

Contents

[What is SQL Server Reporting Services \(SSRS\)?](#)

[Overview](#)

[What's new in Reporting Services](#)

[Change log](#)

[Reporting Services Concepts \(SSRS\)](#)

[Reporting Services Features and Tasks](#)

[Backward Compatibility](#)

[Deprecated features](#)

[Discontinued functionality in SSRS in SQL Server 2016](#)

[Breaking changes in SSRS in SQL Server 2016](#)

[Behavior changes to SQL Server Reporting Services in SQL Server 2016](#)

[Planning](#)

[Choosing BI tools for analysis and reporting](#)

[Browser support for Reporting Services and Power View](#)

[Plan for report design and report deployment](#)

[Reporting Services features in editions of SQL Server](#)

[Install](#)

[Web portal](#)

[Branding the web portal](#)

[Working with shared datasets](#)

[Working with paginated reports](#)

[Working with snapshots](#)

[Working with subscriptions](#)

[Working with KPIs](#)

[My Settings for Power BI Integration](#)

[Report Server](#)

[Report Server \(SharePoint\)](#)

[Pin to Power BI](#)

[Report Builder](#)

[Report Design](#)

[Working with reports](#)

[Report Data](#)

[Mobile reports](#)

[Security and protection](#)

[Subscriptions and delivery](#)

[Tools +](#)

[Data Alerts \(SharePoint\)](#)

[Data Alert Designer](#)

[Data Alert Manager for SharePoint Users](#)

[Data Alert Manager for Alerting Administrators](#)

[Data Alert Messages](#)

[Create a Data Alert in Data Alert Designer](#)

[Edit a Data Alert in Alert Designer](#)

[Manage All Data Alerts on a SharePoint Site in Data Alert Manager](#)

[Manage My Data Alerts in Data Alert Manager](#)

[Grant Permissions to Users and Alerting Administrators](#)

[Extensions](#)

[Customize Rendering Extension Parameters in RSReportServer.Config](#)

[URL Access](#)

[Access Report Server Items Using URL Access](#)

[Pass a Report Parameter Within a URL](#)

[URL Access Parameter Reference](#)

[Set the Language for Report Parameters in a URL](#)

[Specify Device Information Settings in a URL](#)

[Export a Report Using URL Access](#)

[Search a Report Using URL Access](#)

[Render a Report History Snapshot Using URL Access](#)

[WMI provider](#)

[Troubleshooting +](#)

[Developer](#)

[REST API](#)

[REST API samples](#)

[Application integration](#)

[Report server web service](#)

[Extensions](#)

[Custom report items](#)

[Custom assemblies](#)

[Technical Reference](#)

[Feature Reference](#)

[Device Information Settings for Rendering Extensions](#)

- [ATOM Device Information Settings](#)
- [CSV Device Information Settings](#)
- [Excel Device Information Settings](#)
- [HTML Device Information Settings](#)
- [Image Device Information Settings](#)
- [MHTML Device Information Settings](#)
- [PDF Device Information Settings](#)
- [PPTX Device Information Settings](#)
- [XML Device Information Settings](#)
- [Word Device Information Settings](#)
- [RGDI Device Information Settings](#)

[HTML Viewer and the Report Toolbar](#)

[Tutorials](#)

[Create a Basic Table Report \(SSRS Tutorial\)](#)

- [Lesson 1: Creating a Report Server Project](#)
- [Lesson 2: Specifying Connection Information](#)
- [Lesson 3: Defining a Dataset for the Table Report](#)
- [Lesson 4: Adding a Table to the Report](#)
- [Lesson 5: Formatting a Report](#)
- [Lesson 6: Adding Grouping and Totals](#)

[Create a Data-Driven Subscription \(SSRS Tutorial\)](#)

- [Lesson 1: Creating a Sample Subscriber Database](#)
- [Lesson 2: Modifying the Report Data Source Properties](#)

[Lesson 3: Defining a Data-Driven Subscription](#)

[Create a Drillthrough \(RDLC\) Report with Parameters using ReportViewer \(SSRS Tutorial\)](#)

[Lesson 1: Create a New Web Site](#)

[Lesson 2: Define a Data Connection and Data Table for Parent Report](#)

[Lesson 3: Design the Parent Report using the Report Wizard](#)

[Lesson 4: Define a Data Connection and Data Table for Child Report](#)

[Lesson 5: Design the Child Report using the Report Wizard](#)

[Lesson 6: Add a ReportViewer Control to the Application](#)

[Lesson 7: Add Drillthrough Action on Parent Report](#)

[Lesson 8: Create a Data Filter](#)

[Lesson 9: Build and Run the Application](#)

[Report Builder Tutorials](#)

[Prerequisites for Tutorials](#)

[Alternative Ways to Get a Data Connection \(Report Builder\)](#)

[Tutorial: Creating a Basic Table Report](#)

[Tutorial: Creating a Matrix Report](#)

[Tutorial: Creating a Free Form Report](#)

[Tutorial: Format Text \(Report Builder\)](#)

[Tutorial: Add a Column Chart to Your Report \(Report Builder\)](#)

[Tutorial: Add a Pie Chart to Your Report \(Report Builder\)](#)

[Tutorial: Add a Bar Chart to Your Report \(Report Builder\)](#)

[Tutorial: Add a Sparkline to Your Report \(Report Builder\)](#)

[Tutorial: Adding a KPI to Your Report](#)

[Tutorial: Map Report](#)

[Tutorial: Add a Parameter to Your Report \(Report Builder\)](#)

[Tutorial: Creating Drillthrough and Main Reports](#)

[Tutorial: Introducing Expressions](#)

What is SQL Server Reporting Services (SSRS)?

11/30/2018 • 4 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server 2016 Reporting Services and later Power BI Report Server

Looking for Power BI Report Server? See [What is Power BI Report Server?](#).

Create, deploy, and manage mobile and paginated Reporting Services reports on premises with the range of ready-to-use tools and services that SQL Server Reporting Services (SSRS) provides.



Create, deploy, and manage mobile and paginated reports

SQL Server Reporting Services is a solution that customers deploy on their own premises for creating, publishing, and managing reports, then delivering them to the right users in different ways, whether that's viewing them in web browser, on their mobile device, or as an email in their in-box.

SQL Server Reporting Services offers an updated suite of products:

- **"Traditional" paginated reports** brought up to date, so you can create modern-looking reports, with updated tools and new features for creating them.
- **New mobile reports** with a responsive layout that adapts to different devices and the different ways you hold them.
- **A modern web portal** you can view in any modern browser. In the new portal, you can organize and display mobile and paginated Reporting Services reports and KPIs. You can also store Excel workbooks on the portal.

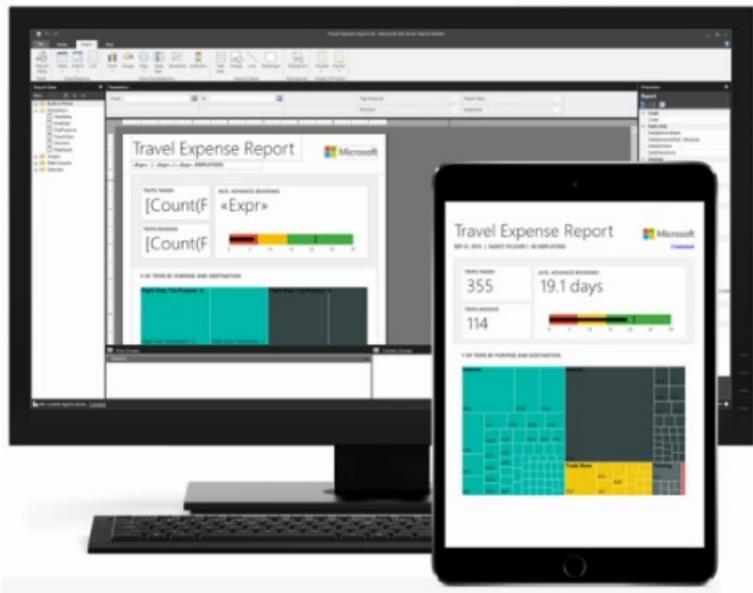
Read on for more about each.

What's new in Reporting Services

These sources will keep you up-to-date on new features in SQL Server Reporting Services.

- [What's New in Reporting Services](#)
- [SQL Server Reporting Services Team Blog](#)
- [The Guy in a Cube YouTube channel](#)

Paginated reports



Reporting Services is associated with "traditional" paginated document-style reports, in which the more data you have, the more rows in the tables, and the more pages the report would have. That's great for generating fixed-layout, pixel-perfect documents optimized for printing, such as PDF and Word files.

That core BI workload still exists today, so we've modernized it. Now you can create modern-looking reports with updated new features, using [Report Builder](#) or Report Designer in [SQL Server Data Tools \(SSDT\)](#).

- We updated all the default styles and color palettes, so by default you create reports with a new minimalist modern style.
- We updated the Parameter pane, so you can arrange parameters however you want.
- You can export to new formats such as PowerPoint. Reporting Services visualizations in PowerPoint are live and editable, not just screen shots.
- You can create a hybrid Power BI/Reporting Services experience: Rather than recreating your on-premises Reporting Services reports in Power BI, you can pin visuals from those reports to your Power BI dashboards. Then you can monitor everything in one place on your Power BI dashboard.

Mobile reports



Mobile computing has shifted the devices we need to work, meaning people today have a different reporting need. The fixed-layout report experience doesn't work well when you introduce tablets and phones. Something designed for a wide PC screen isn't the optimal experience on a small phone screen that's not just smaller but a

portrait or landscape orientation.

What you need with these widely different screen form factors is not a fixed layout, but a responsive layout that adapts to these different devices and the different ways you hold them. For that we've added a new report type: mobile reports, based on the Datazen technology we acquired about a year ago and integrated into the product. You can migrate your existing Datazen reports to Reporting Services with the [SQL Server Migration Assistant for Datazen](#).

You create these mobile reports in the new [Mobile Report Publisher](#) app. Then in the native [Power BI apps for mobile devices](#) for Windows 10, iOS, Android, and HTML5, you can access the data you have in Power BI the cloud, plus your on-premises SQL Server Reporting Services data. As you create visualizations, Mobile Report Publisher automatically generates sample data for each, so you see how the visualization will look with your data, and what kind of data works well in each visualization.

Web portal



For end users of native-mode Reporting Services, the front door is a modern web portal you can view in any modern browser. You can access all your Reporting Services mobile and paginated reports and KPIs in the new portal.

You can apply your own custom branding to your web portal. And you can create KPIs right in the web portal. KPIs can surface key business metrics at a glance in the browser, without having to open a report.

The new web portal is a complete rewrite of Report Manager. Now it's a single-page, standards-based HTML5 app, which modern browsers are optimized for: Edge, Internet Explorer 10 and 11, Chrome, Firefox, Safari, and all the major browsers.

The content on the web portal is organized by type: Reporting Services mobile and paginated reports and KPIs, Excel workbooks, shared datasets, and shared data sources to use as building blocks for your reports. You can store and manage them securely here, in the traditional folder hierarchy. You can tag your favorites, and you can manage the content if you have that role.

And you can still schedule report processing, access reports on demand, and subscribe to published reports in the new web portal.

More about the [Web portal \(SSRS Native Mode\)](#).

Reporting Services in SharePoint integrated mode

You publish reports to Reporting Services in SharePoint integrated mode. You can schedule report processing, access reports on demand, subscribe to published reports, and export reports to other applications such as Microsoft Excel. Create data alerts on reports published to a SharePoint site and receive email messages when report data changes.

More about [Reporting Services Report Server in SharePoint integrated mode](#).

Reporting Services programming features

Take advantage of Reporting Services programming features so you can extend and customize your reporting functionality, with APIs to integrate or extend data and report processing in custom applications.

[More Reporting Services Developer Documentation](#).

Next steps

- [Install Reporting Services](#)
- [Install Report Builder](#)
- [Download SQL Server Data Tools \(SSDT\)](#)

More questions? [Try asking the Reporting Services forum](#)

What's new in SQL Server Reporting Services (SSRS)

12/14/2018 • 8 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server 2016 Reporting Services and later Power BI Report Server

Learn about what's new in SQL Server Reporting Services. This covers the major feature areas and is updated as new items are released.

For the current release notes, see [SQL Server 2017 Release Notes](#).

For information about Power BI Report Server, see [What is Power BI Report Server?](#).

Download

To download SQL Server 2017 Reporting Services, go to the [Microsoft Download Center](#).

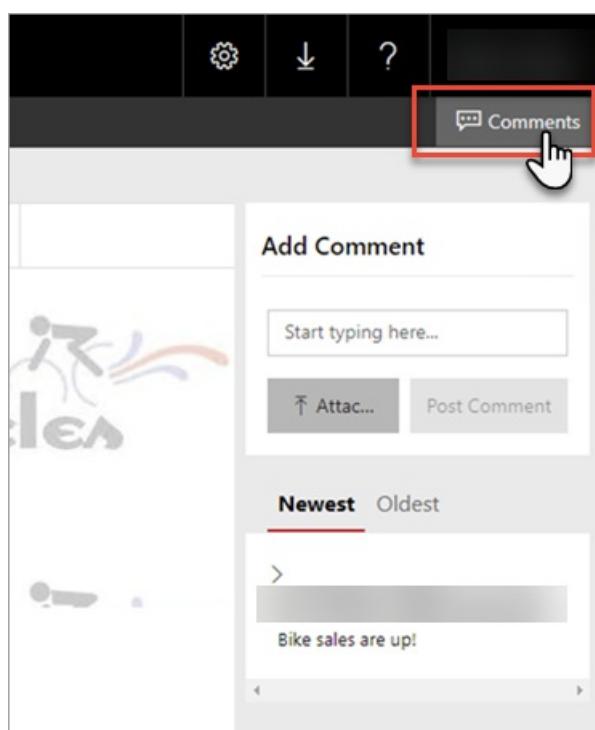
SQL Server 2019 preview Reporting Services

SQL Server 2019 preview Reporting Services isn't available for CTP 2.2. Install the current version, [SQL Server 2017 Reporting Services](#).

SSRS 2017

Comments on reports

Comments are now available for reports, to add perspective and collaborate with others. You can also include attachments with comments.



For more information, see [Add comments to a report in a report server](#).

DAX queries in reporting tools

In the latest releases of Report Builder and SQL Server Data Tools, you can create native DAX queries against supported SQL Server Analysis Services tabular data models by dragging and dropping desired fields in the

query designers. See the [Reporting Services blog](#).

REST API support

To enable development of modern applications and customization, SQL Server Reporting Services now supports a fully OpenAPI compliant RESTful API. The full API specification and documentation can now be found on [swaggerhub](#).

Query designer support for DAX now in Report Builder and SQL Server Data Tools

In Report Builder and SQL Server Data Tools, you can now create native DAX queries against supported SQL Server Analysis Services tabular data models. You can use the query designer in both tools to drag and drop the fields you want and have the DAX query generated for you instead of writing it yourself.

Read more on the [Reporting Services blog](#).

- Download [SQL Server Report Builder](#).
- Download [SQL Server Data Tools - Release Candidate](#).

Note: You can only use the query designer for DAX with SSAS tabular data sources built in SQL Server 2016+.

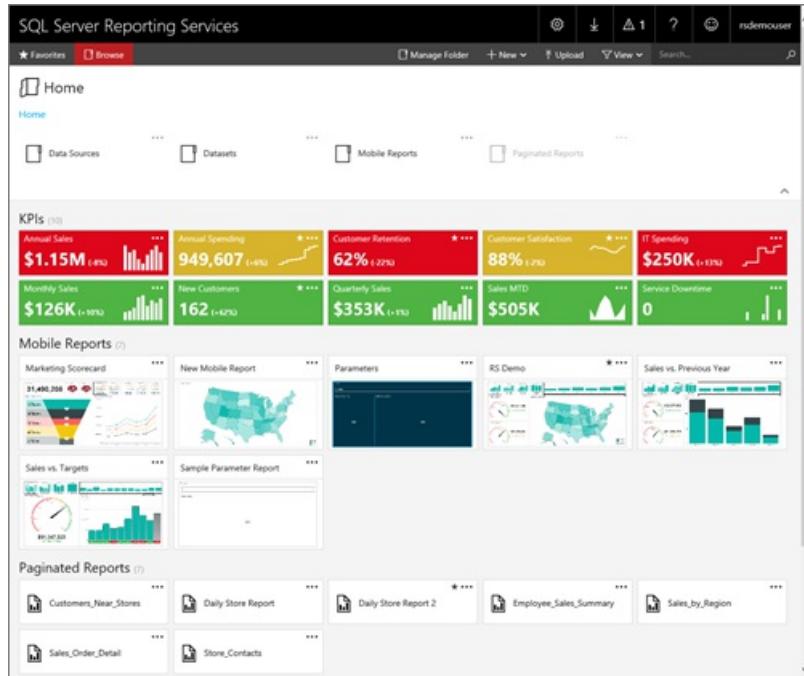
SSRS 2016

Reporting Services web portal

A new Reporting Services web portal is available. This is an updated, modern, portal which incorporates KPIs, Mobile Reports, Paginated Reports, Excel and Power BI Desktop files. The web portal replaces Report Manager from previous releases. You can also download Mobile Report Publisher and Report Builder from the web portal without the need of ClickOnce technology.

To create Mobile Reports, you will need the Mobile Report Publisher.

For more information about the web portal, see [Web portal \(SSRS Native Mode\)](#).



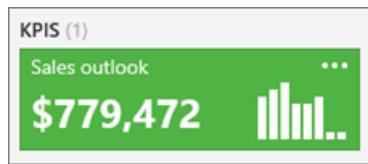
Custom branding for the web portal

You can customize the web portal with your organization's logo and colors by using a branding pack.

For more information about custom branding, see [Branding the web portal](#)

Key performance indicators (KPI) in the web portal

You can create KPIs direct in the web portal that are contextual to the folder you are in. When creating KPIs, you can choose dataset fields and summarize those values. You can also select related content to drill-through to more details.



For more information, see [Working with KPIs in the web portal](#)

Mobile Reports

Reporting Services mobile reports are dedicated reports optimized for a wide variety of form factors and provide an optimal experience for users accessing reports on mobile devices. Mobile reports feature a assortment of visualizations, from time, category, and comparison charts, to treemaps and custom maps. Connect your mobile reports to a range of data sources, including on-premises SQL Server Analysis Services multidimensional and tabular data. Lay out your mobile reports on a design surface with adjusting grid rows and columns, and flexible mobile report elements that scale well to any screen size. Then save these mobile reports to a Reporting Service server, and view and interact with them in a browser or in the Power BI mobile app on iPads, iPhones, Android phones and Windows 10 devices.

Mobile Report Publisher

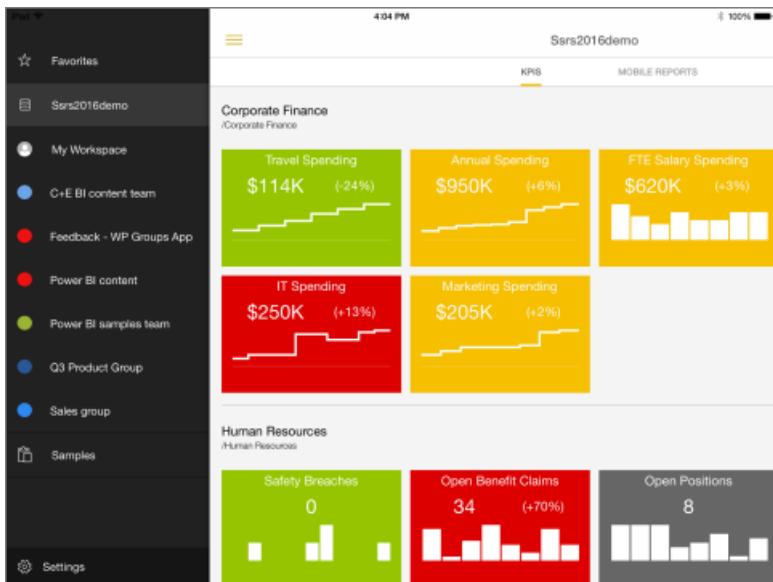
The SQL Server Mobile Report Publisherallows you to create and publish SQL Server mobile reports to your Reporting Services web portal.



For more information, see [Create mobile reports with SQL Server Mobile Report Publisher](#).

SQL Server mobile reports hosted in Reporting Services available in Power BI Mobile app

The Power BI Mobile app for iOS on iPad and iPhone can now display SQL Server mobile reports hosted on your local report server.



You can't connect by default without some configuration changes. For more information on how to allow the Power BI Mobile app to connect to your report server, see [Enable a report server for Power BI Mobile access](#).

Support of SharePoint mode and SharePoint 2016

SQL Server 2016 (13.x) Reporting Services supports integration with SharePoint 2013 and SharePoint 2016.

For more information, see:

- [Supported Combinations of SharePoint and Reporting Services Server and Add-in \(SQL Server 2016\)](#)
- [Where to find the Reporting Services add-in for SharePoint Products](#)
- [Install Reporting Services SharePoint Mode](#)

Microsoft .NET Framework 4 Support

SQL Server 2016 Reporting Services (SSRS) supports the current versions of Microsoft .NET Framework 4. This includes version 4.0 and 4.5.1. If no version of .Net Framework 4.x is installed, SQL Server setup installs .NET 4.0 during the feature installation step.

Report improvements

HTML 5 Rendering Engine: A new HTML5 rendering engine that targets modern web "full" standards mode and modern browsers. The new rendering engine no longer relies on quirks mode used by a few older browsers.

For more information on browser support, see [Browser Support for Reporting Services and Power View](#).

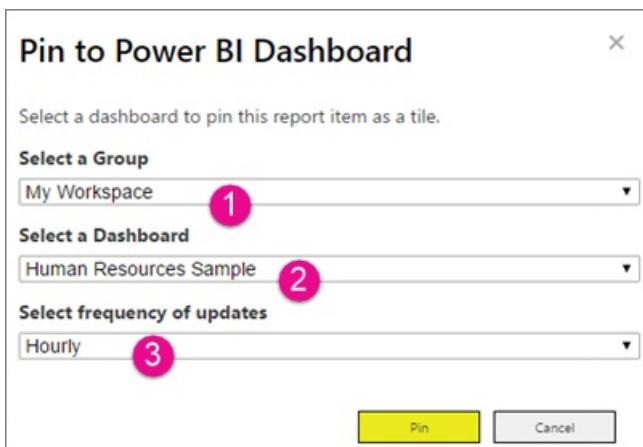
Modern paginated reports: Design beautifully modern paginated reports with new, modern styles for charts, gauges, maps and other data visualizations.



Tree Map and Sunburst Charts: Enhance your reports with Tree Map and Sunburst charts, great ways to display hierarchical data. For more information, see [Tree Map and Sunburst Charts in Reporting Services](#).

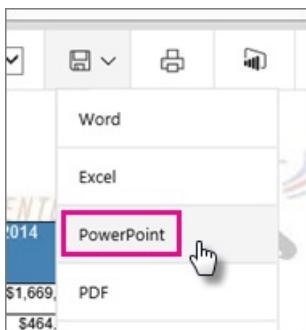
Report embedding: You can now embed mobile and paginated reports in other web pages, and applications, by using an iframe along with URL parameters.

Pin Report Items to a Power BI Dashboard: While viewing a report in the web portal, you can select report items and pin them to a Power BI dashboard. The items you can pin are charts, gauge panels, maps, and images. You can **(1)** select the group that contains the dashboard you want to pin to, **(2)** select the dashboard you want to pin the item too and **(3)** select how frequently you want the tile updated in the dashboard. The refresh is managed by Reporting Services subscriptions and after the item is pinned, you can edit the subscription and configure a different refresh schedule.



For more information, see [Power BI Report Server Integration \(Configuration Manager\)](#) and [Pin Reporting Services items to Power BI Dashboards](#).

PowerPoint Rendering and Export: The Microsoft PowerPoint (PPTX) format is a new SQL Server 2016 Reporting Services (SSRS) rendering extension. You can export reports in the PPTX format from the usual applications; Report Builder, Report Designer (in SSDT), and the web portal. For the example the following image shows the export menu from the web portal.



You can also select the PPTX format for subscription output and use Report Server URL access to render and export a report. For example the following URL command in your browser exports a report from a named instance of the report server.

```
https://servername/ReportServer_THESQLINSTANCE/Pages/ReportViewer.aspx?  
%2freportfolder%2freport+name+with+spaces&rs:Format=pptx
```

For more information, see [Export a Report Using URL Access](#).

PDF Replaces ActiveX for Remote Printing: The report viewer toolbar ActiveX print experience has been replaced with a modern PDF based experience that works across the matrix of supported browsers, including Microsoft Edge. No more ActiveX controls to download! Depending on the browser you use and the PDF viewing applications and services you have installed, Reporting Services will either open a print dialog to print your report or prompt you to download a .PDF file of your report. As an administrator, you can still disable client side printing from Management Studio. For more information, see [Enable and Disable Client-Side Printing for Reporting Services](#).



Subscription Improvements

FEATURE	SUPPORTED SERVER MODE
<p>Enable and disable subscriptions. New user interface options to quickly disable and enable subscriptions. The disabled subscriptions maintain their other configuration properties such as schedule and can be easily enabled.</p>  <p>For more information, see Disable or Pause Report and Subscription Processing.</p>	Native mode
<p>Subscription description. When you create a new subscription, you can now include a description of the report as part of the subscription properties. The description is included on the subscription summary page.</p>	SharePoint and Native mode
<p>Change subscription owner. Enhanced user interface to quickly change the owner of a subscription. Previous versions of Reporting Services allow administrators to change subscription owners using script. Starting with the SQL Server 2016 (13.x) release, you can change subscription owners using the user interface or script. Changing the subscription owner is a common administrative task when users leave or change roles in your organization.</p>	SharePoint and Native mode
<p>Shared credential for file share subscriptions. Two workflows now exist with Reporting Services file share subscriptions:</p> <p>New in this release, your Reporting Services administrator can configure a single file share account, that is used for one to many subscriptions. The file share account is configured in the Reporting Services native mode configuration manager Specify a file share account, and then on the subscription configuration page, users select Use file share account.</p> <p>Configure individual subscriptions with specific credentials for the destination file share.</p> <p>You can also mix the two approaches and have some file share subscriptions use the central file share account while other subscriptions use specific credentials.</p>	Native mode

SQL Server Data Tools (SSDT)

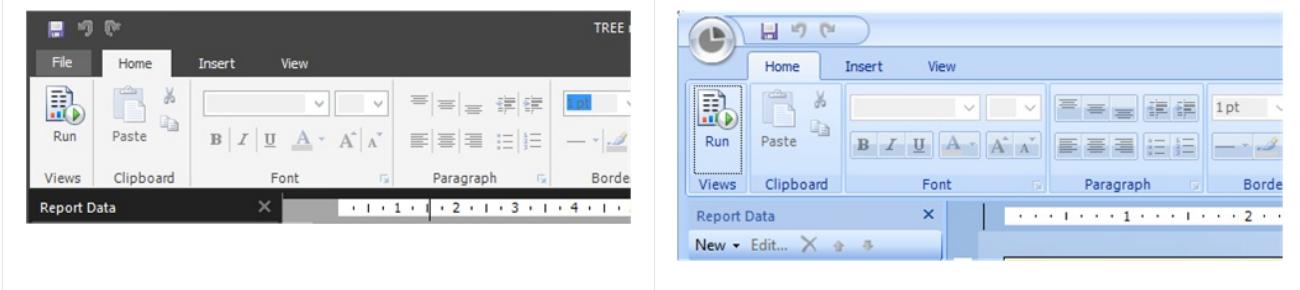
The new release of SSDT includes the project templates for SQL Server 2016 Reporting Services (SSRS): Report Server Project Wizard and Report Server Project. For information about downloading SSDT, see [SQL Server Data Tools for Visual Studio 2015](#).

Report Builder improvements

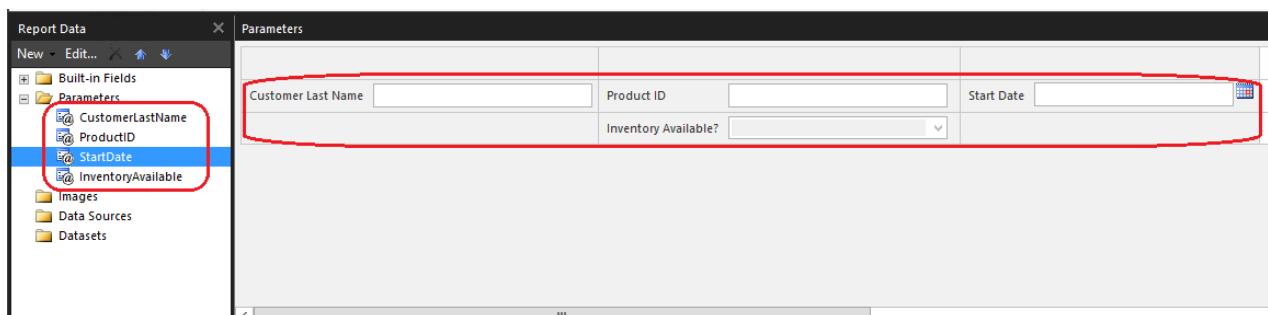
New Report Builder User Interface: The core Report Builder user interface is now a modern look and feel with streamlined UI elements.

New

Previous



Custom Parameters Pane: You can now customize the parameters pane. Using the design surface in Report Builder, you can drag a parameter to a specific column and row in the parameters pane. You can add and remove columns to change the layout of the pane. For more information, see [Customize the Parameters Pane in a Report \(Report Builder\)](#).



High DPI Support: Report Builder supports High DPI (Dots Per Inch) scaling and devices. For more information on High DPI, see the following:

- [Windows 8.1 DPI Scaling Enhancements](#)
- [High DPI and Windows 8.1](#)

Next steps

[What's New in Analysis Services](#)

[Backward Compatibility](#)

[Reporting Services Features supported by the Editions of SQL Server](#)

[Upgrade and Migrate Reporting Services](#)

[Reporting Services](#)

More questions? Try asking the [Reporting Services forum](#)

Change log for SQL Server Reporting Services (SSRS) 2017 and later

10/24/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO:  SQL Server Reporting Services (2017 and later)

This article describes changes in Reporting Services.

SQL Server 2017 Reporting Services

Version 14.0.600.906, Released: September 12, 2018

These bugs have been fixed:

- Custom Authentication not returning correct cookie information

Version 14.0.600.892, Released: August 31, 2018

These bugs have been fixed:

- Textbox inside Rectangle causes rectangle not to grow vertically when rc:Toolbar=False and it has long text
- Text size is not scaling if pageHeight is less than 0.5 in
- Deadlock in SSRS catalog database when used with CRM
- Vertically aligned column headers incorrectly displayed when scrolling down in report
- Users added to SCOM Reporting Role will have access blocked to SSRS web portal
- Thai character is not getting exported correctly in PDF
- Browser Role Behavior Change
- rc:Toolbar=false doesn't work in Express edition
- Missing vertical scrollbar in parameter prompt area
- Updated Mobile Report Runtime

Version 14.0.600.744, Released: April 25, 2018

These bugs have been fixed:

- Data Driven Subscription page does not show the Delivery Option once it is created
- Upgrading SSRS 2012 to SSRS 2017 results in RSManagement throwing an exception every few seconds
- Cannot change defaults values for multi-value parameters in IE11
- Schedules are empty whenever shared schedule is executed

Version 14.0.600.689, Released: February 28, 2018

These bugs have been fixed:

- Report Parameter visibility in a linked report is reverted after editing its properties
- URL Parameter rc:Toolbar=false doesn't work in Express edition
- Having expressions in Textbox with CanGrow property set to false is resulting in values not showing
- Added "Learn more" link for product key in setup
- Web portal with custom forms authentication is ignoring sliding expiration cookie
- Export to Word creates unequal row height if row content is empty

Version 14.0.600.594, Released: January 9, 2018

Security updates

Version 14.0.600.490, Released: November 1, 2017

This bug has been fixed:

- Resolved issues with SKU upgrade

Version 14.0.600.451, Released: September 30, 2017

Initial release

Next steps

[What's New in Reporting Services \(SSRS\)](#)

More questions? [Try asking the Reporting Services forum](#)

Reporting Services Concepts (SSRS)

11/15/2018 • 19 minutes to read • [Edit Online](#)

This topic provides a brief summary of SQL Server Reporting Services concepts.

Applies to: Reporting Services Native mode | Reporting Services SharePoint mode

Report server concepts

A report server is a computer that has an instance of Reporting Services installed. A report server internally stores items such as paginated and mobile reports, report-related items and resources, schedules, and subscriptions. A report server can be configured as a stand-alone single server or as a scale out farm, or it can be integrated with SharePoint Server. You interact with report server items through the Reporting Services Web service, WMI provider, URL access, or programmatically through scripts. The way that you interact with a report server depends on the deployment topology and the configuration.

Native mode report servers

A report server configured in native mode is a computer that has SQL Server Reporting Services installed and configured as a stand-alone server. You interact with the report server, reports, and report related items by using a browser with the web portal or URL access commands, SQL Server Management Studio, or programmatically through scripts. For more information, see [Reporting Services Report Server \(Native Mode\)](#).

SharePoint mode report servers

A report server integrated with SharePoint has two possible configurations. In SQL Server 2016 Reporting Services (SSRS), Reporting Services is installed with SharePoint Server as a SharePoint shared service. In earlier releases, the report server integrates with SharePoint Server by installing the Reporting Services SharePoint Add-in. In both cases, you interact with the report server, reports, and report related items by using application pages on the SharePoint site. You use the SharePoint document library and other libraries that you create to store the content types related to reports. For more information, see [Reporting Services Report Server \(SharePoint Mode\)](#).

Report server items

Report server items include paginated and mobile reports, KPIs, shared data sources, shared datasets, and other items that you can publish, upload, or save to a report server. Organize items in the report server hierarchical folder structure on a native report server, or in SharePoint content libraries on a SharePoint site. For more information, see [Report Server Content Management \(SSRS Native Mode\)](#).

Folders

On a native report server, folders provide the hierarchical navigation structure and path of all addressable items stored in a report server. You use the folder hierarchy and site and folder permissions to help control access to report server items, known as *item-level security*. By default, role assignments that you define for specific folders are inherited by child folders in the folder hierarchy. If you assign specific roles to a folder, the inheritance rules no longer apply. The folder structure consists of a root node named **Home**, and reserved folders that support the optional **My Reports** feature. In a browser, the root node is the name of the report server virtual directory, for example, `https://myreportserver/reports`. For more information, see [Folders](#).

On a SharePoint site, use SharePoint folders in document libraries and content libraries to organize items.

Roles and Permissions

On a native report server, the report server system administrator manages access permissions, configures the report server to process report requests, maintain snapshot histories, and manage permissions for reports, data sources, datasets, and subscriptions. For example, a published report is secured through role assignments using

the Reporting Services role-based security model. For more information, see [Roles and Permissions \(Reporting Services\)](#).

On a SharePoint site, use the SharePoint site administrators page to manage access permissions on reports and report-related site content.

Schedules

On a native report server, you can schedule paginated reports, shared datasets, and subscriptions to retrieve data and deliver reports and dataset queries at specific times or during off-peak hours. Schedules can run once or on a continuous basis at intervals of hours, days, weeks, or months. For more information, see [Schedules](#).

Subscriptions and delivery

A subscription is a standing request to deliver a report at a specific time or in response to an event, and in an application file format that you specify in the subscription. Subscriptions provide an alternative to running a report on demand. On-demand reporting requires that you actively select the report each time you want to view the report. In contrast, subscriptions can be used to schedule and then automate the delivery of a report. You can deliver reports to an e-mail inbox or a file share. For more information, see [Subscriptions and Delivery \(Reporting Services\)](#).

Extensions

SQL Server Reporting Services provides an extensible architecture that you can use to customize report solutions. The report server supports custom authentication extensions, data processing extensions, report processing extensions, rendering extensions, and delivery extensions, and the extensions that are available to the users are configurable in the RSReportServer.config configuration file. For example, you can limit the export formats the report viewer is allowed to use. Delivery and report processing extensions are optional, but necessary if you want to support report distribution or custom controls. For more information, see [Reporting Services Extensions \(SSRS\)](#).

Report access

On-demand access allows users to select the reports from a report viewing tool. Depending on your report server configuration, you can use the web portal, a Microsoft SharePoint 2.0 Web part, a SharePoint library when Reporting Services is installed in SharePoint integrated mode, an embedded ReportViewer control, or a browser using URL access. For more information about on-demand access to reports, see [Finding, Viewing, and Managing Reports \(Report Builder and SSRS\)](#).

Subscriptions provide an alternative to running a report on demand. For more information, see [Subscriptions and Delivery \(Reporting Services\)](#).

For the list of tools to use to interact with the report server, see [Reporting Services Tools](#).

Reports and related item concepts

Reports and report definitions

RDL

A report definition is an XML file that conforms to an XML grammar called Report Definition Language (RDL). In Reporting Services, you create a report definition in a tool such as Report Builder or Report Designer. It includes elements that define data source connections, queries used to retrieve data, expressions, parameters, images, text boxes, tables, and any other design-time layout. For more information, see [Report Definition Language \(SSRS\)](#).

RSMOBILE

You create Reporting Services mobile reports (.rsmobile files) in SQL Server Mobile Report Publisher. They're optimized for mobile devices and connected to on-premises data, with an assortment of data visualizations. Read more about [Reporting Services mobile reports](#).

RDLC

The Visual Studio Report Designer produces client report definition (.rdlc) files in XML format for use with the ReportViewer control.

Report data connections and data sources

Reports use data connections to retrieve data for a report when a query runs or when the report is processed. In a report definition, a data connection is the same as a data source. You choose from a list of built-in data connection types to connect to a relational database, a multidimensional database, a Web service, or some other source of data. The following terms are used when describing data connections.

- **Data connection.** Also known as a *data source*. A data connection includes a name and connection properties that are dependent on the connection type. By design, a data connection does not include credentials. A data connection does not specify which data to retrieve from the external data source. To do that, you specify a query when you create a dataset.
- **Data source definition.** A file that contains the XML representation of a report data source. When a report is published, its data sources are saved on the report server or SharePoint site as data source definitions, independently from the report definition. For example, a report server administrator might update the connection string or credentials. On a native report server, the file type is .rds. On a SharePoint site, the file type is .rsds.
- **Connection string.** A connection string is a string version of the connection properties that are needed to connect to a data source. Connection properties differ based on data connection type.
- **Shared data source.** A data source that is available on a report server or SharePoint site to be used by multiple reports.

Shared data sources are useful when you have data sources that you use often. It is recommended that you use shared data sources as much as possible. They make reports and report access easier to manage, and help to keep reports and the data sources they access more secure. If you need a shared data source, ask your system administrator to create one for you.

In Report Builder, you cannot create a shared data source. You can browse to and select a shared data source from the report server.

In Report Designer, you cannot browse to a shared data source on the report server. You can create shared data sources as part of a project in Solution Explorer and choose whether to deploy them to a report server. You might choose to use them locally only because of differences in credentials required from your computer or from the report server.

- **Embedded data source.** Also known as a *report-specific data source*, an embedded data source is defined in a report and used only by that report.

An embedded data source is a data connection that is saved in the report definition. Embedded data source connection information can be used only by the report in which it is embedded.

- **Credentials.** Credentials are the authentication information that must be provided to allow you access to external data.

Credentials are used to create an embedded data source, to run a query, or to retrieve data during report processing. The owner of the data source determines the type of credentials that you must use to access the data. Credentials are managed independently from the data connection on a report server, a SharePoint site, or on a local computer in a report authoring environment. Depending on the type of data source, credentials can be saved to avoid prompting or set to prompt each user. The credentials that you need might differ depending on whether you are connecting to the data source from your computer or from the report server. For more information, see [Specify Credentials in Report Builder](#).

Report Datasets

In a report, a dataset represents report data that is returned from running a query on an external data source. The dataset depends on the data connection that contains information about the external data source. The data itself is not included in the report definition. The dataset contains a query command, a field collection, parameters, filters, and data options that include case sensitivity and collation. There are two types of datasets:

- **Shared datasets.** A shared dataset is published on a report server and can be used by multiple reports. A shared dataset must be based on a shared data source. A shared dataset can be cached and scheduled by creating a cache refresh plan.
- **Embedded datasets.** Embedded datasets are defined in and used by a single report.

For more information, see [Report Embedded Datasets and Shared Datasets \(Report Builder and SSRS\)](#).

Report parameters

Report parameters are a part of a report definition. You can add parameters to Reporting Services paginated and mobile reports to link related reports, to control the report appearance, to filter report data, or to narrow the scope of a report to specific users or locations. When a paginated report is published to a native report server or SharePoint site, report parameters are saved as a separate report server item. Parameters can be managed independently from the report definition. To create multiple sets of parameters for the same report, create *linked reports*.

Report items

A report item is an internal but basic concept in a Reporting Services paginated report definition. Properties of a report item apply to data regions, maps, text boxes, images, and other design elements that you add to a report. Understanding the properties of a report item can help you to design customized report content and appearance. For example, all report items have a Hidden property to control visibility.

Data regions and maps

A data region is a layout element that displays data from a single dataset in a Reporting Services paginated report. Data region types include tablix, chart, gauge, and indicator. Map is a special type of data region because it can display data from two datasets: one that contains spatial data and one that contains analytical data.

Use data regions to enable common data visualizations: numbers and text in a table, matrix, or list; graphical displays in a chart or gauge; and geographic displays against a map. Tables, matrices, and lists are all based on the tablix data region, which expands as needed to display all the data from the dataset. A tablix data region supports multiple row and column groups and both static and dynamic rows and columns. A chart displays multiple series and category groups in a variety of chart formats. A gauge displays a single value or an aggregated value for a dataset. A map displays spatial data as map elements that can vary in appearance based on aggregated data from a dataset.

- **Table.** A table is a data region that presents data row by row. Table columns are static: you determine the number of columns when you design your report. Table rows are dynamic: they expand downwards to accommodate the data. You can add groups to tables, which organize data by selected fields or expressions. For more information, see [Tables, Matrices, and Lists \(Report Builder and SSRS\)](#).
- **Matrix.** A matrix is also known as a crosstab. A matrix data region contains both dynamic columns and rows: they expand to accommodate the data. A matrix can have dynamic columns and rows and static columns and rows. Columns or rows can contain other columns or rows, and can be used to group data. For more information, see [Tables, Matrices, and Lists \(Report Builder and SSRS\)](#).
- **List.** A list is a data region that presents data arranged in a freeform fashion. You can arrange report items to create a form with text boxes, images, and other data regions placed anywhere within the list. For more information, see [Tables, Matrices, and Lists \(Report Builder and SSRS\)](#).
- **Chart.** A chart presents data graphically. Examples of charts include bar, pie, and line charts, but many more

styles are supported. For more information, see [Charts \(Report Builder and SSRS\)](#).

- **Gauge.** A gauge presents data as a range with an indicator pointing to a specific value within the range. Gauges are used to display key performance indicators (KPIs) and other metrics. Examples of gauges include linear and circular. For more information, see [Gauges \(Report Builder and SSRS\)](#).
- **Map.** A map enables you to present data against a geographical background. Map data can be spatial data from a SQL Server query, an ESRI shapefile, or Microsoft Bing map tiles. Spatial data consists of sets of coordinates that define polygons that represent shapes or areas, lines that represent routes or paths, and points represented by markers. You can associate aggregate data with map elements to automatically vary their color and size. For example, you can vary the marker type for a store based on sales amount or the color for a road based on speed limit. For more information, see [Maps \(Report Builder and SSRS\)](#).

You can also include values from datasets that are not linked to the data region in the following ways:

- Expressions that include calls to aggregate functions that specify a different dataset as the scope parameter, for example, `=Max(Fields!Sales.Value, "AnnualSales")`.
- Use the function **Lookup** to look up values from name/value pairs in a different dataset.

Report parts

A report part definition (.rsc) is a report server item that is an XML fragment of a report definition file. You create report parts by creating a report definition, and then selecting report items in the report to publish separately as report parts. Report parts include data regions, rectangles and their contained items, and images. You can save a report part with its dependent datasets and shared data source references so it can be reused in other reports. For more information, see [Report Parts in Report Designer \(SSRS\)](#).

Data alerts

A data alert is an item stored internally in an alerting database. A data alert definition includes which data to use from existing report data feeds, the conditions to be met, a schedule, and recipients for the alert. Data alerts are available only on reports published to a report server integrated with SharePoint Server. Data alerts are not available on a native report server installation. For more information, see [Reporting Services Data Alerts](#).

Types of Reporting Services paginated reports

In Reporting Services, the term *report* can apply to a specific type of report server item, a layout design, or a solution design. A single Reporting Services paginated report can have characteristics from more than one type; for example, a report can be, at the same time, a stand-alone report, a subreport referenced by a main report, the target of a drillthrough report in a different main report, and a linked report.

Drilldown reports

A drilldown report is a layout design that at first hides complexity and enables the user to toggle conditionally hidden report items to control how much detail data they want to see. Drilldown reports must retrieve all possible data that can be shown in the report. For reports that use large amounts of data, consider drillthrough reports instead. For more information, see [Drilldown Action \(Report Builder and SSRS\)](#).

Subreports

A subreport is a report item that you add to a report as a layout element. A subreport points to a different report and displays inside the body of a main report as an subreport instance. The subreport can use different data sources than the main report. Although a subreport can be repeated in data regions by using a parameter to filter data in each instance of the subreport, subreports are typically used with a main report as a briefing book or as a container for a collection of related reports. Each instance of a subreport switches context for report processing between the main report and the subreport. For reports that use many instances of subreports, consider using drillthrough reports instead. For more information, see [Subreports \(Report Builder and SSRS\)](#).

Main/detail reports and drillthrough reports

A main/detail report solution includes a main report that displays summary information with hyperlinks to one or more reports that display detailed information. The detail report runs only if a report reader clicks a link to it. The drillthrough report opens separately from the main report. A hyperlink can be defined on any report item that has an Action property, for example, text box, placeholder text, or chart series. For more information, see [Drillthrough Reports \(Report Builder and SSRS\)](#).

Linked reports

A linked report is a report server item that contains a pointer to the report definition but has its own set of report properties and settings. These include security, parameters, location, subscriptions, and schedules. Because parameters are managed independently on the server, republishing a main report that uses new parameter settings does not overwrite the existing parameters settings for either the main report or the linked report.

For more information, see [Create a Linked Report](#).

History reports

Report history is a collection of report snapshots. You can use report history to maintain a record of a report over time. Report history is not intended for reports that contain confidential or personal data. For this reason, report history can include only those reports that query a data source using a single set of credentials. Alternatively, you can create a history of a report by defining a schedule and subscription to deliver the report in an exported file format to a file share. For more information, see [Performance, Snapshots, Caching \(Reporting Services\)](#).

Cached reports

A cached report is a saved copy of a compiled report and report data. Cached reports are used to improve performance by reducing the number of processing requests to the report processor and by reducing the time that is required to retrieve large report datasets. They have a mandatory expiration period, usually in minutes. For more information about how to use cached reports, see [Caching Reports \(SSRS\)](#).

You can also cache query results for a shared dataset. For more information, see [Cache Shared Datasets \(SSRS\)](#).

Snapshots

A report snapshot is a report that contains layout information and query results that were retrieved at a specific point in time. Unlike on-demand reports, which get up-to-date query results when you view the report, the report server retrieves the compiled report and report data that were current for the report at the time the snapshot was created. Report snapshots are not saved in a particular rendering format. Instead, report snapshots are rendered in a final viewing format (such as HTML) only when a user or an application requests it. For more information, see [Performance, Snapshots, Caching \(Reporting Services\)](#).

Saved reports

A saved paginated report is a report definition (.rdl) file. A report definition can be saved locally or uploaded to a report server. If you upload a report definition instead of publishing it, no version validation or expression validation occurs. You will not see errors until the report runs. For more information, see [Save and Deploy Reporting Services Reports](#).

Published reports

A published report is a report server item that you publish to a report server from a Reporting Services tool. On a native report server, you publish the report to a folder that you have permissions to. On a SharePoint report server, you can publish the report to a document library that is enabled with report content type. To share the report that uses others, they must have been granted permission to view the report. For more information, see [Save and Deploy Reporting Services Reports](#).

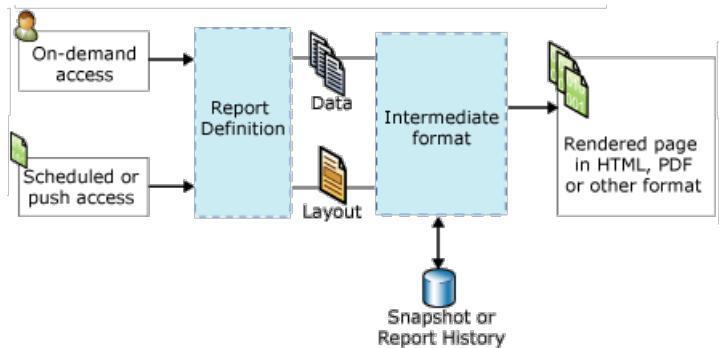
Upgraded reports

An upgraded report is a published report definition that is converted to a newer schema when a report server is upgraded from one version of Reporting Services to a later version. The original report definition is preserved. The report is upgraded in memory, compiled, and the compiled version is saved internally. For more information, see

[Upgrade Reporting Services Reports](#).

Stages of Reporting Services paginated reports

A report definition can be created, published or saved, compiled, processed, cached, rendered, viewed, exported, and saved as history. When you run a report, the report server processes a report in three steps: report processing, data processing, and rendering. Data and report processing are performed on a report definition; the results are in an internal intermediate format. Reports that are in intermediate format are subsequently rendered to a specific viewing format. The following diagram shows the stages and elements of report processing.



Report processing diagram

Report definition

The report definition file (.rdl) stored on a report server. For more information, see [Report Definition Language \(SSRS\)](#).

Compiled report and intermediate report format

The report that uses evaluated expressions, parameters and parameter properties evaluated.

Snapshot or Report History

A snapshot is the set of report data at a specific point in time plus the intermediate format that contains report layout information. For more information, see [Performance, Snapshots, Caching \(Reporting Services\)](#).

Processed report

A fully processed report that contains both data and layout information.

Rendered report

A fully processed report is sent to a report renderer to combine the data and layout on each page of the targeted rendering format. Rendering extensions are customizable and extensible. The default rendering format for a report is HTML 4.0. For more information, see [Page Layout and Rendering \(Report Builder and SSRS\)](#) and [Extensions \(SSRS\)](#).

Exported report

An exported report is a fully paged report saved in a specific file format. Export formats depend on installed rendering extensions and can be customized. By default, export formats include Excel, Word, XML, PDF, TIFF, and CSV. For more information, see [Export Reports \(Report Builder and SSRS\)](#).

See Also

[Reporting Services Features and Tasks \(SSRS\)](#)

[Technical Reference \(SSRS\)](#)

[Reporting Services \(SSRS\)](#)

Reporting Services Features and Tasks (SSRS)

10/1/2018 • 2 minutes to read • [Edit Online](#)

Reporting Services foundational content is organized by reports and report features, report server features, and Reporting Services product features.

In This Section

[Create mobile reports with SQL Server Mobile Report Publisher](#)

[Reporting Services Report Server](#)

[Reporting Services Reports \(SSRS\)](#)

[Report Data \(SSRS\)](#)

[Report Parameters \(Report Builder and Report Designer\)](#)

[Report Parts in Report Designer \(SSRS\)](#)

[Schedules](#)

[Subscriptions and Delivery \(Reporting Services\)](#)

[Reporting Services Data Alerts](#)

[Reporting Services Security and Protection](#)

[URL Access \(SSRS\)](#)

[Extensions \(SSRS\)](#)

[Reporting Services Tools](#)

See Also

[Reporting Services \(SSRS\)](#)

[What's New in Reporting Services \(SSRS\)](#)

Reporting Services Backward Compatibility

10/24/2018 • 2 minutes to read • [Edit Online](#)

Learn about changes in behavior of SQL Server Reporting Services. This covers features that are no longer available or are scheduled to be removed in a future release.

It also describes fundamental changes to the product that are known to break a custom application that includes Reporting Services functionality.

In This Section

TOPIC	DESCRIPTION
Discontinued functionality to SQL Server Reporting Services in SQL Server 2016	Describes features that existed in earlier versions of Reporting Services but that have been removed in later versions.
Deprecated features in SQL Server Reporting Services in SQL Server 2016	Describes features that exist this release of Reporting Services for backward compatibility, but which will be removed in a future version of SQL Server.
Breaking Changes in SQL Server Reporting Services in SQL Server 2016	Describes issues that you might encounter when you upgrade Reporting Services.
Behavior changes to SQL Server Reporting Services in SQL Server 2016	Describes features that have changed in Reporting Services.

See Also

[Backward Compatibility | Analysis Services](#)

Deprecated features in SQL Server Reporting Services in SQL Server 2016

10/24/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2016) SQL Server Reporting Services (2017) Power BI Report Server

For content related to previous versions of SQL Server Reporting Services (SSRS), see [SQL Server 2014 Reporting Services](#).

This topic describes the deprecated SQL Server Reporting Services features. The features are still available in the release in which they are deprecated; however the features are scheduled to be removed in a future release of SQL Server. Don't use deprecated features in new applications.

Features Not Supported in the Next Version of SQL Server Reporting Services

The following SQL Server Reporting Services features won't be supported in the next version of SQL Server. Don't use these features in new development work, and modify applications that currently use these features as soon as possible.

CATEGORY	DEPRECATED FEATURE
Report Server	HTML4.0 renderer. Use the HTML5 renderer.

Features Not Supported in Previous Versions of SQL Server Reporting Services

- [SQL Server 2014 Reporting Services Deprecated Features](#)
- [SQL Server 2012 Reporting Services Deprecated Features](#)

Next steps

[What's New in Reporting Services Backward Compatibility | Reporting Services](#)

[Behavior Changes to SQL Server Reporting Services in SQL Server 2016](#)

[Discontinued Functionality to SQL Server Reporting Services in SQL Server 2016](#)

More questions? Try asking the [Reporting Services forum](#)

Discontinued Functionality in SQL Server Reporting Services (SSRS)

12/17/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2016) SQL Server Reporting Services (2017) Power BI Report Server

For content related to previous versions of SQL Server Reporting Services (SSRS), see [SQL Server 2014 Reporting Services](#).

This topic describes SQL Server Reporting Services features that are no longer available in SQL Server 2016. It does not include announcements about discontinued support for specific versions of the operating system or Microsoft Internet Information Services (IIS). For more information about system prerequisites, see [Hardware and Software Requirements for Installing SQL Server 2016](#).

A *discontinued feature* is one that is no longer supported. It might also be physically removed from the product. The following features are discontinued.

FEATURE	REPLACEMENT OR WORKAROUND
Upload report models through the web portal	This can still be done through the SOAP API.
Manage report models through the web portal	This can still be done through the SOAP API.
Customize style sheets for HTML Viewer and Report Manager	You can still brand the web portal .

Next steps

[What's New in Reporting Services](#)

[Behavior Changes to SQL Server Reporting Services in SQL Server 2016](#)

[Deprecated features in SQL Server Reporting Services in SQL Server 2016](#)

More questions? Try asking the [Reporting Services forum](#)

Breaking changes in SQL Server Reporting Services in SQL Server 2016

12/18/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO:  SQL Server Reporting Services (2016)  SQL Server Reporting Services (2017)  Power BI Report Server

For content related to previous versions of SQL Server Reporting Services (SSRS), see [SQL Server 2014 Reporting Services](#).

This topic describes breaking changes in Reporting Services. These changes might break applications, scripts, or functionalities that are based on earlier versions of SQL Server. You might encounter these issues when you upgrade, or in custom scripts or reports.

Security Extensions

Custom security extensions need some modification to work with the new web portal. Security extensions need to use the `IAuthenticationExtension2` interface.

WMI Provider

The web portal application name changes from "ReportManager" to "ReportServerWebApp".

Next steps

[Behavior changes to SQL Server Reporting Services in SQL Server 2016](#)

[What's New in Reporting Services \(SSRS\)](#)

[Deprecated features in SQL Server Reporting Services in SQL Server 2016](#)

[Discontinued functionality to SQL Server Reporting Services in SQL Server 2016](#)

More questions? Try asking the [Reporting Services forum](#)

Behavior changes to SQL Server Reporting Services in SQL Server 2016

10/24/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO:  SQL Server Reporting Services (2016)  SQL Server Reporting Services (2017)  Power BI Report Server

For content related to previous versions of SQL Server Reporting Services (SSRS), see [SQL Server 2014 Reporting Services](#).

This topic describes behavior changes in Reporting Services. Behavior changes affect how features work or interact in SQL Server 2016 as compared to previous versions of SQL Server.

There are no behavior changes.

Next steps

[What's New in Reporting Services](#)

[Deprecated features in SQL Server Reporting Services in SQL Server 2016](#)

[Discontinued functionality to SQL Server Reporting Services in SQL Server 2016](#)

[Breaking changes in SQL Server Reporting Services in SQL Server 2016](#)

More questions? Try asking the [Reporting Services forum](#)

Analysis and reporting with Microsoft business intelligence (BI) tools

12/17/2018 • 2 minutes to read • [Edit Online](#)

Choosing the right business intelligence tool can be overwhelming. Learn about the different Microsoft offerings and find the one that best fits your needs.

The following table maps workloads for data analysis and reporting to the Microsoft BI tools that are best suited for those workloads. For more information about a product, click the product link in the table.

If you're looking for a brief overview of these tools to help you decide which tools are right for you, see [Introducing Microsoft Business Intelligence \(BI\) Tools](#).

WORKLOADS	USER			BI TOOLS		
		Excel	SharePoint	SharePoint Online	Power BI	SQL Server
Self-Service BI	Analyst/End User					
Easily discover, and access public and corporate data		Excel 2016			Azure Data Catalog	
Create powerful data models		Power Pivot			Power BI Desktop	
Perform self-service predictive analytics						Data Mining Add-ins for Excel
Visualize and explore data		Power View 3D Maps			Power BI Desktop	
Ask questions using natural language query					Q & A	

WORKLOADS	USER		BI TOOLS		
Access reports using mobile devices			HTML 5 (supports viewing <10-MB files)	HTML 5 (supports viewing <250 MB)	
				Power BI mobile app on iOS devices	
				Power BI mobile app on Android devices	
				Power BI mobile app for Windows 10	
Collaborate and share		SharePoint Sites	SharePoint Team Sites	Power BI Sites	
Corporate BI	IT Pro				
Create multi-dimensional/tabular corporate models					Analysis Services
Create ad-hoc data visualizations		Power View for SharePoint			
Create dashboards		SharePoint Dashboards PerformancePoint Services		Dashboards in Power BI	
Create operational reports					*Reporting Services
Create custom and embedded reports				Power BI Embedded	
Advanced Analytics	Data Scientist				
Perform self-service predictive analytics					Data Mining Add-ins for Excel

WORKLOADS	USER		BI TOOLS	
Use data mining algorithms				Data Mining in Analysis Services SQL Server R Services

*Reporting Services has a number of features that support delivering operational reports and custom reports, such as delivering modern, paginated reports.

Browser Support for Reporting Services and Power View

10/24/2018 • 3 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server 2016 Reporting Services and later Power BI Report Server

Learn about what browser versions are supported for managing and viewing SQL Server Reporting Services, the ReportViewer Controls and Power View.

NOTE

Reporting Services integration with SharePoint is no longer available after SQL Server 2016.

Browser requirements for the web portal

The following is the current list of browsers supported for the web portal.

Microsoft Windows

Windows 7, 8.1, 10; Windows Server 2008 R2, 2012, 2012 R2

- Microsoft Edge (+)
- Microsoft Internet Explorer 10 or 11
- Google Chrome (+)
- Mozilla Firefox (+)

Apple OS X

OS X 10.9-10.11

- Apple Safari (+)
- Google Chrome (+)
- Mozilla Firefox (+)

Apple iOS

iPhone and iPad with iOS 9

- Apple Safari (+)

Google Android

Phones and tablets with Android 4.4 (KitKat) or later

- Google Chrome (+)

(+) Latest publicly released version

Browser requirements for the ReportViewer web control (2015)

The following is the current list of browsers supported with the ReportViewer web control (2015). The report viewer supports viewing reports from Reporting Services web portal and SharePoint libraries.

Microsoft Windows

Windows 7, 8.1, 10; Windows Server 2008 R2, 2012, 2012 R2

- Microsoft Edge (+)
- Microsoft Internet Explorer 10 or 11
- Google Chrome (+)
- Mozilla Firefox (+)

Apple OS X

OS X 10.9-10.11

- Apple Safari (+)

(+) Latest publicly released version

If you are using a SharePoint product that is integrated with Reporting Services, see [Plan browser support in SharePoint 2016](#).

Authentication requirements

Browsers support specific authentication schemes that must be handled by the report server in order for the client request to succeed. The following table identifies the default authentication types supported by each browser running on a Windows operating system.

BROWSER TYPE	SUPPORTS	BROWSER DEFAULT	SERVER DEFAULT
Microsoft Edge (+)	Negotiate, Kerberos, NTLM, Basic	Negotiate	Yes. The default authentication settings work with Edge.
Microsoft Internet Explorer	Negotiate, Kerberos, NTLM, Basic	Negotiate	Yes. The default authentication settings work with Internet Explorer.
Google Chrome (+)	Negotiate, NTLM, Basic	Negotiate	Yes. The default authentication settings work with Chrome.
Mozilla Firefox (+)	NTLM, Basic	NTLM	Yes. The default authentication settings work with Firefox.
Apple Safari (+)	NTLM, Basic	Basic	Yes. The default authentication settings work with Safari.

(+) Latest publicly released version

Script requirements for viewing reports

To use the report viewer, configure your browser to run scripts.

If scripting is not enabled, you will see an error message similar to the following when you open a report:

- **Your browser does not support scripts or has been configured to not allow scripts to run. Click here to view this report without scripts.**

If you choose to view the report without script support, the report is rendered in HTML without report viewer capabilities such as the report toolbar and the document map.

NOTE

The report toolbar is part of the HTML Viewer component. By default the toolbar appears at the top of every report that is rendered in a browser window. The report viewer provides features include the ability to search the report for information, scroll to a specific page, and adjust the page size for viewing purposes. For more information about the report toolbar or HTML Viewer, see [HTML Viewer and the Report Toolbar](#).

Browser support for ReportViewer web server controls in Visual Studio

The ReportViewer Web server control is used to embed report functionality in an ASP.NET web application. The controls are included with Visual Studio and support different browsers and browser versions than the other components described in this topic. The type of browser used to view the application determines the kind of ReportViewer functionality that you can provide in your application. Use the table provided in this topic to determine which of the supported browsers are subject to report functionality restrictions and the supported platforms.

Use a browser that has script support enabled. If the browser cannot run scripts, you cannot view the report.

Microsoft Windows

Windows 7, 8.1, 10; Windows Server 2008 R2, 2012, 2012 R2

- Microsoft Edge (+)
- Microsoft Internet Explorer 10 or 11
- Google Chrome (+)
- Mozilla Firefox (+)

(+) Latest publicly released version

Power View browser support

Microsoft Windows

Windows 7, 8.1, 10; Windows Server 2008 R2, 2012, 2012 R2

- Microsoft Internet Explorer 10 or 11
- Mozilla Firefox (+)

Apple OS X

OS X 10.9-10.11

- Apple Safari (+)

(+) Latest publicly released version

For more information on the SharePoint 2016 browser support, see [Plan browser support in SharePoint 2013](#).

Next steps

[Finding and Viewing Reports in the web portal](#)

[Reporting Services Tools](#)

[Web portal \(SSRS Native Mode\)](#)

[HTML Viewer and the Report Toolbar](#)

[URL Access Parameter Reference](#)

More questions? Try asking the [Reporting Services forum](#)

Plan for report design and report deployment | Reporting Services

11/15/2018 • 4 minutes to read • [Edit Online](#)

SQL Server Reporting Services provides several approaches for authoring and deploying paginated reports. Learn how to plan a report authoring and report server environment that work together.

This topic is an overview of report definition support by Reporting Services components. A report definition is an XML file that is written in the Report Definition Language (RDL) or the Report Definition Language for Clients (RDLC). Each report definition conforms to a specific schema version that is listed at the beginning of the file.

RDL files are authored in Report Designer in SQL Server Data Tools - Business Intelligence projects, and in Report Builder. RDLC files are authored by using the ReportViewer controls that are included in Visual Studio.

RDL Schema Versions

The following table lists each available schema version and the abbreviation that is used throughout the rest of this topic:

ABBREVIATION	SCHEMA VERSION
2016 RDL	https://schemas.microsoft.com/sqlserver/reporting/2016/01/reportdefi
2010 RDL	https://schemas.microsoft.com/sqlserver/reporting/2010/01/reportdefi
2008 RDL	https://schemas.microsoft.com/sqlserver/reporting/2008/01/reportdefi
2005 RDL	https://schemas.microsoft.com/sqlserver/reporting/2005/01/reportdefi
2005 RDLC	
2000 RDL	https://schemas.microsoft.com/sqlserver/reporting/2003/10/reportdefi

For more information on RDL and RDLC schemas, see the following:

- [Microsoft SQL Server XML Schemas](#)
- [Report Definition Language Specifications](#)
- [Report Definition Language \(SSRS\)](#)

For more information about ReportViewer controls, see [ReportViewer Controls \(Visual Studio\)](#).

Report Server and RDL Schema Support

A report definition file can be deployed to a SQL Server 2016 Reporting Services (SSRS) report server in the following ways:

- **Report Designer:** Deploy a report from Report Designer in SQL Server Data Tools - Business Intelligence.
- **Report Builder:** Save a report to the report server from Report Builder.
- **Web Portal:** Upload a report to a native mode report server from the web portal.
- **SharePoint:** Upload a report to a SharePoint site that is configured with a SharePoint mode report server.
- **Programmatically:** Programmatically publish a report by using the SOAP API interfaces to a report server. For more information, see [Report Server Web Service](#).

The following table lists the supported rdl schema version by version of the report server.

REPORT SERVER VERSION	RDL SCHEMA VERSION
SQL Server 2016	2016 RDL 2010 RDL 2008 RDL 2005 RDL 2000 RDL
SQL Server 2014 (12.x)	2010 RDL
Or	2008 RDL
SQL Server 2012 (11.x)	2005 RDL
Or	2000 RDL
SQL Server 2008 R2	
SQL Server 2008	2008 RDL 2005 RDL 2000 RDL

When you upload a report definition to the report server or upgrade a report server that contains existing reports, the report server preserves the report definition in the original format. **On first use**, the report server upgrades the report in the report server database to a binary format that is preserved for subsequent views. The report definition (.rdl) itself is not upgraded.

You can extract from the report server a read-only copy of the report definition file (.rdl). On a native mode report server, browse to the web portal, select the report and click **Download**. In a SharePoint mode deployment, browse to the document library, select the report and click **Download a Copy**.

To upgrade the report definition, you must open the report in a report authoring environment, such as SQL Server Data Tools or Report Builder, and then save it.

For more information about report upgrades and the schema versions that are supported, see [Upgrade Reports](#).

Report Authoring and Deployment Support

Report authoring environments are Report Designer in SQL Server Data Tools - Business Intelligence projects, and Report Builder. Report authoring environments provide a variety of support for report upgrade, report design, report preview in local mode, report preview on the report server, and report deployment.

The following table summarizes support for authoring and deploying report definitions for different schema versions:

AUTHORING ENVIRONMENT	RDL VERSION AUTHORED	DEPLOY RDL VERSION	DEPLOY TO REPORT SERVER VERSIONS
SQL Server 2016 Report Builder	Authors 2016 RDL Will upgrade older RDL versions to 2016 RDL	2016 RDL	SQL Server 2016
Report Designer in SQL Server 2016 Data Tools - Business Intelligence for Microsoft Visual Studio 2015	Authors 2016 RDL Will upgrade older RDL versions to 2016 RDL	2016 RDL	SQL Server 2016

AUTHORING ENVIRONMENT	RDL VERSION AUTHORED	DEPLOY RDL VERSION	DEPLOY TO REPORT SERVER VERSIONS
Report Designer in SQL Server 2014 Data Tools - Business Intelligence for Microsoft Visual Studio 2012 Or Report Designer in SQL Server 2012 Data Tools - Business Intelligence for Microsoft Visual Studio 2012 Or Report Designer in SQL Server 2012 (11.x) Data Tools, included in SQL Server 2012 (11.x).	Authors 2010 RDL Will upgrade older RDL versions to 2010 RDL	2010 RDL	SQL Server 2014 (12.x) SQL Server 2012 (11.x) SQL Server 2008 R2
Report Designer in SQL Server 2008 R2 Business Intelligence Development Studio	Authors 2010 RDL Will upgrade older RDL versions to 2010 RDL	2010 RDL	SQL Server 2008 R2
Report Designer in SQL Server 2008 Business Intelligence Development Studio	Authors 2008 RDL Will upgrade older RDL versions to 2008 RDL	2008 RDL	SQL Server 2008

For more information on SQL Server Data Tools (SSDT), see the following:

- [Deployment and Version Support in SQL Server Data Tools \(SSRS\)](#)
- [SQL Server Data Tools for Visual Studio 2015](#).

ReportViewer Controls

A Visual Studio ReportViewer control can display an .rdlc report in local preview mode or in remote mode, the control can display an .rdl file hosted on a Reporting Services report server. The following table provides the list of RDL versions supported by the ReportViewer controls for local processing (.rdlc). Server side RDL support is summarized in the section [Report Server and RDL Schema Support](#).

REPORTVIEWER CONTROL IN PRODUCT	VERSION OF RDL FOR LOCAL PREVIEW
Visual Studio 2015 Or Visual Studio 2013	2008 RDL
Visual Studio 2012 Or Visual Studio 2010	
Visual Studio 2005 Or Visual Studio 2008	2005 RDL

For more information, see the following:

- [Converting RDLC Files to RDL Files](#)
- [ReportViewer Controls \(Visual Studio\)](#)
- [Adding and Configuring the ReportViewer Controls](#)

See Also

[Reports, Report Parts, and Report Definitions \(Report Builder and SSRS\)](#)

[Reporting Services Tools](#)

[Report Definition Language \(SSRS\)](#)

SQL Server Reporting Services features supported by its editions

1/9/2019 • 4 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server 2016 Reporting Services and later Power BI Report Server

This topic explains the SQL Server Reporting Services (SSRS) features supported by the different editions of SQL Server. SQL Server Evaluation edition is available for a 180-day trial period.

For the latest SQL Server release notes, see [SQL Server 2017 Release Notes](#). For the latest information on what's new, see [What's new in SQL Server Reporting Services \(SSRS\)](#).

Try SQL Server 2017



[Download SQL Server 2017 from the Evaluation Center](#)



[Spin up a Virtual Machine with SQL Server 2017 already installed](#)

For features supported by the Evaluation and Developer editions, see the SQL Server Enterprise edition column in the following table.

SQL Server Reporting Services

FEATURE NAME	ENTERPRISE	STANDARD	WEB	EXPRESS WITH ADVANCED SERVICES	DEVELOPER
Mobile reports and analytics	Yes				Yes
Supported catalog database SQL Server edition	Standard or higher	Standard or higher	Web	Express	Standard or higher
Supported data source SQL Server edition	All SQL Server editions	All SQL Server editions	Web	Express	All SQL Server editions
Report server	Yes	Yes	Yes	Yes	Yes
Report designer	Yes	Yes	Yes	Yes	Yes
Report designer web portal	Yes	Yes	Yes	Yes	Yes
Role-based security	Yes	Yes	Yes	Yes	Yes

Feature Name	Enterprise	Standard	Web	Express with Advanced Services	Developer
Export to Excel, PowerPoint, Word, PDF, and images	Yes	Yes	Yes	Yes	Yes
Enhanced gauges and charting	Yes	Yes	Yes	Yes	Yes
Pin report items to Power BI dashboards	Yes	Yes	Yes	Yes	Yes
Custom authentication	Yes	Yes	Yes		Yes
Report as data feeds	Yes	Yes	Yes	Yes	Yes
Model support	Yes	Yes	Yes		Yes
Create custom roles for role-based security	Yes	Yes			Yes
Model item security	Yes	Yes			Yes
Infinite click through	Yes	Yes			Yes
Shared-component library	Yes	Yes			Yes
Email and file share subscriptions and scheduling	Yes	Yes			Yes
Report history, execution snapshots, and caching	Yes	Yes			Yes
SharePoint integration	Yes	Yes			Yes
Remote and non-SQL data source support ¹	Yes	Yes			Yes

FEATURE NAME	ENTERPRISE	STANDARD	WEB	EXPRESS WITH ADVANCED SERVICES	DEVELOPER
Data source, delivery, and rendering and RDCE extensibility	Yes	Yes			Yes
Custom branding	Yes				Yes
Data-driven report subscription	Yes				Yes
Scale-out deployment (web farms)	Yes				Yes
Alerting ² (SSRS 2016)	Yes				Yes
Power view ² (SSRS 2016)	Yes				Yes
Comments ³	Yes	Yes	Yes	Yes	Yes

¹ For more information on supported data sources in SQL Server Reporting Services (SSRS), see [Data sources supported by Reporting Services \(SSRS\)](#).

² Requires SQL Server 2016 Reporting Services in SharePoint mode. For more information, see [Install SQL Server Reporting Services in SharePoint mode](#). Starting in SQL Server 2017 Reporting Services, integration with SharePoint is no longer available.

³ Only in Power BI Report Server and SQL Server 2017 Reporting Services and later.

NOTE

SQL Server Express with Tools and SQL Server Express don't support SQL Server Reporting Services.

Edition requirements for the report server database

When you create a report server database, not all editions of SQL Server SQL Server can be used to host the database. The following table shows you which editions of the Database Engine you can use for specific editions of SQL Server Reporting Services.

FOR THIS EDITION OF SQL SERVER REPORTING SERVICES,	USE THIS EDITION OF THE DATABASE ENGINE INSTANCE TO HOST THE DATABASE.
Enterprise	Enterprise or Standard editions (local or remote)
Standard	Enterprise or Standard editions (local or remote)
Web	Web edition (local only)

FOR THIS EDITION OF SQL SERVER REPORTING SERVICES,	USE THIS EDITION OF THE DATABASE ENGINE INSTANCE TO HOST THE DATABASE.
Express with Advanced Services	Express with Advanced Services (local only)
Evaluation	Evaluation

Business intelligence clients

The following software client applications are available on the Microsoft Download Center. They help you create business intelligence documents that run on a SQL Server instance. When you host these documents in a server environment, use an edition of SQL Server that supports that document type. The following table identifies which SQL Server edition contains the server features required to host the documents created in these client applications.

TOOL NAME	ENTERPRISE	STANDARD	WEB	EXPRESS WITH ADVANCED SERVICES	DEVELOPER
Report Builder, .rdl and .rds	Yes	Yes	Yes	Yes	Yes
SQL Server Mobile Report Publisher, .rsmobile	Yes				Yes
Power BI apps for mobile devices (iOS, Windows 10, and Android), .rsmobile	Yes				Yes

NOTE

- The preceding table identifies the SQL Server editions that are required to enable these client tools. However, these tools can access data hosted on any edition of SQL Server.
- SQL Server Mobile Report Publisher is the single point for creation of mobile reports. Connect to an SSRS server to access data sources and create reports. Then publish them to the SSRS server for others in the organization to access, either on the server or on mobile devices. You can also use SQL Server Mobile Report Publisher stand alone with local data sources.
- Whether you use SQL Server 2016 Reporting Services (SSRS) on-premises, Power BI in the cloud, or both as your report delivery solution, you only need one mobile app to access dashboards and mobile reports on mobile devices. The Power BI apps are available for download from the Windows, iOS, or Android app stores.

Next steps

- Read about [Features supported by the editions of SQL Server 2017](#).
- [Plan a SQL Server installation](#).
- More questions? Ask the [SQL Server Reporting Services forum](#).

Install SQL Server Reporting Services (2017 and later)

11/30/2018 • 4 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2017 and later) Power BI Report Server

SQL Server Reporting Services installation involves server components for storing report items, rendering reports, and processing of subscription and other report services.

To download SQL Server 2017 Reporting Services, go to the [Microsoft Download Center](#).

NOTE

Looking for Power BI Report Server? See [Install Power BI Report Server](#).

Before you begin

Before you install Reporting Services, review the [Hardware and software requirements for installing SQL Server](#).

Install your report server

Installing a report server is straightforward. There are only a few steps to install the files.

NOTE

You do not need a SQL Server Database Engine server available at the time of install. You will need one to configure Reporting Services after install.

1. Find the location of SQLServerReportingServices.exe and launch the installer.

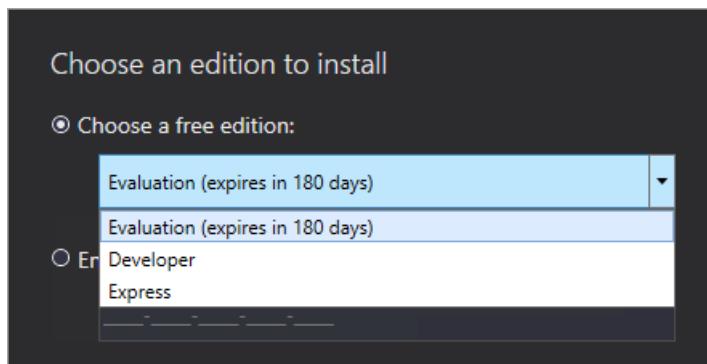
2. Select **Install Reporting Services**.



3. Choose an edition to install and then select **Next**.



For a free edition, choose either Evaluation or Developer from the drop down.



Otherwise, enter a product key. [Find the product key for SQL Server 2017 Reporting Services](#).

4. Read and agree to the license terms and conditions and then select **Next**.
5. You need to have a Database Engine available to store the report server database. Select **Next** to install the report server only.



6. Specify the install location for the report server. Select **Install** to continue.



NOTE

The default path is C:\Program Files\Microsoft SQL Server Reporting Services.

7. After a successful setup, select **Configure Report Server** to launch the Reporting Services Configuration Manager.



Configuration your report server

After you select **Configure Report Server** in the setup, you will be presented with **Report Server Configuration Manager**. For more information, see [Report Server Configuration Manager](#).

You need to [create a report server database](#) to complete the initial configuration of Reporting Services. A SQL Server Database server is required to complete this step.

Creating a database on a different server

If you are creating the report server database on a database server on a different machine, you need to change the service account for the report server to a credential that is recognized on the database server.

By default, the report server uses the virtual service account. If you try to create a database on a different server, you may receive the following error on the Applying connection rights step.

```
System.Data.SqlClient.SqlException (0x80131904): Windows NT user or group '(null)' not found. Check the name again.
```

To work around the error, you can change the service account to either Network Service or a domain account. Changing the service account to Network Service applies rights in the context of the machine account for the report server.

For more information, see [Configure the report server service account](#).

Windows Service

A windows service is created as part of the installation. It is displayed as **SQL Server Reporting Services**. The service name is **SQLServerReportingServices**.

Default URL reservations

URL reservations are composed of a prefix, host name, port, and virtual directory:

PART	DESCRIPTION
------	-------------

PART	DESCRIPTION
Prefix	The default prefix is HTTP. If you previously installed a Secure Sockets Layer (SSL) certificate, Setup tries to create URL reservations that use the HTTPS prefix.
Host name	The default host name is a strong wildcard (+). It specifies that the report server accepts any HTTP request on the designated port for any host name that resolves to the computer, including <code>https://<computername>/reportserver</code> , <code>https://localhost/reportserver</code> , or <code>https://<IPAddress>/reportserver</code> .
Port	The default port is 80. If you use any port other than port 80, you have to explicitly add it to the URL when you open web portal in a browser window.
Virtual directory	By default, virtual directories are created in the format of ReportServer for the Report Server Web service and Reports for the web portal. For the Report Server Web service, the default virtual directory is reportserver . For the web portal, the default virtual directory is reports .

An example of the complete URL string might be as follows:

- `https://+:80/reportserver`, provides access to the report server.
- `https://+:80/reports`, provides access to the web portal.

Firewall

If you are accessing the report server from a remote machine, you want to make sure you have configured any firewall rules if there is a firewall present.

You need to open up the TCP port that you have configured for your Web Service URL and Web Portal URL. By default, these are configured on TCP port 80.

Additional configuration

- To configure integration with the Power BI service so you can pin report items to a Power BI dashboard, see [Integrate with the Power BI service](#).
- To configure email for subscriptions processing, see [E-Mail settings](#) and [E-Mail delivery in a report server](#).
- To configure the web portal so you can access it on a remote computer to view and manage reports, see [Configure a firewall for report server access](#) and [Configure a report server for remote administration](#).

Related information

For information on how to install SQL Server Reporting Services native mode, see [Install Reporting Services native mode report server](#). For information on how to install SQL Server 2016 Reporting Services (and earlier) in SharePoint integration mode, see [Install the first Report Server in SharePoint mode](#).

Next steps

With your report server installed, begin to create reports and deploy those to your report server. For information on how to start with Report Builder, see [Install Report Builder](#).

To create reports using SQL Server Data Tools, [download SQL Server Data Tools](#).

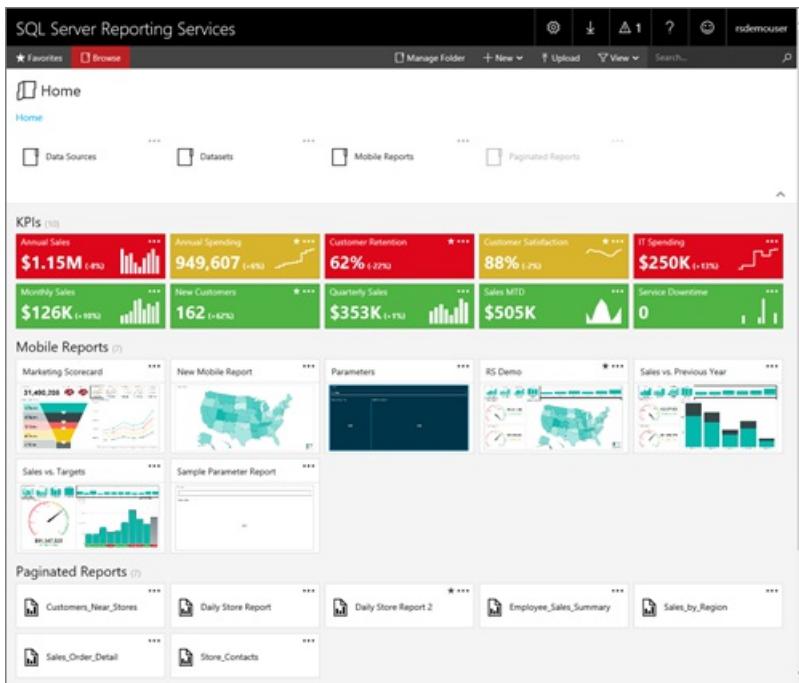
More questions? Try asking the [Reporting Services forum](#)

The web portal of a report server (SSRS Native Mode)

12/6/2018 • 4 minutes to read • [Edit Online](#)

APPLIES TO: ✓ SQL Server 2016 Reporting Services and later ✓ Power BI Report Server

The web portal of a Reporting Services report server is a web-based experience. In the portal, you can view reports, mobile reports, KPIs, and navigate through the elements in your report server instance. You can also use the web portal to administer a single report server instance.



What is the web portal

You can use the web portal to perform the following tasks:

- View, search, print, and subscribe to reports.
- Create, secure, and maintain the folder hierarchy to organize items on the server.
- Configure role-based security that determines access to items and operations.
- Configure report execution properties, report history, and report parameters.
- Create shared schedules and shared data sources to make schedules and data source connections more manageable.
- Create data-driven subscriptions that role out reports to a large recipient list.
- Create linked reports to reuse and repurpose an existing report in different ways.
- Download common tools such as Report Builder and Mobile Report Publisher.
- [Create KPIs](#).
- Send feedback or make feature requests.

You can use the web portal to browse the report server folders or search for specific reports. You can view a report, its general properties and past copies of the report that are captured in report history. Depending on your permissions, you might also be able to subscribe to reports for delivery to an e-mail inbox or a shared folder on the file system.

NOTE

For information on supported browsers and versions, see [Planning for Reporting Services Browser Support](#).

The web portal is used only for a report server that runs in native mode. It is not supported for a report server that you configure for SharePoint integrated mode.

Some web portal features are only available in specified editions of SQL Server. For more information, see [Reporting Services Features supported by the Editions of SQL Server](#).

On a new installation, only local administrators have sufficient permissions to work with content and settings. To grant permissions to other users, a local administrator must create role assignments that provide access to the report server. The application pages and tasks that a user can subsequently access will depend on the role assignments for that user. For more information, see [Grant User Access to a Report Server](#)

NOTE

If you are browsing to the web portal on the local machine that the server is running on, you may see a message indicating that you are not allowed to view this folder. This is due to Universal Access Control (UAC) and that you are not running the browser as an admin. You are not able to run Edge as an admin. You will need to use Internet Explorer. You can either browser to the server remotely, or launch Internet Explorer as admin and browser to the web portal. If you want to use the web portal remotely, you will need to give your account content manager rights on the folder.

Start and use the web portal

The web portal is a web application that you open by typing the web portal URL in the address bar of the browser window. When you start the web portal, the pages, links, and options that you see will vary based on the permissions you have on the report server. To perform a task, you must be assigned to a role that includes the task. A user who is assigned to a role that has full permissions has access to the complete set of application menus and pages available for managing a report server. A user assigned to a role that has permissions to view and run reports sees only the menus and pages that support those activities. Each user can have different role assignments for different report servers, or even for the various reports and folders that are stored on a single report server.

For more information about roles, see [Granting Permissions on a Native Mode Report Server](#).

Start the web portal

To start the web portal from a browser, follow these steps:

1. Open your web browser. For a list of supported web browsers, see [Planning for Reporting Services Browser Support](#).
2. In the address bar of the web browser, type the web portal URL.

By default, the URL is `https://[ComputerName]/reports`.

The report server might be configured to use a specific port. For example, `https://[ComputerName]:80/reports` or `https://[ComputerName]:8080/reports`.

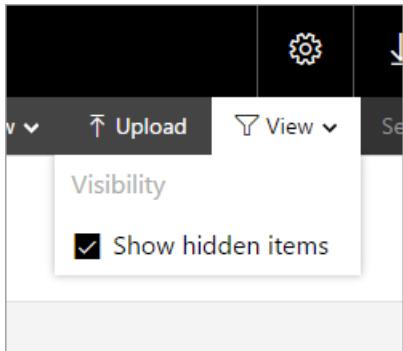
Grouping by categories

The web portal will group items into different categories. The available categories are the following.

- KPIs
- Mobile Reports

- Paginated Reports
- Power BI Desktop Reports
- Excel Workbooks
- Datasets
- Data Sources
- Resources

You can control what is displayed by selecting **View** in the upper right. If you select Show Hidden, those items will be displayed in a lighter color.



Power BI Desktop Reports and Excel Workbooks

You can upload, organize, and manage permissions for Power BI Desktop reports and Excel workbooks. They will be grouped together within the web portal.

Power BI Desktop Reports (1)

- Team Metrics

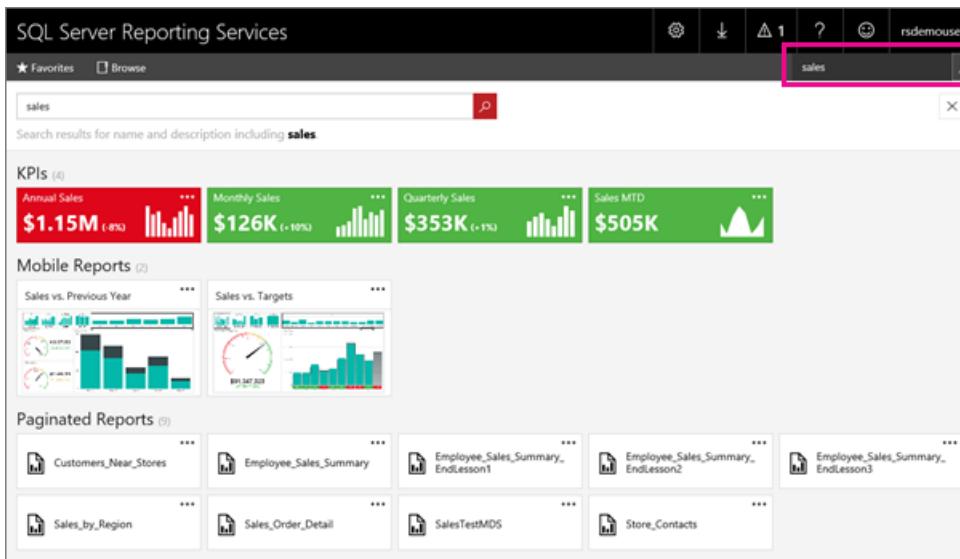
Excel Workbooks (3)

- Election Results
- Sales and Marketing Sample
- SQL Server Reporting Services Survey

The files are stored within Reporting Services, similar to other resource files. Selecting one of these items will download them locally to your desktop. You can save changes you've made by reuploading them to the report server.

Search for items

Enter a search term, and see everything you can access. The results are categorized into KPIs, reports, datasets, and other items. You can then interact with the results and add them to your favorites.



Web portal tasks

[Branding the web portal](#)

[Working with KPIs](#)

[Working with shared datasets](#)

See also

[Create mobile reports with SQL Server Mobile Report Publisher](#)

[Configure a URL \(SSRS Configuration Manager\)](#)

[Reporting Services Tools](#)

[Planning for Reporting Services Browser Support](#)

[Reporting Services Features supported by the Editions of SQL Server](#)

More questions? [Try the Reporting Services forum](#)

Branding the web portal

12/10/2018 • 3 minutes to read • [Edit Online](#)

APPLIES TO: ✓ SQL Server 2016 Reporting Services and later ✓ Power BI Report Server

You can alter the appearance of the web portal by branding it to your business. This is done through a brand package. The brand package is designed so you don't need deep cascading style sheet (CSS) knowledge to create it.

https://www.youtube.com/embed/m08kLuofwFA?list=PLv2BtOtLbIH3F--8WmK9QcLbx6dV_IVkL ## Creating the brand package A brand package for Reporting Services consists of three items and is packaged as a zip file. - color.json - metadata.xml - logo.png (optional) The files must have the names listed above. The zip file can be named however you like. ### metadata.xml The metadata.xml file allows you to set the name of the brand package, and has a reference entry for both your colors.json and logo.png files. To change the name of your brand package, change the **name** attribute of the **SystemResourcePackage** element. name="Multicolored example brand" You can optionally include a logo picture in your brand package. This item would be listed within the Contents element. Example without a logo file. Example with a logo file. ### Colors.json When the brand package is uploaded, the server extracts the appropriate name/value pairs from the colors.json file and merges them with the master LESS stylesheet, brand.less. This LESS file is then processed and the resulting CSS file is served to the client. All colors in the stylesheet follow the six-character hexadecimal representation of a color. The LESS stylesheet contains blocks that reference some predefined LESS variables like the following. /* primary buttons */ .btn-primary { color:@primaryButtonColor; background-color:@primaryButtonBg; } While this resembles CSS syntax, the color values, prefixed with the @symbol, are unique to LESS. They are variables whose values are set by the json file. For example, if the colors.json file had the following values. "primary":"#009900", "primaryContrast":"#ffffff" The processed output would look up the **@primaryButtonBg** LESS variable and see that it maps to the json property called **primary**, which in this example is #009900. It would therefore output the proper CSS. .btn-primary { color:#ffffff; background-color:#009900; } All of the primary buttons would be rendered dark green with white text. The colors.json file, for Reporting Services, has two main categories which items are grouped. - **Interface**: includes items that are specific to the Reporting Services web portal. - **Theme**: includes items that are specific to mobile reports that you create. The interface section is broken down into the following groupings. |Section|Description| --- --- |Primary|Button and hover colors.| |Secondary|Title bar, search bar, left hand menu (if displayed) and text color for those items| |Neutral Primary|Home and report area backgrounds.| |Neutral Secondary|Text box and folder options backgrounds, and the settings menu.| |Neutral Tertiary|Site settings backgrounds.| |Danger/Warning/Success messages|Colors for those messages.| |KPI|Controls the colors for a good (1), neutral (0), neutral (-1) and none.| The first time you connect to a server with the Mobile Report Publisher, that has a brand package deployed, the theme will be added to the available themes you can use in the upper right-hand menu of the app. ! [ssRSBrandingMobileReportPublisher] (./reporting-services/media/ssrsbrandingmobilereportpublisher.png) You can then use that theme for any mobile reports that you create, even if they aren't for the same server that you have the theme deployed on. ### Using a logo If you include a logo with your brand package, it will appear in the web portal in place of the name you set for the web portal in the Site Settings menu. The file you include for the logo must use the PNG file format. The file dimensions will be scaled once uploaded to the server. It should scale to around 290px x 60px. ## Applying the brand package to the web portal To add, download, or remove a brand package, you can do the following. 1. Select the **gear** in the upper right. 2. Select **Site Settings**. ! [ssRSGearMenu] (./reporting-services/media/ssrsgarmenu.png) 3. Select **Branding**. ! [ssRSBranding] (./reporting-services/media/ssrsbranding.png) **Currently installed brand package** will either display the name of the package that has been uploaded, or it will display None. **Upload brand package** will apply the package to the web portal. You will see it take effect immediately. You can also **Download** or **Remove** the package. Removing the package will reset the web portal to the default brand immediately. ## metadata.xml example ## Colors.json example { "name": "Multicolored example brand", "version": "1.0", "interface": { "primary": "#b31e1e", "primaryAlt": "#ca0806", "primaryAlt2": "#621013", "primaryAlt3": "#e40000", "primaryAlt4": "#e14e50",

```
"primaryContrast":"#fff", "secondary":"#042200", "secondaryAlt":"#0f4400", "secondaryAlt2":"#155500",
"secondaryAlt3":"#217700", "secondaryContrast":"#49e63c", "neutralPrimary":"#d8edff",
"neutralPrimaryAlt":"#c9e6ff", "neutralPrimaryAlt2":"#aedaff", "neutralPrimaryAlt3":"#88c8ff",
"neutralPrimaryContrast":"#0a2b4c", "neutralSecondary":"#e9d8eb", "neutralSecondaryAlt":"#d9badc",
"neutralSecondaryAlt2":"#b06cb5", "neutralSecondaryAlt3":"#a75bac", "neutralSecondaryContrast":"#250a26",
"neutralTertiary":"#f79220", "neutralTertiaryAlt":"#f8a54b", "neutralTertiaryAlt2":"#facc9b",
"neutralTertiaryAlt3":"#fce3c7", "neutralTertiaryContrast":"#391d00", "danger":"#ff0000", "success":"#00ff00",
"warning":"#ff8800", "info":"#00ff", "dangerContrast":"#fff", "successContrast":"#fff", "warningContrast":"#fff",
"infoContrast":"#fff", "kpiGood":"#4fb443", "kpiBad":"#de061a", "kpiNeutral":"#d9b42c", "kpiNone":"#333",
"kpiGoodContrast":"#fff", "kpiBadContrast":"#fff", "kpiNeutralContrast":"#fff", "kpiNoneContrast":"#fff"}, "theme":{  
"dataPoints": [ "#0072c6", "#f68c1f", "#269657", "#dd5900", "#5b3573", "#22bdef", "#b4009e", "#008274", "#fdc336",
"#ea3c00", "#00188f", "#9f9f9f" ], "good":"#85ba00", "bad":"#e90000", "neutral":"#edb327", "none":"#333",
"background":"#fff", "foreground":"#222", "mapBase":"#00aeef", "panelBackground":"#f6f6f6",
"panelForeground":"#222", "panelAccent":"#00aeef", "tableAccent":"#00aeef", "altBackground":"#f6f6f6",
"altForeground":"#000", "altMapBase":"#f68c1f", "altPanelBackground":"#235378", "altPanelForeground":"#fff",
"altPanelAccent":"#fdc336", "altTableAccent":"#fdc336" } }
```

Next steps

More questions? Try asking the [Reporting Services forum](#)

Work with shared datasets - web portal

11/28/2018 • 3 minutes to read • [Edit Online](#)

APPLIES TO: ✓ SQL Server 2016 Reporting Services and later ✓ Power BI Report Server

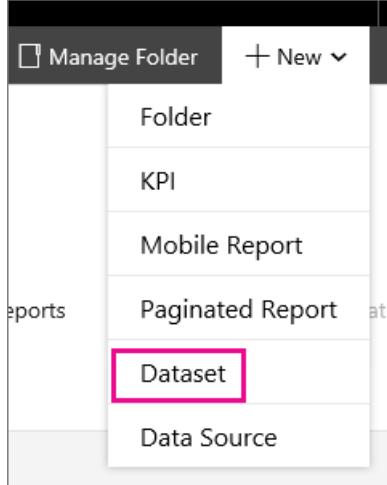
With a shared dataset, you can manage the settings for a dataset separately from reports and other catalog items that use it. Shared datasets can be used with paginated and mobile reports, along with KPIs.

You can view and manage the properties of a shared dataset within the web portal. The web portal can launch you into Report Builder to create or edit shared datasets.

Create a shared dataset

To create a new shared dataset, you can do the following.

1. Select new from the menu bar.
2. Select **Dataset**.



3. This will either launch Report Builder, or prompt you to download it.
4. On the **New Report or Dataset** dialog, select a data source connection to use for this dataset. You may need to browse to the location of the shared data source.
5. Select **Create**.
6. Build your dataset and then select the **save** icon in the upper left to save the dataset back to the report server.

Manage an existing shared dataset

To manage an existing shared dataset, you can do the following.

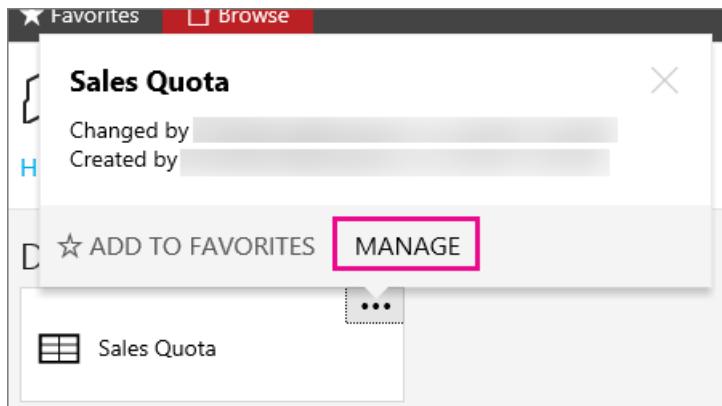
NOTE

If you don't see the shared dataset in the folder, make sure you are viewing datasets. You can select **View** from the menu bar in the upper right of the web portal. Make sure **Datasets** is checked.

1. Select the **ellipsis (...)** for the dataset you want to manage.

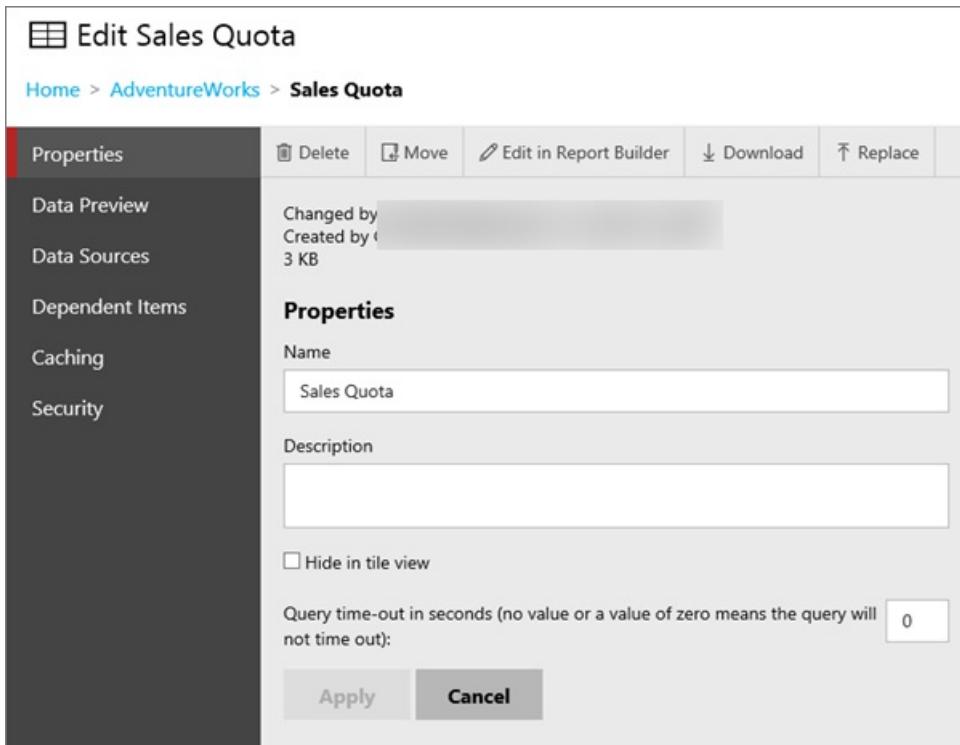


2. Select **Manage** which will take you to the edit screen.



Properties

On the properties screen, you can change the **name** and **description** for the dataset. You can also **Delete, Move, Edit in Report Builder, Download or Replace**.



Caching

You have options when it comes to caching data for a dataset. You will start off with a simple selection.

1. **Always run this report with the most recent data** will issue queries to the data source when requested.
2. **Cache copies of this report and use them when available** will place a temporary copy of the data in a cache for use with items that use this dataset. Caching usually improves performance because the data is returned from the cache instead of running the dataset query again.

Properties
Data Preview
Data Sources
Dependent Items
Caching
Security

Always run this report with the most recent data
 Cache copies of this report and use them when available

Apply

Selecting **Cache Copies of this report and use them when available** will present you with some more options.

Edit Sales Quota

Home > AdventureWorks > Sales Quota

Properties
Data Preview
Data Sources
Dependent Items
Caching
Security

Always run this report with the most recent data
 Cache copies of this report and use them when available

Cache Expiration

Cache expires after Minutes
 Cache expires on a schedule

Cache Refresh Plans

ⓘ You can manage cache refresh plans after you enable caching by clicking Apply on this page.

Description	Last Run	Status

Apply

Cache Expiration

You can control whether you want to expire the cache, for the shared dataset, after a certain amount of time, or if you would prefer to do that on a schedule. You can use a shared schedule.

Cache expires on a schedule
 Shared schedule Select a shared schedule
 Report-specific schedule [Edit schedule](#)

At 2:00 AM every day, starting 3/18/2016

NOTE

Setting an expiration does not refresh the cache. Without a cache refresh plan, the data will be refreshed on the next execution of the dataset.

Cache Refresh Plans

You can use Cache Refresh Plans to create schedules for preloading the cache with temporary copies of data for a shared dataset. A refresh plan includes a schedule and the option to specify or override values for parameters. You cannot override values for parameters that are marked read-only. You can create and use more than one refresh plan.

Default role assignments that enable you to add, delete, and change shared datasets for cache refresh plans are Content Manager, My Reports, and Publisher.

After you apply the cache option above, you can then define a cache refresh plan. To do that select the **Manage Refresh Plans** link that appears after you apply the cache settings. This will take you to the cache refresh plan page.

To create a new cache refresh plan, select **New Cache Refresh Plan**. You can then enter a name for the plan and specify a schedule. If the dataset has parameters defined, you will see those listed and be able to provide values unless they are marked as read-only.

Once you are done, you can select **Create Cache Refresh Plan**.

The screenshot shows the 'Edit Sales Quota' dialog box. At the top, it says 'Home > AdventureWorks > Sales Quota'. Below that, there's a 'Description:' field containing 'My Refresh Plan'. Under 'Refresh the cache on the following schedule:', there are two options: 'Shared schedule' (selected) with a dropdown menu 'Select a shared schedule' and 'Report-specific schedule' with a link 'Edit schedule' and a note 'At 2:00 AM every day, starting 3/21/2016'. A section titled 'Parameter Values:' contains a table with one row. The table has columns 'Parameter', 'Source of Value', and 'Value/Field'. The row shows 'Year:' with 'Enter value' in the dropdown and '2013' in the input field. At the bottom are 'Create Cache Refresh Plan' and 'Cancel' buttons.

NOTE

SQL Server Agent needs to be running to create a cache refresh plan.

You can then **Edit** or **Delete** plans that are listed. The **New From Existing** option is enabled when one, and only one, cache refresh plan is selected. This option will create a new refresh plan which is copied from the original plan. The cache refresh plan page opens pre-populated with details from the plan that was selected. You can then modify the refresh plan options and save the plan with a new description.

More questions? Try asking the [Reporting Services forum](#)

Working with paginated reports (web portal)

11/28/2018 • 10 minutes to read • [Edit Online](#)

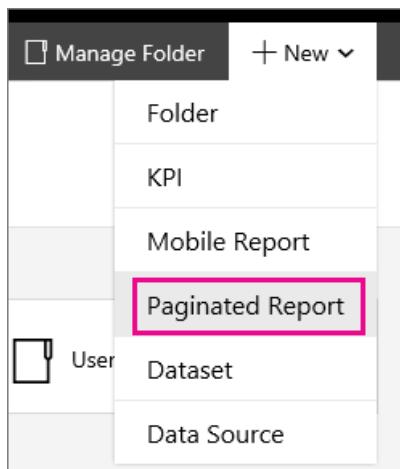
APPLIES TO: ✓ SQL Server 2016 Reporting Services and later ✓ Power BI Report Server

You can view and manage the properties of a paginated report within the web portal. The web portal can launch you into Report Builder to create or edit paginated reports.

Create a paginated report

To create a new shared dataset, you can do the following.

1. Select new from the menu bar.
2. Select **Paginated Report**.



3. This will either launch Report Builder, or prompt you to download it.
4. Build your report and then select the **save** icon in the upper left to save the paginated report back to the report server.

Manage an existing paginated report

To manage an existing paginated report, you can do the following.

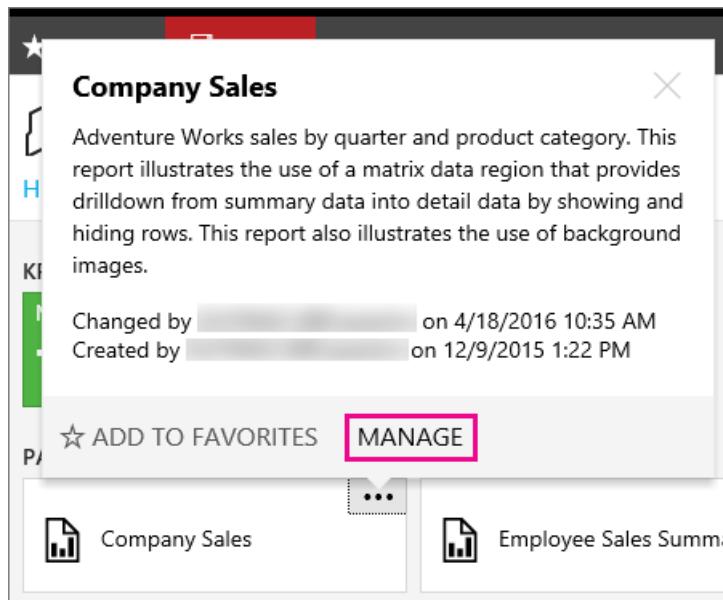
NOTE

If you don't see paginated reports in the folder, make sure you are viewing paginated reports. You can select **View** from the menu bar in the upper right of the web portal. Make sure **Paginated Reports** is checked.

1. Select the **ellipsis (...)** for the dataset you want to manage.



2. Select **Manage** which will take you to the edit screen.



Properties

On the properties screen, you can change the **name** and **description** for the paginated report. You can also **Delete, Move, Create Linked Report, Edit in Report Builder, Download or replace**.

The screenshot shows the 'Edit Company Sales' properties dialog. On the left is a sidebar with navigation links: Properties, Parameters, Data Sources, Shared Datasets, Subscriptions, Dependent Items, Caching, History Snapshots, and Security. The main area has tabs for 'Properties' (selected), 'Parameters', and 'Data Sources'. Under 'Properties', there are fields for 'Name' (set to 'Company Sales') and 'Description' (set to 'Adventure Works sales by quarter and product category. This report illustrates the use of a matrix data region that provides drilldown from'). There's also a checkbox for 'Hide in tile view'. Below these are sections for 'Advanced' settings, including 'Report Timeout' options: 'Use the system default setting' (selected), 'Allow the report to run for 1800 seconds before timing out', and 'Allow the report to run indefinitely (no timeout)'. At the bottom are 'Apply' and 'Cancel' buttons, and a note about creating linked reports.

Parameters

You can modify existing parameters of a paginated report. To add a new parameter, you must edit the report in Report Builder or SQL Server Data Tools.

Name	Data Type	Visibility	Prompt	Use Default	Default Value
Year	Integer	Visible	Year	<input checked="" type="checkbox"/>	2013

Data Source

You can point to a shared data source, or enter connection information for a custom data source.

AdventureWorks

Connect to:

A shared data source
/Data Sources/AdventureWorks ...

A custom data source

Save **Cancel**

The following options are used to specify a custom data source.

Type

Specify a data processing extension that is used to process data from the data source. For a list of built-in data extensions, see [Data Sources Supported by Reporting Services (SSRS)]. Additional data processing extensions may be available from third-party vendors.

Connection string

Specify the connection string that the report server uses to connect to the data source. The connection type determines the syntax you should use. For example, a connection string for the XML data processing extension is a URL to an XML document. In most cases, a typical connection string specifies the database server and a data file. The following example illustrates a connection string used to connect to a SQL Server database that is named MyData:

```
data source=(a SQL Server instance);initial catalog=MyData
```

A connection string can be configured as an expression so that you can specify the data source at run time. Data source expressions are defined in the report in Report Designer. Data source expressions cannot be defined, viewed, or modified in the web portal. However, you can replace a data source expression by clicking **Override Default** to type in a static connection string. If you want to switch back to the expression, click **Revert to Default**. The report server stores the original connection string in case you need to restore it. To use data source expressions, you must use the data source connection information that was originally published in the report. Shared data sources do not support the use of expressions in the connection string.

Credentials

You can specify the option that determines how credentials are obtained.

IMPORTANT

If credentials are provided in the connection string, the options and values provided in this section are ignored. Note that if you specify credentials on the connection string, the values are displayed in clear text to all users who view this page.

As the user viewing the report

Use the Windows credentials of the current user to access the data source. Select this option when the credentials that are used to access a data source are the same as those used to log on to the network domain. This option works best when Kerberos authentication is enabled for your domain, or when the data source is on the same computer as the report server. If Kerberos is not enabled, Windows credentials cannot be passed to another service, or remote machine. If additional computer connections are required, you will get an error instead of the data you expect.

A report server administrator can disable the use of Windows integrated security for accessing report data sources. If this value is grayed out, the feature is not available.

Do not use this option if you plan to schedule or subscribe to this report. Scheduled or unattended report processing requires credentials that can be obtained without user input or the security context of a current user. Only stored credentials provide this capability. For this reason, the report server prevents you from scheduling report or subscription processing if the report is configured for the Windows integrated security credential type. If you choose this option for a report that is already subscribed to or that has scheduled operations, the subscriptions and scheduled operations will stop.

Using these credentials

Store an encrypted user name and password in the report server database. Select this option to run a report unattended (for example, reports that are initiated by schedules or events instead of user action).

You can also choose the type of credential this would be. Either Windows authentication (Windows user name and password), or a specific database credential (Database user name and password) such as SQL authentication.

If the account is a windows credential, the account you specify must have log on locally permissions on the computer that hosts the data source used by the report.

Select **Log in using these credentials, but then try to impersonate the user viewing the report** to allow delegation of credentials, but only if a data source supports impersonation. For SQL Server databases, this option sets the SETUSER function. For Analysis Services, this uses EffectiveUserName.

By Prompting the user viewing the report for credentials

Each user must type in a user name and password to access the data source. You can define the prompt text that requests user credentials. For example, "Enter a user name and password to access the data source."

You can also choose the type of credential this would be. Either Windows authentication (Windows user name and password), or a specific database credential (Database user name and password) such as SQL authentication.

Without any credentials

This allows you to not provide any credentials for the data source. If a data source requires a user logon, choosing this option will have no effect. You should only choose this option if the data source connection does not require user credentials.

To use this option, you must have previously configured the unattended execution account for your report server. The unattended execution account is used to connect to external data sources when other courses of credentials are not available. If you specify this option and the account is not configured, the connection to the report data

source will fail and report processing will not occur. For more information about this account, see [Configure the Unattended Execution Account \(SSRS Configuration Manager\)](#).

Subscriptions

A Reporting Services subscription is a configuration that delivers a report at a specific time or in response to an event, and in a file format that you specify. For example, every Wednesday, save the MonthlySales.rdl report as a Microsoft Word document to a file share. Subscriptions can be used to schedule and automate the delivery of a report and with a specific set of report parameter values. For more information, see [Working with subscriptions](#).

The screenshot shows the 'Edit Company Sales' report page. On the left, a sidebar menu has 'Subscriptions' selected. The main content area displays a table with two columns: 'Edit' and 'Description'. There is one row in the table, but it is empty.

Dependent Items

Use the Dependent Items page to view a list of items that are referencing this report. The icon for each item type indicates what it is. You can then select the **ellipsis (...)** on each item to manage those items further.

Caching

You have options when it comes to caching data for a paginated report. You will start off with a simple selection.

1. **Always run this report with the most recent data** will issue queries to the data source every time you run the report. This results in an on-demand report that contains the most up-to-date data. A new instance of the report will be created each time the report is opened which will contain the results of a new query. With this approach, if ten users open the report at the same time, ten queries are sent to the data source for processing.
2. **Cache copies of this report and use them when available** will place a temporary copy of the data in a cache for later use. Caching usually improves performance because the data is returned from the cache instead of running the dataset query again. With this approach, if ten users open the report, only the first request results a query to the data source. The report is subsequently cached, and the remaining nine users view the cached report.
3. **Always run this report against pregenerated snapshots** will cache the report layout and data for a given time period. You can run a report as a report snapshot to prevent the report from being run at arbitrary times (for example, during a scheduled backup). The snapshot can be refreshed on a schedule.
[Learn more]

Properties
Parameters
Data Sources
Shared Datasets
Subscriptions
Dependent Items
Caching
History Snapshots

Always run this report with the most recent data
 Cache copies of this report and use them when available
 Always run this report against pregenerated snapshots
[Learn more](#)

Apply

Selecting **Cache Copies of this report and use them when available** will present you with some more options.

Home > AdventureWorks Sample Reports > Company Sales

Properties
Parameters
Data Sources
Shared Datasets
Subscriptions
Dependent Items
Caching
History Snapshots
Security

Always run this report with the most recent data
 Cache copies of this report and use them when available
 Always run this report against pregenerated snapshots
[Learn more](#)

Cache Expiration
 Cache expires after Minutes
 Cache expires on a schedule

Cache Refresh Plans

(i) You can manage cache refresh plans after you enable caching by clicking Apply on this page.

Description	Last Run	Status

Apply

For more information, see [Working with snapshots](#).

Cache Expiration

You can control whether you want to expire the cache, for the paginated report, after a certain amount of time, or if you would prefer to do that on a schedule. You can use a shared schedule.

NOTE

This does not refresh the cache.

Cache Refresh Plans

You can use Cache Refresh Plans to create schedules for preloading the cache with temporary copies of data for a paginated report. A refresh plan includes a schedule and the option to specify or override values for parameters. You cannot override values for parameters that are marked read-only. You can create and use more than one refresh plan.

Default role assignments that enable you to add, delete, and change paginated reports for cache refresh plans are Content Manager, My Reports, and Publisher.

After you apply the cache option above, you can then define a cache refresh plan. To do that select the **Manage Refresh Plans** link that appears after you apply the cache settings. This will take you to the cache refresh plan page.

To create a new cache refresh plan, select **New Cache Refresh Plan**. You can then enter a name for the plan and specify a schedule. If the dataset has parameters defined, you will see those listed and be able to provide values unless they are marked as read-only.

Once you are done, you can select **Create Cache Refresh Plan**.

The screenshot shows the 'Edit Company Sales' dialog box. At the top, there's a breadcrumb navigation: Home > AdventureWorks Sample Reports > Company Sales. Below the title, there's a 'Description:' field containing 'My Refresh Plan'. Under 'Refresh the cache on the following schedule:', the 'Report-specific schedule' option is selected, with a sub-option 'At 7:00 AM every day, starting 4/18/2016'. A 'Parameter Values:' section shows a table with one row: 'Year' (Parameter) with a dropdown 'Source of Value' set to 'Use default value' and a 'Value/Field' input field containing '2013'. At the bottom are two buttons: a red 'Create Cache Refresh Plan' button and a grey 'Cancel' button.

NOTE

SQL Server Agent needs to be running to create a cache refresh plan.

You can then **Edit** or **Delete** plans that are listed. The **New From Existing** option is enabled when one, and only one, cache refresh plan is selected. This option will create a new refresh plan which is copied from the original plan. The cache refresh plan page opens pre-populated with details from the plan that was selected. You can then modify the refresh plan options and save the plan with a new description.

History Snapshots

Use the History Snapshots page to view report snapshots that are generated and stored over time. Depending on options that are set, report history may contain only the more recent snapshots.

Report history is always viewed within the context of the report from which it originates. You cannot view the history of all reports on a report server in one place.

To generate a snapshot, the report must be able to run unattended (that is, it must use stored credentials; parameterized reports must contain default parameter values for all parameters). Snapshots can be generated manually or as a scheduled operation.

You can click a report history snapshot to view it. Snapshots that appear in report history are distinguished only by the date and time at which they were created. There is no visual indication to distinguish whether a snapshot was generated in response to a schedule or a manual operation.

Security

Use the Security properties page to view or modify the security settings that determine access to the report. This page is available for items that you have permission to secure.

Access to items is defined through role assignments that specify the tasks that a group or user can perform. A role assignment consists of one user or group name and one or more role definitions that specify a collection of tasks.

Edit Item Security

Select to change how security is defined for the current item.

Next steps

[Web portal](#)

[Work with Shared Datasets](#)

More questions? [Try asking the Reporting Services forum](#)

Working with snapshots (web portal)

11/28/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server 2016 Reporting Services and later Power BI Report Server

You can control if snapshots are created for a report by selecting the **ellipsis (...)** of a report, selecting **Manage** and selecting **Caching** or **History Snapshots**.

NOTE

The SQL Server Agent service needs to be started.

You can create a cache snapshot, to allow for faster loading of specific execution properties. You can also work with history snapshots to capture points in time.

Creating a cache snapshot

You can create a snapshot by doing the following.

1

2

3

1. On the **Caching** page, select **Always run this report against pregenerated snapshots** to enable the options for creating a snapshot.
2. Select **Create Cache snapshots on a schedule** if you want to schedule a recurring snapshot. You can then use a shared schedule, or define a custom schedule to refresh the snapshot.
3. Select **Create a cache snapshot when I click Apply on this page** if you want to create a cache snapshot right now. If you select only this option, the snapshot will not be refreshed.

Create, Modify, and Delete history snapshots

To work with history snapshots, manage a report and select **History Snapshots**.

Use the **History Snapshots** page to view report snapshots that are generated and stored over time. Depending on options that are set on the report server, the history may contain only the more recent snapshots.

Report history is always viewed within the context of the report from which it originates. You cannot view the history of all reports on a report server in one place.

To generate a history snapshot, the report must be able to run unattended (that is, it must use stored credentials; parameterized reports must contain default parameter values for all parameters). Report history can be generated manually or as a scheduled operation. History properties on the report determine the ways in which report history can be created.

The screenshot shows the 'Properties' tab of the 'Company Sales' report. The left sidebar has a red box around the 'History Snapshots' item, which is highlighted with a red circle labeled '1'. At the top, there are buttons for '+ New History Snapshot' (red circle '2'), 'Delete', 'Schedule and Settings', and 'Search...'. Below these are two rows of history snapshots. The first row shows a checkbox, 'View', 'Created' (Apr 26, 2016 8:23:34 AM), and 'Size (Total: 45 KB)'. The second row shows a checkbox, 'View History Snapshot', 'Created' (Apr 26, 2016 8:23:34 AM), and '45 KB'. A red circle labeled '3' points to the 'View History Snapshot' link in the second row.

1. To create a history snapshot, select **+ New History Snapshot**. This will process the report and add an entry to the list.
2. You can go into the settings to define schedules and retention policies.
3. You can select a history snapshot to view it. Snapshots that appear in report history are distinguished only by the date and time at which they were created. There is no visual indication to distinguish whether a snapshot was generated in response to a schedule or a manual operation.

Schedule and settings

Selecting **Schedule and Settings** will provide additional options to schedule and control retention of created snapshots.

The screenshot shows the 'Schedule' and 'Advanced' sections of the 'Schedule and Settings' dialog. In the 'Schedule' section, 'Create history snapshots on a schedule' is checked. Under 'Shared schedule', 'Select a shared schedule' is selected. Under 'Report-specific schedule', 'Edit schedule' is selected with the note 'At 7:00 AM every day, starting 4/26/2016'. In the 'Advanced' section, 'Allow people to create snapshots manually' is checked. The 'Retention' section has 'Use the system default settings' selected. Under 'Retention', 'Retain the 10 most recent history snapshots' is selected. There is also an option 'Save cache snapshots in report history as well' with a 'Learn more' link. At the bottom are 'Apply' and 'Cancel' buttons.

You can optionally create a schedule for the snapshots to get created. You can also prevent other people from

creating new snapshots. Unchecking **Allow people to create snapshots manually** will disable the **+ New Snapshot History button**.

You can also define how you want to retain snapshots.

Save cache snapshots in report history as well

Selecting this will copy a report snapshot that you generate based on report execution properties to report history. You can set report execution properties to run a report from a generated snapshot. By setting this report history property, you can keep a record of all reports snapshots that are generated over time by placing copies of them in report history.

Next steps

[Web portal](#)

[Working with paginated reports](#)

[Work with Shared Datasets](#)

More questions? Try asking the [Reporting Services forum](#)

Working with subscriptions (web portal)

11/28/2018 • 4 minutes to read • [Edit Online](#)

APPLIES TO: ✓ SQL Server 2016 Reporting Services and later ✓ Power BI Report Server

Use the Subscriptions page to list all of the subscriptions for the current report. If you have sufficient permission (as conveyed by the "Manage all subscriptions" task), you can view the subscriptions of all users. Otherwise, this page shows only the subscriptions that you own.

Before you can create a new subscription, you must verify that the report data source uses stored credentials. Use the Data Sources properties page to store credentials.

NOTE

The SQL Server Agent service needs to be started.

You can get to the Subscriptions page by selecting the **ellipsis (...)** of a report, selecting **Manage** and selecting **Subscriptions**.

From the Subscriptions page, you can create new subscriptions by selecting **+ New Subscription**. You can also edit existing subscriptions, or delete subscriptions that you have selected.

This page also provides the result status of subscription runs on the **Result** column. If an error occurred for a subscription, you will want to check the result column first to see what the message was.

Creating, or editing, a subscription

Use the New Subscription or Edit Subscription page to create a new subscription or modify an existing subscription to a report. The options on this page vary depending on your role assignment. Users with advanced permissions can work with additional options.

Subscriptions are supported for reports that can run unattended. At a minimum, the report must use stored or no credentials. If the report uses parameters, a default value must be specified. Subscriptions may become inactive if you change report execution settings or remove the default values used by parameter properties. For more information, see [Create and Manage Subscriptions for Native Mode Report Servers].

Type of Subscription

You can select between a **Standard subscription** and a **Data-driven subscription**.

Type of Subscription

Standard subscription

Generate and deliver one report

Data-driven subscription

Generate and deliver one report for each row in a dataset

[Learn more](#)

A data-driven subscription is one that queries a subscriber database for subscription information each time the subscription runs. Data-driven subscriptions use query results to determine the recipients of the subscription, delivery settings, and report parameter values. At run time, the report server runs a query to get values used for subscription settings.

To create a data-driven subscription, you must know how to write a query or command that gets the data for the subscription. You must also have a data store that contains the subscriber data (for example, subscriber names and e-mail addresses) to use for the subscription.

This option is available to users with advanced permissions. If you are using default security, data-driven subscriptions cannot be used for reports located in a My Reports folder.

Destination

Select the delivery extension to use to distribute the report.

The availability of a delivery extension depends on whether it is installed and configured on the report server. Report Server E-mail is the default delivery extension, but it must be configured before you can use it. File Share delivery does not require configuration, but you must define a shared folder before you can use it.

Destination

Deliver the report to:

Windows File Share ▾

Delivery Options (Windows File Share)

File Name:

Add a file extension when the file is created

Path:

Render Format:

Credentials used to access the file share:
 Use file share account
 Use the following Windows user credentials

User Name:

Password:

Overwrite options: Overwrite an existing file with a newer version
 Do not overwrite the file if a previous version exists
 Increment file names as newer versions are added

Depending on the delivery extension you select, the following settings appear:

- E-mail subscriptions provide fields that are familiar to e-mail users (for example, To, Subject, and Priority fields). Specify **Include Report** to embed or attach the report, or **Include Link** to include a URL to the report. Specify **Render Format** to choose a presentation format for the attached or embedded report.
- File share subscriptions provide fields that allow you to specify a target location. You can deliver any report to a file share. However, reports that support interactive features (including matrix reports that support drill-down to supporting rows and columns) are rendered as static files. You cannot view drill-down rows and columns in a static file. The file share name must be specified in Uniform Naming Convention (UNC) format (for example, \mycomputer\public\myreportfiles). Do not include a trailing backslash in the path name. The report file will be delivered in a file format that is based on the render format (for example, if you

choose Excel, the report is delivered as an .xlsx file).

Data-Driven subscription Dataset

For a data-driven subscription, you will need to define the dataset used for the subscription. Select **Edit Dataset** to supply that information.

Dataset

ⓘ You need to create a dataset for this data-driven subscription.

Edit Dataset

You need to first provide a **data source** to use for the query. This can be either a shared data source, or you can supply a custom data source.

You will need to then supply a **query** that will list the different options needed for the subscription to run. The screen will provide the fields that need to be returned. These fields will vary depending on your delivery method and the parameters of the report.

For best result, run the query in SQL Server Management Studio first, before using it in the data-driven subscription. You can then examine the results to verify that it contains the information you require. Important points to recognize about the query results are:

- Columns in the result set determine the values that you can specify for delivery options and report parameters. For example, if you are creating a data-driven subscription for e-mail delivery, you should have a column of e-mail addresses.
- Rows in the result set determine the number of report deliveries that are generated. If you have 10,000 rows, the report server will generate 10,000 notifications and deliveries.

Query

Enter a query to get the report parameter values and delivery options from your data.

Delivery Options: **File name, Path, Render Format, Write mode, File Extension, User name, Password, Use file share account**

Report Parameters: **Year**

Run this query for seconds before timing out

Validate query

You can then validate the query. You can also define a **query timeout**.

After the query has been created, you can then assign values to the required fields. You can either enter your manual data, or select a field from the dataset you created.

[Web portal](#)

[Working with paginated reports](#)

[Work with Shared Datasets](#)

More questions? [Try asking the Reporting Services forum](#)

Working with KPIs in Reporting Services

12/10/2018 • 3 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server 2016 Reporting Services and later Power BI Report Server

A Key Performance Indicator (KPI) is a visual cue that communicates the amount of progress made toward a goal. Key Performance Indicators are valuable for teams, managers, and businesses to evaluate quickly the progress made against measurable goals.

By using KPIs in SQL Server Reporting Services, you can easily visualize answers to the following questions:

- What am I ahead or behind on?
- How far ahead or behind am I?
- What is the minimum I have completed?

Creating a Dataset

A KPI will only use the first row of data from a shared dataset. Make sure that the data you want to use is located on that first row. To create a shared dataset, you can use either Report Builder or SQL Server Data Tools.

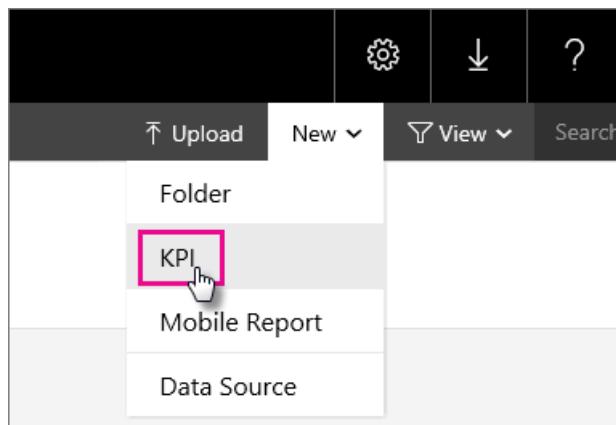
Note: The Dataset does not need to be in the same folder as the KPI.

Placement of KPIs

KPIs can be created in any folder in your report server. Before you create a KPI, you will want to think about where is the right location to place it in. You will want to place it in a folder that will be visible to the users, at the same time being relevant to other reports, and KPIs, around it.

Adding a KPI

After you have determined the location of your KPI, go to that folder and select **New > KPI** from the top menu.



This will present you with the **New KPI** screen.

You can either assign static values, or use data from a shared dataset. When you create a new KPI, it will be populated with a random set of manual data.

FIELD	DESCRIPTION
Value format	Used to change the format of the value being displayed.
Value	The value to display for the KPI.
Goal	Used as a comparison to a numeric value and shown as a percent difference.
Status	Numerical value used to determine the KPI Tile color. Valid values are 1 (green), 0 (amber) and -1 (red).
Trend set	Comma-separated numeric values used for chart visualization. This can also be set to a column of a dataset with values that represent the trend.

Warning: While you can use the word value for the **Status** field at design time, you should use the number value if refreshing a dataset. If you refresh a dataset with the word value, instead of the number, it could corrupt the KPIs on your server.

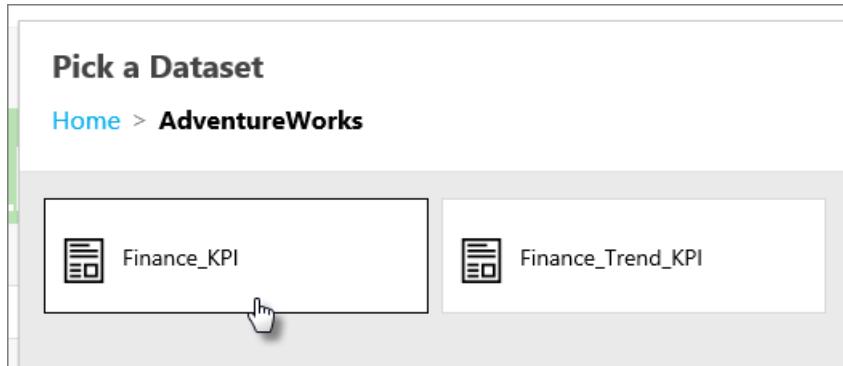
Note: The **Value**, **Goal** and **Status** fields can only choose a value from the first row of a dataset's result. The **Trend set** field, however, can choose which column reflects the trend.

To use data from a shared dataset, you can do the following.

1. Change the fields drop down box from **Set manually**, or **Not set**, to **Dataset field**.

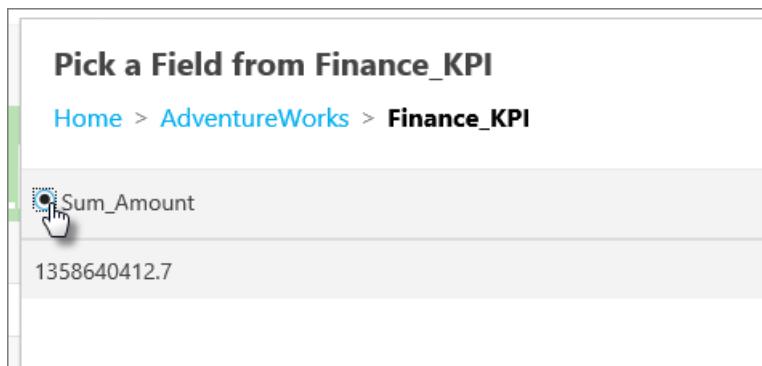
Value	?	Pick dataset field
Dataset field	Not set	...
Set manually		
Goal	?	

2. Select the **ellipsis (...)** in the data box. This will bring up the **Pick a Dataset** screen.



3. Select the dataset that has the data you want to display.

4. Choose the field you want to use. Select **OK**.



5. Change **Value format** to match the format of your value. In this example, the value is a currency.

Preview	Value format
Finance \$1,358,64...	Currency
KPI name	Value ? Pick dataset field
Finance	Dataset field <input type="button" value="▼"/> Finance_KPI.Sum_Amount ...
Description	Goal ?
	Not set <input type="button" value="▼"/>

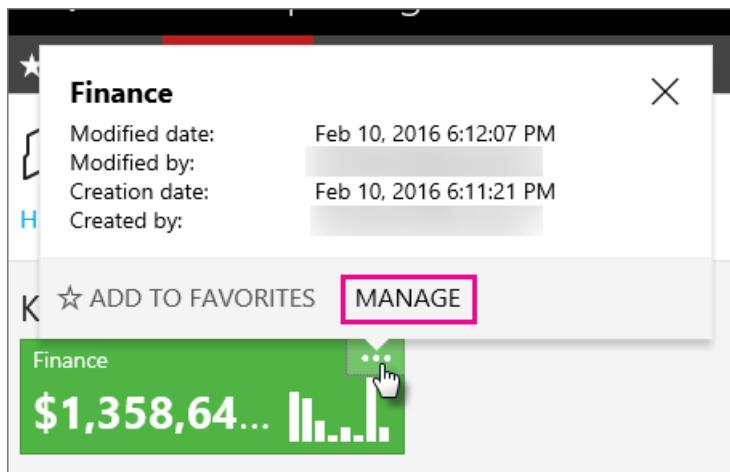
6. Select **Apply**.

The screenshot shows the AdventureWorks dashboard. At the top left is the logo and name "AdventureWorks". Below it is a breadcrumb navigation: "Home > AdventureWorks". Under "KPIs" (1 items), there is a green card for "Finance" with the value "\$1,358,64..." and a bar chart icon. To the right of the card is an ellipsis (...). Under "Datasets" (2 items), there are two cards: "Finance_KPI" and "Finance_Trend_KPI", each with a dataset icon and an ellipsis (...).

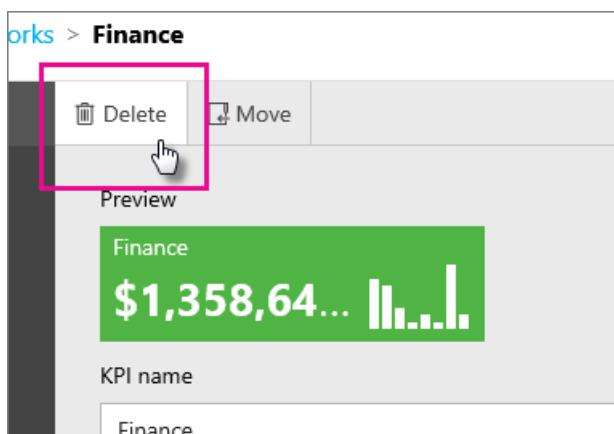
Removing a KPI

To remove a KPI, you can do the following.

1. Select the **ellipsis (...)** of the KPI you want to remove. Select **Manage**.



2. Select **Delete**. Select **Delete** again on the confirmation dialog.



Refreshing a KPI

To refresh the KPI, you will need to configure a caching for the shared dataset. For more information regarding cache refresh plans, see [Work with Shared Datasets](#).

Next steps

[Web portal](#)

[Work with Shared Datasets](#)

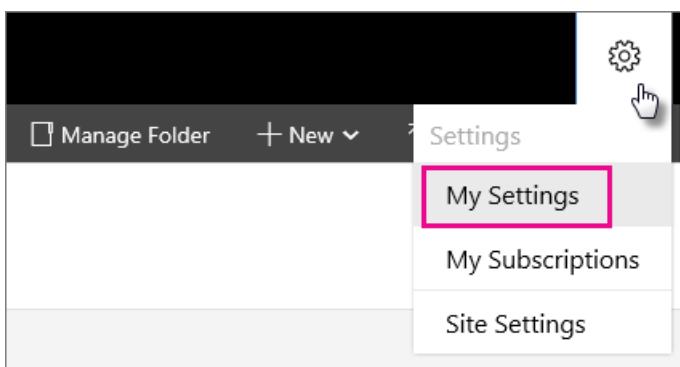
More questions? Try asking the [Reporting Services forum](#)

My Settings for Power BI Integration (web portal)

10/24/2018 • 2 minutes to read • [Edit Online](#)

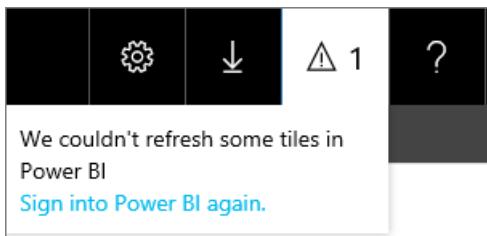
APPLIES TO: SQL Server 2016 Reporting Services and later Power BI Report Server

The **My Settings** page in the Reporting Services web portal is used by individual users to manage their sign-in with Power BI. When you go through the steps to pin a report item, you will automatically be prompted to sign in. However, you can use the **My Settings** page if you need to manually sign in or if you need to sign out. If the **My Settings** menu option is not visible, the report server has not been integrated with Power BI. For more information, see [Power BI Report Server Integration \(Configuration Manager\)](#).

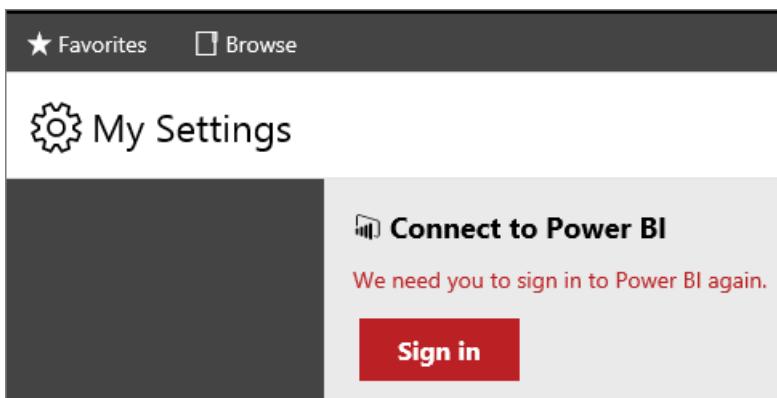


Why Sign-in

When you sign in, you establish a relationship between your Reporting Services user account and your Power BI account. The sign-in creates a security token that is good for 90 days. If the token expires, and you have items pinned to Power BI, you will see a notification.



Tiles within Power BI dashboards will not refresh until you sign in again through **MySettings**.



Once you sign in, a new security token will be created. Your dashboard tiles will begin updating on their previously configured schedules.

Next steps

[Power BI Report Server Integration](#)

[Pin Reporting Services items to Power BI Dashboards](#)

[Dashboards in Power BI](#)

[Web portal](#)

More questions? [Try asking the Reporting Services forum](#)

Reporting Services Report Server (Native Mode)

10/1/2018 • 6 minutes to read • [Edit Online](#)

A report server configured for native mode runs as an application server that provides all processing and management capability exclusively through Reporting Services components.

You can use either SQL Server Management Studio or Report Manager to manage Reporting Services reports. Use the Reporting Services configuration manager to manage a report server in native mode.

If the report server is configured for SharePoint mode, you must use the content management pages on the SharePoint site to manage reports, shared data sources, and other report server items.

This topic contains the following information:

- [Summary of Native Mode](#)
- [Managing content](#)
- [Securing and Managing a Resource](#)
- [Referencing an Image Resource from a Report](#)

Summary of Native Mode

A Reporting Services native mode installation consists of several server-side features that you need to manage and maintain. The server features include the following:

- The Report Server Web service, which runs within the Report Server service.
- The background processing applications, which handle scheduled operations and report delivery.
- The report server database.

To fully administer a Reporting Services installation, you must have the following permissions:

- Membership in the local Administrator group on the report server computer. If your installation includes server features that run on remote computers, you must have administrator permissions on those computers if you want to manage those servers over a remote connection.
- Database administrator permissions for the SQL Server instance that hosts the database.
- If you are installing Reporting Services on a domain controller, you must be a domain administrator.

Managing content

In Reporting Services, content management refers to the management of reports, models, folders, resources, and shared data sources. All these items can be managed independently of each other through properties and security settings. Any item can be moved to a different location in the report server folder namespace. To manage items effectively, you need to know which tasks a content manager performs.

NOTE

Content management is different from report server administration. For more information about how to manage the environment in which a report server runs, see [Configuration and Administration of a Report Server \(Reporting Services SharePoint Mode\)](#).

Content management includes the following tasks:

- Securing the report server site and items by applying the role-based security provided with Reporting Services.
- Structuring the report server folder hierarchy by adding, modifying, and deleting folders.
- Setting defaults and properties that apply to items managed by the report server. For example, you can set baseline maximum values that determine report history storage policies.
- Creating shared data source items that can be used in place of report-specific data source connections. A publisher or content manager can select a data source that is different from the one originally defined for a report; for example, to replace a reference to a test database with a reference to a production database.
- Creating shared schedules that can be used in place of report-specific and subscription-specific schedules, making it easier to maintain schedule information over time.
- Creating data-driven subscriptions that generate recipient lists by retrieving data from a data store.
- Balancing report-processing demands that are placed on the server by scheduling report processing and specifying which ones can be run on demand and which ones are loaded from cache.

Permission to perform management tasks are provided through two predefined roles: **System Administrator** and **Content Manager**. Effective management of report server content requires that you are assigned to both roles.

For more information about these predefined roles, see [Roles and Permissions \(Reporting Services\)](#).

Tools for managing report server content include Management Studio or Report Manager. Management Studio allows you to set defaults and enable features. Report Manager is used to grant user access to report server items and operations, view and use reports and other content types, and view and use all shared items and report distribution features.

Securing and Managing a Resource

A resource is a managed item that is stored on a report server, but is not processed by a report server. Typically, a resource provides external content to report users. Examples include an image in a jpg file or an HTML file that describes the business rules used in a report. The JPG or HTML file is stored on the report server, but the report server passes the file directly to the browser rather than processing it first.

To add a resource to a report server, you upload or publish a file:

OPERATION	FILE TYPE
Upload	All files are uploaded as resources except report definition (.rdl) and report model (.smld) files. To upload a resource, you must use Report Manager if the report server runs in native mode or an application page on a SharePoint site if the server runs in SharePoint integrated mode. For more information, see Upload a File or Report (Report Manager) or Upload Documents to a SharePoint Library (Reporting Services in SharePoint Mode) .

OPERATION	FILE TYPE
Publish	All files in a project are uploaded as resources except for .rdl, .smld, and .rds data source files. To publish a resource, add an existing item to a project in Report Designer and then publish the project to a report server.

All resources originate as files on a file system, which are subsequently uploaded to a report server. Except for the 4 megabyte default file size limitations imposed by ASP.NET, there are no restrictions on the kinds of files you can upload. However, when published to a report server as a resource, file types that have equivalent MIME types are more optimal than others. For example, resources that are based on HTML and JPG files will open in a browser window when the user clicks the resource, rendering the HTML as a Web page and the JPG as an image that the user can see. In contrast, resources that do not have equivalent MIME types, such as desktop application files, for example, may not be rendered in the browser window.

Whether a resource can be viewed by report users depends on the viewing capabilities of the browser. Because resources are not processed by the report server, the browser must provide the viewing capability to render a specific MIME type. If the browser cannot render the content, users who view the resource see only the general properties of the resource.

Resources exist alongside reports, shared data sources, shared schedules, and folders as named items in the report server folder hierarchy. You can search for, view, secure, and set properties on resources just as you would any item stored on a report server. To view or manage a resource, you must have the View resources or Manage resources tasks in your role assignment.

Referencing an Image Resource from a Report

Resources can contain an image that you reference in a report. If report requirements include the use of external images, consider the following advantages to storing the image as a resource:

- Centralized storage in the report server database. If you move the report server database and its contents to another computer, the external image stays with the report. You do not have to keep track of image files that are stored on disk on different computers.
- Secured through role assignments rather than file system security. The same permissions used to view a report can be applied to the resource. In contrast, if you store the image on disk, you must ensure that either the Anonymous user account or the unattended execution account have permission to access the file.

To use an image resource in a report, add the image file to the project and publish it along with the report. Once the image is published, you can update the image reference in the report so that it points to the resource on the report server, and then republish just the report to save your changes. You can now subsequently update the image independently of the report by republishing the resource. The report uses the most current version of the image available on the report server.

See Also

[Configure and Administer a Report Server \(SSRS Native Mode\)](#)

[Troubleshoot a Reporting Services Installation](#)

Reporting Services report server

11/27/2018 • 10 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server 2016 Reporting Services and later SharePoint Power BI Report Server

For content related to previous versions of SQL Server Reporting Services (SSRS), see [SQL Server 2014 Reporting Services](#).

Learn about the central piece of a SQL Server Reporting Services installation. It consists of a processing engine along with extensions to add functionality.

NOTE

Reporting Services integration with SharePoint is no longer available after SQL Server 2016.

A Reporting Services report server runs in one of two deployment modes; Native mode or SharePoint mode. See the [Feature Comparison of SharePoint and Native Mode](#) section for a comparison of features.

Installation: For information on Reporting Services installation, see [Install Reporting Services](#).

Overview of report server modes

Processing engines (processors) are the core of the report server. The processors support the integrity of the reporting system and cannot be modified or extended. Extensions are also processors, but they perform very specific functions. Reporting Services includes one or more default extensions for every type of supported extension. You can add custom extensions to a report server. Doing so allows you to extend a report server to support features that are not supported out of the box; examples of custom functionality might include support for single sign-on technologies, report output in application formats that are not already handled by the default rendering extensions, and report delivery to a printer or application.

A single report server instance is defined by the complete collection of processors and extensions that provide end-to-end processing, from the handling of the initial request to the presentation of a finished report. Through its subcomponents, the report server processes report requests and makes reports available for on-demand access or scheduled distribution.

Functionally, a report server enables report authoring experiences, report rendering, and report delivery experiences for a variety of data sources as well as extensible authentication and authorization schemes.

Additionally a report server contains report server databases that store published reports, shared data sources, shared datasets, report parts, shared schedules and subscriptions, report definition source files, model definitions, compiled reports, snapshots, parameters, and other resources. A report server also enables administration experiences for configuring the report server to process report requests, maintain snapshot histories, and manage permissions for reports, data sources, datasets, and subscriptions.

A Reporting Services report server supports two modes of deployment for report server instances:

- **Native mode:** including native mode with SharePoint web parts, where a report server runs as an application server that provides all processing and management capability exclusively through Reporting Services components. You configure a native mode report server with Reporting Services configuration manager and SQL Server Management Studio.
- **SharePoint mode:** where a report server is installed as part of a SharePoint server farm. Deploy and configure SharePoint mode by using PowerShell commands or SharePoint content management pages.

In SQL Server Reporting Services you cannot switch a report server from one mode to the other. If you want to change the type of report server that your environment uses, you must install the desired mode of report server and then copy or move the report items or report server database from the older versioned report server to the new report server. This process is typically referred to as a 'migration'. The steps needed to migrate depend on the mode you are migrating to and the version you are migrating from. For more information, see [Upgrade and Migrate Reporting Services](#)

Feature comparison of SharePoint and native mode

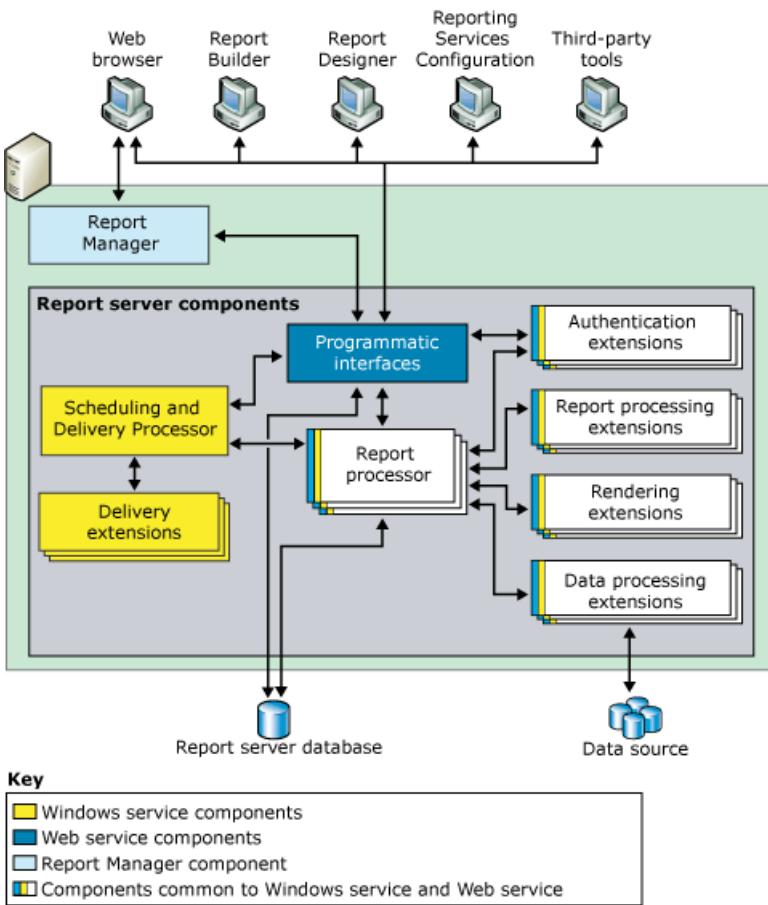
FEATURE OR COMPONENT	NATIVE MODE	SHAREPOINT MODE
URL addressing	Yes	<p>URL addressing is different in SharePoint integrated mode. SharePoint URLs are used to reference reports, report models, shared data sources, and resources. The report server folder hierarchy is not used. If you have custom applications that rely on URL access as supported on a native mode report server, that functionality will no longer work when the report server is configured for SharePoint integration.</p> <p>For more information on URL access, see URL Access Parameter Reference</p>
Custom security extensions	Yes	<p>Reporting Services custom security extensions cannot be deployed or used on the report server. The report server includes a special-purpose security extension that is used whenever you configure a report server to run in SharePoint integrated mode. This security extension is an internal component, and it is required for integrated operations.</p>
Configuration Manager	Yes	<p>** Important ** Configuration Manager cannot be used to manage a SharePoint mode report server. Instead, use SharePoint central administration.</p>
Report Manager	Yes	<p>Report Manager cannot be used to manage SharePoint mode. Use the SharePoint application pages. For more information, see Reporting Services SharePoint Service and Service Applications.</p>
Linked Reports	Yes	No.
My Reports	Yes	No
My Subscriptions and batching methods.	Yes	No
Data Alerts	No	Yes

FEATURE OR COMPONENT	NATIVE MODE	SHAREPOINT MODE
Power View	No	Yes Requires Silverlight in the client browser. For more information on browser requirements, see Browser Support for Reporting Services and Power View
.RDL reports	Yes	.RDL reports can run on Reporting Services report servers in native mode or in SharePoint mode.
.RDLX reports	No	Power View .RDLX reports can only run on Reporting Services report servers in SharePoint mode.
SharePoint user token credentials for the SharePoint list extension	No	Yes
AAM zones for internet facing deployments	No	Yes
SharePoint backup and recovery	No	Yes
ULS log support	No	Yes

Native mode

In native mode, a report server is a stand-alone application server that provides all viewing, management, processing, and delivery of reports and report models. This is the default mode for report server instances. You can install a native mode report server that is configured during setup or you can configure it for native mode operations once setup is complete.

The following diagram shows the three-tier architecture of a Reporting Services Native mode deployment. It shows the report server database and data sources in the data tier, the report server components in the middle tier, and the client applications and built-in or custom tools in the presentation tier. It shows the flow of requests and data among the server components and which components send and retrieve content from a data store.



The report server is implemented as a Microsoft Windows service, called the "Report Server service", that hosts a Web service, background processing, and other operations. In the Services console application, the service is listed as SQL Server Reporting Services (MSSQLSERVER).

Third-party developers can create additional extensions to replace or extend the processing capability of the report server. To learn more about the programmatic interfaces available to application developers, see the [Technical Reference](#).

Native mode with SharePoint web parts

Reporting Services provides two web parts that you can install and register on an instance of Windows SharePoint Services 2.0 or later, or SharePoint Portal Server 2003 or later. From a SharePoint site, you can use the web parts to find and view reports that are stored and processed on a report server that runs in native mode. These web parts were introduced in earlier releases of Reporting Services.

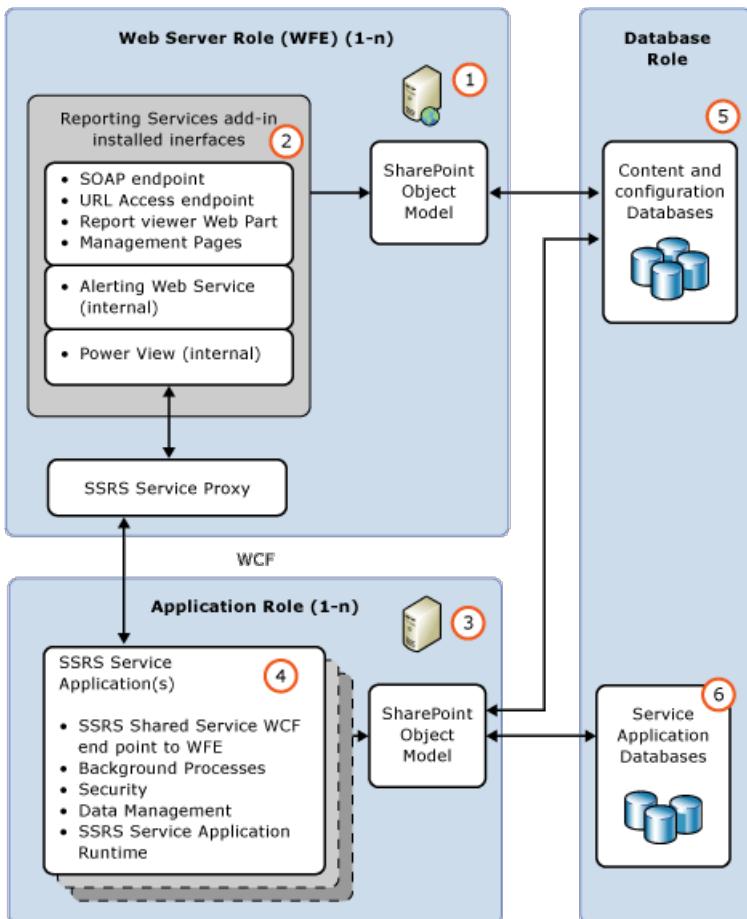
SharePoint mode

In SharePoint mode, a report server must run within a SharePoint server farm. The report server processing, rendering, and management features are represented by a SharePoint application server running the Reporting Services SharePoint shared service and one or more Reporting Services service applications. A SharePoint site provides the front-end access to report server content and operations.

SharePoint mode requires:

- SharePoint Foundation 2010 or SharePoint Server 2010.
- An appropriate version of the Reporting Services Add-in for SharePoint 2010 Products.
- A SharePoint application server with the Reporting Services shared service installed and at least one Reporting Services service application.

The following illustration shows a SharePoint mode Reporting Services environment:



	DESCRIPTION
(1)	Web servers or web front-ends (WFE). The Reporting Services add-in must be installed on each web server from which you want to utilize the web application features such as viewing reports or Reporting Services management pages for tasks such as managing data sources or subscriptions.
(2)	The add-in installs URL and SOAP endpoints for clients to communicate with the Application servers, through the Reporting Services service proxy.
(3)	Application servers running Reporting Services shared service. Scale-out of report processing is managed as part of the SharePoint farm and by adding the Reporting Services service to additional application servers.
(4)	You can create more than one Reporting Services service application, with different configurations including permissions, e-mail, proxy, and subscriptions.
(5)	Reports, data sources, and other items are stored in the SharePoint content databases.
(6)	Reporting Services service applications create three databases for report server, temp, and data alerting features. Configuration settings that apply to all SSRS service applications are stored in the RSReportServer.config file.

Report process and schedule and delivery process

The report server includes two processing engines that perform preliminary and intermediate report processing, and scheduled and delivery operations. The Report Processor retrieves the report definition or model, combines layout information with data from the data processing extension, and renders it in the requested format. The Scheduling and Delivery Process processes reports triggered from a schedule, and delivers reports to target destinations.

Report server database

The report server is a stateless server that stores all properties, objects, and metadata in a SQL Server database. Stored data includes published reports, compiled reports, report models, and the folder hierarchy that provides the addressing for all items managed by the report server. A report server database can provide internal storage for a single Reporting Services installation or for multiple report servers that are part of a scale-out deployment. If you configure a report server to run within a larger deployment of a SharePoint product or technology, the report server uses the SharePoint databases in addition to the report server database. For more information about data stores used in Reporting Services installation, see [Report Server Database \(SSRS Native Mode\)](#).

Authentication, rendering, data, and delivery extensions

The report server supports the following types of extensions: authentication extensions, data processing extensions, report processing extensions, rendering extensions, and delivery extensions. A report server requires at least one authentication extension, data processing extension, and rendering extension. Delivery and custom report processing extensions are optional, but necessary if you want to support report distribution or custom controls.

Reporting Services provides default extensions so that you can use all of the server features without having to develop custom components. The following table describes the default extensions that contribute to a complete report server instance that provides ready-to-use functionality:

TYPE	DEFAULT
Authentication	A default report server instance supports Windows Authentication, including impersonation and delegation features if they are enabled in your domain.
Data processing	A default report server instance includes data processing extensions for SQL Server, Analysis Services, Oracle, Hyperion Essbase, SAPBW, OLE DB, Parallel Data Warehouse, and ODBC data sources.
Rendering	A default report server instance includes rendering extensions for HTML, Excel, CSV, XML, Image, Word, SharePoint list, and PDF.
Delivery	A default report server instance includes an e-mail delivery extension and a file share delivery extension. If the report server is configured for SharePoint integration, you can use a delivery extension that saves reports to a SharePoint library.

NOTE

Reporting Services includes a complete set of tools and applications that you can use to administer the server, create content, and make that content available to users in your organization.

Related tasks

The following topics provide additional information on installing, using, and maintaining a report server:

TASK	LINK
Review Hardware and software requirements.	Hardware and Software Requirements for Reporting Services in SharePoint Mode
Install Reporting Services in SharePoint mode.	Install Reporting Services SharePoint Mode for SharePoint 2010
If you are a Web developer or have expertise in creating cascading style sheets, you can modify the default styles at your own risk to change the colors, fonts, and layout of the toolbar or Report Manager. Neither the default style sheets nor instructions for modifying the style sheets are documented in this release.	Customize Style Sheets for HTML Viewer and Report Manager
Web developers who are familiar with HTML styles and Cascade Style Sheets (CSS) can use the information in this topic to determine which files can be modified to customize the appearance of Report Manager.	Configure the Web Portal to Pass Custom Authentication Cookies
Explains how to tune the memory settings for the Report Server Web service and Windows service.	Configure Available Memory for Report Server Applications
Explains recommended steps to configure a report server for remote administration.	Configure a Report Server for Remote Administration
Provides instructions for configuring the availability of My Reports on a Native report server instance.	Enable and Disable My Reports
Provides instructions for setting up the RSClientPrint control that provides print functionality from within supported browsers. For more information on browser requirements, see Browser Support for Reporting Services and Power View .	Enable and Disable Client-Side Printing for Reporting Services

Next steps

[Reporting Services Extensions](#)

[Reporting Services Tools](#)

[Subscriptions and Delivery \(Reporting Services\)](#)

[Report Server Database \(SSRS Native Mode\)](#)

[Implementing a Security Extension](#)

[Implementing a Data Processing Extension](#)

[Data Sources Supported by Reporting Services \(SSRS\)](#)

More questions? Try asking the [Reporting Services forum](#)

Pin Reporting Services paginated report items to dashboards in Power BI

12/10/2018 • 6 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server 2016 Reporting Services and later Power BI Report Server

You can pin an on-premises Reporting Services paginated report item to a dashboard in the Power BI service, as a new tile. To pin, your administrator needs to first integrate your report server with Azure Active Directory and Power BI.

Requirements to Pin

- The report server is configured for Power BI integration. For more information, see [Power BI Report Server Integration \(Configuration Manager\)](#). If the report server has not been configured, you won't see the **Pin to Power BI Dashboard** button on the report viewer toolbar.



- You pin from the Reporting Services report viewer in the web portal, for example, <https://myserver/Reports>. You can't pin from Report Builder, from report designer in SQL Server Data Tools (SSDT), or from a report server URL. For example, <https://myserver/ReportServer>.
- You need to configure your browser to allow pop-ups from your report server site.
- You need to configure reports for stored credentials, if you want the pinned item to refresh. When you pin an item, a Reporting Services subscription is automatically created to manage the data refresh of the item to the dashboard. If the report does not use stored credentials, when the subscription runs you'll see a message similar to this one on the **My subscriptions** page.

"PowerBI Delivery error: dashboard: IT Spend Analysis Sample, visual: Chart2, error: The current action cannot be completed. The user data source credentials do not meet the requirements to run this report or shared dataset. Either the user data source credential."

See the section "Configure stored credentials for a report-specific data source (Native mode)" in [Store Credentials in a Reporting Services Data Source](#)

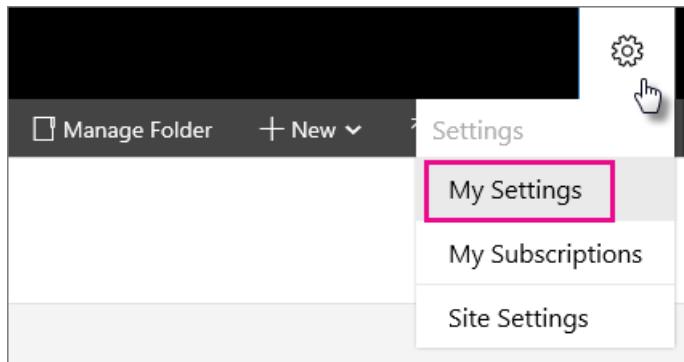
Items You Can Pin

The following report items can be pinned to a Power BI dashboard. You can't pin items that are nested inside a data region. For example, you can't pin an item that is nested inside a Reporting Services table or list.

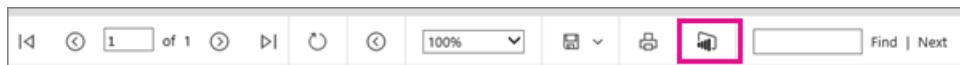
- Charts
- Gauge panels
- Maps
- Images
- Items need to be in the report body. You can't pin items that are in the page header or page footer.
- You can pin individual items that are inside a top-level rectangle but you can't pin them all as a single group.

To Pin a Report Item

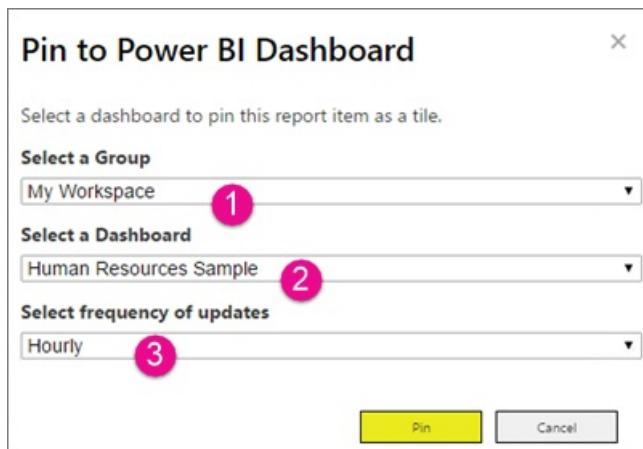
- Verify you are signed into Power BI. In the Reporting Services web portal, select the menu item **My Settings** and sign in. See [My Settings for Power BI Integration \(web portal\)](#) for more information.



- Navigate to the web portal folder that contains your report, and then view the report.
- While viewing the report, select the **Pin to Power BI** button the toolbar. You will be prompted to sign in, if you are not already signed in. If the Power BI button is not visible, the report server has not been integrated with Power BI. For more information, see [Power BI Report Server Integration \(Configuration Manager\)](#).



- Select the report item you want to pin to Power BI. You can only pin one item at a time. The report viewer presents a shaded view of your report and the report items you can pin are highlighted while the items that you can't pin, will be shaded dark.
- (1) select the group that contains the dashboard you want to pin to, (2) select the dashboard you want to pin the item too and (3) select how frequently you want the tile updated in the dashboard. **Tip** The refresh is managed by Reporting Services subscriptions and after the item is pinned, you can edit the subscription and configure a different refresh schedule.



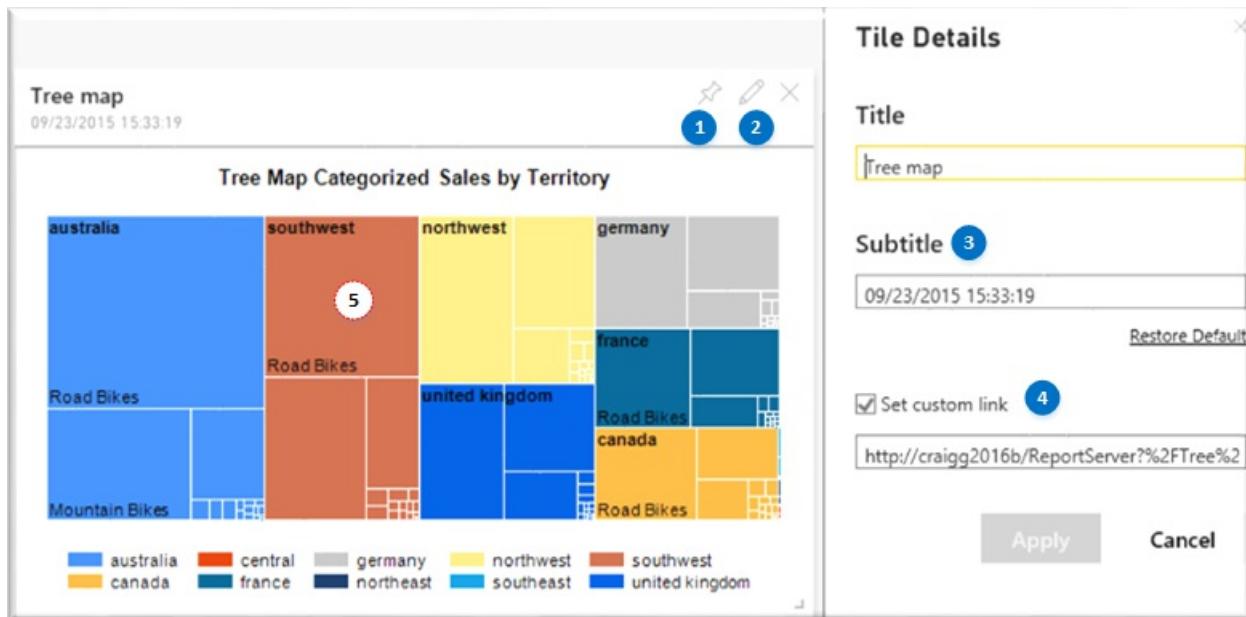
- Select **Pin**
- In the **Pin Successful** dialog, you can select the link **See it in Power BI** to navigate to the dashboard and see the item you just pinned.
- Select **Close** to return the report to the normal view.

In the Dashboard

After your report item is pinned in the dashboard, the tile looks like other dashboard tiles and there is no visible indication the tile came from Reporting Services. The following list summarizes how tile properties are populated from the report item.

From the Power BI dashboard the pinned report item behaves like other tiles:

- (1) You can pin the tile to other dashboards.
- (2) In the **Tile Details** you'll notice the Reporting Services report title is used for the default title of the tile.
- (3) The tile subtitle is based on the date and time the tile was pinned or the data was last refreshed from Reporting Services. The refresh schedule is managed by the Reporting Services subscription that was automatically created when you pinned the report item.
- (5) If you select the tile itself, Power BI uses the (4) **custom link** to navigate to the web portal page of the registered report server. the link was set when the item was pinned from Reporting Services. If you do not have internet connectivity to the report server, you will see an error in the browser.



Troubleshoot Issues

- **No Power BI button on the report viewer toolbar:** This message indicates the report server has not been integrated with Power BI. For more information, see [Power BI Report Server Integration \(Configuration Manager\)](#).
- **Cannot Pin:** When you attempt to pin an item, you see the following error message: See the section [Items You Can Pin](#).

Cannot Pin: There are no report items on this page that you can pin to [!INCLUDE[sspowerbi] (../includes/sspowerbi-md.md)].

- **Pinned items show stale data** in a Power BI dashboard and it did update for a period of time. The user credentials token has expired and you need to sign in again. The user credential registration with Azure and Power BI is good for 90 days. In the web portal, click **My Settings**. For more information, see [My Settings for Power BI Integration \(web portal\)](#).
- **Pinned items show stale data** in a Power BI dashboard and it has not refreshed even once. The issue is the report is not configured to use stored credentials. A report must use stored credentials because the action of pinning a report item creates a Reporting Services subscription to manage the refresh schedule of the tiles. Reporting Services subscriptions require stored credentials. If you review the **My Subscriptions** page, you see an error message similar to this one:

PowerBI Delivery error: dashboard: SSRS items, visual: Image3, error: The current action can't be completed. The user data source credentials do not meet the requirements to run this report or shared dataset. Either the user data source credentials are not stored in the report server database, or the user data source is configured not to require credentials but the unattended execution account is not specified. (rsInvalidDataSourceCredentialSetting)

- **Expired Power BI credentials:** You attempt to pin an item and see the following error message. In the web portal, click **My Settings** and on the My Settings page, click **Sign in**. See [My Settings for Power BI Integration \(web portal\)](#) for more information.

Cannot Pin: Unexpected Server Error: Missing, invalid or expired Power BI credentials.

- **Cannot Pin:** If you attempt to pin an item to a dashboard that is in a read-only state, you will see an error message similar to this one:

Server Error: The item 'Dashboard deleted 015cf022-8e2f-462e-88e5-75ab0a04c4d0' can't be found.
(rsItemNotFound)

Subscription Management

In addition to the subscription-related issues described in the troubleshooting section, the following information will help you maintain Power BI related subscriptions.

- **Item name changed:** If a pinned report item is renamed or deleted, the Power BI tile will no longer update and you will see an error message similar to the following. If you rename the item back the original name, the subscription will start working again and the tile will be refreshed on the subscriptions schedule.

PowerBI Delivery error: dashboard: SSRS items, visual: Image1, error: Error: Report item 'Image1' cannot be found.

You could also edit the subscription properties and change the **Report Visual Name** to the appropriate report item name. **Report Visual Name:**

- **Delete a tile.** If you delete a tile in Power BI, the associated subscription is not deleted in Reporting Services and on the **My subscriptions** page, you see an error similar to the following. You can delete the subscription.

PowerBI Delivery error: dashboard: SSRS items, visual: Image3, error: The item 'Tile deleted af7131d9-5eaf-480f-ba45-943a07d19c9f' cannot be found.

Video

<https://www.youtube.com/embed/QhPQObqmMPc>

See Also

[Power BI Report Server Integration \(Configuration Manager\)](#)

[My Settings for Power BI Integration \(web portal\)](#)

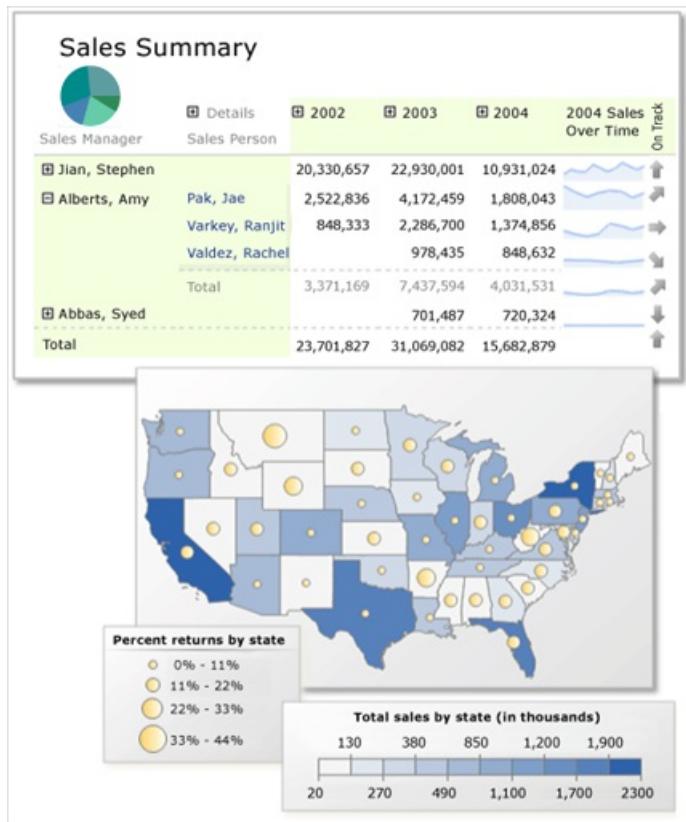
[Dashboards in Power BI](#)

 **Need help?** [MSDN Forum](#), [Stackoverflow](#), [Connect](#)

Report Builder in SQL Server

11/30/2018 • 4 minutes to read • [Edit Online](#)

Report Builder is a tool for authoring paginated reports, for business users who prefer to work in a stand-alone environment instead of using Report Designer in Visual Studio. When you design a paginated report, you're creating a report definition that specifies where to get the data, which data to get, and how to display the data. When you run the report, the report processor takes the report definition you have specified, retrieves the data, and combines it with the report layout to generate the report. You can preview your report in Report Builder. Then publish your report to a Reporting Services report server in native mode or in SharePoint integrated mode (2016 and earlier). You can also publish a paginated report to the Power BI service. Read more about [paginated reports in Power BI Premium \(Preview\)](#).



This paginated report features a matrix with row and column groups, sparklines, indicators, and a summary pie chart in the corner cell, accompanied by a map with two sets of geographic data represented by color and by circle size.

Jump-Start Report Creation

- **Start with a shared dataset.** Shared datasets are queries based on a shared data source and saved to a Reporting Services report server in native mode or in SharePoint integrated mode.
- **Start with the Table, Matrix, or Chart wizard.** Choose a data source connection, drag and drop fields to create a dataset query, select a layout and style, and customize your report.
- **Start with the Map wizard** to create reports that display aggregated data against a geographic or geometric background. Map data can be spatial data from a Transact-SQL query or an Environmental Systems Research Institute, Inc. (ESRI) shapefile. You can also add a Microsoft Bing map tile background.
- **Start your report with report parts.** Report parts are report items that have been published separately

to a Reporting Services report server in native mode or in SharePoint integrated mode. Report parts can be reused in other reports. Report items such as tables, matrices, charts, and images can be published as report parts.

Design Your Report

- **Create paginated reports with table, matrix, chart, and free-form report layouts.** Create table reports for column-based data, matrix reports (like cross-tab or PivotTable reports) for summarized data, chart reports for graphical data, and free-form reports for anything else. Reports can embed other reports and charts, together with lists, graphics, and controls for dynamic Web-based applications.
- **Report from a variety of data sources.** Build reports using data from any data source type that has a Microsoft .NET Framework-managed data provider, OLE DB provider, or ODBC data source. You can create reports that use relational and multidimensional data from SQL Server and Analysis Services, Oracle, Hyperion, and other databases. You can use an XML data processing extension to retrieve data from any XML data source. You can use table-valued functions to design custom data sources.
- **Modify existing reports.** By using Report Builder, you can customize and update reports that were created in SQL Server Data Tools (SSDT)Report Designer.
- **Modify your data** by filtering, grouping, and sorting data, or by adding formulas or expressions.
- **Add charts, gauges, sparklines, and indicators** to summarize data in a visual format, and present large volumes of aggregated information at a glance.
- **Add interactive features** such as document maps, show/hide buttons, and drillthrough links to subreports and drillthrough reports. Use parameters and filters to filter data for customized views.
- **Embed or reference images** and other resources, including external content.

Manage Your Report

- **Save the definition of the report** to your computer or to the report server, where you can manage it and share it with others.
- **Choose a presentation format** when you open the report, or after you open the report. You can select Web-oriented, page-oriented, and desktop application formats. Formats include HTML, MHTML, PDF, XML, CSV, TIFF, Word, and Excel.
- **Set up subscriptions.** After you publish the report to the report server or a report server in SharePoint integrated mode, you can configure your report to run at a specific time, create a report history, and set up e-mail subscriptions.
- **Generate data feeds** from your report by using the Reporting Services Atom rendering extension.

NOTE

Published reports are managed on a report server or a report server in SharePoint integrated mode by a report server administrator. Report server administrators can define security, set properties, and schedule operations such as report history and e-mail report delivery. They can create shared schedules and shared data sources and make them available for general use. Administrators also manage all of the report server folders. The ability to perform management tasks depends on user permissions.

See Also

[Start Report Builder](#)

[Install Report Builder](#)

[What's New in SQL Server Reporting Services and Report Builder](#)

Describes the new features in this version of Reporting Services and Report Builder.

[Tutorial: Creating a Quick Chart Report Offline](#)

Introduces Report Builder and the wizards available to help you create reports. The tutorial provides a set of data for you to work with so you do not need to connect to a data source to get started.

[Planning a Report \(Report Builder\)](#)

Provides information on what you should consider before you start to build your report.

[Report Authoring Concepts \(Report Builder and SSRS\)](#)

Defines key concepts used in throughout Report Builder documentation.

[Report Design View \(Report Builder\)](#)

Explains the different panes and regions of report design view.

[Shared Dataset Design View \(Report Builder\)](#)

Explains the different panes and regions of shared dataset design view.

[Keyboard Shortcuts \(Report Builder\)](#)

Outlines the shortcut keys available for navigating and designing reports in Report Builder.

Planning a Report (Report Builder)

11/30/2018 • 3 minutes to read • [Edit Online](#)

Report Builder lets you create many kinds of paginated reports. For example, you can create reports that show summary or detailed sales data, marketing and sales trends, operational reports, or dashboards. You can also create reports that take advantage of richly formatted text, such as for sales orders, product catalogs, or form letters. All these reports are created by using different combinations of the same basic building blocks in Report Builder. To create a useful, easily understood report, it helps to plan first. Here are some things you might want to consider before you get started:

- **What format do you want the report to appear in?**

You can render reports online in a browser such as the Reporting Services web portal or export them to other formats such as Excel, Word, or PDF. The final form your report takes is an important consideration because not all features are available in all export formats. For more information, see [Export Reports \(Report Builder and SSRS\)](#).

- **What structure do you want to use to present the data in the report?**

You have a choice among tabular, matrix (similar to a cross-tab or PivotTable report), chart, free-form structures, or any combination of these to present data. For more information, see [Tables, Matrices, and Lists \(Report Builder and SSRS\)](#) and [Charts \(Report Builder and SSRS\)](#).

- **What do you want your report to look like?**

Report Builder provides a lot of report items that you can add to your report to make it easier to read, highlight key information, help your audience navigate the report, and so on. Knowing how you want the report to appear can determine whether you need report items such as text boxes, rectangles, images, and lines. You might also want to show or hide items, add a document map, include drillthrough reports or subreports, or link to other reports. For more information, see [Images, Text Boxes, Rectangles, and Lines \(Report Builder and SSRS\)](#) and [Interactive Sort, Document Maps, and Links \(Report Builder and SSRS\)](#).

- **What data do you want your readers to see? Should the data or format be filtered for different audiences?**

You might want to narrow the scope of the report to specific users or locations, or to a particular time period. To filter the report data, use parameters to retrieve and display only the data you want. For more information, see [Report Parameters \(Report Builder and Report Designer\)](#).

- **Do you need to create your own calculations?**

Sometimes, your data source and datasets do not contain the exact fields that you need for your report. In that situation, you might have to create your own calculated fields. For example, you might want to multiply the price per unit times the quantity to get a line item sales amount. Expressions are also used to provide conditional formatting and other advanced features. For more information, see [Expressions \(Report Builder and SSRS\)](#).

- **Do you want to hide report items initially?**

Consider whether you want to hide report items, including data regions, groups and columns, when the report is first run. For example, you can initially present a summary table, and then drill down into the detailed data. For more information, see [Drilldown Action \(Report Builder and SSRS\)](#).

- **How are you going to deliver your report?**

You can save your report to your local computer and continue to work on it, or run it locally for your own information. However, to share your report with others, you need to save the report to a report server that is configured in native mode or a report server in SharePoint integrated mode. Saving it to a server lets others run it whenever they want to. Alternatively, the report server administrator can set up a subscription to the report or set up e-mail delivery of the report to other individuals. You can have the report delivered in a specific export format if you prefer. For more information, see [Finding, Viewing, and Managing Reports \(Report Builder and SSRS\)](#).

See Also

[Report Builder in SQL Server](#)

[Report Authoring Concepts \(Report Builder and SSRS\)](#)

[Report Builder Tutorials](#)

Reporting Services Reports (SSRS)

10/24/2018 • 10 minutes to read • [Edit Online](#)

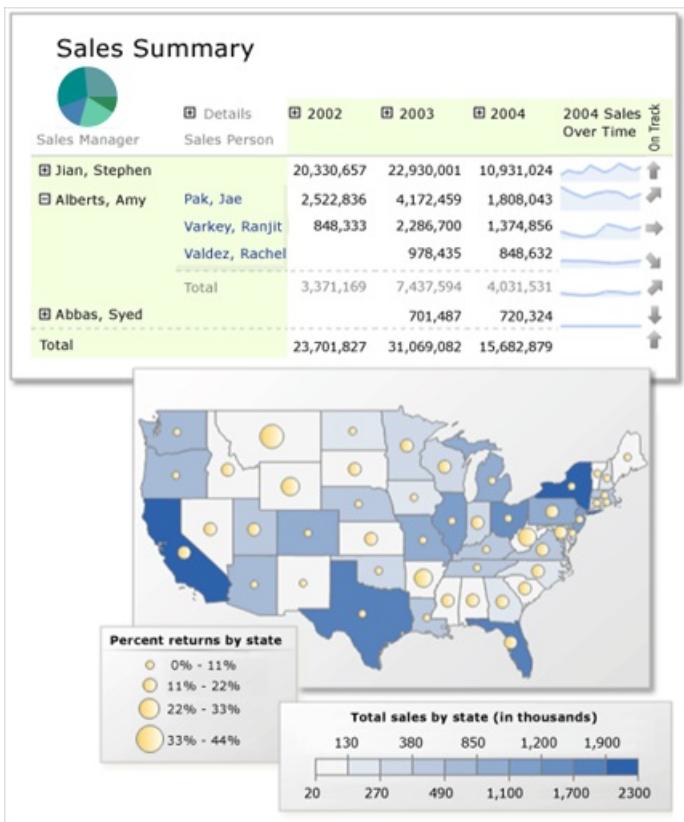
SQL Server Reporting Services paginated reports are XML-based report definitions that include report data and report layout elements. On a client file system, report definitions have the file extension .rdl. After you publish a paginated report, it is a report item stored on the report server or SharePoint site. Paginated reports are one part of the server-based reporting platform provided by Reporting Services. You can also [Create mobile reports with SQL Server Mobile Report Publisher](#).

If you are new to Reporting Services, be sure to review the information in [Reporting Services Concepts \(SSRS\)](#).

Benefits of Reporting Services Paginated Reports

You can use Reporting Services report solutions to:

- Use one set of data sources that provide a single version of the facts. Base reports on those data sources to provide a unified view of data to help make business decisions.
- Visualize your data in multiple, interconnected ways by using data regions. Display data organized in tables, matrices or cross-tabs, expand/collapse groups, charts, gauges, indicators or KPIs, and maps, with the ability to nest charts in tables.
- View reports for your own use or publish reports to a report server or SharePoint site to share with your team or organization.
- Define a report once and display it in a variety of ways. You can export the report to multiple file formats, or deliver the report to subscribers as e-mail or to a shared file. You can create multiple linked reports that apply separate parameter sets to the same report definition.
- Use report parts, shared data sources, shared queries, and subreports to define data visualizations for re-use.
- Manage report data sources separately from the report definition. For example, you can change from a test data source to a production data source without changing the report.
- Design reports in a free-form layout. Report layout is not restricted to bands of information. You can organize data display on the page in a way that promotes understanding, insight, and action.
- Enable drillthrough actions, expand/collapse toggles, sort buttons, Tooltips, and report parameters to enable report reader interactions with the report. Use report parameters combined with expressions that you write to enable report readers to control how data is filtered, grouped, and sorted.
- Define expressions that provide you with the ability to customize how report data is filtered, grouped, and sorted.



Stages of Report Processing

When you create a report, you define a report definition file (.rdl) in XML format. This file contains all the information that is needed to combine report data and report layout by the report processor. When you view a report, the report progresses through the following stages:

- **Compile.** Evaluate expressions in the report definition and store the compiled intermediate format internally on the report server.
- **Process.** Run dataset queries, and combine intermediate format with data and layout.
- **Render.** Send processed report to a rendering extension to determine how much information fits on each page and create the paged report.
- **Export (optional).** Export the report to a different file format.

For more information, see [Stages of reports](#) in [Reporting Services Concepts \(SSRS\)](#).

Create Paginated Reports

To create a paginated report:

- **Determine the purpose of report.** Identify the purpose of the report for the audience that uses it. A well-designed report provides information to report readers that leads to insight and action. Design decisions made during this step influence your choice of report parameters, report layout design, and report viewing experience. For more information, see [Planning a Report \(Report Builder\)](#) and [Report Design Tips \(Report Builder and SSRS\)](#).
- **Choose the type of query.** Determine whether to use a generalized, shared dataset query or a dataset query specific to your set of reports. A shared dataset with a generalized query is easy to maintain for use by multiple reports, but each report designer must filter the data as needed for their specific set of reports. For more information, see [Report Data \(SSRS\)](#).
- **Plan for views of related data.** Plan the viewing experience for your report readers. Summary reports

with the ability to drill down into detail data is a useful approach to handling large amounts of data. For more information, see [Drillthrough, Drilldown, Subreports, and Nested Data Regions \(Report Builder and SSRS\)](#).

- **Configure permissions.** Plan the strategy for granting the right level of permissions. A common strategy is to create a folder structure on the report server and grant access to reports and report related items based roles and folder security. For more information, see [Secure Reports](#).
- **Choose an authoring environment.** Authoring tools vary support for features. For more information, see [Reporting Services Tools](#).
- For each report:
 - **Identify sources of data.** Define report data sources, one for each source of data. For more information, see [Data Connections, Data Sources, and Connection Strings \(Report Builder and SSRS\)](#).
 - **Choose which data to use from each source.** For each data source, define report datasets. Each dataset includes a query to specify which data to use. If you have report parameters, define a dataset to populate the available values list for each parameter. For more information, see [Report Datasets \(SSRS\)](#) and [Report Parameters \(Report Builder and Report Designer\)](#).
 - **Choose a data visualization.** For each dataset, choose which data region to use for displaying the data. Choose from list of tables, charts, gauges, and maps. For more information, see the following topics:
 - [Tables, Matrices, and Lists \(Report Builder and SSRS\)](#)
 - [Charts \(Report Builder and SSRS\)](#)
 - [Sparklines and Data Bars \(Report Builder and SSRS\)](#)
 - [Indicators \(Report Builder and SSRS\)](#)
 - [Maps \(Report Builder and SSRS\)](#)
 - [Gauges \(Report Builder and SSRS\)](#)
 - **Customize the data and layout.** Design the report layout. A report definition has a report body, data sources, datasets, data regions, text boxes, lines, and images. Rectangles are used as containers for layout as well as visual elements. Customize each data region by writing expressions to control filter, group, sort, format, and display the data. Add report names, locations, and other identifying information that helps to manage dozens or hundreds of reports. Add visual elements and containers to organize the layout elements on the page. For more information, see the following topics:
 - [Filter, Group, and Sort Data \(Report Builder and SSRS\)](#)
 - [Report Parameters \(Report Builder and Report Designer\)](#)
 - [Expressions \(Report Builder and SSRS\)](#)
 - [Formatting Report Items \(Report Builder and SSRS\)](#)
 - [Images, Text Boxes, Rectangles, and Lines \(Report Builder and SSRS\)](#)
 - [Page Layout and Rendering \(Report Builder and SSRS\)](#)
 - **Configure interactivity features.** Add interactivity features for your report readers. For example, add sort buttons or toggle items for viewing the queries. For more information, see [Interactive Sort, Document Maps, and Links \(Report Builder and SSRS\)](#).

- **Review and iterate on design.** Preview the report. Publish a preliminary version to get feedback from your report readers. Iterate on the design.
- **Review reporting solution.** Verify that the set of reports interact correctly.
- **Consider which components can be re-used.** Determine if any of the data sources or dataset queries can be shared for re-use. If so, on the report server or SharePoint site, create shared data sources and shared datasets. Determine if the data regions are suitable for re-use as report parts. For more information, see [Report Parts in Report Designer \(SSRS\)](#).

Preview Reports

Each report authoring tool supports previewing a report. For more information, see the [Preview](#) section of [Design Reports with Report Designer \(SSRS\)](#), and [Previewing Reports in Report Builder](#).

Save or Publish Reports

Each authoring tool supports saving a report locally or publishing the report to a report server or SharePoint site. For more information, see the [Save and Deploy](#) section of [Design Reports with Report Designer \(SSRS\)](#), and [Saving Reports \(Report Builder\)](#).

View Reports

In addition to previewing a report saved locally or published to a report server, you can provide a variety of viewing experiences for your report readers. To view a report:

- **Browser.** Use the Report Server Web Service or SharePoint site to view published reports. On a SharePoint site, you can also configure a Web part to view published reports. For more information, see [Browser Support for Reporting Services and Power View, Report Manager \(SSRS Native Mode\)](#), and [URL Access \(SSRS\)](#).
- **Delivery.** Configure a subscription to deliver reports to report readers in e-mail or to a shared file folder. For more information, see [Subscriptions and Delivery \(Reporting Services\)](#).
- **Export.** From the report viewer toolbar, a report reader can export a report to a different file format. Export file formats can be configured by the report server administrator. For more information, see [Export Reports \(Report Builder and SSRS\)](#)
- **Print.** A report reader can print a report or pages of a report depending on the way in which it is viewed. For more information, see [Print Reports \(Report Builder and SSRS\)](#).
- **Web or Windows Form application.** Use Visual Studio to develop an ASP.NET AJAX application or Windows Form application that hosts the Report Viewer control. The control can point to published reports on a report server. For more information, see [Microsoft Reports](#).

Manage Reports

To manage a published report:

- **Data sources.** Shared and embedded data sources are managed independently from the report definition.
- **Datasets.** Shared datasets are managed independently from the report definition.
- **Parameters.** Parameters are managed independently from the report definition. After parameters are changed on the report server, report authoring clients cannot publish over the changes made on the server.
- **Resources.** Images and spatial data in ESRI shapefiles are resources that can be published and managed independently from the report definition.

- **Report cache.** By scheduling large reports to run during off-peak hours, you can reduce processing impact on the report server during core business hours.
- **Snapshots.** Use report snapshots when you want to provide consistent results for multiple users who must work with identical sets of data. With volatile data, an on-demand report can produce different results from one minute to the next. A report snapshot, by contrast, allows you to make valid comparisons against other reports or analytical tools that contain data from the same point in time.
- **Report history.** By creating a series of report snapshots, you can build a history of a report that shows how data changes over time.

For more information about performance, see [Performance, Snapshots, Caching \(Reporting Services\)](#).

Secure Reports

To secure a report:

- From the report server administrator, identify the authorization and authentication system that is used for your Reporting Services installation. By default, Reporting Services uses Windows authentication, integrated security, and role assignment to help control access to published reports. For more information, see [Roles and Permissions \(Reporting Services\)](#) and [Reporting Services Security and Protection](#).

Create Notifications Based on Report Data

You can create data alerts for published reports on a SharePoint site. Data alerts are based on data feeds from data regions in the report. By default, data regions are named automatically. Report authors can make it easier to create data alerts in their reports by naming data regions based on their business purpose. When you create a data alert, you are notified in email when data meets the conditions that you specify. For more information, see [Generating Data Feeds from Reports \(Report Builder and SSRS\)](#), [Create a Data Alert in Data Alert Designer](#) and [Reporting Services Data Alerts](#).

Upgrade Reports

Reporting Services supports multiple versions of report definitions, report servers, and SharePoint sites. To upgrade a report:

- Upgrade a report server installation. Compiled reports stored on the report server are upgraded automatically on first use. The report definition (.rdl) is not changed. For more information, see [Upgrade and Migrate Reporting Services](#).
- Open a report in a report authoring environment. The report definition is upgraded in most circumstances. For more information, see [Upgrade Reports and Deployment and Version Support in SQL Server Data Tools \(SSRS\)](#).

Troubleshoot Reports

To troubleshoot a report:

- **Determine where the issue is occurring.** Review the information in [Stages of a Report](#).
- **Determine where you can find more information.** For example, for report design that includes expressions, the Report Designer tool provides more information about expression evaluation issues than the Report Builder tool. For report processing errors, the log files contain detailed information.

See Also

[Reporting Services Tools](#)

[Extensions \(SSRS\)](#)

[Reporting Services Report Server](#)

Report Data in SQL Server Reporting Services (SSRS)

12/17/2018 • 6 minutes to read • [Edit Online](#)

Report data can come from multiple sources of data in your organization. Your first step in designing a report is to create data sources and datasets that represent the underlying report data. Each data source includes data connection information. Each dataset includes a query command that defines the set of fields to use as data from a data source. To visualize data from each dataset, add a data region, such as a table, matrix, chart, or map. When the report is processed, the queries run on the data source, and each data region expands as needed to display the query results for the dataset.

Terms

- **Data connection.** Also known as a *data source*. A data connection includes a name and connection properties that are dependent on the connection type. By design, a data connection does not include credentials. A data connection does not specify which data to retrieve from the external data source. To do that, you specify a query when you create a dataset.
- **Data source definition.** A file that contains the XML representation of a report data source. When a report is published, its data sources are saved on the report server or SharePoint site as data source definitions, independently from the report definition. For example, a report server administrator might update the connection string or credentials. On a native report server, the file type is .rds. On a SharePoint site, the file type is .rsds.
- **Connection string.** A connection string is a string version of the connection properties that are needed to connect to a data source. Connection properties differ based on data connection type. For examples, see [Data Connections, Data Sources, and Connection Strings in Report Builder](#).
- **Shared data source.** A data source that is available on a report server or SharePoint site to be used by multiple reports.
- **Embedded data source.** Also known as a *report-specific data source*. A data source that is defined in a report and used only by that report.
- **Credentials.** Credentials are the authentication information that must be provided to allow you access to external data.

Tips for Specifying Report Data

Use the following information to design your report data strategy.

- **Data sources** Data sources can be published and managed independently from reports on a report server or SharePoint site. For each data source, you or the database owner can manage connection information in one place. Data source credentials are stored securely on the report server; you do not include passwords in the connection string. You can redirect a data source from a test server to a production server. You can disable a data source to suspend all reports that use it.
- **Datasets** Datasets can be published and managed independently from reports or the shared data sources that they depend on. You or the database owner can provide optimized queries for report authors to use. When you change the query, all reports that use the shared dataset use the updated query. You can enable dataset caching to improve performance. You can schedule query caching for a specific time or use a shared schedule.

- **Data used by report parts** Report parts can include the data that they depend on. For more information about report parts, see [Report Parts in Report Designer \(SSRS\)](#).
- **Filter data** Report data can be filtered in the query or in the report. You can use datasets and query variables to create cascading parameters, and provide a user the ability to narrow choices from thousands of selections to a more manageable number. You can filter data in a table or chart based on parameter values or other values that you specify.
- **Parameters** Dataset query commands that include query variables automatically create matching report parameters. You can also manually create parameters. When you view a report, the report toolbar displays the parameters. Users can select values to control report data or report appearance. To customize report data for specific audiences, you can create sets of report parameters with different default values linked to the same report definition, or use the built-in **UserID** field. For more information, see [Report Parameters \(Report Builder and Report Designer\)](#) and [Built-in Collections in Expressions \(Report Builder and SSRS\)](#).
- **Data alerts** After a report is published, you can create alerts based on report data, and receive email messages when it meets rules that you specify.
- **Group and aggregate data** Report data can be grouped and aggregated in the query or in the report. If you aggregate values in the query, you can continue to combine values in the report within the constraints of what is meaningful. For more information, see [Filter, Group, and Sort Data \(Report Builder and SSRS\)](#) and [Aggregate Function \(Report Builder and SSRS\)](#).
- **Sort data** Report data can be sorted in the query or in the report. In tables, you can also add an interactive sort button to let the user control the sort order.
- **Expression-based data** Because most report properties can be expression-based, and expressions can include references to dataset fields and report parameters, you can write powerful expressions to control report data and appearance. You can provide a user the ability to control the data they see by defining parameters.
- **Display data from a dataset** Data from a dataset is typically displayed on one or more data regions, for example, a table and a chart.
- **Display data from multiple datasets** You can write expressions in a data region based on one dataset that look up values or aggregates in other datasets. You can include subreports in a table based on one dataset to display data from a different data source.

Use the following list to help define sources of data for a report.

- Consider whether to use embedded or shared data sources and datasets. Collaborate with owners of sources of data to implement and use authentication and authorization technology that is appropriate for your organization.
- Understand the software data layer architecture for your organization and the potential issues arising from data types. Understand how data extensions and data processing extensions can affect query results. Data types differ among the source of data, data providers, and the data types stored in the report definition (.rdl) file.
- Understand the Reporting Services client/server architectures and tools. For example, in Report Designer, you author reports on a client machine that uses built-in data source types. When you publish a report, the data source types must be supported on the report server or SharePoint site. For more information, see [Data Sources Supported by Reporting Services \(SSRS\)](#).
- Data sources and datasets are authored in a report and published to a report server or SharePoint site from a client authoring tool. Data sources can be created directly on the report server. After they are published, you can configure credentials and other properties on the report server. For more information, see [Data Connections, Data Sources, and Connection Strings \(Report Builder and SSRS\)](#) and [Reporting Services](#)

Tools.

- The data sources you can use depend on which Reporting Services data extensions are installed. Support for data sources can differ by client authoring tool, report server version, and report server platform. For more information, see [Data Sources Supported by Reporting Services \(SSRS\)](#).
- Data source credentials vary based on data source type and on whether you are viewing reports on your client or report server or SharePoint site. For more information, see [Set Permissions for Report Server Items on a SharePoint Site \(Reporting Services in SharePoint Integrated Mode\)](#), [Specify Credential and Connection Information for Report Data Sources](#), and credential information specific to each tool in [Reporting Services Tools](#).

Related Tasks

Tasks related to creating data connections, adding data from external sources, datasets, and queries.

Common Tasks	Links
Create data connections	Data Connections, Data Sources, and Connection Strings (Report Builder and SSRS)
Create datasets and queries	Report Embedded Datasets and Shared Datasets (Report Builder and SSRS)
Manage data sources after they are published	Manage Report Data Sources
Manage shared datasets after they are published	Manage Shared Datasets
Create and manage data alerts	Reporting Services Data Alerts
Cache a shared dataset	Cache Shared Datasets (SSRS)
Schedule a shared dataset to preload the cache	Schedules
Add a data extension	Implementing a Data Processing Extension

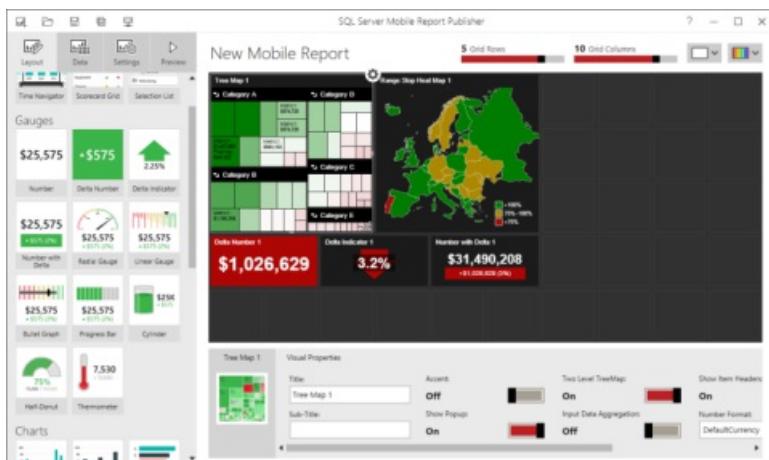
Create mobile reports with SQL Server Mobile Report Publisher

12/10/2018 • 2 minutes to read • [Edit Online](#)

Learn about Reporting Services mobile reports, optimized for mobile devices and connected to on-premises data, with an assortment of data visualizations.

NOTE

Do you need to migrate Datazen Server content such as dashboards and KPIs to a SQL Server Reporting Services server? Try the [SQL Server Migration Assistant for Datazen](#).



With SQL Server Mobile Report Publisher, you can quickly create Reporting Services mobile reports, optimized for mobile devices and a variety of other form factors. Mobile reports feature an assortment of visualizations, from time, category, and comparison charts, to treemaps and custom maps.

- Connect your mobile reports to a range of data sources, including on-premises SQL Server and Analysis Services data.
- Lay out your mobile reports on a design surface with adjusting grid rows and columns, and flexible mobile report elements that scale well to any screen size.
- Then save these mobile reports to a Reporting Services server, and view and interact with them in a browser or in the Power BI mobile app on iPads, iPhones, Android phones and tablets, and Windows 10 devices.

Create Reporting Services mobile reports

These articles will help you get started.

- Download [SQL Server Mobile Report Publisher](#)
- [Create a Reporting Services mobile report](#)
- [End-to-end walkthrough: Create mobile reports and KPIs in SQL Server Reporting Services](#) (Christopher Finlan's blog)
- [Design first, or data first](#): Decide whether to design your report first with simulated data, or start with your own data.
- [Data for Reporting Services mobile reports](#): Use data from shared datasets or prepare data from Excel workbooks to use in your mobile reports.
- [How data refresh works in mobile reports and KPIs in Reporting Services](#) (Christopher Finlan's blog): Read

about setting up caching for shared datasets so you control how often data is refreshed and speed up report performance.

- [Visualizations in mobile reports](#)
- [Gauges in mobile reports](#)
- [Maps in mobile reports](#)
- [Brand your web portal and mobile reports](#) with the colors and logo of your business

SSRS mobile reports in the Power BI mobile apps

- View [Reporting Services mobile reports and KPIs](#) in the Power BI mobile apps for iOS and Android
- View [Reporting Services mobile reports and KPIs](#) in the Power BI app for Windows 10 devices

See Also

- [Create, Modify, and Delete Shared Data Sources \(SSRS\)](#)
- [Manage Shared Datasets](#)
- [Working with KPIs in Reporting Services](#)
- [Enable a report server for Power BI mobile access](#)

Reporting Services Security and Protection

10/1/2018 • 2 minutes to read • [Edit Online](#)

You can use information in this section to learn about the security features of SQL Server Reporting Services. This section also explains the authorization models and authentication providers supported in Reporting Services.

Extended Protection for Authentication

Beginning with SQL Server 2008 R2, support for Extended Protection for Authentication is available. The SQL Server feature supports the use of channel binding and service binding to enhance protection of authentication. The SQL Server features need to be used with an operating system that supports Extended Protection. For more information, see [Extended Protection for Authentication with Reporting Services](#).

Authentication and Authorization

Reporting Services provides different authentication types for users and client applications to authenticate with the report server. Using the right authentication type for your report server enables your organization to achieve the appropriate level of security required by your organization. For more information, see [Authentication with the Report Server](#).

Reporting Services also employs roles and permissions to control user access to content in the report server catalog (in other words, who can access what, and how s/he can access it). For more information, see [Roles and Permissions \(Reporting Services\)](#).

Related Tasks

TASK DESCRIPTIONS	LINKS
Configure the Secure Socket Layer (SSL) to secure client connections to the report server.	Configure SSL Connections on a Native Mode Report Server

Schedules

11/30/2018 • 10 minutes to read • [Edit Online](#)

Reporting Services provides **shared schedules** and **report-specific schedules** to help you control processing and distribution of reports. The difference between the two types of schedules is how they are defined, stored, and managed. The internal construction of the two types of schedules is the same. All schedules specify a type of recurrence: monthly, weekly, or daily. Within the recurrence type, you set the intervals and range for how often an event is to occur. The type of recurrence pattern and how those patterns are specified is the same whether you create a shared schedule or a report-specific schedule.

- Shared schedules are created as separate items. After they are created, you reference them when defining a subscription or some other scheduled operation.
- Report-specific schedules are created when you define a subscription or set report execution properties; filling out schedule information is part of defining a subscription or setting properties. To define a report-specific schedule, you open the report or subscription that uses it.

A shared schedule contains schedule and recurrence information that can be used by any number of published reports and subscriptions that run on a Reporting Services report server. If you have many reports and subscriptions that run at the same time, you can create a shared schedule for those jobs. If you want to subsequently change the recurrence pattern or the end date, you can make the change in one place.

Shared schedules are easier to maintain and give you more flexibility in managing scheduled operations. For example, you can pause and resume shared schedules. Also, if you find that too many scheduled operations are running at the same time, you can create multiple shared schedules that run at different times and then adjust the schedule information until the processing load evens out across the report server.

What you can do with Schedules

You can use the Reporting Services Web portal and SQL Server Management Studio in Native mode and SharePoint site administration pages in SharePoint mode to create and manage your schedules. You can:

- Schedule report delivery in a standard or data-driven subscription.
- Schedule report history so that new snapshots are added to report history at regular intervals.
- Schedule when to refresh the data of a report snapshot.
- Schedule when to refresh the data of a shared dataset
- Schedule the expiration of a cached report or shared dataset to occur at a predefined time so that it can be subsequently refreshed.

You can create a shared schedule if you want to use the same schedule information for many reports or subscriptions. Shared schedules are defined separately, and then referenced in reports, shared datasets, and subscriptions that need schedule information.

When you create a schedule, the report saves the schedule information in the report server database or for SharePoint mode, the service application database. The report server also creates a SQL Server Agent job that is used to trigger the schedule. Schedule processing is based on the local time of the report server that contains the schedule. The time format follows the Microsoft Windows operating system standard.

For details on how to create and manage schedules, see [Create, Modify, and Delete Schedules](#).

NOTE

Schedule operations are not available in every edition of SQL Server. For a list of features that are supported by the editions of SQL Server, see [Editions and supported features of SQL Server 2017](#).

Comparing Shared and Report-Specific Schedules

Both types of schedules yield the same output:

- **Shared schedules** are portable, multipurpose items that contain ready-to-use schedule information.

Because shared schedules are system-level items, creating a shared schedule requires system-level permissions. For this reason, a report server administrator or content manager typically creates the shared schedules that are available on your report server. Shared schedules are stored and managed on the report server by using the Web portal or SharePoint site settings.

In contrast with specific schedules that you define through report, shared dataset, or subscription properties, shared schedules are easier to manage and maintain for the following reasons:

- Shared schedules can be managed from a central location, making it easier to compare schedule properties and adjust frequency and recurrence patterns if scheduled operations are running too close together or conflicting with other processes on your server.
 - Allows you to quickly adapt to changes in the computing environment. For example, suppose you have a set of reports that run at 4:00 A.M. after a data warehouse is refreshed. If the data refresh operation is rescheduled or is delayed, you can easily accommodate that change by updating the schedule information in a single shared schedule.
 - If you use only shared schedules, you know precisely when scheduled operations occur. This makes it easier to anticipate and accommodate server loads before performance issues occur. For example, if you decide to schedule computer backups at a specific hour, you can adjust shared schedules to run at different times.
- **Report-specific schedules** are defined in the context of an individual report, subscription, or report execution operation to determine cache expiration or snapshot updates. These schedules are created inline when you define a subscription or set report execution properties. You can create a report-specific schedule if a shared schedule does not provide the frequency or recurrence pattern that you need. To prevent a report from running, you must edit a report-specific schedule manually. Report-specific schedules can be created by individual users.

Configure the Data Sources

Before you can schedule data or subscription processing for a report, you must configure the report data source to use stored credentials or the unattended report processing account. If you use stored credentials, you can only store one set of credentials, and they will be used by all users who run the report. The credentials can be a Windows user account or a database user account.

The unattended report processing account is a special-purpose account that is configured on the report server. It is used by the report server to connect to remote computers when a scheduled operation requires the retrieval of an external file or processing. If you configure the account, you can use it to connect to external data sources that provide data to a report.

To specify stored credentials or the unattended report processing account, edit the data source properties of the report. If the report uses a shared data source, edit the shared data source instead.

Store Credentials and Processing accounts

How you work with a schedule depends on tasks that are part of your role assignment. If you are using predefined roles, users who are Content Managers and System Administrators can create and manage any schedule. If you use custom role assignments, the role assignment must include tasks that support scheduled operations.

TO DO THIS	INCLUDE THIS TASK	NATIVE MODE PREDEFINED ROLES	SHAREPOINT MODE GROUPS
Create, modify, or delete shared schedules	Manage shared schedules	System Administrator	Owners
Select shared schedules	View shared schedules	System User	Members
Create, modify, or delete report-specific schedules in a user-defined subscription	Manage individual subscriptions	Browser, Report Builder, My Reports, Content Manager	Visitors, Members
Create, modify, or delete report-specific schedules for all other scheduled operations	Manage report history, manage all subscriptions, manage reports	Content Manager	Owners

For more information about security in Native mode Reporting Services, see [Predefined Roles, Granting Permissions on a Native Mode Report Server](#) and [Tasks and Permissions](#). For SharePoint mode, see [Compare Roles and Tasks in Reporting Services to SharePoint Groups and Permissions](#)

How Scheduling and Delivery Processing Works

The Scheduling and Delivery Processor provides the following functionality:

- Maintains a queue of events and notifications in the report server database. In a scale-out deployment, the queue is shared across all of the report servers in the deployment.
- Calls the Report Processor to execute reports, process subscriptions, or clear a cached report. All report processing that occurs as a result of a schedule event is performed as a background process. SharePoint mode utilizes timer jobs to .
- Calls the delivery extension that is specified in a subscription so that the report can be delivered.

Other aspects of a scheduling and delivery operation are handled by other components and services that work with the Scheduling and Delivery Processor. Specifically, the Scheduling and Delivery Processor runs in the Report Server service and uses SQL Server Agent as a timer to generate scheduled events. The following step-by-step description explains how the scheduled operations work in a Reporting Services deployment:

1. A scheduled operation is defined when a user creates a schedule. The schedule defines a date and time that will be used to trigger a subscription for report delivery, refresh a snapshot, or expire a cache.
2. The report server saves the schedule information in the report server database.
3. The report server creates a corresponding job in SQL Server Agent that includes the schedule information provided. The jobs are created through a stored procedure, using the existing open connection to the report server database.
4. SQL Server Agent runs the job on the date and time specified in the schedule. The job creates an event that is added to a queue maintained by Reporting Services.
5. The event causes a report or subscription process to occur. Events are processed when they are detected in the queue, and the report is processed or delivered accordingly.

Before the events are processed, the Scheduling and Delivery Processor performs an authentication step to verify that the subscription owner has permission to view the report.

Reporting Services maintains an event queue for all scheduled operations. It polls the queue at regular intervals to check for new events. By default, the queue is scanned at 10 second intervals. You can change the interval by modifying the **PollingInterval**, **IsNotificationService**, and **IsEventService** configuration settings in the RSReportServer.config file. SharePoint mode also uses the RSreporserver.config for these settings and the values apply to all Reporting Services service applications. For more information, see [RsReportServer.config Configuration File](#).

Server Dependencies

The Scheduling and Delivery Processor requires that the Report Server service and SQL Server Agent are started. The Schedule and Delivery Processing feature must be enabled through the **ScheduleEventsAndReportDeliveryEnabled** property of the **Surface Area Configuration for Reporting Services** facet in Policy-Based Management. Both SQL Server Agent and the Report Server service must running in order for scheduled operations to occur.

NOTE

You can use the **Surface Area Configuration for Reporting Services** facet to stop scheduled operations on a temporary or permanent basis. Although you can create and deploy custom delivery extensions, by itself the Scheduling and Delivery Processor is not extensible. You cannot change how it manages events and notifications. For more information about turn off features, see the **Scheduled Events and Delivery** section of [Turn Reporting Services Features On or Off](#).

Effects of Stopping the SQL Server Agent

Scheduled report processing uses SQL Server Agent by default. If you stop the service, no new processing requests are added to the queue unless you add them programmatically through the [FireEvent](#) method. When you restart the service, the jobs that create report processing requests are resumed. The report server does not try to recreate report processing jobs that might have occurred in the past, while SQL Server Agent was offline. If you stop SQL Server Agent for a week, all scheduled operations are lost for that week.

NOTE

The functionality that SQL Server Agent provides to Reporting Services can be replaced with custom code that uses the [FireEvent](#) method to add schedule events to the queue.

Effects of Stopping the Report Server Service

If you stop the Report Server service, SQL Server Agent continues to add report processing requests to the queue. Status information from SQL Server Agent indicates that the job succeeded. However, because the Report Server service is stopped, no report processing actually occurs. The requests will continue to accumulate in the queue until you restart the Report Server service. Once you restart the Report Server service, all report processing requests that are in the queue are processed in order.

See Also

[Create, Modify, and Delete Snapshots in Report History](#)

[Subscriptions and Delivery \(Reporting Services\)](#)

[Data-Driven Subscriptions](#)

[Caching Reports \(SSRS\)](#)

[Report Server Content Management \(SSRS Native Mode\)](#)

[Cache Shared Datasets \(SSRS\)](#)

Reporting Services Tools

11/30/2018 • 6 minutes to read • [Edit Online](#)

SQL Server Reporting Services contains a set of graphical and scripting tools that support the development and use of rich reports in a managed environment. The tool set includes development tools, configuration and administration tools, and report viewing tools. This topic gives a brief overview of each tool in Reporting Services and how it can be accessed.

To find a tool right away, see [Tutorial: How to Locate and Start Reporting Services Tools \(SSRS\)](#).

Tools for Report Authoring

The following table lists the available tools for report authoring in SQL Server Reporting Services.

TOOL	DESCRIPTION	HOW TO ACCESS
SQL Server Mobile Report Publisher	<p>With Mobile Report Publisher, you can create mobile reports that dynamically adjust the content to fit your screen or browser window and scale well to any screen size.</p> <p>Create mobile reports on a design surface with adjustable grid rows and columns, and flexible mobile report elements.</p> <p>For more information, see Create mobile reports with SQL Server Mobile Report Publisher.</p>	Download the SQL Server Mobile Report Publisher
Power View	An interactive data exploration and visual presentation experience designed to let you create and interact with reports based on Analysis Services tabular models.	Reporting Services in SharePoint mode. Browser with Silverlight.

Tool	Description	How to Access
Report Designer	<p>Use this tool to design reports. Includes the following features:</p> <ul style="list-style-type: none"> Deploy to a native mode or SharePoint mode report server. Hosted in SQL Server Data Tools (SSDT) Report Data pane to organize data used in your report Tabbed views for Design and Preview for interactive report design Query designers to help specify which data to retrieve from data sources and that are associated with data source types in the RSReportDesigner Configuration File Expression editor with IntelliSense to build Visual Basic expressions that customize report content and appearance Supports custom report items and custom query designers <p>For more information, see Reporting Services in SQL Server Data Tools (SSDT).</p>	SQL Server Data Tools (SSDT)
Report Builder	<p>Use this tool to design reports. Includes the following features:</p> <ul style="list-style-type: none"> Deploy to a native mode or SharePoint mode report server. Microsoft Office-like authoring environmentSQL Server Mobile Report Publisher Ability to save report items as report parts A wizard for creating maps Aggregates of aggregates Enhanced support for expressions Query designers to help specify which data to retrieve from a selection of built-in data sources types <p>For more information, see Report Builder in SQL Server.</p>	<p>Download the standalone version of Report Builder</p> <p>Or open from Report Manager/SharePoint</p>

Tools for Report Server Administration

A set of graphical and scripting tools are available for administering the report server in SQL Server Reporting Services. The tools that you use depend on the deployment mode of your report server.

Native Mode

The following table lists the available tools for administering the report server that is deployed in native mode.

TOOL	DESCRIPTION	HOW TO ACCESS
Reporting Services Configuration Manager	<p>Use this tool to configure a Reporting Services installation. Available tasks include:</p> <p>Configuring both local and remote report server instances</p> <p>Configuring the Report Server service account.</p> <p>Creating and configuring one or more Web service URL.</p> <p>Configuring the Report Manager URL</p> <p>Creating and configuring the report server database.</p> <p>Configuring a scale-out deployment.</p> <p>Backup, restoring, or replacing the symmetric key that is used to encrypt stored connection strings and credentials.</p> <p>Configuring the unattended execution account.</p> <p>Configuring an SMTP server for e-mail delivery.</p> <p>Note: The Reporting Services Configuration Manager does not help you manage report server content, enable additional features, or grant access to the server.</p> <p>For more information, see Reporting Services Configuration Manager (Native Mode).</p>	Start menu

Tool	Description	How to Access
SQL Server Management Studio	<p>Use this tool to manage one or more report server instances in a single environment, including:</p> <ul style="list-style-type: none"> Managing both local and remote report server instances Setting report server properties Modifying role definitions Turning off report server features you are not using Managing jobs Managing shared schedules 	Start menu
SQL Server Configuration Manager	<p>Use this tool to:</p> <ul style="list-style-type: none"> Start and stop the Reporting Services Windows services Configure Customer Feedback Reporting, the dump directory location, and error reporting <p>**** Warning ****Do not use this tool to configure service account. Use the Reporting Services Configuration tool instead.</p> <p>For more information, see SQL Server Configuration Manager.</p>	Start menu
Rsconfig Utility	<p>Use this tool to configure and manage a report server connection to the report server database. You can also use it to specify a user account to use for unattended report processing.</p> <p>For more information, see Report Server Command Prompt Utilities (SSRS).</p>	Command prompt

TOOL	DESCRIPTION	HOW TO ACCESS
Rskeymgmt Utility	<p>Use this tool to:</p> <p>Extract, restore, create, and delete the symmetric key used to encrypt report server data</p> <p>Join report server instances in a scale-out deployment</p> <p>For more information, see Report Server Command Prompt Utilities (SSRS).</p>	Command prompt
Windows Management Instrumentation (WMI) Classes	<p>Use these classes to automate the configuration tasks in Reporting Services Configuration Manager without the need to use the graphical user interface.</p> <p>For more information, see Accessing the WMI Provider Programmatically.</p>	Visual Basic script

SharePoint Integrated Mode

In SharePoint mode, the Reporting Services is a service application in the SharePoint architecture, and is administered directly through SharePoint

TOOL	DESCRIPTION	HOW TO ACCESS
SharePoint Central Administration	<p>Use SharePoint Central Administration to create, query, and manage the shared service applications for Reporting Services.</p> <p>For more information, see Configuration and Administration of a Report Server (Reporting Services SharePoint Mode).</p>	Browser to the SharePoint site URL for Central Administration
PowerShell Cmdlets	<p>Use PowerShell cmdlets to create, query, and manage the shared service applications for Reporting Services.</p> <p>For more information, see PowerShell cmdlets for Reporting Services SharePoint Mode.</p>	SharePoint 2010 Management Shell

Tools for Report Content Management

A set of graphical and scripting tools are available for managing content in SQL Server Reporting Services. The tools that you use depend on the deployment mode of your report server.

TOOL	DESCRIPTION	HOW TO ACCESS

Tool	Description	How to Access
Report Server Web service URL	<p>Use this tool to browse content in the report catalog in a generic item navigation page.</p> <p>For more information, see Report Server Web Service.</p>	Browser
Web Portal	<p>(Native mode only) Use this tool to administer a single report server instance from a remote location over an HTTP connection. You can do the following:</p> <ul style="list-style-type: none"> View, search, print, and subscribe to reports. Create, secure, and maintain the folder hierarchy to organize items on the server. Configure role-based security that determines access to items and operations. Configure report execution properties, report history, and report parameters. Create report models that connect to and retrieve data from a Microsoft SQL Server Analysis Services data source or from a SQL Server relational data source. Set model item security to allow access to specific entities in the model, or map entities to predefined clickthrough reports that you create in advance. Create shared schedules and shared data sources to make schedules and data source connections more manageable. Create data-driven subscriptions that roll out reports to a large recipient list. Create linked reports to reuse and repurpose an existing report in different ways. Launch Report Builder to create reports that you can save and run on the report server. For more information, see Web portal (SSRS Native Mode). 	Browser

Tool	Description	How to Access
RS Utility	<p>This tool is a script host that you can use to perform scripted operations. Use this tool to run Visual Basic scripts that copy data between report server databases, publish reports, create items in a report server database, and more.</p> <p>For more information, see Report Server Command Prompt Utilities (SSRS).</p>	Command prompt

See Also

- [Reporting Services Report Server](#)
- [Reporting Services Concepts \(SSRS\)](#)
- [Reporting Services \(SSRS\)](#)

Reporting Services Data Alerts

11/28/2018 • 19 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2016) SQL Server Reporting Services (2017) SharePoint Power BI Report Server

For content related to previous versions of SQL Server Reporting Services (SSRS), see [SQL Server 2014 Reporting Services](#).

SQL Server Reporting Services data alerts are a data driven alerting solution that helps you be informed about report data that is interesting or important to you, and at a relevant time. By using data alerts you no longer have to seek out information, it comes to you.

Data alert messages are sent by email. Depending on the importance of the information, you can choose to send messages more or less frequently and only when results change. You can specify multiple email recipients and this way keep others informed to enhance efficiency and collaboration.

NOTE

Reporting Services integration with SharePoint is no longer available after SQL Server 2016.

Data Alerts Architecture and Workflow

The following summarizes the key areas of Reporting Services data alerts:

- **Define and save data alert definitions**-you run a report, create rules that identify interesting data values, define a recurrence pattern for sending the data alert message, and specify the recipients of the alert message.
- **Run data alert definitions**-Alerting service processes alert definitions at a scheduled time, retrieves report data, creates data alert instances based on rules in the alert definition.
- **Deliver data alert messages to recipients**-Alerting service creates an alert instance and sends an alert message to recipients by email.

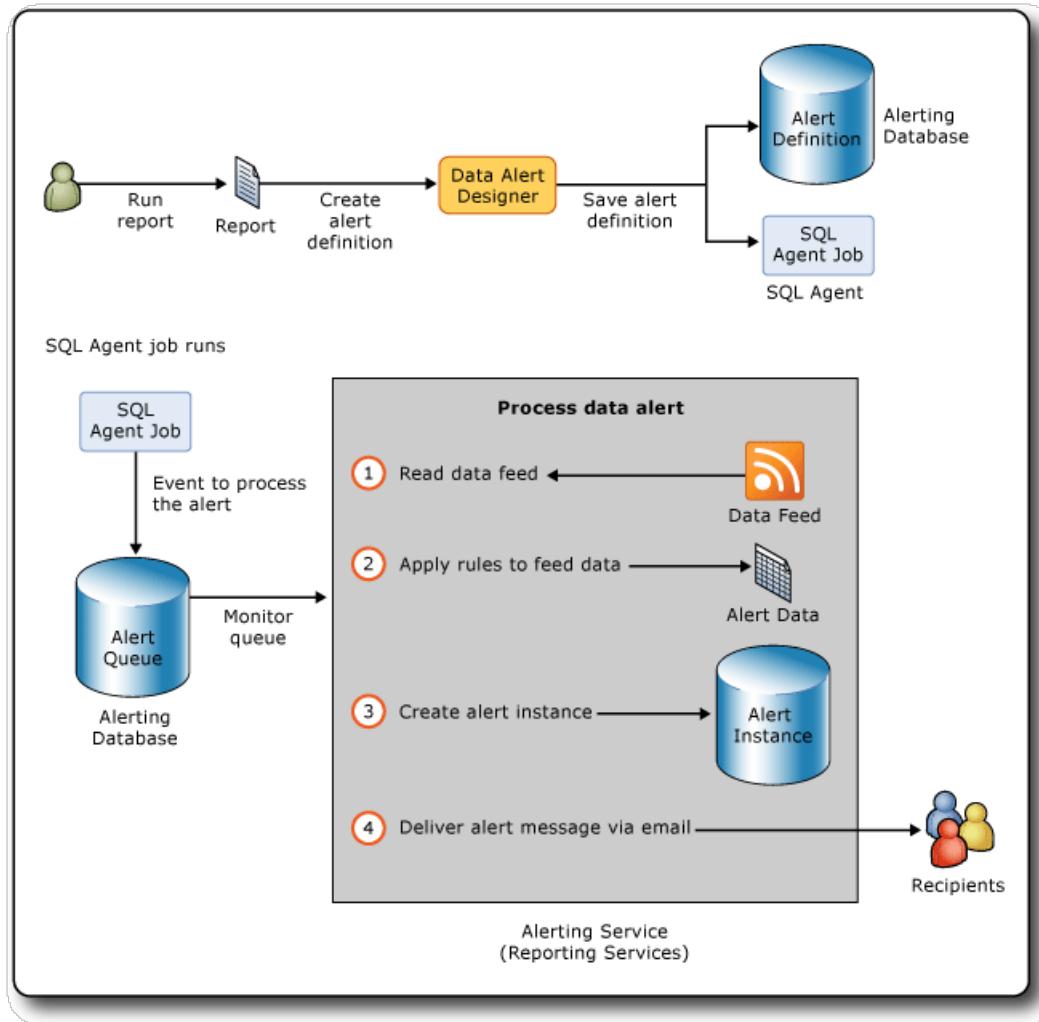
In addition, as a data alert owner you can view information about your data alerts and delete and edit your data alert definitions. An alert has only one owner, the person who created it.

Alerting administrators, users with SharePoint Manage Alerts permission, can manage data alerts at the site level. They can view lists of alerts by each site user and delete alerts.

Reporting Services data alerts are different from SharePoint alerts. You can define SharePoint alerts on any document type, including reports. SharePoint alerts are sent when the document changes. For example, you add a column to a table in a report. In contrast, data alerts are sent when the data shown in a report satisfied rules in the alert definitions. The rules typically reference the data that displays in a report.

By creating data alerts on reports, you can monitor changes in report data and send data alert messages by email when report data follow rules that define data of interest to you and others, and at intervals that meet your business needs. You can also run data alerts on demand. If you have SharePoint Create Alert permission, you can create alerts on any report that you have permissions to view. You can create multiple alerts on a report and multiple users can create the same or different alerts on a report. To collaborate with others, you can specify them as the recipients of alert messages in data alert definitions that you create.

The following diagram shows the workflow of creating and saving a data alert definition, creating a SQL Agent job to begin processing an instance of the data alert, and sending data alert messages that contain the report data that triggered the alert to one or more recipients by email.



Reports Supported by Data Alerts

You can create data alerts on all types of professional reports that are written in the report definition language (RDL) and created in Report Designer or Report Builder. Reports that include data regions such as tables and charts, reports with subreports, and complex reports with multiple parallel column groups and nested data regions. The only requirements are the report includes at least one data region of any type and the report data source is configured to use stored credentials or no credentials. If the report has no data regions, you cannot create an alert on it.

You cannot create data alerts on reports created with Power View.

When you install Reporting Services in native mode or SharePoint mode or use the standalone version of Report Builder, you can save reports to a report server, your computer, or a SharePoint library. To create data alerts on reports, the reports must be saved or uploaded to a SharePoint library. This means that you cannot create alerts on reports saved to a report server in native mode or your computer. Also, you cannot create alerts embedded in custom applications.

Reporting Services supports a variety of credential types in reports. You can create data alerts on reports with data source configured to use stored credentials, or no credentials. You cannot create alerts on reports configured to use integrated security credentials or prompt for credentials. The report is run as part of processing the alert definition and the processing fails without credentials. For more information, see the following:

- [Specify Credential and Connection Information for Report Data Sources](#)

- [Roles and Permissions \(Reporting Services\)](#)
- [Authentication with the Report Server](#)

Run Reports

The first step in creating a data alert definition is to locate the report you want in the SharePoint library, and then run the report. If a report contains no data when you run it, you cannot create an alert on the report at that time.

If the report is parameterized, you specify the parameter values to use when you run the report. The parameter values will be saved in the data alert definitions that you create on a report. The values are used when the report is rerun as a step in processing the data alert definition. If you want to change the parameter values you need to rerun the report with those parameter values and create an alert definition on that version of the report.

Create Data Alert Definitions

The Reporting Services data alerts feature includes the Data Alert Designer, which you use to create data alert definitions.

To create a data alert definition, you run the report and then open Data Alert Designer from the SharePoint Report Viewer **Actions** menu. The report data feeds for the report are generated and the first 100 rows in the data feed display in a data preview table in Data Alert Designer. All the data feeds from a report are cached as long you are working on the alert definition in Data Alert Designer. The caching enables you to switch quickly between data feeds. When you reopen an alert definition in Data Alert Designer, the data feeds are refreshed.

Data alert definitions consist of rules and clauses that report data must satisfy to trigger a data alert message, a schedule that defines the frequency to send the alert message and optionally the dates to start and stop sending the alert message, information such the Subject line and a description to include in the alert message, and the recipients of the message. After you create an alert definition, you save it to the SQL Server alerting database.

Save Data Alert Definitions and Alerting Metadata

When you install Reporting Services in SharePoint mode, the SQL Server alerting database is automatically created.

Data alert definitions and alerting metadata are saved in the alerting database. By default, this database is named ReportingServices<GUID>_Alerting.

When you save the data alert definition, alerting creates a SQL Server Agent job for the alert definition. The job includes a job schedule. The schedule is based on the recurrence pattern you define on the alert definition. Running the job initiates the processing of the data alert definition.

Process Data Alert Definitions

When the schedule of the SQL Server Agent job starts the processing of the alert definition, the report is run to refresh the report data feeds. The alerting service reads the data feeds and applies the rules that the data alert definitions specify to the data values. If one or more data values satisfy the rules, a data alert instance is created and a data alert message with the alert results is sent to all recipients by email. The results are rows of report data that satisfied all rules at the time the alert instance was created. To prevent multiple alert messages with the same results, you can specify that messages are sent only when the results change. In this case, an alert instance is created and saved to the alerting database, but no alert message is generated. If an error occurs, the alert instance is also saved to the alerting database and an alert message with the details about the error is sent to recipients. The Diagnostics and Logging section later in this topic has more information about logging and troubleshooting.

Send Data Alert Messages

Data alert message are sent by email.

The **From** line contains a value provided by the Reporting Services email delivery configuration. The **To** line lists the recipients that you specified when you created the alert in Data Alert Designer.

Besides the email Subject line, which you specify in Data Alert Designer, the data alert message includes the following information:

- The name of the person who created the data alert definition.
- If you provided a description in the alert definition, it displays at the top of the email text.
- The alert results, consisting of the rows in the report data feed that satisfy the rules specified in the alert definition.
- A link to the report that the alert definition is built upon.
- The rules in the alert definition.
- The parameters and values that you used to run the report.
- The contextual values from report items that are outside of the report data regions.

If a data alert instance or data alert message cannot be created an error message is sent to all recipients. Instead of the alert results, the message includes an error description.

For more information, see [Data Alert Messages](#).

Install Data Alerts

The data alerts feature is available only when Reporting Services is installed in SharePoint mode. When you install Reporting Services in SharePoint mode, setup automatically creates the alerting database that stores data alert definitions and alerting metadata, and two SharePoint pages for managing alerts and adds Data Alert Designer to the SharePoint site. There are no special steps to perform or options to set for alerting during installation.

If you want to learn more about installing Reporting Services in SharePoint mode, including the Reporting Services shared service that is new in SQL Server 2012 (11.x) and Reporting Services service application that you must create and configure before you can use Reporting Services features, see [Install Reporting Services SharePoint Mode for SharePoint 2010](#) in MSDN library.

As the diagram earlier in the topic shows, data alerts use SQL Server Agent jobs. To create the jobs, SQL Server Agent must be running. You might have configured SQL Server Agent to start automatically when you installed Reporting Services. If not, you can start SQL Server Agent manually. For more information, see [Configure SQL Server Agent](#) and [Start, Stop, Pause, Resume, Restart the Database Engine, SQL Server Agent, or SQL Server Browser Service](#).

You can use the **Provision Subscriptions and Alerts** page in SharePoint Central Administration to find out whether SQL Server Agent is running and create and download customized Transact-SQL scripts that you then run to grant permissions to SQL Server Agent. It can also generate the Transact-SQL scripts by using PowerShell. For more information, see [Provision Subscriptions and Alerts for SSRS Service Applications](#).

Configure Data Alerts

Starting in SQL Server 2012 (11.x) the settings for Reporting Services features, including data alerts, are distributed between the report server configuration file (rsreportserver.config) and a SharePoint configuration database whenever you install Reporting Services in SharePoint mode. When you create the service application as a step in installing and configuring Reporting Services, the SharePoint configuration database is automatically created. For more information, see [RsReportServer.config Configuration File](#) and [Reporting Services Configuration Files](#).

The settings for Reporting Services data alerts include the intervals for cleaning up alerting data and metadata and the number of retries when sending data alert messages by email. You can update the configuration file and

the configuration database to use different values for data alert settings.

You update the report server configuration file directly. You update the SharePoint configuration database by using Windows PowerShell cmdlets.

The following table lists the configuration elements for data alerts, their default values, descriptions, and locations.

SETTING	DEFAULT VALUE	DESCRIPTION	LOCATION
AlertingCleanupCycleMinutes	20	Number of minutes between starts of the cleanup cycle.	Report Server Configuration File
AlertingExecutionLogCleanupMinutes	10080	Number of minutes to keep execution log entries.	Report Server Configuration File
AlertingDataCleanupMinutes	360	Number of minutes to keep temporary data.	Report Server Configuration File
AlertingMaxDataRetentionDays	180	The number of days until alert execution metadata, alert instances, and execution results is deleted.	Report Server Configuration File
MaxRetries	3	Number of times to retry processing of data alerts.	Service Configuration Database
SecondsBeforeRetry	900	Number of seconds to wait before each retry.	Service Configuration Database

By default, the MaxRetries and SecondsBeforeRetry settings apply to all events that data alerts fire. If you want more granular control of retries and retry delays, you can add elements for any and all event handlers that specify different MaxRetries and SecondsBeforeRetry values.

Event Handlers and Retry

The event handlers are:

EVENT HANDLER	DESCRIPTION
FireAlert	You click Run in Data Alert Manager to initiate immediate processing of an alert definition.
FireSchedule	SQL Server Agent launches the job schedule for an alert definition.
CreateSchedule	You create a data alert definition and a SQL Server Agent job schedule is created based on the frequency interval specified in the alert definition.
UpdateSchedule	You update the frequency interval of the data alert definition and the SQL Server Agent job schedule is updated.
DeleteSchedule	You delete the data alert definition and its SQL Server Agent job is deleted.

EVENT HANDLER	DESCRIPTION
GenerateAlert	The alerting runtime processes the report data feed, applies the rules specified in the data alert definition, determines whether to create an instance of the data alert, and if needed creates an instance of the data alert.
DeliverAlert	The runtime creates the data alert message and sends it to all recipients by email.

The following table summarizes the event handlers and when retry will fire:

ERROR CATEGORY	<	<	EVENT TYPE	>	>	>	
	FireAlert	FireSchedule	CreateSchedule	UpdateSchedule	DeleteSchedule	GenerateAlert	DeliverAlert
Out of memory	X	X	X	X	X	X	X
Thread abort	X	X	X	X	X	X	X
SQL Agent is not running	X		X	X	X		
Transient. Mostly due to connection problems, timeouts, and locks.	X	X	X	X	X	X	X
IOException							X
WebException							X
SocketException							X
SMTPException (*)							X

(*) SMTP errors that will trigger a retry:

- SmtpStatusCode.ServiceNotAvailable
- SmtpStatusCode.MailboxBusy
- SmtpStatusCode.MailboxUnavailable

Disable Data Alerts

If you want to disable the data alert feature, you update the Service section of the configuration file. The

following code shows Service section of the configuration file.

```
<Service>
<IsSchedulingService>True</IsSchedulingService>
<IsNotificationService>True</IsNotificationService>
<IsEventService>True</IsEventService>
<IsAlertingService>True</IsAlertingService>
...
</Service>
```

To disable alerting, change True to False in `<IsAlertingService>True</IsAlertingService>`.

Permissions for Data Alerts

Before you can create data alerts on reports, you must have permission to run the report and create alerts on the SharePoint site. To learn more about report permissions, see the following.

- [Generating Data Feeds from Reports \(Report Builder and SSRS\)](#)
- [Set Permissions for Report Server Items on a SharePoint Site \(Reporting Services in SharePoint Integrated Mode\)](#)

Reporting Services data alerts support two permission levels: information worker and alerting administrator. The following table lists the related SharePoint permissions and user tasks.

USER TYPE	SHAREPOINT PERMISSION	TASK DESCRIPTION
Information worker	View Items	View items such as reports and create data alerts on the reports. Edit and delete alerts.
	Create Alerts	
Alerting administrator	Manage Alerts	View a list of all data alerts saved on the SharePoint site and delete alerts.

Diagnostics and Logging

Data alerts provides a number of ways to help information workers and administrators keep track of alerts and understand why alerts failed and help administrators make use of logs to learn which alert messages were sent to whom, number of alert instances sent, and so forth.

Data Alert Manager

Data Alert Manager lists alert definitions and error information that help information workers and alerting administrators understand why the failure occurred. Some common reasons for failure include:

- The report data feed changed and columns that are used in the data alert definition rules are no longer included in the data feed.
- Permission to view the report was revoked.
- The data type in the underlying data source changed and the alert definition is no longer valid.

Logs

Reporting Services provides a number of logs that can help you learn more the reports that are run when

processing data alert definitions, the data alert instances that are created and so forth. Three logs are particularly useful: the alerting execution log, the report server execution log, and the report server trace log.

For information about other Reporting Services logs, see [Reporting Services Log Files and Sources](#).

Alerting Execution Log

The alerting runtime service writes entries in the ExecutionLogView table in the alerting database. You can query the table or run the following stored procedures to get richer diagnostic information about the data alerts saved to the alerting database.

- ReadAlertData
- ReadAlertHistory
- ReadAlertInstances
- ReadEventHistory
- ReadFeedPollHistory
- ReadFeedPools
- ReadPollData
- ReadSentAlerts

You can use SQL Agent to run the stored procedure on a schedule. For more information, see [SQL Server Agent](#).

Report Server Execution Log

Reports are run to generate the data feeds that data alert definitions are built upon. The report server execution log in the report server database captures information each time the report is run. You can query the ExecutionLog2 view in the database for detailed information. For more information, see [Report Server ExecutionLog and the ExecutionLog3 View](#).

Report Server Trace Log

The report server trace log contains highly detailed information for report server service operations, including operations performed by the report server Web service and background processing. Trace log information might be useful if you are debugging an application that includes a report server, or investigating a specific problem that was written to the event log or execution log. For more information, see [Report Server Service Trace Log](#).

Performance Counters

Data alerts provide their own performance counters. All but one performance counter is related to an event that is part of the alerting runtime service. The performance counter related to the event queue tells the length of the queue of all active events.

EVENT OR EVENT QUEUE	PERFORMANCE COUNTER
ALERTINGQUEUESIZE	Alerting: event queue length
FireAlert	Alerting: events processed - FireAlert
FireSchedule	Alerting: events processed - FireSchedule
CreateSchedule	Alerting: events processed - CreateSchedule
UpdateSchedule	Alerting: events processed - UpdateSchedule

EVENT OR EVENT QUEUE	PERFORMANCE COUNTER
DeleteSchedule	Alerting: events processed - DeleteSchedule
GenerateAlert	Alerting: events processed - GenerateAlert
DeliverAlert	Alerting: events processed - DeliverAlert

Reporting Services provides performance counters for other Reporting Services features. For more information, see [Performance Counters for the ReportServer:Service and ReportServerSharePoint:Service Performance Objects](#), [Performance Counters for the MSRS 2011 Web Service and MSRS 2011 Windows Service Performance Objects \(Native Mode\)](#), and [Performance Counters for the MSRS 2011 Web Service SharePoint Mode and MSRS 2011 Windows Service SharePoint Mode Performance Objects \(SharePoint Mode\)](#).

Support for SSL

Reporting Services can use the HTTP SSL (Secure Sockets Layer) service to establish encrypted connections to a report server or SharePoint site.

The alerting runtime service and data alerts user interface support SSL and works similarly whether you use SSL or HTTP; however, there are some subtle differences. When the data alert definition is created using and SSL connection, the URL that links back to the SharePoint library from the data alert message also uses SSL. You can identify the SSL connection because it uses HTTPS instead of HTTP in its URL. Likewise, if the data alert definition was created using an HTTP connection, the link back to the SharePoint site uses HTTP. Whether the alert definition was created using SSL or HTTP, the experience for users and alerting administrators are identical when using Data Alert Designer or Data Alert Manager. If the protocol (HTTP or SSL) should change between the time that the alert definition was created and then updated and resaved, the original protocol is kept and used in link URLs.

If you create a data alert on a SharePoint site that is configured to use SSL and then remove the SSL requirement the alert continues to work on the site. If the site is deleted, the default zone site is used instead.

Data Alert User Interface

Data alerts provide SharePoint pages for managing alerts and a designer for creating and editing data alert definitions.

- **Data Alert Designer** in which you create or edit data alert definitions. For more information, see [Data Alert Designer](#), [Create a Data Alert in Data Alert Designer](#) and [Edit a Data Alert in Alert Designer](#).
- **Data Alert Manager** in which you view lists of data alerts, delete alerts, and open alerts for editing. Data Alert Manager comes in two versions: one for users to manage the alerts they created, and one for administrators to manage alerts that belong to site users.

For more information about managing data alerts that you created, see [Data Alert Manager for SharePoint Users](#) and [Manage My Data Alerts in Data Alert Manager](#).

For more information about managing all data alerts on a site, see [Data Alert Manager for Alerting Administrators](#) and [Manage All Data Alerts on a SharePoint Site in Data Alert Manager](#).

- **Provision Subscriptions and Data Alerts** in which you find out whether Reporting Services can use SQL Server Agent for data alerts and download scripts that allow access to SQL Server Agent. For more information, see [Provision Subscriptions and Alerts for SSRS Service Applications](#).

Globalization of Data Alerts

Certain script such as Arabic and Hebrew are written right to left. Data alerts support right-to-left scripts as well as left-to-right scripts. Data alerts detect culture and alter the appearance and behavior of the user interface and the layout of data alert messages accordingly. The culture is derived from the regional setting of the operating system on the user's computer. The culture is saved each time you update and then resave the data alert definition.

Whether data satisfies the rules in the alert definition can be affected by the culture in the alert definition. String comparisons are most commonly affected by culture specific rules.

Determining whether report data satisfies the rules in the alert definition can be affected by the culture in the alert definition. This most commonly occurs in strings. For example, in an alert definition with the German culture, a rule that compares the English letter "o" and the German letter "ö" would not be satisfied. In the same alert definition using the English culture the rule would be satisfied.

Data formatting is also based the culture of the alert definition. For example, if the culture uses a period as the decimal symbol, then the value displays as 45.67; whereas a culture that uses a comma as the decimal symbol, displays 45,67.

Depending on which data alert user interface you use, the support for right-to-left varies. Data Alert Designer supports right-to-left script in text boxes, but the layout of the designer is not right to left. Its layout is left to right like other tools. If an alert definition, created with right-to-left text orientation, and then edited in a left-to-right environment, the right-to-left text orientation is preserved when you save the alert definition. Data Alert Manager behaves the same as a SharePoint page. Its layout is right-to left, just like other SharePoint pages. Data alert messages that are based on right-to-left data alert definitions, display message text right to left and the message layout is left to right.

Related Tasks

- [Save a Report to a SharePoint Library \(Report Builder\)](#)
- [Create a Data Alert in Data Alert Designer](#)
- [Edit a Data Alert in Alert Designer](#)
- [Manage My Data Alerts in Data Alert Manager](#)
- [Manage All Data Alerts on a SharePoint Site in Data Alert Manager](#)
- [Grant Permissions to Users and Alerting Administrators](#)

See Also

[Data Alert Designer](#)

[Data Alert Manager for Alerting Administrators](#)

[Data Alert Manager for SharePoint Users](#)

More questions? [Try asking the Reporting Services forum](#)

Data Alert Designer

11/28/2018 • 12 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2016) SQL Server Reporting Services (2017) SharePoint Power BI Report Server

For content related to previous versions of SQL Server Reporting Services (SSRS), see [SQL Server 2014 Reporting Services](#).

You create and edit data alert definitions in Data Alert Designer. An alert definition is a collection of metadata, including the report data that you are interested in, the rules that report data must satisfy to create data alert instances and send data alert messages, the recipients of the alert message, and so forth.

NOTE

Reporting Services integration with SharePoint is no longer available after SQL Server 2016.

To create an alert definition you do a number of related tasks:

- Select the report and the report data feed that includes data that you want to use.
- Define the rules and clauses that cause an alert to be sent. The rules can be simple or complex, using multiple clauses combined by AND operators.
- Define the frequency that the alert message is sent and the date and time the alert starts and stops. Alert messages can be sent only when results change.
- Specify the email addresses of alert message recipients.
- Customize the **Subject** line of the alert message.
- Provide a description of alert to include in alert message.

NOTE

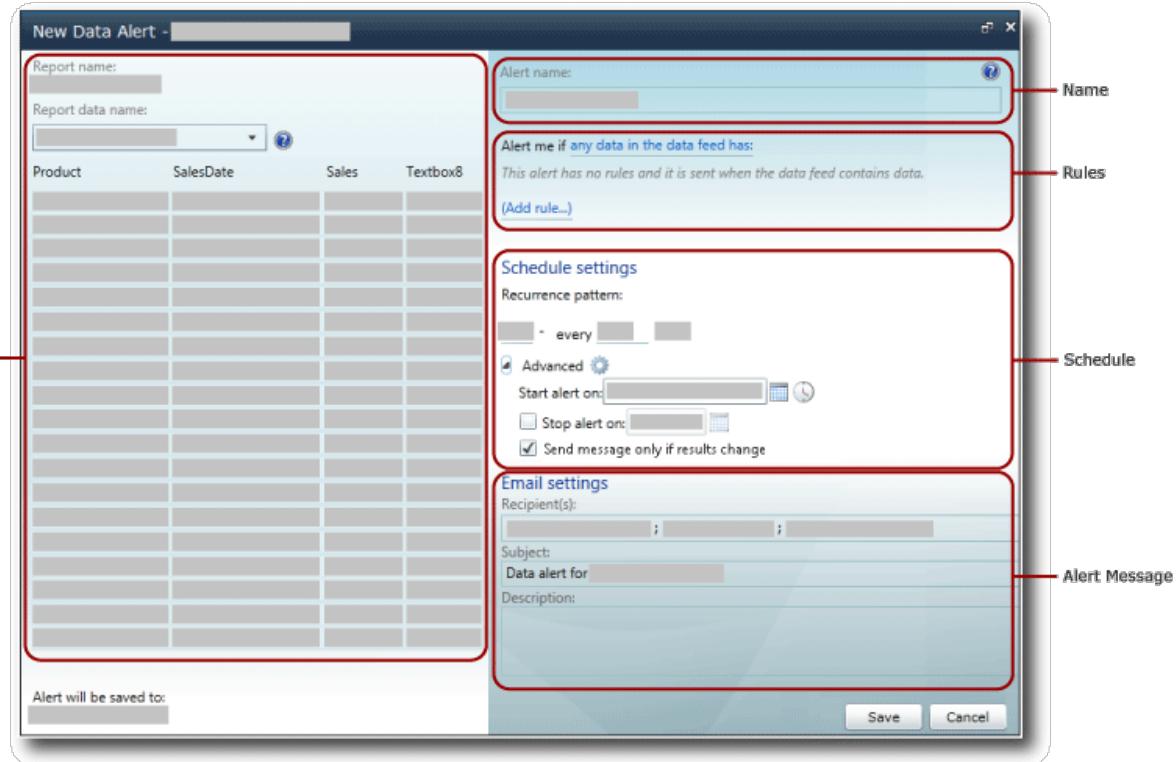
Because the Reporting Services data alerts feature is available only when you install Reporting Services in SharePoint mode, the report on which you want to create an alert must be saved, deployed, or uploaded to a SharePoint document library.

Data alerts cannot be created on reports that use Windows Integrated authentication or prompts for credentials. The reports must use stored credentials. For more information, see [Specify Credential and Connection Information for Report Data Sources](#).

To open Data Alert Designer, you click the **New Data Alert** option on the **Actions** menu on the report toolbar. If you do not see the **New Data Alert** option, the report is not configured to use stored credentials. You can update the credential type by updating the report data source from the SharePoint library.

Data Alert Designer User Interface

The Data Alert Designer is divided into areas. The area where you select the report data feed, the area where you create simple or complex conditions by adding rules to conditions, and so on. The following picture shows the areas in Data Alert Designer.



Alert Data

When you open Data Alert Designer, it generates and makes available all the data feeds from the report and the **Report data name** drop-down list contains the names of the feeds. The data feeds are cached in memory while you are creating the alert definition and the table that display the data feed data is populated quickly when you switch between data feeds to explore the report data.

The first step in creating a data alert definition is to select the report data feed that contains the data that you want the alert to monitor. Reports can have zero or multiple data feeds. If a report has no data feeds, you cannot create alerts on it. A data feed can be generated by any data region, including all types of charts, gauges, indicators as well as tables, matrices, and lists.

If the report is parameterized and you do not see the data and columns that you expect in the report data feed, rerun the report using the appropriate parameter values. The columns and values must be present in the report to be included in the data feed.

Depending on the layout of the report, it might not be intuitive how many data feeds a report has, nor what data is included in which data feed. The Reporting ServicesAtom rendering extension generates the data feeds that you use with alerts. The Atom rendering extension provides report data as flattened rowsets, a tabular format in which all columns have the same number of rows. These rowsets are the contents of the data feeds. Because report layout is often complex and contains multiple peer or nested data regions, multiple data feeds are needed to make available all the report data.. For more information about how data feeds are generated from reports, see [Generating Data Feeds from Reports \(Report Builder and SSRS\)](#) and see [Generate Data Feeds from a Report \(Report Builder and SSRS\)](#).

When you choose a data feed, the data from the feed displays in a table with rows and columns in the alert data pane of Data Alert Designer. The metadata from the data source that the report uses or the report itself specifies the column names and the data feed populates the field list that you use to define rules in the data condition. The data feed also provides metadata such as the data types of table columns that restrict the values and comparison operators that you can use with fields when you create the rules.

Some reports have millions of rows of data. The table shows only the first 100 rows of data in the feed.

Alert Name

By default, the alert definition has the same name as the report. You can change the alert name to be more

meaningful. This makes it easier for you to manage your alerts, determining which alerts to update, delete and so on.

You can create multiple alerts on a report. It is possible to have multiple alert definitions with the same name, but it is recommended that you make alert names unique. It makes it easier to differentiate and manage alert definitions. You can view a list of all the alerts you created in Data Alert Manager. For more information, see [Data Alert Manager for Alerting Administrators](#) and [Manage My Data Alerts in Data Alert Manager](#).

Rules and Clauses

The scope of data changes and the in the alert rules define the data changes that trigger the alert. The scope of the data changes are as follow:

- **Any data has**-at least one value in the data satisfies the rules that the condition specifies.
- **No data has**-no value in the data satisfied the rules that the condition specifies.

A rule contains zero, one, or many clauses. Multiple rules are combined by the AND logical operator. A rule can include multiple clause combined by the OR operator if the column has the string data type. The following shows basic rules that use only one clause, multiple rules combined by using the AND operator, multiple rules that with one or more OR clauses.

Simple rules

- Net sales **is greater than** 100000
- Sales date **is after** 6/1/2010
- Company Name **is not** Contoso

Rules combined by AND operator

- Sales **is greater than** 1500.00
and Units Sold **is less than** 500
Return date **is before** 1/1/2010
- Sales **is greater than or equal to** 1500.00
and Return date **is after** 1/1/2010
and Units Sold **is greater than** 500
- Promotion name **contains** Spring
and Units Sold **is greater than** 500
and Returns **is** 0

Rules with OR clauses

- Last Name **is** Blythe
Or Petulescu
- **Or** Martin
- Return date **is after** 1/1/2010
and Sales Territory **is** Central
Or South

Or North

Depending on the data type of the field, Data Alert Designer provides different comparisons. Data Alert Designer provides comparisons that are tailored to the data type of the field to which values are compared. The following lists comparisons available for different data types. The **Boolean** data type is not supported in rules.

- Date time data type comparisons are: **is**, **is not**, **is before**, and **is after**
- Numeric data type comparisons are: **is**, **is not**, **is less than**, **is less than or equal to**, and **is greater than**, and **is greater than or equal to**
- String data type comparisons are: **is**, **is not**, and **contains**

When you create a rule, you specify whether to use to use a value or field in the comparison by choosing **Value Entry Mode** or **Field Selection Mode**. If you choose **Value Entry Mode**, you provide a list of values to compare to. A comparison that includes multiple OR clauses is very similar to the IN logical comparison in Transact-SQL, which is a list of values to test for a match. For more information, see [IN \(Transact-SQL\)](#).

If you choose **Field Selection Mode**, the comparison is between two fields, row by row. The two fields must have compatible data types (for example, two numeric fields) or the comparison is not valid. A list of fields displays automatically when you choose **Field Selection Mode**.

Data alerts without rules are also valid. This type of alert can be very useful. Imagine a scenario in which you want only to be notified when the report data feed has data. The data feed contains attendee information and the feed is empty until an attendee cancels. In this scenario, you would receive an alert, starting with the first cancellation.

You can delete individual rules and clauses.

Rules and clauses are included in the data alert message.

Schedule Settings

The schedule that you define for the data alert defines the recurrence pattern for sending the data alert message and when to start and stop sending the alert messages. The patterns are: once, minute, daily, and weekly.

Although an alert has only one schedule you can create complex recurrence patterns that meet most business needs by using these intervals. The following are examples of common recurrence patterns to use in schedules:

- **Daily every 10 day(s)** - sends alerts once a day, every 10 days.
- **Weekly every 2 week(s) on Monday** - sends alerts every two weeks on Mondays only.
- **Hourly every 12 hour(s)** - sends alerts every 12 hours.
- **Minute every 30 minute(s)** - sends alerts every 30 minutes.

The recurrence pattern specifies when the alert is sent. If the rules are met during the interval that the pattern specifies, the alert is not sent until the end of the interval.

If you want to receive a data alert message as soon as possible when report data follow the specified rules, you can schedule the alert to run often. When the report data does not change, you and other recipients might receive many redundant messages. If you want to receive messages only, when the results from applying the rules change, select the **Send message only if results change** option.

IMPORTANT

It is recommended that you do not use a recurrence pattern that is more frequent than daily unless you have an important business reason to do so. Processing data alert definition in real time is not a supported scenario. Processing data alert definitions too frequently impacts the performance of the report server and the overall Reporting Services deployment.

Email Settings

You specify the email addresses of recipients to receive data alert messages by email in the **Recipient(s)** option. Multiple email addresses are separated by semicolons, the same way that you do in Microsoft Office Outlook email messages. You can also specify distribution groups as recipients, which makes it easier and more efficient to manage the recipient list. If SharePoint can determine your email address when you are creating an alert definition, your email address is automatically added to the recipients list; otherwise, you need to explicitly add yourself as a recipient.

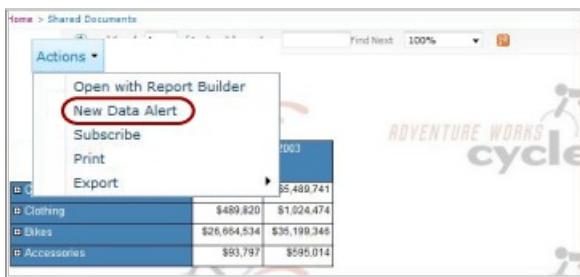
The default subject of the email is **Data alert for <alert name>**. You can change the subject to fit your needs.

You can also provide a description to include in the data alert message in the **Description** option. Including a description, especially if you have data alerts that are similar, will help you quickly differentiate and understand your alert messages. In addition to the alert message that is sent when report data satisfied the specified rules, an alert message is sent to all recipients when an error occurs. For more information, see [Data Alert Messages](#).

For more information about how the email is generated, see [Reporting Services Data Alerts](#).

Create a Data Alert Definition

If you are granted the SharePoint View Items and Create Alerts permissions you can create a data alert definition for any report that you have permission to view as long as the report uses stored credentials or no credentials. You run the report from a SharePoint library. The data that is available for you to use in Data Alert Designer comes from the report. If the report is parameterized, you might need to run the report using different parameter values to ensure the data that you are interested in appears in the report. After the report is open, you click the **New Data Alert** option on the **Actions** menu on the report toolbar to open Data Alert Designer. The following picture shows you how to open Data Alert Designer.



For more information, see [Create a Data Alert in Data Alert Designer](#).

Save a Data Alert Definition

Data Alert Designer displays the URL of the site where the data alert definition will be saved. Data alert definitions are always saved to the same site as the reports.

NOTE

The parameter values you chose to run the report are saved in the alert definition and will be used when report is rerun as a step in processing the alert definition. To use different parameter values, you must create a new alert definition.

Before the alert definition is saved, it is validated. You must correct any errors before the alert definition can be saved successfully. For more information, see [Create a Data Alert in Data Alert Designer](#).

Edit a Data Alert Definition

After you save your data alert definition, you can reopen and then edit it in Data Alert Designer. You can add, change, or delete rules and clauses and change the schedule and email settings. If the report data feed that the alert uses has changed and no longer provides the fields that the alert rules reference or the data types or other metadata of the fields have changed, the alert definition is no longer valid, and you must correct it before you can

resave it. If you want to use a different data feed, you must create a new alert definition.

To edit a data alert definition, you right click it in Data Alert Manager and click **Edit**. The following picture shows you the context menu on a data alert in Data Alert Manager.



For more information, see [Edit a Data Alert in Alert Designer](#).

Related Tasks

This section lists procedures that show you how to create and edit alerts.

- [Edit a Data Alert in Alert Designer](#)
- [Create a Data Alert in Data Alert Designer](#)

See Also

[Reporting Services Data Alerts](#)

[Data Alert Manager for Alerting Administrators](#)

More questions? Try asking the [Reporting Services forum](#)

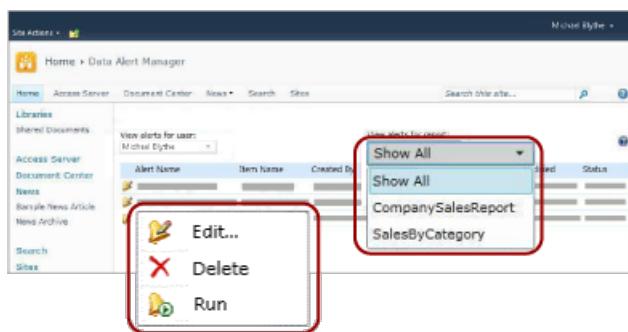
Data Alert Manager for SharePoint Users

11/15/2018 • 4 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2016) SQL Server Reporting Services (2017) SharePoint Power BI Report Server

For content related to previous versions of SQL Server Reporting Services (SSRS), see [SQL Server 2014 Reporting Services](#).

Reporting Services provides Data Alert Manager for SharePoint information workers to manage the data alerts. They can view information about the alerts they created, delete alerts, open alert definitions for editing, and run alerts on demand. They can choose to view alerts for a single report only or alerts for all reports. The following picture shows the features available to information workers in Data Alert Manager.



NOTE

Reporting Services integration with SharePoint is no longer available after SQL Server 2016.

When a SharePoint site is enabled for data alerts, two SharePoint pages, MyDataAlerts.aspx and SiteDataAlerts.aspx are created and added to the SharePoint site. MyDataAlerts.aspx is Data Alert Manager for SharePoint information workers. Information workers open Data Alert Manager from the right-click menu of reports on which they created alerts.

You can also open Data Alert Manager directly by using a URL. The following shows the syntax of the URL:

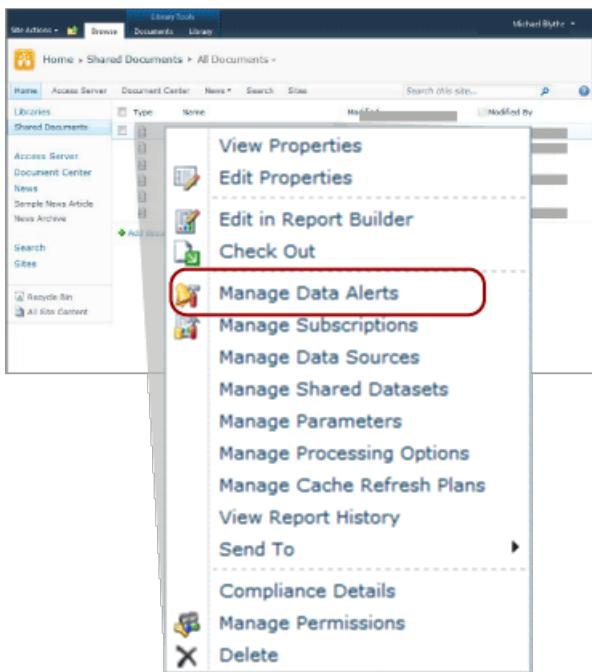
```
https://<site name>/_layouts/ReportServer/MyDataAlerts.aspx
```

NOTE

Before you can use the Reporting Services alerting features, an administrator must grant you permissions. For information about the required permissions, see [Reporting Services Data Alerts](#).

View Data Alert Information

You can view a list of data alerts that you created in Data Alert Designer. To open Data Alert Manager, you right-click a report published to a SharePoint library. The following picture shows the **Manage Data Alerts** option on the report right-click menu.



Data Alert Manager includes a table that lists the alert name, report name, your name as the creator of the alert definition, the number the alert message was sent, the last time the alert was run, the last time the alert definition was modified, and the status of the latest alert message. If the alert message cannot be generated or sent, the status column contains information about the error and helps you troubleshoot the alert. For more information, see [Manage My Data Alerts in Data Alert Manager](#).

The following table shows sample data from a table in Data Alert Manager. When an error occurs, the error message and the identifier of the entry in the log (a GUID) are included in the **Status** field in the table.

ALERT NAME	REPORT NAME	CREATED BY	SENT ALERTS	LAST RUN	LAST MODIFIED	STATUS
SalesQTR	SalesByTerritoryAndQTR	Lauren Johnson	4	6/12/2011	6/1/2011	Last alert ran successfully and alert was sent.
UnitsSold	ProductsSales ByQTR	Lauren Johnson	2	7/1/2011	6/28/2011	Last alert ran successfully, but the data was unchanged and no alert was sent.
TopPromotion	PromotionTracking	Lauren Johnson	0		5/23/2011	Alert created.

Delete Data Alerts

You delete alerts definitions from Data Alert Manager. As information worker you can delete the alert definitions that you created. You cannot delete alert definitions created by others. For more information, see [Manage My Data Alerts in Data Alert Manager](#).

When you delete an alert definition, it is deleted permanently. If you want only to pause alert messages, you should change the recurrence pattern or the start or stop date in the alert definition. For more information, see [Edit a Data Alert in Alert Designer](#).

Edit Data Alerts

As an information worker, you open your alert definitions for edit from Data Alert Manager. You can edit the alert definitions that you created, but not those created by others. When you right-click the alert definition and click **Edit** the Data Alert Designer opens, displaying the alert definition. For more information see, [Data Alert Designer](#) and [Edit a Data Alert in Alert Designer](#).

Run Data Alerts

Data Alert Manager includes information about the last time the alerting service processed the data alert definition and the number of times the data alert message has been sent. You might want to run and send the alert message immediately, instead of waiting for the time the schedule specifies. When you run an alert from Data Alert Manager, the alert schedule is overwritten and processing of the alert definition starts within one to five minutes, depending on the time needed to run the report and how busy the report server is at the time that you chose to run the alert. However, if you specified a message be sent only if results change and the results did not change, then no message is created or sent. For more information, see [Manage My Data Alerts in Data Alert Manager](#).

NOTE

After you click the **Run** option, it takes a few seconds to update the value of the **Status** column to indicate the alert is processing. If you click the **Run** option multiple times, the alert will be processed multiple times. This consumes resources on the report server unnecessarily and might impact performance of the report server. To see updated information about the alert, click the Web browser refresh button to check for status updates as well as other information about the alert.

Related Tasks

This section lists procedures that show you how to manage your alerts and edit your alert definitions.

- [Manage My Data Alerts in Data Alert Manager](#)
- [Edit a Data Alert in Alert Designer](#)

See Also

[Data Alert Designer](#)

[Create a Data Alert in Data Alert Designer](#)

[Reporting Services Data Alerts](#)

More questions? Try asking the [Reporting Services forum](#)

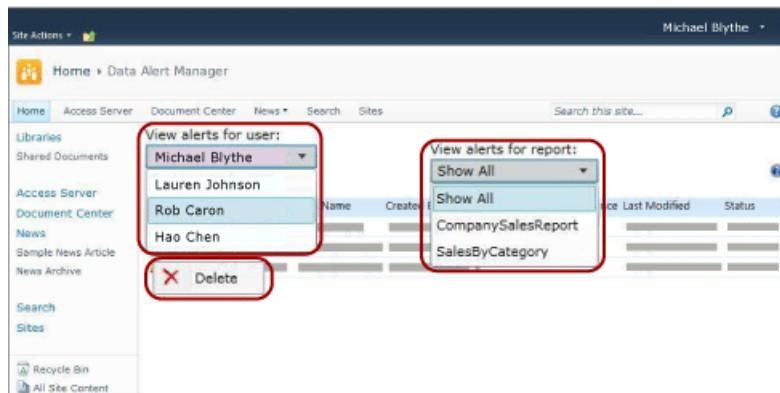
Data Alert Manager for Alerting Administrators

10/24/2018 • 3 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2016) SQL Server Reporting Services (2017)
SharePoint Power BI Report Server

For content related to previous versions of SQL Server Reporting Services (SSRS), see [SQL Server 2014 Reporting Services](#).

SQL Server Reporting Services provides Data Alert Manager for SharePoint alerting administrators to manage data alerts. Alerting administrators can view information about all alerts saved to the site and delete alerts. The following picture shows the features available to SharePoint alerting managers in Data Alert Manager.



NOTE

Reporting Services integration with SharePoint is no longer available after SQL Server 2016.

When the site is enabled for data alerts, two SharePoint pages, MyDataAlerts.aspx and SiteDataAlerts.aspx are created and added to the SharePoint site. SiteDataAlerts.aspx is Data Alert Manager for alerting administrators. Alerting administrators can open Data Alert Manager from the Site Settings SharePoint page. Alerting administrators must have SharePoint Manage Alerts permission to open Data Alert Manager.

You can also open Data Alert Manager directly by using a URL. The following shows the syntax of the URL:

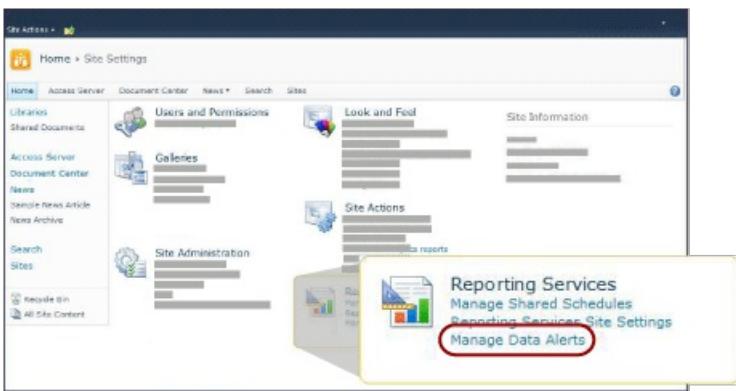
```
http://<site name>/_layouts/ReportServer/ SiteDataAlerts.aspx
```

NOTE

As an alerting administrator, you can grant permission to information workers to access the Reporting Services data alerts features. For more information about the required permissions, see [Reporting Services Data Alerts](#).

Viewing Data Alert Information

When Reporting Services is installed and configured in SharePoint, the Site Settings SharePoint page includes the **Reporting Services** options. Alerting administrators click the **Manage Data Alerts** option within Reporting Service to open Data Alert Manager. The following picture shows from where on the Site Settings page you open Data Alert Manager.



Data Alert Manager includes a table that lists the alert name, report name, the name of the alert owner, the number the alert message was sent, the last time the alert was run, the last time the alert definition was modified, and the status of the alert message. If the alert cannot be generated or generated or sent, the status column contains information about the error and helps you troubleshoot the alert. For more information, see [Manage All Data Alerts on a SharePoint Site in Data Alert Manager](#).

The following table shows sample data from a table in Data Alert Manager. When an error occurs, the error message and the identifier of the entry in the log (a GUID) are included in the **Status** field in the table.

ALERT NAME	REPORT NAME	CREATED BY	SENT ALERTS	LAST RUN	LAST MODIFIED	STATUS
SalesQTR	SalesByTerritoryAndQTR	Lauren Johnson	4	6/12/2011	6/1/2011	Last alert ran successfully and alert was sent.
UnitsSold	ProductsSales ByQTR	Michael Blythe	2	7/1/2011	6/28/2011	Last alert ran successfully, but the data was unchanged and no alert was sent.
InventoryCount	StockStatusBy QTR	Lauren Johnson	7	7/10/2011	7/2/2011	<error message>The log file contains detailed information about the error. Refer to the log entry with the identifier: <GUID>.
TopPromotion	PromotionTracking	Cristian Petculescu	0		5/23/2011	Alert created.

For more information see, [Manage All Data Alerts on a SharePoint Site in Data Alert Manager](#).

You can view all alerts created by site users. You choose a user and then choose whether to view all their alerts or only alerts for a specific report.

Delete Data Alerts

You delete alerts definitions from Data Alert Manager. Every data alert definition has an owner, the SharePoint user who created it. Owners can delete only the alert definitions that they created. For more information, see [Manage My Data Alerts in Data Alert Manager](#).

A SharePoint alerting administrators can list and then delete alert definitions created by all users of the site. For more information, see [Manage All Data Alerts on a SharePoint Site in Data Alert Manager](#)

After you delete the alert definition, no further alerts are sent. However, if you query the alerting database you might find that the alert definition still exists. The alerting service performs clean up on a schedule and the alert definition is deleted permanently in the next cleanup. The default cleanup interval is 20 minutes. This and other cleanup intervals are configurable. For more information, see [Reporting Services Data Alerts](#).

Related Tasks

This section lists a procedure that shows you how to manage your alerts.

- [Manage All Data Alerts on a SharePoint Site in Data Alert Manager](#)

See Also

[Reporting Services Data Alerts](#)

More questions? [Try asking the Reporting Services forum](#)

Data Alert Messages

10/24/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2016) SQL Server Reporting Services (2017)
SharePoint Power BI Report Server

For content related to previous versions of SQL Server Reporting Services (SSRS), see [SQL Server 2014 Reporting Services](#).

SQL Server Reporting Services data alerts deliver two types of data alert messages by email: Messages with data alert results and messages with error descriptions. Messages with results keep all recipients informed about changes in report data that is of common interest and important to business decisions. If for some reason an error occurs and the results are not available, the error message is sent instead.

The owner of the data alert definition also can view information about the data alert instance in Data Alert Manager. For more information, see [Data Alert Manager for SharePoint Users](#).

NOTE

Reporting Services integration with SharePoint is no longer available after SQL Server 2016.

Data Alert Messages

The following pictures show a data alert message with results and an alert message with an error description.

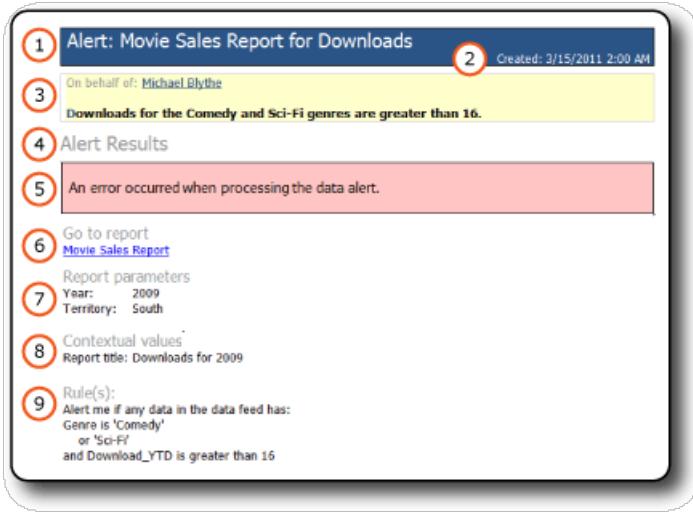
Results message

A screenshot of a data alert message window. The window has a title bar "Alert: Movie Sales Report for Downloads" and a status bar "Created: 3/15/2011 2:00 AM". The main content area is divided into sections:

- Section 1: Alert title "Alert: Movie Sales Report for Downloads".
- Section 2: Alert details "On behalf of: Michael Blythe" and "Downloads for the Comedy and Sci-Fi genres are greater than 16".
- Section 3: Alert Results table:

Date	Genre	Download YTD
3/1/2009	Comedy	17
4/13/2009	Sci-Fi	36
5/1/2009	Comedy	24
- Section 4: Alert links:
 - "Go to report" (Movie Sales Report)
 - "Report parameters": Year: 2009, Territory: South
 - "Contextual values": Report title: Downloads for 2009
 - "Rule(s)": Alert me if any data in the data feed has: Genre is 'Comedy' or 'Sci-Fi' and Download_YTD is greater than 16

Error message



The messages include the same types of information.

1. **On behalf of** contains the name of the person who created the data alert definition.
2. If you provided a description in the alert definition, it displays below **On behalf of**.
3. **Alert Results** display the rows in the report data feed that satisfy the rules specified in the alert definition in a tabular format or display an error description. There is no limit on the number of rows that displays.
4. **Go to report** is a link to the report that the alert definition is built upon. If the link is not valid because the report was moved or deleted, an error message displays.
5. **Rule(s)** lists the rules and clauses in the alert definition. This information helps you verify and understand the alert results and identify rules in the data alert definition that you might want to change to narrow or broaden results.
6. **Report parameters** list the parameters and parameter values that were used when the report was run. Parameters and parameter values help you understand the alert results.
7. **Contextual values** list the names and values of report items that are outside of the report data regions. The items are typically text boxes. For example, a text box with a constant value such as the subject or description of a report.

The only difference between the two message types is item 5, **Alert Results**. If an error occurs when a data alert instance or data alert message is created, **Alert Results** displays an error message that describes the problem. The error message, sent to all recipients, let them know that the alert results that they are expecting and might rely on to make business decisions are not available.

Related Tasks

This section lists procedures that show you how to create and edit the data alert definitions that provide much of the information that you see in data alert messages.

- [Create a Data Alert in Data Alert Designer](#)
- [Edit a Data Alert in Alert Designer](#)

See Also

[Data Alert Designer](#)
[Reporting Services Data Alerts](#)

More questions? Try asking the [Reporting Services forum](#)

Create a Data Alert in Data Alert Designer

11/28/2018 • 4 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2016) Power BI Report Server SharePoint

You create data alert definitions in Data Alert Designer. After you save the alert definitions, you can reopen, edit, and then resave them in Data Alert Designer. For information about editing alert definitions, see [Manage My Data Alerts in Data Alert Manager](#) and [Edit a Data Alert in Alert Designer](#).

NOTE

Reporting Services integration with SharePoint is no longer available after SQL Server 2016.

Create a data alert definition

1. Locate the SharePoint library that contains the report that you want to create a data alert definition for.
2. Click the report.

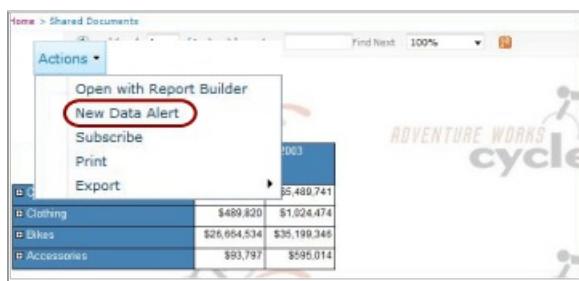
The report runs. If the report is parameterized, verify that the report shows the data that you want to receive alert messages about. If you do not see the columns or values you are interested in, you might want to rerun the report, using different parameter values.

NOTE

The parameter values you chose to run the report are saved in the alert definition and will be used when report is rerun as a step in processing the alert definition. To use different parameter values, you must create a new alert definition.

3. On the **Actions** menu, click **New Data Alert**.

The following picture shows the **Actions** menu.



The Data Alert Designer opens, showing the first 100 rows of the first data feed that the report generates in a table.

NOTE

If you do not see the **New Data Alert** option, the alerting service is not configured on the SharePoint site or the SQL Server edition does not include data alerts. For more information, see [Reporting Services SharePoint Service and Service Applications](#).

If the **New Data Alert** option is grayed, the report data source is configured to use integrated security credentials or prompt for credentials. To make the **New Data Alert** option available, you must update the data source to use stored credentials or no credentials.

The name of the data feed appears in **Report data name** drop-down list.

4. Optionally, select a different data feed in the **Report data name** drop-down list.

If no data feed is generated from the report, you cannot create an alert definition for the report. The layout of the report determines the content of each data feed. For more information see, [Generating Data Feeds from Reports \(Report Builder and SSRS\)](#).

5. Optionally, in the **Alert name** text box, update the default name to be more meaningful.

The default name of the alert definition is the name of the report. Alert definition names do not have to be unique, which can make it difficult to tell them apart when you later view the list of your alerts in Data Alert Manager. It is recommended that you use meaningful and unique names for your alert definitions.

6. Optionally, change the default data option from **any data in the data feed has** to **no data in the data feed has**.

7. Click **Add rule**.

A list of the columns in the data feed appears.

8. In the list, select the column that you want to use in the rule, and then select a comparison operator and enter the threshold value.

Depending on the data type of the selected column, different comparison operators are listed. If the column has a date data type, a calendar icon displays next to threshold value for the rule. You can enter a date by clicking a date in the calendar or typing the date.

Data Alert Designer provides two comparison modes: **Value Entry Mode** and **Field Selection Mode**.

The default mode is **Value Entry Mode**. You can add OR clauses only when you are in **Value Entry Mode** and are using the **is** comparison.

9. To add an OR clause, click the down-arrow, and then click **Value Entry Mode**.

10. Type the comparison value.

11. Optionally, click the ellipsis (...) again.

The ellipsis (...) appears on the line that contains the first clause.

An OR clause is added below and within the AND rule.

12. Optionally, click the down-arrow, select **Field Selection Mode**, and then select a column in the list.

You will notice that the ellipsis (...) that you click to add OR clauses has disappeared.

13. Optionally, click **Add rule** again to add additional rules.

The rules are combined by using the AND logical operator.

14. Select an option in the recurrence list. Depending on the type of recurrence, enter an interval.

15. Optionally, click **Advanced**.
16. Optionally, change the date that the alert message starts on by typing a different date or opening the calendar, and then clicking a date in the calendar.

The default start date is the current date.
17. Optionally, select the checkbox located next to **Stop alert on**, and then choose a date to stop the alert message.

By default, an alert message has no stop date.

NOTE

Stopping an alert message does not delete the alert definition. After you stop an alert message, you can restart it by updating the start and stop dates. For information about deleting alert definitions, see [Manage My Data Alerts in Data Alert Manager](#).

18. Optionally, clear the **Send message only if results change** checkbox.

If you send alert messages frequently, redundant information might not be welcome and you should not clear this checkbox.
19. Enter the email addresses of alert message recipients. Separate addresses with semicolons.

If the email address of the person who created the alert definition is available, it is added to the **Recipient(s)** box.
20. Optionally, in the **Subject** text box, update the Subject line of the alert message.

The default Subject is **Data alert for <data alert name>**.
21. Optionally, in the **Description** text box, type a description of the alert message.
22. Click **Save**.

See Also

[Data Alert Designer](#)

[Data Alert Manager for Alerting Administrators](#)

[Reporting Services Data Alerts](#)

More questions? [Try asking the Reporting Services forum](#)

Edit a Data Alert in Alert Designer

10/24/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO: ✓ SQL Server Reporting Services (2016) ✗ SQL Server Reporting Services (2017) ✓
SharePoint ✗ Power BI Report Server

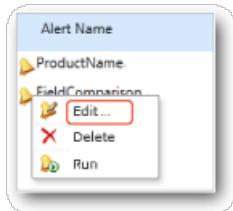
For content related to previous versions of SQL Server Reporting Services (SSRS), see [SQL Server 2014 Reporting Services](#).

You open the data alert definition that you want to edit from Data Alert Manager. Only the user that created the alert definition can edit it. For more information about opening Data Alert Manager, see [Manage My Data Alerts in Data Alert Manager](#).

NOTE

Reporting Services integration with SharePoint is no longer available after SQL Server 2016.

The following picture shows you the context menu on a data alert in Data Alert Manager.



The following procedure includes the steps to open the alert definition for editing in Data Alert Designer from Data Alert Manager.

To edit a data alert definition in Data Alert Designer

1. In Data Alert Manager, right-click the data alert definition that you want to edit and click **Edit**.

The alert definition opens in Data Alert Designer.

2. Update the rules, schedule settings, and email settings. For more information, see [Data Alert Designer](#) and [Create a Data Alert in Data Alert Designer](#).

NOTE

You cannot choose a different data feed. To use a different data feed, you must create a new data alert definition.

3. Click **Save**.

NOTE

If the report has changed and the data feeds generated from the report have changed the alert definition might no longer be valid. This occurs when a column that the alert definition references in its rules is deleted from the report or changes data type or the report is deleted or moved. You can open an alert definition that is not valid, but you cannot resave it until it is valid based on the current version of the report data feed that it is built upon. To learn more about how data feeds are generated from reports, see [Generating Data Feeds from Reports \(Report Builder and SSRS\)](#).

See Also

[Data Alert Manager for Alerting Administrators](#)

[Reporting Services Data Alerts](#)

More questions? [Try asking the Reporting Services forum](#)

Manage All Data Alerts on a SharePoint Site in Data Alert Manager

10/24/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2016) Power BI Report Server SharePoint

SharePoint alerting administrators can view a list of the data alerts that were created by any site user and information about the alerts. Alerting administrators can also delete alerts. The following picture shows the features available to alerting administrators in Data Alert Manager.

The screenshot shows the SharePoint Data Alert Manager interface. On the left, there's a navigation bar with links like Home, Access Server, Document Center, News, Search, and Sites. The main area has two dropdown menus: 'View alerts for user:' (with options Michael Blythe, Lauren Johnson, Rob Caron, Hao Chen) and 'View alerts for report:' (with options Show All, CompanySalesReport, SalesByCategory). Below these dropdowns is a table with columns Name, Created, Last Modified, and Status. A red box highlights the 'Delete' button at the bottom left of the alert list area.

NOTE

Reporting Services integration with SharePoint is no longer available after SQL Server 2016.

View a list of alerts created by a site user

1. Go to the SharePoint site where data alerts definitions are saved.
2. On the Home page, click **Site Actions**.
3. Scroll to the bottom of the list and click **Site Settings**.
4. Under **Reporting Services**, click **Manage Data Alerts**.
5. Click the down arrow by the **View alerts for user** list and select the user whose alerts you want to view.
6. Click the down arrow next to the **View alerts for report** list and select a specific alert to view, or click **Show All** to list all alerts created by the selected user.

A table lists the name, report name, name of the person who created the data alert, the number times the data alert was sent, the last time the data alert definition was modified, and the status of the data alert. If the data alert cannot be generated or sent, the status column contains information about the error and helps you troubleshoot the problem.

Delete an alert definition

- Right-click the data alert that you want to delete and click **Delete**.

NOTE

After you delete the alert, no further alert messages are sent. However, if you query the alerting database you might find that the alert definition still exists. The alerting service performs clean up on a schedule and the alert definition is deleted permanently in the next cleanup. The default cleanup interval is 20 minutes. This and other cleanup intervals are configurable. For more information, see [Reporting Services Data Alerts](#).

See Also

[Data Alert Manager for Alerting Administrators](#)

[Reporting Services Data Alerts](#)

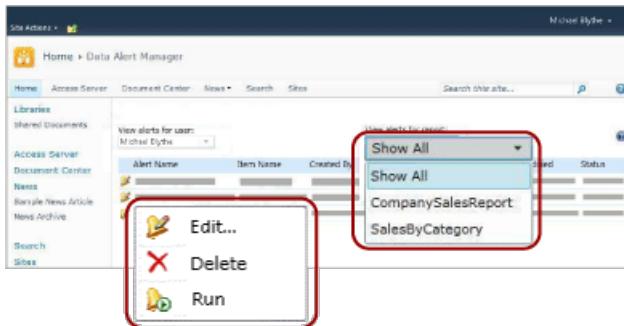
More questions? [Try asking the Reporting Services forum](#)

Manage My Data Alerts in Data Alert Manager

10/24/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2016) Power BI Report Server SharePoint

SharePoint users can view a list of the data alerts that they created and information about the alerts. Users can also delete their alerts, open alert definitions for edit in Data Alert Designer, and run their alerts. The following picture shows the features available to users in Data Alert Manager.

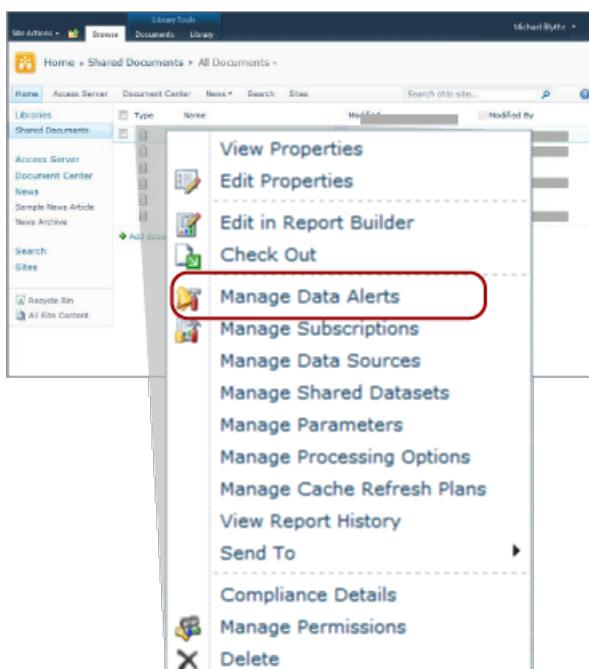


NOTE

Reporting Services integration with SharePoint is no longer available after SQL Server 2016.

To view a list of your alerts

1. Go to the SharePoint library where you saved the reports on which you created data alerts.
2. Click the icon for the expand drop-down menu on a report and click **Manage Data Alerts**. The following picture shows the drop-down menu.



Data Alert Manager opens. By default, it lists the alerts for the report that you selected in the library.

3. Click the down arrow next to the **View alerts for report** list and select a report to view its alerts, or click **Show All** to list all alerts.

NOTE

If the report that you selected does not have any alerts, you do not have to return to the SharePoint library to locate and select a report that has alerts. Instead, click **Show All** to see a list of all your alerts.

A table lists the alert name, report name, your name as the creator of the alert, the number the alert was sent, the last time the alert definition was modified, and the status of the alert. If the alert cannot be generated or sent, the status column contains information about the error and helps you troubleshoot the problem.

To edit an alert definition

- Right-click the data alert for which you want to edit the alert definition and click **Edit**.

The alert definition opens in Data Alert Designer. For more information, see [Edit a Data Alert in Alert Designer](#) and [Data Alert Designer](#).

NOTE

Only the user that created the data alert definition can edit it.

NOTE

If the report has changed and the data feeds generated from the report have changed the alert definition might no longer be valid. This occurs when a column that the alert references in its rules is deleted from the report, changes data type, or is included in a different data feed or the report is deleted or moved. You can open an alert definition that is not valid, but you cannot resave it until it is valid based on the current version of the report data feed that it is built upon. To learn more about how data feeds are generated from reports, see [Generating Data Feeds from Reports \(Report Builder and SSRS\)](#).

To delete an alert definition

- Right-click the data alert that you want to delete and click **Delete**.

When you delete the alert, no further alert messages are sent.

To run an alert

- Right-click the data alert that you want to run and click **Run**.

The alert instance is created and the data alert message is immediately sent, regardless of the schedule options you specified in Data Alert Designer. For example, an alert configured to be sent weekly and then only if the results change is sent.

See Also

[Data Alert Manager for Alerting Administrators](#)
[Reporting Services Data Alerts](#)

More questions? [Try asking the Reporting Services forum](#)

Grant Permissions to Users and Alerting Administrators

11/28/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2016) Power BI Report Server SharePoint

Before users and alerting administrators can create, edit, delete, and view data alerts they must be granted SharePoint permissions. There are no special permissions to use with the Reporting Services data alerting feature, you use the built-in SharePoint permissions.

NOTE

Reporting Services integration with SharePoint is no longer available after SQL Server 2016.

Information workers-Permissions must include the Create Alert and View Items SharePoint permissions. The built-in SharePoint permission levels named Design, Contribute, Read, and View Only include the Create Alert and View Items SharePoint permissions. You can also create a custom permission level with the permissions required to support users that create, edit, run, and view data alerts.

Alerting administrators-Permissions must include the Manage Alert SharePoint permission. By default only the Full Control permission level includes this permission for sites created with the Team Site site template. If you use other site templates, you will see different lists of default SharePoint groups. You can add the Manage Alert permission to one of the built-in permission levels or create a custom permission level with the permission required to support alerting administrators that view and delete data alerts.

To learn more about SharePoint permissions, see [User permissions and permission levels \(SharePoint Server 2010\)](#).

Grant permissions

1. Go to the SharePoint site to which you want to grant permissions.
2. On the toolbar, click **Site Actions** and then click **Site Permissions**.
If you do not see this option, you do not sufficient permission to grant permissions to others.
3. Click **Grant Permissions**.
4. In **Users/Groups**, type the user names, group names, or e-mail addresses you want grant permission to.
5. Select the **Add users to a SharePoint group** or **Grant users permission directly** option. Depending on whether you selected **Add users to a SharePoint group** or **Grant users permissions directly** do one of the following:
 - If you selected **Add users to a SharePoint group**, select a permission level in the drop-down list.
 - If you selected **Grant users permissions directly**, select a permission level.
6. Click **OK**.

See Also

[Set Permissions for Report Server Items on a SharePoint Site \(Reporting Services in SharePoint Integrated Mode\)](#)

[Reporting Services Data Alerts](#)

More questions? Try asking the [Reporting Services forum](#)

Extensions for SQL Server Reporting Services (SSRS)

12/17/2018 • 6 minutes to read • [Edit Online](#)

The report server in SQL Server Reporting Services uses extensions to modularize the types of input or output it accepts for authentication, data processing, report rendering, and report delivery. This makes it easy for existing Reporting Services installations to utilize new software standards in the industry, such as a new authentication scheme, or a custom data source type. The report server supports custom authentication extensions, data processing extensions, report processing extensions, rendering extensions, and delivery extensions, and the extensions that are available to the users are configurable in the RS ReportServer.config configuration file. For example, you can limit the export formats the report viewer is allowed to use. A report server requires at least one authentication extension, data processing extension, and rendering extension. Delivery and report processing extensions are optional, but necessary if you want to support report distribution or custom controls.

This topic describes the extensions that are readily available in Reporting Services.

Security Extensions

Security extensions are used to authenticate and authorize users and groups to a report server. The default security extension is based on Windows Authentication. You can also create a custom security extension to replace default security if your deployment model requires a different authentication approach (for example, if you require forms-based authentication for Internet or extranet deployment). Only one security extension can be used in a single Reporting Services installation. You can replace the default Windows Authentication security extension, but you cannot use it alongside a custom security extension.

Data Processing Extensions

Data Processing extensions are used to query a data source and return a flattened row set. Reporting Services uses different extensions to interact with different types of data sources. You can use the extensions that are included in Reporting Services, or you can develop your own extensions. Data processing extensions for SQL Server, Analysis Services, Oracle, SAP NetWeaver Business Intelligence, Hyperion Essbase, Teradata, OLE DB, and ODBC data sources are provided. Reporting Services can also use any ADO.NET data provider. Data processing extensions process query requests from the Report Processor component by performing the following tasks:

- Open a connection to a data source.
- Analyze a query and return a list of field names.
- Run a query against the data source and return a rowset.
- Pass parameters to a query, if required.
- Iterate through the rowset and retrieve data.

Some extensions can also perform the following tasks:

- Analyze a query and return a list of parameter names used in the query.
- Analyze a query and return the list of fields used for grouping.
- Analyze a query and return the list of fields used for sorting.
- Provide a user name and password to connect to the data source.
- Pass parameters with multiple values to a query.

- Iterate through rows and retrieve auxiliary metadata.

Rendering Extensions

Rendering extensions transform data and layout information from the Report Processor into a device-specific format. Reporting Services includes seven rendering extensions: HTML, Excel, CSV, XML, Image, PDF, and Microsoft Word.

- **HTML Rendering Extension** When you request a report from a report server through a Web browser, the report server uses the HTML rendering extension to render the report. The HTML rendering extension generates all HTML using UTF-8 encoding. For more information, see [Rendering to HTML \(Report Builder and SSRS\)](#) and [Browser Support for Reporting Services and Power View](#).
- **Excel Rendering Extension** The Excel rendering extension renders reports that can be viewed and modified in Microsoft Excel 97 or later. This rendering extension creates files in Binary Interchange File Format (BIFF). BIFF is the native file format for Excel data. Reports that are rendered in Microsoft Excel support all of the features available for any spreadsheet. For more information, see [Exporting to Microsoft Excel \(Report Builder and SSRS\)](#).
- **CSV Rendering Extension** The Comma-Separated Value (CSV) rendering extension renders reports in comma-delimited plain text files, without any formatting. Users can then open these files with a spreadsheet application, such as Microsoft Excel, or any other program that reads text files. For more information, see [Exporting to a CSV File \(Report Builder and SSRS\)](#).
- **XML Rendering Extension** The XML rendering extension renders reports in XML files. These XML files can then be stored or read by other programs. You can also use an XSLT transformation to turn the report into another XML schema for use by another application. The XML generated by the XML rendering extension is UTF-8 encoded. For more information, see [Exporting to XML \(Report Builder and SSRS\)](#).
- **Image Rendering Extension** The Image rendering extension renders reports to bitmaps or metafiles. The extension can render reports in the following formats: BMP, EMF, GIF, JPEG, PNG, TIFF, and WMF. By default, the image is rendered in TIFF format, which can be displayed with the default image viewer of your operating system (for example, Windows Picture and Fax Viewer). You can send the image to a printer from the viewer. Using the Image rendering extension to render reports ensures that the report looks the same on every client. (When a user views a report in HTML, the appearance of that report can vary depending on the version of the user's browser, the user's browser settings, and the fonts that are available.) The Image rendering extension renders the report on the server, so all users see the same image. Because the report is rendered on the server, all fonts that are used in the report must be installed on the server. For more information, see [Exporting to an Image File \(Report Builder and SSRS\)](#).
- **PDF Rendering Extension** The PDF rendering extension renders reports in PDF files that can be opened and viewed with Adobe Acrobat 6.0 or later. For more information, see [Exporting to a PDF File \(Report Builder and SSRS\)](#).
- **Word Rendering Extension** The Microsoft Word rendering extension renders a report as a Word document that is compatible with Microsoft Office Word 2000 or later. For more information, see [Exporting to Microsoft Word \(Report Builder and SSRS\)](#).

Report Processing Extensions

Report processing extensions can be added to provide custom report processing for report items that are not included with Reporting Services. By default, a report server can process tables, charts, matrices, lists, text boxes, images, and all other report items. If you want to add special features to a report that require custom processing during report execution (for example, if you want to embed a Microsoft MapPoint map), you can create a report processing extension to do so.

Delivery Extensions

The background processing application uses delivery extensions to deliver reports to various locations. Reporting Services includes an e-mail delivery extension and a file share delivery extension. The e-mail delivery extension sends an e-mail message through Simple Mail Transport Protocol (SMTP) that includes either the report itself or a URL link to the report. Short notices without the URL link or report can also be sent to pagers, phones, or other devices. The file share delivery extension saves reports to a shared folder on your network. You can specify a location, rendering format, and file name, and overwrite options for the file you create. You can use file share delivery for archiving rendered reports and as part of a strategy for working with very large reports. Delivery extensions work in conjunction with subscriptions. When a user creates a subscription, the user chooses one of the available delivery extensions to determine how the report is delivered.

Customize Rendering Extension Parameters in RSReportServer.Config

10/1/2018 • 5 minutes to read • [Edit Online](#)

You can specify rendering extension parameters in the RSReportServer configuration file to override default report rendering behavior for reports that run on a Reporting Services report server. You can modify rendering extension parameters to achieve the following objectives:

- Change how the rendering extension name appears in the Export list of the report toolbar (for example, to change "Web archive" to "MHTML"), or localize the name to a different language.
- Create multiple instances of the same rendering extension to support different report presentation options (for example, a portrait and landscape mode version of the Image rendering extension).
- Change the default rendering extension parameters to use different values (for example, the Image rendering extension uses TIFF as the default output format; you can modify the extension parameters to use EMF instead).

Changing the rendering extension parameters only affects rendering operations on the report server. You cannot override rendering extension settings in report preview in Report Designer.

Specifying rendering extension parameters in the configuration files affects rendering extensions globally. The settings in the configuration files are used in place of default values whenever a particular rendering extension is used. If you want to set rendering extension parameters for a specific report or render operation, you must specify device information programmatically using the [Render](#) method or by specifying device information settings on a report URL. For more information about specifying device information settings for a render operation, and to view the complete list of device information settings, see [Passing Device Information Settings to Rendering Extensions](#).

Finding and Modifying RSReportServer.config

Configuration settings for report output formats are specified as rendering extension parameters in the RSReportServer.config file. To specify rendering extension parameters in the configuration files, you must know how to define the XML structures that set rendering parameters. There are two XML structures that you can modify:

- The **OverrideNames** element defines the display name and language of the rendering extension.
- The **DeviceInfo** XML structure defines the device information settings that are used by a rendering extension. Most rendering extension parameters are specified as device information settings.

You can use a text editor to modify the file. The RSReportServer.config file can be found in the \Reporting Services\Report Server\Bin folder. For more information about modifying configuration files, see [Modify a Reporting Services Configuration File \(RSreportserver.config\)](#).

Changing the Display Name

The display name for a rendering extension appears in the Export list of the report toolbar. Examples of default display names include Web archive, TIFF file, and Acrobat (PDF) file. You can replace the default display name with a custom value by specifying the **OverrideNames** element in the configuration files. In addition, if you are defining two instances of a single rendering extension, you can use the **OverrideNames** element to distinguish

each instance in the Export list.

Because display names are localized, you must set the **Language** attribute if you are replacing the default display name with a custom value. Otherwise, any name that you specify will be ignored. The language value that you set must be valid for the report server computer. For example, if the report server is running on a French operating system, you should specify "fr-FR" as the attribute value.

The following example illustrates how to provide a custom name on an English report server:

```
<Extension Name="XML"
Type="Microsoft.ReportingServices.Rendering.DataRenderer.XmlDataReport,Microsoft.ReportingServices.DataRender
ing">
<OverrideNames>
    <Name Language="en-US">My Custom Display Name for XML Rendering</Name>
</OverrideNames>
</Extension>
```

Changing Device Information Settings

To modify default device information settings that are used by a rendering extension that is already deployed on your report server, you must type the **DeviceInfo** XML structure into the configuration files. Every rendering extension supports device information settings that are unique to that extension. To view the complete list of device information settings, see [Passing Device Information Settings to Rendering Extensions](#).

The following example provides an illustration of the XML structure and syntax that modifies the default settings of the Image rendering extension:

```
<Render>
    <Extension Name="IMAGE (EMF)"
Type="Microsoft.ReportingServices.Rendering.ImageRenderer.ImageRenderer,Microsoft.ReportingServices.ImageRend
ering">
        <OverrideNames>
            <Name Language="en-US">Image (EMF)</Name>
        </OverrideNames>
        <Configuration>
            <DeviceInfo>
                <ColorDepth>32</ColorDepth>
                <DpiX>300</DpiX>
                <DpiY>300</DpiY>
                <OutputFormat>EMF</OutputFormat>
            </DeviceInfo>
        </Configuration>
    </Extension>
</Render>
```

Configuring Multiple Entries for a Rendering Extension

You can create multiple instances of the same rendering extension to support different report presentation options. Each instance that you define can have a different combination of parameter values. When defining new instances of an existing rendering extension, be sure to do the following:

- Specify a unique name for the extension.

Each instance must have a unique value for the **Name** attribute. The following example uses the names "IMAGE (EMF Landscape)" and "IMAGE (EMF Portrait)" to distinguish between the two instances.

Use caution when changing the name of a rendering extension that is already deployed. Developers who specify rendering extensions programmatically use the extension name to identify which instance to use

for a particular render operation. If you are running custom Reporting Services applications on your report server, make sure that the developer knows if you modify an existing extension name or add a new one.

- Specify a unique display name so that users can understand the differences for each output format.

If you are configuring multiple versions of the same extension, you can give each version a unique name by providing a value for **OverrideNames**. Otherwise, all versions of the extension will appear to have the same name in the Export options list on the report toolbar.

The following example illustrates how to use the default Image rendering extension (which produces TIFF output) to output EMF in Portrait mode alongside a second instance that outputs reports in EMF in Landscape mode. Notice that each extension name is unique. When testing this example, remember to choose reports that do not contain interactive features such as show/hide options, matrices, or drillthrough links (interactive features do not work in the Image rendering extension):

```
<Render>
    <Extension Name="IMAGE (EMF Landscape)"
Type="Microsoft.ReportingServices.Rendering.ImageRenderer.ImageRenderer,Microsoft.ReportingServices.ImageRend
ering">
        <OverrideNames>
            <Name Language="en-US">EMF in Landscape Mode</Name>
        </OverrideNames>
        <Configuration>
            <DeviceInfo>
                <OutputFormat>EMF</OutputFormat>
                <PageHeight>8.5in</PageHeight>
                <PageWidth>11in</PageWidth>
            </DeviceInfo>
        </Configuration>
    </Extension>
    <Extension Name="IMAGE (EMF Portrait)"
Type="Microsoft.ReportingServices.Rendering.ImageRenderer.ImageRenderer,Microsoft.ReportingServices.ImageRend
ering">
        <OverrideNames>
            <Name Language="en-US">EMF in Portait Mode</Name>
        </OverrideNames>
        <Configuration>
            <DeviceInfo>
                <OutputFormat>EMF</OutputFormat>
                <PageHeight>11in</PageHeight>
                <PageWidth>8.5in</PageWidth>
            </DeviceInfo>
        </Configuration>
    </Extension>
</Render>
```

See Also

- [RsReportServer.config Configuration File](#)
- [RSReportDesigner Configuration File](#)
- [CSV Device Information Settings](#)
- [Excel Device Information Settings](#)
- [HTML Device Information Settings](#)
- [Image Device Information Settings](#)
- [MHTML Device Information Settings](#)
- [PDF Device Information Settings](#)
- [XML Device Information Settings](#)

URL Access (SSRS)

11/15/2018 • 4 minutes to read • [Edit Online](#)

URL access of the report server in SQL Server Reporting Services (SSRS) enables you to send commands to a report server through a URL request. For example, you can customize the rendering of a report on a native mode report server or in a SharePoint library. You may have viewed the report using a specific set of report parameter values, or you may have been viewing a particular page of interest in the report. You can encapsulate this information in the URL using predefined URL access parameters. You can further customize how the report server processes the report by embedding parameters for rendering formats or for the look and feel of the report viewer. You can then paste this URL directly into an email or Web page to let others to access your report in the same manner in the browser.

Other actions you can perform through URL access are:

- Send commands to the HTML viewer, such as adjusting its look and feel
- List the children of a catalog folder
- Retrieve the XML definition of a catalog item
- Render a specific report history snapshot
- Manage report sessions

For the complete list of commands and settings available through URL access, see [URL Access Parameter Reference](#).

URL Access Concepts

URL requests to the report server contain parameters that are processed by the report server. The way in which the report server handles URL requests depends on the parameters, parameter prefixes, and types of items that are included in the URL. Report server URLs adhere to the URL formatting guidelines as proposed by the joint World Wide Web Consortium W3C/IETF draft standard. Reporting Services URL functionality is compatible with most Internet browsers or applications that support standard URL addressing.

URL Access Syntax

URL requests can contain multiple parameters that are listed in any order. Parameters are separated by an ampersand (&) and name/value pairs are separated by an equal sign (=).

```
rswebserviceurl  
?  
reportpath  
[&prefix:param=value]...n]
```

Syntax Description

rswebserviceurl

The Web service URL of the report server. For native mode, it is the Web service URL of the report server instance configured in Reporting Services Configuration Manager (see [Configure Report Server URLs \(SSRS Configuration Manager\)](#)). For example:

```
https://myrhost/reportserver  
https://machine.adventure-works.com/reportserver_MYNAMEDINSTANCE
```

For SharePoint integrated mode, it is the URL of the Reporting Services proxy at a SharePoint site integrated with Reporting Services. For example:

```
https://myspsite/subsite/_vti_bin/reportserver
```

TIP

It is important the URL include the `_vti_bin` proxy syntax to route the request through SharePoint and the Reporting Services HTTP proxy. The proxy adds some context to the HTTP request, context that is required to ensure proper execution of the report for SharePoint mode report servers.

pathinfo

The relative path name of the item in the native mode report server database, or the fully qualified URL of the item in a SharePoint catalog.

The path of the catalog item. For native mode, it is the relative path of the item in the report server database, beginning with a slash (/). For example:

```
/AdventureWorks 2008R2/Employee_Sales_Summary_2008R2
```

For SharePoint integrated mode, it is the fully qualified URL of the item in the SharePoint library, including the item extension. For example:

```
https://myspsite/subsite/AdventureWorks 2008R2/Employee_Sales_Summary_2008R2.rdl
```

&

Used to separate name and value pairs of URL access parameters.

prefix

Optional. A prefix for the URL access parameter (for example, `rs:` or `rc:`) that accesses a specific process running within the report server.

NOTE

If a prefix for a URL access parameter is not included, the parameter is processed by the report server as a report parameter. Report parameters do not use a parameter prefix and are case-sensitive.

param

The parameter name.

value

URL text corresponding to the value of the parameter being used.

Note: For a list of the available URL access parameters, see [URL Access Parameter Reference](#). For examples passing report parameters on the URL, see [Pass a Report Parameter Within a URL](#).

Related Tasks

TASK DESCRIPTIONS	LINKS
Access report server items, such as reports, shared data sources, and resources.	Access Report Server Items Using URL Access
Pass report parameters to a report.	Pass a Report Parameter Within a URL
Set the locale of the report parameters in the URL access string, which defines the locale-specific interpretations of dates, currencies, and so on.	Set the Language for Report Parameters in a URL
Send rendering extension specific settings that customize how the report is rendered.	Specify Device Information Settings in a URL
Export a report directly to a file format without viewing it in the browser.	Export a Report Using URL Access
Open a report and navigate directly to the location of a string.	Search a Report Using URL Access
Render a specific report history snapshot.	Render a Report History Snapshot Using URL Access

See Also

[Pass a Report Parameter Within a URL](#)

[URL Access Parameter Reference](#)

[Integrating Reporting Services Using URL Access](#)

[Finding, Viewing, and Managing Reports \(Report Builder and SSRS \)](#)

Access Report Server Items Using URL Access

11/15/2018 • 2 minutes to read • [Edit Online](#)

This topic describes how to access catalog items of different types in a report server data base or in a SharePoint site using `rs:Command=Value`. It is not necessary to actually add this parameter string. If you omit it, the report server evaluates the item type and selects the appropriate parameter value automatically. However, using the `rs:Command=Value` string in the URL improves the performance of the report server.

Note the `_vti_bin` proxy syntax in the examples below. For more information about using the proxy syntax, see [URL Access Parameter Reference](#).

Access a Report

To view a report in the browser, use the `rs:Command=Render` parameter. For example:

- **Native** `https://myrshost/reportserver?/Sales/YearlySalesByCategory&rs:Command=Render`
- **SharePoint**

```
https://myspsite/subsite/_vti_bin/reportserver?  
https://myspsite/subsite/Sales/YearlySalesByCategory&rs:Command=Render
```

TIP

It is important the URL include the `_vti_bin` proxy syntax to route the request through SharePoint and the Reporting Services HTTP proxy. The proxy adds some context to the HTTP request, context that is required to ensure proper execution of the report for SharePoint mode report servers.

Access a Resource

To access a resource, use the `rs:Command=GetResourceContents` parameter. If the resource is compatible with the browser, such as an image, it is opened in the browser. Otherwise, you are prompted to open or save the file or resource to disk.

Native `https://myrshost/reportserver?/Sales/StorePicture&rs:Command=GetResourceContents`

SharePoint

```
https://myspsite/subsite/_vti_bin/reportserver?  
https://myspsite/subsite/Sales/StorePicture.jpg&rs:Command=GetResourceContents
```

Access a Data Source

To access a data source, use the `rs:Command=GetDataSourceContents` parameter. If your browser supports XML, the data source definition is displayed if you are an authenticated user with **Read Contents** permission on the data source. For example:

Native `https://myrshost/reportserver?/Sales/AdventureWorks2012&rs:Command=GetDataSourceContents`

SharePoint

```
https://myspsite/subsite/_vti_bin/reportserver?  
https://myspsite/subsite/Sales/AdventureWorks2012&rs:Command=GetDataSourceContents
```

The XML structure might look similar to the following example:

```
<DataSourceDefinition>
  <Extension>SQL</Extension>
  <ConnectionString>Provider=SQLOLEDB.1;Integrated Security=SSPI;Persist Security Info=False;Initial Catalog=AdventureWorks2012;Data Source=MYSERVER1;</ConnectionString>
  <CredentialRetrieval>Integrated</CredentialRetrieval>
  <WindowsCredentials>False</WindowsCredentials>
  <ImpersonateUser>False</ImpersonateUser>
  <Prompt />
  <Enabled>True</Enabled>
</DataSourceDefinition>
```

The connection string is returned based on the **SecureConnectionLevel** setting of the report server. For more information about the **SecureConnectionLevel** setting, see [Using Secure Web Service Methods](#).

Access the Contents of a Folder

To access the contents of a folder, use the *rs:Command=GetChildren* parameter. A generic folder-navigation page is returned that contains links to the subfolders, reports, data sources, and resources in the requested folder. For example:

Native `https://myrhost/reportserver?/Sales&rs:Command=GetChildren`

SharePoint

`https://myspsite/subsite/_vti_bin/reportserver?https://myspsite/subsite/Sales&rs:Command=GetChildren`

The user interface you see is similar to the directory browsing mode used by Microsoft Internet Information Server (IIS). The version number, including the build number, of the report server is also displayed below the folder listing.

See Also

[URL Access \(SSRS\)](#)

[URL Access Parameter Reference](#)

Pass a Report Parameter Within a URL

12/10/2018 • 2 minutes to read • [Edit Online](#)

You can pass report parameters to a report by including them in a report URL. These URL parameters are not prefixed because they are passed directly to the report processing engine.

IMPORTANT

It is important the URL include the `_vti_bin` proxy syntax to route the request through SharePoint and the Reporting Services HTTP proxy. The proxy adds some context to the HTTP request, context that is required to ensure proper execution of the report for SharePoint mode report servers.

If you don't include the proxy syntax, then you need to prefix the parameter with `rp:`.

All query parameters can have corresponding report parameters. You pass a query parameter to a report by passing the corresponding report parameter. For more information, see [Build a Query in the Relational Query Designer \(Report Builder and SSRS\)](#).

IMPORTANT

Report parameters are case-sensitive.

NOTE

Report parameters are case-sensitive and utilize the following special characters:

- Any space characters in the URL string are replaced with the characters "%20," according to URL encoding standards.
- A space character in the parameter portion of the URL is replaced with a plus character (+).
- A semicolon in any portion of the string is replaced with the characters "%3A."
- Browsers should automatically perform the proper URL encoding. You do not have to encode any of the characters manually.

To set a report parameter within a URL, use the following syntax:

```
parameter=value
```

For example, to specify two parameters, "ReportMonth" and "ReportYear", defined in a report, use the following URL for a native mode report server:

```
https://myrshost/ReportServer?/AdventureWorks  
2008R2/Employee_Sales_Summary_2008R2&ReportMonth=3&ReportYear=2008
```

For example, to specify the same two parameters defined in a report, use the following URL for a SharePoint integrated mode report server. Note the `/vti_bin`:

```
https://myspsite/subsite/_vti_bin/reportserver?https://myspsite/subsite/AdventureWorks  
2008R2/Employee_Sales_Summary_2008R2.rdl&ReportMonth=3&ReportYear=2008
```

To pass a null value for a parameter, use the following syntax:

```
parameter  
:isnull=true
```

For example,

```
SalesOrderNumber:isnull=true
```

To pass a **Boolean** value, use 0 for false and 1 for true. To pass a **Float** value, include the decimal separator of the server locale

NOTE

If your report contains a report parameter that has a default value and the value of the **Prompt** property is **false** (that is, the Prompt User property is not selected in Report Manager), then you cannot pass a value for that report parameter within a URL. This provides administrators an option for preventing end users from adding or modifying the values of certain report parameters.

Additional Examples

The following URL example includes spaces and multiple parameters

- Folder name of "SQL Server User Education Team" includes spaces and therefore the "+" replaces each space.
- Report name of "team project report" includes spaces and therefore the "+" replaces each space.
- Passes two parameters of "teamgrouping2" with a value of "xgroup" and "teamgrouping1" with a value of "ygroup".

```
https://myserver/ReportServer?/SQL+Server+User+Education+Team/_ContentTeams/folder123/team+project+report&teamgrouping2=xgroup&teamgrouping1=ygroup
```

The following URL example includes a multi-value parameter "OrderID". The format for a Multi-Value parameter is to repeat the parameter name for each value.

```
https://myserver/ReportServer?/SQL+Server+User+Education+Team/_ContentTeams/folder123/team+project+report&teamgrouping2=xgroup&teamgrouping1=ygroup&OrderID=747&OrderID=787&OrderID=12
```

The following URL example passes a single parameter of *SellStartDate* with a value of "7/1/2005", for a native mode report server.

```
https://myserver/ReportServer/Pages/ReportViewer.aspx?  
%2fProduct_and_Sales_Report_AdventureWorks&SellStartDate=7/1/2005
```

See Also

[URL Access \(SSRS\)](#)

[URL Access Parameter Reference](#)

URL Access Parameter Reference

11/28/2018 • 10 minutes to read • [Edit Online](#)

You can use the following parameters as part of a URL to configure the look and feel of your SQL Server 2016 Reporting Services (SSRS)reports. The most common parameters are listed in this section. Parameters are case-insensitive and begin with the parameter prefix *rs:* if directed to the report server and *rc:* if directed to an HTML Viewer. You can also specify parameters that are specific to devices or rendering extensions. For more information about device-specific parameters, see [Specify Device Information Settings in a URL](#).

IMPORTANT

For a SharePoint mode report server it is important the URL include the `_vti_bin` proxy syntax to route the request through SharePoint and the Reporting Services HTTP proxy. The proxy adds context to the HTTP request that is required to ensure proper execution of the report for SharePoint mode report servers. For examples, see [Access Report Server Items Using URL Access](#).

For information about including report parameters in a URL, and examples, see [Pass a Report Parameter Within a URL](#).

In this topic

- [HTML Viewer Commands \(rc:\)](#)
- [Report Server Commands \(rs:\)](#)
- [Report Viewer Web Part Commands \(rv:\)](#)

HTML Viewer Commands (rc:)

HTML Viewer commands are used to target the HTML Viewer (for example, from Report Manager) and are prefixed with *rc:*

- *Toolbar* :

Shows or hides the toolbar. If the value of this parameter is **false**, all remaining options are ignored. If you omit this parameter, the toolbar is automatically displayed for rendering formats that support it. The default of this parameter is **true**.

IMPORTANT

rc:Toolbar=false does not work for URL access strings that use an IP address, instead of a domain name, to target a report hosted on a SharePoint site.

- *Parameters* : Shows or hides the parameters area of the toolbar. If you set this parameter to **true**, the parameters area of the toolbar is displayed. If you set this parameter to **false**, the parameters area is not displayed and cannot be displayed by the user. If you set this parameter to a value of **Collapsed**, the parameters area will not be displayed, but can be toggled by the end user. The default value of this parameter is **true**.

For an example in **Native** mode:

```
https://myrshost/reportserver?/Sales&rc:Parameters=Collapsed
```

For an example in **SharePoint** mode:

```
https://myspsite/subsite/_vti_bin/reportserver?https://myspsite/subsite/Sales&rc:Parameters=Collapsed
```



- **Zoom** : Sets the report zoom value as an integer percentage or a string constant. Standard string values include **Page Width** and **Whole Page**. This parameter is ignored by versions of Internet Explorer earlier than Internet Explorer 5.0 and all non-Microsoft browsers. The default value of this parameter is **100**.

For example in **Native** mode:

```
https://myrshost/reportserver?/Sales&rc:Zoom=Page Width
```

For example in **SharePoint** mode.

```
https://myspsite/subsite/_vti_bin/reportserver?https://myspsite/subsite/Sales&rc:Zoom=Page Width
```

- **Section** : Sets which page in the report to display. Any value that is greater than the number of pages in the report displays the last page. Any value that is less than **0** displays page 1 of the report. The default value of this parameter is **1**.

For example in **Native** mode, to display page 2 of the report:

```
https://myrshost/reportserver?/Sales&rc:Section=2
```

For example in **SharePoint** mode, to display page 2 of the report:

```
https://myspsite/subsite/_vti_bin/reportserver?https://myspsite/subsite/Sales&rc:Section=2
```

- **FindString**: Search a report for a specific set of text.

For an example in **Native** mode.

```
https://myrshost/reportserver?/Sales&rc:FindString=Mountain-400
```

For an example in **SharePoint** mode.

```
https://myspsite/subsite/_vti_bin/reportserver?https://myspsite/subsite/Sales&rc:FindString=Mountain-400
```

- **StartFind** : Specifies the last section to search. The default value of this parameter is the last page of the report.

For an example in **Native** mode that searches for the first occurrence of the text "Mountain-400" in the Product Catalog sample report starting with page one and ending with page five.

```
https://server/Reportserver?/SampleReports/Product  
Catalog&rs:Command=Render&rc:StartFind=1&rc:EndFind=5&rc:FindString=Mountain-400
```

- **EndFind** : Sets the number of the last page to use in the search. For example, a value of **5** indicates that the last page to be searched is page 5 of the report. The default value is the number of the current page.

Use this parameter in conjunction with the *StartFind* parameter. See the above example.

- *FallbackPage* : Sets the number of the page to display if a search or a document map selection fails. The default value is the number of the current page.
- *GetImage* : Gets a particular icon for the HTML Viewer user interface.
- *Icon* : Gets the icon of a particular rendering extension.
- *Stylesheet*: Specifies a style sheet to be applied to the HTML Viewer.
- Device Information Setting: Specifies a device information setting in the form of `rc:tag=value`, where *tag* is the name of a device information setting specific to the rendering extension that is currently used (see the description for the *Format* parameter). For example, you can use the *OutputFormat* device information setting for the IMAGE rendering extension to render the report to a JPEG image using the following parameters in the URL access string: `...&rs:Format=IMAGE&rc:OutputFormat=JPEG`. For more information on all extension-specific device information settings, see [Device Information Settings for Rendering Extensions \(Reporting Services\)](#).

Report Server Commands (rs:)

Report server commands are prefixed with *rs:* and are used to target the report server:

- **Command:** Performs an action on a catalog item, depending on its item type. The default value is determined by the type of the catalog item referenced in the URL access string. Valid values are:
 - **ListChildren** and **GetChildren** Displays the contents of a folder. The folder items are displayed within a generic item-navigation page.

For example in **Native** mode.

```
https://myrshost/reportserver?/Sales&rs:Command=GetChildren
```

For example, a named instance in **Native** mode.

```
https://myssrhost/Reportserver_THESQLINSTANCE?/reportfolder&rs:Command=listChildren
```

For example in **SharePoint** mode.

```
https://myspsite/subsite/_vti_bin/reportserver?  
https://myspsite/subsite/Sales&rs:Command=GetChildren
```

- **Render** The report is rendered in the browser so you can view it.

For example in **Native** mode:

```
https://myrshost/reportserver?/Sales/YearlySalesByCategory&rs:Command=Render
```

For example in **SharePoint** mode.

```
https://myspsite/subsite/_vti_bin/reportserver?  
https://myspsite/subsite/Sales/YearlySalesByCategory&rs:Command=Render
```

- **GetSharedDatasetDefinition** Displays the XML definition associated with a shared dataset.

Shared dataset properties, including the query, dataset parameters, default values, dataset filters, and data options such as collation and case sensitivity, are saved in the definition. You must have **Read Report Definition** permission on a shared dataset to use this value.

For example in **Native** mode.

```
https://localhost/reportserver/?/DataSet1&rs:command=GetShareddatasetDefinition
```

- **GetDataSourceContents** Displays the properties of a given shared data source as XML. If your browser supports XML and if you are an authenticated user with **Read Contents** permission on the data source, the data source definition is displayed.

For example in **Native** mode.

```
https://myrhost/reportserver?/Sales/AdventureWorks2012&rs:Command=GetDataSourceContents
```

For example in **SharePoint** mode.

```
https://myspsite/subsite/_vti_bin/reportserver?  
https://myspsite/subsite/Sales/AdventureWorks2012&rs:Command=GetDataSourceContents
```

- **GetResourceContents** Renders a resource and displays it in an HTML page if the resource is compatible with the browser. Otherwise, you are prompted to open or save the file or resource to disk.

For example in **Native** mode.

```
https://myrhost/reportserver?/Sales/StorePicture&rs:Command=GetResourceContents
```

For example in **SharePoint** mode.

```
https://myspsite/subsite/_vti_bin/reportserver?  
https://myspsite/subsite/Sales/StorePicture.jpg&rs:Command=GetResourceContents
```

- **GetComponentDefinition** Displays the XML definition associated with a published report item. You must have **Read Contents** permission on a published report item to use this value.

- *Format :*

Specifies the format in which to render and view a report. Common values include:

- **HTML5**
- **PPTX**
- **ATOM**
- **HTML4.0**
- **MHTML**
- **IMAGE**
- **EXCEL**
- **WORD**

- **CSV**
- **PDF**
- **XML**

The default value is **HTML5**. For more information, see [Export a Report Using URL Access](#).

For a complete list, see the **<Render>** extension section of the report server rsreportserver.config file. For information on where to find the file, see [RsReportServer.config Configuration File](#).

For example, to get a PDF copy of a report directly from a **Native** mode report server:

```
https://myrshost/ReportServer?/myreport&rs:Format=PDF
```

For example, to get a PDF copy of a report directly from a **SharePoint** mode report server:

```
https://myspsite/subsite/_vti_bin/reportserver?https://myspsite/subsite/myrereport.rdl&rs:Format=PDF
```

- *ParameterLanguage*:

Provides a language for parameters passed in a URL that is independent of the browser language. The default value is the browser language. The value can be a culture value, such as **en-us** or **de-de**.

For example in **Native** mode, to override the browser language and specify a culture value of de-DE:

```
https://myrshost/ReportServer?/SampleReports/Product+Line+Sales&rs:Command=Render&StartDate=4/10/2008&EndDate=11/10/2008&rs:ParameterLanguage=de-DE
```

- *Snapshot* : Renders a report based on a report history snapshot. For more information, see [Render a Report History Snapshot Using URL Access](#).

For example in **Native** mode, retrieve a report history snapshot dated 2003-04-07 with a timestamp of 13:40:02:

```
https://myrshost/reportserver?/SampleReports/Company_Sales&rs:Snapshot=2003-04-07T13:40:02
```

- *PersistStreams*:

Renders a report in a single persisted stream. This parameter is used by the Image renderer to transmit the rendered report one chunk at a time. After using this parameter in a URL access string, use the same URL access string with the *GetNextStream* parameter instead of the *PersistStreams* parameter to get the next chunk in the persisted stream. This URL command will eventually return a 0-byte stream to indicate the end of the persisted stream. The default value is **false**.

- *GetNextStream*:

Gets the next data chunk in a persisted stream that is accessed using the *PersistStreams* parameter. For more information, see the description for *PersistStreams*. The default value is **false**.

- *SessionID*:

Specifies an established active report session between the client application and the report server. The value of this parameter is set to the session identifier.

You can specify the session ID as a cookie or as part of the URL. When the report server has been configured not to use session cookies, the first request without a specified session ID results in a redirection with a session ID. For more information about report server sessions, see [Identifying Execution State](#).

- *ClearSession*:
A value of **true** directs the report server to remove a report from the report session. All report instances associated with an authenticated user are removed from the report session. (A report instance is defined as the same report run multiple times with different report parameter values.) The default value is **false**.
- *ResetSession*:
A value of **true** directs the report server to reset the report session by removing the report session's association with all report snapshots. The default value is **false**.
- *ShowHideToggle*:
Toggles the show and hide state of a section of the report. Specify a positive integer to represent the section to toggle.

Report Viewer Web Part Commands (*rv:*)

The following SQL Server reserved report parameter names are used to target the Report Viewer Web Part that is integrated with SharePoint. These parameter names are prefixed with *rv*: The Report Viewer Web Part also accepts the *rs:ParameterLanguage* parameter.

- *Toolbar*: Controls the toolbar display for the Report Viewer Web Part. The default value is **Full**. Values can be:
 - **Full**: display the complete toolbar.
 - **Navigation**: display only pagination in the toolbar.
 - **None**: do not display the toolbar.

For example in **SharePoint** mode, to display only pagination in the toolbar.

```
https://myspsite/_vti_bin/reportserver?
https://myspsite002%fShared+Documents%2fmyreport.rdl&rv:DocMapMode=Displayed&rv:Toolbar=Navigation
```

- *HeaderArea*: Controls the header display for the Report Viewer Web Part. The default value is **Full**. Values can be:
 - **Full**: display the complete header.
 - **BreadCrumbsOnly**: display only the bread-crumb navigation in the header to inform the user where they are in the application.
 - **None**: do not display the header.

For example in **SharePoint** mode, to display only the bread-crumb navigation in the header.

```
https://myspsite/_vti_bin/reportserver?
https://myspsite002%fShared+Documents%2fmyreport.rdl&rv:DocMapMode=Displayed&rv:HeaderArea=BreadCrumb
sOnly
```

- *DocMapAreaWidth*: Controls the display width, in pixels, of the parameter area in the Report Viewer Web Part. The default value is the same as the Report Viewer Web Part default. The value must be a non-negative integer.
- *AsyncRender*: Controls whether a report is rendered asynchronously. The default value is **true**, which specifies that a report be rendered asynchronously. The value must be a Boolean value of **true** or **false**.
- *ParamMode*: Controls how the Report Viewer Web Part's parameter prompt area is displayed in full-page view. The default value is **Full**. Valid values are:

- **Full**: display the parameter prompt area.
- **Collapsed**: collapse the parameter prompt area.
- **Hidden**: hide the parameter prompt area.

For example in **SharePoint** mode, to collapse the parameter prompt area.

```
https://myspsite/_vti_bin/reportserver?
https://myspsite002%fShared+Documents%2fmyreport.rdl&rv:DocMapMode=Displayed&rv:ParamMode=Collapsed
```

- *DocMapMode*: Controls how the Report Viewer Web Part's document map area is displayed in full-page view. The default value is **Full**. Valid values are:
 - **Full**: display the document map area.
 - **Collapsed**: collapse the document map area.
 - **Hidden**: hide the document map area.
- *DockToolBar*: Controls whether the Report Viewer Web Part's toolbar is docked to the top or bottom. Valid values are **Top** and **Bottom**. The default value is **Top**.

For example in **SharePoint** mode, to dock the toolbar to the bottom.

```
https://myspsite/_vti_bin/reportserver?
https://myspsite002%fShared+Documents%2fmyreport.rdl&rv:DocMapMode=Displayed&rv:DockToolBar=Bottom
```

- *ToolBarItemsDisplayMode*: Controls which toolbar items are displayed. This is a bitwise enumeration value. To include a toolbar item, add the item's value to the total value. For example: for no Actions menu, use rv:ToolBarItemsDisplayMode=63 (or 0x3F), which is 1+2+4+8+16+32; for Actions menu items only, use rv:ToolBarItemsDisplayMode=960 (or 0x3C0). The default value is **-1**, which includes all toolbar items. Valid values are:
 - 1 (0x1): the **Back** button
 - 2 (0x2): the text search controls
 - 4 (0x4): the page navigation controls
 - 8 (0x8): the **Refresh** button
 - 16 (0x10): the **Zoom** list box
 - 32 (0x20): the **Atom Feed** button
 - 64 (0x40): the **Print** menu option in **Actions**
 - 128 (0x80): the **Export** submenu in **Actions**
 - 256 (0x100: the **Open with Report Builder** menu option in **Actions**
 - 512 (0x200: the **Subscribe** menu option in **Actions**
 - 1024 (0x400: the **New Data Alert** menu option in **Actions**

For example, in **SharePoint** mode to display only the **Back** button, text search controls, page navigation controls, and the **Refresh** button.

[https://myspsite/_vti_bin/reportserver?
https://myspsite002%fShared+Documents%2fmyreport.rdl&rv:DocMapMode=Displayed&rv:ToolBarItemsDisplayMode=15](https://myspsite/_vti_bin/reportserver?https://myspsite002%fShared+Documents%2fmyreport.rdl&rv:DocMapMode=Displayed&rv:ToolBarItemsDisplayMode=15)

See Also

[URL Access \(SSRS\)](#)

[Export a Report Using URL Access](#)

Set the Language for Report Parameters in a URL

11/15/2018 • 2 minutes to read • [Edit Online](#)

The *rs:ParameterLanguage* URL access parameter alleviates a problem in which culture-sensitive report parameters, such as dates, times, currency, and numbers, are interpreted using the browser language. With *rs:ParameterLanguage*, the URL is now interpreted independently of the browser. For example, if the report server is set to a regional setting of German, but a user is accessing a report via a URL using a browser that is set to English-United States, parameter values that are passed to a report server will be misinterpreted.

Consider the following URL to a report:

```
https://myrshost/Reportserver?/SampleReports/Product+Line+Sales&rs:Command=Render&StartDate=4/10/2008&EndDate=11/10/2008
```

In the above case, the server, running under a culture of "de-de", generates a URL either through an e-mail subscription or a hyperlink. The hyperlink indicates that the report should be parameterized by a start date of October 4, 2008 and an end date of October 11, 2008 according to German date/time standards. However, a user that is accessing the URL through a browser set to "en-us" forces the server to interpret the values as April 10, 2008 and November 10, 2008 under United States English date/time standards. To fix the problem, *rs:ParameterLanguage* can be used to override the browser language for parameter interpretation:

```
https://myrshost/Reportserver?/SampleReports/Product+Line+Sales&rs:Command=Render&StartDate=4/10/2008&EndDate=11/10/2008&rs:ParameterLanguage=de-DE
```

In addition to a value of **true** and **false** for the URL access parameter *rc:Parameters*, you can now pass a value of **Collapsed**. When using *rc:Parameters=Collapsed* on a URL, the parameter prompt area of the HTML viewer is collapsed out of sight, but can still be toggled by the user. A value of **false** removes the parameter prompt area from the HTML viewer toolbar and makes it unavailable to the end-user.

See Also

[URL Access \(SSRS\)](#)

[URL Access Parameter Reference](#)

Specify Device Information Settings in a URL

11/15/2018 • 2 minutes to read • [Edit Online](#)

Device information settings are parameters that are passed to a rendering extension. If you use the methods of the SQL Server Report Server Web service to render a report, a **DeviceInfo** XML element is passed as an input parameter. Child elements of the **DeviceInfo** element are specific to the device information settings of different rendering extensions. You can include device information settings in a URL by using the *rc:tag=value* parameter string, where *tag* is the name of the device information settings element being accessed. For more information about device information settings in SQL Server Reporting Services, see [Passing Device Information Settings to Rendering Extensions](#).

Example

The following example sets the format of the specified report to JPEG by using the *OutputFormat* device information setting of the image rendering extension (the line breaks have been added for legibility):

```
https://servername/reportserver?/SampleReports  
/Employee Sales Summary&EmployeeID=38&rs:  
Command=Render&rs:Format=IMAGE&rc:OutputFormat=JPEG
```

See Also

[URL Access \(SSRS\)](#)

[URL Access Parameter Reference](#)

Export a Report Using URL Access

11/15/2018 • 2 minutes to read • [Edit Online](#)

You can optionally specify the format in which to render a report by using the `rs:Format` URL parameter. The HTML4.0 and HTM5 formats (rendering extension) will render in the browser and for other formats, the browser will prompt to save the report output to a local file.

For example, to get a PDF copy of a report directly from a native mode report server:

```
https://myrshost/ReportServer?/myreport&rs:Format=PDF
```

And, from a SharePoint integrated mode report server:

```
https://myspsite/subsite/_vti_bin/reportserver?https://myspsite/subsite/myrereport.rdl&rs:Format=PDF
```

For example the following URL command in your browser exports a PPTX report from a named instance of the report server:

```
https://servername/ReportServer_THESQLINSTANCE/Pages/ReportViewer.aspx?  
%2freportfolder%2freport+name+with+spaces&rs:Format=pptx
```

Valid values for this parameter are based on the report rendering extensions that are installed on the report server being accessed. Common extensions are HTML4.0, MHTML, IMAGE, EXCELOPENXML (xlsx) , WORDOPENXML (docx), CSV, PDF, XML, and NULL. If a specified rendering extension is not installed on the report server, the report is not rendered and an error is generated and displayed in the browser.

If you do not include the `Format` parameter as part of the URL, the report server detects the browser and renders the report in the appropriate HTML format.

See Also

[URL Access \(SSRS\)](#)

[URL Access Parameter Reference](#)

Search a Report Using URL Access

11/15/2018 • 2 minutes to read • [Edit Online](#)

You can search a report for a specific set of text using URL access. To search a report, set the value of the *rc:FindString* parameter on the URL equal to the text for which you want to search. Additionally, use the *rc:StartFind* and *rc:EndFind* parameters to narrow your search to specific pages within the report.

Example

The following URL access example searches for the first occurrence of the text "Mountain-400" in the Product Catalog sample report starting with page one and ending with page five:

```
https://server/ReportServer?/SampleReports/Product  
Catalog&rs:Command=Render&rc:StartFind=1&rc:EndFind=5&rc:FindString=Mountain-400
```

See Also

[URL Access \(SSRS\)](#)

[URL Access Parameter Reference](#)

Render a Report History Snapshot Using URL Access

11/15/2018 • 2 minutes to read • [Edit Online](#)

You can render a report based on a report history snapshot by supplying the *rs:Snapshot* parameter and setting its value to a valid snapshot ID. The parameter value is in the format YYYY-MM-DDTHH:MM:SS, based on the International Organization for Standardization (ISO) 8601 standard.

If you omit this parameter, the report is rendered according to the report execution and cache management option settings of the report server. For more information about report execution, see [Set Report Processing Properties](#).

Example

The following example shows a URL that retrieves a report history snapshot:

```
https://myrshost/reportserver?/SampleReports/Company_Sales&rs:Snapshot=2003-04-07T13:40:02
```

See Also

[URL Access \(SSRS\)](#)

[URL Access Parameter Reference](#)

Reporting Services WMI Provider Library Reference (SSRS)

10/1/2018 • 2 minutes to read • [Edit Online](#)

The Reporting Services Windows Management Instrumentation (WMI) provider supports WMI operations that enable you to write scripts and code to modify settings of the report server and Report Manager.

For example, if you want to change whether integrated security is used when the report server connects to the report server database, create an instance of the MSReportServer_ConfigurationSetting class and use the DatabaseIntegratedSecurity property of the report server instance. The classes shown in the following table represent Reporting Services components. The classes are defined in either the **root\Microsoft\SqlServer\ReportServer\<InstanceName>\v13** or the **root\Microsoft\SqlServer\ReportServer\<InstanceName>\v13\Admin** namespaces. Each of the classes support read and write operations. Create operations are not supported.

Classes

[MSReportServer_Instance Class](#)

Provides basic information required for a client to connect to an installed report server.

[MSReportServer_ConfigurationSetting Class](#)

Represents the installation and run-time parameters of a report server instance. These parameters are stored in the configuration file for the report server.

For more information about WMI operations, see the WMI SDK documentation included with the Microsoft .NET Framework SDK.

See Also

[Access the Reporting Services WMI Provider](#)

[Technical Reference \(SSRS\)](#)

Troubleshoot Reporting Services

10/1/2018 • 2 minutes to read • [Edit Online](#)

The topics in this section help identify and troubleshoot issues with Reporting Services.



Troubleshoot issues with Reporting Services

- [Troubleshoot Reporting Services Report Issues](#)
- [Troubleshoot Reporting Services Subscriptions and Delivery](#)
- [Troubleshoot Report Design Issues with Reporting Services](#)
- [Troubleshoot Data Retrieval issues with Reporting Services Reports](#)
- [Troubleshoot Processing of Reporting Services Reports](#)
- [Troubleshoot Reporting Services Report Rendering Issues](#)
- [Troubleshoot Publishing or Viewing a Report on a Native Mode Report Server](#)
- [Troubleshoot Server and Database Connection Problems with Reporting Services](#)



Error and Events Reference

- [Errors and Events Reference \(Reporting Services\)](#)
- [Cause and Resolution of Reporting Services Errors](#)

See Also

- [Reporting Services Log Files and Sources](#)
- [Turn on Reporting Services events for the SharePoint trace log \(ULS\)](#)
- [SQL Server Reporting Services on Stack Overflow](#)
- Log an issue or suggestion at [Microsoft SQL Server UserVoice](#).

Develop with the REST APIs for Reporting Services

12/12/2018 • 3 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server Reporting Services (2017 and later) Power BI Report Server

Microsoft SQL Server 2017 Reporting Services support Representational State Transfer (REST) APIs. The REST APIs are service endpoints that support a set of HTTP operations (methods), which provide create, retrieve, update, or delete access for resources within a report server.

The REST API provides programmatic access to the objects in a SQL Server 2017 Reporting Services report server catalog. Examples of objects are folders, reports, KPIs, data sources, datasets, refresh plans, subscriptions, and more. Using the REST API, you can, for example, navigate the folder hierarchy, discover the contents of a folder, or download a report definition. You can also create, update, and delete objects. Examples of working with objects are upload a report, execute a refresh plan, delete a folder, and so on.

NOTE

If you're interested in viewing or deleting personal data, please review Microsoft's guidance in the [Windows Data Subject Requests for the GDPR](#) site. If you're looking for general information about GDPR, see the [GDPR section of the Service Trust portal](#).

Components of a REST API request/response

A REST API request/response pair can be separated into five components:

- The **request URI**, which consists of: `{URI-scheme} :// {URI-host} / {resource-path} ? {query-string}`.
Although the request URI is included in the request message header, we call it out separately here because most languages or frameworks require you to pass it separately from the request message.
 - URI scheme: Indicates the protocol used to transmit the request. For example, `http` or `https`.
 - URI host: Specifies the domain name or IP address of the server where the REST service endpoint is hosted, such as `myserver.contoso.com`.
 - Resource path: Specifies the resource or resource collection, which may include multiple segments used by the service in determining the selection of those resources. For example:
`CatalogItems(01234567-89ab-cdef-0123-456789abcdef)/Properties` can be used to get the specified properties for the CatalogItem.
 - Query string (optional): Provides additional simple parameters, such as the API version or resource selection criteria.
- HTTP request message header fields:
 - A required **HTTP method** (also known as an operation or verb), which tells the service what type of operation you are requesting. Reporting Services REST APIs support DELETE, GET, HEAD, PUT, POST, and PATCH methods.
 - Optional additional header fields, as required by the specified URI and HTTP method.
- Optional HTTP **request message body** fields, to support the URI and HTTP operation. For example, POST operations contain MIME-encoded objects that are passed as complex parameters. For POST or PUT operations, the MIME-encoding type for the body should be specified in the `Content-type` request header as well. Some services require you to use a specific MIME type, such as `application/json`.
- HTTP **response message header** fields:

- An [HTTP status code](#), ranging from 2xx success codes to 4xx or 5xx error codes. Alternatively, a service-defined status code may be returned, as indicated in the API documentation.
 - Optional additional header fields, as required to support the request's response, such as a `Content-type` response header.
- Optional HTTP **response message body** fields:
 - MIME-encoded response objects are returned in the HTTP response body, such as a response from a GET method that is returning data. Typically, these objects are returned in a structured format such as JSON or XML, as indicated by the `Content-type` response header.

API documentation

A modern REST API calls for modern API documentation. The REST API is built on the OpenAPI specification (a.k.a. the swagger specification) and documentation is available on [SwaggerHub](#). Beyond documenting the API, SwaggerHub helps generate a client library in the language of choice - JavaScript, TypeScript, C#, Java, Python, Ruby, and more.

Testing API calls

A tool for testing HTTP request/response messages is [Fiddler](#). Fiddler is a free web debugging proxy that can intercept your REST requests, making it easy to diagnose the HTTP request/ response messages.

Next steps

Review the available APIs over on [SwaggerHub](#).

Samples are available on [GitHub](#). The sample includes an HTML5 app built on TypeScript, React, and webpack along with a PowerShell example.

More questions? [Try asking the Reporting Services forum](#)

Integrating Reporting Services into Applications

10/24/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO:  SQL Server Reporting Services (2016)  SQL Server Reporting Services (2017)  Power BI Report Server

Reporting Services is an open and extensible reporting platform designed to provide developers with a comprehensive set of APIs for developing solutions.

NOTE

Starting with SQL Server 2017 Reporting Services, REST API access is available for developing solutions. SOAP API access has been deprecated. For more information, see [Develop with the REST APIs for Reporting Services](#).

There are three options for integrating Reporting Services into custom applications: the Report Server Web service, also known as the Reporting Services SOAP API, the Report Viewer controls for Microsoft Visual Studio, and URL access. Each option provides a different approach for integrating Reporting Services into your applications.

Report server web service

The Report Server Web service is the primary interface for developing against Reporting Services. Whether you are developing code to manage your report catalog or developing code to render reports to a supported format, the Web service exposes all the necessary methods to integrate Reporting Services into your applications. An example of one such application is Report Manager, which is included with Reporting Services; it uses the Web service to manage the report server database.

Report Viewer controls for Visual Studio

The Report Viewer controls available for Visual Studio are used for integrating report viewing into your applications. There are two controls: one for Windows Forms-based applications and one for Web Forms applications. Each control provides the capability for viewing reports that have been deployed to a report server as well as the ability to render reports that exist in an environment where a report server has not been installed.

URL access

URL access is another option for integrating report viewing into your applications if the Report Viewer controls are not an option. In addition, URL access is useful for sending links to reports to users via e-mail.

In this section

[Integrating Reporting Services Using SOAP](#)

Describes how to integrate Reporting Services report navigation and management into your existing business applications using the Report Server Web service.

[Integrating Reporting Services Using the Report Viewer Controls](#)

Describes how to integrate report viewing into your existing applications using the Report Viewer controls.

[Integrating Reporting Services Using URL Access](#)

Describes how to integrate Reporting Services report navigation into your existing business applications using URL access.

Next steps

For deciding on using URL access or the SOAP APIs, see [Choosing between URL access and SOAP in Reporting Services](#).

For information on the SQL Server 2017 Reporting Services REST API, see [Develop with the REST APIs for Reporting Services](#).

More questions? [Try asking the Reporting Services forum](#)

Report Server Web Service

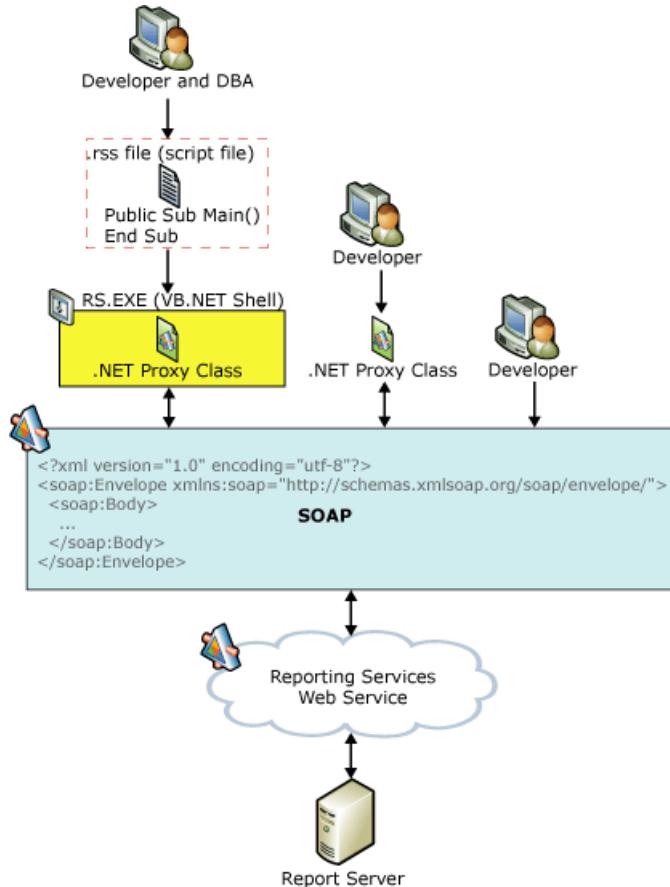
10/1/2018 • 2 minutes to read • [Edit Online](#)

SQL Server Reporting Services provides access to the full functionality of the report server through the Report Server Web service. The Report Server Web service is an XML Web service with a SOAP API. It uses SOAP over HTTP and acts as a communications interface between client programs and the report server. The Web service provides two endpoints - one for report execution and one for report management - with methods that expose the functionality of the report server and enable you to create custom tools for any part of the report life cycle.

There are three primary ways to develop Reporting Services applications based on the Web service. You can:

- Develop applications using Microsoft Visual Studio and the Microsoft .NET Framework SDK. For more information about using the .NET Framework to build Web service applications, see [Building Applications Using the Web Service and the .NET Framework](#).
- Develop applications using the **rs** utility (RS.exe), the Reporting Services script environment. With Reporting Services and Visual Basic scripts, you can run any of the Report Server Web service operations. For more information about scripting in Reporting Services, see [Script with the rs.exe Utility and the Web Service](#).
- Develop applications using any SOAP-enabled set of development tools. For more information, see [The Role of SOAP in Reporting Services](#).

Programming Diagram



Reporting Services available Web service development options

In This Section

[Report Server Web Service Methods](#)

Describes the features and methods of each Report Server Web service.

[The Role of SOAP in Reporting Services](#)

Provides an overview of SOAP and how it is used in the Report Server Web services.

[Accessing the SOAP API](#)

Describes the Web Service Description Language (WSDL) and provides URLs for accessing a Reporting Services WSDL file.

[Building Applications Using the Web Service and the .NET Framework](#)

Contains information about developing applications and Web services that call the Reporting Services SOAP API.

[Script with the rs.exe Utility and the Web Service](#)

Provides an overview of the Reporting Services scripting environment.

[Technical Reference \(SSRS\)](#)

Contains reference material specific to Report Server Web services methods and corresponding complex types.

User Requirements for Web Service Development

To develop applications using the Report Server Web service, you need:

- Microsoft Internet Explorer 5.5 or later installed on a computer with an Internet connection to and access to the report server.
- Microsoft Visual Studio or the Microsoft .NET Framework SDK installed on a computer if you want to develop and deploy Reporting Services applications using the Microsoft .NET Framework.
- An in-depth understanding of Microsoft SQL Server Reporting Services features and capabilities.
- A firm understanding of SOAP and XML Web Services.
- Development experience in a .NET Framework-compatible language such as Microsoft Visual C# or Microsoft Visual Basic, if you plan to use the .NET Framework as your development platform.

See Also

[Report Server Web Service](#)

Reporting Services Extension Library

10/1/2018 • 2 minutes to read • [Edit Online](#)

The Reporting Services Extension Library is a set of classes, interfaces, and value types that are included in Reporting Services. This library provides access to system functionality and is designed to be the foundation on which Microsoft .NET Framework applications can be used to extend Reporting Services components.

Namespaces

The Reporting Services extension library provides the following namespaces.

[Microsoft.ReportingServices.DataProcessing](#)

Contains classes and interfaces that enable you to build components that extend the data processing capability of Reporting Services.

[Microsoft.ReportingServices.Interfaces](#)

Contains classes and interfaces that enable you to construct and send custom notifications to users through your own delivery extensions, and to build custom security extensions for Reporting Services.

[**Microsoft.ReportingServices.ReportRendering**](#)

Contains classes and interfaces that enable you to extending the rendering capabilities of Reporting Services. Using the members of this namespace along with members of the [Microsoft.ReportingServices.Interfaces](#) namespace, you can build your own, custom rendering extensions for Reporting Services.

See Also

[Reporting Services Extensions](#)

Creating a Custom Report Item Design-Time Component

12/10/2018 • 5 minutes to read • [Edit Online](#)

A custom report item design-time component is a control that can be used in the Visual Studio Report Designer environment. The custom report item design-time component provides an activated design surface that can accept drag-and-drop operations, integration with the Visual Studio property browser, and the ability to provide custom property editors.

With a custom report item design-time component, the user can position a custom report item on a report in the design environment, set custom data properties on the custom report item, and then save the custom report item as part of the report project.

The properties that are set using the design-time component in the development environment are serialized and deserialized by the host design environment and then stored as elements in the Report Definition Language (RDL) file. When the report is executed by the report processor, the properties that are set using the design-time component are passed by the report processor to a custom report item run-time component, which renders the custom report item and passes it back to the report processor.

NOTE

The custom report item design-time component is implemented as a Microsoft.NET Framework component. This document will describe implementation details specific to the custom report item design-time component. For more information about developing components using the .NET Framework, see [Components in Visual Studio](#) in the MSDN library.

For a sample of a fully implemented custom report item, see [SQL Server Reporting Services Product Samples](#).

Implementing a Design-Time Component

The main class of a custom report item design-time component is inherited from the **Microsoft.ReportDesigner.CustomReportItemDesigner** class. In addition to the standard attributes used for a .NET Framework control, your component class should define a **CustomReportItem** attribute. This attribute must correspond to the name of the custom report item as it is defined in the reportserver.config file. For a list of .NET Framework attributes, see [Attributes](#) in the .NET Framework SDK documentation.

The following code example shows attributes being applied to a custom report item design-time control:

```
namespace PolygonsCRI
{
    [LocalizedName("Polygons")]
    [Editor(typeof(CustomEditor), typeof(ComponentEditor))]
    [ToolboxBitmap(typeof(PolygonsDesigner),"Polygons.ico")]
    [CustomReportItem("Polygons")]

    public class PolygonsDesigner : CustomReportItemDesigner
    {
        ...
    }
}
```

Initializing the Component

You pass user-specified properties for a custom report item using a **CustomData** class. Your implementation of the **CustomReportItemDesigner** class should override the **InitializeNewComponent** method to create a new

instance of your component's **CustomData** class and set it to default values.

The following code example shows an example of a custom report item design-time component class overriding the **CustomReportItemDesigner.InitializeNewItem** method to initialize the component's **CustomData** class:

```
public override void InitializeNewItem()
{
    CustomData = new CustomData();
    CustomData.DataRowHierarchy = new DataHierarchy();

    // Shape grouping
    CustomData.DataRowHierarchy.DataMembers.Add(new DataMember());
    CustomData.DataRowHierarchy.DataMembers[0].Group = new Group();
    CustomData.DataRowHierarchy.DataMembers[0].Group.Name = Name + "_Shape";
    CustomData.DataRowHierarchy.DataMembers[0].Group.GroupExpressions.Add(new ReportExpression());

    // Point grouping
    CustomData.DataRowHierarchy.DataMembers[0].DataMembers.Add(new DataMember());
    CustomData.DataRowHierarchy.DataMembers[0].DataMembers[0].Group = new Group();
    CustomData.DataRowHierarchy.DataMembers[0].DataMembers[0].Group.Name = Name + "_Point";
    CustomData.DataRowHierarchy.DataMembers[0].DataMembers[0].Group.GroupExpressions.Add(new ReportExpression());

    // Static column
    CustomData.DataColumnHierarchy = new DataHierarchy();
    CustomData.DataColumnHierarchy.DataMembers.Add(new DataMember());

    // Points
    IList<IList<DataValue>> dataValues = new List<IList<DataValue>>();
    CustomData.DataRows.Add(dataValues);
    CustomData.DataRows[0].Add(new List<DataValue>());
    CustomData.DataRows[0][0].Add(NewDataValue("X", ""));
    CustomData.DataRows[0][0].Add(NewDataValue("Y", ""));
}
}
```

Modifying Component Properties

You can modify **CustomData** properties in the design environment in several ways. You can modify any properties that are exposed by the design-time component that are marked with the **BrowsableAttribute** attribute by using the Visual Studio property browser. In addition, you can modify properties by dragging items onto the custom report item's design surface, or by right-clicking the control in the design environment and selecting **Properties** on the shortcut menu to display a custom properties window.

The following code example shows a **Microsoft.ReportDesigner.CustomReportItemDesigner.CustomData** property that has the **BrowsableAttribute** attribute applied:

```
[Browsable(true), Category("Data")]
public string DataSetName
{
    get
    {
        return CustomData.DataSetName;
    }
    set
    {
        CustomData.DataSetName = value;
    }
}
```

You can provide your design-time component with a custom properties editor dialog box. The custom property

editor implementation should inherit from the [ComponentEditor](#) class, and it should create an instance of a dialog box that can be used for property editing.

The following example shows an implementation of a class that inherits from [ComponentEditor](#) and displays a custom property editor dialog box:

```
internal sealed class CustomEditor : ComponentEditor
{
    public override bool EditComponent(
        ITypeDescriptorContext context, object component)
    {
        PolygonsDesigner designer = (PolygonsDesigner)component;
        PolygonProperties dialog = new PolygonProperties();
        dialog.m_designerComponent = designer;
        DialogResult result = dialog.ShowDialog();
        if (result == DialogResult.OK)
        {
            designer.Invalidate();
            designer.ChangeService().OnComponentChanged(designer, null, null, null);
            return true;
        }
        else
            return false;
    }
}
```

Your custom property editor dialog box can invoke the Report Designer Expression Editor. In the following example, the Expression Editor is invoked when the user selects the first element in the combo box:

```
private void EditableCombo_SelectedIndexChanged(object sender,
    EventArgs e)
{
    ComboBox combo = (ComboBox)sender;
    if (combo.SelectedIndex == 0 && m_launchEditor)
    {
        m_launchEditor = false;
        ExpressionEditor editor = new ExpressionEditor();
        string newValue;
        newValue = (string)editor.EditValue(null, m_designerComponent.Site, m_oldComboValue);
        combo.Items[0] = newValue;
    }
}
```

Using Designer Verbs

A designer verb is a menu command linked to an event handler. You can add designer verbs that will appear in a component's shortcut menu when your custom report item run-time control is being used in the design environment. You can return the list of available designer verbs from your run-time component by using the **Verbs** property.

The following code example shows a designer verb and an event handler being added to the [DesignerVerbCollection](#), as well as the event handler code:

```

public override DesignerVerbCollection Verbs
{
    get
    {
        if (_verbs == null)
        {
            _verbs = new DesignerVerbCollection();
            _verbs.Add(new DesignerVerb("Proportional Scaling", new EventHandler(OnProportionalScaling)));
            _verbs[0].Checked = (GetCustomProperty("poly:Proportional") == bool.TrueString);
        }

        return _verbs;
    }
}

private void OnProportionalScaling(object sender, EventArgs e)
{
    bool proportional =
        (GetCustomProperty("poly:Proportional") == bool.TrueString);
    _verbs[0].Checked = proportional;
    SetCustomProperty("poly:Proportional", proportional.ToString());
    ChangeService().OnComponentChanged(this, null, null, null);
    Invalidate();
}

```

Using Adornments

Custom report item classes can also implement a **Microsoft.ReportDesigner.Design.Adornment** class. An adornment allows the custom report item control to provide areas outside the main rectangle of the design surface. These areas can handle user interface events, such as mouse clicks and drag-and-drop operations. The **Adornment** class that is defined in the Reporting Services **Microsoft.ReportDesigner** namespace is a pass-through implementation of the [Adorner](#) class found in Windows Forms. For complete documentation on the **Adorner** class, see [Behavior Service Overview](#) in the MSDN library. For sample code that implements a **Microsoft.ReportDesigner.Design.Adornment** class, see [SQL Server Reporting Services Product Samples](#).

For more information about programming and using Windows Forms in Visual Studio, see these topics in the MSDN Library:

- Design-Time Attributes for Components
- Components in Visual Studio
- Walkthrough: Creating a Windows Forms Control that Takes Advantage of Visual Studio Design-Time Features

See Also

[Custom Report Item Architecture](#)

[Creating a Custom Report Item Run-Time Component](#)

[Custom Report Item Class Libraries](#)

[How to: Deploy a Custom Report Item](#)

Accessing Custom Assemblies Through Expressions

10/1/2018 • 2 minutes to read • [Edit Online](#)

Once you have created a custom assembly, made it available to Report Designer or the report server, added the appropriate security policy, and added a reference to your custom assembly in your report definition, you can access the members of the classes in your assembly using report expressions. To refer to custom code in an expression, you must call the member of a class within the assembly. How you do this depends on whether the method is static or instance-based.

Calling Static Members from a Report Definition File

Static members belong to the class or type itself and not to an instantiated object. These members can be accessed by directly calling them from the class. You should use static members to call custom functions in a report whenever possible, because static members perform best. To call a static member, you need to reference it as an expression that takes the form `=Namespace.Class.Method`.

To call static members

- To call a static member, set your expression equal to the fully qualified name of the member, which includes the namespace, class name, and member name. The following example calls the **ToGBP** method, which converts the **StandardCost** field value from dollars to pounds sterling and displays it in a report:

```
=CurrencyConversion.DollarCurrencyConversion.ToGBP(Fields!StandardCost.Value)
```

Important Information Regarding Static Fields and Properties

Currently, all reports are executed in the same application domain. This means that reports with user-specific, static data expose this data to other instances of the same report. This condition might make it possible for the static data of one user to be available to all users currently running a particular report. For this reason, it is highly recommended that you not use static fields or properties in custom assemblies or in the **Code** element; instead, use instance fields or properties in your reports. Static methods can still be used, because they do not store state or data.

Calling Instance Members from a Report Definition File

If your custom assembly contains instance members that you need to access in a report definition, you must add an instance name for your class to the report. You can add an instance name for a class using the **Code** tab of the **Report Properties** dialog. For more information about adding instances of classes to a report, see [Custom Code and Assembly References in Expressions in Report Designer \(SSRS\)](#).

To call a static member, you need to reference it as an expression that takes the form `=Code*.InstanceName.Method*`.

To call instance members

- To call an instance member of a custom assembly, you must reference the **Code** keyword followed by the instance name and the method. The following example calls an instance method **ToEUR** which converts the **StandardCost** field value from dollars to euros and displays it in a report:

```
=Code.m_myDollarConversion.ToEUR(Fields!StandardCost.Value)
```

See Also

[Using Custom Assemblies with Reports](#)

Technical Reference (SSRS)

10/24/2018 • 2 minutes to read • [Edit Online](#)

Find the tools and PowerShell reference documentation for using or administering SQL Server Reporting Services.



Errors and Events

[Cause and Resolution of Reporting Services Errors](#)



Feature Reference

[Report Designer F1 Help](#)

[Report Manager F1 Help](#)

[Reporting Services Configuration Manager \(Native Mode\)](#)

[Report Wizard Help](#)

[HTML Viewer and the Report Toolbar](#)

[Device Information Settings for Rendering Extensions \(Reporting Services\)](#)



Report Server Command Prompt Utilities

[RS.exe Utility \(SSRS\)](#)

[rsconfig Utility \(SSRS\)](#)

[rskeymgmt Utility \(SSRS\)](#)



Reporting Services WMI Provider Class Library

[Reporting Services WMI Provider Library Reference \(SSRS\)](#)

More questions? [Try asking the Reporting Services forum](#)

Feature Reference (Reporting Services)

11/15/2018 • 2 minutes to read • [Edit Online](#)

SQL Server Reporting Services includes several tools and applications that you can use to create, manage, and view reports. This section provides specific topics describing the dialog boxes, Web pages, and wizards of these tools and applications.

To access a user interface topic while the tool or application is running, press F1 or click **Help** while the dialog box, Web page, or wizard is open. For more information about starting the Reporting Services tools, see [Tutorial: How to Locate and Start Reporting Services Tools \(SSRS\)](#).

In This Section

[Report Designer F1 Help](#)

Provides help for Report Designer and its related dialog boxes, Report Wizard, and Image Wizard.

[Reporting Services Configuration Manager Help Topics](#)

Provides help for the Reporting Services Configuration tool used for deploying and managing report server instances.

[Report Server in Management Studio F1 Help](#)

Provides help for dialog boxes used to manage reports and report servers in SQL Server Management Studio.

[HTML Viewer and the Report Toolbar](#)

Provides help for the HTML Viewer component used for viewing reports in a Web browser.

See Also

[Reporting Services Concepts \(SSRS\)](#)

[Reporting Services Reports \(SSRS\)](#)

[Report Datasets \(SSRS\)](#)

[Getting Started with Report Builder](#)

[Designing and Implementing Reports Using Report Builder 1.0](#)

Device Information Settings for Rendering Extensions (Reporting Services)

10/1/2018 • 2 minutes to read • [Edit Online](#)

In Reporting Services, device information settings are used to pass rendering parameters to a rendering extension. Each rendering extension accepts a specific set of settings.

In This Section

TOPIC	DESCRIPTION
ATOM Device Information Settings	Describes the device information settings that are associated with Atom compliant rendering output.
CSV Device Information Settings	Describes the device information settings that are associated with CSV rendering output.
Excel Device Information Settings	Describes the device information settings that are associated with Excel rendering output.
Word Device Information Settings	Describes the device information settings that are associated with Word rendering output.
HTML Device Information Settings	Describes the device information settings that are associated with HTML rendering output.
Image Device Information Settings	Describes the device information settings that are associated with IMAGE rendering output.
MHTML Device Information Settings	Describes the device information settings that are associated with MHTML rendering output.
PDF Device Information Settings	Describes the device information settings that are associated with PDF rendering output.
XML Device Information Settings	Describes the device information settings that are associated with XML rendering output.
RGDI Device Information Settings	Describes the device information settings that are associated with RGDI rendering output.
PPTX Device Information Settings	Describes the device information settings that are associated with PPTX rendering output.

See Also

[Customize Rendering Extension Parameters in RSReportServer.Config](#)

ATOM Device Information Settings

10/1/2018 • 2 minutes to read • [Edit Online](#)

The device information settings for the Atom rendering extension support submittal of the name of an Atom feed and character encoding to use.

The following table lists the device information settings for rendering to a data service document.

SETTING	VALUE
DataFeed	If specified, renders the Atom feed corresponding to the feed name provided in this setting. If not, renders the Atom service document for the report. The unique auto-generated identifier of the data feed. This value is used internally and you should not change it.
Encoding	The Internet Assigned Numbers Authority (IANA) name of a character encoding that is supported by the .NET Framework. The default value is UTF-8 . Examples of other values include ASCII, UTF-7, and UTF-16.

See Also

[Render](#)

[Passing Device Information Settings to Rendering Extensions](#)

[Customize Rendering Extension Parameters in RSReportServer.Config](#)

[Technical Reference \(SSRS\)](#)

CSV Device Information Settings

10/1/2018 • 2 minutes to read • [Edit Online](#)

The device information settings for the CSV rendering extension allow delimiters and qualifiers to be changed and line break handling to be specified. The extension of the file can also be submitted, as well as the encoding and inclusion of header rows in the output. Because delimiters are likely to be special characters, you should encode them in a CDATA section, if the settings are written as XML.

The following table lists the device information settings for rendering in Text format.

SETTING	VALUE
Encoding	The Internet Assigned Numbers Authority (IANA) name of a character encoding that is supported by the .NET Framework. The default value is UTF-8 . Examples of other values include ASCII, UTF-7, and UTF-16.
ExcelMode	Specifies that the target output is for Excel. The default value is true .
FieldDelimiter	<p>The delimiter string to put in the result. The default value is a comma (,). You should URL encode the value of this device information when passing it on a URL. For example, a tab character as a delimiter should be "%09".</p> <p>You can change the default field delimiter to any character that you want, including TAB, by changing the device information settings in the configuration file. For example, to use TAB, update the FieldDelimiter setting to <FieldDelimiter xml:space="preserve">[TAB]</FieldDelimiter></p> <p>In the example [TAB] is an actual tab character, which means that whitespace appears in the configuration file. The "xml:space" attribute tells parsers to preserve whitespace.</p>
FileExtension	The file extension to put on the result. The default value is .CSV . If both FileExtension and Extension are specified then FileExtension will take precedence.
NoHeader	Indicates whether the header row is excluded from the output. The default value is false .
Qualifier	The qualifier string to put around results that contain the field delimiter or record delimiter. If the results contain the qualifier, the qualifier is repeated. The Qualifier setting must be different from the FieldDelimiter and RecordDelimiter settings. The default value is a quotation mark ("").
RecordDelimiter	The record delimiter to put at the end of each record. The default value is <cr><lf>.

SETTING	VALUE
SuppressLineBreaks	Indicates whether line breaks are removed from the data included in the output. The default value is false . If the value is true , the FieldDelimiter , RecordDelimiter , and Qualifier settings cannot be a space character.
UseFormattedValues	Indicates whether formatted strings are put into the CSV output. The default value is true when ExcelMode is true ; otherwise it is false .

See Also

[Render](#)

[Passing Device Information Settings to Rendering Extensions](#)

[Customize Rendering Extension Parameters in RSReportServer.Config](#)

[Technical Reference \(SSRS\)](#)

Excel Device Information Settings

10/1/2018 • 2 minutes to read • [Edit Online](#)

The following table lists the device information settings for rendering in Microsoft Excel format.

SETTING	VALUE
OmitDocumentMap	Indicates whether to omit the document map for reports that support it. The default value is false .
OmitFormulas	Indicates whether to omit formulas from the rendered report. The default value is false .
SimplePageHeaders	Indicates whether the page header of the report is rendered to the Excel page header. A value of false indicates that the page header is rendered to the first row of the worksheet. The default value is false .

See Also

[Render](#)

[Passing Device Information Settings to Rendering Extensions](#)

[Customize Rendering Extension Parameters in RSReportServer.Config](#)

[Technical Reference \(SSRS\)](#)

HTML Device Information Settings

11/27/2018 • 5 minutes to read • [Edit Online](#)

The following table lists the device information settings for rendering in HTML format.

IMPORTANT

The device information settings listed in the table below with a (*) have been deprecated and they should not be used in new applications. For more information, see [Deprecated Features in SQL Server Reporting Services in SQL Server 2016](#)

SETTING	VALUE
AccessibleTablix	Indicates whether to render with additional accessibility metadata for use with screen readers. The additional accessibility metadata causes the rendered report to be compliant with the following technical standards in the "Web-based Intranet and Internet Information and Applications" section (1194.22) of the Electronic and Information Technology Accessibility Standards (Section 508) document: (g) Row and column headers shall be identified for data tables. (h) Markup shall be used to associate data cells and header cells for data tables that have two or more logical levels of row or column headers.
ActionScript(*)	Specifies the name of the JavaScript function to use when an action event occurs, such as a drillthrough or bookmark click. If this parameter is specified, an action event will trigger the named JavaScript function instead of a postback to the server.
BookmarkID	The bookmark ID to jump to in the report.
DocMap	Indicates whether to show or hide the report document map. The default value of this parameter is true .
ExpandContent	Indicates whether the report should be enclosed in a table structure which constricts horizontal size.
FindString	The text to search for in the report. The default value of this parameter is an empty string.
GetImage (*)	Gets a particular icon for the HTML Viewer user interface.

Setting	Value
HTMLFragment	Indicates whether an HTML fragment is created in place of a full HTML document. An HTML fragment includes the report content in a TABLE element and omits the HTML and BODY elements. The default value is false . If you are rendering to HTML using the M:ReportExecution2005.ReportExecutionService.Render(System.String,System.String,System.String@,System.String@,System.String@,System.String@,ReportExecution2005.Warning[],System.String[]) method of the SOAP API, you need to set this device information to true if you are rendering a report with images. Rendering using SOAP with the HTMLFragment property set to true creates URLs containing session information that can be used to properly request images. The images must be uploaded resources in the report server database.
ImageConsolidation	Indicates whether to consolidate the rendered chart, map, gauge, and indicator images into one large image. The consolidation of images helps improve the performance of the report in the client browser when the report contains many data visualization items. The default value is true for most modern browsers.
JavaScript	Indicates whether JavaScript is supported in the rendered report. The default value is true .
LinkTarget	The target for hyperlinks in the report. You can target a window or frame by providing the name of the window, like LinkTarget=window_name , or you can target a new window using LinkTarget=_blank . Other valid target names include _self , _parent , and _top .
OnlyVisibleStyles(*)	Indicates that only shared styles for currently rendered page are generated.
OutlookCompat	Indicates whether to render with extra metadata that makes the report look better in Outlook. For others, the default value is false .
Parameters	Indicates whether to show or hide the parameters area of the toolbar. If you set this parameter to a value of true , the parameters area of the toolbar is displayed. The default value of this parameter is true .
PrefixId	When used with HTMLFragment , adds the specified prefix to all ID attributes in the HTML fragment that is created.
ReplacementRoot(*)	The string to prepend to all drillthrough, toggle, and bookmark links in the report when rendered outside of the ReportViewer control. For example, this is used for redirecting a user's click to a custom page.
ResourceStreamRoot(*)	The string to prepend to the URL for all image resources, such as images for toggle or sort.

SETTING	VALUE
Section	The page number of the report to render. A value of 0 indicates that all sections of the report are rendered. The default value is 1 .
StreamRoot (*)	The path used for prefixing the value of the src attribute of the IMG element in the HTML report returned by the report server. By default, the report server provides the path. You can use this setting to specify a root path for the images in a report (for example, <a href="https://<servername>/resources/companyimages">https://<servername>/resources/companyimages).
StyleStream	Indicates whether styles and scripts are created as a separate stream instead of in the document. The default value is false .
Toolbar	Indicates whether to show or hide the toolbar. The default of this parameter is true . If the value of this parameter is false , all remaining options (except the document map) are ignored. If you omit this parameter, the toolbar is automatically displayed for rendering formats that support it. The Report Viewer toolbar is rendered when you use URL access to render a report. The toolbar is not rendered through the SOAP API. However, the Toolbar device information setting affects the way that the report is displayed when using the SOAP Render method. If the value of this parameter is true when using SOAP to render to HTML, only the first section of the report is rendered. If the value is false , the entire HTML report is rendered as a single HTML page.
UserAgent	The user-agent string of the browser that is making the request, which is found in the HTTP request.
Zoom (*)	The report zoom value as an integer percentage or a string constant. Standard string values include Page Width and Whole Page . This parameter is ignored by versions of Microsoft Internet Explorer earlier than Internet Explorer 5.0 and all non-Microsoft browsers. The default value of this parameter is 100 .
DataVisualizationFitSizing	Indicates data visualization fit behavior when inside a tablix. This includes chart, gauge, and map. The possible values are Approximate and Exact . The default value is Approximate . If the setting is removed from the rsreportserver.config file then the default behavior is Exact . Enabling Exact may have performance impact because the processing to determine the exact size may take longer.

See Also

- [Passing Device Information Settings to Rendering Extensions](#)
- [Customize Rendering Extension Parameters in RSReportServer.Config](#)
- [Technical Reference \(SSRS\)](#)

Image Device Information Settings

10/1/2018 • 2 minutes to read • [Edit Online](#)

The following table lists the device information settings for rendering in IMAGE format.

SETTING	VALUE
Columns	The number of columns to set for the report. This value overrides the report's original settings.
ColumnSpacing	The column spacing to set for the report. This value overrides the report's original settings.
DpiX	The horizontal resolution of the output image. The default value is 96 . Applies to BMP , GIF , PNG , and TIFF output formats.
DpiY	The vertical resolution of the output image. The default value is 96 . Applies to BMP , GIF , PNG , and TIFF output formats.
EndPage	The last page of the report to render. The default value is the value for StartPage .
MarginBottom	The bottom margin value, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 1in). This value overrides the report's original settings.
MarginLeft	The left margin value, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 1in). This value overrides the report's original settings.
MarginRight	The right margin value, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 1in). This value overrides the report's original settings.
MarginTop	The top margin value, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 1in). This value overrides the report's original settings.
OutputFormat	One of the Graphics Device Interface (GDI) supported output formats: BMP , EMF , GIF , JPEG , PNG , or TIFF .
PageHeight	The page height, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 11in). This value overrides the report's original settings.

SETTING	VALUE
PageWidth	The page width, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 8.5in). This value overrides the report's original settings.
PrintDpiX	The horizontal resolution of the output image. The default value is 300 . Applies to the Enhanced MetaFile (EMF) output format.
PrintDpiY	The vertical resolution of the output image. The default value is 300 . Applies to the Enhanced MetaFile (EMF) output format.
StartPage	The first page of the report to render. A value of 0 indicates that all pages are rendered. The default value is 1 .

See Also

[Render](#)

[Passing Device Information Settings to Rendering Extensions](#)

[Customize Rendering Extension Parameters in RSReportServer.Config](#)

[Technical Reference \(SSRS\)](#)

MHTML Device Information Settings

10/1/2018 • 2 minutes to read • [Edit Online](#)

The following table lists the device information settings for rendering reports in Web archive (MHTML) format.

Setting	Value
JavaScript	Indicates whether JavaScript is supported in the rendered report.
OutlookCompat	Indicates whether to render with extra metadata that makes the report look better in Outlook. The default value is true .
MHTML Fragment	Indicates whether an MHTML fragment is created in place of a full MHTML document. An MHTML fragment includes the report content in a TABLE element and omits the HTML and BODY elements. The default value is false .
DataVisualizationFitSizing	<p>Indicates data visualization fit behavior when inside a tablix. This includes chart, gauge, and map.</p> <p>The possible values are Approximate and Exact.</p> <p>The default value is Approximate. If the setting is removed from the rsreportserver.config file then the default behavior is Exact.</p> <p>Enabling Exact may have performance impact because the processing to determine the exact size may take longer.</p>

See Also

Render

[Passing Device Information Settings to Rendering Extensions](#)

[Customize Rendering Extension Parameters in RSReportServer.Config](#)

[Technical Reference \(SSRS\)](#)

PDF Device Information Settings

10/1/2018 • 2 minutes to read • [Edit Online](#)

The following table lists the device information settings for rendering reports in PDF format.

SETTING	VALUE
AccessiblePDF	Indicates whether to render an accessible/tagged PDF, which is larger in size but easier for screen readers and other assistive technologies to read and navigate. The default value is false . [Available in Power BI Report Server (March 2018) and later]
Columns	The number of columns to set for the report. This value overrides the report's original settings.
ColumnSpacing	The column spacing to set for the report. This value overrides the report's original settings.
DpiX	The resolution of the output device in x-direction.
DpiY	The resolution of the output device in y-direction.
EndPage	The last page of the report to render. The default value is the value for StartPage .
HumanReadablePDF	Indicates whether to render an uncompressed PDF file, which is larger in size but more human-readable in a plain-text editor. The default value is false .
MarginBottom	The bottom margin value, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 1in). This value overrides the report's original settings.
MarginLeft	The left margin value, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 1in). This value overrides the report's original settings.
MarginRight	The right margin value, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 1in). This value overrides the report's original settings.
MarginTop	The top margin value, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 1in). This value overrides the report's original settings.

SETTING	VALUE
PageHeight	The page height, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 11in). This value overrides the report's original settings.
PageWidth	The page width, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 8.5in). This value overrides the report's original settings.
StartPage	The first page of the report to render. A value of 0 indicates that all pages are rendered. The default value is 1 .

See Also

[Passing Device Information Settings to Rendering Extensions](#)

[Customize Rendering Extension Parameters in RSReportServer.Config](#)

[Technical Reference \(SSRS\)](#)

PPTX Device Information Settings

11/27/2018 • 2 minutes to read • [Edit Online](#)

The following table lists the device information settings for rendering Reporting Services reports in the PPTX format.

Setting	Value
Columns	The number of columns to set for the report. This value overrides the report's original settings.
ColumnSpacing	The column spacing to set for the report. This value overrides the report's original settings.
DpiX	The horizontal resolution of the output image. The default value is 96 . Applies to BMP , GIF , PNG , and TIFF output formats.
DpiY	The vertical resolution of the output image. The default value is 96 . Applies to BMP , GIF , PNG , and TIFF output formats.
EndPage	The last page of the report to render. The default value is the value for StartPage .
MarginBottom	The bottom margin value, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 1in). This value overrides the report's original settings.
MarginLeft	The left margin value, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 1in). This value overrides the report's original settings.
MarginRight	The right margin value, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 1in). This value overrides the report's original settings.
MarginTop	The top margin value, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 1in). This value overrides the report's original settings.
PageHeight	The page height, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 11in). This value overrides the report's original settings.
PageWidth	The page width, in inches, to set for the report. You must include an integer or decimal value followed by "in" (for example, 8.5in). This value overrides the report's original settings.

SETTING	VALUE
StartPage	The first page of the report to render. A value of 0 indicates that all pages are rendered. The default value is 1 .
UseReportPageSize	If UseReportPageSize =false then the default slide size is PowerPoint's default of 13.333" x 7.5" (16:9 aspect ratio). If UseReportPageSize =true , then the default slide size is the define page size of the report. The default value is false Note, the PageWidth and PageHeight settings override the default width and height.

See Also

[Render](#)

[Passing Device Information Settings to Rendering Extensions](#)

[Customize Rendering Extension Parameters in RSReportServer.Config](#)

[Technical Reference \(SSRS\)](#)

XML Device Information Settings

11/27/2018 • 2 minutes to read • [Edit Online](#)

The following table lists the device information settings for rendering in XML format.

Setting	Values	Details
XSLT	The path in the report server namespace of an XSLT to apply to the XML file, for example /Transforms/myxslt .	The xsl file must be a published resource on the report server and you must access it through a report server item path. The value of this setting is applied after any XSLT that is specified in the report. If the XSLT setting is applied, the OmitSchema setting is ignored.
MIMEType	The Multipurpose Internet Mail Extensions (MIME) type of the XML file.	
UseFormattedValues	true false	Indicates whether to render the formatted value of a text box when generating the XML data. A value of false indicates that the underlying value of the text box is used.
Indented	true false	Indicates whether to generate indented XML. The default value of false generates non-indented, compressed XML.
OmitNamespace	true false	Indicates whether to omit the default namespace from the XML. If true, the XML does not specify a default namespace. If false, the XML specifies a default namespace with the value of the report's DataSchema property. The DataSchema property defaults to the report name. The default value is false .

SETTING	VALUES	DETAILS
OmitSchema	true false	<p>Indicates whether to omit the schema location from the XML. The location is the SchemaLocation attribute.</p> <p>The default value of OmitSchema depends on the value of OmitNamespace:</p> <p>If OmitNamespace = False, then OmitSchema = False by default. The user can override the default by setting OmitSchema = True.</p> <p>If OmitNamespace = True, then OmitSchema will function as True regardless of the value explicitly configured for OmitSchema.</p>
Encoding	The Internet Assigned Numbers Authority (IANA) name of a character encoding that is supported by the .NET Framework.	The default value is UTF-8 . Examples of other values include ASCII, UTF-7, and UTF-16.
FileExtension	The file extension to use for the generated file.	
Schema	A value of true indicates that an XML schema is rendered. The default value is false .	Indicates whether the XML schema definition (XSD) is rendered or whether the actual XML data is rendered.

See Also

[Render](#)

[Passing Device Information Settings to Rendering Extensions](#)

[Customize Rendering Extension Parameters in RSReportServer.Config](#)

[Technical Reference \(SSRS\)](#)

Word Device Information Settings

10/1/2018 • 2 minutes to read • [Edit Online](#)

The following table lists the device information settings for rendering in Microsoft Word format.

SETTING	VALUE
AutoFit	False. AutoFit is set to false set on any Word table. True. AutoFit is set to true on every Word table. Never. AutoFit values are not set on any Word table and behavior reverts to the Word default. Default. AutoFit is set on tables that are narrower than the physical drawing area (physical page width excluding margins) per logical page.
ExpandToggles	Indicates whether all items that can be toggled should render in their fully-expanded state. The default value is false .
FixedPageWidth	Indicates whether the Page Width written to the DOC file will grow to accommodate the width of the largest page in the Report Body. The default value is false .
OmitHyperlinks	Indicates whether to omit the Hyperlink action on all items where it is set. The default value is false
OmitDrillthroughs	Indicates whether to omit the Drillthrough action on all items where it is set. The default value is false

See Also

[Render](#)

[Passing Device Information Settings to Rendering Extensions](#)

[Customize Rendering Extension Parameters in RSReportServer.Config](#)

[Technical Reference \(SSRS\)](#)

RGDI Device Information Settings

10/1/2018 • 2 minutes to read • [Edit Online](#)

The following table lists the device information settings for rendering in RGDI format.

SETTING	VALUE
Page	Specifies page of the report to render. The default value is 1 .
RGDIVersion	This setting has no effect.

See Also

[Passing Device Information Settings to Rendering Extensions](#)

[Customize Rendering Extension Parameters in RSReportServer.Config](#)

[Technical Reference \(SSRS\)](#)

HTML Viewer and the Report Toolbar

10/1/2018 • 6 minutes to read • [Edit Online](#)

SQL Server Reporting Services provides an HTML Viewer that is used to display reports on demand as they are requested from the report server. HTML Viewer provides a framework for viewing reports in HTML. It includes a report toolbar, a parameter section, a credentials section, and a document map. The report toolbar in HTML Viewer includes features you can use to work with your report, including export options so that you can view your report in formats other than HTML. The parameter section and document map appear only when you open reports that are configured to use parameters and a document map control.

Although you cannot modify the report toolbar, you can configure parameters on a report URL to hide it on a report. For more information about hiding the report toolbar, see [URL Access Parameter Reference](#).

Report Toolbar

The report toolbar provides page navigation, zoom, refresh, search, export, print, and data feed functionality for reports that are rendered in the HTML rendering extension.

Print functionality is optional. When it is available, a Printer icon appears on the report toolbar. On first use, clicking the Printer icon downloads an ActiveX control that you must install. Once the control is installed, clicking the Printer icon opens a Print dialog box so that you can select from the printers that are configured for your computer. Print availability is determined by server settings and browser settings. For more information, see [Print Reports from a Browser with the Print Control \(Report Builder and SSRS\)](#) and [Enable and Disable Client-Side Printing for Reporting Services](#).

The report toolbar is similar to the one shown in the following illustration. The report toolbar that you see may differ from the illustration based on report features or the rendering options that are available.



The following table describes commonly used features of the report toolbar. Each feature is identified by the control that you use to access it.

USE THIS ICON OR CONTROL	TO
	Page navigation controls Open the first or last page of a report, scroll through a report page by page, and open a specific page in a report. To view a specific page, type the page number and press ENTER.
	Page display controls Enlarge or reduce the size of the report page. In addition to percentage-based changes, you can select Page Width to fit the horizontal length of a report page in the browser window, or Whole Page to fit the vertical length of a report in the browser window. A Zoom option is supported by Microsoft Internet Explorer 5.5 and later.

USE THIS ICON OR CONTROL		TO
 Find Next	Search field	Search for content in the report by typing a word or phrase that you want to find (the maximum value length is 256 characters). The search is case-insensitive and starts at the page or section that is currently selected. Only visible content is included in a search operation. To search for subsequent occurrences of the same value, click Next .
	Export formats	Open a new browser window and render the report in the selected format. The formats that are available are determined by the rendering extensions that are installed on the report server. TIFF is recommended for printing. Click Export to view the report in the selected format.
	Document map icon	Show or hide the document map pane in a report that includes a document map. A document map is a report navigation control similar to the navigation pane on a Web site. You can click on items in the document map to navigate to a specific group, page, or subreport.
	Printer icon	Open a Print dialog box so that you can specify print options and print a report. On first use, clicking this icon prompts you to download the print control.
	Show and hide icons	Show or hide parameter value fields and the View Report button in a report that includes parameters.
	Report refresh icon	Refresh the report. Data for live reports will be refreshed. Cached reports will be reloaded from where they are stored.
	Data feed icon	Generated data feeds from reports.
	Pin to Power BI Dashboard	Pin support report items to a Power BI. If the button is not visible, the report server has not been integrated with Power BI. For more information, see Power BI Report Server Integration (Configuration Manager) .

About Export Formats

From the report toolbar, you can select to view your report in a variety of formats. The formats that are available are determined by the rendering extensions that are installed on the report server. When you select another format, a second browser window is used to display the report, using a viewer associated with the export format you selected. If a viewer is not available for the format you select, you can select a different format.

The following export formats are included in a default report server installation. The list of export formats available to you may vary from those listed here.

EXPORT FORMAT	DESCRIPTION
XML	View a report in XML syntax. Reports viewed in XML open in a new browser window.
CSV	View a report in a comma-delimited format. The report opens in an application that is associated with the CSV file type.
PDF	View a report using a client-side PDF viewer. You must have third-party PDF viewer (for example, Adobe Acrobat Reader) to use this format.
MHTML	View the report in an MIME-encoded HTML format that keeps images and linked content together with a report.
Excel	View the report in Microsoft Excel, an .xlsx file.
PowerPoint	View the report in Microsoft PowerPoint, a .pptx file.
TIFF file	View the report in the default TIFF viewer. For some Microsoft Windows clients, this is the Windows Picture and Fax Viewer. Select this format to view a report in a page-oriented layout.
Word	View the report in Microsoft Word, a .docx file.

Parameters

Parameters are values that are used to select specific data (specifically, they are used to complete a query that selects the data for your report, or to filter the result set). Parameters that are commonly used in reports include dates, names, and IDs. When you specify a value for a parameter, the report contains only the data that matches the value; for example, employee data based on an Employee ID parameter. Parameters correspond to fields on the report. After you specify a parameter, click **View Report** to get the data.

The report author defines the parameter values that are valid for each report. A report administrator can also set parameter values. To find out which parameter values are valid for your report, ask your report designer or administrator.

Credentials

Credentials are user name and password values that grant access to a data source. After you specify your credentials, click **View Report** to get the data. If a report requires you to log on, the data that you are authorized to see might differ from the data that another user sees. Consequently, two users can run the same report and get different results. In addition, some reports contain hidden areas that are revealed based on user logon credentials or selections made in the report itself. Hidden areas in the report are excluded from search operations, producing different search results than when all parts of the report are visible.

See Also

- [Specify Credential and Connection Information for Report Data Sources](#)
- [Finding, Viewing, and Managing Reports \(Report Builder and SSRS \)](#)
- [Export Reports \(Report Builder and SSRS\)](#)

Reporting Services Tutorials (SSRS)

10/24/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO:  SQL Server 2016  Power BI Report Server

Explore the capabilities of SQL Server 2016 Reporting Services (SSRS), SQL Server Data Tools (SSDT), and Report Builder with sample data in these tutorials.

Create a Basic Table Report

Follow the steps in this tutorial to learn how to create your first report. This tutorial shows you how to work with SQL Server Data Tools (SSDT) to create a data connection (data source), define a simple query (dataset), and configure a data region to contain your data.

Create a Data-Driven Subscription

Reporting Services provides data-driven subscriptions so that you can customize the distribution of a report based on dynamic list of subscribers that will receive the report. Data-driven subscriptions are typically created and maintained by report server administrators. The ability to create data-driven subscriptions requires expertise in building queries, knowledge of data sources that contain subscriber data, and elevated permissions on a report server.

Create a Drillthrough (RDLC) Report with Parameters using ReportViewer

Follow the steps in this tutorial to learn how to create a drillthrough report with parameters and a filter using the ReportViewer control.

Report Builder Tutorials

These tutorials introduce you a variety of visualizations you can create in Report Builder, such as maps and sparklines, as well as tutorials on how to use parameters and expressions.

See Also

- [AdventureWorks sample databases](#)
- [Reporting Services Samples on the TechNet wiki](#)
- [TechNet Wiki: SQL Server 2012 Samples](#)

Create a Basic Table Report (SSRS Tutorial)

12/18/2018 • 2 minutes to read • [Edit Online](#)

In this tutorial, you use Report Designer in SQL Server Data Tools to create a basic Reporting Services paginated report with a table, based on the **AdventureWorks2014** database. You can also create Reporting Services paginated reports with Report Builder.

As you go through this tutorial, you will create a report project, set up connection information, define a query, add a Table data region, group and total some fields, and preview the report.

Requirements

Your system must have the following installed to use this tutorial:

- Microsoft SQL Server database engine.
- SQL Server 2016 Reporting Services (SSRS) in native mode.
- The **AdventureWorks2014** database. For more information, see [Adventure Works 2014 Sample Databases](#).
- [SQL Server Data Tools](#) with the "SQL Server Reporting Services" components installed so you have the Report Designer.

You must also have read-only permissions to retrieve data from the **AdventureWorks2014** database.

Estimated time to complete the tutorial: 30 minutes.

Tasks

[Lesson 1: Creating a Report Server Project \(Reporting Services\)](#)

[Lesson 2: Specifying Connection Information \(Reporting Services\)](#)

[Lesson 3: Defining a Dataset for the Table Report \(Reporting Services\)](#)

[Lesson 4: Adding a Table to the Report \(Reporting Services\)](#)

[Lesson 5: Formatting a Report \(Reporting Services\)](#)

[Lesson 6: Adding Grouping and Totals \(Reporting Services\)](#)

Next steps

[Reporting Services Tutorials](#)

More questions? Try asking the [Reporting Services forum](#)

Lesson 1: Creating a Report Server Project (Reporting Services)

10/17/2018 • 2 minutes to read • [Edit Online](#)

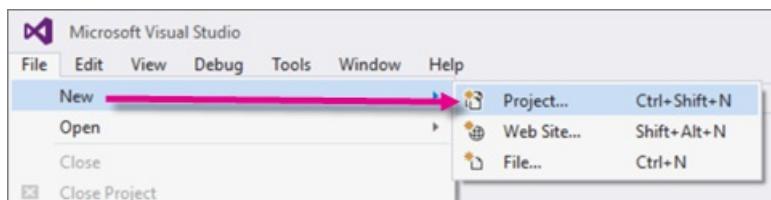
In this lesson, you'll create a *report server project* and a *report definition (.rdl)* file in SQL Server Data Tools within Visual Studio.

To create a report with SQL Server Data Tools (SSDT), first you need a report server project where you can save your report definition (.rdl) file and other resource files you need for your report.

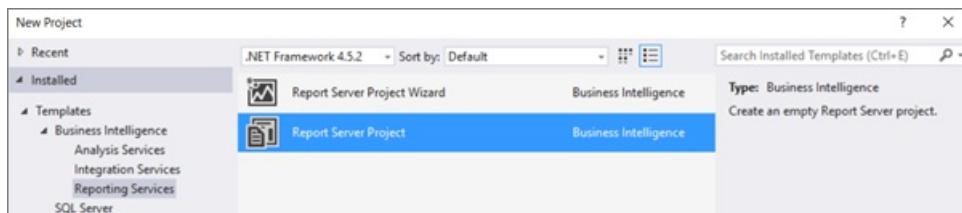
In the following lessons, you define a data source for your report, define a dataset, and define the report layout. When you run the report, the data is retrieved and combined with the layout, and then rendered on your screen. From there you can export it, print it, or save it.

To create a report server project

1. Open SQL Server Data Tools.
2. On the **File** menu > **New > Project**.



3. Under **Installed > Templates > Business Intelligence**, click **Report Server Project**.

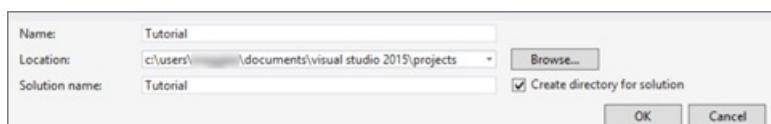


4. Click **Report Server Project** .

Note: If you don't see the **Business Intelligence** or **Report Server Project** options, you need to update SSDT with the Business Intelligence templates. See [Download SQL Server Data Tools \(SSDT\)](#)

5. In **Name**, type **Tutorial**.

By default, it's created in your Visual Studio 2015\Projects folder in a new directory.



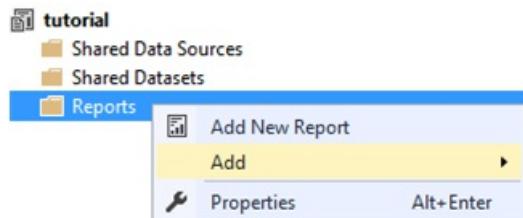
6. Click **OK** to create the project.

The Tutorial project is displayed in the Solution Explorer pane on the right.

To create a new report definition file

1. In the **Solution Explorer** pane, right-click the **Reports > Add > New Item**.

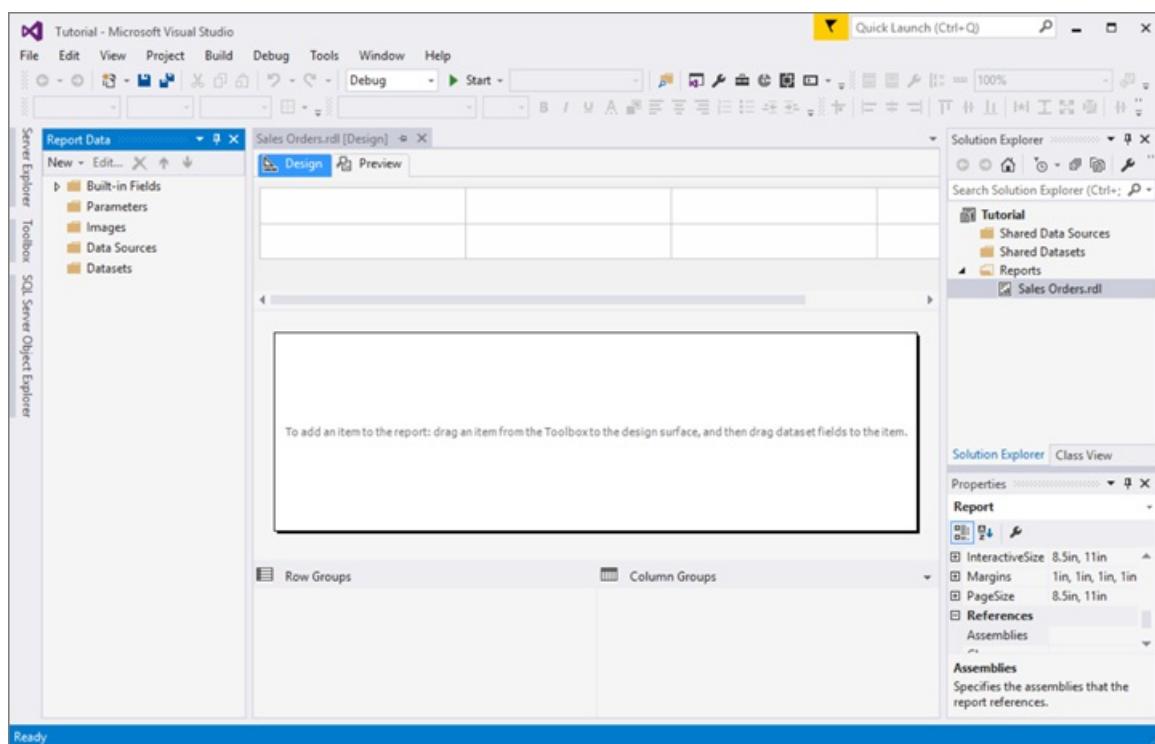
Tip: If you don't see the **Solution Explorer** pane, on the **View** menu, click **Solution Explorer**.



2. In the **Add New Item** window, click **Report**.

3. In **Name**, type **Sales Orders.rdl** and then click **Add**.

Report Designer opens and displays the new .rdl file in Design view.



Report Designer is a Reporting Services component that runs in SQL Server Data Tools (SSDT). It has two views: **Design** and **Preview**. Click each tab to change views.

You define your data in the **Report Data** pane. You define your report layout in **Design** view. You can run the report and see what it looks like in **Preview** view.

Next lesson

You have successfully created a report project called "Tutorial" and added a report definition (.rdl) file to the report project. Next, you will specify a data source to use for the report. See [Lesson 2: Specifying Connection Information \(Reporting Services\)](#).

See Also

[Create a Basic Table Report \(SSRS Tutorial\)](#)

Lesson 2: Specifying Connection Information (Reporting Services)

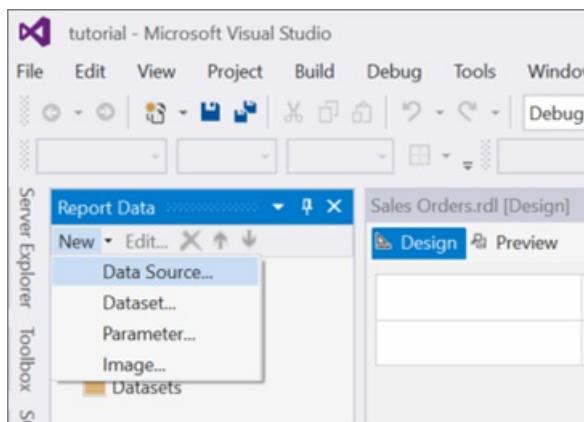
10/1/2018 • 2 minutes to read • [Edit Online](#)

After you add a Reporting Services paginated report to your Tutorial project in Lesson 1, now you need to define a *data source*, which is the connection information the report uses to access data from either a relational database, multidimensional database, or another source.

In this lesson, you use the **AdventureWorks2014** sample database as your data source. This tutorial assumes that this database is located in a default instance of SQL Server Database Engine installed on your local computer.

To set up a connection

1. In the **Report Data** pane, click **New** and then click **Data Source**.
If the **Report Data** pane is not visible, from the **View** menu, click **Report Data**.



2. In **Name**, type *AdventureWorks2014*.
3. Make sure **Embedded connection** is selected.
4. In **Type**, select **Microsoft SQL Server**.
5. In **Connection string**, type the following:

```
Data source=localhost; initial catalog=AdventureWorks2014
```

This connection string assumes that SQL Server Data Tools (SSDT), the report server, and the **AdventureWorks2014** database are all installed on the local computer and that you have permission to log on to the **AdventureWorks2014** database. If your AdventureWorks2014 database is not on the local computer, change the connection string and replace *localhost* with the name of your database server instance.

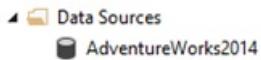
NOTE

If you are using SQL Server Express with Advanced Services or a named instance, the connection string must include instance information:

```
Data source=localhost\SQLEXPRESS; initial catalog=AdventureWorks2014
```

For more information about connection strings, see: [Data Connections, Data Sources, and Connection Strings in Reporting Services](#).

6. Click **Credentials** in the left pane and click **Use Windows Authentication (integrated security)**.
7. Click **OK**. The data source **AdventureWorks2014** is added to the **Report Data** pane.



Next Task

You have successfully defined a connection to the **AdventureWorks2014** sample database. Next, you will create the report. See [Lesson 3: Defining a Dataset for the Table Report \(Reporting Services\)](#).

See Also

[Data Connections, Data Sources, and Connection Strings in Reporting Services](#)

Lesson 3: Defining a Dataset for the Table Report (Reporting Services)

11/28/2018 • 2 minutes to read • [Edit Online](#)

After you define the data source, you need to define a dataset. In Reporting Services, data that you use in reports is contained in a *dataset*. A dataset includes a pointer to a data source and a query to be used by the report, as well as calculated fields and variables.

Use the query designer in Report Designer to design the dataset. For this tutorial, you will create a query that retrieves sales order information from the **AdventureWorks2014** database.

To define a Transact-SQL query for report data

1. In the **Report Data** pane, click **New**, and then click **Dataset...**. The **Dataset Properties** dialog box opens.
2. In the **Name** box, type **AdventureWorksDataset**.
3. Click **Use a dataset embedded in my report**.
4. Select the data source you created in the previous lesson, **AdventureWorks2014**.
5. Select **Text** for the **Query type**.
6. Type, or copy and paste, the following Transact-SQL query into the **Query** box.

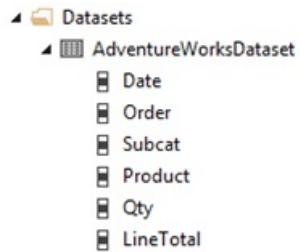
```
SELECT
    soh.OrderDate AS [Date],
    soh.SalesOrderNumber AS [Order],
    pps.Name AS Subcat, pp.Name as Product,
    SUM(sd.OrderQty) AS Qty,
    SUM(sd.LineTotal) AS LineTotal
FROM Sales.SalesPerson sp
    INNER JOIN Sales.SalesOrderHeader AS soh
        ON sp.BusinessEntityID = soh.SalesPersonID
    INNER JOIN Sales.SalesOrderDetail AS sd
        ON sd.SalesOrderID = soh.SalesOrderID
    INNER JOIN Production.Product AS pp
        ON sd.ProductID = pp.ProductID
    INNER JOIN Production.ProductSubcategory AS pps
        ON pp.ProductSubcategoryID = pps.ProductSubcategoryID
    INNER JOIN Production.ProductCategory AS ppc
        ON ppc.ProductCategoryID = pps.ProductCategoryID
GROUP BY ppc.Name, soh.OrderDate, soh.SalesOrderNumber, pps.Name, pp.Name,
    soh.SalesPersonID
HAVING ppc.Name = 'Clothing'
```

7. (Optional) Click the **Query Designer** button. The query is displayed in the text-based query designer. You can toggle to the graphical query designer by clicking **Edit As Text**. View the results of the query by clicking the run  button on the query designer toolbar.

You see the data from six fields from four different tables in the **AdventureWorks2014** database. The query makes use of Transact-SQL functionality such as aliases. For example, the SalesOrderHeader table is called *soh*.

8. Click **OK** to exit the query designer.
9. Click **OK** to exit the **Dataset Properties** dialog box.

Your **AdventureWorksDataset** dataset and fields appear in the Report Data pane.



Next Task

You have successfully specified a query that retrieves data for your report. Next, you will create the report layout. See [Lesson 4: Adding a Table to the Report \(Reporting Services\)](#).

See Also

[Query Design Tools \(SSRS\)](#)

[SQL Server Connection Type \(SSRS\)](#)

[Tutorial: Writing Transact-SQL Statements](#)

Lesson 4: Adding a Table to the Report (Reporting Services)

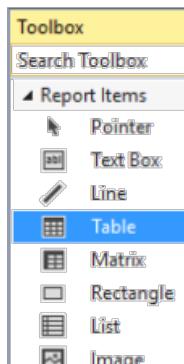
10/1/2018 • 3 minutes to read • [Edit Online](#)

After the dataset is defined, you can start designing the report. You create a report layout by dragging and dropping data regions, text boxes, images, and other items that you want to include in your report to the design surface.

Items that contain repeated rows of data from underlying datasets are called *data regions*. A basic report will have only one data region, but you can add more, for example, if you want to add a chart to your tabular report. After you add a data region, you can add fields to the data region.

To add a Table data region and fields to a report layout

1. In the **Toolbox**, click **Table**, and then click on the design surface and drag the mouse. Report Designer draws a table data region with three columns in the center of the design surface. The **Toolbox** may appear as a tab on the left side of the **Report Data** pane. To open the **Toolbox**, move the pointer over the **Toolbox** tab. If the **Toolbox** is not visible, from the **View** menu, click **Toolbox**.

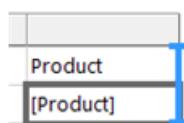


You can also add a table to the report from the design surface. Right-click the design surface, click **Insert** and then click **Table**. 2. In the **Report Data** pane, expand the dataset **AdventureWorksDataset** to display the fields.

3. Drag the *Date* field from the **Report Data** pane to the first column in the table.

When you drop the field into the first column, two things happen. First, the data cell will display the field name, known as the *field expression*, in brackets: `[Date]`. Second, a column header value is automatically added to Header row, just above the field expression. By default, the column is the name of the field. You can select the Header row text and type a new name.

4. Drag the *Order* field from the **Report Data** pane to the second column in the table.
5. Drag the *Product* field from the **Report Data** pane to the third column in the table.
6. Drag the *Qty* field to the right edge of the third column until you get a vertical cursor and the mouse pointer has a plus sign [+]. When you release the mouse button, a fourth column is created for `[Qty]`.



7. Add the *LineTotal* field in the same way, creating a fifth column. The column header is *Line Total*. Report Designer automatically creates a friendly name for the column by splitting *LineTotal* into two words.

The following diagram shows a table data region that has been populated with these fields: Date, Order, Product, Qty, and Line Total.

Date	Order	Product	Qty	Line Total
[Date]	[Order]	[Product]	[Qty]	[LineTotal]

Preview Your Report

Previewing a report enables you to view the rendered report without having to first publish it to a report server. You will probably want to preview your report frequently during design time. Previewing the report will also run validation on the design and data connections so you can correct errors and issues before publishing the report to a report server.

To preview a report

- Click the **Preview** tab. Report Designer runs the report and displays it in Preview view.



The following diagram shows part of the report in Preview view.

Date	Order	Product	Qty	Line Total
8/31/2011 12:00:00 AM	SO44285	Mountain Bike Socks, M	12	64.797600
10/31/2011 12:00:00 AM	SO44792	Mountain Bike Socks, M	12	64.797600

Notice that the currency (in the Line Total column) has six places after the decimal, and the date has includes a time stamp. You're going to fix that formatting in the next lesson.

NOTE

On the **File** menu, click **Save All** to save the report.

Next Steps

You have successfully added a Table data region to your report, added fields to the data region, and previewed your report. Next, you will format column headers and date and currency values. See [Lesson 5: Formatting a Report \(Reporting Services\)](#).

See Also

[Tables \(Report Builder and SSRS\)](#)

[Dataset Fields Collection \(Report Builder and SSRS\)](#)

Lesson 5: Formatting a Report (Reporting Services)

10/1/2018 • 2 minutes to read • [Edit Online](#)

Now that you've added a data region and some fields to the Sales Orders report, you can format the date and currency fields and the column headers.

Format the Date

The Date field displays date and time information by default. You can format it to display only the date.

To format a date field

1. Click the **Design** tab.
2. Right-click the cell with the `[Date]` field expression and then click **Text Box Properties**.
3. Click **Number**, and then in the **Category** field, click **Date**.
4. In the **Type** box, select **January 31, 2000**.
5. Click **OK**.
6. Preview the report to see the change to the `[Date]` field and then change back to design view.

Format the Currency

The **LineTotal** field displays a general number. Format it to display the number as currency.

To format a currency field

1. Right-click the cell with the `[LineTotal]` field expression and then click **Text Box Properties**.
2. Click **Number**, and in the **Category** field, click **Currency**.
3. If your regional setting is English (United States), the defaults should be:
 - **Decimal places: 2**
 - **Negative numbers: (\$12345.00)**
 - **Symbol: \$ English (United States)**
4. Select **Use 1000 separator (,)**.

If the sample text is:\$12,345.00, then your settings are correct.

5. Click **OK**.
6. Preview the report to see the change to the `[LineTotal]` field and then change back to design view.

Change Text Style and Column Widths

You can also change the formatting of the header row to differentiate it from the rows of data in the report. Lastly, you will adjust the widths of the columns.

To format header rows and table columns

1. Click the table so that column and row handles appear above and next to the table. The gray bars along the top and side of the table are the column and row handles.

2. Point to the line between column handles so that the cursor changes into a double arrow. Drag the columns

	Date	Order	Product	Qty	Line Total
	[Date]	[Order]	[Product]	[Qty]	[LineTotal]

to the size you want.

3. Select the row containing column header labels and from the **Format** menu, point to **Font** and then click **Bold**.

4. To preview your report, click the **Preview** tab. It should look something like this:

Date	Order	Product	Qty	Line Total
August 31, 2011	SO44285	Mountain Bike Socks, M	12	\$64.80
October 31, 2011	SO44792	Mountain Bike Socks, M	12	\$64.80
May 30, 2012	SO46604	Men's Bib-Shorts, S	2	\$107.99
June 30, 2012	SO47004	Men's Bib-Shorts, S	5	\$269.97

5. On the **File** menu, click **Save All** to save the report.

Next Steps

You have successfully formatted column headers and date and currency values. Next, you will add grouping and totals to your report. See [Lesson 6: Adding Grouping and Totals \(Reporting Services\)](#).

See Also

[Formatting Numbers and Dates \(Report Builder and SSRS\)](#)

[Rendering Behaviors \(Report Builder and SSRS\)](#)

Lesson 6: Adding Grouping and Totals (Reporting Services)

11/15/2018 • 4 minutes to read • [Edit Online](#)

In this tutorial lesson, you will add grouping and totals to your Reporting Services report to organize and summarize your data.

To group data in a report

1. Click the **Design** tab.
2. If you do not see the **Row Groups** pane, right-click the design surface and click **View** and then click **Grouping**.
3. From the **Report Data** pane, drag the **Date** field to the **Row Groups** pane. Place it above the row called **(Details)**.

Note that the row handle now has a bracket in it, to show a group. The table now also has two Date columns -- one on either side of a vertical dotted line.

The screenshot shows the 'Row Groups' pane in the Reporting Services designer. A row handle for the 'Date' field is selected, highlighted with a yellow border and a bracket symbol. The 'Row Groups' pane lists a single group named 'Date' above the '(Details)' group.

4. From the **Report Data** pane, drag the **Order** field to the **Row Groups** pane. Place it below Date and above **(Details)**.

The screenshot shows the 'Row Groups' pane in the Reporting Services designer. The 'Order' field has been added below the 'Date' group and above the '(Details)' group, indicated by a blue highlight and a double bracket symbol in the row handle.

Note that the row handle now has two brackets in it , to show two groups. The table now has two **Order** columns, too.

5. Delete the original **Date** and **Order** columns to the **right** of the double line. This removes this individual record values so that only the group value is displayed. Select the column handles for the two columns, right-click and click **Delete Columns**.

The screenshot shows a table with four columns. The first two columns are labeled 'Date' and 'Order'. The third and fourth columns are also labeled 'Date' and 'Order'. The second and third columns are merged together, creating a gap in the data structure.

6. To format the new date column, Right-click the cell with the **[Date]** field expression and then click **Text Box Properties**.
7. Click **Number**, and then in the **Category** field, click **Date**.
8. In the **Type** box, select **January 31, 2000**.

9. Click **OK..**

10. Switch to the **Preview** tab to preview the report. It should look similar to the following illustration:

Date	Order	Product	Qty	Line Total
May 31, 2011	SO43659	Long-Sleeve Logo Jersey, M	3	\$86.52
		Long-Sleeve Logo Jersey, XL	1	\$28.84
		Mountain Bike Socks, M	6	\$34.20
		AWC Logo Cap	2	\$10.37
	SO43661	Long-Sleeve Logo Jersey, L	4	\$115.36
		Long-Sleeve Logo Jersey, XL	2	\$57.68
		AWC Logo Cap	4	\$20.75
	SO43664	Long-Sleeve Logo Jersey, XL	1	\$28.84
		Long-Sleeve Logo Jersey, M	1	\$28.84
SO43665		Long-Sleeve Logo Jersey, L	2	\$57.68

To add totals to a report

1. Switch to Design view.

2. Right-click the data region cell that contains the field `[LineTotal]`, and click **Add Total**.

This adds a row with a sum of the dollar amount for each order.

3. Right-click the cell that contains the field `[Qty]`, and click **Add Total**.

This adds a sum of the quantity for each order to the totals row.

4. In the empty cell to the left of `Sum[Qty]`, type the label "**Order Total**".

5. You can add a background color to the totals row. Select the two sum cells and the label cell.

6. On the **Format** menu, click **Background Color**, click **Light Gray**, and click **OK**.

Date	Order	Product	Qty	Line Total
[Date]	[Order]	[Product]	[Qty]	[LineTotal]
Order Total		[Sum(Qt [Sum(LineTotal)])]		

To add a daily total to a report

1. Right-click the **Order** cell, point to **Add Total**, and click **After**.

This adds a new row containing sums of the quantity and dollar amount for each day, and the label "**Total**" to the bottom of the Order column.

2. Type the word **Daily** before the word **Total** in the same cell, so it reads **Daily Total**.

3. Select the **Daily Total** cell, the two **Sum** cells and the empty cell between them.

4. On the **Format** menu, click **Background Color**, click **Orange**, and click **OK**.

Date	Order	Product	Qty	Line Total
[Date]	[Order]	[Product]	[Qty]	[LineTotal]
Daily Total		[Sum(Qt [Sum(LineTotal)])]		

To add a grand total to a report

1. Right-click the Date cell, point to **Add Total**, and click **After**.

This adds a new row containing sums of the quantity and dollar amount for the entire report, and the **Total** label in the **Date** column.

2. Type the word **Grand** before the word **Total** in the same cell, so it reads **Grand Total**.
3. Select the **Grand Total** cell, the two **Sum** cells and the empty cells between them.
4. On the **Format** menu, click **Background Color**, click **Light Blue**, and click **OK**.

	Date	Order	Product	Qty	Line Total
	[Date]	[Order]	[Product]	[Qty]	[Line Total]
			Order Total	[Sum(Qt	[Sum(LineTotal)
		Daily Total		[Sum(Qt	[Sum(LineTotal)
	Grand Total			[Sum(Qt	[Sum(LineTotal)]

5. Click **Preview**.

The last page should look similar to the following image. In the toolbar, click the Last Page  button.

S071950	Women's Mountain Shorts, L	6	\$251.96
	Women's Mountain Shorts, S	3	\$125.98
	Order Total	9	\$377.95
S071951	Women's Mountain Shorts, L	3	\$125.98
	Order Total	3	\$125.98
S071952	Women's Mountain Shorts, L	15	\$548.55
	Women's Mountain Shorts, M	3	\$125.98
	Women's Mountain Shorts, S	3	\$125.98
	Order Total	21	\$800.51
Daily Total		2999	\$83,642.49
Grand Total		84589	\$1,780,769.91

To Publish the Report to the Report Server (Optional)

1. An optional step is to publish the completed report to the native mode report server so you can view the report in the web portal.
2. Click the **Project** menu and then click **tutorial Properties...**
3. In the **TargetServerURL** type the name of your report server, for example
 - `http://<servername>/reportserver`
 - `https://localhost/reportserver` works if your designing the report on the report server.
4. Note the TargetReportFolder is `tutorial`, the name of the project. This is the name of the folder that the report will deploy to in the next steps.
5. Click **OK**

6. On click the **Build** menu and then click **Deploy tutorial**.

If you see a message similar to the following in the output window, it indicates a successful deployment.

```
----- Build started: Project: tutorial, Configuration: Debug -----
Skipping 'Sales Orders.rdl'. Item is up to date.
Build complete -- 0 errors, 0 warnings
----- Deploy started: Project: tutorial, Configuration: Debug -----
Deploying to https://[server name]/reportserver
Deploying report '/tutorial/Sales Orders'.
Deploy complete -- 0 errors, 0 warnings
===== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped =====
===== Deploy: 1 succeeded, 0 failed, 0 skipped =====
```

If you see an error message similar to the following, verify you have permissions on the report server and you have started SQL Server Data Tools with administrator privileges.

"The permissions granted to user 'XXXXXXXX\[your user name]' are insufficient for performing this operation"

7. Browse to the web portal with administrator privileges, for example, right-click the icon for Internet Explorer and click **Run as administrator**.

Browse to Reporting Services web portal URL.

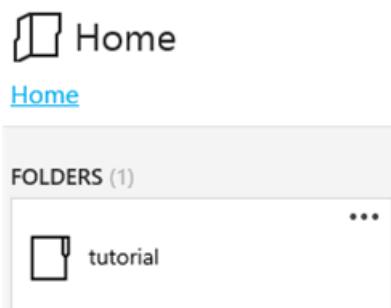
Note: The *portal URL* is "Reports", not the Report Server URL of "Reportserver". For example:

`https://<server name>/reports`.

`https://localhost/reports` works if you're designing the report on the report server.

8. Browse to the folder that contains the report. The default name is *tutorial*, the name of the project or the name you typed into the TargetReportFolder field in the project properties.

Click the name of the report **Sales Orders** to view the rendered report in the browser.



You have successfully completed the Creating a Basic Table Report tutorial.

See Also

[Filter, Group, and Sort Data \(Report Builder and SSRS\)](#)

Create a Data-Driven Subscription (SSRS Tutorial)

12/3/2018 • 3 minutes to read • [Edit Online](#)

This Reporting Services tutorial teaches you the concepts of data-driven subscriptions by walking you through a simple example that creates a data-driven subscription to generate and save filtered report output to a file share. Reporting Services data-driven subscriptions allow you to customize and automate the distribution of a report based on dynamic subscriber data. Data-driven subscriptions are intended for the following kinds of scenarios:

- Distributing reports to a large recipient pool whose membership may change from one distribution to the next. For example, email a monthly report to all current customers.
- Distributing reports to a specific group of recipients based on predefined criteria. For example, send a sales performance report to all of the sales managers in an organization.
- Automate the generation of reports in a wide variety of formats, for example .xlsx and .pdf.

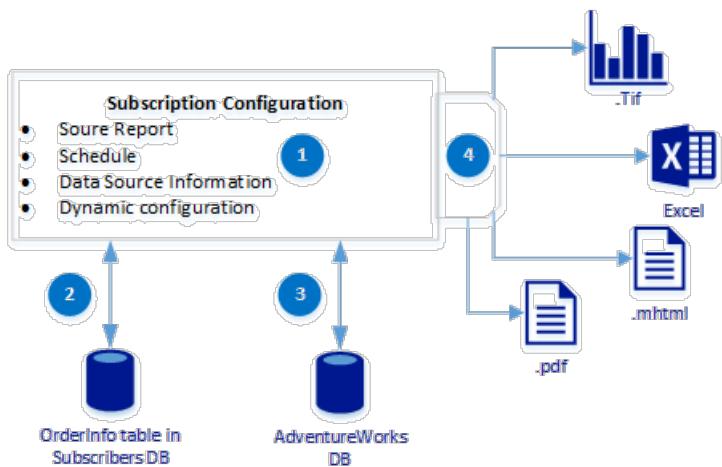
What You Will Learn

This tutorial is divided into three lessons:

[Lesson 1: Create a Sample Subscriber Database](#) | In this lesson you will create a table local SQL Server database that contains subscriber information. the information Order Numbers to use for filtering and output file formats. [Lesson 2: Configure Report Data Source Properties](#) |In this lesson, you will configure a report data source so that the report can run unattended on a schedule. Unattended processing requires stored credentials. You will also modify the report dataset to include a parameter that is supplied by the subscriber data. This parameter is used to filter the report data based on order number. [Lesson 3: Define a Data-Driven Subscription](#) | In this lesson you will create a data-driven subscription. This lesson guides you through each page in the Data-Driven Subscription Wizard.

The Following diagram illustrates the basic workflow of the tutorial

STEP	DESCRIPTION
(1)	The subscription configuration notes the source report, schedule, and the field mapping to the subscribers Database.
(2)	The OrderInfo table contains 4 order numbers to use for filtering, 1 per file. The table also contains the file formats for the generated reports.
(3)	Information from the Adventureworks database is filtered and return in the report.
(4)	The reports are created in the file formats specified in the Orderinfo table.



Requirements

Data-driven subscriptions are typically created and maintained by report server administrators. The steps to create data-driven subscriptions require building queries, knowledge of data sources that contain subscriber data, and elevated permissions on a report server.

The tutorial uses the *Sales order* report created in the tutorial [Create a Basic Table Report \(SSRS Tutorial\)](#) and data from the sample database **AdventureWorks2014**.

Your computer must have the following installed to use this tutorial:

- An edition of SQL Server that supports data-driven subscriptions. For more information, see [Editions and Features of SQL Server 2017](#).
- The report server must be running in native mode. The user interface described in this tutorial is based on a native mode report server. Subscriptions are supported on SharePoint mode report servers but the user interface will be different than what is described in this tutorial.
- SQL Server Agent service must be running.
- A report that includes parameters. This tutorial assumes the sample report, **Sales Orders** you create using the tutorial [Create a Basic Table Report \(SSRS Tutorial\)](#).
- The **AdventureWorks2014** sample database, which provides data to the sample report.
- A Reporting Services role assignment that includes the Manage all subscriptions task on the sample report. This task is required for defining a data-driven subscription. If you are an administrator on the computer, the default role assignment for local administrators provides the permissions necessary for creating data-driven subscriptions. For more information, see [Granting Permissions on a Native Mode Report Server](#).
- A shared folder for which you have write permissions. The shared folder must be accessible over a network connection.

Estimated time to complete the tutorial: 30 minutes. An additional 30 minutes if you have not completed the basic report tutorial.

See Also

[Data-Driven Subscriptions](#)

[Create a Basic Table Report \(SSRS Tutorial\)](#)

Lesson 1: Creating a Sample Subscriber Database

10/24/2018 • 2 minutes to read • [Edit Online](#)

In this Reporting Services tutorial lesson, you create a small "subscriber" database to store subscription data that will be used by a data-driven subscription. When the subscription is processed, the report server retrieves this data and uses it to customize report output. For example, the rows of data include specific order numbers to use for filters and what file format generated reports will be in when they are created.

This lesson assumes you are using SQL Server Management Studio to create a SQL Server database.

To create a sample Subscriber database

1. Start Management Studio, and open a connection to an instance of the SQL Server Database Engine.
2. Right-click on Databases, select **New Database....**
3. In the New Database dialog box, in **Database Name**, type *Subscribers*.
4. Click **OK**.
5. Click the **New Query** button on the toolbar.
6. Copy the following Transact-SQL statements into the empty query:

```
Use Subscribers
CREATE TABLE [dbo].[OrderInfo] (
    [SubscriptionID] [int] NOT NULL PRIMARY KEY ,
    [Order] [nvarchar] (20) NOT NULL,
    [FileType] [bit],
    [Format] [nvarchar] (20) NOT NULL ,
) ON [PRIMARY]
GO

INSERT INTO [dbo].[OrderInfo] (SubscriptionID, [Order], FileType, Format)
VALUES ('1', 'so43659', '1', 'IMAGE')
INSERT INTO [dbo].[OrderInfo] (SubscriptionID, [Order], FileType, Format)
VALUES ('2', 'so43664', '1', 'MHTML')
INSERT INTO [dbo].[OrderInfo] (SubscriptionID, [Order], FileType, Format)
VALUES ('3', 'so43668', '1', 'PDF')
INSERT INTO [dbo].[OrderInfo] (SubscriptionID, [Order], FileType, Format)
VALUES ('4', 'so71949', '1', 'Excel')
GO
```

7. Click **! Execute** on the toolbar.
8. Use a SELECT statement to verify that you have three rows of data. For example: `select * from OrderInfo`

Next Steps

- You have successfully created the subscription data that will drive report distribution and vary the report output for each subscriber.
- Next, you will modify the data source properties of the report to use stored credentials.
- You will also modify the report design to include a parameter that the subscription will use with the subscriber data. [Lesson 2: Modifying the Report Data Source Properties](#).

Next steps

[Create a Data-Driven Subscription](#)

[Create a Database](#)

[Create a Basic Table Report](#)

More questions? [Try asking the Reporting Services forum](#)

Lesson 2: Modifying the Report Data Source Properties

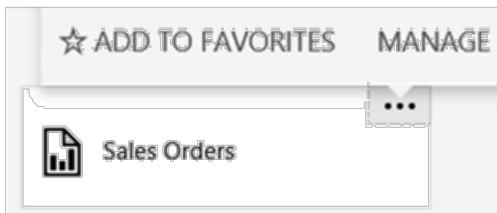
11/27/2018 • 3 minutes to read • [Edit Online](#)

In this Reporting Services tutorial lesson, you use web portal to select a report that will be delivered to recipients. The data-driven subscription that you will define will distribute the **Sales Order** report created in the tutorial [Create a Basic Table Report \(SSRS Tutorial\)](#). In the steps that follow, you will modify the data source connection information used by the report to get data. Only reports that use **stored credentials** to access a report data source can be distributed through a data-driven subscription. Stored credentials are necessary for unattended report processing.

You will also modify the dataset and report to use a parameter to filter the report on the [Order] so the subscription can output different instances of the report for specific orders and rendering formats.

To Modify the Data Source to use stored credentials

1. Browse to the Reporting Services web portal with administrator privileges, for example, right-click the icon for Internet Explorer and click **Run as administrator**.
2. Browse to the web portal URL. For example:
`https://<server name>/reports`.
`https://localhost/reports` **Note:** The web portal URL is "Reports", not the Report Server URL of "Reportserver".
3. Browse to the folder containing the **Sales Orders** report and in the context menu of the report, click **Manage**.



3. Click **Data Sources** in the left pane.
4. Verify the **Connection Type** is **Microsoft SQL Server**.
5. Verify the connection string is the following and it assumes that the sample database is on a local database server:

```
Data source=localhost; initial catalog=AdventureWorks2014
```

6. Click **Use the following credentials**.
7. In the **Type of credentials**, select **Windows user name and password**
8. Type your user name (use the format *domain\user*) and password. If you do not have permission to access the AdventureWorks2014 database, specify a login that does.
9. Click **Test Connection** to verify you can connect to the data source.

10. Click **Save**.

11. Click **Cancel**

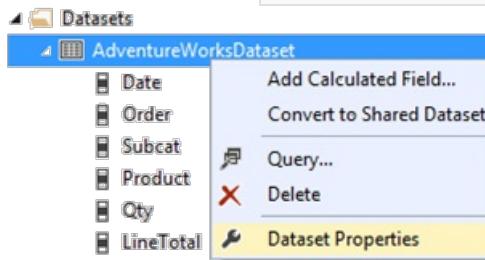
12. View the report to verify that the report runs with the credentials you specified. .

To Modify the AdventureWorksDataset

In the following steps you will modify the dataset to use a parameter to filter the data set based an order number.

1. Open the **Sales Orders** report in SQL Server Data Tools

2. Right-click the dataset **AdventureWorksDataset** and click **Dataset Properties**.



3. Add the statement `WHERE (UPPER(SalesOrderNumber) =UPPER(@OrderNumber) or @OrderNumber IS NULL)` before the **Group By** statement. The full query syntax is the following:

```
SELECT soh.OrderDate AS Date, soh.SalesOrderNumber AS [Order], pps.Name AS Subcat, pp.Name AS Product,
SUM(sd.OrderQty) AS Qty, SUM(sd.LineTotal) AS LineTotal
FROM Sales.SalesPerson AS sp INNER JOIN
Sales.SalesOrderHeader AS soh ON sp.BusinessEntityID = soh.SalesPersonID INNER JOIN
Sales.SalesOrderDetail AS sd ON sd.SalesOrderID = soh.SalesOrderID INNER JOIN
Production.Product AS pp ON sd.ProductID = pp.ProductID
INNER JOIN
Production.ProductSubcategory AS pps ON pp.ProductSubcategoryID = pps.ProductSubcategoryID
INNER JOIN
Production.ProductCategory AS ppc ON ppc.ProductCategoryID = pps.ProductCategoryID
WHERE (UPPER(SalesOrderNumber) =UPPER(@OrderNumber) or @OrderNumber IS NULL)

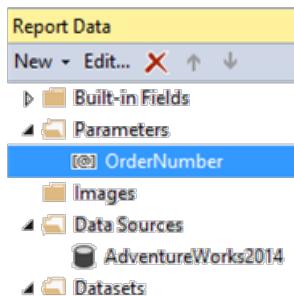
GROUP BY ppc.Name, soh.OrderDate, soh.SalesOrderNumber, pps.Name, pp.Name, soh.SalesPersonID
HAVING (ppc.Name = 'Clothing')
```

4. Click **OK**

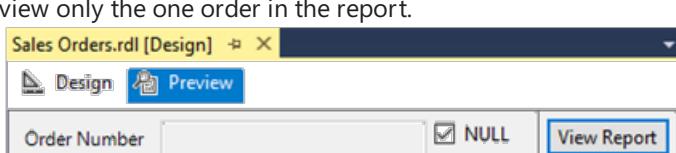
In the following steps you will add a parameter to the report. The report parameter feeds the dataset parameter.

To Add a Report Parameter and Republish the Report

1. In the **Report Data** pane expand the parameters folder and double-click the **OrderNumber** parameter. It was created automatically as part of the previous steps when you added the parameter to the dataset. click **New** and then click **Parameter...**



2. Verify the **Name** is `OrderNumber`.
 3. Verify the **Prompt** is `OrderNumber`.
 4. Select **Allow blank value ("")**.
 5. Select **Allow null value**.
 6. Click **OK**.
7. Click the **Preview** tab to run the report. Note the parameter input box at the top of the report. You can either:
- Click View Report to see the full report without using a parameter.
 - Unselect the **Null** option and type an order number, for example `so71949`, then click **View Report** to view only the one order in the report.



Re-deploy the Report

1. Re-deploy the report so the subscription configuration in the next lesson can utilize the changes you made in this lesson. For more information on the project properties used in the table tutorial, see section 'To Publish the Report to the Report Server (Optional)' of [Lesson 6: Adding Grouping and Totals \(Reporting Services\)](#).
2. On the toolbar click **Build** and then click **Deploy tutorial**.

Next Steps

- You successfully configured the report to get data using stored credentials and the data can be filtered with a parameter.
- In the next lesson, you configure the subscription using the web portal Data-Driven Subscription pages. See [Lesson 3: Defining a Data-Driven Subscription](#).

See Also

[Manage Report Data Sources](#)

[Specify Credential and Connection Information for Report Data Sources](#)

[Create a Data-Driven Subscription \(SSRS Tutorial\)](#)

[Create a Basic Table Report \(SSRS Tutorial\)](#)

Lesson 3: Defining a Data-Driven Subscription

11/26/2018 • 2 minutes to read • [Edit Online](#)

In this Reporting Services tutorial lesson, you use the Reporting Services web portals data-driven subscription pages to connect to a subscription data source, build a query that retrieves subscription data, and map the result set to report and delivery options.

NOTE

Before you start, verify that **SQL Server Agent** service is running. If it is not running, you cannot save the subscription. One method for verification is to open the [SQL Server Configuration Manager](#). This lesson assumes you completed Lesson 1 and Lesson 2 and that the report data source uses stored credentials. For more information, see [Lesson 2: Modifying the Report Data Source Properties](#)

Start the Data-Driven Subscription Wizard

1. In Reporting Services web portal, click **Home**, and navigate to the folder containing the **Sales Orders** report.
2. In the context menu  of the report, click **Manage**, and then click **Subscriptions** in the left pane.
3. Click **+ New Subscription**. If you do not see this button, you do not have Content Manager permissions.

Define a description

1. Type **Sales Order delivery** in description.

Type

1. click **Data-driven subscription..**

Schedule

1. In the schedule section click **Report-specific schedule**.
2. Click **Edit schedule**.
3. In **Schedule Details**, click **Once**.
4. Specify a start time that is a few minutes ahead of the current time.
5. Click **Apply**.

Destination

1. In the Destination section, Select **Windows File Share** for the method of delivery.

Dataset

1. click **Edit Dataset**.
2. Select **A custom data source**.
3. Select **Microsoft SQL Server** as the data source **Connection** type.

4. In Connection string, type the following connection string. *Subscribers* is the database you created in lesson 1.

```
data source=localhost; initial catalog=Subscribers
```

Credentials

1. Select **Using the following credentials**.
2. Select **Windows user name and password**.
3. In **User Name** and **Password**, type your domain user name and password. Include both the domain and user account when specifying **User Name**.

NOTE

Credentials used to connect to a subscriber data source are not passed back to Management Studio. If you modify the subscription later, you must retype the password used to connect to the data source.

Query

1. In the query box, type the following query:

```
Select * from OrderInfo
```

2. Specify a time-out of 30 seconds.
3. Click **Validate query**, and then click **Apply**.

Delivery Options

Fill in the following values:

PARAMETER	SOURCE OF VALUE	VALUE/FIELD
File name	Get value from dataset	Order
Path	Enter value	In the Value, type the name of a public file share for which you have write permissions (for example, <code>\mycomputer\public\myreports</code>).
Render Format	Get value from dataset	Format
Enter Value	Autoincrement	
File Extension	Enter Value	True
User Name	Enter Value	Type your domain user account. Enter it in this format: <domain>\<account>. The user account needs to have permissions to the path you configured.
Password	Enter Value	Type your password

Report parameters

1. In the **OrderNumber** field , select **Get value from dataset**. In Value, select **Order**.
2. Click **Create Subscription**.

Next Steps

When the subscription runs, four report files will be delivered to the file share you specified, one for each order in the *Subscribers* data source. Each delivery should be unique in terms of data (the data should be order-specific), rendering format, and file format. You can open each report from the shared folder to verify that each version is customized based on the subscription options you defined.



The subscription page in the Web portal will contain the **Last Run** date and **Status** of the subscription. **Note:** Refresh the page after the subscription runs to see the updated information.

Report	Description	Status	Type	Folder	Delivery	Last run	Result
Sales Orders	Sales Order delivery	Enabled	Data-driven	/tutorial	Windows File Share	May 26, 2016 12:13:02 AM	Done: 4 processed of 4 total; 0 errors.

This step concludes the tutorial "Define a Data-Driven Subscription".

See Also

- [Subscriptions and Delivery \(Reporting Services\)](#)
- [Data-Driven Subscriptions](#)
- [Create, Modify, and Delete Data-Driven Subscriptions](#)
- [Use an External Data Source for Subscriber Data \(Data-Driven Subscription\)](#)

Create Drillthrough (RDLC) Report with Parameters - ReportViewer

10/24/2018 • 2 minutes to read • [Edit Online](#)

A [drillthrough](#) report is a report that a user opens by clicking a link within another report. Drillthrough reports commonly contain details about an item that is contained in an original summary report. This tutorial will walk you through the following lessons of creating a drillthrough report with parameters and a query, in [local mode reporting](#).

Requirements

To use this walkthrough, you must have access to the **AdventureWorks2014** sample database. For more information about how to get the **AdventureWorks2014** sample database, see [AdventureWorks sample databases](#).

This walkthrough assumes that you are familiar with Transaction-SQL queries and ADO.NET [DataSet](#) and [DataTable](#) objects.

Use Visual Studio 2015, and the ASP.NET Web Application, to create an ASP.NET webpage with a ReportViewer control. The control is configured to view a report that you create. For this walkthrough, you create the application in Microsoft Visual C#.

Tasks

[Lesson 1: Create a New Web Site](#)

[Lesson 2: Define a Data Connection and Data Table for Parent Report](#)

[Lesson 3: Design the Parent Report using the Report Wizard](#)

[Lesson 4: Define a Data Connection and Data Table for Child Report](#)

[Lesson 5: Design the Child Report using the Report Wizard](#)

[Lesson 6: Add a ReportViewer Control to the Application](#)

[Lesson 7: Add Drillthrough Action on Parent Report](#)

[Lesson 8: Create a Data Filter](#)

[Lesson 9: Build and Run the Application](#)

See Also

[Reporting Services Tutorials \(SSRS\)](#)

[Design Reports with Report Designer \(SSRS\)](#)

Lesson 1: Create a New Web Site

11/27/2018 • 2 minutes to read • [Edit Online](#)

In this lesson you'll learn how to create a new website project using the ASP.NET website template for Visual C#.

To create a new website

1. Open Microsoft Visual Studio 2015.
2. On the **File** menu, point to **New**, and select **Web Site**.
3. In the **New Web Site** dialog box, in the **Installed Templates** pane, select **Visual C#** and then choose **ASP.NET Reports Web Site**.
4. In the **Location** box, specify a project directory and select **OK**.

The website project opens and will launch the **Report Wizard**. **Cancel** out of the Data Source and Report Wizard as we will create a custom dataset in the next step.

Next Task

You've successfully created a new website project. Next, you will create a data connection and a data table for the parent report. See [Lesson 2: Define a Data Connection and Data Table for Parent Report](#).

Lesson 2: Define a Data Connection and Data Table for Parent Report

11/27/2018 • 2 minutes to read • [Edit Online](#)

After you create a new website project using the ASP.NET website template for Visual C#, your next step is to create a data connection and a data table for the parent report. In this tutorial the data connection is to the AdventureWorks2014 database.

To define a data connection and Data Table by adding a DataSet (for parent report)

1. On the **Website** menu, select **Add New Item**.
2. In the **Add New Item** dialog box, select **DataSet** and select **Add**. When prompted you should add the item to the **App_Code** folder by selecting **Yes**.

This adds a new XSD file **DataSet1.xsd** to the project and opens the DataSet Designer.

3. From the Toolbox window, drag a **TableAdapter** control to the design surface. This launches the **TableAdapter Configuration Wizard**.
4. On the **Choose Your Data Connection** page, select **New Connection**.
5. If this is the first time you've created a data source in Visual Studio, you will see the **Choose Data Source** page. In the **Data Source** box, select **Microsoft SQL Server**.
6. In the **Add Connection** dialog box, perform the following steps:
 - a. In the **Server name** box, enter the server where the **AdventureWorks2014** database is located.
The default SQL Server Express instance is **(local)\sqlexpress**.
 - b. In the **Log on to the server** section, select the option that provides you access to the data. **Use Windows Authentication** is the default.
 - c. From the **Select or enter a database name** drop-down list, select **AdventureWorks2014**.
 - d. Select **OK**, and then select **Next**.
7. If you selected **Use SQL Server Authentication** in the Step 6 (b), select the option whether to include the sensitive data in the string or set the information in your application code.
8. On the **Save the Connection String to the Application Configuration File** page, type in the name for the connection string or accept the default **AdventureWorks2014ConnectionString**. Select **Next**.
9. On the **Choose a Command Type** page, select **Use SQL Statements**, and then select **Next**.
10. On the **Enter a SQL Statement** page, enter the following Transact-SQL query to retrieve data from the **AdventureWorks2014** database, and then select **Next**.

```
SELECT ProductID, Name, ProductNumber, SafetyStockLevel, ReorderPoint FROM Production.Product Order By ProductID
```

You can also create the query by selecting **Query Builder**, and then verify the query by selecting **Execute**

Query. If the query does not return the expected data, you might be using an earlier version of AdventureWorks. For more information about how to get the **AdventureWorks2014** sample database, see [AdventureWorks sample databases](#).

11. On the **Choose Methods to Generate** page, be sure to uncheck **Create methods to send updates directly to the database (GenerateDBDirectMethods)**, and then select **Finish**.

WARNING

Be sure to uncheck **Create methods to send updates directly to the database (GenerateDBDirectMethods)**

You have now completed configuring the ADO.NET DataTable object as the data source for your report. On the DataSet Designer page in Visual Studio, you should see the DataTable object you added, listing the columns specified in the query. DataSet1 contains the data from the Product table, based on the query.

12. Save the file.
13. To preview the data, select **Preview Data** on the **Data** menu, and then select **Preview**.

Next Task

You have successfully created a data connection and a data table for the parent report. Next, you will design the parent report using the Report Wizard. See [Lesson 3: Design the Parent Report using the Report Wizard](#).

Lesson 3: Design the Parent Report using the Report Wizard

11/27/2018 • 2 minutes to read • [Edit Online](#)

After you create a data connection and a data table for the parent report, your next step is to design the parent report using the Report Wizard in Report Designer. For more information about Report Designer, see [Design Reports with Report Designer \(SSRS\)](#).

To design the parent report using the Report Wizard

1. Make sure that the top-level website is selected in **Solution Explorer**.
2. Right-click on the website and select **Add New Item**.
3. In the **Add New Item** dialog box, select **Report Wizard**, enter a name for the report file, and then select **Add**.

This launches the Report Wizard.

4. On the **Dataset Properties** page, in the **Data source** box, select the **DataSet1** you created in [Lesson 2: Define a Data Connection and Data Table for Parent Report](#).

The **Available datasets** box is automatically updated with the **DataTable** you created above.

5. Select **Next**.
6. In the **Arrange Fields** page do the following:
 - a. Drag **ProductID**, **Name**, **ProductNumber**, **SafetyStockLevel**, and **ReorderLevel** from **Available fields** to the **Values** box.
 - b. Select the arrow next to **Sum(ProductID)**, **Sum(SafetyStockLevel)**, **Sum(ReorderLevel)** and clear the **Sum** selection.
7. Select **Next** twice, then select **Finish** to close the **Report Wizard**.

You've now created the .rdlc file. The file opens in Report Designer. The tablix you designed is now displayed in the design surface.

8. Save the .rdlc file.

Next Task

You have successfully designed the parent report using the Report Wizard. Next, you will create a data connection and a data table for the child report. See [Lesson 4: Define a Data Connection and Data Table for Child Report](#).

Lesson 4: Define a Data Connection and Data Table for Child Report

10/24/2018 • 2 minutes to read • [Edit Online](#)

After you design the parent report, your next step is to create a data connection and a data table for the child report. In this tutorial the data connection is to the AdventureWorks2014 database.

To define a data connection and DataTable by adding a DataSet (for child report)

1. On the **Website** menu, select **Add New Item**.
2. In the **Add New Item** dialog box, select **DataSet** and then select **Add**. When prompted, you should add the item to the **App_Code** folder by selecting **Yes**.

This adds a new XSD file **DataSet2.xsd** to the project and opens the DataSet Designer.

3. From the Toolbox window, drag a **TableAdapter** control to the design surface. This launches the **TableAdapter Configuration Wizard**.
4. On the **Choose Your Data Connection** page, you can select the connection you created in Lesson 2. If you did, select **Next** and go to step 8. Otherwise, select **New Connection**.
5. In the **Add Connection** dialog box, perform the following steps:
 - a. In the **Server name** box, enter the server where the **AdventureWorks2014** database is located.
The default SQL Server Express instance is **(local)\sqlexpress**.
 - b. In the **Log on to the server** section, select the option that provides you access to the data. **Use Windows Authentication** is the default.
 - c. From the **Select or enter a database name** drop-down list, select **AdventureWorks2014**.
 - d. Select **OK**, and then select **Next**.
6. If you selected **Use SQL Server Authentication** in Step 5 (b), select the option whether to include the sensitive data in the string or set the information in your application code.
7. On the **Save the Connection String to the Application Configuration File** page, type in the name for the connection string or accept the default **AdventureWorks2014ConnectionString**. Select **Next**.
8. On the **Choose a Command Type** page, select **Use SQL Statements**, and then select **Next**.
9. On the **Enter a SQL Statement** page, enter the following Transact-SQL query to retrieve data from the **AdventureWorks2014** database, and then select **Next**.

```
SELECT PurchaseOrderID, PurchaseOrderDetailID, OrderQty, ProductID, ReceivedQty, RejectedQty, StockedQty FROM Purchasing.PurchaseOrderDetail
```

You can also create the query by selecting **Query Builder**, and then verify the query by selecting **Execute Query** button. If the query does not return the expected data, you might be using an earlier version of AdventureWorks. For more information about how to get the **AdventureWorks2014** sample database, see [AdventureWorks sample databases](#).

10. On the **Choose Methods to Generate** page, uncheck **Create methods to send updates directly to the**

database (**GenerateDBDirectMethods**), and then select **Finish**.

WARNING

Be sure to uncheck **Create methods to send updates directly to the database (GenerateDBDirectMethods)**

You have now completed configuring the ADO.NET **DataTable** as a data source for your report. On the DataSet Designer page in Visual Studio, you should see the **DataTable** you added, listing the columns specified in the query. DataSet2 contains the data from the PurhcaseOrderDetail table, based on the query.

11. Save the file.
12. To preview the data, select **Preview Data** on the **Data** menu, and then select **Preview**.

Next Task

You have successfully created a data connection and data table for the child report. Next, you will design the child report using the Report Wizard. See [Lesson 5: Design the Child Report using the Report Wizard](#).

Lesson 5: Design the Child Report using the Report Wizard

11/27/2018 • 2 minutes to read • [Edit Online](#)

After you create a data connection and data table for the child report, your next step is to design the child report using the Report Wizard in Report Designer. For more information about Report Designer, see [Design Reports with Report Designer \(SSRS\)](#).

To design the child report using the Report Wizard

1. Make sure that the top-level website is selected in **Solution Explorer**.
2. Right-click on the website and select **Add New Item**.
3. In the **Add New Item** dialog box, click **Report Wizard**, enter a name for the report file, and then select **Add**.

This launches the Report Wizard.

4. In the **Dataset Properties** page, in the **Data source** box, select **DataSet2**.

The **Available datasets** box is automatically updated with the DataTable you created.

5. Select **Next**.
6. In the **Arrange Fields** page do the following:
 - a. Drag **ProductID**, **PurchaseOrderID**, **PurchaseOrderDetailID**, **OrderQty**, **ReceivedQty**, **RejectedQty**, and **StockedQty** from **Available Fields** to the **Values** box.
 - b. Select the arrow next to **Sum(ProductID)**, **Sum(PurchaseOrderID)**, **Sum(PurchaseOrderDetailID)**, **Sum(OrderQty)**, **Sum(ReceivedQty)**, **Sum(RejectedQty)**, and **Sum(StockedQty)** and clear the **Sum** selection.
7. Select **Next** twice, then Select **Finish** to close the **Report Wizard**.

You've now created the .rdlc file. The file opens in Report Designer. The tablix you designed is now displayed in the design surface.

8. With the .rdlc file open, add a parameter by doing the following:
 - a. Right-click **Parameters** in the **Report Data** pane, and then select **Add Parameters**.
 - b. Enter **productid** in the **Name** box.
 - c. Confirm that **Integer** is selected in the **Data Type** list box.
 - d. Click **OK**.
9. Save the .rdlc file.

Next Task

You have successfully designed the child report by using the Report Wizard. Next, you will add a ReportViewer control to the website application. See [Lesson 6: Add a ReportViewer Control to the Application](#).

Lesson 6: Add a ReportViewer Control to the Application

10/24/2018 • 2 minutes to read • [Edit Online](#)

After you design the child report by using the Report Wizard, your next step is to add a ReportViewer control to the website application. If you are using the ASP.NET Reports Web Site, it will have added the ReportViewer control to the default.aspx page.

To add a ReportViewer control to the application

1. In **Solution Explorer**, right-click **Default.aspx**, and then click **View Designer**.
2. If default.aspx already has the ReportViewer Control on it, skip to **Step 4**. Otherwise, From the **AJAX Extensions** group in the **Toolbox** window, drag a **ScriptManager** control to the design surface.
3. From the **Reporting** group, drag a **ReportViewer** control to the design surface below the **ScriptManager** control.
4. Open the **ReportViewer Tasks** window by clicking the arrow in the top right-hand corner of the **ReportViewer** control.
5. In the **Choose Report** box, select the parent report you created.

When you select a report, instances of data sources used in the report are created automatically. Code is generated to instantiate each DataTable (and its **DataSet** container). An **ObjectDataSource** control is added to the design surface, corresponding to each data source used in the report. This data source control is configured automatically.

6. On the Build menu, click Build website.

The report is compiled and any errors such as a syntax error in a report expression appear in the **Error List** area. Click **Error List** at the bottom of the Visual Studio window to display the **Error List** area.

Next Task

You have successfully added a ReportViewer control to the website application. Next, you will add a drillthrough action on the parent report. See [Lesson 7: Add Drillthrough Action on Parent Report](#).

Lesson 7: Add Drillthrough Action on Parent Report

10/1/2018 • 2 minutes to read • [Edit Online](#)

After you add a ReportViewer control to the website application, your next step is to add a drillthrough action on the parent report.

To add drillthrough action on the parent report

1. Go to the parent report.
2. Select the textbox that holds the value of **Name**.
3. Right-click the textbox, and then select **Text Box Properties**.
4. Go to the **Action** tab, and then select the **Go to report** option.
5. Enter the name of the child report in the **Specify a report** section.

NOTE

Do not include the file extension for the report name.

6. Select **Add** under **Use these parameters to run the report** section.
7. Type **productid** in the **name** box, and then select **ProductID** in the **Value** drop-down list.
8. Select **Ok** to finish.

Next Task

You have successfully added a drillthrough action on the parent report. Next, you will create a data filter for the data table that you defined for the child report. See [Lesson 8: Create a Data Filter](#).

Lesson 8: Create a Data Filter

10/1/2018 • 7 minutes to read • [Edit Online](#)

After you add a drillthrough action on the parent report, your next step is to create a data filter for the data table that you defined for the child report.

You can create a table-based filter **or** a query filter for the drillthrough report. This lesson provides instructions for both options.

Table-Based Filter

You need to complete the following tasks to implement a table-based filter.

- Add a filter expression to the tablix in the child report.
- Create a function that selects unfiltered data from the **PurchaseOrderDetail** table.
- Add an event handler that binds the **PurchaseOrderDetail** DataTable to the child report.

To add a filter expression to the tablix in the child report

1. Open the child report.
2. Select a column heading in the tablix, right-click the gray cell that appears above the column heading, and then select **Tablix Properties**.
3. Select on the **Filters** page, and then select **Add**.
4. In the **Expression** field, select **ProductID** from the drop-down list. This is the column to which you apply the filter.
5. Select the equal (=) operator in the **Operator** drop-down list.
6. Select the expression button next to the **Value** field, select **Parameters** in the **Category** area, and then double-click **productid** in the **Values** area. The **Set expression for: Value** field should now contain expression similar to **=Parameters!productid.Value**.
7. Select **OK**, and **OK** again in the **Tablix Properties** dialog box.
8. Save the .rdlc file.

To create a function that selects unfiltered data from the PurchaseOrdeDetail table

1. In Solution Explorer, expand Default.aspx, and then double click Default.aspx.cs.
2. Create a new function that accepts a parameter, **productid**, of type Integer and returns a **datatable** object, and does the following.
 - a. Creates an instance of the dataset, **DataSet2**, which was created in Step 2 of [Lesson 4: Define a Data Connection and Data Table for Child Report](#).
 - b. Create a connection to the SqlServer database to execute the query defined in [Lesson 4: Define a Data Connection and DataTable for Child Report](#).
 - c. The query will return unfiltered data.
 - d. Fill the DataSet instance with the unfiltered data by executing the query.
 - e. Return the **PurchaseOrderDetail** DataTable.

The function will look similar to the one below, (This is just for your reference. You can follow any pattern that you want, to fetch the necessary data for child report).

```
/// <summary>
/// Function to query PurchaseOrderDetail table, fetch the
/// unfiltered data and bind it with the Child report
/// </summary>
/// <returns>A dataTable of type PurchaseOrderDetail</returns>
private DataTable GetPurchaseOrderDetail()
{
    try
    {
        //Create the instance for the typed dataset, DataSet2 which will
        //hold the [PurchaseOrderDetail] table details.
        //The dataset was created as part of the tutorial in Step 4.
        DataSet2 ds = new DataSet2();

        //Create a SQL Connection to the AdventureWorks2008 database using Windows
        Authentication.
        using (SqlConnection sqlconn = new SqlConnection("Data Source=.;Initial
        Catalog=Adventureworks2014;Integrated Security=SSPI"))
        {
            //Building the dynamic query with the parameter ProductID.
            SqlDataAdapter adap = new SqlDataAdapter("SELECT PurchaseOrderID,
            PurchaseOrderDetailID, OrderQty, ProductID, ReceivedQty, RejectedQty, StockedQty FROM
            Purchasing.PurchaseOrderDetail ", sqlconn);
            //Executing the QUERY and fill the dataset with the PurchaseOrderDetail table
            //data.
            adap.Fill(ds, "PurchaseOrderDetail");
        }

        //Return the PurchaseOrderDetail table for the Report Data Source.
        return ds.PurchaseOrderDetail;
    }
    catch
    {
        throw;
    }
}
```

To add an event handler that binds the PurchaseOrderDetail DataTable to the child report

1. Open Default.aspx in designer view.
2. Right-click the ReportViewer control, and then select **Properties**.
3. On the **Properties** page, select the **Events** icon.
4. Double-click the **Drillthrough** event.

This will add an event handler section in the code, which will look similar to the below block.

```
protected void ReportViewer1_Drillthrough(object sender,
Microsoft.Reporting.WebForms.DrillthroughEventArgs e)
{}
```

5. Complete the event handler. It should include the following functionality.

- a. Fetch the child report object reference from the *DrillthroughEventArgs* parameter.
- b. Call the function, **GetPurchaseOrderDetail**

- c. Bind the **PurchaseOrderDetail** DataTable with the report's corresponding data source.

The completed event handler code will look similar to the following.

```
protected void ReportViewer1_Drillthrough(object sender,
Microsoft.Reporting.WebForms.DrillthroughEventArgs e)
{
    try
    {
        //Get the instance of the Target report.
        LocalReport report = (LocalReport)e.Report;

        //Binding the DataTable to the Child report dataset.
        //The name DataSet1 which can be located from,
        //Go to Design view of Child.rdlc, Click View menu -> Report Data
        //You'll see this name under DataSet2.
        report.DataSources.Add(new ReportDataSource("DataSet1", GetPurchaseOrderDetail()));
    }
    catch (Exception ex)
    {
        Response.Write(ex.Message);
    }
}
```

6. Save the file.

Query Filter

You need to complete the following tasks to implement a query filter.

- Create a function that selected filtered data from the **PurchaseOrderDetail** table.
- Add an event handler that retrieves parameter values and binds the **PurchaseOrdeDetail** DataTable to the child report.

To create a function that selects filtered data from the PurchaseOrderDetail table

1. In Solution Explorer, expand Default.aspx, and then double click Default.aspx.cs.
2. Create a new function that accepts a parameter, **productid**, of type Integer and returns a **datatable** object and does the following.
 - a. Creates an instance of the dataset, **DataSet2**, which was created in Step 2 of [Lesson 4: Define a Data Connection and Data Table for Child Report](#).
 - b. Create a connection to the SqlServer database to execute the query defined **Lesson 4: Define a Data Connection and DataTable for Child Report**.
 - c. The query will include a parameter, **productid**, to make sure the data returned is filtered based on the **ProductID** selected in the parent report.
 - d. Fill the DataSet instance with the filtered data by executing the query.
 - e. Return the **PurchaseOrderDetail** DataTable.

The function will look similar to the one below, (This is just for your reference. You can follow any pattern that you want, to fetch the necessary data for child report).

```

/// <summary>
/// Function to query PurchaseOrderDetail table and filter the
/// data for a specific ProductID selected in the Parent report.
/// </summary>
/// <param name="productid">Parameter passed from the Parent report to filter data.</param>
/// <returns>A dataTable of type PurchaseOrderDetail</returns>
private DataTable GetPurchaseOrderDetail(int productid)
{
    try
    {
        //Create the instance for the typed dataset, DataSet2 which will
        //hold the [PurchaseOrderDetail] table details.
        //The dataset was created as part of the tutorial in Step 4.
        DataSet2 ds = new DataSet2();

        //Create a SQL Connection to the AdventureWorks2008 database using Windows
        Authentication.
        using (SqlConnection sqlconn = new SqlConnection("Data Source=.;Initial
Catalog=Adventureworks2014;Integrated Security=SSPI"))
        {
            //Building the dynamic query with the parameter ProductID.
            SqlCommand cmd = new SqlCommand("SELECT PurchaseOrderID, PurchaseOrderDetailID,
OrderQty, ProductID, ReceivedQty, RejectedQty, StockedQty FROM Purchasing.PurchaseOrderDetail
where ProductID = @ProductID", sqlconn);

            // Sets the productid parameter.
            cmd.Parameters.Add((new SqlParameter("@ProductID", SqlDbType.Int)).Value =
productid);

            SqlDataAdapter adap = new SqlDataAdapter(cmd);
            //Executing the QUERY and fill the dataset with the PurchaseOrderDetail table
data.
            adap.Fill(ds, "PurchaseOrderDetail");
        }

        //Return the PurchaseOrderDetail table for the Report Data Source.
        return ds.PurchaseOrderDetail;
    }
    catch
    {
        throw;
    }
}

```

To add an event handler that retrieves parameter values and binds the PurchaseOrdeDetail DataTable to the child report

1. Open Default.aspx in designer view.
2. Right-click the ReportViewer control, and then select **Properties**.
3. On the **Properties** pane, select the **Events** icon.
4. Double-click the **Drillthrough** event.

This will add an event handler section in the code that will look similar to the following.

```

protected void ReportViewer1_Drillthrough(object sender,
Microsoft.Reporting.WebForms.DrillthroughEventArgs e)
{
}

```

5. Complete the event handler. It should include the following functionality.

- a. Fetch the Child report object reference from the *DrillthroughEventArgs* parameter.

- b. Get the child report parameter list from the child report object fetched.
- c. Iterate through the parameter's collection and retrieve the value for the parameter, **ProductID**, passed from the parent report.
- d. Call the function, **GetPurchaseOrderDetail**, and pass the value for parameter **ProductID**.
- e. Bind the **PurchaseOrderDetail** DataTable with the Report's corresponding data source.

The completed event handler code will look similar to the following.

```

protected void ReportViewer1_Drillthrough(object sender,
Microsoft.Reporting.WebForms.DrillthroughEventArgs e)
{
    try
    {
        //Variable to store the parameter value passed from the MainReport.
        int productid = 0;

        //Get the instance of the Target report.
        LocalReport report = (LocalReport)e.Report;

        //Get all the parameters passed from the main report to the target report.
        //OriginalParametersToDrillthrough actually returns a Generic list of
        //type ReportParameter.
        IList<ReportParameter> list = report.OriginalParametersToDrillthrough;

        //Parse through each parameters to fetch the values passed along with them.
        foreach (ReportParameter param in list)
        {
            //Since we know the report has only one parameter and it is not a multivalued,
            //we can directly fetch the first value from the Values array.
            productid = Convert.ToInt32(param.Values[0].ToString());
        }

        //Binding the DataTable to the Child report dataset.
        //The name DataSet1 which can be located from,
        //Go to Design view of Child.rdlc, Click View menu -> Report Data
        //You'll see this name under DataSet2.
        report.DataSources.Add(new ReportDataSource("DataSet1",
GetPurchaseOrderDetail(productid)));
    }
    catch (Exception ex)
    {
        Response.Write(ex.Message);
    }
}

```

6. Save the file.

Next Task

You have successfully created a data filter for the data table that you defined for the child report. Next, you will build and run the website application. See [Lesson 9: Build and Run the Application](#).

Lesson 9: Build and Run the Application

10/1/2018 • 2 minutes to read • [Edit Online](#)

After you create a data filter for the data table, your next step is to build and run the website application.

To build and run the application

1. Press **CTRL+F5** to run the Default.aspx page without debugging, or press F5 to run the page with debugging.

As part of the build process, the report is compiled and any errors found (such as a syntax error in an expression used in the report) are added to the **Task List** that is located at the bottom of the Visual Studio window.

The webpage appears in the browser. The ReportViewer control displays the report. You can use the toolbar to browse through the report, zoom, and export the report to Excel.

2. Hover the mouse over any of the rows under **Name** column. The mouse cursor will display a Hand symbol.
3. Select a value in the **Name** column. The child report is shown with the corresponding filtered data.
4. Select the icon, **Go back to parent report**, in the **ReportViewer** tool bar to navigate back to the **Parent** report.
5. Close the browser to exit.

Report Builder Tutorials

11/30/2018 • 2 minutes to read • [Edit Online](#)

APPLIES TO: SQL Server 2016 Power BI Report Server

Use the following Report Builder tutorials to learn how to create basic Reporting Services paginated reports.

Prerequisites for Tutorials (Report Builder)

To use these tutorials, you must have read-only access to a SQL Server database and permissions to access a SQL Server 2016 Reporting Services (SSRS) report server.

For help with these prerequisites, contact your report server administrator.

Alternative Ways to Get a Data Connection (Report Builder)

Most of these tutorials use embedded data sources. If you have access to shared data sources you can use them instead. This topic provides the steps for using shared data sources.

Tutorial: Creating a Basic Table Report (Report Builder)

Follow the steps in this tutorial to learn how to create your first report. This tutorial shows you how to work with Report Builder to create a data source connection, define a simple query, build a layout to contain your data, format your report, and add grouping and totals.

Tutorial: Creating a Matrix Report (Report Builder)

Follow the steps in this tutorial to learn how to add and configure a matrix. You will use the Table or Matrix Wizard to create the report data source, dataset, and layout, and then enhance the matrix within the **Design** view of Report Builder.

Tutorial: Creating a Free Form Report (Report Builder)

Follow the steps in this tutorial to learn how to create a free-form report from scratch. This report resembles a newsletter.

Tutorial: Format Text (Report Builder)

Follow the steps in this tutorial to learn some of the many ways you can format text in your reports.

Tutorial: Add a Column Chart to Your Report (Report Builder)

Follow the steps in this tutorial to learn how to add a column chart with a moving average to a report.

Tutorial: Add a Pie Chart to Your Report (Report Builder)

Follow the steps in this tutorial to learn how to add a pie chart to a report.

Tutorial: Add a Bar Chart to Your Report (Report Builder)

Follow the steps in this tutorial to learn how to add a bar chart to a report.

Tutorial: Add a Sparkline to Your Report (Report Builder)

Follow the steps in this tutorial to learn how to create a matrix and then add a set of sparkline charts to the matrix.

Tutorial: Adding a KPI to Your Report (Report Builder)

Follow the steps in this tutorial to learn how to add a key performance indicator (KPI) to a report.

Tutorial: Map Report (Report Builder)

Follow the steps in this tutorial to learn how to add a map to a report.

Tutorial: Add a Parameter to Your Report (Report Builder)

Follow the steps in this tutorial to learn how to customize the appearance and content of a report by using

parameters.

[Tutorial: Creating Drillthrough and Main Reports \(Report Builder\)](#)

Follow the steps in this tutorial to learn how to create reports for a drillthrough scenario based on an Analysis Services cube. You will create the main report and enable it for drillthrough and the report that is the target of the drillthrough action.

[Tutorial: Introducing Expressions](#)

Follow the steps in this tutorial to learn how to use expressions to concatenate, calculate, and lookup field values and show them a report. You will also learn how to conditionally display different images and indicator states using expressions.

Next steps

[Report Design View](#)

[Report Builder in SQL Server](#)

[Tutorial: Create a Quick Chart Report Offline](#)

More questions? [Try asking the Reporting Services forum](#)

Prerequisites for Tutorials (Report Builder)

11/15/2018 • 2 minutes to read • [Edit Online](#)

To do the Report Builder tutorials, you need to be able to view and save SQL Server 2016 Reporting Services (SSRS) paginated reports on a report server or SharePoint site that is integrated with a report server. For data, all tutorials use literal queries that must be processed by an instance of SQL Server.

If you do not have access to a report server or site or to a data source, you can learn about Report Builder by building an offline report. See [Tutorial: Create a Quick Chart Report Offline \(Report Builder\)](#).

Requirements

You must have the following prerequisites to complete Report Builder tutorials:

- Access to Report Builder. You can run Report Builder from a Reporting Services report server or a Reporting Services report server in SharePoint integrated mode. Only the first step, how to open Report Builder, is different on the different servers.

On a report server, select **New > Paginated Report**.

On a report server in SharePoint integrated mode, on the **Documents** tab, select **New Document**, and from the drop-down list, select **Report Builder Report**. For example,

`https://<servername>/sites/mySite/reports`. The SharePoint administrator must enable the Report Builder Report feature for each document library.

- The URL to a Reporting Services report server or a SharePoint site that is integrated with a Reporting Services report server. You must have permission to save and view reports, shared data sources, shared datasets, report parts, and models. By default, the URL for a report server is
`https://<servername>/reportserver`. By default, the URL for a SharePoint site is `https://<sitename>` or
`https://<server>/site`.
- The name of a SQL Server instance and credentials sufficient for read-only access to any database. The dataset queries in the tutorials use literal data, but each query must be processed by a SQL Server instance to return the metadata that is required for a report dataset. For example, the following connection string specifies only a server: `data source=<servername>`. You must have read access to the default database that is assigned to you by the system administrator who grants you permission to access the server. You can also specify a database, as shown in the following connection string:
`data source=<servername>;initial catalog=<database>`.
- For the [Tutorial: Map Report \(Report Builder\)](#), the report server must be configured to support Bing maps as a background. For more information, see [Plan for Map Report Support](#).
- The [Tutorial: Creating Drillthrough and Main Reports \(Report Builder\)](#) tutorial requires access to the Contoso Sales cube. See the tutorial for more information.

The report server administrator must grant you the necessary permissions on the report server, configure Reporting Services folder locations, and configure Report Builder default options. For more information, see [Install and Uninstall Report Builder](#).

Next steps

[Report Builder tutorials](#)

More questions? [Try asking the Reporting Services forum](#)

Alternative Ways to Get a Data Connection (Report Builder)

10/1/2018 • 2 minutes to read • [Edit Online](#)

A data connection contains the information to connect to an external data source such as a SQL Server database. Usually, you get the connection information and the type of credentials to use from the data source owner.

To specify a data connection, you can use a shared data source from the report server or create an embedded data source that is used only in a specific report.

In most tutorials you use embedded data sources, but if you have access to shared data sources, then you can use them instead.

Getting a Data Connection From a Shared Data Source

If the report server has available shared data sources that you have permission to use, you can use them instead of an embedded data source. The following procedures tell how to locate the shared data sources and provide any credentials needed to use them.

To use a shared data source, you browse to a report server and select one. Usually, you get the report server URL from the report server administrator.

To specify a data connection from a list of shared data sources

1. In the New Table or Matrix or the New Chart Wizard, on the **Choose a dataset** page, select **Create a dataset**, and then click **Next**. The **Choose a connection to a data source** page opens.
2. From the list of data sources, select a data source that you have permission to access.
3. To verify that you can connect to the data source, click **Test Connection**. The message "Connection created successfully" appears. Click **OK**.
4. Click **Next**.

If necessary, enter your credentials. To save the credentials locally, select **Save password with connection**. If you don't select this option, you will be prompted for credentials every time that you run the report

5. Click **OK**.

To specify a data connection by browsing to a shared data source on a report server

1. In the New Table or Matrix or the New Chart Wizard, on the **Choose a dataset** page, select **Create a dataset**, and then click **Next**. The **Choose a connection to a data source** page opens.
2. Click **Browse**. The **Select Data Source** dialog box opens.
3. From the **Look in** drop-down list, select **Recent Sites and Servers**. In the data source pane, click the URL for your server, and then click **Open**.

The list of data sources or models appears.

4. Alternatively, in **Name**, type the URL to the report server. Click **Open**.

Report Builder connects to the report server and loads the data sources that are available at the root folder.

5. Navigate to a folder that contains a data source that you have sufficient permissions to connect to, select the data source, and then click **Open**.

You are back on the **Choose a connection to a data source** page.

6. To verify that you can connect to the data source, click **Test Connection**.

The message "Connection created successfully" appears. Click **OK**.

7. Click **Next**.

8. If you are prompted for a user name and password, enter your credentials. To save the credentials locally, select **Save password with connection**.

9. Click **OK**.

See Also

[Report Datasets \(SSRS\)](#)

[Report Builder Tutorials](#)

Tutorial: Creating a Basic Table Report (Report Builder)

11/30/2018 • 13 minutes to read • [Edit Online](#)

This tutorial teaches you to create a basic table report based on sample sales data. The following illustration shows the report you will create.

Product Sales				
Sales Date	Subcategory	Product	Quantity	Sales
January 5, 2009	Accessories	Carrying Case	68	\$9,924.60
		Mini Battery Charger	44	\$1,056.00
			112	\$10,980.60
	Digital	Slim Digital	44	\$8,357.80
			44	\$8,357.80
		Total	156	\$19,338.40
January 6, 2009	Accessories	Telephoto Conversion Lens	18	\$1,380.00
		Tripod	18	\$1,350.00
		USB Cable	26	\$780.00
			62	\$3,510.00
		Total	62	\$3,510.00
January 7, 2009	Digital	Compact Digital	84	\$10,836.00
			84	\$10,836.00
	Digital SLR	SLR Camera	88	\$26,576.00
			88	\$26,576.00
		Total	172	\$37,412.00
January 8, 2009	Accessories	Budget Movie-Maker	9	\$3,798.00
			9	\$3,798.00
	Digital	Consumer Digital	17	\$2,550.00
			17	\$2,550.00
		Total	26	\$6,348.00
January 9, 2009	Camcorders	Business Videographer	13	\$10,400.00
			13	\$10,400.00
	Digital SLR	SLR Camera 35mm	34	\$18,530.00
			34	\$18,530.00
		Total	47	\$28,930.00
January 10, 2009	Camcorders	Social Videographer	60	\$3,000.00
			60	\$3,000.00
		Total	60	\$3,000.00
January 11, 2009	Accessories	Lens Adapter	17	\$1,147.50
			17	\$1,147.50
	Digital	Advanced Digital	39	\$7,234.50
			39	\$7,234.50
		Total	56	\$8,382.00
		Total	579	\$106,920.40

Estimated time to complete this tutorial: 20 minutes.

Requirements

For more information about requirements, see [Prerequisites for Tutorials \(Report Builder\)](#).

1. Create a Report using a wizard

Create a table report with the Table or Matrix wizard. There are two modes: report design and shared dataset design. In report design mode, you specify data in the Report Data pane and the report layout on the design surface. In shared dataset design mode, you create dataset queries to share with others. In this tutorial, you will be using report design mode.

To create a report

1. Start Report Builder either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the left pane, verify that **New Report** is selected.
3. In the right pane, select **Table or Matrix Wizard**.

1a. Specify a Data Connection in the Table Wizard

A data connection contains the information to connect to an external data source such as a SQL Server database. Usually, you get the connection information and the type of credentials to use from the data source owner. To specify a data connection, you can use a shared data source from the report server or create an embedded data source that is used only in this report.

In this tutorial, you will use an embedded data source. To learn more about using a shared data sources, see [Alternative Ways to Get a Data Connection \(Report Builder\)](#).

To create an embedded data source

1. On the **Choose a dataset** page, select **Create a dataset**, and then click **Next**. The **Choose a connection to a data source** page opens.
2. Click **New**. The **Data Source Properties** dialog box opens.
3. In **Name**, type **Product_Sales** a name for the data source.
4. In **Select a connection type**, verify that **Microsoft SQL Server** is selected.
5. In **Connection string**, type the following text, where <servername> is the name of an instance of SQL Server:

```
Data Source=<servername>
```

Because you will use a query that contains the data instead of retrieving the data from a database, the connection string does not include the database name. For more information, see [Prerequisites for Tutorials \(Report Builder\)](#).

6. Click the **Credentials** tab. Enter the credentials that you need to access the external data source.
7. Click the General tab again. To verify that you can connect to the data source, click **Test Connection**.

The message "Connection created successfully" appears.

8. Click **OK**.

You are back on the **Choose a connection to a data source** page, with your new data source selected.

9. Click **Next**.

1b. Create a Query in the Table Wizard

In a report, you can use a shared dataset that has a predefined query, or you can create an embedded dataset for use only in this one report. In this tutorial, you will create an embedded dataset.

NOTE

In this tutorial, the query contains the data values, so that it does not need an external data source. This makes the query quite long. In a business environment, a query would not contain the data. This is for learning purposes only.

To create a query

1. On the **Design a query** page, the relational query designer is open. For this tutorial, you will use the text-based query designer.

Click **Edit As Text**. The text-based query designer displays a query pane and a results pane.

2. Paste the following Transact-SQL query into the blank upper box.

```
SELECT CAST('2009-01-05' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Carrying Case' as Product, CAST(9924.60 AS money) AS Sales, 68 as Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Tripod' as Product, CAST(1350.00 AS money) AS Sales, 18 as Quantity
UNION SELECT CAST('2009-01-11' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Lens Adapter' as Product, CAST(1147.50 AS money) AS Sales, 17 as Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Mini Battery Charger' as Product, CAST(1056.00 AS money) AS Sales, 44 as Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Telephoto Conversion Lens' as Product, CAST(1380.00 AS money) AS Sales, 18 as Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Accessories' as Subcategory,
    'USB Cable' as Product, CAST(780.00 AS money) AS Sales, 26 as Quantity
UNION SELECT CAST('2009-01-08' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Budget Movie-Maker' as Product, CAST(3798.00 AS money) AS Sales, 9 as Quantity
UNION SELECT CAST('2009-01-09' AS date) as SalesDate, 'Camcorders' as Subcategory,
    'Business Videographer' as Product, CAST(10400.00 AS money) AS Sales, 13 as Quantity
UNION SELECT CAST('2009-01-10' AS date) as SalesDate, 'Camcorders' as Subcategory,
    'Social Videographer' as Product, CAST(3000.00 AS money) AS Sales, 60 as Quantity
UNION SELECT CAST('2009-01-11' AS date) as SalesDate, 'Digital' as Subcategory,
    'Advanced Digital' as Product, CAST(7234.50 AS money) AS Sales, 39 as Quantity
UNION SELECT CAST('2009-01-07' AS date) as SalesDate, 'Digital' as Subcategory,
    'Compact Digital' as Product, CAST(10836.00 AS money) AS Sales, 84 as Quantity
UNION SELECT CAST('2009-01-08' AS date) as SalesDate, 'Digital' as Subcategory,
    'Consumer Digital' as Product, CAST(2550.00 AS money) AS Sales, 17 as Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Digital' as Subcategory,
    'Slim Digital' as Product, CAST(8357.80 AS money) AS Sales, 44 as Quantity
UNION SELECT CAST('2009-01-09' AS date) as SalesDate, 'Digital SLR' as Subcategory,
    'SLR Camera 35mm' as Product, CAST(18530.00 AS money) AS Sales, 34 as Quantity
UNION SELECT CAST('2009-01-07' AS date) as SalesDate, 'Digital SLR' as Subcategory,
    'SLR Camera' as Product, CAST(26576.00 AS money) AS Sales, 88 as Quantity
```

3. On the query designer toolbar, click **Run (!)**.

The query runs and displays the result set for the fields SalesDate, Subcategory, Product, Sales, and Quantity.

In the result set, the column headings are based on the names in the query. In the dataset, the column headings become the field names, and are saved in the report. After you complete the wizard, you can use the Report Data pane to view the collection of dataset fields.

4. Click **Next**.

1c. Organize Data into Groups in the Table Wizard

When you select fields to group on, you design a table that has rows and columns that display detail data and aggregated data.

To organize data into groups

1. On the **Arrange fields** page, drag Product to **Values**.
2. Drag Quantity to **Values** and place below Product.

Quantity is automatically aggregated by the Sum function, the default aggregate for numeric fields. The value is [Sum(Quantity)].

Select the arrow next to [Sum(Quantity)] to view the other aggregate functions available. Do not change the aggregate function.

3. Drag Sales to **Values** and place below [Sum(Quantity)].

Sales is aggregated by the Sum function. The value is [Sum(Sales)].

Steps 1, 2, and 3 specify the data to display in the table.

4. Drag SalesDate to **Row groups**.

5. Drag Subcategory to **Row groups** and place below SalesDate.

Steps 4 and 5 organize the values for the fields first by date, and then by product subcategory for that date.

6. Click **Next**.

1d. Add Subtotal and Total Rows in the Table Wizard

After you create groups, you can add and format rows on which to display aggregate values for the fields. You can choose whether to show all the data or to let a user expand and collapse grouped data interactively.

To add subtotals and totals

1. On the **Choose the layout** page, under **Options**, verify that **Show subtotals and grand totals** is selected.

2. Verify that **Blocked, subtotal below** is selected.

The wizard Preview pane displays a table with five rows. When you run the report, each row will display in the following way:

- a. The first row will repeat once for the table to show column headings.
 - b. The second row will repeat once for each line item in the sales order and display the product name, order quantity, and line total.
 - c. The third row will repeat once for each sales order to display subtotals per order.
 - d. The fourth row will repeat once for each order date to display the subtotals per day.
 - e. The fifth row will repeat once for the table to display the grand totals.
3. Clear the option **Expand/collapse groups**. In this tutorial, the report you create does not use the drilldown feature that lets a user expand a parent group hierarchy to display child group rows and detail rows.
 4. Click **Next** to preview the table, then click **Finish**.

The table is added to the design surface. The table has 5 columns and 5 rows. The Row Groups pane shows three row groups: SalesDate, Subcategory, and Details. Detail data is all the data that is retrieved by the dataset query.

2. Format Data as Currency

By default, the summary data for the Sales field displays a general number. Format it to display the number as currency.

To format a currency field

1. To see formatted text boxes and placeholder text as sample values in Design View, on the **Home** tab, in the **Number** group, click the arrow next to the **Placeholder Styles** icon > **Sample Values**.
2. Click the cell in the second row (under the column headings row) in the Sales column and drag down to

select all cells that contain **[Sum(Sales)]**.

3. On the **Home** tab, in the **Number** group, click the **Currency** button. The cells change to show the formatted currency.

If your regional setting is English (United States), the default sample text is **[\$12,345.00]**. If you do not see an example currency value, on the **Home** tab, in the **Number** group, click the arrow next to the **Placeholder Styles** icon > **Sample Values**.

4. Click **Run** to preview your report.

The summary values for Sales display as currency.

3. Format Data as Date

By default, the SalesDate field displays both date and time. You can format them to display only the date.

To format a date field as the default format

1. Click **Design** to return to design view.
2. Click the cell that contains **[SalesDate]**.
3. On the Ribbon, on the **Home** tab, in the **Number** group, click the arrow and select **Date**.

The cell displays the example date **[1/31/2000]**. If you do not see an example date, on the **Home** tab, in the **Number** group, click the arrow next to the **Placeholder Styles** icon > **Sample Values**.

4. Click **Run** to preview the report.

The SalesDate values display in the default date format.

To change the date format to a custom format

1. Click **Design** to return to design view.
2. Select the cell that contains **[SalesDate]**.
3. On the **Home** tab, in the **Number** group, click the arrow in the lower-right corner to open the dialog box.

The **Text Box Properties** dialog box opens.

4. In the Category pane, verify that **Date** is selected.
5. In the **Type** pane, select **January 31, 2000**.
6. Click **OK**.

The cell displays the example date **[January 31, 2000]**.

7. Click **Run** to preview your report.

The SalesDate value displays the name of the month instead of the number for the month.

4. Change Column Widths

By default, each cell in a table contains a text box. A text box expands vertically to accommodate text when the page is rendered. In the rendered report, each row expands to the height of the tallest rendered text box in the row. The height of the row on the design surface has no affect on the height of the row in the rendered report.

To reduce the amount of vertical space each row takes, expand the column width to accommodate the expected contents of the text boxes in the column on one line.

To change the width of table columns

1. Click **Design** to return to design view.
2. Click the table so that column and row handles appear above and next to the table.
The gray bars along the top and side of the table are the column and row handles.
3. Point to the line between column handles so that the cursor changes into a double arrow. Drag the columns to the width you want. For example, expand the column for Product so that the product name displays on one line.
4. Click **Run** to preview your report.

5. Add a Report Title

A report title appears at the top of the report. You can place the report title in a report header or if the report does not use one, in a text box at the top of the report body. In this tutorial, you will use the text box that is automatically placed at the top of the report body.

The text can be further enhanced by applying different font styles, sizes, and colors to phrases and individual characters of the text. For more information, see [Format Text in a Text Box \(Report Builder and SSRS\)](#).

To add a report title

1. On the design surface, click **Click to add title**.
2. Type **Product Sales**, and then click outside the text box.
3. Right-click the text box that contains **Product Sales** and click **Text Box Properties**.
4. In the **Text Box Properties** dialog box, click **Font**.
5. In the **Size** list, select **18pt**.
6. In the **Color** list, select **Cornflower Blue**.
7. Select **Bold**.
8. Click **OK**.

6. Save the Report

Save the report to a report server or your computer. If you do not save the report to the report server, a number of Reporting Services features such as report parts and subreports are not available.

To save the report on a report server

1. Click **File > Save As**.
2. Click **Recent Sites and Servers**.
3. Select or type the name of the report server where you have permission to save reports.

The message "Connecting to report server" appears. When the connection is complete, you see the contents of the report folder that the report server administrator specified as the default location for reports.

4. In **Name**, replace **Untitled** with **Product_Sales**.
5. Click **Save**.

The report is saved to the report server. The name of report server that you are connected to appears in the status bar at the bottom of the window.

To save the report on your computer

1. Click **File > Save As**.
2. Click **Desktop, My Documents, or My computer**, and browse to the folder where you want to save the report.
3. In **Name**, replace **Untitled** with **Product Sales**.
4. Click **Save**.

7. Export the Report

Reports can be exported to different formats such Microsoft Excel and comma separated value (CSV) files. For more information, see [Export Reports \(Report Builder and SSRS\)](#).

In this tutorial, you will export the report to Excel and set a property on the report to provide a custom name for the workbook tab.

To specify the workbook tab name

1. Click **Design** to return to design view.
2. Click anywhere on the design surface, outside the report.
3. In the Properties pane, locate the **InitialPageName** property and type **Product Sales Excel**.

NOTE

If the Properties pane is not visible, on the **View** tab, select **Properties**.

If you don't see a property in the Properties pane, try selecting the **Alphabetical** button at the top of the pane to order all the properties alphabetically.

To export a report to Excel

1. Click **Run** to preview the report.
2. On the ribbon, click **Export > Excel**.
The opens.
3. In **Save As** dialog box, browse to where you want to save the file.
4. In the **File name** box, type **Product_Sales_Excel**.
5. Verify that the file type is **Excel (*.xlsx)**.
6. Click **Save**.

To view the report in Excel

1. Open the folder where you save the workbook and double-click **Product_Sales_Excel.xlsx**.
2. Verify that the name of the workbook tab is **Product Sales Excel**.

Next Steps

This concludes the walkthrough for how to create a basic table report. For more information about tables, see [Tables, Matrices, and Lists \(Report Builder and SSRS\)](#).

See Also

[Report Builder Tutorials](#)

[Report Builder in SQL Server](#)

Tutorial: Creating a Matrix Report (Report Builder)

11/30/2018 • 14 minutes to read • [Edit Online](#)

This tutorial teaches you to create a Reporting Services paginated report with a matrix of sample sales data in nested row and column groups.

You also create an adjacent column group, format columns, and rotate text. The following illustration shows a report similar to the one you will create.

Sales by Territory, Subcategory, and Day									
Sales Date	Accessories		Digital		Monday		Tuesday		Total
	Total	Sales	Total	Sales	QTY	Sales	Sales	Sales	
	1/5/2015	\$19,494.10	103	\$17,409.10	117	\$36,903.20		\$36,903.20	220
Central	1/6/2015	\$15,584.80	83	\$13,747.40	82		\$29,332.20	\$29,332.20	165
	Total	\$35,078.90	186	\$31,156.50	199	\$36,903.20	\$29,332.20	\$66,235.40	385
	1/5/2015	\$16,289.75	90	\$17,129.80	112	\$33,419.55		\$33,419.55	202
North	1/6/2015	\$14,630.10	84	\$18,341.70	111		\$32,971.80	\$32,971.80	195
	Total	\$30,919.85	174	\$35,471.50	223	\$33,419.55	\$32,971.80	\$66,391.35	397
	1/5/2015	\$11,790.65	73	\$19,885.55	131	\$31,676.20		\$31,676.20	204
South	1/6/2015	\$12,887.95	78	\$15,291.25	102		\$28,179.20	\$28,179.20	180
	Total	\$24,678.60	151	\$35,176.80	233	\$31,676.20	\$28,179.20	\$59,855.40	384
	Total	\$90,677.35	511	\$101,804.80	655	\$101,998.95	\$90,483.20	\$192,482.15	1166

Estimated time to complete this tutorial: 20 minutes.

Requirements

For information about requirements, see [Prerequisites for Tutorials](#).

1. Create a Matrix Report and Dataset from the New Table or Matrix Wizard

In this section, you choose a shared data source, create an embedded dataset, and then display the data in a matrix.

NOTE

In this tutorial, the query already contains the data values, so that it does not need an external data source. This makes the query quite long. In a business environment, a query would not contain the data. This is for learning purposes only.

To create a matrix

1. Start Report Builder either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the left pane, verify that **New Report** is selected.
3. In the right pane, click **Table or Matrix Wizard**.

4. On the **Choose a dataset** page, click **Create a dataset**.
5. Click **Next**.
6. On the **Choose a connection to a data source** page, select an existing data source, or browse to the report server and select a data source. If no data source is available or you do not have access to a report server, you can use an embedded data source instead. For information about creating an embedded data source, see [Tutorial: Creating a Basic Table Report \(Report Builder\)](#).
7. Click **Next**.
8. On the **Design a query** page, click **Edit as Text**.
9. Copy and paste the following query into the query pane:

```

SELECT CAST('2015-01-05' AS date) as SalesDate, 'Central' as Territory, 'Accessories' as
Subcategory,'Carrying Case' as Product, CAST(16996.60 AS money) AS Sales, 68 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'North' as Territory, 'Accessories' as
Subcategory, 'Carrying Case' as Product, CAST(13747.25 AS money) AS Sales, 55 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'South' as Territory, 'Accessories' as
Subcategory,'Carrying Case' as Product, CAST(9248.15 AS money) As Sales, 37 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Central' as Territory, 'Accessories' as
Subcategory,'Tripod' as Product, CAST(1350.00 AS money) AS Sales, 18 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'North' as Territory, 'Accessories' as
Subcategory,'Tripod' as Product, CAST(1800.00 AS money) AS Sales, 24 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'South' as Territory, 'Accessories' as
Subcategory,'Tripod' as Product, CAST(1125.00 AS money) AS Sales, 15 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Central' as Territory, 'Accessories' as
Subcategory,'Lens Adapter' as Product, CAST(1147.50 AS money) AS Sales, 17 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'North' as Territory, 'Accessories' as
Subcategory, 'Lens Adapter' as Product, CAST(742.50 AS money) AS Sales, 11 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'South' as Territory, 'Accessories' as
Subcategory,'Lens Adapter' as Product, CAST(1417.50 AS money) AS Sales, 21 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Central' as Territory, 'Accessories' as
Subcategory, 'Carrying Case' as Product, CAST(13497.30 AS money) AS Sales, 54 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'North' as Territory, 'Accessories' as
Subcategory, 'Carrying Case' as Product, CAST(11997.60 AS money) AS Sales, 48 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'South' as Territory, 'Accessories' as
Subcategory, 'Carrying Case' as Product, CAST(10247.95 AS money) As Sales, 41 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Central' as Territory, 'Accessories' as
Subcategory, 'Tripod' as Product, CAST(1200.00 AS money) AS Sales, 16 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'North' as Territory, 'Accessories' as
Subcategory,'Tripod' as Product, CAST(2025.00 AS money) AS Sales, 27 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'South' as Territory, 'Accessories' as
Subcategory,'Tripod' as Product, CAST(1425.00 AS money) AS Sales, 19 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Central' as Territory, 'Accessories' as
Subcategory,'Lens Adapter' as Product, CAST(887.50 AS money) AS Sales, 13 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'North' as Territory, 'Accessories' as
Subcategory, 'Lens Adapter' as Product, CAST(607.50 AS money) AS Sales, 9 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'South' as Territory, 'Accessories' as
Subcategory,'Lens Adapter' as Product, CAST(1215.00 AS money) AS Sales, 18 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Central' as Territory, 'Digital' as
Subcategory,'Compact Digital' as Product, CAST(10191.00 AS money) AS Sales, 79 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'North' as Territory, 'Digital' as Subcategory,
'Compact Digital' as Product, CAST(8772.00 AS money) AS Sales, 68 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'South' as Territory, 'Digital' as Subcategory,
'Compact Digital' as Product, CAST(10578.00 AS money) AS Sales, 82 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Central' as Territory,'Digital' as Subcategory,
'Slim Digital' as Product, CAST(7218.10 AS money) AS Sales, 38 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'North' as Territory,'Digital' as Subcategory,
'Slim Digital' as Product, CAST(8357.80 AS money) AS Sales, 44 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'South' as Territory,'Digital' as
Subcategory,'Slim Digital' as Product, CAST(9307.55 AS money) AS Sales, 49 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Central' as Territory, 'Digital' as
Subcategory,'Compact Digital' as Product, CAST(3870.00 AS money) AS Sales, 30 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'North' as Territory, 'Digital' as
Subcategory, 'Compact Digital' as Product, CAST(5805.00 AS money) AS Sales, 45 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'South' as Territory, 'Digital' as Subcategory,
'Compact Digital' as Product, CAST(8643.00 AS money) AS Sales, 67 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Central' as Territory, 'Digital' as Subcategory,
'Slim Digital' as Product, CAST(9877.40 AS money) AS Sales, 52 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'North' as Territory, 'Digital' as Subcategory,
'Slim Digital' as Product, CAST(12536.70 AS money) AS Sales, 66 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'South' as Territory, 'Digital' as Subcategory,
'Slim Digital' as Product, CAST(6648.25 AS money) AS Sales, 35 as Quantity

```

10. (optional) Click the Run icon (!) to run the query and see the data.

11. Click **Next**.

2. Organize Data and Choose Layout from the New Table or Matrix Wizard

Use the wizard to provide a starting design on which to display data. The preview pane in the wizard helps you to visualize the result of grouping data before you complete the matrix design.

1. On the **Arrange fields** page, drag Territory from **Available fields** to **Row groups**.

2. Drag SalesDate to **Row groups** and place it below Territory.

The order in which fields are listed in **Row groups** defines the group hierarchy. Steps 1 and 2 organize the values of the fields first by territory, and then by sales date.

3. Drag Subcategory to **Column groups**.

4. Drag Product to **Column groups** and place it below Subcategory.

Again, the order in which fields are listed in **Column groups** defines the group hierarchy. Steps 3 and 4 organize the values for the fields first by subcategory, and then by product.

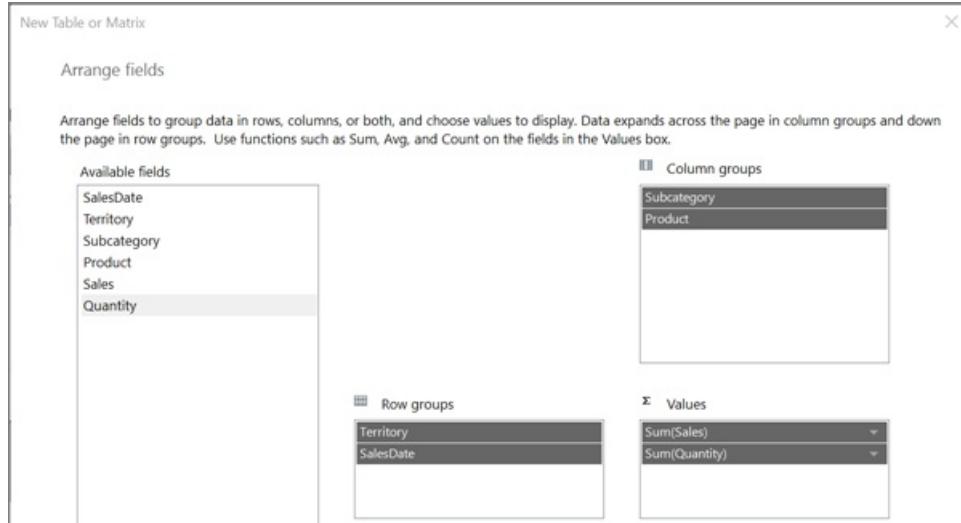
5. Drag Sales to **Values**.

Sales is summarized with the Sum function, the default function to summarize numeric fields.

6. Drag Quantity to **Values**.

Quantity is summarized with the Sum function.

Steps 5 and 6 specify the data to display in the matrix data cells.



7. Click **Next**.

8. On the Choose the Layout page, under **Options**, verify that **Show subtotals and grand totals** is selected.

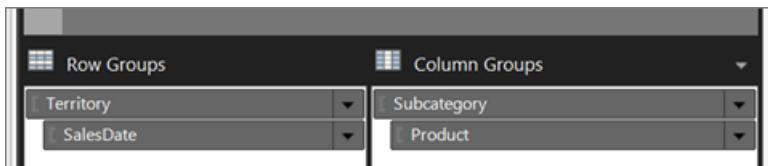
9. Verify that **Blocked, subtotal below** is selected.

10. Verify the option **Expand/collapse groups** is selected.

11. Click **Next**.

12. Click **Finish**.

The matrix is added to the design surface. The Row Groups pane shows two row groups: Territory and SalesDate. The Column Groups pane shows two column groups: Subcategory and Product. Detail data is all the data that is retrieved by the dataset query.



13. Click **Run** to preview the report.

For each product that is sold on a specific date, the matrix shows the subcategory to which the product belongs and the territory of the sales.

14. Expand a subcategory. You can see the report quickly gets quite wide.

		Accessories						Digital		Total	
		Carrying Case		Lens Adapter		Tripod		Total		Total	
Territory	Sales Date	Sales	Quantity	Sales	Quantity	Sales	Quantity	Sales	Quantity	Sales	Quantity
Central	1/5/2015	16996.6000	68	1147.5000	17	1350.0000	18	19494.1000	103	17409.1000	117
	12:00:00 AM										
	1/6/2015	13497.3000	54	887.5000	13	1200.0000	16	15584.8000	83	13747.4000	82
	12:00:00 AM										
Total		30493.9000	122	2035.0000	30	2550.0000	34	35078.9000	186	31156.5000	199
North	Total	25744.8500	103	1350.0000	20	3825.0000	51	30919.8500	174	35471.5000	223
South	Total	19496.1000	78	2632.5000	39	2550.0000	34	24678.6000	151	35176.8000	233
Total		75734.8500	303	6017.5000	89	8925.0000	119	90677.3500	511	101804.8000	655

3. Format Data

By default, the summary data for the Sales field displays a general number and the SalesDate field displays both date and time information. In this section, you format the Sales field to display the number as currency and the SalesDate field to display only the date. Toggle **Placeholder Styles** to display formatted text boxes and placeholder text as sample values.

To format fields

1. Click **Design** to switch to design view.
2. Press the Ctrl key, and then select the nine cells that contain `[Sum(Sales)]`.
3. On the **Home** tab > **Number** > **Currency**. The cells change to show the formatted currency.

If your regional setting is English (United States), the default sample text is **[\$12,345.00]**. If you do not see an example currency value, in the **Numbers** group, click **Placeholder Styles** > **Sample Values**.

4. Click the cell that contains `[SalesDate]`.

5. In the **Number** group > **Date**.

The cell displays the example date **[1/31/2000]**. If you do not see an example date, click **Placeholder Styles** in the **Numbers** group, and then click **Sample Values**.

6. Click **Run** to preview your report.

The date values display only dates and the sales values display as currency.

4. Add Adjacent Column Group

You can nest row and column groups in parent-child relationships, or adjacent in sibling relationships.

In this section, you add a column group adjacent to the Subcategory column group, copy cells to populate the new column group, and then use an expression to create the value of the column group header.

To add an adjacent column group

1. Click **Design** to return to design view.
 2. Right-click the cell that contains [Subcategory], point to **Add Group**, and then click **Adjacent Right**.

The **Tablix Group** dialog box opens.

3. In the **Group By** list, select SalesDate, and then click **OK**.

A new column group is added to the right of the Subcategory column group.

- Right-click the cell in the new column group that contains [SalesDate], and then click **Expression**.
 - Copy the following expression to the expression box.

```
=WeekdayName(DatePart("w",Fields!SalesDate.Value))
```

This expression extracts the weekday name from the sales date. For more information, see [Expressions \(Report Builder and SSRS\)](#).

6. Right-click the cell in the Subcategory column group that contains Total, and then click **Copy**.
 7. Right-click the cell immediately below the cell that contains the expression you created in step 5 and click **Paste**.
 8. Press the Ctrl key.
 9. In the Subcategory group, click the Sales column header and the three cells below it, right-click, and then click **Copy**.
 10. Paste the four cells into the four empty cells in the new column group.
 11. Click **Run** to preview the report.

The report includes columns named Monday and Tuesday. The dataset contains only data for these two days.

		Accessories			
		Total	Total	Total	
Territory	Sales Date	Sales		Sales	Quantity
■ Central	1/5/2015	\$19,494.10		\$36,903.20	\$36,903.20 220
	1/6/2015	\$15,584.80		\$29,332.20	\$29,332.20 165
	Total	\$35,078.90		\$36,903.20	\$66,235.40 385
■ North	Total	\$30,919.85		\$33,419.55	\$66,391.35 397
■ South	Total	\$24,678.60		\$31,676.20	\$59,855.40 384
Total		\$90,677.35		\$101,998.95	\$90,483.20 \$192,482.15 1166

NOTE

If the data included other days, the report would include columns for them as well. Each column has the column header, **Sales**, and sales totals by territory.

5. Change Column Widths

A report that includes a matrix typically expands horizontally as well as vertically when it runs. Controlling horizontal expansion is particularly important if you plan to export the report to formats such as Microsoft Word or Adobe PDF that are used for printed reports. If the report expands horizontally across multiple pages, the printed report is difficult to understand. To minimize horizontal expansion, you can resize columns to be only the width necessary to display the data without wrapping. You can also rename columns so that their titles fit the width needed to display the data.

To rename and resize the columns

1. Click **Design** to return to design view.
2. Select the text in the furthest Quantity column to the left, and then type **QTY**.

The column title is now QTY.

3. Repeat step 2 for the two other columns named Quantity.
4. Click the matrix so that column and row handles appear above and next to the matrix.

The gray bars along the top and side of the table are the column and row handles.

		[Subcategory]		«Expr»		Total
		[Product]		Total		Total
Territory	Sales Date	Sales	QTY	Sales	QTY	Sales
[Territory]	[1/31/2000]	[\$12,345.00]	[Sum(Quantity)]	[\$12,345.00]	[Sum(Quantity)]	[\$12,345.00]
	Total	[\$12,345.00]	[Sum(Quantity)]	[\$12,345.00]	[Sum(Quantity)]	[\$12,345.00]
	Total	[\$12,345.00]	[Sum(Quantity)]	[\$12,345.00]	[Sum(Quantity)]	[\$12,345.00]

5. To resize the QTY column farthest to the left, point to the line between column handles so that the cursor changes into a double arrow. Drag the column towards the left until it is 1/2 inch wide.

A column width of 1/2 inch is adequate to display the quantity.

6. Repeat step 5 for the other columns named QTY.
7. Click **Run** to preview your report.

The columns that contain quantities are now narrower and are named QTY.

6. Merge Matrix Cells

The corner area is in the upper left corner of the matrix. Depending on the number of row and column groups in the matrix, the number of cells in the corner area varies. The matrix, built in this tutorial, has four cells in its corner area. The cells are arranged in two rows and two columns, reflecting the depth of row and column group hierarchies. The four cells are not used in this report and you will merge them into one.

To merge matrix cells

1. Click **Design** to return to design view.
2. Click the matrix so that column and row handles appear above and next to the matrix.
3. Press the Ctrl key and select the four corner cells.

4. Right-click the cells and click **Merge Cells**.
5. Right-click the new merged cell and click **Text Box Properties**.
6. On the **Border** tab > **Presets** > **None**.
7. Click **OK**.
8. Click **Run** to preview your report.

The cell in the upper corner of the matrix is no longer visible.

7. Add a Report Header and Report Title

A report title appears at the top of the report. You can place the report title in a report header or if the report does not use one, in a text box at the top of the report body. In this tutorial, you will remove the text box at the top of the report and add a title to the header.

To add a report header and report title

1. Click **Design** to return to design view.
 2. Select the text box at the top of the report body that contains **Click to add title**, and then press the Delete key.
 3. On the **Insert** tab > **Header** > **Add Header**.
- A header is added to the top of the report body.
4. On the **Insert** tab, click **Text Box**, and then drag a text box inside the report header. Make the text box about 6 inches long and 3/4 inch tall and place it on the left side of the report header.
 5. In the text box, type **Sales by Territory, Subcategory, and Day**.
 6. Select the text you typed, on the **Home** tab > **Font**:
 - **Size 24 pt**
 - **Color Maroon**

7. Click **Run** to preview the report.

The report includes a report title in the report header.

8. Save the Report

You can save reports to a report server, SharePoint library, or your computer.

In this tutorial, save the report to a report server. If you do not have access to a report server, save the report to your computer.

To save the report on a report server

1. From the **Report Builder** button, click **Save As**.
2. Click **Recent Sites and Servers**.
3. Select or type the name of the report server where you have permission to save reports.

The message "Connecting to report server" appears. When the connection is complete, you will see the contents of the report folder that the report server administrator specified as the default report location.

4. In **Name**, replace the default name with **SalesByTerritorySubcategory**.
5. Click **Save**.

The report is saved to the report server. The name of report server that you are connected to appears in the status bar at the bottom of the window.

To save the report on your computer

1. From the **Report Builder** button, click **Save As**.
2. Click **Desktop**, **My Documents**, or **My computer**, and then browse to the folder where you want to save the report.
3. In **Name**, replace the default name with **SalesByTerritorySubcategory**.
4. Click **Save**.

9. (Optional) Rotate Text Box 270 Degrees

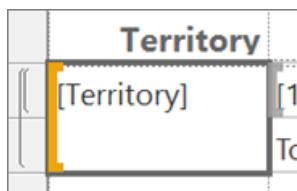
A report with matrices can expand horizontally and vertically when it runs. By rotating text boxes vertically, or 270 degrees, you can save horizontal space. The rendered report is then narrower and if exported to a format such as Microsoft Word, will be more likely to fit on a printed page.

A text box can also display text as horizontal, vertical (top to bottom). For more information, see [Text Boxes \(Report Builder and SSRS\)](#).

To rotate text box 270 degrees

1. Click **Design** to return to design view.
2. Select the cell that contains **[Territory]**.

Note: Select the cell, not the text. The **WritingMode** property is only available for the cell.



3. In the Properties pane, locate the **WritingMode** property and change it from **Default** to **Rotate270**.

If the Properties pane is not open, click the **View** tab of the ribbon, and then select **Properties**.

4. Verify that the **CanGrow** property is set to **True**.
5. On the **Home** tab > **Paragraph** section, select **Middle** and **Center** to locate the text in the center of the cell both vertically and horizontally.
6. Resize the Territory column to be 1/2 inch wide and delete the column title.
7. Click **Run** to preview your report.

The territory name is written vertically, bottom to top. The height of the Territory row group varies by the length of the territory name.

Next Steps

This concludes the tutorial for how to create a matrix report. For more information about matrices, see:

- [Tables, Matrices, and Lists](#)
- [Create a Matrix](#)
- [Tablix Data Region Areas](#)

- [Tablix Data Region Cells, Rows, and Columns](#)

See Also

[Report Builder Tutorials](#)

[Report Builder in SQL Server](#)

Tutorial: Creating a Free Form Report (Report Builder)

1/9/2019 • 17 minutes to read • [Edit Online](#)

In this tutorial, you create a paginated report that acts as a newsletter. Each page displays static text, summary visuals, and detailed sample sales data.

Newsletter for Central

Hello Lauren Johnson,

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin sed dolor in ipsum pulvinar egestas. Sed sed lacus at leo ornare ultricies. Vivamus velit risus, euismod nec sodales gravida, gravida in dui. Etiam ullamcorper elit vitae justo fermentum ut ullamcorper augue sodales.

Ut placerat, nisl quis feugiat adipiscing, nibh est aliquet est, mollis faucibus mauris lectus quis arcu. In mollis tincidunt lacinia. In vitae erat ut lorem tincidunt luctus. Curabitur et magna nunc, sit amet adipiscing nisi. Nulla rhoncus elementum orci nec tincidunt.

Aliquam imperdiet cursus erat vel tincidunt. Donec et neque ac urna rutrum sodales. In id purus et nisl dignissim dapibus. Sed rhoncus metus at felis feugiat eu tempor dolor vehicula. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam faucibus consectetur diam eu pellentesque.

Congratulations on your total sales of \$66,235.40!

Product	Sales Date	Quantity	Sales
Carrying Case	1/5/2009	68	\$16,996.60
Lens Adapter	1/5/2009	17	\$1,147.50
Tripod	1/5/2009	18	\$1,350.00
Compact Digital	1/5/2009	79	\$10,191.00
Slim Digital	1/5/2009	38	\$7,218.10
Carrying Case	1/6/2009	54	\$13,497.30
Lens Adapter	1/6/2009	13	\$887.50
Tripod	1/6/2009	16	\$1,200.00
Compact Digital	1/6/2009	30	\$3,870.00
Slim Digital	1/6/2009	52	\$9,877.40
Total		385	\$66,235.40

Product Quantities Sold



Product Sales



The report groups information by territory and displays the name of the sales manager for the territory as well as detailed and summary sales information. You start with a list data region as the foundation for the free form report, then add a decorative panel with an image, static text with data inserted, a table to show detailed information, and optionally, pie and column charts to display summary information.

Estimated time to complete this tutorial: 20 minutes.

Requirements

For more information about requirements, see [Prerequisites for Tutorials \(Report Builder\)](#).

1. Create a Blank Report, Data Source, and Dataset

NOTE

In this tutorial, the query contains the data values so that it does not need an external data source. This makes the query quite long. In a business environment, a query would not contain the data. This is for learning purposes only.

To create a blank report

1. [Start Report Builder](#) either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the left pane, make sure **New Report** is selected.
3. In the right pane, click **Blank Report**.

To create a new data source

1. In the Report Data pane, click **New > Data Source**.
2. In the **Name** box, type: **ListDataSource**
3. Click **Use a connection embedded in my report**.
4. Verify that the connection type is Microsoft SQL Server, and then in the **Connection string** box type: **Data Source = <servername>**

<servername>, for example Report001, specifies a computer on which an instance of the SQL Server Database Engine is installed. Because the data for this report is not extracted from a SQL Server database, you need not include the name of a database. The default database on the specified server is just used to parse the query.

5. Click **Credentials**, and enter the credentials needed to connect to the instance of the SQL Server Database Engine.
6. Click **OK**.

To create a new dataset

1. In the Report Data pane, click **New > Dataset**.
2. In the **Name** box, type: **ListDataset**.
3. Click **Use a dataset embedded in my report**, and verify that the data source is **ListDataSource**.
4. Verify that the **Text** query type is selected, and then click **Query Designer**.
5. Click **Edit as Text**.
6. Copy and paste the following query into the query pane:

```
SELECT CAST('2009-01-05' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as Territory,
'Accessories' as Subcategory,'Carrying Case' as Product, CAST(16996.60 AS money) AS Sales, 68 as
Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory,
'Accessories' as Subcategory, 'Carrying Case' as Product, CAST(13747.25 AS money) AS Sales, 55 as
Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory,
'Accessories' as Subcategory,'Carrying Case' as Product, CAST(9248.15 AS money) AS Sales, 37 as Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as
Territory, 'Accessories' as Subcategory,'Tripod' as Product, CAST(1350.00 AS money) AS Sales, 18 as
Quantity
```

```

UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory,
'Accessories' as Subcategory,'Tripod' as Product, CAST(1800.00 AS money) AS Sales, 24 as Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory,
'Accessories' as Subcategory,'Tripod' as Product, CAST(1125.00 AS money) AS Sales, 15 as Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as
Territory, 'Accessories' as Subcategory,'Lens Adapter' as Product, CAST(1147.50 AS money) AS Sales, 17
as Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory,
'Accessories' as Subcategory, 'Lens Adapter' as Product, CAST(742.50 AS money) AS Sales, 11 as Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory,
'Accessories' as Subcategory,'Lens Adapter' as Product, CAST(1417.50 AS money) AS Sales, 21 as Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as
Territory, 'Accessories' as Subcategory, 'Carrying Case' as Product, CAST(13497.30 AS money) AS Sales,
54 as Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory,
'Accessories' as Subcategory, 'Carrying Case' as Product, CAST(11997.60 AS money) AS Sales, 48 as
Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory,
'Accessories' as Subcategory, 'Carrying Case' as Product, CAST(10247.95 AS money) AS Sales, 41 as
Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as
Territory, 'Accessories' as Subcategory, 'Tripod' as Product, CAST(1200.00 AS money) AS Sales, 16 as
Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory,
'Accessories' as Subcategory,'Tripod' as Product, CAST(2025.00 AS money) AS Sales, 27 as Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory,
'Accessories' as Subcategory,'Tripod' as Product, CAST(1425.00 AS money) AS Sales, 19 as Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as
Territory, 'Accessories' as Subcategory,'Lens Adapter' as Product, CAST(887.50 AS money) AS Sales, 13 as
Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory,
'Accessories' as Subcategory, 'Lens Adapter' as Product, CAST(607.50 AS money) AS Sales, 9 as Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory,
'Accessories' as Subcategory,'Lens Adapter' as Product, CAST(1215.00 AS money) AS Sales, 18 as Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as
Territory, 'Digital' as Subcategory,'Compact Digital' as Product, CAST(10191.00 AS money) AS Sales, 79
as Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory,
'Digital' as Subcategory, 'Compact Digital' as Product, CAST(8772.00 AS money) AS Sales, 68 as Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory,
'Digital' as Subcategory, 'Compact Digital' as Product, CAST(10578.00 AS money) AS Sales, 82 as Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as
Territory, 'Digital' as Subcategory, 'Slim Digital' as Product, CAST(7218.10 AS money) AS Sales, 38 as
Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as
Territory, 'Digital' as Subcategory, 'Slim Digital' as Product, CAST(8357.80 AS money) AS Sales, 44 as
Quantity
UNION SELECT CAST('2009-01-05' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as
Territory, 'Digital' as Subcategory, 'Slim Digital' as Product, CAST(9307.55 AS money) AS Sales, 49 as
Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as
Territory, 'Digital' as Subcategory,'Compact Digital' as Product, CAST(3870.00 AS money) AS Sales, 30 as
Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory,
'Digital' as Subcategory,'Compact Digital' as Product, CAST(5805.00 AS money) AS Sales, 45 as Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory,
'Digital' as Subcategory, 'Compact Digital' as Product, CAST(8643.00 AS money) AS Sales, 67 as Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as
Territory, 'Digital' as Subcategory, 'Slim Digital' as Product, CAST(9877.40 AS money) AS Sales, 52 as
Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory,
'Digital' as Subcategory, 'Slim Digital' as Product, CAST(12536.70 AS money) AS Sales, 66 as Quantity
UNION SELECT CAST('2009-01-06' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory,
'Digital' as Subcategory, 'Slim Digital' as Product, CAST(6648.25 AS money) AS Sales, 35 as Quantity

```

7. Click the **Run** icon (!) to run the query.

The query results are the data available to display in your report.

The screenshot shows the Microsoft Query Designer window. At the top, there are buttons for 'Edit as Text' and 'Import...', and a dropdown menu 'Command type: Text'. Below the interface, a large text area displays a complex SQL query. Underneath the query, a data grid shows the results. The columns in the grid are: SalesDate, FullName, Territory, Subcategory, Product, Sales, and Quantity. The data grid contains 12 rows of sales information. At the bottom of the window are 'Help', 'OK', and 'Cancel' buttons.

SalesDate	FullName	Territory	Subcategory	Product	Sales	Quantity
1/5/2009 12:00:...	Fernando Ross	South	Accessories	Carrying Case	9248.1500	37
1/5/2009 12:00:...	Fernando Ross	South	Accessories	Lens Adapter	1417.5000	21
1/5/2009 12:00:...	Fernando Ross	South	Accessories	Tripod	1125.0000	15
1/5/2009 12:00:...	Fernando Ross	South	Digital	Compact Digital	10578.0000	82
1/5/2009 12:00:...	Fernando Ross	South	Digital	Slim Digital	9307.5500	49
1/5/2009 12:00:...	Lauren Johnson	Central	Accessories	Carrying Case	16996.6000	68
1/5/2009 12:00:...	Lauren Johnson	Central	Accessories	Lens Adapter	1147.5000	17
1/5/2009 12:00:...	Lauren Johnson	Central	Accessories	Tripod	1350.0000	18
1/5/2009 12:00:...	Lauren Johnson	Central	Digital	Compact Digital	10191.0000	79
1/5/2009 12:00:...	Lauren Johnson	Central	Digital	Slim Digital	7218.1000	38
1/5/2009 12:00:...	Warren Pal	North	Accessories	Carrying Case	13747.2500	55

8. Click **OK**.

2. Add and Configure a List

In Reporting Services the list data region is ideal for creating free-form reports. It's based on the *tablix* data region, as are tables and matrixes. For more information, see [Create Invoices and Forms with Lists](#).

You will use a list to display the sales information for sales territories in a report formatted like a newsletter. The information is grouped by territory. You will add a new row group that groups data by territory, and then delete the built-in Details row group.

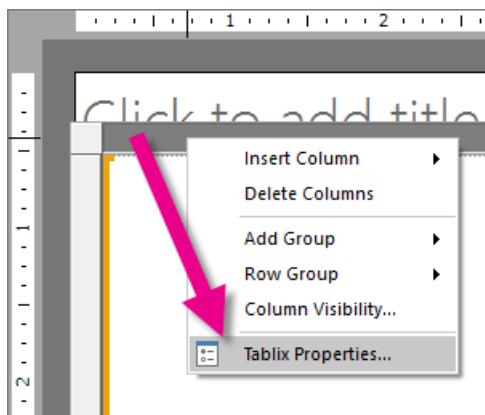
To add a list

1. On the **Insert** tab > **Data Regions** > **List**.
2. Click in the report body (between the title and footer areas) and drag to make the list box. Make the list box 7 inches tall and 6.25 inches wide. To get the exact size, in the **Properties** pane under **Position**, type values for the **Width** and **Height** properties.

NOTE

This report uses the paper size Letter (8.5 X11) and 1 inch margins. A list box taller than 9 inches or wider than 6.5 inches might generate blank pages.

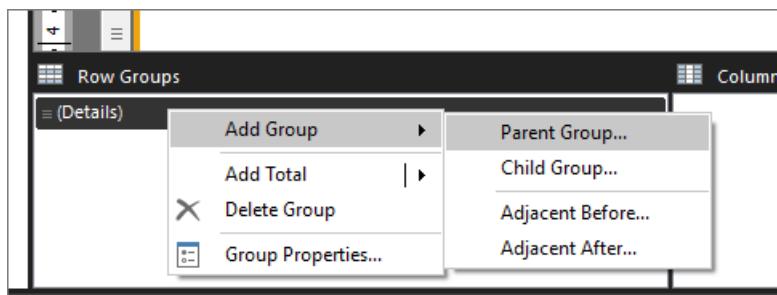
3. Click inside the list box, right-click the bar at the top of the list, and click **Tablix Properties**.



4. In the **Dataset name** drop-down list, select **ListDataset**.
 5. Click **OK**.
 6. Right-click inside the list, and then click **Rectangle Properties**.
 7. On the **General** tab, select the **Add a page break after** check box.
 8. Click **OK**.

To add a new row group and to delete the Details group

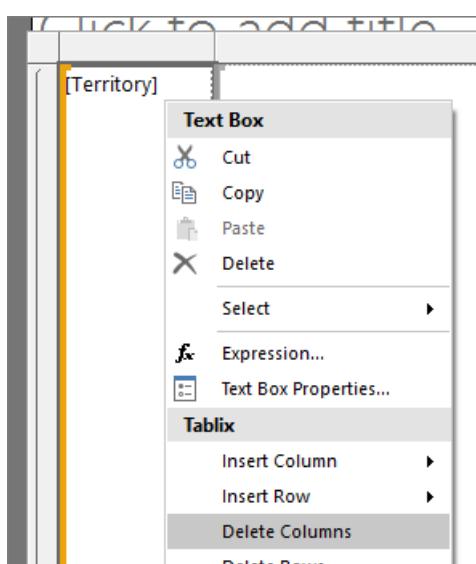
1. In the Row Groups pane, right-click the Details group, point to **Add Group**, and then click **Parent Group**.



2. In the **Group by** list, select **[Territory]**.
 3. Click **OK**.

A column containing the cell **[Territory]** is added to the list.

 4. Right-click the Territory column in the list, and then click **Delete Columns**.



- ## 5. Select **Delete Columns only**.

6. In the Row Groups pane, right-click the **Details** group > **Delete Group**.

7. Select **Delete Group only**.

8. Click **OK**.

3. Add Graphic elements

One advantage of list data regions is that you can add report items such as rectangles and text boxes anywhere, instead of being limited to a tabular layout. You will enhance the appearance of the report by adding a graphic (a rectangle filled with a color).

To add graphic elements to the report

1. On the **Insert** tab, select **Rectangle**.

2. Click in the upper left corner of the list and drag to make the rectangle 7 inches tall and 3.5 inches wide.

Again, to get the exact size, in the **Properties** pane under **Position**, type values for **Width** and **Height**.

3. Right-click the rectangle > **Rectangle Properties**.

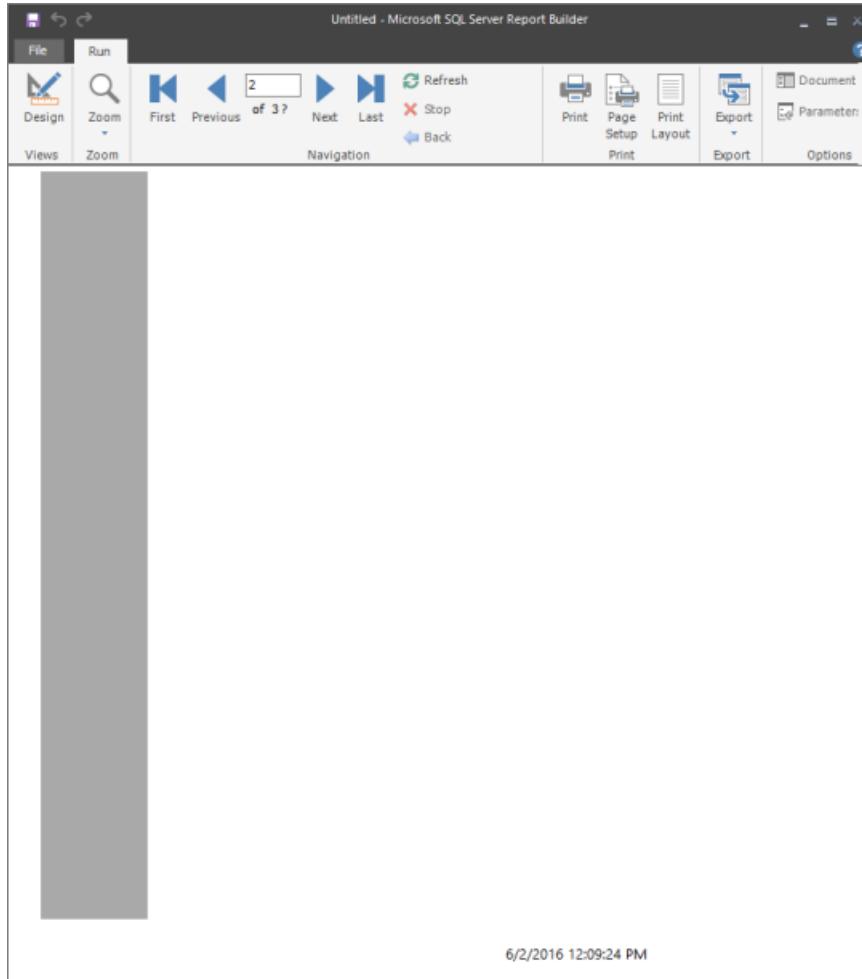
4. Click the **Fill** tab.

5. In **Fill color**, select **Light Gray**.

6. Click **OK**.

7. Click **Run** to preview the report.

The left side of the report now has vertical graphic that consists of a light gray rectangle, as shown in the following image.

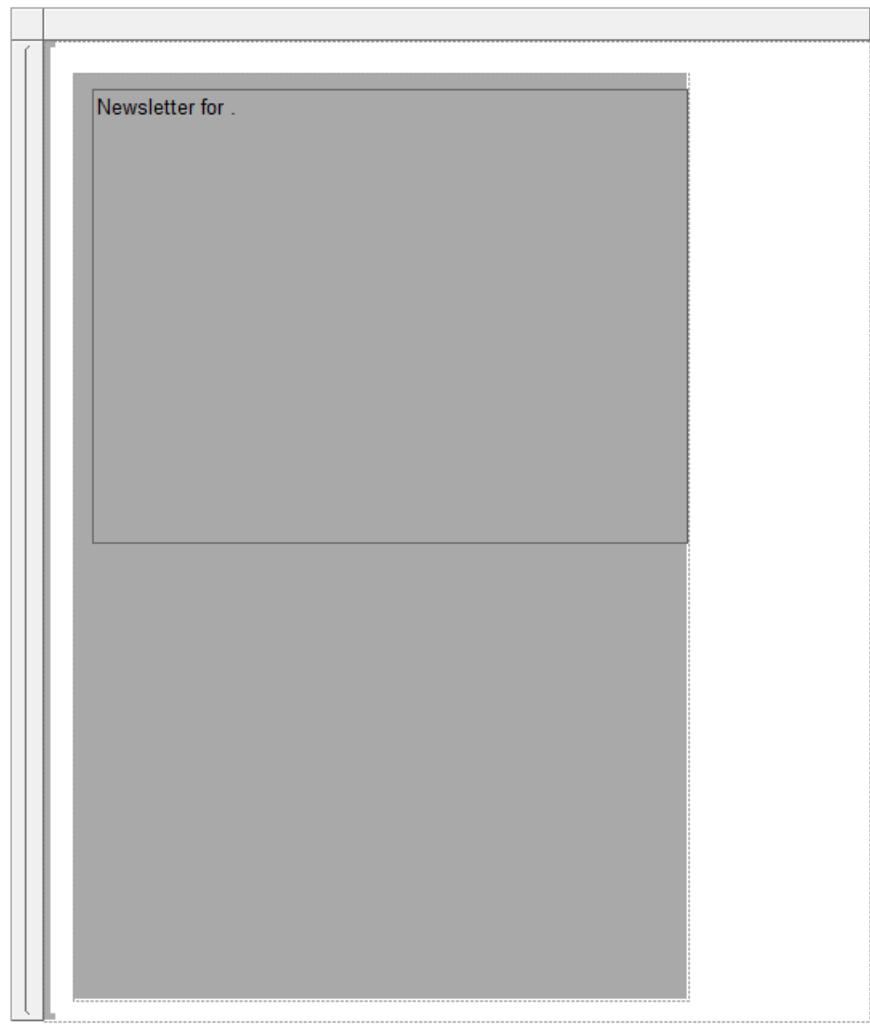


4. Add Free Form Text

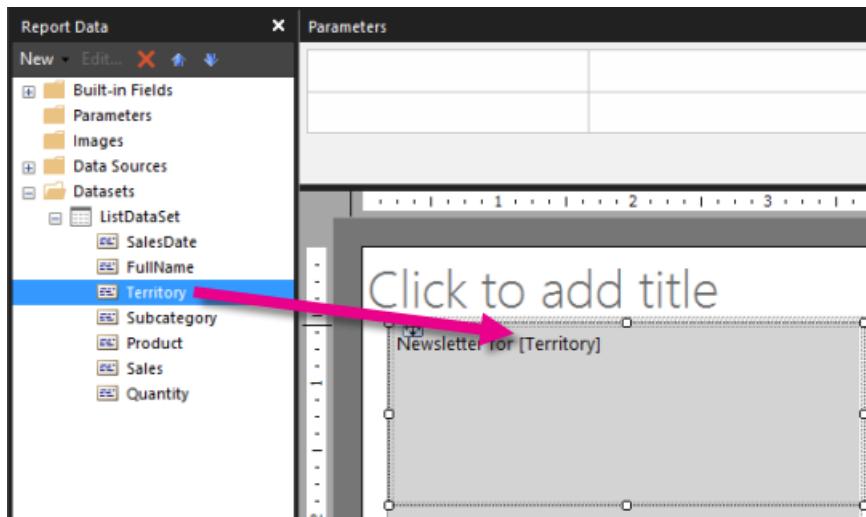
You can add text boxes to display static text that is repeated on each report page, as well as data fields.

To add text to the report

1. Click **Design** to return to design view.
2. On the **Insert** tab > **Text Box**. Click the upper left corner of the list, inside of the rectangle you added previously, and drag to make the text box about 3.45 inches wide and 5 inches tall.
3. With the cursor in the text box and type: **Newsletter for** . Include a space after the word "for", to separate the text from the field you will add in the next step.



4. Drag the **[Territory]** field from ListDataSet in the Report Data pane to the text box and place it after "Newsletter for ".



5. Select the text and the `[Territory]` field.
6. On the **Home** tab > **Font**, select:
 - **Segoe Semibold**.
 - **20 pt.**
 - **Tomato**.
7. Place the cursor below the text you typed in step 3 and type: **Hello** with a space after the word, to separate the text and the field that you will add in the next step.
8. Drag the `[FullName]` field from ListDataSet in the Report Data pane to the text box and place it after "Hello", then type a comma (,),
9. Select the text you added in the previous steps.
10. On the **Home** tab > **Font**, select:
 - **Segoe Semibold**.
 - **16 pt.**
 - **Black**.
11. Place the cursor below the text you added in steps 9 through 13, and then copy and paste the following meaningless text:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin sed dolor in ipsum pulvinar egestas. Sed sed lacus at leo ornare ultricies. Vivamus velit risus, euismod nec sodales gravida, gravida in dui. Etiam ullamcorper elit vitae justo fermentum ut ullamcorper augue sodales. Ut placerat, nisl quis feugiat adipiscing, nibh est aliquet est, mollis faucibus mauris lectus quis arcu. In mollis tincidunt lacinia. In vitae erat ut lorem tincidunt luctus. Curabitur et magna nunc, sit amet adipiscing nisi. Nulla rhoncus elementum orci nec tincidunt. Aliquam imperdiet cursus erat vel tincidunt. Donec et neque ac urna rutrum sodales. In id purus et nisl dignissim dapibus. Sed rhoncus metus at felis feugiat eu tempor dolor vehicula. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam faucibus consectetur diam eu pellentesque.
12. Select the text you just added.
13. On the **Home** tab > **Font**, select:
 - **Segoe UI**.
 - **10 pt.**
 - **Black**.
14. Place the cursor inside the text box, below the meaningless text and type: **Congratulations on your total**

sales of, with a space after the word to separate the text and the field you will add in the next step.

15. Drag the Sales field to the text box, place it after the text you typed in the previous step, then type an exclamation mark (!).
16. Select the text and the field you just added.
17. On the **Home** tab > **Font**, select:
 - **Segoe Semibold**.
 - **16 pt.**
 - **Black**.
18. Select just the **[Sales]** field, right-click the field > **Expression**.
19. In the **Expression** box, change the expression to include the Sum function as follows:

```
=Sum(Fields!Sales.value)
```

20. Click **OK**.



21. With **[Sum(Sales)]** still selected, on the **Home** tab > **Number** group > **Currency**.
22. Right-click the text box with the "Click to add title" text, and then click **Delete**.
23. Select the list box. Select the two double-headed arrows and move it to the top of the page.



24. Click **Run** to preview the report.

The report displays static text and each report page includes data that pertains to a territory. Sales are formatted as currency.

Newsletter for Central

Hello Lauren Johnson,

Loreum ipsum dolor sit amet, consectetur adipiscing elit. Proin sed dolor in ipsum pulvinar egestas. Sed sed lacus at leo ornare ultricies. Vivamus velit risus.

euismod nec sodales gravida, gravida in dui. Etiam ullamcorper elit vitae justus fermentum ut ullamcorper augue sodales.

Ut placerat, nisl quis feugiat adipiscing, nibh est aliquet est, mollis faucibus mauris lectus quis arcu. In mollis tincidunt lacinia. In vitae erat ut lorem tincidunt luctus. Curabitur et magna nunc, sit amet adipiscing nisl. Nulla rhoncus elementum orci nec tincidunt.

Aliquam imperdiet cursus erat vel tincidunt. Donec et neque ac urna rutrum sodales. In id purus et nisl dignissim dapibus. Sed rhoncus metus at felis feugiat eu tempor dolor vehicula. Loreum ipsum dolor sit amet, consectetur adipiscing elit. Nullam faucibus consectetur diam eu pellentesque.

Congratulations on your total sales of \$66,235.40!

6/2/2016 3:37:38 PM

5. Add a Table to Show Sales Details

Use the New Table and Matrix Wizard to add a table to the free form report. After you complete the wizard, you will manually add a row for totals.

To add a table

1. On the **Insert** tab > **Data Regions** area > **Table** > **Table Wizard**.
2. On the **Choose a dataset** page, click **ListDataset** > **Next**.
3. On the **Arrange fields** page, drag the Product field from **Available fields** to **Values**.
4. Repeat step 3 for SalesDate, Quantity, and Sales. Place SalesDate below Product, Quantity below SalesDate, and Sales below SalesDate.
5. Click **Next**.
6. On the **Choose the layout** page, view the layout of the table.

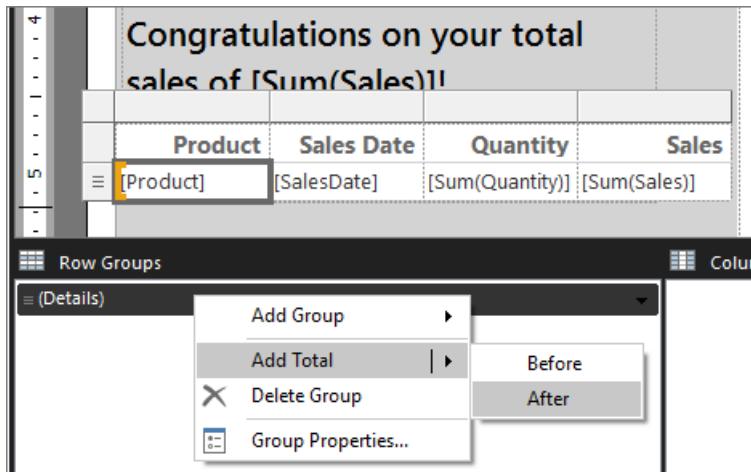
The table is simple: five columns with no row or column groups. Because it has no groups, the layout options related to groups, are not available. You will manually update the table to include a total later in the tutorial.

7. Click **Next**.
8. Click **Finish**.
9. Drag the table to below the text box that you added in lesson 4.

NOTE

Make sure the table is inside the list box and inside the gray rectangle.

10. With the table selected, in the **Row Group** pane right-click **Details** > **Add Total** > **After**.



11. Select the cell in the Product column and type **Total**.

	Product	Sales Date	Quantity	Sales
≡	[Product]	[SalesDate]	[Sum(Quantity)]	[Sum(Sales)]
	Total		[Sum(Quantity)]	[Sum(Sales)]

12. Select the [SalesDate] field. On the **Home** tab > **Number**, change **Default** to **Date**.

13. Select the [Sum(Sales)] fields. On the **Home** tab > **Number**, change **Default** to **Currency**.

Click **Run** to preview the report.

The report displays a table with sales details and totals.

Newsletter for Central

Hello Lauren Johnson,

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin sed dolor in ipsum pulvinar egestas. Sed sed lacus at leo ornare ultricies. Vivamus velit risus, euismod nec sodales gravida, gravida in dui. Etiam ullamcorper elit vitae justo fermentum ut ullamcorper augue sodales.
 Ut placerat, nisl quis feugiat adipiscing, nibh est aliquet est, mollis faucibus mauris lectus quis arcu. In mollis tincidunt lacinia. In vitae erat ut lorem tincidunt luctus. Curabitur et magna nunc, sit amet adipiscing nisi. Nulla rhoncus elementum orci nec tincidunt. Aliquam imperdiet cursus erat vel tincidunt. Donec et neque ac urna rutrum sodales. In id purus et nisl dignissim dapibus. Sed rhoncus metus at felis feugiat eu tempor dolor vehicula. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam faucibus consectetur diam eu pellentesque.

Congratulations on your total sales of \$66,235.40!

Product	Sales Date	Quantity	Sales
Carrying Case	1/5/2009	68	\$16,996.60
Lens Adapter	1/5/2009	17	\$1,147.50
Tripod	1/5/2009	18	\$1,350.00
Compact Digital	1/5/2009	79	\$10,191.00
Slim Digital	1/5/2009	38	\$7,218.10
Carrying Case	1/6/2009	54	\$13,497.30
Lens Adapter	1/6/2009	13	\$887.50
Tripod	1/6/2009	16	\$1,200.00
Compact Digital	1/6/2009	30	\$3,870.00
Slim Digital	1/6/2009	52	\$9,877.40
Total		385	\$66,235.40

6/2/2016 4:42:44 PM

6. Save the Report

You can save reports to a report server, SharePoint library, or your computer.

In this tutorial, save the report to a report server. If you do not have access to a report server, save the report to your computer.

To save the report on a report server

1. From the **Report Builder** button, click **Save As**.
2. Click **Recent Sites and Servers**.
3. Select or type the name of the report server where you have permission to save reports.

The message "Connecting to report server" appears. When the connection is complete, you see the contents of the report folder that the report server administrator specified as the default location for reports.

4. In **Name**, replace the default name with **SalesInformationByTerritory**.
5. Click **Save**.

The report is saved to the report server. The name of report server that you are connected to appears in the status bar at the bottom of the window.

To save the report on your computer

1. From the **Report Builder** button, click **Save As**.
2. Click **Desktop**, **My Documents**, or **My computer**, and then browse to the folder where you want to save the report.

3. In **Name**, replace the default name with **SalesInformationByTerritory**.

4. Click **Save**.

7. (Optional) Add a Line to Separate Areas of the Report

Add a line to separate the editorial and details areas of the report.

To add a line

1. Click **Design** to return to design view.
2. On the **Insert** tab > **Report Items** > **Line**.
3. Draw a line below the text box you added in lesson 4.
4. Click the line, and on the **Home** tab > **Border**, select:
 - **Width** select **3 pt**.
 - **Color** select **Tomato**.

8. (Optional) Add Summary Data Visualizations

Rectangles help you control how the report renders. Place a pie and column chart inside a rectangle to ensure that the report renders the way you want.

To add a rectangle

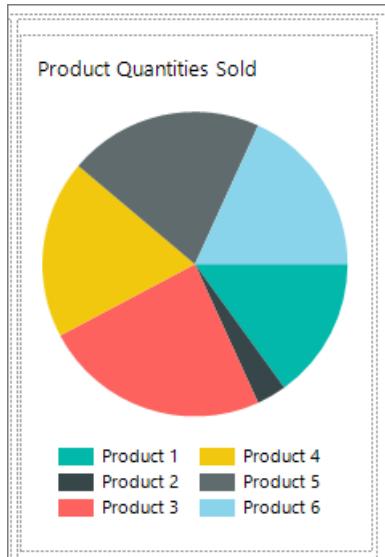
1. Click **Design** to return to design view.
2. On the **Insert** tab > **Report Items** > **Rectangle**. Drag the rectangle inside the list box to the right of the table to make a rectangle about 2.25 inches wide and 7.9 inches tall.
3. With the new rectangle selected, in the Properties pane, make **BorderColor LightGrey**, **BorderStyle Solid**, and **BorderWidth 2 pt**.
4. Align the tops of the rectangle and the table.

To add a pie chart

1. On the **Insert** tab > **Data Visualizations** > **Chart** > **Chart Wizard**.
2. On the **Choose a dataset** page, click **ListDataset** > **Next**.
3. Click **Pie** > **Next**.
4. On the arrange chart fields page, drag Product to **Categories**.
5. Drag Quantity to **Values**, then click **Next**.
6. Click **Finish**.
7. Resize the chart that appears in the upper left corner of the report to be about 2.25 inches wide and 3.6 inches tall.
8. Drag the chart inside the rectangle.
9. Select the chart title and type: **Product Quantities Sold**.
10. On the **Home** tab > **Font**, make the title:
 - **Font Segoe UI Semibold**.
 - **Size 12 pt**.

- **Color Black.**

11. Right-click the legend > **Legend Properties**.
12. On the **General** tab, under **Legend position**, select the center dot at the bottom.
13. Click **OK**.
14. Drag to make the chart region taller, if necessary.



To add a column chart

1. On the **Insert** tab > **Data Visualizations** > **Chart**, > **Chart Wizard**.
2. On the **Choose a dataset** page, click **ListDataset**, then click **Next**.
3. Click **Column**, then click **Next**.
4. On the **Arrange chart fields** page, drag the Product field to **Categories**.
5. Drag Sales to **Values**, and then click **Next**.

Values display on the vertical axis.

6. Click **Finish**.

A column chart is added to the upper left corner of the report.

7. Resize the chart to be about 2.25 inches wide and almost 4 inches tall.
8. Drag the chart inside the rectangle, below the pie chart.
9. Select the chart title and type: **Product Sales**.

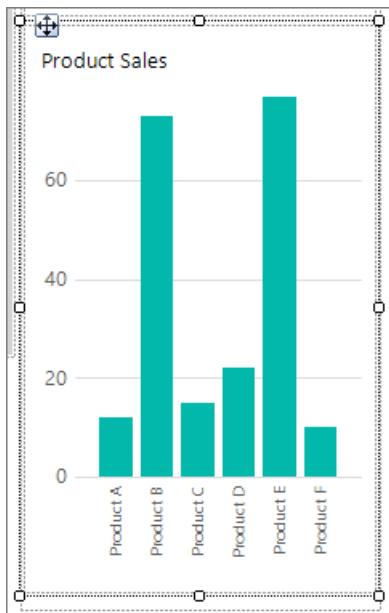
10. On the **Home** tab > **Font**, make the title:

- **Font Segoe UI Semibold**.
- **Size 12 pt**.
- **Color Black**.

11. Right click the legend, and then click **Delete Legend**.

NOTE

Removing the legend makes the chart more readable when the chart is small.



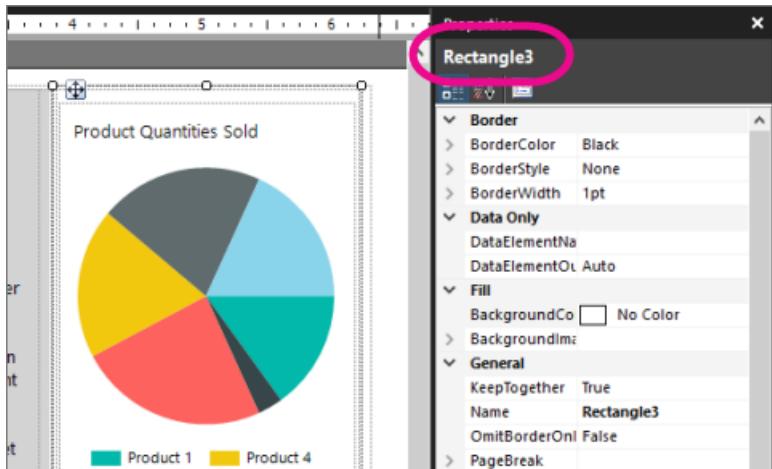
12. Select the chart axis, and on the **Home** tab > **Number** > **Currency**.
13. Select **Decrease Decimal** two times, so the number shows just dollars and no cents.

To verify the charts are inside the rectangle

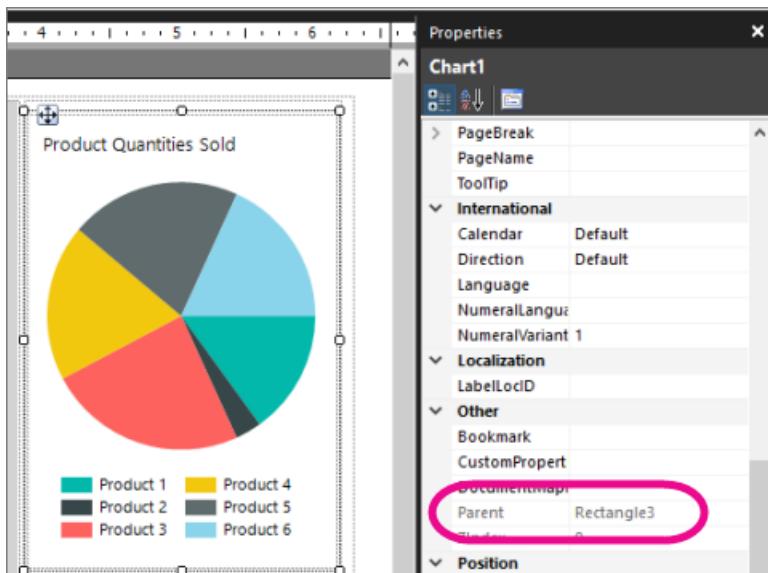
You can use rectangles as containers for other items on a report page. Read more about [rectangles as containers](#).

1. Select the rectangle you created and added the charts to, earlier in this lesson.

In the Properties pane, the **Name** property displays the name of the rectangle.



2. Click the pie chart.
3. In the **Properties** pane, verify that the **Parent** property contains the name of the rectangle.



- Click the column chart and repeat step 3.

NOTE

If the charts are not inside the rectangle, the rendered report does not display the charts together.

To make the charts the same size

- Select the pie chart, press the Ctrl key, and then select the column chart.
- With both charts selected, right-click > **Layout** > **Make Same Width**.

NOTE

The item you click first determines the width of all the selected items.

- Click **Run** to preview the report.

The report now displays summary sales data in pie and column charts.

Next Steps

This concludes the tutorial for how to create a free-form report.

For more information about lists, see:

- [Tables, Matrices, and Lists \(Report Builder and SSRS\)](#)
- [Create Invoices and Forms with Lists](#)
- [Tablix Data Region Cells, Rows, and Columns \(Report Builder\) and SSRS.](#)

For more information about query designers, see [Query Designers \(Report Builder\)](#) and [Text-based Query Designer User Interface \(Report Builder\)](#).

See Also

[Report Builder Tutorials](#)

Tutorial: Format Text (Report Builder)

11/30/2018 • 14 minutes to read • [Edit Online](#)

In this tutorial, you practice formatting text in various ways in a Reporting Services paginated report. You can experiment with different formats.

After you set up the blank report with the data source and dataset, you can pick the formats you want to explore. The following illustration shows a report similar to the one you will create.

Territory	Link Text	Product	Sales
Central	Install Report Builder	Carrying Case	\$30,494
		Compact	\$14,061
		Digital	
		Lens Adapter	\$2,035
		Slim Digital	\$17,096
		Tripod	\$2,550
		Total	\$66,235
North	Report Builder in SQL Server	Carrying Case	\$25,745
		Compact	\$14,577
		Digital	
		Lens Adapter	\$1,350
		Slim Digital	\$20,895
		Tripod	\$3,825
		Total	\$66,391
South	What is New in Reporting Services (SSRS)	Carrying Case	\$19,496
		Compact	\$19,221
		Digital	
		Lens Adapter	\$2,633
		Slim Digital	\$15,956
		Tripod	\$2,550
		Total	\$59,855
Total			\$192,482

In one step, you make a mistake on purpose so you can see why it is a mistake. Then you correct the mistake to achieve the desired effect.

Estimated time to complete this tutorial: 20 minutes.

Requirements

For information about requirements, see [Prerequisites for Tutorials \(Report Builder\)](#).

Create a Blank Report with a Data Source and Dataset

To create a blank report

1. [Start Report Builder](#) either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the left pane of the **Getting Started** dialog box, verify that **New Report** is selected.
3. In the right pane, click **Blank Report**.

To create a data source

1. In the Report Data pane, click **New > Data Source**.

If you don't see the **Report Data** pane, on the **View** tab, check **Report Data**.

2. In the **Name** box, type: **TextDataSource**
3. Click **Use a connection embedded in my report**.
4. Verify that the connection type is Microsoft SQL Server, and then in the **Connection string** box type:

```
Data Source = <servername>
```

NOTE

The expression `<servername>`, for example Report001, specifies a computer on which an instance of the SQL Server Database Engine is installed. This tutorial does not need specific data; it just needs a connection to a SQL Server database. If you already have a data source connection listed under **Data Source Connections**, you can select it and go to the next procedure, "To create a dataset." For more information, see [Alternative Ways to Get a Data Connection \(Report Builder\)](#).

5. Click **OK**.

To create a dataset

1. In the Report Data pane, click **New > Dataset**.
2. Verify that the data source is **TextDataSource**.
3. In the **Name** box, type: **TextDataset**.
4. Verify that the **Text** query type is selected, and then click **Query Designer**.
5. Click **Edit as Text**.
6. Paste the following query into the query pane:

NOTE

In this tutorial, the query already contains the data values, so that it does not need an external data source. This makes the query quite long. In a business environment, a query would not contain the data. This is for learning purposes only.

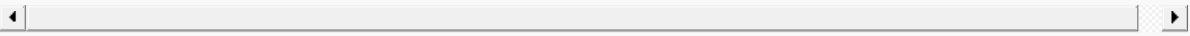
```
SELECT CAST('2015-01-05' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as Territory, 'Accessories' as Subcategory,'Carrying Case' as Product, CAST(16996.60 AS money) AS Sales, 68 as Quantity, 'Install Report Builder' as LinkText, 'https://go.microsoft.com/fwlink/?LinkId=154882' AS URL UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory, 'Accessories' as Subcategory, 'Carrying Case' as Product, CAST(13747.25 AS money) AS Sales, 55 as Quantity, 'Report Builder in SQL Server' as Link, 'https://go.microsoft.com/fwlink/?LinkId=160556' AS URL UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory, 'Accessories' as Subcategory,'Carrying Case' as Product, CAST(9248.15 AS money) AS Sales, 37 as Quantity, 'What is New in Reporting Services (SSRS)' as Link, 'https://go.microsoft.com/fwlink/?LinkId=165064' AS URL
```

```
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as Territory, 'Accessories' as Subcategory,'Tripod' as Product, CAST(1350.00 AS money) AS Sales, 18 as Quantity, 'Install Report Builder' as LinkText, 'https://go.microsoft.com/fwlink/?LinkId=154882' AS URL
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory, 'Accessories' as Subcategory,'Tripod' as Product, CAST(1800.00 AS money) AS Sales, 24 as Quantity, 'Report Builder in SQL Server' as Link, 'https://go.microsoft.com/fwlink/?LinkId=160556' AS URL
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory, 'Accessories' as Subcategory,'Tripod' as Product, CAST(1125.00 AS money) AS Sales, 15 as Quantity, 'What is New in Reporting Services (SSRS)' as Link, 'https://go.microsoft.com/fwlink/?LinkId=165064' AS URL
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as Territory, 'Accessories' as Subcategory,'Lens Adapter' as Product, CAST(1147.50 AS money) AS Sales, 17 as Quantity, 'Install Report Builder' as LinkText, 'https://go.microsoft.com/fwlink/?LinkId=154882' AS URL
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory, 'Accessories' as Subcategory, 'Lens Adapter' as Product, CAST(742.50 AS money) AS Sales, 11 as Quantity, 'Report Builder in SQL Server' as Link, 'https://go.microsoft.com/fwlink/?LinkId=160556' AS URL
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory, 'Accessories' as Subcategory,'Lens Adapter' as Product, CAST(1417.50 AS money) AS Sales, 21 as Quantity, 'What is New in Reporting Services (SSRS)' as Link, 'https://go.microsoft.com/fwlink/?LinkId=165064' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as Territory, 'Accessories' as Subcategory, 'Carrying Case' as Product, CAST(13497.30 AS money) AS Sales, 54 as Quantity, 'Install Report Builder' as LinkText, 'https://go.microsoft.com/fwlink/?LinkId=154882' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory, 'Accessories' as Subcategory, 'Carrying Case' as Product, CAST(11997.60 AS money) AS Sales, 48 as Quantity, 'Report Builder in SQL Server' as Link, 'https://go.microsoft.com/fwlink/?LinkId=160556' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory, 'Accessories' as Subcategory, 'Carrying Case' as Product, CAST(10247.95 AS money) AS Sales, 41 as Quantity, 'What is New in Reporting Services (SSRS)' as Link, 'https://go.microsoft.com/fwlink/?LinkId=165064' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as Territory, 'Accessories' as Subcategory, 'Tripod' as Product, CAST(1200.00 AS money) AS Sales, 16 as Quantity, 'Install Report Builder' as LinkText, 'https://go.microsoft.com/fwlink/?LinkId=154882' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory, 'Accessories' as Subcategory,'Tripod' as Product, CAST(2025.00 AS money) AS Sales, 27 as Quantity, 'Report Builder in SQL Server' as Link, 'https://go.microsoft.com/fwlink/?LinkId=160556' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory, 'Accessories' as Subcategory,'Tripod' as Product, CAST(1425.00 AS money) AS Sales, 19 as Quantity, 'What is New in Reporting Services (SSRS)' as Link, 'https://go.microsoft.com/fwlink/?LinkId=165064' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as Territory, 'Accessories' as Subcategory,'Lens Adapter' as Product, CAST(887.50 AS money) AS Sales, 13 as Quantity, 'Install Report Builder' as LinkText, 'https://go.microsoft.com/fwlink/?LinkId=154882' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory, 'Accessories' as Subcategory, 'Lens Adapter' as Product, CAST(607.50 AS money) AS Sales, 9 as Quantity, 'Report Builder in SQL Server' as Link, 'https://go.microsoft.com/fwlink/?LinkId=160556' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory, 'Accessories' as Subcategory,'Lens Adapter' as Product, CAST(1215.00 AS money) AS Sales, 18 as Quantity, 'What is New in Reporting Services (SSRS)' as Link, 'https://go.microsoft.com/fwlink/?LinkId=165064' AS URL
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as Territory, 'Digital' as Subcategory,'Compact Digital' as Product, CAST(10191.00 AS money) AS Sales, 79 as Quantity, 'Install Report Builder' as LinkText, 'https://go.microsoft.com/fwlink/?LinkId=154882' AS URL
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory, 'Digital' as Subcategory, 'Compact Digital' as Product, CAST(8772.00 AS money) AS Sales, 68 as Quantity, 'Report Builder in SQL Server' as Link, 'https://go.microsoft.com/fwlink/?LinkId=160556' AS URL
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory, 'Digital' as Subcategory, 'Compact Digital' as Product, CAST(10578.00 AS money) AS Sales, 82 as Quantity, 'What is New in Reporting Services (SSRS)' as Link, 'https://go.microsoft.com/fwlink/?LinkId=165064' AS URL
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as Territory, 'Digital' as Subcategory, 'Slim Digital' as Product, CAST(7218.10 AS money) AS Sales, 38 as Quantity, 'Install Report Builder' as LinkText, 'https://go.microsoft.com/fwlink/?LinkId=154882' AS URL
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory, 'Digital' as Subcategory, 'Slim Digital' as Product, CAST(8357.80 AS money) AS Sales, 44 as Quantity, 'Report Builder in SQL Server' as Link, 'https://go.microsoft.com/fwlink/?LinkId=160556' AS
```

```

URL
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory,'Digital' as Subcategory,'Slim Digital' as Product, CAST(9307.55 AS money) AS Sales, 49 as Quantity, 'What is New in Reporting Services (SSRS)' as Link, 'https://go.microsoft.com/fwlink/?LinkId=165064' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as Territory, 'Digital' as Subcategory,'Compact Digital' as Product, CAST(3870.00 AS money) AS Sales, 30 as Quantity, 'Install Report Builder' as LinkText, 'https://go.microsoft.com/fwlink/?LinkId=154882' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory, 'Digital' as Subcategory,'Compact Digital' as Product, CAST(5805.00 AS money) AS Sales, 45 as Quantity, 'Report Builder in SQL Server' as Link, 'https://go.microsoft.com/fwlink/?LinkId=160556' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory, 'Digital' as Subcategory, 'Compact Digital' as Product, CAST(8643.00 AS money) AS Sales, 67 as Quantity, 'What is New in Reporting Services (SSRS)' as Link, 'https://go.microsoft.com/fwlink/?LinkId=165064' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Lauren Johnson' as FullName,'Central' as Territory, 'Digital' as Subcategory, 'Slim Digital' as Product, CAST(9877.40 AS money) AS Sales, 52 as Quantity, 'Install Report Builder' as LinkText, 'https://go.microsoft.com/fwlink/?LinkId=154882' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Warren Pal' as FullName,'North' as Territory, 'Digital' as Subcategory, 'Slim Digital' as Product, CAST(12536.70 AS money) AS Sales, 66 as Quantity, 'Report Builder in SQL Server' as Link, 'https://go.microsoft.com/fwlink/?LinkId=160556' AS URL
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Fernando Ross' as FullName,'South' as Territory, 'Digital' as Subcategory, 'Slim Digital' as Product, CAST(6648.25 AS money) AS Sales, 35 as Quantity, 'What is New in Reporting Services (SSRS)' as Link, 'https://go.microsoft.com/fwlink/?LinkId=165064' AS URL

```



7. Click Run (!) to run the query.

The query results are the data available to display in your report.

8. Click **OK**.

9. Click **OK**.

Add a Field to the Report Design Surface

If you want a field from your dataset to appear in a report, your first impulse may be to drag it directly to the design surface. This exercise points out why that doesn't work and what to do instead.

To add a field to the report (and get the wrong result)

1. Drag the **FullName** field from the Report Data pane to the design surface.

Report Builder creates a text box with an expression in it, represented as `<Expr>`.

2. Click **Run**.

You only see one record, **Fernando Ross**, which is alphabetically the first record in the query. The field does not repeat to show the other records in that field.

3. Click **Design** to return to design view.

4. Select the expression `<Expr>` in the text box.

5. In the Properties pane, for the **Value** property, you see the following (if you don't see the Properties pane, on the **View** tab, check **Properties**):

```
=First(Fields!FullName.Value, "TextDataSet")
```

The `First` function is designed to retrieve only the first value in a field, and that is what it has done.

Dragging the field directly to the design surface created a text box. Text boxes by themselves are not data

regions, so they do not display data from a report dataset. Text boxes in data regions, such as tables, matrices, and lists, do display data.

6. Select the text box (if you have the expression selected, press ESC to select the text box), and press the DELETE key.

To add a field to the report (and get the right result)

1. On the **Insert** tab of the ribbon, in the **Data Regions** area, click **List**. Click the design surface, and then drag to create a box that about two inches wide and one inch tall.
2. Drag the **FullName** field from the Report Data pane to the list box.

This time Report Builder creates a text box with the expression `[FullName]` in it.

3. Click **Run**.

Note that this time the box repeats to show all the records in the query.

4. Click **Design** to return to design view.
5. Select the expression in the text box.
6. In the Properties pane, for the **Value** property, you see the following:

```
=Fields!FullName.Value
```

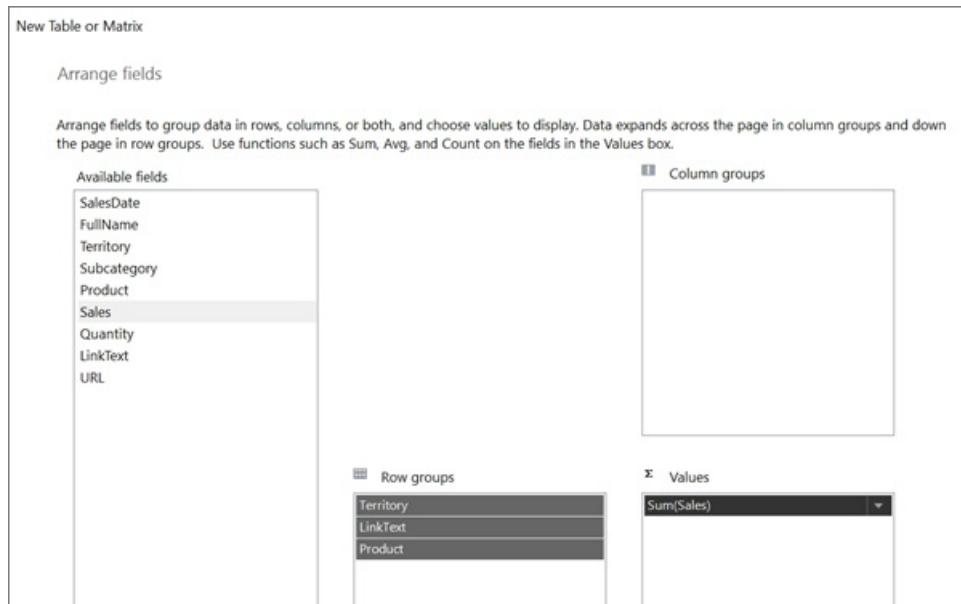
By dragging the text box to the list data region, you display the data that is in that field in the dataset.

7. Select the list box and press the DELETE key.

Add a Table to the Report Design Surface

Create this table so you'll have a place to put hyperlinks and rotated text.

1. On the **Insert** tab > **Table** > **Table Wizard**.
2. On the **Choose a dataset** page of the New Table or Matrix wizard, click **Choose an existing dataset in this report or a shared dataset** > **TextDataset (in this Report)** > **Next**.
3. On the **Arrange fields** page, drag the **Territory**, **LinkText**, and **Product** fields to **Row groups**, drag the **Sales** field to **Values**, then click **Next**.



4. On the **Choose the layout** page, clear the **Expand/collapse groups** check box so you can see the whole table, then click **Next**.

5. Click **Finish**.

6. Click **Run**.

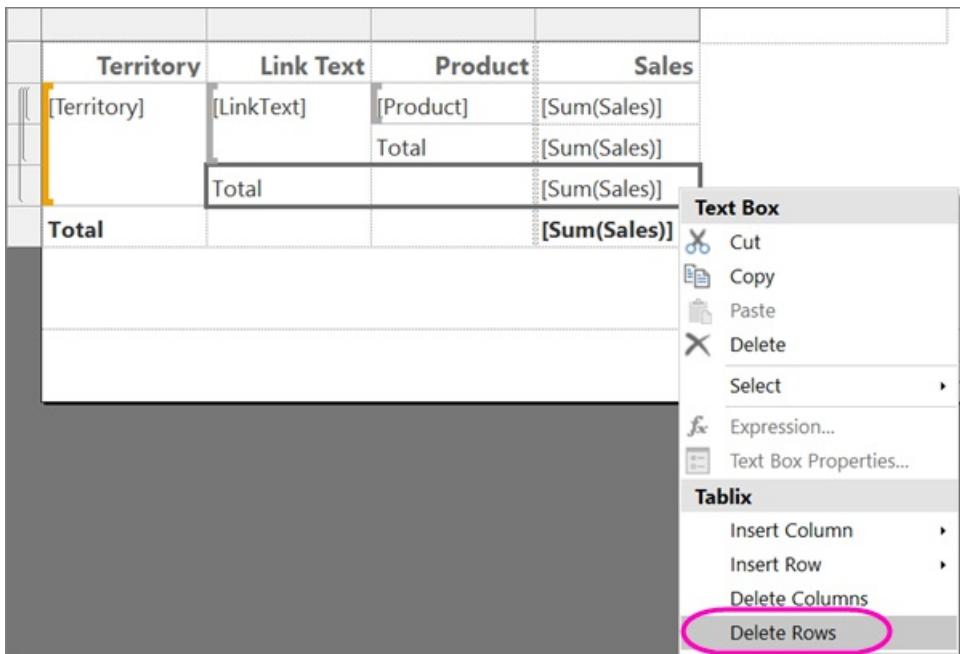
The table looks OK, but it has two Total rows. The **LinkText** column doesn't need a Total row.

Territory	Link Text	Product	Sales
Central	Install Report	Carrying Case	30493.9000
	Builder	Compact	14061.0000
		Digital	
		Lens Adapter	2035.0000
		Slim Digital	17095.5000
		Tripod	2550.0000
		Total	66235.4000
North	Report Builder in SQL Server	Carrying Case	25744.8500
		Compact	14577.0000
		Digital	
		Lens Adapter	1350.0000
		Slim Digital	20894.5000
		Tripod	3825.0000
		Total	66391.3500
South	What is New in Reporting Services (SSRS)	Carrying Case	19496.1000
		Compact	19221.0000
		Digital	
		Lens Adapter	2632.5000
		Slim Digital	15955.8000
		Tripod	2550.0000
		Total	59855.4000
Total			192482.1500

7. Click **Design** to return to design view.

8. Select the **Total** cell in the **LinkText** column, then hold down the SHIFT key and select the two cells to its right: and the empty cell in the **Product** column and the **[Sum(Sales)]** cell in the **Sales** column.

9. With those three cells selected, right-click one of those cells and click **Delete Rows**.



10. Click **Run**.

Now it has only one Total row.

Territory	Link Text	Product	Sales
Central	Install Report Builder	Carrying Case	30493.9000
		Compact	14061.0000
		Digital	
		Lens Adapter	2035.0000
		Slim Digital	17095.5000
		Tripod	2550.0000
		Total	66235.4000
North	Report Builder in SQL Server	Carrying Case	25744.8500
		Compact	14577.0000
		Digital	
		Lens Adapter	1350.0000
		Slim Digital	20894.5000
		Tripod	3825.0000
		Total	66391.3500
South	What is New in Reporting Services (SSRS)	Carrying Case	19496.1000
		Compact	19221.0000
		Digital	
		Lens Adapter	2632.5000
		Slim Digital	15955.8000
		Tripod	2550.0000
		Total	59855.4000
Total			192482.1500

Add a Hyperlink to the Report

In this section, you add a hyperlink to text in the table from the previous section.

1. Click **Design** to return to design view.

2. Right-click in the cell containing **[LinkText]**, and click **Text Box Properties**.
3. On the **Action** tab, click **Go to URL**.
4. In the **Select URL** box, click **[URL]**, then click **OK**.
5. Note that the text does not look any different. You need to make it look like link text.
6. Select **[LinkText]**.
7. On the **Home** tab > **Font**, select **Underline**, and change **Color** to **Blue**.
8. Click **Run**.

The text now looks like a link.

Territory	Link Text	Product	Sales
Central	Install Report Builder	Carrying Case	30493.9000
		Compact	14061.0000
		Digital	
		Lens Adapter	2035.0000
		Slim Digital	17095.5000
		Tripod	2550.0000
		Total	66235.4000
North	Report Builder in SQL Server	Carrying Case	25744.8500
		Compact	14577.0000
		Digital	
		Lens Adapter	1350.0000
		Slim Digital	20894.5000
		Tripod	3825.0000
		Total	66391.3500
South	What is New in Reporting Services (SSRS)	Carrying Case	19496.1000
		Compact	19221.0000
		Digital	
		Lens Adapter	2632.5000
		Slim Digital	15955.8000
		Tripod	2550.0000
		Total	59855.4000
Total			192482.1500

9. Click a link. If the computer is connected to the Internet, a browser will open to a Report Builder Help topic.

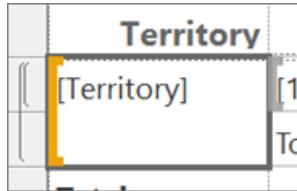
Rotate Text in the Report

In this section, you rotate some of the text in the table from the previous sections.

1. Click **Design** to return to design view.
2. Click in the cell containing **[Territory]**.
3. On the **Home** tab in the **Font** section, click the **Bold** button.
4. If the Properties pane is not open, on the **View** tab, select the **Properties** check box.
5. Locate the **WritingMode** property in the Properties pane, and change it from **Default** to **Rotate270**.

NOTE

When the properties in the Properties pane are organized into categories, WritingMode is in the **Localization** category. Be sure you have selected the cell and not the text. WritingMode is a property of the text box, not of the text.



6. On the **Home** tab > **Paragraph** section, select **Middle** and **Center** to locate the text in the center of the cell both vertically and horizontally.
7. Click Run (!).

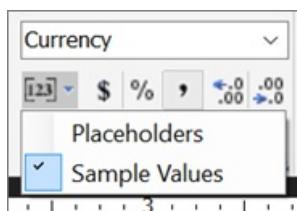
Now the text in the `[Territory]` cell runs vertically from the bottom to the top of the cells.

Territory	Link Text	Product	Sales
Central	Install Report Builder	Carrying Case	30493.9000
		Compact	14061.0000
		Digital	
		Lens Adapter	2035.0000
		Slim Digital	17095.5000
		Tripod	2550.0000
		Total	66235.4000
North	Report Builder in SQL Server	Carrying Case	25744.8500
		Compact	14577.0000
		Digital	
		Lens Adapter	1350.0000
		Slim Digital	20894.5000
		Tripod	3825.0000
		Total	66391.3500
South	What is New in Reporting Services (SSRS)	Carrying Case	19496.1000
		Compact	19221.0000
		Digital	
		Lens Adapter	2632.5000
		Slim Digital	15955.8000
		Tripod	2550.0000
		Total	59855.4000
Total			192482.1500

Format Currency

1. Click **Design** to switch to design view.
2. Click the top table cell that contains `[Sum(Sales)]`, hold down the SHIFT key, and click the bottom table cell that contains `[Sum(Sales)]`.
3. On the **Home** tab > **Number** group > **Currency** button.

4. (Optional) If your regional setting is English (United States), the default sample text is **[\$12,345.00]**. If you do not see an example currency value, in the **Numbers** group, click **Placeholder Styles > Sample Values**.



5. (Optional) On the **Home** tab, in the **Number** group, click the **Decrease Decimals** button twice to display dollar figures with no cents.
6. Click **Run (!)** to preview the report.

The report now displays formatted data and is easier to read.

Territory	Link Text	Product	Sales
Central	Install Report Builder	Carrying Case	\$30,494
		Compact	\$14,061
		Digital	
		Lens Adapter	\$2,035
		Slim Digital	\$17,096
		Tripod	\$2,550
		Total	\$66,235
North	Report Builder in SQL Server	Carrying Case	\$25,745
		Compact	\$14,577
		Digital	
		Lens Adapter	\$1,350
		Slim Digital	\$20,895
		Tripod	\$3,825
		Total	\$66,391
South	What is New in Reporting Services (SSRS)	Carrying Case	\$19,496
		Compact	\$19,221
		Digital	
		Lens Adapter	\$2,633
		Slim Digital	\$15,956
		Tripod	\$2,550
		Total	\$59,855
Total			\$192,482

Displaying Text with HTML Formatting

1. Click **Design** to switch to design view.
2. On the **Insert** tab, click **Text Box**, and then on the design surface, click and drag to create a text box under the table, about four inches wide and three inches tall.
3. Copy this text and paste it into the text box:

```

<h4>Limitations of cascading style sheet attributes</h4>
<p>Only a basic set of <b>cascading style sheet (CSS)</b> attributes are defined:</p>
<ul><li>
    text-align, text-indent
</li><li>
    font-family, font-size
</li><li>
    color
</li><li>
    padding, padding-bottom, padding-top, padding-right, padding-left
</li><li>
    font-weight
</li></ul>

```

4. Drag the lower edge of the text box so all the text fits. You notice the design surface gets larger as you drag.
5. Select all of the text in the text box.
6. Right-click all of the selected text and click **Text Properties**.

This is a property of the text, not the text box, so in one text box you could have a mixture of plain text and text that uses HTML tags as styles.

7. On the **General** tab, under **Markup type**, click **HTML - Interpret HTML tags as styles**.
8. Click **OK**.
9. Click Run (!) to preview the report.

The text in the text box is displayed as a heading, paragraph, and bulleted list.

Sou		
	Lens Adapter	\$2,633
	Slim Digital	\$15,956
	Tripod	\$2,550
	Total	\$59,855
Total		\$192,482
Limitations of cascading style sheet attributes		
Only a basic set of cascading style sheet (CSS) attributes are defined:		
<ul style="list-style-type: none"> • text-align, text-indent • font-family, font-size • color • padding, padding-bottom, padding-top, padding-right, padding-left • font-weight 		

Save the Report

You can save reports to a report server, SharePoint library, or your computer.

In this tutorial, save the report to a report server. If you do not have access to a report server, save the report to your computer.

To save the report on a report server

1. From the **Report Builder** button, click **Save As**.
2. Click **Recent Sites and Servers**.
3. Select or type the name of the report server where you have permission to save reports.

The message "Connecting to report server" appears. When the connection is complete, you see the contents of the report folder that the report server administrator specified as the default location for reports.

4. In **Name**, replace the default name with a name of your choosing.
5. Click **Save**.

The report is saved to the report server. The name of report server that you are connected to appears in the status bar at the bottom of the window.

To save the report on your computer

1. From the **Report Builder** button, click **Save As**.
2. Click **Desktop**, **My Documents**, or **My computer**, and then browse to the folder where you want to save the report.
3. In **Name**, replace the default name with a name of your choosing.
4. Click **Save**.

Next Steps

There are many ways to format text in Report Builder. [Tutorial: Creating a Free Form Report](#) contains more examples.

[Report Builder Tutorials](#) [Formatting Report Items](#)

[Report Builder in SQL Server](#)

More questions? [Try asking the Reporting Services forum](#)

Tutorial: Add a Column Chart to Your Report (Report Builder)

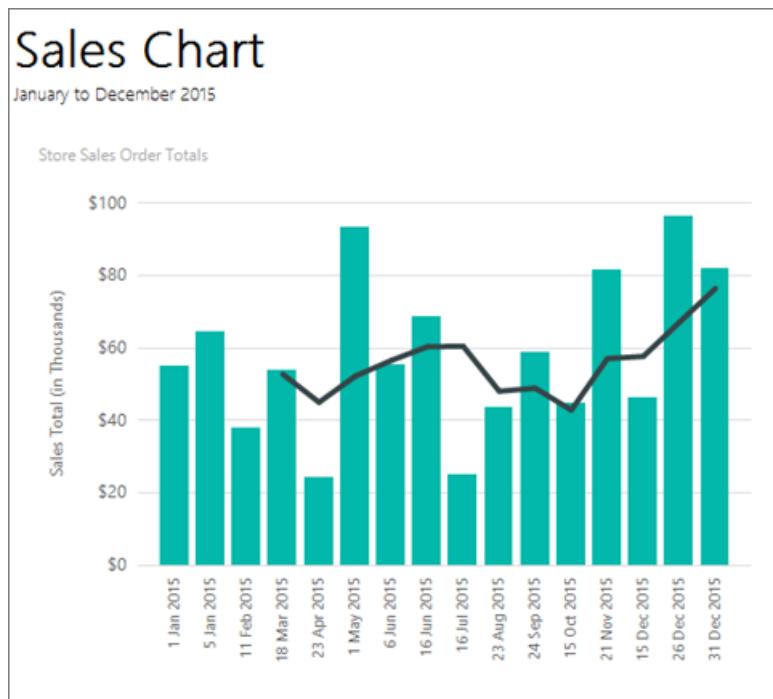
11/30/2018 • 8 minutes to read • [Edit Online](#)

In this tutorial, you create a Reporting Services paginated report with a column chart displaying a series as a set of vertical bars grouped by category.

Column charts are useful to:

- Show data changes over a period of time.
- Compare the relative value of multiple series.
- Display a moving average to show trends.

The following illustration shows the column chart you will create, with a moving average.



NOTE

In this tutorial, the steps for the wizard are consolidated into one procedure. For step-by-step instructions about how to browse to a report server, choose a data source, and create a dataset, see the first tutorial in this series: [Tutorial: Creating a Basic Table Report \(Report Builder\)](#).

Estimated time to complete this tutorial: 15 minutes.

Requirements

For information about requirements, see [Prerequisites for Tutorials \(Report Builder\)](#).

1. Create a Chart Report from the Chart Wizard

In this section, you go through the Chart Wizard to create an embedded dataset, choose a shared data source, and create a column chart.

NOTE

The query in this tutorial contains the data values, so it does not need an external data source. This makes the query quite long. In a business environment, a query would not contain the data. This is for learning purposes only.

To create a chart report

1. [Start Report Builder](#) either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the left pane, verify that **New Report** is selected.
3. In the right pane, click **Chart Wizard**.
4. On the **Choose a dataset page**, click **Create a dataset**, and then click **Next**.
5. On the **Choose a connection to a data source** page, select an existing data source or browse to the report server and select a data source, and then click **Next**. You may need to enter a user name and password.

NOTE

The data source you choose is unimportant, as long as you have adequate permissions. You will not be getting data from the data source. For more information, see [Alternative Ways to Get a Data Connection \(Report Builder\)](#).

6. On the **Design a query** page, click **Edit as Text**.

7. Paste the following query into the query pane:

```
SELECT CAST('2015-01-01' AS date) AS SalesDate, CAST(54995.21 AS money) AS Sales
UNION SELECT CAST('2015-01-05' AS date) AS SalesDate, CAST(64499.04 AS money) AS Sales
UNION SELECT CAST('2015-02-11' AS date) AS SalesDate, CAST(37821.79 AS money) AS Sales
UNION SELECT CAST('2015-03-18' AS date) AS SalesDate, CAST(53633.08 AS money) AS Sales
UNION SELECT CAST('2015-04-23' AS date) AS SalesDate, CAST(24019.3 AS money) AS Sales
UNION SELECT CAST('2015-05-01' AS date) AS SalesDate, CAST(93245.5 AS money) AS Sales
UNION SELECT CAST('2015-06-06' AS date) AS SalesDate, CAST(55288.0 AS money) AS Sales
UNION SELECT CAST('2015-06-16' AS date) AS SalesDate, CAST(68733.5 AS money) AS Sales
UNION SELECT CAST('2015-07-16' AS date) AS SalesDate, CAST(24750.85 AS money) AS Sales
UNION SELECT CAST('2015-08-23' AS date) AS SalesDate, CAST(43452.3 AS money) AS Sales
UNION SELECT CAST('2015-09-24' AS date) AS SalesDate, CAST(58656. AS money) AS Sales
UNION SELECT CAST('2015-10-15' AS date) AS SalesDate, CAST(44583. AS money) AS Sales
UNION SELECT CAST('2015-11-21' AS date) AS SalesDate, CAST(81568. AS money) AS Sales
UNION SELECT CAST('2015-12-15' AS date) AS SalesDate, CAST(45973. AS money) AS Sales
UNION SELECT CAST('2015-12-26' AS date) AS SalesDate, CAST(96357. AS money) AS Sales
UNION SELECT CAST('2015-12-31' AS date) AS SalesDate, CAST(81946. AS money) AS Sales
```

8. (Optional) Click the Run button (!) to see the data your chart will be based on.

9. Click **Next**.

2. Choose the Chart Type

You can choose from several predefined chart types, and then modify the chart after you complete the wizard.

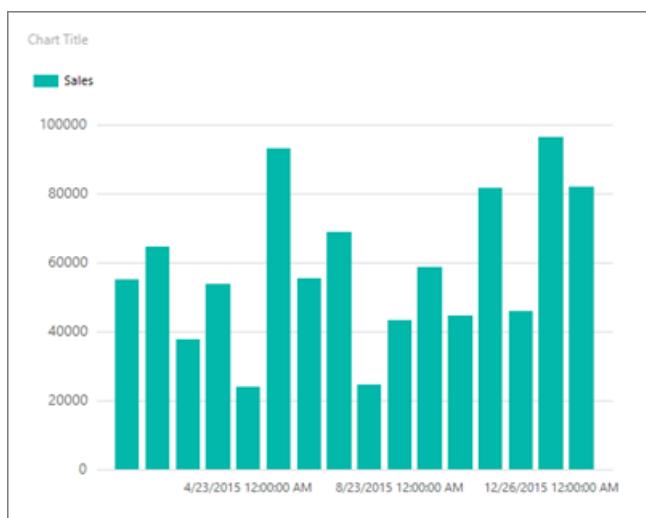
To add a column chart

1. On the **Choose a chart type** page, the column chart is the default chart type. Click **Next**.
2. On the **Arrange chart fields** page, drag the SalesDate field to **Categories**. Categories display on the horizontal axis.
3. Drag the Sales field to **Values**. The **Values** box displays Sum(Sales) because the sum of the sales total value is aggregated for each date. Values display on the vertical axis.
4. Click **Next**.
5. Click **Finish**.

The chart is added to the design surface. Note that the new column chart just shows representational data. The legend reads Sales Date A, Sales Date B, etc., just to give an idea of what your report will look like.



6. Click the chart to display the chart handles. Drag the bottom-right corner of the chart to increase the size of the chart. Note that the report design surface increases in size to accommodate the chart size.
7. Click **Run** to preview the report.



Note that the chart does not label every category on the horizontal axis. By default, only labels that fit next to the axis are included.

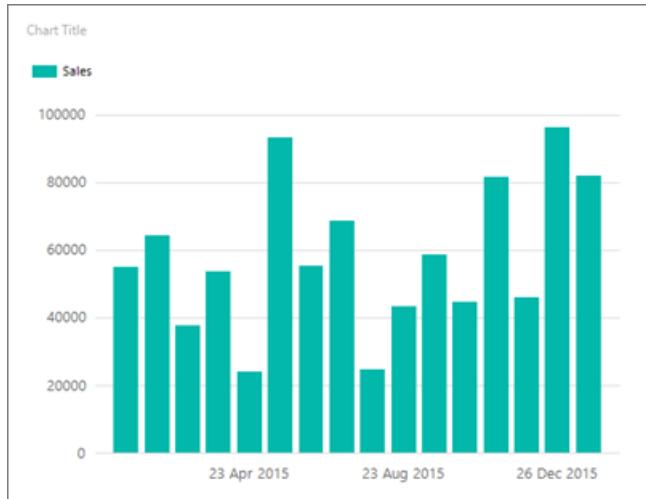
3. Format a Date on the Horizontal Axis

By default, the horizontal axis displays values in a general format that is automatically scaled to fit the size of the chart.

1. Switch to report design view.

2. Right-click the horizontal axis > **Horizontal Axis Properties**.
3. On the **Number** tab, in **Category**, select **Date**.
4. In the **Type** box, select **31 Jan 2000**.
5. Click **OK**.
6. On the Home tab, click **Run** to preview the report.

The date displays in the date format that you selected. The chart still does not label every category on the horizontal axis.

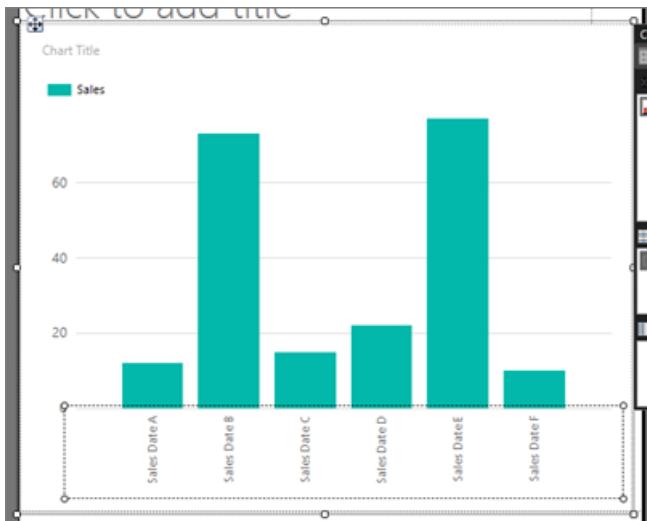


You can customize the label display by rotating the labels and specifying the interval.

4. Rotate the axis labels on the horizontal axis

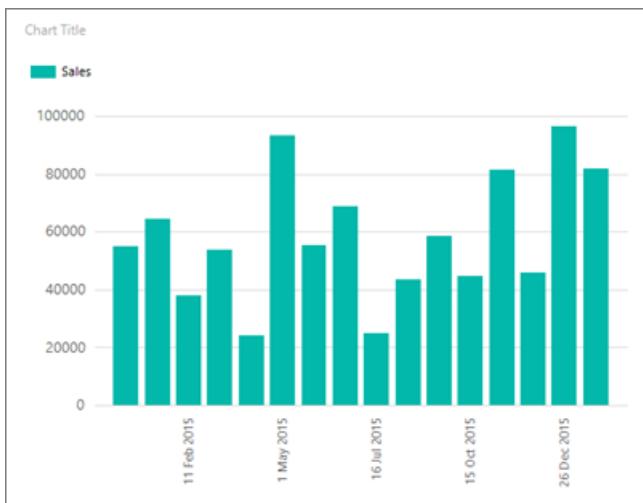
1. Switch to report design view.
2. Right-click the horizontal axis title, then click **Show Axis Title** to remove the title. Because the horizontal axis displays dates, the title is not needed.
3. Right-click the horizontal axis > **Horizontal Axis Properties**.
4. On the **Labels** tab, under **Change axis label auto-fit options**, select **Disable auto-fit**.
5. In **Label rotation angle**, select **-90**.
6. Click **OK**.

The sample text for the horizontal axis rotates by 90 degrees.



7. Click **Run** to preview the report.

On the chart, the labels are rotated.



5. Move the Legend

The legend is automatically created from category and series data. You can move the legend below the chart area of a column chart.

1. Switch to report design view.
2. Right-click the legend on the chart > **Legend Properties**.
3. Under **Layout and Position**, select a different position. For example, select the bottom middle option.

When the legend is placed at the top or bottom of a chart, the layout of the legend changes from vertical to horizontal. You can select a different layout in the **Layout** box.

4. Click **OK**.
5. (Optional) Because there is only one category in this tutorial, the chart doesn't need a legend. To remove it, right-click the legend > **Delete Legend**.
6. Click **Run** to preview the report.

6. Title the Chart

1. Switch to report design view.

2. Select the words **Chart Title** at the top of the chart, then type **Store Sales Order Totals**.

3. Click **Run** to preview the report.

7. Format and Label the Vertical Axis

By default, the vertical axis displays values in a general format that is automatically scaled to fit the size of the chart.

1. Switch to report design view.

2. Click the labels on the vertical axis on the left side of the chart to select them.

3. On the **Home** tab > **Number** group, click the **Currency** button. The axis labels change to show the currency format.

4. Click the **Decrease Decimal** button two times, to show the number rounded to the nearest dollar.

5. Right-click the vertical axis > **Vertical Axis Properties**.

6. On the **Number** tab, note that **Currency** is already selected in the **Category** box, and **Decimal places** is already **0** (zero).

7. Check **Show Values in Thousands** is already selected.

8. Click **OK**.

9. Right-click the vertical axis > **Show Axis Title**.

10. Right-click the vertical axis title > **Axis Title Properties**.

11. Replace the text in the **Title text** field with **Sales Total (in Thousands)**. You can also specify a variety of options related to how the title is formatted.

12. Click **OK**.

13. Click **Run** to preview the report.



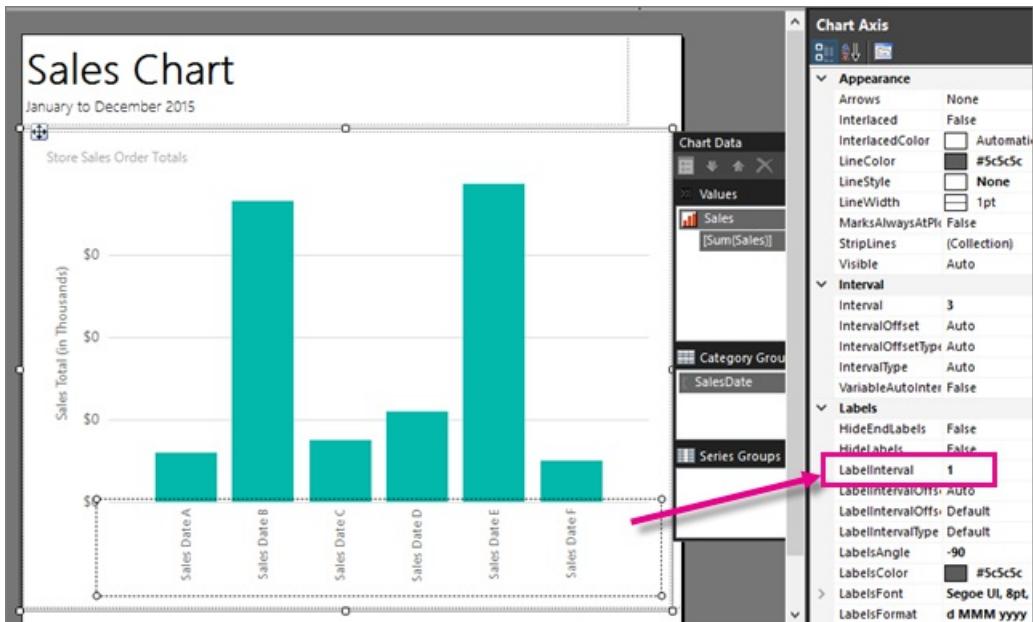
8. Show all the labels on the horizontal (x) axis

You notice that only some of the labels on the x axis are showing. In this section, you set a property in the Properties pane to show them all.

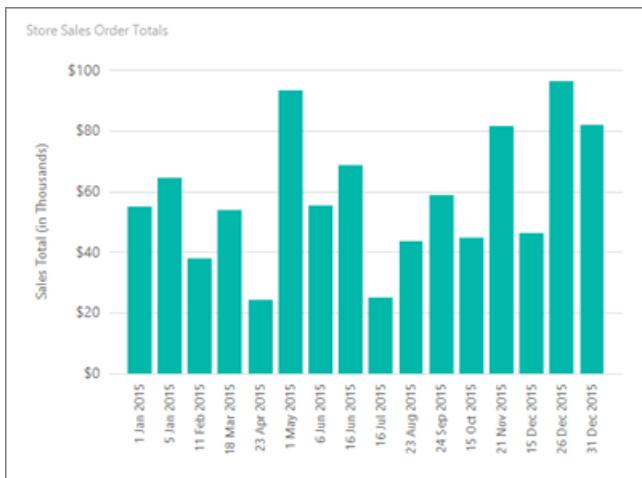
1. Switch to report design view.

2. Click the chart, then select the horizontal axis labels.

- In the Properties pane, set LabelInterval to 1.



- Click **Run** to preview the report.

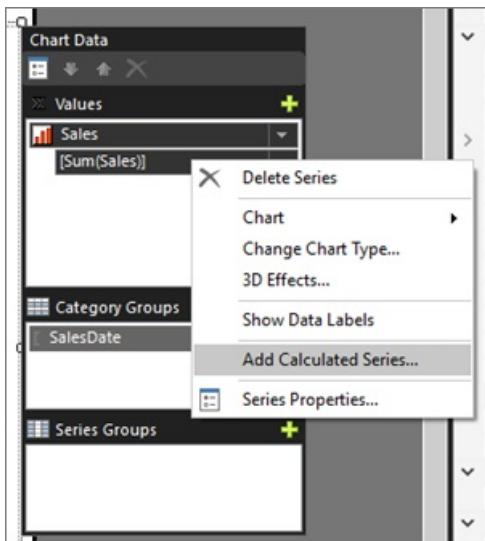


Now the chart displays all its labels.

9. Add a Moving Average with a Calculated Series

A moving average is an average of the data in your series, calculated over time. The moving average can identify trends.

- Switch to report design view.
- Double-click the chart to display the **Chart Data** pane.
- Right-click the **[Sum(Sales)]** field in the **Values** area, then click **Add Calculated Series**.



4. In **Formula**, verify that **Moving average** is selected.
5. In **Set Formula Parameters**, for **Period**, select **4**.
6. On the **Border** tab, in **Line width**, select **3pt**.
7. Click **OK**.
8. Click **Run** to preview the report.

The chart displays a line that shows the moving average for total sales by date, averaged over every four dates. Read more about [adding a moving average to a chart](#).



10. Add a Report Title

1. Switch to report design view.
2. On the design surface, click **Click to add title**.
3. Type **Sales Chart**, press ENTER, and then type **January to December 2015**, so it looks like this:

Sales Chart

January to December 2015

4. Select **Sales Chart**, and on the **Home** tab > **Font** section > **Bold**.
5. Select **January to December 2015**, and on the **Home** tab > **Font** section > set font size to **10**.
6. (Optional) You may need to make the **Title** text box taller to accommodate the two lines of text. Pull down

on the double-headed arrows when you click in the middle of the bottom edge. And you may need to drag the top of the chart so the title doesn't overlap.

This title appears at the top of the report. When there is no page header defined, items at the top of the report body are the equivalent of a report header.

7. Click **Run** to preview the report.

11. Save the Report

To save the report

1. Switch to report design view.
2. From the Report Builder button, click **Save As**.

You can save it either to your computer or to the report server.

3. In **Name**, type **Sales Order Column Chart**.
4. Click **Save**.

Next Steps

You have successfully completed the Adding a Column Chart to Your Report tutorial. To learn more about charts, see [Charts \(Report Builder and SSRS\)](#) and [Sparklines and Data Bars \(Report Builder and SSRS\)](#).

See Also

- [Report Builder tutorials](#)
- [Report Builder in SQL Server](#)

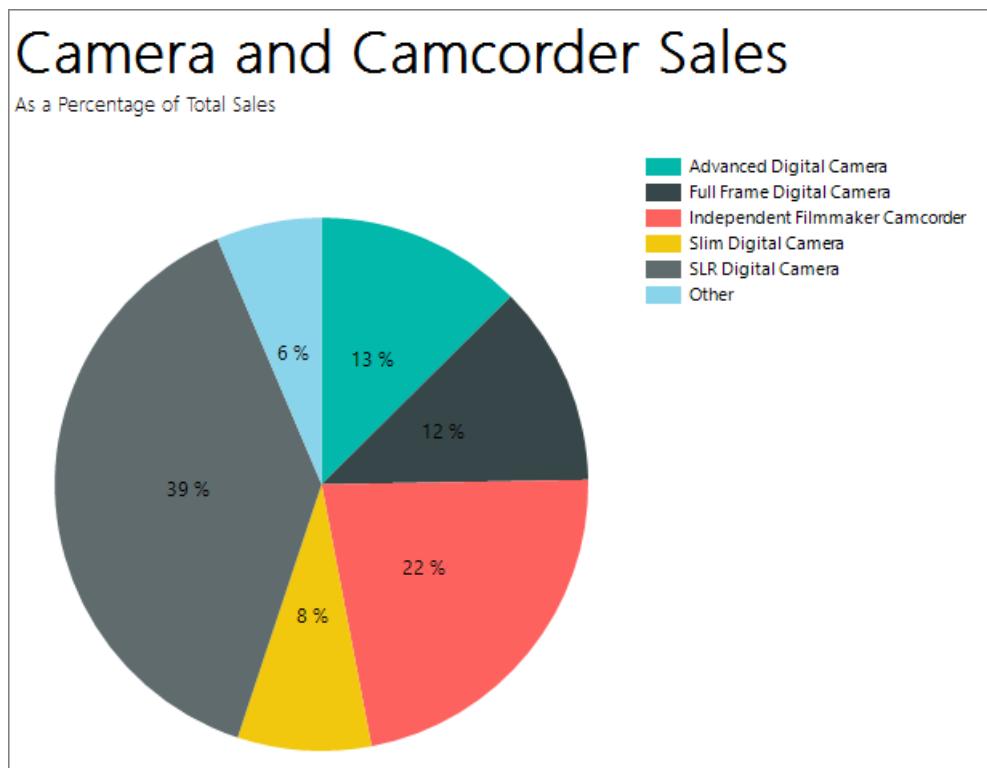
Tutorial: Add a Pie Chart to Your Report (Report Builder)

11/30/2018 • 7 minutes to read • [Edit Online](#)

In this tutorial, you create pie chart in a Reporting Services paginated report. You add percentages and combine small slices into a single slice.

Pie and doughnut charts display data as a proportion of the whole. They have no axes. When you add a numeric field to a pie chart, the chart calculates the percentage of each value to the total.

This illustration shows the pie chart you will create.



If there are too many data points on a pie chart, your data point labels might be too crowded to read. In that case, consider combining a number of small slices into one larger slice. Pie charts are more readable when you have aggregated your data into a few data points.

NOTE

In this tutorial, the steps for the wizard are consolidated into two procedures. For step-by-step instructions about how to browse to a report server, add a data source, and add a dataset, see the first tutorial in this series: [Tutorial: Creating a Basic Table Report \(Report Builder\)](#).

Estimated time to complete this tutorial: 10 minutes

Requirements

For information about requirements, see [Prerequisites for Tutorials \(Report Builder\)](#).

1. Create a Pie Chart from the Chart Wizard

In this section, you use the Chart Wizard to create an embedded dataset, choose a shared data source, and create a pie chart.

1. Start Report Builder either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the left pane, verify that **New Report** is selected.
3. In the right pane, click **Chart Wizard**.
4. On the **Choose a dataset** page, click **Create a dataset**, and then click **Next**.
5. On the **Choose a connection to a data source** page, select an existing data source or browse to the report server and select a data source, and then click **Next**. You may need to enter a user name and password.

NOTE

The data source you choose is unimportant, as long as you have adequate permissions. You will not be getting data from the data source. For more information, see [Alternative Ways to Get a Data Connection \(Report Builder\)](#).

6. On the **Design a Query** page, click **Edit as Text**.

7. Paste the following query into the query pane:

NOTE

In this tutorial, the query contains the data values, so it does not need an external data source. This makes the query long. In a business environment, a query would not contain the data. This is for learning purposes only.

```
SELECT 'Advanced Digital Camera' AS Product, CAST(254995.21 AS money) AS Sales
UNION SELECT 'Slim Digital Camera' AS Product, CAST(164499.04 AS money) AS Sales
UNION SELECT 'SLR Digital Camera' AS Product, CAST(782176.79 AS money) AS Sales
UNION SELECT 'Lens Adapter' AS Product, CAST(36333.08 AS money) AS Sales
UNION SELECT 'Macro Zoom Lens' AS Product, CAST(40199.3 AS money) AS Sales
UNION SELECT 'USB Cable' AS Product, CAST(53245.5 AS money) AS Sales
UNION SELECT 'Independent Filmmaker Camcorder' AS Product, CAST(452288.0 AS money) AS Sales
UNION SELECT 'Full Frame Digital Camera' AS Product, CAST(247250.85 AS money) AS Sales
```

8. (Optional) Click the Run button (!) to see the data your chart will be based on.

9. Click **Next**.

2. Choose the Chart Type

You can choose from a variety of predefined chart types.

1. On the **Choose a chart type** page, click **Pie**, then click **Next**. The **Arrange chart fields** page opens.

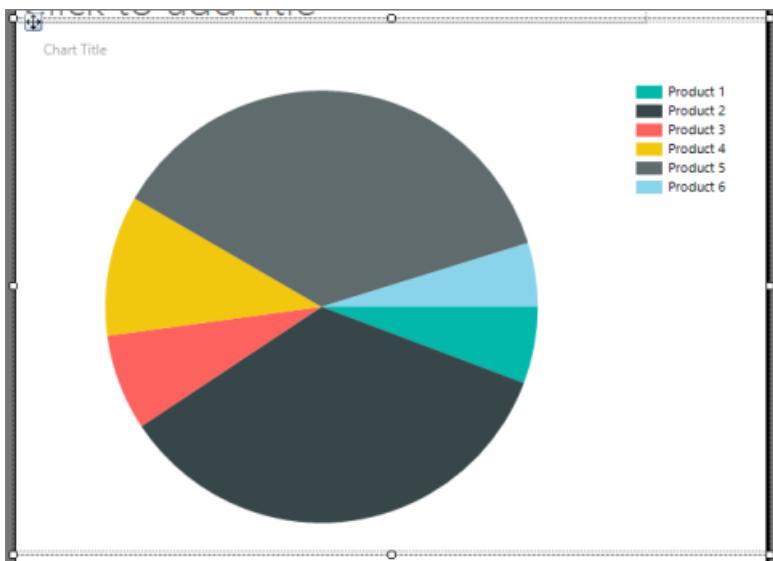
On the **Arrange chart fields** page, drag the Product field to the **Categories** pane. Categories define the number of slices in the pie chart. In this example, there will be eight slices, one for each product.

2. Drag the Sales field to the **Values** pane. Sales represents the sales amount for the subcategory. The **Values** pane displays `[Sum(Sales)]` because the chart displays the aggregate for each product.

3. Click **Next** to see a preview.

4. Click **Finish**.

The chart is added to the design surface. You don't see the actual values of the pie chart -- you see Product 1, Product 2, etc., to give an idea of how the chart will look.

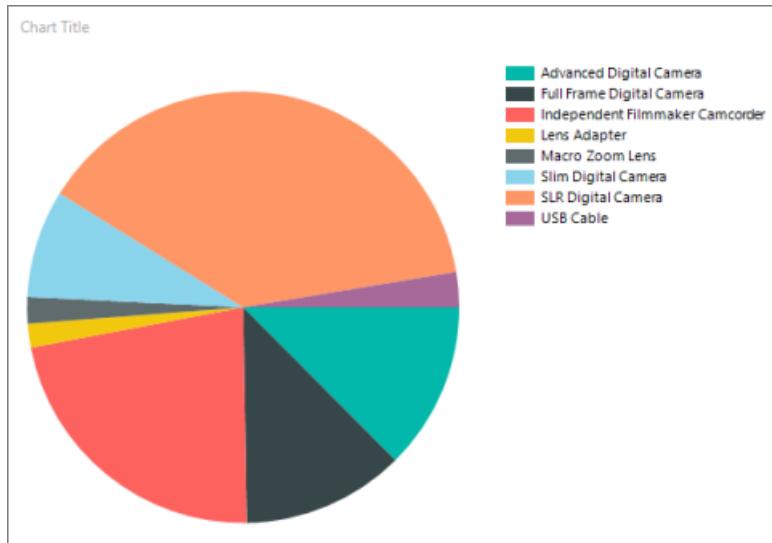


5. Click the chart to display the chart handles. Drag the bottom-right corner of the chart to make it bigger.

Note that the report design surface also gets bigger, to accommodate the chart size.

6. Click **Run** to preview the report.

The report displays the pie chart with eight slices, one for each product. Now you see the actual products and the size of each slice represents the sales for that product. Three of the slices are quite thin.



3. Display Percentages in Each Slice

On each slice of the pie, you can display a percentage for this slice compared to the whole pie.

1. Switch to report design view.
2. Right-click the pie chart and click **Show Data Labels**. The data labels appear on the chart.
3. Right-click a label, then click **Series Label Properties**.
4. In the **Label data** box, select **#PERCENT**.

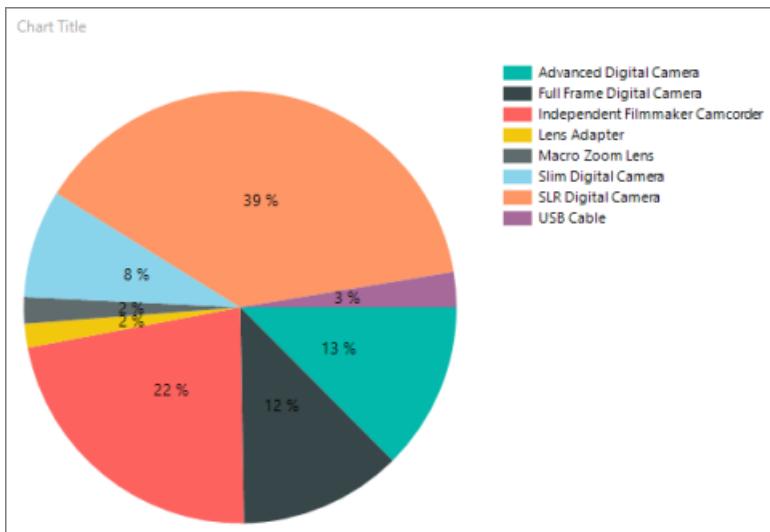
- (Optional) To specify how many decimal places the label shows, in the **Label data** box after **#PERCENT**, type **{Pn}** where *n* is the number of decimal places to display. For example, to display no decimal places, type **#PERCENT{P0}**.
- To display values as percentages, the **UseValueAsLabel** property must be false. If you are prompted to set this value in the **Confirm Action** dialog, click **Yes**.

NOTE

Number Format in the **Series Label Properties** dialog box has no effect when you format percentages. This formats the labels as percentages, but does not calculate the percentage of the pie that each slice represents.

- Click **OK**.
- Click **Run** to preview the report.

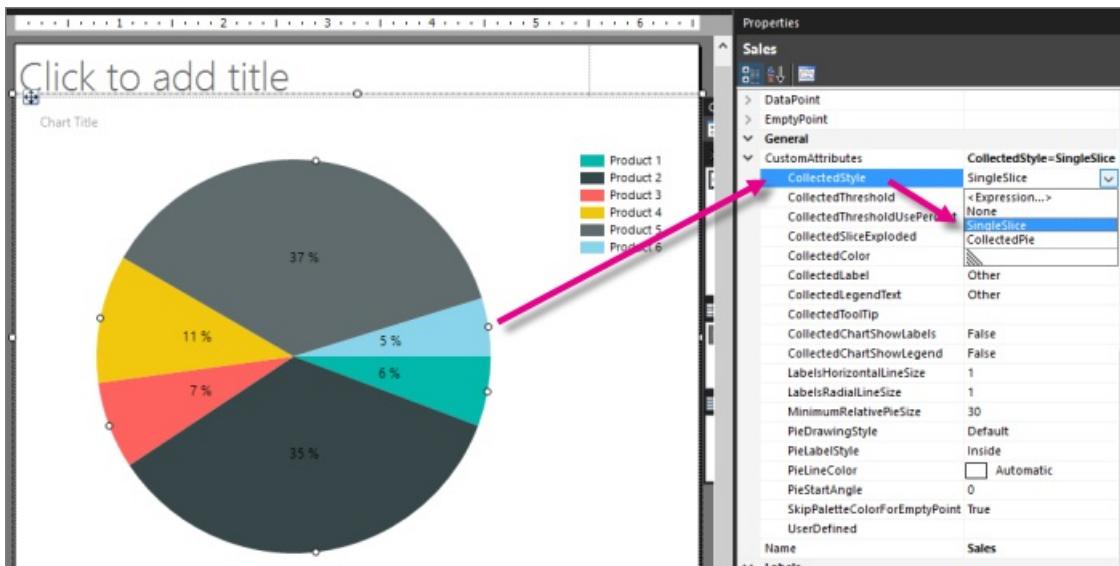
The report displays the percentage of the whole for each pie slice.



4. Combine Small Slices into One Slice

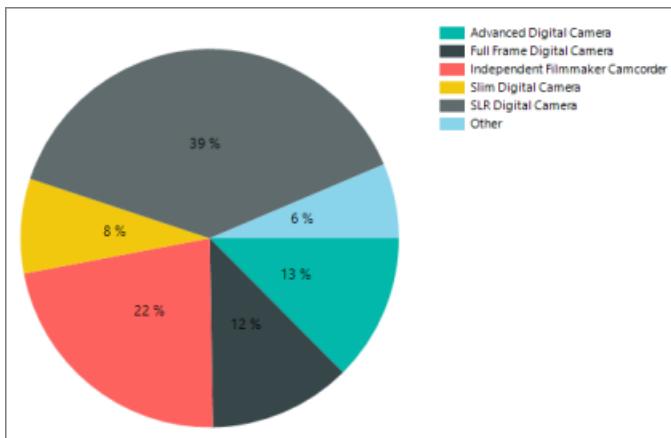
Three of the slices in the pie are quite small. You can combine multiple small slices into one larger "Other" slice that represents all three.

- Switch to report design view.
- If the Properties pane isn't showing, on the **View** tab > **Show/Hide** group > select **Properties**.
- On the design surface, click on any slice of the pie chart. The properties for the series are displayed in the Properties pane.
- In the **General** section, expand the **CustomAttributes** node.
- Set the **CollectedStyle** property to **SingleSlice**.



6. Verify that the **CollectedThreshold** property is set to 5.
7. Verify that the **CollectedThresholdUsePercent** property is set to **True**.
8. On the **Home** tab, click **Run** to preview the report.

In the legend, you now see the category "Other". The new pie slice combines all the slices that were under 5% into one slice that is 6% of the whole pie.



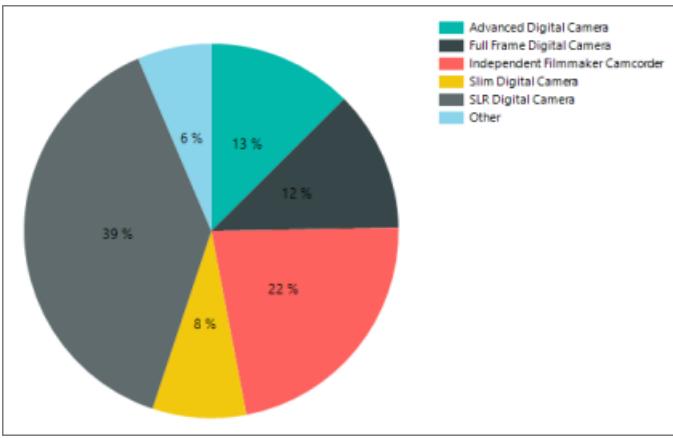
5. Start pie chart values at the top

By default in pie charts, the first value in the dataset starts at 90 degrees from the top of the pie. You see that in the pie chart in the previous sections.

In this section, we'll make the first value start at the top.

1. Switch to report design view.
2. Select the pie itself.
3. In the Properties pane, under **Custom Attributes**, change PieStartAngle from **0** to **270**.
4. Click **Run** to preview your report.

Now the pie chart slices are in alphabetical order, starting at the top, and ending with the "Other" slice.



6. Add a Report Title

Because the pie chart is the only visualization in the report, the chart doesn't need its own title. The report title will do.

1. In the chart, select the Chart Title box and press DELETE.
2. On the design surface, click **Click to add title**.
3. Type **Camera and Camcorder Sales**, press ENTER, and then type **As a Percentage of Total Sales**, so it looks like this:

Camera and Camcorder Sales

As a Percentage of Total Sales

4. Select **Camera and Camcorder Sales**, and on the **Home** tab > **Font** section > click **Bold**.
5. Select **As a Percentage of Total Sales**, and on the **Home** tab > **Font** section > set the font size to **10**.
6. (Optional) You may need to make the Title text box taller to accommodate the two lines of text.

This title will appear at the top of the report. When there is no page header defined, items at the top of the report body are the equivalent of a report header.

7. Click **Run** to preview the report.

7. Save the Report

To save the report

1. Switch to report design view.
2. On the **File** menu, click **Save**.
3. In **Name**, type **Sales Pie Chart**.
4. Click **Save**.

Your report is saved on the report server.

Next Steps

You have successfully completed the Adding a Pie Chart to Your Report tutorial. To learn more about charts, see [Charts \(Report Builder and SSRS\)](#) and [Sparklines and Data Bars \(Report Builder and SSRS\)](#).

See Also

[Report Builder Tutorials](#)

[Report Builder in SQL Server](#)

Tutorial: Add a Bar Chart to Your Report (Report Builder)

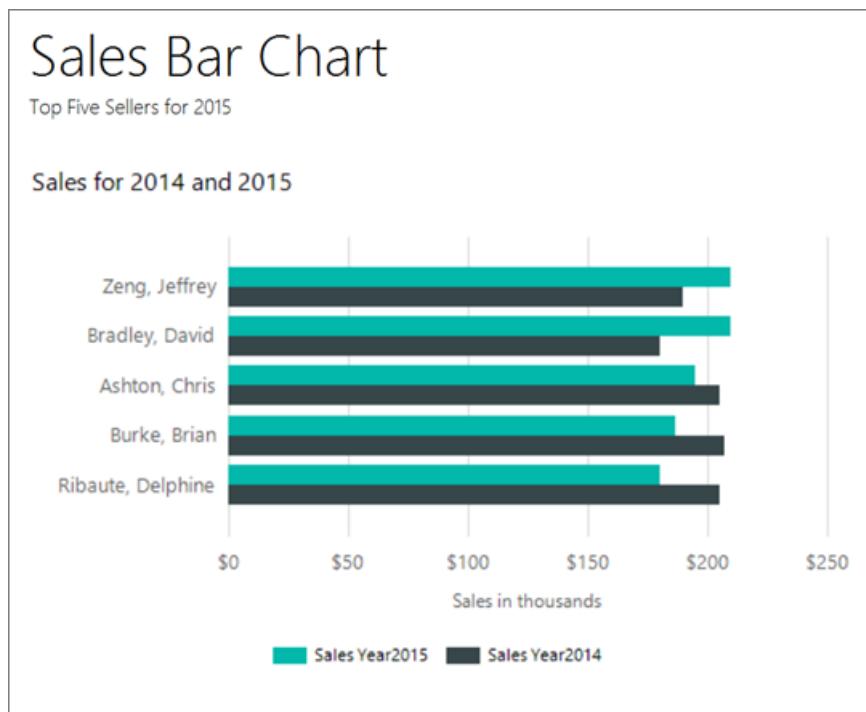
11/30/2018 • 10 minutes to read • [Edit Online](#)

In this tutorial, you use a wizard in Report Builder to create a bar chart in a Reporting Services paginated report. Then you add a filter and enhance the chart.

A bar chart displays category data horizontally. This can help to:

- Improve readability of long category names.
- Improve understandability of times plotted as values.
- Compare the relative value of multiple series.

The following illustration shows the bar chart that you will create, with sales for 2014 and 2015 for the top five salespeople, from most to least 2015 sales.



NOTE

In this tutorial, the steps for the wizard are consolidated into one procedure. For step-by-step instructions about how to browse to a report server, create a dataset, and choose a data source, see the first tutorial in this series: [Tutorial: Creating a Basic Table Report \(Report Builder\)](#).

Estimated time to complete this tutorial: 15 minutes.

Requirements

For more information about requirements, see [Prerequisites for Tutorials \(Report Builder\)](#).

1. Create a Chart Report from the Chart Wizard

In which you create an embedded dataset, choose a shared data source, and create a bar chart by using the Chart

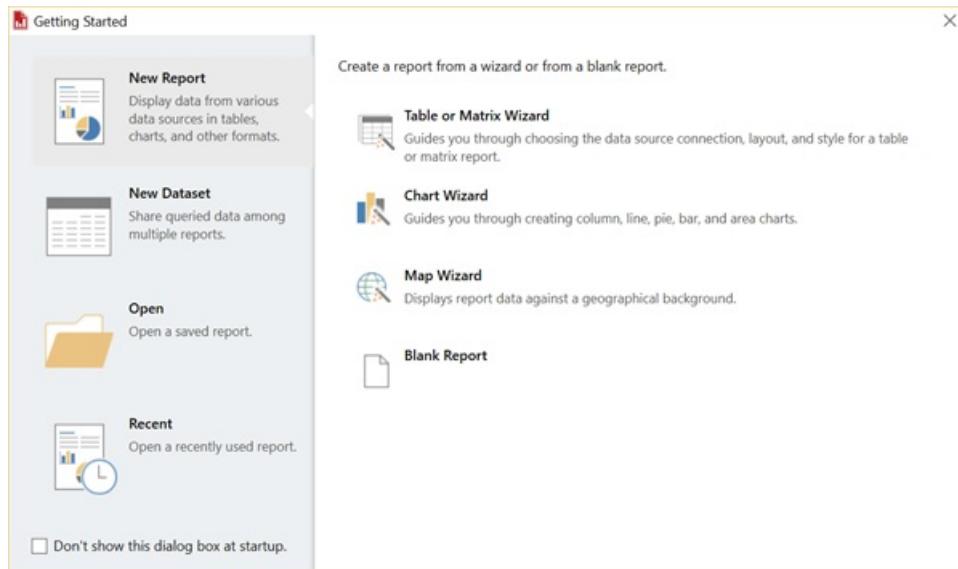
Wizard.

NOTE

In this tutorial, the query contains the data values so that it does not need an external data source. This makes the query quite long. In a business environment, a query would not contain the data. This is for learning purposes only.

1. Start Report Builder from the Reporting Services web portal, from the report server in SharePoint integrated mode, or from your computer.

The **Getting Started** dialog box appears.



If you don't see the **Getting Started** dialog box, click **File > New**. The **New Report or Dataset** dialog box has most of the same contents as the **Getting Started** dialog box.

2. In the left pane, verify that **New Report** is selected.
3. In the right pane, click **Chart Wizard**.
4. On the **Choose a dataset** page, click **Create a dataset**, and then click **Next**.
5. On the **Choose a connection to a data source** page, select an existing data source or browse to the report server and select a data source, and then click **Next**. You may need to enter a user name and password.

NOTE

The data source you choose is unimportant, as long as you have adequate permissions. You will not be getting data from the data source. For more information, see [Alternative Ways to Get a Data Connection \(Report Builder\)](#).

6. On the **Design a query** page, click **Edit as Text**.
7. Paste the following query into the query pane:

```

SELECT 'Luis' as FirstName, 'Alverca' as LastName, CAST(170000.00 AS money) AS SalesYear2015,
CAST(150000. AS money) AS SalesYear2014
UNION SELECT 'Jeffrey' as FirstName, 'Zeng' as LastName, CAST(210000. AS money) AS SalesYear2015,
CAST(190000. AS money) AS SalesYear2014
UNION SELECT 'Houman' as FirstName, 'Pournasseh' as LastName, CAST(150000. AS money) AS SalesYear2015,
CAST(180000. AS money) AS SalesYear2014
UNION SELECT 'Robin' as FirstName, 'Wood' as LastName, CAST(75000. AS money) AS SalesYear2015,
CAST(175000. AS money) AS SalesYear2014
UNION SELECT 'Daniela' as FirstName, 'Guaita' as LastName, CAST(170000. AS money) AS SalesYear2015,
CAST(175000. AS money) AS SalesYear2014
UNION SELECT 'John' as FirstName, 'Yokim' as LastName, CAST(160000. AS money) AS SalesYear2015,
CAST(195000. AS money) AS SalesYear2014
UNION SELECT 'Delphine' as FirstName, 'Ribaute' as LastName, CAST(180000. AS money) AS SalesYear2015,
CAST(205000. AS money) AS SalesYear2014
UNION SELECT 'Robert' as FirstName, 'Hernady' as LastName, CAST(140000. AS money) AS SalesYear2015,
CAST(180000. AS money) AS SalesYear2014
UNION SELECT 'Tanja' as FirstName, 'Plate' as LastName, CAST(150000. AS money) AS SalesYear2015,
CAST(160000. AS money) AS SalesYear2014
UNION SELECT 'David' as FirstName, 'Bradley' as LastName, CAST(210000. AS money) AS SalesYear2015,
CAST(180000. AS money) AS SalesYear2014
UNION SELECT 'Michal' as FirstName, 'Jaworski' as LastName, CAST(175000. AS money) AS SalesYear2015,
CAST(220000. AS money) AS SalesYear2014
UNION SELECT 'Chris' as FirstName, 'Ashton' as LastName, CAST(195000. AS money) AS SalesYear2015,
CAST(205000. AS money) AS SalesYear2014
UNION SELECT 'Pongsiri' as FirstName, 'Hirunyanitiwatna' as LastName, CAST(175000. AS money) AS
SalesYear2015, CAST(215000. AS money) AS SalesYear2014
UNION SELECT 'Brian' as FirstName, 'Burke' as LastName, CAST(187000. AS money) AS SalesYear2015,
CAST(207000. AS money) AS SalesYear2014

```

8. (Optional) Click the Run button (!) to see the data your chart will be based on.

9. Click **Next**.

2. Create a Bar Chart

1. On the **Choose a chart type** page, the column chart is the default chart type.

2. Click **Bar**, and then click **Next**.

On the **Arrange chart fields** page, there are four fields in the **Available fields** pane: FirstName, LastName, SalesYear2015, and SalesYear2014.

3. Drag LastName to the Categories pane.

4. Drag SalesYear2015 to the Values pane. SalesYear2015 represents the sales amount for each salesperson for the year 2015. The Values pane displays **[Sum(SalesYear2015)]** because the chart displays the aggregate for each product.

5. Drag SalesYear2014 to the Values pane under SalesYear2015. SalesYear2014 represents the sales amount for each salesperson for the year 2014.

6. Click **Next**.

7. Click **Finish**.

The chart is added to the design surface. Note that the new bar chart just shows representational data. The legend reads Last Name A, Last Name B, etc., rather than the people's names, just to give an idea of what your report will look like.

8. Click the chart to display the chart handles. Drag the bottom-right corner of the chart to increase the size of the chart. Notice the design surface gets larger as you drag.

9. Click **Run** to preview the report.

The bar chart displays sales for each sales person for the years 2014 and 2015. The length of the bar corresponds to the sales total.

3. Display All the Names on the Vertical Axis

By default, only some of the values on the vertical axis appear. You can change the chart to display all categories.

1. Switch to report design view.
2. Right-click the vertical axis, then click **Vertical Axis Properties**.
3. Under **Axis range and interval**, in the **Interval** box, type **1**.
4. Click **OK**.
5. Click **Run** to preview the report.

NOTE

If you cannot read the salesperson names on the vertical axis, you can make your chart taller or change the formatting options for the axis labels.

Display Last Name and First Name on Vertical Axis

You can change the category expression to include last name followed by first name of each sales person.

1. Switch to report design view.
2. Double-click the chart to display the **Chart Data** pane.
3. In the **Category Groups** area, right-click [LastName], and then click **Category Group Properties**.
4. In Label, click the expression (Fx) button.
5. Type the following expression: `=Fields!LastName.Value & ", " & Fields!FirstName.Value`
This expression concatenates the last name, a comma, and the first name.
6. Click **OK**.
7. Click **OK**.
8. Click **Run** to preview the report.

If the first names do not appear when you run the report, you can refresh the data manually. While still in preview mode, on the **Run** tab in the **Navigation** group, click **Refresh**.

NOTE

If you cannot read the salesperson names on the vertical axis, you can make your chart taller or change the formatting options for the axis labels.

4. Change the Sort Order on the Vertical Axis

When you sort the data on a chart, you are changing the order of values on the category axis.

1. Switch to report design view.
2. Double-click the chart to display the **Chart Data** pane.

3. In the **Category Groups** area, right-click [LastName], and then click **Category Group Properties**.
4. Click **Sorting**. The **Change sorting options** page displays a list of sort expressions. By default, this list has one sort expression that is the same as the original category group expression.
5. In **Sort by**, click **[SalesYear2015]**.
6. in the **Order** list, select **A to Z** so that the names appear in order from largest to smallest 2015 sales.
7. Click **OK**.
8. Click **Run** to preview the report.

The names on the horizontal axis are sorted from largest to smallest 2015 sales, with **Zeng** at the top.

5. Move the Legend

To improve the readability of the chart values, you might want to move the chart legend. For example, in a bar chart where bars are shown horizontally, you can change the position of the legend so that it is above or below the chart area. This gives more horizontal space to the bars.

To display the legend below the chart area of a bar chart

1. Switch to report design view.
2. Right-click the legend on the chart.
3. Select **Legend Properties**.
4. For **Legend position**, select a different position. For example, set the position to the middle bottom option.

When the legend is placed at the top or bottom of a chart, the layout of the legend changes from vertical to horizontal. You can select a different layout from the **Layout** drop-down list.

5. Click **OK**.
6. Click **Run** to preview the report.

6. Title the Chart

1. Switch to report design view.
2. Select the words **Chart Title** at the top of the chart, then type: **Sales for 2014 and 2015**.
3. In the Properties pane, with the title selected, set **Color** to **Black** and **FontSize** to **12pt**.
4. Click **Run** to preview the report.

7. Format and Label the Horizontal Axis

By default, the horizontal axis displays values in a general format that is automatically scaled to fit the size of the chart. You can change it to the currency format.

1. Switch to report design view.
2. Click the horizontal axis along the bottom of the chart to select it.
3. On the **Home** tab > **Number** group > **Currency**. The horizontal axis labels change to currency.
4. (Optional) Remove the decimal digits. Near the **Currency** button, click the **Decrease Decimal** button twice.
5. Right-click the horizontal axis, and click **Horizontal Axis Properties**.

6. On the **Number** tab, select **Show values in Thousands**.
7. Click **OK**.
8. Right-click the horizontal axis, and select **Show Axis Title**.
9. In the **Axis Title** box, type **Sales in thousands** and press Enter.

Note: While you're typing, the Axis Title box appears to be on the vertical axis. But when you press Enter, it goes to the horizontal axis.

10. Click **Run** to preview the report.

The report displays the sales amount on the horizontal axis as currency in thousands, with no decimal digits.

8. Add a Filter to Display the Top Five Values

You can add a filter to the chart to specify which data from the dataset to include or exclude in the chart.

1. Switch to report design view.
2. Double-click the chart to display the **Chart Data** pane.
3. In the **Category Groups** area, right-click the [LastName] field, and then click **Category Group Properties**.
4. Click **Filters**. The **Change filters** page can display a list of filter expressions. By default, this list is empty.
5. Click **Add**. A new blank filter appears.
6. In **Expression**, type **[Sum(SalesYear2015)]**. This creates the underlying expression
`=Sum(Fields!SalesYear2015.Value)`, which you can see if you click the **fx** button.
7. Verify that the data type is **Text**.
8. In **Operator**, select **Top N** from the drop-down list.
9. In **Value**, type the following expression: **=5**
10. Click **OK**.
11. Click **Run** to preview the report.

If the results are not filtered when you run the report, you can refresh the data manually. On the **Run** tab in the **Navigation** group, click **Refresh**.

The chart shows the top five salesperson names from the 2015 sales data.

9. Add a Report Title

1. On the design surface, click **Click to add title**.
2. Type **Sales Bar Chart**, press ENTER, and then type **Top Five Sellers for 2015**, so it looks like this:

Sales Bar Chart

Top Five Sellers for 2015

3. Select **Sales Bar Chart**, and click the **Bold** button.
4. Select **Top Five Sellers for 2015**, and in the **Font** section on the **Home** tab, set the font size to **10**.
5. (Optional) You may need to make the Title text box taller, and bring down the top of the bar chart, to

accommodate the two lines of text.

This title will appear at the top of the report. When there is no page header defined, items at the top of the report body are the equivalent of a report header.

6. Click **Run** to preview the report.

10. Save the Report

1. Switch to report design view.
2. Click **File > Save As**.
3. In **Name**, type **Sales Bar Chart**.

You can save it either to your computer or to the report server.

4. Click **Save**.

Next Steps

You have successfully completed the Adding a Bar Chart to Your Report tutorial. To learn more about charts, see [Charts](#) and [Bar Charts](#).

See Also

[Report Builder Tutorials](#)

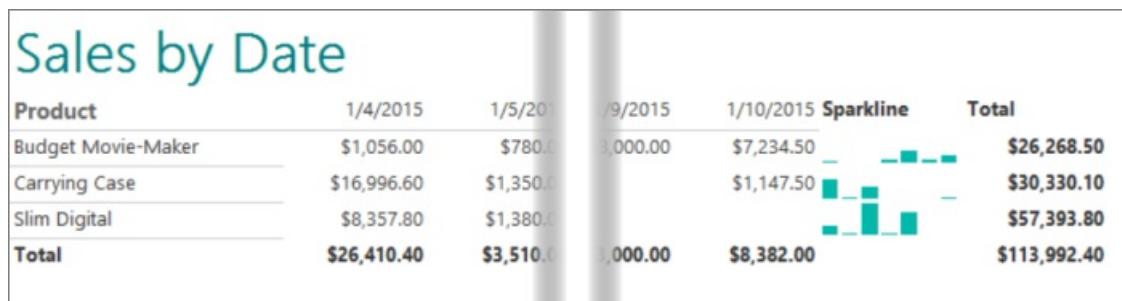
[Report Builder in SQL Server](#)

Tutorial: Add a Sparkline to Your Report (Report Builder)

11/30/2018 • 11 minutes to read • [Edit Online](#)

In this tutorial in Report Builder, you create a basic table with a sparkline chart in a Reporting Services paginated report.

Sparklines and data bars are small, simple charts that convey a lot of information in a little space, often in tables and matrices in Reporting Services reports. The following illustration shows a report similar to the one that you will create.



Estimated time to complete this tutorial: 30 minutes.

Requirements

For more information about requirements, see [Prerequisites for Tutorials \(Report Builder\)](#).

1. Create a Report with a Table

1. [Start Report Builder](#) either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the left pane, verify that **New Report** is selected.
3. In the right pane, click **Table or Matrix Wizard**.
4. On the **Choose a dataset** page, select **Create a dataset** > **Next**. The **Choose a connection to a data source** page opens.

NOTE

This tutorial doesn't need specific data; it just needs a connection to a SQL Server database. If you already have a data source connection listed under **Data Source Connections**, you can select it and go to step 10. For more information, see [Alternative Ways to Get a Data Connection \(Report Builder\)](#).

5. Click **New**. The **Data Source Properties** dialog box opens.
6. In **Name**, type **Product Sales**, a name for the data source.
7. In **Select a connection type**, verify that **Microsoft SQL Server** is selected.

8. In **Connection string**, type the following text:

```
Data Source\=<servername>
```

The expression `<servername>`, for example Report001, specifies a computer on which an instance of the SQL Server Database Engine is installed. Because the report data is not extracted from a SQL Server database, you need not include the name of a database. The default database on the specified server is used to parse the query.

9. Click **Credentials**. Enter the credentials that you need to access the external data source.

10. Click **OK**.

You are back on the **Choose a connection to a data source** page.

11. To verify that you can connect to the data source, click **Test Connection**.

The message "Connection created successfully" appears.

12. Click **OK**.

13. Click **Next**.

2. Create a Query and Table Layout in the Table Wizard

In a report, you can use a shared dataset that has a predefined query, or you can create an embedded dataset for use only in your report. In this tutorial, you will create an embedded dataset.

NOTE

In this tutorial, the query contains the data values, so that it does not need an external data source. This makes the query quite long. In a business environment, a query would not contain the data. This is for learning purposes only.

To create a query and table layout in the Table Wizard

1. On the **Design a query** page, the relational query designer is open. For this tutorial, you will use the text-based query designer.
2. Click **Edit As Text**. The text-based query designer displays a query pane and a results pane.
3. Paste the following Transact-SQL query into the **Query** box.

```

SELECT CAST('2015-01-04' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Carrying Case' as Product, CAST(16996.60 AS money) AS Sales, 68 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Carrying Case' as Product, CAST(1350.00 AS money) AS Sales, 18 as Quantity
UNION SELECT CAST('2015-01-10' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Carrying Case' as Product, CAST(1147.50 AS money) AS Sales, 17 as Quantity
UNION SELECT CAST('2015-01-04' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Budget Movie-Maker' as Product, CAST(1056.00 AS money) AS Sales, 44 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Slim Digital' as Product, CAST(1380.00 AS money) AS Sales, 18 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate,'Accessories' as Subcategory,
    'Budget Movie-Maker' as Product, CAST(780.00 AS money) AS Sales, 26 as Quantity
UNION SELECT CAST('2015-01-07' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Budget Movie-Maker' as Product, CAST(3798.00 AS money) AS Sales, 9 as Quantity
UNION SELECT CAST('2015-01-08' AS date) as SalesDate, 'Camcorders' as Subcategory,
    'Budget Movie-Maker' as Product, CAST(10400.00 AS money) AS Sales, 13 as Quantity
UNION SELECT CAST('2015-01-09' AS date) as SalesDate, 'Camcorders' as Subcategory,
    'Budget Movie-Maker' as Product, CAST(3000.00 AS money) AS Sales, 60 as Quantity
UNION SELECT CAST('2015-01-10' AS date) as SalesDate, 'Digital' as Subcategory,
    'Carrying Case' as Product, CAST(10836.00 AS money) AS Sales, 84 as Quantity
UNION SELECT CAST('2015-01-07' AS date) as SalesDate, 'Digital' as Subcategory,
    'Slim Digital' as Product, CAST(2550.00 AS money) AS Sales, 17 as Quantity
UNION SELECT CAST('2015-01-04' AS date) as SalesDate, 'Digital' as Subcategory,
    'Slim Digital' as Product, CAST(8357.80 AS money) AS Sales, 44 as Quantity
UNION SELECT CAST('2015-01-08' AS date) as SalesDate, 'Digital SLR' as Subcategory,
    'Slim Digital' as Product, CAST(18530.00 AS money) AS Sales, 34 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Digital SLR' as Subcategory,
    'Slim Digital' as Product, CAST(26576.00 AS money) AS Sales, 88 as Quantity

```

- On the query designer toolbar, click Run (!).

The query runs and displays the result set for the fields **SalesDate**, **Subcategory**, **Product**, **Sales**, and **Quantity**.

- Click **Next**.
- On the **Arrange fields** page, drag **Sales** to **Values**.
- Sales** is aggregated by the Sum function. The value is [Sum(Sales)].
- Drag **Product** to **Row groups**.
- Drag **SalesDate** to **Column groups**.



- Click **Next**.
- On the **Choose the layout** page, under **Options**, verify that **Show subtotals and grand totals** is selected.

The wizard Preview pane displays a table with three rows. When you run the report, each row will display in the following way:

- The first row will appear once for the table to show column headings.

- The second row will repeat once for each product and display the product name, total per day, and line total.
- The third row will appear once for the table to display the grand totals.



11. Click **Next**.

12. Click **Finish**.

13. The table is added to the design surface. The table has three columns and three rows.

Look in the Grouping pane. If you can't see the Grouping pane, on the **View** menu, click **Grouping**. The Row Groups pane shows one row group: **Product**. The Column Groups pane shows one column group: **SalesDate**. Detail data is all the data that is retrieved by the dataset query.

14. Click **Run** to preview the report.

2a. Format Data as Currency

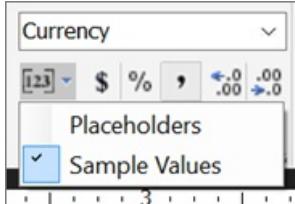
By default, the summary data for the **Sales** field displays a general number. Format it to display the number as currency. Toggle **Placeholder Styles** to display formatted text boxes and placeholder text as sample values.

- Click **Design** to switch to design view.
- Click the cell in the second row (under the column headings row) in the **SalesDate** column. Hold down the **Ctrl** key and select all cells that contain `[Sum(Sales)]`.

- On the **Home** tab > **Number** group, click **Currency**. The cells change to show the formatted currency.

Product		SalesDate	Total
[Product]	[\$12,345.00]	[\$12,345.00]	
Total	[\$12,345.00]	[\$12,345.00]	

If your regional setting is English (United States), the default sample text is **[\$12,345.00]**. If you do not see an example currency value, in the **Numbers** group, click **Placeholder Styles > Sample Values**.



2b. (Optional) Format Data as Dates

By default, the **SalesDate** field displays both date and time information. You can format them to display only the date.

1. Click the cell that contains **[SalesDate]**.
2. On the **Home** tab > **Number** group > **Date**.

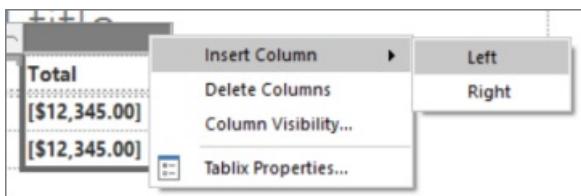
The cell displays the example date **[1/31/2000]**.

3. Click **Run** to preview the report.

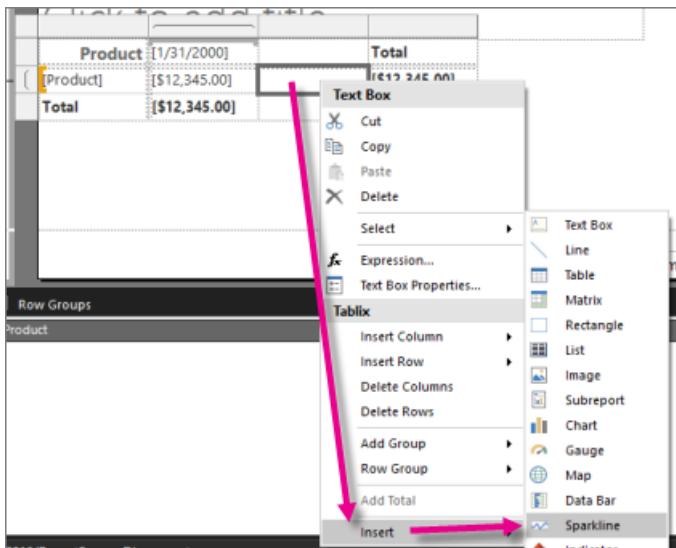
The **SalesDate** values display in the default date format, and the summary values for **Sales** display as currency.

3. Add a Sparkline

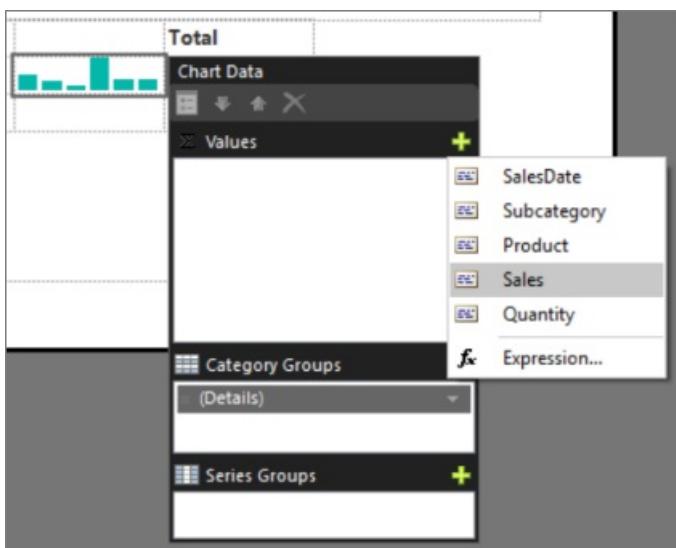
1. Click **Design** to return to design view.
2. Select the Total column in your table.
3. Right-click, point to **Insert Column**, and then click **Left**.



4. In the new column, right-click the cell in the **[Product]** row > **Insert > Sparkline**.



5. In the **Select Sparkline Type** dialog box, make sure the first sparkline in the **Column** row is selected, then click **OK**.
6. Click the sparkline to show the Chart Data pane.
7. Click the plus (+) sign in the Values box, then click **Sales**.



The values in the **Sales** field are now the values for the sparkline.

8. Click the plus (+) sign in the Category Groups box, then click **SalesDate**.
9. Click **Run** to preview your report.

Note that the bars in the sparkline charts don't line up with each other. There are only four bars in the second row of data, so the bars are wider than the bars in the first row, which has six. You can't compare values for each product per day. They need to line up.

Also, for each row the tallest bar is the height of the row. This is misleading, too, because the largest values for each row are not equal: the largest value for Budget Movie-Maker is \$10,400, but for Slim Digital it's \$26,576 - more than twice as large. And yet the largest bars in those two rows are about the same height. All the sparklines need to use the same scale.

1/9/2015	1/10/2015	Sparkline	Total
\$3,000.00	\$7,234.50		\$26,268.50
\$1,147.50			\$30,330.10
			\$57,393.80
\$3,000.00	\$8,382.00		\$113,992.40

4. Align the Sparklines Vertically and Horizontally

Sparklines are hard to read when they don't all use the same measurements. Both the horizontal and vertical axes for each need to match the rest.

1. Click **Design** to return to design view.
2. Right-click the sparkline and click **Vertical Axis Properties**.
3. Check the **Align axes in** check box. Tablix1 is the only option in the list.
This sets the height of the bars in each sparkline relative to the others.
4. Click **OK**.
5. Right-click the sparkline and click **Horizontal Axis Properties**.
6. Check the **Align axes in** check box. Tablix1 is the only option in the list.

This sets the width of the bars in each sparkline relative to the others. If some sparklines have fewer bars than others, then those sparklines will have blank spaces for the missing data.

7. Click **OK**.
8. Click **Run** to preview your report again.

Now all the bars in each sparkline align with the bars in the other sparklines, and the heights are relative.

1/9/2015	1/10/2015	Sparkline	Total
\$3,000.00	\$7,234.50		\$26,268.50
\$1,147.50			\$30,330.10
			\$57,393.80
\$3,000.00	\$8,382.00		\$113,992.40

7. (Optional) Change Column Widths

By default, each cell in a table contains a text box. A text box expands vertically to accommodate text when the page is rendered. In the rendered report, each row expands to the height of the tallest rendered text box in the row. The height of the row on the design surface has no affect on the height of the row in the rendered report.

To reduce the amount of vertical space each row takes, expand the column width to accommodate the expected contents of the text boxes in the column on one line.

To change the width of columns

1. Click **Design** to return to design view.
2. Click the table so gray bars appear above and next to the table. Those are the column and row handles.
3. Point to the line between column handles so that the cursor changes into a double arrow. Drag the **Product** column so that the product name displays on one line.
4. Click **Run** to preview your report and see if you made it wide enough.

8. (Optional) Add a Report Title

A report title appears at the top of the report. You can place the report title in a report header or if the report does not use one, in a text box at the top of the report body. In this tutorial, you will use the text box that is automatically placed at the top of the report body.

The text can be further enhanced by applying different font styles, sizes, and colors to phrases and individual characters of the text. For more information, see [Format Text in a Text Box \(Report Builder and SSRS\)](#).

To add a report title

1. On the design surface, click **Click to add title**.
2. Type **Sales by Date**, and then click outside the text box.
3. Select the text box that contains **Product Sales**.
4. On the Home tab > **Font** group > for **Color**, select **Teal**.
5. Select **Bold**.
6. Click **OK**.

9. Save the Report

Save the report to a report server or your computer. If you do not save the report to the report server, a number of Reporting Services features such as report parts and subreports are not available.

To save the report on a report server

1. From the **Report Builder** button, click **Save As**.
2. Click **Recent Sites and Servers**.
3. Select or type the name of the report server where you have permission to save reports.

The message "Connecting to report server" appears. When the connection is complete, you see the contents of the report folder that the report server administrator specified as the default location for reports.

4. In **Name**, replace the default name with **Product Sales**.
5. Click **Save**.

The report is saved to the report server. The name of report server that you are connected to appears in the status bar at the bottom of the window.

To save the report on your computer

1. From the **Report Builder** button, click **Save As**.
2. Click **Desktop**, **My Documents**, or **My computer**, and browse to the folder where you want to save the report.
3. In **Name**, replace the default name with **Product Sales**.
4. Click **Save**.

Next Steps

This concludes the tutorial for creating a table report with sparkline charts. For more information about sparklines, see [Sparklines and Data Bars](#).

More questions? Try asking the Reporting Services forum

Tutorial: Adding a KPI to Your Report (Report Builder)

11/30/2018 • 12 minutes to read • [Edit Online](#)

In this Report Builder tutorial, you add a key performance indicator (KPI) to a Reporting Services paginated report.

KPIs are measurable values with business significance. In this scenario, the sales summary by product subcategories is the KPI. The current state of the KPI is shown with colors, gauges, and indicators.

The following illustration is similar to the report you will create.

Product Sales KPIs						
Sales Date	Subcategory	Product	Quantity	Sales	Linear KPI	Stoplight KPI
1/5/2015	Accessories	Carrying Case	68	\$16,996.60		
		Mini Battery Charger	44	\$1,056.00		
			112	\$18,052.60	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
	Digital	Slim Digital	44	\$8,357.80	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
	Total		156	\$26,410.40		
1/6/2015	Accessories	Telephoto Conversion Lens	18	\$1,380.00		
		Tripod	18	\$1,350.00		
		USB Cable	26	\$780.00	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
			62	\$3,510.00	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
	Total		62	\$3,510.00		
1/7/2015	Digital	Compact Digital	84	\$10,836.00	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
			84	\$10,836.00	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
		Digital SLR	88	\$26,576.00	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
	Total		172	\$37,412.00		
	1/8/2015	Budget Movie-Maker	9	\$3,798.00	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
	Digital	Consumer Digital	17	\$2,550.00	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
	Total		26	\$6,348.00		
1/9/2015	Camcorders	Business Videographer	13	\$10,400.00	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
			13	\$10,400.00	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
		Digital SLR	34	\$18,530.00	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
	Total		47	\$28,930.00		
	1/10/2015	Social Videographer	60	\$3,000.00	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
	Total		60	\$3,000.00		
1/11/2015	Accessories	Lens Adapter	17	\$1,147.50	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
			17	\$1,147.50	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
		Digital	39	\$7,234.50	<div style="width: 100%;"><div style="width: 100%;"> </div></div>	●
	Total		56	\$8,382.00		
	Total		579	\$113,992.40		

NOTE

In this tutorial, the steps for the wizard are consolidated into two procedures: one to create the dataset and one to create a table. For step-by-step instructions about how to browse to a report server, choose a data source, create a dataset, and run the wizard, see the first tutorial in this series: [Tutorial: Creating a Basic Table Report \(Report Builder\)](#).

Estimated time to complete this tutorial: 15 minutes.

Requirements

For information about requirements, see [Prerequisites for Tutorials \(Report Builder\)](#).

1. Create a Table Report and Dataset from the Table or Matrix Wizard

In this section, you choose a shared data source, create an embedded dataset, and display the data in a table.

To create a table with an embedded dataset

1. [Start Report Builder](#) either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the left pane, verify that **New Report** is selected.
3. In the right pane, click **Table or Matrix Wizard**.
4. On the **Choose a dataset** page, click **Create a dataset**.
5. Click **Next**.
6. On the **Choose a connection to a data source** page, select an existing data source or browse to the report server and select a data source. If there no data source is available or you do not have access to a report server, you can use an embedded data source instead. For more information, see [Tutorial: Creating a Basic Table Report \(Report Builder\)](#).
7. Click **Next**.
8. On the **Design a query** page, click **Edit as Text**.
9. Copy and paste the following query into the query pane:

NOTE

In this tutorial, the query contains the data values, so that it does not need an external data source. This makes the query quite long. In a business environment, a query would not contain the data. This is for learning purposes only.

```

SELECT CAST('2015-01-05' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Carrying Case' as Product, CAST(16996.60 AS money) AS Sales, 68 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Tripod' as Product, CAST(1350.00 AS money) AS Sales, 18 as Quantity
UNION SELECT CAST('2015-01-11' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Lens Adapter' as Product, CAST(1147.50 AS money) AS Sales, 17 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Mini Battery Charger' as Product, CAST(1056.00 AS money) AS Sales, 44 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Telephoto Conversion Lens' as Product, CAST(1380.00 AS money) AS Sales, 18 as Quantity
UNION SELECT CAST('2015-01-06' AS date) as SalesDate, 'Accessories' as Subcategory,
    'USB Cable' as Product, CAST(780.00 AS money) AS Sales, 26 as Quantity
UNION SELECT CAST('2015-01-08' AS date) as SalesDate, 'Accessories' as Subcategory,
    'Budget Movie-Maker' as Product, CAST(3798.00 AS money) AS Sales, 9 as Quantity
UNION SELECT CAST('2015-01-09' AS date) as SalesDate, 'Camcorders' as Subcategory,
    'Business Videographer' as Product, CAST(10400.00 AS money) AS Sales, 13 as Quantity
UNION SELECT CAST('2015-01-10' AS date) as SalesDate, 'Camcorders' as Subcategory,
    'Social Videographer' as Product, CAST(3000.00 AS money) AS Sales, 60 as Quantity
UNION SELECT CAST('2015-01-11' AS date) as SalesDate, 'Digital' as Subcategory,
    'Advanced Digital' as Product, CAST(7234.50 AS money) AS Sales, 39 as Quantity
UNION SELECT CAST('2015-01-07' AS date) as SalesDate, 'Digital' as Subcategory,
    'Compact Digital' as Product, CAST(10836.00 AS money) AS Sales, 84 as Quantity
UNION SELECT CAST('2015-01-08' AS date) as SalesDate, 'Digital' as Subcategory,
    'Consumer Digital' as Product, CAST(2550.00 AS money) AS Sales, 17 as Quantity
UNION SELECT CAST('2015-01-05' AS date) as SalesDate, 'Digital' as Subcategory,
    'Slim Digital' as Product, CAST(8357.80 AS money) AS Sales, 44 as Quantity
UNION SELECT CAST('2015-01-09' AS date) as SalesDate, 'Digital SLR' as Subcategory,
    'SLR Camera 35mm' as Product, CAST(18530.00 AS money) AS Sales, 34 as Quantity
UNION SELECT CAST('2015-01-07' AS date) as SalesDate, 'Digital SLR' as Subcategory,
    'SLR Camera' as Product, CAST(26576.00 AS money) AS Sales, 88 as Quantity

```

10. On the query designer toolbar, click Run (!).

11. Click **Next**.

2. Organize Data and Choose Layout in the Wizard

The Table or Matrix wizard provides a starting design in which to display data. The preview pane in the wizard helps you to visualize the result of grouping data before you complete the table or matrix design.

To organize data into groups and choose a layout

1. On the Arrange fields page, drag Product to **Values**.

2. Drag Quantity to **Values** and place below Product.

Quantity is summarized with the Sum function, the default function to summarize numeric fields.

3. Drag Sales to **Values** and place below Quantity.

Steps 1, 2, and 3 specify the data to display in the table.

4. Drag SalesDate to **Row groups**.

5. Drag Subcategory to **Row groups** and place below SalesDate.

Steps 4 and 5 organize the values for the fields first by date, and then by all sales for that date.

6. Click **Next**.

When you run the report, the table displays each date, all orders for each date, and all products, quantities, and sales totals for each order.

7. On the Choose the Layout page, under **Options**, verify that **Show subtotals and grand totals** is selected.

8. Verify that **Blocked, subtotal below** is selected.

9. Clear the option **Expand/collapse groups**.

In this tutorial, the report you create does not use the drilldown feature that lets a user expand a parent group hierarchy to display child group rows and detail rows.

10. Click **Next**.

11. Click **Finish**.

The table is added to the design surface. The table has five columns and five rows. The Row Groups pane shows three row groups: SalesDate, Subcategory, and Details. Detail data is all the data that is retrieved by the dataset query. The Column Groups pane is empty.

The screenshot shows a report design interface. At the top, there's a placeholder text 'Click to add title'. Below it is a table with five columns: Sales Date, Subcategory, Product, Quantity, and Sales. The table has five rows: three summary rows ('Total' for Sales Date, Subcategory, and Product) and two detail rows. The Row Groups pane at the bottom shows 'SalesDate', 'Subcategory', and '(Details)' grouped under 'Row Groups'. The Column Groups pane is empty.

12. Click **Run** to preview the report.

For each product that is sold on a specific date, the table displays the product name, the quantity sold, and the sales total. The data is organized first by sales date and then by subcategory.

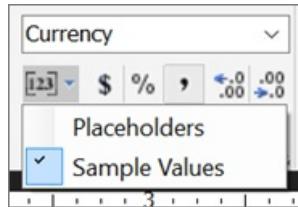
Sales Date	Subcategory	Product	Quantity	Sales
1/5/2015 12:00:00 AM	Accessories	Carrying Case	68	16996.6000
		Mini Battery Charger	44	1056.0000
			112	18052.6000
	Digital	Slim Digital	44	8357.8000
			44	8357.8000
	Total		156	26410.4000
1/6/2015 12:00:00 AM	Accessories	Telephoto Conversion Lens	18	1380.0000
		Tripod	18	1350.0000
		USB Cable	26	780.0000
			62	3510.0000
	Total		62	3510.0000
1/7/2015 12:00:00 AM	Digital	Compact Digital	84	10836.0000
			84	10836.0000
	Digital SLR	SLR Camera	88	26576.0000
			88	26576.0000
	Total		172	37412.0000

Format dates and currency

Let's make the columns wider and set the format for the dates and currency.

1. Click **Design** to go back to Design view.
2. The Product names could use more space. To make the Product column wider, select the whole table and drag the right edge of the column handle at the top of the Product column.
3. Press the Ctrl key, then select the four cells that contain [Sum(Sales)].
4. On the **Home** tab > **Number** > **Currency**. The cells change to show the formatted currency.

If your regional setting is English (United States), the default sample text is [\$12,345.00]. If you don't see an example currency value, in the **Numbers** group, click **Placeholder Styles** > **Sample Values**.



5. (Optional) On the **Home** tab, in the **Number** group, click the **Decrease Decimals** button twice to display dollar figures with no cents.
6. Click the cell that contains [SalesDate].
7. In the **Number** group > **Date**.

The cell displays the example date [1/31/2000].

8. Click **Run** to preview the report.

Sales Date	Subcategory	Product	Quantity	Sales
1/5/2015	Accessories	Carrying Case	68	\$16,996.60
		Mini Battery Charger	44	\$1,056.00
			112	\$18,052.60
	Digital	Slim Digital	44	\$8,357.80
			44	\$8,357.80
	Total		156	\$26,410.40
1/6/2015	Accessories	Telephoto Conversion Lens	18	\$1,380.00
		Tripod	18	\$1,350.00
		USB Cable	26	\$780.00
			62	\$3,510.00
	Total		62	\$3,510.00
1/7/2015	Digital	Compact Digital	84	\$10,836.00
			84	\$10,836.00
	Digital SLR	SLR Camera	88	\$26,576.00
			88	\$26,576.00

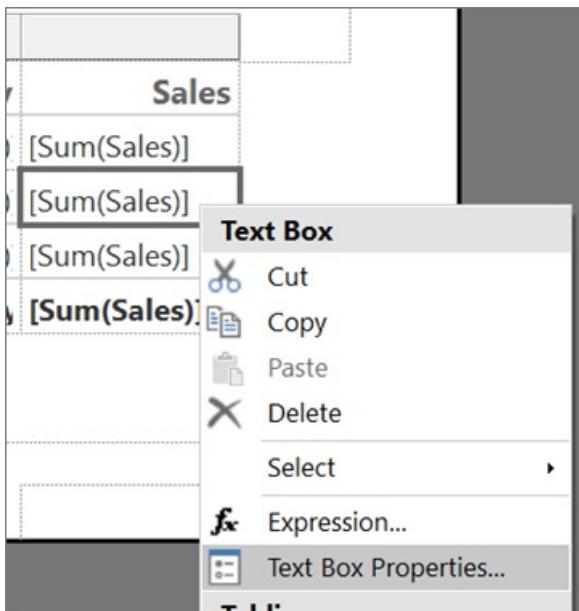
3. Use Background Colors to Display a KPI

Background colors can be set to an expression that is evaluated when you run the report.

To display the present state of a KPI by using background colors

1. In the table, right-click the second [Sum(Sales)] cell (the subtotal row that displays the sales for a subcategory), then click **Text Box Properties**.

Make sure you've selected the cell, not the text in the cell, to view **Text Box Properties**.



2. On the **Fill** tab, click the **fx** button next to **Fill color** and enter the following expression in the **Set expression for: BackgroundColor** field:

```
=IIF(Sum(Fields!Sales.Value) >= 5000 , "Lime", IIF(Sum(Fields!Sales.Value) < 2500, "Red","Yellow"))
```

This changes the background color to "Lime" green for each cell with an aggregated sum for **[Sum(Sales)]** greater than or equal to 5000. Values of **[Sum(Sales)]** between 2500 and 5000 are "Yellow". Values less than 2500 are "Red".

3. Click **OK**.
4. Click **Run** to preview the report.

In the subtotal row that displays the sales for a subcategory, the background color of the cell is red, yellow, or green depending on value of the sales sum.

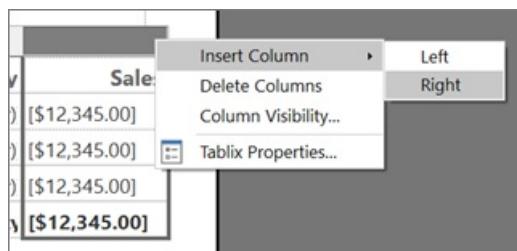
Sales Date	Subcategory	Product	Quantity	Sales
1/5/2015	Accessories	Carrying Case	68	\$16,996.60
		Mini Battery Charger	44	\$1,056.00
	Digital		112	\$18,052.60
		Slim Digital	44	\$8,357.80
1/6/2015	Accessories		44	\$8,357.80
		Telephoto Conversion Lens	18	\$1,380.00
		Tripod	18	\$1,350.00
		USB Cable	26	\$780.00
1/7/2015	Digital		62	\$3,510.00
		Compact Digital	84	\$10,836.00
			84	\$10,836.00
		SLR Camera	88	\$26,576.00
1/11/2015	Digital		88	\$26,576.00
		Total	172	\$37,412.00
			60	\$3,000.00
		Total	60	\$3,000.00
Total	Accessories	Lens Adapter	17	\$1,147.50
			17	\$1,147.50
	Digital	Advanced Digital	39	\$7,234.50
			39	\$7,234.50
	Total		56	\$8,382.00
Total			579	\$113,992.40

4. Display a KPI by Using a Gauge

A gauge depicts a single value in a dataset. This tutorial uses a horizontal linear gauge because its shape and simplicity make it easy to read, even in when it is small and within a table cell. For more information, see [Gauges \(Report Builder and SSRS\)](#).

To display the present state of a KPI using a gauge

1. Switch back to Design view.
2. In the table, right-click the column handle for the Sales column > **Insert Column > Right**. A new column is added to the table.

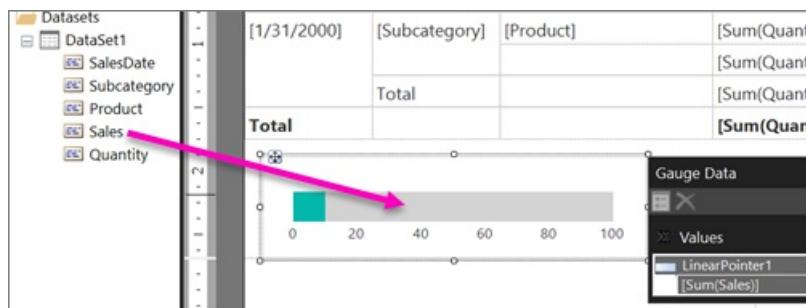


3. Type **Linear KPI** in the column heading.
4. On the **Insert** tab > **Data Visualizations** > **Gauge**, and then click the design surface outside the table.
5. In the **Select Gauge Type** dialog box, select the first linear gauge type, **Horizontal**.
6. Click **OK**.

A gauge is added to the design surface.

7. From the dataset in the Report Data pane, drag the **Sales** field to the gauge. The **Gauge Data** pane opens.

When you drop the `Sales` field onto the gauge, it goes to the **Values** list and is aggregated by using the built-in Sum function.



8. In the **Gauge Data** pane, click the arrow next to **LinearPointer1 > Pointer Properties**.
9. In the **Linear Pointer Properties** dialog box > **Pointer Options** tab > **Pointer Type**, make sure **Bar** is selected.
10. Click **OK**.
11. Right-click the scale in the gauge and click **Scale Properties**.
12. In the **Linear Scale Properties** dialog box > **General** tab, set **Maximum** to 25000.

NOTE

Instead of a constant such as 25000, you can use an expression to dynamically calculate the value of the **Maximum** option. The expression would use the aggregate of aggregate feature and look similar to the expression

```
=Max(Sum(Fields!Sales.value), "Tablix1") .
```

13. On the **Labels** tab, check **Hide scale labels**.
14. Click **OK**.
15. Drag the gauge inside the table to the second empty cell in the Linear KPI column, in the row that displays the subtotal sales for the `Subcategory` field, next to the field where you added the background color formula.

NOTE

You might have to resize the column so the horizontal linear gauge fits into the cell. To resize the column, select the table and drag the column handles. The report design surface resizes to fit the table.

16. Click **Run** to preview the report.

The horizontal length of the green bar in the gauge changes depending on the value of the KPI.

Sales Date	Subcategory	Product	Quantity	Sales	Linear KPI	
1/5/2015	Accessories	Carrying Case	68	\$16,996.60		
		Mini Battery Charger	44	\$1,056.00		
	Digital		112	\$18,052.60		
		Slim Digital	44	\$8,357.80		
Total			156	\$26,410.40		
1/6/2015	Accessories	Telephoto Conversion Lens	18	\$1,380.00		
		Tripod	18	\$1,350.00		
		USB Cable	26	\$780.00		
	Total		62	\$3,510.00		
1/7/2015	Digital SLR		13	\$10,400.00		
		SLR Camera 35mm	34	\$18,530.00		
	Total		34	\$18,530.00		
			47	\$28,930.00		
1/10/2015	Camcorders	Social Videographer	60	\$3,000.00		
			60	\$3,000.00		
	Total		60	\$3,000.00		
1/11/2015	Accessories	Lens Adapter	17	\$1,147.50		
			17	\$1,147.50		
	Digital	Advanced Digital	39	\$7,234.50		
			39	\$7,234.50		
Total			56	\$8,382.00		
Total			579	\$113,992.40		

5. Display a KPI by Using an Indicator

Indicators are small simple gauges that communicate data values at a glance. Because of their size and simplicity, indicators are often used in tables and matrices. For more information, see [Indicators \(Report Builder and SSRS\)](#).

To display the present state of a KPI using an indicator

1. Switch to Design view.
2. In the table, right-click the column handle for the Linear KPI column that you added in the last procedure > **Insert Column > Right**. A new column is added to the table.
3. Type **Stoplight KPI** in the column heading.
4. Click the cell for the subcategory subtotal, next to the linear gauge you added in the last procedure.
5. On the **Insert** tab > **Data Visualizations** > double-click **Indicator**.
6. In the **Select Indicator Type** dialog box, under **Shapes**, select the first shape type, **3 Traffic Lights (Unrimmed)**.
7. Click **OK**.

The indicator is added to the cell in the new Stoplight KPI column.

8. Right-click the indicator and click **Indicator Properties**.
9. On the **Values and States** tab, in the **Value** box, select **[Sum(Sales)]**. Don't change any other options.

By default, data synchronization occurs across the data region and you see the value **Tablix1**, the name of the table data region in the report, in the **Synchronization scope** box.

In this report, you can also change the scope of an indicator placed in the cell of the subcategory subtotal to synchronize across the SalesDate field.

10. Click **OK**.

11. Click **Run** to preview the report.

Sales Date	Subcategory	Product	Quantity	Sales	Linear KPI	Stoplight KPI	
1/5/2015	Accessories	Carrying Case	68	\$16,996.60			
		Mini Battery Charger	44	\$1,056.00			
			112	\$18,052.60	<div style="width: 150px;"><div style="width: 100%; background-color: #00A0A0;"></div></div>	●	
	Digital	Slim Digital	44	\$8,357.80	<div style="width: 100px;"><div style="width: 100%; background-color: #00A0A0;"></div></div>	●	
			44	\$8,357.80	<div style="width: 100px;"><div style="width: 100%; background-color: #00A0A0;"></div></div>	●	
Total			156	\$26,410.40			
1/6/2015	Accessories	Telephoto Conversion Lens	18	\$1,380.00			
		Tripod	18	\$1,350.00			
		USB Cable	26	\$780.00	<div style="width: 150px;"><div style="width: 100%; background-color: #FFFF00;"></div></div>	●	
	Digital SLR		62	\$3,510.00	<div style="width: 100px;"><div style="width: 100%; background-color: #FFFF00;"></div></div>	●	
		SLR Camera 35mm	34	\$18,530.00	<div style="width: 150px;"><div style="width: 100%; background-color: #00A0A0;"></div></div>	●	
Total			47	\$28,930.00			
1/10/2015	Camcorders	Social Videographer	60	\$3,000.00			
			60	\$3,000.00	<div style="width: 100px;"><div style="width: 100%; background-color: #FFFF00;"></div></div>	●	
	Total		60	\$3,000.00			
1/11/2015	Accessories	Lens Adapter	17	\$1,147.50	<div style="width: 100px;"><div style="width: 100%; background-color: #FF0000;"></div></div>	●	
			17	\$1,147.50	<div style="width: 100px;"><div style="width: 100%; background-color: #FF0000;"></div></div>	●	
	Digital	Advanced Digital	39	\$7,234.50	<div style="width: 150px;"><div style="width: 100%; background-color: #00A0A0;"></div></div>	●	
			39	\$7,234.50	<div style="width: 150px;"><div style="width: 100%; background-color: #00A0A0;"></div></div>	●	
Total			56	\$8,382.00			
Total			579	\$113,992.40			

6. Add a Report Title

A report title appears at the top of the report. You can place the report title in a report header or if the report does not use one, in a text box at the top of the report body. In this section, you use the text box that is automatically placed at the top of the report body.

You can further enhance the text by applying different font styles, sizes, and colors to phrases and individual characters of the text. For more information, see [Format Text in a Text Box \(Report Builder and SSRS\)](#).

To add a report title

1. On the design surface, click **Click to add title**.
2. Type **Product Sales KPIs**, and then click outside the text box.
3. Optionally, right-click the text box that contains **Product Sales KPI**, click **Text Box Properties**, and then on the Font tab select different font styles, sizes and colors.
4. Click **Run** to preview the report.

7. Save the Report

Save the report to a report server or your computer. If you do not save the report to the report server, a number of Reporting Services features such as report parts and subreports are not available.

To save the report on a report server

1. From the **Report Builder** button, click **Save As**.
2. Click **Recent Sites and Servers**.
3. Select or type the name of the report server where you have permission to save reports.

The message "Connecting to report server" appears. When the connection is complete, you see the contents

of the report folder that the report server administrator specified as the default location for reports.

4. In **Name**, replace the default name with **Product Sales KPI**.

5. Click **Save**.

The report is saved to the report server. The name of report server that you are connected to appears in the status bar at the bottom of the window.

To save the report on your computer

1. From the **Report Builder** button, click **Save As**.

2. Click **Desktop**, **My Documents**, or **My computer**, and browse to the folder where you want to save the report.

NOTE

If you do not have access to a report server, click **Desktop**, **My Documents**, or **My computer** and save the report to your computer.

1. In **Name**, replace the default name with **Product Sales KPI**.

2. Click **Save**.

Next Steps

You have successfully completed the Adding a KPI to Your Report tutorial. For more information, see:

- [Gauges](#)
- [Indicators](#)

See Also

- [Report Builder Tutorials](#)
- [Report Builder in SQL Server](#)

Tutorial: Map Report (Report Builder)

11/30/2018 • 19 minutes to read • [Edit Online](#)

In this Report Builder tutorial, you learn about map features you can use to display data on a geographic background in an Reporting Services paginated report.

Maps are based on spatial data that typically consists of points, lines, and polygons. For example, a polygon can represent the outline of a county, a line can represent a road, and a point can represent the location of a city. Each type of spatial data is displayed on a separate map layer as a set of map elements.

To vary the appearance of map elements, you specify a field that has values that match the map elements with analytical data from a dataset. You can also define rules that vary color, size, or other properties based on ranges of data.



In this tutorial, you build a map report that displays store locations in New York state counties.

NOTE

In this tutorial, the steps for the wizard are consolidated into two procedures: one to create the dataset and one to create a table. For step-by-step instructions about how to browse to a report server, choose a data source, create a dataset, and run the wizard, see the first tutorial in this series: [Tutorial: Creating a Basic Table Report \(Report Builder\)](#).

Estimated time to complete this tutorial: 30 minutes.

Requirements

For this tutorial, the report server must be configured to support Bing maps as a background. For more information, see [Plan for Map Report Support](#).

For information about other requirements, see [Prerequisites for Tutorials \(Report Builder\)](#).

1. Create a Map with a Polygon Layer from the Map Wizard

In this section, you add a map to your report from the map gallery. The map has one layer that displays the counties in New York state. The shape of each county is a polygon based on spatial data that is embedded in the map from the map gallery.

To add a map with the map wizard in a new report

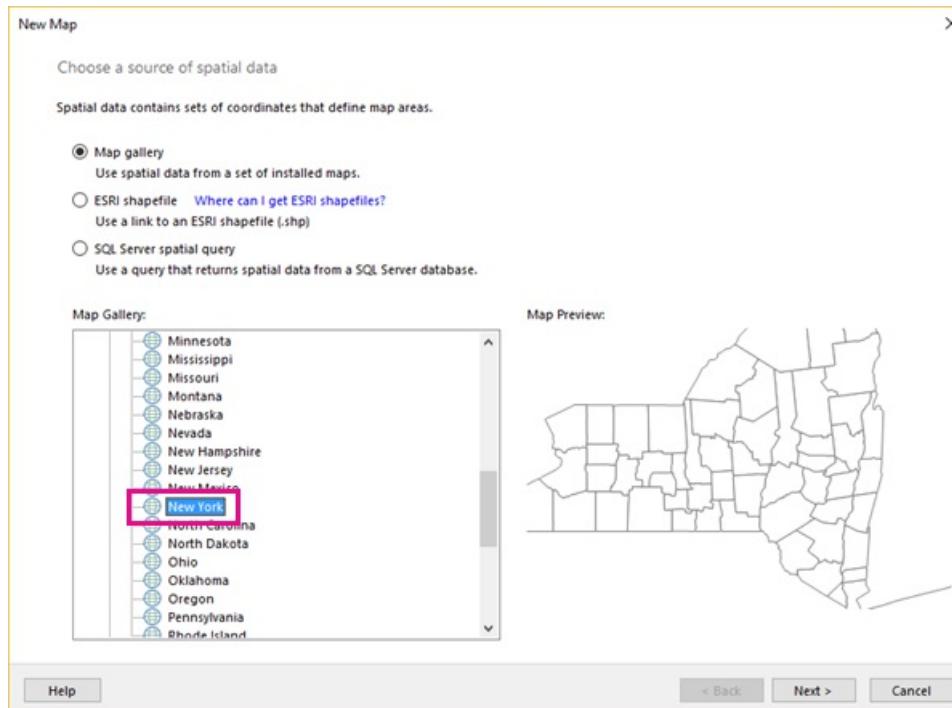
1. Start Report Builder either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the left pane, verify that **New Report** is selected.
3. In the right pane, click **Map Wizard**.
4. On the **Choose a source of spatial data** page, verify that **Map gallery** is selected.
5. In the Map Gallery box, expand **States by County** under **USA**, and click **New York**.

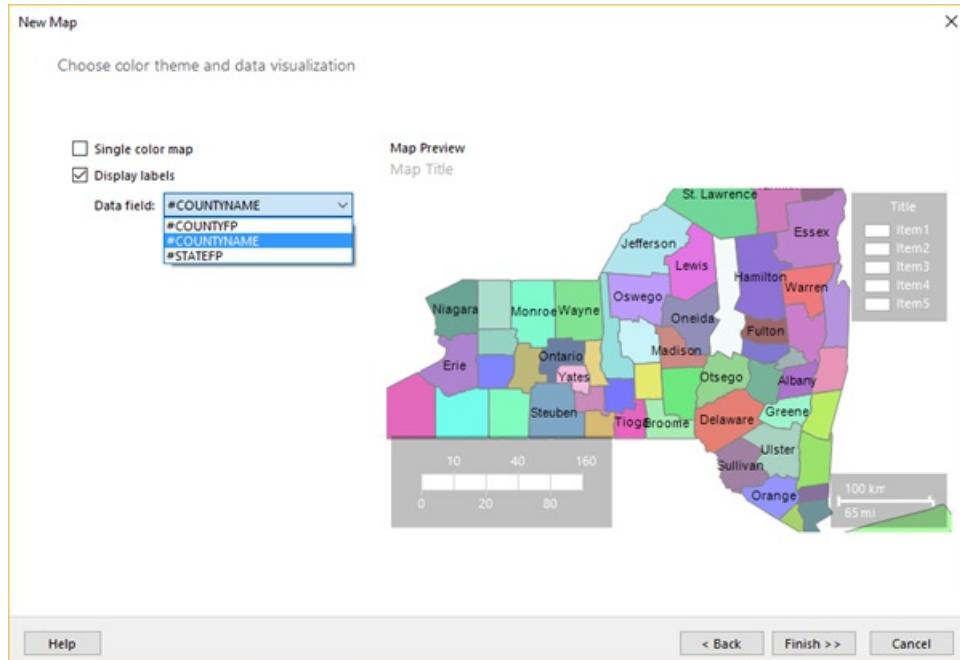
The Map Preview pane displays the New York county map.



6. Click **Next**.
7. On the **Choose spatial data and map view options** page, accept the defaults and click **Next**.

By default, map elements from a map gallery are automatically embedded in the report definition.
8. On the **Choose map visualization** page, verify **Basic Map** is selected, and click **Next**.
9. On the **Choose color theme and data visualization** page, select the **Display labels** option.
10. If it is selected, clear the **Single color map** option.
11. From the **Data field** drop-down list, click **#COUNTYNAME**. The Map Preview pane in the wizard displays the following items:
 - A title with the text **Map Title**.

- A map that displays counties in New York where each county is a different color and the county name appears wherever it fits over the county area.
- A legend that contains a title and a list of items 1 through 5.
- A color scale that contains values 0 to 160 and no color.
- A distance scale that displays kilometers (km) and miles (mi).



12. Click **Finish**.

The map is added to the design surface.

13. Select the "Map Title" text and type **Sales by Store** > ENTER.

14. Double-click the map to display the **Map Layers Pane**. The **Map Layers Pane** shows one polygon layer, **PolygonLayer1**, of layer type **Embedded**. Each county is an embedded map element on this layer.

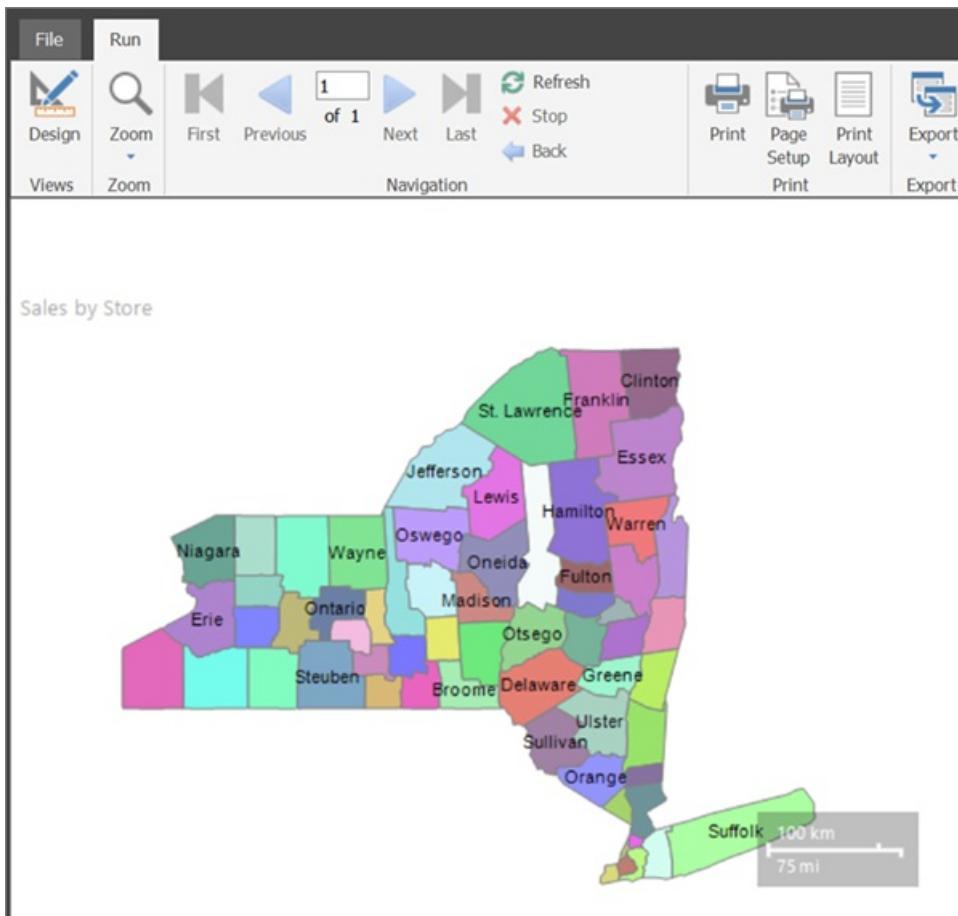
NOTE

If you don't see the **Map Layers** pane, it might be displayed outside your current view. Use the scroll bar at the bottom of the Design view window to change your view. Alternatively, in the **View** tab, clear the **Report Data** option to provide more design surface area.

15. Select the arrow next to **PolygonLayer1** > **Polygon properties**.

16. On the **Font** tab, change the color to **Dim Gray**.

17. On the **Home** tab > **Run** to preview the report.



The rendered report displays the map title, the map, and the distance scale. The counties are on a map polygon layer. Each county is a polygon that varies by color from a color palette, but the colors are not associated with any data. The distance scale displays distances in both kilometers and miles.

The map legend and color scale do not yet appear because there is no analytical data associated with each county. You will add analytical data later in this tutorial.

2. Add a Map Point Layer to Display Store Locations

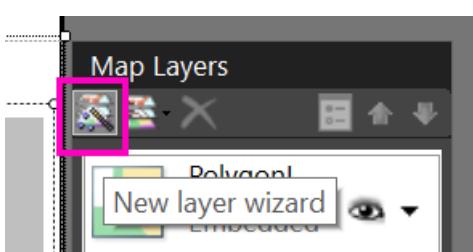
In this section, you use the map layer wizard to add a point layer that displays the locations of stores.

NOTE

In this tutorial, the query contains the data values, so it does not need an external data source. This makes the query quite long. In a business environment, a query would not contain the data. This is for learning purposes only.

To add a point layer based on a SQL Server spatial query

1. On the **Run** tab > **Design** to switch back to Design view.
2. Double-click the map to display the **Map Layers** pane. On the toolbar, click the **New layer wizard** button .



3. On the **Choose a source of spatial data** page, select **SQL Server spatial query**, and click **Next**.

4. On the **Choose a dataset with SQL Server spatial data** page, click **Add a new dataset with SQL Server spatial data > Next.**
5. On the **Choose a connection to a SQL Server spatial data source** page, select an existing data source or browse to the report server and select a data source.

NOTE

The data source you choose is unimportant, as long as you have adequate permissions. You will not be getting data from the data source. For more information, see [Alternative Ways to Get a Data Connection \(Report Builder\)](#).

6. Click **Next**.
7. On the **Design a Query** page, click **Edit as Text**.
8. Copy the following text and paste it in the query pane:

```

Select 114 as StoreKey, 'Contoso Albany Store' as StoreName, 1125 as SellingArea, 'Albany' as City,
'Albany' as County,
CAST(1000000 as money) as Sales, CAST('POINT(-73.7472924218681 42.6564617079878)' as geography) AS
SpatialLocation
UNION ALL SELECT 115 AS StoreKey, 'Contoso New York No.1 Store' AS StoreName, 500 as SellingArea, 'New
York' AS City, 'New York City' as County,
CAST('2000000' as money) as Sales, CAST('POINT(-73.9922069374483 40.7549638237402)' as geography) AS
SpatialLocation
UNION ALL Select 116 as StoreKey, 'Contoso Rochester No.1 Store' as StoreName, 462 as SellingArea,
'Rochester' as City, 'Monroe' as County,
CAST(3000000 as money) as Sales, CAST('POINT(-77.624041566786 43.1547066024338)' as geography) AS
SpatialLocation
UNION ALL Select 117 as StoreKey, 'Contoso New York No.2 Store' as StoreName, 700 as SellingArea, 'New
York' as City, 'New York City' as County,
CAST(4000000 as money) as Sales, CAST('POINT(-73.9712488 40.7830603)' as geography) AS SpatialLocation
UNION ALL Select 118 as StoreKey, 'Contoso Syracuse Store' as StoreName, 680 as SellingArea, 'Syracuse'
as City, 'Onondaga' as County,
CAST(5000000 as money) as Sales, CAST('POINT(-76.1349120532546 43.0610223535974)' as geography) AS
SpatialLocation
UNION ALL Select 120 as StoreKey, 'Contoso Plattsburgh Store' as StoreName, 560 as SellingArea,
'Plattsburgh' as City, 'Clinton' as County,
CAST(6000000 as money) as Sales, CAST('POINT(-73.4728622833178 44.7028831413324)' as geography) AS
SpatialLocation
UNION ALL Select 121 as StoreKey, 'Contoso Brooklyn Store' as StoreName, 1125 as SellingArea, 'Brooklyn'
as City, 'New York City' as County,
CAST(7000000 as money) as Sales, CAST('POINT (-73.9638533447143 40.6785123489351)' as geography) AS
SpatialLocation
UNION ALL Select 122 as StoreKey, 'Contoso Oswego Store' as StoreName, 500 as SellingArea, 'Oswego' as
City, 'Oswego' as County,
CAST(8000000 as money) as Sales, CAST('POINT(-76.4602850815536 43.4353224527794)' as geography) AS
SpatialLocation
UNION ALL Select 123 as StoreKey, 'Contoso Ithaca Store' as StoreName, 460 as SellingArea, 'Ithaca' as
City, 'Tompkins' as County,
CAST(9000000 as money) as Sales, CAST('POINT(-76.5001866085881 42.4310489934743)' as geography) AS
SpatialLocation
UNION ALL Select 124 as StoreKey, 'Contoso Buffalo Store' as StoreName, 700 as SellingArea, 'Buffalo' as
City, 'Erie' as County,
CAST(100000 as money) as Sales, CAST('POINT(-78.8784 42.8864)' as geography) AS SpatialLocation
UNION ALL Select 125 as StoreKey, 'Contoso Queens Store' as StoreName, 700 as SellingArea, 'Queens' as
City, 'New York City' as County,
CAST(500000 as money) as Sales, CAST('POINT(-73.7930979029883 40.7152781765927)' as geography) AS
SpatialLocation
UNION ALL Select 126 as StoreKey, 'Contoso Elmira Store' as StoreName, 680 as SellingArea, 'Elmira' as
City, 'Chemung' as County,
CAST(800000 as money) as Sales, CAST('POINT(-76.7397414783301 42.0736492742663)' as geography) AS
SpatialLocation
UNION ALL Select 127 as StoreKey, 'Contoso Poestenkill Store' as StoreName, 455 as SellingArea,
'Poestenkill' as City, 'Rensselaer' as County,
CAST(1500000 as money) as Sales, CAST('POINT(-73.5626737425063 42.6940551238618)' as geography) AS
SpatialLocation

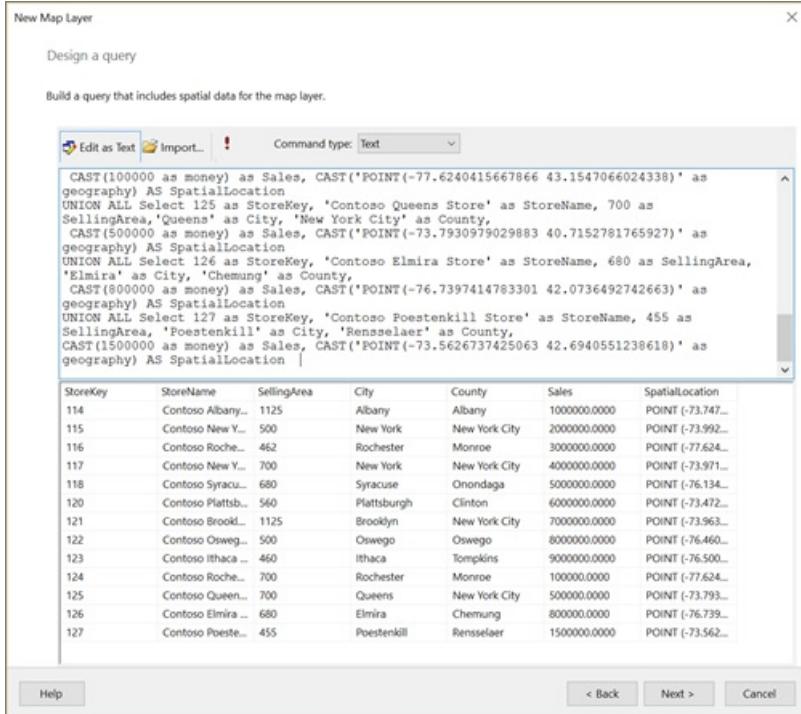
```

9. On the query designer toolbar, click **Run (!)**.

The result set contains seven columns representing a set of stores in New York State that sell consumer goods. Here's a list with explanations for the ones that may not be obvious:

- **StoreKey:** A store identifier.
- **StoreName.**
- **SellingArea:** The area available for product display, ranging from 455 square feet to 1125 square feet.
- **City.**
- **County.**
- **Sales:** Total sales.

- **SpatialLocation:** Location in longitude and latitude.



10. Click **Next**.

The report dataset named DataSet1 is created for you. After you complete the wizard, you can see its field collection in the Report Data pane.

11. On the **Choose a spatial data and map view options** page, verify that the **Spatial field** is **SpatialLocation** and that the **Layer type** is **Point**. Accept the other defaults on this page.

The map view displays circles to mark the location of each store.

12. Click **Next**.

13. On the Choose map visualization page, click **Bubble Map** for a map type that displays markers that vary in size, according to the data. Click **Next**.

14. On the **Choose the analytical dataset** page, click DataSet1, and click **Next**. This dataset contains both analytical data and spatial data that will be displayed on the new point layer.

15. On the **Choose color theme and data visualization** page, select **Use bubble sizes to visualize data**.

16. In **Data field**, select **[Sum(SellingArea)]** to vary bubble size by the size of the area a store sets aside to display the products.

17. Select **Display labels**, and in **Data field**, select **[City]**.

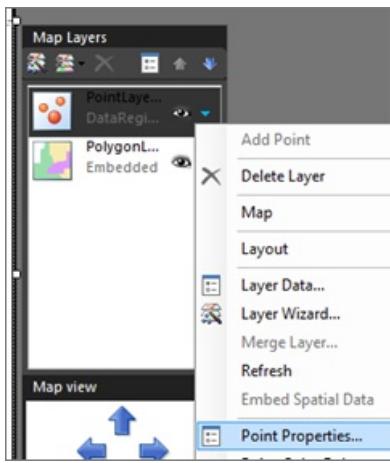
18. Click **Finish**.

The map layer is added to the report. The legend displays bubble sizes based on SellingArea values.

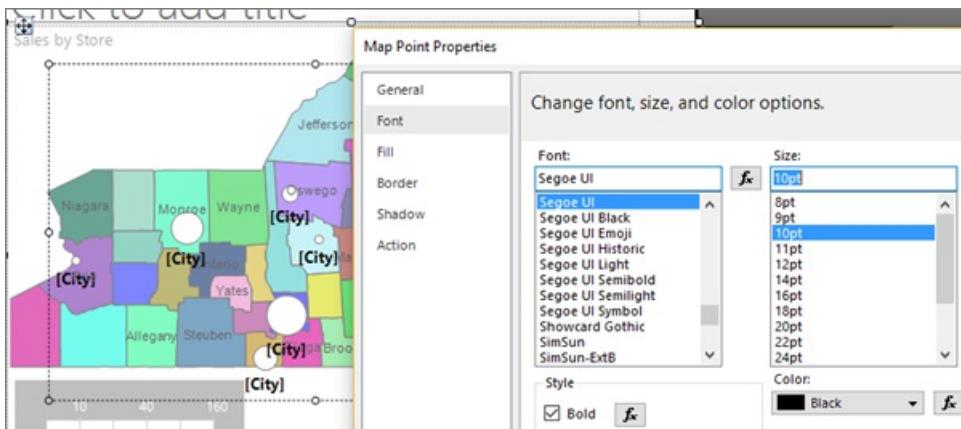
19. Double-click the map to display the **Map Layer** pane. The **Map Layer** pane displays a new layer, PointLayer1, with spatial data source type **DataRegion**.

20. Add a legend title. In the legend, select the text **Title**, type **Display Area (sq. ft.)** and press ENTER.

21. In the **Map Layers Pane**, click the arrow next to PointLayer1, and then click **Point Properties**.



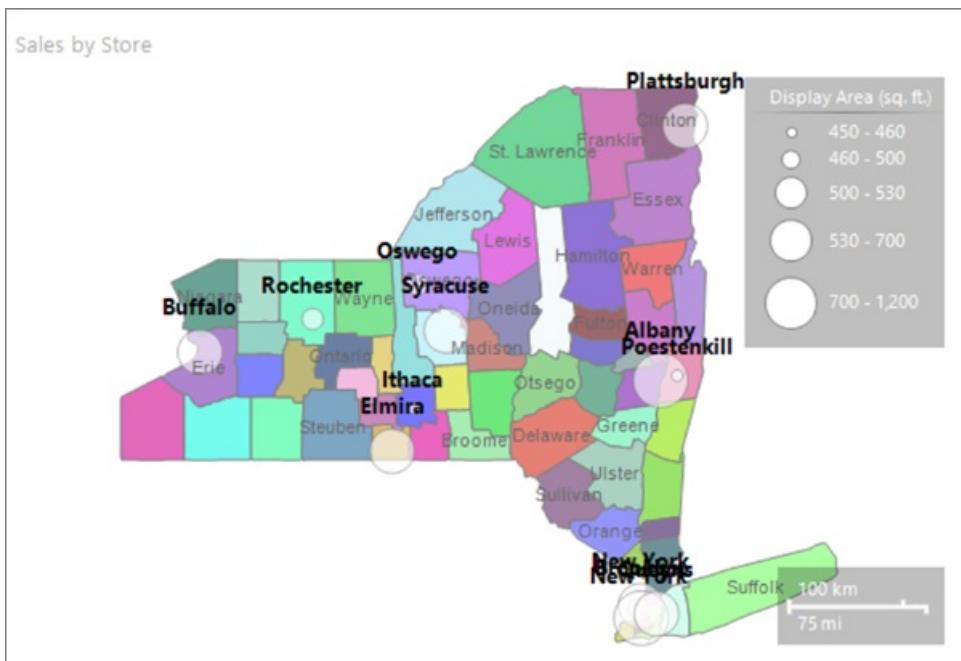
22. On the **Font** tab, make the style **Bold** and the size **10pt**.



23. On the **General** tab, select **Bottom** for **Placement**.

24. Click **OK**.

25. Click **Run** to preview the report.



The map displays the locations of stores in New York state. The marker size for each store is based on the display area. Five ranges of display area were automatically calculated for you.

3. Add a Map Line Layer to Display a Route

Use the map layer wizard to add a map layer that displays a route between two stores. In this tutorial, the path is created from three store locations. In a business application, the path might be the best route between stores.

To add a line layer to map

1. Switch to Design view.
2. Double-click the map to display the **Map Layer** pane. On the toolbar, click the **New layer wizard** button .
3. On the **Choose a source of spatial data** page, select **SQL Server spatial query** and click **Next**.
4. On the **Choose a dataset with SQL Server spatial data** page, click **Add a new dataset with SQL Server spatial data** and click **Next**.
5. On the **Choose a connection to a SQL Server spatial data source**, select the data source that you used in the first procedure.
6. Click **Next**.
7. On the **Design a Query** page, click **Edit as Text**. The query designer switches to text-based mode.
8. Paste the following text in the query pane:

```
SELECT N'Path' AS Name, CAST('LINESTRING
-76.5001866085881 42.4310489934743,
-76.4602850815536 43.4353224527794,
-73.4728622833178 44.7028831413324)' AS geography) as Route
```

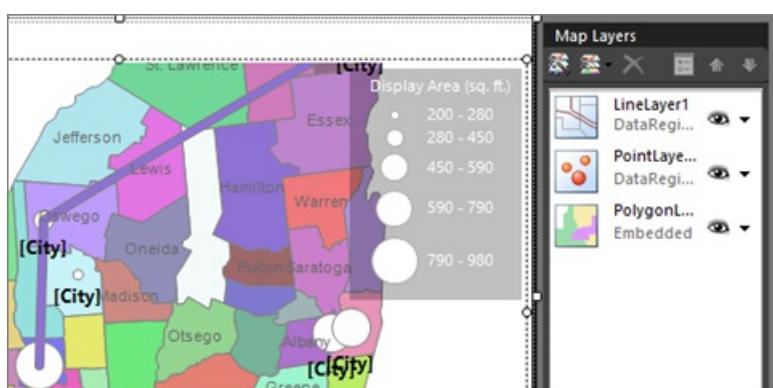
9. Click **Next**.

A path appears on the map that connects three stores.

10. On the **Choose spatial data and map view options** page, verify that the **Spatial field** is **Route** and that the **Layer type** is **Line**. Accept the other defaults.

The map view displays a path from a store in the northern part of New York state to a store in the southern part of New York state.

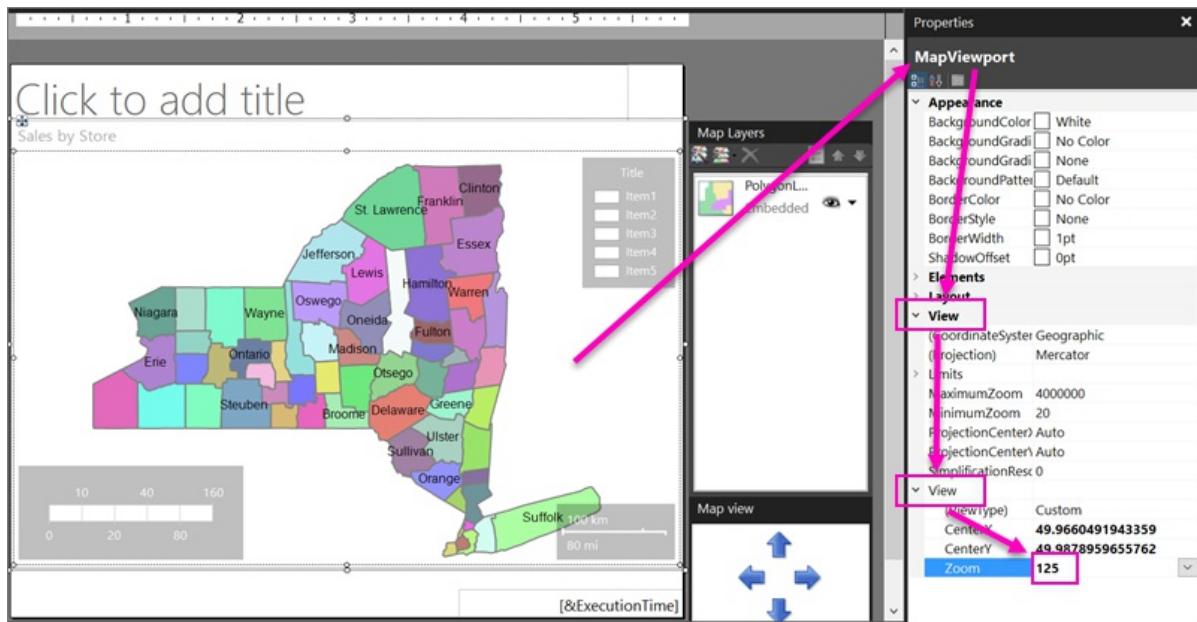
11. Click **Next**.
12. On the **Choose map visualization** page, click **Basic Line Map**, and then click **Next**.
13. On the **Choose color theme and data visualization**, select the option **Single color map**. The path appears as a single color based on the selected theme.
14. Click **Finish**.



The map displays a new line layer with spatial data source type **DataRegion**. In this example, the spatial data comes from a dataset but no analytical data is associated with the line.

Adjust the Zoom

1. If you can't see the whole state of New York, you can adjust the zoom. With the map selected, in the Properties pane you see **MapViewport** properties.
2. Expand the **View** section, then expand **View** so you can see the **Zoom** property. Set it to **125**.



This is the zoom percentage. At 125% you should see the whole state.

4. Add a Bing Maps Tile Background

In this section, you add a map layer that displays a Bing Maps tile background.

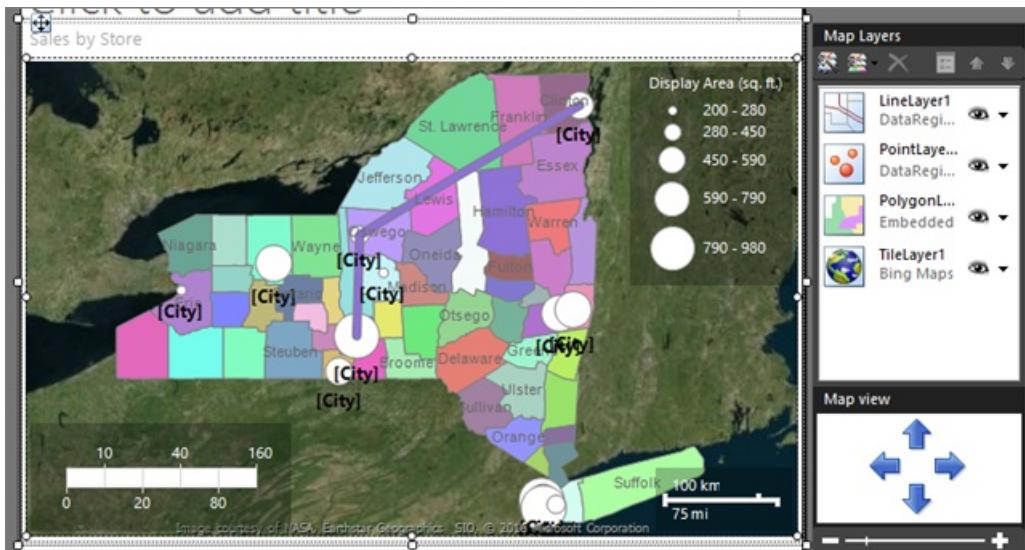
1. Switch to Design view.
2. Double-click the map to display the **Map Layer** pane. On the toolbar, click **Add Layer** .
3. From the drop-down list, click **Tile Layer**.

The last layer in the **Map Layer** pane is TileLayer1. By default, the tile layer displays the road map style.

NOTE

In the wizard, you can also add a tile layer on the **Choose spatial data and map view options** page. To do this, select **Add a Bing Maps background for this map view**. In a rendered report, the tile background displays Bing Maps tiles for the current map viewport center and zoom level.

4. Click the arrow next to TileLayer1 > **Tile Properties**.
5. On the **General** tab, under **Type**, select **Aerial**. The aerial view does not contain text.



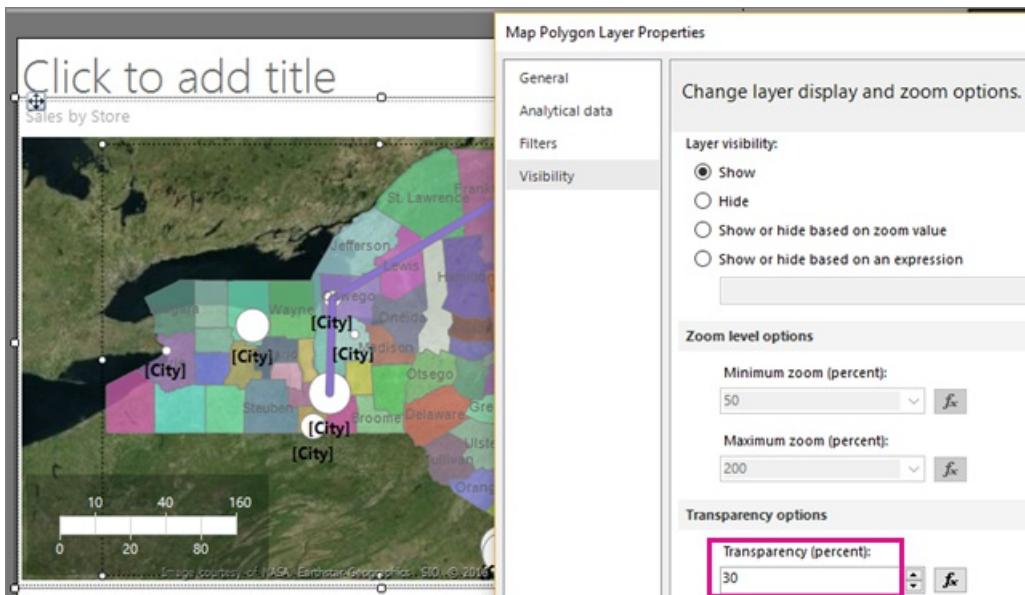
6. Click **OK**.

5. Make a Layer Transparent

In this section, to let the items on one layer show through another layer, you adjust the order and transparency of the layers for the effect that you want. You start with the first layer you created, PolygonLayer1.

1. Double-click the map to display the **Map Layer** pane.
2. Click the arrow next to PolygonLayer1 > **Layer Data**. The **Map Polygon Layer Properties** dialog box opens.
3. On the **Visibility** tab, under **Transparency (percent)**, type **30**.
4. Click **OK**.

The design surface displays the counties as semi-transparent.



6. Vary County Color Based on Sales

Each county on the polygon layer has a different color because the report processor automatically assigns a color value from the color palette based on the theme that you chose on the last page of the map wizard.

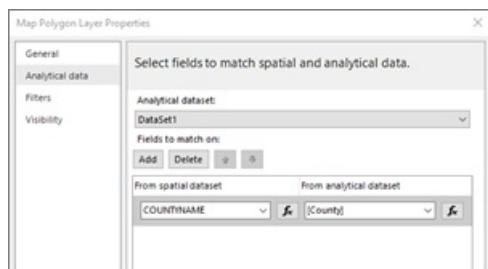
In this section, you specify a color rule to associate specific colors with a range of store sales for each county. The colors red-yellow-green indicate relative high-middle-low sales. Format the color scale to show currency. Display

the annual sales ranges in a new legend. For counties that do not contain stores, use no color to show that there is no associated data.

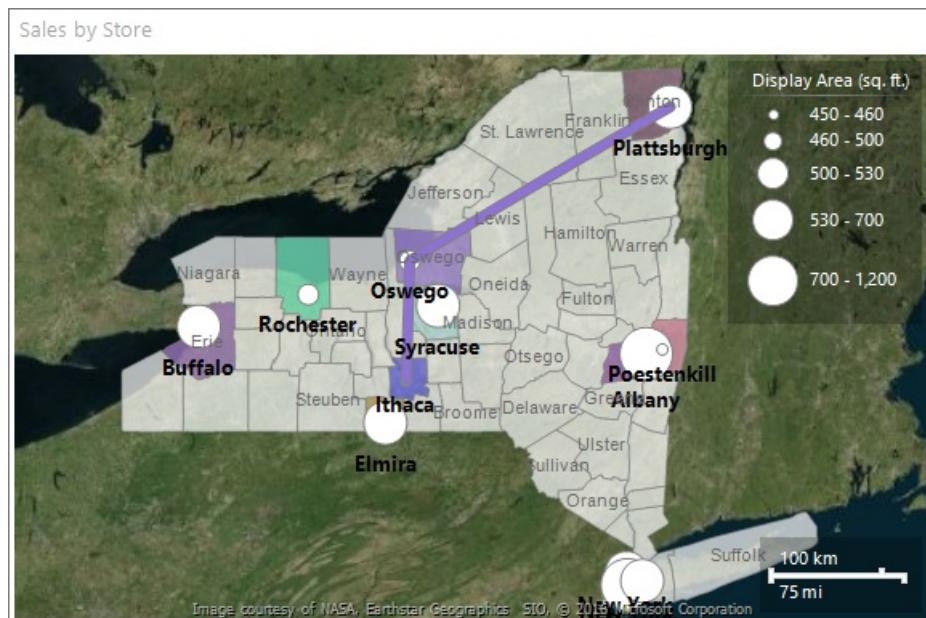
6a. Build a Relationship between Spatial and Analytical Data

To vary the county shapes by color based on analytical data, you first need to associate the analytical data with the spatial data. In this tutorial, you will use the county name to match on.

1. Switch to Design view.
2. Double-click the map to display the **Map Layers** pane.
3. Click the arrow next to PolygonLayer1, then click **Layer Data**. The **Map Polygon Layer Properties** dialog box opens.
4. On the **Analytical data** tab, under **Analytical dataset**, select DataSet1. This dataset was created by the wizard when you created the spatial data query for the counties.
5. Under **Fields to match on**, click **Add**. A new row is added.
6. Under **From spatial dataset**, click COUNTNAME.
7. Under **From analytical dataset**, click [County].



8. Click **OK**.
9. Preview the report.



By specifying a match field from the spatial data source and from the analytical dataset, you enable the report processor to group analytical data based on the map elements. A data-bound map element has a successful match for the values that you specified.

Each county that contains a store has a color that is based on the color palette for the style that you chose in the wizard. The other counties are gray.

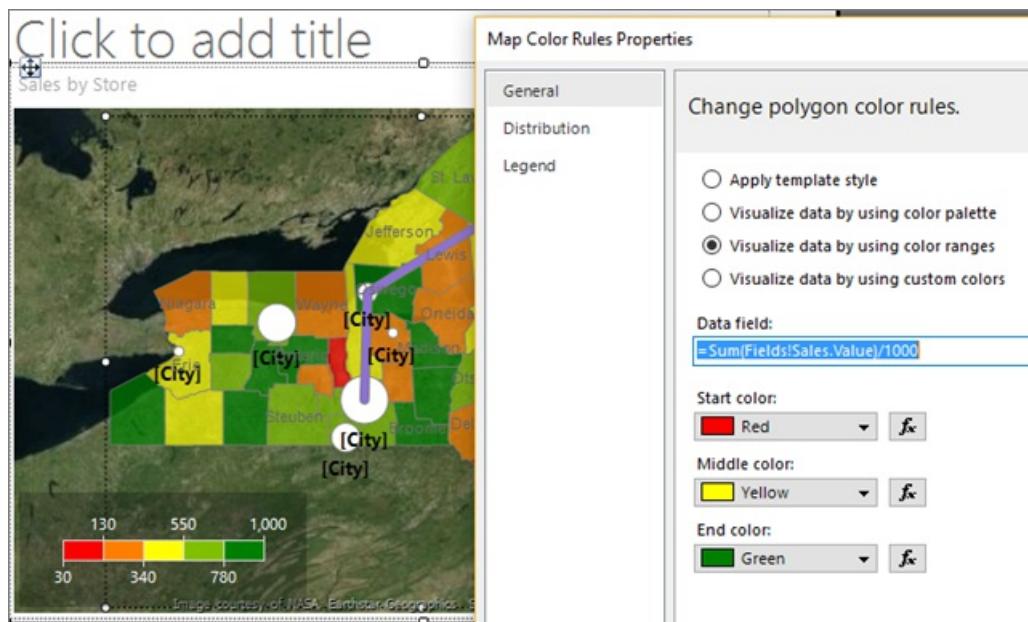
6b. Specify Color Rules for Polygons

To create a rule that varies the color of each county based store sales, you must specify the range values, the number of divisions within that range that you want to display, and the colors to use.

To specify color rules for all polygons that have associated data

1. Switch to Design view.
2. Click the arrow next to PolygonLayer1, then click **Polygon Color Rule**. The **Map Color Rules Properties** dialog box opens. Notice that the color rule option **Visualize data by using color palette** is selected. This option was set by the wizard.
3. Select **Visualize data by using color ranges**. The palette option is replaced by start color, middle color, and end color options.
4. Define range values for sales per county. In **Data field**, from the drop-down list, select `[Sum(Sales)]`.
5. To change the format to display currency in thousands, change the expression to the following:
`=Sum(Fields!Sales.Value)/1000`
6. Change **Start color** to **Red**.
7. Change **End color** to **Green**.

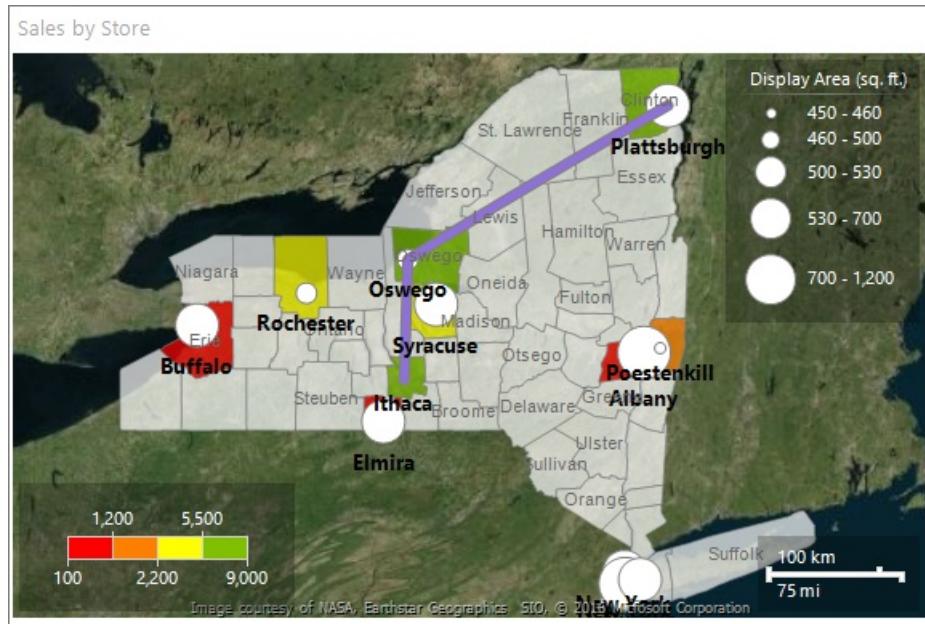
Red represents low sales values, **Yellow** represents middle sales values, and **Green** represents high sales values. The report processor calculates a range of colors based on these values and the options that you choose on the **Distribution** page.



8. Click **Distribution**.
9. Verify that the distribution type is **Optimal**. For the expression from step 5, optimal distribution divides the values into subranges that balance the number of items in each range and the span for each range.
10. Accept the default values for other options on this page. When you select the optimal distribution type, the number of subranges are calculated when the report runs.
11. Click **Legend**.
12. In **Color scale options**, verify that **Show in color scale** is selected.
13. In **Show in this legend**, from the drop-down list, select the blank line. For now, you will show the color ranges only in the color scale.

14. Click **OK**.

15. Preview the report.



The color scale displays four colors: red, orange, yellow, and green. Each color represents a sales range that is automatically calculated based on the sales by county.

6c. Format the Data in the Color Scale as Currency

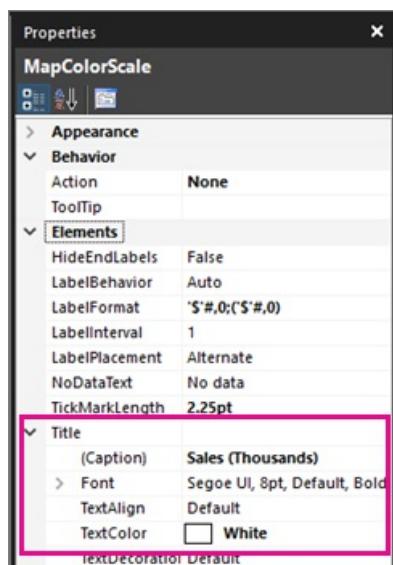
By default, data has a general format. In this section, you apply custom formats.

1. Switch to Design view.
2. Select the color scale. On the **Home** tab > **Number** section, click **Currency**.
3. Still in the **Number** section, click the **Decrease Decimal** button two times.

The color scale displays annual sales in currency format for each range.

6d. Add a Legend Title

1. With the color scale still selected, in the Properties pane you see properties for **MapColorScale**.
2. Expand the **Title** section, and in the **Caption** property, type **Sales (Thousands)**.
3. Change the **TextColor** property to **White**.



4. Preview the report.

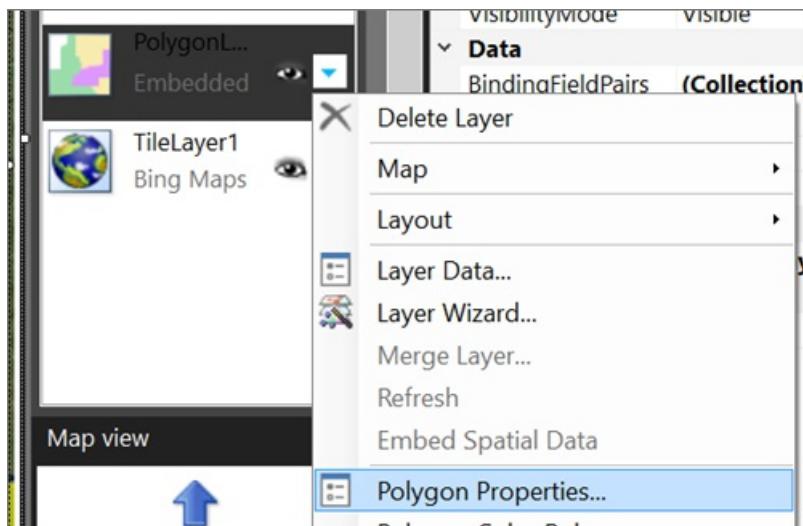
The counties that have associated stores and sales display according to the color rules. Counties that have no sales have no color.

6f. Change Color for Counties with No Data

You can set the default display options for all map elements on a layer. Color rules take precedence over these display options.

To set the display properties for all elements on a layer

1. Switch to Design view.
2. Double-click the map to display the **Map Layer** pane.
3. Click the down arrow on PolygonLayer1, and then click **Polygon Properties**.



The **Map Polygon Properties** dialog box opens. Display options set in this dialog box apply to all polygons on the layer before rule-based display options are applied.

4. On the **Fill** tab, verify that the fill style is **Solid**. Gradients and patterns apply to all colors.
5. In **Color**, select **Light Steel Blue**.
6. Click **OK**.
7. Preview the report.

Counties with no associated data display as gray-blue. Only counties with associated analytical data have the **Red** through **Green** colors from the color rules that you specified.

7. Add a Custom Point

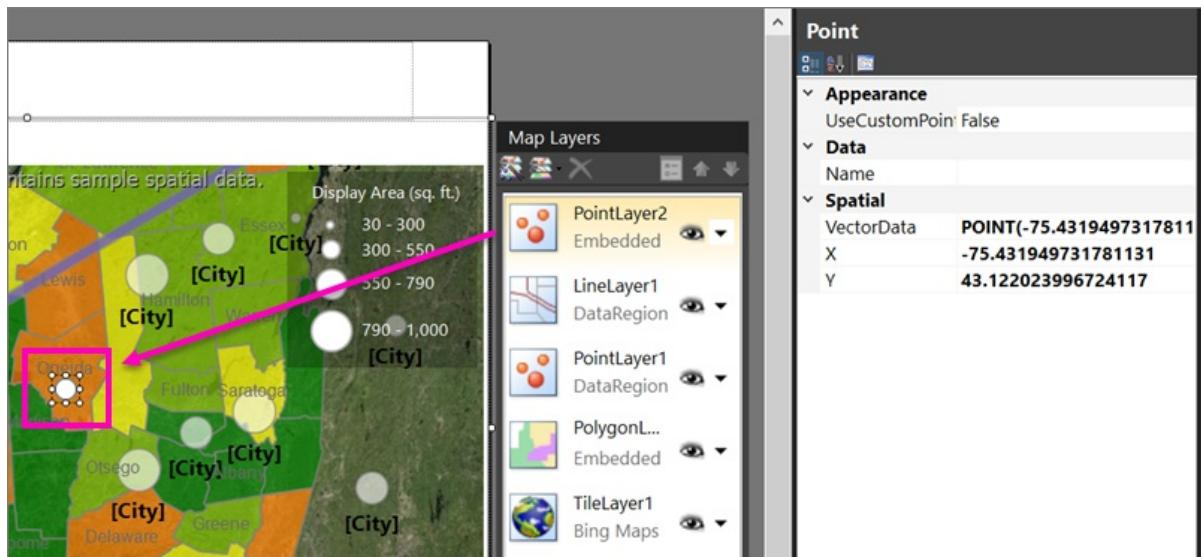
To represent a new store that has not yet been built, in this section you specify a point with the **Star** marker type.

1. Switch to Design view.
2. Double-click the map to display the **Map Layer** pane. On the toolbar, click **Add Layer** ; then click **Point Layer**.

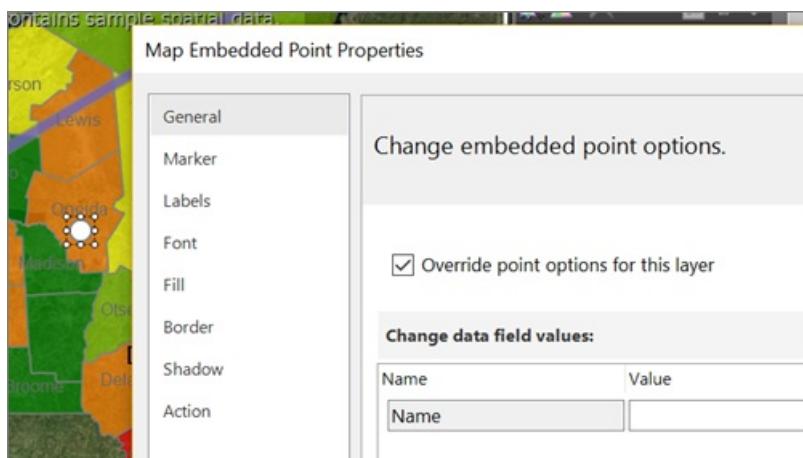
A new point layer is added to the map. By default, the point layer has spatial data type **Embedded**.

3. Click the arrow on PointLayer2 > **Add Point**.
4. Move the pointer over the map viewport. The cursor changes to crosshairs.

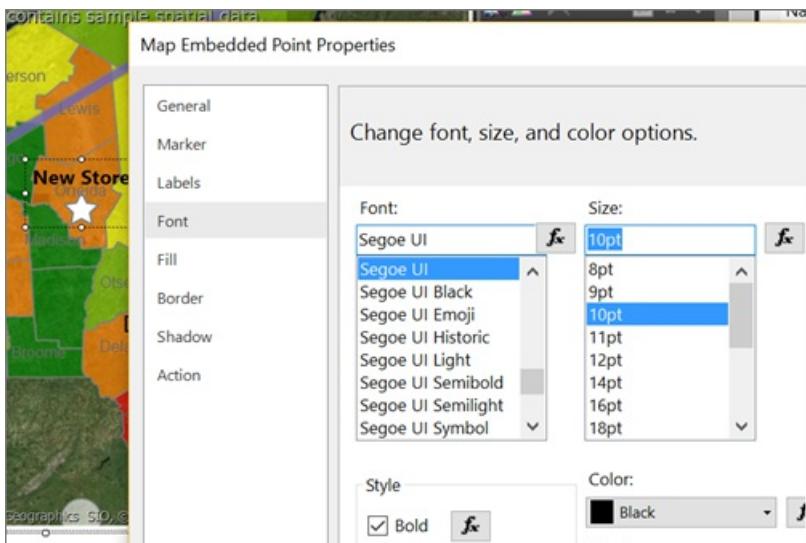
- Click the location on the map where you want to add a point. In this tutorial, click a location in Oneida county. A point marked by a circle is added to the layer at the spot where you clicked. By default, the point is selected.



- Right-click the point you added, and then click **Embedded Point Properties**.
- Select **Override point options for this layer**. Additional pages appear in the dialog box. Values that you set here take precedence over display options for the layer or for color rules.

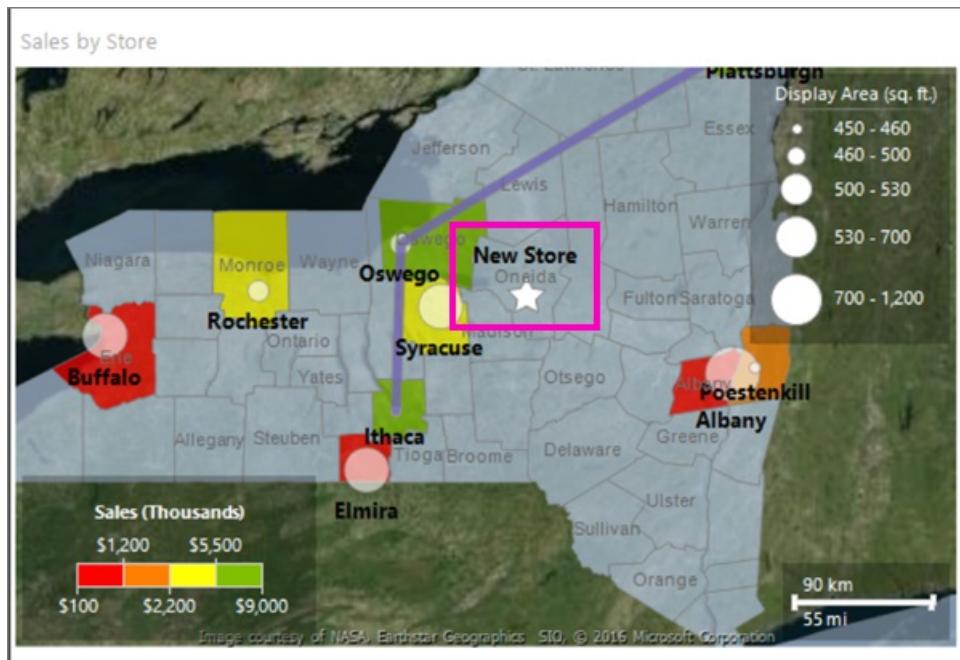


- On the **Marker** tab, for **Marker type**, select **Star**.
- Change **Marker size** to **18pt**.
- On the **Labels** tab, in **Label text**, type **New Store**.
- In **Placement**, click **Top**.
- On the **Font** tab, make the font size **10pt** and **Bold**.



13. Click **OK**.
14. Preview the report.

The label appears above the store location.



8. Center and Resize the Map

In this section, you learn to change the map center, and another way to change the zoom level.

1. Switch to Design view.
2. Select the map, then right-click and click **Viewport Properties**.
3. On the **Center and Zoom** tab, make sure **Set a view center and zoom level** is selected.
4. Set **Zoom level (percent)** to **125**.
5. Click **OK**.
6. Click the map, and drag to center it where you want it.
7. You can also use the mouse wheel to change the zoom level.
8. Preview the report.

In Design view, the map on the display surface and the view is based on sample data. In the rendered report, the map view is centered on the view that you specified.

9. Add a Report Title

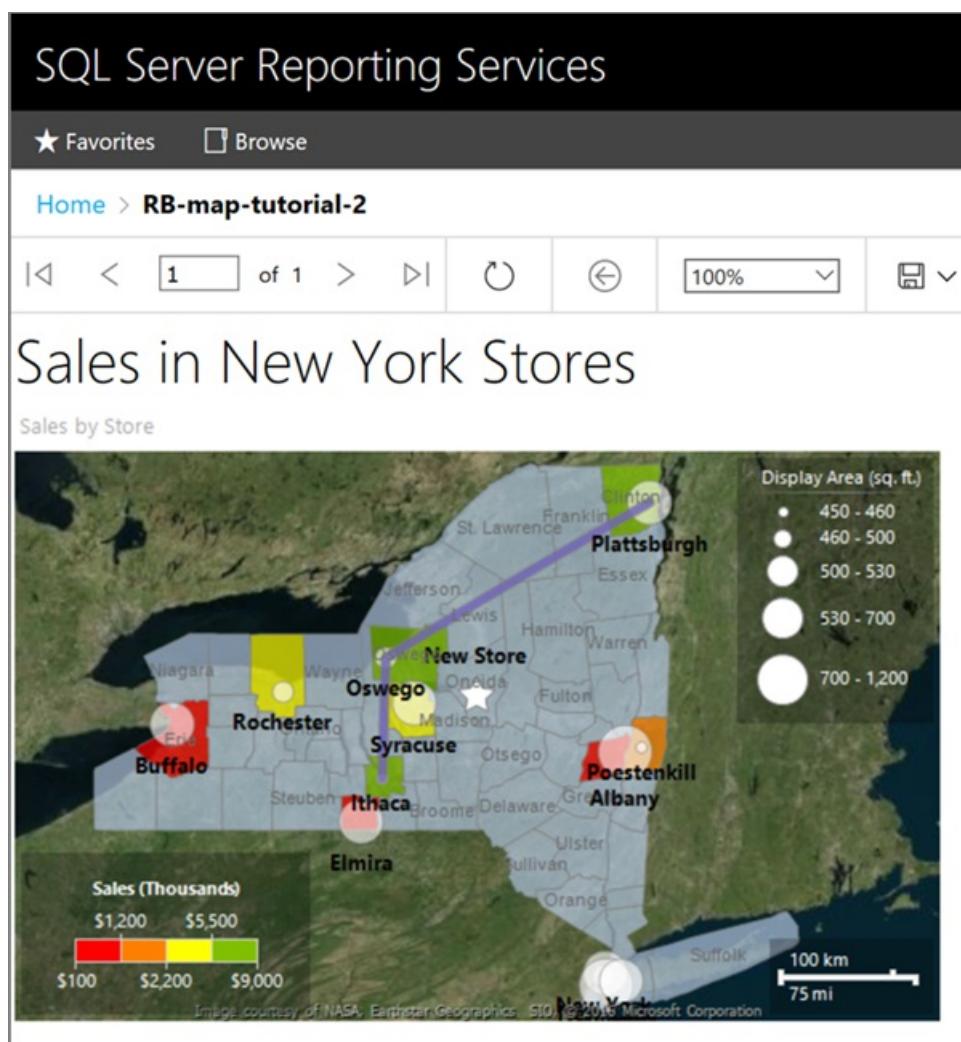
1. Switch to Design view.
2. On the design surface, click **Click to add title**.
3. Type **Sales in New York Stores** and then click outside the text box.

This title will appear at the top of the report. Items at the top of the report body when there is no page header defined are the equivalent of a report header.

10. Save the Report

1. In Design view or Preview, on the **File** menu > **Save As**.
2. In **Name**, type **Store Sales in New York**.
3. Save it to your local computer or to a Reporting Services server.
4. Click **Save**.

If you save it to a report server, you can view it there.



Next Steps

This concludes the walkthrough for how to add a map to your report.

For more information, see [Maps \(Report Builder and SSRS\)](#).

See Also

[Report Builder tutorials](#)

[Report Builder in SQL Server](#)

[Map Wizard and Map Layer Wizard \(Report Builder and SSRS\)](#)

[Vary Polygon, Line, and Point Display by Rules and Analytical Data \(Report Builder and SSRS\)](#)

Tutorial: Add a Parameter to Your Report (Report Builder)

11/30/2018 • 15 minutes to read • [Edit Online](#)

In this tutorial, you add a parameter to a Reporting Services paginated report so report readers can filter report data for one or more values.

The screenshot shows a Report Builder interface with a matrix report. At the top left, there is a parameter dialog for "Store Name?" with a dropdown menu showing "Contoso Europe Online Store, Contoso Asia Online Store". To the right of the dropdown is a "Show selections?" section with a radio button set to "True". Below the parameter dialog is the report's toolbar with icons for back, forward, search, zoom, and print. The main area displays a matrix report with three columns: "Subcategory", "Contoso Europe Online Store", and "Contoso Asia Online Store". The "Total" column is also present. The data rows include: Accessories (1534, 1755, 3289), Camcorders (1561, 1631, 3192), Digital Cameras (1553, 1772, 1553), and Digital SLR Cameras (1579, 1772, 3351). A "Total" row sums up the values. At the bottom of the report, a message states "Parameter values selected: Contoso Europe Online Store, Contoso Asia Online Store" and shows the timestamp "6/1/2016 4:26:41 PM".

Subcategory	Contoso Europe Online Store	Contoso Asia Online Store	Total
Accessories	1534	1755	3289
Camcorders	1561	1631	3192
Digital Cameras	1553	1772	1553
Digital SLR Cameras	1579	1772	3351
Total	6227	5158	11385

Report parameters are created automatically for each query parameter that you include in a dataset query. The parameter data type determines how it appears on the report view toolbar.

NOTE

In this tutorial, the steps for the wizard are consolidated into one procedure. For step-by-step instructions about how to browse to a report server, choose a data source, and create a dataset, see the first tutorial in this series: [Tutorial: Creating a Basic Table Report \(Report Builder\)](#).

Estimated time to complete this tutorial: 25 minutes.

Requirements

For information about requirements, see [Prerequisites for Tutorials \(Report Builder\)](#).

1. Create a Matrix Report and Dataset in the Table or Matrix Wizard

Create a matrix report, a data source, and a dataset.

NOTE

In this tutorial, the query contains the data values, so that it does not need an external data source. This makes the query quite long. In a business environment, a query would not contain the data. This is for learning purposes only.

To create a new matrix report

1. [Start Report Builder](#) either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the left pane, make sure **New Report** is selected.
3. In the right pane, click **Table or Matrix Wizard**.
4. On the **Choose a dataset** page, click **Create a dataset** > **Next**.
5. On the **Choose a connection to a data source** page, select a data source from the list or browse to the report server to select one. Select any data source that is type **SQL Server**.

6. Click **Next**.

You may need to enter your credentials.

7. On the **Design a query** page, click **Edit as Text**.
8. Paste the following query into the empty pane at the top:

```
;WITH CTE (StoreID, Subcategory, Quantity)
AS (
SELECT 200 AS StoreID, 'Digital SLR Cameras' AS Subcategory, 2002 AS Quantity
UNION SELECT 200 AS StoreID, 'Camcorders' AS Subcategory, 1954 AS Quantity
UNION SELECT 200 AS StoreID, 'Accessories' AS Subcategory, 1895 AS Quantity
UNION SELECT 199 AS StoreID, 'Digital Cameras' AS Subcategory, 1849 AS Quantity
UNION SELECT 306 AS StoreID, 'Digital SLR Cameras' AS Subcategory, 1579 AS Quantity
UNION SELECT 306 AS StoreID, 'Camcorders' AS Subcategory, 1561 AS Quantity
UNION SELECT 306 AS StoreID, 'Digital Cameras' AS Subcategory, 1553 AS Quantity
UNION SELECT 306 AS StoreID, 'Accessories' AS Subcategory, 1534 AS Quantity
UNION SELECT 307 AS StoreID, 'Accessories' AS Subcategory, 1755 AS Quantity
UNION SELECT 307 AS StoreID, 'Camcorders' AS Subcategory, 1631 AS Quantity
UNION SELECT 307 AS StoreID, 'Digital SLR Cameras' AS Subcategory, 1772 AS Quantity)
SELECT StoreID, Subcategory, Quantity
FROM CTE
```

This query combines the results of several Transact-SQL SELECT statements inside a common table expression to specify values that are based on simplified sales data for cameras from the Contoso sample database. The subcategories are digital cameras, digital single lens reflex (SLR) cameras, camcorders, and accessories.

9. On the query designer toolbar, click **Run (!)** to see the data.

The result set consists of 11 rows of data that show the quantity of items sold for each subcategory for four stores, in the following columns: StoreID, Subcategory, Quantity. The store name is not part of the result set. Later in this tutorial, you will look up the name of the store that corresponds to the store identifier from a separate dataset.

This query does not contain query parameters. You will add query parameters later in this tutorial.

10. Click **Next**.

2. Organize Data and Choose Layout in the Wizard

The wizard provides a starting design for displaying data. The preview pane in the wizard helps you to visualize the result of grouping data before you complete the table or matrix design.

To organize data into groups

1. On the **Arrange fields** page, drag Subcategory to **Row groups**.
2. Drag StoreID to **Column groups**.
3. Drag Quantity to **Values**.

You have organized the quantity sold values in rows grouped by subcategory, with one column for each store.

4. Click **Next**.
5. On the **Choose the Layout** page, under **Options**, make sure **Show subtotals and grand totals** is selected.

When you run the report, the last column will show the total quantity of each subcategory for all stores, and the last row will show the total quantity for all subcategories for each store.

6. Click **Next**.
7. Click **Finish**.

The matrix is added to the design surface. The matrix displays three columns and three rows. The contents of the cells in the first row are Subcategory, [StoreID], and Total. The contents of the cells in the second row contain expressions that represent the subcategory, the quantity of items sold for each store, and the total quantity for each subcategory for all stores. The cells in the final row display the grand total for each store.

Subcategory	[StoreID]	Total
[Subcategory]	[Sum(Quantity)]	[Sum(Quantity)]
Total	[Sum(Quantity)]	[Sum(Quantity)]

8. Click in the matrix, hover over the edge of the first column, grab the handle, and expand the column width.

Subcategory	[StoreID]	Total
[Subcategory]	[Sum(Quantity)]	[Sum(Quantity)]
Total	[Sum(Quantity)]	[Sum(Quantity)]

9. Click **Run** to preview the report.

The report runs on the report server and displays the title and the time the report processing occurred.

The screenshot shows the Microsoft SQL Server Report Builder interface. The toolbar at the top includes File, Run, Design, Views, Zoom, Navigation, Print, Page Setup, Print Layout, Export, and a timestamp of 4/6/2016 4:49:24 PM.

Subcategory	199	200	306	307	Total
Accessories		1895	1534	1755	5184
Camcorders		1954	1561	1631	5146
Digital Cameras	1849		1553		3402
Digital SLR Cameras		2002	1579	1772	5353
Total	1849	5851	6227	5158	19085

So far the column headings display the store identifier but not the store name. Later, you will add an expression to look up the store name in a dataset that contains store identifier/store name pairs.

3. Add a Query Parameter to Create a Report Parameter

When you add a query parameter to a query, Report Builder automatically creates a single-valued report parameter with default properties for name, prompt, and data type.

To add a query parameter

1. Click **Design** to switch back to Design view.
2. In the Report Data pane, expand the **Datasets** folder, right-click **DataSet1**, and then click **Query**.
3. Add the following Transact-SQL **WHERE** clause as the last line in the query:

```
WHERE StoreID = (@StoreID)
```

The **WHERE** clause limits the retrieved data to the store identifier that is specified by the query parameter ***@StoreID***.

4. On the query designer toolbar, click **Run (!)**. The **Define Query Parameters** dialog box opens and prompts for a value for the query parameter ***@StoreID***.
5. In **Parameter Value**, type **200**.
6. Click **OK**.
- The result set displays the quantities sold for Accessories, Camcorders, and Digital SLR Cameras for the store identifier **200**.
7. Click **OK**.
8. In the Report Data pane, expand the **Parameters** folder.

Note there is now a report parameter named ***@StoreID***, and a Parameters pane where you can lay out the report parameters.

Don't see a Parameters pane? On the **View** menu, select **Parameters**.

4. Change Default Data Type and Other Properties for a Report Parameter

After you create a report parameter, you can adjust the default values for properties.

To change the default data type for a report parameter

By default, the parameter you created has the data type **Text**. Because the store identifier is an integer, you can change the data type to **Integer**.

1. In the Report Data pane under the **Parameters** node, right-click ***@StoreID***, then click **Parameter Properties**.
2. In **Prompt**, type **Store identifier?** This text appears on the report viewer toolbar when you run the report.
3. In **Data type**, from the drop-down list, select **Integer**.
4. Accept the remaining default values in the dialog box.
5. Click **OK**.
6. Click **Run** to preview the report. The report viewer displays the prompt **Store Identifier?** for ***@StoreID***.
7. On the report viewer toolbar, next to Store ID, type **200**, and then click **View Report**.

Subcategory	200 Total
Accessories	1895 1895
Camcorders	1954 1954
Digital SLR Cameras	2002 2002
Total	5851 5851

4a. Add a Dataset to Provide Available Values and Display Names

To make sure your report readers type only valid values for a parameter, you can create a drop-down list of values to choose from. The values can come from a dataset or from a list that you specify. Available values must be supplied from a dataset with a query that does not contain a reference to the parameter.

To create a dataset for valid values for a parameter

1. Click **Design** to switch to Design view.

2. In the Report Data pane, right-click the **Datasets** folder, and then click **Add Dataset**.
3. In **Name**, type **Stores**.
4. Select **Use a dataset embedded in my report**.
5. In **Data source**, from the drop-down list, choose the data source you used in the first procedure.
6. In **Query type**, verify that **Text** is selected.
7. In **Query**, paste the following text:

```

SELECT 200 AS StoreID, 'Contoso Catalog Store' as StoreName
UNION SELECT 199 AS StoreID, 'Contoso North America Online Store' as StoreName
UNION SELECT 307 AS StoreID, 'Contoso Asia Online Store' as StoreName
UNION SELECT 306 AS StoreID, 'Contoso Europe Online Store' as StoreName

```

8. Click **OK**.

The Report Data pane displays the fields StoreID and StoreName under the **Stores** dataset node.

4b. Specify Available Values to Show in a List

After you create a dataset to provide available values, you change the report properties to specify which dataset and which field to use to populate the drop-down list of valid values on the Report Viewer toolbar.

To provide available values for a parameter from a dataset

1. In the Report Data pane, right-click the parameter *@StoreID*, then click **Parameter Properties**.
2. Click **Available Values**, and then click **Get values from a query**.
3. In **Dataset**, from the drop-down list, click **Stores**.
4. In **Value field**, from the drop-down list, click StoreID.
5. In **Label field**, from the drop-down list, click StoreName. The label field specifies the display name for the value.
6. Click **General**.
7. In **Prompt**, change **Store Identifier?** to **Store name?**

Report readers will now select from a list of store names instead of store identifiers. Note that the parameter data type remains **Integer** because the parameter is based on the store identifier, not the store name.

8. Click **OK**.
9. Preview the report.

In the report viewer toolbar, the parameter text box is now a drop-down list that displays **Select a Value**.

10. From the drop-down list, select Contoso Catalog Store, then click **View Report**.

The report displays the quantity sold for Accessories, Camcorders, and Digital SLR Cameras for the store identifier **200**.

4c. Specify a Default Value

You can specify a default value for each parameter so the report runs automatically.

To specify a default value from a dataset

1. Switch to Design view.
2. In the Report Data pane, right-click *@StoreID*, then click **Parameter Properties**.
3. Click **Default Values**, then click **Get values from a query**.
4. In **Dataset**, from the drop-down list, click **Stores**.
5. In **Value field**, from the drop-down list, click StoreID.
6. Click **OK**.
7. Preview the report.

For *@StoreID*, the report viewer displays the value "Contoso North America Online Store" because it's the first value from the result set for the dataset **Stores**. The report displays the quantity sold for Digital Cameras for store identifier **199**.

To specify a custom default value

1. Switch to Design view.
2. In the Report Data pane, right-click *@StoreID*, and then click **Parameter Properties**.
3. Click **Default Values > Specify values > Add**. A new value row is added.
4. In **Value**, type **200**.
5. Click **OK**.
6. Preview the report.

For *@StoreID*, the report viewer displays "Contoso Catalog Store" because it's the display name for store identifier **200**. The report displays the quantity sold for Accessories, Camcorders, and Digital SLR Cameras for the store identifier **200**.

4d. Look up a Name/Value Pair

A dataset might contain both the identifier and the corresponding name field. When you only have an identifier, you can look up the corresponding name in a dataset that you created that includes name/value pairs.

To look up a value from a dataset

1. Switch to Design view.
2. On the design surface, in the matrix, in the first row column header, right-click `[StoreID]` and then click **Expression**.
3. In the expression pane, delete all text except the beginning **equals sign** (=).
4. In **Category**, expand **Common Functions**, and click **Miscellaneous**. The Item pane displays a set of functions.
5. In Item, double-click **Lookup**. The expression pane displays `=Lookup(`. The Example pane displays an example of Lookup syntax.
6. Type the following expression:

```
=Lookup(Fields!StoreID.Value, Fields!StoreID.Value, Fields!StoreName.Value, "Stores")
```

The **Lookup** function takes the value for **StoreID**, looks it up in the "Stores" dataset, and returns the **StoreName** value.

7. Click **OK**.

The store column header contains the display text for a complex expression: **Expr**.

8. Preview the report.

The column header at the top of each column displays the store name instead of the store identifier.

5. Display the Selected Parameter Value in the Report

When your report readers have questions about a report, it helps to know which parameter values they chose. You can preserve user-selected values for each parameter in the report. One way is to display the parameters in a text box in the page footer.

To display the selected parameter value and label on a page footer

1. Switch to Design view.
2. Right-click the page footer > **Insert** > **Text Box**. Drag the text box next to the text box with the time stamp. Grab the side handle of the text box and expand the width.
3. From the Report Data pane, drag the parameter *@StoreID* to the text box. The text box displays `[@StoreID]`.
4. To display the parameter label, click in the text box until the insert cursor appears after the existing expression, type a space, and then drag another copy of the parameter from the Report Data pane to the text box. The text box displays `[@StoreID] [Label]`.
5. Right-click the first `[@StoreID]` and click **Expression**. The **Expression** dialog box opens. Replace the text `Value` with `Label`.
6. Click **OK**.

The text displays: `[@StoreID.Label] [Label]`.

7. Preview the report.

6. Use the Report Parameter in a Filter

Filters help control which data to use in a report after it is retrieved from an external data source. To let report readers control the data they want to see, you can include the report parameter in a filter for the matrix.

To specify a parameter in a matrix filter

1. Switch to Design view.
2. Right-click a row or column header handle on the matrix, and then click **Tablix Properties**.
3. Click **Filters**, and then click **Add**. A new filter row appears.
4. In **Expression**, from the drop-down list, select the dataset field StoreID. The data type displays **Integer**. When the expression value is a dataset field, the data type is set automatically.
5. In **Operator**, verify that the **equals sign (=)** is selected.
6. In **Value**, type `[@StoreID]`.
`[@StoreID]` is the simple expression syntax that represents `=Parameters!StoreID.Value`.
7. Click **OK**.
8. Preview the report.

The matrix displays data only for "Contoso Catalog Store".

9. On the report viewer toolbar, for **Store name?**, select **Contoso Asia Online Store**, and then click **View Report**.

The matrix displays data that corresponds to the store that you selected.

7. Change the Report Parameter to Accept Multiple Values

To change a parameter from single to multivalued, you must change the query and all expressions that contain a reference to the parameter, including filters. A multivalued parameter is an array of values. In a dataset query, query syntax must test for inclusion of one value in a set of values. In a report expression, expression syntax must access an array of values instead of an individual value.

To change a parameter from single to multivalued

1. Switch to Design view.
2. In the Report Data pane, right-click *@StoreID*, and then click **Parameter Properties**.
3. Select **Allow multiple values**.
4. Click **OK**.
5. In the Report Data pane, expand the **Datasets** folder, right-click **DataSet1**, and then click **Query**.
6. Change the **equals sign (=)** to **IN** in the Transact-SQL **WHERE** clause in the last line in the query:

```
WHERE StoreID IN (@StoreID)
```

The **IN** operator tests a value for inclusion in a set of values.

7. Click **OK**.
8. Right-click a row or column header handle on the matrix, and then click **Tablix Properties**.
9. Click **Filters**.
10. In **Operator**, select **In**.
11. Click **OK**.
12. In the text box that displays the parameter in the page footer, delete all text.
13. Right-click the text box, and then click **Expression**. Type the following expression:

```
=Join(Parameters!StoreID.Label, ", ")
```

This expression concatenates all store names that the user selected, separated by a comma and a space.

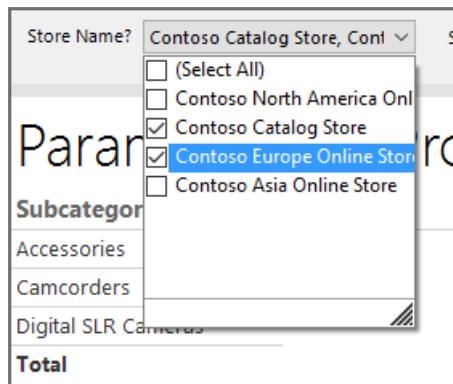
14. Click **OK**.
15. Click in the text box in front of the expression that you just created, and then type the following:

Parameter Values Selected:

16. Preview the report.
 17. Click the drop-down list next to **Store Name?**
- Each valid value appears next to a check box.
18. Click **Select All**, and then click **View Report**.

The report displays the quantity sold for all subcategories for all stores.

19. From the drop-down list, click **Select All** to clear the list, click "Contoso Catalog Store" and "Contoso Asia Online Store", and then click **View Report**.



8. Add a Boolean Parameter for Conditional Visibility

To add a Boolean parameter

1. On the design surface, in the Report Data pane, right-click **Parameters**, and click **Add Parameter**.
2. In **Name**, type ShowSelections.
3. In **Prompt**, type Show selections?
4. In **Data type**, click **Boolean**.
5. Click **Default Values**.
6. Click **Specify value**, and then click **Add**.
7. In **Value**, type **False**.
8. Click **OK**.

To set visibility based on a Boolean parameter

1. On the design surface, right-click the text box in the page footer that displays the parameter values, and then click **Text Box Properties**.
2. Click **Visibility**.
3. Select the option **Show or hide based on an expression**, and then click the expression button **Fx**.
4. Type the following expression: `=Not Parameters!ShowSelections.Value`

The text box Visibility option is controlled by the property Hidden. Apply the **Not** operator so that when the parameter is selected, the Hidden property is false, and the text box will be displayed.

5. Click **OK**.
6. Click **OK**.
7. Preview the report.

The text box that displays the parameter choices in the footer does not appear.

8. In the report viewer toolbar, next to **Show selections**, click **True > View Report**.

The text box in the page footer appears, showing all the store names you selected.

9. Add a Report Title

To add a report title

1. Switch to Design view.
2. On the design surface, click **Click to add title**.
3. Type Parameterized Product Sales, and then click outside the text box.

10. Save the Report

To save the report on a report server

1. From the **Report Builder** button, click **Save As**.
2. Click **Recent Sites and Servers**.
3. Select or type the name of the report server where you have permission to save reports.

The message **Connecting to report server appears**. When the connection is complete, you see the contents of the report folder that the report server administrator specified as the default location for reports.

4. In **Name**, replace the default name with Parameterized Sales Report.
5. Click **Save**.

The report is saved to the report server. The report server that you are connected to appears in the status bar at the bottom of the window.

Next Steps

This concludes the walkthrough for how to add a parameter to your report. To learn more about parameters, see [Report Parameters \(Report Builder and Report Designer\)](#).

See Also

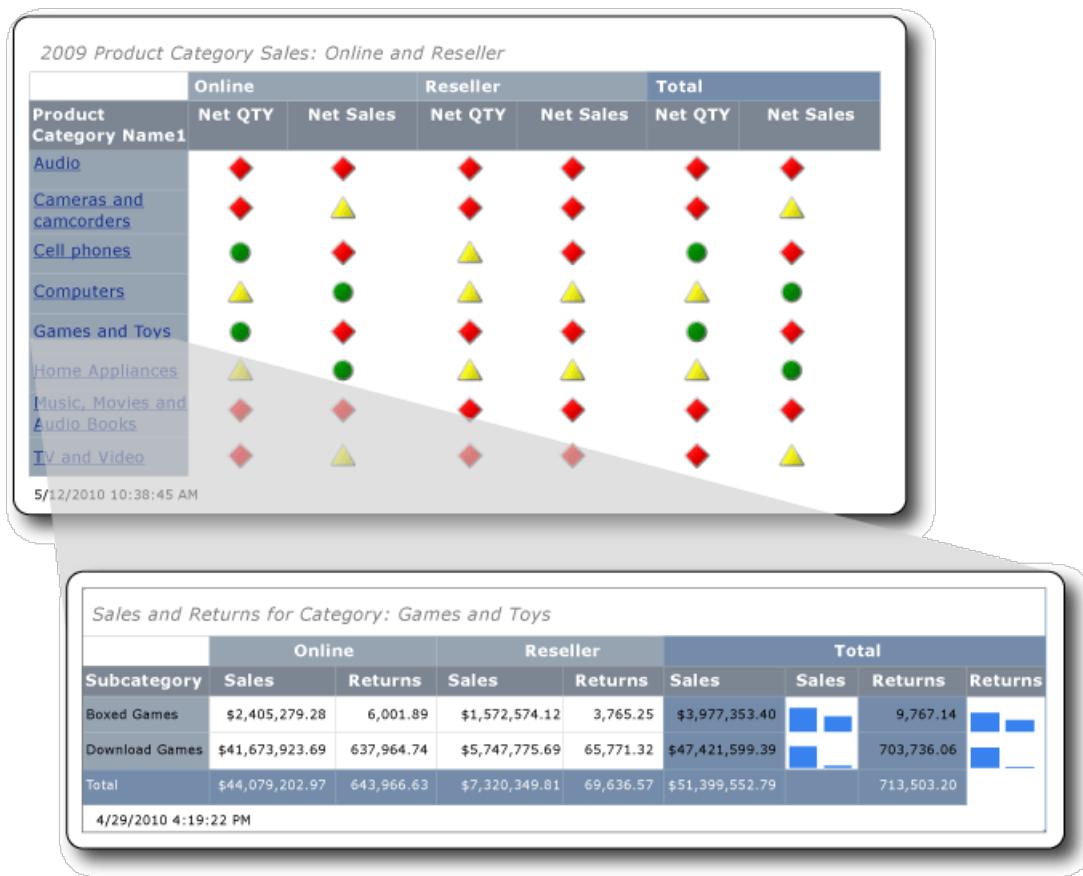
- [Report Builder Tutorials](#)
- [Report Builder in SQL Server](#)
- [Lookup Function](#)

Tutorial: Creating Drillthrough and Main Reports (Report Builder)

11/28/2018 • 21 minutes to read • [Edit Online](#)

This tutorial teaches you how to create two kinds of Reporting Services paginated reports: a drillthrough report and a main report. The sample sales data used in these reports is retrieved from an Analysis Services cube.

The following illustration shows the reports you will create, and shows how the field value, Games and Toys, in the main report displays in the drillthrough report's title. The data in the drillthrough report pertains to the Games and Toys product category.



Estimated time to complete this tutorial: 30 minutes.

Requirements

This tutorial requires access to the Contoso Sales cube for both the drillthrough and the main reports. This dataset comprises the ContosoDW data warehouse and the Contoso_Retail online analytical processing (OLAP) database. The reports you will create in this tutorial retrieve report data from the Contoso Sales cube. The Contoso_Retail OLAP database can be downloaded from [Microsoft Download Center](#). You need only download the file ContosoBldemoABF.exe. It contains the OLAP database.

The other file, ContosoBldemoBAK.exe, is for the ContosoDW data warehouse, which is not used in this tutorial.

The Web site includes instructions extracting and restoring the ContosoRetail.abf backup file to the Contoso_Retail OLAP database.

You must have access to an instance of Analysis Services on which to install the OLAP database.

For more about general requirements, see [Prerequisites for Tutorials \(Report Builder\)](#).

1. Create a Drillthrough Report from the Table or Matrix Wizard

From the Getting Started dialog box, create a matrix report by using the **Table or Matrix Wizard**. There are two modes available in the wizard: report design and shared dataset design. In this tutorial, you will use the report design mode.

To create a new report

1. [Start Report Builder](#) either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the left pane, verify that **New Report** is selected.
3. In the right pane, verify that **Table or Matrix Wizard** is selected.

1a. Specify a Data Connection

A data connection contains the information necessary to connect to an external data source such as an Analysis Services cube or a SQL Server database. To specify a data connection, you can use a shared data source from the report server or create an embedded data source that is used only in this report. In this tutorial, you will use an embedded data source. To learn more about using a shared data source, see [Alternative Ways to Get a Data Connection \(Report Builder\)](#).

To create an embedded data source

1. On the **Choose a dataset** page, select **Create a dataset**, and then click **Next**. The **Choose a connection to a data source** page opens.
2. Click **New**. The **Data Source Properties** dialog box opens.
3. In **Name**, type **Online and Reseller Sales Detail** as the name for the data source.
4. In **Select a connection type**, select **Microsoft SQL Server Analysis Services**, and then click **Build**.
5. In **Data source**, verify that the data source is **Microsoft SQL Server Analysis Services (AdomdClient)**.
6. In **Server name**, type the name of a server where an instance of Analysis Services is installed.
7. In **Select or enter a database name**, select the Contoso cube.
8. Click **OK**.
9. Verify that **Connection string** contains the following syntax:

```
Data Source=<servername>; Initial Catalog = Contoso
```

The `<servername>` is the name of an instance of SQL Server with Analysis Services installed.

10. Click **Credentials type**.

NOTE

Depending on how permissions are configured on the data source, you might need to change the default authentication options. For more information, see [Security \(Report Builder\)](#).

11. Click **OK**.

The **Choose a connection to a data source** page appears.

12. To verify that you can connect to the data source, click **Test Connection**.

The message **Connection created successfully** appears.

13. Click **OK**.

14. Click **Next**.

1b. Create an MDX Query

In a report, you can use a shared dataset that has a predefined query, or you can create an embedded dataset for use only in your report. In this tutorial, you will create an embedded dataset.

To create query filters

1. On the **Design a query** page, in the Metadata pane, click the button (...).

2. In the **Cube Selection** dialog box, click Sales, and then click **OK**.

TIP

If you do not want to build the MDX query manually, click the  icon, toggle the query designer to Query mode, paste the completed MDX to the query designer, and then proceed to step 6 in [To create the dataset](#).

```
SELECT NON EMPTY { [Measures].[Sales Amount], [Measures].[Sales Return Amount] } ON COLUMNS, NON EMPTY { ([Channel].[Channel Name].[Channel Name].ALLMEMBERS * [Product].[Product Category Name].[Product Category Name].ALLMEMBERS * [Product].[Product Subcategory Name].[Product Subcategory Name].ALLMEMBERS ) } DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS FROM ( SELECT ( { [Date].[Calendar Year].&[2009] } ) ON COLUMNS FROM ( SELECT ( { [Sales Territory].[Sales Territory Group].&[North America] } ) ON COLUMNS FROM ( SELECT ( STRTOSET(@ProductProductName, CONSTRAINED) ) ON COLUMNS FROM ( SELECT ( { [Channel].[Channel Name].&[2], [Channel].[Channel Name].&[4] } ) ON COLUMNS FROM [Sales])) ) WHERE ( [Sales Territory].[Sales Territory Group].&[North America], [Date].[Calendar Year].&[2009] ) CELL PROPERTIES VALUE, BACK_COLOR, FORE_COLOR, FORMATTED_VALUE, FORMAT_STRING, FONT_NAME, FONT_SIZE, FONT_FLAGS
```

3. In the Measure Group pane, expand Channel, and then drag Channel Name to the **Hierarchy** column in the filter pane.

The dimension name, Channel, is automatically added to the **Dimension** column. Do not change the **Dimension** or **Operator** columns.

4. To open the **Filter Expression** list, click the down arrow in the **Filter Expression** column.

5. In the filter expression list, expand **All Channel**, click **Online**, click **Reseller**, and then click **OK**.

The query now includes a filter to include only these channels: Online and Reseller.

6. Expand the Sales Territory dimension, and then drag Sales Territory Group to the **Hierarchy** column (below **Channel Name**).

7. Open the **Filter Expression** list, expand **All Sales Territory**, click **North America**, and then click **OK**.

The query now has a filter to include only sales in North America.

8. In the Measure Group pane, expand Date, and then drag Calendar Year to the **Hierarchy** column in the filter pane.

The dimension name, Date, is automatically added to the **Dimension** column. Do not change the

Dimension or **Operator** columns.

9. To open the **Filter Expression** list, click the down arrow in the **Filter Expression** column.
10. In the filter expression list, expand **All Date**, click **Year 2009**, and then click **OK**.

The query now includes a filter to include only the calendar year 2009.

To create the parameter

1. Expand the Product dimension, and then drag the Product Category Name member to the **Hierarchy** column below **Calendar Year**.
2. Open the **Filter Expression** list, click **All Products**, and then click **OK**.
3. Click the **Parameter** checkbox. The query now includes the parameter **ProductProductCategoryName**.

NOTE

The parameter contains the names of product categories. When you click a product category name in the main report, its name is passed to the drillthrough report by using this parameter.

To create the dataset

1. From the Channel dimension, drag Channel Name to the data pane.
2. From the Product dimension, drag Product Category Name to the data pane, and then place it to the right of Channel Name.
3. From the Product dimension, drag Product Subcategory Name to the data pane, and then place it to the right of Product Category Name.
4. In the Metadata pane, expand **Measure**, and then expand Sales.
5. Drag the Sales Amount measure to the data pane, and then place it to the right of Product Subcategory Name.
6. On the query designer toolbar, click **Run (!)**.
7. Click **Next**.

1c. Organize Data into Groups

When you select the fields on which to group the data, you design a matrix with rows and columns that displays detail and aggregated data.

To organize data into groups

1. To switch to design view, click **Design**.
2. On the **Arrange fields** page, drag Product_Subcategory_Name to **Row groups**.

NOTE

The spaces in the names are replaced with underscores (_). For example Product Category Name is Product_Category_Name.

3. Drag Channel_Name to **Column groups**.

4. Drag Sales_Amount to **Values**.

Sales_Amount is automatically aggregated by the Sum function, the default aggregate for numeric fields.

The value is `[Sum(Sales_Amount)]`.

To view the other aggregate functions available, open the drop-down list (do not change the aggregate function).

5. Drag Sales_Return_Amount to **Values**, and then place it below `[Sum(Sales_Amount)]`.

Steps 4 and 5 specify the data to display in the matrix.

6. Click **Next**.

1d. Add Subtotals and Totals

After you create groups, you can add and format rows where the aggregate values for the fields will display. You can also choose whether to show all the data or to let a user expand and collapse grouped data interactively.

To add subtotals and totals

1. On the **Choose the layout** page, under **Options**, verify that **Show subtotals and grand totals** is selected.

The wizard Preview pane displays a matrix with four rows.

2. Click **Next**.

3. Click **Finish**.

The table is added to the design surface.

4. To preview the report, click **Run (!)**.

2. Format Data as Currency

Apply currency formatting to the sales amount fields in the drillthrough report.

To format data as currency

1. To switch to design view, click **Design**.
2. To select and format multiple cells at one time, press the Ctrl key, and then select the cells that contain the numeric sales data.
3. On the **Home** tab, in the **Number** group, click **Currency**.

3. Add Columns to Show Sales Values in Sparklines

Instead of showing sales and sales returns as currency values, the report shows the values in a sparkline.

To add sparklines to columns

1. To switch to design view, click **Design**.
2. In the Total group of the matrix, right-click the **Sales Amount** column, click **Insert Column**, and then click **Right**.

An empty column is added to the right of **Sales Amount**.

3. On the ribbon, click **Rectangle**, and then click the empty cell to the right of the `[Sum(Sales_Amount)]` cell in the [Product_Subcategory] row group.
4. On the ribbon, click the **Sparkline** icon, and then click the cell where the rectangle was added.
5. In the **Select Sparkline Type** dialog box, verify that **Column** type is selected.
6. Click **OK**.

7. Right-click the sparkline.
8. In the Chart Data pane, click the **Add field** icon, and then click Sales_Amount.
9. Right-click the `Sales_Return_Amount` column, and then add a column to the right of it.
10. Repeat steps 2 through 6.
11. Right-click the sparkline.
12. In the Chart Data pane, click the **Add field** icon, and then click Sales_Return_Amount.
13. To preview the report, click **Run**.

4. Add Report Title with Product Category Name

A report title appears at the top of the report. You can place the report title in a report header or, if the report does not use one, in a text box at the top of the report body. In this tutorial, you will use the text box that is automatically placed at the top of the report body.

To add a report title

1. To switch to design view, click **Design**.
2. On the design surface, click **Click to add title**.
3. Type **Sales and Returns for Category:**
4. Right-click, and then click **Create Placeholder**.
5. Click the **(fx)** button to the right of the **Value** list.
6. In the **Expression** dialog box, in the Category pane, click **Dataset**, and then in the **Values** list double-click `First(Product_Category_Name)`.

The **Expression** box contains the following expression:

```
=First(Fields!Product_Category_Name.Value, "DataSet1")
```

7. To preview the report, click **Run**.

The report title includes the name of the first product category. Later, after you run this report as a drillthrough report, the product category name will dynamically change to reflect the name of the product category that was clicked in the main report.

5. Update Parameter Properties

By default parameters are visible, which is not appropriate for this report. You will update the parameter properties for the drillthrough report.

To hide a parameter

1. In the Report Data pane, expand **Parameters**.
2. Right-click `@ProductProductName`, and then click **Parameter Properties**.

NOTE

The @ character next to the name indicates that this is a parameter.

3. On the **General** tab, click **Hidden**.

4. In the **Prompt** box, type **Product Category**.

NOTE

Because the parameter is hidden, this prompt is never used.

5. Optionally, click **Available Values** and **Default Values** and review their options. Do not change any options on these tabs.
6. Click **OK**.

6. Save the Report to a SharePoint Library

You can save the report to a SharePoint library, report server, or your computer. If you save the report to your computer, a number of Reporting Services features such as report parts and subreports are not available. In this tutorial, you will save the report to a SharePoint library.

To save the report

1. From the Report Builder button, click **Save**. The **Save As Report** dialog box opens.

NOTE

If you are resaving a report, it is automatically resaved to its previous location. To change the location, use the **Save As** option.

2. To show a list of recently used report servers and SharePoint sites, click **Recent Sites and Servers**.
3. Select or type the name of the SharePoint site where you have permission to save reports.

The URL of the SharePoint library has the following syntax:

```
Http://<ServerName>/<Sites>/
```

4. Click **Save**.

Recent Sites and Servers lists the libraries on the SharePoint site.

5. Navigate to the library where you will save the report.
6. In the **Name** box, replace the default name with **ResellerVSONlineDrillthrough**.

NOTE

You will save the main report to the same location. If you want to save the main and drillthrough reports to different sites or libraries, you must update the path of the **Go to report** action in the main report.

7. Click **Save**.

1. Create the Main Report from the Table or Matrix Wizard

From the **Getting Started** dialog box, create a matrix report by using the **Table or Matrix Wizard**.

To create the main report

1. [Start Report Builder](#) either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the **Getting Started** dialog box, verify that **New Report** is selected, and then click **Table or Matrix Wizard**.

1a. Specify a Data Connection

You will add an embedded data source to the main report.

To create an embedded data source

1. On the **Choose a dataset** page, select **Create a dataset**, and then click **Next**.
2. Click **New**.
3. In **Name**, type **Online and Reseller Sales Main** as the name for the data source.
4. In **Select a connection type**, select **Microsoft SQL Server Analysis Services**, and then click **Build**.
5. In **Data source**, verify that the data source is **Microsoft SQL Server Analysis Services (AdomdClient)**.
6. In **Server name**, type the name of a server where an instance of MicrosoftAnalysis Services is installed.
7. In **Select or enter a database name**, select the Contoso cube.
8. Click **OK**.
9. Verify that the **Connection string** contains the following syntax:

```
Data Source=<servername>; Initial Catalog = Contoso
```

10. Click **Credentials type**.

Depending on how permissions are configured on the data source, you might need to change the default authentication.

11. Click **OK**.
12. To verify that you can connect to the data source, click **Test Connection**.
13. Click **OK**.
14. Click **Next**.

1b. Create an MDX Query

Next, create an embedded dataset. To do so, you will use the query designer to create filters, parameters, and calculated members as well as the dataset itself.

To create query filters

1. On the **Design a query** page, in the Metadata pane, in the cube section, click the ellipsis (...).
2. In the **Cube Selection** dialog box, click Sales, and then click **OK**.

TIP

If you do not want to build the MDX query manually, click the  icon, toggle the query designer to Query mode, paste the completed MDX to the query designer, and then proceed to step 5 in [To create the dataset](#).

```

WITH MEMBER [Measures].[Net QTY] AS [Measures].[Sales Quantity] - [Measures].[Sales Return Quantity]
MEMBER [Measures].[Net Sales] AS [Measures].[Sales Amount] - [Measures].[Sales Return Amount] SELECT NON
EMPTY { [Measures].[Net QTY], [Measures].[Net Sales] } ON COLUMNS, NON EMPTY { ([Channel].[Channel
Name].[Channel Name].ALLMEMBERS * [Product].[Product Category Name].[Product Category Name].ALLMEMBERS ) }
DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS FROM ( SELECT ( { [Date].[Calendar
Year].&[2009] } ) ON COLUMNS FROM ( SELECT ( STRTOSET(@ProductProductCategoryName, CONSTRAINED) ) ON
COLUMNS FROM ( SELECT ( { [Sales Territory].[Sales Territory Group].&[North America] } ) ON COLUMNS FROM
( SELECT ( { [Channel].[Channel Name].&[2], [Channel].[Channel Name].&[4] } ) ON COLUMNS FROM
[Sales])) ) WHERE ( [Sales Territory].[Sales Territory Group].&[North America], [Date].[Calendar Year].&
[2009] ) CELL PROPERTIES VALUE, BACK_COLOR, FORE_COLOR, FORMATTED_VALUE, FORMAT_STRING, FONT_NAME,
FONT_SIZE, FONT_FLAGSQuery text: Code.

```

3. In the Measure Group pane, expand Channel, and then drag Channel Name to the **Hierarchy** column in the filter pane.

The dimension name, Channel, is automatically added to the **Dimension** column. Do not change the **Dimension** or **Operator** columns.

4. To open the **Filter Expression** list, click the down arrow in the **Filter Expression** column.

5. In the filter expression list, expand **All Channel**, click **Online** and **Reseller**, and then click **OK**.

The query now includes a filter to include only these channels: Online and Reseller.

6. Expand the Sales Territory dimension, and then drag Sales Territory Group to the **Hierarchy** column, below **Channel Name**.

7. Open the **Filter Expression** list, expand **All Sales Territory**, click **North America**, and then click **OK**.

The query now has a filter to include only sales in North America.

8. In the Measure Group pane, expand Date, and drag Calendar Year to the **Hierarchy** column in the filter pane.

The dimension name, Date, is automatically added to the **Dimension** column. Do not change the **Dimension** or **Operator** columns.

9. To open the **Filter Expression** list, click the down arrow in the **Filter Expression** column.

10. In the filter expression list, expand **All Date**, click **Year 2009**, and then click **OK**.

The query now includes a filter to include only the calendar year 2009.

To create the parameter

1. Expand the Product dimension, and then drag the Product Category Name member to the **Hierarchy**

column below **Sales Territory Group**.

2. Open the **Filter Expression** list, click **All Products**, and then click **OK**.

3. Click the **Parameter** checkbox. The query now includes the parameter ProductProductCategoryName.

To create calculated members

1. Place the cursor inside the Calculated Members pane, right-click, and then click **New Calculated Member**.

2. In the Metadata pane, expand **Measures** and then expand Sales.

3. Drag the Sales Quantity measure to the **Expression** box, type the subtraction character (-), and then drag the Sales Return Quantity measure to the **Expression** box; place it after the subtraction character.

The following code shows the expression:

```
[Measures].[Sales Quantity] - [Measures].[Sales Return Quantity]
```

4. In the Name box, type **Net QTY**, and then click **OK**.

The Calculated Members pane lists the **Net QTY** calculated member.

5. Right-click **Calculated Members**, and then click **New Calculated Member**.

6. In the Metadata pane, expand **Measures**, and then expand Sales.

7. Drag the Sales Amount measure to the **Expression** box, type the subtraction character (-), and then drag the Sales Return Amount measure to the **Expression** box; place it after the subtraction character.

The following code shows the expression:

```
[Measures].[Sales Amount] - [Measures].[Sales Return Amount]
```

8. In the **Name** box, type **Net Sales**, and then click **OK**. The Calculated Members pane lists the **Net Sales** calculated member.

To create the dataset

1. From the Channel dimension, drag Channel Name to the data pane.
2. From the Product dimension, drag Product Category Name to the data pane, and then place it to the right of Channel Name.
3. From **Calculated Members**, drag **Net QTY** to the data pane, and then place it to the right of Product Category Name.
4. From Calculated Members, drag Net Sales to the data pane, and then place it to the right of **Net QTY**.
5. On the query designer toolbar, click **Run (!)**.

Review the query result set.

6. Click **Next**.

1c. Organize Data into Groups

When you select the fields on which to group data, you design a matrix with rows and columns that displays detail and aggregated data.

To organize data into groups

1. On the **Arrange fields** page, drag Product_Category_Name to **Row groups**.
2. Drag Channel_Name to **Column groups**.
3. Drag **Net_QTY** to **Values**.

Net_QTY is automatically aggregated by the Sum function, the default aggregate for numeric fields. The value is **[Sum(Net_QTY)]**.

To view the other aggregate functions available, open the drop-down list. Do not change the aggregate function.

4. Drag **Net_Sales_Return** to **Values** and then place it below **[Sum(Net_QTY)]**.

Steps 3 and 4 specify the data to display in the matrix.

1d. Add Subtotals and Totals

You can show subtotals and grand totals in reports. The data in the main report displays as an indicator; you will remove the grand total after you complete the wizard.

To add subtotals and grand totals

1. On the **Choose the layout** page, under **Options**, verify that **Show subtotals and grand totals** is selected.

The wizard Preview pane displays a matrix with four rows. When you run the report, each row will display in the following way: The first row is the column group, the second row contains the column headings, the third row contains the product category data (`[Sum(Net_QTY)]` and `[Sum(Net_Sales)]`), and the fourth row contains the totals.

2. Click **Next**.
3. Click **Finish**.
4. To preview the report, click **Run**.

2. Remove the Grand Total Row

The data values are shown as indicator states, including the column group totals. Remove the row that displays the grand total.

To remove the grand total row

1. To switch to design view, click **Design**.
2. Click the Total row (the last row in the matrix), right-click, and then click **Delete Rows**.
3. To preview the report, click **Run**.

3. Configure Text Box Action for Drillthrough

To enable the drillthrough, specify an action on a text box in the main report.

To enable an action

1. To switch to design view, click **Design**.
2. Right-click the cell that contains `Product_Category_Name`, and then click **Text Box Properties**.
3. Click the **Action** tab.
4. Select **Go to report**.
5. In **Specify a report**, click **Browse**, and then locate the drillthrough report named `ResellerVSONlineDrillthrough`.
6. To add a parameter to run the drillthrough report, click **Add**.
7. In the **Name** list, select `ProductProductCategoryName`.
8. In **Value**, type `[Product_Category_Name.UniqueName]`.

`Product_Category_Name` is a field in the dataset.

IMPORTANT

You must include the **UniqueName** property because the drillthrough action requires a unique value.

9. Click **OK**.

To format the drillthrough field

1. Right-click the cell that contains the `Product_Category_Name`, and then click **Text Box Properties**.
2. Click the **Font** tab.
3. In the **Effects** list, select **Underline**.
4. In the **Color** list, select **Blue**.
5. Click **OK**.
6. To preview your report, click **Run**.

The product category names are in the common link format (blue and underlined).

4. Replace Numeric Values with Indicators

Use indicators to show the state of quantities and sales for Online and Reseller channels.

To add an indicator for Net QTY values

1. To switch to design view, click **Design**.
2. On the ribbon, click the **Rectangle** icon, and then click in the `[Sum(Net_QTY)]` cell in the `[Product_Category_Name]` row group in the `channel_Name` column group.
3. On the ribbon, click the **Indicator** icon, and then click inside the rectangle. The **Select Indicator Type** dialog box opens with the **Directional** indicator selected.
4. Click the **3 Signs** type, and then click **OK**.
5. Right-click the indicator and in the Gauge Data pane, click the down arrow next to **(Unspecified)**. Select `Net_QTY`.
6. Repeat steps 2 through 5 for the `[Sum(Net_QTY)]` cell in the `[Product_Category_Name]` row group within **Total**.

To add an indicator for Net Sales values

1. On the ribbon, click the **Rectangle** icon, and then click inside the `[Sum(Net_Sales)]` cell in the `[Product_Category_Name]` row group in the `channel_Name` column group.
2. On the ribbon, click the **Indicator** icon, and then click inside the rectangle.
3. Click the **3 Signs** type, and then click **OK**.
4. Right-click the indicator and in the Gauge Data pane, click the down arrow next to **(Unspecified)**. Select `Net_Sales`.
5. Repeat steps 1 through 4 for the `[Sum(Net_Sales)]` cell in the `[Product_Category_Name]` row group within **Total**.
6. To preview your report, click **Run**.

5. Update Parameter Properties

By default, parameters are visible, which is not appropriate for this report. You will update the parameter properties to make the parameter internal.

To make the parameter internal

1. In the Report Data pane, expand **Parameters**.
2. Right-click `@ProductCategoryName`, and then click **Parameter Properties**.

3. On the **General** tab, click **Internal**.
4. Optionally, click the **Available Values** and **Default Values** tabs and review their options. Do not change any options on these tabs.
5. Click **OK**.

6. Add a Report Title

Add a title to the main report.

To add a report title

1. On the design surface, click **Click to add title**.
2. Type **2009 Product Category Sales: Online and Reseller Category:**
3. Select the text you typed.
4. On the **Home** tab of the ribbon, in the Font group, select the **Times New Roman** font, **16pt** size, and the **Bold** and **Italic** styles.
5. To preview your report, click **Run**.

7. Save the Main Report to a SharePoint Library

Save the main report to a SharePoint library.

To save the report

1. To switch to design view, click **Design**.
2. From the Report Builder button, click **Save**.
3. Optionally, to show a list of recently used report servers and SharePoint sites, click **Recent Sites and Servers**.
4. Select or type the name of the SharePoint site where you have permission to save reports. The URL of the SharePoint library has the following syntax:

```
Http://<ServerName>/<Sites>/
```

5. Navigate to the library where you want to save the report.
6. In **Name**, replace the default name with **ResellerVSOnlineMain**.

IMPORTANT

Save the main report to the same location where you saved the drillthrough report. To save the main and drillthrough reports to different sites or libraries, confirm that the **Go to report** action in the main report, points to the correct location of the drillthrough report.

7. Click **Save**.

8. Run the Main and Drillthrough Reports

Run the main report, and then click values in the product category column to run the drillthrough report.

To run the reports

1. Open the SharePoint library where the reports are saved.

2. Double-click ResellerVSOnlineMain.

The report runs and displays product category sales information.

3. Click the **Games and Toys** link in the column that contains product category names.

The drillthrough report runs, displaying only the values for the Games and Toys product category.

4. To return to the main report, click the Internet Explorer back button.

5. Optionally, explore other product categories by clicking their names.

See Also

[Report Builder tutorials](#)

Tutorial: Introducing Expressions

11/27/2018 • 18 minutes to read • [Edit Online](#)

In this Report Builder tutorial, you use expressions with common functions and operators to create powerful and flexible Reporting Services paginated reports.

You will write expressions that concatenate name values, look up values in a separate dataset, display different colors based on field values, and so on.

The report is a banded report with alternating rows in white and a color. The report includes a parameter for selecting the color of the non-white rows.

This illustration shows a report similar to the one you will create.

M/F	Name	State Province	Last Purchase	Days Ago	Country	Region	YTD Purchase + or - AVG Sales
Blue	A. Martin	Saxony	11/19/2015	301	Germany		\$2,997.60
Red	A. Nath	Alaska	10/13/2015	338	United States		\$607.50
Red	B. Sanchez	North Dakota	9/17/2015	364	United States		\$6,191.00
Red	B. She	Hamburg	5/10/2015	494	Germany		\$7,497.30
Blue	C. Reed	Nebraska	8/27/2015	385	United States		\$8,772.00
Blue	C. Petulescu	Wisconsin	11/30/2015	290	United States		\$3,470.00
Red	C. Randall	Utah	1/11/2015	613	United States		\$7,218.10
Blue	F. Ross	Alberta	10/17/2015	334	Canada		\$9,248.15
Red	G. Patterson	Kansas	10/18/2015	333	United States		\$1,215.00
Blue	J. Bailey	British Columbia	6/15/2015	458	Canada		\$1,147.50
	I. Danner	England	8/15/2015	307	United Kingdom		\$9,877.50

Estimated time to complete this tutorial: 30 minutes.

Requirements

For information about requirements, see [Prerequisites for Tutorials \(Report Builder\)](#).

1. Create a Table Report and Dataset from the Table or Matrix Wizard

In this section, you create a table report, a data source, and a dataset. When you lay out the table, you will include only a few fields. After you complete the wizard you will manually add columns. The wizard makes it easy to lay out the table.

NOTE

In this tutorial, the query contains the data values, so it does not need an external data source. This makes the query quite long. In a business environment, a query would not contain the data. This is for learning purposes only.

To create a table report

1. [Start Report Builder](#) either from your computer, the Reporting Services web portal, or SharePoint integrated mode.

The **New Report or Dataset** dialog box opens.

If you don't see the **New Report or Dataset** dialog box, on the **File** menu > **New**.

2. In the left pane, verify that **New Report** is selected.
3. In the right pane, click **Table or Matrix Wizard**.
4. On the **Choose a dataset** page, click **Create a dataset** > **Next**.
5. On the **Choose a connection to a data source** page, select a data source that is type **SQL Server**. Select a data source from the list or browse to the report server to select one.

NOTE

The data source you choose isn't important, as long as you have adequate permissions. You will not be getting data from the data source. For more information, see [Alternative Ways to Get a Data Connection \(Report Builder\)](#).

6. Click **Next**.
7. On the **Design a query** page, click **Edit as Text**.
8. Paste the following query into the query pane:

```
SELECT 'Lauren' AS FirstName, 'Johnson' AS LastName, 'American Samoa' AS StateProvince, 1 AS CountryRegionID, 'Female' AS Gender, CAST(9996.60 AS money) AS YTDPurchase, CAST('2015-6-10' AS date) AS LastPurchase
UNION SELECT 'Warren' AS FirstName, 'Pal' AS LastName, 'New South Wales' AS StateProvince, 2 AS CountryRegionID, 'Male' AS Gender, CAST(5747.25 AS money) AS YTDPurchase, CAST('2015-7-3' AS date) AS LastPurchase
UNION SELECT 'Fernando' AS FirstName, 'Ross' AS LastName, 'Alberta' AS StateProvince, 3 AS CountryRegionID, 'Male' AS Gender, CAST(9248.15 AS money) AS YTDPurchase, CAST('2015-10-17' AS date) AS LastPurchase
UNION SELECT 'Rob' AS FirstName, 'Caron' AS LastName, 'Northwest Territories' AS StateProvince, 3 AS CountryRegionID, 'Male' AS Gender, CAST(742.50 AS money) AS YTDPurchase, CAST('2015-4-29' AS date) AS LastPurchase
UNION SELECT 'James' AS FirstName, 'Bailey' AS LastName, 'British Columbia' AS StateProvince, 3 AS CountryRegionID, 'Male' AS Gender, CAST(1147.50 AS money) AS YTDPurchase, CAST('2015-6-15' AS date) AS LastPurchase
UNION SELECT 'Bridget' AS FirstName, 'She' AS LastName, 'Hamburg' AS StateProvince, 4 AS CountryRegionID, 'Female' AS Gender, CAST(7497.30 AS money) AS YTDPurchase, CAST('2015-5-10' AS date) AS LastPurchase
UNION SELECT 'Alexander' AS FirstName, 'Martin' AS LastName, 'Saxony' AS StateProvince, 4 AS CountryRegionID, 'Male' AS Gender, CAST(2997.60 AS money) AS YTDPurchase, CAST('2015-11-19' AS date) AS LastPurchase
UNION SELECT 'Yolanda' AS FirstName, 'Sharma' AS LastName, 'Micronesia' AS StateProvince, 5 AS CountryRegionID, 'Female' AS Gender, CAST(3247.95 AS money) AS YTDPurchase, CAST('2015-8-23' AS date) AS LastPurchase
UNION SELECT 'Marc' AS FirstName, 'Zimmerman' AS LastName, 'Moselle' AS StateProvince, 6 AS CountryRegionID, 'Male' AS Gender, CAST(1200.00 AS money) AS YTDPurchase, CAST('2015-11-16' AS date) AS LastPurchase
UNION SELECT 'Katherine' AS FirstName, 'Abel' AS LastName, 'Moselle' AS StateProvince, 6 AS CountryRegionID, 'Female' AS Gender, CAST(2025.00 AS money) AS YTDPurchase, CAST('2015-12-1' AS date) AS LastPurchase
UNION SELECT 'Nicolas' AS FirstName, 'Anand' AS LastName, 'Seine (Paris)' AS StateProvince, 6 AS CountryRegionID, 'Male' AS Gender, CAST(1425.00 AS money) AS YTDPurchase, CAST('2015-12-11' AS date) AS LastPurchase
UNION SELECT 'James' AS FirstName, 'Peters' AS LastName, 'England' AS StateProvince, 12 AS CountryRegionID, 'Male' AS Gender, CAST(887.50 AS money) AS YTDPurchase, CAST('2015-8-15' AS date) AS LastPurchase
UNION SELECT 'Alison' AS FirstName, 'Nath' AS LastName, 'Alaska' AS StateProvince, 7 AS CountryRegionID, 'Female' AS Gender, CAST(607.50 AS money) AS YTDPurchase, CAST('2015-10-13' AS date) AS LastPurchase
UNION SELECT 'Grace' AS FirstName, 'Patterson' AS LastName, 'Kansas' AS StateProvince, 7 AS CountryRegionID, 'Female' AS Gender, CAST(1215.00 AS money) AS YTDPurchase, CAST('2015-10-18' AS date) AS LastPurchase
```

```

UNION SELECT 'Bobby' AS FirstName, 'Sanchez' AS LastName, 'North Dakota' AS StateProvince, 7 AS
CountryRegionID, 'Female' AS Gender, CAST(6191.00 AS money) AS YTDPurchase, CAST('2015-9-17' AS date) AS
LastPurchase
UNION SELECT 'Charles' AS FirstName, 'Reed' AS LastName, 'Nebraska' AS StateProvince, 7 AS
CountryRegionID, 'Male' AS Gender, CAST(8772.00 AS money) AS YTDPurchase, CAST('2015-8-27' AS date) AS
LastPurchase
UNION SELECT 'Orlando' AS FirstName, 'Romeo' AS LastName, 'Texas' AS StateProvince, 7 AS
CountryRegionID, 'Male' AS Gender, CAST(8578.00 AS money) AS YTDPurchase, CAST('2015-7-29' AS date) AS
LastPurchase
UNION SELECT 'Cynthia' AS FirstName, 'Randall' AS LastName, 'Utah' AS StateProvince, 7 AS
CountryRegionID, 'Female' AS Gender, CAST(7218.10 AS money) AS YTDPurchase, CAST('2015-1-11' AS date) AS
LastPurchase
UNION SELECT 'Rebecca' AS FirstName, 'Roberts' AS LastName, 'Washington' AS StateProvince, 7 AS
CountryRegionID, 'Female' AS Gender, CAST(8357.80 AS money) AS YTDPurchase, CAST('2015-10-28' AS date)
AS LastPurchase
UNION SELECT 'Cristian' AS FirstName, 'Petulescu' AS LastName, 'Wisconsin' AS StateProvince, 7 AS
CountryRegionID, 'Male' AS Gender, CAST(3470.00 AS money) AS YTDPurchase, CAST('2015-11-30' AS date) AS
LastPurchase
UNION SELECT 'Cynthia' AS FirstName, 'Randall' AS LastName, 'Utah' AS StateProvince, 7 AS
CountryRegionID, 'Female' AS Gender, CAST(7218.10 AS money) AS YTDPurchase, CAST('2015-1-11' AS date) AS
LastPurchase
UNION SELECT 'Rebecca' AS FirstName, 'Roberts' AS LastName, 'Washington' AS StateProvince, 7 AS
CountryRegionID, 'Female' AS Gender, CAST(8357.80 AS money) AS YTDPurchase, CAST('2015-10-28' AS date)
AS LastPurchase
UNION SELECT 'Cristian' AS FirstName, 'Petulescu' AS LastName, 'Wisconsin' AS StateProvince, 7 AS
CountryRegionID, 'Male' AS Gender, CAST(3470.00 AS money) AS YTDPurchase, CAST('2015-11-30' AS date) AS
LastPurchase

```

9. On the query designer toolbar, click **Run (!)**. The result set displays 23 rows of data in the following columns: FirstName, LastName, StateProvince, CountryRegionID, Gender, YTDPurchase, and LastPurchase.

The screenshot shows the 'New Table or Matrix' dialog box. At the top, there are tabs for 'Edit as Text', 'Import...', and 'Command type: Text'. Below the tabs, there is a preview area with the query text:

```

UNION SELECT 'Rebecca' AS FirstName, 'Roberts' AS LastName, 'Washington' AS StateProvince, 7 AS
CountryRegionID, 'Female' AS Gender, CAST(8357.80 AS money) AS YTDPurchase, CAST('2015-10-28' AS date)
AS LastPurchase
UNION SELECT 'Cristian' AS FirstName, 'Petulescu' AS LastName, 'Wisconsin' AS StateProvince, 7 AS
CountryRegionID, 'Male' AS Gender, CAST(3470.00 AS money) AS YTDPurchase, CAST('2015-11-30' AS date) AS
LastPurchase

```

Below the preview is a table view showing 23 rows of data with columns: FirstName, LastName, StateProvince, CountryRegionID, Gender, YTDPurchase, and LastPurchase. The data includes entries like Alexander Martin from Saxony, Bobby Sanchez from North Dakota, and Orlando Romeo from Texas.

FirstName	LastName	StateProvince	CountryRegionID	Gender	YTDPurchase	LastPurchase
Alexander	Martin	Saxony	4	Male	2997.6000	11/19/2015 12:...
Alison	Nath	Alaska	7	Female	607.5000	10/13/2015 12:...
Bobby	Sanchez	North Dakota	7	Female	6191.0000	9/17/2015 12:0...
Bridget	She	Hamburg	4	Female	7497.3000	5/10/2015 12:0...
Charles	Reed	Nebraska	7	Male	8772.0000	8/27/2015 12:0...
Cristian	Petulescu	Wisconsin	7	Male	3470.0000	11/30/2015 12:...
Cynthia	Randall	Utah	7	Female	7218.1000	1/11/2015 12:0...
Fernando	Ross	Alberta	3	Male	9248.1500	10/17/2015 12:...
Grace	Patterson	Kansas	7	Female	1215.0000	10/18/2015 12:...
James	Bailey	British Columbia	3	Male	1147.5000	6/15/2015 12:0...
James	Peters	England	12	Male	887.5000	8/15/2015 12:0...
Katherine	Abel	Moselle	6	Female	2025.0000	12/1/2015 12:0...
Lauren	Johnson	American Samoa	1	Unknown	9996.6000	6/10/2015 12:0...
Marc	Zimmerman	Moselle	6	Male	1200.0000	11/16/2015 12:...
Nicolas	Anand	Seine (Paris)	6	Male	1425.0000	12/11/2015 12:...
Orlando	Romeo	Texas	7	Male	8578.0000	7/29/2015 12:0...
Rebecca	Roberts	Washington	7	Female	8357.8000	10/28/2015 12:...
Rob	Caron	Northwest Terri...	3	Male	742.5000	4/29/2015 12:0...
Warren	Pal	New South Wales	2	Male	5747.2500	7/3/2015 12:00:...
Yolanda	Sharma	Micronesia	5	Female	3247.9500	8/23/2015 12:0...

At the bottom of the dialog, there are buttons for 'Help', '< Back', 'Next >', and 'Cancel'.

10. Click **Next**.

11. On the **Arrange fields** page, drag the following fields, in the specified order, from the **Available Fields** list to the **Values** list.

- StateProvince
- CountryRegionID
- LastPurchase

- YTDPurchase

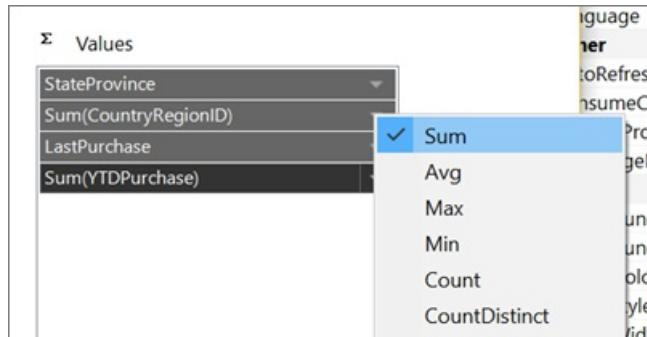
Because the CountryRegionID and YTDPurchase contain numeric data, the SUM aggregate is applied to them by default, but you don't want them to be sums.

12. In the **Values** list, right-click **CountryRegionID** and clear the **Sum** check box.

Sum is no longer applied to CountryRegionID.

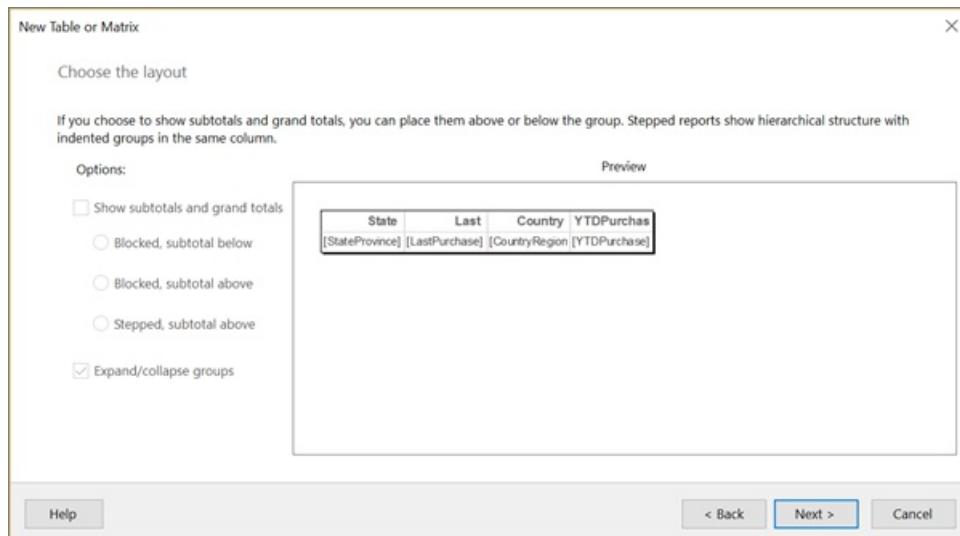
13. In the **Values** list, right-click **YTDPurchase** and click the **Sum** option.

Sum is no longer applied to YTDPurchase.



14. Click **Next**.

15. On the **Choose the layout** page, keep all the default settings and click **Next**.



16. Click **Finish**.

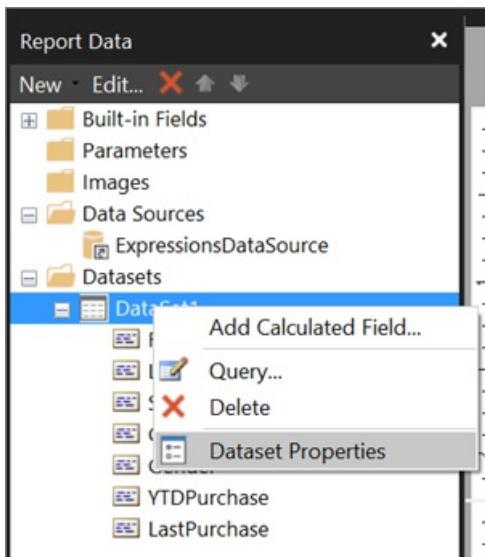
2. Update Default Names of the Data Source and Dataset

To update the default name of the data source

1. In the Report Data pane, expand the **Data Sources** folder.
2. Right-click **DataSource1** and click **Data Source Properties**.
3. In the **Name** box, type **ExpressionsDataSource**
4. Click **OK**.

To update the default name of the dataset

1. In the Report Data pane, expand the **Datasets** folder.
2. Right-click **DataSet1** and click **Dataset Properties**.



3. In the **Name** box, type **Expressions**

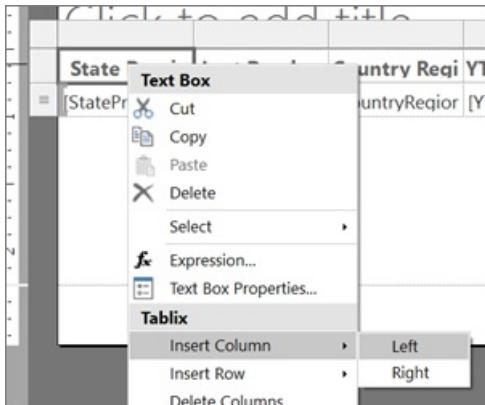
4. Click **OK**.

3. Display First Initial and Last Name

In this section, you use the **Left** function and the **Concatenate (&)** operator in an expression that evaluates to a name that includes an initial and a last name. You can build the expression step by step or skip ahead in the procedure and copy/paste the expression from the tutorial into the **Expression** dialog box.

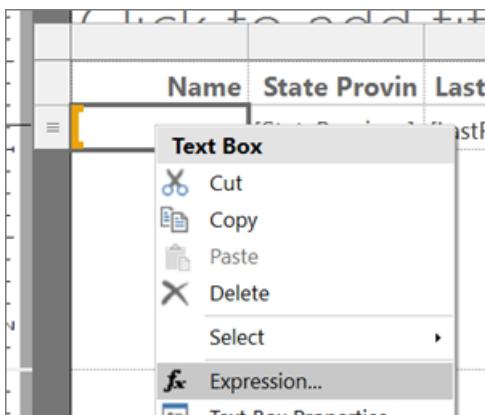
1. Right-click the **StateProvince** column, point to **Insert Column**, and then click **Left**.

A new column is added to the left of the **StateProvince** column.



2. Click the header of the new column and type **Name**.

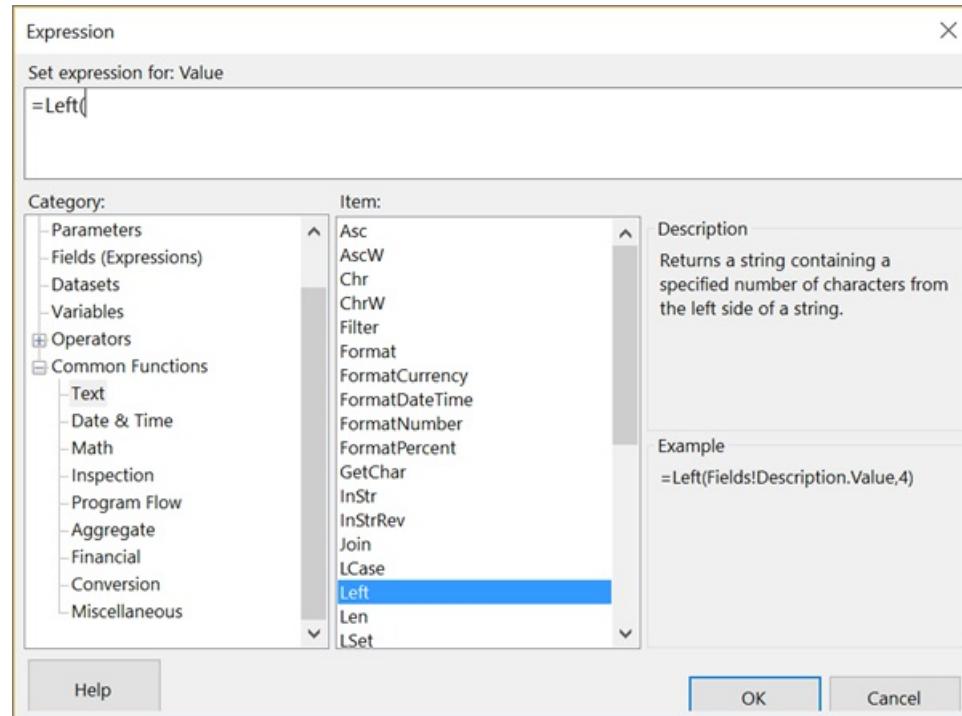
3. Right-click the data cell for the **Name** column and click **Expression**.



4. In the **Expression** dialog box, expand **Common Functions**, and then click **Text**.

5. In the **Item** list, double-click **Left**.

The **Left** function is added to the expression.



6. In the **Category** list, click **Fields (Expressions)**.

7. In the **Values** list, double-click **FirstName**.

8. Type **,** **1**

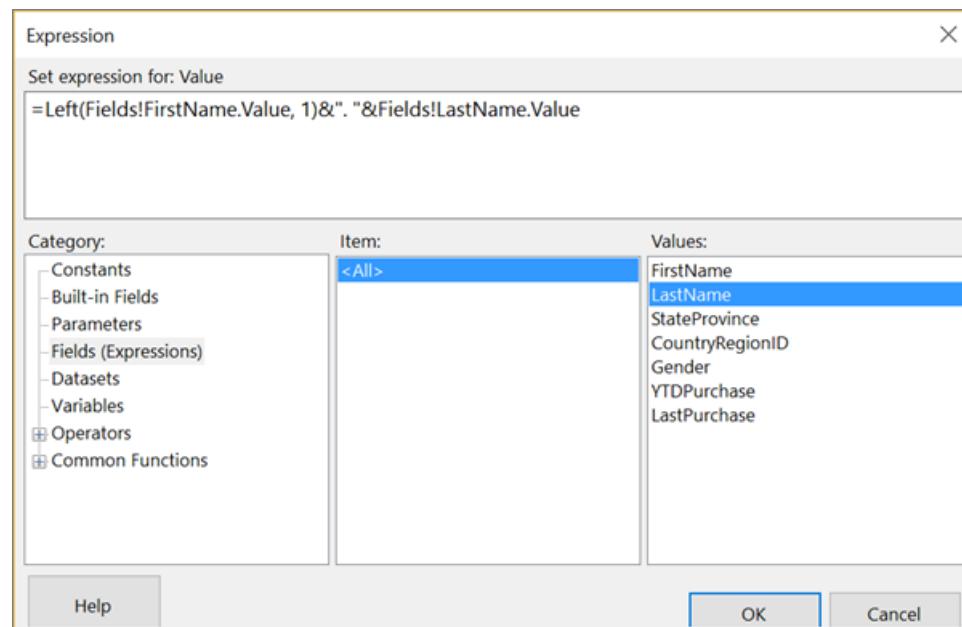
This expression extracts one character from the **FirstName** value, counting from the left.

9. Type **&". "&**

This adds a period and a space after the expression.

10. In the **Values** list, double-click **LastName**.

The completed expression is: `=Left(Fields!FirstName.Value, 1) &". "& Fields!LastName.Value`



11. Click **OK**.

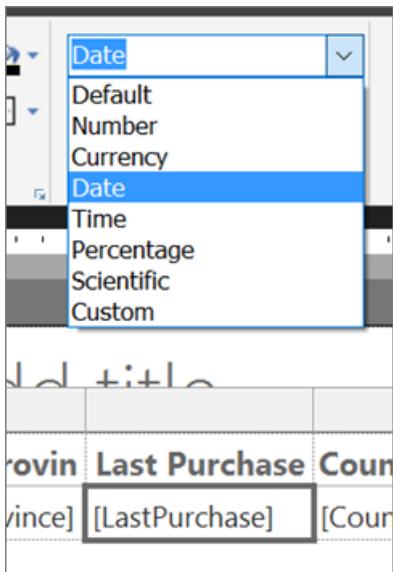
12. Click **Run** to preview the report.

(optional) Format the Date and Currency Columns and Header Row

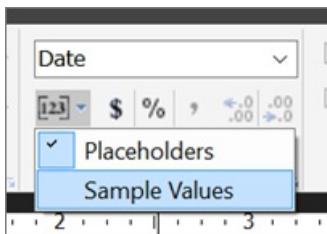
In this section, you format the **Last Purchase** column, which contains dates, and the **YTDPurchase** column, which contains currency. You also format the header row.

To format the date column

1. Click **Design** to return to design view.
2. Select the data cell in the **Last Purchase** column, and on the **Home** tab > **Number** section, select **Date**.



3. Also in the **Number** section, click the arrow next to **Placeholder Styles** and select **Sample Values**.



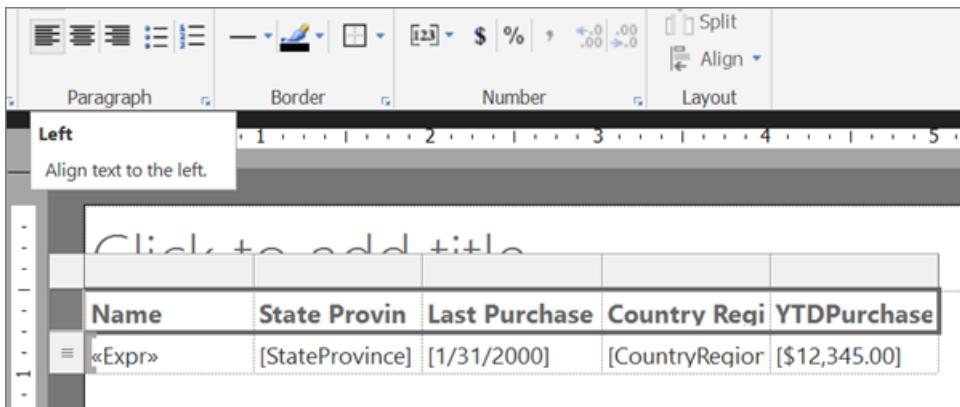
Now you can see an example of the formatting you selected.

To format the currency column

- Select the data cell in the **YTDPurchase** column, and in the **Number** section, select **Currency Symbol**.

To format the column headers

1. Select the row of column headers.
2. On the **Home** tab > **Paragraph** section, select **Left**.



3. Click **Run** to preview the report.

Here's the report so far, with formatted dates, currency, and column headers.

Name	State Province	Last Purchase	Country Region ID	YTDPurchase
A. Martin	Saxony	11/19/2015	4	\$2,997.60
A. Nath	Alaska	10/13/2015	7	\$607.50
B. Sanchez	North Dakota	9/17/2015	7	\$6,191.00
B. She	Hamburg	5/10/2015	4	\$7,497.30
C. Reed	Nebraska	8/27/2015	7	\$8,772.00
C. Petulescu	Wisconsin	11/30/2015	7	\$3,470.00
C. Randall	Utah	1/11/2015	7	\$7,218.10
F. Ross	Alberta	10/17/2015	3	\$9,248.15
G. Patterson	Kansas	10/18/2015	7	\$1,215.00
J. Bailey	British Columbia	6/15/2015	3	\$1,147.50
J. Peters	England	8/15/2015	12	\$887.50
K. Abel	Moselle	12/1/2015	6	\$2,025.00
L. Johnson	American Samoa	6/10/2015	1	\$9,996.60

4. Use Color to Display Gender

In this section, you add color to show the gender of a person. You will add a new column to display the color, and then determine the color that appears in the column based on the value of the Gender field.

To keep the color you've applied in that table cell when you make the report a banded report, you add a rectangle and then add the background color to the rectangle.

To add an M/F column

1. Right-click the **Name** column, point to **Insert Column**, and then click **Left**.

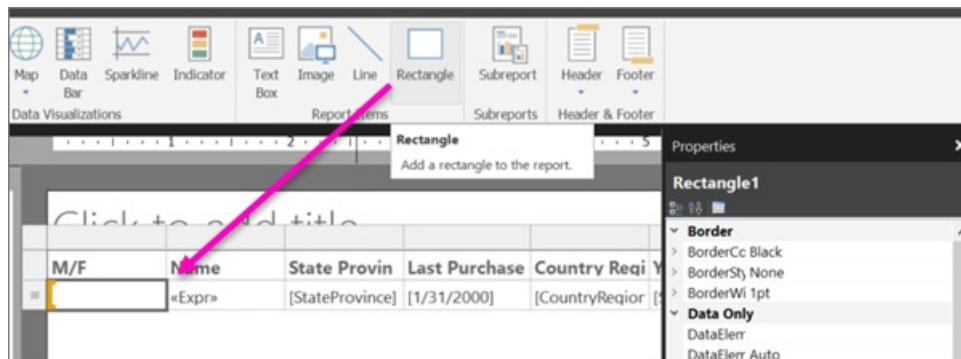
A new column is added to the left of the **Name** column.

2. Click the header of the new column and type **M/F**.

To add a rectangle

1. On the **Insert** tab, click **Rectangle** and then click in the data cell of the **M/F** column.

A rectangle is added to the cell.

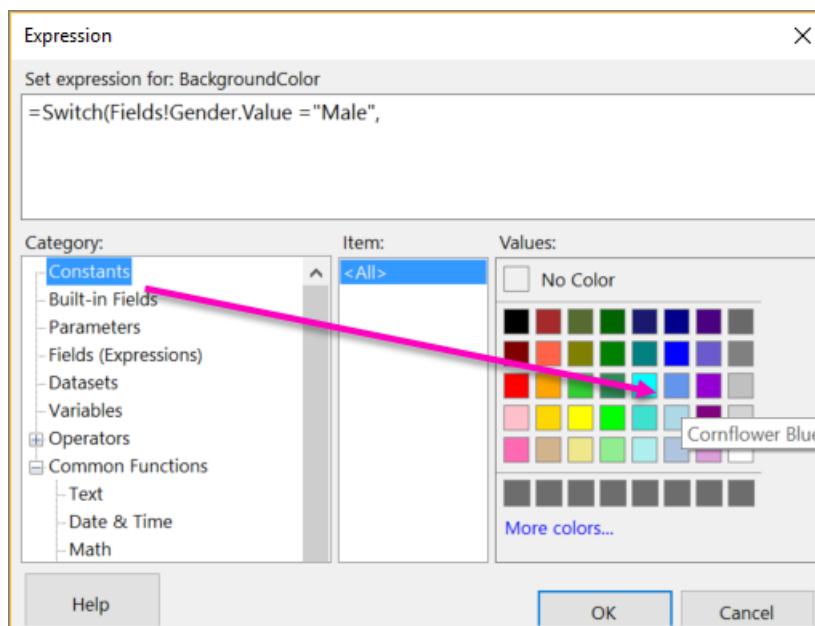


2. Drag the column divider between the **M/F** and the **Name** to make the **M/F** column narrower.



To use color to show gender

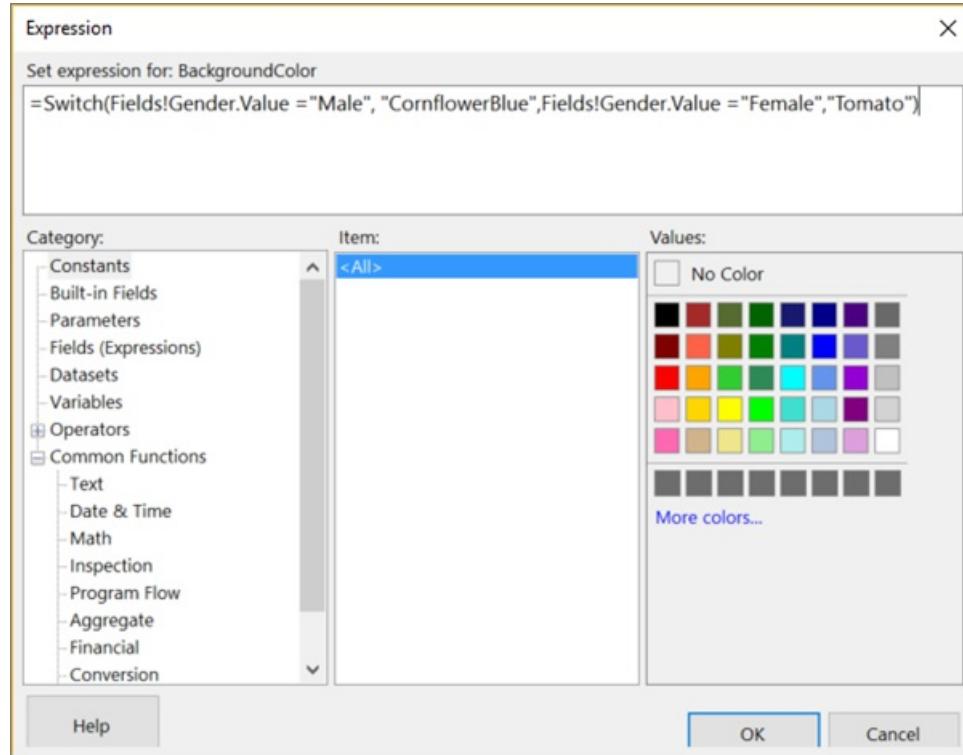
1. Right-click the rectangle in the data cell in the **M/F** column and click **Rectangle Properties**.
2. In the **Rectangle Properties** dialog box > **Fill** tab, click the expression **fx** button next to **Fill color**.
3. In the **Expression** dialog box, expand **Common Functions** and click **Program Flow**.
4. In the **Item** list, double-click **Switch**.
5. In the **Category** list, click **Fields (Expressions)**.
6. In the **Values** list, double-click **Gender**.
7. Type **= "Male"**, (including the comma).
8. In the **Category** list, click **Constants**, and in the **Values** box, click **Cornflower Blue**.



9. Type a comma after it.
10. In the **Category** list, click **Fields (Expressions)**, and in the **Values** list, double-click **Gender** again.
11. Type **= "Female"**, (including the comma).
12. In the **Category** list, click **Constants**, and in the **Values** box, click **Tomato**.
13. Type a closing parenthesis) after it.

The completed expression is:

```
=Switch(Fields!Gender.Value = "Male", "CornflowerBlue", Fields!Gender.Value = "Female", "Tomato")
```



14. Click **OK**, then click **OK** again to close the **Rectangle Properties** dialog box.
15. Click **Run** to preview the report.

The screenshot shows the Microsoft SQL Server Report Designer interface. At the top, there's a ribbon bar with tabs like File, Run, Design, Zoom, First, Previous, Next, Last, Refresh, Stop, Back, Print, Page Setup, Print Layout, and Print. Below the ribbon is a table with 12 rows of data. The first column, 'M/F', contains alternating blue and orange rectangular cells. The other columns are 'Name', 'State Province', 'Last Purchase', 'Country Region ID', and 'YTDPurchase'. The data includes names like A. Martin, A. Nath, B. Sanchez, etc., and purchase details like dates and amounts.

M/F	Name	State Province	Last Purchase	Country Region ID	YTDPurchase
A. Martin	A. Martin	Saxony	11/19/2015	4	\$2,997.60
A. Nath	A. Nath	Alaska	10/13/2015	7	\$607.50
B. Sanchez	B. Sanchez	North Dakota	9/17/2015	7	\$6,191.00
B. She	B. She	Hamburg	5/10/2015	4	\$7,497.30
C. Reed	C. Reed	Nebraska	8/27/2015	7	\$8,772.00
C. Petulescu	C. Petulescu	Wisconsin	11/30/2015	7	\$3,470.00
C. Randall	C. Randall	Utah	1/11/2015	7	\$7,218.10
F. Ross	F. Ross	Alberta	10/17/2015	3	\$9,248.15
G. Patterson	G. Patterson	Kansas	10/18/2015	7	\$1,215.00
J. Bailey	J. Bailey	British Columbia	6/15/2015	3	\$1,147.50
J. Peters	J. Peters	England	8/15/2015	12	\$887.50
K. Abel	K. Abel	Moselle	12/1/2015	6	\$2,025.00
L. Johnson	L. Johnson	American Samoa	6/10/2015	1	\$9,996.60

To format the color rectangles

1. Click **Design** to return to design view.
2. Select the rectangle in the **M/F** column. In the Properties pane, in the Border section, set these properties:
 - BorderColor = White
 - BorderStyle = Solid
 - BorderWidth = 5pt

This screenshot shows the Microsoft SQL Server Report Designer interface. It displays a table with several rows of data. The first column, 'M/F', contains a yellow rectangular cell. To the right, the 'Properties' pane is open, showing settings for 'Rectangle1'. Under the 'Border' section, the 'BorderColor' is set to 'White', 'BorderStyle' to 'Solid', and 'BorderWidth' to '5pt'. The 'Data Only' section is also visible.

3. Click **Run** to preview the report again. This time the color blocks have white space around them.

File	Run										
Design	Zoom	First	Previous	1 of 1	Next	Last	Refresh	Print	Page Setup	Print Layout	
	Views	Zoom					Stop				Print
M/F Name State Province Last Purchase Country Region ID YTDPurchase											
	A. Martin	Saxony		11/19/2015		4	\$2,997.60				
	A. Nath	Alaska		10/13/2015		7	\$607.50				
	B. Sanchez	North Dakota		9/17/2015		7	\$6,191.00				
	B. She	Hamburg		5/10/2015		4	\$7,497.30				
	C. Reed	Nebraska		8/27/2015		7	\$8,772.00				
	C. Petulescu	Wisconsin		11/30/2015		7	\$3,470.00				
	C. Randall	Utah		1/11/2015		7	\$7,218.10				
	F. Ross	Alberta		10/17/2015		3	\$9,248.15				
	G. Patterson	Kansas		10/18/2015		7	\$1,215.00				
	J. Bailey	British Columbia		6/15/2015		3	\$1,147.50				
	J. Peters	England		8/15/2015		12	\$887.50				
	K. Abel	Moselle		12/1/2015		6	\$2,025.00				
	L. Johnson	American Samoa		6/10/2015		1	\$9,996.60				

5. Look Up the CountryRegion Name

In this section, you create the CountryRegion dataset and use the **Lookup** function to display the name of a country/region instead of the identifier of the country/region.

To create the CountryRegion dataset

1. Click **Design** to return to design view.
2. In the Report Data pane, click **New** and then click **Dataset**.
3. In **Dataset Properties, click **Use a dataset embedded in my report**.
4. In the **Data source** list, select ExpressionsDataSource.
5. In the **Name** box, type **CountryRegion**
6. Verify that the **Text** query type is selected and click **Query Designer**.
7. Click **Edit as Text**.
8. Copy and paste the following query into the query pane:

```
SELECT 1 AS ID, 'American Samoa' AS CountryRegion
UNION SELECT 2 AS CountryRegionID, 'Australia' AS CountryRegion
UNION SELECT 3 AS ID, 'Canada' AS CountryRegion
UNION SELECT 4 AS ID, 'Germany' AS CountryRegion
UNION SELECT 5 AS ID, 'Micronesia' AS CountryRegion
UNION SELECT 6 AS ID, 'France' AS CountryRegion
UNION SELECT 7 AS ID, 'United States' AS CountryRegion
UNION SELECT 8 AS ID, 'Brazil' AS CountryRegion
UNION SELECT 9 AS ID, 'Mexico' AS CountryRegion
UNION SELECT 10 AS ID, 'Japan' AS CountryRegion
UNION SELECT 10 AS ID, 'Australia' AS CountryRegion
UNION SELECT 12 AS ID, 'United Kingdom' AS CountryRegion
```

9. Click **Run (!)** to run the query.

The query results are the country/region identifiers and names.

10. Click **OK**.

11. Click **OK** again to close the **Dataset Properties** dialog box.

Now you have a second dataset in the **Report Data** column.

To look up values in the CountryRegion dataset

1. Click the **Country Region ID** column header and delete the text: **ID**, so it reads **Country Region**.
2. Right-click the data cell for the **Country Region** column and click **Expression**.
3. Delete the expression except the initial equal (=) sign.

The remaining expression is:

4. In the **Expression** dialog box, expand **Common Functions** and click **Miscellaneous**, and in the **Item** list, double-click **Lookup**.
5. In the **Category** list, click **Fields (Expressions)**, and in the **Values** list, double-click **CountryRegionID**.
6. Place the cursor immediately after , and type **,Fields!ID.value,**
Fields!CountryRegion.value, "CountryRegion")

The completed expression:

```
=Lookup(Fields!CountryRegionID.Value,Fields!ID.value, Fields!CountryRegion.value, "CountryRegion")
```

The syntax of the **Lookup** function specifies a lookup between CountryRegionID in the Expressions dataset and ID in the CountryRegion dataset that returns the CountryRegion value from the CountryRegion dataset.

7. Click **OK**.

8. Click **Run** to preview the report.

6. Count Days Since Last Purchase

In this section, you add a column and then use the **Now** function or the **ExecutionTime** built-in global variable to calculate the number of days from today since a customer's last purchases.

To add the Days Ago column

1. Click **Design** to return to design view.
2. Right-click the **Last Purchase** column, point to **Insert Column**, and then click **Right**.

A new column is added to the right of the **Last Purchase** column.

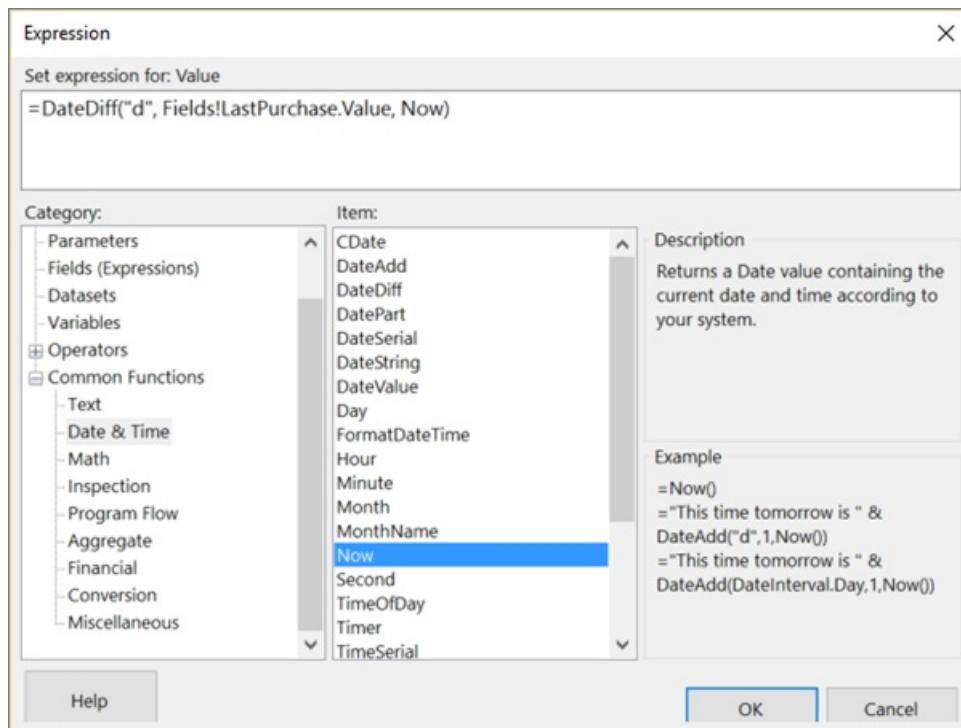
3. In the column header, type **Days Ago**
4. Right-click the data cell for the **Days Ago** column and click **Expression**.
5. In the **Expression** dialog box, expand **Common Functions**, and then click **Date & Time**.
6. In the **Item** list, double-click **DateDiff**.
7. Immediately after `DateDiff(`, type "d", (including the quotation marks "" and comma).
8. In the **Category** list, click **Fields (Expressions)**, and in the **Values** list, double-click **LastPurchase**.
9. Immediately after `Fields!LastPurchase.Value`, type , (a comma).
10. In the **Category** list, click **Date & Time** again, and in the **Item** list, double-click **Now**.

WARNING

In production reports you should not use the **Now** function in expressions that are evaluated multiple times as the report renders (for example, in the detail rows of a report). The value of **Now** changes from row to row and the different values affect the evaluation results of expressions, which leads to results that are subtly inconsistent. Instead, use the `ExecutionTime` global variable that Reporting Services provides.

11. Delete the left parenthesis after `Now(`, and then type a right parenthesis `)`

The completed expression is: `=DateDiff("d", Fields!LastPurchase.Value, Now)`



12. Click **OK**.
13. Click **Run** to preview the report.

7. Use an Indicator to Show Sales Comparison

In this section, you add a new column and use an indicator to show whether a person's year-to-date (YTD) purchases are above or below the average YTD purchases. The **Round** function removes decimals from values.

Configuring the indicator and its states takes many steps. If you want, you can skip ahead in the "To configure the indicator" procedure, and copy/paste the completed expressions from this tutorial into the **Expression** dialog box.

To add the + or - AVG Sales column

1. Right-click the **YTD Purchase** column, point to **Insert Column**, and then click **Right**.

A new column is added to the right of the **YTD Purchase** column.

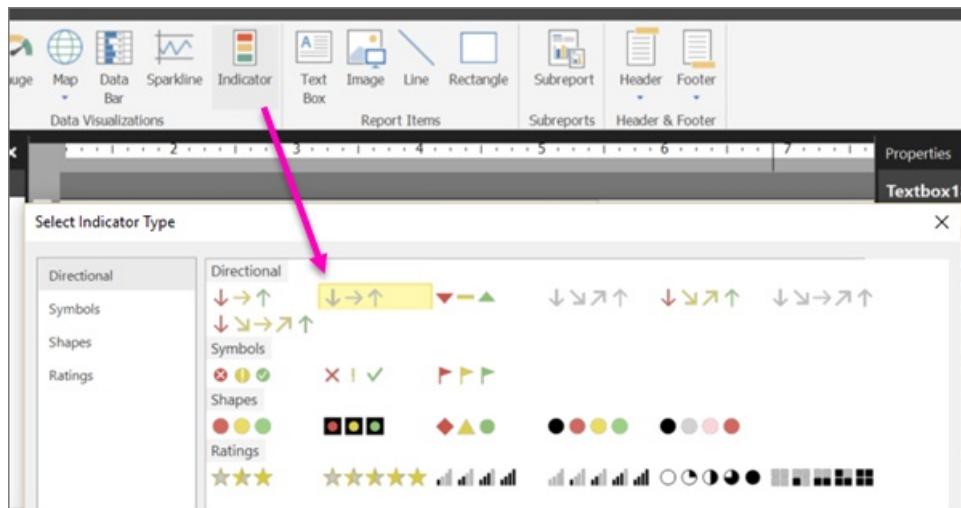
2. Click the column header and type **+ or - AVG Sales**

To add an indicator

1. On the **Insert** tab, click **Indicator**, and then click the data cell for the **+ or - AVG Sales** column.

The **Select Indicator Type** dialog box opens.

2. In the **Directional** group of icon sets, click the set of three gray arrows.



3. Click **OK**.

To configure the indicator

1. Right-click the indicator, click **Indicator Properties**, and then click **Value and States**.
2. Click the expression **fx** button next to the **Value** text box.
3. In the **Expression** dialog box, expand **Common Functions**, and then click **Math**.
4. In the **Item** list, double-click **Round**.
5. In the **Category** list, click **Fields (Expressions)**, and in the **Values** list, double-click **YTDPurchase**.
6. Immediately after `Fields!YTDPurchase.Value`, type `-` (a minus sign).
7. Expand **Common Functions** again, click **Aggregate**, and in the **Item** list, double-click **Avg**.
8. In the **Category** list, click **Fields (Expressions)**, and in the **Values** list, double-click **YTDPurchase**.
9. Immediately after `Fields!YTDPurchase.Value`, type `, "Expressions")`

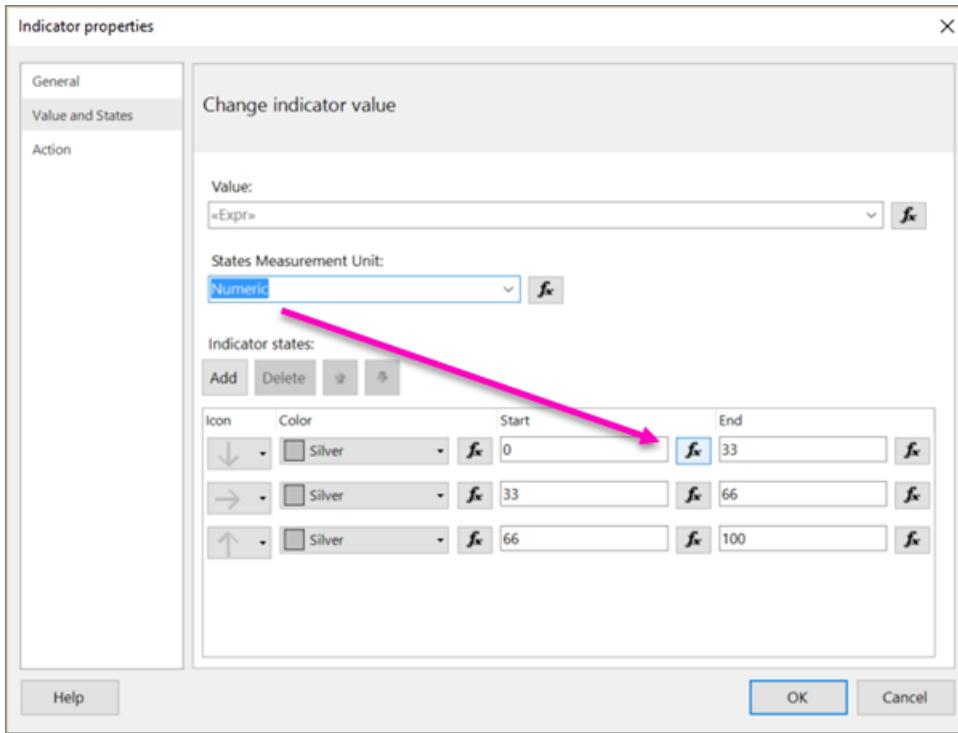
The completed expression is:

```
=Round(Fields!YTDPurchase.Value - Avg(Fields!YTDPurchase.Value, "Expressions"))
```

10. Click **OK**.

11. In the **States Measurement Unit** box, select **Numeric**.

12. In the row with the down-pointing arrow, click the **fx** button to the right of the text box for the **Start** value.



13. In the **Expression** dialog box, expand **Common Functions**, and then click **Math**.
14. In the **Item** list, double-click **Round**.
15. In the **Category** list, click **Fields (Expressions)**, and in the **Values** list, double-click **YTDPurchase**.
16. Immediately after `Fields!YTDPurchase.Value`, type - (a minus sign).
17. Expand **Common Functions** again and click **Aggregate**, and in the **Item** list, double-click **Avg**.
18. In the **Category** list, click **Fields (Expressions)**, and in the **Values** list, double-click **YTDPurchase**.
19. Immediately after `Fields!YTDPurchase.Value`, type `, type , "Expressions") < 0`

The completed expression:

```
=Round(Fields!YTDPurchase.Value - Avg(Fields!YTDPurchase.Value, "Expressions")) < 0
```

20. Click **OK**.
21. In the text box for the **End** value, type **0**
22. Click the row with the horizontal-pointing arrow and click **Delete**.



Now there are only two arrows, either up or down.

23. In the row with the up-pointing arrow, in the **Start** box, type **0**
24. Click the **fx** button to the right of the text box for the **End** value.
25. In the **Expression** dialog box, delete **100** and create the expression:

```
=Round(Fields!YTDPurchase.Value - Avg(Fields!YTDPurchase.Value, "Expressions")) >0
```

26. Click **OK**.
27. Click **OK** again to close the **Indicator properties** dialog box.
28. Click **Run** to preview the report.

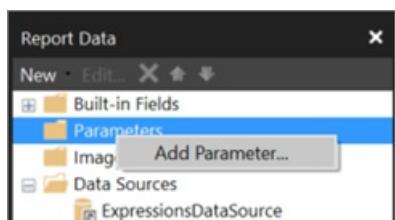
M/F	Name	State Province	Last Purchase	Days Ago	Country Region	YTD Purchase	+ or - AVG Sales
A. Martin	A. Martin	Saxony	11/19/2015	301	Germany	\$2,997.60	↓
A. Nath	A. Nath	Alaska	10/13/2015	338	United States	\$607.50	↓
B. Sanchez	B. Sanchez	North Dakota	9/17/2015	364	United States	\$6,191.00	↑
B. She	B. She	Hamburg	5/10/2015	494	Germany	\$7,497.30	↑
C. Reed	C. Reed	Nebraska	8/27/2015	385	United States	\$8,772.00	↑
C. Petulescu	C. Petulescu	Wisconsin	11/30/2015	290	United States	\$3,470.00	↓
C. Randall	C. Randall	Utah	1/11/2015	613	United States	\$7,218.10	↑
F. Ross	F. Ross	Alberta	10/17/2015	334	Canada	\$9,248.15	↑
G. Patterson	G. Patterson	Kansas	10/18/2015	333	United States	\$1,215.00	↓
J. Bailey	J. Bailey	British Columbia	6/15/2015	458	Canada	\$1,147.50	↓
J. Peters	J. Peters	England	8/15/2015	397	United Kingdom	\$887.50	↓
K. Abel	K. Abel	Moselle	12/1/2015	289	France	\$2,025.00	↓
L. Johnson	L. Johnson	American Samoa	6/10/2015	463	American Samoa	\$9,996.60	↑
M. Zimmerman	M. Zimmerman	Moselle	11/16/2015	304	France	\$1,200.00	↓
N. Anand	N. Anand	Seine (Paris)	12/11/2015	279	France	\$1,425.00	↓
O. Romeo	O. Romeo	Texas	7/29/2015	414	United States	\$8,578.00	↑
R. Roberts	R. Roberts	Washington	10/28/2015	323	United States	\$8,357.80	↑
R. Caron	R. Caron	Northwest Territories	4/29/2015	505	Canada	\$742.50	↓
W. Pal	W. Pal	New South Wales	7/3/2015	440	Australia	\$5,747.25	↑
Y. Sharma	Y. Sharma	Micronesia	8/23/2015	389	Micronesia	\$3,247.95	↓

8. Make a Banded Report

Create a parameter so report readers can specify the color to apply to alternating rows in the report, making it a banded report.

To add a parameter

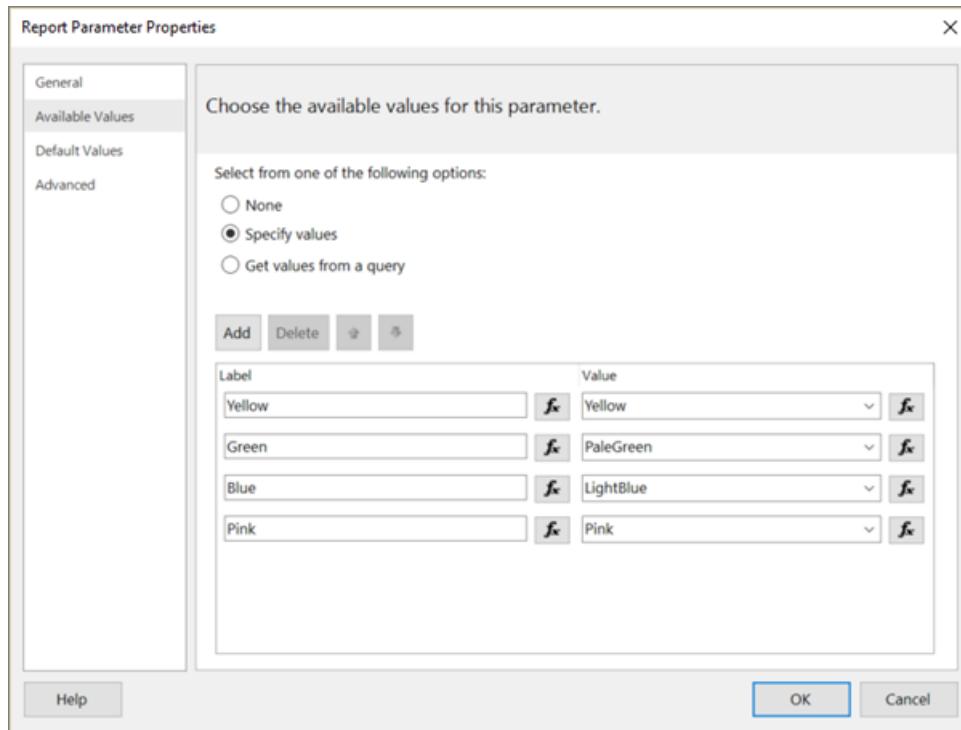
1. Click **Design** to return to design view.
2. In the **Report Data** pane, right-click **Parameters** and click **Add Parameter...**



The **Report Parameter Properties** dialog box opens.

3. In **Prompt**, type **Choose color**
4. In **Name**, type **RowColor**
5. On the **Available Values** tab, click **Specify values**.
6. Click **Add**.
7. In the **Label** box, type **Yellow**
8. In the **Value** box, type **Yellow**

9. Click **Add**.
10. In the **Label** box, type **Green**
11. In the **Value** box, type **PaleGreen**
12. Click **Add**.
13. In the **Label** box, type **Blue**
14. In the **Value** box, type **LightBlue**
15. Click **Add**.
16. In the **Label** box, type **Pink**
17. In the **Value** box, type **Pink**



18. Click **OK**.

To apply alternating colors to detail rows

1. Select all the cells in the data row except the cell in the **M/F** column, which has its own background color.

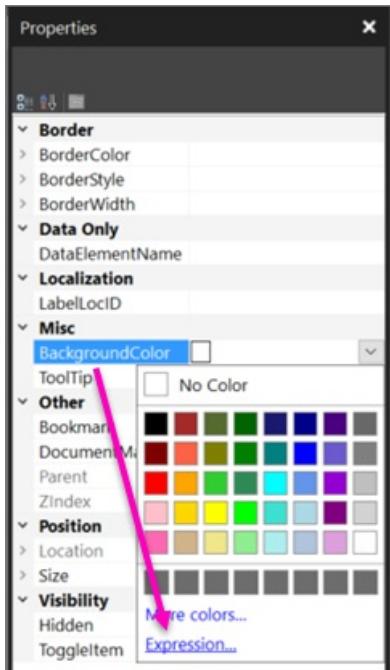
A screenshot of a data grid with a single row. The columns are labeled: M/F, Name, State Provin, Last Purchase, Days Ago, Country Region, YTDPurchase + or - AVG S. The first column (M/F) contains an expression: «Expr». The other columns contain text: [StateProvince], [1/31/2000], «Expr», «Expr», [\$12,345.00]. The entire row is highlighted with a light gray background.

2. In the Properties pane, click **BackgroundColor**.

If you don't see the Properties pane, on the **View** tab select the **Properties** box.

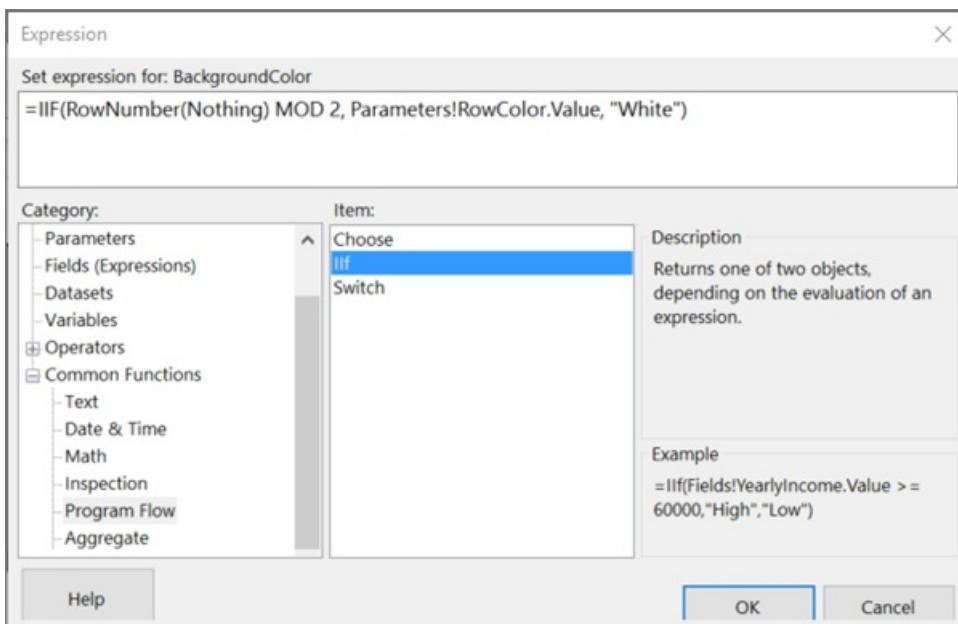
If the properties are listed by category in the Properties pane, you will find **BackgroundColor** in the **Misc** category.

3. Click the down arrow and then click **Expression**.



4. In the **Expression** dialog box, expand **Common Functions**, and then click **Program Flow**.
5. In the **Item** list, double-click **IIf**.
6. Under **Common Functions**, click **Miscellaneous**, and in the **Item** list, double-click **RowNumber**.
7. Immediately after **RowNumber(** type **Nothing) MOD 2,**
8. Click **Parameters** and in the **Values** list, double-click **RowColor**.
9. Immediately after **Parameters!RowColor.Value**, type **,** **"White"**)

The completed expression is: `=IIF(RowNumber(Nothing) MOD 2, Parameters!RowColor.Value, "White")`



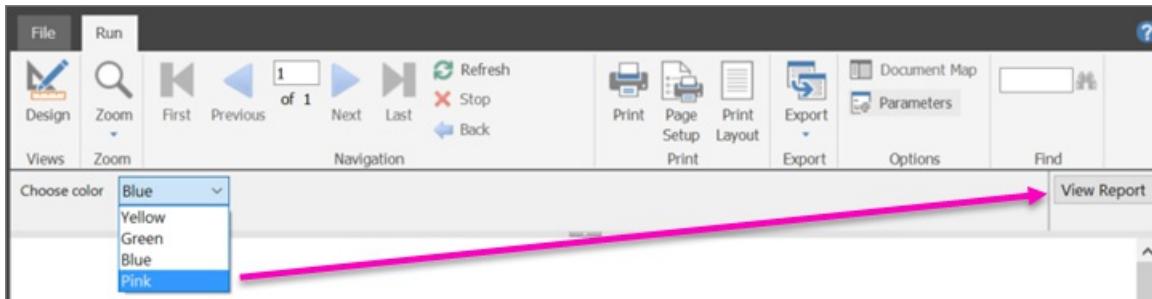
10. Click **OK**.

Run the Report

1. On the **Home** tab, click **Run**.

Now when you run the report, you don't see the report until you choose a color for the non-white bands.

2. In the **Choose color** list, select a color for the non-white bands in the report.



3. Click **View Report**.

The report renders and alternating rows have the background that you chose.

M/F	Name	State Province	Last Purchase Days Ago	Country Region	YTD Purchase + or - AVG Sales
Blue	A. Martin	Saxony	11/19/2015	301 Germany	\$2,997.60 ↓
Red	A. Nath	Alaska	10/13/2015	338 United States	\$607.50 ↓
Red	B. Sanchez	North Dakota	9/17/2015	364 United States	\$6,191.00 ↑
Red	B. She	Hamburg	5/10/2015	494 Germany	\$7,497.30 ↑
Blue	C. Reed	Nebraska	8/27/2015	385 United States	\$8,772.00 ↑
Blue	C. Petulescu	Wisconsin	11/30/2015	290 United States	\$3,470.00 ↓
Red	C. Randall	Utah	1/11/2015	613 United States	\$7,218.10 ↑
Blue	F. Ross	Alberta	10/17/2015	334 Canada	\$9,248.15 ↑
Red	G. Patterson	Kansas	10/18/2015	333 United States	\$1,215.00 ↓
Blue	J. Bailey	British Columbia	6/15/2015	458 Canada	\$1,147.50 ↓
Blue	J. Peters	England	8/15/2015	397 United Kingdom	\$887.50 ↓
Red	K. Abel	Moselle	12/1/2015	289 France	\$2,025.00 ↓
Red	L. Johnson	American Samoa	6/10/2015	463 American Samoa	\$9,996.60 ↑

(optional) Add a Report Title

Add a title to the report.

To add a report title

- On the design surface, click **Click to add title**.
- Type **Sales Comparison Summary**, then select the text.
- On the **Home** tab, in the **Font** box, set:
 - Size = 18
 - Color = Gray
 - Bold
- On the **Home** tab, click **Run**.
- Select a color for the non-white bands in the report, and click **View Report**.

(optional) Save the Report

You can save reports to a report server, SharePoint library, or your computer. For more information, see [Saving](#)

Reports (Report Builder).

In this tutorial, you save the report to a report server. If you do not have access to a report server, save the report to your computer.

To save the report to a report server

1. On the **File** menu > **Save As**.
2. Click **Recent Sites and Servers**.
3. Select or type the name of the report server where you have permission to save reports.

The message "Connecting to report server" appears. When the connection is complete, you will see the contents of the report folder that the report server administrator specified as the default report location.

4. Give the report a name and click **Save**.

The report is saved to the report server. The name of report server that you are connected to appears in the status bar at the bottom of the window.

Now your report readers can view your report in the Reporting Services web portal.

The screenshot shows the SQL Server Reporting Services web interface. At the top, it displays "SQL Server Reporting Services" and the user "Maggie Sparkman". Below the header, there are navigation links for "Favorites" and "Browse", and a breadcrumb trail showing "Home > RB-expressions-tutorial". A color selection dropdown is set to "Blue", and a "View Report" button is visible. The main content area is titled "Sales Comparison Summary". It features a table with the following data:

M/F	Name	State Province	Last Purchase	Days Ago	Country Region	YTD Purchase + or - AVG Sales	Indicator
■	A. Martin	Saxony	11/19/2015	301	Germany	\$2,997.60	↓
■	A. Nath	Alaska	10/13/2015	338	United States	\$607.50	↓
■	B. Sanchez	North Dakota	9/17/2015	364	United States	\$6,191.00	↑
■	B. She	Hamburg	5/10/2015	494	Germany	\$7,497.30	↑
■	C. Reed	Nebraska	8/27/2015	385	United States	\$8,772.00	↑
■	C. Petulescu	Wisconsin	11/30/2015	290	United States	\$3,470.00	↓
■	C. Randall	Utah	1/11/2015	613	United States	\$7,218.10	↑
■	F. Ross	Alberta	10/17/2015	334	Canada	\$9,248.15	↑
■	G. Patterson	Kansas	10/18/2015	333	United States	\$1,215.00	↓
■	J. Bailey	British Columbia	6/15/2015	458	Canada	\$1,147.50	↓
■	J. Peters	England	8/15/2015	397	United Kingdom	\$887.50	↓
■	K. Abel	Moselle	12/1/2015	289	France	\$2,025.00	↓
■	L. Johnson	American Samoa	6/10/2015	463	American Samoa	\$9,996.60	↑
■	M. Zimmerman	Moselle	11/16/2015	304	France	\$1,200.00	↓
■	N. Anand	Seine (Paris)	12/11/2015	279	France	\$1,425.00	↓
■	O. Romeo	Texas	7/29/2015	414	United States	\$8,578.00	↑
■	R. Roberts	Washington	10/28/2015	323	United States	\$8,357.80	↑
■	R. Caron	Northwest Territories	4/29/2015	505	Canada	\$742.50	↓
■	W. Pal	New South Wales	7/3/2015	440	Australia	\$5,747.25	↑
■	Y. Sharma	Micronesia	8/23/2015	389	Micronesia	\$3,247.95	↓

See Also

[Expressions \(Report Builder and SSRS\)](#)

[Expression Examples \(Report Builder and SSRS\)](#)

[Indicators \(Report Builder and SSRS\)](#)

[Images, Text Boxes, Rectangles, and Lines \(Report Builder and SSRS\)](#)

[Tables \(Report Builder and SSRS\)](#)

[Report Datasets \(SSRS\)](#)