You Could Look It Up: An Introduction to SASHELP Dictionary Views

Michael L. Davis
Bassett Consulting Services, Inc.
September 13, 2000

Why Is This Subject Important?

- many experienced SAS® users have never heard about the SASHELP dictionary views
- offers easy way to monitor a SAS session
- building block of automated applications
- everyone can use these techniques

What Are the Dictionary Views

- tables made available at start of SAS session that contain information about the current session
- use dictionary views as you would any read-only SAS data sets

What Information is Listed in Dictionary Tables?

- libraries, catalogs, and data sets
- external files allocated to session
- macro and data set variables and their attributes
- indexes, titles, footnotes, and views
- system options

Dictionary Tables

- standard implementations of Structured Query Language (SQL) maintain dictionary tables
- some dictionary tables available for DBMSs such as Oracle and DB2
- dictionary tables may be accessed only through PROC SQL

Dictionary Tables Available

- CATALOGS
- COLUMNS
- EXTFILES
- INDEXES
- MEMBERS
- OPTIONS
- TABLES
- VIEWS

SASHELP Views

- replicates information in dictionary tables
- can be used outside PROC SQL
- can use DATA steps to access
- can view from ACCESS Window

Dictionary Tables Vs SASHELP Views

- dictionary tables sometimes faster
- speed difference often not material
- WHERE for large views (VCOLUMN)
- SASHELP Views are more flexible
- author prefers SASHELP Views

How to Find SASHELP Views

- see *Tech Report P-222*, pp. 290-91
- P-222 shows PROC SQL syntax used to create each SASHELP View
- run program on next slide for list
- SASHELP libref views prefixed by "V"

Create List of SASHELP Views

```
proc print data=sashelp.vsview;
  where libname = 'SASHELP' and
  substr(memname, 1, 1) = 'V';
  run;
```

Tour the SASHELP Views

- lists created using PROC CONTENTS
- browse the SASHELP views from
  ->Globals ->Access ->Display Libraries
- see ***views*** prefixed by "V"

LIBREF and FILEREF Views

- VSLIB shows allocated LIBREFs and path information
- VEXTFL shows allocated FILEREFs, including engine
- note that FILEREFs allocated by SAS are prefixed with underscore
- watch out for paths greater than 80 characters

Table Views

- VMEMBER includes data sets, views, and catalogs
- use VMEMBER to get physical path
- use VTABLE to get details (e.g. obs length, NOBS, label, dates)
- VVIEW lists views and engine type

Summary Table Views

- VSTABVW summary data sets and views
- VTABLE summarizes only data sets
- VSVIEW summarizes both automatic and user-created views
- VSACCESS summarizes user-created views

Variable Views

- VCOLUMN shows data set variable details such as type, length, position, format, label, indexing
- VMACRO show details about macro variables such as scope and current value

Catalogs and Indexes

- VCATALG shows details about catalogs and their members
- VSCATALG summarizes catalogs within each libref
- VINDEX links variables to their indexes

Titles and Footnotes

- both are in VTITLE
- T in Title Location for titles
- F in Title Location for footnotes

SAS System Options

- VOPTIONS shows status of system options
- includes portable options that don't apply (e.g. NODMS, FSDEVICE, OPLIST, and TAPECLOSE under Windows)
- does not include system specific options (e.g. WINCHARSET)
- could vary among releases

Some Examples to Illustrate How to Exploit the Dictionary Tables

RDB_MAP.SAS

- presented by Eric Losby at SUGI 22
- cross-tabulate variables by data sets or views
- uses the SASHELP.VCOLUMN view coupled with PROC TRANSPOSE
- MEMTYPE variable controls showing of data sets and views
- use to figure out which SASHELP view to use

Sample RDB_MAP.SAS Listing

Relational Database Map of MYLIB

OBS	NAME	SAVEIT	SPID2DSN	PRGM_NO
1	AUTHFILE	X	X	
2	CLIENT			X
3	CLNT_CD			
4	CLNT_ID	X	X	
5	CLNT_NM	X	X	
6	CLNT_NO			X
7	CRLNFILE	X	X	
8	CUSHION			

%PRINTIT

- presented by Janet Stuelpner and Elizabeth Kaptsanov at SUGI 22
- print NOBS table and sample observations
- PROC SQL using DICTIONARY.TABLES
- submacro %PRT prints NOBS table
- submacro %PRTDS prints each data set
- %PRTDS calls written to macro OUTFILE

%PRINTIT NOBS Listing

DATA SETS FOR C:\MYLIB

OBS	LIBNAME	MEMNAME	NOBS
1	MYLIB	CLIENT	11
2	MYLIB	PRGM_NO	343
3	MYLIB	RPTPARMS	1
4	MYLIB	RSIPSYMB	60
5	MYLIB	SAVEIT	166

%PRINTIT Sample Listing

DATA SET=PRINTIT NOBS=60

OBS	VARNAME	TYPE	FORMAT
1	allstgy	CHAR	$6.
2	authdsn	CHAR	$44.
3	clnt_id	NUM	Z4.
4	clnt_nm	CHAR	$20.
5	cloauth	CHAR	$14.

%PARAMS

- presented by John Gerlach at NESUG 97
- uses SASHELP.VMACRO to save existing macro variables to a data set
- %PARAMS restores the macro variables during a subsequent session
- restored macro variables listed in SAS Log
- check out the consecutive ampersands

%PARAMS Data Set, SAS Log

```
OBS      NAME       VALUE
 1      MADDOG     skeeter

Parameters:
MADDOG      : skeeter
```

%FLATFILE

- presented by Ian Whitlock at SUGI 19 and Michelle Buchecker at SUGI 21
- uses DICTIONARY.VCOLUMN to identify variables and their formats
- useful for exporting SAS data sets to other systems
- http://www.sas.com/service/techsup/unotes/SN/000/000520.html

%FLATFILC

- presented by Dave Mabey at NESUG 97
- modified by Chuck Patridge and Rob Krajcik
- variation of %FLATFILE to write an MS Excel compatible file with column names, labels
- see use of “SEP=”

TIP00122

- written by Charles Patridge and Shiling Zhang
- uses DICTIONARY.COLUMNS to delete many variables from a data set
- http://www.sasconsig.com
- -> SAS Tips and Techniques

TIP00112

- written by Christoph Edel
- uses DICTIONARY.MEMBER to put libref members into macro variable
- http://www.sasconsig.com
- -> SAS Tips and Techniques

Does Format Exist?

```
/* Chris Roper, posted on SAS-L */
proc sql;
    create table aa as
    select *
    from dictionary.catalogs
    where objtype contains 'FORMAT'
       and objname = 'TEST';
  quit;
```

%DOBATCH

- presented by Jingren Shi, Shiling Zhang at SUGI 24
- uses DICTIONARY.EXTFILES to look up the full path to a fileref in order to submit it for batch execution under UNIX Display Manager session
- launched by a function key or icon via AUTOCALL library
- Adobe Acrobat® (PDF) copy of paper available at http://home.att.net/~bassett.consulting/p251-24.pdf

Where Did I Save That Entry?

- by Peter Crawford and posted on SAS-L
- creates a comprehensive view of catalog entries in descending saved order
- shows what has been changed recently
- handy when more than one person changes catalog entries during SAS/AF development
- finds entry copied to wrong directory

Where Did I Save That Entry?

```
proc sql noprint ;
 create view sasuser.vlatest as
  select libname, memname, objname, objtype,
   objdesc format=$23. ,
  input( modified, mmddyy8.)format=date7. as mod
  from dictionary.catalogs
  where libname ne 'SASHELP'
      & libname ne 'MAPS'
  order by mod desc
  ;
quit;
```

Hints from Jack Hamilton

- Use the INTO and SEPARATED BY clauses in PROC SQL to create a macro variable
- Use CALL EXECUTE in a data step
- Use a FILENAME pointed to a catalog entry, and %INCLUDE the results (this is preferable to writing to a temporary disk file, which is not platform-independent)

Example – Put Variables in Alphabetical Order

```
data one;
   key='9'; a=1; c=2; b=.;
   output;
run;
```

Solution – Put Variables in Alphabetical Order

```
proc sql noprint;
    select name
    into :newcmd
    separated by ','
    from dictionary.columns
    where libname='WORK' and
    memname='ONE'
    order by name;
```

Solution - Put Variables in Alphabetical Order (con't)

```
create table two as
    select &NEWCMD.
    from one;

proc print; run ;
```

Cautions from Jack Hamilton

- The examples in this presentation don't do any error checking. You should
- Make sure your code works correctly if nothing is selected by the WHERE clause. You probably need to set the macro variable to blank before you start
- The COLUMNS table can get very large, especially if you have SAS/GRAPH maps. Apply a WHERE clause as soon as you can

Conclusion

- introduced scope of information available from dictionary views
- illustrated some applications for this information
- not difficult to apply
- go out and "look it up"

Acknowledgements

- HASUG Steering Committee
- Jack Hamilton, Clint Rickards, Michael Zdeb
- Robert Krajcik, Charles Patridge
- SAS® is a registered trademark of SAS Institute, Cary, North Carolina

Contact Information

Michael L. Davis
Bassett Consulting Services, Inc.
10 Pleasant Drive
North Haven CT 06473
Tel: 203-562-0640
E-mail: michael@bassettconsulting.com
Web: http://www.bassettconsulting.com

Contents of SASHELP Views in Alphabetical Order

SASHELP.VCATALG

Variable	*Type*	*Len*	*Label*
LIBNAME	Char	8	Library Name
MEMNAME	Char	8	Member Name
MEMTYPE	Char	8	Member Type
OBJNAME	Char	8	Object Name
OBJTYPE	Char	8	Object Type
OBJDESC	Char	40	Object Description
MODIFIED	Char	8	Date Modified
ALIAS	Char	8	Object Alias

SASHELP.VCOLUMN

Variable	*Type*	*Len*	*Label*
LIBNAME	Char	8	Library Name
MEMNAME	Char	8	Member Name
MEMTYPE	Char	8	Member Type
NAME	Char	8	Column Name
TYPE	Char	4	Column Type
LENGTH	Num	8	Column Length
NPOS	Num	8	Column Pos

SASHELP.VCOLUMN (con't)

VARNUM	Num	8	Col No in Tbl
LABEL	Char	40	Col Label
FORMAT	Char	16	Col Format
INFORMAT	Char	16	Col Infmt
IDXUSAGE	Char	9	Col Index Type

SASHELP.VEXTFL

Variable	*Type*	*Len*	*Label*
FILEREF	Char	8	Fileref
XPATH	Char	80	Path Name
XENGINE	Char	8	Engine Name

Note: It is possible to have a path longer than 80 characters!

SASHELP.VINDEX

LIBNAME	Char	8	Library Name
MEMNAME	Char	8	Member Name
MEMTYPE	Char	8	Member Type
NAME	Char	8	Column Name
IDXUSAGE	Char	9	Column Index Type
INDXNAME	Char	8	Index Name
INDXPOS	Num	8	Pos of Col in Concatenated Key
NOMISS	Char	3	Nomiss Option
UNIQUE	Char	3	Unique Option

SASHELP.VMACRO

Variable	*Type*	*Len*	*Label*
SCOPE	Char	9	Macro Scope
NAME	Char	8	Macro Var. Name
OFFSET	Var	8	Offset into Var.
VALUE	Char	200	Macro Var. Value

SASHELP.VOPTION

Variable	*Type*	*Len*	*Label*
OPTNAME	Char	16	Session Option Name
SETTING	Char	200	Session Option Setting
OPTDESC	Char	80	Option Description

SASHELP.VSACCESS

Variable	*Type*	*Len*	*Label*
LIBNAME	Char	8	Library Name
MEMNAME	Char	8	Member Name

SASHELP.VSCATLG

Variable	*Type*	*Len*	*Label*
LIBNAME	Char	8	Library Name
MEMNAME	Char	8	Member Name

SASHELP.VSLIB

Variable	*Type*	*Len*	*Label*
LIBNAME	Char	8	Library Name
PATH	Char	80	Path Name

Note: It is possible to have a path longer than 80 characters!

SASHELP.VMEMBER

Variable	*Type*	*Len*	*Label*
LIBNAME	Char	8	Library Name
MEMNAME	Char	8	Member Name
MEMTYPE	Char	8	Member Type
ENGINE	Char	8	Engine Name
INDEX	Char	8	Indexes
PATH	Char	80	Path Name

includes data sets, views, catalogs

SASHELP.VSTABLE

Variable	*Type*	*Len*	*Label*
LIBNAME	Char	8	Library Name
MEMNAME	Char	8	Member Name

SASHELP.VSTABVW

Variable	*Type*	*Len*	*Label*
LIBNAME	Char	8	Library Name
MEMNAME	Char	8	Member Name
MEMTYPE	Char	8	Member Type

SASHELP.VSVIEW

Variable	*Type*	*Len*	*Label*
LIBNAME	Char	8	Library Name
MEMNAME	Char	8	Member Name

SASHELP.VTABLE

Variable	*Type*	*Len*	*Label*
LIBNAME	Char	8	Library Name
MEMNAME	Char	8	Member Name
MEMTYPE	Char	8	Member Type
MEMLABEL	Char	40	Dataset Label
TYPEMEM	Char	8	Dataset Type
CRDATE	Num	8	Date Created
MODATE	Num	8	Date Modified
NOBS	Num	8	Number of Obs

SASHELP.VTABLE (con't)

OBSLEN	Num	8	Obs Length
NVAR	Num	8	Number of Vars
PROTECT	Char	3	Type Password Protect
COMPRESS	Char	8	Compress Routine
REUSE	Char	3	Reuse Space
BUFSIZE	Num	8	Bufsize
DELOBS	Num	8	No of Deleted Obs
INDXTYPE	Char	9	Type of Indexes

SASHELP.VTITLE

Variable	*Type*	*Len*	*Label*
TYPE	Char	1	Title Location
NUMBER	Num	8	Title Number
TEXT	Char	200	Title Text

SASHELP.VVIEW

Variable	*Type*	*Len*	*Label*
LIBNAME	Char	8	Library Name
MEMNAME	Char	8	Member Name
MEMTYPE	Char	8	Member Type
ENGINE	Char	8	Engine Name

BIBLIOGRAPHY

Buchecker, M. Michelle, "%FLATFILE and Make Your Life Easier, " *Proceeding of the the Twenty-First SAS® User Group Conference*, 178-80

Davis, Michael, "Round Pegs into Square Holes: Data Warehouses for the Hardware Impaired, " *Proceedings of the Twenty-Second SAS® User Group International Conference*, 536-41

BIBLIOGRAPHY (con't)

Gerlach, John R., "The Six Ampersand Solution, " *Proceedings of the 1997 Northeast SAS® Users Group Conference*, 629-30

_____, "A Cross-reference for SAS Data Libraries, " *Proceedings of the 1999 SouthEast SAS® Users Group Conference*, 217-220

BIBLIOGRAPHY (con't)

Hamilton, Jack, "Some Utility Applications Of The Dictionary Tables in PROC SQL, " *Proceedings of the 1998 Western Users of SAS® Software Conference*, 85-90

Jin, Jiang, "Where, When, and How to Use Dictionary Tables in SAS®, " *Proceedings of the 1999 Northeast SAS® Users Group Conference*, 215-18

BIBILIOGRAPHY (con't)

Losby, Eric, "How Are All of These Tables Related – Relational Database Map – RDB_MAP.SAS, " *Proceedings of the Twenty-Second SAS® User Group International Conference*, 1145-49

Mabey, David A., "Eliminating Tedium by Building Applications That Use SQL Generated SAS® Code, " *Proceedings of the 1997 Northeast SAS® Users Group Conference*, **755-60**

BIBILOGRAPHY (con't)

SAS Institute Inc., SAS Technical Report P-222, *Changes and Enhancements to Base SAS® Software, Release 6.07*, Cary, NC: SAS Institute Inc., 1991. 290-91

Shi, Jingren and Zhang, Shiling, "Submitting a Batch SAS® Job within the Display Manager Mode under UNIX®, " *Proceedings of the Twenty-Fourth SAS® User Group International Conference*, 1458-59

BIBLIOGRAPHY (con't)

Stuelpner, Janet E. and Kaptsanov, Elizabeth, "All the Data That Fits, We Print, " *Proceeding of the the Twenty-Second SAS® User Group Conference*, 474-76

Whitlock, H. Ian, "How to Write A Macro to Make External Flat Files," *Proceeding of the the Twenty-First SAS® User Group Conference*, 163-71