SESUG Paper 151-2019

HOW AM I DIFFERENT?

Dr. Kannan Deivasigamani, Wellcare – TSS

Abstract

This SESUG paper demonstrates a practical application of how Dictionary tables in SAS® can be used to make a programmer's task easier. While DICTIONARY is rich in meta-data, the example presented will just be one of the many possible ways COLUMNS table can be used to spot and present differences in column names between 2 different datasets easily. This paper is intended for audiences who wish to find the differences between 2 SAS tables/datasets as an alternative to PROC COMPARE to tabulate similarities and commonalities at a variable level. This is commonly used when there is new development or analysis, migration or reengineering of processes take place within organizations.

INTRODUCTION

Programmers often seek to use the procedure from SAS with the hope of finding differences between 2 datasets. One step further, the interest focuses on the variables that are different between the two datasets in question. Fortunately, "PROC COMPARE" provides information about the number of variables that are common between the 2 datasets and also prints the number of variables in the BASE dataset but not in the COMPARE dataset and vice versa. However, there is some inconvenience posed when it comes to listing out which variables are different and show them side by side in a table format. In many occasions, it turns out to be a manual process of eyeballing the fields or transferring the metadata to an Excel spreadsheet and using a VLOOKUP or some other function and then compiling the variables. As an alternative, the tool presented in this paper in the form of a macro can take two SAS dataset names as input and it provides a print of the common columns, differences in the 2 datasets by listing out columns in

BASE but not in COMPARE dataset and vice versa. It would save several minutes of one's time without scrolling through much or rerunning the PROC COMPARE code with LISTVAR, LISTBASEVAR, or LISTCOMPVAR options to get similar information. If this is done on multiple occasions, by multiple programmers, the benefit multiplies. The goal is to provide to make the process convenient just by calling the macro with the 2 dataset names intended for tabulation showing the differences and similarities.

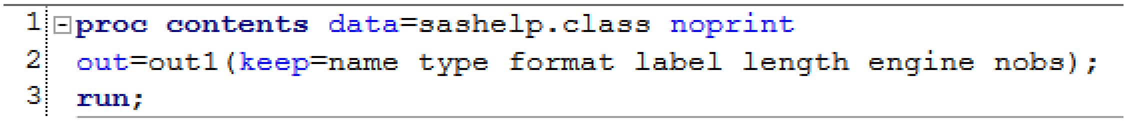
BUSINESS REQUIREMENT

During development of new datasets that are similar but have variations there may often arise a need to find the differences between the two datasets. The differences that we are talking about are not at the data-level. The structural differences, to be more specific, are the ones of interest in this context. There may also be testing effort between two different environments such as "Production" and "QA" where there may be a need to identify the differences. Another situation that can very likely require such an exercise is where there may be a conversion of one system to another that may be better or even an enhancement requiring a comparison. No matter what the situation is, the tool presented in this paper will come in handy.

PRACTICAL EXAMPLE

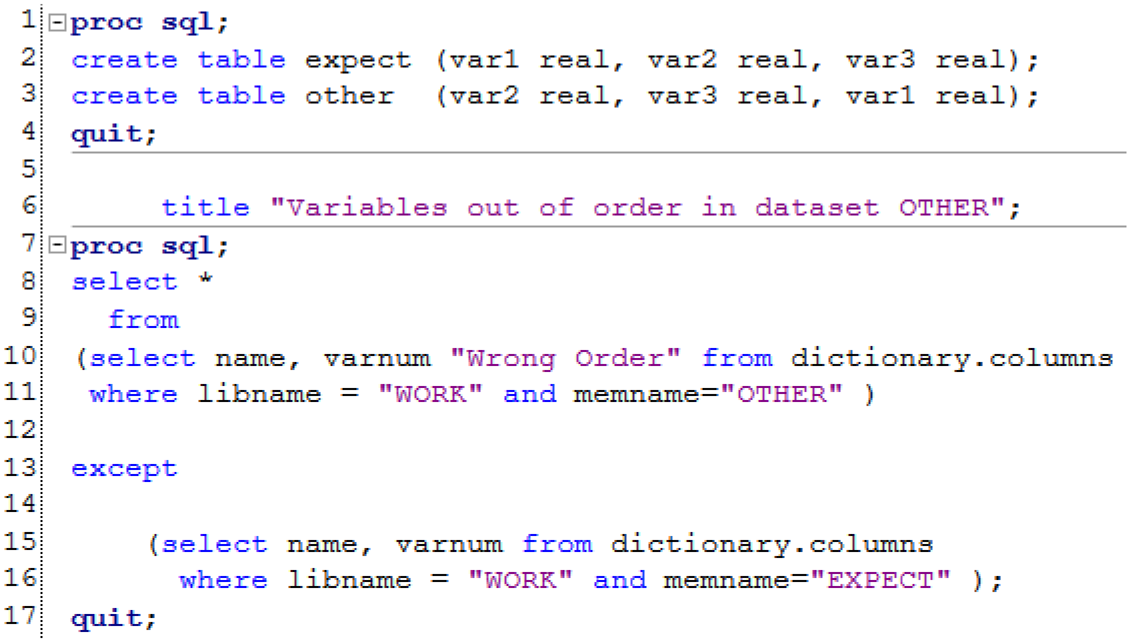
The tool will accept the names of the two datasets to compare and will provide an output that will print common variables, variables in dataset-1 but not in dataset-2, and variables in dataset-2 but not in dataset-1 in 3 columns side by side. This can be presented to the individual who can address or justify the differences. The tool is a SAS macro that will accept two inputs. There are 2 examples presented here to demonstrate that the tool can be used with normal SAS datasets as well as SAS generation datasets.

Bennett (2001) and Lee (2012) adopted an alternate way of capturing meta-data with use of the procedure commonly used by SAS programmers, "PROC CONTENTS".



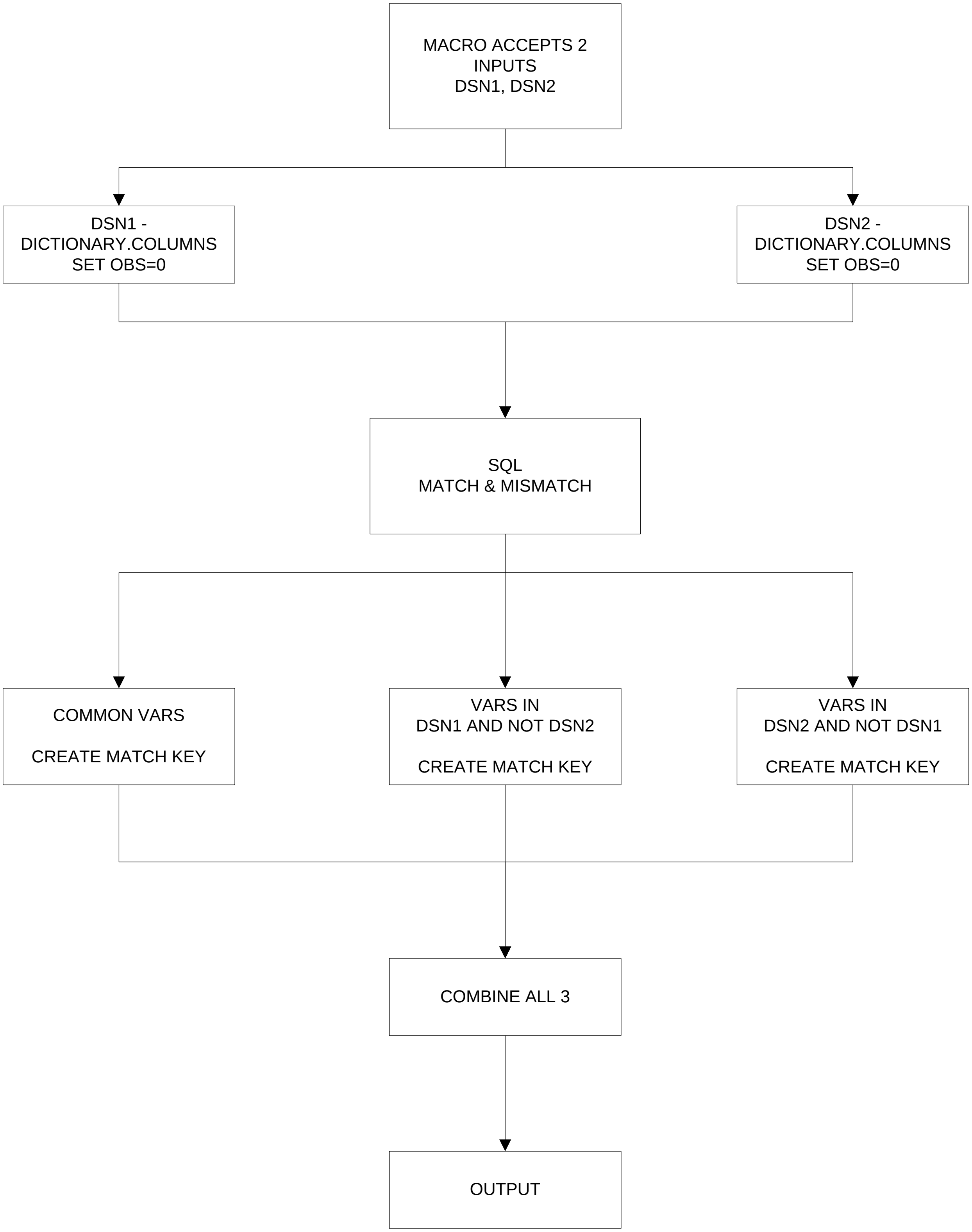
```
proc contents data=sashelp.class noprint
out=out1(keep=name type format label length engine nobs);
run;
```

If one needs to flag the order of the variables in a SAS dataset can take advantage of PG Stats' (Esteemed Advisor) code in the user forum using the Dictionary meta-data shown below:

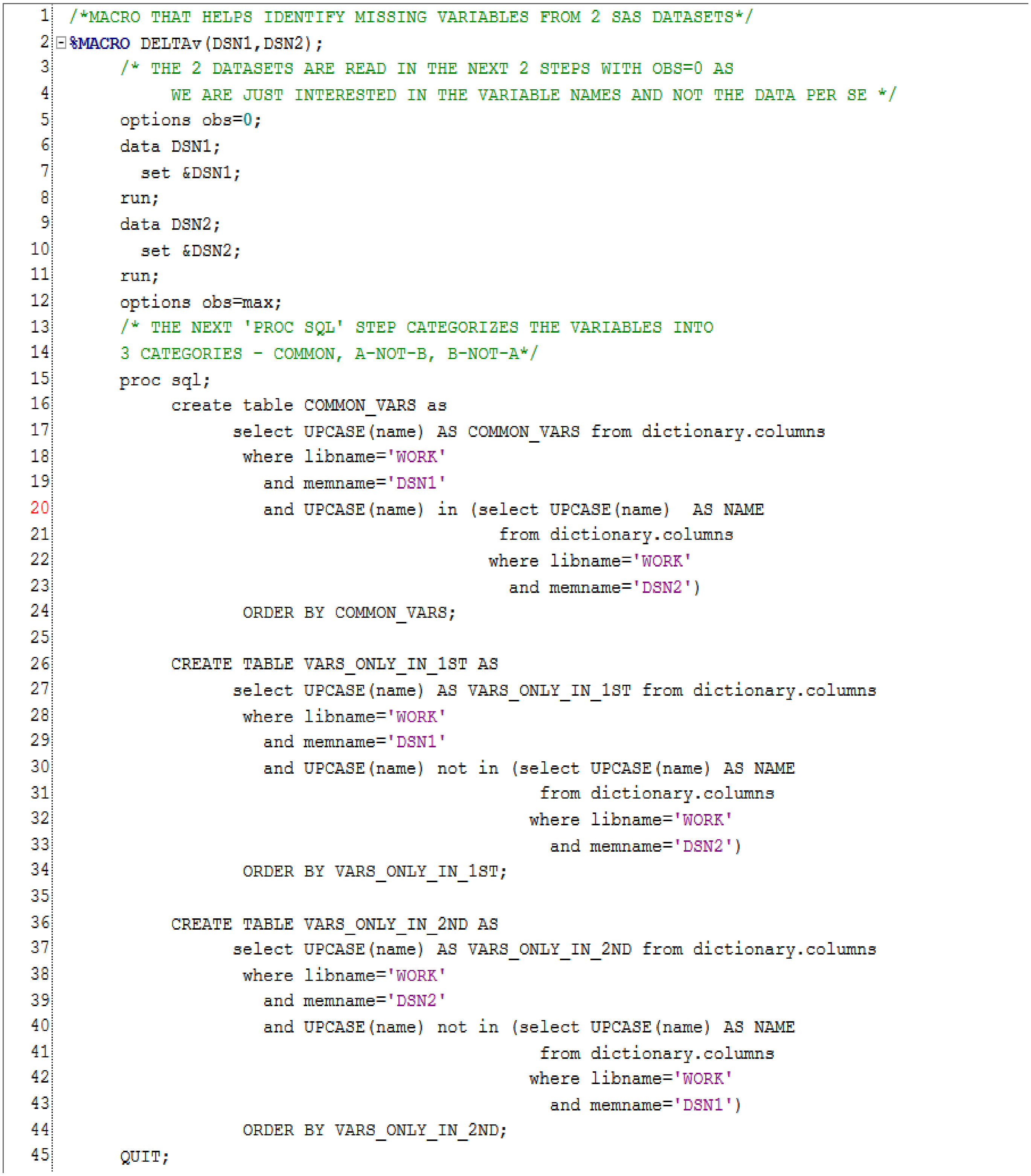


```
proc sql;
create table expect (var1 real, var2 real, var3 real);
create table other  (var2 real, var3 real, var1 real);
quit;

     title "Variables out of order in dataset OTHER";
proc sql;
select *
  from
 (select name, varnum "Wrong Order" from dictionary.columns
  where libname = "WORK" and memname="OTHER" )

except

    (select name, varnum from dictionary.columns
      where libname = "WORK" and memname="EXPECT" );
quit;
```

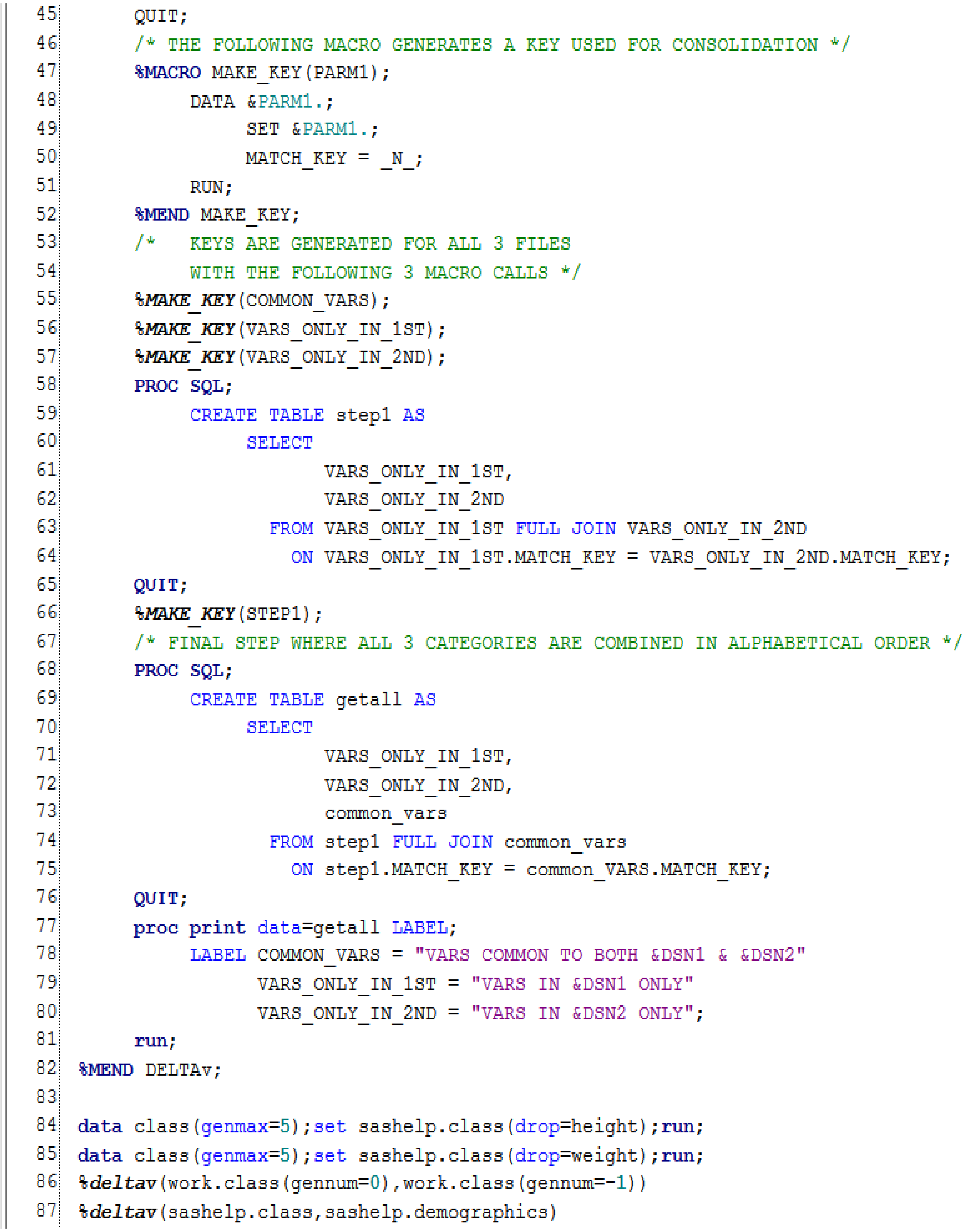
A PICTORIAL REPRESENTATION OF THE PROCESS


MACRO ACCEPTS 2 INPUTS DSN1, DSN2
DSN1 - DICTIONARY.COLUMNS SET OBS=0
DSN2 - DICTIONARY.COLUMNS SET OBS=0
SQL MATCH & MISMATCH
COMMON VARS CREATE MATCH KEY
VARS IN DSN1 AND NOT DSN2 CREATE MATCH KEY
VARS IN DSN2 AND NOT DSN1 CREATE MATCH KEY
COMBINE ALL 3
OUTPUT

CODE

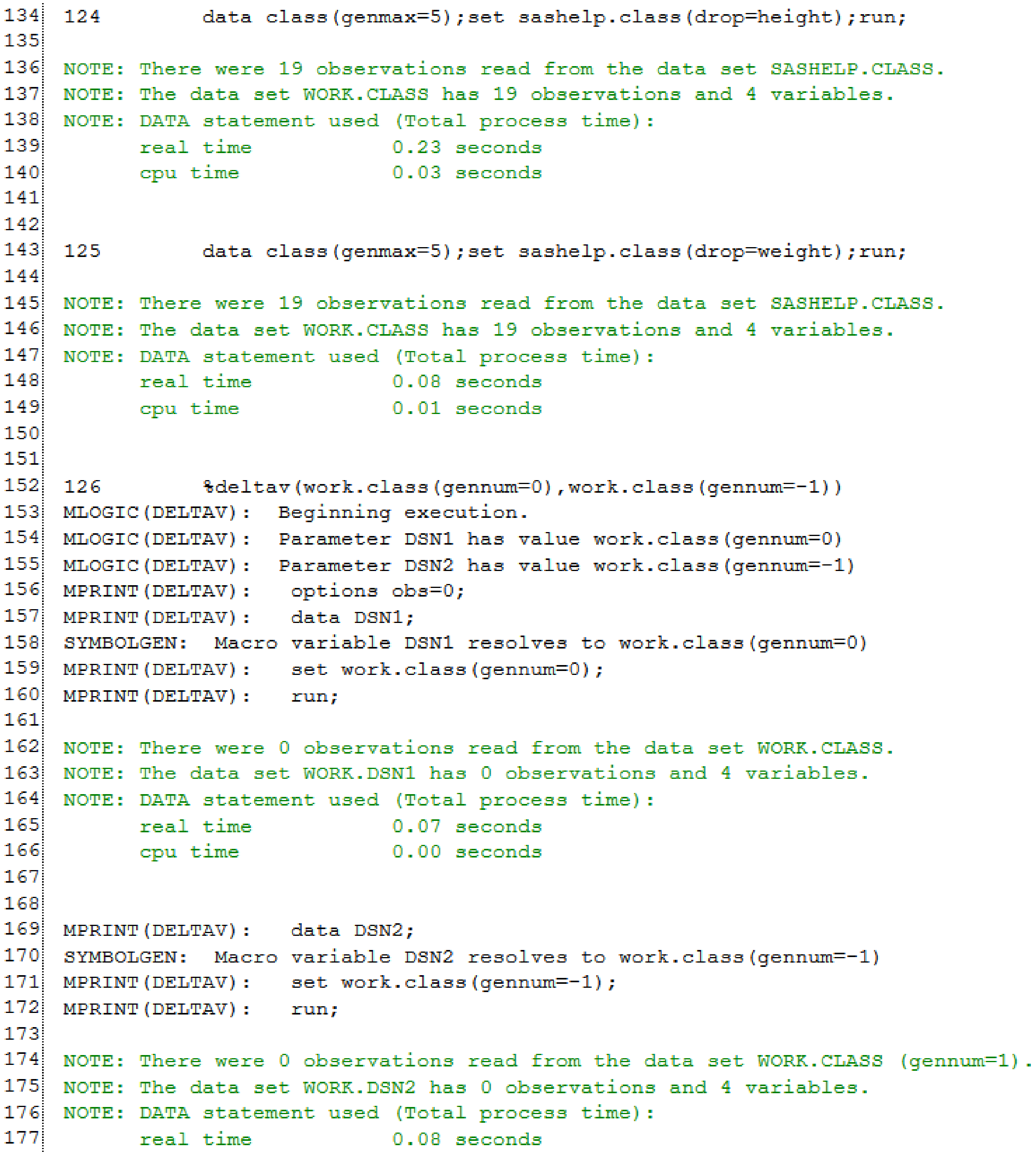


```
/*MACRO THAT HELPS IDENTIFY MISSING VARIABLES FROM 2 SAS DATASETS*/
%MACRO DELTAv(DSN1,DSN2);
     /* THE 2 DATASETS ARE READ IN THE NEXT 2 STEPS WITH OBS=0 AS
          WE ARE JUST INTERESTED IN THE VARIABLE NAMES AND NOT THE DATA PER SE */
     options obs=0;
     data DSN1;
       set &DSN1;
     run;
     data DSN2;
       set &DSN2;
     run;
     options obs=max;
     /* THE NEXT 'PROC SQL' STEP CATEGORIZES THE VARIABLES INTO
     3 CATEGORIES - COMMON, A-NOT-B, B-NOT-A*/
     proc sql;
          create table COMMON_VARS as
                select UPCASE(name) AS COMMON_VARS from dictionary.columns
                 where libname='WORK'
                   and memname='DSN1'
                   and UPCASE(name) in (select UPCASE(name)  AS NAME
                                          from dictionary.columns
                                         where libname='WORK'
                                           and memname='DSN2')
                 ORDER BY COMMON_VARS;

          CREATE TABLE VARS_ONLY_IN_1ST AS
                select UPCASE(name) AS VARS_ONLY_IN_1ST from dictionary.columns
                 where libname='WORK'
                   and memname='DSN1'
                   and UPCASE(name) not in (select UPCASE(name) AS NAME
                                              from dictionary.columns
                                             where libname='WORK'
                                               and memname='DSN2')
                 ORDER BY VARS_ONLY_IN_1ST;

          CREATE TABLE VARS_ONLY_IN_2ND AS
                select UPCASE(name) AS VARS_ONLY_IN_2ND from dictionary.columns
                 where libname='WORK'
                   and memname='DSN2'
                   and UPCASE(name) not in (select UPCASE(name) AS NAME
                                              from dictionary.columns
                                             where libname='WORK'
                                               and memname='DSN1')
                 ORDER BY VARS_ONLY_IN_2ND;
     QUIT;
```

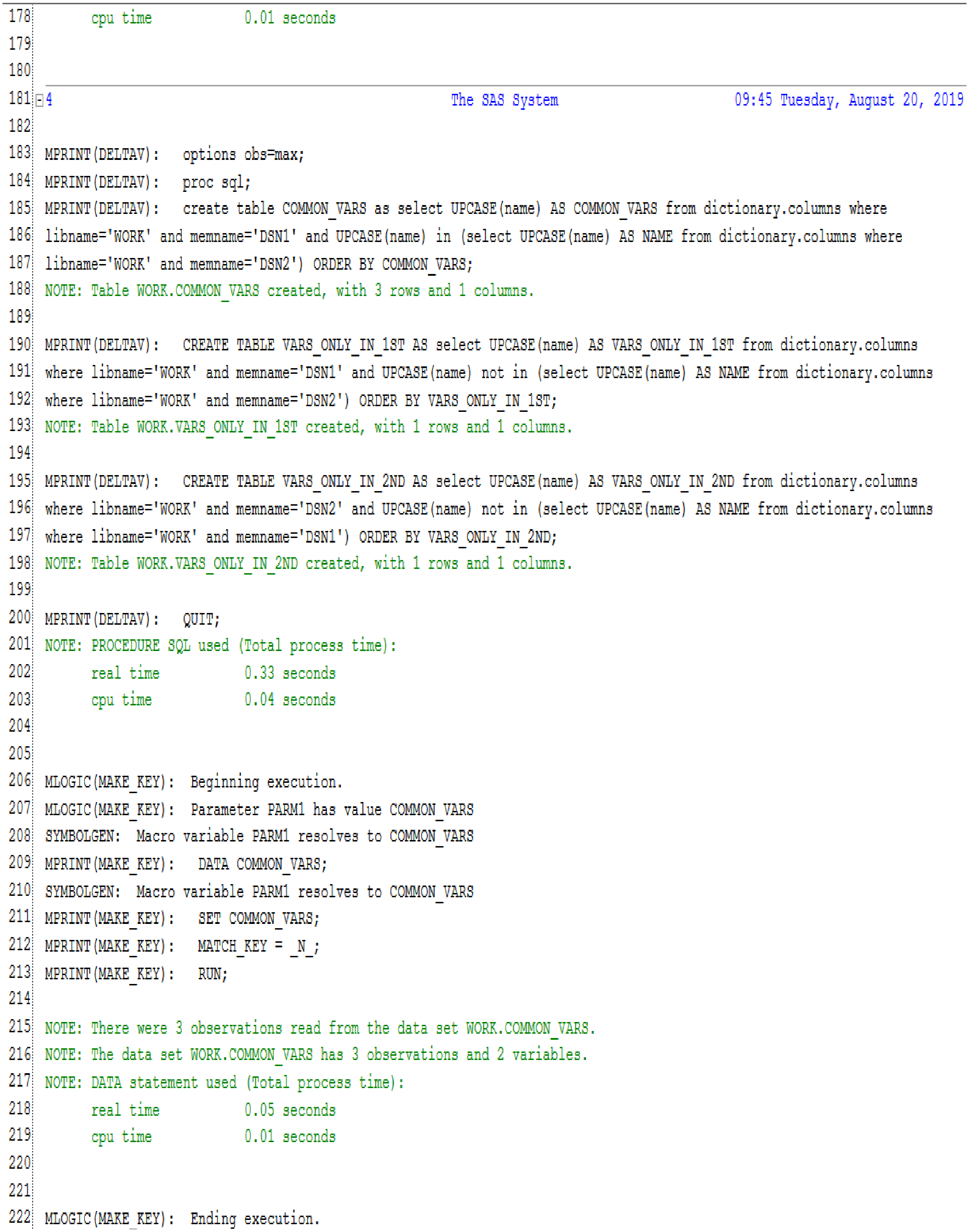


```
     QUIT;
     /* THE FOLLOWING MACRO GENERATES A KEY USED FOR CONSOLIDATION */
     %MACRO MAKE_KEY(PARM1);
          DATA &PARM1.;
               SET &PARM1.;
               MATCH_KEY = _N_;
          RUN;
     %MEND MAKE_KEY;
     /*   KEYS ARE GENERATED FOR ALL 3 FILES
          WITH THE FOLLOWING 3 MACRO CALLS */
     %MAKE_KEY(COMMON_VARS);
     %MAKE_KEY(VARS_ONLY_IN_1ST);
     %MAKE_KEY(VARS_ONLY_IN_2ND);
     PROC SQL;
          CREATE TABLE step1 AS
               SELECT
                      VARS_ONLY_IN_1ST,
                      VARS_ONLY_IN_2ND
                 FROM VARS_ONLY_IN_1ST FULL JOIN VARS_ONLY_IN_2ND
                   ON VARS_ONLY_IN_1ST.MATCH_KEY = VARS_ONLY_IN_2ND.MATCH_KEY;
     QUIT;
     %MAKE_KEY(STEP1);
     /* FINAL STEP WHERE ALL 3 CATEGORIES ARE COMBINED IN ALPHABETICAL ORDER */
     PROC SQL;
          CREATE TABLE getall AS
               SELECT
                      VARS_ONLY_IN_1ST,
                      VARS_ONLY_IN_2ND,
                      common_vars
                 FROM step1 FULL JOIN common_vars
                   ON step1.MATCH_KEY = common_VARS.MATCH_KEY;
     QUIT;
     proc print data=getall LABEL;
          LABEL COMMON_VARS = "VARS COMMON TO BOTH &DSN1 & &DSN2"
                VARS_ONLY_IN_1ST = "VARS IN &DSN1 ONLY"
                VARS_ONLY_IN_2ND = "VARS IN &DSN2 ONLY";
     run;
%MEND DELTAv;

data class(genmax=5);set sashelp.class(drop=height);run;
data class(genmax=5);set sashelp.class(drop=weight);run;
%deltav(work.class(gennum=0),work.class(gennum=-1))
%deltav(sashelp.class,sashelp.demographics)
```

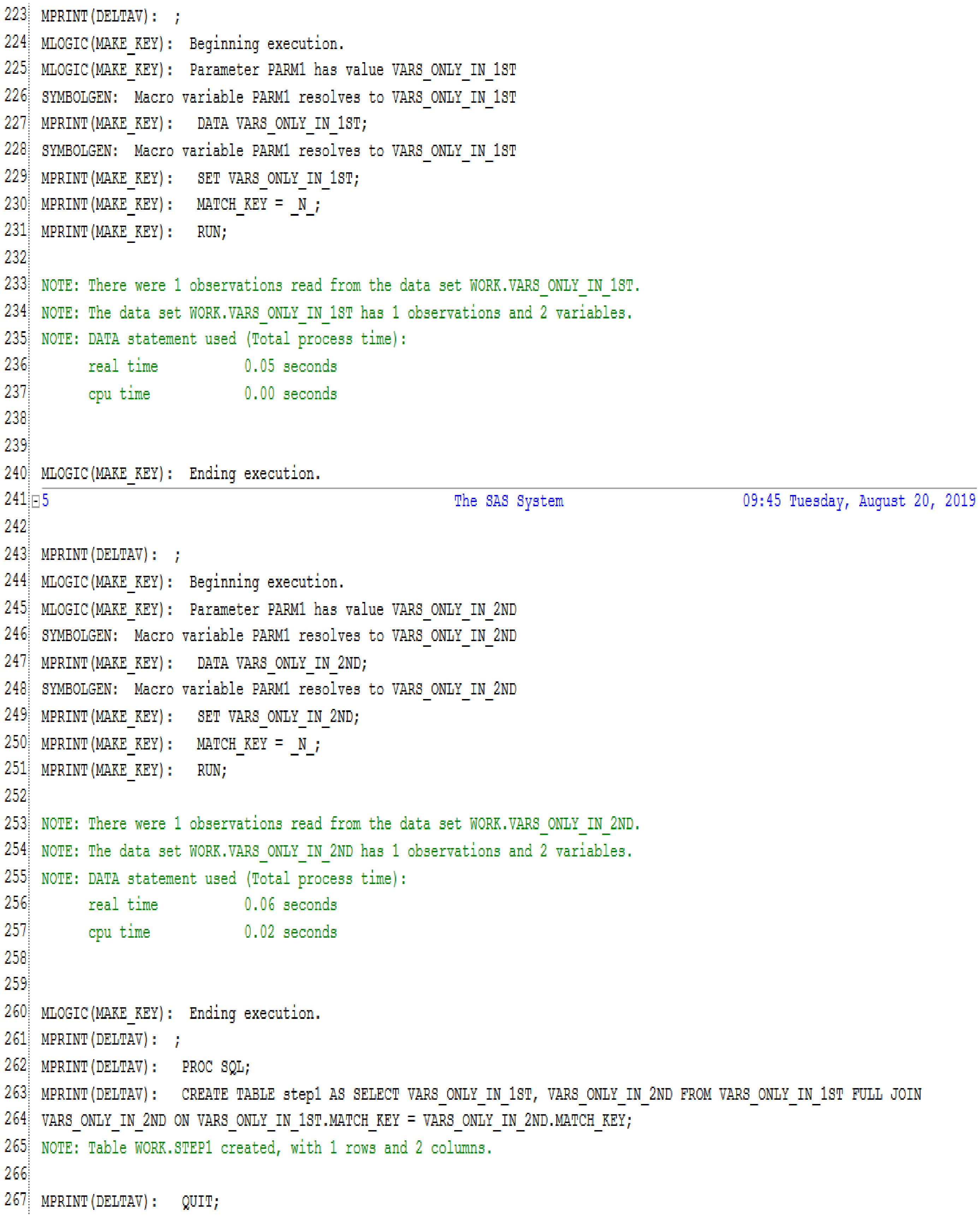
LOG



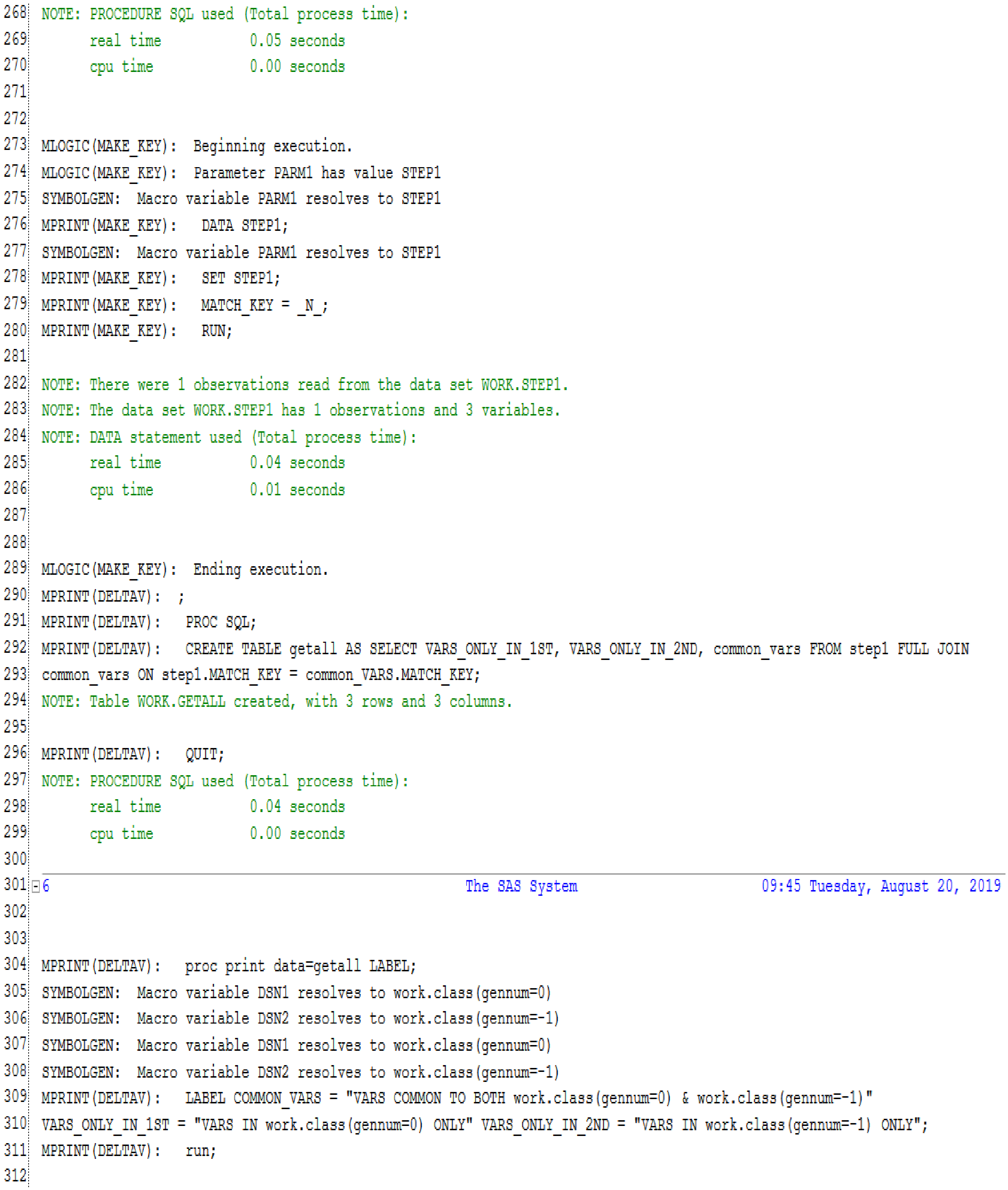
```
124        data class(genmax=5);set sashelp.class(drop=height);run;

NOTE: There were 19 observations read from the data set SASHELP.CLASS.
NOTE: The data set WORK.CLASS has 19 observations and 4 variables.
NOTE: DATA statement used (Total process time):
      real time           0.23 seconds
      cpu time            0.03 seconds


125        data class(genmax=5);set sashelp.class(drop=weight);run;

NOTE: There were 19 observations read from the data set SASHELP.CLASS.
NOTE: The data set WORK.CLASS has 19 observations and 4 variables.
NOTE: DATA statement used (Total process time):
      real time           0.08 seconds
      cpu time            0.01 seconds


126        %deltav(work.class(gennum=0),work.class(gennum=-1))
MLOGIC(DELTAV):  Beginning execution.
MLOGIC(DELTAV):  Parameter DSN1 has value work.class(gennum=0)
MLOGIC(DELTAV):  Parameter DSN2 has value work.class(gennum=-1)
MPRINT(DELTAV):   options obs=0;
MPRINT(DELTAV):   data DSN1;
SYMBOLGEN:  Macro variable DSN1 resolves to work.class(gennum=0)
MPRINT(DELTAV):   set work.class(gennum=0);
MPRINT(DELTAV):   run;

NOTE: There were 0 observations read from the data set WORK.CLASS.
NOTE: The data set WORK.DSN1 has 0 observations and 4 variables.
NOTE: DATA statement used (Total process time):
      real time           0.07 seconds
      cpu time            0.00 seconds


MPRINT(DELTAV):   data DSN2;
SYMBOLGEN:  Macro variable DSN2 resolves to work.class(gennum=-1)
MPRINT(DELTAV):   set work.class(gennum=-1);
MPRINT(DELTAV):   run;

NOTE: There were 0 observations read from the data set WORK.CLASS (gennum=1).
NOTE: The data set WORK.DSN2 has 0 observations and 4 variables.
NOTE: DATA statement used (Total process time):
      real time           0.08 seconds
```



```
      cpu time            0.01 seconds


4                                         The SAS System                    09:45 Tuesday, August 20, 2019

MPRINT(DELTAV):   options obs=max;
MPRINT(DELTAV):   proc sql;
MPRINT(DELTAV):   create table COMMON_VARS as select UPCASE(name) AS COMMON_VARS from dictionary.columns where
libname='WORK' and memname='DSN1' and UPCASE(name) in (select UPCASE(name) AS NAME from dictionary.columns where
libname='WORK' and memname='DSN2') ORDER BY COMMON_VARS;
NOTE: Table WORK.COMMON_VARS created, with 3 rows and 1 columns.

MPRINT(DELTAV):   CREATE TABLE VARS_ONLY_IN_1ST AS select UPCASE(name) AS VARS_ONLY_IN_1ST from dictionary.columns
where libname='WORK' and memname='DSN1' and UPCASE(name) not in (select UPCASE(name) AS NAME from dictionary.columns
where libname='WORK' and memname='DSN2') ORDER BY VARS_ONLY_IN_1ST;
NOTE: Table WORK.VARS_ONLY_IN_1ST created, with 1 rows and 1 columns.

MPRINT(DELTAV):   CREATE TABLE VARS_ONLY_IN_2ND AS select UPCASE(name) AS VARS_ONLY_IN_2ND from dictionary.columns
where libname='WORK' and memname='DSN2' and UPCASE(name) not in (select UPCASE(name) AS NAME from dictionary.columns
where libname='WORK' and memname='DSN1') ORDER BY VARS_ONLY_IN_2ND;
NOTE: Table WORK.VARS_ONLY_IN_2ND created, with 1 rows and 1 columns.

MPRINT(DELTAV):   QUIT;
NOTE: PROCEDURE SQL used (Total process time):
      real time           0.33 seconds
      cpu time            0.04 seconds


MLOGIC(MAKE_KEY):  Beginning execution.
MLOGIC(MAKE_KEY):  Parameter PARM1 has value COMMON_VARS
SYMBOLGEN:  Macro variable PARM1 resolves to COMMON_VARS
MPRINT(MAKE_KEY):   DATA COMMON_VARS;
SYMBOLGEN:  Macro variable PARM1 resolves to COMMON_VARS
MPRINT(MAKE_KEY):   SET COMMON_VARS;
MPRINT(MAKE_KEY):   MATCH_KEY = _N_;
MPRINT(MAKE_KEY):   RUN;

NOTE: There were 3 observations read from the data set WORK.COMMON_VARS.
NOTE: The data set WORK.COMMON_VARS has 3 observations and 2 variables.
NOTE: DATA statement used (Total process time):
      real time           0.05 seconds
      cpu time            0.01 seconds


MLOGIC(MAKE_KEY):  Ending execution.
```

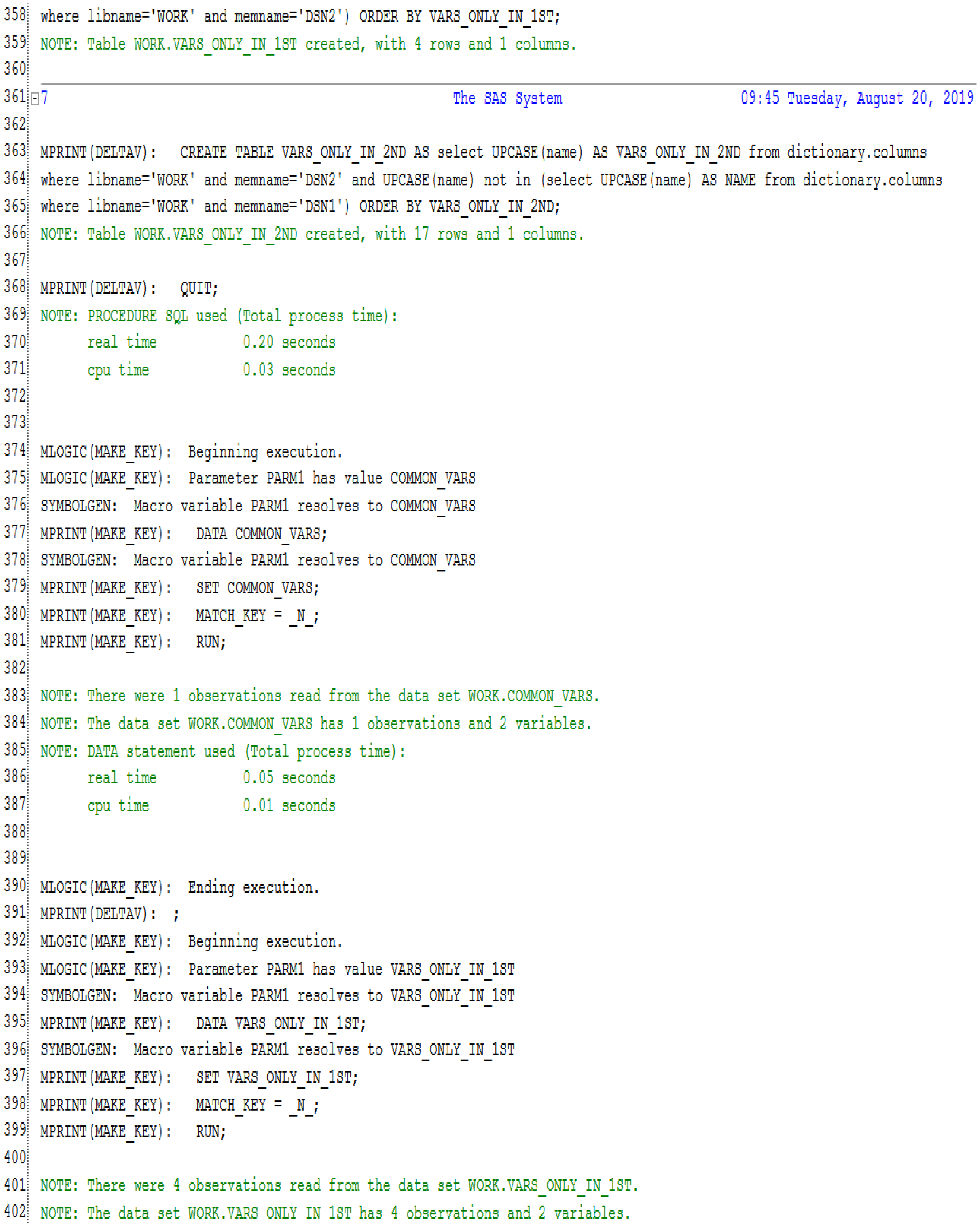


```
MPRINT(DELTAV):  ;
MLOGIC(MAKE_KEY):  Beginning execution.
MLOGIC(MAKE_KEY):  Parameter PARM1 has value VARS_ONLY_IN_1ST
SYMBOLGEN:  Macro variable PARM1 resolves to VARS_ONLY_IN_1ST
MPRINT(MAKE_KEY):   DATA VARS_ONLY_IN_1ST;
SYMBOLGEN:  Macro variable PARM1 resolves to VARS_ONLY_IN_1ST
MPRINT(MAKE_KEY):   SET VARS_ONLY_IN_1ST;
MPRINT(MAKE_KEY):   MATCH_KEY = _N_;
MPRINT(MAKE_KEY):   RUN;

NOTE: There were 1 observations read from the data set WORK.VARS_ONLY_IN_1ST.
NOTE: The data set WORK.VARS_ONLY_IN_1ST has 1 observations and 2 variables.
NOTE: DATA statement used (Total process time):
      real time           0.05 seconds
      cpu time            0.00 seconds


MLOGIC(MAKE_KEY):  Ending execution.
5                                                          The SAS System                              09:45 Tuesday, August 20, 2019

MPRINT(DELTAV):  ;
MLOGIC(MAKE_KEY):  Beginning execution.
MLOGIC(MAKE_KEY):  Parameter PARM1 has value VARS_ONLY_IN_2ND
SYMBOLGEN:  Macro variable PARM1 resolves to VARS_ONLY_IN_2ND
MPRINT(MAKE_KEY):   DATA VARS_ONLY_IN_2ND;
SYMBOLGEN:  Macro variable PARM1 resolves to VARS_ONLY_IN_2ND
MPRINT(MAKE_KEY):   SET VARS_ONLY_IN_2ND;
MPRINT(MAKE_KEY):   MATCH_KEY = _N_;
MPRINT(MAKE_KEY):   RUN;

NOTE: There were 1 observations read from the data set WORK.VARS_ONLY_IN_2ND.
NOTE: The data set WORK.VARS_ONLY_IN_2ND has 1 observations and 2 variables.
NOTE: DATA statement used (Total process time):
      real time           0.06 seconds
      cpu time            0.02 seconds


MLOGIC(MAKE_KEY):  Ending execution.
MPRINT(DELTAV):  ;
MPRINT(DELTAV):   PROC SQL;
MPRINT(DELTAV):   CREATE TABLE step1 AS SELECT VARS_ONLY_IN_1ST, VARS_ONLY_IN_2ND FROM VARS_ONLY_IN_1ST FULL JOIN
VARS_ONLY_IN_2ND ON VARS_ONLY_IN_1ST.MATCH_KEY = VARS_ONLY_IN_2ND.MATCH_KEY;
NOTE: Table WORK.STEP1 created, with 1 rows and 2 columns.

MPRINT(DELTAV):   QUIT;
```

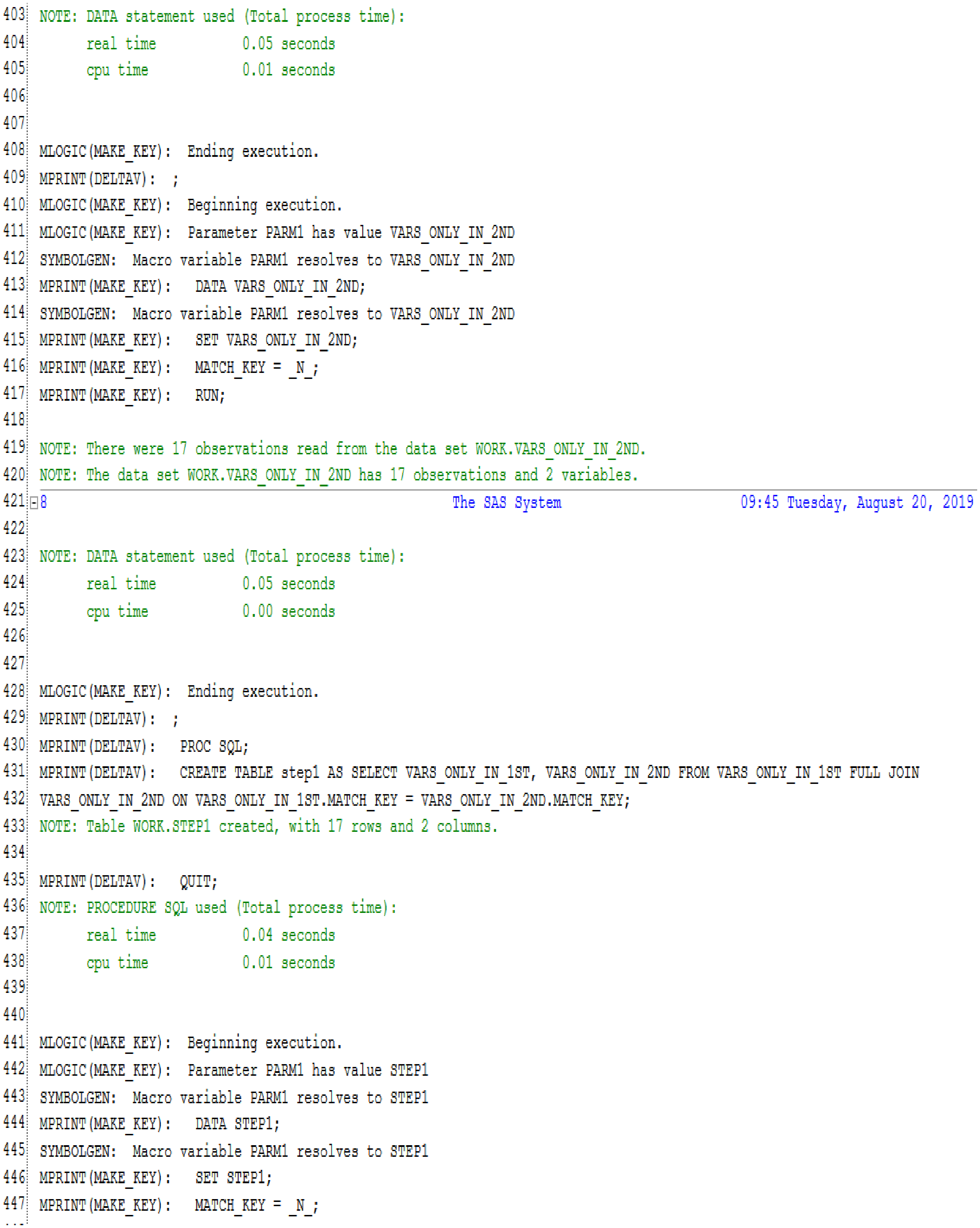


```
NOTE: PROCEDURE SQL used (Total process time):
      real time           0.05 seconds
      cpu time            0.00 seconds


MLOGIC(MAKE_KEY):  Beginning execution.
MLOGIC(MAKE_KEY):  Parameter PARM1 has value STEP1
SYMBOLGEN:  Macro variable PARM1 resolves to STEP1
MPRINT(MAKE_KEY):   DATA STEP1;
SYMBOLGEN:  Macro variable PARM1 resolves to STEP1
MPRINT(MAKE_KEY):   SET STEP1;
MPRINT(MAKE_KEY):   MATCH_KEY = _N_;
MPRINT(MAKE_KEY):   RUN;

NOTE: There were 1 observations read from the data set WORK.STEP1.
NOTE: The data set WORK.STEP1 has 1 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time           0.04 seconds
      cpu time            0.01 seconds


MLOGIC(MAKE_KEY):  Ending execution.
MPRINT(DELTAV):  ;
MPRINT(DELTAV):   PROC SQL;
MPRINT(DELTAV):   CREATE TABLE getall AS SELECT VARS_ONLY_IN_1ST, VARS_ONLY_IN_2ND, common_vars FROM step1 FULL JOIN
common_vars ON step1.MATCH_KEY = common_VARS.MATCH_KEY;
NOTE: Table WORK.GETALL created, with 3 rows and 3 columns.

MPRINT(DELTAV):   QUIT;
NOTE: PROCEDURE SQL used (Total process time):
      real time           0.04 seconds
      cpu time            0.00 seconds

6                                                    The SAS System                       09:45 Tuesday, August 20, 2019


MPRINT(DELTAV):   proc print data=getall LABEL;
SYMBOLGEN:  Macro variable DSN1 resolves to work.class(gennum=0)
SYMBOLGEN:  Macro variable DSN2 resolves to work.class(gennum=-1)
SYMBOLGEN:  Macro variable DSN1 resolves to work.class(gennum=0)
SYMBOLGEN:  Macro variable DSN2 resolves to work.class(gennum=-1)
MPRINT(DELTAV):   LABEL COMMON_VARS = "VARS COMMON TO BOTH work.class(gennum=0) & work.class(gennum=-1)"
VARS_ONLY_IN_1ST = "VARS IN work.class(gennum=0) ONLY" VARS_ONLY_IN_2ND = "VARS IN work.class(gennum=-1) ONLY";
MPRINT(DELTAV):   run;

```

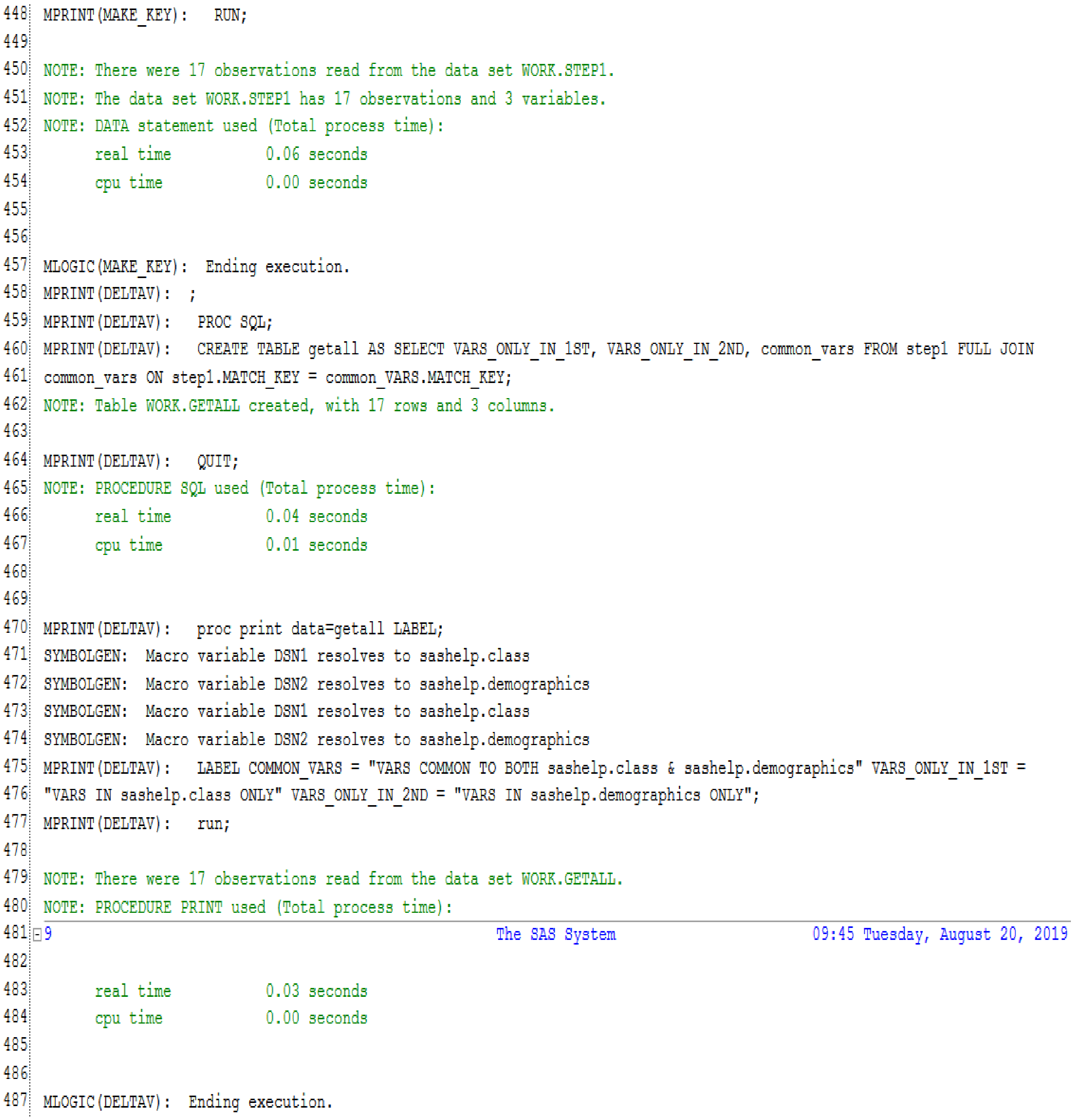
```
NOTE: There were 3 observations read from the data set WORK.GETALL.
NOTE: PROCEDURE PRINT used (Total process time):
      real time           0.04 seconds
      cpu time            0.00 seconds


MLOGIC(DELTAV):  Ending execution.
127         %deltav(sashelp.class,sashelp.demographics)
MLOGIC(DELTAV):  Beginning execution.
MLOGIC(DELTAV):  Parameter DSN1 has value sashelp.class
MLOGIC(DELTAV):  Parameter DSN2 has value sashelp.demographics
MPRINT(DELTAV):   options obs=0;
MPRINT(DELTAV):   data DSN1;
SYMBOLGEN:  Macro variable DSN1 resolves to sashelp.class
MPRINT(DELTAV):   set sashelp.class;
MPRINT(DELTAV):   run;

NOTE: There were 0 observations read from the data set SASHELP.CLASS.
NOTE: The data set WORK.DSN1 has 0 observations and 5 variables.
NOTE: DATA statement used (Total process time):
      real time           0.06 seconds
      cpu time            0.00 seconds


MPRINT(DELTAV):   data DSN2;
SYMBOLGEN:  Macro variable DSN2 resolves to sashelp.demographics
MPRINT(DELTAV):   set sashelp.demographics;
MPRINT(DELTAV):   run;

NOTE: There were 0 observations read from the data set SASHELP.DEMOGRAPHICS.
NOTE: The data set WORK.DSN2 has 0 observations and 18 variables.
NOTE: DATA statement used (Total process time):
      real time           0.10 seconds
      cpu time            0.02 seconds


MPRINT(DELTAV):   options obs=max;
MPRINT(DELTAV):   proc sql;
MPRINT(DELTAV):   create table COMMON_VARS as select UPCASE(name) AS COMMON_VARS from dictionary.columns where
libname='WORK' and memname='DSN1' and UPCASE(name) in (select UPCASE(name) AS NAME from dictionary.columns where
libname='WORK' and memname='DSN2') ORDER BY COMMON_VARS;
NOTE: Table WORK.COMMON_VARS created, with 1 rows and 1 columns.

MPRINT(DELTAV):   CREATE TABLE VARS_ONLY_IN_1ST AS select UPCASE(name) AS VARS_ONLY_IN_1ST from dictionary.columns
where libname='WORK' and memname='DSN1' and UPCASE(name) not in (select UPCASE(name) AS NAME from dictionary.columns
```



```
where libname='WORK' and memname='DSN2') ORDER BY VARS_ONLY_IN_1ST;
NOTE: Table WORK.VARS_ONLY_IN_1ST created, with 4 rows and 1 columns.

7                                                    The SAS System                    09:45 Tuesday, August 20, 2019

MPRINT(DELTAV):   CREATE TABLE VARS_ONLY_IN_2ND AS select UPCASE(name) AS VARS_ONLY_IN_2ND from dictionary.columns
where libname='WORK' and memname='DSN2' and UPCASE(name) not in (select UPCASE(name) AS NAME from dictionary.columns
where libname='WORK' and memname='DSN1') ORDER BY VARS_ONLY_IN_2ND;
NOTE: Table WORK.VARS_ONLY_IN_2ND created, with 17 rows and 1 columns.

MPRINT(DELTAV):   QUIT;
NOTE: PROCEDURE SQL used (Total process time):
      real time           0.20 seconds
      cpu time            0.03 seconds


MLOGIC(MAKE_KEY):  Beginning execution.
MLOGIC(MAKE_KEY):  Parameter PARM1 has value COMMON_VARS
SYMBOLGEN:  Macro variable PARM1 resolves to COMMON_VARS
MPRINT(MAKE_KEY):   DATA COMMON_VARS;
SYMBOLGEN:  Macro variable PARM1 resolves to COMMON_VARS
MPRINT(MAKE_KEY):   SET COMMON_VARS;
MPRINT(MAKE_KEY):   MATCH_KEY = _N_;
MPRINT(MAKE_KEY):   RUN;

NOTE: There were 1 observations read from the data set WORK.COMMON_VARS.
NOTE: The data set WORK.COMMON_VARS has 1 observations and 2 variables.
NOTE: DATA statement used (Total process time):
      real time           0.05 seconds
      cpu time            0.01 seconds


MLOGIC(MAKE_KEY):  Ending execution.
MPRINT(DELTAV):  ;
MLOGIC(MAKE_KEY):  Beginning execution.
MLOGIC(MAKE_KEY):  Parameter PARM1 has value VARS_ONLY_IN_1ST
SYMBOLGEN:  Macro variable PARM1 resolves to VARS_ONLY_IN_1ST
MPRINT(MAKE_KEY):   DATA VARS_ONLY_IN_1ST;
SYMBOLGEN:  Macro variable PARM1 resolves to VARS_ONLY_IN_1ST
MPRINT(MAKE_KEY):   SET VARS_ONLY_IN_1ST;
MPRINT(MAKE_KEY):   MATCH_KEY = _N_;
MPRINT(MAKE_KEY):   RUN;

NOTE: There were 4 observations read from the data set WORK.VARS_ONLY_IN_1ST.
NOTE: The data set WORK.VARS ONLY IN 1ST has 4 observations and 2 variables.
```

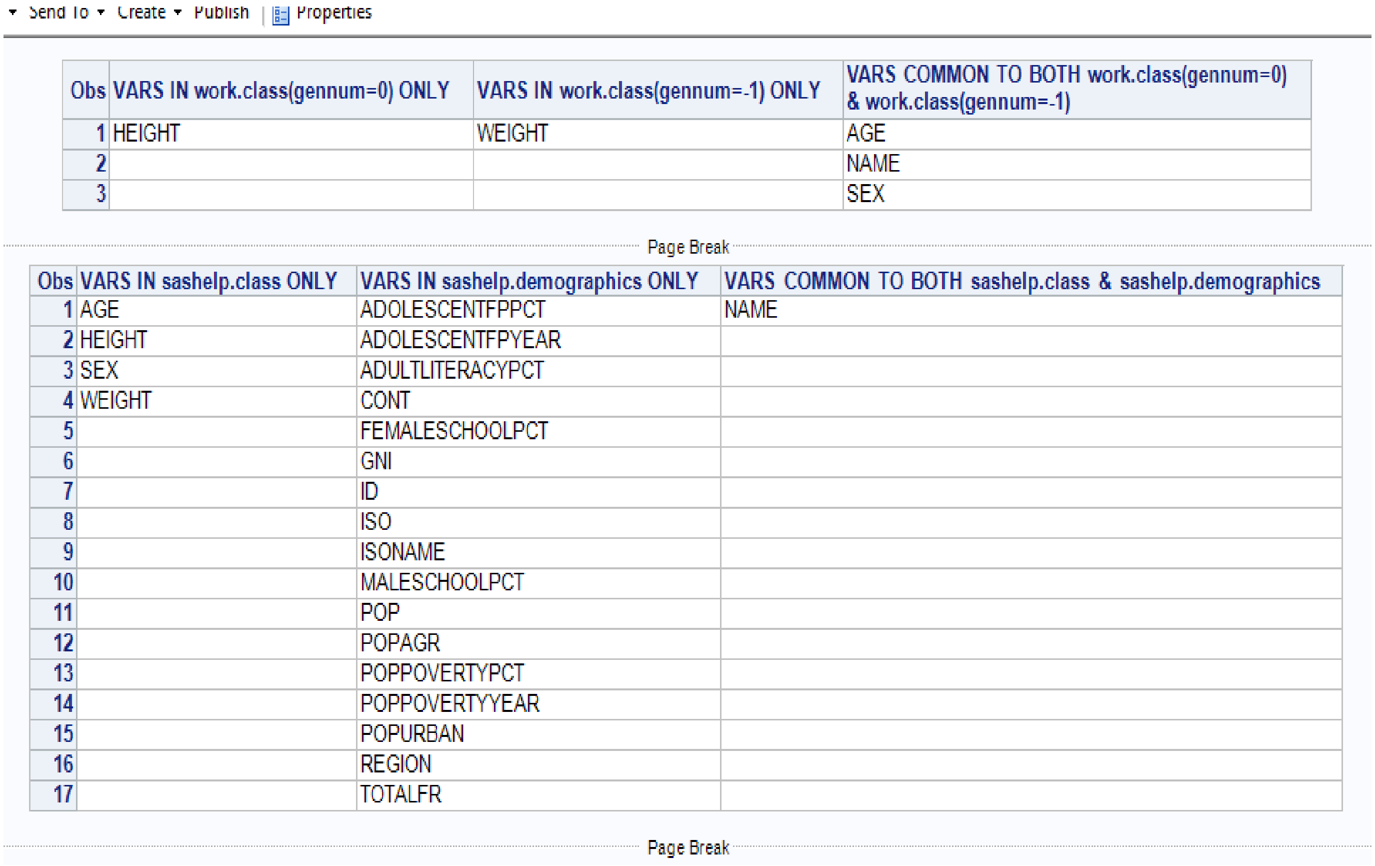


```
NOTE: DATA statement used (Total process time):
      real time           0.05 seconds
      cpu time            0.01 seconds


MLOGIC(MAKE_KEY):  Ending execution.
MPRINT(DELTAV):  ;
MLOGIC(MAKE_KEY):  Beginning execution.
MLOGIC(MAKE_KEY):  Parameter PARM1 has value VARS_ONLY_IN_2ND
SYMBOLGEN:  Macro variable PARM1 resolves to VARS_ONLY_IN_2ND
MPRINT(MAKE_KEY):   DATA VARS_ONLY_IN_2ND;
SYMBOLGEN:  Macro variable PARM1 resolves to VARS_ONLY_IN_2ND
MPRINT(MAKE_KEY):   SET VARS_ONLY_IN_2ND;
MPRINT(MAKE_KEY):   MATCH_KEY = _N_;
MPRINT(MAKE_KEY):   RUN;

NOTE: There were 17 observations read from the data set WORK.VARS_ONLY_IN_2ND.
NOTE: The data set WORK.VARS_ONLY_IN_2ND has 17 observations and 2 variables.
8                                                    The SAS System                      09:45 Tuesday, August 20, 2019

NOTE: DATA statement used (Total process time):
      real time           0.05 seconds
      cpu time            0.00 seconds


MLOGIC(MAKE_KEY):  Ending execution.
MPRINT(DELTAV):  ;
MPRINT(DELTAV):   PROC SQL;
MPRINT(DELTAV):   CREATE TABLE step1 AS SELECT VARS_ONLY_IN_1ST, VARS_ONLY_IN_2ND FROM VARS_ONLY_IN_1ST FULL JOIN
VARS_ONLY_IN_2ND ON VARS_ONLY_IN_1ST.MATCH_KEY = VARS_ONLY_IN_2ND.MATCH_KEY;
NOTE: Table WORK.STEP1 created, with 17 rows and 2 columns.

MPRINT(DELTAV):   QUIT;
NOTE: PROCEDURE SQL used (Total process time):
      real time           0.04 seconds
      cpu time            0.01 seconds


MLOGIC(MAKE_KEY):  Beginning execution.
MLOGIC(MAKE_KEY):  Parameter PARM1 has value STEP1
SYMBOLGEN:  Macro variable PARM1 resolves to STEP1
MPRINT(MAKE_KEY):   DATA STEP1;
SYMBOLGEN:  Macro variable PARM1 resolves to STEP1
MPRINT(MAKE_KEY):   SET STEP1;
MPRINT(MAKE_KEY):   MATCH_KEY = _N_;
```



```
MPRINT(MAKE_KEY):   RUN;

NOTE: There were 17 observations read from the data set WORK.STEP1.
NOTE: The data set WORK.STEP1 has 17 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time           0.06 seconds
      cpu time            0.00 seconds


MLOGIC(MAKE_KEY):  Ending execution.
MPRINT(DELTAV):  ;
MPRINT(DELTAV):   PROC SQL;
MPRINT(DELTAV):   CREATE TABLE getall AS SELECT VARS_ONLY_IN_1ST, VARS_ONLY_IN_2ND, common_vars FROM step1 FULL JOIN
common_vars ON step1.MATCH_KEY = common_VARS.MATCH_KEY;
NOTE: Table WORK.GETALL created, with 17 rows and 3 columns.

MPRINT(DELTAV):   QUIT;
NOTE: PROCEDURE SQL used (Total process time):
      real time           0.04 seconds
      cpu time            0.01 seconds


MPRINT(DELTAV):   proc print data=getall LABEL;
SYMBOLGEN:  Macro variable DSN1 resolves to sashelp.class
SYMBOLGEN:  Macro variable DSN2 resolves to sashelp.demographics
SYMBOLGEN:  Macro variable DSN1 resolves to sashelp.class
SYMBOLGEN:  Macro variable DSN2 resolves to sashelp.demographics
MPRINT(DELTAV):   LABEL COMMON_VARS = "VARS COMMON TO BOTH sashelp.class & sashelp.demographics" VARS_ONLY_IN_1ST =
"VARS IN sashelp.class ONLY" VARS_ONLY_IN_2ND = "VARS IN sashelp.demographics ONLY";
MPRINT(DELTAV):   run;

NOTE: There were 17 observations read from the data set WORK.GETALL.
NOTE: PROCEDURE PRINT used (Total process time):
9                                                   The SAS System                       09:45 Tuesday, August 20, 2019

      real time           0.03 seconds
      cpu time            0.00 seconds


MLOGIC(DELTAV):  Ending execution.
```

RESULTS



Obs	VARS IN work.class(gennum=0) ONLY	VARS IN work.class(gennum=-1) ONLY	VARS COMMON TO BOTH work.class(gennum=0) & work.class(gennum=-1)
1	HEIGHT	WEIGHT	AGE
2			NAME
3			SEX

Obs	VARS IN sashelp.class ONLY	VARS IN sashelp.demographics ONLY	VARS COMMON TO BOTH sashelp.class & sashelp.demographics
1	AGE	ADOLESCENTFPPCT	NAME
2	HEIGHT	ADOLESCENTFPYEAR	
3	SEX	ADULTLITERACYPCT	
4	WEIGHT	CONT	
5		FEMALESCHOOLPCT	
6		GNI	
7		ID	
8		ISO	
9		ISONAME	
10		MALESCHOOLPCT	
11		POP	
12		POPAGR	
13		POPPOVERTYPCT	
14		POPPOVERTYYEAR	
15		POPURBAN	
16		REGION	
17		TOTALFR	

CONCLUSION

The macro can be further enhanced for comparing 3 or 4 or even multiple datasets at once if necessary just by cloning the code presented based on a user's requirement. The more frequently this process is used, the more benefit the team reaps as each time a comparison is created, the differences are presented right in front categorized. This is just another example that shows how DICTIONARY tables can be harnessed to its maximum benefit.

REFERENCES

Bennett, A. D. (2001). SUGI26: Side By Side: Comparing Two Data Sets Using SAS® Macros. In *SUGI26: Side By Side: Comparing Two Data Sets Using SAS® Macros*.

Lee, B. C. (2012). A Simple yet Effective Way to Perform a Variable Cross Walk Between Multiple Datasets. In *A Simple yet Effective Way to Perform a Variable Cross Walk Between Multiple Datasets*. SAS Global Forum.

PG Stats, E. A. (2015, April 12). Retrieved from https://communities.sas.com/t5/SAS-Procedures/Comparing-variable-order-between-2-datasets/td-p/197876

ACKNOWLEDGMENTS

Wellcare management and leadership, for the support and encouragement.

RECOMMENDED READING

- Base SAS® Procedures Guide
- Advanced SAS

CONTACT INFORMATION

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

Dr. Kannan Deivasigamani
TEXTdrk@gmail.com
https://www.linkedin.com/in/kannandeivasigamani