Character Variables – Concatenating, Creating from Numeric Variables, and More

It is possible to create one character variable out of more than one variable – even if the variables are numeric. This is sometimes necessary. For example, you might want to create a total address variable from parts of an address. Or, you might want to create a single identifying variable from two or more variables that uniquely identify a case.

In the following example, I will use the Vietnam Longitudinal Survey (1995-present) available at the following web site, http://csde.washington.edu/csde/vietnam/data.html. The documentation for these data are at: http://csde.washington.edu/csde/vietnam/documents.html.

In this data set, there is not a single variable that uniquely identifies each household. The variable H5Q7 (Survey Household Number) must be used along with the variable H5Q2 (Name of commune (phuong/xa)) to uniquely identify households. The variable H5Q7 is unique within each commune. To demonstrate how to create character variables from numeric variables, and demonstrate how to manipulate character variables in a couple of ways, the following program uses this data set to create a unique character variable from two numeric variables. (It shows several ways - commhh4 is the final and best variable.)

Comments are green, and preceded by an asterisk enclosed by /* and */. The comments explain the program. Read them to help you understand what is going on in the program, and why.

```sas
* charsas.sas;
title1 'c:\all\help\helpnew\charsas.sas';
* Show how to create character variables from numeric variables,
* concatenate character variables, and compress them, test the length,
* and test to make sure you have created a unique ID. ;
*
* Documentation and data from following sites;
* http://csde.washington.edu/csde/vietnam/documents.html;
* http://csde.washington.edu/csde/vietnam/data.html;
** Put Household and Individual Level Data together;
** Store SAS data set with all variables and cases;
options linesize=72 compress = yes;
libname mylib 'c:\all\sasclass\vn95\saslib';
/*
 H5Q2 Name of commune (phuong/xa)
 H5Q7 Survey Household Number */
data hh ; set mylib.hh95vls1 ( keep=h5q2 h5q7) ;
* The "length" statement can be used to initialize a character ;
* variable and specify how long it will be. If you do not specify ;
* sufficient length, the variable will be truncated -- VERY BAD! ;
* It is also possible to initialize a character variable without a ;
* length statement - but, danger exists. The length of the variable ;
* will be set to that of the first value it encounters - which may not ;
* be long enough. ;
length commid $2. hhid $3. commhh $6. ;
commid = h5q2;
hhid = h5q7;
* Two consecutive "PIPES" are used as the concatenate command;
commhh1 = commid || hhid ;  * Value for h5q2=1 and hhid = 1: 1 1
```

---

1Prepared by Patty Glynn, University of Washington. March 12, 2001
** But, that leaves spaces. The compress command can be used to remove spaces.;
commhh2 = compress(commid || hhid); * Value for h5q2=1 and hhid = 1: 11 ;

* But, then there is no indication of where COMMID ends and HHID begins.;
** Another character can be inserted using a literal string.;
commhh3 = compress(commid || '.' || hhid); * Value for h5q2=1 and hhid = 1: 1.1 ;

/* But this is not ideal either. It would be better if, when sorted, households were in numerical order.

<table>
<thead>
<tr>
<th>Obs</th>
<th>commid</th>
<th>H5Q2</th>
<th>hhid</th>
<th>H5Q7</th>
<th>commhh3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>1.10</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>100</td>
<td>100</td>
<td>1.100</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>101</td>
<td>101</td>
<td>1.101</td>
</tr>
</tbody>
</table>

skipped cases
20 1 1 116 116 1.116
21 1 1 117 117 1.117
22 1 1 118 118 1.118
23 1 1 119 119 1.119
24 1 1 12 12 1.12 */

* To fix this, test the length of hhid, add 0s to the beginning if 1 or 2 digit. ;
length hhid2 $3. ;
if length(compress(hhid)) = 1 then hhid2 = compress('00'||hhid) ;
if length(compress(hhid)) = 2 then hhid2 = compress('0'||hhid) ;
if length(compress(hhid)) = 3 then hhid2 = hhid ;
commhh4 = compress(commid || '.' || hhid2 ) ; * Value for h5q2=1 and hhid = 1: 1.001 ;
label commhh4 = 'Unique ID for each HH, Character Variable, best' ;

/*

<table>
<thead>
<tr>
<th>Obs</th>
<th>commid</th>
<th>H5Q2</th>
<th>hhid</th>
<th>H5Q7</th>
<th>commhh1</th>
<th>commhh2</th>
<th>commhh3</th>
<th>hhid2</th>
<th>commhh4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.1</td>
<td>001</td>
<td>1.001</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>12</td>
<td>002</td>
<td>1.002</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>13</td>
<td>003</td>
<td>1.003</td>
<td></td>
</tr>
</tbody>
</table>

skipped cases
1565 9 9 10 10 9 10 910 9.10 010 9.010
1567 9 9 12 12 9 12 912 9.12 012 9.012
1568 9 9 13 13 9 13 913 9.13 013 9.013
1569 9 9 14 14 9 14 914 9.14 014 9.014
1570 9 9 15 15 9 15 915 9.15 015 9.015 */

proc sort; by commhh4 ;
proc print uniform ; var commid h5q2 hhid h5q7 commhh1 commhh2 commhh3 hhid2 commhh4 ;
run ;

** Finally, if you are creating an ID that you think is unique, TEST! ;
** Above, we sorted by the ID that we believe uniquely identifies each household. ;
data two; set hh ;
* Initialize a variable, setting it to zero ;
flag = 0 ;
* Set the value to 1 if the previous case has the same value as the present case. ;
** The data must be sorted in order of this variable BEFORE this test for this to work. ;
if lag(commhh4) = commhh4 then flag = 1 ;
proc freq; tables flag ; run ;
/* The FREQ Procedure

<table>
<thead>
<tr>
<th>flag</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1855</td>
<td>100.00</td>
<td>1855</td>
<td>100.00</td>
</tr>
</tbody>
</table>