Wrox Programmer to Programmer™

Professional

Microsoft® SQL Server® 2008 Reporting Services

Paul Turley, Thiago Silva, Bryan C. Smith, Ken Withee
Hitachi Consulting
Forewords by:
Jason Carlson, Product Unit Manager, SQL Server Reporting Services, Microsoft Corporation
Thierry D'hers, Group Program Manager, SQL Server Reporting Services, Microsoft Corporation

wrox™

Updates, source code, and Wrox technical support at **www.wrox.com**

5

Basic Report Design

If you are new to Reporting Services, you'll get started in this chapter with some basic report design concepts. If you have had prior experience with earlier versions of Reporting Services, you may be able to skip ahead after we introduce a few things that have recently changed. In order to meet the needs of those readers who are new to the product and also those who have done report design with SQL Server 2000 or 2005, we have organized this section on report design to make it easy for you to learn what you need without having to read each of these chapters from start to finish. Using this approach, you also shouldn't have to learn about all the low-level details if you simply want to know the basic steps. But you also shouldn't have to read through an elementary introduction of the topic if you are ready to take on advanced report design.

Part II, "Report Design," consists of the following four chapters:

- Chapter 5 introduces the report design environment and then teaches you the essentials of the report design elements. The theme of this chapter is *what* you can do, rather than *how* to do it. You'll learn the fundamental components and building blocks: data sources, datasets, report body, report items, data ranges, and page layout properties. You'll see how to use the Report Wizard to get started with common report design features. In this chapter, you will learn to use both the new Report Builder 2.0 designer and the integrated development report designer in Business Intelligence Development Studio (BIDS).
- In Chapter 6, you will see *how* different types of reports are created using these report items and design elements. Once you understand the core components and how to put all the pieces together, you will learn design patterns for common report styles used in creating real-world report solutions.
- Chapter 7 is about data access — designing data sources and datasets. You will learn techniques for writing efficient queries to filter data and apply business logic. You will use parameters to prompt users for input and filter and change the report output based on user selections and values passed to the report. You also learn to use expressions to modify report properties and to dynamically change the way data is displayed based on parameterized input or field values.

Excerpted with permission from Professional Microsoft SQL Server 2008 Reporting Services
ISBN: 978-0-470-24201-8, Wrox Press

- In Chapter 8, you will learn to take reporting to the next level by combining the skills learned in these previous chapters with techniques learned from many report projects. You will learn to apply design patterns to solve business problems in creative ways. You'll see how to combine different report items and data ranges with data groups, parameters, variables, and expressions to create "super reports" that have advanced and dynamic functionality. In this chapter, we will demonstrate several actual reports based on real business problems encountered over the years in report solution consulting.

The four chapters in this section will progressively explore more functionality. It is the nature of this technology that different report design elements will be introduced and then covered at more advanced levels. To help you better understand our approach, refer to the following table of report design elements and the degree to which each will be covered.

Design Element	Chapter 5 Basic Report Design	Chapter 6 Report Layout and Formatting	Chapter 7 Designing Data Access	Chapter 8 Advanced Report Design	Chapter 10 Report Solution Patterns and Recipes
Textbox	Introduction	Example		Detail	Detail
Table	Example	Exercise		Detail	Detail
Matrix	Example	Exercise	Example	Detail	Detail
List				Introduction	Detail
Chart	Example	Exercise		Detail	Detail
Gauge	Introduction	Example		Detail	Detail
Composite reports		Introduction		Exercise	
Row and column groups	Introduction			Detail	Detail
Parameters and filtering			Introduction and exercise	Detail	Detail

Report Design 101

You're going to learn how to design reports using the simplest of all methods. If you are new to report design and Visual Studio, you will find the Report Wizard to be a convenient way to design simple reports. If you are an experienced report designer or application developer, or if you need to learn to design complex, custom reports, you're likely to use the Report Wizard a few times and then leave it behind.

Let's take a look at the big picture of designing reports in SQL Server Reporting Services. We will examine most of the important features of Reporting Services just to get an idea of what you can do with the product. We'll also point you to later chapters to get more information and to learn about the details. We will be using Visual Studio to design and create reports. You may use any edition of Visual Studio 2003 or later.

Before you read on, you need to get your bearings and get a sense of this chapter's direction. In any technical book, it's necessary to get every reader to a basic level of understanding before moving on to advanced material. Different readers may have varying levels of expertise or experience with Visual Studio, so let's start with the basics. Don't worry — whether you've never seen Visual Studio before or you are a tenured Visual Studio developer, we're going to cover the right material at the right depth at the right time. If you have used Visual Studio for application development, please be patient as you read through the next section. If you have never written a line of code in your life or if you are new to Visual Studio, you're in luck.

This chapter covers the following topics:

- Using the Report Wizard
- Importing reports
- Planning for extensibility
- Report items and data regions
- Formatting considerations
- Pagination and printing considerations

Report Designers

One of the most significant changes in the SQL Server 2008 reporting arsenal is the introduction of a new report design tool — and a major overhaul of the old one. That's right; there are now two different report design tools that you can use to create the same type of reports. Before we get into the longer version of the story about these two different design tools, I'd like to introduce them at a very high level. Chances are, as a report designer, your needs fall into one of two general categories:

If you work in a small business unit and you just need to design a simple report, you aren't concerned about managing the reports that have been created by other report designers. You will likely find the Report Builder 2.0 tool the easiest and best choice. This is for new report designers who don't necessarily want to contend with a lot of technical sophistication and features they won't need to use.

If you work in the IT group for your company and need to coordinate your report design efforts with others, you may need a more capable tool. If you are familiar with project-based development tools, work within a formal IT project methodology, and require version control, development, testing, and deployment management, you will likely benefit from the Report Designer integrated with BIDS or Visual Studio 2008.

To understand how this came to be, here's a quick Reporting Services history lesson. In 2003, I taught the first beta delivery of the Reporting Services course we had written for Microsoft at a training center near

Boston. The product was a few months from official release, but many people in the industry were anxiously awaiting Microsoft's first enterprise-ready reporting tool and wanted to learn to design reports. I had been teaching Microsoft development technologies for years and was accustomed to teaching classes for programmers and database professionals. On Monday morning, I had a room full of report designers who were versed in using a competing report product. After I introduced the Reporting Services architecture, I asked everyone to open Visual Studio and to create a new project. They just looked at me, not understanding what I had asked them to do. I learned a valuable lesson that day. Like the SQL Server product development teams at Microsoft, I was reminded that not everyone on the planet who may need to design reports was a seasoned IT professional. That morning in Boston, it occurred to me that these weren't programmers, and most of them had never used Visual Studio before.

As the release of SQL Server 2005 neared, which was to include the second edition of Reporting Services, the product team added a tool designed specifically to enable nontechnical users to design their own reports. This tool was called *Report Builder*, based on a newly acquired, third-party reporting technology. Although the report design experience was somewhat similar to the programmers' tool, the resulting report didn't have the same capabilities as the standard reports designed using Visual Studio. When the product shipped with SQL Server 2005, an edition of Visual Studio, called *Business Intelligence Development Studio (BIDS)*, installed with the SQL Server client tools. Users could design their own reports using the Report Builder technology, which was accessible from the central Report Manager web site. Once users and IT managers learned about the limitations of Report Builder, when compared to the more powerful features in standard Reporting Services, many chose to put the majority of their effort into developing standard Reporting Services reports rather than using the Report Builder technology.

The Report Builder experience taught us that users wanted a simple report design tool, but they also wanted the option to leverage the more capable Reporting Services features, prompting an overhaul of the Report Designer for creating standard reports. The result is a sleeker, easier-to-use report designer that now comes in two flavors: a stand-alone application for dedicated report designer/power users and an integrated version that runs in the Microsoft Visual Studio environment.

Report Builder 2.0

Let's clear up any confusion about the name of this tool. If you are a SQL Server 2005 Reporting Services user and are familiar with the old Report Builder, this isn't the same thing. If you haven't used the 2005 Report Builder, well, don't worry about it.

> **Report Builder 2.0 A vailability**
>
> As of this printing, the Report Builder 2.0 report design tool is just wrapping up development and is available for download from `www.codeplex.com`. This tool will also be included with later service releases of SQL Server 2008.

Report Builder 2.0 is a fully capable, stand-alone report designer used to create standard Reporting Services reports. It's simpler, though, and designed with the information worker in mind. Unlike Visual Studio, it isn't used to manage projects or IT solutions. It doesn't have integrated version control or

multiple deployment configurations. It's simply a report designer with all the features necessary to design simple and advanced reports.

After installing Report Builder 2.0, you can start it by choosing Start ⇨ All Programs ⇨ SQL Server 2008 Report Builder> ⇨ Report Builder 2.0.

Rather than managing reports in a project, as you would with Visual Studio or BIDS, a single report is opened from any location, just like a Word or Excel document (see Figure 5-1.).

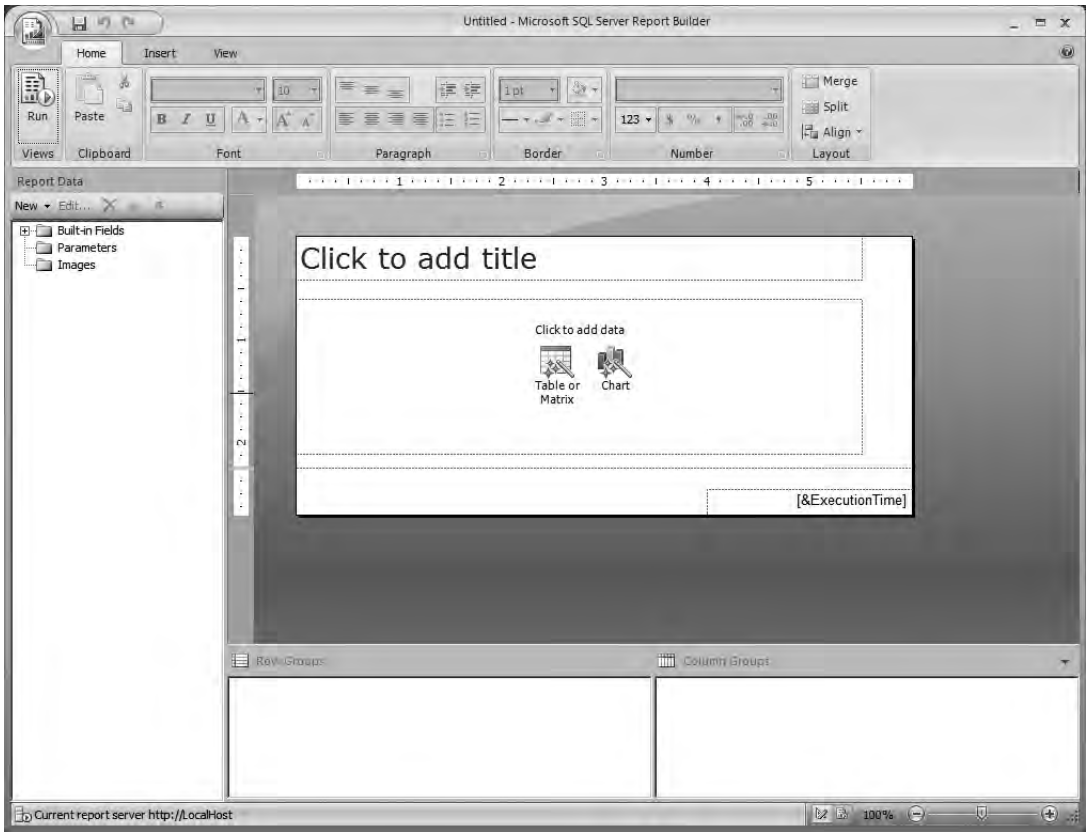

Figure 5-1

Office Tabs and Ribbons

The Report Builder 2.0 Designer sports a look much like the Office 2007 applications. Large icon buttons are arranged on ribbons that may be accessed using tabs that are arranged along the top of the interface. Tabs and ribbons are part of the Microsoft Office Fluent user interface and replace the dropdown menus found in earlier Microsoft applications. When a tab is selected, the corresponding ribbon displays icon buttons and commands related to different activities. Commands are organized into logical groups that are enabled only when the appropriate object is selected or when you are in the appropriate designer mode.

Home Tab

The Home tab, as shown in Figure 5-2, is the starting point in the Report Designer and contains common layout and formatting features for designing reports.

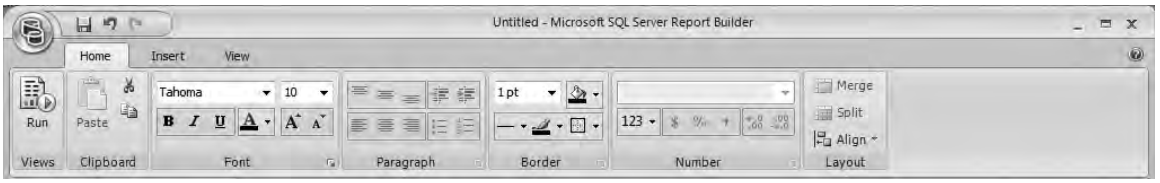

Figure 5-2

The formatting commands on the Home tab ribbon will work with multiple selected items. To select multiple items, draw a marquee box entirely around a group of items while holding the left mouse button, or click individual items while holding the Shift or Ctrl key. Note that most report item properties may also be set using a custom Properties dialog that appears after right-clicking on an item, but this option doesn't work with multiple selected items. Using the Properties window or Ribbon commands does work with multiple selected items.

On the Home tab, you will find the following groups and commands:

- **Clipboard** — This group contains common Windows Clipboard features including Copy, Cut, and Paste. The Clipboard is an essential tool in report design and is used often for duplicating captions and expressions. You can use the Clipboard with objects such as report items and images, in addition to text values that you might want to cut, copy, or paste using the Windows Clipboard.
- **Font** — The items in this group correspond to the font-related properties for textboxes. Using the command buttons on the ribbon is typically much faster than setting all the individual properties for each object in the Properties window.
- **Alignment** — The Alignment commands allow you to easily format multiple report items by aligning the edges or centers of all items selected. When this option is used with a group of items, all items are aligned with the first selected item.
- **Border** — The Border commands allow you to quickly set the border style, weight, and color for the top, left, right, and bottom borders of any object. Using the ribbon commands to set an outside border for an item actually sets all these individual properties. Keep in mind that nearly all items have border properties and may contain child items that also have borders. For example, setting the borders for a table does not set the borders for all the cells within the table. If you need to fine-tune borders, you may find it useful to set the borders for the entire table, rows, columns, or a range of cells and then adjust the borders for individual cells afterward.
- **Arrange** — Commands in this group are used to change the layered order of report items and are used for placing items in front of or behind other items. This feature is useful for managing items in the design environment. Except in rare cases, it is not advisable to stack report items unless an item is contained within another report item or data range. For example, two

textboxes or images shouldn't share the same screen space, but a textbox or an image may be placed into the cell of a table or into a rectangle item. Some report renderers will move stacked items to make reports compatible with different viewers and browsers.

- **Preview** — Toggles between the Report preview and Design view of the report.

Insert Tab

The Insert tab and corresponding ribbon contain report design components that you can place on the report body to visualize data or format the report layout (see Figure 5-3). Items in the Data Regions, Report Items, and Subreports groups are added to the top-left corner of the report body when the ribbon button is clicked. Note that this behavior differs from the Visual Studio report design Toolbox items. In the other report designer, items are dragged or drawn onto the report body.

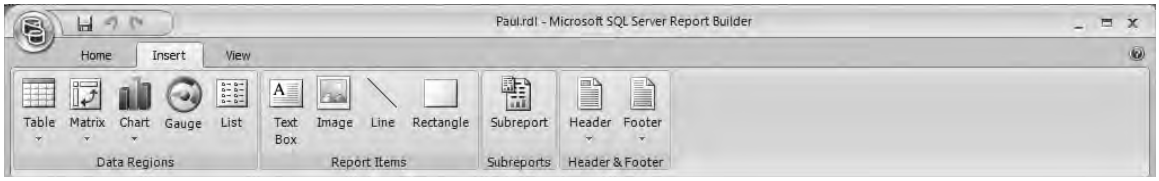

Figure 5-3

These items are organized into the following groups:

- **Data Regions** — The data regions consume data from a dataset and visualize it into rows, columns, chart items, and gauges. Data regions are actually report items, but they are different from standard report items because they consume whole sets of data rather than just individual values.

  To add a data region to the report in Design view, click the ribbon button for the appropriate data region. The data region is added to the top-left corner of the report body. Click once on the item to show the repositioning handle, and then use the mouse to drag and drop the data region to the right place on the report.

- **Report Items** — Report items may be visual elements used to enhance the report layout, such as lines and rectangles. Some report items are designed to display or visualize a specific value or aggregated value from a dataset, such as a textbox or image.

  Report items are added to the report body in the same manner as data regions. Often, you will want to place a report item into a data region or other report item so that it becomes a container.

- **Subreports** — This group contains only one item — a subreport. This is a special report item that allows you to place an existing report within this report body. If a subreport is placed directly on the report body, it will be rendered once, at that location, within the report. If it is placed within a data region, a separate instance of the subreport will be rendered for each row or column of the data region. Subreports can be filtered based on a field value to define master/detail relationships between datasets based on separate data sources.

- **Header & Footer** — This group is used to enable or disable page headers and page footers for the report. Unlike the other items in the Insert table, headers and footers are not "added" to the report body but are designated sections of the report.

View Tab

The View tab and ribbon, as shown in Figure 5-4, includes the Report Views group. You can view the report design in one of two different modes. This group on the View tab enables you to view the report in Design mode or to preview the report as it would appear if it were deployed to the report server. You can toggle the view mode by using the Preview group/button on the Home tab and ribbon.

The View tab also contains a group to show or hide utility panes and rulers. The Data pane, displayed on the left side of the Report Designer, is used to manage built-in fields, parameters, images, data sources, datasets, and data-set fields. This pane replaces the Data tab in earlier versions of the Report Designer that contained some of these options.

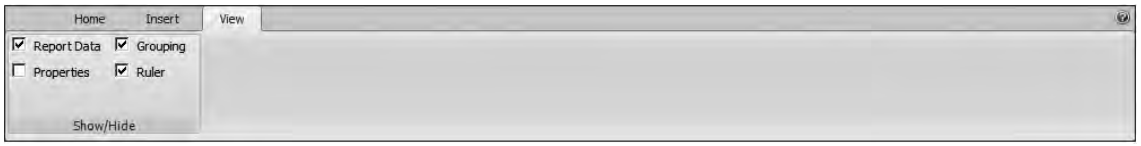

Figure 5-4

The grouping pane is displayed at the bottom of the Report Designer and shows the row and column groups associated with the currently selected data range object. You can drag fields into the row and column list boxes to create data groups. Each group is displayed with a set of tools to enable adding, editing, and deleting a group. The dropdown button displayed on each group item enables a dropdown menu with additional group management options.

The Properties pane is used to browse and change nearly all the properties for the currently selected design object. This is the traditional method used to manage property values. Although it duplicates some property settings that may be set using more convenient methods, it provides a consistent interface for managing all properties without the need to use different methods to access them.

Run Tab

When a report is previewed, the Run tab and ribbon are displayed with several feature groups and corresponding icons.

- The Views group includes the Design button, which is used to return to the report designer.
- The Zoom group and button are used to change the magnification of the report preview.
- The Navigation group includes several buttons. The First, Previous, Next, and Last buttons provide a convenient way to navigate through the report pages, and the page textbox allows you to navigate to a specific page. The Refresh button will force a requery of all data sources and rerender the report. The Stop button suspends report execution and stops the queries and the report from running. The Back button navigates to a previous report when using a drill-through report action.
- The Print group contains client-side printer controls.
- The Export button allows a report to be rendered into selected formats and to be exported to a separate file.

- The Options button allows you to hide and show optional toolbars in the report preview interface, including a document map (if the previewed report contains document map settings) and the parameters selection pane.
- The Find group includes a search textbox and corresponding button used to find a string of text within the body of the rendered report.

Figure 5-5 shows a report in preview mode. While the report is previewed and the Run tab options are visible, a smaller pane is displayed between the ribbon and the report with a link to change user credentials and the View Report button. Click this button after making parameter selections to execute the report with these selections.

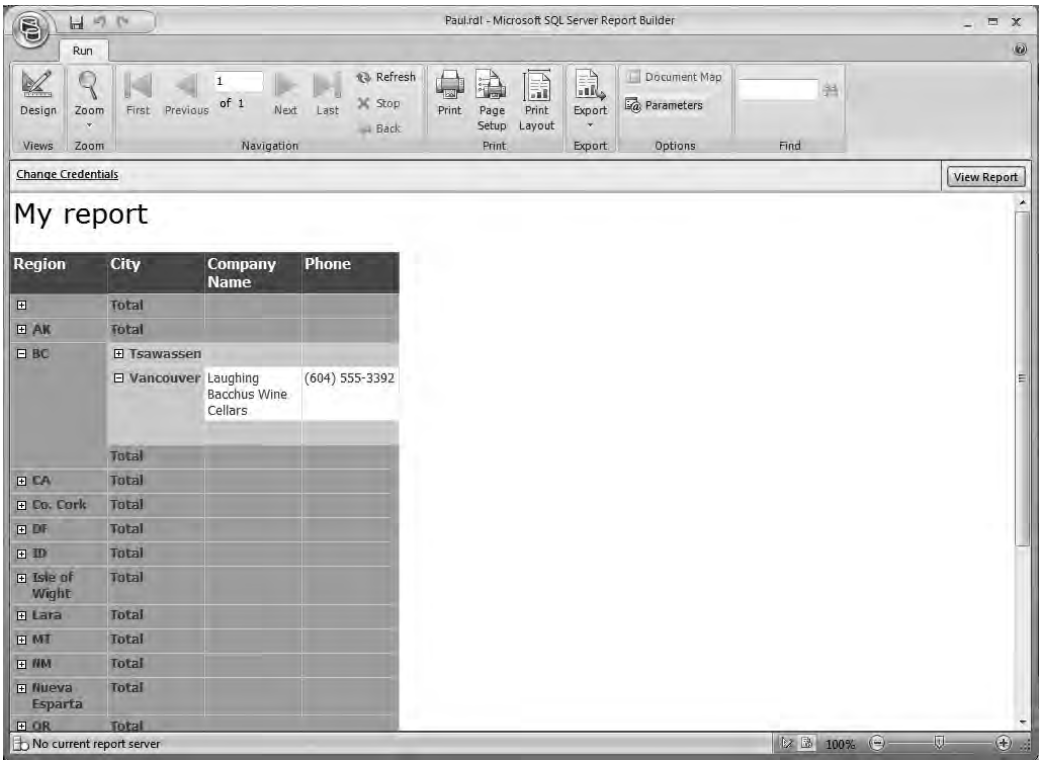

Figure 5-5

When the Run tab is displayed, you can switch back to design view using the Design button in the Views group. This will replace the Run tab and ribbons with the Home, Insert, and View tabs that are visible while in design mode. You can also always toggle between report design and preview using the smaller icons in the status bar, to the left of the zoom slider control.

Viewing and Setting Properties

You can set properties for different objects in the Designer using three different methods. The first and most convenient method is to use the Home tab on the Ribbon. For simple report items, like a textbox, just click the report item, and then use the Ribbon commands to make changes. You can also choose multiple objects of the same type. This is easy to do with the properties font size, color, background color, and so on.

Another method is to use the Properties pane to display all the properties for a selected object or group of objects. With more than one object selected, those properties with common values are displayed and can be set as a group.

The final method is to right-click on the object and then use the object-specific Properties dialog. This window typically displays specific properties and has an interface designed specifically to manage the selected object. This technique cannot, however, be used to change properties for multiple objects at one time.

Data Sources

A new addition to the Report Builder 2.0 interface is the ability to use shared data sources that have been deployed to the report server. Figure 5-6 shows the data source selection dialog in the New Table or Matrix wizard. When using the report or report item wizard or when creating a dataset, you can select a deployed data source, browse to a data source file in the local file system, or create a new data source, using corresponding buttons in this interface.

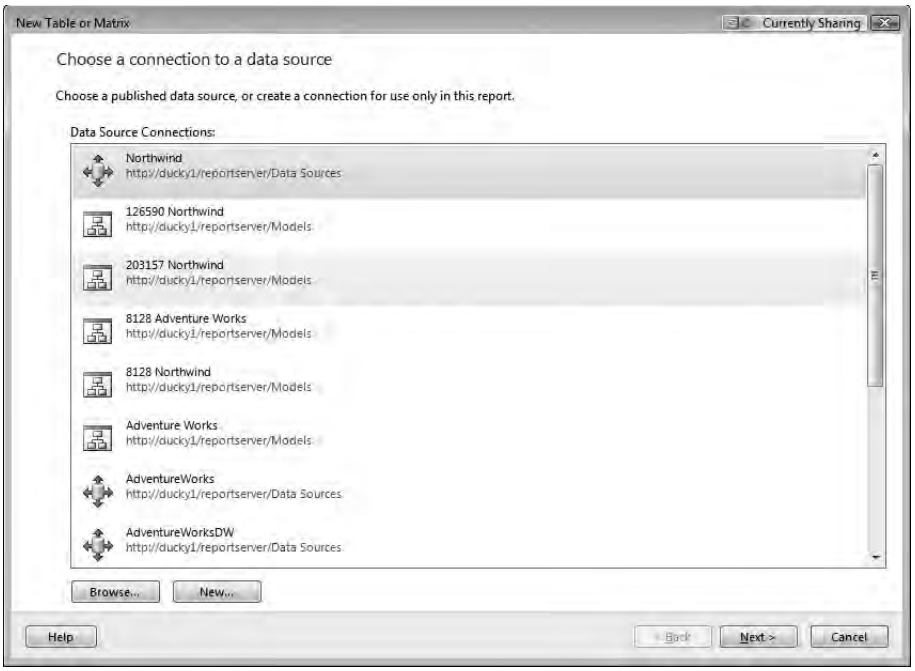

Figure 5-6

Keep in mind that when prompted for a data source file location, you may use a traditional file path or report server URL. If you are using SharePoint integrated mode, you may also have the option to provide the URL for data sources located in a SharePoint data connection library.

Server Reports

The new Report Builder 2.0 is a true ad-hoc reporting tool that allows users to create reports directly on the server without using the local file system. This capability is not enabled in a new Reporting Services installation by default. To enable server reports, a script file is provided that will enable this capability and set up folders on the report server for server-based report design for users running Report Builder 2.0. The following command should be executed on the command line, substituting your report server name for the hostname in this example:

```
rs.exe -i SetDefaultFolders.rs -s http://hostname/reportserver
```

With server reports enabled on the report server, any standard reports to which the user has access can be opened directly in Report Builder 2.0 using the standard Open dialog. To open a report on the server, local, or network file system, click the "orb" start button in the top-left corner and choose Open from the menu. You can navigate to a folder using the most recent items or other folder shortcuts on the Open Report dialog window.

Report Design with Report Builder 2.0

Many of the report design features in Report Builder 2.0 are the same or very similar to the BIDS report designer, but it's a more sleek and uncomplicated interface. To create a new report, you can either use the Report Wizard or manually build a data region and a dataset to supply its data. The basic theme in Report Builder 2.0 is simplicity. To this end, designer and wizard dialogs will automatically open when a required object (such as a data source, dataset, or report data region) must be defined. Although I do a lot of report design in Visual Studio projects, I often prefer to start designing new reports with Report Builder 2.0 because it's so much more convenient.

Figure 5-7 shows a basic report in the designer. In this example, the matrix data region was added to the report body from the ribbon on the Insert tab. After defining a dataset, fields are dragged from the Report Data pane on the left side of the designer directly into cells of the data region or to the Row Groups or Column Groups lists below the report design surface.

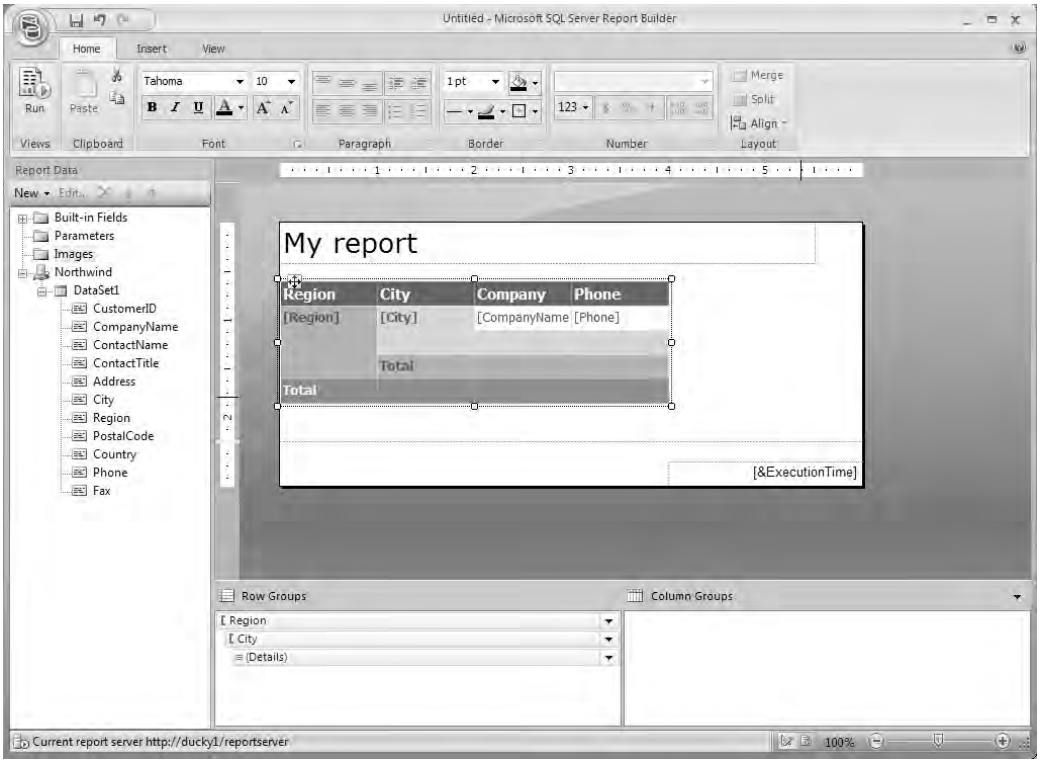

Figure 5-7

You'll learn more about specific report design techniques in the chapters that follow. Whether using the BIDS report designer or Report Builder 2.0, the process is much the same.

Data Region Wizards

A new addition to Report Builder 2.0 is a set of data region-specific design wizards. Back in Figure 5-2, you can see that when a new report is opened (or you simply open Report Builder 2.0), two icons are displayed in the center of the new report design surface. Click either of these to launch the appropriate design wizard that will lead you through the process of adding a data source and dataset query and organizing data fields and groups in a table, matrix, or chart. Figure 5-8 shows the New Table or Matrix dialog. After defining a query, fields are simply dragged into row or column groups, or into the data area of a table or matrix report. Again, you will learn to design all these report types in the chapters that follow.

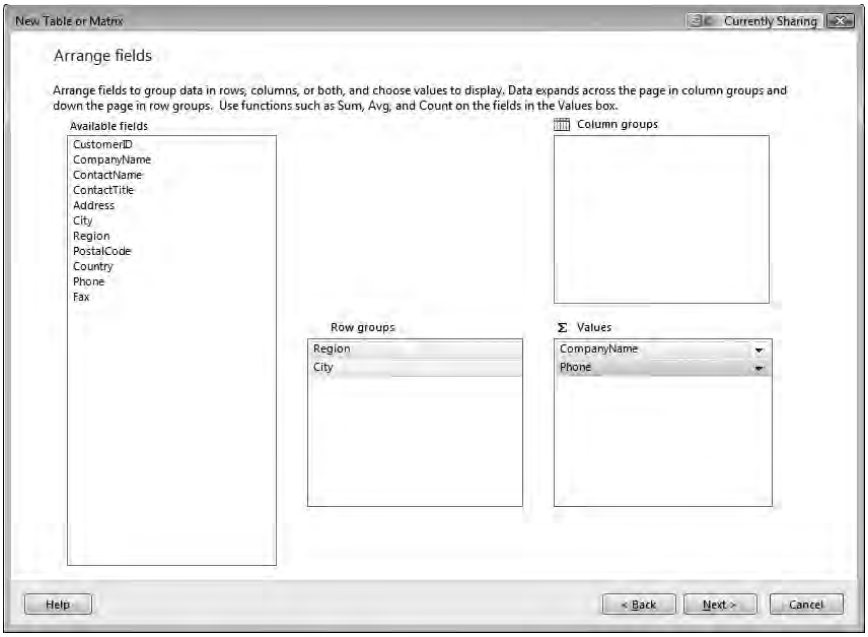

Figure 5-8

Another convenient, new feature is shown in Figure 5-9 on the report wizard page titled Choose the Layout. After you add field groups, a preview of the report using actual data is displayed in the following wizard page. I'm impressed with this behavior because this is an actual preview of the report, with all the features chosen in the report wizard up to this point in the design process.

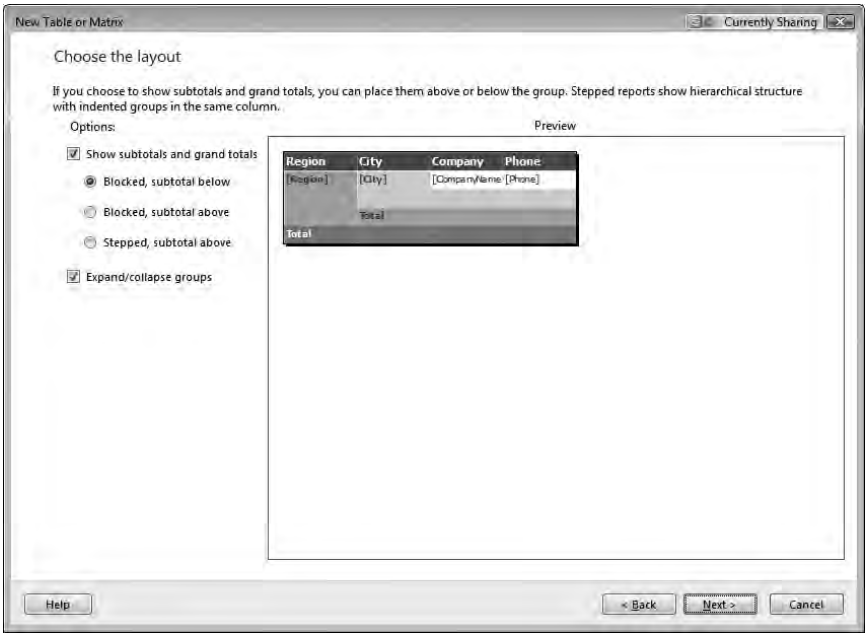

Figure 5-9

Excerpted with permission from Professional Microsoft SQL Server 2008 Reporting Services
ISBN: 978-0-470-24201-8, Wrox Press

Note the selections on this page, which include the option to include subtotals and totals. You can alter the way totals are arranged and presented — in blocked headers rows (with totals above or below) or in-line with group summaries. You can also include drill-down report functionality, in which group details can be dynamically expanded or collapsed by the user.

Figure 5-10 shows the report wizard page titled Choose a Style. These styles are based on templates that affect default fonts, borders, and colors on the report.

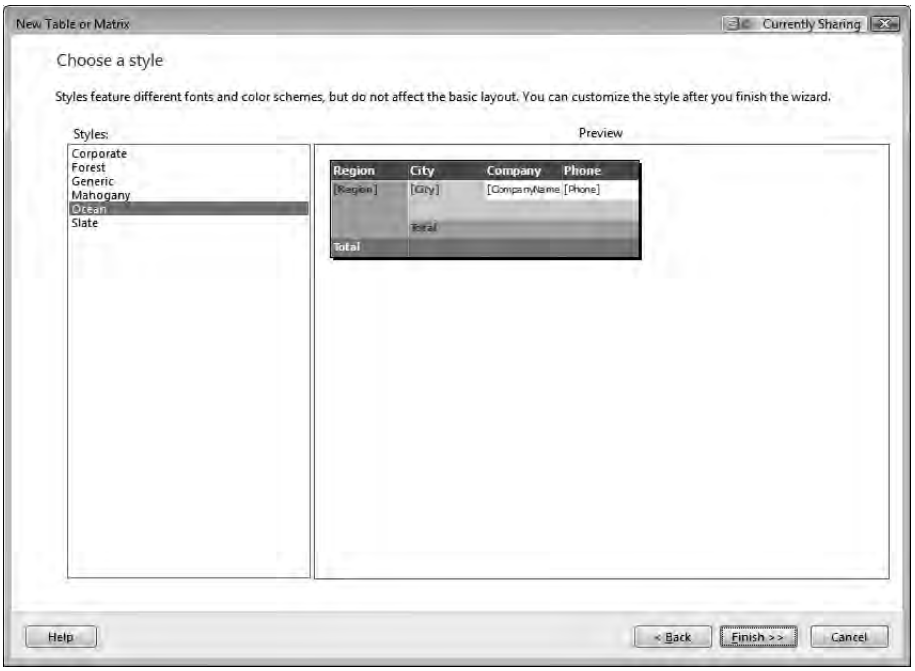

Figure 5-10

Formatting and Sample Values

Numeric formatting can be applied very easily using the ribbon. This is another convenient design feature and a significant improvement over previous versions of the report designer. Not only can you apply a variety of predefined format options to a field on the report, but you can also preview the format at design time. Figure 5-11 shows the Sample Values preview feature. Select the field value textbox in the designer. A format can be applied using the Number group on the Home tab ribbon. Choose Sample Values from the numbers drop-down list to view a representative formatted numerical value in the textbox.

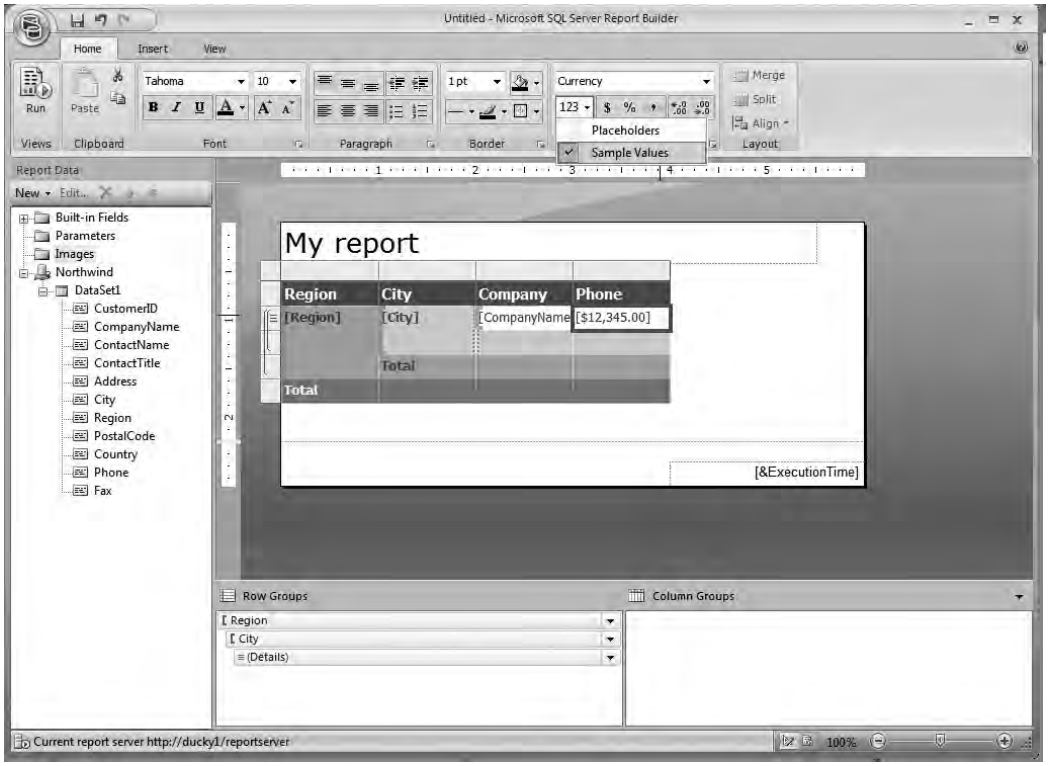

Figure 5-11

Report Builder 2.0 brings reporting to the business user. It enables information workers to design reports without using tools that were created for application developers. The Report Builder tool introduced in Reporting Services for SQL Server 2005 (now called Report Builder 1.0) made simple report design possible for less-sophisticated users, but it doesn't expose the powerful capabilities of the Reporting Services architecture. The older tool, Report Builder 1.0, is still fully integrated with SQL Server 2008 Reporting Services and is available to use if you need to support the older Report Builder-style reports. You can learn more about this tool in Chapter 12.

Report Builder 2.0 is a simple, easy-to-use report design tool that creates standard Reporting Services reports with all the features and capabilities of those created with more complex design tools. This tool effectively bridges the gap between ad-hoc, self-service reporting and enterprise reports that are typically designed and supported by information technology professionals.

Integrated Development Environment

Report design for Reporting Services has been performed using the Microsoft Visual Studio integrated development environment (IDE). This wouldn't be a true Microsoft product if they didn't change the name every few years. This tradition continues, and a special, license-free version of the Visual Studio .NET IDE was first incorporated into the SQL Server 2005 client tools called Business Intelligence Development Studio, commonly referred to as *BIDS*.

When Reporting Services was introduced in 2004 for SQL Server 2000, the report designer was integrated into Microsoft Visual Studio 2003. Anyone needing to design reports had to acquire a separate copy and license for Visual Studio. In the SQL Server 2005 version, the relatively unchanged Report Designer could be used in Visual Studio 2005 or in BIDS. Whether you had a retail version of Visual Studio or you used BIDS, the report design experience was pretty much the same. The Report Designer has once again been updated to work with the current version of Visual Studio or BIDS. However, the recent overhaul of the Report Designer components has changed a few things in this familiar environment, with several notable improvements.

You should find it relatively easy to switch between the stand-alone Report Builder 2.0 designer and the BIDS designer, as many of the components are the same. Some of the utility windows and menus are unique to BIDS and work differently from the ribbon commands in the stand-alone designer.

Using Business Intelligence Development Studio

Business Intelligence Development Studio, or BIDS, is an edition of Microsoft Visual Studio 2008 with project templates to create solutions that include Integration Services packages, Analysis Services cubes and databases, and, of course, Reporting Services reports. With the SQL Server 2008 client tools installed, you can open BIDS by choosing Start ⇨ All Programs ⇨ Microsoft SQL Server 2008 ⇨ SQL Server Business Intelligence Development Studio.

If you have installed any edition of Microsoft Visual Studio 2008, that application will open when you launch BIDS. You will note in Figure 5-12 that I have Visual Studio Team System 2008 installed, which shows up in the Start Page header. If you are not using Visual Studio, you will see the Business Intelligence Development Studio edition. The only significant different between these editions will be the list of available project types and languages.

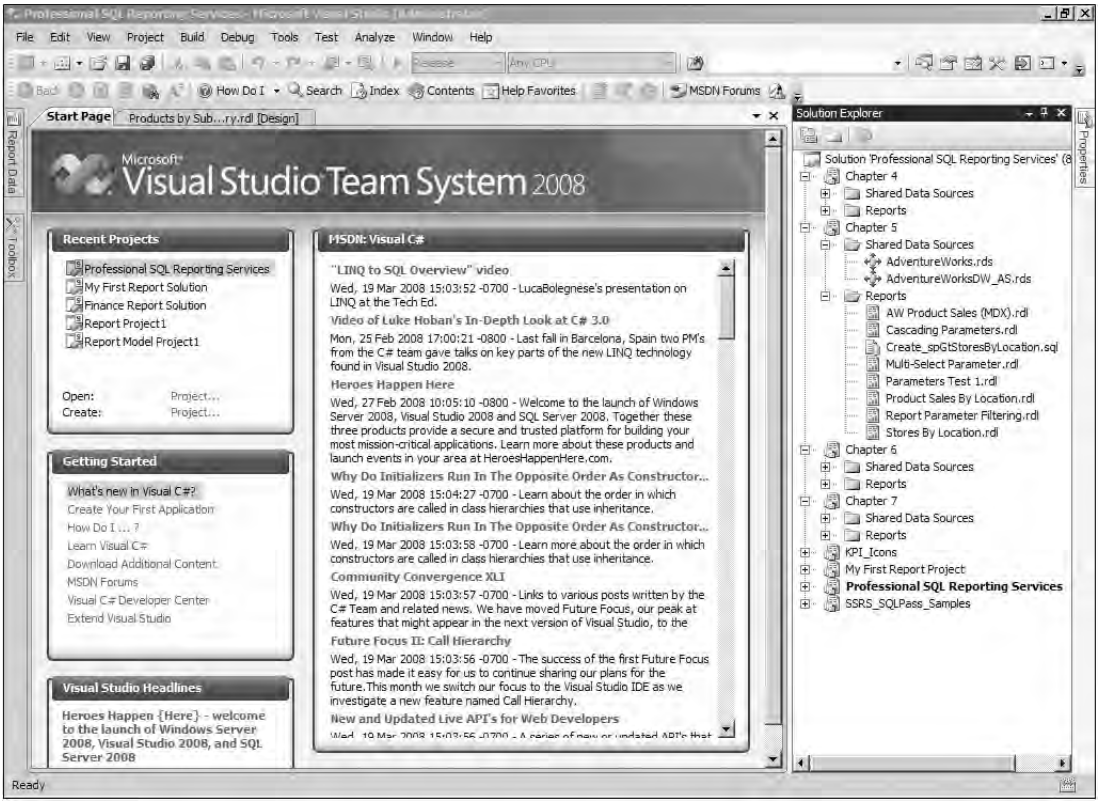

Figure 5-12

When the development environment is opened, a start page is displayed with the Visual Studio 2008 logo. The first order of business is to open an existing project or create a new one. To create a project, you can either use the File menu or the left-most button on the toolbar. When you use either of these options to create a new project, the New Project dialog is displayed (see Figure 5-13).

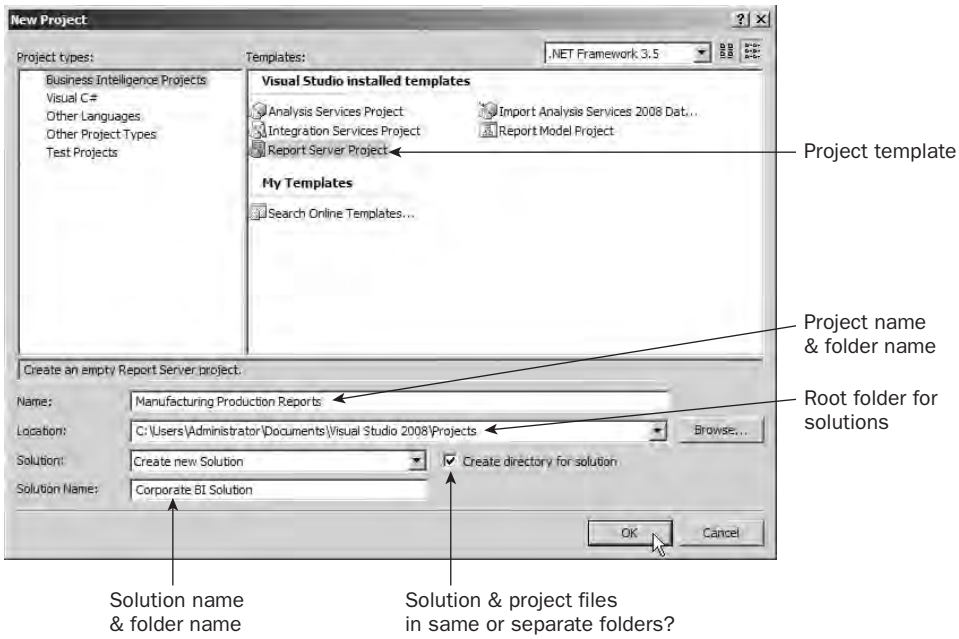

Figure 5-13

The Reporting Services installation adds a project type category to the Microsoft Integrated Development Environment called *Business Intelligence Projects*, which you see in this dialog under the "Project types" pane. Select this item to display the installed project templates in the pane on the right.

Choosing Report Server Project enables the Report Designer for the new project. Pay attention to the properties at the bottom of the New Project dialog, and enter an appropriate project name and solution name before you click the OK button. It's important to take your time with these settings because this will result in creating several folders and files. It's easier to name these correctly the first time than to go back and reorganize an existing project. By default, new solution and project folders are stored under the path in the Location field. Note that the default path is under the user-profile-specific My Documents folder. In this case, C:\Users\Administrator\Documents translates to the Documents user folder when the Administrator is logged in to this Windows Server 2008 system. If another user were logged into this computer, the Documents folder would represent a different path. Where you store your projects is up to you, but you should have a plan. Personally, I like to create a Projects folder on my C: drive and keep all the projects organized in subfolders for clients, classes, and other events. I also make it a point to name the solution and the project differently, especially if my solution will include multiple projects.

Designer Utility Windows

The Visual Studio interface, as shown in Figure 5-14, includes several utility windows that can be docked to the inside edges of the main application window or set to float independently. Each of these utility windows can be shown, hidden, or set to auto-hide when the mouse pointer is placed over an icon. This gives you access to features while maximizing report design screen real estate by moving these windows out of the way when not in use.

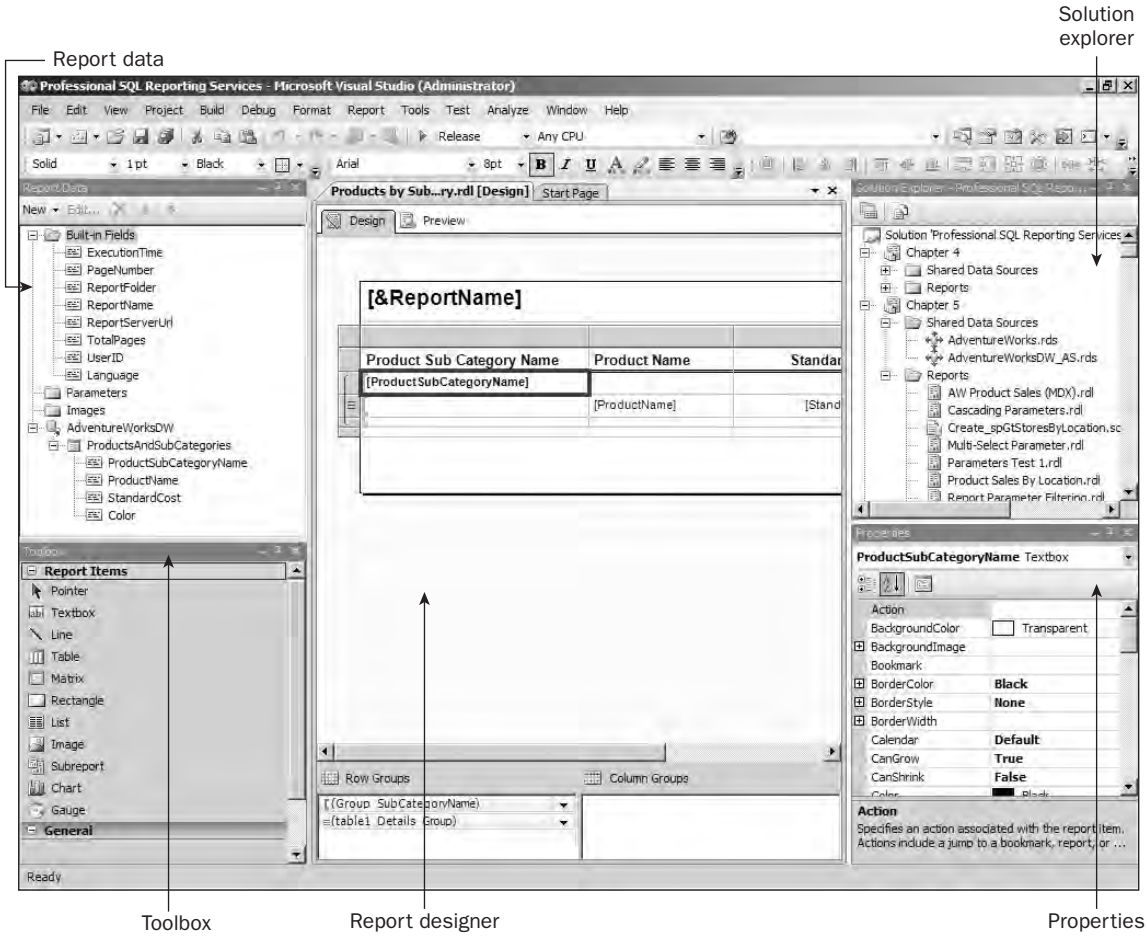

Figure 5-14

The "dockable" utility windows used for report design include:

- **Report Data** — Tree view shows built-in fields, parameters, data sources, datasets, and fields that can be dragged into the Report Designer to assemble report design elements.
- **Toolbox** — Lists report items available to be added to the report design surface.
- **Solution Explorer** — Used to manage all projects and project files. For a report project, contains shared data sources and reports.
- **Properties** — Shows all the properties for the object currently selected in the Report Designer.

The tabbed Report Designer windows are located in the center. Unlike Report Builder 2.0, you can have multiple report designers open at the same time. Each has a tab that allows you to switch between them.

Report Design Elements

Reporting Services uses a modular approach, making the report design process logical and flexible. A typical report consists of data sources, datasets, a report body with headers and footers, and various report items and data regions bound to fields in a dataset — that are placed in the report body.

Data Sources

A *data source* stores connection information used to access databases or other resources that return data to a report. A data source can be either embedded into a report definition or stored as a separate file and shared among multiple reports. Shared data sources are easier to maintain in formal business environments where you have several reports using multiple database servers. If the location or name of a server changes, it will be much more convenient to modify one data source than to open and modify each report. On the other hand, it does require a little more coordination among report designers and administrative personnel.

A data source references a data provider installed on the report designer's computer and on the report server. Data providers enable connectivity to database products and may include .NET native providers, OLE DB providers, and ODBC drivers. SQL Server 2008 installs several data providers for Microsoft and third-party products that will allow reports to retrieve data from different versions of SQL Server, SQL Server Analysis Services, Microsoft Access, Oracle, XML data, and other third-party sources.

Datasets

A *dataset* is typically a query or database object reference used to retrieve a set of records for reporting. The query language used for a dataset is specific to the data provider or processing extension specified in the data source. A dataset must have only one data source. But a data source, whether embedded or shared, can serve any number of datasets. Because the query command text is processed by the data provider, queries must be in the native query language of the data source. For example, a dataset for SQL Server uses the T-SQL query language. When using an Oracle data source, the query is written in Oracle P/L SQL. For SQL Server Analysis Services, queries are written using the MDX query language.

> *Appendixes B, C, and D provide detailed information about specific T-SQL and MDX query language commands.*

Reports

There is a specific hierarchy of objects used to manage all the items and properties within a report. Keep in mind that a report definition is stored in an XML file and all the objects accessed in the Properties window relate to nested XML element tags in the XML structure of a Report Definition Language (RDL) file. The Report object contains properties related to the report itself but not the data regions on the report. These data regions are contained within the report body. Report properties are defined within these categories.

Page Setup

These properties consist of the page scale units, page orientation, height, width, and margins. The Report object also contains the following advanced properties used for adding custom internal or externally referenced programming code, which can be used to extend a report's capabilities:

- **Code** — Custom Visual Basic functions may be written and stored in the report and then called from expressions on any of the properties for this report.
- **References** — Like custom functions stored in the Code element, external code libraries may be referenced and also used in property expressions. The advantage of this approach is that a single code library may be used by multiple reports.
- **Variables** — Custom variables may be defined and used within the report to enable dynamic and advanced functionality. These variables are typically set and used within expressions or custom code functions in the report.

Body

The Body object contains just a few properties and serves as a container for all the data ranges and report items within the report. In the Designer, the report body really is a blank canvas on which you place report items and data ranges. This is a unique approach when comparing Reporting Services with most other reporting products. This is a very flexible approach to report design that encourages free-form report formatting and unconstrained layout. Rather than being constrained to placing items at specific rows or columns, you have the freedom to place items anywhere within the report body. Later, you'll see how the List data region extends this pattern by repeating a free-form region for each record.

Headers and Footers

As a result of the free-form approach, there is no need to designate a specific area to be the report header or footer. Essentially, the *report header* is all the space at the top of the report body before the first data range. Likewise, the space between the last data range and the end of the report body is the *report footer*.

This is because of the way a report is rendered. Like the carriage of a typewriter (or inkjet printer, for those of you who might have no idea what a typewriter is), the report is rendered from the top-left-hand corner, from left to right, and then down the page until it reaches the bottom-right corner of the report body. Any singe report items, like textboxes and images, are rendered once. Data ranges, like tables and matrices, cause rows and columns to be repeated, making the report body grow. It's really quite an elegant approach — and the area of the report body above and below these data ranges is the report header and footer, respectively.

A report, however, does have a specific area defined for the page header and page footer. In the Designer, these areas are enabled using the Header & Footer group on the Home ribbon or the Report menu in the BIDS designer.

Report Definition Language

One very compelling aspect of this product is that the definition of each report is managed in a standard, text-based file format called *Report Definition Language* (RDL). An *RDL file* is an XML document with a standard definition for markup tags that define all the properties for a report. All objects added to a report in the Report Designer and the related property settings result in entries made to the RDL content

for that report. This simple approach makes it easy for independent software vendors and custom solution developers to generate a report definition from a variety of sources and tools. It also makes it easy for report designers and developers to open the report definition in a text editor to make changes outside of the Report Designer. Contrast this with the proprietary binary formats used in other popular reporting products.

As an example, the following is a small snippet of RDL file content describing a textbox report item:

```
<Textbox Name="textbox1">
  <Style>
    <PaddingLeft>2pt</PaddingLeft>
    <PaddingBottom>2pt</PaddingBottom>
    <PaddingTop>2pt</PaddingTop>
    <PaddingRight>2pt</PaddingRight>
  </Style>
  <Top>0.25in</Top>
  <rd:DefaultName>textbox1</rd:DefaultName>
  <Height>0.25in</Height>
  <Width>1in</Width>
  <CanGrow>true</CanGrow>
  <Value />
  <Left>0.375in</Left>
</Textbox>
```

The textbox described by this XML element has default padding properties of 2 points all around. It's located a quarter-inch from the top of the report body and .375 inches from the left edge. It's a quarter-inch tall and one inch wide.

Report Migration and Integration

There are now several applications and products that have the ability to create report definitions for Reporting Services. The extensible RDL allows reports to be created, converted, or modified by custom tools. For example, I've worked with products from Panorama and Cizer that provide custom report designer front-ends within their own web browser–based business intelligence reporting applications. These products put report design capabilities in front of corporate business users without installing complex desktop report authoring software.

Because RDL is simply an XML grammar, building reports can be performed programmatically with relative ease. Because of the complexities of parsing and deciphering proprietary report formats, converting existing reports from other products is more complicated. To date, there are no universal report conversion utilities on the market. Report conversion is a common request from businesses that have already invested in older, expensive reporting products and want to migrate to Reporting Services. Hitachi Consulting offers report migration as a service rather than a product for this reason. If this is an option that you or your company are considering, report migration may be more cost-effective than starting from scratch.

A point to consider is that the fundamental approach for designing reports most effectively may be quite different using different products. A "converted" report, like the one you design in another tool, may not run as efficiently or give you the flexibility to use Reporting Services to its full capability.

Report Design Elements in Detail

Report design elements — or items than can be added to the report body in the Designer — consist of data regions and report items. Technically, a *data region* is a type of report item, but we differentiate between data regions, which consume a whole dataset, and report items, which may consume a single row and field value.

Data Regions

Data regions are discussed and demonstrated in Chapter 8, "Advanced Report Design." In brief, *data regions* are used to render and display the results of an entire dataset query. The *tablix*, which can be designed to behave as a table, matrix, or list, renders rows and columns containing constituent report items. A table, for example, contains a textbox in each cell, used to display a specific row and column value.

Charts don't grow and shrink with the data but translate rows and columns into graphical form. A chart's data region can be configured to render several different standard chart presentations as row, column, line, area, pie, and radial charts, just to name a few.

Textboxes

The textbox item can be used to display data from a dataset, calculations, or expressions, or static data, much like a label control in a Windows Forms project. When you drag fields from the Fields list onto the Report Designer, data-bound textbox items are created. Common expressions can refer to a field in the report.

Figure 5-15 shows a textbox used as a label and another textbox bound to the `LastName` field of the report data source.

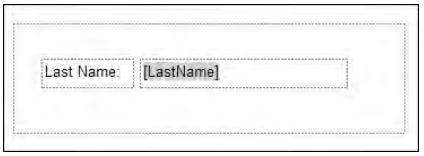

Figure 5-15

Right-click on the textbox, and select Properties from the pop-up menu to display the Text Box Properties dialog, as shown in Figure 5-16.

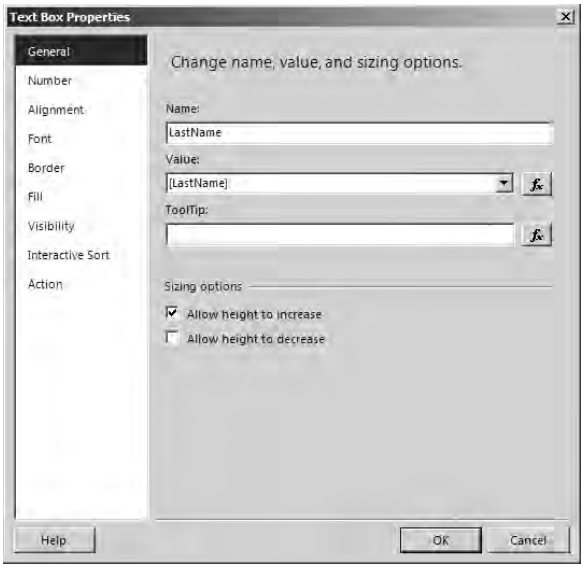

Figure 5-16

You can also view and set properties by using the standard Properties pane located to the right of the Designer. This window may be pinned down or will auto-hide by default. As in Figure 5-17, this window contains quite a bit more detail than the custom Properties window. However, the property information is not as conveniently organized. As a general rule, use the right-click Properties dialog to get to the most common properties, and use the Property pane when you need to set other properties.

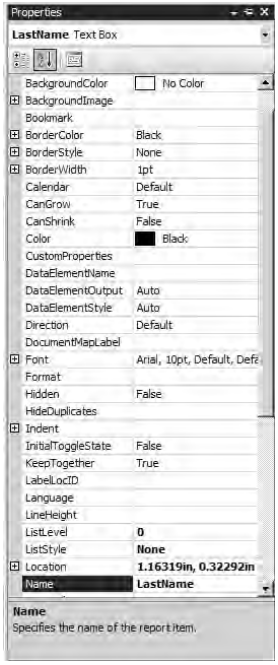

Figure 5-17

Lines

Lines can be drawn in any direction and can be set to a variety of styles and colors. The properties for a line are simple and can be set using the Properties window or Designer toolbar.

It's the job of each format-rendering extension on the Report Server to use the appropriate technique to build each report element output. Because each extension creates a different output format, elements as simple as lines are output in different ways. For example, the Excel renderer will use cell borders. Some clever techniques are used to render lines in HTML. Depending on the need, lines may be rendered as table borders, as a `DIV` tag filled using a JavaScript function, or even using Virtual Reality Modeling Language (VRML) commands.

Rectangles

A rectangle item can have many different uses. A rectangle is simply used to visually separate a region of the report. It can be used to visually contain other items. If items such as textboxes, grids, and so on are placed into a rectangle, all these items can be moved together by simply moving the rectangle. A rectangle may also be used as a data container for data items and can be related to and repeated with a parent container.

Images

Images can be embedded into a report, linked to an external file or URL location, or obtained from a data source. Images can be of the BMP, GIF, JPEG, JPE, PNG, or X-PNG type. Adding an image in the Designer is pretty straightforward. A critical factor is that images are sized and cropped prior to being added to a report. You can resize the image in the Report Designer, but this will not result in a smaller file size. Use a graphics editing tool like the Office Picture Library, the free Paint.NET, Adobe PhotoShop, or Adobe Fireworks to resize or crop the image, and then save it to a new file. You can scale and fit an image to fit the image item container, but it's advisable to use image files that are already the correct size. This conserves disk space, improves performance, and prevents image distortion.

Dropping an image item from the Insert ribbon or Toolbox onto the report will launch the Image Properties dialog (see Figure 5-18). Select the method you want to use; the image can be from a table in the database or a file and may be linked or embedded into the report. Getting external image files to render correctly can be a bit tricky at times owing to file access permissions on the server. If in doubt, it may be easiest to either store the image in the database or embed it into the report definition.

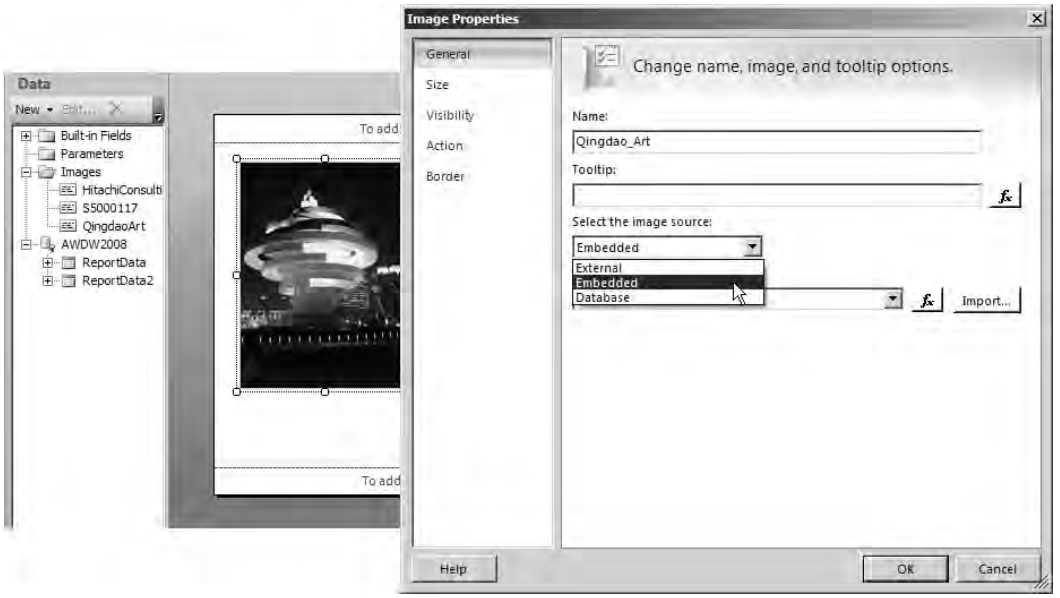

Figure 5-18

Embedded images are encoded as text and stored in the report definition file. Although this increases the size of the report definition file, it can simplify the deployment and configuration. Selecting the Database option will allow you to extract an image stored in an Image or Binary type column within a database, exposed through your dataset. The External option allows you to use a URL to reference an existing file either on the Report Server or elsewhere. If your picture data is stored in the database and the Database option is selected, the Database Field page is displayed in the Wizard. This gives you the option to derive an image file type from the image.

Generally, the JPEG format is most conservative, and PNG graphics are higher quality and more flexible. The GIF and JPEG formats are the most widely used on the Internet and are supported by all web browsers. The GIF or PNG formats support transparency If you need an image to appear non-rectangular (such as an icon and indicator graphic), set backgrounds to white over a white report area, or place the transparent image inside a rectangle item.

Subreports

A *subreport* is a container for another report embedded into a parent report. The subreport can contain practically any other report with its own, independent data source. It can optionally have its data linked to a key or value in the main report, often referred to as a *master/detail report*. Subreports are an important element in complex report designs. Figure 5-19 shows a simple report containing a master record and related detail records in the subreport.

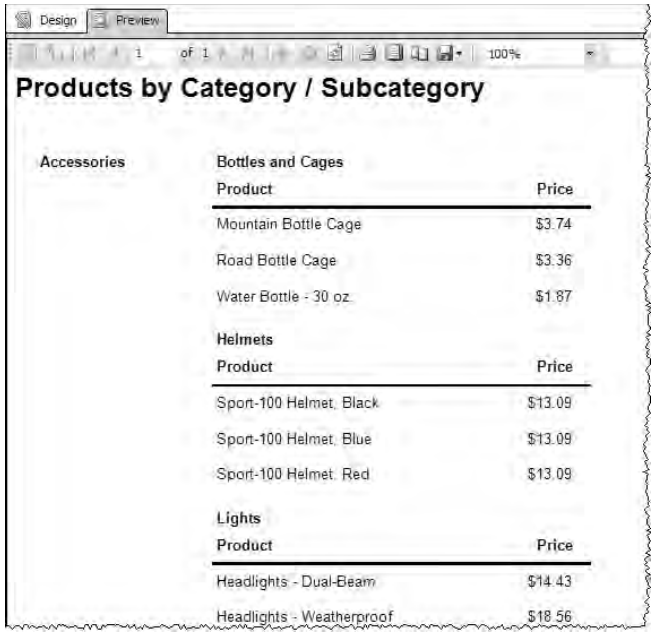

Figure 5-19

The design details of the subreport are not visible in the Designer. The report shown in Figure 5-20 is designed separately and then inserted into the main report as a subreport item.

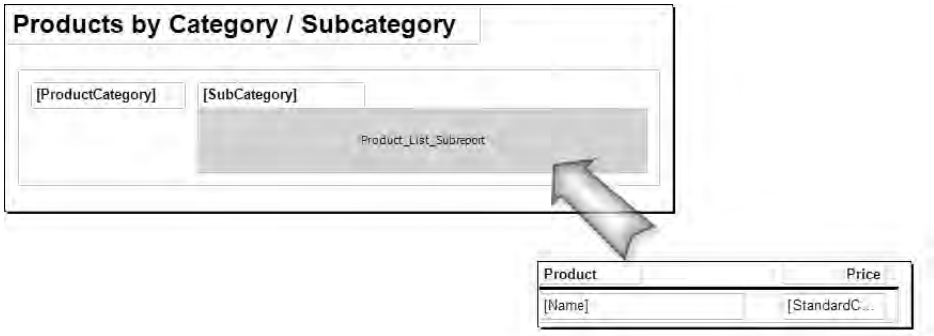

Figure 5-20

Be cautious about using subreports with large results. This report item is appropriate for embedding unrelated content within a report that is bound to a different data source or for detail rows related to few master records. Although this may be a useful technique for consolidating reusable report content, it can be very inefficient when compared with some other techniques. For example, if you create a complex query to return all related data in a single result set, a single table item may be used in place of the subreport and may prove to be more efficient. Significant improvements have been made in the 2008 product for subreport support. In addition to the inherent performance issues, rendering subreports to

different formats has been problematic in the past. For example, a report containing a subreport wouldn't render correctly to an Excel workbook. Most of these types of issues have been resolved in the current product, but if you test the limits of report design and rendering, you will likely find the boundary with subreports first. As a rule, use subreports when necessary, and test them thoroughly when you need to render reports to different formats.

The Tablix

In earlier versions of Reporting Services, there were three different data range items that could be used to group data across rows or columns. The list, table, and matrix had some similar characteristics and capabilities but were three distinctly different objects, each with its own design goals, capabilities, and limitations. Shortly after the release of SQL Server 2005 Reporting Services, the product team began an effort to combine all these capabilities to create a super data-region item to replace the former list, table, and matrix items. Now, the *tablix* serves this purpose — and, yes, as the name suggests, it's a "table/matrix."

Combinations of properties will cause an instance of the tablix to behave and exhibit the characteristics of a list, a table, or a matrix. The advantage of this design is that you are no longer bound by the limitations of any one of these objects. For example, the characteristic of a table is that it has static columns that are typically bound to individual fields in the dataset. You will often define dynamic row groups that expand with each distinct field value according to the row group definition. But, if you decide, after the original design, to add a dynamic column group, this is now a possibility without having to start over with a different data range item.

During report design, you will drag a table, matrix, or list onto the report body, which creates a tablix with present row and columns with these pre-set property values.

Static and Dynamic Columns and Rows

An additional concept introduced with the tablix is that of static and dynamic columns and rows. A *static column or row* is a band of cells with an associated field expression. Take a row group, for example; for records in the dataset, as long as the group expression's field value remains the same, this causes only one instance of the row group to be rendered. All detail rows below this grouping may either be displayed as lower-level detail rows, or field values may be aggregated at the group level. Previous versions of the matrix have always treated columns in this way. Now, in the Tablix, a column may be inserted at any position and designated to behave as a static column (without grouping) or as a dynamic (or grouped) column. By taking this approach, the Tablix has more flexibility than the previous table, list, or matrix data regions.

When the row group spits out a new distinct value, a new row for the group will be rendered, perhaps with lower-detail rows under it or with values to be rolled up and aggregated for inclusion in the row.

By far, the majority of report designers will not concern themselves with adding dynamic columns to change the natural order of things and contend with the delicate balance of a happily working table or matrix, which is why each of these fine, preconfigured items — the table, matrix, and list — stands alone in both Designers. As far as we are concerned, a table is a table, and a matrix is a matrix.

On the off chance that you are brave enough to contend with the natural order of things and rewire a table so that it behaves somewhat like a list or a matrix, this could get interesting. The practical use for these concepts will make more sense as we apply them to an understandable business case.

Row Groups in a Table

A table has a static list of columns — one for each field — and rows are either grouped on a field value or simply output one per row from the dataset. Any number of row groups may be added to a single row, and a row can have no row groups, one, or any number of row groups. This provides a tremendous amount of flexibility to table design. Compared with Reporting Services in SQL Server 2000 and 2005, the model has changed, but the same design patterns are reproducible, with even more options.

The example report shown in Figure 5-21 has row headers with simplified field labels. Five columns are used to return values for the Year, Category, Product, Order Quantity, and Extended Amount fields, respectively. Note that in the Designer, three row groups are defined by dragging fields into the Row Groups pane in the lower-left-hand corner of the Designer window.

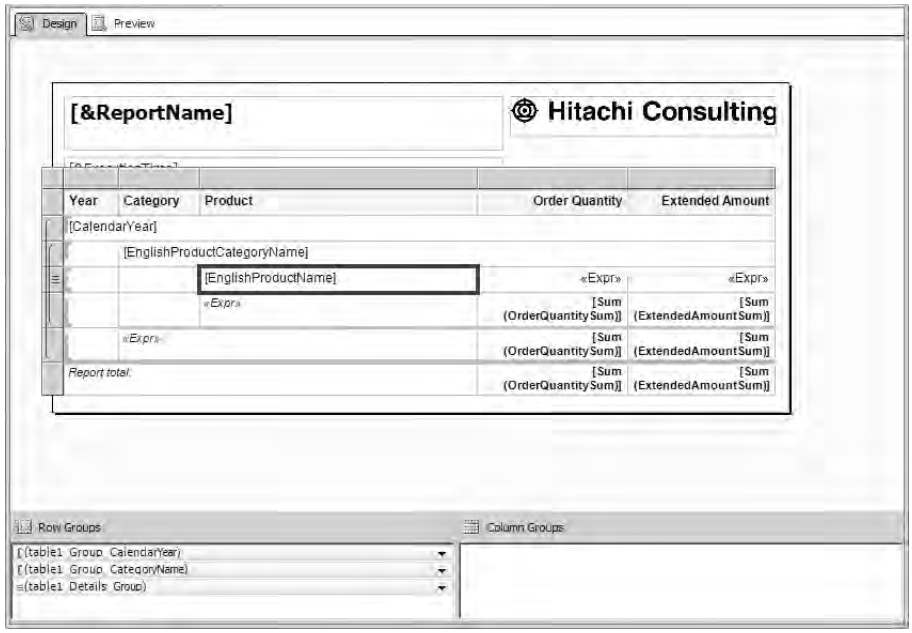

Figure 5-21

Figure 5-22 shows the first page of this report in Preview mode. Note that the 2001 year header precedes the Accessories category, and then three products are listed before a subtotal row for this category.

Excerpted with permission from Professional Microsoft SQL Server 2008 Reporting Services
ISBN: 978-0-470-24201-8, Wrox Press

Product Sales by Year by Category Table Hitachi Consulting

Year	Category	Product	Order Quantity	Extended Amount
2001				
	Accessories			
		Sport-100 Helmet, Black	331	$6,681.73
		Sport-100 Helmet, Blue	353	$7,118.43
		Sport-100 Helmet, Red	319	$6,439.49
		Accessories total:	**1,003**	**$20,239.66**
	Bikes			
		Mountain-100 Black, 38	312	$630,178.13
		Mountain-100 Black, 42	297	$600,613.22
		Mountain-100 Black, 44	315	$636,320.61
		Mountain-100 Black, 48	273	$552,823.36
		Mountain-100 Silver, 38	289	$589,558.27
		Mountain-100 Silver, 42	263	$535,566.42
		Mountain-100 Silver, 44	268	$545,086.40

Figure 5-22

When this type of report is used, a fixed number of columns are rendered. In this example, the five columns are fixed, but rows are generated from row groups that can span many pages in the final report.

Column Groups in a Matrix

A *matrix* is simply a table with dynamically grouped columns — so that each distinct group value outputs a new column. The example shown in Figure 5-23 uses the same query to output groups of data in the same order. Note the groups defined in the Row Groups and Column Groups panes at the bottom of the Designer window. The difference between this and the previous report is that the CalendarYear group is on columns rather than rows.

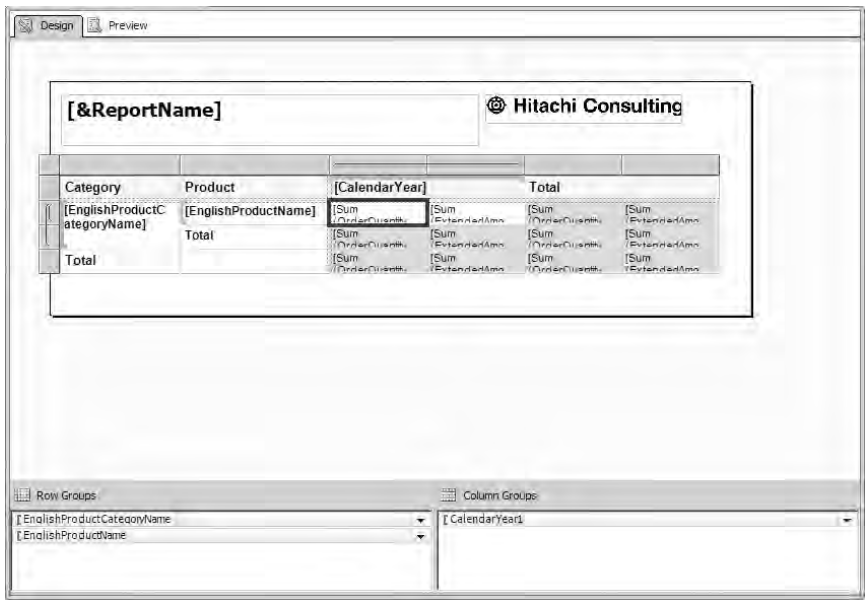

Figure 5-23

When this report is previewed, column headers are generated for each year, and then rows are generated for the category and product field values. Figure 5-24 shows the first page of this report.

Product Sales by Year by Category Matrix

Hitachi Consulting

Category	Product	2001		2002		
Accessories	Sport-100 Helmet, Black	331	$6,682	1,279	$25,189	1,967
	Sport-100 Helmet, Blue	353	$7,118	1,365	$26,822	2,035
	Sport-100 Helmet, Red	319	$6,439	1,188	$23,298	1,810
	Cable Lock			676	$10,105	410
	Minipump			701	$8,383	429
	Bike Wash - Dissolver					1,461
	Hitch Rack - 4-Bike					1,721
	Hydration Pack - 70 oz.					1,291
	Patch Kit/8 Patches					458
	Water Bottle - 30 oz.					1,554
	Total	1,003	$20,240	6,207	$93,797	13,136
Bikes	Mountain-100 Black, 38	312	$630,178	321	$566,526	
	Mountain-100 Black, 42	297	$600,613	292	$520,423	
	Mountain-100 Black, 44	315	$636,321	303	$545,061	
	Mountain-100 Black, 48	273	$552,823	286	$507,082	

Figure 5-24

Chart Essentials

In 2007, Microsoft acquired the latest generation of charting gauge components from Dundas Data Visualization, Inc., the company that originally developed the charting components that were integrated into Reporting Services for SQL Server 2000. The result is that in SQL Server 2008, several new chart types were added, along with many new charting features and capabilities. This is a very capable and easy-to-use charting solution with a variety of available chart types.

Data Aggregation

Numeric data values are always aggregated, usually using the `SUM()` function, along axis groups. This grouping and aggregation is functionally similar to a matrix, with aggregated values representing the intersection points of distinct group values. The difference is that a chart plots or visualizes the values rather than displaying the number in a cell.

Series Groups

The *series group* is the axis associated with the chart's legend, if you choose to include it. In a column chart, for example, series values are displayed in the legend and/or column clusters.

Category Groups

Category groups are the labeled group values that are usually represented with a single column, point, or bar. In a column chart, categories are plotted along the *X*-axis with labels typically along the bottom of the chart.

Chart Type Categories

There are now many different types of charts available. Following are most of the categories and all the essential charts used in basic report design. Some of the advanced charts are showcased in later chapters to demonstrate more advanced features.

Column Charts

Probably the most common and most recognizable chart type is the column chart. The example in Figure 5-25 shows sales data for a given year, grouped by quarter and the sales territory country. The total sales amount is plotted on the category or *Y*-axis (columns) of the chart.

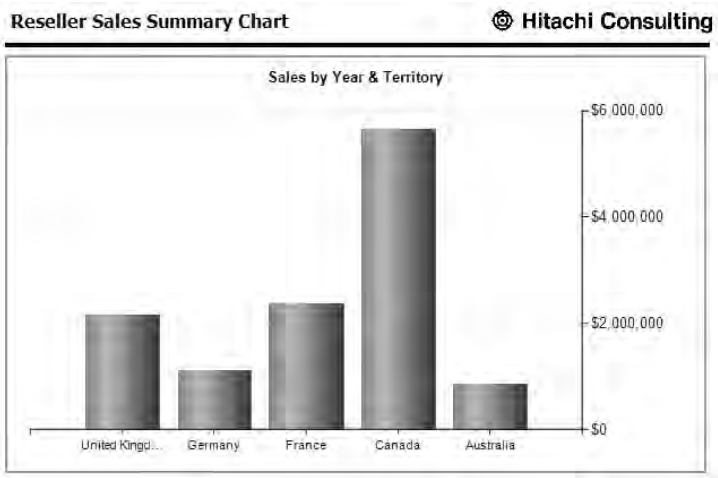

Figure 5-25

Adding a group to the series axis can cause the column chart to show a cluster of columns side by side. The columns are color-coded with a color key shown in the legend. Figure 5-26 shows a category group based on the CalendarYear field with a cluster of two columns. Column clusters may be displayed in a flattened, two-dimensional view like this or arranged along the *Z*-axis when three-dimensional (3D) visualizations are enabled. Figure 5-27 shows the same chart in 3D mode.

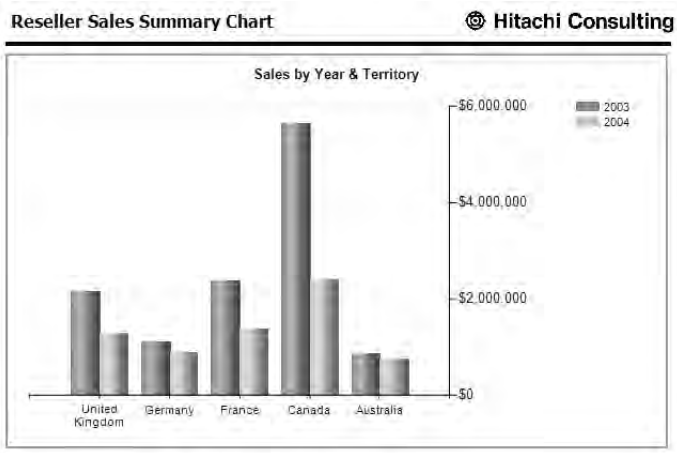

Figure 5-26

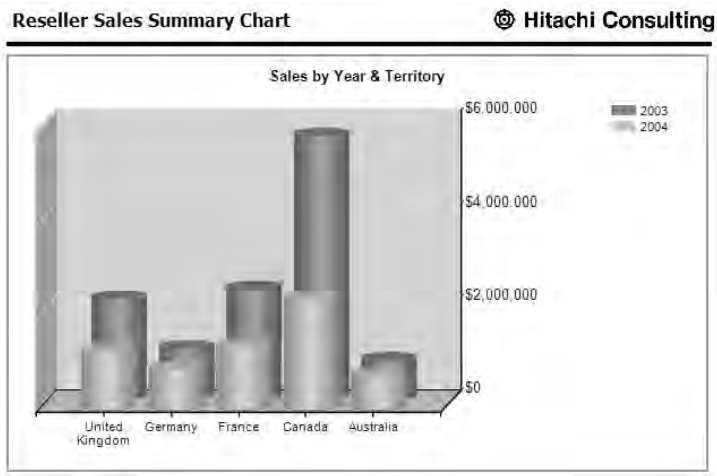

Figure 5-27

One of the powerful features of the chart item is the ability to group data within each axis. Figure 5-28 shows a simple column chart with two field groups on the *X*-axis, representing related categories. In this example, columns are grouped by calendar year and then by the sales territory country.

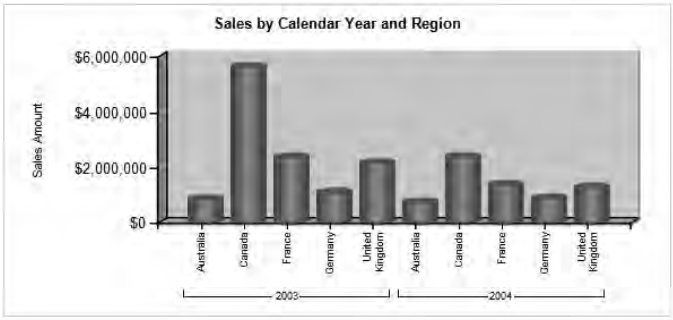

Figure 5-28

A number of these features — including multiple axes, clustering, and nested groups — can be combined to create some interesting and compelling chart visualizations.

Some charts require only one axis group. For example, the pie chart shown in Figure 5-29 uses only a series group, based on the SalesTerritoryRegion field. Pie charts put proportional values into perspective. This type of chart comes in two pastry types: pie and doughnut. Values are presented visually as a percentage of the total for all values in a series. Pie and doughnut chart views may be either *simple* or *exploded*. The exploded presentation may help to visually separate values, especially the smaller slices. These types of charts can be useful for placing values into comparative perspective.

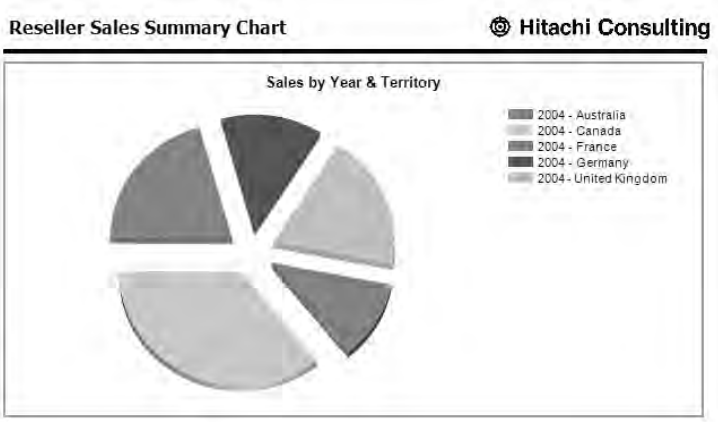

Figure 5-29

All the chart types discussed so far existed in the SQL Server 2000 and 2005 versions of Reporting Services. Following are just a few of the newer chart types introduced in the 2008 product.

Polar and Radar Charts

Figure 5-30 shows one of these new charts. This is a radar chart, one of the new and polar chart types in the new product. This is an interesting visualization that combines elements of a line chart with a pie chart-like format.

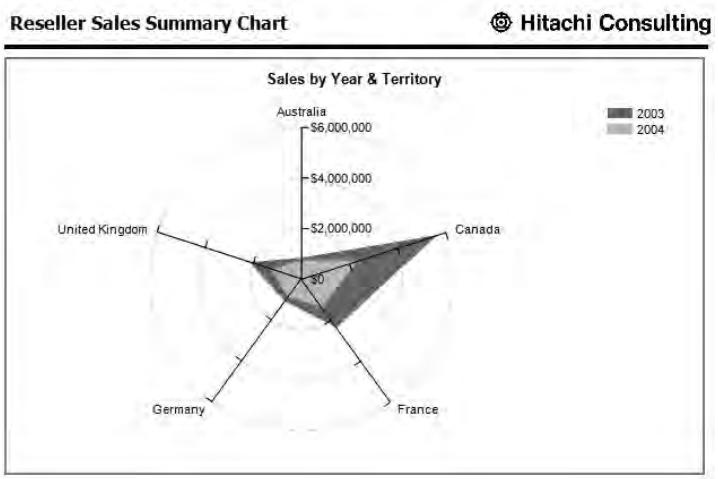

Figure 5-30

Shape Charts

Some special-purpose charts are commonly used to visualize data for certain vertical applications. For example, the funnel chart, shown in Figure 5-31, is commonly used as an illustration of customer sales leads "flowing down" through the funnel in customer relationship management (CRM) solutions, to help manage a sales opportunity pipeline.

Figure 5-31

Figure 5-32 shows another unique shape-type chart that expresses data values in a pyramid. This is commonly used for organizational ranking and progression.

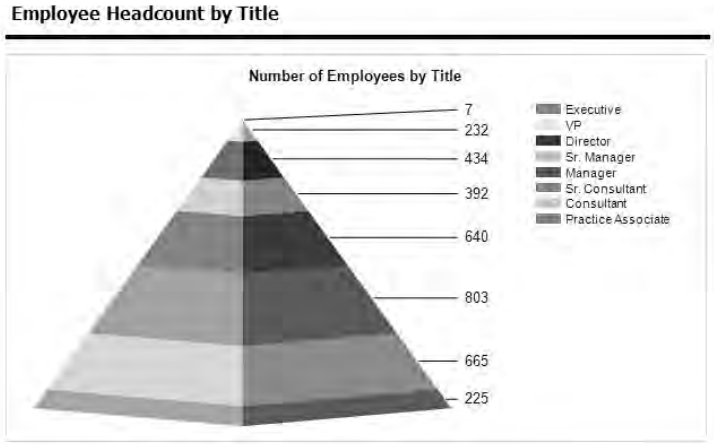

Figure 5-32

Bar Charts

Bar charts and column charts are pretty much the same in functionality. You can tilt your head to the side to get the same view as the other. Figure 5-33 shows the same data from the previous column chart, in a bar chart. When this chart type is chosen, category group values are visualized with values plotted on the *Y*-axis and the value scale along the *X*-axis — a 90-degree rotation of the same column chart.

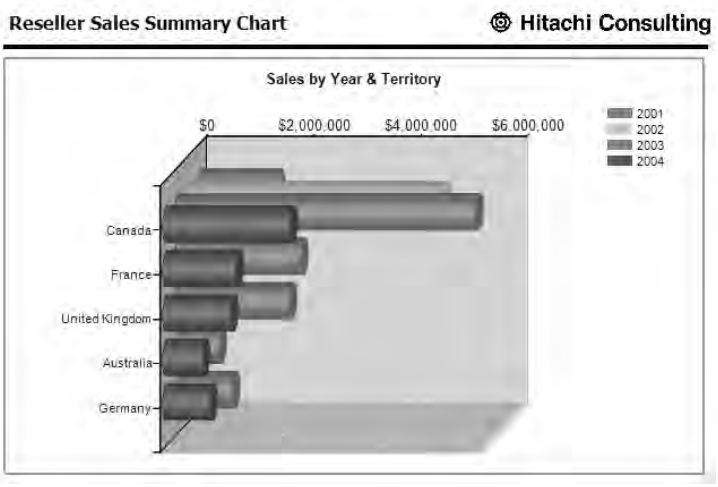

Figure 5-33

In addition to the standard, single-bar view, the stacked view provides a consolidated look at a series of values by using fewer bars or columns. Each bar is like a mini pie chart, where each value in the bar's range is in proportion to the others. A series of related values is stacked in the column to show the aggregate sum of values and their proportional values. A variation, the 100 percent stacked bar or chart, displays each bar with the same height or length as others, regardless of the total values. This type of chart is useful for comparing values within the bar's range but not for comparing the aggregates represented by each bar.

There are many chart variations and special chart types that are not pictured here. Some of the more advanced chart implementations are covered in Chapters 8 and 10.

In addition to the standard report items that ship with the product, application developers and third-party companies can create custom report items (CRI) that can be installed and used in the Designer. You are likely to see more CRI suites for Reporting Services that will add even more capabilities to your reports.

Gauges

Humans have been measuring things for a long time, and we use a lot of different ways to keep track of things like air and water pressure, speed, torque, and temperature. Whether using a ruler, meter stick, progress bar, or the dashboard of a car, we are all accustomed to using a variety of standard gauges. In the right situations, gauges are an ideal visualization for business data using a familiar display metaphor. Many users will immediately identify with important information displayed using a variety

of linear and radial gauge indicators that apply context and importance to measurements, status indicators, and business metrics. The new gauge report items introduced in Reporting Services 2008 enable you to control virtually every aspect of gauges, including background colors and shading, borders, scales, pointers, and markers. Practically any characteristic you've seen in a physical gauge or meter is possible. You can even reproduce the tachometer and speedometer in a Mercedes S550, the pressure value on a fire extinguisher, or the Sick Bay vital sign monitors in the Star Ship Enterprise.

Although there are many properties, most are fairly easy to work with and to discover with just a little bit of guidance. Figure 5-34 shows the chart types dialog, which provides a thumbnail preview of all the available gauge types.

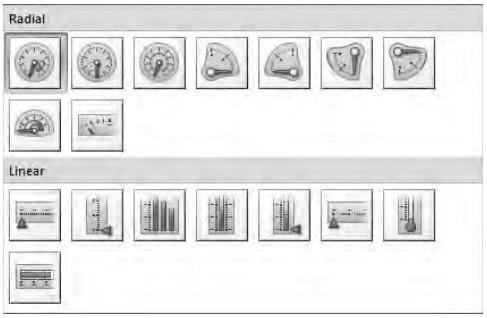

Figure 5-34

When I started working with the Dundas Gauges product a few years ago, before it was integrated into Reporting Services as a standard feature, I spent several hours configuring the interface to get each one just right, simply because this tool gave me so much control. I warn you that this cool new report capability can lead to late night obsessive report design if you have a propensity toward design perfection.

Like the new chart data range, a single gauge can contain several gauge areas. This essentially lets you create an entire dashboard with one gauge item. Each gauge area has its own scales, pointers, ranges, and markers.

Scales

The *scale* is the set of markers and reference numbers around the dial of a radial gauge or along the range of a linear gauge. Any number of scales can be added to a single gauge.

Pointers and Markers

In a radial gauge, pointers are typically needles and arrows or small "tick" marks that can extend from the center of the gauge and point to a scale, or simply indicate certain points on a scale. Several pre-set pointers and markers are available in different shapes. Each can be resized and, like every other element, can be set with solid or gradient shading.

Ranges

Ranges typically are used to provide some sort of context and are typically displayed on or near the scale, behind pointers and markers. A range can be used to indicate that a value is within acceptable or exceptional boundaries. Ranges can be color-coded, tapered, or shaded with solid or gradient fills.

Radial Gauges

Radial gauges can be circular or partially circular, with one or more pointers extending from a central fulcrum. The simplest form of gauge, shown in Figure 5-35, has a single scale and pointer. The maximum scale value in this example is set with an expression to indicate the total number of sales units for a calendar year. The pointer value gets its value from an expression to show the total units in a quarter, thus showing the proportion of the quarter units to the year.

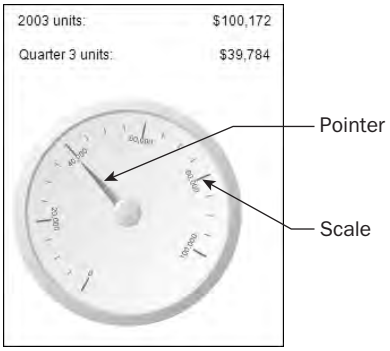

Figure 5-35

Figure 5-36 shows an example of the same gauge with a second pointer added. The larger arrow pointer indicates units sold for the quarter, and the smaller pointer shows units for the month on the same scale as the previous example.

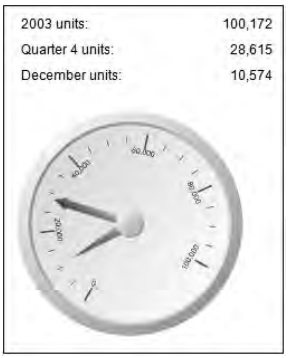

Figure 5-36

Linear Gauges

Linear gauges can be arranged vertically or horizontally in a variety of formats. Markers are typically used in the place of radial pointers but behave in much the same way. The example in Figure 5-37 shows a linear gauge with a single marker. This gauge has three color-coded ranges, which are used to indicate threshold values along the scale.

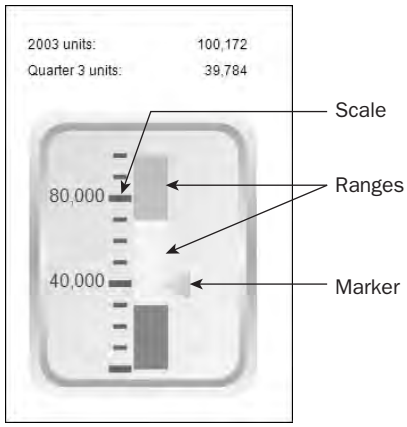

Figure 5-37

Gauges become very useful when they are used to compare a group of related values side by side. For example, Figure 5-38 shows a composite report designed by embedding a single gauge into a tablix with repeating columns.

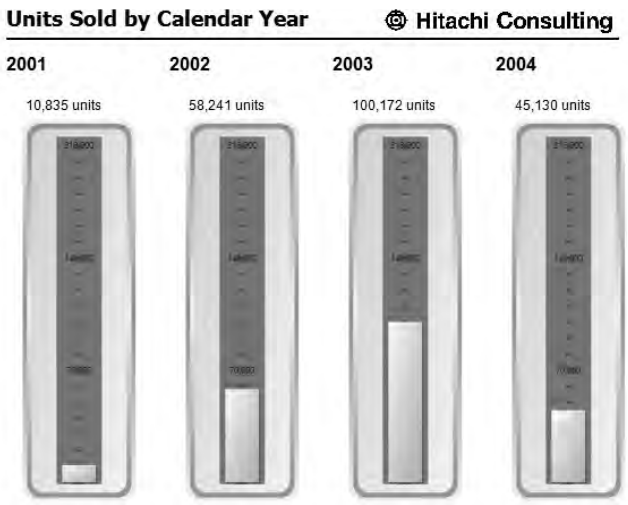

Figure 5-38

Summary

In this chapter, you learned about the essentials of report design and were introduced to the report building blocks. You were introduced to the report design tools, which include Report Builder 2008, for information workers to create their own ad hoc and server-hosted reports. For the corporate information technology staff, you also learned how advanced report design capabilities continue to be part of the Microsoft integrated development environment, through Business Intelligence Development Studio and Visual Studio. These tools allow application developers and solution teams to share report source code and manage reports as part of a business intelligence project or corporate solution.

You used the Report Wizard to create a simple report with little effort. Like the other report design tools, this defines a report as an RDL file. The visual report design tools provide a graphical user environment to manage the objects and properties in an RDL file.

You learned that the building blocks of a report are the report, report body, data regions, and report items. This modular object approach provides greater flexibility to report design and management. Report items such as a textbox contain properties that control the value and formatting of report data and content. Nearly all properties can be set using expressions to provide dynamic behavior and formatting.

The tablix is the new replacement for the data regions in previous Reporting Services versions. This one object provides greater flexibility and capabilities than the old table, matrix, and list. The Report Designer offers these three objects as predefined instances of the tablix data region, used to present entire sets of data returned from dataset queries.

A table is used to present grouped rows of columnar data — where columns correspond to query fields, and rows are rendered from the records returned from a data table or query. These values may be grouped, with separate headers, footers, and subtotals. The matrix is a pivot table and, like a table, renders grouped rows with headers and summaries. But a matrix also groups values into columns, which may also be summarized. A list is a simple data region that repeats a section of the report based on a row grouping. It is typically used to repeat other data regions or report items presented in a free-form layout.

Subreports allow you to combine the functionality of an existing report into another report. This makes it easy to combine prebuilt reports into composite reports with reusable functionality. Using a list or other data region, a subreport may be used to integrate data sources and to federate data into repeating groups and sections.

The new charting engine includes many new capabilities including multiple chart areas — used to create composite charts. Several new chart types have been added in the 2008 product, making this a much more powerful tool than before. Gauges were also added to the toolbox, enabling compelling new visualizations and options to present linear data in creative ways.

The chapters that follow expand these topics. You will learn to build advanced reporting solutions using the elements discussed in this chapter.

Contents

Foreword xxv
Introduction xxvii

Part I: Getting Started

Chapter 1: Introducing Reporting Services 3

Not Your Father's Reporting Tool 4
Who Uses Reporting Services? 5
Application and Reporting Technology 7
Blurring the Application/Reporting Line 8
Information, Now! 8
Solution Types 9
Out-of-the-Box Reports 10
Server-Based Reports 11
User-Designed and Ad hoc Reports 11
Report Design Tools 12
Report Builder 2.0 12
Business Intelligence Development Studio 13
Designing Reports 13
Simple Application Integration 15
Launching Reports from an Application 15
User Interaction and Dynamic Reporting 16
Intranet and Internet Report Access 17
Seamless Application Integration 17
Web Application Integration 18
Portal Integration 19
Windows Application Integration 19
Managing and Customizing the Report Server 20
Summary 21

Chapter 2: Business Intelligence Solutions 23

Reporting Tool Options 24
Scalable Architecture 24

Corporate Reporting **24**
Department and Personal Reporting **25**
Ad hoc and Self-Service Reporting **26**
BI Solution Components **26**
Report Data Sources 26
The BI Data Process 28
The BI Maturity Continuum 29
Report Types 30
Data Complexity and Report Performance 35
Summary **38**

Chapter 3: Reporting Services Installation **39**

The Basic Installation **39**
Installing Reporting Services 40
Installing the Reporting Services Samples and SQL Server Sample Databases 58
The Enterprise Deployment **59**
SQL Server Editions 60
Named Instances 62
Topology 62
Modes 63
Installation Options 63
Command-Line Installation 64
Summary **64**

Chapter 4: Reporting Services Architecture **65**

The Reporting Life Cycle **66**
Authoring 66
Management 67
Delivery 67
Reporting Services Tools **67**
Report Designer 67
Report Builder 68
Report Builder 2.0 68
Third-Party Authoring Tools 68
Report Manager 68
SharePoint Libraries and Web Parts 69
Reporting Services Configuration Manager 69
SQL Server Management Applications 69
Command-Line Utilities 70
HTML Viewer 70

Report Viewer Control 70
Reporting Services Web Service 71
Reporting Services Windows Service 72
HTTP.SYS and the HTTP Listener 73
The Security Sublayer 74
Report Manager and the Web Service 75
Core Processing 75
Service Management 75
WMI and the RPC Interface 76
Reporting Services Processors and Extensions 77
The Report Processor 78
Data Processing Extensions 79
Report Items 80
Rendering Extensions 81
The Scheduling and Delivery Processor 83
Delivery Extensions 83
Reporting Services Application Databases 84
ReportServer 84
ReportServerTempDB 86
Summary 86

Part II: Report Design

Chapter 5: Basic Report Design 89

Report Design 101 90
Report Designers 91
Report Builder 2.0 92
Viewing and Setting Properties 98
Report Design with Report Builder 2.0 99
Integrated Development Environment 104
Report Design Elements 108
Data Sources 108
Data Sets 108
Reports 108
Report Definition Language 109
Report Migration and Integration 110
Report Design Elements in Detail 111
Data Regions 111
Textboxes 111
Lines 113
Rectangles 113

Images 113
Subreports 114
The Tablix 116
Chart Essentials 119
Chart Type Categories 120
Gauges 124
Summary 128

Chapter 6: Repor t Layout and F ormatting 129

Report Layout Types 130
Tabular Reports 130
Matrix Reports 131
List Reports 131
Chart Reports 131
Gauge Reports and Dashboards 132
Page Layout 132
Designing Tabular Reports 134
Defining Table Groups 138
Adding Totals and Subtotals 154
Formatting Report Data 154
Introduction to Dynamic Formatting 155
Designing Multicolumn Repor ts 156
Designing Matrix Repor ts 156
Designing Char t Repor ts 162
Designing Gauge Repor ts 170
Converting Repor ts from Other F ormats and Products 173
Importing Access Reports 174
Designing for Extensibility 174
Summary 176

Chapter 7: Designing Data Access 177

Federating Data Sources 179
Linked Servers and Ad hoc Distributed Queries 179
Business Intelligence Repor ting 180
Reporting for Relational Data 182
Data and Quer y Basics 182
Data Sources 182
Data Sources and Query Languages 188
T-SQL Quer y Design 189
Filtering Techniques 197
Filtering a Query 199

Parameter Concepts 199
Filtering Data with Query Parameters 202
Using Stored Procedures 214
Using Other Data Sources 218
Microsoft Access 219
Microsoft Excel 222
Oracle P/L SQL 223
SyBase Adaptive Server 224
Best Practices 225
Summary 225

Chapter 8: Advanced Repor t Design 227

Configuring Headers and Footers 228
Aggregate Functions and Totals 230
Adding Totals to a Table or Matrix Report 232
Creating Repor t Templates 235
Creating Composite Repor ts 238
Anatomy of a Textbox 239
Padding and Indenting 240
Embedded Formatting 240
Designing Master/Detail Reports 244
Groups and Data-Set Scope 244
Designing Subrepor ts 252
Federating Data with a Subreport 253
Navigating Repor ts 259
Creating a Document Map 259
Links and Drill-through Reports 261
Reporting on Recur sive Relationships 263
Using Expressions and Custom Code 268
Using the Expression Builder 269
Calculated Fields 271
Conditional Expressions 273
`IIF()` Is Your Friend 274
Using Custom Code 277
Chart Repor ts 282
Chart Types 283
Column Charts 285
Area and Line Charts 288
Pie and Doughnut Charts 288
Bubble and Stock Charts 290
The Anatomy of a Chart 291

Chart Design Basics **292**
Adding a Data Series 298
Adding a Secondary Axis 299
Using Multiple Chart Areas 302
Summary 304

Part III: Business Intelligence Repor ting

Chapter 9: Repor ting with Analysis Ser vices **309**

Why Analysis Ser vices for Repor ting? **309**
Using Repor ting Ser vices with Analysis Ser vices Data **311**
Multidimensional Expression Language **312**
MDX: Simple or Complex? 312
The MDX Builder 313
Non-Additive Measures **336**
Using the Aggregate Function 338
MDX Proper ties and Cube F ormatting **339**
Drill-Through Repor ts **340**
Cube Report Actions 342
Parameter Safety Precautions 342
Best Practices and Pro visions **342**
Summary 343

Chapter 10: Repor t Solution P atter ns and Recipes **345**

Reporting Project Requirement Guidelines **346**
Key Success Factors 346
Reporting on Existing Data Sources 348
Building an End-to-End Reporting Solution 348
Report Specifications 350
Development Phases 351
Migrating and Converting Reports 354
Working with the Strengths and Limitations of the Architecture 355
Report Recipes **357**
Multiple Criterion Report Filtering 357
Customizing Gauges with External Images 360
Creating a Business Scorecard 362
Reporting on SharePoint 3.0 List Data 366
Report Localization 371
Dynamic Grouping 377

Dynamic Fields and Columns 380
Using Advanced and Third-Party Controls for Parameter Selection 382
Creating Sparklines 384
Summary 386

Part IV: Enabling End-User Reporting with Report Builder 1.0

Chapter 11: Report Models 391

Getting Started 391
Creating the Report Model Data Source 392
Building a Data Source View 395
Manipulating the Data Source View 398
Building the Report Model 404
Using the Report Model Wizard 404
Working with Reporting Services Report Models 408
Deploying the Report Model 413
Building Report Models from Analysis Services Databases 414
Summary 416

Chapter 12: Report Builder 1.0 419

Report Model Overview 419
Accessing Report Builder 1.0 420
Building Reports 421
Table Layout 422
Matrix Layout 425
Chart Layout 430
Formatting Reports 434
Adding Text 434
Adjusting Column Width and Alignment 435
Modifying Font and Background Color 436
Filtering and Sorting Reports 438
Filtering Reports 438
Sorting Reports 443
Adding Calculations with Expressions 443
Administration 446
The Future of Report Builder 1.0 447
Summary 448

Part V: Administering Reporting Services

Chapter 13: Content Management 451

Using Report Manager 452
Content-Management Activities 456
Folders 457
Shared Data Sources 461
Report Models 463
Reports 466
Report Resources 480
Shared Schedules 480
Item-Level Security 483
Content-Management Automation 492
The `RS` Utility 492
Reporting Services Scripts 494
The RSScripter 500
Summary 501

Chapter 14: Report Server Administration 503

Security 503
Account Management 504
System-Level Roles 508
Surface Area Management 510
Backup and Recovery 511
Application Databases 511
Encryption Keys 513
Configuration Files 515
Other Items 516
Monitoring 516
Set-up Logs 516
Windows Application Event Logs 517
Trace Logs 517
Execution Logs 520
Performance Counters 522
Server Management Reports 527
Configuration 528
Memory Management 528
URL Reservations 529
E-mail Delivery 531
Rendering Extensions 533

My Reports 535
Summary 537

Part VI: Reporting Services Integration and Custom Programming

Chapter 15: Integrating Reports into Custom Applications 541

URL Access 542
URL Syntax 542
Accessing Reporting Services Objects 543
Reporting Services URL Parameters 549
Passing Report Information through the URL 555
Programmatic Rendering 557
Common Scenarios 558
Rendering through Windows 559
Rendering to the Web 578
Using the `MicrosoftReportViewer` Control 587
Embedding a Server-Side Report in a Windows Application 590
Summary 594

Chapter 16: Integrating Reports with SharePoint 595

The SharePoint Technologies 596
Windows SharePoint Services (WSS) 597
Microsoft Office SharePoint Server (MOSS) 597
SharePoint Web Parts 598
Native Mode 598
Installation 598
Report Viewer 600
Report Explorer 601
Integrated Mode 603
Installation/Configuration 603
Publishing Reports 611
SharePoint Site Settings 620
Report Models 621
Report Builder 1.0 624
Report Management 626
SQL Server Reporting Services Report Viewer for Integrated Mode 629
Architecture 631
Native Mode versus Integrated Mode 632
Summary 634

Chapter 17: Extending Reporting Services 635

Extension through Interfaces 637
What Is an Interface? 637
Interface Language Differences 638
Data Processing Extensions — A Detailed Look 640
Creating a Custom Data Processing Extension 643
The Scenario 643
Creating and Setting up the Project 644
Creating the `DataSetConnection` Object 647
Creating the `DataSetParameter` Class 655
Implementing `IDataParameter` 656
Creating the `DataSetParameterCollection` Class 658
Creating the `DataSetCommand` Class 660
Creating the `DataReader` Object 673
Installing the `DataSetDataProcessing` Extension 677
Testing the `DataSetDataExtension` 680
Summary 684

Appendix A: RDL Object Model 685

Appendix B: T-SQL Command Syntax Reference 693

Appendix C: T-SQL System Variables and Functions 715

Appendix D: MDX Reference 735

Index 759