# Contents

*About This Book* .................................................. v  
*What’s New in SAS Studio 5.2* .................................. vii  

**Chapter 1 / Introduction to SAS Studio** .......................... 1  
   About SAS Studio ................................................. 1  
   Using SAS Studio .................................................. 3  

**Chapter 2 / Working with Programs** .............................. 21  
   About the Code Editor .......................................... 21  
   Opening and Creating Programs .................................. 22  
   Using the DATA Step Debugger ................................... 36  
   Working with Snippets ............................................ 44  
   Customizing the Code Editor .................................... 56  

**Chapter 3 / Working with Queries** ............................... 57  
   What Is a Query? .................................................. 57  
   Creating a Query .................................................. 58  
   Understanding Joins .............................................. 59  
   Selecting Data ................................................... 63  
   Filtering Data ..................................................... 70  
   Managing Output .................................................. 72  
   Generating a FedSQL Query ..................................... 79  

**Chapter 4 / Working with Data** ................................ 83  
   About the Table Viewer .......................................... 83  
   Opening and Viewing Data ....................................... 85  
   Viewing the Code That Is Used to Create a Table ............... 89  
   Opening Data in a Task from the Table Viewer .................. 89  
   Refreshing Your Data ............................................. 91  
   Sorting and Freezing Data ...................................... 91  
   Filtering Data .................................................... 92  
   Importing Data ................................................... 95  
   Exporting Data ................................................... 101  

**Chapter 5 / Working with Results** .............................. 103  
   Viewing Results .................................................. 103  
   Default SAS Studio Output ...................................... 103  
   Sending Your Results to Another User ............................ 105  
   About the SAS Output Delivery System and SAS ODS Statistical Graphics ........................................ 106  

**Chapter 6 / Understanding Git Integration in SAS Studio** ...... 109  
   About Git Integration in SAS Studio ............................ 109  
   Working with Git Profiles ....................................... 110  
   Cloning and Opening a Git Repository ........................... 112  
   Creating a Local Repository .................................... 114  
   Understanding the Git Repository Tab in SAS Studio ........... 115  
   Viewing the Commit History .................................... 116
Appendix 4 / Customized Output Environment .................................................. 161
  Overview ............................................................................................................... 161
  Generate Output for Other Output Destinations ............................................... 162
  Send Your Results to Another Location ........................................................... 162
  Use a Custom Style for Your Output .................................................................. 163
  Use an Image Format Other Than the Default .................................................. 163
  Create a Drill-down Graph .................................................................................. 164
  Create an Animated GIF or SVG Image ............................................................... 164

Appendix 5 / SAS Studio Command Line ............................................................... 165
  About the Command Line Interface .................................................................... 165
  Commands in Standard Perspective .................................................................... 166
  Commands in Interactive Perspective ................................................................. 171
  Specifying the Path to the Content: SAS Content or SAS Compute ................. 175

Chapter 7 / Understanding Tasks in SAS Studio ................................................ 137
  What Is a Task? ..................................................................................................... 137
  How to Run a Task ............................................................................................... 137
  Edit a Predefined Task ....................................................................................... 138
  Create a New Task .............................................................................................. 139
  Specifying the Options for the Task Code ........................................................ 140

Appendix 1 / Customizing SAS Studio ................................................................. 141
  About Customizing SAS Studio .......................................................................... 141
  Creating Global Settings ..................................................................................... 142
  Setting Your Preferences ..................................................................................... 145

Appendix 2 / Using the Expression Builder .......................................................... 157
  Building an Expression ....................................................................................... 157

Appendix 3 / Text Encoding Options and Language Mappings ........................... 159
  About the Text Encoding to Language Mappings ............................................. 159
  Text Encoding Options and Language Mappings .............................................. 159
About This Book

Audience

This book is designed for all users of SAS Studio. SAS Studio was initially released with the first maintenance release for SAS 9.4. SAS Studio 5.2 is the latest release.
vi About This Book
What’s New in SAS Studio 5.2

Overview

SAS Studio 5.2 includes these new features and enhancements:

- ability to create a query. For more information, see “Query Interface” on page viii.
- Git integration user interface. For more information, see “SAS Studio Git Integration” on page viii.
- Import data functionality. For more information, see “Import Data” on page viii.
- ability to access files and folders from both the SAS Content server and your server file system. For more information, see “Enhanced File System Navigation” on page viii.
- DATA Step Debugger. For more information, see “DATA Step Debugger” on page viii.
- ability to create jobs. For more information, see “SAS Studio Jobs” on page ix.
- ability to run a saved program as a background job. For more information, see “Background Job Submissions” on page ix.
- new command line interface. For more information, see “Command Line Interface” on page ix.
- ability to create a quick filter on your data. For more information, see “Quick Filter” on page ix.
- updated macro variable name. For more information, see “Updated Macro Variable Name” on page x.
- ability to share SAS Studio content. For more information, see “Share SAS Studio Content” on page x.
- ability to insert custom code. For more information, see “Insert Custom Code” on page x.
- ability to manage keyboard shortcuts. For more information, see “Managing Keyboard Shortcuts” on page x.
- several new preferences. For more information, see “New Preferences” on page xi.
- several new and enhanced tasks. For more information, see “SAS Studio Tasks” on page xii.
- updates to the common task model. For more information, see “Enhancements to the Common Task Model” on page xv.
Query Interface

You can create a query to extract data from one or more tables according to criteria that you specify. You can generate your query using either SQL or FedSQL query code. For more information, see “Creating a Query” on page 58.

SAS Studio Git Integration

Git is a version control system for tracking changes in files and coordinating work on those files among multiple people. You can use the user interface that integrates some of the most common Git features into SAS Studio. You can clone repositories, stage changes and create commits, create, merge, and rebase branches, and resolve merge conflicts from within SAS Studio. For more information, see “About Git Integration in SAS Studio” on page 109.

Note: The SAS Studio documentation assumes that you are already familiar with Git.

Import Data

Using SAS Studio, you can import several basic file types into data sets. For more information, see “Importing Data” on page 95.

Enhanced File System Navigation

You can now access files and folders on both the SAS Content server and your server file system from the Explorer section in the navigation pane. For more information, see “Using the Explorer” on page 4.

DATA Step Debugger

You can use the new DATA Step Debugger to find logic errors in a DATA step program. With the DATA Step Debugger, you can watch the variable values in a
program change as the program runs. You can execute the program line by line, and you can also set specific breakpoints in the program. For more information, see “Using the DATA Step Debugger” on page 36.

---

**SAS Studio Jobs**

You can use jobs for web reporting, performing analytics, building web applications, and delivering content to clients such as SAS Studio. A SAS Viya job consists of a program and its definition. The job definition includes information such as the job name, the author, and the creation date and time. After you have created a job definition, you have an execution URL that you can share with others at your site. The execution URL can be entered into a web browser and run without opening SAS Studio.

You can create an HTML form or task prompt to provide a user interface to the job. When the user selects an option to submit the information, the data that is specified in the form or task prompt is passed to a SAS session as global macro variables. The SAS program runs and the results are returned to the web browser.

For more information, see *SAS Studio Developer’s Guide: Working with Jobs*

---

**Background Job Submissions**

You can run a saved program as a background job, which means that the program can run while you continue to use SAS Studio or after you sign out of SAS Studio. You can specify the location of the log and output files and the action to take if a log or output file already exists. For more information, see “Using the Background Submit Feature” on page 25.

---

**Command Line Interface**

You can use the command line interface to access SAS Studio by using the keyboard. A command can have options and arguments. Commands are executed in order and the next command starts when the previous command completes. For more information, see Appendix 5, “SAS Studio Command Line,” on page 165.

---

**Quick Filter**

You can create a filter based on a single column by right-clicking the column heading in the table viewer and selecting **Quick Filter**. Specify the operator and
values that you want to use. For more information, see “Creating a Quick Filter” on page 93.

Updated Macro Variable Name

The CLIENTAPPABREV macro variable has been renamed to CLIENTAPPABBREV. If you have any saved programs that use this macro variable, you need to update the name.

Share SAS Studio Content

You can send a copy of your results and the associated code and log files to another user through email. Files that you can send include results in HTML5, RTF, and PDF formats as well as the code and log files that are associated with the results. For more information, see “Sending Your Results to Another User” on page 105.

Insert Custom Code

You can now specify custom SAS code to run before or after the code for programs, tasks, queries, and imports. This code persists between SAS Studio sessions. If you change the custom code during a SAS Studio session, the updated custom code (not any previously saved custom code) is run.

Managing Keyboard Shortcuts

SAS Studio includes many keyboard shortcuts that you can use to perform a task. Some keyboard shortcuts can be customized. To manage your keyboard shortcuts, select Options ⇒ Manage keyboard shortcuts. When you enter a new keyboard shortcut, you are prompted to confirm your changes.

To improve accessibility, these new keyboard features were added:

- To open the Open window, press Ctrl+O in Microsoft Windows or Mac OS X.
- To move the active tab with focus to the previous position, press Ctrl+Shift+left arrow in Microsoft Windows or Mac OS X.
- To move the active tab with focus to the next position, press Ctrl+Shift+right arrow in Microsoft Windows or Mac OS X.
To open or close the Submission Status window, press Alt+U in Microsoft Windows or Control+U on Mac OS X.

To recall in the Interactive Perspective, press Alt+R in Microsoft Windows or Control+R on Mac OS X.

Shortcuts are available in the DATA Step Debugger.

Keyboard shortcuts are available for the new command line interface.

For more information, see "Managing Keyboard Shortcuts" on page 19.

New Preferences

SAS Studio includes many new preferences that you can use to customize the application. To view the preferences, select Options ➔ Preferences.

A new Start Up preference enables you to reopen folders that were expanded in the Explorer section in the navigation pane. For more information, see "Setting the Start Up Preferences" on page 145.

New General preferences enable you to determine which output data tabs to open and which command options to display when you use the command line. For more information, see "Setting General Preferences" on page 146.

New Code and Log preferences enable you to include custom code in the log, specify the arrangement of the program tab layout, and specify which tabs to include in the preview section of the program tab. For more information, see "Setting the Code and Log Preferences" on page 147.

A new Results preference enables you to determine whether to generate HTML output. For more information, see "Setting the Results Preferences" on page 148.

New General Code Editor preferences enable you to automatically insert matching brackets and other paired characters as well as display invisible characters. You can also display a line-length guide. For more information, see "Setting General Code Editor Preferences" on page 151.

A new Code Editor Appearance preference enables you to change the background color of the editor. A new dark syntax theme is also now available. For more information, see "Setting Code Editor Appearance Preferences" on page 152.

A new Tables preference enables you to display up to 1,000 rows and columns per page. The UPCASE variable name policy is now supported as well. For more information, see "Setting Tables Preferences" on page 153.

A new Background Submit preference enables you to cancel the background job submission when there are log and output files of the same name. For more information, see "Setting Background Submit Preferences" on page 155.

New Query preferences enable you to specify the columns that are displayed on the Select tab and determine whether to display the preview section of the query tab. You can also use PROC FEDSQL to generate the query code. For more information, see "Setting Query Preferences" on page 156.
A new Import preference enables you to determine whether to display the preview section of the import tab. For more information, see “Setting Import Preferences” on page 156.

SAS Studio Tasks

The tasks that are shipped with SAS Studio are documented in SAS Studio: Task Reference Guide. All of the tasks that are available with SAS Studio are documented. However, the tasks that are available at your site depend on whether you license and install other SAS products.

Many of the tasks were reorganized for this release. If you cannot find a task in the user interface, check the documentation for that task.

New Tasks

SAS Viya: Cloud Analytic Services

- Use the new Connect to CAS task to establish a connection to an existing CAS server. For more information, see “SAS Viya: Connect to CAS” in SAS Studio: Task Reference Guide.

- The new CAS Operations task enables users to perform operations to manage CAS sessions, data stores, data, servers, and formats. For more information, see “SAS Viya: CAS Operations” in SAS Studio: Task Reference Guide.

SAS Viya: Machine Learning

- The Automated Feature Engineering task explores the variables in the input data automatically and performs feature engineering in a parallel, scalable way. For more information, see “SAS Viya: Automated Feature Engineering” in SAS Studio: Task Reference Guide.

- The Fast k-Nearest Neighbors task searches for the k-nearest neighbors (KNN) of the specified data. For more information, see “SAS Viya: Fast k-Nearest Neighbor” in SAS Studio: Task Reference Guide.


- The Semi-supervised Learning task iteratively propagates the labels from the labeled data to the unlabeled data by computing the similarity measure between pairs of data. For more information, see “SAS Viya: Semi-supervised Learning” in SAS Studio: Task Reference Guide.

- The Load Image task loads images from a path into a CAS table. For more information, see “SAS Viya: Load Images” in SAS Studio: Task Reference Guide.
SAS Viya: Forecasting > Time Series Mining

- The Motif Discovery task provides methods to discover frequent patterns or repeated subsequences in time series data. For more information, see “SAS Viya: Motif Discovery” in SAS Studio: Task Reference Guide.

- The Motif Scoring task tries to find subsequences that are most similar to the target sequence in a new time series. For more information, see “SAS Viya: Motif Scoring” in SAS Studio: Task Reference Guide.

- The Subsequence Anomaly Detection task is a motif-based technique that finds anomaly subsequences in a specified input sequence. For more information, see “SAS Viya: Subsequence Anomaly Detection” in SAS Studio: Task Reference Guide.

- The Similarity Analysis task provides methods to measure the similarity between two time series or among sequences in temporal data. For more information, see “SAS Viya: Similarity Analysis” in SAS Studio: Task Reference Guide.

New Tasks from SAS Studio 3.8

These tasks were originally added in SAS Studio 3.8. If you are upgrading from SAS Studio 5.1, these tasks are new to you.

Prepare Data Tasks

The Combine Tables task provides a variety of methods for combining two data sets. For more information, see “Combine Tables” in SAS Studio: Task Reference Guide.

Map Tasks

- The Choropleth Map task creates a map of polygonal areas. For more information, see “Choropleth Map” in SAS Studio: Task Reference Guide.

- The Text Map task creates a map that is overlaid with a text scatter plot. For more information, see “Text Map” in SAS Studio: Task Reference Guide.

- The Bubble Map task creates a map that is overlaid with a bubble plot. For more information, see “Bubble Map” in SAS Studio: Task Reference Guide.

- The Scatter Map task creates a map that is overlaid with a scatter plot. For more information, see “Scatter Map” in SAS Studio: Task Reference Guide.

- The Series Map task creates a map that is overlaid with a series plot. For more information, see “Series Map” in SAS Studio: Task Reference Guide.

Econometrics

- The Severity Models task estimates parameters of any arbitrary continuous probability distribution that is used to model the magnitude (severity) of a continuous-valued event of interest. For more information, see “Severity Models” in SAS Studio: Task Reference Guide.
The Spatial Regression Models task analyzes a class of linear spatial econometric models for cross-sectional data whose observations are spatially referenced or georeferenced. For more information, see “Spatial Regression Models” in SAS Studio: Task Reference Guide.

The Time Series Analysis task has been divided into these two tasks:

- The Univariate Time Series Analysis task analyzes a single time series where the values are equally spaced. This task provides analysis for these model types: ARIMA (autoregressive integrated moving average), ARIMAX, unobserved components, and regression with autocorrelated and heteroscedastic errors. For more information, see “Univariate Time Series Analysis” in SAS Studio: Task Reference Guide.

- The Multivariate Time Series Analysis task analyzes a vector of time series that are equally spaced. For more information, see “Multivariate Time Series Analysis” in SAS Studio: Task Reference Guide.

Statistics > Survival Analysis

These Survival Analysis tasks are new:

- The Nonparametric Survival Analysis task computes nonparametric estimates of the survival distribution function and compares survival functions of two or more groups. For more information, see “Nonparametric Survival Analysis” in SAS Studio: Task Reference Guide.

- The Proportional Hazards Regression task fits proportional hazards regression models to survival data. For more information, see “Proportional Hazards Regression” in SAS Studio: Task Reference Guide.

Enhanced Tasks

These tasks were enhanced:

- In the Filter Data task, you can now create an advanced filter. For more information, see “Filter Data” in SAS Studio: Task Reference Guide.

- Code generation options were added to these SAS Viya Econometrics tasks: Aggregate Loss Models, Cross-sectional Data Models, Panel Data Model, Hidden Markov Models, Spatial Regression Models. These new code generation options enable you to run these tasks using action sets in the CAS procedure.

- In the Replace Missing Values Task, you can now replace the missing values for a variable with a user-specified value. For more information, see “Replace Missing Values” in SAS Studio: Task Reference Guide.


- Language support for Kazakh has been added to these SAS Viya Text Analytics tasks: Text Summarization, Text Parsing and Topic Discovery, Boolean Rules, and Segmentation.

- Singular Spectrum Analysis is now available from the Time Series Exploration Task and the SAS Viya: Time Series Exploration task. For more information, see...
Enhancements to the Common Task Model

Use the common task model (CTM) to create your own SAS Studio tasks or to create a task prompt interface for SAS Studio jobs. Here are all the changes for this release. For more information, see SAS Studio: Developer's Guide to Writing Custom Tasks.

- In the DataSources element, you can now specify a default data source by using the defaultValue attribute. You can also specify whether the task user can change the data source by using the readOnly attribute.
  
  In the Velocity code, the getColumns, getEngine, and getRowsCount methods are new.

- In the Roles element, you can specify the default roles for the role control by using the new defaultValue attribute. You can also specify whether the task user can change the role values by using the readOnly attribute.
  
  In the Velocity code, the getDistinctCount and getDistinctValues methods are new.

- Filters enable you to populate option controls with the values from a column in the data source. The addition of filters supports dynamic and cascading prompts.
  
  These input types support filters: combobox, dualSelector, and select.
  
  To give you the most choice when selecting dynamic values, combobox, dualSelector, and select also support these attributes:

  - Use the allowAllValues attribute to specify whether the task user can select all possible values.
  - Use the allowMissingValues attribute to specify whether the task user can select missing values.

- In the Options element, you can now perform these tasks:

  - Use the active attribute to specify that the option value is available even when hidden, disabled, or when the value is not used in the user interface.
Use the `helpMessageRef` attribute to display help content.
Use the `hide` attribute to specify whether to display the option.

For the combobox input type, these attributes are new:
- Use the `allowAllValues` attribute to specify whether the task user can select all possible values.
- Use the `allowMissingValues` attribute to specify whether the task user can select missing values.
- Use the `filter` attribute to specify how to get the values for this option.
- Use the `sourceLink` attribute to specify that the values for the combobox control should be pulled from another source option.

For the datepicker input type, the new `minDate` and `maxDate` attributes enable you to specify a minimum and maximum value for the date values.

The dualSelector input type has these new attributes:
- Use the `allowAllValues` attribute to specify whether the task user can select all possible values.
- Use the `allowMissingValues` attribute to specify whether the task user can select missing values.
- Use the `dataType` attribute to specify the type of data for a user-specified value.
- Use the `filter` attribute to specify how to get the values for this option.
- Use the `maxItems` and `minItems` attributes to specify the maximum and minimum number of values that must be selected from the list of available values.
- Use the `sourceLink` attribute to specify that the data for the dualSelector control should come from another option.

Use the new markdown input type to display formatted text to the user. Standard markdown is accepted.

In the multientry input type, you can specify the data type for user-specified values by using the new `dataType` attribute.

In the numbertext and numberstepper input type, these attributes are new:
- Use the `formatValue` attribute to format the number to include locale-specific delimiters in the user interface.
- Use the `maxInclusive` and `minInclusive` attributes to specify whether to include the maximum or minimum value in the list of values.

The new numericrange input type enables you to specify a range of numeric values.

In the radio input type, you can specify that the value for the Velocity variable is the return value rather than the name of the radio control by using the new `returnValue` attribute.

You can use these new input types to create date controls:
- `daterange`
- `datetimepicker`
- `monthpicker`
- quarterpicker
- timepicker
- weekpicker

For the select input type, these attributes are new:

- Use the `allowAllValues` attribute to specify whether the task user can select all possible values.
- Use the `allowMissingValues` attribute to specify whether the task user can select missing values.
- Use the `filter` attribute to specify what data to use for this option.
## About SAS Studio

SAS Studio is a development application for SAS that you access through your web browser. With SAS Studio, you can access your data files, libraries, and existing programs, and you can write new programs. You can also use the predefined tasks in SAS Studio to generate SAS code. When you run a program or task, SAS Studio connects to a SAS server to process the SAS code. The SAS server can be a hosted server in a cloud environment or a remote server in your local environment. After the code is processed, the results are returned to SAS Studio in your browser.
SAS Studio supports multiple web browsers, such as Microsoft Edge, Apple Safari, Mozilla Firefox, and Google Chrome. For more information, see *SAS Viya for Linux: Deployment Guide*.

In addition to writing and running your own SAS programs, you can use the predefined tasks that are included with SAS Studio to analyze your data. The tasks are based on SAS procedures and provide access to some of the most commonly used graph and analytical procedures. You can also use the default task template to write your own tasks.

SAS Studio includes two different perspectives: the Standard perspective and the Interactive perspective. A *perspective* is a predetermined set of features that is customized to meet the needs of a specific user type. By selecting a specific perspective, you can narrow the choices that are available in the interface and focus on the features that you need to use regularly. By default, when you open SAS Studio, the Standard perspective is selected. After you open SAS Studio, you can change the perspective by using the *Options* menu on the toolbar. For more information, see “Understanding Perspectives” on page 15.
Using SAS Studio

About Using SAS Studio

When you sign on to SAS Studio, the Start Page is available, so you can quickly get started writing a new SAS program, importing data, or creating a query. Use the icons to access the six sections of the navigation pane.

Note: To sign out of SAS Studio, click the button with the first initial of your user name on the far right of the application toolbar and select Sign out. Do not use the Back button on your web browser.

At the top, the application bar enables you to access other SAS applications. Click the search icon to search for items across SAS applications.

The main window of SAS Studio consists of a navigation pane on the left and a work area on the right. The navigation pane provides easy access to your open files, your folder shortcuts and SAS content, your tasks and snippets, the libraries that you have access to, your Git repositories, and your file references. The Open Files section is displayed by default. The File References section is not displayed by default, but can be opened by selecting View ➤ Navigation panes ➤ File References. For more information, see “Using File References” on page 7.
The work area is used to display your data, code, tasks, logs, and results. As you open these items, they are added to the work area as windows in a tabbed interface. When you first open SAS Studio, the Start Page tab is displayed in the work area by default. For more information, see “Using the Work Area” on page 9.

Using the Navigation Pane

About Using the Navigation Pane

You can expand the sections of the navigation pane by clicking the section that you want to view. To close and reopen the navigation pane, click a section button twice.

Note: If you want to drag an item to another location in a section of the navigation pane that has a scroll bar, drag the item to the top right or lower right corner of the pane to scroll upward or downward.

Accessing Your Open Files

The Open Files section of the navigation pane enables you to quickly view and access all of the files that you have opened in your current SAS Studio session. To access a file from the list of open files, click the file that you want to view. You can also save changes to all files in the list by clicking Save all.

Using the Explorer

The Explorer section in the navigation pane enables you to access files and folders from your folder shortcuts, your server file system, and your SAS Content server locations. For more information, see SAS Drive: Getting Started.

Note: The ability to access files on a SAS Content server is available only in SAS Studio 5.2 (Enterprise). For more information, contact your site administrator.

You can use the Explorer section to create folders and folder shortcuts as well as download and upload files. From the folders tree, you can expand and collapse folders, copy and move items, and open items in folders by double-clicking them or dragging them to the work area. In addition, you can open files from your folders and folder shortcuts by clicking Open on the SAS Studio toolbar.

You can change the files that are displayed in the Explorer section in the following ways:

- To view any hidden files and folders in a folder, right-click the folder and select Show hidden items.
- To display only SAS program files, click and select Show .sas files.
- To display only SAS log files, click and select Show .log files.

You can view items in a folder as text by right-clicking the file and selecting View file as text.

To open a file as a SAS program, right-click the file and select Open as program.
To create a folder shortcut:

1. Click in the navigation pane to open the Explorer section. Next, click and select Folder shortcut. The New Folder Shortcut window appears.

2. In the Name box, enter the name of the folder shortcut.

3. Click to select a folder. To create a folder, click .

4. Click OK to create the folder shortcut. The new shortcut is added to the list of folder shortcuts.

Note: You can rename a folder shortcut by selecting the folder shortcut in the Explorer section and clicking . In the Shortcut Properties window, enter a new name in the Name box.

To create a folder, select the folder in the Explorer section in which you want to create the folder. Click and select Folder. The New Folder window appears. Enter the name of the new folder. The new folder is added to the list of folders.

To download one or more files, right-click the files that you want to download and select Download file.

To upload one or more files from your local computer, select the folder to which you want to upload the files and click . The Upload Files window appears. Click to browse for the files that you want to upload. The maximum size of each uploaded file cannot exceed 100 MB.

Note: You can sort the files that are listed in the Explorer section by right-clicking the folder that you want to sort and selecting Sort by. You can sort the files by name, modified date, or size or choose no sort order. By default, the files are sorted in ascending order by file name.

Working with Tasks

The Tasks section of the navigation pane enables you to access tasks in SAS Studio. Tasks are based on SAS procedures and generate SAS code and formatted results for you. SAS Studio is shipped with several predefined tasks that you can run. You can also edit a copy of these predefined tasks, and you can create your own new tasks.

To create a new task, click and select Task. SAS Studio creates a template in the work area that you can use to create custom tasks for your site. Custom tasks can be accessed from the My Tasks folder or from the Explorer section of the navigation pane. For more information, see “Create a New Task” on page 139.

To edit a task that you have created, right-click the task in the My Tasks folder and select Edit task template. The XML code that is used to create the task is opened in the work area. If you want to edit a predefined task, you must first right-click the task and select Copy as Task Template and then select either My Tasks or Explorer. For more information, see “Edit a Predefined Task” on page 138.
Working with Snippets

The Snippets section in the navigation pane enables you to access your saved snippets. Snippets are lines of commonly used code or text that you can save and reuse. Snippets can contain SAS code, XML code, and text. You can insert snippets in locations including the Program Editor, XML editor, text editor, job definitions, job forms, and job prompts. To insert a snippet, right-click the location where you want to insert the snippet and select **Insert snippet**.

SAS Studio is shipped with predefined code snippets that you can use. You can also edit a copy of these snippets and create your own custom snippets. Your custom snippets can be accessed from the My Snippets folder. For more information, see Chapter 2, “Working with Programs,” on page 21.

To edit a snippet that you have created, double-click the snippet to open it in the Program Editor and enter your changes. If you want to edit a predefined snippet, you must first right-click the snippet and select **Copy snippet to**, and then select either **My Snippets** or **Explorer**.

**Note:** You can edit only the snippets that are in the **My Snippets** folder or a folder that you can access from the **Explorer** section.

Working with Libraries

The Libraries section of the navigation pane enables you to access your SAS libraries. SAS tables are stored in SAS libraries. From the Libraries section, you can open SAS tables and add them to your programs. You can use the Libraries section to expand a table and view the columns in that table. The icon in front of the column name indicates the type.

**Note:** Caslib names cannot be longer than 256 characters. SAS librefs are limited to eight characters.

Here are examples of common icons for the column types.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Type of Column</th>
</tr>
</thead>
<tbody>
<tr>
<td>📛</td>
<td>Character</td>
</tr>
<tr>
<td>📉</td>
<td>Numeric</td>
</tr>
<tr>
<td>📟</td>
<td>Date</td>
</tr>
<tr>
<td>⌚️</td>
<td>Datetime</td>
</tr>
</tbody>
</table>

You can drag tables and columns from the Libraries section to a program, and SAS Studio adds the names of the dragged items to your program. The cursor changes as you drag an item to indicate when you can drop the item to insert it in the program. If the item name is a reserved word in a database, SAS Studio creates a name literal by automatically adding quotation marks and a lowercase letter n to the
item name. For more information, see “Opening and Creating Programs” on page 22.

You can also drag tables from the Libraries and Explorer sections to other locations in the work area to open them.

Note: The Sasuser library is read only, as in any SAS server environment. You cannot save content to this library.

You can also create new libraries and assign existing libraries.

To create a new library:

1. Click in the navigation pane and then click *. The New Library window appears.

   ![New Library Window](image)

   - In the Name box, enter the libref for the library. The libref must be eight characters or fewer.
   - In the Path box, enter the physical path where the library resides. The path must be a location that you can access on the SAS server to which SAS Studio is connected.
   - In the Options box, specify any configuration options that you need. For the appropriate options, see the documentation for your operating environment.
   - If you want to access this library each time you use SAS Studio by adding it to your SAS autoexec file, select Re-create this library at start-up.
   - Click OK to create the library. The new library is added to the list of libraries in the navigation pane.

Using File References

File references enable you to quickly access files that you specify. You can create a file reference to a file in your SAS Content folder, via a URL, or on your SAS server. You can use the SAS Studio user interface to create a file reference to a file in your
SAS Content folder or via a URL. To create a file reference to a file on your SAS server, you must use the FILENAME statement.

**Note:** You can also use the FILENAME statement to create a file reference to a file in a remote location that is accessible by your SAS server.

To create a new file reference to a file in your SAS Content folder or via a URL:

1. Click 🗄 in the navigation pane and then click 🎁. The New File Reference window opens.

2. In the **Name** box, enter the file reference name.

3. Use the **Access method** box to specify how to access the file:
   - **DISK** - accesses the file from your SAS file system. In the **Location** box, browse to find the file that you want to use.
   - **FILESRVC** - accesses the file from your SAS Content folder. In the **Location** box, browse to find the file that you want to use.
   - **URL** - accesses the file by using the specified URL. In the **URL** box, enter the URL for the file.

4. If you want this reference to be available the next time you use SAS Studio, select **Re-create this file reference at start-up**.

5. Click **OK** to create the file reference. The new file reference is added to the list of file references in the navigation pane.

To create a file reference to a file on or accessible by your SAS server:

- In the Program Editor, enter and run a FILENAME statement using the following syntax:

  ```
  FILENAME fileref 'external-file';
  ```

  **Note:** Make sure that you specify the complete pathname for the external file and use the proper path syntax for the operating system that your SAS server is
running. For example, Microsoft Windows uses backslashes (\) in its pathnames, and UNIX uses forward slashes (/).

You can open a file from a file reference by double-clicking it or dragging it to the work area.

Working with Git Repositories

The **Git Repositories** section in the navigation pane enables you to access the basic Git features from within SAS Studio. These features include cloning repositories, committing and stashing file changes, pulling and pushing files, viewing your repository history, creating and merging branches, rebasing a branch, and performing a basic differentiation between files in your local repository. For more information, see “About Git Integration in SAS Studio” on page 109.

Customizing the Navigation Pane

By default, five sections of the navigation pane are displayed when you open SAS Studio in the Standard perspective. To customize which sections are displayed, click **View** and select **Navigation panes**. Select or clear any sections that you want to add or remove. The navigation pane is updated immediately.

**Note:** The **File References** section is not displayed by default.

Using the Work Area

**About Using the Work Area**

The work area is the main portion of the SAS Studio application for accessing programs and tasks and for viewing data. The work area is always displayed and cannot be closed. When you open a program, task, or table, the windows open as new tabs in the work area. The code, log, and results that are associated with programs and tasks are grouped together under the main tab for the program or task.

When you first open SAS Studio, the Start Page tab is displayed in the work area by default. The Start Page provides quick access to a few basic tasks (for example, creating a program or query) and links to learning resources. If you close the Start Page tab, you can re-open it by selecting **View** ⇒ **Start Page**. You can use the **Show Start Page** option to control whether to display the Start Page tab each time you start SAS Studio. For more information, see “Setting the Start Up Preferences” on page 145.
Customizing the Work Area

By default, the work area is displayed beside the navigation pane, but you can maximize the work area and hide the navigation pane. You can also close all of the tabs in the work area at once.

To maximize the work area, select View → Fullscreen.

Note: To reopen the navigation pane, double-click any tab in the work area or right-click a tab and select Restore.

You can also close some or all of the open tabs in the work area. To close all tabs that are open in the work area, right-click any tab and select Close all. To close all tabs except the active tab, right-click the active tab and select Close others. You are prompted to save any unsaved programs or tasks.

Rearranging the Tabs in the Work Area

In the work area, you can rearrange the tabs by moving them to the left or right. You can also create separate vertical and horizontal groups and move tabs between the groups. To rearrange a tab, right-click the tab that you want to move and select the appropriate option.
Searching in SAS Studio

You can use the Search feature to search all of the sections of the navigation pane. You can specify which sections and which types of items you want to include in your search. You can also specify whether you want the search to be case-sensitive and match whole words only.

To access the Search feature, select View > Search. The Search tab opens in the work area.
To search for an item, enter the text that you want to search for in the search term box. To specify the areas of the navigation pane to search, click . By default, SAS Studio searches your user home folders in the Explorer section of the navigation pane.
To make your search case-sensitive, click $A_a$. To match the whole word in your search, click $^W^W$.

When you have specified your search criteria, click $\text{Search}$. You can open items in the search results by double-clicking them.
Using the Console

The Console window displays a list of all of the messages, such as error messages and warnings, that are generated as you run programs and tasks. To open the console, select View → Console. If you minimize the console tab, you can click in the lower right corner of the work area to restore the console or to maximize the console.

You can use the buttons at the top of the window to filter the types of messages that are displayed. By default, all types of messages are displayed. The messages are cleared by default when you sign out of SAS Studio. You can manually clear the messages from the console by clicking Clear All.

Using the Submission Status Window

The Submission Status window displays information about the programs, tasks, and queries that you run in SAS Studio. To open the Submission Status window, select View → Submission Status or click Submission in the lower right corner of your SAS Studio browser window.
The Submission Status window includes an entry for each time you have run a program, task, or query. You can use the Submission Status window to return to an earlier version of a program, task, or query. When you click a program, task, or query from the Submission Status window, you open the version of the program, task, or query that was run at that time. The program or automatically generated task or query code opens in a new tab from which you can make and save changes.

When you position your mouse pointer over the name of a program, task, or query, the following options become available:

- Removes the submission from the Submission Status window.
- Cancels the submission if it is currently running.

You can subset the messages that are displayed by clicking the type of message that you want to view. You can also click Clear All to remove all entries from the list and Cancel All to cancel all programs, tasks, and queries that are currently running, excluding background submitted jobs.

### Understanding Perspectives

Because SAS Studio can be used by a variety of people and groups within an organization, you can choose to view a specific subset of features, or perspective, that meets your needs best. Perspectives are sets of functionality that are customized to meet the needs of different types of users. SAS Studio includes two perspectives: the Standard perspective and the Interactive perspective.

The Standard perspective is for users who want to be able to both write SAS programs and use the predefined tasks. In the Standard perspective, SAS program files are run as a whole. By default, the Standard perspective opens with the Start Page tab and includes these sections of the navigation pane: Open Files, Explorer, Tasks, Snippets, Libraries, and Git.

The Interactive perspective is designed for users who intend to use SAS Studio mainly for writing and editing SAS programs that use interactive SAS procedures. By default, the Interactive perspective opens with the Start Page tab, and the Log and Results tabs are displayed in a separate horizontal tab group. In addition, the log data and results from all of your program tabs are appended together in one log tab and one results tab. In the Standard perspective, there are separate log and results tabs for each program. For more information, see “Working with the Interactive Perspective” on page 35.
The differences between the perspectives can be viewed in the following table:

<table>
<thead>
<tr>
<th>Element in SAS Studio</th>
<th>Available in Standard?</th>
<th>Available in Interactive?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Navigation Pane sections</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open Files</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Explorer</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Tasks</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Snippets</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Libraries</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Git</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>File References</td>
<td>Yes, but not displayed by default</td>
<td>Yes, but not displayed by default</td>
</tr>
<tr>
<td>Element in SAS Studio</td>
<td>Available in Standard?</td>
<td>Available in Interactive?</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Ability to create and open queries</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Ability to create and run job definitions</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Options to recall code and clear code on submit</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Options to generate PDF, RTF, Excel, and PowerPoint output</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Ability to import data</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Ability to export data</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Ability to print data</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Ability to validate an expression in the Filter Table Rows window</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Ability to validate the autoexec.sas file</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Ability to insert custom code</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Ability to update column properties</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

After you have started SAS Studio, you can change the perspective that you are using by selecting Options ➤ Change perspective and then selecting the perspective that you want to use.

You can specify which sections of the navigation pane are displayed in SAS Studio by selecting View ➤ Navigation panes.

Editing the Autoexec File

The autoexec.sas file includes SAS statements that run each time you start SAS Studio and connect to your SAS server. For example, you can use the autoexec.sas file to assign libraries that you want to be available every time you use SAS Studio in both the Standard and Interactive perspectives.

Note: If you create a new library by using the New Library window, you can select the Re-create this library at start-up option to automatically add the LIBNAME
statement to the autoexec.sas file. For more information, see “Working with Libraries” on page 6.

To edit the autoexec.sas file:

1. Select Options ⇒ Autoexec file.
2. Enter the code that you want to include in the autoexec.sas file.
3. To validate your syntax, click Run. The Log tab opens so that you can view the log.

Note: The option to run the autoexec file is available only in the Standard perspective. For more information, see “Understanding Perspectives” on page 15.

4. Click Save to save and close the autoexec file.

Inserting Custom SAS Code

You can specify the SAS code to run before or after the code for programs, tasks, queries, and imports. This custom code is always run and persists between sessions.

To include custom SAS code:

1. Select Options ⇒ Custom code.
2. Specify the preamble and postamble SAS code. Click Run to validate the syntax.
3. On the Options tab, specify where to insert the custom code. You can choose from programs, tasks, queries, and imports. You can also specify whether to include the code if you are running a background submit.
4. Click Save.

Resetting Your SAS Session

You can terminate your current SAS session and create a new workspace session by using the Reset SAS Session option. The Reset SAS Session option terminates your current workspace session by running the ENDSAS statement and then creates a new workspace session. When the new workspace session is started, SAS Studio runs the autoexec file to initialize the session. To reset your SAS session, select Options ⇒ Reset SAS session.

Note: All files in the Work library are deleted when you reset your SAS session.
Managing Keyboard Shortcuts

SAS Studio includes many keyboard shortcuts that you can use to perform a task. For a list of keyboard shortcuts, see “Keyboard Shortcuts” in SAS Studio: Accessibility Features.

Some keyboard shortcuts can be customized. To manage your keyboard shortcuts, select Options ⇒ Manage keyboard shortcuts. When you enter a new keyboard shortcut, you are prompted to confirm your changes.
About the Code Editor

SAS Studio includes a color-coded, syntax-checking editor for editing new or existing SAS programs. You can also edit SOURCE entries in SAS catalogs. The editor includes a wide variety of features such as autocompletion, automatic formatting, and pop-up syntax help. With the code editor, you can write, run, and...
save SAS programs. You can also modify and save the code that is automatically generated when you run a task.

SAS Studio also includes several sample code snippets that you can use to make programming common tasks easier.

Opening and Creating Programs

Opening a Program

You can open SAS programs from the Explorer section of the navigation pane. To open a program, expand the appropriate folder and double-click the program that you want to open, or right-click the program in the navigation pane and select Open. The program opens in a new tab in the work area.

Note: You can also open a file by clicking Open on the toolbar and selecting a file from your available folders. If you are opening a program or text file, then you can specify the text encoding that you want to use by selecting Open with encoding and clicking Open. Use the Select Encoding window to specify the encoding that you want.

Note: Opening very large program files can affect your performance. By default, if you open a program file that is greater than 10 MB, you are prompted to confirm whether you want to continue opening the file. You can use the Display warning if text files are larger than specified size option to change the file size that determines when a warning is displayed. For more information, see “Setting General Preferences” on page 146.

In the lower left corner of the Program Editor, you can view the current line and column position of the cursor. If you are viewing a *.sas or *.txt file, the text encoding is displayed as well. To go to a specific line in the program, right-click in the program and select Go to line. Enter the line number in the Go to line box. You can also right-click in the program and select Encoding and select another character-set encoding option.
Creating a New Program

To create a new program, select New ⇒ SAS Program. A program window opens in a new tab in the work area.

Note: You can also use the New menu to begin importing a file as well as create new queries, tasks, jobs, text files, and XML files.

Running a Program

After you have written your program, you can run the entire program or you can select specific lines of code to run. To run the entire program, click Run on the toolbar. To run a portion of the program, select the lines of code that you want to run and then click Run.

By default, if there are no errors, the results open automatically, and if there are errors, the Log tab opens. You can click one or more of the Errors, Warnings, and Notes sections to view the messages. When you click a message, SAS Studio highlights it for you in the log so that you can see exactly where the message occurs in the log.
**Note:** You can specify which tab you want to display after you run a program or task by using the General preferences. For more information, see “Setting General Preferences” on page 146.

**Note:** Because you are working in a server environment, do not include the ENDSAS statement in your SAS programs. If you run a program that contains ENDSAS, reset your SAS session by selecting **Options → Reset SAS session**.

### Saving a Program

To save updates to an existing program in its current location, click ![on the toolbar.](image) To save the entire program with a new name or location, click ![on the toolbar.](image) If you want to specify the text encoding when you save a program or text file, use the text encoding button in the lower left corner of the application window to select a new encoding option before you save the file.

**Note:** You can also create a SAS program package or a program summary when you save your program. For more information, see “Creating a SAS Program Package” on page 32 and “Creating a Program Summary” on page 33.
Scheduling a Program as a Job

About Scheduling a Program

You can scheduled a saved program to run at a specified time and frequency. Scheduled programs still run even when you are not using SAS Studio.

Scheduling a Program

To schedule a program to run at a specified time:

1. Right-click the program in the Explorer section of the navigation pane and select Schedule as a job.

2. In the New Trigger window, specify how frequently to run the job. You can also specify the time at which the job should run and the start date and the end date for running the job.

3. Click Save.

To view all scheduled programs and jobs, select View ⇒ Selected Jobs.

Using the Background Submit Feature

About the Background Submit Feature

You can run a saved SAS program as a background job, which means that the program can run while you continue to use SAS Studio. You can view the status of programs that have been submitted as background jobs, and you can cancel programs that are currently running in the background.

Running a Program as a Background Job

To run a program as a background job, right-click the program in the navigation pane and select Background submit. Before the program is run, the background process changes the current working directory to the directory in which the program is located.

Note: You cannot run a program as a background job if the program is saved on an FTP server or on a SAS server running on the native z/OS file system. Programs that are saved on a z/OS SAS server running the HFS file system can be run as background jobs.

By default, a notification message is displayed when the program is submitted and again when the program has finished running. If you log off from SAS Studio while the program is running, the program continues to run, but the notification message that indicates when the program is finished is not displayed.
Note: Because a background job uses a separate workspace server, any libraries that are created by your program do not appear in the Libraries section of the navigation pane in SAS Studio.

To view the status and properties of your background jobs, click View and select Submission Status. Background jobs are indicated by 🔄.

You can click 🔄 to cancel the submission. Any output that has been created by the job is not deleted. Click 🛠️ to remove the submission from the list.

Customizing Your Background Job Submissions

The Preferences window enables you to customize how to handle background job submissions.

To change whether existing log and output files are deleted or overwritten when you rerun a background job, select Options ➔ Preferences. Click Background Submit.

For more information about each option, see “Setting Background Submit Preferences” on page 155.

Using the Autocomplete Feature

About the Autocomplete Feature

The autocomplete, or code completion, feature in the code editor can predict the next word that you want to enter before you actually enter it completely. The autocomplete feature can complete keywords that are associated with SAS procedures, statements, macros, functions, CALL routines, formats, informats, macro variables, SAS colors, style elements, style attributes, and statistics keywords, and various SAS statement and procedure options. The autocomplete feature can also complete librefs and table names.

Note: The autocomplete feature is available only for editing SAS programs.

This example shows the keywords and help that appear when you enter `proc a` in the code editor.
In this example, you select **APPEND** from the list of procedures, so that `proc append` appears in the code editor. When you enter a space, the code editor displays a list of options for the APPEND procedure.

How to Use the Autocomplete Feature

To use the autocomplete feature:

1. How you open the autocomplete list depends on the keyword that you want to add.
If you want to add a global statement, DATA step statement, CALL routine, procedure, macro statement, or automatic macro variable, enter the first one or more letters of the keyword that you want to use.

A window opens with a list of suggested keywords that begin with those letters.

If you want to specify colors, formats, informats, macro functions, SAS functions, statistics keywords, style elements, or style attributes, position your mouse pointer in a comment and press Ctrl+spacebar. To navigate through the list of options backward, press Ctrl+Shift+spacebar or use the arrow keys at the top of the pop-up window.

**Note:** These shortcuts work even if you have deselected the **Enable autocomplete** option in the Preferences window. For more information, see “Customizing the Code Editor” on page 56.

You can navigate to the keyword that you want to use in several ways:

- Continue to type until the correct keyword is selected (because the matching improves as you type).
Scroll through the list by using the up and down arrow keys, the Page Up and Page Down keys, or your mouse.

3. You can add the keyword to your program by double-clicking the selected keyword or by pressing the Enter key.

Using the Syntax Help

The code editor displays brief SAS syntax documentation as you write and edit your programs. You can display the Help in the following ways:

- Right-click a keyword in your program and select Syntax help.
- Start entering a valid SAS keyword, and then click a suggested keyword in the autocomplete window.
- Position the mouse pointer over a valid SAS keyword in your program. This works only if you have selected the Enable hint when you hover over keywords option in the Editors preferences. For more information, see “Customizing the Code Editor” on page 56.

The SAS Product Documentation provides more comprehensive usage information about the SAS language, but the syntax help in the code editor can get you started with a hint about the syntax or a brief description of the keyword. You can get additional help by clicking links in the syntax help window as follows:

- Click the keyword link at the top of the window to search the support.sas.com website for the keyword.
- Click the links at the bottom of the window to search for the keyword in the SAS Product Documentation, Samples and SAS Notes, and SAS Technical Papers.
Matching Parentheses

You can use the parenthesis matching feature to track nested parentheses within a program. The code editor highlights both the open and close parentheses. If only one parenthesis is highlighted, then you know that you are missing a parenthesis. This feature can be used to match parentheses, square brackets, angle brackets, and braces.

To match parentheses, position the cursor in front of the open parenthesis or just after the close parenthesis that you want to match. The parenthesis and its match are highlighted. If the parenthesis does not have a match, then it is not highlighted.

Note: You can use the Code Editor Settings page to turn the Highlight matching brackets option off. You can also select an option to automatically insert a closing bracket or parenthesis. For more information, see “Setting General Code Editor Preferences” on page 151.

Selecting Columns of Text

You do not have to select entire horizontal lines of text. You can select columns or vertical blocks of text.
To select a column or vertical block of text:

- In Window environments, press the Alt key while you select the text with the left mouse button.
- In Mac OS X environments, press the Option key while you select the text with the left mouse button.

### Adding Table Names and Column Names

From the **Libraries** section of the navigation pane, you can use a drag-and-drop operation to move table names and column names into the SAS code. For example, you can move the Sashelp.Cars table into the DATA option for the PRINT procedure. When you release the mouse, the fully qualified name for the table appears in your code.

**Note:** If the table or column name is a reserved word in a database, SAS Studio creates a name literal by automatically adding quotation marks and a lowercase letter _a_ to ensure that the name is evaluated correctly by your program.

### Inserting a Pathname

From the **Explorer** section of the navigation pane, you can right-click a folder or a *.sas file and select **Insert as path** to add the pathname of the folder or the pathname and filename of the file to your program. For example, you can right-click a folder such as `/products/salesdata/` and select **Insert as path** to add it to a LIBNAME statement.

### Editing the Code from a Task

You can edit the code that is generated automatically when you run a task and then run it with your modifications. When you edit the code, SAS Studio opens it in a separate program window. The code is no longer associated with the original task.

To edit a program generated by a task:

1. On the appropriate task tab in the work area, click **Code** to display the code that is associated with the task.
2. On the toolbar, click **Edit**. The code is opened in a new program window.
Creating a SAS Program Package

A SAS Program Package is a file that contains a snapshot of a SAS program along with its log and HTML results. You can create a program package from code that you have written as well as code that is automatically generated when you run a task. When you open a program package in SAS Studio, you can access the code, log, and results without running the program again. If you make changes to the code and rerun it, the package is not automatically updated. You must save the package again to keep the changes.

Note:
- The program package does not include PDF or RTF results.
- You cannot create a program package if you are using the Interactive perspective.

To create a program package file, open the code that you want to use and click \( \text{Package} \). Specify the file location and name, and then select \( \text{Package} \) as the Type option.

Note: If you open a program package file and want to save the program or log individually or download the results as an HTML, PDF, or RTF file, you must rerun the program after you open the program package file.
Creating a Program Summary

You can create a summary page for code that you have written as well as code that is automatically generated when you run a task. The Program Summary page is an HTML file that opens in a separate browser tab and includes information about the program execution, the complete SAS source code, the complete SAS log, and the results. To view the Program Summary page for a program, click "Open in a browser tab" and select "Summary."  

Note: The Program Summary is available only after you have run the program.

You can also save a Program Summary file to a folder that you specify by clicking "Summary" as the "Save as type" option. The Program Summary is saved as an HTML file.

Using Macro Variables

Macro variables can be used to add information that is obtained when a program or task is run, such as the name and version number of the application. You can reference these items within code, titles, or footnotes by preceding them with "&". For example, you can use macros in a footnote like the following:

Generated with &_CLIENTAPP &_CLIENTVERSION

Note: In addition to the following macro variables, you can run this code to see other user-generated and automatic macro variables that are available:

`%PUT _ALL_;`

For information about SAS macro functions and variables, see SAS Macro Language Reference.

<table>
<thead>
<tr>
<th>Macro Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_CLIENTAPP</td>
<td>Name of the client application.</td>
</tr>
<tr>
<td>_CLIENTAPPABBREV</td>
<td>Abbreviated name of the client application.</td>
</tr>
<tr>
<td>_CLIENTMACHINE</td>
<td>Node name of the client machine.</td>
</tr>
<tr>
<td>_CLIENTMODE</td>
<td>Type of SAS Studio deployment: Mid-Tier (enterprise), Single User, Basic, or SAS Viya.</td>
</tr>
<tr>
<td>_CLIENTVERSION</td>
<td>Application version.</td>
</tr>
</tbody>
</table>
### Macro Variable Description

<table>
<thead>
<tr>
<th>Macro Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_SASHOSTNAME</code></td>
<td>Server node name (IP address or DNS name).</td>
</tr>
<tr>
<td><code>_SASPROGRAMFILE</code></td>
<td>The full path and filename of the SAS program that is currently being run. This macro variable is available only for SAS program files that are saved on the same server on which your SAS Studio code is running.</td>
</tr>
<tr>
<td><code>_SASPROGRAMFILEHOST</code></td>
<td>The server node name on which the current SAS program is being run.</td>
</tr>
<tr>
<td><code>_SASWORKINGDIR</code></td>
<td>Current working directory.</td>
</tr>
<tr>
<td><code>SASWORKLOCATION</code></td>
<td>Location of the Work library.</td>
</tr>
<tr>
<td><code>SYSPROCESSNAME</code></td>
<td>Name of the current SAS process.</td>
</tr>
<tr>
<td><code>SYSPROCESSMODE</code></td>
<td>Current SAS session run mode or server type name.</td>
</tr>
<tr>
<td><code>SYSVLONG4</code></td>
<td>SAS software release number, maintenance level, and four-digit year.</td>
</tr>
</tbody>
</table>

**Note:** If you specify `%put _all_` or `%put _global_` in your SAS program, the output does not include any special characters. For example, slashes are not included in directory paths. To view the output with these special characters, you must specify the individual macro variable by name, such as `%put &_sasprogramfile;`.

### Using Your Submission History

You can use the Submission Status window to access prior versions of your submitted code. Select View ⇒ Submission Status on the toolbar, or click Submission in the lower right corner of your SAS Studio browser window. The Submission Status tab includes an entry for each time you have run a program. When you click a program from the Submission Status window, you open the version of the program that was run at that time. The program opens in a new tab in the work area from which you can make and save changes.

**Note:** The submission history is cleared automatically when you sign off from SAS Studio.

For more information, see “Using the Submission Status Window” on page 14.
Automatically Formatting Your SAS Code

You can use the code editor to make your programs easier to read by automatically formatting your code. When you automatically format your code, line breaks are added, and each line is correctly indented according to its nesting level. To format the code in the code editor, click the toolbar. You can also choose to format a selected section of your code.

For example, the following code is difficult to read because it lacks indentation and logical line breaks:

```sas
data topn;
  length rank 8; label rank="Rank";
  set topn; by &category descending &measure;
  if first.&category then rank=0; rank+1;
  if rank le &n then output;
run;
```

After you use the automatic code-formatting feature, the program looks like this:

```sas
data topn;
  length rank 8;
  label rank="Rank";
  set topn;
  by &category descending &measure;
  if first.&category then
    rank=0;
    rank+1;
  if rank le &n then
    output;
run;
```

Working with the Interactive Perspective

What Is the Interactive Perspective?

Some SAS procedures are interactive, which means they remain active until you submit a QUIT statement, or until you submit a new PROC or DATA step. In SAS Studio, you can use the code editor to run these procedures, as well as other SAS procedures, in the Interactive perspective.

By using the Interactive perspective, you can run selected lines of code from your SAS program and use the results to determine your next steps. For example, the OPTMODEL procedure in SAS/OR enables you to model and solve mathematical programming models. By running this procedure interactively, you can quickly check results for parts of the program and determine whether you need to make any modifications without running the entire program.
Running a Program in the Interactive Perspective

When you run a program in the Interactive perspective, SAS Studio does not add any automatically generated code, such as ODS and %LET statements, to your program. In addition, results are generated only in HTML. In the Interactive perspective, the log and results are appended to the existing log and results. Previously submitted code remains active until you terminate it.

For example, suppose you have the following program:

```sas
proc sql;
    select * from sashelp.cars;

    select * from sashelp.class;
quit;
```

In the Standard perspective, if you select the first two lines of code and run them, the code runs successfully. If you then select the last two lines of code and run them, the code fails because the PROC SQL statement is missing.

If you switch to the Interactive perspective and follow the same steps, the last two lines of code run successfully because the PROC SQL statement is still active.

Note: For documentation about specific procedures, see the SAS Programmer’s Bookshelf on support.sas.com.

Using the DATA Step Debugger

About the DATA Step Debugger

The DATA Step Debugger is a tool that enables you to find logic errors in a DATA step program. With the DATA Step Debugger, you can watch the variable values in a program change as the program runs. You can execute the program line by line, and you can also set specific breakpoints in the program.

Note: The DATA Step Debugger works only with DATA step programs.

Getting Started with the DATA Step Debugger

What Is Debugging?

_Debugging_ is the process of removing logic errors from a program. Unlike syntax errors, logic errors do not stop a program from running. Instead, logic errors cause
the program to produce unexpected results. For example, if you create a DATA step that tracks inventory, and your program shows that you are out of stock when your warehouse is actually full, you have a logic error in your program.

Without the DATA Step Debugger, you could debug your program by adding code, such as PUT statements, to your program and examining the results and log. Although the SAS log helps you identify data errors, the DATA Step Debugger offers you an easier, interactive way to identify logic errors, and sometimes data errors, in DATA steps.

How Debugging Works

When you open a DATA step in the DATA Step Debugger, SAS compiles the step, displays the code in the DATA Step Debugger window, and pauses until you begin execution. Each time execution is paused, the current values of each variable are displayed in the variables pane. For more information, see “Inspecting Variable Values” on page 270.

You can execute your program in the DATA Step Debugger in several ways:

- If you begin execution by clicking ➤, SAS executes each statement in the DATA step.
- To suspend execution at a particular line in the DATA step program, select one or more statements on which to set breakpoints. When you click ➤, SAS executes the program until a breakpoint is reached. For more information, see “Setting and Clearing Breakpoints” on page 40.
- To execute the DATA step one statement at a time, click ⊕. You can examine the variable values after each step in the program. For more information, see “Executing the Program Step by Step” on page 41.
- To execute the DATA step until the value of a specific variable changes, select the variable to watch and click ➤. SAS executes the program until the value of the specified variable changes. For more information, see “Watching a Variable Value” on page 42.

Note: When you use the DATA Step Debugger, note these reminders:

- The DATA Step Debugger is available only in the Standard perspective. For more information, see “Understanding Perspectives” on page 15.
- When you are debugging a DATA step, statements in the DATA step can iterate as many times as they would outside the debugging session. When the last iteration has finished, a message appears in the Debug Console.
- DATA steps that include the DATALINES and CARDS statements cannot be debugged. You can work around this limitation in these ways:
  - Store your data in a flat file, and use the FILENAME statement to read the data into the DATA step.
  - Create two separate DATA steps: one DATA step that includes the DATALINES or CARDS statement and another debuggable DATA step to process the data.
- You can debug only one DATA step at a time. You can use the DATA Step Debugger only with a DATA step and not with a PROC step.
Using the DATA Step Debugger

About the DATA Step Debugger Window

The DATA Step Debugger window includes many features that make it easier to debug your DATA step program:

- The currently executing line is highlighted in blue.
- The variables pane on the right side of the window lists each variable in the program with its current value.
- You can use the filter and search box to narrow the list of variables in the variables pane.
- As you step through a program, variable values that change are displayed in red.
- The Debug Console at the bottom of the window displays logging information as you execute the program.
- You can enter debugger commands directly on the command line at the bottom of the window. To view a list of valid commands, click ⬇️.

Opening the DATA Step Debugger

You can run the DATA Step Debugger on any program that contains a DATA step. To get started using the DATA Step Debugger, you must first enable the DATA Step Debugger and select the valid DATA step code that you want to debug.
1. Open your program in the Program Editor and click **Debug** on the toolbar. All sections of DATA step code in the program are highlighted with a green bar in the margin to indicate that they can be debugged.

2. Click in the margin beside the section of code that you want to debug. The DATA step code opens in the DATA Step Debugger window.

   Note: You can also open the DATA Step Debugger window by right-clicking in the DATA step code and selecting **Debug DATA step**.
Setting and Clearing Breakpoints

You can set or clear a breakpoint on any executable line in the program. A breakpoint suspends execution of the program at the specified line. Each time you set or clear a breakpoint, information about the breakpoint is displayed in the Debug Console at the bottom of the window.

To set and clear breakpoints:

- You can toggle a breakpoint on and off by selecting the line and clicking ■ on the toolbar.
- To clear all breakpoints in the program, click •.
Executing the Program Step by Step

You can execute the statements in the program one at a time and view the variable values with each step. As you step through your program, any variable values that change are displayed in red.

To execute the statement starting at the point at which execution was suspended, click 🔄. Only one statement is executed at a time.

Starting and Stopping Execution of the Program

When you start execution of the program in the DATA Step Debugger, the program runs until it reaches a breakpoint or until the value of a watched variable changes. If no breakpoints or watched variables are specified, the program runs to completion.

- To start or continue execution of the program, click ⏯.
- To stop execution of the program, click ☐.

Note: You can specify a line in the program at which to restart execution by right-clicking the appropriate line in the program and selecting Jump.

Inspecting Variable Values

As you debug a program, you can view the values of each variable in the program. You can view the complete list of variables in the variables pane, and you can use the filter and search box to narrow the list of variables. Variable values that have changed are displayed in red.
Watching a Variable Value

You can watch any variable in the program. SAS suspends execution of the program whenever the value of a watched variable changes. Variable values that have changed are displayed in the variables pane in red.

To watch a variable, select the check box for the variable in the variables pane.
Debugging with the Command Line

You can use the command line to enter debugger commands directly. To view a list of valid commands, click 🕵️‍♂️.

**Note:** For more information about DATA Step debugger commands, see “Dictionary of DATA Step Debugger Commands” in *SAS Code Debugger and SAS DATA Step Debugger: User’s Guide*.

For example, you can use the command line to set a conditional breakpoint so that execution of the program is stopped when a specified condition is met. The following
example uses this debugger command to set a breakpoint at line 11 when the value of the variable \_n\_ is 10:

\texttt{break 11 when \_n\_ eq 10}

A breakpoint is set at line 11, and the program execution stops when the value of \_n\_ is 10.

## Working with Snippets

### Why Use Snippets?

Snippets are lines of commonly used code or text that you can save and reuse. Snippets can contain SAS code, XML code, and text. You can insert snippets in...
locations including the Program Editor, XML editor, text editor, job definitions, job forms, and job prompts. SAS Studio is shipped with several code snippets.

<table>
<thead>
<tr>
<th>Snippet Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Provides a template for a DS2 program. DS2 is a SAS programming language that is appropriate for advanced data manipulation. DS2 is included with Base SAS and shares core features with the SAS DATA step. DS2 exceeds the DATA step by adding variable scoping, user-defined methods, ANSI SQL data types, and user-defined packages. The DS2 SET statement accepts embedded FedSQL syntax, and the run-time-generated queries can exchange data interactively between DS2 and any supported database. This allows SQL preprocessing of input tables, which effectively combines the power of the two languages. For more information, see SAS DS2 Language Reference.</td>
</tr>
<tr>
<td>DS2 Code</td>
<td>Provides a template for a DS2 package. A package is similar to a DS2 program. The package body consists of a set of global declarations and a list of methods. The main syntactical differences are the PACKAGE and ENDPACKAGE statements. These statements define a block with global scope. For more information, see SAS DS2 Language Reference.</td>
</tr>
<tr>
<td>DS2 Package</td>
<td>Provides a template for a DS2 threaded program. Typically, DS2 code runs sequentially. That is, one process runs to completion before the next process begins. It is possible to run more than one process concurrently, using threaded processing. In threaded processing, each concurrently executing section of code is said to be running in a thread. For more information, see SAS DS2 Language Reference.</td>
</tr>
<tr>
<td>Generate CSV File</td>
<td>Creates a CSV version of a SAS data set in the file that is specified in the FILENAME statement. The file name should be fully qualified and end with .csv.</td>
</tr>
<tr>
<td>Generate PowerPoint Slide</td>
<td>Uses PROC SGPLOT to create Microsoft PowerPoint output in the file that is specified in the FILENAME statement. The file name should be fully qualified and end with .ppt.</td>
</tr>
<tr>
<td>Generate XML File</td>
<td>Creates an XML version of a SAS data set in the directory that is specified in the FILENAME statement. The directory name should be fully qualified.</td>
</tr>
<tr>
<td>Import CSV File</td>
<td>Imports a comma-separated file from a SAS Drive directory and writes the output to a SAS data set.</td>
</tr>
<tr>
<td>Snippet Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Import XLSX File</td>
<td>Imports a Microsoft XLSX file from a SAS Drive directory and writes the output to a SAS data set.</td>
</tr>
<tr>
<td>Simulate Linear Regression Data</td>
<td>Creates an input data source that you can use for linear regression analysis. Linear regression analysis tries to assign a linear function to your data by using the least squares method.</td>
</tr>
<tr>
<td>Simulate One-Way ANOVA Data</td>
<td>Creates an input data source that considers one treatment factor with three treatment levels. When you analyze this data by using the One-Way ANOVA task, the goal is to test for differences among the means of the levels and to quantify these differences.</td>
</tr>
</tbody>
</table>

**Descriptive**

<table>
<thead>
<tr>
<th>Custom ODS Output</th>
<th>Provides a template for creating HTML, PDF, and RTF output by using the SAS Output Delivery System. For more information, see <em>SAS Output Delivery System: User’s Guide</em>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROC SQL</td>
<td>Provides a template for writing SQL queries. For more information, see <em>SAS SQL Procedure User’s Guide</em>.</td>
</tr>
</tbody>
</table>

**Graph**

*Note:* For more information about the SGPLOT, SGPANEL, and SGSCATTER procedures, see *SAS ODS Graphics: Procedures Guide*.

<table>
<thead>
<tr>
<th>Bar Panel</th>
<th>Uses the VBAR statement in the SGPANEL procedure and enables you to create multiple bar charts.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Box Panel</td>
<td>Uses the VBOX statement in the SGPANEL procedure and enables you to create multiple box plots.</td>
</tr>
<tr>
<td>Comparative Scatter Plot</td>
<td>Uses the COMPARE statement in the SGSCATTER procedure. This code snippet creates a comparative panel of scatter plots with shared axes.</td>
</tr>
<tr>
<td>Dot Plot</td>
<td>Uses the DOT statement in the SGPLOT procedure. Dot plots summarize horizontally the values of a category variable. By default, each dot represents the frequency for each value of the category variable.</td>
</tr>
<tr>
<td>Fit Plot</td>
<td>Uses the REG statement in the SGPLOT procedure. This code snippet produces a regression plot with a quadratic fit and includes confidence limits.</td>
</tr>
<tr>
<td>Snippet Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>HBar Plot</td>
<td>Uses the HBAR statement in the SGPLOT procedure. This code snippet creates a horizontal bar chart that summarizes the values of a category variable.</td>
</tr>
<tr>
<td>HighLow Plot</td>
<td>Uses the HIGHLow statement in the SGPLOT procedure. High-low charts show how several values of one variable relate to one value of another variable. Typically, each variable value on the horizontal axis has several corresponding values on the vertical axis.</td>
</tr>
<tr>
<td>Histogram Plot</td>
<td>Uses the HISTOGRAM statement in the SGPLOT procedure. This code snippet produces a histogram with two density plots. In this snippet, one density plot uses a normal density estimate and the other density plot uses a kernel density estimate.</td>
</tr>
<tr>
<td>Scatter Plot Matrix</td>
<td>Uses the MATRIX statement in the SGSCATTER procedure. This code snippet creates a scatter plot matrix.</td>
</tr>
<tr>
<td>VBox Plot</td>
<td>Uses the VBOX statement in the SGPLOT procedure. A box plot summarizes the data and indicates the median, upper and lower quartiles, and minimum and maximum values. The plot provides a quick visual summary that easily shows center, spread, range, and any outliers. The SGPLOT and the SGPANEL procedures have separate statements for creating horizontal and vertical box plots.</td>
</tr>
</tbody>
</table>

**Note:** These snippets are available only if your site licenses SAS/IML.

<table>
<thead>
<tr>
<th>Snippet Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find Roots of Nonlinear Equation</td>
<td>Enables you to find the roots of a function of one variable. Finding the root (or zero) of a function enables you to solve nonlinear equations.</td>
</tr>
<tr>
<td>Fit by using Maximum Likelihood</td>
<td>Uses maximum likelihood estimation to estimate parameters for the normal density estimate.</td>
</tr>
<tr>
<td>Generate a Bootstrap Distribution</td>
<td>Uses the IML procedure to create and analyze a bootstrap distribution of the sample mean.</td>
</tr>
<tr>
<td>Integrate a Function</td>
<td>Enables you to numerically integrate a one-dimensional function by using the QUAD subroutine in SAS/IML software. Use the QUAD subroutine to numerically find the definite integral of a function on a finite, semi-infinite, or infinite domain.</td>
</tr>
<tr>
<td>Simulate Multivariate Normal Data</td>
<td>Simulates data from a multivariate normal distribution with a specified mean and covariance.</td>
</tr>
<tr>
<td>Snippet Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Macro</td>
<td>Provides a basic template for working with SAS macros. Macros enable you to perform many tasks, including substituting text in a program. A SAS program can contain any number of macros, and you can invoke a macro multiple times in a single program. For more information, see SAS Macro Language: Reference.</td>
</tr>
<tr>
<td>SAS Macro</td>
<td>Provides several examples of these SAS macros that work with character values:</td>
</tr>
<tr>
<td></td>
<td>- The %EVAL function evaluates arithmetic and logical expressions by using integer arithmetic. This function operates by converting its argument from a character value to a numeric or logical expression. After the expression is evaluated, the result is converted back to a character value. This function is useful because the SAS Macro Facility is basically a text generator. As a result, an arithmetic expression is first converted to a numeric expression. After this numeric expression is evaluated, it is converted back to an arithmetic expression.</td>
</tr>
<tr>
<td></td>
<td>- The %INDEX function returns the position of the first character of a string.</td>
</tr>
<tr>
<td></td>
<td>- The %LENGTH function returns the length of a string.</td>
</tr>
<tr>
<td></td>
<td>- The %SCAN function searches for a word that is specified by its position in a string.</td>
</tr>
<tr>
<td></td>
<td>- The %SUBSTR function produces a substring of a character string.</td>
</tr>
<tr>
<td></td>
<td>- The %UPCASE function converts values to uppercase.</td>
</tr>
<tr>
<td>Snippet Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SAS Macro Do Statement</td>
<td>Designates the beginning of a section of a macro definition that is treated as a unit until a matching <code>%END</code> statement is encountered. This macro section is called a <code>%DO</code> group. A simple <code>%DO</code> statement often appears in conjunction with <code>%IF-%THEN-%ELSE</code> statements to designate a section of the macro to be processed depending on whether the <code>%IF</code> condition is true or false. Note: SAS also provides a <code>%DO</code> iterative statement, which is different from the code that is generated by this snippet. For more information, see <em>SAS Macro Language: Reference</em>.</td>
</tr>
<tr>
<td>SAS Macro If Statement</td>
<td>Conditionally processes a portion of a macro. The expression that is the condition for the <code>%IF-%THEN-%ELSE</code> statement can contain only operands that are constant text or text expressions that generate text.</td>
</tr>
<tr>
<td>SAS Macro Parameters</td>
<td>Names one or more local macro variables whose values you specify when you invoke the macro. There are two types of macro variables: positional and keyword. Parameters are local to the macro that defines them. You must supply each parameter name. You cannot use a text expression to generate it. A parameter list can contain any number of macro parameters separated by commas. The macro variables in the parameter list are usually referenced in the macro.</td>
</tr>
<tr>
<td>Snippet Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SAS Macro Quoting</td>
<td>Provides examples of macro functions that tell the macro processor to interpret special characters and mnemonics as text rather than as part of the macro language.</td>
</tr>
<tr>
<td></td>
<td>- The %STR function masks special characters and mnemonic operators in constant text at macro compilation. This function masks these special characters and mnemonic operators: + - * / &lt; &gt; = ¬ ^ ~ ; , # blank AND OR NOT EQ LE LT GE GT IN</td>
</tr>
<tr>
<td></td>
<td>This function also masks these characters when they occur in pairs and when they are not matched and are marked by a preceding %: ' &quot; ()</td>
</tr>
<tr>
<td></td>
<td>- The %NRSTR function masks special characters and mnemonic operators in constant text at macro compilation. This function masks all of the special characters and mnemonic operators listed for the %STR function. In addition, the %NRSTR function masks these characters: &amp; %</td>
</tr>
<tr>
<td></td>
<td>- The %BQUOTE function masks special characters and mnemonic operators in a resolved value at macro execution. This function masks these special characters and mnemonic operators: ' &quot; () + - * / &lt; &gt; = ¬ ^ ~ ; , # blank AND OR NOT EQ LE LT GE GT IN</td>
</tr>
<tr>
<td></td>
<td>- The %SUPERQ function masks all special characters and mnemonic operators at macro execution but prevents further resolution of the value. This function masks these special characters and mnemonic operators: &amp; % ' &quot; () + - * / &lt; &gt; = ¬ ^ ~ ; , # blank AND OR NOT EQ LE LT GE GT IN</td>
</tr>
<tr>
<td></td>
<td>- The %QSCAN function searches for a word and masks special characters and mnemonic operators.</td>
</tr>
<tr>
<td></td>
<td>- The %QSUBSTR function produces a substring and masks special characters and mnemonic operators.</td>
</tr>
<tr>
<td></td>
<td>- The %QUPCASE function converts a value to uppercase and returns a result that masks special characters and mnemonic operators.</td>
</tr>
<tr>
<td></td>
<td>- The %UNQUOTE function unmasks a value during macro execution so that any special characters and mnemonic operators are interpreted as macro language elements instead of text.</td>
</tr>
</tbody>
</table>

For more information about macro complication and macro execution, see *SAS Macro Language: Reference*. 
<table>
<thead>
<tr>
<th>Snippet Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAS Macro Variables</td>
<td>Provides examples of how to create user-defined global and local macro variables. Macro variables are tools that enable you to dynamically modify the text in a SAS program through symbolic substitution. You can assign large or small amounts of text to macro variables. Then you can use that text by simply referencing the variable that contains the text. Macro variables that are defined by the macro programmer are called user-defined macro variables. Macro variables that are defined by the macro processor are called automatic macro variables. You can define and use macro variables anywhere in SAS programs, except within data lines.</td>
</tr>
<tr>
<td>SAS Viya Cloud Analytic Services</td>
<td></td>
</tr>
<tr>
<td>Create CAS Connection</td>
<td>Creates a connection to a CAS server. You must specify values for the CASHOST= and CASPORT= system options. The CAS statement connects the default session to the specified CAS server and CAS port.</td>
</tr>
<tr>
<td>New CAS Session</td>
<td>Starts a new CAS session named mySession using the existing CAS server connection. When starting a new session, you can specify the CAS library to use, the time-out (in seconds) for the session, and the locale of the session.</td>
</tr>
<tr>
<td>Disconnect CAS Session</td>
<td>Disconnects from the CAS session named mySession. Before you disconnect, you can specify a value for the time-out (in seconds). You can reconnect to the session before the time-out expires. Otherwise, the session is terminated.</td>
</tr>
<tr>
<td>Reconnect CAS Session</td>
<td>Reconnects to a CAS session named mySession.</td>
</tr>
<tr>
<td>Terminate CAS Session</td>
<td>Terminates the CAS session named mySession. No reconnection is possible.</td>
</tr>
<tr>
<td>List CAS Session Options</td>
<td>Lists session options for the specified CAS session.</td>
</tr>
<tr>
<td>List CAS Sessions for SAS Client</td>
<td>Lists all the CAS sessions and session properties that are created by the SAS client or reconnected to by the SAS client.</td>
</tr>
<tr>
<td>List CAS Sessions for User ID</td>
<td>Lists all the CAS sessions that are known to the CAS server for the user ID that is associated with yourSessionName.</td>
</tr>
<tr>
<td>Snippet Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>New caslib for Path</td>
<td>Creates a CAS library (myCaslib) for the specified path (/filePath/) and session (mySession). If you omit the SESSREF= option, the caslib is created and activated for the current session. The SUBDIRS option extends the scope of myCaslib to include subdirectories of (/filePath/). The LIBNAME statement creates a SAS libref for the CAS library.</td>
</tr>
<tr>
<td>Generate SAS librefs for caslibs</td>
<td>Creates a default CAS session and generates SAS librefs for existing CAS libraries so that the librefs are visible in the Libraries pane.</td>
</tr>
<tr>
<td>Save Table to caslib</td>
<td>Creates a permanent copy of an in-memory table (table-name) from sourceCaslib. The in-memory table is saved to the data source that is associated with the target caslib targetCaslib using the specified name (filename). Note: You can determine the caslib that is associated with a CAS engine libref by right-clicking the libref in the Libraries pane and selecting Properties. The Server Session CASLIB field displays the caslib.</td>
</tr>
<tr>
<td>Load Data to caslib</td>
<td>Loads a table to the specified CAS library. The snippet includes the PROMOTE option so that the loaded data is available to all your active sessions.</td>
</tr>
<tr>
<td>Delete Table or File from caslib</td>
<td>Deletes a table or file from the CAS library. You can also remove an in-memory table.</td>
</tr>
<tr>
<td>Delete caslib</td>
<td>Deletes the specified CAS library.</td>
</tr>
<tr>
<td>Load Data</td>
<td>Demonstrates how to load local data into CAS.</td>
</tr>
<tr>
<td>Prepare and Explore Data</td>
<td>Modifying, and preparing data prior to modeling. This snippet uses the Hmeq data set from the Machine Learning sample data library. The Hmeq data set is used as input and creates the Hmeq_prepped data set. The Hmeq_prepped data set is used in subsequent examples. Note: You might see a warning message about missing values when you run this snippet. The sample data includes some missing values that can be used by other procedures.</td>
</tr>
<tr>
<td>Snippet Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Compare Two ML Algorithms</td>
<td>Demonstrates fitting and comparing two Machine Learning algorithms for predicting the binary target in the Hmeq data set, which is included in the Machine Learning sample data library. You must run the Load Data and Prepare and Explore Data snippets before you run this snippet.</td>
</tr>
<tr>
<td>Compare Several ML Algorithms</td>
<td>Demonstrates fitting and comparing several Machine Learning algorithms for predicting the binary target in the Hmeq data set, which is included in the Machine Learning sample data library. You must run the Load Data and Prepare and Explore Data snippets before you run this snippet.</td>
</tr>
<tr>
<td>Generalized Linear Models</td>
<td>Demonstrates fitting and assessing generalized linear models using the GENSELECT procedure. You must run the Load Data and Prepare and Explore Data snippets before you run this snippet.</td>
</tr>
<tr>
<td>Unsupervised Learning</td>
<td>Shows the entire workflow process, including data preparation, analysis, and visualization of the results. This snippet uses the Sashelp.Iris data set.</td>
</tr>
<tr>
<td>Supervised Learning</td>
<td>Shows the entire workflow process, including data exploration and preparation, modeling, and evaluation. This snippet uses the Sampleml.Hmeq data set, which is included in the Machine Learning sample data library.</td>
</tr>
<tr>
<td>SAS Viya Image Processing</td>
<td>You must license and install SAS Visual Data Mining and Machine Learning to use these snippets. For more information, see SAS Visual Statistics: Procedures.</td>
</tr>
</tbody>
</table>
| Load Images                      | Loads the image action set from the specified path and creates a CAS table. The path parameter points to the directory that contains the images that you want to load. The path parameter is a subdirectory in the caslib directory where the images are stored. 
A typical workflow includes loading images and then processing them by using one of the other snippets to resize, rescale, or mutate them. The output from one snippet can be used as input for another snippet. Images can be saved to your computer for further processing by using the Save Images snippet. |
<table>
<thead>
<tr>
<th>Snippet Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resize Images</td>
<td>Resizes one or more images based on the HEIGHT and WIDTH parameters that you specify. The HEIGHT parameter corresponds to the number of rows, and the WIDTH parameter corresponds to the number of columns. The input is a CAS table that contains the images that you want to resize, and the output is a CAS table that contains the resized images.</td>
</tr>
<tr>
<td>Rescale Images</td>
<td>Changes the depth of one or more images based on the options that you specify. You can specify any of these values for the TYPE parameter: “TO_8U”, “TO_32F”, or “TO_64F”. You can also use the ALPHA and BETA parameters to scale the values. The input is a CAS table that contains the images that you want to rescale, and the output is a CAS table that contains the rescaled images.</td>
</tr>
<tr>
<td>Mutate Images</td>
<td>Mutates one or more images using different augmentation techniques. You can specify any of these values for the TYPE parameter: ‘COLOR_JITTERING’, ‘COLOR_SHIFTING’, ‘DARKEN’, ‘HORIZONTAL_FLIP’, ‘INVERT_PIXELS’, ‘LIGHTEN’, ‘ROTATE_LEFT’, ‘ROTATE_RIGHT’, ‘SHARPEN’, or ‘VERTICAL_FLIP’. The input is a CAS table that contains the images that you want to change, and the output is a CAS table that contains the mutated images.</td>
</tr>
<tr>
<td>Convert Color</td>
<td>Converts the color space of one or more images. You can specify any of these values for the TYPE parameter: ‘BGR2HSV’, ‘BGR2RGB’, ‘BGR2YUV’, ‘COLOR2GRAY’, ‘GRAY2COLOR’, ‘HSV2BGR’, ‘RGB2BGR’, or ‘YUV2BGR’. The input is a CAS table that contains the images that you want to convert, and the output is a CAS table that contains the converted images.</td>
</tr>
<tr>
<td>Save Images</td>
<td>Saves the specified images to the specified subdirectory of a previously defined caslib, such as CASUSER. You must specify the name of a subdirectory in the caslib root directory. The names of the saved images start with the value of the PREFIX parameter.</td>
</tr>
<tr>
<td>Display Images</td>
<td>Displays an image that is saved from SAS Studio to your computer. This snippet creates an annotation data set that contains a reference to the image and uses PROC SGPLOT to display the image. Use the IMAGE= variable to specify the fully qualified name and location of the image that you want to display. The SAS server must be able to access this location.</td>
</tr>
</tbody>
</table>
Create a Snippet

To create your own snippet:

1. Open a file in SAS Studio and select the lines or code or text that you want to save as a snippet. If you do not select any lines of code or text, the entire file is saved as a snippet.

2. On the toolbar, click **Copy to My Snippets**. The Copy to My Snippets window appears.

   **Note:** You can also create a snippet in the following ways:
   - Right-click in the file and select **Copy to My Snippets**.
   - Right-click an existing snippet in the Snippets section in the navigation pane and select **Copy snippet to My Snippets** or **Copy snippet to Explorer**. If you copy the snippet to the Explorer section in the navigation pane, the snippet is saved as a *.csm file in the folder that you specify.

3. Enter a name for the snippet.

4. If you want to quickly access the snippet by entering @snippet-abbreviation in the Program Editor, enter an abbreviation for the snippet. You can also add a description.

5. Select the folder in which you want to save the snippet and click **OK**. The snippet is added to the folder that you specified.

   **Note:** You can also upload existing snippet files to your My Snippets folder and subfolders. The file type must be *.csm. Select the folder that you want to use and click **Upload**.

   To add a saved *.sas file to your My Snippets folder, right-click the .sas file in the Explorer section in the navigation pane and select **Copy to My Snippets**.

How to Insert a Code Snippet

To insert a snippet:

1. Click the location where you want to insert the snippet.

2. In the navigation pane, click **Snippets** to open the Snippets section.

3. You can add a snippet to your file in these ways:
   - use a drag-and-drop operation to insert the snippet into your file.
   - right-click the name of the snippet and select **Insert**.
   - right-click in the file and select **Insert snippet**, and then select the snippet that you want to use.
Enter `@snippet-name` in your code and press Enter. You can also enter `@` and use the pop-up window to select from a list of snippets in your My Snippets folder.

---

**Note:** In order to use this functionality, you must specify an abbreviation for the snippet. To verify whether your snippet has an abbreviation, select the snippet and click **Edit**. To add or edit the abbreviation, right-click the snippet and select **Edit**. For more information, see “Create a Snippet” on page 55. You must also select the **Enable snippet abbreviations** preference. For more information, see “Setting Editors Preferences” on page 150.

---

**Customizing the Code Editor**

The Preferences window enables you to change several options that affect the features in the code editor, including autocompletion and color coding.

To access the editor options, select **Options → Preferences**. Click **Editors**.

For more information, see “Setting Editors Preferences” on page 150.
What Is a Query?

A query enables you to extract data from one or more tables according to criteria that you specify. You can create a query that is based on only one table, or you can join tables. When you create a query, SAS Studio generates Structured Query Language (SQL) code, which you can view and edit. You can also choose to generate your query using the FEDSQL procedure. For more information, see “Generating a FedSQL Query” on page 79.
Creating a Query

On the main SAS Studio toolbar, select New ⇒ Query. A new query tab opens in the work area.

Note: You can also create a query by right-clicking on one or more tables in the Libraries section in the navigation pane and selecting Create query.

By default, the query tab contains three main areas:

- **Columns area**: provides access to the tables and columns in your query.
- **Query definition area**: enables you to specify and group columns in the query, create joins between tables, filter the data in the query, sort the output, and specify output options.
- **Code, Log, and Results area**: the Code tab displays the code that is automatically generated as you build the query. The Log tab displays the query log when you run the query. If your output type is Table or View, an Output Data tab is added to this area when you run the query. If your output type is Report, a Results tab is added to this area when you run the query.
Adding Tables to a Query

In the Columns area of the query tab, click Add. Use the Select Table window to browse for the library and table that you want to use, and click OK. The table is added to the query.

Note: You can also add tables to the query by using the Libraries section in the navigation pane. Click in the navigation pane to open the Libraries section and expand the appropriate library. Drag the table that you want to use to the Columns area of the query tab.

Understanding Joins

Joining Tables

When you create a query, you can join multiple tables. SAS Studio can automatically join the tables for you, or you can manually create the join. SAS Studio attempts to join tables by columns that have the same name and type. If no matches for column name and type are found, the tables are joined using the first
column in each table. If this is not the correct join for your query, you can edit the join and specify the join criteria that you want to use.

Note: If you have more than one table in your query and you do not specify any join criteria, then your output data includes the Cartesian product, or every possible combination, of the data values.

Creating a Join

When you add a table to an existing table in your query, SAS Studio automatically creates a join for you.

To add another table to your query, click Add in the Columns area of the query tab. Use the Select Table window to browse for the library and table that you want to use and click OK. You can also drag a table from the Libraries section in the navigation pane to the Columns area.

Click the Join tab to view the join criteria. In the following example, the Classfit table is automatically joined to the Class table by using the Name column in both tables.
If a suitable join cannot be created automatically, you can specify the join condition manually. For more information, see “Modifying an Existing Join” on page 62.

Understanding the Types of Joins

SAS Studio supports four different types of joins. You can select the type of join you want by modifying an existing join.

You can select the join option that you want to use in the Join window.

<table>
<thead>
<tr>
<th>SAS Studio Join Type</th>
<th>Join Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inner join</td>
<td>![Inner join icon]</td>
<td>The output rows include only those for which the column in the first table matches the joining criterion of the column in the second table. Joins are inner joins by default.</td>
</tr>
<tr>
<td>Left join</td>
<td>![Left join icon]</td>
<td>The output rows include all rows from the first table and the rows from the second table in which the joining criterion is met.</td>
</tr>
</tbody>
</table>
### SAS Studio Join Type

<table>
<thead>
<tr>
<th>SAS Studio Join Type</th>
<th>Join Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right join</td>
<td>![Right Join Icon]</td>
<td>The output rows include all rows from the second table and the rows from the first table in which the joining criterion is met.</td>
</tr>
<tr>
<td>Full join</td>
<td>![Full Join Icon]</td>
<td>The output rows include all matching and nonmatching rows from both tables.</td>
</tr>
</tbody>
</table>

### Modifying an Existing Join

You can modify an existing join by selecting a different type of join or by changing the columns that are used in the join condition. You can also add and remove join conditions or remove the entire join.

**Note:** You must have at least one join and one join condition for each additional table that you add to the query.

To modify a join:

1. Click the **Join** tab of the query window. You can edit the join in the following ways:
   - Use the table drop-down lists to change the tables that are used in the join.
   - Click the join indicator between the table names to edit the join type. The default join type is **Inner join**. For more information, see “Understanding the Types of Joins” on page 61.
   - Click the operator between the column names to select a new operator. The default operator is =.
   - Click on each side of the join to change the columns that are used in the join.

2. To add a new join condition, position the mouse pointer over the existing join condition, and click and specify the columns to use in the join. To remove a join condition, position the mouse pointer over the join condition, and click next to the appropriate condition.

**Note:** If the join has only one join condition, you cannot delete the join condition.
Selecting Data

Specifying Columns in the Output

By default, no columns are included in the output. You must specify the columns that you want to appear in the output table. You can also specify an alias to use in place of the column name in the output table.

The order in which the columns are listed on the Select tab is the order in which they appear in the output table.

To select columns for the output table:

1. In the Columns area of the query tab, expand the tables and drag the columns that you want to include in the output to the Columns subtab of the Select tab.
2. You can add one or more columns to the output data by dragging them from the columns list to the Select tab.

**Note:** To add all of the columns in a table to the query, drag the table name to the Select tab. You can also right-click the table in the Columns area and select Select all columns.

To specify an alias for a column:

- On the Select tab, use the Column Name column to specify any aliases that you want to use for the columns. The alias is used as the column heading for the output data.
Creating a Calculated Column

You can insert a new column into your query that is calculated from other columns or values. You can use a calculated column to summarize values, replace values based on a condition, and perform string and numeric calculations. After you create a calculated column, you can use the column to sort or group your output.

To create a calculated column:

1. In the Columns area of the query tab, click Add ⇒ Calculated column.

2. Use the expression builder in the Calculated Column window to create your column. For more information, see “Building an Expression” on page 157.

3. After you create the expression for the column, use the Properties tab in the expression builder window to specify a name for the column. You can also specify a label, length, and format.

4. Click OK to create the calculated column.
Using Summary Functions

You can perform aggregation, or summary, functions on any of the columns in your query. To perform an aggregation function, select the column on which you want to perform an aggregation function. Use the drop-down list in the Aggregate column to select the function that you want to use. By default, the query generates an output data set. The following example shows you how to find the average age of all of the students:
By default, the query displays the results on the **Output Data** tab and generates an output table in the Work library.
By default, when you aggregate a column, your output is grouped by all of the columns without aggregation. For more information, see “Grouping Your Output” on page 73.

Working with Conditional Expressions

Creating a Conditional Expression

You can create a conditional expression in your query by creating a calculated column that uses the CASE function. A conditional expression enables you to use logic similar to "IF-THEN/ELSE" to return values that can be used to create a column.

Note: You cannot create conditional expressions if you are generating a FedSQL query. For more information, see “Generating a FedSQL Query” on page 79.

For example, you could create an expression to extract the first letter of a last name and replace last names starting with "A" through "F" with the value "A - F," replace last names starting with "G" through "L" with the value "G - L," and so on.
To create a conditional expression:

1. In the Columns area of the query tab, click **Add** ⇒ **Calculated column**.

2. Use the following template for the CASE expression syntax to enter your own CASE expression in the expression box:

   ```
   CASE <CaseOperand>
   WHEN <whenCondition> THEN <resultExpression>
   WHEN <whenCondition> THEN <resultExpression>
   ELSE <resultExpression>
   END
   ```

   - **<CaseOperand>** – the name of the column that you are using in the expression.
   - **<whenCondition>** – the existing value of the column.
   - **<resultExpression>** – the value that you want to assign to the column.

3. After you create the expression for the column, use the **Properties** tab in the lower half of the expression builder window to specify a name for the column. You can also specify a label, length, and format.

---

**Note:** For more information, see “CASE Expression” in *SAS SQL Procedure User’s Guide*.

---

### Sample Conditional Expressions

The following example uses the SUBSTR function to extract the first letter of the last name, and then assign a replacement value based on that letter.

```sql
CASE WHEN (SUBSTR(tablename.LastName,1,1) between "A" and "F") THEN "A - F"
    WHEN (SUBSTR(tablename.LastName,1,1) between "G" and "L") THEN "G - L"
    WHEN (SUBSTR(tablename.LastName,1,1) between "M" and "R") THEN "M - R"
    WHEN (SUBSTR(tablename.LastName,1,1) between "S" and "Z") THEN "S - Z"
END
```

The following example replaces age values with "Under 35," "35 - 50," and "Over 50."

```sql
CASE
    WHEN tablename.Age < 35 THEN "Under 35"
    WHEN tablename.Age between 35 and 50 THEN "35 - 50"
    ELSE "Over 50"
END
```
Filtering Data

Creating a Filter

When you query data, you might want to retrieve only rows that meet certain criteria, based on values of columns in the data. The process of telling SAS Studio which rows to retrieve is called setting a filter and is done on the Filter tab. This process corresponds to using a WHERE clause in an SQL query.

1. Click the Filter tab, and then drag one or more columns from the Columns area.

   Note: It is possible to filter the output table by columns that are not selected for the output.

2. In the filter window, select a comparison operator from the operator drop-down list. The default value is Equal.
3 If the operator that you have selected requires a value, enter or select a value in the **Value** box. To choose from a list of values, click **Get Values** and click **Get Values** in the values window. Select the values that you want to use and click **OK**.

**Note:** If you want to search for a value in the value window, use the unformatted value.

4 Depending on the data type of the column that you are using in the filter, you can choose from among 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. If you are using a numeric calculated column, you should select this option.

- **Quote string** – encloses values in single quotation marks. This option is selected by default. If you are using a numeric calculated column or 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.

5 Click **Filter** to add the values to the filter.

---

**Changing the Relationship between Filter Expressions**

You can use only one column in a filter expression, but you can use multiple columns to create several expressions. If you create more than one comparison expression in your filter, the default relationship between these filter elements is **AND**. You can change the relationship between filter elements from **AND** to **OR**.

To change the relationship between filters:

- On the **Filter** tab, click the **Operator** value and select a new value.
Managing Output

Sorting Your Output

You can sort the output from your query by one or more columns from the tables that are used in the query.

**Note:** It is possible to sort the output table by columns that are not selected for the output.

**Note:** You cannot sort the output from your query if you are generating a FedSQL query. For more information, see “Generating a FedSQL Query” on page 79.

To sort your output:

1. Click the **Sort** tab and drag one or more columns from the Columns area.

2. Click the **Sort** box for the column on which you want to sort the data. From the drop-down list, select **Ascending** or **Descending**. The default sort direction is **Ascending**.
If you are sorting by multiple columns, the output table is sorted first by the column that is listed first. Within each level of the first column, the rows are sorted by the second column in the list, and so on. You can change the sort order by selecting a column and clicking ⬆ and ⬇ to move the column up and down the list.

### Eliminating Duplicate Rows in Output

Some types of queries generate multiple, identical rows. Because these duplicate rows are generally not useful, SAS Studio enables you to keep only one of the identical rows and eliminate the duplicates.

To eliminate duplicate rows, click `distinct rows only` in the Columns area and select the **Select distinct rows only** check box.

### Grouping Your Output

If you have created a summarized column, you can choose to classify your data into groups based on the values in a column. This action is equivalent to using the GROUP BY clause in an SQL query. For example, if you are calculating the average height of a group of students, you might want to group the results by age so that you can see the average height for each age group.

By default, the **Automatically group data** option is selected on the **Groups** subtab of the **Select** tab. When this option is selected and you have performed a summary function on a column, your query is automatically grouped by all columns without
summary functions. You can choose to edit the list of columns that the query is grouped by.

To group your output:

1. Click the Groups subtab of the Select tab and drag one or more columns from the Columns area.

   Note: It is possible to group your output by columns that are not selected for the output.

   Note: When you add columns to the Groups subtab, the Automatically group data option is automatically cleared. To remove the columns that you have added to the Groups subtab, select the Automatically group data option again. You can also remove individual columns by selecting the column that you want to remove and clicking .

2. To change the order in which the columns are used to group the data, select the column that you want to move and click ↑ and ↓.

   The following example shows you how to find the average weight of students in each age group. First, add the Age and Weight columns to the Select tab, and then select the AVG summary function for the Weight column.
To see the average weight of students by age, the query is grouped by the Age column. The results show the average weight for each age group.
Subsetting Grouped Data

If you have created a summarized column or a calculated column that contains a summary function, you can choose to classify your data into groups based on the values in a column. When you group your output by one or more columns, you can create a filter to subset the grouped data based on specified conditions. This action is equivalent to using the HAVING expression in an SQL query.

Note: You can also choose to filter the raw, unsummarized data. For more information, see “Creating a Filter” on page 70.

You can use one column in a filter, or you can use multiple columns to create several comparison expressions. If you create more than one comparison expression in your filter, the default relationship between these filter elements is AND. You can change the relationship between filter elements from AND to OR, and you can group elements together.
You can choose to create either a basic filter or you can create an advanced filter by using the expression builder:

1 Click the **Filter Groups** subtab of the **Select** tab and drag one or more columns from the Columns area.

   Note: It is possible to subset your grouped data by columns that are not selected for the output.

2 In the filter window, select a comparison operator from the operator drop-down list. The default value is **Equal**.

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

4 Depending on the data type of the column that you are using in the filter, you can choose from among the following options:
   - **Match case** – retrieves only rows that match the capitalization of the value that you specify.
   - **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.

5 Click **Filter** to add the filter expression to the query.

   Note: If you create more than one filter expression, you can change the relationship between the expressions. For more information, see “Changing the Relationship between Filter Expressions” on page 71.

---

**Saving Your Results**

You can choose to generate your results in any one of three formats: report, data table, or data view.

If you save your results as a data table or data view, you can specify the library and table name that you want to use. If you do not specify the library and table name, the results are saved in the Work library.

To specify the results format:

1 In the query window, click the **Output Options** tab.

2 Select the format that you want to use from the **Output type** drop-down list.
   - **Report** saves the query results as a report that you can download as an HTML, PDF, or RTF file. Query results in this format are not updated until you rerun the query. You cannot run tasks against query results in this format.
Table saves the query results as a static data table against which you can run tasks. Query results in this format are not updated until you rerun the query. By default, the data table is stored in the Work library.

View saves the query results as a dynamic data view against which you can run tasks. Each time you open query results in the data view format, the results are updated with any changes to the data that is used in the query. By default, the data view is stored in the Work library.

---

**Note:** This option is not available if you are generating a FedSQL query. For more information, see “Generating a FedSQL Query” on page 79.

---

To save your output data to a specific location:

1. On the query tab, click the **Output Options** tab.
2. If you are generating a FedSQL query using CAS tables, you must use the **Session Reference** option to specify the name of the default CAS session to use. For more information, see “Generating a FedSQL Query” on page 79.
3. Enter the name of the library in which you want to save your results in the **Library** box.
4. To specify a name for the output data, enter the name that you want to use in the **Table name** box.

### Specifying Limits for Input Processing

When you create a query, you can restrict the number of rows that your query processes from each source table. This option corresponds to the INOBS= option in PROC SQL and is useful when you want to reduce processing time in order to test a query.

---

**Note:** This option is not available if you are generating a FedSQL query. For more information, see “Generating a FedSQL Query” on page 79.

---

When you set a limit $n$ on the number of rows to process, SAS Studio uses the first $n$ rows from each source table in the query. If you have added a filter to your query, SAS Studio uses the first $n$ rows from each table that match your filter criteria. This option does not apply when the output format is a data view. For more information, see “Saving Your Results” on page 77.

---

**Note:** If you have a computed column in your query data that performs a function on the whole table, such as sum or average, then limiting the number of input rows affects the value that is calculated for that column. The function is performed only on the rows that are processed.

---

To limit the number of rows to process:
Click the Output Options tab. Select Limit number of matching rows to process (INOBS) and specify the number of rows that you want the query to process.

Note: When you limit the number of rows to process and you have joined tables, the maximum number of rows that the query can return is the Cartesian product of the number of rows from each table. For example, if you set the limit to five rows and you join two tables, the maximum number of rows that the query can return is 25 (5 * 5). If you join three tables, the maximum number of rows that the query can return is 125 (5 * 5 * 5).

Specifying Limits for Output Processing

When you create a query, you can restrict the number of rows that your query saves in the output. This option corresponds to the OUTOBS= option in PROC SQL and is useful when you want to restrict the size of your output for testing purposes.

Note: This option is not available if you are generating a FedSQL query. For more information, see “Generating a FedSQL Query” on page 79.

This option does not apply when the output format is a data view. For more information, see “Saving Your Results” on page 77.

Note: When you limit the size of your output, you are not reducing the processing time of the query. The query still processes all of the data in each source table. You are reducing only the size of the output.

Click the Output Options tab. Select Limit number of matching rows to save in output (OUTOBS) and specify the number of rows that you want in your output.

Running a Query

After you specify all the criteria for your query, you can generate your results by clicking Run on the query window toolbar. The output data or results are added to the results area of the query tab.

Generating a FedSQL Query

You can generate your query using FedSQL instead of SQL. FedSQL provides many benefits if you are working in an environment in which you need more features than are provided in the SQL procedure:
FedSQL conforms to the ANSI SQL:1999 core standard.

FedSQL supports many more data types than previous SAS SQL implementations.

FedSQL handles federated queries.

FedSQL language can create data in any of the supported data sources, even if the target data source is not represented in a query.

For more information, see *Overview: PROC FEDSQL*.

To generate a FedSQL query, select Options ⇒ Preferences and click Query. Select Generate FedSQL.

**Note:** This option is applied only to queries that you create after you have selected the option.

The query functionality for creating SQL and FedSQL queries is very similar. The following table lists some of the differences between SQL and FedSQL queries:

<table>
<thead>
<tr>
<th>Query Feature</th>
<th>Available in SQL?</th>
<th>Available in FedSQL?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output type – View</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Sorting output</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Output Options – Session Reference</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Output Options: Limit number of matching rows to process, Limit number of matching rows to save in output</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Operators: Contains, Does not contain, Sounds like, Does not sound like, Any, All</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Functions: CASE, CAT, CATS, CATX, COMPBL, COMPRESS, FIND, INPUT, INT, INTCK, INTNX, LEFT, LENGTH, MDY, N, NLDATE, NLDATM, PROPcase, SCAN, SUBSTR, TRANWRD, WEEK, YRDIF</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Query Feature</td>
<td>Available in SQL?</td>
<td>Available in FedSQL?</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Aggregations: CV, FREQ, MEAN, N, PRT, STD, SUMWGT, T, VAR</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

The following aggregations are available only in FedSQL:
- PROBT (replaces PRT)
- STDDEV (replaces STD)
- STUDENT_T (replaces T)
- VARIANCE (replaces VAR)
About the Table Viewer

The table viewer is used to display your data in SAS Studio. You can click ↓ and ↑ to navigate through the data.

Note: You can click ↓ and ↑ to advance to the last or first page in your data. You cannot navigate directly to the last page of a CAS table, any table with an unknown number of columns, or a data view.
**Note:** By default, the table viewer displays the first 200 rows of the table. If the structure or data values of the table change while the table is open, you must refresh the table viewer to see the changes. If the structure of the table changes and you do not refresh the table, the columns that are listed in the Libraries section of the navigation pane might be different from the columns that are displayed in the table viewer. For more information, see "Refreshing Your Data" on page 91.

You can view the properties of the table and its columns by clicking ☐ on the toolbar of the Libraries section in the navigation pane.
Opening and Viewing Data

You can open files in SAS Studio in several ways:

- You can double-click a file in the Explorer and Libraries sections of the navigation pane.
- You can drag a file from the Explorer and Libraries sections of the navigation pane to the work area.
- You can search for a file and open it by double-clicking it in the search results.
You can open a file by using a file reference in the File References section of the navigation pane. You can open the file by double-clicking it or by dragging it to the work area.

**Note:** You cannot access password-protected data in SAS Studio.

**Note:** In SAS Studio 5.2, files of type *.sas7bdat are not supported in the SAS Content folder.

**Note:** If you open a table that was created with an encoding that has a different byte size than the encoding of your SAS compute server, you must use the CVP (character variable padding) engine in your LIBNAME statement. For more information, see "Avoiding Character Data Truncation by using the CVP Engine" in *SAS National Language Support (NLS): Reference Guide*.

When you open a table, the first 100 columns in the table are displayed by default. To specify which columns you want to include in the table viewer, click on the toolbar and select **Column selection**. You can use the Column Selection window to search for columns and to specify whether to show or hide specific columns in the data. You can also right-click a column in the table viewer and select **Hide** to hide a column.

**Note:** The number of columns that can be displayed is determined by the **Columns displayed** option on the Tables page of the Preferences window. For more information, see “Setting Tables Preferences” on page 153.
By default, the column names are displayed, but you can choose to display the column labels by clicking on the toolbar and selecting **Show labels**.
Note: By default, the table viewer displays the total number of rows in the table and the total number of filtered rows, if you have filtered the data. However, if SAS Studio is unable to determine the row counts without affecting performance, then the row counts are not displayed. Row counts are not available for relational tables, views, and Hadoop data.

You can change the order of the columns in the table viewer by dragging a column to a new position in the table viewer or in the Columns area. You can also sort and freeze the data by one or more columns. For more information, see “Sorting and Freezing Data” on page 91.
Viewing the Code That Is Used to Create a Table

While you select options and customize the table to look the way you want it to, SAS Studio is generating SAS code that you can use. To view the code, click on the toolbar and select **Generate code**. A new program window appears with the code that was used to create the view of the table in the table viewer. The program is a copy of the code and is no longer associated with the original code. Editing the code does not affect the data that is displayed in the table viewer, and modifying the table viewer does not affect the contents of the code.

Opening Data in a Task from the Table Viewer

You can use the table viewer to open a task with the data that you are currently viewing. The task opens with the data automatically selected as the input source for the task. To open data in a task, click on the toolbar and select **Open in a task**.
Refreshing Your Data

You can refresh the table viewer to update the data that is displayed in the table. When you refresh your data, you can choose to either reset any settings, such as filters or sort order, that you specified, or you can keep your settings.

To refresh the data and reset your settings, click $\Rightarrow$ on the table viewer toolbar and select Refresh $\Rightarrow$ Table.

To refresh the data and keep your settings, click $\Rightarrow$ on the table viewer toolbar and select Refresh $\Rightarrow$ Data.

Sorting and Freezing Data

In the table viewer, you can right-click a column heading to sort the data by that column or freeze the data at that column. You can sort the data in ascending or descending alphabetical order or display the columns in the order in which they appear in the data table.
To sort the data, right-click the appropriate column and select **Sort → sort-order**. If the data is already sorted by one or more columns, you can add another column to the sort criteria by right-clicking the column and selecting **Sort → Add to sort sort-order**.

To freeze a column so that column and any columns to the left remain visible while you scroll through the remaining columns, right-click the appropriate column and select **Freeze column here**. To freeze only the selected column and move it to the leftmost position in the table viewer, right-click the column and select **Freeze column**.

---

### Filtering Data

#### About Filtering Data

You can filter your data in order to display only rows that meet certain criteria, based on values in the data. You can filter data in the table viewer either by creating a quick filter based on a single column or by creating a filter expression with one or more columns.

**Note:** If you are creating a filter with the `equals` operator on a numeric column that includes values with a fractional component, you must create the filter by using a range of values to ensure that there are no rounding errors. For example, suppose your data includes a value of 123.45678. To create a filter that displays rows that include that value, you should use the following filter expression:

\[(\text{column-name} \geq 123.456775 \text{ and } \text{column-name} < 123.456785)\]
Note: If you want to search for a value when you are creating a filter, use the unformatted value.

Creating a Quick Filter

You can create a quick filter on a single column in your data by right-clicking the column heading and selecting **Quick Filter**. In the quick filter window, specify the operator and values that you want to use.

**Note:** If you are using the **In a list** or **Not in a list** operator, the values must be separated by spaces.

**Note:** If your filter is based on a formatted date column, you might want to select the **Use raw values** option so that the filter is based on the raw date values instead of the formatted values.
Creating a Filter Expression

You can create a filter expression to filter your data by one or more columns. Click to use the expression builder in the Filter Table Rows window to create a filter expression. For more information, see “Building an Expression” on page 157.
Importing Data

About Importing Data to SAS Studio

You can import these types of data files into SAS Studio:

- delimited files, such as CSV, TXT, TSV, DLM.
- dBASE V, IV, III+, and III (DBF).
- Stata files (DTA).
- Microsoft Excel files (XLS, XLSX). To import XLSX files, you must license and install SAS/ACCESS to PC Files.
- JMP files.
- Paradox DB files.
- SPSS files.
- Lotus 1-2-3 files from Releases 2, 3, 4, or 5.

Whether data from another locale imports correctly depends on whether the SAS server supports the locale of the data that is being imported. If you are importing data that contains characters that are different from the current locale, use a
Unicode (UTF–8) server to import your data. If you do not use a UTF–8 server and the locale of the data is not supported, unsupported values might appear as question marks (?) in your imported data. For more information about how to set the Default text encoding option, see “Setting General Preferences” on page 146.

Note: You cannot import remote files (files that are available through FTP file shortcuts).

Import an Excel Worksheet

To import an Excel worksheet:

1. Select New ➔ Import data.
2. In the work area, click Select Server File.
3. Select the file that you want to import and click Open. The Import.ctl tab shows the name and location of this Excel file. This example shows importing the as_products.xls file.

4. To import the data from a specific worksheet, enter the name of that worksheet in the Worksheet name box. By default, SAS Studio imports the data from the first worksheet.

5. To specify the location to save the output data set, click . By default, the output data set is saved to the Work library, which is a temporary location. The contents in this library are deleted when you exit SAS Studio.

6. Click Options.
Select the file type.

To generate SAS variable names from the data values in the first row of the worksheet, select Generate SAS variable names. If a data value in the first row in the input file is read and it contains special characters that are not valid in a SAS name, such as a blank, then SAS converts the character to an underscore.

Note: Data from a column whose name includes a forward slash, such as (Height/cm) cannot be imported if you are running SAS Studio 5.2 (Enterprise).

Specify the encoding.

7 To import the Excel worksheet, click .

---

Import a Delimited File (CSV, TXT, TSV, DLM)

Note: To import a tab-delimited file, the filename must have a TSV extension.

To import a delimited file:

1 Select New Import data.

2 In the work area, click Select Server File.

3 Select the file that you want to import and click Open. The Import.ctl tab shows the name and location of this text file.

This example shows importing a text file called CommaSep.txt.
To specify the location to save the output data set, click [Browse]. By default, the output data set is saved to the Work library, which is a temporary location. The contents in this library are deleted when you exit SAS Studio.

5 Click **Options**.

- Specify the file type.
- To generate SAS variable names from the data values in the first row in the text file, select **Generate SAS variable names**. If a data value in the first row in the input file is read and it contains special characters that are not valid in a SAS name (such as a blank), SAS converts the character to an underscore.
- In the **Delimiter** box, enter the delimiter for the values in the file. The default delimiter is a space.

**Note:** If you use a hexadecimal value to specify the delimiter, you do not need to select the **Quote delimiter** check box.

- To start reading data from a specified row in the delimited text file, enter the starting row in the **Start reading data at row** box. You might want to use this option if you have comments at the top of the text file or the first row of the file is column headings. By default, SAS Studio starts reading at row 2.
- For SAS Studio to determine the appropriate data type and length of the variables, enter a value in the **Guessing rows** box. The task scans the input data file from row 1 to the number that you specified. By default, the first 20 rows are scanned.
- Specify the encoding.

6 To import the data, click [Import].
Click the **Results** tab to see the attributes of the imported data set.

Click the **Output Data** tab to view the new SAS data set. If this data set is in the Work library (as shown in this example), you might want to save it to a more permanent location. Data in the Work library is temporary and is deleted when you exit SAS Studio.
Import a DBMS File

When you import a file from a database management system (DBMS), the available options depend on the file type. For a list of the supported file types, see "About Importing Data to SAS Studio" on page 95.

1. Select **New** ➔ **Import data**.
2. In the work area, click **Select Server File**.
3. Select the file that you want to import and click **Open**. The Import.ctl tab shows the name and location of this text file.
4. To specify the location to save the output data set, click . By default, the output data set is saved to the Work library, which is a temporary location. The contents in this library are deleted when you exit SAS Studio.
5. Click **Options**.
   - Specify the file type.
   - To generate SAS variable names from the data values in the first row in the text file, select **Generate SAS variable names**. If a data value in the first row in the input file is read and it contains special characters that are not valid in a SAS name (such as a blank), SAS converts the character to an underscore.
   - Specify the encoding.
6. To import the data, click .

Save the Import Task

You might want to save an instance of the Import task so that you can share these settings for importing a specific file with others at your site. SAS Studio saves these instances as a CTL file. CTL files must be run in the same operating environment where they were created. For example, if you create a CTL file using Windows, this CTL file must be run in Windows.

To save the Import task:

1. Click .
2 In the Save As window, specify the name and location, and then click Save. The file is saved with a CTL extension.

Exporting Data

In the Standard perspective, you can use SAS Studio to export your data as another file type to a folder that you specify.

Note: You cannot export password-protected data in SAS Studio, and you cannot export data from the Interactive perspective. For more information, see “Understanding Perspectives” on page 15.

To export your data:

1 Click Libraries in the navigation pane and browse to find the file that you want to export.

2 Right-click the file that you want to export and select Export. The Export Table window opens.

3 Select the folder in which you want to save the exported file.

4 In the Filename box, enter the name of the exported file.

5 From the Type drop-down list, select the format of the exported file.
6 Click **Save** to export the file.

**Note:** When you export data, the entire file is exported. Any filters you might have applied to the data in the table viewer are not applied to the exported data.
Working with Results

Viewing Results

When you run a task or a program in SAS Studio, the results are displayed in the work area. By default, results are generated as HTML5 output and displayed on the Results tab. Here are ways that you can work with your results:

- Add additional output formats, such as PDF, RTF, Excel, and PowerPoint, by using the Preferences window.
- Download your generated output. You can download results that have been created in any of the available formats.
- Change the default output style for each format by using the Preferences window.

Default SAS Studio Output

Viewing Default Results

In SAS Studio, by default, output is generated in HTML5.

If you want to add output formats, you can use the Preferences window to select other formats, such as PDF and RTF. You can also change the default style for your output to any of the ODS styles that are available. For more information, see "Setting the Results Preferences" on page 148.
By default, the HTML5 results are the only results that are displayed on the Results tab. When you view your results, you can use the table of contents to navigate the different sections. By default, the table of contents is expanded, and you can click the section that you want to navigate to.

Downloading Generated Data

To download results that were generated in other formats, such as PDF or RTF, or as other types of data, such as a .csv or XML file, for the Download and selecting the results that you want to download.

1. Click ☰ for the Results tab.
2 Select **Download** and the type for the results.

## Sending Your Results to Another User

You can send a copy of your results and the associated code and log files to another user through email. Files that you can send include results in HTML5, RTF, and PDF formats as well as the code and log files that are associated with the results. The code is sent as a SAS program file, and the log and program summary files are sent as HTML5 files. To send files through email, you need access to an SMTP server. For more information, contact your site administrator.

**Note:** If your SAS Studio email messages are being marked as junk, see "Configure Mail" in *SAS Viya Administration: Programming Run-Time Servers*.

To send results by email:

1. On the toolbar for the task, click 📩.

   ![Image of email button](image.png)

   Oct 26, 2019, 10:51:52 AM 📧 (0) 🔄

2. Select **E-Mail**. The E-Mail File window appears.

   ![Image of E-Mail File window](image2.png)
3 In the To box, enter the email addresses to which you want to send the files. Separate addresses with a semicolon.

4 If you want to send a copy of the email to another address, enter the address in the CC box.

5 In the Subject box, enter a subject for the email. You can also add a message to include in the body of the email.

6 Select the items that you want to include as attachments to your email. By default, the code is selected. If you select Results, all results that were generated are included in the email.

7 Click Send to send the message and attachments.

About the SAS Output Delivery System and SAS ODS Statistical Graphics

The SAS Output Delivery System (ODS) gives you greater flexibility in generating, storing, and reproducing SAS procedure and DATA step output along with a wide range of formatting options. ODS provides formatting functionality that is not available when using individual procedures or the DATA step without ODS.

SAS Studio uses very specific ODS options and the GOPTIONS statements so that the output is displayed properly in the web environment. To view all of the ODS options in your code, click Options > Preferences. In the Preferences window,
select **SAS Programs > Code and Log**, and then select the **Show generated code in the SAS log** option.

**Note:** To ensure that your output is displayed properly, do not change the settings of the ODS options or GOPTIONS statements in the generated code.

SAS ODS Statistical Graphics, more commonly referred to as SAS ODS Graphics, is an extension of the SAS Output Delivery System (ODS). ODS manages all output that is created by procedures and enables you to display the output in a variety of forms, including HTML and PDF.

Many SAS analytical procedures use ODS Graphics functionality to produce graphs. ODS Graphics uses the Graph Template Language (GTL) syntax, which provides the power and flexibility to create many complex graphs. The GTL is a comprehensive language for defining statistical graphics.
# Understanding Git Integration in SAS Studio

SAS Studio includes integration with Git, a system for tracking changes and managing version control among multiple users. Git can be used with many different repository hosting services such as GitHub and Bitbucket.

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>About Git Integration in SAS Studio</td>
<td>109</td>
</tr>
<tr>
<td>Working with Git Profiles</td>
<td>110</td>
</tr>
<tr>
<td>Creating a Git Profile</td>
<td>110</td>
</tr>
<tr>
<td>Deleting a Git Profile</td>
<td>111</td>
</tr>
<tr>
<td>Cloning and Opening a Git Repository</td>
<td>112</td>
</tr>
<tr>
<td>Creating a Local Repository</td>
<td>114</td>
</tr>
<tr>
<td>Understanding the Git Repository Tab in SAS Studio</td>
<td>115</td>
</tr>
<tr>
<td>Viewing the Commit History</td>
<td>116</td>
</tr>
<tr>
<td>Committing Changes to Your Local Repository</td>
<td>117</td>
</tr>
<tr>
<td>Pulling and Fetching Files</td>
<td>121</td>
</tr>
<tr>
<td>Pushing Files</td>
<td>121</td>
</tr>
<tr>
<td>Resetting Your Local Repository</td>
<td>121</td>
</tr>
<tr>
<td>Working with Branches in Git</td>
<td>122</td>
</tr>
<tr>
<td>Creating a Branch</td>
<td>123</td>
</tr>
<tr>
<td>Checking Out Branches</td>
<td>124</td>
</tr>
<tr>
<td>Understanding Merging and Rebasing</td>
<td>124</td>
</tr>
<tr>
<td>Merging Branches</td>
<td>125</td>
</tr>
<tr>
<td>Rebasing the Current Branch</td>
<td>126</td>
</tr>
<tr>
<td>Stashing Changes</td>
<td>126</td>
</tr>
<tr>
<td>Deleting a Repository</td>
<td>127</td>
</tr>
<tr>
<td>Sample Git Workflow Scenario</td>
<td>127</td>
</tr>
</tbody>
</table>
Note: The requirements for using the Git functionality in SAS Studio include the following:

- Standard perspective. For more information, see “Understanding Perspectives” on page 15.
- Red Hat Enterprise Linux (RHEL), Windows WX6, or Windows W32 operating system. If you are using a SUSE Linux Enterprise Server, you need to install additional libraries. Contact SAS Technical Support at support@sas.com for assistance.

You can use the Git Repositories section in the navigation pane to access the basic Git features from within SAS Studio. These features include cloning repositories, committing and stashing file changes, pulling and pushing files, viewing your repository history, creating and merging branches, rebasing a branch, and performing a basic differentiation between files in your local repository. By default, SAS Studio uses SSH keys to authenticate with your repository hosting service.

Note: For information about authenticating using HTTPS with a user name and password, contact your site administrator.

You can also interact with a Git repository from SAS by using SAS Git functions in a SAS program. For more information, see “Using Git Functions in SAS” in SAS Functions and CALL Routines: Reference.

---

Working with Git Profiles

Creating a Git Profile

Before you can get started using any of the Git features in SAS Studio, you must create a Git profile. By default, SAS Studio uses SSH keys for authentication. In order to use the SSH authentication, you must upload your public and private SSH keys to the server file system. For information about generating SSH keys, see the documentation for your repository hosting service.

Note: You can upload your SSH key files to the server file system by selecting the appropriate directory in the Explorer section in the navigation pane and clicking . Click + and browse for the SSH key files. Select the files and click Open. Click Upload to upload the files to the directory on your server file system.

To create a Git profile, select Options ⇒ Manage Git Connections and click +. The Add a Profile window appears.

To create a Git profile using SSH keys, enter your profile name, user name, and a valid email address in the appropriate boxes. Click Browse to find your public and private SSH key files or enter the path names in the appropriate boxes. Click OK to create your profile.
To create a Git profile using HTTPS authentication, contact your site administrator about enabling the password field for HTTPS authentication.

**Note:** To specify a default profile, select **Options ➤ Manage Git Connections** and click **Profiles**. Select the profile that you want to use as your default profile, and click **Set as default**.

---

**Deleting a Git Profile**

To delete a profile, select **Options ➤ Manage Git Connections** and click **Profiles**. Select the profile that you want to delete and click **Delete**.

If the profile is used in the active repository, you cannot delete it until you close the active repository. If you delete a profile that is associated with other repositories, you cannot open those repositories until you use the Manage Git Connections window to associate them with another profile.
Cloning and Opening a Git Repository

In order to access the files from a Git repository in SAS Studio, you must clone the repository to a local repository on your workspace server. When you clone the repository, a copy of all the files on the remote repository are copied to a working directory that you specify on your workspace server.

**Note:** If an existing Git repository has been cloned to the workspace server, then the Git repository is already a local repository and is available for you to point to. However, if the Git repository is not on the workspace server, you must clone it.

You can access the files in the cloned repository by navigating to the directory in the *Explorer* section in the navigation pane. You can make changes to the files in the working directory, and then commit those changes to your local repository.

**Note:** Changes that you make to files in your local repository are not visible to other users until you push those changes to the remote repository.

To clone a Git repository:

1. In the navigation pane, click ☐, and then click **Clone Repository**. The Clone a Repository window appears.

**Note:** If you already have an existing Git repository, you can access the option to clone a repository in two ways:

- In the *Git Repositories* section of the navigation pane, click **Clone**.
- Select **Options > Manage Git Connections**. In the Manage Git Connections window, click **Repositories**. Click ✈ and select **Clone a repository**.
Note: To open an existing local repository, click Add Repository or use the Add toolbar button in the Git Repositories section in the navigation pane. In the Select a Folder window, browse to find the repository that you want to open and specify the Git profile.

2 In the Repository box, enter the remote repository that you want to clone. For example, if you are using SSH key authentication, your repository name might look like one of these two examples:

   git@github.com:folder/sasgit.git

   or

   git@github.com:jquery/jquery-ui.git

   If you are using HTTPS authentication, your repository name might look like this example:

   https://github.com/example/repo.git

3 In the Server location box, enter the path name of a folder on your workspace server for the cloned repository.
Note: You must select an empty folder for the cloned repository. If there are any files in the folder, the cloning operation cannot be completed.

4. From the Profile drop-down list, select the profile that you want to use to connect to the Git repository.

5. Click Clone to clone the repository. A Git repository tab opens in the work area. You can also view the files in the cloned repository by using the Explorer section in the navigation pane.

Note: You can rename a repository from the Explorer section in the navigation pane. Right-click the repository folder and select Properties. Enter the new name in the Name box.

Creating a Local Repository

You can create a local repository from a folder in the Explorer section in the navigation pane. You can access the local repository in the Git Repositories section in the navigation pane and track and commit your changes. You can also associate the local repository with a remote repository and push your changes to the remote repository.

To create a local repository:

1. In the Explorer section in the navigation pane, right-click the folder that you want to use as a local repository and select Git initialize. The folders tree refreshes and the folder is identified in the tree as a Git repository.

2. To view the new local repository, click the folder in the navigation pane. Double-click the new local repository to open it in the work area. Notice that all of the files in the folder have been added to the repository as unstaged changes. To add the files
to the repository, you must first stage the files and then commit them. For more information, see “Committing Changes to Your Local Repository” on page 117.

3 To push the committed files to a remote repository, click **Push** on the Git repository tab. In the Create Remote Repository window, enter the location of the remote repository and click **Create**.

---

**Understanding the Git Repository Tab in SAS Studio**

When you clone a Git repository or open an existing Git repository from your server, a Git repository tab opens in the work area.

The Git repository tab contains two subtabs: History and Commit. The History subtab displays the history of all the changes, or commits, in a repository and enables you to view the differences between the selected and parent commits. The Commit subtab enables you to stage, unstage, reset, and commit changes to files in a repository. You can also view the differences between versions of a file.

To open your local Git repository tab, click in the navigation pane and double-click the repository that you want to open. The Git repository tab opens in the work area.

**Note:** You can view only one Git repository in SAS Studio at a time.

From the main Git repository toolbar, you can perform these tasks:

- Specify the local repository that you want to open.
- Check out a local branch.
- Pull or fetch data from the remote repository. For more information, see “Pulling and Fetching Files” on page 121.
- Push committed changes to the remote repository. For more information, see “Pushing Files” on page 121.
- Stash and reapply changes. For more information, see “Stashing Changes” on page 126.
- Refresh the local repository.

**Note:** You can refresh the Git Repositories section in the navigation pane by clicking and selecting **Refresh** on the Git Repositories toolbar.
Viewing the Commit History

You can use the commit history to view the history of all the changes to a repository. You can also view the details of a specific commit as well as the differences between two commits.

After you have saved your changes to the files in your working directory, make sure the tab for your local Git repository tab is open in the work area. Click the History subtab. A list of all the commits made in that repository is displayed in the work area in reverse chronological order. For each commit, the commit message, the author, the date and time of the commit, and the commit ID are displayed by default. To modify which columns are displayed, click the columns button and select the columns that you want to be displayed.

Note: By default, up to 1,000 rows are loaded in the commit history. You can specify a different limit by selecting Options \(\Rightarrow\) Manage Git connections. Click Options and enter a new value in the Maximum rows loaded in history log option. Increasing the number of rows that are loaded might affect your performance.
To view the details of a specific commit, select the commit from the commit history. The details of the commit are displayed in the lower section of the work area. You can view the details of the parent of the selected commit by clicking the parent commit ID in the details.

To view the differences between changes in the selected commit and the parent commits, click a file in the lower section of the work area. You can also compare two adjacent commits from the commit history list by selecting the commits that you want to compare.

**Note:** To view the file differences in a separate window, right-click the differences section of the work area and select **Open in new window**.

---

**Committing Changes to Your Local Repository**

Before you can push one or more changes from your local Git repository to the remote repository, you must first commit the changes from your working directory to your local repository. When you commit files, you stage the files that you want to include in the commit by moving the files from the Unstaged changes area to the Staged changes area. Next, you must enter a comment describing the commit, and then commit the files. When you select files to commit, it is a good practice to group changes that are related and represent a state in your project development that you want to capture.

**Note:** When you commit a change to the local repository, those changes are still not viewable by other users, but the changes are available for you to push to the remote repository.

To commit changes to your local repository:

1. After you have saved your changes to the files in your working directory, make sure your local Git repository tab is open in the work area. Click the **Commit** subtab.

2. If your changed files are not listed in the Unstaged changes area, click to refresh the list.
Files in the staging area are identified by a status icon that indicates the type of change:

- 🔄 – a new file to be added to the local repository.
- ⏩ – a renamed file.
- 🔄 – an existing file with changes.
- ⏪ – a deleted file.
- ⛔️ – a file with content that is very similar or identical to another file in the local repository.
- ☢ – a file with a merge conflict.
Note: Autosaved files that include ~ in the file name are not included in the list of unstaged files.

3 Select one or more changes that you want to commit and click . When you select files to commit, it is a good practice to group changes that are related and belong together in a single commit.

**TIP** You can also stage and unstage files by double-clicking them.

Note: Changes that you do not stage are not included in the commit. Your changes remain in your working directory and in the Unstaged changes area until you commit them.
Note: To stage all changed files at one time, click.

TIP You can view the differences between the versions of a file in the staging area and your local repository by clicking the file on the Commit tab. To discard the changes to a file in the staging area and reset the file to the version in your local repository, right-click the file and select Reset.

4 In the comments box, enter a description of the changes in the files that you are committing, and then click Commit Staged. The changed files are now available to be pushed to the remote repository.
Pulling and Fetching Files

You can download data from the remote repository to your local repository in two ways: pulling and fetching. The pull feature enables you to download the files from the remote repository and update your local repository to make sure you have the latest content. If you have made changes to some of the files in your local repository, Git merges the content from the remote repository with your local files. To pull files from the remote repository, make sure the tab for your local Git repository is open in the work area, and then select **Pull branch-name** ⇒ **Pull**.

**Note:** SAS Studio displays a message if there is a conflict between files in the remote and local repositories when you pull the remote files. If a conflict occurs, the conflicted files are moved to the Unstaged changes area. After you resolve the conflicts, you can commit the files again. Files that are pulled from the remote repository and do not have any conflicts are staged for the commit.

The fetch feature downloads the files from the remote repository, but fetching does not merge any changes into your current working branch. To fetch files from the remote repository, make sure the tab for your local Git repository is open in the work area. Next, select **Pull branch-name** ⇒ **Fetch** and specify whether you want to fetch changes from a specific branch or from all branches.

Pushing Files

After you have staged and committed your changed files from your working directory to your local repository, you can push them to the remote repository. When you push your committed changes to the remote repository, you are synchronizing files in your local repository with the remote repository.

To push your files to the remote repository, make sure the tab for your local Git repository is open in the work area. Click **Push branch-name** on the toolbar. The commit history shows your committed changes merged into the remote repository.

Resetting Your Local Repository

You can undo commits that you have made by resetting your local repository to a prior commit.

To reset your local repository:

1. Make sure the commit history is open for your local repository. Right-click the commit that you want to reset and select **Reset**. The Reset Local Repository window appears.
2 Select the type of reset that you want to perform:

- **Soft** – resets the current branch to the prior commit. No changes are made to the staged changes (the index) or your working directory.

- **Mixed** – resets the current branch to the prior commit and resets your staged changes (the index). No changes are made to your working directory.

- **Hard** – resets the current branch to the prior commit, resets your staged changes (the index), and discards all changes in your working directory. You cannot undo a hard reset.

3 Click **OK** to reset your repository.

---

### Working with Branches in Git

Your work in Git is managed in branches. By default, the main branch in your remote repository is identified as the origin/master branch. By default, the main branch in your local repository is identified as the master branch. You can create additional branches and merge branches. When you push your committed changes to the remote repository, you are synchronizing files in a branch in your local repository with the branch in the remote repository. You can use the History subtab to see the commits that are associated with a branch and whether branches are synchronized. For more information, see “Viewing the Commit History” on page 116.

In this example, the master branch is ahead of the origin/master commit by one commit. This lack of synchronization indicates that you have made changes to the local master branch that are not included in the remote origin/master branch because the "performance enhancements" commit has not yet been pushed to the remote repository. For more information, see “Pushing Files” on page 121.

After the “performance enhancements” commit has been pushed to the remote repository, the origin/master and master branches are synchronized and are listed on the same line in the commit history.
Creating a Branch

You can create a branch in Git in order to separate some of your work from other work that you are doing. When you create a branch, the commits in that branch are separated from your other commits until you merge the branch with another branch.

Note: When you have multiple branches, you can check out the branch that you want to work in by using the Current branch drop-down list at the top of your Git repository tab. You can also right-click the branch in the commit history and select Checkout branch \( \text{branch-name} \). For more information, see “Checking Out Branches” on page 124.

To create a branch:

1. Make sure the commit history is open for your local repository. Right-click the commit with which you want to start a new branch and select Create new branch. The Create a Branch window appears.

2. In the Branch name box, enter a name for your new branch. The branch name can contain only letters, numbers, and special characters such as hyphens or underscores.

3. If you want to create a branch but remain in the current branch, clear the Checkout after create check box. This option is selected by default.

4. If you want to discard any changes in your working directory that you have not committed when the branch is created, select the Force if duplicate branch name exists check box. This option is not selected by default.

5. Click Create to create the branch. Your local repository now contains the files from the commit that you selected for creating the branch.
Checking Out Branches

To open a branch in your working directory, you must check out the branch.

You can check out local branches by using the Current branch drop-down list at the top of the Git tab or by right-clicking the local branch in the commit history and selecting Checkout branch \( \Rightarrow \) branch-name.

If you want to work with a remote branch that does not have a corresponding local branch, you can create a local copy of that branch in your working directory in one of the following ways:

- Create a local branch with the same name as the remote branch. For more information, see "Creating a Branch" on page 123.
- Right-click the remote branch in the commit history and select Checkout branch \( \Rightarrow \) branch-name. A local version of the branch is created and checked out.

Note: If you check out a remote branch that already has a corresponding local branch, the remote branch is merged into the local branch.

Understanding Merging and Rebasing

Both merging and rebasing are used to integrate changes from one branch into another branch.
When you merge one branch into another, Git combines the changes from the commit sequences in each branch. A merge conflict occurs when Git encounters something in a file that has been changed in both branches. You must resolve any conflicts manually before the merge can be completed. After the merge is complete, you can still view the history of both branches. The existing branches are not changed in any way.

When you rebase a branch, you are moving an entire branch so that it begins on the tip of another branch. Rebasing enables you to incorporate all of the commits in one branch into another branch. You must resolve any conflicts manually before the rebase can be completed. Unlike merging, rebasing rewrites the project history and creates a linear project history. You can no longer view the original branch histories separately.

**Note:** You should never rebase a public branch, such as master, onto a new branch because this action can create problems for other users of the public branch.

---

### Merging Branches

When you have completed work in a branch, you can merge the work in that branch into another branch.

To merge a branch:

1. Make sure the commit history is open for your local repository. In the **Current branch** drop-down list, select the branch into which you want to merge the new branch. For example, to merge `newbranch` with the master branch, you would need to check out the master branch.

2. Right-click the branch that you are merging, and select **Merge into current-branch-name ⇒ new-branch-name**. For example, to merge `newbranch` with the master branch, you would right-click `newbranch` and select **Merge into master ⇒ newbranch**. The current branch in the commit history list is updated with the latest commit from the new branch.

**Note:** If you are merging a branch into a branch that is not checked out, right-click the latest commit in the current branch, and select **Merge with branch ⇒ branch-name**.

**Note:** SAS Studio displays a message if there is a conflict between files when you merge the branches. If a conflict occurs, the conflicted files are moved to the Unstaged changes area on the Commit subtab. After you resolve the conflicts, you can complete the merge by staging and committing the files.
Rebasing the Current Branch

You can use the rebase feature to move a branch to a new base commit. When you rebase a branch, the branch appears to have been created from a different commit. Rebasing is a powerful feature that enables you to change the history of your repository.

To rebase a branch:

1. Make sure the commit history is open for your local repository. From the Current branch drop-down list, select the branch that you want to rebase. For example, to rebase newbranch onto the master branch, you would need to check out the newbranch branch.

2. Right-click the branch that you want to use as the new base commit, and select Rebase branch-name on → Selected commit.

Note: SAS Studio displays a message if there is a conflict between files when you rebase the branches. If a conflict occurs, the conflicted files are moved to the Unstaged changes area on the Commit subtab. After you resolve the conflicts, stage and commit the files and continue the rebase.

Stashing Changes

You can use the stash feature to save any uncommitted changes (both staged and unstaged) that you are not yet ready to push to the remote repository. When you stash changes that you have made to one or more files, the changes are removed from the working copy of the files and saved until you are ready to reapply them.

When you reapply the changes to your files, you can choose to delete or save the changes in your stash.

Note: When you stash changes, all updated and uncommitted files in the repository are affected.

To use the stash feature, make sure the tab for your local Git repository is open in the work area. Click on the toolbar and choose from the following options:

- **Stash** – saves all uncommitted changes to the files in your local repository and removes the changes from the local files.
- **Pop stash** – reapplies all saved changes to the files in your local repository and deletes the changes from your stash.
- **Delete stash** – deletes the saved changes in your stash. When you delete the changes in your stash, you cannot reapply them to your local files.
Apply stash – reapplies all saved changes to the files in your local repository and saves the changes in your stash.

Deleting a Repository

You can delete the local repository on your workspace server. You can choose whether you want to delete all the files in the working directory when you delete the repository.

To delete the local repository:

1. Select **Options ➤ Manage Git connections**. Click **Repositories** and select the repository that you want to delete. Click **The Confirm Delete window appears.**

2. Select the type of delete that you want to perform:

   - **Soft** – deletes only the Git repository definition.
   
   Note: After a soft delete, you can still view the folder using the **Explorer** section in the navigation pane. If you want to use the folder as a local Git repository again, you must right-click the folder and select **Git initialize**.

   - **Mixed** – deletes the Git repository definition and removes any Git configuration information from the local repository directory. The files in your working directory are not deleted.
   
   Note: After a mixed delete, you can still view the folder using the **Explorer** section in the navigation pane. If you want to use the folder as a local Git repository again, you must right-click the folder and select **Git initialize**.

   - **Hard** – deletes the Git repository definition, the local repository directory, and all files in the local working directory.

3. Click **Delete** to delete the repository.

Note: You can also delete a repository from the **Explorer** section in the navigation pane. Right-click the working directory and select **Delete**. Click **Delete** in the confirmation window, and then select the type of delete that you want to perform.

Sample Git Workflow Scenario

Suppose you need to help out on a project that stores its SAS programs in a Git repository. You need to make some minor changes to one program and add a new program to the project.
In this sample Git scenario, you clone a repository to your workspace server, commit and push some changes, and create a branch. Then you merge the new branch with the master branch and push the changes.

1. Create a profile so that you can access Git repositories in SAS Studio.
   a. Select Options ➔ Manage Git connections. Make sure Profiles is selected, and then click ➔ to open the Add a Profile window.
   b. Enter the credentials necessary to access your Git repository and click OK.

2. Clone your team’s repository to your workspace server so that you can access the files in the project.
   a. In the Git Repositories section in the navigation pane, click Clone a Repository.
   b. In the Clone a Repository window, specify the remote repository and the location on your workspace server that you want to use.
c Click **Clone** to create the repository. The repository is added to the **Git Repositories** section in the navigation pane.

3 Open the program that you need to update from your local repository, make your changes, and save the program.

   a Click in the navigation pane and navigate to your working directory.

   b Double-click the SAS program that you want to edit and make your changes. Click to save your changes.
4 Stage your changed files, enter a commit message, and commit your files to your local repository.

a Click in the navigation pane, and then double-click the repository to open it in the work area. Make sure the Commit tab is selected. If your changed file is not listed in the Unstaged changes area, click to refresh the list.

b Select the updated file and click to move the file to the Staged changes area.

c Enter your commit comments in the commit box and click Commit Staged.
Your changed file is committed to the local repository.

View your commit history by clicking **History** on your Git repository tab. Notice that your commit is listed at the top of the commit history list. Your commit currently exists only in the master branch of your local repository. The master branch of your local repository is identified by 

The second commit in the list represents the state of the master branch on the remote repository. The master branch on the remote repository is identified by

When you push your changes to the remote repository, your commit is then part of the origin/master branch on the remote repository.
5 Pull the files from the remote repository and push your committed files.

Note: It is good practice to pull the files from the remote repository to make sure you do not have any conflicts.

a On your Git repository tab, select Pull branch-name » Pull. Your local repository is updated with the latest content from the remote repository. Any changes in the remote repository are merged with your local files. If there is a conflict between files in the remote and local repositories, SAS Studio displays a message so that you can resolve the conflict before you proceed.

b Click Push branch-name to push your changed file to the remote repository. Your commit history is updated. Notice that your commit is now part of the remote origin/master branch.
Create a branch to keep your work separate from the master branch. Add a new program, test it, and commit your changes to that branch.

a In the commit history list, right-click your commit and select **Create new branch**. The Create a Branch window appears.

b Enter a name for the branch and click **Create**. Notice that the **Checkout after create** option is selected so that the new branch is checked out and is the active branch that you are working in.

In the commit history list, notice that the new classdata branch, the remote origin/master branch, and your local repository master branch are all synchronized and displayed on the same line.
Use the **Explorer** section in the navigation pane to create a program and save it in your working directory.

Stage the new program and commit it to the new classdata branch in your local repository. Notice in the commit history list that your commit in the classdata branch is not synchronized with the remote origin/master branch or the local master branch.
When your new program is complete, merge the new branch back into the master branch and perform a final pull from and push to the remote repository.

When you are ready to merge the classdata branch into the master branch, you must first check out the master branch. Select the master branch from the Current branch drop-down list. In the confirmation window, click **Reload** to make sure that all of your changes are saved.

In the commit history list, right-click the classdata branch and select **Merge into master** → **classdata**.
**Notice in the commit history list that the master branch has been updated with the new file from the classdata branch.**

- **On the Git tab, select Pull master → Pull.**

- **Click Push to push your changes to the remote repository. Your commit history is updated. Notice that the local master branch, which was merged with the classdata branch, and the remote origin/master branch are now synchronized.**
What Is a Task?

A task is an XML and Apache Velocity code file that generates SAS code and formats results for you. Tasks include SAS procedures from simple data listings to complex analytical procedures. SAS Studio is shipped with several predefined tasks that are available in the Standard perspective. The tasks are organized into categories. Some categories and their tasks might not be available at your site because you do not have the required SAS product.

For more information about each of these tasks, see SAS Studio: Task Reference Guide.

How to Run a Task

To run a predefined task:

1. In the navigation pane, click to open the Tasks pane.
2. Expand the folder that contains the task.
3. Right-click the task name and select Open. Alternatively, you can double-click the task to open it.

   The task opens to the right of the work area.
4 If the Data tab is available, specify an input data source and select columns for the roles in the data source. A role is a description of a variable’s purpose in the task. To add a column to a role, click . A list of available columns for that role appears. If multiple columns can be assigned, you can press Ctrl or Shift to select multiple columns from the list and click OK.

5 On the remaining tabs, specify any other required options, which are denoted with a red asterisk. As you assign values to the task, the relevant SAS code is generated. For more information about the options available for each task, see SAS Studio: Task Reference Guide.

TIP The Task Console displays any information required to run the task.

6 To run the task, click Run.

Edit a Predefined Task

To customize the predefined tasks for your site, you can edit the XML code that is used to create the task.

To edit a predefined task:

1 In the navigation pane, click to open the Tasks pane.

2 Expand the folder that contains the task.
3 Right-click the name of the task that you want to edit and select **Copy as task template**. You can choose to save the task in My Tasks or the Explorer.

4 Specify a name and description for the task. By default, the name and description from the predefined task is used.

   Click **OK**.

5 Right-click the copied task and select **Edit task template**. The CTM file for the task appears in a new tab.

6 Edit the XML file and save your changes. To preview your changes, click **Open**.

---

Create a New Task

SAS Studio provides a template that you can use to create custom tasks for your site.

To create a custom task:

1 In the navigation pane, click **Tasks** to open the **Tasks** pane.

2 Click **Task** and select **Task**. A blank task template opens.

3 Edit the code in the task template to create your task. To view the user interface for the task template, click **Open**. For more information about this file, see *SAS Studio: Developer’s Guide to Writing Custom Tasks*.

4 Click **Edit**.
Specifying the Options for the Task Code

The Preferences window enables you to change several options that affect what and how the task code is displayed.

To access these options, select Options ⇒ Preferences. In the Preferences window, click Tasks.

For more information, see “Setting Task Preferences” on page 154.
About Customizing SAS Studio

You can customize SAS Studio by changing the global settings, such as theme and locale, and by specifying application preferences, such as code editor and table viewer options.
Creating Global Settings

About the Settings Window

The Settings window enables you to specify settings for the appearance theme, locale, and accessibility. To change the global settings, click your initial on the toolbar and select Settings.

Note: The global settings are available only in SAS Studio 5.2 (Enterprise).

Specifying the General Settings

From the General page, you can specify these options for SAS Studio.
<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Theme</strong></td>
<td>Enables you to change the appearance of the SAS Studio web application. The theme specifies the collection of colors, graphics, and fonts that appear in the application. You can choose from SAS themes or custom themes, if you have any available.</td>
</tr>
<tr>
<td></td>
<td>- Use the default theme (<em>theme-name</em>) — uses the default theme as specified by your system administrator.</td>
</tr>
<tr>
<td></td>
<td>- Choose a theme — enables you to select a theme. You can choose from these SAS themes:</td>
</tr>
<tr>
<td></td>
<td>- High Contrast: Presents a dark background with high-contrast foreground elements to meet the needs of users with low vision.</td>
</tr>
<tr>
<td></td>
<td>- Ignite: Presents a dark user interface that enables graphs, visualizations, and other elements to stand out.</td>
</tr>
<tr>
<td></td>
<td>- Illuminate: Includes a clean and uncomplicated color palette that is easy to use. This is the default theme.</td>
</tr>
<tr>
<td></td>
<td>- Inspire: Consists of vibrant and cohesive colors that shift the emphasis from the application to the content.</td>
</tr>
<tr>
<td></td>
<td>The theme change takes effect after you close the Settings window.</td>
</tr>
<tr>
<td><strong>Reset to show all warning and information messages</strong></td>
<td>Displays warnings and messages that you previously chose to hide.</td>
</tr>
<tr>
<td><strong>Profile picture</strong></td>
<td>Displays your profile picture.</td>
</tr>
</tbody>
</table>

**Specifying the Region and Language Settings**

From the Region and Language page, you can specify these options.
<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Locale for regional formats and sorting</td>
<td>Specifies the locale that is used for sorting data and formatting values such as dates, times, numbers, and currency. The default setting is the browser locale. Changes take effect after you sign out and sign back in.</td>
</tr>
<tr>
<td>Locale for offline processes</td>
<td>Specifies the locale that is used for offline jobs or background processes such as report distributions or notifications. The default setting is the locale of the Java Runtime Environment.</td>
</tr>
</tbody>
</table>

**Specifying the Accessibility Settings**

From the Accessibility page, you can specify these options.

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable sounds</td>
<td>Enables audio indicators for events that occur within the user interface.</td>
</tr>
<tr>
<td>Enable visual effects</td>
<td>Adds visual effects that indicate state changes. For example, when this setting is enabled, you see a subtle movement in the user interface if you delete an item.</td>
</tr>
<tr>
<td>Adjust the display duration for pop-up notices</td>
<td>Enables you to specify the length of time in seconds that pop-up notifications are displayed. The default value is 5 seconds.</td>
</tr>
<tr>
<td>Invert application colors</td>
<td>Displays SAS Studio in inverted colors.</td>
</tr>
<tr>
<td>Display tooltips when using the keyboard to navigate</td>
<td>Displays the tooltips for each item in the application in a pop-up window as you navigate with the keyboard. You can specify where in the browser window you want to display the tooltips.</td>
</tr>
<tr>
<td>Customize the focus indicator settings</td>
<td>Enables you to customize the attributes of the outline that indicates which user interface component is active.</td>
</tr>
</tbody>
</table>
Setting Your Preferences

About the Preferences Window

The Preferences window enables you to customize several options in SAS Studio. To change your preferences, select **Options ➤ Preferences**.

Setting the Start Up Preferences

From the **Start Up** page, you can specify these options for starting SAS Studio.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tabs to Open</strong></td>
<td>Specifies the tabs to be displayed when you start the SAS Studio application. You can choose from these options:</td>
</tr>
<tr>
<td></td>
<td>- <strong>None</strong> starts SAS Studio with no tabs open in the work area. This option is selected by default.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Restore tabs listed in Open Files pane</strong> restores the tabs that were open in your prior session of SAS Studio. When this option is selected, the tabs are also restored if you switch between the Standard and Interactive perspectives in a SAS Studio session. Any temporary files that were open are not restored. Note: The tabs are not necessarily restored in the same order they were in when you closed SAS Studio. Tabs that display a large amount of data can take longer to open and might be opened last.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Open a new file</strong> opens a new program tab.</td>
</tr>
<tr>
<td><strong>Show Start page</strong></td>
<td>Opens the Start Page tab when you start the SAS Studio application. To automatically close the Start Page tab when you open another tab in the work area, select <strong>Automatically dismiss</strong>.</td>
</tr>
</tbody>
</table>
### Option Description

- **Restore open folder state in Explorer pane**
  
  Automatically reopens the folders that were expanded in the **Explorer** section of the navigation pane in your prior session of SAS Studio.

---

### Setting General Preferences

From the **General** page, you can set these options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Display tab after submission</strong></td>
<td>Specifies the tab to display after you run a program or a task. You can choose to display the code, log, or results tabs. If you are running the Standard perspective, you can also choose to display output data tab. The Results tab is displayed by default.</td>
</tr>
<tr>
<td><strong>When activating output data tabs, automatically open</strong></td>
<td>Specifies which output data tab to display when you run a program or task that generates data.</td>
</tr>
<tr>
<td><strong>Display warning if text files are larger than specified size</strong></td>
<td>Displays a warning when you open a text file that is larger than the specified size in megabytes (MB). Text files include file types such as *.sas, *.txt, and *.csv. If you open a file that is larger than the specified size, the file is truncated and opened in Read-Only mode. The default size is 10 MB.</td>
</tr>
<tr>
<td><strong>Default text encoding</strong></td>
<td>Specifies the character-set encoding that is used when text files, such as *.sas programs, are read or written. The default value is UTF-8. For a list of some of the encoding options and the languages that they are associated with, see <strong>Appendix 3, &quot;Text Encoding Options and Language Mappings,&quot; on page 159</strong>.</td>
</tr>
<tr>
<td><strong>Enable document recovery</strong></td>
<td>Enables you to recover the most recently saved version of your files if SAS Studio times out while you are away or closes unexpectedly due to a situation such as a power loss. If you select this option, the Document Recovery window opens when you restart SAS Studio, and you can choose the files that you want to recover. This option is selected by default. You can use the <strong>Document save interval</strong> option to specify how often you want to save your files. The default interval is 30 seconds.</td>
</tr>
</tbody>
</table>
## Setting Your Preferences

### From the SAS Programs ➤ Code and Log page, you can specify these options for the code and log tabs.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Show generated code in the SAS log</strong></td>
<td>Displays the ODS statements, %LET statements, and any other code that is automatically generated by SAS in the log file. This option applies to both SAS tasks and SAS program files.</td>
</tr>
<tr>
<td><strong>Show custom code in the SAS log</strong></td>
<td>Includes custom code in the log file.</td>
</tr>
</tbody>
</table>
### Setting the Results Preferences

From the **Results** page, you can specify these options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Display warning if results are larger than specific size</strong></td>
<td>Displays a warning message when you attempt to open a results file that is larger than the size that is specified in the <strong>Maximum size</strong> option. You can enter a value between 0.5 and 10 MB. The default value is 3 MB.</td>
</tr>
<tr>
<td><strong>Generate table of contents</strong></td>
<td>Creates a table of contents for the output.</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Produce HTML output</td>
<td>Generates results in HTML format. This is the default results format. If you clear this option, then the Results tab is not displayed when you run a program.</td>
</tr>
<tr>
<td>Specify an HTML output style</td>
<td>Enables you to specify the style that is applied to results in HTML. If this option is not selected, SAS Studio uses the style that is associated with the application theme. If this option is selected, you can use the Output style drop-down list to select the style that is applied to the results.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This option is available only in SAS Studio 5.2 (Enterprise).</td>
</tr>
<tr>
<td>Generate HTML graphs as SVG</td>
<td>Creates SVG graphs instead of PNG graphs in HTML output. SVG graphs maintain clarity when you zoom in and out.</td>
</tr>
<tr>
<td>Enable accessible graph option</td>
<td>Adds accessibility metadata to graphs that are created by ODS Graphics. Users with disabilities access the accessibility metadata using SAS Graphics Accelerator. This option is available only if you are running SAS 9.4M4 or later. For more information, see the ODS HTML5 Statement in SAS Output Delivery System: User’s Guide.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The SAS code that is associated with this option is displayed in the log file only if you have selected the Show generated code in the SAS log option in the General preferences.</td>
</tr>
<tr>
<td>Produce PDF output</td>
<td>Generates results in PDF format. This option is not selected by default and is available only if you are running the Standard perspective.</td>
</tr>
<tr>
<td>Output style</td>
<td>Displays the style that is applied to results in PDF. To change the style that is applied to the results, select another style from the drop-down list. This option is available only if you are running the Standard perspective.</td>
</tr>
<tr>
<td>Generate table of contents</td>
<td>Creates a table of contents in the PDF file. This option is available only if you are running the Standard perspective.</td>
</tr>
<tr>
<td>Enable accessible PDF option</td>
<td>Adds accessibility metadata to the PDF file that enables the file to be accessed by assistive technology such as a screen reader. When metadata is added, the file is often called a &quot;tagged PDF&quot; and follows the PDF/Universal Accessibility (PDF/UA) format. This option is available only if you are running SAS 9.4M4 or later and the Standard perspective.</td>
</tr>
<tr>
<td>Produce RTF output</td>
<td>Generates results in RTF format. This option is not selected by default and is available only if you are running the Standard perspective.</td>
</tr>
</tbody>
</table>
### Setting Editors Preferences

From the **Editors** page, you can set the options for the SAS and XML language editors.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enable snippet abbreviations</strong></td>
<td>Enables you to insert snippets into a program by entering <code>@snippet-name</code> or entering @ and using the pop-up window to select from a list of snippets in your My Snippets folder. This option is selected by default. For more information, see “Create a Snippet” on page 55.</td>
</tr>
<tr>
<td><strong>Editor Options</strong></td>
<td>Enables you to specify editing and display options as well as appearance options for the code editor. For more information, see “Setting General Code Editor Preferences” on page 151 and “Setting Code Editor Appearance Preferences” on page 152.</td>
</tr>
</tbody>
</table>
### Setting General Code Editor Preferences

From the **Editors** page, click **Editor Options** and then click **Behavior and Appearance** to set editing and display options for the code editor.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Show autocomplete list</strong></td>
<td>Turns on the autocomplete feature of the code editor. This feature can predict the next keyword that you want to type before you actually type it completely. For more information, see “Using the Autocomplete Feature” on page 26.</td>
</tr>
<tr>
<td><strong>Auto-indent</strong></td>
<td>Automatically indents a new line by the amount of space that the previous line is indented.</td>
</tr>
<tr>
<td><strong>Auto-insert block comments</strong></td>
<td>Automatically inserts the closing comment tag when you enter /* and press Enter so that you can insert comments that span multiple lines.</td>
</tr>
<tr>
<td><strong>Use overtype mode</strong></td>
<td>Replaces existing characters with new characters when you type in the code editor.</td>
</tr>
<tr>
<td><strong>Auto-pair quotation marks</strong></td>
<td>Automatically inserts a closing quotation mark when you type a quotation mark and positions the cursor in between them.</td>
</tr>
<tr>
<td><strong>Highlight matching brackets</strong></td>
<td>Highlights the matching bracket when you position the cursor on a bracket.</td>
</tr>
<tr>
<td><strong>Auto-pair brackets</strong></td>
<td>Automatically inserts a closing bracket when you type a left bracket and positions the cursor in between them.</td>
</tr>
<tr>
<td><strong>Enable code folding</strong></td>
<td>Enables you to collapse and expand sections of code in the code editor.</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Enable line highlighting</td>
<td>Automatically highlights the line of code that you specify when you use the <strong>Go to line</strong> feature.</td>
</tr>
<tr>
<td>Show indent guides</td>
<td>Displays vertical lines that indicate where an indented section of code begins and ends.</td>
</tr>
<tr>
<td>Show line-length guide</td>
<td>Displays a vertical line in the Program Editor at the column position that you specify. The default column position is 80.</td>
</tr>
<tr>
<td>Show invisible characters</td>
<td>Displays characters such as spaces and carriage returns in the Program Editor.</td>
</tr>
<tr>
<td>Show line numbers</td>
<td>Displays line numbers in the leftmost column of the program tab.</td>
</tr>
<tr>
<td>Print line numbers</td>
<td>Includes line numbers in the leftmost column when you print a program.</td>
</tr>
<tr>
<td>Wrap text</td>
<td>Displays text that continues past the right edge of the page on the next line.</td>
</tr>
<tr>
<td>Tab width</td>
<td>Specifies the number of spaces that are inserted into your text when you insert a tab character. The default value is four spaces for each tab character.</td>
</tr>
<tr>
<td>Substitute spaces for tabs</td>
<td>Inserts the number of spaces specified in the <strong>Tab width</strong> option instead of a single tab character.</td>
</tr>
</tbody>
</table>

### Setting Code Editor Appearance Preferences

You can specify appearance options for both SAS code and XML code in the code editor. From the **Editors** page, click **Editor Options** and then click **Colors and Fonts for SAS** or **Colors and Fonts for XML** to set appearance options for the code editor.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax theme</td>
<td>Specifies whether to use a light or dark background in the code editor.</td>
</tr>
<tr>
<td>Font</td>
<td>Specifies the font that is used in the code editor.</td>
</tr>
<tr>
<td>Editor background</td>
<td>Specifies the background color that is used in the code editor.</td>
</tr>
</tbody>
</table>
### Setting Your Preferences

From the **Tables** page, you can set the options for the table viewer.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enable syntax highlighting</strong></td>
<td>Enables you to specify the text style and background color of specific file elements in the code editor.</td>
</tr>
</tbody>
</table>

### Setting Tables Preferences

From the **Tables** page, you can set the options for the table viewer.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Column headers</strong></td>
<td>Specifies whether column name or labels are displayed in the table viewer. Column names are displayed by default.</td>
</tr>
<tr>
<td><strong>Rows per page</strong></td>
<td>Specifies the number of rows to display per page of data. The default value is 200. Note: You can override this option for an open table in the table viewer by clicking on the table viewer toolbar and selecting <strong>Row paging</strong>.</td>
</tr>
<tr>
<td><strong>Columns displayed</strong></td>
<td>Specifies the maximum number of columns to display in the table viewer. The default value is 100. Increasing the number of columns that are displayed might affect your performance.</td>
</tr>
</tbody>
</table>
| **SAS variable name policy** | Enables you to specify one of the following sets of rules to apply to SAS variable names.  

- **ANY** specifies that the variable names can begin with or contain any characters, including blanks, must contain at least one character, and cannot contain any null bytes. Variable names can contain mixed-case letters as well as special and multi-byte characters. Names can be up to 32 bytes in length. This option is selected by default. Leading blanks are preserved, but trailing blanks are ignored.

- **V7** specifies that the variable names must begin with a letter of the Latin alphabet (A-Z, a-z) or the underscore character. They cannot contain blanks or special characters except for the underscore and cannot be assigned the names of special SAS automatic variables or variable list names. Variable names can contain mixed-case letters and can be up to 32 bytes in length.

- **UPCASE** specifies that the variable name follows the same rules as V7, except that the variable name is uppercase, as in earlier versions of SAS. |
### Option

**SAS member name policy**

Enables you to specify one of the following sets of rules to apply to SAS data set names, SAS data view names, and item store names. Any changes you make to this option take effect when you start SAS Studio or reset your SAS session.

- **EXTEND** (Extended member names) specifies that names must contain at least one character (letters, numbers, valid special characters, and national characters), cannot begin with a blank or a period, and cannot include a null byte. Names can contain mixed-case letters and can be up to 32 bytes in length. Leading blanks are preserved, but trailing blanks are ignored. Special characters cannot include the / \ * ? " < > : - characters. This option is selected by default.

- **COMPATIBLE** (Basic member names) specifies that the names must begin with a letter of the Latin alphabet (A-Z, a-z) or the underscore character and cannot contain blanks or special characters except for the underscore. Names can contain mixed-case letters and can be up to 32 characters in length.

### Setting Task Preferences

From the **Tasks** page, you can set the options for the generated SAS code and the task layout in the SAS Studio workspace. These options are available only if you are running the Standard perspective.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Generate header comments for task code</strong></td>
<td>Adds comments before the generated code for a SAS task.</td>
</tr>
<tr>
<td><strong>Automatically format generated code</strong></td>
<td>Automatically formats any code that is generated by a task.</td>
</tr>
<tr>
<td><strong>Show Preview section</strong></td>
<td>Specifies whether to display the preview section of the task tab after you run a task and which tabs to include in the preview.</td>
</tr>
</tbody>
</table>
### Setting Background Submit Preferences

From the **Background Submit** page, you can set the options for background job submissions. These options are available only if you are running the Standard perspective.

<table>
<thead>
<tr>
<th><strong>Location of log and output files</strong></th>
<th>Specifies where to save the output and log files. You can choose from these options:</th>
</tr>
</thead>
<tbody>
<tr>
<td>- <strong>Use same name and location as file</strong></td>
<td>uses the name and location of the program file for the log and output files.</td>
</tr>
<tr>
<td>- <strong>Specify names and locations</strong></td>
<td>enables you to specify a name and location for the log and output files.</td>
</tr>
<tr>
<td>- <strong>Prompt for output and log file names</strong></td>
<td>prompts you to specify a location in which to save the log and output files.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Log or output files exist</strong></th>
<th>Specifies how to handle the background job submission if a log and output file already exist. You can choose from these options:</th>
</tr>
</thead>
<tbody>
<tr>
<td>- <strong>Replace existing files</strong></td>
<td>automatically replaces existing log and output files of the same name.</td>
</tr>
<tr>
<td>- <strong>Cancel the submission</strong></td>
<td>cancels the background job submission when there are existing log and output files of the same name.</td>
</tr>
<tr>
<td>- <strong>Automatically add a timestamp</strong></td>
<td>saves all log and output files by creating a unique filename for each file. The log and output files are saved as <code>program-name(YYYY-MM-DD HH:MM:SS)</code> .</td>
</tr>
<tr>
<td>- <strong>Prompt</strong></td>
<td>displays a message window to confirm that you want to replace or specify a new location for the existing log and output files before submitting the background job. If you select <strong>Cancel</strong>, the background job submission is canceled.</td>
</tr>
</tbody>
</table>
### Setting Query Preferences

From the **Query** page, you can set the options for queries. These options are available only if you are running the Standard perspective.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Generate FedSQL</strong></td>
<td>Uses PROC FEDSQL to generate query code. If you do not select this option, your query code is generated using PROC SQL. This option is not selected by default. For more information, see “Generating a FedSQL Query” on page 79.</td>
</tr>
<tr>
<td><strong>Columns to display on Select tab</strong></td>
<td>Specifies the columns that are displayed on the Select tab in addition to the table and column name.</td>
</tr>
<tr>
<td><strong>Show Preview section</strong></td>
<td>Specifies whether to display the preview section of the query tab after you run a query and which tabs to include in the preview.</td>
</tr>
</tbody>
</table>

### Setting Import Preferences

From the **Import** page, you can set the options for importing data. These options are available only if you are running the Standard perspective.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Show Preview section</strong></td>
<td>Specifies whether to display the preview section of the import tab after you import data and which tabs to include in the preview.</td>
</tr>
</tbody>
</table>
Building an Expression

You can use the Expression Builder when you are creating a filter on your data or when you are creating a filter, join, or calculated column in a query. The Expression Builder window contains three main areas:

- **Data and Functions area**: provides access to the columns in your data and the functions that you can use.

- **Expression area**: displays the expression as you build it. You can also type an expression in this area.

- **Values and Log area**: the Values tab displays distinct values for columns. The Log tab displays the validation log. If you are using the Expression Builder to create a calculated column, this area also includes a Properties tab.

You can add columns and functions to the expression by double-clicking the name. You can also add column names by right-clicking the column and selecting **Add to expression** or by dragging the column to the expression box.

To populate a list of distinct values for the column, select the column on the Data tab, and then click **Get Values** on the Values tab. You can add values to the expression by double-clicking the value, clicking **Add to Expression** on the toolbar, or dragging the value to the expression box.

**Note:** You can also type the expression directly in the expression box located above the table viewer.

To validate the syntax of your expression, click **Validate**. The Log tab displays any errors in your expression syntax.

**Note:** The option to validate your syntax is available only in the Standard perspective. For more information, see “Understanding Perspectives” on page 15.
Note: The = operator does not give expected results with decimal values. To create a filter with decimal values, use the Between operator.
# About the Text Encoding to Language Mappings

The following table lists some of the text encoding options and the languages they are associated with. For more information, see “Setting General Preferences” on page 146.

## Text Encoding Options and Language Mappings

<table>
<thead>
<tr>
<th>Text Encoding Option</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows-1250</td>
<td>(Central European languages): Polish, Czech, Slovak, Hungarian, Slovenian, Serbian Latin, Croatian, Bosnian, Romanian, Albanian</td>
</tr>
<tr>
<td>Windows-1251</td>
<td>(Cyrillic languages): Russian, Byelorussian, Bulgarian, Serbian Cyrillic, Macedonian, Ukrainian</td>
</tr>
<tr>
<td>Text Encoding Option</td>
<td>Language</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------</td>
</tr>
<tr>
<td>Windows-1252</td>
<td>(Western European languages): Afrikaans, Basque, Catalan, Valencian, Welsh, Danish, German, English, Spanish, Basque, Finnish, Faroese, French, Western Frisian, Irish, Galician, Indonesian, Icelandic, Italian, Inuktitut, Luxembourgish, Malay, Norwegian Bokmål, Dutch, Norwegian Nynorsk, Portuguese, Quechua, Romansh, Northern Sami, Swedish, Swahili, Tswana, Xhosa, Zulu</td>
</tr>
<tr>
<td>Windows-1253</td>
<td>Greek</td>
</tr>
<tr>
<td>Windows-1254</td>
<td>Turkish</td>
</tr>
<tr>
<td>Windows-1255</td>
<td>Hebrew</td>
</tr>
<tr>
<td>Windows-1256</td>
<td>Arabic</td>
</tr>
<tr>
<td>Windows-1257</td>
<td>(Baltic languages): Estonian, Latvian, Lithuanian</td>
</tr>
<tr>
<td>Windows-1258</td>
<td>Vietnamese</td>
</tr>
</tbody>
</table>
Overview

You must customize the SAS Studio output environment to perform any of these tasks:

- generate output for other output destinations
- send your results to another location
- use a custom style for your output
- use an image format other than the default
- create a drill-down graph
- create an animated GIF or SVG image

To customize the SAS Studio output environment, first disable the default output environment in order to conserve system resources. Next, establish your own output environment, and then execute the SAS statements that are required to generate your output. Use ODS statements, ODS procedures, or ODS options in your SAS program to define the environment that you need.

As a best practice, if your SAS program requires a customized output environment in SAS Studio, your program should always perform these steps:

1. Create a file reference for your ODS output. You can use the &_SASWS_ macro variable that is defined in SAS Studio to reference your home directory as shown in the following statement:

   ```
   filename odsout "&_SASWS_/charts";
   ```

   If you want to store your image files in a separate directory, create a second file reference for your image files as shown in the following statement:
filename ods1out "&_SASWS_/charts/images";

Note: The directories that you specify must already exist, and you must have Write access to the directories.

2 To conserve system resources, disable the default output environment by using the following statement:

ods _all_ close;

3 Open the desired ODS destination. Use the PATH= option to specify the file reference that you created for your ODS output. If you created a separate file reference for your image files, use the GPATH= option to specify the image output file reference. Here is an example:

ods html path=odsout gpath=ods1out file="saleschart.html";

4 Execute the SAS statements that are required to generate your output.

5 Close your ODS destination.

When you disable the default SAS Studio output environment, results are no longer displayed on the Results tab for the duration of your program. The results are generated only by the ODS destination that you open.

Generate Output for Other Output Destinations

If you need to generate output other than the default HTML5, PDF, or RTF output, you must open your own ODS destination. Examples of output destinations include HTML, PowerPoint, and LISTING. After you disable the default output environment, use an ODS statement to open your own output destination. Here is an example:

filename odsout "&_SASWS_/charts";
ods _all_ close;
ods powerpoint path=odsout file="filename";

To access the dictionary of ODS statements, see SAS Output Delivery System: User’s Guide.

Send Your Results to Another Location

When you execute a program in SAS Studio, you can download the output from the Results tab to your local machine. If you want to send your output directly to another location, you must open your own ODS destination. By default, output files that are generated by the ODS destinations that you open are written to your home directory.

If you want to send the results to a specific location, use a FILENAME statement to define a file reference to the desired location. You can use the &_SASWS_ macro
variable to reference your home directory. After you create the file reference, use the PATH=\texttt{file-reference} option in your ODS statement. Here is an example:

```sas
filename odsout "&_SASWS_/charts";
ods _all_ close;
ods html path=odsout file="sales.htm";
```

In this case, file sales.htm and any image files that are generated are written to subdirectory \texttt{charts} in your home directory.

---

### Use a Custom Style for Your Output

When you need to use a custom ODS style such as a corporate style for your results in SAS Studio, you must open your own ODS destination. You cannot specify a custom style for the default results. Use the \texttt{STYLE=} option in your ODS statement to specify your custom style. Here is an example:

```sas
filename odsout =&_SASWS_/charts;   
ods _all_ close; 
ods html path=odsout file="filename.htm" style=style-name;
```

To create a custom style, use the ODS TEMPLATE procedure, CSSStyles, or the \texttt{STYLE=} option. For more information, see \textit{SAS Output Delivery System: User's Guide}.

---

### Use an Image Format Other Than the Default

When you need to use an image format other than the default, you must specify the desired output format, and then open your own ODS destination. To specify the image format:

- If you are using SAS/GRAPH to create your graphs, specify the \texttt{DEVICE=} option in an OPTIONS or GOPTIONS statement. For more information, see \textit{SAS/GRAPH: Reference}.

\begin{verbatim}
Note: The ACTIVEX and JAVA graphics output devices are not recommended in SAS Studio 5.2. Your graphs might not work as expected if you use these output devices.
\end{verbatim}

- If you are using ODS Graphics to create your graphs, specify the \texttt{OUTPUTFMT=} option in an ODS GRAPHICS statement. For more information, see \textit{SAS Output Delivery System: User's Guide}.
Create a Drill-down Graph

When you need to create a drill-down graph in SAS Studio, you must open your own ODS destination. Drill-down graphs provide a convenient means for users to explore complex data. In a drill-down graph, certain elements of the graph contain active links. When a user clicks a linked element, the linked resource appears in a new browser window by default.

For more information, see the following documents:

- If you are using SAS/GRAPH to create the graph, see SAS/GRAPH: Reference.
- If you are using the Graph Template Language to create the graph, see SAS Graph Template Language: User’s Guide.

Create an Animated GIF or SVG Image

When you need to create an animated graph in SAS Studio, you must open your own ODS destination. An animated graph displays a series of charts automatically when the graph is viewed in a web browser or other viewer that supports animation. The animation plays as a sequence of graphs in a slide-show fashion with a delay between each graph. The sequence can play only one time, loop a fixed number of times and then stop, or loop indefinitely.

For more information, see the following documents:

- If you are using SAS/GRAPH to create the graph, see SAS/GRAPH: Reference.
- If you are using the Graph Template Language to create the graph, see SAS Graph Template Language: User’s Guide.
Appendix 5

SAS Studio Command Line

About the Command Line Interface .......................................................... 165
Commands in Standard Perspective ......................................................... 166
Commands in Interactive Perspective ....................................................... 171
Specifying the Path to the Content: SAS Content or SAS Compute .......... 175

About the Command Line Interface

You can use the command line interface to access SAS Studio using the keyboard to control the application. To open the command line, select View ⇒ Command or press Alt+M in Windows environments and Option+M in MacOS environments.

A command can have options and arguments. Options have a dash at the beginning. Arguments that contain white space characters must be enclosed in quotes. Arguments for a command must be in a specified order, but options can be mixed into the argument list in any order.

Enter the command you want to run in the command line. When you are entering text in the command line, SAS Studio suggests possible commands. To select a command from the list of suggestions, use the up and down arrow keys and press Enter. To run the command, press Enter again. To run multiple commands, enter the commands separated by semicolons.

Commands are executed in order and the next command starts when the previous one completes. If a command does not complete and processing halts, use Ctrl +Shift+X to clear the command queue.

Note:

When a command runs, information is added to the SAS Studio console. If the command fails, an error message appears in the console. To open the console, select View ⇒ Console or press Alt+C.
To view previous commands, press Ctrl+Arrow Up to go up one step in the command history. Press Ctrl+Arrow Down to go down one step in the command history.

To abort the execution of a list of commands:
- In Windows environments, press Ctrl+Shift+X.
- In OS X environments, press Command+Shift+X.

## Commands in Standard Perspective

These commands are available in the standard perspective.

### Table A5.1 Commands in Standard Perspective

<table>
<thead>
<tr>
<th>Command Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BGSUBMIT</td>
<td>Starts a background submission and runs the code currently selected in the code editor. If no code is selected, all the code is run.</td>
</tr>
<tr>
<td>CANCEL</td>
<td>Stops running the current submission.</td>
</tr>
<tr>
<td>CLEAR</td>
<td>Deletes the content in the Log and Results tabs.</td>
</tr>
<tr>
<td>CLEARCODE</td>
<td>Deletes all code from the Code tab.</td>
</tr>
<tr>
<td>Command Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>CLEARHISTORY</td>
<td>Deletes submission history.</td>
</tr>
<tr>
<td>CLEARLOG</td>
<td>Deletes the content in the <strong>Log</strong> tab.</td>
</tr>
<tr>
<td>CLEARRESULTS</td>
<td>Deletes the content in the <strong>Results</strong> tab.</td>
</tr>
<tr>
<td>CLOSE</td>
<td>Closes the selected tab.</td>
</tr>
<tr>
<td>CLOSEALL</td>
<td>Closes all tabs.</td>
</tr>
<tr>
<td>CONSOLE</td>
<td>Opens the console for SAS Studio.</td>
</tr>
</tbody>
</table>
| DEBUG          | Starts the DATA Step debugger.  
                 **Note:** The cursor must in the DATA step code for this command to work. |
| EXPORTTABLE <path-to-output-folder> <filename.ext> <-csv | -dlm | -tab><-force> | Opens the Export Table window, when the data view is open and if output path is specified.  
If you specify the output path and filename of the table view, the Export Window does not open. For more information, see “Specifying the Path to the Content: SAS Content or SAS Compute” on page 175.  
These options are available:  
- `-csv` | `-dlm` | `-tab` specifies the export type for the file. If this option is not specified, the export type is determined from the extension on the filename.  
- `-force` overwrites an existing file with the same name. |
| FIND target-string <-case | -c> <-up | -u><-word | -w> | Searches for the specified target string in the code. These options are available.  
- `-case` | `-c` specifies that the search is case sensitive. By default, the case is ignored.  
- `-up` | `-u` searches up. By default, the search goes down the file.  
- `-word` | `-w` searches for whole word matches. By default, the search does not check for word boundaries.  
**Note:** Use the NEX command to move to the next instance of the target string. Use the PREV commands to move to the previous instance of the target string. |
<p>| GOTOLINE &lt;line-number&gt; | Moves the cursor to the specified line number. If you do not specify a line number, SAS Studio prompts you for this information. |</p>
<table>
<thead>
<tr>
<th>Command Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>KEYS &lt;edit&gt;</strong></td>
<td>Opens the Keyboard Shortcuts window. If you specify the edit option, the Manage Keyboard Shortcuts window opens enabling you to edit an existing shortcut. From this window, you can also import and export shortcuts and reset all shortcuts.</td>
</tr>
<tr>
<td><strong>NEWIMPORT</strong></td>
<td>Opens an import data window on a new tab in the work area.</td>
</tr>
<tr>
<td><strong>NEWPROGRAM</strong></td>
<td>Adds a new Program tab to the work area.</td>
</tr>
<tr>
<td><strong>NEWQUERY</strong></td>
<td>Opens a query window on a new tab in the work area.</td>
</tr>
<tr>
<td><strong>NEWTASKTEMPLATE</strong></td>
<td>Opens a blank task template on a new tab in the work area.</td>
</tr>
<tr>
<td><strong>NEXT</strong></td>
<td>Finds the next occurrence of the pattern specified in the FIND command.</td>
</tr>
<tr>
<td><strong>OPENENCODED</strong></td>
<td>Opens the Open File window, if no options are specified. This window does not open if you specify a filename. To open a specific file, specify the path and filename. For more information, see “Specifying the Path to the Content: SAS Content or SAS Compute” on page 175. Use the -encoding option to specify the encoding of the file (for example, -UTF8).</td>
</tr>
<tr>
<td><strong>OPENFILE</strong></td>
<td>Opens the Open File window, if no filename is specified. This window does not open if you specify a filename. To open a specific file, enter the path and filename. For more information, see “Specifying the Path to the Content: SAS Content or SAS Compute” on page 175.</td>
</tr>
<tr>
<td><strong>PREFERENCES</strong></td>
<td>Opens the Preferences window.</td>
</tr>
<tr>
<td><strong>PREV</strong></td>
<td>Finds the previous occurrence of the pattern specified in the FIND command.</td>
</tr>
<tr>
<td><strong>REPLACE</strong></td>
<td>Searches for the specified target string in the code and replaces it with the specified content. These arguments are available.</td>
</tr>
<tr>
<td>target-string</td>
<td>-case</td>
</tr>
<tr>
<td>replacement-string</td>
<td>-up</td>
</tr>
<tr>
<td>-w</td>
<td>-w searches for whole word matches. By default, the search does not check for word boundaries.</td>
</tr>
<tr>
<td>-all</td>
<td>-a replaces all occurrence of the target string.</td>
</tr>
<tr>
<td>Command Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>RESET</strong></td>
<td>Resets the SAS Studio session.</td>
</tr>
</tbody>
</table>
| **SAVE** <output-path> <filename.ext> <-force> <-encoding> | Opens the Save As window, if no output path is specified. This window does not open if you specify an output path. For more information, see “Specifying the Path to the Content: SAS Content or SAS Compute” on page 175. If no filename is specified, the file keeps its original filename. To change the filename, you must specify the filename.ext argument. You can also specify these options:  
- **-force** overwrites an existing file with the same name.  
- **-encoding** specifies the encoding of the file, such as UTF8. |
| **SAVEALL**  | Saves all open files. |
| **SAVECODE** <output-path> <filename.ext> <-force> <-encoding> | Opens the Save As window, if no output path is specified. This window does not open if you specify an output path. For more information, see “Specifying the Path to the Content: SAS Content or SAS Compute” on page 175. If no filename is specified, the original filename is used. To change the filename, you must specify the filename.ext argument. You can also specify these options:  
- **-force** overwrites an existing file with the same name.  
- **-encoding** specifies the encoding of the file, such as UTF8. |
| **SAVELOG** <output-path> <filename.ext> <-force> <-encoding> | Opens the Save As window, if no output path is specified. This window does not open if you specify an output path. For more information, see “Specifying the Path to the Content: SAS Content or SAS Compute” on page 175. If no filename is specified, the original filename is used. To change the filename, you must specify the filename.ext argument. You can also specify these options:  
- **-force** overwrites an existing file with the same name.  
- **-encoding** specifies the encoding of the file, such as UTF8. |
<table>
<thead>
<tr>
<th>Command Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **SAVEPACKAGE** | <output-path>  
<filename.ext>  
<-force>  
<-encoding>  

Opens the Save As window, if no output path is specified. This window does not open if you specify an output path. For more information, see “Specifying the Path to the Content: SAS Content or SAS Compute” on page 175. If no filename is specified, the original filename is used. To change the filename, you must specify the `filename.ext` argument. You can also specify these options:  
- `-force` overwrites an existing file with the same name.  
- `-encoding` specifies the encoding of the file, such as `-UTF8`. |
| **SAVERESULTS** | <output-path>  
<filename.ext>  
<-force>  
<-encoding>  

Opens the Save As window, if no output path is specified. This window does not open if you specify an output path. For more information, see “Specifying the Path to the Content: SAS Content or SAS Compute” on page 175. If no filename is specified, the original filename is used. To change the filename, you must specify the `filename.ext` argument. You can also specify these options:  
- `-force` overwrites an existing file with the same name.  
- `-encoding` specifies the encoding of the file, such as `-UTF8`. |
| **SAVESUMMARY** | <output-path>  
<filename.ext>  
<-force>  
<-encoding>  

Opens the Save As window, if no output path is specified. This window does not open if you specify an output path. For more information, see “Specifying the Path to the Content: SAS Content or SAS Compute” on page 175. If no filename is specified, the file keeps its original filename. To change the filename, you must specify the `filename.ext` argument. You can also specify these options:  
- `-force` overwrites an existing file with the same name.  
- `-encoding` specifies the encoding of the file, such as `-UTF8`. |
| **SEARCH**     | `-max` | `-restore` | `-off` | Displays the Search tab. These options are available:  
- `-max` maximizes the preview area.  
- `-restore` restores the preview area.  
- `-off` closes the Search tab. |
| **SELECTCODE** | Selects the Code tab.                                                                                                                                                                                                                                                                       |
| **SELECTLOG**  | Selects the Log tab.                                                                                                                                                                                                                                                                       |
### Commands in Interactive Perspective

These commands are available in the interactive perspective.

<table>
<thead>
<tr>
<th>Command Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CANCEL</td>
<td>Stops running the current submission.</td>
</tr>
<tr>
<td>CLEAR</td>
<td>Deletes the content in the Log and Results tabs.</td>
</tr>
<tr>
<td>CLEARCODE</td>
<td>Deletes all code from the Code tab.</td>
</tr>
<tr>
<td>CLEARLOG</td>
<td>Deletes the content in the log.</td>
</tr>
<tr>
<td>Command Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CLEARRESULTS</td>
<td>Deletes the content in the <strong>Results</strong> tab.</td>
</tr>
<tr>
<td>CLOSE</td>
<td>Closes the selected tab.</td>
</tr>
<tr>
<td>CLOSEALL</td>
<td>Closes all tabs.</td>
</tr>
<tr>
<td>CONSOLE</td>
<td>Opens the console for the command line.</td>
</tr>
<tr>
<td>FIND target-string</td>
<td>Searches for the specified target string in the code. These options are available.</td>
</tr>
<tr>
<td></td>
<td>-case</td>
</tr>
<tr>
<td></td>
<td>-up</td>
</tr>
<tr>
<td></td>
<td>-word</td>
</tr>
<tr>
<td>GOTOLINE &lt;line-number&gt;</td>
<td>Moves the cursor to the specified line number. If you do not specify a line number, SAS Studio prompts you for this information.</td>
</tr>
<tr>
<td>KEYS &lt;edit&gt;</td>
<td>Opens the Keyboard Shortcuts window. If you specify the edit option, the Manage Keyboard Shortcuts window opens enabling you to edit an existing shortcut. From this window, you can also import and export shortcuts and reset all shortcuts.</td>
</tr>
<tr>
<td>NEWIMPORT</td>
<td>Opens an import data window on a new tab in the work area.</td>
</tr>
<tr>
<td>NEWPROGRAM</td>
<td>Adds a new <strong>Program</strong> tab to the work area.</td>
</tr>
<tr>
<td>NEWQUERY</td>
<td>Opens a query window on a new tab in the work area.</td>
</tr>
<tr>
<td>NEWTASKTEMPLATE</td>
<td>Opens a blank task template on a new tab in the work area.</td>
</tr>
<tr>
<td>NEXT</td>
<td>Finds the next occurrence of the pattern specified in the FIND command.</td>
</tr>
<tr>
<td>Command Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| OPENENCODED <path-to-file/filename.ext> <encoding> | Opens the Open File window, if no options are specified.  
This window does not open if you specify a filename. To open a specific file, specify the path and filename. For more information, see “Specifying the Path to the Content: SAS Content or SAS Compute” on page 175.  
Use the -encoding option to specify the encoding of the file (for example, -UTF8). |
| OPENFILE <path-to-file/filename.ext>             | Opens the Open File window, if no filename is specified.  
This window does not open if you specify a filename. To open a specific file, enter the path and filename. For more information, see “Specifying the Path to the Content: SAS Content or SAS Compute” on page 175. |
| PREFERENCES          | Opens the Preferences window.                                                                                                               |
| PREV                 | Finds the previous occurrence of the pattern specified in the FIND command.                                                                  |
| REPLACE target-string replacement-string <-case | -c> <-up | -u> <-word | -w> <-all | -a> | Searches for the specified target string in the code and replaces it with the specified content. These arguments are available.  
- -case | -c specifies that the search is case sensitive. By default, the case is ignored.  
- -up | -u searches up. By default, the search goes down the file.  
- -word | -w searches for whole word matches. By default, the search does not check for word boundaries.  
- -all | -a replaces all occurrence of the target string. |
<p>| RESET                | Resets the SAS Studio session.                                                                                                               |</p>
<table>
<thead>
<tr>
<th>Command Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SAVE</strong> <code>&lt;path-to-parent-folder&gt;</code> <code>&lt;filename.ext&gt;</code> <code>&lt;-force&gt;</code> <code>&lt;-encoding&gt;</code></td>
<td>Opens the Save As window, if no options are specified.</td>
</tr>
<tr>
<td></td>
<td>This window does not open if you specify a filename. To save a specific file, enter the folder name. For more information, see “Specifying the Path to the Content: SAS Content or SAS Compute” on page 175.</td>
</tr>
<tr>
<td></td>
<td>If no filename is specified, the file keeps its original filename. To change the filename, you must specify the filename.ext argument.</td>
</tr>
<tr>
<td></td>
<td>You can also specify these options:</td>
</tr>
<tr>
<td></td>
<td>- <code>force</code> overwrites an existing file with the same name.</td>
</tr>
<tr>
<td></td>
<td>- <code>encoding</code> specifies the encoding of the file, such as <code>-UTF8</code>.</td>
</tr>
<tr>
<td><strong>SAVEALL</strong></td>
<td>Saves all open files.</td>
</tr>
<tr>
<td><strong>SEARCH</strong> `&lt;-max</td>
<td>-restore</td>
</tr>
<tr>
<td></td>
<td>- <code>max</code> maximizes the preview area.</td>
</tr>
<tr>
<td></td>
<td>- <code>restore</code> restores the preview area.</td>
</tr>
<tr>
<td></td>
<td>- <code>off</code> closes the Search tab.</td>
</tr>
<tr>
<td><strong>SHOWLOG</strong> `&lt;max</td>
<td>-restore</td>
</tr>
<tr>
<td></td>
<td>- <code>max</code> maximizes the preview area.</td>
</tr>
<tr>
<td></td>
<td>- <code>restore</code> restores the preview area.</td>
</tr>
<tr>
<td></td>
<td>- <code>off</code> closes the Log tab.</td>
</tr>
<tr>
<td><strong>SHOWRESULTS</strong> `&lt;max</td>
<td>-restore</td>
</tr>
<tr>
<td></td>
<td>- <code>max</code> maximizes the preview area.</td>
</tr>
<tr>
<td></td>
<td>- <code>restore</code> restores the preview area.</td>
</tr>
<tr>
<td></td>
<td>- <code>off</code> closes the Results tab.</td>
</tr>
<tr>
<td><strong>SUBMISSIONS</strong> `&lt;max</td>
<td>-restore</td>
</tr>
<tr>
<td></td>
<td>- <code>max</code> maximizes the preview area.</td>
</tr>
<tr>
<td></td>
<td>- <code>restore</code> restores the preview area.</td>
</tr>
<tr>
<td></td>
<td>- <code>off</code> closes the Submissions tab.</td>
</tr>
<tr>
<td>Command Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SUBMIT</td>
<td>Runs the currently selected code or all the code in the editor if no code is selected.</td>
</tr>
<tr>
<td>VIEWTABLE &lt;library.table&gt; &lt;last&gt;</td>
<td>Opens the Select Table window. Use the library.table argument to specify the table to view. If no table is specified, use the -last option to open the last table that was generated in the current session.</td>
</tr>
<tr>
<td>ZOOM &lt;-max&gt;&lt;-restore&gt;</td>
<td>Specifies the size of the tab area. These arguments are available. -max specifies to maximize the size of the tab area. -restore returns the tab area to the default size.</td>
</tr>
</tbody>
</table>

**Specifying the Path to the Content: SAS Content or SAS Compute**

When specifying the file path for these commands, you must prefix the directory path argument with sascontent or sascompute to specify where the file or folder is located.

- EXPORTTABLE
- OPENFILE
- OPENCODED
- SAVE
- SAVECODE
- SAVELOG
- SAVERESULTS
- SAVESUMMARY
- SAVEPACKAGE

Here are some examples using the EXPORTTABLE command:

```plaintext
exporttable "sascontent:/Users/userID/My Folder/Import" "air.csv"
exporttable "sascompute:/your-server.com/userID/SAS Studio Content/Import" "air.csv"
```

If you are working in the programming only environment and only have the SAS Compute Server, sascompute is the default, so you can omit the sascompute prefix.
exporttable "/your-server.com/userID/SAS Studio Content/Import" "air.csv"

In all other environments, sascontent is the default, so you can omit the sascontent prefix.

exporttable "/Users/userID/My Folder/Import" "air.csv"