C# Programming

Ngôn ngữ truy vấn tích hợp - LINQ

By Hoàng Hữu Việt
Email: viethh@vinhuni.edu.vn
Viện Kỹ thuật và Công nghệ, Đại học Vinh

Vinh, 9/2020

Mục đích, chuẩn đầu ra và nội dung

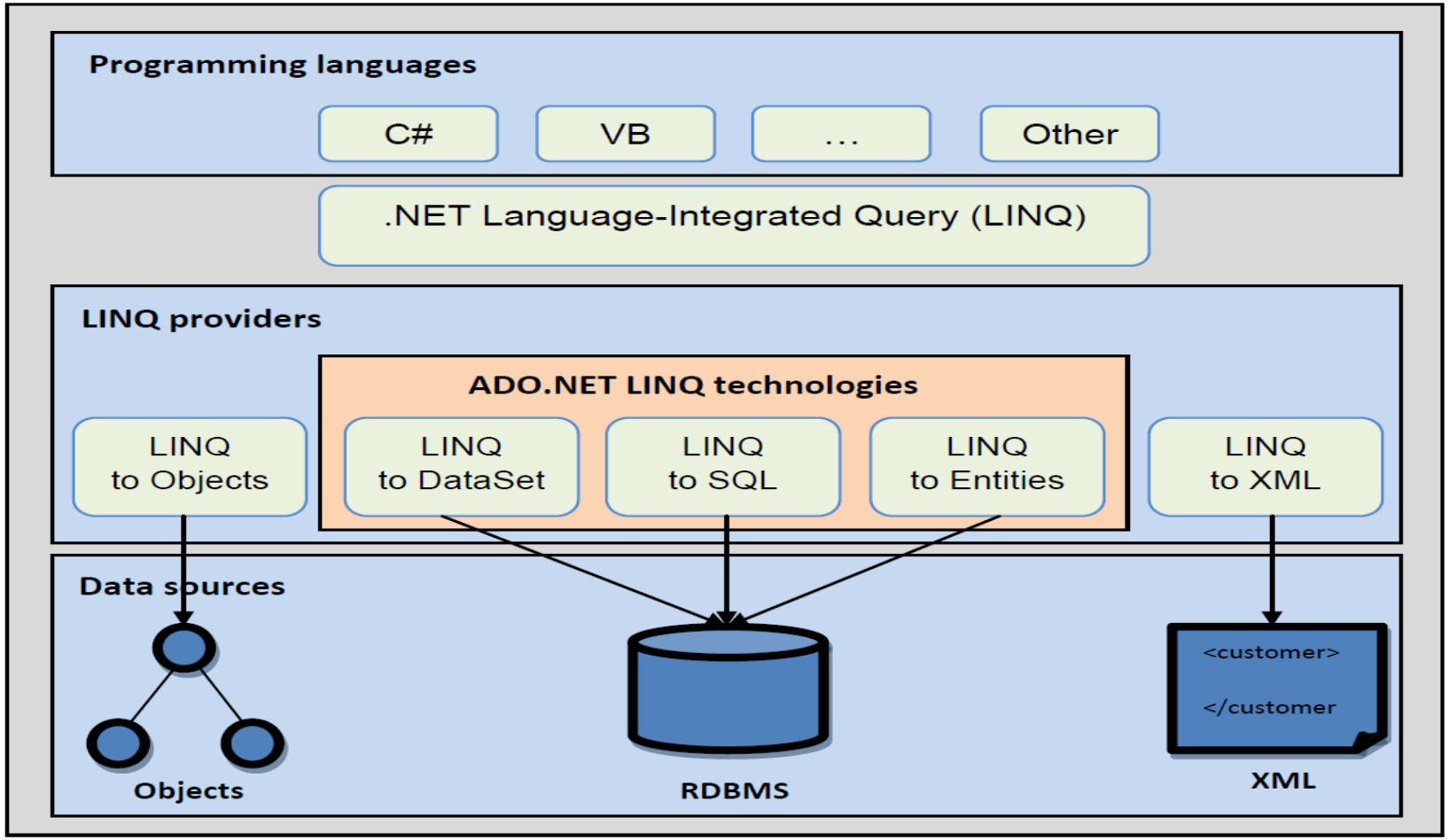
- Mục đích
  - Giới thiệu về lập trình cơ sở dữ liệu với LINQ (Language Integrated Query)
- Chuẩn đầu ra
  - Lập trình cơ sở dữ liệu sử dụng kỹ thuật LINQ
  - Kiểm thử và giải quyết các lỗi của ứng dụng
  - Phát triển kỹ năng lập trình, kỹ năng tìm kiếm và đọc hiểu tài liệu
- Nội dung
  - Giới thiệu
  - Lập trình LINQ to Objects
  - Lập trình LINQ to Entities

Mục đích của LINQ

- LINQ là được giới thiệu trong Visual Studio 2008 và .NET Framework 3.5 vào tháng 11 năm 2007.
- Khi LINQ chưa ra đời, các loại dữ liệu khác nhau phải truy vấn theo các phương pháp khác nhau:
  - Mảng, danh sách : Xây dựng thuật toán duyệt phần tử.
  - Cơ sở dữ liệu quan hệ: Ngôn ngữ SQL.
- LINQ được thiết kế cho mục đích:

  Đơn giản hóa và thống nhất thực hiện truy vấn các loại dữ liệu khác nhau mà không cần quan tâm đến kiểu lưu trữ của dữ liệu.

Mô hình LINQ



- Tài liệu này đề cập đến `LINQ to Objects` và `LINQ to Entities`.

Các thư viện của LINQ

- `System.Linq`: Chứa các lớp và các giao diện cho truy vấn `LINQ`.
- `System.Data.Linq`: Chứa các lớp làm việc của `LINQ to SQL`.
- `System.Data.Entities`: Chứa các lớp làm việc của `LINQ to Entities`.
- `System.Xml.Linq`: Chứa các lớp làm việc của `LINQ to XML`.
- `System.Data.Objects`: Chứa các lớp làm việc của mô hình dữ liệu thực thể `ADO.NET`.

Các phương pháp truy vấn với LINQ

- Dạng biểu thức truy vấn (query expression)
- Dạng phương thức (method-based query)
- Dạng hỗn hợp (mixed query), nghĩa là kết hợp biểu thức và phương thức.

Lập trình LINQ to Objects

- LINQ to Objects được thiết kế để truy vấn các đối tượng trong bộ nhớ như mảng và danh sách.
- Nội dung:
  - Truy vấn các phần tử của mảng.
  - Truy vấn các phân tử của danh sách.

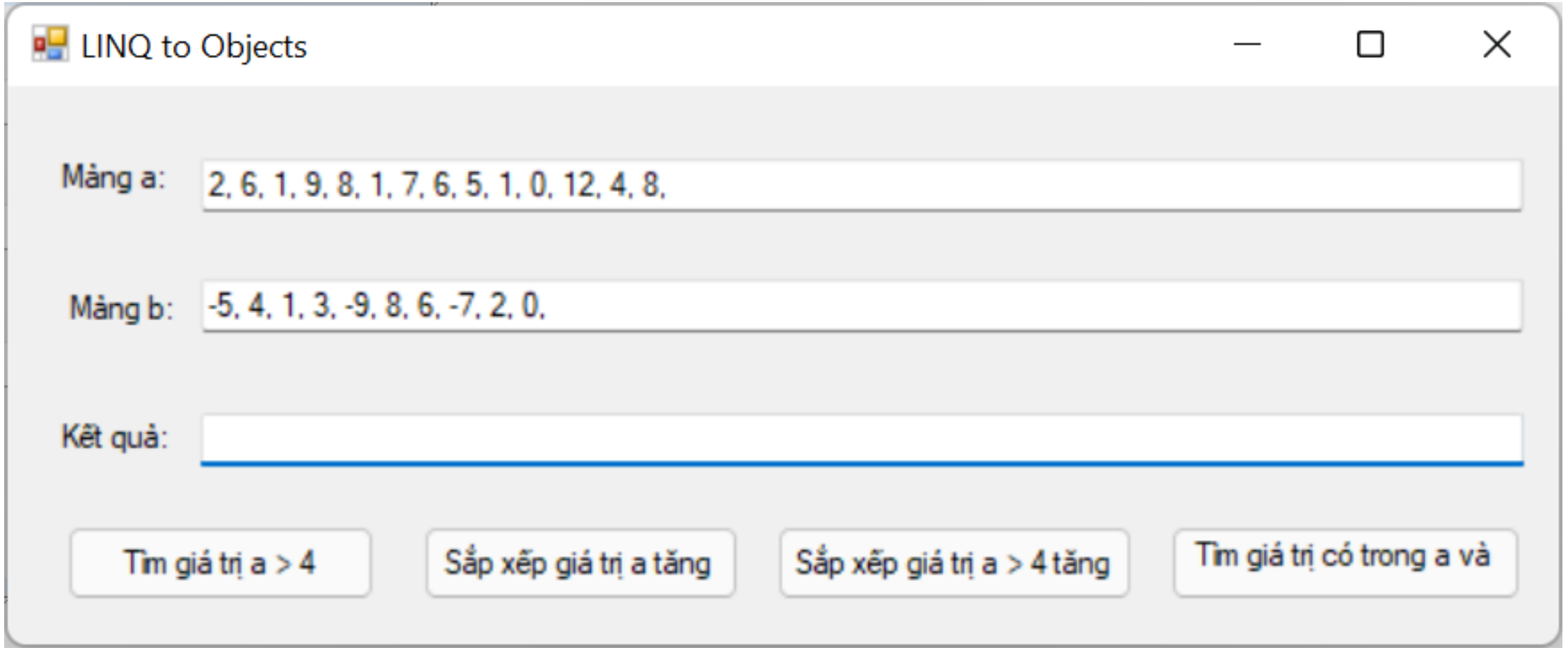
Dạng biểu thức truy vấn

```
query-expression = from itemName in srcExpr [join itemName in
srcExpr on keyExpr equals keyExpr (into itemName)]
[where predExpr]
[orderby keyExpr (ascending | descending)]
[select selExpr]
[group selExpr by keyExpr]
```

- Mệnh đề `from`: Truy vấn dữ liệu từ dữ liệu nguồn.
- Mệnh đề `join`: Kết nối dữ liệu giữa các biểu thức chứa dữ liệu truy vấn.
- Mệnh đề `where`: Lọc dữ liệu ứng với biểu thức điều kiện có giá trị đúng.
- Mệnh đề `orderby`: Sắp xếp kết quả truy vấn tăng dần hoặc giảm dần.
- Mệnh đề `select`: Dùng để chọn biểu thức trong kết quả truy vấn.
- Mệnh đề `group`: Dùng để nhóm dữ liệu của biểu thức truy vấn.

Lập trình LINQ to Objects

- Ví dụ thiết kế Form:



Truy vấn mảng dựa trên biểu thức

```
int[] a = {2,6,1,9,8,1,7,6,5,1,0,12,4,8};
int[] b = {-5,4,1,3,-9,8,6,-7,2,0};
```

- Truy vấn các phần tử lớn hơn 4 của mảng a:

```
var c = from i in a
        where i > 4
        select i;
```

  - Từ khóa var tự xác định kiểu của biểu thức truy vấn.

- Hiển thị kết quả:

```
textBox3.Clear();
foreach (var i in c)
     textBox3.Text = textBox3.Text + i.ToString() + ", ";
```

- Truy vấn các phần tử của mảng a, kết quả sắp xếp tăng dần:

```
var c = from i in a
        orderby i
        select i;
```

Truy vấn mảng dựa trên biểu thức

```
int[] a = {2,6,1,9,8,1,7,6,5,1,0,12,4,8};
int[] b = {-5,4,1,3,-9,8,6,-7,2,0 };
```

- Truy vấn các phần tử của mảng a lớn 4 với kết quả sắp xếp giảm dần:

```
var c = from i in a
        where i > 4
        orderby i descending
        select i;
```

  - Kết quả: c = {12,9,8,8,7,6,6,5};

- Truy vấn các phần tử của mảng a trong mảng b:

```
var c = from i in a
        join j in b on i equals j
        select i;
```

  - Kết quả: c = {2,6,1,8,1,6,1,0,4,8};

Truy vấn mảng dựa trên phương thức

- Truy vấn dựa trên phương thức sử dụng biểu thức Lambda như là tham số.
- Dạng cơ bản của biểu thức Lambda:

```
(input parameters) => expression|{statement;}
```

- Ví dụ: x `=>` $x*x$`;` chỉ ra một tham số với tên x nhận giá trị là x^2.

Truy vấn mảng dựa trên phương thức

- Truy vấn các phần tử lớn hơn 4 của mảng a:

```
var c = a.Where(i => i > 4)
         .Select(i => i);
```

  - Kết quả: c = {6,9,8,7,6,5,12,8};

- Truy vấn các phần tử của mảng a với kết quả sắp xếp tăng dần:

```
var c = a.OrderBy(i => i)
         .Select(i => i);
```

  - Kết quả: c = {0,1,1,1,2,4,5,6,6,7,8,8,9,12};

Truy vấn mảng dựa trên phương thức

- Truy vấn các phần tử của mảng a lớn 4 với kết quả sắp xếp giảm dần:

```
var c = a.Where(i => i > 4)
         .OrderByDescending(i => i)
         .Select(i => i);
```

  - Kết quả: c = {12, 9, 8, 8, 7, 6, 6, 5};

- Truy vấn các phần tử của mảng a trong mảng b:

```
var c = a.Join(b, i=>i, j => j, (i,j) =>j)
         .Select (i=>i);
```

  - Kết quả: c = {2, 6, 1, 8, 1, 6, 1, 0, 4, 8}

Phương thức và mệnh đề truy vấn

Phương thức	Mệnh đề truy vấn
`GroupBy()`	`group...by`
`Join()`	`join...in`
`OrderBy()`	`orderby`
`OrderByDescending()`	`orderby...descending`
`Select()`	`select`
`ThenBy()`	`orderby...,...`
`ThenByDescending()`	`orderby...,...descending`
`Where()`	`where`

Truy vấn các phần tử của danh sách

- Lớp `List` dùng để lưu trữ các phần tử của một danh sách.
- Khai báo: `List <T> <variable>;`

  trong đó <`T`> là kiểu dữ liệu và <`variable`> là một biến có kiểu danh sách.
- Một số phương thức:
  - `Add()`: Tạo một phần tử cuối danh sách.
  - `Clear()`: Xóa tất cả các phân tử của danh sách.
  - `Insert()`: Chèn phần tử ở vị trí chỉ ra.
  - `Remove()`: Xóa phần tử đầu tiên của giá trị chỉ ra.
  - `RemoveAt()`: Xóa phần tử ở vị trí chỉ ra trong danh sách.
  - `Sort()`: Sắp xếp danh sách.

Truy vấn các phần tử của danh sách

```
List <string> myList = new List <string>();
myList.Add("One");
myList.Add("Two");
myList.Add("Three");
myList.Add(“Four");
myList.Add(“Five");
...
myList.Add("Nine");
myList.Add("Ten");

var queryList = from item in myList
                where item.Contains("e")
                orderby item
                select item;
```

LINQ to Entities

- LINQ to Entities được thiết kế để cung cấp một giao diện làm việc với ADO.NET Entity Framework.
- LINQ to Entities làm việc với cơ sở dữ liệu như thực hiện các truy vấn, cập nhật và xóa dữ liệu thông qua một mô hình dữ liệu thực thể (EDM-entity data model).
- LINQ to Entities được thiết kế để làm việc với mọi cơ sở dữ liệu quan hệ.

Xây dựng mô hình thực thể ADO.NET

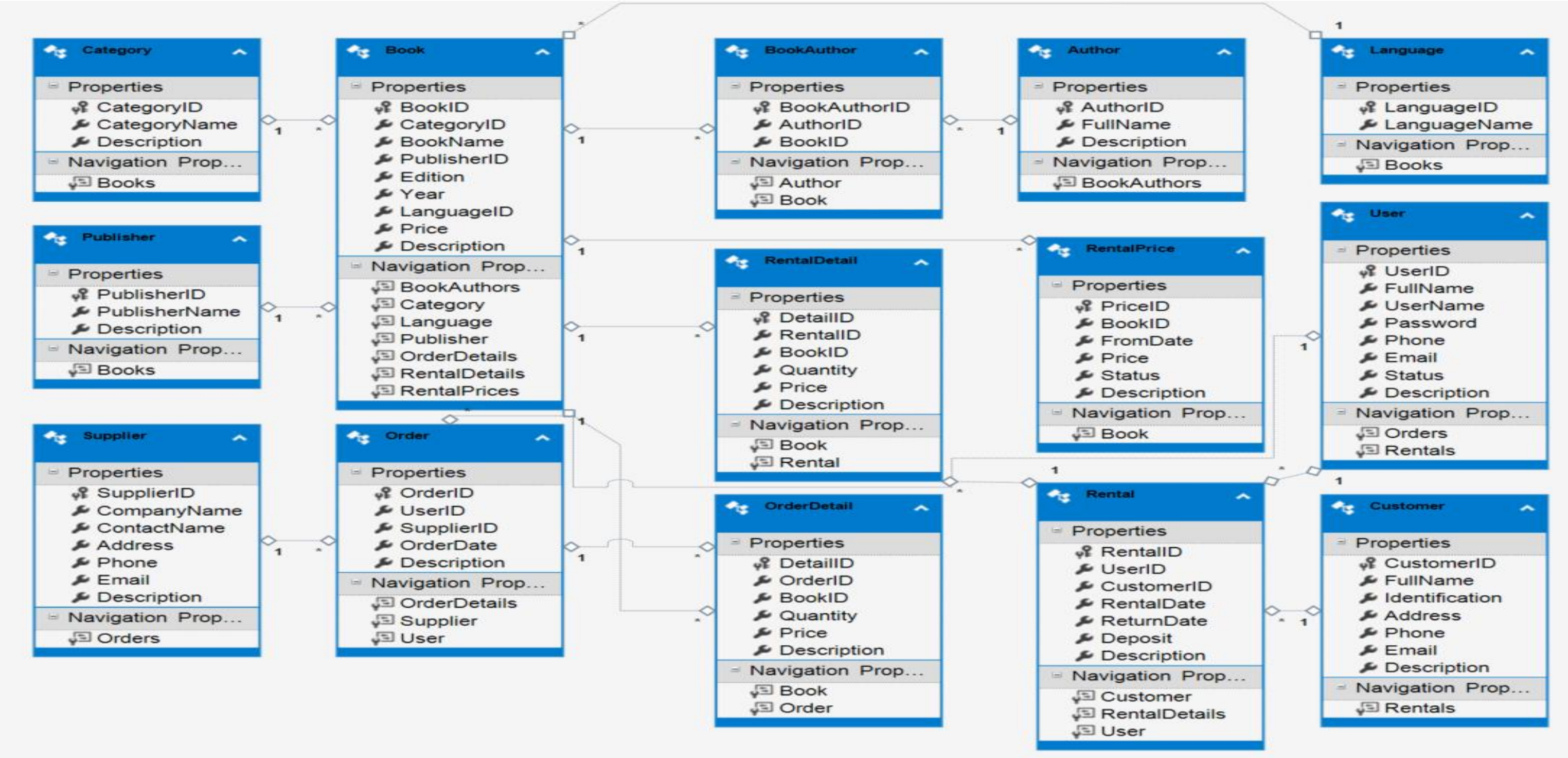
- Tạo tệp mô hình dữ liệu thực thể ADO.NET: Chọn menu Project, chọn Add > New Item.
- Chọn ADO.NET Entity Data Model, nhập tên tệp (ví dụ BooksEntities.edmx) và chọn Add.
- Chọn kiểu tạo mô hình dữ liệu thực thể: Trong cửa sổ Entity Data Model Wizard, chọn EF Designer from database và chọn Next.
- Thiết lập kết nối tới cơ sở dữ liệu: Chọn New Connection, xuất hiện cửa sổ Connection Properties.
- Chọn Server name > Authentication > Database name > chọn Ok.

Xây dựng mô hình thực thể ADO.NET

- Chọn kiểu lưu trữ xâu kết nối cơ sở dữ liệu: Chọn `Yes, include the sensitive data` và chọn `Next`.
- Nhập tên mô hình thực thể: Chọn `Save entity connection settings`, nhập tên mô hình thực thể (ví dụ `BookEntities`) và chọn `Next`.
  - Chọn Entity Framework 6.x/ Entity Framework 5.x.
- Chọn các đối tượng của cơ sở dữ liệu: Chọn `Tables`, chọn các bảng dữ liệu, nhập không gian tên chứa mô hình thực thể (ví dụ `BooksModel`) và chọn `Finish`.
- Biên dịch mô hình: Chọn thực đơn `Build > Build Solution`.

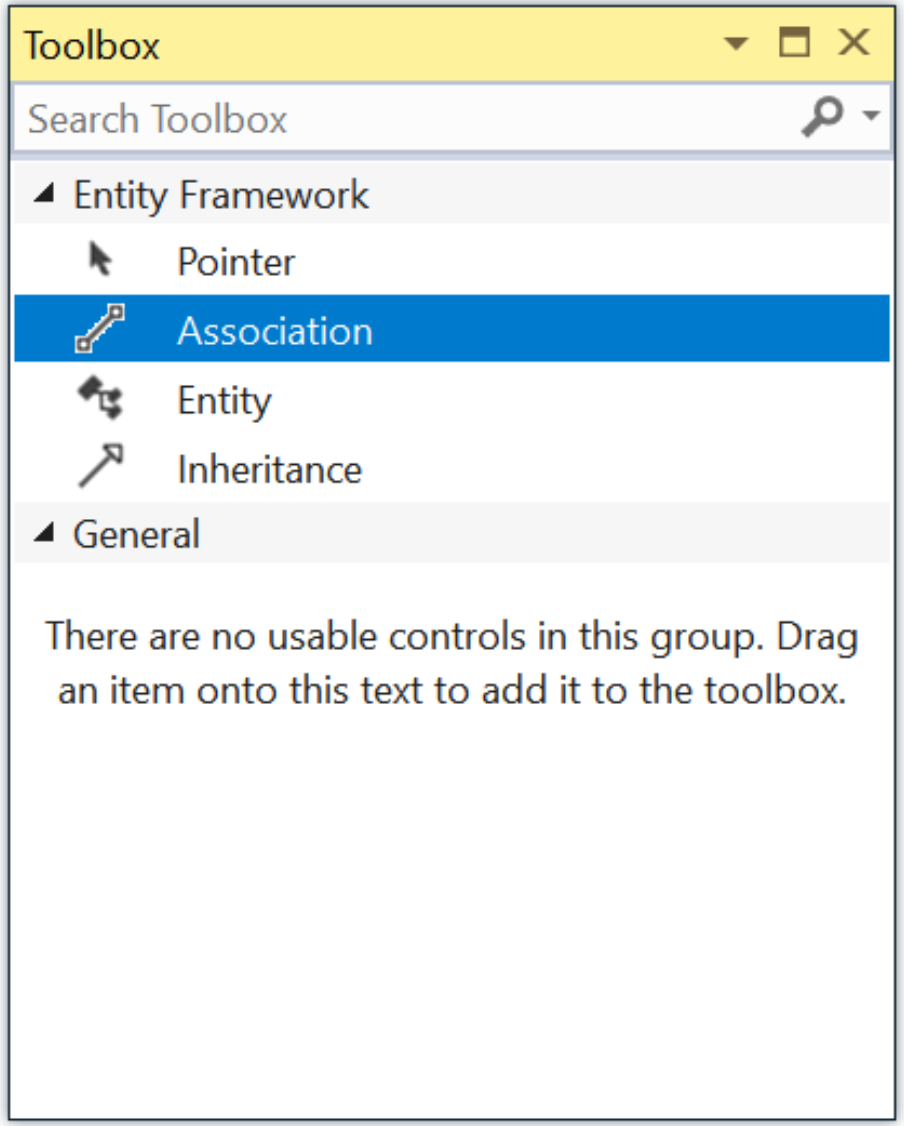
Xây dựng mô hình thực thể ADO.NET

- Mô hình quan hệ giữa các lớp thực thể:



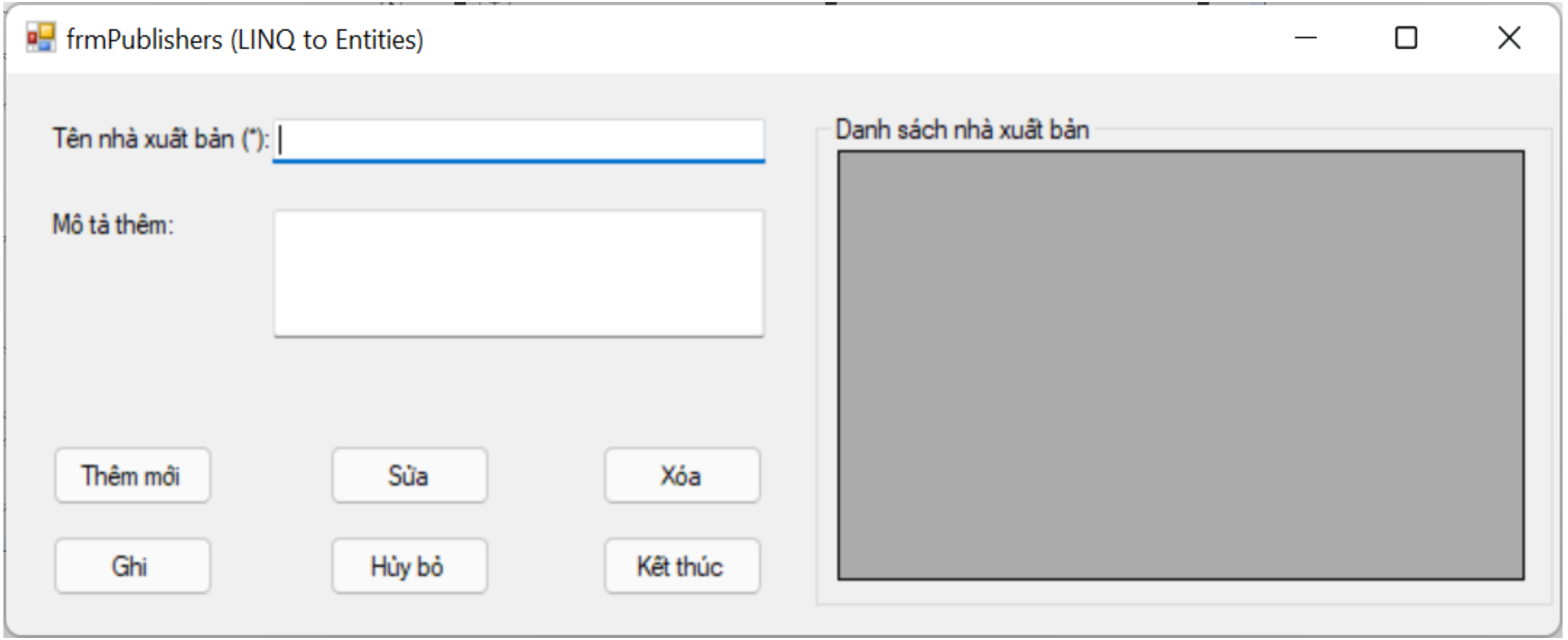
Xây dựng mô hình thực thể ADO.NET

- Sửa quan hệ các lớp trong mô hình thực thể:
  - Chọn cửa sổ `Toolbox`, chọn `Association`.
  - Tạo quan hệ giữa các lớp: kéo thuộc tính khóa của lớp thực thể nối với nhau.
  - Sửa quan hệ giữa các lớp: chọn đường liên kết các thực thể và chọn cửa sổ `Properties`.
  - Xóa quan hệ giữa các lớp: chọn đường liên kết và ấn phím delete.



Xây dựng ứng dụng với LINQ to Entities

- Xây dựng Form:



Truy vấn dữ liệu với LINQ to Entities

- Tạo đối tượng lớp mô hình dữ liệu thực thể ADO.NET, ví dụ tạo đối tượng lớp BooksEntities:

```
BooksEntities myBooksEntities = new BooksEntities();
```

- Định nghĩa và thực hiện câu lệnh truy vấn dữ liệu.
  - Ví dụ truy vấn tất cả các nhà xuất bản:

```
var queryPublishers = from item in myBooksEntities.Publishers
                      orderby item.PublisherName
                      select item;
```

- Hiển thị lên dataGridView1:

```
dataGridView1.DataSource = queryPublishers.ToList();
```

Nhập dữ liệu với LINQ to Entities

- Tạo đối tượng lớp thực thể:

```
Publisher itemPublisher = new Publisher();
```

- Gán giá trị cho đối tượng lớp thực thể:

```
itemPublisher.PublisherName = txtPublisherName.Text;
itemPublisher.Description = txtDescription.Text;
```

- Thêm `itemPublisher` vào `Publisher`:

```
myBooksEntities.Publishers.Add(itemPublisher);
```

- Cập nhật lại bảng dữ liệu trong CSDL:

```
myBooksEntities.SaveChanges();
```

Sửa dữ liệu với LINQ to Entities

- Tìm bản ghi cần sửa, ví dụ tìm theo PublisherID:

```
var queryPublishers = from item in myBooksEntities.Publishers
                      where (item.PublisherID == PublisherID)
                      select item;
```

- Lấy đối tượng đầu tiên trong bảng dữ liệu truy vấn, ví dụ:

```
Publisher itemPublisher = queryPublishers.First();
```

- Gán giá trị mới cho đối tượng lớp thực thể, ví dụ:

```
itemPublisher.PublisherName = txtPublisherName.Text;
itemPublisher.Description = txtDescription.Text;
```

- Cập nhật lại bảng dữ liệu trong CSDL:

```
myBooksEntities.SaveChanges();
```

Xóa dữ liệu với LINQ to Entities

- Lấy mã PublisherID của dòng được chọn:

```
int r = dataGridView1.CurrentRow.Index;
int PublisherID = Convert.ToInt32(dataGridView1.Rows[r].Cells[0].Value.ToString());
```

- Tìm bản ghi cần xóa :

```
var queryPublishers = from item in myBooksEntities.Publishers
                      where (item.PublisherID == PublisherID)
                      select item;
```

- Xóa đối tượng lớp thực thể trong lớp thực thể:

```
myBooksEntities.Publishers.Remove(queryPublishers.First());
```

- Cập nhật lại bảng trong CSDL:

```
myBooksEntities.SaveChanges();
```

Kết thúc

HỎI & TRẢ LỜI