Using Excel as a Data Source

Overview

XLReporter provides connectors to report on data from many different data sources including live data from PLCs and HMIs, historians from the leading SCADA vendors in industry and relational databases like Microsoft SQL Server, Access, Oracle and MySQL.

In addition, **XLReporter** provides a few different ways to report on data stored in other Excel workbooks including the ability to take data from one report generated by **XLReporter** and add it to another report.

This spotlight document shows the different ways **XLReporter** can report on data from other Excel workbooks.

Query an Excel Workbook

On a machine where Excel is installed, drivers are installed to query Excel workbooks like they are relational databases. There is a driver to query Excel 97-2003 workbook formats (.XLS files) and another that can be used to query Excel 2007 and above workbook formats (.XLSX, .XLSM and .XLSB files).

For Excel 2007 and above format, **XLReporter** can only query from these files if the 32 bit version of Excel is installed on the machine. If the 64 bit version of Excel is installed, only the .XLS file format is supported.

Workbook Format

To create a workbook to query from there are a few things that need to be configured. This depends on the file format of the workbook.

Excel 97-2003 Format (.XLS)

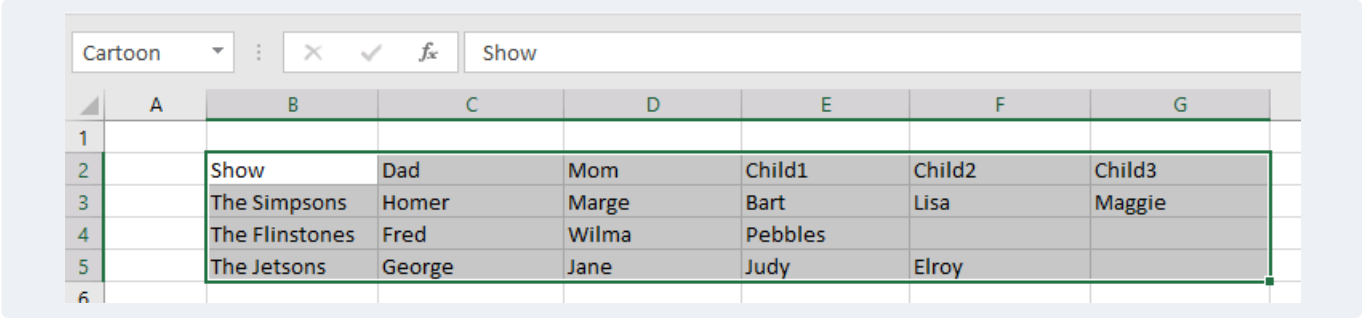
In .XLS files each worksheet can be considered as a single table. The first row can be used for the column names with the rows underneath for data. Excel is smart enough to detect how many rows of data are in the table.

If multiple tables are configured on the same worksheet, each table must be defined with a **Name** in Excel so each can be presented as separate tables to query from. For more information on setting up Excel Names, see the section below.

Excel 2007 and above Format (.XLSX, .XLSM and .XLSB)

In .XLSX, .XLSM and .XLSB files, any range of cells considered as a table must be defined with a **Name** is Excel so it can be presented as a table to query from.

In **Excel** a **Name** can be defined for a cell or range of cells so that it can be referenced in **Excel** objects like formulas and charts. For the purposes of this document, **Names** are used to identify tables to query from.

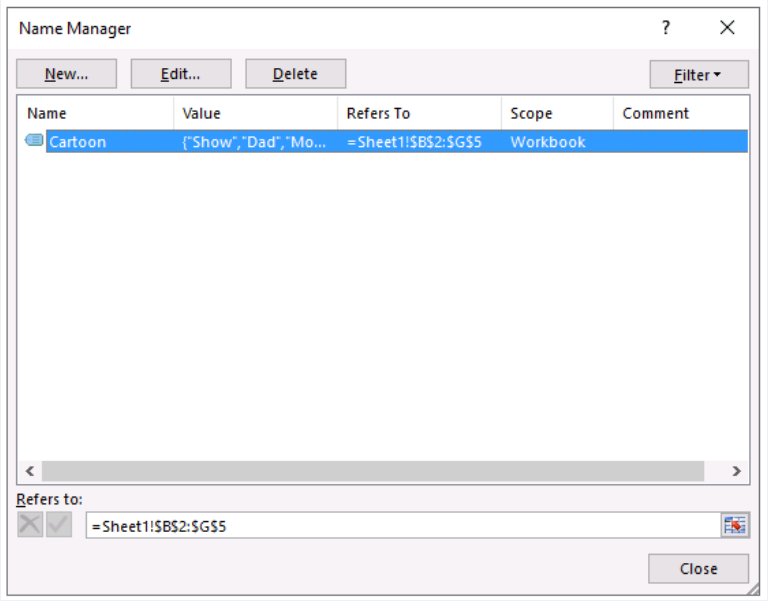


The simplest way to define a **Name** is to select the cell or range of cells on the worksheet and then enter the **Name** at the top left corner and then click the *Enter* key to apply.

This defines a **Name** that applies to the entire workbook. A **Name** can also be applied to a specific worksheet. This would allow you to use the same **Name** on multiple worksheets if needed.

For .XLS format, **Names** defined for the workbook or for a specific worksheet are presented as tables. However for .XLSX, .XLSM and .XLSB files, only Names defined for the workbook are presented as tables.

To view all the **Names** configured in the workbook, in **Excel**, under the **Formulas** tab, select **Name Manager**.



Here, existing **Names** can be edited or deleted and new names can be configured.

When using a **Name** as a table, the **Name** must be defined for the entire range of the table. This means that if new rows are added to the table, the **Name** must be updated to account for the new rows.

Database Data Connector

Excel 2007 and above Format (.XLSX, .XLSM and .XLSB)

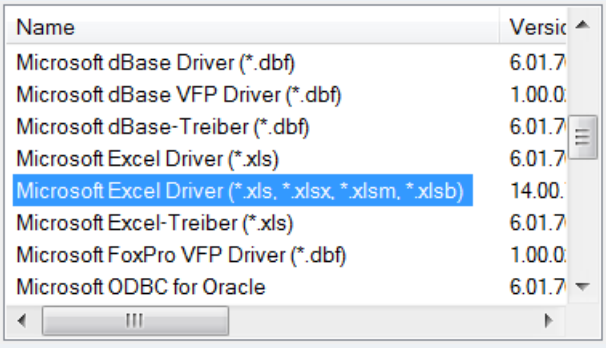
If the Excel workbook is the Excel 2007 and above format the Microsoft Access Database Engine needs to be installed. For Excel 2010 this is installed automatically but for Excel 2013 and above this must be installed separately.

Download the Microsoft Access Database Engine 2010 Redistributable for 32 bit (x86) from Microsoft's website and install.

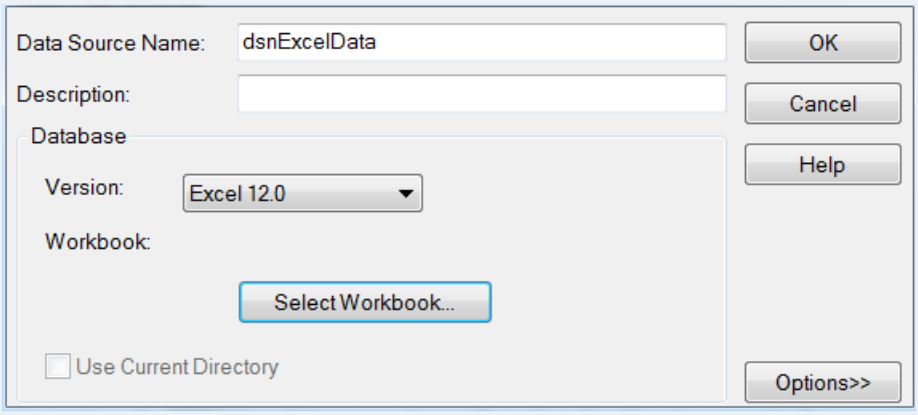
Once this is installed, a Data Source Name (DSN) must be configured to define the workbook to query from. The easiest way to define a DSN is from **XLReporter's Project Explorer** under the **Tools** tab by selecting **DSN Settings**.

It is recommended to create a **System DSN** so that any user on the system has access to it. However, this does require Administrator rights.

A new DSN is created by clicking the **Add** button.



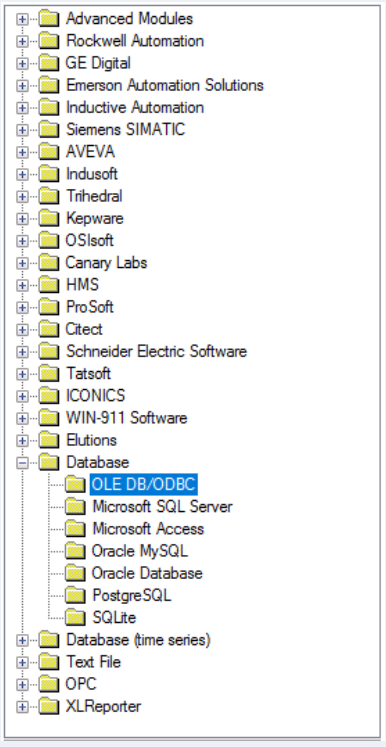
From the list of drivers select the *Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *.xlsb)*.



For the DSN the **Version** should be set to *Excel 12.0*. Click the **Select Workbook** button to pick the Excel workbook file containing the data you want to query.

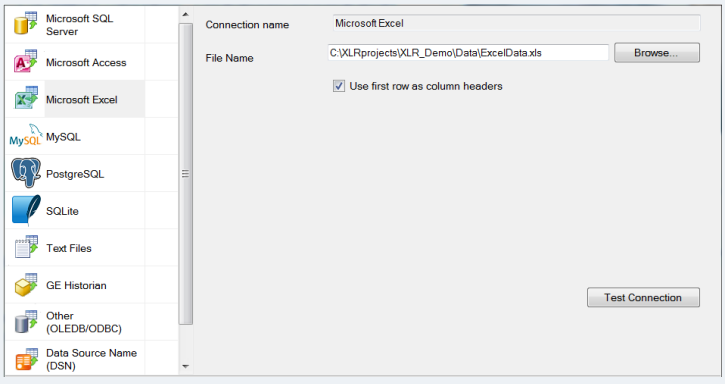
Connector Definition

A connector is defined in **XLReporter** in the **Project Explorer**, under the **Data** tab by selecting **Connectors**. A new connector is defined by clicking **Add**.



When creating a new connector for a database, expand the **Database** folder and select the **OLE DB/ODBC** option.

To define the connection to the Excel workbook, click the browse pushbutton [...] for **Primary Database**.



For .XLS workbooks select **Microsoft Excel** from the left and specify the **File Name** of the workbook. If the workbook has headings to use as column headers, make sure **Use first row as column headers** is checked. Otherwise leave this unchecked. When unchecked the columns are presented as *F1*, *F2*, *F3*, etc.

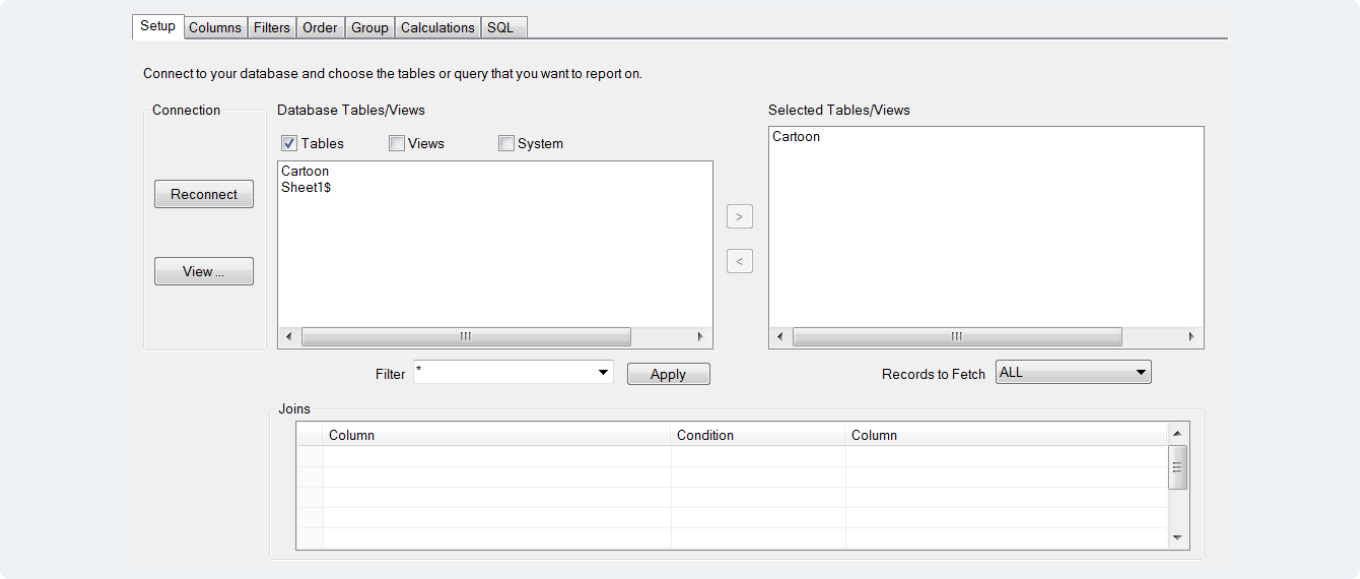
For .XLSX, .XLSM and .XLSB workbooks, select Data Source Name (DSN) and select the DSN configured for the Excel workbook.

Database Data Group

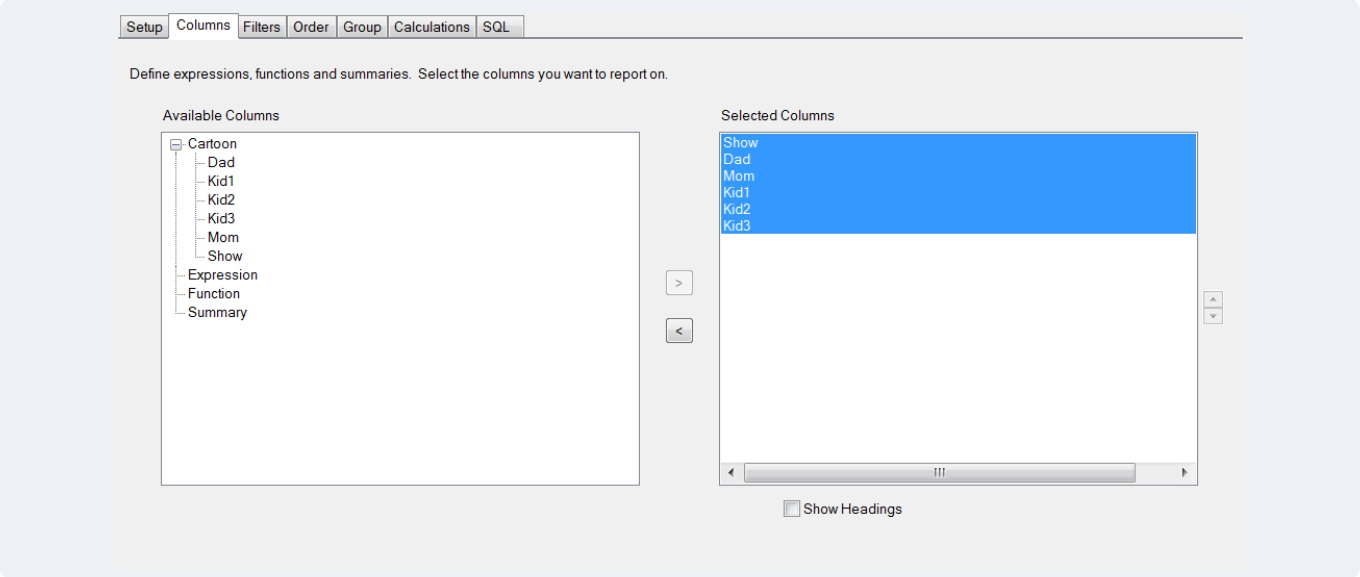
To query the data from an Excel workbook a database data group must be configured.

To configure, in a report template select **Data**, **Connect**.

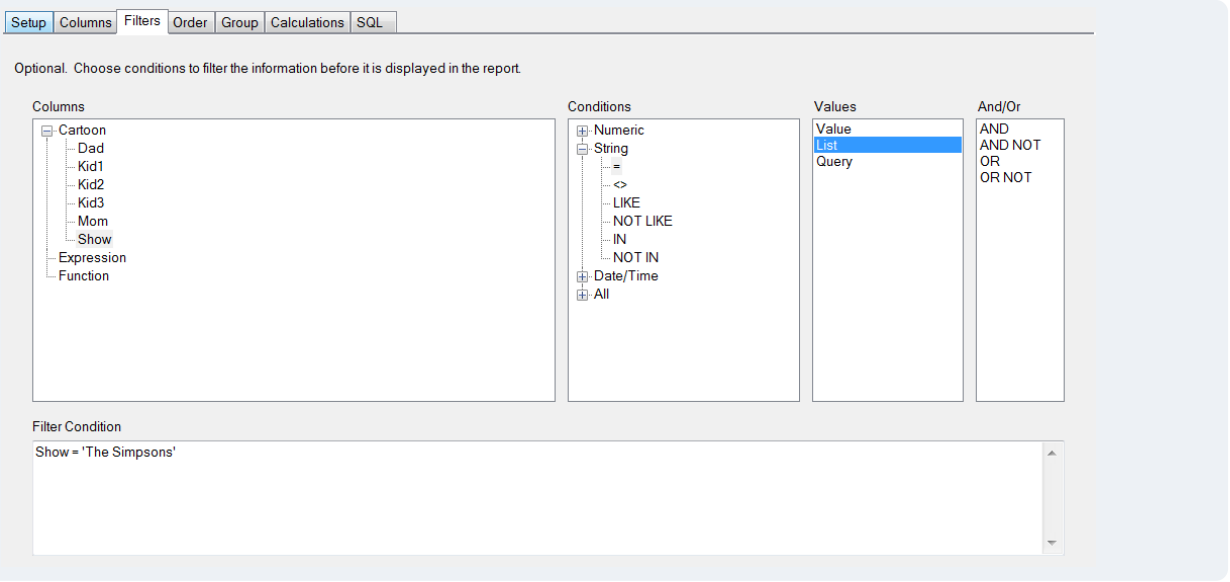
Under **Source**, set **Connector** to the **Database Connector** configured for the Excel workbook. Click the browse pushbutton [...] for **Name**.



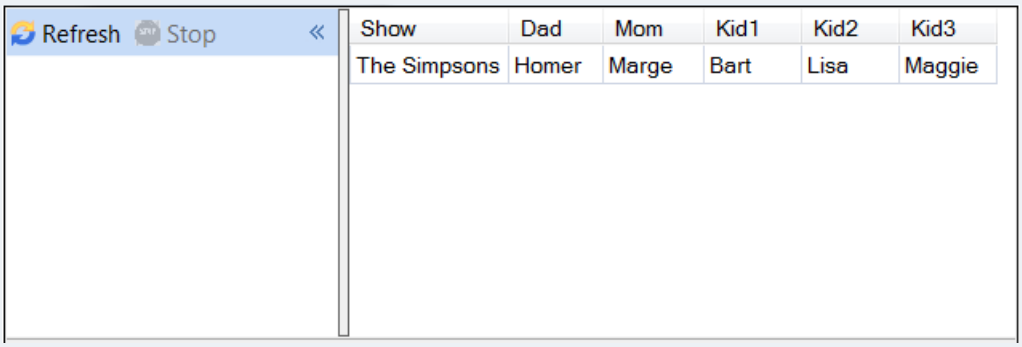
With .XLS workbooks a list of worksheets and defined **Names** are presented as **Tables**.



Under **Columns** all the headings for the worksheet or range are available to select from.



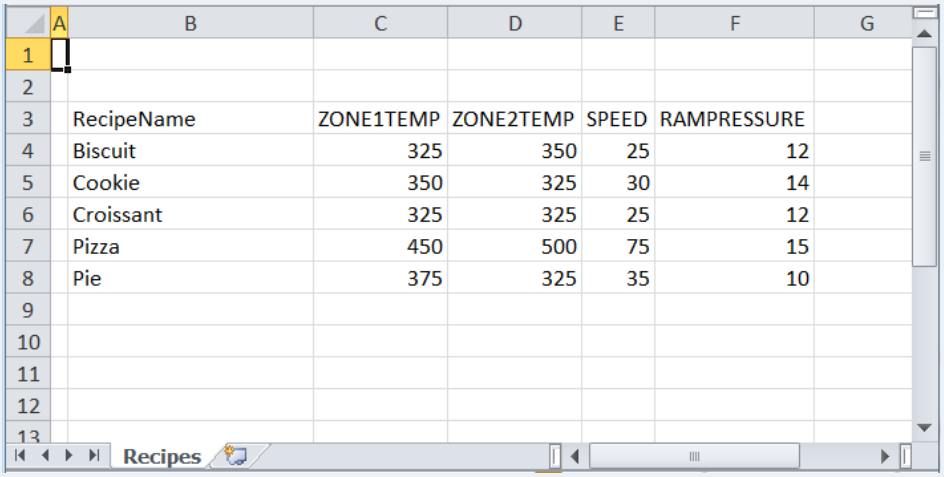
The **Filters** tab is available to restrict the data returned from the workbook.



Typical Scenario

Consider the following scenario: In an Excel workbook is a list of recipes with specific settings. I need **XLReporter** to present a list of available recipes to the operator for them to select from and then download the settings for the selected recipe to the process.

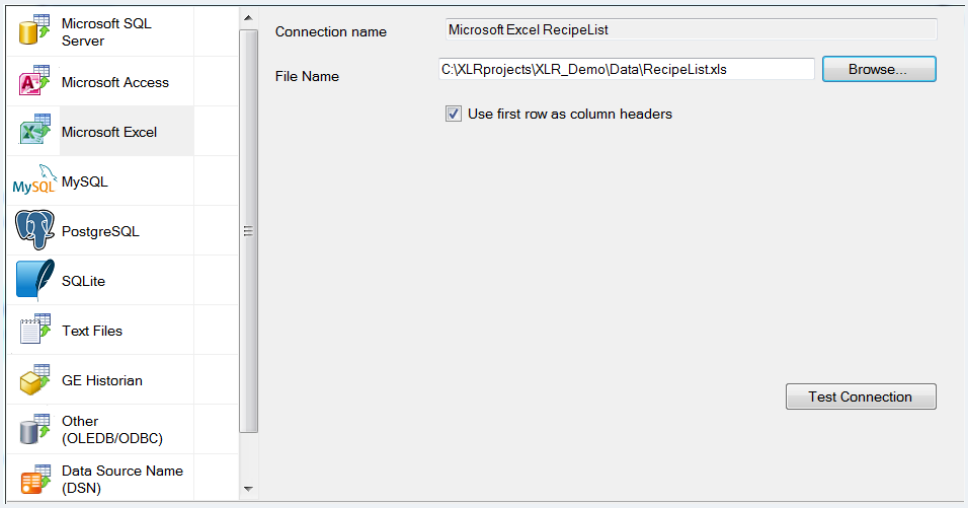
Recipe Workbook



The recipe workbook (RecipeList.xls) contains each recipe and the corresponding settings.

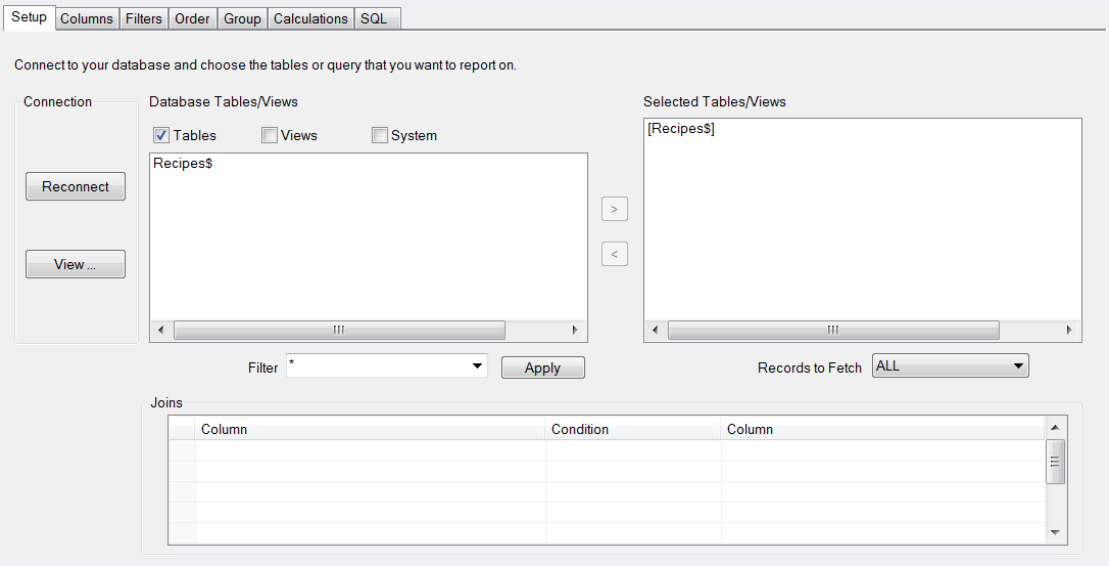
Connector Definition

The connector is defined in the project to the **RecipeList** workbook.

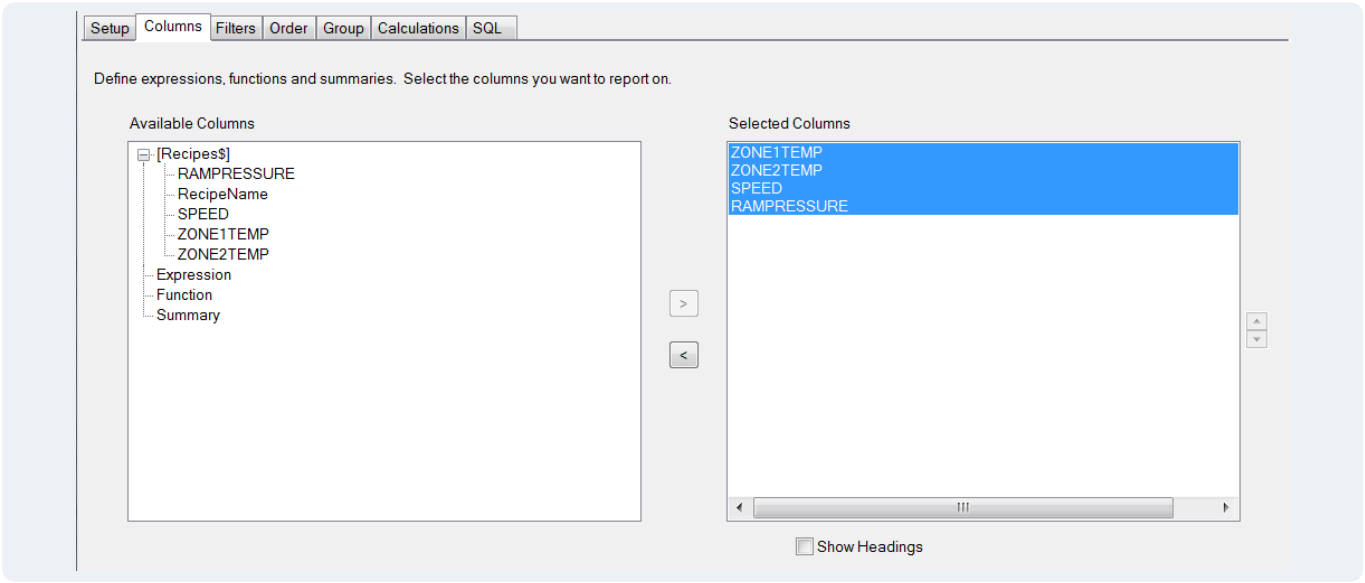


Template Design

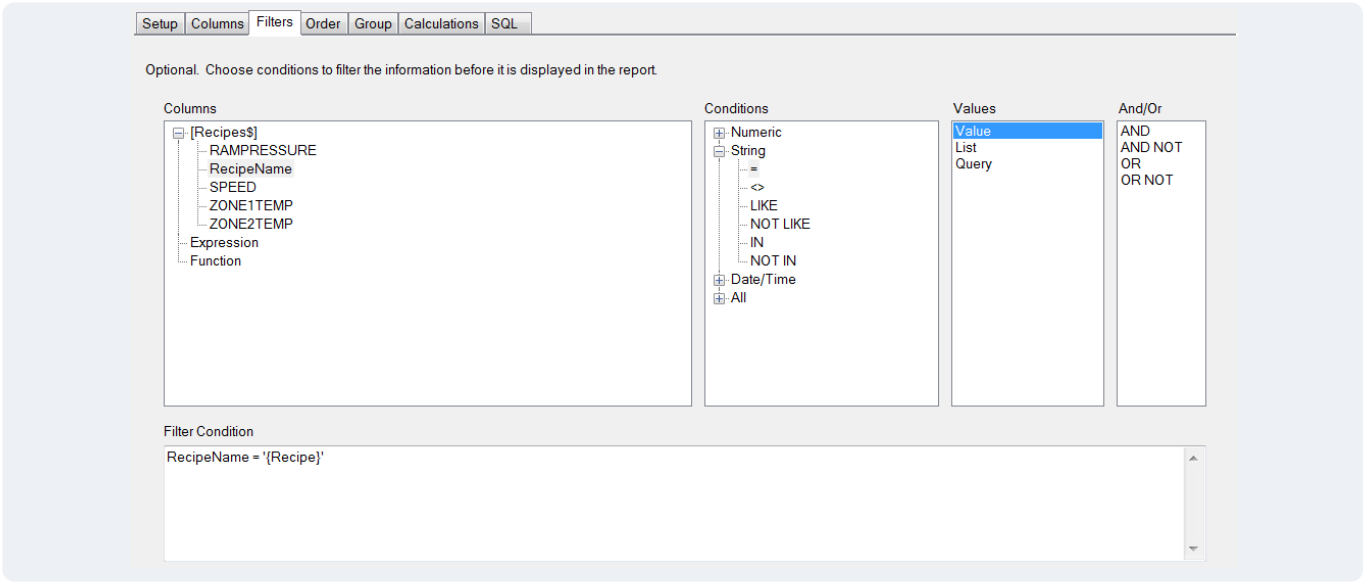
In the template for this, a **Data Connector** is configured to retrieve the recipe from the **RecipeList** workbook based on the user selection.



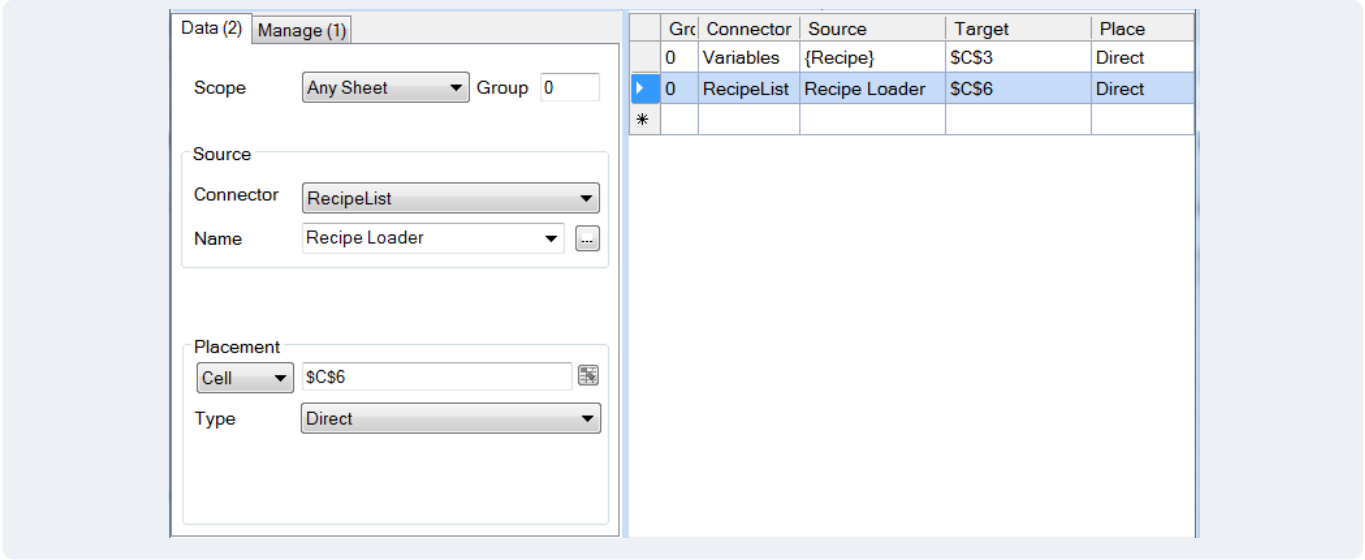
Under the **Setup** tab the *Recipe$* table is selected to get the list of recipes.



Under **Columns**, the four recipe values are selected.

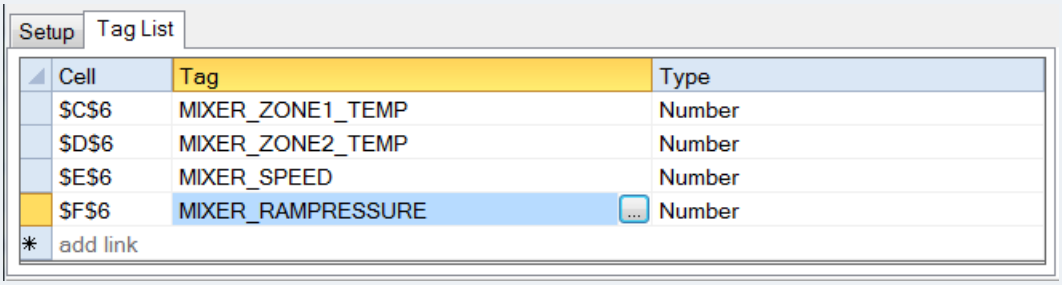


Under the **Filters** tab, the **Filter Condition** is set with a variable *{Recipe}* that will be specified by the user during runtime.



This **Data Group** is connected to the template along with another connection to show the recipe at the top of the report.

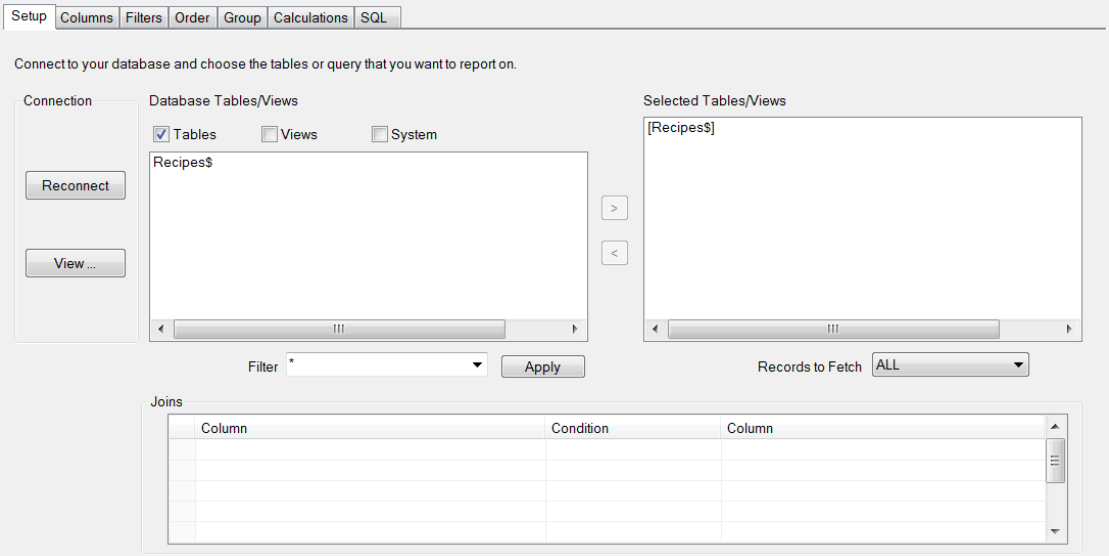
Under the **Manage** tab a **Data Export To Tag List** connection is configured to export the recipe values to the process.



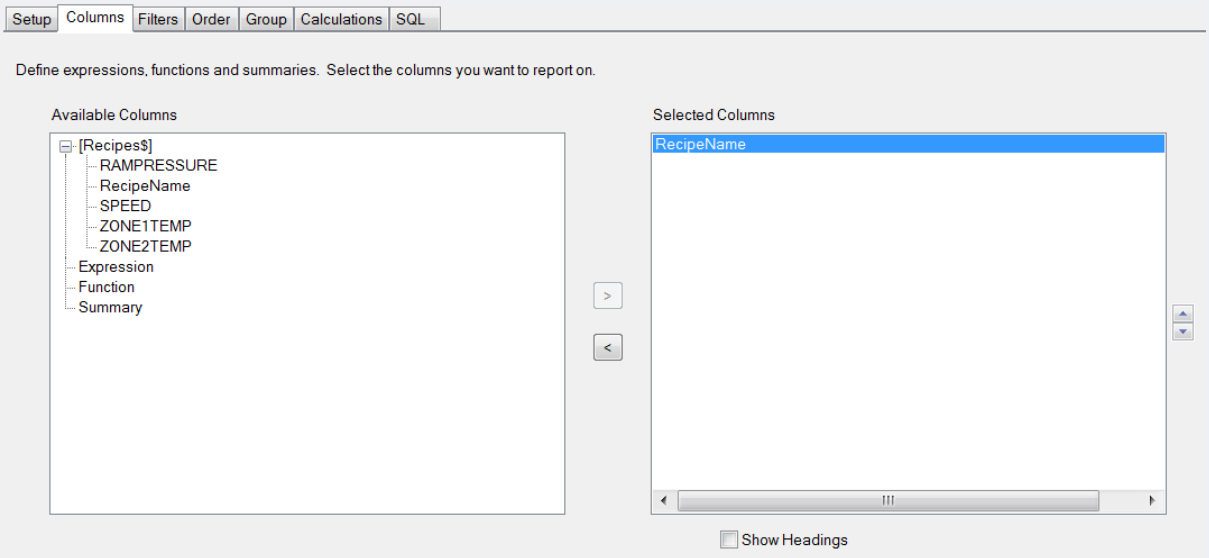
On-Demand Report

For the On-Demand report, the user must be presented with a list of available recipes to choose from. First, a **Database Data Group** is needed to present this list.

This is done most easily from the **Project Explorer** under the **Tools** tab by selecting **Connector Groups** and adding a new group for the connector (**RecipeList**).



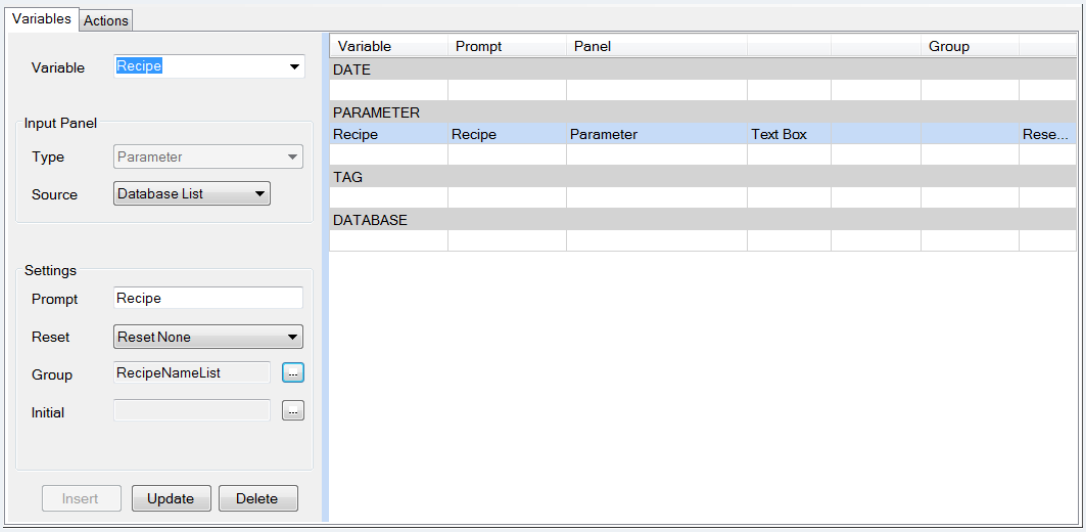
Under the **Setup** tab the *Recipe$* table is selected to get the list of recipes names.



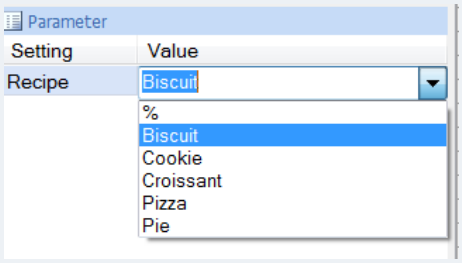
Under the **Columns** tab *RecipeName* is selected.

This group is saved as *RecipeNameList*.

Back in the template, the On-Demand reports application is set up to present this list. This is done in by selecting **On-Demand Designer**.



There are a few ways to present this list. In this case, for the **PARAMETER** panel is configured to prompt for the *Recipe* variable using the *RecipeNameList* **Group** as a **Database List**.

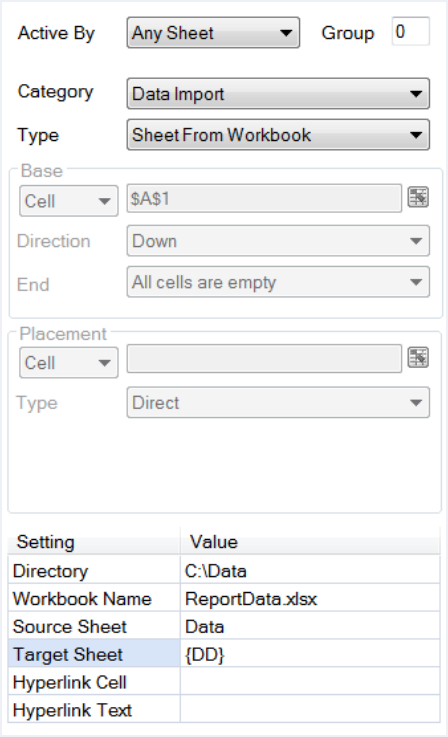


In **On-Demand Reports**, the user simply clicks to the right of *Recipe* and selects an available recipe from the drop down list.

Import a Worksheet

Rather than querying the data from another workbook, **XLReporter** provides a **Management** function, **Data Import Sheet From Workbook**, that can be used to import an entire worksheet from any workbook on the local system or across the network.

Data Management



The **Directory** can be used to define the folder on the file system where the workbook is saved. This can be a full path, a subfolder of the **Project Workbook** folder or can be left blank which implies the **Project Workbook** folder.

The **Workbook Name** setting defines the workbook where the worksheet is imported from. This can be a file within the **Directory** or can be full file name, in which case the **Directory** setting is ignored.

The **Workbook Name** can also contain a wildcards rather than a specific file. In this case, the **Directory** is analyzed for the most recent file based on the name specified and the modified date. This is the workbook used to import the worksheet from.

The **Source Sheet** defines the name of the sheet to import. The **Target Sheet** defines the name of the sheet in the report workbook. In the example above, the *Data* worksheet is imported and renamed to the current day of the month *{DD}* when imported.

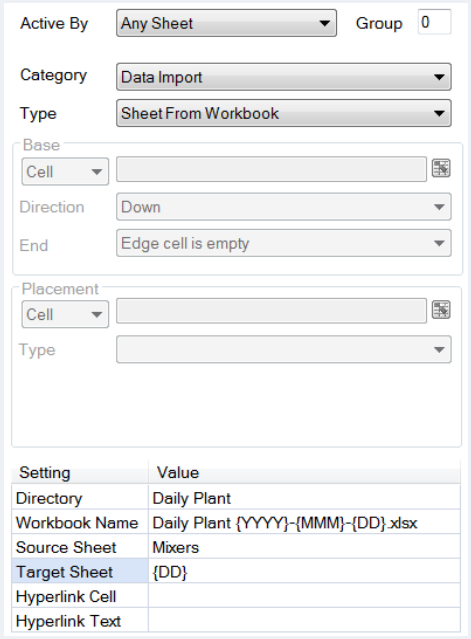
The **Hyperlink Cell** and **Hyperlink Text** settings can be used to add a hyperlink to a cell on the imported worksheet that can link back to the worksheet set in **Active By**. These settings are optional.

There is a second **Management** function that can be used to multiple worksheets at one time. This is the **Sheet From Workbook List** function. This works just like the **Sheet From Workbook** function except that one or more parameters can be specified as cells which can be iterated over.

Typical Scenario

Consider the following scenario: Every day a report is generated with three distinct report worksheets: *Mixers*, *Extruders* and *Ovens*. The template is *Daily Plant* and is stored in reports named *Daily Plant* with the current date, e.g. for January 1st, 2020 the file is *Daily Plant 2020-Jan-01*.

A second report is required that is a monthly workbook that contains the daily reports from the *Mixer*.



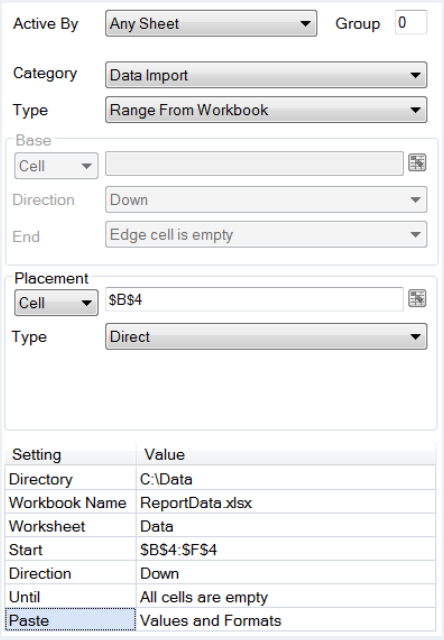
In the template for the monthly mixers report the **Data Management** connection for **Sheet From Workbook** is configured to import the *Mixers* worksheet from the *Daily Plant* report with the current date and name the imported worksheet after the day of the month.

This template can be scheduled to run at the same time as the *Daily Plant* report as long as it is listed under the action(s) that generate the *Daily Plant* report.

Import a Range

As an alternative to importing the entire worksheet from another workbook, a range can imported from the worksheet of another workbook using the **Data Management** function **Data Import Range From Workbook**.

Data Management



This function works very similar to the **Sheet From Workbook** function with two main exceptions:

- The **Placement** settings are enabled. This defines where the imported range will appear on the worksheet.
- Rather than a **Target Sheet** parameter, there is a set of **Start**, **Direction**, **Until** and **Paste** parameters.
  The **Start**, **Direction** and **Until** are used to define the range to copy. This works just like the **Apply To** or **Base** settings

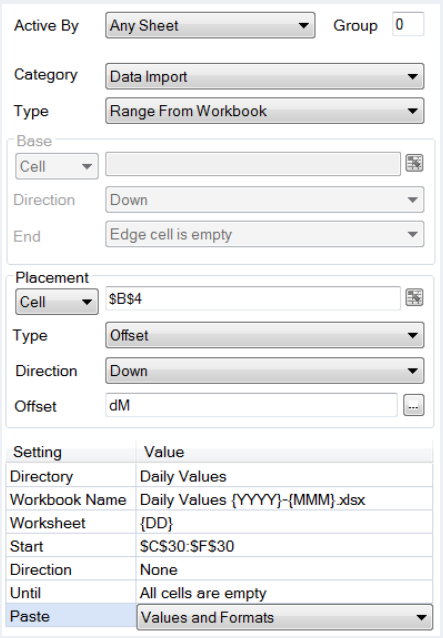
in other **Management** functions where the top row or column is defined as **Start**, the **Direction** defines the direction to search for the end of the range the **Until** defines how to determine where the range ends.
The **Paste** parameters determine what is pasted into the report from the other workbook. If the range to import contains formulas that reference other worksheets in the source workbook, consider pasting *Values and Formats* to prevent invalid formulas in the report.

There is a second **Management** function that can be used to multiple ranges at one time. This is the **Range From Workbook List** function. This works just like the **Range From Workbook** function except that one or more parameters can be specified as cells which can be iterated over.

Typical Scenario

Consider the following scenario: Every day a report is generated with hourly values logged from the process directly(no historian). This report also contains a row of summary data to calculate the daily average of these hourly values. The template is named *Daily Values* and the reports generated are *Daily Values* plus the current year. The report workbook contains daily worksheets with the hourly data and summary.

A second, summary report is also required to show the daily averages in monthly worksheet.



In the template for the monthly mixers report the **Data Management** connection for **Range From Workbook** is configured to import the summary range from the *Daily* worksheet in the monthly *Daily Values* report with the current year and month name.

This template can be scheduled to run at after the *Daily Values* report is complete which can be *11:00:00 PM* every day. This action should be listed under the action(s) that generate the *Daily Values* report.