RadControls for Winforms

RadControls for Winforms

Table of Contents

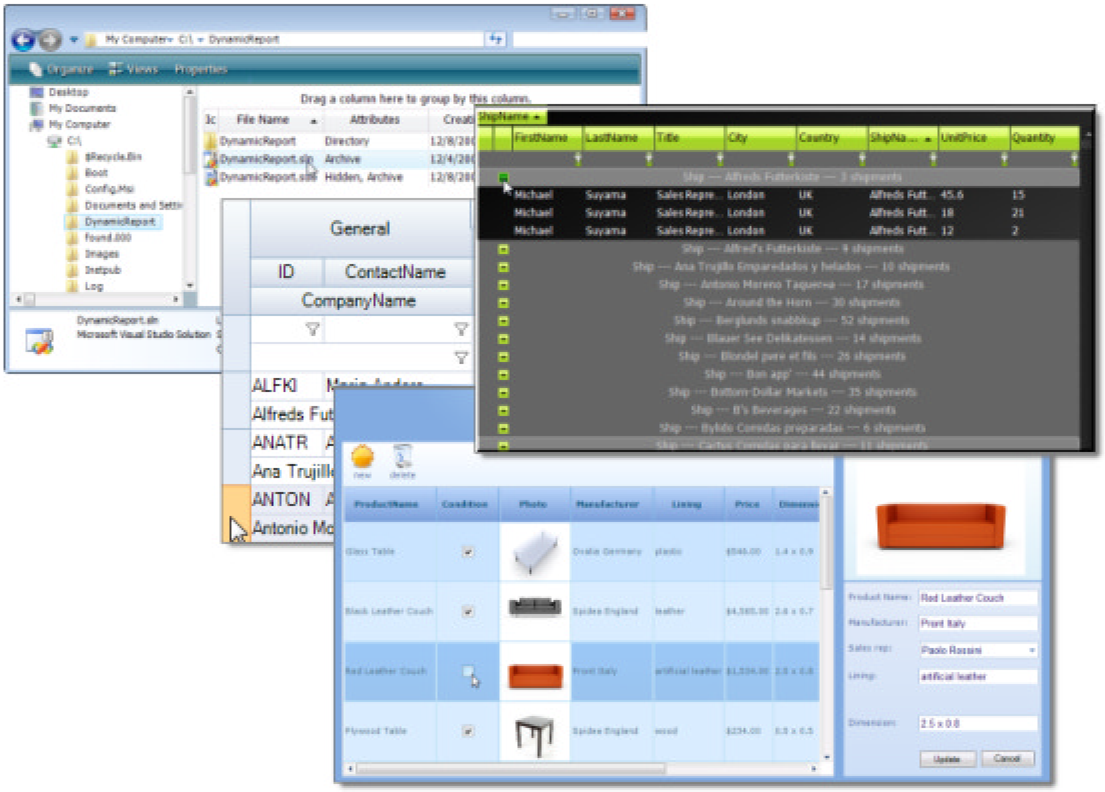
1 Grid

1.1 Objectives

- Get familiar with the RadGridView control by binding to a data source and configuring the grid using the Property Builder.
- Learn how to add columns for any data type at design-time and using code.
- Learn how to group, filter and sort data based on user input at runtime, design-time configuration and using code.
- Display hierarchical data from multiple related tables in the grid.
- Get low-level programmatic control over the RadGridView update process using Virtual Mode.
- Learn to use RadGridView with LINQ data sources to implement dynamic paging, sorting, and filtering on large datasets.
- Export RadGridView data to Excel or to the Telerik Reporting engine.

1.2 Introduction

RadGridView is a powerful, highly performant grid component from Telerik developed on top of the innovative Telerik Presentation Framework, which allows for unprecedented combination of performance, extensibility, customizability, and ease of use.



Some of the key features of RadGridView are:

- **Hierarchical data presentation** - RadGridView has the ability to represent hierarchical master-detail data. Its hierarchical schema could be set up either at design-time, at runtime using the control API, or handled automatically for you based on the structure of the data.

- Easily customizable theming mechanism - RadGridView is shipped with a consistent set of themes that allow you to easily build slick interfaces. Or you can use the codeless visual approach to build a new custom theme.
- **Grouping** - RadGridView allows easy implementation of multilevel grouping of data from a single table. Simply drag the column header(s) to the group panel on the top to define the grouping order and hierarchy. You can also programmatically group data using group-by expressions. Another unique feature is the ability to sort grouped data by single or multiple columns. Grouping programmatically also allows you to perform aggregate operations (e.g. sum, min, max, count, first and last) and to output custom formatted strings into the group heading.
- **Multi-column sorting** - in addition to the simple one-column sorting RadGridView allows you to sort data by several columns just like in Microsoft Excel. With the help of sorting expressions, you can also do custom sorting.
- **Filtering** - RadGridView can perform filtering operations for all columns that support filtering. For each column there will be a dropdown menu with the available filter expressions
- **Column resizing** - RadGridView supports convenient column and row resizing with features like real-time resizing, best fit sizing, resizing of the grid on column/row resizing, clipping of the cell content on column resizing
- **Column reordering with drag-and-drop** - column reordering is a nice interface feature which lets users reorder columns, based on their personal preference. Telerik RadGridView allows users to quickly reorder columns at runtime by simply drag-and-dropping their headers, with visual indication of the header being dragged.
- **Keyboard navigation** - RadGridView can be navigated using the keyboard. You can focus on a grid with the TAB key, navigate through the rows, and edit cells.
- **Rich set of column types** - RadGridView supports the most commonly used column types to provide editing functionality.
- **Pinned rows and Pinned columns support** - RadGridView enhances the simple scrolling by supporting pinned rows and columns. You can scroll the grid data, while pinned rows stay visible and next to the header row; pinned columns stay visible and on the left of the grid.
- **DataBinding to LINQ data sources, business objects, nullable objects and properties of sub-objects** - You can use a wide variety of data-sources for grid structure generation (the only requirement is that these custom objects implement the ITypedList/IEnumarable/ICustomTypeDescriptor interfaces). Furthermore, RadGridView supports out-of-the-box binding to LINQ data sources, sub-objects, nullable and business objects.
- **Conditional formatting** - RadGridView enables you to apply conditional formatting to grid elements for enhanced readability and usability of the displayed data. Conditional formatting can be applied programmatically or by the user at run-time.
- **Context menu support** - The context menu provides extra usability and richness to any application. The default RadGridView context menu provides support for sorting, formatting, grouping, column selection and automatic column width adjustment. You can extend the context menu to add your own menu items and display your menus conditionally.

1.3 Getting Started

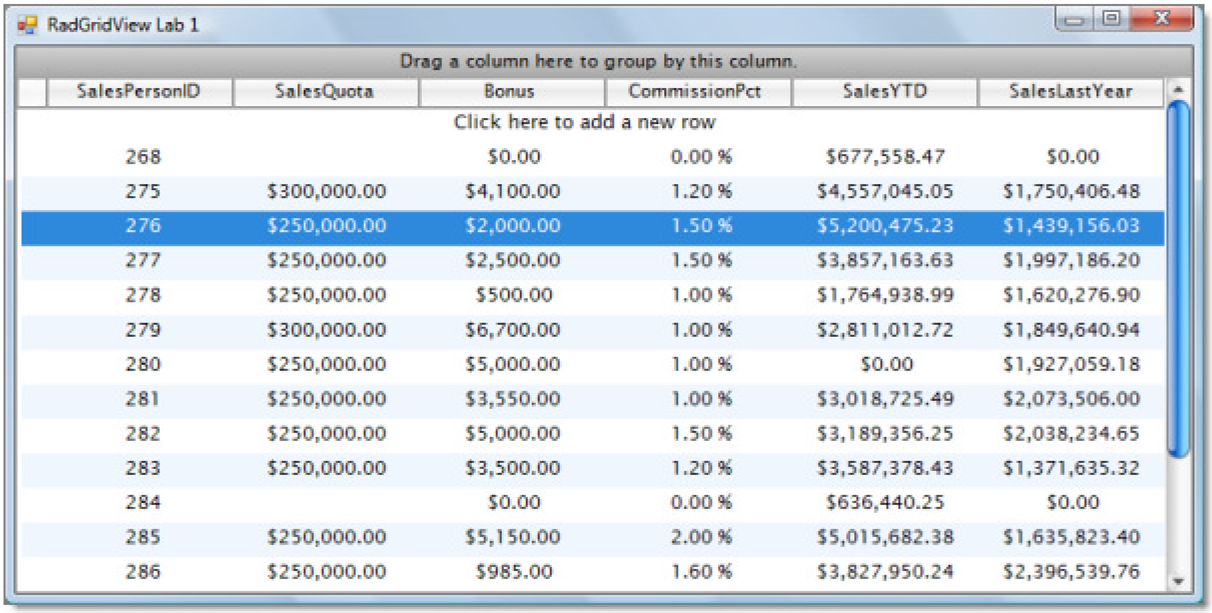
RadGridView control’s many features include flexible setup during design time, quick and easy binding to database data, using autogenerated columns from the DataSource, and extensive Property Builder options. This lab will introduce the Telerik RadGridView control, demonstrate connecting to a DataSource, using the PropertyBuilder and applying a theme.

You can find the complete source for this project at:

\VS Projects\Grid\<VB|CS>\RadGridView\01_RadGridViewIntro

1. Start a new Windows Forms Project. Change the form size to 800x400, or something sufficiently large to view several columns of data.
2. Drag a RadGridView grid onto the form, and set the **Dock** property to "Fill".
3. Open the RadGridView’s Smart Tag, and under "Choose DataSource" select "Add Project DataSource".
4. Use the DataSource wizard to set up a DataBase DataSource to the AdventureWorks database, and the Sales.SalesPerson table. Select all the columns in the table, and save the BindingSource to the project.

   The RadGridView should now be configured to show all of the columns in the table.
5. Use the Properties window for the RadGridView to set the **AutoSizeColumnsMode** property of the MasterGridView Template to "Fill",
6. Drag an AquaTheme component from the Toolbox onto the form.
7. Using the Smart Tag, set the RadGridView Theme name to "Aqua"
8. Use the Smart Tag to open the Property Builder and uncheck the following columns: "TerritoryID", "rowguid", and "ModifiedDate" from the columns list.
9. Also in the Property Builder, Advanced tab, set the FormatString of the SalesQuota, Bonus, SalesYTD, and SalesLastYear to "{0:C}", the FormatString of CommissionPct to "{0:P}". In the and all the columns’ text alignment properties to MiddleCenter.
10. Close the Property Builder.
11. Press Ctl-F5 to run the project and view the data in the RadGridView.



RadGridView Lab 1

Drag a column here to group by this column.

SalesPersonID	SalesQuota	Bonus	CommissionPct	SalesYTD	SalesLastYear
Click here to add a new row					
268		$0.00	0.00 %	$677,558.47	$0.00
275	$300,000.00	$4,100.00	1.20 %	$4,557,045.05	$1,750,406.48
276	$250,000.00	$2,000.00	1.50 %	$5,200,475.23	$1,439,156.03
277	$250,000.00	$2,500.00	1.50 %	$3,857,163.63	$1,997,186.20
278	$250,000.00	$500.00	1.00 %	$1,764,938.99	$1,620,276.90
279	$300,000.00	$6,700.00	1.00 %	$2,811,012.72	$1,849,640.94
280	$250,000.00	$5,000.00	1.00 %	$0.00	$1,927,059.18
281	$250,000.00	$3,550.00	1.00 %	$3,018,725.49	$2,073,506.00
282	$250,000.00	$5,000.00	1.50 %	$3,189,356.25	$2,038,234.65
283	$250,000.00	$3,500.00	1.20 %	$3,587,378.43	$1,371,635.32
284		$0.00	0.00 %	$636,440.25	$0.00
285	$250,000.00	$5,150.00	2.00 %	$5,015,682.38	$1,635,823.40
286	$250,000.00	$985.00	1.60 %	$3,827,950.24	$2,396,539.76

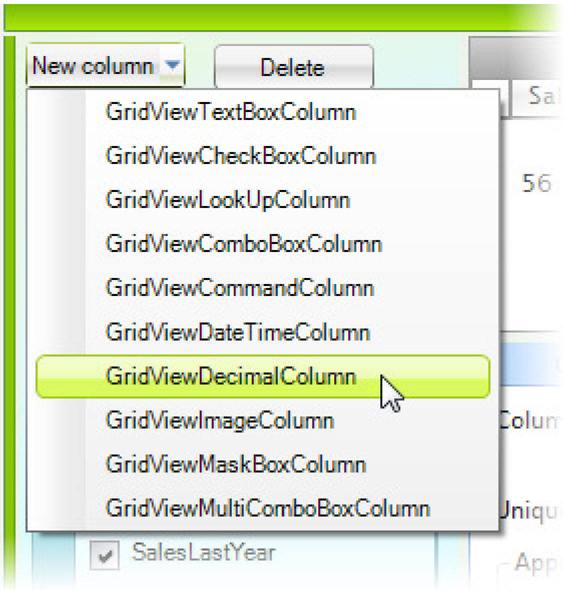
1.4 Column Manipulation

Additional columns can be added to the RadGridView table either visually by using the Property Builder, or programmatically. Setting the column expression and using different column data types allows for further flexibility within the RadGridView control, including the ability to add columns which are calculated at runtime from values in existing fields. This lab will demonstrate these techniques.

You can find the complete source for this project at:

\VS Projects\Grid\<VB|CS>\RadGridView\02_AddingColumns

1. Start a new Windows Forms Project. Change the form size to 800x400, or something sufficiently large to view several columns of data.
2. Drag a RadGridView grid onto the form, and set the **Dock** property to "Fill".
3. Open the RadGridView's Smart Tag, and under "Choose DataSource" select "Add Project DataSource".
4. Use the DataSource wizard to set up a DataBase DataSource to the AdventureWorks database, and the Sales.SalesPerson table. Select all the columns in the table, and save the BindingSource to the project.
5. Drag an AquaTheme component from the Toolbox onto the form.
6. Use the Smart Tag to open the Property Builder.
7. Uncheck the following columns: "TerritoryID", "rowguid", and "ModifiedDate" from the columns list.
8. Optionally, you can change the Header Text of the columns to more meaningfull names.
9. Use the Property Builder to add a new GridViewDecimalColumn to the grid. **Note**: Be sure to set focus to the grid itself by clicking on one of the Column names in the left column before trying to add a column. Name the new column's UniqueName to "PercentQuota" and the Header Text to "Percent of Quota".



10. Close the Property Builder.
11. Add an namespace reference to the code-behind of the form.

[VB] Adding Namespace Reference

```
Imports Telerik.WinControls.UI
```

[C#] Adding Namespace Reference

```
using Telerik.WinControls.UI;
```

12. Add the code below to the Load method of the form.

Notice that the "PercentQuota" expression is assigned in code. Then a new column "Over 15 Percent" is created completely from scratch and added to the columns collection. This code programmatically creates a CheckBox column which shows if a salesperson has reached over 15% of their sales quota for the year. Then column formats are set and finally the GridViewTemplate BestFitColumns() method is called. BestFitColumns calculates the best fit for each column based on the header text and data width so that neither header nor data is obscured.

[VB] Handling the Form Load Event

```
Private Sub RadGridViewLab2_Load(ByVal sender As Object, ByVal e As EventArgs)
 ' load the dataset
 Me.salesPersonTableAdapter.Fill(Me.adventureWorksDataSet.SalesPerson)
 ' assign the "PercentQuota" expression, and set format to be a percentage
 Me.radGridView1.Columns("PercentQuota").Expression = "SalesYTD/SalesQuota"
 (DirectCast(Me.radGridView1.Columns("PercentQuota"), GridViewDataColumn)).FormatString =
"{0:P}"
 ' create a new "Over 15 Percent" checkbox
 Dim checkboxColumn As New GridViewCheckBoxColumn()
 checkboxColumn.UniqueName = "CheckBoxColumn"
 checkboxColumn.HeaderText = "Over 15%"
 checkboxColumn.FieldName = "Over15Percent"
 checkboxColumn.Width = 60
 checkboxColumn.Expression = "PercentQuota > 15" radGridView1.Columns.Add(checkboxColumn
 ' set column formats
 (DirectCast(Me.radGridView1.Columns("SalesQuota"), GridViewDataColumn)).FormatString =
"{0:C}"
 (DirectCast(Me.radGridView1.Columns("Bonus"), GridViewDataColumn)).FormatString = "{0:C
 (DirectCast(Me.radGridView1.Columns("SalesYTD"), GridViewDataColumn)).FormatString =
"{0:C}"
 (DirectCast(Me.radGridView1.Columns("SalesLastYear"), GridViewDataColumn)).FormatString
"{0:C}"
 (DirectCast(Me.radGridView1.Columns("CommissionPct"), GridViewDataColumn)).FormatString
"{0:P}"
 ' get the best fit for each column based on header text and data
 radGridView1.MasterGridViewTemplate.BestFitColumns()
End Sub
```

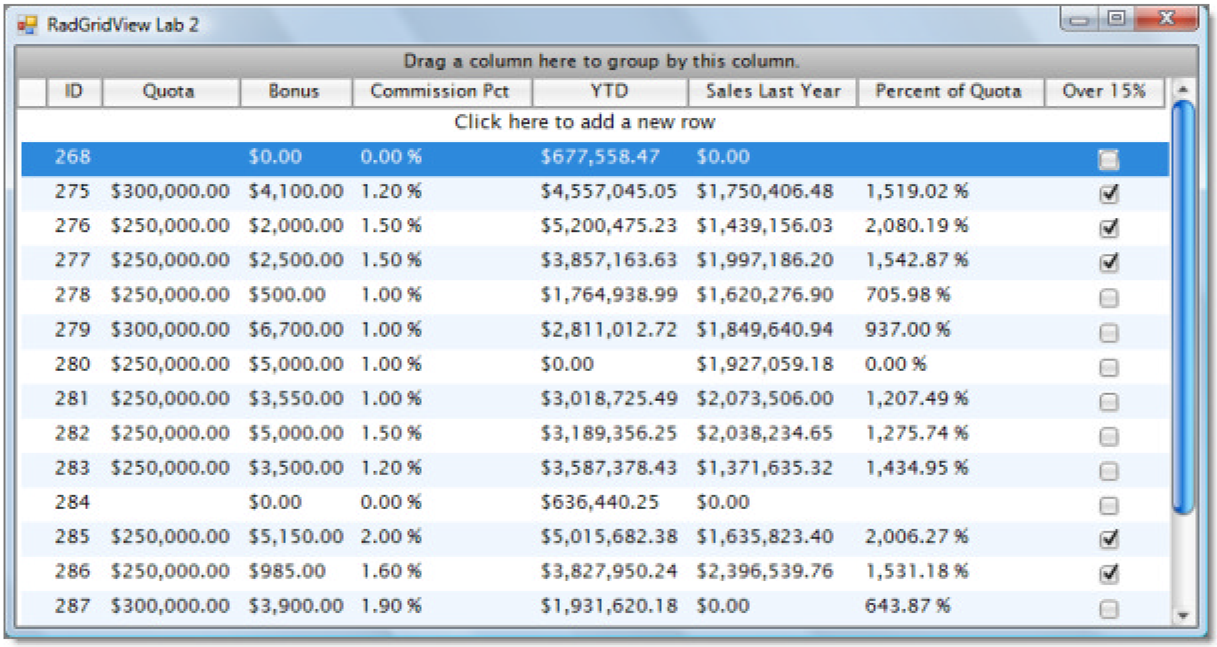
[C#] Handling the Form Load Event

```
private void RadGridViewLab2_Load(object sender, EventArgs e)
{
 // load the dataset
 this.salesPersonTableAdapter.Fill(this.adventureWorksDataSet.SalesPerson);
 // assign the "PercentQuota" expression, and set format to be a percentage
 this.radGridView1.Columns["PercentQuota"].Expression = "SalesYTD/SalesQuota";
 ((GridViewDataColumn)this.radGridView1.Columns["PercentQuota"]).FormatString = "{0:P}";
 // create a new "Over 15 Percent" checkbox
 GridViewCheckBoxColumn checkboxColumn = new GridViewCheckBoxColumn();
 checkboxColumn.UniqueName = "CheckBoxColumn";
 checkboxColumn.HeaderText = "Over 15%";
 checkboxColumn.FieldName = "Over15Percent";
 checkboxColumn.Width = 60;
 radGridView1.Columns.Add(checkboxColumn);
 checkboxColumn.Expression = "PercentQuota > 15";

 // set column formats
```

```
    ((GridViewDataColumn)this.radGridView1.Columns["SalesQuota"]).FormatString = "{0:C}";
    ((GridViewDataColumn)this.radGridView1.Columns["Bonus"]).FormatString = "{0:C}";
    ((GridViewDataColumn)this.radGridView1.Columns["SalesYTD"]).FormatString = "{0:C}";
    ((GridViewDataColumn)this.radGridView1.Columns["SalesLastYear"]).FormatString = "{0:C}"
    ((GridViewDataColumn)this.radGridView1.Columns["CommissionPct"]).FormatString = "{0:P}"
    // get the best fit for each column based on header text and data
    radGridView1.MasterGridViewTemplate.BestFitColumns();
  }
```

13. Run the project to view the column created in design view and the column created programmatically with the rest of the data.



RadGridView Lab 2
Drag a column here to group by this column.
Click here to add a new row

ID	Quota	Bonus	Commission Pct	YTD	Sales Last Year	Percent of Quota	Over 15%
268		$0.00	0.00 %	$677,558.47	$0.00		□
275	$300,000.00	$4,100.00	1.20 %	$4,557,045.05	$1,750,406.48	1,519.02 %	☑
276	$250,000.00	$2,000.00	1.50 %	$5,200,475.23	$1,439,156.03	2,080.19 %	☑
277	$250,000.00	$2,500.00	1.50 %	$3,857,163.63	$1,997,186.20	1,542.87 %	☑
278	$250,000.00	$500.00	1.00 %	$1,764,938.99	$1,620,276.90	705.98 %	□
279	$300,000.00	$6,700.00	1.00 %	$2,811,012.72	$1,849,640.94	937.00 %	□
280	$250,000.00	$5,000.00	1.00 %	$0.00	$1,927,059.18	0.00 %	□
281	$250,000.00	$3,550.00	1.00 %	$3,018,725.49	$2,073,506.00	1,207.49 %	□
282	$250,000.00	$5,000.00	1.50 %	$3,189,356.25	$2,038,234.65	1,275.74 %	□
283	$250,000.00	$3,500.00	1.20 %	$3,587,378.43	$1,371,635.32	1,434.95 %	□
284		$0.00	0.00 %	$636,440.25	$0.00		□
285	$250,000.00	$5,150.00	2.00 %	$5,015,682.38	$1,635,823.40	2,006.27 %	☑
286	$250,000.00	$985.00	1.60 %	$3,827,950.24	$2,396,539.76	1,531.18 %	☑
287	$300,000.00	$3,900.00	1.90 %	$1,931,620.18	$0.00	643.87 %	□

1.5 Grouping, Sorting and Filtering

It is often useful to organize data for a clearer presentation. The RadGridView control allows you to perform this organization either at run-time, through options available to the user, or beforehand from inside your project. Grouping allows data to be organized according to commonalities between records, filtering shows only data meeting certain criteria, and sorting changes the order based on a particular field. The following labs will demonstrate how these functions can be performed by the user at run-time, from inside the designer, or programmatically using the API.

Grouping

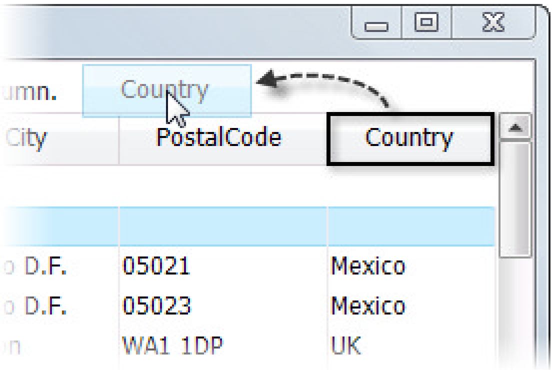
1.5.1 User Grouping at Runtime

Grouping is supported at runtime by dragging and dropping column names into the Grouping Panel, using the Property Builder to perform the drag and drop grouping ahead of time, or programmatically from the code-behind. We will use the same project to perform these grouping functions, but demonstrate each type separately.

You can find the complete source for this project at:

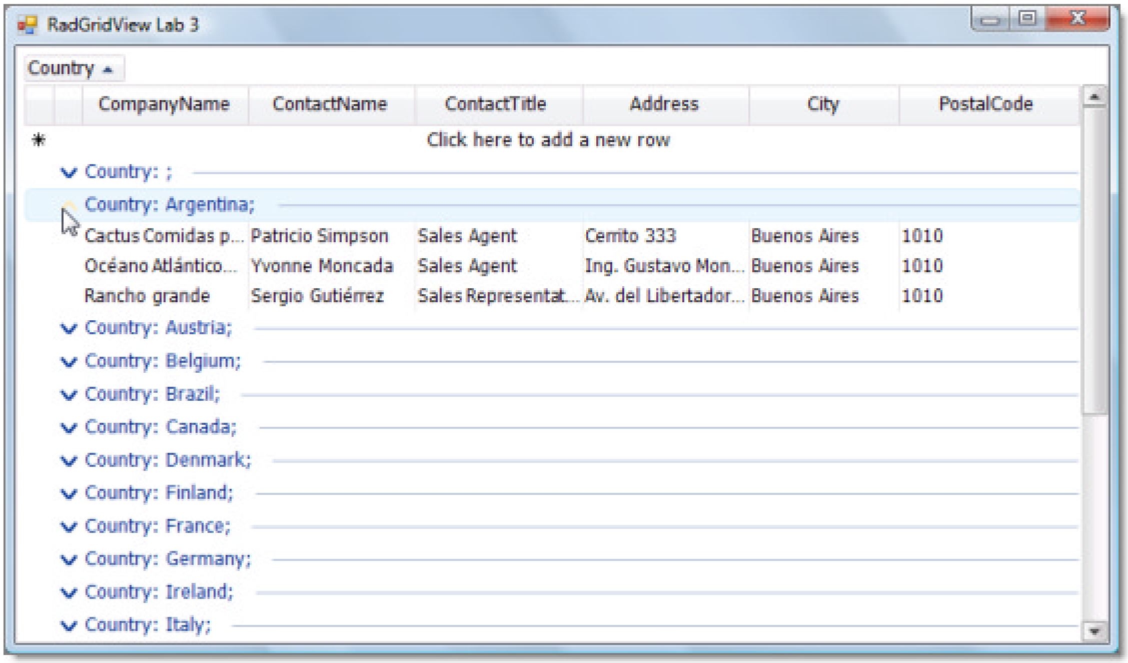
\VS Projects\Grid\<VB|CS>\RadGridView\03_Grouping

1. Create a new Windows Forms Project
2. Drag a RadGridView control onto the form, and connect its DataSource to the Customers table of the NorthWind sample database.
3. Use the RadGridView's Smart Tag to open the Property Builder, and un-select the following fields: "CustomerID", "Region", "Phone", and "Fax", then close the Property Builder.
4. Run the project to view the data in the grid. To group the data by Country, grab the Country column header, and drag it to the space above the grid. Or, right-click on the column header, and choose "Group By Column" from the context menu.



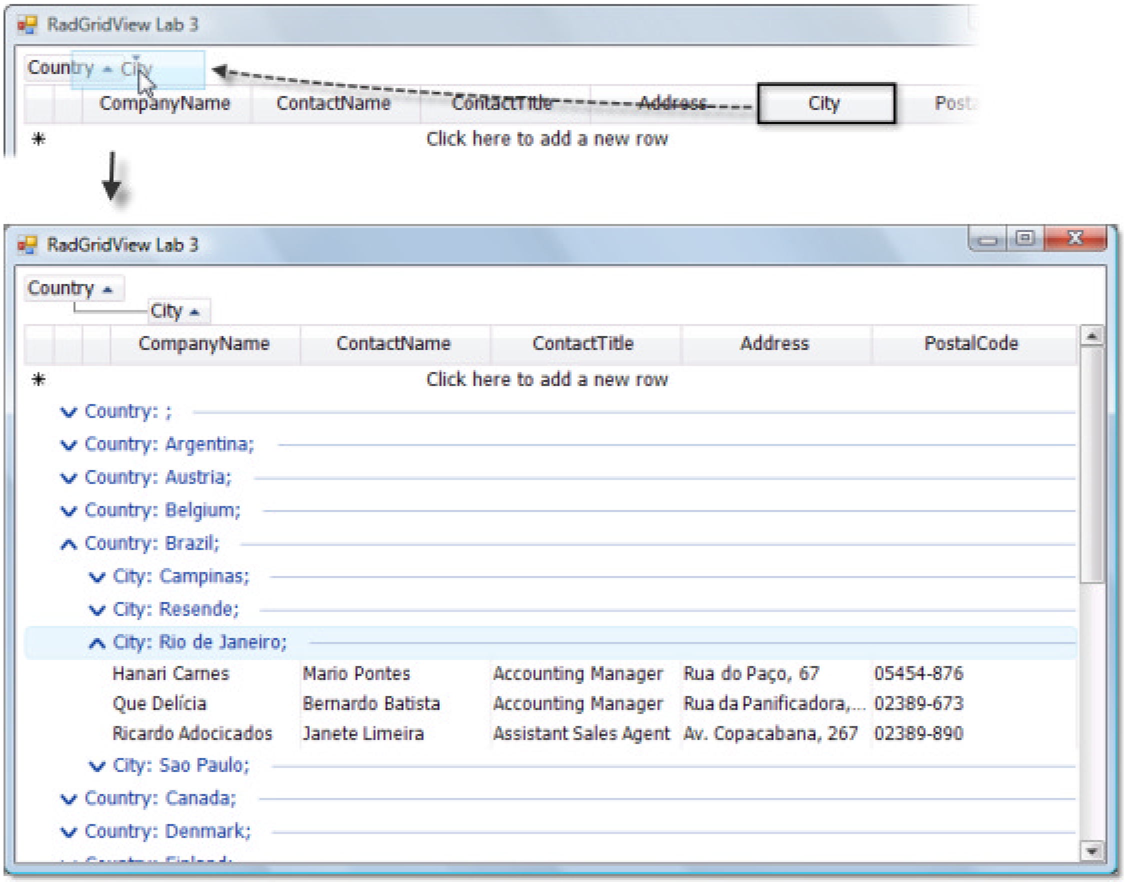
City	PostalCode	Country
o D.F.	05021	Mexico
o D.F.	05023	Mexico
n	WA1 1DP	UK

5. Now, you can expand a Country group to view the data in that group. Also, the order of the grouping can be changed by clicking the sort arrow on the right side of the group header in the Grouping Panel.



CompanyName	ContactName	ContactTitle	Address	City	PostalCode
Click here to add a new row					
Country: ;					
Country: Argentina;					
Cactus Comidas p...	Patricio Simpson	Sales Agent	Cerrito 333	Buenos Aires	1010
Océano Atlántico...	Yvonne Moncada	Sales Agent	Ing. Gustavo Mon...	Buenos Aires	1010
Rancho grande	Sergio Gutiérrez	Sales Representat...	Av. del Libertador...	Buenos Aires	1010
Country: Austria;					
Country: Belgium;					
Country: Brazil;					
Country: Canada;					
Country: Denmark;					
Country: Finland;					
Country: France;					
Country: Germany;					
Country: Ireland;					
Country: Italy;					

6. To add an additional subgroup by City, drag the City column header to the right-hand side of the Country grouping block.

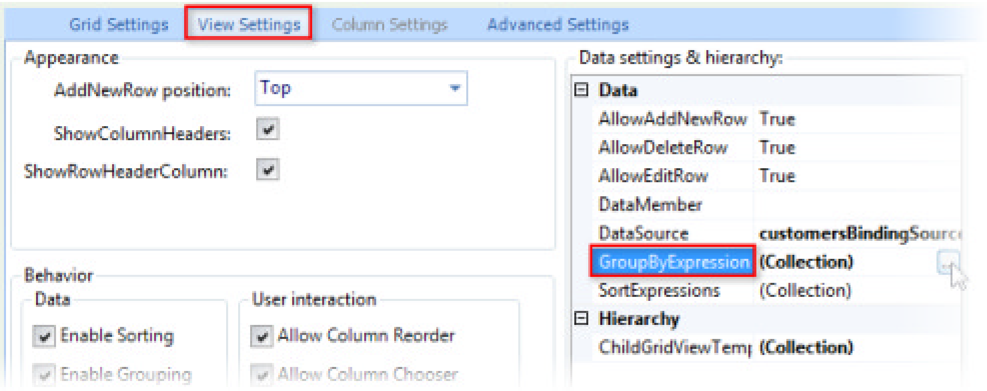


CompanyName	ContactName	ContactTitle	Address	PostalCode
Country: ;				
Country: Argentina;				
Country: Austria;				
Country: Belgium;				
Country: Brazil;				
City: Campinas;				
City: Resende;				
City: Rio de Janeiro;				
Hanari Carnes	Mario Pontes	Accounting Manager	Rua do Paço, 67	05454-876
Que Delícia	Bernardo Batista	Accounting Manager	Rua da Panificadora,...	02389-673
Ricardo Adocicados	Janete Limeira	Assistant Sales Agent	Av. Copacabana, 267	02389-890
City: Sao Paulo;				
Country: Canada;				
Country: Denmark;				

The hierarchy of the grouping can be re-arranged by changing the header blocks within the Grouping Panel. To Ungroup by a column, simply drag the column header back to the header row.

1.5.2 Grouping Using the Property Builder at Design-Time

1. Use the Smart Tag to open the Property Builder on the RadGridView control.
2. In the View Settings tab, "Data settings & hierarchy" area, open the **GroupByExpressions** collection.



AddNewRow position: Top

ShowColumnHeaders: ☑

ShowRowHeaderColumn: ☑

☑ Enable Sorting

☑ Enable Grouping

☑ Allow Column Reorder

☑ Allow Column Chooser

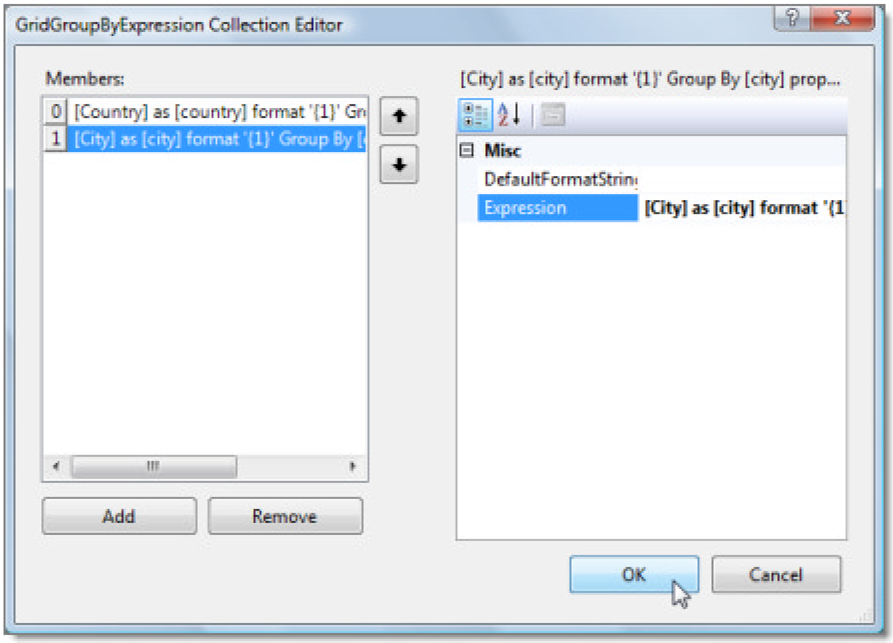
Data	
AllowAddNewRow	True
AllowDeleteRow	True
AllowEditRow	True
DataMember	
DataSource	customersBindingSourc
GroupByExpression	(Collection)
SortExpressions	(Collection)
Hierarchy	
ChildGridViewTemp	(Collection)

3. In the Collection Editor, add a new GroupByExpression and set the **Expression** property to:

[Country] as [country] format '{1}' Group By [country] DESC

4. Add a second GroupByExpression and set its Expression to

   [City] as [city] format '{1}' Group By [city]



0 [Country] as [country] format '{1}' Gr

1 [City] as [city] format '{1}' Group By [

[City] as [city] format '{1}' Group By [city] prop...

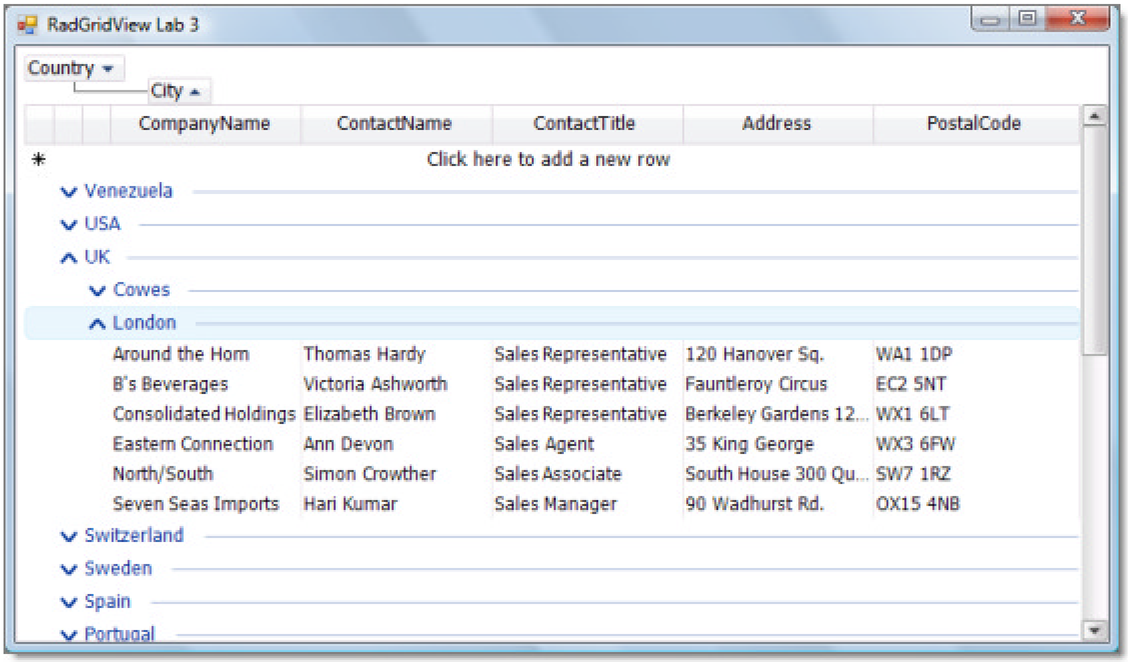
Expression [City] as [city] format '{1

5. Click **OK** to close the Collection Editor
6. Click **OK** to close the Property Builder.

   You should now see your new groupings in the Grouping Panel in design view, just as earlier when they were created by the user at run-time.

7. Run the project to view the grouped data, this time in descending order for country, and ascending for city.

> Although the groupings set up in the designer will be applied on start-up, at runtime, the user can still drag the grouped header to re-arrange, add, or delete groupings from the grid, as long as the EnableGrouping property is set to True.



CompanyName	ContactName	ContactTitle	Address	PostalCode
Click here to add a new row				
Venezuela				
USA				
UK				
Cowes				
London				
Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.	WA1 1DP
B's Beverages	Victoria Ashworth	Sales Representative	Fauntleroy Circus	EC2 5NT
Consolidated Holdings	Elizabeth Brown	Sales Representative	Berkeley Gardens 12...	WX1 6LT
Eastern Connection	Ann Devon	Sales Agent	35 King George	WX3 6FW
North/South	Simon Crowther	Sales Associate	South House 300 Qu...	SW7 1RZ
Seven Seas Imports	Hari Kumar	Sales Manager	90 Wadhurst Rd.	OX15 4NB
Switzerland				
Sweden				
Spain				
Portugal				

1.5.3 Programmatic Grouping at Run-Time

1. Add the following code to the form's Load event to remove the city grouping added in the designer, and add a grouping by Contact Title.

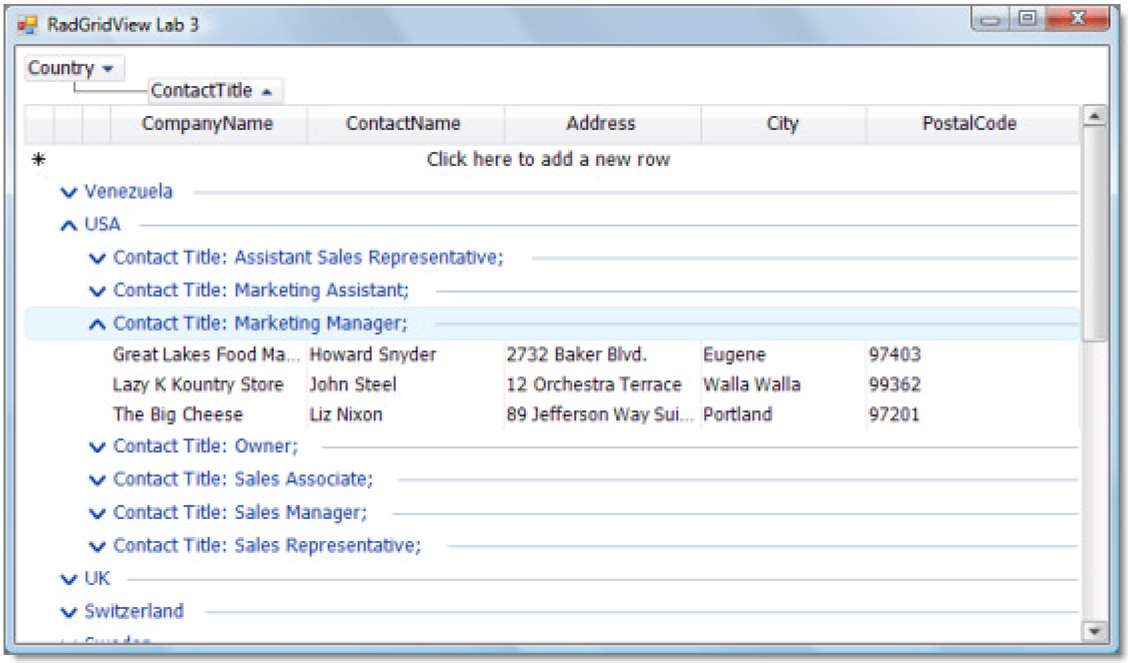
   [VB] Removing and Adding Groupings

```
Private Sub RadGridViewLab3_Load(ByVal sender As Object, ByVal e As EventArgs)
 Me.customersTableAdapter.Fill(Me.dataSet1.Customers)
 ' remove the "City" grouping added in the designer
 radGridView1.MasterGridViewTemplate.GroupByExpressions.RemoveAt(1)
 ' add a new grouping by "Contact Title"
 radGridView1.MasterGridViewTemplate.GroupByExpressions.Add("[ContactTitle] as [Contact
Title] Group By [Contact Title] ASC")
End Sub
```

   [C#] Removing and Adding Groupings

```
private void RadGridViewLab3_Load(object sender, EventArgs e)
{
 this.customersTableAdapter.Fill(this.dataSet1.Customers);
 // remove the "City" grouping added in the designer
 radGridView1.MasterGridViewTemplate.GroupByExpressions.RemoveAt(1);
 // add a new grouping by "Contact Title"
 radGridView1.MasterGridViewTemplate.GroupByExpressions.Add(
     "[ContactTitle] as [Contact Title] Group By [Contact Title] ASC");
}
```

2. Run the project, and view the data, now grouped by country, and then sub-grouped by Contact Type within the country groups.



Click here to add a new row

Venezuela

USA

Contact Title: Assistant Sales Representative;

Contact Title: Marketing Assistant;

Contact Title: Marketing Manager;

CompanyName	ContactName	Address	City	PostalCode
Great Lakes Food Ma...	Howard Snyder	2732 Baker Blvd.	Eugene	97403
Lazy K Kountry Store	John Steel	12 Orchestra Terrace	Walla Walla	99362
The Big Cheese	Liz Nixon	89 Jefferson Way Sui...	Portland	97201

Contact Title: Owner;

Contact Title: Sales Associate;

Contact Title: Sales Manager;

Contact Title: Sales Representative;

UK

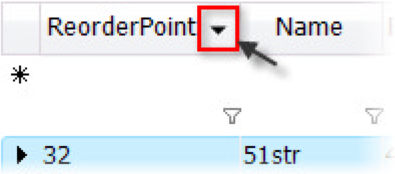
Switzerland

Sorting and Filtering

You can find the complete source for this project at:

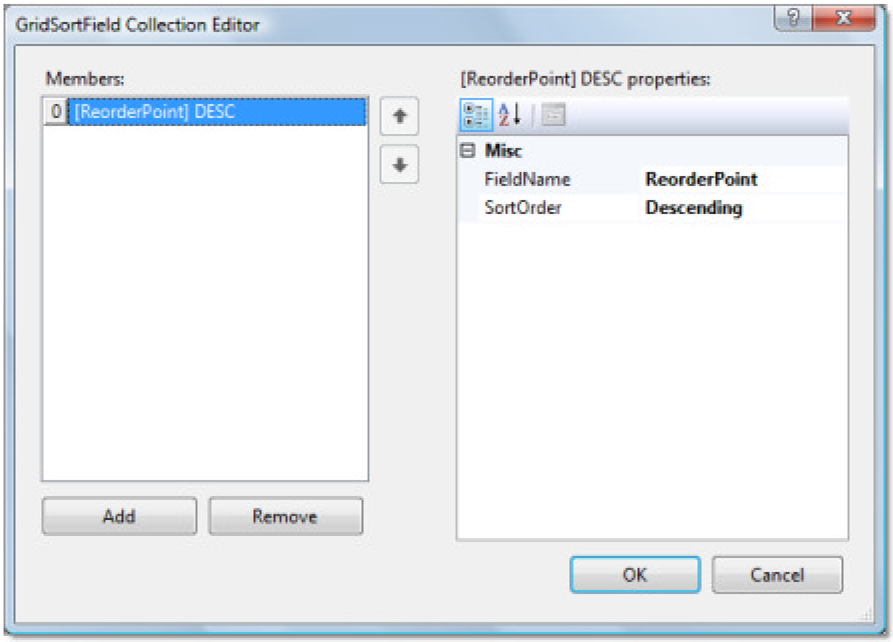
\VS Projects\Grid\<VB|CS>\RadGridView\04_SortingandFiltering

1. In a new Windows Forms Project, drag a RadGridView control onto the form.
2. Connect its DataSource to the Production.Product table of the AdventureWorks sample database.
3. In the Property Window for the RadGridView, set the MasterGridView Template's EnableFiltering property to true.
4. Use the RadGridView's Smart Tag to open the Property Builder and un-select the following fields: ProductID, Size, SizeUnitMeasure, rowguid, Weight, WeightUnitMeasure, Style, ProductSubcategoryID, ProductModelID, SellStartDate, SellEndDate, DiscontinuedDate, and ModifiedDate.
5. Still in the Property Builder, use the Preview Panel to drag the ReorderPoint column to the furthest left position, then click the header block itself until it shows a downward arrow, indicating a descending sort on the column.



ReorderPoint	Name
32	51str

6. To view the sorting applied to the grid using the Preview Panel, navigate to the Advanced Settings tab, find the Data Group and click the SortExpressions property ellipses to open the GridSortField Collection Editor. Here, you can also add or delete sorting expressions on the grid. You should now see the descending sort on the Reorder Point column.



GridSortField Collection Editor

Members:

0 [ReorderPoint] DESC

[ReorderPoint] DESC properties:

Misc

FieldName	ReorderPoint
SortOrder	Descending

7. Add namespace references to the "Imports" (VB) or "uses" (C#) clause of your code:

[VB] Adding References

```
Imports Telerik.WinControls.UI
Imports Telerik.WinControls.Data
```

[C#] Adding References

```
using Telerik.WinControls.UI;
using Telerik.WinControls.Data;
```

8. To add an additional expression to sort by Color programmatically, close the Property Builder and add the following code to the Load handler of the form:

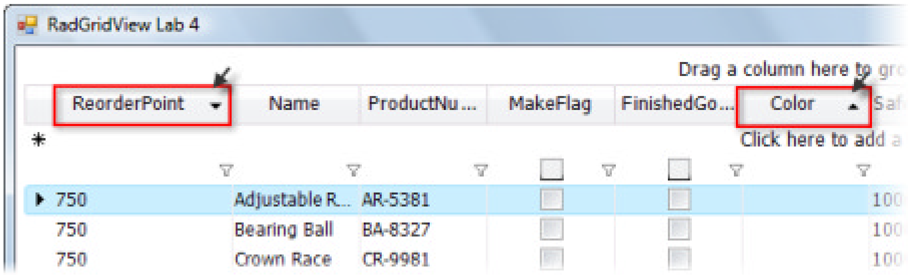
[VB] Adding a Sort Expression

```
radGridView1.MasterGridViewTemplate.SortExpressions.Add("Color", RadSortOrder.Ascending)
```

[C#] Adding a Sort Expression

```
radGridView1.MasterGridViewTemplate.SortExpressions.Add("Color", RadSortOrder.Ascending)
```

9. Run the project to view your sorted data. Notice that the data is first sorted by Reorder Point, then within that sorting, is sorted by Color.



RadGridView Lab 4

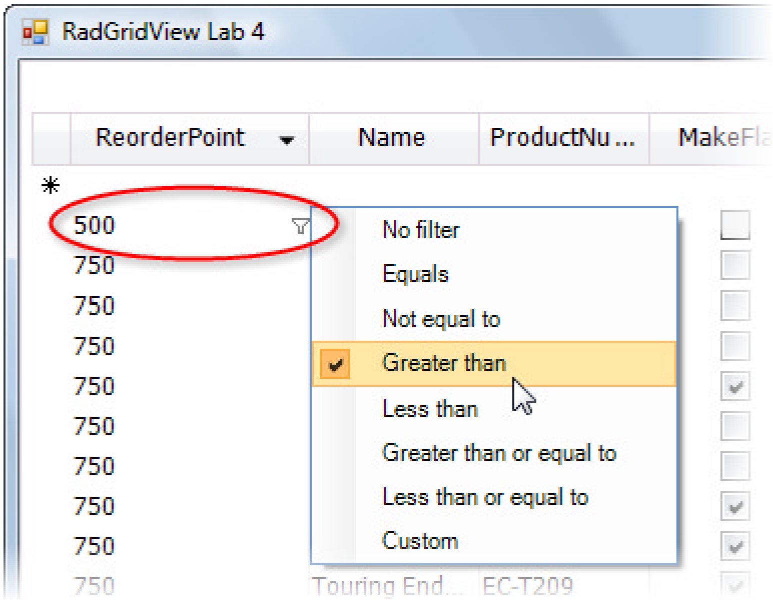
Drag a column here to gro

Click here to add a

ReorderPoint	Name	ProductNu...	MakeFlag	FinishedGo...	Color	Saf
750	Adjustable R...	AR-5381	□	□		100
750	Bearing Ball	BA-8327	□	□		100
750	Crown Race	CR-9981	□	□		100

If EnableSorting is set to true, users will still be able to change your default sorting at runtime using the column headers. Setting EnableSorting to false freezes the sort order to the configuration set at either design time or programmatically.

10. To add run-time filtering to the table, right-click the filter icon on the Reorder Point column, and choose "Greater than" from the context menu. Now enter "500" into the filter row textbox to only show products with a Reorder Point greater than 500.



You can right-click and choose "No filter" to remove the filtering criteria.

11. Stop the project and return to the Form’s Load handler. Now programmatically add a filtering expression to only show products whose "MakeFlag" is true. Add the code below to the Load handler.

This step creates a FilterExpression object and assigns it to the Column's Filter property. The FilterExpression constructor has several overloads, but the parameters for this example are:

- A BinaryOperation enumeration member that can be "OR"/"AND".
- A GridKnownFunction enumeration member that represent one of the filtering criteria in the drop down list, e.g. "GreaterThan", "EqualTo", etc.
- A criteria value, i.e. the value that the column value is being compared to. In the example below the value is "True".

[VB] Adding a Filter Expression

```
radGridView1.Columns("MakeFlag").Filter = New FilterExpression
(FilterExpression.BinaryOperation.[AND], GridKnownFunction.EqualTo, True)
```

[C#] Adding a Filter Expression

```
radGridView1.Columns["MakeFlag"].Filter =
 new FilterExpression(FilterExpression.BinaryOperation.AND, GridKnownFunction.EqualTc
true);
```

12. Add a custom filtering expression to additionally filter results to only those with ProductNumbers starting with the letters "C" or "R".

    This is similar to the last filter, but instead of adding the operation, function and criteria to the constructor, they are encapsulated within a "FilterPredicate" object and added to the Columns FilterPredicates collection. This allows you to create more complex, multi-part filters.

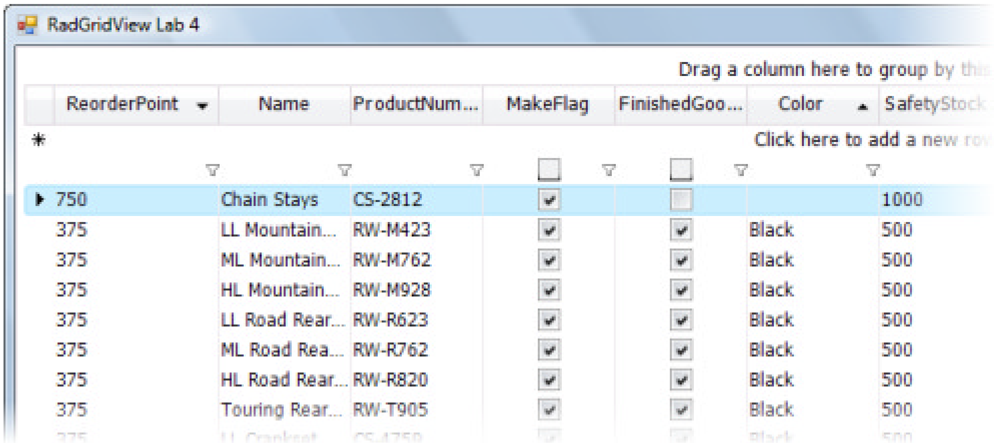
    [VB] Adding a Multiple Part Filter

```
radGridView1.Columns("ProductNumber").Filter = New FilterExpression("ProductNumber")
radGridView1.Columns("ProductNumber").Filter.Predicates.Add
(FilterExpression.BinaryOperation.[OR], GridKnownFunction.StartsWith, "C")
radGridView1.Columns("ProductNumber").Filter.Predicates.Add
(FilterExpression.BinaryOperation.[OR], GridKnownFunction.StartsWith, "R")
```

    [C#] Adding a Multiple Part Filter

```
radGridView1.Columns["ProductNumber"].Filter = new FilterExpression("ProductNumber");
radGridView1.Columns["ProductNumber"].Filter.Predicates.Add(
 FilterExpression.BinaryOperation.OR, GridKnownFunction.StartsWith, "C");
radGridView1.Columns["ProductNumber"].Filter.Predicates.Add(
 FilterExpression.BinaryOperation.OR, GridKnownFunction.StartsWith, "R");
```

13. Run the project and notice the effect of our filter expressions. Only Products with "True" MakeFlags whose number starts with "C" or "R" are shown.



ReorderPoint	Name	ProductNum...	MakeFlag	FinishedGoo...	Color	SafetyStock
750	Chain Stays	CS-2812	☑	□		1000
375	LL Mountain...	RW-M423	☑	☑	Black	500
375	ML Mountain...	RW-M762	☑	☑	Black	500
375	HL Mountain...	RW-M928	☑	☑	Black	500
375	LL Road Rear...	RW-R623	☑	☑	Black	500
375	ML Road Rea...	RW-R762	☑	☑	Black	500
375	HL Road Rear...	RW-R820	☑	☑	Black	500
375	Touring Rear...	RW-T905	☑	☑	Black	500
375	LL Crankset	CS-4759	☑	☑	Black	500

Programmatic filter expressions do not show up in the Filter Row at runtime, nor can they be changed by the user at run-time.

1.6 Hierarchy Support

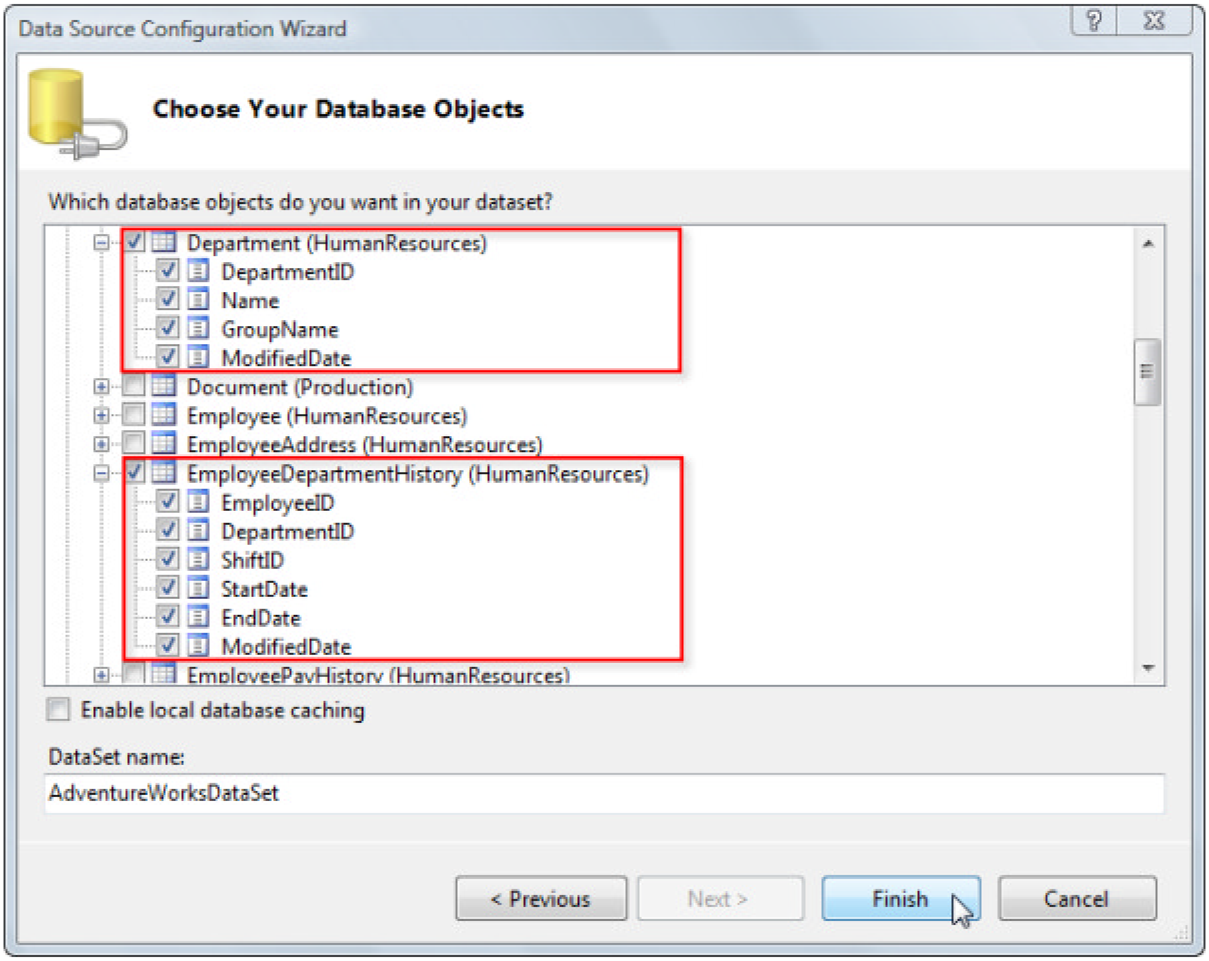
With some data, it is desirable to show tables within tables, also known as a Hierarchical view. Multiple tables can be related through data keys and RadGridView allows almost effortless setup to display such relationships. Hierarchical tables can be set up through either the designer manually, automatically, or programmatically in the code-behind. The following labs will demonstrate all three implementation methods.

Hierarchy Table Setup in the Designer

You can find the complete source for this project at:

\VS Projects\Grid\<VB|CS>\RadGridView\05_HierarchyDesigner

1. Create a new Windows Forms Project.
2. Drag a RadGridView control onto the form, and use the Smart Tag to create a new DataSource. Use the AdventureWorks database, and select both the HumanResources.Department and HumanResources.EmployeeDepartmentHistory tables.



Data Source Configuration Wizard
?
Choose Your Database Objects
< Previous
Next >
Finish
Cancel

Which database objects do you want in your dataset?

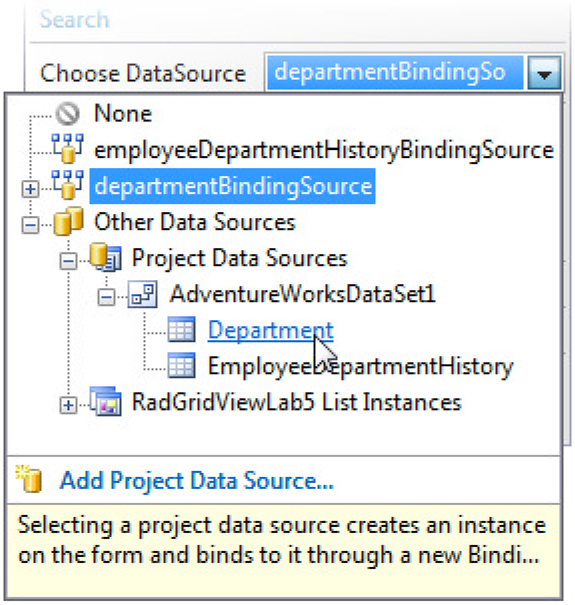
- ☑ Department (HumanResources)
  - ☑ DepartmentID
  - ☑ Name
  - ☑ GroupName
  - ☑ ModifiedDate
- ☐ Document (Production)
- ☐ Employee (HumanResources)
- ☐ EmployeeAddress (HumanResources)
- ☑ EmployeeDepartmentHistory (HumanResources)
  - ☑ EmployeeID
  - ☑ DepartmentID
  - ☑ ShiftID
  - ☑ StartDate
  - ☑ EndDate
  - ☑ ModifiedDate
- ☐ EmployeePayHistory (HumanResources)

☐ Enable local database caching

DataSet name:

AdventureWorksDataSet

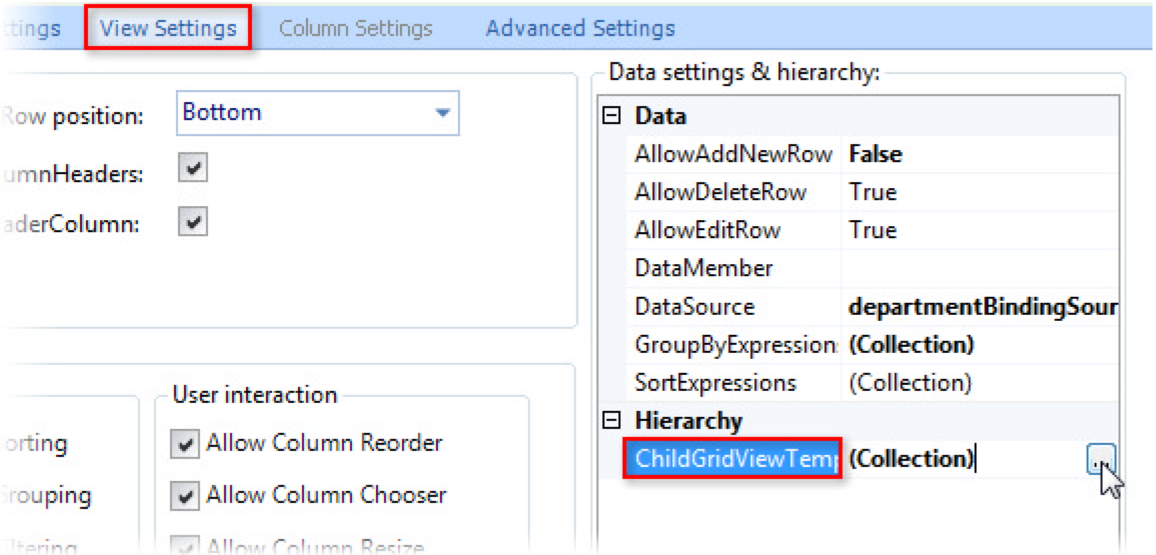
3. Again using the Smart Tag, select the Department Table as the source for the main grid.


Search
None
employeeDepartmentHistoryBindingSource
departmentBindingSource
Other Data Sources
Project Data Sources
AdventureWorksDataSet1
Department
EmployeeDepartmentHistory
RadGridViewLab5 List Instances
Add Project Data Source...

Choose DataSource: departmentBindingSo

Selecting a project data source creates an instance on the form and binds to it through a new Bindi...

4. Now, open the Property Builder for the grid. Select the View Settings tab, and in the Data settings & hierarchy pane, open the Collection Editor for the ChildGridViewTemplates.



Row position: Bottom

umnHeaders: ☑

aderColumn: ☑

☑ Allow Column Reorder

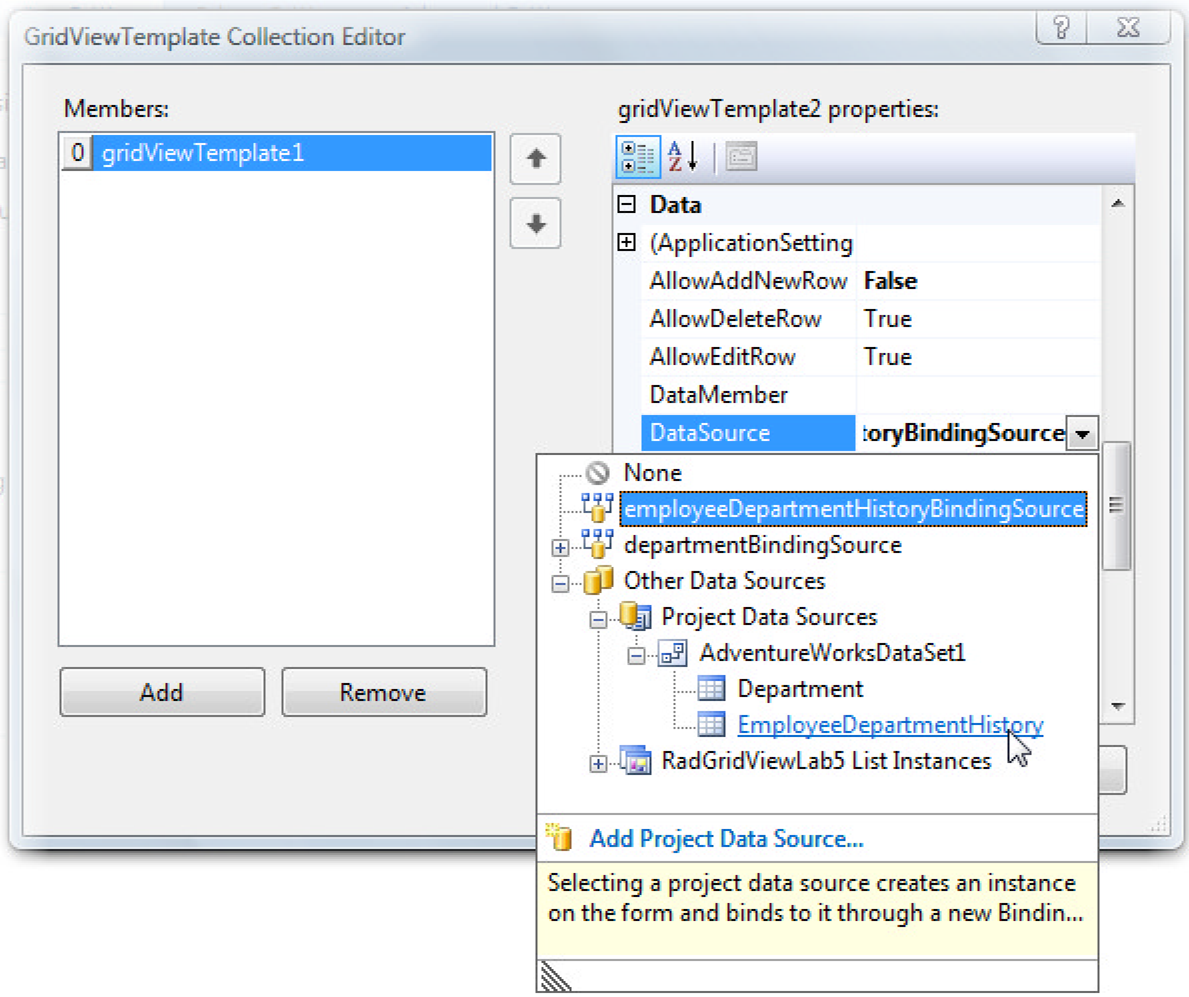
☑ Allow Column Chooser

☑ Allow Column Resize

Data settings & hierarchy:

⊟ Data	
AllowAddNewRow	**False**
AllowDeleteRow	True
AllowEditRow	True
DataMember	
DataSource	**departmentBindingSour**
GroupByExpression	**(Collection)**
SortExpressions	(Collection)
⊟ Hierarchy	
ChildGridViewTem	**(Collection)**

5. Add a new GridViewTemplate to the collection, and choose the EmployeeDepartmentHistory table as its DataSource. This will be the child table within the grid. Afterwards, click **OK** to save the ChildGridViewTemplates collection.


GridViewTemplate Collection Editor
Members:
0 gridViewTemplate1
Add
Remove
gridViewTemplate2 properties:
None
employeeDepartmentHistoryBindingSource
departmentBindingSource
Other Data Sources
Project Data Sources
AdventureWorksDataSet1
Department
EmployeeDepartmentHistory
RadGridViewLab5 List Instances
Add Project Data Source...

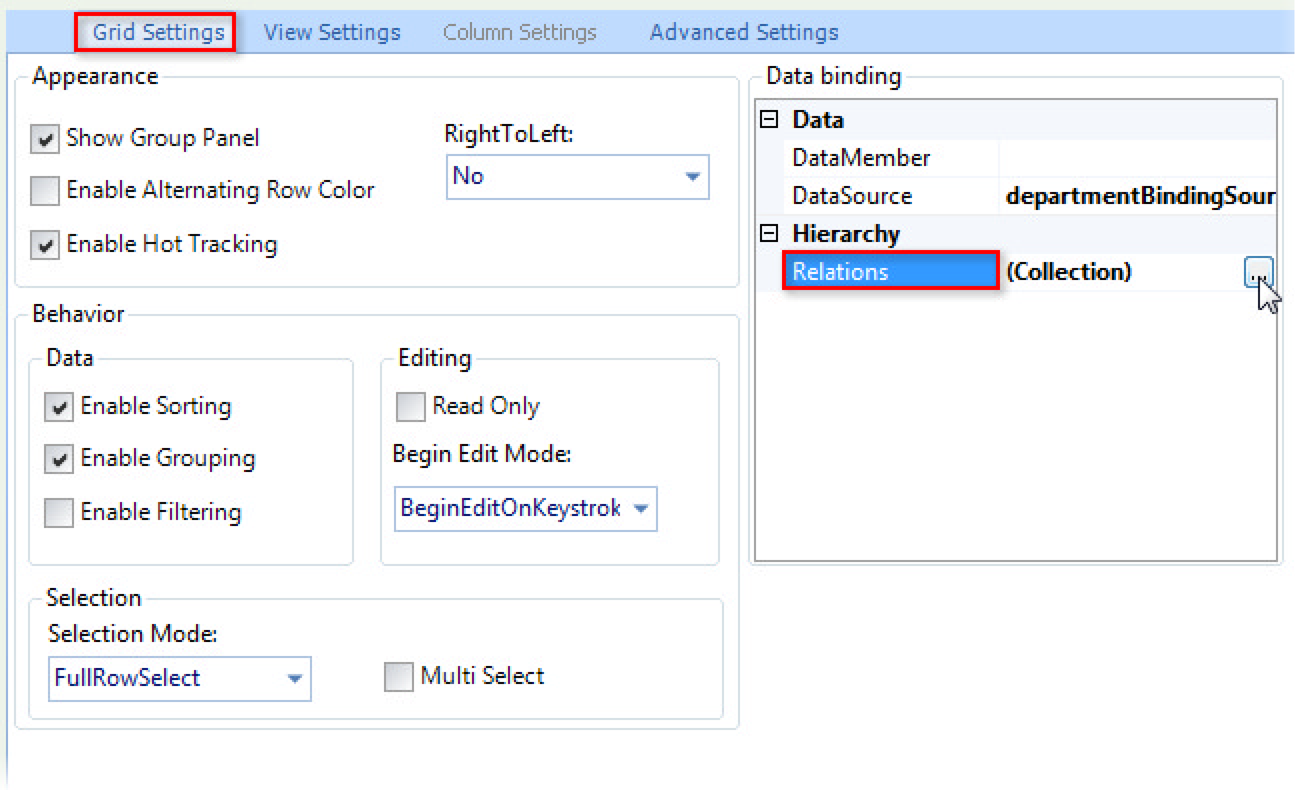
Data

(ApplicationSetting

AllowAddNewRow	False
AllowDeleteRow	True
AllowEditRow	True
DataMember	
DataSource	toryBindingSource

Selecting a project data source creates an instance on the form and binds to it through a new Bindin...

6. In the Grid Settings tab, open the Collections editor for the Relations collection.



Appearance

☑ Show Group Panel

☐ Enable Alternating Row Color

☑ Enable Hot Tracking

RightToLeft: No

Behavior

Data

☑ Enable Sorting

☑ Enable Grouping

☐ Enable Filtering

Editing

☐ Read Only

Begin Edit Mode: BeginEditOnKeystrok

Selection

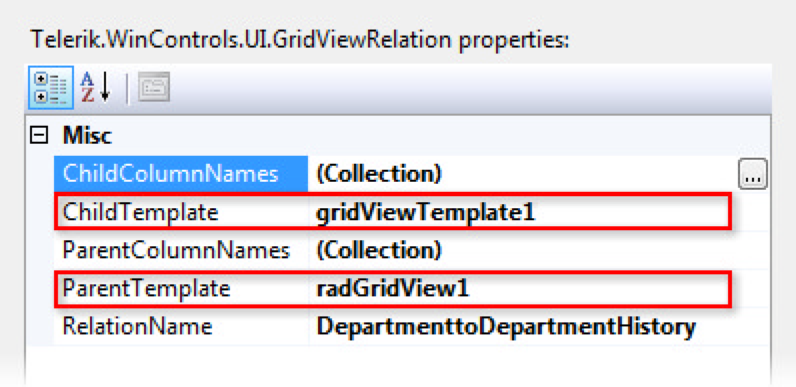
Selection Mode: FullRowSelect

☐ Multi Select

Data binding

Data	
DataMember	
DataSource	**departmentBindingSour**
Hierarchy	
Relations	**(Collection)**

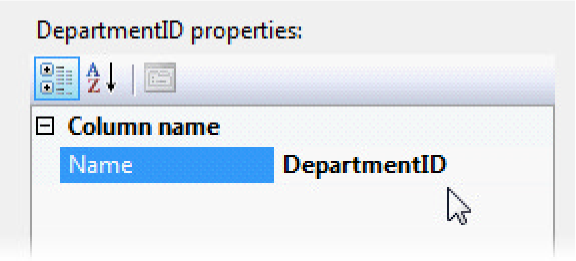
7. In the Relations Collection Editor, add a new relation by clicking the "Add" button. Then, name your relation in the properties pane on the right side of the Collection Editor. Here, the relation is named "DepartmenttoDepartmentHistory". Set the templates of the relation to your newly created Child template for the ChildTemplate property, and the MasterGridView template for the ParentTemplate.



Telerik.WinControls.UI.GridViewRelation properties:

Misc	
ChildColumnNames	**(Collection)**
ChildTemplate	**gridViewTemplate1**
ParentColumnNames	**(Collection)**
ParentTemplate	**radGridView1**
RelationName	**DepartmenttoDepartmentHistory**

8. Finally, set the related column by adding the Column Name “DepartmentID” to the collections of the ChildColumnNames and ParentColumnNames properties of the relation. Both collections should look like the figure below. Afterwards, click **OK** to close both dialogs and the Property Editor.

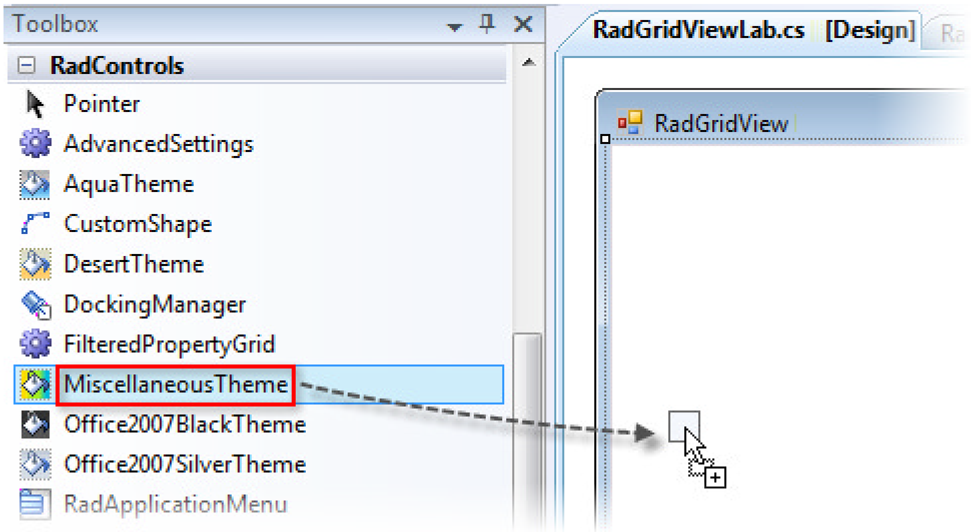


DepartmentID properties:

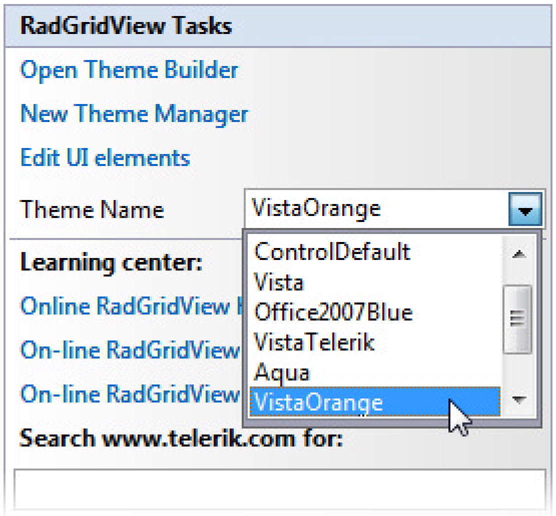
Column name	
Name	**DepartmentID**

9. Although not necessary for functionality, for this example we will use a Theme on our RadGridView control to show a bit more contrast on our parent and child tables. Drag a "Miscellaneous Theme" object from the toolbox onto the form itself.

   This step will place a MiscellaneousTheme component instance into the component tray below the form design surface.

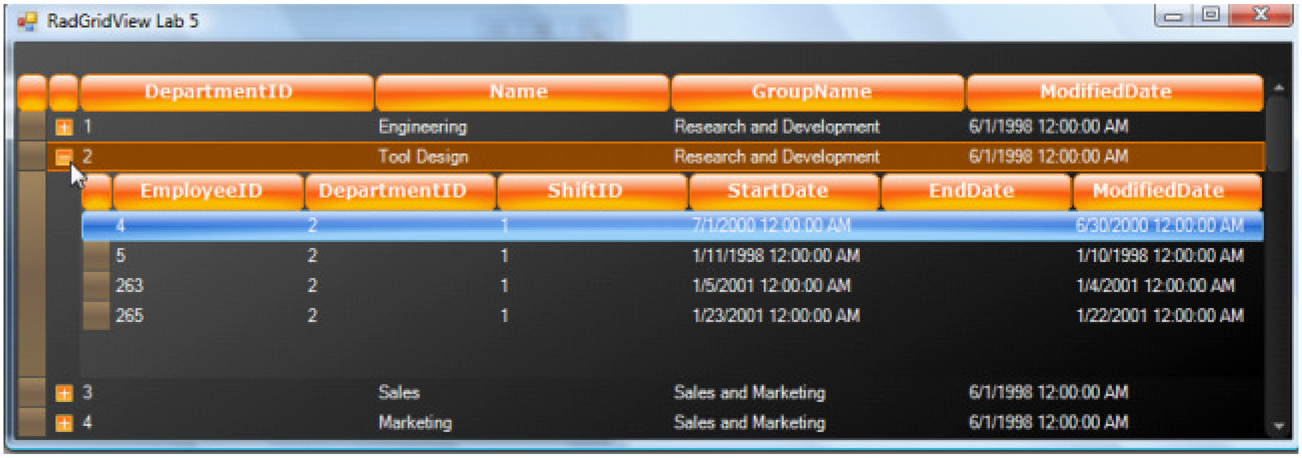


10. Using the Smart Tag of the RadGridView control, set the Theme to "VistaOrange".



Theme Name VistaOrange

11. Run the project and view the hierarchical tables in the form:



RadGridView Lab 5

DepartmentID	Name	GroupName	ModifiedDate
1	Engineering	Research and Development	6/1/1998 12:00:00 AM
2	Tool Design	Research and Development	6/1/1998 12:00:00 AM

EmployeeID	DepartmentID	ShiftID	StartDate	EndDate	ModifiedDate
4	2	1	7/1/2000 12:00:00 AM		6/30/2000 12:00:00 AM
5	2	1	1/11/1998 12:00:00 AM		1/10/1998 12:00:00 AM
263	2	1	1/5/2001 12:00:00 AM		1/4/2001 12:00:00 AM
265	2	1	1/23/2001 12:00:00 AM		1/22/2001 12:00:00 AM

DepartmentID	Name	GroupName	ModifiedDate
3	Sales	Sales and Marketing	6/1/1998 12:00:00 AM
4	Marketing	Sales and Marketing	6/1/1998 12:00:00 AM

Notice that the Main Grid shows the data from the Departments database table, and by clicking the plus sign to the left of a Department record, you can view the Child table, which displays the Employees who have the same DepartmentID in the EmployeeDepartmentHistory table.

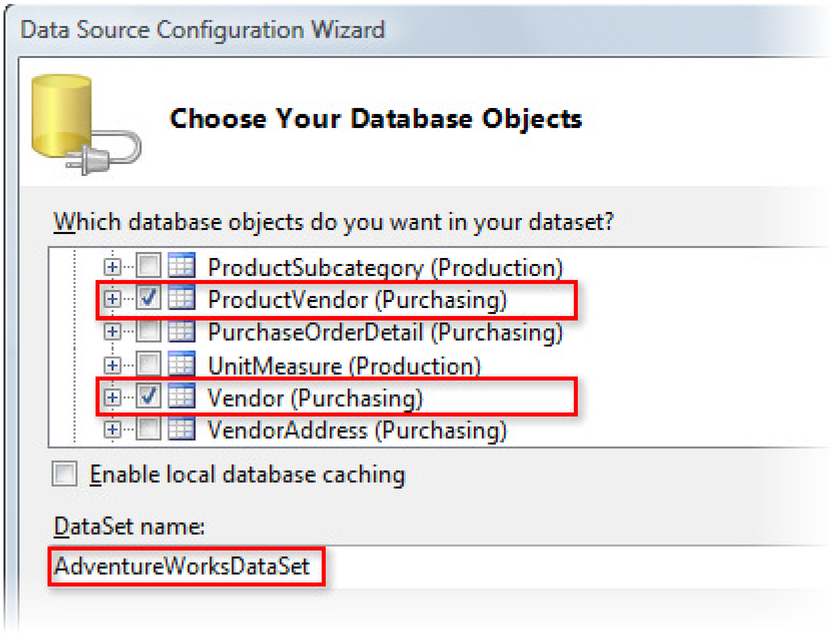
Automatic Hierarchy Table Setup

You can find the complete source for this project at:

\VS Projects\Grid\<VB|CS>\RadGridView\06_HierarcyAuto

Besides creating the table relationships yourself, the RadGridView control can automatically generate hierarchical tables for use based upon Dataset relationships.

1. Create a new Windows Forms Project.
2. Drag a RadGridView control onto the form, and use the Smart Tag to create a new DataSource. Use the AdventureWorks database, and select both the Purchasing.Vender and Purchasing.ProductVender tables. This will create a DataSet which includes both of the tables. Note the name of your new Dataset. In this example the name of the dataset is "AdventureWorksDataSet".



Data Source Configuration Wizard

Choose Your Database Objects

Which database objects do you want in your dataset?

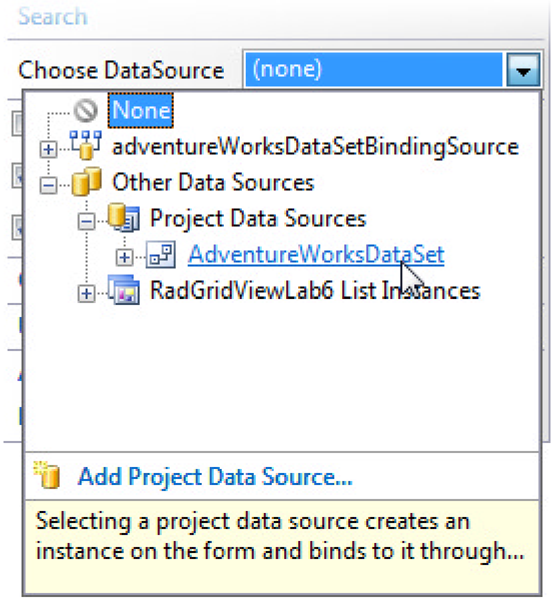
- ⊞ □ ProductSubcategory (Production)
- ⊞ ☑ ProductVendor (Purchasing)
- ⊞ □ PurchaseOrderDetail (Purchasing)
- ⊞ □ UnitMeasure (Production)
- ⊞ ☑ Vendor (Purchasing)
- ⊞ □ VendorAddress (Purchasing)

□ Enable local database caching

DataSet name:

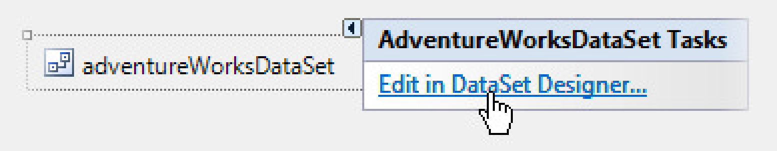
AdventureWorksDataSet

3. Use the Smart Tag to set the DataSource property to the DataSet itself. This will give RadGridView access to both tables and the relationship between them.

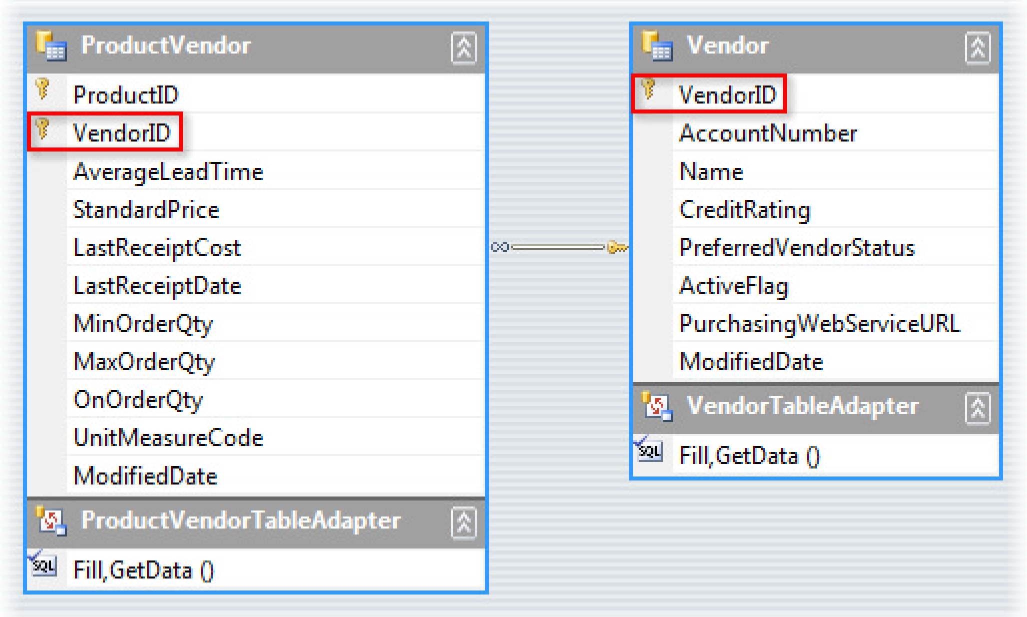


Selecting a project data source creates an instance on the form and binds to it through...

4. To view the relationships that are automatically created in the new Dataset, close the Property Builder and use the SmartTag on the Dataset object placeholder in the Form’s design panel to choose "Edit in DataSet Designer".



5. In the DataSet Designer, notice that the two tables are linked by a relationship line. The relationship between the tables is the VendorID field, which is the Primary Key for the Vendor table, and a Foreign Key for the ProductVendor table. This relationship will be the basis for our auto-generated hierarchical table view.



ProductVendor

ProductID
VendorID
AverageLeadTime
StandardPrice
LastReceiptCost
LastReceiptDate
MinOrderQty
MaxOrderQty
OnOrderQty
UnitMeasureCode
ModifiedDate

ProductVendorTableAdapter

Fill,GetData ()

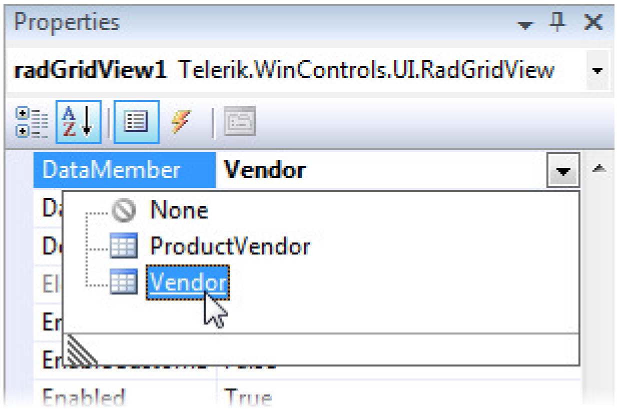
Vendor

VendorID
AccountNumber
Name
CreditRating
PreferredVendorStatus
ActiveFlag
PurchasingWebServiceURL
ModifiedDate

VendorTableAdapter

Fill,GetData ()

6. Close the DataSet Designer and return to the design view of the main form. Again, to show contrast in the tables, you may add the Miscellanious Theme to the form, and set the table’s theme to VistaOrange.
7. Now, in the Properties Window for the RadGridView, set the DataMember property to Vendor. This sets the parent grid to the Vendor data. When the hierarchy is generated, ProductVendor will now be the child table.



Properties

radGridView1 Telerik.WinControls.UI.RadGridView

DataMember Vendor

8. Also, we must tell the RadGridView to generate the hierarchy automatically. In the Properties window, set the **AutoGenerateHierarchy** property to true.
9. Since we used the DataSet as the source for our grid, we must add TableAdapters to load data from the database. Navigate to the code-behind for the Form, and add the following declarations in the form's Load event handler. The code instantiates table adapters for both tables and fills each table.

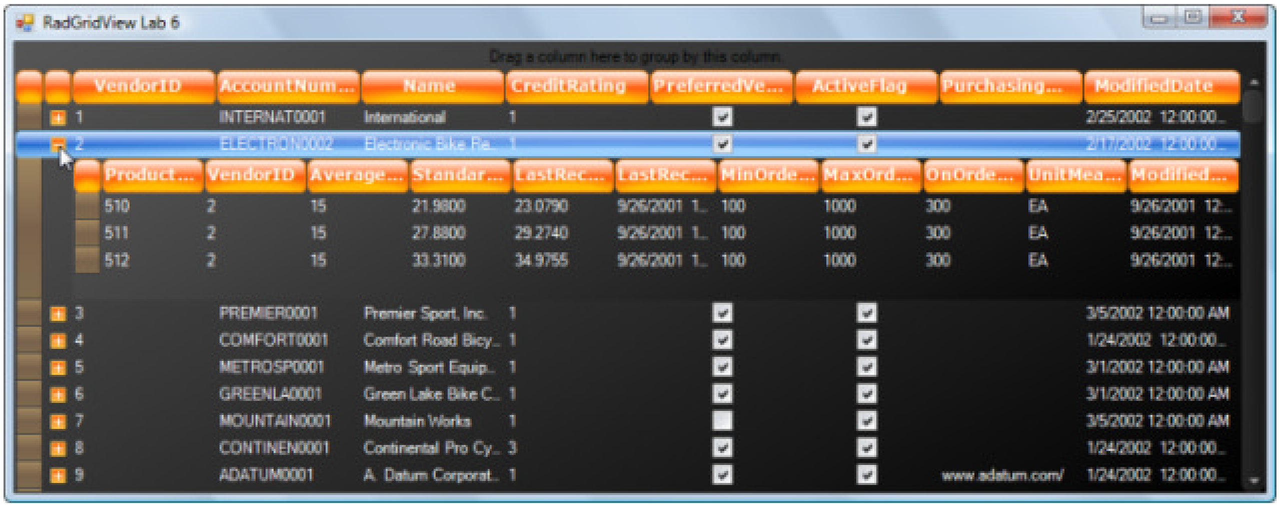
[VB] Loading the DataSet

```
Private Sub RadGridViewLab6_Load(ByVal sender As Object, ByVal e As EventArgs)
 Dim vendorTA As New AdventureWorksDataSetTableAdapters.VendorTableAdapter()
 Dim productvendorTA As New AdventureWorksDataSetTableAdapters.ProductVendorTableAdapter
 vendorTA.Fill(adventureWorksDataSet.Vendor)
 productvendorTA.Fill(adventureWorksDataSet.ProductVendor)
End Sub
```

[C#] Loading the DataSet

```
private void RadGridViewLab6_Load(object sender, EventArgs e)
{
 AdventureWorksDataSetTableAdapters.VendorTableAdapter vendorTA =
   new AdventureWorksDataSetTableAdapters.VendorTableAdapter();
 AdventureWorksDataSetTableAdapters.ProductVendorTableAdapter productvendorTA =
   new AdventureWorksDataSetTableAdapters.ProductVendorTableAdapter();
 vendorTA.Fill(adventureWorksDataSet.Vendor);
 productvendorTA.Fill(adventureWorksDataSet.ProductVendor);
}
```

10. Run the project and test the hiearchy functionality:



VendorID	AccountNum...	Name	CreditRating	PreferredVe...	ActiveFlag	Purchasing...	ModifiedDate
1	INTERNAT0001	International	1	☑	☑		2/25/2002 12:00:00...
2	ELECTRON0002	Electronic Bike Re...	1	☑	☑		2/17/2002 12:00:00...
3	PREMIER0001	Premier Sport, Inc.	1	☑	☑		3/5/2002 12:00:00 AM
4	COMFORT0001	Comfort Road Bicy...	1	☑	☑		1/24/2002 12:00:00...
5	METROSP0001	Metro Sport Equip...	1	☑	☑		3/1/2002 12:00:00 AM
6	GREENLA0001	Green Lake Bike C...	1	☑	☑		3/1/2002 12:00:00 AM
7	MOUNTAIN0001	Mountain Works	1	☐	☑		3/5/2002 12:00:00 AM
8	CONTINEN0001	Continental Pro Cy...	3	☑	☑		1/24/2002 12:00:00...
9	ADATUM0001	A. Datum Corporat...	1	☑	☑	www.adatum.com/	1/24/2002 12:00:00...

Product...	VendorID	Average...	Standar...	LastRec...	LastRec...	MinOrde...	MaxOrd...	OnOrde...	UnitMea...	Modified...
510	2	15	21.9800	23.0790	9/26/2001 1...	100	1000	300	EA	9/26/2001 12...
511	2	15	27.8800	29.2740	9/26/2001 1...	100	1000	300	EA	9/26/2001 12...
512	2	15	33.3100	34.9755	9/26/2001 1...	100	1000	300	EA	9/26/2001 12...

Notice that the table of Vendors is the parent table, and by expanding a Vendor Record, we can view a child table displaying the Products associated with that Vendor from the ProductsVendor table.

For columns that expand to fill the form, set the AutoSizeColumnsMode property of both the parent and child tables to "Fill".

Programmatic Hierarchical Table Setup

You can find the complete source for this project at:

\VS Projects\Grid\<VB|CS>\RadGridView\07_HierarchyCode

In the final lab on hierarchical tables, we will construct the hierarchy programmatically in the codebehind for the form itself.

1. Create a new Windows Forms Project.
2. Drag a RadGridView control onto the form. You can also drag a Miscellaneous Theme object if you wish to use the VistaOrange theme.
3. We will be doing all the setup for the RadGridView control itself programmatically, but first we still have to create a Dataset in our project to connect. Again, use the RadGridView's Smart Tag to create a new project DataSource.
4. Use the AdventureWorks database, and this time select both the Sales.SalesTerritory and Sales.

SalesPerson tables. Note the name of your new Dataset: again, ours is named AdventureWorksDataSet. **Note**: After creating the Dataset, be sure to leave the RadGridView's DataSource set to "none", as we will be setting this in code.

5. Add a reference to the Telerik.Winforms.UI namespace in the "Imports" (VB) or "uses" (C#) clause of the code.

6. Add the code below to the form's Load event handler. This will create the DataSet and table adapter object instances, then load the table data:

[VB] Initializing the Data Access

```
Private Sub RadGridViewLab7_Load(ByVal sender As Object, ByVal e As EventArgs)
 Dim adventureWorksDS As New AdventureWorksDataSet()
 Dim salesPersonTA As New AdventureWorksDataSetTableAdapters.SalesPersonTableAdapter()
 Dim salesTerritoryTA As New AdventureWorksDataSetTableAdapters.SalesTerritoryTableAdapt
()
 salesTerritoryTA.Fill(adventureWorksDS.SalesTerritory)
 salesPersonTA.Fill(adventureWorksDS.SalesPerson)
'. . .
End Sub
```

[C#] Initializing the Data Access

```
private void RadGridViewLab7_Load(object sender, EventArgs e)
{
 AdventureWorksDataSet adventureWorksDS = new AdventureWorksDataSet();
 AdventureWorksDataSetTableAdapters.SalesPersonTableAdapter salesPersonTA =
   new AdventureWorksDataSetTableAdapters.SalesPersonTableAdapter();
 AdventureWorksDataSetTableAdapters.SalesTerritoryTableAdapter salesTerritoryTA =
   new AdventureWorksDataSetTableAdapters.SalesTerritoryTableAdapter();
 salesTerritoryTA.Fill(adventureWorksDS.SalesTerritory);
 salesPersonTA.Fill(adventureWorksDS.SalesPerson);
 //. . .
}
```

7. Add code to the end of the form's Load event handler to configure the master grid view.

   This code assigns the DataSource for the master grid view and also performs some miscellaneous housekeeping to configure how columns are sized and to disallow adding new rows.

[VB] Configuring the Master Grid View Template

```
Private Sub RadGridViewLab7_Load(ByVal sender As Object, ByVal e As EventArgs)
'. . .
 radGridView1.DataSource = adventureWorksDS.SalesTerritory
 radGridView1.MasterGridViewTemplate.AutoSizeColumnsMode = GridViewAutoSizeColumnsMode.F
 radGridView1.MasterGridViewTemplate.AllowAddNewRow = False
'. . .
End Sub
```

[C#] Configuring the Master Grid View Template

```
private void RadGridViewLab7_Load(object sender, EventArgs e)
{
 //. . .
 radGridView1.DataSource = adventureWorksDS.SalesTerritory;
 radGridView1.MasterGridViewTemplate.AutoSizeColumnsMode =
GridViewAutoSizeColumnsMode.Fill;
 radGridView1.MasterGridViewTemplate.AllowAddNewRow = false;
 //. . .
}
```

8. Add code to the end of the form's Load event handler to configure the child grid view.

   This code creates a GridViewTemplate object instance to represent the child view, assigns the SalesPerson table as the DataSource and adds the template to the master grid view's ChildGridViewTemplates collection.

   [VB] Configuring the Child Grid View Template

```
Private Sub RadGridViewLab7_Load(ByVal sender As Object, ByVal e As EventArgs)
'. . .
 Dim childTmpt As New GridViewTemplate()
 childTmpt.DataSource = adventureWorksDS.SalesPerson
 childTmpt.AutoSizeColumnsMode = GridViewAutoSizeColumnsMode.Fill
 childTmpt.AllowAddNewRow = False
 radGridView1.MasterGridViewTemplate.ChildGridViewTemplates.Add(childTmpt)
'. . .
End Sub
```

   [C#] Configuring the Child Grid View Template

```
private void RadGridViewLab7_Load(object sender, EventArgs e)
{
 //. . .
 GridViewTemplate childTmpt = new GridViewTemplate();
 childTmpt.DataSource = adventureWorksDS.SalesPerson;
 childTmpt.AutoSizeColumnsMode = GridViewAutoSizeColumnsMode.Fill;
 childTmpt.AllowAddNewRow = false;
 radGridView1.MasterGridViewTemplate.ChildGridViewTemplates.Add(childTmpt);
 //. . .
}
```

9. Now that our parent and child templates are in place, we need to set a relationship between them. Add the following code, which relates the tables using the TerritoryID field, to the end of the Load handler.

   [VB] Configuring the Child Grid View Template

```
Private Sub RadGridViewLab7_Load(ByVal sender As Object, ByVal e As EventArgs)
'. . .
 Dim relation As New GridViewRelation(radGridView1.MasterGridViewTemplate)
 relation.ChildTemplate = childTmpt
 relation.RelationName = "SalesTerritoryPerson"
 relation.ParentColumnNames.Add("TerritoryID")
 relation.ChildColumnNames.Add("TerritoryID")
 radGridView1.Relations.Add(relation)

'. . .
End Sub
```

   [C#] Configuring the Child Grid View Template

```
private void RadGridViewLab7_Load(object sender, EventArgs e)
{
 //. . .
 GridViewRelation relation = new GridViewRelation(radGridView1.MasterGridViewTemplate);
 relation.ChildTemplate = childTmpt;
 relation.RelationName = "SalesTerritoryPerson";
 relation.ParentColumnNames.Add("TerritoryID");
 relation.ChildColumnNames.Add("TerritoryID");
 radGridView1.Relations.Add(relation);
 //. . .
}
```

10. Finally, add code to the end of the form's Load handler to hide the columns containing GUID's and set the theme name for the grid:

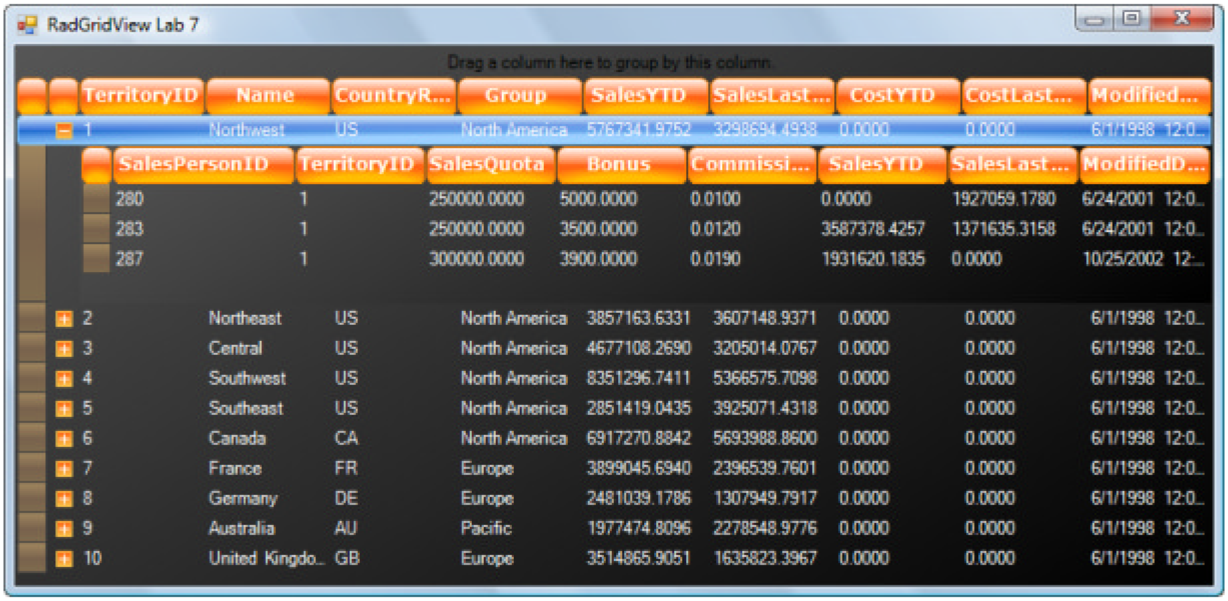
    [VB] Configuring the Child Grid View Template

    ```
    Private Sub RadGridViewLab7_Load(ByVal sender As Object, ByVal e As EventArgs)
    '. . .
     radGridView1.Columns("rowguid").IsVisible = False
     childTmpt.Columns("rowguid").IsVisible = False
     radGridView1.ThemeName = "VistaOrange"
    End Sub
    ```

    [C#] Configuring the Child Grid View Template

    ```
    private void RadGridViewLab7_Load(object sender, EventArgs e)
    {
     //. . .
     radGridView1.Columns["rowguid"].IsVisible = false;
     childTmpt.Columns["rowguid"].IsVisible = false;
     radGridView1.ThemeName = "VistaOrange";
    }
    ```

11. Run the project and expand a Territory record to view the SalesPerson records with matching TerritoriyID values.



TerritoryID	Name	CountryR...	Group	SalesYTD	SalesLast...	CostYTD	CostLast...	Modified...
1	Northwest	US	North America	5767341.9752	3298694.4938	0.0000	0.0000	6/1/1998 12:0...
2	Northeast	US	North America	3857163.6331	3607148.9371	0.0000	0.0000	6/1/1998 12:0...
3	Central	US	North America	4677108.2690	3205014.0767	0.0000	0.0000	6/1/1998 12:0...
4	Southwest	US	North America	8351296.7411	5366575.7098	0.0000	0.0000	6/1/1998 12:0...
5	Southeast	US	North America	2851419.0435	3925071.4318	0.0000	0.0000	6/1/1998 12:0...
6	Canada	CA	North America	6917270.8842	5693988.8600	0.0000	0.0000	6/1/1998 12:0...
7	France	FR	Europe	3899045.6940	2396539.7601	0.0000	0.0000	6/1/1998 12:0...
8	Germany	DE	Europe	2481039.1786	1307949.7917	0.0000	0.0000	6/1/1998 12:0...
9	Australia	AU	Pacific	1977474.8096	2278548.9776	0.0000	0.0000	6/1/1998 12:0...
10	United Kingdo...	GB	Europe	3514865.9051	1635823.3967	0.0000	0.0000	6/1/1998 12:0...

SalesPersonID	TerritoryID	SalesQuota	Bonus	Commissi...	SalesYTD	SalesLast...	ModifiedD...
280	1	250000.0000	5000.0000	0.0100	0.0000	1927059.1780	6/24/2001 12:0...
283	1	250000.0000	3500.0000	0.0120	3587378.4257	1371635.3158	6/24/2001 12:0...
287	1	300000.0000	3900.0000	0.0190	1931620.1835	0.0000	10/25/2002 12:...

1.7 Virtual Mode

"Virtual Mode" provides a way to explicitly implement the data management of your RadGridView control. This is especially useful when binding to large groups of data, because it can let you only load the data currently being used, thus improving the performance of the grid. Virtual mode is also necessary when bound and unbound columns are used together, but sorted by the bound column’s values.

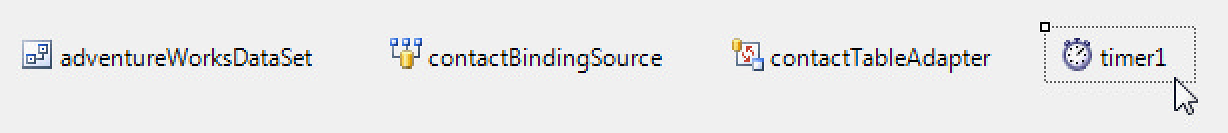
In the following simplistic example of Virtual Mode, the RadGridView receives its data by calling the CellValueNeeded event handler. To allow us to see the updates, we will use a table of constantly changing randomly selected employee records as the RadGridView’s contents.

You can find the complete source for this project at:

\VS Projects\Grid\<VB|CS>\RadGridView\08_VirtualMode

1. Create a new Windows Forms Project.
2. Drag a RadGridView control onto the form, as well as a Timer object that will be used to control our data updates. Set the Timer **Interval** property to "100".
3. We will be doing all the setup for the RadGridView control itself programmatically, but first we still have to create a Dataset in our project to connect. Again, use the RadGridView’s Smart Tag to create a new project DataSource.
4. Use the AdventureWorks database, and select the Person.Contact table only. We will again be using the default name AdventureWorksDataSet. **Note**: After creating the Dataset, be sure to leave the RadGridView’s DataSource set to "none", as we will be using Virtual Mode to provide the data contents.

   Notice that now you have an adventureWorksDataSet object, along with a corresponding Binding Source and Table Adapter in the component tray. The Timer component will also appear in the tray.



Code to populate the DataSet using the Table Adapter is automatically placed in the Form’s Load handler. Do not remove this code!

5. Add the following declarations to the Form class, directly above the constructor method. These lines declare the numbers of columns and rows for our data, along with the List of string Lists that will hold the current data.

[VB] Declaring Private Variables

```
Private ContactTable As New List(Of List(Of String))()
Private NumberOfRows As Integer = 20
Private NumberOfCols As Integer = 4
```

[C#] Declaring Private Variables

```
private List<List<string>> ContactTable = new List<List<string>>();
private int NumberOfRows = 20;
private int NumberOfCols = 4;
```

6. We also need to create a method which will update the data in our ContactTable. Add the following method to the Form’s class.

   This method uses the current time’s tick value as the seed to generate a new random number from 0 to 1000; then uses that index value to pull a Contact record from the dataset and add it to the table. The method will be called each time our Timer object’s Tick event is triggered.

[VB] Refresh the Contact Table

```
Private Sub RefreshContactData()
 Dim random As New Random(DirectCast(DateTime.Now.Ticks, Integer))
 Dim i As Integer = 0
 While i < NumberOfRows
  Dim index As Integer = random.[Next](1000)
  Dim cr As AdventureWorksDataSet.ContactRow = adventureWorksDataSet.Contact(index)
  ContactTable(i)(0) = cr.FirstName
  ContactTable(i)(1) = cr.LastName
  ContactTable(i)(2) = cr.EmailAddress
  ContactTable(i)(3) = cr.Phone
  System.Math.Max(System.Threading.Interlocked.Increment(i),i - 1)
 End While
End Sub
```

[C#] Refresh the Contact Table

```
private void RefreshContactData()
{
 Random random = new Random((int)DateTime.Now.Ticks);
 for (int i = 0; i < NumberOfRows; i++)
 {
   int index = random.Next(1000);
   AdventureWorksDataSet.ContactRow cr =
     adventureWorksDataSet.Contact[index];
   ContactTable[i][0] = cr.FirstName;
   ContactTable[i][1] = cr.LastName;
   ContactTable[i][2] = cr.EmailAddress;
   ContactTable[i][3] = cr.Phone;
 }
}
```

7. In the form's Load event handler we will set some properties of the RadGridView control. To simplify our example we will not allow editing, sorting, or filtering on the grid. If they were allowed, we would simply need to implement more functionality for our grid in Virtual Mode. Add the following code to the start of the Load event handler, just below the automatically generated statement that loads the Contact table.

[VB] Initialize Grid, Master Grid View

```
Private Sub RadGridViewLab8_Load(ByVal sender As Object, ByVal e As EventArgs) Handles
MyBase.Load
  Me.contactTableAdapter.Fill(Me.adventureWorksDataSet.Contact)
  ' set grid properties
  radGridView1.MasterGridViewTemplate.AllowAddNewRow = False
  radGridView1.MasterGridViewTemplate.AllowCellContextMenu = False
  radGridView1.MasterGridViewTemplate.AllowDeleteRow = False
  radGridView1.MasterGridViewTemplate.AllowEditRow = False
  radGridView1.EnableSorting = False
  radGridView1.EnableFiltering = False
  radGridView1.EnableGrouping = False
  '. . .
}
```

[C#] Initialize Grid and Master Grid View

```
private void RadGridViewLab8_Load(object sender, EventArgs e)
{
 this.contactTableAdapter.Fill(this.adventureWorksDataSet.Contact);
 // set grid properties
 radGridView1.MasterGridViewTemplate.AllowAddNewRow = false;
 radGridView1.MasterGridViewTemplate.AllowCellContextMenu = false;
 radGridView1.MasterGridViewTemplate.AllowDeleteRow = false;
 radGridView1.MasterGridViewTemplate.AllowEditRow = false;
 radGridView1.EnableSorting = false;
 radGridView1.EnableFiltering = false;
 radGridView1.EnableGrouping = false;
 //. . .
}
```

> The grid and master table view properties could also easily be set within the Property Builder or the Properties Window.

8. We need to initialize our ContactTable variable, by adding the following lines of code to the form's Load event handler.

   The "ContactTable" generic list is loaded with yet other generic lists that each contain four empty strings. The four empty strings will be loaded later with the four columns worth of contact data.

   [VB] Initialize "ContactTable"

```
Private Sub RadGridViewLab8_Load(ByVal sender As Object, ByVal e As EventArgs) Handles
MyBase.Load
  ' . . .
  For i As Integer = 0 To NumberOfRows - 1
    Dim list As List(Of String) = New List(Of String)(New String() {String.Empty,
String.Empty, String.Empty, String.Empty})
    ContactTable.Add(list)
  Next i
  '. . .
}
```

   [C#] Initialize "ContactTable"

```
private void RadGridViewLab8_Load(object sender, EventArgs e)
{
 this.contactTableAdapter.Fill(this.adventureWorksDataSet.Contact);
 // . . .
 for (int i = 0; i < NumberOfRows; i++)
   ContactTable.Add(new List<string> { string.Empty, string.Empty, string.Empty,
string.Empty });
 //. . .
}
```

9. Add the code to set up the data columns and rows, size the columns, and start the timer. Add the following lines to the end of the Form’s Load handler to finish setting up the grid for Virtual Mode and start the timer.

   [VB] Setup Columns/Rows and Start Timer

```
Private Sub RadGridViewLab8_Load(ByVal sender As Object, ByVal e As EventArgs)
 '. . .
 radGridView1.VirtualMode = True
 radGridView1.ColumnCount = NumberOfCols
 radGridView1.Columns(0).HeaderText = "First Name"
 radGridView1.Columns(1).HeaderText = "Last Name"
 radGridView1.Columns(2).HeaderText = "Email"
 radGridView1.Columns(3).HeaderText = "Phone Number"
 radGridView1.RowCount = NumberOfRows
 radGridView1.MasterGridViewTemplate.AutoSizeColumnsMode =
Telerik.WinControls.UI.GridViewAutoSizeColumnsMode.Fill
```

```
 timer1.Start()
End Sub
```

[C#] Setup Columns/Rows and Start Timer

```
private void RadGridViewLab8_Load(object sender, EventArgs e)
{
 //. . .
 radGridView1.VirtualMode = true;
 radGridView1.ColumnCount = NumberOfCols;
 radGridView1.Columns[0].HeaderText = "First Name";
 radGridView1.Columns[1].HeaderText = "Last Name";
 radGridView1.Columns[2].HeaderText = "Email";
 radGridView1.Columns[3].HeaderText = "Phone Number";
 radGridView1.RowCount = NumberOfRows;
 radGridView1.MasterGridViewTemplate.AutoSizeColumnsMode =
Telerik.WinControls.UI.GridViewAutoSizeColumnsMode.Fill;
 timer1.Start();
}
```

The number of rows and columns must be explicitly set when using Virtual Mode so that the control can request the correct cells from the CellValueNeeded handler.

10. Back in the design view of the form, double-click the Timer component to create a Tick event handler. Add the code below to refresh the table and trigger the grid update. The call to Update() notifies the grid that the data has changed by passing the BatchDataChanged value.

[VB] Handling the Tick Event

```
Private Sub timer1_Tick(ByVal sender As Object, ByVal e As EventArgs)
 ' reload the contact table
 RefreshContactData()
 ' signal that the grid should be updated
 radGridView1.GridElement.Update
(Telerik.WinControls.UI.GridUINotifyAction.BatchDataChanged)
End Sub
```

[C#] Handling the Tick Event

```
private void timer1_Tick(object sender, EventArgs e)
{
 // reload the contact table
 RefreshContactData();
 // signal that the grid should be updated
 radGridView1.GridElement.Update
(Telerik.WinControls.UI.GridUINotifyAction.BatchDataChanged);
}
```

11. The call to GridElement.Update() in the previous step will precipitate a CellValueNeeded event. Back in the design view of the form, select the grid and in the Events tab (⚡) of the Properties window, double-click the CellValueNeeded event to create an event handler and add the code below.

The code supplies the cell value from the ContactTable using the row and column index passed in the GridViewCellValueEventArgs parameter.

[VB] Handling the CellValueNeeded Event

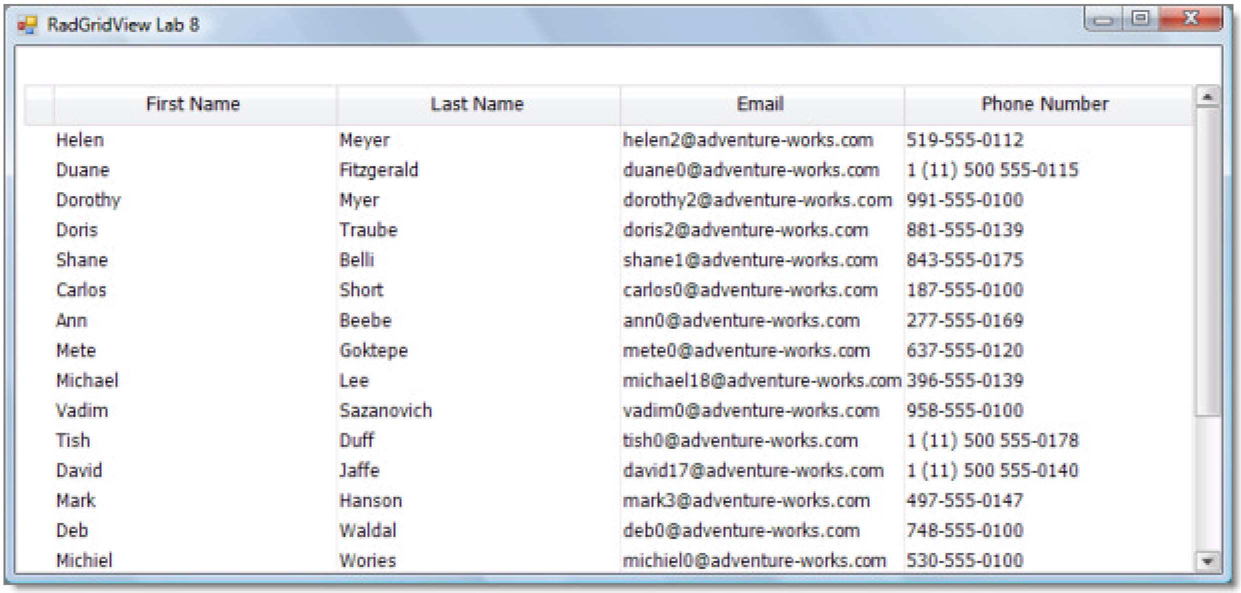
```
Private Sub radGridView1_CellValueNeeded_1(ByVal sender As Object, ByVal e As
Telerik.WinControls.UI.GridViewCellValueEventArgs)
 e.Value = ContactTable(e.RowIndex)(e.ColumnIndex)
End Sub
```

[C#] Handling the CellValueNeeded Event

```
private void radGridView1_CellValueNeeded_1(object sender,
Telerik.WinControls.UI.GridViewCellValueEventArgs e)
{
 e.Value = ContactTable[e.RowIndex][e.ColumnIndex];
}
```

> If you have allowed column re-ordering at runtime, changes to the column order will need to be compensated for in this handler.

12. Run the project to see the data updating automatically as the grid runs in Virtual Mode.



First Name	Last Name	Email	Phone Number
Helen	Meyer	helen2@adventure-works.com	519-555-0112
Duane	Fitzgerald	duane0@adventure-works.com	1 (11) 500 555-0115
Dorothy	Myer	dorothy2@adventure-works.com	991-555-0100
Doris	Traube	doris2@adventure-works.com	881-555-0139
Shane	Belli	shane1@adventure-works.com	843-555-0175
Carlos	Short	carlos0@adventure-works.com	187-555-0100
Ann	Beebe	ann0@adventure-works.com	277-555-0169
Mete	Goktepe	mete0@adventure-works.com	637-555-0120
Michael	Lee	michael18@adventure-works.com	396-555-0139
Vadim	Sazanovich	vadim0@adventure-works.com	958-555-0100
Tish	Duff	tish0@adventure-works.com	1 (11) 500 555-0178
David	Jaffe	david17@adventure-works.com	1 (11) 500 555-0140
Mark	Hanson	mark3@adventure-works.com	497-555-0147
Deb	Waldal	deb0@adventure-works.com	748-555-0100
Michiel	Wories	michiel0@adventure-works.com	530-555-0100

1.8 Using RadGridView with Dynamic LINQ Query Library

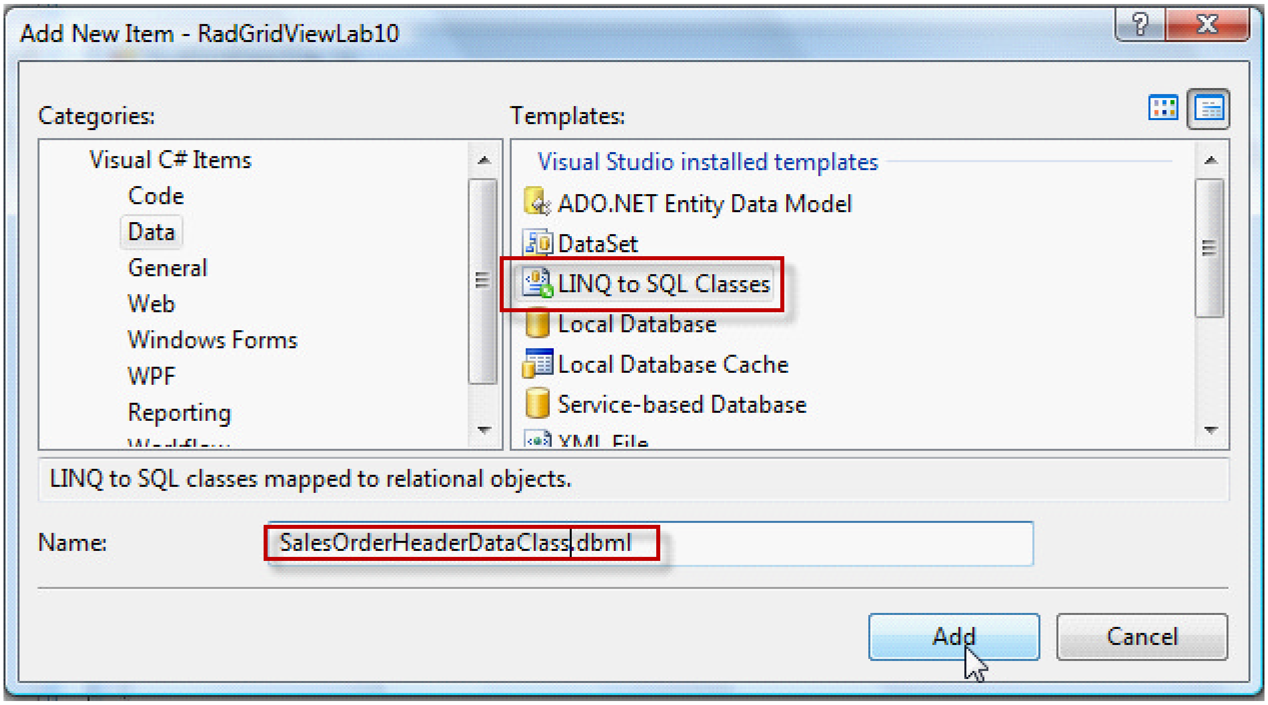
Using the Dynamic LINQ Query Library, the RadGridView control can implement dynamic paging, sorting, and filtering to millions of records, allowing unmatched performance because the record manipulation is done by dynamically created LINQ queries performed on the database itself.

For this lab, we will be using the Dynamic.cs class, which can be downloaded along with the CS or VB code samples here: **Visual Studio 2008 Samples (http://msdn.microsoft.com/en-us/bb330936.aspx)**

> You can find the complete source for this project at:
>
> \VS Projects\Grid\<VB|CS>\RadGridView\10_DynamicLINQLibrary

1. Create a new Windows Forms Project.
2. Add the Dynamic.cs file to the project directory, and use the **Project** | **Add** | **Add Existing Item** option to add to the project as well. This class will add the LINQ extensions we will be using to build our dynamic LINQ queries.

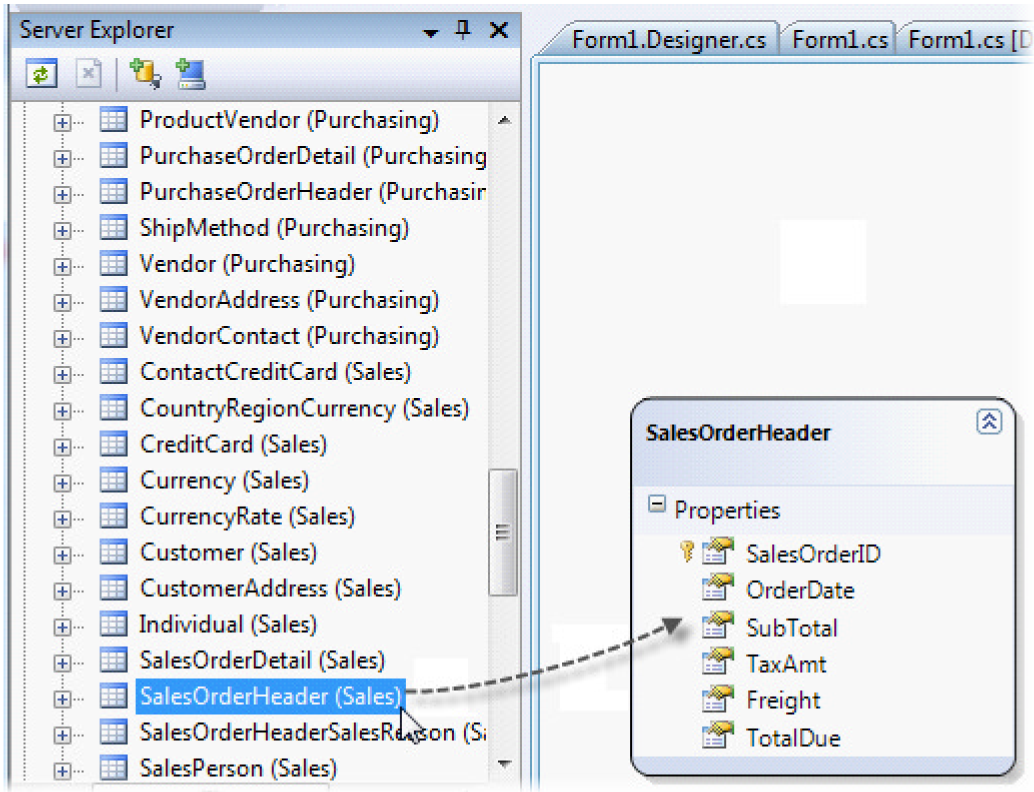
3. Use the **Project** | **Add** | **Add New Item** option to add a LINQ to SQL Data Class to the project. Name the class SalesOrderHeaderDataClass.dbml. This class will be the basis for our LINQ dynamic queries.



LINQ to SQL classes mapped to relational objects.

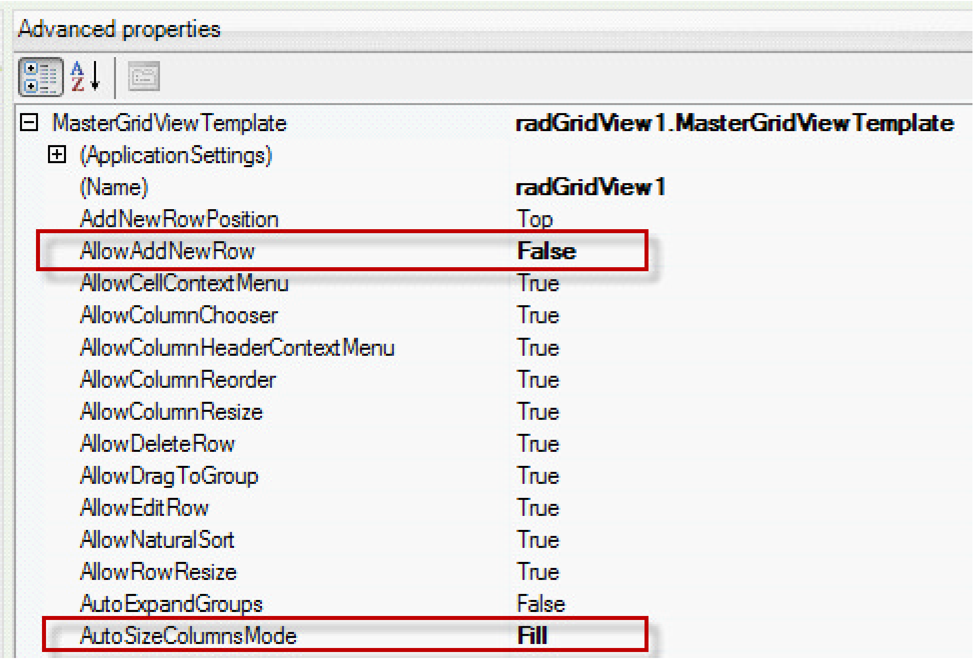
Name: SalesOrderHeaderDataClass.dbml

4. Next we will add the data connection to our LINQ to SQL class. While in the design view for the SalesOrderHeaderDataClass, expand the Server Explorer window of Visual Studio to the Sales.SalesOrderHeader table of the AdventureWorks database. You may need to open a connection to the database to view the tables. Then, drag the SalesOrderHeader table onto the design surface to add the table to the class.



Remove all but the following fields from the Table object: SalesOrderID, OrderDate, SubTotal, TaxAmt, Freight, and TotalDue.

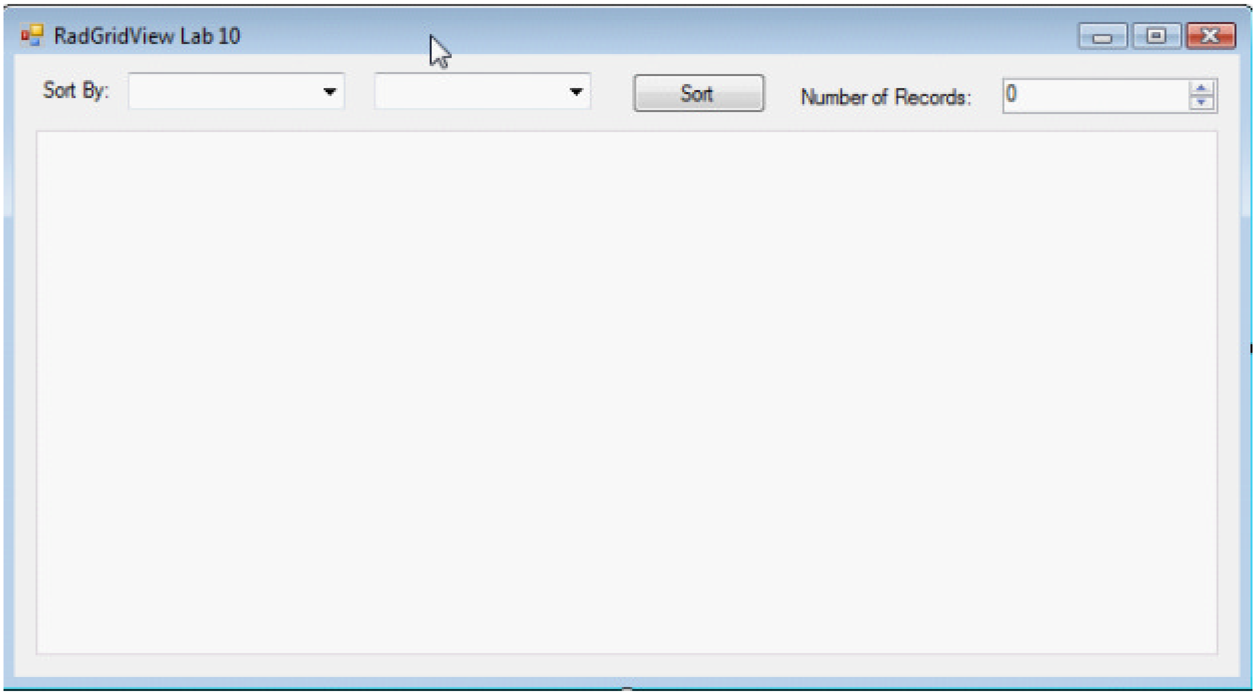
5. Now, to set up the user interface of our example, drag a RadGridView control onto the Form, and set the following properties of the MasterGridViewTemplate using the Property Builder.


Advanced properties

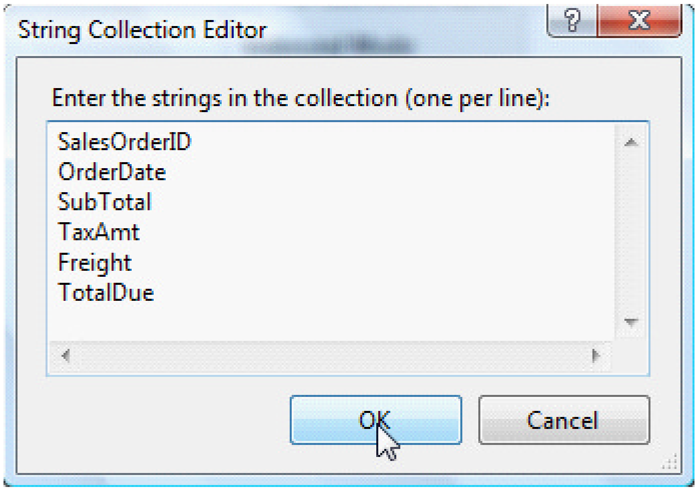
⊟ MasterGridViewTemplate	**radGridView1.MasterGridViewTemplate**
⊞ (ApplicationSettings)	
(Name)	**radGridView1**
AddNewRowPosition	Top
AllowAddNewRow	**False**
AllowCellContextMenu	True
AllowColumnChooser	True
AllowColumnHeaderContextMenu	True
AllowColumnReorder	True
AllowColumnResize	True
AllowDeleteRow	True
AllowDragToGroup	True
AllowEditRow	True
AllowNaturalSort	True
AllowRowResize	True
AutoExpandGroups	False
AutoSizeColumnsMode	**Fill**

6. Also drag onto the Form the following controls: two ComboBoxes, a NumberUpDown control, a Button, and two Labels. Arrange the controls similar to the layout shown below, and name the controls as follows, from

left to right: cbField, cbSortType, btnSort, and numRecords.

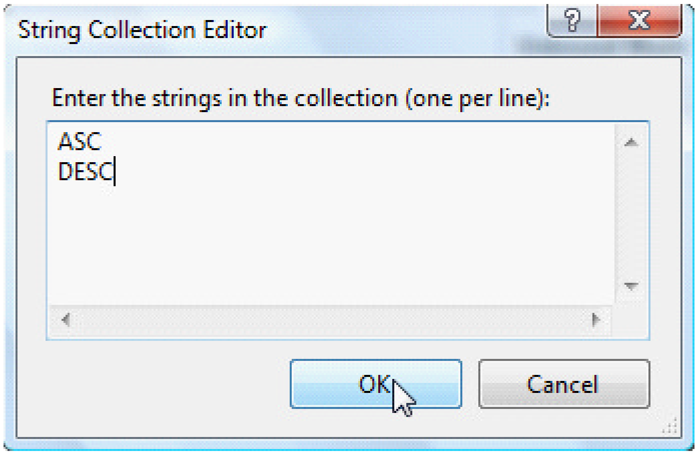


7. To populate the Field combobox, use the Smart Tag to open the Items list editor, and add the following Fields:



```
SalesOrderID
OrderDate
SubTotal
TaxAmt
Freight
TotalDue
```

8. Populate the SortType combobox as well, adding the following items:



ASC
DESC

9. To add a Click event handler for the Button control, double click the button in design view, and add the following code to the new event handler:

[VB] Loading Data Via LINQ

```
Private Sub btnSort_Click(ByVal sender As Object, ByVal e As EventArgs)
 Me.radGridView1.GridElement.BeginUpdate()
 Dim queryable As IQueryable = New SalesOrderHeaderDataClassDataContext
().SalesOrderHeaders.AsQueryable()
 queryable = queryable.OrderBy([String].Format("{0} {1}", cbField.Text, cbSortType.Text)
 radGridView1.DataSource = queryable.Take(Convert.ToInt32(numRecords.Value))
 Me.radGridView1.GridElement.EndUpdate(True)
End Sub
```

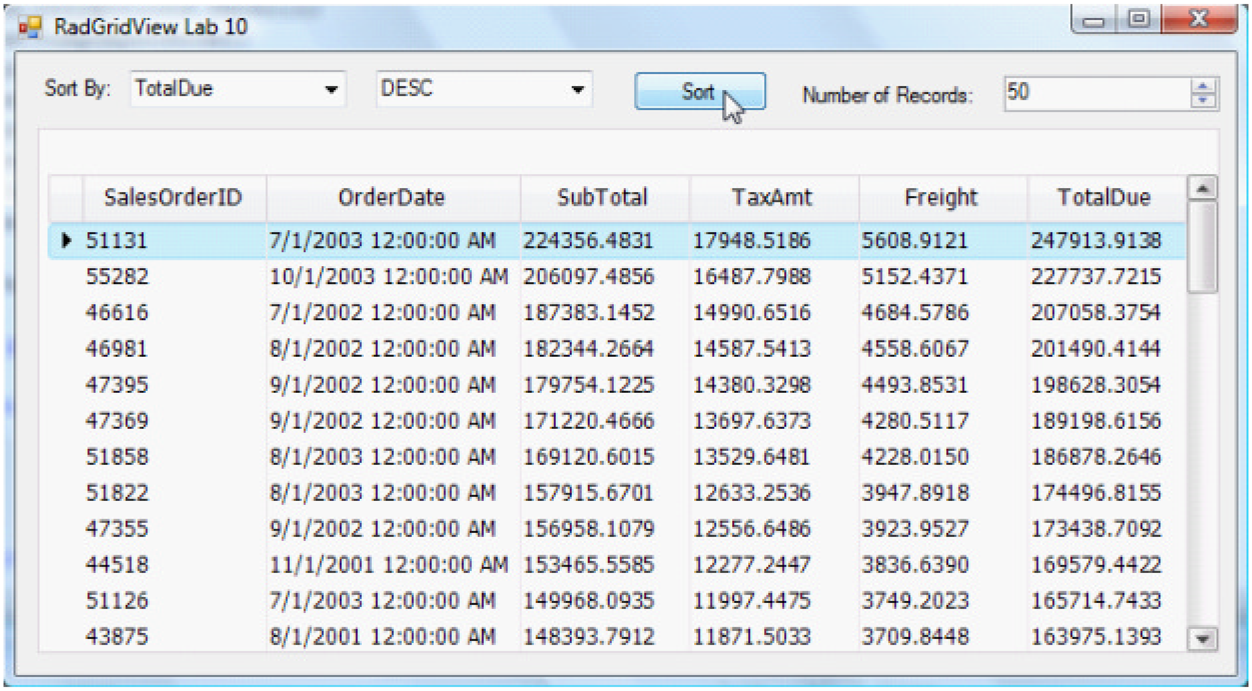
[C#] Loading Data Via LINQ

```
private void btnSort_Click(object sender, EventArgs e)
{
 this.radGridView1.GridElement.BeginUpdate();
 IQueryable queryable = new SalesOrderHeaderDataClassDataContext
().SalesOrderHeaders.AsQueryable();
 queryable = queryable.OrderBy(String.Format("{0} {1}",
 cbField.Text, cbSortType.Text));
 radGridView1.DataSource =
 queryable.Take(Convert.ToInt32(numRecords.Value));
 this.radGridView1.GridElement.EndUpdate(true);
}
```

This code temporarily disables updating on the grid, then assembles the new query from the options selected by the user, and uses the query to retrieve data from the table using the LINQ to SQL data class.

10. Add a reference to System.Linq.Dynamic in the "Imports" (VB) or "uses" (C#) section of the code.

11. Run the project to view the form. Select a Field and Sort direction from the comboboxes, and set the number of records to return. Click the Sort button to view the results, which are retrieved from the database using the dynamic query.



Sort By: TotalDue DESC Number of Records: 50

	SalesOrderID	OrderDate	SubTotal	TaxAmt	Freight	TotalDue
▶	51131	7/1/2003 12:00:00 AM	224356.4831	17948.5186	5608.9121	247913.9138
	55282	10/1/2003 12:00:00 AM	206097.4856	16487.7988	5152.4371	227737.7215
	46616	7/1/2002 12:00:00 AM	187383.1452	14990.6516	4684.5786	207058.3754
	46981	8/1/2002 12:00:00 AM	182344.2664	14587.5413	4558.6067	201490.4144
	47395	9/1/2002 12:00:00 AM	179754.1225	14380.3298	4493.8531	198628.3054
	47369	9/1/2002 12:00:00 AM	171220.4666	13697.6373	4280.5117	189198.6156
	51858	8/1/2003 12:00:00 AM	169120.6015	13529.6481	4228.0150	186878.2646
	51822	8/1/2003 12:00:00 AM	157915.6701	12633.2536	3947.8918	174496.8155
	47355	9/1/2002 12:00:00 AM	156958.1079	12556.6486	3923.9527	173438.7092
	44518	11/1/2001 12:00:00 AM	153465.5585	12277.2447	3836.6390	169579.4422
	51126	7/1/2003 12:00:00 AM	149968.0935	11997.4475	3749.2023	165714.7433
	43875	8/1/2001 12:00:00 AM	148393.7912	11871.5033	3709.8448	163975.1393

Even though the actual data table is over 30,000 records, the grid is only loading the values retrieved by the query, and so maintains high performance even with very large record sets. This same technique can be used to implement custom paging and filtering on the RadGridView control.

1.9 Exporting RadGridView Data

Exporting Data

Telerik's RadGridView control makes for simple data exports to multiple formats including Microsoft Excel and Telerik Reports. RadGridView can be exported to Excel using either the ExcelML format, which does not even require an Office installation on the machine, or through the Primary Interop Assemblies (PIA).

1.9.1 Exporting to Excel

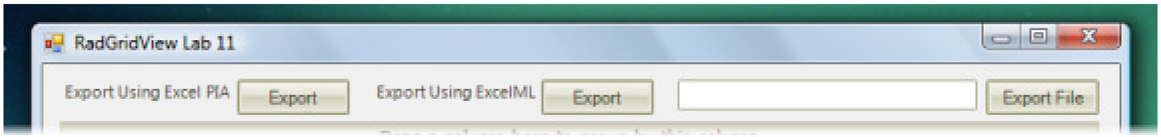
There are some tradeoffs between the two methods of exporting. Using PIA's you can add a progress event handler, but the PIA route appears to be less performant than using ExcelML. When testing you may want to reduce the number of records you're working with in the dataset, or use the RadGrid's filters to subset the data. The ExcelML export takes advantage of the RadGridView's conditional formatting capability so that formatting in the grid is passed on to the exported spreadsheet.

This lab will demonstrate some of the exporting capabilities of the RadGridView control, some conditional formatting, and both types of Excel exporting.

You can find the complete source for this project at:

\VS Projects\Grid\<VB|CS>\RadGridView\11_ExportExcel

1. Create a new Windows Forms Project.
2. Drag a RadGridView control onto the form, and also add some additional controls to the top of the form, in a layout similar to the one shown below. You can use RadLabel, RadButton and RadTextbox or the standard Label, Button and TextBox equivelents.



3. Name the new controls (from left to right): "btnExcelPIA", "btnExcelML", "tbFileName", and "btnExportFile".
4. Below the grid add a RadStatusStrip control. Using the status strip Smart Tag add a ProgressBarElement and a LabelElement. Name them "pbStatus" and "lblStatus" respectively.
5. Using the Smart Tag on the RadGridView control, add a new project datasource that connects the RadGridView to the Person.Contact table of the AdventureWorks database.
6. Also using the Smart Tag, open the Property Builder of the RadGridView control, and disable viewing of the following fields on the Person.Contact table: NameStyle, Title, EmailPromotion, PasswordHash, PasswordSalt, AdditionalContactInfo, rowguid, and ModifiedDate.
7. Set the following properties of the RadGridView control's MasterGridView Template in the Property Builder:
   - AllowAddNewRow = False
   - EnableFiltering = True
   - AutoSizeColumnsMode = Fill
8. Close the Property Builder, and navigate to the form's code-behind file. You will need to add the following using declarations to the code file, and also add the corresponding .dll references to the project.

[VB] Adding References

```
Imports Telerik.Data
Imports Telerik.WinControls.UI
Imports Telerik.WinControls.UI.Export
```

[C#] Adding References

```
using Telerik.Data;
using Telerik.WinControls.UI;
using Telerik.WinControls.UI.Export;
```

9. In Design View, double-click the "Export File" button to add a new click event handler, which will contain the code to open a File dialog to set the name and location of the Excel file where the data will be exported. Notice we have set the filter to be for Excel files only.

[VB] Populating the Export File Textbox

```
Private Sub btnExportFile_Click(ByVal sender As Object, ByVal e As EventArgs)
 ' populate the file name textbox using the "save" dialog
 Dim sfd As New SaveFileDialog()
 sfd.Filter = [String].Format("{0} (*{1})|*{1}", "Excel Files", ".xls")
 If sfd.ShowDialog() = DialogResult.OK Then
  tbFileName.Text = sfd.FileName
 End If
End Sub
```

[C#] Populating the Export File Textbox

```
private void btnExportFile_Click(object sender, EventArgs e)
{
 // populate the file name textbox using the "save" dialog
 SaveFileDialog sfd = new SaveFileDialog();
 sfd.Filter = String.Format("{0} (*{1})|*{1}",
     "Excel Files",
     ".xls");
 if (sfd.ShowDialog() == DialogResult.OK)
   tbFileName.Text = sfd.FileName;
}
```

10. Also add event handlers for the ExcelPIA button click. Notice that the RadGridViewExcelExporter object from the Telerik.Data namespace allows an optional "Progress" event to be hooked up.

The ProgressValue property available from the ProgressEventArgs passed into the event contains an integer with the percentage of completion. For example, if the grid has 4 records, the first time the Progress event is fired, ProgressValue will be "25" (i.e. 25%).

To get the number of records being exported, use the current GridViewTemplate's RowCount property. Unlike attempting to use the table's Row.Count, this will take grid filtering into account. In the example below we're using the grid's MasterGridViewTemplate.RowCount.

[VB] Handle the Click Event for the PIA Export

```
Private Sub btnExcelPIA_Click(ByVal sender As Object, ByVal e As EventArgs)
 If Not tbFileName.Text.Equals(String.Empty) Then
  Dim exporter As New RadGridViewExcelExporter()
  ' assign a progress handling event that will update the status bar
  AddHandler exporter.Progress, AddressOf exporter_Progress
  ' set the progress bar max to the number of rows in the dataset
  pbStatus.Maximum = radGridView1.MasterGridViewTemplate.RowCount + 1
  ' trigger the export
  exporter.Export(radGridView1, tbFileName.Text, "Sheet1")
  MessageBox.Show("Export Complete!")
  ' reset the progress bar and label
  pbStatus.Value1 = 0
  lblStatus.Text = [String].Empty
 End If
End Sub
Sub exporter_Progress(ByVal sender As Object, ByVal e As ProgressEventArgs)
 Dim rows As Integer = radGridView1.MasterGridViewTemplate.RowCount
 pbStatus.Value1 = DirectCast(((e.ProgressValue * 0.01) * rows), Integer)
 ' show the number or records processed and progress bar
 lblStatus.Text = [String].Format("{0} of {1}", pbStatus.Value1, rows)
End Sub
```

[C#] Handle the Click Event for the PIA Export

```
private void btnExcelPIA_Click(object sender, EventArgs e)
{
 if (!tbFileName.Text.Equals(string.Empty))
 {
   RadGridViewExcelExporter exporter = new RadGridViewExcelExporter();
   // assign a progress handling event that will update the status bar
   exporter.Progress += new ProgressHandler(exporter_Progress);
   // set the progress bar max to the number of rows in the dataset
   pbStatus.Maximum = radGridView1.MasterGridViewTemplate.RowCount + 1;
   // trigger the export
   exporter.Export(radGridView1, tbFileName.Text, "Sheet1");
   MessageBox.Show("Export Complete!");
   // reset the progress bar and label
   pbStatus.Value1 = 0;
   lblStatus.Text = String.Empty;
 }
}
void exporter_Progress(object sender, ProgressEventArgs e)
{
 int rows = radGridView1.MasterGridViewTemplate.RowCount;
 pbStatus.Value1 = (int)((e.ProgressValue * .01) * rows);
 // show the number or records processed and progress bar
 lblStatus.Text = String.Format("{0} of {1}",
   pbStatus.Value1, rows);
}
```

11. Handle the click for the ExcelML export button. This export uses the ExportToExcelML object from the Telerik.WinControls.UI.Export namespace. Notice that for the ExcelML exporting, we are adding a conditional format object to the Phone field of the table, where those numbers starting with the digits 396 will be highlighted in a yellow color.

[VB] Handle the Click Event for the ExcelML Export

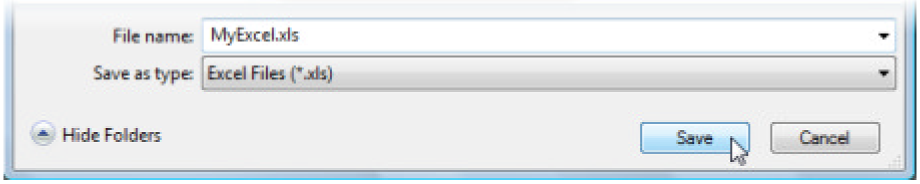
```
Private Sub btnExcelML_Click(ByVal sender As Object, ByVal e As EventArgs)
 If Not tbFileName.Text.Equals(String.Empty) Then
  ' configure conditional formatting where rows with phone numbers starting with "396" ar
shaded yellow
  Dim cfo As New ConditionalFormattingObject("MyConditionalFormatting",
ConditionTypes.StartsWith, "396", "", True)
  cfo.RowBackColor = Color.Yellow
  radGridView1.Columns("Phone").ConditionalFormattingObjectList.Add(cfo)
  ' create and trigger the export
  Dim exporter As New ExportToExcelML()
  exporter.RunExport(radGridView1, tbFileName.Text, ExportToExcelML.ExcelMaxRows._65536,
True)
  MessageBox.Show("Export Complete!")
 End If
End Sub
```

[C#] Handle the Click Event for the ExcelML Export

```
private void btnExcelML_Click(object sender, EventArgs e)
{
 if (!tbFileName.Text.Equals(string.Empty))
 {
   // configure conditional formatting where rows with phone numbers starting with "396"
are shaded yellow
   ConditionalFormattingObject cfo = new ConditionalFormattingObject
("MyConditionalFormatting",
     ConditionTypes.StartsWith, "396", "", true);
   cfo.RowBackColor = Color.Yellow;
   radGridView1.Columns["Phone"].ConditionalFormattingObjectList.Add(cfo);
   // create and trigger the export
   ExportToExcelML exporter = new ExportToExcelML();
   exporter.RunExport(radGridView1, tbFileName.Text, ExportToExcelML.ExcelMaxRows._65536,
true);
   MessageBox.Show("Export Complete!");
 }
}
```

12. Save the project and set it as the Startup Project, then run to view the Exporting Form. To run a simple

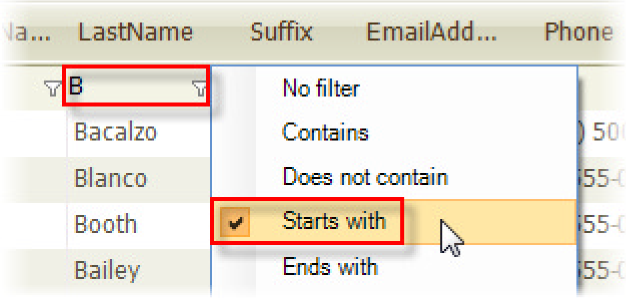
export, click the Export File button, and in the dialog window, choose a location and file name for the exported file. **Note:** Be sure to add the .xls file extension to the file name.



File name: MyExcel.xls

Save as type: Excel Files (*.xls)

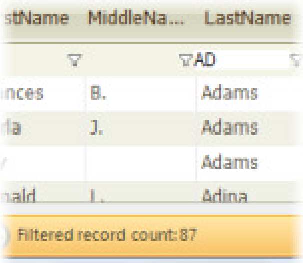
13. To give us a smaller set of records to view in Excel, add a “Starts with” filter using the letter "B" on the LastName field of the table.



la...	LastName	Suffix	EmailAdd...	Phone
	B			
	Bacalzo			50
	Blanco			55-
	Booth			55-
	Bailey			55-

This step is not necessary for functionality, but for demonstration purposes makes the exported file smaller and easier to view.

If you want to know the current number of records after the data has been filtered, handle the grid's FilterChanged event. This example displays the filtered record count in the status strip:



stName	MiddleNa...	LastName
		AD
nces	B.	Adams
la	J.	Adams
/		Adams
nald	L.	Adina

Filtered record count: 87

The arguments passed in to the event include the current GridViewTemplate. Use the template's RowCount to get the number of records after filtering:

[VB] Handling the FilterChanged Event

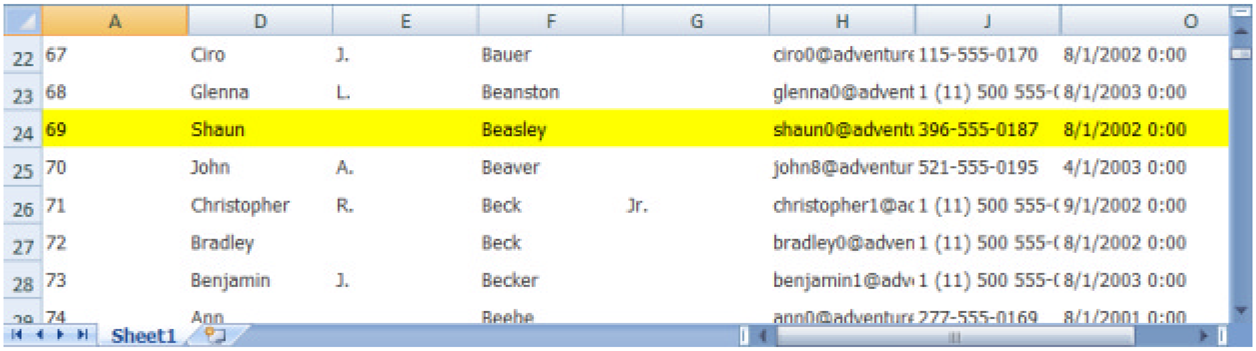
```
Private Sub radGridView1_FilterChanged(ByVal sender As Object, ByVal e As
GridViewCollectionChangedEventArgs)
 lblStatus.Text = [String].Format("Filtered record count: {0}",
e.GridViewTemplate.RowCount)
End Sub
```

[C#] Handling the FilterChanged Event

```
private void radGridView1_FilterChanged(object sender,
GridViewCollectionChangedEventArgs e)
```

```
          {
           lblStatus.Text = String.Format("Filtered record count: {0}",
          e.GridViewTemplate.RowCount);
          }
```

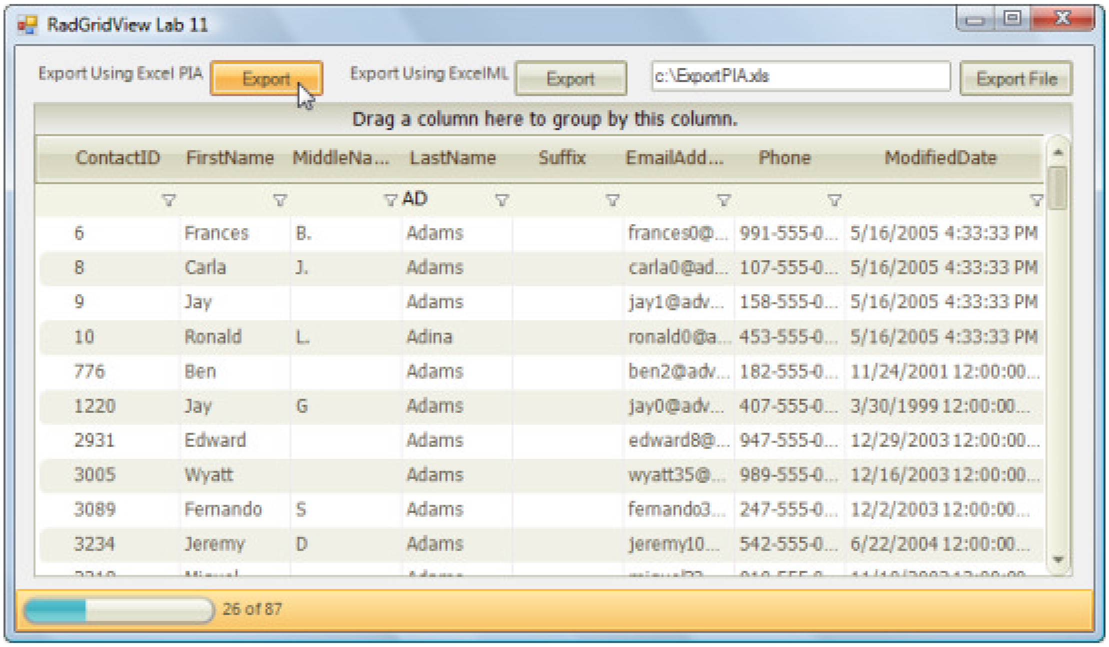
14. Click the Export button that uses the ExcelML method and wait for the completion notification. Open the newly exported file, and notice that our conditional formatting has highlighted those records with phone numbers matching our criteria.



	A	D	E	F	G	H	J	O
22	67	Ciro	J.	Bauer		ciro0@adventure	115-555-0170	8/1/2002 0:00
23	68	Glenna	L.	Beanston		glenna0@advent	1 (11) 500 555-(	8/1/2003 0:00
24	69	Shaun		Beasley		shaun0@advent	396-555-0187	8/1/2002 0:00
25	70	John	A.	Beaver		john8@adventur	521-555-0195	4/1/2003 0:00
26	71	Christopher	R.	Beck	Jr.	christopher1@ac	1 (11) 500 555-(	9/1/2002 0:00
27	72	Bradley		Beck		bradley0@adven	1 (11) 500 555-(	8/1/2002 0:00
28	73	Benjamin	J.	Becker		benjamin1@adv	1 (11) 500 555-(	8/1/2003 0:00
29	74	Ann		Beebe		ann0@adventure	277-555-0169	8/1/2001 0:00

15. Change the LastName filter so that it "Starts with" the characters "AD".

16. Click the "Export Using Excel PIA" button and wait for the completion notification. Notice that the Export PIA button displays the progress as the records are processed:

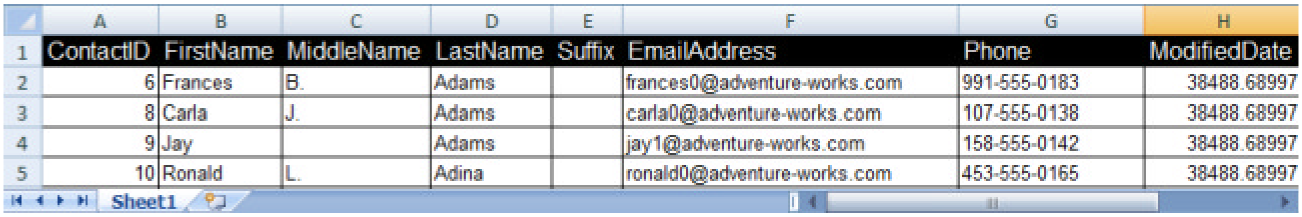


c:\ExportPIA.xls

ContactID	FirstName	MiddleNa...	LastName	Suffix	EmailAdd...	Phone	ModifiedDate
			AD				
6	Frances	B.	Adams		frances0@...	991-555-0...	5/16/2005 4:33:33 PM
8	Carla	J.	Adams		carla0@ad...	107-555-0...	5/16/2005 4:33:33 PM
9	Jay		Adams		jay1@adv...	158-555-0...	5/16/2005 4:33:33 PM
10	Ronald	L.	Adina		ronald0@a...	453-555-0...	5/16/2005 4:33:33 PM
776	Ben		Adams		ben2@adv...	182-555-0...	11/24/2001 12:00:00...
1220	Jay	G	Adams		jay0@adv...	407-555-0...	3/30/1999 12:00:00...
2931	Edward		Adams		edward8@...	947-555-0...	12/29/2003 12:00:00...
3005	Wyatt		Adams		wyatt35@...	989-555-0...	12/16/2003 12:00:00...
3089	Fernando	S	Adams		fernando3...	247-555-0...	12/2/2003 12:00:00...
3234	Jeremy	D	Adams		jeremy10...	542-555-0...	6/22/2004 12:00:00...

26 of 87

17. Open the newly exported file in Excel to view the data:



	A	B	C	D	E	F	G	H
1	ContactID	FirstName	MiddleName	LastName	Suffix	EmailAddress	Phone	ModifiedDate
2	6	Frances	B.	Adams		frances0@adventure-works.com	991-555-0183	38488.68997
3	8	Carla	J.	Adams		carla0@adventure-works.com	107-555-0138	38488.68997
4	9	Jay		Adams		jay1@adventure-works.com	158-555-0142	38488.68997
5	10	Ronald	L.	Adina		ronald0@adventure-works.com	453-555-0165	38488.68997

1.9.2 Displaying Grid Data in Telerik Reports

Telerik Reporting is a powerful report generation tool with a myriad of applications for producing custom professional reports. In addition to its Excel exporting capabilities, RadGridView data can also be exported using Telerik Reporting tools.

For this lab, you will need the Reporting product from Telerik in either purchased or demo version, as well as the Telerik RadGridReportingLite dll, downloadable from the following location:

http://www.telerik.com/community/code-library/submission/b311D-bedcch.aspx

You can find the complete source for this project at:

\VS Projects\Grid\<VB|CS>\RadGridView\12_ExportReporting

1. Create a new Windows Forms Project.
2. Add the RadGridReportingLite.dll file as a project reference, and to the Form's code-behind class:

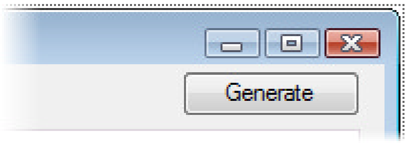
   [VB] Adding the RadGridReportingLite reference

   ```
   Imports RadGridReportingLite
   ```

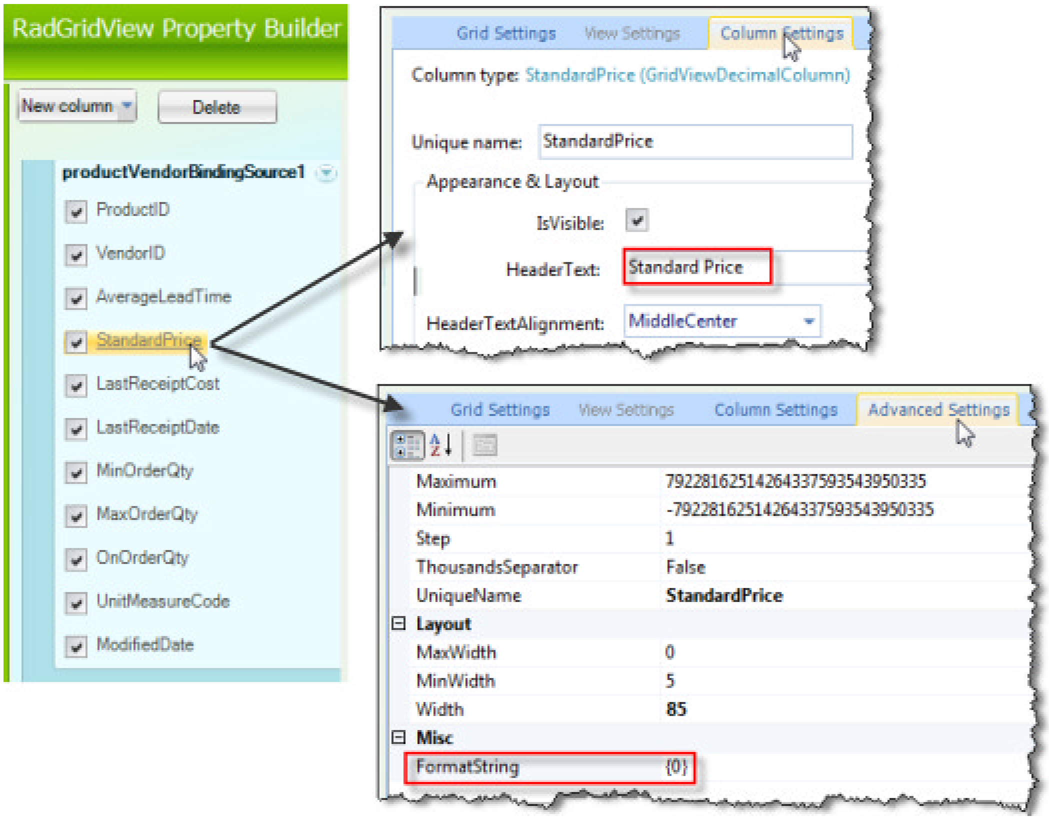
   [C#] Adding the RadGridReportingLite reference

   ```
   using RadGridReportingLite;
   ```

3. Now in design view, drag a RadGridView control onto the Form.
4. Add a button to the top of the form and name it "btnGenReport".



5. Using the RadGridView control's Smart Tag, add a project datasource connecting the RadGridView to the Purchasing.ProductVendor table of the AdventureWorks sample database.
6. Again using the Smart Tag, open the Property Builder and set the following properties of the RadGridView control's MasterGridView Template:
   - AllowAddNewRow = False
   - EnableFiltering = True
   - AutoSizeColumnsMode = Fill
7. To get a cleaner-looking report, we will add some formatting to a few of the table fields and modify the header text for several of the columns. Still in the Property Builder, select the StandardPrice column, and set the Header Text to "Standard Price". Also change the FormatString to "{0:c}" in the property pane.



productVendorBindingSource1

- ☑ ProductID
- ☑ VendorID
- ☑ AverageLeadTime
- ☑ StandardPrice
- ☑ LastReceiptCost
- ☑ LastReceiptDate
- ☑ MinOrderQty
- ☑ MaxOrderQty
- ☑ OnOrderQty
- ☑ UnitMeasureCode
- ☑ ModifiedDate

Column type: StandardPrice (GridViewDecimalColumn)

Unique name: StandardPrice

Appearance & Layout

IsVisible: ☑

HeaderText: Standard Price

HeaderTextAlignment: MiddleCenter

Maximum	79228162514264337593543950335
Minimum	-79228162514264337593543950335
Step	1
ThousandsSeparator	False
UniqueName	**StandardPrice**
⊟ **Layout**	
MaxWidth	0
MinWidth	5
Width	**85**
⊟ **Misc**	
FormatString	{0}

8. In a similar fashion, set the following columns’ HeaderText and FormatString values.
   - AverageLeadTime: Header Text = Average Lead Time
   - LastReceiptCost: Header Text = Last Receipt Cost, Format String = {0:c}
   - LastReceiptDate: Header Text=Last Receipt Date, Format String = {0:d}
   - MinOrderQty: Header Text = Min Order Quantity
   - MaxOrderQty: Header Text = Max Order Quantity
   - OnOrderQty: Header Text = On Order Quantity
   - UnitMeasureCode: Header Text = Unit Measure Code
   - ModifiedDate: Header Text = Modified Date, Format String = {0:d}

9. Add a Click event handler to the Generate Report button by double-clicking the button in design view, and add the following code to the event handler. This code does the actual report generation using the RadGridReportingLite functions.

[VB] Generating the Report

```
Private Sub btnGenReport_Click(ByVal sender As Object, ByVal e As EventArgs)
 ' create the report exporting object and pass it the report name
 Dim report As New RadGridReport("Sample Report")
 ' configure the report
 report.FitToPageSize = True
 report.AllMargins = 1
 report.PageLandScape = True
 report.RepeatTableHeader = True
 report.UseGridColors = True
 ' display the report preview
 report.ReportFormShow(Me, Me.radGridView1)
```

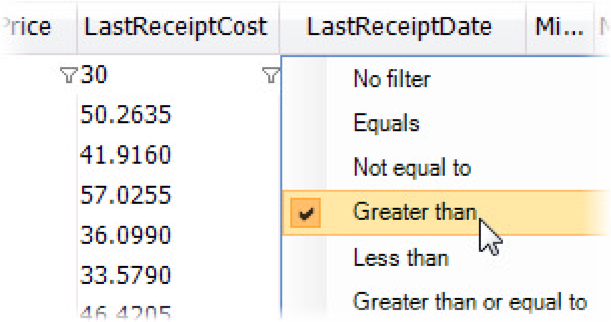
```
End Sub
```

[C#] Generating the Report

```
private void btnGenReport_Click(object sender, EventArgs e)
{
 // create the report exporting object and pass it the report name
 RadGridReport report = new RadGridReport("Sample Report");
 // configure the report
 report.FitToPageSize = true;
 report.AllMargins = 1;
 report.PageLandScape = true;
 report.RepeatTableHeader = true;
 report.UseGridColors = true;
 // display the report preview
 report.ReportFormShow(this, this.radGridView1);
}
```

Here we are hard-coding several of the report generation options for simplicity's sake, but these options could also be provided to the user in a dialog window, for example.

10. Set the project as the Startup Project, then save and run to view the Form. Add a "Greater than" filter to the Last Receipt Cost column with a value of 30 dollars.



This step is not necessary for functionality and is only used here to show how the Report displays only the data currently contained in the RadGridView control at runtime.

11. Click the Generate Report button. You will see the Generating report... message. Then the sample report will display in a new window, neatly formatted and suitable for printing or other presentation.

This sample does not even scratch the surface of the functionality of Telerik Reporting, but instead demonstrates a way to quickly produce neatly formatted printable copies of data shown in the RadGridView control using the RadGridViewReportingLite library.

1.10 Summary

In this chapter you became familiar with the RadGridView control, using the Smart Tag, Property Builder and Properties window to configure the grid and to bind the grid to data. You learned how to add special purpose column types at design-time and in code, how to group/filter/sort data based on user input at runtime, by hand at design-time and programmatically in code. You displayed hierarchical data from multiple related tables in the grid. You used the RadGridView Virtual Mode feature to take low-level control over the grid's refresh process. You learned how to bind RadGridView to LINQ data sources. Finally you learned multiple methods of exporting grid data to Excel and out through the Telerik Reporting engine.

Index