SAS® Studio: Working with Flows

2023.12*

* This document might apply to additional versions of the software. Open this document in SAS Help Center and click on the version in the banner to see all available versions.

SAS® Documentation
December 8, 2023

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2023. *SAS® Studio: Working with Flows*. Cary, NC: SAS Institute Inc.

SAS® Studio: Working with Flows

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

December 2023

v_014-P1:webeditorflows

Contents

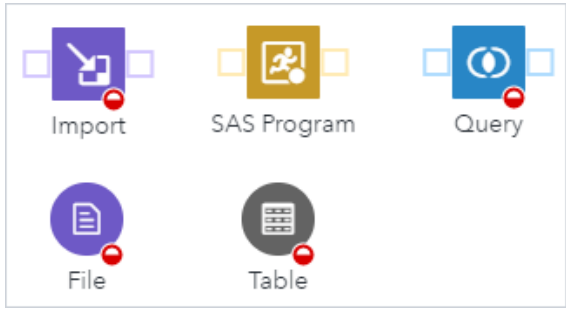
1

Introduction to Flows

What Is a Flow?

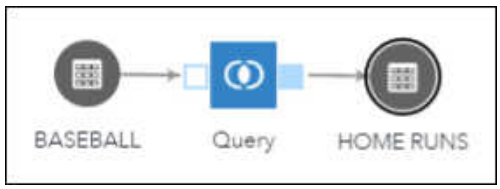
A *flow* is a sequence of operations on data. Data and operations are represented in SAS Studio by steps that you can access from the **Steps** section of the navigation pane. Each step in a flow is represented by a node on the flow canvas.

The nodes on this flow canvas represent some of the steps that are available in SAS Studio.



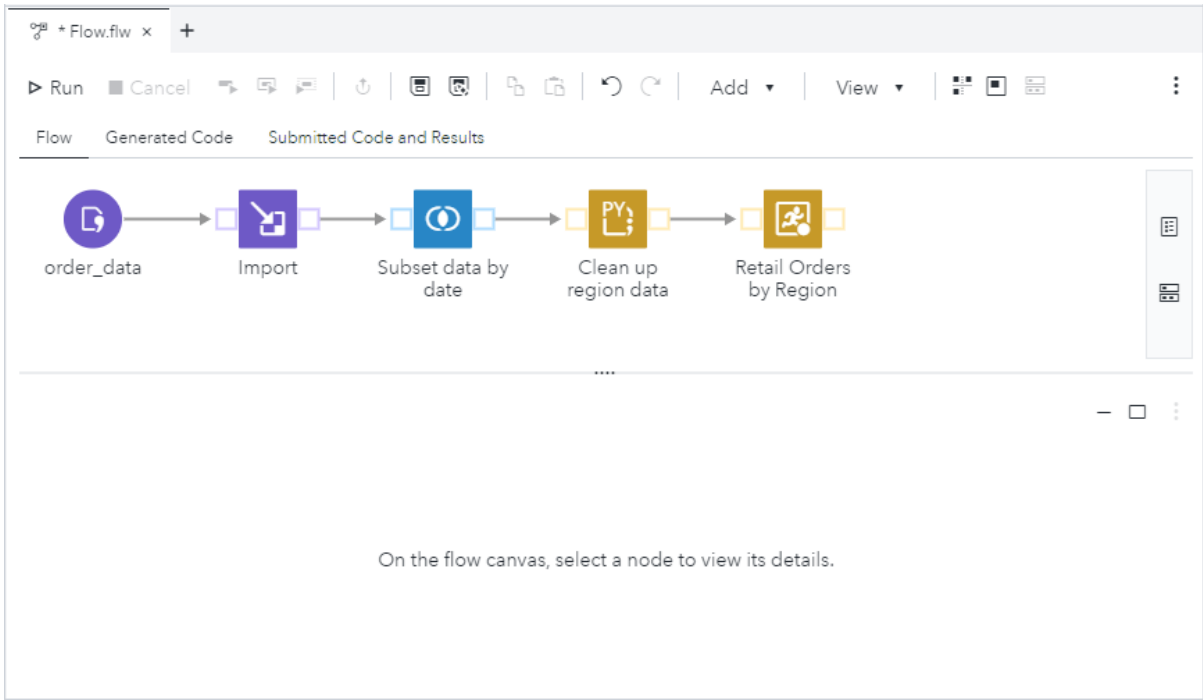
A flow can include a series of nodes in which the output of one node is the input to another node. As you build a flow, SAS Studio automatically generates SAS code for each node. You can use flows to prepare data for reporting and analysis.

This example uses the Table and Query steps to create a flow that analyzes baseball data.



- BASEBALL is a Table node that provides the input data to the query.
- Query is a Query node that selects the number of home runs for each player in the BASEBALL table.
- HOME RUNS is a Table node that specifies the output table for the query.

You can also use a flow to combine different technologies in a single workflow. For example, you can create a flow that uses the Import step to convert an external .csv file to a SAS table, the Query step to subset the data, the Python Program step to cleanse the data, and the SAS Program step to create a report.

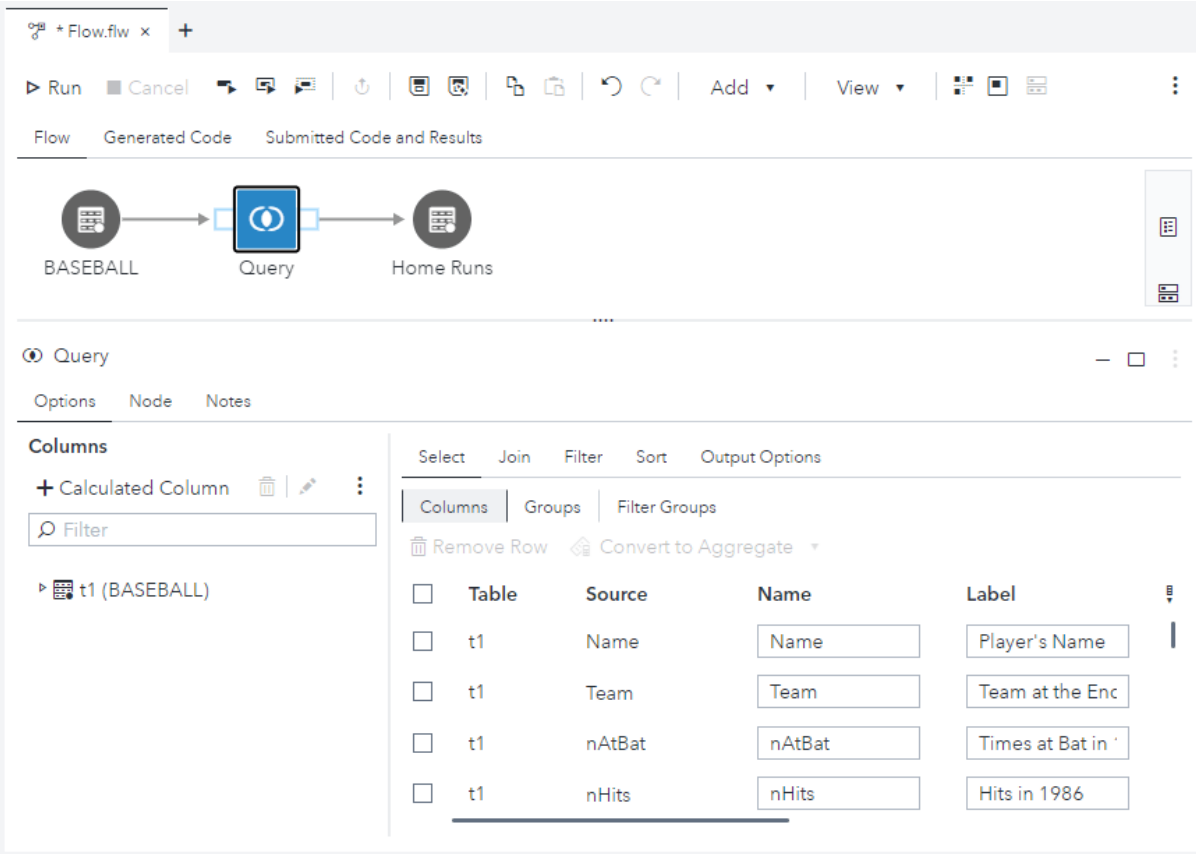


The types of steps that you can add to a flow are listed in the **Steps** section of the navigation pane. For more information, see “What Types of Steps Can I Add to a Flow?” on page 9.

Understanding the Flow Tab

About the Flow Tab

The flow functionality is available from the **Flow** tab.



The Flow tab has two areas:

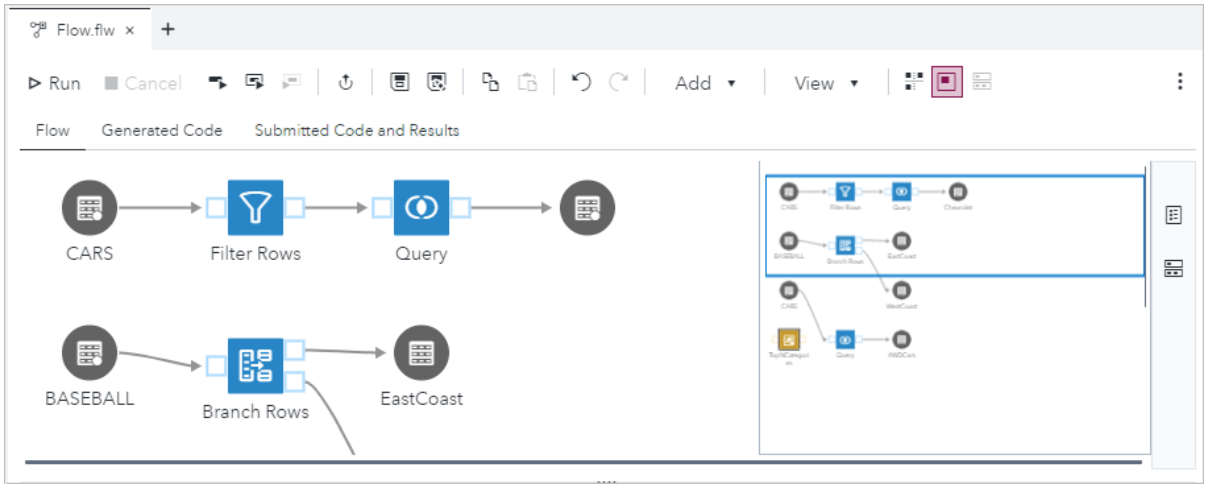
- The flow canvas enables you to build the sequence of nodes and manage the flow as a whole. You can also use the **Generated Code** and **Submitted Code and Results** tabs to view the code and log that SAS Studio automatically generates as you build the flow. The **Submitted Code and Results** tab also displays any results and output data that are generated when you run a node.
- The node details under the canvas enable you to manage the attributes of a selected node in the flow, including node options and node properties. If you select a node on the flow canvas, the details for that node appear under the flow. You can specify various options that define the behavior of the selected node.

You must specify a minimum set of options for each node in the flow in order to run the flow successfully.

Using the Overview Map

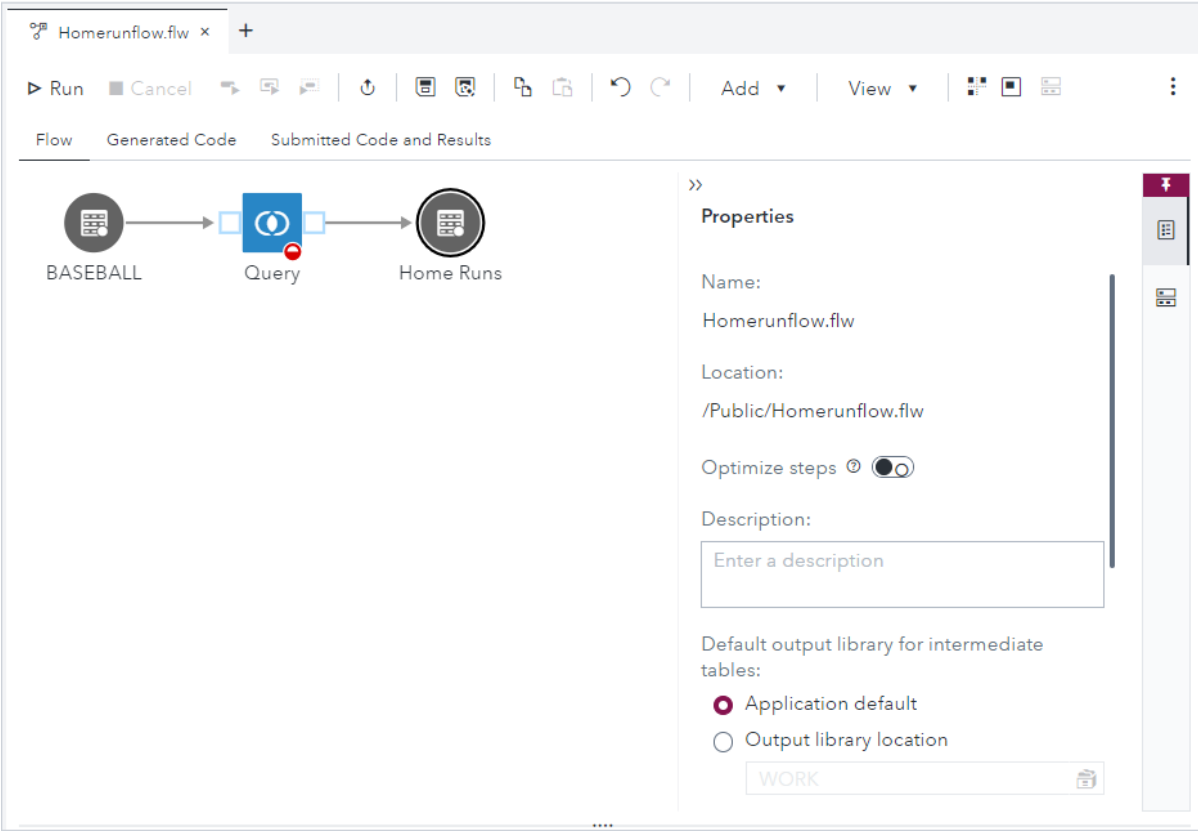
You can use the overview map to view your entire flow in a small window on the flow canvas. This feature is useful when you have a large, complex flow that you must scroll to view. Drag the rectangle on the map to move the view vertically and horizontally.

To toggle the overview map on and off, click ▣ on the flow toolbar.



Viewing Flow Properties

By default, the flow properties are collapsed on the right side of the canvas. When you expand the Properties pane, you can edit the Description property to specify information about the flow. You can also select options to optimize steps in the flow and change the default output library location for the flow.



You can display, collapse, and pin the Properties pane:

- To display the Properties pane when it is collapsed, click .
- To collapse the Properties pane, click .
- To pin the Properties pane so that it remains visible when you scroll the flow horizontally, click . The Properties pane is pinned by default when it is displayed. If you do not pin the Properties pane, some nodes on the right side of the flow canvas might not be visible.

You can optimize the performance of your flow by combining the code generation of adjacent nodes so that the table that is associated with the output port of the first node is not created. For more information, see "Optimizing Steps in a Flow" on page 295.

You can use the **Default output library for intermediate tables** property to change the default output library for temporary tables in the current flow:

- **Application default** uses the default library that is specified for the application. The default library is determined by the **Default output library for intermediate tables** application preference. For more information, see "Setting Tables Preferences" in *SAS Studio: User's Guide*.
- **Output library location** uses the library that you specify.

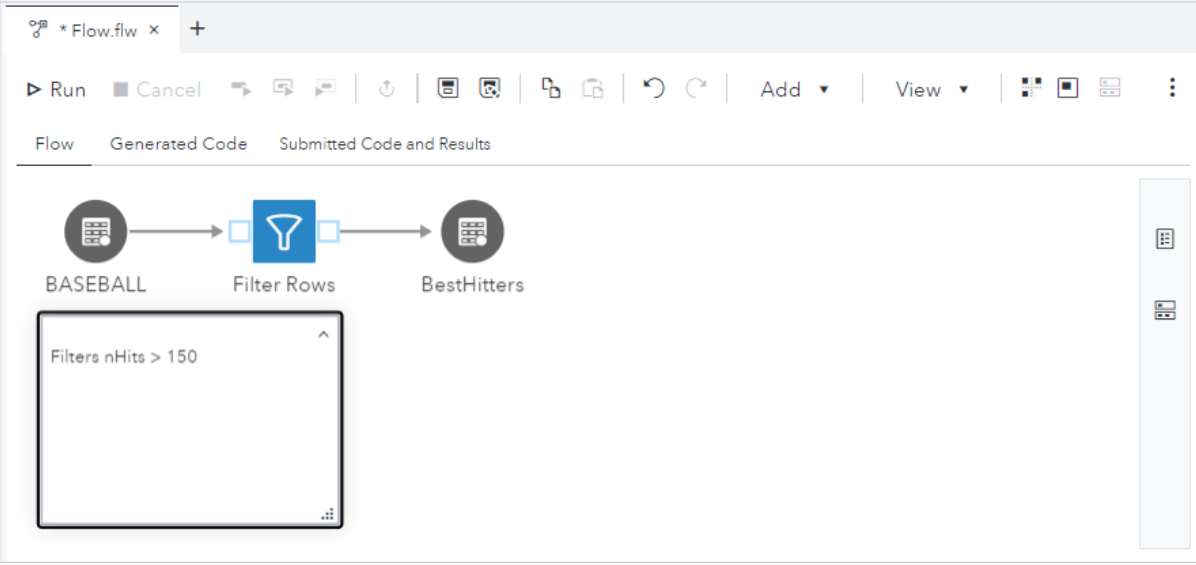
Adding a Note to a Flow

You can annotate your flow by adding notes to the flow canvas. Notes that are added to the flow are not associated with a specific node. You can position the notes anywhere in the flow, and you can cut, copy, and paste notes within and between flows.

To add a note to a flow:

1. On the flow toolbar, click **Add** ⇨ **Notes**.
2. Enter the text of the note in the note box.

Note: To delete a note, right-click the note and select **Delete**.



You can collapse and expand a note in a flow:

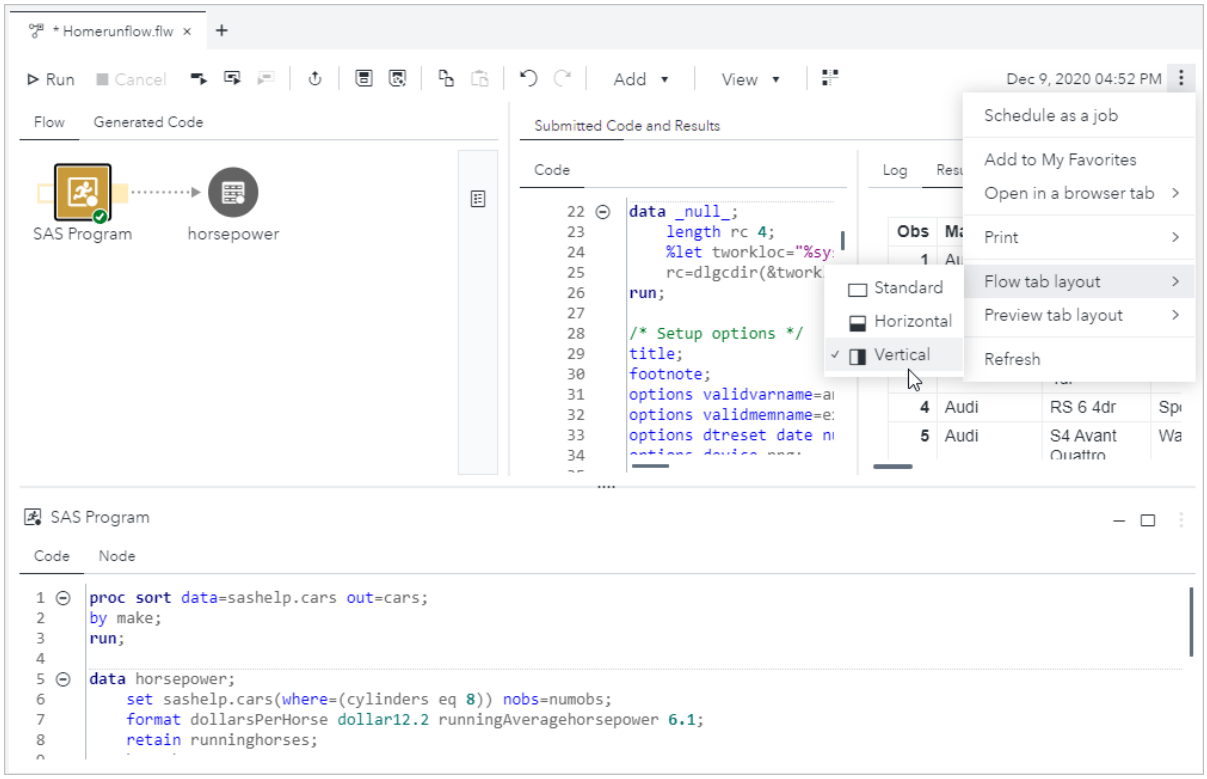
- To collapse a note in the flow, click ^ in the upper right corner of the note.
- To expand a collapsed note, double-click the note.

Customizing the Flow Tab

You can change the layout of the default flow tabs so that the **Submitted Code and Results** tab is displayed separately from the **Flow** and **Generated Code** tabs. You can also choose to display the preview tabs horizontally or vertically.

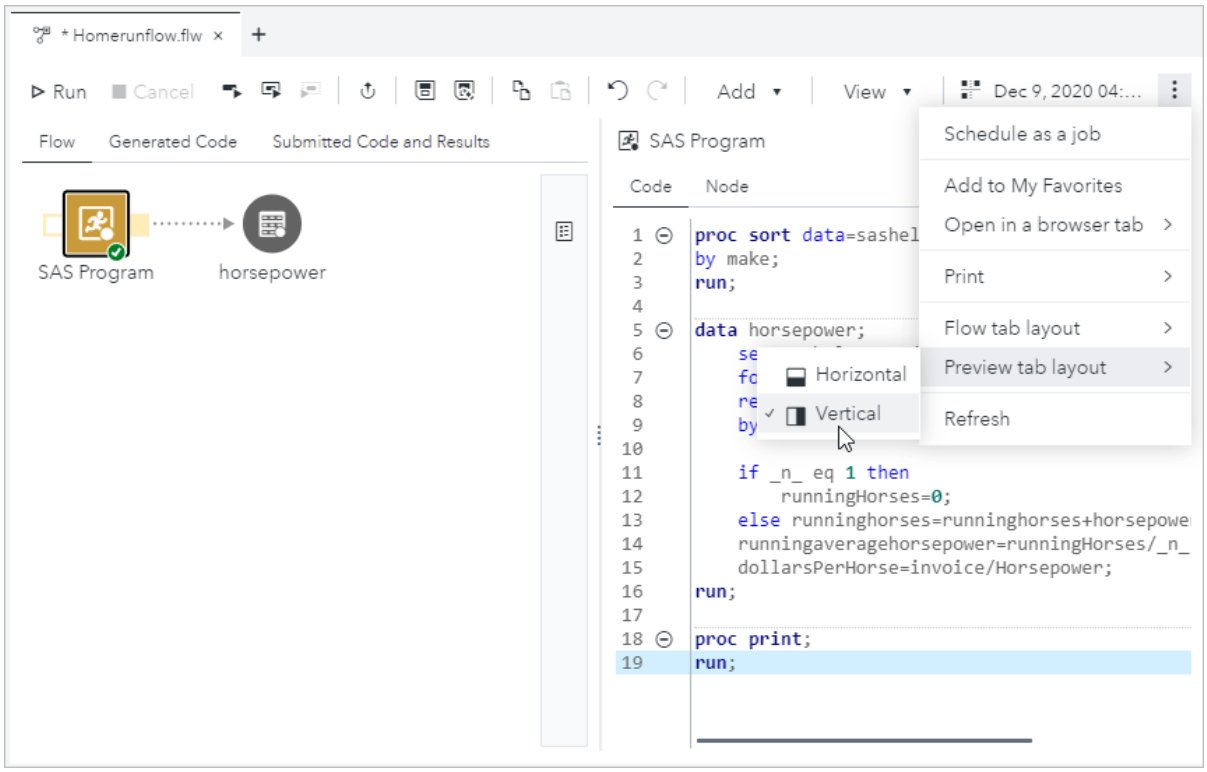
To change the layout of the default flow tabs:

- On the flow toolbar, click ⋮ and select **Flow tab layout**. Select the layout that you want to use. The default layout is `Standard`.



To change the layout of the preview tabs:

- On the flow toolbar, click ⋮ and select **Preview tab layout**. Select the layout that you want to use. The default layout is `Horizontal`.



Creating a Flow

You can create a flow in these ways:

- On the **Start Page** tab, click **Build a flow**.
- From the SAS Studio menu, select **New** ⇨ **Flow**.

Opening a Flow

Flows can be saved to SAS Content or to a SAS Compute Server folder in the **Explorer** section of the navigation pane. Flows are saved as *.flw files. To open a saved flow, navigate to the appropriate folder and perform one of these actions:

- Double-click a flow to open it.
- Right-click a flow and select **Open**.
- Select a flow and drag it to an open space next to any open tab in the work area. The tip changes from `Invalid` to `Open`. Release the flow to open it.

Understanding Steps and Nodes

About Steps and Flow Nodes

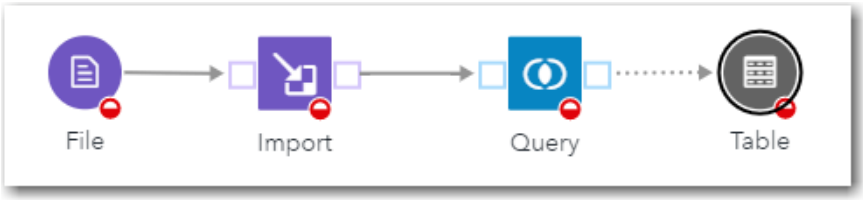
SAS Studio is shipped with many predefined steps that include queries and data transformations. Steps are represented as nodes in a flow. Flows are built from a sequence of data and operational nodes, which you can connect in order to specify the order of execution.

Data steps, such as the File and Table steps, represent data in a flow. Operational steps, such as the Import, Query, and SAS Program steps, represent operations that can be performed on data. The File and Table steps are not considered operational steps.

You can add steps to a flow as placeholders in order to create the structure of your flow, and then specify the attributes and content of the nodes later. For example, you could create a flow with placeholders for an external file, an Import step, a Query step, and a SAS Table step. When you are ready, you can specify the

external file and output table that you want to use as well as the options for the Import and Query nodes.

Note: You cannot undo changes to nodes by clicking ↶ on the toolbar.



You can also add saved files to a flow from different sections of the navigation pane. Each node can have input and output ports, depending on the node type.

TIP You can move individual nodes by dragging them around the flow canvas. To move multiple nodes as a group, use your mouse pointer to draw a box around the nodes that you want to move. The nodes in the box are selected. Click one of the selected nodes to drag the entire group to another location on the flow canvas.

What Types of Steps Can I Add to a Flow?

The types of steps that you can add to a flow are listed in the **Steps** section of the navigation pane. The steps are organized into categories that indicate the function that they perform. Here are the basic steps that are available in SAS Studio:

- Export - exports data to an external file.
- File - references an external file.
- Import - converts an external file to a SAS data set.
- Table - references a SAS data set from a SAS library.
- Python Program - enables you to write a new Python program or open a saved Python program.
- SAS Program - enables you to write a new SAS program or open a saved SAS program or snippet.
- Query - extracts data from one or more tables according to criteria that you specify.
- Sort - enables you to order your data by the values of one or more columns.

You can use the node details under the flow canvas to specify the attributes and content of the node. Every node has a Node tab, which you can use to specify a name and description of the node.

For a complete list of the steps that are available in each license of SAS Studio, see “Summary of Flow Functionality” in *SAS Studio: User’s Guide*.

Note: Not all steps are available in all licenses of SAS Studio.

Adding Steps to a Flow

When you have a flow open in the work area, you can use the **Steps** section of the navigation pane or click **Add** on the flow toolbar to add steps to a flow.

To add one or more saved files to a flow, expand the **Explorer** section of the navigation pane and drag the appropriate files to the flow canvas. You can add the following types of saved files to a flow:

- SAS program files (*.sas)
- Python program files (*.py)
- Comma-separated values files (*.csv)
- Delimited files (*.tsv, *.tab, or *.dlm)
- Text files (*.txt)
- Flow files (*.flw)

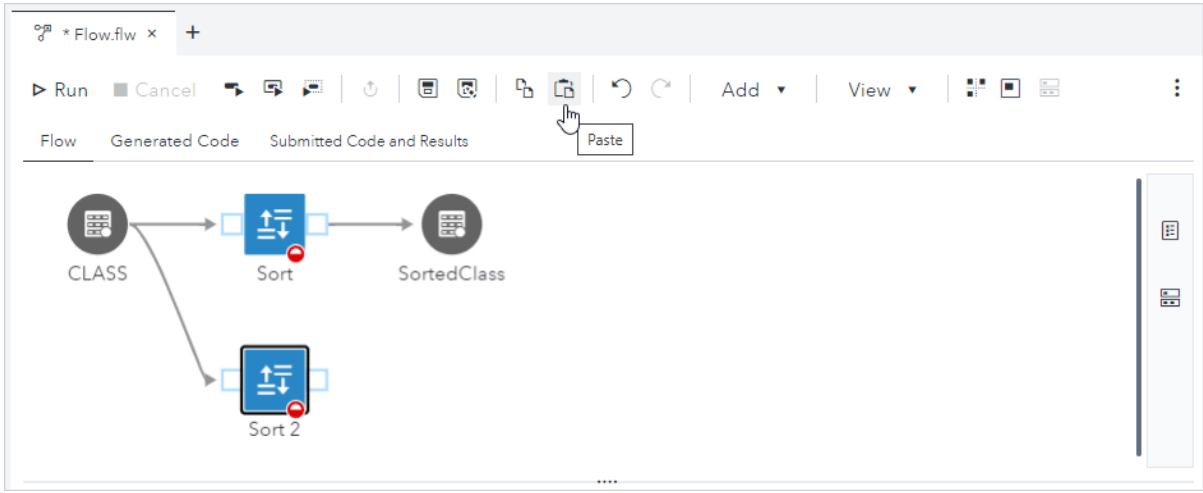
Note: The subflow feature is available only if your site licenses SAS Studio Analyst. For more information, see “About Subflows” on page 39.

You cannot add *.sas7bdat, *.ctm, *.ctk, *.ctl, or *.cqy files to a flow.

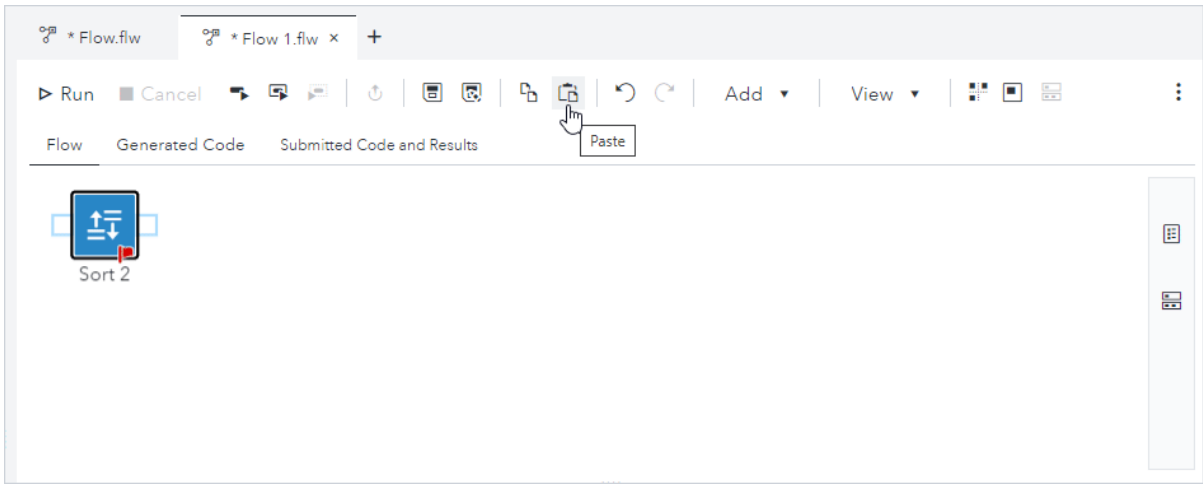
Cutting, Copying, and Pasting Nodes

You can cut, copy, and paste one or more nodes within a flow. You can also cut, copy, and paste nodes between flows. When you cut or copy nodes, the properties and values that are associated with the nodes are retained.

When you copy and paste an operational node within a flow, any connections to the input port of the original node are automatically created to the input port of the pasted node. You must manually re-create any connections from the output port of the pasted node.



When you copy and paste an operational node into a different flow, connections are not automatically created unless you have also selected and copied the connected nodes. The pasted node displays an error state until you add the required connections and update the node options as necessary.



To copy one or more nodes, select the nodes that you want to copy from the flow canvas and click .

To cut one or more nodes, select the nodes that you want to cut from the flow canvas. Right-click a selected node and select **Cut**.

To paste the nodes, open the flow into which you want to paste the nodes and click .

> **TIP** SAS Studio can auto-arrange the nodes in your flow by repositioning nodes, ports, and connections into a logical arrangement. You can modify any changes SAS Studio makes by manually updating the flow. To auto-arrange the nodes in the flow, click on the toolbar.

Connecting Nodes

Nodes are joined by connections to either the node itself or to a port on the node.

Depending on the node type, you can drag a node to the flow canvas, drop it on another node, and automatically create a connection between the nodes. The tip on your mouse pointer changes to indicate valid locations in which you can insert the node in the flow, connect the node to the input port of another node, or connect the node to the output port of another node.

For example, you can drag a Table node and connect it to either the input or output port of a Query node, or you can drag an external .csv file from the **Explorer** section of the navigation pane and connect it to the input port of an Import node.

> **TIP** SAS Studio can auto-arrange the nodes in your flow by repositioning nodes, ports, and connections into a logical arrangement. You can modify any changes SAS Studio makes by manually updating the flow. To auto-arrange the nodes in the flow, click on the toolbar.

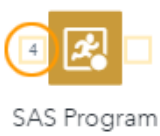
Expanding and Collapsing Node Ports

Some node types, such as SAS Program, Query, and Import, can contain one or more input and output ports. Ports represent either an input source or an output target for connecting data with nodes.

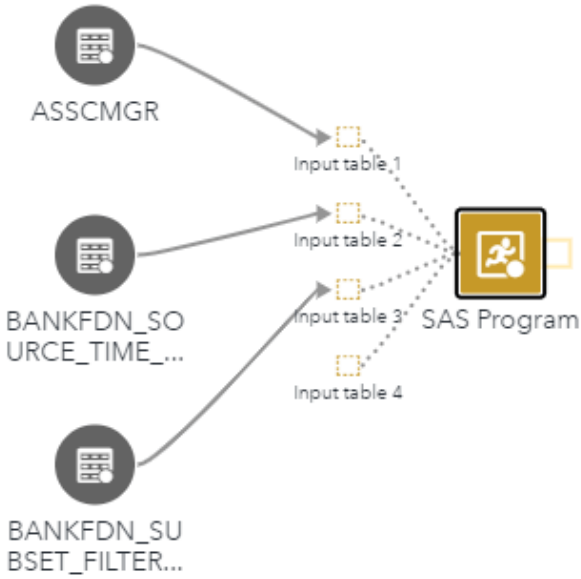
By default, ports are collapsed. To expand or collapse a port, right-click the port and select **Expand** or **Collapse**. You can expand or collapse all ports by clicking **View** on the toolbar and selecting **Expand all ports** or **Collapse all ports**.

You can assign source data to an input port by connecting another node to the input port. Depending on the node type, input can be either data or the output from another node.

SAS Studio automatically adds ports to a node as they are needed when you connect nodes. You can also manually add input and output ports to some node types by right-clicking the node and selecting **Add input port** or **Add output port**. When a node has more than two ports, they are displayed as a single port with the number of ports noted.



When a node has multiple input ports, the ports automatically expand when you create a connection.



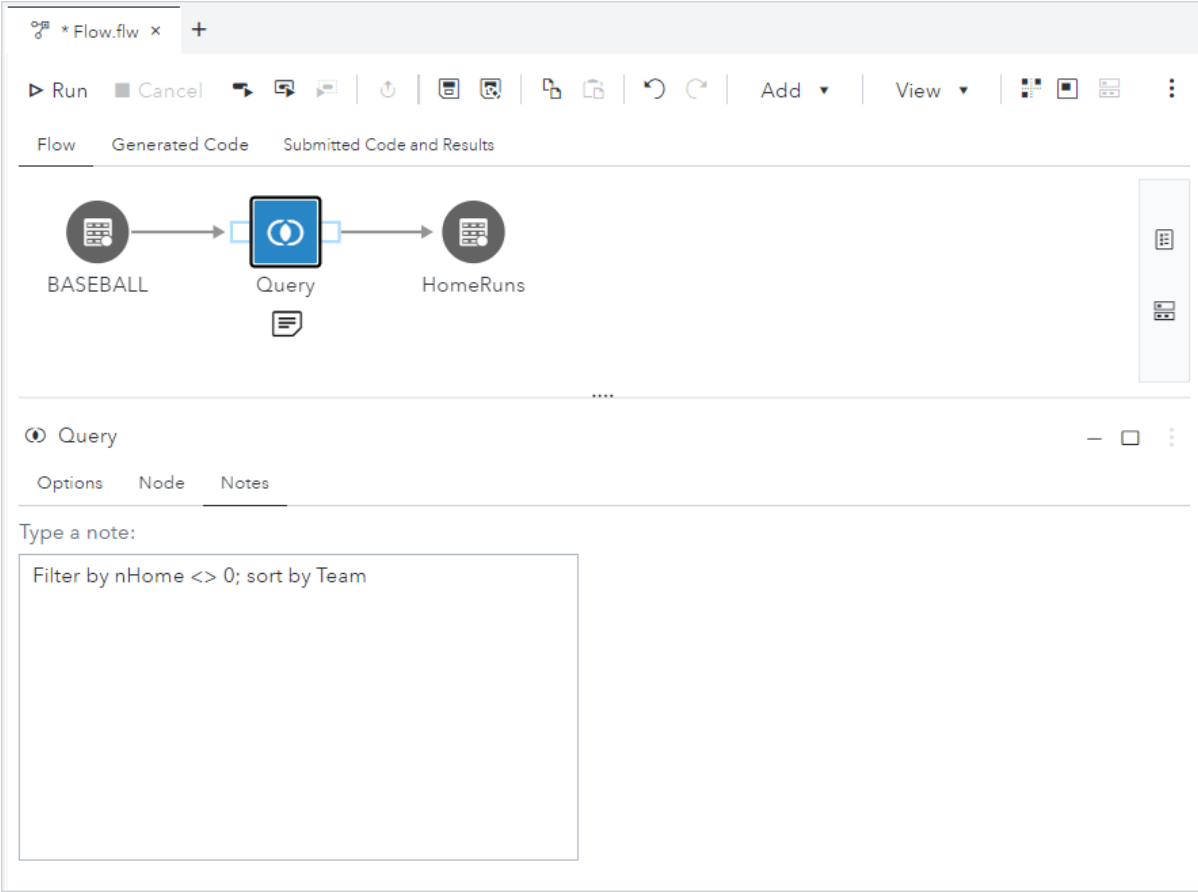
Note: When a node has multiple output ports, you must first expand the ports before creating a connection to another node.

By default, output ports represent tables in the Work library. To specify a location for the data, add a Table node to the flow and connect the node to the output port. An empty output port indicates that there is no data available in the port.

When a node has run successfully and data is available from the output node, the node is displayed as filled in.

Adding Notes to a Flow Node

You can add notes to a specific node in a flow by using the Note tab in the node details. When you add a note to a node, a note icon is displayed with the node on the flow canvas.



To add a note to a node:

1 Click the appropriate node on the flow canvas.

2 Click the **Notes** tab in the node details area and enter the note text.

You can edit a note in these ways:

- Click the **Notes** tab in the node details area and update the note text.
- Click the note icon on the flow canvas to expand the note text, and then double-click the note text box. Enter your changes in the note text box. Click anywhere on the flow canvas to save your changes.

To expand a node note, single-click the note icon.

To collapse a node note, click ^ in the upper right corner of the note.

To remove a note from a node, delete the text in the note.

2

Working with Data in a Flow

About the Table Node

Table nodes are used to reference SAS data sets in SAS libraries, CAS tables in CAS libraries, or DBMS tables in DBMS libraries. You can use table nodes as input and output for operational nodes.

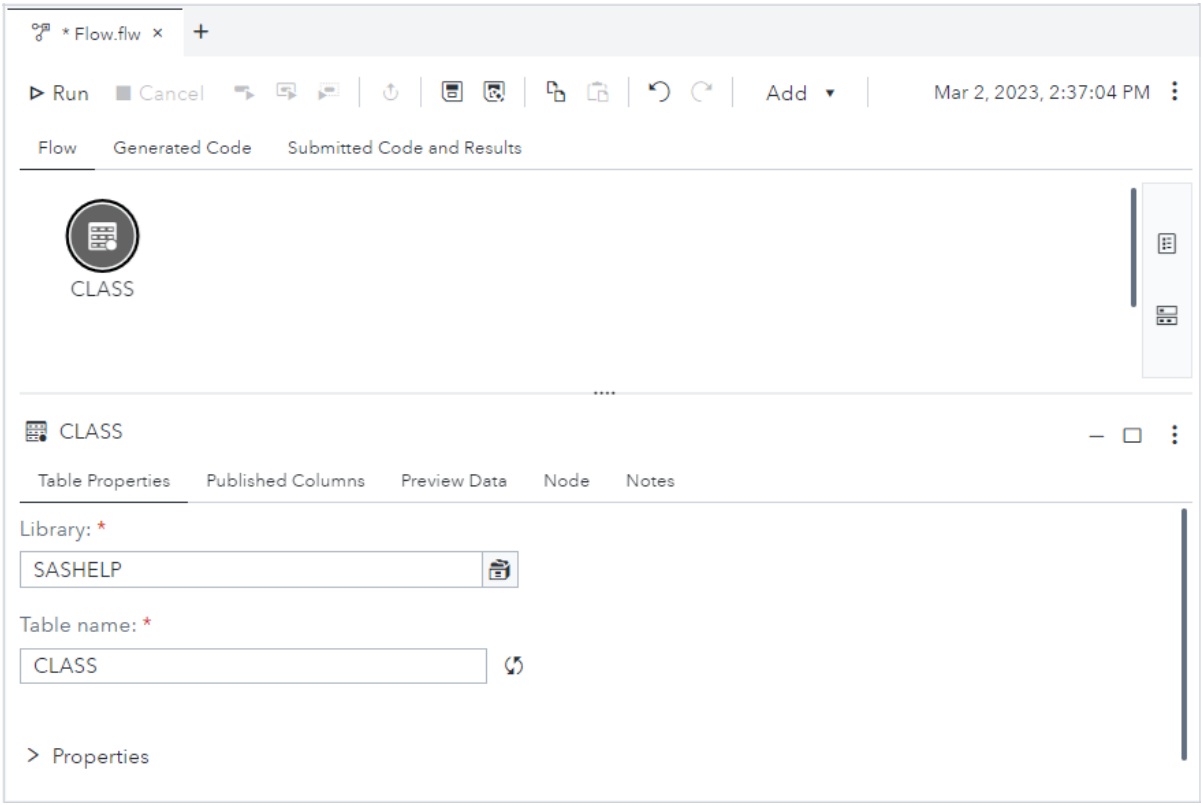
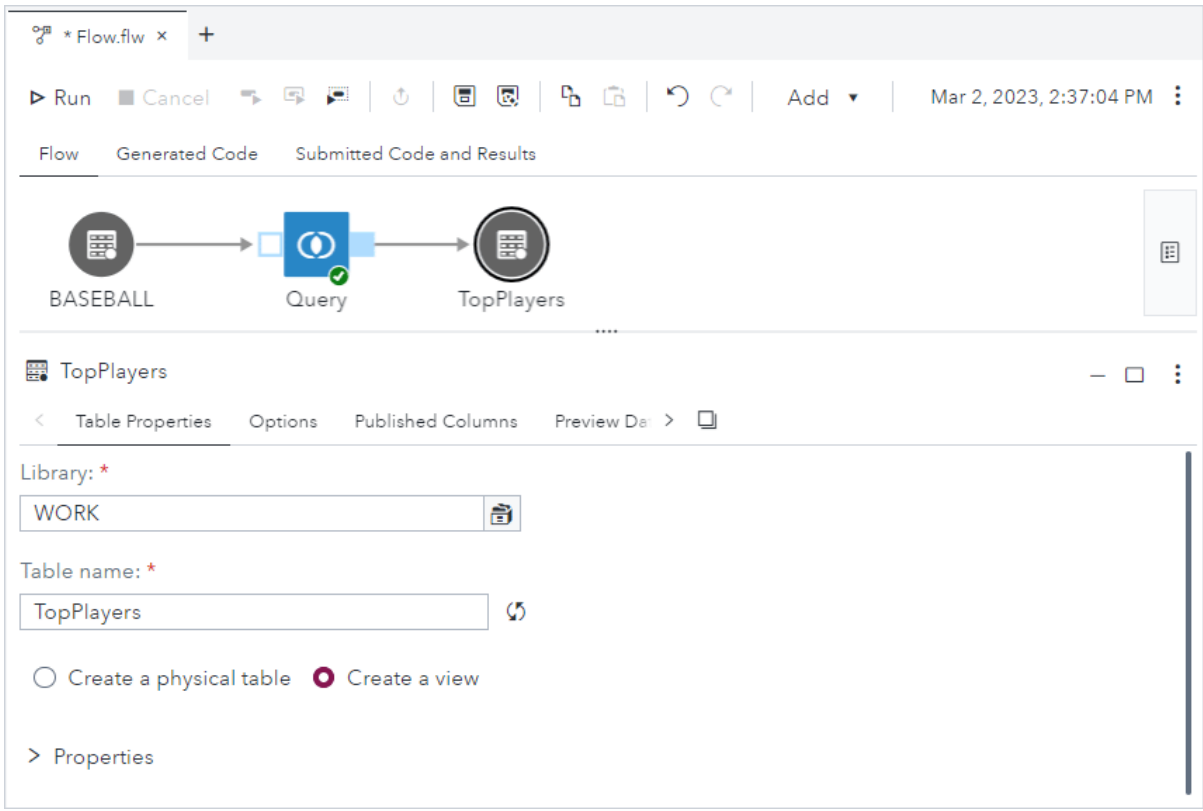


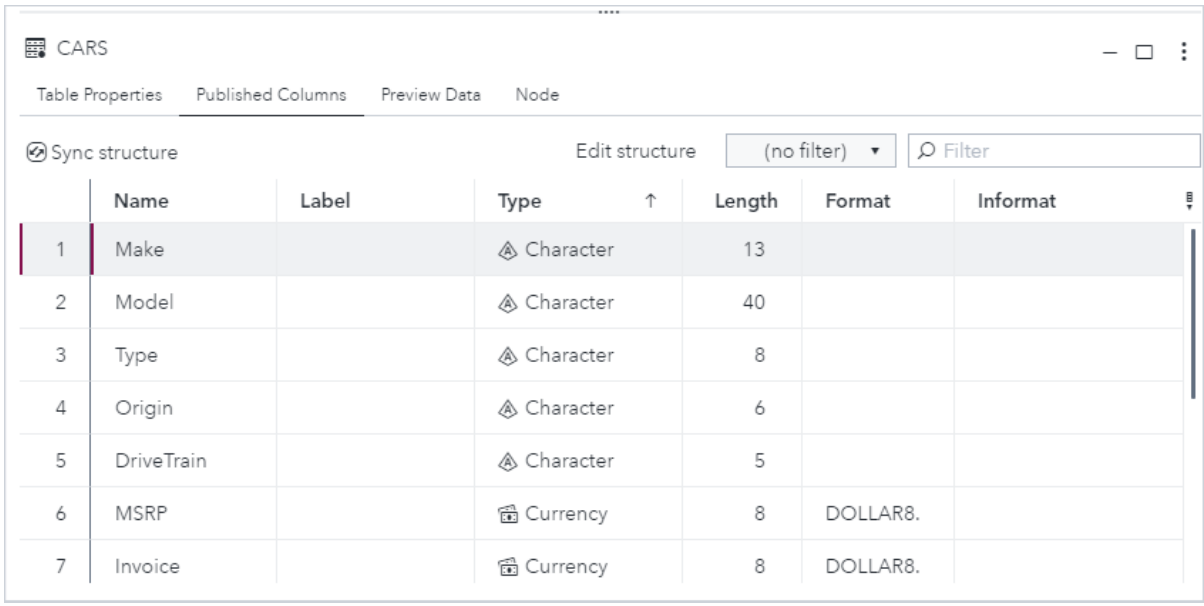
Table nodes can also reference data views. A view is a stored query that generates a result set when you access the view. Views do not store data. A view is a virtual table and can be used like a table.

Note: You can create a view by connecting a Table node to the output port of a Query node. The Query node must use PROC SQL to generate the results as a view. For more information, see “Query” on page 103.

To generate a view, click the Table node on the flow canvas, and then click the **Table Properties** tab. Select **Create a view**.



The column structure of the table or view is displayed on the Published Columns tab of the node details.



You can use the Published Columns tab for these tasks:

- To filter the columns on the tab by data type, click the filter drop-down list and select the filter criteria. You can also enter filter criteria for the Name and Label columns in the filter box.
- To add or delete columns or to change the properties of an existing column, click **Edit structure**. To exit edit mode and reset the column structure to the structure

of the table that is specified on the Table Properties tab, click **Edit structure** again. Any changes that you have made to the column structure are overwritten.

- To synchronize the column structure with the structure of the table that is specified on the Table Properties tab, click **Sync structure**. This option is not available when you are in edit mode.

TIP You can use the **Sync structure** and **Edit structure** options to copy the column structure of an existing table to a new table.

1. Click the table node to which you want to copy a new column structure, and then click the **Table Properties** tab.

   Note: Before you can copy the column structure of an existing table, you must specify a library and table name for the new table (for example, Work.NewTable).

2. Change the values of the **Library** and **Table name** fields to match the library and table names of the table whose structure you want to copy (for example, Sashelp.Class).
3. Click the **Published Columns** tab. Click **Sync structure** to synchronize the column structure of the new table with the existing table, and then click **Edit structure**. You can add, delete, or edit columns, if necessary.
4. Click the **Table Properties** tab again and change the values of the **Library** and **Table name** fields back to the values of the original table (for example, Work.NewTable).

You can view the data in the table or view on the Preview Data tab. The Preview Data tab interface is very similar to the interface for the stand-alone table viewer. For more information, see "About the Table Viewer" in *SAS Studio: User's Guide* and other topics in the "Working with Data" chapter.

Specifying Advanced Output Table Options

You can specify additional options to apply when an output table is created or when data is updated or inserted in the table. Often, the additional options are used to improve performance. The advanced table options can be applied to tables in these situations:

- Table node that is connected to the output port of an operational node – advanced options for a Table node can include options that are associated with creating a table. For example, you could use additional options to specify a label, specify how observations in the output table are compressed, or you could create single and composite indexes for a SAS output table.

Note: The advanced output table options are not available if the Table node is connected to the output port of a SAS Program, Python Program, Insert Rows, Execute Decisions, or custom step node.

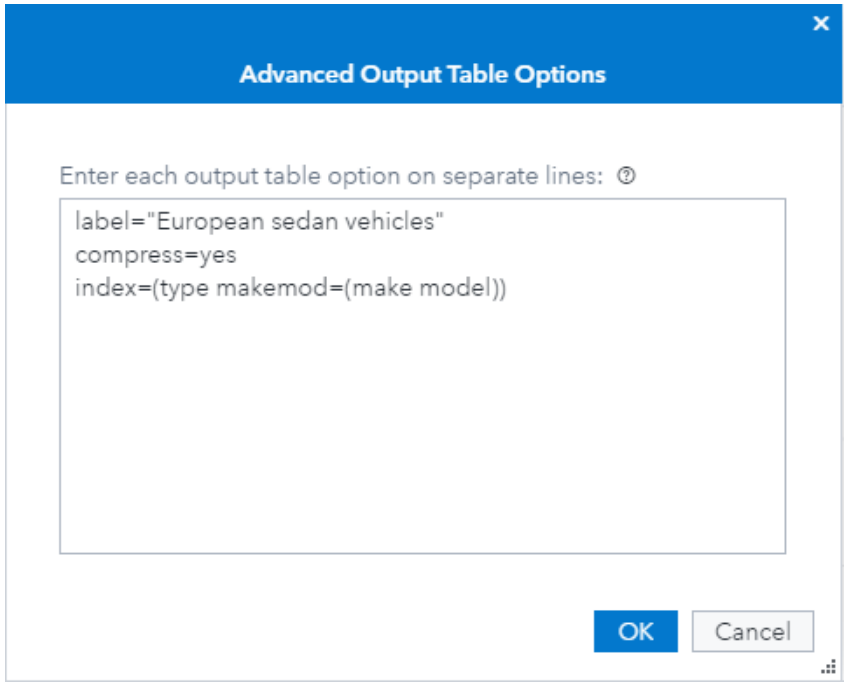
- Target table that is defined in the node details of an operational node – advanced options for the target table in an operational node can include options that are associated with the operations performed by the node on the table. For example, in a Load Table node, you could use options that are associated with inserting and updating data in a table, such as options to specify the number of rows to insert or to specify whether to use a DBMS-specific bulk-load facility.

To specify advanced table options:

1. If you are specifying options for a Table node, click the output table in the flow, and then click the **Table Properties** tab.

   If you are specifying options for the target table in an operational node, click the operational node and then click the **Options** tab.

2. Expand **Output Table Options**, and click **Advanced table options**. Enter each option on a separate line. This example shows options for a Table node.



The syntax for the advanced options can vary depending on the code that is generated for the step. In most cases, the output options should be data set options, but some steps require CASL table parameters or FedSQL table options. You must include any quotation marks that are required for string values. You can find more information about options here:

- SAS data set options
- CASL table parameters
- FedSQL table options

Note: This is an option for advanced users. It is recommended that you avoid using any options that might change the column metadata, such as DROP=, KEEP=, or

RENAME=. If you use options that change the column metadata, you can generate syntactically valid code that can cause errors in downstream nodes. If you need to subset columns in the table or change column names or labels, you should use an additional step such as Manage Columns.

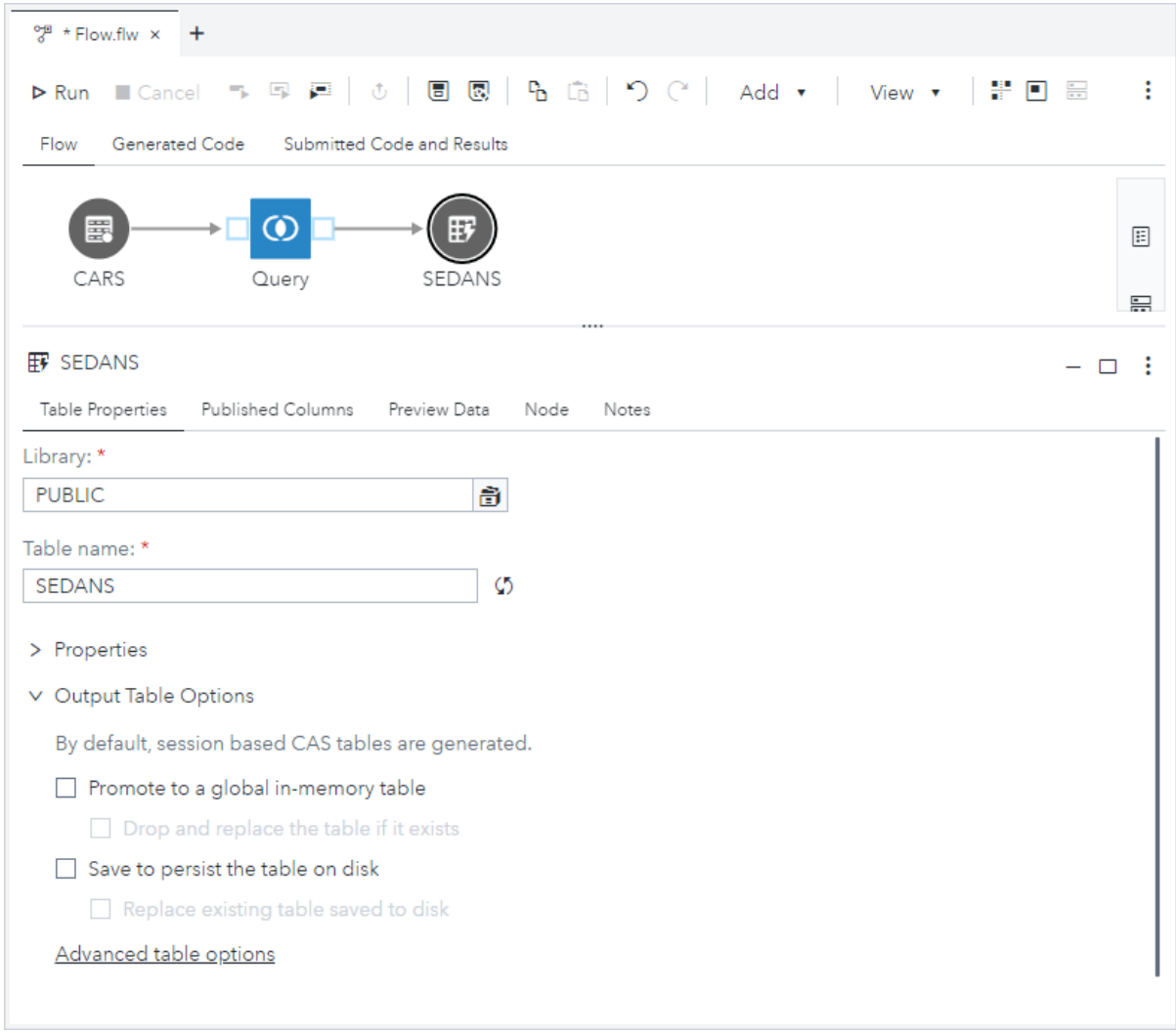
Setting Options for CAS Output Data

If you are running an operational node that creates CAS output data, you can specify whether the table is a session-scope table or a global-scope table, and you can save the table to the CAS server. You can also specify additional options to apply when the table is created or updated.

Note: The output table options are not available if the table node is connected to the output port of a SAS Program, Python Program, Insert Rows, or custom step node.

To specify CAS output table options:

1 Click the output table in the flow, and then click the **Table Properties** tab. Expand **Output Table Options**.



2 Select from the following options:

- **Promote to a global in-memory table** - creates a global-scope table that can be viewed by other sessions. Global-scope tables persist after the current CAS session is terminated. If you want to automatically replace an existing global-scope table of the same name, select **Drop and replace the table if it exists**. If you do not create a global-scope table, a session-scope table is created. A session-scope table can be viewed only by your current CAS session and is dropped when the session ends. Session-scope tables are created by default.
- **Save to persist the table on disk** - saves the table to the CAS server as a SASHDAT file. If you want to automatically replace an existing table of the same name, select **Replace existing table saved to disk**.

3 To specify additional options to apply when the table is created or updated, click **Advanced table options** and enter each option on a separate line. For more information, see “Specifying Advanced Output Table Options” on page 18.

Exporting Data to an External File

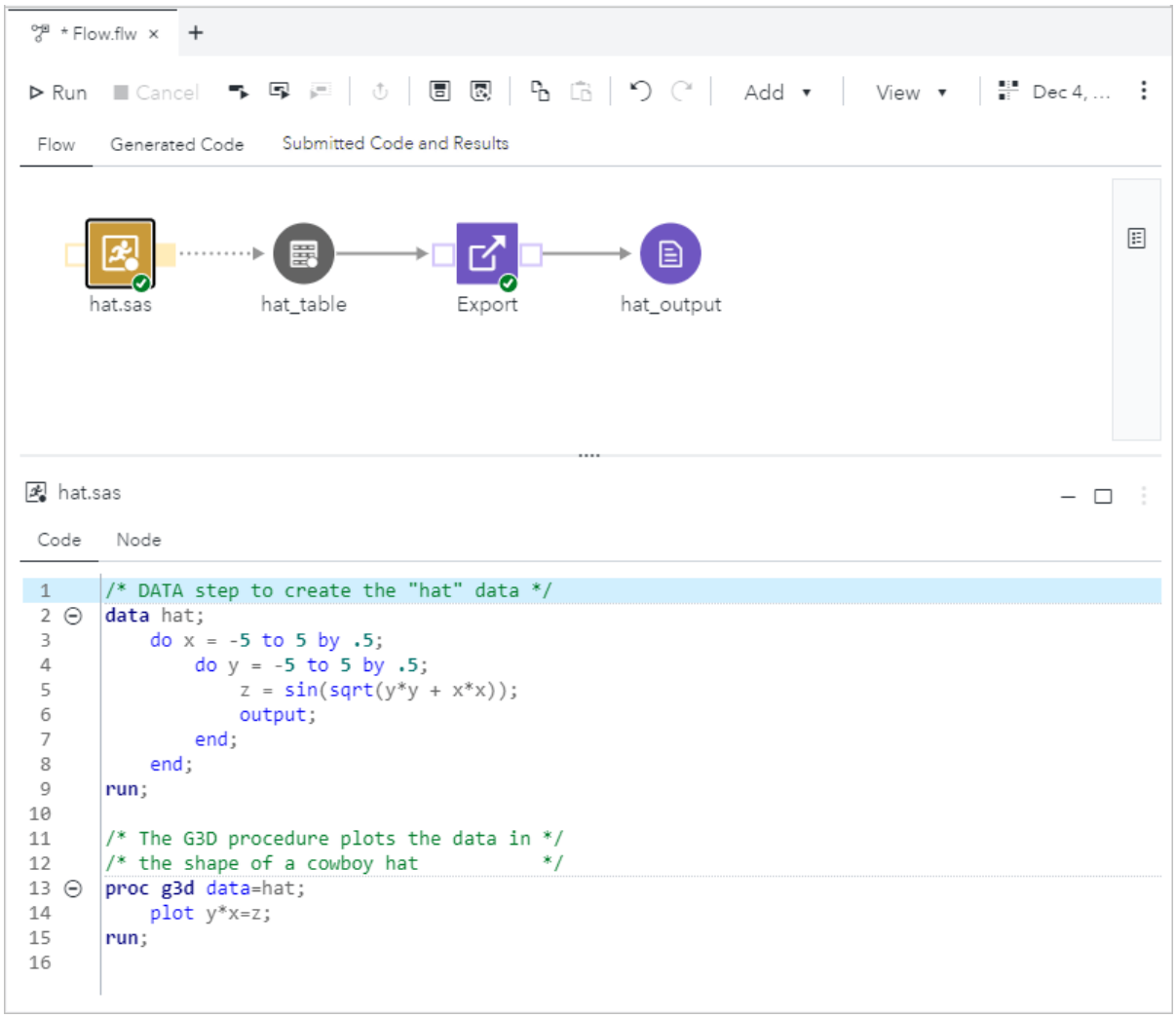
To export data in a flow:

1. Click the **Steps** section of the navigation pane.
2. Expand the **Data** folder and double-click the **Export** step to add an Export node to the flow.
3. Connect the input port of the Export node to a data source, such as a Table node or another node that creates an output table, such as a Query node or an Import node. For more information, see “Connecting Nodes” on page 12.
4. Connect the output port of the Export node to a File node. Use the File node to specify options for the output file, including the name and location of the file. The file extension determines the format of the exported file. For more information, see “Adding an External File to a Flow” on page 23.

   > **TIP** You can export data to an unconnected File node in your flow by right-clicking the **File** node and selecting **Add an export**. You must also connect a data source to the input port of the Export node.

5. Click **Run**.

The following example shows the `hat.sas` SAS Program node that creates an output table in the `hat_table` Table node. The `hat_table` node is connected to the input port of the Export node and is exported as a text file to the `hat_output` File node.



Adding an External File to a Flow

File nodes are used to point to external files. You can use a File node as input to a step such as the Import step in order to convert a file to a table in a SAS library. File nodes can also be used as output to a step such as the Export step.

File nodes can point to these file types:

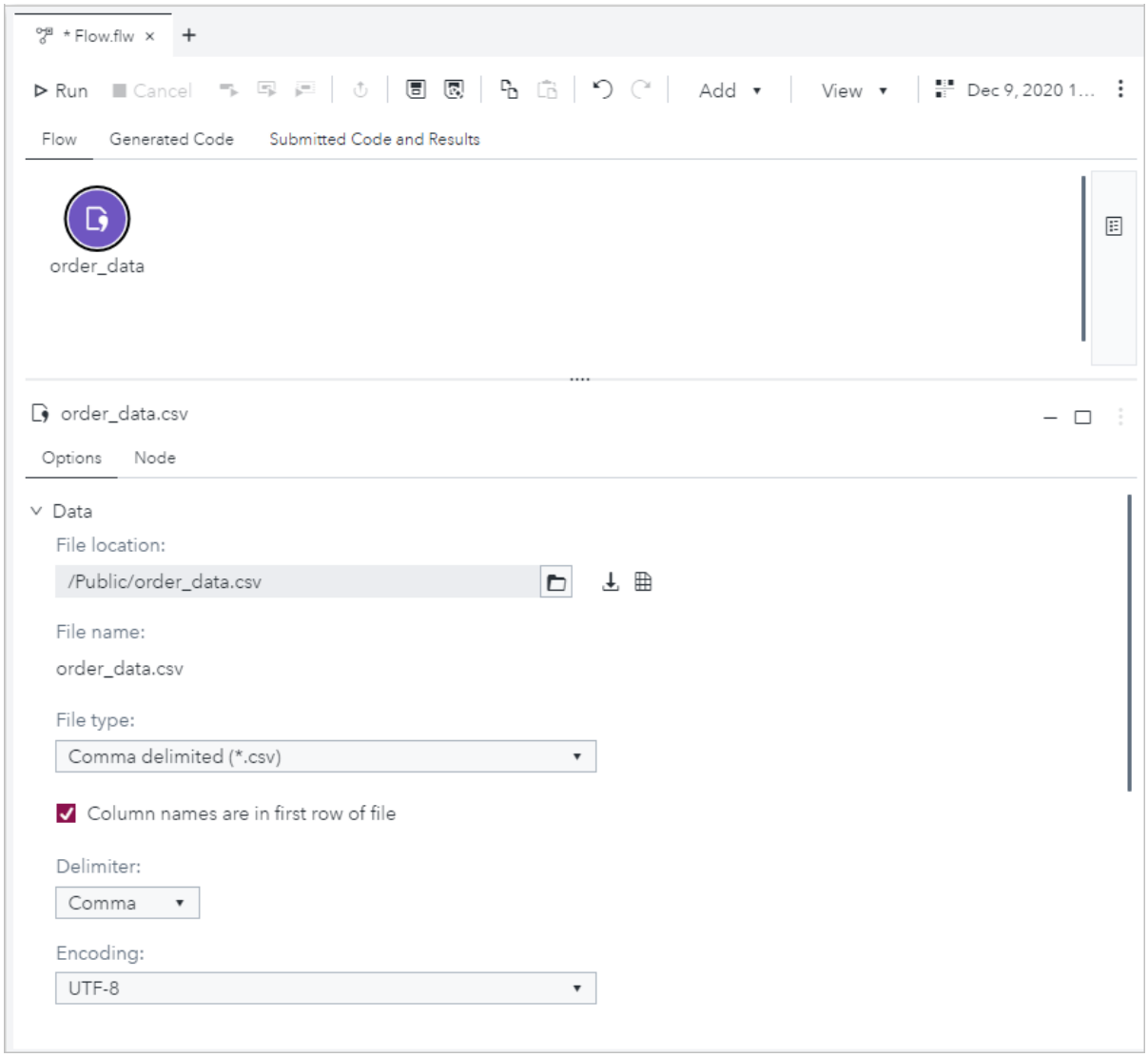
- text (*.txt).
- tab-delimited (*.tab, *.tsv).
- delimited (*.dlm).
- comma-delimited (*.csv) .
- Microsoft Excel (*.xlsx). To import XLSX files, you must license and install SAS/ACCESS to PC Files.
- Microsoft Excel 97–2003 (*.xls).

The file that is referenced by a File node must be located in SAS Content or on the file system for the current SAS Compute Server.

To add an external file to the flow:

1. Click the **Explorer** section of the navigation pane and navigate to the appropriate folder.
2. Right-click the file that you want to add, and then select **Add to flow**. You can also drag the file into the flow.

The File node options include information about the properties of the file. Depending on the file type, the options also enable you to download and view the file and update some of the data that is associated with the node.



- To download the file, click ⭳. Depending on your browser settings, the file might open in a viewer.
- To view the raw data, click ▦.

Note: This option is not available for Microsoft Excel files.

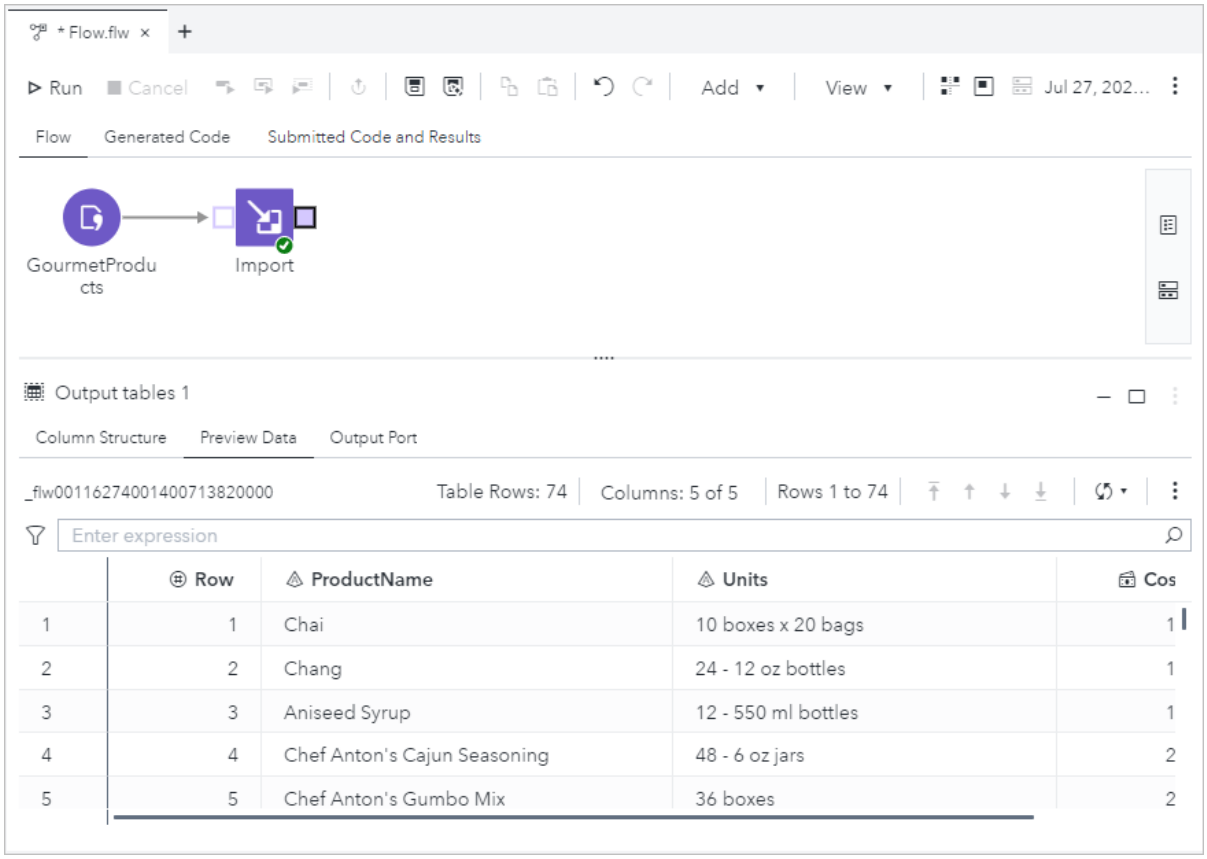
- You can update the following fields:
  - **File location**.
  - **File type**.
  - **Column names are in first row of file** - indicates whether the data includes a header row.
  - **Delimiter**.
  - **Encoding**.

Importing Data from an External File

About the Import Step

The information in external files, such as delimited files and Microsoft Excel spreadsheets, must be converted to a SAS table in order to be queried, filtered, or analyzed by SAS. The Import node enables you to convert the data in external delimited, text, fixed width, and Microsoft Excel files to SAS tables.

The next figure shows a flow that converts the information in a delimited file (GourmetProducts) and writes the result to the output port of the Import node. By default, output for the Import node is written to a table in the Work library. Import node output can be connected to any flow node that accepts SAS data as an input, such as a Table node, a Query node, or a SAS Program node.



There are three main steps to import a file:

1. Analyze the data, and then verify that SAS Studio is reading the data correctly. If necessary, update the options that are used to analyze the data or load the column structure from an external file.
2. Verify that the column properties for the output data are correct. If necessary, update the options that are used to analyze the data or load the column structure from an external file.
3. Run the flow to import the data.

Node Connection Requirements

In order to run the Import step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ File node	No connection is required. **Note:** Import node output can be connected to any flow node that accepts SAS data as an input, such as a Table node, a Query node, or a SAS Program node. By default, the output data is written to a temporary table in the Work library. You can specify the library and name of the

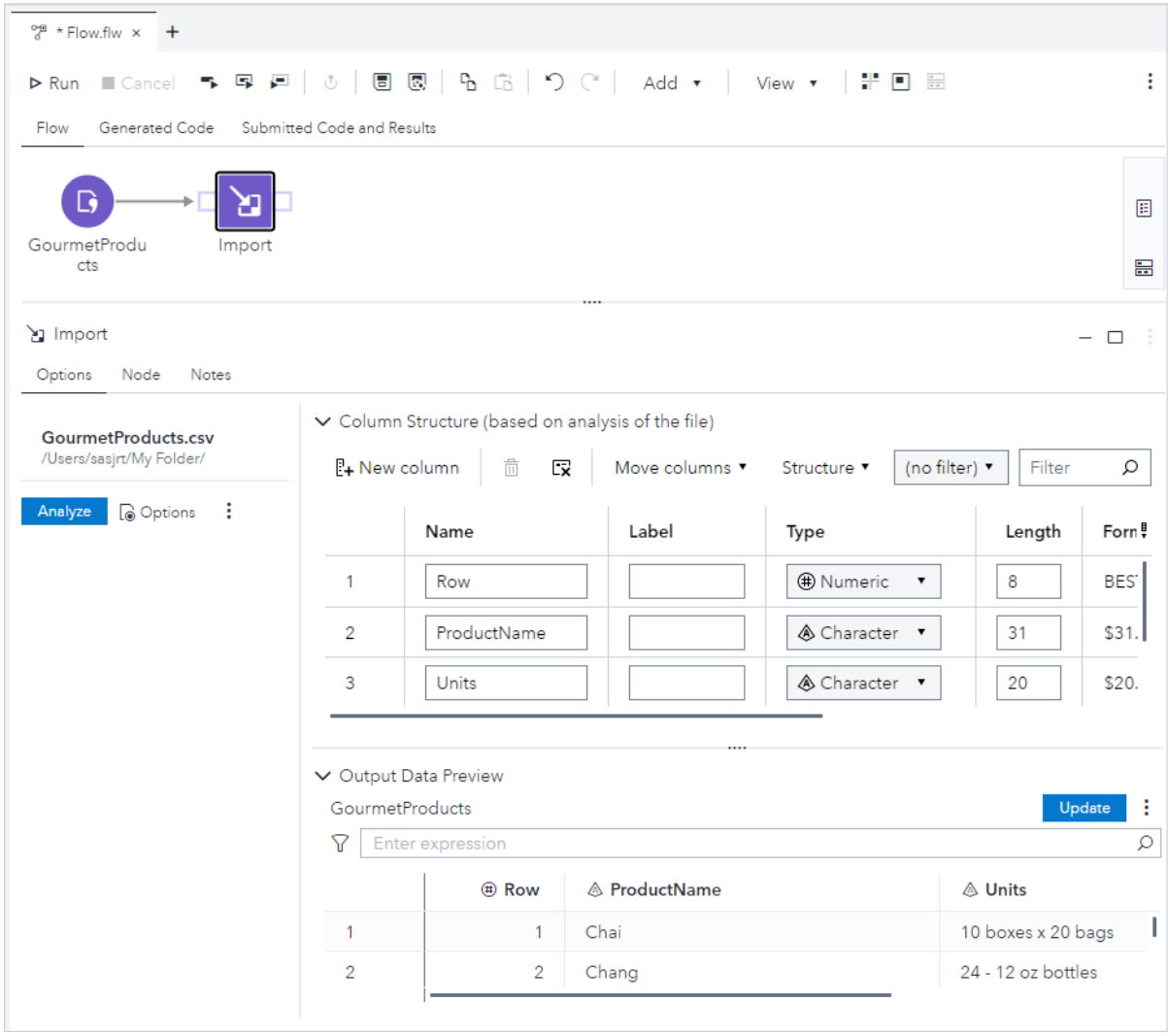
Input Port	Output Port
	output table by connecting the output port to a Table node. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Import an External File

Step 1: Analyze the Data

To analyze the data in the file that you want to import:

1. Right-click the File node in the flow and select **Add an import**. An Import node is added to the flow, and the File node is connected to the input port of the Import node.
2. Click the Import node, and then click the **Options** tab. Click **Analyze** to identify the structure of the external file. The following figure shows the results from analyzing an external delimited file:



3 If you are importing a Microsoft Excel spreadsheet with multiple worksheets, select the worksheets that you want to import. By default, all of the worksheets are selected.

4 If you are importing a fixed-width file, the data is loaded into a single column. Use the Column Structure area to add the other columns to the output table and specify column properties. Use the Start Position and End Position columns to specify where in the data each column starts and ends.

Note: If you change the file type of the file node that you are importing, any changes that you made to the column structure are not saved.

Step 2: Review Column Structure for Output File and Adjust As Needed

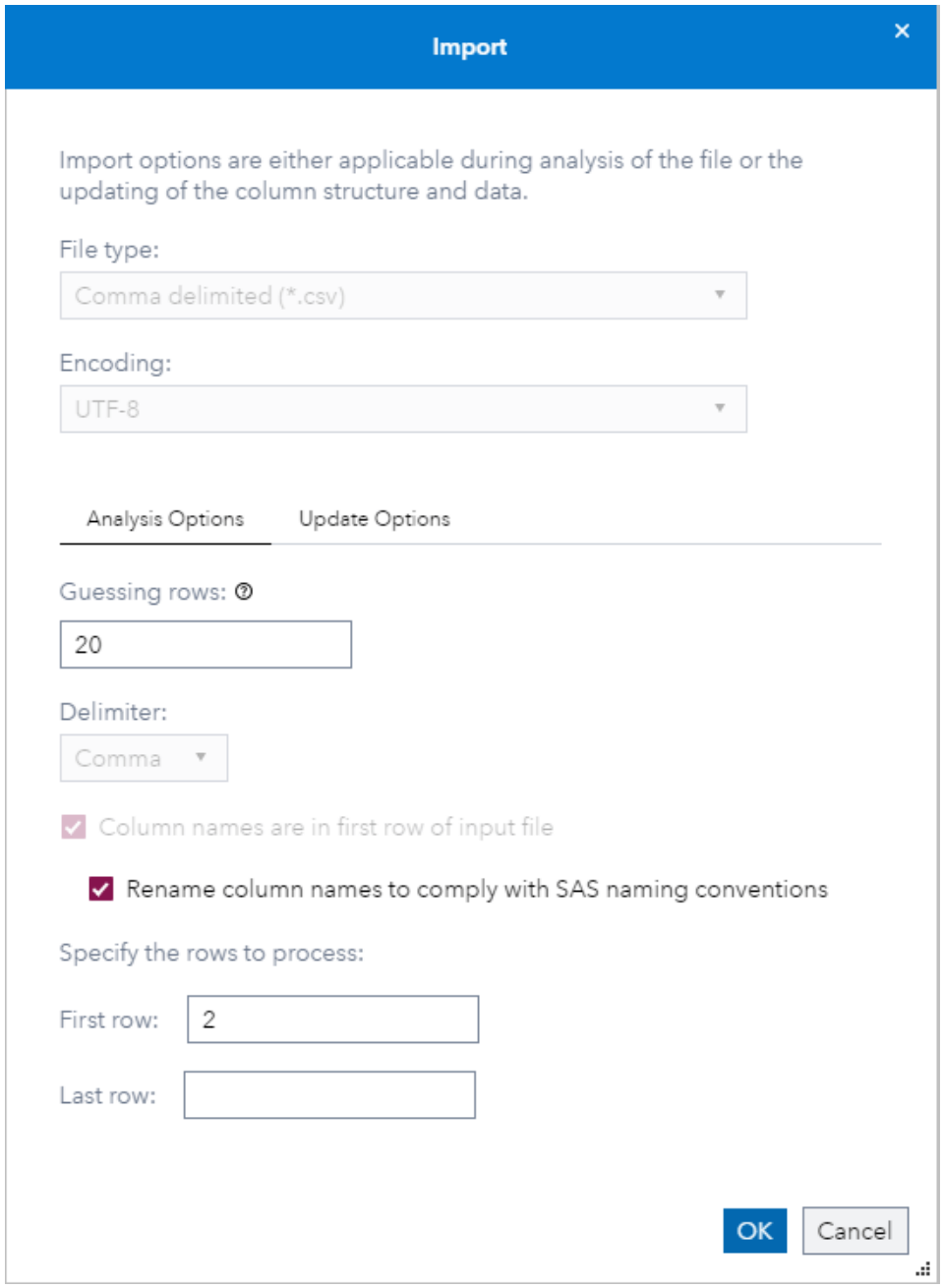
Review the column structure for the output file. To make adjustments to the column structure, update the options that are used to analyze the data or load the column structure from an external file.

- Update the Analysis Options

  To update the options that are used to analyze the column structure of the file that you want to import, click **Options**, and then click the **Analysis Options** tab.

  Note: If **Options** is not visible on the Import Options toolbar, click ⋮ and select **Options**.

  The following figure shows the Analysis Options tab for a comma-delimited file:



The data options depend on the file type.

Table 2.1 *Data Options for Each File Type*

File Type	Available Data Option
Delimited file	■ **Guessing rows** - indicates the number of rows to scan in the input file to determine the appropriate data type, informat, format, and length of columns. Setting this option to higher values could adversely affect performance or cause SAS Studio to time out. It is recommended that you set this option to a low value initially to avoid potential performance or time-out issues. If guessed column properties are not as expected, the option

File Type	Available Data Option
	value can be increased and the file re-scanned. The limit is 2,147,483,647 rows. ■ **Rename column names to comply with SAS naming conventions** - updates column names to comply with SAS variable name rules. ■ **Specify the rows to process** - specifies the first and last rows to process in a delimited text file. To update the **Column names are in the first row of input file** or **Delimiter** options, click the File node in the flow and update the options on the Options tab.
Fixed-width file	**Specify the rows to process** - specifies the first and last rows to process in a delimited text file.
Microsoft Excel file	■ **Rename column names to comply with SAS naming conventions** - updates column names to comply with SAS variable name rules. ■ **Limit the range of imported columns and rows** - enables you to specify a range of cells to import from the Microsoft Excel file.

After you update the Analysis options, click **OK**, and then click **Analyze** again to apply the changes to the column structure.

- Load Column Structure from an External File

  You can use a column structure metadata file to load the structure of the file that you are importing. Using a column structure metadata file can increase the reliability and speed of your import process. You can use the column structure metadata file to make necessary edits to the column structure rather than manually editing each row of your data files.

  The column structure metadata file must meet these requirements:

  - All values must be comma-delimited.
  - The first line must include the column properties and the order in which they appear. Column property names are case-sensitive.
  - Each column must be listed on a separate line with its properties.
  - The file can be saved as a comma-separated values (*.csv) or a text (*.txt) file.

  Note: You can add comments to the column structure metadata file by entering # at the start of the line.

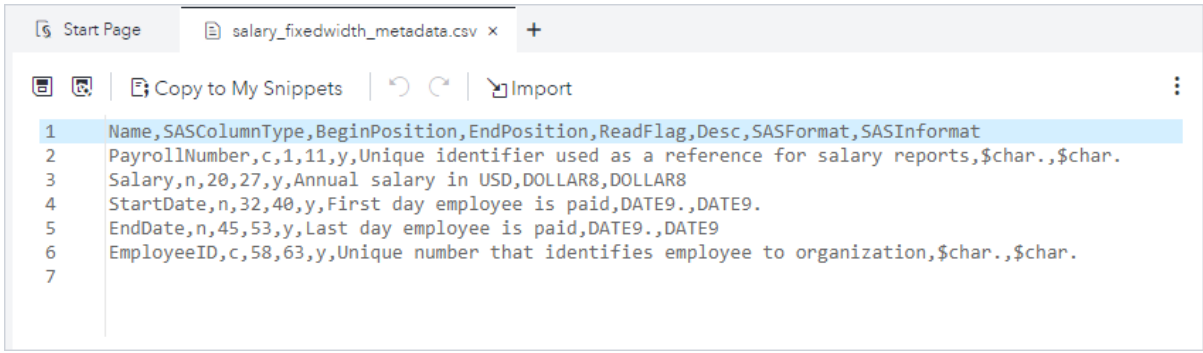
  To load the column structure from an external file, select **Structure** ⇨ **Load structure**. Select the column structure metadata file that you want to use and click **OK**. The new column structure appears in the Column Structure area.

  Note: You can generate a column structure metadata file from the column structure by selecting **Structure** ⇨ **Generate structure**. You can save the file as

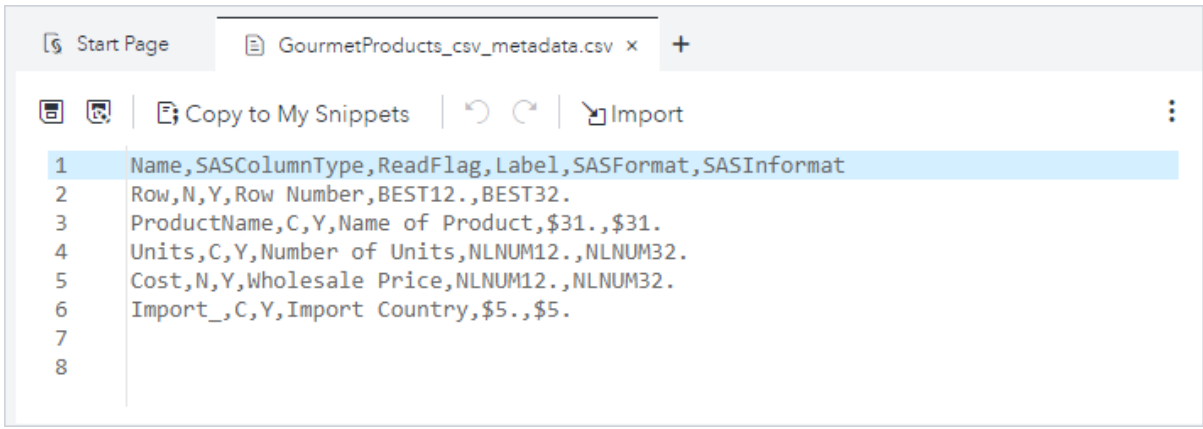
a comma-delimited (*.csv) file or a text (*.txt) file. The column structure metadata file is always saved with UTF-8 encoding.

Here is an example of a column structure metadata file for a fixed-width file:

- Line 1 of the salary_fixedwidth_metadata.csv file defines the column properties and the order in which they appear:
  `Name,SASColumnType,BeginPosition,EndPosition,ReadFlag,Desc,SASFormat,SASInformat`
- Lines 2–6 list the properties for these five columns: PayrollNumber, Salary, StartDate, EndDate, and EmployeeID.



Here is an example of a column structure metadata file for a CSV file. Note that the first line does not include the BeginPosition and EndPosition columns because the imported file is not a fixed-width file. In addition, the metadata in this example includes a column label instead of a description.



When you create a column structure metadata, you can include all of these column properties or a subset of these properties. The properties can be listed in any order depending on what is required for the data that you want to import.

Name of Metadata Column	Required?	Description
Name	Yes.	Name of the column in the data file. In the example, PayrollNumber is the first column name.
SASColumnType	No.	SAS column type. Columns can be

Name of Metadata Column	Required?	Description
		`character` (c) or `numeric` (n). If you do not specify a value, the type defaults to `character`.
BeginPosition	Yes, if you are importing a fixed-width file. This value is ignored if you are importing a delimited file.	The first position in a record where the data for this column begins. If you specify a non-numeric value for this position, the property is not set in the column metadata and the column length defaults to `8` for both numeric and character values.
EndPosition	Yes, if you are importing a fixed-width file. This value is ignored if you are importing a delimited file.	The last position in a record where the data for this column ends. If you specify a non-numeric value for this position, the property is not set in the column metadata and the column length defaults to `8` for both numeric and character values.
ReadFlag	No.	Specifies whether to read the records for this column. Valid values are `y` or `n`. If you specify `n`, the column is not read. If you do not specify a value for this property, the record is read.
Desc or Label	No.	Specifies either a description of the column or a label for the column.
SASFormat	No.	The SAS format that is used to display the data in the imported table.
SASInformat	No.	The SAS informat that is used to read the data in from the fixed-width file.

Step 3: Update the Column Properties

In the Column Structure area, you can update the properties for each column, including the column name, label, data type, length, format, and informat. You can also add, delete, and reorder columns.

- If you are importing a fixed-width file, click **New column** to add columns to the output table. Use the **Start Position** and **End Position** columns to specify where in the data each column starts and ends.
- You cannot change the column information for Microsoft Excel files.

After you make changes to the column properties, click **Update** to apply the changes to the Output Data Preview window.

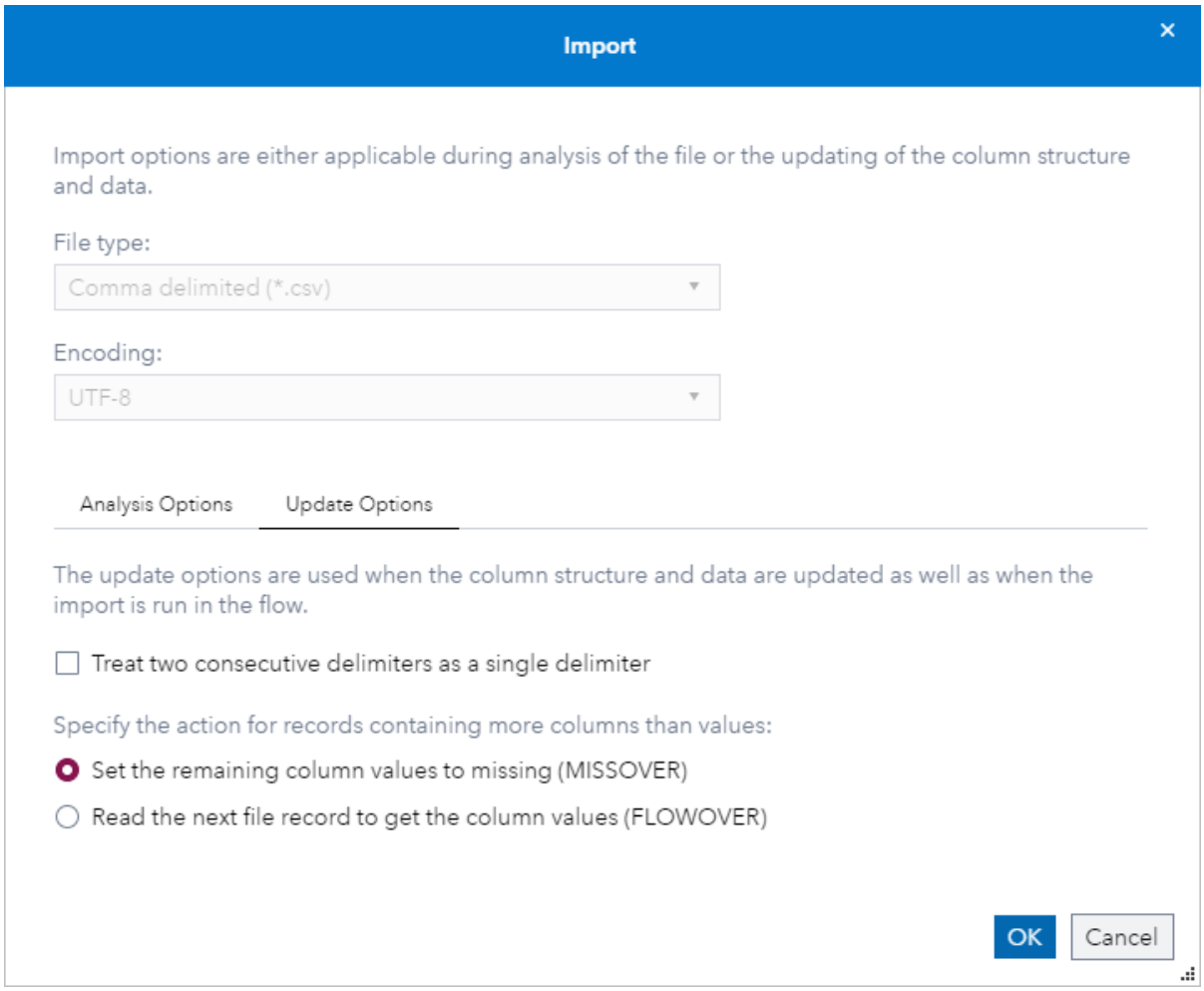
Note: If you click **Analyze** again, any changes you have made to the column properties are overwritten.

Step 4: Update the Options for the Output Table

If necessary, you can update the options that are used when the output table is generated by clicking **Options**, and then clicking the **Update Options** tab. This functionality is not available if you are importing a Microsoft Excel file.

Note: If **Options** is not visible on the Import Options toolbar, click ⋮ and select **Options**.

The following figure shows the Update Options tab for a comma-delimited file:



The data options depend on the file type.

Table 2.2 *Data Options for Each File Type*

File Type	Available Data Option
Delimited File	■ **Treat two consecutive delimiters as a single delimiter** - specifies that two consecutive delimiters should be treated as a single delimiter. If you do not select this option, the consecutive delimiters are treated as a missing value. ■ Select the option that you want to use for rows that have more columns than values: □ **Set the remaining column values to missing (MISSOVER)** - specifies that columns without any values are treated as missing values. □ **Read the next file record to get the column values (FLOWOVER)** - reads the next row of data when there are columns in the previous row without values.
Fixed-width File	■ **Pad short lines with blanks up to the specified record length** - adds blanks to the ends of lines that are shorter than the specified record length.

File Type	Available Data Option
	■ Select the option that you want to use for rows that have more columns than values: □ **Read the next file record to get the column values (FLOWOVER)** - reads the next row of data when there are columns in the previous row without values.

After you make changes to the Update options, click **OK**. Click **Update** again to apply the changes to the Output Data Preview window. The output table is not actually generated until you run the flow.

Step 5: Run the Flow and Import the Data

To import the data and generate the output table, click ▷ Run.

Adding a Table from a SAS Library to a Flow

You can use Table nodes to add SAS tables to your flow. Table nodes enable you to specify the library and table name of a table in the flow and can be used to connect to the input and output ports of operational nodes.

If you have Read access to a table in a SAS library, you can specify the table in a Table node. You can use the Table node as the input for an operational node in the flow, such as a Query node or a SAS Program node.

If you have Write access to a SAS library, you can use a Table node to create or update a table in that library. The table can be an existing table or a table that is created when a flow runs. You can then use the Table node as the output for an operational node in the flow, such as an Import node, a Query node, or a SAS Program node. By default, output ports represent tables in the Work library. To specify a name and location for the data, add a Table node to the flow and connect the node to the output port.

To add a Table node to a flow:

1 Click the **Steps** section of the navigation pane.

2 Expand the **Data** folder and double-click the **Table** step.

> **TIP** You can quickly add a Table node to the flow and connect the node to the output data source of an operational node by right-clicking the output port of the operational node and selecting **Add a table**. The operational nodes include nodes such as Import, Query, and SAS Program. File nodes and Table nodes are not considered operational nodes.

3. In the flow, click the **Table** node and use the **Table Properties** tab to specify a library and table.
4. Use the **Published Columns** tab to edit and add columns. To edit and add columns to the table, click **Edit structure**. To copy the column definitions from an existing input table, click **Sync structure**.

   Note: When you click **Sync structure**, you are prompted to replace your existing column structure with the column structure of the table that is specified on the Table Properties tab. Any changes you have made to the column structure are overwritten.

To add an existing table to a flow:

- From the **Libraries** section of the navigation pane, drag one or more tables to the flow canvas. A Table node is automatically created for each table.

3

Creating a Subflow

About Subflows

You can create a subflow by adding a saved flow to your flow. A subflow can be useful if you have a group of nodes that you want to use in multiple locations. You can also use a subflow to reduce the complexity of a flow.

Note: When you create a subflow, note these reminders:

- Only flows that are saved in a SAS Content folder can be used as subflows.
- You cannot run a flow that contains itself as a subflow.
- Subflows cannot be connected to other nodes in the flow.
- The subflow feature is available only if your site licenses SAS Studio Analyst.

Adding a Saved Flow to Another Flow

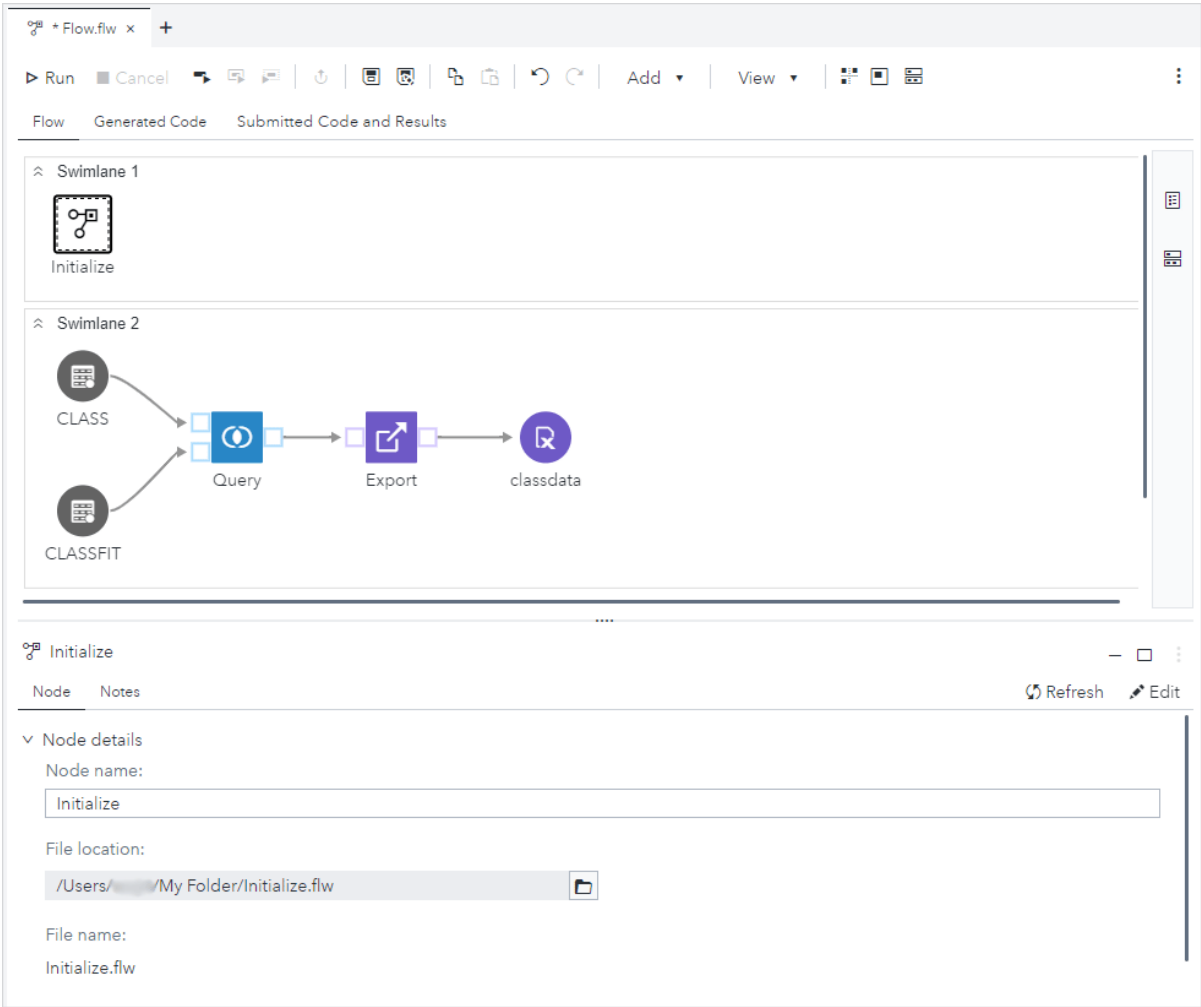
You can add saved flows from a SAS Content folder to another flow by using the **Explorer** section of the navigation pane.

To add a saved flow to another flow:

- From the **Explorer** section of the navigation pane, drag the flow file (*.flw) to the flow canvas. You can also right-click a flow file and select **Add to flow**.

Note: You must have a flow tab active in the work area in order to use the **Add to flow** option. If you do not have an open flow tab, you are prompted to create a flow.

TIP You can use swimlanes to control the order in which the subflow is run in your flow. For more information, see "Controlling the Submission Order of a Flow" on page 296.



Editing a Subflow

You cannot make changes to a subflow from within another flow.

To edit a subflow:

1 You can open a subflow for editing in these ways:

- Click the subflow node on the flow canvas, and then click **Edit** in the node details.
- Double-click the subflow node on the flow canvas.
- Right-click the subflow node on the flow canvas and select **Open and edit**.

The subflow opens in a separate tab.

2 Edit the flow and save your changes.

Note: If you edit the subflow from within the same SAS Studio session, SAS Studio automatically refreshes the subflow content in your flow. If the subflow is edited in another SAS Studio session, you must click **Refresh** in the node details to update the subflow content in your flow.

Note: You can also edit the subflow by opening it from the **Explorer** section of the navigation pane and saving your changes.

4

Developing Code in a Flow

SAS Program Step: Writing SAS Code in a Flow

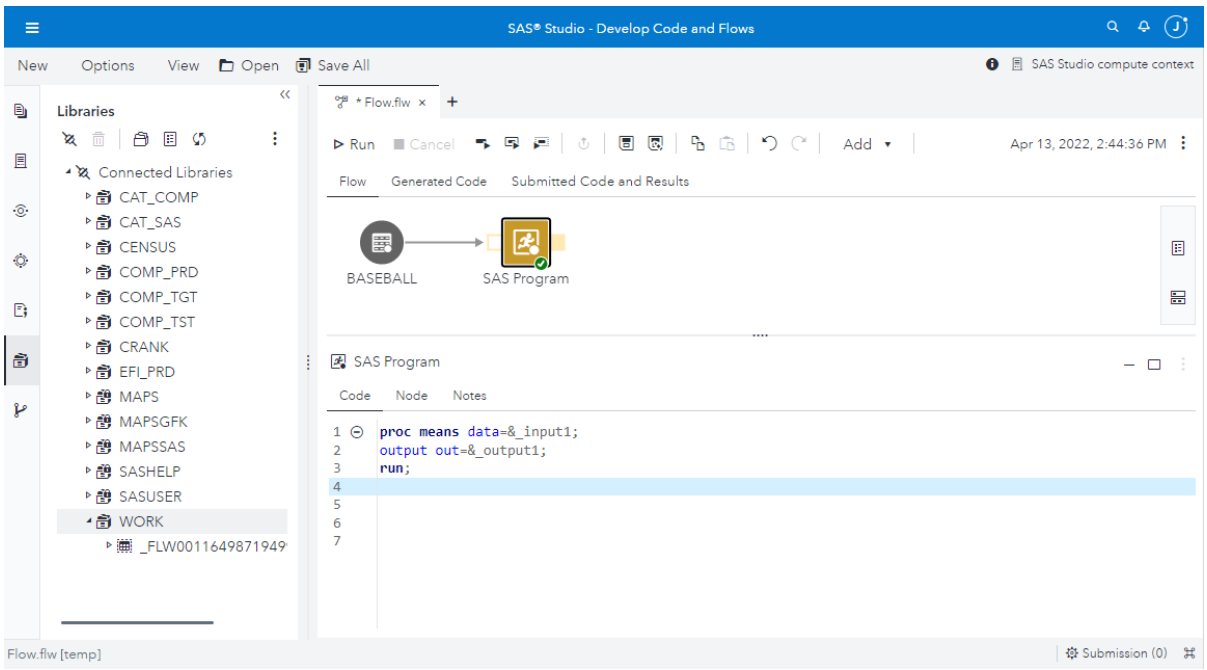
About the SAS Program Step

You can use the SAS Program step to include any type of SAS program that is required in a flow. The code that is associated with the SAS Program node is displayed in the node details on the Code tab. The Code tab includes the same

color-coded, syntax-checking editor that you use to write stand-alone SAS programs. For more information, see "About the SAS Code Editor" in *SAS Studio: User's Guide*.

In a SAS Program node, your SAS program can either be embedded in the flow or reference an external program file. Embedded programs can be accessed only from within the flow; externally referenced programs are saved program files. For more information, see "Understanding Embedded and Externally Referenced Programs" on page 49.

By default, output data from a SAS Program node is written to a table in the Work library. The next example shows a flow in which a SAS Program node runs the MEANS procedure against the BASEBALL table. The results are written to a table in the Work library. The table name is automatically generated by SAS Studio because the output port is not connected to a Table node. The example uses the `&_input1` and `&_output1` macro variables to reference the names of the tables that are associated with the input and output ports of the SAS Program node. For more information, see "Using Macro Variables to Reference Input and Output Ports" on page 53.



Node Connection Requirements

No node connections are required to run the SAS Program step. You can connect the SAS Program input and output ports to Table nodes or operational nodes that create an output table, such as a Query node or an Import node, and then use macro variables to reference the tables in your program. For more information, see "Using Macro Variables to Reference Input and Output Ports" on page 53.

Note: By default, any output data is written to temporary tables in the Work library. You can specify the library and name of the output table by connecting the output

port to a Table node. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

SAS Program: Step-by-Step Instructions

The SAS Program step requires no node connections.

1 In the **Steps** section of the navigation pane, expand the **Develop** folder and double-click **SAS Program**.

2 Click the **SAS Program** node, and then use the Code tab to enter your SAS code. The SAS Program node programming interface is the same as the interface for stand-alone SAS programs.

Creating a Flow from a SAS Program

You can convert a SAS program file to a flow. The input tables, procedures, and output tables in the program are used to create nodes in the flow.

Note: You might need to manually edit the flow that is created by the **Create a Flow from a Program** option in these cases:

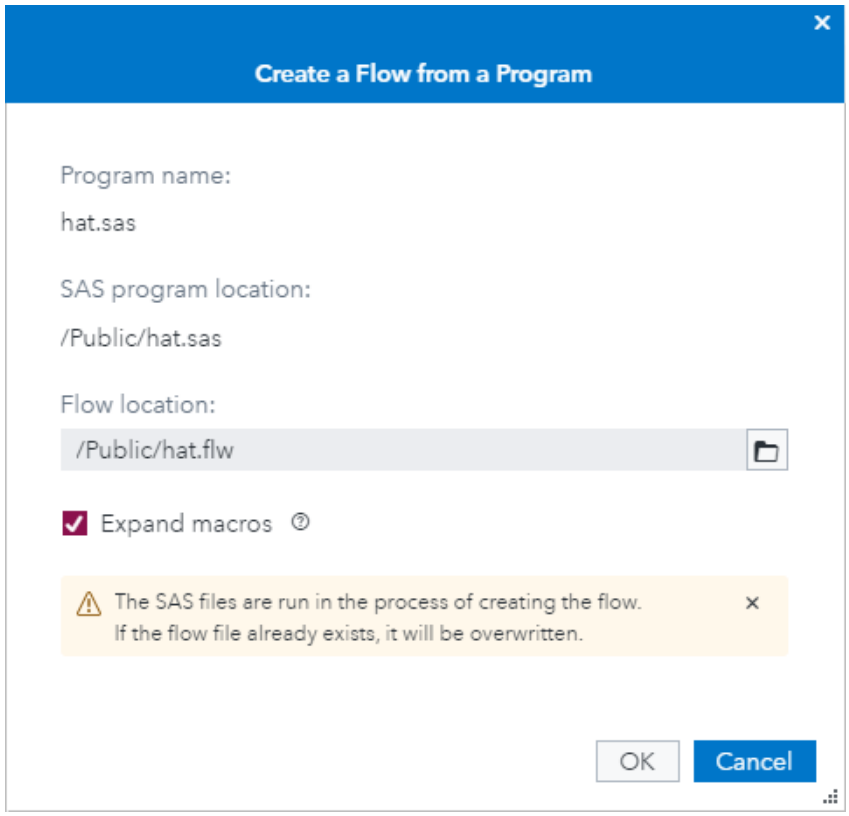
- The **Create a Flow from a Program** option cannot parse all of the possible LIBNAME options for DBMS engines. If you are creating a flow from a program that contains LIBNAME options for DBMS engines, you need to verify that the LIBNAME statement in the SAS Program node in the flow is correct and that you can access the appropriate library. You might need to manually edit the SAS Program node to add any missing LIBNAME options.
- If you are creating a flow from a program that uses CAS procedures, you might need to manually add input and output ports to the SAS Program nodes and create connections between nodes to ensure that the nodes are run in the correct order.

To create a flow from a SAS program:

1 Right-click the SAS program in the **Explorer** section of the navigation pane and select **Create flow from program**.

> **TIP** You can also create a flow from an open program by clicking ⋮ on the program toolbar and selecting **More options** ⇨ **Create flow from program**.

Note: In order to create the flow, SAS Studio runs the program.



2 By default, the new flow is created in the same location as the SAS program file. If a flow file with the same name already exists, the file is overwritten. To create the flow in a different location, click and browse to find the location that you want to use.

3 If your program includes macros and you want to create a separate SAS Program node for each macro, select **Expand macros**. This option is selected by default.

Note: Any programs that are created are embedded in the flow. For more information, see "Understanding Embedded and Externally Referenced Programs" on page 49.

4 Click **OK** to create the flow. The flow opens on a new tab in the work area.

Python Program Step: Writing Python Code in a Flow

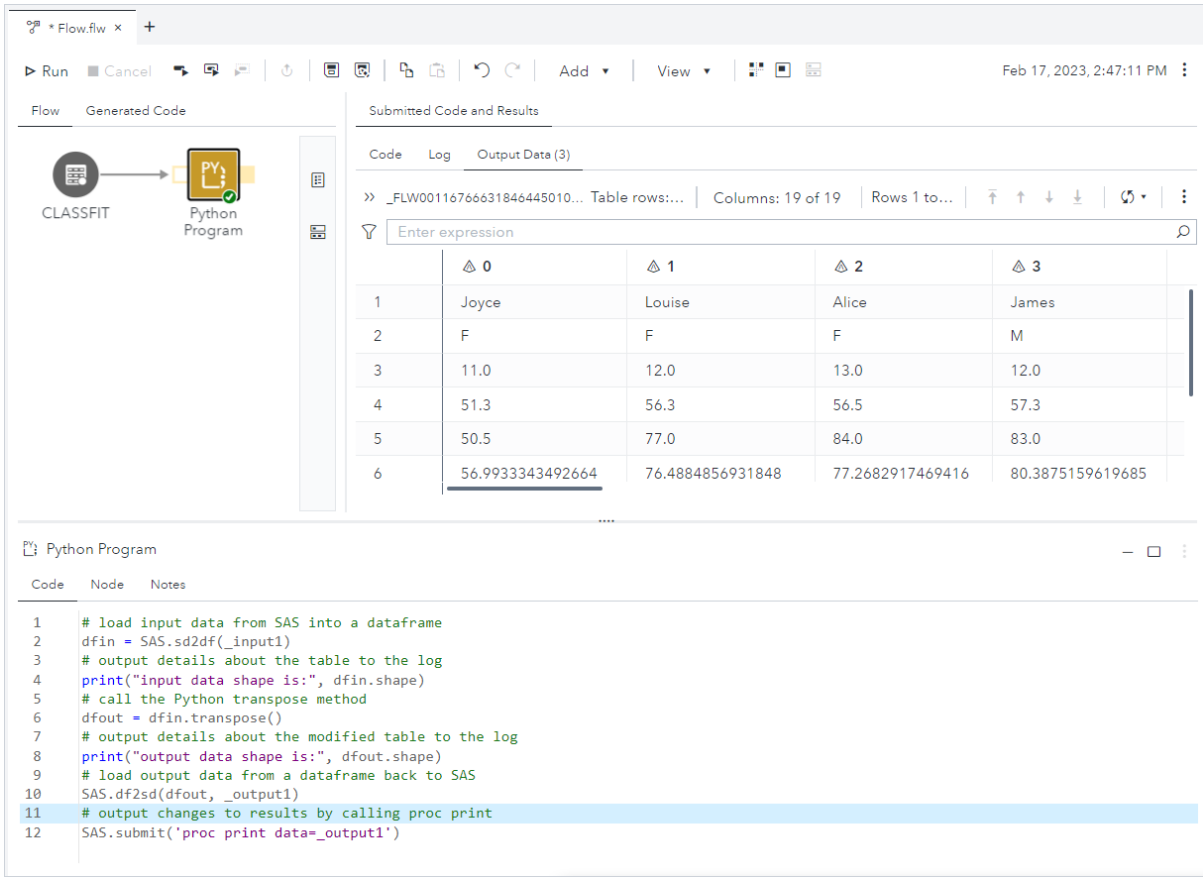
About the Python Program Step

You can use the Python Program step to run Python code in a flow without explicitly using PROC PYTHON. Your PROC PYTHON options are stored in SAS Environment Manager and do not need to be included in your Python program. SAS Studio automatically uses your Python code to generate a SAS program by using PROC PYTHON. For more information, see "PYTHON Procedure" in *Base SAS Procedures Guide*.

The code that is associated with the Python Program node is displayed in the node details on the Code tab. The Code tab includes the same color-coded, syntax-checking editor that you use to write stand-alone Python programs. For more information, see "About the Python Code Editor" in *SAS Studio: User's Guide*.

In a Python Program node, your Python program can either be embedded in the flow or reference an external program file. Embedded programs can be accessed only from within the flow; externally referenced programs are saved program files. For more information, see "Understanding Embedded and Externally Referenced Programs" on page 49.

The next example shows a flow in which a Python Program node transposes the rows and columns of the CLASSFIT table and writes the output data to the Work library. The table name is automatically generated by SAS Studio because the output port is not connected to a Table node. The example uses the `_input1` and `_output1` Python variables to reference the names of the tables that are associated with the input and output ports of the Python Program node. For more information, see "Using Macro Variables to Reference Input and Output Ports" on page 53.



Node Connection Requirements

No node connections are required to run the Python Program step. You can connect the Python Program input and output ports to Table nodes or operational nodes that create output tables, such as a Query node or an Import node, and then use macro variables to reference the tables in your program. For more information, see “Using Macro Variables to Reference Input and Output Ports” on page 53.

Note: By default, any output data is written to temporary tables in the Work library. You can specify the library and name of the output table by connecting the output port to a Table node. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Python Program: Step-by-Step Instructions

The Python Program step requires no node connections.

1 In the **Steps** section of the navigation pane, expand the **Develop** folder and double-click **Python Program**.

2 Click the **Python Program** node, and then use the Code tab to enter your Python code. The Python Program node programming interface is the same as the interface for stand-alone Python programs.

Understanding Embedded and Externally Referenced Programs

About Embedded and Externally Referenced Programs

The programs that are associated with SAS or Python Program nodes can either be embedded in the flow or externally referenced. Embedded programs are stored in the flow. You can access and edit an embedded program only from within the flow. When you create a program in your flow, the program is embedded until you save it as a file.

An externally referenced program is saved in an external location such as a SAS Content or SAS server folder. When you add an existing program to a flow, the path name of the program is stored in the program node. When you run an externally referenced program node, SAS Studio runs the external program file. You cannot make changes to an externally referenced program from within a flow. You must edit the program outside of the flow. For more information, see “Editing an Externally Referenced Program” on page 51.

TIP If you need to use a program in multiple flows, consider saving the program and adding it as an externally referenced program to each flow. If you need to make changes to the program later, you can edit the program in one place.

Changing How a Program Is Stored

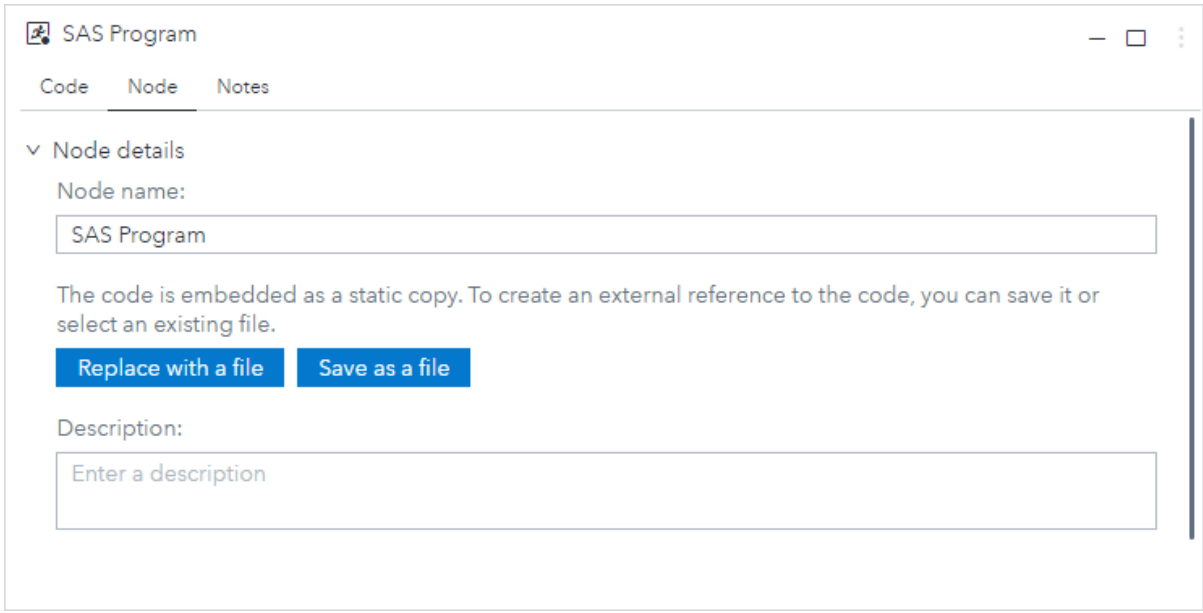
You can change whether a program is embedded in the flow or externally referenced.

To change how a program is stored:

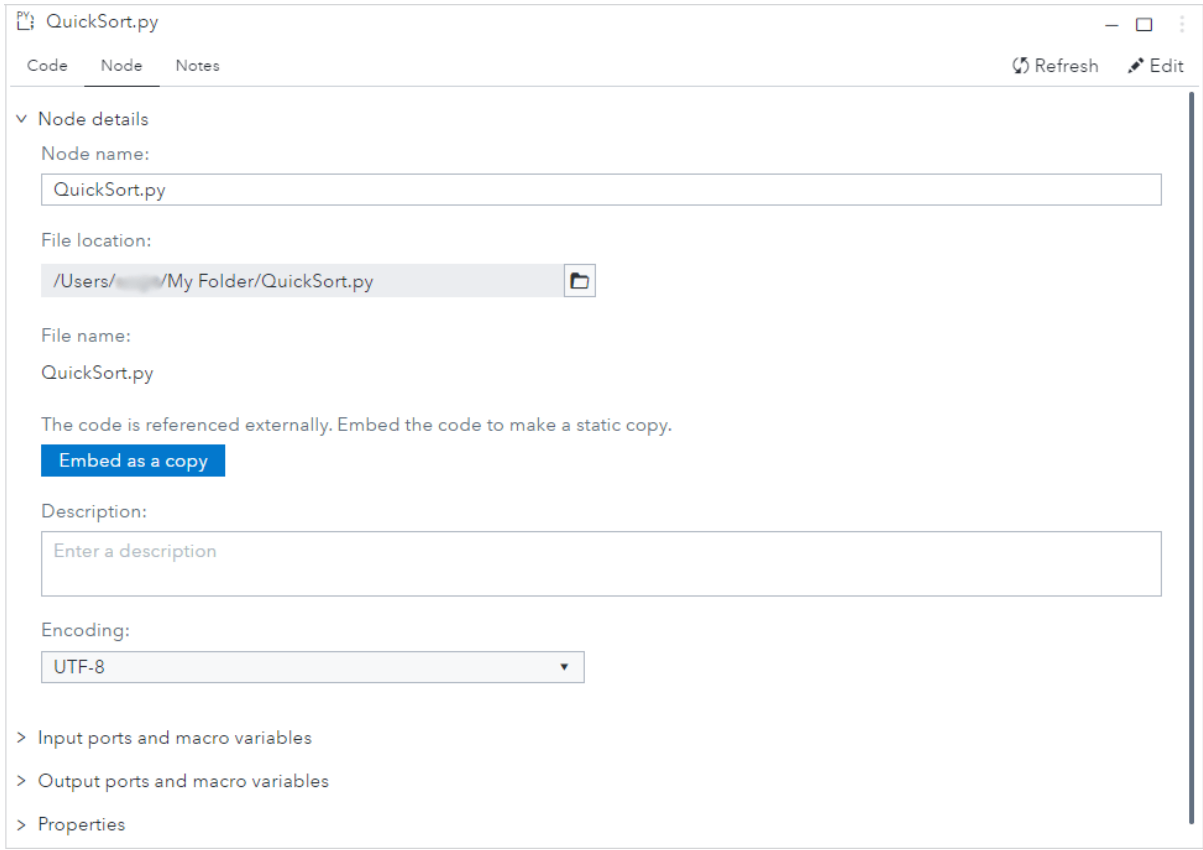
1 Click the SAS Program or Python Program node on the flow canvas, and then click the **Node** tab in the node details.

2 You can use these options on the Node tab of the node details to change how a program is stored:

- **Replace with a file** - replaces an embedded program with a saved program.
- **Save as a file** - saves an embedded program as an external program.



- **Embed as a copy** - saves a copy of an externally referenced program as an embedded program. The program no longer references the external file.



Editing an Externally Referenced Program

You cannot make changes to an externally referenced program from within a flow. To edit an externally referenced program, you must edit the program outside of the flow or embed the program in your flow.

To edit an externally referenced program:

1. Click the program node on the flow canvas, and then click the **Node** tab in the node details.
2. You can use these options to edit the program:
   - Click **Edit** on the node details toolbar to open the external program in a new tab. Edit the program and save your changes. If necessary, click **Refresh** on the node details toolbar to update the program in your flow.

     Note: You can also double-click an external program on the flow canvas to open the program in a new tab for editing.
   - Click **Embed as a copy** to embed the program in your flow. Edit the program on the Code tab of the node details. The program no longer references the external file.

Importing and Exporting Flows with Referenced Files

There are some important considerations to keep in mind when you import and export flows using SAS Environment Manager or the commmand-line interface (CLI) for the SAS Viya platform. Not all externally referenced files, including SAS programs, Python programs, and data files, are supported when you import and export a flow. For more information, see "SAS Studio Flows" in *SAS Viya Platform: Content Migration from SAS Viya 4*.

Adding Existing Programs to a Flow

You can add an existing SAS or Python program to a flow by using the **Explorer** section of the navigation pane.

Note: SAS and Python programs are added to the flow as externally referenced programs. For more information, see "Understanding Embedded and Externally Referenced Programs" on page 49.

To add an existing program to a flow:

- From the **Explorer** section of the navigation pane, drag the SAS or Python program file to the flow canvas. You can also right-click a program and select **Add to flow**.

  Note: You must have a flow tab active in the work area in order to use the **Add to flow** option.

  TIP You can also add snippets to the flow from the **Snippets** section of the navigation pane. Snippets are added to the flow as embedded programs. For more information, see "Understanding Embedded and Externally Referenced Programs" on page 49.

Copying Code to a Flow

You can copy the SAS code that is automatically generated by a task, or you can copy the code from a SAS or Python program to a program node in a flow from which you can make and save changes.

Note: The code in the SAS Program or Python Program node is embedded in the flow and is no longer associated with the original program or task. Editing this code does not affect the original program or task. For more information, see "Understanding Embedded and Externally Referenced Programs" on page 49.

To copy code to a flow:

- Open the program or task from which you want to copy code. On the toolbar, click **Code to Flow** and select the flow to which you want to add a program node.

  Note: The **Code to Flow** option is available only if these conditions are met:

- A flow must be open in the work area.
- If you are copying code from a task, the input data source must be specified.

Using Macro Variables to Reference Input and Output Ports

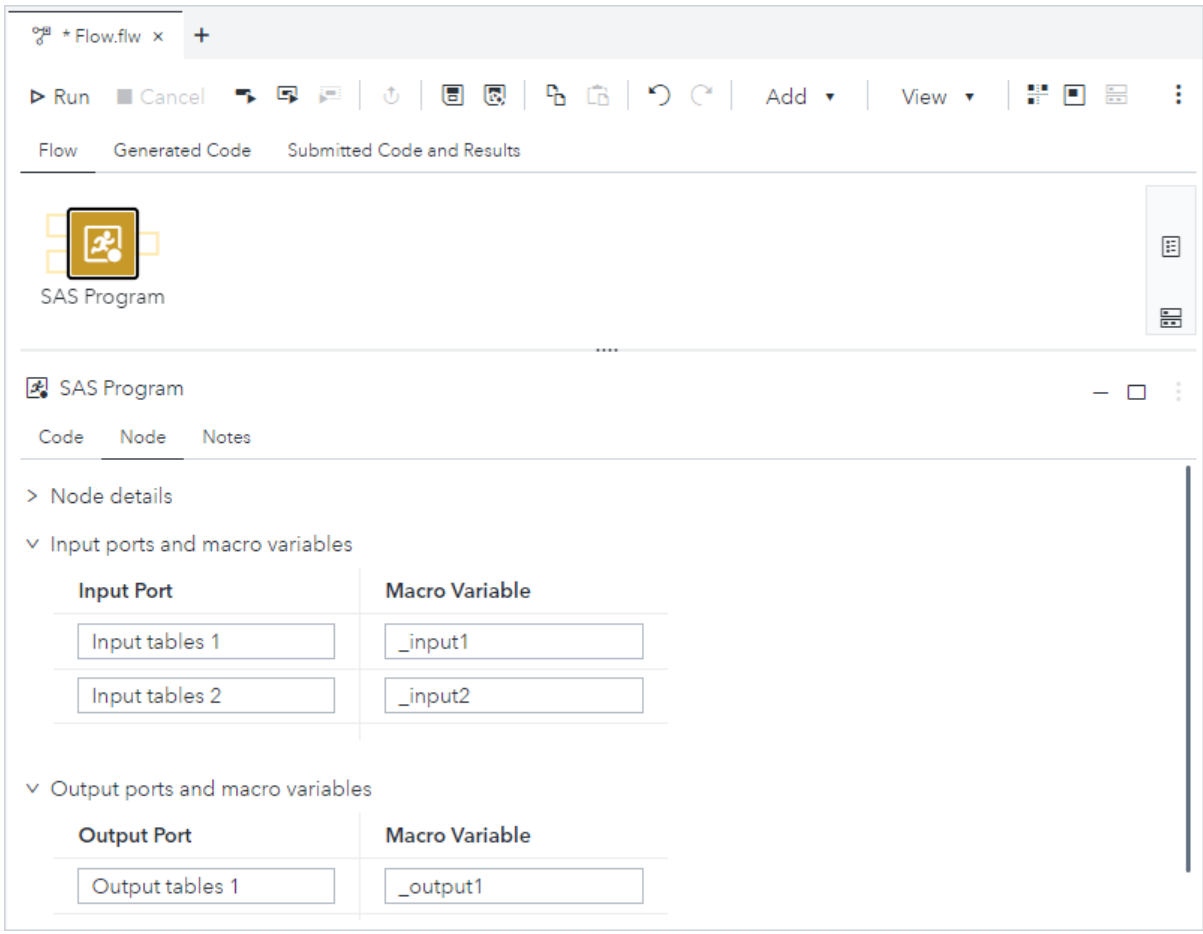
SAS Studio automatically generates SAS macro variables that represent the input and output ports of SAS Program nodes and Python variables for Python Program nodes. You can hard-code data references in your programs, or you can connect data sources to the input and output ports and use the macro variable names in your programs. By using the macro variables, you do not have to update your code if your data sources change. SAS Studio also generates macro variables that represent the number of input and output ports on the node.

Here are the default variables that SAS Studio automatically generates for both SAS Program nodes and Python Program nodes:

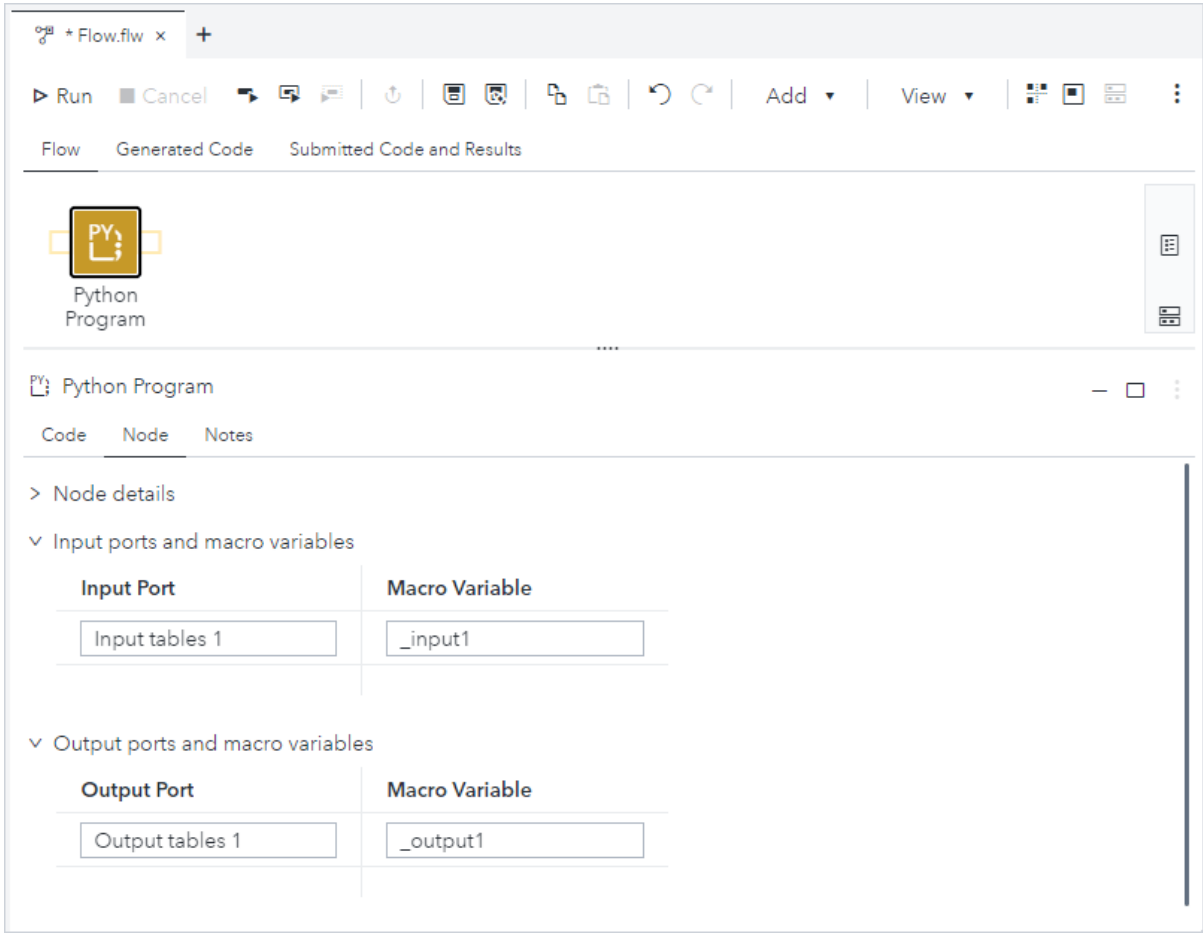
- `_input1`
- `_output1`
- `_inputCount`
- `_outputCount`

Note: If you add any input or output ports to the node, additional dedicated variables are generated for each new port.

You can view the default input and output ports and the associated macro variables of a program node by clicking the node in the flow, and then clicking the **Node** tab. The following example shows the macro variables that are generated for a SAS Program node with two input ports and one output port.

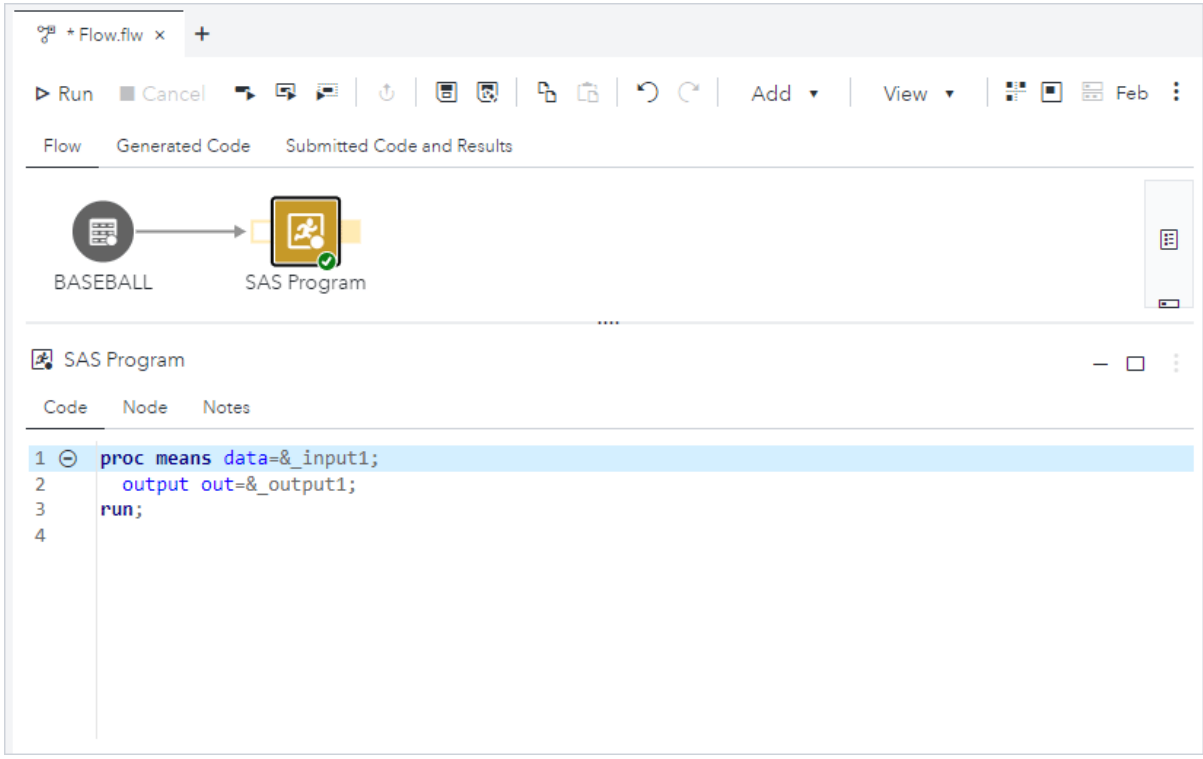


The next example shows the variables that are generated for a Python Program node with one input port and one output port.

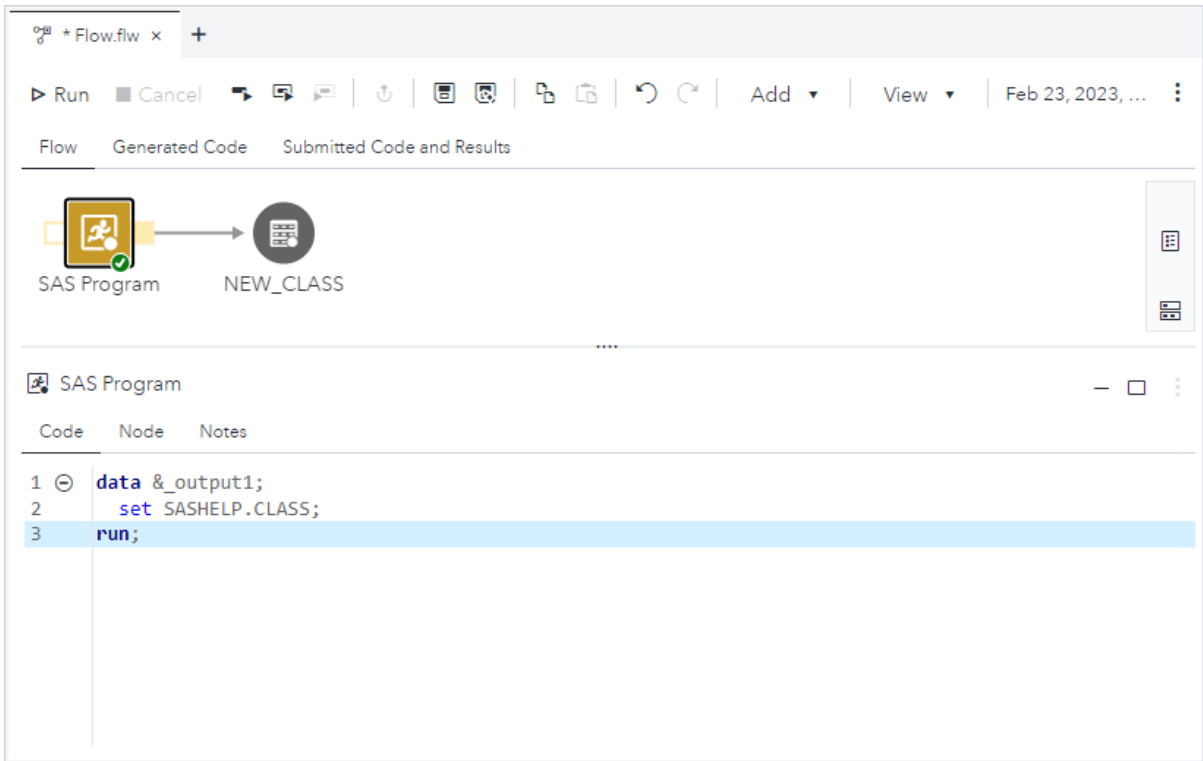


Note: You can change the default names of the macro variables. You can then reference the custom macro variable names in your code. For more information, see "Macro Variables Defined by Users" in *SAS Macro Language: Reference*.

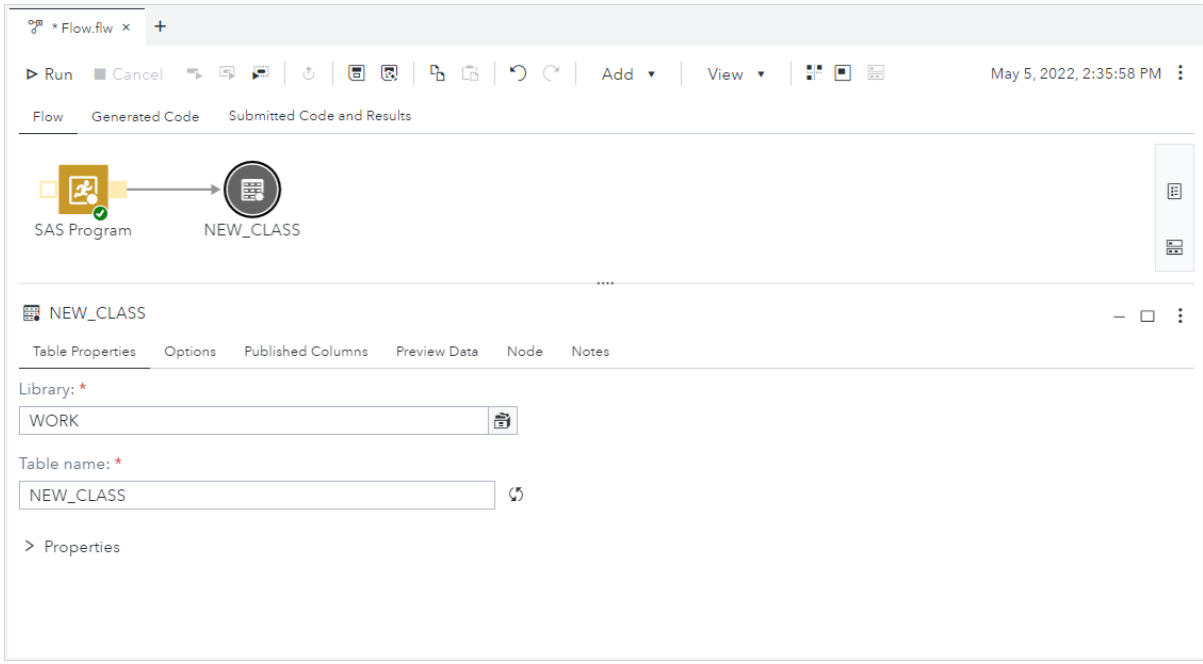
In the following example, the BASEBALL table is connected to a SAS Program node. The SAS Program node runs the MEANS procedure against the BASEBALL table. The code that is associated with the SAS Program node specifies the `&_input1` macro variable as the data source for the MEANS procedure and the `&_output1` macro variable as the destination for the output data. No output table is connected to the program node, so the results are written to a table in the Work library.



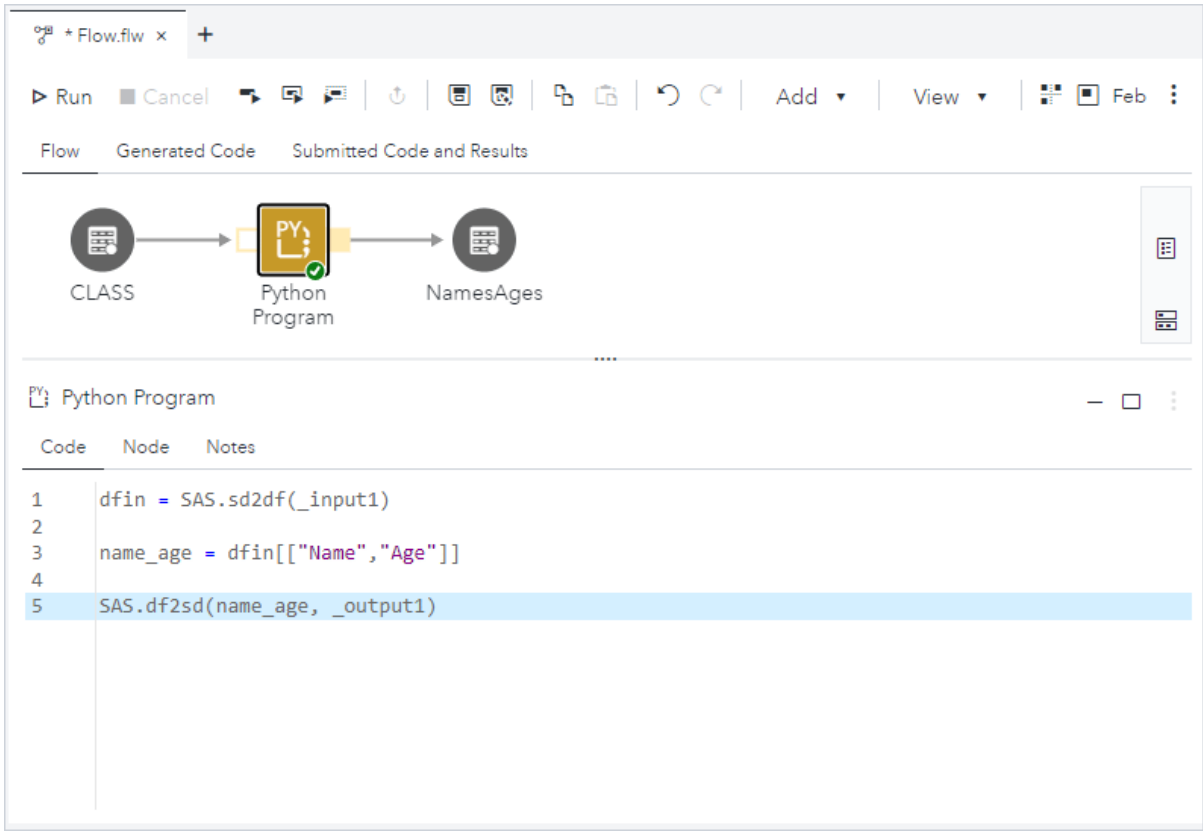
The following example shows a SAS Program node that writes output data to the NEW_CLASS table. The SAS Program node runs a DATA step to create a table that is based on the CLASS table. The code that is associated with the SAS Program node specifies the `&_output1` macro variable as the destination for the output data.



The table properties for the Table node specify the Work library and NEW_CLASS table name.



The next example shows a Python program that uses the CLASS table as input and writes two columns from the table to the NamesAges table in the Work library. The code that is associated with the Python Program node specifies the `_input1` variable as the data source and the `_output1` variable as the destination for the output data.



5

Transforming Data

Understanding the Steps That Subset Data

You can choose from among three different steps to select a subset of data from an input table. Each step can be useful in different situations, depending on your needs:

- **Branch Rows** - splits a table into multiple output tables based on conditions that you specify by using column values. The Branch Rows step performs only a single function in a flow, which makes it easy to understand the function of a Branch Rows node in a flow that is complicated and includes many nodes. For more information, see "Branch Rows" on page 62.
- **Filter Rows** - selects a subset of rows from an input table and writes the rows to a single output table. The Filter Rows step performs only a single function in a flow, which makes it easy to understand the function of a Filter Rows node in a flow that is complicated and includes many nodes. For more information, see "Filter Rows" on page 73.
- **Query** - selects, filters, and sorts columns from one or more input tables that can be joined together and writes the rows to a single output table. The Query step can perform multiple functions, which can make understanding a flow more complicated. To understand the functions that a Query node performs in a flow, you must examine the options that are specified for the node. For more information, see "Query" on page 103.

Note: The Branch Rows and Filter Rows steps are available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Table 5.1 *Comparison of Steps That Subset Data*

	Branch Rows	Filter Rows	Query
Selects a subset of rows based on one or more filter conditions	x	x	x
Includes option to use the Expression Builder for creating expressions	x	x	x
Includes a condition for rows that do not match any other condition	x		
Writes rows to multiple output tables	x		
Selects rows from multiple input tables			x
Filters, sorts, and groups columns			x
Creates calculated columns			x

Branch Rows

About the Branch Rows Step

By default, the Branch Rows step splits a table into two output tables based on conditions that you specify by using column values. The conditions that you create do not need to be mutually exclusive: a row can be written to more than one output table.

Note: The Branch Rows step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

TIP You can also use the Filter Rows step and the Query step to select a subset of rows from an input table. For more information, see "Understanding the Steps That Subset Data" on page 60.

Node Connection Requirements

In order to run the Branch Rows step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to temporary tables in the Work library. You can specify the library and name of the output tables by connecting the output ports to Table nodes. For more information, see "Adding a Table from a SAS Library to a Flow" on page 36.

Example: Splitting the Sashelp.Class Data Set

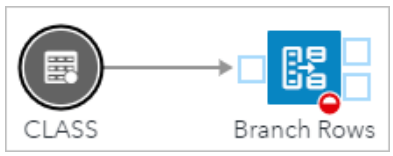
In this example, you split the Sashelp.Class data set into two tables: one for male students and another for female students.

To create this example:

1 From the main SAS Studio menu, select **New** ⇨ **Flow**.

2 In the **Steps** section of the navigation pane, expand the **Transform Data** folder, and double-click **Branch Rows**.

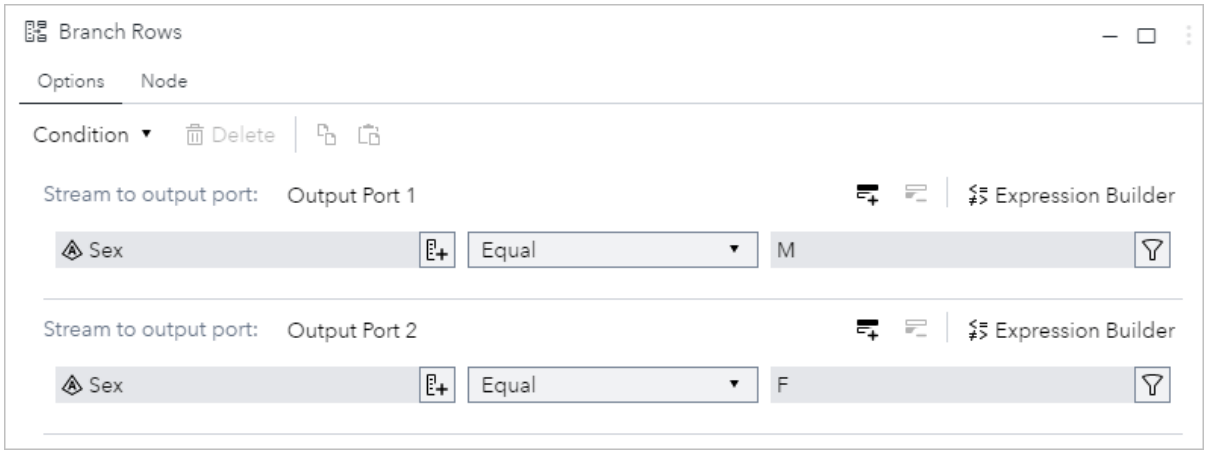
3 In the **Libraries** section of the navigation pane, expand the Sashelp library. Drag the Class data set onto the input port of the Branch Rows node. Drop the data set on the flow canvas when the tooltip on your mouse pointer changes to `Connect to input port`.



4 Click the **Branch Rows** node to specify the conditions for the output port. On the Options tab, click for Output Port 1 and select the *Sex* column. Accept the default condition of `Equal`, and click .

5 In the Add Filter window, click . Click **Get Values** in the Select Value window. Select `M` and click **OK**. Click **Filter** to create the condition.

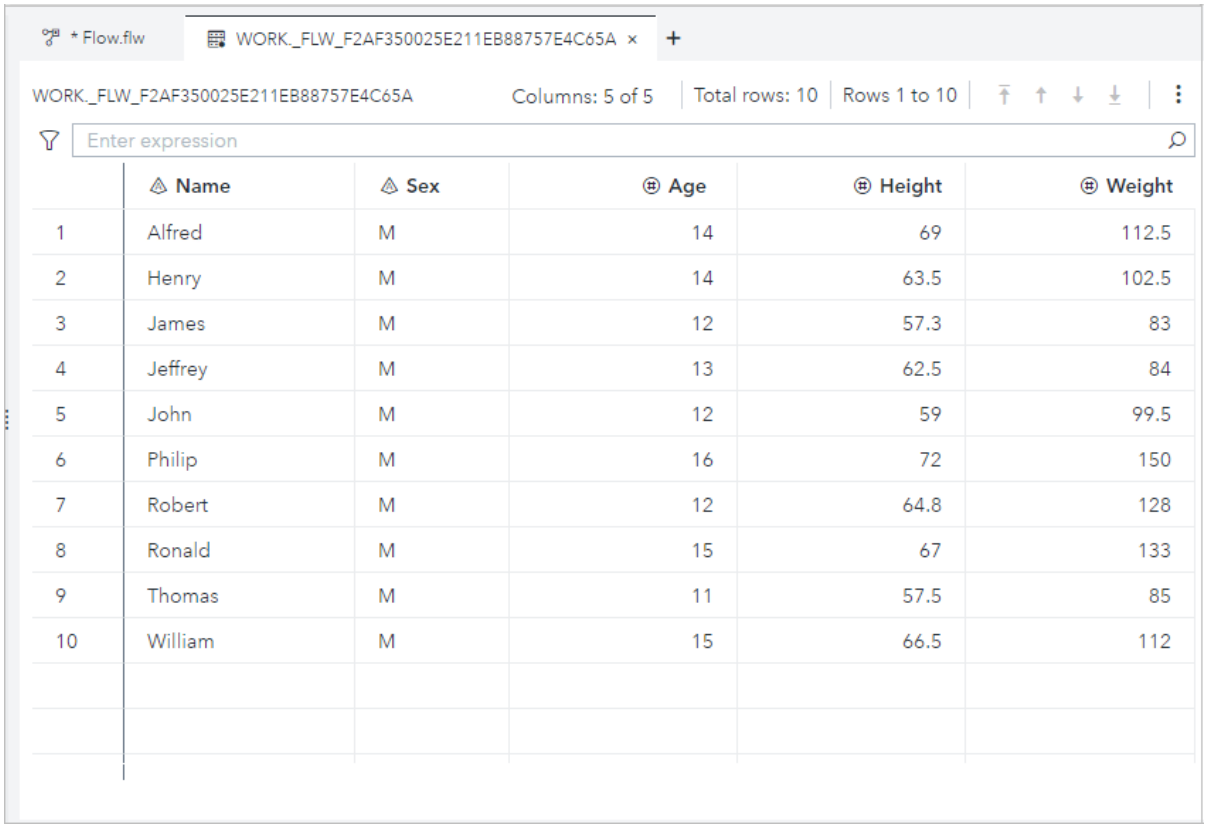
6 Repeat steps 4 and 5 to create a similar condition for Output Port 2 for female students.



7 To run the flow, click Run.

Here is the default output table of male students.

TIP You can specify the library and name of the output tables by connecting the output ports to Table nodes. For more information, see "Adding a Table from a SAS Library to a Flow" on page 36.

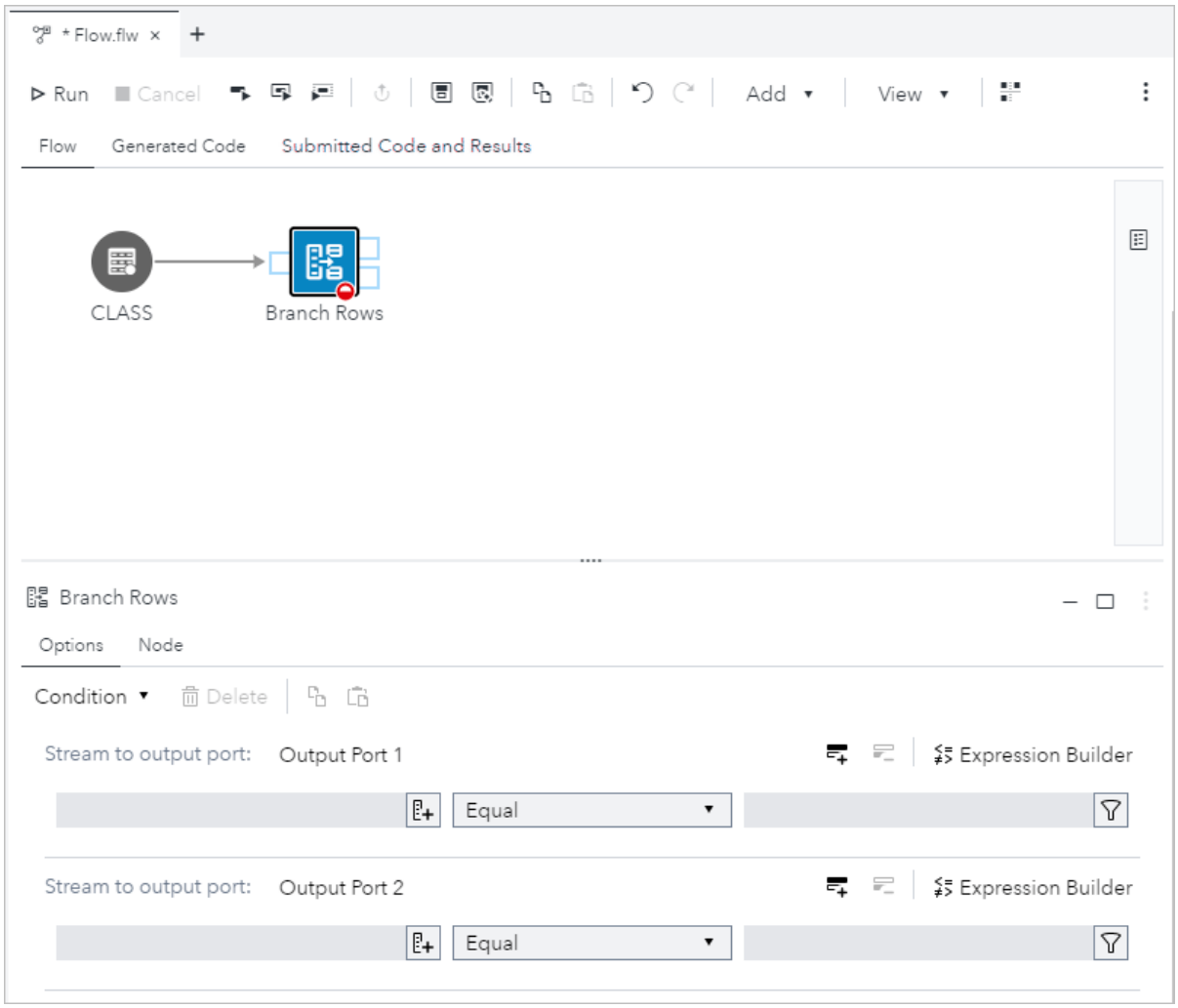


	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5
2	Henry	M	14	63.5	102.5
3	James	M	12	57.3	83
4	Jeffrey	M	13	62.5	84
5	John	M	12	59	99.5
6	Philip	M	16	72	150
7	Robert	M	12	64.8	128
8	Ronald	M	15	67	133
9	Thomas	M	11	57.5	85
10	William	M	15	66.5	112

Branch Rows: Step-by-Step Instructions

By default, the Branch Rows step includes two output ports and the options for two conditions. You can choose to specify only one condition and output port, or you can add additional conditions and output ports. You can also add a nonmatching condition that writes the rows that do not meet any condition to a separate output port.

1. In the **Steps** section of the navigation pane, expand the **Transform Data** folder, and double-click **Branch Rows**.
2. Connect the input port of the Branch Rows node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.
3. Click the **Branch Rows** node, and then click the **Options** tab.



4 Click  for Output Port 1, and select the column that you want to use. Click **OK**.

> **TIP** To use the expression builder to create a condition, click **Expression Builder** on the toolbar for the condition. For more information, see "Building an Expression" in *SAS Studio: User's Guide*.

5 Select a comparison operator from the operator drop-down list. The default value is `Equal to`.

6 If the operator that you have selected requires a value, click  . In the Add Filter window, enter or select a value in the **Value** box. To choose from a list of values, click and click **Get Values** in the Select Value window. Select the values that you want to use and click **OK**.

> **TIP** If you want to search for a value in the Select Value window, use the unformatted value.

7 Depending on the data type of the column that you are using in the condition, you can choose from the following options:

- **Match case** – retrieves only rows that match the capitalization of the value that you specify. If this option is not selected, the UPPER function is applied to the expression. This option is not selected by default.
- **Quote string** – encloses values in single quotation marks. This option is selected by default. If you are using a macro variable or other value that is evaluated when the filter is run, you should clear this option.
- **Allow macros** – enables you to enter character values in a filter on a numeric column.

8 Click **Filter** to add values to the condition.

9 To add another row to the condition, click and repeat steps 4–8. If you create more than one comparison expression in your condition, the default relationship between these filter elements is AND. You can change the relationship between filter elements from AND to OR by clicking the relationship drop-down list.

Note: To delete a row from a condition, select the row and click

10 Repeat steps 4–9 to create a condition for Output Port 2.

11 To create additional conditions, select **Condition** ⇨ **Add condition** and repeat steps 4–9.

To add a condition for the rows that do not match any other condition, select **Condition** ⇨ **Add nonmatching condition**.

12 To run the flow, click Run.

Calculate Columns

About the Calculate Columns Step

The Calculate Columns step enables you to create an output table that is based on the input table. The output table includes all of the columns from the input table. You can replace a column in the output table by applying a function to the corresponding column in the input table. You can also create additional columns that are based on columns from the input table.

When you replace or create a column for the output table, a calculation card is created for the column. You can use the card to edit the expression that is associated with the column and view the properties of the column. You can create multiple cards for the same column if you want to apply several functions to a column. The cards are processed in order from top to bottom. You can move columns up and down the list to change the processing order.

New columns are added to the beginning of the output table. Replaced columns are also added to the beginning of the output table, if you change any of the column attributes, including Label, Type, Length, Format, and Informat. The order in which new and replaced columns are added to the beginning of the table is based on the order of the cards.

Note: The Calculate Columns step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Node Connection Requirements

In order to run the Calculate Columns step, you must create connections to the input and output ports of the node as indicated here:

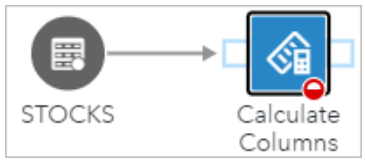
Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to temporary tables in the Work library. You can specify the library and name of the output tables by connecting the output ports to Table nodes. For more information, see "Adding a Table from a SAS Library to a Flow" on page 36.

Example: Creating a Table Based on the Sashelp.Stocks Data Set

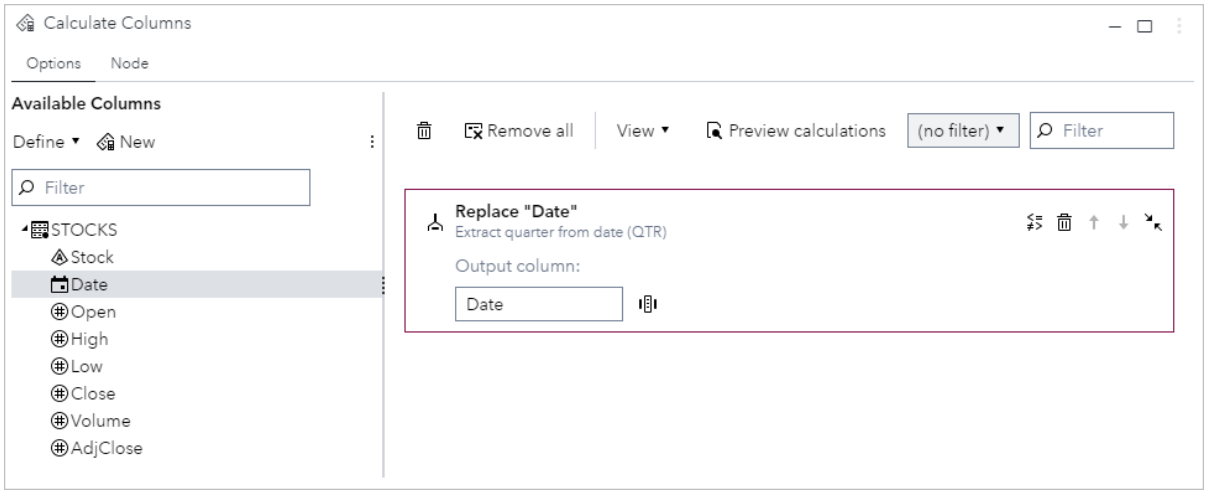
In this example, you create a table from the Sashelp.Stocks data set. In the new table, the date column is replaced with a column of calendar quarter values. You also add a column to calculate the stock price change.

To create this example:

1. From the main SAS Studio menu, select **New** ⇨ **Flow**.
2. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Calculate Columns**.
3. In the **Libraries** section of the navigation pane, expand the Sashelp library. Drag the Stocks data set onto the input port of the Calculate Columns node. Drop the data set on the flow canvas when the tooltip on your mouse pointer changes to `Connect to input port`.



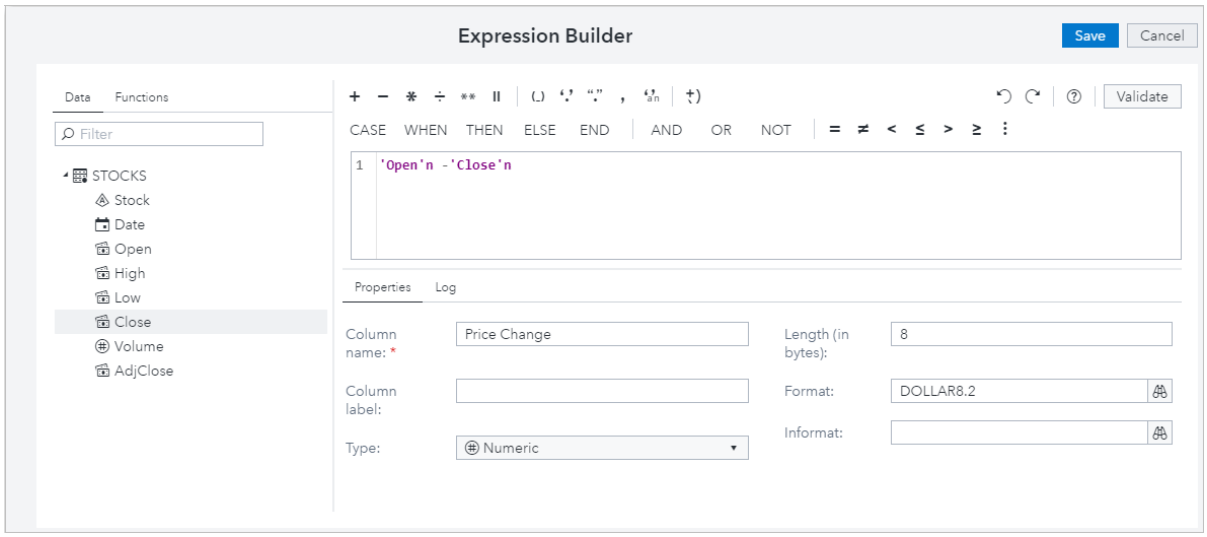
4 Click the **Calculate Columns** node to define the columns for the output table. On the Options tab, select the Date column. Select **Define** ⇨ **Extract** ⇨ **Date** ⇨ **Quarter** to extract the quarter from the Date data. A `Replace "Date"` card is added to the node calculations. In the output table, the data in the Date column is replaced with the calendar quarter.



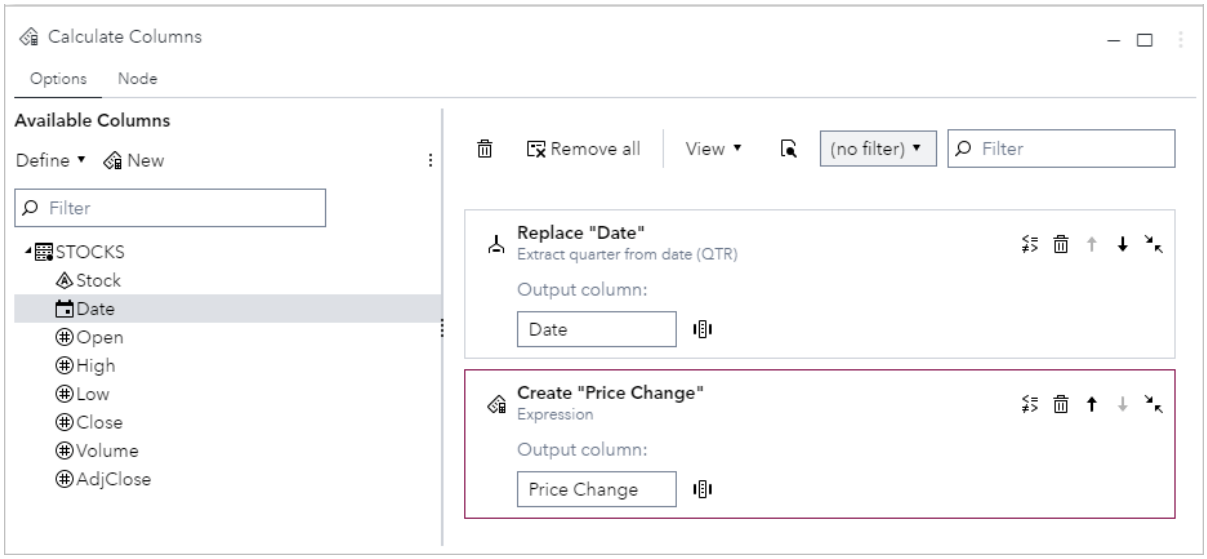
5 Add a column to the table by clicking **New** in the Available Columns area. In the Expression Builder, drag the `Open` column to the expression box. Click the minus operator, and then drag the `Close` column to the expression box. The expression box should contain this expression:

```
'Open'n -'Close'n
```

6 On the Properties tab, enter *Price Change* in the **Column name** box. Change **Type** to `Numeric`, and specify `DOLLAR8.2` as the format.



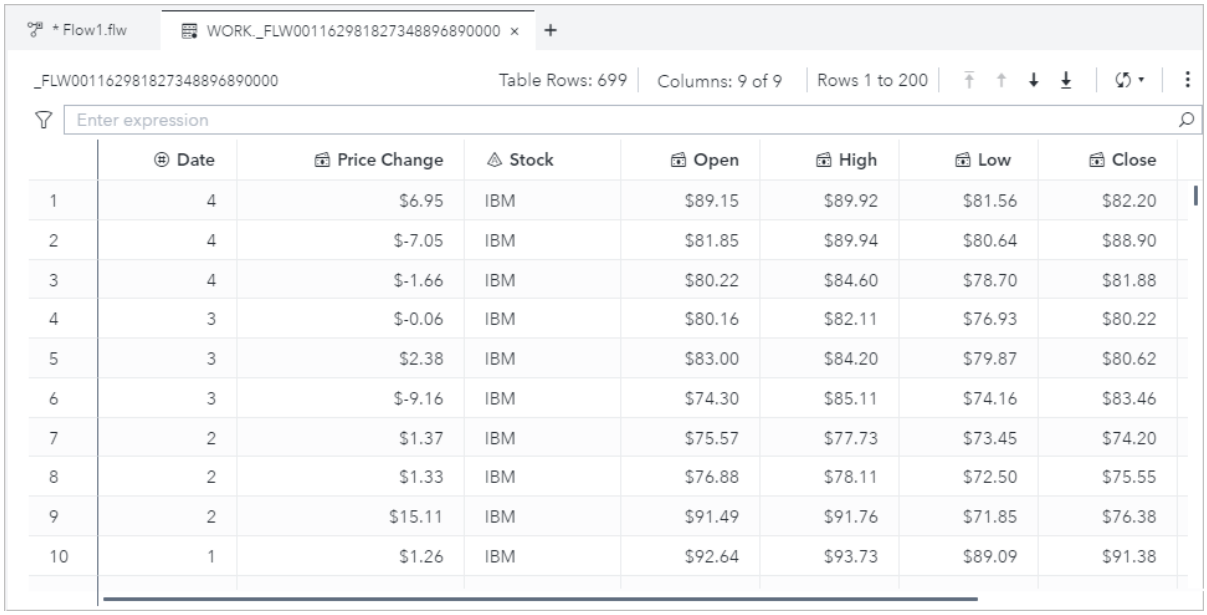
7 Click **Save** to add the new column to the table. A `Create "Price Change"` card is added to the node calculations.



8 To run the flow, click ▷ Run.

Here is the output table with the replaced Date column and the new Price Change column.

> **TIP** You can specify the library and name of the output table by connecting the output port to a Table node. For more information, see "Adding a Table from a SAS Library to a Flow" on page 36.

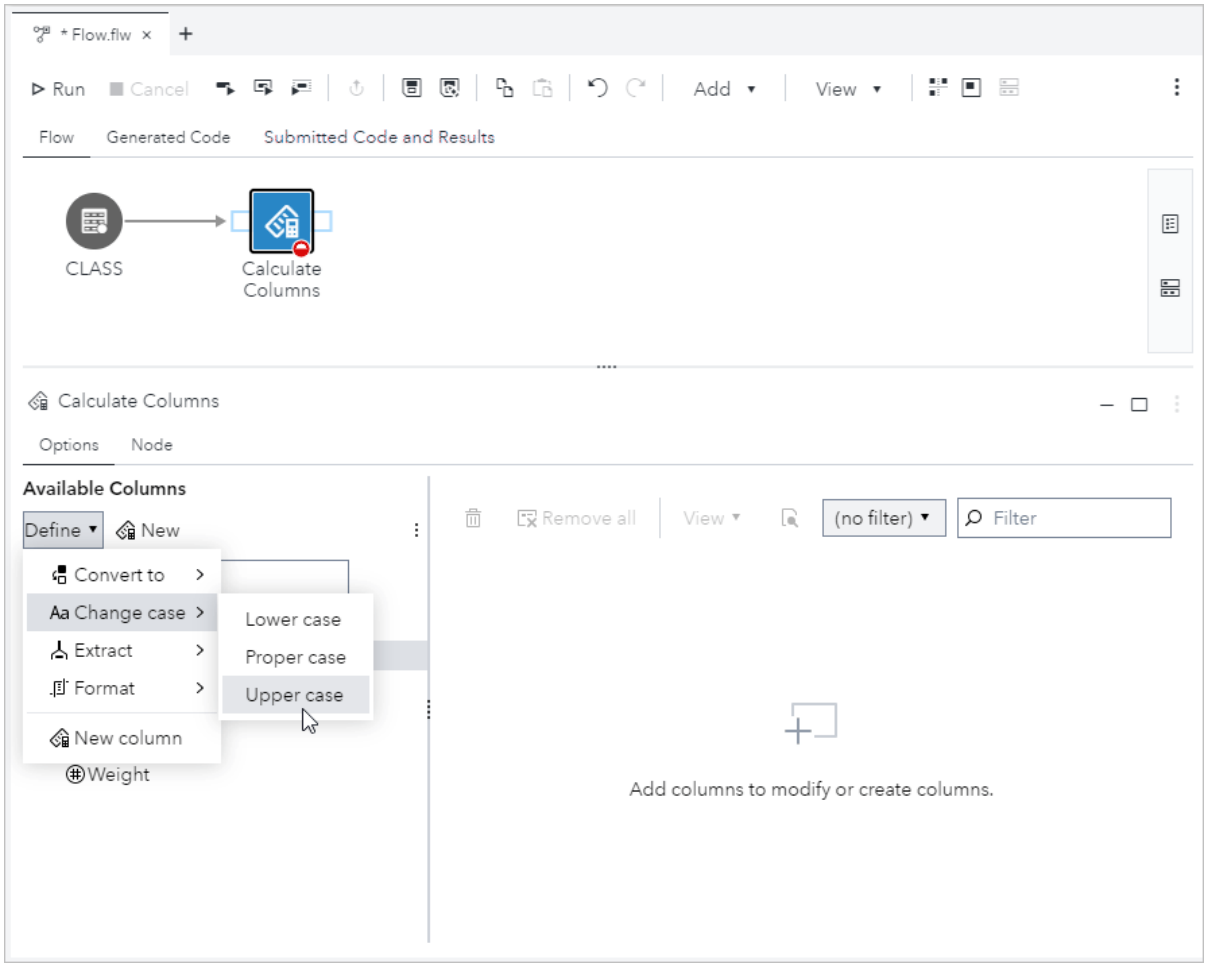


	Date	Price Change	Stock	Open	High	Low	Close
1	4	$6.95	IBM	$89.15	$89.92	$81.56	$82.20
2	4	$-7.05	IBM	$81.85	$89.94	$80.64	$88.90
3	4	$-1.66	IBM	$80.22	$84.60	$78.70	$81.88
4	3	$-0.06	IBM	$80.16	$82.11	$76.93	$80.22
5	3	$2.38	IBM	$83.00	$84.20	$79.87	$80.62
6	3	$-9.16	IBM	$74.30	$85.11	$74.16	$83.46
7	2	$1.37	IBM	$75.57	$77.73	$73.45	$74.20
8	2	$1.33	IBM	$76.88	$78.11	$72.50	$75.55
9	2	$15.11	IBM	$91.49	$91.76	$71.85	$76.38
10	1	$1.26	IBM	$92.64	$93.73	$89.09	$91.38

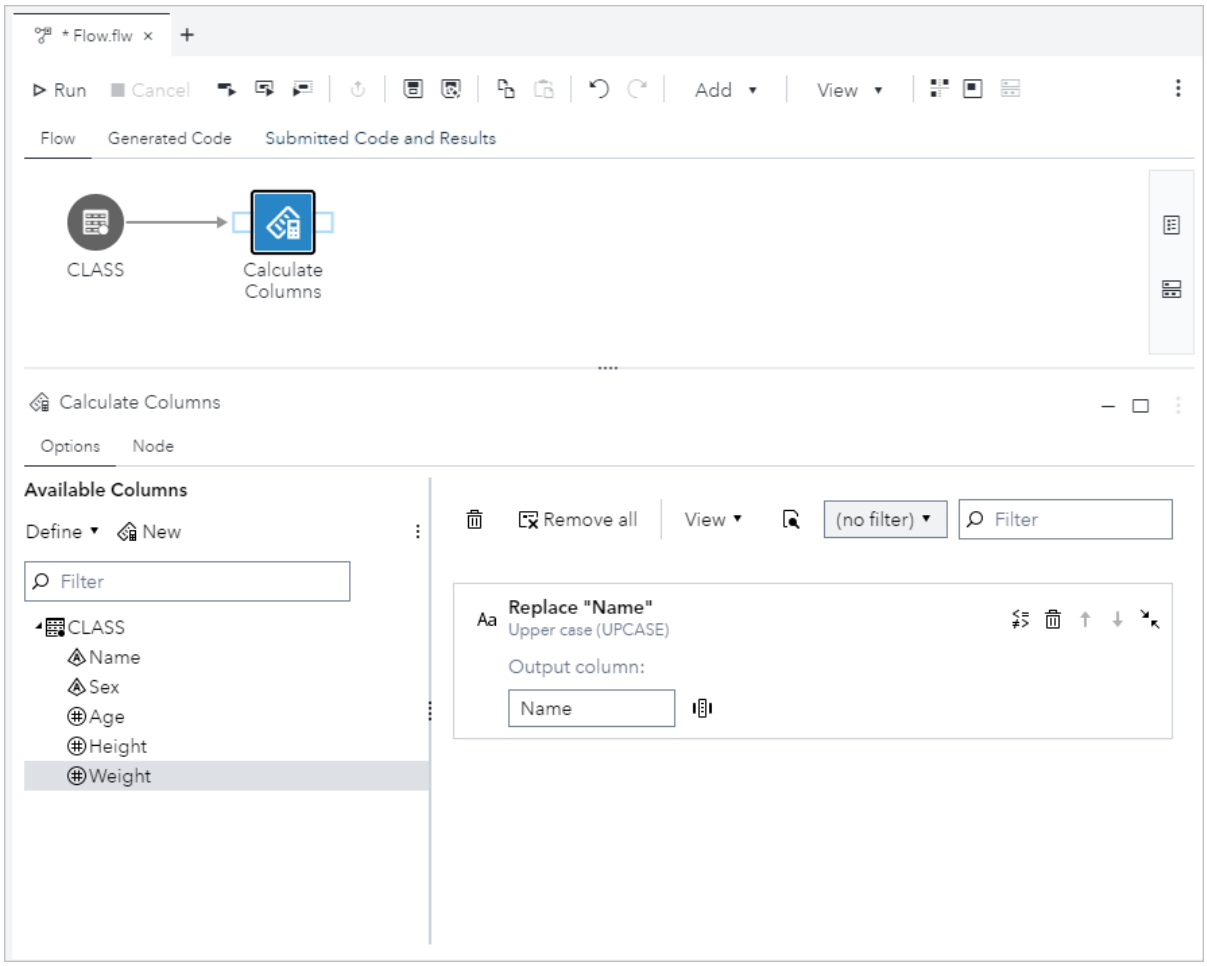
Calculate Columns: Step-by-Step Instructions

The Calculate Columns step requires one input port and one output port. You must select at least one column from the input table to replace in the output table or to use as the basis for creating a column.

1. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Calculate Columns**.
2. Connect the input port of the Calculate Columns node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see "Connecting Nodes" on page 12.
3. Click the **Calculate Columns** node, and then use the Options tab to select a column to replace or use as the basis for a new column.
4. To replace a column:
   a. On the Options tab, select one or more columns that you want to replace. Click **Define** and select the function that you want to apply to the columns.



A card for each replaced column is added to the node calculations. The following figure shows a card that replaces the values in the Name column with uppercase values.



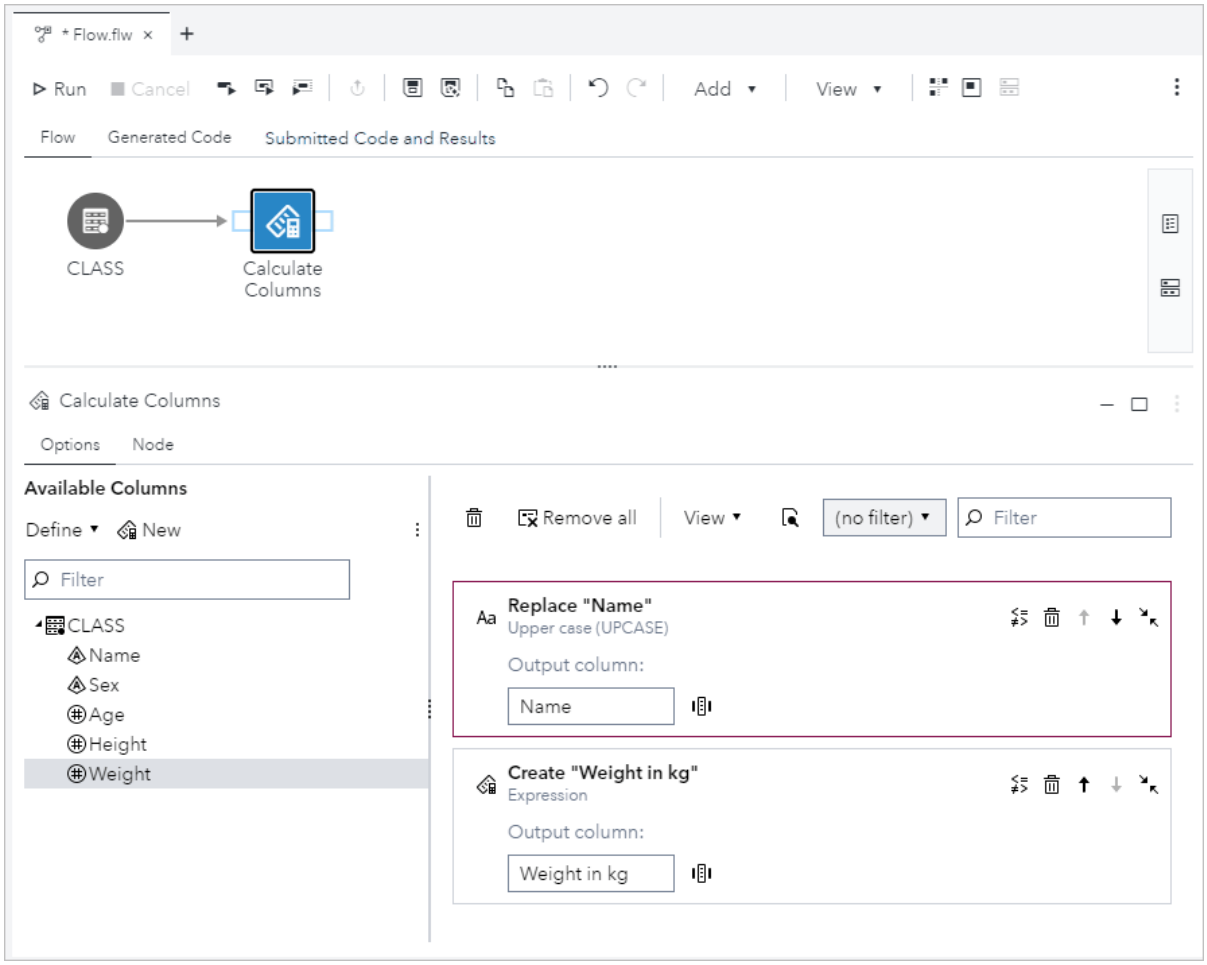
TIP You can create a column in the output table by entering a new name for the output column in a Replace card. For example, if you create a card to replace the values in the Name column with uppercase values and rename the output column to `UpperName`, then the output table includes both the original Name column and the new UpperName column.

To create a column based on an existing column:

a On the Options tab, select the column that you want to use as the basis for the new column. Click **New**.

b In the Expression Builder window, use the options to create the expression for the new column, or enter the expression in the expression box.

c Use the Properties tab to specify the column name, data type, and other properties. Click **Save**. A Create card is added to the node calculations. The following figure shows a new card that is based on the Weight column and creates a column for weight in kilograms.



5 You can use the calculation cards to perform these actions:

	View or edit the expression by using the Expression Builder.
	View the output column properties.
	Delete the column calculation.
,	Move the calculation card up or down the list.
,	Expand or collapse a card.

Note: The calculation cards are processed by SAS Studio in order from top to bottom. You can move cards up and down the list to change the processing order.

> **TIP** You can use the Manage Columns step to change the order of the columns in the output table. For more information, see "Manage Columns" on page 85.

6 To preview all the calculations that have been generated for the node, click . To copy the calculations to the clipboard, click **Copy**.

7 To run the flow, click ▷ Run.

Filter Rows

About the Filter Rows Step

You can use the Filter Rows step to select a subset of rows from an input table and write the rows to an output table. The Filter Rows step can be combined with other steps in which you want to work with only a subset of the data in a table. For example, you can create a flow that uses the Filter Rows step to create a table that contains only baseball players who have had more than 10 home runs, and then use the Branch Rows step to split the filtered data into separate tables for each team.

> **TIP** You can also use the Branch Rows step and the Query step to select a subset of rows from an input table. For more information, see "Understanding the Steps That Subset Data" on page 60.

You can choose to create filter expressions by using the user interface in the step or you can create the filter by using the Expression Builder. For more information, see "Building an Expression" in *SAS Studio: User's Guide*. SAS Studio automatically generates the appropriate DATA step WHERE statement for the filter when the step is run.

Note: The Filter Rows step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Node Connection Requirements

In order to run the Filter Rows step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	■ Table node or ■ Operational node that requires an input table, such as a Query node, a SAS Program node, or an Export node

Example: Subsetting Data from the Sashelp.Baseball Data Set

In this example, you select a subset of rows from the Sashelp.Baseball data set and write the rows to a new output table named BestHitters. The BestHitters table contains a subset of the rows and all of the columns that are in the Sashelp.Baseball data set.

To create this example:

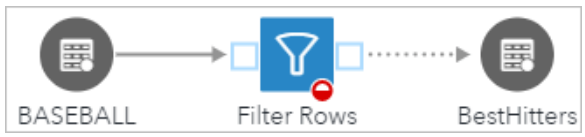
1 From the main SAS Studio menu, select **New** ⇨ **Flow**.

2 In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Filter Rows**.

3 In the **Libraries** section of the navigation pane, expand the Sashelp library. Drag the Baseball data set onto the input port of the Filter Rows node. Drop the data set on the flow canvas when the tooltip on your mouse pointer changes to `Connect to input port`.

4 On the flow toolbar, click **Add** ⇨ **Table**. Connect the Filter Rows node to the Table node by clicking the Filter Rows output port and dragging the mouse pointer to the Table node.

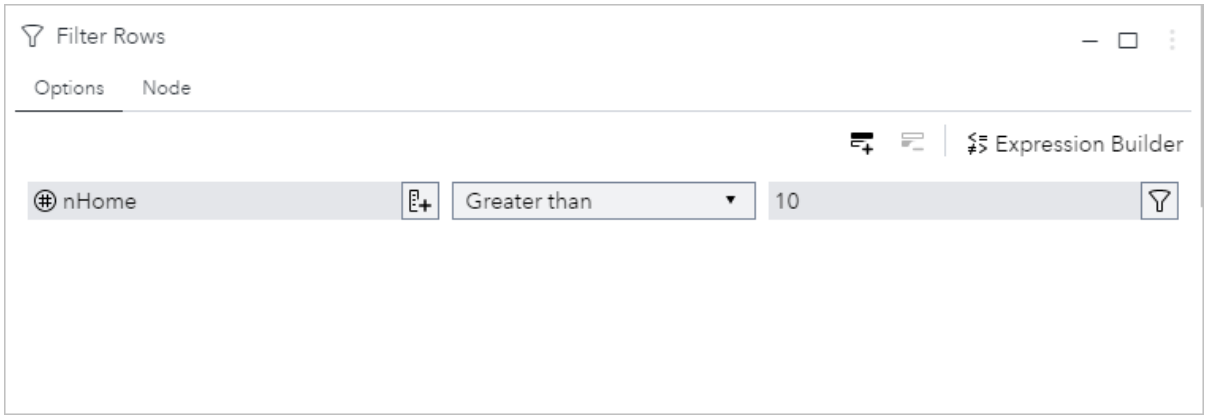
5 Click the **Table** node. On the **Table Properties** tab, click  beside the **Library** box, and select the Work library. In the **Table** box, enter *BestHitters*.



6 Click the **Filter Rows** node to specify the filter condition for the output table. On the Options tab, click  and select the `nHome` column. Click **OK**.

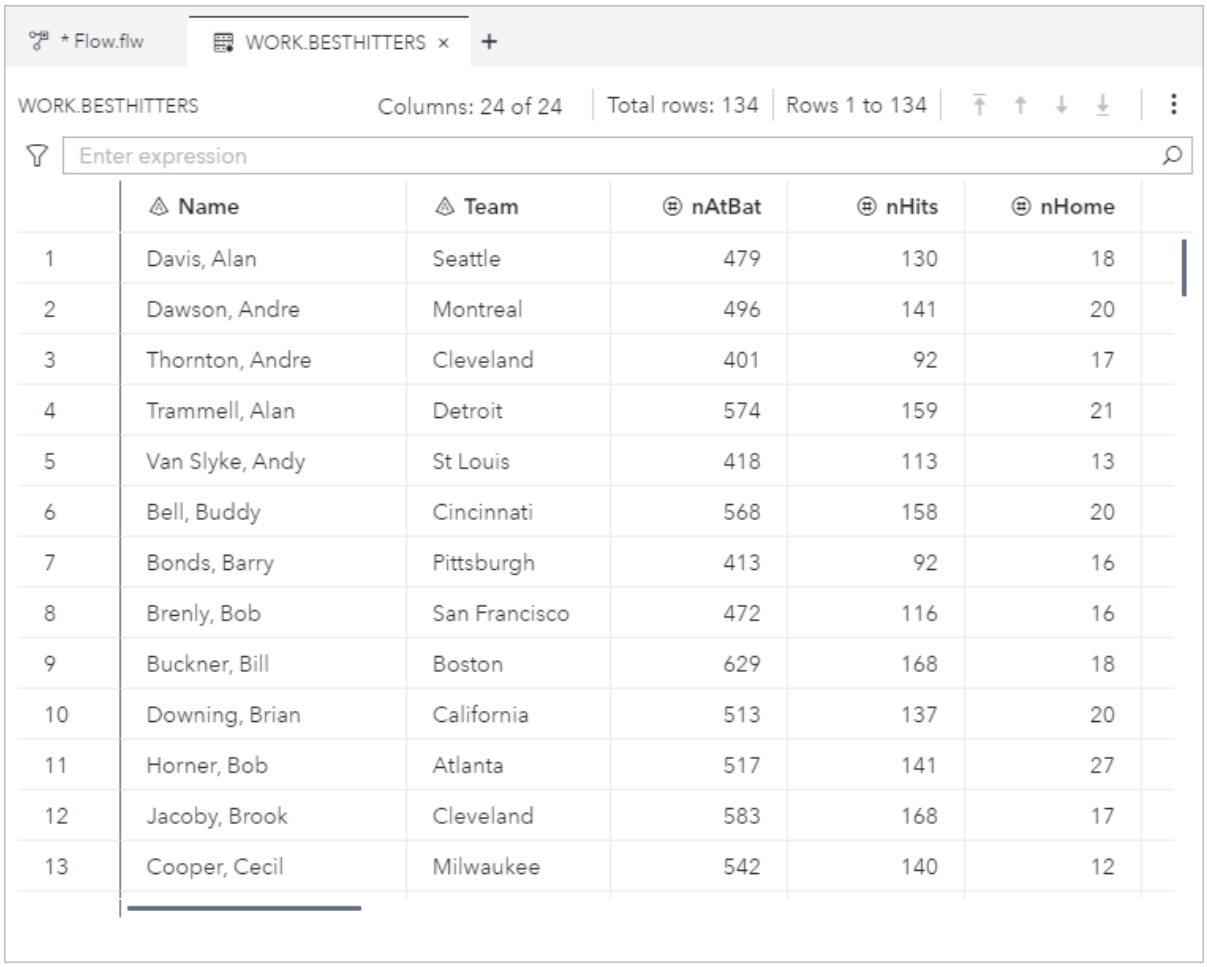
7 From the operator drop-down list, select `Greater than`, and then click  .

8 In the Add Filter window, click . Click **Get Values** in the Select Value window. Select `10` and click **OK**. Click **Filter** to create the condition.



9 To run the flow, click Run.

Here is the output table of players with more than 10 home runs.



	Name	Team	nAtBat	nHits	nHome
1	Davis, Alan	Seattle	479	130	18
2	Dawson, Andre	Montreal	496	141	20
3	Thornton, Andre	Cleveland	401	92	17
4	Trammell, Alan	Detroit	574	159	21
5	Van Slyke, Andy	St Louis	418	113	13
6	Bell, Buddy	Cincinnati	568	158	20
7	Bonds, Barry	Pittsburgh	413	92	16
8	Brenly, Bob	San Francisco	472	116	16
9	Buckner, Bill	Boston	629	168	18
10	Downing, Brian	California	513	137	20
11	Horner, Bob	Atlanta	517	141	27
12	Jacoby, Brook	Cleveland	583	168	17
13	Cooper, Cecil	Milwaukee	542	140	12

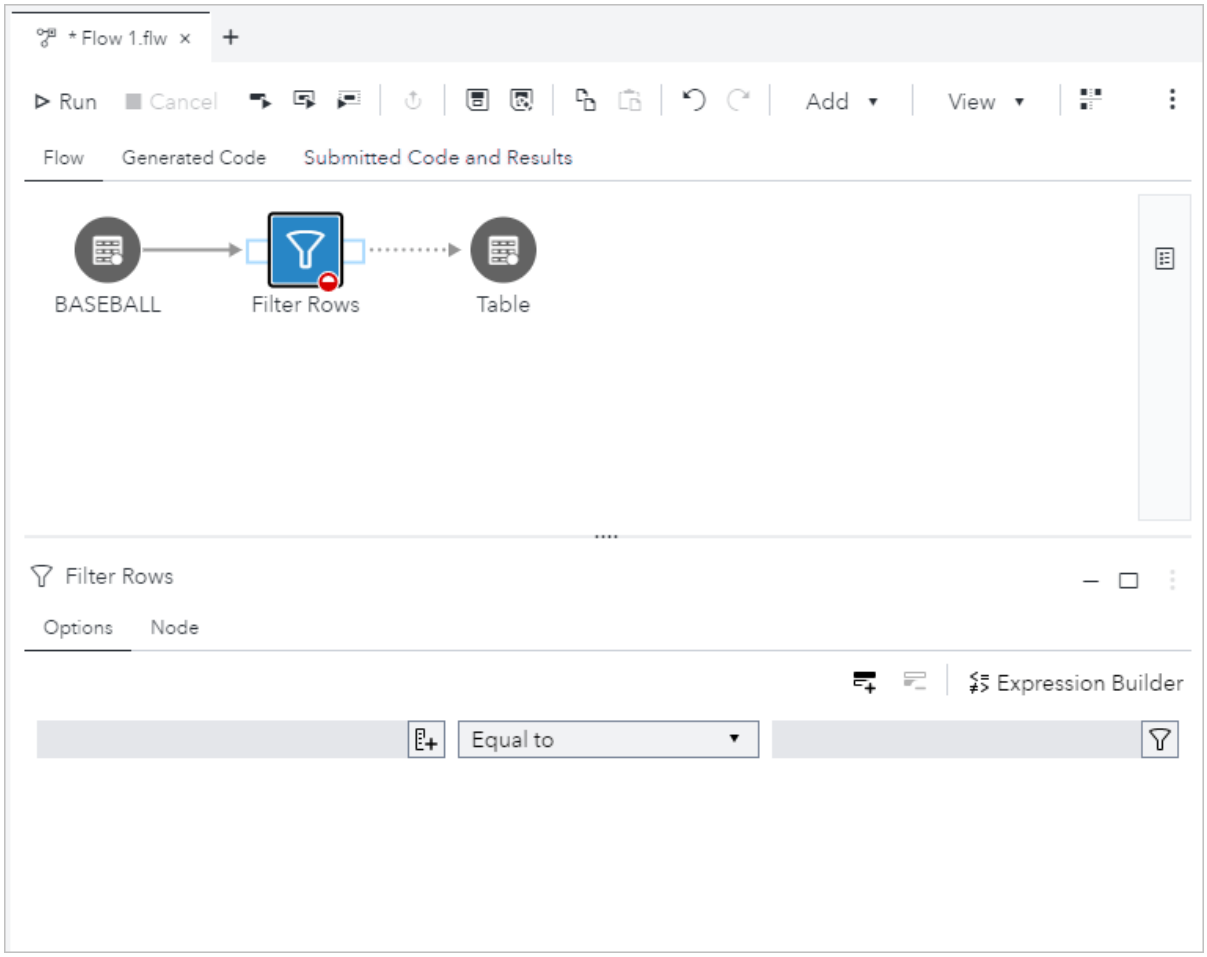
Filter Rows: Step-by-Step Instructions

The Filter Rows step requires one input port and one output port. You can create one or more filter conditions to select a subset of the data that is written to the output table.

1. In the **Steps** section of the navigation pane, expand the **Transform Data** folder, and double-click **Filter Rows**.
2. Connect the input port of the Filter Rows node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.
3. Connect the output port of the Filter Rows node to a data source by using a Table node or another node that requires an input table, such as a Query Node, a SAS Program node, or an Export node.

   Note: If you are using a Table node, click the node, and use the Table Properties tab to make sure that the node specifies a library and name for the table. Any existing data in the output table node is overwritten when you run the Filter Rows step.

4. Click the **Filter Rows** node, and then click the **Options** tab.



5 Click , and select the column that you want to use. Click **OK**.

> **TIP** To use the expression builder to create a condition, click **Expression Builder** on the toolbar for the condition. For more information, see "Building an Expression" in *SAS Studio: User's Guide*.

6 Select a comparison operator from the operator drop-down list. The default value is `Equal to`.

7 If the operator that you have selected requires a value, click . In the Add Filter window, enter or select a value in the **Value** box. To choose from a list of values, click and click **Get Values** in the Select Value window. Select the values that you want to use and click **OK**.

> **TIP** If you want to search for a value in the Select Value window, use the unformatted value.

8 Depending on the data type of the column that you are using in the condition, you can choose from the following options:

- **Match case** – retrieves only rows that match the capitalization of the value that you specify. If this option is not selected, the UPPER function is applied to the expression. This option is not selected by default.
- **Quote string** – encloses values in single quotation marks. This option is selected by default. If you are using a macro variable or other value that is evaluated when the filter is run, you should clear this option.
- **Allow macros** – enables you to enter character values in a filter on a numeric column.

9 Click **Filter** to add values to the condition.

10 To add another row to the condition, click and repeat steps 5–9. If you create more than one comparison expression in your condition, the default relationship between these filter elements is AND. You can change the relationship between filter elements from AND to OR by clicking the relationship drop-down list.

Note: To delete a row from a condition, select the row and click

11 To run the flow, click Run.

Insert Rows

About the Insert Rows Step

You can use the Insert Rows step to insert the rows from an input table into an output table. For example, you can create a flow that uses the Insert Rows step to create and add data to an output table. You can regularly update the data in the output table by scheduling the flow as a job.

The output table must contain one or more columns that have the same name and data type as columns in the input table. SAS Studio automatically generates PROC SQL or PROC FEDSQL code when the step is run.

The Insert Rows step enables you to add rows to an output table in any of the following ways:

- append the rows in the input table to existing rows in the output table
- replace the existing rows in the output table with the rows from the input table
- add the rows from the input table to a new output table

Note: The Insert Rows step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Node Connection Requirements

In order to run the Insert Rows step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	Table node that contains one or more columns that have the same name and data type as columns in the input table

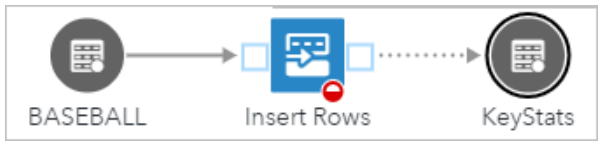
Example: Inserting Rows into a New Output Table from the Sashelp.Baseball Data Set

In this example, you insert rows from the Sashelp.Baseball data set into a new output table named KeyStats. The KeyStats table contains a subset of the columns that are in the Baseball data set.

To create this example:

1. From the main SAS Studio menu, select **New** ⇨ **Flow**.
2. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Insert Rows**.
3. In the **Libraries** section of the navigation pane, expand the Sashelp library. Drag the Baseball data set onto the input port of the Insert Rows node. Drop the data set on the flow canvas when the tooltip on your mouse pointer changes to `Connect to input port`.

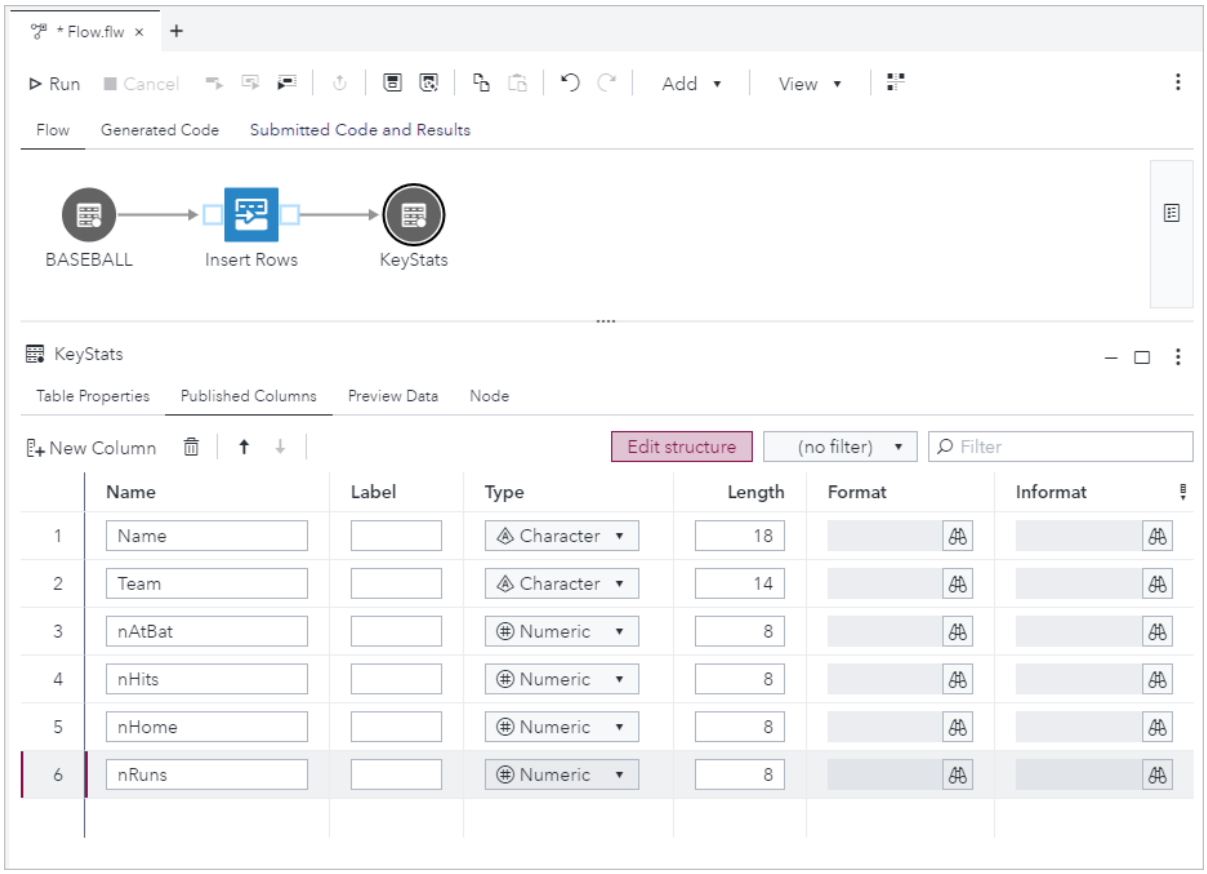
4. On the flow toolbar, click **Add** ⇨ **Table**. Connect the Insert Rows node to the Table node by clicking the Insert Rows output port and dragging the mouse pointer to the Table node.
5. Click the **Table** node. On the **Table Properties** tab, click beside the **Library** box, and select the Work library. In the Table box, enter *KeyStats*.



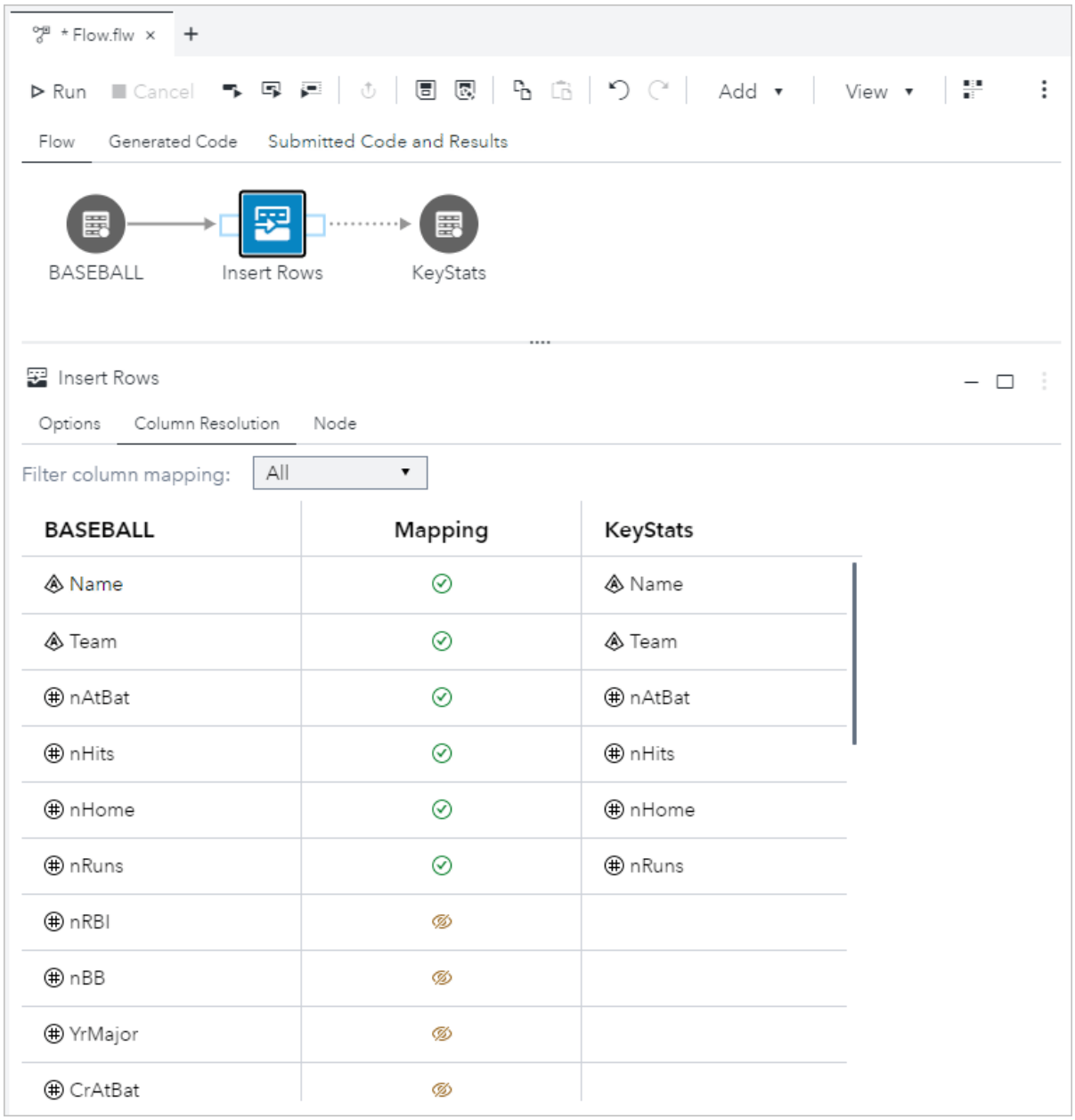
6 Click the **Published Columns** tab to add some columns from the Baseball data set. Change the table to Edit mode by clicking **Edit structure**, and then clicking **Yes** in the confirmation window. Click **New Column** and enter *Name* in the **Name** column. Accept the default data type of `Character` and enter *18* for the Length. Continue adding five more columns by using these column names, data types, and lengths:

Column Name	Data Type	Length
Team	Character	14
nAtBat	Numeric	8
nHits	Numeric	8
nHome	Numeric	8
nRuns	Numeric	8

Note: For information about how to copy the entire column structure of the input table to the output table, see the Tip in "About the Table Node" on page 15.

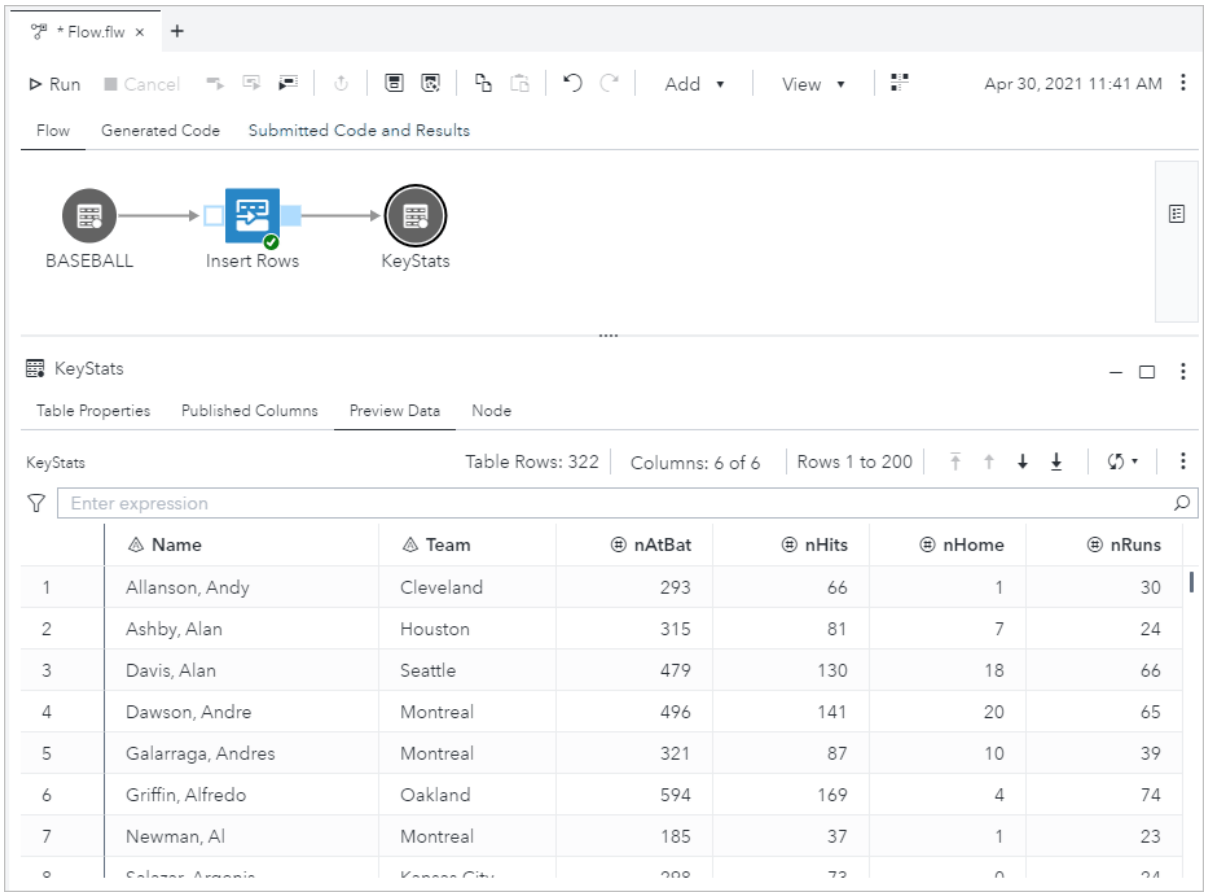


7 Click the **Insert Rows** node, and then click the **Column Resolution** tab. Notice that the columns in the KeyStats table are all mapped to columns in the Baseball data set. There are additional columns in the Baseball data set that do not map to columns in the KeyStats table.



8 To run the flow, click ▷ Run.

Here is the KeyStats output table:

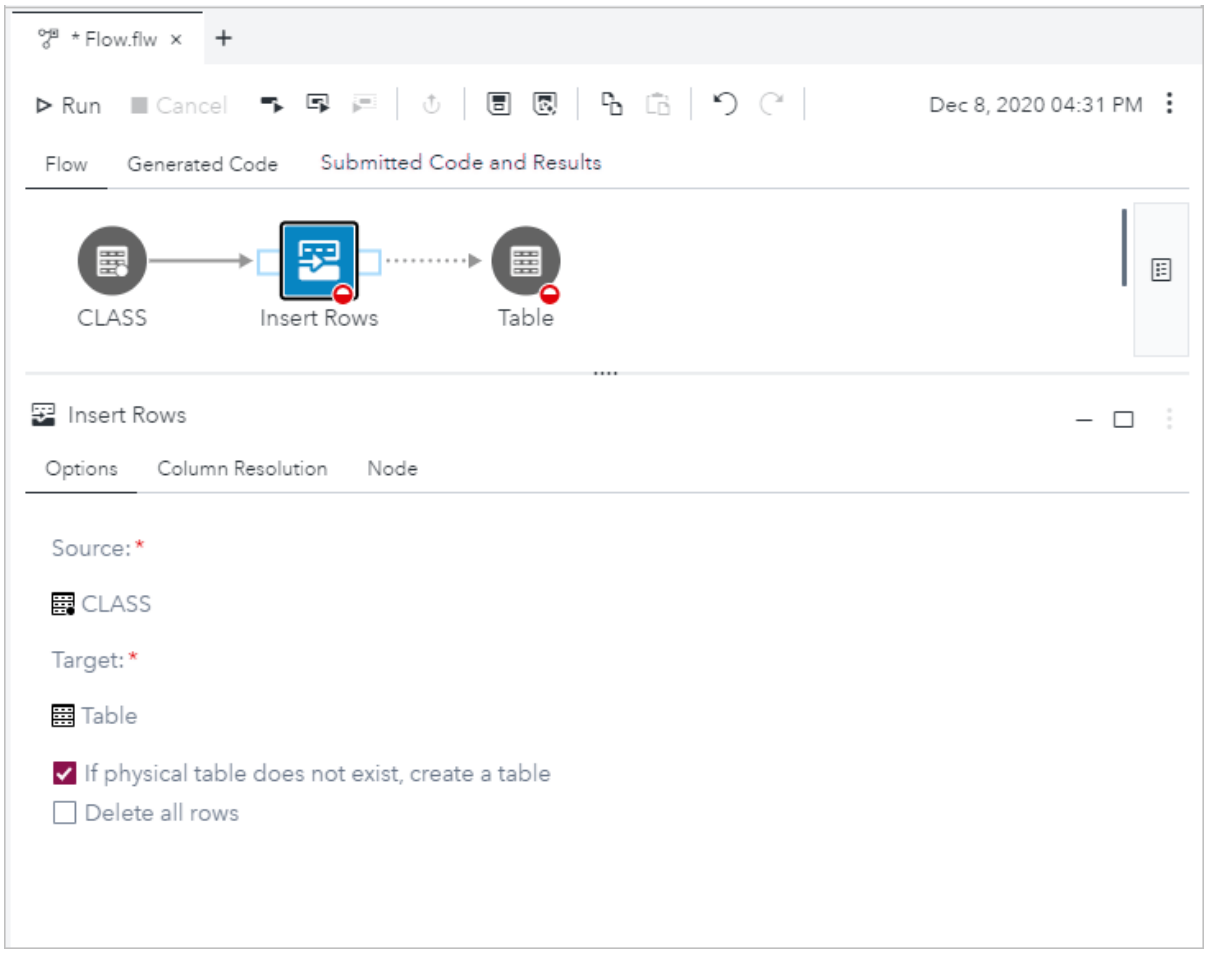


	Name	Team	nAtBat	nHits	nHome	nRuns
1	Allanson, Andy	Cleveland	293	66	1	30
2	Ashby, Alan	Houston	315	81	7	24
3	Davis, Alan	Seattle	479	130	18	66
4	Dawson, Andre	Montreal	496	141	20	65
5	Galarraga, Andres	Montreal	321	87	10	39
6	Griffin, Alfredo	Oakland	594	169	4	74
7	Newman, Al	Montreal	185	37	1	23

Insert Rows: Step-by-Step Instructions

By default, the Insert Rows step appends rows from an input table to the end of the output table. You can also add rows to a new table or choose to delete any existing rows in the output table before you insert the new rows.

1. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Insert Rows**.
2. Connect the input port of the Insert Rows node to a data source by using a Table node or an operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.
3. Connect the output port of the Insert Rows node to a Table node. The Table node must include at least one column with the same name and data type as a column in the input table. The Table node can reference an existing table, or you can create a new table when the step runs.



4 Click the output table node, and use the Table Properties and Published Columns tabs to make sure that the node specifies a library and name for the table and that the node includes one or more columns from the input table. For more information, see "Adding a Table from a SAS Library to a Flow" on page 36.

Note: If the length of a character column in the output table does not match the length of the corresponding column in the input table, the data is truncated when it is inserted in the output table.

TIP When you are inserting columns into a new output table, you can define the columns in the output table manually on the Published Columns tab of the table node, or you can copy the column structure of the input table. For information about how to copy the entire column structure of the input table to the output table, see the Tip in "About the Table Node" on page 15.

5 Click the **Insert Rows** node.

- On the Options tab, use the **If physical table does not exist, create a table** option to create an output table if the table does not already exist. This option is selected by default. If you clear this option, and the table does not already exist when you run the flow, the flow fails with an error.

- If you want to delete all of the existing rows in the output table before the rows from the input table are inserted, select **Delete all rows**. This option is not selected by default.
- On the Column Resolution tab, you can view the column mappings between the input and output tables. Use the **Filter column mapping** drop-down list to filter the mappings that are displayed. You can filter the mappings by the following categories:
  - Successful (✓) - the input column is successfully mapped to a column in the output table. Data for this column is inserted in the corresponding column in the output table.
  - Ignored (⊘) - the input column does not have a corresponding column in the output table. Data for this column is ignored in the output table.
  - Information (ⓘ) - the input column cannot be mapped to a column in the output table. For example, an unresolved column can occur when columns have the same name and different data types or when a column exists in the output table and not the input table.

6 To run the flow, click ▷ Run.

Manage Columns

About the Manage Columns Step

The Manage Columns step enables you to select a subset of columns from an input table and write the columns to an output table. You can use the Manage Columns step to change the names, labels, data types, lengths, formats, informats, and order of columns in the output table. You can also use the Expression Builder to create a new column in the output table. The Manage Columns step can be combined with other steps in which you want to work with only a subset of the columns in a table.

For example, you might want to analyze only the career statistics of players in the Baseball table. You can use the Manage Columns step to create an output table that is based on the Baseball table and includes only the player names, teams, and the column names that begin with “Cr.” Reducing the number of columns in the data that you are analyzing can improve your performance.

You can choose to generate the output as a view instead of a physical table. A view is a stored query that generates a result set when you access the view. Views do not store data. A view is a virtual table and can be used like a table.

Node Connection Requirements

In order to run the Manage Columns step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and name of the output tables by connecting the output port to a Table node. You can also choose to generate the output as a view instead of a physical table. ■ To create a view in the Work library using the output port, click the output port on the Manage Columns node. On the Output Port tab, select **Create a view**. ■ To create a view using a Table node, click the **Table Properties** tab, and select **Create a view**. For more information, see "Adding a Table from a SAS Library to a Flow" on page 36.

Example: Subsetting Columns from the Sashelp.Baseball Data Set

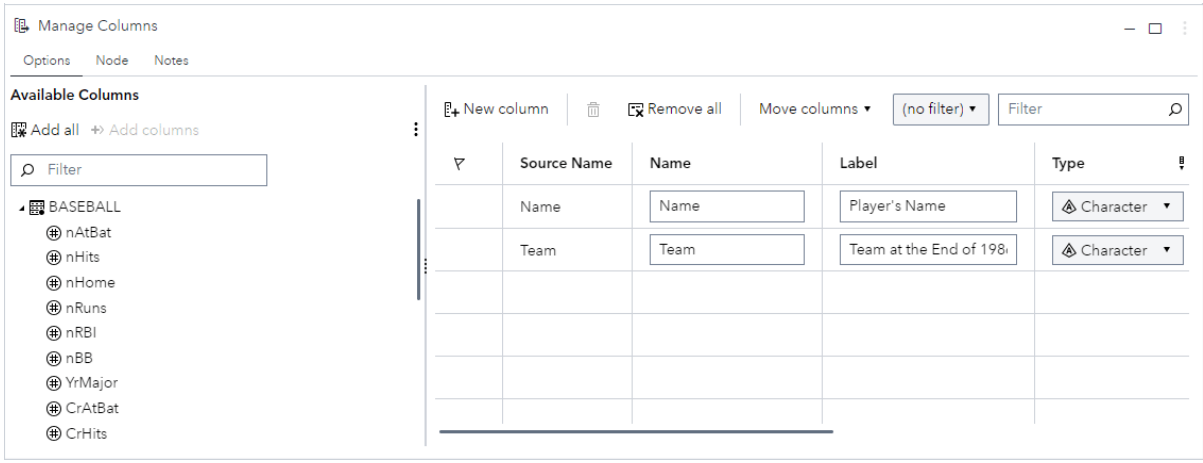
In this example, you select a subset of columns that contain the career statistics of players from the Sashelp.Baseball data set and create a new output table named CareerStats. The CareerStats table contains a subset of the columns and all of the rows that are in the Sashelp.Baseball data set.

To create this example:

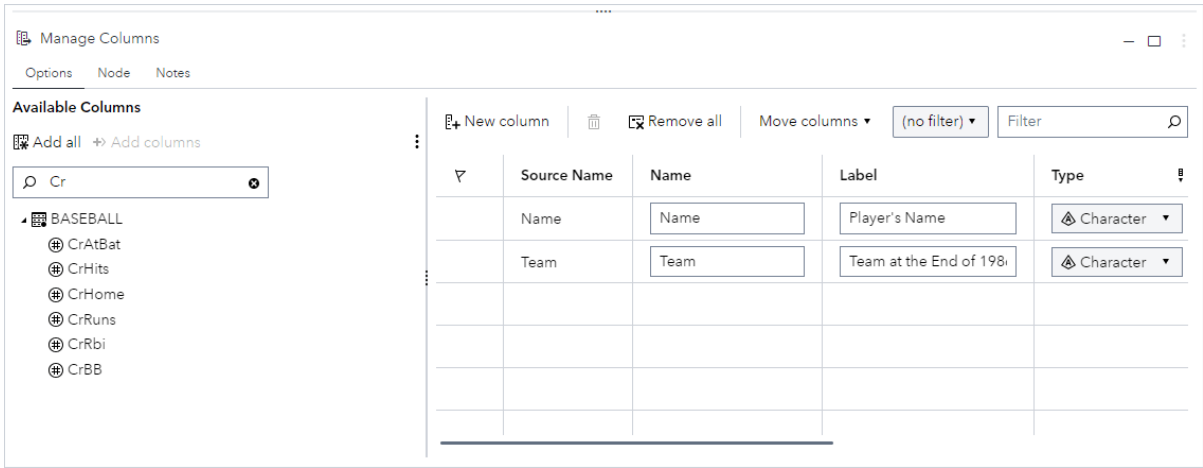
1. From the main SAS Studio menu, select **New** ⇨ **Flow**.
2. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Manage Columns**.
3. In the **Libraries** section of the navigation pane, expand the Sashelp library. Drag the Baseball data set onto the input port of the Manage Columns node. Drop the data set on the flow canvas when the tooltip on your mouse pointer changes to `Connect to input port`.

4. On the flow toolbar, click **Add** ⇨ **Table**. Connect the Manage Columns node to the Table node by clicking the Manage Columns output port and dragging the mouse pointer to the Table node.
5. Click the Table node. On the Table Properties tab, click  beside the **Library** box, and then select the Work library. In the Table box, enter `CareerStats` and click **OK**.

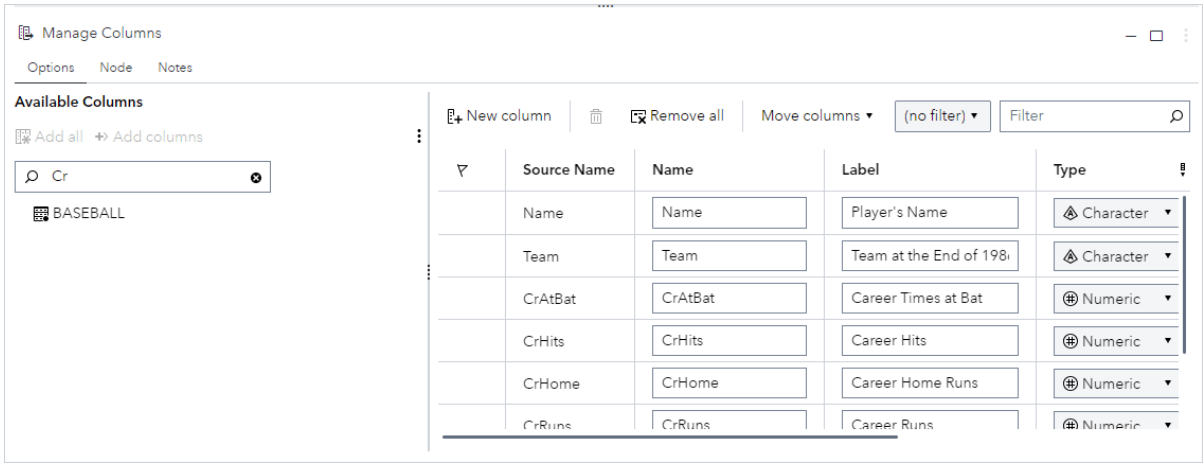
6 Click the Manage Columns node to specify columns for the output table. On the Options tab, select the `Name` and `Team` columns in the Available Columns area and click **Add columns**.



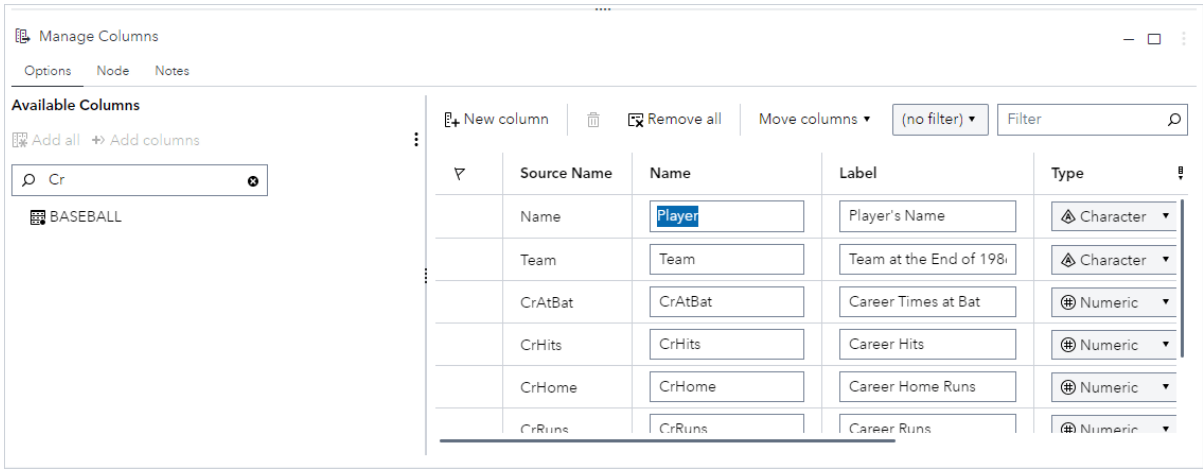
7 In the Available Columns area, enter *Cr* in the Filter box. The list of columns is filtered and displays only the career statistics columns that start with `Cr`.



8 Click **Add all** to add all of the visible columns to the list of columns for the output table.

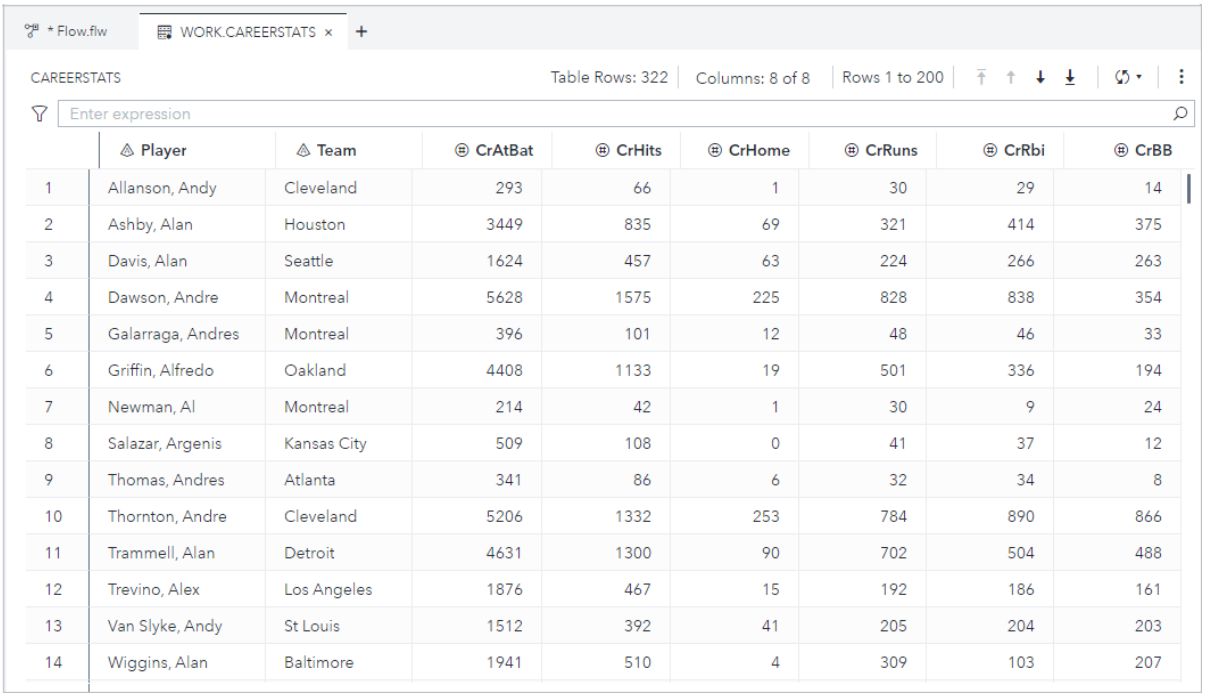


9 Change the Name column to Player by entering *Player* as the new name value.



10 To run the flow, click ▷ Run.

Here is the output table that includes all of the rows from the Sashelp.Baseball data set and only the Player, Team, and career statistics columns.



* Flow.flw | WORK.CAREERSTATS ×

CAREERSTATS | Table Rows: 322 | Columns: 8 of 8 | Rows 1 to 200

Enter expression

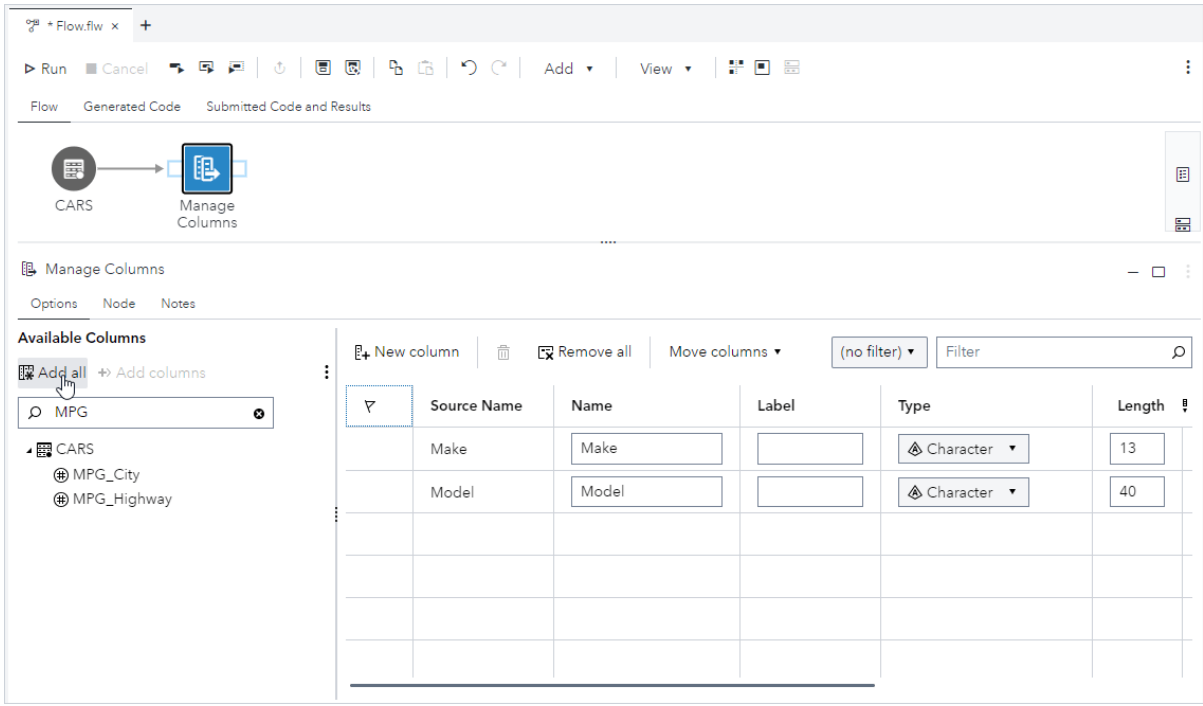
	Player	Team	CrAtBat	CrHits	CrHome	CrRuns	CrRbi	CrBB
1	Allanson, Andy	Cleveland	293	66	1	30	29	14
2	Ashby, Alan	Houston	3449	835	69	321	414	375
3	Davis, Alan	Seattle	1624	457	63	224	266	263
4	Dawson, Andre	Montreal	5628	1575	225	828	838	354
5	Galarraga, Andres	Montreal	396	101	12	48	46	33
6	Griffin, Alfredo	Oakland	4408	1133	19	501	336	194
7	Newman, Al	Montreal	214	42	1	30	9	24
8	Salazar, Argenis	Kansas City	509	108	0	41	37	12
9	Thomas, Andres	Atlanta	341	86	6	32	34	8
10	Thornton, Andre	Cleveland	5206	1332	253	784	890	866
11	Trammell, Alan	Detroit	4631	1300	90	702	504	488
12	Trevino, Alex	Los Angeles	1876	467	15	192	186	161
13	Van Slyke, Andy	St Louis	1512	392	41	205	204	203
14	Wiggins, Alan	Baltimore	1941	510	4	309	103	207

Manage Columns: Step-by-Step Instructions

The Manage Columns step requires one input port and one output port. You must select at least one column for the output table.

1. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Manage Columns**.
2. Connect the input port of the Manage Columns node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.
3. Click the **Manage Columns** node, and then use the Options tab to specify one or more columns for the output table. To add one or more columns, select the columns that you want to add to the output table, and click **Add columns**. To add all of the columns that are currently displayed in the Available Columns area, click **Add all**.

> **TIP** You can specify filter criteria in the filter box, and then click **Add all** to add all of the columns that match your filter criteria.



4 In the right pane of the Options tab, you can modify the columns that are selected for the output table in the following ways:

- To add a new column to the output table, click **New column**. Use the Expression Builder to define the new column. For more information, see "Building an Expression" in *SAS Studio: User's Guide*. The new column is added to the bottom of the list of columns.
- To remove columns from the right pane, select one or more columns and click 🗑. To remove all columns that are currently displayed in the right pane, click **Remove all**.
- To change the order in which the columns appear in the output table, select the column that you want to move. Click **Move columns** and select the appropriate option. The order in which the columns are listed in the right pane of the Options tab determines the order in which the columns appear in the output table.
- To filter the columns that are displayed, click the filter drop-down list and select the filter criteria. You can also enter filter criteria for the Source Name, Name, and Label columns in the filter box.

  Note: Filters affect only which columns are displayed in the right pane. All columns in the right pane are added to the output table even if they are not displayed because of a filter.

- Use the appropriate boxes to change the name, label, data type, length, format, or informat.

5 To run the flow, click ▷ Run.

Note: You can choose to generate the output as a view instead of a physical table.

- To create a view in the Work library using the output port, click the output port on the Manage Columns node. On the Output Port tab, select **Create a view**.
- To create a view using a Table node, click the **Table Properties** tab, and select **Create a view**. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Mask Data

About the Mask Data Step

Use the Mask Data step to perform data masking by using three different obfuscation methods: masking, hashing, or substitution.

- Masking and hashing are one-way encryption methodologies and prevent you from analyzing the obfuscated data.

  Hashing is the transformation of data (known as a message) into a short fixed-length value (known as a digest) that represents the original string. Hashing is used to index and retrieve items in a database because it is faster to find the item using the hashed key than using the original value. Hashing functions are useful when you are developing shell scripts for software installation, file comparison, and detection of file corruption and tampering.
- Substitution enables you to analyze obfuscated data without compromising data integrity. The custom step enables you to implement substitution with a lookup table. You can specify a key column and a value column for each column that you want to substitute. As a result, the original data can be restored by referencing the lookup table.

Mask Data: Step-by-Step Instructions for Masking Data

Step 1: Add the Mask Data Step and Connect a Source Table

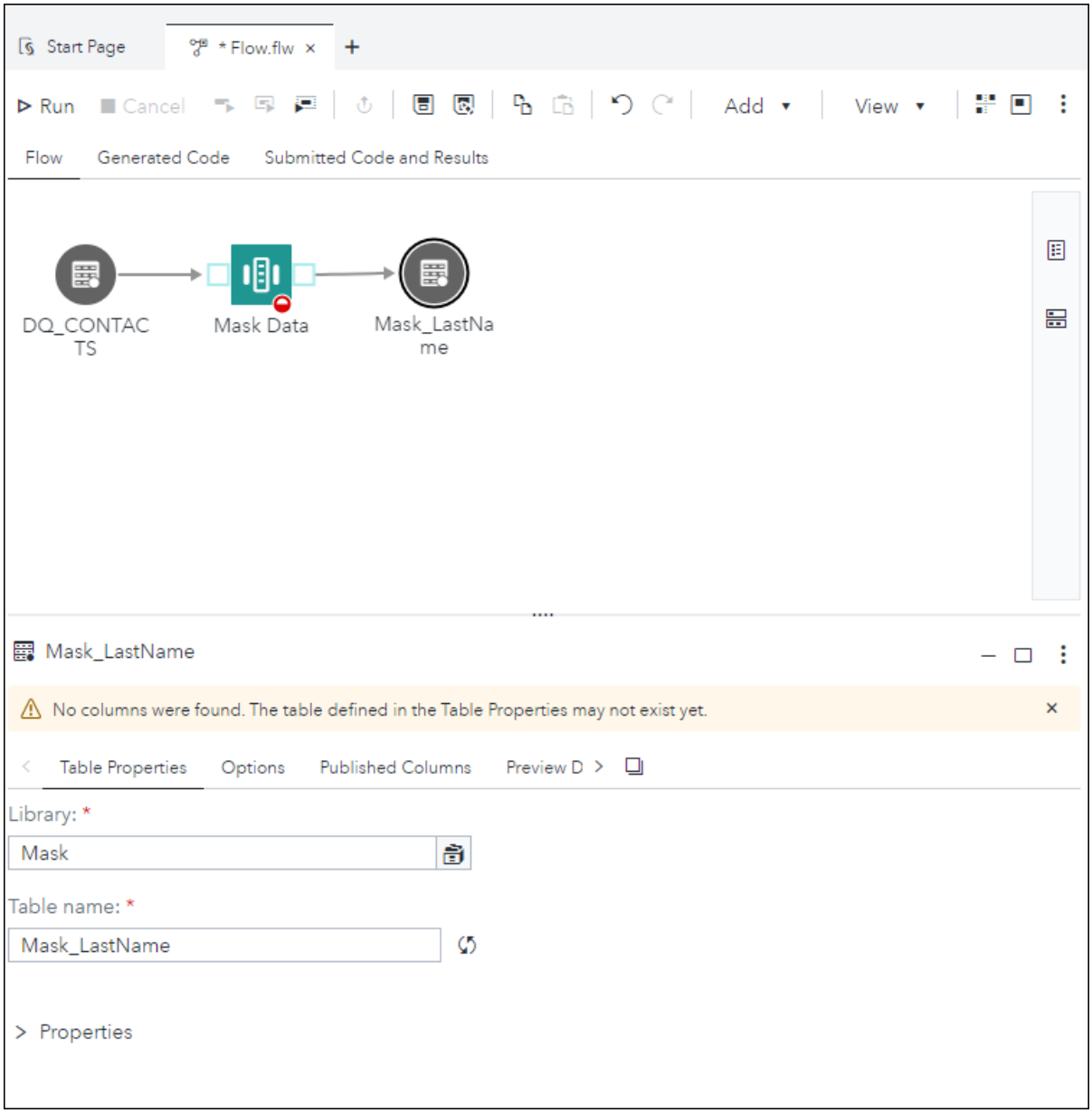
1. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Mask Data** to add the step to your flow.
2. Connect the input port of the Mask Data node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS

Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

3 To save the results from the Mask Data step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table. (In this example, the output table is named Mask_LastName.)

Next, connect the output port of the Mask Data node to the table node.

Note: The output table does not have any columns until you run the flow.



Step 2: Select the Masking Method for Data Obfuscation

1 Select the QKB locale. You can use the global locale or select a locale.

2 Select **Masking** as the obfuscation method.

Here are the valid masking methods.

Note: All of these definitions are available if you select a QKB locale. If you select a custom locale, the available definitions depend on the selected locale.

- **Anonymize E-mail Domain** - This definition replaces the email domain with nodomain.com. For example, if the input is john.smith@xyz.com or john.smith@xyz.org, the output is john.smith@nodomain.com.

  Note: Any text that is not part of the email address is removed. For example, if the input is John Smith <john.smith@xyz.org>, the output is john.smith@nodomain.com.

- **Mask Address Except Street Type** - This definition replaces every digit with a zero and every alphanumeric character except those that refer to the street type with the * (asterisk) character. For example, if the input is `52 Commerce Street`, the output is `00 ******** Street`.

  Note: The address strings must be in a valid format for the country of the locale. Invalid input entries might not be masked.

- **Mask All Characters** - This definition replaces every alphanumeric, punctuation, and symbol character with the * (asterisk) character. For example, if the input is `John Smith 123`, the output is `**** ****** ***`.
- **Mask All Characters Except First and Last** - This definition replaces all of the characters between the first character and the last character with three * (asterisk) characters. For example, if the input is `John Smith 123`, the output is `J***3`.

  Note: If the input string has three or fewer characters, the entire string is replaced with three * characters. White space is considered a character.

- **Mask Digits** - This definition replaces every digit with the * (asterisk) character. For example, if the input is `John Smith 123`, the output is `John Smith ***`.
- **Mask E-mail Domain** - This definition replaces every non-space character in the complete email domain with the * character. For example, if the input is `john.smith@xyz.com` or `john.smith@xyz.org`, the output is `john.smith@*******`.

Note: Any text that is not part of the email address is removed.

- **Mask E-mail Mailbox** - This definition replaces every non-space character in the mailbox portion of the email address with the * (asterisk) character. For example, if the input is `john.smith@xyz.com` or `John Smith <john.smith@xyz.com>`, the output is `**********@xyz.com`.

  Note: Any text that is not part of the email address is removed.

- **Mask E-mail Mailbox Except First and Last Characters** - This definition replaces all the characters between the first and last in the mailbox portion of the email address with three * (asterisk) characters. For example, if the input is `john.smith@xyz.com`, the output is `j***h@xyz.com`.

  Note: If the mailbox portion of the email address has three or fewer characters, the entire mailbox is replaced with three * characters. For example, if the input is `a@xyz.com`, the output is `***@xyz.com`. White space is considered a character.

- **Mask E-mail Sub-Domain** - This definition replaces every non-space character in the email sub-domain with the * (asterisk) character. For example, if the input is `john.smith@xyz.com`, the output is `john.smith@***.com`.

  Note: Any text that is not part of the email address is removed. For example, if the input is `John Smith <john.smith@xyz.com>`, the output is `john.smith@***.com`.

- **Mask Letters** - This definition replaces every letter with the * (asterisk) character. For example, if the input is `John Smith 123`, the output is `**** ***** 123`.

- **Mask Partial String** - This definition replaces every alphanumeric character except the last four with the * (asterisk) character. For example, if the input is `IT8U676539TX99`, the output is `**********TX99`.

  Note: This string must be longer than four characters. Any strings that contain four or fewer characters are not masked.

- **Mask Phone Number** - This definition replaces every alphanumeric character except the last four alphanumeric characters of the base number with the * (asterisk) character. For example, if the input is `(507) 334-1871 ext151`, the output is `*** ***1871 ***`.

  Note: This definition is designed for single phone numbers in a valid format for the country of the locale. Invalid input might not be masked. There might be some format changes, such as parenthesis removal, that occurs prior to masking.

- **Truncate Word to Initial** - This definition removes all characters except the first letter or digit of each alphanumeric sub-string. For example, if the input is `John Smith 123`, the output is `J S 1`.

Note: Sub-strings are delimited by punctuation, symbols, and white space with the exception of periods and commas in decimal numbers.

3 Select the column to mask.

4 Specify whether to replace an existing column or create a column for the masked data.

By default, the masked data replaces the input column, and the column name remains the same. If a column is created, the new column name is *InputColumnName*_Masked.

Step 3: Set the Options

On the **Options** tab, select **Show detailed log information** to include debugging information in the log.

Step 4: Specify the Output Data

Select the **Output** tab to set these options.

1 (Optional) Select **Replace existing output table with the same name** to replace an existing table with the new table. Any output tables are replaced by default.

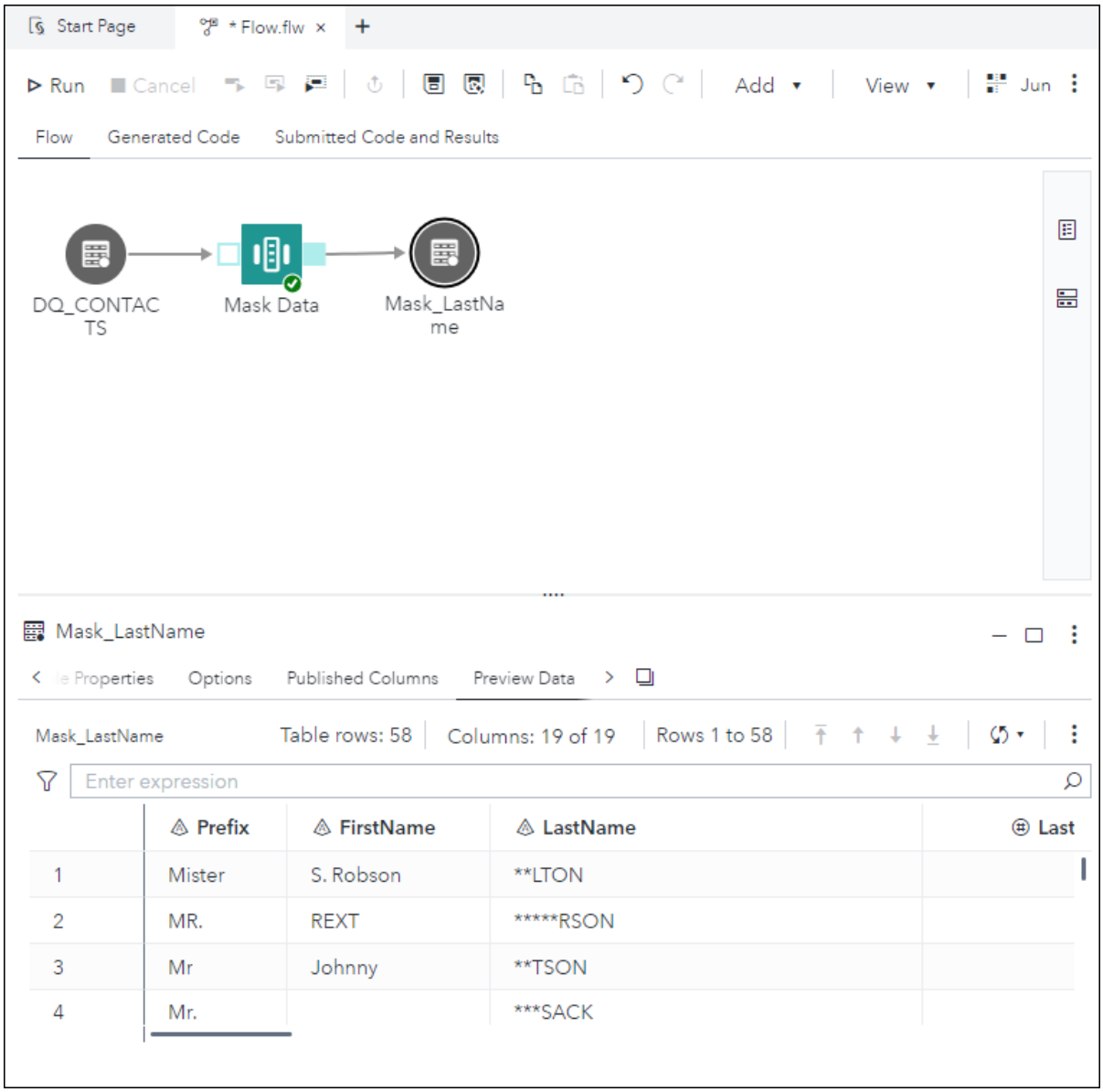
2 To create a CAS table from the output data, select these options:

- **Promote to a global in-memory table**
- **Save to persist the table on disk**
- **File format**

Step 5: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.



Mask Data: Step-by-Step Instructions for Hashing Data

Step 1: Add the Mask Data Step and Connect a Source Table

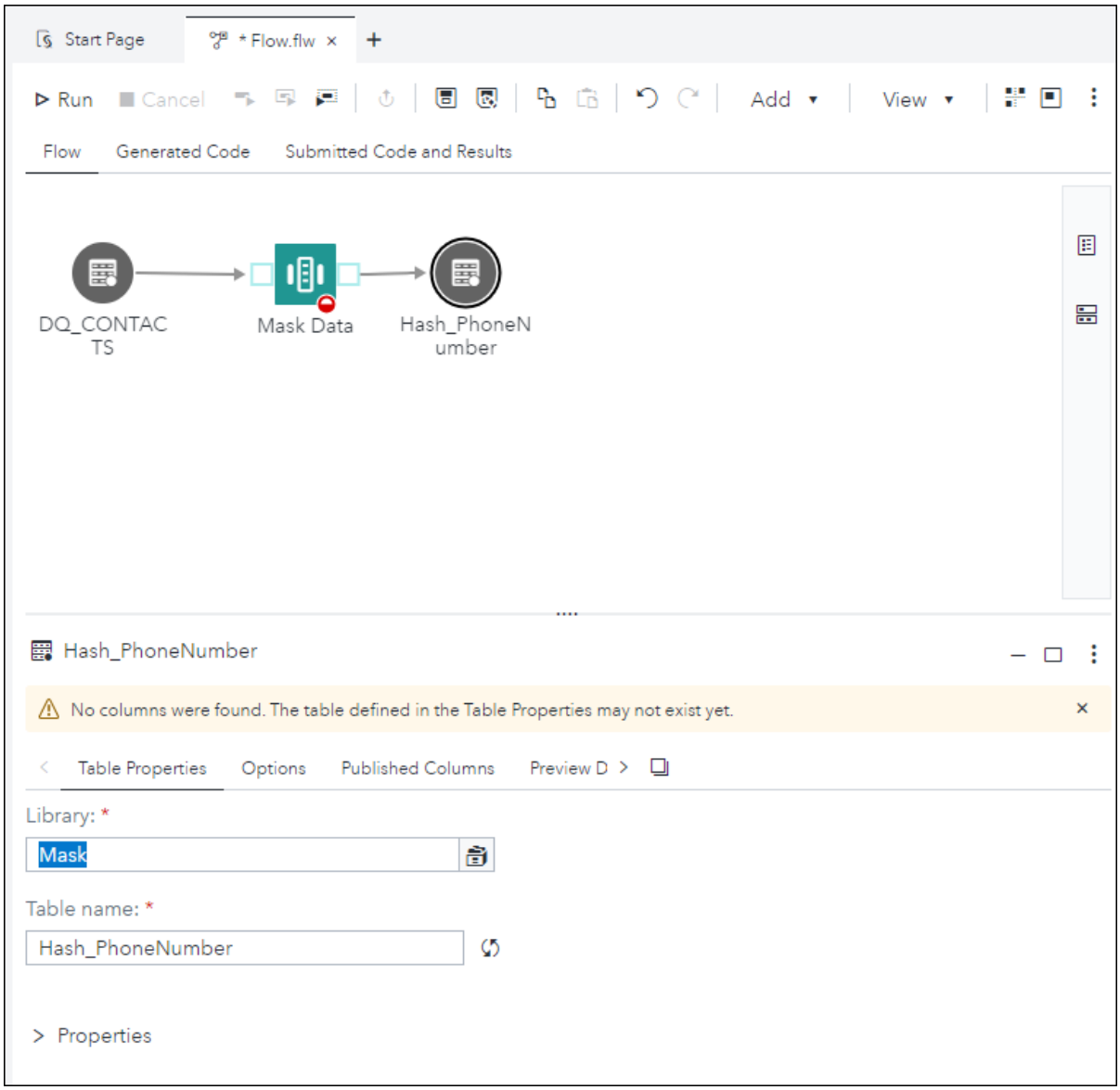
1 In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Mask Data** to add the step to your flow.

2 Connect the input port of the Mask Data node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

3 To save the results from the Mask Data step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table . (In this example, the output table is Hash_PhoneNumber.)

Next, connect the output port of the Mask Data node to the table node.

Note: The output table does not have any columns until you run the flow.



Step 2: Select the Hashing Method for Data Obfuscation

Note: The Hashing method ignores the QKB locale.

1. Select **Hashing** as the obfuscation method.
2. Select the hashing method.
3. Select the hash column.
4. Specify whether to replace an existing column or create a column for the hashed data.

   By default, the hashed data replaces the input column, and the column name remains the same. If a column is created, the new column name is *InputColumnName*_Hashed.
5. (Optional) Select **Transcode data to UTF-8**.
6. If you want to authenticate your data by using a shared secret key, select **Use Hash-based Message Authentication Code (HMAC)**. Then specify the secret key. The strength of the HMAC depends on the size of the secret key.
7. (Optional) To mask additional columns, set the options under the Data Obfuscation heading.

Step 3: Set the Options

On the **Options** tab, select **Show detailed log information** to include debugging information in the log.

Step 4: Specify the Output Data

Select the **Output** tab to set these options.

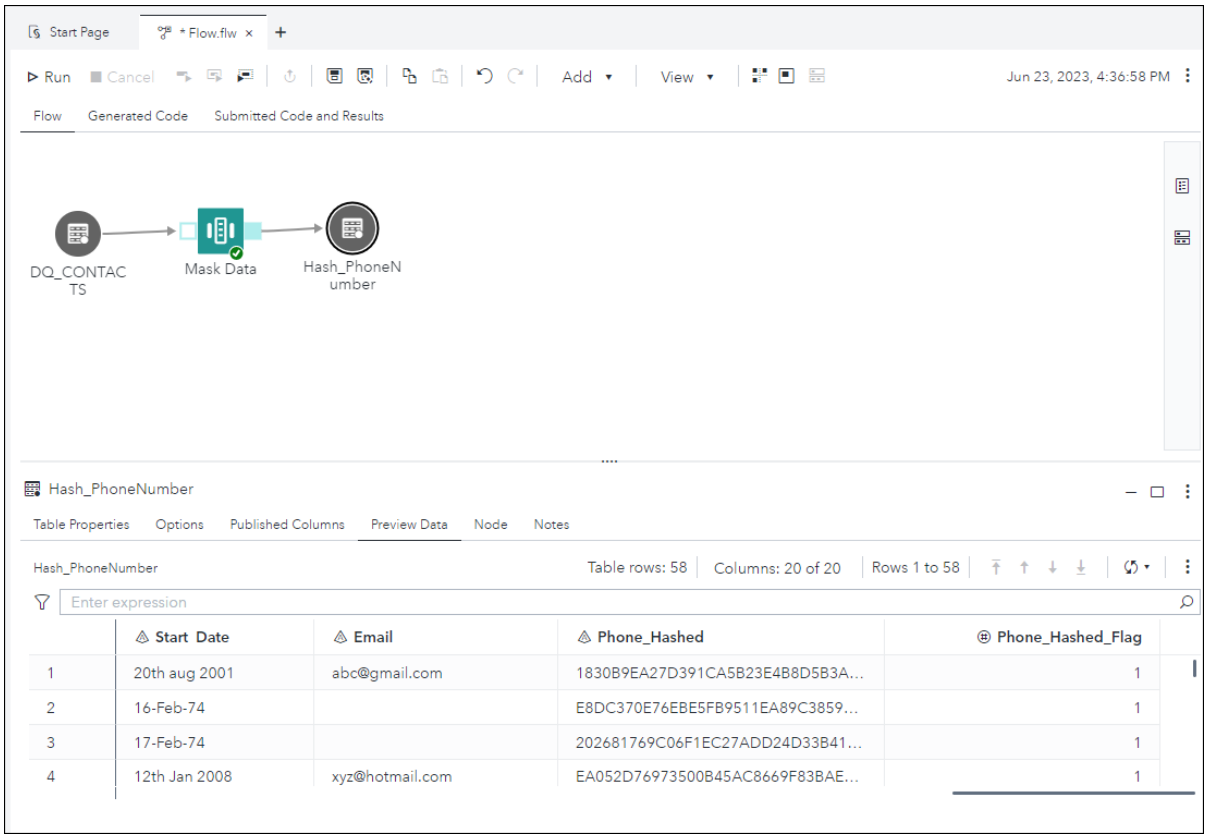
1. (Optional) Select **Replace existing output table with the same name** to replace an existing table with the new table. Any output tables are replaced by default.
2. To create a CAS table from the output data, select these options:
   - **Promote to a global in-memory table**
   - **Save to persist the table on disk**
   - **File format**

Step 5: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.

In this example, the new Phone_Hashed column contains the masked data. The Phone_Hashed_Flag column contains a value of `1` if the value from the Phone column was successfully hashed. The Phone_Hashed_Flag column contains a `0` if the value from the Phone column could not be hashed.



Mask Data: Step-by-Step Instructions for Substituting Data

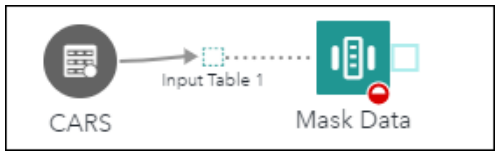
Step 1: Add the Mask Data Step and Connect a Source Table

To use the substitution method, you must have an input table and a lookup table.

1 In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Mask Data** to add the step to your flow.

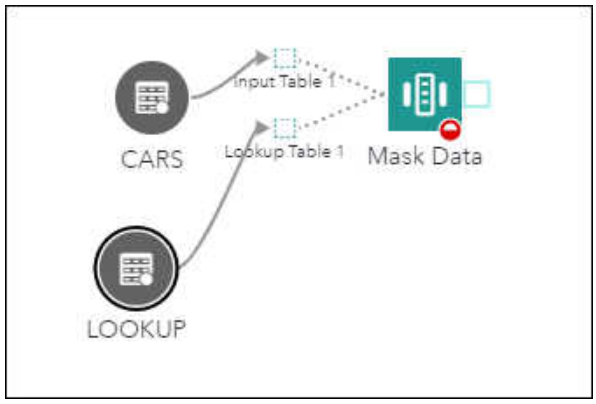
2 Connect the input port of the Mask Data node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. (In this example, the input table is Cars.) For more information, see “Connecting Nodes” on page 12.

3 Right-click the Mask Data node and select **Expand**.



4 Right-click the Mask Data node again and select **Add input port**. The flow now includes a port called Lookup Table 1.

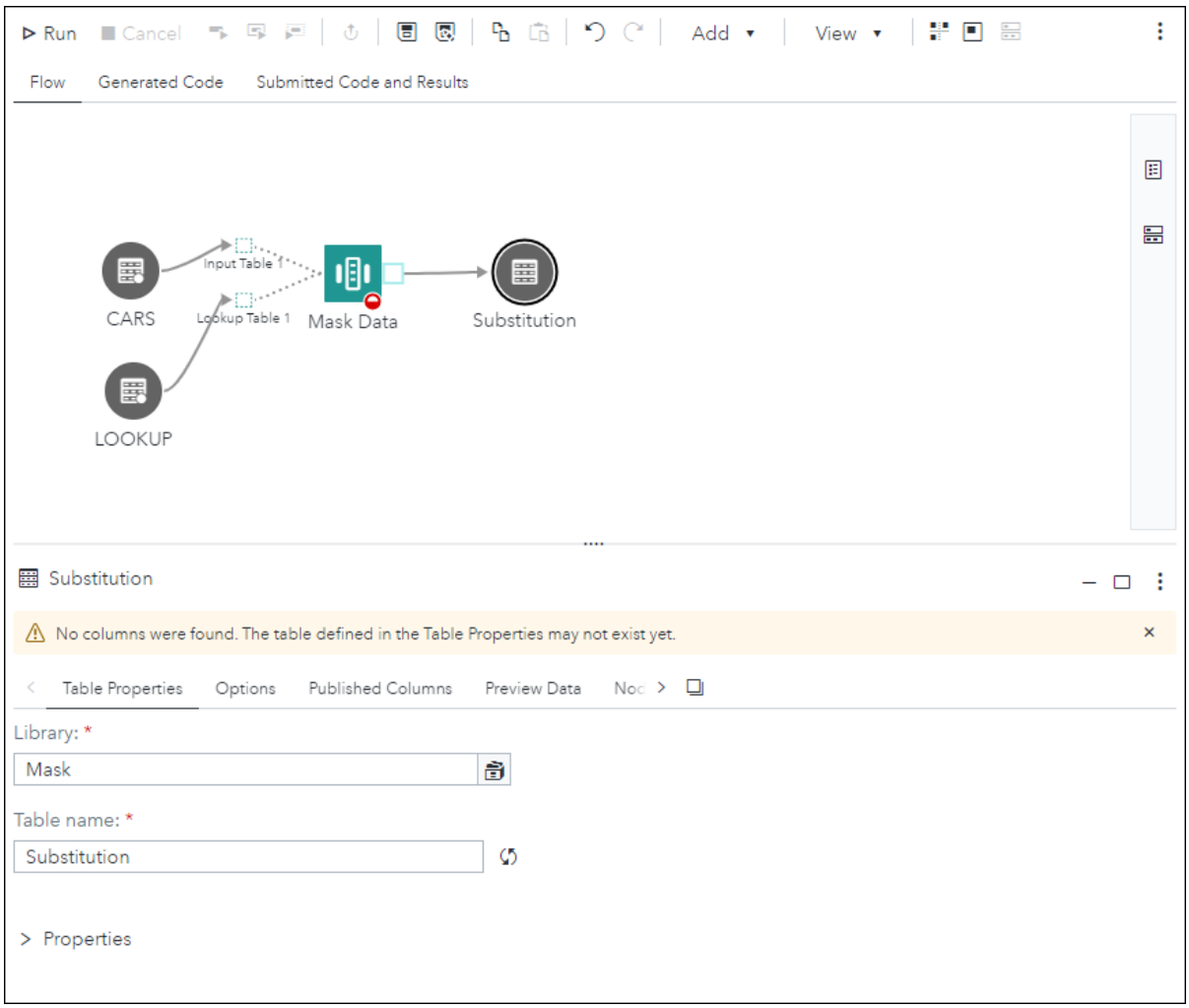
5 Add the lookup table to the flow and connect it to the Lookup Table 1 port. (In this example, the lookup table is Lookup.)



6 To save the results from the Mask Data step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table. (In this example, the output table is Substitution.)

Next, connect the output port of the Geocode Data node to the table node.

Note: The output table does not have any columns until you run the flow.



Step 2: Select the Substitution Method for Data Obfuscation

Note: The Substitution method ignores the QKB locale.

1. Select **Substitution** as the obfuscation method.
2. Assign a column in the lookup table to the **Key column** role.
3. Assign a column in the lookup table to the **Value column** role.
4. Assign a column to the **Substitute column** role.
5. If the value in the input column does not match any value in the key column, specify whether to return the original value or a missing value. By default, the original value is returned.

6 Specify whether to replace the existing column or create a column.

   By default, the substitute column replaces the input column and uses the same name. If a column is created, the name of the new column is *InputColumnName*_Sub.

7 (Optional) To mask additional columns, set the options under the Data Obfuscation heading.

Step 3: Set the Options

On the **Options** tab, select **Show detailed log information** to include debugging information in the log.

Step 4: Specify the Output Data

Select the **Output** tab to set these options.

1 (Optional) Select **Replace existing output table with the same name** to replace an existing table with the new table. Any output tables are replaced by default.

2 To create a CAS table from the output data, select these options:

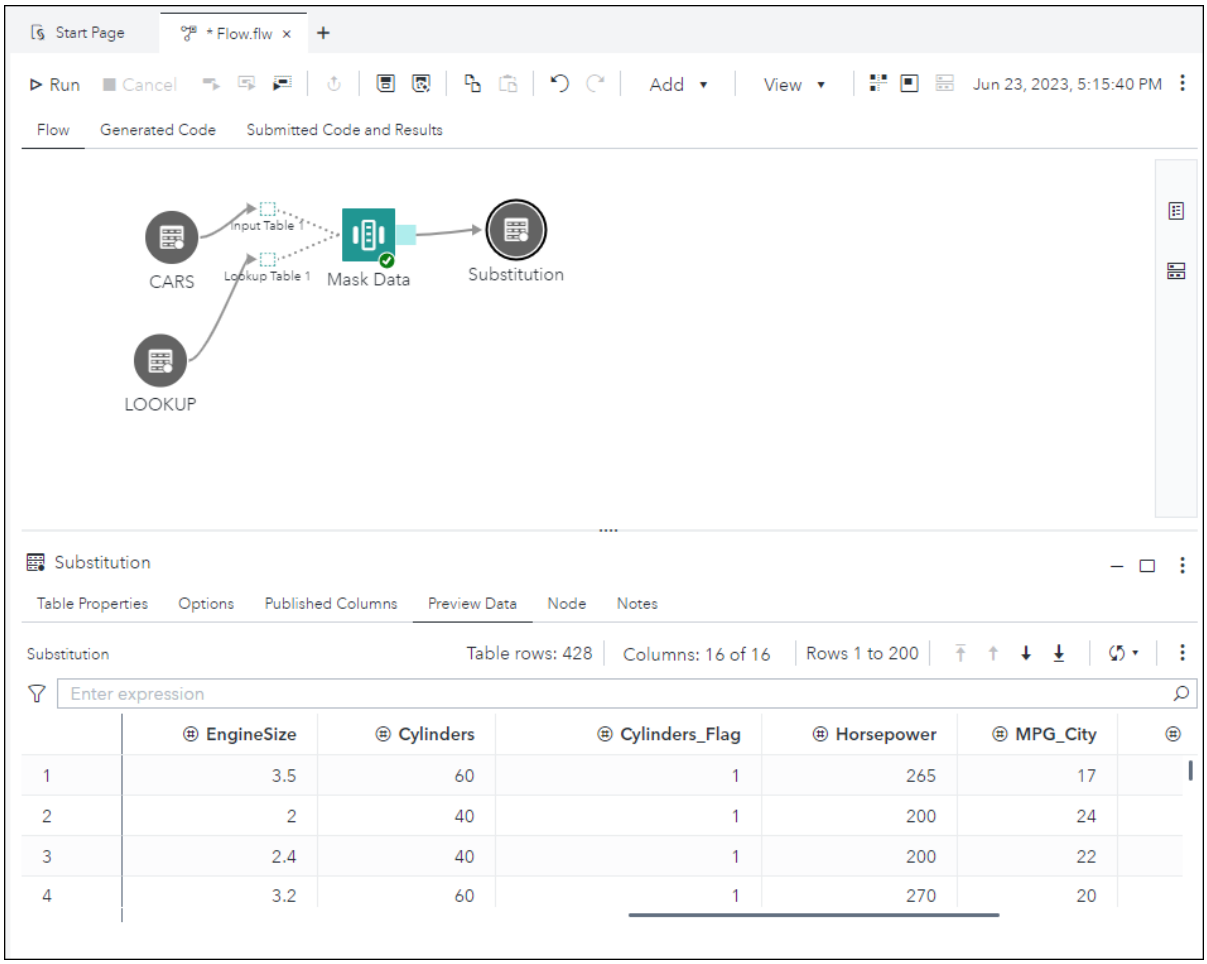
- **Promote to a global in-memory table**
- **Save to persist the table on disk**
- **File format**

Step 5: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.

In this example, the values in the Cylinders column were replaced. The Cylinders_Flag column contains a `1` if the value in the Cylinders column was replaced. If the value was not replaced, the value in the Cylinders_Flag column is `0`.



Query

About the Query Step

Use the Query step to select, join, filter, and sort columns from a table in a flow. By default, output for the Query step is written to a table in the Work library. Query node output ports can be connected to a Table node, a Query node, or a SAS Program node.

Note: If the query results are generated using PROC SQL, you can choose to create a view instead of a physical table. A view is a stored query that generates a result set when you access the view. Views do not store data. A view is a virtual table and can be used like a table. For information about using PROC SQL and PROC FedSQL to generate queries, see "Generating a FedSQL Query" in *SAS Studio: User's Guide*.

> **TIP** You can also use the Branch Rows step and the Filter Rows step to select a subset of rows from an input table. For more information, see "Understanding the Steps That Subset Data" on page 60.

Node Connection Requirements

In order to run the Query step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and the name of the output data by connecting the output port to a Table node. If the query results are generated using PROC SQL, you can choose to create a view instead of a physical table. ■ To create a view in the Work library using the output port, click the output port on the Query node. On the Output Port tab, select **Create a view**. ■ To create a view using a Table node, click the **Table Properties** tab, and select **Create a view**. For more information, see "About the Table Node" on page 15.

Query: Step-by-Step Instructions

The Query step requires one input port and one output port. You must select at least one column for the output. The Query node interface is mostly the same as the interface for stand-alone queries. For more information, see "Working with Queries" in *SAS Studio: User's Guide*.

1 Select **Add** ⇨ **Query**.

2 Connect the Query node to a data source by using a Table node, an Import node, a Query node, or a SAS Program node.

> **TIP** You can quickly add a Query node to the flow and connect the node to an input data source by right-clicking the input data source and selecting **Add a query**. Valid input data sources include a Table node or the output ports of the operational nodes, which include Import, SAS Program, and Query.

3 Click the **Query** node, click the **Options** tab, and then select output columns.

4 By default, the output data is written to a temporary table in the Work library. You can specify the library and the name of the output data by connecting the output port to a Table node.

Note: If the query results are generated using PROC SQL, you can choose to create a view instead of a physical table.

- To create a view in the Work library using the output port, click the output port on the Query node. On the Output Port tab, select **Create a view**.
- To create a view using a Table node, click the **Table Properties** tab, and select **Create a view**. For more information, see "About the Table Node" on page 15.

Rank Data

About the Rank Data Step

The Rank Data step computes ranks for one or more numeric variables across the rows in a table and includes the ranks in an output table.

An example of when you might use the Rank Data step is to rank product sales. In this case, the ranking variable would show the order of product sales. The product with the highest number of sales would be ranked first.

Note: The Rank Data step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Node Connection Requirements

In order to run the Rank Data step, you must create connections to the input and output ports of the node as indicated here:

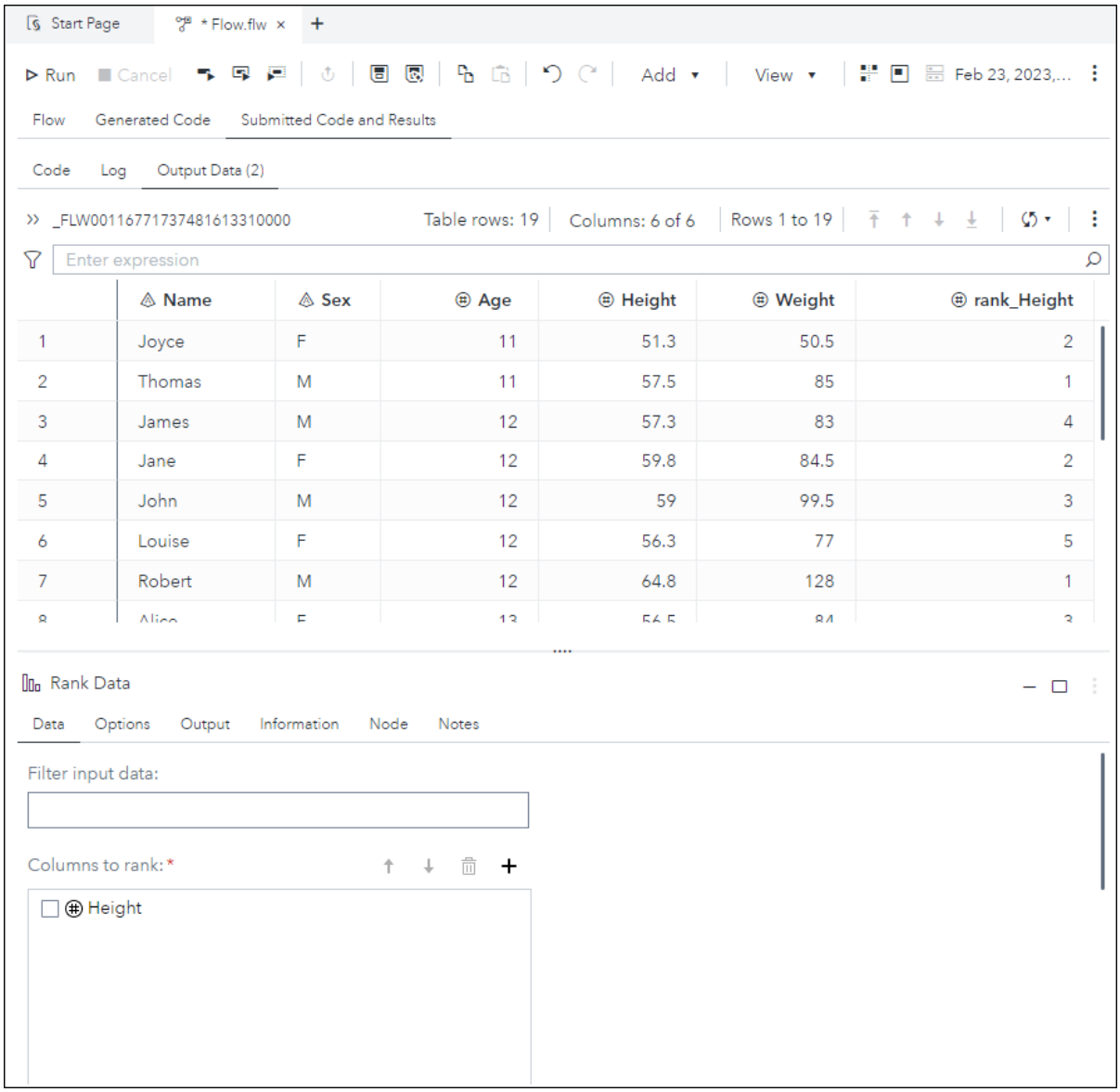
Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and the name of the output tables by connecting the output port to a Table node. For

Input Port	Output Port
SAS Program node, or an Import node	more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Example: Ranking Students by Height within Age

1. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Rank Data** to add the step to your flow.
2. Add the Sashelp.Class table to your flow. Connect the input port of the Rank Data node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the Rank Data step, click the Rank Data node.
4. On the **Data** tab, assign columns to these roles:
   - For the **Columns to rank** role, assign **Height**.
   - For the **Rank by** role, assign **Age**.
5. Open the **Options** tab. From the **Rank order** drop-down list, select **Largest to smallest**.
6. Run the flow.

Open the **Submitted Code and Results** tab to view the output data. This table contains the additional rank_Height column, which shows where that student ranks within her age group. For example, in the 11-year-old age group, Joyce is ranked number 2. In the 12-year-old age group, Louise is ranked number 5.



Note: In this example, the output data is saved to the temporary Work library. To save this data to a permanent table, connect a table to the output port of the Rank Data node.

Rank Data: Step-by-Step Instructions

Step 1: Assign Data to Roles

1 (Optional) To filter the input data source, enter the filter expression in the **Filter** field.

2 Assign at least one variable to the **Columns to rank** role. The rankings column is given the name `rank_`*column-name*, where *column-name* is the name of the original column.

3 (Optional) Assign one or more variables to this role. The input table is sorted by the selected column or columns and rankings are calculated within each group.

Step 2: Set the Method Options

On the **Options** tab, you can set these options for the method.

1 Select the ranking method. Here are the valid values:

Ranks
partitions the original values into 100 groups, in which the smallest values receive a percentile value of 0 and the largest values receive a percentile value of 99.

Quantiles
partitions the original values into one of these quantiles.

- **Percentiles** partitions the data into 100 groups, in which the smallest values receive a percentage value of 0 and the largest values receive a percentage value of 99.
- **Deciles** partitions the original values into 10 groups, in which the smallest values receive a decile value of 0 and the largest values receive a decile value of 9.
- **Quartiles** partitions the original values into four groups, in which the smallest values receive a quartile value of 0 and the largest values receive a quartile value of 3.
- **N-tile groups** partitions the original values into *n* groups, in which the smallest values receive a value of 0 and the largest values receive a value of *n*–1. Specify the value of *n* in the **Number of groups** box.

Fractional ranks
computes the fractional ranks by using either a denominator of N or N+1. A denominator of N computes fractional ranks by dividing each rank by the number of observations that have nonmissing values of the ranking variable. A denominator of N+1 computes fractional ranks by dividing each rank by the denominator *n*+1, where *n* is the number of observations that have nonmissing values of the ranking variable.

Percentages
divides each rank by the number of observations that have nonmissing values of the variable and multiplies the result by 100 to get a percentage.

Normal scores of ranks
computes normal scores from the ranks. The resulting variables appear normally distributed.

Note: If you set the **If values are tied, use** option, the Rank Data step computes the normal score from the ranks based on non-tied values and applies the ties specification to the resulting score.

Savage scores of ranks
computes Savage (or exponential) scores from the ranks.

Note: If you set the **If values are tied, use** option, the Rank Data step computes the Savage score from the ranks based on non-tied values and applies the ties specification to the resulting score.

2 Specify how to compute normal scores or ranks for tied values.

Default method
assigns the default method for your ranking method. If you select **Percentages** or **Fractional** ranks as the ranking method, the high value is the default. For all other ranking methods, the mean is the default.

Mean of ranks
assigns the mean of the corresponding rank or normal scores.

High rank
assigns the largest of the corresponding ranks or normal scores.

Low rank
assigns the smallest of the corresponding ranks or normal scores.

Dense rank (ties are the same rank)
computes scores and ranks by treating tied values as a single-order statistic. For the default method, ranks are consecutive integers that begin with the number 1 and end with the number of unique, nonmissing values of the variable that is being ranked. Tied values are assigned the same rank.

3 Use the **Rank order** option to list the values from smallest to largest or from largest to smallest.

Step 3: Specify the Output Data

On the **Output** tab, you can set these options.

1 To replace an existing output table with the same name, select **Replace existing output table**.

2 To replace the original column with the ranked columns, clear the **Create new variables for the ranked variables** check box. If this option is selected, the output table contains the original columns as well as the ranked columns.

The ranked column is given the name `rank_`*column-name*, where *column-name* is the name of the original column.

3 Use the **Specify data to print** to print none of the data, a subset of the observations, or all the observations.

Remove Duplicates

The Remove Duplicates node enables you to remove duplicate rows from an input data source.

- If your input data and output data are in a SAS library, the step runs the SORT procedure by using the NODUPKEY option to remove the duplicate data.
- If your input data and output data are in a CAS library, the step uses the simple.GroupByInfo action to remove the duplicate data.

To remove duplicates from a data source:

1. Click the **Steps** section of the navigation pane.
2. Expand the **Transform Data** folder and double-click the **Remove Duplicates** node.
3. From the **Libraries** section of the navigation pane, drag the table that you want to sort onto the **Remove Duplicates** node. When the tooltip changes to `Connect to input port`, drop the file.
4. Click the **Remove Duplicates** node. Select the **Remove duplicates across all columns** check box. Next, select the columns to group by to determine whether there are duplicate rows in the data.
5. Specify these output options:
   - **Replace existing output table with the same name**
   - For CAS output tables:
     - Specify whether to promote the table and save the table.
     - (Optional) Specify the file format (in lowercase) for the output table.

Select Random Sample

About the Select Random Sample Step

The Select Random Sample task creates an output table that contains a random sample of the rows in the input table.

You might use this task when you need a subset of the data. For example, suppose you want to audit employee travel expenses in an effort to improve the expense reporting procedure and possibly reduce expenses. Because you do not have the

resources to examine all expense reports, you can use statistical sampling to objectively select expense reports for audit.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Node Connection Requirements

In order to run the Select Random Sample step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and name of the output tables by connecting the output port to a Table node. For more information, see "Adding a Table from a SAS Library to a Flow" on page 36.

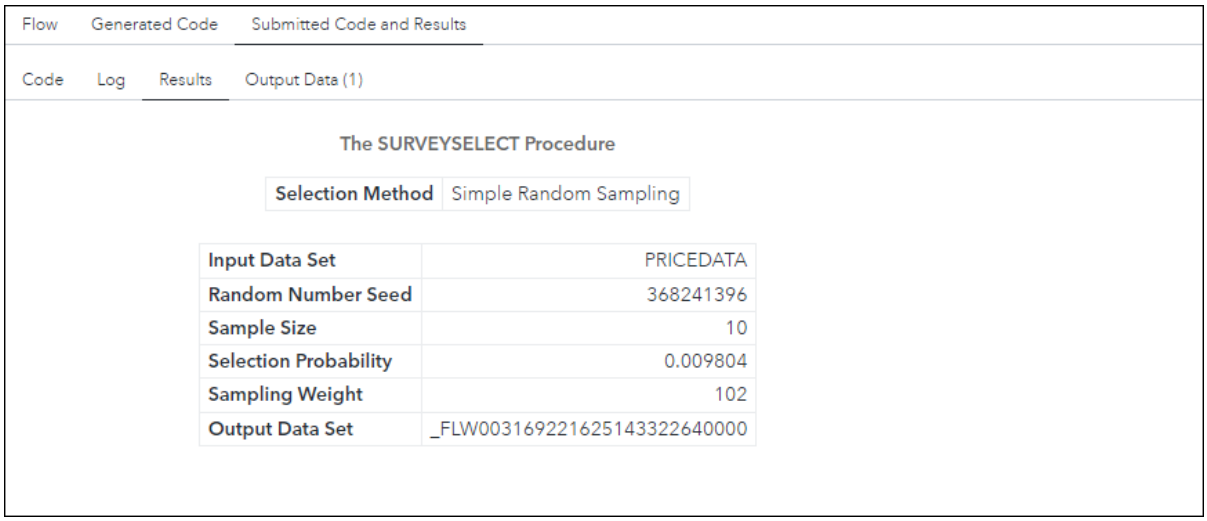
Example: Creating a Random Sample of the Sashelp.Pricedata Data Set

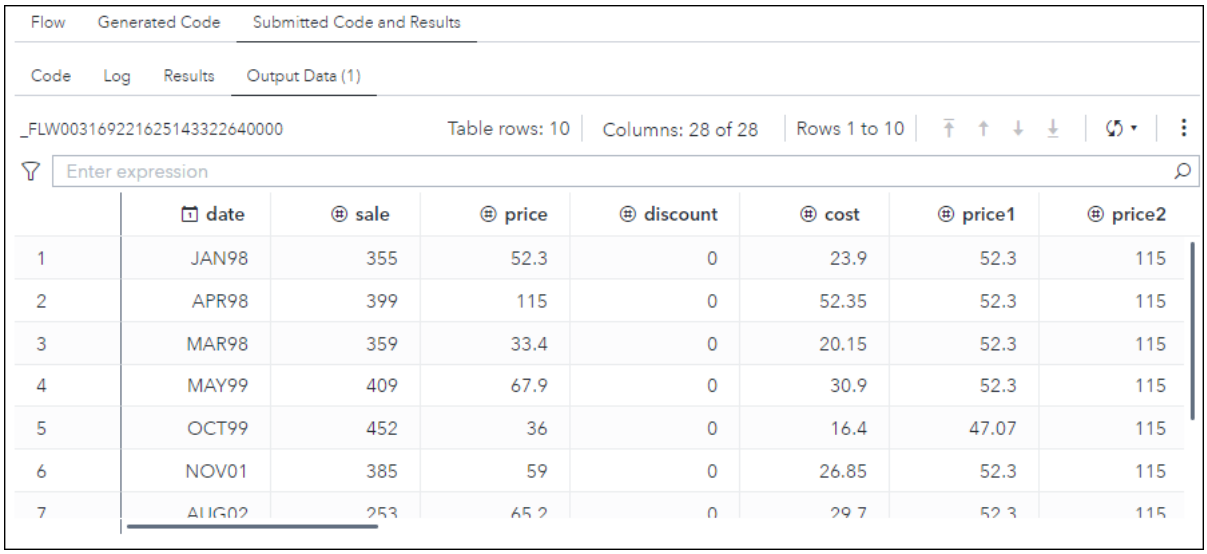
In this example, you create a subset of the data in the Sashelp.Pricedata data set.

To create this example:

1. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Select Random Sample**.
2. Add the Sashelp.Pricedata table to your flow. Connect the input port of the Select Random Sample node to the data source. For more information, see "Connecting Nodes" on page 12.
3. To set the options for the Select Random Sample step, click the Select Random Sample node.
4. On the **Options** tab, specify `10` as the sample size.
5. Run the flow.

Open the **Submitted Code and Results** tab to view the results.





Select Random Sample: Step-by-Step Instructions

Step 1: Add the Select Random Sample and Connect a Source Table

1 In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Select Random Sample** to add the step to your flow.

2 Connect the input port of the Select Random Sample node to a data source by using a Table node or another node that creates an output table such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

Step 2: Assign Data to Roles

To run the step, you must specify a sampling method and a sample size on the **Options** tab. No roles are required.

To partition the input table into mutually exclusive, non-overlapping subsets, you can specify columns to the **Stratify by** role.

Each stratum is defined by a set of values of the strata variables, and each stratum is sampled separately. The complete sample is the union of the samples that are taken from all the strata.

Note: If you do not assign any variables to this role, then the entire input table is treated as a single stratum.

This example shows how the total sample size among the strata is in proportion to the size of the stratum. For this example, the variable GENDER has possible values of M and F, and the variable VOTED has possible values of Y and N. If you assign both GENDER and VOTED to the **Stratify by** role, then the input table is partitioned into four strata: males who voted, males who did not vote, females who voted, and females who did not vote.

The input table contains 20,000 rows, and the values are distributed as follows:

- 7,000 males who voted
- 4,000 males who did not vote
- 5,000 females who voted
- 4,000 females who did not vote

Therefore, the proportion of males who voted is 7,000/20,000=0.35 or 35%. The proportions in the sample should reflect the proportions of the strata in the input table. For example, if your sample table contains 100 observations, then 35% of the values in the sample must be selected from the males who voted to reflect the proportions in the input table.

Step 3: Set the Sampling Options

On the **Options** tab, you can specify these options.

1 Specify the method to use when sampling the data by using the **Sampling Method**drop-down list.

   - **With replacement** specifies that when a row is selected, it is removed from eligibility for subsequent selections. This removal makes it impossible to select the same row more than once.
   - **Without replacement** specifies that when a row is selected, it remains eligible for subsequent selections. This eligibility makes it possible to select the same row more than once. You can specify how multiple selections of the same row are recorded in the output table.

2 Use the **Sample type** drop-down list to specify whether to specify the same size as number or rows or as a percentage of input rows. For example, if you specify 3% of rows and there are 400 input rows, then the resulting sample has 12 rows.

Note: If you assign columns to the **Stratify by** role, then the sample size specification applies to each stratum rather than to the entire input table.

3 **Sample type**

4 Select **Specify the random seed** to specify the initial seed for the generation of random numbers. This value must be an integer. If you do not specify a random seed number, then a seed that is based on the system clock is used to produce the sample.

5 To generate a table that includes the seed that was used to produce the sample, select **Generate a sample selection summary**. By specifying this same seed later with the same input table, you can reproduce the same sample.

Step 4: Specify the Output Options

On the **Output** tab, you can specify these options.

1 To replace an existing output table, select **Replace existing output table**.

2 To include all variables in the output, select **Include all variables in the output data**. By default, all variables are included in the output table. However, you can select the variables to include in the output.

3 Specify whether to display the output data. You can choose to display none of the data, all of the data, or a subset of the output data.

Step 5: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

Open the **Submitted Code and Results** tab to view the results.

Sort

About the Sort Step

The Sort node enables you to order your data by the values of one or more columns. By default, output for the Sort node is written to a table in the Work library. Sort node output ports can be connected to Table, Query, and SAS Program nodes.

Node Connection Requirements

In order to run the Sort step, you must create connections to the input and output ports of the node as indicated here:

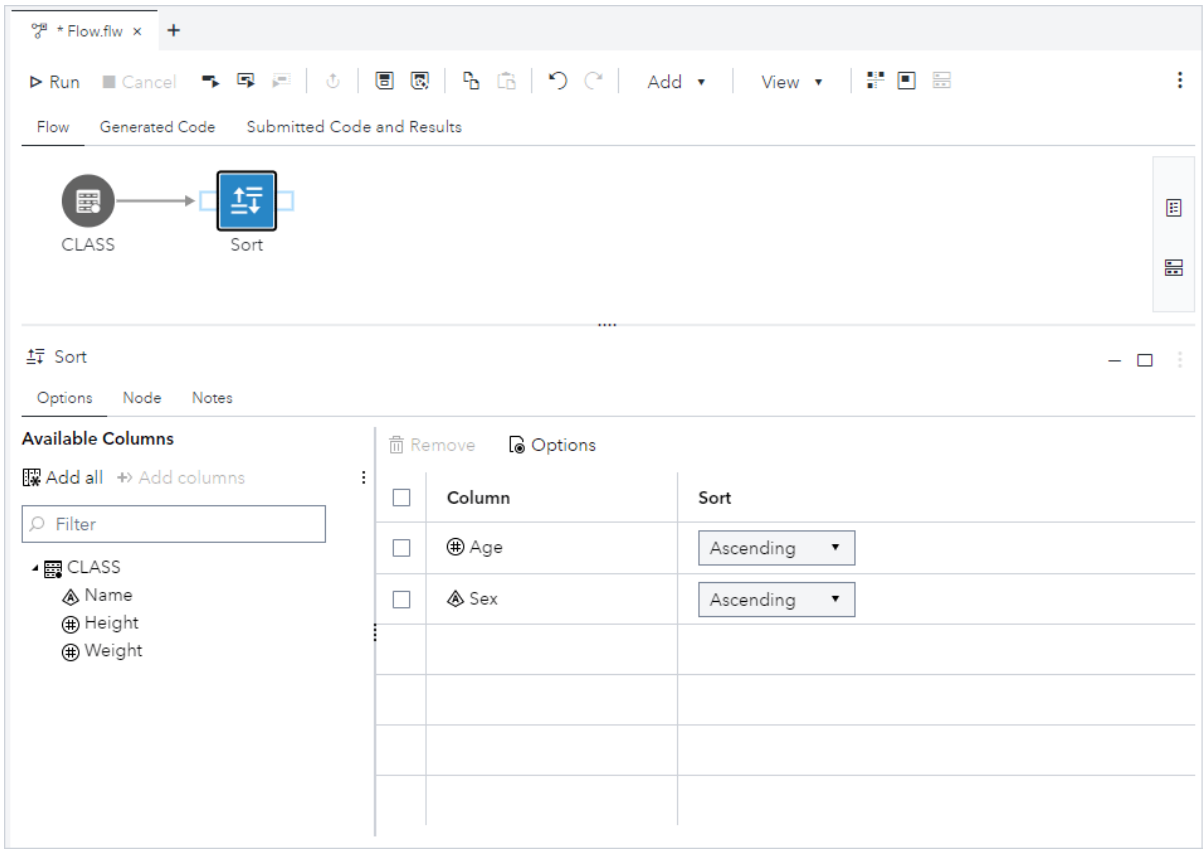
Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and the name of the output tables by connecting the output port to a Table node. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Sort: Step-by-Step Instructions

To sort a table:

1. Click the **Steps** section of the navigation pane.
2. Expand the **Transform Data** folder and double-click the **Sort** node.
3. From the **Libraries** section of the navigation pane, drag the table that you want to sort onto the **Sort** node. When the tooltip changes to `Connect to input port`, drop the file.
4. Click the **Sort** node, and use the **Options** tab to add one or more columns. To add one or more columns, select the columns that you want to add to the sort, and then click **Add columns**. To add all of the columns, click **Add all**.

The order in which the columns are listed on the **Options** tab determines which variable is the primary sort key, which variable is the secondary sort key, and so on. The primary sort key is always the first variable in the list.



5 To specify the following optional arguments to the sort criteria, click **Options** on the toolbar.

- **Specify the output order** - specifies the order of the rows in the output data set.
- **Duplicate rows** - specifies whether to keep all rows that are in the output table, including duplicate rows.
- **Force redundant sorting** - sorts and replaces the data set and destroys all user-created indexes for the data set.
- **Use tags to sort large data** - reduces temporary disk usage by storing only the BY variables and observation numbers in temporary files.

Click **OK** to save your changes.

Split Columns

About the Split Columns Step

The Split Columns step creates an output table by splitting the unique combination of values of the selected columns in the input table into multiple columns. You can use the output table to individually analyze the columns that contain multiple rows of the input table.

This functionality is useful when you have a table in which one column contains multiple observations for different subgroups and you want to split the subgroup measures into separate columns. For example, you could transpose a column that contains the monthly temperature readings for various locations across a geographic region. The output table would contain the monthly temperature readings for each location in a column for each month.

Note: The Split Columns step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirements for Node Connection

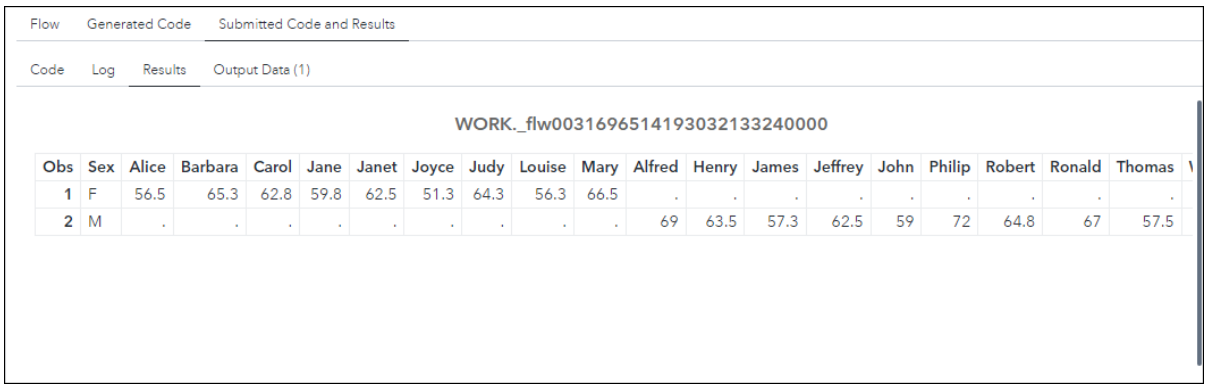
In order to run the Split Columns step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and name of the output tables by connecting the output port to a Table node. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Example: Splitting the Height Column in the Class Table

1. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Split Columns** to add the step to your flow.
2. Add the Sashelp.Class table to your flow. Connect the input port of the Split Columns node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the Split Columns step, click the Split Columns node.
4. On the **Data** tab, assign columns to these roles:
   - To the **Column to split** role, assign the **Height** variable.
   - To the **Formatted identifier values** role, assign the **Name** variable.
   - Expand the Additional Roles heading. To the **Group analysis by** role, assign the **Sex** variable.
5. On the **Output** tab, set these options:
   - Clear the **Use column name prefix** check box.
   - From the **Specify data to print** drop-down list, select **All data**.
6. Run the flow.

Open the **Submitted Code and Results** tab to view the results.



Obs	Sex	Alice	Barbara	Carol	Jane	Janet	Joyce	Judy	Louise	Mary	Alfred	Henry	James	Jeffrey	John	Philip	Robert	Ronald	Thomas
1	F	56.5	65.3	62.8	59.8	62.5	51.3	64.3	56.3	66.5	.	.	.	.	.	.	.	.	.
2	M	.	.	.	.	.	.	.	.	.	69	63.5	57.3	62.5	59	72	64.8	67	57.5

Split Columns: Step-by-Step Instructions

Step 1: Assign Data to Roles

1. (Optional) To filter the input data source, enter the filter expression in the **Filter** field.
2. Assign to the **Column to split** role the variable that contains the values that you want to split into multiple columns.
3. (Optional) Assign to the **Formatted identifier values** role the column that provides the name for the new columns that contain the transposed values of **Columns to split**.
4. (Optional) Assign to the **Labels for new columns** role the column that provides the labels for the new columns that contain the transposed values of **Columns to split**.
5. (Optional) To select a grouping variable, expand the **Additional Roles** heading and assign a column to the **Group analysis by** role.

   Each variable that you assign to this role is used to segment the rows of **Column to split** into subgroups that are transposed into new columns. Each subgroup, which is defined using a unique combination of the values of the grouping variables, becomes a row in the output table. The number of rows that form a subgroup is determined by the number of unique values in the **Formatted identifier values** column.

Step 2: Specify the Output Data

On the **Output** tab, you can set these options:

1. (Optional) To replace an existing output table, select **Replace existing output table**. Any existing table is replaced by default.
2. (Optional) To use a prefix when constructing the names for the transposed variables in the output table, select **Use column name prefix**. When you use a prefix, the variable name begins with the prefix value and is followed by the number 1, 2, and so on.

   By default, the new columns are prefixed with `Column`. You can customize this prefix.
3. Specify the data to print in the results. By default, no output table is displayed in the results.

Stack Columns

About the Stack Columns Step

The Stack Columns step creates an output table by restructuring the selected columns in the input table so that these columns are transposed into observations. You can use the output table to analyze values across multiple columns of the input table. If you group the observations, the selected columns are divided into subgroups that are based on the unique combinations of the grouping values. Each subgroup forms a row of the output table.

This functionality is useful when you have a table in which each observation contains the same type of data in multiple columns and you want to analyze the data across several columns. For example, you could transpose columns that contain monthly temperature readings for various locations across a geographic region. The output table would contain the monthly temperature readings by location in a single column.

Note: The Stack Columns step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirements for Node Connection

In order to run the Split Columns step, you must create connections to the input and output ports of the node as indicated here:

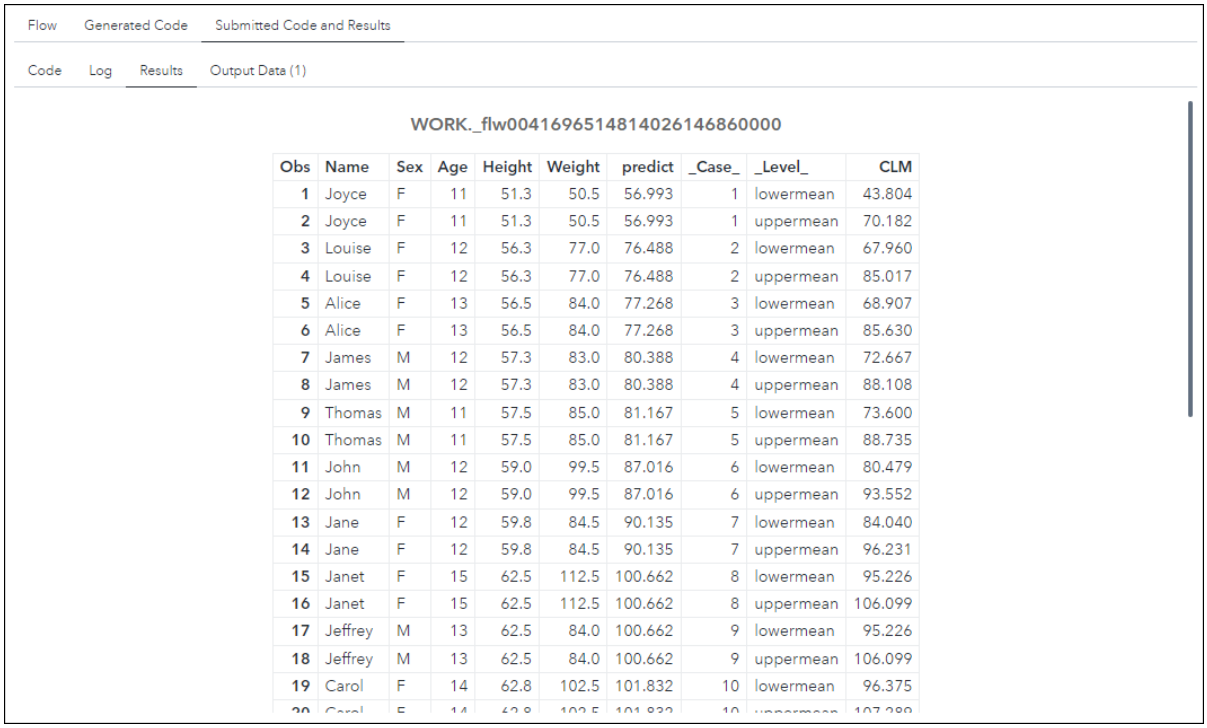
Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and name of the output tables by connecting the output port to a Table node. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Example: Stacking Columns in the ClassFit Table

1. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Split Columns** to add the step to your flow.
2. Add the Sashelp.ClassFit table to your flow. Connect the input port of the Stack Columns node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the Stack Columns step, click the Stack Columns node.
4. On the **Data** tab, assign the **lowermean** and **uppermean** columns to the **Columns to stack** role.
5. On the **Output** tab, complete these steps:
   - Enter `CLM` as the name of the new column.
   - Include these variables in the output data table:
     - **Name**
     - **Sex**
     - **Age**
     - **Height**
     - **Weight**
     - **predict**
   - From the **Specify data to print** drop-down list, select **All data**.
6. Run the flow.

Open the **Submitted Code and Results** tab to view the results.

The results contain three new variables: _Case_, _Level_, and CLM. The _Case_ variable contains the case identifier. A case is the data for an individual student. The _Level_ variable contains the names of the stacked columns. The new CLM variable contains the value of the lower mean or upper mean.



WORK._flw0041696514814026146860000

Obs	Name	Sex	Age	Height	Weight	predict	_Case_	_Level_	CLM
1	Joyce	F	11	51.3	50.5	56.993	1	lowermean	43.804
2	Joyce	F	11	51.3	50.5	56.993	1	uppermean	70.182
3	Louise	F	12	56.3	77.0	76.488	2	lowermean	67.960
4	Louise	F	12	56.3	77.0	76.488	2	uppermean	85.017
5	Alice	F	13	56.5	84.0	77.268	3	lowermean	68.907
6	Alice	F	13	56.5	84.0	77.268	3	uppermean	85.630
7	James	M	12	57.3	83.0	80.388	4	lowermean	72.667
8	James	M	12	57.3	83.0	80.388	4	uppermean	88.108
9	Thomas	M	11	57.5	85.0	81.167	5	lowermean	73.600
10	Thomas	M	11	57.5	85.0	81.167	5	uppermean	88.735
11	John	M	12	59.0	99.5	87.016	6	lowermean	80.479
12	John	M	12	59.0	99.5	87.016	6	uppermean	93.552
13	Jane	F	12	59.8	84.5	90.135	7	lowermean	84.040
14	Jane	F	12	59.8	84.5	90.135	7	uppermean	96.231
15	Janet	F	15	62.5	112.5	100.662	8	lowermean	95.226
16	Janet	F	15	62.5	112.5	100.662	8	uppermean	106.099
17	Jeffrey	M	13	62.5	84.0	100.662	9	lowermean	95.226
18	Jeffrey	M	13	62.5	84.0	100.662	9	uppermean	106.099
19	Carol	F	14	62.8	102.5	101.832	10	lowermean	96.375
20	Carol	F	14	62.8	102.5	101.832	10	uppermean	107.288

Stack Columns: Step-by-Step Instructions

Step 1: Assign Data to Roles

1. (Optional) To filter the input data source, enter the filter expression in the **Filter** field.
2. Assign to the **Columns to stack** role the variables that contain the values that you want to stack. Each column that you assign to this role becomes one or more rows of the output table. You must assign at least one column to this role.
3. Specify the number of stacked variables to create.

   Note: The number of variables in the **Columns to stack** role must be a multiple of the number of stacked variables that you want to create.

4. (Optional) Assign to the **Group analysis by** role the variables to form BY groups.

   Each variable that you assign to this role is used to segment the columns to stack into subgroups that are transposed separately. Each subgroup, which is defined by using a unique combination of the values of the grouping variables, becomes a row of the output data set. Each unique combination of the values of the grouping variable occurs only once.

Step 2: Specify the Output Data

On the **Output** tab, you can set these options:

1. (Optional) To replace an existing output table, select **Replace existing output table**. Any existing table is replaced by default.
2. Specify the name of the new column.
3. Select the case identifier, which is the new column that contains the values that identify a particular case. You can select whether the step creates a case identifier variable, or you can select identifier variables from the input table.
4. Specify the name of the new column that contains the values of the case identifier. The default name is `_Case_`.
5. Specify the name of the new column that contains the levels. The default name is `_Level_`.
6. Select other variables from the input table that you want to include in the output table.
7. Specify the data to print in the results. By default, no output table is displayed in the results.

Transpose Data

About the Transpose Data Step

The Transpose Data step turns selected columns of an input table into the rows of an output table. If you do not use grouping variables, then each selected column is turned into a single row. If you use grouping variables, then the selected columns are divided into subcolumns based on the values of the grouping variables. Each subcolumn is turned into a row of the output table.

Note: The Transpose Data step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Node Connection Requirements

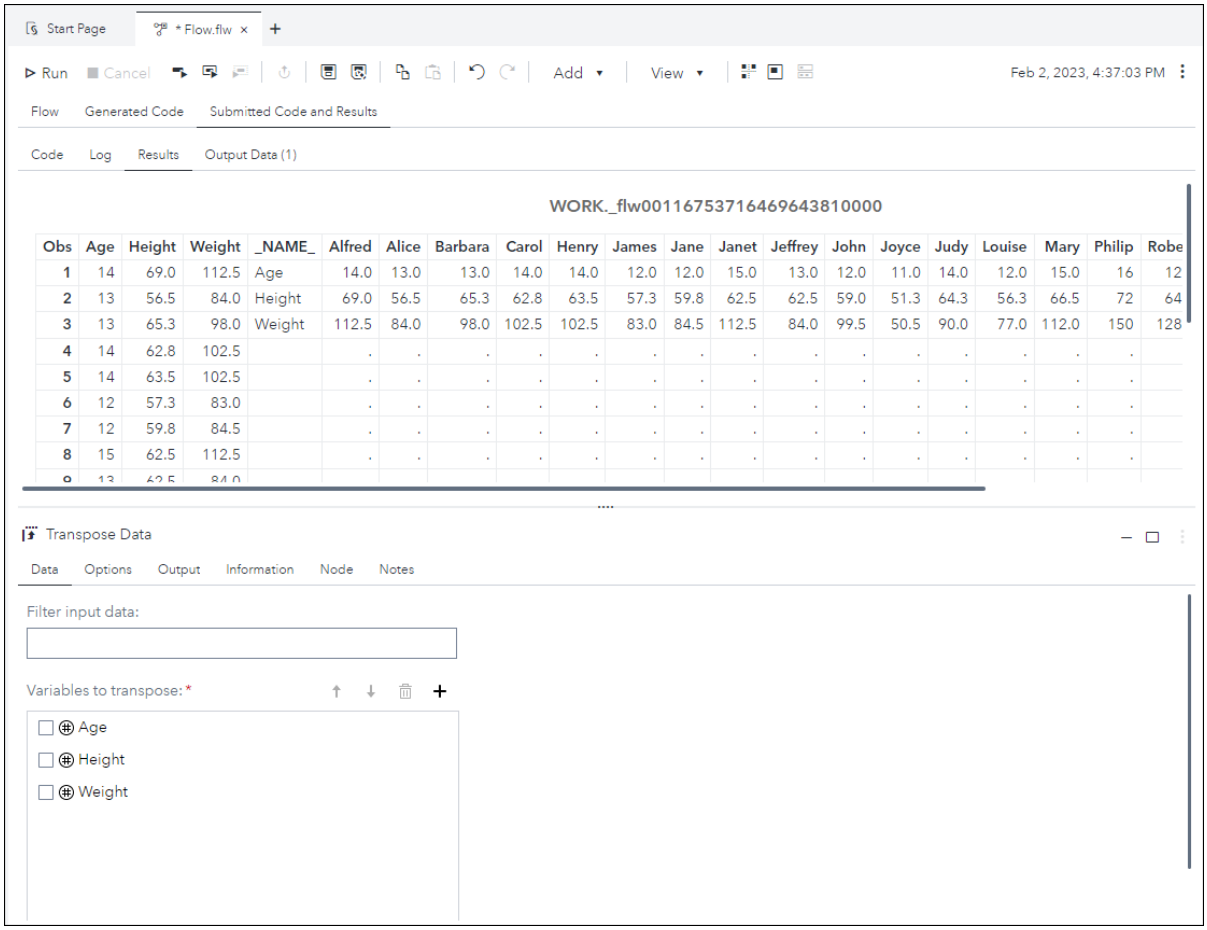
In order to run the Transpose Data step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and name of the output tables by connecting the output port to a Table node. For more information, see "Adding a Table from a SAS Library to a Flow" on page 36.

Example: Transposing Data in the CLASS Data Set

1. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Transpose Data** to add the step to your flow.
2. Add the Sashelp.Class table to your flow. Connect the input port of the Transpose Data node to the data source. For more information, see "Connecting Nodes" on page 12.
3. To set the options for the Transpose Data step, click the Transpose Data node.
4. On the **Data** tab, assign the **Age**, **Height**, and **Weight** variables.
5. On the **Options** tab, complete these steps:
   - Clear the **Use prefix** check box.
   - Select the **Select a variable that contains the names of the new variables** check box.
   - To the **New column names** role, assign the **Name** variable.
6. Run the flow.

Open the **Submitted Code and Results** tab to view the results.



To view the generated SAS code:

1 Open the **Log** tab.

2 Search for MPRINT_CODE.

Transpose Data: Step-by-Step Instructions

Step 1: Assign Data to Roles

1. (Optional) To filter the input data source, enter the filter expression in the **Filter** field.
2. Assign at least one variable to the **Variables to transpose** role.

   Each variable that you assign to this role becomes one or more rows of the output table. If you do not select any grouping variables, then an entire column is turned into a single row. If you select one or more grouping variables, then the grouping variables are used to segment each column into subcolumns, each of which is turned into a row. In this case, a column is transposed to the number of rows that is equal to the number of groups that are defined by the grouping variables.
3. To select a grouping variable, assign a column to the **Group analysis by** role.

   Each variable that you assign to this role is used to segment the about-to-be-transposed columns into subcolumns that are transposed separately. Each subcolumn, defined by a set of values of the grouping variables, becomes a row of the output table.

Step 2: Set Options for Transposed and Original Variables

On the **Options** tab, you can set these options for the transposed and original variables.

1 To construct new variable names, specify these options:

- Select the **Use prefix**option. You can specify a prefix to use in constructing the names for the transposed variables in the output data set. When you use a prefix, the variable name begins with the prefix value and is followed by the number 1, 2, and so on.
- Select **Select a variable that contains the names of the new variables**.

  The variable that you assign to the **New column names** role is used to name the transposed variables in the output data set. If you specified to use a prefix in the name, the name for the new variable begins with the prefix and is followed by the value of the **New column names** variable. If you select the **Allow duplicate of ID values** check box, the transposed output data set contains only the last observation for each BY group.

2 To construct new variable labels, select **Select a variable that contains the labels of the new variables**.

  The values of the variable that you assign to the **New column labels** role are used to label the variables in the output data set.

3 To put the original variable names and labels in a new variable, use these options:

- **Put original variable names in a new variable** - Each row of the output table includes the name of the variable in the input table to which the values in that output row belong. To specify a heading for the output column that contains these variable names, enter the heading in the **Name** box. The name can include special characters, leading numbers, and white space, but it cannot exceed 32 characters. The default name is _Name_.
- **Put original variables labels in a new variable** - Each row of the output table includes the label of the variable in the input table to which the values in that output row belong. To specify a heading for the output column that contains these variable labels, enter the heading in the **Label** box. The label can include special characters, leading numbers, and white space, but it cannot exceed 32 characters. The default label is _Label_.

Step 3: Specify the Output Data

On the **Output** tab, you can set these options.

1 To replace an existing output table with the same name, select **Replace existing output table**.

2 (Optional) Assign one or more variables to the **Copy to output data** role.

   Each variable that you assign to this role is copied directly from the input table to the output table without being transposed. Because these columns are copied directly to the output table, the number of rows in the output table equals the number of rows in the input table. The output table is padded with missing values if the number of rows in the input table does not equal the number of variables that it transposes.

3 Use **Specify data to print**to print none of the data, a subset of the observations, or all the observations.

Union Rows

About the Union Rows Step

The Union Rows step enables you to combine data from multiple sources into a single target table. You can subset the source data and choose from multiple set operators to determine how to combine the data sources.

Note: The Union Rows step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

There are five main steps to use the Union Rows step:

1 Add the Union Rows step to your flow and connect source data to each input port.

2 Specify the set operators that you want to use to combine the data sources.

3 (Optional) Specify source data options.

4 (Optional) Specify set operator options.

5 Run the flow to combine the data.

Node Connection Requirements

In order to run the Union Rows step, you must create connections to the input and output ports of the node as indicated here:

Input Ports	Output Port
Two or more connections are required. You can connect either of the following types of nodes: ■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and name of the output tables by connecting the output port to a Table node. For more information, see "Adding a Table from a SAS Library to a Flow" on page 36.

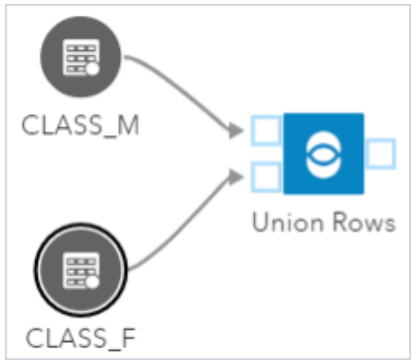
Example: Subsetting and Combining Data

In this example, you use the output data from the Branch Rows example as your source data. In the Branch Rows example, you split the Sashelp.Class data set into two separate tables: one with only male students and the other with only female students. You can use the Union Rows step to combine the two tables, but include only the male students under the age of 14 and the female students over the age of 12.

To replicate this example, start by following the steps in the Branch Rows example and specify two output tables: Work.Class_M and Work.Class_F. For more information, see "Example: Splitting the Sashelp.Class Data Set" on page 62.

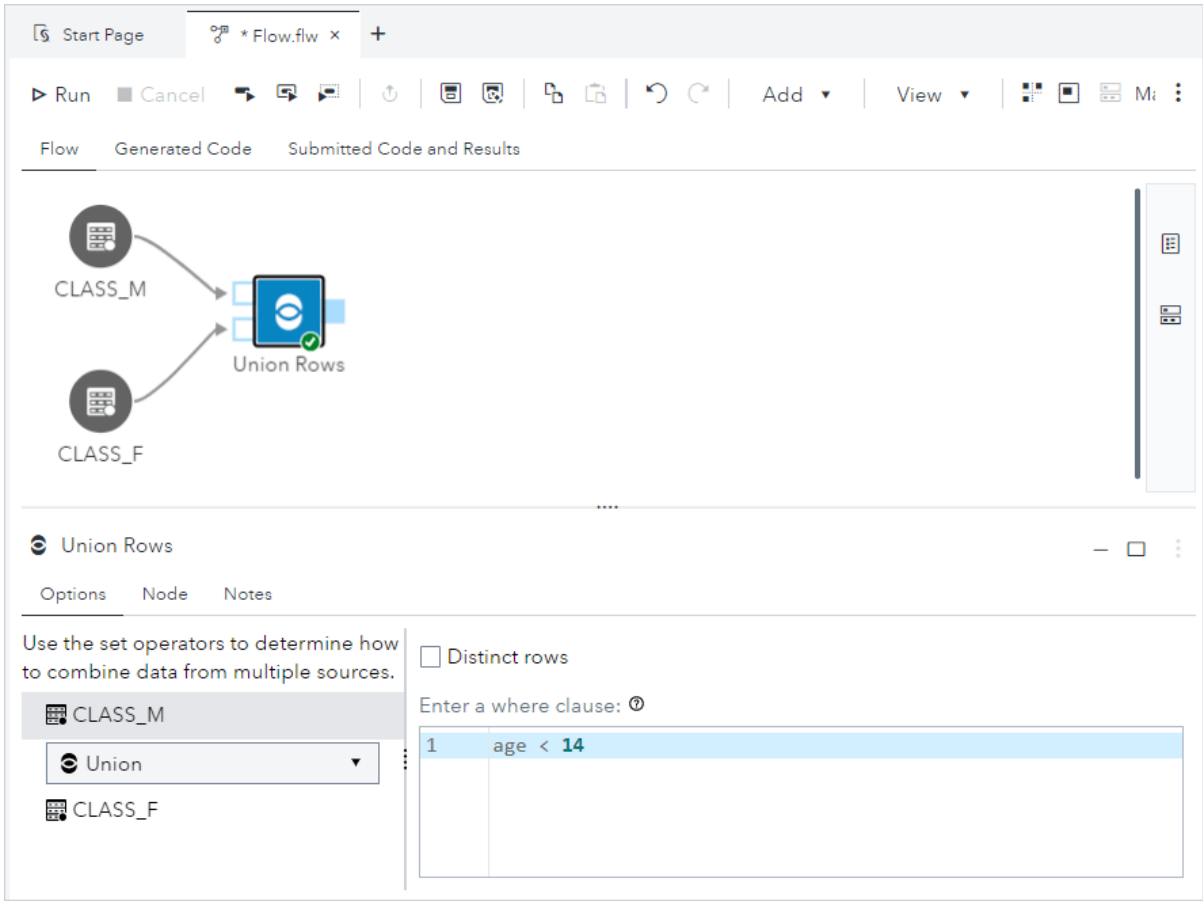
To create this example:

1. From the main SAS Studio menu, select **New** ⇨ **Flow**.
2. In the **Steps** section of the navigation pane, expand the **Transform Data** folder and double-click **Union Rows**.
3. In the **Libraries** section of the navigation pane, expand the Work library. Drag the Class_M table onto the first input port of the Union Rows node. Drop the table on the flow canvas when the tooltip on your mouse pointer changes to `Connect to input port`.
4. Repeat step 3 for the Class_F table and the second input port.



5 Click the Union Rows node. On the **Options** tab, accept `Union` as the default set operator.

6 On the **Options** tab, click CLASS_M. In the **Enter a where clause** box, enter `age < 14`.

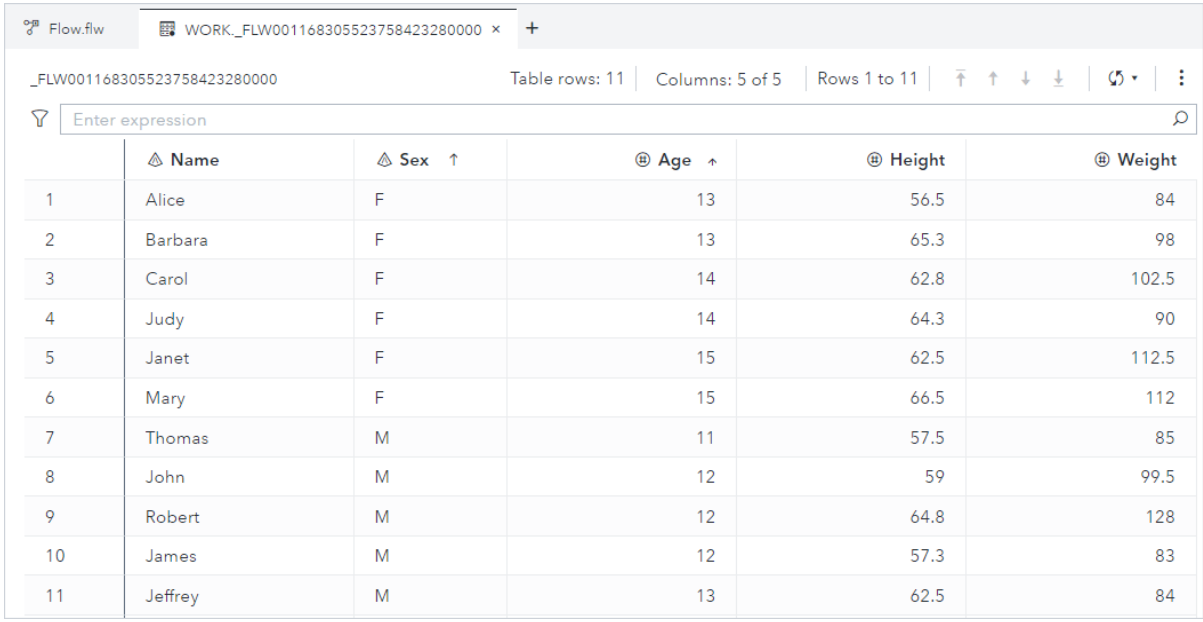


7 Click CLASS_F, and in the **Enter a where clause** box, enter `age > 12`.

8 To run the flow, click Run.

> **TIP** You can specify the library and name of the output tables by connecting the output ports to Table nodes. For more information, see "Adding a Table from a SAS Library to a Flow" on page 36.

Here is the default output table of male students under age 14 and female students over age 12.


Flow.flw
WORK._FLW0011683055237584232800000 ×
_FLW0011683055237584232800000 Table rows: 11 Columns: 5 of 5 Rows 1 to 11
Enter expression

	Name	Sex	Age	Height	Weight
1	Alice	F	13	56.5	84
2	Barbara	F	13	65.3	98
3	Carol	F	14	62.8	102.5
4	Judy	F	14	64.3	90
5	Janet	F	15	62.5	112.5
6	Mary	F	15	66.5	112
7	Thomas	M	11	57.5	85
8	John	M	12	59	99.5
9	Robert	M	12	64.8	128
10	James	M	12	57.3	83
11	Jeffrey	M	13	62.5	84

Union Rows: Step-by-Step Instructions

Step 1: Add the Union Rows Step and Connect the Source Data

To add the Union Rows step to your flow:

1 In the **Steps** section of the navigation pane, expand the **Transform Data** folder, and double-click **Union Rows**.

2 Connect each input port of the Union Rows node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

Note: At least one source table or the target table must have a data type other than CAS.

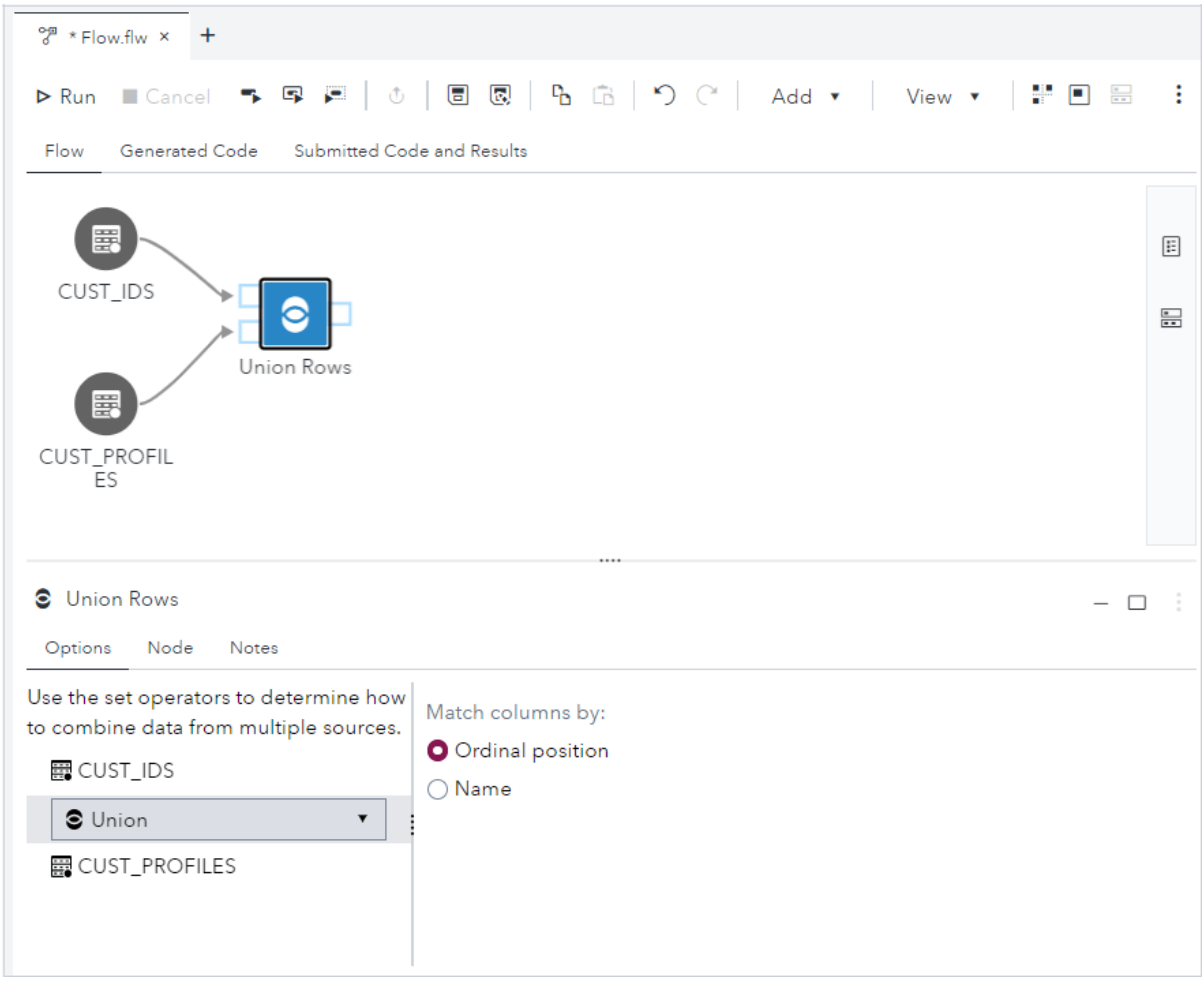
Step 2: Specify the Set Operators to Combine the Data Sources

Select a set operator for each pair of data sources that you are combining.

- Click the **Options** tab and select the set operator that you want to use to combine each pair of data sources.

  You can choose from among these set operators:

  - Union - returns all unique matching rows from both data sources. This is the default operator.
  - Union all - returns all matching rows from both data sources, including duplicate rows.
  - Outer Union - returns a concatenation of both data sources.
  - Except - returns the unique rows in the first data source that are not found in the second data source.
  - Except all - returns all unique and duplicate rows in the first data source that do not have a matching row in the second data source.
  - Intersect - returns the distinct rows that are common in both data sources.
  - Intersect all - returns the rows that are common in both data sources, including duplicates.



(Optional) Step 3: Specify Source Data Options

You can choose to include only the distinct rows and specify a WHERE clause for each input data source.

1 Click the **Options** tab and select the data source that you want to subset.

2 To include only the distinct rows from the selected data source, select **Distinct rows**. This option is not selected by default.

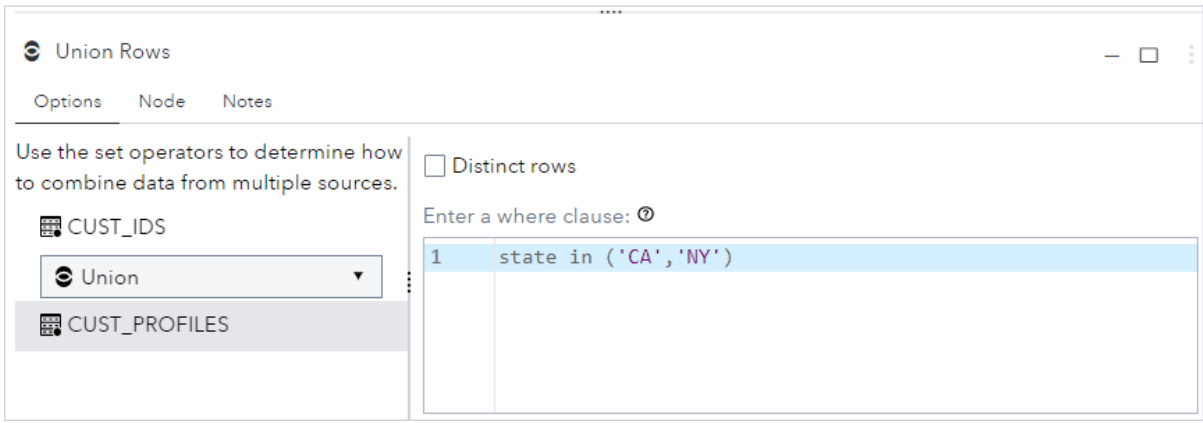
3 To specify a WHERE clause for the selected data source, enter the code for the WHERE clause conditions in the **Enter a where clause** box. Enter only the code for the condition. Do not include the WHERE keyword or the semicolon statement terminator in the text that you enter.

Note: The syntax for the WHERE clause conditions can vary depending on the code that is generated for the step and the data type of the source table. When PROC SQL code is generated, the WHERE clause code that you enter is used to generate a WHERE clause for the specific source table. When DATA step code is generated, the WHERE clause code that you enter is used to generate a WHERE= data set option for the specific source table.

For example, enter this condition to subset the Sashelp.Cars data by country of origin using PROC SQL syntax:

```
origin in ('Asia', 'Europe')
```

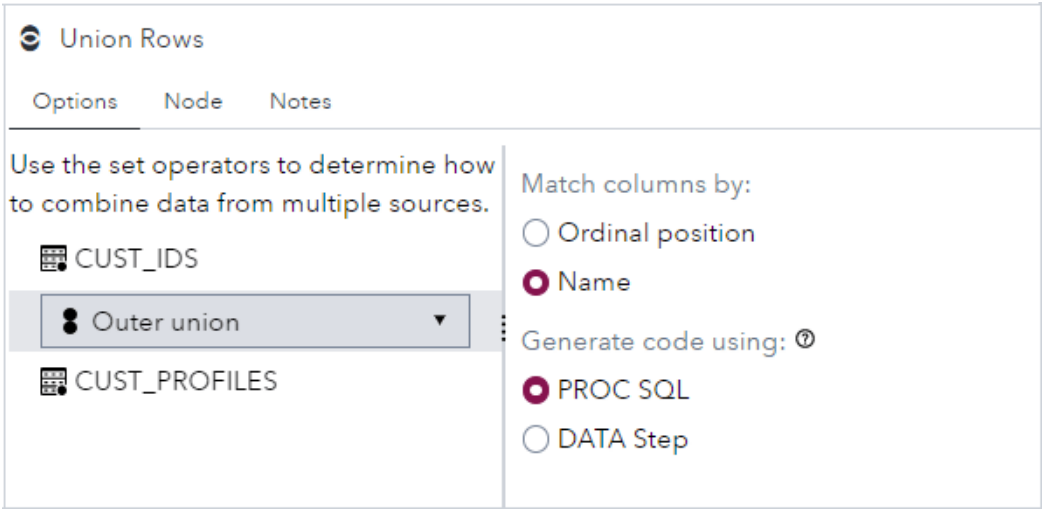
Note: The WHERE clause syntax is validated only when the step runs.



(Optional) Step 4: Specify Set Operator Options

You can specify whether to match columns by their ordinal position in the data or by name.

1. Click the **Options** tab and select the set operator for which you want to set options.
2. Use the **Match columns by** option to specify how to match columns.
   - **Ordinal position** - matches columns by their position in the data source. This is the default value.
   - **Name** - matches columns by name. Columns that do not match by name are excluded from the output data unless you are using the Outer Union set operator.
3. If you are using the Outer Union set operator and matching columns by name, you can use the **Generate code using** option to specify whether to generate code using PROC SQL or a DATA step. The default value is PROC SQL. If you are combining more than two tables, you can generate code using both PROC SQL and DATA steps in a single Union Rows node. In this case, SAS Studio generates intermediary tables prior to creating the final target table.



Note: When you switch between generating code using PROC SQL and DATA Step, you might get errors if you have specified any conditions for a WHERE clause. You can edit the code that you entered to fix any errors.

Step 5: Run the Flow and Combine the Data

To combine the data sources into the target table, click ▷ Run.

6

Enriching Your Data

Geocode Data

About the Geocode Data Step

Geocoding is the process of adding geographic coordinates (latitude and longitude values) to an address. This process provides a way to convert address data into

map locations. The geographic coordinates typically represent the center of a ZIP code, a city, an address, or any geographic region. After geocoding, the coordinates can be used to display a point on a map or to calculate distances. Geocoding also enables you to add attribute values such as census blocks to the geocoded address.

The Geocode Data step uses the SAS GEOCODE procedure. The GEOCODE procedure uses two types of input data:

- an input table with columns that relate to specific geographic locations. For example, a table might contain mailing address variables such as ZIP codes and street addresses, or custom geographic variables such as sales regions.
- lookup tables that contain reference variables and geographic coordinates. For example, a lookup data table for the ZIP method contains ZIP codes and the geographic coordinates that are associated with the ZIP codes. Some geocoding methods require multiple lookup data tables. This data is essential to transform address data into location information that can be viewed on a map.

By default, the GEOCODE procedure produces an output data set that contains all of the variables from the input table. The data set also contains the X, Y or LONG, LAT, and _MATCHED_ variables. The X or LONG and Y or LAT coordinates must be in the same coordinate system as the lookup data set.

The Geocode step supports seven geocode methods:

- U.S. City
- World City
- Street
- Zip
- Plus4
- Custom
- Internet Protocol (IP) addresses or other ranges

Note: The process of adding coordinates for IP addresses is usually called *geolocating*. IP data is a form of range data and was not designed to be geographic.

For more information, see "GEOCODE Procedure" in *SAS/GRAPH and Base SAS: Mapping Reference*.

Node Connection Requirements

The Geocode Data step includes an input port and an output port. In order to run the Geocode Data step, you must create the input port of the node as indicated here:

Input Port	Output Port
■ Table node or	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify

Input Port	Output Port
■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	the library and name of the output table by connecting the output port to a Table node. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

TIP Depending on the geocode method, you might need more than one input port on the Geocode Data node. To view the labels for the ports on the Geocode Data node, right-click the Geocode Data node and select **Expand ports**.

Using the U.S. City Geocode Method

About the U.S. City Geocode Method

For U.S. city geocoding, each observation in your input data must contain a city and a state. The recommended state value is the two-character state abbreviation, but the complete state name can also be used.

U.S. City Geocoding: Step-by-Step Instructions

Step 1: Add the Geocode Data Step and Connect a Source Table

1 In the **Steps** section of the navigation pane, expand the **Enrichment** folder and double-click **Geocode Data** to add the step to your flow.

2 Connect the input port of the Geocode Data node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

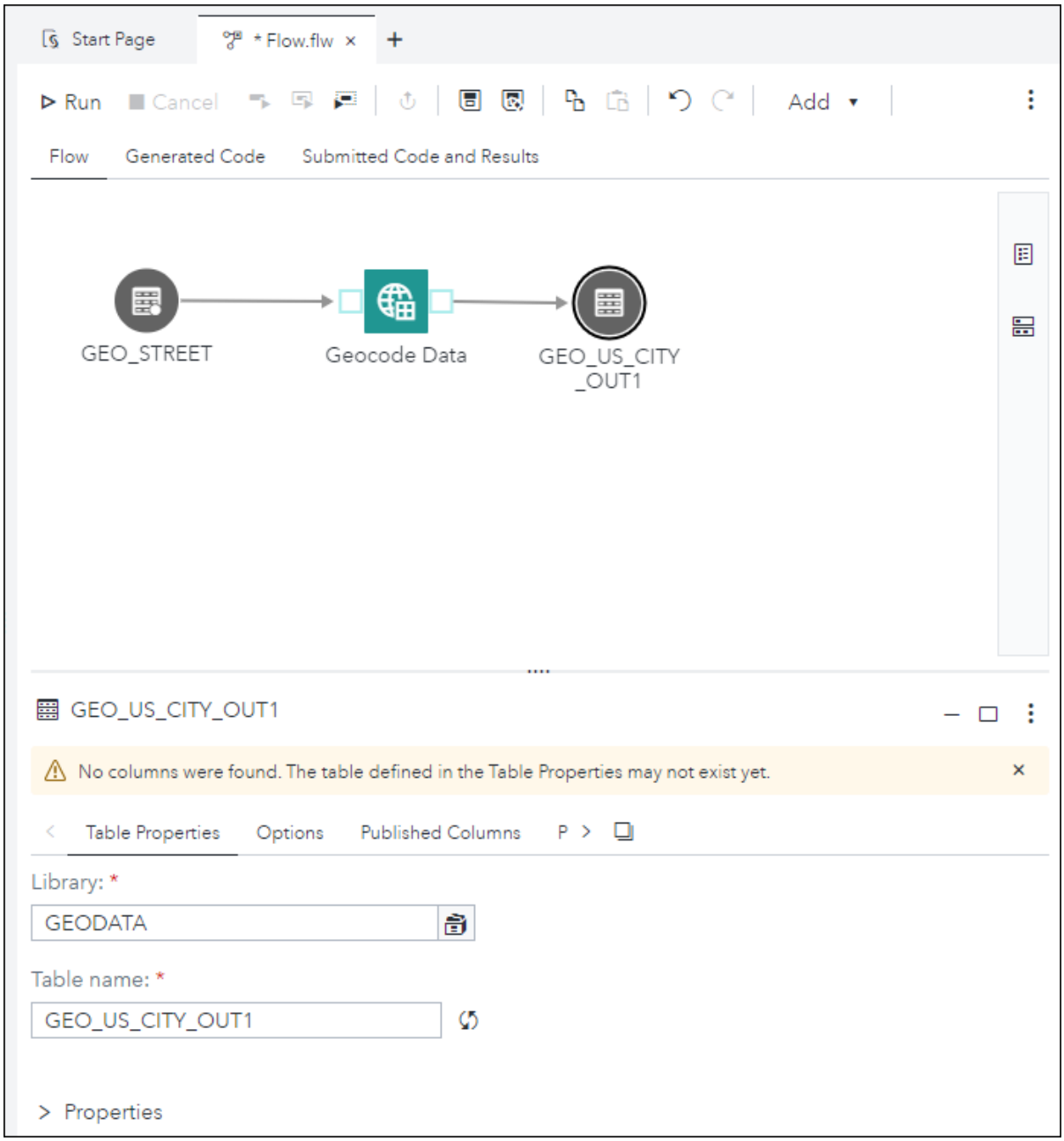
Note: The input data must contain columns for the city and state.

3 To save the results from the Geocode Data step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and

name for the output table. (In this example, the output table is named Geo_US_City_Out1.)

Next, connect the output port of the Geocode Data node to the table node.

Note: The output table does not have any columns until you run the flow.



Step 2: Select U.S. City as the Geocoding Method

1 Select **World city** as the geocoding method.

2 Select the **U.S. city** check box.

3 Map these fields for geocoding:
   - **City**
   - **State/Province**
   - **Postal code**

(Optional) Step 3: Set Additional Options

1 To convert the data to use standard two-letter abbreviation, select **Perform state two-letter abbreviation before processing**.

2 Specify any optional arguments from the GEOCODE procedure.

   For example, add `LOOKUPCITY=SASHELP.ZIPCODE` to specify the lookup table for associating coordinates with addresses.

(Optional) Step 4: Set the Debugging Options

To add extra SAS debugging for macros, select **Debug SAS macros**.

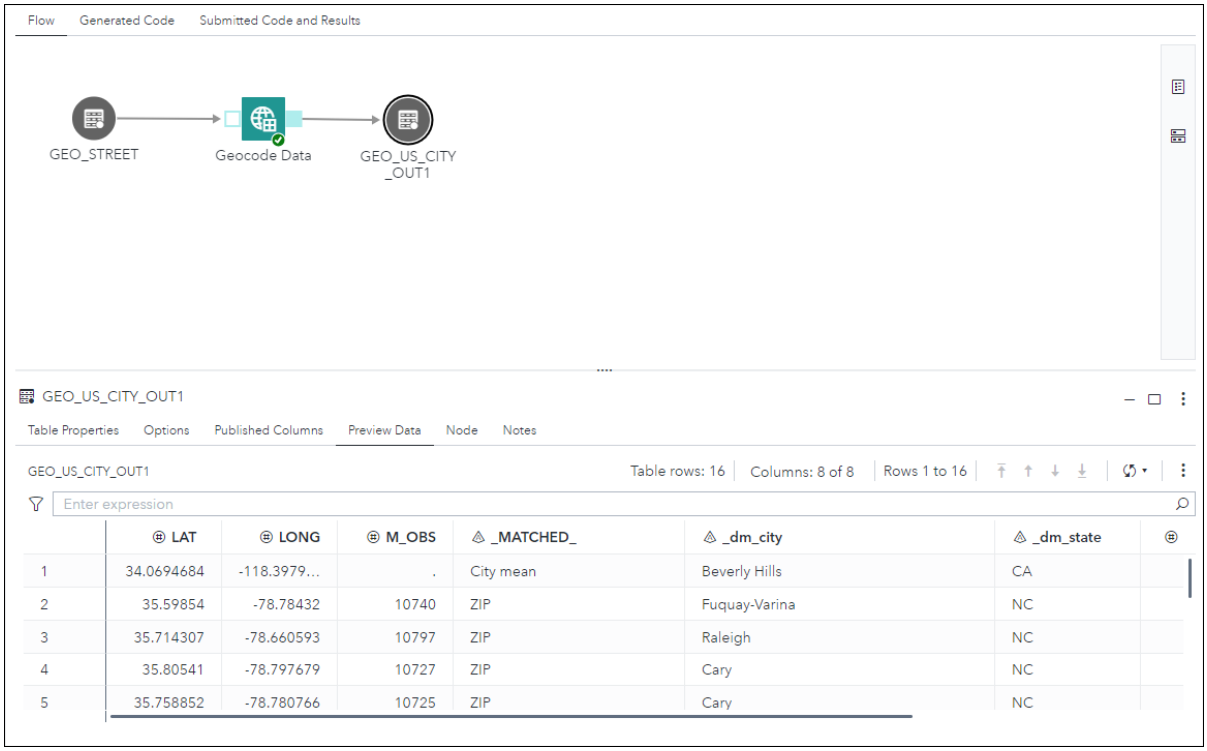
Step 5: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.

Items to note in the output table:

- Columns from the SAS data source are prefixed with _dm.
- The GEOCODE procedure generates a _MATCHED_ column that indicates how the coordinates were found.



Using the World City Geocode Method

About the World City Geocode Method

For worldwide geocoding, each observation in your input address data must include the city name and the country. The country value can be the complete country name or its two- or three-character country abbreviation. An example is GB or GBR for the United Kingdom. An optional state, province, or region name can also be specified.

World City Geocoding: Step-by-Step Instructions

Step 1: Add the Geocode Data Step and Connect a Source Table

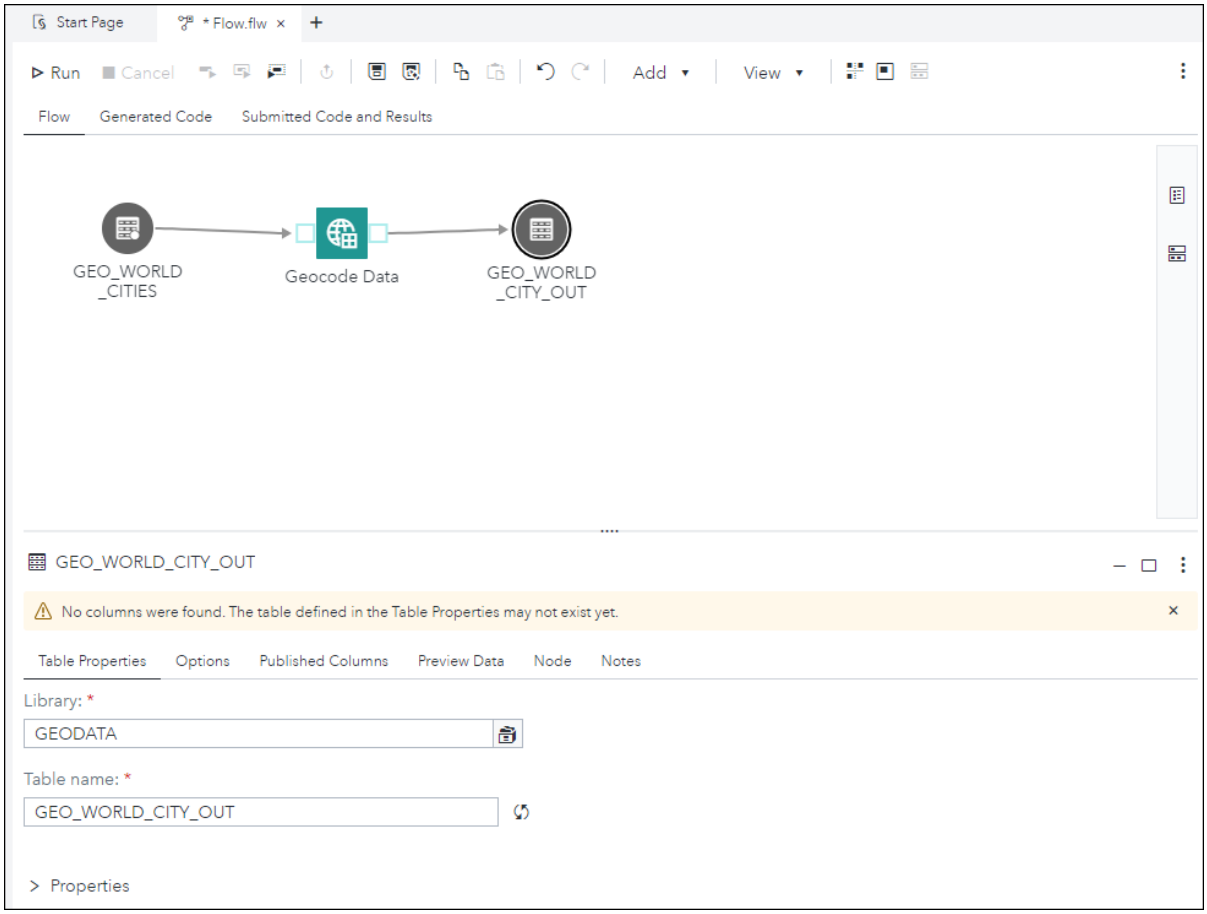
1 In the **Steps** section of the navigation pane, expand the **Enrichment** folder and double-click **Geocode Data** to add the step to your flow.

2 Connect the input port of the Geocode Data node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

3 To save the results from the Geocode Data step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table. (In this example, the output table is named Geo_World_City_Out.)

Next, connect the output port of the Geocode Data node to the table node.

Note: The output table does not have any columns until you run the flow.



Step 2: Select World City as the Geocoding Method

1 In the Geocode Data step, select **World city** as the geocoding method.

2 Map these fields for geocoding:

- **City**
- **Country**

Step 3: Specify Lookup Table

1 To view the labels for the ports on the Geocode Data node, right-click the Geocode Data node in the flow and select **Expand ports**. In the flow, the input port for the input data source (in this example, GEO_WORLD_CITIES) is labeled Input Table 1.

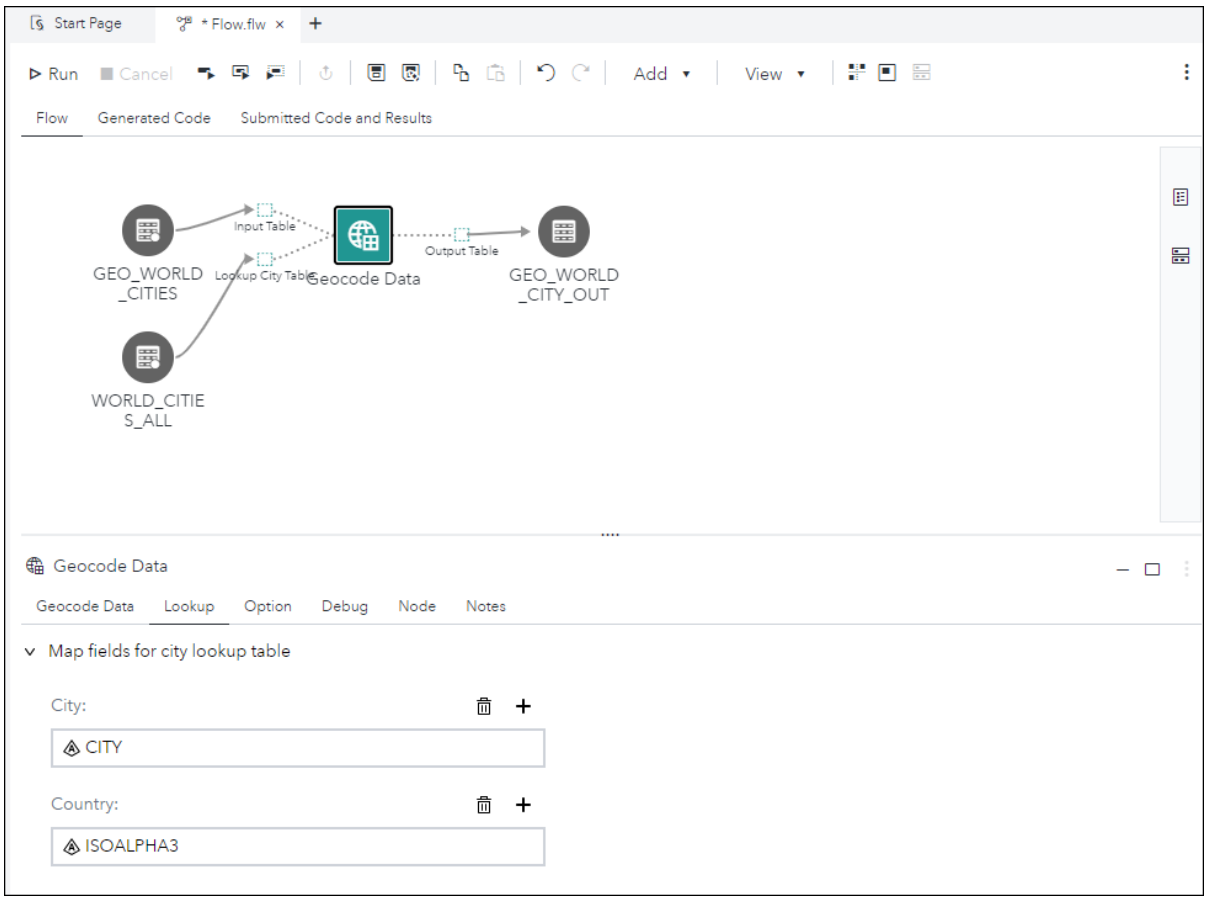
2 Add the input port and data for the city lookup table.

   a Right-click the Geocode Data node and select **Add input port** ⇨ **Lookup City Table**.

   b Add the lookup table to the flow. (In this example, the lookup table is World_Cities_All.) Connect the lookup table node to the Lookup City Table port.

   c On the **Lookup** tab in the Geocode Data step, map these fields for the city lookup table:

   - **City**
   - **Country**



(Optional) Step 4: Set Additional Options

1. To convert the data to use the three-letter ISO standard for the country, select **Perform country-three letter ISO standardization before processing**.
2. Specify any optional arguments from the GEOCODE procedure.

   Here is an example: `attributevar=(isoname mapidname1) addressstatevar=province`

(Optional) Step 5: Set the Debugging Options

To add extra SAS debugging for macros, select **Debug SAS macros**.

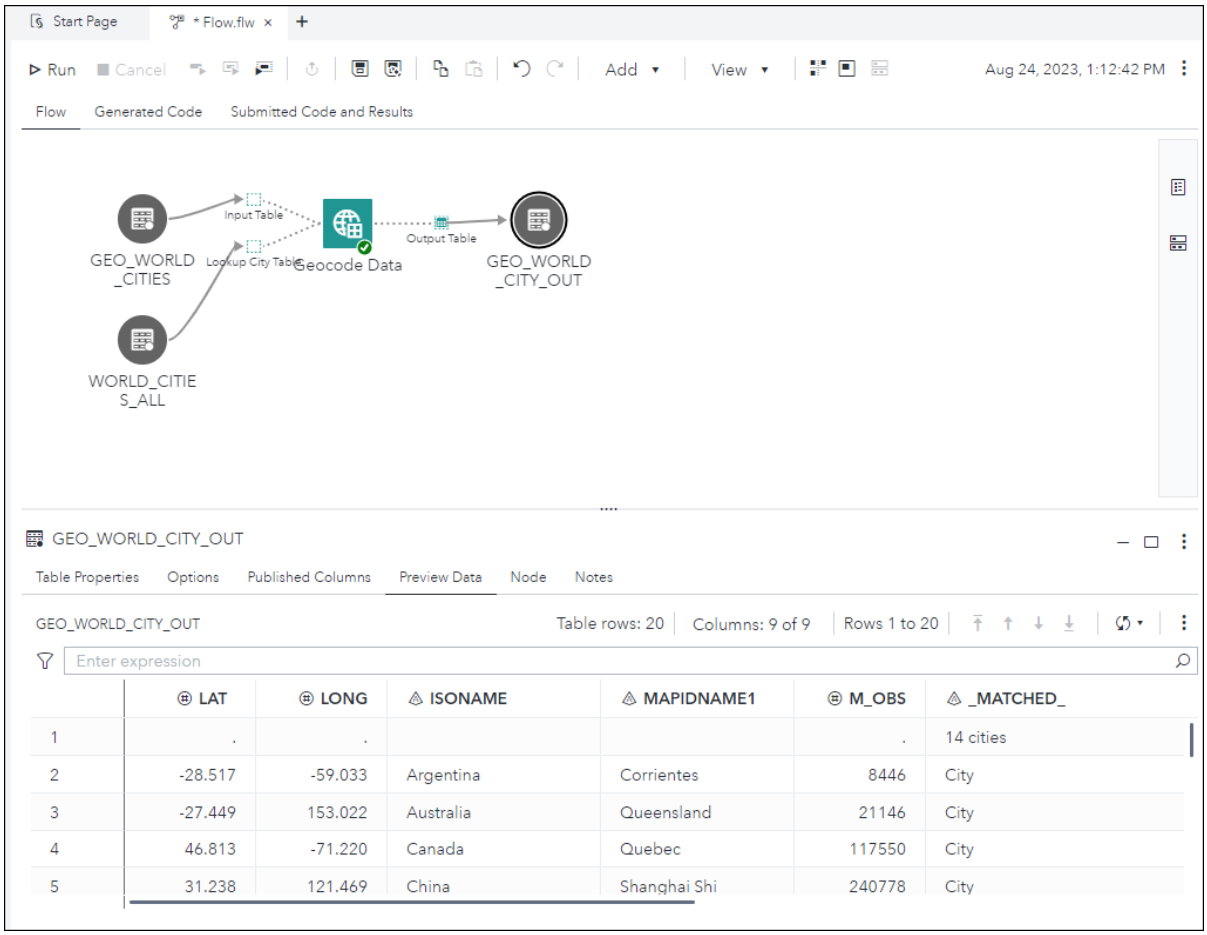
Step 6: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.

Items to note in the output table:

- Columns from the original data source are prefixed with _dm.
- The GEOCODE procedure generates a _MATCHED_ column that indicates how the coordinates were found.



Using the Street Geocode Method

About the Street Geocode Method

The street geocode method computes geographical coordinates for specified U.S. or Canadian street addresses. This method converts a full street address that includes a house or building number, street name, city, state (or province), and ZIP (or postal) code to a map location.

Street Geocoding: Step-by-Step Instructions

Step 1: Add the Geocode Data Step and Connect a Source Table

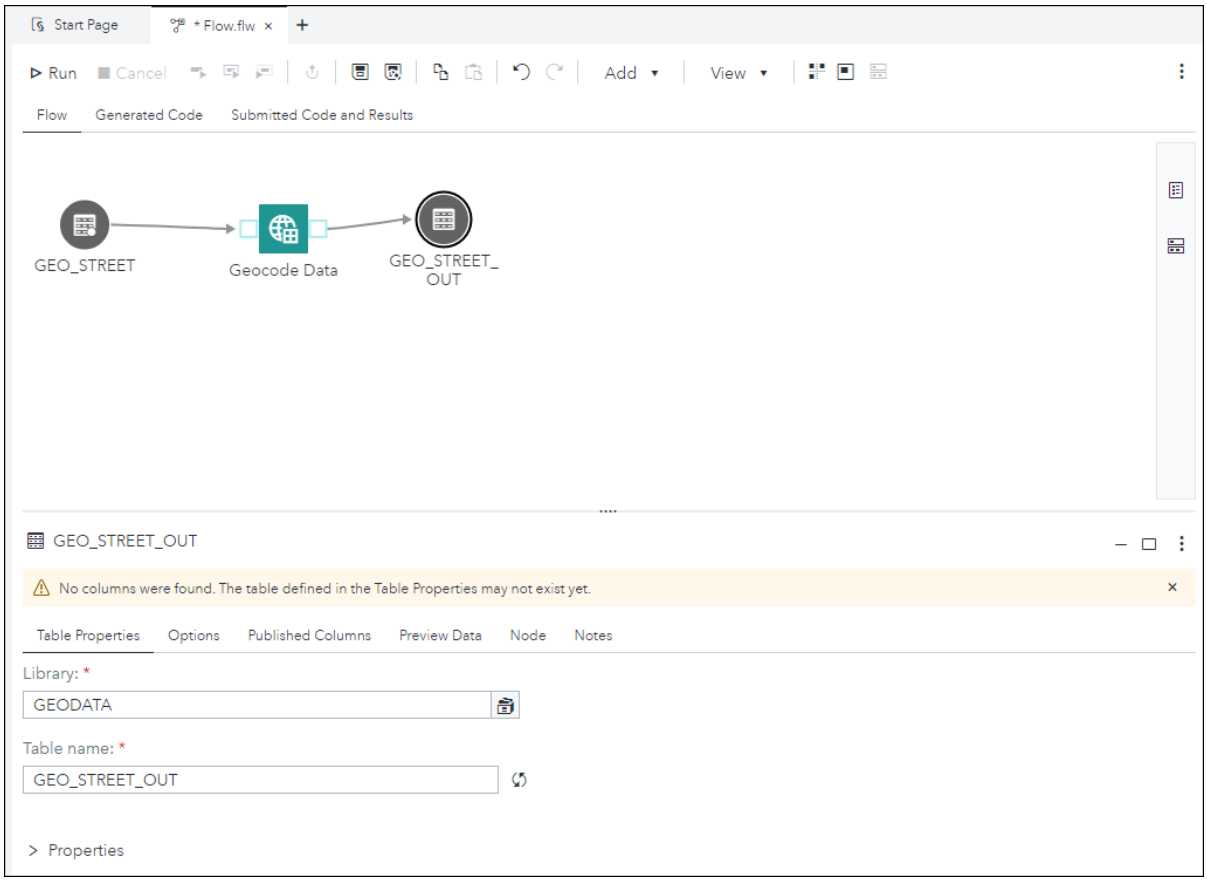
1 In the **Steps** section of the navigation pane, expand the **Enrichment** folder and double-click **Geocode Data** to add the step to your flow.

2 Connect the input port of the Geocode Data node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

3 To save the results from the Geocode Data step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table. (In this example, the output table is named Geo_Street_Out.)

Next, connect the output port of the Geocode Data node to the table node.

Note: The output table does not have any columns until you run the flow.



Step 2: Select Street as the Geocoding Method

1 In the Geocode Data step, select **Street** as the geocoding method.

2 Select the street method:

- **Default**
- **No city**
- **No zip**

3 Map these fields for geocoding:

- **Address**
- **City**

  Note: This field is not available when **No city** is selected as the street method.

- **State/Province**

  Note: This field is not available when **No city** is selected as the street method.

- **Postal code**

Note: This field is not available when **No zip** is selected as the street method.

Step 3: Specify Lookup Table

By default, the Geocode Data step uses SasHelp.USM as the lookup table. You do not need to map any fields for the city lookup table or the lookup table. You must specify the Lookup library where all the lookup reference data is stored.

(Optional) Step 4: Set Additional Options

Specify any optional arguments from the GEOCODE procedure.

(Optional) Step 5: Set the Debugging Options

To add extra SAS debugging for macros, select **Debug SAS macros**.

Step 6: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.

Items to note in the output table:

- Columns from the original data source are prefixed with _dm.
- The GEOCODE procedure generates a _MATCHED_ column that indicates how the coordinates were found.

Using the Plus4 Geocode Method

About the Plus4 Geocode Method

With Plus4 geocoding, the GEOCODE procedure attempts to match the five-digit ZIP code and ZIP+4 extension from your address data set with the lookup data set.

Plus4 Geocoding: Step-by-Step Instructions

Step 1: Add the Geocode Data Step and Connect a Source Table

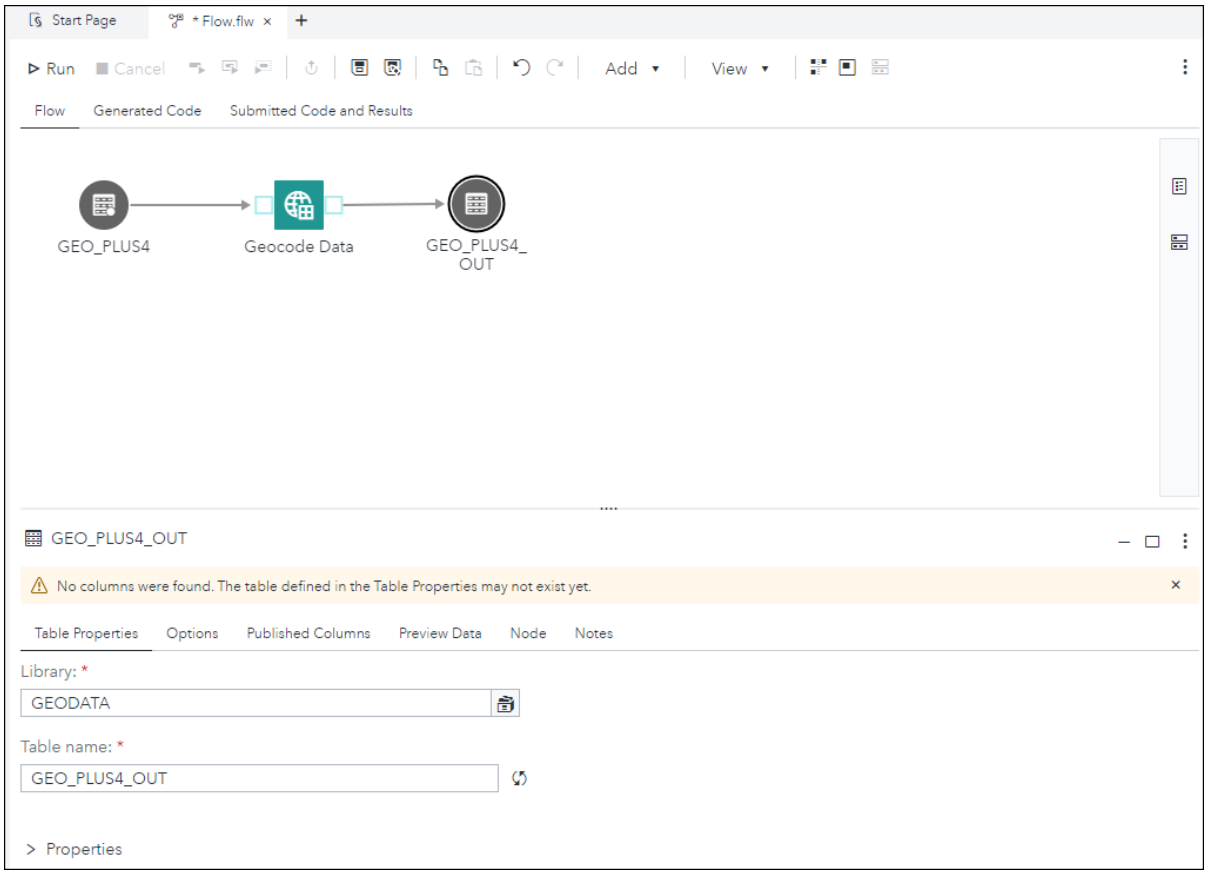
1 In the **Steps** section of the navigation pane, expand the **Enrichment** folder and double-click **Geocode Data** to add the step to your flow.

2 Connect the input port of the Geocode Data node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

3 To save the results from the Geocode Data step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table. (In this example, the output table is named Geo_Plus4_Out.)

   Next, connect the output port of the Geocode Data node to the table node.

   Note: The output table does not have any columns until you run the flow.



Step 2: Select Plus4 as the Geocoding Method

1 In the Geocode Data step, select **Plus4** as the geocoding method.

2 Map these fields for geocoding:

- **Postal code**
- **Zip+4**

Step 3: Specify Lookup Table

1 To view the labels for the ports on the Geocode Data node, right-click the Geocode Data node and select **Expand ports**. In the flow, the input port for the input data source (in this example, GEO_PLUS4) is labeled Input Table 1.

2 To add the input port and data for the lookup table:

a Right-click the Geocode Data node and select **Add input port** ⇨ **Lookup Table**.

b Add the lookup table to the flow. Connect the lookup table node to the Lookup Table 1 port.

c On the **Lookup** tab in the Geocode Data step, map these fields for the lookup table:

- **Postal code**
- **Zip+4**

(Optional) Step 4: Set Additional Options

Specify any optional arguments from the GEOCODE procedure.

(Optional) Step 5: Set the Debugging Options

To add extra SAS debugging for macros, select **Debug SAS macros**.

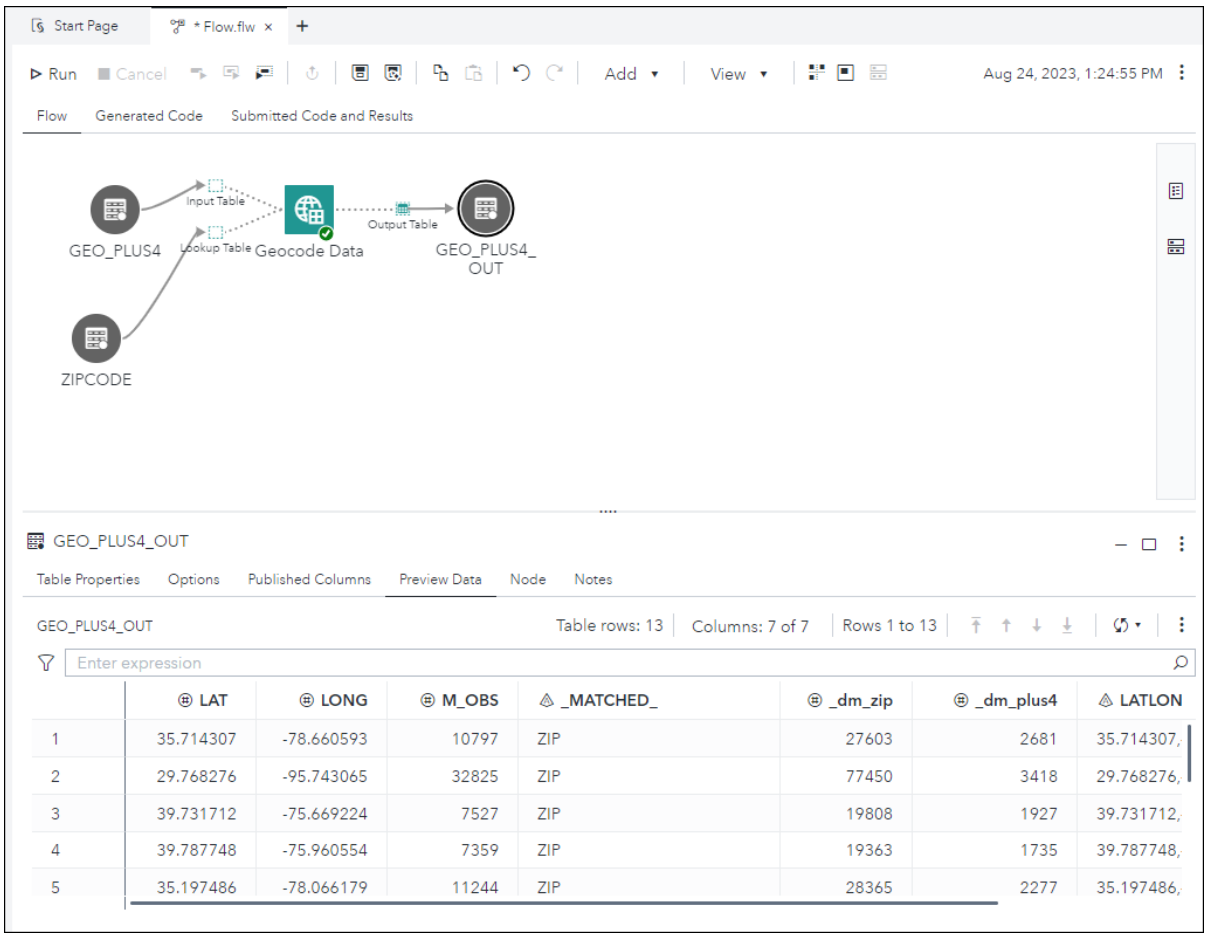
Step 6: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.

Items to note in the output table:

- Columns from the original data source are prefixed with _dm.
- The GEOCODE procedure generates a _MATCHED_ column that indicates how the coordinates were found.



	LAT	LONG	M_OBS	_MATCHED_	_dm_zip	_dm_plus4	LATLON
1	35.714307	-78.660593	10797	ZIP	27603	2681	35.714307,
2	29.768276	-95.743065	32825	ZIP	77450	3418	29.768276,
3	39.731712	-75.669224	7527	ZIP	19808	1927	39.731712,
4	39.787748	-75.960554	7359	ZIP	19363	1735	39.787748,
5	35.197486	-78.066179	11244	ZIP	28365	2277	35.197486,

Using the Zip Geocode Method

About the Zip Geocode Method

Your input data must contain a valid ZIP or postal code for each observation. If a ZIP code is not found, then the GEOCODE procedure attempts to find the city center location. If you are interested in the ZIP code location only, you can turn off this cascading behavior by selecting the **No city** option.

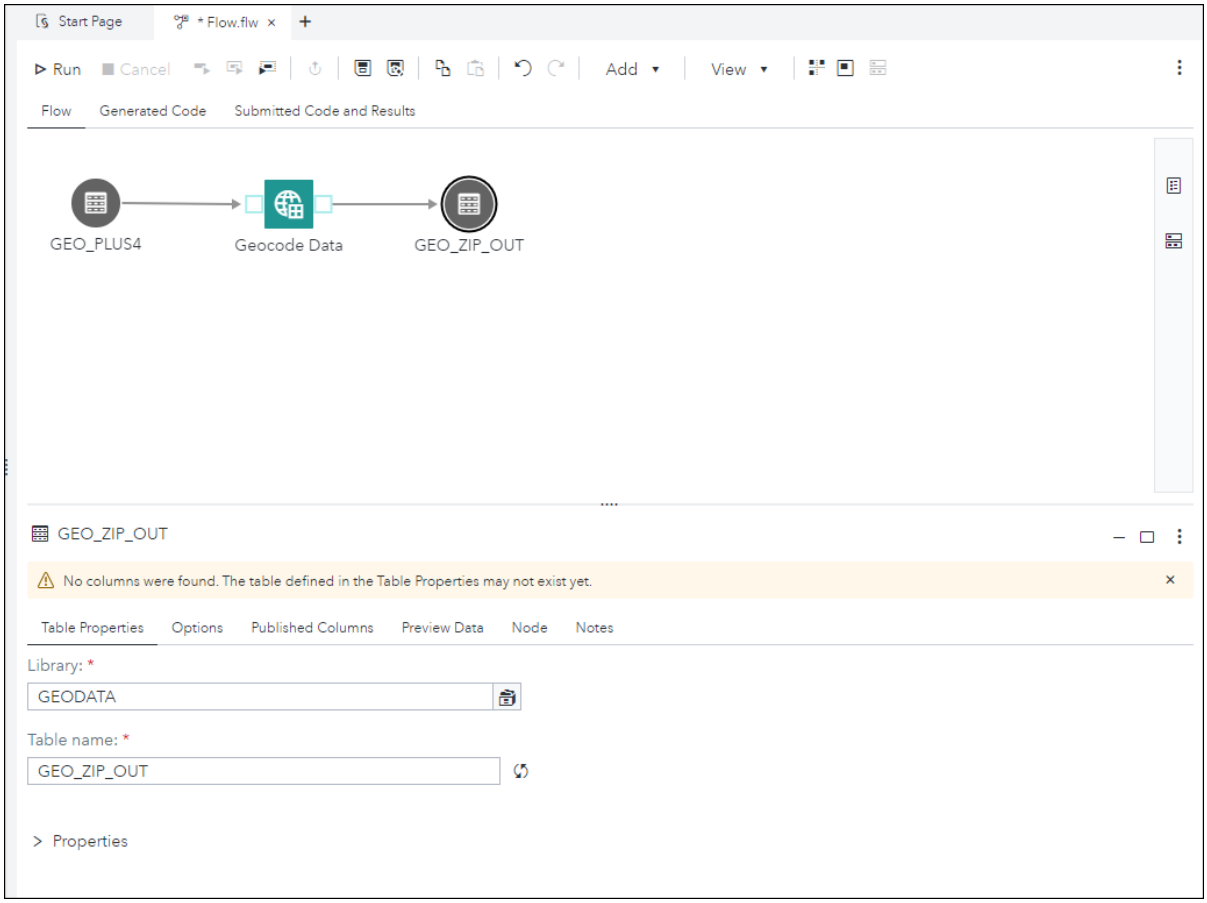
Zip Geocoding: Step-by-Step Instructions

Step 1: Add the Geocode Data Step and Connect a Source Table

1. In the **Steps** section of the navigation pane, expand the **Enrichment** folder and double-click **Geocode Data** to add the step to your flow.
2. Connect the input port of the Geocode Data node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.
3. To save the results from the Geocode Data step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table. (In this example, the output table is named Geo_Zip_Out.)

   Next, connect the output port of the Geocode Data node to the table node.

   Note: The output table does not have any columns until you run the flow.



Step 2: Select Zip as the Geocoding Method

1. In the **Geocode Data** step, select **Zip** as the geocoding method.
2. Select the **No city** check box if your input data does not include a column for city mappings.
3. Map these fields for geocoding:
   - **City**

     Note: This field is not available if you select the **No city** check box.
   - **State/Province**

     Note: This field is not available if you select the **No city** check box.
   - **Postal code**

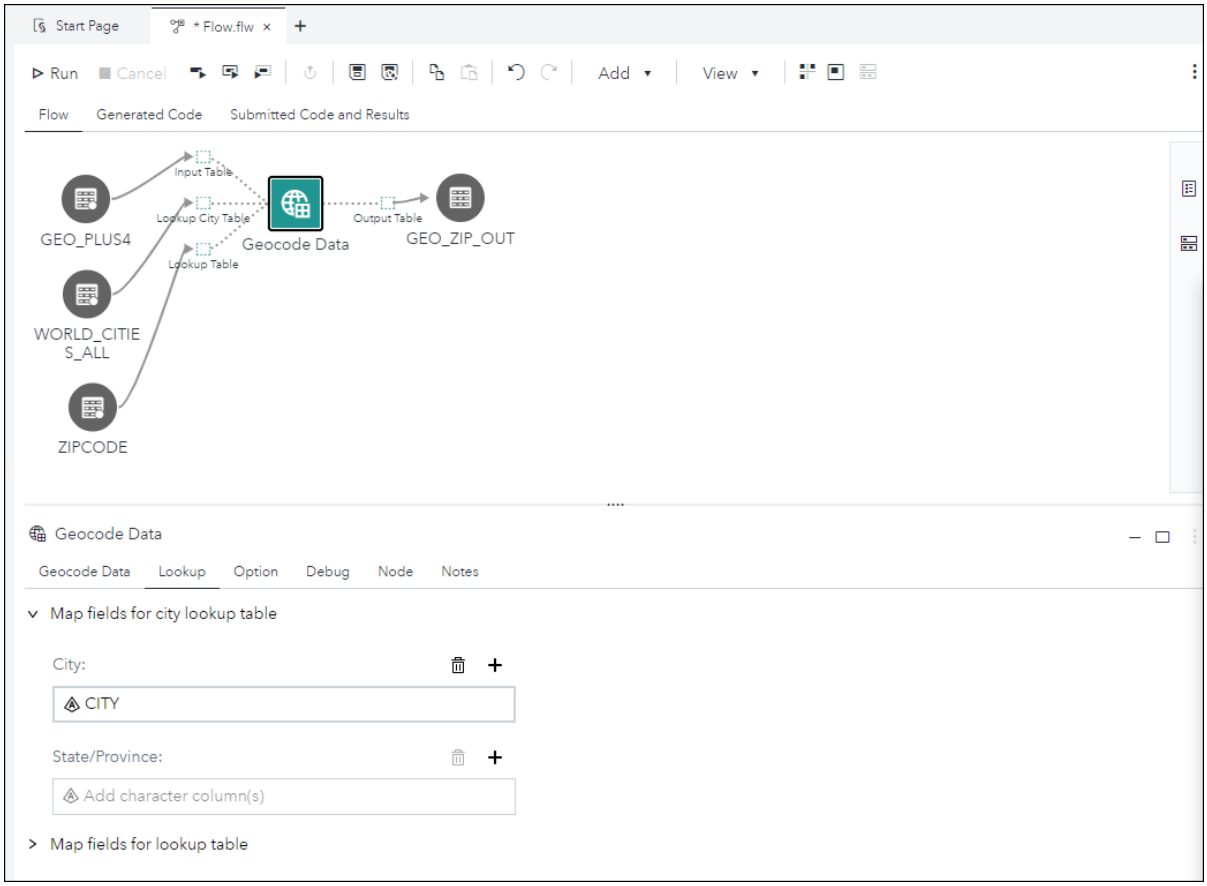
Step 3: Specify Lookup Table

For the Zip geocode method, you must specify a lookup table. If you did not select the **No city** check box on the **Geocode Data** tab, you also must add a city lookup table.

1. To view the labels for the ports on the Geocode Data node, right-click the Geocode Data node and select **Expand ports**. In the flow, the input port for the input data source (in this example, GEO_PLUS4) is labeled Input Table.
2. (Optional) Add the input port and data for the city lookup table.
   a. Right-click the Geocode node and select **Add input port** ⇨ **Lookup City Table**.
   b. Add the lookup table to the flow. (In this example, the lookup table is World_Cities_All.) Connect the lookup table node to the Lookup City Table port.
   c. On the **Lookup** tab in the Geocode Data step, map these fields for the lookup table:
      - **City**
      - **State/Province**

   Note: This step is required only if you selected the **No city** check box on the Geocode Data tab.

3. To add the input port and data for the lookup table:
   a. Right-click the Geocode Data node again and select **Add input port** ⇨ **Lookup Table**.
   b. Add the lookup table to the flow. (In this example, the lookup table is ZIPCODE.) Connect the lookup table node to the Lookup Table port.
   c. Map the **Postal code** field for the lookup table.



(Optional) Step 4: Set Additional Options

Specify any optional arguments from the GEOCODE procedure.

(Optional) Step 5: Set the Debugging Options

To add extra SAS debugging for macros, select **Debug SAS macros**.

Step 6: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.

Items to note in the output table:

- Columns from the original data source are prefixed with _dm.
- The GEOCODE procedure generates a _MATCHED_ column that indicates how the coordinates were found.

Using the Range of IP Addresses Geocode Method

About the Range of IP Addresses Geocode Method

Range geocoding matches individual address values to a lookup data set that contains a range of values. IP address data is a form of range data. IP data was not designed to be geographic like street addresses. For this reason, the process of adding coordinates to IP addresses is usually called geolocating rather than geocoding.

Generally, IP addresses are collected from visitors to websites and indicate the connection the visitor used. IP address lookup data contains information that matches ranges of IP addresses to particular geographic locations. A range of IP addresses usually belongs to a company or an internet provider. The location found is not at the street- or even ZIP-code level, but might indicate the city, state, or country where the IP address is registered.

Range geocoding with IPv4 addresses requires a lookup data set and an additional range data set. Two data sets are required because a range of addresses can map to more than one location. Range geocoding with IPv6 addresses is possible and requires a single lookup data set that contains all geographic coordinates and the ranges of IPv6 addresses.

- The lookup data set contains geographic coordinates (latitude and longitude).
- The range data set identifies the ranges (of IPv4 addresses or of other items).

A location ID key variable links the two data sets. Both data sets must contain this variable in order to identify locations for each IPv4 range. Internally, the proper range is found, and then the key value is used to access the lookup data set to find the latitude and longitude for that key. This key variable can map a range of addresses in the range data set to more than one location in the lookup data set.

Range of IP Addresses Geocoding: Step-by-Step Instructions

Step 1: Add the Geocode Data Step and Connect a Source Table

1. In the **Steps** section of the navigation pane, expand the **Enrichment** folder and double-click **Geocode Data** to add the step to your flow.
2. Connect the input port of the Geocode Data node to a data source by using a Table node or another node that creates an output table, such as a Query node,

a SAS Program node, or an Import node. For more information, see "Connecting Nodes" on page 12.

3 To save the results from the Geocode Data step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table. (In this example, the output table is named Geo_Plus4_Out.)

Next, connect the output port of the Geocode Data node to the table node.

Note: The output table does not have any columns until you run the flow.

Step 2: Select Range of IP Addresses as the Geocoding Method

1 Select **Range of IP Addresses** as the geocoding method.

2 Map these fields for geocoding:

- **Address**
- **Begin range**
- **End range**

Step 3: Specify Lookup Table

For the Range of IP Addresses geocode method, you must specify a lookup table.

1 To view the labels for the ports on the Geocode Data node, right-click the Geocode node and select **Expand ports**. In the flow, the input port for the input data source is now labeled Input Table 1.

2 Right-click the Geocode Data node again and select **Add input port** ⇨ **Lookup Table**.

3 Add a table to the flow for the lookup table. Connect this table to the Lookup Table 1 port of the Geocode Data step.

4 On the **Lookup** tab in the Geocode Data step, map these fields for the lookup table:

- **Key**
- **Longitude**

5 Right-click the Geocode Data node again and select **Add input port** ⇨ **Range Table**.

6 Add a table to the flow for the range table. Connect this table to the Range Table 1 port of the Geocode Data step.

7 Map the key value.

(Optional) Step 4: Set Additional Options

Specify any optional arguments from the GEOCODE procedure.

(Optional) Step 5: Set the Debugging Options

To add extra SAS debugging for macros, select **Debug SAS macros**.

Step 6: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.

Items to note in the output table:

- Columns from the original data source are prefixed with _dm.
- The GEOCODE procedure generates a _MATCHED_ column that indicates how the coordinates were found.

Using the Custom Geocode Method

About the Custom Geocode Method

Any data can be used as lookup data with the Custom method of geocoding. The only requirement is that you have at least three variables. The variables must be the projected or unprojected coordinate values (X and Y or LAT and LONG) and include a key variable to look up.

Custom Geocoding: Step-by-Step Instructions

Step 1: Add the Geocode Data Step and Connect a Source Table

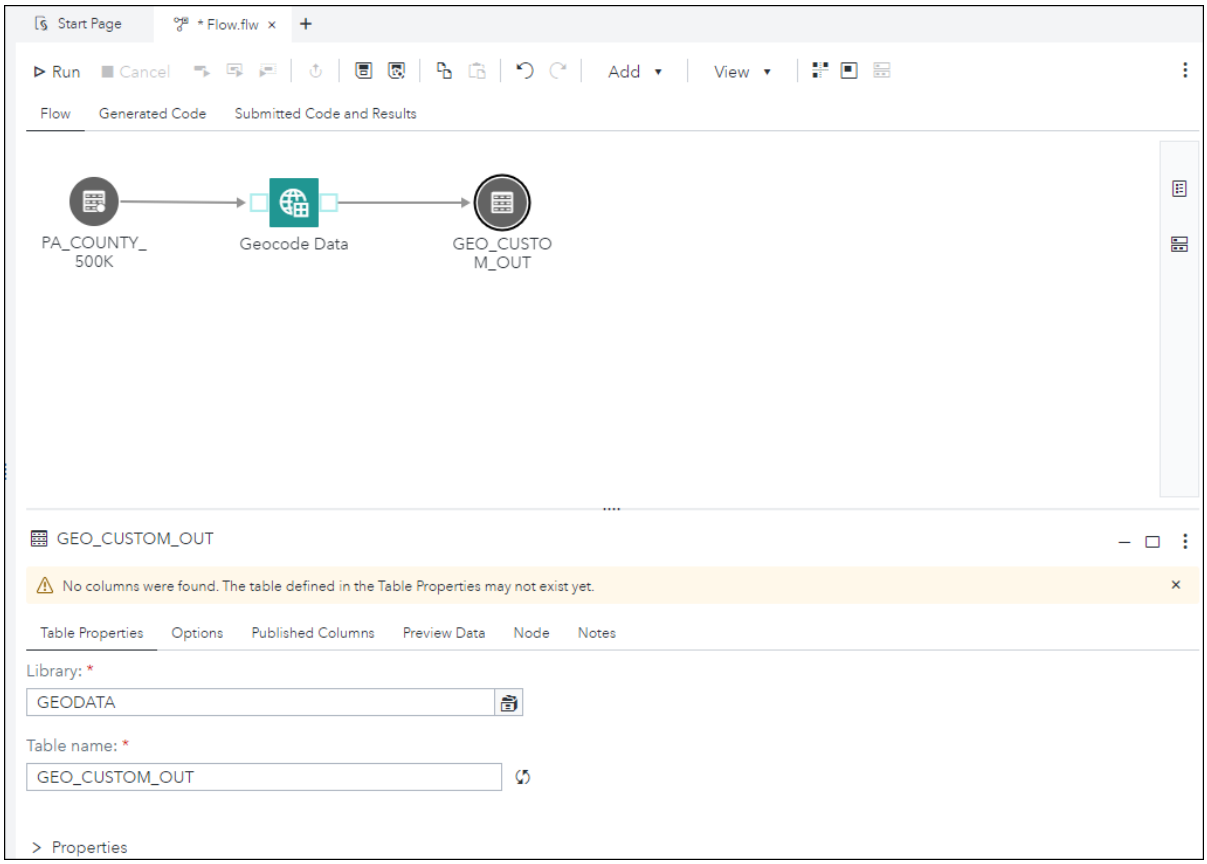
1. In the **Steps** section of the navigation pane, expand the **Enrichment** folder and double-click **Geocode Data** to add the step to your flow.
2. Connect the input port of the Geocode Data node to a data source by using a Table node or another node that creates an output table, such as a Query node,

a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

3 To save the results from the Geocode Data step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table. (In this example, the output table is named Geo_Custom_Out.)

Next, connect the output port of the Geocode Data node to the table node.

Note: The output table does not have any columns until you run the flow.



Step 2: Select Custom as the Geocoding Method

1 Select **Custom** as the geocoding method.

2 Map the **Address** field.

Step 3: Specify Lookup Table

For the Custom geocode method, you must specify a lookup table.

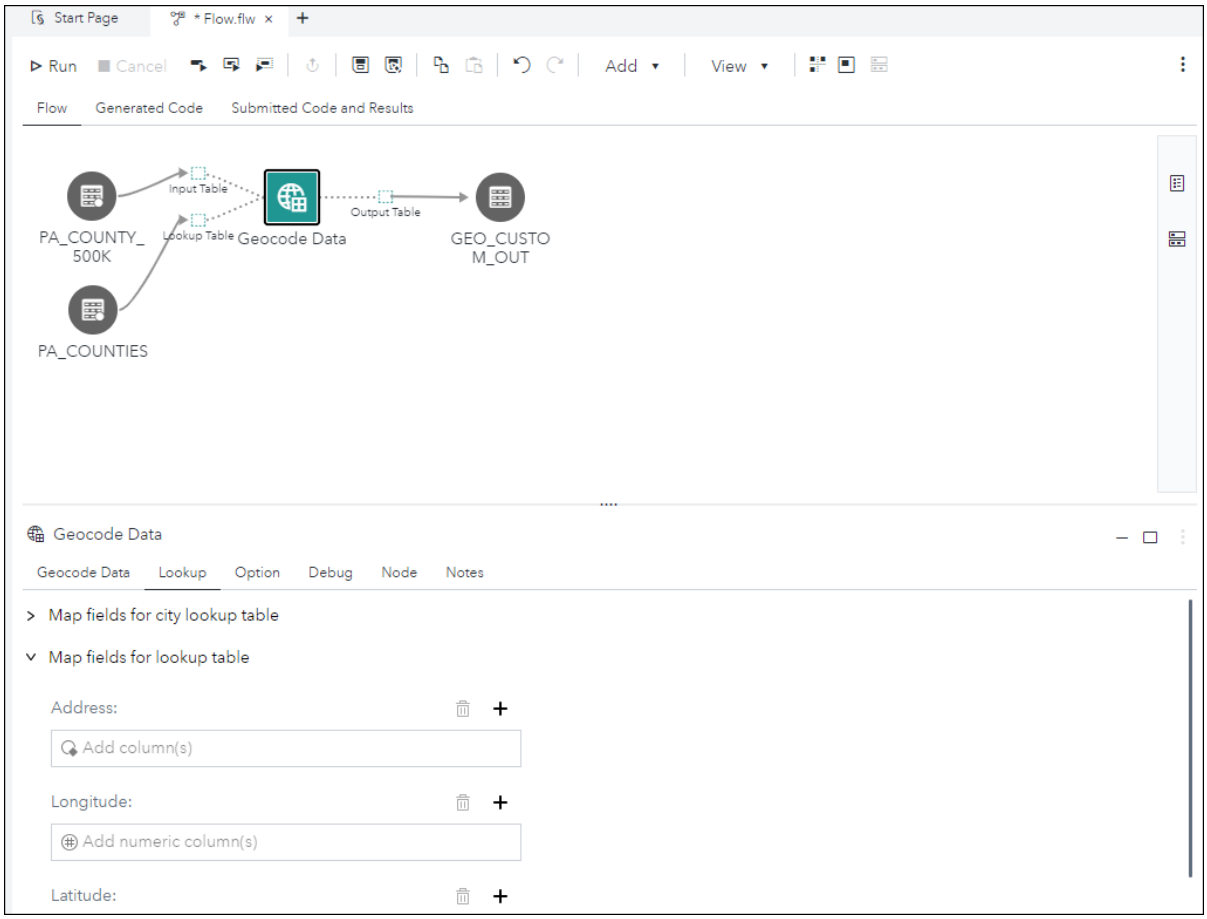
1 To view the labels for the ports on the Geocode Data node, right-click the Geocode node and select **Expand ports**. In the flow, the input port for the input data source is now labeled Input Table.

2 Right-click the Geocode node again and select **Add input port** ⇨ **Lookup Table**.

3 Add the lookup table to the flow. Connect the lookup table node to the Lookup Table port.

4 On the **Lookup** tab in the Geocode Data step, map these fields for the lookup table:

- **Address**
- **Longitude**
- **Latitude**



(Optional) Step 4: Set Additional Options

Specify any optional arguments from the GEOCODE procedure.

(Optional) Step 5: Set the Debugging Options

To add extra SAS debugging for macros, select **Debug SAS macros**.

Step 6: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.

Items to note in the output table:

- Columns from the original data source are prefixed with _dm.
- The GEOCODE procedure generates a _MATCHED_ column that indicates how the coordinates were found.

Verify & Geocode Addresses - Loqate

About the Verify & Geocode Addresses - Loqate Step

The Verify & Geocode Addresses - Loqate step enables you to verify address information and obtain geocode coordinates from the Loqate Verify Address API. Loqate is a third-party provider that specializes in address verification. The step uses PROC HTTP to connect to the Loqate Verify Address API. The Loqate API returns JSON code to SAS with the enriched data.

Note: Loqate associates a cost for each verification. For more information, see the Loqate website.

Node Connection Requirements

The Verify & Geocode Address step includes an input port and an output port. In order to run the Verify & Geocode Addresses - Loqate step, you must create the input port of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and name of the output table by connecting the output port to a Table node. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Input Port	Output Port
Program node, or an Import node	

Verify & Geocode Addresses - Loqate: Step-by-Step Instructions

Step 1: Add the Verify & Geocode Addresses - Loqate Step and Connect a Source Table

To add the Verify & Geocode Addresses - Loqate step to your flow:

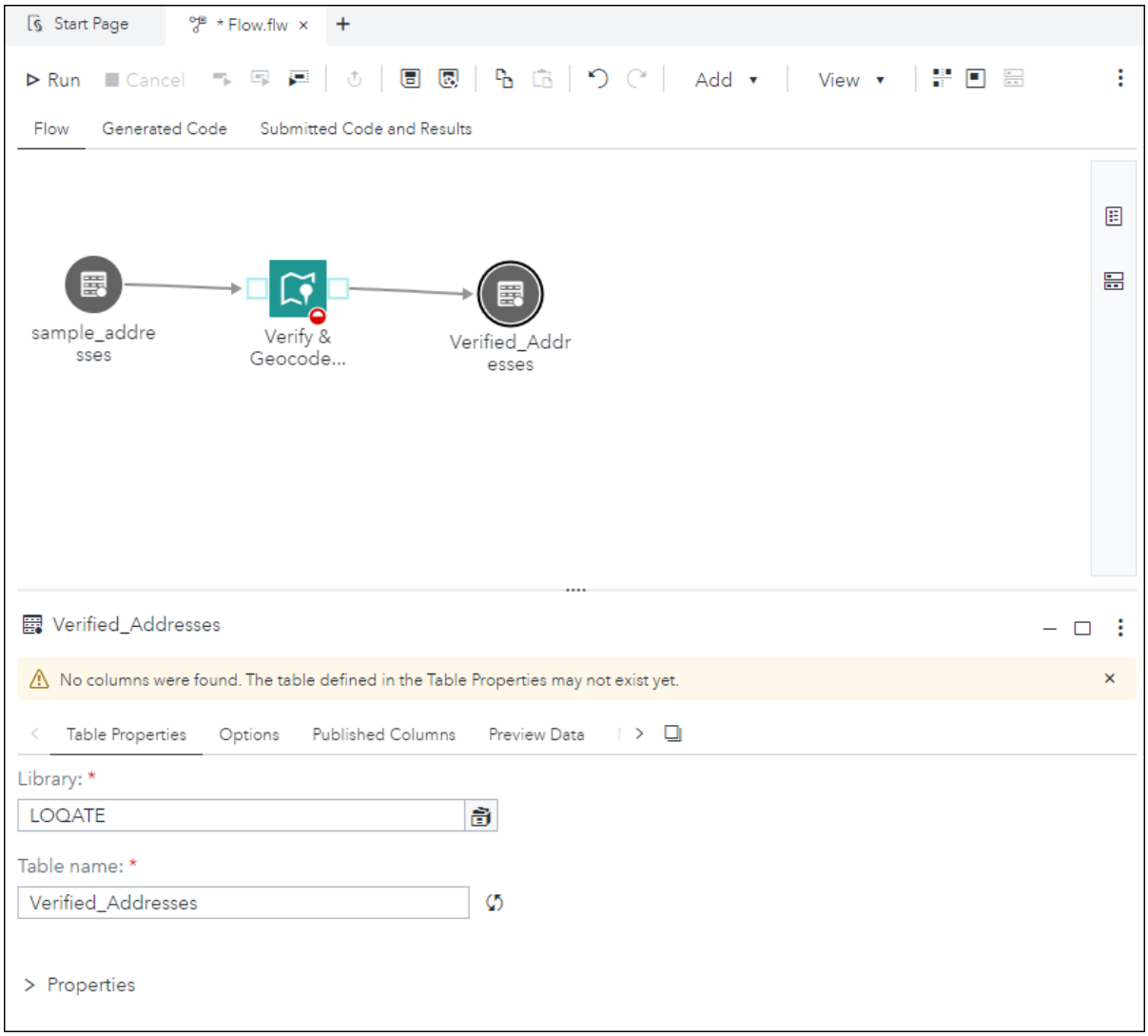
1. In the **Steps** section of the navigation pane, expand the **Enrichment** folder and double-click **Verify & Geocode Addresses - Loqate**.
2. Connect the input port of the Verify & Geocode Addresses - Loqate node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

   Note: The data source must include address and country information.

3. To save the results from the Verify & Geocode Addresses - Loqate step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table. (In this example, the output table is named Verified_Addresses.)

   Next, connect the output port of the Verify & Geocode Addresses - Loqate node to the table node.

   Note: The output table does not have any columns until you run the flow.



Step 2: Map the Fields for Address Verification

Use the options to map the columns in your data source to the corresponding fields in the Loqate database. Depending on your data, you might not specify values for all of these fields. The **Country** and **Address 1** fields are required.

(Optional) Step 3: Set Additional Options

1. To retrieve the latitude and longitude coordinates for the address from Loqate, select **Geocode the address**.
2. To standardize the country code in your data source, select **Perform country ISO standardization before processing**.

Note: The Loqate Verify Address API requires that country name or code be in an ISO 3166 two-character country code or ISO 3166 three-character country code. When you select the **Perform country ISO standardization before processing** option, SAS Studio standardizes the country value to an ISO 3166 three-character country code. If this option is not selected, the Loqate Verify Address API processes only country values in the ISO two-character code or ISO three-character code format. In the sample_addresses data source, some countries are in an ISO format (such as AUS, which is in the ISO 3166 three-character country code format), but other values (such as United States of America) are not. In order for the Loqate Verify Address API to process all these country values correctly, you need to select the **Perform country ISO standardization before processing** option.

3 Specify the number of records to send to Loqate at one time. The maximum value is 500.

4 Specify whether to replace the existing output table.

5 To set the debugging and logging options to identify and quickly troubleshoot problems in the flow before running against the Loqate Verify Address API, expand the **Advanced Options** heading.

   a To show the request from SAS and the response from Loqate in the log, select **Show API input and output JSON in the log** .

   b To test your flow without making the call to Loqate, select **Test mode**.

   Note: Because each call to the Loqate API incurs a cost, the **Test mode** option is selected by default. When you are ready to run your flow against the Loqate API, deselect this check box.

   c To view detailed debugging information, select **Show detailed log information**.

Step 4: Specify the Loqate Key

A Loqate key is required to run this step. You can get this key from the Loqate website.

Step 5: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.

Items to note in the output table:

- Columns from the SAS data source that were mapped to Loqate fields are prefixed with _dm.

- The remaining columns are from the Loqate Verify Address API. To understand the codes in your results, see the Loqate documentation.

Verify Email Addresses - Loqate

About the Verify Email Addresses - Loqate Step

The Verify Email Addresses - Loqate step enables you to verify email addresses that use the Loqate Verify Email Addresses API. Loqate is a third-party provider that specializes in email address verification. The step uses PROC HTTP to connect to the Loqate Verify Email Addresses API. The Loqate API returns a CSV file to SAS with the enriched data.

Note: Loqate associates a cost for each verification. For more information, see the Loqate website.

Node Connection Requirements

The Verify Email Addresses - Loqate step includes an input port and an output port. In order to run the Verify Email Addresses - Loqate step, you must create the input port of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and name of the output table by connecting the output port to a Table node. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Verify Email Addresses - Loqate: Step-by-Step Instructions

Step 1: Add the Verify Email Addresses - Loqate Step and Connect a Source Table

1 In the **Steps** section of the navigation pane, expand the **Enrichment** folder and double-click **Verify Email Addresses - Loqate** to add the step to your flow.

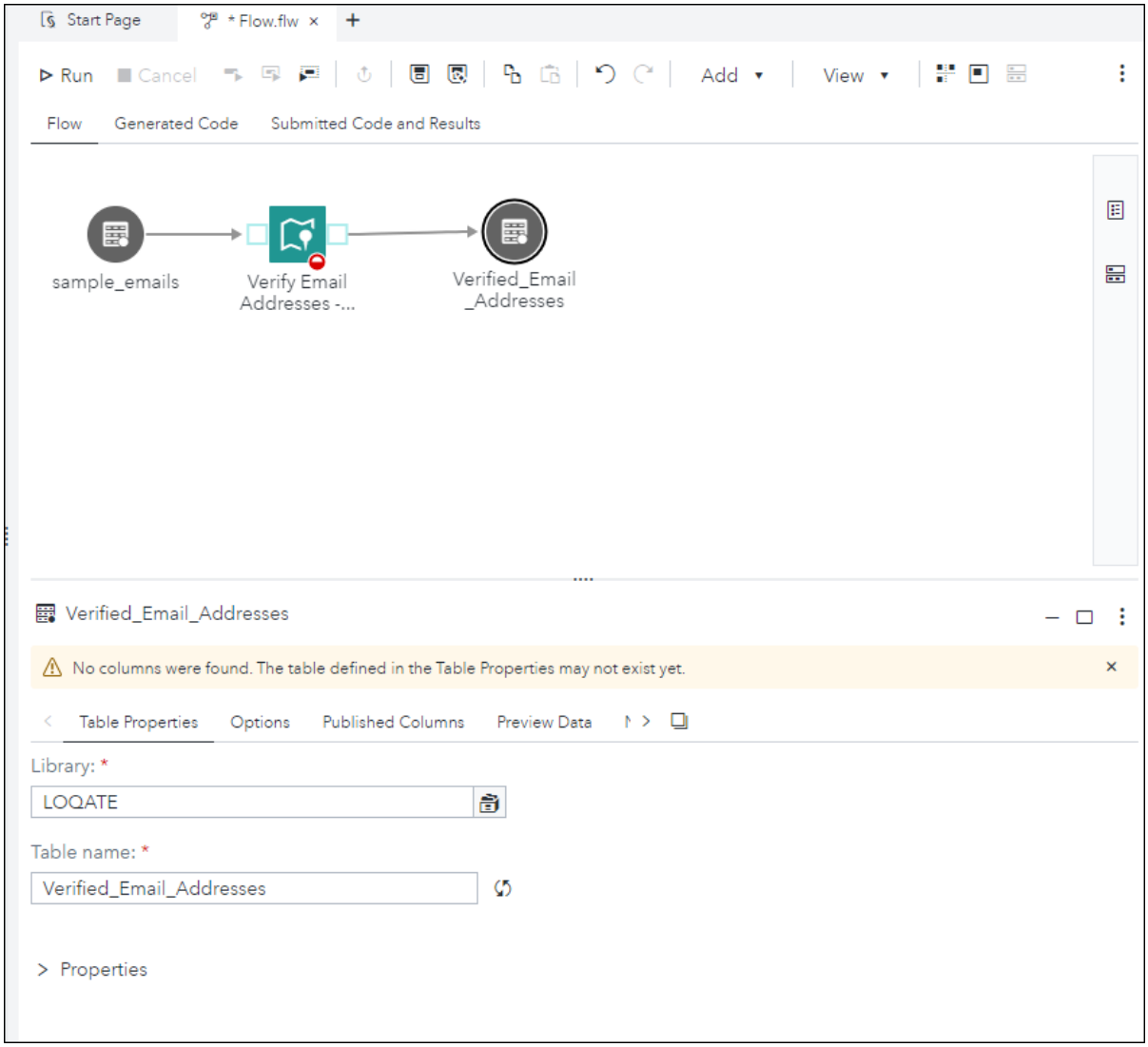
2 Connect the input port of the Verify Email Addresses - Loqate node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

Note: The data source must include email addresses.

3 To save the results from the Verify Email Addresses - Loqate step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table. (In this example, the output table is named Verified_Email_Addresses.)

Next, connect the output port of the Verify Email Addresses - Loqate node to the table node.

Note: The output table does not have any columns until you run the flow.



Step 2: Map the Fields for Email Address Verification

Use the options to map the columns in your data source to the corresponding fields in the Loqate database. The **Email address** field is required. Only English characters are supported.

For the sample_emails data source, the Email column is mapped to the **Email** field.

(Optional) Step 3: Set Additional Options

1. In the **Batch size** field, specify the number of records to send to Loqate at one time. The maximum value is 100.
2. Specify whether to replace the existing output table.

3 To set the debugging and logging options to identify and quickly troubleshoot problems in the flow before running against the Loqate Verify Email Address API, expand the **Advanced Options**.

   a To show the request from SAS and the response from Loqate in the log, select **Show API Input/Output CSV in the log** .

   b To test your flow without making the call to Loqate, select **Test mode**.

   Note: Because each call to the Loqate API incurs a cost, the **Test mode** option is selected by default. When you are ready to run your flow against the Loqate API, deselect this check box.

   c To view detailed debugging information, select **Show detailed log information**.

Step 4 Specify the Loqate Key

A Loqate key is required to run this step. You can get this key from the Loqate website.

Step 5: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.

Items to note in the output table:

- Columns from the SAS data source that were mapped to Loqate fields are prefixed with _dm.
- The remaining columns are from the Loqate API. To understand the codes in your results, see the Loqate documentation.

Verify Phone Numbers - Loqate

About the Verify Phone Numbers - Loqate Step

The Verify Phone Numbers - Loqate step enables you to verify phone numbers that use the Loqate Verify Phone Numbers API. Loqate is a third-party provider that specializes in phone number verification. The step uses PROC HTTP to connect to

the Loqate Verify Phone Numbers API. The Loqate API returns a CSV code to SAS with the enriched data.

Note: Loqate associates a cost for each verification. For more information, see the Loqate website.

Node Connection Requirements

The Verify Phone Numbers - Loqate step includes an input port and an output port. In order to run the Verify Phone Numbers - Loqate step, you must create the input port of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and name of the output table by connecting the output port to a Table node. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Verify Phone Numbers - Loqate: Step-by-Step Instructions

Step 1: Add the Verify Phone Numbers - Loqate Step and Connect a Source Table

To add the Verify Phone Numbers - Loqate step to your flow:

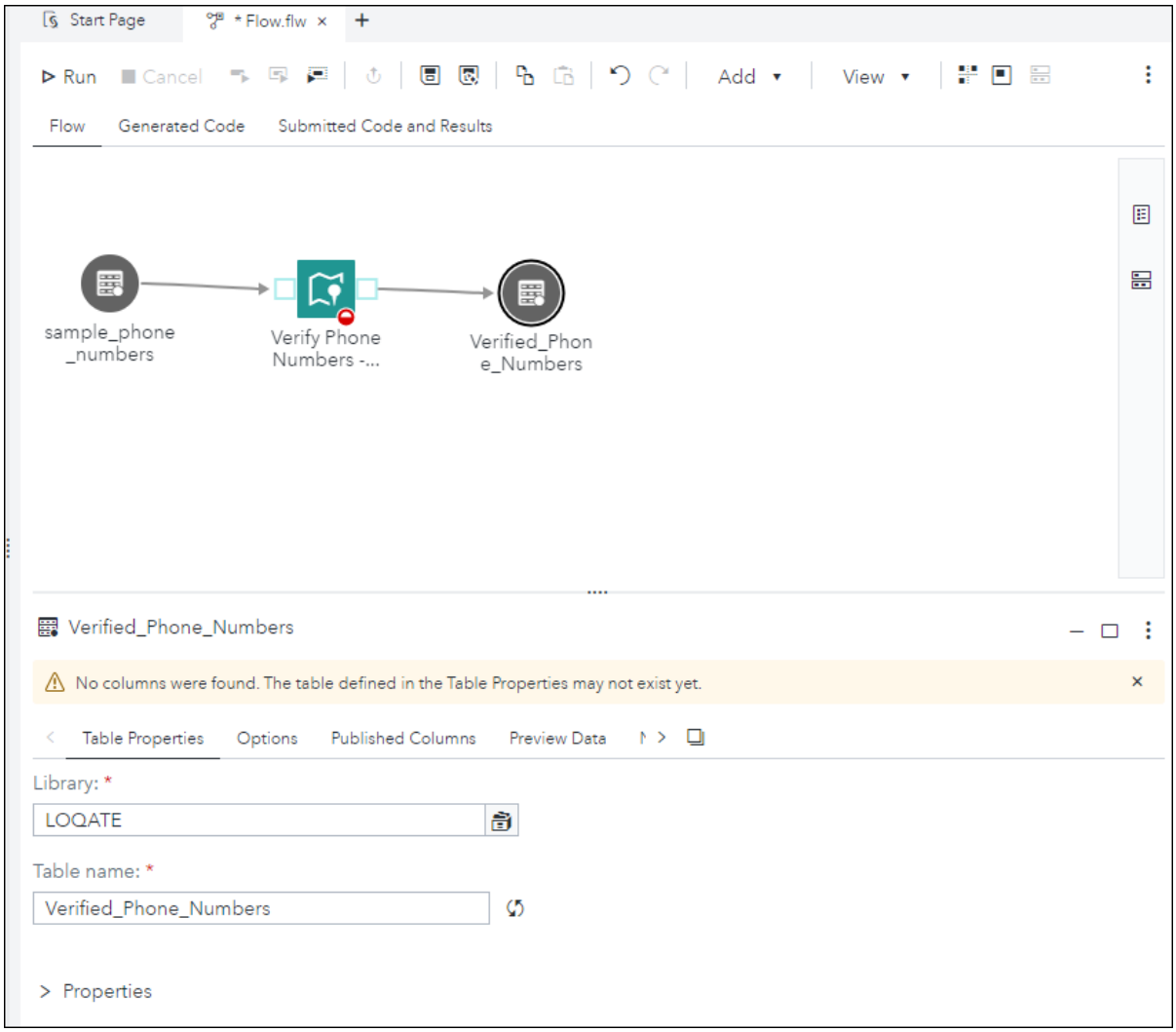
1. In the **Steps** section of the navigation pane, expand the **Enrichment** folder and double-click **Verify Phone Numbers - Loqate**.
2. Connect the input port of the Verify Phone Numbers - Loqate node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

   Note: The data source must include phone information.

3 To save the results from the Verify Phone Numbers - Loqate step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table. (In this example, the output table is named Verified_Phone_Numbers.)

Next, connect the output port of the Verify Phone Numbers - Loqate node to the table node.

Note: The output table does not have any columns until you run the flow.



Step 2: Map the Fields for Phone Verification

Use the options to map the columns in your data source to the corresponding fields in the Loqate database. The **Phone number** field is required.

For the sample_phone_numbers data source, the phone column is mapped to the **Phone number** field, and the country column is mapped to the **Country** field.

(Optional) Step 3: Set Additional Options

1 To standardize the country value to an ISO 3166 two-character country code, select the **Perform country ISO standardization before processing** option.

Note: The Loqate Verify Phone API requires that country name be in an ISO 3166 two-character country code format. If this option is not selected, the Loqate Verify Phone Numbers API processes only country values in the ISO two-character code format.

In the sample_phone_numbers data source, none of the country values are in an ISO 3166 two-character country code format. In order for the Loqate Verify Phone Numbers API to process all these country values correctly, you need to select the **Perform country ISO standardization before processing** option.

2 To set the debugging and logging options to identify and quickly troubleshoot problems in the flow before running against the Loqate Verify Phone Numbers API, expand the **Advanced Options** heading.

a To show the request from SAS and the response from Loqate in the log, select **Show API input and output CSV in the log** .

b To test your flow without making the call to Loqate, select **Test mode**.

Note: Because each call to the Loqate API incurs a cost, the **Test mode** option is selected by default. When you are ready to run your flow against the Loqate API, deselect this check box.

c To view detailed debugging information, select **Show detailed log information**.

Step 4: Specify the Loqate Key

A Loqate key is required to run this step. You can get this key from the Loqate website.

Step 5: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab.

Items to note in the output table:

- Columns from the SAS data source that were mapped to Loqate fields are prefixed with _dm.
- The remaining columns are from the Loqate API. To understand the codes in your results, see the Loqate documentation.

7

Integrating Data

Execute Decisions

About the Execute Decisions Step

You can use the Execute Decisions step to add a published decision from SAS Intelligent Decisioning to your flow. Decisions enable you to create a database of rules, combine those rules into decisions, and publish the decisions for use by other applications such as SAS Studio. For more information, see "Introduction to SAS Intelligent Decisioning" in *SAS Intelligent Decisioning: User's Guide*.

To use a decision in SAS Studio, the decision must meet the following criteria:

- The decision must be published using SAS Intelligent Decisioning.
- The decision must be published to a CAS destination.
- The input table must be a CAS table.

Note: By default, the output data is written to the output port as a session-scoped CAS table in the same CAS library as the source table.

Note: This step is available only if your site licenses SAS Studio Engineer.

Node Connection Requirements

In order to run the Execute Decisions step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node	No connection is required.
Note: The input table for the Execute Decisions node must be a CAS table.	**Note:** Execute Decisions node output can be connected to any flow node that accepts a SAS or CAS table as an input, such as a Table node, a Query node, or a SAS Program node. By default, the output data is written to a session-scoped CAS table in the same CAS library as the source table. You can specify the library and name of the output tables by connecting the output port to a Table node. It is recommended that a CAS output table reside in the same CAS session as the input table.For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Execute Decisions: Step-by-Step Instructions

The Execute Decisions step requires one input port and one output port.

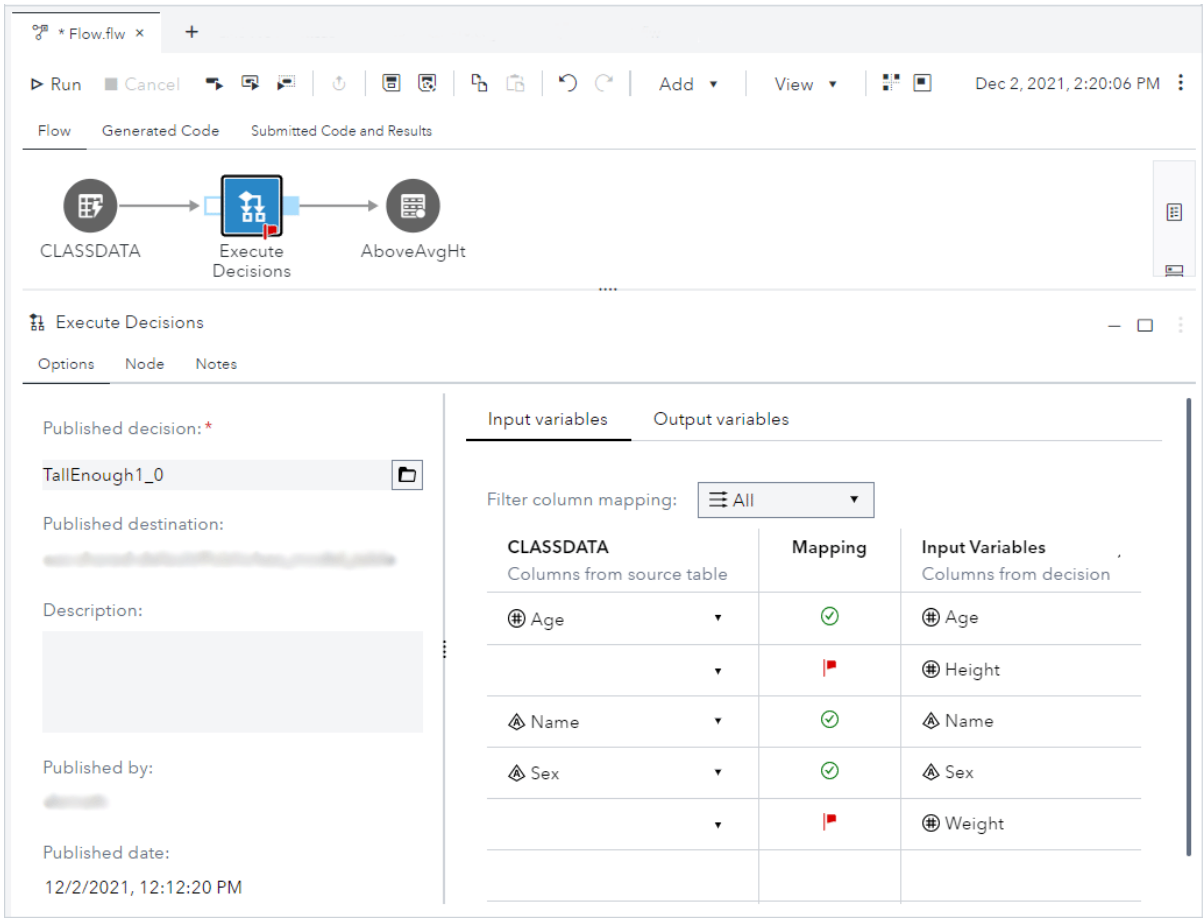
1. In the **Steps** section of the navigation pane, expand the **Integrate** folder and double-click **Execute Decisions**.
2. Click the **Execute Decisions** node, and then use the Options tab to select the decision that you want to use. To select a decision, click 🗀. In the **Select a Published Decision** box, specify the CAS destination and decision that you want to use.
3. Connect the input port of the Execute Decisions node to the appropriate data source for the selected decision. The input data source must be a Table node that references a CAS table. For more information, see “Connecting Nodes” on page 12.

4 Click the **Execute Decisions** node to verify the column mapping for the decision. In the right pane of the Options tab, use the Input variables tab to view the mapping between the columns in the input table and the columns that are required by the decision. Columns are mapped automatically by name and data type.

Note: If the length of a column in the input table is different from the expected length of the corresponding column in the decision, the length of the column in the decision output table is overwritten, by default. If the length of an input column is shorter than the length of the output column, values in the output table can be truncated. You can use the `sas.decisions.variableLengthOverridden` property in SAS Environment Manager to keep the length of the output column. For more information, see "sas.decisions.variableLengthOverridden" in *SAS Intelligent Decisioning: Administrator's Guide*.

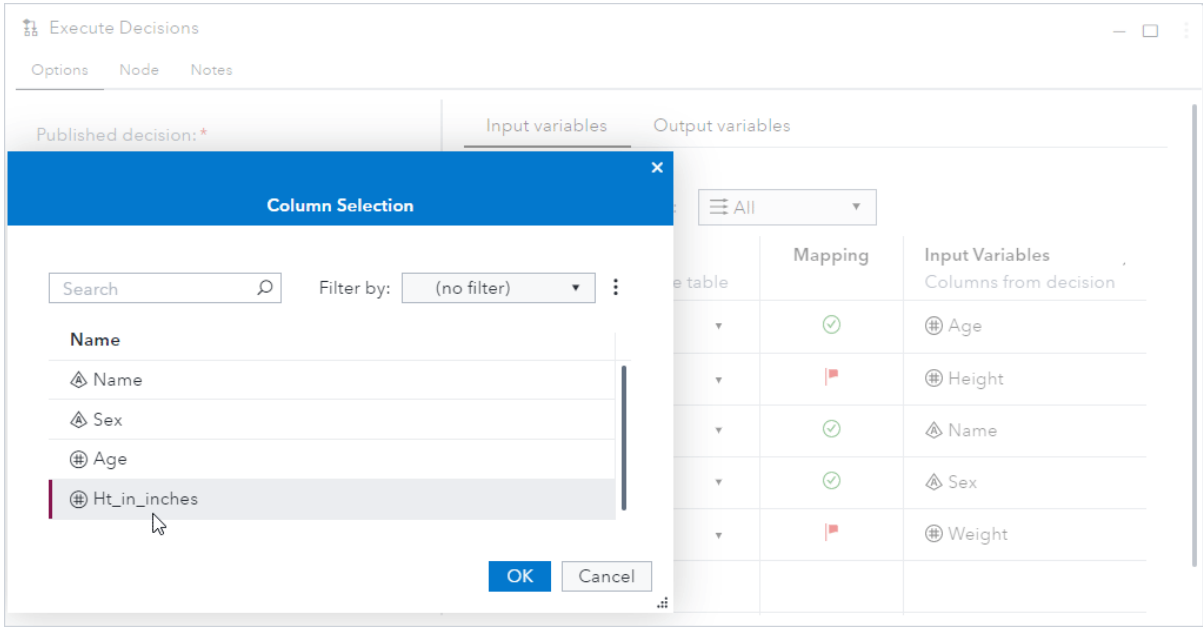
Use the **Filter column mapping** drop-down list to filter the mappings that are displayed. You can filter the mappings by the following categories:

- Successful (✓) - the input column is successfully mapped to a column that is used in the decision.
- Unresolved (⚑) - an input column is not yet mapped to the column in the decision. An unresolved column can occur when SAS Studio cannot automatically map columns based on name and data type. You must manually map any unresolved column mappings.



5 If you have any unresolved column mappings, click ▾ beside the unmapped column and select the appropriate column to map to the column in the decision.

Note: The Execute Decisions node can run only when all columns are mapped successfully.



6 To view the output columns for the decision, click the **Output variables** tab.

7 To run the flow, click ▷ Run.

Implement SCD

About the Implement SCD Step

The Implement SCD step enables you to use the slowly changing dimensions (SCD) process to load data into target dimension tables. The data changes slowly, rather than changing on a time-based, regular schedule. The target tables are structured so that they can retain a history of changes to their data. This record of data changes can provide a basis for analysis. You can also choose to overwrite your data with new values and not preserve the historical data.

For example, a table named Customers might have columns for customer ID, home address, age, and income. Each time the address or income changes for a customer, a new row could be created for that customer, and the old row could be retained. This historical record of changes could be combined with purchasing information to forecast buying trends and to direct customer marketing campaigns.

SAS Studio supports two types of slowly changing scenarios:

- Type 1 SCD - no history of data changes. Type 1 SCD overwrites the specified columns in the target table without retaining a history of changes. Type 1 SCD is useful for maintaining columns that are not used in historical analysis.
- Type 2 SCD - history of data changes is maintained. Type 2 SCD maintains multiple records for each business key in the target table. The latest entry is the current entry for that business key. Other rows comprise the historical record of data changes.

Note: This step is available only if your site licenses SAS Studio Engineer.

There are seven main steps to use the Implement SCD step:

1. Add the Implement SCD step to your flow and connect the source table.
2. Specify the library and name of the target table.
3. Specify one or more key columns that are used to match the rows between the source and target tables.
4. (Optional) Specify columns from the target table for SCD Type 1 changes. The selected columns are updated with values from the source table.
5. (Optional) Specify columns from the target table for SCD Type 2 changes. The selected columns are used in the new current row that is created. The columns are populated with data from the corresponding columns in the source table.
6. (Optional) Specify columns, which have not been selected for SCD Type 1 or Type 2 changes, to include when a new row is added to the target table.
7. Run the flow to update the target table.

Note: In order to run the Implement SCD step, you must specify either SCD Type 1 or SCD Type 2 changes or both.

Node Connection Requirements

The Implement SCD node includes only one input port. The target table is specified in the node details rather than with an output port. In order to run the Implement SCD step, you must create a connection to the input port of the node as indicated here:

Input Port	Output Port
■ Table node. The source and target tables must reside on the same database server. The tables must reside in one of the following databases: □ Oracle □ Singlestore □ Snowflake □ SQL Server	Not applicable. The target table is specified in the node details.

Input Port	Output Port
□ Teradata	

Implement SCD: Step-by-Step Instructions

Step 1: Add the Implement SCD Step and Connect a SourceTable

To add the Implement SCD step to your flow:

1. In the **Steps** section of the navigation pane, expand the **Integrate** folder and double-click **Implement SCD**.
2. Connect the input port of the Implement SCD node to a data source by using a Table node. The table must reside in one of the following databases: Oracle, Singlestore, Snowflake, SQL Server, or Teradata. For more information, see “Connecting Nodes” on page 12.

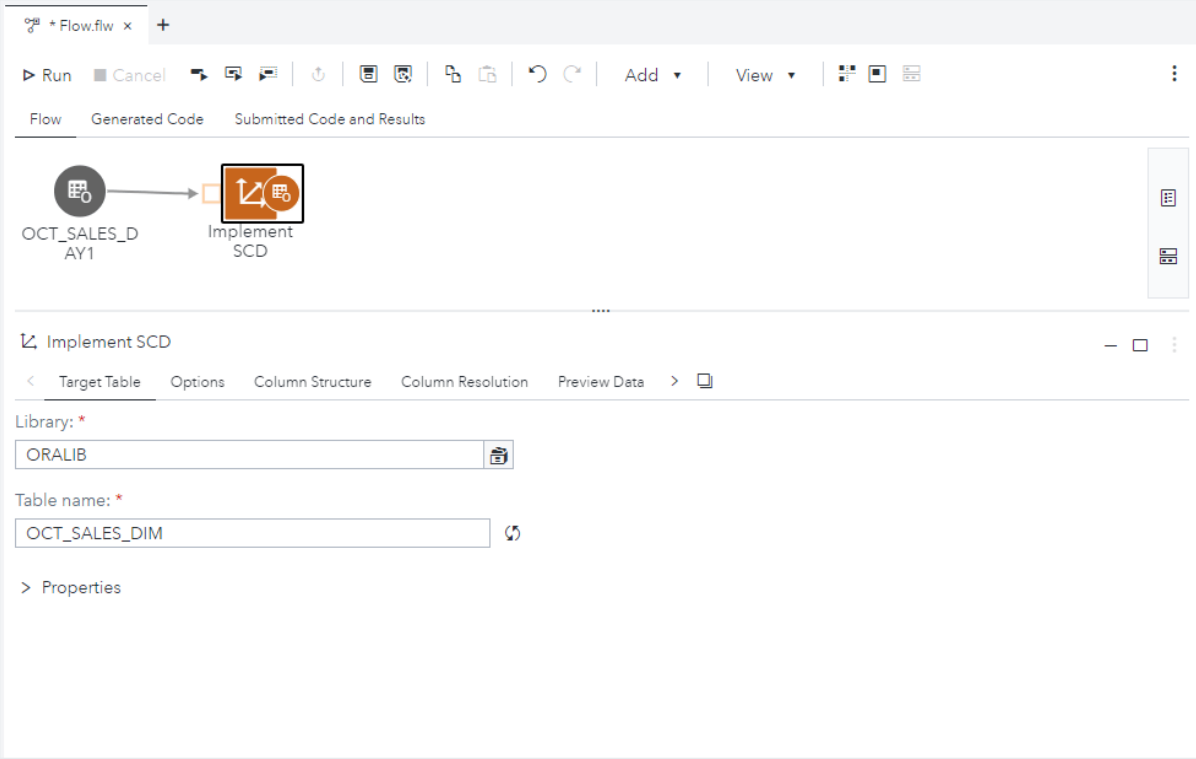
   Note: If you are connecting to Singlestore data, you must specify `s2` as the SAS/ACCESS engine name and set DEFAULT_AUTH_PLUGIN=mysql_native_password in the LIBNAME statement. For more information, see “DEFAULT_AUTH_PLUGIN=” in *SAS/ACCESS for Relational Databases: Reference*.

Step 2: Specify the Target Table

The source and target tables must reside on the same database server.

1. Click the **Implement SCD** node, and then click the **Target Table** tab.
2. Use the **Library** and **Table name** boxes to specify the target table.

   Note: The Implement SCD node does not have an output port. You must specify the target table in the Implement SCD node details. However, you can connect the Implement SCD node to another operational node that requires a table as input.



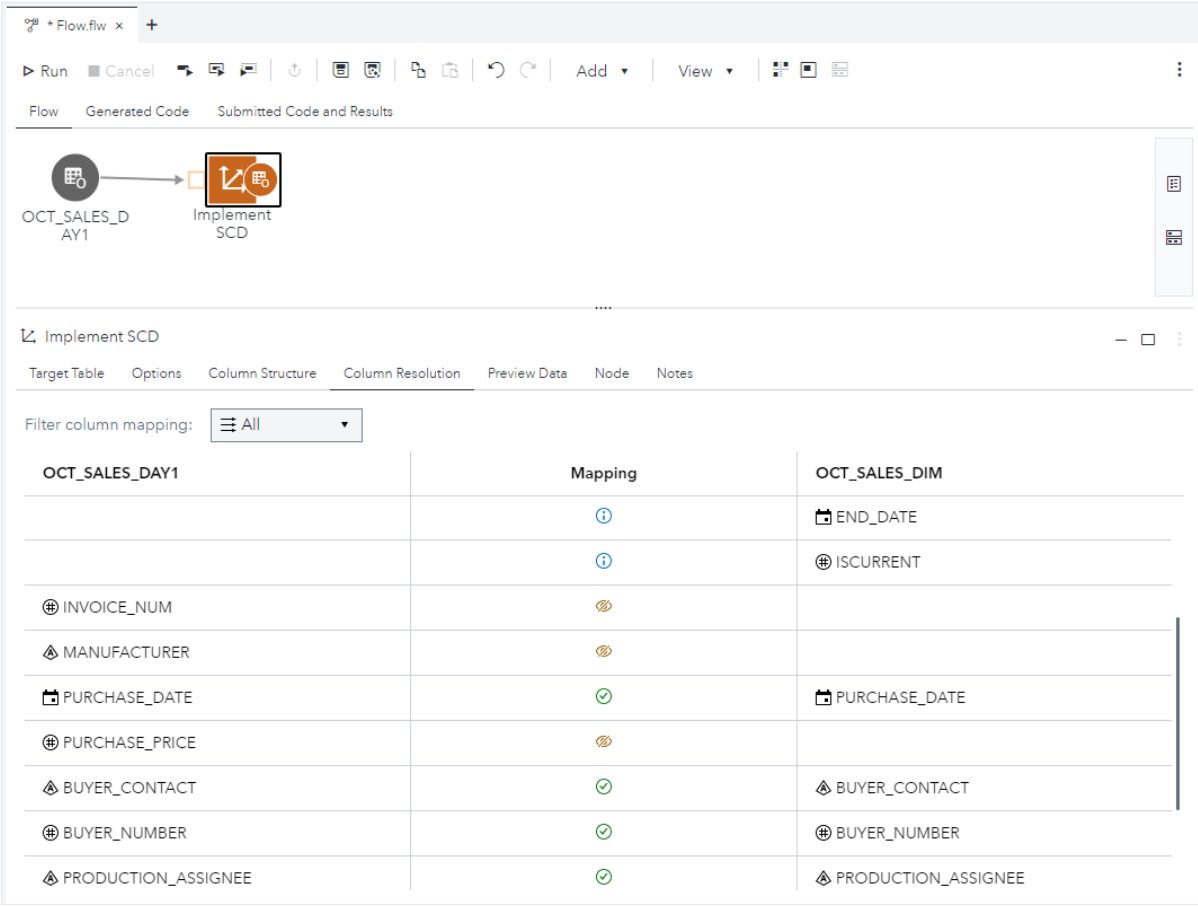
3 To view the structure of the target table, click the **Column Structure** tab.

To view the column mapping between the source and target tables, click the **Column Resolution** tab.

Note: On the Column Resolution tab, you can use the **Filter column mapping** drop-down list to filter the mappings that are displayed. You can filter the mappings by the following categories:

- Successful (✓) - the source column is successfully mapped to a column in the target table.
- Ignored (👁) - the source column does not have a corresponding column in the target table. Data for this column is ignored in the target table.
- Information (ⓘ) - a source column is not mapped to the column in the target table. An unresolved column can occur when SAS Studio cannot automatically map columns based on name and data type.

In this example, there are two columns in the target table that do not have corresponding columns in the source table, and there are three columns in the source table that do not have corresponding columns in the target table.



Step 3: Specify One or More Key Columns

You must specify at least one business key column to run the Implement SCD step.

Note: To preview the expression that is generated by SAS Studio, click **Preview expression**. You can also copy the code to use in a SAS program. This option is available only when you have specified one or more key columns and all of the options for at least one SCD Type 1 or Type 2 change.

1. Click the **Options** tab and select **Identify business keys**.
2. To add a key column, click **Select columns**.
3. In the Select Columns window, select one or more key columns. You must select at least one key column that uniquely identifies each row in the source table. Do not select all of the columns in the list. Click **OK**.
4. SAS Studio attempts to match key columns in the source and target tables based on column name, regardless of case and data type. If a match is not found, click to select the source business key column. Click **OK**.

(Optional) Step 4: Specify the SCD Type 1 Changes

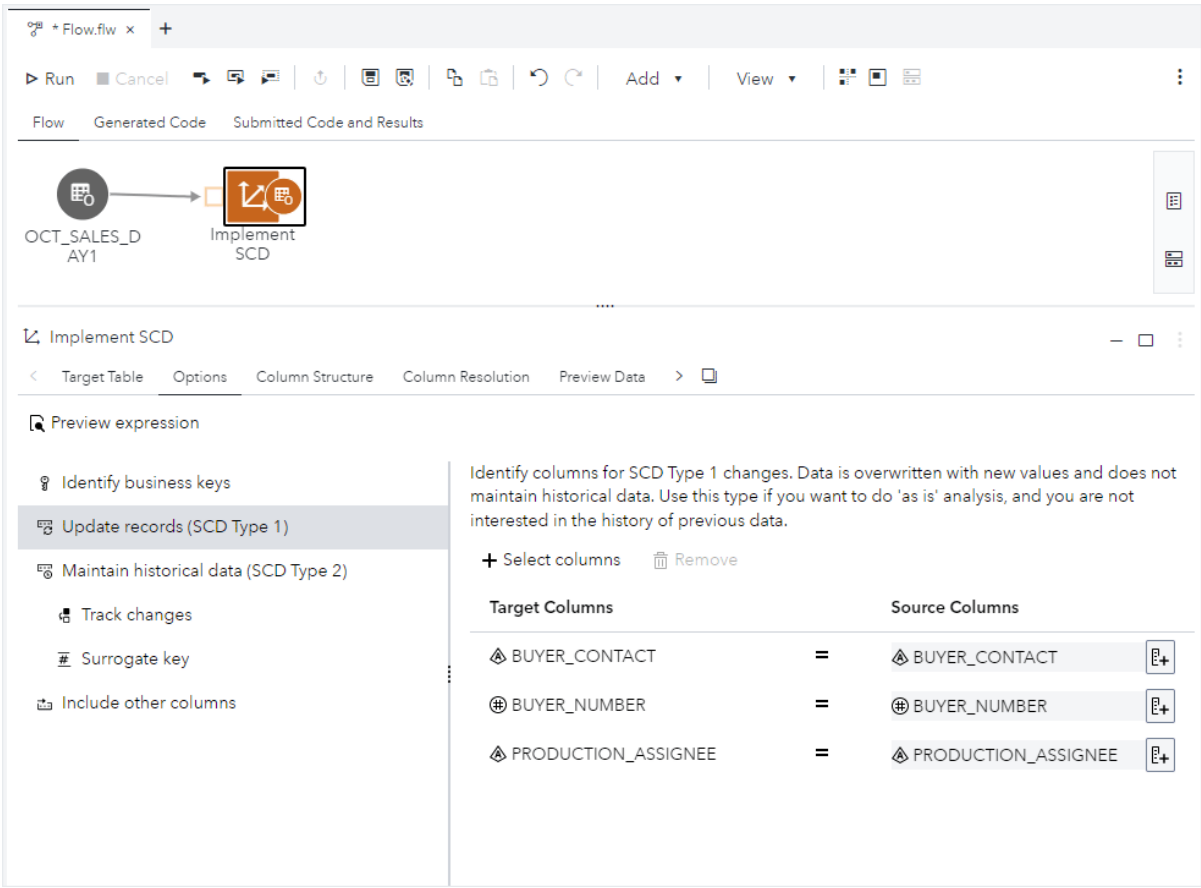
You can specify columns of data in the target table to overwrite with new data. SCD Type 1 changes do not preserve the historical data.

Note: You cannot update multiple Type 1 target columns with the same source column. Each source column can be used to update only one Type 1 target column.

1. Click the **Options** tab and select **Update records (SCD Type 1)**.
2. To select columns to update, click **Select columns**.
3. In the Select Columns window, select one or more target columns to update in the target table. Click **OK**.

   Note: Any columns that you have selected for other options are not available.

4. SAS Studio attempts to match columns in the source and target tables based on column name, regardless of case and data type. If a match is not found, click to select the column that you want to use from the source table. Click **OK**.



(Optional) Step 5: Specify the SCD Type 2 Changes

You can specify columns to insert in a new row in the target table for Type 2 changes. A new row is created in the target table and marked as the current row, and the old rows are preserved.

Note: You cannot match multiple Type 2 target columns to the same source column. Each source column can be matched with only one Type 2 target column.

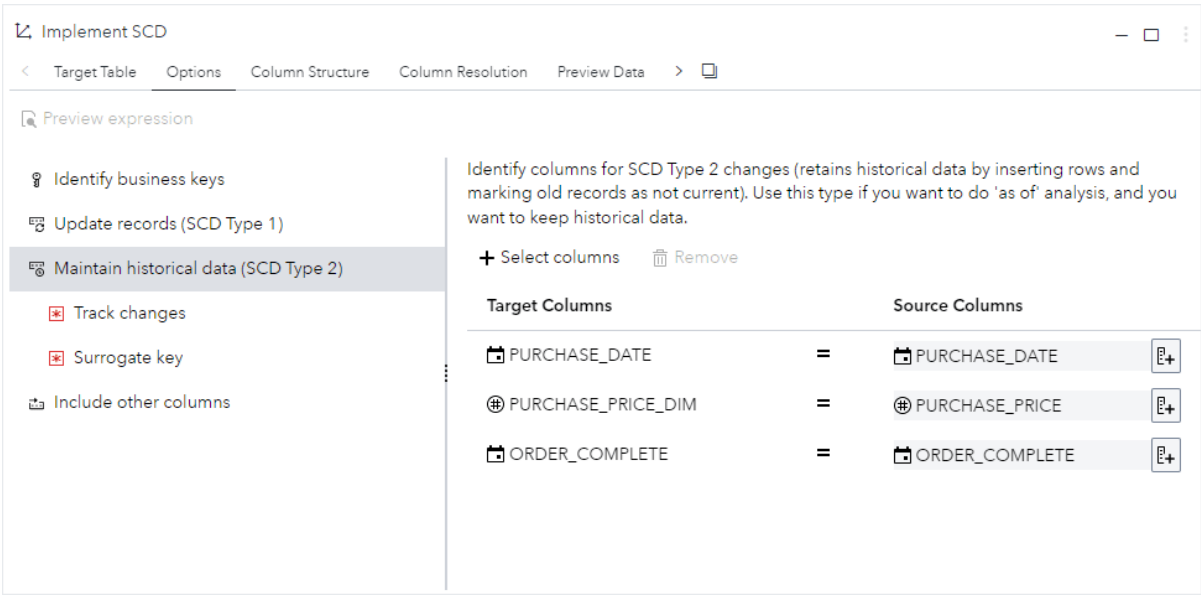
1 Click the **Options** tab and select **Maintain historical data (SCD Type 2)**.

2 To select columns to add to the new row, click **Select columns**.

3 In the Select Columns window, select one or more target columns. Click **OK**.

Note: Any columns that you have selected for other options are not available.

4 SAS Studio attempts to match columns in the source and target tables based on column name, regardless of case and data type. If a match is not found, click to select the column that you want to use from the target table. Click **OK**.



5 To specify how the progression of changes is tracked, click **Track changes**. You must specify both the effective date and flagging methods.

a In the **Effective Date Method** area, specify the column that contains the start date timestamp that identifies the effective start date and time for the row.

Note: The column that you specify for the start date must be either a date column or a numeric column that contains date values. If the column does not contain date values, the flow cannot run.

Use the **Start on** field to specify the timestamp when the row was created. The default value is the current date.

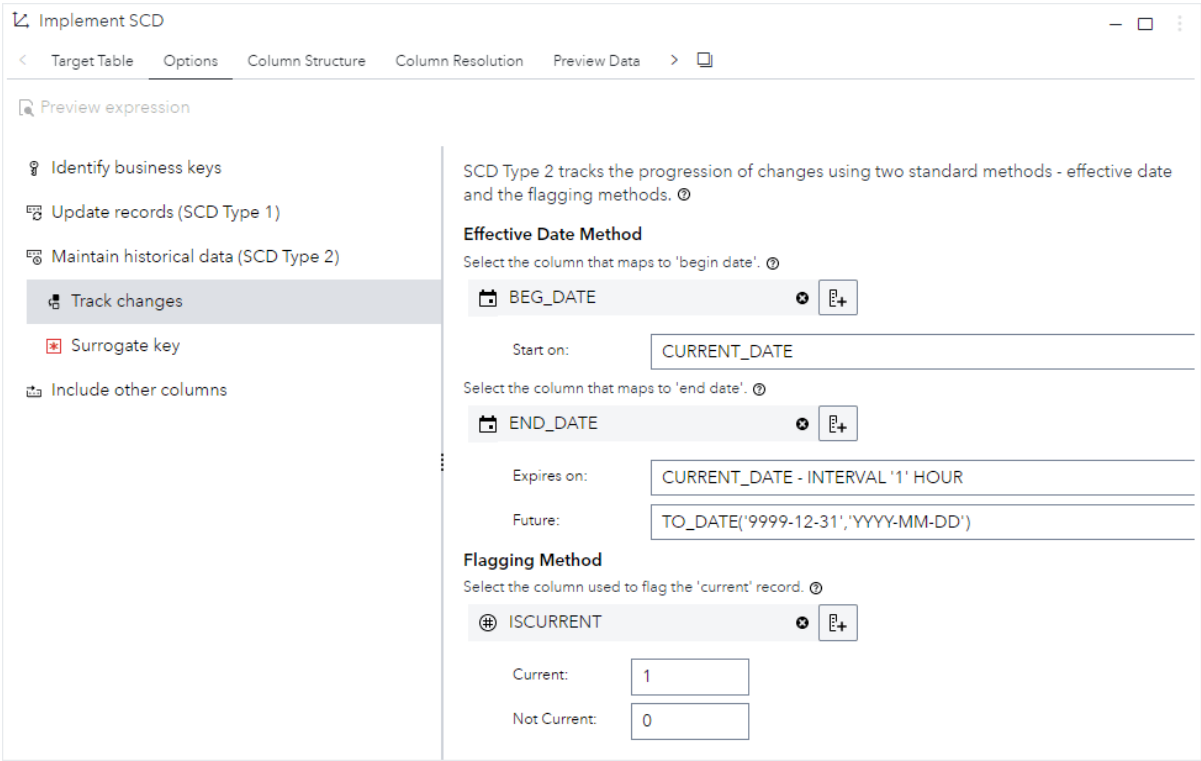
b Specify the column that contains the end date that identifies the current row.

Note: The column that you specify for the end date must be either a date column or a numeric column that contains date values. If the column does not contain date values, the flow cannot run.

Use the **Expires on** field to specify an expression that defines the end date value when closing out a row that was previously current. The default expression sets the effective end date timestamp to one hour earlier than the effective start date timestamp of the newly created current row.

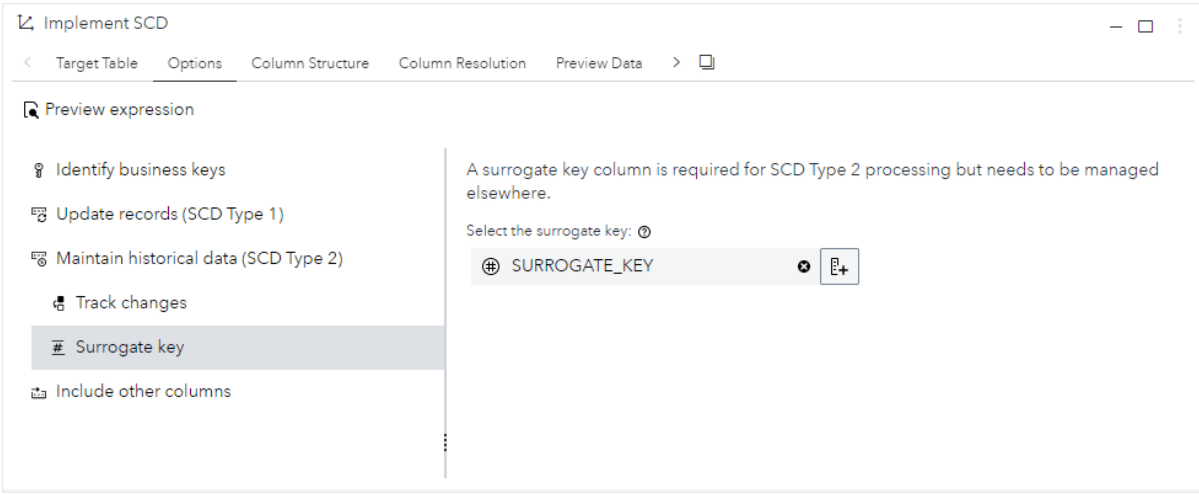
Use the **Future** field to specify the end date value for a row that is current. Typically, the future date value is set far into the future to ensure that the end date value does not expire while the row is still current. The default value is `9999-12-31`.

c In the **Flagging Method** area, specify the column that contains the Boolean-type value that identifies the current row. Use the **Current** and **Not Current** fields to specify the values that are used to label the rows. By default, for character columns, the values of `Y` and `N` are used to identify current and historical rows. For numeric columns, the default values are `1` and `0`.



6 Click **Surrogate key** to specify the surrogate key in the target dimension table. To select the surrogate key column, click and select the target column. Click

OK. The surrogate key column values are system-generated values that are used to uniquely identify a row in the table. The surrogate key and its values are required for Type 2 changes and must be generated outside of SAS Studio.



(Optional) Step 6: Specify Other Columns to Include in the Target Table

When new rows are created, you can include additional columns that have not already been identified as a surrogate key, business key, Type 1 column, Type 2 column, effective start and end date column, or current record indicator column. Columns that are not selected are added as missing values when new rows are created in the target table.

1 Click the **Options** tab and select **Include other columns**.

2 To select columns to update, click **Select columns**.

3 In the Select Columns window, select one or more columns to update in the target table. Click **OK**.

Note: The columns that you have already selected for other options are not available.

Step 7: Run the Flow and Update the Target Table

To run the Implement SCD step and update the target table, click ▷ Run.

Load Table

About the Load Table Step

The Load Table step enables you to load a source table into a target table. When you use the Load Table step, you can control how data is loaded into the target table. You can choose to insert new source rows into the target table, update existing rows in the target table, or both. You can also control how existing rows in the target table are removed before new rows are inserted.

For example, you might have a sales table that you need to update periodically with new pricing data. You can use the Load Table step to update the price column of the sales table by using the product ID as the key column.

When you load a table, you can choose the load technique that you want to use:

- Insert rows – inserts all of the rows in the source table into the output table.

  Note: If you are inserting rows from the source table into the target table, you can specify a preprocessing action that determines how existing rows in the target table are removed before rows are inserted.

- Update rows – updates existing rows in the target table by using matching rows from the source table based on one or more key columns.
- Upsert rows – inserts new rows into the target table and updates existing rows in the target table based on one or more key columns.

Note: This step is available only if your site licenses SAS Studio Engineer.

There are five main steps to use the Load Table step:

1. Add the Load Table step to your flow and connect the source table.

   Note: The Load Table node does not have an output port. You must specify the target table in the Load Table node details.

2. Specify the library and name of the target table.
3. Select the load technique that you want to use: Insert rows, Update rows, or Upsert rows. If you are using the Insert rows technique, you can also specify a preprocessing action. If you are using the Update rows or Upsert rows techniques, you must specify at least one key column.
4. (Optional) Specify output table options.
5. Run the flow to load the data.

Node Connection Requirements

The Load Table step includes only one input port. The target table is specified in the node details rather than with an output port. In order to run the Load Table step, you must create a connection to the input port of the node as indicated here:

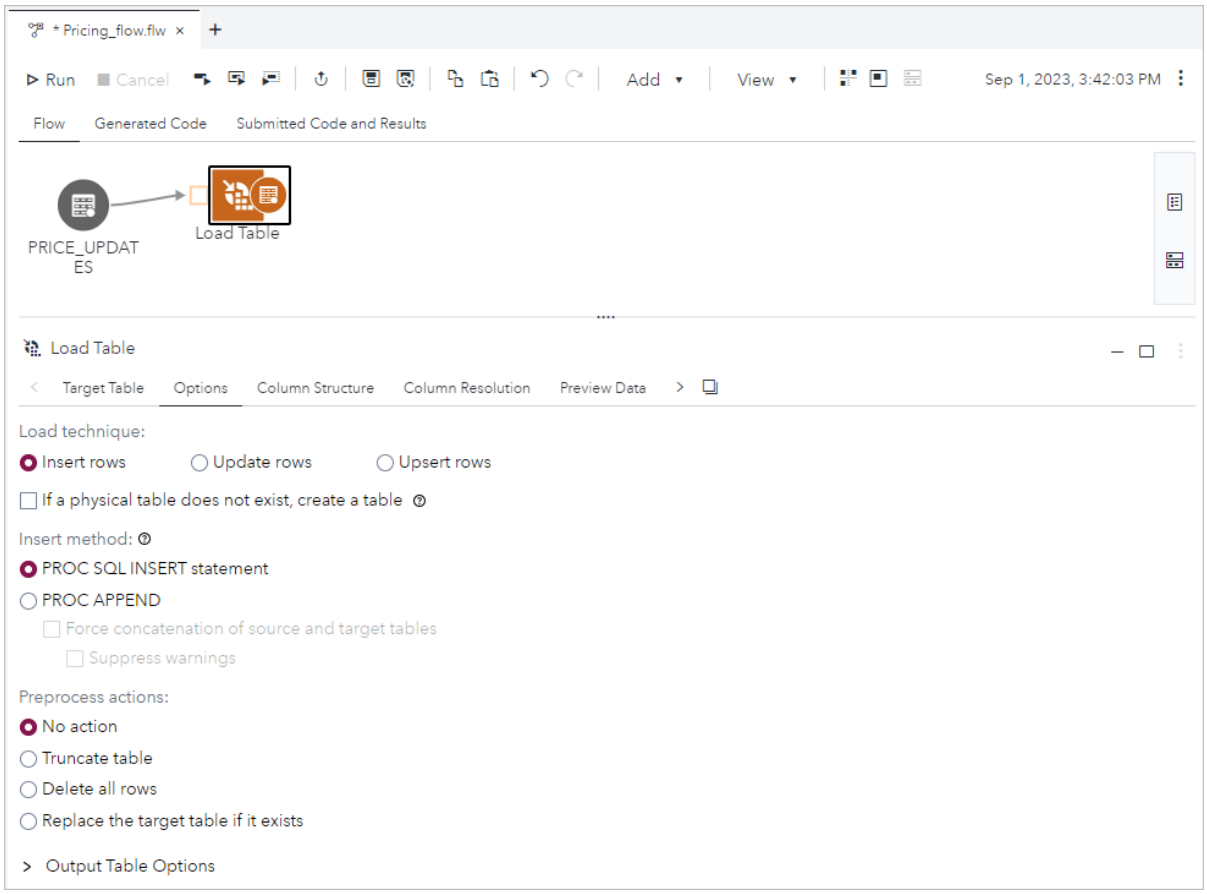
Input Port	Output Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node	Not applicable.

Load Table: Step-by-Step Instructions

Step 1: Add the Load Table Step and Connect a SourceTable

To add the Load Table step to your flow:

1. In the **Steps** section of the navigation pane, expand the **Integrate** folder, and double-click **Load Table**.
2. Connect the input port of the Load Table node to a data source by using a Table node or another node that creates an output table, such a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.



Step 2: Specify the Target Table

The Load Table step can run only if the metadata that defines the column structure of the target table exists.

To specify the target table:

1 Click the **Load Table** node, and then click the **Target Table** tab.

2 Use the **Library** and **Table name** boxes to specify the target table.

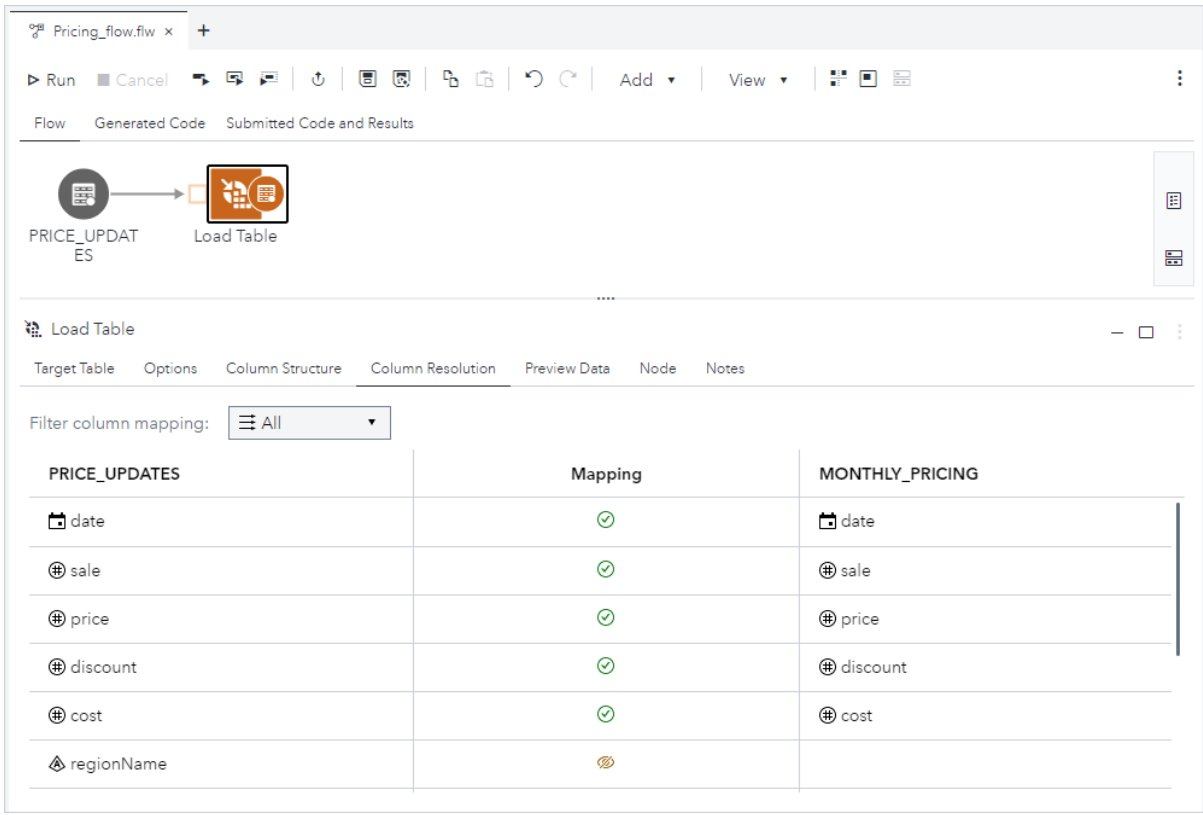
Note: If no metadata is found for the target table that you selected, you must either select a new target table or use the Column Structure tab to specify the column structure. You can copy the column structure from the source table or load the structure from a structure definition file. For more information, see "Specifying the Column Structure for the Target Table" on page 194.

3 To view the structure of the target table, click the **Column Structure** tab.

To view the column mapping between the source and target tables, click the **Column Resolution** tab.

Note: On the Column Resolution tab, you can use the **Filter column mapping** drop-down list to filter the mappings that are displayed. You can filter the mappings by the following categories:

- Successful (✓) - the source column is successfully mapped to a column in the target table.
- Ignored (👁) - the source column does not have a corresponding column in the target table. Data for this column is ignored in the target table.
- Information (ⓘ) - a source column is not mapped to the column in the target table. An unresolved column can occur when SAS Studio cannot automatically map columns based on name and data type.



Step 3: Select the Load Technique

- Load Data Using the Insert Rows Technique

  1. Click the **Options** tab and select **Insert rows**.
  2. If the physical target table does not exist, you can automatically create the table by selecting **If a physical table does not exist, create a table**. If no metadata is found for the target table that you specified, you must either select a new target table or use the Column Structure tab to specify the column structure. You can copy the column structure from the source table or load the structure from a structure definition file. For more information, see "Specifying the Column Structure for the Target Table" on page 194.

If the physical target table does not exist and you do not select this option, the step cannot run successfully.

3 Select the insert method that you want to use:

- PROC SQL INSERT statement - uses the PROC SQL INSERT statement to concatenate the source and target tables.
- PROC APPEND - uses PROC APPEND to concatenate the source and target tables. If the target table is a SAS table type, such as a Base SAS or SPD Server table, the PROC APPEND option can improve performance.

  Select **Force concatenation of source and target tables** to force concatenation in these cases:

  - Source table columns are not in the target table.
  - Source table column data types do not match target table column types.
  - Source table column lengths are longer than target table column lengths.

  If you do not want to write a warning to the log, select **Suppress warnings**.

4 Select the preprocess action that you want to use:

- **No action** - no preprocess action is applied. This option is selected by default.
- **Truncate table** - removes all rows from the table before the source data is loaded, but maintains the table structure, metadata, constraints, and indexes. Foreign key constraints are not maintained. This option uses the database TRUNCATE statement.

  Note: The **Truncate table** option is available only for SAS data sets and databases that support the TRUNCATE statement, such as Oracle, Snowflake, Singlestore, and Azure Synapse.

- **Delete all rows** - deletes all rows from the target table before the source data is loaded. If the target table is an Oracle, Teradata, Snowflake, Singlestore or Azure Synapse table, this option uses the DELETE statement that is associated with the data source. For all other data types, this option uses the PROC SQL DELETE statement.

  Note: When the **Delete all rows** action is used repeatedly to clear a SAS table, the size of that table should be monitored over time. **Delete all rows** performs only logical deletes for SAS tables. Therefore, the table's physical size continues to increase, which can negatively affect performance.

- **Replace the target table if it exists** - replaces the entire table with an empty table before the source data is loaded. This option does not preserve the existing table structure, metadata, constraints, and indexes.

- Load Data Using the Update Rows Technique

  You must specify at least one key column in order to use the Update rows technique.

  1 Click the **Options** tab and select **Update Rows**.

2 Click + and select one or more key columns. You must select at least one key column that uniquely identifies each row in the source table. Do not select all of the columns in the list. Click **OK**.

Note: For some databases, strict matching rules apply to key columns. Depending on the data type of the target table, you might not be able to select a column as a key column if the case of the source and target columns does not match. The Select Key Columns window indicates whether the case of the source and target columns matches. Use the Column Resolution tab to view the source and target column names.

Note: If you are using the Update rows technique to load data into a Snowflake or Singlestore table, a primary key must be defined for the target table. For more information, see "Understanding Snowflake Update and Delete Rules" in *SAS/ACCESS for Relational Databases: Reference*.

- Load Data Using the Upsert Rows Technique

  You must specify at least one key column in order to use the Upsert rows technique.

  1 Click the **Options** tab and select **Upsert Rows**.

  2 If the physical target table does not exist, you can automatically create the table by selecting **If a physical table does not exist, create a table**. If no metadata is found for the target table that you specified, you must either select a new target table or use the Column Structure tab to specify the column structure. You can copy the column structure from the source table or load the structure from a structure definition file. For more information, see "Specifying the Column Structure for the Target Table" on page 194.

  If the physical target table does not exist and you do not select this option, the step cannot run successfully.

  3 Click + and select one or more key columns. You must select at least one key column that uniquely identifies each row in the source table. Do not select all of the columns in the list. Click **OK**.

  For some databases, strict matching rules apply to key columns. Depending on the data type of the target table, you might not be able to select a column as a key column if the case of the source and target columns does not match. The Select Key Columns window indicates whether the case of the source and target columns matches. Use the Column Resolution tab to view the source and target column names.

  Note: If you are using the Upsert rows technique to load data into a Snowflake or Singlestore table, a primary key must be defined for the target table. For more information, see "Understanding Snowflake Update and Delete Rules" in *SAS/ACCESS for Relational Databases: Reference*.

  4 Select the insert method that you want to use:

  - PROC SQL INSERT statement - uses the PROC SQL INSERT statement to concatenate the source and target tables.

- PROC APPEND - uses PROC APPEND to concatenate the source and target tables. If the target table is a SAS table type, such as a Base SAS or SPD Server table, the PROC APPEND option can improve performance.

  Select **Force concatenation of source and target tables** to force concatenation in these cases:

  - Source table columns are not in the target table.
  - Source table column data types do not match target table column types.
  - Source table column lengths are longer than target table column lengths.

  If you do not want to write a warning to the log, select **Suppress warnings**.

(Optional) Step 4: Specify Output Table Options

You can specify options to optimize how the data is loaded into the target table.

1. Click the **Options** tab and expand **Output Table Options**.
2. If you are loading data into an Oracle, Snowflake, Singlestore, Azure Synapse, or Teradata target table, you can select one of the following options:
   - **Use default table options** - uses the default SAS/ACCESS table options that are associated with the data type of the target table.
   - **Bulk load** - loads the rows of data to the target table using the native bulk load facility that is associated with the data type of the target table.

     Note: If you are loading data into an Azure Synapse table, authentication to Azure Data Lake Storage is required. Single sign-on (SSO) is preferred. For more information, see "Authentication for Bulk Loading to Azure" in *SAS/ACCESS for Relational Databases: Reference*. If you need to specify any of the required authentication options as data set options instead of LIBNAME options, click **Advanced table options** and enter each option on a separate line.

   - **Commit all rows in a single transaction** - changes to the target table are committed after all the rows have been written. This option uses the DBCOMMIT = 0 data set option for Oracle, Teradata, Snowflake, Singlestore, and Azure Synapse target tables.
   - **Commit every *n* rows** - changes to the target table are committed after the specified number of rows is written. The default value is 100,000 rows. This option uses the DBCOMMIT = n data set option for Oracle, Teradata, Snowflake, Singlestore, and Azure Synapse target tables.
3. If necessary, you can specify advanced output table options by clicking **Advanced table options**. Enter each option on a separate line. Advanced table options can include any DATA step options that are associated with inserting and updating data in a table, such as options to specify the number of rows to process prior to committing the data or to specify whether to use a DBMS-specific bulk-load facility. You must include any quotation marks that are required for string values.

Step 5: Run the Flow and Load the Data

To load the source data into the target table, click ▷ Run.

Specifying the Column Structure for the Target Table

If no metadata is found for the target table that you have specified, you can use the Column Structure tab in the node details to specify the column structure. You can copy the column structure from the source table or load the structure from a structure definition file. You can also use the Column Structure tab to create a structure definition file that you can use to define the metadata for other target tables.

To copy the structure from the source table:

- Click the **Column Structure** tab and select **Structure** ⇨ **Copy structure from source**.

To load the column structure from an external file:

1. Click the **Column Structure** tab.
2. Select **Structure** ⇨ **Load structure definition**, and then select the metadata definition file that you want to use.

To create a structure definition file that is based on the current structure of the target table:

1. Click the **Column Structure** tab and select **Structure** ⇨ **Generate structure definition**.
2. In the Generate a Column Structure File window, specify the location and name of the file. You can save the file as a comma-delimited (*.csv) or text (*.txt) file. The column structure metadata file is always saved with UTF-8 encoding.

Merge Table

About the Merge Table Step

You can use the Merge Table step to make changes to columns in a target table based on values from a source table. Rows in the source and target columns are

matched using one or more key columns. The Merge Table step enables you to combine update and insert operations in one step.

The Merge Table step works with DBMS tables that support ANSI SQL standards, including Oracle, Teradata, Snowflake, and SQL Server. Both the source and target tables must reside on the same database server.

Note: This step is available only if your site licenses SAS Studio Engineer.

There are six main steps to use the Merge Table step:

1. Add the Merge Table step to your flow and connect the source table.
2. Specify the library and name of the target table.
3. Specify one or more key columns that are used to match the rows between the source and target tables.
4. (Optional) Specify the columns in the target table that should be updated with values from the source table.
5. (Optional) Specify the columns in the target table into which new values should be inserted when no matching row is found.
6. Run the flow to merge the tables.

Note: To run the flow, you must specify at least one column to update or insert new values.

Node Connection Requirements

The Merge Table step includes only one input port. The target table is specified in the node details rather than in an output port. In order to run the Merge Table step, you must create a connection to the input port of the node as indicated here:

Input Port	Output Port
Table node. The table must reside in a database that supports ANSI SQL, including Oracle, Teradata, Snowflake, and SQL Server.	Not applicable.

Merge Table: Step-by-Step Instructions

Step 1: Add the Merge Table Step and Connect a Source Table

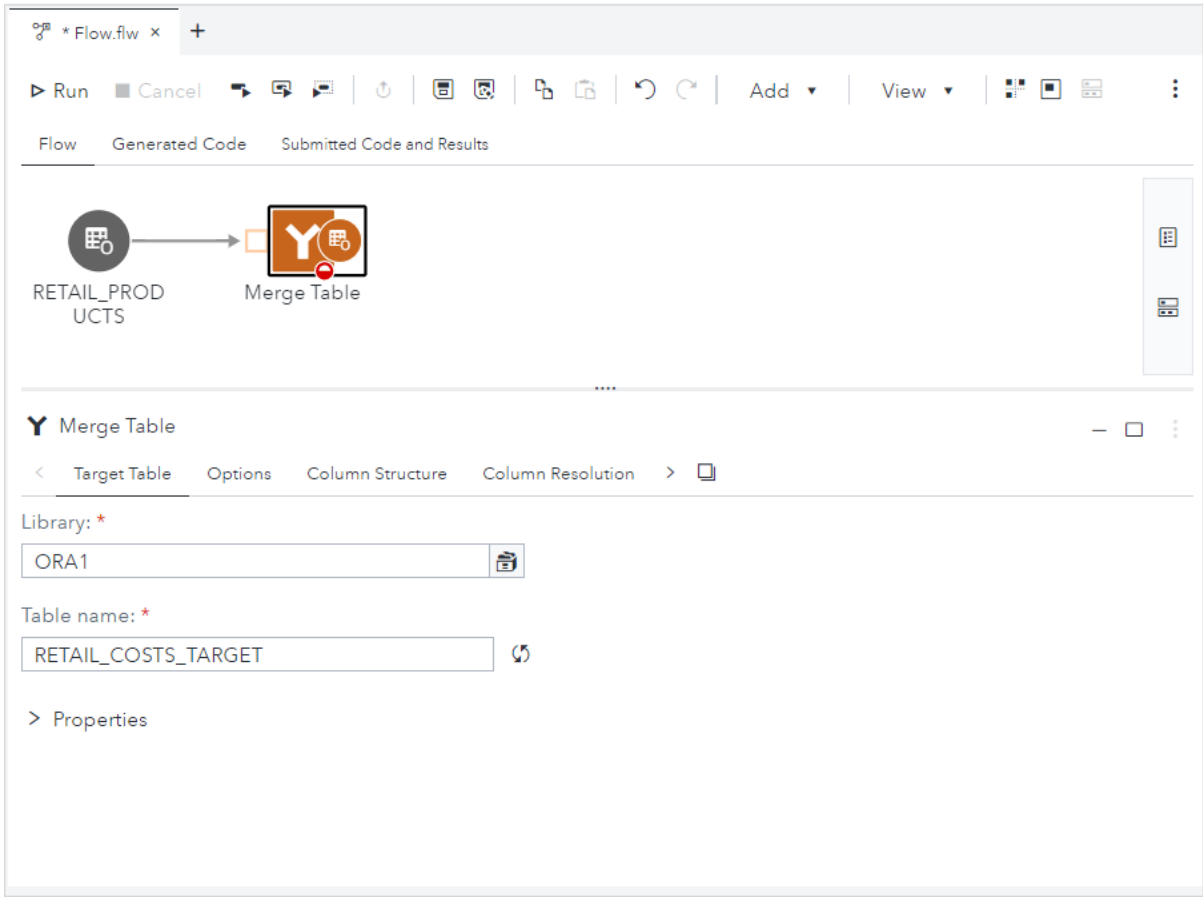
1. In the **Steps** section of the navigation pane, expand the **Integrate** folder, and double-click **Merge Table**.
2. Connect the input port of the Merge Table node to a data source by using a Table node. The table must reside in a database that supports ANSI SQL, such as Oracle, Teradata, Snowflake, and SQL Server. For more information, see “Connecting Nodes” on page 12.

Step 2: Specify the Target Table

The source and target tables must reside on the same database server.

1. Click the **Merge Table** node, and then click the **Target Table** tab.
2. Use the **Library** and **Table name** boxes to specify the target table.

Note: The Merge Table node does not have an output port. You must specify the target table in the Merge Table node details. However, you can connect the Merge Table node to another operational node that requires a table as input.



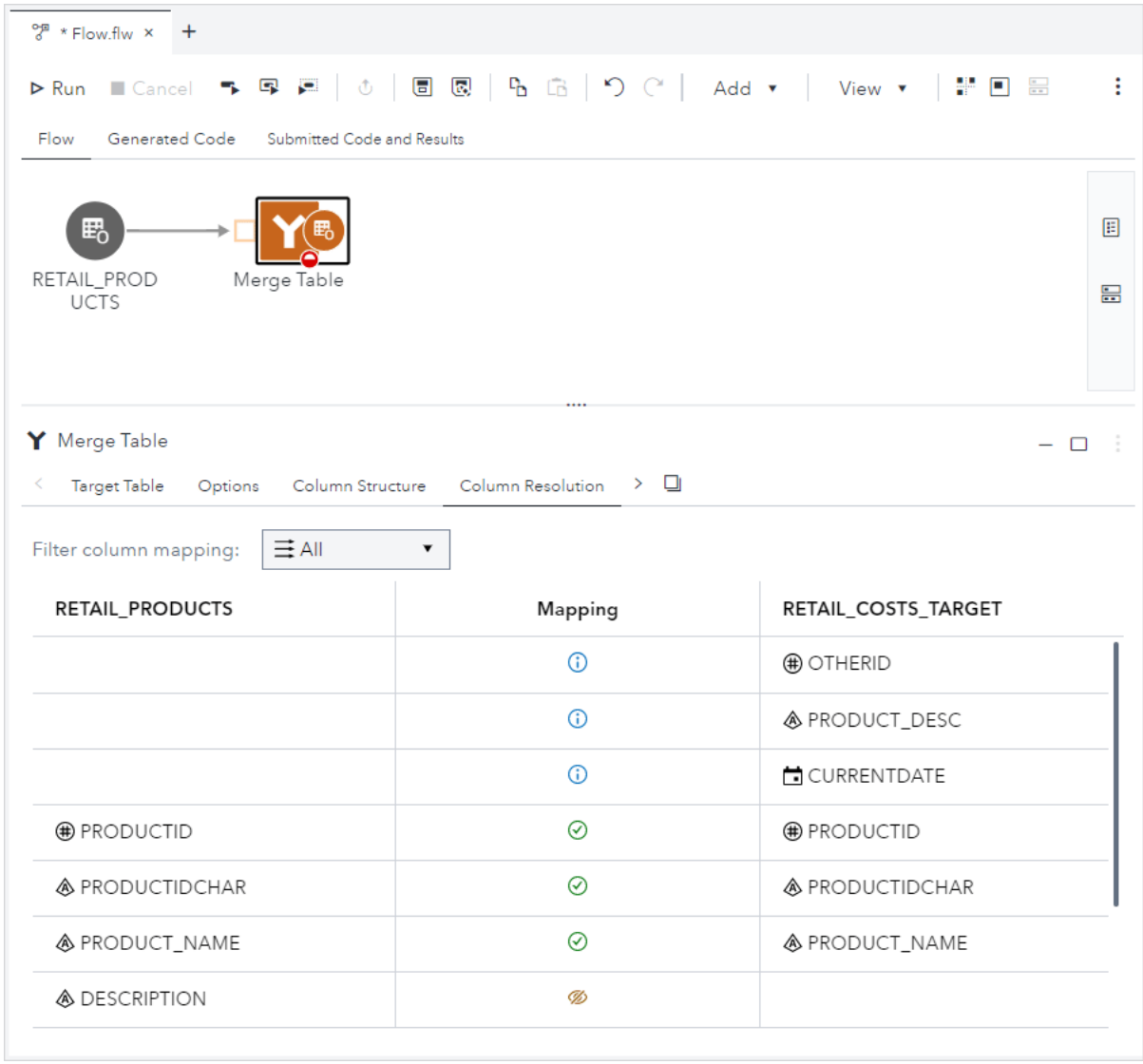
3 To view the structure of the target table, click the **Column Structure** tab.

To view the column mapping between the source and target tables, click the **Column Resolution** tab.

Note: On the Column Resolution tab, you can use the **Filter column mapping** drop-down list to filter the mappings that are displayed. You can filter the mappings by the following categories:

- Successful (✓) - the source column is successfully mapped to a column in the target table.
- Ignored (⊘) - the source column does not have a corresponding column in the target table. Data for this column is ignored in the target table.
- Information (ⓘ) - a source column is not mapped to the column in the target table. An unresolved column can occur when SAS Studio cannot automatically map columns based on name and data type.

In this example, there are three columns in the target table that do not have corresponding columns in the source table, and one column in the source table that does not have a corresponding column in the target table.



Step 3: Specify One or More Key Columns

You must specify at least one key column to run the Merge Table step.

Note: To use the SQL editor to create SQL code for the Merge Table expression, click **SQL editor**. If you have used the Options tab to specify one or more key columns and at least one other column to update or insert, SAS Studio automatically generates the SQL code for those options. The SQL code is passed to the target database for processing by using explicit pass-through, so you should use the SQL syntax of your data source. The expression must include the code to specify the key columns, as well as any target columns to update and target column values to insert. When you select this option, you cannot revert to using the graphical expression builder.

To preview the merge expression that is generated by SAS Studio, click **Preview expression**. You can also copy the code to use in a SAS program. This option is

available only when you have specified one or more key columns and at least one column to update or insert.

1 Click the **Options** tab and select **Key columns**.

2 To add a key column, click **Add columns**.

3 In the Select a Column window, select one or more key columns. You must select at least one key column that uniquely identifies each row in the source table. Do not select all of the columns in the list. Click **OK**.

4 SAS Studio attempts to match key columns in the source and target tables based on column name, regardless of case and data type. If a match is not found, click to select the column that you want to use from the target table. Click **OK**.

(Optional) Step 4: Specify the Target Columns to Update

You can specify columns in the target table to update with values from the corresponding source columns for each row that is matched by a key column. This step is optional if you specify a column to insert values.

Note: Columns that are used as key columns cannot be updated.

1 Click the **Options** tab and select **Update rows**.

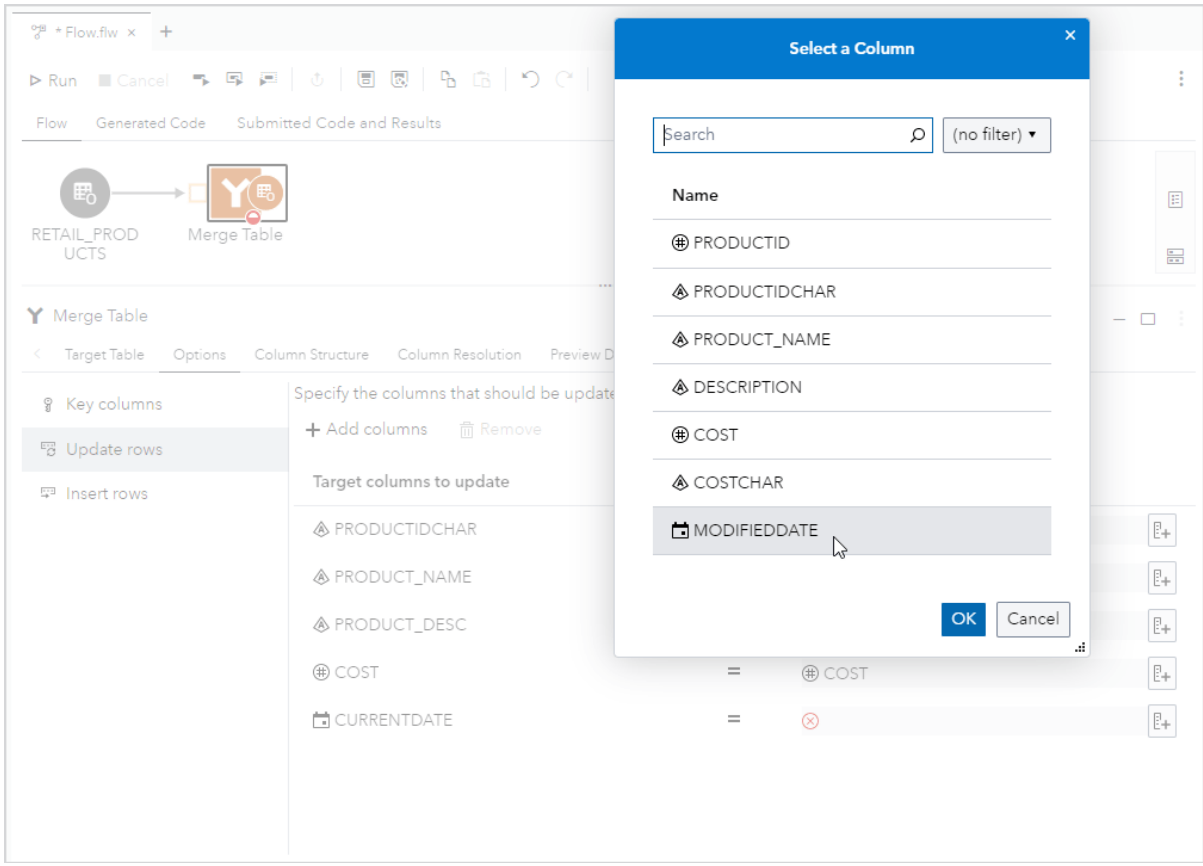
2 To select a column to update, click **Add columns**.

   Note: You can add all of the columns in the target table by clicking **Add all columns**.

3 In the Select a Column window, select one or more columns.

4 SAS Studio attempts to match a column in the source table to the column that you selected from the target table based on column name, regardless of case and data type. If a match is not found, click to select the column that you want to use from the source table. Click **OK**.

   In this example, the MODIFIEDDATE column is selected to map to the CURRENTDATE column.



(Optional) Step 5: Specify Target Column Values to Insert

Specify the columns where new values should be inserted when no matching row is found. This step is optional if you specify a column to update.

1. Click the **Options** tab and select **Insert rows**.
2. To select a column to insert, click **Add columns**.
3. In the Select a Column window, select one or more columns.
4. SAS Studio attempts to match a column in the source table to the column that you selected from the target table based on column name, regardless of case and data type. If a match is not found, click to select the column that you want to use from the source table. Click **OK**.

Step 6: Run the Flow and Merge the Tables

To merge the source and target tables, click ▷ Run.

8

Generating Statistics

Correlation Analysis

About the Correlation Analysis Step

Correlation is a statistical procedure for describing the relationship between numeric variables. The relationship is described by calculating correlation coefficients for the variables. The correlations range from –1 to 1. The Correlation Analysis step provides graphs and statistics for investigating associations among variables.

You might want to use the Correlations Analysis step to learn more about product sales. For example, you can use this step to determine whether there is a correlation between a customer's size (as measured by sales) and the amount of product that the customer purchases from your company.

Note: The Correlation Analysis step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Node Connection Requirements

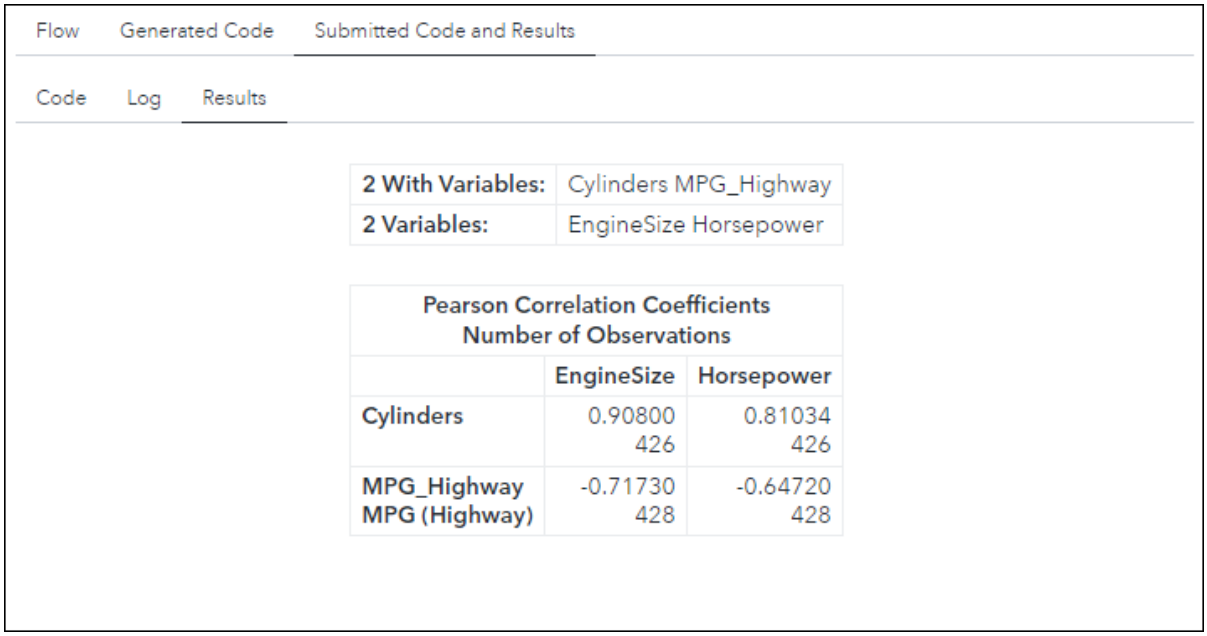
In order to run the Correlation Analysis step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table such as a Query node, a SAS Program node, or an Import node	By default, no output data is created. No connection is required. To create an output data table, you must select the **Create output data** check box. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and the name of the output table by connecting the output port to a Table node. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Example: Correlations in Sashelp.Cars

1. In the **Steps** section of the navigation pane, expand the **Statistics** folder and double-click **Correlation Analysis** to add the step to your flow.
2. Add the Sashelp.Cars table to your flow. Connect the input port of the Correlation Analysis node to the data source. For more information, see "Connecting Nodes" on page 12.
3. To set the options for the Correlation Analysis step, click the Correlation Analysis node.
4. On the **Data** tab, assign columns to these rows:
   - To the **Analysis variables** role, assign **EngineSize** and **Horsepower**.
   - To the **Correlate with** role, assign **Cylinders** and **MPG_Highway** roles.
5. Run the flow.

Open the **Submitted Code and Results** tab to view the results. Here are the results:



2 With Variables:	Cylinders MPG_Highway
2 Variables:	EngineSize Horsepower

Pearson Correlation Coefficients Number of Observations		
	EngineSize	**Horsepower**
Cylinders	0.90800 426	0.81034 426
MPG_Highway **MPG (Highway)**	-0.71730 428	-0.64720 428

Correlation Analysis: Step-by-Step Instructions

Step 1: Assign Data to Roles

To run this step, you must perform one of these tasks:

- assign two or more Analysis variables
- assign at least one Analysis variable and one Correlate with variable

1. (Optional) To filter the input data source, enter the filter expression in the **Filter** field.
2. To the **Analysis variables** role, assign the variables for which to compute correlation coefficients.
3. To the **Correlate with** role, assign the variables with which the correlations of the analysis variables are to be computed.
4. (Optional) To remove the correlation of variables from the analysis, assign those variables to the **Partial variables** role.
5. (Optional) To specify the numeric variable whose value represents the frequency of the observation, assign a variable to the **Frequency count** role. If you assign a variable to this role, the task assumes that each observation represents *n* observations, where *n* is the value of the frequency variable. If *n* is not an integer, SAS truncates it. If *n* is less than 1 or is missing, the observation is excluded from the analysis. The sum of the frequency variable represents the total number of observations.
6. To the **Weight** role, assign the variables that contain the weights to use in the calculation of the weighted product-moment correlation.
7. (Optional) To obtain separate analyses of observations for each unique group, assign a variable to the **Group analysis by** role.

Step 2: Select the Options

On the **Options** tab, you can set these options:

1. Select **Missing values** to specify how to treat observations with missing values.
   - If you select **Use nonmissing values for all selected variables**, all observations with missing values are excluded from the analysis.
   - If you select **Use nonmissing values for pairs of variables**, the correlation statistics are computed using the nonmissing pairs of variables. This is the default.

2 Select the statistics to display in the results. By default, only the correlations are displayed.

- **Correlations** includes the correlations in the results. You can also specify probabilities that are associated with each correlation coefficient and whether to order the correlations from highest to lowest in absolute value.
- **Display p-values** includes the *p*-values in the results.
- **Order correlations from highest to lowest (in absolute value)** ranks the correlation coefficients from highest to lowest. For each analysis variable, the **Correlate with** variables are listed in order of descending absolute value of correlation coefficient. By default, the correlation coefficients are not ranked.
- **Covariances** includes the variance and covariance matrix in the results. Also, the Pearson correlations are displayed. If you assign a column to the **Partial variables** role, the task computes a partial covariance matrix.
- **Sum of squares and cross-products** displays the sum of squares and cross-products in the results. If you assign a column to the **Partial variables** roles, then the unpartial sum of squares and cross-products matrix is computed.
- **Corrected sum of squares and cross-products** displays a table of the corrected sums of squares and cross-products. The Pearson correlations are also included in the results. If you assign a column to the **Partial variables** role, the task computes both an unpartial and a partial corrected sum of squares and cross-products matrix.
- **Descriptive statistics** displays the number of observations, mean, standard deviation, minimum value, maximum value, and label for each variable in the analysis.
- **Fisher's z transformation** works with a Pearson correlation. For a Pearson correlation, you can use the Fisher transformation options to request confidence limits and *p*-values under a specified alternative (null) hypothesis, $H_0: \rho = \rho_0$, for correlation coefficients that use Fisher's z transformation. If you select the **Fisher's z transformation** check box, you must specify a value in the **Null hypothesis** box.
  - **Two-sided confidence limits** requests two-sided confidence limits for the test of the null hypothesis, $H_0: \rho = \rho_0$. This is the default.
  - **Lower confidence limit** requests a lower confidence limit for the test of the one-sided null hypothesis, $H_0: \rho \leq \rho_0$.
  - **Upper confidence limit** requests an upper confidence limit for the test of the one-sided null hypothesis, $H_0: \rho \geq \rho_0$.
- You can also specify these nonparametric correlations:
  - **Spearman's rank-order correlation** calculates Spearman rank-order correlation. This is a nonparametric measure of association that is based on the rank of the data values. The correlations range from –1 to 1.
  - **Kendall's tau-b** calculates Kendall tau-b. This is a nonparametric measure of association that is based on the number of concordances and discordances in paired observations. Concordance occurs when paired observations vary together, and discordance occurs when paired observations vary differently. Kendall's tau-b ranges from –1 to 1.
  - **Hoeffding's measure of dependence** calculates Hoeffding's measure of dependence, D. This is a nonparametric measure of association that

detects more general departures from independence. This D statistic is 30 times larger than the usual definition and scales the range between –0.5 and 1 so that only large positive values indicate dependence.

Note: This option is not available if you assign a variable to the **Partial variables** role.

Note: The Nonparametric Correlations options are not available if you assign a variable to the **Weight variable** role.

3 Specify whether to include a plot in the results. By default, no plot is included.

- **Matrix of scatter plots** includes a matrix of scatter plots in the results. You can also choose to include a histogram of the analysis variables in the symmetric matrix plot.
- **Individual scatter plots** includes a scatter plot for each applicable pair of distinct variables from the analysis variables. You can specify whether to display the prediction ellipses for new observations or the confidence ellipses for the mean.

  You can also specify the number of variables to plot and the maximum number of points to plot.

Step 3: Specify the Output Options

On the **Output** tab, you can specify these options:

1 Select **Create output data** to save the results to an output table. By default, SAS Studio overwrites any existing output tables.

2 Select the statistics to show in the results:

- **Correlations** - By default, the output table contains the correlation coefficients with the corresponding _TYPE_ variable value of 'CORR'.
- **Covariances** - When you select this option, the output table contains the covariance matrix with the corresponding _TYPE_ variable value of ‘COV’.
- **Sum of squares and cross-products** - If you assign a column to the **Partial variables** role, the output table does not contain a sum of squares and cross-products matrix.
- **Corrected sum of squares and cross-products** - If you assign a column to the **Partial variables** role, the output table contains a partial corrected sum of squares and cross-products matrix.

Distribution Analysis

About the Distribution Analysis Step

Distribution analysis provides information about the distribution of numeric variables. A variety of plots such as histograms, probability plots, and quantile-quantile plots can be used in this analysis.

Note: The Distribution Analysis step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Node Connection Requirements

In order to run the Distribution Analysis step, you must create connections to the input and output ports of the node as indicated here:

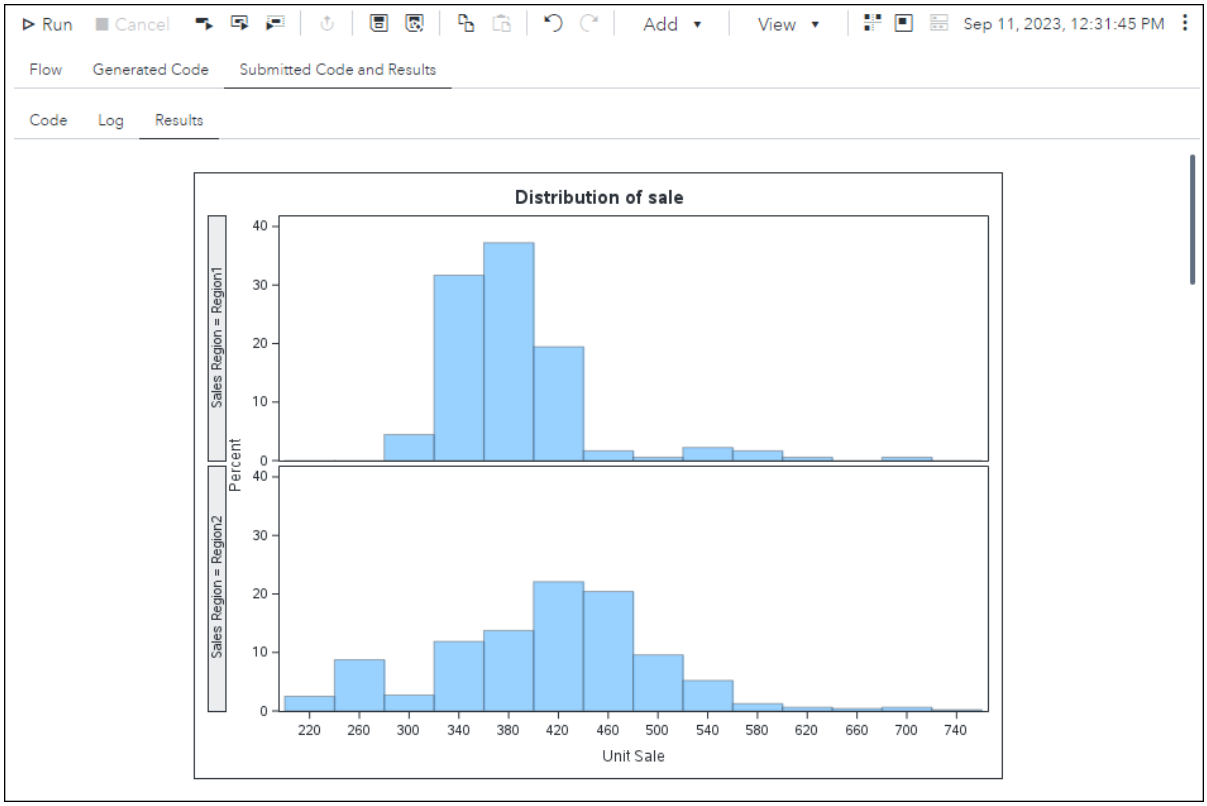
Input Port	Output Port
■ Table node or ■ Operational node that creates an output table such as a Query node, a SAS Program node, or an Import node	By default, no output data is created. No connection is required. To create an output data table, you must select the **Create output data** check box. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and the name of the output tables by connecting the output port to a Table node. For more information, see "Adding a Table from a SAS Library to a Flow" on page 36.

Example: Distribution Analysis of Sales for Each Region

In this example, you want to analyze the sales for each region. Because the data contains three regions, you get three sets of results.

1. In the **Steps** section of the navigation pane, expand the **Statistics** folder and double-click **Distribution Analysis** to add the step to your flow.
2. Add the Sashelp.PriceData table to your flow. Connect the input port of the Distribution Analysis node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the Distribution Analysis step, click the Distribution Analysis node.
4. On the **Data** tab, assign **sale** to the **Analysis variables** role.
5. On the **Options** tab, set these options:
   - In the Exploring Data group, assign the regionName variable to the **Classification variables** role.
   - In the Checking for Normality group, select the **Histogram and goodness-of-fit tests** and **Normal quantile-quantile plot** options.
6. Run the flow.

Open the **Submitted Code and Results** tab to view the results. Here is a sample of the results:



Distribution Analysis: Step-by-Step Instructions

Step 1: Assign Data to Roles

1 (Optional) To filter the input data source, enter the filter expression in the **Filter** field.

2 Assign a numeric variable to the **Analysis variables** role. You can specify the order of the analysis variables in the results.

3 (Optional) To specify the numeric variable whose value represents the frequency of the observation, assign a variable to the **Frequency count** role. If you assign a variable to this role, the task assumes that each observation represents *n* observations, where *n* is the value of the frequency variable. If *n* is not an integer, SAS truncates it. If *n* is less than 1 or is missing, the observation is excluded from the analysis. The sum of the frequency variable represents the total number of observations.

4 (Optional) To obtain separate analyses of observations for each unique group, assign a variable to the **Group analysis by** role.

Step 2: Select the Options

On the **Options** tab, you can set these options.

1 Select **Histogram** to create a graph that is used to determine the distribution of the data. A histogram is created by default.

You can also specify these options:

- In the **Classifications variables** role, specify the variables that are used to group the analysis variables into classification levels. You can assign a maximum of two columns to this role.
- If you select **Add normal curve**, the task uses the sample mean and sample standard deviation for μ and σ.
- If you select **Add kernel density estimate**, the task uses the AMISE method to compute the kernel density estimates.
- Select **Add inset statistics** to select where to display the statistics in relation to the histogram.

2 Under the Checking for Normality heading, select these options:

- **Histogram and goodness-of-fit tests** requests tests for normality that include a series of goodness-of-fit tests based on the empirical distribution function. The table provides test statistics and *p*-values for the Shapiro-Wilk test (provided the sample size is less than or equal to 2,000), the

Kolmogorov-Smirnov test, the Anderson-Darling test, and the Cramér-von Mises test.

You can also specify whether to include an inset box of selected statistics in the graph.

- **Normal probability plot** creates a probability plot that compares ordered variable values with the percentiles of the normal distribution. If the data distribution matches the normal distribution, the points on the plot form a linear pattern. Probability plots are preferable for graphical estimation of percentiles.

  The distribution reference line on the plot is created from the maximum likelihood estimate for the parameter.

- **Normal quantile-quantile plot** creates quantile-quantile plots (Q-Q plots) and compares ordered variable values with quantiles of the normal distribution. If the data distribution matches the normal distribution, the points on the plot form a linear pattern. Q-Q plots are preferable for graphical estimation of distribution parameters.

  The distribution reference line on the plot is created from the maximum likelihood estimate for the parameter.

  You can also specify whether to include an inset box of selected statistics in the graph.

3 Under the Fitting Distributions heading, select from these fitting distributions:

Beta
- **Histogram and goodness-of-fit tests** fits beta distribution with a threshold parameter, scale parameter, and shape parameter.
- **Probability Plot** specifies a beta probability plot for the shape parameters.
- **Quantile-quantile plot** specifies a beta Q-Q plot for the shape parameters.

Exponential
- **Histogram and goodness-of-fit tests** fits exponential distribution with a threshold parameter and scale parameter.
- **Probability Plot** specifies an exponential probability plot.
- **Quantile-quantile plot** specifies an exponential Q-Q plot.

Gamma
- **Histogram and goodness-of-fit tests** fits a gamma distribution with a threshold parameter, scale parameter, and shape parameter.
- **Probability Plot** specifies a gamma probability plot for the shape parameter.
- **Quantile-quantile plot** specifies a gamma Q-Q plot for the shape parameter.

Lognormal
- **Histogram and goodness-of-fit tests** fits lognormal distribution with a threshold parameter, scale parameter, and shape parameter.
- **Probability Plot** specifies a lognormal probability plot for the shape parameter.
- **Quantile-quantile plot** specifies a lognormal Q-Q plot for the shape parameter.

Weibull
- **Histogram and goodness-of-fit tests** fits Weibull distribution with a threshold parameter, scale parameter, and shape parameter.

- **Probability Plot** specifies a two-parameter Weibull probability plot.
- **Quantile-quantile plot** specifies a two-parameter Weibull Q-Q plot.

One-Way Frequencies

About the One-Way Frequencies Step

The One-Way Frequencies step generates frequency tables from your data. You can also use this step to perform binomial and chi-square tests.

You might want to use this step to analyze the efficiency of a new drug. For example, suppose a group of medical researchers are interested in evaluating the efficacy of a new treatment for a skin condition. Dermatologists from participating clinics are trained to conduct the study and to evaluate the condition. After the training, two dermatologists examine patients with the skin condition from a pilot study and rate the same patients. The One-Way Frequencies step can be used to evaluate the agreement of the diagnoses.

Note: The Table Analysis step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Node Connection Requirements

In order to run the One-Way Frequencies step, you must create the input port of the node as indicated here:

Input Port
■ Table node or ■ Operational node that creates an output table, such as a Query node, a SAS Program node, or an Import node

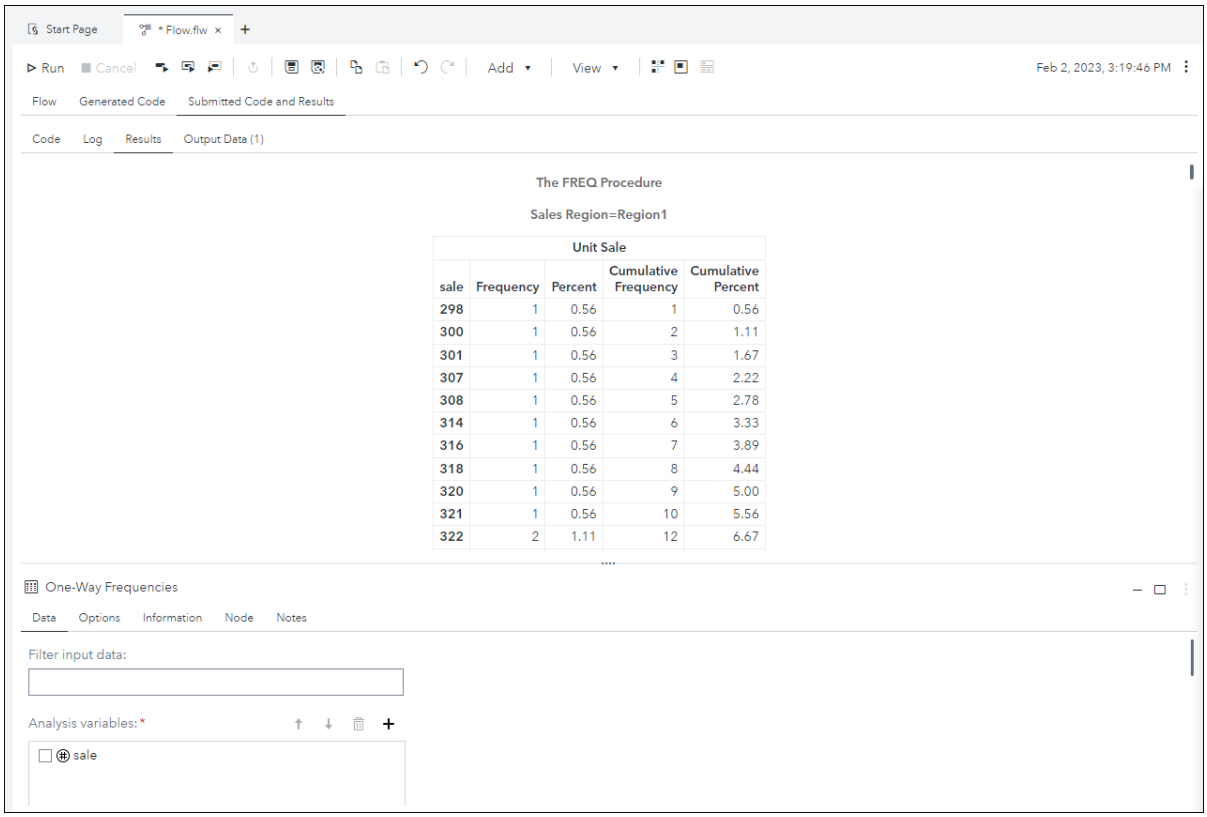
Example: One-Way Frequencies of Unit Sales

In this example, you want to analyze unit sales for each sales region.

To create this example:

1. In the **Steps** section of the navigation pane, expand the **Statistics** folder and double-click **One-Way Frequencies** to add the step to your flow.
2. Add the Sashelp.PriceData table to your flow. Connect the input port of the One-Way Frequencies node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the One-Way Frequencies step, click the One-Way Frequencies node.
4. On the **Data** tab, assign columns to these roles:
   - Assign the **sale** column to the **Analysis variables** role.
   - Expand the **Additional Roles** heading. Assign the **regionName** column to the **Group analysis by** role.
5. Run the flow.

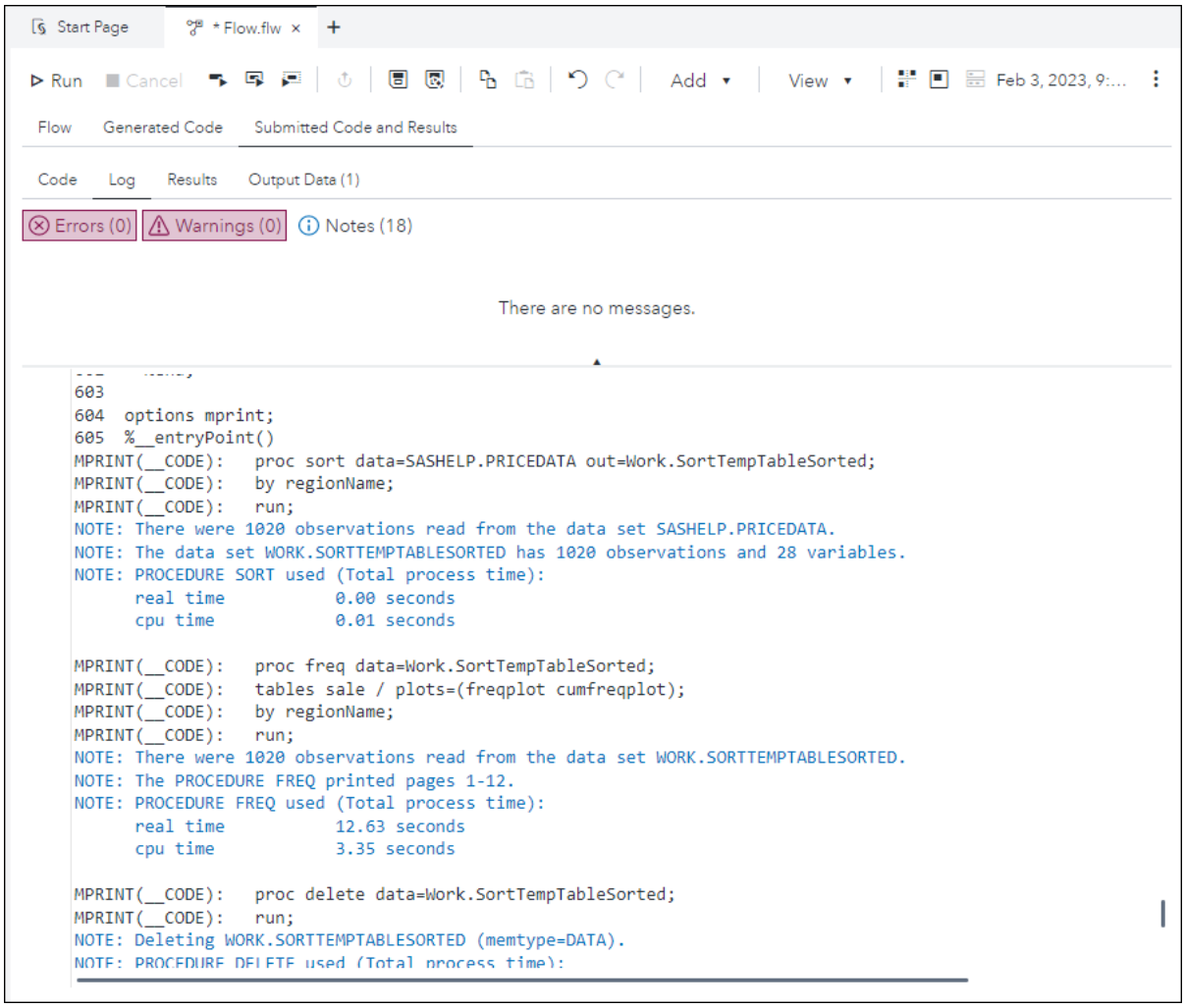
Open the **Submitted Code and Results** tab to view the results.



To view the generated SAS code:

1. Open the **Log** tab.
2. Search for MPRINT_CODE.

In this example, here is the generated SAS code.



One-Way Frequencies: Step-By-Step Instructions

Step 1: Assign Data to Roles

1 (Optional) To filter the input data source, enter the filter expression in the **Filter** field.

2 Assign the variables to be analyzed to the **Analysis variables** role. For each variable that you assign to this role, the task creates a one-way frequency table. You must assign at least one variable to this role.

3 (Optional) Assign a variable to the **Frequency count** role.

If you assign a variable to this role, the task assumes that each observation represents *n* observations, where *n*is the value of the frequency variable. If *n* is not an integer, SAS truncates it. If *n* is less than 1 or is missing, the observation is excluded from the analysis. The sum of the frequency variable represents the total number of observations.

4 (Optional) To obtain separate analyses of observations for each unique group, assign a variable to the **Group analysis by** role.

Step 2: Setting Options

On the **Options** tab, you can set these options:

1 Select the frequencies and percentages to calculate. By default, all of these options are selected.

- **Frequency table** specifies whether to create the frequencies table.
- **Include percentages** creates a table that contains the cumulative frequencies and percentages and the percentages of total frequencies for each value of the analysis variable.
- **Include cumulative frequencies and percentages** creates a table that contains the frequencies and cumulative frequencies for each value of the analysis variable.

2 Specify the order of the rows.

- **Unformatted values** to order the values by their unformatted values. This is the default.
- **Data set order** to order the values by their appearance in the data set.
- **Formatted values** to order the values by their ascending formatted values.
- **Descending frequency** to order the values so that the levels with the highest frequency counts come first.

3 (Optional) Specify whether to calculate asymptotic tests for binomial proportion and chi-square goodness-of-fit tests.

- For binomial proportions, specify a null hypothesis proportion and a confidence level. To compute the exact *p*-values, select **Exact test**.
- For chi-square goodness-of-fit, specify whether to compute exact tests.

  Select **Limit computation time** to specify the time limit (in seconds) for the computation of each *p*-value for each crosstabulation table. The default is 300 seconds (or 5 minutes).

  To compute the Monte Carlo estimates of the exact *p*-values instead of directly computing the exact *p*-values, select **Use Monte Carlo estimation**. Monte Carlo estimation can be useful for large problems that require a great amount of time and memory for exact computations but for which asymptotic approximations might be insufficient.

4 (Optional) Select the options for plots and missing values.

- **Suppress plots** suppresses the plots from the results.
- **Include in frequency table** includes missing values in the frequency tables.
- **Include in percentages and statistics** includes the frequencies of missing values in binomial or chi-square tests and in the calculations of percentages.

Summary Statistics

About the Summary Statistics Step

The Summary Statistics step provides descriptive statistics for variables across all observations and within groups of observations. You can also summarize your data in a graphical display such as histograms and box plots.

For example, you could use this step to create a report on the number of new sales, arranged by product type and country.

Note: The Summary Statistics step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Node Connection Requirements

In order to run the Summary Statistics step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table such as a Query node, a SAS Program node, or an Import node	By default, no output data is created. No connection is required. To create an output data table, you must select the **Create output data** check box. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and the name of the output tables by connecting the output port to a Table node. For more information, see “Adding a Table from a SAS Library to a Flow” on page 36.

Example: Summary Statistics of Unit Sales

1 In the **Steps** section of the navigation pane, expand the **Statistics** folder and double-click **Summary Statistics** to add the step to your flow.

2 Add the Sashelp.PriceData table to your flow. Connect the input port of the Summary Statistics node to the data source. For more information, see “Connecting Nodes” on page 12.

3 To set the options for the Summary Statistics step, click the Summary Statistics node.

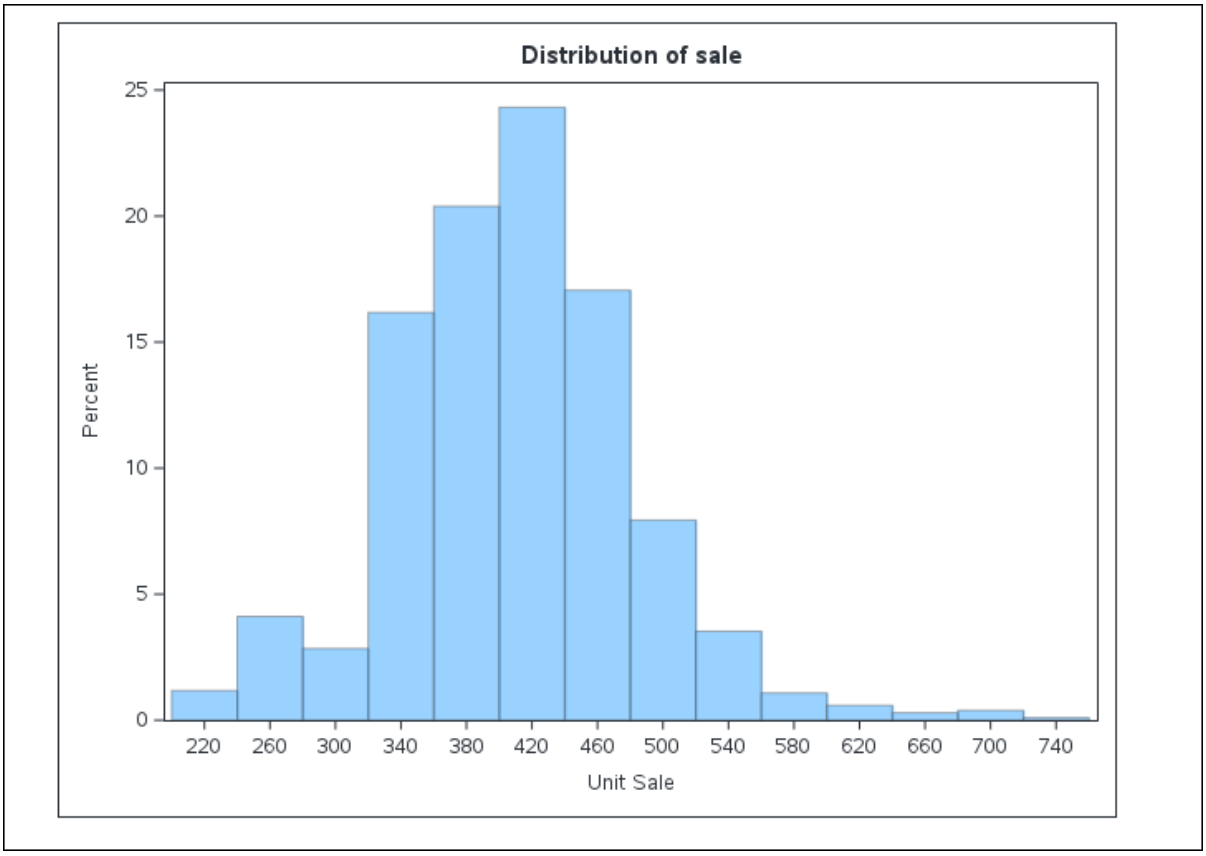
4 On the **Data** tab, assign **sale** to the **Analysis variables** role.

5 On the **Options** tab, expand the **Plots** section and select the **Histogram** check box.

6 Run the flow.

Open the **Submitted Code and Results** tab to view the results. Here is a sample of the results:

Analysis Variable : sale Unit Sale				
Mean	Std Dev	Minimum	Maximum	N
408.5558824	73.0840041	203.0000000	747.0000000	1020



Summary Statistics: Step-by-Step Instructions

Step 1: Assign Data to Roles

1. (Optional) To filter the input data source, enter the filter expression in the **Filter** field.
2. Assign a numeric variable to the **Analysis variables** role.
3. Assign a character or discrete numeric variable to the **Classification variables** role. The statistics are calculated on all selected analysis variables for each unique combination of classification variables.

   Specify the combinations of classification variables.

   - **N-way only** creates a single table that shows the summary statistics for the n-way combination of the classification variables. For example, suppose that you want to compare the sales that result from two different marketing campaigns. You assign the variable Profit to the **Analysis variables** role and the variables Month, Marketing Campaign, and Region to the **Classification variables** role. Then selecting the **N-way only** option results in a table of Month*Campaign*Region.
   - **All ways** creates a separate table of summary statistics for each unique combination of the classification variables. For example, using the variables Month, Campaign, and Region, and then selecting the **All ways** option results in the following tables:
     - a table of overall sales. This is called a zero-way table, and the classification variables are not included.
     - a one-way table that displays the summary statistics for each month (for example, May, June, and July).
     - a one-way table that displays the summary statistics for each region (for example, north, south, east, and west).
     - a one-way table that displays the summary statistics for each type of marketing campaign (for example, TV commercial, magazine ad).
     - a two-way table that displays the summary statistics for Month*Region.
     - a two-way table that displays the summary statistics for Month*Campaign.
     - a two-way table that displays the summary statistics for Region*Campaign.
     - a three-way table that displays the summary statistics for Month*Campaign*Region.
   - **Specify ways** creates a separate table of summary statistics for each unique combination of the classification variables that you specify. For example, if you enter `1`, `2`, or `3` in the **Specify ways** box, the results contain all the possible one-way, two-way, or three-way tables. If you specify `0` in this box, then the results contain a zero-way table, which does not include the classification variables.

4 (Optional) To obtain separate analyses of observations for each unique group, assign a variable to the **Group analysis by** role.

5 (Optional) To specify the numeric variable whose value represents the frequency of the observation, assign a variable to the **Frequency count** role. If you assign a variable to this role, the task assumes that each observation represents *n* observations, where *n* is the value of the frequency variable. If *n* is not an integer, SAS truncates it. If *n* is less than 1 or is missing, the observation is excluded from the analysis. The sum of the frequency variable represents the total number of observations.

6 (Optional) Assign a variable to the **Weight variable** role. If you assign a variable to this role, the value of the variable for each observation is used to calculate weighted means, variances, and sums. You can assign a maximum of one variable to this role.

Step 2: Select the Statistics

On the **Options** tab, you can set these options.

1 Select the basic statistics to include in the analysis.

- **Mean** is the arithmetic average, calculated by adding the values of an analysis variable and dividing this sum by the number of nonmissing observations.
- **Standard deviation** is a statistical measure of the variability of a group of data values. This measure, which is the most widely used measure of the dispersion of a frequency distribution, is equal to the positive square root of the variance.
- **Minimum value** is the smallest value for an analysis variable.
- **Maximum value** is the largest value for an analysis variable.
- **Number of observations** is the total number of observations with nonmissing values.
- **Number of missing values** is the total number of observations with missing values.

2 Specify the maximum number of decimal places for the calculated statistics. By default, a statistic is displayed by using the best fit, which is usually seven decimal places.

3 Specify the divisor to use in the calculation of the variance and standard deviation. By default, the divisor is the degrees of freedom.

4 Select any additional statistics:

- **Standard error** is the standard deviation of the sample mean. The standard error is defined as the ratio of the sample standard deviation to the square root of the sample size.
- **Variance** is a statistical measure of dispersion of data values. This measure is an average of the total squared dispersion between each observation and the sample mean.
- **Mode** is the most frequent value for the analysis variable.

Note: This option is not available if you assigned a weight variable.

- **Range** is the difference between the largest and smallest values in the data.
- **Sum** is the sum of all values in the analysis variable.
- **Sum of weights** is the sum of the numeric variable that is used to weight each observation.
- **Confidence limits for the mean** is the two-sided confidence limits for the mean.
- **t statistic and Prob > |t|** produces *t* statistics for the regression coefficients and probability levels for the *t* statistics.
- **Coefficient of variation** is a unitless measure of relative variability. This measure is defined as the ratio of the standard deviation to the mean expressed as a percentage. The coefficient of variation is meaningful only if the variable is measured on a ratio scale.
- **Uncorrected sum of squares** displays the uncorrected sum of squares.
- **Skewness** measures the tendency of the deviations to be larger in one direction than in the other.

  Note: This option is not available if you assign a weight variable.

- **Kurtosis** measures the heaviness of tails.

  Note: This option is not available if you assign a weight variable.

5 Select the percentiles to compute.

6 Select whether to include a histogram or box plot in the results.

Note: Weight variables are ignored when calculating plots.

- **Histogram** creates a graph that is used to determine the distribution of the data. You can also specify these options:
  - If you select **Add normal density curve**, the task uses the sample mean and sample standard deviation for μ and σ.
  - If you select **Add kernel density estimate**, the task uses the AMISE method to compute the kernel density estimates.
  - Select **Add inset statistics** to select where to display the statistics in relation to the histogram.
  - If you assigned only one analysis variable, select **Use midpoints** to specify the starting and ending midpoints. You can also specify the step interval to use to calculate the other midpoints. The midpoint is the middle value of the range of values represented by each bar.
- **Box plot** creates a box plot that shows a measure of central location (the median), two measures of dispersion (the range and interquartile range), the skewness (from the orientation of the median relative to the quartiles), and potential outliers. Box plots are especially useful in comparing two or more sets of data.

Step 3: Specify the Titles and Footnotes for the Results

On the **Titles** tab, you specify whether to use the default text for the title and the footnote. Alternatively, you can customize this text. You can use macro variables in titles and footnotes.

Step 4: Specify the Output Options

On the **Output** tab, you can specify these options.

1. Select **Create output data** to save the results to an output table. By default, SAS Studio overwrites any existing output tables.
2. Select **Show statistics** to show the statistics for the combinations of the classification variables in the results.
3. Select **Show Analysis labels** to display the labels for the analysis variables.

Table Analysis

About the Table Analysis Step

The Table Analysis step provides one-way to n-way frequency and contingency (crosstabulation) tables. This step also generates statistics about the association between rows and columns.

Note: The Table Analysis step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Node Connection Requirements

In order to run the Table Analysis step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node or ■ Operational node that creates an output table such as a Query node, a SAS Program node, or an Import node	No connection is required. **Note:** By default, the output data is written to a temporary table in the Work library. You can specify the library and the name of the output tables by connecting the output port to a Table node. For more information, see "Adding a Table from a SAS Library to a Flow" on page 36.

Example: Distribution of Type by DriveTrain

1. In the **Steps** section of the navigation pane, expand the **Statistics** folder and double-click **Table Analysis** to add the step to your flow.
2. Add the Sashelp.Cars table to your flow. Connect the input port of the Table Analysis node to the data source. For more information, see "Connecting Nodes" on page 12.
3. To set the options for the Table Analysis step, click the Table Analysis node.
4. On the **Data** tab, assign columns to these roles:
   - For the **Row variables** role, assign **Type**.
   - For the **Column variables** role, assign **DriveTrain**.
5. Run the flow.

Open the **Submitted Code and Results** tab to view the results. Here is a sample of the results:

Frequency

Table of Type by DriveTrain

	DriveTrain			
Type	All	Front	Rear	Total
Hybrid	0	3	0	3
SUV	38	22	0	60
Sedan	28	179	55	262
Sports	5	8	36	49
Truck	12	0	12	24
Wagon	9	14	7	30
Total	92	226	110	428

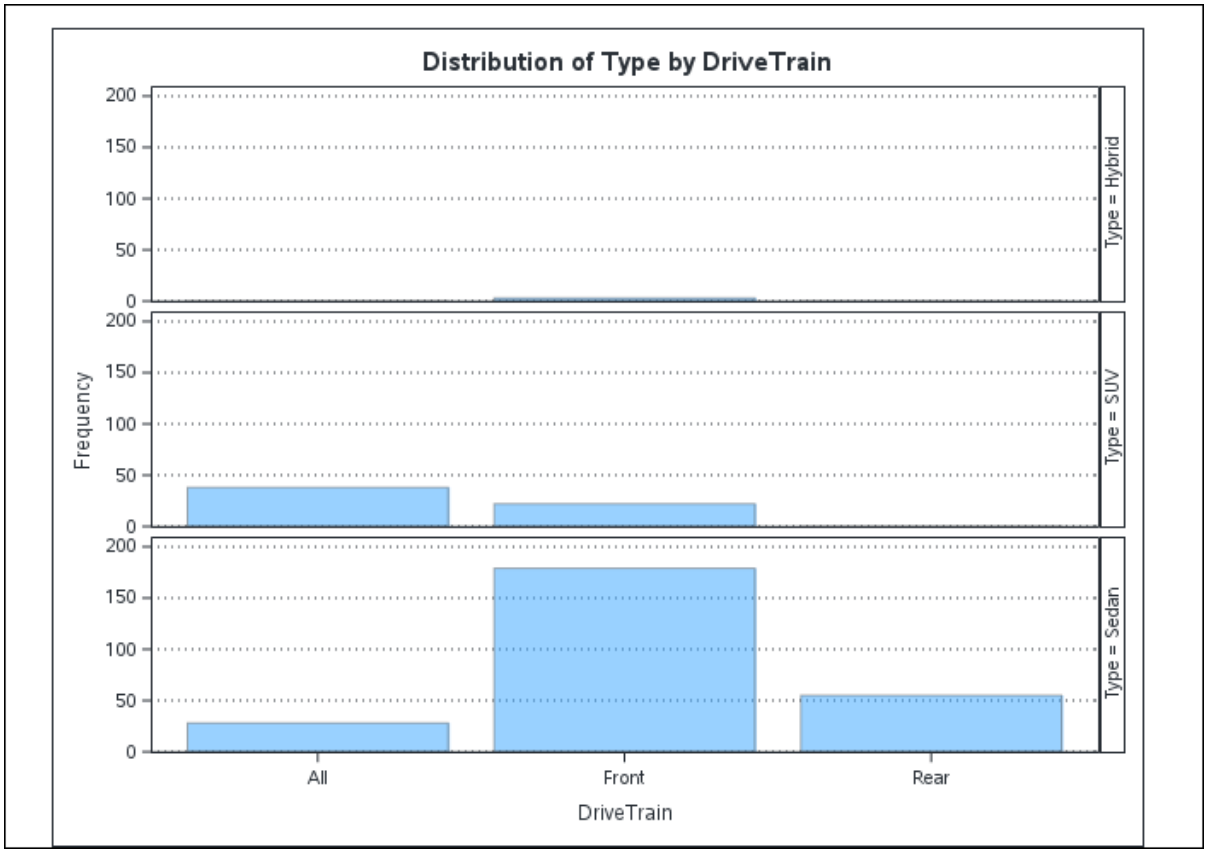


Table Analysis: Step-by-Step Instructions

Step 1: Assign Data to Roles

1. (Optional) To filter the input data source, enter the filter expression in the **Filter** field.
2. Assign the row for one-way table analysis to the **Row variables** role. If multiple variables are assigned to this role, the task performs multiple one-way table analyses.
3. Assign a column to the **Column variables** role. This role creates the columns for one-way table analysis. If only column variables are assigned, the task performs multiple one-way table analyses.
4. In the **Strata variables** role, assign a variable to create separate tables for n-way frequency and n-way frequency and crosstabulation tables.

   Note: You must assign columns to both the **Row variables** and **Column variables** roles to use a strata variable.

5. (Optional) Assign a variable to the **Frequency count** role.

If you assign a variable to this role, the task assumes that each observation represents *n* observations, where *n* is the value of the frequency variable. If *n* is not an integer, SAS truncates it. If *n* is less than 1 or is missing, the observation is excluded from the analysis. The sum of the frequency variable represents the total number of observations.

Step 2: Select the Cell Statistics

On the **Cell Statistics** tab, you can set these options.

1 Select the frequency statistics.

Observed
displays the frequency count for each cell.

Expected
displays the expected cell frequency for each cell.

Deviation
displays the deviation of the cell frequency from the expected value for each cell.

Missing values
specifies whether to display the frequencies for the missing values.

2 Select the percentages.

Cell
displays overall percentages in crosstabulation tables.

Row
displays row percentages in crosstabulation table cells.

Column
displays column percentages in crosstabulation table cells.

You can also choose from these options:

- **Include percentages in the data set** displays the percentages for the columns and rows in the output data set.
- **Percentages of total frequency** displays the percentage of total frequency on *n*-way tables when *n*>2.

3 Select the cumulative statistics.

Column percentages
displays the cumulative column percentages in each cell.

Frequencies and percentages
displays the cumulative frequencies and percentages in one-way frequency tables.

4 Select the chi-square statistics.

Cell contributions to the chi-square statistics
displays each table cell's contribution to the Pearson chi-square statistic in the crosstabulation table.

Step 3: Select the Table Statistics

On the **Table Statistics** tab, you can set these options.

1 To compute tests and measures of classification agreement for square tables, select **Measures**.

This option provides McNemar's test for 2×2 tables and Bowker's test of symmetry for tables with more than two response categories. It also produces the simple kappa coefficient, the weighted kappa coefficient, the asymptotic standard errors for the simple and weighted kappas, and the corresponding confidence limits. When there are multiple strata and two response categories, this option also computes Cochran's Q test.

2 To calculate the ordered differences, select **Jonckheere-Terpstra test**.

This option computes the Jonckheere-Terpstra test, which is a nonparametric test for ordered differences among classes. It tests the null hypothesis that the distribution of the response variable does not differ among classes.

3 To calculate a trend test, select **Cochran-Armitage test (for 2 x c and r x 2 tables)**.

This option computes the Cochran-Armitage test for trend, which tests for trend in binomial proportions across levels of a single factor or covariate. This test is appropriate for a contingency table where one variable has two levels and the other variable is ordinal. The two-level variable represents the response, and the other variable represents an explanatory variable with ordered levels.

Step 4: Select the Association Statistics

On the **Association Statistics** tab, you can set these options.

1 Select the tests of association.

Chi-square statistics
This test requests chi-square tests of homogeneity or independence and measures of association that are based on the chi-square statistic. The tests include the Pearson chi-square, likelihood-ratio chi-square, and Mantel-Haenszel chi-square. For 2×2 tables, this test includes Fisher's exact test and the continuity-adjusted chi-square.

You can also select these options:

- **Exact p-values** computes the exact *p*-values for the following statistics: chi-square goodness-of-fit test for one-way tables; Pearson chi-square, likelihood-ratio chi-square, and Mantel-Haenszel chi-square tests for two-way tables.
- **Fisher's exact test (for r x c tables)** computes Fisher's exact test for tables that are larger than 2×2. This test is also known as the Freeman-Halton test.

Note: For some large problems, computation of exact tests might require a considerable amount of time and memory.

2 Select the measures of association.

Measures of association
This test computes several measures of association and their asymptotic standard errors (ASE). The measures include gamma, Kendall's tau-b, Stuart's tau-c, Somers' D (C|R), Somers' D (R|C), the Pearson and Spearman correlation coefficients, lambda (symmetric and asymmetric), and uncertainty coefficients (symmetric and asymmetric).

Binomial proportions and risk difference (for 2x2 tables)
This test requests risks (binomial proportions) and risk differences for 2x2 tables.

Odds ratio and relative risk (for 2x2 tables)
This test requests relative risk measures and their asymptotic Walk confidence limits for 2x2 tables.

3 To calculate the Cochran-Mantel-Haenszel statistics, select **CMH statistics**.

This test requests Cochran-Mantel-Haenszel statistics, which test for association between the row and column variables after adjusting for the remaining variables in a multiway table. These statistics include the CMH correlation statistic, the row mean scores (ANOVA), and the adjusted relative risks and odds ratios.

Step 5: Select the Computation Options

On the **Computation Options** tab, you can set these options.

1 Select the score type for any association, agreement, or trend tests.

The score type specifies the type of row and column scores to use with the Mantel-Haenszel chi-square statistic, Cochran-Mantel-Haenszel statistics, Pearson correlation, Cochran-Armitage test for trend, and weighted kappa coefficient. You can choose from these options:

- **Table** - For numeric variables, table scores are the values of the row and column levels. If the row or column variables are formatted, then the table score is the internal numeric value corresponding to that level. If two or more numeric values are classified into the same formatted level, then the internal numeric value for that level is the smallest of these values.

  For character variables, table scores are defined as the row numbers and column numbers—that is, 1 for the first row, 2 for the second row, and so on.
- **Rank** - Rank scores are defined by row scores and column scores.
- **RIDIT** - Ridit scores are derived from rank scores.
- **Modified RIDIT** - Modified ridit (MODRIDIT) scores represent the expected values of the order statistics for the uniform distribution on (0,1). Modified ridit scores are derived from rank scores.

2 Specify how to treat missing values.

Exclude missing values
specifies that an observation is excluded from a table if the observation has a missing value for any of the variables.

Display missing value frequencies
displays the frequencies of the missing values in the frequency and crosstabulation tables. These frequencies are not included in any computations of percentages, tests, or measures.

Include missing values in calculations
includes missing values in the calculations of percentages and other statistics.

3 Select the confidence level for the calculations. The default value is `95%`.

4 Specify the order in which the values of the variables in the crosstabulation tables are reported. Here are the valid values:

- **Unformatted values** orders the values by their unformatted values. This is the default.
- **Data set order** orders the values by their appearance in the data set.
- **Formatted values** orders the values by their ascending formatted values.
- **Descending frequency** orders the values so that the levels with the highest frequency counts come first.

5 If you are computing exact tests, select the maximum computation time for these calculations.

To compute the Monte Carlo estimates of the exact *p*-values instead of directly computing the exact **p**-values, select the **Use Monte Carlo estimation** check box. Monte Carlo estimation can be useful for large problems that require a great amount of time and memory for exact computations but for which asymptotic approximations might not be sufficient.

6 To display the entire multiway table in one table instead of displaying a separate two-way table for each stratum, select **Display tables in a list format**.

Note: This option is ignored if all output is suppressed.

7 To suppress the plots from the output, select **Suppress plots**. By default, plots are included in the results.

Step 6: Specify the Title and Footnote in the Output

On the **Titles** tab, you specify whether to use the default text for the title and the footnote. Alternatively, you can customize this text. You can use macro variables in titles and footnotes.

Step 7: Create Output Data for the Cell Statistics

On the **Output** tab, you specify whether to create an output data table for the cell statistics. By default, this output data table is saved in the Work library.

9

Visualizing Your Data

Bar Chart

About Bar Chart Step

The Bar Chart step creates horizontal or vertical bar charts that compare numeric values or statistics between different values of a chart variable. Bar charts show the relative magnitude of data by displaying bars of varying height. Each bar represents a category of data.

The bar chart is a bi-directional plot. It uses Category and Measure as role names, and the corresponding axes are called Category Axis and Measure Axis.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirement for Node Connection

In order to run the Bar Chart step, you must connect a table to the input node.

Bar Chart: Assigning Data to Roles

You can create either a vertical or horizontal bar chart. To run the Bar Chart step, you must assign a column to the **Category** role.

To filter the input data source, enter the filter expression in the **Filter** field.

Table 9.1 *Roles in the Bar Chart Step*

Role	Description
Roles	
Category	Specifies the variable that classifies the observations into distinct subsets.
Subcategory	Specifies a variable that is used to group the data. You can specify how the grouped bars are displayed and where the legend appears in relation to the graph.
Measure	Determines how to calculate the bar height. You can use the frequency count (the default), the frequency percent, or the value of a variable in the input data source. If you select **Variable** from the drop-down list, you must assign a column to the **Variable** role. Then you must specify the statistic to use in the calculation of the variable. You can choose from sum, mean, or percent of sum. The default statistic is sum. If you select **Mean**, you can include error bars on your chart.
Additional Roles	
Group analysis by	Creates a separate graph for each BY group.
Weight	Lists the numeric variable whose value represents the weight for each observation in the input data set. If the value for the weight is less than or equal to 0 or the weight is a missing value, the observation is excluded from the analysis.

Bar Chart: Setting Appearance Options

On the **Appearance** tab, you can set these options.

Table 9.2 *Bars*

Option Name	Description
Show labels	Displays the measure values as data labels.
Set color	Specifies the color of the bars. **Note:** This option is not available if you assign a variable to the **Subcategory** role.
Color transparency	Specifies the transparency for the bars. The range is from 0% (completely opaque) to 100% (completely transparent).
Apply color gradient	Applies a gradient to each bar.
Effect	Specifies a special effect to be used on the plot. The data skin affects all filled graph elements. The effect that a data skin has on a filled area depends on the skin type, the graph style, and the color of the skinned element. Most of the skins work best with lighter colors over a medium-to-large filled area.
URL variable	Specifies a character variable that contains URLs for web pages. In the results, click the links to view the website that is associated with that graphical element.

Table 9.3 *Category Axis*

Option Name	Description
Reverse tick values	Specifies that the values for the tick marks be displayed in reverse (descending) order.

Option Name	Description
Show tick values in data order	Places the discrete values for the tick marks in the order in which they appear in the data.
Display label	Enables you to display a label for the axis. By default, the axis label is the name of the variable. However, you can create a custom label. To suppress the label, select **No label**.
Rotate labels in case of tick collisions	Rotates the labels on the X axis by 45 or 90 degrees to prevent the tick marks from overwriting the label text.
Create a reference line	Enables you to create a reference line for the axis. Select the value of the reference line from the drop-down list. Reference lines can be offset. Select the offset from the **Line offset** drop-down list. You can also specify the label for the reference line.

Table 9.4 *Measure Axis*

Option Name	Description
Specify minimum value	Specifies the minimum value on the axis.
Specify maximum value	Specifies the maximum value on the axis.
Show grid lines	Creates grid lines at each tick mark on the measure axis.
Use logarithmic scale	Specifies whether to use a logarithmic scale for the Measure axis. You can specify the base value for the scale as either **10 (default)**, **2**, or **e**. You must specify the baseline for the axis.
Create a reference line	Enables you to create a reference line for the Measure axis. Because only numerical variables can be assigned to the **Measure** role, you must specify the value of the reference line.

You can specify a custom title and footnote for the output. You can also specify the font size for this text.

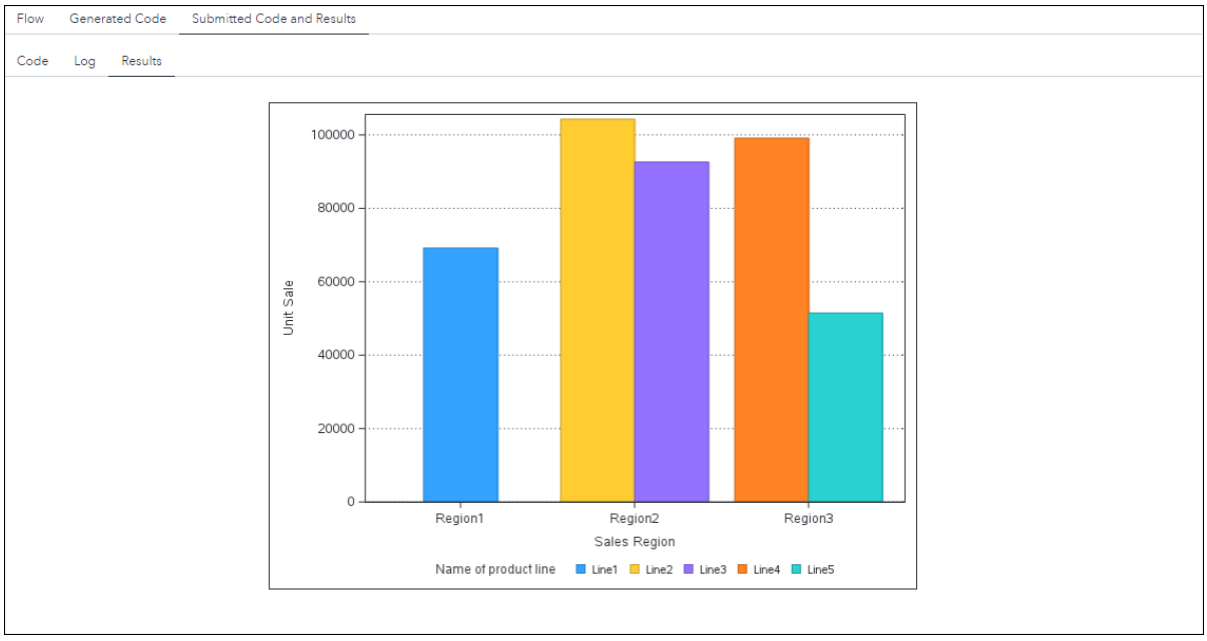
You can specify the width and height of the graph in inches, centimeters, or pixels.

Examples: Bar Chart

Total Sales for Each Product Line

1. In the **Steps** section of the navigation pane, expand the **Visualize Data** folder and double-click **Bar Chart** to add the step to your flow.
2. Add the Sashelp.PriceData table to your flow. Connect the input port of the Bar Chart node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the Bar Chart step, click the Bar Chart node.
4. On the **Data** tab, assign columns to these roles:
   - Assign the **regionName** column to the **Category** role.
   - Assign the **productLine** column to the **Subcategory** role.
5. From the **Measure** drop-down list, select **Variable**. Assign **sale** to the **Variable** role. From the **Statistic** drop-down list, select **Sum (default)**.
6. Run the flow.

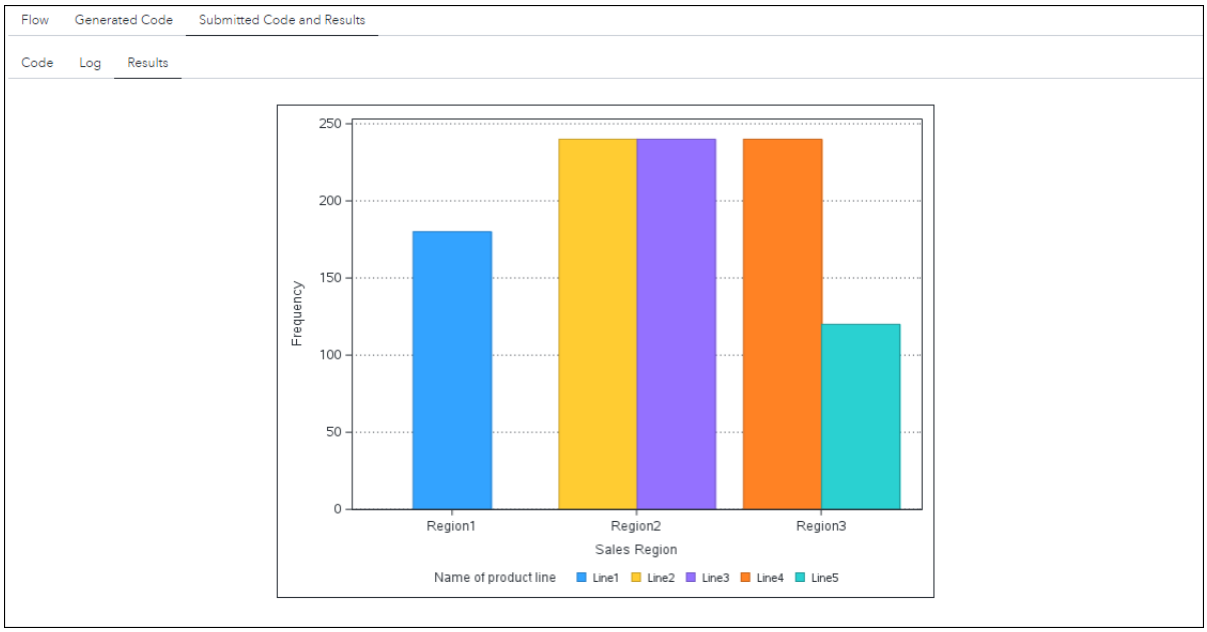
Open the **Submitted Code and Results** tab to view the results.



Total Sales by Frequency

1. In the **Steps** section of the navigation pane, expand the **Visualize Data** folder and double-click **Bar Chart** to add the step to your flow.
2. Add the Sashelp.PriceData table to your flow. Connect the input port of the Bar Chart node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the Bar Chart step, click the Bar Chart node.
4. On the **Data** tab, assign columns to these roles:
   - Assign the **regionName** column to the **Category** role.
   - Assign the **productLine** column to the **Subcategory** role.
5. From the **Measure** drop-down list, select **Frequency count (default)**.
6. Run the flow.

Open the **Submitted Code and Results** tab to view the results.



Include Links in Your Bar Chart

1. In the flow, add a SAS Program node.
2. Add this code to the SAS Program node:

```
data class_url;
  length URL $300;
  set sashelp.class;
```

```
    if Sex='F'
      then URL='http://www.sas.com';
      else URL='http://support.sas.com/';
run;
```

3 Run the SAS Program node. The class_url table is created in the Work library.

4 In the **Steps** section of the navigation pane, expand the **Visualize Data** folder and double-click **Bar Chart** to add the step to your flow.

5 Add the Work.class_url table to your flow. Connect the input port of the Bar Chart node to the data source. For more information, see “Connecting Nodes” on page 12.

6 To set the options for the Bar Chart step, click the Bar Chart node.

7 On the **Data** tab, assign columns to these roles:

- Assign the **Age** column to the **Category** role.
- Assign the **Sex** column to the **Subcategory** role.

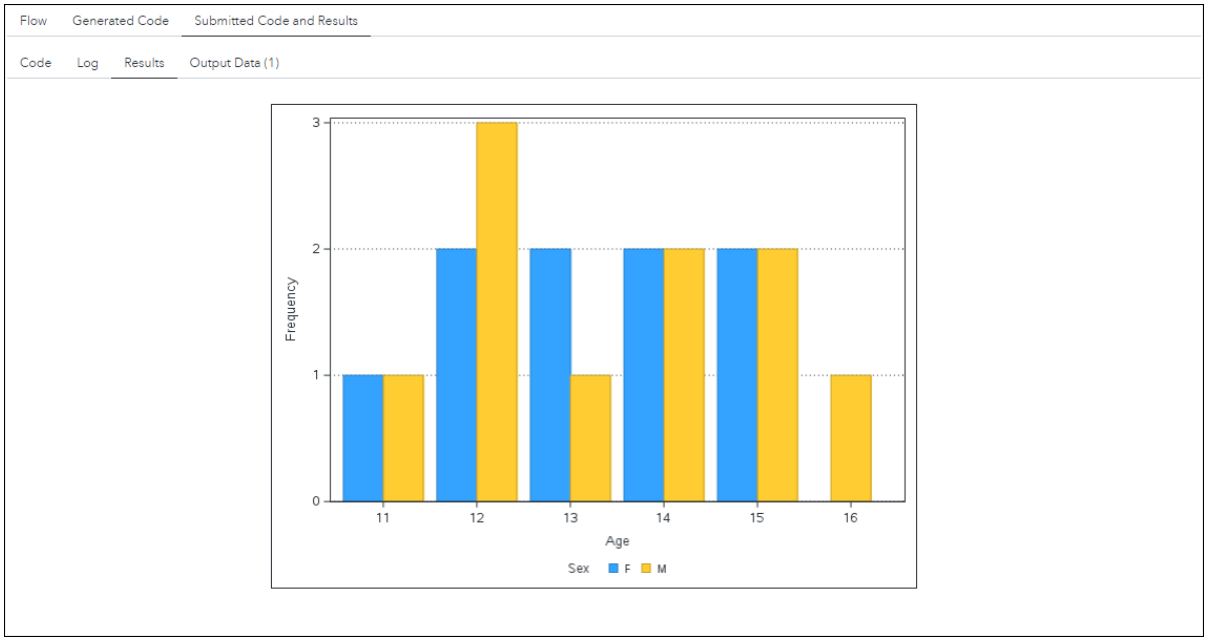
8 On the **Appearance** tab, expand the **Bars > Details**heading.

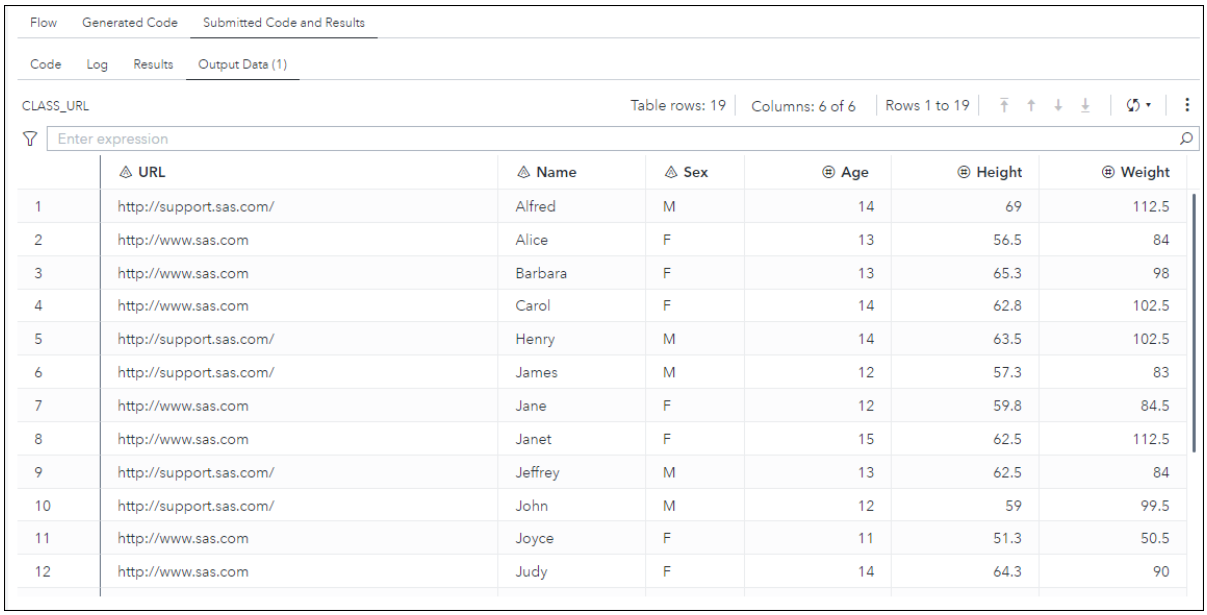
Assign **URL** to the **URL variable** role.

9 Run the flow.

Open the **Submitted Code and Results** tab to view the results.

In the SAS Studio results, click a bar to view the associated website.




Flow Generated Code Submitted Code and Results

Code Log Results Output Data (1)

CLASS_URL Table rows: 19 Columns: 6 of 6 Rows 1 to 19

Enter expression

	URL	Name	Sex	Age	Height	Weight
1	http://support.sas.com/	Alfred	M	14	69	112.5
2	http://www.sas.com	Alice	F	13	56.5	84
3	http://www.sas.com	Barbara	F	13	65.3	98
4	http://www.sas.com	Carol	F	14	62.8	102.5
5	http://support.sas.com/	Henry	M	14	63.5	102.5
6	http://support.sas.com/	James	M	12	57.3	83
7	http://www.sas.com	Jane	F	12	59.8	84.5
8	http://www.sas.com	Janet	F	15	62.5	112.5
9	http://support.sas.com/	Jeffrey	M	13	62.5	84
10	http://support.sas.com/	John	M	12	59	99.5
11	http://www.sas.com	Joyce	F	11	51.3	50.5
12	http://www.sas.com	Judy	F	14	64.3	90

Bar-Line Chart

About the Bar-Line Chart Step

The Bar-Line Chart step creates a vertical bar chart with a line chart overlay.

You can use the Bar-Line Chart step to perform these tasks:

- display and compare exact and relative magnitudes
- examine the contribution of each part to the whole
- determine trends and patterns in the data

Requirement for Node Connection

In order to run the Bar-Line Chart step, you must connect a table to the input node.

Bar-Line Chart: Assigning Data to Roles

To run the Bar-Line Chart step, you must assign a column to the **Category**, **Bar variable**, and **Line variable** roles.

To filter the input data source, enter the filter expression in the **Filter** field.

Table 9.5 Roles in the Bar-Line Chart Step

Role	Description
Roles	
Category	Specifies the variable that classifies the observations into distinct subsets.
Subcategory	Specifies how to group the categorized data. You can specify how the grouped bars are displayed and where the legend appears in relation to the graph.
Bar variable	Specifies a numeric response variable for the bar chart. You can calculate the mean or the sum for the bar variable. By default, the mean is used.
Line variable	Specifies a numeric response variable for the line plot. You can calculate the mean or sum for the line plot. By default, the mean is used.
Additional Roles	
Group analysis by	Creates a separate graph for each BY group.
Weight	Lists the numeric variable whose value represents the weight for each observation in the input data set. If the value for the weight is less than or equal to 0 or the weight is a missing value, the observation is excluded from the analysis.

Bar-Line Chart: Setting Appearance Options

On the **Appearance** tab, you can set these options.

Table 9.6 Bars

Option Name	Description
Show labels	Displays the statistics on each bar.
Set color	Specifies the color of the bars.

Option Name	Description
	Note: This option is not available if you assign a variable to the **Subcategory** role.
Color transparency	Specifies the transparency for the bars. The range is from 0% (completely opaque) to 100% (completely transparent).
Apply color gradient	Applies a gradient to each bar.
Effect	Specifies a special effect to be used on the plot. The data skin affects all filled graph elements. The effect that a data skin has on a filled area depends on the skin type, the graph style, and the color of the skinned element. Most of the skins work best with lighter colors over a medium-to-large filled area.
URL variable	Specifies a character variable that contains URLs for web pages. In the results, click the links to view the website that is associated with that graphical element.

Table 9.7 *Lines*

Option Name	Description
Show labels	Displays the statistic for each data point in the line.
Set color	Enables you to specify the color of the line. **Note:** This option is not available if you assign a variable to the **Subcategory** role.
Thickness	Specifies the thickness (in pixels) of the line. The default is 1 pixel.
Color transparency	Specifies the transparency for the line. The range is from 0% (completely opaque) to 100% (completely transparent).
Line style	Specifies whether to use a solid line or a patterned line.

Option Name	Description
URL variable	Specifies a character variable that contains URLs for web pages. In the results, click the links to view the website that is associated with that graphical element.

Table 9.8 *X Axis*

Option Name	Description
Reverse tick values	Specifies that the values for the tick marks be displayed in reverse (descending) order.
Show tick values in data order	Places the discrete values for the tick marks in the order in which they appear in the data.
Display label	Enables you to display a label for the axis. By default, the axis label is the name of the variable. However, you can create a custom label. To suppress the label, select **No label**.
Rotate labels in case of tick collisions	Rotates the labels on the X axis by 45 or 90 degrees to prevent the tick marks from overwriting the label text.
Create a reference line	Enables you to create a reference line for the axis. Select the value of the reference line from the drop-down list. Reference lines can be offset. Select the offset from the **Line offset** drop-down list. You can also specify the label for the reference line.

Table 9.9 *Y Axis*

Option Name	Description
Use zero baseline	Specifies that the minimum for the Y axis is 0. There is no offset for the lowest data value on the axis. If you do not select this option, the offset for the lowest data value is determined by the step.
Use uniform scale	Uses the same scale for both response axes.

Option Name	Description
Show grid lines	Creates grid lines at each tick mark on the Y axis.
Display label	Enables you to display a label for the axis. By default, the axis label is the name of the variable. However, you can create a custom label. To suppress the label, select **No label**.
Create a reference line	Enables you to create a reference line for the bar or line axis. You must specify the value of this line. You can also specify whether to label the reference line with this value or use a custom label.

You can specify the location of the legend and a custom title and footnote for the output. You can also specify the font size for this text.

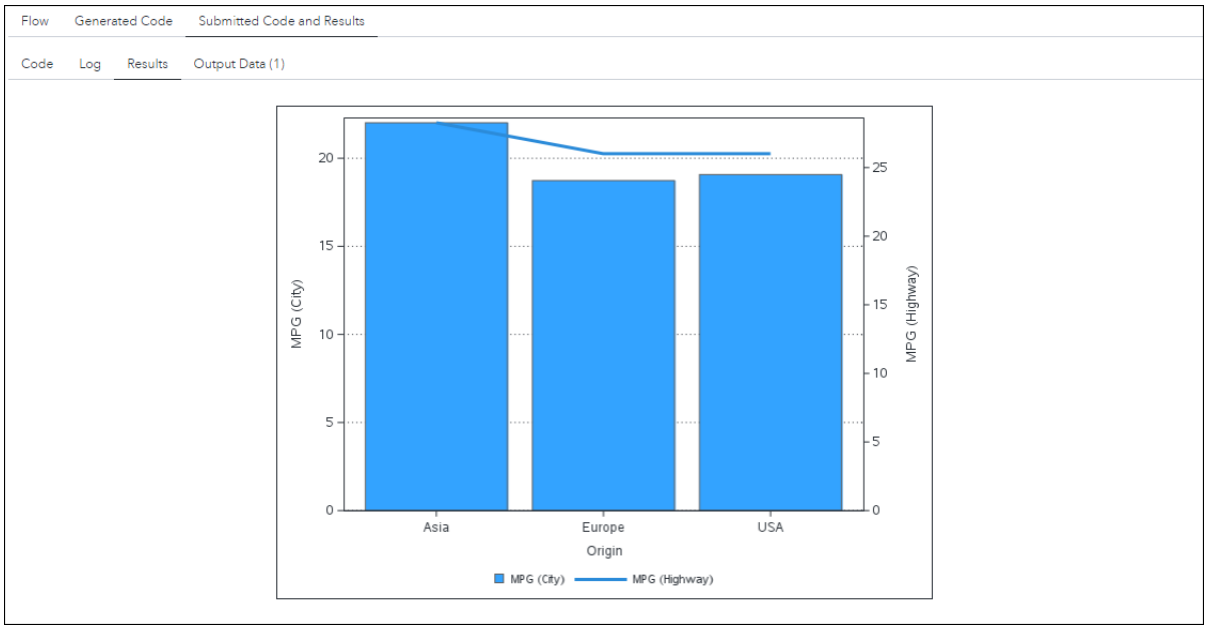
You can specify the width and height of the graph in inches, centimeters, or pixels.

Example: City and Highway Mileage by Country of Origin

You can create a bar-line chart that compares the number of miles per gallon (in the city and on the highway) that cars use depending on their country of origin. The step calculates the mean of the number of miles per gallon in the city and on the highway for each country. This bar-line chart shows that cars from Asia tend to get the highest number of miles per gallon in city and highway driving.

1. In the **Steps** section of the navigation pane, expand the **Visualize Data** folder and double-click **Bar-line Chart** to add the step to your flow.
2. Add the SasHelp.Cars table to your flow. Connect the input port of the Bar-line Chart node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the Bar-line Chart step, click the Bar-line Chart node.
4. On the **Data** tab, assign columns to these roles:
   - Assign the **Origin** column to the **Category** role.
   - Assign the **MPG_City** column to the **Bar variable** role.
   - Assign the **MPG_Highway** column to the **Line variable** role.
5. Run the flow.

Open the **Submitted Code and Results** tab to view the results.



Box Plot

About Box Plot

The Box Plot step represents numeric values measured as intervals. The box plot is a bi-directional plot. It uses Category and Analysis as role names, and the corresponding axes are called Category Axis and Analysis Axis.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirement for Node Connection

In order to run the Box Plot step, you must connect a table to the input node.

Box Plot: Assigning Data to Roles

You can create a vertical or horizontal box plot. To run the Box Plot step, you must assign a column to the **Analysis variable** role.

To filter the input data source, enter the filter expression in the **Filter** field.

Table 9.10 *Roles in the Box Plot Step*

Role	Description
Roles	
Analysis variable	Specifies the analysis variable for the plot.
Category	Creates a box plot for each distinct value of the category variable.
Subcategory	Creates a box plot for each distinct value of the subcategory within the category. If you assign a variable to this role, you can order the subcategories in these ways: ■ in the order in which subcategories appear in the data ■ in the reverse order that the subcategories appear in the data ■ in ascending alphabetical order ■ in descending alphabetical order To help distinguish between the subcategories, a legend is created. You can place this legend inside or outside the graph.
Additional Roles	
Group analysis by	Creates a separate graph for each BY group.
Weight	Lists the numeric variable whose value represents the weight for each observation in the input data set. If the value for the weight is less than or equal to 0 or the weight is a missing value, the observation is excluded from the analysis.

Box Plot: Setting Options

On the **Options** tab, you can set these options.

Table 9.11 Boxes Option

Option Name	Description
Width	Specifies the width of each box.
Notches	Specifies that the boxes be notched.
Color transparency	Specifies the transparency for the bars. The range is from 0% (completely opaque) to 100% (completely transparent).
Effect	Specifies a special effect to be used on the plot. The data skin affects all filled graph elements. The effect that a data skin has on a filled area depends on the skin type, the graph style, and the color of the skinned element. Most of the skins work best with lighter colors over a medium-to-large filled area.
Cap shape	Specifies whether to display the cap lines for the whiskers. If you select this option, you can select the shape of the whisker cap lines. Here are the valid values: ■ **Bracket** displays a straight line with brackets. ■ **Line** displays a straight line. **Serif** displays a short straight line.

Table 9.12 Category Axis

Option Name	Description
Reverse tick values	Specifies that the values for the tick marks be displayed in reverse (descending) order.
Show tick values in data order	Places the discrete values for the tick marks in the order in which they appear in the data.
Display label	Enables you to display a label for the axis. By default, the axis label is the name of the variable. However, you can create a custom label. To suppress the label, select **No label**.

Option Name	Description
Rotate labels in case of tick collisions	Rotates the labels on the X axis by 45 or 90 degrees to prevent the tick marks from overwriting the label text.
Create a reference line	Enables you to create a reference line for the axis. Select the value of the reference line from the drop-down list. Reference lines can be offset. Select the offset from the **Line offset** drop-down list. You can also specify the label for the reference line.

Table 9.13 *Analysis Axis*

Option Name	Description
Specify minimum value	Specifies the minimum value on the axis.
Specify maximum value	Specifies the maximum value on the axis.
Show grid lines	Creates grid lines at each tick mark on the measure axis.
Use logarithmic scale	Specifies whether to use a logarithmic scale for the Measure axis. You can specify the base value for the scale as either **10 (default)**, **2**, or **e**. You must specify the baseline for the axis.
Create a reference line	Enables you to create a reference line for the Analysis axis. Specify a value for this line. You can also specify whether to label the reference line with this value or use a custom value.

You can specify a custom title and footnote for the output. You can also specify the font size for this text.

You can specify the width and height of the graph in inches, centimeters, or pixels.

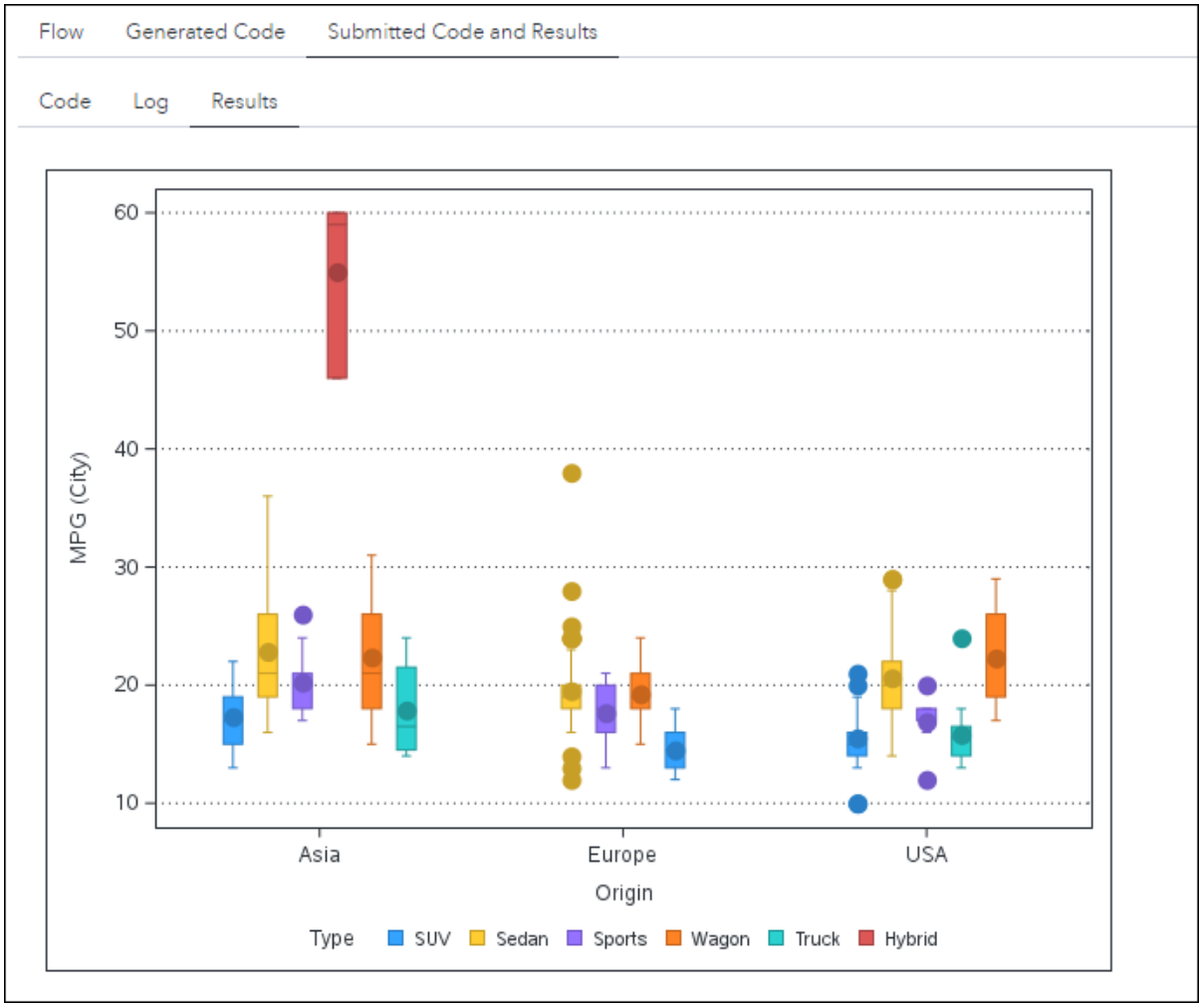
Example: Box Plots Comparing MPG (City) for Cars

This example creates three groups of box plots that compare how many miles per gallon (city) cars consume depending on their area of origin (Asia, Europe, and the United States).

To create this example:

1. In the **Steps** section of the navigation pane, expand the **Visualize Data** folder and double-click **Box Plot** to add the step to your flow.
2. Add the Sashelp.Cars table to your flow. Connect the input port of the Box Plot node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the Box Plot step, click the Box Plot node.
4. On the **Data** tab, assign columns to these roles:
   - To the **Analysis variable** role, assign **MPG_City**.
   - To the **Category** role, assign **Origin**.
   - To the **Subcategory** role, assign **Type**.
5. Run the flow.

Open the **Submitted Code and Results** tab to view the results.



Bubble Plot

About the Bubble Plot Step

The Bubble Plot step explores the relationship between three or more variables. In a bubble plot, two variables determine the location of the bubble centers, and a third variable specifies the size of each bubble. A fourth variable can be used to determine the colors of the bubbles.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirement for Node Connection

In order to run the Bubble Plot step, you must connect a table to the input node.

Bubble Plot: Assigning Data to Roles

To filter the input data source, enter the filter expression in the **Filter** field.

Table 9.14 *Roles in the Bubble Plot Step*

Role	Description
X axis	Specifies the variable for the X axis.
Y axis	Specifies the variable for the Y axis.
Bubble size	Specifies a numeric variable that controls the size of the bubbles. The minimum and maximum values automatically provide the range that is used to determine bubble size. You can set these values on the **Options** tab.
Bubble color	Specifies the color of the bubbles.

Bubble Plot: Setting Appearance Options

On the **Options** tab, set these options.

Table 9.15 *Bubble Options*

Option Name	Description
Set color	Specifies the color for the bubbles when a column is not assigned to the **Bubble color** role.
Minimum radius	Specifies the radius of the smallest bubble.
Maximum radius	Specifies the radius of the largest bubble.
Color transparency	Specifies the transparency of the bars. The range is from 0% (completely opaque) to 100% (completely transparent).
Effect	Specifies a special effect to be used on the plot. The data skin affects all filled graph elements. The effect that a data skin has on a filled area depends on the skin type, the graph style, and the color of the skinned element. Most of the skins work best with lighter colors over a medium to large filled area.
URL variable	Specifies a character variable that contains URLs for web pages. In the results, click the links to view the website associated with that graphical element.

Table 9.16 *Bubble Labels Options*

Option Name	Description
Bubble label	If you assign a variable to the Bubble label role, you can determine the label color, the font size for the label text, and the label position.

Table 9.17 X and Y Axis Options

Option Name	Description
Specify minimum value (numeric variables only)	Specifies the minimum value on the axis.
Specify maximum value (numeric variables only)	Specifies the maximum value on the axis.
Show grid lines	Creates grid lines at each tick on the axis.
Display Label	Enables you to display a label for the axis. By default, the axis label is the name of the variable. However, you can create a custom label. To suppress the label, select **No label**.
Use logarithmic scale (numeric variables only)	Specifies whether to use a logarithmic scale on the axis. You can specify the base value for the scale as either **10 (default)**, **2**, or **e**.
Rotate labels in case of tick collisions	Rotates the labels on the X axis by 45 or 90 degrees to prevent the tick marks from overwriting the label text. **Note:** This option is not available for the Y axis.
Create a reference line	Enables you to create a reference line for the axis. You must specify the value of this line. You can also specify whether to label the reference line with this value or use a custom value.

You can specify a custom title and footnote for the output. You can also specify the font size for this text.

You can specify the width and height of the graph in inches, centimeters, or pixels.

Example: Bubble Plot

To create this example:

1. In the **Steps** section of the navigation pane, expand the **Visualize Data** folder and double-click **Bubble Plot** to add the step to your flow.
2. Add the Sashelp.Class table to your flow. Connect the input port of the Bubble Plot node to the data source. For more information, see “Connecting Nodes” on page 12.

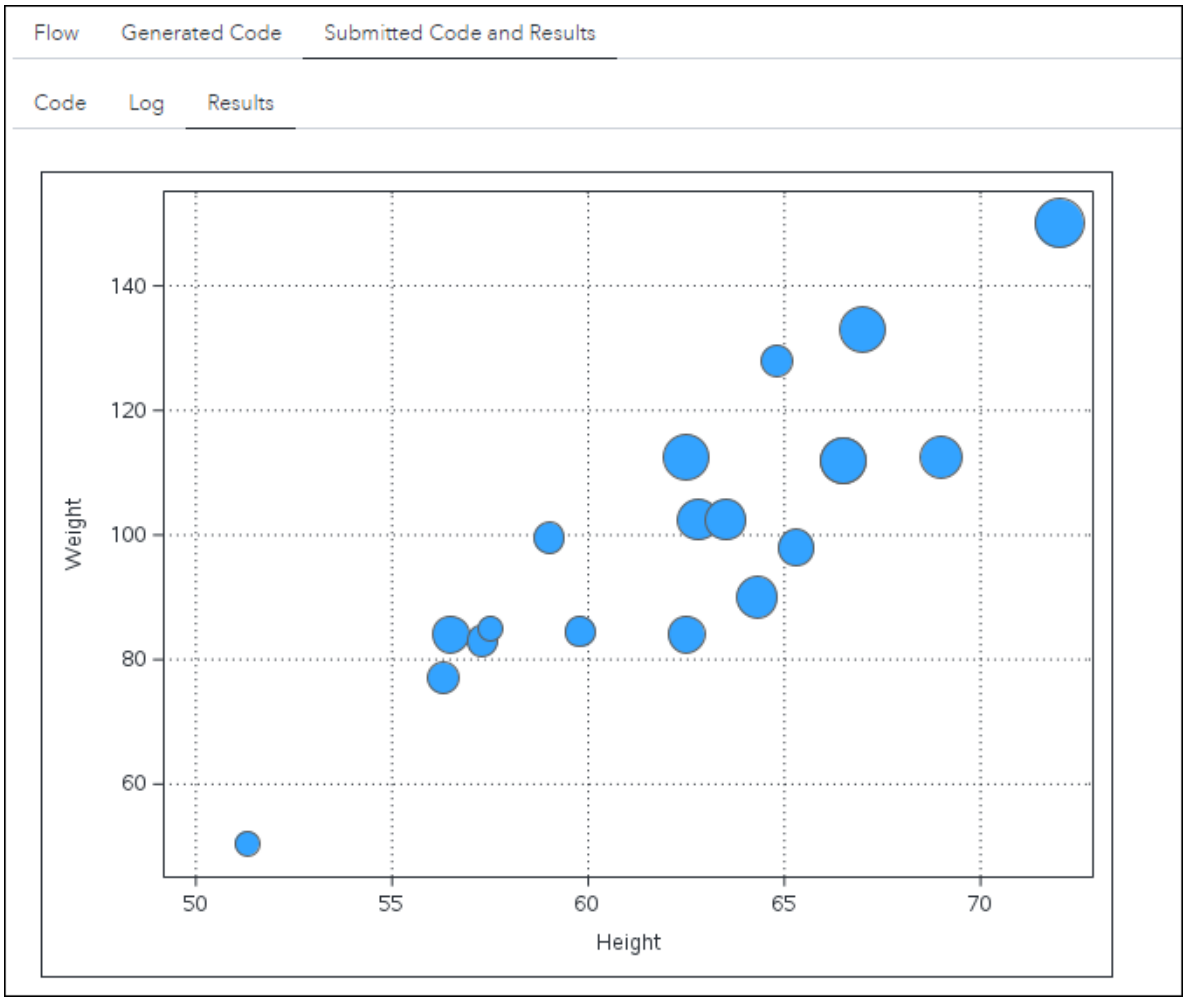
3 To set the options for the Bubble Plot step, click the Bubble Plot node.

4 On the **Data** tab:

- For the X axis, assign **Height** to the **Column** role.
- For the Y axis, assign **Weight** to the **Column** role.
- To the **Bubble size** role, assign the **Age** column.

5 Run the flow.

Open the **Submitted Code and Results** tab to view the results.



Heat Map

About the Heat Map Step

The Heat Map step displays the magnitude of the response based on two variables. The response is represented as a color value from a color gradient.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirement for Node Connection

In order to run the Heat Map step, you must connect a table to the input node.

Heat Map: Assigning Data to Roles

You can create a vertical or horizontal box plot. To run the Heat Map step, you must assign a column to the **X axis** role.

To filter the input data source, enter the filter expression in the **Filter** field.

Table 9.18 *Roles in the Heat Map Step*

Role	Description
Roles	
X axis	Specifies the variable for the X axis. Specify the variable type and whether the values are discrete or continuous. If the values are continuous, you can specify the number of bins. The default number of bins is `30`.
Y axis	Specifies the variable for the Y axis. Specify whether the values are discrete or continuous. If the values are continuous, you can specify the number of bins. The default number of bins is `40`.

Role	Description
Color	Specifies the numeric variable to use as the color response variable. Use the **Statistic** drop-down list to specify the calculation for this variable. The default statistic is the sum.
Additional Roles	
Group analysis by	Creates a separate graph for each BY group.
Weight	Lists the numeric variable whose value represents the weight for each observation in the input data set. If the value for the weight is less than or equal to 0 or the weight is a missing value, the observation is excluded from the analysis.

Heat Map: Setting the Appearance Options

On the **Options** tab, you can set these options.

Table 9.19 *Regression Options*

Option Name	Description
Add a regression line	Creates a fitted regression line or curve. **Note:** This option is available if you assigned variables to both the X-axis and Y-axis roles, and both roles contain continuous variables.
Degree of the polynomial fit	Specifies the degree of the polynomial fit. For example, 1 specifies a linear fit, 2 specifies a quadratic fit, and 3 specifies a cubic fit.
Prediction limits for individuals	Specifies whether to create the prediction limits for the individual predicted values. You can specify the confidence level for the confidence limits.
Legend location	Specifies whether the legend is placed outside or inside the axis area.

Table 9.20 Intensity Color Options

Option Name	Description
Intensity color options	You can specify the color scheme for showing the response frequency in the heat map. The higher the frequency, the more saturated the color in the heat map. ■ The default color model uses an intensity color scheme of blue and red. The default colors are white, green, yellow, and red. ■ You can create a custom color scheme by selecting four colors.

Table 9.21 X Axis and Y Axis Options

Option Name	Description
Specify minimum value (numeric variables only)	Specifies the minimum value on the axis.
Specify the maximum value (numeric variables only)	Specifies the maximum value on the axis.
Show grid lines	Creates grid lines at each tick on the X or Y axis.
Display label	Enables you to display a label for the axis. By default, the axis label is the name of the variable. However, you can create a custom label. To suppress the label, select **No label**.
Use logarithmic scale (continuous numeric variables only)	Specifies whether to use a logarithmic scale for the X or Y axis. You can specify the base value as either `10` (default), `2`, or `e`.
Rotate labels in case of tick collisons	Specifies whether to rotate the labels by 45 degrees or 90 degrees so that the labels are not overwritten when the ticks on the axis are close together. **Note:** This option is available only on the X axis.
Create a reference line	Enables you to create a reference line for the axis. ■ If the axis variable is discrete, select the value of the reference line from the drop-down list. These values are

Option Name	Description
	determined by the variable that you assigned to the axis. You can also specify whether to label the reference line with this value or use a custom label. ■ If the axis variable is continuous, specify the value of the reference line. You can also specify whether to label the reference line with this value or use a custom label.

You can specify a custom title and footnote for the output. You can also specify the font size for this text.

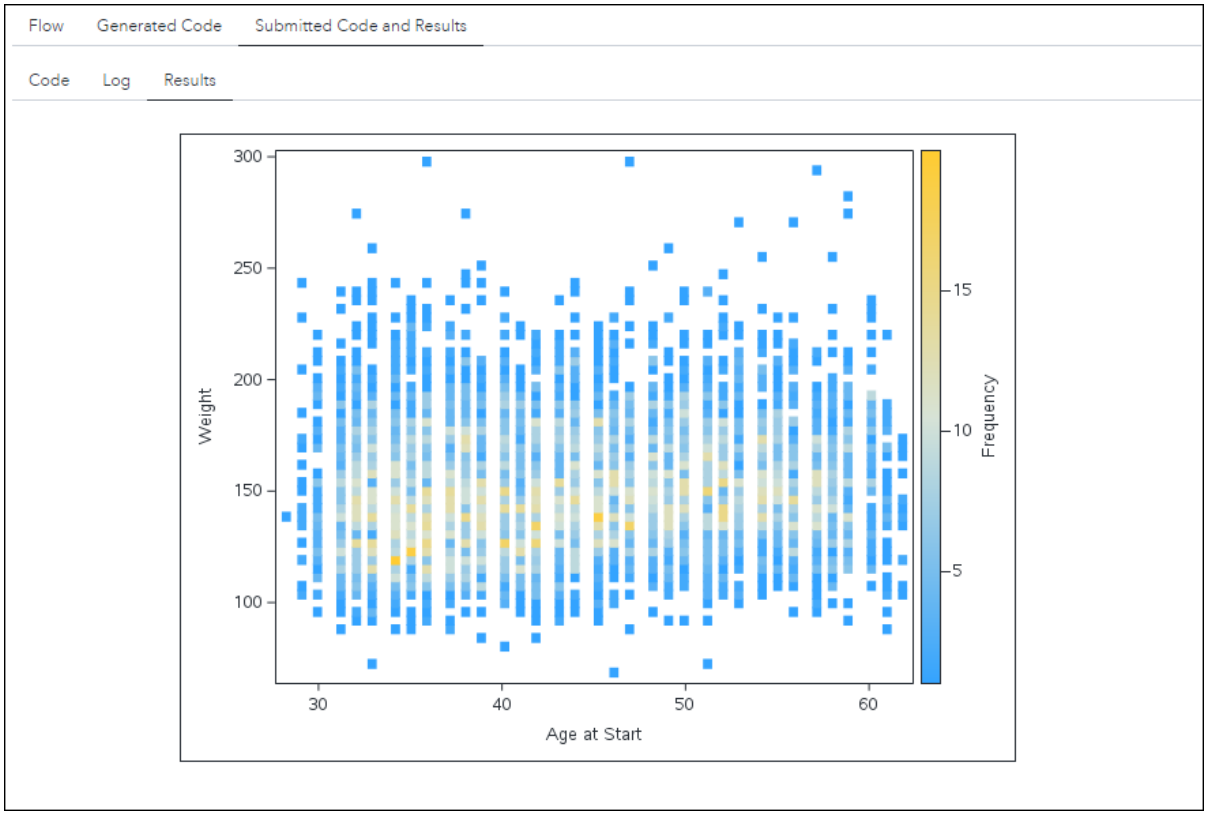
You can specify the width and height of the graph in inches, centimeters, or pixels.

Example: Heat Map

To create this example:

1. In the **Steps** section of the navigation pane, expand the **Visualize Data** folder and double-click **Heat Map** to add the step to your flow.
2. Add the Sashelp.Heart table to your flow. Connect the input port of the Heat Map node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the Heat Map step, click the Heat Map node.
4. On the **Data** tab, assign columns to these roles:
   - To the **X axis** role, assign **AgeAtStart**.
   - To the **Y axis** role, assign **Weight**.
5. Run the flow.

Open the **Submitted Code and Results** tab to view the results.



Histogram

About the Histogram Step

The Histogram step creates a chart that displays the frequency distribution of a numeric variable.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirement for Node Connection

In order to run the Histogram step, you must connect a table to the input node.

Histogram: Assigning Data to Roles

To filter the input data source, enter the filter expression in the **Filter** field.

Table 9.22 *Roles in the Histogram Step*

Role	Description
Roles	
Analysis variable	Specifies the variable to use in the analysis. Use the **Scale** option to specify the scaling that is applied to the vertical axis. You can choose from these options: ■ **Percent (default)** - The axis displays values as a percentage of the total. ■ **Count** - The axis displays the frequency count. ■ **Proportion** - The axis displays values as proportions (0.0 to 1.0) of the total.
Additional Roles	
Group analysis by	Creates a separate graph for each BY group.
Weight	Lists the numeric variable whose value represents the weight for each observation in the input data set. If the value for the weight is less than or equal to 0 or the weight is a missing value, the observation is excluded from the analysis.

Histogram: Setting the Appearance Options

On the **Options** tab, you can set these options.

Table 9.23 *Density Curve Options*

Option Name	Description
Density Curves	You can specify whether to create a density curve that shows the distribution

Option Name	Description
	of values for a numeric variable. You can create density curves for normal and kernel distributions.

Table 9.24 *Bin Options*

Option Name	Description
Specify number of bins	Specifies the number of bins in the histogram. Valid values range from `2` to `10000`. The bins always span the range of data. The step tries to produce tick values that are easily interpreted (for example, `5`, `10`, `15`, `20`). Sometimes the location of the first bin and the bin width might be adjusted. By default, the step automatically determines the number of bins.
Set color	Specifies the color of the bins in the histogram.
Color transparency	Specifies the transparency for the bars. The range is from 0% (completely opaque) to 100% (completely transparent).
Apply color gradient	Specifies whether to apply a gradient to the bins.
Effect	Specifies a special effect to be used on the plot. The data skin affects all filled graph elements. The effect that a data skin has on a filled area depends on the skin type, the graph style, and the color of the skinned element. Most of the skins work best with lighter colors over a medium to large filled area.

Table 9.25 *X Axis and Y Axis Options*

Option Name	Description
Show tick marks at bin midpoints	Creates tick marks at the midpoints of the value bins on the X axis. **Note:** This option is available only on the X axis.

Option Name	Description
Specify minimum value	Specifies the minimum value on the axis.
Specify the maximum value	Specifies the maximum value on the axis.
Show grid lines	Creates grid lines at each tick on the X or Y axis.
Display label	Enables you to display a label for the axis. By default, the axis label is the name of the variable. However, you can create a custom label. To suppress the label, select **No label**.
Rotate labels in case of tick collisons	Specifies whether to rotate the labels by 45 degrees or 90 degrees so that the labels are not overwritten when the ticks on the axis are close together. **Note:** This option is available only on the X axis.
Create a reference line	Enables you to specify a reference line for the axis. You can also specify whether to label the reference line with this value or use a custom label.

You can specify a custom title and footnote for the output. You can also specify the font size for this text.

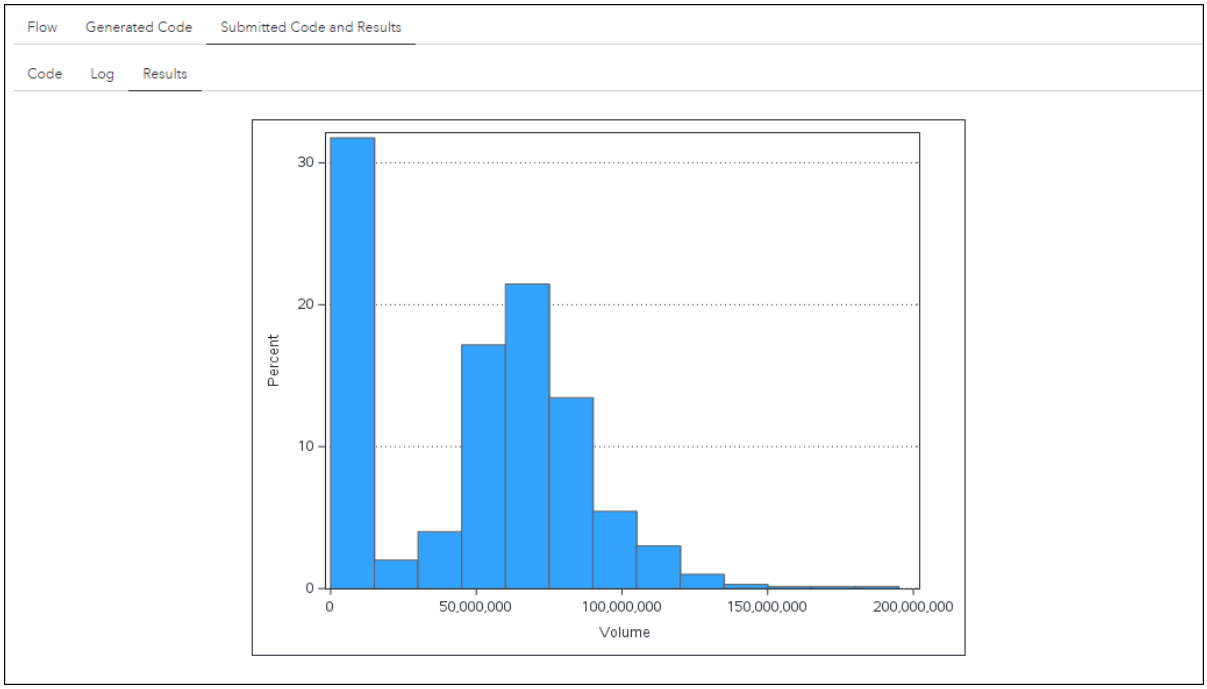
You can specify the width and height of the graph in inches, centimeters, or pixels.

Example: Histogram of Stock Volume

To create this example:

1. In the **Steps** section of the navigation pane, expand the **Visualize Data** folder and double-click **Histogram** to add the step to your flow.
2. Add the Sashelp.Stocks table to your flow. Connect the input port of the Histogram node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the Histogram step, click the Histogram node.
4. To the **Analysis variable** role, assign the **Volume** column.
5. Run the flow.

Open the **Submitted Code and Results** tab to view the results.



Line Chart

About Line Chart

The Line Chart step shows the mathematical relationships between variables by revealing trends or patterns of data points.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirement for Node Connection

In order to run the Line Chart step, you must connect a table to the input node.

Line Chart: Assigning Data to Roles

To run the Line Chart step, you must assign a column to the **Category** role.

To filter the input data source, enter the filter expression in the **Filter** field.

Table 9.26 *Roles in the Line Chart Step*

Role	Description
Roles	
Category	Specifies the variable that classifies the observations into distinct categories.
Subcategory	Specifies a variable that further classifies the data into smaller subcategories. If you assign a variable to this role, you can include a legend on the line chart or use labels to identify the lines. You can place the legend inside or outside the graph.
Measure	Determines how to calculate the data point on the line chart. You can use the frequency count (the default), the frequency percent, or the value of a selected column. After you select a column, you must specify the statistic to use in the calculation of the column. You can choose from sum (the default), mean, or percent of sum. If you select the mean, you can also include error bars on your chart.
Additional Roles	
Group analysis by	Creates a separate graph for each BY group.
Weight	Lists the numeric variable whose value represents the weight for each observation in the input data set. If the value for the weight is less than or equal to 0 or the weight is a missing value, the observation is excluded from the analysis.

Line Chart: Setting Options

On the **Options** tab, you can set these options.

Table 9.27 Lines

Option Name	Description
Show data labels	Displays the value of each data point on the line chart.
Show line label	Displays on the line the label from the response variable (Y axis). You can also specify a custom label. **Note:** This option is not available if you assign a column to the **Subcategory** role.
Set color	Specifies the color for the line. This option is not available if you assign a column to the **Subcategory** role.
Thickness	Specifies the thickness (in pixels) of the line.
Color transparency	Specifies the transparency for the bars. The range is from 0% (completely opaque) to 100% (completely transparent).
Line style	Specifies the type of line to use in the chart.
URL variable	Specifies a character variable that contains URLs for web pages. In the results, click the links to view the website associated with that graphical element.

Table 9.28 X Axis

Option Name	Description
Reverse tick values	Specifies that the values for the tick marks be displayed in reverse (descending) order.
Show tick values in data order	Places the discrete values for the tick marks in the order in which they appear in the data.
Display label	Enables you to display a label for the axis. By default, the axis label is the name of the variable. However, you can create a custom label. To suppress the label, select **No label**.

Option Name	Description
Rotate labels in case of tick collisions	Rotates the labels on the X axis by 45 or 90 degrees to prevent the tick marks from overwriting the label text.
Create a reference line	Enables you to create a reference line for the axis. Select the value of the reference line from the drop-down list. Reference lines can be offset. Select the offset from the **Line offset** drop-down list. You can also specify the label for the reference line.

Table 9.29 Y Axis

Option Name	Description
Specify minimum value	Specifies the minimum value on the axis.
Specify maximum value	Specifies the maximum value on the axis.
Show grid lines	Creates grid lines at each tick mark on the measure axis.
Use logarithmic scale	Specifies whether to use a logarithmic scale for the Measure axis. You can specify the base value for the scale as either **10 (default)**, **2**, or **e**. You must specify the baseline for the axis.
Create a reference line	Enables you to create a reference line for the Analysis axis. Specify a value for this line. You can also specify whether to label the reference line with this value or use a custom value.

You can specify a custom title and footnote for the output. You can also specify the font size for this text.

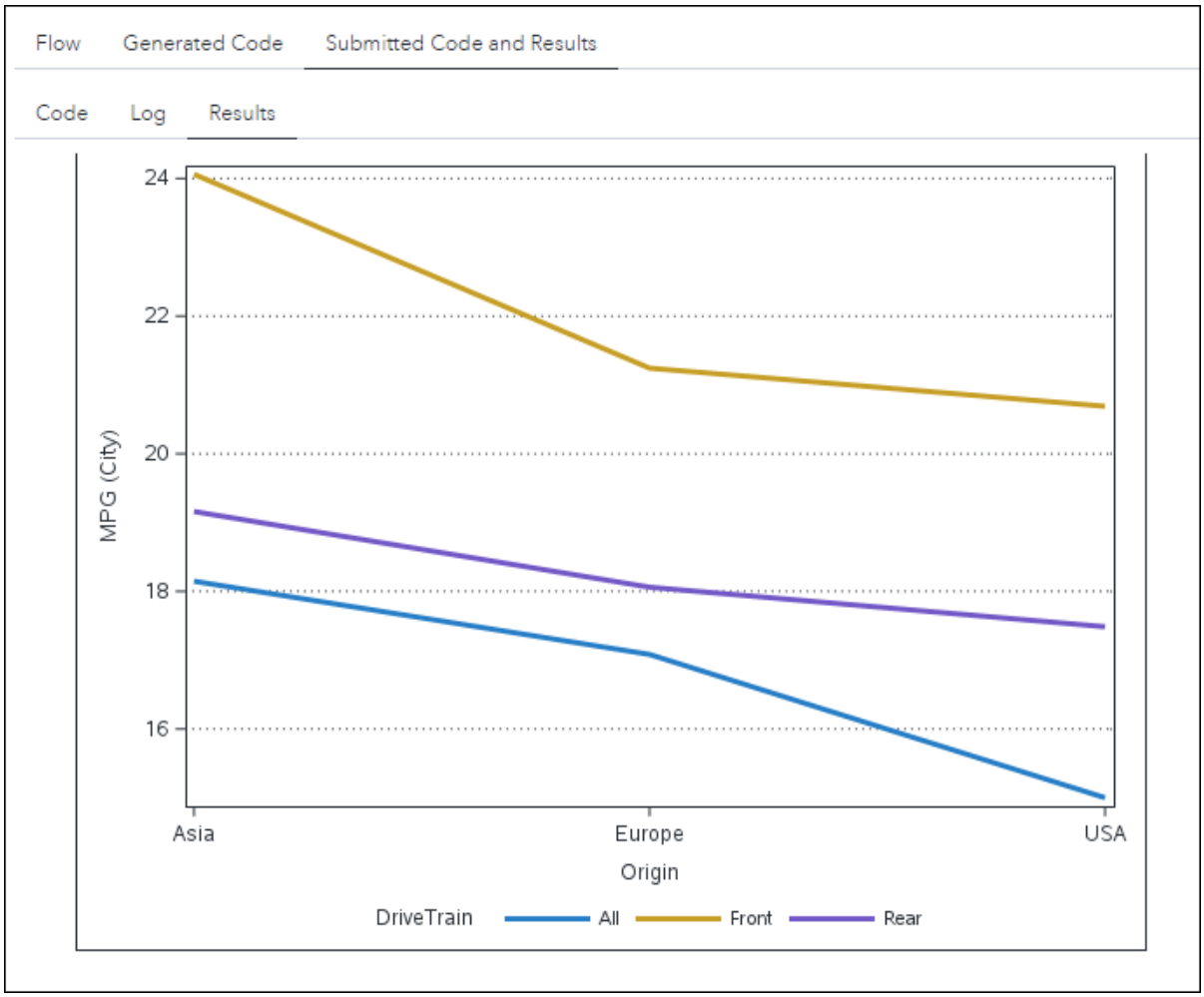
You can specify the width and height of the graph in inches, centimeters, or pixels.

Example: Displaying the Mean MPG per Origin and DriveTrain

To create this example:

1. In the **Steps** section of the navigation pane, expand the **Visualize Data** folder and double-click **Line Chart** to add the step to your flow.
2. Add the Sashelp.Cars table to your flow. Connect the input port of the Line Chart node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the Line Chart step, click the Line Chart node.
4. On the **Data** tab, assign columns to these roles:
   - To the **Category** role, assign **Origin**.
   - To the **Subcategory** role, assign **DriveTrain**.
5. Specify the measure for the analysis.
   a. From the **Measure** drop-down list, select **Select a column from the input data**.
   b. Select **MPG_City** as the column.
   c. From the **Statistic** drop-down list, select **Mean**.
6. Run the flow.

Open the **Submitted Code and Results** tab to view the results.



Pie Chart

About the Pie Chart Step

The Pie Chart step creates pie charts that represent the relative contribution of the parts to the whole by displaying data as wedge-shaped "slices" of a circle. Each slice represents a category of data. The size of a slice represents the contribution of the data to the total chart statistic.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirement for Node Connection

In order to run the Pie Chart step, you must connect a table to the input node.

Pie Chart: Assigning Data to Roles

To filter the input data source, enter the filter expression in the **Filter** field.

Table 9.30 *Roles in the Pie Chart Step*

Role	Description
Category	Specifies the variable that classifies the observations into distinct subsets.
Subcategory	Specifies a variable that is used to group the data within each category.
Measure	Determines how to calculate the size of the slice. You can use the frequency count (the default), the frequency percent, or the value of a variable in the input data source. If you select **Variable** from the drop-down list, you must assign a column to the **Variable** role. Then you must specify the statistic to use in the calculation of the

Role	Description
	variable. You can choose from sum or mean.

Pie Chart: Setting Appearance Options

On the **Options** tab, you can set these options.

Table 9.31 *Pie Options*

Option Name	Description
Starting point	Specifies where to create the first slice in the pie chart.
Label location	Specifies where to display the label for the pie slice. By default, the Pie Chart step determines the best location for the slice. **Note:** This option is not available if you assign a variable to the **Subcategory** role.
Set label font	Specifies the font size for the label.
Color transparency	Specifies the transparency for the bars. The range is from 0% (completely opaque) to 100% (completely transparent).
Effect	Specifies a special effect to be used on the plot. The data skin affects all filled graph elements. The effect that a data skin has on a filled area depends on the skin type, the graph style, and the color of the skinned element. Most of the skins work best with lighter colors over a medium to large filled area.
URL variable	Specifies a character variable that contains URLs for web pages. In the results, click the links to view the website associated with that graphical element. **Note:** If the step generates an “Other” slice in the pie chart, there is no URL associated with this slice. Therefore, this slice does not contain a link.

You can specify a custom title and footnote for the output. You can also specify the font size for this text.

You can specify the width and height of the graph in inches, centimeters, or pixels.

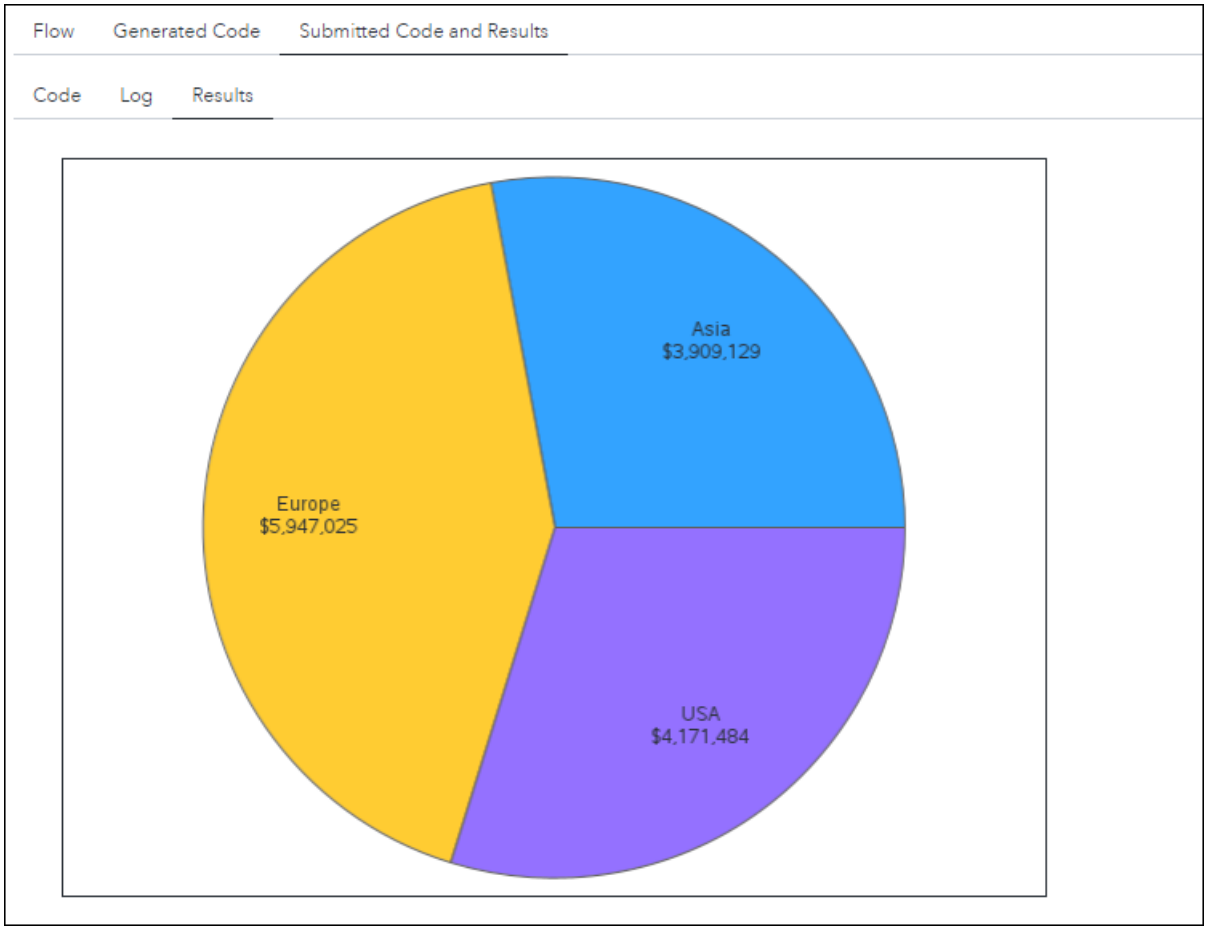
Example: Pie Chart That Shows Total MSRP by Region

In this example, you want to compare the manufacturer's suggested retail price (MSRP) by region of origin. The resulting pie chart consists of six rings—one for each car type. The rings are then subset into the MSRP values for the three regions: Asia, Europe, and USA. Using this chart, you can compare the total MSRP values for each region. The ring for the SUV car type shows that the USA has the highest MSRP and that Europe has the lowest MSRP.

To create this example:

1. In the **Steps** section of the navigation pane, expand the **Visualize Data** folder and double-click **Pie Chart** to add the step to your flow.
2. Add the Sashelp.Cars table to your flow. Connect the input port of the Pie Chart node to the data source. For more information, see "Connecting Nodes" on page 12.
3. To set the options for the Pie Chart step, click the Pie Chart node.
4. On the **Data** tab, assign **Origin** to the **Category** role.
5. From the **Measure** drop-down list, select **Select a column from input data**. Assign **MSRP** to the **Column** role. From the **Statistic** drop-down list, select **Sum (default)**.
6. Run the flow.

Open the **Submitted Code and Results** tab to view the results.



Scatter Plot

About the Scatter Plot Step

The Scatter Plot step creates plots that show the relationships between two or three variables by revealing patterns or concentrations of data points. For example, a two-dimensional scatter plot can display the heights and weights of all students in a class.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirement for Node Connection

In order to run the Scatter Plot step, you must connect a table to the input node.

Scatter Plot: Assigning Data to Roles

To filter the input data source, enter the filter expression in the **Filter** field.

Table 9.32 *Roles in the Scatter Plot Step*

Role	Description
Variable type	Specifies whether the input data is numeric or character.
Column	Specifies the column for the X or Y axis.
Group	Specifies a variable that is used to group the data. The plot elements for each group value are automatically distinguished by different visual attributes. You can specify whether to include the legend outside or inside the axis area.
Additional Roles	
Group analysis by	Creates a separate graph for each BY group.

Scatter Plot: Setting Appearance Options

Table 9.33 *Fit Curves Options*

Option Name	Description
Regression	Creates a plot with the fitted regression line. You must specify the degree of the polynomial fit. The default value is `1`. You can specify whether to include the prediction limits for the individual predicted values.

Option Name	Description
Loess	Creates a fitted LOESS curve. You can specify a smoothing value and whether to include the confidence limits for the means.
Penalized B-spline	Creates a fitted penalized B-spline curve. You must specify the degree of the spline transformation. You can specify the prediction limits for the individual predicted values.

Table 9.34 *Markers*

Option Name	Description
Marker label	Displays a label for each data point. If you specify a variable, the values of that variable are used for the data labels. You can specify the symbol type, color, and size of the markers. You can also specify the degree of transparency for the plot. The range is from 0% (completely opaque) to 100% (completely transparent). **Note:** The **Set color** option is not available if you assign a variable to the **Group** role.
URL variable	Specifies a character variable that contains URLs for web pages. In the results, click the links to view the website that is associated with that graphical element.

Table 9.35 *Options for the X Axis and Y Axis*

Option Name	Description
Specify minimum value (numeric variables only)	Specifies the minimum value on the axis.
Specify maximum value (numeric variables only)	Specifies the maximum value.
Show grid lines	Creates grid lines at each tick on the axis.
Display label	Enables you to display a label for the axis. By default, the axis label is the name of the variable. However, you can

Option Name	Description
	create a custom label. To suppress the label, select **No label**.
Use logarithmic scale (numeric variables only; cannot be used with linear regression fits)	Specifies whether to use a logarithmic scale for the X or Y axis. You can specify the base value for the axis as either **10 (default)**, **2**, or **e**.
Rotate labels in case of tick collisions	Rotates the labels on the X axis by 45 or 90 degrees to prevent the tick marks from overwriting the label text. **Note:** This option is not available for the Y axis.
Create a reference line	Enables you to create a reference line for the axis. ■ If the axis variable contains character values, select the value of the reference line from the drop-down list. These values are determined by the variable that you assigned to the axis. You can also specify whether to label the reference line with this value or use a custom label. ■ If the axis variable contains numeric values, specify the value of the reference line. You can also specify whether to label the reference line with this value or use a custom label.

You can specify a custom title and footnote for the output. You can also specify the font size for this text.

You can specify the width and height of the graph in inches, centimeters, or pixels.

Example: Series Plot of Stock Trends

1 In the **Steps** section of the navigation pane, expand the **Visualize Data** folder and double-click **Scatter Plot** to add the step to your flow.

2 Add the Sashelp.Stocks table to your flow. Connect the input port of the Scatter Plot node to the data source. For more information, see “Connecting Nodes” on page 12.

3 To set the options for the Scatter Plot step, click the Scatter Plot node.

4 On the **Data** tab, assign columns to these roles:

   ■ Under the X Axis heading, select **Numeric (default)** as the variable type. Assign **Date** to the **Column** role.

- Under the Y Axis heading, select **Numeric (default)** as the variable type. Assign **Open** to the **Column** role.
- Assign the **Stock** column to the **Group** role.

5 Run the flow.

Open the **Submitted Code and Results** tab to view the results.

10

Examining Your Data

Characterize Data

About the Characterize Data Step

The Characterize Data step creates a summary report of tables and graphs that describe the variables in the input data set. This step can also create frequency and univariate output tables that describe the main characteristics of the data.

The Characterize Data step is useful when you are working with a new data source. This step enables you to better understand the scope and range of the variables in the data.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirements for Node Connection

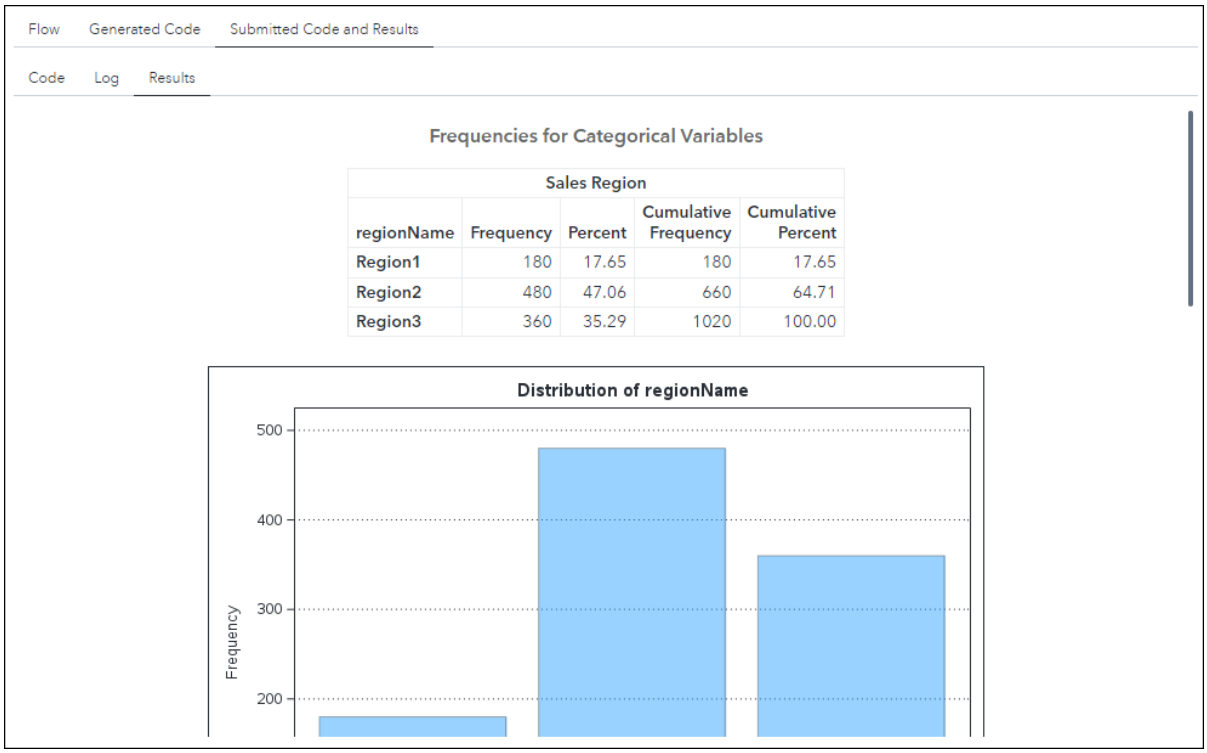
In order to run the Characterize Data step, you must create connections to the input node.

Example: Contents of Sashelp.PriceData

In this example, you want a better understanding of the contents in the Sashelp.PriceData table.

1. In the **Steps** section of the navigation pane, expand the **Examine Data** folder and double-click **Characterize Data**.
2. Add the Sashelp.PriceData table to your flow. Connect the input port of the Characterize Data node to the data source. For more information, see "Connecting Nodes" on page 12.
3. To set the options for the Characterize Data step, click the Characterize Data node.
4. On the **Data** tab, assign columns to these roles:
   - To the **Variables** role, assign **sale**.
   - Expand the Custom Categorization heading.
     - To the **Categorical variables** role, assign **regionName**.
     - To the **Date variables** role, assign **date**.
5. To run the flow, click ▷ Run.

Open the **Submitted Code and Results** tab to view the results. Here is a sample of the results:



Characterize Data: Step-by-Step Instructions

Step 1: Assign Data to Roles

You must select at least one variable to characterize. This step uses automatic characterization to determine the type for your variable. However, you can override this characterization by using the Custom Characterization options.

For example, in the Sashelp.Class table, Age is automatically treated as a numeric variable. You could override this characterization and specify that Age should be treated as a categorical variable. As a result, the step treats each value of Age as a group.

1. (Optional) To filter the input data source, enter the filter expression in the **Filter** field.
2. To the **Variables** role, assign the variables that you want to automatically characterize.
   - Character variables are categorized as categorical variables.
   - If the numeric variable has a date format, it is categorized as a date variable.
   - All other variables are categorized as numeric variables.
3. (Optional) To view the Custom Characterization options, expand the Custom Characterization heading.

- To the **Categorical variables** role, assign the variables to use to produce the frequency tables.
- To the **Date variables** role, assign the date variables to analyze.

4 (Optional) To the **Grouping variable** role, assign any variables to specify how the table is sorted. The step also generates a list for each distinct value, or BY group, in the variable or combination of variables.

Step 2: Set the Options

On the **Options** tab, you must select at least one option for each type of column that you selected on the **Data** tab. If no option is selected for an assigned column type, the step does not run, and you receive an error.

1 Select the options for the categorical variables:

- **Frequency table** displays a frequency table in the results. This table is included by default.
- **Frequency chart** displays a frequency chart in the results.
- **Treat missing values as valid level** treats missing values as a valid nonmissing level for all variables in the table.
- **Limit categorical values** specifies the maximum number of variable levels to display in one-way frequency tables.

2 Select the options for the numeric variables:

- **Descriptive statistics** displays the descriptive statistics for any numeric variables that you assigned to the **Variables** role. This option is selected by default.
- **Histogram** displays a histogram for any numeric variables that you assigned to the **Variables** role. This option is selected by default.

3 Select the options for the date variables:

- **Display minimum and maximum date** shows the minimum and maximum dates for each variable that you assigned to the **Date variables** role. This option is selected by default.
- **Frequency plot** displays a frequency plot in the results.

Describe Missing Data

About the Missing Data Step

The Describe Missing Data step displays the frequencies and percentages of missing values for each selected variable. If two or more variables are assigned to this step, the step displays the pattern of missing data across variables.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

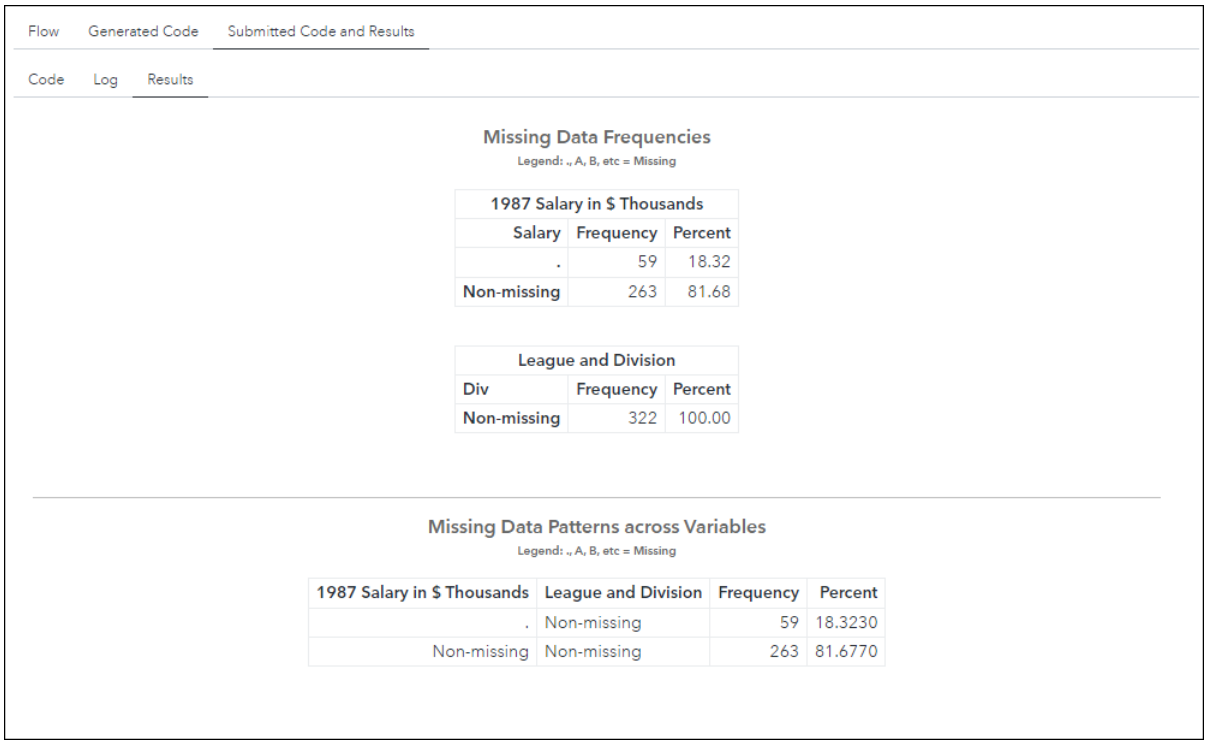
Requirements for Node Connection

In order to run the Describe Missing Data step, you must create connections to the input node.

Example: Describing Missing Data for Sashelp.Baseball

1. In the **Steps** section of the navigation pane, expand the **Examine Data** folder and double-click **Describe Missing Data**.
2. Add the Sashelp.Baseball table to your flow. Connect the input port of the Describe Missing Data node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the Describe Missing Data step, click the Describe Missing Data node.
4. To the **Analysis variables** role, assign **Salary** and **Div**.
5. To run the flow, click ▷ Run.

Open the **Submitted Code and Results** tab to view the results. Here are the results:



Describe Missing Data: Step-by-Step Instructions

1. (Optional) To filter the input data source, enter the filter expression in the **Filter** field.
2. To the **Analysis variables** role, assign the numeric and character variables to use in the analysis.
3. (Optional) To the **Frequncy count** role, assign a variable to specify that each observation in the table is assumed to represent *n* observations, where *n* is the frequency count for that row.
4. (Optional) To the **Group analysis role** role, assign the variables to compute separate statistics for each distinct value or combination of values of the Group analysis by variables.

List Data

About the List Data Step

The List Data step displays the contents of a table as a report. For example, you can use the List Data step to create a report that sums the expenses and revenues for each sales region.

For more information, see "SORT Procedure" in *Base SAS Procedures Guide*.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirements for Node Connection

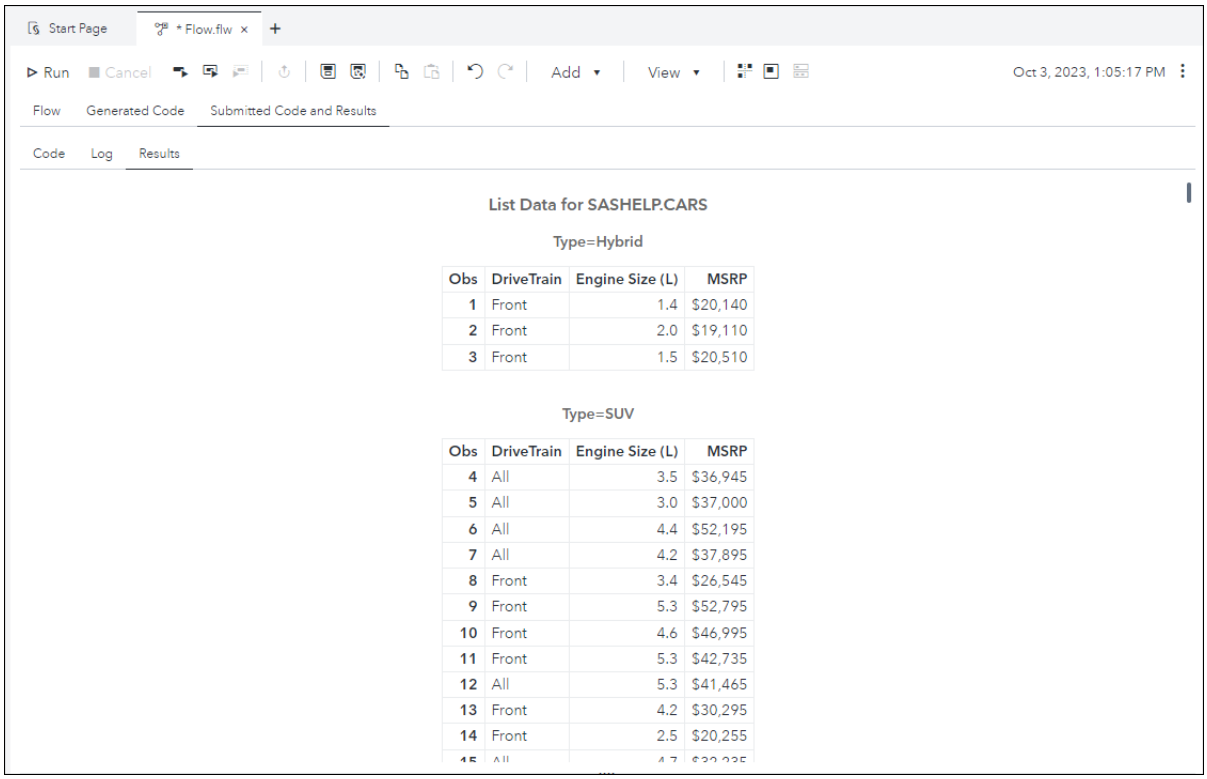
In order to run the List Data step, you must create connections to the input node.

Example: Reports of Drive Train, Engine Size, and MSRP by Car Type

In this example, you want to create reports for each car type. Each report lists the drive train, MSRP, and engine size.

1. In the **Steps** section of the navigation pane, expand the **Examine Data** folder and double-click **List Data**.
2. Add the Sashelp.Cars table to your flow. Connect the input port of the List Data node to the data source. For more information, see "Connecting Nodes" on page 12.
3. To set the options for the List Data step, click the List Data node.
4. On the **Data** tab, assign columns to these roles:
   - To the **List variables** role, assign **DriveTrain**, **EngineSize**, and **MSRP**.
   - To the **Group analysis by** role, assign **Type**.
5. To run the flow, click ▷ Run.

Open the **Submitted Code and Results** tab to view the results. Here is a sample of the results:



List Data: Step-by-Step Instructions

Step 1: Assign Data to Roles

1. (Optional) To filter the input data source, enter the filter expression in the **Filter** field.
2. To the **List variables** role, assign the variables that you want to print. You can specify the order of the variables.
3. (Optional) To obtain separate analyses of observations for each unique group, assign a variable to the **Group analysis by** role.
4. (Optional) To print the sum of a selected variable at the bottom of the listing report, assign the variable to the **Total of** role.
5. (Optional) To use the formatted values of specific variables to identify the rows (rather than the observation numbers), assign one or more variables to the **Identifying label** role.

Step 2: Set the Options

On the **Options** tab, you can set these options:

1. To include in the output a column that lists the row number for each observation, select **Display row numbers**. The row numbers are included by default.

   You can specify a label for this column in the **Column label** text box.

2. Specify whether to use the column labels as column headings. The column labels are used by default.

3. To show the number of rows in the table at the end of the output or the number of rows in each BY group at the end of each BY group's output, select **Display number of rows**.

4. To round each numeric value to the number of decimal places in its format or to two decimal places if no format is specified, select **Round values before summing the variable**. If this option is selected, the List Data step performs the rounding before summing the variable.

5. Specify the direction of the column heading. Column headings can be printed horizontally or vertically, or you can select **Default** and let SAS determine the optimal arrangement for each column.

6. Specify the column widths for the output. You can choose from these options:

   - **Default** determines the column widths on a per-page basis.
   - **Full** uses a format width (or default width if no format is specified) for all pages.
   - **Minimum** uses the smallest possible column width on a per-page basis.
   - **Uniform** reads the entire table to determine the appropriate column widths before generating output. When this option is not selected, different pages could have different widths for the same column.
   - **Uniform by** formats all columns uniformly within a BY group by using each variable's formatted width as its column width. If the variable does not have a format that explicitly specifies a field width, the task uses the widest data value as the column width.

7. Specify whether to split the variable labels.

   If the variable labels contain one of the split characters (*, !, @, #, $, %, ^, &, or +), the labels split at the specified character or characters. For example, for a variable label that reads "This is*a label" and the * character is selected as the split character, the column heading reads

   ```
   This is
   a label
   ```

   You do not need to select both the **Use variable label as column headings** and **Split labels** options. The **Split labels** option implies that you want to use variable labels.

8. To specify the number of rows to list in the output, use the **Rows to list** option. By default, all rows are listed.

List Table Attributes

About the List Table Attributes Step

The List Table Attributes step enables you to quickly see the date on which the table was created and last modified, the number of rows, the encoding, any engine-dependent or host-dependent information, and an alphabetical list of the variables and their attributes. You can also view any directory and host or engine information by using this step.

Note: This step is available only if your site licenses SAS Studio Analyst or SAS Studio Engineer.

Requirements for Node Connection

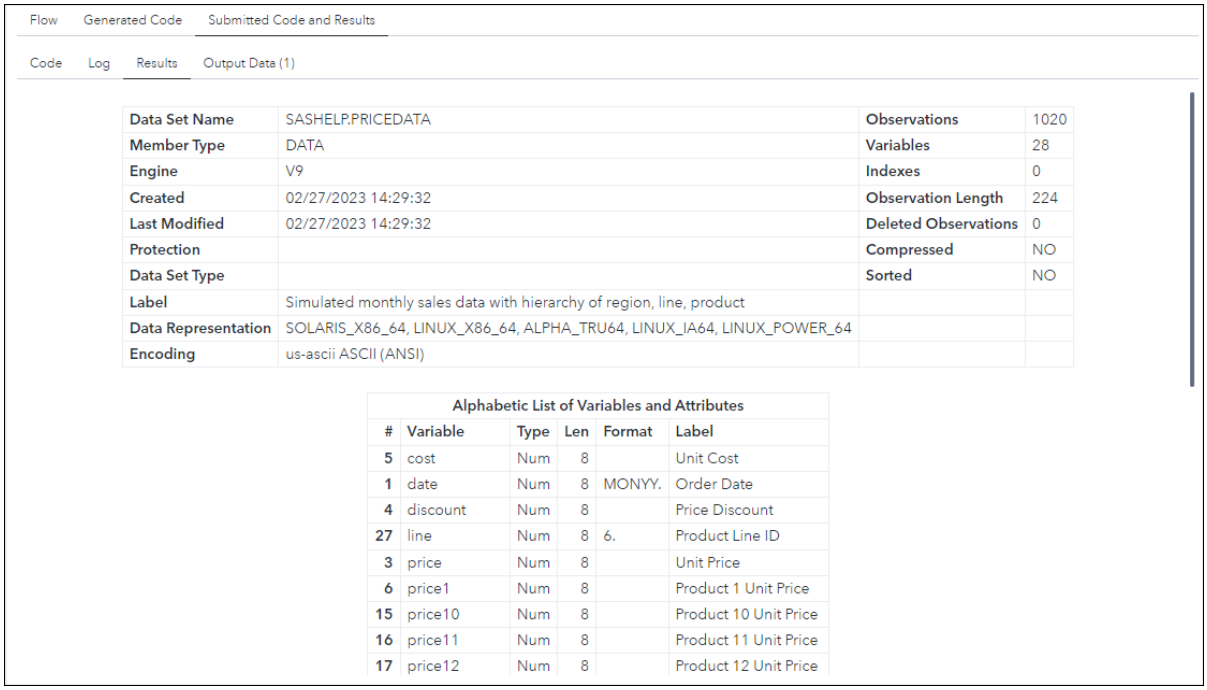
In order to run the List Table Attributes step, you must create connections to the input and output ports of the node as indicated here:

Input Port	Output Port
■ Table node	No connection is required.

Example: Table Attributes for Sashelp.PriceData

1. In the **Steps** section of the navigation pane, expand the **Examine Data** folder and double-click **List Table Attributes**.
2. Add the Sashelp.PriceData table to your flow. Connect the input port of the List Table Attributes node to the data source. For more information, see “Connecting Nodes” on page 12.
3. To set the options for the List Table Attributes step, click the List Table Attributes node.
4. On the **Output** tab, select **Create output data**.
5. To run the flow, click ▷ Run.

Open the **Submitted Code and Results** tab to view the results. Here is a sample of the results:



Flow | Generated Code | Submitted Code and Results

Code | Log | Results | Output Data (1)

Data Set Name	SASHELP.PRICEDATA	**Observations**	1020
Member Type	DATA	**Variables**	28
Engine	V9	**Indexes**	0
Created	02/27/2023 14:29:32	**Observation Length**	224
Last Modified	02/27/2023 14:29:32	**Deleted Observations**	0
Protection		**Compressed**	NO
Data Set Type		**Sorted**	NO
Label	Simulated monthly sales data with hierarchy of region, line, product		
Data Representation	SOLARIS_X86_64, LINUX_X86_64, ALPHA_TRU64, LINUX_IA64, LINUX_POWER_64		
Encoding	us-ascii ASCII (ANSI)		

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Label
5	cost	Num	8		Unit Cost
1	date	Num	8	MONYY.	Order Date
4	discount	Num	8		Price Discount
27	line	Num	8	6.	Product Line ID
3	price	Num	8		Unit Price
6	price1	Num	8		Product 1 Unit Price
15	price10	Num	8		Product 10 Unit Price
16	price11	Num	8		Product 11 Unit Price
17	price12	Num	8		Product 12 Unit Price

List Table Attributes: Step-by-Step Instructions

Step 1: Select the Information to Display

On the **Options** tab, you can set these options:

1. (Optional) To view attributes of the table (such as the table name, member type, when the table was created, when the table was last modified, encoding, and so on), select **Data set attributes**. The table attributes are included by default.
2. (Optional) To view the list of all the variables and their attributes (such as variable name, type, length, and so on), select **Variables list**. The variables are listed in the results by default.
3. Specify the order of the variables in the output. You can choose to display the variables in alphabetical order or in the order in which they appear in the table.
4. (Optional) To view the name of the directory where this table is located, select **Directory information**.
5. (Optional) To display the SAS engine, physical name, and file name for each level in the directory, select **Host/Engine information**.

Step 2: Set the Output Options

On the **Output** tab, you can set these options:

1. To save the table attributes in an output table, select **Create output data**. You can specify whether to replace an existing table.
2. To print the output table as part of the results, select **Print output data**.

11

Managing Models

Register Python Model

About the Register Python Model Step

Using this step, you can register into SAS Model Manager your Python models that have been trained in SAS Studio. Then you can use the functionality in SAS Model Manager to manage and govern your trained models.This step is available only for classification models with a binary target.

Note: The Register Python Model step is available in SAS Studio Analyst and SAS Studio Engineer.

Step-by-Step: Register a Python Model

Step 1: Register a Model

1 In the **Steps** section of the navigation pane, expand the **Manage Models** folder and double-click **Register Python Model**.

2 Click the Register Python Model node.

3 Enter a name and description for the model.

4 Select the file that contains the Python training code.

Note: This file must be specified in either the previous node or within the current session.

5 Specify the name of the trained model.

6 Specify the name of the project.

7 Specify whether to register the model in a **New project version**.

8 To run the flow, click ▷ Run.

9 Open SAS Model Manager. Refresh your display to view the model in your project.

Step 2: Select the Variables

On the **Variables** tab, specify these options.

1 Specify the model function. The binary target level is automatically detected for a model.

2 Select the target variable.

3 Select the target values.

4 Select the value for the target event.

5 Select any input variables.

6 Select the output variable.

7 Select the variable whose values contain the output event probability.

Register SAS Model

About the Register SAS Model Step

This step is used with SAS Model Manager.

The Register SAS Model step imports a scoring model from SAS Studio into SAS Model Manager. This step accepts both scoring models that are analytical objects in a CAS table (ASTORE) as well as score code, which is a file containing DATA step code.

For more information, see these examples:

- "Import an Analytic Store File into a Project" in *SAS Model Manager: Macro Reference*
- "Import a Model Using a DATA Step Score Code File" in *SAS Model Manager: Macro Reference*

Note: The Register SAS Model step is available in SAS Studio Analyst and SAS Studio Engineer.

Register a Model

Note: A prerequisite for this is task is an existing SAS Model Manager project. If no project exists, you must create one in SAS Model Manager.

1. In the **Steps** section of the navigation pane, expand the **Manage Models** folder and double-click **Register SAS Model**.
2. Click the **Register SAS Model** node.
3. Enter a name and description for the model.
4. Select the model type: **Analytic store** or **DATA step**.
5. Under the **Model Manager** heading, enter the name of the project that you created in SAS Model Manager.
6. Specify the location of the project in SAS Model Manager. The default model repositories are `Public` and `DMRepository`. For more information, see "About Model Repositories" in *SAS Model Manager: User's Guide*.

   Note: White space and special characters are not allowed in the model name.

7 (Optional) To create a new project with an incremental number, select **New project version**.

8 To run the flow, click ▷ Run.

9 Open SAS Model Manager. Refresh your display to view the model in your project.

12

Enhancing Data Quality

Match Codes

About the Match Codes Step

Use the Match Codes step to create match codes that can be used as a basis for standardization or other transformations. Using the Match Codes step, you can create a match code for a column based on a locale and rule definition. (SAS provides these locale and rule definitions in the SASDqRef.DM_Madef data table.)

The amount of information that is contained in match codes is determined by a specified sensitivity level. Changing the sensitivity level enables you to change what is considered a match. Match codes that are created at lower levels of sensitivity capture little information about the input values. The result is more matches, fewer clusters, and more values in each cluster.

Higher sensitivity levels require that input values be more similar to receive the same match code. Clusters are more numerous, and each cluster contains fewer entries. For example, when collecting customer data that is based on account numbers, cluster on account numbers with a high sensitivity value.

For example, your input data source contains these three values in a Name column: James Briggs, Jim Briggs, and Mr. James E. Briggs. The generated match code is based on the English (US) locale and the Name definition. The Match Codes step saves the generated match code in a new column in the output data table. In this example, the Match Codes step generates an identical match code for the three name values. You can now use the match code to help you identify duplicate rows in your input data source.

You can generate up to 10 match codes for an input table.

Match Codes: Step-by-Step Instructions for Matching Code

Step 1: Add the Match Codes Step and Connect a Source Table

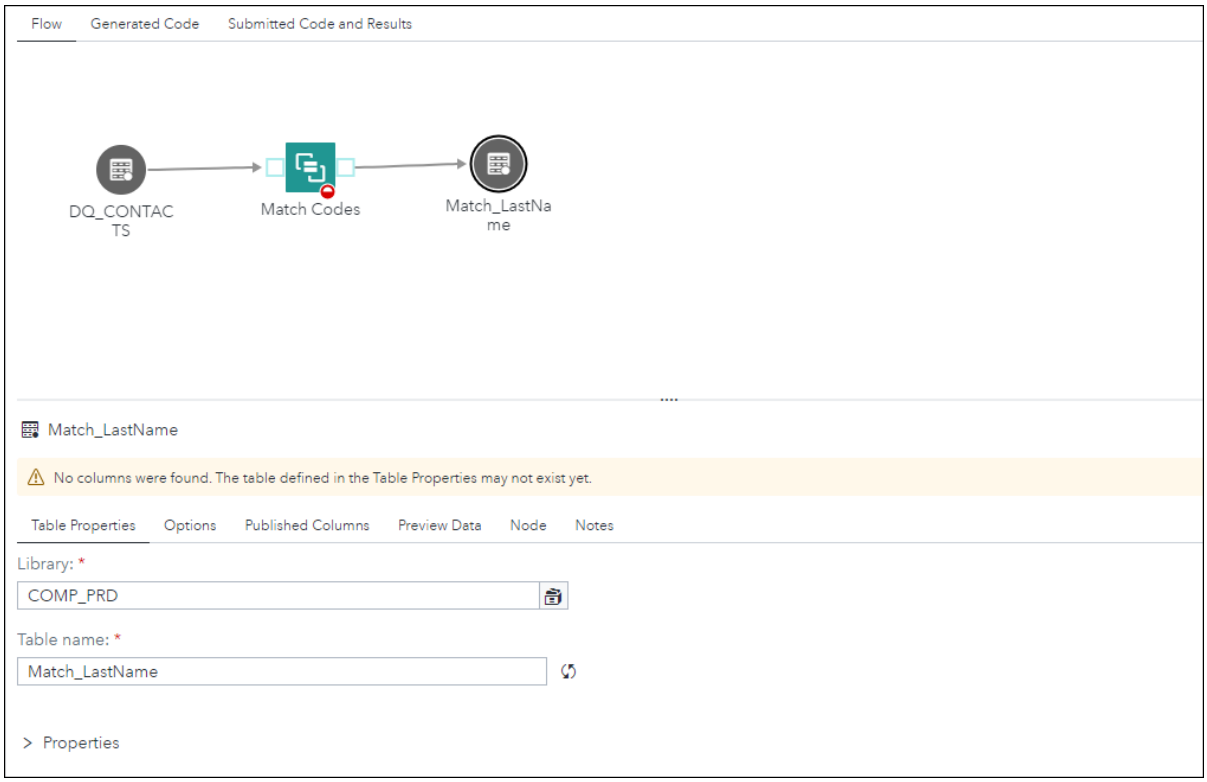
1 In the **Steps** section of the navigation pane, expand the **Data Quality** folder and double-click **Match Codes** to add the step to your flow.

2 Connect the input port of the Match Codes node to a data source by using a Table node or another node that creates an output table, such as a Query node, a SAS Program node, or an Import node. For more information, see “Connecting Nodes” on page 12.

3 To save the results from the Match Codes step to a permanent output table, select **Add** ⇨ **Table**. A table node is added to the flow. Specify the library and name for the output table. (In this example, the output table is named DQ.Match_Name.)

Next, connect the output port of the Match Codes node to the table node.

Note: The output table does not have any columns until you run the flow.



Step 2: Generating Match Codes

1 Select the QKB locale.

2 Select a column from the input data source.

3 (Optional) Specify a name for the column that contains the match code. Sensitivities are available only increments of five with a range between 50-95. The default value is 85.

   By default, this column is the *InputColumnName*_MC*Sensitivity-suffix*. An example of a column name is ColumnName_MC85.

4 Select the definition to use for the match code. These definitions are in the SASDqRef.DM_Madef data table.

5 (Optional) Create additional match codes. You can generate up to 10 match codes for an input table.

Step 3: Set the Options

On the **Options** tab, select **Show detailed log information** to include debugging information in the log.

Step 4: Specify the Output Data

Click the **Output** tab to set these options.

1 (Optional) Select **Replace existing output table with the same name** to replace an existing table with the new table. Any output tables are replaced by default.

2 (Optional) Select **Perserve missing values** to show missing values in the results.

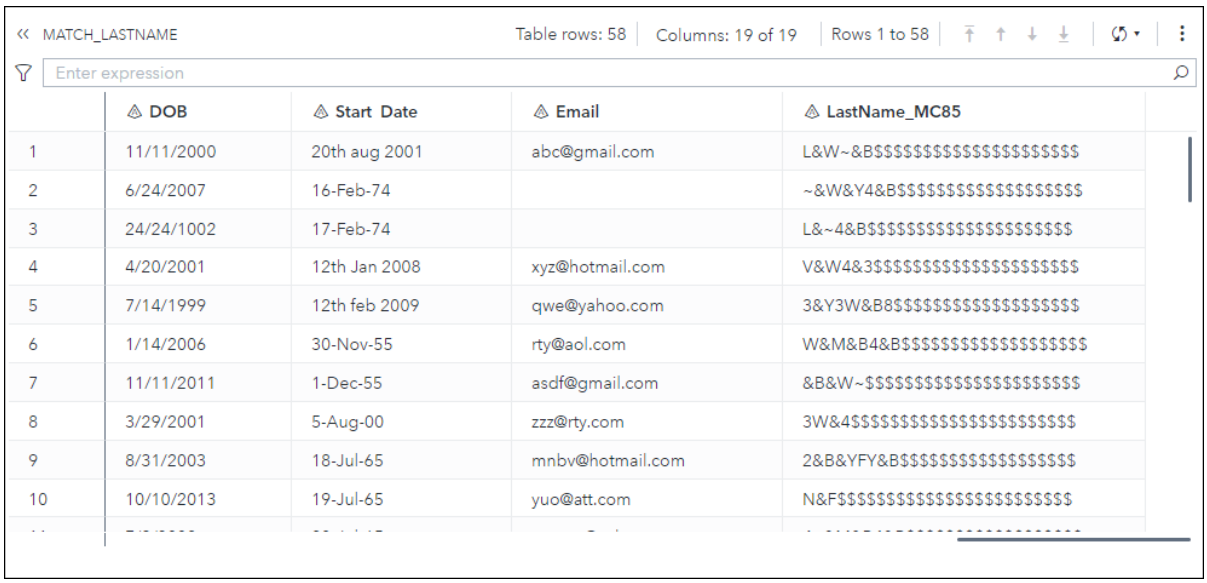
3 To create a CAS table from the output data, select these options:

- **Promote to a global in-memory table**
- **Save to persist the table on disk**
- **File format**

Step 5: Run the Flow and View the Output Table

To run the flow, click ▷ Run.

To view the output data, click the table node, and then click the **Preview Data** tab. In this example, the LastName_MC85 column contains the generated match code for the values in the Last Name column.



	DOB	Start Date	Email	LastName_MC85
1	11/11/2000	20th aug 2001	abc@gmail.com	L&W~&B$$$$$$$$$$$$$$$$$$$$
2	6/24/2007	16-Feb-74		~&W&Y4&B$$$$$$$$$$$$$$$$$$
3	24/24/1002	17-Feb-74		L&~4&B$$$$$$$$$$$$$$$$$$$$
4	4/20/2001	12th Jan 2008	xyz@hotmail.com	V&W4&3$$$$$$$$$$$$$$$$$$$$
5	7/14/1999	12th feb 2009	qwe@yahoo.com	3&Y3W&B8$$$$$$$$$$$$$$$$$$
6	1/14/2006	30-Nov-55	rty@aol.com	W&M&B4&B$$$$$$$$$$$$$$$$$$
7	11/11/2011	1-Dec-55	asdf@gmail.com	&B&W~$$$$$$$$$$$$$$$$$$$$
8	3/29/2001	5-Aug-00	zzz@rty.com	3W&4$$$$$$$$$$$$$$$$$$$$
9	8/31/2003	18-Jul-65	mnbv@hotmail.com	2&B&YFY&B$$$$$$$$$$$$$$$$$
10	10/10/2013	19-Jul-65	yuo@att.com	N&F$$$$$$$$$$$$$$$$$$$$$

13

Optimizing and Running a Flow

Optimizing Steps in a Flow

When you use the output of an operational node as the input to another node, it might not be necessary to create and store the output table from the first node. You can optimize the performance of your flow by combining the code generation of adjacent nodes so that the table that is associated with the output port of the first node is not created.

The following step combinations can be optimized:

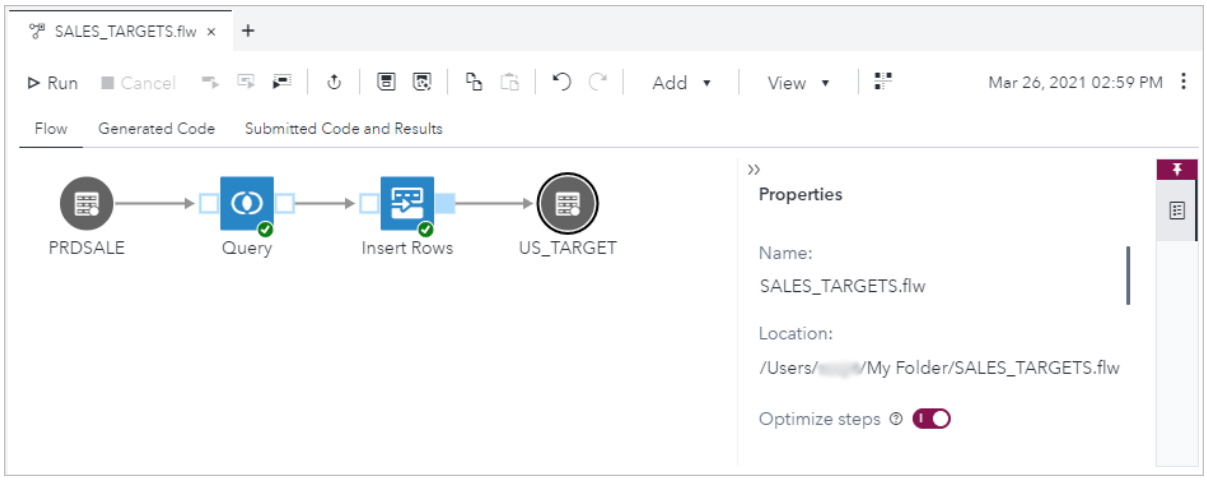
- Query node connected to an Insert Rows node
- Filter Rows node connected to a Sort node

Note: The optimization option works only when the nodes that can be optimized are run at the same time.

For example, you could use the Query step to calculate local sales data that you want to append regularly to a regional sales table by using the Insert Rows step. You can optimize the performance of your flow by combining the code that is generated by the Query and Insert Rows nodes. When the code for the Query and Insert Rows nodes is combined, the output table for the Query node is not explicitly created.

To optimize the performance of your flow:

1. On the flow canvas, expand the Properties pane by clicking .
2. Click to turn on the **Optimize steps** option.

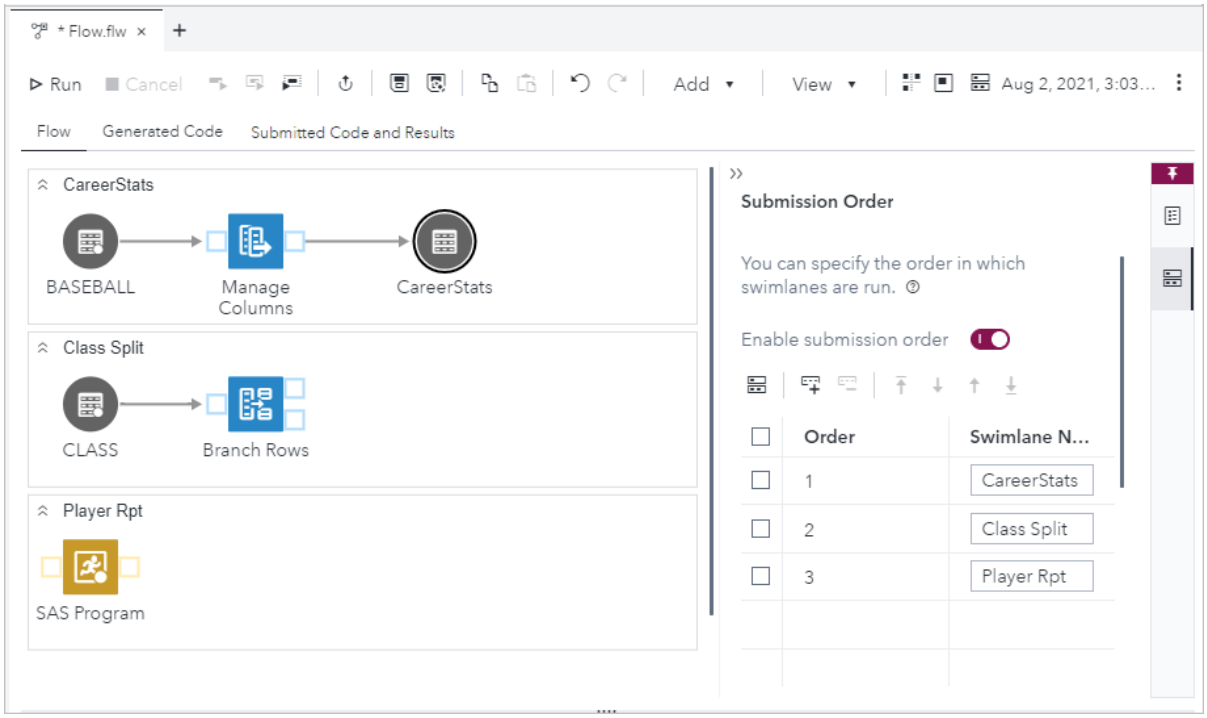


Controlling the Submission Order of a Flow

By default, the order of the nodes on the flow canvas determines the order in which the nodes run. You can control the submission order by grouping connected nodes into swimlanes, and then specifying the order of the swimlanes.

To specify the submission order:

1. On the flow canvas, expand the Submission Order pane by clicking on the right side of the canvas.
2. Click to turn on the **Enable submission order** option. Each group of connected nodes is organized into a separate swimlane.



3 You can perform the following actions in the Submission Order pane:

- To change the order in which the swimlanes are run, use the buttons on the toolbar to move the swimlanes up and down the list. You can also drag swimlanes to the flow canvas.
- To move one or more nodes from one swimlane to another, select the nodes and drag them to the appropriate swimlane. You can also cut, copy, and paste nodes between swimlanes.
- To add a new swimlane to the flow, click on the Submission Order toolbar.
- To delete a swimlane and the associated nodes, select one or more swimlanes from the list of swimlanes and click on the Submission Order toolbar.
- To rename a swimlane, enter the new name in the Swimlane name column. The name is updated on the flow canvas.
- To refresh the arrangement of the nodes within swimlanes and remove empty swimlanes, click on the toolbar.

Note: When you disable the submission order feature, any changes that you have made to swimlane names and order are not saved.

Running a Flow

To run all the nodes in a flow, click ▷ Run.

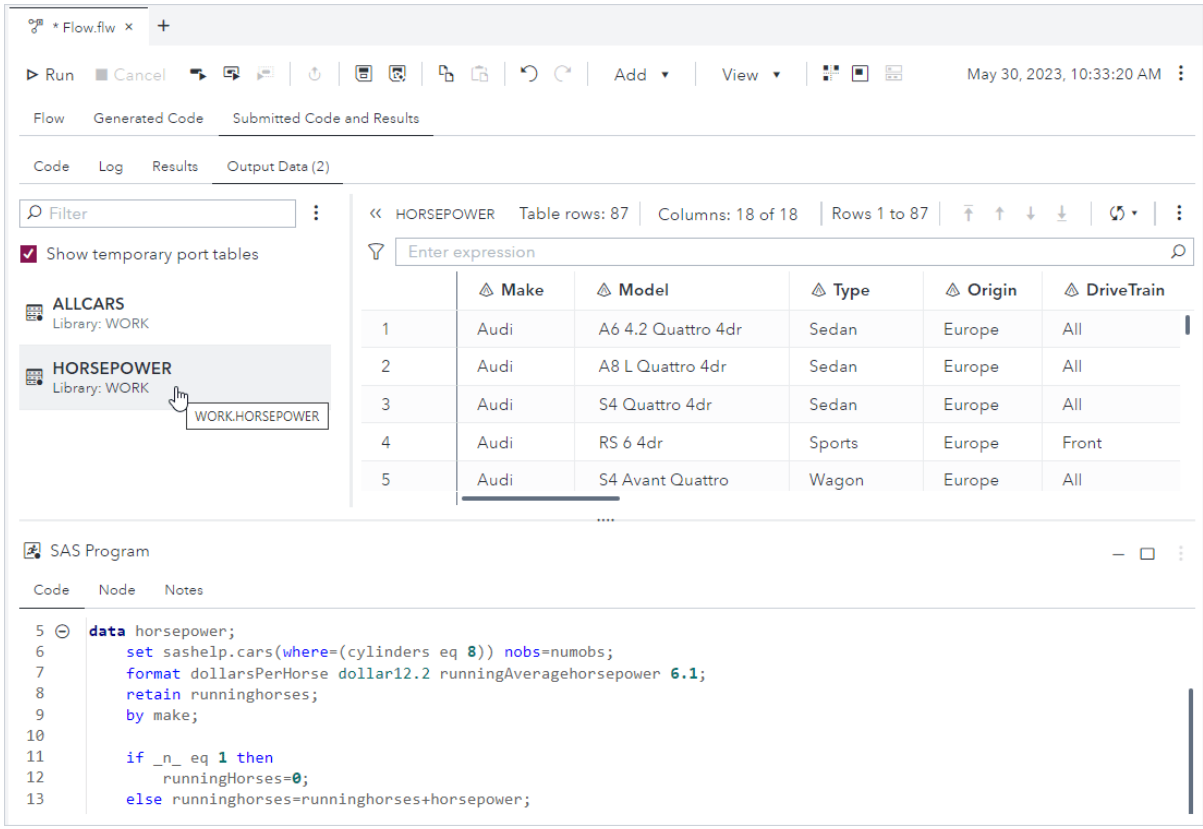
You can also run a subset of the flow by using the following buttons on the flow toolbar:

- - runs only the currently selected node in the flow.
- - runs the currently selected node and all nodes that occur downstream from the currently selected node.
- - runs all nodes that occur upstream from the currently selected node. The currently selected node is not run.

Note: To cancel your flow, click **Cancel** on the toolbar. Currently processing nodes might take a few minutes to be canceled.

You can view the automatically generated code and log for a flow by clicking the **Generated Code** and **Submitted Code and Results** tabs on the flow canvas. The **Submitted Code and Results** tab also displays any results and output data that are generated when you run a node. If more than one output table is generated, you can use the output list to select which table to display.

TIP You can interact with a flow while it is running by scrolling the flow canvas and by clicking nodes and tabs on the flow canvas. You cannot modify the flow or any node details until the flow is finished running.



To view the code or log that is associated with a particular node, right-click the node and select **Go to last submitted code** or **Go to last submitted log**.

To run a saved flow as a background job, open the flow and click on the toolbar. For more information, see "Using the Background Submit Feature" in *SAS Studio: User's Guide*.

or right-click the flow in the **Explorer** section of the navigation pane and select **Background submit**.

14

Creating a Job from a Flow

Creating a Job from a Flow

To create a job from a saved flow, open the Explorer pane. Navigate to the location of the saved FLW file. Right-click the name and select **Create job**. The new job opens as a job definition in the workspace. For more information, see "Create a New Job Definition" in *SAS Studio Developer's Guide: Working with Jobs*.

To deploy a flow as a job, open the Explorer pane. Navigate to the location of the saved FLW file. Right-click the name and select **Deploy as a job**. For more information, see "Managing Deployed and Scheduled Jobs" in *SAS Studio Developer's Guide: Working with Jobs*.

You can also schedule a job from the Explorer pane. Right-click the file name of the flow and select **Schedule a job**. For more information, see "Schedule a Job" in *SAS Studio Developer's Guide: Working with Jobs*.

15

SAS Information Catalog and SAS Lineage Viewer Integration

SAS Information Catalog and SAS Lineage Viewer Integration

SAS Studio flows are automatically indexed in SAS Information Catalog and SAS Lineage Viewer. SAS Information Catalog enables you to capture and enrich metadata for files, tables, and other information assets. This metadata is stored in a catalog. You can search the catalog to find SAS Studio flows and other assets that you need to meet your business goals.

To access SAS Information Catalog, select **Discover Information Assets** from the Applications menu in the top left of the application window. For more information, see SAS Information Catalog: User's Guide.

SAS Lineage Viewer enables you to better understand the relationships between objects in your applications on the SAS Viya platform. These objects include SAS Studio flows, data, transformation processes, reports, and visualizations.

To access SAS Lineage Viewer, select **Explore Lineage** from the Applications menu in the top left of the application window. For more information, see SAS Lineage: User's Guide.