SQL Server Reporting Services

SQL Server Reporting Services is one of the most popular components of SQL Server, but it has always been surprisingly difficult to get from one place all the basic facts you need to get up and running from scratch, to the point of producing reports.

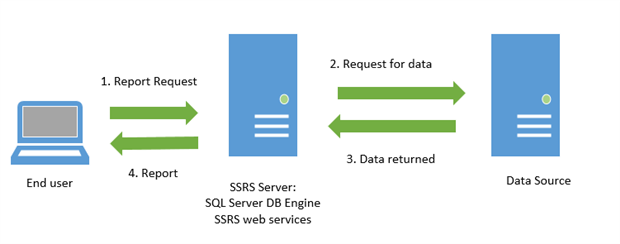
SSRS Architecture

This series of articles focuses on the development of reports. Keep in mind that SSRS can be deployed to SharePoint, but this series of article only covers the native deployment architecture.

An SSRS deployment must be associated with a SQL Server instance. On the instance will be two databases, by default:

* ReportServer – contains the report definitions, configuration, history, security of deployed reports and more
* ReportServerTempdb – much like tempdb, it is used as a workspace for building reports and doesn’t maintain any objects permanently.

We will also need a location for the [Report Server Web Service](http://technet.microsoft.com/en-us/library/ms152787.aspx), which can be on the same server as the databases, as in the simple deployment architecture shown in Figure 1, or on a different server. On whichever server we choose, we will have access to a [Report Manager](http://technet.microsoft.com/en-us/library/ms157147.aspx) website that allows us to deploy and manage the reports. End users can run reports from Report Manager, create subscriptions, and publish their own reports if they have permission.

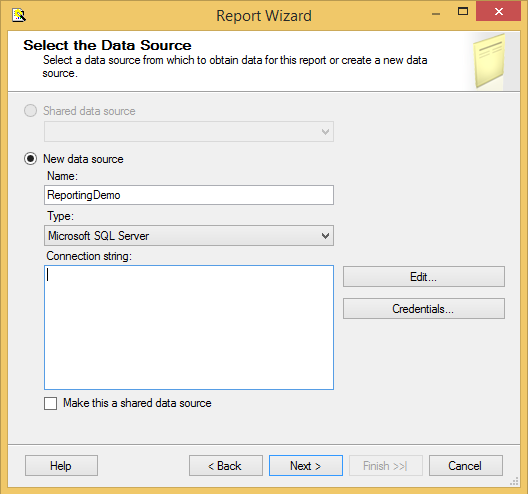


Our very first report project, **FirstProject**, demonstrates how to create a report with grouping levels including one of the dynamic features, collapsing and expanding sections. The report is essentially just a list of customers that we’ll group by state. The end result is not perfect, but it is a good start.

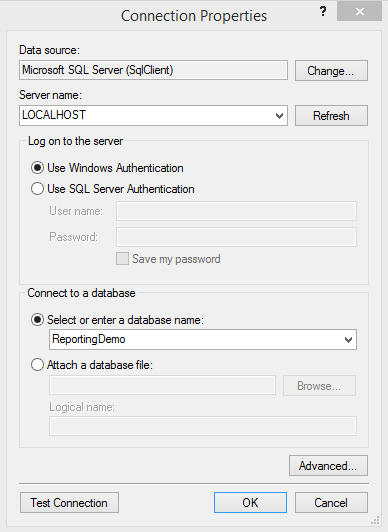
Open and create a new project. Select the **Report Server Project Wizard** type. This will open up the **New Project** dialog. Create a new **Reporting Services** project called **FirstProject**. Click **OK** to start up the new report wizard. We can also kick off the wizard from within an existing project by right-clicking the **Reports** folder in **Solution Explorer** and selecting **Add New Report**.

The first dialog of note is **Select the Data Source**. Since this is our first data source, our only option is to create a new one.

Select the **New data source** radio button and give the data source a name, usually referring to the database name, so in this case ReportingDemo. Leave the data source type as the default (**Microsoft SQL Server**)



Click the **Edit** button to bring up the **Connection Properties** dialog. Enter the name of the SQL Server instance hosting the data source, in this case the ReportingDemo database. The default option is to log on using Windows Authentication. If you are using SQL Server Authentication, choose that setting, and enter the username and password. Finally, select ReportingDemo and make sure you test the connection before you click **OK**.

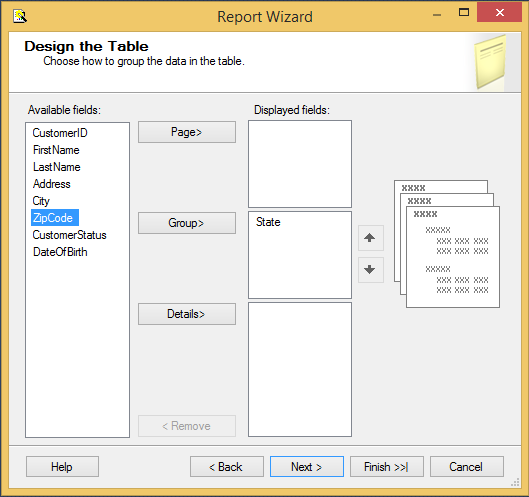


Next up is the **Design the Query** screen. We can use the Query Builder by clicking the button at the top left but, in this case, simply enter SELECT \* FROM Customer; into the Query string text box and click **Next**.

On the **Select the Report Type** dialog, we can choose between a tabular or matrix report. A tabular report is a traditional grid with column headings and rows of data, and it might contain grouping sections at the row level. A matrix report is like a pivot table. It can have column headings that expand across the top of the report. It can have grouping sections at the row and column levels. We’re going to start with a simple tabular report so choose **Tabular** and hit **Next**.

This brings up the **Design the Table** screen, where we specify what customer data we wish to include in our report, and how we wish to group it. Essentially, we need to specify how we will use each field (column) in the report. We might display some fields at the top of each page, use others for grouping, and others will form the detail level of the report.

In this case, we simply want to group the customer data by state, so select State in the **Available fields** box and click the **Group** button. If you group by more than one field, then make sure the fields are ordered (use the Up and Down arrows) to reflect the grouping level hierarchy you wish to see in the report. For example, State would be higher than City.

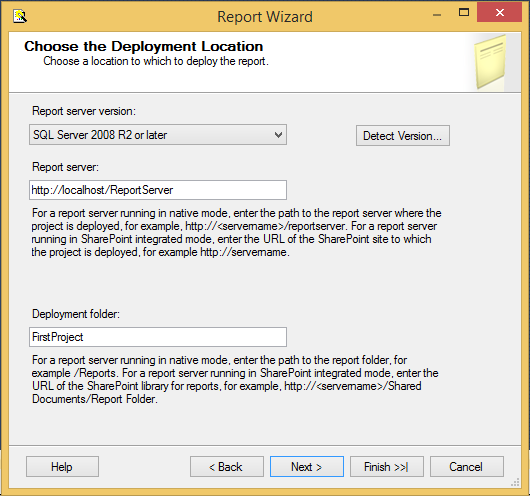


On the **Choose the Table Layout** page, we have the choice of **Stepped** or **Block**. I don’t see a lot of difference between these layouts. After some experimentation, I found that the Stepped report had a row dedicated to the group label and that row was formatted with a background color. The **Block**report shows the group label on the first row of the detail and no special background. With the Block report, we do not have the option for drilldowns.

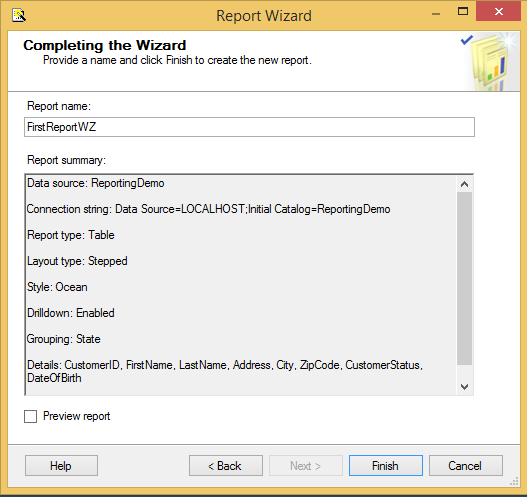
Select the **Stepped** option and check the **Enable Drilldown** checkbox, which will allow us to collapse and expand the data by state (the grouping column)

On the next screen, choose a style for your report (I chose Ocean).

The **Choose the Deployment Location** screen allows us to specify the Report Server to which we wish to deploy the reports in the project. For now, we will be working within SSDT-BI and not deploying the reports to the server, so we can just accept the defaults. When we are ready to deploy the reports, we can revisit these settings by right-clicking the project name and selecting **Properties**.



This brings up the final screen, **Completing the Wizard**, which simply summarizes our chosen report options, and lets us name the report and preview it. Name the report **ReportWZ** and click **Finish** to end the wizard.



After the wizard completes, we will see the report in the **Design** tab of the standard Report Designer.

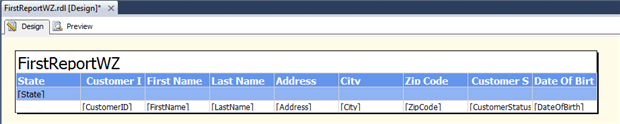
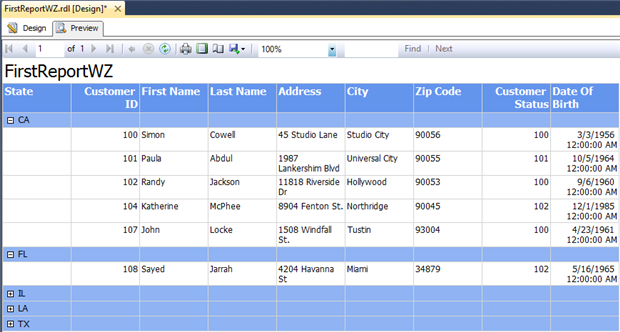


Figure 14

Click on the **Preview** tab to view the report. The **Preview** tab allows us to run the report from SSDT-BI, without having to publish it first to the Report Server. If the report takes parameters, the Preview tab will ask us to fill them out before it runs the report. Since this report doesn’t take any arguments, it will display immediately.



# Adding parameters to a report

There are two ways to add parameters to our reports. The first is simply to define parameters in the queries in the dataset, and then these will automatically be available as report parameters. The second way is to manually add report parameters through the **Parameters** section of the Report Data window.

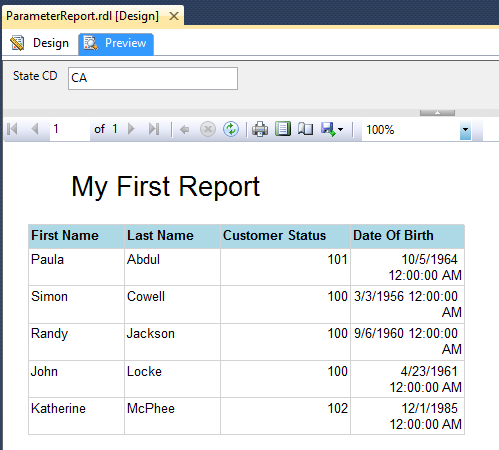
We can use parameters to filter the data at the source, or for other reasons like giving the end-user control over one of the properties. We could also have a parameter that is used for both functions.

## Parameters created by the dataset query

First, we will set up parameters using a dataset query. You can either continue working with the same report or create a new one that is a copy of the original.

To create a copy, right-click on **FirstReportMan** in the Solution Explorer and select **Copy.**Then right-click on the project name and select **Paste**. Rename the new report **ParameterReport**. Double-click the new report in Solution Explorer to open it in the designer. In the Report Data window, navigate to the **Customers** dataset and double-click it to bring up the properties. The Dataset dialog box opens and allows us to edit the query for the dataset. Change the stored procedure from spr\_CustomerSelectAll to spr\_CustomerSelectByState. A parameter called @StateCDshould appear in the **Parameters** folder. This parameter allows us to filter the data from the Customer table by State. Again, the ReportingDemoDatabaseScript.sql script in the code download includes the creation script for spr\_CustomerSelectByState.

Switch to the Preview tab to view the report. Instead of the report just running and displaying all the data from the data source, there should be a place for us to enter a value for the @StateCDparameter. Enter “CA” and click the **View Report** button to see only Californian customers.

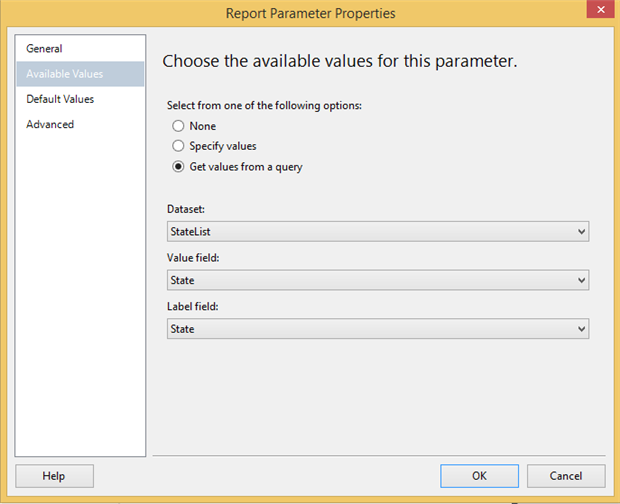


To edit existing parameters, or to add new ones, switch to the Design tab of the Report Designer and expand the **Parameters** folder. Double-click the StateCD parameter to bring up the properties. First, we are going to make the prompt a little more friendly. On the **General** page, change the **Prompt** value to **State**.

We can also provide a list of possible values for the parameter. Select the **Available Values** page and from here we can provide either a hard-coded list, or connect the parameter to a dataset. The second method is the generally the best so that we can avoid the need to maintain the list manually.

Close the parameter properties. To connect a parameter to a dataset, we first need to create a new dataset. Using the skills you have learned so far, add a dataset to the report called **StateList** based on this query: SELECT DISTINCT State FROM Customer ORDER BY State;.

Open the parameter properties once again and select **Available Values**. Choose **Get values from a query.** Under **Dataset**, select **StateList**. For the **Value** and **Label** fields, select **State**.



In this case, the **Value** and **Label** fields are the same. The **Value** field is the field that the database needs for the query. The **Label** field is the field that the end user should see. For example, if we have a list that has an ID and a description, the ID would be the **Value** field while the description would be the **Label** field.

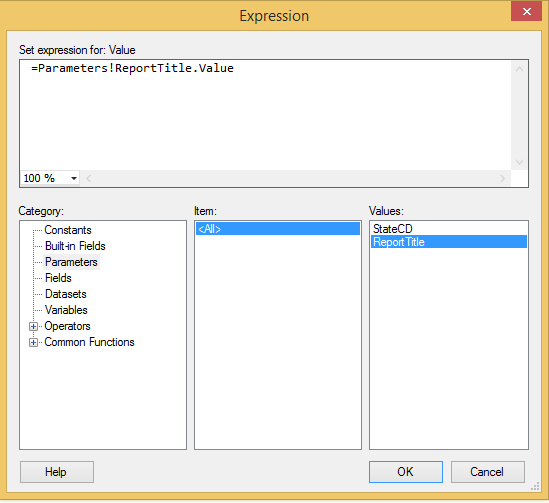
## Manually created parameters

#### Tip: Shared data sources, embedded datasets

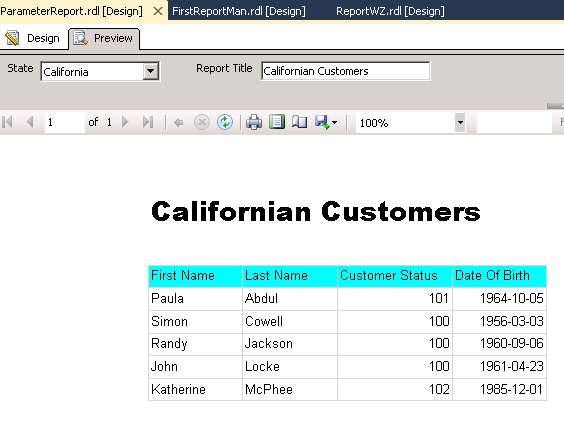
Generally, data sources are best shared and datasets are best embedded. However, queries to populate parameter lists are good examples of datasets you may want to share instead, since they can often be used for several reports.

Now let’s add a parameter that is not defined in a query and, therefore, must be created manually. In the Report Data window, click **Add** | **New Parameter**. Give the new parameter the name ReportTitle, and enter “Report Title” for the **Prompt**. Its data type should be **Text**. Click **OK**.

Back on the Design tab, right-click the report header textbox and choose the **Expression** option. Remove the original “My First Report” text and select **Parameters** under **Categories**. Double-click on the ReportTitle parameter. This text box will now display the value we pass to the ReportTitle parameter. Click **OK**.



Switch to the Preview tab and try it out.



# Add Cascading Parameters to a Report

Cascading parameters provide a way of managing large amounts of report data. You can define a set of related parameters so that the list of values for one parameter depends on the value chosen in another parameter. For example, the first parameter is independent and might present a list of product categories. When the user selects a category, the second parameter is dependent on the value of the first parameter. Its values are updated with a list of subcategories within the chosen category. When the user views the report, the values for both the category and subcategory parameters are used to filter report data

To create cascading parameters, you define the dataset query first and include a query parameter for each cascading parameter that you need. You must also create a separate dataset for each cascading parameter to provide available values. For more information, see [Add, Change, or Delete Available Values for a Report Parameter (Report Builder and SSRS)](https://docs.microsoft.com/en-us/sql/reporting-services/report-design/add-change-or-delete-available-values-for-a-report-parameter?view=sql-server-2017).

Order is important for cascading parameters because the dataset query for a parameter later in the list includes a reference to each parameter that is earlier in the list. At run time, the order of the parameters in the Report Data pane determines the order in which the parameter queries appear in the report, and therefore, the order in which a user chooses each successive parameter value.

## To create the main dataset with a query that includes multiple related parameters

1. In the Report Data pane, right-click a data source, and then click Add Dataset.
2. In Name, type the name of the dataset.
3. In Data source, choose the name of the data source or click New to create one.
4. In Query type, choose the type of query for the selected data source. In this topic, query type Text is assumed.
5. In Query, type the query to use to retrieve data for this report. The query must include the following parts:
   1. A list of data source fields. For example, in a Transact-SQL statement, the SELECT statement specifies a list of database column names from a given table or view.
   2. One query parameter for each cascading parameter. A query parameter limits the data retrieved from the data source by specifying certain values to include or exclude from the query. Typically, query parameters occur in a restriction clause in the query. For example, in a Transact-SQL SELECT statement, query parameters occur in the WHERE clause.
6. Click Run (!). After you include query parameters and then run the query, report parameters that correspond to the query parameters are automatically created.

## To create a dataset to provide values for an independent parameter

1. In the Report Data pane, right-click a data source, and then click Add Dataset.
2. In Name, type the name of the dataset.
3. In Data source, verify the name is the name of the data source you chose in step 1.
4. In Query type, choose the type of query for the selected data source. In this topic, query type Text is assumed.
5. In Query, type the query to use to retrieve values for this parameter. Queries for independent parameters typically do not contain query parameters. For example, to create a query for a parameter that provides all category values, you might use a Transact-SQL statement similar to the following:

SELECT DISTINCT <column name> FROM <table>

The SELECT DISTINCT command removes duplicate values from the result set so that you get each unique value from the specified column in the specified table.

Click Run (!). The result set shows the values that are available for this first parameter.

1. Click OK.

Next, you will set the properties of the first parameter to use this dataset to populate its available values at run-time.

## To set available values for a report parameter

1. In the Report Data pane, in the Parameters folder, right-click the first parameter, and then click Parameter Properties.
2. In Name, verify that the name of the parameter is correct.
3. Click Available Values.
4. Click Get values from a query. Three fields appear.
5. In Dataset, from the drop-down list, click the name of the dataset you created in the previous procedure.
6. In Value field, click the name of the field that provides the parameter value.
7. In Label field, click the name of the field that provides the parameter label.
8. Click OK.

Next, you will create a dataset that provides the values for a dependent parameter.

## To create a dataset to provide values for a dependent parameter

1. In the Report Data pane, right-click a data source, and then click Add Dataset.
2. In Name, type the name of the dataset.
3. In Data source, verify the name is the name of the data source you chose in step 1.
4. In Query type, choose the type of query for the selected data source. In this topic, query type Text is assumed.
5. In Query, type the query to use to retrieve values for this parameter. Queries for dependent parameters typically include query parameters for each parameter that this parameter is dependent on. For example, to create a query for a parameter that provides all subcategory (dependent parameter) values for a category (independent parameter), you might use a Transact-SQL statement similar to the following:

SELECT DISTINCT Subcategory FROM <table>

WHERE (Category = @Category)

In the WHERE clause, Category is the name of a field from <table> and @Category is a query parameter. This statement produces a list of subcategories for the category specified in @Category. At run time, this value will be filled in with the value that the user chooses for the report parameter that has the same name.

1. Click OK.

Next, you will set the properties of the second parameter to use this dataset to populate its available values at run time.

## To set available values for a report parameter

1. In the Report Data pane, in the Parameters folder, right-click the first parameter, and then click Parameter Properties.
2. In Name, verify that the name of the parameter is correct.
3. Click Available Values.
4. Click Get values from a query.
5. In Dataset, from the drop-down list, click the name of the dataset you created in the previous procedure.
6. In Value field, click the name of the field that provides the parameter value.
7. In Label field, click the name of the field that provides the parameter label.
8. Click OK.

## To test the cascading parameters

1. Click Run.
2. From the drop-down list for the first, independent parameter, choose a value. The report processor runs the dataset query for the next parameter and passes it the value you chose for the first parameter. The drop-down list for the second parameter is populated with the available values based on the first parameter value.
3. From the drop-down list for the second, dependent parameter, choose a value. The report does not run automatically after you choose the last parameter so that you can change your choice.
4. Click View Report. The report updates the display based on the parameters you have chosen.

Expression

Expressions are used frequently in Reporting Services paginated reports to control content and report appearance. Expressions are written in Microsoft Visual Basic, and can use built-in functions, custom code, report and group variables, and user-defined variables. Expressions begin with an equal sign (=)

Example:

### Date Functions

* The Today function provides the current date. This expression can be used in a text box to display the date on the report, or in a parameter to filter data based on the current date.

=Today()

=DatePart(DateInterval.WeekOfYear, today())

This expression will show the number of the week in the current year for today's date.

We can change almost everything on the report by using expression like, font color, format etc.

# Tables, Matrices, and Lists

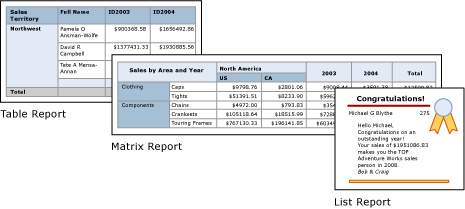
In Reporting Services, tables, matrices, and lists are data regions that display paginated report data in cells that are organized into rows and columns. The cells typically contain text data such as text, dates, and numbers but they can also contain gauges, charts, or report items such as images. Collectively, tables, matrices, and lists are frequently referred to as tablix data regions.

The table, matrix, and list templates are built on the tablix data region, which is a flexible grid that can display data in cells. In the table and matrix templates, cells are organized into rows and columns.

The table and matrix data regions can display complex data relationships by including nested tables, matrices, lists, charts and gauges. Tables and matrices have a tabular layout and their data comes from a single dataset, built on a single data source. The key difference between tables and matrices is that tables can include only row groups, where as matrices have row groups and column groups.

Lists are a little different. They support a free-layout that and can include multiple peer tables or matrices, each using data from a different dataset. Lists can also be used for forms, such as invoices.

The following pictures show simple reports with a table, matrix, or list.



## To add a table or matrix to a report by using the New Table or New Matrix Wizard

1. On the Insert tab, click Table or Matrix, and then click Table Wizard or Matrix Wizard.
2. Follow the steps in the New Table or New Matrix wizard.
3. On the Home tab, click Run to see the rendered report.
4. On the Run tab, click Design to continue working on the report.

## To add a data region

1. On the Ribbon, in the Data Regions group, click the data region to add.
2. Click the design surface, and then drag to create a box that is the desired size of the data region.
3. Drag a report dataset field from the Report Data pane onto a data region cell. The data region is now bound to data from the report dataset.

## To select a data region

* For a tablix data region, right-click the corner handle. For a chart or gauge data region, click in the data region.

A selection handle and eight resizing handles appear.

For nested data regions, right-click in the nested data region, click Select, and then select the report item you want. To verify which report item is selected, use the Properties pane. The name of the selected item on the design surface appears in the toolbar of the Properties pane.

## To move a data region

* To move a data region, click the selection handle of the data region and drag it. Use snaplines to align it to existing report items.

If the ruler is not visible, click the View tab and select the Ruler option.

Alternatively, use the arrow keys to move the selected data region on the design surface.

## To delete a data region

* Select the data region, right-click in the data region, and then click Delete.

SSRS Graph and visualization:

When you want to summarize data in a visual format in a Reporting Services paginated report, use a Chart data region. It is important to choose an appropriate chart type for the type of data that you are presenting. This affects how well the data can be interpreted when put in chart form

## To add a chart to a report by using the Chart Wizard

1. On the Insert tab, click Chart, and then click Chart Wizard.
2. Follow the steps in the New Chart wizard.
3. On the Home tab, click Run to see the rendered report.
4. On the Run tab, click Design to continue working on the report.

## To add a chart to a report

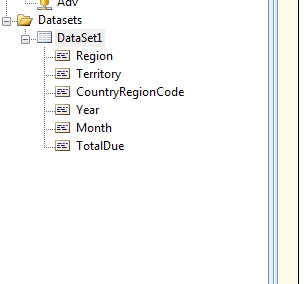
1. Create a report and define a dataset. For more information, see [Report Datasets (SSRS)](https://docs.microsoft.com/en-us/sql/reporting-services/report-data/report-datasets-ssrs?view=sql-server-2017).
2. On the Insert tab, click Chart, and then click Insert Chart.
3. Click on the design surface where you want the upper-left corner of the chart, and then drag to where you want the lower-right corner of the chart.

The Select Chart Type dialog box appears.

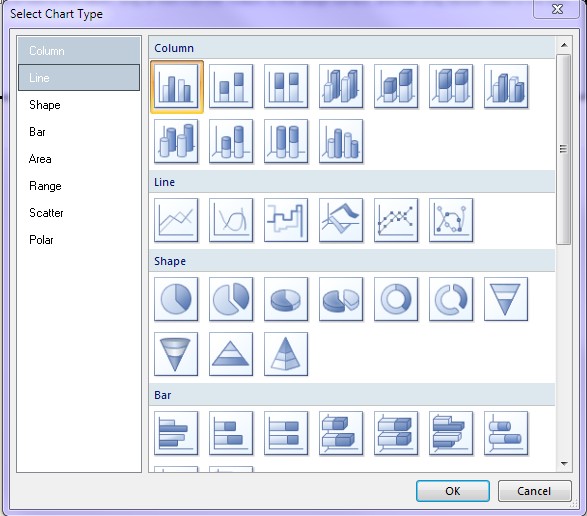
1. Select the type of chart you want to add. Click OK.
2. Click the chart to display the Chart Data pane.
3. Add one or more fields to the Values area. This information will be plotted on the value axis.
4. Add a grouping field to the Category Groups area. When you add this field to the Category Groups area, a grouping field is automatically created for you. Each group represents a data point in your series.
5. To summarize the data by category, right-click the data field and click Series Properties. In the Category box, select the category field from the drop-down list. Click OK.
6. On the Home tab, click Run to see the rendered report.
7. On the Run tab, click Design to continue working on the report.

On charts with axes, such as bar and column charts, the category axis may not display all the category labels.

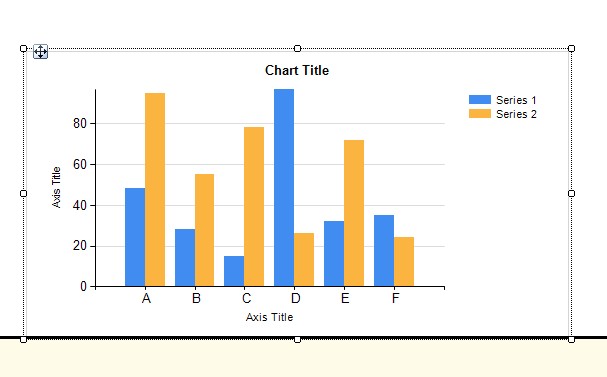
Lets look at some basic graphs and shapes.  
  
Step 1:  
As usual, lets create a blank report, connecting to adventure works database with dataset:  
  
  
SELECT  Sales.SalesTerritory.Name as Region,  
        Sales.SalesTerritory.CountryRegionCode,  
        Sales.SalesTerritory.[Group] as Territory,  
        Year(Sales.SalesOrderHeader.DueDate) as Year,  
        Month(Sales.SalesOrderHeader.DueDate) as Month,  
        Sales.SalesOrderHeader.TotalDue  
FROM Sales.SalesTerritory  
INNER JOIN Sales.SalesOrderHeader  
ON Sales.SalesTerritory.TerritoryID = Sales.SalesOrderHeader.TerritoryID

[](http://4.bp.blogspot.com/-APWZliPLSYw/UIB31jdSBcI/AAAAAAAAAT4/4QxyIDZuafg/s1600/1.Jpg)

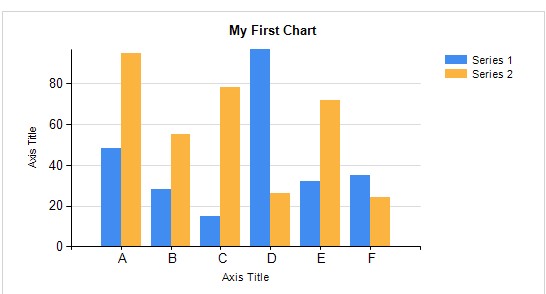
Step 2:  
Now lets drag and drop a chart control from the toolbox.  
As soon as you do that , you get a pop-up, displaying various image shapes for the chart control.

[](http://4.bp.blogspot.com/-o5iVJhZj7Hk/UIB11fDARQI/AAAAAAAAATg/cCIvN3KMkCw/s1600/1.Jpg)

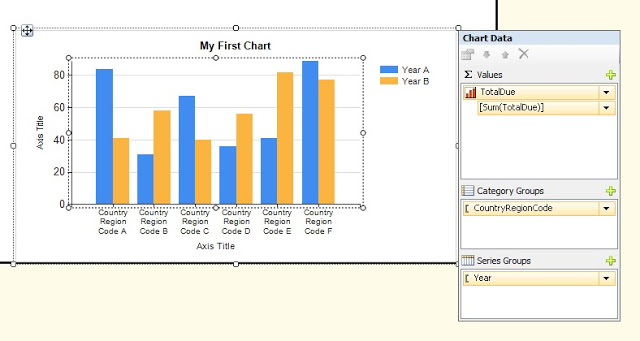
Step 3: These are the set of shapes provided by SSRS for visualization purposes.  
Let's choose the first shape.  
You should be getting something like this on your scree:

[](http://2.bp.blogspot.com/-LJntH9Ql3hc/UIB2eJ8oIOI/AAAAAAAAATo/M5fUyIqz6M0/s1600/1.Jpg)

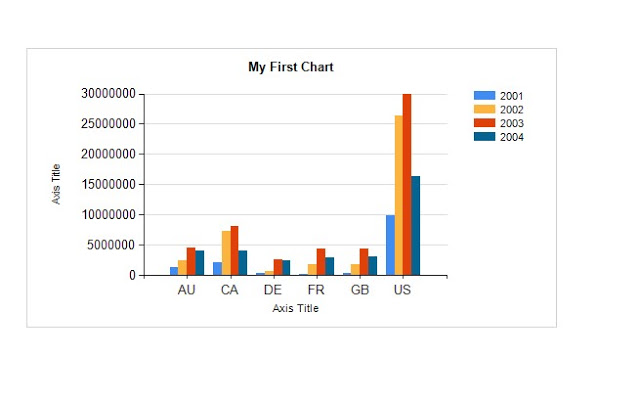
Step 4: This is a blank chart,with no data in it.  
You can click on the headings/text in the axis and change accordingly.  
Alternatively , you can also write SSRS expressions to change text/descriptions dynamically.

[](http://3.bp.blogspot.com/-6uxya0UfsVY/UIB285_T4rI/AAAAAAAAATw/ODc34boQ1es/s1600/1.Jpg)

Step 5: In the above example, changed the chart title to 'My First Chart'  
Now drag and drop TotalDue to the summation of values, CountryRegionCode to the CategoryGroups and year to the series Groups.  
Something like this...

[](http://1.bp.blogspot.com/-TVUR6t-MWgY/UIB4epZEOHI/AAAAAAAAAUA/rhZtRuEKc80/s1600/1.Jpg)

Step 6: Hit Preview, and you be getting something like this:

[](http://2.bp.blogspot.com/-QXBXcZUoSLc/UIB4q-iLdmI/AAAAAAAAAUI/J5pMMB6RYYI/s1600/1.Jpg)

Step 7: Try these steps, with a number of other shapes and graphs/Sparklines/Indicators.

# Add a Subreport and Parameters

Add subreports to a report when you want to create a main report that is a container for multiple related reports. A subreport is a reference to another report. To relate the reports through data values (for example, to have multiple reports show data for the same customer), you must design a parameterized report (for example, a report that shows the details for a specific customer) as the subreport. When you add a subreport to the main report, you can specify parameters to pass to the subreport.

You can also add subreports to dynamic rows or columns in a table or matrix. When the main report is processed, the subreport is processed for each row. In this case, consider whether you can achieve the desired effect by using data regions or nested data regions.

To add a subreport to a report, you must first create the report that will act as the subreport.

### To add a subreport

1. On the Insert tab, click Subreport.
2. On the design surface, click a location on the report and then drag a box to the desired size of the subreport. Alternatively, click the design surface to create a subreport of default size.
3. Right-click the subreport, and then click Subreport Properties.
4. In the Subreport Properties dialog box, type a name in the Name text box or accept the default. The name must be unique within the report. By default, a general name such as Subreport1 or Subreport2 is assigned.
5. In the Use this report as a subreport box, click Browse, or type the name of the report. Clicking Browse is preferred because the path to the subreport will be specified automatically. You can specify the report in the several ways. For more information, see [Specifying Paths to External Items (Report Builder and SSRS)](https://docs.microsoft.com/en-us/sql/reporting-services/report-design/specifying-paths-to-external-items-report-builder-and-ssrs?view=sql-server-2017).
6. (Optional) Click Yes for Omit border on page break to prevent a border from being rendered in the middle of the subreport if the subreport spans more than one page.
7. Click OK.

### To specify parameters to pass to a subreport

1. In Design view, right-click the subreport and then click Subreport Properties.
2. In the Subreport Properties dialog box, click Parameters.
3. Click Add. A new row is added to the parameter grid.
4. In the Name text box, type the name of a parameter in the subreport or choose it from the list box. This name must match a report parameter, not a query parameter, in the subreport.
5. In the Value list box, type or select a value to pass to the subreport. This value can be static text or an expression that references a field or other object in the main report.
6. Repeat steps 3-5 to specify a name and value for each subreport parameter.
7. To delete a subreport parameter, click the parameter in the parameter grid, and then click Delete.
8. To change the order of a subreport parameter, click the parameter, and then click the up button or the down button.

Changing the order of a subreport parameter does not affect the processing of the subreport.