Paper 3136-2019

ETL Macro ToolBox Proc Summary - VRDC Version

Simplifying Adding Class Vars and Low Volume Limits

Zeke Torres SAS Architect
RedMane Technology Chicago Illinois

ABSTRACT

The Center for Medicare and Medicaid Services (CMS) Virtual Research Data Center (VRDC) is a resource that has strict stipulations on how to access and report on data. The ETL-related macros covered in this paper help with understanding the data. Also included is a useful SUMMARY procedure utility to help with obtaining useful facts from data (not just VRDC data).

This set of ETL macros also includes a way to set Low Volume Limits and satisfy the reporting requirements of the VRDC. That "Low Volume Limits" format and the function in this tool box allows for an easier way of reporting and downloading data and meeting the requirements of the VRDC.

MAIN PROBLEM THIS SOLVES

Users who work with the CMS CCW and VRDC will find this paper useful. It covers a typical scenario where reports and results must be created on the VRDC system – and downloaded. This download process must be done via a request for approval by an authorized agent at the CMS CCW VRDC. If that request is rejected – the process (code) must be revised and a new download request made.

With this set of code examples you will see how to quickly enable/disable that Low Volume Limit without the need to revise your code extensively. You'll also have a set of codes to use and customize that are meant to help explore data – typically something that has to be explored during ETL stages.

I use this code to start with small data to put thru and learn about. Then put thru more data as I learn about the data.

HOW TO QUICKLY IMPLEMENT – PROOF OF CONCEPT

Quickest way to this Solution: See the Appendix, Copy Code, run code - go!

READER REQUIREMENTS – SAMPLE DATA - CODE OVERVIEW

USER SKILL LEVEL REQUIREMENTS

The intended reader and user should have a basic or better comprehension of:

- SAS® Macro Language
- PROC FORMAT – with PROC TEMPLATE
- PROC SUMMARY
- PROC FREQ
- Linux/Unix
- Data Used for this Paper/Topic

Extensive knowledge of these items is not a pre-requisite. Great – but not needed.

The paper should provide a useful template for someone to observe those elements working together. Being familiar with those elements is simply going to determine how quickly and

easily the reader/user can take this code configure it to work and get it customized to their situational needs.

DO YOU NEED THE VRDC – WHAT SAMPLE DATA CAN I USE?

I have written this paper to work with "SASHELP(.) Data" and therefore it does not require the use of any VRDC data or access to that environment. A part of this paper is devoted to explaining the CMS VRDC – but access to it is NOT a requirement.

A user with access to Base SAS should have all the necessary components to test and evaluate this code.

SYSTEM AND ENVIRONMENT EXPECTATIONS

The goal is to deploy this code within the CMS VRDC environment. However this code was crafted to work on any operating system where Base SAS can work. For the sake of consistency and simplicity of authoring – I have decided to use a "Linux/Unix" style of code and environmental considerations. This is because the CMS VRDC uses that at its core.

However – I have made every attempt to reduce or remove the influence of "OS" that impact the usefulness and functionality of this ETL ToolBox.

Where it might be relevant – I will call that out to enable you to find the settings, configuration elements needed in your environment.

CODE PERFORMANCE AND EFFICIENCY

This code is intended to be run while you evaluate and gauge the data you are working with. Some parts of it may be useful in your final solution or operational implementation. Some not.

As with many SAS topics – yes there are countless ways to solve this problem and each brings its own efficiency considerations. I encourage you to explore those differences at your leisure and if you could – please share them.

When I crafted this – it was not to push the boundaries of hyper optimal efficiency to the limits. I often wish I had the time and resources to explore those differences and even more to draft them into a paper to share with everyone.

So this paper demonstrates how I've chosen to collect and gather my code "ToolBox" in a way that I can optimize, upgrade and explore the differences as time goes. I frequently use this code and want to have it easily ready and accessible. A set of standard tools handy and ready to interrogate data in a familiar and easy way.

DATA TO EFFICIENCY CONSIDERATIONS

With the exception of Very Large Data, Unstructured Data the following is true:

- Most of the ideas this paper explores will perform just fine for many sizes of data.
- There are other suitable SAS Procedures and/or functions you can swap out if you wish.
- There may be SAS Options within these Procedures that can impact performance.

I've done my best to indicate where some of those differences exist that might be useful for you to be aware of. Regrettably I have not delved deeply into details.

CMS VRDC - BACKGROUND

COMMON HIGH LEVEL TERMS

This paper includes links in the References Section that can help readers learn more about the CMS CCW VRDC.

- CMS – Center for Medicaid and Medicare Services;
- CCW – Chronic Condition Warehouse;
- VRDC – Virtual Resource Data Center.
- DUA – Data Use Agreement
- VRDC User – the person who is using and accessing the VRDC data and systems
- DUA Organization – the organization and/or company who holds the overall DUA agreement with CMS/CCW and is responsible for its VRDC Users and Licenses.
- SAS and SAS EG – SAS Enterprise Guide. The system and software tool of choice for CMS/CCW.

VRDC BASICS

The VRDC is a secure, hosted environment in which CMS/CCW has individual seats or access available to licensed users. These users and the organizations or employers they represent have to undergo a rigorous application process to be granted access to the system. This also requires financial application fees and system fees to be paid.

The VRDC system is a virtual remote desktop which is stripped down to the most basic Windows Operating system and a fully operational license of SAS Enterprise Guide.

That SAS Enterprise Guide license and environment – brings with it access to CMS/CCW data. The Remote Desk Top and VRDC from here in will be referred to as "VRDC" for simplicity.

PATIENT PRIVACY ON VRDC AND CMS DATA

The data that is held within the VRDC system is Medicare and Medicaid claims for patients across the United States healthcare system. The system is not directly accessible to the web and secured. There is sample CMS data available on the web.

This data is for about 100m patients. The entire United States – with some exceptions on some states, counties. But for this paper and practical purposes of the paper – lets just agree – it's a lot of healthcare data.

The important idea to understand is that CMS/CCW strive to enforce and protect patient facts and data. So much so that the system and people using it are monitored.

Key patient facts are de-identified. And I will refrain from delving into details further about the environment and its encryption and processing in an open forum and format to ensure the system continues being safe and secure. I encourage you to read on topics concerning HIPPA and Patient Privacy as well as IFRB and CMS DUA (See References).

MORE ABOUT THE DATA

The quick list of "types" of data on the VRDC includes: Inpatient, Outpatient, Skilled Nursing Facilities, Home Hospice, Part A/B/C just to name a few. Including: Costs, Plan types, Line Items etc.

And it spans numerous years going back in time. Typically cataloged quarterly chunks of the most immediately retrospective date we are on – become available in exploratory releases. Until the conclusion of a 'year' and then with CMS/CCW closing a year – and releasing the final year(s) version.

The data is stored or organized by CMS/CCW in a method such as: File_type/YYYYMM.

So while the data is not exactly real time – it is very much – large data and very much important to ensure security and privacy. Its also updated in a useful frequency to suite most typical research needs.

The subject of the VRDC itself, how the SAS data and views are kept as well as performance considerations regarding SQL, SAS in general – all warrant their own paper and presentation. There are plenty of details and worthwhile bits of information to share.

With that much data – for that many patients its clearly a valuable resource for research and organizations.

The data can allow an organization to learn about:

- Patient Care
- Hospital Performance
- Physician Specialties
- Physician Performance
- Medication and Outcomes
- Treatment Outcomes
- Patient Complications
- Adverse Outcomes for Patients
- Patient Long Term Care
- Episode Analysis
- Cost and Revenue Analysis
- Length of Stay Analysis

These are just a small sampling of possible uses of the data.

GETTING ACCESS TO THE VRDC

Access to that data and the VRDC system – all fall under a DUA and contractual agreement. The DUA is typically something that isn't at a 'user' level – its often part of an organizational contract and agreement. Users of the VRDC have a 'seat' or VRDC license which is only granted after the Organization the user belongs to passes the application process – but the user also completes the correct application and background checks.

The application process of an Organization to obtain an approved DUA and subsequent VRDC seat or license will quickly demonstrate to anyone why this isn't just accessible to just anyone.

Your approved DUA could grant you access to 100% of the data. The DUA request, application you have will determine the data and access you are granted for the research and purposes you disclose and adhere to.

If your organization/employer wrote the DUA correctly and it passes the CMS/CCW review and audits you could be granted access to the data.

If you do not have access to the VRDC – there are still sampling of data available by CMS/CCW for research work. Typical dataset samples are in the 5% range.

RELEVANCE TO THIS PAPER AND CODE

With the requirements for DUA's and applications and fees associated with getting access – its clear that no one wants to jeopardize access to the VRDC. Especially if it involves violating some of the protocols and rules in place.

The important one for this paper is: Low Volume Limits.

Due to the nature of the data and its contents the VRDC imposes some rules on not only what can be uploaded but downloaded.

The reporting and summarization of data on the VRDC can not display facts in figures or groups of patients of LESS THAN 10. If its not – the VRDC will not allow it to be downloaded.

It's a simple rule – and important. In order to protect patients and privacy – we should not ever really construct code or facts that become so low in "counts" that we transcend into questionable forms of practice which inherently jeopardize patient privacy.

The data is meant to help us learn about our healthcare system and ideally provide a benefit to patients – which include us and our community, our families, our loved ones. The things we learn should go towards improving things and not misusing this data.

It's a simple idea to keep in mind as you create reports and code.

In working for years on the VRDC – this solution set has helped.

At best – someone will have a download request of an excel output rejected by the VRDC agent auditing the request. They will inform you why and that person will need to revise the work and adjust the output. The time spent depends on how fast they can make the necessary adjustments and re-request the download.

At worst the VRDC agent can alert your organization that repeated attempts are being made concerning this topic. With obvious escalations that I am happy to have little knowledge of and can not describe here.

Just follow the rules – everyone will be happy.

SAS MACROS

The solution that we are crafting will all revolve around the use of SAS Macros to simplify when we want to include this in to code we are working on.

We will have one long ".SAS" that will hold our code for this paper. However I recommend you learn about how to create SAS Macro Catalogs in order to make your code more portable and accessible.

One thing to be aware of is that many of these macros and code work together.

So as you: evaluate and test, upgrade this code, create more code that uses this code – do not forget how interdependent they could become if they are not already.

KEY MACRO COMPONENTS

MACRO NAME	DESCRIPTION/PURPOSE
DSREPORT1 **See Appendix 4**	This is my version of a more useful Proc Contents. It allows me to see the contents in a way that I can quickly share and send to others who are not typically familiar with SAS Proc Contents Output.
FIELD_STATS **See Appendix 5**	I use this to determine if a field that is say an "ID" or "patient id" – truly is unique. I want to use this when I'm trying to determine – missing, duplicate etc.
FIELD_TOP15 **See Appendix 6**	I use this to quickly get the "top 15" of a field/variable. Giving me an idea of the diversity of a field. And if there are missing observations.
FIELD_FREQ **See Appendix 7**	This is basically just a Proc Freq – but pushed to a table or dataset. I can then see the results better.
FIELD_FREQDT **See Appendix 8**	Same as the Field_Freq – but I'd like to push Dates Thru. And not have issues with forgetting to format the date as : YYYY/MM. Because if I forget – I'll get some really long output that might not be useful.
FIELD_NUMBERS **See Appendix 9**	Having learned from all those other macros – now I'd like to use this to push thru numeric fields and learn more. This is ideal for telling me about: Amounts, Dates, Numbers in general. Since I'm going to obtain things like: min max mean p10 p50 etc. With a more structured output in a table layout.

VRDC AND SQL – PERFORMANCE CONSIDERATIONS

I am eager to hear from SAS and SQL colleagues far and wide about other ways to solve the task and get the same results − and likely with better performance.

Drafting a paper to the SAS community invites countless discussions, conversations and likely arguments. I invite all those constructive comments and ideas. Certainly, ways to use more SQL. But I've taken these Base SAS over SQL or other methods because my experience with SQL on the VRDC. Often a SQL approach has been observed to be inconsistent on that platform. Its resulted in some inconsistent performance. I am not sure where that can be directly attributed to. The VRDC is not open enough at the OS to enable monitoring and gauging how they have configured the SQL engines.

I do look forward to meeting and talking with those who handle and administer this platform to confirm some suspicions I have about this. One is that I believe they are trying to optimize SQL engines and other SQL components. Which they may not announce consistently. So what happens is you might have your code with SQL working great one day/week − then soon(er) or later − it will perform horribly and/or crash.

We could also attribute some of these performance issues to the shared nature of the VRDC. But I've also had mixed results or inconclusive metrics on that.

So in order to facilitate running the same code locally that I would on the VRDC − I've opted for a very Base SAS Centric approach.

GETTING CODE SETUP

Obtain a copy of the code from either the Appendix portion of this paper and/or GitHub. Get the code copied to your local machine with a valid operational Base SAS license.

Put the code in a folder − open Base SAS − initiate the macro(s) and your set.

I have tried this code on SAS University Edition via AWS and Local SAS.

OVERALL CODE SEQUENCE

The key in this process is because we will follow these steps:

1. Using the Format to declare the "Low Value" we want to assign.
2. Use a Proc Template to assign the Format Value to the SAS Template
3. Generate reports or information we need.
4. Remove the Proc Template from our session – because we are done. We don't want to keep it around and cause issues if we leave it. The next report we do may not have the Low Volume Limit set the same.

Here in this four step process – we will be able to swap out "reports" within step 3 with just about anything we choose. You can continue to find ways to adjust or modify to your situation.

The rest of the macros found in this code are meant to facilitate studying your data.

See Appendix 10

PROC TEMPLATE

The statements we need for Proc Template are simply meant to:

1. Before we generate our report/output – we will make a copy of an existing template.
2. Edit or adjust that copy so that it now has the format value we generated from Proc Format.
3. Clean up Step – is also quick – we simply delete the template we adjusted.

PROC FORMAT

The format we are using is straight forward and simple. We will use this format in our process to enable/disable values that are too low for us to report.

What will be a bit different is the use of PROC TEMPLATE to adjust the SAS default to keep this on/off when we run our code.

The useful part about this technique is we can adjust or modify the format as needed. We do not need to use the Proc Template to apply the format. However its useful because doing so reduces the places and areas where we need to modify our code.

COMBINING PROC TEMPLATE - PROC FORMAT – PROC FREQ

The useful part about this technique is we can adjust or modify the format as needed. We do not need to use the Proc Template to apply the format. It offers flexibility by reducing the need to modify code in many places. If we didn't combine this approach – we would need to explicitly declare the: Variable and FORMAT to apply within lines and code at many steps of our process. This method – reduces that. It allows us to run the code with or without the FORMAT – and evaluate. Once we are pleased with the output – we can implement the format – and know the values are adjusted. But the majority of the code stayed intact.

Everyone who writes SAS code has had a chance to explore parts of PROC FREQ. It's a very useful procedure which provides information on the field/variable we would like to study. What we are doing is simply combining the template, PROC FORMAT and PROC FREQ.

LOW VOLUME LIMITS COMBINED WITH THIS PROCESS

See Appendix 10

What we are doing is represented below in these sequence of events.

The area where we "RUN YOUR SAS CODE" can be where you can include or run your code.

Here you can use the Format or use the PROC TEMPLATE method where updates to the template and your desired procedure match up.

So the macros below and in this ETL toolbox – can be used in that "RUN YOUR SAS CODE".

This allows the Low Volume Format and Proc Freq to work well together.

RUNNING THE CODE

I've tried to keep the code consistent from naming perspective. I take the data and create a copy/subset of it. This is a matter of precaution. I do have other versions of my code where I don't make a copy. But I wanted to share this version. With this method – your source data is not jeopardized if you didn't configure the macros correctly or if you had some kind of freak naming convention mishap.
Special notice: It is important for you to decide if the prefix mentioned is ok to use.

The macro naming conventions and items to be aware of are:

Main "libname" for temp data is: work.
You can adjust this by updating to suite your needs

Temp dataset prefix are: tmpx_ and tmpz_
These two are stored within work. lib and
If they are created they are deleted using Proc Datasets with the use of prefix ":"

Above all pay notice to the overall naming convention to avoid any mix up with your existing or future code.

The macro value: **inds** is meant to indicate the "incoming dataset"
The macro value: **vartochk** is meant to hold variable name and/or names
The vars should be listed one by one and space delimited. Here is an example:

```
vartochk= var_one  var_two  var_three
```

WHAT GETS RUN

Here is what you can run once you initiate and run the macro code.

```
%dsreport1(inds=sashelp.baseball);
%field_stats(inds=sashelp.baseball , vartochk=team );
%field_top15(inds=sashelp.baseball , vartochk=team );
%field_freq(inds=sashelp.baseball , vartochk=team );
%field_freqdt(inds=sashelp.stocks  , vartochk=Date );
%field_numbers(inds=sashelp.baseball , vartochk=nhits nhome );
```

Note – I often create smaller data like "work.somesubset" and use that rather than my source data. Because what I'll do when I first start to investigate what is in my data I don't want to put TOO much data in one of these macros.

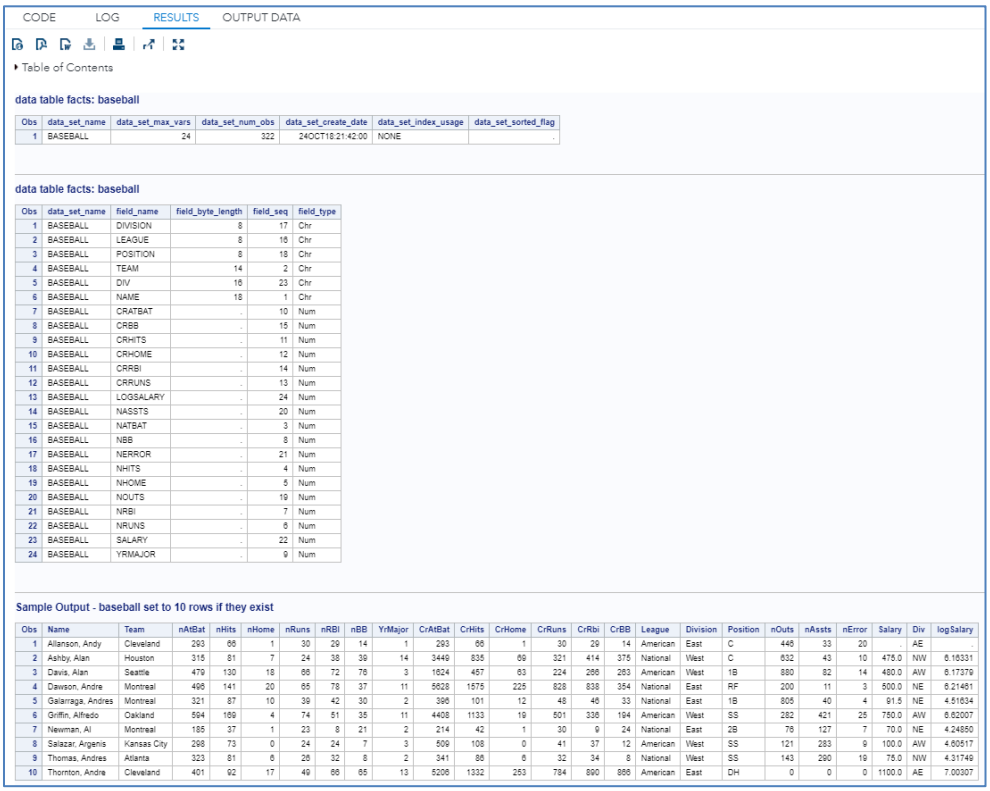
An example of a bad scenario – a FREQ on a variable like: PHONE_NUMBER.

So the rest of this report will cover what the output is once its "run" as shown above.

DSREPORT1

Here is what my optimal or preferred PROC CONTENTS would look like.

```
%dsreport1(inds=sashelp.baseball);
```



CODE LOG RESULTS OUTPUT DATA

Table of Contents

data table facts: baseball

Obs	data_set_name	data_set_max_vars	data_set_num_obs	data_set_create_date	data_set_index_usage	data_set_sorted_flag
1	BASEBALL	24	322	24OCT18:21:42:00	NONE	

data table facts: baseball

Obs	data_set_name	field_name	field_byte_length	field_seq	field_type
1	BASEBALL	DIVISION	8	17	Chr
2	BASEBALL	LEAGUE	8	16	Chr
3	BASEBALL	POSITION	8	18	Chr
4	BASEBALL	TEAM	14	2	Chr
5	BASEBALL	DIV	16	23	Chr
6	BASEBALL	NAME	18	1	Chr
7	BASEBALL	CRATBAT	.	10	Num
8	BASEBALL	CRBB	.	15	Num
9	BASEBALL	CRHITS	.	11	Num
10	BASEBALL	CRHOME	.	12	Num
11	BASEBALL	CRRBI	.	14	Num
12	BASEBALL	CRRUNS	.	13	Num
13	BASEBALL	LOGSALARY	.	24	Num
14	BASEBALL	NASSTS	.	20	Num
15	BASEBALL	NATBAT	.	3	Num
16	BASEBALL	NBB	.	8	Num
17	BASEBALL	NERROR	.	21	Num
18	BASEBALL	NHITS	.	4	Num
19	BASEBALL	NHOME	.	5	Num
20	BASEBALL	NOUTS	.	19	Num
21	BASEBALL	NRBI	.	7	Num
22	BASEBALL	NRUNS	.	6	Num
23	BASEBALL	SALARY	.	22	Num
24	BASEBALL	YRMAJOR	.	9	Num

Sample Output - baseball set to 10 rows if they exist

Obs	Name	Team	nAtBat	nHits	nHome	nRuns	nRBI	nBB	YrMajor	CrAtBat	CrHits	CrHome	CrRuns	CrRbi	CrBB	League	Division	Position	nOuts	nAssts	nError	Salary	Div	logSalary
1	Allanson, Andy	Cleveland	293	66	1	30	29	14	1	293	66	1	30	29	14	American	East	C	446	33	20	.	AE	.
2	Ashby, Alan	Houston	315	81	7	24	38	39	14	3449	835	69	321	414	375	National	West	C	632	43	10	475.0	NW	6.16331
3	Davis, Alan	Seattle	479	130	18	66	72	76	3	1624	457	63	224	266	263	American	West	1B	880	82	14	480.0	AW	6.17379
4	Dawson, Andre	Montreal	496	141	20	65	78	37	11	5628	1575	225	828	838	354	National	East	RF	200	11	3	500.0	NE	6.21461
5	Galarraga, Andres	Montreal	321	87	10	39	42	30	2	396	101	12	48	46	33	National	East	1B	805	40	4	91.5	NE	4.51634
6	Griffin, Alfredo	Oakland	594	169	4	74	51	35	11	4408	1133	19	501	336	194	American	West	SS	282	421	25	750.0	AW	6.62007
7	Newman, Al	Montreal	185	37	1	23	8	21	2	214	42	1	30	9	24	National	East	2B	76	127	7	70.0	NE	4.24850
8	Salazar, Argenis	Kansas City	298	73	0	24	24	7	3	509	108	0	41	37	12	American	West	SS	121	283	9	100.0	AW	4.60517
9	Thomas, Andres	Atlanta	323	81	6	26	32	8	2	341	86	6	32	34	8	National	West	SS	143	290	19	75.0	NW	4.31749
10	Thornton, Andre	Cleveland	401	92	17	49	66	65	13	5206	1332	253	784	890	866	American	East	DH	0	0	0	1100.0	AE	7.00307

The reason I like my Contents – I get an output that I can share with someone who isn't familiar with a standard SAS Proc Contents. Along with a small sample of data.

FIELD_STATS

This macro tells me if the field I've put thru is distinct and how many distinct values there are.

```
%field_stats(inds=sashelp.baseball , vartochk=team );
```

validating: team variable in: baseball - sashelp.baseball is field unique check

Obs	validation	count_of_distinct_values	is_field_unique
1	team	24	no

FIELD_TOP15

Sometimes I just want to see if the field has the right amount of values for the "major" values I'm expecting. A good use for this is – say a Hospital field in claims. Or Physician field in claims. Who are my top 15 of those?

```
%field_top15(inds=sashelp.baseball , vartochk=team );
```

Top 15 of: team in: baseball

Team	COUNT	PERCENT
Chicago	24	7.45342
New York	24	7.45342
Baltimore	15	4.65839
Kansas City	14	4.34783
Los Angeles	14	4.34783
Milwaukee	14	4.34783
Montreal	14	4.34783
San Francisco	14	4.34783
California	13	4.03727
Minneapolis	13	4.03727
San Diego	13	4.03727
Texas	13	4.03727
Cincinnati	12	3.72671
Cleveland	12	3.72671
Detroit	12	3.72671

FIELD_FREQ

If I do want the full frequency – this is what is useful to me. Not because I've done anything fancy – but because I can spin thru a number of fields with one macro. So you can see the same output – but now we can do this:

```
%field_freq(inds=sashelp.baseball , vartochk=team );
```

Freq of team in baseball

The FREQ Procedure

Team at the End of 1986				
Team	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Atlanta	11	3.42	11	3.42
Baltimore	15	4.66	26	8.07
Boston	10	3.11	36	11.18
California	13	4.04	49	15.22
Chicago	24	7.45	73	22.67
Cincinnati	12	3.73	85	26.40
Cleveland	12	3.73	97	30.12
Detroit	12	3.73	109	33.85
Houston	11	3.42	120	37.27
Kansas City	14	4.35	134	41.61
Los Angeles	14	4.35	148	45.96
Milwaukee	14	4.35	162	50.31
Minneapolis	13	4.04	175	54.35
Montreal	14	4.35	189	58.70
New York	24	7.45	213	66.15
Oakland	12	3.73	225	69.88
Philadelphia	12	3.73	237	73.60
Pittsburgh	11	3.42	248	77.02
San Diego	13	4.04	261	81.06
San Francisco	14	4.35	275	85.40
Seattle	12	3.73	287	89.13
St Louis	11	3.42	298	92.55
Texas	13	4.04	311	96.58
Toronto	11	3.42	322	100.00

Freq of league in baseball

The FREQ Procedure

League at the End of 1986				
League	Frequency	Percent	Cumulative Frequency	Cumulative Percent
American	175	54.35	175	54.35
National	147	45.65	322	100.00

Freq of position in baseball

The FREQ Procedure

Position(s) in 1986				
Position	Frequency	Percent	Cumulative Frequency	Cumulative Percent
13	1	0.31	1	0.31
1B	31	9.63	32	9.94
1O	1	0.31	33	10.25
23	1	0.31	34	10.56
2B	31	9.63	65	20.19
2S	1	0.31	66	20.50
32	1	0.31	67	20.81
3B	32	9.94	99	30.75
3O	1	0.31	100	31.06
3S	3	0.93	103	31.99
C	40	12.42	143	44.41
CD	1	0.31	144	44.72
CF	26	8.07	170	52.80
CS	1	0.31	171	53.11
DH	16	4.97	187	58.07
DO	2	0.62	189	58.70
LF	25	7.76	214	66.46
O1	4	1.24	218	67.70
OD	1	0.31	219	68.01
OF	30	9.32	249	77.33
OS	2	0.62	251	77.95
RF	26	8.07	277	86.02
S3	1	0.31	278	86.34
SS	30	9.32	308	95.65
UT	14	4.35	322	100.00

FIELD_FREQDT

This macro uses Proc Freq – but for Dates. And I don't have to remember to apply a useful format. All we need to have is a valid SAS Date value. This macro does the rest. It also works if you add more fields to "VARTOCHK". So can quickly examine many fields.

```
%field_freqdt(inds=sashelp.stocks , vartochk=Date );
```

Please note that in the other macro examples here in this paper I was able to use sashelp.baseball. For this macro I need a date field. Sashelp.stocks has that such field.

Freq of Date in stocks

The FREQ Procedure

Date	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1986AUG	3	0.43	3	0.43
1986SEP	3	0.43	6	0.86
1986OCT	3	0.43	9	1.29
1986NOV	3	0.43	12	1.72
1986DEC	3	0.43	15	2.15
1987JAN	3	0.43	18	2.58
1987FEB	3	0.43	21	3.00
1987MAR	3	0.43	24	3.43
1987APR	3	0.43	27	3.86

This is not a full visual of all the values in sashelp.stocks. It's a small cut/paste image.

FIELD_NUMBERS

This macro allows me to see the numeric variables in my data and the variety of facts about them. I've configured my macro to use some common stat output in Proc Summary. This way I can see if fields are viable for analytic use.

```
%field_numbers(inds=sashelp.baseball , vartochk=nhits nhome );
```

validating: nhits variable in: baseball - sashelp.baseball
numeric field stats

Obs	validation	cnt_distinct_values	min	max	mean	stdv	p10	p25	p50	p75	p90
1	nhits	322	31	238	103.398	44.1795	49	68	98.5	138	163

validating: nhome variable in: baseball - sashelp.baseball
numeric field stats

Obs	validation	cnt_distinct_values	min	max	mean	stdv	p10	p25	p50	p75	p90
1	nhome	322	0	40	11.1025	8.69877	2	4	8.5	16	24

So with this method – we now have a quick way to learn about the data.

CONCLUSION

This paper describes how to bring together a few useful macros into a "ToolBox". This allows you to customize some common things you would use in your day to day work.

How employing a simple Proc Format with these macros can allow you a way to toggle between the report you want to review before downloading and the final report to create when you actually do download from the VRDC.

THE FINE PRINT

This software is provided to you "as is" without any warranties, express or implied, including but not limited to implied warranties or merchantability and/or fitness for a particular purpose. The Author, SAS Institute and its licensor(s) disclaim any liability connected with the use of the software and/or proposed code solution presented. The Author, SAS Institute offers no technical support for the software and/or proposed code solution presented.

ABOUT CODE PERFORMANCE AND DATA

In the early part of the paper I have stipulated that performance and the size/scope of data are relevant. It is imperative that if you did configure this code to run on data like that found in CMS CCW VRDC – that you be aware of the idea of code performance and the data.

I do not recommend anyone simply run any of these code elements in part or in full against very large data in the first few attempts or while becoming familiar with the code and its operation. Where there should be no severe consequences – likely the code will not perform at its optimum. However – while I will not provide warranty over how this code works on the VRDC because you overlooked these facts – I will take phone calls about how I and my team can help you and your team solve the problems together under a services contract.

BUT WAIT…

After all that in "the fine print" if you still want to exchange some ideas, comment, critique or ask questions – just contact me and lets see what we can figure out together.

REFERENCES

http://support.sas.com/documentation/cdl/en/odsug/61723/HTML/default/viewer.htm#a003139134.htm

https://support.sas.com/resources/papers/proceedings/proceedings/sugi29/092-29.pdf

http://support.sas.com/publishing/pubcat/chaps/62007.pdf

https://support.sas.com/rnd/base/ods/scratch/reporting-styles.pdf

http://support.sas.com/documentation/cdl/en/odsug/61723/HTML/default/viewer.htm#a001020001.htm

http://support.sas.com/resources/papers/proceedings10/090-2010.pdf

ACKNOWLEDGMENTS

I want to thank some awesome co-workers that tolerated my numerous attempts to show them this code and equally tolerated me getting it to work easy enough for anyone to use.

Kay Whitman	MPA Healthcare
Bryn David	NORC
Bartosz Jabłoński	Citibank Europe
Kean Chew	Kemper Insurance
Allan Bowe	sasensei.com

RECOMMENDED READING OR LEARNING

- *SAS® Programming 1: Essentials – Found on the sas.com website under learning.*
- *SAS® For Dummies®*

CONTACT INFORMATION

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

Zeke Torres
SAS Architect
RedMane Technology
Chicago Illinois USA

Email:	ezequiel_torres@redmane.com
LinkedIn:	https://www.linkedin.com/in/zeketorres/
Local SAS User Group:	www.wcsug.com

ABOUT THE AUTHOR

I am a data engineer. The roles I play are preparing and constructing complex data for analytic and statistical use. I enjoy designing and building data driven decision support systems. A typical role for me is that of: data management, reporting and complex analytics. Including the governance of metadata.

I translate complex business definitions, requirements and convert them into algorithms and code. Then tackle the some of the toughest, ugliest most unstructured data (lots of it) and make it valuable and useful.

Crafting end to end "Data to Decision" solutions that matter to an organization. I bridge the gap between the IT DBA's and Statistics Quantitative Analysts. My main tool of choice is SAS®. I'm also learning Python and enjoying it.

One of my hobbies are Local SAS User Groups. I lead the Chicago Area SAS Users Group – WCSUG.com – if you are ever in the area stop by and say hello.

TRADEMARK CITATIONS

APPENDIX 1: MACRO CODE – MACRO DOCUMENTATION

```
/*************************************************************
* PROGRAM NAME :  macros_toolbox_various.sas
* DESCRIPTION :   basic stats and contents on sas data
* AUTHOR:         zeke torres
*************************************************************
* macro naming conventions to be aware of
*       main libname for temp data is:   work.
*          - you can adjust this by updating to suite your needs
*       temp dataset prefix is: tmpx_ and tmpz_
*          - these two are stored within work. lib and
*            if they are created they are deleted using
*            proc datasets delete and that prefix
*       *** special notice ***
*       it is important for you to decide if the prefix mentioned
*       above and overall naming convention will cause any mix up
*       with existing or future code
*
*       inds - is meant to indicate the "incoming dataset"
*       vartochk - is meant to hold variable name and/or names
*            the vars should be listed one by one and space delimited
*            Here is an example:
*            vartochk= var_one  var_two  var_three
*
*************************************************************
* macro mkworktsrc()
*   this macro creates subset data for the parent macro that calls it
*   this way the original ds/table isnt going to be tampered or locked
*     caution on using this method and tool box to explore large data
*     without obs limits first to determine which next method to use
*     this macro is used within these next macros:
*        dsreport1
*        field_stats
*        field_top15
*        field_freq
*        field_freqdt
*        field_numbers
*
*************************************************************
* macro dsreport1(inds=)
*       performs proc contents on the data and prints sample 10
*       rows (first 10) the contents information is condensed
*************************************************************
* macro field_stats(inds=,vartochk=)
*       this will check fields like -id- to see if they are unique
*       or not this is an ideal macro for fields like
*          patient id, index type ids, claim number
*************************************************************
* macro field_top15(inds=,vartochk=)
*       this will check fields and provide a freq which then is
```

```
*       sorted by descending occrnce only the top 15 are printed
*************************************************************
* macro field_freq(inds=,vartochk=)
*       performs a proc freq on field specified. Caution
*       against using cms fields like provider/hospital, phone
*************************************************************
* macro field_freqdt(inds=,vartochk=)
*       performs a proc freq on field specified. Caution
*       against using cms fields like but for a date field
*       the date will be formated in yyyymm.
*************************************************************
* macro field_numbers(inds=,vartochk=)
*       this will check amount fields to see get
*       basic info on them like
*       min max mean p10 p50 etc
*       with just a few rows of output
*************************************************************/
```

APPENDIX 2: MKWORKTSRC

```
%macro mkworktsrc();
         /* keep data for use in freq and counts */
         data work.tmpz_&sysjobid ;
             set &inds (keep=&vartochk);
              length validation $35.;
              /* use this field -validation - to allow easier
presentation */
              validation='number_of:';
         run;
%mend;
```

APPENDIX 3: DELWORKSRC

```
%macro delworksrc();
      /*** this will clean up tables created - pay special attention
to the prefix ***/
      /*** if you happen to use this kind of prefix in your existing
code you will ***/
      /*** need to modify this macro code to a prefix style that wont
cause issues ***/

      proc datasets lib= work noprint;
           delete tmpy_: ;
           delete tmpz_: ;
      run;

%mend;
```

APPENDIX 4: DSREPORT1

```
%macro dsreport1(inds=);
         data _null_;
             src_scan=scan("&inds",2,'.');
             call symput('src',src_scan);
         run;

         ods proclabel="Contents of file: &src";
         /* only keep certain fields from contents */
         proc contents data=&inds     noprint
                       out=work.tmpz_table_details
                       (keep=memname varnum   nobs
                             Crdate  idxusage sorted
                             Name    type     length
                             rename=(memname=data_set_name)
                             );
         run;

         data work.tmpz_table_facts
                        (keep=data_set_name
                              varnum
                              nobs
                              idxusage
                              sorted
                              crdate
                              rename=(varnum  =data_set_max_vars
                                      nobs    =data_set_num_obs
                                      idxusage=data_set_index_usage
                                      sorted  =data_set_sorted_flag
                                      crdate  =data_set_create_date
                                      ))
              work.tmpz_table_details
                        (keep=data_set_name
                              field_type
                              length
                              name
                              varnum
                              rename=(name  =field_name
                                      length=field_byte_length
                                      varnum=field_seq
                                      ));
              set work.tmpz_table_details;

             if type='1' then length=.;
             if type='1' then field_type='Num';
             if type='2' then field_type='Chr';
             drop type;
              name=upcase(name);
         run;
```

```
        /* get totals from content details on data */
        proc sort data=work.tmpz_table_facts;
        by data_set_name descending data_set_max_vars;
        run;

        /* rename fields to something client can understand */
        proc sort data=work.tmpz_table_facts nodupkey;
        by data_set_name ;
        run;

        /* print totals for the table */
        ods proclabel="Details of rows in file: &src";
        proc print data=work.tmpz_table_facts;
        title "data table facts: &src";
        run;
        title;

        proc sort data=work.tmpz_table_details ;
        by data_set_name field_type field_byte_length field_name;
        run;

        ods proclabel="Details of columns file: &src";
        proc print data=work.tmpz_table_details width=uniform;
        title "data table facts: &src";
        run;
        title;

        /* print 30 rows of data for sample */
        ods proclabel="Sample of Rows: &src";
        proc print data=&inds (obs=10) width=uniform;
        title "Sample Output - &src set to 10 rows if they exist";
        run;
        title;

%mend;  ****************** end macro;
```

APPENDIX 5: FIELD_STATS

```
/* this will check fields like -id- to see if they are unique or not
*/
%macro field_stats(inds=,vartochk=);
      data _null_;
           src_scan=scan("&inds",2,'.');
           call symput('src',src_scan);
      run;

          %mkworktsrc;

      %let dschk_cnt = %sysfunc(countw(&vartochk, ' '));
      %do ggg = 1 %to &dschk_cnt;
          %let local_var = %scan(&vartochk, &ggg);

      proc freq data=work.tmpz_&sysjobid  noprint;
      title1 "audit of data - &inds - for field &local_var";
      by validation;
         table &local_var /nocol norow nopercent
                                 out=work.tmpy_&sysjobid (drop=percent);
      run;
      title;

      proc summary data=work.tmpy_&sysjobid  nway noprint;
          class validation;
          var count;
          output out=work.tmpy_&sysjobid  (drop=_type_ _freq_)
                       n=count_of_distinct_values
                       max(count)=validate_unique;
      run;

          data work.tmpy_&sysjobid;
              set work.tmpy_&sysjobid;
              length validation $35.;
              validation="&local_var";
                is_field_unique='yes';
                if validate_unique gt 1 then do;
                   is_field_unique='no';
                end;
                drop validate_unique;
          run;

      ods proclabel="Validation of &local_var in: &src";
      proc print data=work.tmpy_&sysjobid  width=uniform;
          title1 "validating: &local_var  variable  in: &src - &inds";
          title2 "is field unique check";
      run;
      title;

      %end; *** end of do loop;
          %delworksrc();

%mend;  ******************** end macro;
```

APPENDIX 6: FIELD_TOP15

```
%macro field_top15(inds=,vartochk=);
     data _null_;
          src_scan=scan("&inds",2,'.');
          call symput('src',src_scan);
     run;

         %mkworktsrc;

     %let dschk_cnt = %sysfunc(countw(&vartochk, ' '));
     %do ggg = 1 %to &dschk_cnt;
         %let local_var = %scan(&vartochk, &ggg);


         proc freq data=work.tmpz_&sysjobid   order=freq noprint;
         title1 "audit of data - &inds - for field &local_var";
             table &local_var  /nocol norow  out=work.tmpy_&sysjobid;
         run;
         title;

         data work.tmpy_&sysjobid ;
             set work.tmpy_&sysjobid  (obs=15);
             if percent lt 0 then do;
                 &local_var = 'missing';
             end;
         run;

         ods proclabel="Top 15 of &local_var variable in: &src";
         proc print data=work.tmpy_&sysjobid  noobs width=uniform;
         where percent gt 0;
         title "Top 15 of: &local_var  in: &src ";
         run;
         title;

         ods proclabel="Any missing for &local_var variable in: &src";
         proc print data=work.tmpy_&sysjobid  noobs width=uniform;
             where percent lt 0;
             var &local_var  count;
             title "Number of Missing: &local_var variable in: &src ";
         run;
         title;

     %end; *** end of do loop;

         %delworksrc();

%mend;  ****************** end macro;
```

APPENDIX 7: FIELD_FREQ

```
%macro field_freq(inds=,vartochk=);
         data _null_;
             src_scan=scan("&inds",2,'.');
             call symput('src',src_scan);
         run;

         %mkworktsrc;

     %let dschk_cnt = %sysfunc(countw(&vartochk, ' '));
     %do ggg = 1 %to &dschk_cnt;
         %let local_var = %scan(&vartochk, &ggg);

         ods proclabel="Freq of &local_var in &src";
         proc freq data=work.tmpz_&sysjobid;
         title "Freq of &local_var in &src";
             table &local_var  /nocol norow ;
         run;
         title;

     %end; *** end of do loop;
         %delworksrc();
%mend;  *************end macro;
```

APPENDIX 8: FIELD_FREQDT

```
%macro field_freqdt(inds=,vartochk=);
         data _null_;
             src_scan=scan("&inds",2,'.');
             call symput('src',src_scan);
         run;

         %mkworktsrc;

     %let dschk_cnt = %sysfunc(countw(&vartochk, ' '));
     %do ggg = 1 %to &dschk_cnt;
         %let local_var = %scan(&vartochk, &ggg);

         ods proclabel="Freq of &local_var in &src";
         proc freq data=work.tmpz_&sysjobid ;
         title "Freq of &local_var in &src";
             table &local_var  /nocol norow ;
             format &local_var yymon8.;
         run;
         title;

     %end; *** end of do loop;
         %delworksrc();
%mend;  *************end macro;
```

APPENDIX 9: FIELD_NUMBERS

```
/* this will check amount fields to see get basic info on them */
%macro field_numbers(inds=,vartochk=);
      data _null_;
           src_scan=scan("&inds",2,'.');
           call symput('src',src_scan);
      run;

          %mkworktsrc;

      %let dschk_cnt = %sysfunc(countw(&vartochk, ' '));
      %do ggg = 1 %to &dschk_cnt;
          %let local_var = %scan(&vartochk, &ggg);

          data work.tmpy_&sysjobid;
              set work.tmpz_&sysjobid  (keep= &local_var);
              length validation $35.;
              validation="&local_var";
              audit=&local_var;
              drop &local_var;
          run;

          proc summary data=work.tmpy_&sysjobid  nway noprint;
              class validation;
              var   audit;
              output out=work.tmpy_&sysjobid  (drop=_type_ _freq_)
                           n=cnt_distinct_values
                           min(audit) =min
                           max(audit) =max
                           mean(audit)=mean
                           std(audit) =stdv
                           p10(audit) =p10
                           p25(audit) =p25
                           p50(audit) =p50
                           p75(audit) =p75
                           p90(audit) =p90
                           ;
          run;

      ods proclabel="Validation of &local_var in: &src";
      proc print data=work.tmpy_&sysjobid  width=uniform;
          title1 "validating: &local_var  variable  in: &src - &inds";
          title2 "numeric field stats";
      run;
      title;

      %end; *** end of do loop;

          %delworksrc();
%mend; ************ end macro;
```

APPENDIX 10: LOW VOLUME FORMAT CODE WITH TEMPLATE

```
%let LOW_VOL_THRESHOLD = 10;   /* set to 10 for VRDC downloads */

proc format;
    /* Default is to scrub values 10 or fewer */
    value scrub
       0-&LOW_VOL_THRESHOLD. = -999
       other=[12.0]
       ;
run;

/* Format the freq output to allow scrubbing. */
proc template;
    edit base.freq.OneWayList;
        edit Frequency;
           format=scrub.;
        end;
    end;
    edit base.freq.CrossTabFreqs;
        edit Frequency;
            format=scrub.;
        end;
    end;
run;


proc freq data=sashelp.baseball;
    table team /list missing nocum ;
run;


/* clean up template */
proc template;
    delete base.freq.CrossTabFreqs;
    delete base.freq.OneWayList;
run;
```