Contents

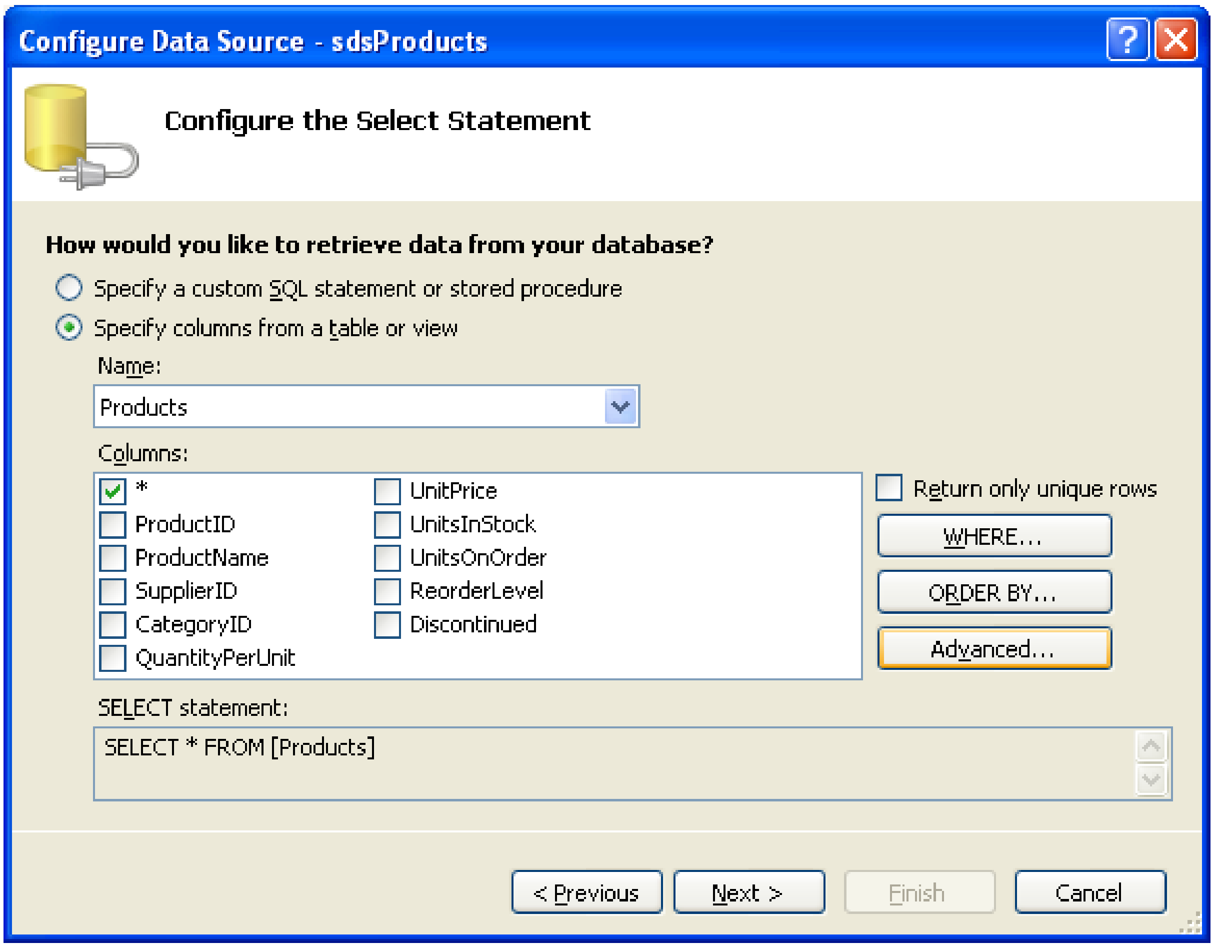
Edit

The gridview control allows editing of data. This is achieved with very little work.

Activity 1. Editing the product table

editProducts.aspx

Create a new web page, add a gridview to the web page. This grid view will link to a datasource which selects all the columns from the products table.


Configure Data Source - sdsProducts
Configure the Select Statement
How would you like to retrieve data from your database?
Specify a custom SQL statement or stored procedure
Specify columns from a table or view
Name:
Columns:
Return only unique rows
WHERE...
ORDER BY...
Advanced...
SELECT statement:
< Previous
Next >
Finish
Cancel

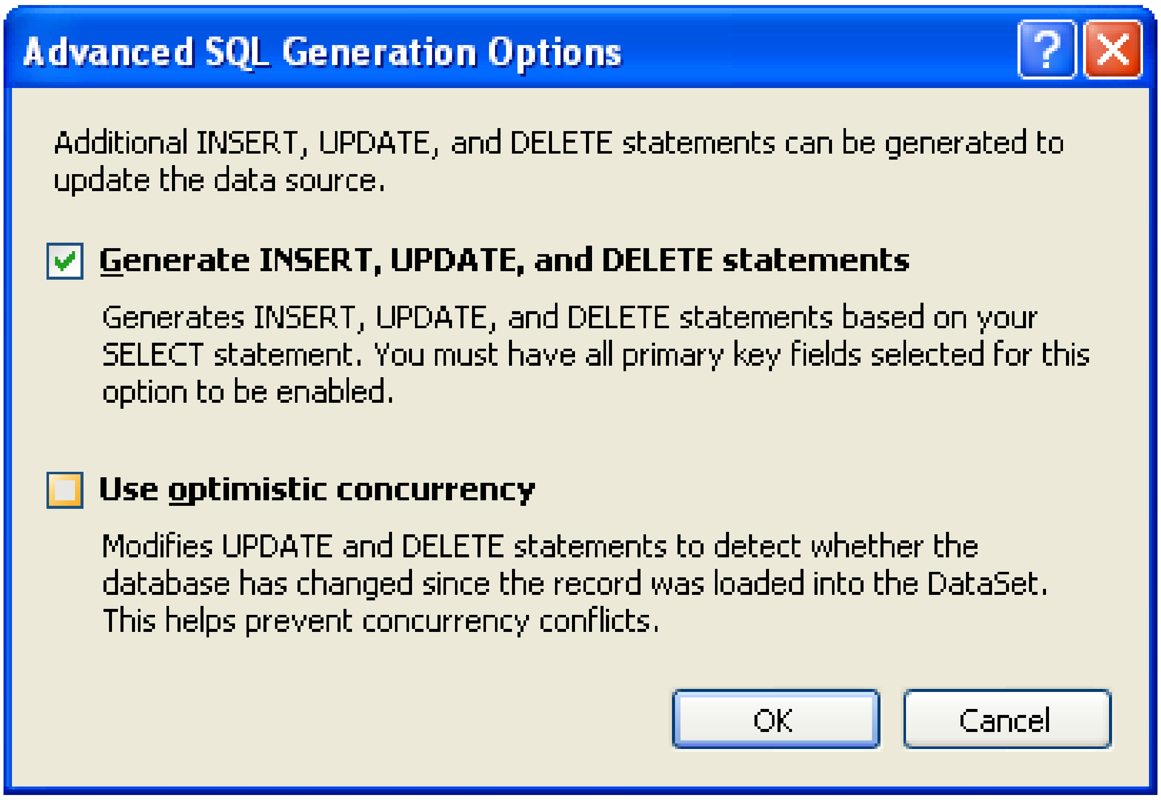
Name: Products

Columns: [✓ *] ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued

SELECT statement:

```
SELECT * FROM [Products]
```

Click on the Advanced button. This will open the following screen:


Advanced SQL Generation Options

Additional INSERT, UPDATE, and DELETE statements can be generated to update the data source.

[✓] Generate INSERT, UPDATE, and DELETE statements

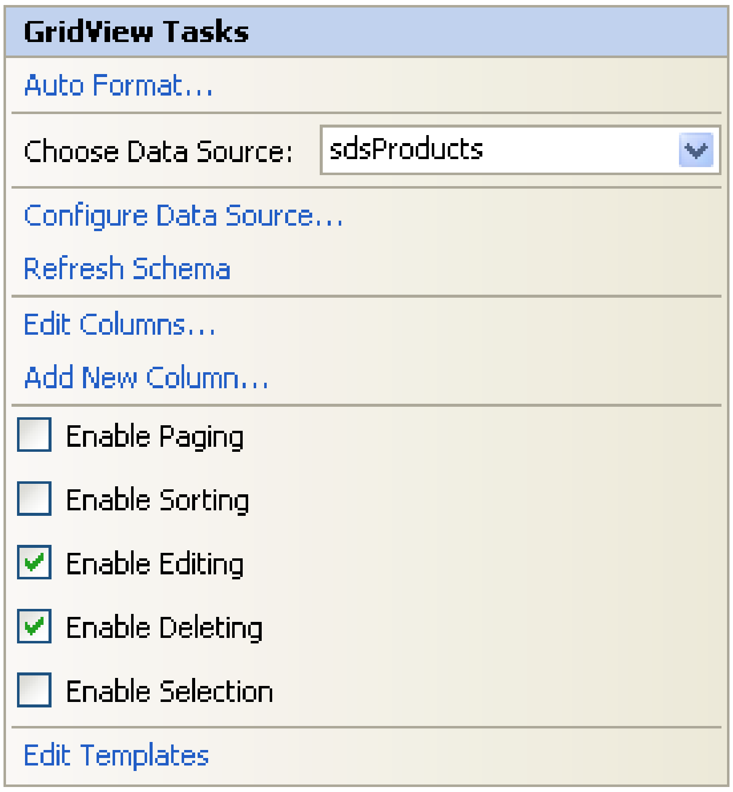
Generates INSERT, UPDATE, and DELETE statements based on your SELECT statement. You must have all primary key fields selected for this option to be enabled.

[ ] Use optimistic concurrency

Modifies UPDATE and DELETE statements to detect whether the database has changed since the record was loaded into the DataSet. This helps prevent concurrency conflicts.

Place a tick in the Generate INSERT, UPDATE and DELETE statements and click OK. This will generate the correct SQL Insert, update and delete statements for the product table.

Now when you click the gridview's smart tag you will notice a few new checkboxes:


GridView Tasks

Choose Data Source: sdsProducts

[ ] Enable Paging
[ ] Enable Sorting
[✓] Enable Editing
[✓] Enable Deleting
[ ] Enable Selection

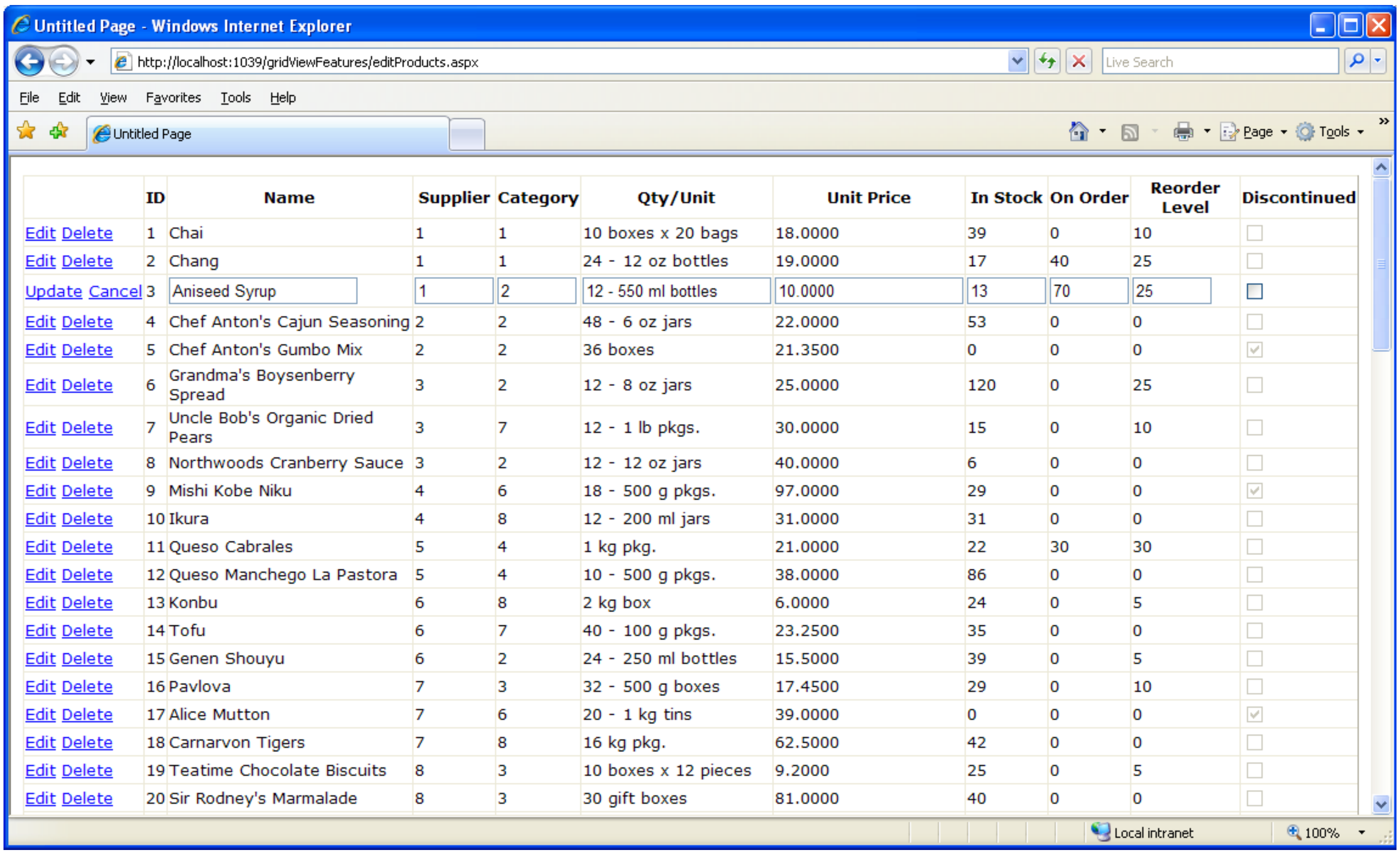
Place a tick in enable editing and enable deleting.

View the page in the browser. Notice I renamed the column headings. If you click on the edit link the grid view changes the row and replaces the text in column with a textbox. For each row except for the discontinued column which becomes an enabled checkbox and the product id which is the primary key. The primary key of a table should never be updatable as this could potentially break the referential integrity of the database.

This allows the user to make a change and then click on the update link to update the change in the database.

Change the unit price to 15.00 and click on update.

NOTE: Before you delete any rows from this gridview add a few to the products table that way you won't be breaking the referential integrity.


Untitled Page - Windows Internet Explorer
http://localhost:1039/gridViewFeatures/editProducts.aspx
Live Search
File Edit View Favorites Tools Help
Untitled Page
Page Tools
Local intranet
100%

	ID	Name	Supplier	Category	Qty/Unit	Unit Price	In Stock	On Order	Reorder Level	Discontinued
Edit Delete	1	Chai	1	1	10 boxes x 20 bags	18.0000	39	0	10	☐
Edit Delete	2	Chang	1	1	24 - 12 oz bottles	19.0000	17	40	25	☐
Update Cancel	3	Aniseed Syrup	1	2	12 - 550 ml bottles	10.0000	13	70	25	☐
Edit Delete	4	Chef Anton's Cajun Seasoning	2	2	48 - 6 oz jars	22.0000	53	0	0	☐
Edit Delete	5	Chef Anton's Gumbo Mix	2	2	36 boxes	21.3500	0	0	0	✓
Edit Delete	6	Grandma's Boysenberry Spread	3	2	12 - 8 oz jars	25.0000	120	0	25	☐
Edit Delete	7	Uncle Bob's Organic Dried Pears	3	7	12 - 1 lb pkgs.	30.0000	15	0	10	☐
Edit Delete	8	Northwoods Cranberry Sauce	3	2	12 - 12 oz jars	40.0000	6	0	0	☐
Edit Delete	9	Mishi Kobe Niku	4	6	18 - 500 g pkgs.	97.0000	29	0	0	✓
Edit Delete	10	Ikura	4	8	12 - 200 ml jars	31.0000	31	0	0	☐
Edit Delete	11	Queso Cabrales	5	4	1 kg pkg.	21.0000	22	30	30	☐
Edit Delete	12	Queso Manchego La Pastora	5	4	10 - 500 g pkgs.	38.0000	86	0	0	☐
Edit Delete	13	Konbu	6	8	2 kg box	6.0000	24	0	5	☐
Edit Delete	14	Tofu	6	7	40 - 100 g pkgs.	23.2500	35	0	0	☐
Edit Delete	15	Genen Shouyu	6	2	24 - 250 ml bottles	15.5000	39	0	5	☐
Edit Delete	16	Pavlova	7	3	32 - 500 g boxes	17.4500	29	0	10	☐
Edit Delete	17	Alice Mutton	7	6	20 - 1 kg tins	39.0000	0	0	0	✓
Edit Delete	18	Carnarvon Tigers	7	8	16 kg pkg.	62.5000	42	0	0	☐
Edit Delete	19	Teatime Chocolate Biscuits	8	3	10 boxes x 12 pieces	9.2000	25	0	5	☐
Edit Delete	20	Sir Rodney's Marmalade	8	3	30 gift boxes	81.0000	40	0	0	☐

To verify the change updated check your database.

The sqlDataSource and gridview controls generated the following code for you:

```
<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False"
DataKeyNames="ProductID"
DataSourceID="sdsProducts">

<Columns>
<asp:CommandField ShowDeleteButton="True" ShowEditButton="True" />

<asp:BoundField DataField="ProductID" HeaderText="ID"
InsertVisible="False" ReadOnly="True" SortExpression="ProductID" />

<asp:BoundField DataField="ProductName" HeaderText="Name"
SortExpression="ProductName" />

<asp:BoundField DataField="SupplierID"
HeaderText="Supplier" SortExpression="SupplierID" />

<asp:BoundField DataField="CategoryID"
HeaderText="Category" SortExpression="CategoryID" />

<asp:BoundField DataField="QuantityPerUnit"
HeaderText="Qty/Unit" SortExpression="QuantityPerUnit" />

<asp:BoundField DataField="UnitPrice" HeaderText="Unit
Price" SortExpression="UnitPrice" />

<asp:BoundField DataField="UnitsInStock" HeaderText="In
Stock" SortExpression="UnitsInStock" />
```

```
<asp:BoundField DataField="UnitsOnOrder" HeaderText="On
Order" SortExpression="UnitsOnOrder" />

<asp:BoundField DataField="ReorderLevel"
HeaderText="Reorder Level" SortExpression="ReorderLevel" />

<asp:CheckBoxField DataField="Discontinued"
HeaderText="Discontinued" SortExpression="Discontinued" />
</Columns>
</asp:GridView>

<asp:SqlDataSource ID="sdsProducts" runat="server" ConnectionString="<%$
ConnectionStrings:ConnectionString %>"
DeleteCommand="DELETE FROM [Products] WHERE [ProductID] = @ProductID"
InsertCommand="INSERT INTO [Products] ([ProductName], [SupplierID], [CategoryID],
[QuantityPerUnit], [UnitPrice], [UnitsInStock], [UnitsOnOrder], [ReorderLevel],
[Discontinued]) VALUES (@ProductName, @SupplierID, @CategoryID, @QuantityPerUnit,
@UnitPrice, @UnitsInStock, @UnitsOnOrder, @ReorderLevel, @Discontinued)"
SelectCommand="SELECT * FROM [Products]"
UpdateCommand="UPDATE [Products] SET [ProductName] = @ProductName, [SupplierID] =
@SupplierID, [CategoryID] = @CategoryID, [QuantityPerUnit] = @QuantityPerUnit,
[UnitPrice] = @UnitPrice, [UnitsInStock] = @UnitsInStock, [UnitsOnOrder] =
@UnitsOnOrder, [ReorderLevel] = @ReorderLevel, [Discontinued] = @Discontinued WHERE
[ProductID] = @ProductID">
<DeleteParameters>
<asp:Parameter Name="ProductID" Type="Int32" />
</DeleteParameters>
<UpdateParameters>
<asp:Parameter Name="ProductName" Type="String" />
<asp:Parameter Name="SupplierID" Type="Int32" />
<asp:Parameter Name="CategoryID" Type="Int32" />
<asp:Parameter Name="QuantityPerUnit" Type="String" />
<asp:Parameter Name="UnitPrice" Type="Decimal" />
<asp:Parameter Name="UnitsInStock" Type="Int16" />
<asp:Parameter Name="UnitsOnOrder" Type="Int16" />
<asp:Parameter Name="ReorderLevel" Type="Int16" />
<asp:Parameter Name="Discontinued" Type="Boolean" />
<asp:Parameter Name="ProductID" Type="Int32" />
</UpdateParameters>
<InsertParameters>
<asp:Parameter Name="ProductName" Type="String" />
<asp:Parameter Name="SupplierID" Type="Int32" />
<asp:Parameter Name="CategoryID" Type="Int32" />
<asp:Parameter Name="QuantityPerUnit" Type="String" />
<asp:Parameter Name="UnitPrice" Type="Decimal" />
<asp:Parameter Name="UnitsInStock" Type="Int16" />
            <asp:Parameter Name="UnitsOnOrder" Type="Int16" />
<asp:Parameter Name="ReorderLevel" Type="Int16" />
<asp:Parameter Name="Discontinued" Type="Boolean" />
</InsertParameters>
</asp:SqlDataSource>
```

Notice the gridview has now got a command field which allows the display of the edit, delete, update and cancel text.

View the page in the browser again scroll down to the last few products and click on the edit link. Notice the page is refreshed and starts at the top again meaning that you don't get to see the row you originally clicked on.

You can set "MaintainScrollPositionOnPostback" to true in the page directive to maintain the scroll position between postbacks. This works in IE6+, firefox1+, and opera 8+.

Activity 2. Scrolling

editProductsScroll.aspx

Add the “MaintainScrollPositionOnPostback” attribute to the page directive and set it to true. Test it out.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="editProductsScroll.aspx.cs"
Inherits="editProductsScroll" MaintainScrollPositionOnPostback="true" %>
```

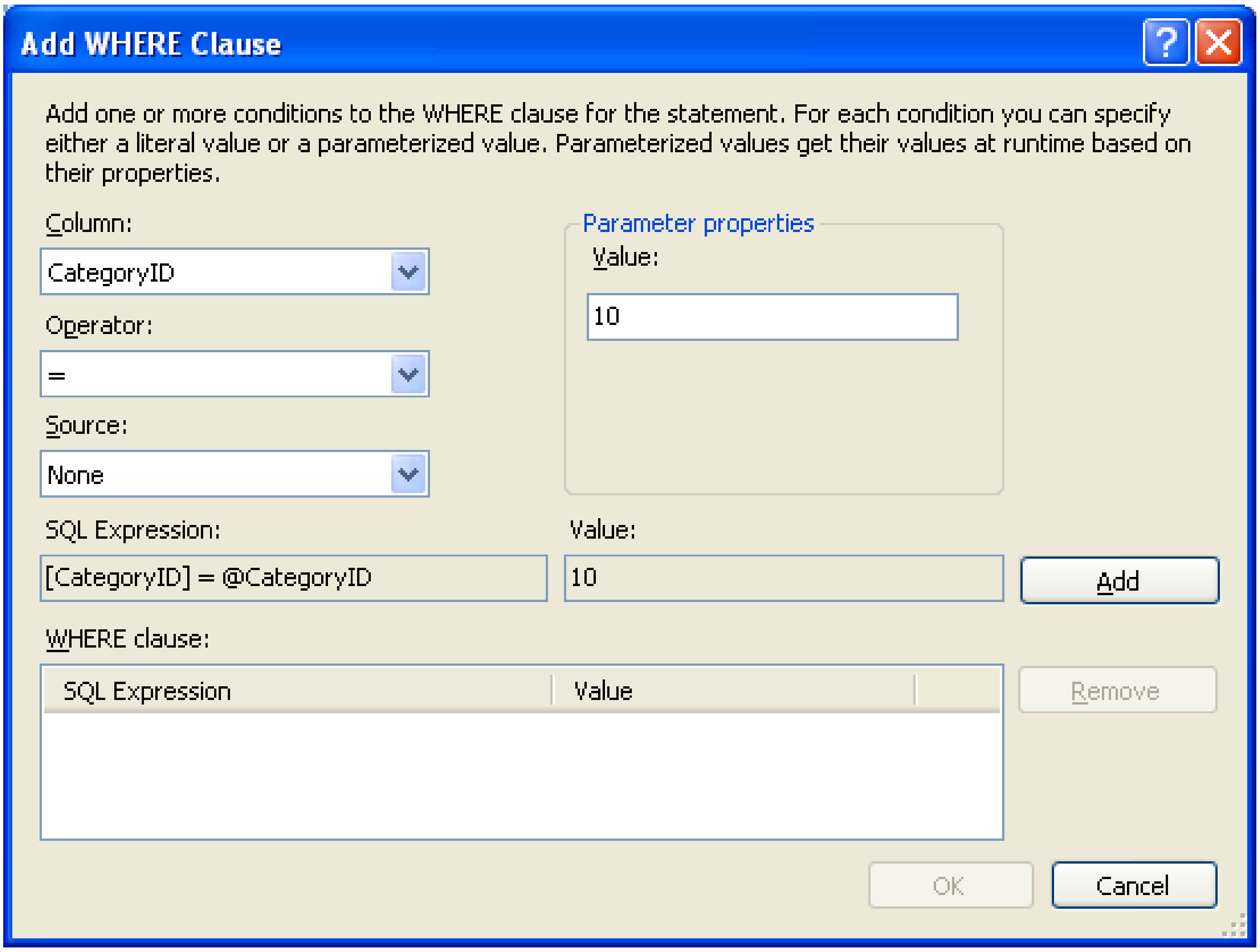
Displaying an empty grid

The gridview includes 2 properties that allow you to display empty data. You can use either the emptyDataText property or the emptyDataTemplate property.

Activity 3. Displaying empty data text

emptyGrid.aspx

Create a web page which displays the product price and unit name for the category 10. You will need to configure the data source to contain the where clause.



Add WHERE Clause

Add one or more conditions to the WHERE clause for the statement. For each condition you can specify either a literal value or a parameterized value. Parameterized values get their values at runtime based on their properties.

Column: CategoryID

Operator: =

Source: None

Parameter properties

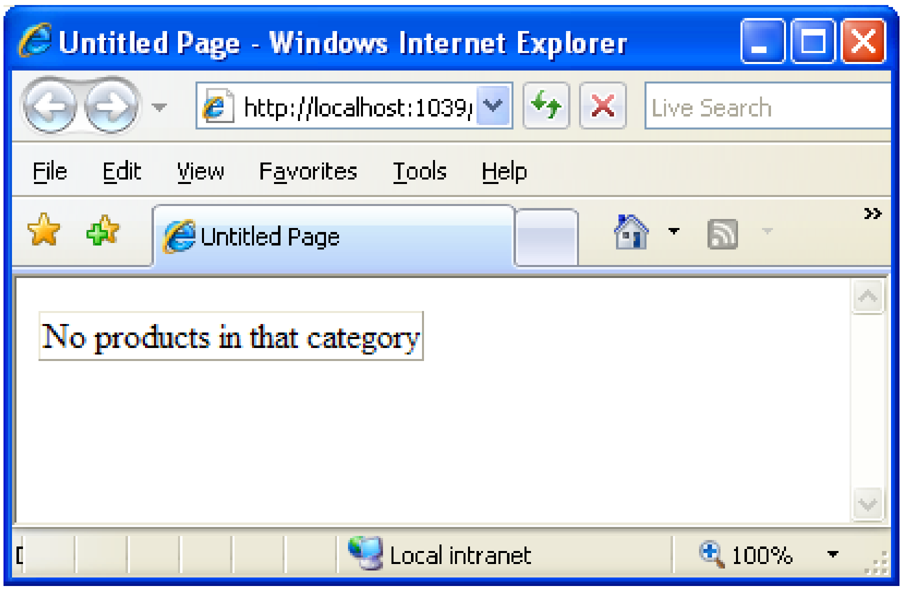
Value: 10

SQL Expression: [CategoryID] = @CategoryID

Value: 10

Modify the grid view so that the text “No products in that category” is displayed. This text will need to be added to the EmptyDataText property of the grid view.

This should be displayed when the page is loaded



No products in that category

Empty data template

You can customize what is displayed if the gridview is empty by using the emptyDataTemplate This allows you to add any HTML or web form control.

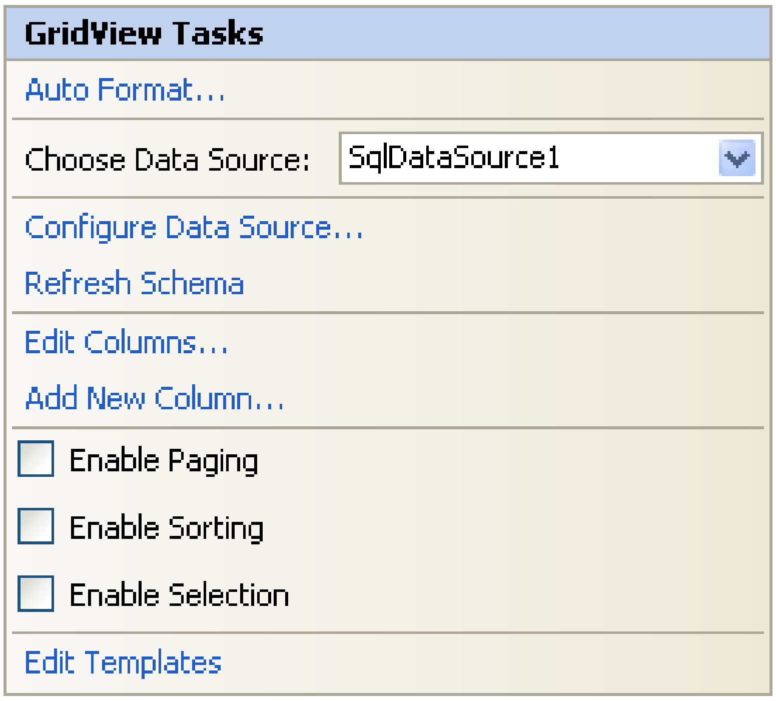
Activity 4. Empty data template

emptyGridTemplate.aspx

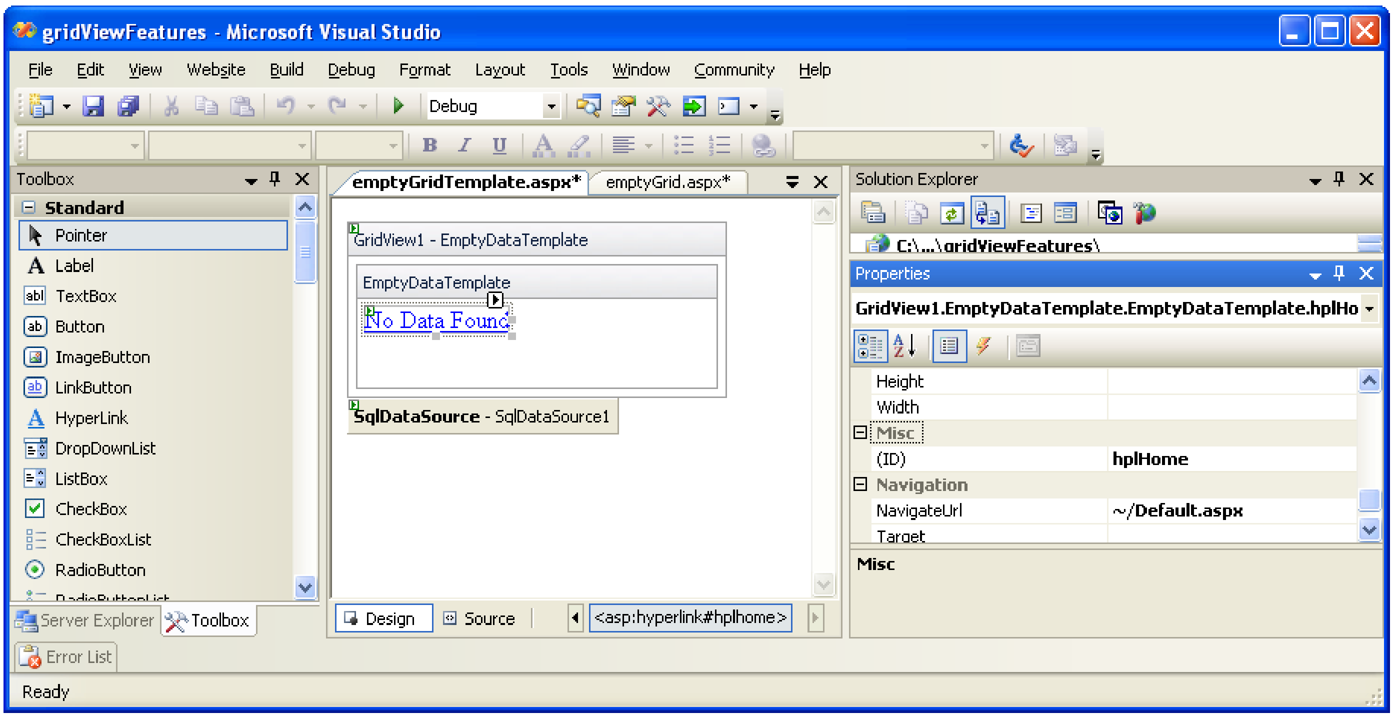
In this activity we will set the empty data template to display a hyperlink back to another page if no data is displayed.

Create a web page which displays the product price and unit name for the category 10. You will need to configure the data source to contain the where clause.

To add an emptyDataTemplate go to the smart tag of the grid view and select the “Edit Templates” link



This will put the grid view in template mode. Make sure the emptyDataTemplate is displayed in the drop down. Add a hyperlink control to the grid view. Make it link to default.aspx.



This is the code that has been generated for you.

```
<asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False"
DataSourceID="SqlDataSource1">
<Columns>
<asp:BoundField DataField="ProductName" HeaderText="ProductName"
SortExpression="ProductName" />
<asp:BoundField DataField="UnitPrice" HeaderText="UnitPrice"
SortExpression="UnitPrice" />
</Columns>
<EmptyDataTemplate>
<asp:HyperLink ID="hplHome" runat="server"
NavigateUrl="~/Default.aspx">No Data Found </asp:HyperLink>
</EmptyDataTemplate>
</asp:GridView>
```

Template column

Empty data template is not the only type of template available. A template field allows you to add any content that you need. You can add HTML or web form controls.

In one of our previous exercises you could edit the products in a grid view and the gridview would display most of the columns as a textbox allowing the user to type in the updated text. This is fine for most columns but not for the supplier ID and the category ID. You cannot expect the user to know the primary keys of the data. A better option would be to display the suppliers and categories as a drop down list and allow the user to select the category or supplier from the list of available options.

Activity 5. Category as a template column.

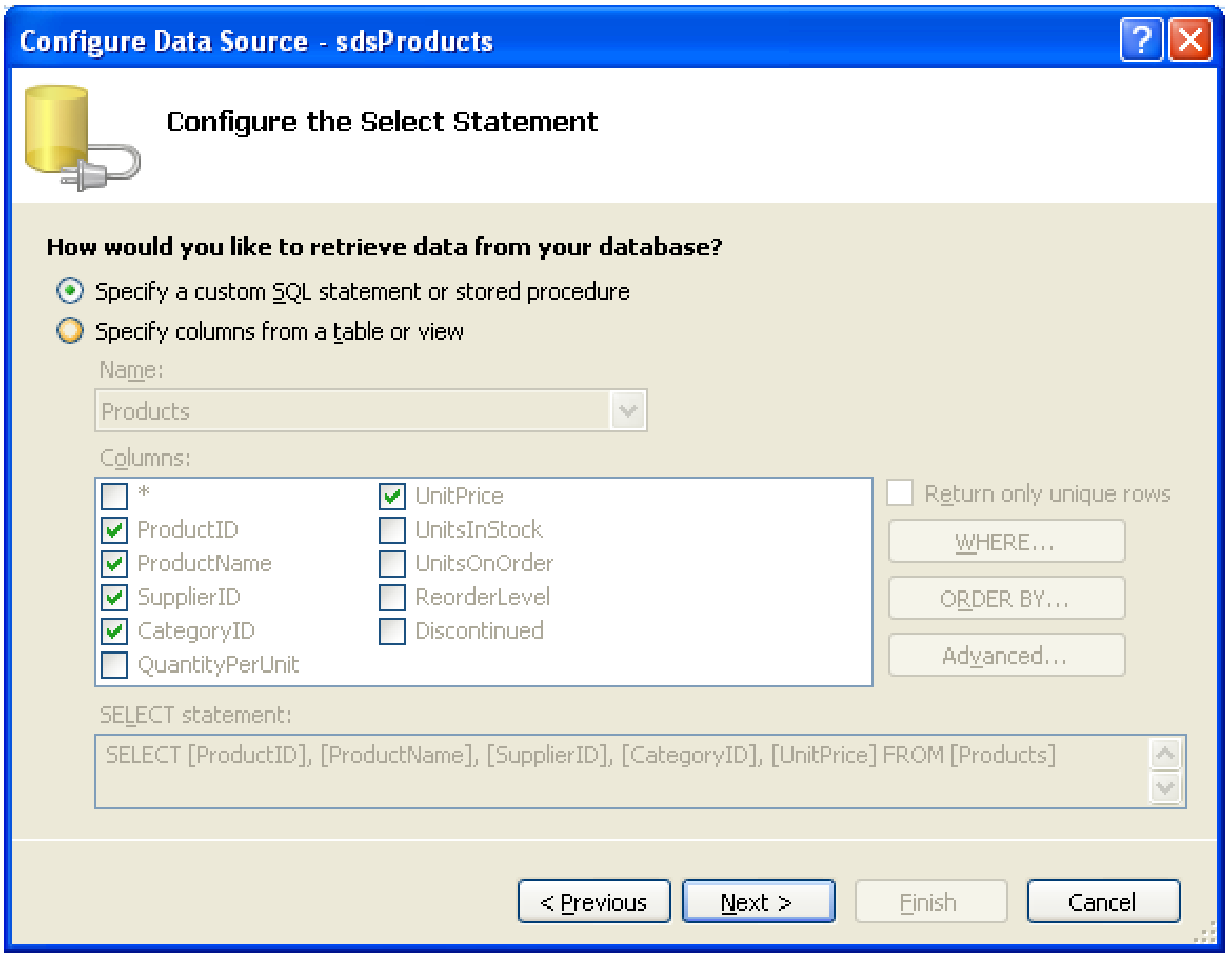
Create a web page which displays the productID, product name, categoryID, supplierID, and unit price in a gridview to the page.

Set up the data source control so that the insert, update and delete statements are generated for you.

Set up the gridview to allow editing of the data. (Similar to Activity 1)

View your page to make sure all looks fine before continuing.

Now modify the SQL statement so that the category name and supplier company name are displayed instead of the ID. You will need to join with the category and supplier tables. To modify the SQL statement go back into the sqlDataSource control and select “Specify a custom SQL statement or stored procedure:


Configure Data Source - sdsProducts

Configure the Select Statement

How would you like to retrieve data from your database?

[✓ Specify a custom SQL statement or stored procedure]
Specify columns from a table or view

Name: Products

Columns:

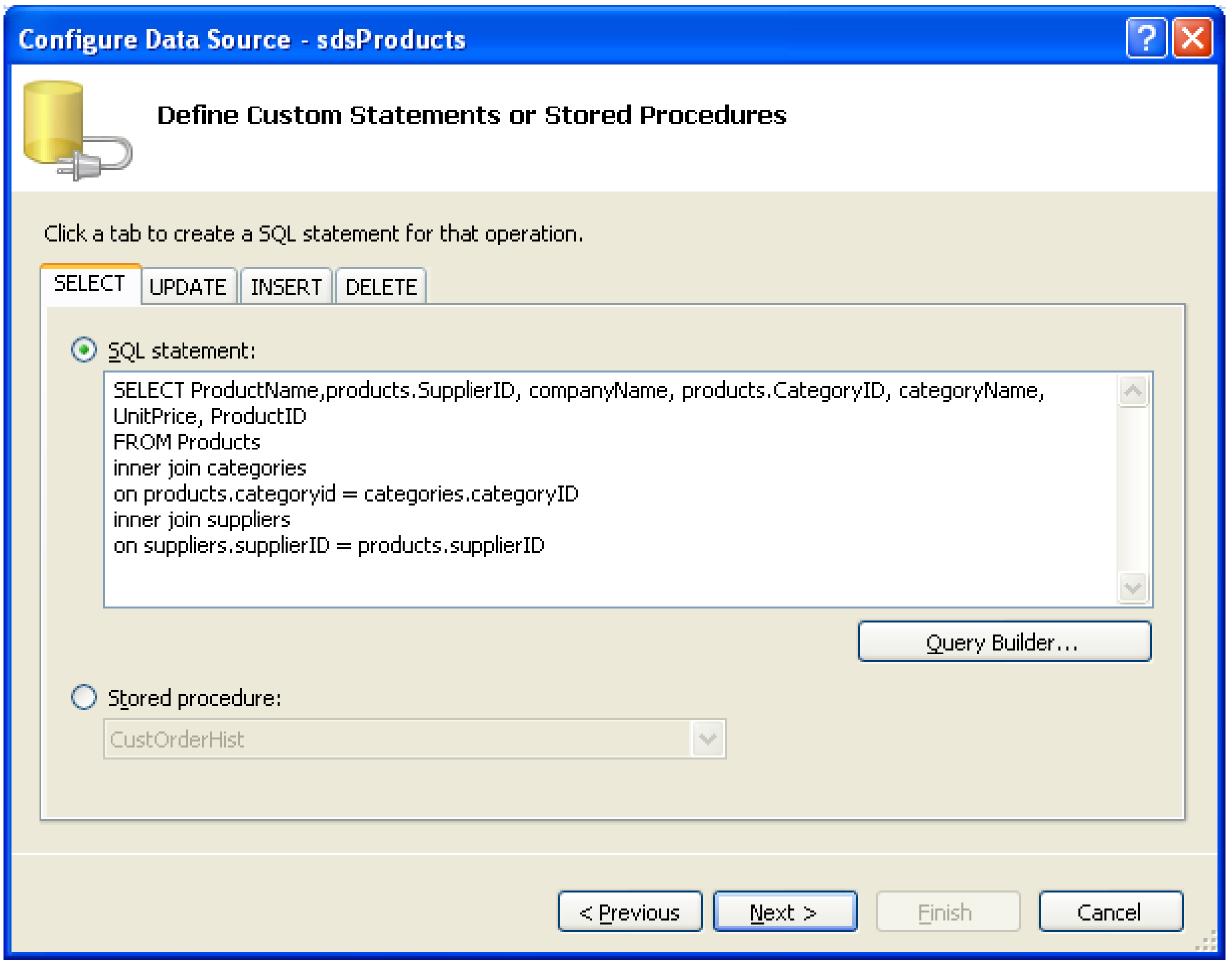
*	✓ UnitPrice
✓ ProductID	UnitsInStock
✓ ProductName	UnitsOnOrder
✓ SupplierID	ReorderLevel
✓ CategoryID	Discontinued
QuantityPerUnit	

Return only unique rows

SELECT statement:

```
SELECT [ProductID], [ProductName], [SupplierID], [CategoryID], [UnitPrice] FROM [Products]
```

The SQL statement should be something like this:


Configure Data Source - sdsProducts

Define Custom Statements or Stored Procedures

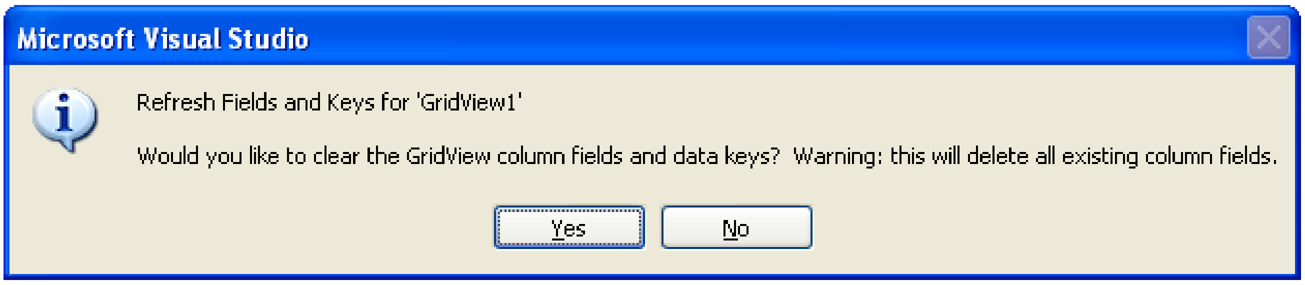
Click a tab to create a SQL statement for that operation.

[✓ SQL statement:]

```
SELECT ProductName,products.SupplierID, companyName, products.CategoryID, categoryName,
UnitPrice, ProductID
FROM Products
inner join categories
on products.categoryid = categories.categoryID
inner join suppliers
on suppliers.supplierID = products.supplierID
```

Stored procedure: CustOrderHist

Click on Next and complete the wizard. The following will be displayed:

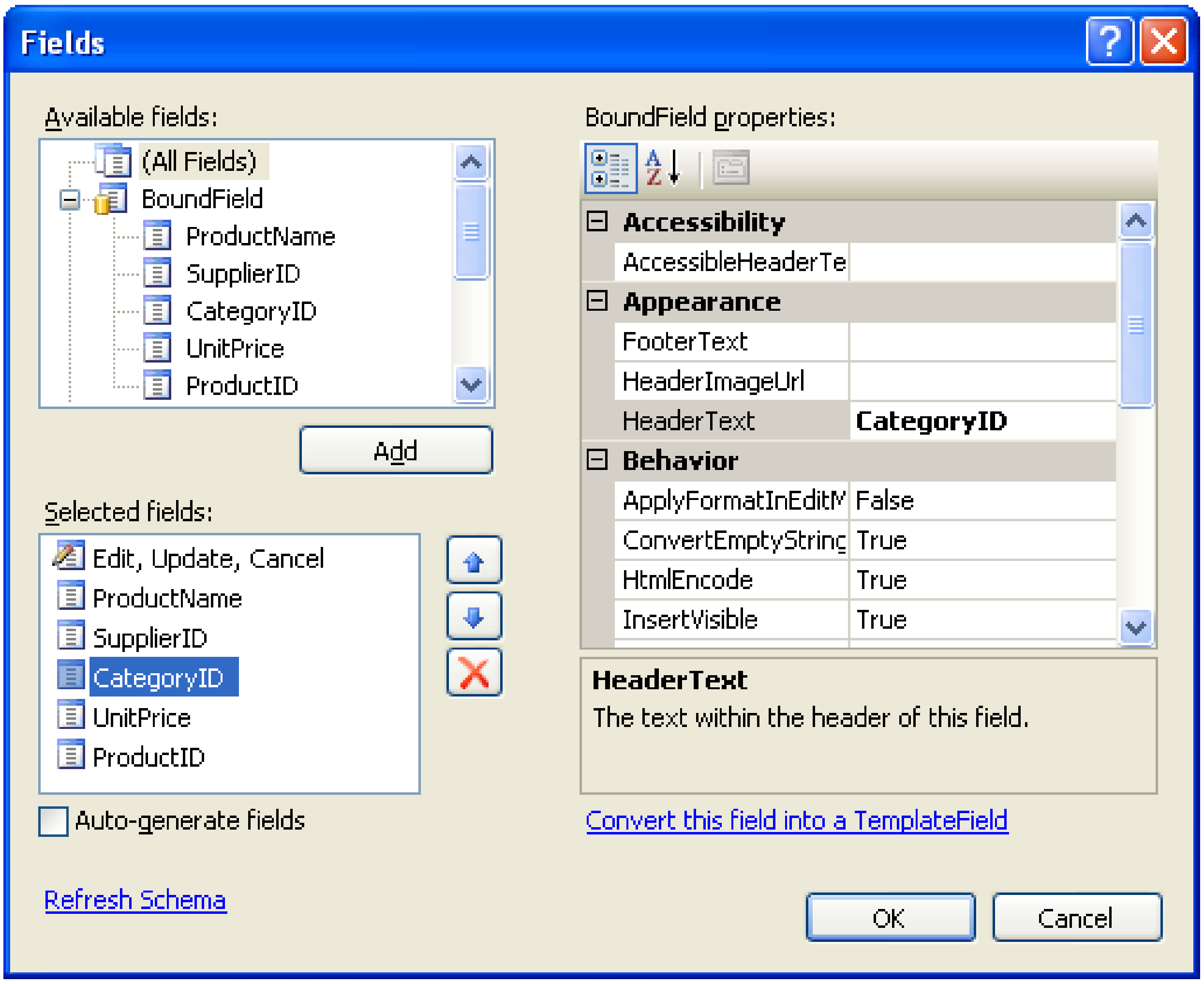

Microsoft Visual Studio

Refresh Fields and Keys for 'GridView1'

Would you like to clear the GridView column fields and data keys? Warning: this will delete all existing column fields.

Make sure you say No to this.

From the grid view smart tag select the edit columns link and make category a template field by clicking on the link "Convert this field into a templateField"


Fields

Available fields:
(All Fields)
BoundField
ProductName
SupplierID
CategoryID
UnitPrice
ProductID

Selected fields:
Edit, Update, Cancel
ProductName
SupplierID
CategoryID
UnitPrice
ProductID

Auto-generate fields

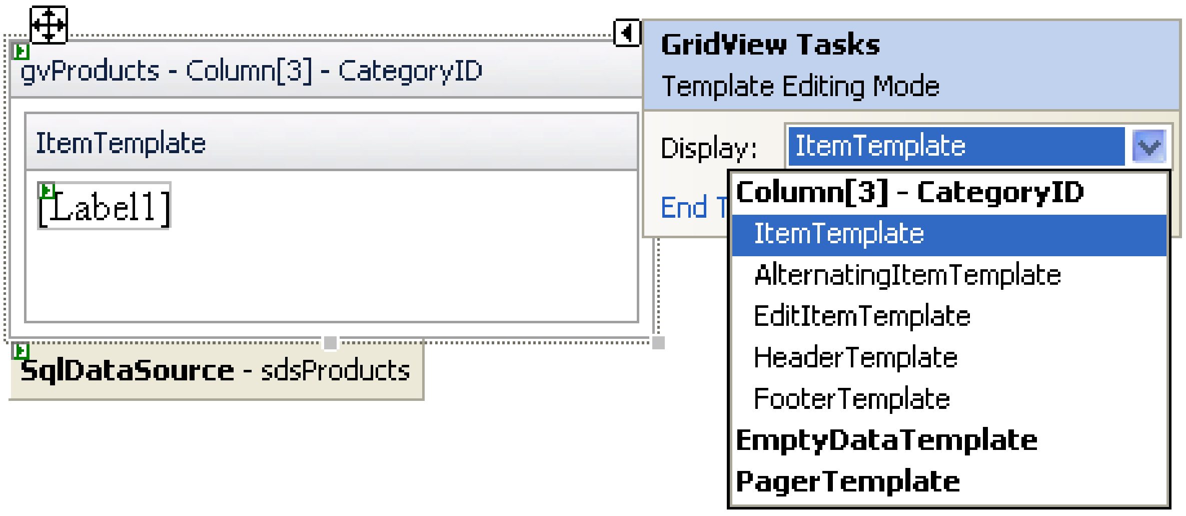
BoundField properties:

Accessibility	
AccessibleHeaderTe	
Appearance	
FooterText	
HeaderImageUrl	
HeaderText	**CategoryID**
Behavior	
ApplyFormatInEditM	False
ConvertEmptyStrinc	True
HtmlEncode	True
InsertVisible	True

HeaderText
The text within the header of this field.

Click on OK.

Now from the grid view smart tag select "edit templates"



A template field has been created for the category ID. A template field contains several templates.

Item template – The contents of this template are displayed for every row rendered by the grid view

Alternating Item template - The contents of this template are displayed for every second row rendered by the grid view

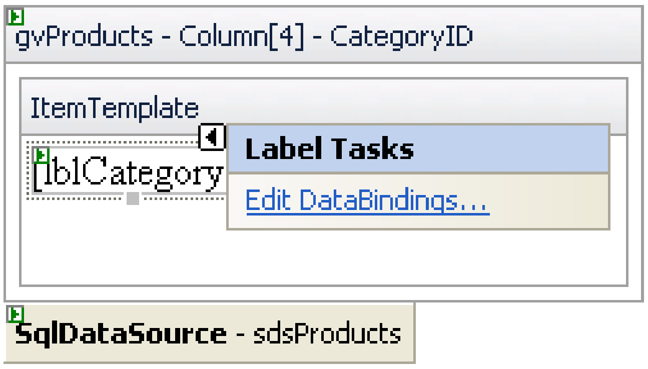
Edit Item template – The contents of this template are displayed when a row is selected for editing

Header template – The contents of this template are displayed for the column heading

Footer template – The contents of this template are displayed for the column footer

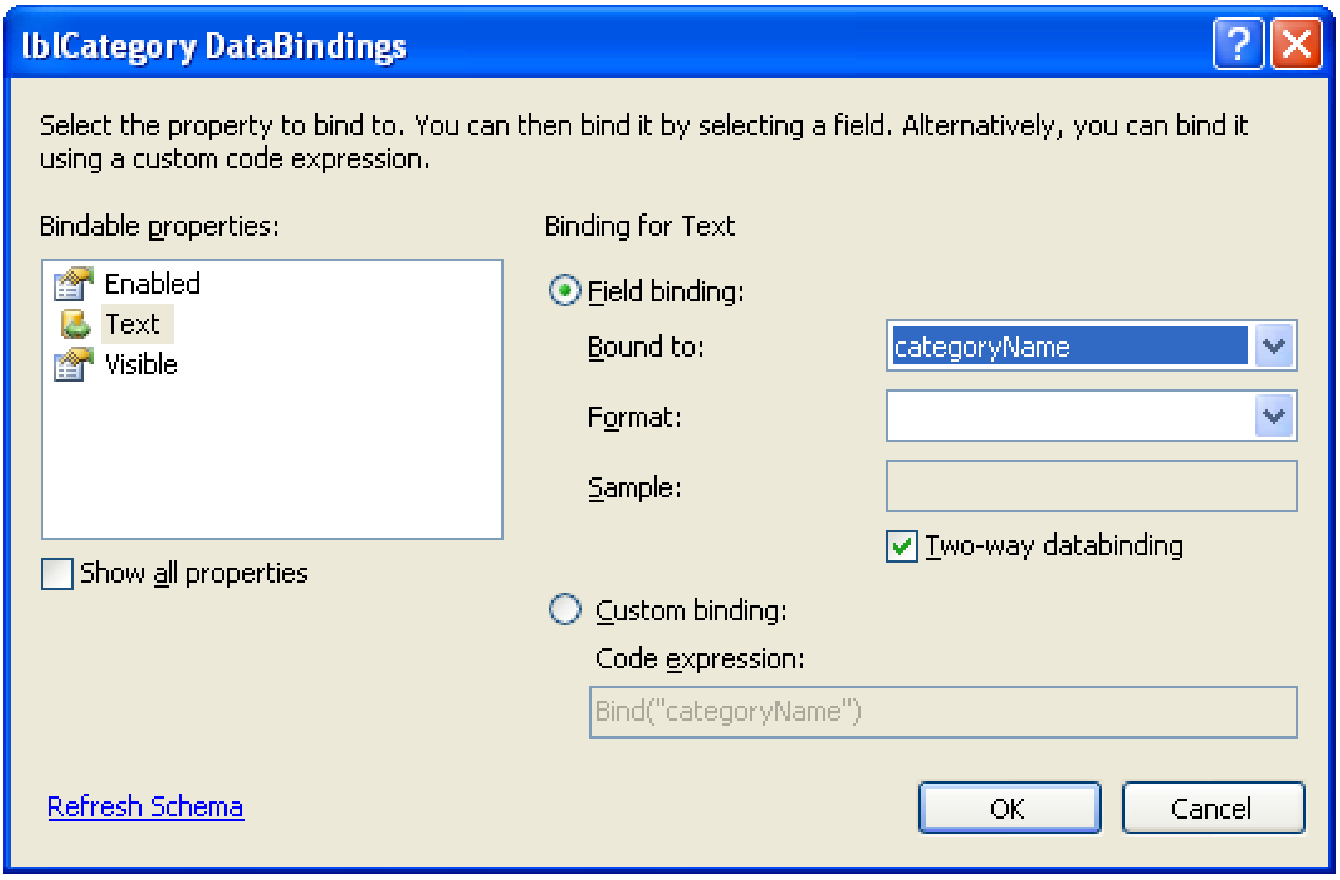
For the item template of the category ID column we would like to display the category name (as names make much more sense that Ids)

Rename the label to lblCategory and click on the smart tag for the label.


gvProducts - Column[4] - CategoryID
ItemTemplate
[lblCategory
Label Tasks
Edit DataBindings...
SqlDataSource - sdsProducts

Select “Edit DataBindings”

Change the text property to categoryName



lblCategory DataBindings

Select the property to bind to. You can then bind it by selecting a field. Alternatively, you can bind it using a custom code expression.

Bindable properties:
- Enabled
- Text
- Visible

☐ Show all properties

Binding for Text

● Field binding:

Bound to: categoryName

Format:

Sample:

☑ Two-way databinding

○ Custom binding:

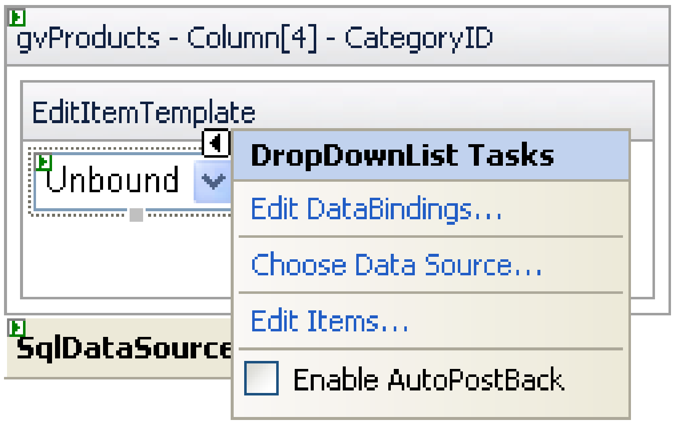
Code expression:

Bind("categoryName")

Click on OK

Now select the editItem template.

A text box will be displayed. Remove the text box and replace it with a drop down list.



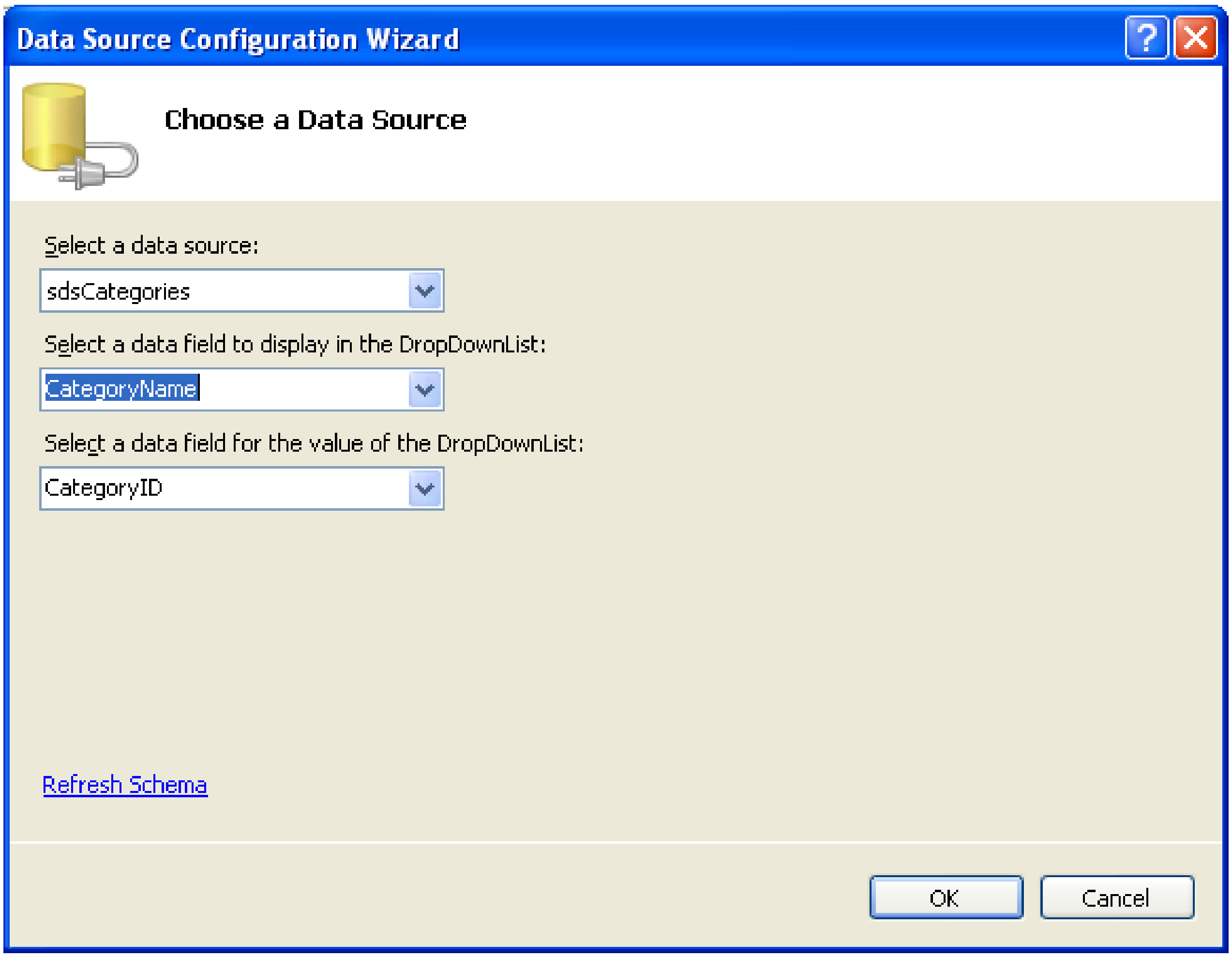
gvProducts - Column[4] - CategoryID

EditItemTemplate

Unbound

Click on choose data source. Add a new data source and set it up to retrieve the category ID and category name from the categories table.

Set up the data source for the drop down list to display the category name and store the category ID as the value.


Data Source Configuration Wizard

Choose a Data Source

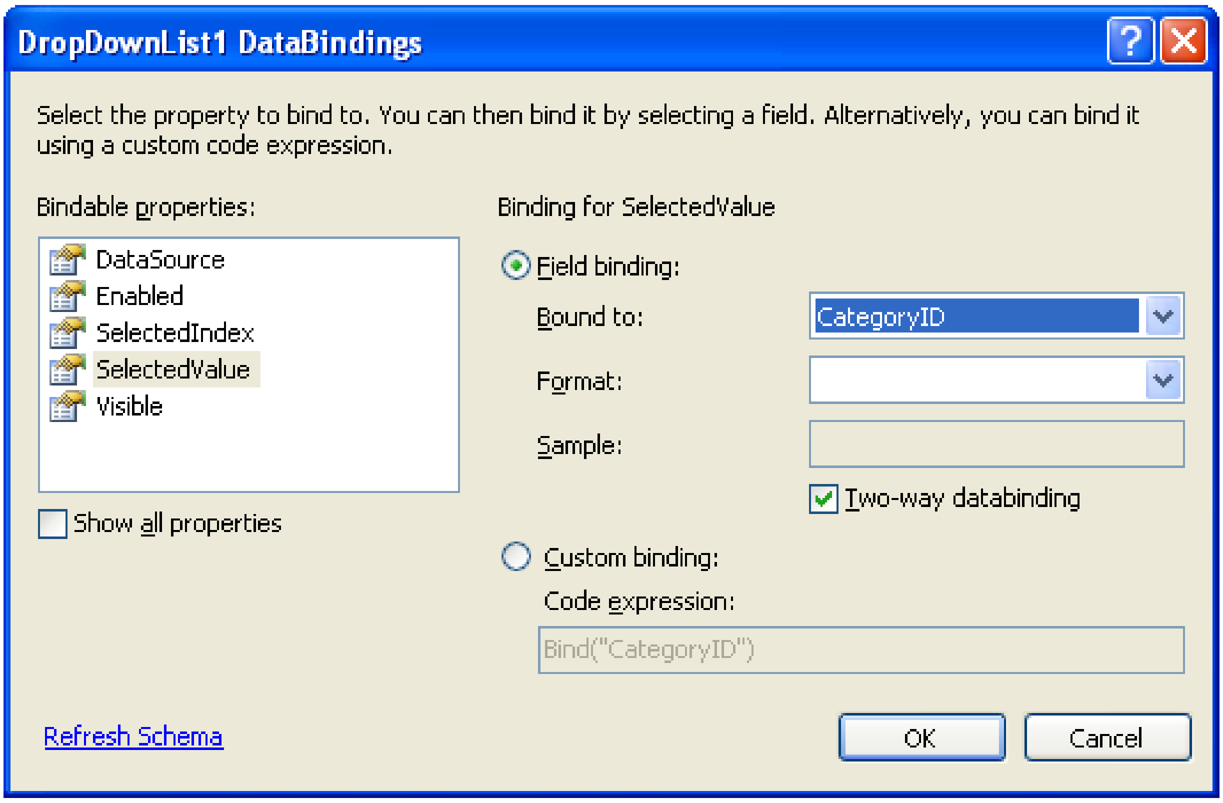
Select a data source: sdsCategories

Select a data field to display in the DropDownList: CategoryName

Select a data field for the value of the DropDownList: CategoryID

Click on OK

Select the “edit dataBinding” link from the drop down list smart tag and set up the dropdown list to bind to the category ID field.


DropDownList1 DataBindings

Select the property to bind to. You can then bind it by selecting a field. Alternatively, you can bind it using a custom code expression.

Bindable properties:

- DataSource
- Enabled
- SelectedIndex
- SelectedValue
- Visible

Show all properties: ☐

Binding for SelectedValue

[✓ Field binding:]

Bound to: CategoryID

Format:

Sample:

Two-way databinding: ✓

Custom binding:

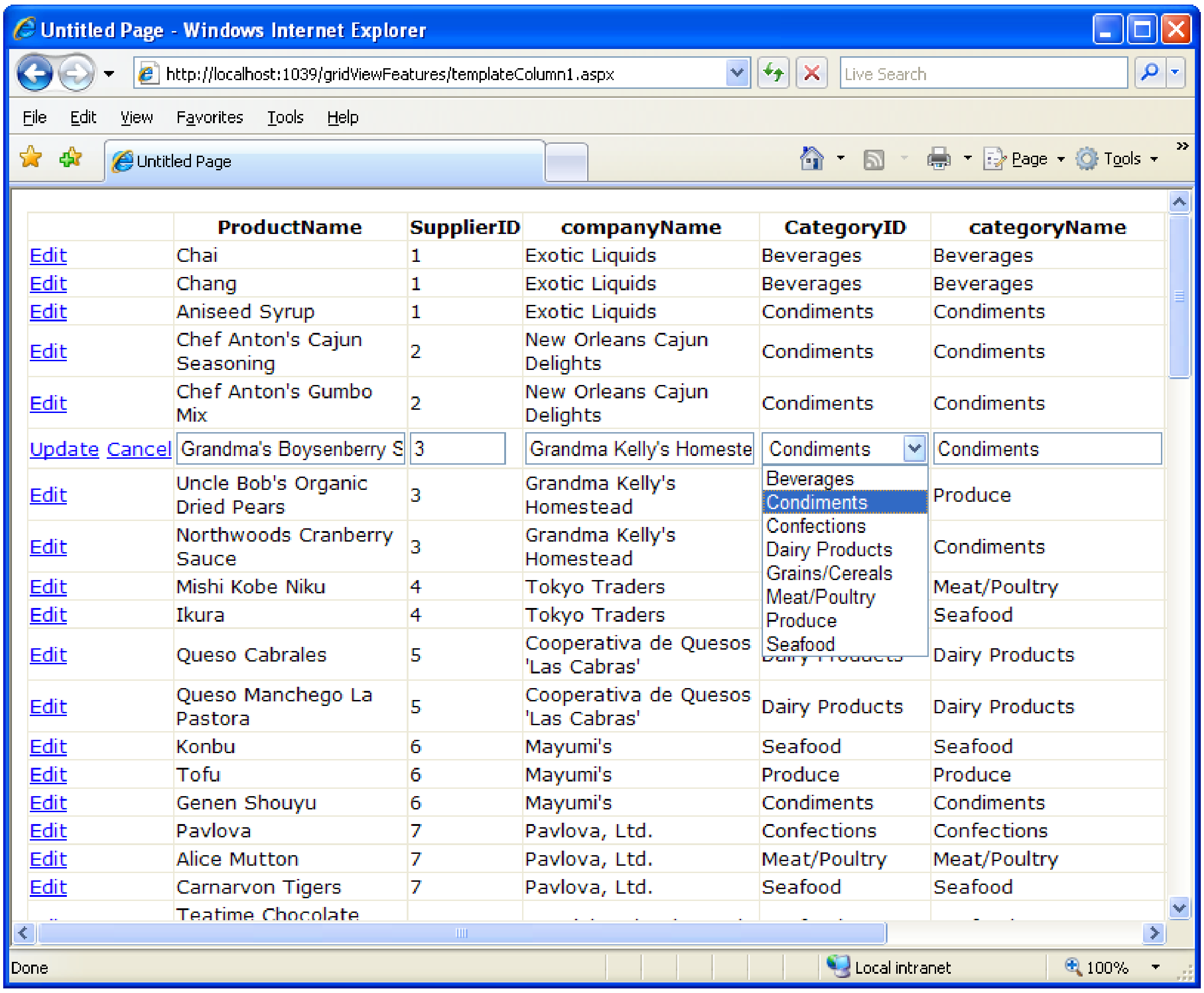
Code expression: Bind("CategoryID")

Click on OK

What we have just created for the category ID column is the following:

- When the data is displayed the category name will be displayed as we humans work better with words than numbers
- When the data is displayed in edit mode a drop down list will be displayed displaying the category name but storing the category ID for its value. When the drop down list is displayed the category name selected will match the category for the current data.

Click on end template editing and view the grid view in your browser.



	ProductName	SupplierID	companyName	CategoryID	categoryName
Edit	Chai	1	Exotic Liquids	Beverages	Beverages
Edit	Chang	1	Exotic Liquids	Beverages	Beverages
Edit	Aniseed Syrup	1	Exotic Liquids	Condiments	Condiments
Edit	Chef Anton's Cajun Seasoning	2	New Orleans Cajun Delights	Condiments	Condiments
Edit	Chef Anton's Gumbo Mix	2	New Orleans Cajun Delights	Condiments	Condiments
Update Cancel	Grandma's Boysenberry S	3	Grandma Kelly's Homeste	Condiments	Condiments
Edit	Uncle Bob's Organic Dried Pears	3	Grandma Kelly's Homestead		Produce
Edit	Northwoods Cranberry Sauce	3	Grandma Kelly's Homestead		Condiments
Edit	Mishi Kobe Niku	4	Tokyo Traders		Meat/Poultry
Edit	Ikura	4	Tokyo Traders		Seafood
Edit	Queso Cabrales	5	Cooperativa de Quesos 'Las Cabras'		Dairy Products
Edit	Queso Manchego La Pastora	5	Cooperativa de Quesos 'Las Cabras'	Dairy Products	Dairy Products
Edit	Konbu	6	Mayumi's	Seafood	Seafood
Edit	Tofu	6	Mayumi's	Produce	Produce
Edit	Genen Shouyu	6	Mayumi's	Condiments	Condiments
Edit	Pavlova	7	Pavlova, Ltd.	Confections	Confections
Edit	Alice Mutton	7	Pavlova, Ltd.	Meat/Poultry	Meat/Poultry
Edit	Carnarvon Tigers	7	Pavlova, Ltd.	Seafood	Seafood
	Teatime Chocolate				

Notice the category ID column displays the category name for each row and displays a drop down list containing all the categories for the row in edit mode.

We should now remove the categoryName column from being displayed.

Now you need to repeat the same process for the supplier ID column.

Rename all column names so that we have product, supplier, category and unit price.

View state

By default the view state stores the values of all the data displayed in the grid view. When a large number of rows are displayed the view state field can become very large. You can disable the view state by setting the enableViewState property to false.

Gridview control events

The grid view control includes a few events that you can write code to. Just as you can write code for a click event from the code behind.

The following events are raised when the control displays its rows:

- DataBinding – Raised immediately before the grid view is bound to its data source
- dataBound – Raised immediately after the grid view is bound to its data source
- rowCreated – Raised when each row in the grid view is created

- rowDataBound – Raised when each row in the grid view is bound to data

The following events are raised when you are editing the data:

- RowCommand – Raised when an event is raised by a control inside the grid view
- RowUpdating – Raised immediately before a grid view updates a record
- RowUpdated – Raised immediately after a grid view updates a record
- RowDeleting - Raised immediately before a grid view deletes a record
- RowDeleted - Raised immediately after a grid view updates a record
- RowCancelingEdit – Raised when you cancel updating a record

There are also a few events relating to the sorting, selecting and paging:

- pageIndexChanging – raised immediately before the current page is changed
- pageIndexChanged – raised immediately after the current page is changed
- Sorting – raised immediately before sorting
- Sorted – raised immediately after sorting
- SelectedIndexChanged – raised immediately before a row is selected
- SelectedIndexChanging – raised immediately after a row is selected

So what you can do is write code to these events from the code behind.

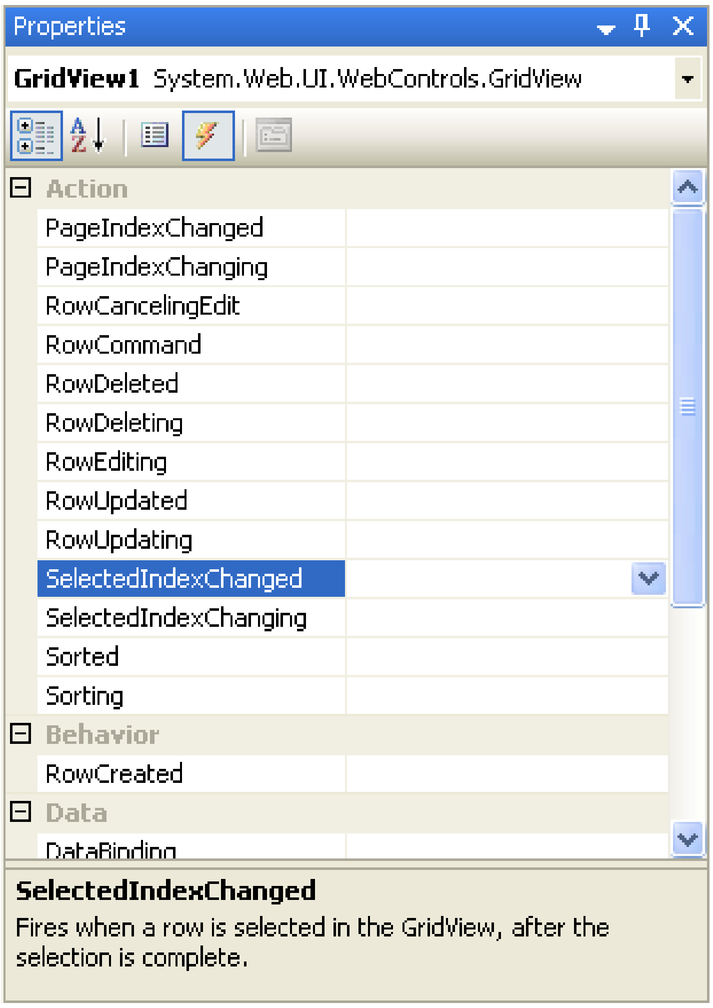
Activity 6. Displaying a different background colour for discontinued items

This activity is going to make use of the rowDataBound event. This event is raised when each row in the grid view is bound to the data.

We will test the value of the discontinued column if that is set to true then we will change the background colour for that row.

Create a new web page and place a grid view onto that page. The grid view will display the product name, unit price and discontinued columns from the products table.

Select the gridview in design view and click on the lighting bolt in the properties panel.



SelectedIndexChanged
Fires when a row is selected in the GridView, after the selection is complete.

This will list the events of the gridview. The event we are interested in is the rowDataBound event. This event is raised when the griview renders each of its rows including the header and footer rows. To add code for that event double click on the event name. This will open up the code behind file:

```
protected void GridView1_RowDataBound(object sender, GridViewRowEventArgs e)
{

}
```

The above method will be executed for each row bound to the gridview. We can place some code in here to manipulate the way the data is displayed. The second parameter, e, is of type GridViewRowEventArgs. This is a class which contains a gridViewRow object that represents the current row being bound.

Some common properties of the gridViewRow object include:

- Cells – Represents a collection of table row cells associated with the row being bound.
- DataItem – represents the dataItem associated with the row being bound (this is the data from the select statement)
- DataItemIndex – the index of the data item row
- RowIndex – index of the row being bound
- rowState – represents the state of the row being bound. Can be (alternate, normal, selected, edit)
- rowType – represents the type of row being bound. Can be (dataRow, footer, header, nullRow, pager, separator)

Place the following code in the method:

```
protected void GridView1_RowDataBound(object sender, GridViewRowEventArgs e)
{
      bool blnDiscontinued;

      //if the discontinued column is set to true
      //change the background colour of the row

      //only change the BG colour for a datarow
      //i.e. not heading or footer
      if (e.Row.RowType == DataControlRowType.DataRow)
      {
        //store the value of the discontinued column
        //inside a boolean variable blnDiscontinued
        //DataBinder.Eval is used to retreive
        //the value of the discontinued column
        blnDiscontinued = (bool)(DataBinder.Eval(e.Row.DataItem, "discontinued"));

        //check if it is set to true
        //this means the product is discontinued
        if (blnDiscontinued == true)
        {
          e.Row.BackColor = System.Drawing.Color.Aquamarine;
        }
      }
}
```

Row Created event

This event is raised when a row is created. We can add some code to that event.

Activity 7. Row created event

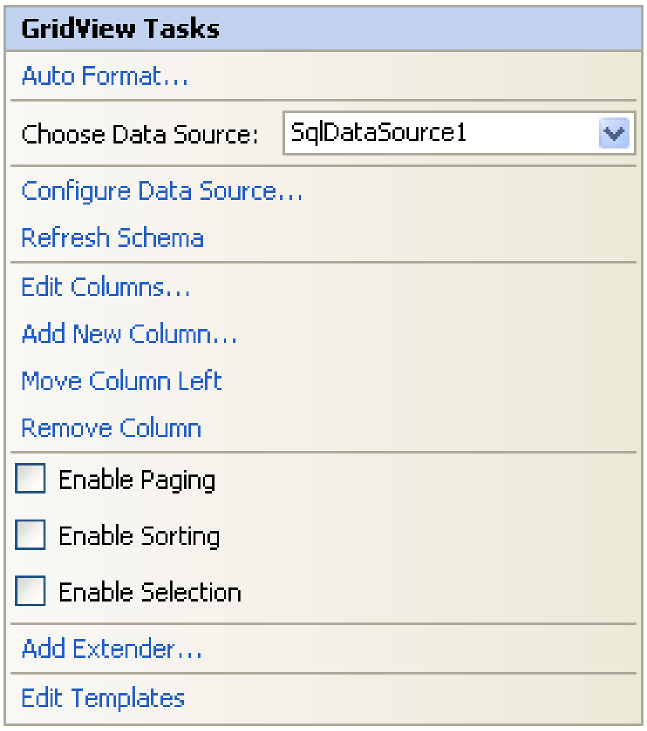
gridViewEvents.aspx

In this activity we will display a record count in a column.

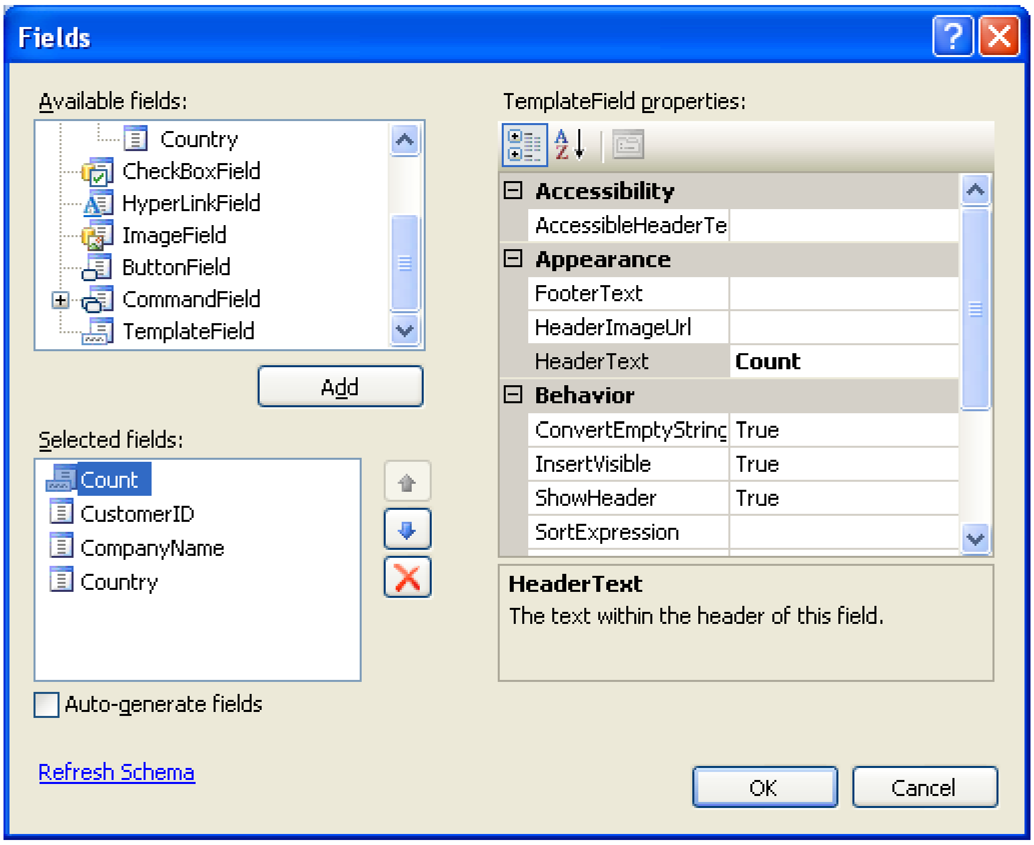
Add to your web page an sqlDataSource control and configure it to retrieve the customerID, companyName and country from the customers table.

Add a gridview to the page.

Select editColumns from the gridview smart menu.



Add a template field. Give it the heading text “Count”.



AccessibleHeaderTe	
FooterText	
HeaderImageUrl	
HeaderText	Count
ConvertEmptyStrinc	True
InsertVisible	True
ShowHeader	True
SortExpression	

HeaderText
The text within the header of this field.

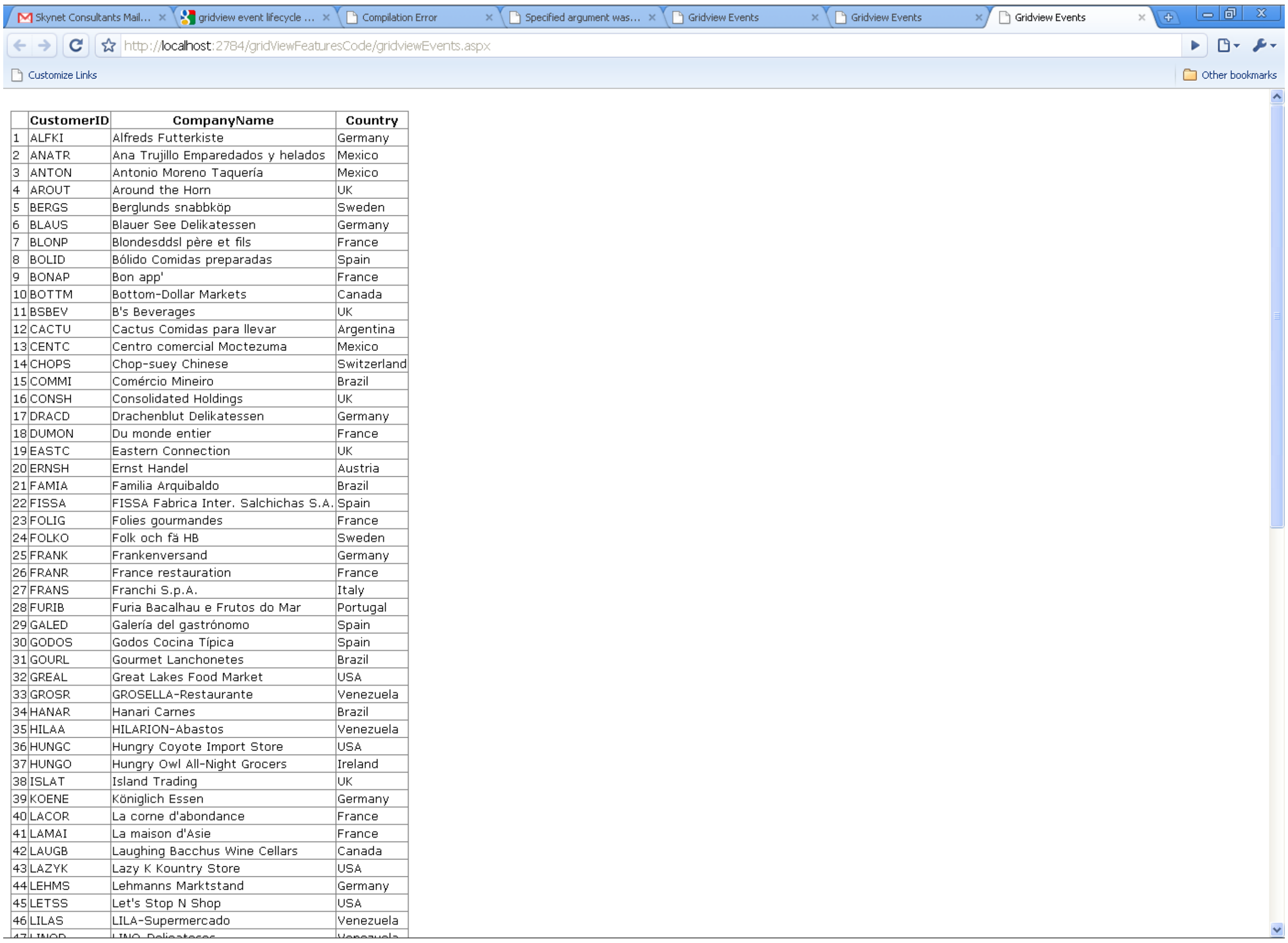
While the gridview is selected select the lighting bolt to view the gridview events.

Double click on the RowCreated event.

Add the following code:

```
protected void GridView1_RowCreated(object sender, GridViewRowEventArgs e)
{
        //only do this for a data row i.e not header or footer
        if (e.Row.RowType == DataControlRowType.DataRow)
        {
            //set the text for the first column
            //to be equal to the rowindex + 1
            e.Row.Cells[0].Text = (e.Row.RowIndex + 1).ToString();
        }
}
```

When you view the page you should see thegridview displaying a row number for each row:


Skynet Consultants Mail...
gridview event lifecycle ...
Compilation Error
Specified argument was...
Gridview Events
Gridview Events
Gridview Events
http://localhost:2784/gridViewFeaturesCode/gridviewEvents.aspx
Customize Links
Other bookmarks

	CustomerID	CompanyName	Country
1	ALFKI	Alfreds Futterkiste	Germany
2	ANATR	Ana Trujillo Emparedados y helados	Mexico
3	ANTON	Antonio Moreno Taquería	Mexico
4	AROUT	Around the Horn	UK
5	BERGS	Berglunds snabbköp	Sweden
6	BLAUS	Blauer See Delikatessen	Germany
7	BLONP	Blondesddsl père et fils	France
8	BOLID	Bólido Comidas preparadas	Spain
9	BONAP	Bon app'	France
10	BOTTM	Bottom-Dollar Markets	Canada
11	BSBEV	B's Beverages	UK
12	CACTU	Cactus Comidas para llevar	Argentina
13	CENTC	Centro comercial Moctezuma	Mexico
14	CHOPS	Chop-suey Chinese	Switzerland
15	COMMI	Comércio Mineiro	Brazil
16	CONSH	Consolidated Holdings	UK
17	DRACD	Drachenblut Delikatessen	Germany
18	DUMON	Du monde entier	France
19	EASTC	Eastern Connection	UK
20	ERNSH	Ernst Handel	Austria
21	FAMIA	Familia Arquibaldo	Brazil
22	FISSA	FISSA Fabrica Inter. Salchichas S.A.	Spain
23	FOLIG	Folies gourmandes	France
24	FOLKO	Folk och fä HB	Sweden
25	FRANK	Frankenversand	Germany
26	FRANR	France restauration	France
27	FRANS	Franchi S.p.A.	Italy
28	FURIB	Furia Bacalhau e Frutos do Mar	Portugal
29	GALED	Galería del gastrónomo	Spain
30	GODOS	Godos Cocina Típica	Spain
31	GOURL	Gourmet Lanchonetes	Brazil
32	GREAL	Great Lakes Food Market	USA
33	GROSR	GROSELLA-Restaurante	Venezuela
34	HANAR	Hanari Carnes	Brazil
35	HILAA	HILARION-Abastos	Venezuela
36	HUNGC	Hungry Coyote Import Store	USA
37	HUNGO	Hungry Owl All-Night Grocers	Ireland
38	ISLAT	Island Trading	UK
39	KOENE	Königlich Essen	Germany
40	LACOR	La corne d'abondance	France
41	LAMAI	La maison d'Asie	France
42	LAUGB	Laughing Bacchus Wine Cellars	Canada
43	LAZYK	Lazy K Kountry Store	USA
44	LEHMS	Lehmanns Marktstand	Germany
45	LETSS	Let's Stop N Shop	USA
46	LILAS	LILA-Supermercado	Venezuela

References

ASP.NET 2.0 Unleashed chapter 11