




# How Do I Modify SAS<sup>®</sup> 9 Programs to Run in SAS<sup>®</sup> Viya<sup>®</sup>?

Stephen Foerster  
Data Management Lead, 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.

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

5. In the Explorer, expand **Files > Home > workshop > HODV**. Double-click **ComputeCAS\_Start.sas**.
6. 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.
7. What needs to change to run this program in SAS Viya? The path in the LIBNAME statements must be updated to reference the location of the data in the new environment. Change the provided paths to those shown below.

```
libname srcData "/home/student/workshop/HODV";  
libname outpData "/home/student/workshop/";
```

8. No additional code modifications are necessary. Highlight the code in Section 1 and click **Run**.
9. 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.
10. Examine the log and note the execution time for each step.

## Modify a SAS Program to Execute in CAS

### Start a CAS Session

11. 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.
12. Insert a line at the top of the program and add the following CAS statement to initiate a CAS session.

```
cas mySess sessopts=(metrics=true) ;
```

### Define a Caslib

13. 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. 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.

```
caslib mycas path="/home/student/workshop/HODV" libref=mycas;
```

14. Highlight and run the CAS 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 a Table into CAS

15. In this demo, we load a regular SAS Data Set (sas7bdat) into CAS.
16. In the Snippets pane, expand SAS Viya 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.
17. 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.sas7bdat** 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 CAS if it already exists.

```
proc casutil;  
  load casdata="orders.sas7bdat" incaslib="mycas"  
  outcaslib="mycas" casout="orders" replace;
```

```
run;
```

18. Highlight and run the CASUTIL procedure step. Examine the log. The process takes less than a second to load the data.

### Run a DATA Step in CAS

19. Modify the DATA Step to run in CAS. Change the library references to **Mycas**. For a DATA Step to run in CAS, both the input and output data sets must be in CAS. Highlight the DATA step and click **Run**.

```
data mycas.orders_clean;  
  set mycas.orders;  
  Name=catx(' ',  
            scan(Customer_Name,2,' '),  
            scan(Customer_Name,1,' '));  
run;
```

### Run Procedures in CAS

20. Many SAS procedures have corresponding CAS-enabled procedures
21. On the SAS Program tab, make the following changes:
- Change **FREQ** to **FREQTAB**.
  - Change **MEANS** to **MDSUMMARY**.
  - Change **Mydata** to **Mycas** to reference in-memory tables in the CONTENTS, FREQTAB, and MDSUMMARY procedures.

```
proc contents data=mycas.orders;  
run;  
  
proc freqtab data=mycas.orders;  
  tables Country OrderType;  
run;  
  
proc mdsummary data=mycas.orders;  
  var RetailPrice;  
  output out=mycas.orders_sum;  
run;
```

22. 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.

23. 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.

## CASL and CAS Actions

24. 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 dataStep.runCode action executes the DATA step code stored previously in the **DS** variable.
  - d. The table.columnInfo action generates a report with column attributes, similar to PROC CONTENTS.
  - e. The simple.freq action generates a report with frequency counts for **Country** and **OrderType**, similar to PROC FREQ or PROC FREQTAB.
  - f. The simple.summary action creates an output table named **orders\_sum** that includes summary statistics for **RetailPrice**, similar to PROC MEANS or PROC MDSUMMARY.
25. Highlight the code in Section 3 and click **Run**. Examine the log and output.

## Processing Time Comparison

26. In the Explorer, double-click **CustomerCAS\_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.
27. Click **Run** to execute the entire program. Examine the log and compare run times for each scenario. Your actual times will vary, but should notice that the CAS-enabled steps and CASL significantly reduce execution time compared to the Compute server program.