sas THE POWER TO KNOW®

SAS® Visual Data Builder 6.2

User's Guide

SAS® Documentation

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2013. *SAS® Visual Data Builder 6.2: User's Guide*. Cary, NC: SAS Institute Inc.

SAS® Visual Data Builder 6.2: User's Guide

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

July 2013

Contents

What’s New in SAS Visual Data Builder 6.2

Enhancements to Working with Data

- You can use in-memory tables from SAS LASR Analytic Server as input tables for a query.
- You can join tables from more than one library.
- You can import a SAS Information Map into a query. Maps that are based on tables are supported. Maps based on OLAP cubes are not supported.
- For deployments that use Greenplum Data Computing Appliance, the generated SAS statements use the HPDS2 procedure to distribute data for staging tables. This enhancement is an efficient method for distributing data.

Enhancements to Importing Data

The import local data feature is enhanced as follows:

- You can preview a spreadsheet or delimited file before you import it.
- When you import a spreadsheet, you can also specify one or more worksheets to import. You can append the worksheets and import them as a single table or as individual tables.
- When you import a delimited file, you can specify the encoding to use for reading the file.
- You can import a SAS data set from your desktop.
- You can import files up to 2 GB in size. The application reads the data in blocks and transfers them to the server.

You can import remote data by browsing for SAS data sets that are available to your SAS Application Server and load them to SAS LASR Analytic Server.

Enhancements to Customizing Code

- You can add preprocessing and postprocessing SAS statements to your query.

- You can view all the code for the query, including the SQL procedure statements and LIBNAME statements.
- You can customize the code for the entire query when you unlock the code. In the previous release, you could modify the SELECT statement only.

Accessibility Features of SAS Visual Data Builder

Overview

SAS Visual Data Builder has been tested with assistive technology tools. It includes accessibility and compatibility features that improve the usability of the product for users with disabilities. (Some accessibility issues remain and are noted below.) These features are related to accessibility standards for electronic information technology that were adopted by the U.S. Government under Section 508 of the U.S. Rehabilitation Act of 1973 (2008 draft proposal initiative update). Applications are also tested against Web Content Accessibility Guidelines (WCAG) 2.0, part of the Web Accessibility Initiative (WAI) of the Worldwide Web Consortium (W3C). For detailed information about the accessibility of this product, send e-mail to **accessibility@sas.com** or call SAS Technical Support.

Documentation Format

Please contact **accessibility@sas.com** if you need this document in an alternative digital format.

Landmarks

Landmarks are references to the primary areas of an application’s user interface. They provide a quick and easy way for keyboard users to navigate to these areas of the application.

To access the list of landmarks that are available for a specific context, press Ctrl+F6 to open the Landmarks window. Use the arrow keys to select a landmark, and then press Enter to navigate to that area of the application.

User Interface Layout

SAS Visual Data Builder enables you to perform data preparation for analytics. The application window contains the following main sections:

- The top of the window contains the application name and an application bar that includes a menu bar and a **Log Off** button.

- The left side of the window contains a collapsible navigation pane. This pane contains a view of the SAS Folders tree. You can navigate the tree to select a table (one table at a time) for use in preparing data for analysis.
- The center of the window (the workspace) contains tabs that you use for designing queries. Each query is designed on a tab.
- The bottom of the window contains a series of tabs that you use for creating calculated columns and for filtering data.
- The right side of the window contains a **Properties** tab that you use for managing the properties of queries and tables. The information is dynamic and shows the properties of the item that is selected in the workspace.

Themes

An application's theme is the collection of colors, graphics, and fonts that appear in the application. The following themes are provided with this application: SAS Corporate, SAS Blue Steel, SAS Light, and SAS Dark. To change the theme for the application, select **File** ⇨ **Preferences** and go to the **Global Preferences** page.

You can also use keyboard shortcuts to magnify the contents of the browser window or to invert the application colors. For more information, see "Keyboard Shortcuts" on page viii.

Note: If you have special requirements for your themes, then contact your system administrator or visual designer about using the SAS Theme Designer for Flex application to build custom themes. SAS Theme Designer for Flex is installed with SAS themes. For more information about this tool, see *SAS Theme Designer for Flex: User's Guide*.

Keyboard Shortcuts

The following table contains the keyboard shortcuts for the application. In the user interface, the shortcuts are displayed within parentheses in tooltips and menu labels.

Note: Some application-level keyboard shortcuts do not work when you first open an application. When that happens, press Tab to place the focus on the application, and then try the keyboard shortcut again.

Note: When you use a keyboard shortcut to activate a button, first give the focus to the field or section that the button is associated with before you use the keyboard shortcut. For example, if a table has an associated ? button, you must first move the focus to the table before you press Ctrl+?.

Keyboard Shortcuts

Task	Keyboard Shortcut
Open a Help pop-up window from the ? button.	Ctrl+? *Note:* This shortcut does not work on some keyboards (for example, the Italian keyboard).
Zoom in.	Ctrl+plus sign
Zoom out.	Ctrl+minus sign
Reset the zoom state.	Ctrl+0
Maximize view (collapses the category pane and the tile pane, and hides the status bar and the application bar, which includes the menu bar and the workspace bar). or Exit maximized view (expands the category pane and the tile pane, and shows the status bar and the application bar).	Ctrl+Alt+Shift+M *Note:* This keyboard shortcut does not work when the focus is on the workspace bar.
Open a pop-up menu.	Shift+F9 (if a menu is available in that context) *Note:* If you use Shift+F9 to display the pop-up menu, then it is always displayed in the top left corner of the user interface control that you are using.
Open the Landmarks window.	Ctrl+F6
Temporarily invert or revert application colors (for the current session only). *Note:* You can set the **Invert application colors** preference in the Preferences window if you want the color change to persist across sessions.	Ctrl+~
Rename the selected tab.	Make sure that focus is on the tab. Press F2 and specify the new name. To commit your changes, press Enter. To cancel your changes, press Esc.
Close the selected tab.	Make sure that focus is on the tab, and then press Delete. *Note:* Some tabs cannot be closed.
Switch in and out of Edit mode for a table cell.	To enter Edit mode, select a cell and press F2. To exit Edit mode, press Esc.

Task	Keyboard Shortcut
Navigate between table headings and table content.	For a two-dimensional table, first make sure that focus is on the table and that you are not in Edit mode. Press Ctrl+F8 to switch focus between column headings and table cells. Use the arrow keys to navigate from heading to heading. For a multidimensional table, first make sure that focus is on a table cell and that you are not in Edit mode. Press Ctrl+F8 to switch focus between column headings, row headings, and table cells. Use the arrow keys to navigate from heading to heading.
Navigate the content rows of a table.	When table cells are in Edit mode: • Press Tab and Shift+Tab to move from cell to cell horizontally across columns. • Press Enter and Shift+Enter to move from cell to cell vertically across rows. When table cells are not in Edit mode, use the arrow keys to move from cell to cell.
Sort columns in a table.	To sort a single column, navigate to the column heading of the column that you want to sort. Press spacebar to sort the column. To sort additional columns, navigate to the column heading of each additional column that you want to sort. Press Ctrl+spacebar.

Exceptions to Accessibility Standards

Exceptions to accessibility standards are documented in the following table.

Note: The JAWS issues occur when JAWS is used with Internet Explorer. Other browsers were not tested with JAWS, unless noted.

Exceptions to Accessibility Standards

Accessibility Issue	Workaround
Sometimes, you cannot use the keyboard to sequentially navigate through the interface and move the focus in a meaningful order.	No workaround is available.
The SAS High Contrast theme has a few unresolved focus and contrast issues.	For contrast issues, select a different theme and then press Ctrl+~ to invert the colors.

Accessibility Issue	Workaround
The SAS Light theme and SAS Dark theme might not provide sufficient color contrast for some users.	Use the SAS Corporate theme or the SAS High Contrast theme.
JAWS cannot read some of the controls in the application, such as images, icons, and buttons.	No workaround is available.
JAWS cannot read the tooltips of items in trees, lists, and menus.	No workaround is available.
JAWS refers to table controls as list boxes.	When JAWS reports that a control is a list box, keep in mind that it might actually be a table.
JAWS can sometimes read controls that have been disabled.	No workaround is available.
Sometimes, JAWS does not correctly work with the controls in the Preferences window.	While you are in Virtual PC cursor mode, traverse the entire window to familiarize yourself with its contents before you change any of the settings. You might need to switch between Forms mode and Virtual PC cursor mode to access all of the controls.
JAWS does not correctly read the states in a tri-state check box tree if JAWS is not in Forms mode.	Disable the JAWS Virtual PC cursor when you work with the check box tree. Tab to the tree, and press Insert+Z to disable the Virtual PC cursor. When you finish interacting with the tree, press Insert+Z to re-enable the Virtual PC cursor.
The keyboard shortcuts that are used to interact with editable tables can conflict with keyboard shortcuts for the JAWS forms mode.	As a best practice, disable the JAWS Virtual PC cursor when you work with tables. Tab to the table, and press Insert+Z to disable the Virtual PC cursor. When you finish interacting with the table, press Insert+Z to re-enable the Virtual PC cursor.
JAWS cannot read two-column property tables.	No workaround is available.

Accessibility Issue	Workaround
JAWS does not correctly read the information in a table: • JAWS cannot read the column headings of a table. • When table cells are not editable and the focus is on the body of the table, JAWS reads an entire row at a time instead of cell by cell. • When table cells are editable, and the focus is on the body of the table, JAWS reads only the first row of the table. If you use the arrow keys to select a cell or row, then JAWS does not read anything. If you press Enter to edit a cell, then JAWS reads the row that contains the edited cell.	No workaround is available.
When you are in a table cell, if you press Home, End, Page Up, or Page Down, the selected cell will change to be one in the first column of the currently displayed columns for the table.	Use the arrow keys to navigate through the cells of the table.
You cannot use the keyboard to scroll to the left and the right in some tables.	No workaround is available.
You cannot use the keyboard to activate the links within how-to topics and Help pop-up windows.	Use the **Help** menu to access the linked documents.
You cannot use Shift+F10 to open a pop-up menu.	Use Shift+F9 to open pop-up menus that are created for the SAS application. The generic menu that is provided by the Flash player cannot be opened by Shift+F9. *Note:* If you press Shift+F10 in Internet Explorer and no context menu is available, the browser will move the focus to the **File** menu for the browser tab. To return focus to the application area of the browser window, press Esc.
You cannot use the keyboard to access the close (x) button that is in the top right corner of a tab.	Make sure that the focus is on the tab, and then press Delete to close the tab.
You cannot use the keyboard to access the close (x) button that is in the top right corner of a tile in the tile pane.	Make sure that the focus is on the tile, and then press Delete to close the tile. (The object that is displayed in the tile is not deleted.)
Visual focus for the menu bar is indicated with an outline around the entire menu bar instead of around individual menus.	To select individual menus, use the left or right arrow key.

Accessibility Issue	Workaround
Sometimes, you cannot use the Tab key to move the focus to the application area of a web browser (that is, the part of the browser window that is controlled by the Flash player).	The following workaround is applicable to Internet Explorer only. Press Ctrl+*number*, where *number* is the ordinal position of the application’s tab in the set of tabs that are open in your browser window. Then press Tab to move the focus into the application area.
You cannot use the Tab key to move the focus outside of a code or expression editor. Pressing Tab within the editor only inserts tabs.	For Internet Explorer, press Shift+F10 and then press Esc to move the focus outside of the editor. For Firefox, press Alt+Tab to switch to another application. When you switch back, the focus will be outside of the editor.
You cannot use Ctrl+Alt+Shift+M to minimize or maximize the view if the focus is on the workspace bar.	No workaround is available.
If you tab to an item that is partially or entirely off-screen, the item is not automatically scrolled back into view.	Sometimes, you can use the arrow keys or the Tab key to scroll the item back into view.
When you use the Ctrl+plus sign keyboard shortcut to zoom in, some portions of the interface can become hidden from view.	Use the keyboard to access the hidden parts of the interface.
The Ctrl+plus sign and Ctrl+minus sign keyboard shortcuts for zooming in and out do not work on some menus unless the menus are first opened.	Open the menu before you use the keyboard shortcut.
The Ctrl+plus sign and Ctrl+minus sign keyboard shortcuts for zooming in and out do not work on all elements in the application window (for example, tooltips and button labels).	No workaround is available.
If you maximize a tile in the **Home** workspace and then use the Tab key to navigate, the focus appears to be lost after you tab away from the **Log Off** button.	After you maximize the tile, press the Tab key 10 to 25 times, and eventually, the focus will return to the maximized tile.
You cannot use the keyboard to navigate in the **Layout** section because it is a Read-Only interface that is used for the visual verification of the elements that have been created.	Use the test button that is in the **Layout** section to preview your elements in a secondary window. The items that are displayed in the secondary window are identical to the items that are displayed in the **Layout** section, but unlike the items in the Read-Only **Layout** section, you can interact with the items in the secondary window. *Note:* After the application opens the secondary window, press Tab to place the focus in the window.

Accessibility Issue	Workaround
JAWS cannot read the labels for the **Red**, **Green**, and **Blue** fields in the Custom Colors window.	No workaround is available.
You cannot use the keyboard to access the color blocks in the **Recently used** section of the color selection control.	No workaround is available.
JAWS does not explain how to open a drop-down menu or drop-down list.	Press Ctrl+down arrow to open the control.
When JAWS reads the control names in a breadcrumb, it does not distinguish between the breadcrumb buttons that contain drop-down menus and those that do not.	Check for a drop-down menu by pressing Ctrl +down arrow on a breadcrumb button. A drop-down menu will open if one exists for that button.
When you use the down arrow to scroll through the items in a "combo box," any item that opens a secondary window will do so when you arrow down to it and will prevent you from navigating to items that are farther down in the drop-down list.	Press Ctrl+down arrow to scroll through the items in the drop-down list and then press Enter or Tab to make a selection.
When you add a date value to the predefined list for a date element, you cannot use the keyboard to access the date-selection button in the table cells in the Customize Data window for the predefined list.	Enter the date value in the field that is next to the date-selection button.
JAWS cannot read the contents of a tree table (that is, a table that contains a tree) unless the table is in Edit mode.	Make sure that the focus is in the tree table, and press F2 to enter Edit mode.
JAWS cannot read the content selection tree.	No workaround is available.
Sometimes, after you close a tab to hide it from view, you can still use the keyboard to access the contents of the tab.	No workaround is available.
When you use the keyboard to view the table properties from the **Properties** tab, the application loses focus and you can no longer navigate using the keyboard only.	No workaround is available.
When you use the keyboard to select a check box on the **Properties** tab, the application loses focus and you can no longer navigate using the keyboard only.	No workaround is available.
When the application prompts you for a user ID and password to access data, you cannot navigate to the window using the keyboard only.	No workaround is available.

Accessibility Issue	Workaround
When you schedule a query, and tab to the **Select one or more triggers for this query** radio button, the **New Time Event** does not become enabled. You cannot schedule a new query using only the keyboard.	No workaround is available.
When you schedule a query, you cannot tab to the radio buttons in the **Grouping conditions** area.	Press the Tab key to move forward to the **OK** button, and then press Shift+Tab to move backward to the **Grouping conditions** radio buttons.
When you schedule a query, and specify a new time event, you cannot use the Tab key to access all of the radio buttons.	No workaround is available.
JAWS cannot read the diagram for building queries.	No workaround is available.

Chapter 1

Overview of SAS Visual Data Builder

What Is SAS Visual Data Builder?

SAS Visual Data Builder enables analysts and data administrators to perform data preparation for analytics. You can design queries to perform joins, add calculated columns, and subset and sort data. Several productivity features speed the creation of columns based on common aggregation functions.

Once you design your queries, you can reuse them as subqueries for more sophisticated queries, export them as jobs for scheduling, or schedule them directly from the user interface.

The application has data import features that enable you to access data from spreadsheets, delimited files, and SAS data sets. Once you import the data, you can prepare it for analysis or join it with existing data.

The application provides a series of features that are used in deployments that include SAS LASR Analytic Server. You can load an existing table directly into memory, load the results of a query into memory, or append rows to an in-memory table on a server.

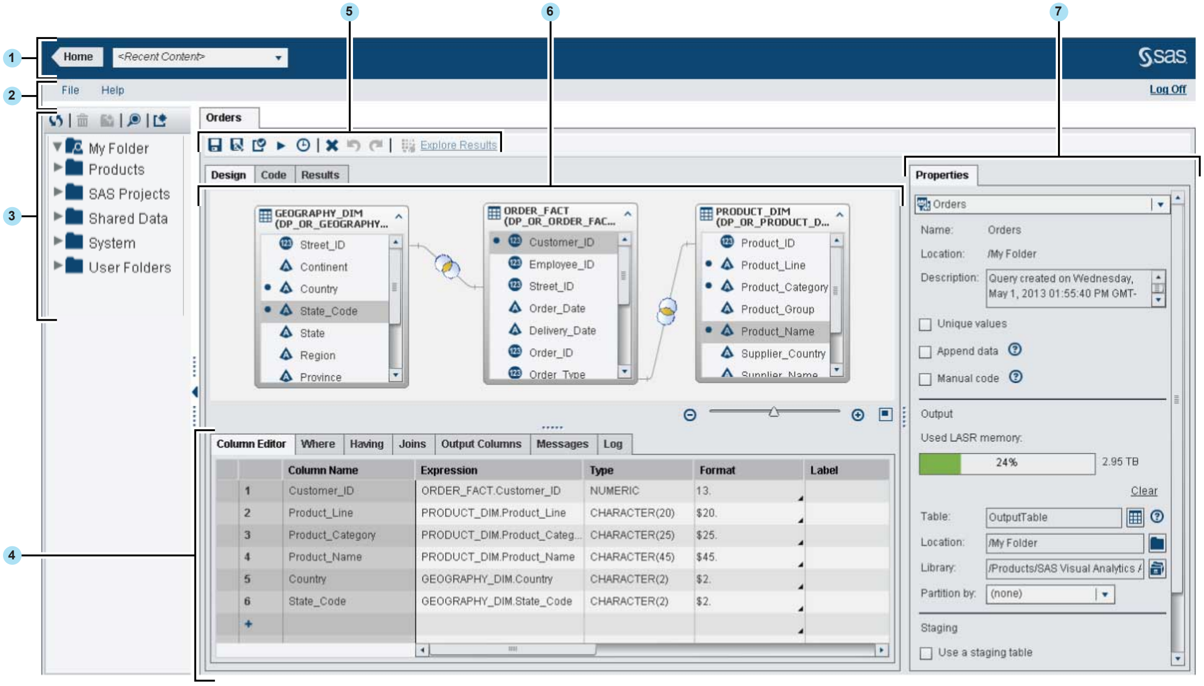
Accessing SAS Visual Data Builder

SAS Visual Data Builder is a web application. To log on, open your equivalent of the following URL:

```
http://hostname.example.com/SASVisualDataBuilder
```

Your First Look at SAS Visual Data Builder

The SAS Visual Data Builder interface contains the following elements:



	Column Name	Expression	Type	Format	Label
1	Customer_ID	ORDER_FACT.Customer_ID	NUMERIC	13.	
2	Product_Line	PRODUCT_DIM.Product_Line	CHARACTER(20)	$20.	
3	Product_Category	PRODUCT_DIM.Product_Categ...	CHARACTER(25)	$25.	
4	Product_Name	PRODUCT_DIM.Product_Name	CHARACTER(45)	$45.	
5	Country	GEOGRAPHY_DIM.Country	CHARACTER(2)	$2.	
6	State_Code	GEOGRAPHY_DIM.State_Code	CHARACTER(2)	$2.	
+					

Name: Orders
Location: /My Folder
Description: Query created on Wednesday, May 1, 2013 01:55:40 PM GMT-

□ Unique values
□ Append data
□ Manual code

Used LASR memory: 24% 2.95 TB

Table: OutputTable
Location: /My Folder
Library: /Products/SAS Visual Analytics /
Partition by: (none)

□ Use a staging table

1 The application bar enables you to return to the home page.

2 The menu bar contains menus that enable you to perform tasks.

3 The navigation pane displays a tree of tables and queries.

4 The lower pane contains tabs that enable you to specify column expressions and subset data.

5 The toolbar contains icons that enable you to develop and manage your queries.

6 The workspace enables you to design queries.

7 The properties panel enables you to manage the properties of the item that is selected in the workspace.

About Access to DBMS Data

SAS Visual Data Builder can be used to read source tables from third-party vendor databases and to write tables to them as well. The SAS/ACCESS Interface products provide this feature. The user ID and password that you use to log on to SAS Visual Data Builder might not be valid for a third-party vendor database. If this is the case, then you are prompted for credentials to the DBMS when you access the database with a Read or a Write operation. As an alternative to being prompted, you can store a login in metadata that has valid DBMS credentials. For more information, see "How to Store

Passwords for a Third-Party Server" in *SAS Intelligence Platform: Security Administration Guide*.

If you are prompted for credentials and supply an invalid user ID or password, you are denied access to the data. In this case, you can select **File ⇨ Clear Credentials Cache** to remove the invalid credentials from your session. The next time you access the data source, you are prompted again.

Third-party databases are often more restrictive about column names or length than SAS. For more information, see "DBMS-Specific Naming Conventions" in *SAS/ACCESS for Relational Databases: Reference*.

Working with User-Defined Formats

The preferred method for making user-defined formats available to a SAS Application Server is to name the format catalog formats.sas7bcat and to place it in ***`SAS-config-dir`*`/Lev1/SASApp/SASEnvironment/SASFormats`**. For more information about using user-defined formats, see *SAS Intelligence Platform: Data Administration Guide*.

SAS Visual Data Builder can use existing user-defined formats. The **Format** column displays the user-defined format name. However, if you change the format, then you cannot restore the original user-defined format.

Chapter 2
Specifying Preferences

Specifying Your Global Preferences

You can specify global preferences that are applied to all SAS web applications that are displayed with the Adobe Flash Player. These preferences are set by each user.

To specify global preferences:

1. Select **File** ⇨ **Preferences** to open the Preferences window.
2. Make sure that **Global Preferences** is selected in the left pane.
3. Select a **User locale** to specify your language and geographic region.

   Select a **Theme** to change the color scheme and other visual settings for all of your SAS web applications.

   Select **Invert application colors** to invert all of the colors in your SAS web applications.

   Select **Override settings for focus indicator** to change the color, thickness, and opacity of the focus in your SAS web applications.
4. Click **OK** to apply your changes.

Note: To restore all of the global preferences options, click **Reset to Defaults**.

Specifying Your Preferences for SAS Visual Data Builder

To specify preferences that are specific to SAS Visual Data Builder:

1. Select **File** ⇨ **Preferences** to open the Preferences dialog box.
2. Select **SAS Visual Data Builder** ⇨ **General**.

   Select a default SAS Application Server to use from **Default application server**. If you have added SAS Application Server instances to your deployment, make sure that the Job Execution Service has been configured to use the application server. For more information, see *SAS Intelligence Platform: Middle-Tier Administration Guide*.

3. Select **SAS Visual Data Builder** ⇨ **Scheduling**.

   Specify a default scheduling server, batch server, and deployment directory. For more information, see “Scheduling Preferences” on page 54.

4. Click **OK** to apply your changes.

Chapter 3

Designing Queries

What Is a Query?

A query is a metadata object that manages the references to input tables, output tables, staging tables, joins, and summarizations from a SAS Visual Data Builder session. You can save queries and open them later to edit the data preparation operations that are performed in the query. You can use saved queries as subqueries when you design a new query.

About Designing Queries

You can use SAS Visual Data Builder to design queries that perform analytic data preparation. You can use a query to subset, sort, join, and add calculated columns to tables.

As you design your query, you can click ☑ to validate your query. Use the ▶ button to preview or run the query.

The preview option uses a temporary table for the results. You can run the query after you specify an output table for the query and then save the query.

Creating a New Query

You can create a new query by clicking from the navigation pane.

Saving Your Query

To save your current query, click 💾, and enter a name and select a location.

When you save the query, you clear the undo and redo history.

Saving a Query as a New Query

Open an existing query from the SAS Folders tree that closely matches the query that you want to create. Click , and then enter a new name and select a location. Perform any customizations.

Note: If the existing query has output properties (table, location, library, and partition by), then the new query is saved with the same output properties as the existing query. If you do not want to use the existing output table, clear the output table information, and specify a new table.

See Also

"Output Tables" on page 14

Using the Design Tab

The **Design** tab is the default view for working with queries. This tab provides an easy-to-use interface for designing a query. You can perform the following tasks with the **Design** tab:

- Drag and drop tables or queries from the SAS Folders tree onto the workspace.
- Join tables by using your pointer to select the source column, and then drag the pointer to the corresponding column in the joined table.
- Add columns to the **Column Editor** tab by clicking the column name from the table in the workspace or by right-clicking on the table and selecting **Add All Columns**.
- Use the **Column Editor** tab to specify column expressions, aggregations, and sort. You can set the attributes for a column, such as the type, format, and label.

- View the **Output Columns** tab to see the number of output columns and attributes. When you specify aggregations and pivot by columns, the number of output columns can increase dramatically.
- Click ☑ to check that the query is valid.
- Check the **Messages** tab for information about warnings and errors such as invalid column expressions.
- Check the **Log** tab to view the SAS log. A SAS log is generated when you preview, run, or validate a query.
- Use the **Properties** tab in the right pane to specify output table information.

See Also

- "Working with Tables" on page 13
- "Working with Columns" on page 19
- "Working with Query Filters" on page 27
- "Working with Joins" on page 31

Adding a Data Source to a Query

Adding a Table

To add a table to a query, use the SAS Folders tree to locate the table (▦) and then drag and drop the table onto the workspace.

TIP You can also click 🔍 and search for the table by name and location.

Adding a Subquery

After you have designed a query and saved it, it can be used as an input data source to a new query. To add a subquery to a query, use the SAS Folders tree to locate the query (▦), and then drag and drop the query onto the workspace.

The subquery is represented in the workspace by the structure of the output table.

Best Practices for Adding Data Sources

When you plan to join data sources (tables or queries), the order in which you add the data sources to the workspace matters. The first data source that is added to the workspace is automatically assigned as the left table for any joins that you add to the query.

If you are designing a query that uses a fact table and dimension tables, drag and drop the fact table onto the workspace first. You can perform left, right, or full joins with the dimension tables faster because you need to specify the join type only. However, if you drag and drop a dimension table first, you can easily use the ⇄ button on the **Joins** tab to switch the left and right tables in the join.

If you are not using a fact table or dimension tables, then the sequence for adding tables to the workspace might not be very important. Just remember that the first table dropped onto the workspace is assigned as the left table, and you can switch the left table and right table on the **Joins** tab.SAS Visual Data Builder takes advantage of vendor-specific features in SQL processing whenever possible. If the source tables are from a third-party vendor database, the SAS/ACCESS Interface engine can optimize the performance of the query by passing the SQL statements through to the database.

Removing a Data Source from a Query

To remove a data source from a query, select the item in the workspace, right-click, and select **Remove Table**. This action applies to tables and subqueries.

Tips for Appending Data

The following list includes some strategies for appending data:

- Design a query to load the baseline data to the output table and run it.
- Modify the query, or use **Save As** to create an identical query and modify that one.
- Select the **Append data** check box.
- Add a filter on the **Where** or **Having** tab to select the changed data.
- Schedule the query.

Specifying Properties for a Query

On the **Properties** tab, you can specify the following properties:

Name
displays the name for the query.

Location
displays the metadata folder location for the query object.

Description
specifies a description of the query.

Unique values
specifies whether the SQL keyword DISTINCT is applied to the SELECT statement that is used to generate the result set for the query.

Append data
specifies whether the result set for the query is appended to the output table. If a staging table is used, then the staging table is replaced with the result set before appending to the output table.

Manual code
specifies that the query uses the SQL that is written by the user on the **Code** tab when the check box is selected. For more information, see “Customizing Code” on page 49.

Table
specifies the table name for either the staging output or the final output.

Location
specifies the metadata folder location to use for registering the staging table or output table metadata.

Library
specifies the library to use for the staging table or output table.

Partition by
specifies the column to use for partitioning the output table. This property applies when the output table is in a SAS LASR Analytic Server library or SAS Data in HDFS library only. For more information, see "Partitioning Tables" on page 37.

See Also

"Specifying Properties for a Source Table" on page 13

Chapter 4
Working with Tables

Source Tables

SAS Visual Data Builder supports using tables and queries as input data for a query. When you drop a table onto the workspace, SAS Visual Data Builder connects to the SAS Metadata Server to determine the column names and data types for the table. When you drop a subquery onto the workspace, the data builder makes the same request, but determines the column names and data types for the output table of the subquery.

Because SAS Visual Data Builder uses metadata for source tables, you must register the table metadata before you can access the data. Source tables can be registered in metadata with SAS Visual Analytics Administrator or SAS Management Console.

See Also

- "Adding a Data Source to a Query" on page 9
- *SAS Visual Analytics: Administration Guide*
- *SAS Intelligence Platform: Data Administration Guide*

Specifying Properties for a Source Table

When a table is selected in the workspace, or when you select it from the **Properties** menu, you can specify the following:

Alias
displays the SAS table name that is stored in metadata. You can specify a new value to use as a table alias. The alias name is stored with the query, and it does not affect the metadata information for the table.

Name
displays the metadata object name. You can change the metadata object name in the SAS Folders tree by using **Rename**.

Location
displays the table's metadata folder.

Library
displays the table's library.

Auto-aggregate
specifies whether to apply aggregations to the columns for this table when the columns are added to the query.

Functions
specifies the aggregations to apply to the columns for this table when the columns are added to the query.

See Also

"Using the Auto-Aggregate Functions" on page 24

Output Tables

About Output Tables

When you design a query, you specify an output table on the **Properties** tab. When you save the query, the output table is registered (or updated) in the metadata. When you run the query, the physical output table is created. The metadata that is created when you save the query enables you to use the table as a source table for another query or another SAS application.

Specifying the Output Table

Every query must have an output table in order to run it. How you specify the output table affects whether metadata is updated or created.

The following table shows the different ways to specify an output table.

Action	How To
Create a new output table *.	Click **Clear**, and specify a table name, location, and library. If the query was already saved with an output table, that table is not changed and is no longer associated with the query.

Action	How To
Reuse an existing table.	Select a metadata table ⊞. If the query was already saved with an output table, that table is not changed and is no longer associated with the query.
Change the name of the output table *.	Enter the new name in the **Table** field, and save the query. This changes both the name and physical name in metadata in order to make them the same. All queries, reports, explorations, and so on, still reference the same table object.

* If you replace the default table name, OutputTable, with another name before you save the query, the data builder registers a new output table and uses it with the query.

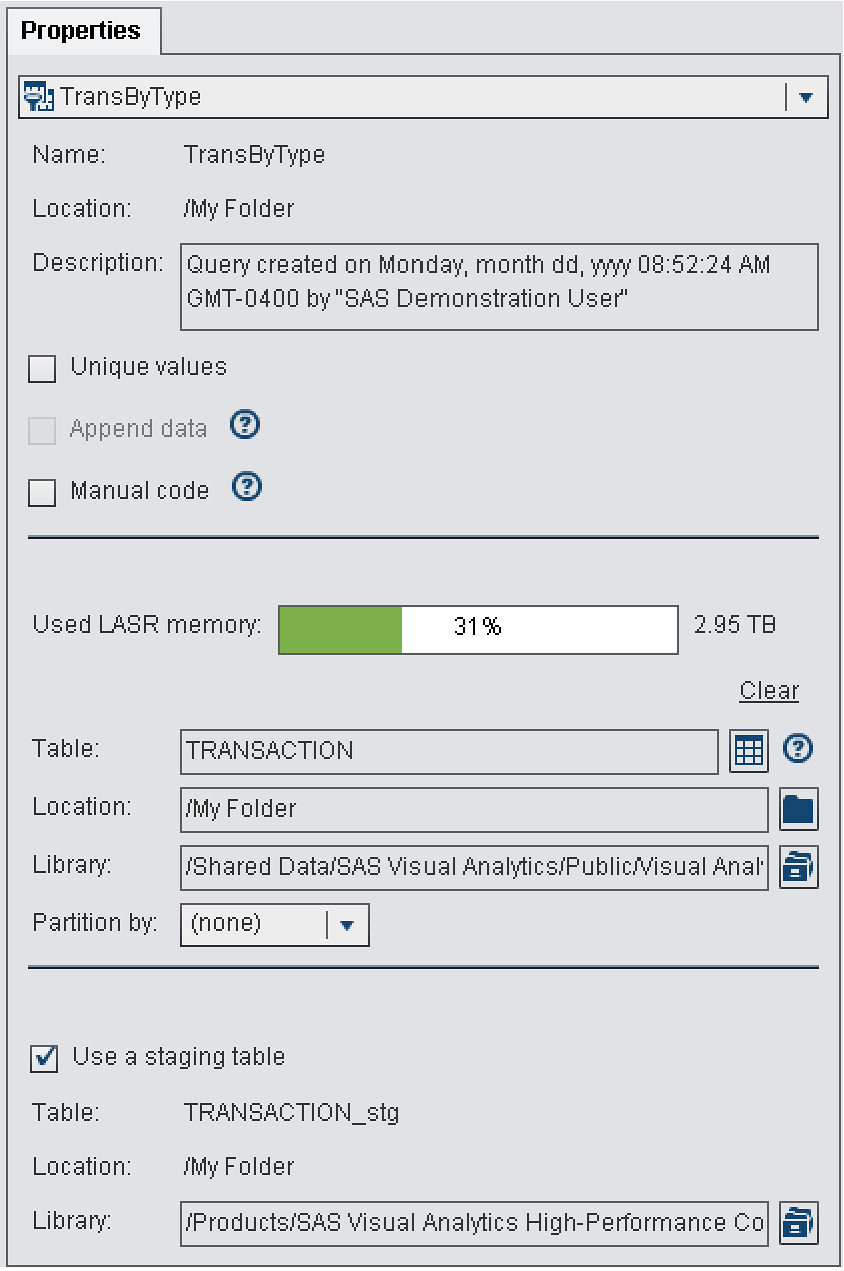
To specify an output table:

1. Select the top-level query object from the **Properties** tab.
2. Specify an output table name, location, and library. For SAS libraries, use 32 characters or less, and do not include spaces or special characters. Third-party vendor databases can have restrictions as well.

If you select a SAS LASR Analytic Server or SAS Data in HDFS library, the **Partition by** menu becomes available. After the settings for the output table are specified, the **Staging** section becomes available.

The following display shows an example of the Output and Staging properties.

Display 4.1 *Output and Staging Properties*



TransByType

Name: TransByType

Location: /My Folder

Description: Query created on Monday, month dd, yyyy 08:52:24 AM GMT-0400 by "SAS Demonstration User"

□ Unique values

□ Append data

□ Manual code

Used LASR memory: 31 % 2.95 TB

Table: TRANSACTION

Location: /My Folder

Library: /Shared Data/SAS Visual Analytics/Public/Visual Anal

Partition by: (none)

☑ Use a staging table

Table: TRANSACTION_stg

Location: /My Folder

Library: /Products/SAS Visual Analytics High-Performance Co

Staging Tables

About Staging Tables

SAS Visual Data Builder supports staging data. Staging data is a best practice because you can use SAS Visual Data Builder to access and transfer data from operational systems once, rather than frequently interfere with the operational systems and reduce their performance. Using SAS Visual Data Builder to stage data can provide the advantage of adding calculated columns when you stage the data. Like the output table, the staging table is registered in metadata when you save the query. The physical table for the staging table is created when you run the query. The physical table is always replaced with the results of the query.

You cannot specify the name for a staging table. The name of the output table is used, and an `__STG` (with two underscores) suffix is applied to the name. The suffix is used for the table name in metadata. The physical name of the staging table does not include the suffix.

Specifying a Staging Table

To use a staging table:

1. Select the top-level query object from the **Properties** tab.
2. The settings in the **Output** section must be set.
3. Expand the **Staging** section, and select the **Use a staging table** check box.
4. Specify a library.

Output and Staging Table Interactions

The following table identifies the supported combinations for output tables and staging tables. In addition, whether you can append data to tables is indicated.

Table 4.1 *Output Table and Staging Table Interactions*

Output Table	Staging Table	Append Data
SAS or DBMS *	None	Supported
SAS Data in HDFS	None	Not supported
Teradata or Greenplum **	None	Supported
SAS LASR Analytic Server	None	Supported
SAS or DBMS *	SAS or DBMS *	Supported
SAS LASR Analytic Server	SAS or DBMS *	Supported ***
SAS LASR Analytic Server	SAS Data in HDFS *Note:* SAS LASR Analytic Server is the only supported output.	Not supported
SAS LASR Analytic Server	Teradata or Greenplum **	Supported ***

Output Table	Staging Table	Append Data
Any	SAS LASR Analytic Server *Note:* SAS LASR Analytic Server libraries are not supported for staging tables.	Not supported

* The SAS or DBMS value represents data stored in SAS data sets or a third-party vendor database, respectively.

** The Teradata or Greenplum value represents deployments that use a third-party vendor appliance as a co-located data provider for SAS LASR Analytic Server.

*** Appending data is performed by the SAS LASR Analytic Server engine. Appends are not performed by having the server read data in parallel.

See Also

“Working with SAS LASR Analytic Server” on page 35

Chapter 5
Working with Columns

Adding Columns to a Query

Understanding How SAS Visual Data Builder Works with Columns

When you add a data source (a table or subquery) to the workspace, the columns from the data source are not automatically added as output columns to the query.

You need to add the columns to the query that you want to use as input. After the columns are added, you can specify column expressions and aggregations and use the sort and pivot by features.

Note: As an exception, the auto-aggregate feature requires that you set the default aggregations for the table before they are added to the query.

When a data source is dropped onto the workspace, the column types are represented by the following icons:

Table 5.1 *Icons for Data Types*

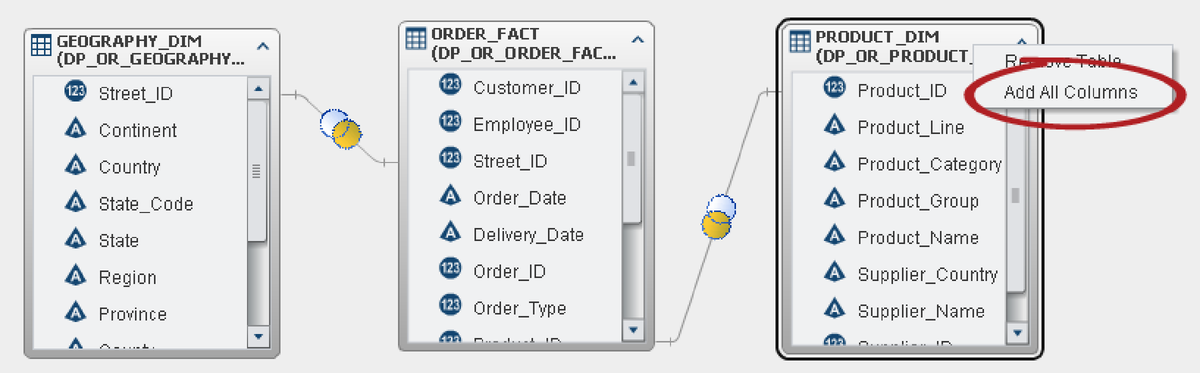
Icon	Description
123	This icon represents numeric data.

Icon	Description
A	This icon represents character data. *Note:* Date, time, and datetime data use this icon. After the column is added, the **Type** and **Format** columns are updated with information about the new column.

Adding All of the Columns from a Table

To add all of the columns from a table to a query, select the table in the workspace, right-click, and select **Add All Columns**.

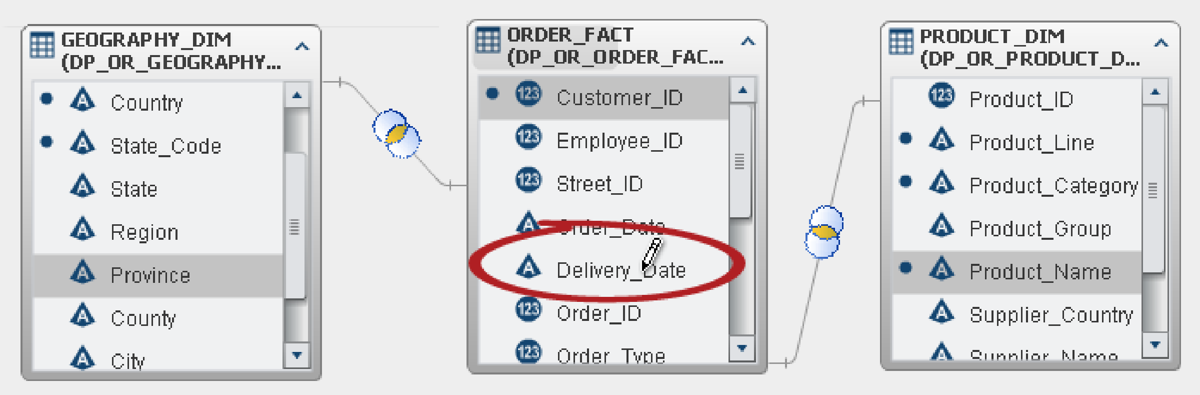
Display 5.1 *Add All Columns*



Adding a Single Column from a Table

To add one column from a table to a query, select the column name with your pointer.

Display 5.2 *Add One Column*



TIP If you select the column name an additional time, the column is added to the query again. This can be helpful if you want to use a column for both numeric and character data. When you add a column more than once, a number is added to the column name. If you change the column name, make sure that you do not have more than one column with the same name.

Adding a New Column Manually

To add a new column manually:

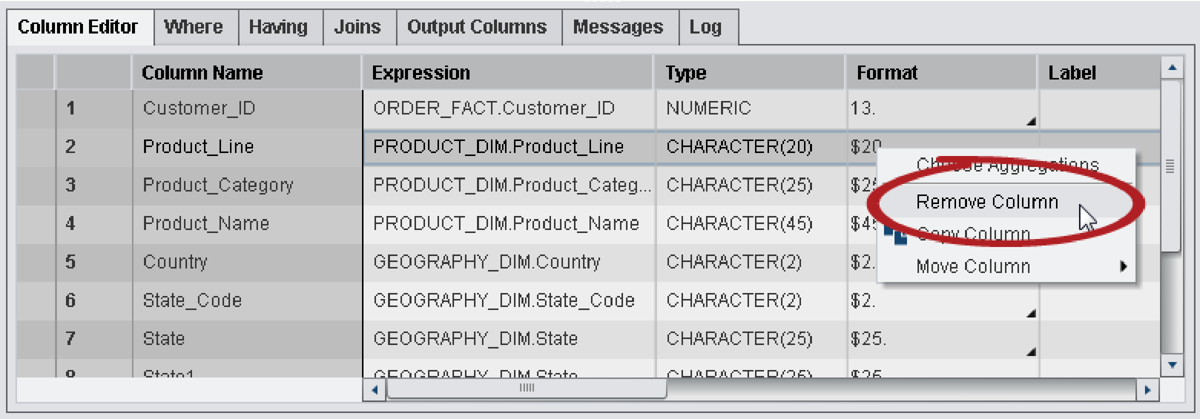
1. Click the **Column Editor** tab.
2. Click + at the bottom of the table.
3. Specify values for the **Column Name**, **Expression**, and **Type** fields. The remaining fields are optional.

Removing Columns

To remove a column from a query:

1. Click the **Column Editor** tab.
2. Select the column to remove, right-click, and select **Remove Column**.

Display 5.3 *Removing a Column*



	Column Name	Expression	Type	Format	Label
1	Customer_ID	ORDER_FACT.Customer_ID	NUMERIC	13.	
2	Product_Line	PRODUCT_DIM.Product_Line	CHARACTER(20)	$20	
3	Product_Category	PRODUCT_DIM.Product_Categ...	CHARACTER(25)	$2	
4	Product_Name	PRODUCT_DIM.Product_Name	CHARACTER(45)	$4	
5	Country	GEOGRAPHY_DIM.Country	CHARACTER(2)	$2.	
6	State_Code	GEOGRAPHY_DIM.State_Code	CHARACTER(2)	$2.	
7	State	GEOGRAPHY_DIM.State	CHARACTER(25)	$25.	
8	State1	GEOGRAPHY_DIM.State	CHARACTER(25)	$25	

TIP Hold down the Shift key or Ctrl key and click to select multiple columns.

Specifying a Column Expression

To specify a column expression:

1. On the **Column Editor** tab, select **`table-name.column-name`** from the **Expression** column. If you added a new column manually, then you need to make sure that you specified a table name and a column name.
2. Click to open the expression builder. The table name and column name are added automatically as the default SQL expression.
3. On the **Fields** tab, select columns from the source tables that you have added to the query.

   TIP You can begin entering the table name and column name in the **SQL expression** area to use the autocomplete feature.

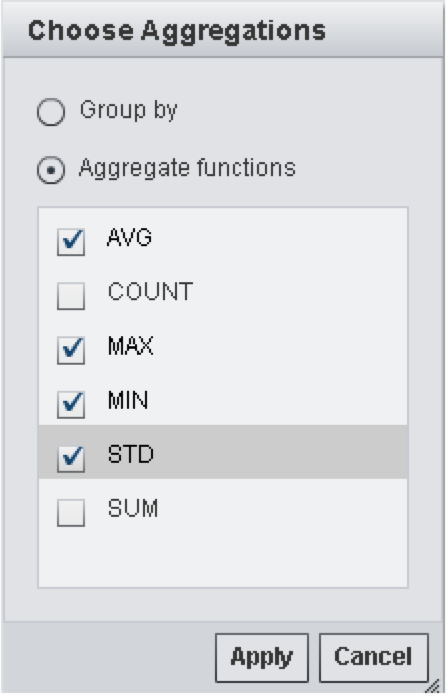
4. On the **Functions** tab, select the functions to apply to the source column.
5. Enter arithmetic operators and expressions such as CASE statements directly in the **SQL expression** area.
6. Click **Apply** to save the column expression.

Specifying Aggregations

To specify an aggregation for a column:

1. On the **Column Editor** tab, place your pointer in the **Aggregations** cell for the column and click. Click the ellipsis button to select the aggregations to use.
2. In the Choose Aggregations dialog box, select the **Aggregate functions** radio button.

Display 5.4 *Choose Aggregations Dialog Box*



Choose Aggregations

○ Group by

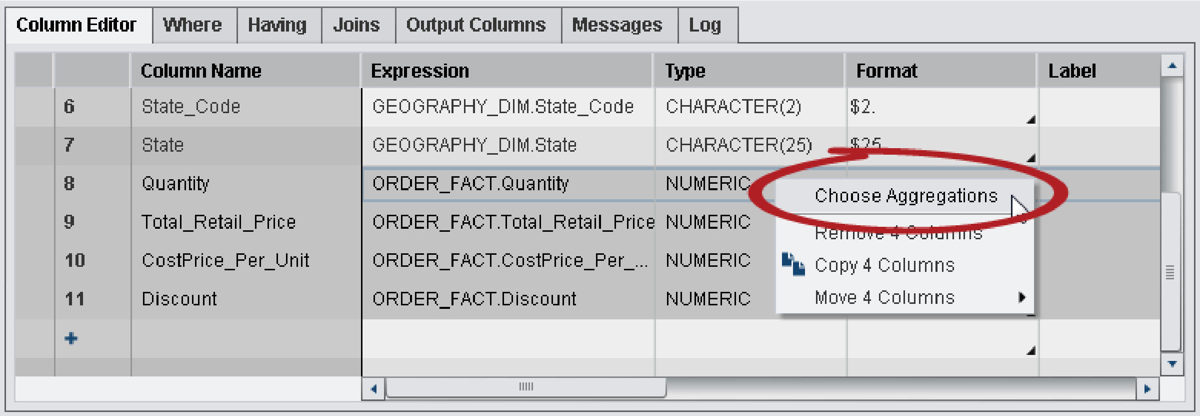
● Aggregate functions

☑ AVG
□ COUNT
☑ MAX
☑ MIN
☑ STD
□ SUM

3. Select the check boxes for the aggregate functions to use. Click **Apply**.

If you want to apply a common set of aggregate functions to a group of columns, hold down the Ctrl key, select the columns on the **Column Editor** tab, right-click, and select **Choose Aggregations**.

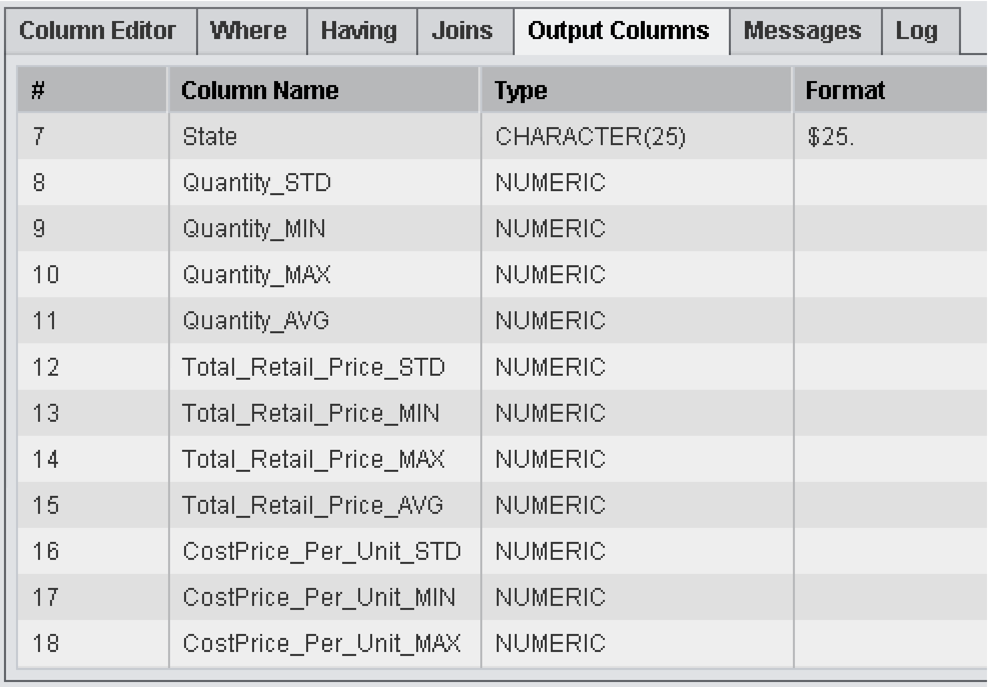
Display 5.5 *Choose Aggregations Menu Item*



	Column Name	Expression	Type	Format	Label
6	State_Code	GEOGRAPHY_DIM.State_Code	CHARACTER(2)	$2.	
7	State	GEOGRAPHY_DIM.State	CHARACTER(25)	$25.	
8	Quantity	ORDER_FACT.Quantity	NUMERIC		
9	Total_Retail_Price	ORDER_FACT.Total_Retail_Price	NUMERIC		
10	CostPrice_Per_Unit	ORDER_FACT.CostPrice_Per_...	NUMERIC		
11	Discount	ORDER_FACT.Discount	NUMERIC		

The following display shows how adding aggregations results in additional output columns. The aggregate function is appended to the column name.

Display 5.6 *Output Columns Tab with Aggregations*



#	Column Name	Type	Format
7	State	CHARACTER(25)	$25.
8	Quantity_STD	NUMERIC	
9	Quantity_MIN	NUMERIC	
10	Quantity_MAX	NUMERIC	
11	Quantity_AVG	NUMERIC	
12	Total_Retail_Price_STD	NUMERIC	
13	Total_Retail_Price_MIN	NUMERIC	
14	Total_Retail_Price_MAX	NUMERIC	
15	Total_Retail_Price_AVG	NUMERIC	
16	CostPrice_Per_Unit_STD	NUMERIC	
17	CostPrice_Per_Unit_MIN	NUMERIC	
18	CostPrice_Per_Unit_MAX	NUMERIC	

Removing All Aggregations

To remove all the aggregate functions and group by settings:

1. On the **Column Editor** tab, select all the columns.

2. Right-click, and select **Remove Aggregations**.

   TIP This menu option is available only when all of the columns are selected.

Using Group By Variables

When you add an aggregation to a column, the remaining columns are automatically used as group by variables. The **Aggregations** column displays `GROUP BY` for these variables.

You can use a column as a group by variable by following the steps in "Specifying Aggregations" and selecting the **Group by** radio button.

Using the Auto-Aggregate Functions

The auto-aggregate feature is a productivity feature that enables you to specify a set of aggregations to apply as default aggregations to numeric columns for a specific table. A typical use is to automatically aggregate some of the columns in a fact table.

To use the auto-aggregate feature:

1. Select a table on the **Design** tab. On the **Properties** tab, the table properties are displayed.
2. Select **Enable** for **Auto-Aggregate**.
3. Click the ellipsis button next to **Functions** to open the Choose Aggregations window.
4. Select the check boxes for the aggregate functions that you want to apply, and then click **Apply**.

Whenever you add a column to the query, the selected aggregate functions are automatically applied.

Using the Pivot By Feature

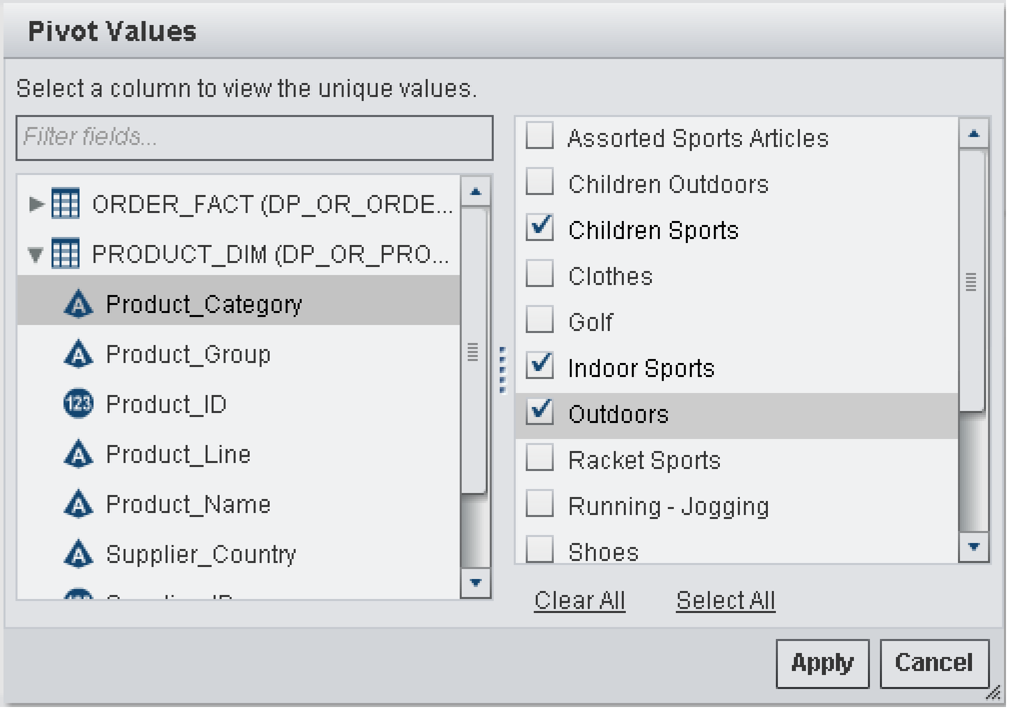
The pivot by feature provides an easy and powerful way to summarize data for analytics. You can specify a column to use as a categorical variable and the unique values to use. When the query is run, the output table is summarized with the aggregations that you apply.

To use the pivot by feature:

1. On the **Column Editor** tab, place your pointer in the **Pivot By** cell for the column to use as the pivot column. Click the ellipsis button to select the pivot column and values.
2. In the Pivot Values dialog box, select the pivot by column. You can enter search criteria in **Filter fields** to filter the column names.

The following display shows an example of pivoting by three values in the Product_Category column.

Display 5.7 *Pivot Values*



Pivot Values

Select a column to view the unique values.

Filter fields...

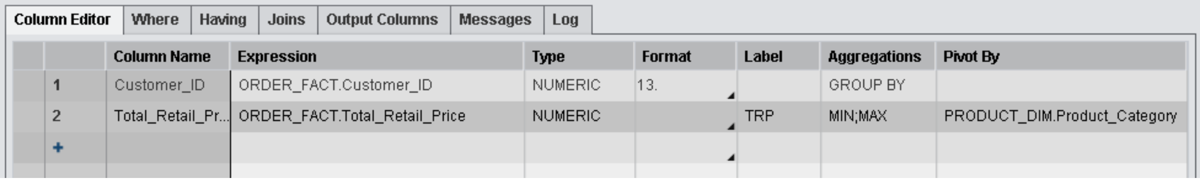
- ▶ ORDER_FACT (DP_OR_ORDE...
- ▼ PRODUCT_DIM (DP_OR_PRO...
  - Product_Category
  - Product_Group
  - Product_ID
  - Product_Line
  - Product_Name
  - Supplier_Country

□ Assorted Sports Articles
□ Children Outdoors
☑ Children Sports
□ Clothes
□ Golf
☑ Indoor Sports
☑ Outdoors
□ Racket Sports
□ Running - Jogging
□ Shoes

3. After the unique values for the column are loaded, select the check boxes for the values to use in the summarization. Click **Apply**.

The following display shows an example of the **Column Editor** tab when a pivot by column is used. The minimum and maximum Total_Retail_Price are calculated for each Customer_ID and are then pivoted by (transposed by) three values of the Product_Category column.

Display 5.8 *Column Editor Tab with a Pivot By Column*

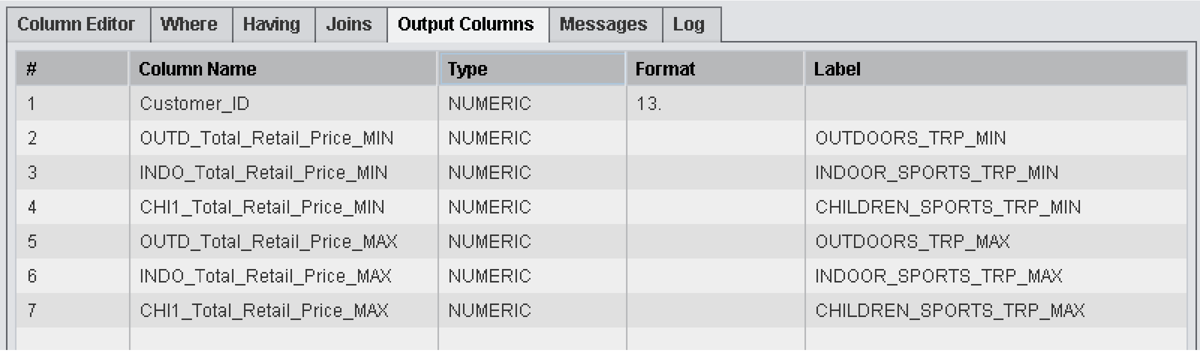


	Column Name	Expression	Type	Format	Label	Aggregations	Pivot By
1	Customer_ID	ORDER_FACT.Customer_ID	NUMERIC	13.		GROUP BY	
2	Total_Retail_Pr...	ORDER_FACT.Total_Retail_Price	NUMERIC		TRP	MIN;MAX	PRODUCT_DIM.Product_Category
+							

TIP TRP is specified as a label for the Total_Retail_Price column. Look at the next display to see how the label is used to create labels for the new columns.

The following display shows how pivoting the Customer_ID column by three values of the Product_Category column results in additional output columns. A substring of the pivot by values is used as a prefix to each column name and the aggregate function is used as a suffix. The pivot by column label and aggregate function are used in the output column label.

Display 5.9 Output Columns Tab with Pivot By Values



#	Column Name	Type	Format	Label
1	Customer_ID	NUMERIC	13.	
2	OUTD_Total_Retail_Price_MIN	NUMERIC		OUTDOORS_TRP_MIN
3	INDO_Total_Retail_Price_MIN	NUMERIC		INDOOR_SPORTS_TRP_MIN
4	CHI1_Total_Retail_Price_MIN	NUMERIC		CHILDREN_SPORTS_TRP_MIN
5	OUTD_Total_Retail_Price_MAX	NUMERIC		OUTDOORS_TRP_MAX
6	INDO_Total_Retail_Price_MAX	NUMERIC		INDOOR_SPORTS_TRP_MAX
7	CHI1_Total_Retail_Price_MAX	NUMERIC		CHILDREN_SPORTS_TRP_MAX

Chapter 6

Working with Query Filters

About Filtering Data

You can use the **Where** and **Having** tabs to filter data in SAS Visual Data Builder.

The SQL expression on the **Where** tab is applied to the input data. This SQL expression is often used to subset data on the columns in the source tables.

The SQL expression on the **Having** tab can be applied to either the input data or the calculated columns. This SQL expression is typically used to subset data on calculated columns in the output table.

Specifying a WHERE Clause

To add a WHERE clause to a query:

1. Click the **Where** tab.
2. On the **Fields** tab, expand the table node, and select the column to use for filtering data.

   TIP You can type in **Filter fields** to locate the column.
3. Double-click the column or drag and drop it in the **SQL expression** area.
4. (Optional) You can click the **Functions** tab and select the functions to use with filtering.
5. For character variables, you can click **Unique Values** to load the unique values of the column. Select the check boxes to specify the unique values to include in the filter.

   Click **Apply** to add the unique values to the **SQL expression** area.
6. Edit the WHERE clause in the **SQL expression** area as follows:

Number of Values	Edit	Example
A single character value	Add an equal sign between the column name and the unique value.	`CARS.Make = 'Acura'`
More than one character value	Specify an IN operator and enclose the unique values in parentheses.	`CARS.Make IN ( 'Acura', 'Audi')`

Numeric comparison	Specify a numeric operator and a constant, or specify a numeric operator and another column name.	`CARS.Cylinders >= 6`
		`PRDSALE.Actual > PRDSALE.Estimate`

7. Click 💾 to save the filter with the query.

Specifying a HAVING Clause

To add a HAVING clause to a query:

1. Click the **Having** tab.
2. On the **Fields** tab, expand the table node, and select the column to use for filtering data. The **Output Columns** node includes the calculated columns.

   TIP You can type in **Filter fields** to locate the column.
3. Double-click the column or drag and drop it in the **SQL expression** area.
4. (Optional) You can click the **Functions** tab and select the functions to use with filtering.
5. For character variables, you can click **Unique Values** to load the unique values of the column. Select the check boxes to specify the unique values to include in the filter.

   Click **Apply** to add the unique values to the **SQL expression** area.
6. Edit the HAVING clause in the **SQL expression** area as follows:

Number of Values	Edit	Example
A single character value	Add an equal sign between the column name and the unique value.	`CARS.Make = 'Acura'`
More than one character value	Specify an IN operator and enclose the unique values in parentheses.	`CARS.Make IN ( 'Acura', 'Audi')`

Number of Values	Edit	Example

Numeric comparison	Specify a numeric operator and a constant, or specify a numeric operator and another column name.	`AVG(CARS.Horsepower) >= 165`
		`PRDSALE.Actual > PRDSALE.Estimate`

7. Click 💾 to save the filter with the query.

Best Practices for Filters

SAS Visual Data Builder provides the **Where** and **Having** tabs for filtering data. Follow these best practices:

- Filter on the **Where** tab first because the WHERE clause reduces the rows to consider for further subsetting.
- When you specify a filter on a column that is an index or primary key, avoid using a function whenever it is possible. Using a function, such as **`CAST(order_id as DOUBLE)`**, risks performing a full-table scan rather than using the index.
- Filter on the table that has the smallest number of rows first.
- If you need to remove a table that is used on the **Where** or **Having** tabs, then remove the reference to the table from the filters before you remove the table from the query. Otherwise, if you remove a table that is referenced in a filter, then all of the filtering conditions are cleared.

Chapter 7

Working with Joins

About Joins

SAS Visual Data Builder supports joins for tables and subqueries. You can join tables to each other, including self joins. You can join subqueries to tables and join subqueries to subqueries. When you use a subquery in a join, the join condition is made against the output table for the subquery. The data builder supports joining up to 256 tables.

When you drop a table or subquery onto the workspace, the data builder attempts to determine a join condition automatically. When the data builder creates a join automatically using foreign keys or by matching columns, the join is added as an inner join. The data builder also supports left, right, and full joins. You need to specify the join type manually if you do not want an inner join.

In addition to easily adding joins to a query, SAS Visual Data Builder generates an SQL statement with all of the joins declared explicitly. For example, you can specify an inner join in a WHERE clause, such as **`WHERE t1.order_id=t2.order_id`**. However, mixing inner joins in a WHERE clause and outer join types in a single query can be complex to read and understand.

SAS Visual Data Builder always generates an SQL statement with inner joins declared explicitly. For example, see the following code sample:

```
LEFT JOIN
        LIB1.TRANSACTION_TYPE_DIM TRANSACTION_TYPE_DIM
            ON CASH_FLOW_FACT.TRANSACTION_TYPE_KEY =
                TRANSACTION_TYPE_DIM.TRANSACTION_TYPE_KEY
    INNER JOIN
        LIB1.TRANSACTION_DIM TRANSACTION_DIM
            ON PARTY_DIM.SEGMENT_ID = TRANSACTION_DIM.SEGMENT_ID
            AND PARTY_DIM.TRANSACTION_KEY = TRANSACTION_DIM.TRANSACTION_KEY
```

How Does the Automatic Join Feature Work?

Feature Overview

SAS Visual Data Builder attempts to join tables and queries automatically as you add them to the workspace. When you drag and drop a table or query onto the workspace, information about the table or query is retrieved from the SAS Metadata Server. For subqueries, the metadata for the output table is retrieved.

Using Foreign and Primary Keys

If primary key or foreign key information is registered in the metadata for the table that you drag and drop onto the workspace, then the data builder retrieves the foreign key and primary key information.

The data builder then iterates over each of the tables that are already in the workspace in the same sequence in which they were added to the workspace. The data builder retrieves the foreign key and primary key information for the table, and compares the length, type, and name with the key columns for the newly added table. If a match is found, then the tables are used in the join, and the columns are added as a join condition. The data builder continues to search for matches between the two tables, and it adds join conditions when possible. After a set of join tables is identified, the data builder does not continue iterating over the tables that are already in the workspace.

Matching by Name

If there is no foreign key or primary key information for the table that is dragged and dropped onto the workspace, then the data builder does not use foreign key or primary key information for the tables already in the workspace.

The data builder retrieves the column information for all of the columns in the newly added table. The data builder then iterates over each of the tables that are already in the workspace in the same sequence in which they were added to the workspace. The data builder compares the length, type, and name for each column with each column in the newly added table. If a match is found, then the tables are used in the join, and the columns are added as a join condition. The data builder does not continue to search for matches between the two tables, and it does not continue iterating over the tables that are already in the workspace.

Selecting the Join Type

When the data builder finds the first set of matching columns by comparing keys or matching names, it sets the join type for the two tables. The data builder checks the metadata for the columns to determine whether the columns are nullable. The data builder performs the following steps to set the join type:

1. If the column for the existing table is not nullable, but the column for the newly added table is nullable, the tables use a left join.
2. If the column for the existing table is nullable, but the column for the newly added table is not nullable, the tables use a right join.
3. If both columns for the existing table and newly added table are nullable, the tables use a full join.
4. If none of the previous conditions are met, the tables use an inner join.

You can specify the join type by right-clicking on the join icon, and selecting the join type from the menu.

Adding a Join

Dragging and Dropping Join Lines

To add a join by selecting a column name and dragging your pointer to another column:

1. Place your pointer on one of the columns in a table to use, and then click and drag your pointer to a column in the other table to use in the join.
2. If you want a join type other than an inner join, select the join, right-click, and then select the join type.

TIP The first table that is added to the workspace is always set as the left table. You can switch the right table and left table by using the Joins tab.

Using the Joins Tab

To add a join condition by specifying the join properties:

1. Click the **Joins** tab.
2. Click ✚.
3. Use the menus to replace **Not Selected** with the table names to use in the join. Click **Save**.
4. The default join type is an inner join. Use the menu in the **Join Type** column to select a different join type.
5. Click ✚ in the **Join conditions** area.
6. Use the menus to replace **Not Selected** with the column names to use in the join condition. Click **Save**.
7. Click 💾 to save the join condition with the query.

Removing a Join

You can remove a join by using either of the following methods:

- Select the join in the workspace, right-click, and select **Remove Join Condition**.
- Click the **Joins** tab, select the row in the table, and click ✖.

Managing Joins in a Query

Once a join is added to a query, you can change the join by selecting it in the workspace, right-clicking, and changing the join type or removing the join condition.

You can also change a join by clicking the **Joins** tab, and then selecting the row in the table. You can make the following changes:

- add and remove tables from the join list
- reorder the sequence of joins by moving them up or down
- switch the left table and right table assignments for a join
- add, remove, and change the columns that are used in the join condition

The **Joins** tab shows the join condition for the entire query. Make sure that you select a row in the upper table to set the columns in the **Join conditions** area.

The workspace shows a link between the tables that are used in a join. If you specify a left join or right join, the Venn diagram icon reflects which table provides the bulk of the data. If you rearrange the tables in the workspace (switching the left table and right table), the Venn diagram icon continues to reflect which table provides the bulk of the data. In order to change the data relationship, use either of the following methods:

- Switch the left table and right table by selecting a row from the table, and clicking ⇄.
- Change the join type from left to right or from right to left.

Best Practices for Managing Joins

In most cases, the sequence in which joins are specified on the **Joins** tab does not matter. However, it is possible that the query optimizer for the data source can perform the joins in a sequence that reduces performance. In these rare cases, you can change the sequence in which joins are specified on the **Joins** tab by selecting the join in the list, and clicking ⬆ or ⬇. This join sequence is still determined by the query optimizer, but you can control how the SQL for the join is presented to the query optimizer.

Chapter 8

Working with SAS LASR Analytic Server

Understanding Co-Located Data

One of the most powerful benefits of SAS LASR Analytic Server is the ability to read data in parallel from a co-located data provider. In this configuration, the SAS LASR Analytic Server software is installed on the same hardware as the data provider. The currently supported data providers are the following:

- SAS High-Performance Deployment of Hadoop
- Teradata Data Warehouse Appliance
- Greenplum Data Computing Appliance

SAS Visual Data Builder excels at accessing data from a variety of data sources, performing analytic data preparation, and then staging the prepared data with a co-located data provider. After the data is staged, the server can load the data to memory for exploration with SAS Visual Analytics Explorer.

The SAS Deployment Wizard registers a default library for the SAS LASR Analytic Server and a library for the co-located data provider. For the third-party vendor products, the default library name and location are as follows:

Teradata Data Warehouse Appliance
`/Products/SAS Visual Analytics High-Performance Configuration/Visual Analytics Teradata`

Greenplum Data Computing Appliance
`/Products/SAS Visual Analytics High-Performance Configuration/Visual Analytics Greenplum`

Using SAS Data in HDFS Libraries

Default Library

When your deployment includes SAS High-Performance Deployment of Hadoop, the SAS Deployment Wizard registers a library for it. This library is available for use in the SAS Folders tree, and it is located in **`/Products/SAS Visual Analytics High-Performance Configuration/Visual Analytics HDFS`**.

Staging Library

You can specify a SAS Data in HDFS library as a staging library. This is a common use because the rows for the output table are distributed among the machines in the cluster. The server can then read the data in parallel when it loads the table to memory.

You must specify a SAS LASR Analytic Server library for the output library when you use a SAS Data in HDFS library for staging.

Output Library

You can specify a SAS Data in HDFS library as an output library. The engine distributes the rows for the table to the machines in the cluster. Afterward, you can select the table from the SAS Folders tree, right-click, and select **Load a Table**. This menu option loads the table from HDFS to memory on a SAS LASR Analytic Server.

You can partition SAS Data in HDFS tables when they are used in an output library. You can select a column to use from the **Partition by** menu. Partitioning the table ensures that all of the rows with the same formatted value as the selected column are distributed to one machine in the cluster. The rows are also placed in the same HDFS block. When you load a partitioned table to memory, the partitioning information is retained, and the result is a partitioned in-memory table.

See Also

“Partitioning Tables” on page 37

Restrictions

The following restrictions apply to using SAS Data in HDFS libraries with SAS Visual Data Builder:

- You cannot specify a SAS Data in HDFS library as an input library because the SAS Data in HDFS engine is a Write-only engine.
- The **Append data** check box on the **Properties** tab is disabled. The SAS Data in HDFS engine does not support appending data.

- If you specify a SAS Data in HDFS library as an output library, you cannot view the results on the **Results** tab because the SAS Data in HDFS engine is a Write-only engine.

Using SAS LASR Analytic Server Libraries

Default Library

During installation, the SAS Deployment Wizard registers a library for SAS LASR Analytic Server. This library is available for use in the SAS Folders tree, and it is located in **`/Products/SAS Visual Analytics Administrator/Visual Analytics LASR`**.

Input Libraries

When you select a SAS LASR Analytic Server table as an input table, be aware of the following best practices if the table is large. If the table is not large, then using it for input requires no special considerations.

Here are the considerations for using a large SAS LASR Analytic Server table as an input table:

- A WHERE clause is processed in memory by the server if no aggregations or joins are used. Specify a filter on the **Where** tab so that you use only the rows that you want.
- If you want to join the table, then design one query that copies the data to the same library as the table that you want to join it with. Specify a filter on the **Where** tab, if applicable. Then, design another query that performs the join.

Output Libraries

Using a SAS LASR Analytic Server library for output is common for SAS Visual Data Builder. When you run the query, you can click **Explore Results** to automatically select the table and explore it in SAS Visual Analytics Explorer.

You can use the **Append data** check box on the **Properties** tab to add rows to an in-memory table. However, this option is not available if you use a SAS Data in HDFS library as a staging library.

Partitioning Tables

When you specify a SAS LASR Analytic Server or SAS Data in HDFS library as an output library, you can specify a partition key for the table. You can select a column to use from the **Partition by** menu.

Partitioning uses the formatted values of the partition key to group rows that have the same value for the key. All of the rows that have the same value for the key are loaded to a single machine in the cluster. For SAS LASR Analytic Server libraries, this means that the rows that have the same value for the key are in memory on one machine. For SAS

Data in HDFS libraries, all of the rows that have the same value for the key are written to a single file block on one machine. (The block is replicated to other machines for redundancy.) When the partitioned table is loaded onto a server, the partitioning remains when it is in memory.

If you select a partition key and also specify sort options for columns on the **Column Editor** tab, the sort options are passed to the engine in an ORDERBY= option. This enhancement applies to SAS LASR Analytic Server and SAS Data in HDFS libraries and can improve performance once the data is in memory.

When you specify a partition key, avoid using a variable that has few unique values. For example, partitioning by a flag column that is Boolean results in all rows on two machines because only two values are available. At the other end of the spectrum, partitioning large tables by a nearly unique key results in many partitions that have few rows.

Determining the optimal partition key can be a challenging task. However, as an example, if you tend to access data based on a customer ID, then you might improve performance by partitioning the data by customer.

See Also

SAS LASR Analytic Server: Administration Guide

Exploring Results

When you design a query that uses a SAS LASR Analytic Server library for the output table, and you run the query, the **Explore Results** button becomes available. Click the button to access SAS Visual Analytics Explorer and the results of the query.

Loading a Table to SAS LASR Analytic Server

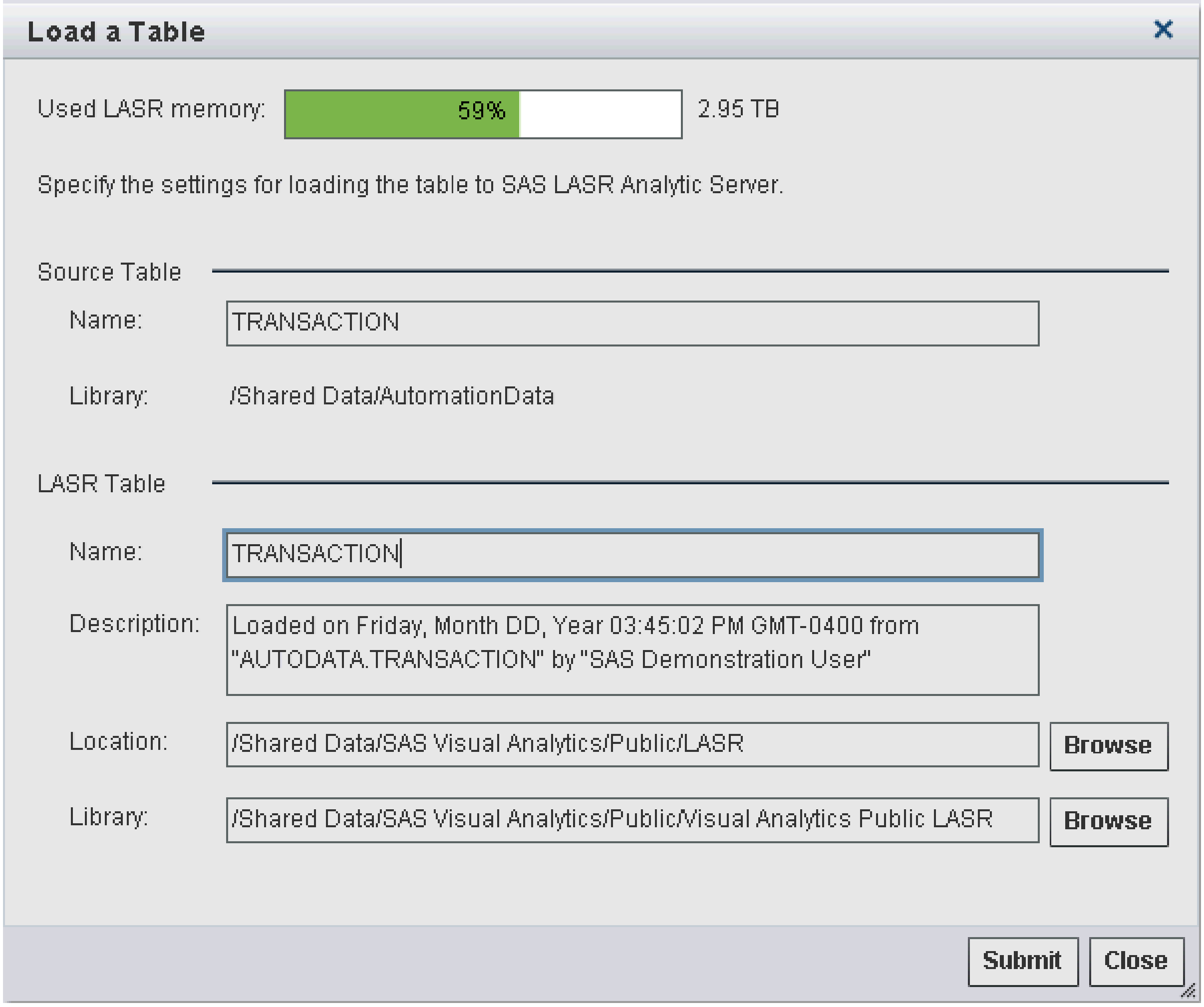
To load a table into memory on a SAS LASR Analytic Server:

1. Use the SAS Folders tree to locate the table.

   TIP You can also click , and search for the table by name and location.

2. Select the table, right-click, and select **Load a Table**.

Display 8.1 *Load a Table Dialog Box*



Load a Table

Used LASR memory: 59% 2.95 TB

Specify the settings for loading the table to SAS LASR Analytic Server.

Source Table

Name: TRANSACTION

Library: /Shared Data/AutomationData

LASR Table

Name: TRANSACTION

Description: Loaded on Friday, Month DD, Year 03:45:02 PM GMT-0400 from "AUTODATA.TRANSACTION" by "SAS Demonstration User"

Location: /Shared Data/SAS Visual Analytics/Public/LASR

Library: /Shared Data/SAS Visual Analytics/Public/Visual Analytics Public LASR

3. The fields in the **Source Table** section are filled automatically.

   Specify the following settings in the dialog box:

   Table 8.1 *Load a Table Dialog Box Properties*

Field	Description
Name	Enter the name to use for the table. This name is registered in metadata with the SAS LASR Analytic Server library.
Description	Enter a description of the data. This description is displayed by SAS clients such as SAS Visual Analytics Explorer.
Location	Click **Browse**, and select a folder to use for the table metadata.
Library	Click **Browse**, and select the SAS LASR Analytic Server library to use with the table.

4. Click **Submit**.

Monitoring Memory Usage

For deployments that use a distributed SAS LASR Analytic Server, a memory gauge is visible on multiple windows. You can use the gauge to make sure that you do not overload the server or the hardware. An example of the gauge is shown in Display 8.1 on page 39.

The server rejects requests to add tables or append rows when 80% or more of the memory is used. (The default value for the server is 75%. SAS Visual Analytics deployments set the value at 80%.) System administrators can specify the value for the server with SAS Management Console.

See Also

SAS Visual Analytics: Administration Guide

Chapter 9
Importing Data

About the Import Data Features

Why Use the Import Data Features?

The import data features enable you to perform self-service, ad hoc data analysis. You can take a Microsoft Excel spreadsheet, delimited text file, or SAS data set from your desktop and transfer it to SAS LASR Analytic Server. Or, you can add the data as a SAS data set, a database table, or SAS Data in HDFS table.

If you have SAS data sets that are stored remotely with a SAS Application Server, you can load them to memory on SAS LASR Analytic Server.

The imported data can be used to enrich existing data that is already available in your SAS deployment. You can import data directly to memory on a SAS LASR Analytic Server.

What is Local Data and Remote Data?

To understand the difference between local data and remote data, consider the following:

local data
: is available from the file systems on your desktop. This includes local file systems such as `C:\` on Windows machines and paths such as `/home/$USER` on UNIX machines. Network file systems and shares are also included, such as UNC paths like `\\nas\spreadsheets`.

remote data
: is not available from the file systems on your desktop. You can direct the SAS Application Server to access the data that is available through file systems on the remote machine.

Default Values

The import data features for both local data and remote data are configured with default values to load data to SAS LASR Analytic Server.

If you want to use a library other than the default SAS LASR Analytic Server library, you can specify different values in the **Advanced** section of the import windows. You can specify a different SAS LASR Analytic Server library, a different library type (for imported local data only, imported remote data is restricted to SAS LASR Analytic Server libraries), and a different location for the table.

Importing Large Local Data Files

When you import local data (spreadsheets, SAS data sets, or delimited text files), memory on the client PC is used to access the data files. For data files that are larger than 1G, make sure that the client machine meets the following requirements:

- The client machine should have at least 4G RAM. 8G is recommended.
- The client machine should use a 64-bit web browser and Adobe Flash Player.

  Note: For a list of supported browsers, see `http://support.sas.com/resources/thirdpartysupport/v94/browsers.html`.

When using a 64-bit web browser and Adobe Flash Player, the theoretical limit for importing local data files is 2G.

Note: Be aware that importing large local data files can require a long time to transfer and process the data. A progress indicator is provided to help you track the import.

Working with Spreadsheets

Appending Worksheets

When you import a spreadsheet (from your PC) that has multiple worksheets, you must specify how to handle the worksheets.

- Append all of the worksheets into a single table.
- Create a table for each worksheet.

TIP When you append all of the worksheets into a single table, the default output table name is the same as the filename for the spreadsheet. You can specify a different name.

Keep in mind the following considerations:

- If the worksheets are appended, it is better if the worksheets have the same number of columns. If the number of columns differs, then the table is created with the greatest number of columns.
- When you clear the **Append worksheets together** check box, a table is created for each worksheet and the table is named based on the following pattern:

  filename`_worksheet-name`

  You can specify a different name for the filename.

Special Characters and Missing Values

When you import a spreadsheet that uses special characters or has missing values, keep in mind the following considerations:

- If the filename or worksheet names use special characters, the import can fail.

  TIP Table names are restricted by the VALIDMEMNAME= SAS system option. Column names are restricted by the VALIDVARNAME= SAS system option.

- If a column has no values, it is skipped during the import process.
- If a row has missing values, but the cells are formatted (for example, Text), the import can introduce an error. The data builder combines the row with missing values with values from the following row. You can avoid this problem by formatting the cells as General.

Pivot Tables

Importing pivot tables is not supported.

Importing Local Data

To import data from a spreadsheet, delimited text file, or SAS data set, follow these steps:

1. Select **File** ⇨ **Import Local Data**, and then select the data file.
2. In the Import Local Data window, specify the following input file parameters:

   Note: For SAS data sets, there are no parameters to specify.

   Delimiter$^{\text{Applies to CSV}}$
   Select the delimiter that is used in the text file to import. You can specify a single character to use as a user-defined delimiter.

   Select worksheet$^{\text{Applies to XLS}}$
   Select **All** or the check boxes for the worksheets to import. For more information, see "Working with Spreadsheets" on page 42.

   Use data in the first row as SAS variable names
   Select this check box when the worksheet has the variable names in the first row.

   When importing spreadsheets, if only one worksheet is used, or you are not appending the worksheets, the variable names are retrieved from the first non-empty row. When you are appending worksheets, the variable names are retrieved from the first non-empty row in the first worksheet that you select for import.

   Trim spaces
   Select this check box to remove leading spaces and trailing spaces from character variables.

   Data records begin on row
   The default is to import data records from the first row. If you select **Use data in the first row as SAS variable names**, then this value increments to 2. This check box is disabled if you import a spreadsheet and select multiple worksheets.

   Encoding$^{\text{Applies to CSV}}$
   Select the encoding of the file. If you are importing UTF-8 or UTF-16 data, make sure the SAS Application Server is a Unicode server or that all of the file contents can be successfully transcoded to the encoding of the SAS Application

Server. If you are importing a UTF-16 (big-endian or little-endian) file, make sure it contains a BOM (byte-order mark).

3. (Optional) Click **Preview** to view the data. You can preview delimited text files and spreadsheets only.

   Preview displays up to 500 rows from the first 2 MB of the file.

4. Review the **Advanced** properties, and adjust them if needed.

Field	Description
Name	Specify a table name for the imported data.
Location	Click 📁, and select a metadata folder. The imported data is registered as a table in the specified location.
Library	Click 🗄, and select the library in which to import the data. If you select the **Show SAS LASR Analytic Server libraries only** check box, then the list is filtered to show those libraries only.

5. Click **OK**.

Importing Remote Data

To import a SAS data set that is accessible from your SAS Application Server, follow these steps:

1. Select **File** ⇨ **Import Remote Data**, and then select the data file.

   Note: Remember that the data files and directories are on the remote machine, not on your PC. In the **Basic Properties** table, the **Date Created**, **Description**, and **Keywords** fields are not used.

2. In the Import Remote Data window, review the **Advanced** properties, and adjust them if needed:

Field	Description
Name	Specify a table name for the imported data.
Location	Click 📁, and select a metadata folder. The imported data is registered as a table in the specified location.
Library	Click 🗄, and select the library in which to import the data.

3. Click **OK**.

Chapter 10
Importing SAS Information Maps

About SAS Information Maps

A SAS Information Map is business metadata that is applied on top of the data sources in your data warehouse. (Metadata is information about the structure and content of data. An information map does not contain any physical data.) Information maps provide business users with a user-friendly way to query data and get results for themselves.

SAS Information Maps are created with either SAS Information Map Studio or the INFOMAPS procedure. SAS Visual Data Builder cannot create, edit, or save an information map. However, the data builder can import the business logic from a relational information map and represent it as SAS statements that use the SQL procedure.

See Also

Base SAS Guide to Information Maps

Importing a SAS Information Map

To import a SAS Information Map:

1. Use the SAS Folders tree to locate the map (▦).
2. Select the map, right-click, and select **Import Query**.

   A new query is created with the same name as the map. The output table is automatically set with the same name as the map.

The business logic is imported from the map and represented as SAS statements on the **Code** tab.

TIP If you want to join a table, save the query, and then create a new query and add it as a subquery.

See Also

"Customizing Code" on page 49

Limitations and Restrictions

SAS Visual Data Builder has the following limitations and restrictions for working with information maps:

- The information map cannot use more than 50 physical tables. A table that is used more than once in a self join counts as one physical table.
- The information map cannot use more that 5000 physical columns. A column that is used in more than one data item counts as one physical column.
- Prompts are not supported. Even if a prompt has a default value, the default value is not included.
- Data items that are based on business data are not supported. For example, the equation `Dataitem1 = Year + 2` (where Year is a data item) is not included in the query.
- Data items that are based on physical columns are not supported. For example, the equation `Dataitem2 = FirstName || LastName` (where FirstName and LastName are columns) is not included in the query.
- The output table is automatically named with the same name as the map. The name field is limited to 32 characters.

Chapter 11

Customizing Code

Using the Code Tab

You can click the **Code** tab to view the SQL statements that are generated by the data preparation expressions on the **Design** tab.

The **Code** tab can also be used to enter custom code.

- The Preprocess and Postprocess views enable you to enter SAS statements that are run before and after the query runs.
- The All Code view shows the generated SQL statements. You can click the 🔒 button to unlock the view so that you can manually edit all of the SAS statements for the query. After you have unlocked the view, the button icon changes to 🔓.

After you have unlocked the view, you cannot continue using the **Design** tab to edit the query, and the Preprocess and Postprocess views are disabled. However, if you have not saved your changes to the code, you can click the ↶ button to be able to use the **Design** tab.

Preprocess and Postprocess Code

In some cases, you might want to assign SAS options, load format catalogs, use LIBNAME statements, or run macros before running a query. You can unlock the code to enter the statements, but that prevents you from using the **Design** tab. However, the Preprocess view or Postprocess view might enable you to supplement the query while using the code generation features of the **Design** tab.

Any SAS statements that you enter in the Preprocess view or Postprocess view are included in the All Code view.

Considerations for Manually Editing Code

Before You Manually Edit Code

The code generation features in SAS Visual Data Builder can generate a lot of SAS statements with a few simple clicks. Depending on your needs, you might benefit by using the **Design** tab for adding input tables, performing joins, and filtering data before you customize the generated code.

The following features and settings must be specified before you unlock the code:

Unique values
Selecting this check box determines whether the DISTINCT keyword is included in the generated PROC SQL statement. After you unlock the code, the check box cannot be used to include the keyword.

Append data
Selecting this check box enables you to use the generated SAS statements for appending data to the output table before you unlock the code.

Output table
You must specify the table name, location, and library, and you must save the query before you unlock the code. When you save the query, the metadata for the table is registered.

Use a staging table
Selecting this check box registers the metadata for the table when you save the query. Select the check box, specify the library, and save before you unlock the code.

Input and Output Libraries

When you unlock the code for manual editing, you must specify the libraries to use for your tables. You can do this with LIBNAME statements. If you dragged and dropped the tables onto the workspace before you unlocked the code for manual editing, the LIBNAME statements for the tables are automatically included.

See Also

- *SAS Statements: Reference*
- *SAS/ACCESS for Relational Databases: Reference*

Output and Staging Tables

Specify the table names, locations, and libraries on the **Properties** tab just as you do on the **Design** tab.

Note: Specify the tables and save the query before you unlock the code. If you do not specify the table name or the table name is already in use, you are not able to save the query.

See Also

SAS Language Interfaces to Metadata

Validating Custom Code

If you entered SAS statements in the Preprocess or Postprocess views, those statements are not validated when you click ☑.

If you unlock the code in the All Code view, the ☑ button is disabled. You cannot validate customized code.

Chapter 12
Scheduling Queries

About Scheduling Queries

How Does the Scheduling Feature Work?

When you have a query open in the workspace, you can click 🕒 to schedule the query. When you schedule a query, SAS Visual Data Builder performs the following operations:

1. Creates a *job* that performs the query operations.
2. Creates a *deployed job* from the job.
3. Places the job into a new *deployed flow*.
4. Schedules the flow on a *scheduling server*.

You can use SAS Visual Data Builder to reschedule the query based on specified conditions (for example, run immediately or run whenever a trigger condition is met).

The job, deployed job, and deployed flow are metadata objects. The data builder stores them in the same metadata folder with the query. The metadata objects are named based on the following pattern:

```
vdb_query_id_timestamp
```

Editing Queries That Are Already Scheduled

If you edit a query that is already scheduled, you must click 🕘 again so that the SAS statements for the query are regenerated and saved.

Caution about Scheduling Queries to Run Now

When you schedule a query, one of the options is to run the query immediately. Select **Run now** in the Schedule window.

Performing the following steps results in an error condition:

1. Use a SAS data set for the output table of the query.
2. Run the query.
3. Click the **Results** tab to look at the output.
4. Schedule the query by selecting **Run now**.

These steps result in an error condition because SAS unlocks a SAS data set when it is opened for reading. When step 3 is performed, the output table is locked, and no other process can overwrite the output table. The following message is included in the SAS log:

Locked Error Message

```
ERROR: A lock is not available for OUTPUTTABLE.

ERROR: Lock held by process xxxx.
```

You can avoid this error condition. If you want to schedule the query to run now, close the query, open it again, and then schedule it to run now. Alternatively, you can schedule the query to run in the future, and then close the query.

Scheduling Preferences

Default Scheduling Server

You can specify a different scheduling server in your application preferences. Any queries that you schedule after you specify a different scheduling server will use the new scheduling server.

Use the Server Manager plug-in to SAS Management Console to identify the scheduling servers that are included in your deployment. By default, your deployment includes a server that is named `Operating System Services -` *hostname.example.com*. This server is used as the default scheduling server.

The SAS Distributed In-Process Services Scheduling Server is an alternative server. To use this server, change the scheduling server. The default name is `SAS Distributed In-Process Services Scheduling Server`.

Some deployments include the Platform Suite for SAS server. To use this server, change the scheduling server. The default name is `Platform Process Manager`.

In all cases, when you schedule a new query, SAS Visual Data Builder retrieves your default scheduling server, and uses that value to look up the scheduling server in SAS metadata. The data builder uses the first server that matches the value in SAS metadata.

Including the host name, such as `Operating System Services -` *hostname.example.com* ensures that the data builder uses the server that you specify.

Default Batch Server

You can specify a different batch server in your application preferences. Consider the following before you change the default batch server:

- The batch server must be registered in metadata as a component of a SAS Application Server that you can access.
- You must specify the same SAS Application Server as your default application server in your preferences.

As with the default scheduling server, SAS Visual Data Builder retrieves your default batch server, and uses that value to look up the batch server in SAS metadata the first time you schedule the query. The data builder uses the first server that matches the value in SAS metadata.

Default Deployment Directory

A deployment directory is a SAS metadata object that represents the following items:

- the name of the SAS Application Server with which the deployment directory is associated (the default value is **`SASApp`**)
- a name for the deployment directory (the default value is **`Batch Jobs`**)
- the path to the deployment directory (the default value is ***`SAS-config-dir`***`/`**`Lev1/SASApp/SASEnvironment/SASCode/Jobs`**)

When you schedule a query, the SAS statements for the query are saved in a file. The file is saved in the path that is associated with the deployment directory. The file is named based on the same pattern that is described in "How Does the Scheduling Feature Work?" on page 53.

SAS Visual Data Builder retrieves your deployment directory, and uses that value to look up the deployment directory in SAS metadata the first time you schedule a query. The data builder uses the first server that matches the value in SAS metadata.

You can specify a different name for the default deployment directory. For more information about deployment directories and using the Server Manager plug-in to SAS Management Console, see *Scheduling in SAS*.

When Are the Scheduling Preferences Used?

Any of the preferences that you change are used the next time you create a query and schedule it. If you edit an existing query that is already scheduled, the existing settings for the scheduling server, batch server, and deployment directory are not updated with the changes. To change the settings for existing queries that are already scheduled, use SAS Management Console to redeploy the deployed job for the query.

Creating Events

Why Use Events?

Events specify conditions that must be met before a step in the flow can take place. You can use SAS Visual Data Builder to create two types of events.

- *Time events* are evaluated based on a specified time being reached.
- *File events* are evaluated based on the state of a specified file.

You can create file events if your deployment includes a scheduling server that supports them, and the flow is deployed to that scheduling server. Time and file events can be used as triggers (conditions that must be met in order to run a flow on the scheduling server).

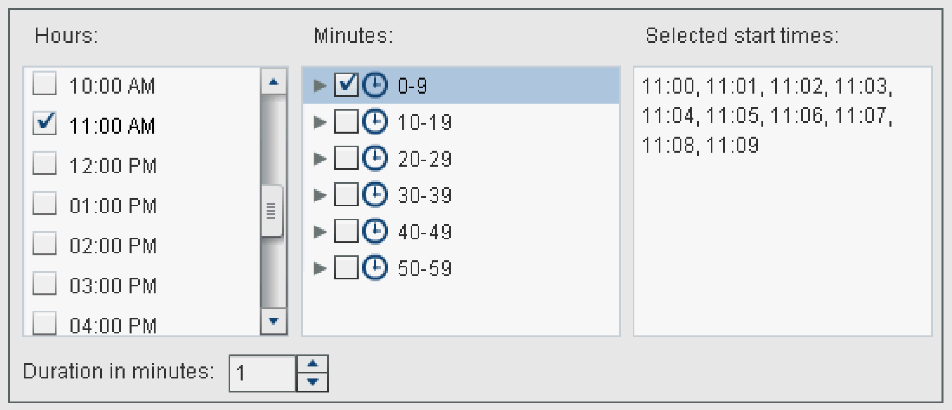
Creating a Time Event

You can create a time event and use it as a trigger.

To create a time event:

1. In the Schedule window, select **Select one or more triggers for this query**, and then click **New Time Event**.
2. In the New Time Event window, specify whether the time event should happen one time only or more than once. If the time event should happen one time only, specify the date and time for the time event.
3. If the time event should happen more than once, select **More than once**, and then select a radio button for how often the time event should repeat (such as hourly, weekly, or yearly).
4. Specify the details of when the time event should repeat. The specific fields that are available depend on the recurrence interval that you select.

   If the recurrence interval requires you to select start times, use the **Hours** and **Minutes** check boxes to select the times. The **Minutes** area contains groupings of 10-minute intervals. Selecting a check box for a minute grouping selects all of the minutes in that grouping.

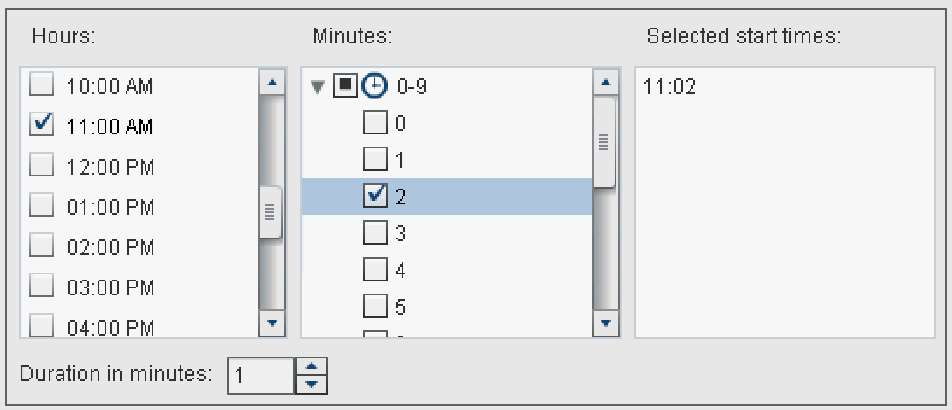
   Hours:
   - □ 10:00 AM
   - ☑ 11:00 AM
   - □ 12:00 PM
   - □ 01:00 PM
   - □ 02:00 PM
   - □ 03:00 PM
   - □ 04:00 PM

   Minutes:
   - ▶ ☑ 0-9
   - ▶ □ 10-19
   - ▶ □ 20-29
   - ▶ □ 30-39
   - ▶ □ 40-49
   - ▶ □ 50-59

   Selected start times:
   11:00, 11:01, 11:02, 11:03, 11:04, 11:05, 11:06, 11:07, 11:08, 11:09

   Duration in minutes: 1

   To select individual minutes, expand the grouping.



Hours:

□ 10:00 AM
☑ 11:00 AM
□ 12:00 PM
□ 01:00 PM
□ 02:00 PM
□ 03:00 PM
□ 04:00 PM

Minutes:

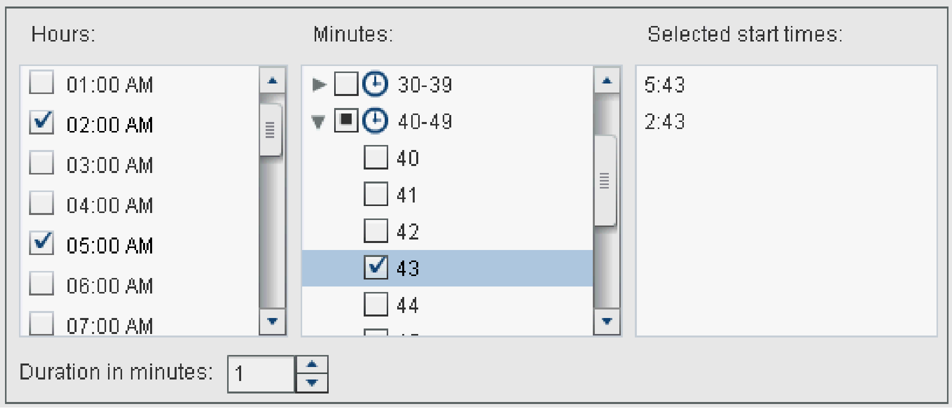
▼ ■ 0-9
□ 0
□ 1
☑ 2
□ 3
□ 4
□ 5

Selected start times:

11:02

Duration in minutes: 1

If you select multiple values for **Hours**, all of the selections for **Minutes** apply to all of the selected hours. For example, if you select 02:00 AM and 05:00 AM in the **Hours** area, and 43 in the **Minutes** area, the time event is scheduled for 2:43 AM and 5:43 AM.



Hours:

□ 01:00 AM
☑ 02:00 AM
□ 03:00 AM
□ 04:00 AM
☑ 05:00 AM
□ 06:00 AM
□ 07:00 AM

Minutes:

▶ □ 30-39
▼ ■ 40-49
□ 40
□ 41
□ 42
☑ 43
□ 44

Selected start times:

5:43
2:43

Duration in minutes: 1

5. If needed, specify the start date and end date for the time event. The default is to start at the current date and time and not to have an end date.

Creating a File Event

You can create a file event and use it as a trigger. The file event can check for various file conditions, such as its existence, size, or age.

Note: Not all scheduling servers support file events. Platform Suite for SAS and the SAS Distributed In-Process Services Scheduling Server are two servers that do support file events. The **New File Event** button that is used to create a file event is available only when the scheduling server supports file events.

To create a file event:

1. In the Schedule window, select **Select one or more triggers for this query**, and then click **New File Event**.
2. In the New File Event window, specify or select the file to use for the file event.
3. Select the condition to evaluate the file to make the file event true. For example, if you selected **not exist** for the condition, the file event would be true only if the selected file was not in the specified location.
4. If needed, specify the details (such as size or age) about the condition.

Exporting Queries as Jobs

For deployments that include SAS Data Integration Studio and prefer to deploy jobs, flows, and schedule flows manually, you can export a query as a job, and then perform the deployment steps.

This feature enables you to work with a query interactively while you design it, and then you can export the query as a job for automating the operations. After the job is exported, you can deploy the job for scheduling with SAS Data Integration Studio. For more information about deploying jobs, see *Scheduling in SAS*.

After a job is stored in metadata, you can open the job with SAS Data Integration Studio and edit it. This might be necessary if you need to add or remove columns or change the column expression in a calculated column.

If the deployment does not include SAS Data Integration Studio, then you can modify the query and export it as a job again. However, you cannot overwrite job objects in metadata with SAS Visual Data Builder. If you need to delete objects from metadata (such as jobs, libraries, or tables), use SAS Management Console.

To export a query as a job:

1. Use the SAS Folders tree to locate the query.
2. Select the query, right-click, and select **Export as a job**.
3. In the Export as a Job window, enter a name and specify a location. Click **Export**.

Additional Scheduling Resources

SAS Visual Data Builder provides an easy-to-use method for taking a query and making it available for scheduling as a flow. In addition, the Schedule Manager plug-in to SAS Management Console provides additional resources for managing deployed flows, job dependencies, and scheduling servers.

For users that are familiar with the Schedule Manager plug-in, be aware that SAS Visual Data Builder does not support scheduling a query without a trigger. In the Schedule Manager plug-in, this option is specified as the **Manually to the scheduling server** option. If this is your preferred method for scheduling, you can use the Schedule Manager plug-in to modify the flow for the scheduled query. If you use SAS Visual Data Builder to schedule the query afterward, the setting is lost, and you will need to repeat your steps with the Schedule Manager plug-in.

See Also

Scheduling in SAS

Chapter 13
Using the Results Tab

About the Results Tab

After you have designed your query and clicked ▶, the query runs, and you are prompted to determine whether you want to view the results.

If you click **Yes**, the data builder retrieves the data and shows you the results on the **Results** tab.

Note: For large data sets, retrieving and displaying the data can require a long time.

Data Pages

When you view data, a SAS server retrieves the data from the data set. Instead of returning all of the data, the data viewer requests a page of data. You can set the page size to between 20 and 2000 rows.

A slider is provided at the bottom of the **Results** tab so that you can navigate through the pages and browse the entire data set.

Navigate within the Data

Navigate to a row in the table by entering the row number in the **Go to row** field. The **Go to first row** and **Go to last row** buttons provide navigation to the first and last rows of the table.

Find

Search for text or numbers in the data set by clicking , typing in the **Find** field, and pressing Enter. The find feature searches through the rows in the data set for the value, and it highlights the first row with the value. You can navigate through the results by clicking **Find next** and **Find previous**.

Click to set advanced find options. The following list identifies the options:

- **Options** tab
  - **Exactly matches the specified string**
  - **Contains the specified string** (default setting)
  - **Starts with the specified string**
  - **Case sensitive**
  - **Trim leading and trailing spaces** (default setting)
- **Columns** tab

  Enable and clear check boxes to specify the columns to search.

Filter and Sort

Sort the data by clicking , and then choosing the columns and sort order. The **Generated SQL statement** field shows the sort criteria.

Filter the data by clicking , and setting the following options:

- The **Sample** tab is used to limit the number of rows to return or to select distinct values.
- The **Row Filter** table is used to subset data with a WHERE clause. You can filter on the values in multiple columns, set ranges for numeric values, and use IN and NOT IN criteria for character values. If the number of distinct values is less than 50, you can select check boxes for the values. If there are more than 50 distinct values, you must enter the values to use for filtering.
- The **Column Filter** tab is used to select the columns to display.

Export Data

You can save the data that you view as a comma-separated values file. After you click ↗, you can choose to export the rows on the current data page, all of the rows, or a range of rows. The export feature has a limit of exporting 200,000 rows.

If you choose to export all of the data or a range of rows, you have to click **Retrieve Data** before you can click **Export as CSV**.

Print

You can print the data that you view. After you click 🖨, you can set the following options:

- number of rows to print
- margin sizes
- header and footer
- scaling

You can click **Print Preview** to preview the settings. Be aware that options for page size and orientation apply to the preview only. You must set those options when you click **Print** and select your printer.

Column Headings

Use the **Headings** menu to control the appearance of the column headings. You can view the column name from the data set, the column label from the data set, or a combination of the two. If a column does not have a label in the data set, then the column name is used.

Glossary

dependency
a trigger condition that must be met before a job can run in a scheduled flow.

deployed job
a job that has been saved in a deployment directory and can be scheduled.

deployment directory
the location for generated SAS DATA step programs that will be executed by the batch server as part of a scheduled flow.

file event
a file-related occurrence that is used as a trigger in a scheduled flow. For example, a file event occurs when a scheduling server determines that a specified file exists.

flow
a set of jobs and associated dependencies that is scheduled in the Schedule Manager plug-in in SAS Management Console.

job
a collection of SAS tasks that can create output.

job event
a job-related occurrence that is used as a trigger in a scheduled flow. For example, a job event occurs when the scheduling server issues a command to determine whether a job ran successfully.

job flow
a group of jobs and their dependencies, including dependencies on other jobs, on files, or on specified dates and times.

local data
data that is accessible through the file systems on a computer. This includes data on hard drives or available through network file systems.

query
a set of instructions that requests particular information from one or more data sources.

remote data

data that is not accessible through the file systems available to a computer. To use remote data, you must direct a SAS server to access the data that is available through file systems on the remote machine.

SAS Management Console

a Java application that provides a single user interface for performing SAS administrative tasks.

scheduling server

a server that runs deployed jobs in a scheduled flow. Before running a job, the scheduling server determines when the schedule for the deployed job as well as all of the dependencies for the job have been met.

subquery

a query-expression that is nested as part of another query- expression. Depending on the clause that contains it, a subquery can return a single value or multiple values.

Index