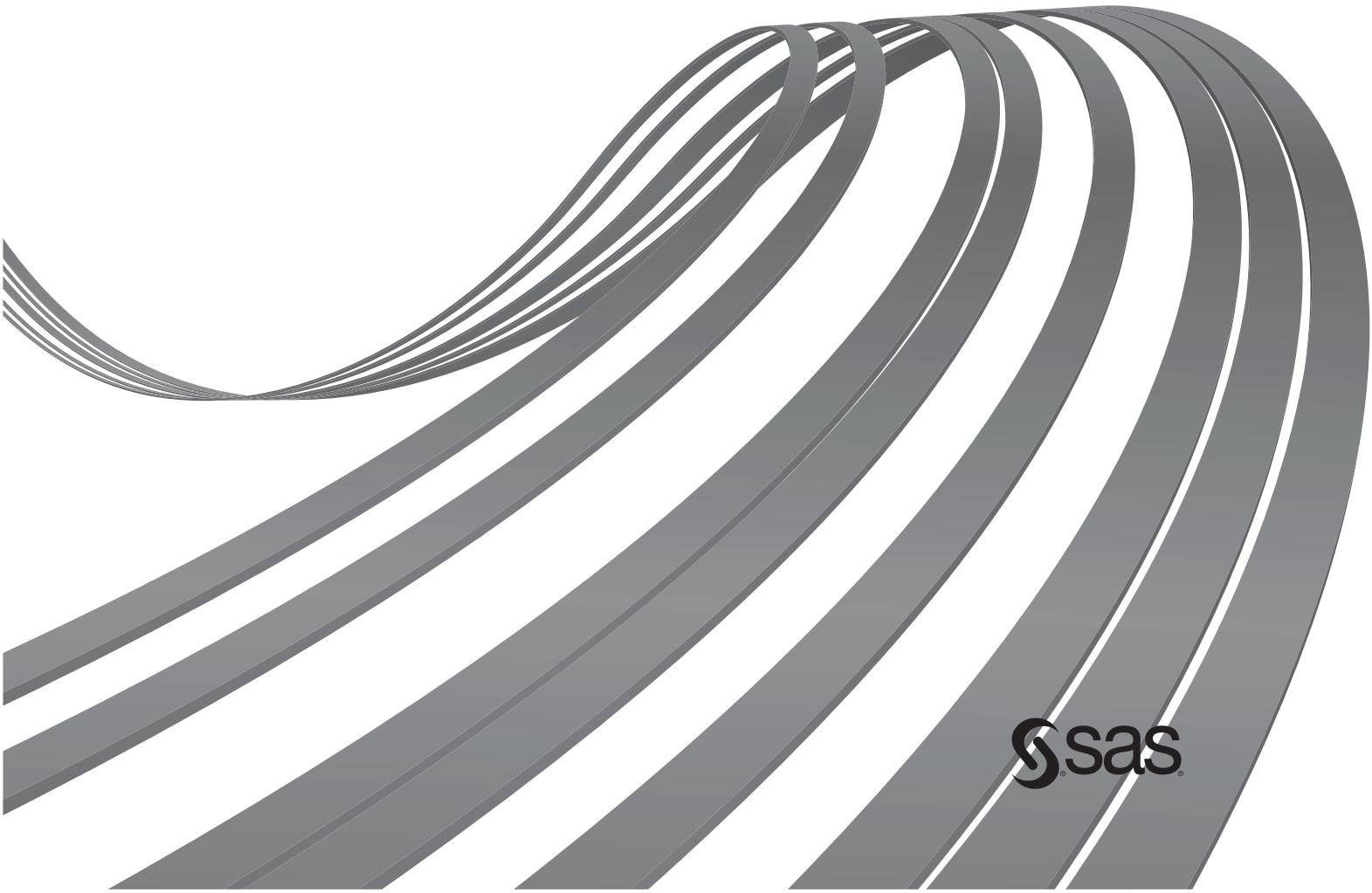
sas innovate

How Do I Modify SAS®9 Programs to Run in SAS® Viya®?

Stacey Syphus
Director, SAS Education



How Do I Modify SAS®9 Programs to Run in SAS® Viya®?

Explore the SAS Studio Interface

1. Launch Google Chrome from the desktop and select **SAS Studio** from the Bookmarks bar.
2. Enter **student** in the **User ID** field and **Metadata0** in the **Password** field. Click **Sign in**. Select **Yes** when prompted to opt in to all assumable groups.
3. Click the **applications menu** icon  in the upper left corner. The applications menu shows all the available applications. Depending on your environment, these applications might differ. Select **Develop Code and Flows** to open SAS Studio. We use SAS Studio to write and submit code in this workshop.
4. The Start Page tab is open by default, and it provides convenient links to start writing a program or use other point-and-click tools, including importing and querying data. Click **Program in SAS** to open a new tab with the Program Editor. Each item that you open in SAS Studio has its own tab.
5. The navigation pane is on the left side of SAS Studio. Each section in the navigation pane provides access to different resources in your SAS environment.
    - **Open Files**  enables you to quickly view and access files that are currently open in your SAS Studio session. You can see that the list of open files matches the tabs.
    - The **Explorer**  provides access to files and folders in your SAS environment. Keep in mind that the files and folders that you see here are not on your local machine. They're on the server where SAS is running. If you have the available permissions, you can use the **Upload files**  button to load local files to the SAS server. The Explorer also enables you to perform other tasks, such as background submit and schedule programs and basic file operations, such as copy, paste, and delete.
    - **Steps**  enables you to build flows in SAS Studio. In a flow, you can build a sequence of steps, including programs, queries, and data transformations, as well as your own custom steps.
    - **Tasks**  enables you to access tasks in SAS Studio. Tasks are based on SAS procedures and generate SAS code and formatted results for you.
    - **Snippets**  enables you to access snippets. Snippets are lines of commonly used code or text that you can save and reuse. SAS Studio provides several predefined snippets, and you can create your own.
    - **Libraries**  provides access to your SAS libraries. You can open SAS tables and add them to your programs. You can even expand a table and view its columns.
    - **Git Repositories**  enables you to access the basic Git features from within SAS Studio for maintaining and viewing a history of program edits.

Modify a SAS®9 Program to Execute on the SAS Compute Server

6. In the Explorer, expand **Files** > **Home** > **ViyaProgramming**. Double-click **ComputeCAS_Start.sas**.
7. Section 1 of the program includes code that was originally written for SAS®9. When executed in SAS Viya, the program is processed on the Compute Server. Examine the program and notice that it includes traditional code to access data, create a new table, and generate summary reports.

8. What needs to change to run this program in SAS Viya? The path in the LIBNAME statement must be updated to reference the location of the data in the new environment. Highlight the path in quotation marks and then use the Explorer to right-click the **ViyaProgramming** folder and select **Insert as path**.

```
libname mydata "/home/student/ViyaProgramming";
```

9. No additional code modifications are necessary. Highlight the code in Section 1 and click **Run**.
10. This program executes on the Compute Server. It processes the data just like the SAS®9 Workspace Server and returns results for the CONTENTS and FREQ procedures, and output tables for the DATA step and the MEANS procedure.
11. Examine the log and note the execution time for each step.

Modify a SAS Program to Execute in CAS

Start a CAS Session

12. Highlight the program in Section 1 and press CTRL+C. Select **New** > **SAS Program**. Click in the new program tab and press CTRL+V to paste the code.
13. Insert a line at the top of the program and add the following CAS statement to initiate a CAS session:

```
cas mySession;

libname mydata "/home/student/ViyaProgramming";
```

Define a Caslib

14. The **Orders** table must be loaded into memory. First, you must define a caslib that enables CAS to read and load the data source file from disk. The caslib points to the location where your data source files are stored. After the LIBNAME statement, add the following CASLIB statement to define a caslib named **mycas** that points to the workshop folder. The LIBREF= option maps a library reference to the caslib that can be used in DATA and PROC steps. It is recommended that the libref match the caslib name.

```
libname mydata "/home/student/ViyaProgramming";
caslib mycas path="/home/student/ViyaProgramming" libref=mycas;
```

15. Highlight and run the CAS, LIBNAME, and CASLIB statements. Confirm in the log that the **mycas** caslib was added. Select the Libraries pane. Notice that **mycas** is displayed with a cloud icon, indicating the library is mapped to a caslib. **mycas** appears empty because there are no in-memory tables loaded yet.

Load Table into Memory

16. In this demo, we load a SASHDAT file into memory. The SASHDAT file format is optimized for loading data in CAS to the available worker nodes. It supports parallel loading, meaning that data can be loaded simultaneously to the multiple workers, which can significantly reduce load time.
17. In the Snippets pane, expand **Viya Foundation** > **Cloud Analytic Services** and double-click **Load Data to caslib**. There are multiple options to load data into memory. This sample code includes three scenarios for loading data and template code for each one. Highlight the last PROC CASUTIL step and press CTRL+C to copy the code.
18. Return to the **SAS Program 1.sas** tab and press CTRL+V to paste the snippet code after the CASLIB statement. Modify the values in quotation marks to read the **orders.sashdat** file from the **mycas** caslib and then create an in-memory table in the **mycas** caslib named **orders**. Add the REPLACE option to replace the in-memory table if it exists.

```
proc casutil;
     load casdata="orders.sashdat" incaslib="mycas"
          outcaslib="mycas" casout="orders" replace;
run;
```

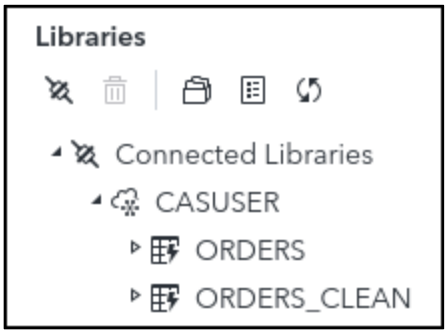
19. Highlight and run the CASUTIL procedure step. Examine the log and confirm that the **Orders** table is available in the **mycas** caslib.

Modify and Load Data Via the DATA Step

20. Another method to load data into memory is the DATA step. This enables you to make modifications to the table before loading it to CAS. If you designate an in-memory table in the DATA statement as your output table, the step executes on the Compute Server, but the result is loaded into memory.
21. Modify the DATA statement to write the output table to **mycas**. Highlight the DATA step and click **Run**.

```
data mycas.orders_clean;
```

22. This step processes on the Compute Server, but the result is an in-memory table available in CAS. Expand **Mycas** in the Libraries pane and confirm that **ORDERS** and **ORDERS_CLEAN** are listed. The lightning icon indicates in-memory tables.



Process In-Memory Data in CAS

DATA Step

23. If both the input table in the SET statement and the output table in the DATA statement are in-memory tables, the entire step can process in CAS. Change the input table to **mycas.orders** and then rerun this DATA step. The log indicates that the step ran in Cloud Analytic Services. For complex DATA steps and large tables, this can result in significantly faster execution.

```
data mycas.orders_clean;
    set mycas.orders;
    ...
```

Procedures

24. Many SAS procedures have corresponding CAS-enabled procedures that process in-memory data. SAS documentation provides a helpful comparison list. In Chrome, add a new browser tab. From the bookmarks bar, select **Documentation** > **CAS Procedures**.
25. The SAS Procedures and Corresponding CAS Procedures and Actions page provides a helpful comparison of procedures. Notice that for the SAS procedure PROC FREQ, the equivalent CAS procedure is PROC FREQTAB. The corresponding CAS procedure for PROC MEANS is PROC MDSUMMARY.
26. Return to the SAS Studio tab in Chrome. On the SAS Program tab, make the following changes:
    a. Change **FREQ** to **FREQTAB**.
    b. Change **MEANS** to **MDSUMMARY**.
    c. Change **Mydata** to **Mycas** to reference in-memory tables in the CONTENTS, FREQTAB, and MDSUMMARY procedures.

```
proc contents data=mycas.orders

proc freqtab data=mycas.orders;
    tables Country OrderType;
run;

proc mdsummary data=mycas.orders;
    var RetailPrice;
    output out=mycas.orders_sum;
run;
```

27. Highlight and run the three procedures and TITLE statements. The results look the same compared to the previous program. However, the log indicates that PROC FREQTAB and PROC MDSUMMARY executed in CAS. Behind the scenes, these steps are converted to CAS actions and then executed in CAS.

28. Select the Libraries pane and expand **Mycas**. Confirm that **ORDERS**, **ORDERS_CLEAN**, and **ORDERS_SUM** are listed. These tables are defined as session scope, which means that they are available only in SAS Studio and in the current CAS session.

29. Add the following CAS statement after the TITLE statement to terminate the session. Ending the CAS session drops session-scope tables from memory and de-assigns all librefs mapped to caslibs.

```
title;

cas mySession terminate;
```

30. Highlight and run the CAS statement. Confirm that the **Mycas** caslib no longer appears in the Libraries list.

Note: Tables promoted to global scope persist in memory when the CAS session ends and are available in other SAS Viya applications. To promote the table using PROC CASUTIL, add the PROMOTE option in the LOAD statement.

```
proc casutil;
   load data=mydata.orders casout="orders" outcaslib="mycas"
        promote;
quit;
```

Or you can also promote a table in the DATA step with the PROMOTE=YES data set option.

```
data mycas.orders_clean(promote=yes);
```

CASL and CAS Actions

31. CAS-enabled procedures and DATA steps are converted to CAS actions behind the scenes and submitted to CAS for execution. You can write CASL code in PROC CAS to submit your own CAS actions directly. Return to the **ComputeCAS_Start.sas** tab and scroll down to Section 3. Examine the code in PROC CAS and the following statements.
    a. A dictionary named **TBL** that references the **Orders** table in **Mycas** is defined.
    b. The SOURCE statement begins a block of code that stores the DATA step code in the **DS** variable.
    c. The table.addCaslib action defines a caslib pointing to the workshop files.
    d. The table.dropTable action drops the in-memory **Orders** table from **Mycas** if it exists.
    e. The table.loadTable action reads the **orders.sashdat** file from the **mycas** caslib and creates an in-memory table named **Orders** in the **Mycas** caslib.
    f. The dataStep.runCode action executes the DATA step code stored previously in the **DS** variable.
    g. The table.columnInfo action generates a report with column attributes, similar to PROC CONTENTS.
    h. The simple.freq action generates a report with frequency counts for **Country** and **OrderType**, similar to PROC FREQ or PROC FREQTAB.
    i. The simple.summary action creates an output table named **orders_sum** that includes summary statistics for **RetailPrice**, similar to PROC MEANS or PROC MDSUMMARY.
32. Highlight the code in Section 3 and click **Run**. Examine the log and output.

Processing Time Comparison

33. In the Explorer, double-click **ComputeCAS_Benchmark.sas** to open the program. This program includes the complete code for the Compute Server steps, the CAS-enabled steps, and CASL. It also includes total timing information in the log for each scenario.
34. Click **Run** to execute the entire program. Examine the log and compare run times for each scenario. Your actual times will vary, but you should notice that the CAS-enabled steps and CASL significantly reduce execution time compared to the Compute Server program.