SAS 9 PROC TEMPLATE Tables Tip Sheet

Conditional Statements

Traffic Lighting

```
cellstyle expr-1 as style-element{style-attributes},
          expr-2 as style-element{style-attributes},
            …
          expr-n as style-element{style-attributes};
```

The CELLSTYLE-AS statement is valid in both table and column definitions, although only one style is applied in a single data cell. Each expression is a WHERE expression that is evaluated for each data cell. If the result of the expression is true, the given style element and attributes are applied. Once a matching expression is found, execution stops.

Translating Values

```
translate expr-1a into expr-1b,
          expr-2a into expr-2b,
            …
          expr-na into expr-nb;
```

The TRANSLATE-INTO statement is valid in both table and column definitions, although only one result is applied in a single data cell. Each expression is a WHERE expression. The (a) expressions are evaluated for each data cell. If the result of the (a) expression is true, the result of the (b) expression is used in place of the data value in the output. Once a matching expression is found, execution stops.

Expression Variables

column name	name of any column in the table template or data set
col	current column number
dataname	name of the current data column
label	label on the current data column
row	current row number
style	current style element name
val	current data value

Table Template Example

```
proc template;
  define table classroom;
    notes "Classroom BMI Table";

    column name age sex (height weight) bmi;

    define header classroomheader;
      notes "Table header";
      text "Classroom BMI Information";
    end;

    define column height;
      define header heightweight;
        notes "Header for Height/Weight";
        text "Height/Weight";
        split = "/";
      end;
      header = heightweight;
    end;

    define bmi;
      notes "Compute BMI (Body Mass Index)";
      header = "BMI";
      compute as (weight*703) /
                 (height*height);
      cellstyle _val_ < 18
                as {background=yellow};
      format = d5.1;
    end;
  end;
run;

data _null_;
  set sashelp.class;
  file print ods=(template="classroom");
  put _ods_;
run;
```

Classroom BMI Information				
Name	**Age**	**Sex**	**Height Weight**	**BMI**
Alfred	14	M	69 112.5	16.6
Alice	13	F	56.5 84	18.5
Barbara	13	F	65.3	16.2

SAS 9

PROC TEMPLATE Tables Tip Sheet

This tip sheet places frequently used information in one place, on one sheet of paper, so you don't have to search through the online documentation. It also gives you something to take home, type in, and try.

The PROC TEMPLATE table family (e.g., table, column, header, and footer templates) allows you to build custom tables for your reports. Using table templates, you can create tables that display only the data you want to see, you can format the data using SAS® formats, you can add headers and footers to tables and columns, and you can use conditional formatting to enhance the readability of the data in your tables. This tip sheet presents the most common statements and attributes used in creating tables with PROC TEMPLATE.

SAS® 9 PROC TEMPLATE Tables Tip Sheet

Browsing Templates

The table templates supplied by SAS® are stored in the SASHELP.TMPLMST itemstore. You can browse the templates using either a graphical interface or PROC TEMPLATE code.

odstemplates
Type **odstemplates** into the command bar to open the Template Browser. Double-clicking a template displays its source code.

proc template; list *directory-name*; run;
Lists all templates in ***directory-name***.

proc template; source *template-name*; run;
Displays the source of ***template-name*** to the log.

ods trace on;
Displays information about tables as they are created. This information includes the name of the object, the name of the template, labels, paths, and label paths.

Using a Table Template

The following code applies a SAS data set to a PROC TEMPLATE table template.

```
data _null_;
  set data-set-name;
  file print ods=(
       template="table-name");
  put _ods_;
run;
```

data-set-name is the name of the data set to apply to the table template. *table-name* is the name of the table template to use.

Defining a Table Template

Basic Structure

```
proc template;
  define table table-name;
     ... statements / attributes / headers / footers / columns ...
  end;
run;
```

Commonly Used Statements

cellstyle *expr* as *style-element{style-attributes}*;
Conditionally applies styles to data cells.

column *column-specification(s)*;
Lists the names of the columns to appear in the table. Column names grouped in parentheses stack the data values on top of each other in a single data cell. Asterisks (*) stack groups of columns.

Each column rendered separately
```
column name age sex height weight;
```

Height stacked on weight
```
column name age sex (height weight);
```

Age stacked on height, sex stacked on weight
```
column name (age sex) * (height weight);
```

notes "*text*";
Specifies comments about the table that are stored with the template.

translate *expr-1* into *expr-2*;
Conditionally alters the values in the data cells.

Commonly Used Attributes

parent=*table-name*;
Specifies the parent table template for inheritance.

style=*style-element{style-attributes}*;
Applies specified styles to the table.

Defining Header and Footer Templates

Basic Structure

```
proc template;
  define header header-name;
     ... statements / attributes ...
  end;
  define footer footer-name;
     ... statements / attributes ...
  end;
run;
```

Commonly Used Statements

notes "*text*";
Specifies comments about the header or footer that are stored with the template.

text "*text*";
Specifies the text to be used as the content of the header or footer.

Commonly Used Attributes

just=left | center | right;
vjust=top | middle | bottom;
Specifies horizontal and vertical text alignment, respectively.

parent=*header-name*;
Specifies the parent header or footer template for inheritance.

split="*character*";
Specifies a character that should be rendered as a line break in the output.

start=*column-name*;
end=*column-name*;
Specifies the first and final columns, respectively, in a multi-column header or footer.

style=*style-element{style-attribute}*;
Applies specified styles to the header or footer.

Defining Column Templates

Basic Structure

```
proc template;
  define column column-name;
     ... statements / attributes / header / footer ...
  end;
run;
```

Commonly Used Statements

cellstyle *expr* as *style-element{style-attributes}*;
Conditionally applies styles to data cells.

compute as *expr*;
Computes values for data cells instead of using a column from a SAS data set.

notes "*text*";
Specifies comments about the column that are stored with the template.

translate *expr-1* into *expr-2*;
Conditionally alters the values in the data cells.

Commonly Used Attributes

format=*SAS-format*;
Specifies a SAS format for the data.

header="*text*"; or **header=*header-name*;**
Specifies the text for the column header, or the name of the header template to use for the header.

just=left | center | right
vjust=top | middle | bottom
Specifies horizontal and vertical text alignment, respectively.

parent=*column-name*;
Specifies the parent column template for inheritance.

style=*style-element{style-attributes}*;
Applies specified styles to the data cells.