%ANYTL: A Versatile Table/Listing Macro
Yang Chen, Forest Research Institute, Jersey City, NJ

ABSTRACT
Unlike traditional table macros, %ANYTL has only 3 macro parameters which correspond to the row, column and title/footnote of a table respectively) and can produce most typical tables/listings. It uses simple language, and grammar similar to PROC TABULATE which defines a table/listing with variable names and a few special characters.

%ANYTL is very flexible which allows one to manipulate the summary result prior to outputting to a table. The big N, column width, pagination and wrapping of the table can be either be automatically calculated or specified. This macro has proven to be powerful while succinct and easy to use.

1. INTRODUCTION
In the pharmaceutical industry, numerous macros have been developed to simplify the process of tables/listings production. Such macros typically have many parameters for specifying the content and layout of the table since there are many factors to consider. In most cases, a series of macros are often needed to handle different tables and data. This dual complexity makes it very difficult for programmers to use most of the macros that are currently available. One has to frequently refer to manuals to ensure parameter names are typed in correctly.

In contrast to such traditional macros, %ANYTL uses only three parameters to describe the three dimensions of a table, thus eliminating the need to remember any parameter names. Instead, it dissects the parameters to get the descriptive information. Its simple grammar uses special characters such as '*, .', and '(', similar to that used in PROC TABULATE in SAS®. In order to understand the logic of %ANYTL, let’s revisit the structure and content of a table/listing and the analysis data type.

1.1 THREE DIMENSIONS OF A TABLE/LISTING
No matter how complex a table/listing is, it can always be defined by 3 dimensions (row, column and table number).

In the example Table 1, the row definitions of the demographic table are Age and Sex; the column definition is Treatment; and the table number definition is Table 1 with associated titles and footnotes.

<table>
<thead>
<tr>
<th>Demographic Parameter</th>
<th>Treatment A (N=XXX)</th>
<th>Treatment B (N=XXX)</th>
<th>Total (N=XXX)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age (years)</td>
<td>xx.x</td>
<td>xx.x</td>
<td>xx.x</td>
</tr>
<tr>
<td>SD</td>
<td>x.xx</td>
<td>x.xx</td>
<td>x.xx</td>
</tr>
<tr>
<td>Median</td>
<td>xx, xx</td>
<td>xx, xx</td>
<td>xx, xx</td>
</tr>
<tr>
<td>Min, Max</td>
<td>xx, xx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>n</td>
<td>xxx</td>
<td>xxx</td>
<td></td>
</tr>
</tbody>
</table>

Sex, n (%)
| Male                  | xx (xx.x)           | xx (xx.x)           | xx (xx.x)    |
| Female                | xx (xx.x)           | xx (xx.x)           | xx (xx.x)    |

Table 1

Based on this important fact, the %ANYTL macro is designed to have 3 corresponding parameters:
ROW_DEF provides the source dataset names, analysis variable names and type, class variable names if any, and options such as display text, select range, statistics, etc.

COL_DEF provides the class variable names corresponding to the column/subcolumns of the statistical output. Usually it will be the treatment variable, with or without subgroup class variable(s).

TAB_DEF provides the output table number so that %ANYTL can find the corresponding titles and footnotes in the prepared title/footnote dataset and output the summary result to a table file.

1.2 THREE TYPES OF DATA

Another fact is that a table/listing contains 3 types of data: continuous data, categorical data and text. A few special characters are used in %ANYTL to distinguish them.

- CONTINUOUS DATA

By default, a variable specified in ROW_DEF of %ANYTL is considered a continuous data. For example, to get the continuous output for the Age section in Table 1, call the following:

%ANYTL(MYLIB.PROF.AGE, ..., ...)

This will tell %ANYTL to calculate N, SD, Mean, Min, Max and Median of the continuous variable AGE from PROF dataset from MYLIB library (library name optional). '.' is the separator between the dataset name and variable name.

- CATEGORICAL DATA

Categorical data is always summarized as counts with or without the corresponding percentages for each group. '/' is used in %ANYTL to separate the numerator and denominator variable names, and thus indicate categorical data.

To produce the Sex section in Table 1, call the following:

%ANYTL(PROF.SEX/PROF.SAFETY(=1), ..., ...)

This above will calculate the count and percentage of records in each category of the categorical variable SEX from the PROF dataset divided by number of records where SAFETY=1 from PROF dataset. Note that the denominator variable can come from a different source dataset. Since the denominator variable comes from the same dataset as the numerator variable, we can simplify things further by doing the following:

%ANYTL(PROF.SEX/SAFETY(=1), ..., ...)

The display of the categorical summary is conveniently controlled by the number of '/' used.

- VAR/1 -> X
- VAR1/VAR2 -> X (%)
- VAR1//VAR2 -> X/XXX
- VAR1///VAR2 -> X/XXX (%)

By default, the count will be number of records in each category. In some cases, unique patients instead of records should be counted. Under such situation, you can simply add '#' to at the end of the statement.

%ANYTL(PROF.SEX/SAFETY(=1)#, ..., ...)

By default %ANYTL will count the number of records for all of the categories of a variable which is determined by its associated format. Therefore, even if the count of a category is 0, you can specify the range that need to be displayed simply by using the ‘=’ sign in parentheses. For example,

%ANYTL(PROF.SEX(=1=2)/SAFETY(=1)#, ..., ...)
This means that male and female (WHERE SEX in (1, 2)) will be displayed.

• TEXT

Sometimes, text that is not based on the data is also displayed either as header for a summary or simply an individual line in the table. To display this, simply use quotation marks or double quotation marks.

For example,

%ANYTL(PROF.AGE'Age (years)', ..., ...) adds the title of ‘Age (years)’ to the summary of AGE.

%ANYTL(’Text 1’ “Text 2 with ’ sign”, ..., ...) will produce the last two lines in Table 1.

2. APPLICATION OF %ANYTL FOR TABLES

2.1. QUICK START: THREE BASIC TABLES

This section will demonstrate how %ANYTL can produce three tables that are commonly used in the pharmaceutical industry quickly and easily.

• DEMOGRAPHIC TABLE

Table 1 is a typical demographic table, which contains a mixture of continuous and categorical data without class variables. This can be produced with the following macro call:

```
%ANYTL(PROF.AGE 'Age (years)',
    PROF.SEX 'Sex, n(%)'/PROF.SAFETY(=1)
    'Text 1'
    “Text 2 with ’ sign”,
    TREAT+TOTAL,
    Table 1 'Demographic Parameter');
```

The 1st parameter (ROW_DEF) tells %ANYTL to calculate the statistics of the continuous variable AGE from the PROF dataset and then count each of the categories of the categorical variable SEX (from PROF dataset) as well as calculate its percentages based on the safety population (SAFETY=1 in PROF dataset). In addition, the macro adds two lines of text. Each individual item is separated by space(s).

Note that each analysis variable can come from different source datasets. In this example, since they all come from the same dataset PROF, the dataset name then can be omitted after its first appearance. So this macro call can be simplified to:

```
%ANYTL(PROF.AGE 'Age (years)'
    PROF.SEX 'Sex, n(%)'/PROF.SAFETY(=1)
    'Text 1'
    “Text 2 with ’ sign”,
    TREAT+TOTAL,
    Table 1 'Demographic Parameter');
```

The 2nd parameter (COL_DEF) tells %ANYTL to calculate the summaries by variable TREAT. ’+TOTAL’ is a reserved
term and produces an additional total column.

The 3rd parameter (TAB_DEF) outputs the statistical summary to a table. By specifying the table number, %ANYTL will find and display the corresponding titles and footnotes stored in a title_footnote text file. The corresponding N=xxx under each treatment column will be automatically determined by reading the last line of title. In our example, ‘Safety Population’ is the last line, and %ANYTL will find count of patients WHERE SAFETY=1 in the PROF dataset. If there is some text ‘Demographic Parameter’ in the first column header in our example - type it in a pair of quotation marks after the table number. If this is not done, the first column header will be blank.

%ANYTL will calculate the width of each column automatically if not specified (to specify the width, refer to section 2.2.6).

- **HIERARCHICAL TABLE**

A hierarchical table is another common table type, in which the summary statistics are calculated by some class variables. In the example below (Table 2), the statistics of the continuous variable VALUE from the LAB dataset is calculated by Clinical Lab Parameters (TEST) and then by Visit (VISIT).

<table>
<thead>
<tr>
<th>Laboratory Parameter</th>
<th>Treatment A (N=xx)</th>
<th>Treatment B (N=xx)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hemoglobin (g/dL)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>xx.x</td>
<td>xx.x</td>
</tr>
<tr>
<td>SD</td>
<td>x.xx</td>
<td>x.xx</td>
</tr>
<tr>
<td>Median</td>
<td>xx.xx</td>
<td>xx.xx</td>
</tr>
<tr>
<td>Min, Max</td>
<td>xx, xx</td>
<td>xx, xx</td>
</tr>
<tr>
<td>n</td>
<td>xxx</td>
<td>xxx</td>
</tr>
<tr>
<td>Visit 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>xx.x</td>
<td>xx.x</td>
</tr>
<tr>
<td>SD</td>
<td>x.xx</td>
<td>x.xx</td>
</tr>
<tr>
<td>Median</td>
<td>xx.xx</td>
<td>xx.xx</td>
</tr>
<tr>
<td>Min, Max</td>
<td>xx, xx</td>
<td>xx, xx</td>
</tr>
<tr>
<td>n</td>
<td>xxx</td>
<td>xxx</td>
</tr>
</tbody>
</table>

...  

Similar to PROC TABULATE, ‘*’ is used to indicate the class variable TEST and VISIT before the analysis variable. The call will be:

```
%ANYTL (LAB.TEST*VISIT*VALUE,
    TREAT,
    Table 2 ‘Laboratory Parameter~ Visit’);
```

Don’t forget that if needed, you can always select the display range by adding (=...) option after the corresponding variable name. For example, to display only Hemoglobin test at Baseline and Visit 1, call:

```
%ANYTL (LAB.TEST(=‘Hemoglobin (g/dL)’) *VISIT(=0=1) *VALUE,
    TREAT,
    Table 2 ‘Laboratory Parameter~ Visit’);
```

A hierarchical categorical table can be handled in the same fashion. For example, the nominator, denominator and percentage of Table 3 are all calculated by VISIT.
Table 3
Abnormal Physical Examination
Safety Population

<table>
<thead>
<tr>
<th>Visit</th>
<th>Body System</th>
<th>Treatment A (N=XXX)</th>
<th>Treatment B (N=XXX)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>n/N1 (%)</td>
<td>n/N1 (%)</td>
</tr>
<tr>
<td>Baseline</td>
<td>Cardiovascular</td>
<td>x/xxx (x.xx)</td>
<td>x/xxx (x.xx)</td>
</tr>
<tr>
<td></td>
<td>Respiratory</td>
<td>x/xxx (x.xx)</td>
<td>x/xxx (x.xx)</td>
</tr>
<tr>
<td></td>
<td>Gastrointestinal</td>
<td>x/xxx (x.xx)</td>
<td>x/xxx (x.xx)</td>
</tr>
<tr>
<td></td>
<td>Skin</td>
<td>x/xxx (x.xx)</td>
<td>x/xxx (x.xx)</td>
</tr>
<tr>
<td>Visit 1</td>
<td></td>
<td>x/xxx (x.xx)</td>
<td>x/xxx (x.xx)</td>
</tr>
</tbody>
</table>

To do this, simply put the class variable VISIT* in both numerator part and denominator portion:

\[
\%\text{ANYTL(PE.VISIT*PETERM///VISIT*SAFETY(=1)#, TREAT, Table 3 ‘Visit Body System’);}
\]

- **ADVERSE EVENT (AE) TABLE**

An AE table is a special type of hierarchical table (see Table 4) in that:

1. The percentages will be calculated for every term level.

2. Under each treatment column, there could be sub-columns for relationship or severity. If one subject has multiple degrees of relationship (or severity), only the most related (or severe) AE incidence will be used in the calculation.

<table>
<thead>
<tr>
<th>System Organ Class/Preferred Term</th>
<th>Treatment A (N=XXX)</th>
<th>Treatment B (N=XXX)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unrelated n (%)</td>
<td>Related n (%)</td>
</tr>
<tr>
<td>System Organ Class 1 [bodsys]</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
</tr>
<tr>
<td>Preferred Term 1 [prefterm]</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
</tr>
<tr>
<td>Preferred Term 2</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
</tr>
<tr>
<td>System Organ Class 2</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
</tr>
</tbody>
</table>

In \%\text{ANYTL, ,}**’ is used to distinguish AE tables from regular hierarchical tables. Here, the call will be:

\[
\%\text{ANYTL(TEAE(WHERE=(SAFETY=1 and TEAE=1)).BODSYS**PRETERM/PROF.SAFETY(=1)#, TREAT**RELATED, Table 4 ‘System Organ Class/Preferred Term’);}
\]

The ** in ‘BODSYS**PRETERM’ in the 1st parameter tells \%\text{ANYTL} to produce the TEAE count by both system organ class (BODSYS) and Preferred Term (PRETERM) for the safety population. ‘/SAFETY(=1)’ tells \%\text{ANYTL} that the denominator of the percentage is number of subjects in the safety population from PROF dataset.

\text{TREAT**RELATED} in the 2nd parameter indicates that there are subcolumns for relationship under each treatment column, and only most related incidence will be counted if there are multiple incidences of a AE for a subject. If there are multiple levels of subcolumns, express it in the same way. For example, \text{PROTOCOL*TREAT**RELATED} will produce three level subcolumns.
2.2 ADVANCED TABLE APPLICATION

Most tables, including the three basic table types, can be produced using the methods described above. In this section, some advanced features of %ANYTL will be introduced.

2.2.1 MODIFY THE RESULT BEFORE OUTPUTTING TO A TABLE

A very unique feature of %ANYTL is that it allows one to modify the summary dataset before outputting it to a table which will accommodate non-standard requests. To do this, simply leave the 3rd parameter (TAB_DEF) empty and %ANYTL will output a summary dataset named _ANYSUM instead of outputting a table. After the DATA STEP manipulation, call %ANYTL again, and fill in the 3rd parameter - a table will be output.

For example, in Table 1, when the percentage is 100.0%, the default display format of %ANYTL is (100). %ANYTL does not have an option to display it as (100.0). Alternatively, it can be easily achieved using a datastep after implementing the macro. The steps to demonstrate this modification are shown below:

First produce the output dataset (Note that the 3rd parameter is now blank):

```
%ANYTL(PROF.AGE`Age (years)` SEX`Sex, n(%)’/SAFETY(=1), TREAT+TOTAL, );
```

A dataset named _ANYSUM will be produced, which looks like:

```
_NAME_   _1_   _2_   _3_
Age (years) 41.3 38.7 40.2
SD 6.7 4.3 5.2
Median 38.8 42.5 41.1
Min, Max 35, 52 36, 55 35, 55
n 50 50 100
Sex, n (%) Male 50 (100) 10 (20.0) 60 (60.0)
Female 0 40 (80.0) 40 (40.0)
```

Each variable corresponds to a column that will be displayed in the table. So you can use DATA STEP to modify the dataset _ANYSUM to get the desired result:

```
DATA _ANYSUM;
SET _ANYSUM;
_1=TRANWRD(_1, ’(100)’, ’(100.0)’);
RUN;
```

Then call %ANYTL again (this time leave the fist 2 parameters blank):

```
%ANYTL( , , Table 1 ‘Demographic Parameter’);
```

Table 1 will be output, now using the modified result:

```
Table 1  
Demographic Characteristics
Safety Population

<table>
<thead>
<tr>
<th>Demographic Parameter</th>
<th>Treatment A (N=XXX)</th>
<th>Treatment B (N=XXX)</th>
<th>Total (N=XXX)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age (years)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>41.3</td>
<td>38.7</td>
<td>40.2</td>
</tr>
<tr>
<td>SD</td>
<td>6.7</td>
<td>4.3</td>
<td>5.2</td>
</tr>
<tr>
<td>Median</td>
<td>38.8</td>
<td>42.5</td>
<td>41.1</td>
</tr>
<tr>
<td>Min, Max</td>
<td>35, 52</td>
<td>36, 55</td>
<td>35, 55</td>
</tr>
<tr>
<td>n</td>
<td>50</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>Sex, n (%)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Male</td>
<td>50 (100.0)</td>
<td>10 (20.0)</td>
<td>60 (60.0)</td>
</tr>
<tr>
<td>Female</td>
<td>0</td>
<td>40 (80.0)</td>
<td>40 (40.0)</td>
</tr>
</tbody>
</table>
```
2.2.2 STATISTICAL MODEL

Statistics like P-value, CI, etc. are often needed. %ANYTL has a statistical model library for common statistical methods. Simply add the corresponding statistical model name after the response variable, separated by ‘+’. For example, to produce Table 1b, call %ANYTL as:

```
%ANYTL(PROF.AGE’Age (years)’+wilcoxon
   SEX’Sex, n(%)’/SAFETY(=1)+fisher,
   TREAT+TOTAL,
   Table 1b ‘Demographic Parameter’);
```

Then p-value of Wilcoxon Test for AGE and the Fisher’s Exact Test for SEX will be added.

Table 1b
Demographic Characteristics
Safety Population

<table>
<thead>
<tr>
<th>Demographic Parameter</th>
<th>Treatment A (N=XXX)</th>
<th>Treatment B (N=XXX)</th>
<th>Total (N=XXX)</th>
<th>P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age (years)</td>
<td>Mean xx.x, SD x.xx, Median xx., Min, Max xx, xx, n xxx</td>
<td>Mean xx.x, SD x.xx, Median xx., Min, Max xx, xx, n xxx</td>
<td>Mean xx.x, SD x.xx, Median xx., Min, Max xx, xx, n xxx</td>
<td>0.xxx</td>
</tr>
<tr>
<td>Sex, n (%)</td>
<td>Male xx (xx.x), Female xx (xx.x)</td>
<td>Male xx (xx.x), Female xx (xx.x)</td>
<td>Male xx (xx.x), Female xx (xx.x)</td>
<td>0.xxx</td>
</tr>
</tbody>
</table>

2.2.3 PARALLEL SUMMARIES

Table 5
Vital Signs
Safety Population

<table>
<thead>
<tr>
<th>Visit Statistics</th>
<th>Treatment A (N=XXX)</th>
<th>Treatment B (N=XXX)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>Actual Change</td>
<td>Baseline Actual Change</td>
</tr>
<tr>
<td>Day 1</td>
<td>Mean xx.x, SD x.xx, Median xx, Min, Max xx, xx, n xxx</td>
<td>Mean xx.x, SD x.xx, Median xx, Min, Max xx, xx, n xxx</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In Table 5, the subcolumns Baseline, Actual and Change are based on 3 variables: BSLVAL, VALUE and CHANGE respectively. %ANYTL uses '|' to separate them. The macro call is:

```
%ANYTL(VITAL.VISIT*BSLVAL|VALUE|CHANGE,
       TREAT,
       Table 5 ‘Visit Statistics’)
```

Note that BSLVAL, VALUE and CHANGE are the analysis variables, not the class variables. Therefore they are placed in the 1st parameter (ROW_DEF), not the 2nd parameter (COL_DEF).
2.2.4 GROUP VARIABLES TOGETHER

### Table 1c
Demographic Characteristics
Safety Population

<table>
<thead>
<tr>
<th>Demographic Parameter</th>
<th>Treatment A (N=XXX)</th>
<th>Treatment B (N=XXX)</th>
<th>Total (N=XXX)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Race, n (%)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>White</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
</tr>
<tr>
<td>American Indian or Alaska Native</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
</tr>
<tr>
<td>Asian</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
</tr>
<tr>
<td>Black or African American</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
</tr>
<tr>
<td>Pacific Islander</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
</tr>
</tbody>
</table>

Sometimes the results for several variables need to be grouped together. In Table 1c, the categories of race actually come from different variables (WHITE, INDIAN, ASIAN, BLACK and PACIF, value=1 or missing). In %ANYTL a parenthesis can be used and the results will be grouped together:

```plaintext
%ANYTL(PROF.('Race, n (%)'
  WHITE' White'/SAFETY(=1)
  INDIAN' American Indian or Alaska Native'/SAFETY(=1)
  ASIAN' Asian'/SAFETY(=1)
  BLACK' Black or African American'/SAFETY(=1)
  PACIF' Pacific islander'/SAFETY(=1)),
TREAT,
Table 1c 'Demographic Parameter');
```

Since the denominators for these variables are identical, the call can be further simplified:

```plaintext
%ANYTL(PROF.('Race, n (%)'
  WHITE' White'
  INDIAN' American Indian or Alaska Native'
  ASIAN' Asian'
  BLACK' Black or African American'
  PACIF' Pacific islander')/SAFETY(=1),
TREAT,
Table 1c 'Demographic Parameter');
```

2.2.5 MORE THAN ONE TEXT COLUMNS

### Table 6
Response by Region and Country
Safety Population

<table>
<thead>
<tr>
<th>Region/Country</th>
<th>Response</th>
<th>Treatment A n (%)</th>
<th>Treatment B n (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eastern Europe</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Romania</td>
<td>N</td>
<td>XXX</td>
<td>XXX</td>
</tr>
<tr>
<td></td>
<td>Cure</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
</tr>
<tr>
<td></td>
<td>Failure</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
</tr>
<tr>
<td></td>
<td>Indeterminate</td>
<td>xx (xx.x)</td>
<td>xx (xx.x)</td>
</tr>
<tr>
<td>Russia</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In hierarchical Table 6, there are 2 header columns. The first column are the class variables (REGION and COUNTRY) and the second column is for the n and the response categories under each country. To create 2 columns, insert `-' after the class variable(s) in the first column. Parentheses are used to group the statistics by region and country. Correspondingly, the header texts ('Region/Country', 'Response') for the two columns are separated by `-' in the 3rd parameter. Call below:

```plaintext
%ANYTL(CLIN_REGION*COUNTRY-* (SAFETY 'N' (=1)/1)
```
2.2.6 SPECIFY THE COLUMN WIDTH

%ANYTL will automatically calculate the width of each column in the table and flow the text if needed. However, you can choose to specify the width of the header columns after the corresponding column header text in the 3rd parameter. For example, to limit the width of the first column to be 20 characters and the second column to be 25 in Table 6, call the following:

%ANYTL(CLIN.REGION*COUNTRY-*(SAFETY'N' (=1)/1
    RESP/REGION*COUNTRY*SAFETY(=1)#),
    TREAT,
    Table 6 ‘Region/Country’-‘Response’/20-’Response’/25);

2.2.7 SORTING AE TABLE

You can specify how each AE term level should be sorted in after each term level name. There are 4 options:

- AC: ascending alphabetically, can be omitted since it is the default
- DC: descending alphabetically
- AN: ascending by count(s) of specified column(s)
- DN: descending by count(s) of specified column(s)

For example, you can create a AE table, in which every AE term is sorted differently.

%ANYTL( TEAE.TERM1[dc]**TERM2[an(=2)]**TERM3[dn(=2)**(=2=3)]/PROF.SAFETY(=1)#,
    TREAT**SEVERITY,
    Table xx);

TERM1[dc] means TERM1 will be sorted descending alphabetically.

TERM2[an(=2)] means TERM2 will be sorted by the sum of the counts of the columns under TERM =2.

TERM3[dn(=2)**(=2=3)] means TERM3 will be sorted by the sum of the counts of the columns of TERM =2 and SEVERITY in (2, 3).

2.2.8 MULTIPLE PAGES

If there are too many columns to fit onto one page, they will have to be displayed in multiple pages. In %ANYTL, you can choose either to specify the maximum column number allowed on each page or provide the page assignment group variable in the 2nd parameter.

For example, %ANYTL(..., TREAT/4, ...) will put only 4 summary columns on each page.

%ANYTL(..., TREAT/PAGEGRP, ...) will put the summary columns on a different page defined by the group variable PAGEGRP.
3 OUTPUT A LISTING QUICKLY

%ANYTL is also capable of producing a simple listing quickly. A listing is regarded as a table defined solely by columns and table number in %ANYTL.

Listing 1
Subjects with at least one Serious AE
Safety Population

<table>
<thead>
<tr>
<th>Treatment Group</th>
<th>Subject ID</th>
<th>Age/Sex/Race</th>
<th>Date of first/last dose</th>
<th>Preferred Term/Investigator Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>Treatment A</td>
<td>001</td>
<td>25/Male/Asian</td>
<td>2001-01-01/2001-07-02</td>
<td>Pref. Term 1/Inv. Term2</td>
</tr>
<tr>
<td></td>
<td>002</td>
<td>35/Female/Caucasion</td>
<td>2001-02-02/2001-08-02</td>
<td>Pref. Term 1/Inv. Term2</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Treatment B</td>
<td>007</td>
<td>55/Female/Asian</td>
<td>2001-03-03/2001-09-02</td>
<td>Pref. Term 1/Inv. Term2</td>
</tr>
<tr>
<td></td>
<td>008</td>
<td>25/Male/Asian</td>
<td>2001-04-04/2001-10-02</td>
<td>Pref. Term 1/Inv. Term2</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

To produce the example Listing 1, call the following:

```
%ANYTL(

AE(WHERE=(SAE=1)).('Demographic Information'
TREAT'Treatment Group'
PID'Subject ID'
(AGED/SEX/~RACEC)'Age/Sex/Race'
(FDSDT/~LDSDT)'Date of First/Last Dose of Study Drug'
(PT91/~AE)'Preferred Term/Investigator Term'/20or,

Listing 1);
```

The 1st parameter is left blank to indicate this is a listing.

In the 2nd parameter, dataset name (AE) and column variable names are separated with \'.\'.

Each column is described as VARNAME'header text'/options and separated with each other by spaces.

If there is a column header that spans multiple columns (\'Demographic Information\' in the example), put the text (in quotation marks) before the column items and use parentheses as COLUMN statement in PROC REPORT.

If a column is concatenated by several variables, simply put them within a set of parenthesis. All the special characters between the variables will be kept. For example, the 3rd column \'(AGED/SEX/~RACEC)\' is equivalent to:

```
var3=PUT(AGED, best.)||'/'||PUT(SEX, sexf.)||'/~'||RACEC;
```

If the variables need to be ordered but not displayed, put '/np' option after the variable name (=NOPRINT option). If the variables need to be ordered and grouped, put '/or' option after the variable name of the LAST group variable. If the width of a column needs to be fixed, put the width after it. Otherwise, it will be automatically determined.

For example, the 5th column \'(PT91/ AE)\'Preferred Term/Investigator Term'/20or is equivalent to:

```
Var5=STRIP(PT91)||'/'||STRIP(AE); ...

DEFINE VAR5/'Preferred Term/Investigator Term' width=20 order order=internal;
```
CONCLUSION

Only a portion of the usage details is presented here. For example, %ANYTL can also be a useful tool for validating tables (refer to my 'Three Step Validation', Pharmasug, 2007). But this paper does not intend to serve as a user manual since %ANYTL is designed to fit my company table template and may not be directly applicable for other companies. However, the concept of the three dimensions of Tables/listings and using special characters to indicate each component to describe the table structure can be universally applied. As long as your Table/Listings follow a relatively standard template, you can create similar macro.

Admittedly it was a challenge to write this macro compared with traditional macros since it must use basic SAS® macro functions such as %SCAN, %LENGTH and DATA _NULL_ etc to dissect the parameters before actually analyzing the source datasets. In addition, there were numerous issues to consider since lots of information is not explicitly provided. Type and format of variables, quoted special characters, alignment, missing data handling, order of each item etc added to the complexity. Nonetheless, the advantages of %ANYTL is apparent to all programmers in the pharmaceutical industry:

- %ANYTL is a single macro, not a series of macros and the grammar is consistent, logical and easy to follow. You will never have to memorize numerous parameter names any more.
- Instead of being a black box, %ANYTL allows you to predict the Table output even just by reading the code. Thus it is comprehensible for any %ANYTL user.
- Code in %ANYTL is very succinct and minimizes the task of pre-handling data before table production.
- It is very versatile. You can specify different source datasets, different denominators, selection ranges, sorting orders and display styles for each item in the table. The result can also be modified before outputting to a Table. These tasks are usually awkward to accomplish in traditional macros. Therefore, this single macro can handle most table/listing tasks that, in the past, were performed by several complicated macros.
- It is expandable. Without a doubt, there will be new studies and new requests that the current version cannot handle. %ANYTL is built is such a manner that I can always add more symbols or symbol combinations, functions and statistics to meet the need of the evolving industry.

REFERENCES

Three Step Table Validation, Yang Chen, Pharmasug, 2007

CONTACT INFORMATION

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

Name: Yang Chen
Enterprise: Forest Research Institute
Address: Forest Lab, Financial Plaza V
City, State ZIP: Jersey City, NJ 07311
Work Phone: (201) 253-6625
E-mail: snail_chen@hotmail.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. © indicates USA registration.

Other brand and product names are trademarks of their respective companies.