Microsoft

Microsoft SQL Server 2012 Reporting Services

Professional

Stacia Misner

Microsoft SQL Server 2012 Reporting Services

Use your SQL Server skills to create and manage key data reports

Deliver interactive business intelligence reports using SQL Server 2012 Reporting Services, and help facilitate real-time decision making throughout the enterprise. SQL Server expert Stacia Misner shows you in detail how to design, manage, and access custom reports that capture data from multiple sources. This book is ideal for experienced SQL Server developers, whether you're new to Reporting Services or upgrading from an earlier version.

Discover how to:

- Use the latest features, including the Power View ad hoc reporting tool
- Design reports by understanding how your audience accesses and uses data
- Add interactive features to help end users sort and filter data
- Create rich data visualization with charts, gauges, indicators, and data maps
- Configure your report server, provide security, and monitor report contents
- Use SharePoint or Report Manager to search and access reports online
- Create a library of report parts that clients can use for ad hoc reporting
- Build and manage reports using Reporting Services as a development platform

Download sample files at:

http://aka.ms/SQL2012RS/files
To install sample files, see the Introduction.

microsoft.com/mspress

ISBN: 978-0-7356-5820-2

U.S.A. $61.99
Canada $65.99
[*Recommended*]

Programming/Microsoft SQL Server

About the Author

Stacia Misner, MCITP, MCTS, is a consultant and trainer specializing in Microsoft business intelligence solutions. With more than 20 years of IT experience, she has written several books on SQL Server and earned a SQL Server Analysis Services Maestro certification.

Microsoft

Microsoft

Microsoft SQL Server 2012 Reporting Services

Stacia Misner

ISBN: 978-0-735-65820-2

1 2 3 4 5 6 7 8 9 LSI 8 7 6 5 4 3

Printed and bound in the United States of America.

Microsoft Press books are available through booksellers and distributors worldwide. If you need support related to this book, email Microsoft Press Book Support at *mspinput@microsoft.com*. Please tell us what you think of this book at *http://www.microsoft.com/learning/booksurvey*.

Acquisitions and Developmental Editor: Michael Bolinger

Production Editor: Holly Bauer

Technical Reviewer: Jeff Rush

Copyeditor: Richard Carey

Indexer: Ellen Troutman Zaig

Cover Design: Twist Creative • Seattle

Cover Composition: Karen Montgomery

Illustrator: Rebecca Demarest

To Gerry, best of friends forever.

—Stacia Misner

To my loving mother, who got me into this mess, in every sense of the phrase.

—Erika Bakse

Contents at a Glance

Contents

PART VII USING REPORTING SERVICES AS A DEVELOPMENT PLATFORM

Introduction

Microsoft Reporting Services is the component of Microsoft SQL Server 2012 that provides an enterprise-ready and extensible presentation layer for the Microsoft business intelligence platform. In its fifth release, Reporting Services continues its support for the three stages of the reporting life cycle, adds a new option for self-service reporting, and provides tighter integration with Microsoft SharePoint 2010 technologies. This book provides in-depth explanations of these new features and also includes comprehensive coverage of all other features that have made Reporting Services a popular choice for implementing a reporting platform in organizations of all sizes since its initial release in 2004.

We have written previously about Reporting Services as part of the Step by Step series published by Microsoft Press. Although we have borrowed many of our explanations for performing tasks from our previous work, the format of the current book allows us to expand on these explanations, explore a variety of techniques, and describe aspects of Reporting Services that are not fully documented elsewhere. We want this book to be useful to anyone new to Reporting Services as well as to seasoned professionals who need a good reference for properties that are not used often but come in handy when the need arises.

Note A few months after the manuscript for this book was completed, the release of Microsoft SQL Server 2012 Service Pack 1 (SP1) and Microsoft SharePoint 2013 introduced some new features that modify or replace some of the topics we cover in this book related to installation and to Power View. If you are running Reporting Services in native mode, these new features do not affect you. If you are running Reporting Services in SharePoint integrated mode with Microsoft SharePoint Server 2010, the installation of SP1 adds pie charts, maps, and a variety of other features about which you can learn more at *http://officepreview.microsoft.com/en-us/excel-help/whats-new-in-power-view-in-excel-2013-and-in-sharepoint-2013-HA102901475.aspx* to enhance the coverage in Chapter 30, "Preparing to use Power View," and Chapter 31, "Using Power View." If you prefer to use Microsoft SharePoint Server 2013 to run Reporting Services in integrated mode, replace the instructions we provide in Chapter 4, "Installing Reporting Services," with the instructions available at *http://msdn.microsoft.com/en-us/library/jj219068.aspx*.

Who should read this book

Because Reporting Services is a platform of technologies rather than a single application, a variety of people serving in different roles and having different skill sets all need to learn how to work with some aspect of Reporting Services. Whether you're a report developer, IT administrator, or business user, you can use this book to learn how to perform the tasks for which you are responsible. If you're completely new to Reporting Services, this book teaches you the fundamental concepts required to build, manage, and access reports. You need no prior experience with Reporting Services to use this book successfully. If you're already familiar with an earlier version of Reporting Services, this book highlights new features in SQL Server 2012 and provides information about many seemingly mysterious properties and functions that have long been part of Reporting Services.

Assumptions

Throughout most of the book, we assume you currently know nothing about Reporting Services. We make an exception in Chapter 1, "What's new in SQL Server 2012 Reporting Services," to provide you with a jump-start into the new features if you're already using an earlier version of the product. We also assume that you know something about relational databases, but we don't require you to know how to write a query. We have samples that you can use to explore the features of Reporting Services without writing queries. There are some chapters that application developers can use to extend the functionality of Reporting Services, and we expect that you are familiar with Visual Basic or Visual C# before you read those chapters.

Organization of this book

Even if your primary responsibility with Reporting Services requires you to focus on only one component, you benefit from an understanding of all the capabilities that Reporting Services supports. For this reason, if you have time, you should read through the book from beginning to end for the best experience.

Because this book can be used by readers with different needs, you can also choose to read through only the chapters that focus on your area of responsibility. Each part of the book contains chapters related to a different part of the reporting life cycle. However, unlike our previous Step by Step books, in which we used procedures to build up your skills progressively, we have written each chapter without interdependencies. By using this approach, we were able to fully explore options available for performing a

task or configuring a feature without the constraints of developing a linear, task-based sequence of steps to follow.

In Part I, we introduce Reporting Services, starting with the new features in SQL Server 2012 in Chapter 1, and continuing with an explanation of its usage and architecture in Chapters 2 and 3. We also explain how to install Reporting Services in either native mode or SharePoint integrated mode in Chapter 4. We conclude this part of the book with Chapter 5, which provides an overview of Report Builder as a user-friendly report development tool and introduces the deployment and report access stages of the report life cycle.

Part II focuses on report development. We start first by providing some guidelines for preparing to develop reports in Chapter 6 and then explain how to work with report projects in SQL Server Data Tools in Chapter 7. Chapter 8 covers all the different ways that you can retrieve data for your reports. Finally, in Chapter 9, you learn how to create the report itself by using different report items. In Chapter 10, we describe the nuances of the tablix in detail so that you can better take advantage of its flexibility. Then in Chapters 11 and 12 we explore the many different ways that you can create expressions and work with functions to change the behavior and appearance of reports at runtime. Chapter 13 describes the interactive features you can add to reports, while Chapter 14 describes another type of interactivity possible through the use of parameters and filters. In Chapter 15, we discuss many of the options you have for controlling pagination in reports. Then we wrap up report development in Chapter 16, where we explain how to use Analysis Services as a data source for your reports.

In Part III, we continue our focus on report development but with an emphasis on data visualizations. Chapters 17 and 18 explain everything you need to know to successfully develop charts. Chapters 19 and 20 show you how to display key performance data as gauges and indicators. Then Chapters 21 and 22 detail spatial data visualization techniques with the map feature.

Part IV moves on to the management stage of the reporting life cycle. In Chapter 23, you learn the different ways to make reports available to users on the report server. After reports are deployed, you can secure access to reports by using the techniques described in Chapter 24. Chapter 25 addresses all the remaining administrative tasks, such as configuration of the report server, backup and recovery procedures, and report server monitoring.

Part V includes chapters describing the user-focused stage of the reporting life cycle, report access. In Chapter 26, you learn the basics of working with Report Manager or SharePoint to view reports online. You can also set up subscriptions to receive reports in other ways, as you learn in Chapter 27. You can also receive notifications about changes to report data, as described in Chapter 28.

For a more interactive reporting experience, Part VI covers ad hoc reporting. Chapter 29 explains a simple approach to ad hoc reporting through the use of report parts. Then Chapters 30 and 31 introduce the newest ad hoc reporting tool in Reporting Services, Power View.

If you need Reporting Services to provide additional functionality, you can customize it through simple scripts or complex applications. The chapters of Part VII describe the various options you have for each stage of the reporting life cycle. Chapter 32 begins with an explanation of Report Definition Language, and then Chapter 33 shows you how to use code in reports, either as custom functions or as custom assemblies, to expand the capabilities of Reporting Services, such as retrieving data from an unsupported source. Chapter 34 shows you examples of how to use scripts or applications to manage the report server, and Chapter 35 caps off the book with techniques for retrieving reports programmatically.

Conventions and features in this book

This book presents information using conventions designed to make the information readable and easy to follow:

- Boxed elements with labels such as "Note" provide additional information or alternative methods for completing a step successfully.
- Text that you type (apart from code blocks) appears in bold.
- A plus sign (+) between two key names means that you must press those keys at the same time. For example, "Press Alt+Tab" means that you hold down the Alt key while you press the Tab key.
- A vertical bar between two or more menu items (for example, File | Close) means that you should select the first menu or menu item, then the next, and so on.

System requirements

You will need the following hardware and software to install the code samples and sample database used in this book:

- Windows Vista SP2, Windows 7, Windows Server 2008 SP2 or greater, 32-bit or 64-bit editions.
- At least 6 GB of free space on disk.
- At least 1 GB of RAM.
- A 1.0GHz x86 or 1.4GHz x64 processor or better.
- At minimum, an instance of SQL Server 2012 Database Engine and Reporting Services plus client components. Optionally, you can also install SQL Server 2012 Analysis Services in multidimensional mode and another instance in tabular mode. Reporting Services can be a native-mode or SharePoint integrated-mode installation. Full instructions about how to install the required components are provided in Chapter 4, "Installing Reporting Services."

Code samples

The database used for examples in this book is based primarily on Microsoft's Adventure Works DW 2012 sample database and, in a few cases, on some sample databases we provide. We recommend that you download the database from the link below rather than use your own copy of Adventure Works to reproduce the examples in this book.

All sample projects and the sample databases can be downloaded from the following page:

http://aka.ms/SQL2012RS/files

Follow the instructions to download the SSRS2012Samples.zip file and the sample database.

Installing the code samples

Follow these steps to install the code samples on your computer so that you can follow the examples in this book:

1. Unzip the samples file onto your hard drive.
2. Each chapter has its own directory containing code samples. In many cases, this takes the form of a report server project that must be opened in SQL Server Data Tools. You can find installation instructions for SQL Server Data Tools in Chapter 4.
3. Each chapter relies on the Adventure Works DW 2012 database. In most cases, you use the relational database, but Chapter 16 requires the multidimensional database and Chapters 30 and 31 require the tabular database. You can find instructions for obtaining and installing these databases in Chapter 4.
4. The code samples in Chapter 21 and Chapter 22 require you to use the Spatial-Data database. You can restore it from the BAK file in the Chapter 21 directory. Similarly, the code samples in Chapter 32 and Chapter 35 require the Custom-Reports database, which you can restore from the BAK file in the Chapter 32 or Chapter 35 directory. Refer to *http://msdn.microsoft.com/en-us/library/ms177429.aspx* to learn how to restore a database backup.

Acknowledgments

We'd like to thank the following people for their help, advice, and the insights they have provided over the years that we have worked with Reporting Services: Robert Bruckner, Carolyn Chau, Thierry D'Hers, Dan English, Teo Lachev, Peter Myers, Grant Paisley, Lukasz Pawlowski, Carl Rabeler, Simon Sabin, and Paul Turley.

We'd also like to thank Aaron Nelson for his help with PowerShell and Jeff Rush for technical editing. In addition, there is a whole team of people to thank for helping this book transition from idea to print, including Ken Jones, Michael Bolinger, Holly Bauer, Richard Carey, and the production team at O'Reilly Media. Thanks, everyone!

Errata & book support

We've made every effort to ensure the accuracy of this book and its companion content. Any errors that have been reported since this book was published are listed on our Microsoft Press site:

http://aka.ms/SQL2012RS/errata

If you find an error that is not already listed, you can report it to us through the same page.

If you need additional support, email Microsoft Press Book Support at *mspinput@microsoft.com*.

Please note that product support for Microsoft software is not offered through the addresses above.

We want to hear from you

At Microsoft Press, your satisfaction is our top priority and your feedback our most valuable asset. Please tell us what you think of this book at:

http://www.microsoft.com/learning/booksurvey

The survey is short, and we read every one of your comments and ideas. Thanks in advance for your input!

Stay in touch

Let's keep the conversation going! We're on Twitter: *http://twitter.com/MicrosoftPress*.

PART I

Getting started with Reporting Services

The chapters in Part I provide you with a foundation for Microsoft SQL Server 2012 Reporting Services in preparation for learning about its features in detail throughout the remainder of this book. In Chapter 1, "What's new in Microsoft SQL Server 2012 Reporting Services," you learn how to use the new features in Reporting Services and how to upgrade reports if you already have an earlier version of Reporting Services in place. In Chapter 2, "Introducing the Reporting Services environment," you learn about the features and architecture of Reporting Services, which we expand upon in Chapter 3, "Understanding deployment scenarios," by describing the different deployment topologies so that you can decide which is best for you. In Chapter 4, "Installing Reporting Services," you learn how to plan for and complete an installation. Chapter 5, "Exploring Reporting Services fundamentals," provides an overview of report development with Report Builder and introduces the deployment process and report access with Report Manager.

CHAPTER 1

What's new in Microsoft SQL Server 2012 Reporting Services

Of all the business intelligence components in Microsoft SQL Server, Reporting Services has consistently expanded its feature base in each release since it first appeared as an update to SQL Server 2000. If you have used earlier versions of Reporting Services, you can use this chapter to familiarize yourself with these major new features and other miscellaneous changes. We also let you know about features that have been deprecated in this release and how to upgrade Reporting Services from prior versions.

Introducing new features

The most significant new feature in the latest version of Reporting Services is the introduction of Power View, an ad hoc reporting tool that combines data visualization and self-service reporting in a SharePoint integrated-mode deployment. Another change in this release is the new architecture of Reporting Services as a SharePoint service application, which improves the administration and scalability of your reporting environment in this mode. You can also take advantage of the new data alerts feature in SharePoint integrated mode when you want to receive an email notification when a report contains data that matches rules you have defined.

Power View

As an ad hoc tool, the target audience for Power View is a business user rather than a professional report developer. It's simpler to use than Report Builder or Report Designer in many ways and yet provides some more advanced data visualization capabilities. To use Power View, you must have access to a server running SharePoint Server 2010 Enterprise Edition and a supported data source. In this section, we provide an introduction to Power View to highlight its main features. You can learn much more about it in Chapter 31, "Using Power View."

Data sources

To get started, you select a data source and then launch Power View to explore its data. You can choose from one of the following data source types:

- **PowerPivot workbook** Locate a PowerPivot workbook in the PowerPivot Gallery or in a SharePoint document library, and then click the Create Power View Report icon in the upper-right corner of the PowerPivot workbook tile.
- **Shared data source** A Reporting Services Shared Data Source (RSDS) file can have its *Data Source Type* property set to Microsoft BI Semantic Model For Power View. For this type of data source, the connection string references a PowerPivot workbook (such as *http://SharePointServer/PowerPivot Gallery/myWorkbook.xlsx*) or an Analysis Services tabular model (such as *Data Source=MyAnalysisServer; Initial Catalog=MyTabularModel*). To use this type of shared data source for a Power View report, point to the shared data source name in the document library, click the down arrow, and then click Create Power View Report.
- **Business Intelligence Semantic Model (BISM) connection file** Like a shared data source, a BISM file can connect either to a PowerPivot workbook or to an Analysis Services tabular model. However, you can use a BISM file not only as a data source for Power View reports but also for Microsoft Excel workbooks. To create a new Power View report using a BISM file as a data source, point to the file name in a SharePoint document library, click the down arrow, and then click Create Power View Report.

Data visualization

Power View is unique because it allows you to interact directly with the data, starting with a table layout to select and group fields, switching to alternate data visualizations such as a column or line chart, combining multiple data visualizations on a page, as shown in Figure 1-1, and highlighting values or applying filters to one or data visualizations on the page. You can even create multiple pages, known as views, in the same report.

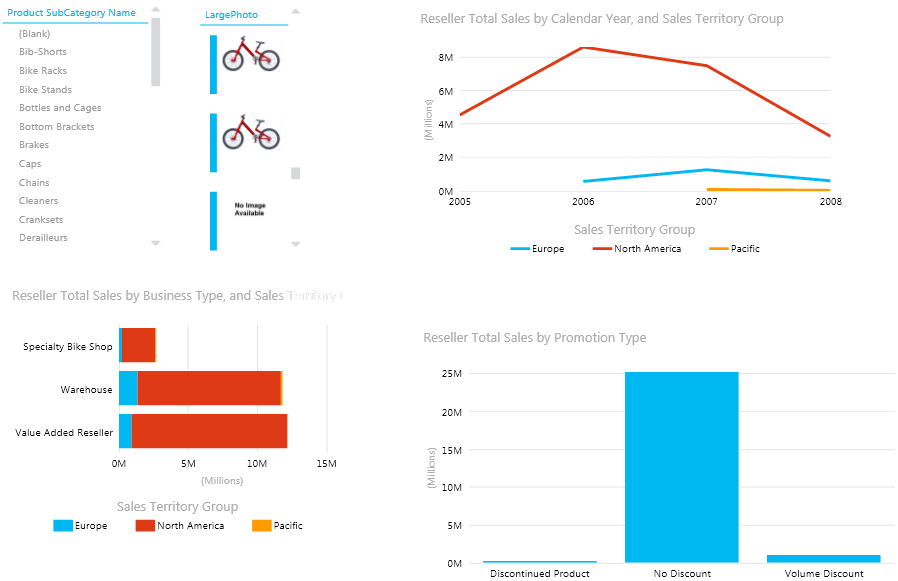


FIGURE 1-1 You can create a variety of data visualizations in Power View.

You start with a blank view workspace, as shown in Figure 1-2. As you develop the report, you add tables and data visualizations to the view workspace. The view size is a fixed height and width, just like the slide size in PowerPoint. If you need more space to display data, you can add more views to the report and then navigate between views by using the Views pane.

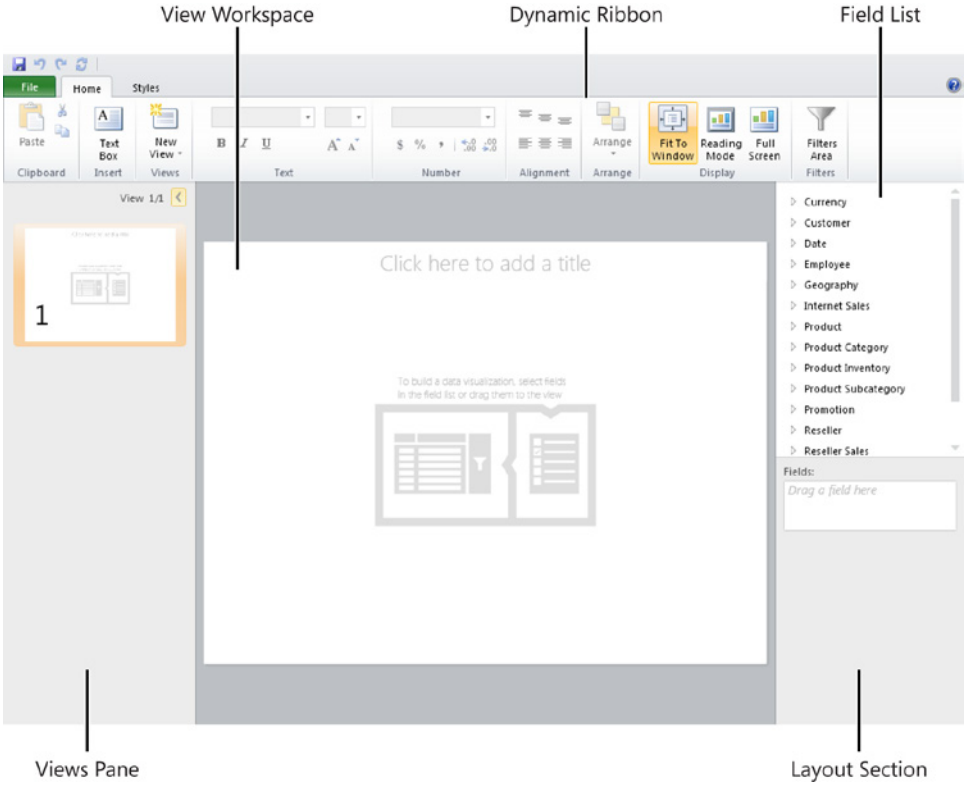


FIGURE 1-2 You use the Power View design environment to explore the data source and develop data visualizations.

To begin your exploration of the data, you select fields from the field list on the right. Fields are organized by table name. You can double-click a table to create a tabular arrangement of default fields defined in the data source, or you can expand the table to select individual fields. As you add fields to the view workspace or select an item in the view workspace, the dynamic ribbon at the top of the screen changes to show the buttons and menus applicable to your current selection. For example, you can select the table of fields in the view and use the Column button on the Chart Tools Design tab of the ribbon to change the visualization type.

Power View displays only as much data as it can fit in the space allotted to the current visualization. If more data is available, scroll bars display to allow you to access the remaining data incrementally. By limiting the amount of data you can see at one time, Power View can optimize the query performance of very large data sources.

As you add fields to a data visualization, your selections display in the layout section in the bottom right of your screen. The structure of the layout section depends on the current visualization. For example, the layout section for a column, bar, or line chart, shown in Figure 1-3, allows you to define which fields to display as values, the horizontal axis category, and the series grouping. You can rearrange the sequence of fields in the layout section by using drag-and-drop, or you can click the arrow next to the field name to change the behavior of a field, such as changing which aggregation function to use or whether to display rows with no data.

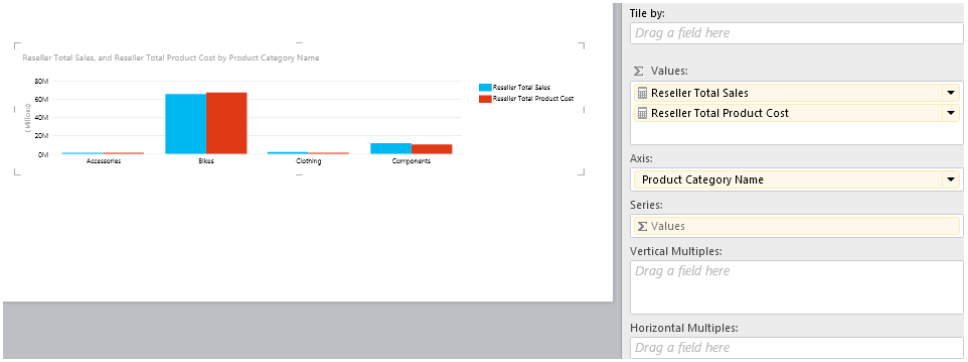


FIGURE 1-3 You use the layout section to arrange fields in a data visualization.

You can move or resize a data visualization at any time. You can move it by pointing to its border and then dragging it to another location in the same view when the cursor changes to a hand. Another option is to use the data visualization as the starting point for a new data visualization. When you see the cursor change to a hand, click the Copy button on the Home tab of the ribbon, and then click the Paste button to add the copy to the same view. You can also click the New View button on the ribbon if you want to add another page to your report. Either way, you can then move the copy to a new position. Also, when you point to the border and see the cursor change to a double-headed arrow, you can drag the border to make the data visualization larger or smaller.

Of course, Power View contains standard chart types, such as column, bar, and line charts, but it does not support other chart types that you can create in Report Builder or Report Designer. However, some unique data visualizations are available, such as the cards visualization shown in Figure 1-4, which is a scrollable list of grouped fields. You can use the cards visualization simply to display information about an entity, or you can use it as a type of filter for other data visualizations on the same view.

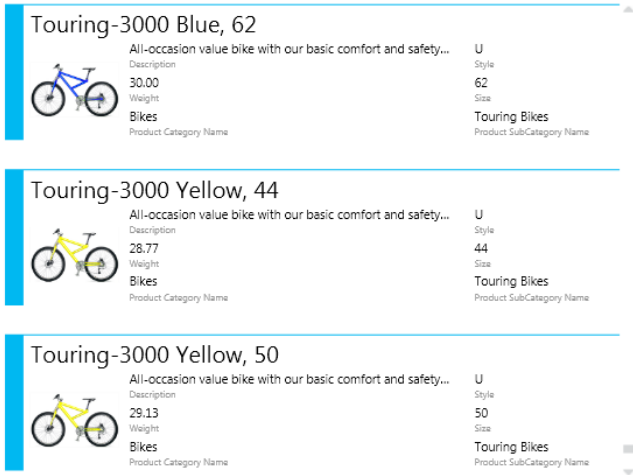


FIGURE 1-4 You can arrange a set of fields as a cards visualization.

Interactive scatter chart

Another feature of Power View is the ability to create an interactive scatter chart. This type of chart requires two numeric values. You place one value on the X axis and another value on the Y axis and then add a Details field that Power View uses to calculate corresponding X and Y coordinates. You can also add a third value to display points as bubbles for which the size increases as that value increases. A scatter chart can also include a field in the Color section to use for grouping. The interactive feature of the scatter chart applies when you add a field to the Play Axis, as shown in Figure 1-5.

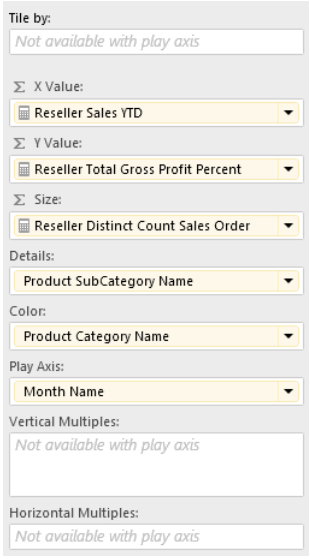


FIGURE 1-5 An interactive scatter chart includes X Value, Y Value, Details, and Play Axis fields, and optionally includes Size and Color fields.

The addition of a Play Axis field displays a play button below the scatter chart in the view. When you click the play button, the scatter chart cycles through each distinct value specified as the Play Axis field (which is typically a time period). You can also use the slider to select a specific value on the Play Axis. Whether you use the play button or the slider, you can see a watermark in the background of the chart for the current item in the Play Axis. You can filter the scatter chart to focus on a single Details value and see the path that it travels for each Play Axis value, as shown in Figure 1-6. Like other data visualizations, you can point to an area of the scatter chart to display a tooltip for more information.

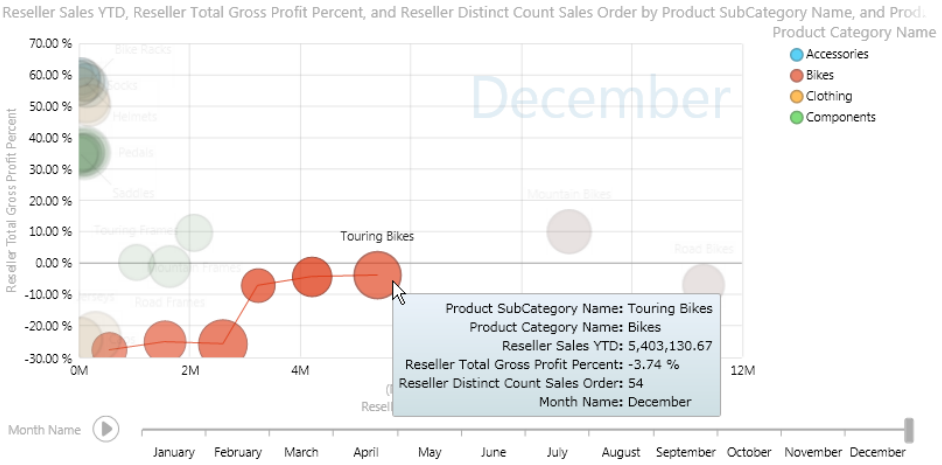


FIGURE 1-6 You can click a point or bubble in a scatter chart to see its path up to the selected value in the play axis.

Multiples

You can compare trends in data by creating multiple copies of the same chart. To do this, add a field to the Vertical Multiples or Horizontal Multiples area of the layout section, as shown in Figure 1-7. You can fine-tune the arrangement of tiles by using the Grid button on the Layout tab of the ribbon.

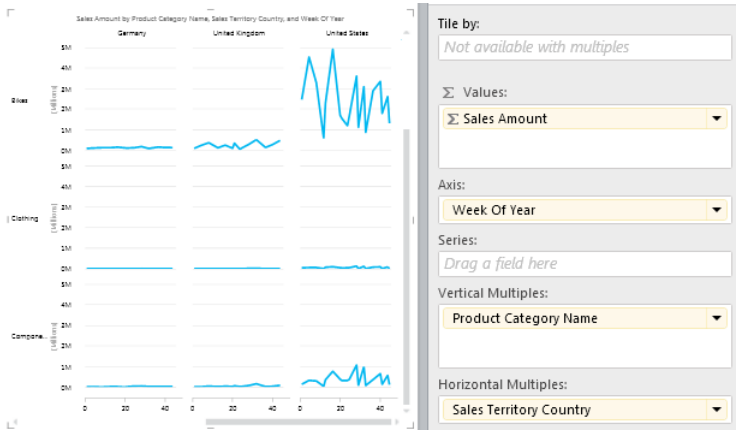
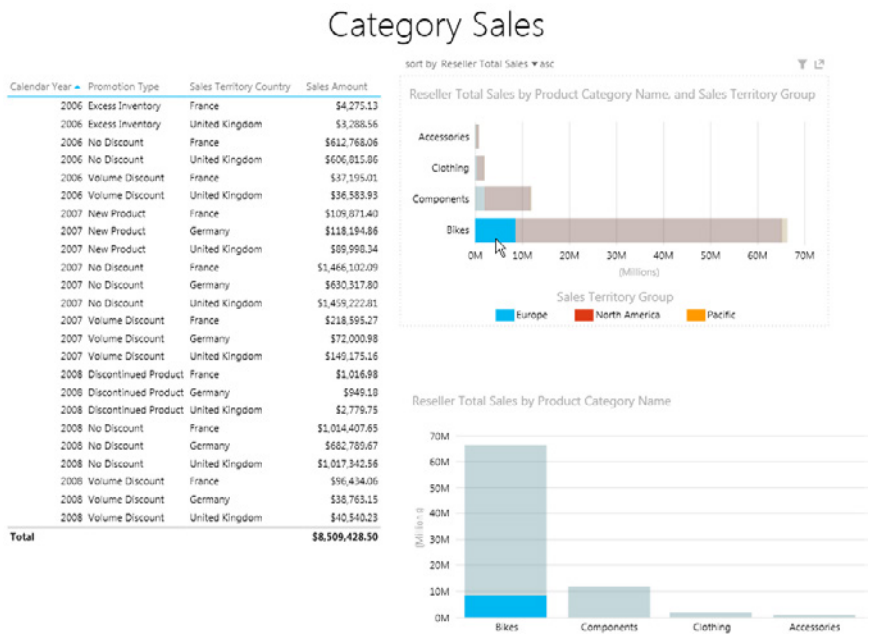


FIGURE 1-7 You can break out a chart by groupings called multiples.

Highlighted values

When you have multiple visualizations in the same view, you can select a value, such as a column or a legend item, to see relationships across all visualizations. Power View highlights related values in all visualizations and dims the other values to help you see the proportional relationships. For example, in Figure 1-8, the selection of Europe in the bar chart's legend highlights the proportion of sales in Europe compared to all other sales in each chart. Furthermore, the table displays only sales data for Europe. To clear the highlighting, click elsewhere in the chart without clicking another value.



Calendar Year	Promotion Type	Sales Territory Country	Sales Amount
2006	Excess Inventory	France	$4,275.13
2006	Excess Inventory	United Kingdom	$3,288.56
2006	No Discount	France	$612,768.06
2006	No Discount	United Kingdom	$606,815.86
2006	Volume Discount	France	$37,195.01
2006	Volume Discount	United Kingdom	$36,583.93
2007	New Product	France	$109,871.40
2007	New Product	Germany	$118,194.86
2007	New Product	United Kingdom	$89,998.34
2007	No Discount	France	$1,466,102.09
2007	No Discount	Germany	$630,317.80
2007	No Discount	United Kingdom	$1,459,222.81
2007	Volume Discount	France	$218,595.27
2007	Volume Discount	Germany	$72,000.98
2007	Volume Discount	United Kingdom	$149,175.16
2008	Discontinued Product	France	$1,016.98
2008	Discontinued Product	Germany	$949.18
2008	Discontinued Product	United Kingdom	$2,779.75
2008	No Discount	France	$1,014,407.65
2008	No Discount	Germany	$682,780.67
2008	No Discount	United Kingdom	$1,017,342.56
2008	Volume Discount	France	$96,434.06
2008	Volume Discount	Germany	$38,763.15
2008	Volume Discount	United Kingdom	$40,540.23
Total			$8,509,428.50

FIGURE 1-8 You can click a value to see highlighted values that are in all charts in a view and filter other visualizations, such as a table or matrix.

Slicer

A slicer is a single-column table that you add to the view for use as a filter. The table can contain labels or images. When you select one or more items in the slicer, Power View filters everything else in the same view, including other slicers, as shown in Figure 1-9. You can restore the unfiltered view by clicking the Clear Filter icon in the upper-right corner of the slicer.

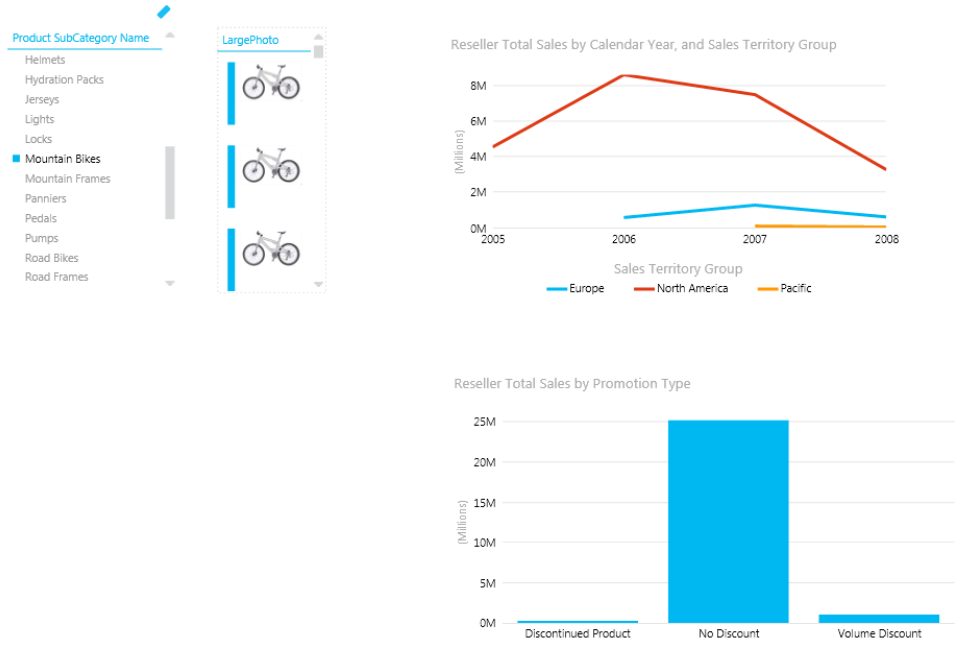


FIGURE 1-9 You can use a slicer to filter all content in a view.

Tiles

Rather than filter everything in a view, you can use a tile container to filter only visualizations inside the container, as shown in Figure 1-10. You can convert an existing table or matrix to a tile container by clicking the Tiles button on the Design tab of the ribbon. You then transform the table or matrix to another visualization type, if you like, and define the Tiles By field in the layout section to create a tile strip. To filter the tile container, you select one of the items in the tile strip. You can resize the tile container and add more visualizations to it if you want the filter to have broader scope without filtering the entire view.

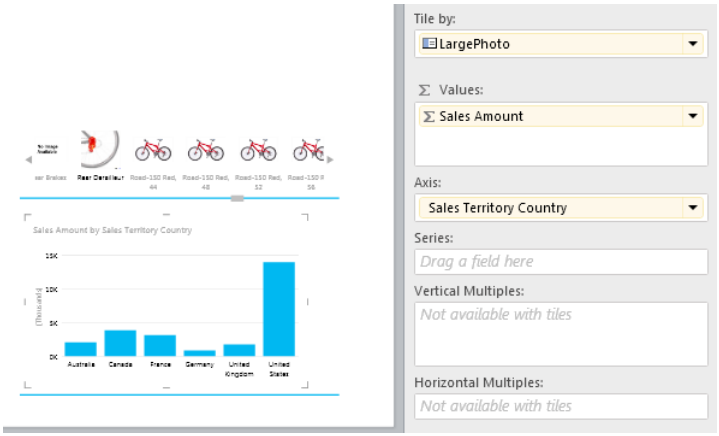


FIGURE 1-10 You can use a tile container to filter only the visualization it contains.

View filter

To use yet another type of filtering, you can set up filter criteria for the current view. Click the Filters Area button on the Home tab of the ribbon to toggle the visibility of the Filters area, which is not visible initially in a new report. Then drag a field to the Filters area and configure the criteria for basic filters, as shown in Figure 1-11.

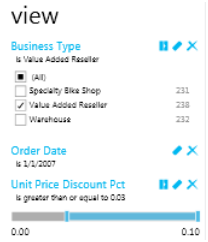


FIGURE 1-11 You can add basic filters to a view by selecting specific items, dates, or numeric value ranges.

The first icon in the toolbar to the right of the field name in the Filters area opens the advanced filter mode for that field. When a field has a string data type, you can create filter criteria based on partial words by using operators such as Contains, Starts With, and so on. A filter for a numeric field allows you to use a comparison operator like Greater Than Or Equal To, while a filter for a date data type allows you to use a calendar control in combination with a comparison operator like Is On Or After. You can also create compound filters consisting of two conditions by using a logical AND or logical OR operator, as shown in Figure 1-12.

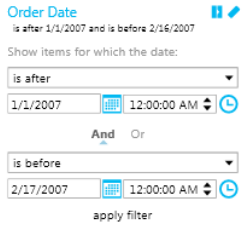


FIGURE 1-12 You can create a compound filter by using the logical AND or logical OR operator.

Visualization filter

To use the basic or advanced filter modes for a single visualization, click the Filter icon in the top-right corner of the visualization. The Filters area then displays a tab for the selected visualization where you can add fields and define filter criteria, as shown in Figure 1-13.

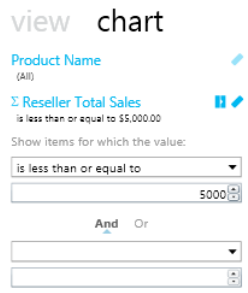


FIGURE 1-13 You can create basic or advanced filters to apply only to a selected visualization.

Report sharing

When you want to save your report to review later or to share with others, you can save it by using the File menu, as long as you have the Add Items permission on the target document library. If you save it to a PowerPivot Gallery, you can use the Save Preview Images With Report option to include thumbnail images for display in the gallery views. Otherwise, save the file without images. Either way, your report saves as an RDLX file, which you can open only from SharePoint in Power View. This file type does not work with Report Builder or Report Designer.

You can also print the current view when you use Fit To Window or Reading mode only. You can switch to these viewing modes by using the applicable button on the Home tab of the ribbon. Then open the File menu and then click Print to print the current view in landscape orientation.

You can also share your report with other users by saving the report as an RDLX file to a SharePoint document library. If you prefer, you can export the report as a Microsoft PowerPoint file. In the latter case, other users can interact with the data within PowerPoint as long as they have an active connection to SharePoint and have the necessary permissions to view the tabular model. When a user displays a slide in Reading View or Slideshow modes, a Click To Interact link displays in the lower-right

corner of the slide. Clicking this link loads the view from Power View and allows the user to change filter values in the Filters area, in slicers, in tile containers, and to highlight values.

SharePoint integration

The integration of Reporting Services into a SharePoint technology was first possible in SQL Server 2005. Since that introduction, incremental improvements to Reporting Services integration with SharePoint have been made in subsequent releases of SQL Server. However, the architecture received a complete overhaul in SQL Server 2012. Reporting Services is now a shared service application, which simplifies configuration. Chapter 4, "Installing Reporting Services," provides detailed instructions on installation and configuration.

The shared service application architecture means that you can now scale Reporting Services across web applications and SharePoint Server 2010 farms more easily. This architecture allows Reporting Services to take advantage of built-in network load balancing and use fewer resources. Consequently, Reporting Services delivers better performance overall as compared to previous versions. You can learn more about scalability in Chapter 3, "Understanding deployment scenarios."

Furthermore, deeper integration with SharePoint technology also simplifies administration. If you're using claims-based authentication in your SharePoint farm, you can extend it for use with Reporting Services. Another integration point is the backup and recovery process built into SharePoint backup that you can now use for Reporting Services content. We describe backup and recovery in Chapter 25, "Performing administrative tasks."

Note For more information about claims-based authentication, see *http://msdn.microsoft.com/en-us/library/hh231678.aspx*.

Data alerts

Ever since the first release of Reporting Services, you have had the option to configure standard and data-driven subscriptions. A data-driven subscription is one in which you use a query to provide values for a subscription definition, such as recipient names or parameters to set. However, nothing in Reporting Services has been available to monitor the report content. That is, the data in the reports could not be evaluated against a set of predefined criteria. In SQL Server 2012, you can create a data alert to send an email when conditions in the predefined criteria are satisfied, but only if you are running Reporting Services in integrated mode on a SharePoint Server 2010 Enterprise Edition server. In this section, you learn the basic operations of data alerts; you can learn more by referring to Chapter 28, "Data alerting."

Data Alert Designer

If the report returns data when you open it, contains at least one data region, and all data sources use stored credentials or no credentials, you can create a new data alert. In fact, you can create one or more data alerts for any report that you can access, as long as the prerequisites are met. To do this,

open a report in a SharePoint document library and select New Data Alert from the report's Action menu. You use the Data Alert Designer, shown in Figure 1-14, to define rules for each data region in the report, assign a schedule for checking the rules, configure email settings for recipients of the alerts, and provide a static subject and description.

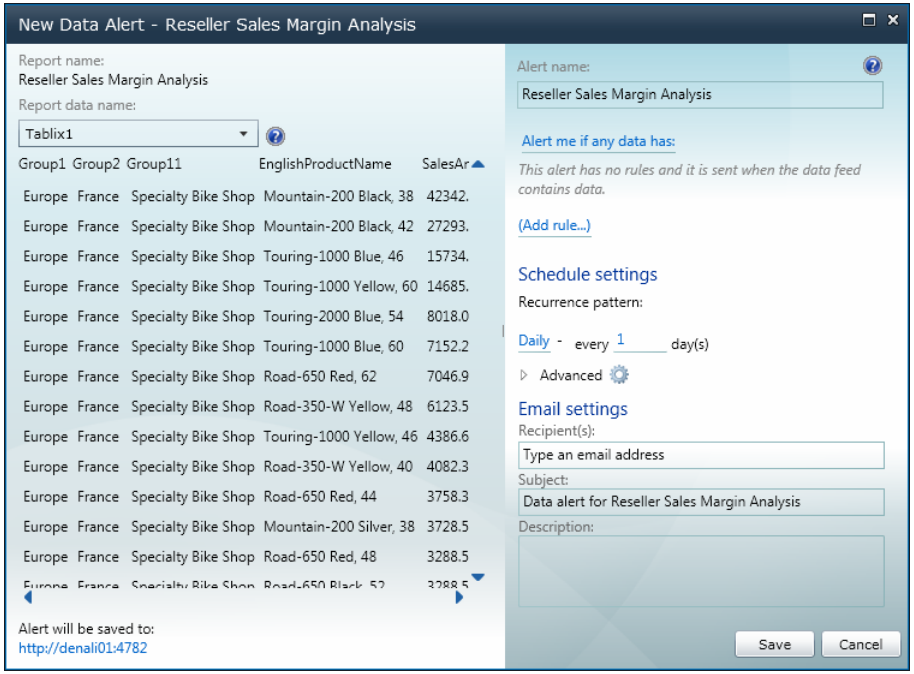


FIGURE 1-14 You can define criteria and a schedule for each data region in a report.

If the report runs successfully at the scheduled time and the report data satisfies the rules you specify, the alerting service sends an email message to each recipient. The email includes the description that you define when creating the alert and lists the raw data values that satisfied the alert criteria, similar to the one shown in Figure 1-15. If the report uses report parameters, the report parameter values also display in the rendered report. However, if the report execution fails, each recipient specified in the Data Alert Designer receives an email describing the error.

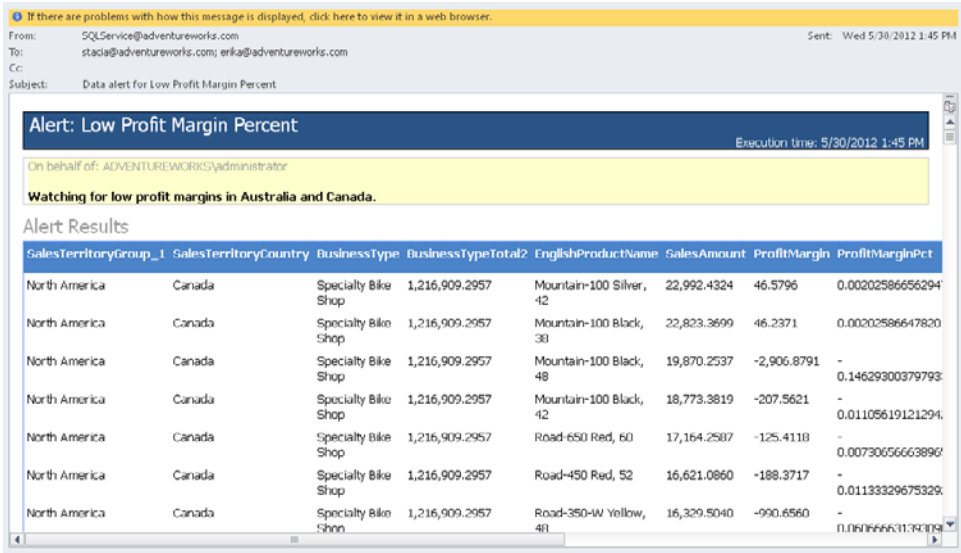


FIGURE 1-15 An email notification is sent to you with raw data values satisfying the alert rules and the description that you specify for the data alert.

Data Alert Manager

The Data Alert Manager keeps track of all your data alerts. To open the Data Alert Manager, open the document library containing your report, click the down arrow to the right of the report name, and then select Manage Data Alerts. The Data Alert Manager shows you the number of alerts sent by data alert, the last time it was run, the last time it was modified, and the status of its last execution. A portion of the Data Alert Manager page is shown in Figure 1-16.

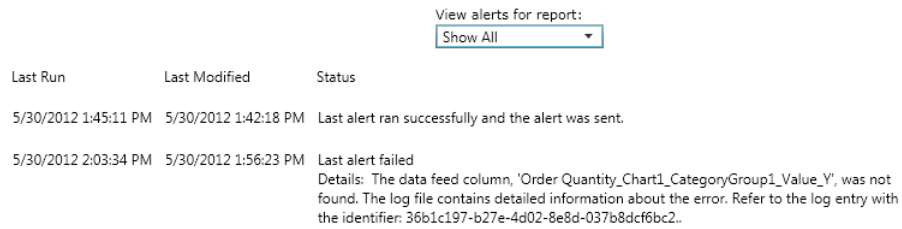


FIGURE 1-16 You use the Manage Data Alerts page to keep track of alert execution and to change, delete, or execute a data alert.

You can right-click a data alert here to edit, delete, or run it on demand. A site administrator can view and delete your data alerts, but no one other than you can edit and run your data alerts on demand.

Reviewing other changes

Although the majority of changes to Reporting Services in SQL Server 2012 are new to the product, two others don't change your usage of Reporting Services as significantly but are important to know about. First, the design environment moves to a Microsoft Visual Studio 2010 Shell called SQL Server Data Tools (SSDT). Second, two new rendering extensions are available.

SQL Server Data Tools

Business Intelligence Development Studio (BIDS) has been the host of the design environment for Reporting Services since SQL Server 2005. The only place you see this name for the tool is in the program group for SQL Server on the Start menu. After you launch it, you never see the name again. That's because it's really just a shell for the version of Visual Studio for which it was made. The release cycles of SQL Server and Visual Studio have never been aligned, so a newer version of Visual Studio has always been available long before SQL Server business intelligence components begin using that version.

SQL Server 2012 is no exception. The difference this time is that the BIDS is gone from the Start menu, replaced by SSDT. Otherwise, the experience is the same. The same business intelligence project types, Report Server Project Wizard and Report Server Project, are available. You use the same user interface for developing reports as you did in BIDS, although the color scheme of Visual Studio is different. The advantage of having SSDT is the ability to add related projects to the solution containing your report project and use the Visual Studio 2010 integrated development environment for developing a class library or custom application.

New rendering extensions

This version of Reporting Services adds support for the most current versions of Microsoft Excel and Microsoft Word as output formats. Previously, you could export a report only as an XLS file in all versions of Reporting Services or as a .doc file in SQL Server 2008 and SQL Server 2008 R2. These file formats provide compatibility with Excel 2003 and Word 2003, respectively. Although users of Excel 2010 and Word 2010 can also open these file formats, there were some limitations that affected the size of the report that you could export to Excel and the size of the output file for both Excel and Word documents. These limitations are removed in SQL Server 2012.

Excel 2010 renderer

When you select Excel as the export option in SSDT, Report Manager, or the SharePoint Report Viewer, you now save the report as an XLSX file in Open Office XML format. That means you can open the file in either Excel 2007 or Excel 2010, as long as you have the client installed on your computer. It also

means you can export a larger report to Excel, which supports 1,048,576 rows and 16,384 columns. In addition, the XLSX format supports 16 million colors in the 24-bit color spectrum. Another benefit of the new renderer is the smaller file size for the exported report, made possible by better compression.

Tip You can download the Microsoft Office Compatibility Pack for Word, Excel, and PowerPoint at *http://office.microsoft.com/en-us/products/microsoft-office-compatibility-pack-for-word-excel-and-powerpoint-HA010168676.aspx* if you prefer to produce XLS files for Excel 2003. If you need to support both formats, you can enable the Excel 2003 renderer in the RSReportServer.config and RSReportDesigner.config files by following the instructions at *http://msdn.microsoft.com/en-us/library/dd255234(SQL.110).aspx#AvailabilityExcel*.

Word 2010 renderer

Exporting a report as a Word document in prior versions of Reporting Services did not have the same limitations as exporting an Excel workbook. However, the benefit of having a Word 2010 renderer is in maintaining consistency across the Office products as well as producing a smaller file than the Word 2003 renderer. The new renderer compresses the report to generate a .docx file in Open Office XML format.

Tip If you want to open the exported document in Word 2003, you can install the Microsoft Office Compatibility Pack for Word, Excel, and PowerPoint, available for download at *http://office.microsoft.com/en-us/products/microsoft-office-compatibility-pack-for-word-excel-and-powerpoint-HA010168676.aspx*. Another option is to enable the Word 2003 renderer in the RSReportServer.config and RSReportDesigner.config files by following the instructions at *http://msdn.microsoft.com/en-us/library/dd283105(SQL.110).aspx#AvailabilityWord*.

Parting with Report Builder 1.0

Report Builder 1.0 was known simply as Report Builder when it was first released in SQL Server 2005 as a simple ad hoc reporting tool. Then SQL Server 2008 released Report Builder 2.0, and the first Report Builder was known thereafter as Report Builder 1.0. Although both versions of Report Builder are intended for self-service reporting, the user interface and report development process in each version are completely different. Report Builder 1.0 requires you to create a semantic model describing the tables and fields available for reporting, whereas Report Builder 2.0 allows you to create data sources and data sets much like you do in SSDT. In fact, Report Builder 2.0 is the predecessor to Report Builder in SQL Server 2012. Likewise, the current version of Report Builder (described in Chapter 5, "Exploring Reporting Services fundamentals") is nothing like the original Report Builder in SQL Server 2005.

In SQL Server 2008 and SQL Server 2008 R2, Report Builder 1.0 coexists with Report Builder 2.0 and Report Builder 3.0, respectively. In those versions, BIDS also includes a Report Model Project that you use to develop the semantic model required to use Report Builder 1.0. This project type is no longer available in SQL Server 2012, nor is Report Builder 1.0. Nonetheless, if you have an existing report model, you can continue to use it as a data source in the SQL Server 2012 Report Builder or SSDT, as described in Chapter 8, "Retrieving data for a report."

Upgrading from prior versions

You can transition your reports from a prior version of Reporting Services to SQL Server 2012 in two ways. One way is to perform an in-place upgrade to update your existing environment and keep everything in its current location. Another way is to migrate by creating a new environment and copying your reports to that new environment. You can use SQL Server Setup or a command-line utility to upgrade your existing report server database to the schema required for SQL Server 2012. When you run the setup wizard on a computer containing an earlier version of SQL Server, you can choose to upgrade or to install a new instance of SQL Server for a side-by-side installation. Regardless of which option you choose, the System Configuration Checker runs during setup to ensure that your computer meets the requirements for a successful installation.

Important This section provides an overview of the process to upgrade or to migrate a standard Reporting Services environment. If you have customized your environment by adding custom applications, custom assemblies, or modifying configuration files, you should refer to *http://msdn.microsoft.com/en-us/library/ms143747.aspx* for more detailed information.

Preparation

Although SQL Server Setup proceeds with an upgrade only when your computer meets the requirements, you should also prepare for the upgrade by running the Upgrade Advisor to analyze your environment to uncover any potential problems in advance. Even if you are performing a migration, you should run the Upgrade Advisor to identify potential issues and to show custom settings of which you might not be aware if you did not configure the existing report server.

Tip You can download the Upgrade Advisor from *http://www.microsoft.com/en-us/download/details.aspx?id=29065&ocid=aff-n-in-loc--pd* or install it from the Servers\redist\Upgrade Advisor folder of your SQL Server 2012 installation media.

It's also a good idea to back up important files and databases before you perform an upgrade or migration. That way, if the process fails before completing, you can restore everything and resume

normal operations in the prior version until you can resolve the problem. You should back up the following items:

- **Symmetric key** Use Reporting Services Configuration Manager or the Rskeymgmt command-line utility to back up the symmetric key. The procedure to do this in earlier versions is the same as we describe for SQL Server 2012 in Chapter 25. In SQL Server 2012, you use Report Configuration Manager only for a native-mode report server, but you can use it for both types of report servers in all previous versions.
- **Report server database** Back up the ReportServer database (and optionally the ReportServerTempDB database) by using the same process you use to back up other databases in SQL Server. It's possible that the report server database has a different name, so be sure to correctly identify the database. This database exists for both a native-mode and SharePoint integrated-mode report server.
- **Rsreportserver.config** Save a copy of this file, which contains important settings for your report server. You can find it in the folder for the Reporting Services instance, which is in the Program Files\Microsoft SQL Server folder. The instance might be in a folder like MSSQL.3 or MSRS10.MSSQLServer. Within that folder, you can navigate through the folder hierarchy to locate the ReportServer folder, which contains this configuration file.
- **Rssvrpolicy.config** Save a copy of this file, which contains the security policies for the report server. You can find it in the ReportServer folder.
- **Reportservicesservice.exe.config** If you are running a native-mode report, this file is located in the bin folder that you find in the ReportServer folder.
- **Rswebapplication.config** Save a copy of this file, which contains settings for Report Manager and is used only for a native-mode report server. It's found in the ReportManager folder, which is a sibling to the ReportServer folder.
- **Rsmgrpolicy.config** If you are running a native-mode report server, save a copy of this file, which contains security policies for Report Manager. You can find it in the ReportManager folder.
- **Web.config** Locate and save each Web.config file for your report server. Both types of report servers have a Web.config file in the ReportServer folder, and a native-mode report server has one in the ReportManager folder.
- **Machine.config** If you have modified the Machine.config file for use with Reporting Services (which is not a common occurrence), you should save a copy of this file as well. You can locate this file in the applicable Windows folder for your operating system in the Microsoft .NET\Framework\v.1.1.4322\CONFIG folder for SQL Server 2005 and in the Microsoft.NET\Framework\v2.0.50727\CONFIG folder for SQL Server 2008 and SQL Server 2008 R2.
- **Virtual directories** If you are upgrading from SQL Server 2005 and have added files to the virtual directories used by the report server, back up those files. Reporting Services no longer uses Internet Information Services (IIS).

Important If you have invalid or expired SSL certificates on the report server, you must remove them first. Otherwise, the upgrade will fail.

In-place upgrade

If your computer passes the System Configuration Checker in SQL Server 2012 Setup, you can proceed with an in-place upgrade of a native-mode or SharePoint integrated-mode report server. Depending on the version of Reporting Services that you are upgrading, you might need to manually complete some steps before or after the upgrade process.

Note If you prefer to use the Setup.exe command-line utility to perform the upgrade, refer to *http://msdn.microsoft.com/en-us/library/ms144259.aspx* for instructions.

Native-mode report server upgrade

You can perform an in-place upgrade for the following versions of Reporting Services running in native mode:

- SQL Server 2005 Reporting Services Service Pack 4
- SQL Server 2008 Reporting Services Service Pack 2
- SQL Server 2008 R2 Reporting Services Service Pack 1

SQL Server Setup creates folders for a new Reporting Services instance in the Program Files\Microsoft SQL Server folder. The root folder for this collection of folders is named MSRS11.MSSQLServer for a default instance or MSRS11.<*instance*> for a named instance. SQL Server Setup also adds all the files necessary to support the report server, configuration tools, and utilities and upgrades the schema of the report server databases.

Setup does not upgrade existing installations of client tools like BIDS or SQL Server Management Studio. Instead, it adds new versions as a side-by-side installation. You can delete these tools following the upgrade if you have no further need for these tools for working with other Reporting Services instances.

Important If the report server database is on a separate computer from the report server, your Windows login must have sysadmin or database update permissions. Otherwise, SQL Server Setup prompts you for credentials with the required permissions.

Each version of Reporting Services relies on a set of configuration files. Depending on the version you are upgrading, Setup creates new configuration files by merging settings from the existing files and leaves the existing files in place. You can manually remove the folders for the previous Reporting Services instance after you confirm that the upgrade is successful. Reporting Services will use only files in the folder with the MSRS11 prefix.

If you are upgrading from SQL Server 2005, Setup uses the virtual directory settings for Reporting Services to reserve corresponding URLs. It might be necessary to manually remove the virtual directories when the upgrade is complete. Also, if your version of SQL Server 2005 is prior to Service Pack 1, Setup adds Database Owner permissions to the RSExecRole.

SharePoint integrated-mode report server upgrade

You can perform an in-place upgrade for the following versions of Reporting Services as long as you have already upgraded the SharePoint farm to SharePoint Server 2010:

- SQL Server 2005 Service Pack 2
- SQL Server 2008 Reporting Services Service Pack 2
- SQL Server 2008 R2

Regardless of the version you are upgrading from, you start the upgrade process on your SharePoint farm by installing the Reporting Services add-in on all web front-end servers. Then start the SQL Server 2012 upgrade process by running Setup on the report server. During the upgrade process, you might see a prompt asking you to provide credentials for SQL Server Reporting Services SharePoint Mode Authentication. The upgrade process uses these credentials to create a new SharePoint application pool.

Important The SharePoint Server 2010 prerequisite installer installs the SQL Server 2008 R2 add-in for Reporting Services. You must download and install the SQL Server 2012 version of this add-in from *http://www.microsoft.com/en-us/download/details.aspx?id=29068*.

Reports upgrade

Reports are stored in the ReportServer database. Setup upgrades only the schema of this database, without modifying the reports it contains. When a user or a background process executes a report, the report server upgrades the report and replaces the compiled version of the report in the ReportServer database. Any RDL files found on the same computer are not upgraded. You must open these files in SSDT to upgrade them to the current RDL schema.

Migration

If you are unable to perform an upgrade in place, you can perform a migration instead. You might also opt to perform a migration to minimize downtime. Your report server is unavailable to users during an upgrade but remains available during a migration.

Native-mode report server migration

You can perform a migration of a native-mode report server running SQL Server 2005 or later by installing SQL Server 2012, as described in Chapter 4, on the same computer as a side-by-side installation. If you prefer, you can install SQL Server 2012 on a separate computer. Either way, choose the Install Only option on the Reporting Services Configuration page of Setup so that you can configure Reporting Services to use your existing report server database after installation completes.

Important If you want to use a new Database Engine instance to host your report server databases, move the databases before configuring the new report server. To preserve all security settings, it's important that you move the databases by performing detach and attach operations rather than copying these databases. If the Database Engine instance has never hosted report server databases, you must create the RSExecRole in the *master* and *msdb* databases as described at *http://msdn.microsoft.com/en-us/library/cc281308*. Then use the Reporting Services Configuration Manager to restore the encryption key that you saved as part of the preparation process.

For a scale-out deployment, you must take each report server node offline and perform a migration on each one individually. You must also manually delete records from the Keys table in the ReportServer database before you configure Reporting Services.

Use Reporting Services Configuration Manager to configure the Report Manager and web service URLs and connect to the report server databases. When you connect the first report server to the report server database, an upgrade is performed on the database to make it compatible with SQL Server 2012 Reporting Services. You also use the Reporting Services Configuration Manager to restore the symmetric key that you saved during the preparation process. Last, you can add the other report servers if you have a scale-out deployment.

Make sure port 80 is open if you have a firewall running on the report server. You can use the Reporting Services Configuration Manager to use a different port if you prefer, but you must open that port manually in your firewall to enable remote access to the report server.

Open Report Manager and run several reports to ensure that the new installation works correctly. If so, you can uninstall the previous version of Reporting Services. You might also need to manually remove the following items:

- RSExecRole (in *master* and *msdb* databases of original Database Engine), if you moved the report server databases to a new Database Engine instance

- Service account running Reporting Services
- Reporting Services log files

If you migrated from SQL Server 2005, you should remove the following items:

- Application pool for the Report Server web service
- Virtual directories for Report Manager and web service
- IIS server, if not used for other purposes

SharePoint integrated-mode report server migration

You must perform the report server migration within the same SharePoint farm due to the use of unique identifiers within the farm. An exception to this requirement is a migration of all SharePoint content to a new farm. If you plan to host SharePoint configuration and content databases, including those used for Reporting Services in a SQL Server 2012 Database Engine instance, you must upgrade the farm to SharePoint 2010 and apply Service Pack 1.

In addition to performing a backup of the items listed in the "Preparation" section of this chapter, you should back up the SharePoint content databases and detach them. Also, back up the following files in Program Files\Common Files\Microsoft Shared\Web Server Extensions\14\Web Services\Reporting:

- Rsreportserver.config
- Rssvrpolicy.config
- Web.config

Next, to perform a complete migration of a farm, create the new SharePoint 2010 farm and install Reporting Services in SharePoint mode, as described in Chapter 4, but don't create the Reporting Services service application yet. Restore the SharePoint content database on the new server, and attach it to a web application. You can also restore the report server databases at this time.

Important You can attach a SharePoint content database to a web application by using Windows PowerShell or the stsadm utility. SharePoint Central Administration does not include a page to perform this task. For more information, see *http://technet.microsoft.com/en-us/library/cc263299.aspx#AddDB*.

Copy the Rsreportserver.config file to the new server. You need to place it in the folder that has the same name as the one from which you originally backed it up, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14\Web Services\Reporting.

Next, create the Reporting Services service application by following the instructions in Chapter 4, with one exception. On the page you use to create the new service application, change the Database Server and the Database Name to the correct values for the restored report server database.

The last step is to restore the Reporting Services encryption key. To do this, click the Manage Service Applications link in SharePoint Central Administration and click the Reporting Services service application link, the Key Management link, and the Restore Encryption Key link. Click the Browse button, navigate to the location where you stored the symmetric key that you saved in the "Preparation" section, and then provide the password.

CHAPTER 6

Planning your report design

Before you read this chapter, you should have Reporting Services installed and have a general understanding of how to create a basic report. If you are still fairly new to Reporting Services report development, you might not fully understand how to implement some of the concepts that we discuss in this chapter, but that should not deter you from skimming through it. The purpose of this chapter is to help you think about how to prepare for the report development process. Knowing who will view reports and how they want or need to interact with reports has an impact on how you should design those reports.

Reporting Services provides a lot of flexibility in report design, which can be overwhelming at first. You can spend a lot of time trying out different approaches to report design before you discover that a feature that users need most conflicts with the features you've been implementing. With some advance planning, you can decide which features are most important for your current project and be more productive throughout the report development process. The suggestions we make in this chapter are not exhaustive, but they describe situations we commonly encounter in our consulting practice and strategies we have found useful.

Most importantly, we recommend that you involve users as much as possible during the development process. Prototype early in the development cycle with a subset of data (or even fake data if necessary) and solicit feedback. Make sure you ask questions to clarify your understanding of the requirements, and working with a prototype can help enormously with this process. A prototype is something tangible around which you can generate a discussion. With a prototype, uses can confirm that the design is right, wrong, or maybe close but not quite what it needs to be. Furthermore, users are better able to answer your questions about whether a particular design feature would be useful if they can see it in action.

Knowing your audience

Knowing your audience is important. An understanding of how users typically interact with information, their technical skill and interest level, and whether they plan to use the data in other ways will affect how you approach the design of reports. More likely, you will have a mixed audience and will need to create different styles of reports to accommodate a diverse audience.

One of the first things to know is how users will be viewing reports. The access method they use to view a report can give you more freedom for creativity in the report design or can impose certain restrictions. As you gather information about report access, consider the following scenarios:

- **Will users go online to find a report and view it there?** If users are viewing reports online, you can consider any interactive feature that Reporting Services supports. However, many of these features are useless if users will be referring to printed reports. For interactivity features that you can add to online reports, see Chapter 13, "Adding interactivity." If you must plan for printed reports, see the "Printed reports" section later in this chapter.

  When designing a report for online consumption, users typically don't want to scroll through pages and pages of a report to find needed information. The addition of report parameters, in conjunction with query parameters, can enable the user to focus on information of interest. You can learn more about parameters in Chapter 14, "Creating dynamic reports with parameters and filters."

- **Will users primarily receive reports via an email subscription?** If so, you need to find out whether the email should contain an embedded copy of the report, include an attachment to the report, or simply provide a link to the report online. In the latter case, you can design your report by using the same criteria for an online report.

  If the report must be embedded in the email, the report should be relatively small in size, containing a minimal number of records and using a layout that fits easily within the width of the user's monitor if they usually use a computer to view emails. If the users rely primarily on mobile devices for email, consider sending reports as an attachment instead. A report sent as a PDF attachment must be designed with pagination in mind, which we explain in Chapter 15, "Managing the page layout." However, if users need access to the data to combine with other sources or to perform computations, you can send the report as an Excel attachment, and you can focus more on the data structure and less on conforming the report appearance to a corporate standard.

- **Will users be using mobile devices to access reports?** A recent development is the increasing use of tablets and smartphones to access corporate information. Rather than try to fit everything into a one-size-fits-all dashboard-style of report, consider focusing on a few key subject areas and then provide access to additional information through links that open separate reports or jump to another section of the same report.

  When users are in the office, they're often accessing the BI system to check on the general status of operations. They'll be looking at the big picture to see whether things are going as expected or not, and they can then explore areas of interest to get into more specific information as needed. By contrast, the mobile audience is usually responding to a circumstance, either searching for something specific to answer a question or looking up the high-level status of some business process. They've either encountered a problem that needs to get resolved or anticipate that they're on the verge of a problem that they want to pre-empt. Mobile users might be under pressure standing in front of a client or in an otherwise distracting environment, so the reports must be designed to make it as easy as possible to get to specific information and to follow a train of thought by drilling from one type of information to another type of information, and to filter the data down to relevant details as needed. Parameters and actions are useful features to implement in mobile reports. See Chapters 13 and 14 to learn how to use these features.

The technical skills and interest level of your users also play a role in determining how best to make information available. Technical skills among users can vary widely in an organization or even within a department. You should be prepared to provide building blocks for users who are capable of some level of self-service reporting if interest exists. The following are some strategies to consider for different types of users:

- **Data-savvy advanced users** Some users spend their entire day focused on data. This group of users likely has the technical skills necessary to acquire and manipulate data from relational or multidimensional data sources. They want as much unfettered access to data as you're willing to give them. You can set this group up to work with Report Builder and allow them to build reports from scratch, or you can create a library of starter reports that they can modify as needed.

- **Motivated power users** This group of users knows their data very well. Perhaps they are capable of building reports but are unable to write their own queries. You can give this group permission to use Report Builder and publish shared datasets for them to use when creating their own reports. That way, they have the freedom to create the reports they want and you can control the data that they use. See Chapter 8, "Retrieving data for a report," for more information about shared datasets.

  Within this group, you might even have people who know what they want when they see it but are unable to put together their own queries and are intimidated by the thought of building a report from scratch. For this group, you can publish report parts that represent commonly accessed data in a variety of data structures—tables, charts, maps, and so on. Then this group can use the Report Part Gallery in Report Builder to browse published report parts and piece together multiple report parts to build a report just the way they like it. Chapter 29, "Reusing report parts," explains how to do this.

- **Basic users** Some users lack the time, the skills, or the interest to explore data and build their own reports. They just want access to information as quickly and flexibly as possible. When creating reports for this group, you might find a lot of similarities between reports. To save yourself from creating a maintenance nightmare and also save users the trouble of trying to determine which of the many possible reports has the answers they need, try to consolidate multiple reports into a single report as much as you can. Use parameters to modify the content of the reports, either by filtering the data or by hiding or showing elements dynamically. You can create linked reports, as we describe in Chapter 23, "Deploying reports to a server," to create the illusion of multiple reports if necessary and hide the parameters that you don't want users to change.

Yet another aspect to understand about your audience is how they plan to use the report. The intended use can make a difference in the direction you should focus your development effort. The following are some common ways that people use reports and related development considerations:

- **Standard reporting** The most common reason to implement Reporting Services is to provide standard reporting at the department, division, or organizational level. This type of reporting typically requires the use of standard styling to produce a consistency of appearance across multiple reports.

  After the standards for your reporting environment are established, you can create one or more base reports to use as templates. For example, you might create a portrait version and a landscape version of a table layout and include common parameters in each version. You can store the report definition files in the Program Files (x86)\Microsoft Visual Studio 10.0\Common7\IDE\PrivateAssemblies\ProjectItems\ReportProject folder on your computer. When you use the Add A New Item command in SSDT to create a new report, you can select the template and continue report development by adding a dataset and assigning fields to a data region.

- **Raw data** Reporting Services can export data to Excel, CSV, XML, and ATOM formats. Some people just want the raw data so that they can develop their own Excel workbooks or import it into PowerPivot for Excel or other applications. Creating a report to deliver data might be preferable to allowing direct access to data sources because you might need to enhance the data with calculations, you might want to schedule data retrieval to manage the impact on source systems or your network, or you might want to deliver the data by using subscriptions rather than require the user to go to the report server. If the report is simply a means to an end, you don't need to spend a lot of time conforming a report to a standard format with page heads and footers, and so on. You should keep these reports in a separate folder to emphasize their purpose as data structures.

- **Quick information retrieval** Sometimes people need access to information without a lot of formality. This information might come in the form of an email to report the status of a process, or it might be a lookup to get customer contact information. Either way, this type of report is not typically something a user prints out to share with others or present at a meeting. Therefore, the layout can be simple, without the formatting that you would require for official corporate reports.

- **Operations monitoring** This type of reporting can take the form of a dashboard or detailed reports that include conditional formatting to highlight trends and exceptions. When designing a dashboard, you should consider the standard size screen for the majority of users so that you can use the screen effectively. A dashboard should convey summarized information clearly at a glance, using data visualization techniques to provide comparisons, display trends, or highlight exceptions. Ideally, the user can see the entire dashboard without scrolling. Use the chapters in Part III, "Adding data visualizations," to learn about the variety of data visualizations at your disposal.

  You should also consider the questions that might result from viewing a dashboard and prepare supporting detailed reports. You can add actions to each data visualization to open a related report and pass the context of the user's click to that report. For example, in a chart that displays sales trends by week, a user can click a specific week to open a report that displays the daily detail for that week. See Chapter 13 for information about actions.

- **Management or external stakeholder reporting** The most formal type of reporting that you might have is the set of reports prepared for management or external stakeholders. Although these reports might be viewed online, you should anticipate that they might be printed or shared as PDF documents. You typically apply consistent styling and branding to this type of report and must take care to adjust the design so that the report produces a clean layout in print or PDF format. We provide suggestions for styling in the "Developing standards" section of this chapter.
- **Information as a service** You might produce a series of reports as a service to customers. This type of reporting often requires consistency in appearance. You might provide access to reports online, or you might deliver a set of reports as a single PDF document. To combine reports into one document, you create individual reports and then create one parent report that uses subreports to organize the individual reports in the correct sequence. When working with subreports in this way, to avoid introducing blank pages, you must take care to manage the page sizes of the individual reports as well as the parent report. See Chapter 14 for information about subreports, Chapter 15 for page size management, and Chapter 32, "Understanding Report Definition Language," for an alternative approach to combining reports.

Reviewing report options

Although many changes to a report can be made at any time during report development, the most important decision you need to make early in the process is the type of layout to use. However, there are other options to consider as well. You need to understand any limitations you might have when retrieving data, and you must give some thought to whether to implement parameters and how best to configure them. If your users intend to access reports by using mobile devices, you should also consider how best to design reports to accommodate the smaller form factor. Similarly, you need to consider how printing a report affects the layout.

Layout decisions

With some practice and a good understanding of how a tablix works, you can convert a table to a matrix or vice versa, but when you're new to Reporting Services, you might find the necessary steps to be unintuitive. Moreover, you cannot change a table into a chart or map nor turn a list with various nested data regions into a table. Therefore, deciding the layout to use is a critical decision to make.

You can use a table layout when you need to present a simple list of items with a relatively small number of fields per item. If there are a lot of fields, you need to consider whether it's acceptable for the user to scroll horizontally to view all the fields online. If the user is going to print the report, there are additional considerations we discuss in the "Printed reports" section of this chapter.

Generally, you should design the reports for online viewing to display a limited number of columns. It's much easier for users to scroll up and down to view information than to scroll left and right. For that matter, you should question whether users really need to see all available columns at once. It

might make more sense to set up a parameter to allow users to choose the columns they want to see. Then you can hide or show columns on demand.

You can also use multiple detail rows and stack the fields vertically if necessary, as shown in Figure 6-1. That way, you can include a larger number of fields in the layout while keeping the width of the data region smaller than the user's screen width or the printed page.

FIGURE 6-1 You can use multiple detail rows to reduce the number of columns required to display all fields for a dataset row.

Another option for handling a large number detail fields is to create a grouped list. The advantage of a list is the ability to arrange text boxes in a freeform fashion. You can then accommodate text boxes of varying sizes on each "row" of the list, as shown in Figure 6-2.

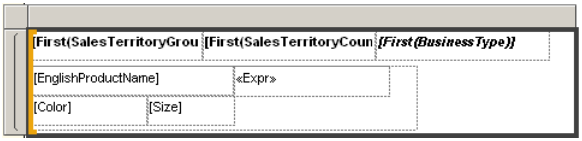


FIGURE 6-2 You can arrange text boxes inside a grouped list to minimize the width of a report.

A matrix is useful when you need to provide a crosstab layout. However, you run the risk of having too many columns on the screen or printed page if you don't plan ahead. A table has a fixed number of columns that you define during report development, but a matrix can have a variable number of columns because the data determines the column groupings. Therefore, if your goal is to fit a matrix to a specific size, you should consider implementing a filter to reduce the number of possible column groupings.

On a more granular level, you need to consider how wide each text box should be to comfortably accommodate data. A text box will never grow wider than the size you define, but it has a CanGrow property that is set to True by default, which expands the text box vertically if the data is wider than the text box, as shown in Figure 6-3. This property might be acceptable for reports that you view online but can disrupt the alignment of printed reports by pushing report items from one page to another.

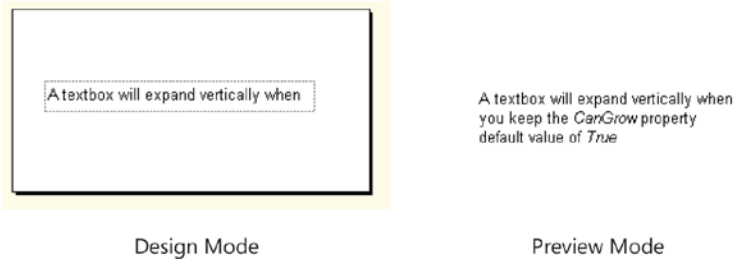


FIGURE 6-3 When the *CanGrow* property is set to *True*, the text box expands vertically to display the entire text it contains.

Data considerations

Rather than create a large report that contains hundreds or thousands of pages, find out whether all that data is really necessary. Usually it's better to add a filter or create a query that aggregates data at the source. When users view reports online, they usually want to look up a piece of information and don't need to scan an entire report to find that one item of needed information.

Rather than create a report that includes pages of detail records with group totals, create a summary report to show only group totals. Have users start their review process from a summary report, and then include an action on the report to drill through to a detail report. The detail report displays only a subset of data related to the portion of the summary report that the user wants to investigate. In both cases, you have a report that retrieves a smaller number of records and generally runs faster than a report that retrieves thousands of records.

Another aspect of managing data in reports is the data definition itself. Some organizations have strict control over access to data sources, requiring report developers to use stored procedures or views maintained in the source database. This requirement simplifies maintenance of commonly used queries by centralizing the query.

Using stored procedures and views is not an option for data that comes from non-relational data sources. For those situations, you might consider using shared datasets to achieve a similar effect. The downside of shared datasets is that they are primarily intended for use in Report Builder. If you use SSDT to develop reports, you must either obtain the current version of a shared dataset from a source control system or download it from the report server. SSDT does not provide direct access to published shared datasets.

Parameters

Parameters are useful elements of a report. You can use them to filter data, set report item properties dynamically at run time, or provide input for calculations. Throughout this chapter, we suggest using parameters to address a variety of scenarios. As you plan your report, you must make several decisions about implementing each parameter, including the following:

- **User input** You must decide whether a parameter accepts user input. In most cases, you should avoid this option due to the increased security risk of an injection attack and due to the limited validation available in Reporting Services. Usually it's better to give the user a list of values from which to make a selection, but you might prompt the user when you want to create a filter by using a user-defined string and wildcard or when the number of possible values is not practical to present in a list, such as a sales invoice number.
- **List of values** The preferred method for prompting the user for a value is to provide a list of valid values. You can hard-code this list if necessary, but using a query gives you more flexibility in the long run and allows you to easily reproduce the same list of values when using the same parameter in multiple reports. You can create an independent query, or you can use a query that depends on the user's selection of a value for a separate parameter. This technique is known as cascading parameters and is described in Chapter 14. You must also decide whether the user can select only one value or multiple values.
- **Default value** If you do not include a default value for a report parameter, the report cannot execute until the user provides a value. This might be a desired behavior when there is no reasonable default, such as a report that displays information about a specific sales invoice. Wherever possible, you should provide a default value. This default can be hard-coded in the report or produced as the result of a query.

Mobile devices

We pointed out earlier in this chapter that you should design reports to use screen space effectively and minimize scrolling when people are working on a desktop. However, with mobile devices, the same rules do not necessarily apply. Users are accustomed to not seeing all information on the screen at one time. A useful feature of tablet devices is the ability to use gestures on the screen to navigate, either by zooming in and out as necessary or scrolling. Therefore, you might not need to worry as much about fitting the contents of a report to the screen, but always check with your users to understand their expectations before establishing a policy one way or the other.

If you are reproducing a dashboard for mobile devices, you might consider limiting the number of perspectives or views of the data to no more than four. If you have four views laid out in a quadrant format, keep in mind that the most important information should be placed in the top-left quadrant. That's where our eyes naturally start, and on the tablet, that should be the information that's most likely to fit completely on the screen.

Another suggestion is to keep the report simple for mobile users. Having too much information on the screen makes it more difficult for users to focus on the information they're looking for, especially if they're in a hurry. Having legends or lots of parameters for filtering can create visual noise and interfere with their ability to locate the data. In particular, when you know your users will be viewing information on tablets, don't use tiny fonts. In addition, if there are actions in a report, associate the actions with report items big enough to handle comfortably on a touch screen.

Printed reports

When designing reports that are destined for printing at some point, you must factor in the page size as you position items in the report layout. You can set report properties to fix the orientation as portrait or landscape, which we explain in Chapter 15. Many of the same considerations that we describe in the "Layout decisions" section of this chapter also apply to printed reports, and perhaps more so. Preview your report, and switch to Print Layout mode so that you can see how the rendered report fits on each page. You should also export the report as a PDF file to check the rendering more carefully.

If a report spans multiple pages, you should configure repeating column headers in the tablix. That way, users don't have to flip back to the first page to determine what a particular column contains. Chapter 15 explains how to set properties in the tablix to repeat the column headers. The same principle applies to row headers for a matrix. You can repeat the row headers on the second page if a matrix is too wide to fit on a single page.

By default, Reporting Services tries to fit as much content as it can on a page and inserts logical page breaks for a report. For greater control over the location of page breaks, you can define explicit page breaks relative to report items or groups within a data region.

Regardless of whether you set explicit page breaks or rely on the default logical page breaks, each page of the report is assigned a page number. You can reset this page number to maintain a separate page numbering system within sections of a report, but there is also an overall page number that you can use. When you know a report will be printed, it's a good idea to include the current page number and the overall page number in the page header or page footer, which are the only locations in which the page number can display. That way, if a user inadvertently shuffles the pages of the report or misplaces ones, the presence of page numbers helps the user reorder the pages correctly or notice the missing page.

You should also include the report execution date and time in the page header or page footer. That way, the user can easily tell if the report is recent enough to remain useful. You might also consider including the report server name and report path to help the user locate the report online when it's time to execute a fresh report.

Sometimes you can design a report that looks good both online and in print, but that's not always possible. You might need to design two separate reports. Rather than have users open one report and then have to switch to another report, consider having a link visible in online mode only that users click to output the report as PDF or Excel. Reporting Services doesn't actually allow you to create links dynamically, but you can create an action and then underline the associated text to make it appear to be a link. In the action definition, you can use an expression to dynamically create the URL to produce the requested output. See Chapter 13 for information about actions and Chapter 35, "Programming report access," for information about URL access.

Developing standards

When your reports have a consistent look and feel, users know what to expect when they view a report and can focus on its important elements rather than spending time trying to figure out how to use the report. Similarly, during the report development cycle, standards provide clear guidelines that you can apply to your report quickly without resorting to trial and error to see what might look good. When you have a set of standards established, you can create a variety of templates with common elements in place to shortcut the development process, as we explain in the "Knowing your audience" section of this chapter.

If you don't already have reporting standards already established in your organization, consider defining standards for the following report elements:

- **Fonts** Use a single font family. Be consistent with font sizes for different elements. For example, you might use a 10pt font for detail records and a 12pt font for column headers. Use 14pt or 16pt for titles. Use smaller fonts for text in page headers and page footers. Consider using a lighter shading for text in page headers and page footers, such as gray rather than black, to reduce the emphasis on the text. The user's eye should be drawn towards the primary information on the report through the use of larger fonts, darker colors, and bolding where appropriate.
- **Colors** Use a consistent palette of colors for text and elements (other than data visualizations) that place color onto the report, such as lines, borders, or fill. Ideally, colors should be soft and neutral and should not distract from the information on the report.
- **Logo** If your company has a logo to include on reports, upload it directly to the report server and then reference it as an external image in each report by using its URL access path (described in Chapter 35). That way, if the logo changes in the future, you can replace the file on the report server without the need to update each report individually.
- **Page Header** Include a text box to display the report title and another text box to display the page number and total pages (such as Page 1 of 3). If you prefer, you can place the page number in the page footer. Suppress the page header on the first page if using a text box to display a title on the report body. Use a font smaller than the detail row fonts in the report.
- **Report Header** The report does not include an explicit report header. We use this term to describe any report items, such as a free-standing text box, a rectangle, a line, or an image, that display at the top of the report in the body above a data region. The report header displays only once in the rendered report at the top of the first page. You can use this area to display a title and supplemental information as a subtitle, such as parameter selections and a date. Use a large font for the report title and a slightly smaller font for the subtitle.

- **Group Header or Footer** Use a consistent styling for group headers and footers when you include them in a Tablix. Typically, you use a bold font to distinguish headers and footers from details. Consider using border or background color properties as another means to differentiate group headers and footers from details. Decide whether to include totals in the group header or footer. You should be consistent with the placement of totals in all reports.
- **Page Footer** Include a text box to display the report execution date, report path, and the page numbering if you don't include it in the page header. Use a font smaller than the detail row fonts in the report.
- **Margins** Set a minimum margin of 0.5in on each side of the page.
- **Parameters** When defining a common set of parameters across multiple reports, use the same sequence in each report.
- **Calculations** Decide how to handle a potential divide by zero error. You might use a conditional expression to test the calculation first and return a default string such as N/A or a dash (-).
- **Empty strings and NULL values** You should use a consistent rule for handling empty strings or NULL values returned in the dataset. You might prefer to display nothing, convert numeric data types to a 0, or display a string. Whatever you decide to do, be consistent.
- **No rows** If it is possible for a dataset query to return no rows, you can display a message to the user by setting the *NoRowsMessage* property of the data region. If you leave this property blank, the data region does not display. If it's the only element in the report, the user might not realize the problem is related to data and might think that the report server has failed to render the report correctly.

Index

Symbols

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

About the authors

STACIA MISNER is the founder of Data Inspirations, which delivers global business intelligence (BI) consulting and education services. She is a consultant, educator, mentor, and author specializing in BI and performance management solutions using Microsoft technologies. Stacia has more than 28 years of experience in information technology and has focused exclusively on Microsoft BI technologies since 2000. She is the author of *Microsoft SQL Server 2000 Reporting Services Step by Step*, *Microsoft SQL Server 2005 Reporting Services Step by Step*, *Microsoft SQL Server 2008 Reporting Services Step by Step*, and *Microsoft SQL Server 2005 Express Edition: Start Now!* and the coauthor of *Business Intelligence: Making Better Decisions Faster*, *Microsoft SQL Server 2005 Analysis Services Step by Step*, *Microsoft SQL Server 2005 Administrator's Companion, Introducing Microsoft SQL Server 2008 R2, and Introducing Microsoft SQL Server 2012*. She is a SQL Server Most Valuable Professional (MVP) and also a Microsoft Certified IT Professional-BI and a Microsoft Certified Technology Specialist-BI. In addition, she has achieved the SSAS Maestro certification. Stacia lives on a remote island in Southeast Alaska with her husband, Gerry, her two dogs, and lots of bears, wolves, and bald eagles in the surrounding wilderness.

ERIKA BAKSE is currently a consultant with Symmetry Corporation, having got her start in Business Intelligence at Data Inspirations under the incomparable tutelage of Stacia Misner. Erika is the contributing author of *Microsoft SQL Server 2008 Reporting Services Step by Step*. She is a Microsoft Certified Technology Specialist with a love of theoretical mathematics and parrotkind, residing in Berkeley, CA.