

Paper CC76

Beautiful PROC CONTENTS Output Using the ODS Excel Destination

Suzanne Dorinski¹, U.S. Census Bureau

ABSTRACT

A member of the Census Bureau's in-house SAS® users group asked how to export the output of PROC CONTENTS (variable name, type, length, and format) from several Oracle database tables within the same database to separate worksheets in an Excel file. You can use the `_all_` keyword to work with all the tables within a library. The ODS Excel destination, which is production in SAS 9.4 maintenance release 3, displays the output beautifully.

INTRODUCTION

The variables section of PROC CONTENTS output shows all the variables in a given SAS data set. The Output Delivery System (ODS) allows you to save just the variables section from PROC CONTENTS to a SAS data set. A SAS programmer working with an Oracle database can quickly document all the variables available in an Excel file, with each worksheet representing a table in the Oracle database. This technique also works for a library with native SAS data sets.

The ODS Excel destination, which is production in SAS 9.4 maintenance release 3, easily handles multiple worksheets in an Excel file, and produces beautiful output with just two options specified in the program. The ODS Excel destination is pre-production in SAS 9.4 maintenance releases 1 and 2. We used SAS 9.4 maintenance release 2 to write the code in this paper.

THE FULL CODE FOR THIS EXAMPLE

Below is the full code for this example.

```
libname asj oracle
  user=asjview
  password=asjview
  path=jailprod.world
  schema=asjdba
  access=readonly;
run;

ods output variables=AllOracleVarOut;

proc contents data=asj._all_;
run;

proc sort data=AllOracleVarOut;
  by member num;
run;

options nobyline;

ODS EXCEL FILE="H:\SESUG 2015\all_variables_from_Oracle_database.xlsx"
  options(sheet_name="#BYVAL(member)"
    embedded_titles='yes');

proc print data=AllOracleVarOut noobs;
  by member;
  pageby member;
  title "Variables in #BYVAL(member) table";
run;

ODS EXCEL CLOSE;
```

¹ Disclaimer: Any views expressed are those of the author and not necessarily those of the U.S. Census Bureau.

CONNECT TO THE ORACLE DATABASE

We use a LIBNAME statement to connect to an Oracle database. See below for an example LIBNAME statement. Contact your Oracle database administrator for the user, password, path, schema, and access parameters.

```
libname asj oracle
  user=asjview
  password=asjview
  path=jailprod.world
  schema=asjdba
  access=readonly;
run;
```

In this example, the Oracle database has eight tables. See Figure 1.

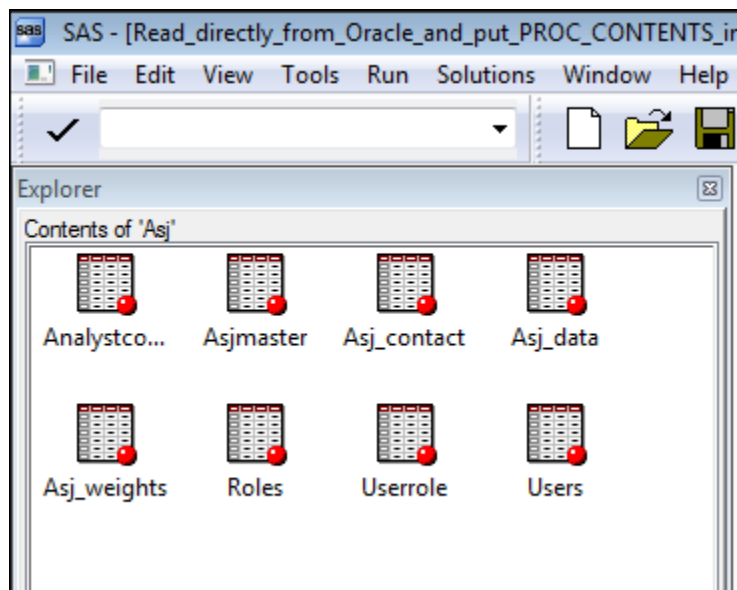


Figure 1. Explorer window shows eight tables in Oracle database

USE ODS OUTPUT STATEMENT TO SAVE INFORMATION TO A DATA SET

The ODS output statement saves procedure output to a SAS data set. PROC CONTENTS produces three tables of output for a data set: attributes, engine/host information, and variables. In this example, we want the variables, but not the attributes nor the engine/host information. In the code below, the information from the variables section of PROC CONTENTS will be stored in a temporary data set named AllOracleVarOut.

```
ods output variables=AllOracleVarOut;
```

If you do not know the name of the procedure output object, use the ODS TRACE statement. The ODS TRACE statement will write notes in the log, listing the names of the objects produced by SAS procedures. Figure 2 shows the notes in the log from the TRACE statement.

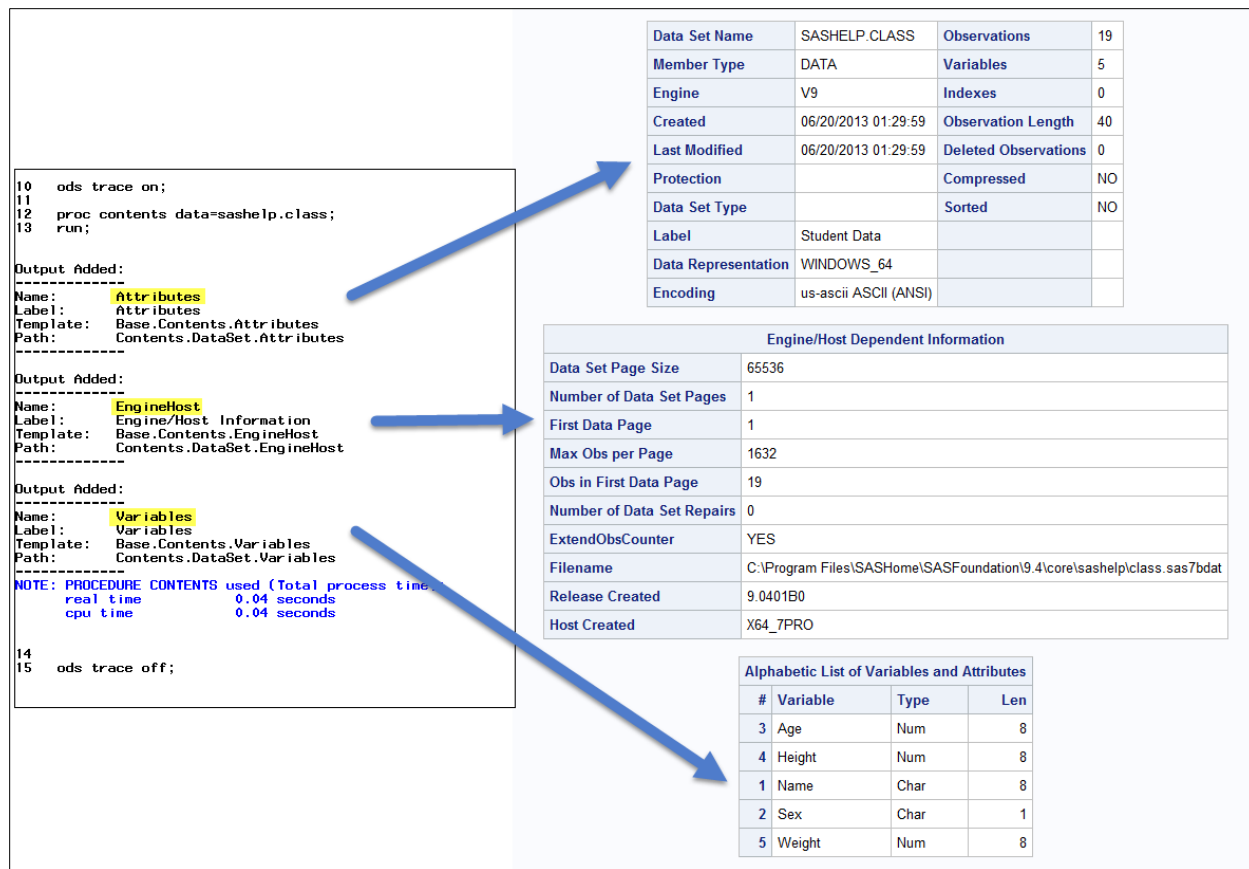


Figure 2. TRACE statement in log file shows that PROC CONTENTS produces three output objects

USE THE _ALL_ KEYWORD TO WORK WITH ALL THE DATA SETS IN A LIBRARY

Instead of running PROC CONTENTS separately for each data set, we can use the _ALL_ keyword to run PROC CONTENTS once and get all the variable information for all the data sets in the library.

```

proc contents data=sasj._all_;
run;

```

SORT BY DATA SET AND VARIABLE POSITION

In the PROC SORT, we sort by MEMBER and NUM so that the worksheets in the Excel file will be in alphabetical order by data set name, and the variables in each data set are in order by position.

```

proc sort data=AllOracleVarOut;
  by member num;
run;

```

PRODUCE THE OUTPUT

In the code below, we use the NOBYLINE option to suppress the default text above each worksheet, which is MEMBER=<data set>. We prefer a customized title on each worksheet to make the output a bit more reader-friendly.

The ODS EXCEL statement opens the Excel destination. The Excel destination uses the same style (HTMLBlue) that the Results Viewer uses by default. The SHEET_NAME option names the worksheet. Setting SHEET_NAME to #BYVAL(member) tells SAS to use the current value of MEMBER as the worksheet name. The EMBEDDED_TITLES option set to yes makes the title show up in the body of the worksheet.

We use the PAGE and PAGEBY statements in PROC PRINT to show each database on a separate worksheet. Using #BYVAL(member) in the TITLE statement tells SAS to use the current value of MEMBER in the title.

```

options nobyline;

ODS EXCEL FILE="H:\SESUG 2015\all_variables_from_Oracle_database.xlsx"
  options(sheet_name="#BYVAL(member)"
    embedded_titles='yes');

proc print data=AllOracleVarOut noobs;
  by member;
  pageby member;
  title "Variables in #BYVAL(member) table";
run;

ODS EXCEL CLOSE;

```

VIEW THE OUTPUT

Figure 3 shows the output for this example. The worksheet name in the Excel file is the same as the data set it describes. The title above each table of variable information also shows the name of the data set.

Num	Variable	Type	Len	Pos	Format	Informat	Label
1	ID	Char	21	0	\$21.	\$21.	ID
2	CONFPERSONS	Char	6	24	\$6.	\$6.	CONFPERSONS
3	CONFPERSONSFLAG	Num	8	32	2	2	CONFPERSONSFLAG
4	NONCONFPERSONS	Char	6	40	\$6.	\$6.	NONCONFPERSONS
5	NONCONFPERSONSFLAG	Num	8	48	2	2	NONCONFPERSONSFLAG
6	TOTAL1C	Char	6	56	\$6.	\$6.	TOTAL1C
7	TOTAL1CFLAG	Num	8	64	2	2	TOTAL1CFLAG
8	NONCITIZEN	Char	6	72	\$6.	\$6.	NONCITIZEN
9	NONCITIZENFLAG	Num	8	80	2	2	NONCITIZENFLAG
10	WEEKENDERPROG	Num	8	88	2	2	WEEKENDERPROG
11	WEEKENDER	Char	6	96	\$6.	\$6.	WEEKENDER
12	WEEKENDERFLAG	Num	8	104	2	2	WEEKENDERFLAG
13	ADULTMALES	Char	6	112	\$6.	\$6.	ADULTMALES
14	ADULTMALESFLAG	Num	8	120	2	2	ADULTMALESFLAG
15	ADULTFEMALES	Char	6	128	\$6.	\$6.	ADULTFEMALES
16	ADULTFEMALESFLAG	Num	8	136	2	2	ADULTFEMALESFLAG
17	JUVMALES	Char	6	144	\$6.	\$6.	JUVMALES

Figure 3. Variable information from PROC CONTENTS displayed in ODS Excel destination

USE THE SAME TECHNIQUE WITH NATIVE SAS DATA SETS

The motivation for this paper was to document tables in an Oracle database, but the same technique also works for a library with native SAS data sets.

The code example on the next page works for the SASHELP library. We used the MEMTYPE option on the PROC CONTENTS statement to process data sets, but not views.

```
ODS OUTPUT variables=allvarout;

proc contents data=sashelp._all_ memtype=data;
run;

proc sort data=allvarout;
  by member num;
run;

options nobyline;

ODS EXCEL FILE="H:\SESUG 2015\all_variables_in_SASHELP_data_sets.xlsx"
  options(sheet_name="#BYVAL(member)"
    embedded_titles='yes');

proc print data=allvarout noobs;
  by member;
  pageby member;
  title "Variables in #BYVAL(member) data set";
run;

ODS EXCEL CLOSE;
```

SASHELP RESULTS MAY VARY

Note that the number of data sets available in the SASHELP library depends on the products installed. There are 185 data sets in the SASHELP library in the Census Bureau's virtual desktop infrastructure installation. There are 86 data sets in the SASHELP library in SAS University Edition.

SAS STUDIO MAY ISSUE WARNING ABOUT INSUFFICIENT MEMORY

If you are using SAS Studio, you may see a message about insufficient memory. Figure 4 on the next page shows the warning when using the SAS Studio interface in SAS University Edition. If you choose "Don't Display", you will not see anything in the Results tab, but SAS Studio does create the Excel file.

To avoid that warning in SAS Studio, close the HTML5 destination at the top of the program with the following line of code:

```
ODS HTML5(id=web) CLOSE;
```

SAS Studio will create the Excel file, but nothing will show in the Results tab if the HTML5 destination is closed.

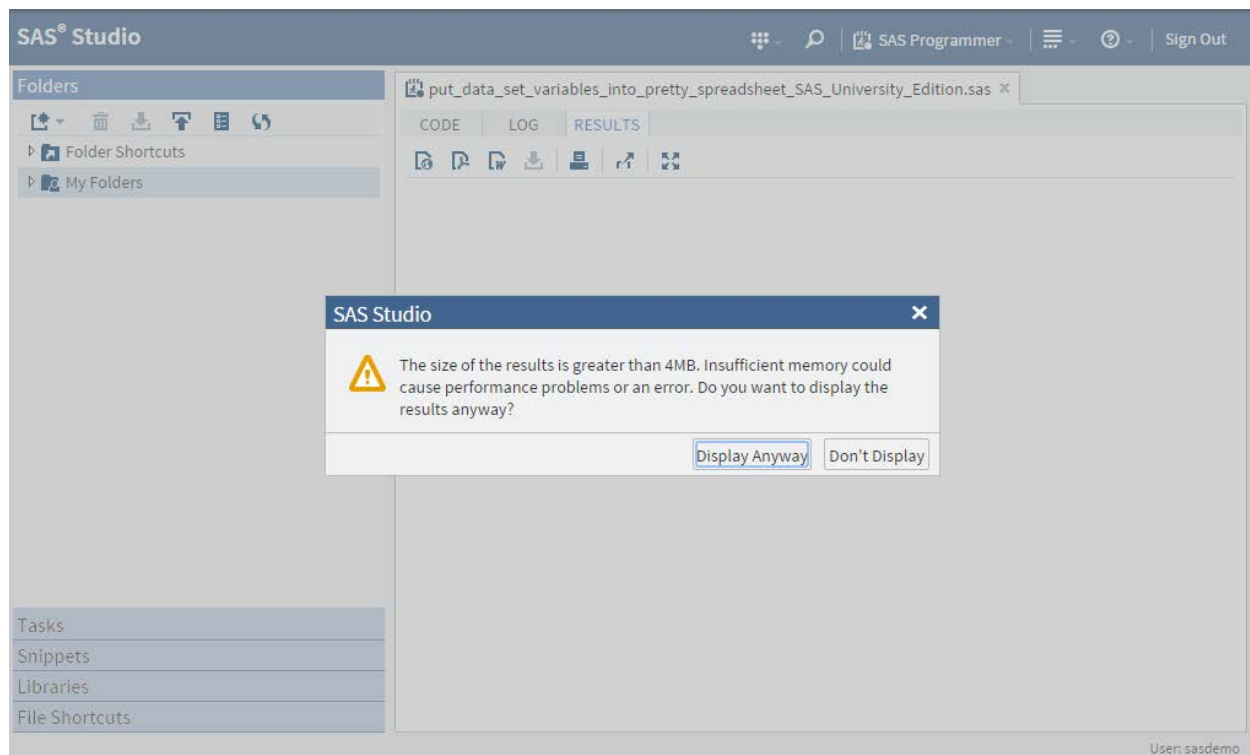


Figure 4. SAS Studio warning about insufficient memory

Figure 5 shows what the output looks like for the data sets in the SASHELP library.

The screenshot shows a Microsoft Excel spreadsheet titled "all_variables_in_SASHELP_data_sets.xlsx". The spreadsheet displays the output of PROC CONTENTS for the SASHELP.BASEBALL data set. The data is organized into columns: Num, Variable, Type, Len, Pos, Format, Label, and Informat. The rows list various baseball statistics and their corresponding data types and lengths.

Num	Variable	Type	Len	Pos	Format	Label	Informat
1	Name	Char	18	144		Player's Name	
2	Team	Char	14	162		Team at the End of 1986	
3	nAtBat	Num	8	0		Times at Bat in 1986	
4	nHits	Num	8	8		Hits in 1986	
5	nHome	Num	8	16		Home Runs in 1986	
6	nRuns	Num	8	24		Runs in 1986	
7	nRBI	Num	8	32		RBIs in 1986	
8	nBB	Num	8	40		Walks in 1986	
9	YrMajor	Num	8	48		Years in the Major Leagues	
10	CrAtBat	Num	8	56		Career Times at Bat	
11	CrHits	Num	8	64		Career Hits	
12	CrHome	Num	8	72		Career Home Runs	
13	CrRuns	Num	8	80		Career Runs	
14	CrRbi	Num	8	88		Career RBIs	
15	CrBB	Num	8	96		Career Walks	
16	League	Char	8	176		League at the End of 1986	
17	Division	Char	8	184		Division at the End of 1986	

Figure 5. Variables in SASHELP data sets

CONCLUSION

The ODS Excel destination produces beautiful output with minimal coding. We can use the technique in this paper to document quickly all the variables available in a SAS library, whether that library is a connection to an Oracle database or a collection of native SAS data sets.

REFERENCES

SAS Institute Inc. 2014. *Base SAS® 9.4 Procedures Guide, Third Edition*. Cary, NC: SAS Institute Inc.
See *Example 1: Using PROC CONTENTS to Extract Only Attributes from Data Sets* in the CONTENTS Procedure section, which shows how to get the attributes information for all the data sets in a SAS library.

The SAS code and Excel files are available at the sasCommunity wiki page for this paper,
[http://www.sascommunity.org/wiki/Beautiful PROC CONTENTS Output Using the ODS Excel Destination](http://www.sascommunity.org/wiki/Beautiful_PROC_CONTENTS_Output_Using_the_ODS_Excel_Destination).

ACKNOWLEDGMENTS

Thanks to Ellen Cohen for asking the question that prompted this paper. Thanks to Terri Craig, Franklin Winters, and Freda Spence for reviewing the draft of this paper. Thanks to Scott Huntley, Nancy Goodling, Amy Peters, and Michael Monaco, all of SAS Institute, for answering questions in the Quad at SAS Global Forum 2015.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name:	Suzanne Dorinski
Organization:	U.S. Census Bureau
Address:	ESMD HQ-6K062E 4600 Silver Hill Road
City, State ZIP:	Washington, DC 20233
Work Phone:	301-763-4869
Email:	Suzanne.Marie.Dorinski@census.gov
sascommunity.org:	http://www.sascommunity.org/wiki/Presentations:Dorinski Papers and Presentations

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.