How to Build a Data Dictionary – In One Easy Lesson

Gary E. Schlegelmilch, Enterprise Information Services (EIS), Inc.

ABSTRACT

In the wonderful world of programming, the Child Left Behind is usually documentation. The requirements may be thoroughly analyzed (usually on a combination of phoned-in notes, e-mails, draft documents, and the occasional cocktail napkin). Design is often on the fly, due to various restraints, deadlines, and in-process modifications. And doing documentation after the fact, once the program is running, is, well, a great idea – but it often doesn't happen.

Some software tools allow you to build flow diagrams and descriptions from existing code and/or comments embedded in the program. But in a recent situation, there was a lament that a system that had been running in the field for quite a while had no Data Dictionary – and one would be really handy for data standardization and data flow. SAS® to the rescue!

INTRODUCTION

The idea behind this paper is to show how easily a fairly detailed Data Dictionary can be built with any SAS library of data as input. The example shown is actually data from Oracle tables, which shows you its flexibility.

The idea was to build a process that would create a Data Dictionary dataset that could be updated regularly. Eventually, it could them be exported to Excel for common use, and then periodically run against the library (or libraries) to get a fresh picture of the structure, then apply any external updates so no information is lost. Once you see how easy it is to get the base information, you'll also see how flexible you can make it.

KEYWORDS

Data dictionary, PROC TRANSPOSE, PROC SQL

HOW IT'S DONE

I've included the full source code as an Appendix, so you don't have to make yourself crazy cutting and pasting. But this will take you through the process, step by step. The things you will have to put in according to your local processes are in <>.

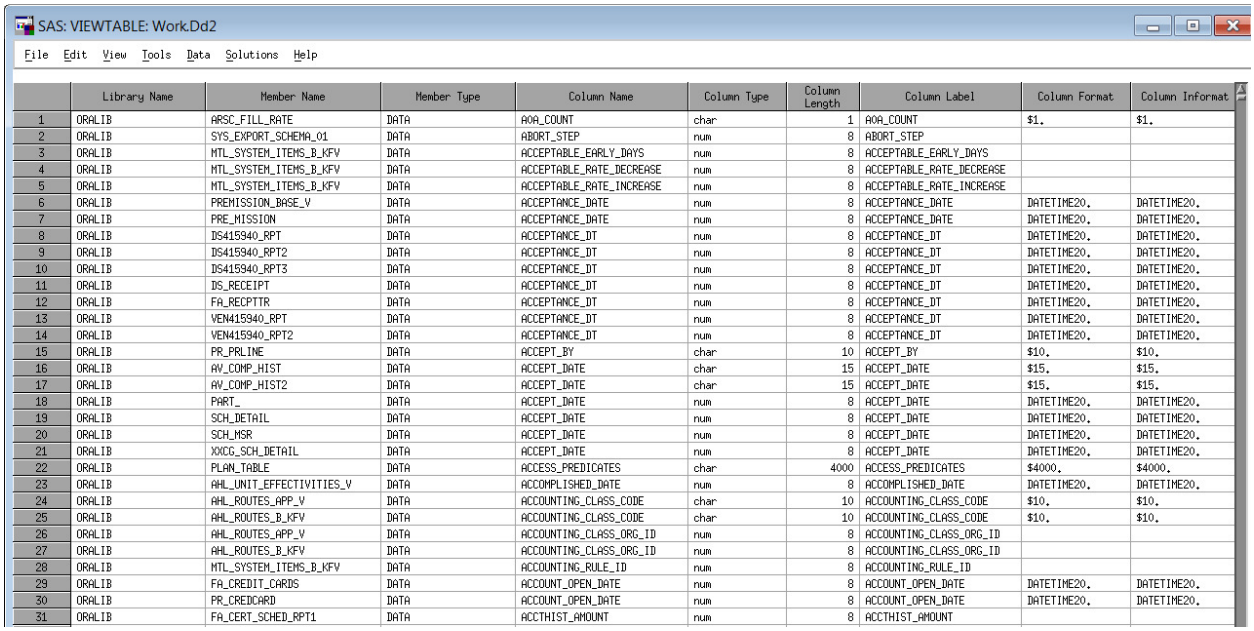
First, define the library of datasets. I named it ORALIB for this paper.

```
libname ORALIB oracle path=    '<location of the data library>'
                       schema=   <schema name>
                       user=     <user id>
                       password="<password>";
```

Use PROC SQL to read in the data. We use PROC SQL here because the DICTIONARY data is only available from PROC SQL. We sort it inline to prepare the data for the PROC TRANSPOSE step. In this case, we are not only looking for data that is listed in more than one dataset, but we are also looking for the potential of problems by having data items with the same name defined differently in different datasets, which could explain things like truncated data or error for using numeric data in a string process.

```
/*  First, read the COLUMNS dataset and get individual field attributes.  */

proc sql;
  create table WORK.DD2
    (DROP=npos varnum idxusage sortedby xtype notnull precision scale
          transcode)
  AS
    select * from DICTIONARY.COLUMNS
             where libname='ORALIB'
             order by name, type, length, format, informat, memname;
quit;
```

This results in this initial dataset:



SAS: VIEWTABLE: Work.Dd2

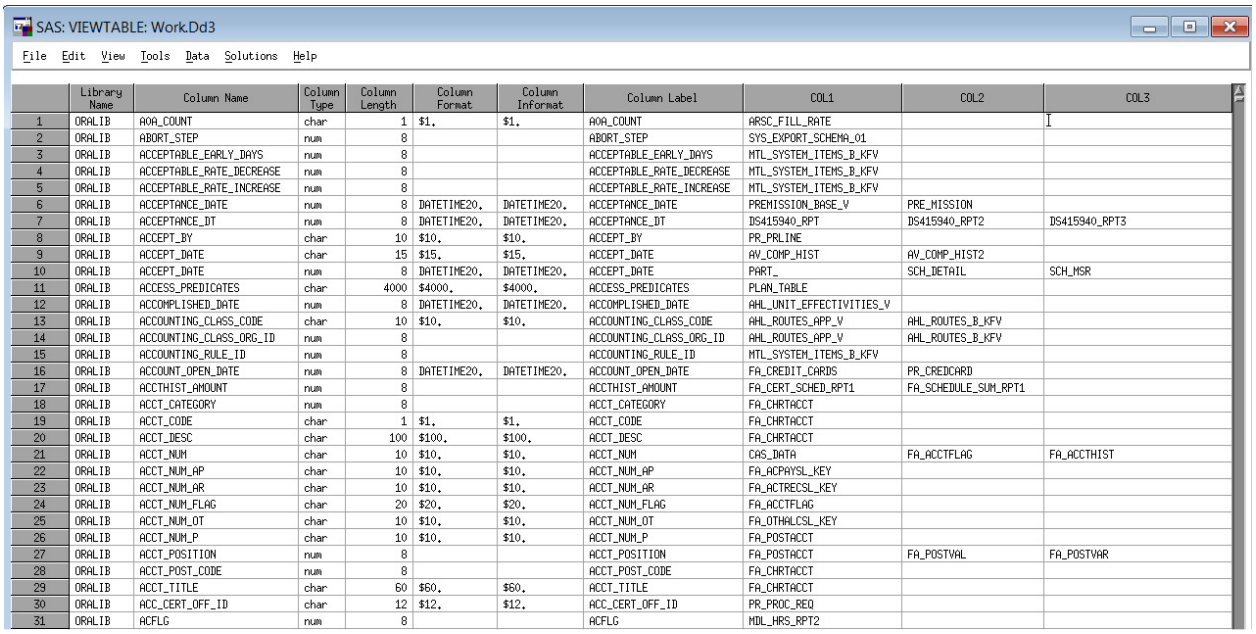
File Edit View Tools Data Solutions Help

	Library Name	Member Name	Member Type	Column Name	Column Type	Column Length	Column Label	Column Format	Column Informat
1	ORALIB	ARSC_FILL_RATE	DATA	AOA_COUNT	char	1	AOA_COUNT	$1.	$1.
2	ORALIB	SYS_EXPORT_SCHEMA_01	DATA	ABORT_STEP	num	8	ABORT_STEP		
3	ORALIB	MTL_SYSTEM_ITEMS_B_KFV	DATA	ACCEPTABLE_EARLY_DAYS	num	8	ACCEPTABLE_EARLY_DAYS		
4	ORALIB	MTL_SYSTEM_ITEMS_B_KFV	DATA	ACCEPTABLE_RATE_DECREASE	num	8	ACCEPTABLE_RATE_DECREASE		
5	ORALIB	MTL_SYSTEM_ITEMS_B_KFV	DATA	ACCEPTABLE_RATE_INCREASE	num	8	ACCEPTABLE_RATE_INCREASE		
6	ORALIB	PREMISSION_BASE_V	DATA	ACCEPTANCE_DATE	num	8	ACCEPTANCE_DATE	DATETIME20.	DATETIME20.
7	ORALIB	PRE_MISSION	DATA	ACCEPTANCE_DATE	num	8	ACCEPTANCE_DATE	DATETIME20.	DATETIME20.
8	ORALIB	DS415940_RPT	DATA	ACCEPTANCE_DT	num	8	ACCEPTANCE_DT	DATETIME20.	DATETIME20.
9	ORALIB	DS415940_RPT2	DATA	ACCEPTANCE_DT	num	8	ACCEPTANCE_DT	DATETIME20.	DATETIME20.
10	ORALIB	DS415940_RPT3	DATA	ACCEPTANCE_DT	num	8	ACCEPTANCE_DT	DATETIME20.	DATETIME20.
11	ORALIB	DS_RECEIPT	DATA	ACCEPTANCE_DT	num	8	ACCEPTANCE_DT	DATETIME20.	DATETIME20.
12	ORALIB	FA_RECPTTR	DATA	ACCEPTANCE_DT	num	8	ACCEPTANCE_DT	DATETIME20.	DATETIME20.
13	ORALIB	VEN415940_RPT	DATA	ACCEPTANCE_DT	num	8	ACCEPTANCE_DT	DATETIME20.	DATETIME20.
14	ORALIB	VEN415940_RPT2	DATA	ACCEPTANCE_DT	num	8	ACCEPTANCE_DT	DATETIME20.	DATETIME20.
15	ORALIB	PR_PRLINE	DATA	ACCEPT_BY	char	10	ACCEPT_BY	$10.	$10.
16	ORALIB	AV_COMP_HIST	DATA	ACCEPT_DATE	char	15	ACCEPT_DATE	$15.	$15.
17	ORALIB	AV_COMP_HIST2	DATA	ACCEPT_DATE	char	15	ACCEPT_DATE	$15.	$15.
18	ORALIB	PART_	DATA	ACCEPT_DATE	num	8	ACCEPT_DATE	DATETIME20.	DATETIME20.
19	ORALIB	SCH_DETAIL	DATA	ACCEPT_DATE	num	8	ACCEPT_DATE	DATETIME20.	DATETIME20.
20	ORALIB	SCH_MSR	DATA	ACCEPT_DATE	num	8	ACCEPT_DATE	DATETIME20.	DATETIME20.
21	ORALIB	XXCG_SCH_DETAIL	DATA	ACCEPT_DATE	num	8	ACCEPT_DATE	DATETIME20.	DATETIME20.
22	ORALIB	PLAN_TABLE	DATA	ACCESS_PREDICATES	char	4000	ACCESS_PREDICATES	$4000.	$4000.
23	ORALIB	AHL_UNIT_EFFECTIVITIES_V	DATA	ACCOMPLISHED_DATE	num	8	ACCOMPLISHED_DATE	DATETIME20.	DATETIME20.
24	ORALIB	AHL_ROUTES_APP_V	DATA	ACCOUNTING_CLASS_CODE	char	10	ACCOUNTING_CLASS_CODE	$10.	$10.
25	ORALIB	AHL_ROUTES_B_KFV	DATA	ACCOUNTING_CLASS_CODE	char	10	ACCOUNTING_CLASS_CODE	$10.	$10.
26	ORALIB	AHL_ROUTES_APP_V	DATA	ACCOUNTING_CLASS_ORG_ID	num	8	ACCOUNTING_CLASS_ORG_ID		
27	ORALIB	AHL_ROUTES_B_KFV	DATA	ACCOUNTING_CLASS_ORG_ID	num	8	ACCOUNTING_CLASS_ORG_ID		
28	ORALIB	MTL_SYSTEM_ITEMS_B_KFV	DATA	ACCOUNTING_RULE_ID	num	8	ACCOUNTING_RULE_ID		
29	ORALIB	FA_CREDIT_CARDS	DATA	ACCOUNT_OPEN_DATE	num	8	ACCOUNT_OPEN_DATE	DATETIME20.	DATETIME20.
30	ORALIB	PR_CREDCARD	DATA	ACCOUNT_OPEN_DATE	num	8	ACCOUNT_OPEN_DATE	DATETIME20.	DATETIME20.
31	ORALIB	FA_CERT_SCHED_RPT1	DATA	ACCTHIST_AMOUNT	num	8	ACCTHIST_AMOUNT		

The PROC TRANSPOSE will put all the gathered dataset names (`memname`) on a single observation. Each dataset name will be put in a generic field name (`COLnnn`). The comment shows the data breakdown and why the VAR and BY statements are used as they are.

```
/*  get all the dataset fields into a single observation, but */
/*  break them out by type, length, format and informat - in  */
/*  case of different utilizations of the same fieldname,     */
/*  and/or aid in data standardization.                       */

proc transpose data=work.dd2
               out= work.dd3 (drop=_name_ _label_);
  var memname;
  by libname name type length format informat label;
run;

proc datasets lib=work;
  delete dd2;
quit;
```

This will produce this:



SAS: VIEWTABLE: Work.Dd3

File Edit View Tools Data Solutions Help

	Library Name	Column Name	Column Type	Column Length	Column Format	Column Informat	Column Label	COL1	COL2	COL3
1	ORALIB	AOA_COUNT	char	1	$1.	$1.	AOA_COUNT	ARSC_FILL_RATE		
2	ORALIB	ABORT_STEP	num	8			ABORT_STEP	SYS_EXPORT_SCHEMA_01		
3	ORALIB	ACCEPTABLE_EARLY_DAYS	num	8			ACCEPTABLE_EARLY_DAYS	MTL_SYSTEM_ITEMS_B_KFV		
4	ORALIB	ACCEPTABLE_RATE_DECREASE	num	8			ACCEPTABLE_RATE_DECREASE	MTL_SYSTEM_ITEMS_B_KFV		
5	ORALIB	ACCEPTABLE_RATE_INCREASE	num	8			ACCEPTABLE_RATE_INCREASE	MTL_SYSTEM_ITEMS_B_KFV		
6	ORALIB	ACCEPTANCE_DATE	num	8	DATETIME20.	DATETIME20.	ACCEPTANCE_DATE	PREMISSION_BASE_V	PRE_MISSION	
7	ORALIB	ACCEPTANCE_DT	num	8	DATETIME20.	DATETIME20.	ACCEPTANCE_DT	DS415940_RPT	DS415940_RPT2	DS415940_RPT3
8	ORALIB	ACCEPT_BY	char	10	$10.	$10.	ACCEPT_BY	PR_PRLINE		
9	ORALIB	ACCEPT_DATE	char	15	$15.	$15.	ACCEPT_DATE	AV_COMP_HIST	AV_COMP_HIST2	
10	ORALIB	ACCEPT_DATE	num	8	DATETIME20.	DATETIME20.	ACCEPT_DATE	PART_	SCH_DETAIL	SCH_MSR
11	ORALIB	ACCESS_PREDICATES	char	4000	$4000.	$4000.	ACCESS_PREDICATES	PLAN_TABLE		
12	ORALIB	ACCOMPLISHED_DATE	num	8	DATETIME20.	DATETIME20.	ACCOMPLISHED_DATE	AHL_UNIT_EFFECTIVITIES_V		
13	ORALIB	ACCOUNTING_CLASS_CODE	char	10	$10.	$10.	ACCOUNTING_CLASS_CODE	AHL_ROUTES_APP_V	AHL_ROUTES_B_KFV	
14	ORALIB	ACCOUNTING_CLASS_ORG_ID	num	8			ACCOUNTING_CLASS_ORG_ID	AHL_ROUTES_APP_V	AHL_ROUTES_B_KFV	
15	ORALIB	ACCOUNTING_RULE_ID	num	8			ACCOUNTING_RULE_ID	MTL_SYSTEM_ITEMS_B_KFV		
16	ORALIB	ACCOUNT_OPEN_DATE	num	8	DATETIME20.	DATETIME20.	ACCOUNT_OPEN_DATE	FA_CREDIT_CARDS	PR_CREDCARD	
17	ORALIB	ACCTHIST_AMOUNT	num	8			ACCTHIST_AMOUNT	FA_CERT_SCHED_RPT1	FA_SCHEDULE_SUM_RPT1	
18	ORALIB	ACCT_CATEGORY	num	8			ACCT_CATEGORY	FA_CHRTACCT		
19	ORALIB	ACCT_CODE	char	1	$1.	$1.	ACCT_CODE	FA_CHRTACCT		
20	ORALIB	ACCT_DESC	char	100	$100.	$100.	ACCT_DESC	FA_CHRTACCT		
21	ORALIB	ACCT_NUM	char	10	$10.	$10.	ACCT_NUM	CAS_DATA	FA_ACCTFLAG	FA_ACCTHIST
22	ORALIB	ACCT_NUM_AP	char	10	$10.	$10.	ACCT_NUM_AP	FA_ACPAYSL_KEY		
23	ORALIB	ACCT_NUM_AR	char	10	$10.	$10.	ACCT_NUM_AR	FA_ACTRECSL_KEY		
24	ORALIB	ACCT_NUM_FLAG	char	20	$20.	$20.	ACCT_NUM_FLAG	FA_ACCTFLAG		
25	ORALIB	ACCT_NUM_OT	char	10	$10.	$10.	ACCT_NUM_OT	FA_OTHALCSL_KEY		
26	ORALIB	ACCT_NUM_P	char	10	$10.	$10.	ACCT_NUM_P	FA_POSTACCT		
27	ORALIB	ACCT_POSITION	num	8			ACCT_POSITION	FA_POSTACCT	FA_POSTVAL	FA_POSTVAR
28	ORALIB	ACCT_POST_CODE	num	8			ACCT_POST_CODE	FA_CHRTACCT		
29	ORALIB	ACCT_TITLE	char	60	$60.	$60.	ACCT_TITLE	FA_CHRTACCT		
30	ORALIB	ACC_CERT_OFF_ID	char	12	$12.	$12.	ACC_CERT_OFF_ID	PR_PROC_REQ		
31	ORALIB	ACFLG	num	8			ACFLG	MDL_HRS_RPT2		

Do a little more cleanup – don't want to waste resources! If you're looking at dozens of datasets with hundreds or perhaps even thousands of total field names, resulting in potentially large numbers of observations in your working datasets, WORK can fill up fast! As an FYI, the resulting count in these datasets were DD2 (just the DICTIONARY.COLUMNS data) was over 14,000; DD3 (results of the PROC TRANSPOSE) and DD4 (final product) was over 5800.

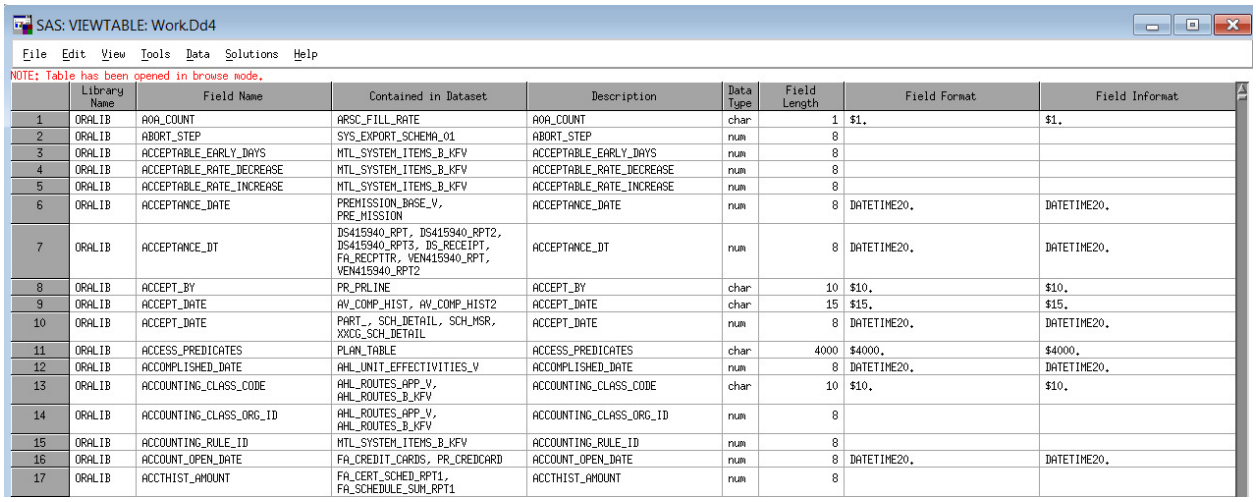
```
proc datasets lib=work;
  delete dd2;
quit;
```

Now we get to the fun part – actually building the Data Dictionary.

```
/*  Build the data dictionary dataset.  */

data work.dd4;

  /*  The LABEL statement serves also to put  */
  /*  the Data Dictionary fields in a logical */
  /*  reading order.                          */

  LABEL    libname  ="Library Name"
           name     ="Field Name"
           libraries="Contained in Dataset"
           label    ="Description"
           type     ="Data Type"
           length   ="Field Length"
           format   ="Field Format"
           informat ="Field Informat";

  length LIBRARIES $2100.;
  set work.dd3;
  LIBRARIES=catx(', ', of col:);  /*  use the OF and ':' wildcard to      */
                                  /*  concatenate ALL COLnnn fields, with */
                                  /*  ', ' as delimiter for readability   */
  drop col:;  /*  use wildcard ':' to remove all COLnnn fields */
run;
```

```
proc datasets lib=work;
  delete dd3;
quit;
```

And now, you're the proud owner of your Data Dictionary!



SAS: VIEWTABLE: Work.Dd4

File Edit View Tools Data Solutions Help

NOTE: Table has been opened in browse mode.

	Library Name	Field Name	Contained in Dataset	Description	Data Type	Field Length	Field Format	Field Informat
1	ORALIB	AOA_COUNT	ARSC_FILL_RATE	AOA_COUNT	char	1	$1.	$1.
2	ORALIB	ABORT_STEP	SYS_EXPORT_SCHEMA_01	ABORT_STEP	num	8		
3	ORALIB	ACCEPTABLE_EARLY_DAYS	MTL_SYSTEM_ITEMS_B_KFV	ACCEPTABLE_EARLY_DAYS	num	8		
4	ORALIB	ACCEPTABLE_RATE_DECREASE	MTL_SYSTEM_ITEMS_B_KFV	ACCEPTABLE_RATE_DECREASE	num	8		
5	ORALIB	ACCEPTABLE_RATE_INCREASE	MTL_SYSTEM_ITEMS_B_KFV	ACCEPTABLE_RATE_INCREASE	num	8		
6	ORALIB	ACCEPTANCE_DATE	PREMISSION_BASE_V, PRE_MISSION	ACCEPTANCE_DATE	num	8	DATETIME20.	DATETIME20.
7	ORALIB	ACCEPTANCE_DT	DS415940_RPT, DS415940_RPT2, DS415940_RPT3, DS_RECEIPT, FA_RECPTTR, VEN415940_RPT, VEN415940_RPT2	ACCEPTANCE_DT	num	8	DATETIME20.	DATETIME20.
8	ORALIB	ACCEPT_BY	PR_PRLINE	ACCEPT_BY	char	10	$10.	$10.
9	ORALIB	ACCEPT_DATE	AV_COMP_HIST, AV_COMP_HIST2	ACCEPT_DATE	char	15	$15.	$15.
10	ORALIB	ACCEPT_DATE	PART_, SCH_DETAIL, SCH_MSR, XXCG_SCH_DETAIL	ACCEPT_DATE	num	8	DATETIME20.	DATETIME20.
11	ORALIB	ACCESS_PREDICATES	PLAN_TABLE	ACCESS_PREDICATES	char	4000	$4000.	$4000.
12	ORALIB	ACCOMPLISHED_DATE	AHL_UNIT_EFFECTIVITIES_V	ACCOMPLISHED_DATE	num	8	DATETIME20.	DATETIME20.
13	ORALIB	ACCOUNTING_CLASS_CODE	AHL_ROUTES_APP_V, AHL_ROUTES_B_KFV	ACCOUNTING_CLASS_CODE	char	10	$10.	$10.
14	ORALIB	ACCOUNTING_CLASS_ORG_ID	AHL_ROUTES_APP_V, AHL_ROUTES_B_KFV	ACCOUNTING_CLASS_ORG_ID	num	8		
15	ORALIB	ACCOUNTING_RULE_ID	MTL_SYSTEM_ITEMS_B_KFV	ACCOUNTING_RULE_ID	num	8		
16	ORALIB	ACCOUNT_OPEN_DATE	FA_CREDIT_CARDS, PR_CREDCARD	ACCOUNT_OPEN_DATE	num	8	DATETIME20.	DATETIME20.
17	ORALIB	ACCTHIST_AMOUNT	FA_CERT_SCHED_RPT1, FA_SCHEDULE_SUM_RPT1	ACCTHIST_AMOUNT	num	8		

CONCLUSION

In just about 60 lines of executable code (and in this case, less than 12 seconds clock time), you have a good tool to always keep an eye on your data.

REFERENCES

Kirk Paul Lafler. October 2013. *PROC SQL Beyond the Basics Using SAS, Second Edition*. Cary, NC: SAS Institute, Inc.

SAS Institute Inc. 2011. *SAS® 9.3 SQL Procedure User's Guide*. Cary, NC: SAS Institute Inc.

SAS Institute Inc. 2012. *Base SAS® 9.3 Procedures Guide, Second Edition*. Cary, NC: SAS Institute Inc.

ACKNOWLEDGMENTS

Much thanks to Kirk Paul Lafler for writing such a great book on PROC SQL; and the hard-working chairs of the 2014 SESUG conference, Abbas Tavakoli and Darryl Putnam.

RECOMMENDED READING

- *Professional SAS Programmer's Pocket Reference,* Rick Aster
- *Learning SAS by Example – A Programmer's Guide,* Ron Cody

CONTACT INFORMATION

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

Name:	Gary E. Schlegelmilch
Enterprise:	Enterprise Information Services (EIS), Inc.
Address:	USCG ISD Bldg 63 ALC
City, State ZIP:	Elizabeth City, NC 27909
Work Phone:	(252) 384-7215
E-mail:	Gary.E.Schlegelmilch@uscg.mil
Web:	www.goeis.com

Appendix. Source Code

```
/*************************************************************************
 * Job:           create_data_dictionary.sas                             *
 *                                                                       *
 * Description:   Small SAS program to create a working Data             *
 *                Dictionary from any SAS library                        *
 *                                                                       *
 * Source Tables: Any SAS data library, DICTIONARY.COLUMNS               *
 *                                                                       *
 * Target Table:  WORK.DD4, output Data Dictionary dataset               *
 *                                                                       *
 * Generated on:  Wednesday, February 5th, 2014                          *
 *                                                                       *
 *  Generated by: Gary E. Schlegelmilch, EIS                             *
 *                                                                       *
 * Version:       1.0                                                    *
 *************************************************************************/

libname ORALIB oracle path=    '<location of the data library>'
                      schema=   <schema name>
                      user=     <user id>
                      password="<password>";

/*  First, read the COLUMNS dataset and get individual field attributes.  */

proc sql;
  create table WORK.DD2
    (DROP=npos varnum idxusage sortedby xtype notnull precision scale
          transcode)
  AS
    select * from DICTIONARY.COLUMNS
             where libname='ORALIB'
             order by name, type, length, format, informat, memname;
quit;

/*  get all the dataset fields into a single observation, but */
/*  break them out by type, length, format and informat - in  */
/*  case of different utilizations of the same fieldname,     */
/*  and/or aid in data standardization.                       */

proc transpose data=work.dd2
               out= work.dd3 (drop=_name_ _label_);
  var memname;
  by libname name type length format informat label;
run;

proc datasets lib=work;
  delete dd2;
quit;

/*  Build the data dictionary dataset.  */

data work.dd4;

  /*  The LABEL statement serves also to put  */
```

```
  /*  the Data Dictionary fields in a logical */
  /*  reading order.                          */

  LABEL    libname  ="Library Name"
           name     ="Field Name"
           libraries="Contained in Dataset"
           label    ="Description"
           type     ="Data Type"
           length   ="Field Length"
           format   ="Field Format"
           informat ="Field Informat";

  length LIBRARIES $2100.;
  set work.dd3;
  LIBRARIES=catx(', ', of col:);  /*  use the OF and ':' wildcard to      */
                                  /*  concatenate ALL COLnnn fields, with */
                                  /*  ', ' as delimiter for readability   */
  drop col:;  /*  use wildcard ':' to remove all COLnnn fields */
run;

proc datasets lib=work;
  delete dd3;
quit;
```