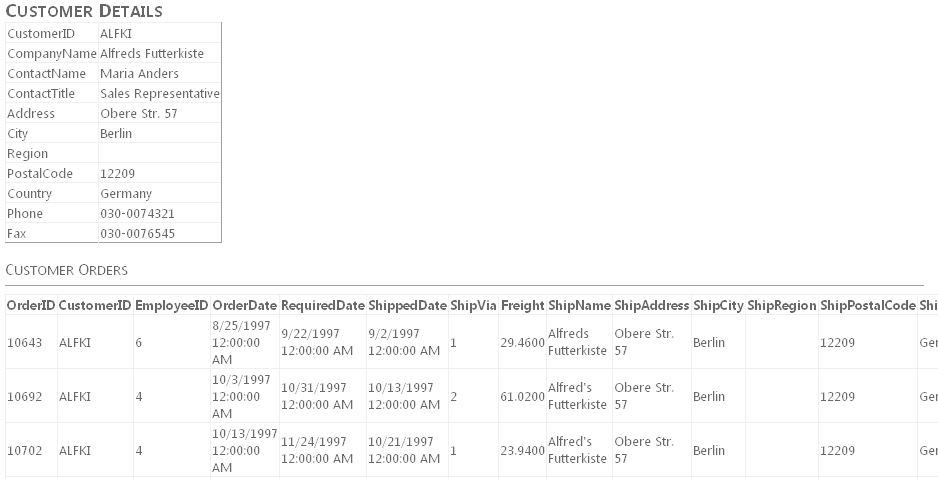


Select a view order and the orders page is shown



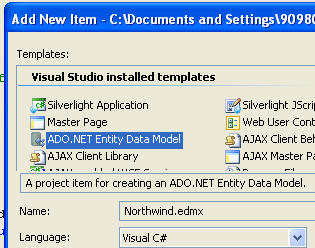
**Exercise 1 Creating the Website and Defining the Entity Model**

In this exercise, you create a new website and define the Entity Model.

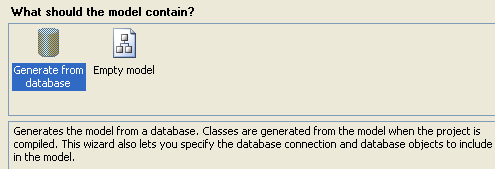
1. Create a new website

2. Add the northwnd.mdf file to your App\_Data directory.

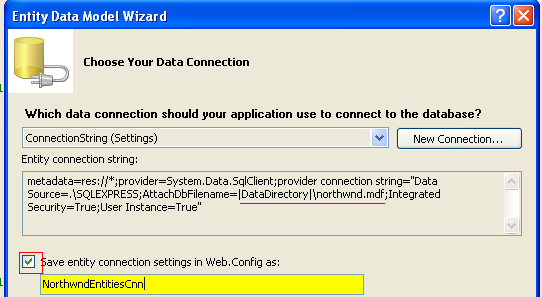
3. Add a new ADO.NET Entity Data Model to your website. Name this model **Northwind.edmx**. When prompted, allow Visual Studio to add the model to the App\_Code directory.



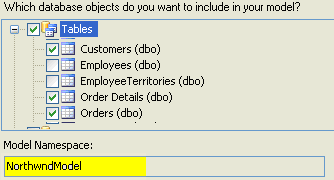
4. Using the Entity Data Model Wizard, select Generate from database on the first step and click Next.



5. On the next page, select the northwnd.mdf data file. Make sure to select the check box to save the connection string in the Web.config file. Name this connection string **NorthwndEntitiesCnn** and click Next.



6. On the next page, select the Customers, Order Details, and Orders tables. Make sure that both check boxes are selected, and set the Model Namespace to **NorthwndModel**.

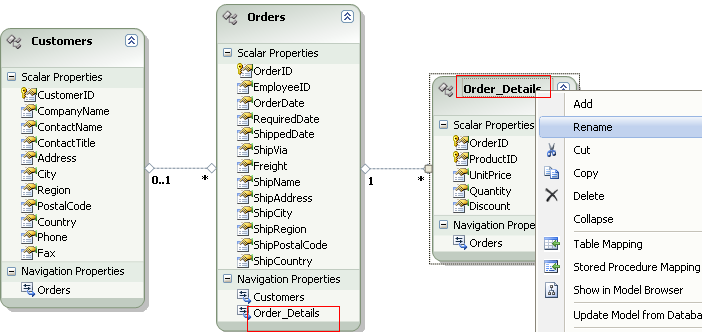


7. Rename items in the model to better identify them.

- Select the Order\_Detail entity in the model, right-click it, and choose Rename.

- Rename the entity to **OrderDetail**.

- Then select the Order\_Details navigation property from the Order entity and rename it to **OrderDetails**.



8. Save and close the model.

**Exercise 2 Binding to the EntityDataSource Control**

In this exercise, you create a webpage that defines an EntityDataSource that will be used to work with the entity model created in the previous exercise.

1. Continue editing the project you created in the previous exercise.

2. Open the Default.aspx page. Set the title to **Customers.**

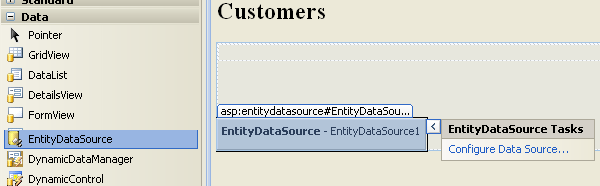
<head runat="server">

<title></title>

<h2>Customers</h2>

</head>

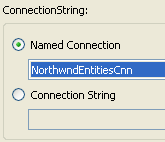
3. In Design view, drag an EntityDataSource control onto the page from the Data tab of the Toolbox. This control will be used to expose a list of customers in the database.



4. Click the smart tag in the upper-right corner of the EntityDataSource control to open the EntityDataSource Tasks list.

5. Select Configure Data Source to open the Configure Data Source Wizard.

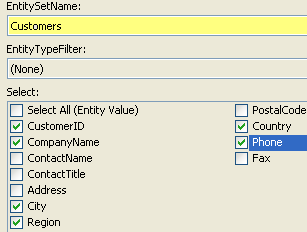
a. On the first page of the wizard, from the Named Connection list, select NorthwndEntitiesCnn.



b. On the second page of the wizard,

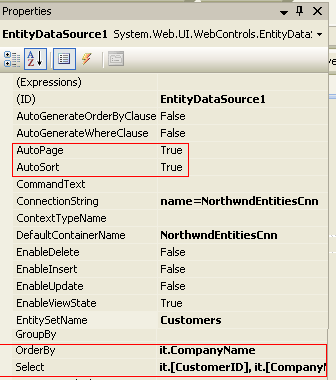
- Select the Customers EntitySetName.

- Set the Select fields to CustomerID, CompanyName, City, Region, Country, and Phone.



c. Finish the wizard, and switch to Source view for your page. Edit the EntityDataSource markup to include

* AutoPage
* AutoSort
* OrderBy statement to order the results by CompanyName ( it.CompanyName )



<asp:EntityDataSource ID="EntityDataSource1" runat="server"

ConnectionString="name=NorthwndEntitiesCnn"

DefaultContainerName="NorthwndEntitiesCnn"

EnableFlattening="False"

EntitySetName="Customers"

AutoPage="true"

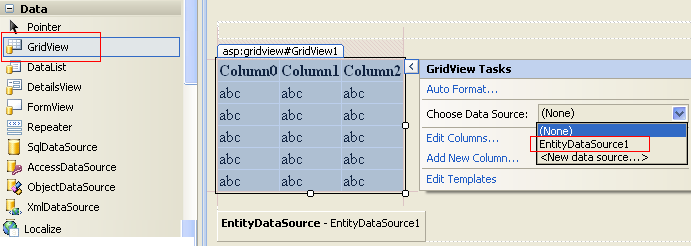
AutoSort="true

OrderBy="it.CompanyName"

Select="it.[CustomerID], it.[CompanyName], it.[City], it.[Region], it.[Country], it.[Phone]">

</asp:EntityDataSource>

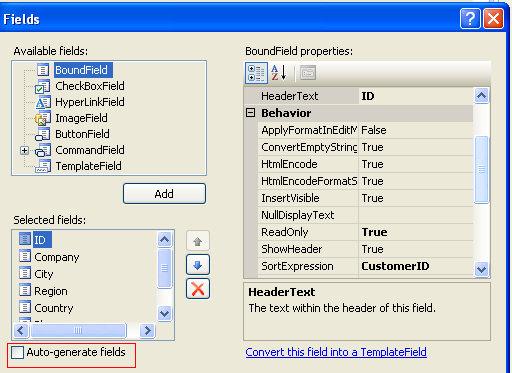
6. Add a GridView control to the page. Set the DataSourceID property to point to the EntityDataSource created previously.



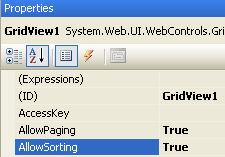
7. Define bound columns and change the HeaderText for each field in the result set.

|  |  |  |
| --- | --- | --- |
| **DataField** | **HeaderText** | **SortExpression** |
| CustomerID | ID | CustomerID |
| CompanyName | Company | CompanyName |
| City | City | City |
| Region | Region | Region |
| Country | Country | Country |
| Phone | Phone | Phone |
|  |  |  |

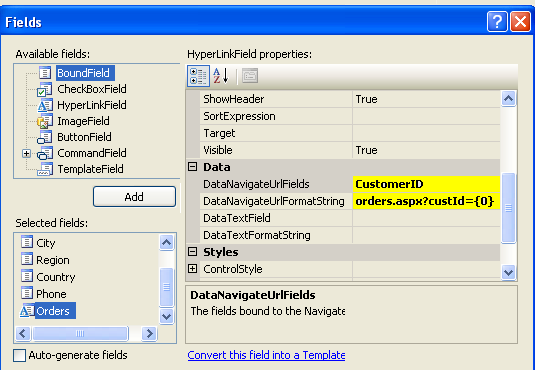




8. Enable AllowPaging and AllowSorting.



9. Add a HyperLinkField to the GridView to call **orders.aspx** and pass the **customer ID** as a query string parameter.



Your GridView markup should look as follows.

<asp:GridView ID="GridViewCustomers" runat="server" AllowPaging="True"

AllowSorting="True" AutoGenerateColumns="False"

DataSourceID="EntityDataSource1">

<Columns>

<asp:BoundField DataField="CustomerID" HeaderText="**ID**" ReadOnly="**True**" SortExpression="**CustomerID**" />

<asp:BoundField DataField="CompanyName" HeaderText="**Company**" ReadOnly="True" SortExpression="CompanyName" />

<asp:BoundField DataField="City" HeaderText="**City**" ReadOnly="True" SortExpression="City" />

<asp:BoundField DataField="Region" HeaderText="**Region**" ReadOnly="True" SortExpression="Region" />

<asp:BoundField DataField="Country" HeaderText="**Country**" ReadOnly="True" SortExpression="Country" />

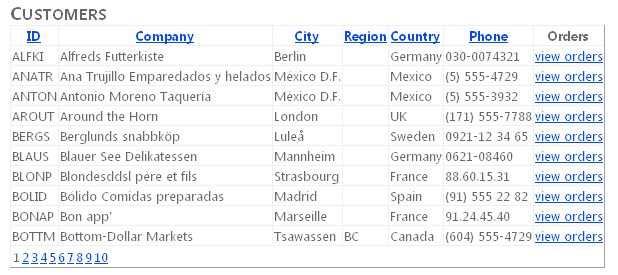
<asp:BoundField DataField="Phone" HeaderText="**Phone**" ReadOnly="True" SortExpression="Phone" />

<asp:HyperLinkField DataNavigateUrlFields="**CustomerID**" DataNavigateUrlFormatString="**orders.aspx?custId={0}"**

HeaderText="**Orders**" Text="**view orders**" />

</Columns>

</asp:GridView>



**B) CREATE ORDER PAGE**

10. Create the ORDERS PAGE Add an Orders.aspx page to your site.

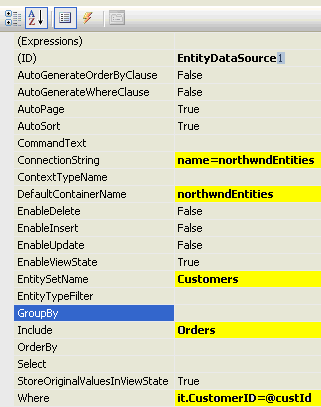
11. Add an **EntityDataSource** control to the page.



12. In properties

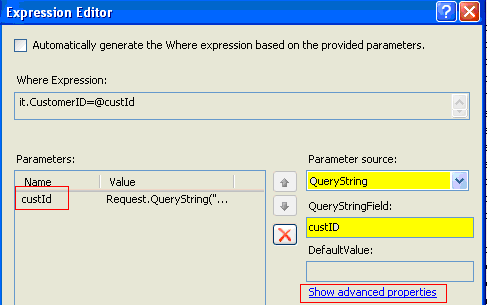
- Set EntitySetName to Customers

- Set Include to Orders

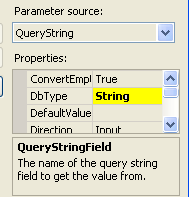


13. Define a Where clause type [it.CustomerID=@custId](mailto:it.CustomerID=@custId)

14. Click the button and set the parameter. Select a customer by a specific ID. Set the ID as a QueryStringParameter.



15. Click Show advanced properties and select String



The following markup shows an example.

<asp:EntityDataSource ID="EntityDataSource1" runat="server"

ConnectionString="name=NorthwndEntitiesCnn"

DefaultContainerName="NorthwndEntitiesCnn"

EntitySetName="Customers"

**EnableFlattening="False"**

Include="Orders"

Where="it.CustomerID=@custId">

<WhereParameters>

<asp:QueryStringParameter

DbType="String"

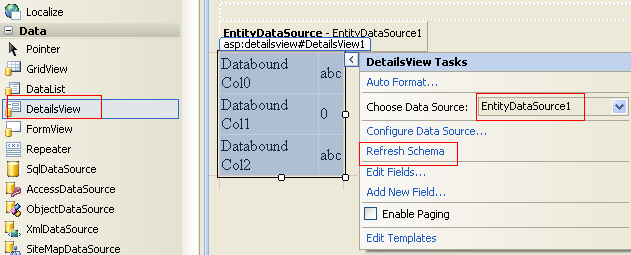
Name="custId"

QueryStringField="custID" />

</WhereParameters>

</asp:EntityDataSource>

11. Next, add a DetailsView control to the page and name it **DetailsViewCust**. In Design view, use the smart tag to set the data source to the EntityDataSource set previously.



Select Refresh Schema from the same smart tag task list to generate columns bound to the data source.

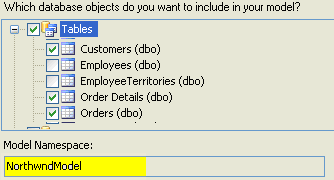
**C) BOUND UPPER DETAIL VIEW TO THE GRID VIEW**

12. Add a GridView control under the DetailsView and name it **GridViewOrders**.

This will be used to show custom orders. You cannot, at present, bind this to the included collections of the EntityDataSource by using markup. Instead, you must write some code.

13. Add an event handler for the **DetailsViewCust** DataBound event. This event fires when the DetailsView control has been bound to data. You can use it to pull the bound Customer from the control and use its Orders collection for the GridView control. Your code should read as follows.

Use the Model Namespace define in this step: (6)



protected void **DetailsViewCust\_DataBound**(object sender, EventArgs e)

{

**NorthwndModel**.**Customers** cust = (**NorthwndMode**l.**Customers**)DetailsViewCust.DataItem;

if (cust != null)

{

this.GridViewOrders.DataSource = cust.Orders;

this.GridViewOrders.DataBind();

}

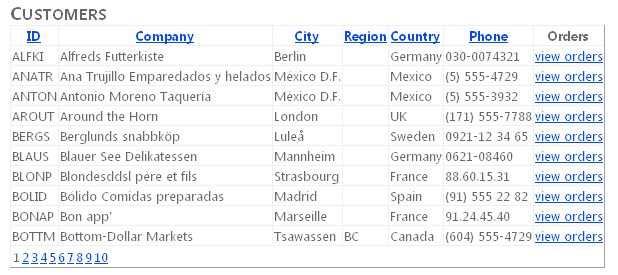
}

If the following message appears when doing the following instruction

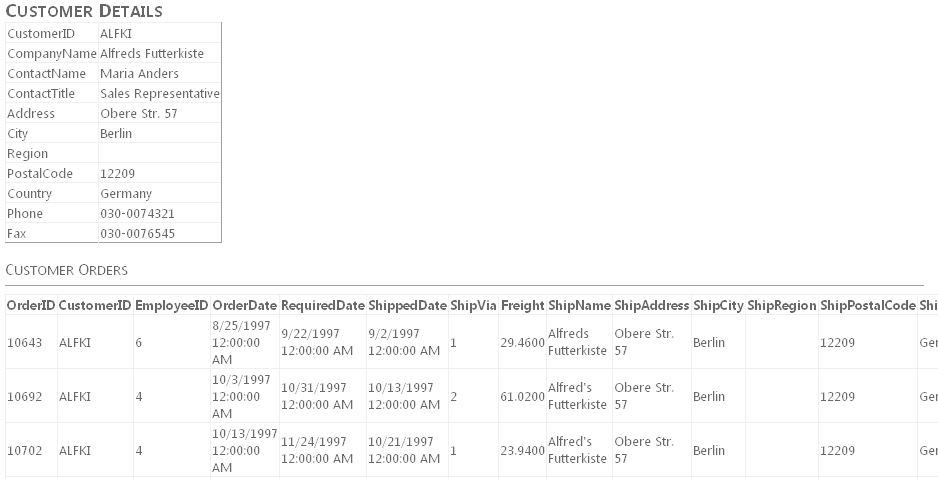
**NorthwndModel**.**Customers** cust = (**NorthwndMode**l.**Customers**)DetailsViewCust.DataItem;

Unable to cast object of type 'System.Web.UI.WebControls.EntityDataSourceWrapper' to type 'NorthwndModel.Customers'.

In the You need to include EnableFlattening="False" In the EntityDataSource, but this is only possible in VS 2010, not in VS 2008

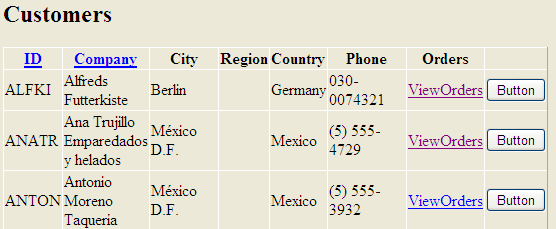


Select a view order and the orders page is shown



**D) IN ONE PAGE**

If you want to include everything in the first page



1. Include the button

<asp:ButtonField runat="server"

Text="Button"

CommandName="DetalleOrdenes"

ButtonType="Button" />

2. Include the code

protected void **GridView1\_RowCommand**(object sender, GridViewCommandEventArgs e)

{

if (e.CommandName == "DetalleOrdenes")

{

int currentRow = int.Parse(e.CommandArgument.ToString());

string custId = GridView1.Rows[currentRow].Cells[0].Text.ToString();

string strSQL = "select \* from orders where custId ="+ custId;

SqlConnection conn = new SqlConnection(System.Configuration.ConfigurationManager.AppSettings["conString"]);

SqlCommand cmd = new SqlCommand(strSQL, conn);

SqlDataAdapter SqlAdap = new SqlDataAdapter(cmd);

System.Data.DataTable dt = new System.Data.DataTable("Data1");

SqlAdap.Fill(dt);

GridViewOrders.DataSource = dt;

GridViewOrders.DataBind();

}

}

Otra alternative mas sencilla es con EntityDataSource

protected void **GridView1\_RowCommand**(object sender, GridViewCommandEventArgs e)

{

if (e.CommandName == "DetalleOrdenes")

{

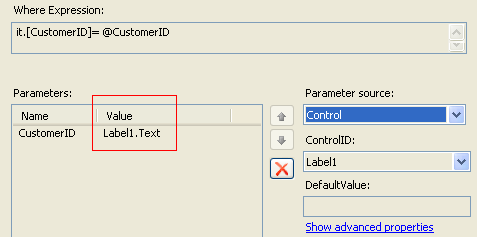
int currentRow = int.Parse(e.CommandArgument.ToString());

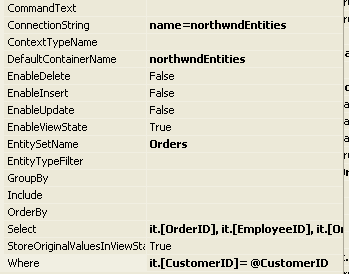
Label1.Text = GridView1.Rows[currentRow].Cells[0].Text.ToString();

}

}

Al momento de actualizar Label1 se carga los datos si se le asigna un **EntityDataSource** al nuevo **GridView** y se le asigna el Where que se dispare al momento de actualizar el Control **Label1.**





Pero la mejor alternativa es con una variable session **Session[“CustomerID”]**