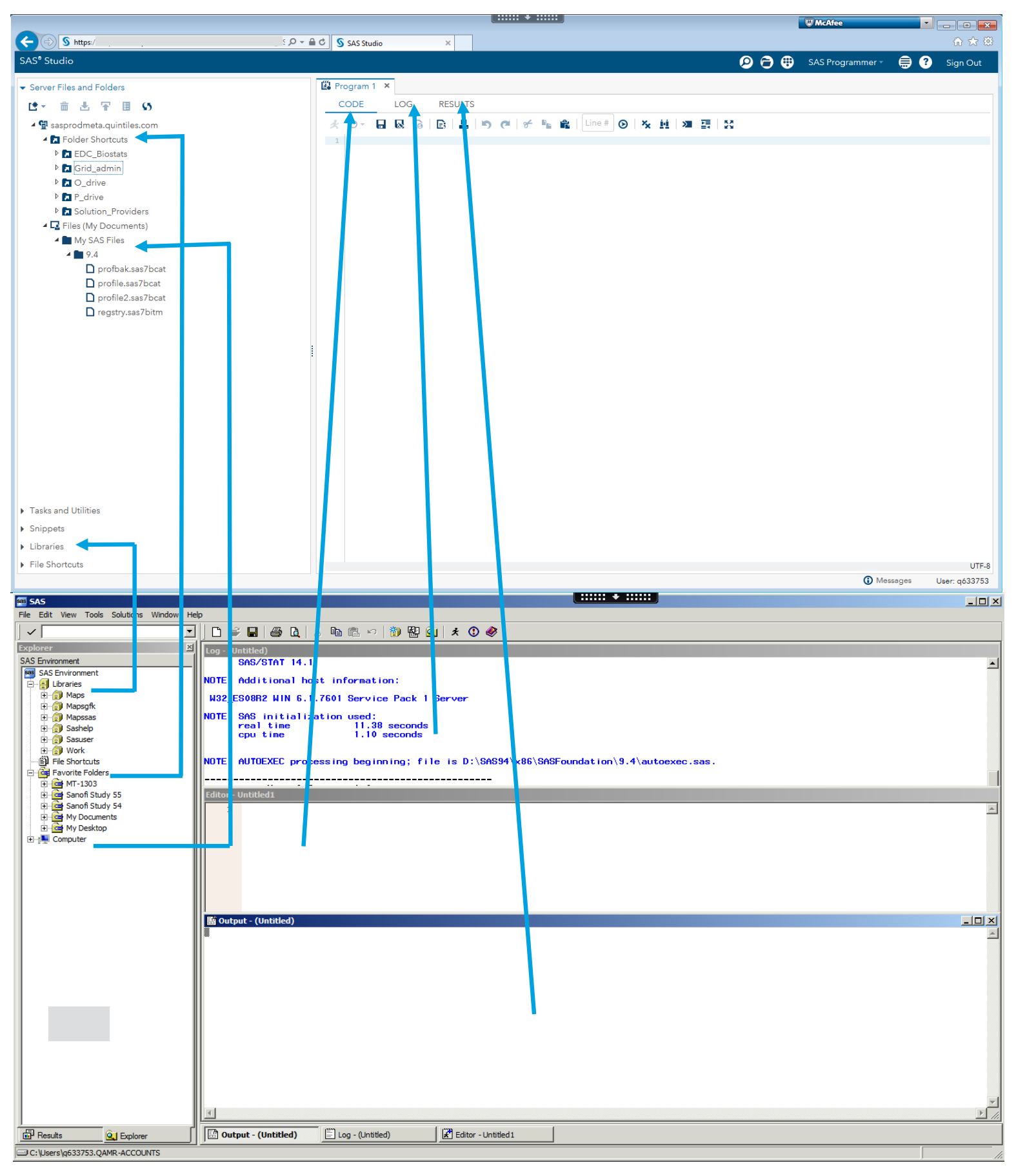
PP27 - Adventures in SAS® Studio

Lawrence Heaton-Wright, QuintilesIMS

With the increasing adoption of SAS Grid environments, the old-school programming environment (SAS Display Manager System – DMS) is unable to operate in a Grid environment due to the way that SAS Grid is deployed. As we move away from directly interacting with SAS we are being forced into using environments like Enterprise Guide (EG) and SAS Studio. As an old-school programmer I am pre-disposed to dislike these new-fangled environments and their fancy code-writing wizards … after all who wants to be replaced by a robot?

Having made the move to a SAS Grid we made the choice to adopt SAS Studio as our front end for access to SAS because of several reasons: there are many similarities between DMS and Studio, the centralised distribution of the web-based Studio and the ability to customise and develop tasks and utilities alongside the centralised deployment of them. SAS Studio is still, at heart, a SAS editor but it has a multitude of added features that can enhance your programming experience.

SAS Studio and DMS Similarities

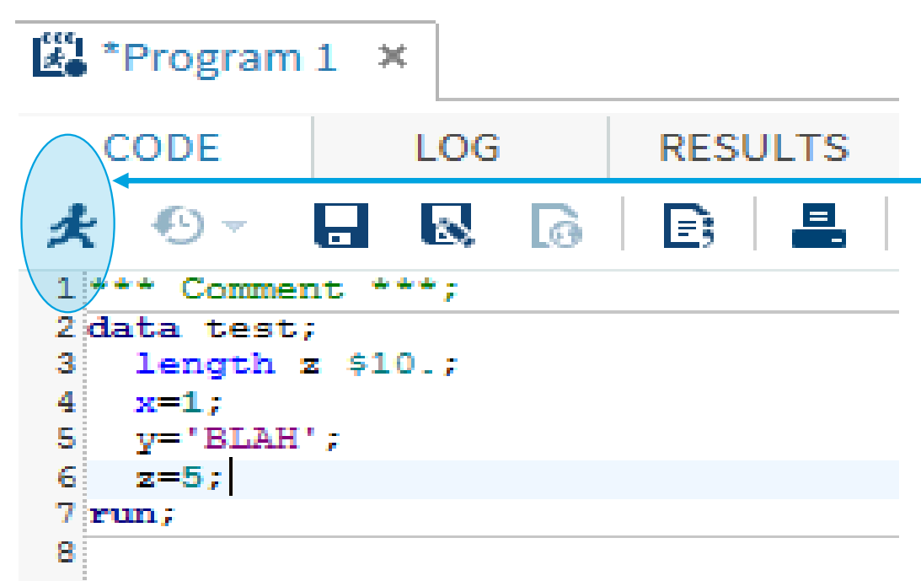


The Studio environment looks quite similar to the classic SAS Display Manager.

It has program, log and output tabs (program, log and output windows previously), files and folders menus, and library browser menus.

It has the additions of "Task and Utilities" and "Snippets" menus.

As with Display Manager, multiple program windows (which contain the Code, Log and Results tabs) can be opened in Studio.

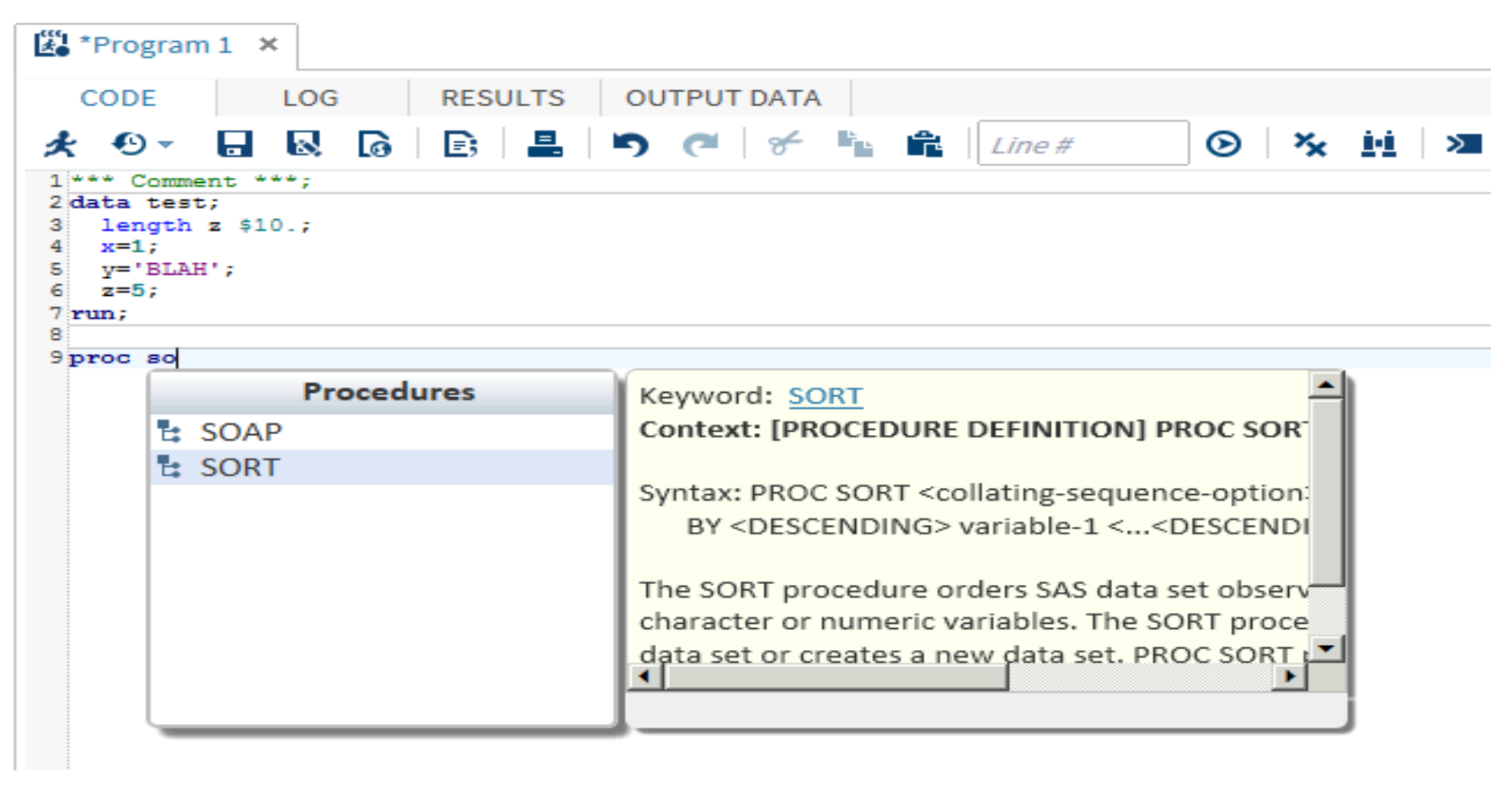


```
*** Comment ***;
data test;
  length z $10.;
  x=1;
  y='BLAH';
  z=5;
run;
```

The code window has the colour-coding that the Enhanced Editor in DMS has.

We can still highlight sections of code and submit those sections using the running man icon

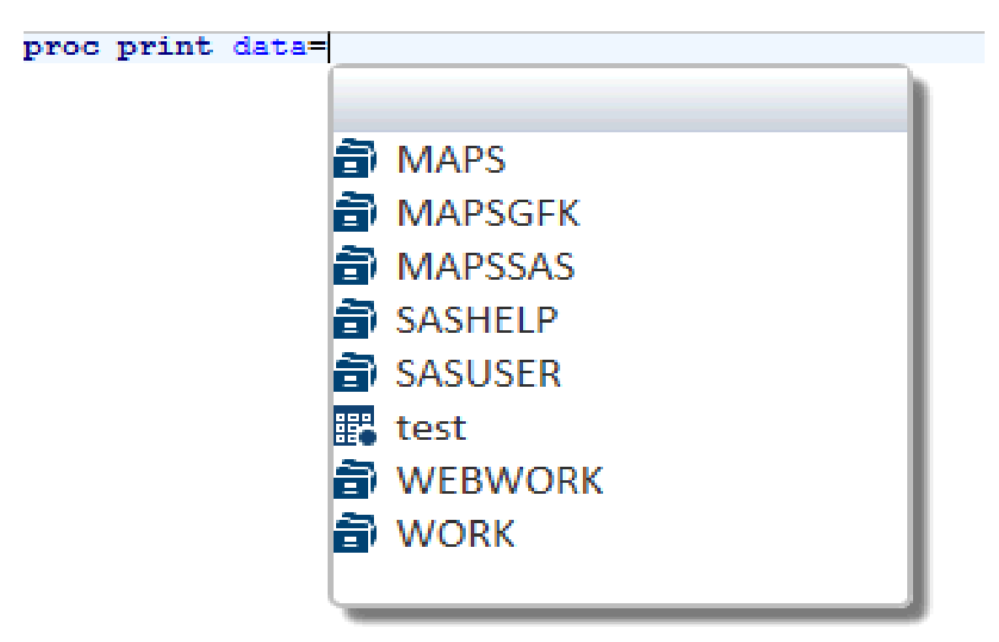
What's New in the Code Editor?



```
*** Comment ***;
data test;
  length z $10.;
  x=1;
  y='BLAH';
  z=5;
run;

proc so
```

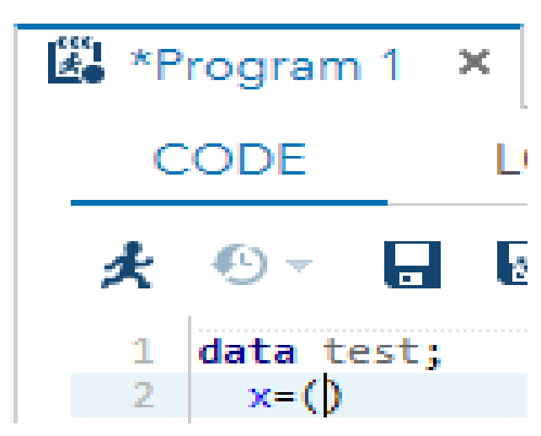
Keyword: SORT
Context: [PROCEDURE DEFINITION] PROC SORT
Syntax: PROC SORT <collating-sequence-option
BY <DESCENDING> variable-1 <...<DESCENDI
The SORT procedure orders SAS data set observ character or numeric variables. The SORT proce data set or creates a new data set. PROC SORT

The Code tab now has code prediction as well as context-sensitive help with links to syntax guides as well as full web-based help documentation.



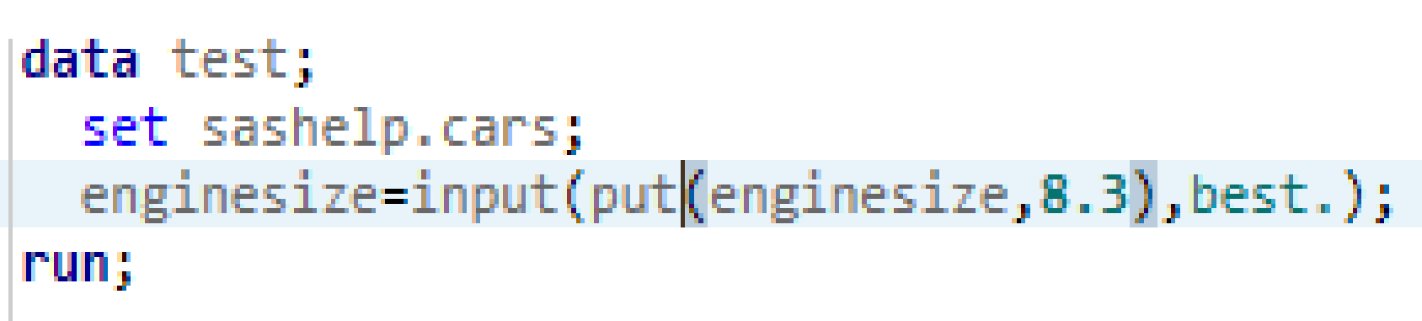
```
proc print data=
```

The auto-complete code prediction shows data libraries and datasets available within the session in a similar manner to EG. Unlike EG variable availability is not yet available. Similar functionality is available for macro variables.



```
data test;
  x=()
```

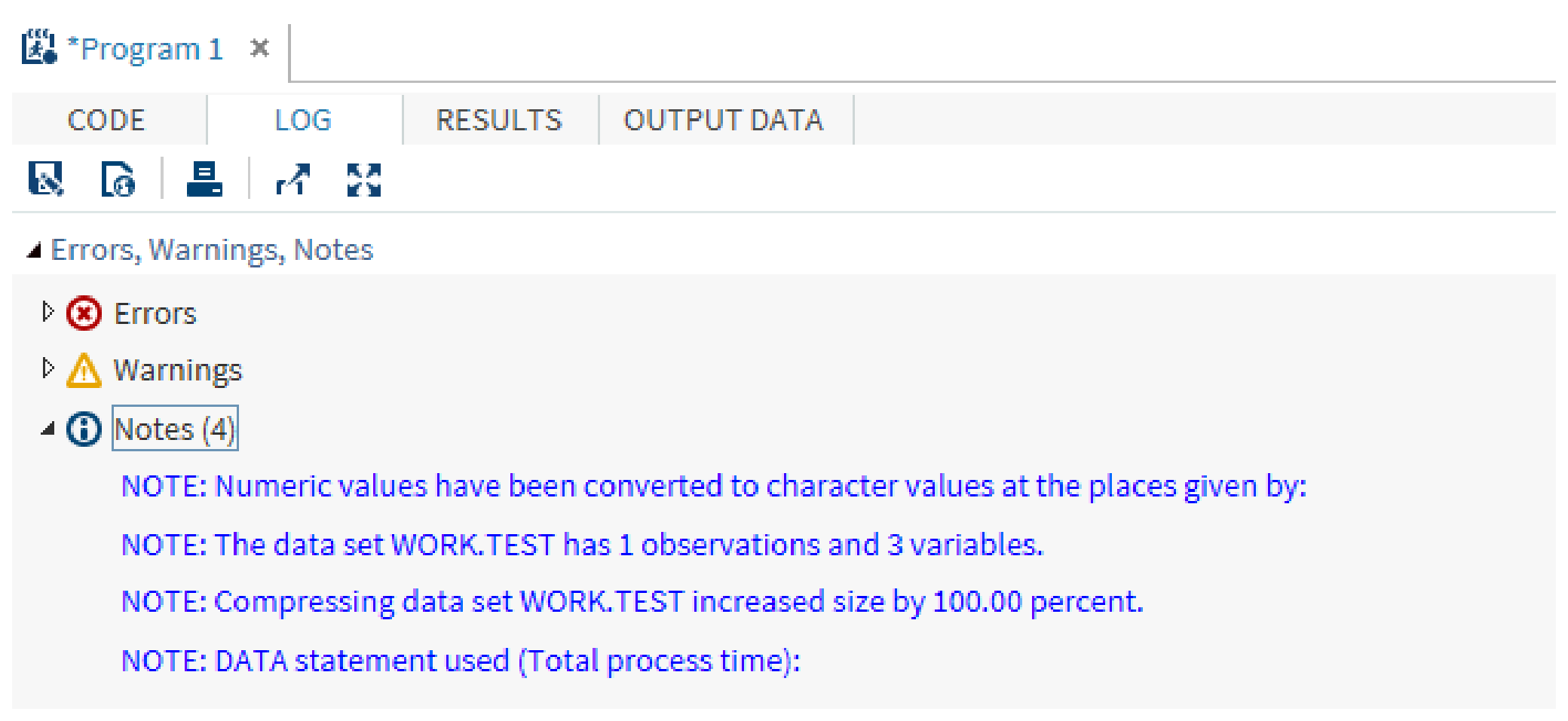
The SAS Studio Code editor has been enhanced (for version 3.6 of Studio) with bracket matching as well as prediction. From this snippet you can see that the cursor is between the 2 parentheses. The first bracket was the only one that was typed. This works for {} and [] as well.



```
data test;
  set sashelp.cars;
  enginesize=input(put(enginesize,8.3),best.);
run;
```

As can be seen from this enhanced snippet, the cursor is placed at the second bracket. The matching end bracket for this is highlighted (as a grey box). It's quite easy to see within the editor and makes identifying unmatched parentheses much easier to spot.

The LOG tab



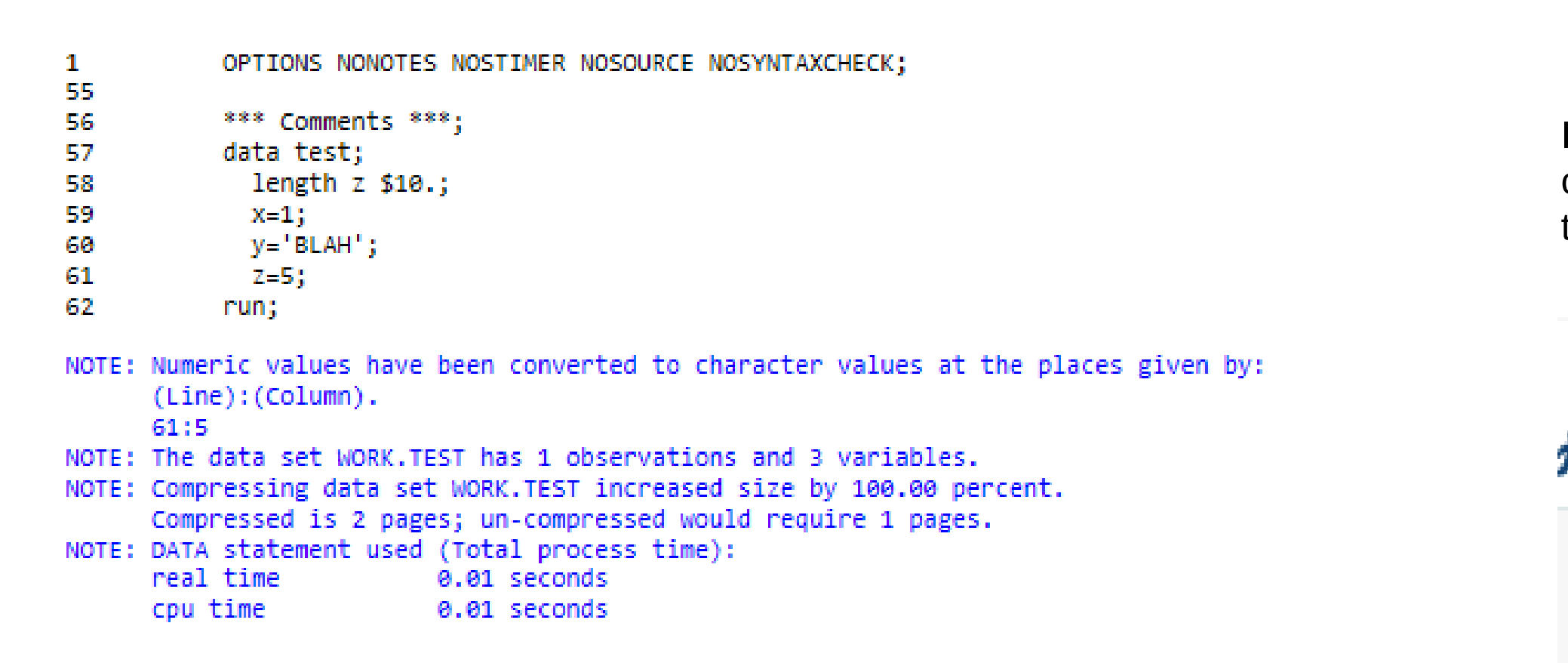
Errors, Warnings, Notes
Errors
Warnings
Notes (4)
NOTE: Numeric values have been converted to character values at the places given by:
NOTE: The data set WORK.TEST has 1 observations and 3 variables.
NOTE: Compressing data set WORK.TEST increased size by 100.00 percent.
NOTE: DATA statement used (Total process time):

The Log tab shows the SAS LOG for the code just executed.

At the top is a summary of all the Errors, Warnings and Notes.

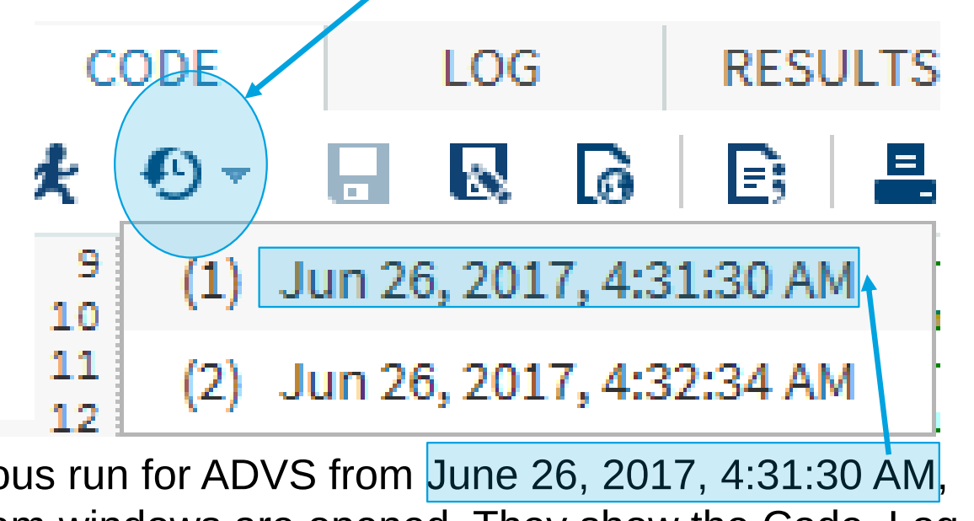
This is quite useful as you can double-click on a note, warning or error and the cursor will be taken directly to that line in the Log tab.

It should be noted that this is replaced every time you execute code from the Code tab.

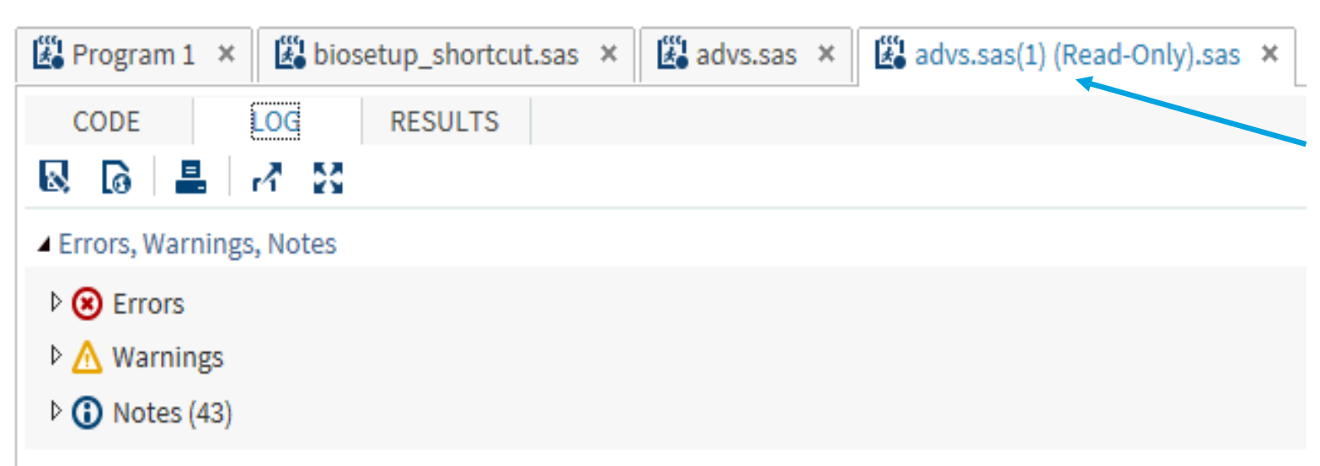


```
1          OPTIONS NONOTES NOSTIMER NOSOURCE NOSYNTAXCHECK;
55
56         *** Comments ***;
57         data test;
58           length z $10.;
59           x=1;
60           y='BLAH';
61           z=5;
62         run;

NOTE: Numeric values have been converted to character values at the places given by:
      (Line):(Column).
      61:5
NOTE: The data set WORK.TEST has 1 observations and 3 variables.
NOTE: Compressing data set WORK.TEST increased size by 100.00 percent.
      Compressed is 2 pages; un-compressed would require 1 pages.
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds
```

If you want to view previous LOGs then you can switch back to the Code tab and click on the Submission history button



(1) Jun 26, 2017, 4:31:30 AM
(2) Jun 26, 2017, 4:32:34 AM



Program 1 | biosetup_shortcut.sas | advs.sas | advs.sas(1) (Read-Only).sas
Errors, Warnings, Notes
Errors
Warnings
Notes (43)

If we open the previous run for ADVS from June 26, 2017, 4:31:30 AM, another set of program windows are opened. They show the Code, Log and Results with a copied filename set to Read-Only (the (1) relates to the (1) in the submission history).

If we'd opened the run from 4:32:34 AM then the tab would have read "advs.sas (2) (Read-Only).sas"). The submission history lasts for as long as your SAS Studio session is in existence.

Results/Output Data tabs

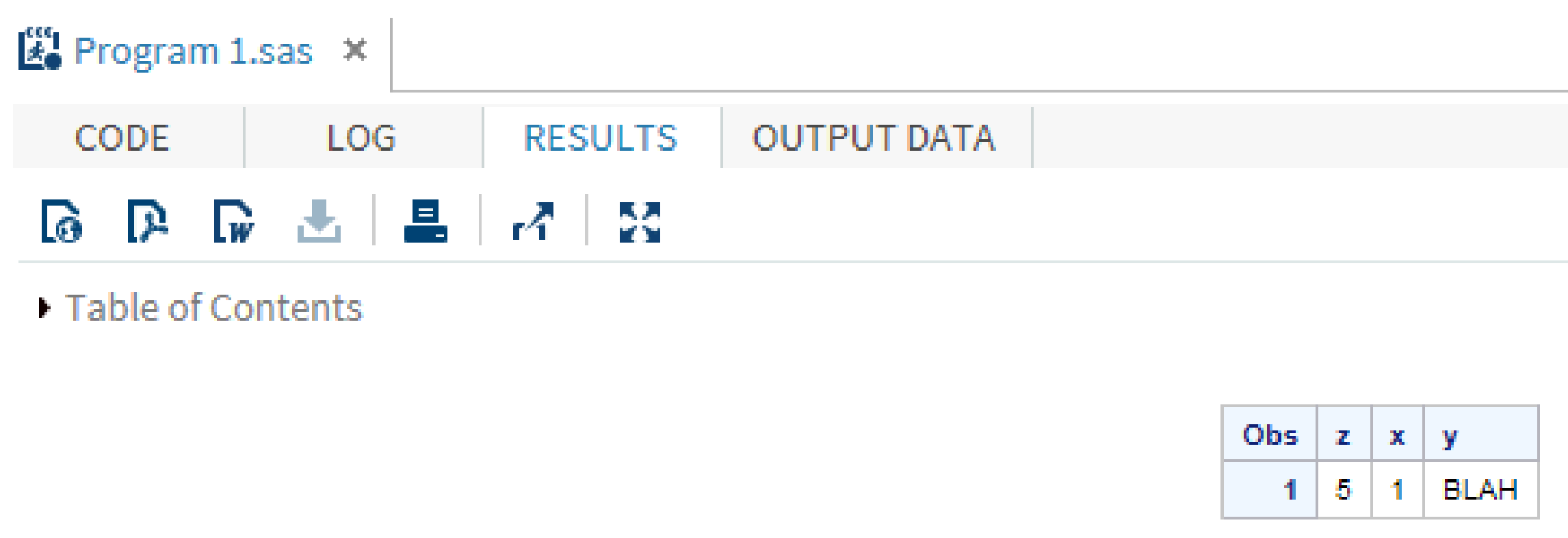
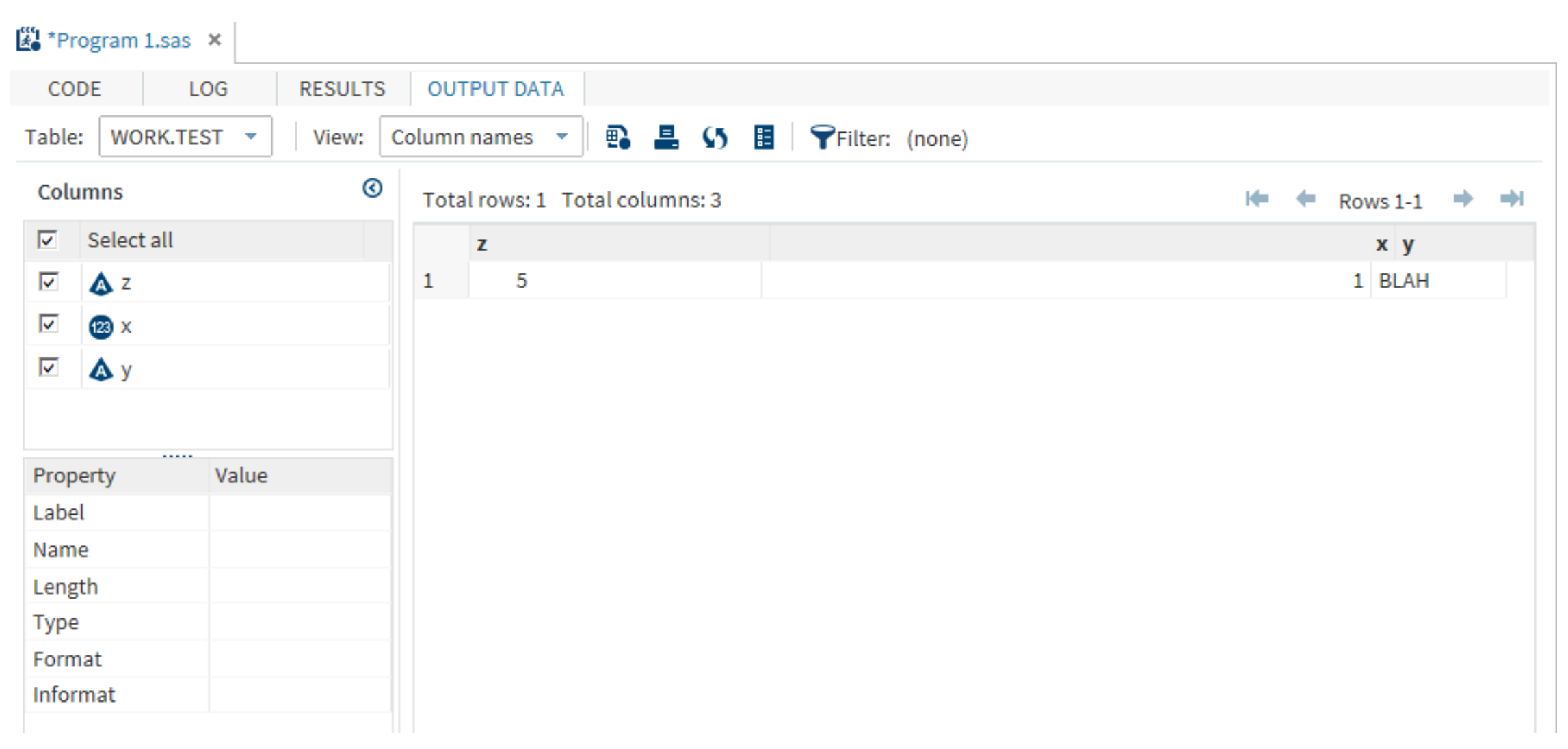


Table of Contents

Obs	z	x	y
1	5	1	BLAH

The default output for SAS v9.4 is ODS HTML which is why the output looks like this rather than an ODS LISTING style.

The "classic" output window is no longer available but when programs are batch-submitted the LST file is an ODS LISTING style output.



Any datasets that are produced during the program execution are available in the Output Data tab for browsing.

The data browser is much more advanced than the viewable from DMS. It is much more like the EG data browser.

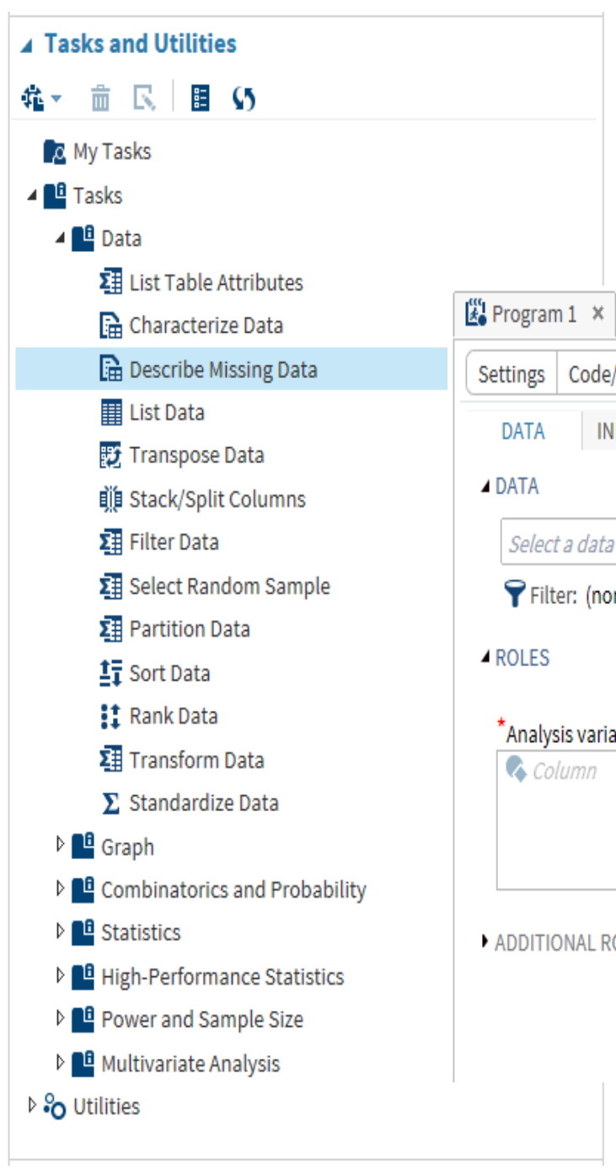
It allows you to see the variable attributes by clicking on the variable in the columns section. You can also select/deselect variables to view.

Filters are able to be added to each variable. The code behind these filters and variable selections can be used (and saved) if required.

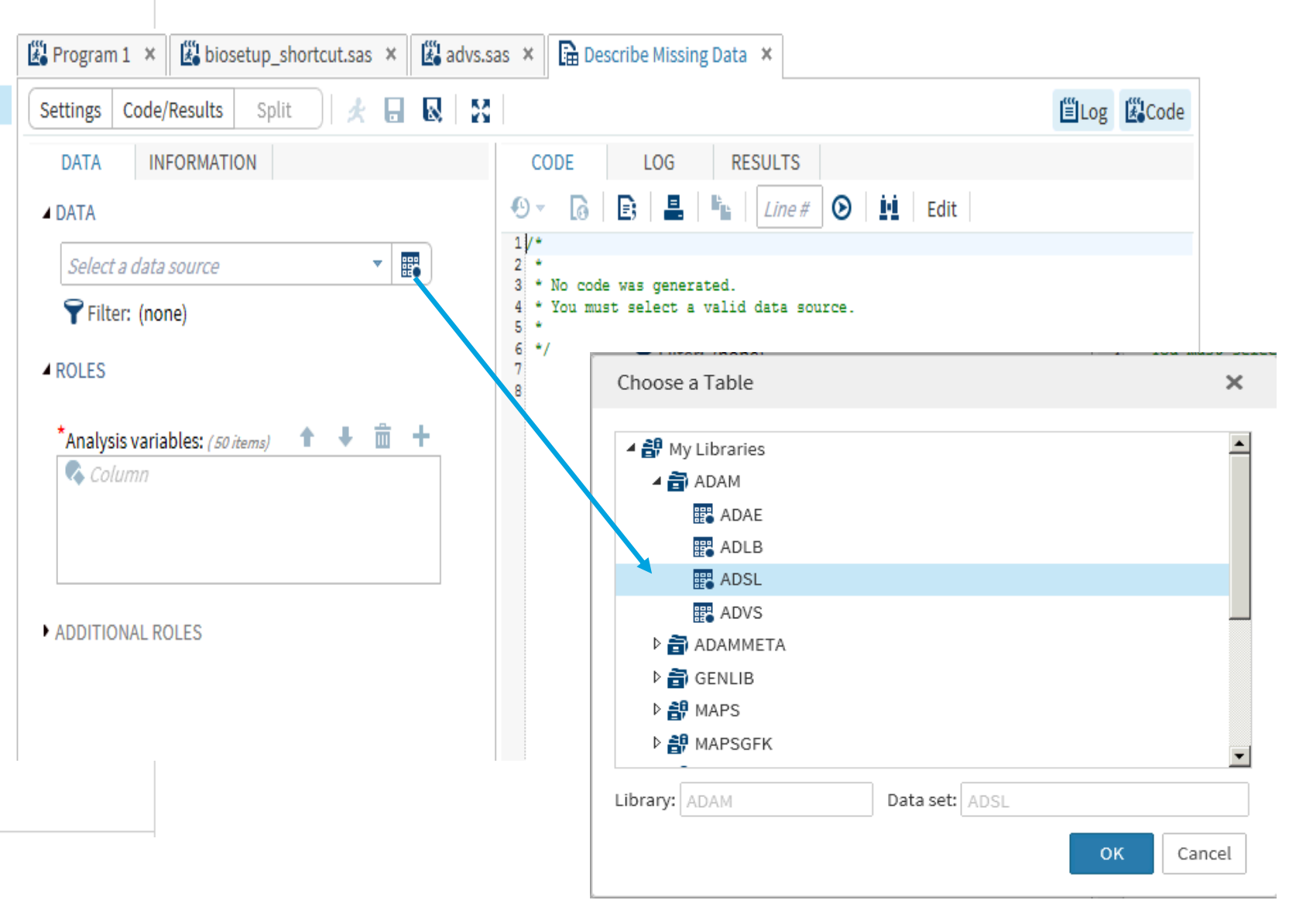
Tasks, Utilities and Snippets

The real difference between DMS and Studio is the addition of Graphical User Interface (GUI) Tasks, Utilities and Snippets. Tasks and Utilities can be used to create SAS code based on the results of a GUI. Snippets are code fragments which are basically the same as Keyboard Macro Files from DMS except that instead of being initiated by a keystroke they are selected using a point-and-click interface. SAS provides a large repository of Tasks, Utilities and Snippets as part of the SAS Studio installation and you can create and save user-defined Tasks and Snippets for personal use as well as creating a globally accessible repository for customised Tasks and Snippets. You'll need to learn some basic XML as well as some Apache Velocity to create your own tasks. An example of a supplied task is shown below:

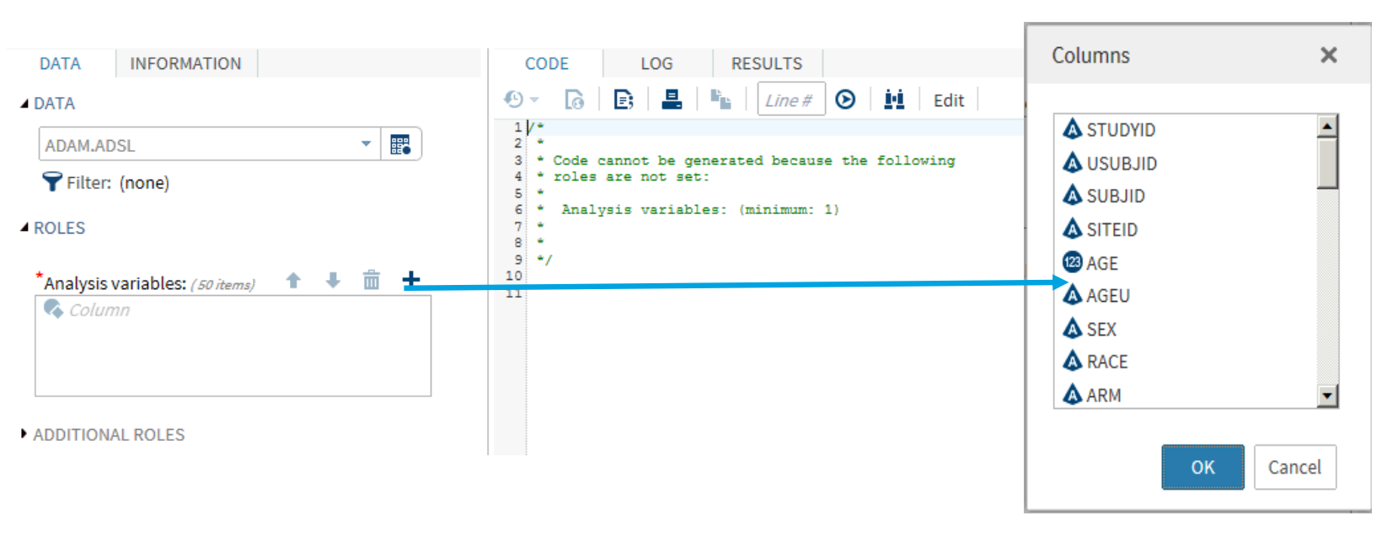
To demonstrate how tasks work we need to open the "Tasks and Utilities" menu. SAS supplies a large amount of tasks as standard. As an example we shall look at the "Describe Missing Data" task. By double-clicking on the task we open a "Describe Missing Data tab which has several interfaces to use to select the data for the required task.



The first thing that is required is a data source. In the "Data" section we open the data browser and select the required dataset (in this case ADAM.ADSL).



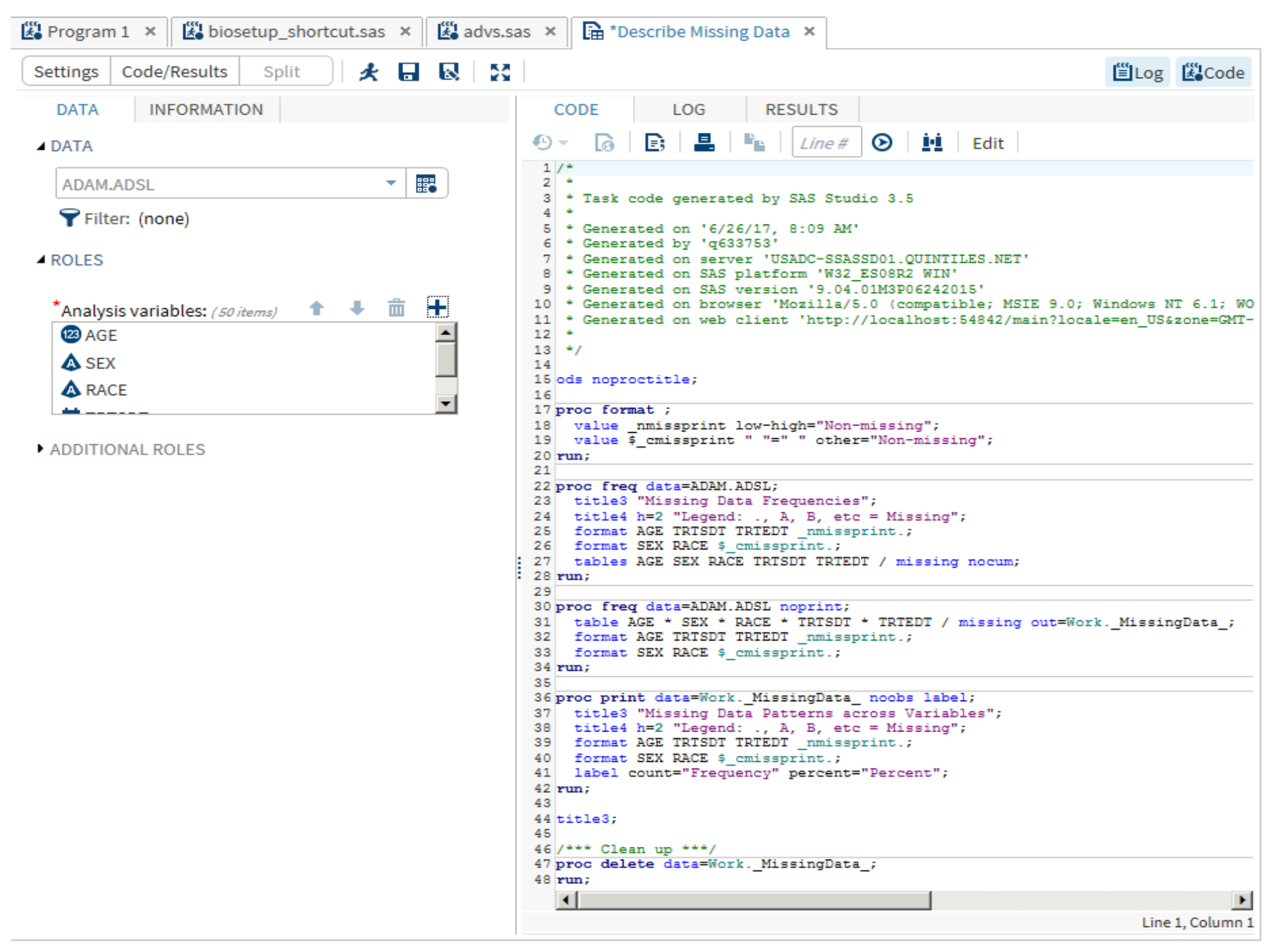
At this point we still don't have any code available as we need to select variables from the input data source to analyse. We select these variables using the Columns selection browser (we can select multiple variables in this task).



Now we have selected our data source and variables to analyse the task has created SAS code to enable us to describe missing data. We've produced a program of 40 lines with a few clicks of the mouse.

We can submit this code using the "Running Man" button as the code produced is "just" a SAS program.

There are more options available to us to customise the reports further (the Additional Roles section).

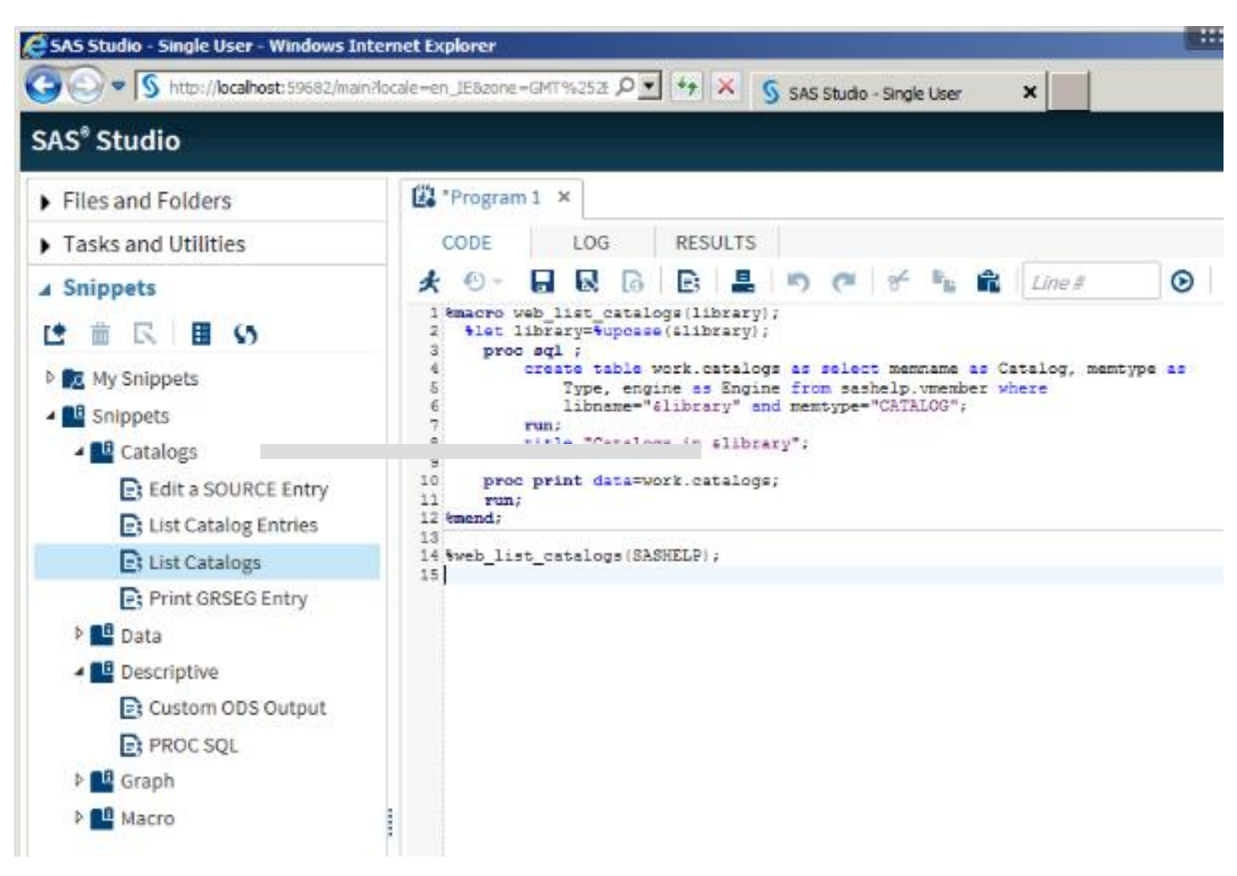


```
/*
 *
 * Task code generated by SAS Studio 3.5
 *
 * Generated on '6/26/17, 8:09 AM'
 * Generated by 'q633753'
 * Generated on server 'USADC-SSASSD01.QUINTILES.NET'
 * Generated on SAS platform 'W32_ES08R2 WIN'
 * Generated on SAS version '9.04.01M3P06242015'
 * Generated on browser 'Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WO
 * Generated on web client 'http://localhost:54842/main?locale=en_US&zone=GMT-
 *
 */

ods noproctitle;

proc format ;
  value _nmissprint low-high="Non-missing";
  value $_cmissprint " "="" " other="Non-missing";
run;

proc freq data=ADAM.ADSL;
  title3 "Missing Data Frequencies";
  title4 h=2 "Legend: ., A, B, etc = Missing";
  format AGE TRTSDT TRTEDT _nmissprint.;
  format SEX RACE $_cmissprint.;
  tables AGE SEX RACE TRTSDT TRTEDT / missing nocum;
run;

proc freq data=ADAM.ADSL noprint;
  table AGE * SEX * RACE * TRTSDT * TRTEDT / missing out=Work._MissingData_;
  format AGE TRTSDT TRTEDT _nmissprint.;
  format SEX RACE $_cmissprint.;
run;

proc print data=Work._MissingData_ noobs label;
  title3 "Missing Data Patterns across Variables";
  title4 h=2 "Legend: ., A, B, etc = Missing";
  format AGE TRTSDT TRTEDT _nmissprint.;
  format SEX RACE $_cmissprint.;
  label count="Frequency" percent="Percent";
run;

title3;

/*** Clean up ***/
proc delete data=Work._MissingData_;
run;
```

Once we execute the code we will produce results like below where we can see a report across the ADAM.ADSL dataset for the 5 selected variables:

The code we have produced can be saved or copied and pasted into an existing program or copied to a new program window and edited.

Snippets allow you to quickly place SAS template code into your program. SAS provides a lot of code snippets within SAS Studio. In the example below shows a SAS-supplied snippet for detailing catalog entries in a libref.



```
%macro web_list_catalogs(library);
 %let library=%upcase(&library);
  proc sql ;
      create table work.catalogs as select memname as Catalog, memtype as
          Type, engine as Engine from sashelp.vmember where
          libname="&library" and memtype="CATALOG";
      run;
      title "Catalogs in &library";

  proc print data=work.catalogs;
  run;
%mend;

%web_list_catalogs(SASHELP);
```

All these programs that are produced by whatever method are still "just" SAS programs. SAS Studio is still a SAS program editor so you can edit and modify the code produced just as you could in the DMS environment. However, unlike the DMS environment code can be produced with a few mouse clicks or you can continue to write SAS code completely from scratch or you can use a combination … one of SAS' great strengths is its versatility and this is now reflected in the programming environments.

SAS Studio 3.6 / User's Guide: http://documentation.sas.com/?cdcId=webeditorcdc&cdcVersion=3.6&docsetId=webeditorug&docsetTarget=titlepage.htm&locale=en

For further details contact the author at lawrence.heaton-wright@quintilesims.com or +44 (0)118 450 8320.