Data Controls Combined

Contents

So far we have seen the controls that are provided when you need to display, edit, add and delete data. These controls are:

- Gridview
- Datalist
- Repeater
- DetailsView
- FormView
- ListView

Each of the above controls have their own advantages and disadvantages. It's up to you as the web developer to decide which control will best match your needs. You can also use these controls together.

Master / detail

Some applications display many rows of data allowing one to be selected, once selected the complete details of that row are displayed in a details view or form view control. The master data control could be a gridview, listview, datalist or repeater control.

Activity 1. Creating a master detail page

masterDetail.aspx

In this exercise we will create a master table displaying the product name and unit price.

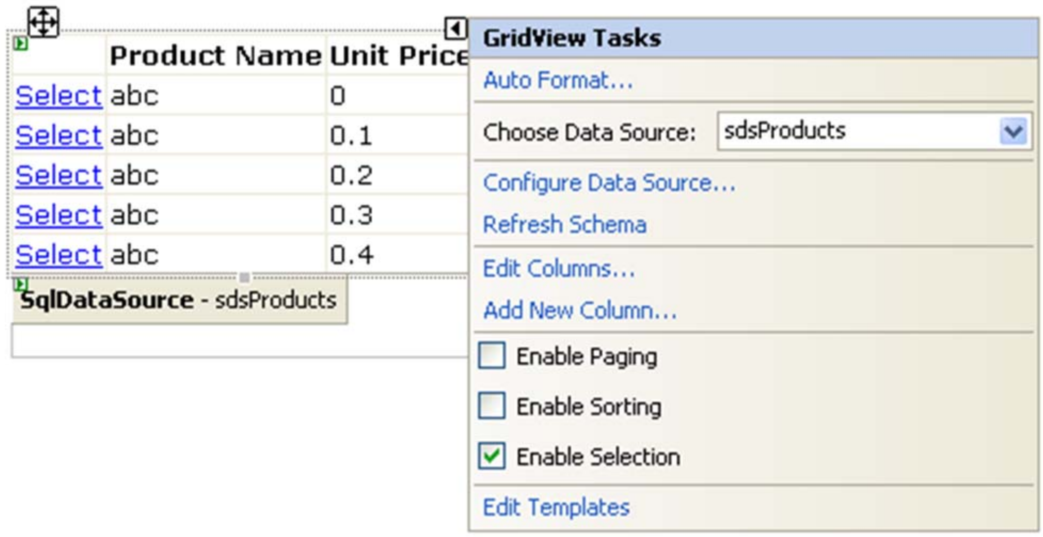
Create a new web page and add a gridview control to it.

Set up the datasource for the control so that it retrieves the product ID, product name and unit price for all products.

Only display the product name and unit price by deleting the product ID column. We need to retrieve the product ID in our SQL statement so that when the row is selected the details of the product can be retrieved using the product ID.

Enable selection.

Set up your style sheet so that a selected product will appear with a different back ground colour.



	Product Name	Unit Price
Select	abc	0
Select	abc	0.1
Select	abc	0.2
Select	abc	0.3
Select	abc	0.4

SqlDataSource - sdsProducts

GridView Tasks

Auto Format...

Choose Data Source: sdsProducts

Configure Data Source...

Refresh Schema

Edit Columns...

Add New Column...

☐ Enable Paging

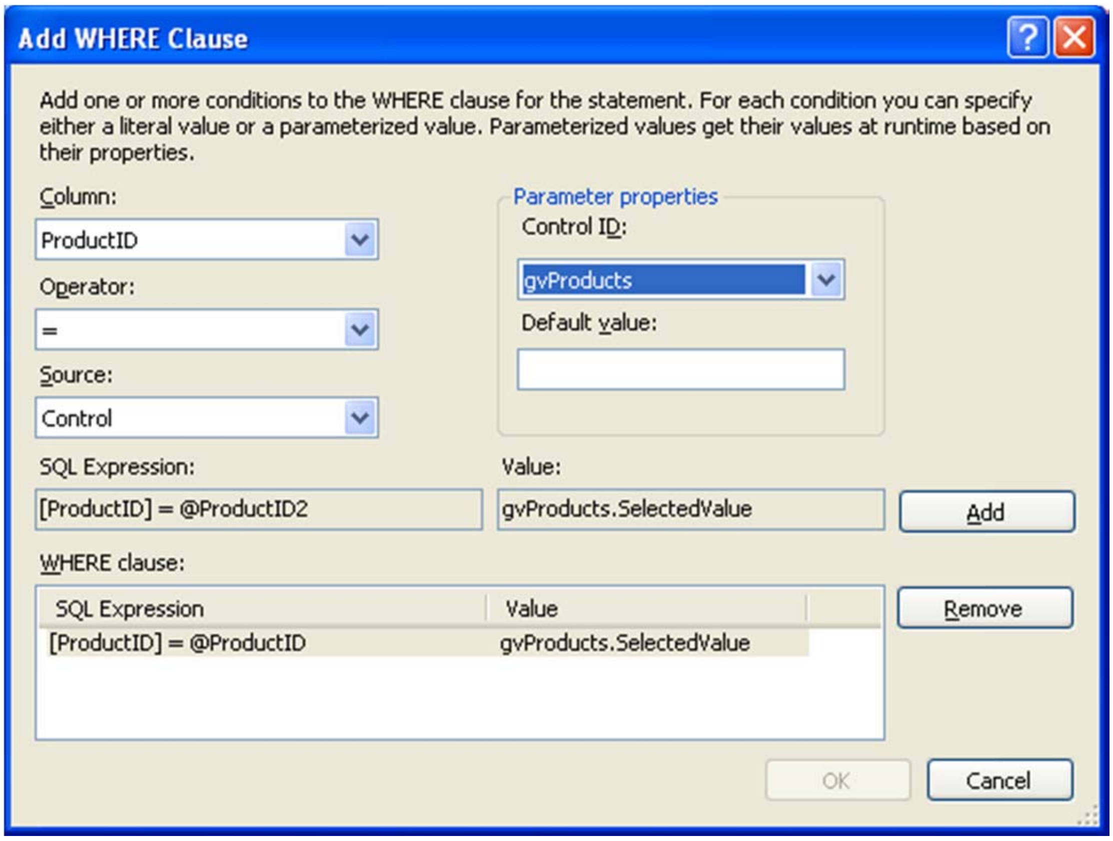
☐ Enable Sorting

☑ Enable Selection

Edit Templates

Add a details view control to the page.

Set up the data source control to retrieve all the data from the products table for the selected product. Display the category name and supplier company name instead of the IDs. The where clause will need to get the product ID from the gridView control.



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: ProductID

Operator: =

Source: Control

Parameter properties

Control ID: gvProducts

Default value:

SQL Expression: [ProductID] = @ProductID2

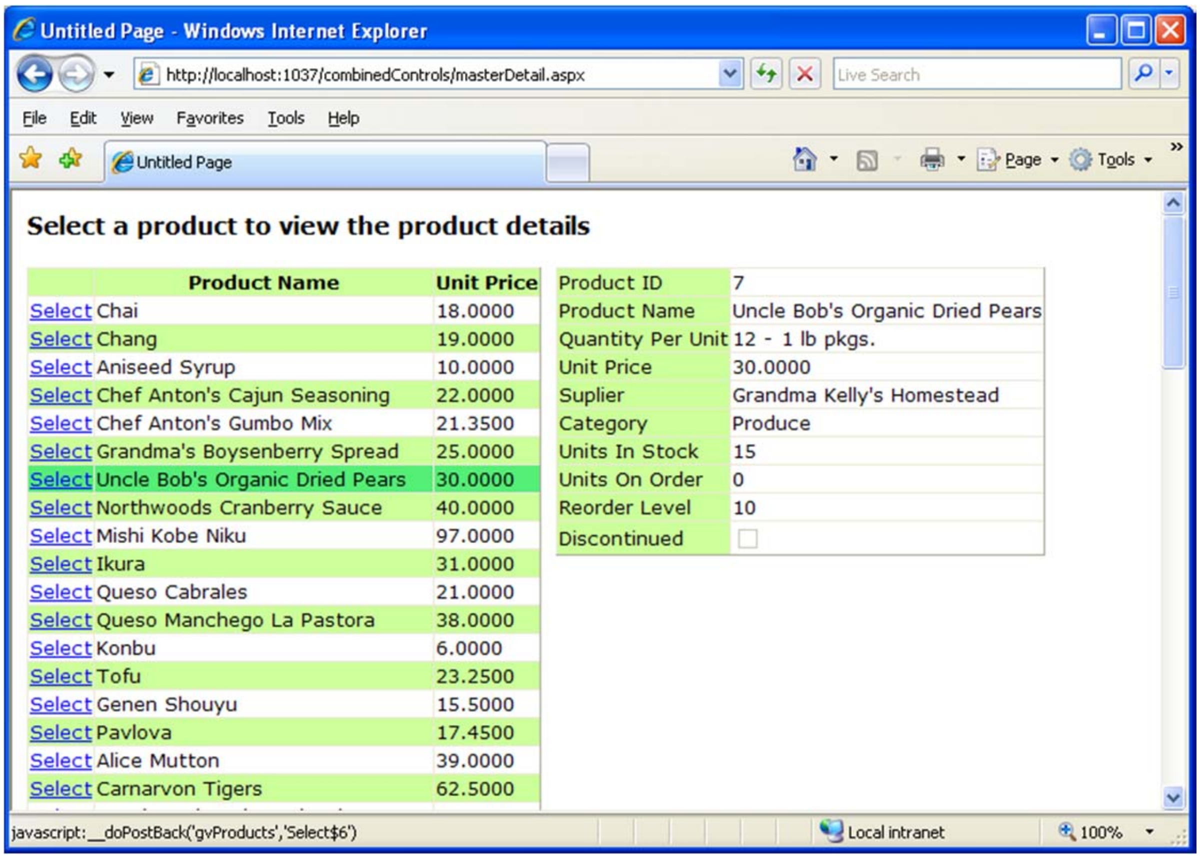
Value: gvProducts.SelectedValue

WHERE clause:

SQL Expression	Value
[ProductID] = @ProductID	gvProducts.SelectedValue

The resulting SQL statement is

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


Untitled Page - Windows Internet Explorer
http://localhost:1037/combinedControls/masterDetail.aspx
Live Search
File Edit View Favorites Tools Help
Untitled Page
Page Tools

Select a product to view the product details

	Product Name	Unit Price
Select	Chai	18.0000
Select	Chang	19.0000
Select	Aniseed Syrup	10.0000
Select	Chef Anton's Cajun Seasoning	22.0000
Select	Chef Anton's Gumbo Mix	21.3500
Select	Grandma's Boysenberry Spread	25.0000
Select	Uncle Bob's Organic Dried Pears	30.0000
Select	Northwoods Cranberry Sauce	40.0000
Select	Mishi Kobe Niku	97.0000
Select	Ikura	31.0000
Select	Queso Cabrales	21.0000
Select	Queso Manchego La Pastora	38.0000
Select	Konbu	6.0000
Select	Tofu	23.2500
Select	Genen Shouyu	15.5000
Select	Pavlova	17.4500
Select	Alice Mutton	39.0000
Select	Carnarvon Tigers	62.5000

Product ID	7
Product Name	Uncle Bob's Organic Dried Pears
Quantity Per Unit	12 - 1 lb pkgs.
Unit Price	30.0000
Suplier	Grandma Kelly's Homestead
Category	Produce
Units In Stock	15
Units On Order	0
Reorder Level	10
Discontinued	☐

javascript:__doPostBack('gvProducts','Select$6')

Nested master/details

You can also display the details inside the gridview, datalist, listview or repeater control itself. To do this we need to make use of the control's events.

Activity 2. Nested master/detail

nested.aspx

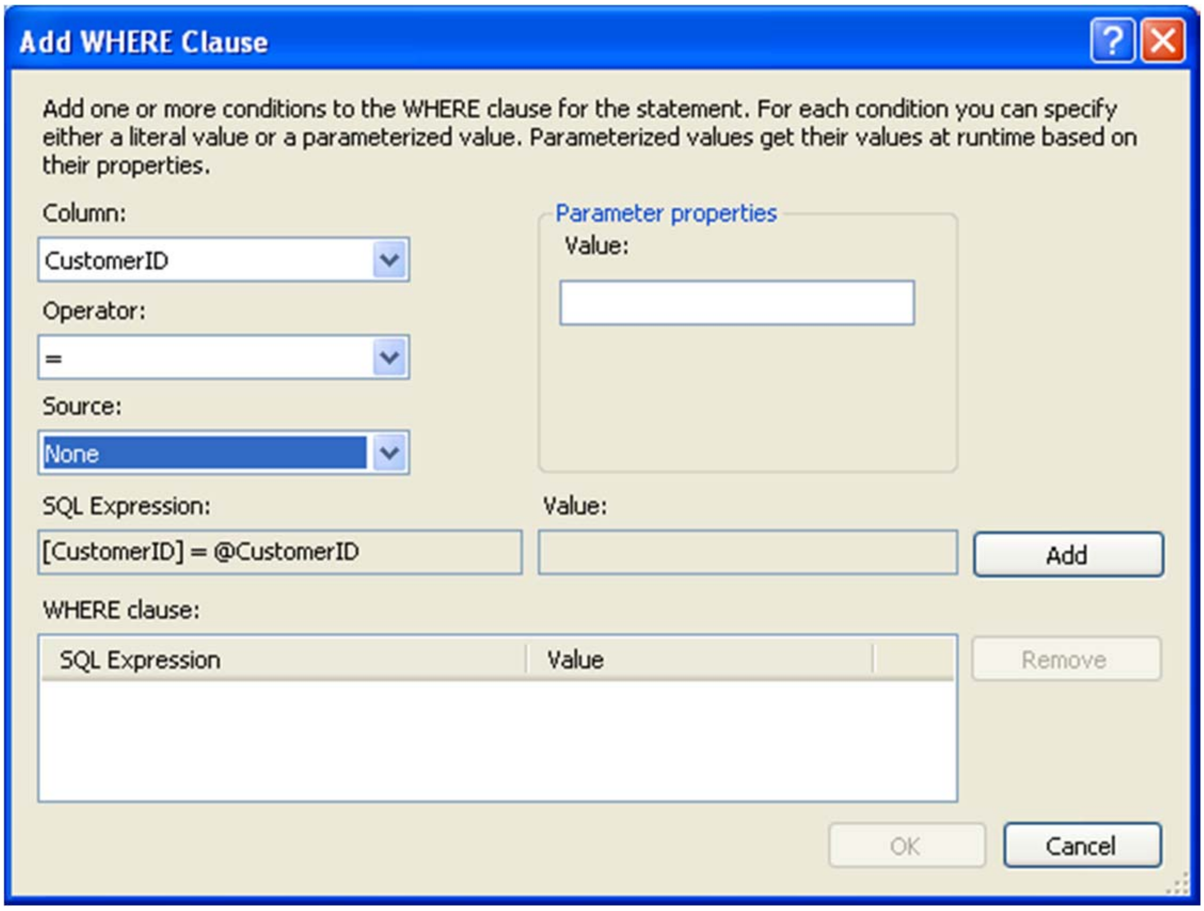
Create a web page which displays all the customers, display the customer ID, the company name, customer contact name and phone number. Use a datalist control.

Format it so that the customer ID and company name are on one line in bold and the customer contact name and phone number on the line below. You will need to edit the item template.

```
<asp:DataList ID="dlCustomers" runat="server" DataKeyField="CustomerID"
DataSourceID="sdsCustomers">
     <ItemTemplate>
          <asp:Label ID="CustomerIDLabel" runat="server"
          CssClass="bold" Text='<%# Eval("CustomerID") %>'>
          </asp:Label>
          :<asp:Label ID="CompanyNameLabel" runat="server" CssClass="bold"
          Text='<%# Eval("CompanyName") %>'>
          </asp:Label><br />
          <asp:Label ID="ContactNameLabel" runat="server"
          Text='<%# Eval("ContactName") %>'></asp:Label>
          Phone:
          <asp:Label ID="PhoneLabel" runat="server"
          Text='<%# Eval("Phone") %>'></asp:Label>
     </ItemTemplate>
</asp:DataList>
```

Make every second row a different colour so that it is easier to read.

Now modify the item template so that it contains an SQL data source control. Set up the SQLDataSource control so that it retrieves the order ID and order date from the orders table. Add a where clause and set it up to retrieve the orders for the customer:


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.

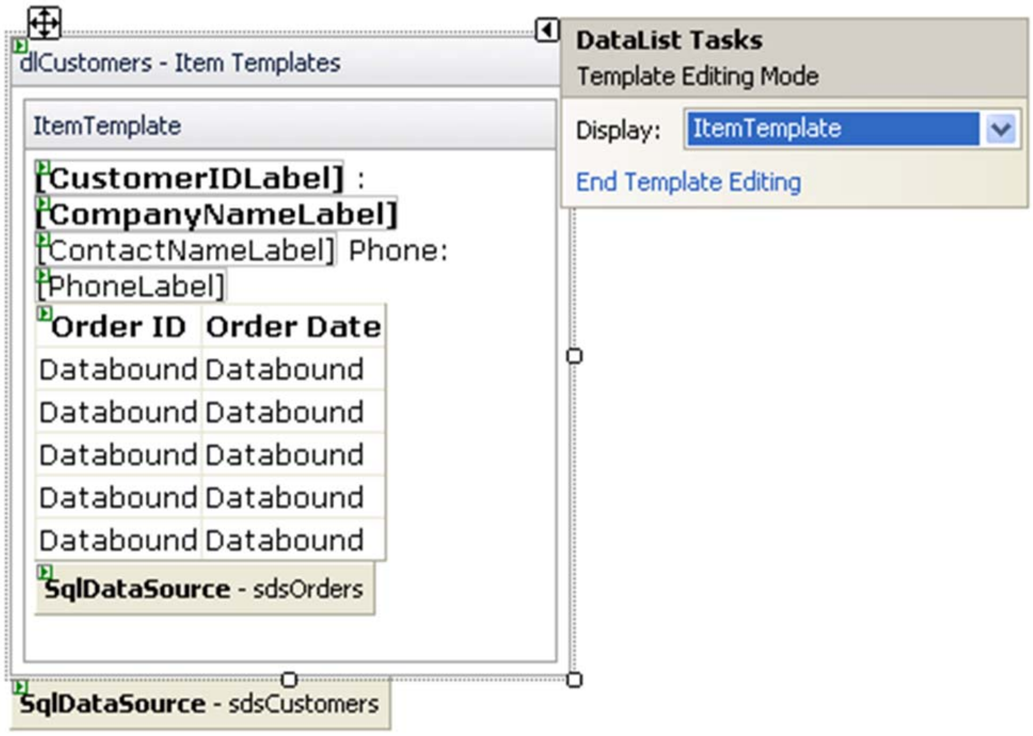
Column: CustomerID

Operator: =

Source: None

SQL Expression: [CustomerID] = @CustomerID

Add a gridview control to the item template and assign the data source control to it.



So what we have done so far is set up the data source control to retrieve all the orders from the orders table based on a customer ID passed to it.

Now we need to give the data source control the value of the customer ID. This will be different for each customer so we need to pass it each time a row (item) is bound to the data list. The event which handles this is the itemDataBound event it is executed each time a row (item) is bound to the datalist.

From the code behind file add the following code:

```
protected void dlCustomers_ItemDataBound(object sender, DataListItemEventArgs
e)
```

```
{
   string strCustID;
   SqlDataSource sdsOrdersTemp;

   //only do this for an item or alternating item
   if ((e.Item.ItemType == ListItemType.Item) ||
      (e.Item.ItemType == ListItemType.AlternatingItem))
   {
      strCustID = (string)(DataBinder.Eval(e.Item.DataItem,
      "customerID"));
      sdsOrdersTemp = (SqlDataSource)(e.Item.FindControl("sdsOrder"));
      sdsOrdersTemp.SelectParameters["customerID"].DefaultValue
      = strCustID;
   }
}
```

The if statement:

```
if ((e.Item.ItemType == ListItemType.Item) ||
   (e.Item.ItemType == ListItemType.AlternatingItem))
```

Checks to see if the item type is of type item or alternating item. This means it's not the header or footer.

Next we read in the customer ID for the datalist:

```
strCustID = (string)(DataBinder.Eval(e.Item.DataItem, "customerID"));
```

We read the customer ID by accessing the customerID column in the dataItem (a data item is the current row)

Then we create a data source control so that we can allocate the customer ID for the parameters:

```
sdsOrdersTemp = (SqlDataSource)(e.Item.FindControl("sdsOrder"));
```

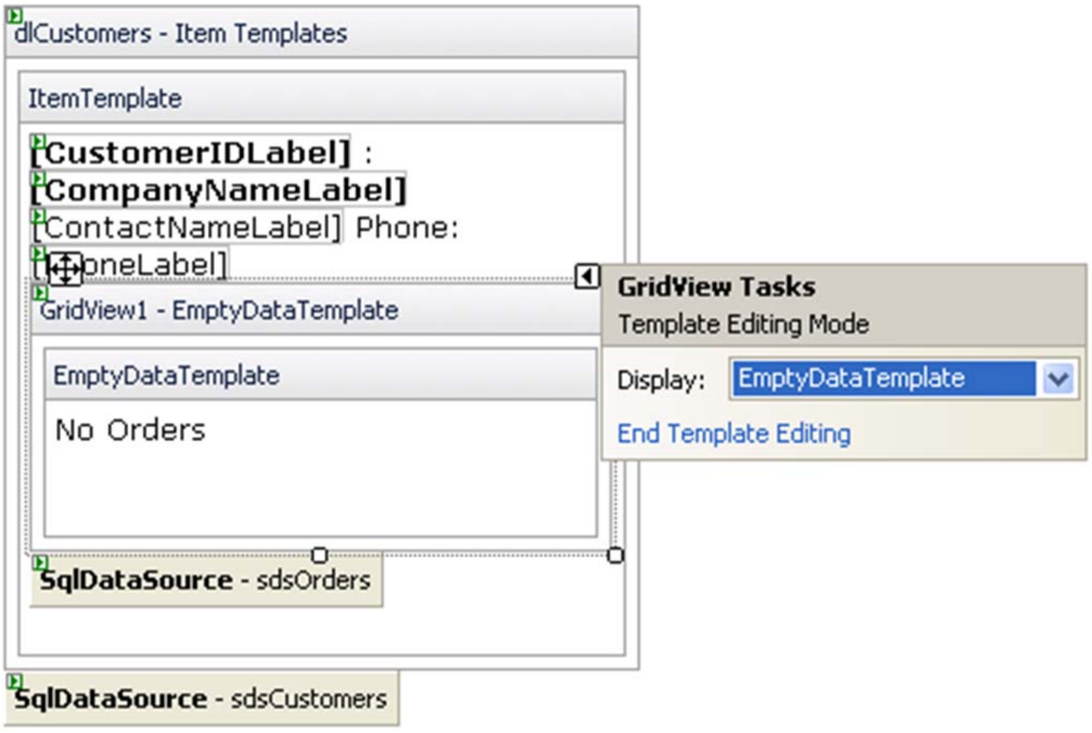
Lastly we allocate the customer ID read in above to the select parameter:

```
sdsOrdersTemp.SelectParameters["customerID"].DefaultValue = strCustID;
```

This results in the current customer ID being allocated to the where clause of the SQL statement.

Have a look at the web page.

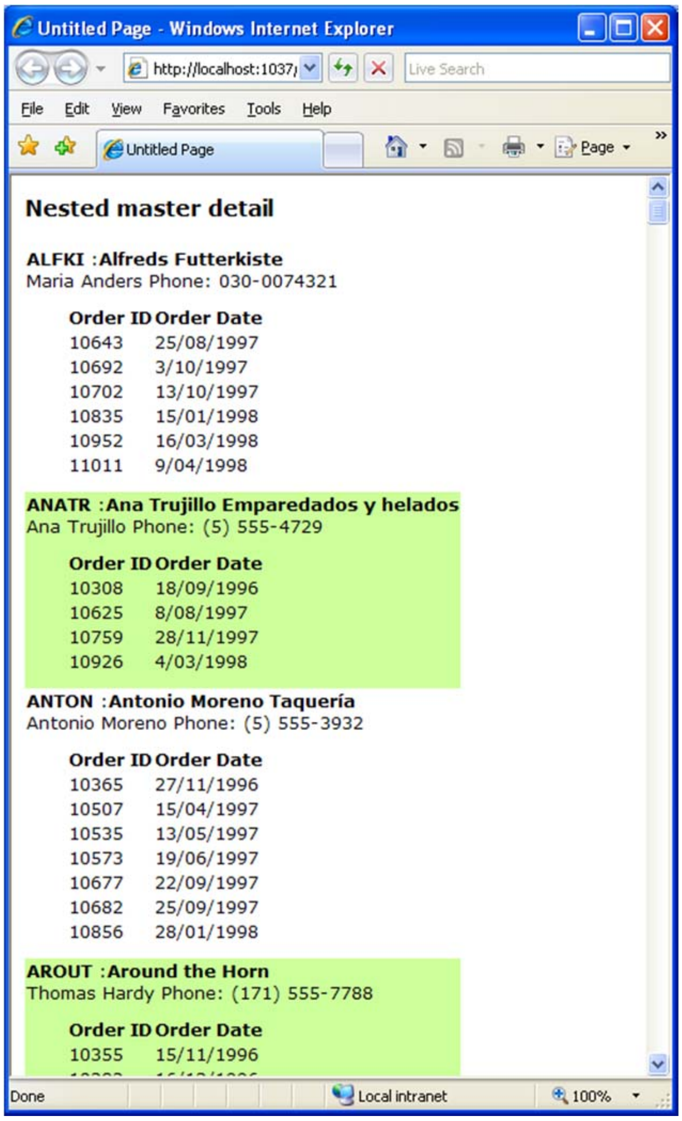
Now all that needs to be done is some formatting so that it is more readable. Firstly some customers don't have any orders, it would be better if we could display "No orders" for those customers. To do this edit the emptyDataTemplate for the gridView and add the text "No Orders"



dlCustomers - Item Templates
ItemTemplate
[CustomerIDLabel] :
[CompanyNameLabel]
[ContactNameLabel] Phone:
[PhoneLabel]
GridView1 - EmptyDataTemplate
EmptyDataTemplate
No Orders
GridView Tasks
Template Editing Mode
Display: EmptyDataTemplate
End Template Editing
SqlDataSource - sdsOrders
SqlDataSource - sdsCustomers

The orders for the grid view are displayed in a table it would look better if that table was indented and the borders removed.

The resulting page should look similar to this:


Untitled Page - Windows Internet Explorer
http://localhost:1037/
Live Search
File Edit View Favorites Tools Help
Untitled Page
Page
Done
Local intranet
100%

Nested master detail

ALFKI :Alfreds Futterkiste
Maria Anders Phone: 030-0074321

Order ID	**Order Date**
10643	25/08/1997
10692	3/10/1997
10702	13/10/1997
10835	15/01/1998
10952	16/03/1998
11011	9/04/1998

ANATR :Ana Trujillo Emparedados y helados
Ana Trujillo Phone: (5) 555-4729

Order ID	**Order Date**
10308	18/09/1996
10625	8/08/1997
10759	28/11/1997
10926	4/03/1998

ANTON :Antonio Moreno Taquería
Antonio Moreno Phone: (5) 555-3932

Order ID	**Order Date**
10365	27/11/1996
10507	15/04/1997
10535	13/05/1997
10573	19/06/1997
10677	22/09/1997
10682	25/09/1997
10856	28/01/1998

AROUT :Around the Horn
Thomas Hardy Phone: (171) 555-7788

Order ID	**Order Date**
10355	15/11/1996

The resulting source code is as follows:

```
<asp:DataList ID="dlCustomers" runat="server" DataKeyField="CustomerID"
DataSourceID="sdsCustomers">
      <ItemTemplate>
            <asp:Label ID="CustomerIDLabel" runat="server"
            CssClass="bold" Text='<%# Eval("CustomerID") %>'>
            </asp:Label>
            :<asp:Label ID="CompanyNameLabel" runat="server"
            CssClass="bold" Text='<%# Eval("CompanyName") %>'>
            </asp:Label><br />
            <asp:Label ID="ContactNameLabel" runat="server"
            Text='<%# Eval("ContactName") %>'></asp:Label>
            Phone:
            <asp:Label ID="PhoneLabel" runat="server"
            Text='<%# Eval("Phone") %>'></asp:Label><br />
            <asp:GridView ID="GridView1" runat="server"
            AutoGenerateColumns="False" DataKeyNames="OrderID"
            DataSourceID="sdsOrders" CssClass="orders"
            GridLines="None">
                  <Columns>
                  <asp:BoundField DataField="OrderID"
                  HeaderText="Order ID" InsertVisible="False"
                  ReadOnly="True" SortExpression="OrderID" />
                  <asp:BoundField DataField="OrderDate"
                  DataFormatString="{0:d}" HeaderText="Order Date"
                  HtmlEncode="False" SortExpression="OrderDate" />
```

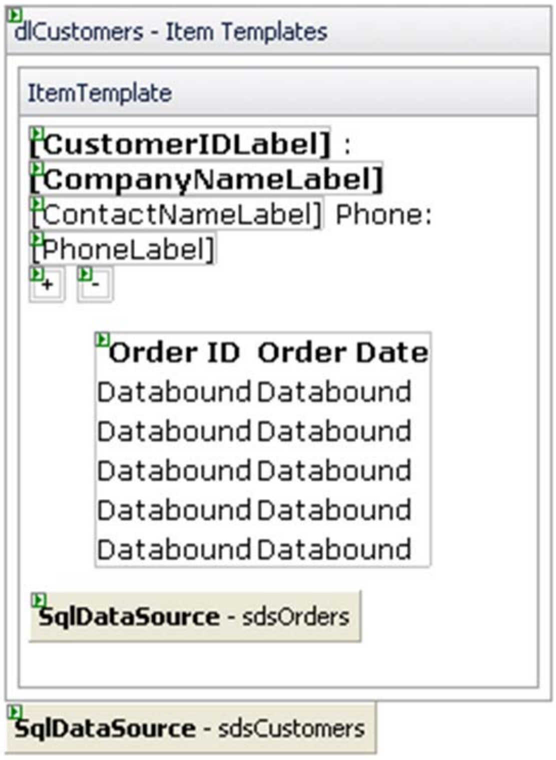
```
                </Columns>
                <EmptyDataTemplate>
                        No Orders
                </EmptyDataTemplate>
            </asp:GridView>
            <asp:SqlDataSource ID="sdsOrders" runat="server"
      ConnectionString="<%$ ConnectionStrings:ConnectionString %>"
      SelectCommand="SELECT [OrderID], [OrderDate] FROM [Orders]
      WHERE ([CustomerID] = @CustomerID)">
            <SelectParameters>
                 <asp:Parameter Name="CustomerID" Type="String" />
             </SelectParameters>
        </asp:SqlDataSource>
      </ItemTemplate>
      <AlternatingItemStyle CssClass="alternateStyle" />
</asp:DataList>
<asp:SqlDataSource ID="sdsCustomers" runat="server" ConnectionString="<%$
ConnectionStrings:ConnectionString %>"
SelectCommand="SELECT [CustomerID], [CompanyName], [ContactName], [Phone] FROM
[Customers]">
</asp:SqlDataSource>
```

Activity 3. Nested gridview part 2

nested2.aspx

It would be even better if the gridview containing the orders was only displayed when the user clicked on an expand button.

To do this we need to add 2 image buttons, one for expand and one for collapse, to the itemTemplate of the datalist.



Make the collapse image button invisible by setting the visible property to false. Make the gridview invisible by setting the visible attribute to false.

For the expand image button allocate the value “expand” to the commandArgument and allocate the value “collapse” to the commandArgument for the collapse image button.

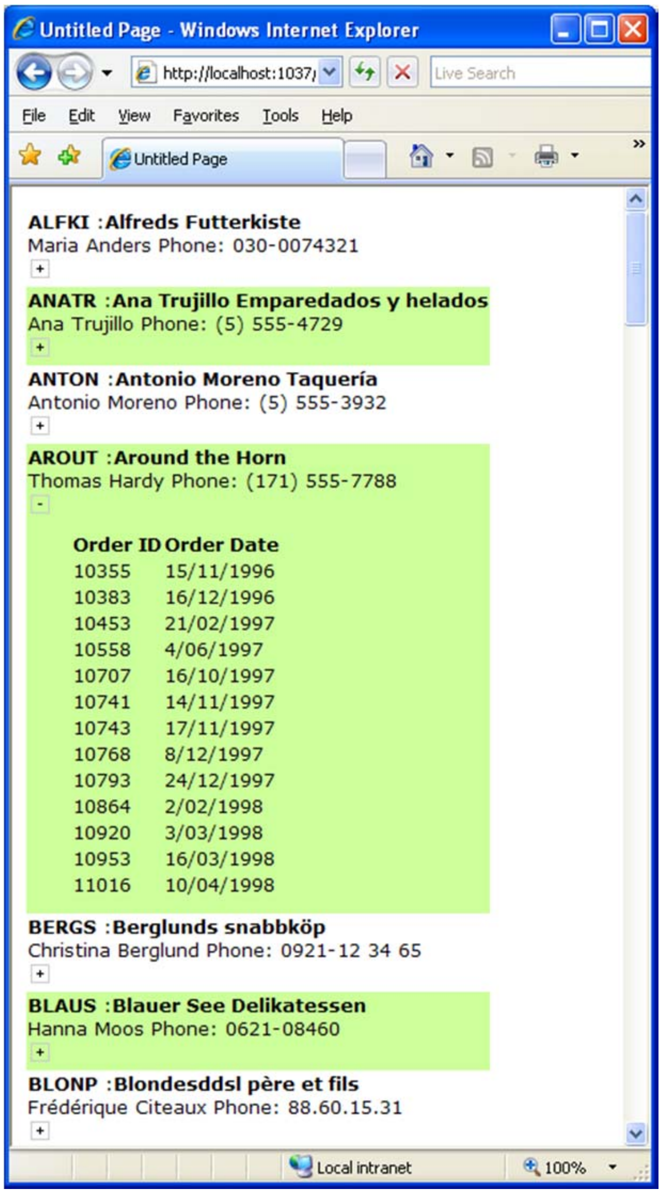
We no longer need to retreive the grid view for each row so you can remove the itemDatabound event. Instead we want to retreive the gridview when the expand button is clicked. To do this you need to add the following code to the code behind file:

```
protected void dlCustomers_ItemCommand(object source, DataListCommandEventArgs
e)
{
   string strCustID;
   SqlDataSource sdsOrdersTemp;
   ImageButton imbExpandTemp;
   ImageButton imbCollapseTemp;
   GridView gvOrdersTemp;
   Label lblCustIDTemp;

   //store the 2 image buttons in variables so that they
   //can have their visible property set
   imbCollapseTemp = (ImageButton)e.Item.FindControl("imbCollapse");
   imbExpandTemp = (ImageButton)e.Item.FindControl("imbExpand");

   //store the gridview in a variable so it
   //can have its visible attribute changed and
   //data bound to it
   gvOrdersTemp = (GridView)e.Item.FindControl("gvOrders");

   //if the expand button is clicked
   if (e.CommandArgument == "expand")
   {
      //read the customer ID from the label
      lblCustIDTemp = (Label)e.Item.FindControl("CustomerIDLabel");
      strCustID = lblCustIDTemp.Text;

      //set the customer ID read from the label
      //to the data source control
      sdsOrdersTemp =  (SqlDataSource)e.Item.FindControl("sdsOrder");
      sdsOrdersTemp.SelectParameters["customerID"].DefaultValue = strCustID;

      //make the gridview visible and bind the data
      gvOrdersTemp.DataBind();
      gvOrdersTemp.Visible = true;

      //make the expand image button invisible
      //and the collapse button visible
      imbExpandTemp.Visible = false;
      imbCollapseTemp.Visible = true;
   }
   else
   {
      //make the gridview invisible
      //make the expand button visible
      //make the collapse button invisible
      gvOrdersTemp.Visible = false;
      imbCollapseTemp.Visible = false;
      imbExpandTemp.Visible = true;
   }
}
```

You should also set the following in the page directive so that when the expand button is clicked the page doesn't go back up to the top.

```
maintainScrollPositionOnPostBack="true"
```

Now what we end up with is the following:


Untitled Page - Windows Internet Explorer
http://localhost:1037/
Live Search
File Edit View Favorites Tools Help
Untitled Page
Local intranet
100%

ALFKI :Alfreds Futterkiste
Maria Anders Phone: 030-0074321
+

ANATR :Ana Trujillo Emparedados y helados
Ana Trujillo Phone: (5) 555-4729
+

ANTON :Antonio Moreno Taquería
Antonio Moreno Phone: (5) 555-3932
+

AROUT :Around the Horn
Thomas Hardy Phone: (171) 555-7788
-

Order ID	Order Date
10355	15/11/1996
10383	16/12/1996
10453	21/02/1997
10558	4/06/1997
10707	16/10/1997
10741	14/11/1997
10743	17/11/1997
10768	8/12/1997
10793	24/12/1997
10864	2/02/1998
10920	3/03/1998
10953	16/03/1998
11016	10/04/1998

BERGS :Berglunds snabbköp
Christina Berglund Phone: 0921-12 34 65
+

BLAUS :Blauer See Delikatessen
Hanna Moos Phone: 0621-08460
+

BLONP :Blondesddsl père et fils
Frédérique Citeaux Phone: 88.60.15.31
+

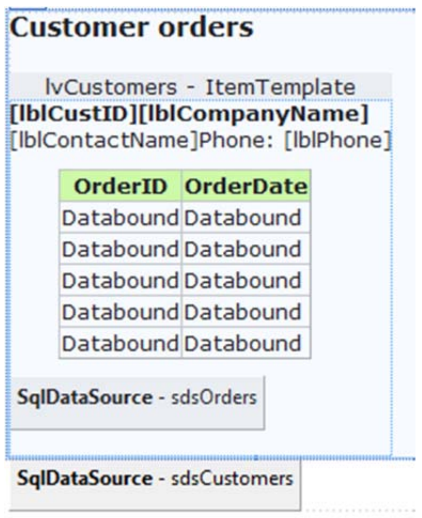
Nested listview control

You can achieve the same thing using a list view control. The code is slightly different.

Activity 4. Combined listview

In this exercise we will create a listview control to display the customers. For each customer we will display the orders they have placed in a gridview.

Aspx page:



Customer orders

lvCustomers - ItemTemplate

[lblCustID][lblCompanyName]

[lblContactName]Phone: [lblPhone]

OrderID	OrderDate
Databound	Databound
Databound	Databound
Databound	Databound
Databound	Databound
Databound	Databound

SqlDataSource - sdsOrders

SqlDataSource - sdsCustomers

Notice the sqlDataSource control for orders is inside the itemTemplate.

Code:

```
<h1>Customer orders</h1>
<asp:ListView ID="lvCustomers" runat="server" DataSourceID="sdsCustomers"
onitemdatabound="lvCustomers_ItemDataBound">
<LayoutTemplate>
        <div id="customers">
                <div>
                <asp:PlaceHolder ID="itemPlaceholder"
        runat="server"></asp:PlaceHolder>
                <br />
                </div>
        </div>
</LayoutTemplate>

<ItemTemplate>
        <asp:Label ID="lblCustID" runat="server" CssClass="bold"
        Text='<%# Eval("customerID") %>'></asp:Label>
        <asp:Label ID="lblCompanyName" runat="server" CssClass="bold"
        Text='<%# Eval("companyName") %>'></asp:Label>
        <br />
        <asp:Label ID="lblContactName" runat="server"
        Text='<%# Eval("contactname") %>'></asp:Label>
        Phone:
        <asp:Label ID="lblPhone" runat="server" Text='<%# Eval("Phone")
%>'></asp:Label>
        <br />
        <div class="orders">
                <asp:GridView ID="gvOrders" AutoGenerateColumns="False"
                runat="server" DataSourceID="sdsOrders">
                        <Columns>
                        <asp:BoundField DataField="OrderID" HeaderText="OrderID"
                        InsertVisible="False" ReadOnly="True" SortExpression="OrderID"
                        />
                        <asp:BoundField DataField="OrderDate" HeaderText="OrderDate"
                        SortExpression="OrderDate"  DataFormatString="{0:d}" />
                        </Columns>

                        <HeaderStyle CssClass="headerStyle" />

                </asp:GridView>
        </div>
        <asp:SqlDataSource ID="sdsOrders" runat="server"
        ConnectionString="<%$ ConnectionStrings:ConnectionString %>"
        SelectCommand="SELECT [OrderID], [CustomerID], [OrderDate] FROM [Orders]
        WHERE ([CustomerID] = @CustomerID)">
        <SelectParameters>
             <asp:Parameter Name="CustomerID" Type="String" />
        </SelectParameters>
        </asp:SqlDataSource>
</ItemTemplate>

</asp:ListView>
<br />
</div>
<asp:SqlDataSource ID="sdsCustomers" runat="server"
ConnectionString="<%$ ConnectionStrings:ConnectionString %>"
SelectCommand="SELECT [CustomerID], [CompanyName], [ContactName], [Phone] FROM
[Customers]">
</asp:SqlDataSource>
```

Codebehind

For this code to work you will need to use the System.Data name space

Add the following code to the item data bound event:

```
protected void lvCustomers_ItemDataBound(object sender, ListViewItemEventArgs
e)
{
        SqlDataSource sdsCustOrders;
        string custID;

        if (e.Item.ItemType == ListViewItemType.DataItem)
        {
            //find control
            sdsCustOrders = (SqlDataSource)e.Item.FindControl("sdsOrders");

            //get ID
            ListViewDataItem dataItem = (ListViewDataItem)e.Item;
            DataRowView rowView = (DataRowView)dataItem.DataItem;
            custID = rowView["customerID"].ToString();

            sdsCustOrders.SelectParameters["customerID"].DefaultValue = custID;
        }

}
```

Remember to create this event with the lighting bolt.

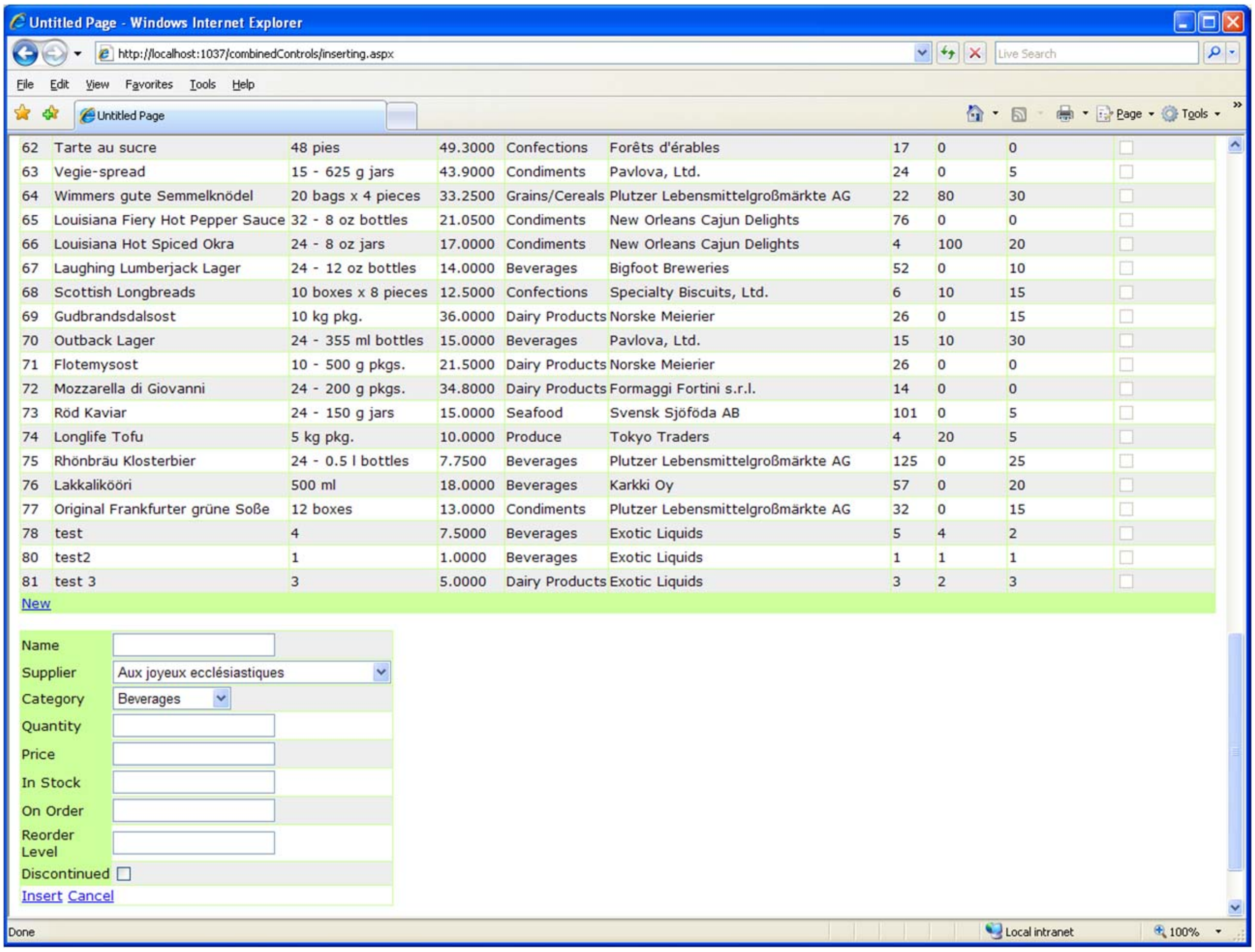
Inserting new row

You can use the details view control or form view control to insert a new row of data.

Activity 5. Inserting a new row

inserting.aspx

In this exercise we will display all the products in a gridview control. Allow the footer row to include a link button control. When the link button is clicked a details view control will be displayed, allowing the insert of new product.


Untitled Page - Windows Internet Explorer

http://localhost:1037/combinedControls/inserting.aspx

62	Tarte au sucre	48 pies	49.3000	Confections	Forêts d'érables	17	0	0	☐
63	Vegie-spread	15 - 625 g jars	43.9000	Condiments	Pavlova, Ltd.	24	0	5	☐
64	Wimmers gute Semmelknödel	20 bags x 4 pieces	33.2500	Grains/Cereals	Plutzer Lebensmittelgroßmärkte AG	22	80	30	☐
65	Louisiana Fiery Hot Pepper Sauce	32 - 8 oz bottles	21.0500	Condiments	New Orleans Cajun Delights	76	0	0	☐
66	Louisiana Hot Spiced Okra	24 - 8 oz jars	17.0000	Condiments	New Orleans Cajun Delights	4	100	20	☐
67	Laughing Lumberjack Lager	24 - 12 oz bottles	14.0000	Beverages	Bigfoot Breweries	52	0	10	☐
68	Scottish Longbreads	10 boxes x 8 pieces	12.5000	Confections	Specialty Biscuits, Ltd.	6	10	15	☐
69	Gudbrandsdalsost	10 kg pkg.	36.0000	Dairy Products	Norske Meierier	26	0	15	☐
70	Outback Lager	24 - 355 ml bottles	15.0000	Beverages	Pavlova, Ltd.	15	10	30	☐
71	Flotemysost	10 - 500 g pkgs.	21.5000	Dairy Products	Norske Meierier	26	0	0	☐
72	Mozzarella di Giovanni	24 - 200 g pkgs.	34.8000	Dairy Products	Formaggi Fortini s.r.l.	14	0	0	☐
73	Röd Kaviar	24 - 150 g jars	15.0000	Seafood	Svensk Sjöföda AB	101	0	5	☐
74	Longlife Tofu	5 kg pkg.	10.0000	Produce	Tokyo Traders	4	20	5	☐
75	Rhönbräu Klosterbier	24 - 0.5 l bottles	7.7500	Beverages	Plutzer Lebensmittelgroßmärkte AG	125	0	25	☐
76	Lakkalikööri	500 ml	18.0000	Beverages	Karkki Oy	57	0	20	☐
77	Original Frankfurter grüne Soße	12 boxes	13.0000	Condiments	Plutzer Lebensmittelgroßmärkte AG	32	0	15	☐
78	test	4	7.5000	Beverages	Exotic Liquids	5	4	2	☐
80	test2	1	1.0000	Beverages	Exotic Liquids	1	1	1	☐
81	test 3	3	5.0000	Dairy Products	Exotic Liquids	3	2	3	☐
New									

Name	
Supplier	Aux joyeux ecclésiastiques
Category	Beverages
Quantity	
Price	
In Stock	
On Order	
Reorder Level	
Discontinued	☐
Insert Cancel	

The first step is to create a new web page and add a gridview to it. The data source control needed to display the data needs to retrieve the products from the product table. As we will be inserting products the data source control also needs to have the insert, update and delete statements generated.

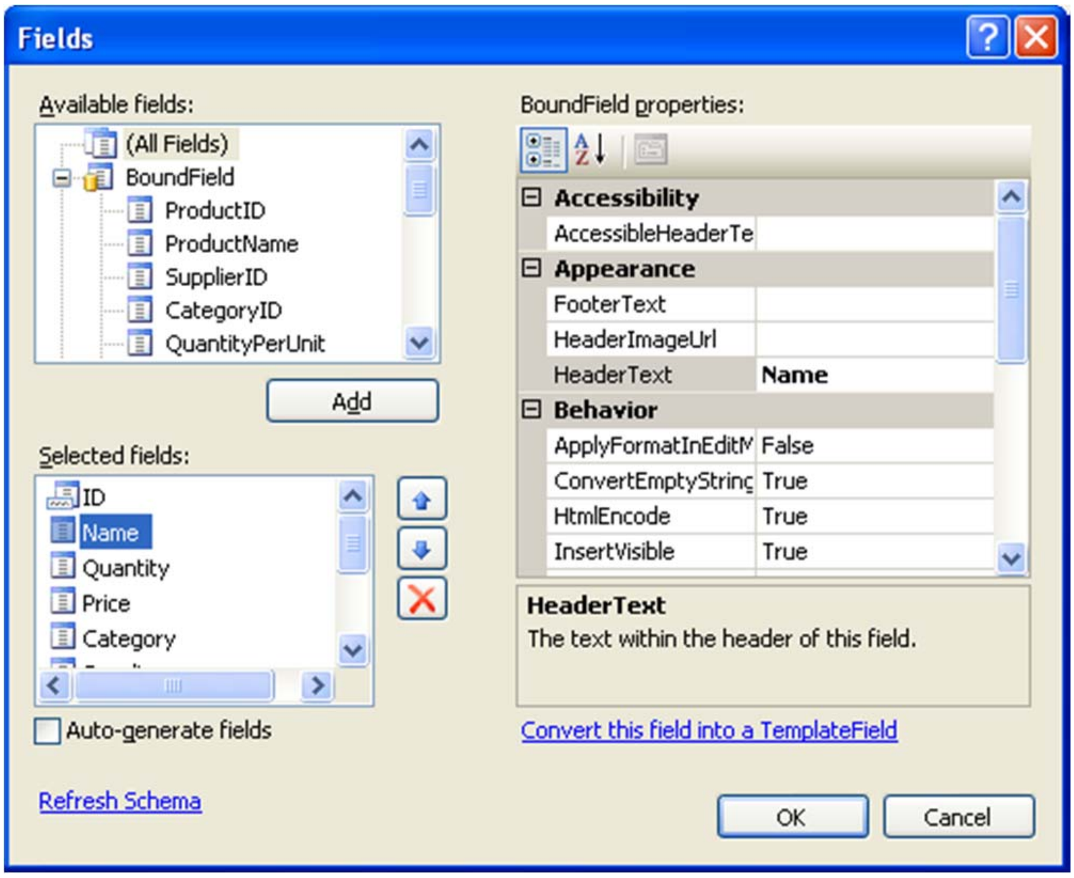
Add the data source control and set it up to retrieve everything from the products table. Configure it so that the insert, update and delete statements are also generated.

So that we display the category name and supplier company name instead of the ID go back into the data source control. Add the category name and company name to our select statement.

```
SELECT Products.ProductID, Products.ProductName, Products.SupplierID,
Products.CategoryID, Products.QuantityPerUnit, Products.UnitPrice,
Products.UnitsInStock, Products.UnitsOnOrder, Products.ReorderLevel,
Products.Discontinued, Categories.CategoryName, Suppliers.CompanyName
FROM Products INNER JOIN Categories
ON Products.CategoryID = Categories.CategoryID
INNER JOIN Suppliers ON Products.SupplierID = Suppliers.SupplierID
```

Assign the data source control to the gridview.

To be able to have the New link appear in the footer we need to have at least one column as a template column. Make the product ID a template column by clicking on “Convert this field into a template field”.


Fields

Available fields:

- (All Fields)
- BoundField
  - ProductID
  - ProductName
  - SupplierID
  - CategoryID
  - QuantityPerUnit

Selected fields:

- ID
- Name
- Quantity
- Price
- Category

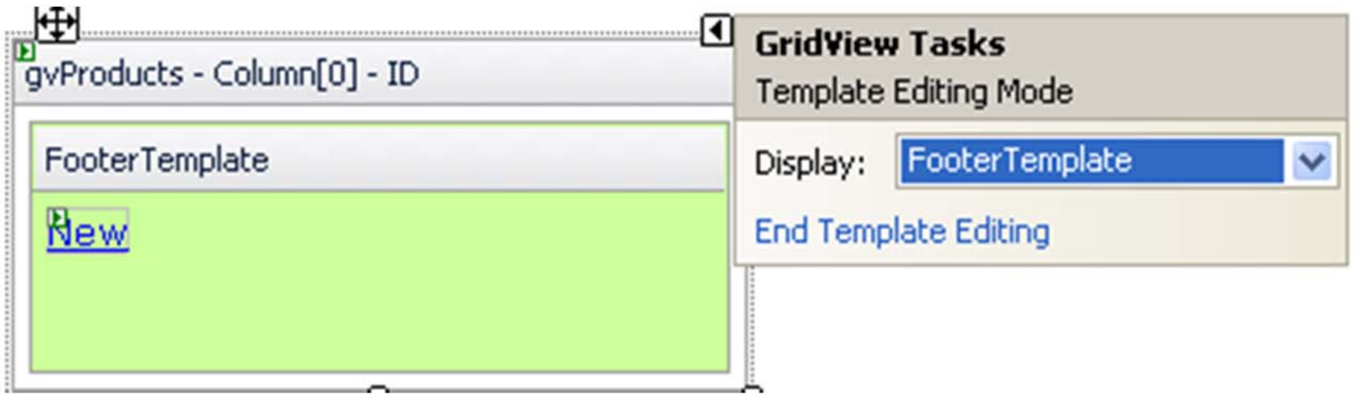
Auto-generate fields

BoundField properties:

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

HeaderText
The text within the header of this field.

You can now edit templates by selecting edit templates from the gridview smart tag.


gvProducts - Column[0] - ID

FooterTemplate

New

GridView Tasks
Template Editing Mode

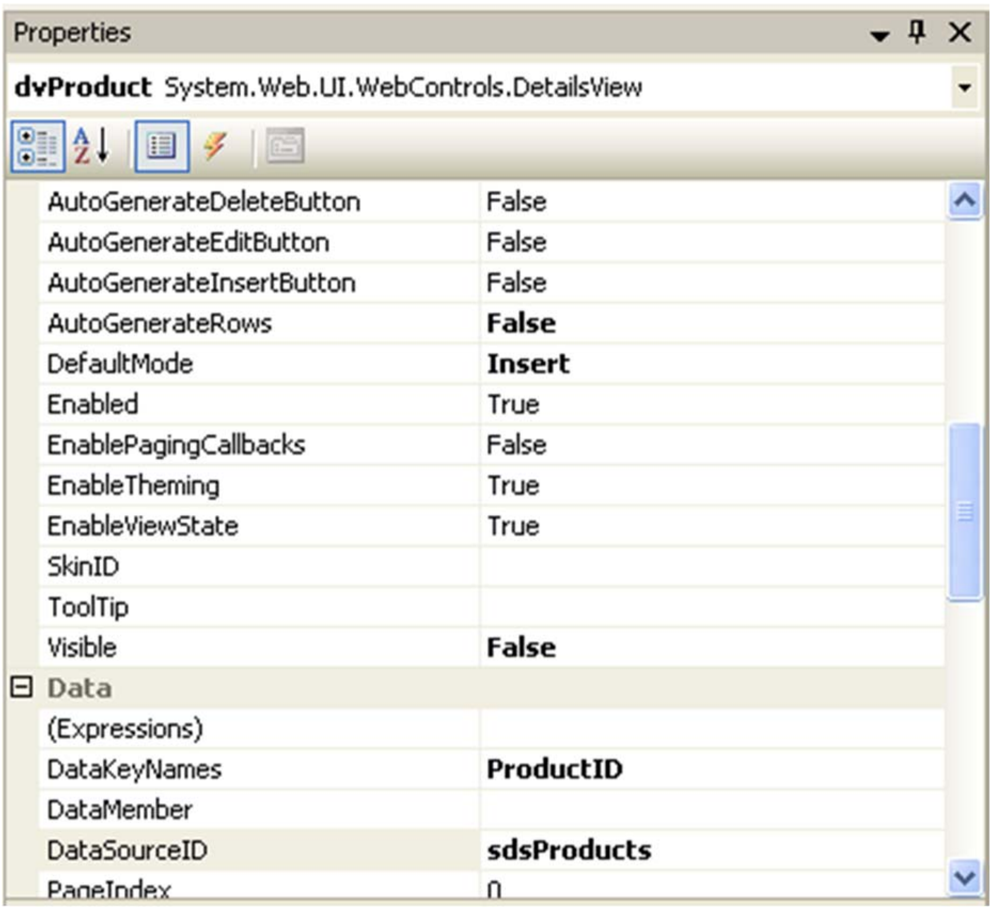
Display: FooterTemplate

Add a link button control to the footer template.

Set the “showFooter” property to true for the grid view.

Now we need to create the details view control that will be used to insert new data. This details view control will have the same data source control as the gridview.

As we are going to be using the details view control for insert only the defaultMode property needs to be set to insert. We will also set the visible property to false so that it will only be displayed after the user clicks on the “New” link.

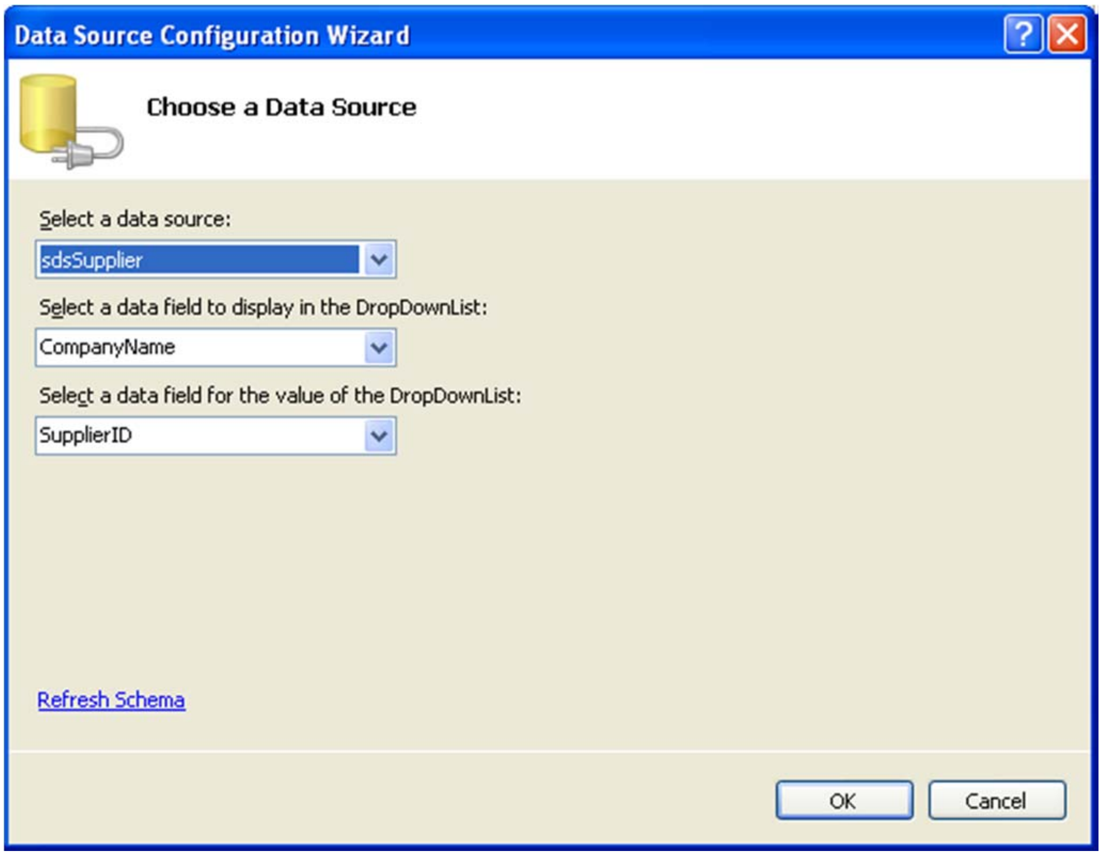

Properties

dvProduct System.Web.UI.WebControls.DetailsView

AutoGenerateDeleteButton	False
AutoGenerateEditButton	False
AutoGenerateInsertButton	False
AutoGenerateRows	**False**
DefaultMode	**Insert**
Enabled	True
EnablePagingCallbacks	False
EnableTheming	True
EnableViewState	True
SkinID	
ToolTip	
Visible	**False**
Data	
(Expressions)	
DataKeyNames	**ProductID**
DataMember	
DataSourceID	**sdsProducts**
PageIndex	0

The category and supplier should be displayed as drop down lists instead of textboxes. To display these fields as drop down lists we need to convert them to template fields. You can do this by selecting “Convert this field into a templateField”. Once the fields have been converted to template fields you can access the insertItemTemplate, remove the textbox and add a drop down list. Each drop down list will need to get its data from a data source control. Add 2 data source controls the first one will select the category ID and category name from the categories table and the second will select the supplier ID and company name from the suppliers table.

You can then set the data source for the drop down lists.


Data Source Configuration Wizard

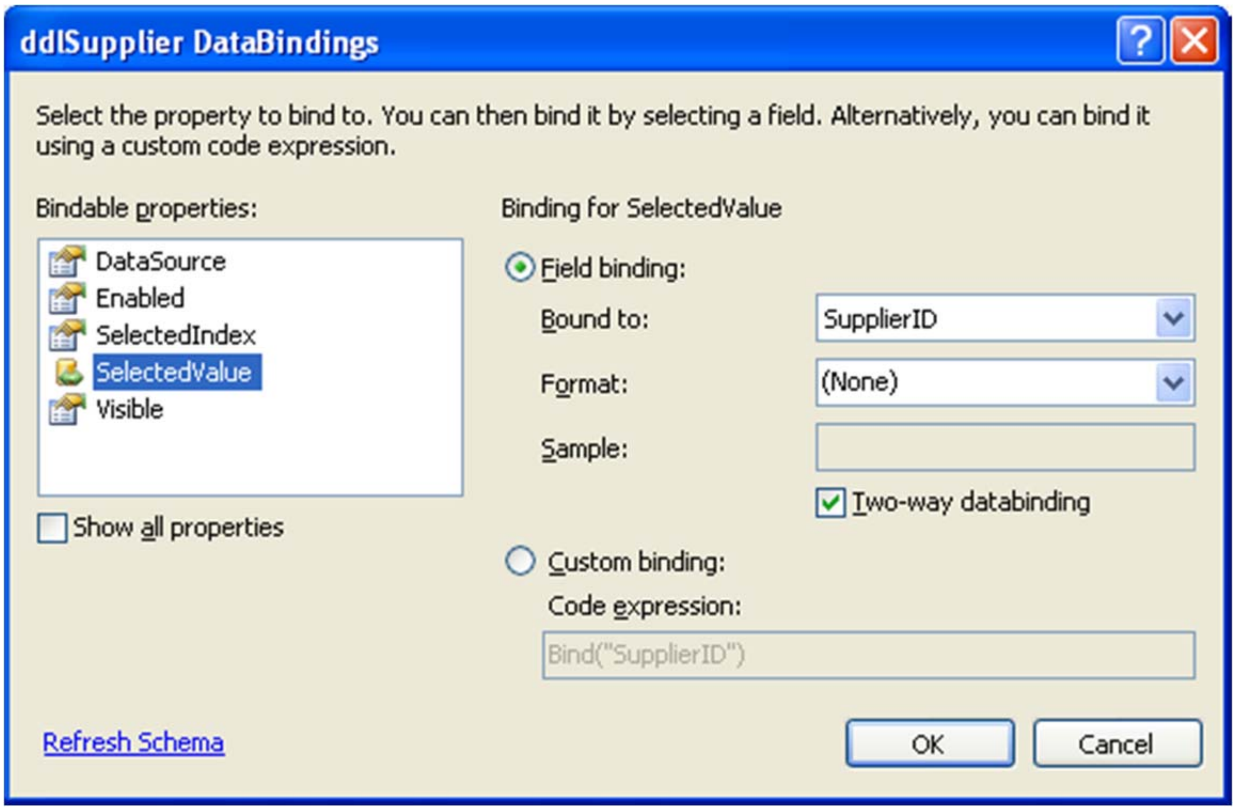
Choose a Data Source

Select a data source: sdsSupplier

Select a data field to display in the DropDownList: CompanyName

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

You also need to set the data binding by selecting edit data binding from the drop down list smart tag:


ddlSupplier 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 (selected), Visible

Show all properties: [ ]

Binding for SelectedValue

Field binding: [✓]

Bound to: SupplierID

Format: (None)

Sample:

Two-way databinding: [✓]

Custom binding: [ ]

Code expression: Bind("SupplierID")

This needs to be done for the category and the supplier.

This is the resulting source code:

```
<asp:GridView ID="gvProducts" runat="server" AutoGenerateColumns="False"
CssClass="productTable"
DataKeyNames="ProductID" DataSourceID="sdsProducts" GridLines="None"
ShowFooter="True">
    <FooterStyle CssClass="footerStyle" />
     <Columns>
```

```
            <asp:TemplateField HeaderText="ID" InsertVisible="False"
            SortExpression="ProductID">
                  <EditItemTemplate>
                         <asp:Label ID="Label1" runat="server"
                      Text='<%# Eval("ProductID") %>'></asp:Label>
                      </EditItemTemplate>
                  <FooterTemplate>
                      <asp:LinkButton ID="lbtNew"
                      runat="server">New</asp:LinkButton>
                  </FooterTemplate>
                  <ItemTemplate>
                      <asp:Label ID="Label1" runat="server"
                      Text='<%# Bind("ProductID") %>'></asp:Label>
                  </ItemTemplate>
          </asp:TemplateField>
          <asp:BoundField DataField="ProductName" HeaderText="Name"
          SortExpression="ProductName" />
          <asp:BoundField DataField="QuantityPerUnit"
          HeaderText="Quantity" SortExpression="QuantityPerUnit" />
          <asp:BoundField DataField="UnitPrice" HeaderText="Price"
          SortExpression="UnitPrice" />
          <asp:BoundField DataField="CategoryName"
          HeaderText="Category" SortExpression="CategoryName" />
          <asp:BoundField DataField="CompanyName"
          HeaderText="Supplier" SortExpression="CompanyName" />
          <asp:BoundField DataField="UnitsInStock" HeaderText="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>
    <HeaderStyle CssClass="headerStyle" />
    <AlternatingRowStyle CssClass="productAltStyle" />
</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 Products.ProductID, Products.ProductName,
Products.SupplierID, Products.CategoryID, Products.QuantityPerUnit,
Products.UnitPrice, Products.UnitsInStock, Products.UnitsOnOrder,
Products.ReorderLevel, Products.Discontinued, Categories.CategoryName,
Suppliers.CompanyName FROM Products INNER JOIN Categories ON
Products.CategoryID = Categories.CategoryID INNER JOIN Suppliers ON
Products.SupplierID = Suppliers.SupplierID"
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>
<br />
<asp:DetailsView ID="dvProduct" runat="server" AutoGenerateRows="False"
CssClass="productTable"
DataKeyNames="ProductID" DataSourceID="sdsProducts" DefaultMode="Insert"
Height="50px"
Visible="False" Width="125px">
    <Fields>
        <asp:BoundField DataField="ProductID" HeaderText="ID"
        InsertVisible="False" ReadOnly="True"
        SortExpression="ProductID" />
        <asp:BoundField DataField="ProductName" HeaderText="Name"
        SortExpression="ProductName" />
        <asp:TemplateField HeaderText="Supplier"
        SortExpression="SupplierID">
        <EditItemTemplate>
            <asp:TextBox ID="TextBox2" runat="server"
            Text='<%# Bind("SupplierID") %>'></asp:TextBox>
        </EditItemTemplate>
        <InsertItemTemplate>
            <asp:DropDownList ID="ddlSupplier" runat="server"
            DataSourceID="sdsSupplier" DataTextField="CompanyName"
            DataValueField="SupplierID"
            SelectedValue='<%# Bind("SupplierID") %>'>
            </asp:DropDownList>
        </InsertItemTemplate>
        <ItemTemplate>
            <asp:Label ID="Label2" runat="server"
            Text='<%# Bind("SupplierID") %>'></asp:Label>
        </ItemTemplate>
        </asp:TemplateField>
        <asp:TemplateField HeaderText="Category"
        SortExpression="CategoryID">
        <EditItemTemplate>
            <asp:TextBox ID="TextBox1" runat="server"
            Text='<%# Bind("CategoryID") %>'></asp:TextBox>
```

```
          </EditItemTemplate>
          <InsertItemTemplate>
              <asp:DropDownList ID="ddlCategory" runat="server"
              DataSourceID="sdsCategory" DataTextField="CategoryName"
              DataValueField="CategoryID"
              SelectedValue='<%# Bind("CategoryID") %>'>
              </asp:DropDownList>
          </InsertItemTemplate>
          <ItemTemplate>
              <asp:Label ID="Label1" runat="server"
              Text='<%# Bind("CategoryID") %>'></asp:Label>
          </ItemTemplate>
          </asp:TemplateField>
          <asp:BoundField DataField="QuantityPerUnit"
          HeaderText="Quantity" SortExpression="QuantityPerUnit" />
          <asp:BoundField DataField="UnitPrice" HeaderText="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" />
          <asp:CommandField ShowInsertButton="True" />
     </Fields>
     <FieldHeaderStyle CssClass="headerStyle" />
     <AlternatingRowStyle CssClass="productAltStyle" />
</asp:DetailsView>
<br />
<asp:Label ID="lblInsert" runat="server"></asp:Label><br />
<br />
<asp:SqlDataSource ID="sdsSupplier" runat="server" ConnectionString="<%$
ConnectionStrings:ConnectionString %>"
SelectCommand="SELECT [SupplierID], [CompanyName] FROM
[Suppliers]"></asp:SqlDataSource>
<br />
<asp:SqlDataSource ID="sdsCategory" runat="server" ConnectionString="<%$
ConnectionStrings:ConnectionString %>"
SelectCommand="SELECT [CategoryID], [CategoryName] FROM
[Categories]"></asp:SqlDataSource>
<br />
```

Code needs to be added to the code behind file so that the details view control is displayed when the “New” link is clicked and focus set to the details view control

```
protected void gvProducts_RowCommand(object sender, GridViewCommandEventArgs e)
{
        //show the details view control
        dvProduct.Visible = true;

        //set focus to the details view
        dvProduct.Focus();
 }
```

When the insert or cancel link is clicked in the details view control it should be hiden from the page.

```
protected void dvProduct_ItemCommand(object sender, DetailsViewCommandEventArgs
e)
{
        //hide the details view
        dvProduct.Visible = false;
}
```
After the product has been inserted a message indicating the insert worked is displayed to the screen.
```
protected void dvProduct_ItemInserted(object sender,
DetailsViewInsertedEventArgs e)
{
        //display insert message
        lblInsert.Text = "Product Inserted";
}
```
Displaying empty data

Sometimes we have situations where there is no data to be displayed. For example if we are looking up a customer's previous orders and they have never placed an order. We have already seen how to display text for an empty data template. We can add any ASP.NET or HTML control to an empty data template even a details view or form view control.

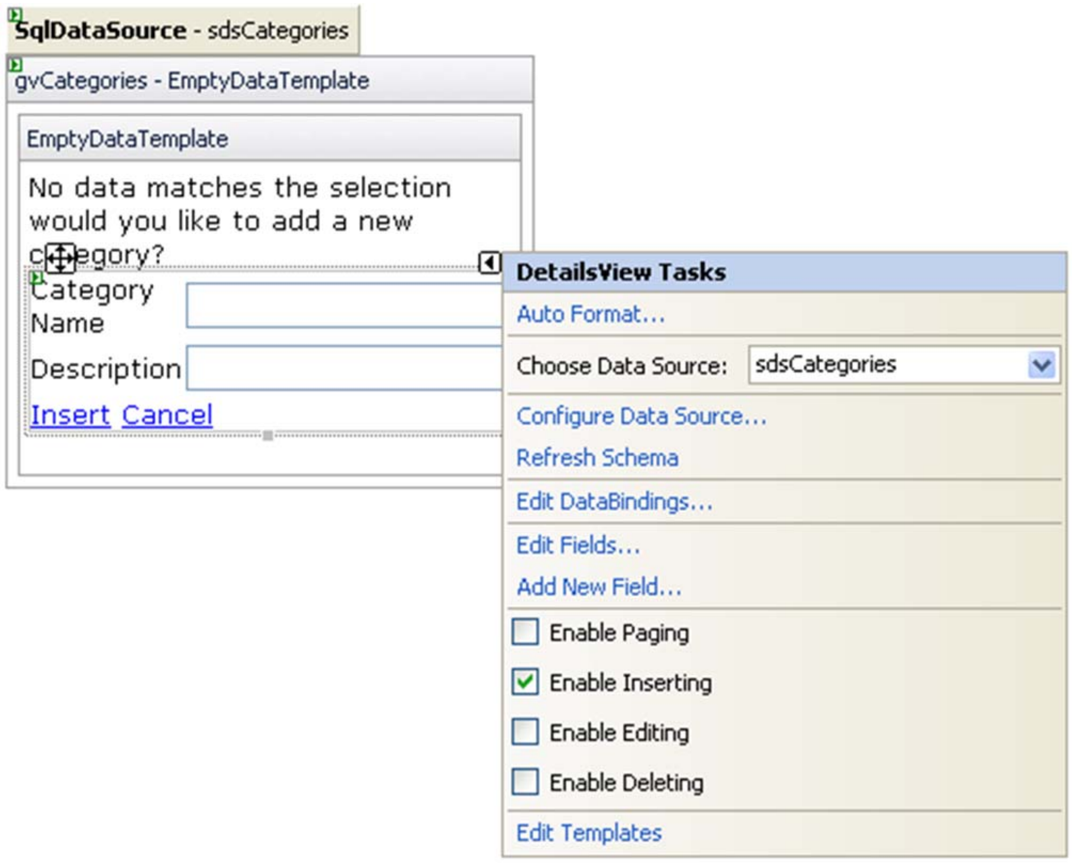
Activity 6. Empty data

Empty data.aspx

In this exercise we will display a grid view control. This gridview control is linked to a data source control which selects the category ID, name and description from the categories table. Make sure the insert, update and delete statements are also generated.

So that no data is returned set up the SQL statement to retrieve this data where category ID = 100.

Select edit templates for the gridview control and edit the empty data template. Add a details view control and allocate the same data source control to it.



SqlDataSource - sdsCategories

gvCategories - EmptyDataTemplate

EmptyDataTemplate

No data matches the selection would you like to add a new category?

Category Name

Description

Insert Cancel

DetailsView Tasks

Auto Format...

Choose Data Source: sdsCategories

Configure Data Source...

Refresh Schema

Edit DataBindings...

Edit Fields...

Add New Field...

☐ Enable Paging

☑ Enable Inserting

☐ Enable Editing

☐ Enable Deleting

Edit Templates

Set the defaultMode to insert for the details view control. When the page is viewed in the browser the details view control will be displayed as there are no categories that have an id of 100.

References

ASP.NET 2.0 Unleashed

www.asp.net/learn/dataaccess/default.aspx?tabid=63