngoại lệ trong trường hợp SQL Server lỗi và cảnh báo  
7. SqlParameter : tham sô biên command  
8. SqlTransaction : transaction của SQL Server

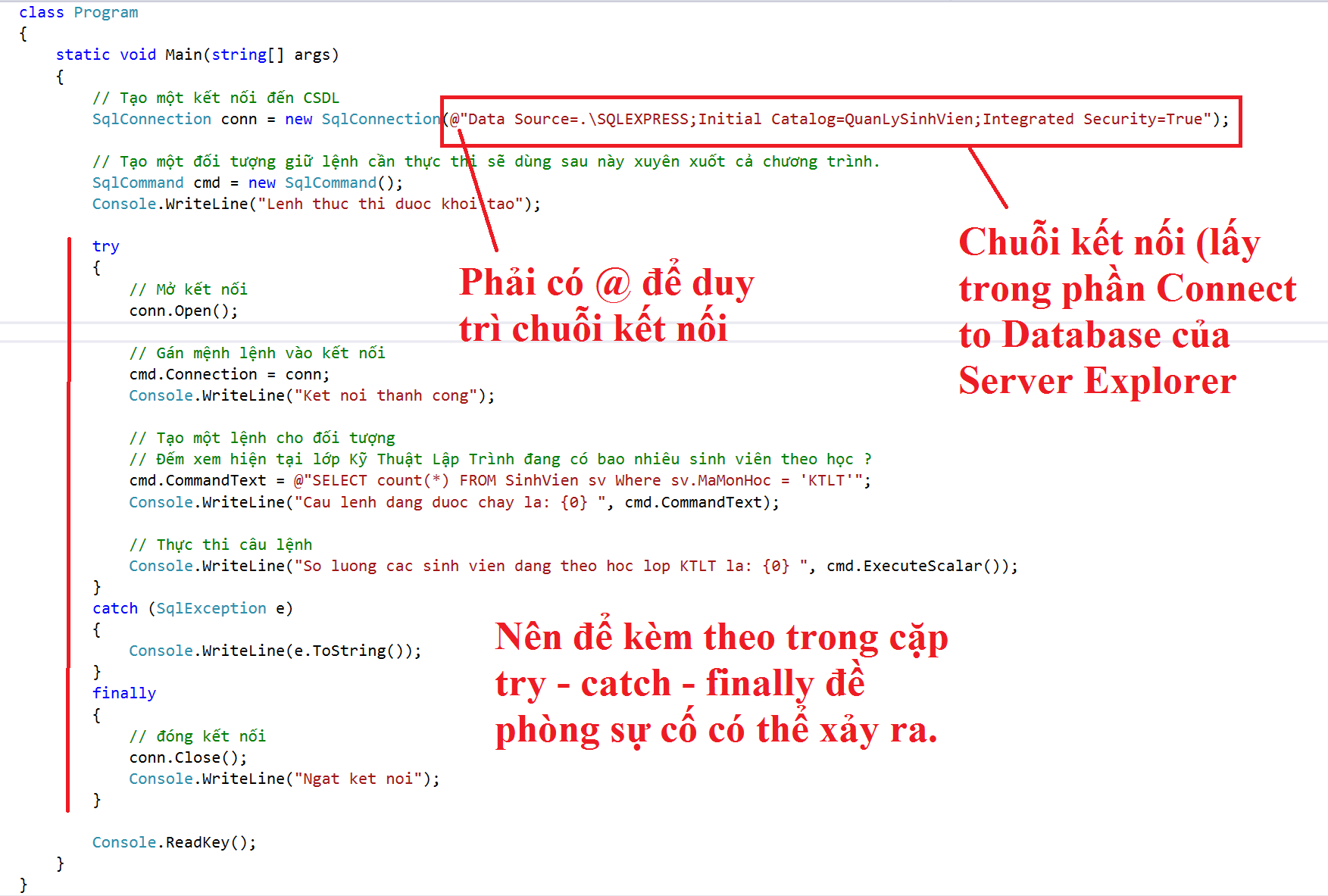
Các bước thao tác CSDL

* Bước 1: Tạo chuỗi kết nối cnStr
* Bước 2: Kết nối CSDL với đối tượng XXXConnection
* Bước 3: Tạo chuỗi strSQL thao tác CSDL
* Bước 4: Thực thi chuỗi strSQL với đối tượng XXXCommand hoặc XXXDataAdapter
* Bước 5: Đóng kết nối

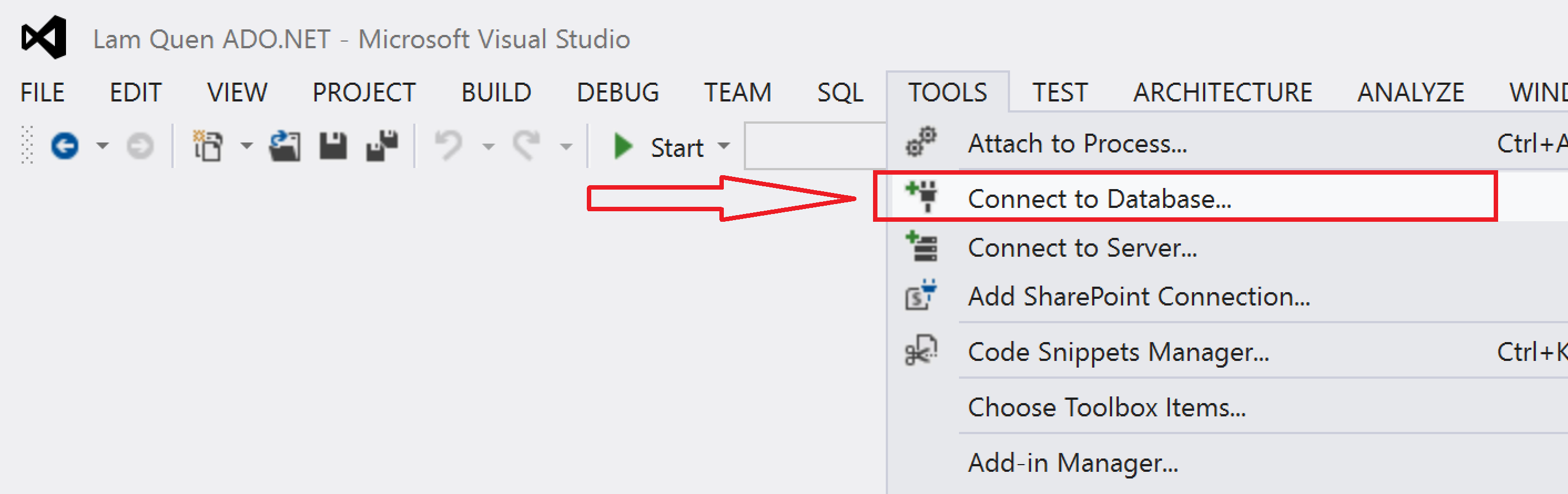
Chạy thử đoạn mã lệnh:

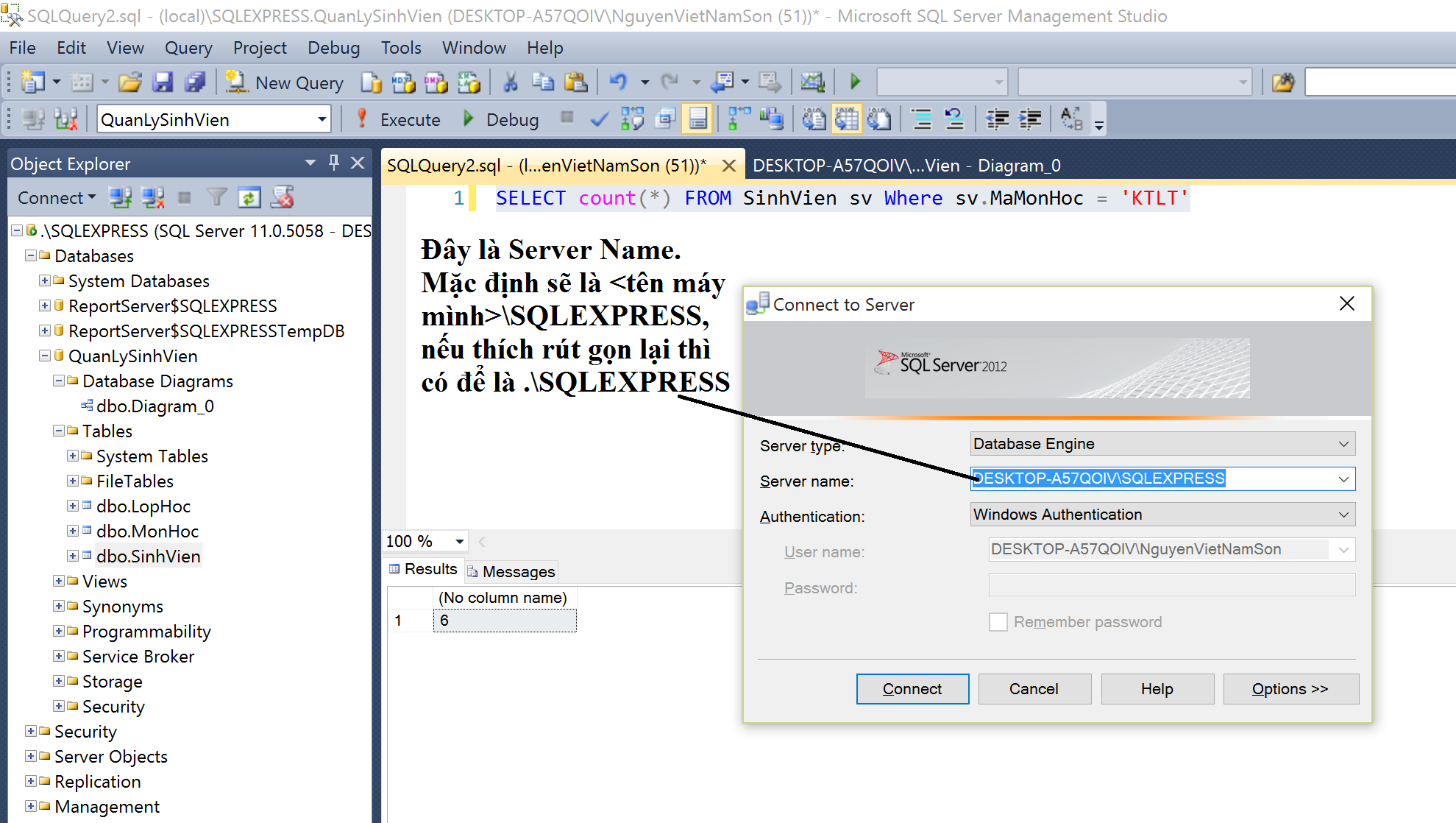
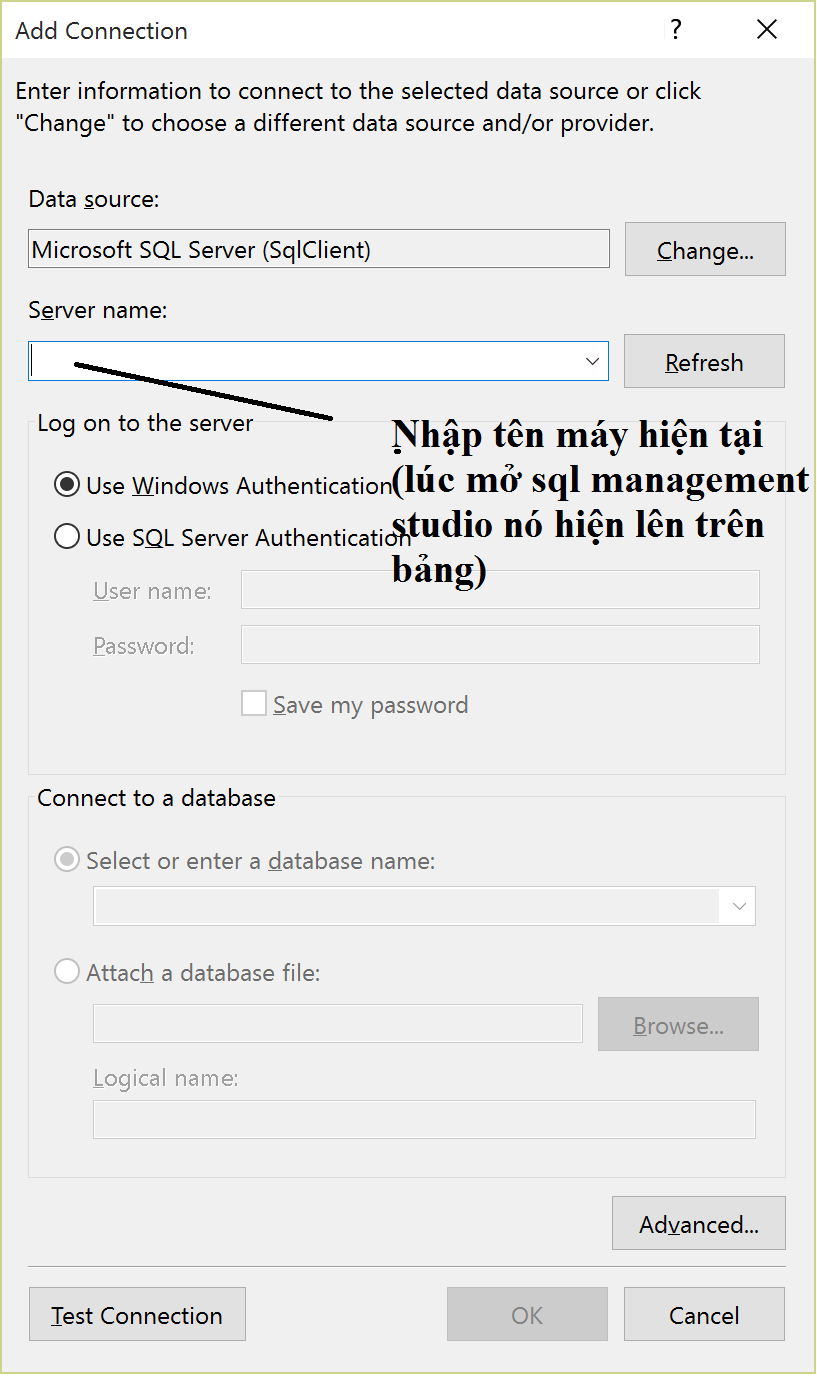
Đếm xem có bao nhiêu sinh viên đăng ký môn học Kỹ Thuật Lập Trình ?

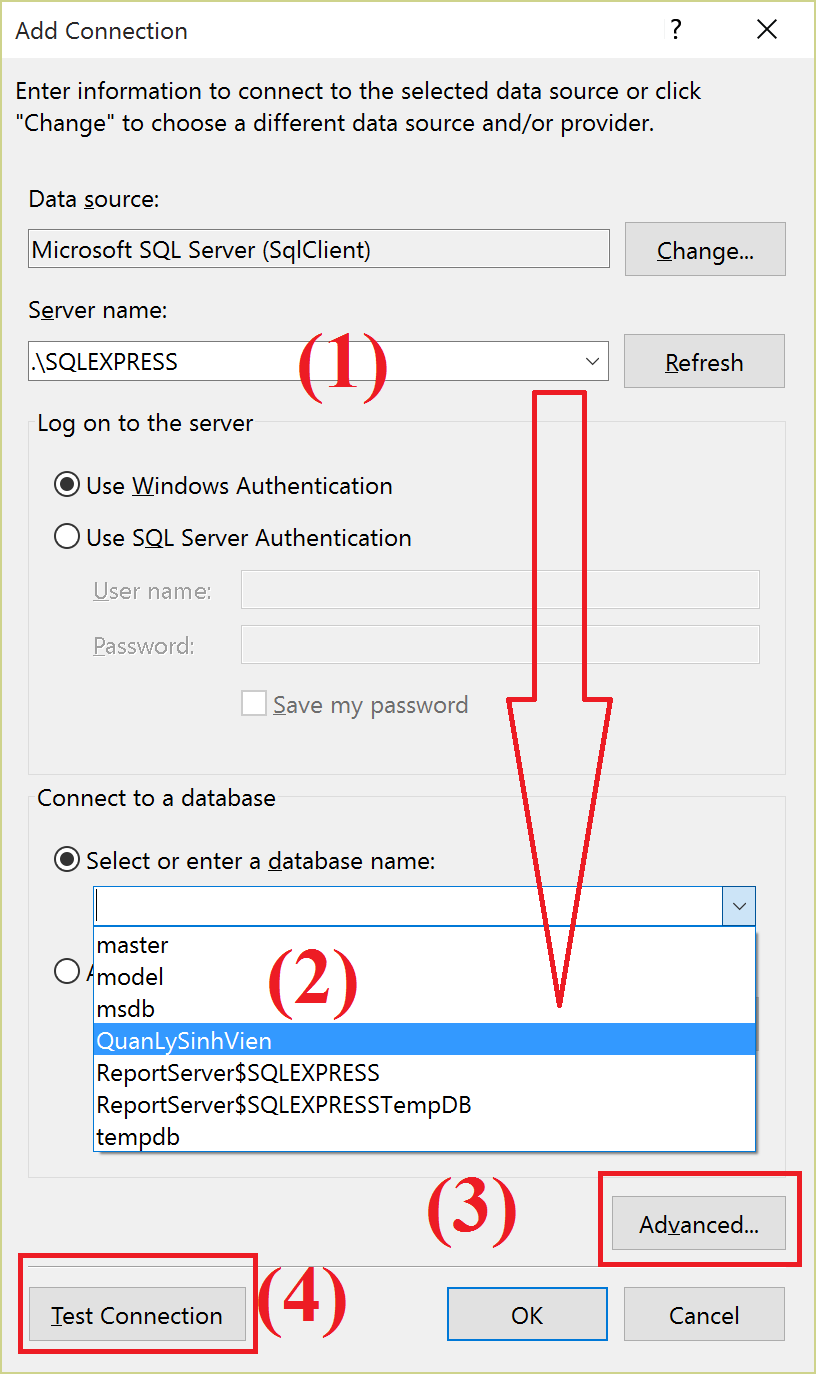
***Xem Project\_1***



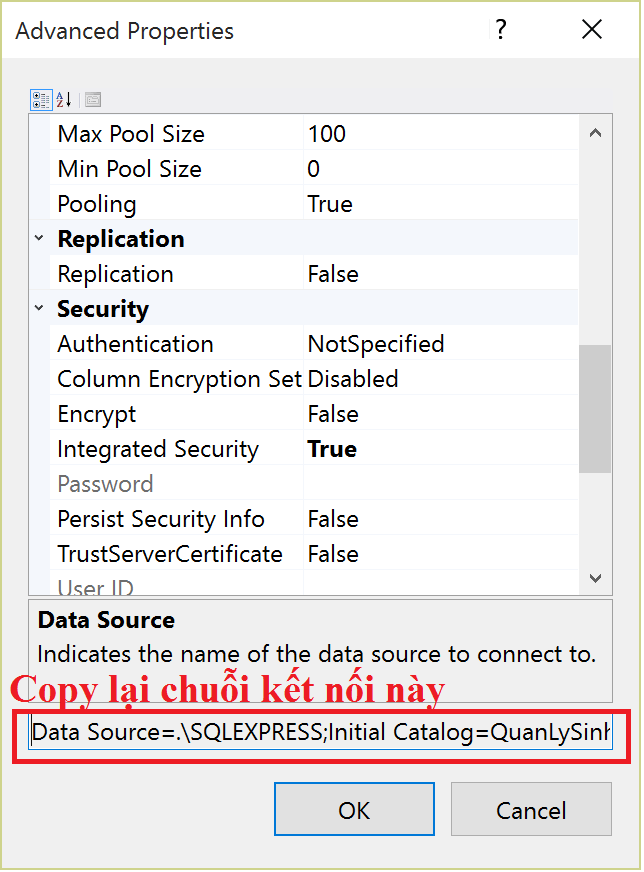
**CÁCH THỨC LẤY CHUỖI KẾT NỐI TRONG VISUAL STUDIO**

****

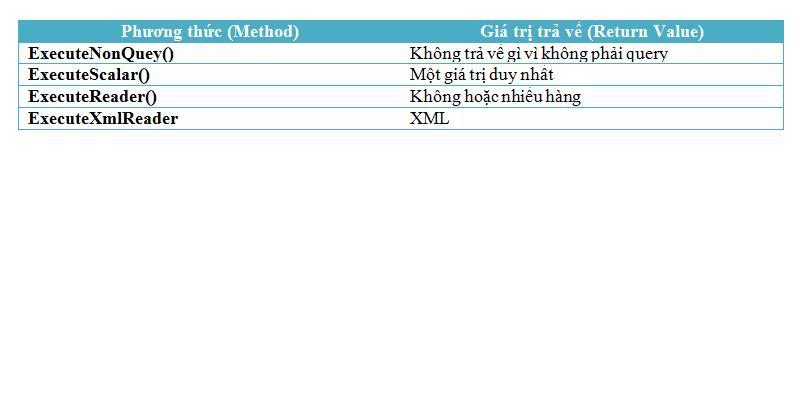
****

****

**Ở Bước (3) chúng ta sẽ hiện ra bảng như sau**

****

**CÁC LOẠI CÂU LỆNH TRONG SQL SERVER**

****

**ExecuteScalar(): Trả về một giá trị duy nhất**

* **Trong trường hợp Project\_1 đã làm ví dụ về ExecuteScalar**

**ExecuteNonQuery():**

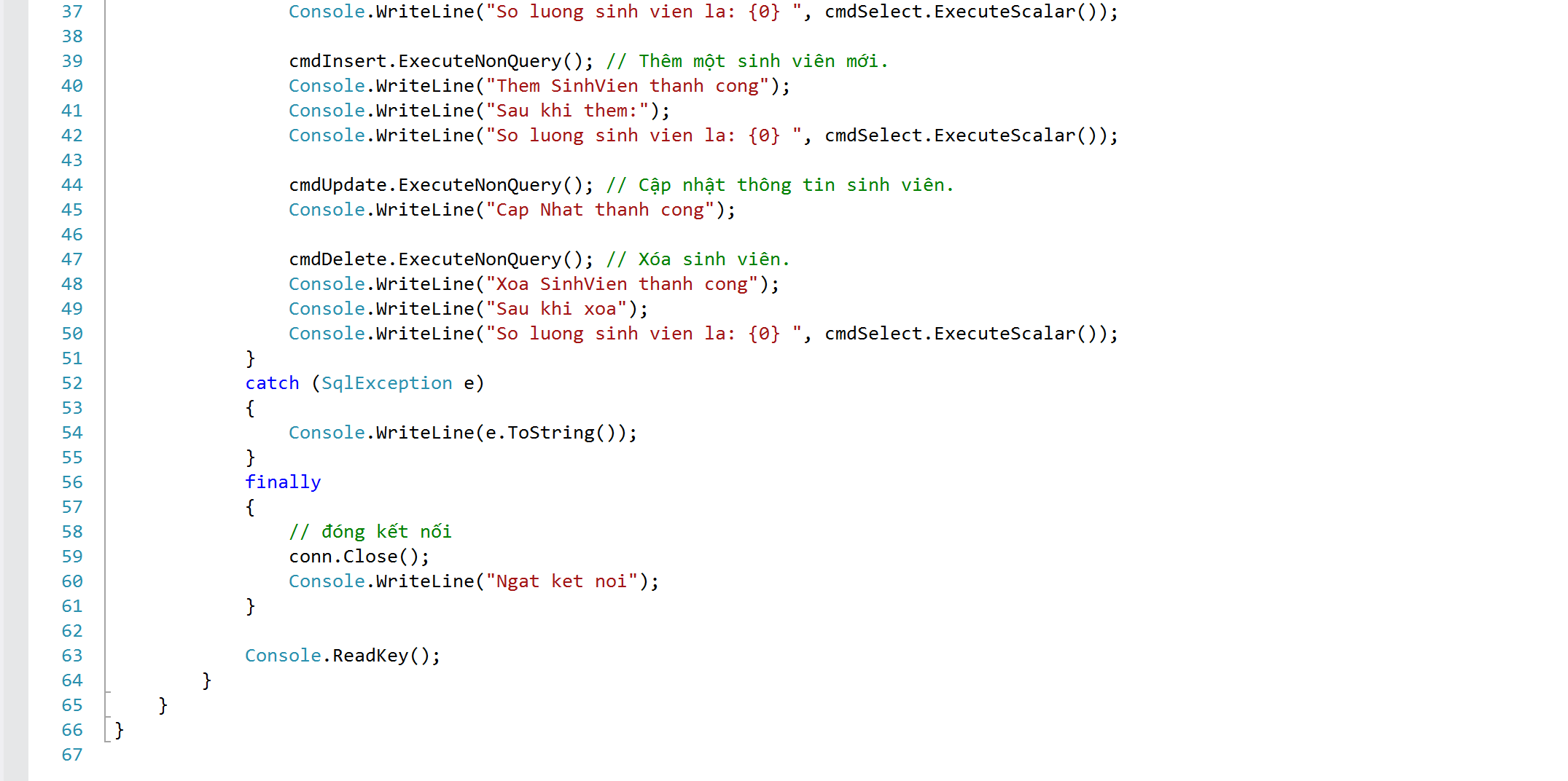
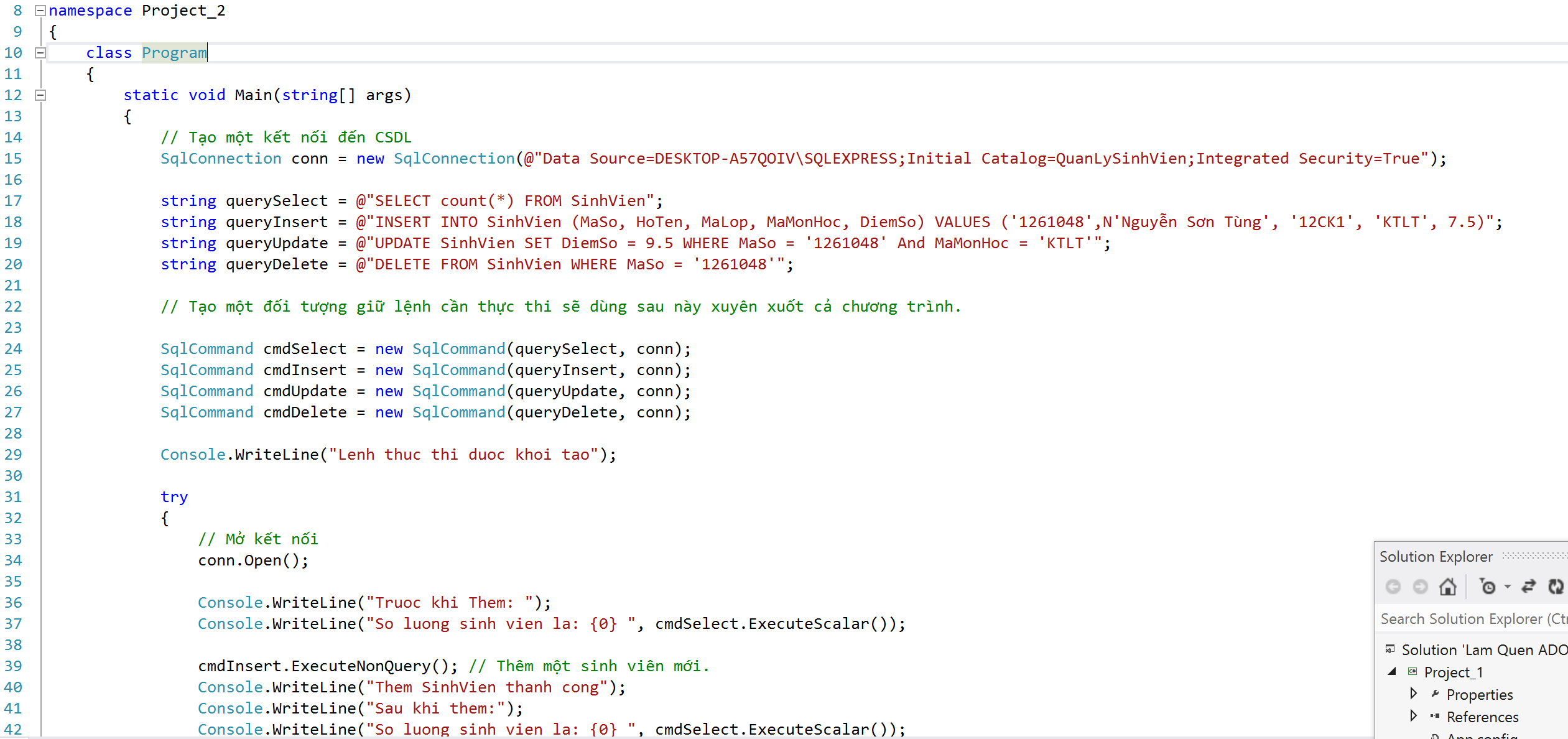
**Dùng trong những tình huống chỉ thực thi chứ không trả về gì cả**

**+ Insert**

**+ Update**

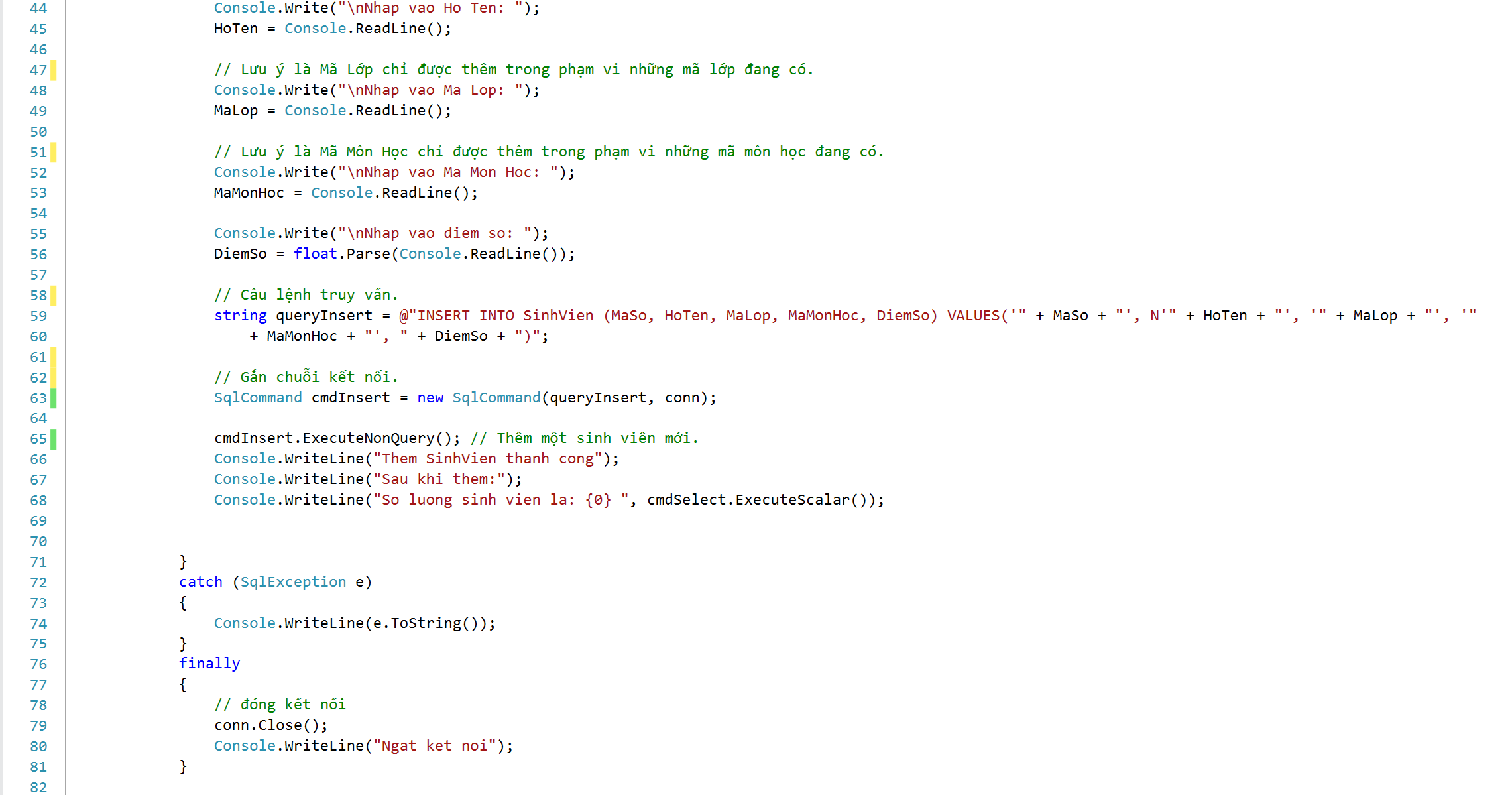
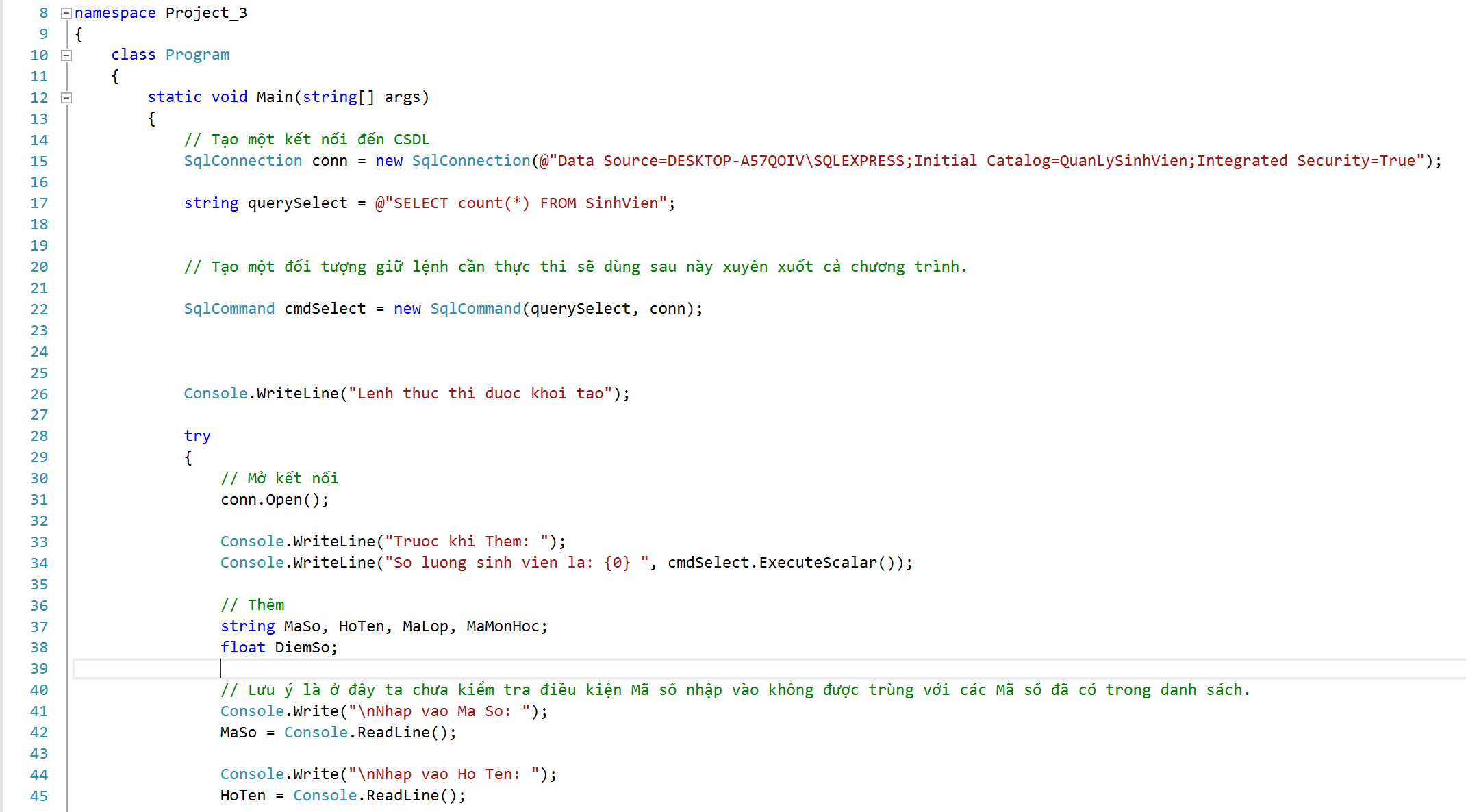
**+ Delete**

**Xem Project\_2**

****

**Nhưng trường hợp ở Project\_2 ta làm là phải truyền cố định vào những giá trị, vậy nếu như người dùng có nhu cầu tự nhập dữ liệu cần them thì làm sao ?**

**Tức là lúc này sẽ không phải truyền cố định vào mà sẽ là tùy ý người dùng nhập (Xem kỹ vì bên C# Form cũng làm giống như vậy nhưng thay vì từ bàn phím thì C# Form sẽ nhập vào các ô TextBox …)**

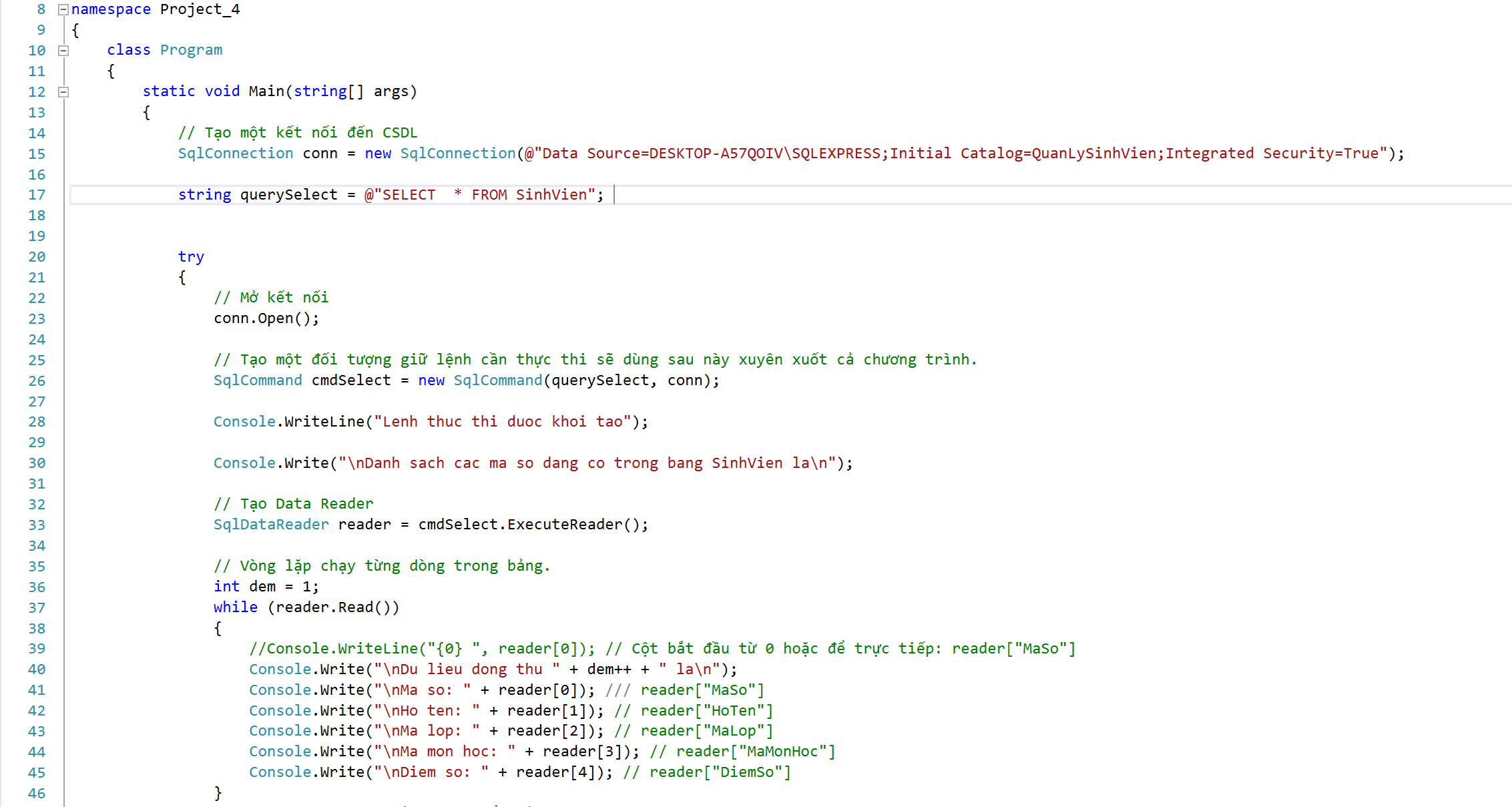
****

**ExecuteReader():**

**Chạy vào trong một bảng và xét từng dòng trong bảng đó, có thể lấy ra giá trị của từng cột để xem**

**Vd: In ra các Mã Số sinh viên hiện đang có trong bảng.**

**Xem Project\_4**

****

**Kết quả:**

****

**DataReader sẽ có ý nghĩa rất lớn khi thực hiện truy vấn với số lượng kết quả nhiều (triệu dòng) thì tốc độ truy vấn sẽ nhanh và tối ưu hơn, đặc biệt là tiết kiệm bộ nhớ hơn.**

**Lưu ý: Chỉ để xem chứ không thể thao tác trên dữ liệu lấy ra được.**

**Trường hợp nâng cao của DataReader**

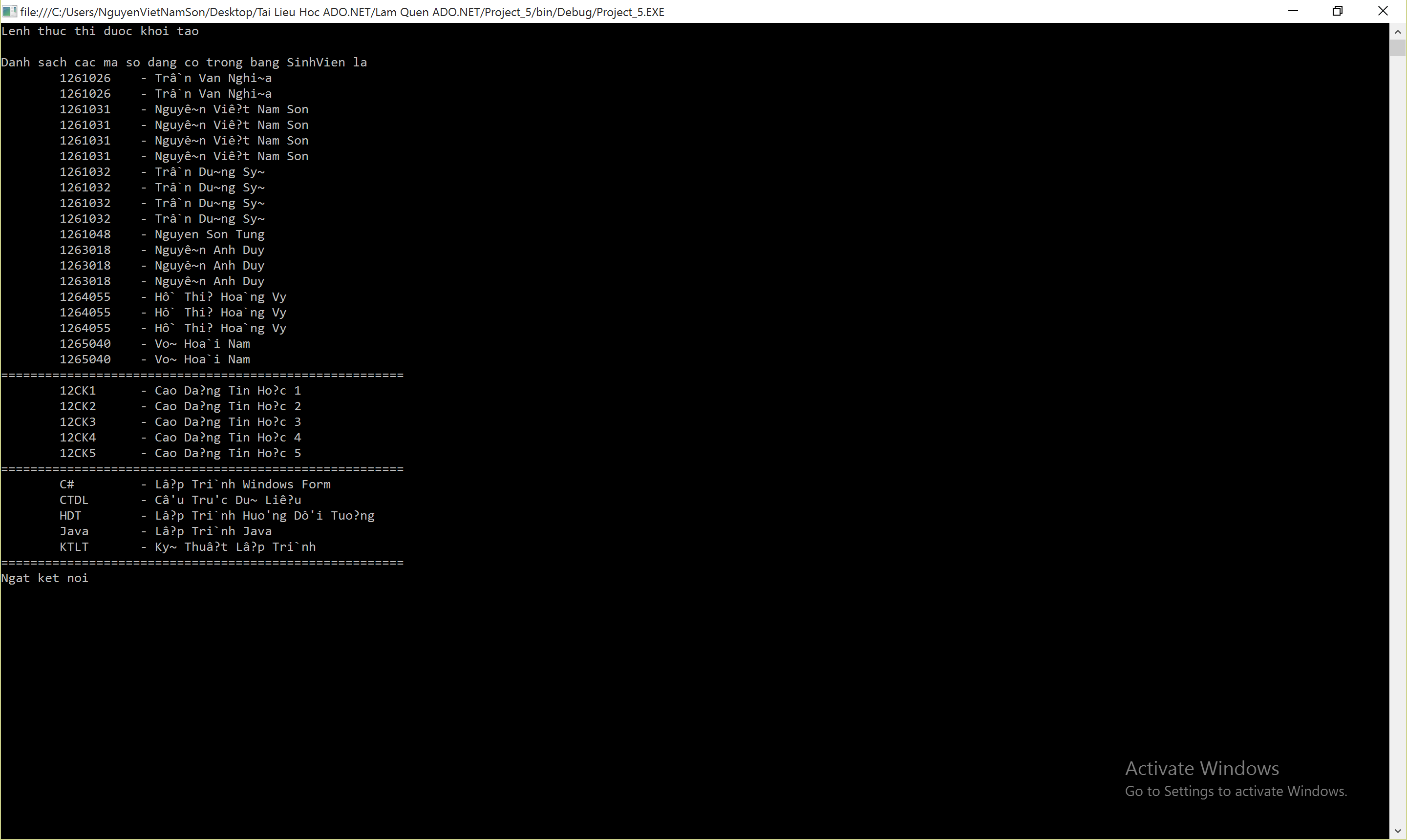
Xử lý nhiều result sets với một data reader  
Ví dụ gọi một lúc nhiều query và một data reader thì việc xử lý dữ liệu thu được dường như phức tạp nhưng .NET có cung cấp một phương thức để làm việc với thao tác này ‘NextResult()’  
Ví dụ dưới đây minh họa thao tác xử lý nhiều result sets với một reader

**Xem Project\_5**

Ở đây có 3 query và từng query được xử lý lần lượt. Chú ý các query phải có một kí tự ngăn cách để kiểm tra hợp lệ, trong SQL Server thì đơn giản chỉ cần một dấu cách (space) là được.

****

**Kết quả**

****

**BÀI TẬP TỰ LÀM**