

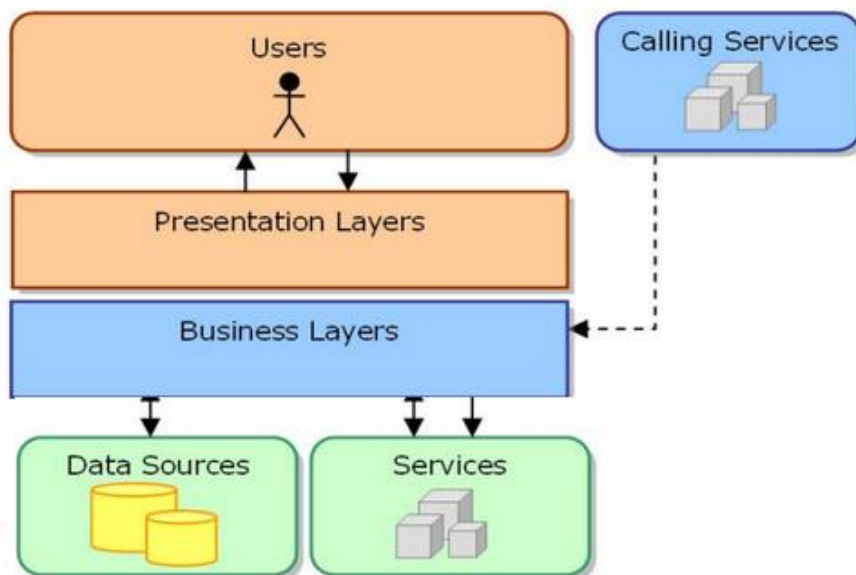
Chương 13**LINQ TO SQL & ASP.NET**

Sau khi học xong bài này, học viên có khả năng :

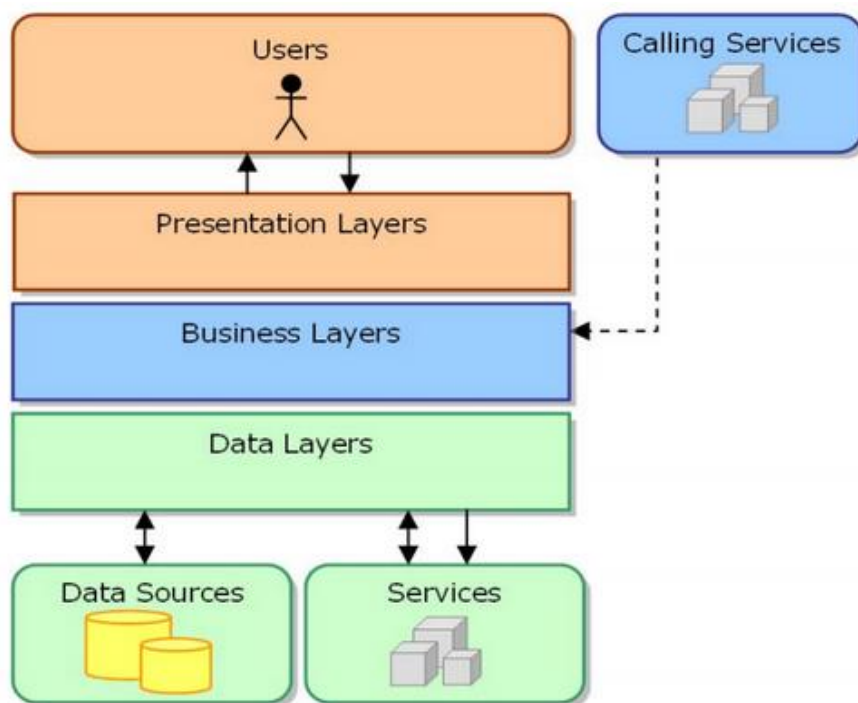
- Xây dựng được ứng dụng ASP.NET tương tác với Cơ sở dữ liệu thông qua mô hình 2 tier
- Sử dụng được LINQ to Object để truy vấn dữ liệu trong mô hình 2 tier
- Mô tả và sử dụng được ObjectDataSource control với các Data Controls
- Xây dựng được ứng dụng ASP.NET tương tác với Cơ sở dữ liệu thông qua LINQ to SQL
- Thực hiện được truy vấn dữ liệu sử dụng LINQ to DataSet

13.1 Kiến Trúc 2 Lớp

Khi xây dựng các ứng dụng , việc lập trình bắt đầu trở lên phức tạp khi dự án lớn dần. Bởi vậy để dễ quản lý các thành phần của hệ thống, cũng như không bị ảnh hưởng bởi các thay đổi, nhà phát triển hay nhóm các thành phần có cùng chức năng lại với nhau và phân chia trách nhiệm cho từng nhóm để công việc không bị chồng chéo và ảnh hưởng lẫn nhau. Các mô hình lập trình phổ biến : Mô hình 2 lớp (Two Layers) , Mô hình 3 lớp (Three Layers) và n lớp (n Layers)



Mô hình 2 lớp



Mô hình 3 lớp

Trong giáo trình này, chúng ta chỉ khảo sát mô hình 2 lớp để áp dụng và phát triển các ứng dụng ASP.NET. Mô hình 2 lớp được cấu thành từ: **Presentation Layers** và **Business Layers**. Các lớp này sẽ giao tiếp với nhau thông qua các dịch vụ (services) mà mỗi lớp cung cấp để tạo nên ứng dụng, lớp này cũng không cần biết bên trong lớp kia làm gì mà chỉ cần biết lớp kia cung cấp dịch vụ gì cho mình và sử dụng nó mà thôi .

▪ **Presentation Layers :**

Lớp này làm nhiệm vụ giao tiếp với người dùng cuối để thu thập dữ liệu và hiển thị kết quả/dữ liệu thông qua các thành phần trong giao diện người sử dụng. Lớp này sẽ sử dụng các dịch vụ do lớp Business Logic cung cấp. Trong .NET chúng ta có thể dùng **Windows Forms, ASP.NET** hay **Mobile Forms**,... để hiện thực lớp này.

▪ **Business Logic Layer**

Lớp này thực hiện các *nghiệp vụ chính* của hệ thống và *dịch vụ dữ liệu* , sau đó cung cấp các dịch vụ cho lớp **Presentation**. Lớp này chịu trách nhiệm kiểm tra các ràng buộc logic (constraints), các qui tắc nghiệp vụ (Business Rules), sử dụng các dịch vụ bên ngoài khác để thực hiện các yêu cầu của ứng dụng.

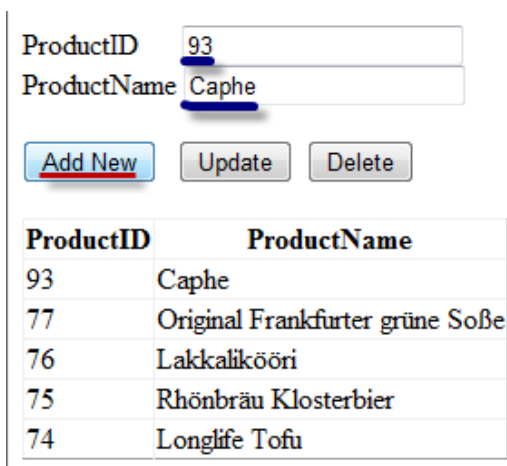
❖ Trong mô hình 3 lớp có thêm Data Layers

▪ Data Layers

Lớp này thực hiện các nghiệp vụ liên quan đến lưu trữ và truy xuất dữ liệu của ứng dụng. Thường lớp này sẽ sử dụng các dịch vụ của các hệ quản trị cơ sở dữ liệu như SQL Server, Oracle, Access, XML... để thực hiện nhiệm vụ của mình. Khi đó **Business Logic Layer** sẽ truy cập dữ liệu từ **Data Layer**.

13.2 Xây dựng ứng dụng ASP.NET theo kiến Trúc 2 Lớp

Trong phần này, chúng ta xây dựng ứng dụng ASP.NET gồm các chức năng Thêm, Xóa, Sửa 1 mặt hàng :



ProductID	ProductName
93	Caphe
77	Original Frankfurter grüne Soße
76	Lakkalikööri
75	Rhönbräu Klosterbier
74	Longlife Tofu

Bước 1

1.1 Tạo ứng dụng ASP.NET đặt tên là SaleApp

1.2 Trên SQL Server 2008 tạo 1 database tên Sale gồm 1 bảng tên Products (ProductID int Primary Key, ProductName varchar(50))

Bước 2 Xây dựng Business Logic Layer

2.1 Thêm vào thư mục *App_Code* 2 lớp với 2 tập tin : ProductData.cs và Products.cs

2.2 Viết code cho từng lớp như sau:

- Lớp Product trong tập tin Product.cs

```
App_Code/Products.cs X
Product
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

public class Product
{
    public int ProductID { get; set; }
    public string ProductName { get; set; }
}
```

- Lớp ProductData trong tập tin ProductData .cs

```
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Collections;
using System.Data;
public class ProductData{
    public ProductData(){ }
    //@:Thay doi chuoi ket noi cho phu hop
    string strConnection = "server=.\SQL2K8;database=Sale;uid=sa;pwd=123";
    public void InsertProduct(Product p) {
        SqlConnection cnn = new SqlConnection(strConnection);
        string SQLInsert =
            "Insert Products(ProductID,ProductName) values(@ProID,@ProName)";
        SqlCommand cmd = new SqlCommand(SQLInsert, cnn);
        cmd.Parameters.Add("@ProID", p.ProductID);
        cmd.Parameters.Add("@ProName", p.ProductName);
        try{
            cnn.Open();
            cmd.ExecuteNonQuery();
        }
        catch(Exception ex) {
            throw new Exception("Error:"+ex.Message);
        }
        finally {
            cnn.Close();
        }
    }
}
```

```
public void UpdateProduct(Product p) {
    SqlConnection cnn = new SqlConnection(strConnection);
    string SQLUpdate =
        "Update Products set ProductName = @Name Where ProductID=@ProID";
    SqlCommand cmd = new SqlCommand(SQLUpdate, cnn);
    cmd.Parameters.Add("@Name", p.ProductName);
    cmd.Parameters.Add("@ProID", p.ProductID);
    try{
        cnn.Open();
        cmd.ExecuteNonQuery();
    }
    catch{
        throw new Exception("Error");
    }
    finally {
        cnn.Close();
    }
}

public void DeleteProduct(Product p) {
    SqlConnection cnn = new SqlConnection(strConnection);
    string SQLInsert =
        "Delete Products where ProductID=@ProID";
    SqlCommand cmd = new SqlCommand(SQLInsert, cnn);
    cmd.Parameters.AddWithValue("@ProID", p.ProductID);
    try{
        cnn.Open();
        cmd.ExecuteNonQuery();
    }
    catch{
        throw new Exception("Error");
    }
    finally{
        cnn.Close();
    }
}
```

```

public List<Product> GetProducts(){
    List<Product> data = new List<Product>();
    string SQL = "select ProductID,ProductName from Products";
    SqlConnection cnn = new SqlConnection(strConnection);
    SqlCommand cmd = new SqlCommand(SQL, cnn);
    cnn.Open();//Phải mở kết nối
    SqlDataReader rd = cmd.ExecuteReader
        (CommandBehavior.CloseConnection);
    if (rd.HasRows) //nếu có dữ liệu trả về từ câu lệnh Select
    {
        while (rd.Read()) //đọc từng dòng từ bảng Products
        {
            Product p = new Product(){
                ProductID=int.Parse(rd["ProductID"].ToString()),
                ProductName = rd.GetString(1)
            };
            data.Add(p);
        }
    }
    //sử dụng LINQ to Object để sắp xếp ProductID giảm dần và lấy 5 product
    return data.OrderByDescending
        (p=>p.ProductID).Take(5).ToList();
}

```

Bước 3 Xây dựng Presentation Layers

3.1 Thiết kế giao diện cho trang Default.aspx như hình sau :

The screenshot shows a web page titled "Default.aspx" with a "Start Page" tab. The page contains a form with two input fields: "ProductID" and "ProductName". Below these fields are three buttons: "Add New", "Update", and "Delete". At the bottom of the form is a table with three columns labeled "Column0", "Column1", and "Column2". The table contains five rows of data, each with the value "abc" in all three columns.

Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

[illegible]

Default.aspx.cs như sau:

```
public partial class _Default : System.Web.UI.Page {
    ProductData pdata = new ProductData();
    protected void Page_Load(object sender, EventArgs e) {
        if (!IsPostBack)
            LoadData();
    }
    public void LoadData() {
        gvProducts.DataSource = pdata.GetProducts();
        gvProducts.DataBind();
    }
    protected void btnAddNew_Click(object sender, EventArgs e) {
        int ProID = int.Parse(txtProductID.Text);
        Product p = new Product { ProductID = ProID, ProductName = txtProductName.Text };
        pdata.InsertProduct(p);
        LoadData();
    }
    protected void btnUpdate_Click(object sender, EventArgs e) {
        int ProID = int.Parse(txtProductID.Text);
        Product p = new Product { ProductID=ProID, ProductName=txtProductName.Text };
        pdata.UpdateProduct(p);
        LoadData();
    }
    protected void btnDelete_Click(object sender, EventArgs e) {
        int ProID = int.Parse(txtProductID.Text);
        Product p = new Product { ProductID=ProID };
        pdata.DeleteProduct(p);
        LoadData();
    }
}
```

Bước 4 : Nhấn Ctrl+F5 để chạy ứng dụng và kiểm tra các chức năng .

ProductID

ProductName

ProductID	ProductName
93	Caphe
77	Original Frankfurter grüne Soße
76	Lakkaikööri
75	Rhönbräu Klosterbier
74	Longlife Tofu

13.3 ObjectDataSource control

Khi ta phát triển ứng dụng theo mô hình 2 lớp hay 3 lớp, tất cả chức năng của ứng dụng sẽ thực thi thông qua Bussiness Layers và chúng ta cũng phải viết code cho các sự kiện của các controls ở tầng Presentation. ObjectDataSource controls sẽ thực thi các chức năng của các Data controls một cách tự động thông qua các phương thức đã khai báo trong lớp ProductData . Như vậy , trang Default.aspx thay vì thiết kế và viết code như trên sẽ được thiết kế lại như sau :

FormView - FormView1

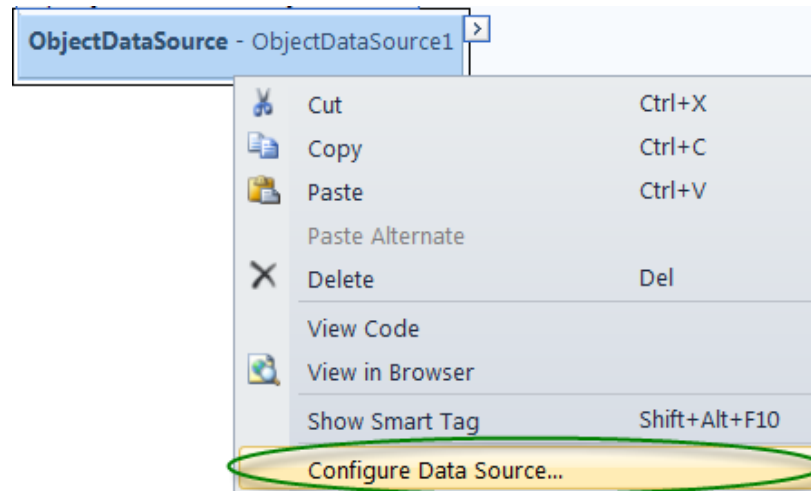
Right-click or choose the Edit Templates task to edit template content.
The ItemTemplate is required.

Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

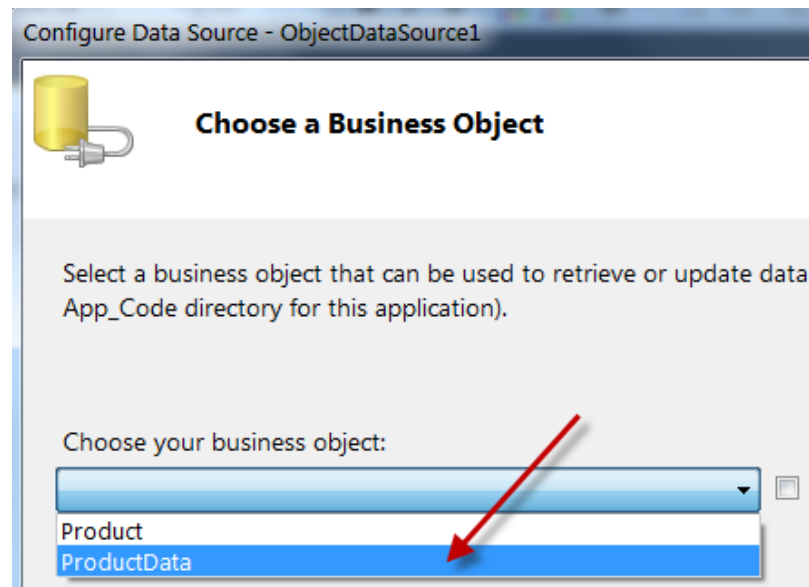
ObjectDataSource - ObjectDataSource1

- Cấu hình cho ObjectDataSource1 theo các bước sau đây :

Bước 1. Chọn ObjectDataSource1 -> nhấp phải chuột và chọn **Config Data Source...**



Bước 2. Trên hộp thoại Config Data Source , chọn ProductData | nhấp Next để cấu hình các lệnh select , insert , update và delete và nhấn Finish để kết thúc



SELECT UPDATE INSERT DELETE

Choose a method of the business object that returns data to associate with the SELECT operation. The method can return a DataSet, DataReader, or strongly-typed collection.

Example: GetProducts(Int32 categoryId), returns a DataSet.

Choose a method:

GetProducts(), returns List<Product>

Method signature:

GetProducts(), returns List<Product>

Bước 3.

- Thiết lập thuộc tính DataSource là ObjectDataSource1 cho GridView control và FormView Control.
- Thiết lập thuộc tính DataKeyNames là ProductID cho FormView control

ProductID: 0
 ProductName: abc
[Edit](#) [Delete](#) [New](#)

ProductID	ProductName
0	abc
1	abc
2	abc
3	abc
4	abc

ObjectDataSource - ObjectDataSource1

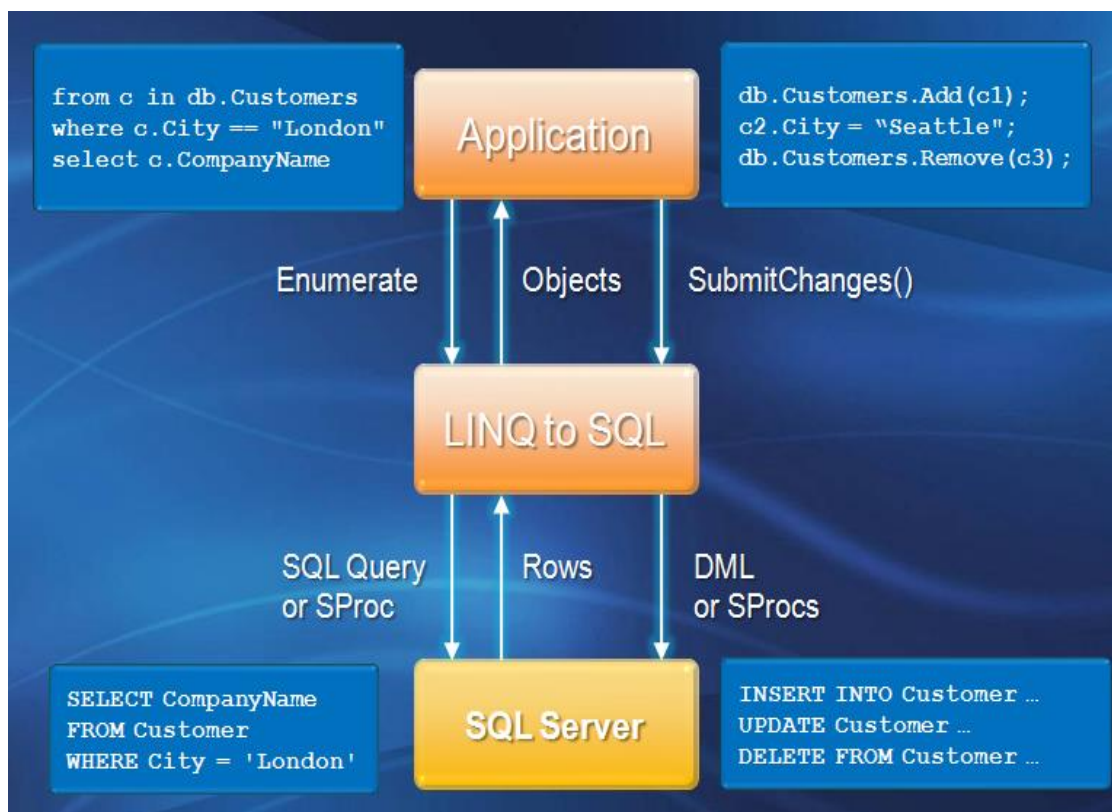
Bước 4. Nhấn Ctrl+F5 để chạy ứng dụng và kiểm tra các chức năng

ProductID: 77
 ProductName: Original Frankfurter grüne Soße
[Edit](#) [Delete](#) [New](#)

ProductID	ProductName
77	Original Frankfurter grüne Soße
76	Lakkalikööri
75	Rhönbräu Klosterbier
74	Longlife Tofu
73	Röd Kaviar

13.4 LINQ to SQL

LINQ to SQL là một phiên bản hiện thực hóa của O/RM (object relational mapping) có bên trong .NET Framework bản "Orcas" (nay là .NET 4.0) , nó cho phép các nhà phát triển mô hình hóa một cơ sở dữ liệu dùng các lớp .NET. Sau đó có thể truy vấn cơ sở dữ liệu (CSDL) dùng LINQ, cũng như thực hiện các thao tác cập nhật , thêm , xóa dữ liệu từ đó.

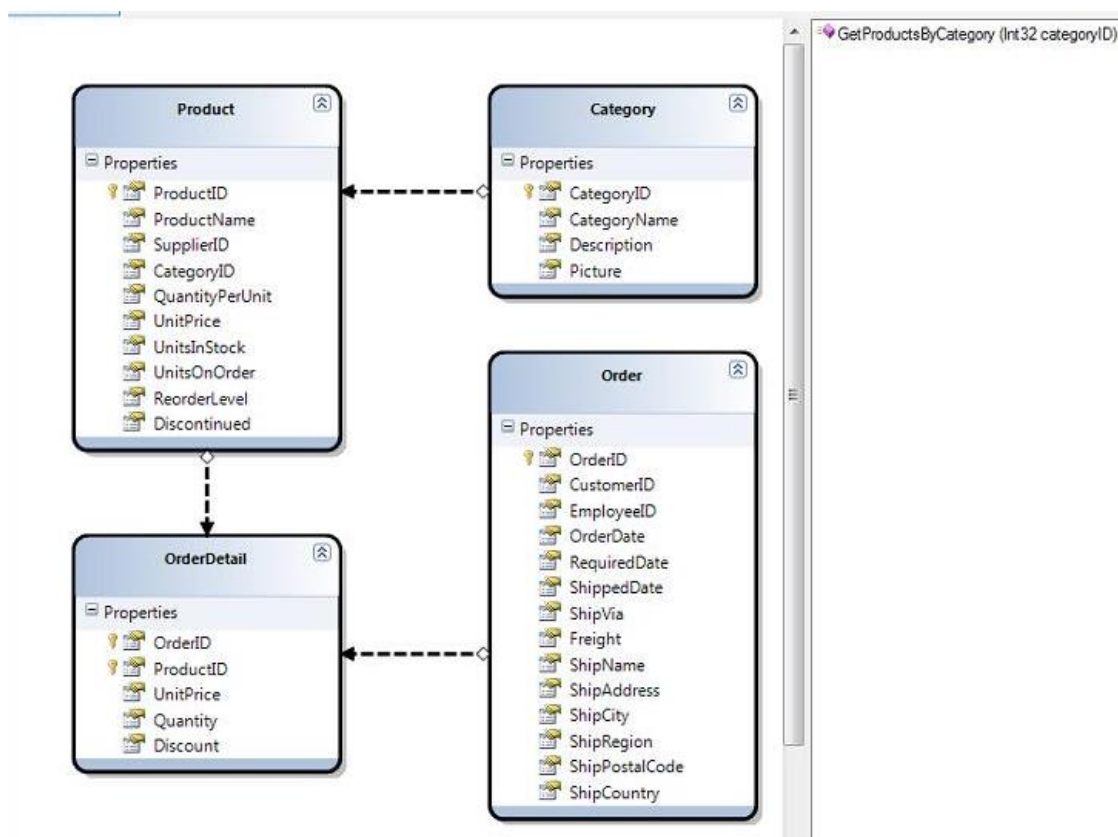


Mô hình hoạt động của LINQ to SQL

13.4.1 Mô hình hóa cơ sở dữ liệu dùng LINQ to SQL

LINQ to SQL hỗ trợ đầy đủ transaction, view và các stored procedure (SP). Nó cũng cung cấp một cách dễ dàng để thêm khả năng kiểm tra tính hợp lệ của dữ liệu và các quy tắc vào trong mô hình dữ liệu.

- Visual Studio "Orcas" đã tích hợp thêm một trình thiết kế LINQ to SQL như một công cụ dễ dàng cho việc mô hình hóa một cách trực quan các CSDL dùng LINQ to SQL.
- Bằng cách dùng trình thiết kế LINQ to SQL, chúng ta có thể dễ dàng tạo một mô hình cho CSDL mẫu "Northwind" sau đây:



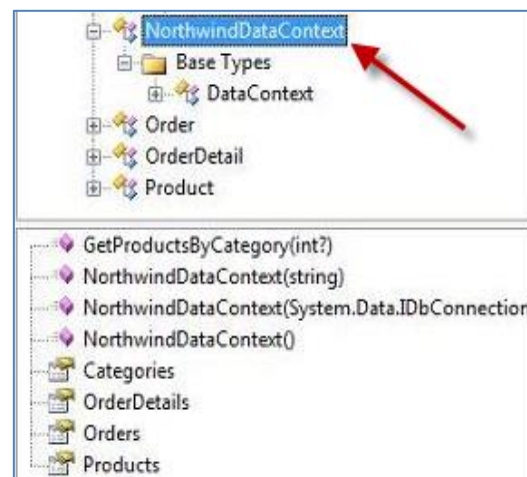
- Mô hình LINQ to SQL ở trên định nghĩa bốn lớp thực thể: Product, Category, Order và OrderDetail. Các thuộc tính của mỗi lớp ánh xạ vào các cột của bảng tương ứng trong CSDL. Mỗi instance của một lớp biểu diễn một dòng trong bảng dữ liệu.
- Các mũi tên giữa bốn lớp thực thể trên biểu diễn quan hệ giữa các thực thể khác nhau, chúng được tạo ra dựa trên các mối quan hệ primary-key/foreign-key trong CSDL. Hướng của mũi tên chỉ ra mối quan hệ là một – một hay một – nhiều. Các thuộc tính tương ứng sẽ được thêm vào các lớp thực thể trong các trường hợp này. Lấy ví dụ, lớp Category ở trên có một mối quan hệ một nhiều

với lớp Product, điều này có nghĩa nó sẽ có một thuộc tính "Categories" là một tập hợp các đối tượng Product trong Category này. Lớp Product cũng sẽ có một thuộc tính "Category" chỉ đến đối tượng "Category" chứa Product này bên trong.

- Bảng các phương thức bên tay phải bên trong trình thiết kế LINQ to SQL ở trên chứa một danh sách các Store Procedure (SP) để tương tác với mô hình dữ liệu của chúng ta. Trong mô hình trên ta đã thêm một thủ tục có tên "GetProductsByCategory" mà nó nhận vào một categoryID và trả về một dãy các Product. Chúng ta sẽ tìm hiểu cách gọi được thủ tục này trong một đoạn code tiếp theo .

13.4.2 Lớp DataContext trong LINQ to SQL

Khi chúng ta lưu màn hình thiết kế LINQ to SQL, Visual Studio sẽ lưu các lớp .NET biểu diễn các thực thể và quan hệ bên trong CSDL mà chúng ta vừa mô hình hóa. Cứ mỗi một file LINQ to SQL chúng ta thêm vào solution, một lớp DataContext sẽ được tạo ra, nó sẽ được dùng khi cần truy vấn hay cập nhật lại các thay đổi. Lớp DataContext được tạo sẽ có các thuộc tính để biểu diễn mỗi bảng được mô hình hóa từ CSDL, cũng như các phương thức cho mỗi Store Procedure mà chúng ta đã thêm vào.



13.4.3 Một số thao tác dữ liệu trong LINQ to SQL

Một khi đã mô hình hóa CSDL dùng trình thiết kế LINQ to SQL, chúng ta có thể dễ dàng viết các đoạn lệnh để làm việc với nó. Dưới đây là một số ví dụ minh họa về các thao tác chung khi xử lý dữ liệu :

- **Lấy danh sách các Products**

Đoạn lệnh dưới đây dùng cú pháp LINQ để lấy về một tập IEnumerable các đối tượng Product. Các sản phẩm được lấy ra phải thuộc phân loại "Beverages"

```
NorthwindDataContext db = new NorthwindDataContext();  
var products = from p in db.Products  
                where p.Category.CategoryName == "Beverages"  
                select p;
```

- **Cập nhật thông tin Product**

Đoạn lệnh dưới đây dùng cú pháp LINQ để cập nhật UnitPrice và UnitsInStock của Product

```
NorthwindDataContext db = new NorthwindDataContext();  
Product product = db.Products.Single(p => p.ProductName == "Chai" );  
product.UnitPrice = 99;  
product.UnitsInStock = 5;  
db.SubmitChanges();
```

- **Thêm mới 1 Category**

Đoạn lệnh dưới đây dùng cú pháp LINQ để thêm 1 Category vào bảng Categories

```
NorthwindDataContext db = new NorthwindDataContext();  
// Create new Category  
Category category = new Category();  
category.CategoryName = "Scott's Toys";  
// Add category to database and save changes  
db.Categories.Add(category);  
db.SubmitChanges();
```

- **Xóa 1 Product**

Đoạn lệnh dưới đây dùng cú pháp LINQ để xóa 1 Product trong bảng Products

```
NorthwindDataContext db = new NorthwindDataContext();  
var p = from p in db.Products  
        where p.ProductID= 1  
        select p;  
db.Products.DeleteOn Submit(p);  
db.SubmitChanges();
```

- **Gọi 1 thủ tục (store procedure)**

Đoạn lệnh dưới đây dùng cú pháp LINQ gọi 1 store procedure

```
NorthwindDataContext db = new NorthwindDataContext();
var products = db.GetProductsByCategory(5);
foreach (Product product in products)
{
    //Viết code ở đây....
}
```

13.4.4 Các bước phát triển ứng dụng LINQ to SQL

Bước 1

- 1.1 Tạo ứng dụng ASP.NET đặt tên là TestLINQtoSQL gồm trang Default.aspx có giao diện giống ví dụ trong mục **13.2**

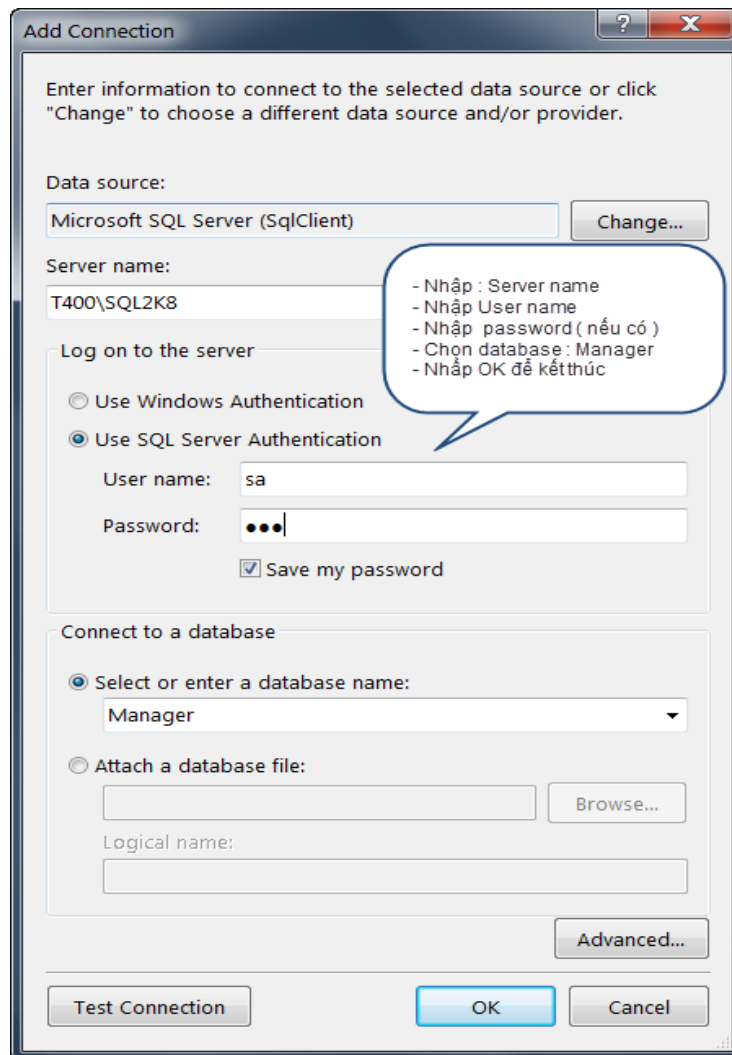
Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

- 1.2 Trên SQL Server 2008 tạo 1 cơ sở dữ liệu tên Manager gồm 1 bảng tên Products (ProductID int Primary Key, ProductName varchar(50))

dbo.Products: Table(ibm.Manager)			
	Column Name	Data Type	Allow Nulls
	ProductID	int	<input type="checkbox"/>
	ProductName	varchar(50)	<input type="checkbox"/>

Bước 2 Xây dựng lớp LINQ to SQL2.1 Tạo DataConnection để kết nối với cơ sở dữ liệu *Manager*

Trên thanh công cụ *Server Explorer* | *Data Connection*, nhấp phải chuột chọn *Add New Connection* (để mở hộp thoại *Server Explorer* vào *View* | *Server Explorer*)



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:
T400\SQL2K8

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...

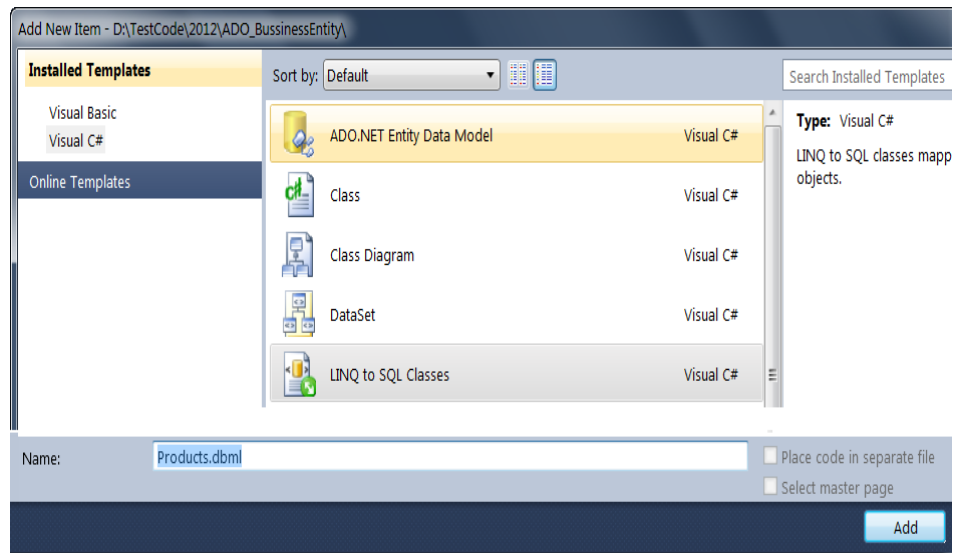
Logical name:

Advanced...

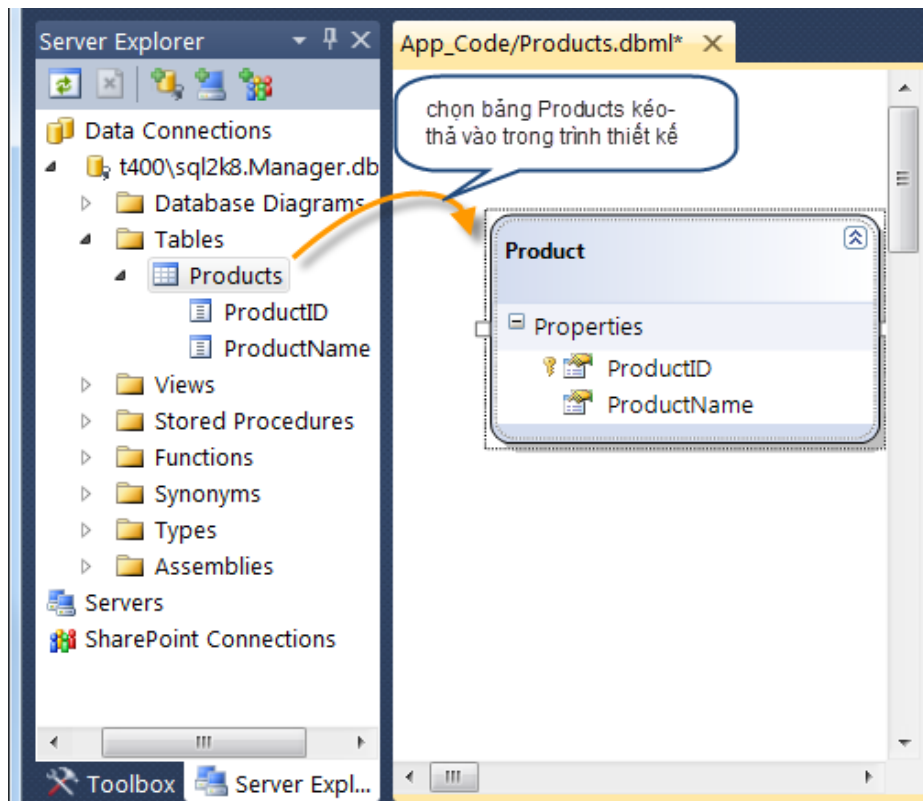
Test Connection OK Cancel

- Nhập : Server name
- Nhập User name
- Nhập password (nếu có)
- Chọn database : Manager
- Nhập OK để kết thúc

2.2 Vào menu *Project | Add New Items* , trên hộp thoại Add New Items , chọn LINQ to SQL Classes và đặt tên **Products.dbml** , nhấp Add để kết thúc.



Từ hộp thoại Server Explorer chọn bảng Products kéo thả vào Products.dbml và nhấn Shift+F6 để biên dịch ứng dụng.



Bước 3 Viết code cho sự kiện *Page_Load* và sự kiện *Click* của các Button controls trong tập tin Default.aspx.cs

```
protected void Page_Load(object sender, EventArgs e){
    if (!IsPostBack)
        LoadData();
}
public void LoadData() {
    ProductsDataContext pdata = new ProductsDataContext();
    gvProducts.DataSource = pdata.Products;
    gvProducts.DataBind();
}
protected void btnAddNew_Click(object sender, EventArgs e){
    ProductsDataContext pdata = new ProductsDataContext();
    int ProID = int.Parse(txtProductID.Text);
    Product p = new Product{
        ProductID= ProID,
        ProductName = txtProductName.Text };
    pdata.Products.InsertOnSubmit(p);
    pdata.SubmitChanges();
    LoadData();
}

protected void btnUpdate_Click(object sender, EventArgs e){
    ProductsDataContext pdata = new ProductsDataContext();
    int ProID = int.Parse(txtProductID.Text);
    Product p = pdata.Products.SingleOrDefault(pro => pro.ProductID == ProID);
    p.ProductName = txtProductName.Text;
    pdata.SubmitChanges();
    LoadData();
}

protected void btnDelete_Click(object sender, EventArgs e){
    ProductsDataContext pdata = new ProductsDataContext();
    int ProID = int.Parse(txtProductID.Text);
    Product p = pdata.Products.SingleOrDefault(pro => pro.ProductID == ProID);
    pdata.Products.DeleteOnSubmit(p);
    pdata.SubmitChanges();
    LoadData();
}
```

Bước 4. Nhấn Ctrl+F5 để chạy ứng dụng và kiểm tra các chức năng

ProductID	ProductName
1	caphe

13.5 LinqDataSource control

LinqDataSource control cho phép thao tác dữ liệu tự động với các Data controls như : GridView, FormView, ... thông qua LINQ to SQL Classes thay vì chúng ta phải viết code cho trang các controls trên trang ASP.NET ,do đó trang Default.aspx ở trên chúng ta thiết kế lại như hình dưới đây.

Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

LinqDataSource - LinqDataSource1

Để cấu hình cho LinqDataSource control ta thực hiện các bước sau đây :

Bước 1. Chọn LinqDataSource1 , nhấp phải chuột chọn *Configure Data Source...*

Bước 2. Trên hộp thoại Configure Data Source chọn ProductsDataContext , nhấp next để tiếp tục.

Configure Data Source - LinqDataSource1

Choose a Context Object

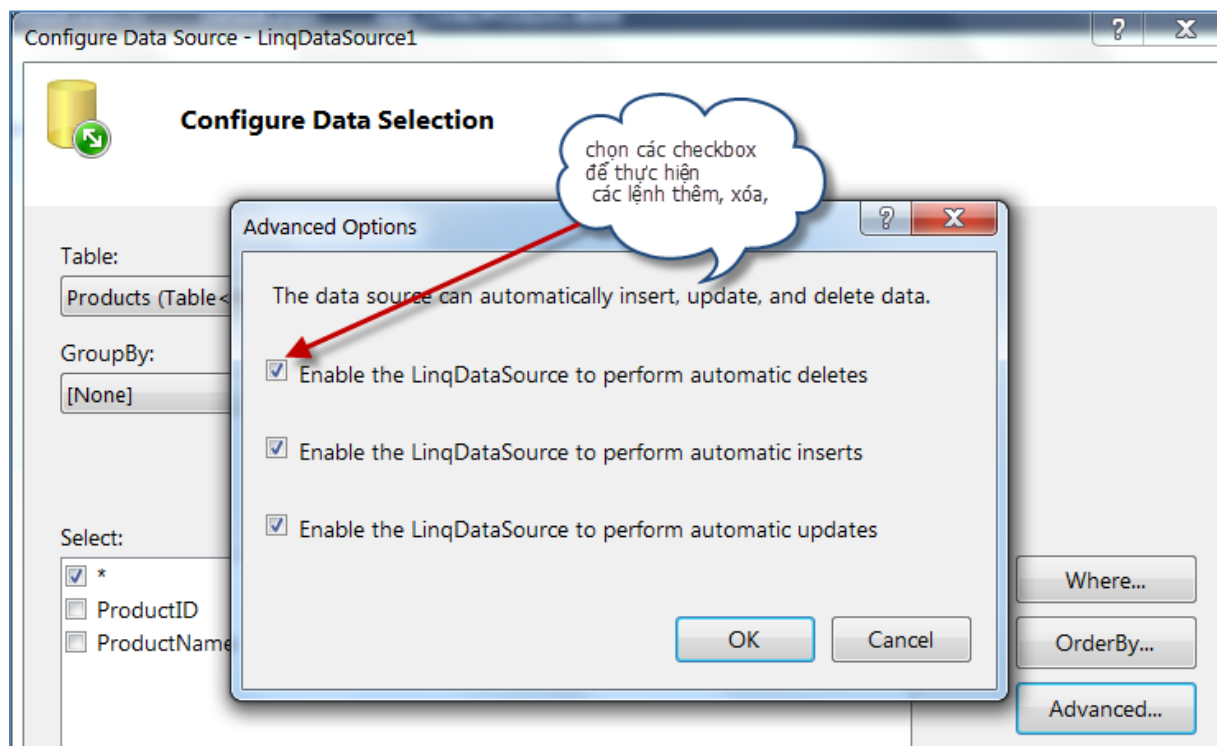
Select a context object that can be used to retrieve or update data.

☐ Show only DataContext objects

Choose your context object:

- Product
- Product
- ProductsDataContext**

Bước 3. Nhấp vào nút *Advanced* và chọn các checkbox : insert, update, delete . Nhấp OK -> Finish để kết thúc.



Bước 4. Thiết lập giá trị cho thuộc tính DataSource của FormView1 và GridView1 là LinqDataSource1.

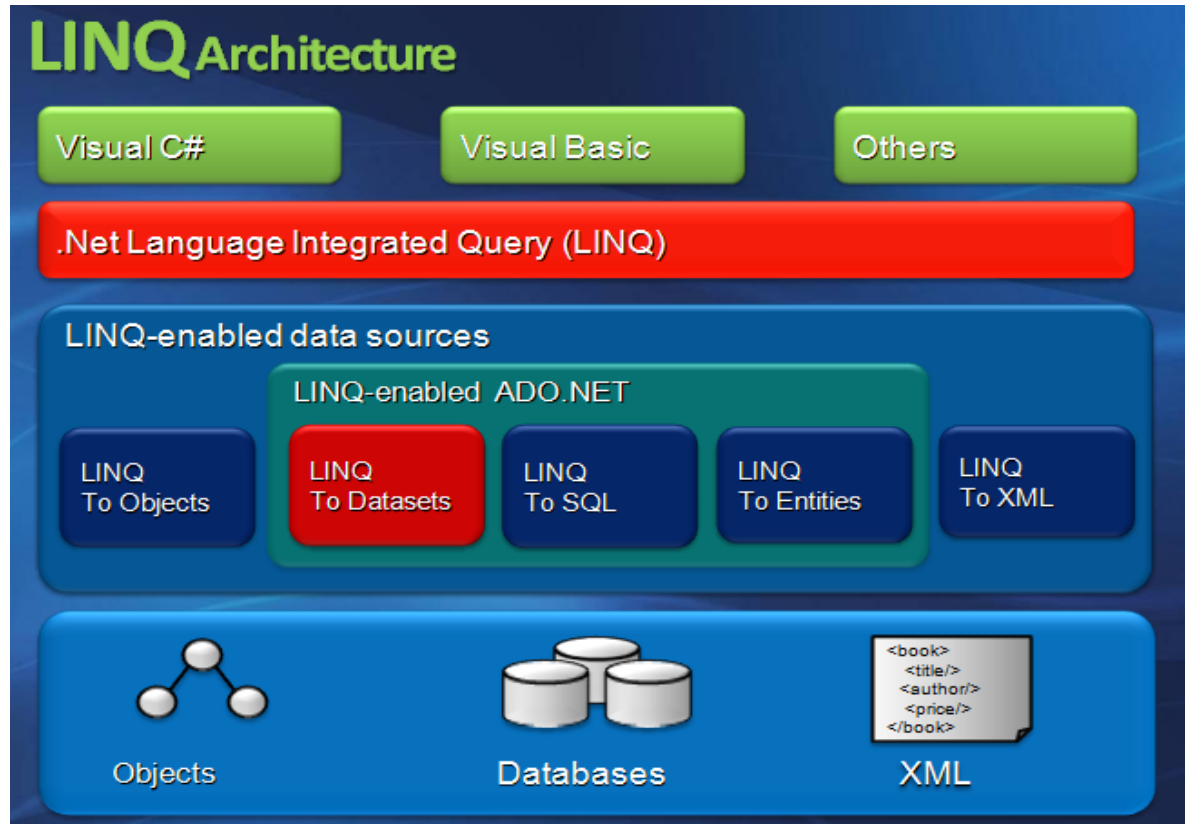
Bước 5. Nhấn Ctrl+F5 để chạy ứng dụng và kiểm tra kết quả .

ProductID: 3
ProductName: Caphe
[Edit](#) [Delete](#) [New](#)

ProductID	ProductName
3	Caphe

13.6 LINQ to DataSet

LINQ to DataSet dùng để thao tác với các cơ sở dữ liệu khác chẳng hạn như : Access, Oracle, Excel,.. sử dụng cú pháp chung của LINQ



Để tìm hiểu về LINQ to DataSet chúng ta từng bước xây dựng ứng dụng tìm thông tin mặt hàng theo đơn giá

Bước 1

- 1.1 Tạo ứng dụng ASP.NET đặt tên là TestLINQtoDataSet gồm trang SearchProducts.aspx có giao diện như hình dưới đây :

ProductID

Search


Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

Giao diện trang SearchProduct.aspx

```
<form id="form1" runat="server">
<div>
    ProductID
    <asp:TextBox ID="txtProductID" runat="server"></asp:TextBox>
    <br />
    <asp:Button ID="btnSearch" runat="server" Text="Search" />
    <br />
    <asp:GridView ID="gvProductDetail" runat="server">
    </asp:GridView>
</div>
</form>
```

Mã HTML

1.2 Tạo cơ sở dữ liệu tên Manager gồm 1 bảng tên Products

dbo.Products: Table(ibm.Manager)			
	Column Name	Data Type	Allow Nulls
	ProductID	int	<input type="checkbox"/>
	ProductName	varchar(50)	<input type="checkbox"/>

Bước 2.. Viết code cho nút Search để tìm Product theo ProductID

```
protected void btnSearch_Click(object sender, EventArgs e)
{
    string chuoKetNoi = "server=SwordLake\\sql2k8;database=Manager;uid=sa;pwd=123";
    string cauLenhSQL = "select ProductID,ProductName from Products";
    SqlDataAdapter da = new SqlDataAdapter(cauLenhSQL, chuoKetNoi);
    DataSet dsProducts = new DataSet();
    da.Fill(dsProducts);

    int ProID = int.Parse(txtProductID.Text);
    var product = from p in dsProducts.Tables[0].AsEnumerable()
                  where p.Field<int>("ProductID") == ProID
                  select new
                  {
                      ProductID = p.Field<int>("ProductID"),
                      ProductName = p.Field<string>("ProductName")
                  };
    gvProductDetail.DataSource = product;
    gvProductDetail.DataBind();
}
```

Bước 3. Chạy ứng dụng và kiểm tra kết quả

ProductID

Search

ProductID	ProductName
1	chai

-----oOo-----