# DAL(Data Access Layer)

**نویسنده : مهدی نگاهی**

## اتصال به دیتابیس

در اینجا باید شما یک اتصال به DB خود برقرار کنید که همانطور که گفتم دیتا بیس ما از Northwind است و از SQL-Server 2005 هم استفاده میکنیم.....

در پنجره Server Explorer روی Add New Connection کلیک کرده و در قسمت Server Nameنام سرور خود را وارد کنید ور قسمت Select Or enter database name پایگاه داده Northwind را وارد کنید برای اطمینان روی Test connection کلیک کرده تا از اتصال خود اطمینان داشته باشید .

## مفهوم DAL(data access layer)

وقتی شما از لایه بندی در پروژه خود استفاده نمیکنید شما تمام کارهایتان در لایه presentation انجام می دهید . منظور از لایه Presentation همان صفحات ASP.NET هستند . ادر واقع شما با این کار لایه DAL را داخل لایه presentation قرار می دهید که این کار مشکلات خاص خود را دارد که عمده ترین این مشکلات به ترتیب زیر است :

1- اگر شما 10 صفحه داشته باشد که کار با DB را انجام میدهد برای تمام این 10 صفحه باید کدهای مربوط به DB را بنویسید
2- تغییرات مشکل است . مثلا فرض کینید بنا به دلایلی نام یک جدول عوض میشود تصور کنید که این تغییر چقدر شما را عذاب می دهد

البته مشکلات یک لایه کار کردن به اینجا ختم نمیشود ولی خوب خودتان دنبال این مشکلات بروید :چشمک:

برای رفع این مشکلات شما باید لایه Presentation و DAL را از هم جدا کنید.
تام کدهایی که به مشخص در زیر Data Sourceهستند مثل توابعی که کار Insert,Update و Delete ....را انجام میدهند باید در لایه DAL باشد پس یه طور کلی DAL حاوی متدهایی است که برای کار با DB مورد نیاز است.
بطور مثال در پایگاه داده ما جدولی یه نام Products و Categories وجود دارد درDAL ما متدهای با ویژگی و نامهای زیر وجود دارد (البته بخش از متدها در اینجا آورده شده است)

1-()GetCategories که اطلاعاتی درباره تمام طبقه بندی محصولات میدهد.
2-()GetProducts اطلاعاتی درباره تمام محصولات را میدهد.
3-GetProductByCategorieID(categorieID)

این متده وقتی فراخوانی میشوند به DB وصل شده و Query مورد نظر را اجرا کرده و نتیجه را برمی گرداند. البته نوع این نتجه برگشت داده شده مهم است (type = نوع)
.این نوع میتواند یک DataSet ساده یا یک DataReader باشد.اما مطلوب ان است که نتیجه برگشتی از*strongly-typed objects*s استفاده کند
.

***strongly-typed objects : نوع داده ای است که شمای آن قبل کمپایل معلوم باشد***

**loosely-typed object : این نوع داده قبل از زمان کامپایل و اجرا شمایش معلوم نیست.**

به طور مثال DataReader و DataSet معمولی یک **loosely-typed object** هستند و برای دسترسی به یک ستون مشخص از یک **loosely-typed DataTable**باید از Syntaxزیر استفاده کرد

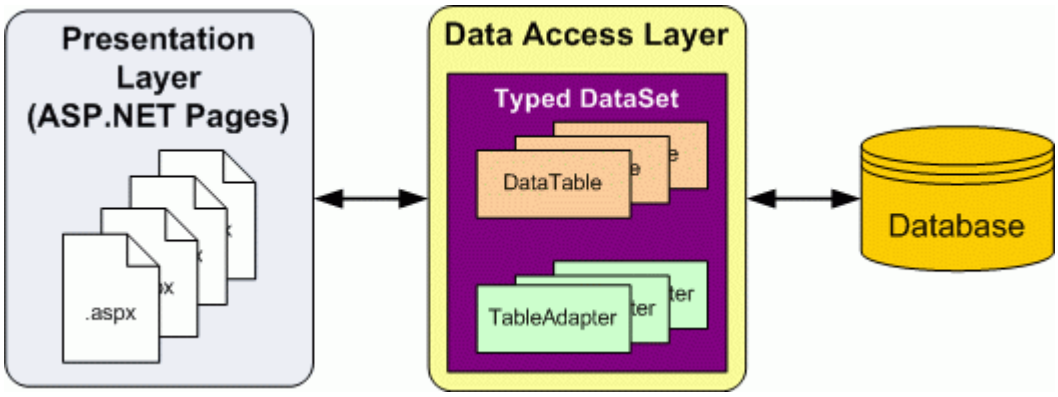
*DataTable*.Rows[*index*]["*columnName*"]

ولی برای دسترسی به یک ستون مشخص از ***strongly-typed DataTable*** باید از Syntax زیر استفاده کرد

*DataTable*.Rows[*index*].*columnName*

برای داشتن یک **strongly-typed object** میتونید خودتان یک کلاس تعریف کنید و بطور خاص تمام مفاهیم گفته شده را پیاده سازی کنید اما را آسانتری نیز وجود دارد که آنهم این است که Typed DataSet درست کنید که خاصیتهای بالا را داشته باشد .

در آخر هم این تصویر DAL ونحوه قرار گرفتن آن در لایه های ما است .



# توضیح درباره typed DataSet

Typed DataSet برای شما به وسیله Visual studio ساخته میشود و شما نیاز به کد نوشتن اضافی ندارید البته شاید این تصور راکنید که در ورژن قبلی VS، DataSet زیاد جالب نبودند ولی در ورژن 2005 روی DataSet خیلی کار شده است اگر تا آخر این تاپیک همراه ما باشید میبینید چه امکانات جالبی در اختیار برنامه نویس میگذارد .یک typed DataSet خودش تشکیل شده از کلاسهایی مانند DataTable,DataRow و همچینین چیزی که اضافه شده در دیتا ست جدید کلاسی به نام TableAdapter است که کار همان Adapter ورژن قبلی را انجام میدهد ولی با شرایطی متفاوت که خود مشاهده میکنید.

یک Typed DataSetمثلstrongly -typed collection of data بکار می رود (معادل فارسی پیدا نکردم) . آنها ترکیب نمونه هایی از **strongly-typed DataTable** که هر کدام از این نکنونه ها حاوی **strongly-type DataRow** هستند . ما برای هر جدول در دیتا بیس که میخواهیم روی این جداول کار کنیم یک **strongly-typed DataTable** میسازیم . در ایجا ما برای جدول products میسازیم.

توجه : **strongly-typed DataTable حاوی هیچ گونه اطلاعاتی برای دستیابی به دیتاهای درون جدول متناظر خود نیست ( بطور ساده مثلا strongly-typed DataTable که برای جدول products دیتا یبس ساخته ایم نمیداند چگونه دیتا از دیتابیس وارد این strongly-typed DataTable میشود). برای بازیابی دیتا از دیتابیس و پرکردن dataTable از TableAdapter استفاده میشود .**
**به طور مثال برای products DataTable ، TableAdapter حاوی متد هایی است که کار پر کردن دیتا تیبل ما را انجام می دهد نمونه ای از متد ها عبارتند از**

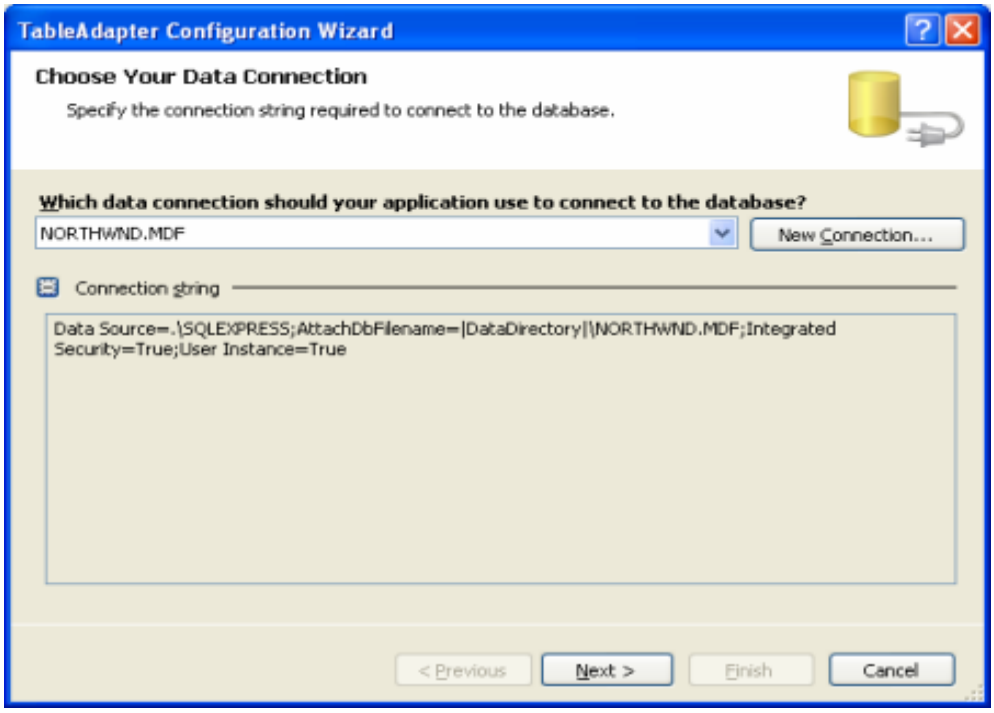
```
GetProducts()
GetProductByCategoryID(categoryID)
```

متدها از لایه presentation صدا زده میشوند

# ساخت یک Typed DataSet و یک Table Adapter

برای شروع ساخت یک DAL شما باید یک Typed DataSet به پروژه خود اضافه کنید برای این کار روی پروژه خود راست کلیک کرده و Add New Item را کلیک کنید و از صفحه باز شده DataSet را انتخاب کنید و اسم آن را Northwind بگذارید و روی دکمه add کلیک کنید . بعد از کلیک به شما پیغام داده می شود که DataSet را در فولدرApp_code اضافه شود شما این را قبول میکنید.

سپس TableAdapter Configuration Wizard نمایان میشود شما باید اتصال به دیتا بیس که در بخش قبل ساختیم از combo Box انتخاب کنید

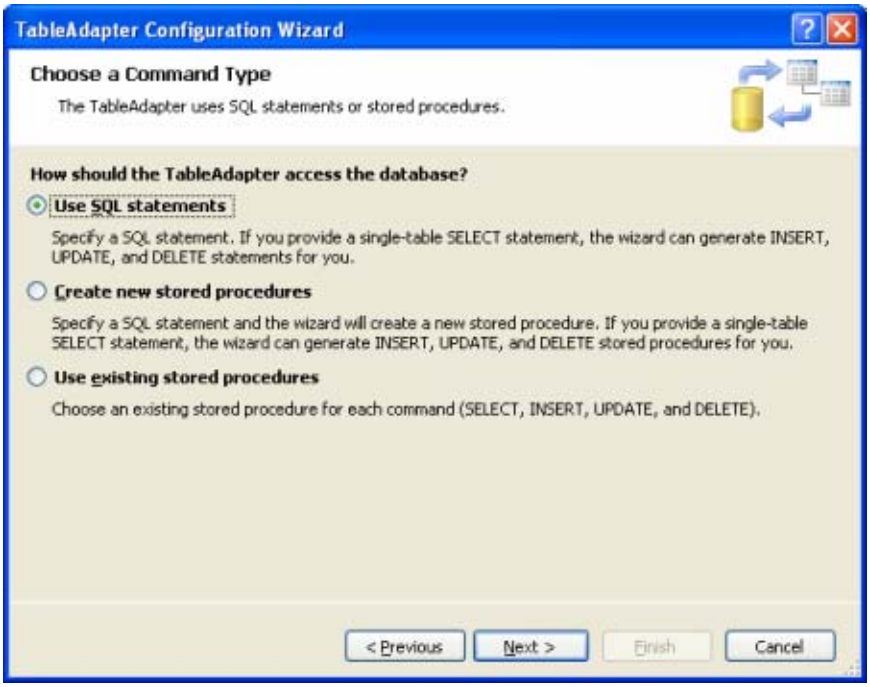



Next

در صفحه بعد از شما میپرسد که مایل هستید که Connection String را به فایل Web.Config اضافه کنم که شما قبول میکنید(در این صفحه تغییری نمیدهید)

Next

در این صفحه شما باید طریقه دستیابی به database را از این سه گزینه انخاب کنید که ما گزینه اول را انتخاب میکنیم .(شکل 2)

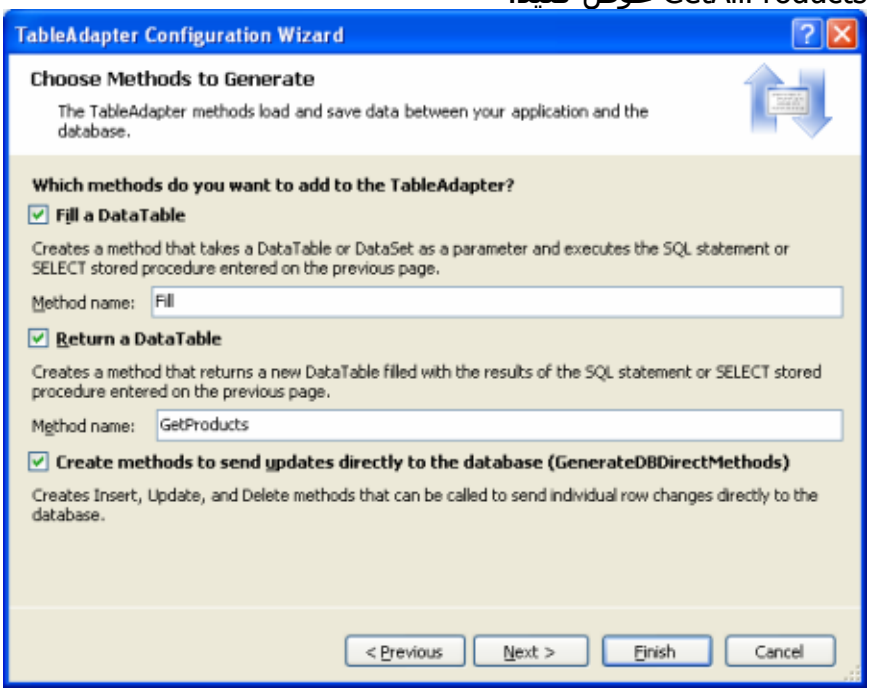



Next

در این صفحه باید Query اس کیو ال خود را بنوسید می توانید روی Query Bulder کیکک کنید از پنجره باز شده جدول Products را add کنید  و سپس روی close کلیک کرده و تمام ستونها را انتخاب کنید ( توجه برای راحتی ار انتخاب * به جای تمامه ستونها استفاده نکنید) وسپس Ok را بزنید (شکل3)
روی Advance Option هم کلیک کنید و مطمئن شوید که خط اول انتخاب شده

Next

این جا جالترین و جدیدترین جا است انجا را کسی در VS2003 ندیده
در اینجا ما نام متدهای که در TableAdapter هستند را انتخاب میکنیم نام GetData را به GetAllProducts عوض کنید.




تمام الان شما در  App_Code/Northwind.xsd شمایل جدول products را میبینید.این الان یک DAL بسیار ساده است که در بخشهای بعدی آن را کامل میکنیم

# استفاده از typed-DataSet

خوب دوستان حالا باید از این کاری هایی که کردیم یک استفاده کنیم خوب در یک صفحه ASP.NET یک GridView و در Page_Load صفحه تان کدهای زیر را وارد کنید

```
using NorthwindTableAdapters;
```

Page_Load

```
ProductsTableAdapter productsAdapter = new  ProductsTableAdapter();
```

```
GridView1.DataSource = productsAdapter.GetAllProducts();
GridView1.DataBind();
```

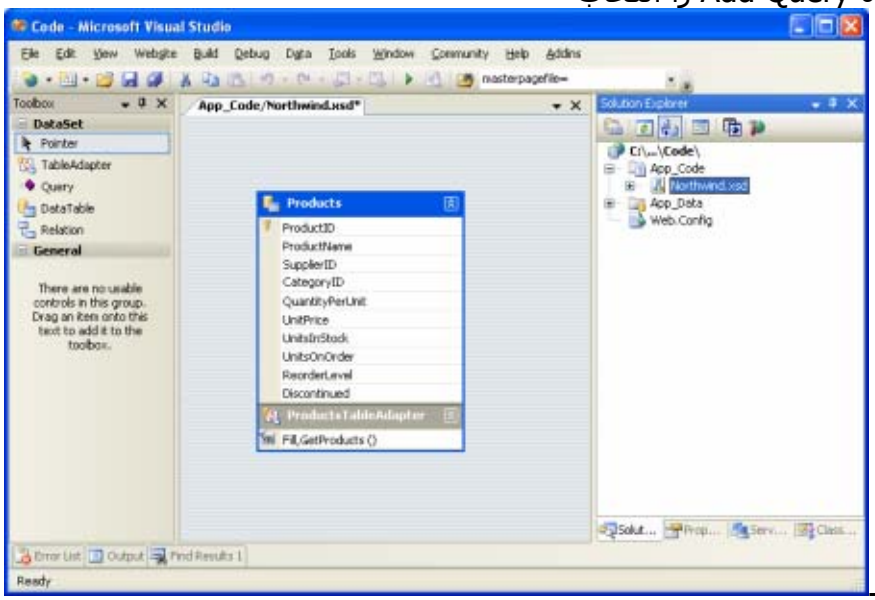
حالا اجرا کنید !!! میبینید که GridView شما دیتا ها نمایش داده شده اند .همانطور که میبینید در این کد ما حتی یک کلاس از کلاسهای ADO.NET نمونه سازی نکردیم و حتی کانکشن به DataBase خود هم باز نکردیم.شما میتوانید با این کد حتی DropDownList ،CheckBoxها را هم بایند کنید .

# اضافه نمودن متدها با پارامتر ورودی DAL

خوب تا اینجا ما DAL خود را ساخته ایم ولی مطمئنا این DAL ما دردی از ما دوا نمیکند .در حال حاضر productsTableAdapter ما فقط یک متد دارد که آنهم تمام دیتا ها را برمیگرداند در این بخش میخواهیم متدی اضافه کنیم که با دادن ID، Product مشخصات رکورد مورد نظر را به ما دهد خوب پس شروع کنیم.

قدم اول :

روی فایل Northwind.xsd در فولدر App_Code کلیک کنید همانطور که در شکل میبینید شمای جدول Product را میبینید در پایین جدول در قسمت ProductTableAdapter نام متدی که قبلا درست کردیم را میبینید روی نام متد قبلی راست کلیک کنید و از منوی باز شده Add Query را انتخاب




کنید

قدم دوم :

در اینجا از ما پرسیده میشود که چگونه میخواهیم دسترسی به DataBase داشته باشید که می گزینه اول یعنیSql– Ad hoc را انتخاب میکنیم .

قدم سوم :

صفحه نمایش داده شده را هم در قسمت قبل دیده اید. در اینجا برای سرعت کار من مستقیما SQL را می نویسم . البته این همان دستور قبلی است با این تفاوت که این شرط Where اضافه شده : CategoryID = @CategoryID
که مقدار @CategoryID در زمان اجرا مشخص میشود. . و پارامتر ورودی تابع ما است Next

قدم چهارم:

نام تابع را GetProductByProductID(*productID*) و Finish را فشار دهید.خود VS نام پارامتر ورودی را میگذارد و شما نیازی به دادن پارامتر ورودی ندارید.

قدم پنجم:

در فایل Northwind.xsd روی تابع بالا راست کلیک کرده و گزینه .Preview Data را انتخاب کنید و در بخش پارامتر مقداری را وارد کنید و روی Preview کلیک کنید تا دیتاها نمایش داده شود

# طرز استفاده

کدهای زیر را در HTML

```
</head>
<body>
<form id="form1" runat="server">
<div>
<h1>Beverages</h1>
<p>
<asp:GridView ID="GridView1" runat="server"
CssClass="DataWebControlStyle">
<HeaderStyle CssClass="HeaderStyle" />
<AlternatingRowStyle CssClass="AlternatingRowStyle" />
</asp:GridView>
 </p>
</div>
</form>
</body>
```

کدهای زیر را در Code Behind:

```
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using NorthwindTableAdapters;
```

```
public partial class Beverages : System.Web.UI.Page
{
protected void Page_Load(object sender, EventArgs e)
{
ProductsTableAdapter productsAdapter = new
ProductsTableAdapter();
GridView1.DataSource =
productsAdapter.GetProductsByCategoryID(1);
GridView1.DataBind();
}
}
```

# اضافه،تغییر و حذف دیتاها

به طور کلی  برای حذف ، تغییر و اضافه کردن دیتا به دیتا بیس دو روش داریم.در روش اول ما صدا میزنیم متدهای را که مستقیما روی دیتا بیس اثر گذاشته و با اجرا شدن این متدها دستورات Insetr,Update,Delete اس کیو ال هم اجرا شده و یک سطر از دیتا بیس ما تغییر میکند . این گونه متدها یک توالی از داده های یکتا دارند  (متل  , integer, string DataTime) که برای انجام این اعمال لازم است.به طور مثال برای این روش برای جدول products متد Delete مقدار productsID از نوع صحیح را میگیرد و سطر متناظر این داده را پاک میکند و همچنین متد Insert یک مقدار ProductsName از نوع String و مقدار unitPrice از نوع Decimal و .... میگیرد و عمل اضافه کردن را انجام میدهد ( شکل 1)

خودتان تصور کنید بنا به دلایلی باید 10 سطر ار جدول را حذف کنید خودتان تصور کنید که این کار چقدر کسل آور است .روش دوم اشکالات این روش را  از یبن برده است.

در روش دوم یکbatch update  را بکار میبریم .این متد در پارامتر ورودی خودش یک dataset یا datatable یا مجموعه ای از DataRow ها را میگیرد.جالب اینجاست که خود این متد تصمیم میگیرد که کدام یک از دستورات SQL یعنیINSERT, UPDATE  و یا DELETE را اجرا کند.(شکل2)

یک TableAdapter به طور پیش فرض دارای یک Batch Update است .هنگاهی که ما گزینه  Generate Insert , Update and Delete statement در Advance Property انتخاب کردیم (در یخش دوم) ProductsAdapter حاوی متدی میشود به نام ()Update که  batch update را پیاده سازی میکند. اگر شما چک باکس GenerateDBDirectsMethod را فعال بگذارید TableAdapter شما حاوی متهای مستقیمInsert  , Update و Delete است.

هر دوی این روش ها  استفاده میکنند ازخاصیت(property) InsertCommand و DeleteCommand وUpdateCommand ، TableAdapter برای انجام اعمال اضافه کردن و حذف و تغییر در دیتا بیس.
شما میتوانید این خاصیت ها را بررسی کنید با راست کلیک روی DataSet خود در DataSet Designer و روی property کلیک کنید در پنجره می توانید این خاصیت ها را مشاهده کنید.و میتوانید حتی آنها را تغییر دهید .

# طرز استفاده

مثال زیر را بررسی میکنیم

```
NorthwindTableAdapters.ProductsTableAdapter productsAdapter =
new NorthwindTableAdapters.ProductsTableAdapter();


Northwind.ProductsDataTable products = productsAdapter.GetProducts();
foreach (Northwind.ProductsRow product in products)
        if (!product.Discontinued && product.UnitsInStock <= 25)
product.UnitPrice *= 2;
// Update the products
productsAdapter.Update(products);
```

**اینجای کد از متدهای مستقسم استفاده میکند**

```
NorthwindTableAdapters.ProductsTableAdapter productsAdapter =
new NorthwindTableAdapters.ProductsTableAdapter();
// Delete the product with ProductID 3
productsAdapter.Delete(3);
// Update Chai (ProductID of 1), setting the UnitsOnOrder to 15
productsAdapter.Update("Chai", 1, 1, "10 boxes x 20 bags",
18.0m, 39, 15, 10, false, 1);
// Add a new product
productsAdapter.Insert("New Product", 1, 1,
"12 tins per carton", 14.95m, 15, 0, 10, false);
```

## اضافه کردن متدهایInsert,Update,Delete

در این قسمت ما برای DAL خود متدهای Update،Insert و Delete را خودمان می سازیم البته در این بخش ما فقط متد Insert را ساخته و متدهای بعدی به عهده خودتان باشد . تا به حال برایتان اتفاق افتاده که وقتی یک ایتم جدیدی به دیتا بیس خود اضافه می کنید به خواهید شماره آی دی این آیتم جدید را درست بعد از اضافه کردن داشته باشید .در این بخش ما متد Insert را طوری تغییر می دهیم که نیاز ما را برآورده سازد .خوب حال شروع میکنیم .

روی Northwind.xsd کلیک کرده و در قسمت متدها راست کلیک کنید و Add Query را انتخاب کنید .

در قسمت Choose a Query type قسمت insert را انتخاب کرده و کلید Nextرا فشار دهید

همانطور که میبینید VS دستورات لازم برای اضافه کردن یک آیتم را نوشته آخر دستور یک سمی کالن(;) گذاشته و دستور زیر را اضافه میکنیم

```
Select SCOP_IDENTITY
```

کلیدNext را فشار می دهیم .

خوب برای نام هم از نام InsetProduct استفاده کرده و Finish می زنیم.

حال روی متد در DataSet Designer راست کلیک کرده و از منوی باز شده Propertyرا انتخاب کرده و در قسمت ExecuteMode گزینه Scalar را انتخاب کنید . می دانید که Insert یک Query است که هیچ داده را بر نمی گرداند با این کار ما تعریف کردیم با هر فرا خوانی این متد یک مقدار برگردانده می شود که آنهم ای دی آیتم جدید است.

**طرز استفاده :**

```
  ProductsTableAdapter adap = new ProductsTableAdapter();
  int ID = Convert.ToInt32(adap.Insert("New Product", 1, 1, "12 tins
per carton", 14.95m, 10, 0, 10, false));
```

**این بخش را انجام دهید چون در بخش بعدی باهاش کار داریم**

با توجه به روشهای قبلی و کدهای زیر DAL خود را کامل کنید :

- **ProductsTableAdapter**
  - **GetProducts**:

```
SELECT ProductID, ProductName, SupplierID, CategoryID,
QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder,
ReorderLevel, Discontinued , (SELECT CategoryName FROM
Categories WHERE Categories.CategoryID =
Products.ProductID) as CategoryName, (SELECT CompanyName
FROM Suppliers WHERE Suppliers.SupplierID =
Products.SupplierID) as SupplierName
FROM Products
```

  - **GetProductsByCategoryID**:

```
SELECT ProductID, ProductName, SupplierID, CategoryID,
QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder,
ReorderLevel, Discontinued , (SELECT CategoryName FROM
Categories WHERE Categories.CategoryID =
Products.ProductID) as CategoryName,
(SELECT CompanyName FROM Suppliers WHERE
Suppliers.SupplierID = Products.SupplierID) as SupplierName
FROM Products
WHERE CategoryID = @CategoryID
```

- **GetProductsBySupplierID**

```
SELECT ProductID, ProductName, SupplierID, CategoryID,
QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder,
ReorderLevel, Discontinued ,
(SELECT CategoryName FROM Categories WHERE
Categories.CategoryID = Products.ProductID)
as CategoryName, (SELECT CompanyName FROM Suppliers
WHERE Suppliers.SupplierID = Products.SupplierID)
as SupplierName
FROM Products
WHERE SupplierID = @SupplierID
```

- **GetProductByProductID**

```
SELECT ProductID, ProductName, SupplierID, CategoryID,
QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder,
ReorderLevel, Discontinued , (SELECT CategoryName
FROM Categories WHERE Categories.CategoryID =
Products.ProductID) as CategoryName,
(SELECT CompanyName FROM Suppliers
WHERE Suppliers.SupplierID = Products.SupplierID)
as SupplierName
FROM Products
WHERE ProductID = @ProductID
```

- **CategoriesTableAdapter**
  - **GetCategories**

```
SELECT CategoryID, CategoryName, Description
FROM Categories
```

  - **GetCategoryByCategoryID**

```
SELECT CategoryID, CategoryName, Description
FROM Categories
WHERE CategoryID = @CategoryID
```

- **SuppliersTableAdapter**
  - **GetSuppliers**

```
SELECT SupplierID, CompanyName, Address, City,
Country, Phone
```

FROM Suppliers

- **GetSuppliersByCountry**

```
SELECT SupplierID, CompanyName, Address,
City, Country, Phone
FROM Suppliers
WHERE Country = @Country
```

- **GetSupplierBySupplierID**

```
SELECT SupplierID, CompanyName, Address,
City, Country, Phone
FROM Suppliers
WHERE SupplierID = @SupplierID
```

- **EmployeesTableAdapter**
  - **GetEmployees**

```
SELECT EmployeeID, LastName, FirstName,
Title, HireDate, ReportsTo, Country
FROM Employees
```

  - **GetEmployeesByManager**

```
SELECT EmployeeID, LastName, FirstName,
Title, HireDate, ReportsTo, Country
FROM Employees
WHERE ReportsTo = @ManagerID
```

  - **GetEmployeeByEmployeeID**

```
SELECT EmployeeID, LastName, FirstName,
Title, HireDate, ReportsTo, Country
FROM Employees
WHERE EmployeeID = @EmployeeID
```

# اضافه کردن کدهای دلخواه به DAL

TableAdapter و DataTable هایی که به Typed-DataSet اضافه میشوند دارای شمای XML هستند که شما با راست کلیک در Solution Explorer روی Northwind.xsd وانتخاب Viewcode میتوانید اکدهای XML را ببینید.

این Schema در زمان کامپایل یا اجرا به کدهای C# یا VB تبدیل می شود. برای دیدن این کدها در Class View (ctrl+shift+c) روی مثبت NorthwindTableAdapter کلیک کنید در

اینجا تمام TableAdapter هایی که تا به حال ساختید را میبینید . روی ProdouctsTableAdapter کلیک کرده و میبینید در پنجره پایین آن تمام متدها و Property ها ی این کلاس نمایش داده می شود و شما روی GetAllProduct دوبار کلیک کرده تا کدهای متناظرش را ببینید.

همانطود که گفتم این کدها را خود VS اضافه میکند حال اگر بخواهیم ما به این کدها چیزی اضافه کنیم این خطر وجود دارد که در زمان کامپایل یا اجرا این کدها دوباره تولید شده و کدهای اضافه شده ما حذف شود . **تکلیف چیست ؟**

برای رفع این مشکل ما از خاصیت جدیدی به نام Partial Class استفاده میکنیم.
**Partial Class**: این امکان را میدهد که ما کلاسهای همنام را در فایلهای مختلف یک پروژه توسعه دهیم.

برای این کار کلاس جدیدی در فولدرApp_Code درست کنید و نام آن را SuppliersRow.cs و کدهای زیر را اضافه کنید.

```
using System;
using System.Data;
using NorthwindTableAdapters;
public partial class Northwind
{
        public partial class SuppliersRow
        {
                public Northwind.ProductsDataTable GetProducts()
                {
                        ProductsTableAdapter productsAdapter =
                        new ProductsTableAdapter();
                        return
                productsAdapter.GetProductsBySupplierID(this.SupplierID);
                }
        }
}
```

**SuppliersRow** نشان دهنده یک سطر از جدول Suppliers است .هر فروشنده میتواند صفر یا بیشتر محصول را تهیه کند. متد **()GetProducts** نشان دهنده محصولاتی که یک فروشنده مخصوص تهیه کرده است.

این **Partial Class در زمان اجرا یا کامپایل به Norhwind.SuppliersRow فضای نامی NorthwindTableAdapter اضافه میشود .(در Class view نگاه کنید)**

## طرز استفاده :

### SuppliersAndProducts.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="SuppliersAndProducts.aspx.cs"
Inherits="SuppliersAndProducts" %>
<!DOCTYPE html PUBLIC "//
W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1transitional.
dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
<title>Untitled Page</title>
<link href="Styles.css" rel="stylesheet" type="text/css" />
</head>
<body>
<form id="form1" runat="server">
<div>
<h1> Suppliers and Their Products</h1>
<p>
<asp:GridView ID="GridView1" runat="server"
AutoGenerateColumns="False"
CssClass="DataWebControlStyle">
<HeaderStyle CssClass="HeaderStyle" />
<AlternatingRowStyle CssClass="AlternatingRowStyle" />
<Columns>
<asp:BoundField DataField="CompanyName"
HeaderText="Supplier" />
<asp:TemplateField HeaderText="Products">
<ItemTemplate>
<asp:BulletedList ID="BulletedList1"
runat="server" DataSource='<%#
((Northwind.SuppliersRow)((System.Data.DataRowView)
Container.DataItem).Row).GetProducts() %>'
DataTextField="ProductName">
</asp:BulletedList>
</ItemTemplate>
</asp:TemplateField>
</Columns>
</asp:GridView>
 </p>
```

```
</div>
</form>
</body>
</html>
```

## SuppliersAndProducts.aspx.cs

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using NorthwindTableAdapters;
public partial class SuppliersAndProducts : System.Web.UI.Page
{
protected void Page_Load(object sender, EventArgs e)
{
SuppliersTableAdapter suppliersAdapter = new
SuppliersTableAdapter();
GridView1.DataSource = suppliersAdapter.GetSuppliers();
GridView1.DataBind();
}
}
```