

VIEWS

International SAS Programmer Community



News, Hints, Tips and Information for SAS® Users

Issue 35, 3rd Quarter 2006

After a quiet summer, we are approaching the Autumn conference season, with SAS-related events in Prague (SASTek), Dublin (PhUSE) and Birmingham (SAS Forum UK) before the next issue of *VIEWS News*. I'll be at all 3 conferences reporting for *VIEWS News*, so if you see me at any of them please feel free to stop me for a chat.

In this issue of *VIEWS News* I have again included 5 articles from well-known sources: another graphical style tip from LeRoy Bessler, some advice on the Hash object from Steve Morton, new ways of finding out about formats and informats from Peter Crawford, and some more useful tips from the Amadeus Software Consultancy Team. I hope you will find them all interesting and helpful.

If you would like to contribute an article, or discuss the possibility of doing so, please send an email to me at newsletter@views-uk.org. A list of possible subjects for your articles can be found on the *VIEWS* web site, and anyone is very welcome to add to that list by sending emails with your own questions to newsletter@views-uk.org.

Philip R Holland (Newsletter Editor)

Membership of *VIEWS* is **free** and you can register for notification of all future *VIEWS* events by emailing us at membership@views-uk.org, making certain you include your name and postal address. If you would like to receive email notification whenever a new issue of the free quarterly *VIEWS News* is published on the web site, please remember to also include your email address.

Did You Know?

If you have a useful hint or tip, please send it to the Editor, and share it with the *VIEWS* membership.

The Irrefutable Power of Simplicity

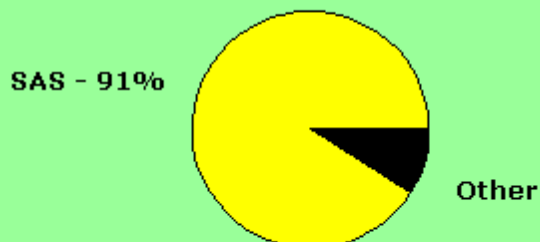
I have spent considerable hours on ways to produce a SAS/GRAPH pie chart without overlap for the trio of labels required for each slice: name/description, value, and percent of whole, i.e. avoiding the situation where the labels for one slice overlap those for an adjacent slice. Also, I have been a strong advocate for refusing to "solve" this overlap problem by consolidating small slices into an Other Slice. Any graph should anticipate and answer questions, not prompt the inevitable question of "What is in Other?"

Here, instead, my "PacMan Pie Chart" in "The Extremes of Other" minimizes labelling and exploits an Other slice in two opposite ways. When you have suitable data and a suitable objective, these maximally simple charts have visual

communication power, with added benefits of easy construction and immunity to label overlap. Their design principle is: "Show Them What's Important". These pie charts deliver an unmistakable message without any distracting constituents of Other.

Case 1. When You Want To Emphasize Largeness of Share, and That What Is in Other Just Doesn't Matter.

Case 1. Content of Other Is Insignificant Mainframe Data Analysis Market Shares - 1993



This example is from the era when mainframe computing was dominant, and I have never been able to get more up-to-date numbers. In principle, you could supply the suppressed detail either with pop-up text (if the graph were HTML-packaged), or with a companion table on the same hard-copy page, or even with merely some footnotes. But, in my opinion, though the identity of the competitive products might be of some interest, clearly their market shares are too small to matter.

Here is the code used to create the graph for Case 1:

```
DATA ForSimplestPie1;
  INFILE DATALINES;
  INPUT @1 software $9. @18 PctShare 2.;
  DATALINES;
SAS - 91%          91
Other              9
RUN;

GOPTIONS RESET = ALL;
GOPTIONS DEVICE = GIF
          GSFNAME = case1 GSFMODE = REPLACE
          XPIXELS = 347 YPIXELS = 261
          HPOS = 38 VPOS = 19
          CBACK = CX99FF99 BORDER
          HTEXT = 1.2 FTEXT = 'Verdana/Bold';

TITLE1 H = 0.5 F = 'Georgia' ' ';
TITLE2 H = 1.2 F = 'Georgia/Bold' J = L
' Case 1. Content of Other Is Insignificant';
TITLE3 H = 0.5 F = 'Georgia' ' ';
TITLE4 H = 1.2 F = 'Georgia/Bold' J = C
'Mainframe Data Analysis Market Shares - 1993';
```

```

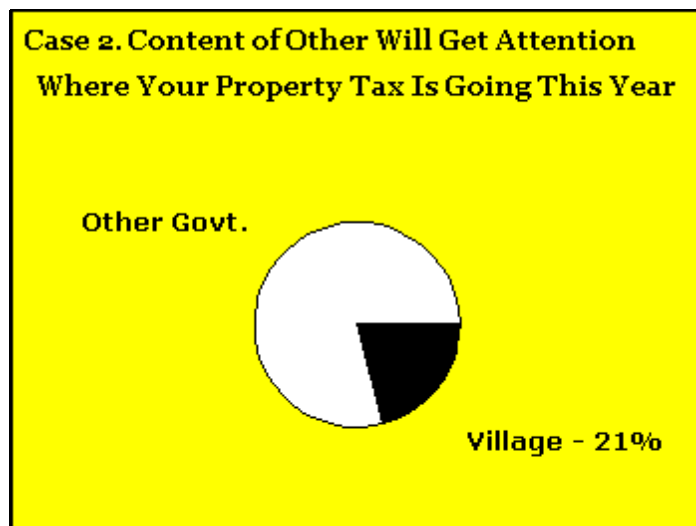
PATTERN1 V = PSOLID C = CXFFFF00;
PATTERN2 V = PSOLID C = CX000000;

FILENAME case1 'c:\views\case1.gif';

PROC GCHART DATA = ForSimplestPie1;
  PIE software / SUMVAR = PctShare
  NOHEADING COUTLINE = CX000000
  SLICE = OUTSIDE VALUE = NONE
  PERCENT = NONE /* same as the default */
  MIDPOINTS = 'SAS - 91%' 'Other';
RUN;
QUIT;

```

Case 2. When You Want to Emphasize Smallness of Share, and Force the Question of “What is in Other?”



I developed this chart while serving as an elected Trustee for the Village of Fox Point. Since our Treasurer must also collect property tax on behalf of other taxing units (such as county, elementary and high school districts, technical college district, sewerage district, etc.), Village officials too often received misdirected complaints about the total size of the tax bill. We began the practice of distributing this style of pie chart with the tax bill to discourage misguided blame. All of the details suppressed in the chart (as well as a comparison with one year ago and five years ago) were/are provided in a companion table.

LeRoy Bessler, Assurant Health

A Comparison of Lookup Techniques in SAS Data Step

Or why you really ought to try the Hash object!

A common requirement in data processing is to look up values of a data column in some form of reference table. This may be to validate the value against a domain value set, to determine whether a ‘new’ value already exists in a table or to make a substitution from a reference table – commonly used to substitute a surrogate key for a business key in star schema fact data. Techniques to do this have been available to SAS programmers from the early days, but SAS 9 introduced a new one: the Hash Table. But how does it compare against ‘traditional’ SAS techniques? and how should you decide between the alternatives?

The main choices for lookup in a data step are now:

- User-defined format or informat with PUT or INPUT function
- Index-based read from a table with the SET statement
- Hash table lookup

It should be noted that other techniques exist for matching an entire data set – particularly SQL processing and data step MERGE by key – but this article is concerned with techniques that are used within data step logic directly.

For the examples I will assume we are using lookup to replace a business key with a surrogate – so the lookup is against a Dimension table, and the data being processed comprises new records for a Fact table.

First, let me recap the traditional methods, starting with **Formats or Informats**.

The basic technique is:-

- Read the Dimension to create a CNTLIN data set for the FORMAT procedure
- Run PROC FORMAT with this to create the format or informat required
- Run a data step with the new Fact data, using PUT or INPUT function to reference the format or informat

If the result required is to create a numeric value by looking up a character value we normally use a numeric informat with the INPUT function; if a character result is required, or if we simply want to know if the input key is found, then we normally use a format with the PUT function.

For example:

```

*create control dataset for proc format;
DATA ctl;
  SET Dimension;
  fmtname = 'lookup';
  type = 'I'; /*Informat*/
  start = business_key;
  label = surrogate;
  KEEP fmtname start label type;
RUN;

*execute proc to create lookup format;
PROC FORMAT CNTLIN = ctl;
RUN;

*process new Fact data to substitute key;
DATA Facts_out;
  SET Facts;
  surrogate = INPUT(business_key,lookup.);
RUN;

```

This is efficient to run, but there is significant overhead involved in running PROC FORMAT, which can be very demanding of memory when a large number of rows are in the Dimension table; this can give a practical limit to how large the Dimension can become before the PROC FORMAT step will fail through lack of memory.

It can also be a considerable overhead if the hit-rate for lookup is very low – for example, if you have a million-row Dimension table but the new Facts are only a few hundred records.

So for these cases, an alternative is to use the Dimension table directly and exploit an index on the business key column. The basic technique for **Index-based Read** is:-

- Create an index on the business key in the Dimension table (if one does not exist – quite often this will already be maintained)
- Run a data step with the new Fact data, using SET Dimension KEY=... to look up values directly in the Dimension table

For example:

```
*look up keys using an index;
DATA Facts_out;
  SET Facts;
  IF business_key NE LAG(business_key) THEN DO;
    SET Dimension
      (KEEP = business_key surrogate)
      KEY = business_key;
    *if desired, check _IORC_ value here;
    RETAIN surrogate;
  END;
RUN;
```

Note, this IF statement construct with the RETAIN is needed in case of multiple fact rows with the same business_key value – if the key-based SET statement is executed a second time with the same key value then it will look for *a second record* that matches the value and hence it will fail to find a record.

This method is very economical on memory, because the Dimension table is read dynamically – only a few buffers full of data are in memory at a time along with the index itself; however it may consume significantly more I/O resources if the same part of the Dimension table has to be re-read many times. We can avoid that issue by ensuring the Fact data is sorted by business_key before processing it.

If you need to check to ensure that a value is found, you can add code to use the _IORC_ value that the SET statement returns. For simplicity I have left out such coding from the example, but in real applications you would expect to have some checking in place. Likewise I would normally drop the business_key after substituting it with a surrogate.

The new method available with SAS 9 is the **Hash Table**. This uses object method calls in the data step to build, maintain and search an in-memory table with very fast hash indexing to locate the item in the table. If you have done any SCL programming using Lists you will be familiar with using functions to manage in-memory structures and the hash object's methods should make sense right away; if not then don't worry, they are well explained in the SAS Online Doc.

Hash tables can be loaded directly from a SAS table when you declare the hash object, or you can create an empty hash object and then add data to it with ADD method calls. You can even output the contents of a hash table to a SAS table so that you can reprocess it next time. For full details see *hash objects* in the SAS Language Reference: Dictionary for SAS 9.1.3

So the basic method for **Hash Lookup**, within a single Data step, is:

- Declare and load a Hash object with data from the Dimension table
- Use the FIND method of the Hash object to locate the business_key value and return the corresponding surrogate value

For example:-

```
DATA Facts_out;
  IF _N_ = 0 THEN DO;
    *reference lookup table to ensure the
    correct data type definitions for
    keys. This code will never execute;
    SET Dimension
      (KEEP = business_key surrogate);
  END;

  IF _N_ = 1 THEN DO;
    *declare hash table to look up key,
    loading Dimension data;
    DECLARE HASH _h1
      (DATASET: "Dimension");
    rc = _h1.DEFINEKEY("business_key");
    rc = _h1.DEFINEDATA("surrogate");
    rc = _h1.DEFINEDONE();
    CALL MISSING(business_key, surrogate);
  END;

  *read input table and find a match in
  the hash table;
  SET Facts;
  rc = _h1.FIND();
  *if desired, check the rc value here;
  DROP rc;
RUN;
```

This technique uses significantly less memory overall than the format/informat method and also executes more quickly in many cases than either the format/informat or index read techniques.

Running a comparison between the three methods using some test data showed the following comparative timings. This was using a Facts table of around 1 million rows with lookup against a Dimension table of half a million rows and a 30% hit-rate (i.e. only 30% of the Dimension business_key values actually appear in the Facts).

Technique	Elapsed time	User CPU	System CPU	Memory (max)
Informat with INPUT (total incl. PROC FORMAT execution)	8.55	6.52	0.55	39120k
Indexed read SET statement	3.82	1.46	1.71	213k
Hash table lookup	1.84	1.50	0.31	24185k

A decision between the alternatives will depend on system constraints and priorities for reducing resources used. Clearly if you have extreme memory limits you would prefer the Indexed Read technique – but if lowest elapsed time is your priority then the Hash Table comes out the best on these results.

If you are using SAS Data Integration Studio (a.k.a. ETL Studio) you may be interested to know that release 3.3 of this software offers a new Lookup transform that uses the Hash lookup technique, as an alternative to the original Fact Lookup transform which uses formats. This now means that you can use this great new feature without needing to code it yourself.

Overall performance results will be dependent on many factors – including the number of records in Dimension and Fact tables, the percentage of the Dimension table that the Facts actually reference, size of business key and surrogate columns etc., as well as your SAS environment itself – as with all performance tips, try it with your own data before committing to the technique in many jobs!

Steve Morton, Applied System Knowledge

Ask The Consultant

This part of VIEWS News is where you can get your technical questions answered. Send your questions to the Editor.

Moving Datasets

Q: Can you move datasets instead of copying them?

A: If you would like to move a dataset from one library to another, rather than copy it, you can use the MOVE option with either PROC DATASETS or PROC COPY.

```
PROC DATASETS LIB = old;
  COPY OUT = new MOVE;
  SELECT policies;
RUN;

PROC COPY IN = old OUT = new MOVE;
  SELECT policies;
RUN;
```

Both of these examples will move the dataset called 'policies' from the library called 'old' to the library called 'new'.

Amadeus Software Consultancy Team

Customising Table of Contents Labels in ODS HTML Output

Q: Can you change the labels in an ODS table of contents?

A: [SAS 8] Here is a snippet of code using the contents options and ODS PROCLABEL in order to customise the titles displayed in the table of contents. Note that these options can also be used in PROC REPORT and PROC TABULATE.

```
ODS HTML FILE = 'c:\temp\body.html'
  CONTENTS = 'c:\temp\contents.html'
  FRAME = 'c:\temp\frame.html';
TITLE;

ODS PROCLABEL 'Listing 1';
PROC PRINT DATA = sashelp.class
  CONTENTS = 'Listing of Sashelp';
RUN;

ODS PROCLABEL 'Listing 2';
PROC PRINT DATA = sashelp.class
  CONTENTS = 'Listing of Females in Sashelp';
  WHERE UPCASE(sex) = 'F';
RUN;

ODS PROCLABEL 'Listing 3';
PROC PRINT DATA = sashelp.class
  CONTENTS = 'Listing of Males in Sashelp';
  WHERE UPCASE(sex) = 'M';
RUN;

ODS HTML CLOSE;
```

Produces the following output:

The screenshot shows a web browser window titled 'SAS Output Frame - Microsoft Internet Explorer'. The address bar shows 'C:\Temp\frame.html'. The page content is divided into two main sections. On the left is a 'Table of Contents' with three entries: '1. Listing 1 - Listing of Sashelp', '2. Listing 2 - Listing of Females in Sashelp', and '3. Listing 3 - Listing of Males in Sashelp'. On the right is a data table with 11 rows and 6 columns: Obs, Name, Sex, Age, Height, and Weight. The data represents the first 11 observations of the SASHelp.CLASS dataset.

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83.0
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84.0
10	John	M	12	59.0	99.5
11	Joyce	F	11	51.3	50.5

Amadeus Software Consultancy Team

Formats, Options, and Functions

This section is devoted to the description of useful, or unusual, SAS Formats, Options, and Functions.

I've included a tip from Peter Crawford in this issue, rather than my usual section. It describes a method of finding applicable SAS formats and informats using a SAS Dictionary view, SASHELP.VFORMAT.

Philip R Holland (Newsletter Editor)

SASHELP.VFORMAT

[SAS 9] Use sashelp.vformat:

- to test your value on all the possible formats.
- to test inputting your string with all available informats.
- research what is available compared with SAS Online Doc and/or Technical Support document **ts486.txt**.

Here is a sample program that tries all SAS informats for a text string. It outputs a dataset that may be worth a look - for any informat that does not cause _ERROR_ to be set.

```
DATA vinfmt_test;
  RETAIN triC '10mar06';

  * test the text string against all informats
  ignoring error cases *;
  LENGTH demoN 8;
  SET sashelp.vformat;
  WHERE fmtype = 'I';
  WHERE ALSO fmtname NE : '$';
  _ERROR_ = 0;

  * test the informat *;
  demoN = INPUTN(triC, fmtname);
  FILE LOG COL = col;
```

```

IF _ERROR_ THEN DO;
  _ERROR_ = 0;
  * list informats invalid for string *;
  PUT +2  fmtname +2 @@;
  DELETE;
END;
ELSE DO;
  IF col > 1 THEN PUT;
  PUT fmtname @33 demon BEST20.;
END;
RUN;

```

Peter Crawford, Crawford Software Consultancy

PhUSE News

PhUSE Conference 2006

PhUSE dates & location:

9-11 October 2006 at The Burlington Hotel, Upper Leeson Street, Dublin 4, Ireland

PhUSE registration & hotel booking:

Please visit www.phuse.info/registration.php

PhUSE registration fees:

- Full ticket price: € 600,00 + VAT

PhUSE contact details:

You can find more detailed information at our web site: www.phuse.info. For general enquiries please send an e-mail to: info@phuse.info

PhUSE Committee

In Brief

- Back issues of *VIEWS News* are available from the *VIEWS* web site.
- Phil Mason now has a SAS tips newsletter, which can be requested by going to www.woodstreet.org.uk and then clicking on Tips. Previous tips can also be found on the web site.
- More SAS hints and tips can be found on the SAS FAQ pages www.hollandnumerics.com/sasfaq and also on the Amadeus Software Tips pages at www.amadeus.co.uk/knowledge/tips/index.php.

Diary

Are you organising an event that would be of interest to the VIEWS readership? Let us know as we are interested in all non-profit making events related to SAS.

September 2006

6-8 SASTek, Prague, Czech Republic

For further details see the SASTek 2006 web site, <https://congress.cis-heidelberg.com/SAS/SASTek-2006/start.htm>.

14 Changes and Enhancements in SAS9 for the Applied Statistician, Oxfordshire

See the Amadeus Software web site, www.amadeus.co.uk, for further details.

October 2006

9-11 PhUSE, Dublin

The Pharmaceutical Users Software Exchange (PhUSE) conference is the premier programming event for the pharmaceutical industry in Europe.

19 Tricks of the Trade, Oxfordshire

See the Amadeus Software web site, www.amadeus.co.uk, for further details.

November 2006

1-2 SAS Forum UK, Birmingham

The SAS Forum UK is held each year and focuses on the latest SAS software releases.

28 Changes and Enhancements in SAS9, Oxfordshire

See the Amadeus Software web site, www.amadeus.co.uk, for further details.

December 2006

5 Tricks of the Trade, Edinburgh

See the Amadeus Software web site, www.amadeus.co.uk, for further details.

April 2007

16-19 SAS Global Forum, Orlando, USA

The SAS Global Forum (formerly called SUGI) is held each year and focuses on the technical aspects of SAS software.

May 2007

22-24 SAS Forum International, Stockholm, Sweden

The SAS Forum International is held each year and focuses on the more commercial aspects of SAS software.

All VIEWS event information will be posted and/or emailed to registered members of VIEWS, and will include an event application form. Please send your queries about any VIEWS events to event@views-uk.org, and don't forget to look at the web site for the latest news.

Contacts

Amadeus Software	Email: info@amadeus.co.uk Tel: +44-(0)1993-878287 Web: www.amadeus.co.uk
LeRoy Bessler Assurant Health, Milwaukee, USA	Email: bessler@execpc.com
Peter Crawford Crawford Software Consultancy	Email: peter.crawford@blueyonder.co.uk
Steve Morton Applied System Knowledge	Email: steve.morton@appliedsystem.co.uk
PhUSE	Web: www.phuse.info
SAS UK	Tel: +44-(0)1628-486933 Web: www.sas.com/uk
<u>IEWS</u>	
Web Site	Web: www.views-uk.org
Chairman	Email: chair@views-uk.org
Newsletter Editor	Email: newsletter@views-uk.org
Membership secretary	Email: membership@views-uk.org
Events secretary	Email: event@views-uk.org

IEWS News is published quarterly by **IEWS International SAS Programmer Community**. **IEWS** is a not-for-profit organisation. Our Mission Statement and Rules can be viewed on our web site together with a list of current committee members.