

TRƯỜNG ĐẠI HỌC TIỀN GIANG
KHOA CÔNG NGHỆ THÔNG TIN

LÝ THIÊN TRANG
NGUYỄN THỊ PHƯƠNG LINH

LẬP TRÌNH CƠ SỞ DỮ LIỆU
TÀI LIỆU GIẢNG DẠY DÙNG CHO SINH VIÊN ĐẠI HỌC
NGÀNH CÔNG NGHỆ THÔNG TIN

LUU HÀNH NỘI BỘ
NĂM 2017

TRƯỜNG ĐẠI HỌC TIỀN GIANG
KHOA CÔNG NGHỆ THÔNG TIN

LÝ THIÊN TRANG
NGUYỄN THỊ PHƯƠNG LINH

LẬP TRÌNH CƠ SỞ DỮ LIỆU
TÀI LIỆU GIẢNG DẠY DÙNG CHO SINH VIÊN ĐẠI HỌC
NGÀNH CÔNG NGHỆ THÔNG TIN
Số tín chỉ: 3

LƯU HÀNH NỘI BỘ
NĂM 2017

LỜI NÓI ĐẦU

LẬP TRÌNH CƠ SỞ DỮ LIỆU là một trong những học phần quan trọng đối với sinh viên ngành công nghệ thông tin, được giảng dạy sau học phần Lập trình Windows. Tài liệu giảng dạy này được biên soạn nhằm phục vụ nhu cầu học lập trình của các sinh viên có thể nắm bắt những kiến thức cơ bản và thiết yếu nhất về lập trình cơ sở dữ liệu. Tài liệu được biên soạn dựa trên nhiều nguồn tài liệu khác nhau bao gồm cả tài liệu tiếng Anh. Để thuận lợi cho sinh viên theo dõi, qui ước như sau: những thuật ngữ chỉ đối tượng, thành phần thì các từ tiếng Anh được tách rời; các thuật ngữ chỉ thuộc tính, phương thức, điều khiển thì các từ tiếng Anh viết không khoảng trắng. Nội dung biên soạn được chọn lọc, sắp xếp và trình bày từ kinh nghiệm giảng dạy của tác giả giúp người đọc:

- Nắm vững kiến thức và kỹ năng lập trình, phương pháp kết nối cơ sở dữ liệu bằng ngôn ngữ lập trình cấp cao (Visual Basic.Net, C#) kết hợp với hệ quản trị CSDL (SQL Server, Access), thiết kế chương trình theo mô hình nhiều lớp, sử dụng các đối tượng trong .Net kết hợp XML, LINQ, Crystal Report,...Cách bảo mật và đóng gói ứng dụng.

- Trong quá trình học tập, sinh viên được giới thiệu cả 2 ngôn ngữ VB.Net, C# trong chương 1, 2. Từ chương 3 trở về sau, trong mỗi ví dụ chỉ minh họa bằng 1 ngôn ngữ cụ thể (hoặc VB.NET hoặc C#). Cuối mỗi chương, có bài tập tổng hợp, giúp sinh viên có thể vận dụng các kiến thức đã học.

- Sinh viên có khả năng thiết kế và cài đặt chương trình ứng dụng quản lý, bảo trì, vận hành hiệu quả các chương trình quản lý.

Tác giả hy vọng tài liệu này sẽ hỗ trợ cho các bạn sinh viên trong quá trình tự học tập, nghiên cứu. Tác giả rất mong nhận được nhiều ý kiến đóng góp của các đồng nghiệp và các bạn sinh viên.

Tác giả
Lý Thiên Trang
Nguyễn Thị Phương Linh

MỤC LỤC

Chương 1. GIỚI THIỆU ADO.NET	1
1. Giới thiệu.....	1
2. Các đặc điểm của ADO.NET	2
3. Đối tượng dữ liệu (Content Component)	4
4. Đối tượng kết nối dữ liệu (Managed Provider Component)	6
CÂU HỎI ÔN TẬP	8
Chương 2.KỸ THUẬT KẾT NỐI CƠ SỞ DỮ LIỆU.....	9
1. Tổng quan.....	9
2. Các đối tượng cốt lõi.....	10
2.1. Đối tượng Connection	10
2.2. Đối tượng Command.....	12
2.3. Đối tượng DataReader	16
2.4. Đối tượng Dataset	18
2.5. Đối tượng DataAdapter.....	29
BÀI TẬP	34
Chương 3. XÂY DỰNG ỨNG DỤNG.....	36
1. Hiện thị và thao tác dữ liệu	36
1.1. Các điều khiển hiển thị dữ liệu	36
1.1.1. ComboBox, ListBox, CheckListBox	36
1.1.2. ListView	37
1.1.3. TreeView.....	38
1.1.4. DataGridView	39
1.2. Liên kết dữ liệu qua DataBindings	42
2. Xây dựng ứng dụng.....	46
2.1. Xây dựng ứng dụng theo mô hình 3 lớp	46
2.2. Xây dựng mô hình trên Solutions	47
2.3. Ứng dụng minh họa trên Solution.....	48
2.3.1. Tổ chức cây thư mục.....	48
2.3.2. Triển khai ứng dụng minh họa.....	49
BÀI TẬP	55
Chương 4. LINQ	65
1. Tổng quan về Linq	65
2. Sự biến đổi dữ liệu trong truy vấn Linq	69
3. Truy vấn Linq theo biểu thức.....	69
4. Truy vấn Linq theo phương thức	75
4.1. Phương thức Lambda	75
4.2. Phương thức mở rộng (Extension Methods).....	76
5. Linq to SQL.....	80
5.1. Tổng quan Linq to SQL	80
5.2. Các khái niệm cơ bản.....	81
5.3. Tạo Object model sử dụng công cụ Object Relational Designer.....	82
5.4. Các thao tác với Linq to SQL	86

5.4.1. Đọc dữ liệu.....	86
5.4.2. Cập nhật dữ liệu	89
BÀI TẬP	93
Chương 5. XML	96
1. Tổng quan về XML	96
2. Cấu trúc tài liệu XML	97
3. Truy xuất tài liệu XML	99
3.1. Giới thiệu.....	99
3.2. Các đối tượng của DOM	100
3.3. Các thao tác cơ bản trên DOM.....	101
4. DataSet và XML.....	103
4.1. Đọc tài liệu XML	103
4.2. Ghi tài liệu XML từ SQL Server	103
5. Sử dụng XML trong ứng dụng Windows	104
5.1. Giới thiệu.....	104
5.2. Sử dụng File cấu hình app.config	104
5.3. Sử dụng XML để lưu dữ liệu của ứng dụng	106
5.3.1. Đọc dữ liệu.....	107
5.3.2. Thêm dữ liệu	109
5.3.3. Xóa dữ liệu.....	109
5.3.4. Cập nhật dữ liệu	110
BÀI TẬP	111
Chương 6. CRYSTAL REPORT	114
1. Giới thiệu.....	114
2. Xây dựng các chức năng của Report với Crystal Report.....	114
2.1. Các thành phần trong report.....	115
2.2. Thiết lập nguồn dữ liệu cho các report	115
2.2.1. Xây dựng chức năng report.....	124
2.2.2. Thiết kế report.....	126
2.2.3. Hiển thị dữ liệu bằng Crystal Report Viewer	129
2.3. Xây dựng chức năng report có phân nhóm.....	129
2.3.1. Tạo tập tin report và chọn chức năng cho report	129
2.3.2. Thiết kế report.....	129
2.3.3. Hiển thị dữ liệu	134
2.4. Sử dụng tham số, biểu đồ, Stored Procedure cho report.....	135
2.4.1. Xây dựng chức năng report có tham số	135
2.4.2. Xây dựng chức năng report có biểu đồ	138
2.4.3. Sử dụng report với Store Procedure.....	140
BÀI TẬP	141
Chương 7. BẢO MẬT VÀ ĐÓNG GÓI ỨNG DỤNG	145
1. Bảo mật thông tin	145
2. Tổng quan về đóng gói.....	148
3. Thao tác đóng gói.....	149
3.1. Sử dụng chức năng đóng gói trong VS.NET	149

3.2. Sử dụng phần mềm Advanced Installer	152
BÀI TẬP	159
TÀI LIỆU THAM KHẢO.....	161

DANH MỤC HÌNH

Hình 1.1. Kiến trúc ADO.NET	2
Hình 1.2. Content Component.	4
Hình 1.3. Đối tượng kết nối dữ liệu	6
Hình 2.1. Minh họa kết nối CSDL	11
Hình 2.2. Minh họa thao tác với đối tượng Command.	14
Hình 2.3. Minh họa các thao tác với CSDL	14
Hình 2.4. Minh họa sử dụng DataReader.	17
Hình 2.5. Các đối tượng trong Dataset	19
Hình 2.6. Minh họa các thao tác dữ liệu trong DataSet	25
Hình 2.7. Minh họa tạo mối quan hệ giữa các bảng.	29
Hình 2.8. Minh họa đọc dữ liệu bằng DataAdapter	32
Hình 3.1. Minh họa hiển thị dữ liệu bằng TreeView và DataGridView.	41
Hình 3.2. Minh họa sử dụng DataBinding để di chuyển các mẫu tin	45
Hình 3.3. Dữ liệu ban đầu của bảng LOP.	45
Hình 3.4. Việc trao đổi, liên lạc giữa các lớp.	47
Hình 3.5. Sự phụ thuộc của các tầng	48
Hình 3.6. Minh họa tổ chức cây thư mục	49
Hình 3.7. Minh họa giao diện Form chính	53
Hình 3.8. Hiển thị thông tin các phòng ban.	53
Hình 4.1. Minh họa kết quả thực thi câu lệnh Linq.	66
Hình 4.2. Kiến trúc và các thành phần của Linq	66
Hình 4.3. Trình tự xử lý câu lệnh Linq.	68
Hình 4.4. Minh họa kết quả thực thi câu lệnh Linq.	72
Hình 4.5. Minh họa kết quả thực thi câu lệnh Linq.	73
Hình 4.6. Minh họa kết quả thực thi câu lệnh Linq.	74
Hình 4.7. Minh họa kết quả thực thi câu lệnh Linq.	74
Hình 4.8. Minh họa kết quả thực thi câu lệnh Linq.	75
Hình 4.9. Minh họa kết quả thực thi câu lệnh Linq.	78
Hình 4.10. Minh họa kết quả thực thi câu lệnh Linq.	78
Hình 4.11. Minh họa kết quả thực thi câu lệnh Linq.	79
Hình 4.12. Minh họa kết quả thực thi câu lệnh Linq.	79
Hình 4.13. Kiến trúc thực thi của Linq to SQL.	81
Hình 4.14. Minh họa thao tác tạo Object model	83
Hình 4.15. Minh họa chọn dữ liệu nguồn để kết nối.	84
Hình 4.16. Mô hình dữ liệu đối tượng được ánh xạ từ CSDL	85
Hình 4.17. Cửa sổ Class View.	85
Hình 4.18. Minh họa kết quả thực thi câu truy vấn Linq to SQL	86
Hình 4.19. Minh họa kết quả thực thi câu truy vấn Linq to SQL	87
Hình 4.20. Minh họa kết quả thực thi câu truy vấn Linq to SQL	87
Hình 4.21. Minh họa kết quả thực thi câu truy vấn Linq to SQL	88
Hình 4.22. Minh họa kết quả thực thi câu truy vấn Linq to SQL	88
Hình 4.23. Minh họa kết quả thực thi câu truy vấn Linq to SQL	90

Hình 4.24. Minh họa kết quả thực thi câu truy vấn Linq to SQL.....	91
Hình 4.25. Dữ liệu bảng LOP.....	91
Hình 4.26. Dữ liệu bảng LOP sau khi thực hiện truy vấn cập nhập.....	91
Hình 4.27. Dữ liệu bảng KHOA.....	92
Hình 4.28. Dữ liệu bảng KHOA sau khi thực hiện thao tác xóa.....	92
Hình 4.29. Dữ liệu bảng KHOA.....	92
Hình 4.30. Dữ liệu bảng KHOA sau khi thực hiện thao tác xóa.....	92
Hình 5.1. Minh họa dữ liệu File XML.....	99
Hình 5.2. Các đối tượng của DOM.....	100
Hình 5.3. Minh họa đọc dữ liệu từ File XML.....	103
Hình 5.4. Minh họa thao tác với dữ liệu XML.....	108
Hình 6.1. Minh họa thiết lập dữ liệu nguồn cho report.....	115
Hình 6.2. Minh họa thiết lập dữ liệu nguồn cho report.....	116
Hình 6.3. Minh họa thiết lập dữ liệu nguồn cho report.....	117
Hình 6.4. Minh họa thiết lập dữ liệu nguồn cho report.....	118
Hình 6.5. Minh họa thiết lập dữ liệu nguồn cho report.....	118
Hình 6.6. Minh họa thiết lập dữ liệu nguồn cho report.....	119
Hình 6.7. Minh họa thiết lập dữ liệu nguồn cho report.....	120
Hình 6.8. Minh họa thiết lập dữ liệu nguồn cho report.....	121
Hình 6.9. Minh họa thiết lập dữ liệu nguồn cho report.....	122
Hình 6.10. Minh họa thiết lập dữ liệu nguồn cho report.....	123
Hình 6.11. Minh họa thiết lập dữ liệu nguồn cho report.....	124
Hình 6.12. Minh họa xây dựng chức năng report.....	125
Hình 6.13. Minh họa xây dựng chức năng report.....	126
Hình 6.14. Minh họa xây dựng chức năng report.....	126
Hình 6.15. Minh họa thao tác thiết kế report.....	127
Hình 6.16. Minh họa thao tác thiết kế report.....	127
Hình 6.17. Minh họa thao tác thiết kế report.....	129
Hình 6.18. Minh họa thao tác thiết kế report.....	129
Hình 6.19. Chọn dữ liệu nguồn cho report.....	130
Hình 6.20. Tạo liên kết cho dữ liệu nguồn.....	131
Hình 6.21. Thêm nhóm vào report.....	131
Hình 6.22. Minh họa chọn trường phân nhóm.....	132
Hình 6.23. Minh họa tạo số thứ tự mỗi nhóm.....	133
Hình 6.24. Minh họa thêm các trường vào report.....	133
Hình 6.25. Minh họa kết quả sau khi chương trình thực thi.....	135
Hình 6.26. Minh họa chọn dữ liệu nguồn cho report có tham số.....	136
Hình 6.27. Minh họa chọn dữ liệu nguồn cho report có tham số.....	136
Hình 6.28. Minh họa kết quả sau khi thực thi Report có tham số.....	138
Hình 6.29. Minh họa thao tác report có biểu đồ.....	139
Hình 6.30. Minh họa thao tác report có biểu đồ.....	140
Hình 7.1. Minh họa bảo mật thông tin.....	146
Hình 7.2. Minh họa đóng gói.....	149
Hình 7.3. Minh họa thao tác đóng gói phần mềm.....	149

Hình 7.4. Minh họa thao tác đóng gói phần mềm.....	150
Hình 7.5. Minh họa thao tác đóng gói phần mềm.....	150
Hình 7.6. Minh họa thao tác đóng gói phần mềm.....	151
Hình 7.7. Minh họa thao tác đóng gói phần mềm.....	151
Hình 7.8. Minh họa thao tác đóng gói phần mềm.....	152
Hình 7.9. Minh họa thao tác đóng gói phần mềm.....	153
Hình 7.10. Minh họa thao tác đóng gói phần mềm.	153
Hình 7.11. Minh họa thao tác đóng gói phần mềm.	154
Hình 7.12. Minh họa thao tác đóng gói phần mềm.	155
Hình 7.13. Minh họa thao tác đóng gói phần mềm.	155
Hình 7.14. Minh họa thao tác đóng gói phần mềm.	155
Hình 7.15. Minh họa thao tác đóng gói phần mềm.	156
Hình 7.16. Minh họa thao tác đóng gói phần mềm.	156
Hình 7.17. Minh họa thao tác đóng gói phần mềm.	157
Hình 7.18. Minh họa thao tác đóng gói phần mềm.	157
Hình 7.19. Minh họa thao tác đóng gói phần mềm.	158

DANH MỤC BẢNG

Bảng 1.1. Các giá trị của ForeignKeyConstraint	6
Bảng 3.1. Các thuộc tính sau cần quan tâm khi ComboBox, ListBox, CheckListBox kết nối với dữ liệu.	36
Bảng 3.2. Các thuộc tính chính của ListView.....	37
Bảng 3.3. Mô tả ý nghĩa các thuộc tính thường sử dụng của TreeView.....	38
Bảng 3.4. Các thuộc tính chính trên DataGrid.	39
Bảng 3.5. Các sự kiện chính của DataGrid.	40
Bảng 3.6. Các thuộc tính chính của Binding.....	43
Bảng 3.7. Các sự kiện chính của Binding	43
Bảng 3.8. Thuộc tính dùng liên kết dữ liệu của các điều khiển	44
Bảng 3.9. Các thuộc tính của DataBinding	44
Bảng 3.10. Bảng mô tả chức năng các tầng.	47
Bảng 4.1. Bảng mô tả các toán tử.....	70
Bảng 4.2. Ánh xạ từ CSDL sang LINQ Object.....	82
Bảng 5.1. Các phương thức thường dùng của XML Node.	100
Bảng 5.2. Các phương thức thường dùng của XmlDocument.	101
Bảng 5.3. Các phương thức thường dùng của XmlElement.	101
Bảng 6.1. Giải thích các lệnh truy vấn.	134

Chương 1. GIỚI THIỆU ADO.NET

Mục tiêu:

Sau khi học xong chương này, sinh viên có thể:

1. Kiến thức

- + Nhận biết các đặc điểm và kiến trúc của ADO.NET.
- + Trình bày các khái niệm cơ bản về các đối tượng: DataSet, DataTable, DataRelation, Connection, Command, DataAdapter.

2. Kỹ năng

- + Vận dụng kiến trúc ADO.NET và thực hiện được các kỹ thuật cơ bản của lập trình ADO.NET.

3. Thái độ

- + Tự tin trình bày các đặc điểm của ADO.NET, Content Component, Managed Provider Component.

1. Giới thiệu

Hầu hết các ứng dụng hiện nay đều có liên quan đến dữ liệu. Dữ liệu thường được tổ chức dưới dạng cơ sở dữ liệu (CSDL) quan hệ và lưu trữ vật lý trên hệ quản trị CSDL như Access, SQL Server, Oracle, DB2, ...

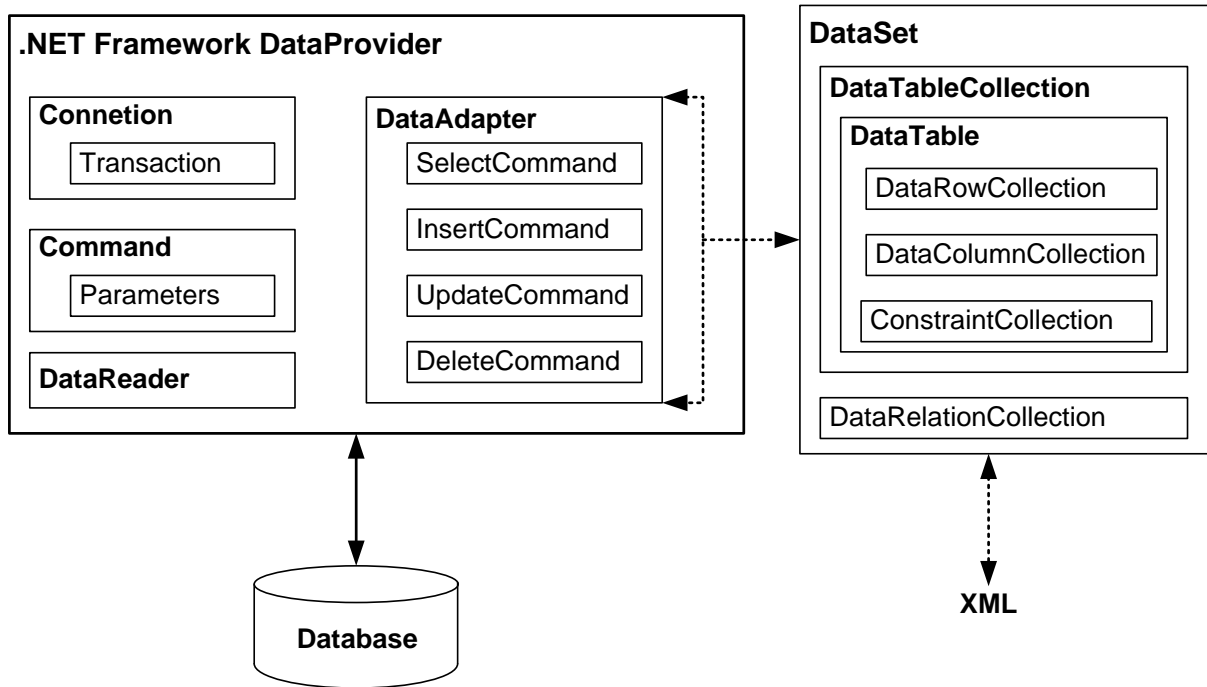
Các đặc điểm dữ liệu cần được quan tâm khi xây dựng ứng dụng là:

- Dữ liệu lưu trữ dưới dạng tập trung hay phân tán.
- Đảm bảo tính nhất quán và toàn vẹn dữ liệu.
- Đảm bảo khả năng truy xuất đồng thời của nhiều người trên dữ liệu.
- Đảm bảo thời gian xử lý nhanh.
- Bảo mật dữ liệu.
- Trao đổi dữ liệu giữa các hệ thống khác nhau.

Tất cả những vấn đề về dữ liệu được giải quyết nhờ vào đặc trưng của hệ quản trị CSDL. ADO (Active Data Object) là mô hình đối tượng CSDL mới nhất do Microsoft tạo ra cho phép người lập trình truy cập dữ liệu từ bất kỳ hệ quản trị CSDL nào.

ADO.NET cung cấp khả năng truy xuất và xử lý dữ liệu lớn, đồng thời trên hệ thống ứng dụng phân tán hoặc hệ thống tập trung nhiều người dùng. ADO.NET có thể làm việc với dữ liệu không kết nối bằng cách lưu dữ liệu trong bộ nhớ như một CSDL thu nhỏ gọi là DataSet, nhằm tăng tốc độ tính toán, xử lý tối đa và hạn chế việc sử dụng tài nguyên trên cơ sở dữ liệu chính. Thêm nữa là khả năng xử lý dữ liệu dạng XML, dữ liệu XML có thể trao đổi giữa bất kỳ hệ thống nào nên có nhiều khả năng làm việc với nhiều ứng dụng khác nhau.

Kiến trúc ADO.NET:



Hình 1.1. Kiến trúc ADO.NET

Kiến trúc ADO.NET bao gồm hai phần chính:

- Managed Provider Component: bao gồm các đối tượng như DataAdapter, DataReader, ... giữ nhiệm vụ làm việc trực tiếp với dữ liệu như: Database, file,...
- Content Component: bao gồm các đối tượng như DataSet, DataTable, ...
 - + DataSet: giúp truy cập dữ liệu nhanh chóng, DataSet được coi là một bản sao gọn nhẹ của CSDL trong bộ nhớ với nhiều bảng và các mối quan hệ.
 - + DataAdapter: là đối tượng kết nối giữa DataSet và CSDL, bao gồm hai đối tượng Connection và Command để truyền dữ liệu cho DataSet cũng như cập nhật dữ liệu từ DataSet đến CSDL chính.
 - + DataTable: được hình thành từ các cột (DataColumn) và các dòng (DataRow).
 - + DataReader: phục vụ việc truy cập vào CSDL nhanh và hiệu quả, cụ thể là việc đọc dữ liệu từ Database.

2. Các đặc điểm của ADO.NET

ADO.NET có một số đặc điểm nổi bật: Interoperability (khả năng tương tác), Scalability (hỗ trợ nhiều người dùng), Productivity (mở rộng khả năng làm việc), Performance (tính hiệu quả).

– Tương tác giữa nhiều hệ thống khác nhau (Interoperability): ADO.NET thay đổi bản chất của việc đóng gói dữ liệu trước khi truyền trên mạng:

+ Với ADO, từng mẫu tin được đóng gói ở dạng Network Data Representation (NDR) trước khi truyền trên mạng.

+ ADO.NET thay thế NDR bằng định dạng XML. XML là định dạng chuẩn để trao đổi dữ liệu đang được hỗ trợ rộng rãi, với bản chất Text, XML có thể sử dụng HTTP để trao đổi dữ liệu.

– Hỗ trợ nhiều người dùng (Scalability): ADO.NET sử dụng dữ liệu ở dạng không kết nối theo quy trình:

+ Tạo kết nối giữa Client với Server để lấy dữ liệu.

+ Server gửi dữ liệu về cho Client.

+ Ngắt kết nối Client với Server.

+ Khi cần cập nhật dữ liệu, kết nối giữa Client và Server được phục hồi.

Với cơ chế không kết nối, thời gian kết nối giữa Client và Server không còn lâu dài như trước, Server có khả năng phục vụ nhiều Client hơn so với cơ chế ADO trước đây.

Cơ chế Disconnected Data (không kết nối) có 2 điểm cần lưu ý:

+ Dữ liệu sẽ có một độ trễ và sai lệch không đáng kể trong một hệ thống Client/Server với nhiều người dùng.

+ Với cơ chế Connection pooling, thông tin về kết nối vẫn còn được lưu giữ ở Server trong một khoảng thời gian nhất định và khi Server còn đủ tài nguyên, mỗi khi Client cần kết nối lại với Server, thông tin kết nối được lấy từ Connection pool nên tốc độ không bị ảnh hưởng.

– Mở rộng khả năng làm việc với CSDL (Productivity):

+ Điểm mạnh hơn của ADO.NET so với ADO là DataSet tích nhiều chức năng hơn.

+ Có sự tập trung về các chức năng của các đối tượng, ví dụ Connection không còn thi hành một câu lệnh SQL hay Store Procedure mà giao hoàn toàn cho Command đảm nhận.

+ ADO.NET phát triển trên .NET là thành phần có thể kế thừa được. Người lập trình có thể kế thừa các đối tượng của ADO.NET để xây dựng các đối tượng mới tốt hơn và phù hợp với nhu cầu phát triển ứng dụng.

+ Visual Studio .NET với thành phần thiết kế giúp người lập trình tạo ra các DataSet nhanh chóng với nội dung rõ ràng hơn.

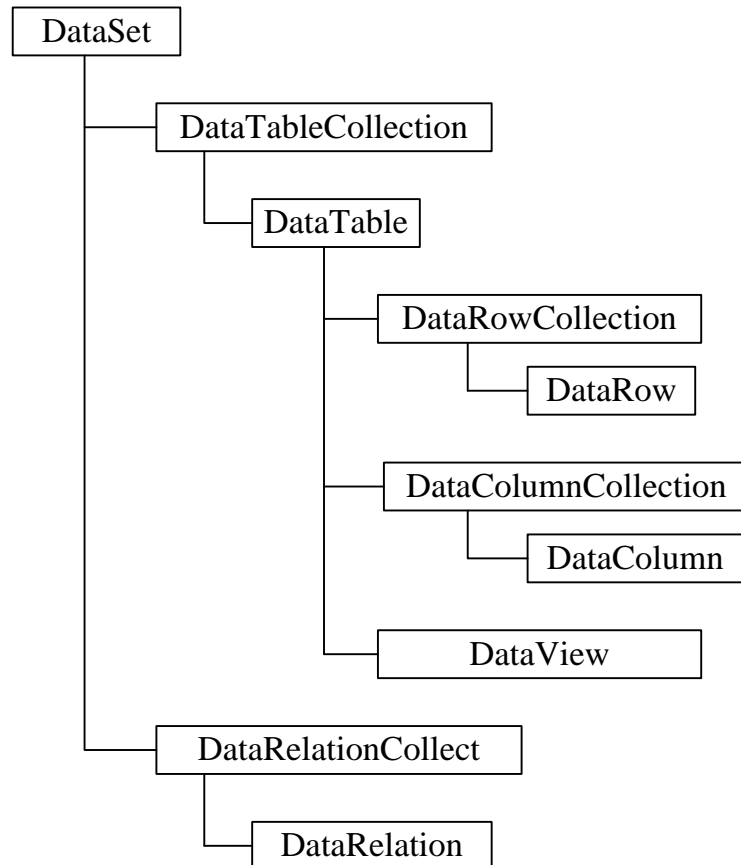
– Hiệu quả trong xử lý dữ liệu (Performance):

+ Chế độ không kết nối trực tiếp dữ liệu đảm bảo ứng dụng trên Client ít gặp lỗi khi xử lý dữ liệu, đồng thời giúp mở rộng ứng dụng tới nhiều người dùng.

+ Định dạng XML thay thế cho NDR trong trao đổi dữ liệu giúp cho dữ liệu có thể được xử lý bởi nhiều Client hơn, giúp việc truyền thông tin nhanh hơn vì không cần đóng gói như NDR.

3. Đối tượng dữ liệu (Content Component)

Content Component là các đối tượng dữ liệu dùng để kết nối dữ liệu cần xử lý. Trong ADO.NET, dữ liệu được lưu trữ bởi DataSet có nhiều bảng (Table) và các mối quan hệ (Relationship). Các đối tượng chính giúp tạo nên Content Component gồm: DataSet, DataTable, DataView, DataRow, DataColumn, DataRelation.



Hình 1.2. Content Component.

– DataSet:

- + ADO.NET chứa dữ liệu trong DataSet cũng với mối quan hệ giữa các bảng giống như hình ảnh CSDL thu nhỏ, có thể có nhiều DataTable và các DataRelation.

- + Các chức năng tìm kiếm, cập nhật dữ liệu được thực hiện qua các phương thức: HasChanges(), HasError(), GetChanges(), AcceptChanges(), RejectChanges().

- HasChanges(): kiểm tra sự thay đổi của tất cả các dòng dữ liệu trên các bảng (thêm mới, xóa bỏ, sửa đổi), trả về True nếu có, ngược lại là False.

- HasError(): cho biết có dòng nào trên DataTable bị lỗi không (True: có dòng bị lỗi, False: không có dòng nào bị lỗi).

- `GetChanges()`: phương thức trả về một `DataTable` gồm những dòng dữ liệu đã thay đổi kể từ lần lấy dữ liệu từ nguồn về hoặc từ lần cập nhật trước vào bảng bằng phương pháp `AcceptChanges`.

- `AcceptChanges()`: phương thức cập nhật các thay đổi kể từ lần cập nhật trước. Sau khi thực hiện tất cả các dòng đều có tình trạng `Unchanged`, `Deleted` đều bị loại khỏi bảng.

- `RejectChanges()`: hủy bỏ các thay đổi từ lần cập nhật trước.

- `DataTable`: chứa dữ liệu của 1 bảng trong `DataSet` và thuộc lớp `DataTable`. `DataTable` gồm:

- + Tập hợp các cột thuộc lớp `DataColumnCollection` trong đó mỗi cột là một đối tượng thuộc lớp `DataColumn`.

- + Tập hợp các Rows thuộc lớp `DataRowCollection` trong đó mỗi dòng là một đối tượng thuộc lớp `DataRow`.

- `DataRelation`: định nghĩa mối quan hệ giữa các bảng, duyệt dữ liệu trong các bảng theo kiểu quan hệ Master – Detail. Một đối tượng `DataRelation` bao gồm các thông tin: tên của bảng cha (Master), tên của bảng con (Detail). Các Column trong `DataRelation` đại diện cho khóa chính trong bảng cha và khóa ngoại trong bảng con.

Ví dụ: yêu cầu truy cập dữ liệu kiểu Master – Detail trong bảng cha có 10 dòng, mỗi dòng trên bảng cha có quan hệ với 10 dòng trong bảng con. Trong ADO.NET cho phép truy vấn một lần dữ liệu nhưng các dữ liệu vẫn nằm trên các bảng, dễ dàng thao tác và xử lý.

- Tập hợp các quan hệ giữa các bảng trong một `DataSet` được lưu trong đối tượng `Relations` thuộc lớp `DataRelationCollections`. Ngoài ra, một bảng khi tham gia vào một quan hệ cũng lưu trữ thông tin về quan hệ đó.

- + `ChildRelations`: liệt kê tất cả các quan hệ mà bảng tham gia với vai trò là phía bảng cha.

- + `ParentRelations`: liệt kê tất cả các quan hệ mà bảng tham gia với vai trò là phía bảng con.

ADO.NET xác định hai loại ràng buộc:

- + Ràng buộc duy nhất (`Unique Constraint`): đảm bảo tính duy nhất về giá trị của một cột trong bảng.

- + Ràng buộc phụ thuộc tồn tại (`ForeignKey Constraint`): chỉ ra mối quan hệ ràng buộc giữa bảng cha và con, với các giá trị:

Bảng 1.1. Các giá trị của ForeignKeyConstraint

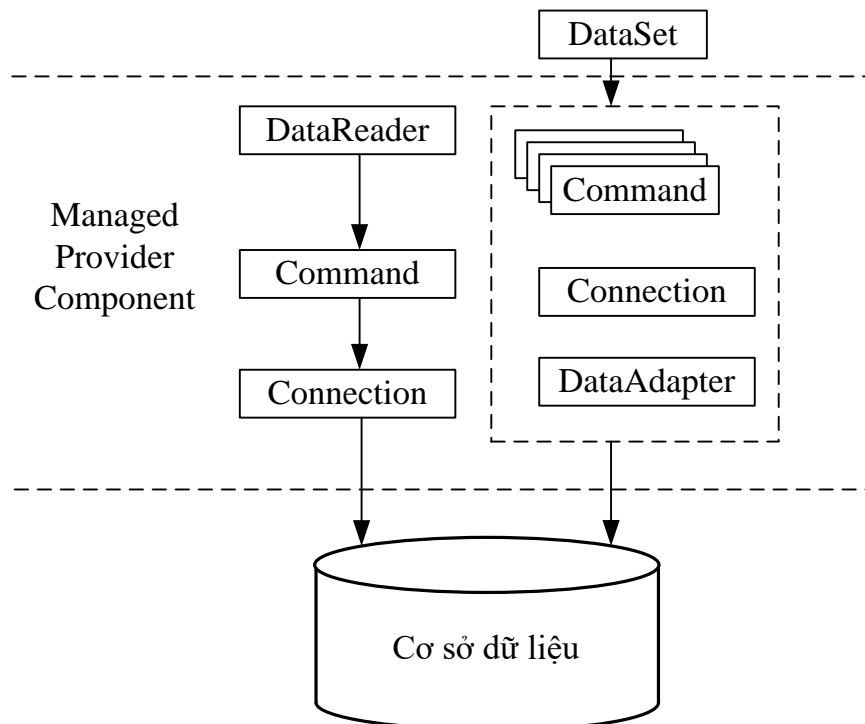
Giá trị	Diễn giải
None	Không làm gì cả.
Cascade	Phụ thuộc vào dòng trên Parent Table sẽ cập nhật.
SetDefault	Giá trị của cột khóa ngoại trên Detail Table được đặt về giá trị mặc định khi dòng trên Parent Table bị xóa.
SetNull	Giống SetDefault, giá trị được đặt là Null.

– DataView: một DataTable có thể tạo nhiều DataView với các điều kiện lọc, sắp xếp dữ liệu khác nhau. Trên DataView có thể xem hay thay đổi giá trị các mẫu tin và giữ nhiệm vụ kết nối với các Control của Window Form và Web Form.

4. Đối tượng kết nối dữ liệu (Managed Provider Component)

Microsoft cung cấp hai bộ đối tượng kết nối dữ liệu là OLEDB và SQL. OLEDB dùng để kết nối với hệ quản trị CSDL Access bao gồm các đối tượng: OleDbConnection, OleDbCommand, OleDbParameter, OleDbDataReader, OleDbDataAdapter.

Tương tự để kết nối dữ liệu với hệ quản trị CSDL SQL Server cũng có các đối tượng giống của OLEDB nhưng bắt đầu với SQL thay cho OLEDB.



Hình 1.3. Đối tượng kết nối dữ liệu

– Connection: cả OleDbConnection và SqlConnection đều có các thuộc tính và phương thức giống nhau như: ConnectionString, State hay Open.

– Command: thuộc tính Connection của SqlCommand hoặc OleDbCommand chỉ ra kết nối CSDL được sử dụng. Command cũng có thể sử dụng tham số để thi hành các câu truy vấn SQL hoặc Store Procedure. Các cách thức thực hiện lệnh:

+ ExecuteReader(): dùng để thực hiện các câu truy vấn trả về dữ liệu. Dữ liệu trả về sẽ là một đối tượng DataReader.

+ ExecuteNonQuery(): dùng để thực hiện các lệnh cập nhật dữ liệu.

+ ExecuteScalar(): dùng để thực hiện các lệnh tính toán và trả về một dòng, một cột chứa kết quả của lệnh tính toán.

– DataReader: hỗ trợ về mặt tốc độ khi truy cập dữ liệu, tuy nhiên nó có sử dụng con trỏ ở phía Server và kết nối với Sever trong suốt quá trình đọc dữ liệu, việc này làm giảm khả năng mở rộng của ứng dụng. Vì vậy, khi sử dụng một đối tượng DataReader, cần phải tiến hành việc đọc dữ liệu càng nhanh càng tốt.

– DataAdapter: DataAdapter được ADO.NET cung cấp như một công cụ giúp kết nối dữ liệu của một bảng trong DataSet với Database trên Server.

CÂU HỎI ÔN TẬP

Câu 1. Trình bày kiến trúc của ADO.NET.

Câu 2. Trình bày các thành phần và nhiệm vụ của Content Component.

Câu 3. Trình bày đặc điểm và các thành phần của Managed Provider Component.

Câu 4. Nêu đặc điểm của đối tượng DataSet, DataAdapter, DataTable, DataRelation.

Chương 2. KỸ THUẬT KẾT NỐI CƠ SỞ DỮ LIỆU

Mục tiêu:

Sau khi học xong chương này, sinh viên có thể:

1. Kiến thức

+ Nhận biết được các thuộc tính, phương thức, chức năng của các đối tượng: Connection, Command, DataAdapter, DataSet.

2. Kỹ năng

+ Thiết lập các thuộc tính và phương thức của các đối tượng: Connection, Command, DataAdapter, DataSet.

3. Thái độ

+ Tự tin viết mã lệnh sử dụng các đối tượng Connection, Command, DataAdapter, DataSet trong thiết lập kết nối và truy vấn dữ liệu cũng như tương tác dữ liệu trong CSDL.

1. Tổng quan

Thành phần dùng để kết nối và tương tác (nhận, thêm, sửa, xóa,...) với CSDL vật lý trong ADO.NET được gọi là bộ kết nối và tương tác dữ liệu (.NET Data Provider). .NET Data Provider là mã được quản lý trong .NET tương đương với OLEDB Provider hoặc ODBC Driver. .NET Data Provider bao gồm một số đối tượng được cài đặt theo chức năng yêu cầu.

Ba bộ ADO.NET Data Provider có sẵn, mỗi bộ nằm trong không gian tên riêng (namespace) là OleDb, Sql, Odbc.

– OleDb: OleDb sử dụng bộ OLEDB gốc với các dịch vụ tương tác COM của .NET để truy xuất CSDL, cho phép truy xuất bất kỳ CSDL nào có OleDb Data Provider.

– ODBC: ODBC Data Provider là bộ kết nối và tương tác dữ liệu được sử dụng khi đang truy xuất CSDL mà không có .NET Data Provider hoặc OleDb Data Provider. ODBC Data Provider có thể cung cấp hiệu suất thực thi tốt hơn OleDb Data Provider. Các đối tượng của ODBC Data Provider được chứa trong System.Data.Odbc.

– SqlClient: bộ kết nối và tương tác dữ liệu SqlClient được tối ưu hóa để làm việc với SQL Server:

+ SqlClient kết nối và tương tác trực tiếp với CSDL thông qua giao thức Tabular Data Stream (TDS).

+ Các đối tượng của SqlClient Data Provider được chứa trong không gian tên System.Data.SqlClient.

2. Các đối tượng cốt lõi

Mỗi bộ kết nối và tương tác dữ liệu bao gồm bốn đối tượng chính:

- Connection: thiết lập kết nối với nguồn dữ liệu cụ thể.
- Command: thực thi lệnh ở nguồn dữ liệu, bao gồm các đối tượng Parameter và phương thức để thực thi các kiểu lệnh khác nhau.
- DataReader: chỉ đọc và trả về dữ liệu từ nguồn dữ liệu.
- DataAdapter: cầu nối giữa DataSet và nguồn dữ liệu để nhận và lưu trữ.

Mỗi đối tượng nhận được từ lớp cơ sở chung và cài đặt các giao tiếp chung, nhưng đồng thời có thêm cài đặt của riêng nó.

- System.Data.SqlClient gồm 4 đối tượng: SqlConnection, SqlCommand, SqlDataReader, SqlDataAdapter sử dụng trong kết nối với SQL Server.
- System.Data.OleDb gồm 4 đối tượng: OleDbConnection, OleDbCommand, OleDbDataReader, OleDbDataAdapter sử dụng trong kết nối với Access.

2.1. Đối tượng Connection

Connection là đối tượng tạo kết nối giữa ứng dụng và nguồn dữ liệu.

ConnectionString: khai báo các thông tin cần thiết cho Connection như nguồn dữ liệu, nơi chứa dữ liệu, tên tập tin, người dùng, mật khẩu, ... tùy thuộc vào nguồn dữ liệu.

Với OLEDB Provider có các thành phần chính như sau:

- Provider: khai báo Data Provider.
- Data Source: nguồn dữ liệu.
- User ID: tên người dùng.
- Password: mật khẩu.

Ví dụ 1: tạo Connection đến CSDL Access:

```
Dim sqlCon As String = "Provider = Microsoft.Jet.OLEDB.4.0; Data source=dulieu.mdb"
```

Khi sử dụng SqlConnection với các thành phần chính như sau:

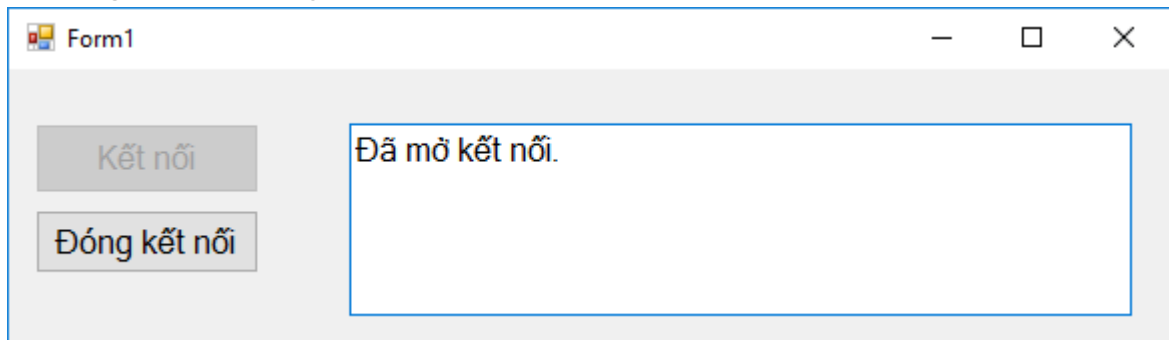
- Data Source: tên máy nơi cài đặt SQL Server có nguồn dữ liệu muốn kết nối.
- Initial Catalog: tên database muốn kết nối.
- Integrated Security: True/False chỉ định dùng cơ chế bảo mật của Windows Login (True) hay cơ chế bảo mật của SQL Server (False).
- User ID: tên người dùng (phải khai báo khi Integrated Security = False).
- Password: mật khẩu (phải khai báo khi Integrated Security = False).

Việc sử dụng cơ chế bảo mật nào tùy thuộc SQL Server quy định.

Các thuộc tính của Connection:

- Database: (chỉ đọc) tương ứng với Initial Catalog (SQL) hay tên Database muốn làm việc (OLEDB).
- DataSource: (chỉ đọc) tương ứng với nguồn dữ liệu nguồn (OLEDB và SQL).
- Provider: (chỉ đọc) tương ứng với Provider (OLEDB).
- State: tình trạng kết nối của Connection với các giá trị:
 - + Broken: kết nối với nguồn dữ liệu đã bị ngắt, tình trạng này chỉ xảy ra sau khi đã kết nối.
 - + Closed: kết nối đã đóng.
 - + Connecting: đang kết nối với nguồn dữ liệu.
 - + Executing: kết nối đang thực hiện một lệnh.
 - + Fetching: kết nối đang truy xuất dữ liệu.
 - + Open: kết nối đang mở.

Ví dụ 2: tạo một Form, với nút Kết nối (bntKetNoi) và nút Đóng kết nối (bntDongKetNoi) và Textbox (txtKetQua). Khi nhấp chuột vào nút Kết nối thì Textbox sẽ thể hiện các trạng thái kết nối với CSDL NhanVien, còn nhấp vào nút Đóng kết nối sẽ ngắt kết nối với CSDL NhanVien.



Hình 2.1. Minh họa kết nối CSDL

– VB.NET:

Tạo Connection đến CSDL Sql:

```
Imports System.Data
Imports System.Data.SqlClient
'Tạo thể hiện của đối tượng Connection
Dim cnn As SqlConnection = New SqlConnection
'Thiết lập chuỗi kết nối
cnn.ConnectionString = "Data Source=DESKTOP-4SL47N2\SQLEXPRESS;
Initial Catalog=nv; Persist Security Info=True; User ID=sa;
Password=12345678"
```

Sự kiện Click của nút Kết nối:

```
If (cnn.State = System.Data.ConnectionState.Closed) Then
    cnn.Open()
    txtKeQua.Text = "Đã mở kết nối."
    bntKetNoi.Enabled = False
    bntDongKetNoi.Enabled = True
End If
```

Sự kiện Click của nút Đóng kết nối:

```
If (cnn.State = System.Data.ConnectionState.Open) Then
    cnn.Close()
    txtKeQua.Text = "Kết nối đã đóng."
    bntKetNoi.Enabled = True
    bntDongKetNoi.Enabled = False
End If
```

– C#:

Tạo Connection đến CSDL Sql:

```
using System.Data;
using System.Data.SqlClient;
SqlConnection cnn = new SqlConnection();
cnn.ConnectionString = @"Data Source=DESKTOP-4SL47N2\SQLEXPRESS;
Initial Catalog=nv; Persist Security Info=True; User ID=sa;
Password=12345678";
```

Sự kiện Click của nút Kết nối:

```
if (cnn.State == System.Data.ConnectionState.Closed)
{
    cnn.Open();
    txtKeQua.Text = "Đã mở kết nối.";
    bntKetNoi.Enabled = False;
    bntDongKetNoi.Enabled = True;
}
```

Sự kiện Click của nút Đóng kết nối:

```
if (cnn.State == System.Data.ConnectionState.Open)
{
    cnn.Close();
    txtKeQua.Text = "Kết nối đã đóng."
    bntKetNoi.Enabled = True
    bntDongKetNoi.Enabled = False
}
```

2.2. Đối tượng Command

Sau khi tạo kết nối với nguồn dữ liệu, mọi thao tác trên nguồn dữ liệu đó đều được thực thi thông qua đối tượng Command, đối tượng Command cho phép thực thi các câu lệnh lên nguồn dữ liệu và nhận dữ liệu hoặc kết quả trả về. Tùy theo loại Connection, đối tượng Command thuộc tên miền như sau:

System.Data.OleDb.OleDbCommand.

System.Data.SqlClient.SqlCommand.

Command có các thuộc tính:

- CommandText: lệnh SQL hoặc tên Stored Procedure **muốn** thực hiện trên nguồn dữ liệu.

- CommandType: giá trị cho biết nội dung CommandText là gì với các giá trị sau:

- + Text: (mặc định) một câu lệnh SQL.

- + Storeprocedure: tên một thủ tục.

- + TableDirect: tên của bảng, khi CommandType có giá trị này, CommandText là tên của một bảng. Khi Command thực hiện sẽ trả về đủ các dòng và các cột (chỉ dùng cho OleDbCommand).

- Connection: chỉ ra kết nối được dùng để thực thi lệnh.

- CommandTimeout: thiết lập thời gian chờ một lệnh hoàn tất trước khi nó phát ra thông báo lỗi.

- Parameters: là một tập hợp các tham số được truyền đến/về từ lệnh được thực thi.

- Transaction: chỉ ra giao dịch trong đó lệnh được thực thi.

Đối tượng Command trong ba bộ kết nối Sql, OleDb và Odbc đều có các thuộc tính và phương thức giống nhau. Riêng SqlCommand có thêm phương thức ExecuteXmlReader (trả về dữ liệu dạng XML).

Ví dụ 3: thiết kế giao diện như ví dụ 1, khi nhấp vào Button Kết nối sẽ hiển thị câu truy vấn lên Textbox Kết quả.

Với CSDL NhanVien gồm 2 table NhanVien, PhongBan

NhanVien: **MaNV**, HoTen, NgaySinh, DiaChi, MaPhong.

PhongBan: **MaPhong**, TenPhong.

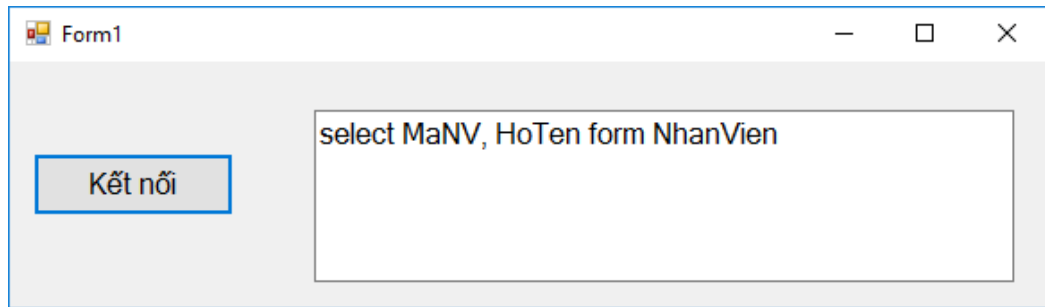
Khi thực thi chương trình, nhấp vào nút Kết nối thì câu lệnh SQL gán cho thuộc tính CommandText của đối tượng SqlCommand được hiển thị trong TextBox Kết quả.

Viết mã lệnh cho nút lệnh Kết nối:

- VB.NET:

```
Dim cnn As SqlConnection = New SqlConnection
cnn.ConnectionString = "Data Source=DESKTOP-4SL47N2\SQLEXPRESS;
Initial Catalog=nv; Persist Security Info=True; User ID=sa;
Password=12345678"
Dim cmd As SqlCommand = New SqlCommand
txtKeQua.Clear()
cmd.Connection = cnn
cmd.CommandType = CommandType.Text
```

```
cmd.CommandText = "select MaNV, HoTen form NhanVien"
txtKeQua.Text = cmd.CommandText
```



Hình 2.2. Minh họa thao tác với đối tượng Command.

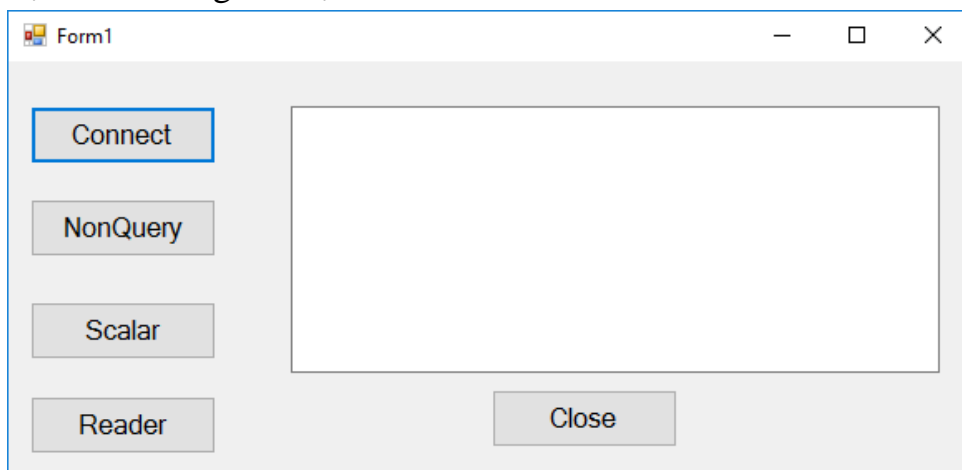
– C#:

```
SqlConnection cnn = new SqlConnection();
cnn.ConnectionString = @"Data Source=DESKTOP-
4SL47N2\SQLEXPRESS; Initial Catalog=nv; Persist Security
Info=True; User ID=sa; Password=12345678";
SqlCommand cmd = new SqlCommand();
txtKetQua.Clear();
cmd.Connection = cnn;
cmd.CommandType = CommandType.Text;
cmd.CommandText = "select MaNV, HoTen form NhanVien";
txtKetQua.Text = cmd.CommandText;
```

Thực thi phát biểu lệnh: có ba phương thức để thực thi các phát biểu lệnh được định nghĩa bởi đối tượng Command là:

- ExecuteNonQuery: thực thi phát biểu lệnh SQL nhưng không trả về bất kỳ mẫu tin nào.
- ExcuteScalar: thực thi phát biểu lệnh SQL và trả về giá trị đầu tiên của cột đầu tiên.
- ExcuteReader: thực thi phát biểu lệnh SQL và trả về tập kết quả thông qua đối tượng DataReader.

Ví dụ 4: thiết kế giao diện như hình bên dưới:



Hình 2.3. Minh họa các thao tác với CSDL

Nút Connect: thực hiện kết nối CSDL, khi nhấp chuột vào nút Connect nếu kết nối thành công Textbox sẽ xuất hiện thông báo “Đã mở kết nối CSDL”, nếu không kết nối được CSDL sẽ xuất hiện thông báo “Kết nối CSDL không thành công!”.

Sự kiện Click của nút Connect:

– VB.NET:

```
cnn.ConnectionString = "Data Source=DESKTOP-4SL47N2\SQLEXPRESS;  
Initial Catalog=nv; Persist Security Info=True; User ID=sa;  
Password=12345678"  
cnn.Open()  
If cnn.State = ConnectionState.Open Then  
    txtKeQua.Text = "Đã mở kết nối CSDL!"  
Else  
    txtKeQua.Text = "Kết nối CSDL không thành công!"  
End If
```

– C#:

```
cnn.ConnectionString = @"Data Source=DESKTOP-4SL47N2\SQLEXPRESS;  
Initial Catalog=nv; Persist Security Info=True; User ID=sa;  
Password=12345678";  
cnn.Open();  
if (cnn.State == System.Data.ConnectionState.Open)  
{  
    txtKetQua.Text = "Đã mở kết nối CSDL!" ;  
}  
else  
{  
    txtKetQua.Text = "Kết nối CSDL không thành công!" ;  
}
```

Nút NonQuery: thực hiện đổi tên nhân viên có mã NV001 thành: Nguyễn Văn A. Sự kiện Click của nút NonQuery:

– VB.NET:

```
Dim sql As String  
Dim kq As Integer = 0  
sql = "UPDATE NhanVien set HoTen = N'Nguyễn Văn A' Where MaNV  
='NV0001'"  
Dim cmd As New SqlCommand(sql, cnn)  
Try  
    cmd.ExecuteNonQuery()  
    kq = 1  
Catch ex As Exception  
    MessageBox.Show(ex.Message)  
End Try  
txtKeQua.Text = ""  
If kq = 1 Then  
    txtKeQua.Text = "Cập nhật dữ liệu thành công!"
```

```
Else
    txtKeQua.Text = "Cập nhật dữ liệu KHÔNG thành công!"
End If
```

– C#:

```
string sql = "UPDATE NhanVien set HoTen = N'Nguyễn Văn A' Where
MaNV = 'NV0001'";
int kq = 0;
SqlCommand cmd = new SqlCommand(sql, cnn);
try
{
    cmd.ExecuteNonQuery ();
    kq = 1;
}
catch
{
    MessageBox.Show("cập nhật dữ liệu không thành công!");
}
if (kq == 1)
{
    txtKetQua.Text = "Cập nhật dữ liệu thành công!";
}
else
{
    txtKetQua.Text = "Cập nhật dữ liệu KHÔNG thành công!";
}
```

Ví dụ 5: Nút Scalar thực hiện đếm số lượng nhân viên. Sự kiện Click của nút Scalar:

– VB.NET:

```
Dim sql As String
Dim kq As Integer = 0
sql = "Select Count(*) From NhanVien"
Dim cmd As New SqlCommand(sql, cnn)
kq = cmd.ExecuteScalar
txtKeQua.Text = "Tổng số nhân viên: " & kq
```

– C#:

```
string sql = "Select Count(*) From NhanVien";
int kq = 0;
SqlCommand cmd = new SqlCommand(sql, cnn);
kq = Convert.ToInt32 (cmd.ExecuteScalar());
txtKetQua.Text = "Tổng số nhân viên " + kq;
```

2.3. Đối tượng DataReader

DataReader cung cấp khả năng chỉ đọc, hướng tới và không lưu giữ trong bộ nhớ đệm các dòng dữ liệu được tạo bởi phương thức ExecuteReader của đối tượng Command. DataReader thuộc tên miền System.Data.OleDbDataReader hoặc System.Data.SqlDataReader

Ví dụ 6: nút Reader lọc ra danh sách những nhân viên có địa chỉ ở Tiền Giang:

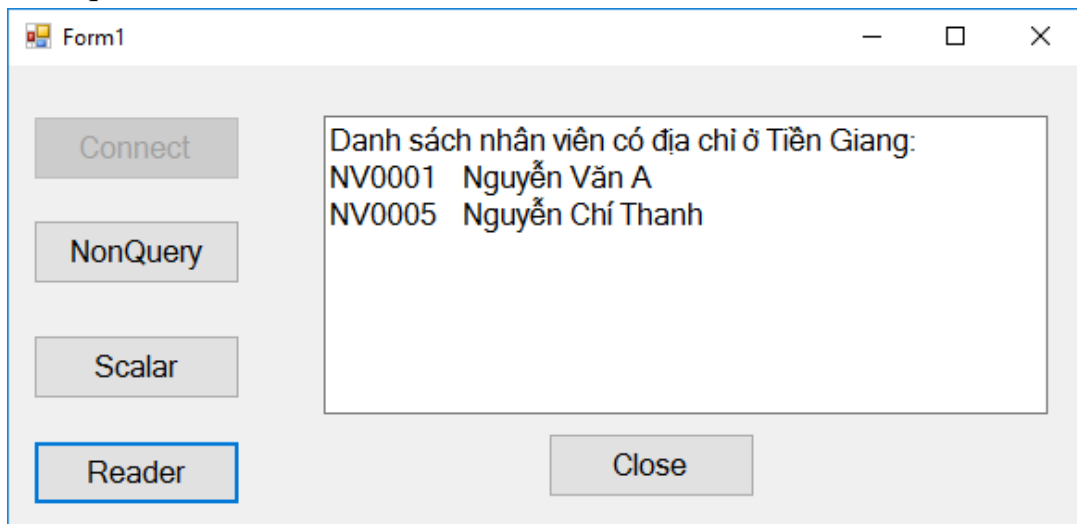
– VB.NET:

```
Dim sql As String
Dim kq As Integer = 0
txtKeQua.Text = "Danh sách nhân viên có địa chỉ ở Tiền Giang: "
& vbNewLine
sql = "Select MaNV, HoTen From NhanVien Where DiaChi like N'Tiền
Giang'"
Dim cmd As New SqlCommand(sql, cnn)
Dim rd As SqlDataReader
rd = cmd.ExecuteReader
While rd.Read
    txtKeQua.Text &= rd("MaNV") & ControlChars.Tab & rd("HoTen")
    & vbNewLine
End While
rd.Close()
```

– C#:

```
string sql = "Select MaNV, HoTen From NhanVien Where DiaChi like
N'Tiền Giang'";
int kq = 0;
txtKetQua.Text = "Danh sách nhân viên có địa chỉ ở Tiền Giang:";
SqlCommand cmd = new SqlCommand(sql, cnn);
SqlDataReader rd = cmd.ExecuteReader();
while (rd.Read())
{
    txtKetQua.Text += (string)rd["manv"] + "\t" +
    (string)rd["hoten"] + "\n";
}
rd.Close();
```

Kết quả sẽ hiển thị ở Textbox sau khi nhấn nút Reader:



Hình 2.4. Minh họa sử dụng DataReader.

Trong khi DataReader đang mở, các thao tác dữ liệu khác trên nguồn dữ liệu đều không thể thao tác được đến khi DataReader đóng lại bằng lệnh Close().

2.4. Đối tượng DataSet

DataSet là đối tượng chính trong ADO.NET. DataSet cung cấp mô hình dữ liệu trong bộ nhớ, hoàn toàn không giữ kết nối với nguồn dữ liệu và dễ dàng truyền qua lại giữa các không gian lưu trữ.

Các ứng dụng tiêu biểu sử dụng trong DataSet bao gồm:

- Dữ liệu ứng dụng: DataSet là một đối tượng linh động và đơn giản để sử dụng cho việc lưu trữ dữ liệu ứng dụng cục bộ. Việc truy xuất dữ liệu dễ dàng như truy xuất bảng, DataSet còn cung cấp tính năng sắp xếp và lọc dữ liệu.

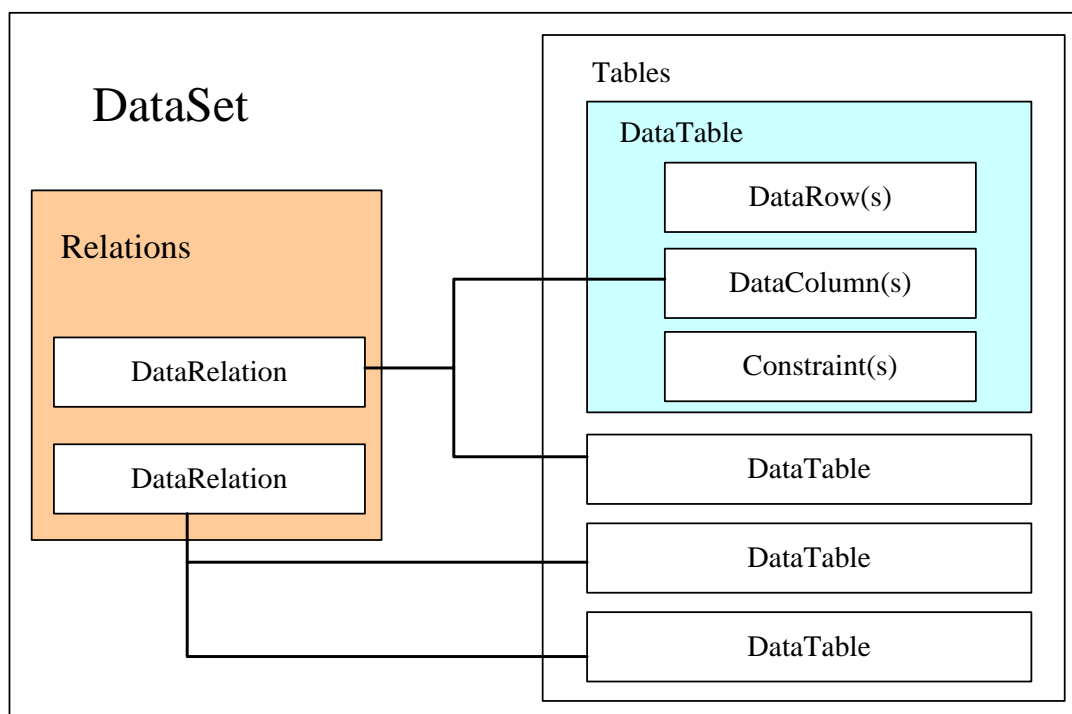
- Truy xuất dữ liệu từ xa: DataSet tự động dùng định dạng XML để truyền dữ liệu từ máy tính này sang máy tính khác. Khả năng này giúp thuận lợi hơn cho việc phát triển ứng dụng Web Service.

- Lưu trữ dữ liệu vào bộ nhớ: DataSet có thể lưu đệm dữ liệu trong khi phát triển hoặc dùng cho các kiểu ứng dụng phân tán.

- Lưu dữ liệu vào thiết bị lưu trữ (đĩa cứng,...): DataSet cung cấp phương thức lưu trữ dữ liệu và thông tin giảm đồ ở dạng XML chuẩn dễ dàng lưu vào thiết bị lưu trữ sử dụng cho mục đích khác.

- Giao tiếp người dùng: DataSet hỗ trợ hiệu quả về giao tiếp người dùng cho các giao diện ứng dụng khác nhau bằng cách kết hợp sắp xếp, lọc và khả năng ràng buộc với các View dữ liệu khác nhau cho cả Windows Form và Web Form.

Dữ liệu trong DataSet được tổ chức thành một hoặc nhiều DataTable. Hình bên dưới cho thấy cấu trúc của DataSet và các đối tượng bên trong của nó.



Hình 2.5. Các đối tượng trong Dataset

Các đối tượng có thể được mô tả như sau:

- DataTable: mỗi DataTable chứa tập các đối tượng DataRow, DataColumn và Constraint cùng với tập các đối tượng DataRelation liên kết các bảng cha, con.
- DataColumn: là đơn vị cơ bản để nhận và định nghĩa giản đồ của DataTable, chứa thông tin cụ thể của từng cột trong DataTable bao gồm tên, kiểu dữ liệu và các thuộc tính khác (Unique, ReadOnly, AllowDBNull, AutoIncrement). DataColumn cũng có thuộc tính Expression cho phép tính toán giá trị cho cột hoặc để tạo cột tính toán.
- DataRow: biểu diễn một dòng (mẫu tin) dữ liệu trong DataTable. DataRow cũng được dùng để nhận, thêm, sửa đổi dữ liệu trong DataTable, có thể duyệt tuần tự tập các dòng của DataTable hoặc truy xuất một dòng xác định.
- DataRelation: định nghĩa quan hệ giữa các bảng trong DataSet, thể hiện quan hệ giữa các dòng dữ liệu trong hai bảng, có thể duyệt quan hệ giữa các bảng trong DataTable.
- Constraint: định nghĩa các luật bắt buộc phải tuân thủ để đảm bảo tính toàn vẹn của dữ liệu. UniqueConstraint đảm bảo tất cả giá trị trong bảng là duy nhất. ForeignKeyConstraint quyết định hành động nào được thực hiện trên các dòng trong bảng quan hệ. Ràng buộc được định nghĩa có thể bao gồm một hoặc nhiều DataColumn. Mỗi một DataTable có một thuộc tính Constraint.

Tương tác với DataSet:

- DataTable:
 - + Thêm một bảng vào DataSet:
 - `Tables.Add()`: một bảng mới được tạo ra với tên mặc định là Table1, Table2,... và đưa vào tập hợp Tables.
 - `Tables.Add(<Tên bảng>)`
 Ví dụ 7: thêm một bảng vào DataSet.

```
Dim ds As New DataSet("ví dụ")
Dim bang As New DataTable("bảng 1")
Ds.Tables.Add(bang)
```
 - + Thêm nhiều bảng vào DataSet:
 `Tables.AddRange(<mảng bảng>)`
 <mảng bảng>: là một mảng các bảng đã tạo ra muốn đưa vào DataSet.
 Ví dụ 8: thêm nhiều bảng vào DataSet.

```
Dim ds As New DataSet("Ví dụ")
Dim bang1 As New DataTable("bảng 1")
Dim bang2 As New DataTable("bảng 2")
ds.Tables.AddRange(New DataTable(){bang1, bang2})
```
 - + Xóa bảng khỏi DataSet:
 - `Tables.Remove(<bảng>)`
 - `Tables.RemoveAt(<chỉ số của bảng>)`
 - Xóa tất cả các bảng: `Tables.Clear()`
 - Nếu bảng đang hiển thị dữ liệu trên Form thì không thể xóa được. Vì vậy trước khi xóa cần kiểm tra xem dữ liệu có thể xóa được không: `Tables.CanRemove(<bảng>)` (True: có thể xóa được, False: không thể xóa).
 Ví dụ 9: Xóa bảng

```
Dim bang As DataTable = ds.Table(0)
If ds.Tables.CanRemove(bang) Then
    ds.Tables.Remove(bang)
End If
```
 - + Kiểm tra bảng có thuộc về DataSet:
 `Tables.Contains(<tên bảng>)` (True: nếu trong Tables có bảng tên <tên bảng>, False: không có).
 - + Lấy chỉ số của bảng:
 `Tables.IndexOf(<tên bảng>)`
 - + Đếm số bảng có trong DataSet:
 `Tables.Count()`
 - + Lấy ra một bảng trong DataSet:
 `Tables(<chỉ số>)` hoặc `Tables.Item(<chỉ số>)`
 Ví dụ 10: thiết kế Form như hình bên dưới, với nút **Create DataSet** (bntCreateDataSet) dùng để thiết lập định nghĩa cấu trúc các bảng và các thuộc tính cho mỗi cột trong bảng, thêm dữ liệu vào các bảng này, hiển thị dữ liệu từ các bảng này vào ListBox (lstOutput).

+ VB.NET:

Sự kiện Click của nút Create DataSet:

CreateDataSet()

AddData()

Với thủ tục CreateDataSet: tạo một DataSet và định nghĩa cấu trúc các bảng sẽ sử dụng, bao gồm định nghĩa đối tượng DataColumn và thiết lập các thuộc tính cho mỗi cột trong bảng.

VB.NET:

```
Private Sub CreateDataSet()  
    dsNV = New DataSet  
    'Tạo Table NhanVien  
    Dim dtNV As DataTable = New DataTable("NhanVien")  
    dtNV.CaseSensitive = False  
    dtNV.Columns.Add("MaNV", Type.GetType("System.String"))  
    dtNV.Columns.Add("HoTen", Type.GetType("System.String"))  
    dtNV.Columns.Add("DiaChi", Type.GetType("System.String"))  
    dtNV.Columns.Add("NgaySinh",  
Type.GetType("System.DateTime"))  
    dtNV.Columns.Add("MaPhong", Type.GetType("System.String"))  
    'Thêm bảng Nhân viên vào dsNV  
    dsNV.Tables.Add(dtNV)  
  
    Dim dtPB As DataTable = New DataTable("PhongBan")  
    dtPB.CaseSensitive = False  
    dtPB.Columns.Add("MaPhong", Type.GetType("System.String"))  
    dtPB.Columns.Add("TenPhong", Type.GetType("System.String"))  
    dsNV.Tables.Add(dtPB)  
End Sub
```

C#:

```
DataSet dsNV;  
private void CreateDataSet()  
{  
    dsNV = new DataSet();  
    //Tạo Table NhanVien  
    DataTable dtNV = new DataTable("NhanVien");  
    dtNV.CaseSensitive = false;  
    dtNV.Columns.Add("MaNV", Type.GetType("System.String"));  
    dtNV.Columns.Add("HoTen", Type.GetType("System.String"));  
    dtNV.Columns.Add("DiaChi", Type.GetType("System.String"));  
    dtNV.Columns.Add("NgaySinh", Type.GetType("System.DateTime"));  
    dtNV.Columns.Add("MaPhong", Type.GetType("System.String"));  
    dsNV.Tables.Add(dtNV); //Thêm bảng Nhân viên vào dsNV  
    DataTable dtPB = new DataTable("PhongBan");  
    dtPB.CaseSensitive = false;  
    dtPB.Columns.Add("MaPhong", Type.GetType("System.String"));  
    dtPB.Columns.Add("TenPhong", Type.GetType("System.String"));  
    dsNV.Tables.Add(dtPB);  
}
```

– Thêm dữ liệu vào DataTable: ngay sau khi định nghĩa DataTable và giải đồ của nó, có thể thêm dữ liệu vào bảng. Thủ tục AddData: thêm 2 dòng vào bảng PhongBan, 4 dòng vào bảng NhanVien theo quy trình như sau:

+ Tạo thể hiện DataRow cho bảng muốn thêm dữ liệu bằng cách gọi phương thức NewRow của DataTable.

+ Gán giá trị cho các cột của dòng đó.

+ Thêm dòng vào tập Rows của bảng bằng phương thức Add (Row.Add).

VB.NET:

```
Private Sub AddData()  
    Dim dtPB As DataTable = dsNV.Tables("PhongBan")  
    Dim dtNV As DataTable = dsNV.Tables("NhanVien")  
    'Thêm 2 dòng vào bảng Phòng ban  
    Dim rPB As DataRow  
    rPB = dsNV.Tables("PhongBan").NewRow  
    rPB("MaPhong") = "P001"  
    rPB("TenPhong") = "Phòng kte"  
    dtPB.Rows.Add(rPB)  
    'Dim rPB As DataRow  
    rPB = dsNV.Tables("PhongBan").NewRow  
    rPB("MaPhong") = "P002"  
    rPB("TenPhong") = "Phòng Tin học"  
    dtPB.Rows.Add(rPB)  
    'Thêm 4 dòng vào bảng Nhân Viên  
    Dim rNV As DataRow  
    rNV = dsNV.Tables("NhanVien").NewRow  
    rNV("MaNV") = "NV001"  
    rNV("HoTen") = "Nguyễn Văn A"  
    rNV("DiaChi") = "Tiền Giang"  
    rNV("NgaySinh") = "12/1/2000"  
    rNV("MaPhong") = "P001"  
    dtNV.Rows.Add(rNV)  
    rNV = dsNV.Tables("NhanVien").NewRow  
    rNV("MaNV") = "NV002"  
    rNV("HoTen") = "Nguyễn Văn B"  
    rNV("DiaChi") = "Bến Tre"  
    rNV("NgaySinh") = "23/1/2000"  
    rNV("MaPhong") = "P001"  
    dtNV.Rows.Add(rNV)  
    rNV = dsNV.Tables("NhanVien").NewRow  
    rNV("MaNV") = "NV003"  
    rNV("HoTen") = "Nguyễn Văn C"  
    rNV("DiaChi") = "Long An"  
    rNV("NgaySinh") = "23/1/2000"  
    rNV("MaPhong") = "P002"  
    dtNV.Rows.Add(rNV)
```



```

rNV = dsNV.Tables("NhanVien").NewRow
rNV("MaNV") = "NV004"
rNV("HoTen") = "Nguyễn Văn D"
rNV("DiaChi") = "Tiền Giang"
rNV("NgaySinh") = "13/10/2000"
rNV("MaPhong") = "P002"
dtNV.Rows.Add(rNV)
MsgBox("Thêm thành công!")
End Sub

```

C#:

```

private void AddData()
{
    DataTable dtPB = dsNV.Tables ["PhongBan"];
    DataTable dtNV = dsNV.Tables["NhanVien"];
    //Thêm 2 dòng vào bảng Phòng ban
    DataRow rPB;
    rPB = dsNV.Tables["PhongBan"].NewRow ();
    rPB["MaPhong"] = "P001";
    rPB["TenPhong"] = "Phòng kte";
    dtPB.Rows.Add(rPB);
    rPB = dsNV.Tables["PhongBan"].NewRow();
    rPB["MaPhong"] = "P002";
    rPB["TenPhong"] = "Phòng Tin học";
    dtPB.Rows.Add(rPB);
    //Thêm 4 dòng vào bảng Nhân Viên
    DataRow rNV;
    rNV = dsNV.Tables["NhanVien"].NewRow();
    rNV["MaNV"] = "NV001";
    rNV["HoTen"] = "Nguyễn Văn A";
    rNV["DiaChi"] = "Tiền Giang";
    rNV["NgaySinh"] = "12/1/2000";
    rNV["MaPhong"] = "P001";
    dtNV.Rows.Add(rNV);
    rNV = dsNV.Tables["NhanVien"].NewRow();
    rNV["MaNV"] = "NV002";
    rNV["HoTen"] = "Nguyễn Văn B";
    rNV["DiaChi"] = "Bến Tre";
    rNV["NgaySinh"] = "23/1/2000";
    rNV["MaPhong"] = "P001";
    dtNV.Rows.Add(rNV);
    rNV = dsNV.Tables["NhanVien"].NewRow();
    rNV["MaNV"] = "NV003";
    rNV["HoTen"] = "Nguyễn Văn C";
    rNV["DiaChi"] = "Long An";
    rNV["NgaySinh"] = "23/1/2000";
    rNV["MaPhong"] = "P002";
    dtNV.Rows.Add(rNV);
    rNV = dsNV.Tables["NhanVien"].NewRow();
    rNV["MaNV"] = "NV004";

```

```

        rNV["HoTen"] = "Nguyễn Văn D";
        rNV["DiaChi"] = "Tiền Giang";
        rNV["NgaySinh"] = "13/10/2000";
        rNV["MaPhong"] = "P002";
        dtNV.Rows.Add(rNV);
    }

```

– Truy xuất dữ liệu từ DataTable: do DataSet và các bảng DataTable mà nó chứa luôn được truy xuất và không cần kết nối tới nguồn dữ liệu. Mỗi DataTable đều có thuộc tính Rows là tập các đối tượng DataRow. Mỗi đối tượng DataRow được truy xuất bằng chỉ mục hoặc dùng *For Each*. Thủ tục DisplayDataSet() để hiển thị nội dung các bảng đã tạo trước đó.

VB.NET:

```

Private Sub DisplayDataSet(ByVal ds As DataSet)
    Dim dt As DataTable
    Dim dr As DataRow
    Dim dc As DataColumn
    lstOutput.Items.Add("DataSet")
    lstOutput.Items.Add("=====")
    For Each dt In ds.Tables
        lstOutput.Items.Add(" ")
        lstOutput.Items.Add("Table: " & dt.TableName)
        lstOutput.Items.Add("-----")
        For Each dr In dt.Rows
            For Each dc In dt.Columns
                lstOutput.Items.Add(dc.ColumnName & ":" & dr(dc))
            Next
            lstOutput.Items.Add("_____")
        Next
    Next
End Sub

```

C#:

```

private void DisplayDataSet(DataSet ds)
{
    lstOutput.Items.Add("DataSet");
    lstOutput.Items.Add("=====");
    foreach (DataTable dt in ds.Tables)
    {
        lstOutput.Items.Add(" ");
        lstOutput.Items.Add("Table: " + dt.TableName);
        lstOutput.Items.Add("-----");
        foreach (DataRow dr in dt.Rows)
        {
            foreach (DataColumn dc in dt.Columns)
            {
                lstOutput.Items.Add(dc.ColumnName + ":" + dr[dc]);
            }
        }
    }
}

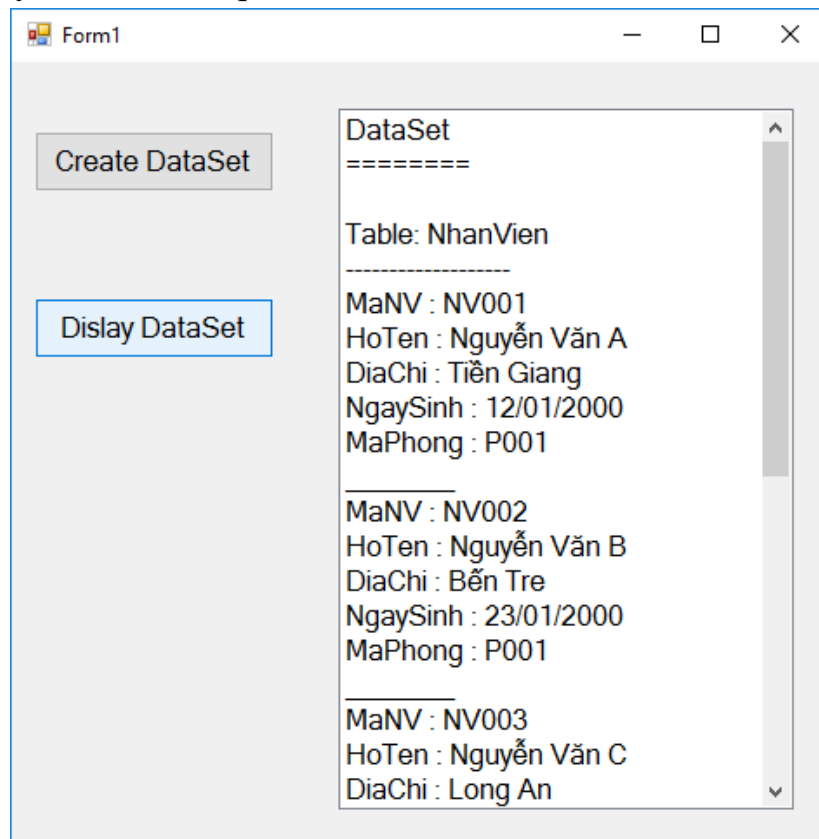
```

```

        1stOutput.Items.Add("_____");
    }
}
}

```

Chạy thử chương trình, nhấp chuột vào nút Create DataSet rồi nhấp chuột vào nút Display DataSet, kết quả hiển thị như hình bên dưới:



Hình 2.6. Minh họa các thao tác dữ liệu trong DataSet

- Tìm kiếm, lọc và sắp xếp:

Find là phương thức của thuộc tính Rows trong đối tượng DataTable, phương thức này được dùng để tìm kiếm và trả về một dòng được xác định bằng giá trị khóa chính của bảng. Trước khi dùng phương thức Find phải định nghĩa khóa chính cho bảng bằng thuộc tính PrimaryKey của bảng.

Ví dụ 11: để tạo thuộc tính MaPhong làm khóa chính cho bảng PhongBan, thêm đoạn Code sau vào cuối thủ tục CreateDataSet()

- VB.NET:

```

Dim pk(0) As DataColumn
pk(0) = dtPB.Columns("MaPhong")
dtPB.PrimaryKey = pk

```

– C#:

```
DataColumn[] pk = new DataColumn[1];  
pk[0] = dtPB.Columns["MaPhong"];  
dtPB.PrimaryKey = pk;
```

Lưu ý: ngay cả khi khóa chính chỉ gồm một cột thì thuộc tính PrimaryKey cũng là một mảng kiểu DataColumn.

Sau khi định nghĩa khóa chính, sử dụng Find để tìm dòng theo điều kiện:

– VB.NET:

```
Dim frow As DataRow  
Dim value(0) As Object  
value(0) = "P001"  
frow = dsNV.Tables("PhongBan").Rows.Find(value)  
lstOutput.Items.Add(frow("MaPhong"))  
lstOutput.Items.Add(frow("TenPhong"))
```

– C#:

```
Object[] value = new Object [1];  
value[0] = "P001";  
DataRow frow = dsNV.Tables["PhongBan"].Rows.Find(value);  
lstOutput.Items.Add(frow["MaPhong"]);  
lstOutput.Items.Add(frow["TenPhong"]);
```

Phương thức Select của DataTable trả về một mảng các đối tượng DataRow. Có thể sắp xếp thứ tự các dòng thỏa điều kiện lọc.

Ví dụ 12: tìm những nhân viên có địa chỉ ở Tiền Giang và sắp xếp giảm dần theo MaNV:

– VB.NET:

```
Dim selectrow(5) As DataRow  
selectrow = dsNV.Tables("NhanVien").Select("DiaChi='Tiền  
Giang'", "MaNV DESC")  
For i = 0 To selectrow.GetUpperBound(0)  
    lstOutput.Items.Add(selectrow(i)("MaNV"))  
    lstOutput.Items.Add(selectrow(i)("HoTen"))  
    lstOutput.Items.Add(selectrow(i)("DiaChi"))  
Next
```

– C#:

```
DataRow[] selectrow = new DataRow[5];  
selectrow = dsNV.Tables["NhanVien"].Select("DiaChi='Tiền  
Giang'", "MaNV DESC");  
for (int i = 0; i <= selectrow.GetUpperBound(0); i++ )  
{  
    lstOutput.Items.Add(selectrow[i]["MaNV"]);  
    lstOutput.Items.Add(selectrow[i]["HoTen"]);  
    lstOutput.Items.Add(selectrow[i]["DiaChi"]);  
}
```

– Cập nhật DataSet:

Để cập nhật dòng dữ liệu trong bảng, truy xuất dữ liệu dòng đó rồi gán giá trị mới cho một trong các cột của nó.

Ví dụ 13: thay đổi địa chỉ của nhân viên của dòng thứ 3 thành “TP.HCM”:

```
dsNV.Tables(“NhanVien”).Rows(2)(“DiaChi”) = “TP.HCM”
```

Có thể thay đổi bao nhiêu lần tùy ý nhưng giá trị thay đổi vẫn ở trạng thái chờ cho đến khi phương thức *AcceptChanges* được gọi và xác nhận thay đổi. Phương thức *RejectChanges* được dùng để hủy bỏ bất kỳ thay đổi được thực hiện kể từ lúc dữ liệu được nạp hoặc *AcceptChanges* được gọi lần sau cùng.

Chú ý: Phương thức *AcceptChanges* và *RejectChanges* có mặt ở các cấp độ khác nhau. Các lớp *DataSet*, *DataTable* và *DataRow* đều hỗ trợ phương thức này. Nếu *AcceptChanges* của *DataSet* được gọi thì *AcceptChanges* của tất cả các bảng trong *DataSet* cũng được gọi. Nếu *AcceptChanges* của *DataTable* được gọi thì *AcceptChanges* của tất cả các dòng trong bảng cũng được gọi.

RowState: trạng thái của *DataRow*.

Các giá trị của RowState

- + Unchanged: không có sự thay đổi nào kể từ lần gọi *AcceptChanges* sau cùng hoặc dòng được nạp ban đầu bởi *DataAdapter*.

- + Added: dòng được thêm vào bảng nhưng *AcceptChanges* chưa được gọi.

- + Deleted: phương thức này được gọi để xóa dòng nhưng *AcceptChanges* chưa được gọi.

- + Modified: dòng được thay đổi nhưng *AcceptChanges* chưa được gọi.

- + Detached: dòng được tạo ra nhưng chưa được thêm vào bảng hoặc phương thức *Remove* được gọi để xóa dòng khỏi bảng hoặc phương thức *Delete* được gọi và *AcceptChanges*

Các giá trị của *DataRowVersion*:

- + Original: các giá trị gốc của dòng, phiên bản này không tồn tại đối với dòng có trạng thái là Added.

- + Current: các giá trị hiện thời của dòng, phiên bản này không tồn tại với dòng có trạng thái là Deleted.

- + Default: phiên bản dòng mặc định của dòng tùy thuộc vào trạng thái hiện tại của dòng. Nếu trạng thái dòng là Deleted thì phiên bản dòng mặc định là Original. Nếu trạng thái dòng là Detached thì phiên bản dòng là Proposed. Ngược lại phiên bản dòng là Current.

- + Proposed: giá trị đề nghị của dòng, phiên bản này chỉ tồn tại khi soạn thảo hoặc đối với dòng không được thêm vào bảng.

Có thể dùng phương thức HasVersion của DataRow để kiểm tra sự hiện diện của một phiên bản trong dòng hiện hành.

- Tạo quan hệ giữa các bảng

Để thiết lập mối quan hệ cha – con giữa hai bảng trong một DataSet, phải thỏa các yêu cầu sau:

- + Dữ liệu của Field hoặc các Field của bảng cha trong quan hệ phải là duy nhất.

- + Chỉ có thể thiết lập mối quan hệ giữa hai bảng trong DataSet.

Sử dụng phương thức Add của tập hợp Relations trong DataSet với cú pháp như sau: Relations.Add(<đối tượng DataRelation>).

<đối tượng DataRelation>: đối tượng này phải được tạo sẵn.

Relations.Add(<DataColumn trên bảng cha>, <DataColumn trên bảng con>)

Relations.Add(<mảng DataColumn trên bảng cha>, <mảng DataColumn trên bảng con>)

Relations.Add(<tên quan hệ>, <DataColumn trên bảng cha>, <DataColumn trên bảng con>)

Relations.Add(<tên quan hệ>, <mảng DataColumn trên bảng cha>, <mảng DataColumn trên bảng con>)

- Ràng buộc giữa các bảng

- + Ràng buộc là những quy tắc được dùng để bắt buộc một số những giới hạn nhất định trên một hoặc nhiều cột của bảng để đảm bảo tính toàn vẹn của dữ liệu. ADO.NET hỗ trợ hai kiểu ràng buộc trong bảng: UniqueConstraint và ForeignKeyConstraint. UniqueConstraint: đảm bảo tất cả dữ liệu trong các cột chỉ ra là duy nhất trong bảng. ForeignKeyConstraint: định nghĩa quan hệ khóa chính – khóa ngoại giữa các cột trong hai bảng và cho phép mô tả hành động.

- + Tất cả các ràng buộc chỉ có hiệu lực khi thuộc tính EnforceConstraints của DataSet có giá trị là True, giá trị mặc định của thuộc tính này là True.

- + UniqueConstraint tự động được tạo và được thêm vào tập Constraints của DataTable khi thiết lập thuộc tính Unique của DataColumn thành True và khi tạo khóa chính cho DataTable. Cả UniqueConstraint và ForeignKeyConstraint cũng được tạo tự động khi tạo quan hệ giữa hai bảng, UniqueConstraint: được tạo trên các cột quan hệ trong bảng cha và ForeignKeyConstraint được tạo trên các cột quan hệ của bảng con.

Ví dụ 14: thiết lập nút btnCreateRelations dùng để thiết lập mối quan hệ giữa các bảng trong DataSet (PhongBan, NhanVien)

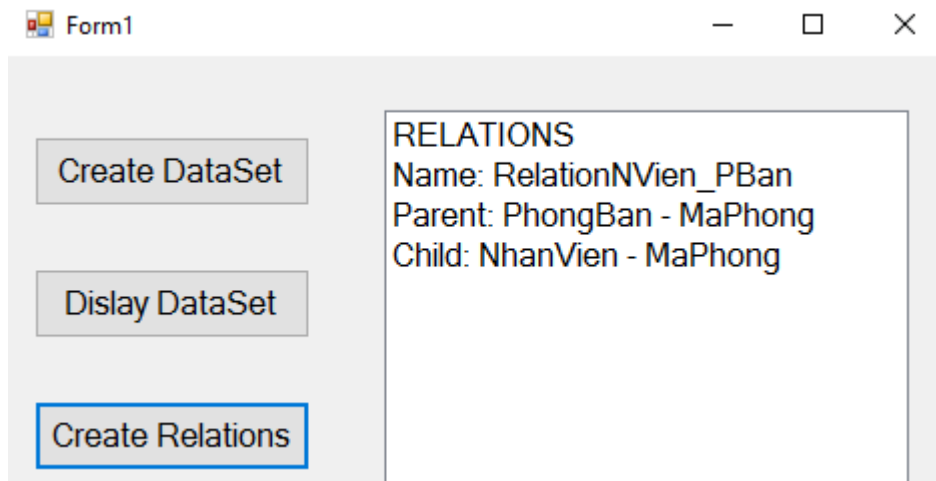
Sự kiện Click của nút btnCreateRelations:

VB.NET:

```
Dim rl As DataRelation
CreateDataSet()
'Tạo quan hệ giữa bảng NhanVien và PhongBan
rl = dsNV.Relations.Add("RelationNVien_PBan",
dsNV.Tables("PhongBan").Columns("MaPhong"),
dsNV.Tables("NhanVien").Columns("MaPhong"))
DislayRelations(dsNV)

Public Sub DislayRelations(ByVal ds As DataSet)
    Dim rl As DataRelation
    lstOutput.Items.Add("RELATIONS")
    For Each rl In ds.Relations
        'hiển thị tên quan hệ
        lstOutput.Items.Add("Name: " & rl.RelationName)
        'bảng cha
        lstOutput.Items.Add("Parent: " &
rl.ParentTable.ToString & " - " &
rl.ParentColumns(0).ColumnName)
        'bảng con
        lstOutput.Items.Add("Child: " & rl.ChildTable.ToString
& " - " & rl.ChildColumns(0).ColumnName)
    Next
End Sub
```

Kết quả hiển thị sau khi chương trình thực thi:



Hình 2.7. Minh họa tạo mối quan hệ giữa các bảng.

2.5. Đối tượng DataAdapter

DataAdapter lấy dữ liệu nguồn về cho ứng dụng, lấy cấu trúc và dữ liệu của các bảng trong nguồn dữ liệu.

DataAdapter là một bộ gồm bốn đối tượng Command:

- SelectCommand: cho phép lấy dữ liệu từ nguồn dữ liệu.
- InsertCommand: cho phép thêm dữ liệu vào bảng trong nguồn dữ liệu.
- UpdateCommand: cho phép sửa dữ liệu trên bảng trong nguồn dữ liệu.

- DeleteCommand: cho phép xóa dữ liệu trên bảng trong nguồn dữ liệu.

Thông thường chỉ cần khai báo nội dung lệnh cho SelectCommand, các nội dung của Command còn lại có thể được phát sinh nhờ đối tượng CommandBuilder dựa vào nội dung của SelectCommand.

DataAdapter thuộc tên miền System.Data.OleDb.OleDbDataAdapter hoặc System.Data.SqlClient.SqlDataAdapter.

Cú pháp tạo DataAdapter như sau:

- New <loại> DataAdapter().
- New <loại> DataAdapter(<đối tượng SelectCommand>).
+ <đối tượng SelectCommand>: đối tượng Command có sẵn với nội dung lệnh truy xuất.
- New <loại> DataAdapter(<lệnh>, <đối tượng Connection>)
+ <lệnh>: câu lệnh Sql, tên StoreProcedure để thực hiện truy xuất từ nguồn dữ liệu.
+ <đối tượng Connection>: đối tượng Connection mà DataAdapter thao tác với nguồn dữ liệu.
- New <loại> DataAdapter(<lệnh>, <chuỗi Connection>).
+ <chuỗi Connection>: chuỗiConnectionString để tạo một đối tượng Connection.

DataAdapter chỉ thao tác được với nguồn dữ liệu qua một đối tượng Connection đang kết nối. Nếu Connection chưa mở, DataAdapter sẽ tự động mở kết nối khi cần và tự đóng lại.

Ví dụ 15: viết câu kết nối với CSDL SQL Server:

- VB.NET:

```
Dim cnn As New SqlConnection
cnn.ConnectionString = "Data Source=DESKTOP-4SL47N2\SQLEXPRESS;
Initial Catalog=nv; Persist Security Info=True; User ID=sa;
Password=12345678"
cnn.Open()
Dim da As New SqlDataAdapter("Select *from NhanVien", cnn)
```

- C#:

```
SqlConnection cnn = new SqlConnection();
cnn.ConnectionString = @"Data Source=DESKTOP-4SL47N2\SQLEXPRESS;
Initial Catalog=nv; Persist Security Info=True; User ID=sa;
Password=12345678";
cnn.Open();
SqlDataAdapter da;
da = new SqlDataAdapter("Select *from NhanVien", cnn);
```

Các thuộc tính chính của DataAdapter:

- SelectCommand: chứa nội dung lệnh truy xuất các mẫu tin từ nguồn dữ liệu.

- InsertCommand: chứa nội dung lệnh chèn các mẫu tin mới vào nguồn dữ liệu.
- DeleteCommand: chứa nội dung lệnh hủy các mẫu tin trên nguồn dữ liệu.
- UpdateCommand: chứa nội dung lệnh cập nhật các mẫu tin trên nguồn dữ liệu.
- TableMappings: tập hợp các ánh xạ tên bảng khi DataAdapter đổ dữ liệu hay cấu trúc vào đối tượng DataTable hoặc DataSet.

Các chức năng chính của DataAdapter:

- Tạo bộ lệnh cập nhật cho DataAdapter

Sử dụng CommandBuilder thuộc tên miền

System.Data.SqlClient.SqlCommandBuilder

Cách khai báo:

New SqlCommandBuilder(<đối tượng DataAdapter>)

Mỗi DataAdapter chỉ có thể kết hợp với một CommandBuilder:

+ CommandBuilder chỉ phát sinh nội dung lệnh cập nhật cho các DataAdapter có nội dung SelectCommand chỉ truy xuất đến một bảng nguồn.

+ Nội dung trong SelectCommand phải có khóa chính để DataAdapter phân biệt các dòng khi cập nhật. Nếu không, CommandBuilder sẽ không phát sinh được nội dung lệnh cho các Command Select, Update, Delete.

+ Nếu nội dung của SelectCommand là truy vấn từ nhiều bảng hoặc từ một StoreProcedure thì các Command cập nhật không tự động phát sinh nhưng phải khai báo các Command cập nhật.

+ Sau khi lệnh được phát sinh, nếu nội dung lệnh của SelectCommand thay đổi, lệnh của các Command khác không tự động thay đổi theo cho đến khi phương thức RefreshSchema của CommandBuilder được gọi.

+ Nếu nội dung của SelectCommand là nhiều câu Select SQL, CommandBuilder chỉ tạo được lệnh cập nhật cho lệnh truy vấn đầu tiên.

- Để cập nhật dữ liệu về nguồn

+ Update(<mảng dòng>): cập nhật các dòng <mảng dòng> vào nguồn dữ liệu.

<mảng dòng>: mảng các đối tượng lớp DataRow, thường mảng này là kết quả trả về của phương thức GetChanges của DataSet.

+ Update(<dataset>): cập nhật các thay đổi trên tất cả các bảng của <dataset> vào nguồn dữ liệu.

<dataset>: đối tượng DataSet mà DataAdapter sẽ cập nhật vào nguồn.

+ Update(<datatable>): cập nhật các thay đổi trên DataTable vào nguồn dữ liệu.

<datatable>: đối tượng DataTable mà DataAdapter sẽ cập nhật vào nguồn.

+ Update(<dataset>, <tên bảng>): cập nhật các thay đổi trên bảng có tên <tên bảng> trong DataSet vào nguồn dữ liệu.

Khi phương thức Update được gọi, DataAdapter sẽ kiểm tra tình trạng các dòng là thêm mới, sửa đổi, xóa và gọi thực hiện tự động các Command tương ứng cho mỗi dòng. Nếu các Command chưa được tạo sẽ phát sinh lỗi. Có thể tạo và gán cho mỗi loại Command trên DataAdapter hoặc tự động phát sinh thông qua CommandBuilder. Mỗi Command trên DataAdapter có một tập hợp tham số kết hợp với nó. Các tham số này được ánh xạ với dòng đang cập nhật thông qua thuộc tính:

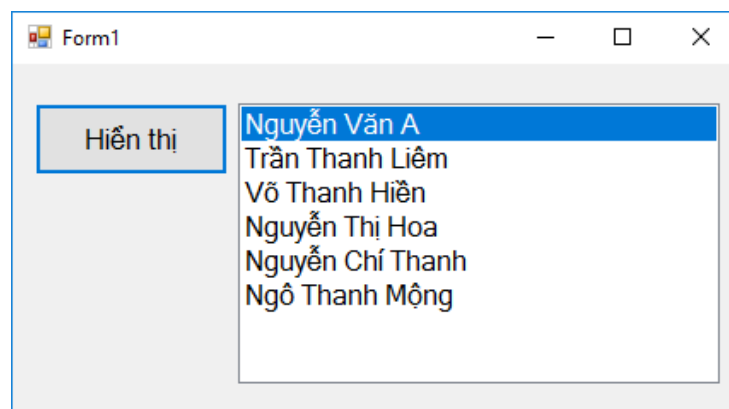
+ SourceColumn: cho biết tên cột trên dòng được cập nhật liên kết với tham số.

+ SourceVersion: cho biết giá trị của phiên bản nào trên dòng được cập nhật truyền vào vị trí tham số.

Các giá trị của SourceVersion:

- ✓ Current: tham số dùng giá trị hiện hành của cột, đây là giá trị mặc định.
- ✓ Default: tham số dùng giá trị mặc định đã qui định cho cột.
- ✓ Original: tham số dùng giá trị gốc: giá trị từ khi bảng được lấy về từ nguồn dữ liệu hay từ lần gọi cập nhật lần trước.
- ✓ Proposed: tham số sử dụng một giá trị đề nghị.

Ví dụ với CSDL quản lý nhân viên được xây dựng trong SQL Server: hiển thị danh sách nhân viên lên ListBox như hình:



Hình 2.8. Minh họa đọc dữ liệu bằng DataAdapter.

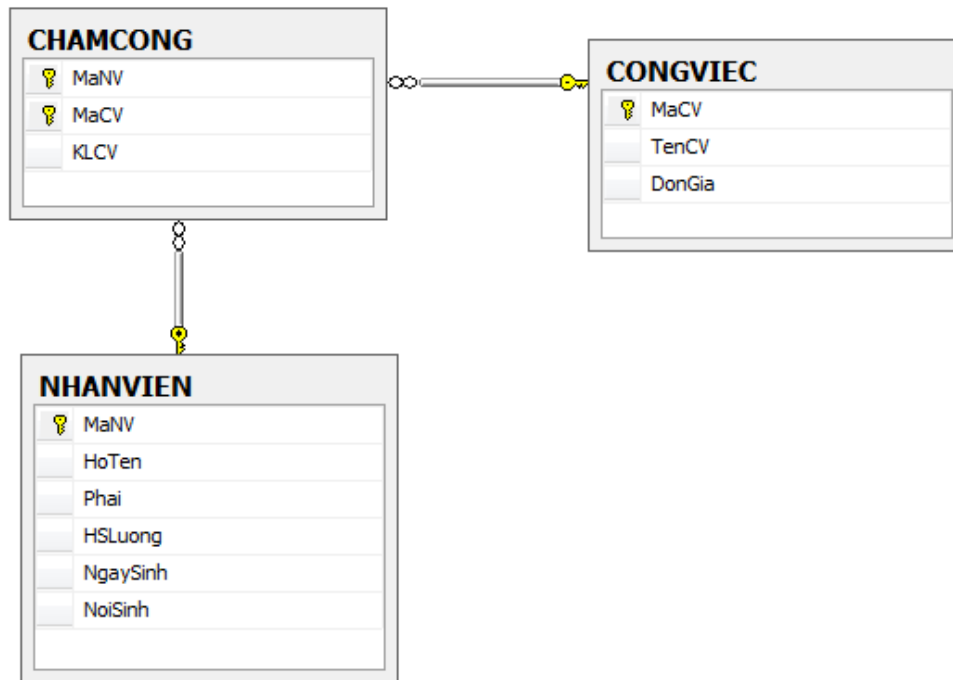
Sự kiện Click của nút Hiển thị: DSNhanVien()

```
Public Sub DSNhanVien()  
    MoKetNoiCSDL()  
    Dim sql As String  
    Dim ds As New DataSet  
    sql = "select *from NhanVien"  
    Dim da As New SqlDataAdapter(sql, con)  
    da.Fill(ds)  
    lstNhanVien.DataSource = ds.Tables(0)  
    lstNhanVien.DisplayMember = "HoTen"  
    lstNhanVien.ValueMember = "MaNV"  
    DongKetNoiCSDL()  
End Sub
```

Với thủ tục MoKetNoiCSDL() thực hiện mở kết nối CSDL, DongKetNoiCSDL(): thực hiện đóng kết nối CSDL.

BÀI TẬP

Sử dụng VB.Net hoặc C# để tạo CSDL và quan hệ giữa các bảng trong DataSet như hình bên dưới và thực hiện các yêu cầu sau:



MaCV	TenCV	DonGia
BV	Bảo vệ	5000
CM	Cạo mủ	3000
HC	Hành chính	7000
QL	Quản lý	10000
TC	Chăm sóc cây	3000

MaNV	HoTen	Phai	HSLuong	NgaySinh	NoiSinh
A001	Trần Thị Nhung	False	2.000	12/12/1982 12:00:00	Tiền Giang
A002	Hồ Thị Lan	False	1.000	1/12/1980 12:00:00	Bến Tre
A003	Nguyễn Lan Chi	False	1.300	12/2/1981 12:00:00	Vĩnh Long
A004	Trần Thanh Sang	True	1.500	5/6/1982 12:00:00	Tiền Giang
A005	Lê Xuân Thu	True	1.200	12/12/1982 12:00:00	Trà Vinh

MaNV	MaCV	KLCV
A001	BV	10
A001	CM	20
A002	CM	15
A002	QL	2
A004	BV	10
A004	QL	3
A005	HC	10

- Thiết kế Form và hiển thị thông tin như hình bên dưới:

MaNV	Họ tên	Phái	HS lương	Ngày sinh	Nơi sinh
A01	Trần Thị Nhung	Nữ	2	12/12/1982	Tiền Giang
A02	Hồ Thị Lan	Nữ	1	23/11/1980	Long An
A04	Trần Thanh Sang	Nam	1.5	02/02/1978	An Giang
A05	Lâm Xuân Thu	Nam	1.2	03/05/1980	Quản Trị
A03	Nguyễn Lan Chi	Nữ	1.3	30/12/1979	TP.HCM

- Khi nhập các thông tin MaNV, Họ tên, Phái, HS Lương, Ngày sinh, Nơi sinh và nhấp chọn Thêm thì thông tin của nhân viên này sẽ được lưu vào bảng NhanVien.
- Khi chọn 1 nhân viên trên ListView và chọn nút Xóa thì thông tin của nhân viên sẽ được xóa khỏi bảng NhanVien (có xác nhận trước khi xóa).
- Khi người sử dụng chỉnh sửa các thông tin Họ tên, Phái, HS Lương, Ngày sinh, Nơi sinh và nhấp chọn nút Sửa thì thông tin chi tiết của NhanVien sẽ được cập nhật.

Chương 3. XÂY DỰNG ỨNG DỤNG

Mục tiêu:

Sau khi học xong chương này, sinh viên có thể:

1. Kiến thức

+ Trình bày được các thuộc tính, phương thức, các sự kiện của các điều khiển hiển thị dữ liệu.

+ Xác định liên kết dữ liệu qua màn hình đơn, màn hình đơn 2 trang, màn hình một – nhiều, màn hình một – nhiều – nhiều.

+ Xây dựng cách liên kết dữ liệu qua DataBindings, DataGrid.

+ Phân loại các mã lệnh hiển thị dữ liệu ra các điều khiển; đồng bộ hóa dữ liệu giữa các điều khiển; thêm, xóa, cập nhật CSDL.

2. Kỹ năng

+ Vận dụng các kiến thức trên để thiết kế và lập trình được chương trình theo mô hình nhiều lớp.

3. Thái độ

+ Tự tin thiết kế ứng dụng kết nối CSDL SQL Server.

1. Hiển thị và thao tác dữ liệu

1.1. Các điều khiển hiển thị dữ liệu

1.1.1. ComboBox, ListBox, CheckListBox

Bảng 3.1. Các thuộc tính sau cần quan tâm khi ComboBox, ListBox, CheckListBox kết nối với dữ liệu.

Thuộc tính	Ý nghĩa
DataSource	Chỉ ra nguồn dữ liệu để lấy giá trị liệt kê chọn lựa.
DisplayMember	Cho biết tên cột trên nguồn dữ liệu liệt kê sẽ được hiển thị trên điều khiển.
ValueMember	Cho biết tên cột trên nguồn dữ liệu liệt kê sẽ được lấy giá trị để cập nhật vào nguồn dữ liệu khi chọn một dòng trên điều khiển.
SelectValue	Giá trị của dòng được chọn ứng với cột có tên trùng với ValueMember.

1.1.2. ListView

Bảng 3.2. Các thuộc tính chính của ListView.

Thuộc tính	Ý nghĩa
Items	Danh sách dữ liệu hiển thị trên ListView.
SelectedItems	Trả về danh sách các dòng dữ liệu đang được chọn trên ListView
MultiSelect	Cho phép chọn cùng một lúc nhiều dòng dữ liệu.
View	Thay đổi chế độ xem: LargeIcon, Details, SmallIcon, List Title.
Column	Thêm các dòng dữ liệu vào ListView
CheckBoxes	Có hiển thị CheckBox hay không.
GridLines	Có hiển thị đường kẻ dòng cột khi View là Details hay không.

Để thêm dữ liệu 1 dòng 3 cột:

```
Dòng thứ 1:  
Listview.Items.Add("STT")  
Listview.Items(0).SubItems.Add("Nguyễn Văn A")  
Listview.Items(0).SubItems.Add("12/10/2000")  
Dòng thứ 2:  
Listview.Items.Add("STT")  
Listview.Items(1).SubItems.Add("Nguyễn Văn B")  
Listview.Items(1).SubItems.Add("12/10/2000")
```

...
Để lấy dữ liệu 1 dòng đang chọn:

```
textbox1.Text = Listview.SelectedItems(0).SubItems(0).Text  
textbox2.Text = Listview.SelectedItems(0).SubItems(1).Text  
textbox3.Text = Listview.SelectedItems(0).SubItems(2).Text
```

1.1.3. TreeView

- Thuộc tính thường sử dụng:

Bảng 3.3. Mô tả ý nghĩa các thuộc tính thường sử dụng của TreeView

Thuộc tính	Ý nghĩa
BorderStyle	Chọn dạng viền cho TreeView
CheckBox	Có hiển thị CheckBox để chọn hay không
SelectionNode	Trả về hoặc gán nút được chọn
ImageList	Danh sách hình hiển thị trên Image List
ImageIndex	Hình hiển thị trên các nút của TreeView
SelectImageIndex	Hình hiển thị khi nút được chọn

- **Nodes**: tập hợp các nút thuộc cây, để thêm nút vào cây:

- **Bước 1**: định nghĩa và khởi tạo cho nút gốc.

Cú pháp:

```
Dim Nut_Goc As TreeNode
Nut_Goc = New TreeNode
Nut_Goc.Text = "chuỗi hiển thị"
Nut_Goc.tag = "chuỗi trả về"
```

Đưa Nut_Goc vào cây:

```
Ten_TreeView.Nodes.Add(Nut_Goc)
```

- **Bước 2**: định nghĩa và khởi tạo giá trị cho nút con:

Cú pháp:

```
Dim Nut_Con As TreeNode
Nut_Con = New TreeNode
Nut_Con.Text = "chuỗi hiển thị"
Nut_Con.tag = "chuỗi trả về"
```

Đưa nút con vào nút gốc:

```
Nut_Goc.Nodes.Add(Nut_Con)
```

Tương tự cho nút Cháu.

- Các phương thức:

- ExpandAll: hiển thị tất cả các thành phần của cây
- CollapseAll: thu gọn cây lại chỉ còn nút gốc

- Các sự kiện:

AfterSelect: xảy ra khi nút được chọn trên cây thay đổi, trong sự kiện này có thể truy cập vào thuộc tính: **Ten_TreeView.SelectNode.Text/.Tag** để biết nút được chọn.

1.1.4. DataGridView

DataGridView là điều khiển cho phép hiển thị dữ liệu dạng bảng gồm các dòng và cột. DataGridView trong .NET có thể liên kết với DataSet chứa nhiều bảng, quan hệ và có thể hiển thị từng bảng dữ liệu theo từng quan hệ trên lưới như cây thư mục, chọn bảng nào lưới sẽ chuyển sang hiển thị dữ liệu của bảng đó và xuất hiện các nút di chuyển về bảng trước.

Bảng 3.4. Các thuộc tính chính trên DataGridView.

Thuộc tính	Ý nghĩa
AllowNavigation	Cho biết có thể chuyển đổi từ cấp này sang cấp khác khi liên kết với DataSet.
CaptionFont	Kiểu chữ cho phần tiêu đề lưới.
CaptionText	Nội dung tiêu đề lưới.
CaptionVisible	Cho biết có thể hiển thị tiêu đề lưới hay không.
ContextMenu	Thực đơn ngữ cảnh cho điều khiển.
Controls	Trả về tập hợp các điều khiển trên lưới.
CurrentCell	Ô hiện hành trên lưới, không sử dụng được ở chế độ thiết kế.
CurrentRowIndex	Chỉ số dòng hiện hành.
DataBindings	Tập hợp các Binding của lưới.
DataMember	Thành phần nhánh dữ liệu hiển thị trên lưới.
DataSource	Nguồn dữ liệu của lưới.
Item	Trị của ô có chỉ số truyền vào. (Item(<dòng>, <cột>).
Parent	Đối tượng chứa điều khiển.
PreferredColumnWidth	Độ rộng mỗi cột trên lưới theo đơn vị Pixels.
PreferredRowHeight	Chiều cao mỗi cột trên lưới.
Text	Nội dung trên ô hiện hành.
Collapse	Làm ẩn đi do co lại các dòng dữ liệu nhánh con tương ứng với dòng truyền vào. Nếu truyền vào -1 sẽ

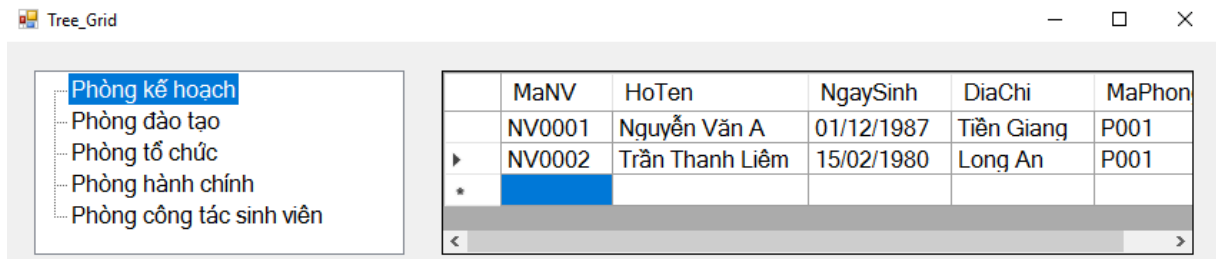
Thuộc tính	Ý nghĩa
	làm ẩn đi nhánh con của tất cả các dòng. Collapse(<chỉ số>).
Contains	True/False cho biết điều khiển truyền vào có thuộc trong các điều khiển con của lưới không. Contains(<điều khiển>).
Expand	Làm hiện ra co giãn các dòng dữ liệu nhánh con tương ứng với dòng truyền vào. Nếu truyền vào -1 sẽ hiện ra nhánh con của tất cả các dòng.
FindForm	Truy xuất Form đang chứa điều khiển lưới.
IsExpanded	Cho biết dòng truyền vào đang ở trạng thái Expand hay Collapsed. IsExpanded(<chỉ số dòng>).
IsSelected	Cho biết dòng truyền vào có được chọn hay không. IsSelected(<chỉ số dòng>).
SetDataBinding	Liên kết dữ liệu với lưới thông qua DataSource và DataMember. SetDataBinding(<datasource>, <datamember>).
UnSelect	Bỏ chọn dòng chỉ ra qua chỉ số truyền vào. UnSelect(<chỉ số dòng>).

Bảng 3.5.Các sự kiện chính của DataGrid.

Sự kiện	Ý nghĩa
CurrentCellChanged	Sự kiện xảy ra khi ô hiện hành thay đổi.
DataSourceChanged	Sự kiện xảy ra khi DataSource lưới thay đổi.
Enter	Sự kiện xảy ra khi lưới chuẩn bị nhận Focus.
Leave	Sự kiện xảy ra khi Focus chuẩn bị rời khỏi lưới
TextChanged	Sự kiện xảy ra khi thuộc tính Text của lưới thay đổi.

Khi liên kết dữ liệu cần quan tâm đến thuộc tính: DataSource.

Ví dụ 1: hiển thị dữ liệu trên TreeView và DataGridView như hình bên dưới:



Hình 3.1. Minh họa hiển thị dữ liệu bằng TreeView và DataGridView.

Load dữ liệu lên Tree_PB, trong sự kiện Form_Load của Form Tree_Grid:
Tree_PhongBan(Me.TreePB)

Thủ tục Tree_PhongBan:

```
Public Sub Tree_PhongBan(ByVal tree As TreeView)
    Dim dt As New DataTable
    dt = ThôngTinPhongBan()
    Dim sl As Integer = dt.Rows.Count
    For i = 0 To sl - 1
        Dim node As New TreeNode
        node.Text = dt.Rows(i)("TenPhong")
        node.Tag = dt.Rows(i)("MaPhong")
        tree.Nodes.Add(node)
    Next
End Sub
```

Hàm ThôngTinPhongBan()

```
Dim kn As New KetNoiCSDL
Public Function ThôngTinPhongBan() As DataTable
    Dim sql As String
    Dim dt As New DataTable
    sql = "select * from PhongBan"
    dt = kn.ExceQuery(sql)
    Return dt
End Function
```

Để Load dữ liệu lên DataGridView theo từng phòng trên Tree_PB, bắt sự kiện TreePB_AfterSelect

```
Dim dt As DataTable
If TreePB.SelectedNode.Parent Is Nothing Then
    dt = DS_NhanVien_theoPhong(TreePB.SelectedNode.Tag)
    dtGridView.DataSource = dt
End If
```

Với DS_NhanVien_theoPhong:

```
Public Function DS_NhanVien_theoPhong(ByVal maphong As String)
    As DataTable
    Dim sql As String
    Dim dt As New DataTable
    sql = "select * from NhanVien where MaPhong='" & maphong & "'"
    dt = kn.ExceQuery(sql)
    Return dt
End Function
```

1.2. Liên kết dữ liệu qua DataBindings

Có thể liên kết một thuộc tính của điều khiển với một thuộc tính, giá trị của một đối tượng nói chung thông qua đối tượng Binding.

Mỗi điều khiển có một tập hợp DataBindings cho phép điều khiển liên kết nhiều thuộc tính của nó với các giá trị khác nhau thông qua Binding.

Binding thuộc tên miền: System.Windows.Form.Binding.

Cú pháp khai báo:

New Binding(<tên thuộc tính trên điều khiển>, <đối tượng nguồn>, <thuộc tính trên nguồn>)

– <tên thuộc tính điều khiển>: tên thuộc tính trên điều khiển muốn liên kết với đối tượng nguồn.

– <đối tượng nguồn>: đối tượng cung cấp giá trị để liên kết có thể là các đối tượng DataSet, DataTable, DataView, DataViewManager, bất kỳ Class nào.

– <thuộc tính trên nguồn>: thuộc tính trên đối tượng nguồn có giá trị liên kết với điều khiển.

Ví dụ 2: liên kết nội dung của Label1 với nội dung của TextBox1

Dim bd As New Binding("Text", TextBox1, "Text").

Label1.DataBindings.Add(bd).

Bảng 3.6. Các thuộc tính chính của Binding

Thuộc tính	Ý nghĩa
BindingManagerBase	Trả về đối tượng BindingManagerBase quản lý Binding này, quản lý các Binding cùng liên kết đến một đối tượng nguồn và thuộc tính.
BindingMemberInfo	Trả về đối tượng chứa thông tin về Binding dựa trên tham số <tên thuộc tính trên điều khiển>.
Control	Trả về điều khiển có Binding này.
DataSource	Trả về đối tượng nguồn của Binding này.
IsBinding	Trả về giá trị cho biết Binding đang ở tình trạng hoạt động hay không.
PropertyName	Tên thuộc tính của điều khiển.

Bảng 3.7. Các sự kiện chính của Binding

Sự kiện	Ý nghĩa
Format	<ul style="list-style-type: none"> - Sự kiện xảy ra khi dữ liệu được đưa từ nguồn vào điều khiển và khi dữ liệu được lấy từ điều khiển về nguồn. Binding dùng sự kiện này để định dạng dữ liệu và hiển thị trên điều khiển. Khi dữ liệu chuyển từ điều khiển về nguồn, Binding sẽ kiểm tra dữ liệu thông qua sự kiện Parse rồi định dạng và hiển thị trên điều khiển. - Sự kiện Format xảy ra khi giá trị hiện hành của BindingmanagerBase thay đổi khi: lần đầu tiên liên kết, vị trí hiện hành (Position của BindingManagerBase) thay đổi, khi danh sách dữ liệu được sắp xếp, lọc.
Parse	Xảy ra khi giá trị của điều khiển liên kết thay đổi nhằm chuyển đổi dữ liệu trước khi chuyển về nguồn dữ liệu. Sự kiện xảy ra: sau khi sự kiện Validated của điều khiển xảy ra, phương thức EndCurrentEdit của BindingManager được gọi, khi vị trí hiện hành của BindingManagerBase thay đổi.

Bảng 3.8. Thuộc tính dùng liên kết dữ liệu của các điều khiển

Điều khiển	Thuộc tính
Label	Text
TextBox	Text
CheckBox	Checked
RadioButton	Checked
ComboBox	SelectedValue
ListBox	SelectValue
CheckListBox	SelectValue
DateTimePicker	Value
NumericUpDown	Value

Tập hợp DataBindings của các điều khiển:

Bảng 3.9. Các thuộc tính của DataBinding

Thuộc tính	Ý nghĩa
Control	Trả về điều khiển có tập hợp DataBinding.
Count	Trả về số Binding trong tập hợp.
IsReadOnly	Trả về giá trị cho biết tập hợp có thuộc tính chỉ đọc hay không.
Item	Tham chiếu đến một Binding qua tham số truyền vào (chỉ số hoặc tên).
RemoveBinding	Trả về mảng các tên Binding đã loại bỏ khỏi tập hợp.

- Thêm một Binding vào tập hợp:
 - + Add(<đối tượng Binding>).
 - + Add(<tên thuộc tính trên điều khiển>, <đối tượng nguồn>, <thuộc tính trên nguồn>)
- Loại bỏ Binding khỏi tập hợp:

- + Remove(<đối tượng binding>): loại bỏ Binding truyền vào.
- + RemoveAt(<chỉ số>): loại bỏ Binding có chỉ số là <chỉ số>.
- + Clear(): xóa hết các Binding.

Ví dụ 3: thiết kế Form như hình bên dưới có 4 nút di chuyển các mẫu tin: về đầu, về trước, về sau, về cuối. Khi nhấp chuột vào các nút di chuyển sẽ hiển thị thông tin chi tiết của lớp trên các TextBox đồng thời thể hiện số mẫu tin hiện hành/tổng số mẫu tin.

Hình 3.2. Minh họa sử dụng DataBinding để di chuyển các mẫu tin.

Với dữ liệu bảng LOP:

MaLop	TenLop	NienKhoa	MaKhoa	GVCN
CDCKA	Cao đẳng cơ khí A	2000-2003	CK	Lê Thành Nhân
CDCKB	Cao đẳng cơ khí B	2000-2003	CK	Lê Thành Nhân
CDDTA	Cao đẳng điện tử A	2001-2004	DT	Phùng Đăng Khoa
CDTHA	Cao đẳng tin học A	2000-2003	CNTT	Phạm Ngọc Giàu
CDTHB	Cao đẳng tin học B	2000-2003	CNTT	Phạm Ngọc Giàu
TA1	Tiếng Anh 1	2000-2003	CNTT	Phạm Ngọc Giàu
TA2	Tiếng Anh 2	2000-2003	CK	Trần Bình Trọng
THA	Tin học Cơ bản	2000-2003	CK	Nguyễn Văn Anh

Hình 3.3. Dữ liệu ban đầu của bảng LOP.

Sự kiện Form_Load:

```
dt = TTLop()
mbm = BindingContext(dt)
```

Với:

```
Dim kn As New KetNoiCSDL
Dim mbm As BindingManagerBase
```

```

Dim dt As New DataTable
Public Function TTLop() As DataTable
    Dim sql As String
    sql = "select * from Lop"
    dt = kn.ExceQuery(sql)
    Return dt
End Function

```

Sự kiện Click nút << (về đầu):

```

mbm.Position = 0
HienThiSoMauTin(dt)

```

Sự kiện Click nút < (về trước):

```

If mbm.Position > 0 Then
    mbm.Position = -1
Else
    mbm.Position = mbm.Count - 1
End If
HienThiSoMauTin(dt)

```

Sự kiện Click nút > (về sau):

```

If mbm.Position >= mbm.Count - 1 Then
    mbm.Position = 0
Else
    mbm.Position += 1
End If
HienThiSoMauTin(dt)

```

Sự kiện Click nút >> (về cuối):

```

mbm.Position = mbm.Count - 1
HienThiSoMauTin(dt)

Private Sub HienThiSoMauTin(ByVal dt As DataTable)
    Dim tsdong, donghientai As Integer
    tsdong = dt.Rows.Count
    If tsdong = 0 Then
        txtMauTin.Text = "0"
    Else
        donghientai = mbm.Position + 1
        txtMauTin.Text = donghientai & "/" & tsdong
        txtMaLop.Text = dt.Rows(donghientai - 1)("MaLop")
        txtTenLop.Text = dt.Rows(donghientai - 1)("TenLop")
        txtMaKhoa.Text = dt.Rows(donghientai - 1)("MaKhoa")
        txtGVCN.Text = dt.Rows(donghientai - 1)("GVCN")
    End If
End Sub

```

2. Xây dựng ứng dụng

2.1. Xây dựng ứng dụng theo mô hình 3 lớp

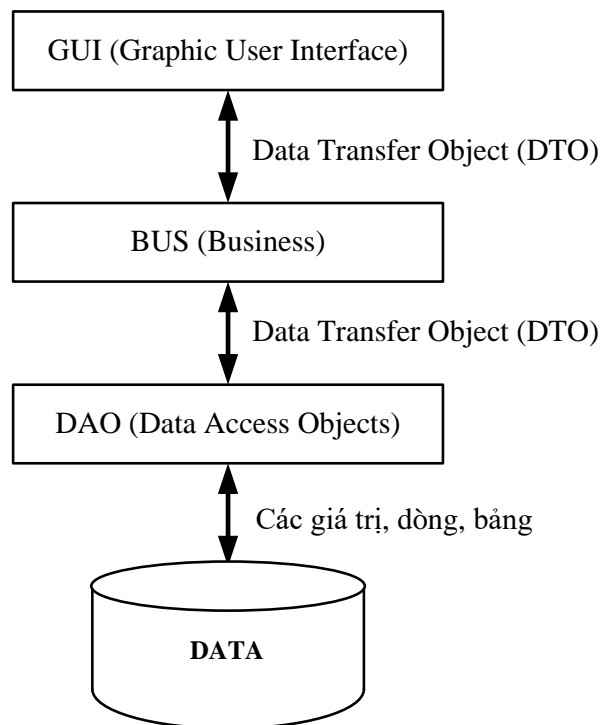
Ứng dụng đa tầng là ứng dụng mà trong đó mỗi chức năng của ứng dụng có thể được viết mã lệnh ở các thành phần riêng biệt và các thành phần này có thể

được lưu trữ và thực thi trên các máy tính khác nhau. Một trong những hướng tiếp cận phổ biến này chính là mô hình 3 lớp.

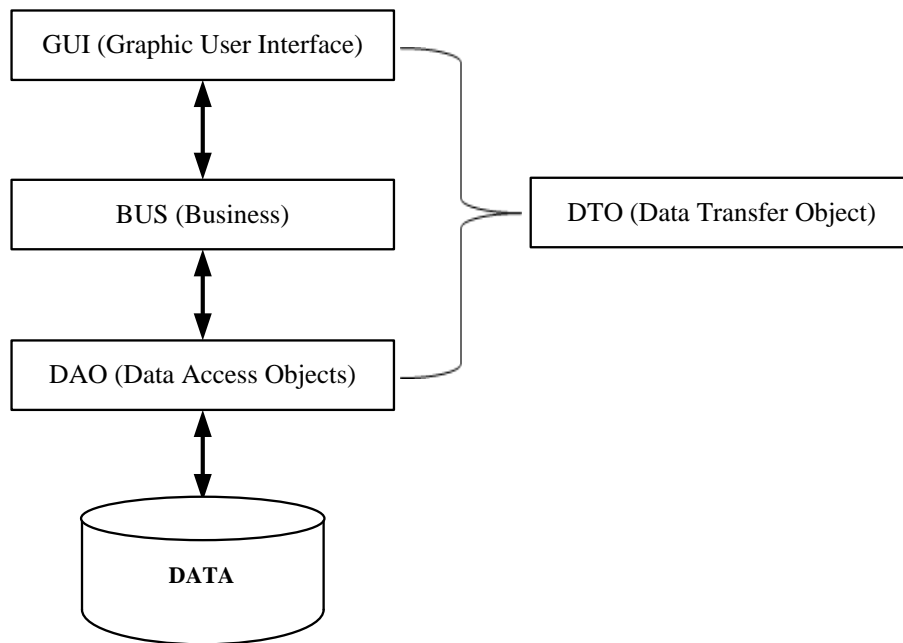
Bảng 3.10. Bảng mô tả chức năng các tầng.

Tầng thiết kế	Tầng nghiệp vụ	Tầng dữ liệu
Giao diện - Form, control, menu	Các nghiệp vụ: kiểm tra, tính toán, quy tắc kinh doanh, các quy luật.	Thẻ hiện dữ liệu - Lưu trữ dữ liệu

2.2. Xây dựng mô hình trên Solutions



Hình 3.4. Việc trao đổi, liên lạc giữa các lớp.



Hình 3.5. Sự phụ thuộc của các tầng.

- Tầng DTO (Data Transfer Object): tầng DTO đại diện cho các đối tượng dữ liệu được truyền đi giữa các lớp.
- Tầng DAO (Data Access Objects): thực hiện các phương thức truy cập dữ liệu như là: tìm kiếm, thêm, xóa, cập nhật, xử lý,... trên CSDL.
- Tầng BUS: kiểm tra các yêu cầu nghiệp vụ khi truy cập CSDL.
- Tầng GUI (Graphic User Interface): nhập liệu và trình bày dữ liệu.

Ưu điểm:

- + Phân tách các xử lý giao diện, nghiệp vụ, và lưu trữ làm giảm sự kết dính giữa các đối tượng phần mềm.
- + Dễ theo dõi, sửa chữa.
- + Chia sẻ trách nhiệm.
- + Tái sử dụng.

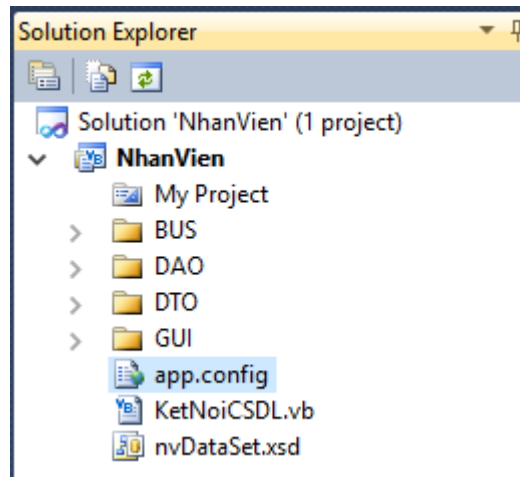
Khuyết điểm:

- + Kém linh hoạt: khi có sự thay đổi ở bất cứ xử lý nào, đều phải biên dịch, triển khai lại toàn bộ ứng dụng.

2.3. Ứng dụng minh họa trên Solution

2.3.1. Tổ chức cây thư mục

Việc tổ chức cây thư mục sẽ giúp quản lý các tập tin trong ứng dụng được thuận lợi hơn.



Hình 3.6. Minh họa tổ chức cây thư mục.

2.3.2. Triển khai ứng dụng minh họa

– CSDL quản lý nhân viên, xây dựng ứng dụng kết nối CSDL cơ bản với các chức năng: thêm, xóa, sửa, dữ liệu.

NhanVien(**MaNV**, HoTen, NgaySinh, DiaChi, MaPhong)

PhongBan(**MaPhong**, TenPhong)

– Tạo Class kết nối CSDL

Thiết kế Class KetNoiCSDL có chứa các hàm và thủ tục: mở kết nối CSDL, đóng kết nối CSDL, thực thi câu truy vấn Select, thực thi câu truy vấn Update, Insert, Delete:

```
Imports System.Data
Imports System.Data.SqlClient
Public Class KetNoiCSDL
    'Mở kết nối csdl
    Private con As New SqlConnection
    Public Sub MoKetNoiCSDL()
        Try
            'Câu lệnh kết nối CSDL
            Dim sqlcon As String = "Data Source=PLINH-PC;Initial
Catalog=Nv;Integrated Security=True"
            con.ConnectionString = sqlcon
            If con.State = ConnectionState.Closed Then
                con.Open()
            End If
            Catch ex As Exception
                MsgBox("Kết nối csdl không thành công!")
            End Try
        End Sub
        'Đóng kết nối csdl
        Public Sub DongKetNoiCSDL()
            If con.State = ConnectionState.Open Then
                con.Close()
            End Sub
        End Sub
    End Class
```

```

        End If
    End Sub
    'Dùng trong câu truy vấn Select
    Public Function ExceQuery(ByVal sql As String) As DataTable
        MoKetNoiCSDL()
        Dim dt As New DataTable
        Dim cd As New SqlCommand(sql, con)
        Dim dr As SqlDataReader = cd.ExecuteReader()
        dt.Load(dr)
        DongKetNoiCSDL()
        Return dt
    End Function
    'Dùng trong câu truy vấn: Insert, Update, Delete
    Public Sub ExceNonQuery(ByVal sql As String)
        MoKetNoiCSDL()
        Dim cmd As New SqlCommand(sql, con)
        cmd.ExecuteNonQuery()
        DongKetNoiCSDL()
    End Sub
End Class

```

2.3.2.1. Tầng DTO (Data Transfer Object)

Lớp DTO đại diện cho các đối tượng dữ liệu được truyền đi giữa các lớp. Thông thường với mỗi bảng dữ liệu sẽ tạo một lớp DTO tương ứng.

```

Public Class PhongBanDTO
    Public _maphong As String
    Public _tenphong As String

    Public Property MaPhong() As String
        Get
            Return _maphong
        End Get
        Set(ByVal value As String)
            _maphong = value
        End Set
    End Property

    Public Property TenPhong() As String
        Get
            Return _tenphong
        End Get
        Set(ByVal value As String)
            _tenphong = value
        End Set
    End Property
End Class

```

2.3.2.2. Tầng DAO (Data Access Objects)

Các lớp DAO sẽ thực hiện các yêu cầu truy vấn trên CSDL của các lớp nghiệp vụ (BUS). Thông thường với mỗi bảng dữ liệu sẽ tạo một lớp DAO tương ứng với các phương thức truy cập dữ liệu như là: tìm kiếm, thêm, xóa, cập nhật, xử lý,... trên CSDL.

Trong lớp DAO do sử dụng các lớp DTO, các hàm và thủ tục của Class KetNoiCSDL do đó phải khai báo sử dụng các Namespace của các lớp DTO:

```
Dim pb As New PhongBanDTO
Dim csdl As New KetNoiCSDL

Public Class PhongBanDAO
    Dim pb As New PhongBanDTO
    Dim csdl As New KetNoiCSDL
    Public Function ThôngTinPhongBan() As DataTable
        Dim sql As String
        Dim dt As New DataTable
        sql = "select * from PhongBan"
        dt = csdl.ExceQuery(sql)
        Return dt
    End Function

    Public Sub ThemPhongBan(ByVal pb As PhongBanDTO)
        Try
            Dim sql As String
            sql = "INSERT INTO PhongBan([MaPhong],
[TenPhong])VALUES('" & pb.MaPhong & "',N'" & pb.TenPhong & "')"
            csdl.ExceNonQuery(sql)
        Catch ex As Exception
            MsgBox("Thêm phòng ban không thành công!")
        End Try
    End Sub

    Public Sub XoaPhongBan(ByVal pb As PhongBanDTO)
        Try
            Dim sql As String
            sql = "DELETE FROM PhongBan WHERE MaPhong='" &
pb.MaPhong & "'"
            csdl.ExceNonQuery(sql)
        Catch ex As Exception
            MsgBox("Xóa phòng ban không thành công!")
        End Try
    End Sub

    Public Sub CapNhatPhongBan(ByVal pb As PhongBanDTO)
        Try
            Dim sql As String
            sql = "UPDATE PhongBan SET TenPhong='" & pb.TenPhong &
"' WHERE MaPhong='" & pb.MaPhong & "'"
            csdl.ExceNonQuery(sql)
        End Try
    End Sub
End Class
```

```

        Catch ex As Exception
            MsgBox("Cập nhật phòng ban không thành công!")
        End Try
    End Sub
End Class

```

2.3.2.3. Tầng BUS (Business)

Các lớp BUS là các lớp xử lý nghiệp vụ thực hiện các yêu cầu từ lớp giao diện, thực hiện một số kiểm tra các yêu cầu nghiệp vụ và gọi các lớp DAO thực hiện truy xuất dữ liệu để trả cho các lớp giao diện.

```

Public Class PhongBanBUS
    Dim pb As New PhongBanDAO

    Public Function TTPhongBan() As DataTable
        Dim dt As New DataTable
        dt = pb.ThongTinPhongBan()
        Return dt
    End Function

    Public Function ThemPhongBan(ByVal Phong As PhongBanDTO) As Integer
        Dim kq = 0
        Try
            pb.ThemPhongBan(Phong)
            kq = 1
        Catch ex As Exception
            MsgBox("Thêm dữ liệu phòng ban không thành công!")
        End Try
        Return kq
    End Function

    Public Function XoaPhongBan(ByVal Phong As PhongBanDTO) As Integer
        Dim kq = 0
        Try
            pb.XoaPhongBan(Phong)
            kq = 1
        Catch ex As Exception
            MsgBox("Xóa dữ liệu phòng ban không thành công!")
        End Try
        Return kq
    End Function

    Public Function CapNhatPhongBan(ByVal Phong As PhongBanDTO) As Integer
        Dim kq = 0
        Try
            pb.CapNhatPhongBan(Phong)
            kq = 1
        Catch ex As Exception
            MsgBox("Cập nhật dữ liệu phòng ban không thành công!")
        End Try
    End Function

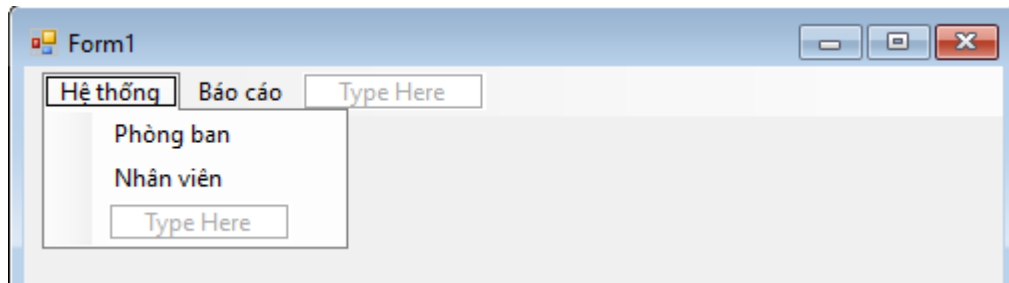
```

```
Return kq
End Function
```

2.3.2.4. Tầng GUI (Graphic User Interface)

Giao diện của ứng dụng. Các lớp giao diện sẽ gọi các lớp nghiệp vụ (BUS) để thực hiện các yêu cầu nghiệp vụ. Có nhiệm vụ nhập và trình bày dữ liệu, có thể bao gồm các bước kiểm tra dữ liệu trước khi gọi lớp BUS.

Giao diện Form chính:



Hình 3.7. Minh họa giao diện Form chính.

Giao diện Form Phòng ban:

Mã phòng	Tên phòng	
P001	Phòng kế hoạch	
P002	Phòng đào tạo	
P003	Phòng tổ chức	
P004	Phòng hành chính	
P005	Phòng công tác sinh viên	

Hình 3.8. Hiện thị thông tin các phòng ban.

```
Public Class PhongGUI
Dim pb As New PhongBanBUS
Dim phongban As New PhongBanDTO
Private Sub PhongGUI_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
    LoadDuLieu_ListView()
End Sub
Public Sub LoadDuLieu_ListView()
```

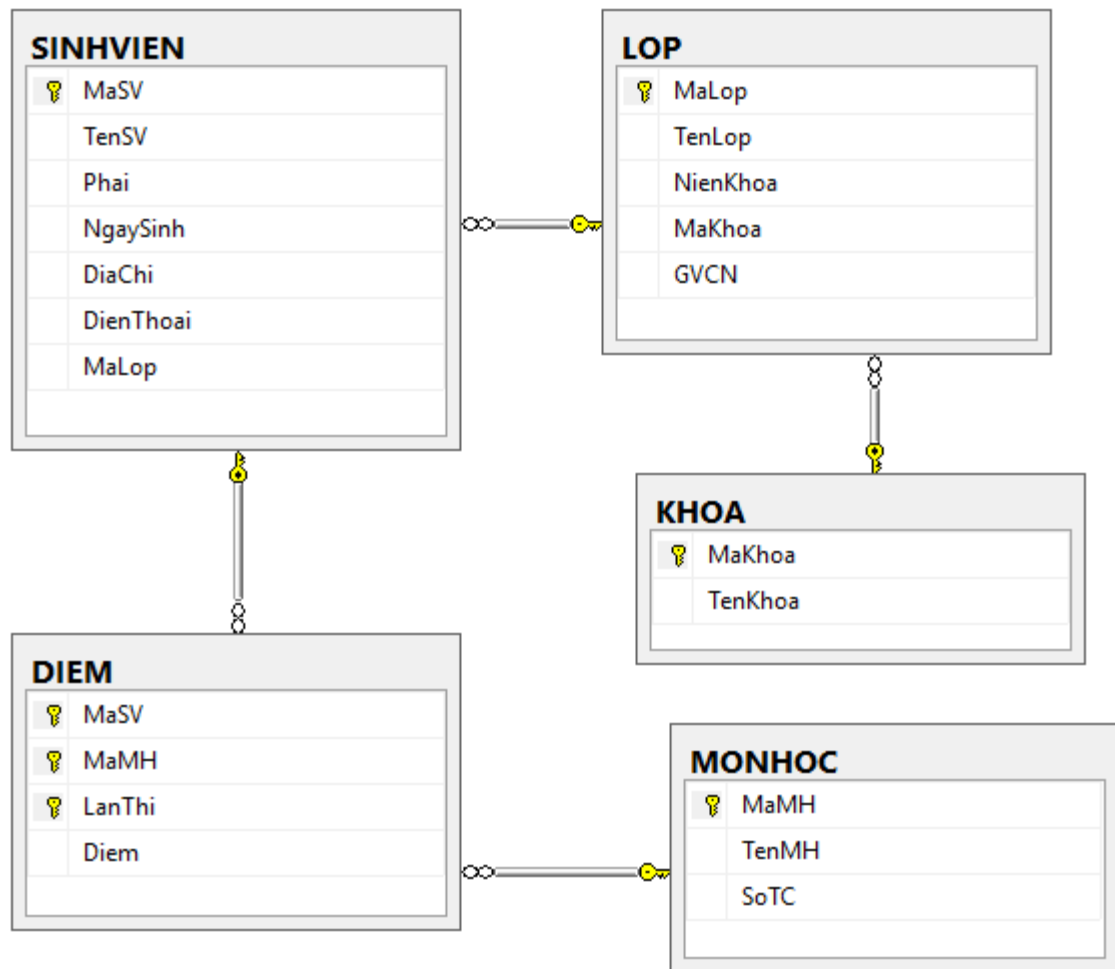
```

Me.ListView1.Items.Clear()
Dim dt As New DataTable
dt = pb.TTPhongBan
Dim sodong As Integer = dt.Rows.Count
Dim i As Integer
'Duyet tung dong lay gia tri gan vao ListView
For i = 0 To sodong - 1
    ListView1.Items.Add((dt.Rows(i)(0).ToString))
    ListView1.Items(i).SubItems.Add(dt.Rows(i)(1).ToString)
Next
End Sub
Private Sub bntThem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles bntThem.Click
    phongban.MaPhong = txtMaPhong.Text
    phongban.TenPhong = txtTenPhong.Text
    If pb.ThemPhongBan(phongban) = 1 Then
        MsgBox("Thêm phòng ban thành công!")
    End If
    LoadDuLieu_ListView()
End Sub
Private Sub bntXoa_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles bntXoa.Click
    phongban.MaPhong = txtMaPhong.Text
    phongban.TenPhong = txtTenPhong.Text
    If pb.XoaPhongBan(phongban) = 1 Then
        MsgBox("Xóa phòng ban thành công!")
    End If
    LoadDuLieu_ListView()
End Sub
Private Sub bntCapNhat_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles bntCapNhat.Click
    phongban.MaPhong = txtMaPhong.Text
    phongban.TenPhong = txtTenPhong.Text
    If pb.CapNhatPhongBan(phongban) = 1 Then
        MsgBox("Cập nhật phòng ban thành công!")
    End If
    LoadDuLieu_ListView()
End Sub
End Class

```


BÀI TẬP

Bài 1. Tạo CSDL quản lý sinh viên trong SQL Server với cấu trúc, quan hệ, và dữ liệu được minh họa như sau:



Dữ liệu mẫu:

DESKTOP-4SL47N2\...LQLSV - dbo.KHOA ×

	MaKhoa	TenKhoa
	CK	Cơ khí
	CNTT	Công nghệ thô...
	DT	Điện tử

DESKTOP-4SL47N2...SV - dbo.MONHOC ×

	MaMH	TenMH	SoTC
	csdl	Cơ So Du Lieu	4
	ctdl	Cau Truc du Lieu	4
	dth	Dien tu Hoa	3
	java	Lap trinh Java	5
	lsl	Lich Su Dang	2
	trr	Toan Roi rac	3
	vl2	Vat Ly 2	3

DESKTOP-4SL47N2\....DLQLSV - dbo.LOP ✕					
	MaLop	TenLop	NienKhoa	MaKhoa	GVCN
	CDCKA	Cao đẳng cơ khí A	2000-2003	CK	Lê Thành Nhân
	CDCKB	Cao đẳng cơ khí B	2000-2003	CK	Lê Thành Nhân
	CDDTA	Cao đẳng điện tử A	2001-2004	DT	Phùng Đăng Khoa
	CDTHA	Cao đẳng tin học A	2000-2003	CNTT	Phạm Ngọc Giàu
	CDTHB	Cao đẳng tin học B	2000-2003	CNTT	Phạm Ngọc Giàu
	TA1	Tiếng Anh 1	2000-2003	CNTT	Phạm Ngọc Giàu
	TA2	Tiếng Anh 2	2000-2003	CK	Trần Bình Trọng
	THA	Tin học Cơ bản	2000-2003	CK	Nguyễn Văn Anh

DESKTOP-4SL47N2\...SV - dbo.SINHVIEN ✕							
	MaSV	TenSV	Phai	NgaySinh	DiaChi	DienThoai	MaLop
	CKA01	Trần Văn An	True	1986-10-11 00:00:00.000	14 Dinh Tien Hoang Q1	07601210	CDCKA
	CKA02	Nguyễn Thị Thúy	False	1988-12-02 00:00:00.000	16 Hai Ba Trung Q2	07603453	CDCKA
	CKA03	Nguyễn Tu Nhân	True	1988-11-11 00:00:00.000	113 Tran Hung Dao Q1	07677765	CDCKA
	CKB01	Trần Trọng Nhân	True	1986-01-09 00:00:00.000	12 Hung Vuong Q1	07645675	CDCKB
	CKB02	Võ Hữu Nghĩa	True	1986-01-10 00:00:00.000	144 Cach mang Thang Tam Q2	07673455	CDCKB
	DTA02	Lê Bá Hải	True	1986-05-02 00:00:00.000	141 Hai Ba Trung Q2	07656475	CDDTA
	DTA03	Phạm Thị Dương	False	1986-08-05 00:00:00.000	374 Hai Ba Trung Q2	07655665	CDDTA
	DTA04	Lê Vĩnh Phúc	True	1986-02-02 00:00:00.000	123 Dinh Tien Hoang Q1	07345677	CDDTA
	DTA05	Phạm Văn Nhật	True	1986-01-02 00:00:00.000	13 Tran Hung Dao Q1	07675888	CDDTA
	THA01	Nguyễn Thanh ...	True	1986-07-03 00:00:00.000	455 Tran Hung Dao Q1	07625325	CDTHA
	THA02	Nguyễn Thị Hồ...	False	1986-08-08 00:00:00.000	64 Dinh Tien Hoang Q1	07677342	CDTHA
	THA03	Lê Đoàn Phương	True	1986-06-01 00:00:00.000	44 Cach mang Thang Tam Q2	07677543	CDTHA
	THA04	Lê Thanh Đoàn	True	1986-07-11 00:00:00.000	14 Dinh Tien Hoang Q1	07677453	CDTHA
	THB01	Lê Thanh Phong	True	1986-11-01 00:00:00.000	53 Dinh Tien Hoang Q1	07653532	CDTHB
	THB02	Nguyễn Phước...	False	1986-05-06 00:00:00.000	534 Cach mang Thang Tam Q2	07698978	CDTHB
	THB03	Trịnh Hữu Tín	True	1986-08-03 00:00:00.000	152 Hung Vuong Q1	07677687	CDTHB
	THB04	Đào Thị Trinh	False	1986-10-11 00:00:00.000	234 Hung Vuong Q1	07677068	CDTHB

	MaSV	MaMH	LanThi	Diem
	DTA04	lsd	2	8
▶	DTA05	dth	1	8
	DTA05	lsd	1	6
	DTA05	vl2	1	10
	THA01	csdl	1	9
	THA01	lsd	1	5
	THA02	csdl	1	7
	THA02	lsd	1	9
	THA02	trr	1	9
	THA03	csdl	1	10
	THA03	ctdl	1	8
	THA03	lsd	1	10
	THA03	trr	1	9
	THA04	csdl	1	5
	THA04	ctdl	1	2
	THA04	java	1	7
	THA04	trr	1	3
	THB01	ctdl	1	7
	THB01	java	1	2
	THB01	lsd	1	6
	THB01	trr	1	3
	THB02	csdl	1	2
	THB02	java	1	7
	THB02	lsd	2	8

	MaSV	MaMH	LanThi	Diem
▶	CKA01	dth	1	9
	CKA01	lsd	1	5
	CKA02	dth	1	7
	CKA02	lsd	1	9
	CKA02	trr	1	9
	CKA03	dth	1	10
	CKA03	lsd	1	10
	CKA03	trr	1	9
	CKA03	vl2	1	8
	CKB01	dth	1	5
	CKB01	lsd	1	7
	CKB01	trr	1	3
	CKB01	vl2	1	2
	CKB02	trr	1	3
	CKB02	vl2	1	2
	DTA02	dth	1	8
	DTA02	java	1	10
	DTA02	lsd	2	3
	DTA02	vl2	1	9
	DTA03	dth	1	7
	DTA03	lsd	1	6
	DTA04	dth	1	2

Thiết kế và hiển thị thông tin lên Form theo các mẫu sau:

- Hiển thị thông tin sinh viên theo từng lớp: ComboBox là tên tất cả các lớp học, khi chọn một lớp học trong ComboBox thì thông tin các sinh viên của lớp học này sẽ được hiển thị bên dưới DataGridView hoặc ListView.

Form1

Lớp:

Danh sách sinh viên của lớp Cao đẳng điện tử A

	MaSV	TenSV	Phai	NgaySinh	DiaChi	DienThoai	MaLop
	DTA02	Lê Bá Hải	<input checked="" type="checkbox"/>	02/05/1986	141 Hai Ba ...	07656475	CDDTA
	DTA03	Phạm Thị D...	<input type="checkbox"/>	05/08/1986	374 Hai Ba ...	07655665	CDDTA
	DTA04	Lê Vĩnh Phúc	<input checked="" type="checkbox"/>	02/02/1986	123 Dinh Tie...	07345677	CDDTA
	DTA05	Phạm Văn ...	<input checked="" type="checkbox"/>	02/01/1986	13 Tran Hun...	07675888	CDDTA
▶*			<input type="checkbox"/>				

b. Hiện thị thông tin các môn học trong ComboBox, khi chọn một môn học sẽ hiển thị số thứ tự sinh viên hiện hành/tổng số sinh viên của lớp học này ở Label (trong khung duyệt dữ liệu). Khi nhấp chuột vào các nút di chuyển thông tin sinh viên tương ứng sẽ hiển thị trên các TextBox: mã sinh viên, tên sinh viên, lần thi, điểm.

ThongTinSV_MonHoc

THÔNG TIN SINH VIÊN THEO MÔN HỌC

Tên môn học:

Xem thông tin sinh viên

Mã sinh viên:

Tên sinh viên:

Lần thi:

Điểm:

Duyệt dữ liệu

c. Hiển thị thông tin lớp học trong ComboBox, khi chọn một lớp học thì thông tin chi tiết của sinh viên được hiển thị trong DataGridView.

– Nhấp vào nút Thêm mã sinh viên sẽ tự động tăng, nhập: Tên SV, chọn lớp, Điện thoại, giới tính, ngày sinh, địa chỉ và nhấp nút Lưu thông tin sinh viên sẽ được thêm vào CSDL.

– Khi chọn 1 dòng trên DataGridView thông tin chi tiết của sinh viên sẽ được hiển thị trên các TextBox và ComboBox.

NhậpSVTheoLop

Nhập thông tin sinh viên theo lớp

Mã SV:

Tên SV:

Lớp:

Điện thoại: ☒ Nam/Nữ ☐

Ngày sinh: / /

Địa chỉ:

Thêm Xóa Lưu Thoát

	MaSV	TenSV	Phai	NgaySinh	DiaChi	DienThoai	MaLop
	DTA02	Lê Bá Hải	<input checked="" type="checkbox"/>	02/05/1986	141 Hai Ba Tru...	07656475	CDDTA
	DTA03	Phạm Thị Dương	<input type="checkbox"/>	05/08/1986	374 Hai Ba Tru...	07655665	CDDTA
▶	DTA04	Lê Vĩnh Phúc	<input checked="" type="checkbox"/>	02/02/1986	123 Đinh Tiên ...	07345677	CDDTA
	DTA05	Phạm Văn Nhật	<input checked="" type="checkbox"/>	02/01/1986	13 Trần Hưng ...	07675888	CDDTA
*			<input type="checkbox"/>				

d. Thiết kế form hiển thị dạng Tree_ListView

Hiển thị dữ liệu dạng tree_ListView

Danh mục lớp

- Danh mục lớp
 - Cao đẳng cơ khí A
 - Cao đẳng cơ khí B
 - Cao đẳng điện tử A
 - Cao đẳng tin học A
 - Cao đẳng tin học B
 - Tiếng Anh 1
 - Tiếng Anh 2
 - Tin học Cơ bản

Thông tin sinh viên của từng lớp

MaSV	TenSV	Phai	NgàySinh	DiaChi	DienThoai
CKA01	Trần Văn An	True	11/10/1986	14 Dinh Tien Hoang Q1	07601210
CKA02	Nguyễn Thị Thủy	False	02/12/1988	16 Hai Ba Trung Q2	07603453
CKA03	Nguyễn Tu Nhân	True	11/11/1988	113 Tran Hung Dao Q1	07677765

e. Thiết kế Form có định dạng như sau:

HIỂN THỊ THÔNG TIN DẠNG TREEVIEW

Danh sách lớp_sinh viên

- Danh mục lớp
 - cdck1
 - Ma SV:ck04
 - Ma SV:dd
 - Ma SV:t008
 - cddta
 - cdtha
 - cdthb
 - cdcka

Mã SV: ck04 **Thêm** +

Họ tên: Nguyễn Tu Nhân **Xóa** -

Phái: Nam **Lưu**

Ngày sinh: 11/11/80 12:00:00 AM **IN**

Địa chỉ: 15 Nguyễn Trãi Q1

Điện thoại: 07685788

Mã lớp: cdck1

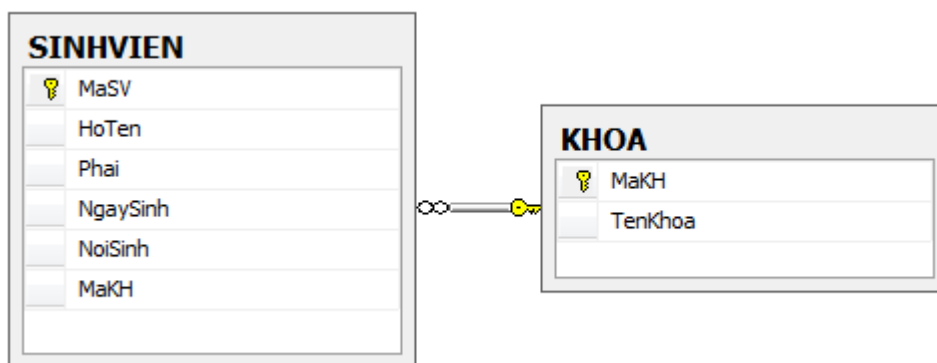
Danh sách sinh viên theo lớp

	MaLop	MaSV	TenSV	Phai	NgàyS
▶	cdck1	ck04	Nguyễn Tu Nhân	<input checked="" type="checkbox"/>	11/11/
	cdck1	dd	Minh Nhi	<input checked="" type="checkbox"/>	08/05/
	cdck1	t008	aaaaaa	<input type="checkbox"/>	03/11/
*				<input type="checkbox"/>	

Yêu cầu:

- Khi chọn nút gốc cây, hiển thị thông tin về lớp đang chọn trên dataGridview
- Chọn nút con hiển thị thông tin chi tiết về sinh viên trên các Textbox.
- Các nút lệnh
 - + **Thêm:** nhập thêm sinh viên mới. mã sinh viên tự động tăng theo lớp đang chọn trên **treeview**
 - + **Lưu:** giá trị vừa thêm sẽ cập nhật vào cây và lưới tương ứng.
 - + **Xóa:** Xóa thông tin về sinh viên, cập nhật dữ liệu cho cây và lưới.

Bài 2. Cho CSDL quản lý sinh viên của khoa như sau:



MaKH	TenKhoa
NN	Ngoại ngữ
SP	Sư phạm
TH	Tin học
VL	Vật lý

MaSV	HoTen	Phai	NgaySinh	NoiSinh	MaKH
SV01	Trần Văn Triều	True	1995-01-01 ...	Tiền Giang	NN
SV02	Nguyễn Thanh Mai	False	1995-10-25 ...	Long An	NN
SV03	Nguyễn Thanh Tuấn	True	1995-11-15 ...	Bến Tre	TH
SV04	Dương Công Định	True	1995-11-21 ...	Vĩnh Long	TH
SV05	Dương Đông Nhi	False	1995-01-02 ...	Tiền Giang	VL
SV06	Nguyễn Quỳnh Như	False	1995-09-10 ...	An Giang	SP
SV07	Nguyễn Thiên Kim	False	1995-03-01 ...	Cần Thơ	SP

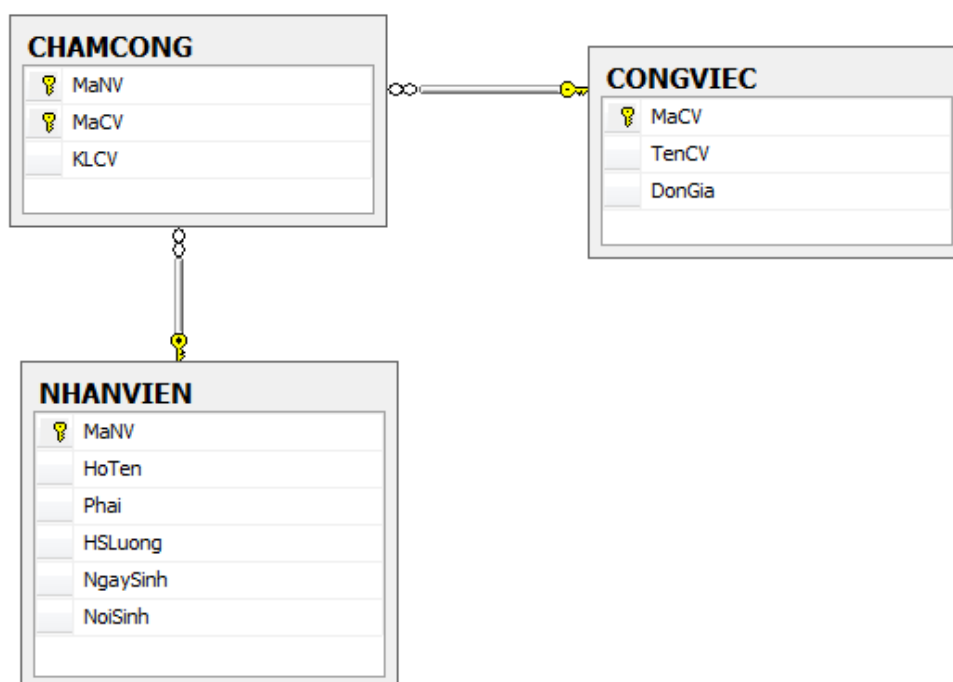
Thiết kế giao diện như sau:

MaSV	HoTen	Phai	NgaySinh	NoiSinh	MaKH
SV01	Trần Văn Triều	<input checked="" type="checkbox"/>	01/01/1995	Tiền Giang	NN
SV02	Nguyễn Thanh Mai	<input type="checkbox"/>	25/10/1995	Long An	NN
*		<input type="checkbox"/>			

Sử dụng mô hình 3 lớp thực hiện các yêu cầu sau:

- Tạo TreeView với nút cha là TenKhoa, nút con là danh sách MaSV thuộc khoa.
- Khi click vào nút cha trên TreeView (TenKhoa) sẽ hiển thị thông tin các sinh viên tương ứng thuộc khoa. Thông tin gồm: MaSV, HoTen, Phai, NgaySinh, NoiSinh, MaKH.
- Khi nhấn nút con trên TreeView (MaSV) thông tin chi tiết sẽ hiển thị tương ứng trên các textbox.
- Khi nhấn nút **Thêm**: nút **Ghi** và **Không** sáng, nút **Thêm** mờ
 - + Các thông tin trên textbox sẽ xóa trắng, mã sinh viên tự động tăng theo phương thức SVxxx.
 - + Combobox Tên khoa hiển thị thông tin các khoa cho người dùng chọn 1 khoa để thêm vào CSDL (*thông tin hiển thị là TenKhoa, giá trị là MaKH*).
- Nhấn nút **Ghi**: thông tin nhập trên các textbox sẽ được lưu vào CSDL.
- Nhấn nút **Không**: các thông tin trên textbox sẽ xóa trắng, nút **Ghi**, **Không** mờ. Nhấn nút **Thoát**: chương trình kết thúc.

Bài 3. Cho cơ sở dữ liệu như sau:



MaCV	TenCV	DonGia
BV	Bảo vệ	5000
CM	Cạo mủ	3000
HC	Hành chính	7000
QL	Quản lý	10000
TC	Chăm sóc cây	3000

MaNV	HoTen	Phai	HSLuong	NgaySinh	NoiSinh
A001	Trần Thị Nhung	False	2.000	12/12/1982 12:...	Tiền Giang
A002	Hồ Thị Lan	False	1.000	1/12/1980 12:0...	Bến Tre
A003	Nguyễn Lan Chi	False	1.300	12/2/1981 12:0...	Vĩnh Long
A004	Trần Thanh Sang	True	1.500	5/6/1982 12:00:...	Tiền Giang
A005	Lê Xuân Thu	True	1.200	12/12/1982 12:...	Trà Vinh

MaNV	MaCV	KLCV
A001	BV	10
A001	CM	20
A002	CM	15
A002	QL	2
A004	BV	10
A004	QL	3
A005	HC	10

Form1

THÔNG TIN CHẤM CÔNG CỦA NHÂN VIÊN

Danh sách nhân viên

- Trần Thị Nhung
- Hồ Thị Lan**
- Nguyễn Lan Chi
- Trần Thanh Sang
- Lê Xuân Thu

Thông tin chi tiết:

Mã nhân viên:

Họ tên:

Giới tính:

Hệ số lương:

Ngày sinh:

Nơi sinh:

	MaCV	TenCV	KLCV	DonGia	ThanhTien
▶	CM	Cạo mủ	15	3000	45000.000
	QL	Quản lý	2	10000	20000.000
*					

- Hiển thị danh sách nhân viên trên listbox.
- Khi click vào một nhân viên thông tin chi tiết được thể hiện trên các textbox, đồng thời thông tin chi tiết được hiển thị trên lưới.
- Khi nhấn nút **Thêm** cho phép các textbox xóa rỗng, **Mã nhân viên (MaNV)** tự động tăng, chỉ cho phép đọc không được phép chỉnh sửa. Nút **Ghi**, **Không** sáng. Nút **Thêm**, **In**, **Thoát** mờ.
- Nhấn **Ghi** cho phép lưu mẫu tin vào **table NhanVien**. Nút **Ghi**, **Không** mờ. Nút **Thêm**, **In**, **Thoát** sáng.
- Nhấn nút **Không** bỏ qua thao tác thêm dữ liệu. Nút **Ghi**, **Không** mờ. Nút **Thêm**, **In**, **Thoát**. Nhấn **Thoát** cho phép thoát khỏi Form.

Chương 4. LINQ

Mục tiêu:

Sau khi học xong chương này, sinh viên có thể:

1. Kiến thức

- + Trình bày các kỹ thuật nền tảng về Linq.
- + Sử dụng Linq để truy vấn dữ liệu từ các đối tượng.
- + Nhận biết cách xây dựng các phương thức mở rộng của Linq.
- + Làm rõ mô hình đối tượng dựa trên CSDL SQL Server.

2. Kỹ năng

- + Sử dụng câu lệnh Linq để truy vấn đến các nguồn dữ liệu ở mức độ tương đơn giản.
- + Thực hiện được các câu truy vấn Linq với CSDL SQL Server.
- + Xây dựng màn hình hiển thị và cập nhật dữ liệu.

3. Thái độ

- + Tự tin sử dụng Linq để xây dựng ứng dụng.

1. Tổng quan về Linq

Linq (Language Integrated Query), Linq cho phép người lập trình thực hiện truy vấn trên nhiều dạng dữ liệu trong .NET như:

- .NET Objects (List, Queue, Array,...).
- Database (DLinq).
- XML (XLinq).
- Parallel Linq (PLinq).

Ví dụ 1: thiết kế Form gồm có TextBox txtKetQua và nút Thực hiện, có 1 mảng số nguyên thực hiện câu truy vấn để lấy ra những số chẵn và sắp xếp giảm dần. Khi nhấp nút Thực hiện kết quả sẽ hiển thị trên TextBox txtKetQua. Cho một mảng các số nguyên, tạo mảng truy vấn để thực thi

```
int[] MangSo = { 50, 5, 40, 15, 25, 30, 10, 9};  
//tạo mảng truy vấn  
IEnumerable<int> query = from n in MangSo  
                        where (n % 2) == 0  
                        orderby n descending  
                        select n;
```

Viết code trong sự kiện Click của nút Thực hiện:

```
//khai báo nguồn dữ liệu  
int[] MangSo = { 50, 5, 40, 15, 25, 30, 10, 9};  
//tạo mảng truy vấn  
IEnumerable<int> query = from n in MangSo  
                        where (n % 2) == 0  
                        orderby n descending
```

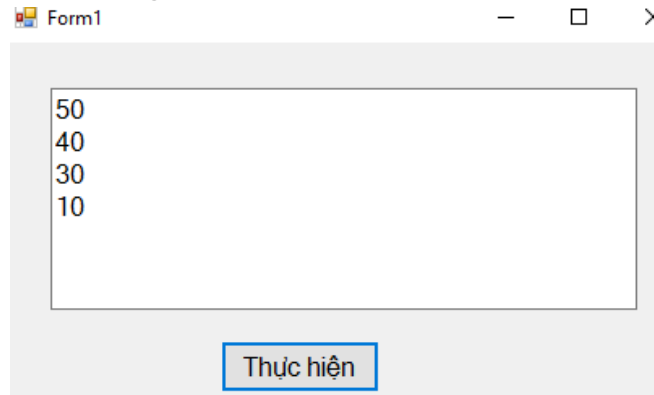
```

        select n;

//thi hành truy vấn
StringBuilder kq = new StringBuilder();
foreach (int pt in query)
kq.AppendLine(pt.ToString());
txtKetQua.Text = kq.ToString();

```

Kết quả sau khi chương trình thực thi:



Hình 4.1. Minh họa kết quả thực thi câu lệnh Linq.

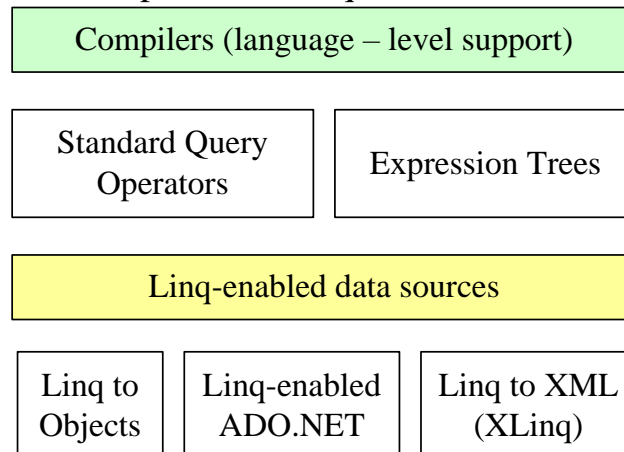
Đặc điểm:

- Truy vấn Linq luôn sử dụng cú pháp nhất quán trong việc truy vấn dữ liệu đến các nguồn dữ liệu khác nhau.
- Trong Linq Query, luôn làm việc với các đối tượng (Objects).
- Có hai dạng cú pháp sử dụng Linq: Query Syntax và Method Syntax

Các Namespace hỗ trợ Linq:

- + System.Linq: hỗ trợ sử dụng các Object.
- + System.Data.Linq: hỗ trợ sử dụng các CSDL quan hệ.
- + System.Data.Objects: hỗ trợ sử dụng các Entities.
- + System.XML.Linq: hỗ trợ sử dụng XML.

Kiến trúc và các thành phần của Linq:



Hình 4.2. Kiến trúc và các thành phần của Linq.

- Tầng Compilers: tầng hỗ trợ ngôn ngữ, ví dụ: VB.NET, C#,...

– Standard Query Operators: hỗ trợ các toán tử dùng trong câu truy vấn LINQ.

– Expression Trees: nối các toán tử lại trước khi thực hiện câu truy vấn.

– Linq-enabled data source: là tầng trung gian hỗ trợ các đối tượng làm việc với các nguồn dữ liệu cụ thể như: Linq to Object, Linq-enabled ADO.NET, Linq to XML,...

– Linq to Object: làm việc với Objects.

– Linq-enabled ADO.NET: làm việc với CSDL, bao gồm:

+ Linq to SQL: dùng để truy vấn đến các dữ liệu từ hệ quản trị CSDL

SQL.

+ Linq to DataSet: dùng để làm việc với dữ liệu lưu trữ trong các DataSet.

+ Linq to entities: dùng để làm việc với các đối tượng entities.

– Linq to XML: làm việc với XML.

Hoạt động truy vấn của LINQ:

– Cú pháp truy vấn LINQ:

```
from id in source
where condition
orderby ordering, ordering,...
select expr
```

– Thành phần hoạt động của truy vấn:

+ Nhận nguồn dữ liệu.

+ Tạo truy vấn.

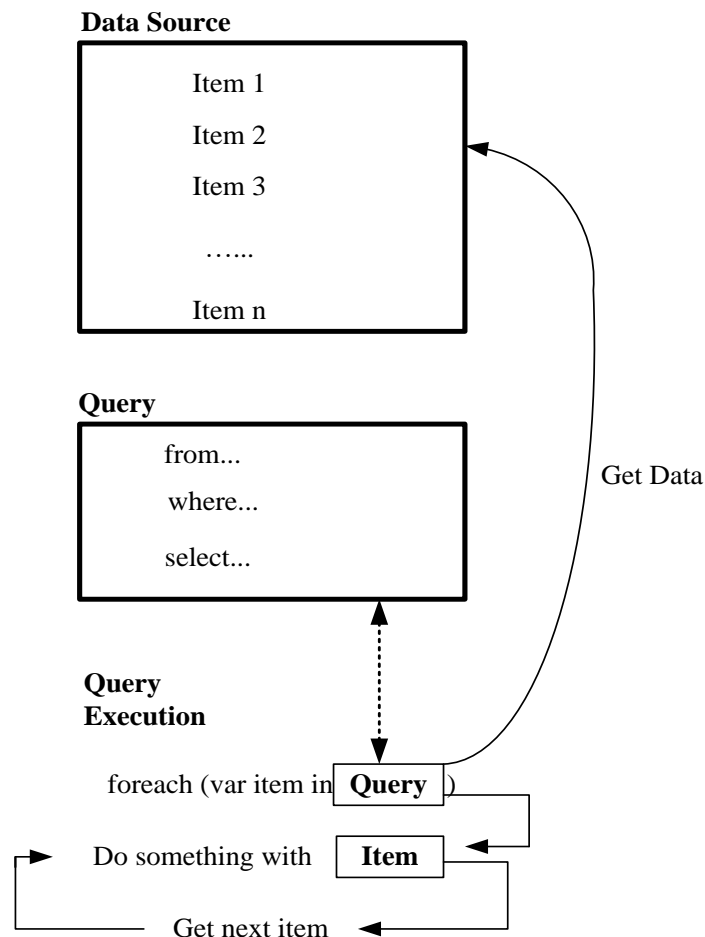
+ Thực thi truy vấn.

Ví dụ 2: thành phần của truy vấn Linq

```
//khai báo nguồn dữ liệu
int[] MangSo = { 12, 5, 4, 5, 15, 30, 10, 9};
//tạo mảng truy vấn
IEnumerable<int> query = from n in MangSo
                        where n >= 10
                        orderby n descending
                        select n;

//thi hành truy vấn
StringBuilder kq = new StringBuilder();
foreach (int pt in query)
    kq.AppendLine(pt.ToString());
txtKetQua.Text = kq.ToString();
```

– Trình tự xử lý:



Hình 4.3. Trình tự xử lý câu lệnh Linq.

– Linq và Generic Types:

- + Các Linq query đều dựa trên kiểu Generic.
- + Các biến Linq query thường được khai báo có kiểu là `IEnumerable<T>` và `IQueryable<T>`, với `<T>` là kiểu dữ liệu truyền vào.
- + Tuy nhiên, vẫn có thể sử dụng từ khóa **var** để thay thế biến kiểu Generic trong một số trường hợp.

Ví dụ 3:

```
string[] MangChuoi = {"khoa", "công", "nghệ", "thông", "tin"};
IEnumerable<string> query = from n in MangChuoi
                             select n.ToLower() + ", " + n.ToUpper()
StringBuilder str = new StringBuilder();
foreach (var p in query)
    str.AppendLine(p);
```

→

```
string[] MangChuoi = {"khoa", "công", "nghệ", "thông", "tin"};
var query = from n in MangChuoi
             select new {thuong=n.ToLower(), hoa=n.ToUpper()};
```

```
StringBuilder str = new StringBuilder();
foreach (var p in query)
    str.AppendLine(p.thuong+ “,” + p.hoa);
```

2. Sự biến đổi dữ liệu trong truy vấn Linq

Linq không những truy xuất dữ liệu mà còn là một công cụ mạnh trong việc biến đổi dữ liệu.

Sử dụng một nguồn làm Input và tạo ra một nguồn mới là Output.

Nguồn mới có thể là một kiểu khác (kiểu cơ sở, Class,...), nguồn mới có thể là một định dạng XML.

Các mối liên hệ kiểu dữ liệu trong hoạt động Linq:

– Truy vấn không làm thay đổi nguồn dữ liệu Input và Output có cùng kiểu dữ liệu.

Ví dụ 4:

```
List<string> MangChuoi = {“khoa”, “công”, “nghệ”, “thông”,
“tin”};
IEnumerable<string> query = from n in MangChuoi
                             select n.ToLower() + “, ” + n.ToUpper()
```

– Truy vấn làm thay đổi dữ liệu nguồn: input và output khác kiểu.

```
Table<Customer> Customers = db.GetTable<Customers>();
IQueryable<string> cusNameQuery = from cus in Customers
```

3. Truy vấn Linq theo biểu thức

Cú pháp:

```
from id in source
{
    from id in source |
    join id in source on expr equals expr[into id]|
    let id = expr | where condition|
    orderby ordering, ordering,...
}
select expr|group expr by key
[into id query]
```

- Toán tử from: dùng để khai báo nguồn dữ liệu nhận vào.
- Từ khóa select: dùng để khai báo nguồn dữ liệu trả về.
- Toán tử join: dùng để liên kết các nguồn dữ liệu.
- Let: dùng để tính toán.
- Where: dùng để lọc dữ liệu.
- Orderby dùng để sắp xếp.

- Intro: tiếp tục với các truy vấn khác.

Với biểu thức truy vấn, có thể lọc, sắp xếp và nhóm dữ liệu.

Linq cung cấp API (Application Programming Interface) được xem như là Standard Query Operations (SQOs) dùng để hỗ trợ các thao tác truy vấn.

Các toán tử được chia thành từng nhóm chức năng như:

- Thống kê: Average, Count, Max, Min,...
- Chuyển đổi: ToArray, ToList, ToDictionary,...
- Nhóm: GroupBy.
- Liên kết: Join, GroupJoin,...
- Sắp xếp: OrderBy, OrderByDescending,...
- Phép chiếu: Select, SelectMany,...
- Giới hạn: where.

Bảng 4.1. Bảng mô tả các toán tử.

Toán tử	Ý nghĩa
where, distinct, take, takewhile, skip, skipwhile	Giới hạn
select, selectmany	Phép chiếu
join, groupjoin	Liên kết
orderby, orderbydescending, thenby, thenbydesending, reverse	Sắp xếp
concat, union, intersect, except	Thiết lập
oftype, cast	Nhập dữ liệu
ToArray, ToList, ToDictionary, ToLookup, AsEnumerable, AsQueryable	Xuất dữ liệu
First, FirstOrDefault, Last, LastOrDefault, Single, SingleOrDefault, ElementAt, ElementAtOrDefault, DefaultIfEmpty	Lọc dữ liệu
Aggregate, Average, Count, LongCount, Sum, Max, Min	Thống kê

- Phép chiếu (Projection Operators): thường sử dụng 2 phương thức:
 - + select: khai báo nguồn dữ liệu trả về trong câu truy vấn.
 - + selectMany: giống select nhưng sử dụng nhiều nguồn dữ liệu (from).

Ví dụ 5: sử dụng selectMany lấy dữ liệu từ mảng numberA và numberB, lọc ra những cặp số $a < b$, với a là một phần tử mảng numberA, b là một phần tử thuộc mảng numberB.

```

IEnumerable<int> query = from n in MangSo
                        select n;
int[] numberA = {0, 2, 4, 6, 8, 9};
int[] numberB = {1, 3, 5, 7, 8};
//trả về các cặp số có điều kiện a<b

```



```
Var pairs = from a in numberA
            from b in numberB
            where a < b
            select new {a, b};
```

– Toán tử giới hạn (Restriction Operator): where

Ví dụ 6: lấy các phần tử chia hết cho 2 và bé hơn 20.

```
IEnumerable<int> query = from n in MangSo
                        where n%2==0 && n<20
                        select n;
```

– Toán tử sắp xếp (Ordering Operation): orderby...[descending]

```
IEnumerable<int> query = from n in MangSo
                        where n%2==0 && n<20
                        orderby n descending
                        select n;
```

Ví dụ 7: thiết kế thông tin các môn học và hệ, sau đó thực thi câu truy vấn lọc ra những môn học có số tiết > 60 và sắp tăng dần theo He:

Khai báo Class MonHoc và He:

```
class MonHoc
{
    public string MaMon { get; set; }
    public string TenMon { get; set; }
    public string He { get; set; }
    public decimal SoTiet { get; set; }
}
class He
{
    public string MaHe { get; set; }
    public string TenHe { get; set; }
}
```

Trong sự kiện Click của nút *Thực thi*:

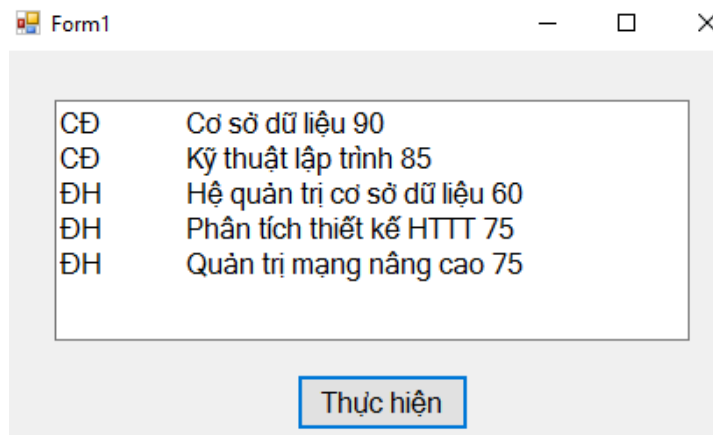
```
//khai báo nguồn dữ liệu MonHoc
List<MonHoc> dsmh = new List<MonHoc>(){
    new MonHoc{MaMon="3230103", TenMon = "Cơ sở dữ liệu", He="CD",
    SoTiet=90},
    new MonHoc{MaMon="3103202", TenMon = "Hệ quản trị cơ sở dữ
    liệu", He="DH", SoTiet=60},
    new MonHoc{MaMon="3234301", TenMon = "Phân tích thiết kế HTTT",
    He="DH", SoTiet=75},
    new MonHoc{MaMon="3133301", TenMon = "Kỹ thuật lập trình",
    He="CD", SoTiet=85},
    new MonHoc{MaMon="3341201", TenMon = "Quản trị mạng nâng cao",
    He="DH", SoTiet=75},
};
//khai báo nguồn dữ liệu He
List<He> dshe = new List<He>()
```

```

{
    new He{MaHe="KTV", TenHe="Kỹ thuật viên"},
    new He{MaHe="CĐ", TenHe="Cao đẳng"},
    new He{MaHe="ĐH", TenHe="Đại học"},
};
//tạo truy vấn
var query = from mh in dsmh
             where mh.SoTiet >= 60
             orderby mh.He
             select new { mh.He, mh.TenMon, mh.SoTiet };
//thi hành câu lệnh truy vấn
StringBuilder str = new StringBuilder();
foreach (var mh in query)
    str.AppendLine(mh.He + "\t" + mh.TenMon + " " +
mh.SoTiet);
txtKetQua.Text = str.ToString();

```

Kết quả sau khi chương trình thực thi:



CĐ	Cơ sở dữ liệu 90
CĐ	Kỹ thuật lập trình 85
ĐH	Hệ quản trị cơ sở dữ liệu 60
ĐH	Phân tích thiết kế HTTT 75
ĐH	Quản trị mạng nâng cao 75

Thực hiện

Hình 4.4. Minh họa kết quả thực thi câu lệnh Linq.

- Phép kết (join Operator) gồm hai toán tử:
 - join: giống INNER JOIN trong SQL, sử dụng: join...in...on...equals...
 - groupjoin: giống join nhưng sử dụng nhiều toán tử from, sử dụng join...in...on...equals...into

Ví dụ 8: liệt kê thêm tên hệ từ nguồn danh sách MonHoc và danh sách He:

```

//tạo truy vấn
var query = from mh in dsmh
             join h in dshe on mh.He equals h.MaHe
             where mh.SoTiet >= 60
             orderby mh.He
             select new { mh.He, h.TenHe, mh.TenMon, mh.SoTiet };
//thi hành câu lệnh truy vấn

```

```

StringBuilder str = new StringBuilder();
foreach (var mh in query)
    str.AppendLine(mh.He + "   " + mh.TenHe + "\t" + mh.TenMon +
": " + mh.SoTiet);
txtKetQua.Text = str.ToString();

```

Kết quả sau khi chương trình thực thi:

CĐ Cao đẳng	Cơ sở dữ liệu: 90
CĐ Cao đẳng	Kỹ thuật lập trình: 85
ĐH Đại học	Hệ quản trị cơ sở dữ liệu: 60
ĐH Đại học	Phân tích thiết kế HTTT: 75
ĐH Đại học	Quản trị mạng nâng cao: 75

Thực hiện

Hình 4.5. Minh họa kết quả thực thi câu lệnh Linq.

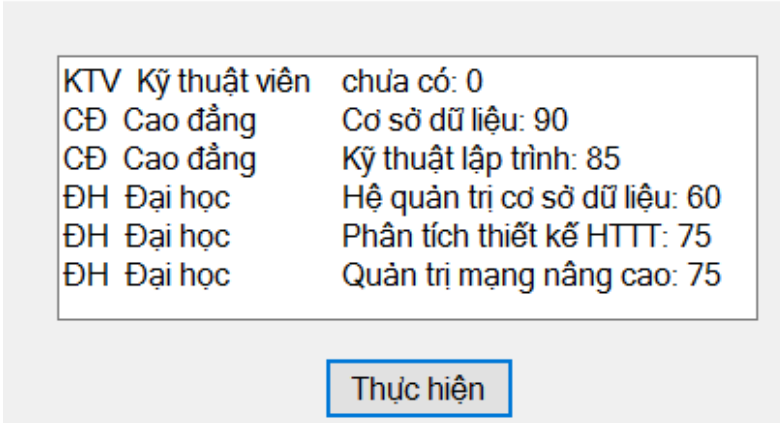
Câu truy vấn lấy ra danh sách tất cả các hệ nếu hệ chưa có môn học thì ghi “chưa có”, thông tin gồm: MaHe, TenHe, TenMon, SoTiet.

```

//tạo truy vấn
var query = from h in dshe
             join mh in dsmh on h.MaHe equals mh.He into qh
             from q in qh.DefaultIfEmpty()
             select new { h.MaHe, h.TenHe, Tenmon = (q == null) ?
"chưa có" : q.TenMon, Sotiet = (q == null) ? 0 : q.SoTiet };
//thi hành câu lệnh truy vấn
StringBuilder str = new StringBuilder();
foreach (var mh in query)
    str.AppendLine(mh.MaHe + "   " + mh.TenHe + "\t" + mh.Tenmon +
": " + mh.Sotiet);
txtKetQua.Text = str.ToString();

```

Kết quả sau khi chương trình thực thi:



The screenshot shows a Windows application window titled 'Form1'. Inside the window, there is a text box containing the following data:

KTV Kỹ thuật viên	chưa có: 0
CĐ Cao đẳng	Cơ sở dữ liệu: 90
CĐ Cao đẳng	Kỹ thuật lập trình: 85
ĐH Đại học	Hệ quản trị cơ sở dữ liệu: 60
ĐH Đại học	Phân tích thiết kế HTTT: 75
ĐH Đại học	Quản trị mạng nâng cao: 75

Below the text box is a button labeled 'Thực hiện'.

Hình 4.6. Minh họa kết quả thực thi câu lệnh Linq.

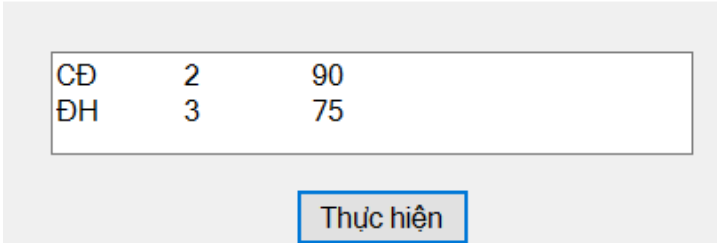
– Toán tử nhóm (Group Operator): `group...by...into`.

Các toán tử thống kê trong **Linq**: `Count`, `LongCount`, `Sum`, `Min`, `Max`,...

Ví dụ 9: thực hiện truy vấn thống kê tổng số môn học của từng hệ và số tiết cao nhất của từng hệ:

```
//tạo truy vấn
var query = from mh in dsmh
             group mh by mh.He into nh
             select new {Mahe=nh.Key, Somon=nh.Count(),
Caonhat=nh.Max(tt=>tt.SoTiet) };
//thi hành câu lệnh truy vấn
StringBuilder str = new StringBuilder();
foreach (var mh in query)
    str.AppendLine(mh.Mahe + "\t" + mh.Somon + "\t" + mh.Caonhat);
txtKetQua.Text = str.ToString();
```

Kết quả sau khi chương trình thực thi:



The screenshot shows a Windows application window titled 'Form1'. Inside the window, there is a text box containing the following data:

CĐ	2	90
ĐH	3	75

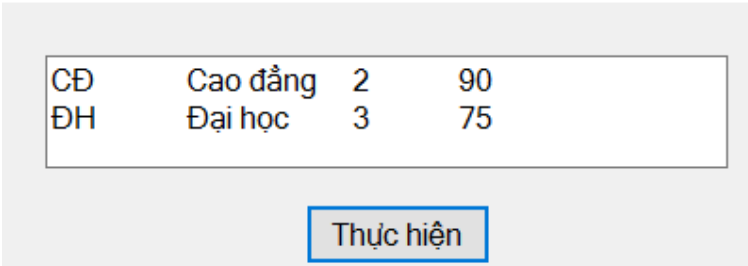
Below the text box is a button labeled 'Thực hiện'.

Hình 4.7. Minh họa kết quả thực thi câu lệnh Linq.

Để hiện thêm thông tin TenHe:

```
//tạo truy vấn
var query = from h in dshe
             join mh in dsmh on h.MaHe equals mh.He
             group mh by new {h.MaHe, h.TenHe} into nh
             select new {Mahe=nh.Key.MaHe , Tenhe = nh.Key.TenHe,
Somon=nh.Count(), Caonhat=nh.Max(tt=>tt.SoTiet) };
//thi hành câu lệnh truy vấn
StringBuilder str = new StringBuilder();
foreach (var mh in query)
    str.AppendLine(mh.Mahe + "\t" + mh.Tenhe + "\t" + mh.Somon
+" \t" + mh.Caonhat);
txtKetQua.Text = str.ToString();
```

Kết quả sau khi chương trình thực thi:



CĐ	Cao đẳng	2	90
ĐH	Đại học	3	75

Thực hiện

Hình 4.8. Minh họa kết quả thực thi câu lệnh Linq.

- Các đối tượng tham gia vào biểu thức LINQ:
 - + IEnumerable<T>:
 - + List<T>
 - + Dictionary<Tkey>, <Tvalue>

Ví dụ 10:

```
Dictionary<int, string> DanhSach = new Dictionary<int, string>
{
    {0, "Không", {1, "Một"}, {2, "Hai"}, {3, "Ba"}, {4, "Bốn"}, {5,
    "Năm"}, {6, "Sáu"}, {7, "Bảy"}, {8, "Tám"}, {9, "Chín"}
};
var query = from sochan in DanhSach
             where (sochan.Key%2)==0
             select sochan;
```

4. Truy vấn Linq theo phương thức

4.1. Phương thức Lambda

Giới thiệu Lambda:

- Bản chất của Linq là các lệnh truy vấn được viết dưới dạng cú pháp Lambda.
- Cú pháp Query dễ đọc, dễ hiểu hơn so với cú pháp Lambda.

– Khi thực thi, cú pháp query sẽ được compiler chuyển về cú pháp Lambda.

– Dùng Lambda mới có thể tận dụng được hết sức mạnh của Linq.

– Cú pháp:

<Input parameter> => Expression hoặc statement block (khối lệnh xử lý)

Với => là toán tử Lambda.

Ví dụ 11:

```
PhepTinh pt = (x, y) =>
{
    int kq = x + y;
    return kq;
};
int tong = pt.Invoke(7, 9);
```

Giới thiệu Delegate **Func** và **Action**:

– Func<> là một họ của nhóm Generic delegate đa năng trong System.Linq, mục đích là hỗ trợ cách dùng delegate ngắn hơn.

Cú pháp:

+ Func<TResult>.

+ Func<T1,... T4, TResult>. Với T1, T2,... là kiểu tham số truyền vào, TResult: là kiểu kết quả trả về.

Func<int, bool> là cách viết gọn của *public delegate bool Func(int A);*

Ví dụ 12:

```
//khởi tạo
Func<int, bool> del = delegate(int A)
{
    return A%2==0;
};
//khởi tạo với Lambda expression
Func<int, bool>del2 = A => A%2==0;
```

– Action<> giống Func nhưng không trả về giá trị.

Cú pháp:

+ Action<T1,...T4>, trong đó T1,...T4: kiểu tham số truyền vào.

4.2. Phương thức mở rộng (Extension Methods)

– Khái niệm:

+ Extension methods: là phương thức tĩnh (static) được kết hợp với một kiểu của .NET.

+ Tính năng này giúp bổ sung các phương thức mới vào các kiểu hiện có mà không cần tạo kiểu (class) dẫn xuất.

+ Hỗ trợ truyền Delegate Function trong đó có Lambda Expression.

+ Phương thức mở rộng có dạng Ten_pt<>.

Ví dụ 13:

```
var query = from c in dsSach
             where c.MaSach == 5
             select new{c.MaSach, c.TenSach};
```

Truy vấn theo phương thức:

```
var query = dsSach
             .where(c=>c.MaSach==5)
             .select(c=>new{c.MaSach, c.TenSach});
```

– Sử dụng phương thức do .NET cấp:

```
.Where
.OrderBy
.Select,...
```

Chú ý:

+ Phải khai báo Namespace System.Linq;

+ Các phương thức mở rộng có dạng Ten_pt<>.

+ Tham số của phương thức này thường là biểu thức Lambda.

Ví dụ 14: lọc ra những môn học có số tiết >60 và sắp tăng dần theo He:

```
//Tạo truy vấn
var query = dsmh
             .Where(mh => mh.SoTiet > 60)
             .OrderBy(mh => mh.He)
             .Select(mh => new {mh.He, mh.TenMon, mh.SoTiet});
//thi hành câu lệnh truy vấn
StringBuilder str = new StringBuilder();
foreach (var mh in query)
    str.AppendLine(mh.He + "\t" + mh.TenMon + "\t" + mh.SoTiet);
txtKetQua.Text = str.ToString();
```

Kết quả sau khi chương trình thực thi:

CĐ	Cơ sở dữ liệu	90	
CĐ	Kỹ thuật lập trình	85	
ĐH	Phân tích thiết kế HTTT	75	
ĐH	Quản trị mạng nâng cao	75	

Thực hiện

Hình 4.9. Minh họa kết quả thực thi câu lệnh Linq.

Như ví dụ 14 nhưng hiển thị thêm TenHe:

```
//Tạo truy vấn
var query = dsmh
    .Join(dshe, mh => mh.He, h => h.MaHe,
        (mh, h) => new {h.MaHe, h.TenHe, mh.TenMon, mh.SoTiet});
//thi hành câu lệnh truy vấn
StringBuilder str = new StringBuilder();
foreach (var mh in query)
    str.AppendLine(mh.MaHe + "\t" + mh.TenHe + "\t" + mh.TenMon
        + ": " + mh.SoTiet);
txtKetQua.Text = str.ToString();
```

Kết quả sau khi chương trình thực thi:

CĐ	Cao đẳng	Cơ sở dữ liệu: 90
ĐH	Đại học	Hệ quản trị cơ sở dữ liệu: 60
ĐH	Đại học	Phân tích thiết kế HTTT: 75
CĐ	Cao đẳng	Kỹ thuật lập trình: 85
ĐH	Đại học	Quản trị mạng nâng cao: 75

Thực hiện

Hình 4.10. Minh họa kết quả thực thi câu lệnh Linq.

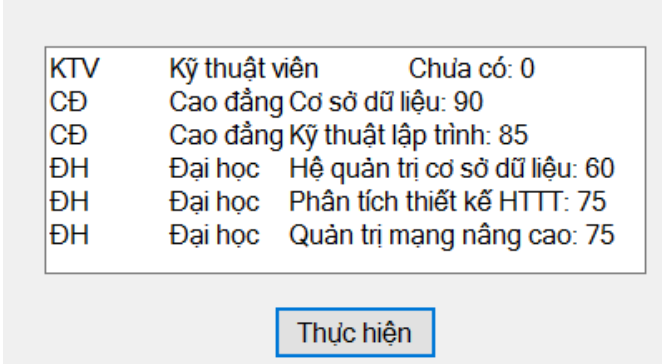
Ví dụ 15: hiển thị tất cả các hệ kể cả những hệ chưa có môn học:

```
//Tạo truy vấn
var query = dshe
    .GroupJoin(dsmh, h=>h.MaHe, mh=>mh.He, (h, qh) => new {h, qh})
    .SelectMany (mh=>mh.qh.DefaultIfEmpty (),
        (hmq, mh)=>new{Mahe=hmq.h.MaHe, Tenhe=hmq.h.TenHe,
        Tenmon=(mh==null)?"Chưa có": mh.TenMon, Sotiet=(mh==null)?0:
        mh.SoTiet});
```



```
//thi hành câu lệnh truy vấn
StringBuilder str = new StringBuilder();
foreach (var mh in query)
    str.AppendLine(mh.Mahe + "\t" + mh.Tenhe + "\t" +
mh.Tenmon + ": " + mh.Sotiet);
txtKetQua.Text = str.ToString();
```

Kết quả sau khi chương trình thực thi:



KTV	Kỹ thuật viên	Chưa có: 0
CĐ	Cao đẳng	Cơ sở dữ liệu: 90
CĐ	Cao đẳng	Kỹ thuật lập trình: 85
ĐH	Đại học	Hệ quản trị cơ sở dữ liệu: 60
ĐH	Đại học	Phân tích thiết kế HTTT: 75
ĐH	Đại học	Quản trị mạng nâng cao: 75

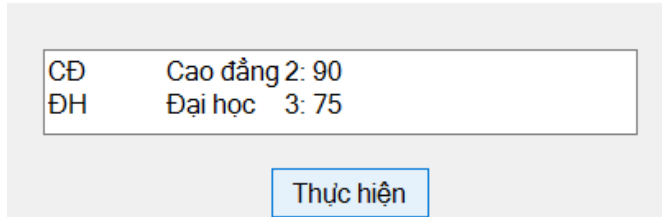
Thực hiện

Hình 4.11. Minh họa kết quả thực thi câu lệnh Linq.

Ví dụ 16: thực hiện truy vấn lấy thông tin: MaHe, TenHe, tổng số môn của từng hệ, số tiết cao nhất của từng hệ:

```
//tạo truy vấn
var query = dsmh
    .Join(dshe, mh => mh.He, h => h.MaHe, (mh, h) => new { mh, h })
    .GroupBy(mh => new { mh.h.MaHe, mh.h.TenHe })
    .Select(mh => new { Mahe=mh.Key.MaHe, Tenhe=mh.Key.TenHe,
Somon=mh.Count(), Caonhat=mh.Max(tt=>tt.mh.SoTiet)});
//thi hành câu lệnh truy vấn
StringBuilder str = new StringBuilder();
foreach (var mh in query)
    str.AppendLine(mh.Mahe + "\t" + mh.Tenhe + "\t" + mh.Somon
+ ": " + mh.Caonhat);
txtKetQua.Text = str.ToString();
```

Kết quả sau khi chương trình thực thi:



CĐ	Cao đẳng 2:	90
ĐH	Đại học 3:	75

Thực hiện

Hình 4.12. Minh họa kết quả thực thi câu lệnh Linq.

- Xây dựng phương thức mở rộng:

Cú pháp:

```
static class <Tên class>
{
    static <kiểu><tên phương thức>(this<kiểu dữ liệu><t.số>,...)
    {
        //xử lý.
    }
}
```

Trong đó:

- + Class chứa nó phải là static.
- + Phương thức phải là static.
- + Tham số đầu tiên phải dùng từ khóa this để khai báo.

Ví dụ 17:

```
public static class PTMR
{
    //kiểm tra tập hợp các chuỗi con có chứa trong chuỗi nguồn
    public static bool CoChua(this string source, string[] chuoicon)
    {
        bool chua = false;
        string[] mang = source.Split('');
        var query = mang.Intersect(chuoicon);
        chua = query.Count()>0;
        return chua;
    }
}
```

Chú ý:

- + Nếu phương thức mở rộng có cùng tên, cùng signature với phương thức thể hiện thì .NET sẽ ưu tiên cho phương thức thể hiện.
- + Biến thành viên, thuộc tính và sự kiện không được mở rộng.

5. Linq to SQL

5.1. Tổng quan Linq to SQL

- Linq to SQL là phương pháp sử dụng Linq để truy vấn các Object có thể liệt kê trong ADO.NET.

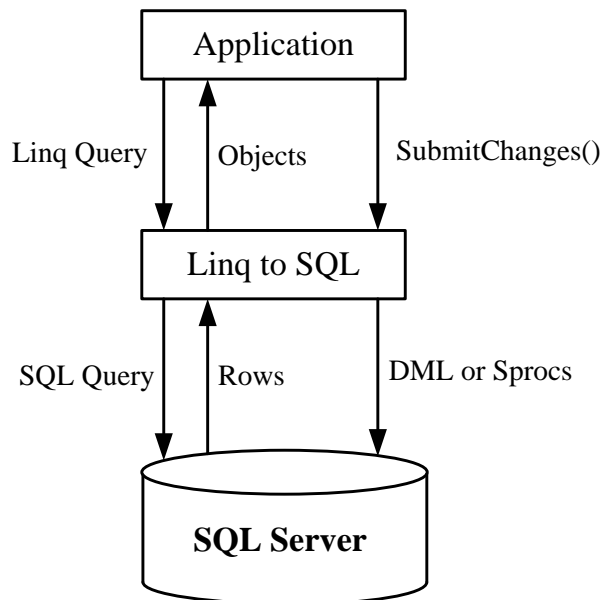
Giới thiệu LINQ to SQL

- Linq to SQL là thành phần của .NET Framework 3.5, cung cấp cơ sở hạ tầng run-time cho việc quản lý dữ liệu quan hệ tương tự như quản lý các đối tượng.
- Sử dụng phương pháp ánh xạ 1 - 1 giữa các đối tượng Database và các đối tượng trong lập trình thông qua ORM (Object – Relational Mapping).

Đặc điểm:

- Cho phép truy vấn, thêm, xóa, sửa dữ liệu từ mô hình đối tượng (Object model).
- Hỗ trợ đầy đủ Transaction, Stored Procedure.
- Cung cấp tính năng kiểm tra tính hợp lệ của dữ liệu và các quy tắc vào trong mô hình đối tượng.

Kiến trúc thi hành của Linq to SQL (Execution Architecture)



Hình 4.13. Kiến trúc thực thi của Linq to SQL.

– Có thể dùng câu lệnh Linq để truy vấn dữ liệu từ CSDL SQL Server, công dụng của Linq to SQL sẽ chuyển các câu lệnh truy vấn Linq thành các câu lệnh SQL và thực thi các câu lệnh SQL với nguồn dữ liệu. Linq to SQL sẽ chuyển các dòng dữ liệu thành các Object...

– Có 3 kỹ thuật trong Linq to ADO.NET:

+ Linq to DataSets: truy vấn dữ liệu với hai đối tượng là DataSet và DataTable.

+ Linq to SQL: dùng truy vấn trên CSDL quan hệ như SQL Server.

+ Linq to Entities: truy vấn dữ liệu với các thực thể do người lập trình định nghĩa.

5.2. Các khái niệm cơ bản

– Object Model: là mô hình được ánh xạ từ mô hình dữ liệu của CSDL.

– ORM (Object Relational Mapping):

+ Là phương pháp ánh xạ trực tiếp 1-1 giữa các đối tượng trong CSDL quan hệ với các Class của .NET.

- + Visual Studio IDE từ phiên bản 2008 có hỗ trợ 2 ORM: Linq to SQL, ADO.NET Entity Framework.
- + O/R Designer (Object Relational Designer)
- + Là một công cụ trong VS IDE có từ phiên bản 2008, dùng để hỗ trợ việc tạo ra Object Model.
- + Phát sinh ra các Entity class trong Project đang làm việc.
- Entity Class:
 - + Các Table, View từ CSDL được ánh xạ thành các Class được gọi là Entity Class.
 - + Khi sử dụng, Entity Class sẽ ánh xạ đến một Table và một Property sẽ ánh xạ đến một Column của Table.

Bảng 4.2. Ánh xạ từ CSDL sang LINQ Object.

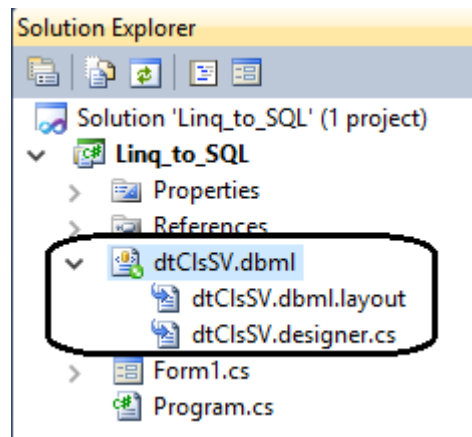
Database Object	LINQ Object
Database	DataContext
Table	Class and Collection
View	Class and Collection
Column	Property
Relationship	Nested Collection
Store Procedure	Method

Association: mỗi kết hợp hay còn gọi là mối quan hệ giữa 2 Entity dựa trên Primary key và Foreign key.

- DataContext:
 - + Là một lớp trong .NET, dùng để hỗ trợ việc kết nối CSDL. Ngoài ra, còn quản lý các định danh (Identity) của các đối tượng trong CSDL như Table, View,...
 - + Có thể tạo ra một Strong-Typed DataContext cho riêng ứng dụng → đây là cách thông dụng nhất.

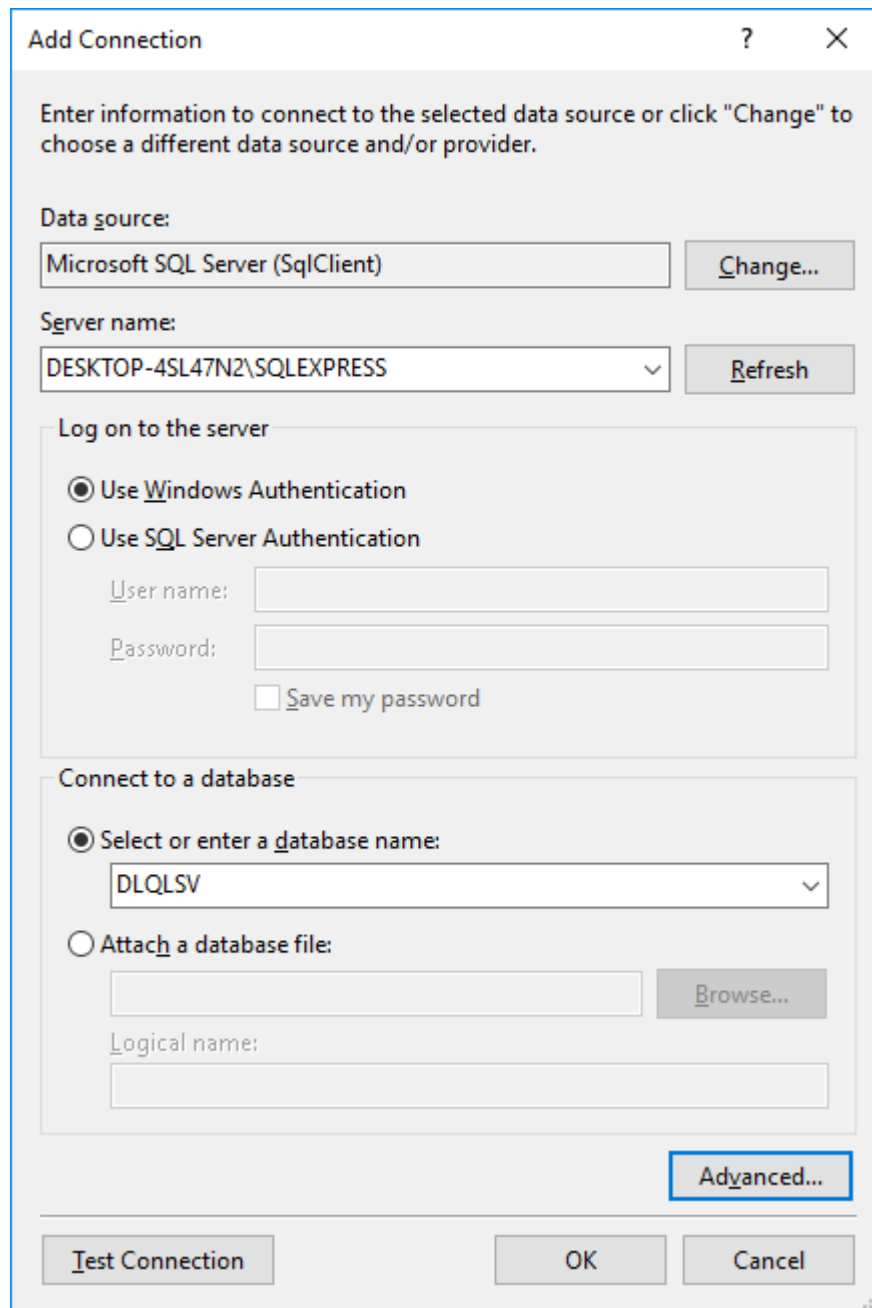
5.3. Tạo Object model sử dụng công cụ Object Relational Designer

- Tạo một CSDL quản lý sinh viên như bài tập 1 chương 3 trong SQL Server, sau đó chuyển sang mô hình tạo Object model.
- Nhấp chuột phải vào tên Project → chọn Add → New Item... → Linq to SQL Classes, đặt tên là: dtClsSV.dbml.



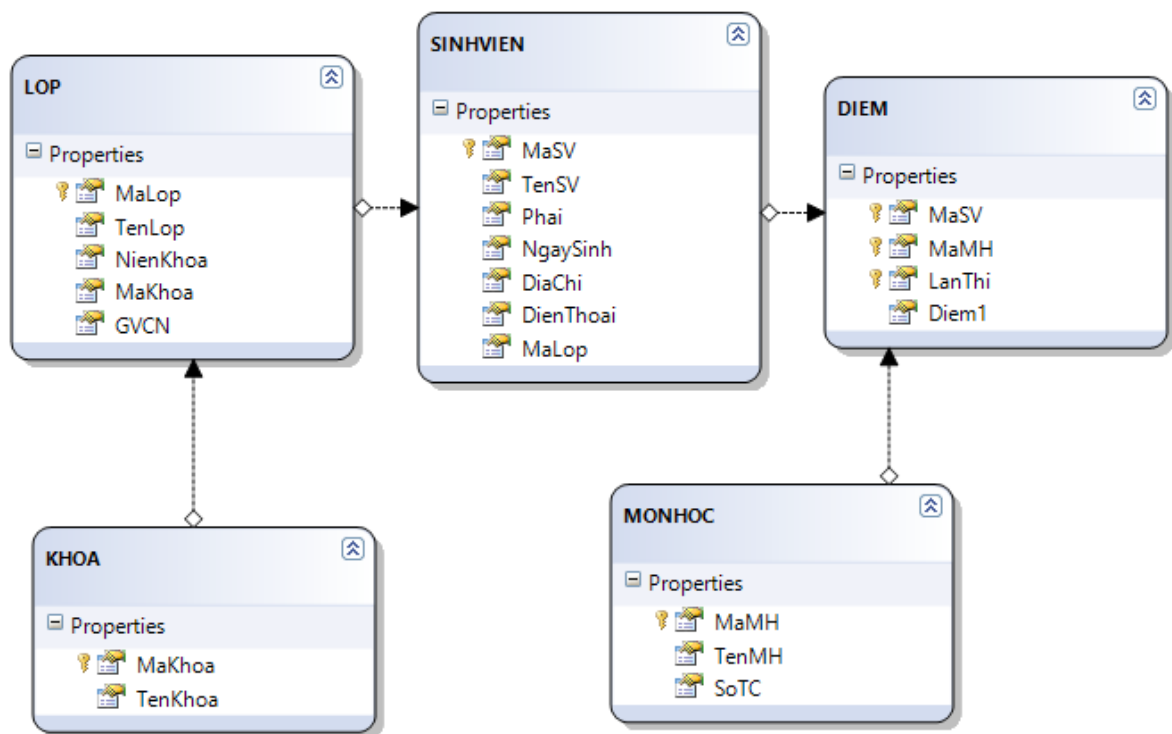
Hình 4.14. Minh họa thao tác tạo Object model.

- Chọn dữ liệu nguồn để kết nối: vào View chọn Server Explorer, nhấp chuột phải vào Data Connections → Add Connection... → điền Server name và chọn tên Database như hình bên dưới, chọn OK.



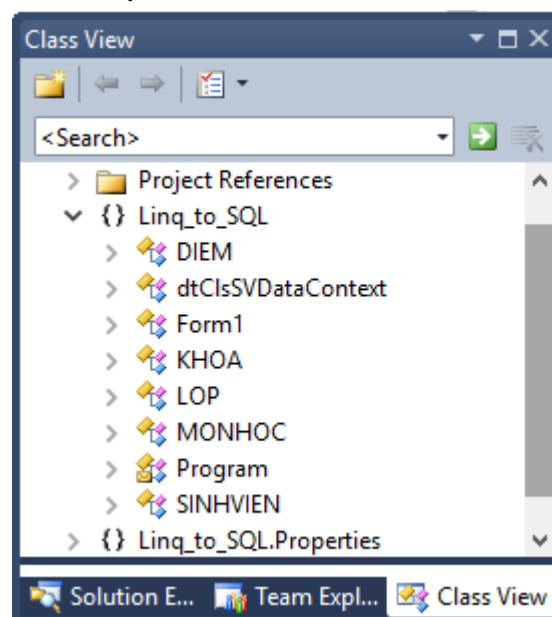
Hình 4.15. Minh họa chọn dữ liệu nguồn để kết nối.

- Sau khi kết nối, sẽ thấy xuất hiện các Table và mối quan hệ giữa các Table trong CSDL quản lý sinh viên.
- Để tạo mô hình đối tượng: chọn các đối tượng từ CSDL quản lý sinh viên và kéo vào cửa sổ Designer:



Hình 4.16. Mô hình dữ liệu đối tượng được ánh xạ từ CSDL.

- Tương ứng với từng Table bên CSDL sẽ tạo ra các Entity Class tương ứng và trong mỗi Column trong Table là các Property tương ứng trong Entity Class. Có thể thêm các Properties hoặc sửa tên các Properties trong Entity Class.
- Mở cửa sổ View để xem các đối tượng được phát sinh trong mô hình đối tượng quản lý sinh viên vừa tạo: View → Class View:



Hình 4.17. Cửa sổ Class View.

- Lớp dtClsSVDDataContext: đây là lớp quản lý các đối tượng bên trong mô hình đối tượng và quản lý thông tin kết nối đến CSDL SQL Server. Thông tin

kết nối này sẽ được lưu trữ trong tập tin app.config, có thể thay đổi thông tin kết nối trong tập tin này.

5.4. Các thao tác với Linq to SQL

5.4.1. Đọc dữ liệu

– Nguồn dữ liệu của câu truy vấn có thể là một Table, một View hoặc một Store Procedure.

– Các bước thực hiện:

+ Khai báo nguồn dữ liệu.

+ Tạo truy vấn.

Ví dụ 18: tạo truy vấn lấy thông tin của những sinh viên có tên bắt đầu bằng chữ “T”, thiết kế Form như hình bên dưới có DataGridView (dtg_HienThi) và nút Hiện thị:

Viết mã lệnh cho sự kiện Click của nút Thực hiện:

```
//khai báo nguồn dữ liệu
dtClsSVDDataContext db = new dtClsSVDDataContext();
Table<SINHVIEN> Sinh_vien = db.GetTable<SINHVIEN>();
//tạo truy vấn
var query = from sv in Sinh_vien
             where sv.TenSV.StartsWith("T")
             orderby sv.MaLop
             select new {sv.MaLop, sv.MaSV, sv.TenSV,
sv.NgaySinh, sv.DiaChi};
//xuất kết quả
dtg_HienThi.DataSource = query;
```

Kết quả sau khi chương trình thực thi:

The screenshot shows a Windows Form titled 'Form1'. It contains a DataGridView with 6 columns: an empty column, 'MaLop', 'MaSV', 'TenSV', 'NgaySinh', and 'DiaChi'. The data rows are:

	MaLop	MaSV	TenSV	NgaySinh	DiaChi
▶	CDCKA	CKA01	Trần Văn An	11/10/1986	14 Dinh Tien Hoang Q1
	CDCKB	CKB01	Trần Trọng Nhân	09/01/1986	12 Hung Vuong Q1
	CDTHB	THB03	Trịnh Hữu Tín	03/08/1986	152 Hung Vuong Q1

Below the DataGridView is a button labeled 'Thực hiện'.

Hình 4.18. Minh họa kết quả thực thi câu truy vấn Linq to SQL.

– Lấy thêm thông tin TenLop từ bảng LỚP:

```
//khai báo nguồn dữ liệu
dtClsSVDDataContext db = new dtClsSVDDataContext();
Table<SINHVIEN> Sinhvien = db.GetTable<SINHVIEN>();
Table<LOP> Lops = db.GetTable<LOP>();
//tạo truy vấn
```



```
var query = from l in Lops
            join sv in Sinhviens on l.MaLop equals sv.MaLop
            where sv.TenSV.StartsWith("T")
            orderby sv.MaLop
            select new {sv.MaLop, l.TenLop, sv.MaSV, sv.TenSV,
sv.NgaySinh, sv.DiaChi};
//xuất kết quả
dtg_HienThi.DataSource = query;
```

Kết quả sau khi chương trình thực thi:

Form1

	MaLop	TenLop	MaSV	TenSV	NgaySinh	DiaChi
▶	CDCKA	Cao đẳng cơ khí A	CKA01	Trần Văn An	11/10/1986	14 Dinh Ti...
	CDCKB	Cao đẳng cơ khí B	CKB01	Trần Trọng...	09/01/1986	12 Hung V...
	CDTHB	Cao đẳng tin học B	THB03	Trịnh Hữu ...	03/08/1986	152 Hung ...

Thực hiện

Hình 4.19. Minh họa kết quả thực thi câu truy vấn Linq to SQL.

– Hiện thị thông tin tất cả các khoa kể cả các khoa không có sinh viên:

```
//khai báo nguồn dữ liệu
dtClsSVDataContext db = new dtClsSVDataContext();
Table<SINHVIEN> Sinhviens = db.GetTable<SINHVIEN>();
Table<LOP> Lops = db.GetTable<LOP>();
//tạo truy vấn
var query = from l in Lops
            join sv in Sinhviens on l.MaLop equals sv.MaLop into qh
            from q in qh.DefaultIfEmpty ()
            orderby l.MaLop
            select new {l.MaLop, l.TenLop, masv=(q==null)?"chưa
có":q.MaSV, tensv=(q==null)?"chưa có":q.TenSV,
diachi=(q==null)?"chưa có":q.DiaChi};
//xuất kết quả
dtg_HienThi.DataSource = query;
```

Kết quả sau khi chương trình thực thi:

Form1

	MaLop	TenLop	masv	tensv	diachi
	CDTHB	Cao đẳng tin h...	THB01	Lê Thanh Phong	53 Dinh Tien Hoang...
	CDTHB	Cao đẳng tin h...	THB02	Nguyễn Phương ...	534 Cach mang Th...
	CDTHB	Cao đẳng tin h...	THB03	Trịnh Hữu Tín	152 Hung Vương Q1
	CDTHB	Cao đẳng tin h...	THB04	Đào Thị Trinh	234 Hung Vương Q1
	THA	Tin học Cơ bản	chưa có	chưa có	chưa có

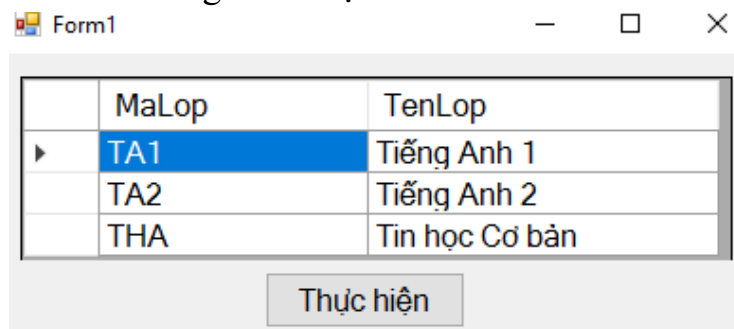
Thực hiện

Hình 4.20. Minh họa kết quả thực thi câu truy vấn Linq to SQL.

- Hiển thị thông tin các lớp mà các lớp đó chưa có sinh viên theo học:

```
//khai báo nguồn dữ liệu
dtClsSVDataContext db = new dtClsSVDataContext();
Table<SINHVIEN> Sinhviens = db.GetTable<SINHVIEN>();
Table<LOP> Lops = db.GetTable<LOP>();
//tạo truy vấn
var query = from l in Lops
            where !(from sv in Sinhviens select
                    sv.MaLop).Contains(l.MaLop)
            select new { l.MaLop,l.TenLop };
//xuất kết quả
dtg_HienThi.DataSource = query;
```

Kết quả sau khi chương trình thực thi:



	MaLop	TenLop
▶	TA1	Tiếng Anh 1
	TA2	Tiếng Anh 2
	THA	Tin học Cơ bản

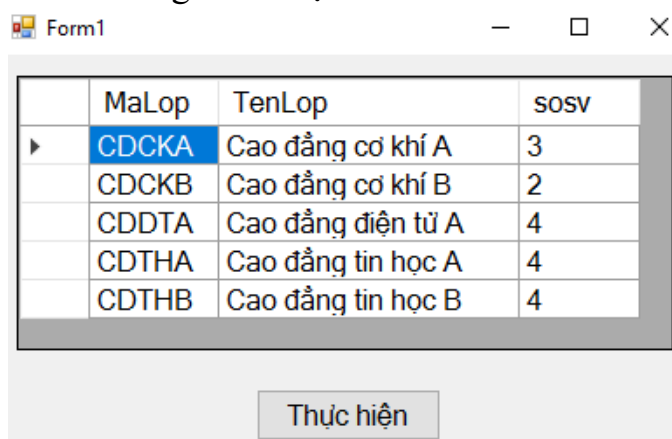
Thực hiện

Hình 4.21. Minh họa kết quả thực thi câu truy vấn Linq to SQL.

- Đếm số sinh viên của từng lớp:

```
//khai báo nguồn dữ liệu
dtClsSVDataContext db = new dtClsSVDataContext();
Table<SINHVIEN> Sinhviens = db.GetTable<SINHVIEN>();
Table<LOP> Lops = db.GetTable<LOP>();
//tạo truy vấn
var query = from l in Lops
            join sv in Sinhviens on l.MaLop equals sv.MaLop
            group sv by new { l.MaLop, l.TenLop } into nh
            select new { nh.Key .MaLop, nh.Key .TenLop, sosv = nh.Count()};
```

Kết quả sau khi chương trình thực thi:



	MaLop	TenLop	sosv
▶	CDCKA	Cao đẳng cơ khí A	3
	CDCKB	Cao đẳng cơ khí B	2
	CDDTA	Cao đẳng điện tử A	4
	CDTHA	Cao đẳng tin học A	4
	CDTHB	Cao đẳng tin học B	4

Thực hiện

Hình 4.22. Minh họa kết quả thực thi câu truy vấn Linq to SQL.

- Cho biết lớp có nhiều sinh viên nhất:

```
var query = (from l in Lops
              join sv in Sinhviens on l.MaLop equals sv.MaLop
              group sv by new { l.MaLop, l.TenLop } into nh
              orderby nh.Count() descending
              select new { nh.Key.MaLop, nh.Key.TenLop, SoSVMax =
nh.Count() }).Skip (0).Take (1);
```

5.4.2. Cập nhật dữ liệu

- Thêm dữ liệu:

+ Cú pháp: `Table<TEntity>.InsertOnSubmit(Tentity)`

Ví dụ:

```
KHOA k = new KHOA();
k.MaKhoa = "HH";
k.TenKhoa = "Khoa Hóa";
Khoas.InsertOnSubmit(k);
```

- Xóa dữ liệu:

+ Cú pháp: `Table<TEntity>.DeleteOnSubmit(Tentity)`

Ví dụ 19: xóa khoa có mã "HH":

```
var k = Khoas.Single(kh=>kh.MaKhoa == "HH");
Khoas.DeleteOnSubmit(k);
```

- Sửa dữ liệu:

+ Truy vấn lấy ra một dòng cần thay đổi có kiểu TEntity, gán giá trị mới cho cột.

Ví dụ 20: sửa tên khoa của khoa có mã HH thành "Khoa Hóa học":

```
var k = Khoas.Single(kh=>kh.MaKhoa == "HH");
k.TenKhoa = "Khoa Hóa học";
```

- Cập nhật sự thay đổi về CSDL:

+ Cú pháp: `<DataContext>.SubmitChanges();`

```
dtClsSVDataContext db = new dtClsSVDataContext();
//Thêm, sửa, xóa dữ liệu
.....
db.SubmitChanges();
```

Ví dụ 21: thêm mới một khoa, với MaKhoa = "HH", TenKhoa = "Khoa Hóa"

Khai báo dữ liệu nguồn:

```

public Form1()
{
    InitializeComponent();
}
//khai báo nguồn dữ liệu
dtClsSVDataContext db = new dtClsSVDataContext();
Table<KHOA> Khoas;

```

Sự kiện Form_Load: `Khoas = db.GetTable<KHOA>();`

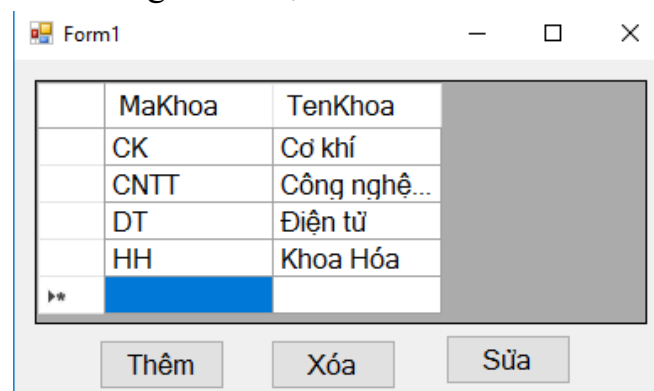
Sự kiện Click của nút Thêm:

```

KHOA k = new KHOA();
k.MaKhoa = "HH";
k.TenKhoa = "Khoa Hóa";
Khoas.InsertOnSubmit(k);
db.SubmitChanges();
dtg_HienThi.DataSource = Khoas;

```

Kết quả sau khi chương trình thực thi:



Hình 4.23. Minh họa kết quả thực thi câu truy vấn Linq to SQL.

– Sửa tên khoa của có mã “HH” thành “Khoa Hóa học”:

Sự kiện Click của nút Sửa:

```

KHOA k = Khoas.Single(kh=>kh.MaKhoa == "HH");
k.TenKhoa = "Khoa Hóa học";
db.SubmitChanges();
dtg_HienThi.DataSource = Khoas;

```

Kết quả sau khi chương trình thực thi:

MaKhoa	TenKhoa
CK	Cơ khí
CNTT	Công nghệ thông tin
DT	Điện tử
HH	Khoa Hóa học

Thêm Xóa Sửa

Hình 4.24. Minh họa kết quả thực thi câu truy vấn Linq to SQL.

Ví dụ 22: Cập nhật GVCN của tất cả các lớp của Khoa công nghệ thông tin thành “Phạm Ngọc Giàu”. Với dữ liệu ban đầu của Table LOP như sau:

MaLop	TenLop	NienKhoa	MaKhoa	GVCN
CDCKA	Cao đẳng cơ kh...	2000-2003	CK	Lê Thành Nhân
CDCKB	Cao đẳng cơ kh...	2000-2003	CK	Lê Thành Nhân
CDDTA	Cao đẳng điện ...	2001-2004	DT	Phùng Đăng Khoa
CDTHA	Cao đẳng tin h...	2000-2003	CNTT	Trần Bình Trọng
CDTHB	Cao đẳng tin h...	2000-2003	CNTT	Nguyễn Thanh Bình
TA1	Tiếng Anh 1	2000-2003	CNTT	Nguyễn Thành Nhân
TA2	Tiếng Anh 2	2000-2003	CK	Trần Bình Trọng
THA	Tin học Cơ bản	2000-2003	CK	Nguyễn Văn Anh

Hình 4.25. Dữ liệu bảng LOP.

Sự kiện Click của nút Sửa:

```
var dsLop = from lop in Lops
            where lop.MaKhoa == "CNTT"
            select lop;
foreach (var lop in dsLop)
    lop.GVCN = "Phạm Ngọc Giàu";
db.SubmitChanges();
dtg_HienThi.DataSource = Lops;
```

Kết quả sau khi chương trình thực thi:

MaLop	TenLop	NienKhoa	MaKhoa	GVCN
CDCKA	Cao đẳng cơ khí A	2000-2003	CK	Lê Thành Nhân
CDCKB	Cao đẳng cơ khí B	2000-2003	CK	Lê Thành Nhân
CDDTA	Cao đẳng điện tử A	2001-2004	DT	Phùng Đăng Khoa
CDTHA	Cao đẳng tin học A	2000-2003	CNTT	Phạm Ngọc Giàu
CDTHB	Cao đẳng tin học B	2000-2003	CNTT	Phạm Ngọc Giàu
TA1	Tiếng Anh 1	2000-2003	CNTT	Phạm Ngọc Giàu
TA2	Tiếng Anh 2	2000-2003	CK	Trần Bình Trọng
THA	Tin học Cơ bản	2000-2003	CK	Nguyễn Văn Anh

Hình 4.26. Dữ liệu bảng LOP sau khi thực hiện truy vấn cập nhật.

- Xóa dữ liệu: xóa thông tin của khoa có mã là “HH”:

MaKhoa	TenKhoa
CK	Cơ khí
CNTT	Công nghệ thông tin
DT	Điện tử
HH	Khoa Hóa học

Hình 4.27. Dữ liệu bảng KHOA.

Sự kiện Click của nút xóa:

```
KHOA k = Khoas.Single(kh => kh.MaKhoa == "HH");
Khoas.DeleteOnSubmit(k);
db.SubmitChanges ();
dtg_HienThi .DataSource = Khoas;
```

Kết quả sau khi chương trình thực thi:

MaKhoa	TenKhoa
CK	Cơ khí
CNTT	Công nghệ thông tin
DT	Điện tử
NULL	NULL

Hình 4.28. Dữ liệu bảng KHOA sau khi thực hiện thao tác xóa.

Xóa thông tin của tất cả các khoa có mã khoa bắt đầu bằng chữ “N”. Với dữ liệu ban đầu của Table KHOA như sau:

MaKhoa	TenKhoa
CK	Cơ khí
CNTT	Công nghệ thông tin
DT	Điện tử
N1	Tiếng Anh 1
N2	Tiếng Anh 2
N3	Tiếng Anh chuyên ngành

Hình 4.29. Dữ liệu bảng KHOA.

Sự kiện Click của nút Xóa:

```
var dsKhoaXoa = from kh in Khoas
                where kh.MaKhoa.StartsWith("N")
                select kh;
Khoas.DeleteAllOnSubmit(dsKhoaXoa);
db.SubmitChanges();
```

Kết quả sau khi chương trình thực thi:

MaKhoa	TenKhoa
CK	Cơ khí
CNTT	Công nghệ thông tin
DT	Điện tử
NULL	NULL

Hình 4.30. Dữ liệu bảng KHOA sau khi thực hiện thao tác xóa.

BÀI TẬP

Sử dụng Linq để làm các bài tập sau:

Bài 1. Cho mảng nguyên sau:

`int[] MangSo = {50, 42, 15, 14, 3, 9, 8, 7, 12, 24, 1, 0}`

- Hãy liệt kê các số chia hết cho 5 và 3.
- Hãy liệt kê các số chia hết cho 3 và không chia hết cho 5 trong mảng.
- Hãy liệt kê các phần tử là số lẻ.
- Tìm các số > 3 và < 10 .
- Hãy liệt kê các phần tử thành một mảng mới với điều kiện: nếu phần tử là số chẵn thì chia 2 ngược lại giữ nguyên giá trị, kết quả như sau:
`{25, 21, 15, 7, 3, 9, 4, 7, 6, 12, 1, 0}`

Bài 2. Cho mảng chuỗi sau:

`string[] MangChuoi = {"sài", "gòn", "cô", "gái"}`

- Hãy liệt kê các số có 4 ký tự và sắp xếp tăng dần theo ký tự đầu tiên.
- Hãy liệt kê các số và thể hiện thông tin có dạng <chữ thường> - <chữ hoa>.
- Hãy liệt kê các số có chứa chữ “a”.
- Hãy liệt kê các số có ký tự đầu tiên là chữ hoa.

Bài 3. Cho thông tin Phim và TheLoai như sau:

`Phim(TenPhim, NuocSX, MaTheLoai, SoTap).`

`TheLoai(MaTheLoai, TenTheLoai).`

Phim:

MaPhim	TenPhim	NuocSX	DonGia	MaTheLoai	SoTap
P0001	Thập diện mai phục	Hồng Kông	200000	TC	10
P0002	Anh em nhà bác sĩ	Hàn Quốc	150000	TC	50
P0003	Cánh đồng bất tận	Việt Nam	250000	TC	5
P0004	Ván bài lật ngửa	Việt Nam	300000	HĐ	3
P0005	Tom and Jerry	Hà Lan	190000	HH	90
P0006	Con đường mưa	Việt Nam	200000	HH	20
P0007	Cô dâu 18 tuổi	Ấn Độ	21000	TC	15

TheLoai:

MaTheLoai	TenTheLoai
TC	Tình cảm
HĐ	Hành động
HH	Hoạt hình
KH	Kiểm hiệp

– Hãy liệt kê thông tin phim thuộc thể loại “HĐ”: MaTheLoai, TenTheLoai, SoTap. Sắp xếp TenPhim tăng dần.

– Hãy liệt kê thông tin: MaTheLoai, TenTheLoai, TenPhim, SoTap.

– Hãy liệt kê thông tin: MaTheLoai, TenTheLoai, TenPhim, SoTap, liệt kê cả những thể loại chưa có phim.

– Cho biết thông tin phim của mỗi thể loại, thông tin gồm: MaTheLoai, Số phim, tổng số tập.

– Cho biết thông tin phim của mỗi thể loại, thông tin gồm: MaTheLoai, Số phim, tổng số tập, số tập nhiều nhất.

Bài 4. Sử dụng truy vấn LINQ theo phương thức Lambda và mở rộng thực hiện các yêu cầu như bài 3.

Bài 5. Cho cơ sở dữ liệu trong SQL Server quản lý phim có dữ liệu giống bài 3, thực hiện các yêu cầu sau:

– Hãy liệt kê thông tin phim bắt đầu bằng ký tự “C” như sau: MaTheLoai, MaPhim, DonGia. Sắp xếp MaTheLoai tăng dần.

– Hãy liệt kê thông tin phim có DonGia ≥ 200000 như sau: TenTheLoai, MaPhim, TenPhim, DonGia. Sắp xếp theo MaTheLoai tăng dần.

– Hãy liệt kê thông tin phim như sau: TenTheLoai, MaPhim, TenPhim, DonGia. Sắp xếp theo MaTheLoai tăng dần, liệt kê cả những thể loại chưa có phim.

– Hãy liệt kê các thể loại chưa có phim, thông tin gồm: MaTheLoai, TenTheLoai.

– Hãy thống kê phim của mỗi thể loại, thông tin gồm: MaTheLoai, TenTheLoai, số phim, đơn giá cao nhất. Chỉ liệt kê những thể loại có từ 2 phim trở lên.

Bài 6. Cho cơ sở dữ liệu trong SQL Server quản lý phim có dữ liệu giống bài 3, thực hiện các yêu cầu sau:

– Thêm một thể loại mới vào Table TheLoai, với các thông tin sau: MaTheLoai: TL, TenTheLoai: Tâm lý.

– Thêm một phim mới thuộc thể loại Tâm lý vào Table Phim với các thông tin sau: MaPhim: P0006, TenPhim: Chuyện nhà mẹ, DonGia: 200000.

- Cập nhật lại TenTheLoai của thể loại TL thành *Tâm lý xã hội*.
- Tăng đơn giá của các Phim thuộc thể loại Tình cảm thêm 10%.
- Xóa phim: Chuyện nhà mẹ.
- Xóa thể loại phim: TL.

Chương 5. XML

Mục tiêu:

Sau khi học xong chương này, sinh viên có thể:

1. Kiến thức
 - + Nhận biết cấu trúc tài liệu dạng XML.
 - + Thực hiện truy xuất, thêm, xóa, sửa dữ liệu dạng XML.
2. Kỹ năng
 - + Thực hiện lưu trữ, thực hiện truy xuất, cập nhật dữ liệu dưới dạng XML.
3. Thái độ
 - + Tự tin thực hiện các thao tác với dữ liệu dạng XML.

1. Tổng quan về XML

– XML (eXtensible Markup Language): ngôn ngữ định dạng mở rộng. XML là ngôn ngữ xây dựng cấu trúc tài liệu văn bản dựa theo chuẩn SGML (Standard Generalized Markup Language) do W3C (World Wide Web Consortium) phát triển, tương tự như CSDL, dùng văn bản (Text) để mô tả thông tin.

– Công nghệ XML là kết quả của các nghiên cứu về dạng biểu diễn thông tin khi cần trao đổi giữa các hệ thống tin học. Dạng biểu diễn thỏa mãn các yêu cầu sau:

- + Cho phép trao đổi trên phạm vi rộng (Internet).
- + Dễ dàng cho việc kết xuất và tiếp nhận khi trao đổi.
- + Tuân theo một định chuẩn chung được chấp nhận và hỗ trợ của nhiều môi trường phát triển phần mềm.

XML có thể được sử dụng để:

- + Trao đổi thông tin giữa các tầng của một ứng dụng được thiết kế theo mô hình kiến trúc đa tầng.
- + Trao đổi thông tin giữa 1 tầng với hệ thống khác bên ngoài.

– XML và vấn đề lưu trữ thông tin:

Có 3 ứng dụng chính của XML để lưu trữ dữ liệu bên trong hệ thống tin học:

- + Cách 1: chỉ sử dụng các tập tin XML để lưu trữ dữ liệu.
 - Ưu: không cần sự hỗ trợ của các hệ quản trị CSDL → dễ cài đặt, dễ triển khai.
 - Nhược: hiệu quả không cao khi khối lượng dữ liệu lớn.
- + Cách 2: một số dữ liệu lưu trữ dưới dạng tập tin XML, một số khác lưu trữ bên trong dữ liệu.

+ Cách 3: lưu trữ toàn bộ bên trong CSDL, tài liệu XML khi đó được nhúng vào nội dung các bảng dữ liệu.

- Ưu (cách 2 và 3): có thể kết hợp tốt ưu điểm của hai mô hình lưu trữ thông tin: XML, CSDL.

- Khuyết (cách 2, 3): cần có sự cân nhắc và quyết định đúng đắn loại thông tin nào sẽ đúng với hình thức lưu trữ nào.

- XML và cấu trúc dữ liệu:

- + Đọc/Ghi dễ dàng:

Đọc tài liệu XML: `Tai_Lieu.Load(Ten_tap_tin_XML)`.

Ghi tài liệu XML: `Tai_Lieu.Save(Ten_tap_tin_XML)`.

- + Khả năng truy vấn cao:

Việc truy vấn các thành phần/tập hợp các thành phần của cấu trúc dữ liệu động thường phải sử dụng các vòng lặp duyệt đến từng phần tử. Với tài liệu XML có thể sử dụng ngôn ngữ truy vấn Xpath để truy xuất đến thành phần/tập hợp thành phần một cách rất dễ dàng.

Ví dụ 1: có `Tai_lieu`, để lập danh sách các nút có giá trị dương:

`Nut_duong = Tai_lieu.SelectSingleNode("Pha_tu[@Gia_tri>0]")`.

2. Cấu trúc tài liệu XML

Một tài liệu XML được chia thành hai phần chính:

- Phân khai báo: khai báo tài liệu XML: khai báo phiên bản, bảng mã ký tự được sử dụng trong tài liệu. Phần này có thể có hoặc không, nếu có phải là dòng đầu tiên của văn bản.

`<?xml version= "1.0" encoding = "utf-8">`

- Phần thân: chứa nội dung dữ liệu. Gồm một hay nhiều phần tử (thẻ), mỗi phần tử được chứa trong một cặp thẻ mở và đóng, phần tử đầu tiên là thẻ đánh gốc (nút gốc). Một tài liệu XML được xem là hợp lệ khi: các khai báo phải được đặt ở dòng đầu tiên của tài liệu, chỉ có một thẻ đánh dấu gốc, các thẻ đánh dấu thành phần đều nằm giữa cặp thẻ đánh dấu gốc và phải lồng nhau một cách hợp lý.

Ví dụ 2: tài liệu XML biểu diễn thông tin của phân số.

`<?xml version= "1.0" encoding = "utf-8">`

`<PHAN_SO>`

`<TU_SO>3</TU_SO>`

`<MAU_SO>4</MAU_SO>`

`</PHAN_SO>`

- + Thẻ đánh dấu gốc (nút gốc): chứa tất cả các phần tử trong văn bản. Mỗi tài liệu chỉ có một nút gốc.

Ví dụ 3: tài liệu XML biểu diễn thông tin của phân số có nút gốc là PHAN_SO:

```
<?xml version= "1.0" encoding = "utf-8">
  <PHAN_SO>
    <TU_SO>3</TU_SO>
    <MAU_SO>4</MAU_SO>
  </PHAN_SO>
```

+ Thẻ đánh dấu (nút): là mẫu thông tin được đánh dấu bằng một cặp thẻ, thẻ mở và thẻ đóng.

- Có thể chứa các thuộc tính được đặt trong thẻ bắt đầu ngay sau tên thẻ.
- Giá trị của thuộc tính phải được đặt trong dấu nháy đơn hoặc dấu nháy kép.
- Mỗi tên thuộc tính chỉ được xuất hiện một lần trong mỗi thẻ chỉ thị.
- Để gán giá trị cho thuộc tính, thường dùng dấu “=”.

Ví dụ 4: File XML chứa thông tin học viên:

```
<?xml version= "1.0" encoding = "utf-8">
<danhsach>
  <hoc_vien ma_hoc_vien= "001">
    <ho_ten>Nguyễn Minh Khang</ho_ten>
    <ngay_sinh>2000-02-12</ngay_sinh>
    <dia_chi>Tiền Giang</dia_chi>
  </hoc_vien>
</danhsach>
```

+ Thẻ rỗng: là thẻ không kèm theo dữ liệu.

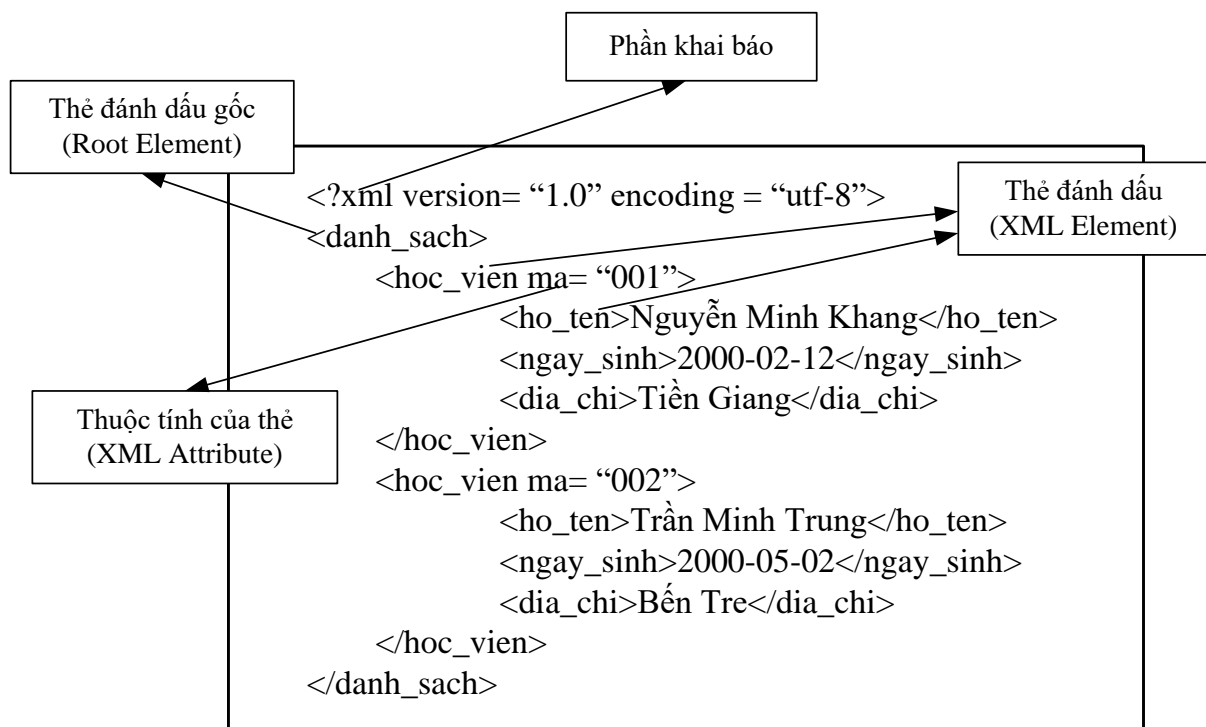
- Chỉ có duy nhất một thẻ.
- Chỉ có thẻ bắt đầu không có thẻ kết thúc.
- Ký hiệu thẻ <./>.
- Tất cả các thông tin được lưu trong các thuộc tính.

Ví dụ 5: dữ liệu XML có thẻ name là thẻ rỗng

```
<?xml version= "1.0" encoding = "utf-8">
<Danh_sach>
  <name TEXT= "Nguyễn Văn An" />
  <address>
    <street> 270 Phường 7</street>
    <city> Mỹ Tho </city>
  </address>
</Danh_sach>
```

- Quy tắc đặt tên các thẻ:
 - + Có thể chứa các ký tự, hay dấu “_”, không được bắt đầu bằng số hay dấu câu. Sau ký tự đầu tiên có thể dùng số và dấu “.”.
 - + Không được chứa ký tự khoảng trắng, dấu “:”.
 - + Không được bắt đầu bằng các ký tự: xml, Xml, XML.
 - + Không được chứa khoảng trắng ngay sau dấu mở thẻ “<”, nhưng có thể chứa ký tự khoảng trắng trước dấu đóng thẻ “>”.
 - + Tên thẻ có phân biệt chữ hoa, chữ thường.

Ví dụ 6:



Hình 5.1. Minh họa dữ liệu File XML.

- Quan hệ lồng nhau giữa các thẻ có nội dung:

Ví dụ 7: thẻ hoc_vien là thẻ cha của thẻ ho_ten và dia_chi:

```

<hoc_vien ma= "001">
  <ho_ten>Nguyễn Minh Khang</ho_ten>
  <ngay_sinh>2000-02-12</ngay_sinh>
  <dia_chi>Tiền Giang</dia_chi>
</hoc_vien>
  
```

3. Truy xuất tài liệu XML

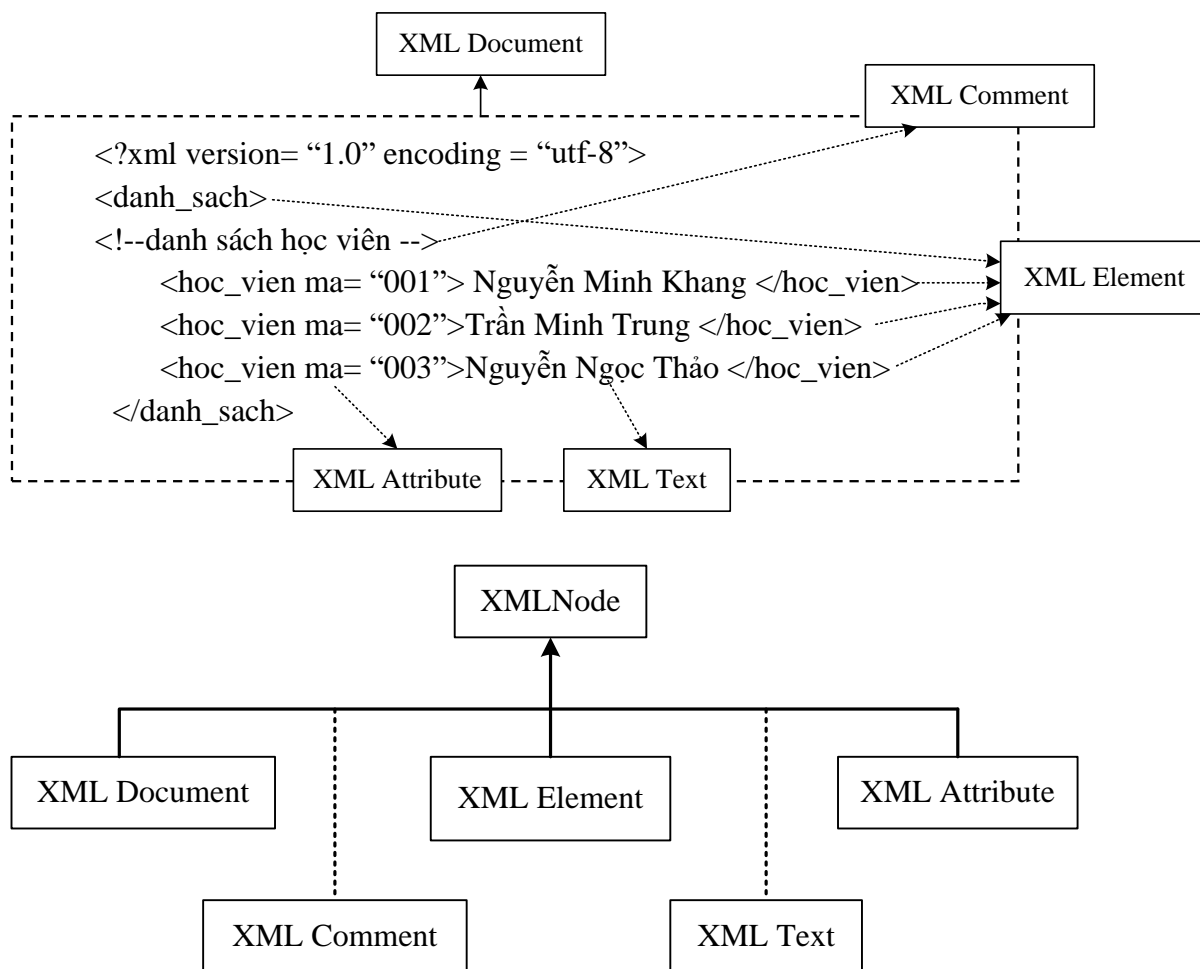
3.1. Giới thiệu

- Trong các ứng dụng .NET trên nền window, để truy xuất tài liệu XML có thể dùng mô hình DOM hoặc DataSet.

- Đối với các tài liệu XML lưu trữ dạng bảng giống với CSDL thì có thể dùng DataSet để đọc hay ghi dữ liệu dễ dàng.
- Mô hình DOM là mô hình chuẩn, đa số các ngôn ngữ lập trình đều hỗ trợ.

3.2. Các đối tượng của DOM

DOM – Document Object Model là mô hình truy xuất dữ liệu dạng XML, được định nghĩa bởi tổ chức W3C. Đây là mô hình đối tượng cho phép xử lý tài liệu XML từ các ngôn ngữ lập trình



Hình 5.2. Các đối tượng của DOM.

- XML Node: là Class cơ sở của hầu hết các Class trong DOM

Bảng 5.1. Các phương thức thường dùng của XML Node.

Tên phương thức	Mô tả
SelectNode	Lấy danh sách nút thỏa điều kiện.
SelectSingleNode	Lấy nút đầu tiên thỏa điều kiện.
AppendChild	Thêm 1 nút vào cuối danh sách.
InsertAfter	Thêm 1 nút vào danh sách, nút vừa thêm nằm sau nút chỉ định.
InsertBefore	Thêm 1 nút vào danh sách, nút vừa thêm nằm

Tên phương thức	Mô tả
	trước nút chỉ định
RemoveChild	Xóa 1 nút.
RemoveAll	Xóa tất cả các nút con và thuộc tính của nút.

- XmlDocument: đại diện cho tài liệu XML.

Bảng 5.2. Các phương thức thường dùng của XmlDocument.

Tên	Mô tả
DocumentElement	Nút gốc.
Load	Đọc tài liệu XML.
LoadXML	Đọc chuỗi XML.
GetElementById	Truy xuất nút thông qua thuộc tính Id.
GetElementsByTagName	Lấy danh sách nút thông qua tên nút.
CreateElement	Tạo một đối tượng XmlElement.
CreateTextNode	Tạo một đối tượng XmlText.
Save	Lưu tài liệu XML.

- XmlElement: đại diện cho một nút trong tài liệu XML.

Bảng 5.3. Các phương thức thường dùng của XmlElement.

Tên	Mô tả
GetElementsByTagName	Lấy danh sách nút theo tên nút.
GetAttribute	Lấy giá trị thuộc tính của nút.
SetAttribute	Gán giá trị cho thuộc tính của nút.
RemoveAttribute	Xóa một thuộc tính.

3.3. Các thao tác cơ bản trên DOM

- Khai báo:
 - + Khai báo sử dụng thư viện DOM: `using System.Xml;`
 - + Khai báo sử dụng tài liệu XML:
`XmlDocument Tai_lieu = new XmlDocument();`
- Đọc tài liệu XML:
`Tai_lieu.Load(Ten_tap_tin);`

Ví dụ 8: đọc tài liệu XML có tên danh_sach_hoc_vien.xml lưu trong thư mục Data:

```
//khai báo tài liệu XML
XmlDocument tai_lieu = new XmlDocument();
string duong_dan = @"Data\danh_sach_hoc_vien.xml";
//đọc tài liệu XML
tai_lieu.Load (duong_dan);
MessageBox .Show (tai_lieu.InnerXml); //kiểm tra đọc file XML
```

– Truy xuất nút gốc:

+ Truy xuất nút gốc thông qua thuộc tính DocumentElement của đối tượng tai_lieu.

DocumentElement có kiểu là XmlElement.

XmlElement nut_goc = tai_lieu.DocumentElement;

+ Truy xuất nút con đầu tiên:

XmlNode nut_con = nut_goc.FirstChild;

+ Truy xuất nút kế tiếp:

XmlNode nut_ke = nut_con.NextSibling;

+ Truy xuất nút trước đó:

XmlNode nut_truoc = nut_con.PreviousSibling;

+ Truy xuất nút con cuối cùng:

XmlNode nut_truoc = nut_goc.LastChild;

+ Truy xuất nút con thứ i:

XmlNode nut_con = nut_goc.ChildNodes[i];

+ Truy xuất các nút con trực tiếp qua tên nút:

XMLNodeList kq = SelectNodes("ten_nut_con");

+ Truy xuất thuộc tính của nút: sử dụng phương thức GetAttribute của nút:

string kq = nut_goc.GetAttribute("ten_thuoc_tinh");

Ví dụ 9: truy xuất các thuộc tính ho_ten, dia_chi của nút hoc_vien:

string ho_ten = hoc_vien.GetAttribute("ho_ten");

string dia_chi = hoc_vien.GetAttribute("dia_chi");

– Tạo nút mới:

+ XmlElement nut_con = tai_lieu.CreateElement("ten_nut");

– Thêm nút:

+ Nut_cha.AppendChild(nut_con);

– Xóa nút:

+ Nut_cha.RemoveChild(nut_con);

– Ghi tài liệu XML:

+ Tai_lieu.Save(duong_dan);

4. DataSet và XML

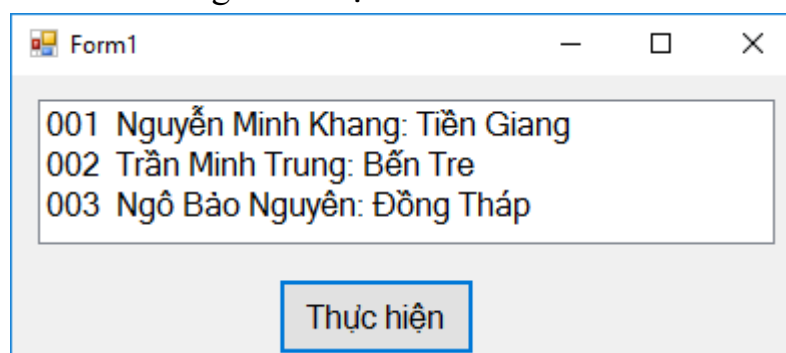
4.1. Đọc tài liệu XML

- Tạo đối tượng DataSet: `DataSet ds = new DataSet();`
- Đọc tài liệu XML: `ds.ReadXML(duong_dan);`

Ví dụ 10: thiết kế Form gồm 1 ListBox (lstOutput) và 1 nút Thực hiện, khi người dùng nhấp vào nút thực hiện thì thông tin học viên sẽ được hiển thị lên ListBox, sự kiện Click của nút thực hiện:

```
//khai báo tài liệu XML
XmlDocument tai_lieu = new XmlDocument();
string duong_dan = @"Data\danh_sach_hoc_vien.xml";
//đọc tài liệu XML
tai_lieu.Load(duong_dan);
DataSet ds = new DataSet();
ds.ReadXml(duong_dan);
DataTable dshv = ds.Tables[0];
foreach (DataRow row in dshv.Rows)
{
    string s = row["ma"] + row["ho_ten"] + ":" + row["dia_chi"];
    lstOutput.Items.Add(s);
}
```

Kết quả sau khi chương trình thực thi:



Hình 5.3. Minh họa đọc dữ liệu từ File XML.

4.2. Ghi tài liệu XML từ SQL Server

- Đặt tên DataSet (tên nút gốc của tài liệu XML).
`DataSet1.DataSetName = "danh_sach_hoc_vien";`
- Đặt tên cho các DataTable trong DataSet (tên các nút dòng dữ liệu).
`DataSet1.Tables[0].TableName = "hoc_vien";`
- Ghi DataSet ra File XML:
`DataSet1.WriteXML(duong_dan);`

Ví dụ 11: lấy CSDL bảng LOP từ CSDL Quản lý sinh viên của SQL Server thêm vào file XML.

Sự kiện Click của nút Thực hiện:

```
MoKetNoi();
DataSet ds = new DataSet();
string sql = "Select * from Lop";
SqlDataAdapter da = new SqlDataAdapter(sql, con);
da.Fill(ds);
//đặt tên cho DataSet
ds.DataSetName = "danh_sach_lop";
//đặt tên cho DataTable
ds.Tables[0].TableName = "Lop";
string duong_dan = @"Data\danh_sach_lop.xml";
ds.WriteXml(duong_dan);
DongKetNoi();

SqlConnection con = new SqlConnection();
public void MoKetNoi()
{
    string sqlCon = @"Data Source=DESKTOP-4SL47N2\SQLEXPRESS;
Initial Catalog=DLQLSV; User ID=sa; Password=12345678";
    con.ConnectionString = sqlCon;
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
}
public void DongKetNoi()
{
    if (con.State == ConnectionState.Open )
    {
        con.Close();
    }
}
```

5. Sử dụng XML trong ứng dụng Windows

5.1. Giới thiệu

Trong các ứng dụng Windows, XML thường được dùng:

- Lưu thông tin cấu hình của ứng dụng (file app.config).
- Lưu dữ liệu của ứng dụng:
 - + Nếu dữ liệu của ứng dụng ít, có thể dùng XML để lưu toàn bộ dữ liệu của ứng dụng.
 - + Thường dùng dữ liệu XML để lưu một phần dữ liệu của ứng dụng.

5.2. Sử dụng File cấu hình app.config

- Giới thiệu: trong ứng dụng Windows được tạo bởi VS có sẵn file XML cho phép lưu thông tin cấu hình của ứng dụng app.config và lớp tiện ích cho

phép dễ dàng truy xuất các thông tin cấu hình trong File app.config: Class ConfigurationManager.

– Thẻ connectionStrings:

- + Là thẻ trực tiếp của thẻ configuration trong File app.config.
- + Dùng để lưu các chuỗi kết nối CSDL.
- + Cú pháp thêm một chuỗi kết nối vào thẻ connectionStrings:

```
....  
<connectionStrings>  
    <add name = "ten_chuoi_ket_noi"  
        connectionString = "noi_dung_chuoi_ket_noi"/>  
....  
</connectionStrings>
```

– Thẻ appSettings:

- + Là thẻ con trực tiếp của thẻ configuration trong File app.config.
- + Dùng để lưu các tham số của ứng dụng như:
 - ✓ Các đường dẫn đến thư mục dữ liệu XML.
 - ✓ Các tham số liên quan đến cơ quan sở hữu ứng dụng.
 - ✓ Các tham số liên quan đến hiển thị như: Font chữ, kích cỡ Font, màu sắc,...

Ví dụ 12: để kết nối đến File dữ liệu XML là Khoa_SV.xml lưu trong thư mục Data:

```
<?xml version="1.0" encoding="utf-8" ?>  
<configuration>  
    <appSettings>  
        <add key ="duong_dan_du_lieu" value ="Data\Khoa_SV.xml"/>  
    </appSettings>  
</configuration>
```

– Class ConfigurationManager: đọc chuỗi kết nối, đọc tham số.

Ví dụ 13: để kết nối đến CSDL quan hệ QL_SinhVien lưu trong SQL Server:

```
<?xml version="1.0" encoding="utf-8" ?>  
<configuration>  
    <connectionStrings>  
        <add name="QL_SanPham"  
            connectionString="Data Source=DESKTOP-  
4SL47N2\SQLEXPRESS; Initial Catalog=Khoa_SV; Persist Security  
Info=True; User ID=sa; Password=12345678"/>  
    </connectionStrings>  
</configuration>
```

5.3. Sử dụng XML để lưu dữ liệu của ứng dụng

- Giới thiệu CSDL quản lý sinh viên của khoa.

Tạo File XML Khoa_SinhVien.xml để lưu trữ dữ liệu quản lý sinh viên khoa như bài 2 chương 3:

Minh họa dữ liệu File Khoa_SV.xml:

```
<?xml version= "1.0" encoding = "utf-8">
<dataroot>
<KHOA>
<MaKh>NN</MaKh>
<TenKhoa>Ngoại ngữ</TenKhoa>
  <SINHVIEN>
    <MaSV>SV01</MaSV>
    <HoTen>Trần Văn Triều</HoTen>
    <Phai>1</Phai>
    <NgaySinh>1995-01-01</NgaySinh>
    <NoiSinh>Tiền Giang</NoiSinh>
    <MaKh>NN</MaKh>
  </SINHVIEN>
  <SINHVIEN>
    <MaSV>SV02</MaSV>
    <HoTen>Nguyễn Thanh Mai</HoTen>
    <Phai>0</Phai>
    <NgaySinh>1995-10-25</NgaySinh>
    <NoiSinh>Long An</NoiSinh>
    <MaKh>NN</MaKh>
  </SINHVIEN>
</KHOA>
<KHOA>
<MaKh>SP</MaKh>
<TenKhoa>Sư Phạm</TenKhoa>
  <SINHVIEN>
    <MaSV>SV06</MaSV>
    <HoTen>Nguyễn Quỳnh Như</HoTen>
    <Phai>0</Phai>
    <NgaySinh>1995-09-10</NgaySinh>
    <NoiSinh>An Giang</NoiSinh>
    <MaKh>SP</MaKh>
  </SINHVIEN>
  <SINHVIEN>
    <MaSV>SV07</MaSV>
    <HoTen>Nguyễn Thiên Kim</HoTen>
    <Phai>0</Phai>
    <NgaySinh>1995-03-01</NgaySinh>
    <NoiSinh>Cần Thơ</NoiSinh>
    <MaKh>SP</MaKh>
  </SINHVIEN>
</KHOA>
```

```

<KHOA>
<MaKh>TH</MaKh>
<TenKhoa>Tin học</TenKhoa>
  <SINHVIEN>
    <MaSV>SV03</MaSV>
    <HoTen>Nguyễn Thanh Tuấn</HoTen>
    <Phai>1</Phai>
    <NgaySinh>1995-11-15</NgaySinh>
    <NoiSinh>Bến Tre</NoiSinh>
    <MaKh>TH</MaKh>
  </SINHVIEN>
  <SINHVIEN>
    <MaSV>SV04</MaSV>
    <HoTen>Dương Công Định</HoTen>
    <Phai>1</Phai>
    <NgaySinh>1995-11-21</NgaySinh>
    <NoiSinh>Vĩnh Long</NoiSinh>
    <MaKh>TH</MaKh>
  </SINHVIEN>
</KHOA>
<KHOA>
<MaKh>VL</MaKh>
<TenKhoa>Vật lý</TenKhoa>
  <SINHVIEN>
    <MaSV>SV05</MaSV>
    <HoTen>Dương Đông Nhi</HoTen>
    <Phai>0</Phai>
    <NgaySinh>1995-01-02</NgaySinh>
    <NoiSinh>Tiền Giang</NoiSinh>
    <MaKh>VL</MaKh>
  </SINHVIEN>
</KHOA>
</dataroot>

```

5.3.1. Đọc dữ liệu

+ Đọc dữ liệu XML vào DataSet:

```

duong_dan = ConfigurationManager.AppSettings["duong_dan_du_lieu"];
//khai báo tài liệu XML
tai_lieu = new XmlDocument();
tai_lieu.Load(duong_dan);
ds = new DataSet();
ds.ReadXml(duong_dan);

```

+ Đọc dữ liệu từ các bảng trong DataSet.

Ví dụ 14: sử dụng DataSet để đọc tài liệu XML Khoa_SV.xml và hiển thị danh sách Khoa trên ListBox, khi chọn 1 dòng trên ListBox thì thông tin chi tiết của khoa được hiển thị trên các TextBox, thông tin các sinh viên của khoa sẽ được hiển thị trên DataGridView, như hình bên dưới:

	Mã SV	Họ Tên	Phái	Ngày sinh	Nơi sinh
▶	SV03	Nguyễn Thanh Tuấn	1	1995-11-15...	Bến Tre
	SV04	Dương Công Định	1	1995-11-21...	Vĩnh Long
*					

Hình 5.4. Minh họa thao tác với dữ liệu XML.

Đọc dữ liệu bảng KHOA load lên ListBox (lstOutput) sau khi đã đọc dữ liệu từ File XML vào DataSet (ds):

```
DataTable dskhoa = ds.Tables["Khoa"];
lstOutput.DataSource = dskhoa;
lstOutput.DisplayMember = "TenKhoa";
lstOutput.ValueMember = "MaKH";
Sự kiện Click của ListBox:
dtgSinhVien.Rows.Clear();
string makh = lstOutput.SelectedValue.ToString();
//Lấy thông tin của khoa đang chọn load lên TextBox
DataRow[] kq = ds.Tables["Khoa"].Select("Makh='" + makh + "'");
DataRow dong = kq[0];
txtMaKhoa.Text = dong["makh"].ToString();
txtTenKhoa.Text = dong["TenKhoa"].ToString();
//Đọc dữ liệu Load lên DataGridView
DataRow [] SVs = ds.Tables["SinhVien"].Select("Makh='" + makh + "'");
int sd = SVs.Length;
```

```

for (int i = 0; i < sd ; i++)
{
    dtgSinhVien.Rows.Add(SVs[i]["MaSv"].ToString(),
SVs[i]["hoten"].ToString(), SVs[i]["Phai"].ToString(),
SVs[i]["NgaySinh"].ToString(), SVs[i]["NoiSinh"].ToString());
}

```

5.3.2. Thêm dữ liệu

Thực hiện các bước sau:

- + Tạo nút mới
- + Gán thông tin cho nút.
- + Thêm nút vào tài liệu.
- + Lưu tài liệu.

Ví dụ 15: thêm 1 khoa mới với giá trị mã khoa và tên khoa nhập từ TextBox txtMaKhoa và txtTenKhoa vào File Khoa_SV.xml:

```

//tạo nút mới - dòng mới
XmlNode khoa_ = tai_lieu.CreateElement("KHOA");
//tạo các nút con
XmlNode Makh_ = tai_lieu.CreateElement("MaKh");
XmlNode TenKhoa_ = tai_lieu.CreateElement("TenKhoa");
//thêm nút con vào nút mới
khoa_.AppendChild(Makh_);
khoa_.AppendChild(TenKhoa_);
//gán thông tin
khoa_.SelectSingleNode("MaKh").InnerText = txtMaKhoa.Text;
khoa_.SelectSingleNode("TenKhoa").InnerText = txtTenKhoa.Text;
XmlNode goc = tai_lieu.FirstChild;
goc.AppendChild(khoa_);
tai_lieu.Save(duong_dan);

```

5.3.3. Xóa dữ liệu

- + Tìm dòng cần xóa.
- + Thực hiện việc xóa.

Ví dụ 16: xóa khoa đang chọn trên ListBox (lstOutput):

```

//Lấy thông tin của khoa đang được chọn trên ListBox
string makh = lstOutput.SelectedValue.ToString();
//lấy nút đầu tiên - nút gốc
XmlNode goc = tai_lieu.FirstChild;
//lấy danh sách các nút KHOA - bảng KHOA
XmlNodeList dsKHOA = goc.SelectNodes("KHOA");
//duyệt từng nút trong danh sách các nút của KHOA
foreach (XmlNode kh in dsKHOA)
{
    //Tìm nút cần xóa
    if (kh.SelectSingleNode("MaKh").InnerText == makh )
    {

```

```

        //Tiến hành xóa
        goc.RemoveChild(kh);
    }
}
tai_lieu.Save(duong_dan);

```

5.3.4. Cập nhật dữ liệu

- + Tìm dòng cần cập nhật.
- + Tiến hành cập nhật.

Ví dụ 17: khi chọn một khoa trên ListBox thông tin chi tiết của khoa sẽ được hiển thị chi tiết trên các TextBox như hình minh họa ở ví dụ 13, nếu người dùng sửa nội dung TenKhoa và nhấp chuột vào nút Cập nhật thì tên khoa mới sẽ được cập nhập vào File XML (Khoa_SV.xml).

```

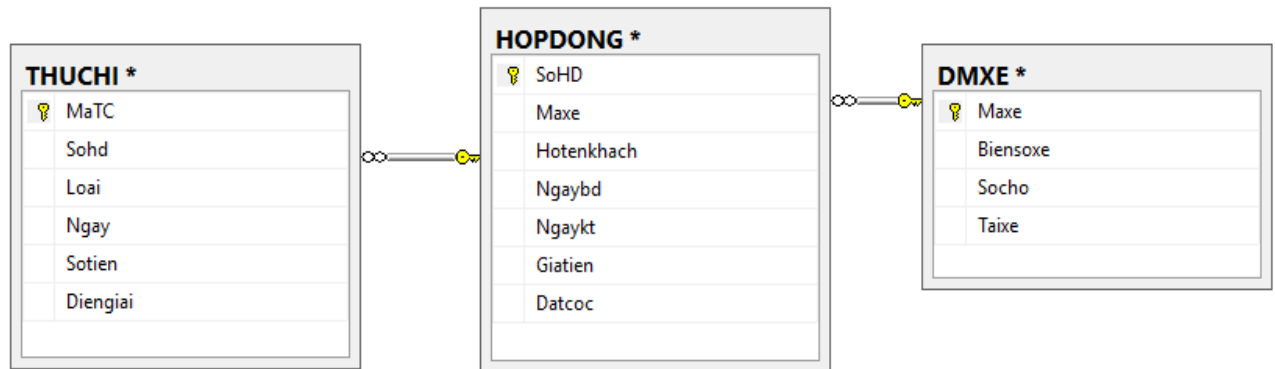
//Lấy thông tin của khoa đang được chọn trên ListBox
string makh = lstOutput.SelectedValue.ToString();
//lấy nút đầu tiên - nút gốc
XmlNode goc = tai_lieu.FirstChild;
//lấy danh sách các nút KHOA - bảng KHOA
XmlNodeList dsKHOA = goc.SelectNodes("KHOA");
//duyệt từng nút trong danh sách các nút của KHOA
foreach (XmlNode kh in dsKHOA)
{
    //Tìm nút cần cập nhật
    if (kh.SelectSingleNode("MaKh").InnerText == makh)
    {
        //Tiến hành cập nhật tên khoa
        kh.SelectSingleNode("TenKhoa").InnerText =
        txtTenKhoa.Text;
    }
}
tai_lieu.Save(duong_dan);

```


BÀI TẬP

Bài 1. Thực hiện lại bài ví dụ trong phần lý thuyết bằng cách lập trình theo mô hình 3 lớp.

Bài 2. Tổ chức lưu trữ dữ liệu bên dưới trong File hopdong.xml, thực hiện kết nối với hopdong.xml, và thực hiện các yêu cầu bên dưới:



Dữ liệu:

Maxe	Biensoxe	Socho	Taixe
1	63K-4013	4	Lý Đức Huy
2	63K-4082	30	Lê Đức
3	63K-5418	25	Trần Minh Hương
4	63K-5416	25	Nguyễn Văn Mẫn
5	63K-6090	50	Nguyễn Văn Giàu

SoHD	Maxe	Hotenkhach	Ngaybd	Ngaykt	Giatien	Datcoc
201001	1	Nguyễn Thế Anh	2010-01-01 00:00:00.000	2010-01-01 00:00:00.000	550000	100000
201002	2	Nguyễn Thị Ngọc Giàu	2010-01-23 00:00:00.000	2010-10-24 00:00:00.000	3000000	1000000
201003	5	Trần Văn Hoàng	2010-02-03 00:00:00.000	2010-02-03 00:00:00.000	1000000	400000
201004	2	Mai Thị Ngọc Nhân	2010-03-02 00:00:00.000	2010-03-02 00:00:00.000	1000000	200000
201005	3	Lý Thế Bằng	2010-02-01 00:00:00.000	2010-02-02 00:00:00.000	2000000	500000

MaTC	Sohd	Loai	Ngay	Sotien	Diengiai
PC001	201002	Chi	2010-12-02 00:00:00.000	400000	Xăng, dầu
PC002	201002	Chi	2010-12-03 00:00:00.000	567000	Thay vỏ xe
PC003	201003	Thu	2010-02-02 00:00:00.000	345000	Xăng dầu
PC004	201001	Chi	2009-12-12 00:00:00.000	450000	Thanh toán hợp đồng
PC005	201002	Thu	2009-12-23 00:00:00.000	2000000	Thanh toán hợp đồng
PC006	201003	Chi	2010-02-03 00:00:00.000	600000	Thanh toán hợp đồng
PC007	201004	Chi	2010-01-01 00:00:00.000	800000	Thanh toán hợp đồng
PC008	201005	Thu	2010-02-01 00:00:00.000	1500000	Thanh toán hợp đồng

HỢP ĐỒNG THUÊ XE

DANH SÁCH HỢP ĐỒNG:

Số hợp đồng	Họ tên khách hàng	Giá tiền
201001	Nguyễn Thế Anh	550000
201002	Nguyễn Thị Ngọc Giàu	3000000
201003	Trần Văn Hoàng	1000000
201004	Mai Thị Ngọc Nhân	1000000
201005	Lý Thế Bằng	2000000

THÔNG TIN CHI TIẾT HỢP ĐỒNG:

Số hợp đồng: 201002

Biển số xe: 63K-4082

Họ tên khách: Nguyễn Thị Ngọc Giàu

Ngày bắt đầu: 23/01/2010

Ngày kết thúc: 24/10/2010

Giá tiền: 3000000

Đặt cọc: 1000000

Sắp xếp giảm:

☐ Số hợp đồng
☐ Giá tiền

Sắp xếp

Thêm

Ghi

Không

Cập nhật

Thoát

THÔNG TIN THU CHI CỦA HỢP ĐỒNG:

	MaTC	Loai	Ngay	Sotien	Diengiai
▶	PC001	Chi	02/12/2010	400000	Xăng, dầu
	PC002	Chi	03/12/2010	567000	Thay vỏ xe
	PC005	Thu	23/12/2009	2000000	Thanh toán hợp đồng
*					

Tổng tiền thu: 2,000,000 VND

Tổng tiền chi: 967,000 VND

Yêu cầu:

- Thiết kế Form như trên.
- Trong Listview hiển thị danh sách các hợp đồng thông tin gồm: SoHD, HoTenKhach, GiaTien.
- Sắp xếp giảm theo hợp đồng, giá tiền.
- Khi nhấp vào 1 khách hàng sẽ hiển thị thông tin chi tiết của hợp đồng trên các textbox: SoHD, BienSoXe, HoTenKhach, NgayBD, NgayKT, GiaTien, DatCoc.

+ Đồng thời hiển thị thông tin thu chi của hợp đồng trên lưới. Thông tin trên lưới gồm: MaTC, Loai, Ngay, SoTien, DienGiai.

+ Tổng số tiền thu, tổng số tiền chi của hợp đồng.

- Khi nhấn nút **Thêm** cho phép các textbox xóa rỗng, **Số hợp đồng (SoHD)** tự động tăng, chỉ cho phép đọc không được phép chỉnh sửa. Nút **Ghi, Không** sáng. Nút **Thêm, Cập nhật** mờ.

Combobox biển số xe hiển thị danh sách các biển số xe cho phép chọn để thêm vào cơ sở dữ liệu (*giá trị hiển thị là BienSoXe, giá trị lưu là MaXe*).

- Nhấn **Ghi** cho phép lưu mẫu tin vào **table HopDong**. Nút **Ghi**, **Không** mờ. Nút **Thêm**, **Cập nhật** sáng.
- Nhấn **Không** bỏ qua thao tác thêm dữ liệu. Nút **Ghi**, **Không** mờ. Nút **Thêm**, **Cập nhật** sáng. Nhấn **Thoát** cho phép thoát khỏi Form.
- Nhấn nút **Cập nhật** cho phép cập nhật thông tin hợp đồng vào cơ sở dữ liệu.

Chương 6. CRYSTAL REPORT

Mục tiêu:

Sau khi học xong chương này, sinh viên có thể:

1. Kiến thức

- + Xác định các thành phần cơ bản của Report.
- + Phân biệt Report có phân nhóm, có tham số, có tính toán, thống kê theo từng nhóm.

2. Kỹ năng

- + Vận dụng linh hoạt các loại Report vào các ứng dụng có sử dụng Crystal Report.

3. Thái độ

- + Tự tin thiết kế các ứng dụng có chức năng kết xuất report.

1. Giới thiệu

Crystal Report là phần mềm thiết kế report chuyên nghiệp được tích hợp trong các phiên bản của Visual Studio.

Crystal Report là phần mềm tạo report độc lập với rất nhiều chức năng thiết kế Report và dịch vụ. Người dùng có thể kết nối với nhiều nguồn dữ liệu khác nhau. Report khi tạo ra được lưu trữ thành những tập tin .rpt độc lập, ở dạng dữ liệu hay không có dữ liệu. Sau đó, tập tin .rpt có thể kết hợp với các ứng dụng viết bằng Visual Basic.NET, C#,...

Crystal Report cung cấp các chức năng định dạng dữ liệu, phân nhóm, tính toán, thiết kế biểu đồ dựa trên nguồn dữ liệu lấy từ SQL Server, Access,...

Thông tin trong report thường do Tables hay Queries cung cấp. Các thông tin khác như tiêu đề, ngày tháng, số trang,... được lưu trong thiết kế của report.

Có thể liên kết giữa report và dữ liệu nguồn bằng cách dùng các Controls.

2. Xây dựng các chức năng của Report với Crystal Report

Crystal Report là công cụ thiết kế report cho phép nhận và định dạng dữ liệu từ cơ sở dữ liệu, tạo tính toán thống kê, thể hiện dữ liệu dưới dạng đồ thị, biểu đồ, ...

Các bước tạo report như sau:

- Bước 1: tạo mẫu report.
- Bước 2: thiết lập nguồn dữ liệu cho các report.
- Bước 3: tạo tập tin report và chọn dữ liệu nguồn cho report.
- Bước 4: thiết kế report sử dụng công cụ Crystal Report.
- Bước 5: hiển thị report bằng Crystal Report Viewer.

2.1. Các thành phần trong report

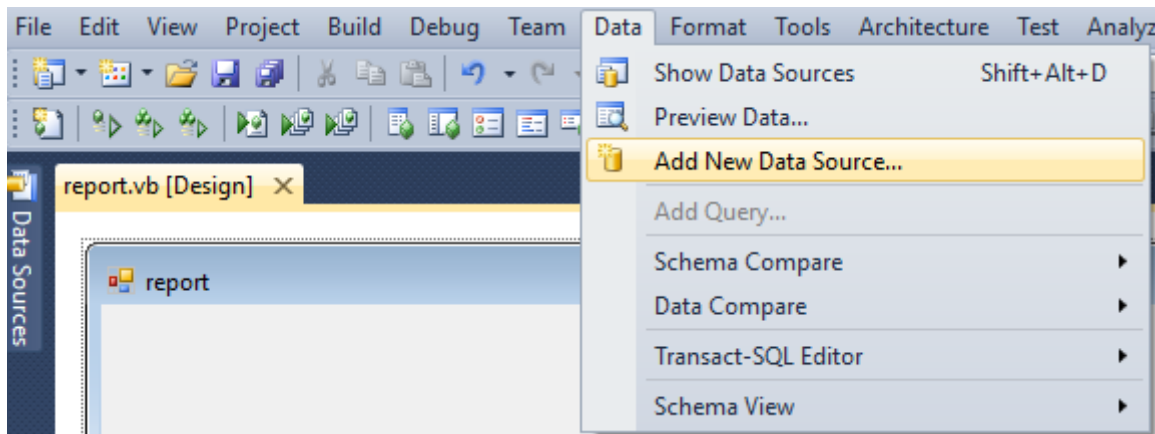
Bao gồm các thành phần sau :

- Report Header: tiêu đề report, thường xuất hiện ở đầu report, thường trình bày các phần như Logo, tiêu đề của report.
- Page Header: tiêu đề của trang, xuất hiện đầu mỗi trang, có thể trình bày các phần như: tiêu đề các cột, số trang,...
- Detail: phần dữ liệu của report, xuất hiện ở nhiều trang.
- Report Footer: chỉ xuất hiện ở cuối report, thường trình bày các phần như: ngày/tháng/năm, họ tên và người lập report,...
- Page Footer: xuất hiện cuối mỗi trang.
- GroupHeader: xuất hiện đầu mỗi nhóm dữ liệu trong các report cần thống kê, tính toán theo từng đối tượng thì tạo thêm Group.
- GroupFooter: xuất hiện dưới mỗi nhóm dữ liệu.

2.2. Thiết lập nguồn dữ liệu cho các report

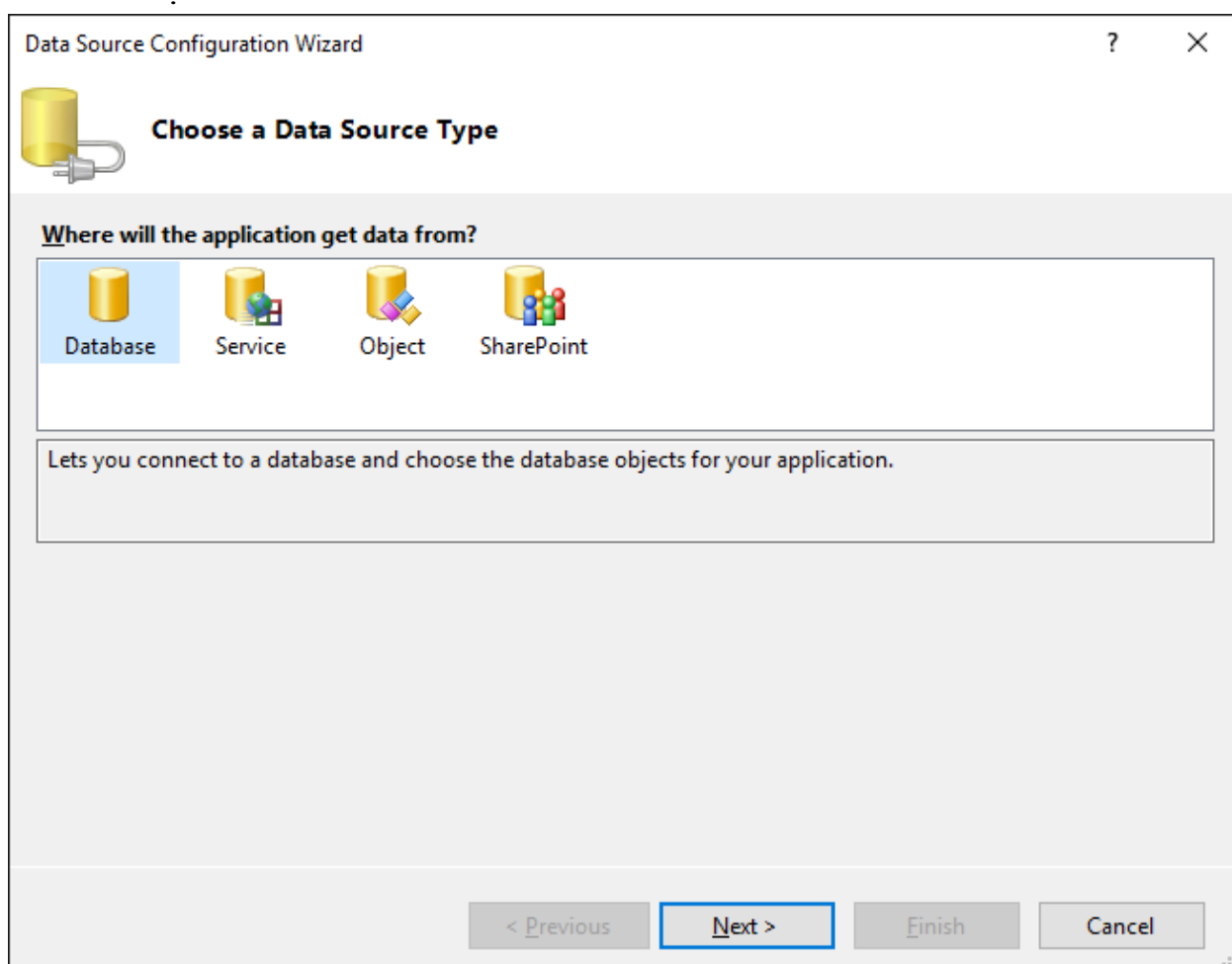
Các bước thực hiện:

- Từ VS.NET, Data → Add New DataSource.



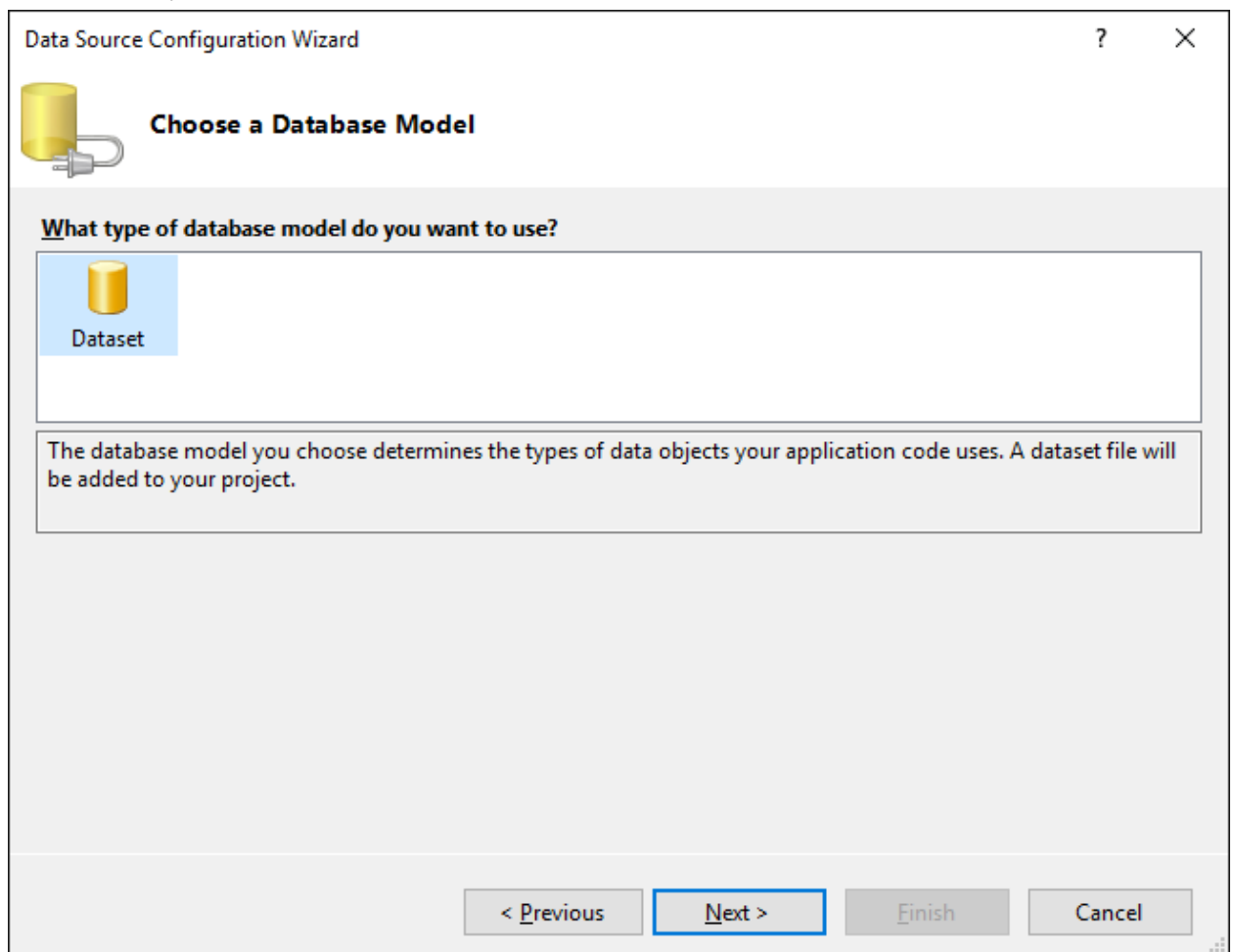
Hình 6.1. Minh họa thiết lập dữ liệu nguồn cho report.

– Chọn DataBase.



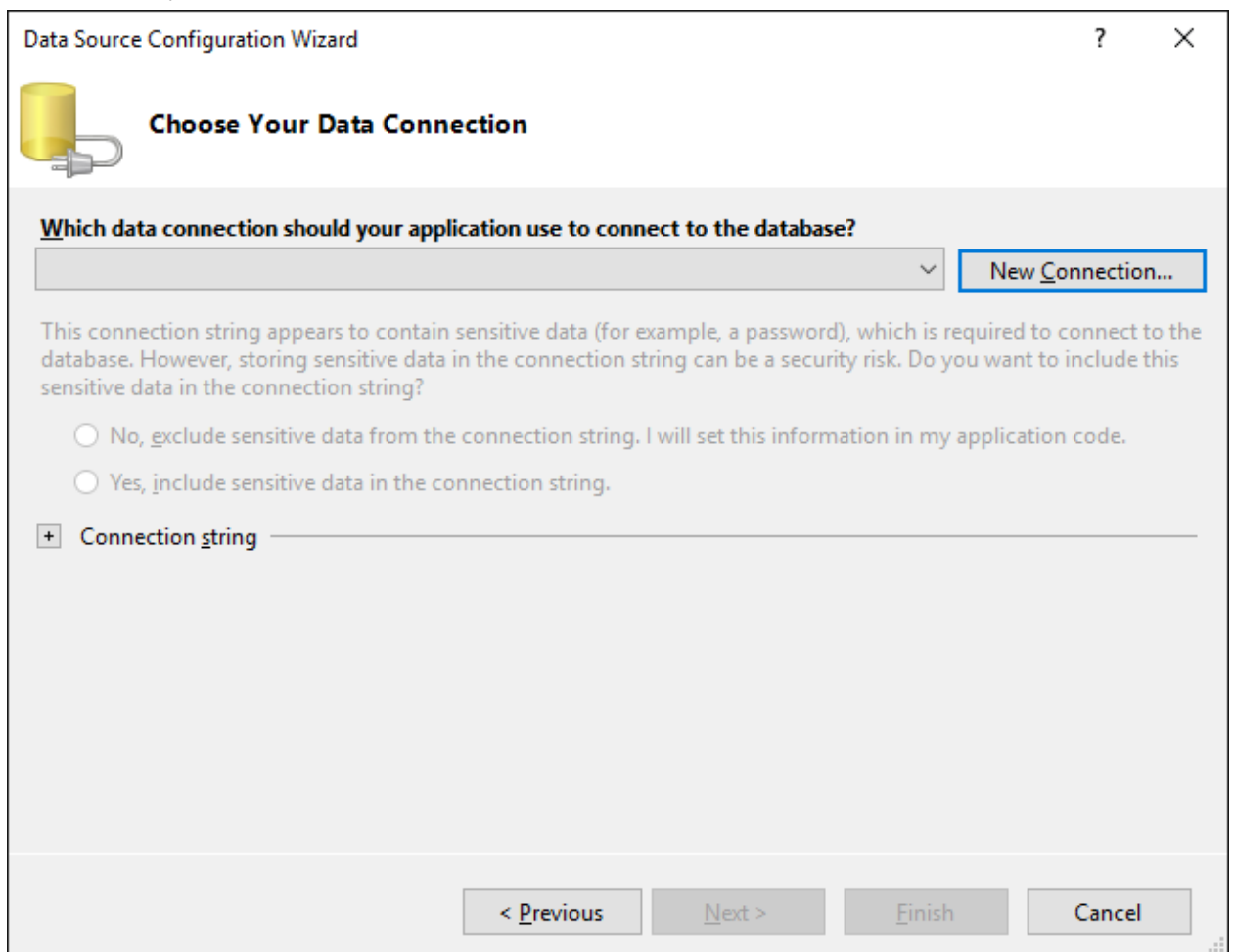
Hình 6.2. Minh họa thiết lập dữ liệu nguồn cho report.

– Chọn DataSet.



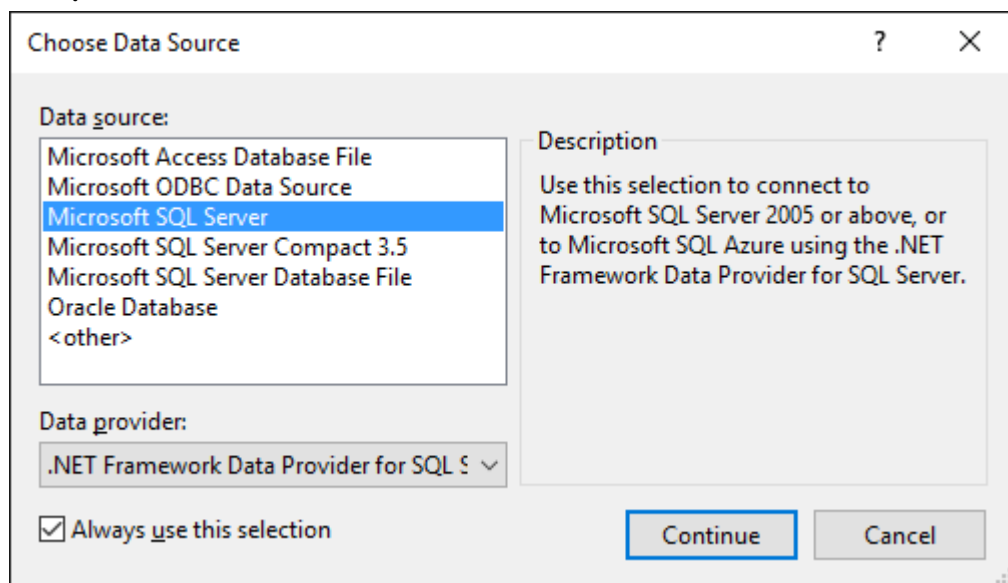
Hình 6.3. Minh họa thiết lập dữ liệu nguồn cho report.

– Chọn Next.



Hình 6.4. Minh họa thiết lập dữ liệu nguồn cho report.

– Chọn New Connection...



Hình 6.5. Minh họa thiết lập dữ liệu nguồn cho report.

- Chọn Microsoft SQL Server

Add Connection

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source:
Microsoft SQL Server (SqlClient) Change...

Server name:
DESKTOP-4SL47N2\SQL Refresh

Log on to the server

☒ Use Windows Authentication
☐ Use SQL Server Authentication

User name:
Password:
☐ Save my password

Connect to a database

☒ Select or enter a database name:

☐ Attach a database file:
 Browse...
Logical name:

Advanced...

Test Connection OK Cancel

Hình 6.6. Minh họa thiết lập dữ liệu nguồn cho report.

- Nhập tên Server của SQL Server vào Server name → chọn Refresh để cập nhật tên của các Database.

Add Connection ? X

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source:
Microsoft SQL Server (SqlClient) Change...

Server name:
DESKTOP-4SL47N2\SQL Refresh

Log on to the server

☒ Use Windows Authentication
☐ Use SQL Server Authentication

User name:
Password:
☐ Save my password

Connect to a database

☒ Select or enter a database name:

☐ Attach a database file:
 Browse...
Logical name:

Advanced...

Test Connection OK Cancel

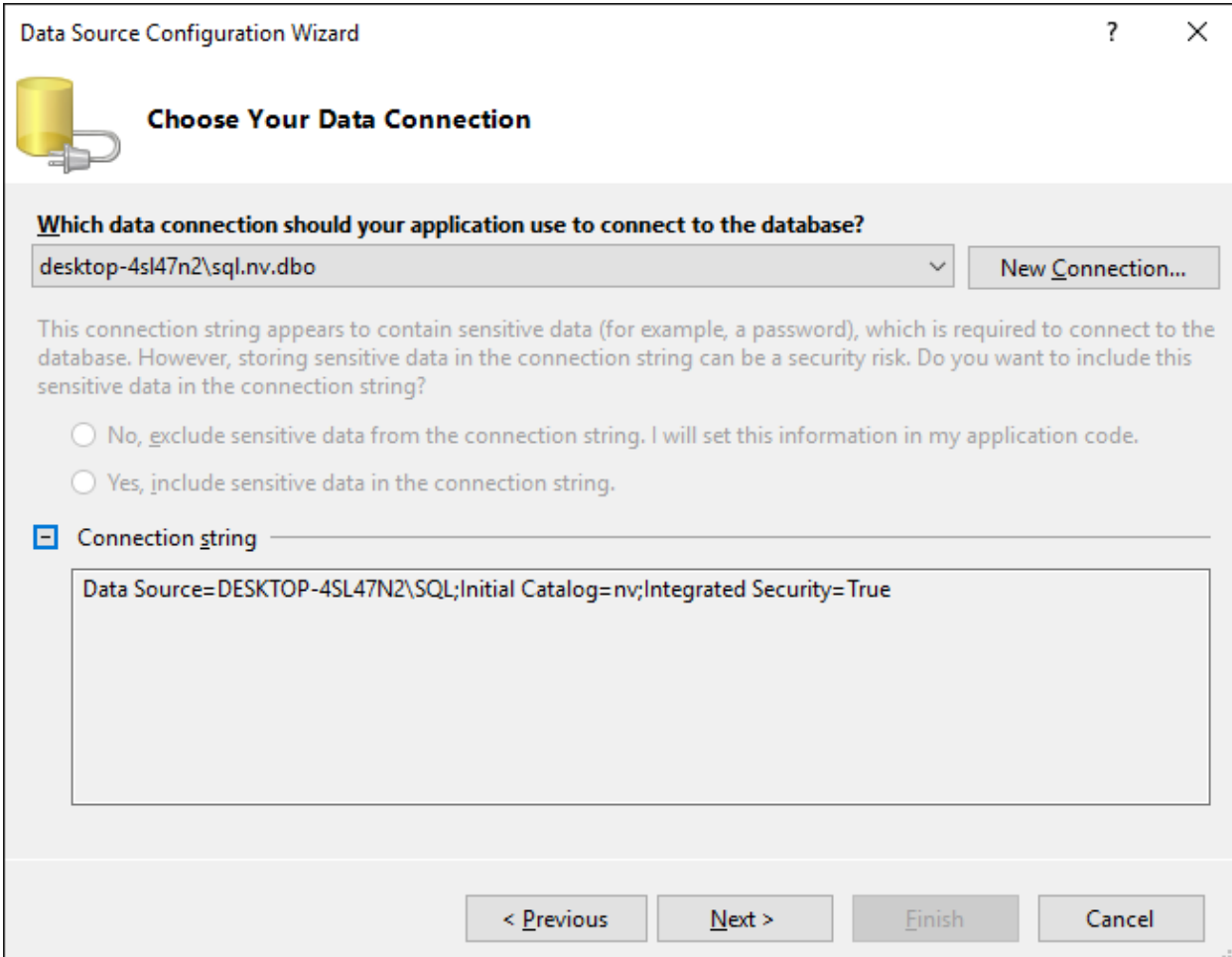
Hình 6.7. Minh họa thiết lập dữ liệu nguồn cho report.

- Chọn tên Database muốn kết nối tại mục Select or enter a database name.

The screenshot shows the 'Add Connection' dialog box. The 'Data source' is set to 'Microsoft SQL Server (SqlClient)'. The 'Server name' is 'DESKTOP-4SL47N2\SQL'. Under 'Log on to the server', 'Use Windows Authentication' is selected. Under 'Connect to a database', 'Select or enter a database name' is selected, and a dropdown menu is open showing a list of databases: master, model, msdb, nv (highlighted), ReportServer\$SQL, ReportServer\$SQLTempDB, and tempdb. Buttons at the bottom include 'Test Connection', 'OK', 'Cancel', and 'Advanced...'.

Hình 6.8. Minh họa thiết lập dữ liệu nguồn cho report.

– Chọn OK.



The screenshot shows the 'Data Source Configuration Wizard' window. The title bar includes a question mark and a close button. The main heading is 'Choose Your Data Connection' with a yellow plug icon. Below this, a question asks which data connection to use. A dropdown menu shows 'desktop-4sl47n2\sql.nv.dbo'. To the right is a 'New Connection...' button. A paragraph explains that the connection string might contain sensitive data like a password and asks if it should be included. Two radio buttons are present: 'No, exclude sensitive data from the connection string. I will set this information in my application code.' and 'Yes, include sensitive data in the connection string.' The 'No' option is selected. Below the radio buttons is a checkbox labeled 'Connection string' which is also checked. A text box below the checkbox contains the connection string: 'Data Source=DESKTOP-4SL47N2\SQL;Initial Catalog=nv;Integrated Security=True'. At the bottom are four buttons: '< Previous', 'Next >', 'Finish', and 'Cancel'.

Data Source Configuration Wizard

Choose Your Data Connection

Which data connection should your application use to connect to the database?

desktop-4sl47n2\sql.nv.dbo

New Connection...

This connection string appears to contain sensitive data (for example, a password), which is required to connect to the database. However, storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

☐ No, exclude sensitive data from the connection string. I will set this information in my application code.

☐ Yes, include sensitive data in the connection string.

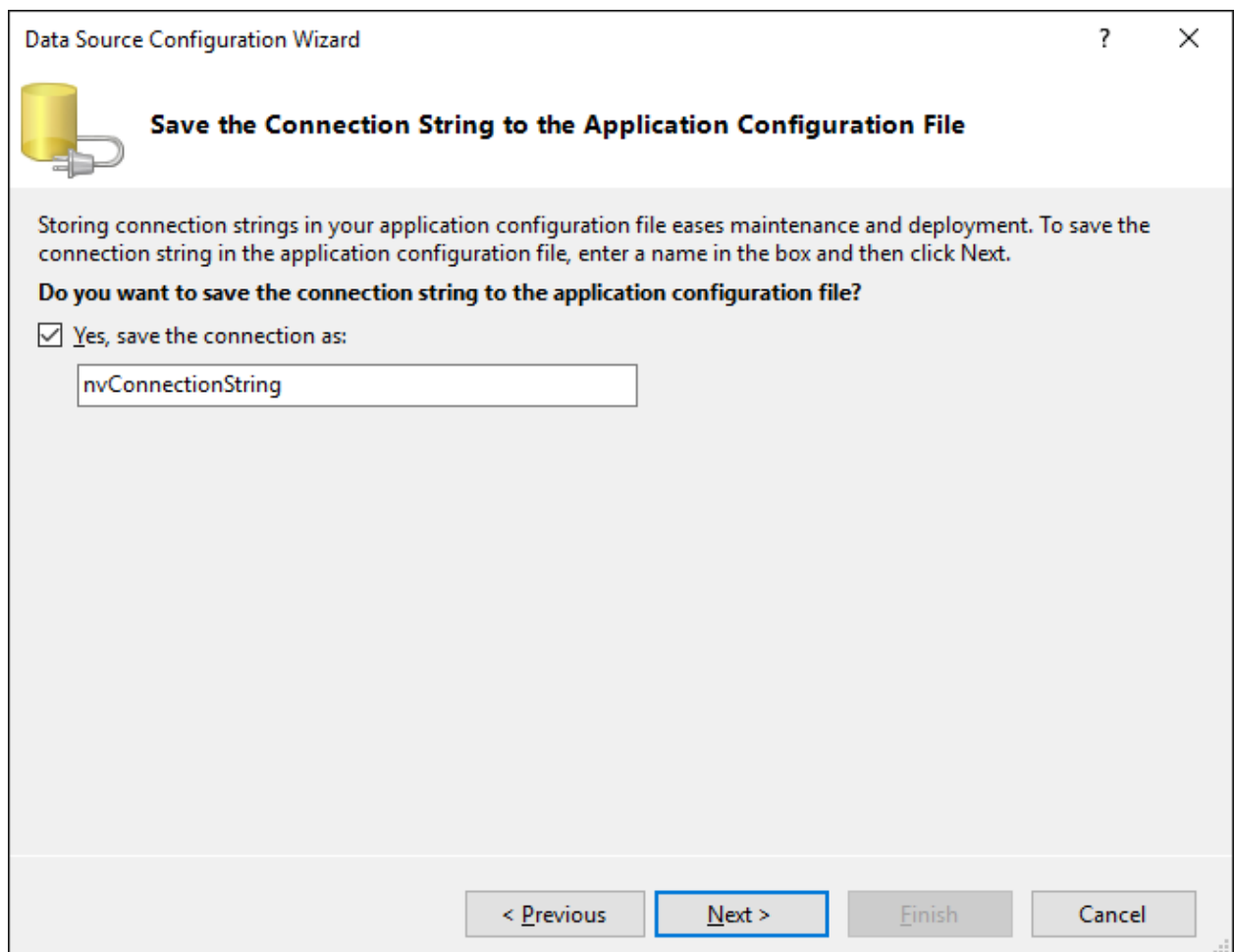
☒ Connection string

Data Source=DESKTOP-4SL47N2\SQL;Initial Catalog=nv;Integrated Security=True

< Previous Next > Finish Cancel

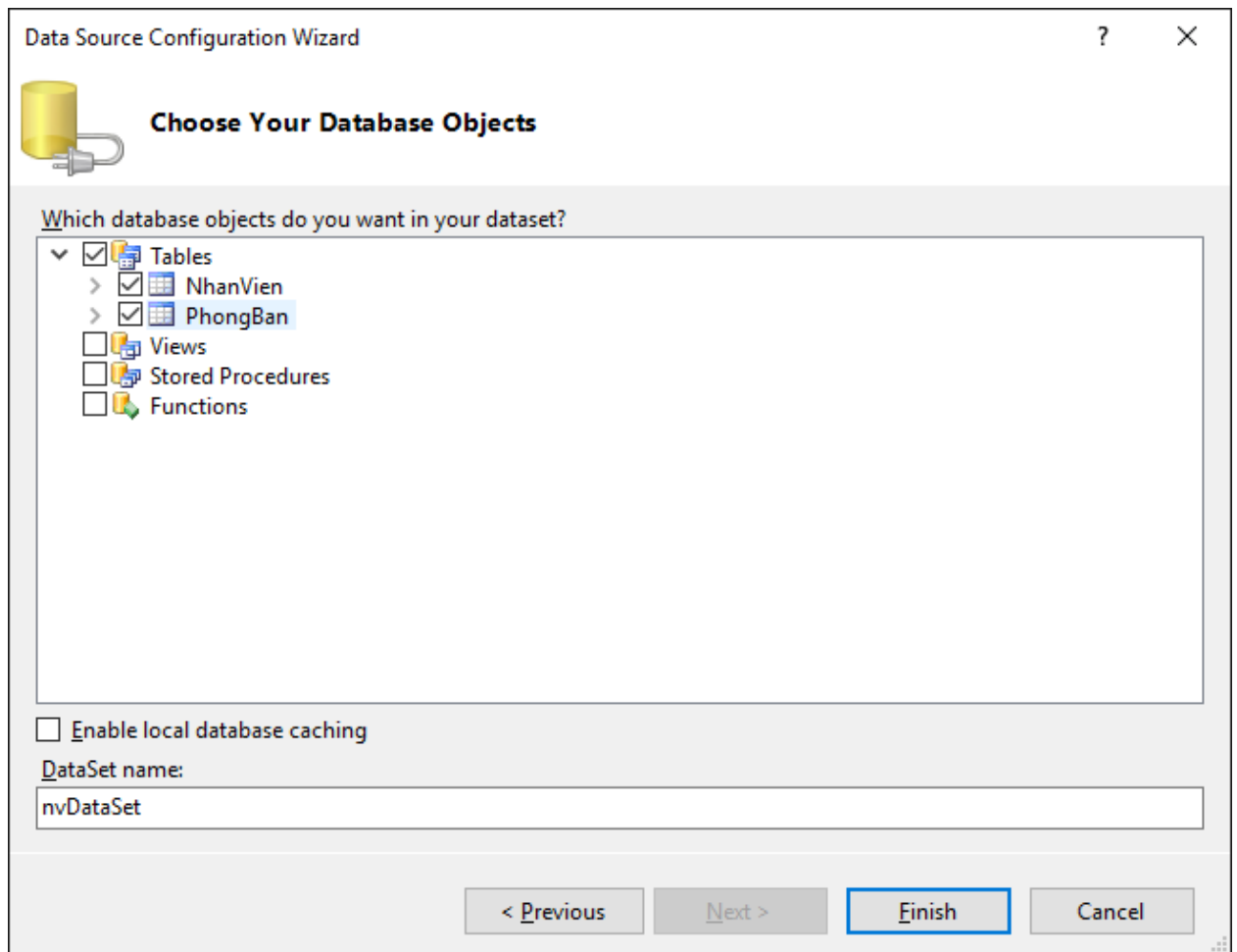
Hình 6.9. Minh họa thiết lập dữ liệu nguồn cho report.

– Chọn Next.



Hình 6.10. Minh họa thiết lập dữ liệu nguồn cho report.

- Chọn Next.

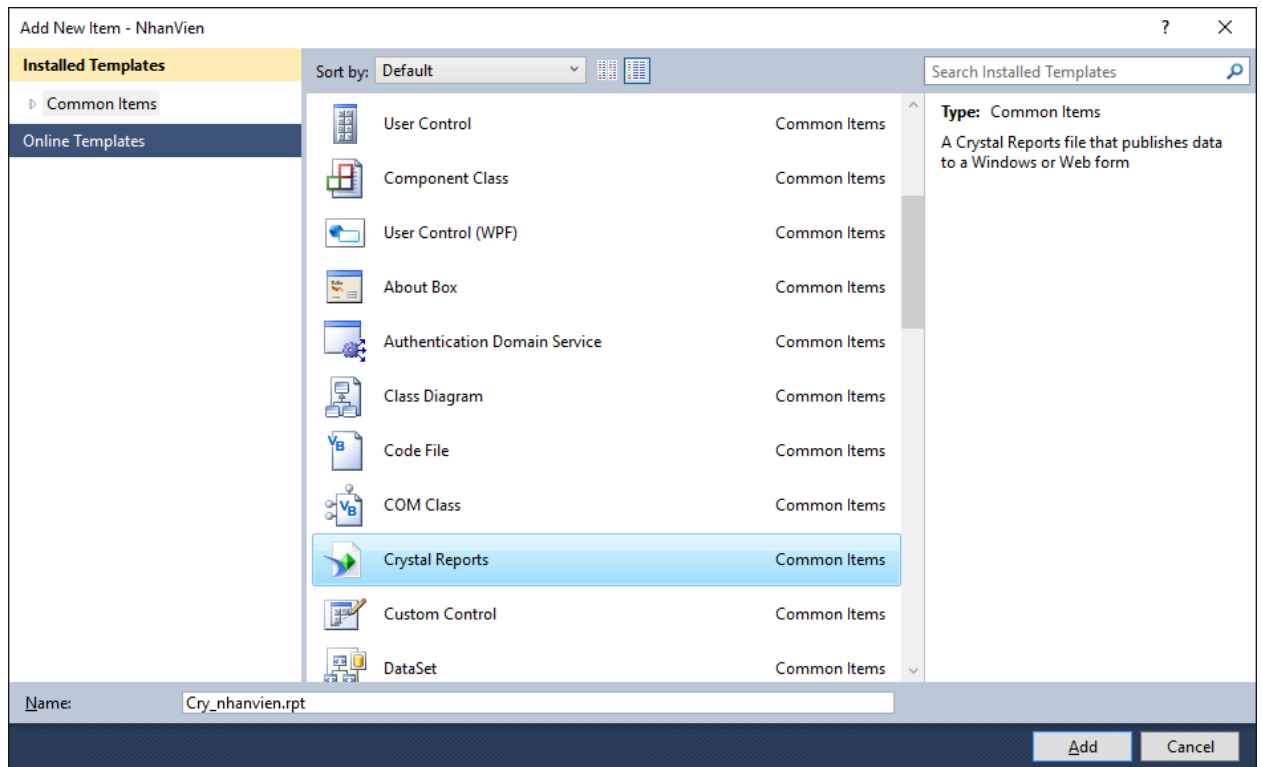


Hình 6.11. Minh họa thiết lập dữ liệu nguồn cho report.

- Chọn những Table cần lấy dữ liệu và đặt tên DataSet vào mục DataSet name.

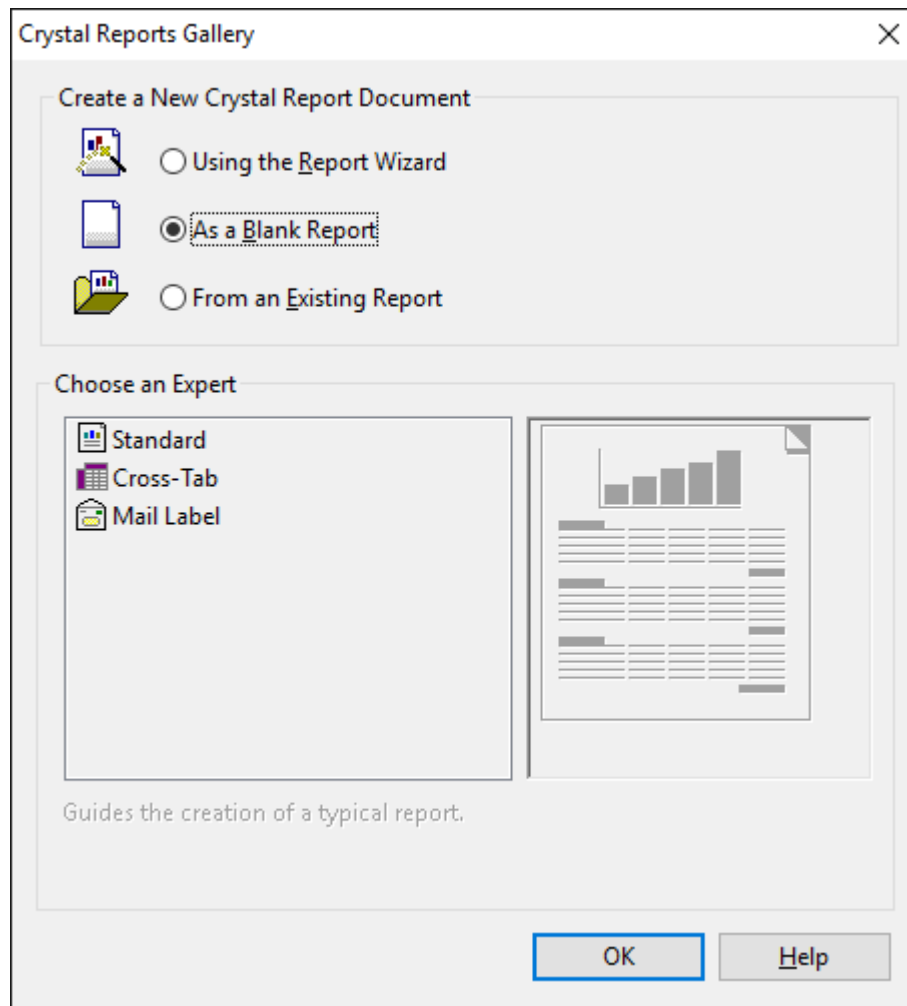
2.2.1. Xây dựng chức năng report

- Nhấp chuột phải vào tên Project → Add → New Items... → chọn loại tập tin là Crystal Report.



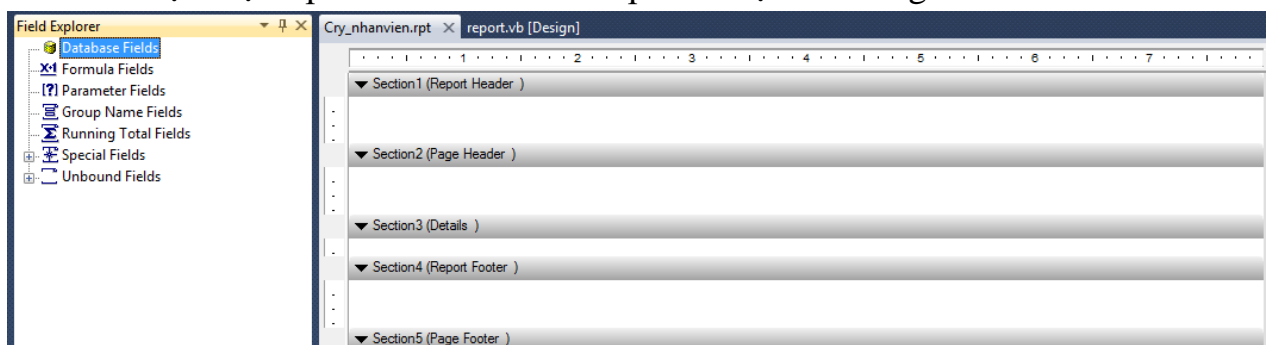
Hình 6.12. Minh họa xây dựng chức năng report.

– Chọn Add.



Hình 6.13. Minh họa xây dựng chức năng report.

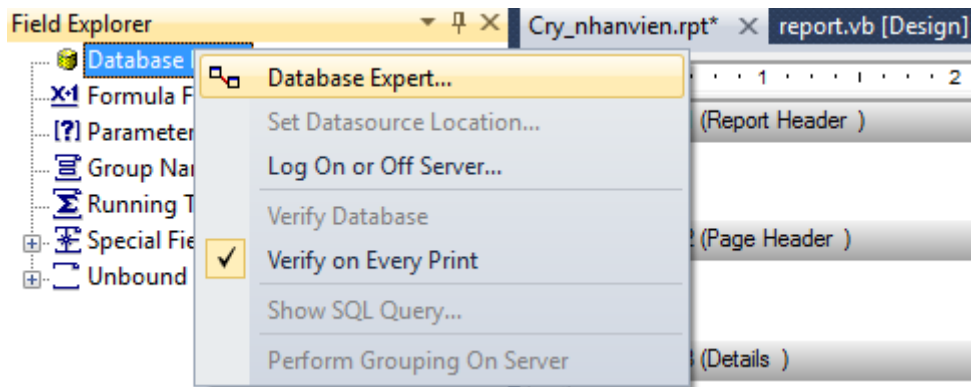
- Chọn loại report là As a Blank Report để tạo thủ công.



Hình 6.14. Minh họa xây dựng chức năng report.

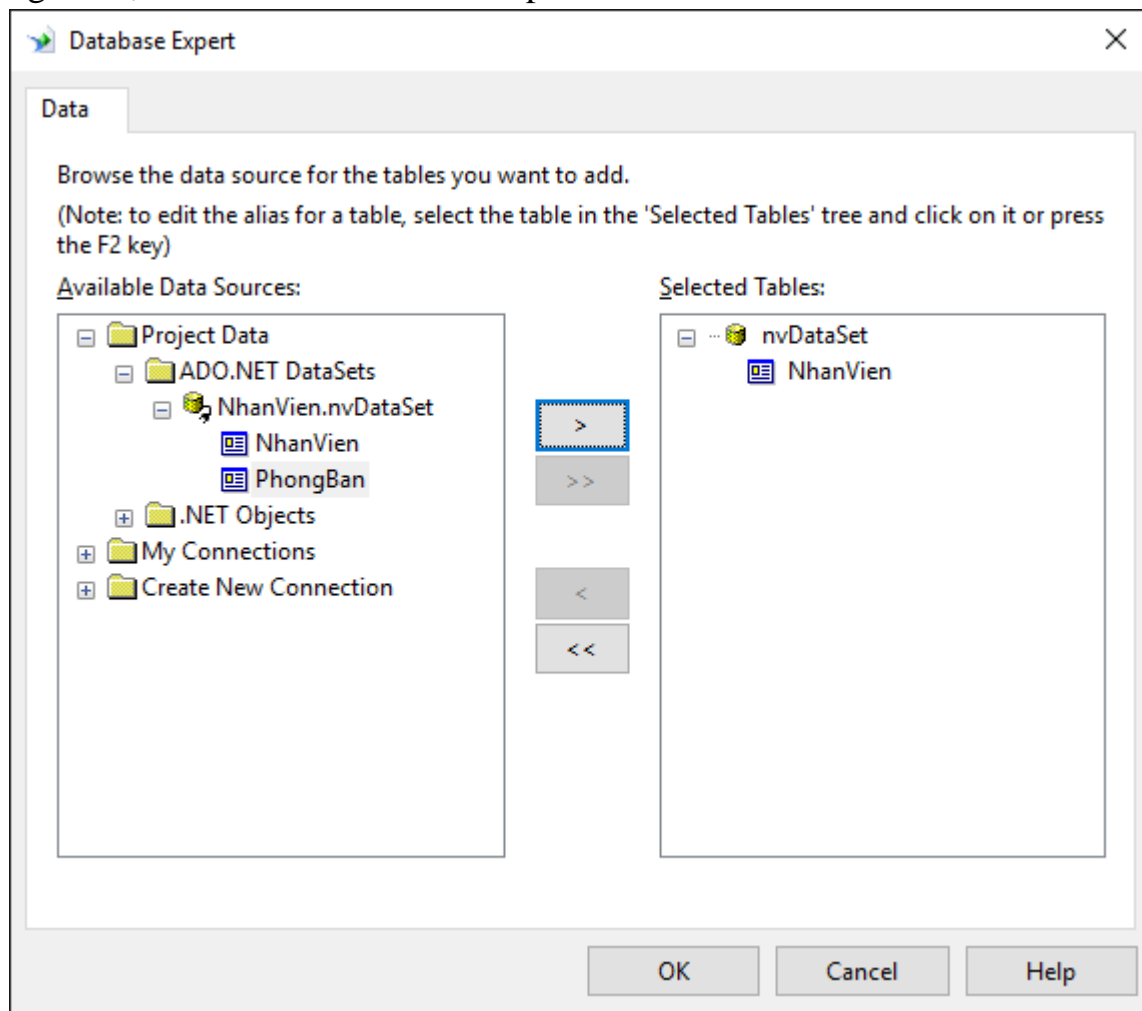
2.2.2. Thiết kế report

- Nhấp chuột phải vào Database Fields → chọn Database Expert...



Hình 6.15. Minh họa thao tác thiết kế report.

– Chọn các nguồn dữ liệu cho report. Mỗi Report có thể sử dụng các nguồn dữ liệu khác nhau như các bảng dữ liệu hoặc Store Procedure. Ở đây chọn bảng dữ liệu NhanVien để thiết kế report danh sách nhân viên.



Hình 6.16. Minh họa thao tác thiết kế report.

– Thiết kế report theo mẫu bằng cách kéo các đối tượng dữ liệu, các TextObject, LineObject, BoxObject vào report.

– Phần nội dung report có thể sử dụng các đối tượng dữ liệu sau:

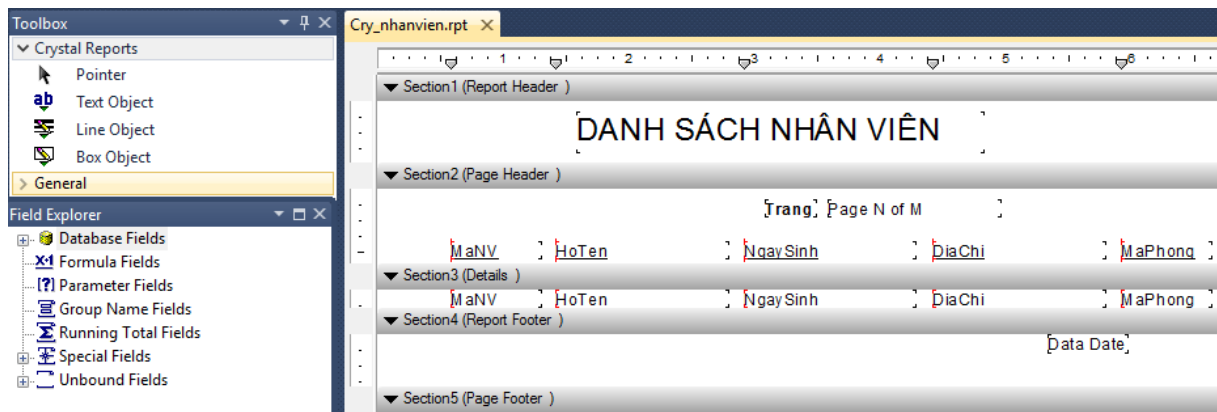
- + Database Fields: là các trường dữ liệu trong các bảng hoặc Stored Procedures.
- + Groupname Fields: sử dụng khi report có phân nhóm dữ liệu.
- + Special Fields: một số trường dữ liệu đặc biệt như số trang, ngày in report, số thứ tự, tổng số trang,...
- + Formular Fields: một số trường được xây dựng từ các công thức. Ví dụ: có thể xuất ra năm nếu đã có trường NgaySinh trong Database.
- + Running total Fields: là các trường dữ liệu để tính toán tổng hợp. Ví dụ: tính số lượng dòng của mỗi nhóm, tính tổng lương nhân viên của mỗi nhóm hay toàn bộ,...
- + Parametes Fields: có thể tạo report có tham số, khi đó nội dung report sẽ hiển thị theo tham số truyền vào. Ví dụ: báo cáo lương theo một tháng/năm nào đó.

Các bước thiết kế report danh sách nhân viên:

- + Kéo một đối tượng TextObject vào phần Report Header là tiêu đề của báo cáo.
- + Kéo các trường dữ liệu Database Fields vào phần Details. Khi kéo trường dữ liệu vào phần Details thì các đối tượng TextObject sẽ tự động được tạo ra trong phần Page Header là tiêu đề của các trường dữ liệu đó, có thể thay đổi lại tiêu đề cho phù hợp.
- + Trong mục Special Fields: **Page N of M** vào phần Page Header, **Data Date** vào phần Report Footer.
- + Có thể vẽ thêm các đối tượng Line Object, Box Object để tạo các đường viền cho các ô trong report.

Lưu ý:

- + Các thành phần trong Section 3 (Details) sẽ được lặp đi lặp lại.
- + Các thành phần trong Section 2 (Page Header) và Section 5 (Page Footer) sẽ được lặp đi lặp lại mỗi trang.
- + Các thành phần trong Section 1 (Report Header) và Section 4 (Report Footer) sẽ xuất hiện trong phần đầu tiên và cuối report.



Hình 6.17. Minh họa thao tác thiết kế report.

2.2.3. Hiển thị dữ liệu bằng Crystal Report Viewer

Chọn điều khiển CrystalReportViewer trong tab Reporting của thanh công cụ Toolbox và kéo vào Form muốn hiển thị report, đặt tên cho điều khiển.

Viết mã nguồn để hiển thị report trong sự kiện Load của Form.

```
Dim dt As New DataTable
Dim rp As New Cry_nhanvien
dt = nv.TTNhanVien
rp.SetDataSource(dt)
Me.CrystalReportViewer1.ReportSource = rp
```

DANH SÁCH NHÂN VIÊN					
Trang Page 1 of 1					
MaNV	HoTen	NgaySinh		DiaChi	MaPhong
NV0001	Nguyễn Văn A	01/12/1987	12:00:00AI	Tiên Giang	P001
NV0002	Trần Thanh Liêm	15/02/1980	12:00:00AI	Long An	P001
NV0003	Võ Thanh Hiền	25/07/1988	12:00:00AI	Bến Tre	P002
NV0004	Nguyễn Thị Hoa	10/05/1981	12:00:00AI	Vĩnh Long	P002
NV0005	Nguyễn Chí Thanh	21/12/1980	12:00:00AI	Tiên Giang	P003
NV0006	Ngô Thanh Mộng	18/09/1982	12:00:00AI	Bến Tre	P003

12/04/2017

Hình 6.18. Minh họa thao tác thiết kế report.

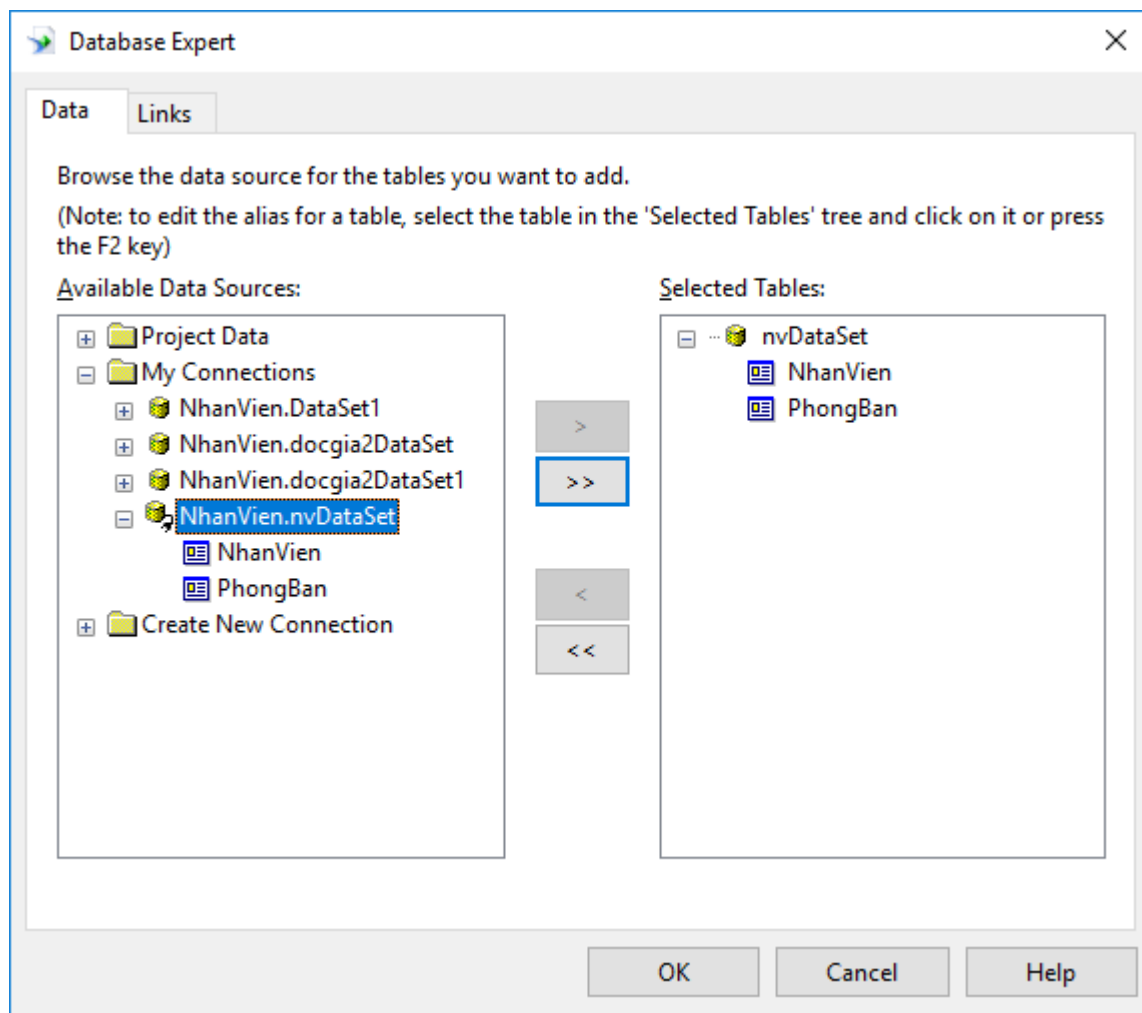
2.3. Xây dựng chức năng report có phân nhóm

2.3.1. Tạo tập tin report và chọn chức năng cho report

Nhấp chuột phải vào tên Project → Add → New Items... → chọn loại tập tin là Crystal Reports, đặt tên report.

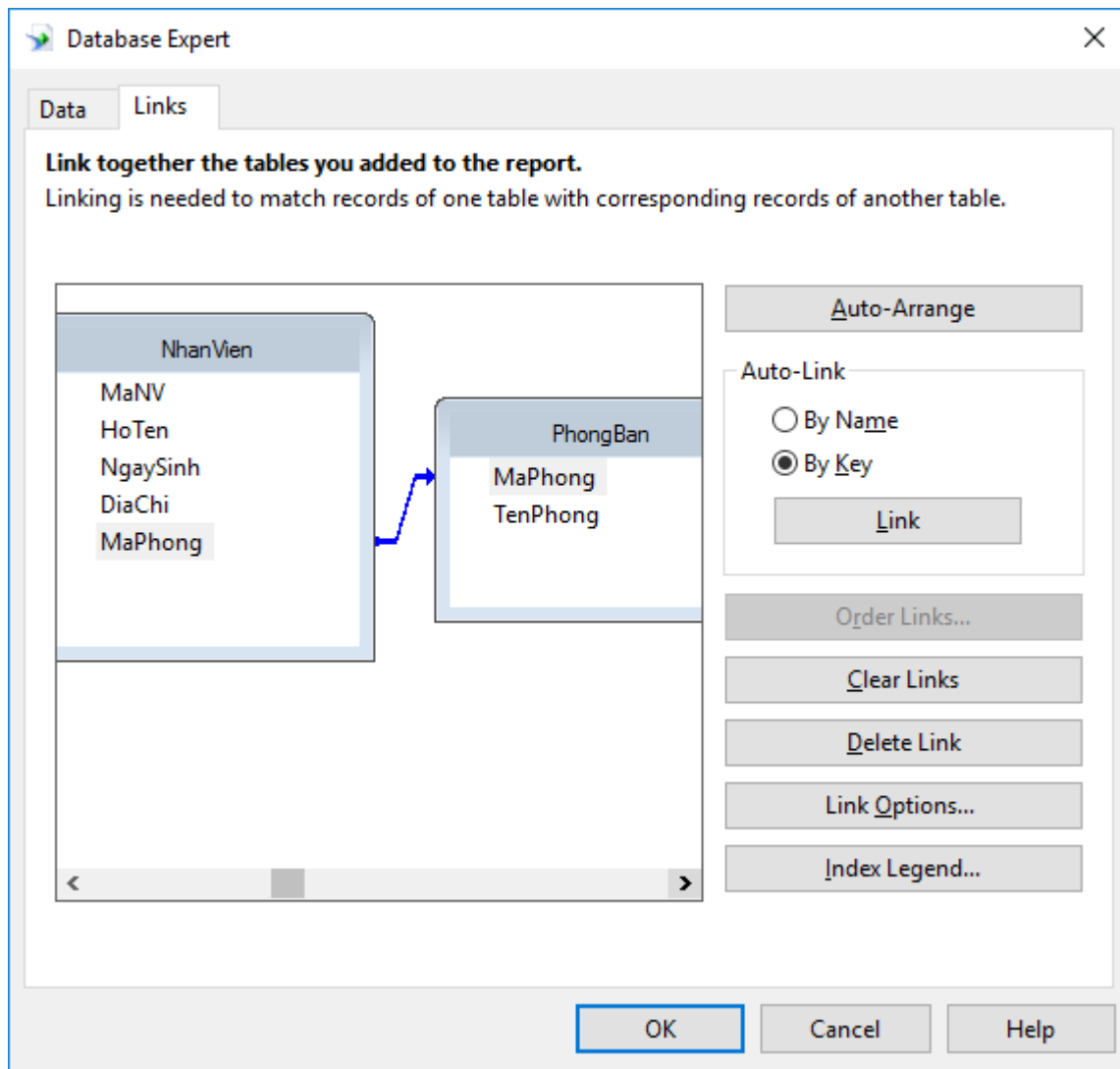
2.3.2. Thiết kế report

– Nhấp chuột phải Database Fields để chọn dữ liệu nguồn cho report.



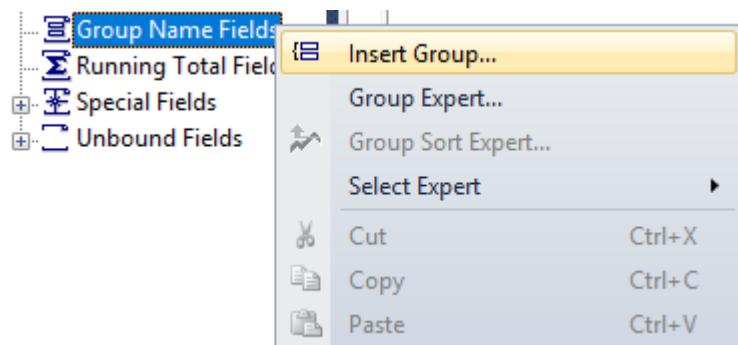
Hình 6.19. Chọn dữ liệu nguồn cho report.

– Chỉnh sửa liên kết của hai bảng này nếu cần thiết. Liên kết này để Crystal Report tự động kết các bảng dữ liệu để lấy thông tin giống như phép kết trong truy vấn.



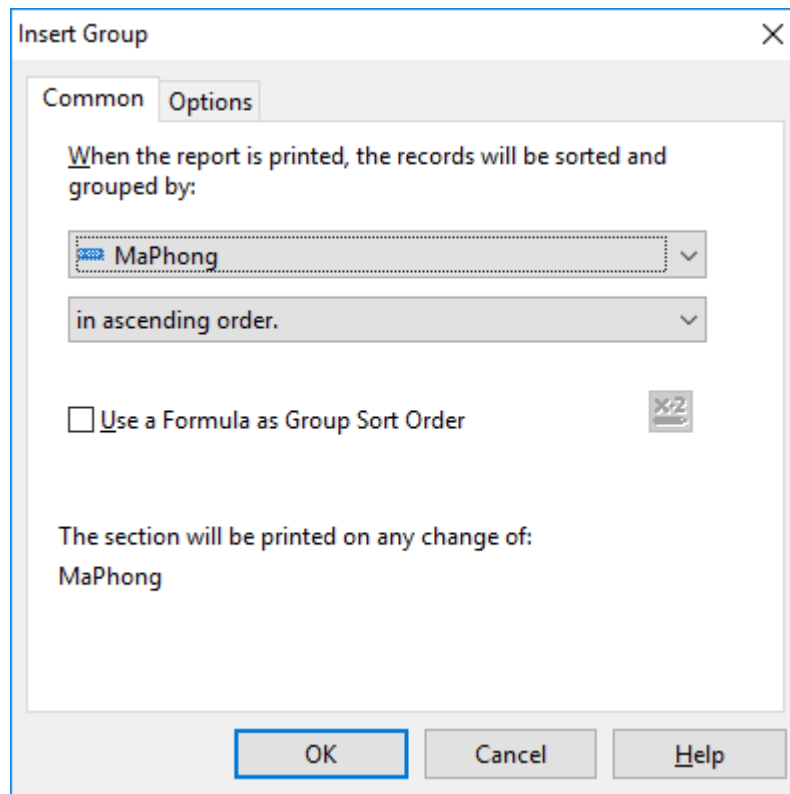
Hình 6.20. Tạo liên kết cho dữ liệu nguồn.

- Thêm các trường dữ liệu Group Name Fields cho report.
- Thêm nhóm cho report, nhấp chuột phải vào Group Name Fields → chọn Insert Group.



Hình 6.21. Thêm nhóm vào report.

- Chọn trường MaPhong để phân nhóm.

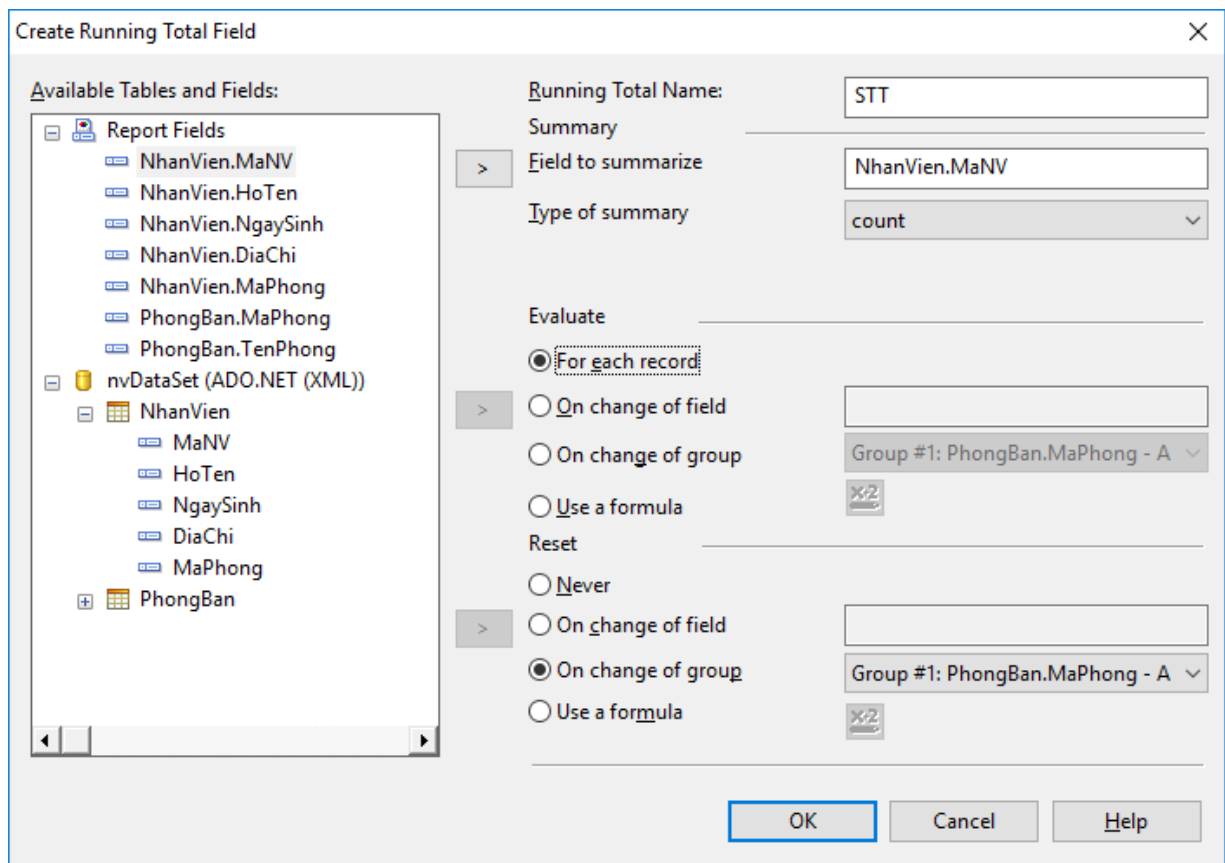


Hình 6.22. Minh họa chọn trường phân nhóm.

– Tạo số thứ tự mỗi nhóm, nhấp chuột phải Running Total Fields, thiết kế như hình bên dưới.

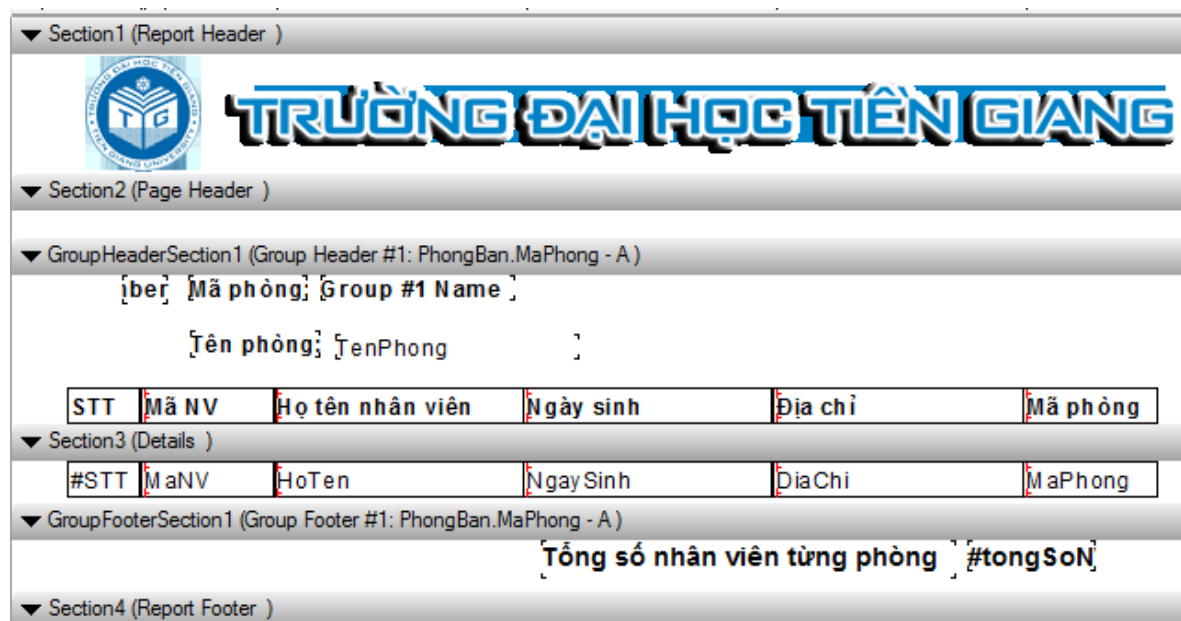
Running Total Name: STT thể hiện số thứ tự của nhân viên trong phòng, bắt đầu là 1. Khi sang phòng khác thì số thứ tự Reset lại là 1.

- + Chọn Field ton summarize là MaNV.
- + Type of summary chọn count.
- + Evaluate: chọn For each record: vì mỗi lần xử lý một Record thì tăng lên.
- + Reset chọn On change of group: vì mỗi lần sang nhóm mới số thứ tự là 1.



Hình 6.23. Minh họa tạo số thứ tự mỗi nhóm.

- Thống kê ở cuối mỗi nhóm. Ví dụ, đếm số lượng nhân viên của từng phòng.
- Thêm hình: nhấp chuột phải → Insert Picture → chọn đường dẫn lưu hình → chọn hình.



Hình 6.24. Minh họa thêm các trường vào report.

2.3.3. Hiện thị dữ liệu

– Trong thư mục GUI tạo 1 Form và kéo CrystalReportViewer vào Form để hiện thị dữ liệu của report.

– Viết mã lệnh trong sự kiện Form_Load của Form.

```
Dim sql As String
'Đưa dữ liệu bảng NhanVien, PhongBan vào DataSet
sql = "select * from NhanVien; select * from PhongBan"
Dim dtset As DataSet
dtset = csdl.ExceQuery1(sql)
'Khai báo biến Report
Dim rp As New Crys_DS_NV_theo_Phong
'Gán DataSet cho Report
rp.SetDataSource(dtset)
CrystalReportViewer1.ReportSource = rp
```

Với csdl được khai báo: Dim csdl As New KetNoiCSDL

– Trong Class KetNoiCSDL viết hàm ExceQuery1.

```
Public Function ExceQuery1(ByVal sql As String) As DataSet
MoKetNoiCSDL()
Dim adapter As New SqlDataAdapter(sql, con)
adapter.TableMappings.Add("Table", "NhanVien")
adapter.TableMappings.Add("Table1", "PhongBan")
Dim dtset As New nvDataSet
adapter.Fill(dtset)
DongKetNoiCSDL()
Return dtset
End Function
```

Bảng 6.1. Giải thích các lệnh truy vấn.

Lệnh	Diễn giải
<code>sql = "select * from NhanVien; select * from PhongBan"</code>	Câu truy vấn có thể lấy dữ liệu từ nhiều bảng như NhanVien, PhongBan và sau khi thực thi các bảng chứa trong DataAdapter lần lượt có tên mặc định là : Table, Table1,...
<code>adapter.TableMappings.Add("Table", "NhanVien") adapter.TableMappings.Add("Table1", "PhongBan")</code>	Ánh xạ Table, Table1 thành các bảng NhanVien và PhongBan để đưa vào sử dụng cho Report.
<code>adapter.Fill(dtset)</code>	Đưa toàn bộ dữ liệu của các bảng trong adapter sang DataSet.

Kết quả sau khi chương trình thực thi:



TRƯỜNG ĐẠI HỌC TIỀN GIANG

1 Mã phòng: P001

Tên phòng: Phòng kế hoạch

STT	Mã NV	Họ tên nhân viên	Ngày sinh	Địa chỉ	Mã phòng
1	NV0001	Nguyễn Văn A	01/12/1987 12:00:00A	Tiền Giang	P001
2	NV0002	Trần Thanh Liêm	15/02/1980 12:00:00A	Long An	P001

Tổng số nhân viên của phòng 2

2 Mã phòng: P002

Tên phòng: Phòng đào tạo

STT	Mã NV	Họ tên nhân viên	Ngày sinh	Địa chỉ	Mã phòng
1	NV0003	Võ Thanh Hiền	25/07/1988 12:00:00A	Bến Tre	P002
2	NV0004	Nguyễn Thị Hoa	10/05/1981 12:00:00A	Vĩnh Long	P002

Tổng số nhân viên của phòng 2

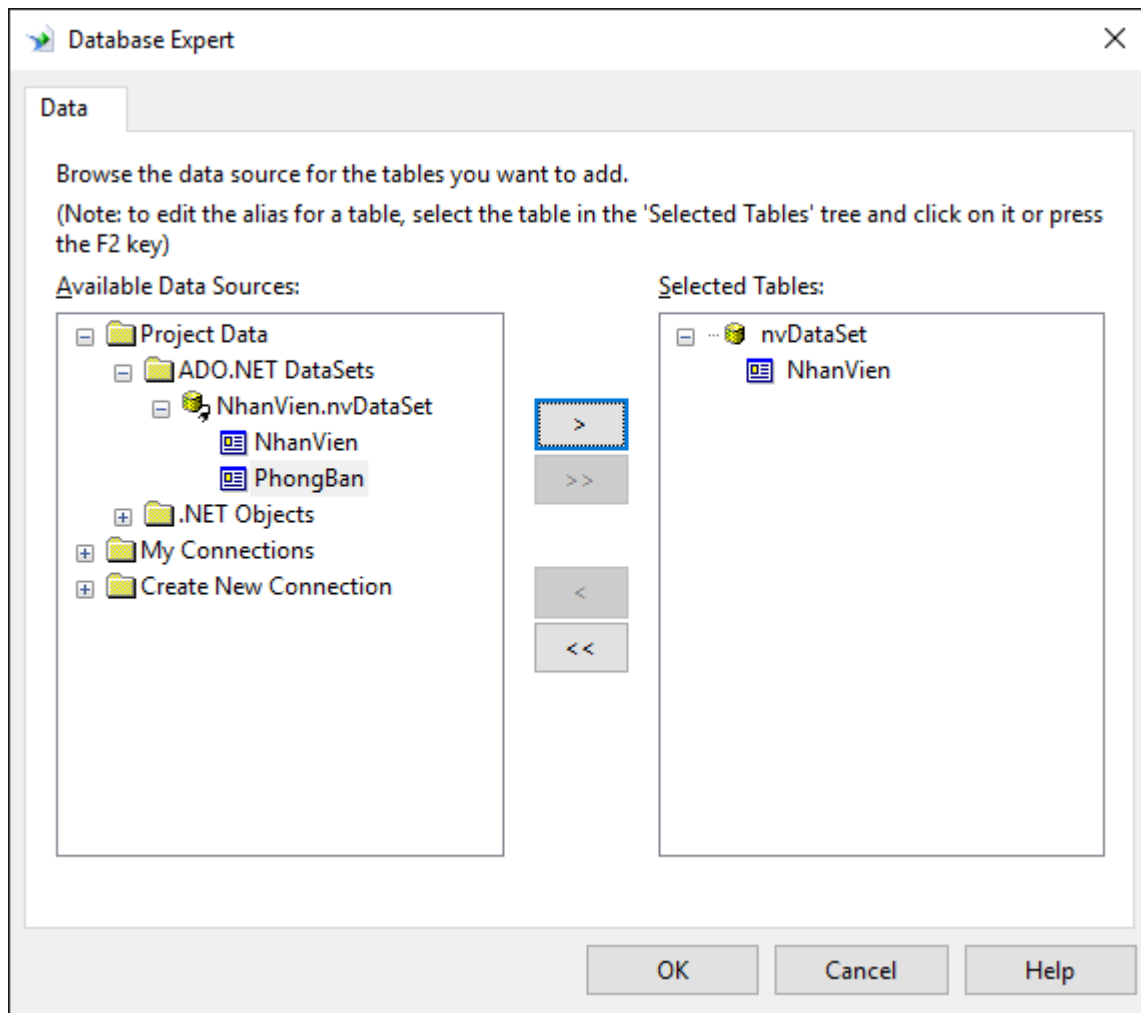
Hình 6.25. Minh họa kết quả sau khi chương trình thực thi.

2.4. Sử dụng tham số, biểu đồ, Stored Procedure cho report

2.4.1. Xây dựng chức năng report có tham số

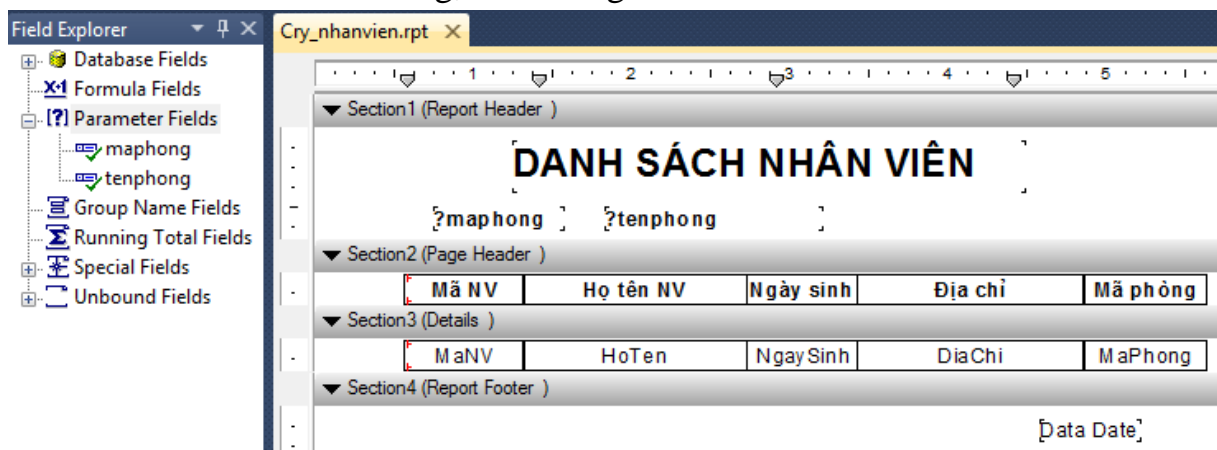
– Thêm các trường dữ liệu từ CSDL cho report, nhấp chuột phải vào Database Fields → chọn Database Expert...

– Chọn bảng dữ liệu NhanVien để thiết kế report danh sách nhân viên theo phòng ban.



Hình 6.26. Minh họa chọn dữ liệu nguồn cho report có tham số.

- Thiết kế report bằng cách kéo các đối tượng dữ liệu vào report. Sử dụng 2 Parameter Fields: MaPhong, TenPhong.



Hình 6.27. Minh họa chọn dữ liệu nguồn cho report có tham số.

- Chọn Crystal Report Viewer trong Tab Reporting của thanh công cụ kéo vào Form muốn hiển thị report.
- Sự kiện Form_Load:

```

Dim sql As String
'Đưa dữ liệu bảng PhongBan vào DataSet
sql = "select * from PhongBan"
Dim dt As New DataTable
dt = csdl.ExceQuery(sql)
Dim r As DataRow
r = dt.NewRow
r("MaPhong") = "-1"
r("TenPhong") = "Tất cả"
dt.Rows.InsertAt(r, 0)
cbPhongBan.DataSource = dt
cbPhongBan.ValueMember = "MaPhong"
cbPhongBan.DisplayMember = "TenPhong"

```

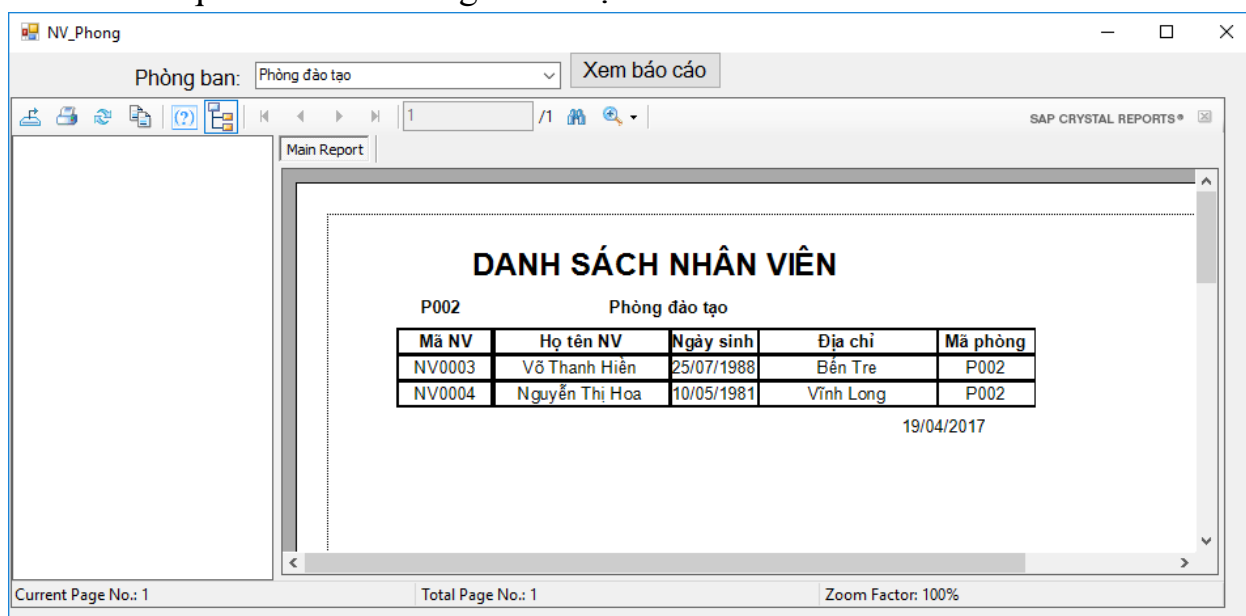
Sự kiện Click của nút *Xem báo cáo*:

```

Dim maphong As String = cbPhongBan.SelectedValue
Dim tenphong As String = cbPhongBan.Text
Dim sql As String
If maphong <> "-1" Then
    sql = "select * from NhanVien where MaPhong = '" & maphong & "'"
Else
    sql = "select * from NhanVien"
End If
Dim dt As New DataTable
dt = csdl.ExceQuery(sql)
'Khai báo biến Report
Dim rp As New Cry_nhanvien
'Gán dữ liệu nguồn cho Report
rp.SetDataSource(dt)
'Gán 2 tham số cho Report, nếu không thực hiện lệnh gán thì khi
Report hiển thị lúc chương trình đang chạy sẽ yêu cầu người dùng
nhập vào 2 tham số này
rp.SetParameterValue("MaPhong", maphong)
rp.SetParameterValue("TenPhong", tenphong)
CrystalReportViewer1.ReportSource = rp

```

– Kết quả sau khi chương trình thực thi:




The screenshot shows the SAP Crystal Reports interface. The report is titled 'DANH SÁCH NHÂN VIÊN' (Employee List) and is for the 'Phòng đào tạo' (Training Department). The report displays a table with the following data:

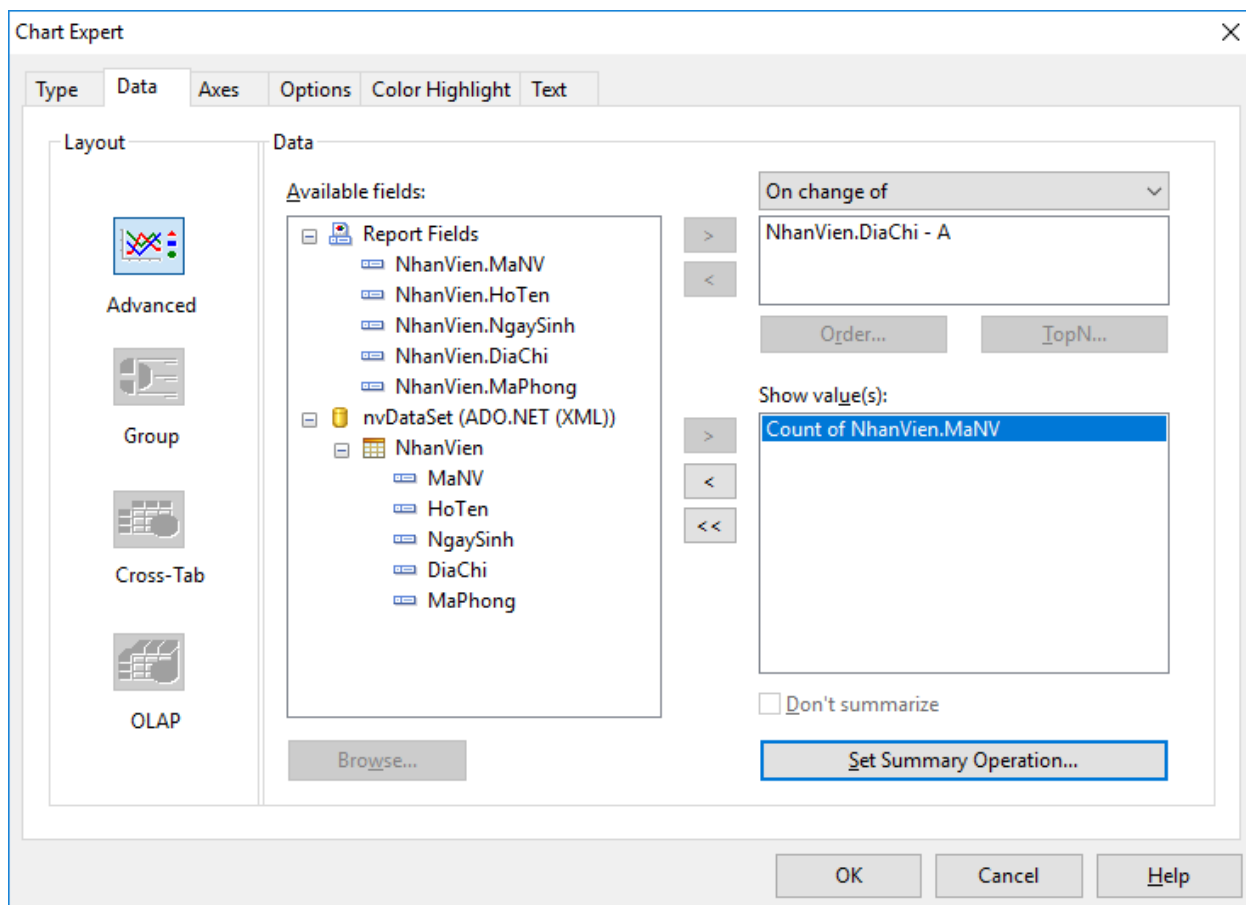
Mã NV	Họ tên NV	Ngày sinh	Địa chỉ	Mã phòng
NV0003	Võ Thanh Hiền	25/07/1988	Bến Tre	P002
NV0004	Nguyễn Thị Hoa	10/05/1981	Vĩnh Long	P002

The report is dated 19/04/2017. The interface also shows the 'Phòng ban: Phòng đào tạo' (Department: Training Department) and a 'Xem báo cáo' (View Report) button.

Hình 6.28. Minh họa kết quả sau khi thực thi Report có tham số.

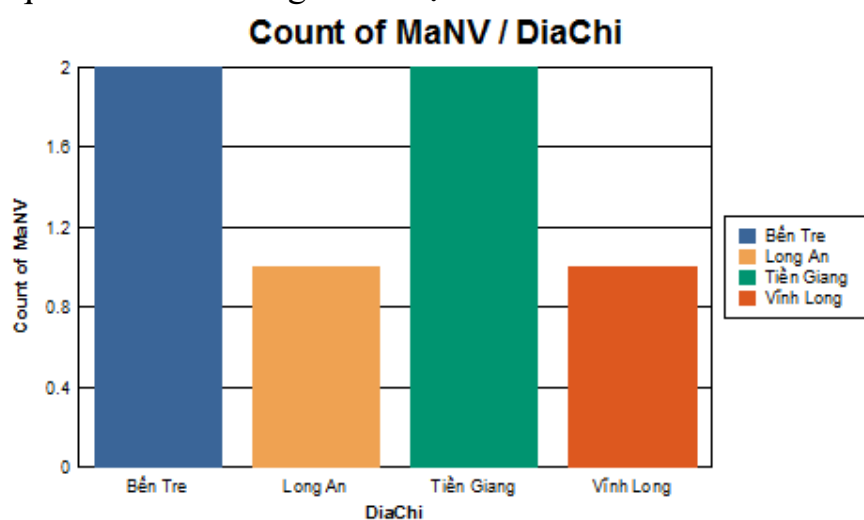
2.4.2. Xây dựng chức năng report có biểu đồ

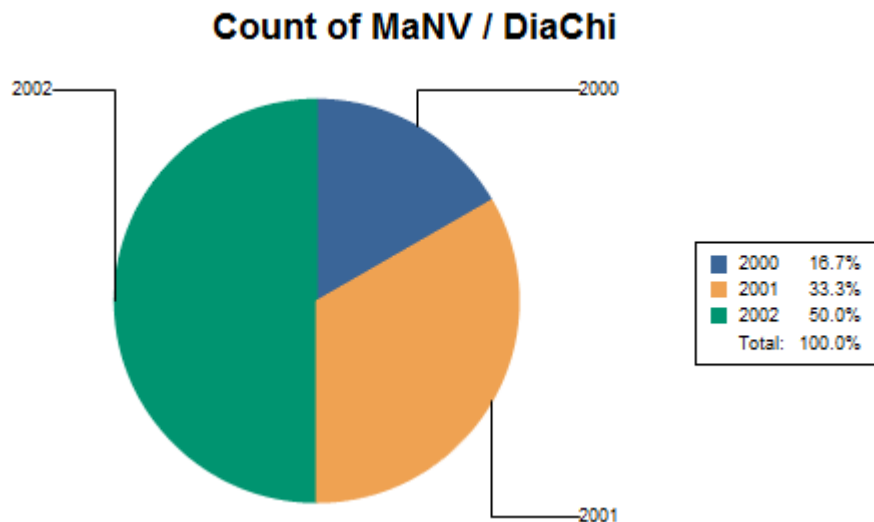
– Chọn Icon  trên thanh công cụ, sau đó đặt biểu đồ vào một vị trí nào đó trong report. Ví dụ, để tạo biểu đồ thống kê số lượng nhân viên theo từng địa chỉ. Với mỗi nhóm có cùng địa chỉ, sử dụng hàm Count trên MaNV. Có thể thay đổi hàm tính toán : Sum, Count, Average,... bằng cách chọn Set Summary Operation...



Hình 6.29. Minh họa thao tác report có biểu đồ.

– Kết quả sau khi chương trình thực thi:





Hình 6.30. Minh họa thao tác report có biểu đồ.

2.4.3. Sử dụng report với Store Procedure

Khi sử dụng Store Procedure để lấy dữ liệu cho Report thì Store Procedure được xem là nguồn dữ liệu giống với các bảng, và danh sách các cột sẽ được xác định bởi câu Select có trong Store Procedure.

Nếu trong Store Procedure có tham số thì tham số của Store Procedure chính là tham số của report. Khi sử dụng Report thì phải thiết lập giá trị cho tham số này bằng câu lệnh:

```
report.SetParameterValue("Tenthamsố", giá trị)
```

BÀI TẬP

Bài 1. Sử dụng CSDL quản lý sinh viên tạo các report **theo mẫu sau**

a.

Thông kê DS Lớp theo khoa

1 Mã Khoa ck

Tên Khoa cơ khí

Mã Lớp	Tên Lớp	Niên Khóa	GVCN
cdcka	cao đẳng cơ khí A	2000-2003	Lê Thành Nhân
cdck1	cao đẳng cơ khí a 1	2000-2003	Lê Thành Nhân

2 Mã Khoa cntt

Tên Khoa công nghệ thông tin

Mã Lớp	Tên Lớp	Niên Khóa	GVCN
cdthb	cao đẳng tin học B	2000-2003	Nguyễn Thanh Bình
cdtha	cao đẳng tin học A	2000-2003	Trần Bình Trọng

b. Nhóm dữ liệu theo sinh viên, Tạo công thức dùng Unbound Field xếp loại theo điểm trung bình của từng sinh viên.

THỐNG KÊ KẾT QUẢ HỌC TẬP CỦA TỪNG SINH VIÊN

MÃ SV	ck02	Tên SV	Trần Văn An
PHÁI	T	15/11/76 12:00:	Ngày Sinh

Mã MH	Lần Thi	Điểm
Isd	1	9
TRR	1	9
dth	1	7

ĐTB **8.33**

Xếp loại **gioi**

MÃ SV	ck03	Tên SV	Nguyễn Thị Thúy Hiền
PHÁI	F	12/02/78 12:00:	Ngày Sinh

Mã MH	Lần Thi	Điểm
dth	1	10
Isd	1	10
TRR	1	9
VL2	1	8

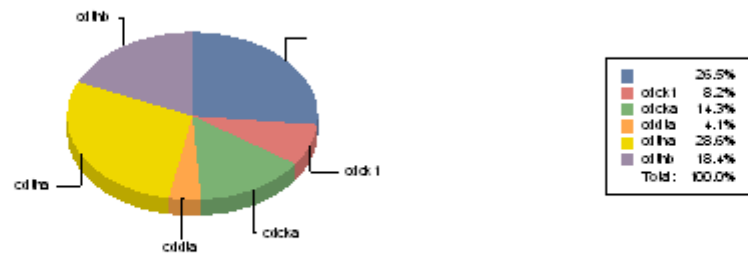
ĐTB **9.25**

Xếp loại **gioi**

c. Tạo mẫu giống câu b, nhưng xử lý thêm phần tham số cho báo cáo phục vụ cho nút in theo Masv hiện hành trên form bài 7 (phần thiết kế form).

d. Tạo báo cáo hiển thị dạng biểu đồ thống kê kết quả thi của sinh viên theo lớp

Thống kê kết quả học tập của từng lớp



Mã Lớp

Mã MH

csdl

Tên MH

cơ sở dữ liệu

Số TC

4

Mã SV

Tên SV

Phái

Lần Thi

Điểm

Kết quả

dt01

Lê Bá Hải thang

True

1

5

dau

Điểm lớn nhất

5.00

Điểm nhỏ nhất

5.00

Mã SV

Tên SV

Phái

Lần Thi

Điểm

Kết quả

dt03

Lê Vĩnh Phúc

True

1

7

dau

dt02

Phạm Thị Dương tt

False

1

8

dau

dt05

Nguyễn Thanh Tâm x

False

3

5

dau

dt01

Lê Bá Hải thang

True

2

4

rot

Điểm lớn nhất

8.00

Điểm nhỏ nhất

4.00

Bài 2. Sử dụng CSDL quản lý nhân viên (chương 3) thiết kế report với giao diện được thiết kế bên dưới.

THÔNG TIN CHẤM CÔNG CỦA NHÂN VIÊN

1 Mã nhân viên: A001
 Họ tên nhân viên: Trần Thị Nhung

STT	Mã công việc	Tên công việc	KL công việc	Đơn giá	Thành tiền
1	BV	Bảo vệ	10	5,000	100,000.00
2	CM	Cạo mũ	20	3,000	120,000.00

2 Mã nhân viên: A002
 Họ tên nhân viên: Hồ Thị Lan

STT	Mã công việc	Tên công việc	KL công việc	Đơn giá	Thành tiền
1	CM	Cạo mũ	15	3,000	45,000.00
2	QL	Quản lý	2	10,000	20,000.00

Bài 3. Sử dụng CSDL quản lý sinh viên khoa (chương 3) thiết kế report hiển thị thông tin như sau:

DANH SÁCH SINH VIÊN THEO KHOA

1 Mã khoa: NN
 Tên khoa: Ngoại ngữ

STT	MaSV	HoTen	Phai	NgaySinh
1	SV01	Trần Văn Triều	True	01/01/1995 12:00:00AM
2	SV02	Nguyễn Thanh Mai	False	25/10/1995 12:00:00AM

Tổng số sinh viên nữ: 1.00

Tổng số sinh viên nam: 1.00

...Tiếp theo cho các khoa khác...

Chương 7. BẢO MẬT VÀ ĐÓNG GÓI ỨNG DỤNG

Mục tiêu:

Sau khi học xong chương này, sinh viên có thể:

1. Kiến thức

- + Nhận biết được quy trình đóng gói đơn giản.
- + Trình bày các thành phần cơ bản trong đóng gói.
- + Phân biệt cách sử dụng Visual Studio .NET hoặc InstallShield để đóng gói ứng dụng.
- + Liệt kê cách mã hóa CSDL.

2. Kỹ năng

- + Đóng gói được một phần mềm ứng dụng và cài đặt sản phẩm sau khi đóng gói.
- + Thực hiện được mã hóa CSDL.

3. Thái độ

- + Tự tin đóng gói phần mềm ứng dụng.

1. Bảo mật thông tin

Trước đây khi công nghệ máy tính chưa phát triển, khi nói đến vấn đề an toàn bảo mật thông tin (Information Security), chúng ta thường hay nghĩ đến các biện pháp nhằm đảm bảo cho thông tin được trao đổi hay cất giữ một cách an toàn và bí mật. Ví dụ như:

- Đóng dấu và ký niêm phong một bức thư để biết rằng lá thư có được chuyển nguyên vẹn đến người nhận hay không.
- Dùng mật mã mã hóa thông điệp để chỉ có người gửi và người nhận hiểu được thông điệp. Phương pháp này thường được sử dụng trong chính trị và quân sự.
- Lưu giữ tài liệu mật trong các két sắt có khóa, tại các nơi được bảo vệ nghiêm ngặt, chỉ có những người được cấp quyền mới có thể xem tài liệu.

Với sự phát triển mạnh mẽ của công nghệ thông tin, đặc biệt là sự phát triển của mạng Internet, ngày càng có nhiều thông tin được lưu giữ trên máy vi tính và gửi đi trên mạng Internet. Do đó xuất hiện nhu cầu về an toàn và bảo mật thông tin trên máy tính. Có thể phân loại mô hình an toàn bảo mật thông tin trên máy tính theo hai hướng chính như sau:

1) Bảo vệ thông tin trong quá trình truyền thông tin trên mạng (Network Security)

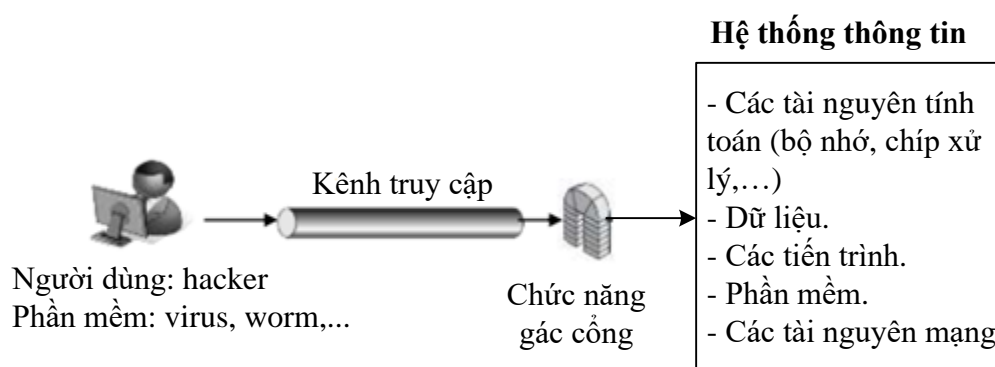
2) Bảo vệ hệ thống máy tính, và mạng máy tính, khỏi sự xâm nhập phá hoại từ bên ngoài (System Security).

Trong giới hạn của học phần này, chỉ đề cập đến vấn đề bảo mật thông tin tránh sự xâm nhập từ bên ngoài. Để thực hiện việc bảo vệ này, người ta dùng khái niệm “kiểm soát truy cập” (Access Control). Khái niệm kiểm soát truy cập bao gồm hai yếu tố sau:

- Chứng thực truy cập (Authentication): xác nhận rằng đối tượng (con người hay chương trình máy tính) được cấp phép truy cập vào hệ thống. Ví dụ: để sử dụng máy tính thì trước tiên đối tượng phải login vào máy tính bằng username và password. Ngoài ra, còn có các phương pháp chứng thực khác như sinh trắc học (dấu vân tay, móng mắt...) hay dùng thẻ (thẻ ATM...).

- Phân quyền (Authorization): các hành động được phép thực hiện sau khi đã truy cập vào hệ thống. Ví dụ: được cấp username và password để login vào hệ điều hành, tuy nhiên bạn chỉ được cấp quyền để đọc một file nào đó. Hoặc bạn chỉ có quyền đọc file mà không có quyền xóa file.

Để khắc phục các hành động phá hoại này, người ta dùng các chương trình có chức năng gác cổng, phòng chống. Những chương trình này dò tìm virus hoặc dò tìm các hành vi xâm phạm để ngăn chặn chúng, không cho chúng thực hiện hoặc xâm nhập. Đó là các chương trình chống virus, chương trình firewall... Ngoài ra, các nhà phát triển phần mềm cần có quy trình xây dựng và kiểm lỗi phần mềm nhằm hạn chế tối đa những lỗ hổng bảo mật có thể có.



Hình 7.1. Minh họa bảo mật thông tin.

Ví dụ: mã hóa thông tin: tên người dùng và mật khẩu khi THÊM và ĐĂNG NHẬP vào CSDL bằng thuật toán MD5:

Class DangNhapDTO:

```
Public _user As String  
Public _pass As String
```

```

Public Property User() As String
    Get
        Return _user
    End Get
    Set(ByVal value As String)
        _user = value
    End Set
End Property
Public Property Pass() As String
    Get
        Return _pass
    End Get
    Set(ByVal value As String)
        _pass = value
    End Set
End Property

```

Class DangNhapDAO:

```

Dim dn As New DangNhapDTO
Dim csdl As New KetNoiCSDL
Public Function ThongTinDangNhap() As DataTable
    Dim sql As String
    Dim dt As New DataTable
    sql = "select * from DangNhap"
    dt = csdl.ExceQuery(sql)
    Return dt
End Function
Public Sub ThemTaiKhoan(ByVal dn As DangNhapDTO)
    Try
        Dim sql As String
        sql = "INSERT INTO DangNhap([User], [Pass])VALUES('"
& dn.User & "','N'" & dn.Pass & "')"
        csdl.ExceNonQuery(sql)
    Catch ex As Exception
        MsgBox("Thêm tài khoản không thành công!")
    End Try
End Sub

```

Class DangNhapBUS:

```

Dim dn As New DangNhapDAO
Public Function TTDangNhap() As DataTable
    Dim dt As New DataTable
    dt = dn.ThongTinDangNhap
    Return dt
End Function
Public Function ThemTaiKhoan(ByVal tk As DangNhapDTO) As Integer
    Dim kq = 0
    Try
        dn.ThemTaiKhoan(tk)
        kq = 1
    End Try
End Function

```

```

Catch ex As Exception
    MsgBox("Thêm dữ liệu đăng nhập không thành công!")
End Try
Return kq
End Function

```

Sự kiện Click nút Thêm:

```

tk.User = EncryptMD5(txtUser.Text)
tk.Pass = EncryptMD5(txtPass.Text)
If dn.ThemTaiKhoan(tk) = 1 Then
    MsgBox("Thêm dữ liệu thành công.")
End If

```

Sự kiện Click nút đăng nhập:

```

Dim ten As String = EncryptMD5(txtUser.Text)
Dim mk As String = EncryptMD5(txtPass.Text)
Dim dt As New DataTable
dt = dn.TTDangNhap
Dim n As Integer
n = dt.Rows.Count
For i = 0 To n - 1
    If dt.Rows(i)("User") = ten And dt.Rows(i)("Pass") = mk Then
        MsgBox("Đăng nhập thành công")
    End If
Next
Dim dn As New DangNhapBUS
Dim tk As New DangNhapDTO
Public Function EncryptMD5(ByVal strPass As String) As String
Dim result As New System.Text.StringBuilder
Try
    Dim x As New
System.Security.Cryptography.MD5CryptoServiceProvider
    Dim bs() As Byte
    bs = System.Text.Encoding.UTF8.GetBytes(strPass)
    bs = x.ComputeHash(bs)
    For Each b As Byte In bs
        result.Append(b.ToString("x2").ToLower())
    Next
Catch ex As Exception
    MsgBox(ex.Message)
End Try
Return result.ToString.ToUpper
End Function

```

2. Tổng quan về đóng gói

Đóng gói là giai đoạn trung gian chuyển tiếp trong một dự án công nghệ thông tin nhằm tạo ra một phần mềm ứng dụng phục vụ cho một mục đích nhất định đồng thời bảo mật các mã lệnh trong chương trình.



Hình 7.2. Minh họa đóng gói.

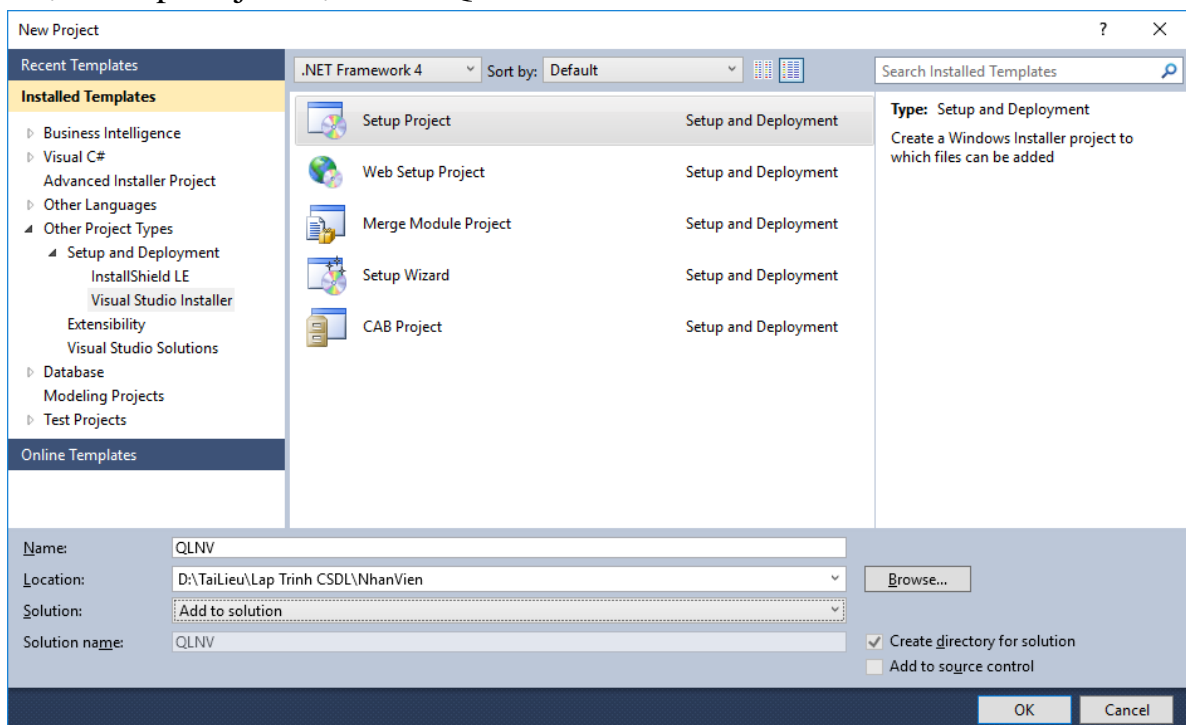
Một số công cụ đóng gói phổ biến hiện nay là: Smart Install Maker 3.2, Inno Setup, InstallShield, Advanced Installer.

3. Thao tác đóng gói

3.1 Sử dụng chức năng đóng gói trong VS.NET

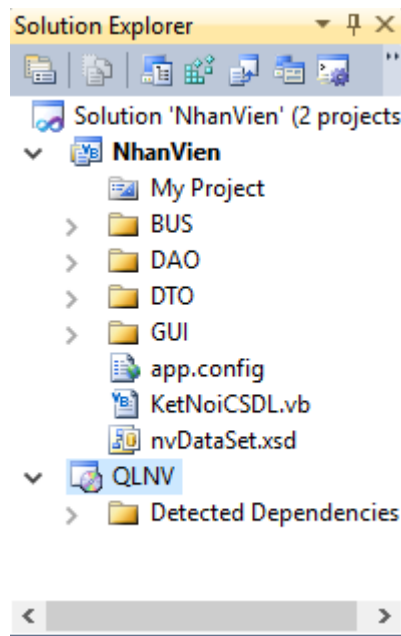
Sau khi đã hoàn thành phần viết mã lệnh của chương trình, lần lượt tiến hành các thao tác sau để đóng gói tạo thành một chương trình ứng dụng hoàn chỉnh.

– File → New Project → Other Project Types → Visual Studio Installer → chọn Setup Project đặt tên là QLNV → OK.



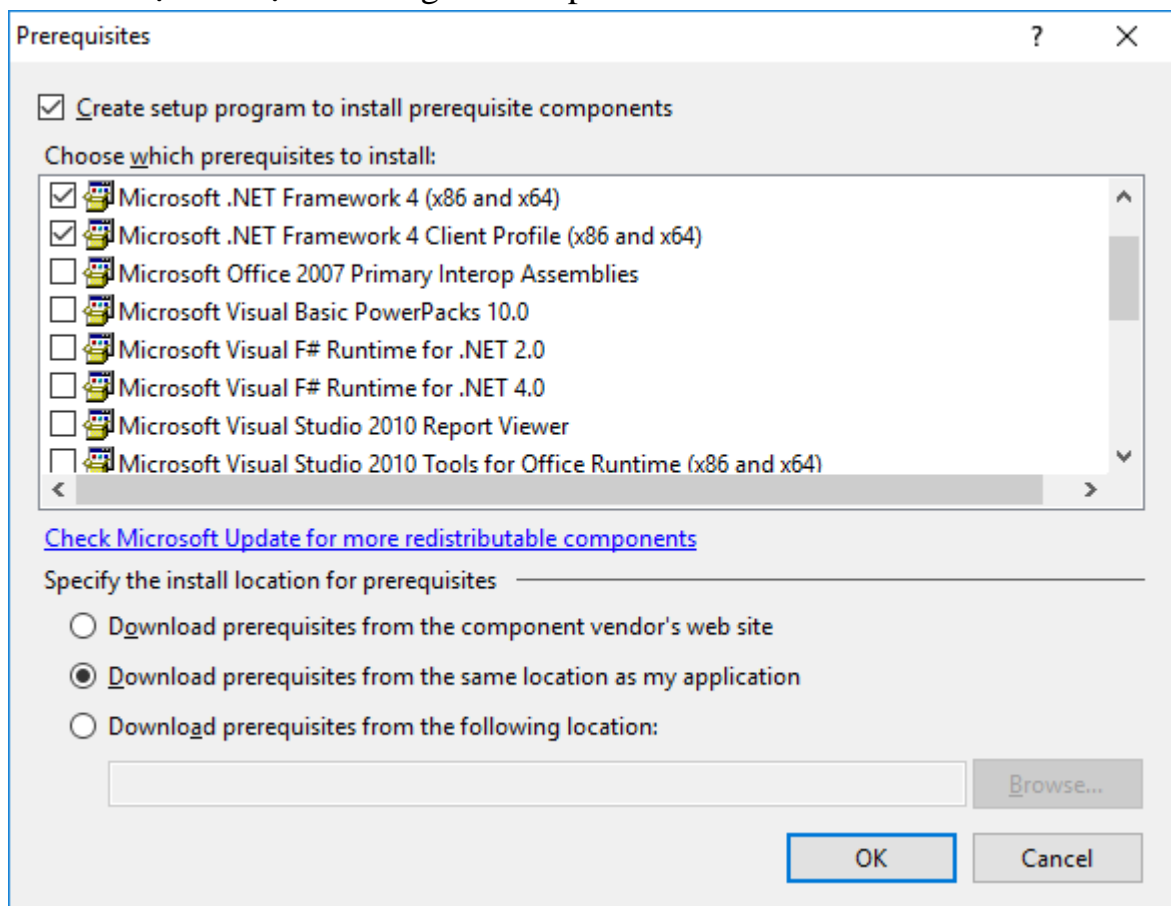
Hình 7.3. Minh họa thao tác đóng gói phần mềm.

– Sau khi chọn OK, trong cửa sổ Solution Explorer xuất hiện Setup Project QLNV như hình bên dưới:



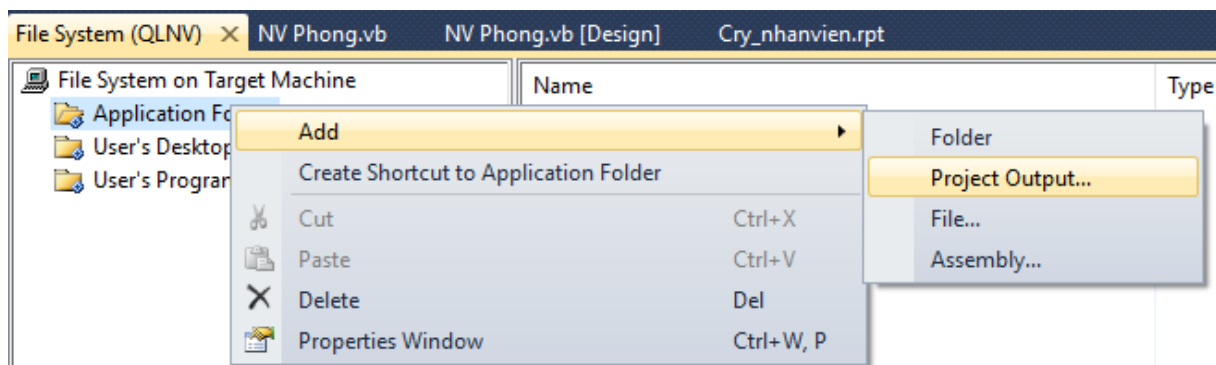
Hình 7.4. Minh họa thao tác đóng gói phần mềm.

– Nhấp chuột phải vào QLNv → chọn Properties → Chọn Prerequisites...
 Chọn như hình bên dưới để các thư viện hoặc các dll đính kèm theo hỗ trợ phần mềm sẽ được cài đặt khi trong file setup.



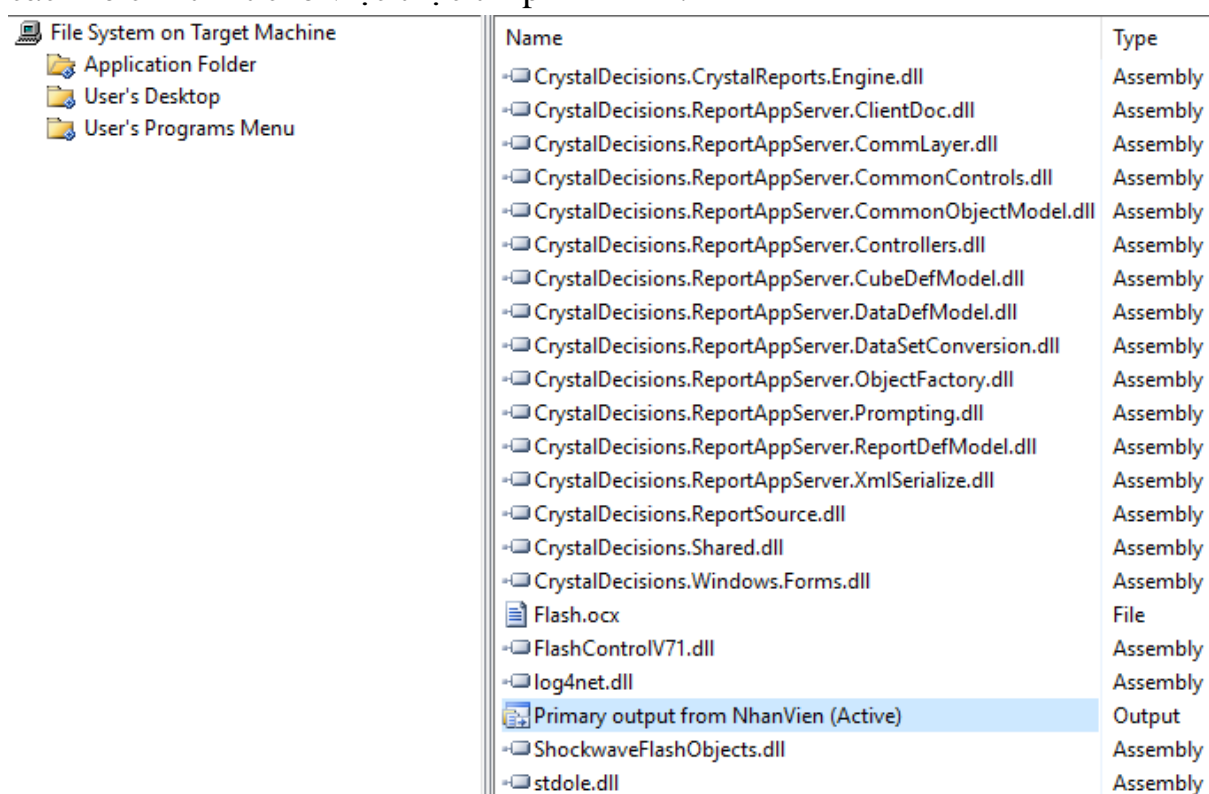
Hình 7.5. Minh họa thao tác đóng gói phần mềm.

Nhấp chuột phải vào Application Folder → Add → Project Output...:



Hình 7.6. Minh họa thao tác đóng gói phần mềm.

– Chọn Primary output → Ok, Visual Studio sẽ đính kèm các file .dll và các file cần thiết cho việc thực thi phần mềm.

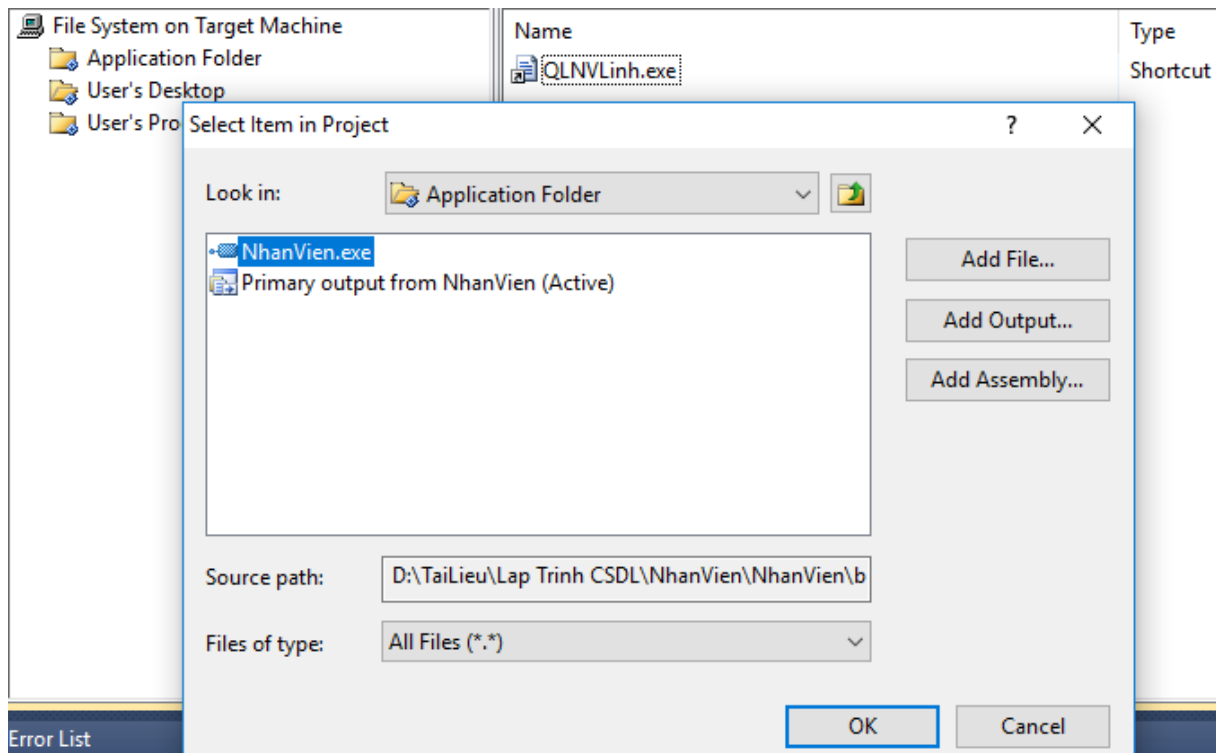


Hình 7.7. Minh họa thao tác đóng gói phần mềm.

– Nhấp chuột phải vào Application Folder → Add → File → di chuyển đến thư mục phần mềm → chọn Bin → Debug → chọn File NhanVien.exe.

– Để đặt biểu tượng hình cho File Setup sau khi cài đặt : nhấp chuột phải Application Folder → Add → File → chọn hình làm biểu tượng.

– Tạo shortcut cho phần mềm: khi cài đặt File Setup, chương trình sẽ tạo shortcut trên màn hình Desktop. Nhấp chuột phải vào User's Desktop → chọn Create New Shortcut → chọn Application Folder → chọn File NhanVien.exe, có thể đổi tên shortcut .

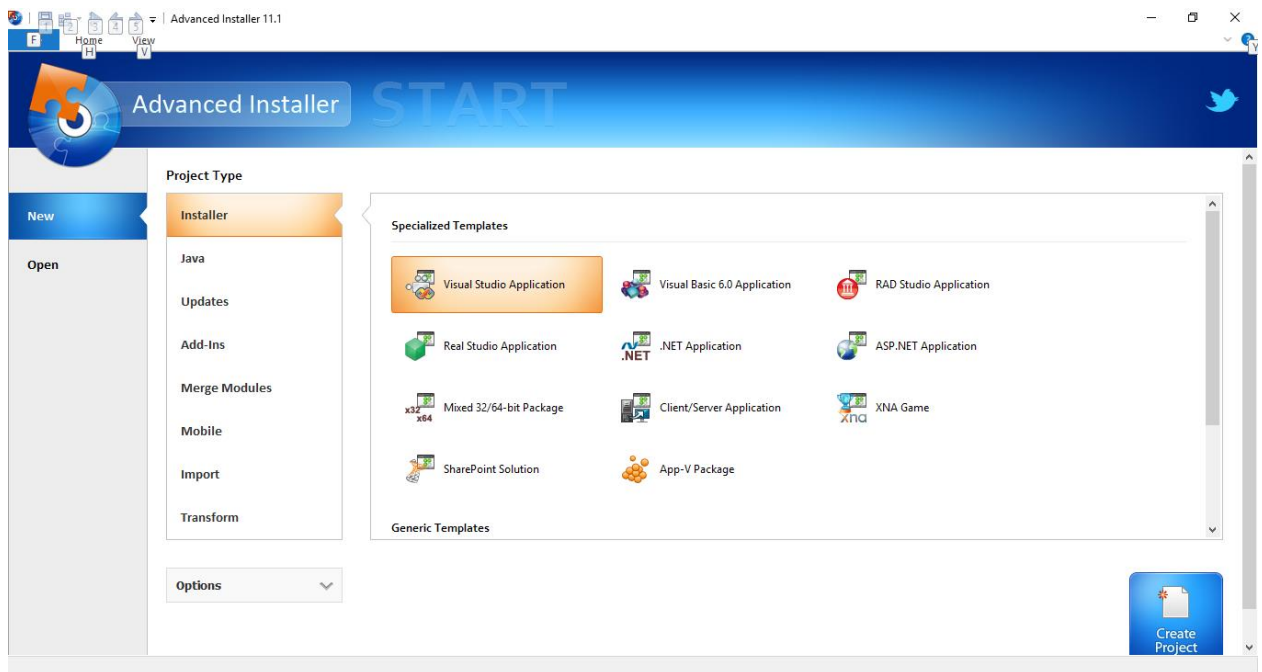


Hình 7.8. Minh họa thao tác đóng gói phần mềm.

- Nhấp chuột phải vào shortcut → Properties Windows → chọn Icon làm biểu tượng cho File Setup.
- Nhấp chuột phải vào Project vừa tạo (QLNV) → chọn Build.
- Sau khi có thông báo Build Succeeded, vào thư mục Debug của QLVN setup.exe đã được tạo.

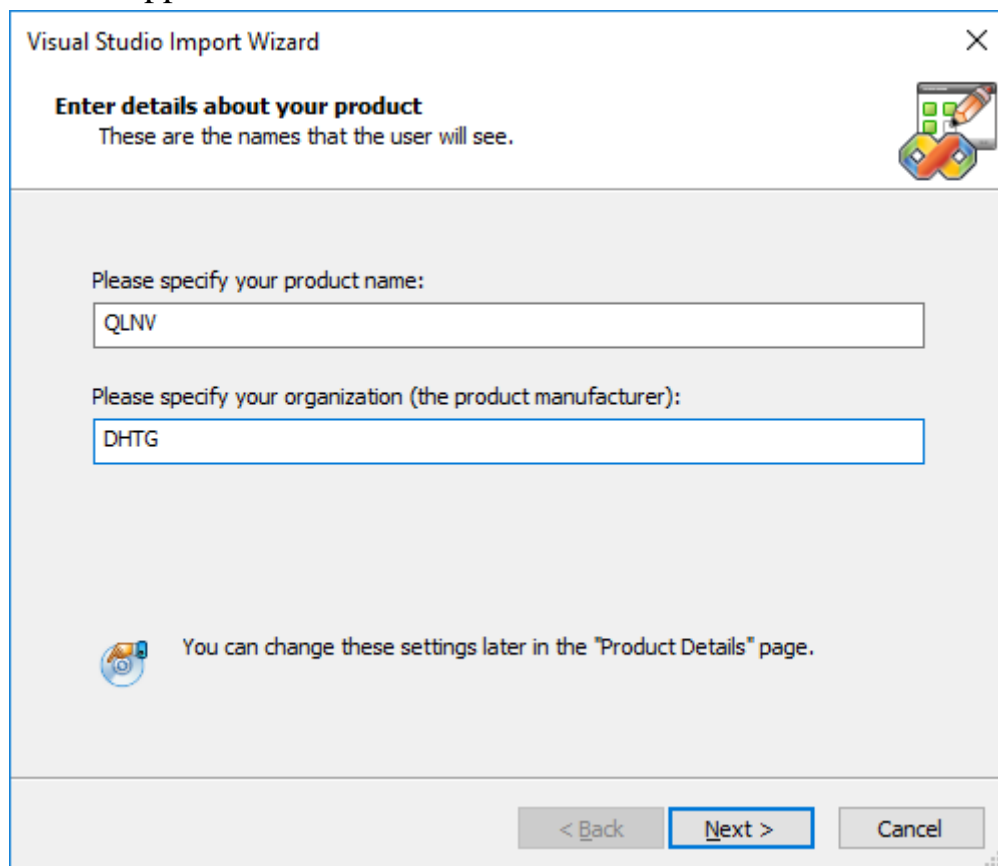
3.2 Sử dụng phần mềm Advanced Installer

- Cài đặt phần mềm Advanced Installer. Đây là phần mềm đóng gói cho phép tùy chỉnh giao diện cài đặt và các thiết lập nâng cao trên Windows.
- Mở Advanced Installer, chọn Visual Studio Application:



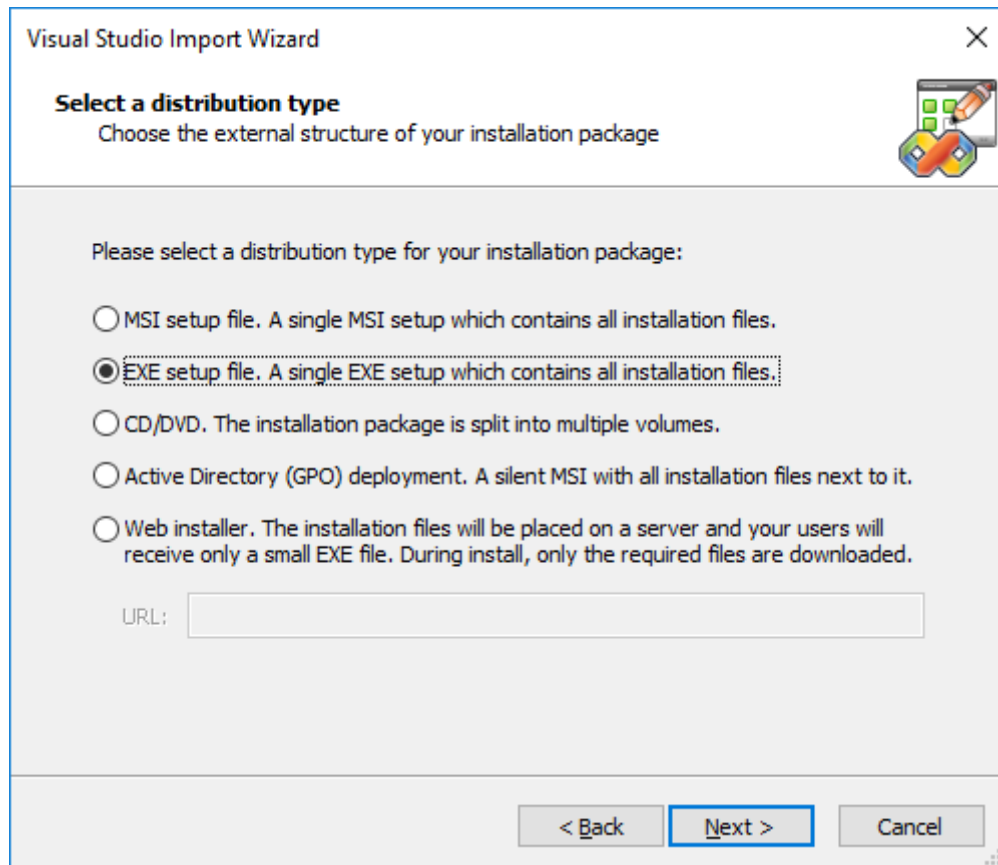
Hình 7.9. Minh họa thao tác đóng gói phần mềm.

– Nhấp chuột vào Create Project hoặc nhấp chuột hai lần liên tiếp vào Visual Studio Application:



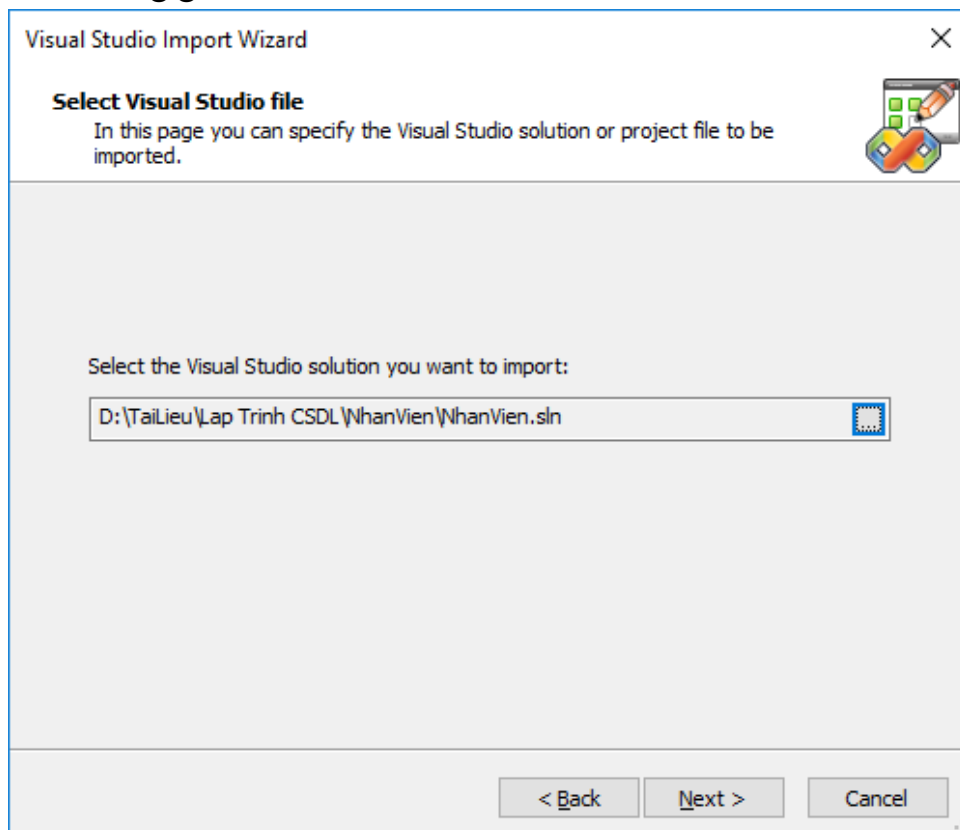
Hình 7.10. Minh họa thao tác đóng gói phần mềm.

– Nhấp Next và chọn loại File xuất ra:



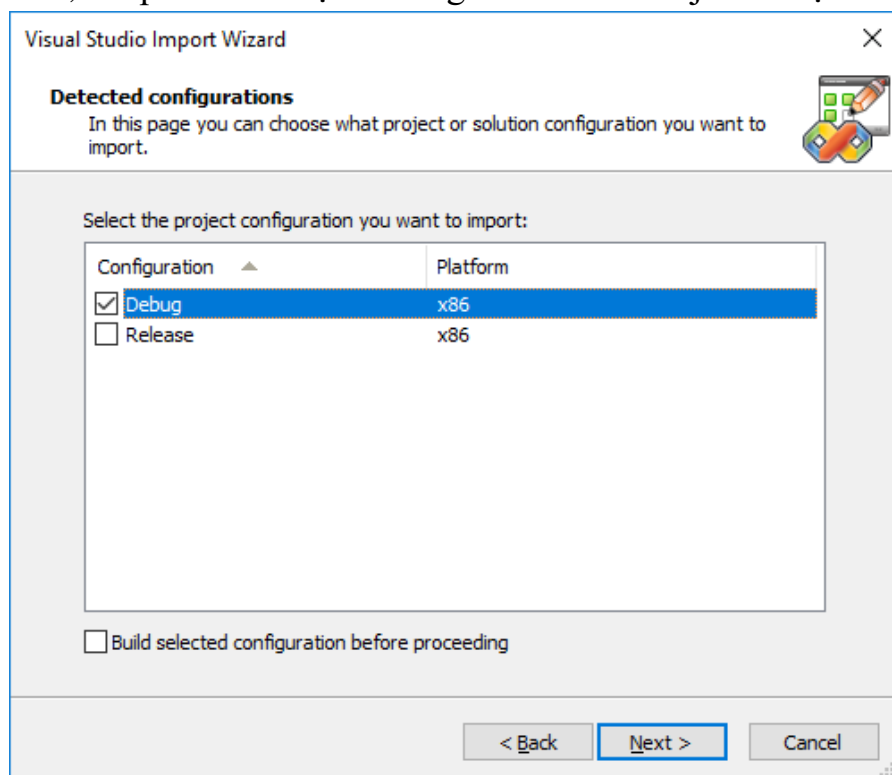
Hình 7.11. Minh họa thao tác đóng gói phần mềm.

- Nhấp Next hai lần liên tiếp, sau đó chọn đường dẫn lưu file .sln của Project muốn đóng gói.



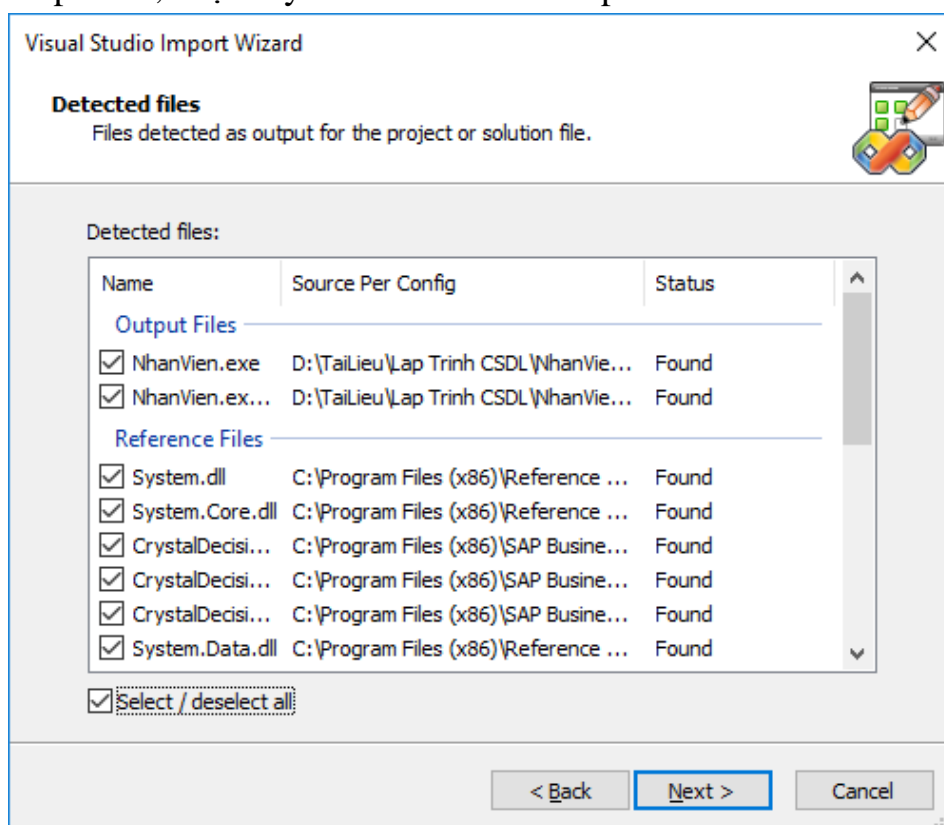
Hình 7.12. Minh họa thao tác đóng gói phần mềm.

Tiếp theo, nhấp Next và đợi chương trình Load Project. Chọn Debug.



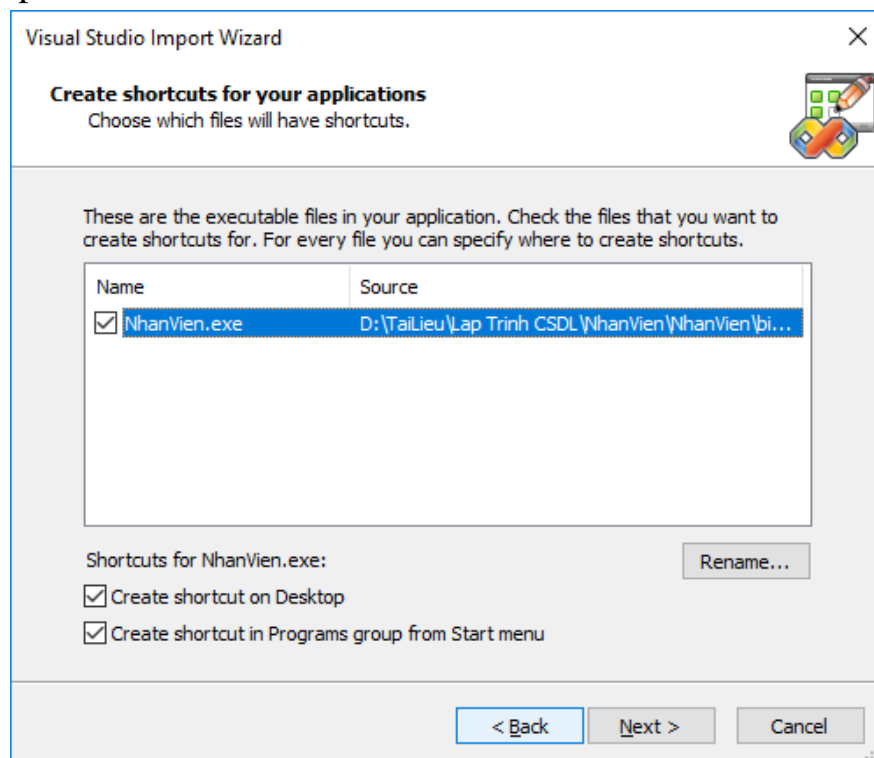
Hình 7.13. Minh họa thao tác đóng gói phần mềm.

– Nhấp Next, chọn tùy chỉnh cho File Setup:



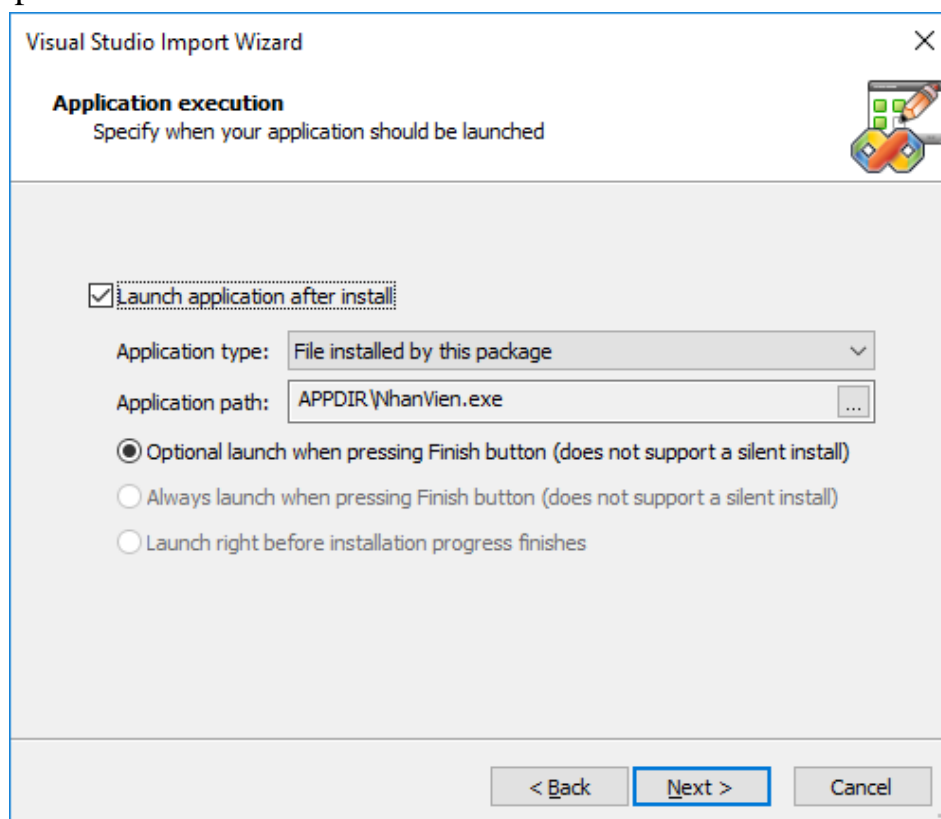
Hình 7.14. Minh họa thao tác đóng gói phần mềm.

– Nhấp Next



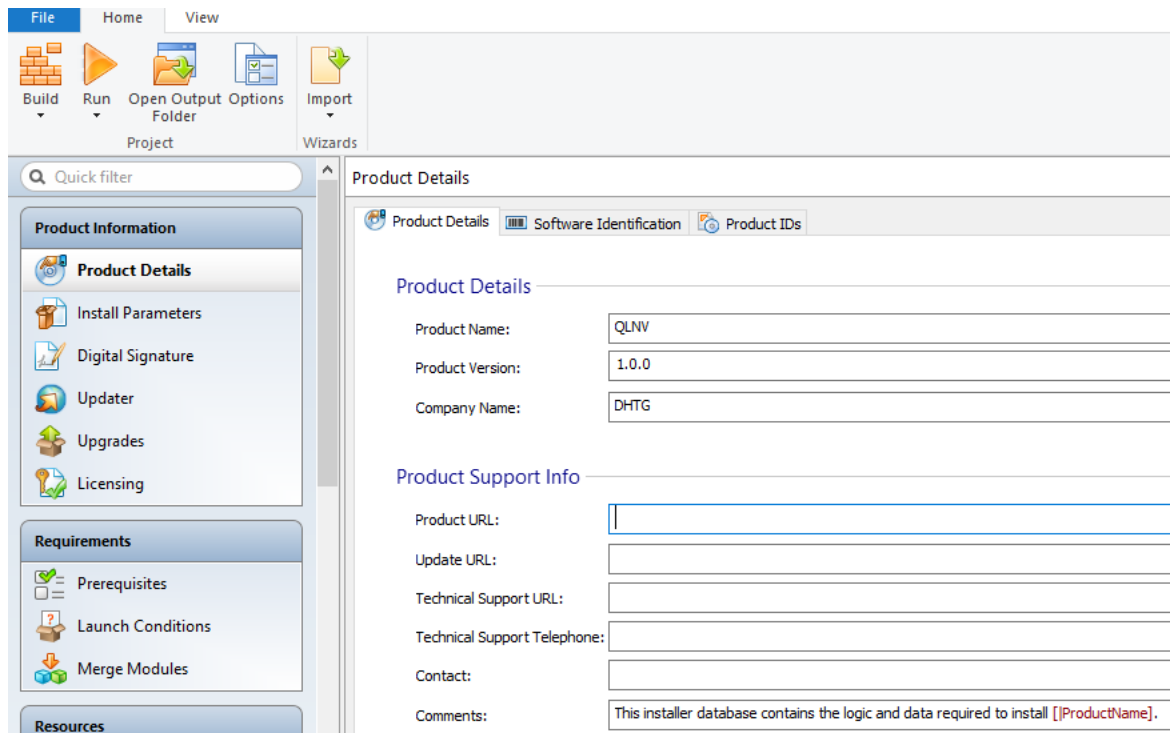
Hình 7.15. Minh họa thao tác đóng gói phần mềm.

Nhấp Next



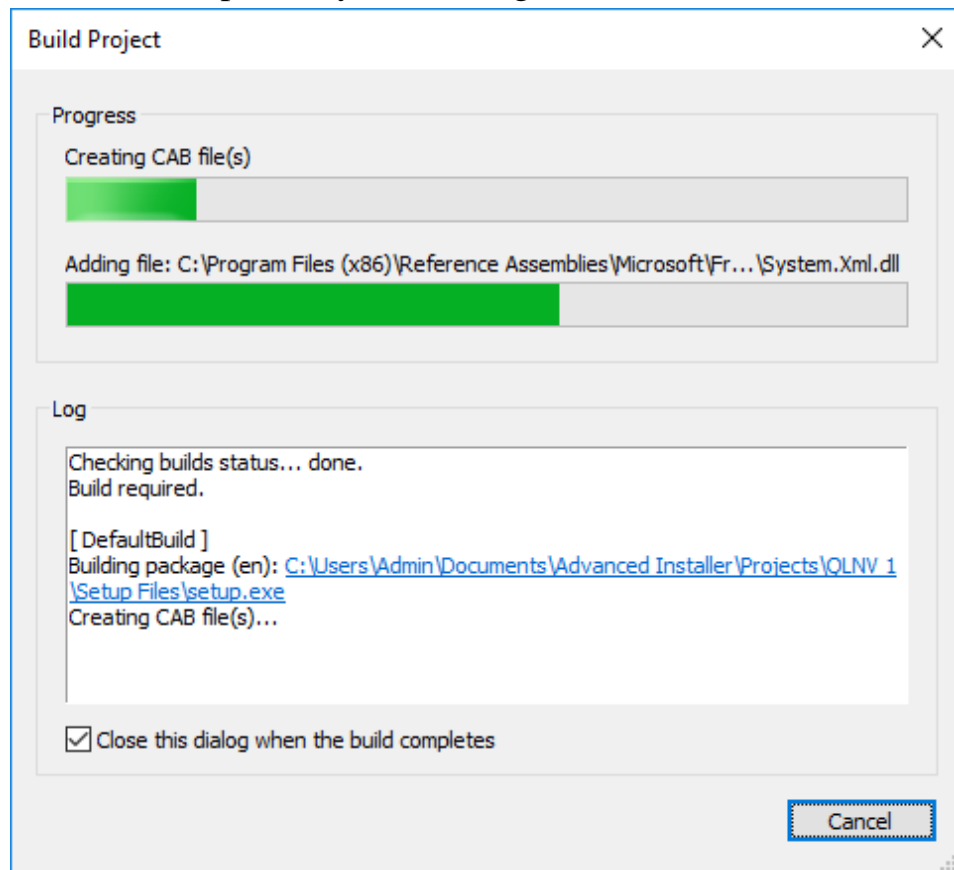
Hình 7.16. Minh họa thao tác đóng gói phần mềm.

– Nhấp Next → Finish, đến phần tùy chỉnh thông tin cho phần mềm.



Hình 7.17. Minh họa thao tác đóng gói phần mềm.


- Sau khi thiết lập các tùy chỉnh xong, chọn Build



Hình 7.18. Minh họa thao tác đóng gói phần mềm.

- Nhấp vào Open Output Folder để mở File Setup

This PC > Local Disk (C:) > Users > Admin > Documents > Advanced Installer > Projects > QLN1 > Setup Files

Name	Date modified	Type	Size
 setup.exe	19/04/2017 3:51 PM	Application	5,324 KB

Hình 7.19. Minh họa thao tác đóng gói phần mềm.

BÀI TẬP

Bài 1. Thực hiện đóng gói các bài tập chương 3, 4, 5 (sau khi đã hoàn thành) bằng Visual Studio .NET và sử dụng phần mềm Advanced Installer.

Bài 2. Xây dựng ứng dụng quản lý phòng trọ với các chức năng mô tả như sau:

- Người dùng: chủ phòng trọ.
- Quy trình khách đến thuê trọ.
 - + Khách trọ đến đăng ký thuê phòng trọ (thuê 1 phòng: số người phải ít hơn số người tối đa).
 - + Chủ nhà trọ thu tiền đặt cọc (đi sớm hơn 3 tháng là không trả cọc), phổ biến qui định và ký hợp đồng.
 - + Ngày cuối mỗi tháng chủ nhà trọ sẽ thu tiền và xuất phiếu thu (hóa đơn) gồm tiền phòng (cố định cho mỗi phòng), điện, nước, internet, ...)
 - + Trong quá trình thuê, khách có thể xin chủ nhà trọ cho thêm người vào phòng (số người phải nhỏ hơn hoặc bằng số người tối đa).
 - + Khách trả phòng, chủ nhà trọ trả lại tiền (nếu có).

Danh sách chức năng:

- Quản lý người dùng: chủ nhà trọ và người nhà của chủ nhà trọ
 - + Tên đăng nhập, mật khẩu.
 - + Thông tin tóm tắt.
- Quản lý khách trọ:
 - + Thông tin khách trọ: họ tên, cmnd, địa chỉ.
 - + Quản lý danh sách khu vực/tầng/dãy.
- Quản lý danh sách phòng trọ:
 - + Khu vực/tầng/dãy.
 - + Số phòng.
 - + Giá.
 - + Trạng thái phòng.
 - + Số lượng người tối đa.
 - + Thiết bị trong phòng (hiển thị các thiết bị thông dụng).
 - + Thiết bị khác.
- Chức năng cho thuê phòng.
- Chức năng trả phòng: tính toán các chi phí, xuất 1 phiếu chi.
- Quản lý danh sách phiếu chi:
 - + Ngày.
 - + Số tiền.
 - + Lý do.

- Quản lý danh sách phiếu thu hàng tháng (tính theo phòng): In hóa đơn và ký tên khi thu tiền:
 - + Tiền đặt cọc (tạo 1 phiếu thu).
 - + Tiền mỗi tháng: tiền nhà, tiền Internet: tính theo đầu người, tiền điện/tiền nước (2 hình thức: tính theo đầu người, tính theo số điện/nước).
- Mỗi tháng hiện trước danh sách phiếu thu của các phòng: khi hiện phiếu thu thì gợi ý trước (nếu tính tiền điện/nước theo chỉ số điện nước thì người dùng chỉ nhập điện nước là ra tổng số tiền) => phòng nào trả tiền xong thì ẩn đi.
- Thống kê:
 - + Thu/chi theo tháng của tất cả các phòng.
 - + Thu/chi của từng phòng theo thời gian.
 - + Lịch sử tiền điện, tiền nước của các phòng theo tháng.
 - + Lịch sử các lần nợ (mỗi lần các phòng trả tiền trễ thì lưu lại).
- Sao lưu và phục hồi dữ liệu.
- Phân quyền người dùng: hiện tại chỉ có 2 quyền (chủ: tất cả, người nhà chủ: xem và thu tiền).

TÀI LIỆU THAM KHẢO

- [1]. Đâu Quang Tuấn. *Tự học lập trình cơ sở dữ liệu SQL Server 2000 và Visual Basic.NET*. NXB Giao thông vận tải. 2005.
- [2]. Nguyễn Hữu Thiện, *Visual Basic.NET – tập 2*. NXB Đại Học Quốc Gia TPHCM. 2007.
- [3]. Nguyễn Thiên Bằng, Phương Lan. *Lập trình cơ sở dữ liệu với Visual Basic 2005 & ADO.NET 2.0*. NXB Lao Động Xã Hội. 2007.
- [4]. Nguyễn Tiến Huy. *Bài giảng Công nghệ XML và ứng dụng*. Đại học Khoa học Tự nhiên TP.HCM. 2016.
- [5]. Trần Minh Văn. *Bài giảng An toàn và bảo mật thông tin*. Đại học Nha Trang. 2008.
- [6]. Video hướng dẫn thao tác với XML, Linq. Trung tâm tin học Đại học Khoa học Tự nhiên TP.HCM. 2016.