

Distributing Organisation Charts via the Web without SAS/IntrNet™

by Philip R Holland, Consultant, Holland Numerics Ltd, UK

Abstract

In large corporations the maintenance of organisation charts is a necessary operation. Staff need to have up-to-date information on who is working where. Printed organisation charts can only be distributed quickly enough in small companies, so intranets tend to be used in larger organisations.

This paper demonstrates the techniques for generating and maintaining organisation charts based on simple parent/child records held in spreadsheets or SAS® Datasets. The resulting organisation chart is then be converted to an HTML table, using only Base SAS software, for rapid distribution via the intranet.

1. Introduction

The processing described in this paper can be broken up into 3 main phases:

- a. Reading the hierarchy information in the form of Employee-Manager pairs, with only 2 columns, Employee and Manager, required in the SAS Dataset.
- b. Generate the hierarchy above each employee by combining the pairings, so that each manager is treated as another employee, until the top of the tree is reached for each employee. This is simplified by the creation of an Employee-Manager SAS Format, which can be used to recursively trace back the hierarchy for each employee in a SAS Data Step.
- c. Convert this hierarchy table into an HTML table using a SAS Data Step with PUT statements, making sure that second and subsequent references to each employee are not printed.

2. Setting up the Data Environment

The following code is specific to this data and should be adapted for your own particular data source. There are only 3 requirements from this code:

- a. Create as many file references as required to point to the output HTML files.
- b. Choose an identifiable string for use in the SAS Format, so that employees which have no links into the main hierarchy structure can be filtered out.
- c. Fix the maximum length of string permitted for names of employees and managers, which should not be too short, otherwise truncation may affect the matching of names in the SAS Format, or too long, otherwise the report will become too wide to be easily readable. A process of trial and error may be required to determine the optimum value, which will also allow for future data changes.

e.g.

```
*****;  
* Allocate input and output files *;  
*****;  
  
%let path=c:\my documents\papers\usergrou.ps\seugi\seugi17;  
  
libname db "&path\data";  
filename hier1 "&path\data\hierarch.htm"; * Full hierarchy *;  
filename hier2 "&path\data\hier_lim.htm"; * Limited hierarchy *;  
  
*****;  
* Set execution parameters *;  
*****;  
  
%let nomgr=?;  
%let namewidth=25;
```

3. Layout of the Input Data

As described in the introduction, the hierarchy processing only requires a SAS WORK Dataset containing 2 columns, Employee and Manager, with the same variable length, as each column contains text that could appear in both columns somewhere in the data. The Employee variable must also be unique.

e.g.

```
*****;
* Input demo data into hierarchy SAS dataset *;
*****;

data db.hierarch (keep=manager employee);
  length manager employee $&namewide.;
  input @1 manager $char&namewide..
        @26 employee $char&namewide..
        ;
  manager=left(manager);
  employee=left(employee);
*...5...10...15...20...25...30...35...40...45...50...*;
  cards;

The Holland Family
The Holland Family      The Holland Family
Henry Holland           Henry Holland
Henry Holland           George Holland
Henry Holland           Arthur Holland
Arthur Holland          Alexander Holland
Alexander Holland       Raymond Holland
Raymond Holland         Philip Holland
Philip Holland          Sarah Holland
Philip Holland          Rachel Holland
Philip Holland          Jessica Holland
Raymond Holland        Richard Holland
Richard Holland         Jack Holland
Richard Holland         Chloe Holland
Alexander Holland      Alan Holland
Alan Holland            Duncan Holland
Alan Holland            Claire Holland
;
run;

*****;
* Read hierarchy SAS dataset *;
*****;

data work.hierarch (keep=manager employee id);
  set db.hierarch;
  if manager=' ' then return;
  id=_n_;
  output;
run;
```

4. Generating the Hierarchy Lookup Format

PROC FORMAT can be used to generate SAS Formats from hard-coded VALUE and PICTURE statements, but can also create SAS Formats from data held in SAS Datasets, using the CNTLIN= option. The following code generates a SAS Format, \$MANAGER, which maps will return the Manager for each Employee in the original SAS Dataset. All other Employee values, i.e. names stored in the Manager column that are not found in the Employee column, are given a supplied default value, so they can be filtered out later.

e.g.

```
*****;
* Generate SAS Format from hierarchy *;
```

```

*****;

data work.cntlin (keep=fmtname start label hlo type);
  length fmtname $8 start label $&namewide. hlo type $1 length 8;
  set work.hierarch end=eof;
  fmtname='manager';
  type='C';
  hlo=' ';
  length=&namewide.;
  start=employee;
  label=manager;
  output;
  if eof then do;
    fmtname='manager';
    start='*other*';
    label="&nomgr";
    hlo='O';
    output;
  end;
run;

proc sort data=work.cntlin nodupkeys;
  by fmtname start;
run;

proc format cntlin=work.cntlin;
run;

```

5. Generating the Full Hierarchy Structure Dataset

Starting from the 2-column pairing SAS Dataset, the Manager names further up the hierarchy are generated using the \$MANAGER SAS Format. Note that the original Employee name is initially placed in the 10th Name column, but, once the full hierarchy above it has been generated, is moved across to make sure that the top name in the hierarchy is always in the 1st Name column. This means that there is no need to supply the position of each Employee in the hierarchy in the input SAS Dataset.

e.g.

```

*****;
* Generate full hierarchy *;
*****;

data work.fullhier;
  length name01-name10 $&namewide..;
  retain maxname 0;
  array name $ name01-name10;
  set work.hierarch (keep=employee manager) end=eof;
  * Start with original employee and manager names *;
  name{10}=employee;
  name{9}=manager;
  i=9;
  * Generate all manager names up the hierarchy *;
  do while (not (name{i} in (' ','&nomgr"))) and i > 1);
    i=i-1;
    name{i}=put(name{i+1},$manager&namewide..);
  end;
  * Shift all the names across to fill in the gaps *;
  if (10-i) > maxname then maxname=(10-i);
  do j=1 to 10;
    if j le (10-i)
      then name{j}=name{j+i};
    else name{j}=' ';
  end;
end;

```

```

* Record the number of columns actually in use for later *;
if eof then call symput('maxfull',trim(left(put(maxname,z2.))));
run;

```

Finally discard the records which only contain the name at the top of the hierarchy, and sort the list of Employee names according to their respective Managers. The `&MAXFULL` macro variable holds the maximum number of names down the hierarchy and is used heavily throughout the rest of the program.

e.g.

```

proc sort data=work.fullhier (where=(name02 ne ' '))
          out=db.fullhier (keep=name01-name&maxfull);
  by name01 name02-name&maxfull;
Run;

```

6. Building the HTML Table and Web Page

6.1 Brief Introduction to HTML

HTML (HyperText Markup Language) is written as a text file, with formatting instructions included in the form of tags. A tag is a string surrounded by angle brackets (< and >), and most of the tags used are placed in front of the text to be modified (<tag>), with a reversing action tag after that text (</tag>).

In HTML:

For example, this HTML produces a paragraph of text with the selected text either printed in `bold type` or `<U>underlined</U>`.

As printed:

For example, this HTML produces a paragraph of text with the selected text either printed in **bold type** or underlined.

In general a web page is made up of 2 sections between the `<HTML>` and `</HTML>` tags:

```

<HTML>
<HEAD>
  Header section for titles and documentation.
</HEAD>
<BODY>
  Body section for the main content of the page, including graphics, text,
  tables, etc.
</BODY>
</HTML>

```

Between the `<HEAD>` and `</HEAD>` tags, the generated web page that will be created later uses the following additional tag:

- a. `<TITLE>`Title which is placed in the title bar of the browser window`</TITLE>`

Between the `<BODY>` and `</BODY>` tags, the generated web page that will be created later uses the following additional tags:

- a. `<HR>` - draws horizontal line across the web page.
- b. `<CENTER>`Centred text`</CENTER>`
- c. `<H1>`Header level 1 (large bold text)`</H1>`
- d. `<TABLE BORDER>` - create a table with visible cell borders.
`<TR>` - create new row.

```

<TD ALIGN=left> - create new cell.
  Table cell: column 1, row 1 (left aligned)
</TD> - end cell.
<TD ALIGN=center> - create new cell.
  Table cell: column 2, row 1 (centred)
</TD> - end cell.
</TR> - end row.
<TR> - create new row.
  <TD ALIGN=right> - create new cell.
    Table cell: column 1, row 2 (right aligned)
  </TD> - end cell.
  <TD WIDTH=150> - create new cell.
    Table cell: column 2, row 2 (width 150 pixels)
  </TD> - end cell.
</TR> - end row.
</TABLE> - end table.

```

6.2 Generating the HTML Web Page using a SAS Data Step

The SAS processing required to generate the HTML web page can be divided into 3 main actions:

- At the beginning of the Data Step (see the statement `if _n_=1` in the SAS code below), the HTML page is started, the HTML header section is written to the web page file, the HTML body section started, and the HTML table created (see the label `tabhead` in the SAS code below).
- At each record, the hierarchy level is tested. If the record describes data from the very top of a hierarchy, any existing HTML table is closed and a new HTML table created (see the label `tabhead` in the SAS code below). The record is then written out to the HTML table so that only the lowest name in the hierarchy is included in the HTML table.
- At the end of the Data Step (see the statement `if eof` of the SAS code below) the current HTML table is closed, the HTML body section is completed, and the HTML page is closed.

e.g.

```

*****;
* Macro to generate HTML for hierarchy table *;
*****;

%macro _htmlgen(fileref=,select=);

*****;
* Generate HTML for hierarchy table *;
*****;

data _null_;
  length lagnm01-lagnm&maxfull $&namewide. head $80;
  set db.fullhier (where=(&select)) end=eof;
  by name01 name02-name&maxfull;
  array name name01-name&maxfull;
  array lagname lagnm01-lagnm&maxfull;
  file &fileref;
  if _n_=1 then do;
    head="Hierarchy Table";
    put '<HTML>';
    put '<HEAD>';
    put '<TITLE>' head '</TITLE>';
    put '</HEAD>';
    put '<BODY>';
    put '<CENTER><H1>' head '</H1></CENTER>';
    link tabhead;
  end;
  put '<TR>';
  do i=1 to dim(name);

```

```

lagname{i}=lag(name{i});
if name{i} ne lagname{i} then do;
  if i = 1 and _n_ > 1 then do;
    put '</TR>';
    put '</TABLE></CENTER>';
    link tabhead;
    put '<TR>';
  end;
  put '<TD ALIGN=LEFT WIDTH=150>' name{i} '</TD>';
end;
else do;
  put '<TD WIDTH=150> </TD>';
end;
end;
put '</TR>';
if eof then do;
  put '</TABLE></CENTER>';
  put '</BODY>';
  put '</HTML>';
end;
return;
tabhead:
put '<HR>';
put '<CENTER><TABLE BORDER>';
put '<TR>';
do j=1 to dim(name);
  put '<TD ALIGN=CENTER WIDTH=150><B><U>Level ' j '</U></B></TD>';
end;
put '</TR>';
return;
run;
%mend;

```

The macro %_htmlgen has 2 parameters which allow you to specify where the HTML file is written (**fileref**) and the subset of the hierarchy data that is used to produce the organisation chart (**select**).

e.g.

```

*****;
* Generate HTML pages for hierarchy tables *;
*****;

%_htmlgen(fileref=hier1,select=name01 ne ' ')
%_htmlgen(fileref=hier2,select=name05 eq 'Raymond Holland')

```

7. Summary

- Using a data-driven SAS Format to generate the hierarchy reduces the SAS code and data required to create the full structure of the organisation chart.
- Storing the organisation chart in the form of Employee-Manager pairs allows groups of Employees to be relocated by changing the Manager of the top Employee in that group. This should reduce the maintenance effort required.
- A web page is generated to simplify the distribution of the organisation chart to all employees in a large corporation, as they are more likely to have a web browser on their desktop than SAS software.
- The web page is created using only standard Base SAS software, as installed at every SAS installation, and consists of HTML header section, HTML title and one or more HTML tables, which will be compatible with a wide range of web browsers.

8. **References and further reading**

- SAS Language: Reference, Version 6, First Edition.
- SAS Macro Language: Reference, First Edition.

The author is a consultant for Holland Numerics Ltd and can be contacted at the following address:

Philip R Holland
Holland Numerics Ltd
94 Green Drift
Royston
Herts. SG8 5BT
UK
E-mail: <phil.holland@bcs.org.uk>
Web: <http://www.hollandnumerics.demon.co.uk/>
tel. (mobile): +44-(0)7714-279085

SAS and SAS/IntrNet are a registered trademarks of SAS Institute Inc., Cary, NC, USA.