

Descriptive Summary Table Made Easy

Murshed Siddick & Rafi Rahi, Shafi Consultancy Ltd., United Kingdom

1. Introduction

Descriptive summary tables are the bread and butter of the clinical trial reporting world. It is one of the most repeated features in any reporting process. However, the amount of time spent to produce these tables is still extraordinary.

- We present a simple solution whereby a programmer can call a set of small, easy to understand macros in successive order to produce the descriptive summary tables within minutes.
- This not only reduces the time spent in writing individual codes for producing summary tables but also makes it easy to maintain and validate them over time.

2. Project Goal

- Divide the complete table production process into a number of easy to follow, independent steps.
- Develop macros for each of these steps in a way that they work independently of each other (i.e. only reads the output files produced by the preceding macros) and are easy to understand and adapt.
- Finally, present a coding structure which produces the summary tables from scratch by calling these macros in successive order.

3. Current Practice

Often multiple programmers are involved in producing the codes for tables in the same clinical trial. Each member either produces their own code or adjusts their code from a previous study. Once the codes are produced, they are checked and validated by resolving the issues found before submitting the final tables to the client.

4. Drawback of Current Practice

- Individual programmers spend a lot of time writing the table codes, testing and validating them before producing the tables.
- As many programmers re-produce similar programs in different ways, they are time-consuming to validate and maintain over time.

5. Code Overview

The production of a table is divided into five key steps (see diagram 1):

- Collate data into one dataset
- Calculate top row and big N
- Calculate summaries
- Combine summaries
- Produce table output

All the datasets produced during the macro executions for creating a particular table are marked with a single prefix name in order to identify them easily.

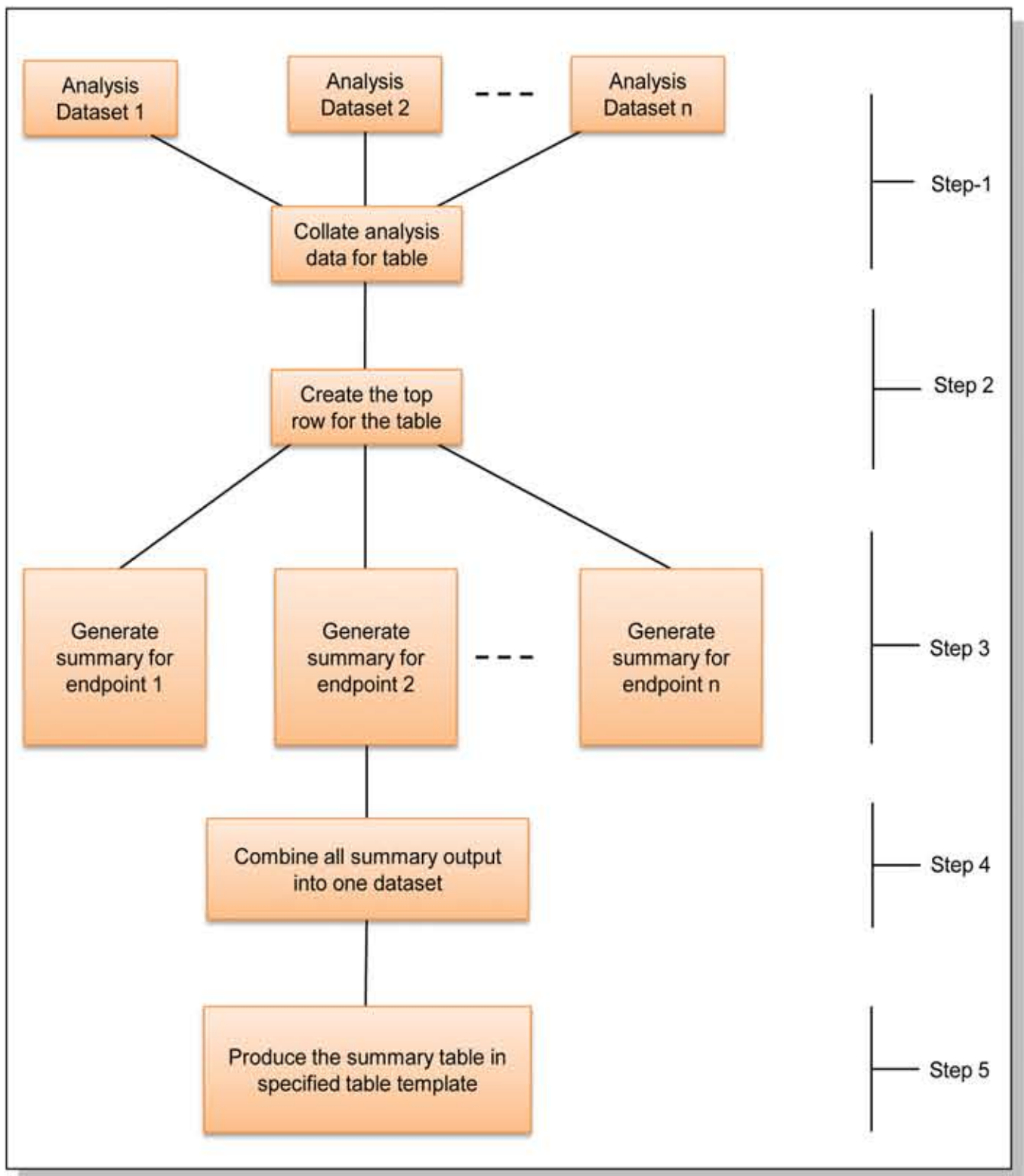


DIAGRAM 1

6. An Example

A typical macro code is displayed in diagram 2, where the macro collects the baseline values of variables 'age', 'race', 'sex' and other relevant information from analysis datasets and collates them into a single dataset 'demog_alldata'.

The final output will then contain the descriptive summary for each of these demography variables (see diagram 3).

The prefix name 'demog_' is provided for this table. The macro code thus creates all the datasets with the 'demog_' prefix, which makes it easy to identify the datasets for the demography table.

7. Components of Macro code

get_data()

Collates all the endpoint data in a specific data structure. We have chosen a longitudinal data structure as this is the easiest and any data stored in any different structure can be easily transformed into this.

```
TITLE "The Demographic Data Summary";

%LET prefix=demog_;

%get_data(inpopu = popu,      popu   = TREATED,
          intrt  = trt,      inbase  = base,
          outdata = &prefix.alldata);

%count_summary(indata = &prefix.alldata,
               v_sort  = 0,
               v_label = "Total treated",
               outdata = &prefix.x_toprow,
               outbign = &prefix.bign);

%stat_summary(indata = &prefix.alldata,
              v_val   = age,
              v_label = "Age[years]",
              v_sort  = 1,
              outdata = &prefix.x_age);

%count_summary(indata = &prefix.alldata,
               inbign = &prefix.bign,
               v_val   = sex,
               v_label = "Gender",
               v_valdc = sexdc,
               v_sort  = 2,
               outdata = &prefix.x_sex);

%count_summary(indata = &prefix.alldata,
               inbign = &prefix.bign,
               v_val   = race,
               v_label = "Race",
               v_valdc = racedc,
               v_sort  = 3,
               outdata = &prefix.x_race);

%combine(indata = &prefix.x_,
         outdata = &prefix.table);

%report1(indata = &prefix.table,
         pgbrk opt = 2,
         brk after = 1 2
         outdata = &prefix.2report);

TITLE;
```

DIAGRAM 2

stat_summary()

Calculates the descriptive summary statistics for continuous data e.g. age, weight etc.

count_summary()

Calculates the N and percent for categorical data e.g. race, smoking category etc.

We can call these two macros as many times as required to calculate summaries of more than one variable.

Top row and big N

In most cases, a descriptive summary table requires a top row containing the total count (number of patients in each treatment group).

The top row is basically the output of count_summary() macro without any end-point and time point classification specified. The total count is referred as 'big N' and is used in N and percent calculation for categorical data.

combine()

Combines all the summary outputs into one final/result dataset.

report()

Produces the descriptive summary table in specified layout and file format.

8. Output

Most of the descriptive summary tables follow a simple layout where the columns represent treatment groups and rows represent summary categories e.g. N, Percent, Mean, Median etc for age, height, weight etc.

All the summaries for a single variable e.g. age or sex are put together and labelled with an appropriate table header.

The Demographic Data Summary				
	TREATMENT GROUP			
	TRT_A	TRT_B	TRT_C	
Total treated	135 (100.0)	135 (100.0)	133 (100.0)	
Gender				
Male	77 (57.0)	70 (51.9)	65 (48.9)	
Female	58 (43.0)	65 (48.1)	68 (51.1)	
Age [years]				
N	135	135	133	
Mean	57.9	57.0	56.8	
SD	10.5	11.8	10.9	
Min	34.0	24.0	22.0	
Median	58.0	58.0	56.0	
Max	78.0	78.0	80.0	
Race				
Amer.Ind./Alaska Nat	0 (0.0)	2 (1.5)	1 (0.8)	
Asian	1 (0.7)	3 (2.2)	3 (2.3)	
Black/African Amer.	14 (10.4)	18 (13.3)	16 (12.0)	
Hawaiian/Pacif.Isle	0 (0.0)	1 (0.7)	0 (0.0)	
White	120 (88.9)	111 (82.2)	113 (85.0)	

DIAGRAM 3

Some variables such as some study end-points are recorded over multiple time points. In such case, the code allows the option to summarise variable values over multiple time points.

9. Adaptation

If the analysis datasets are arranged in a different data structure and the requested summary tables are in a different layout, then the structure of the macro in step one [get_data()] and in step four [combine()] can be easily updated and all the programs calling these macros will not have to be modified.

10. Conclusion

By identifying the key steps and developing macros for each of these steps, we have streamlined the table production process and resolved the time issues associated with them being programmed independently.

More importantly, the scheme allows central control over the work flow and makes it easy to maintain the codes and update the tables centrally whenever necessary (see diagram 4).

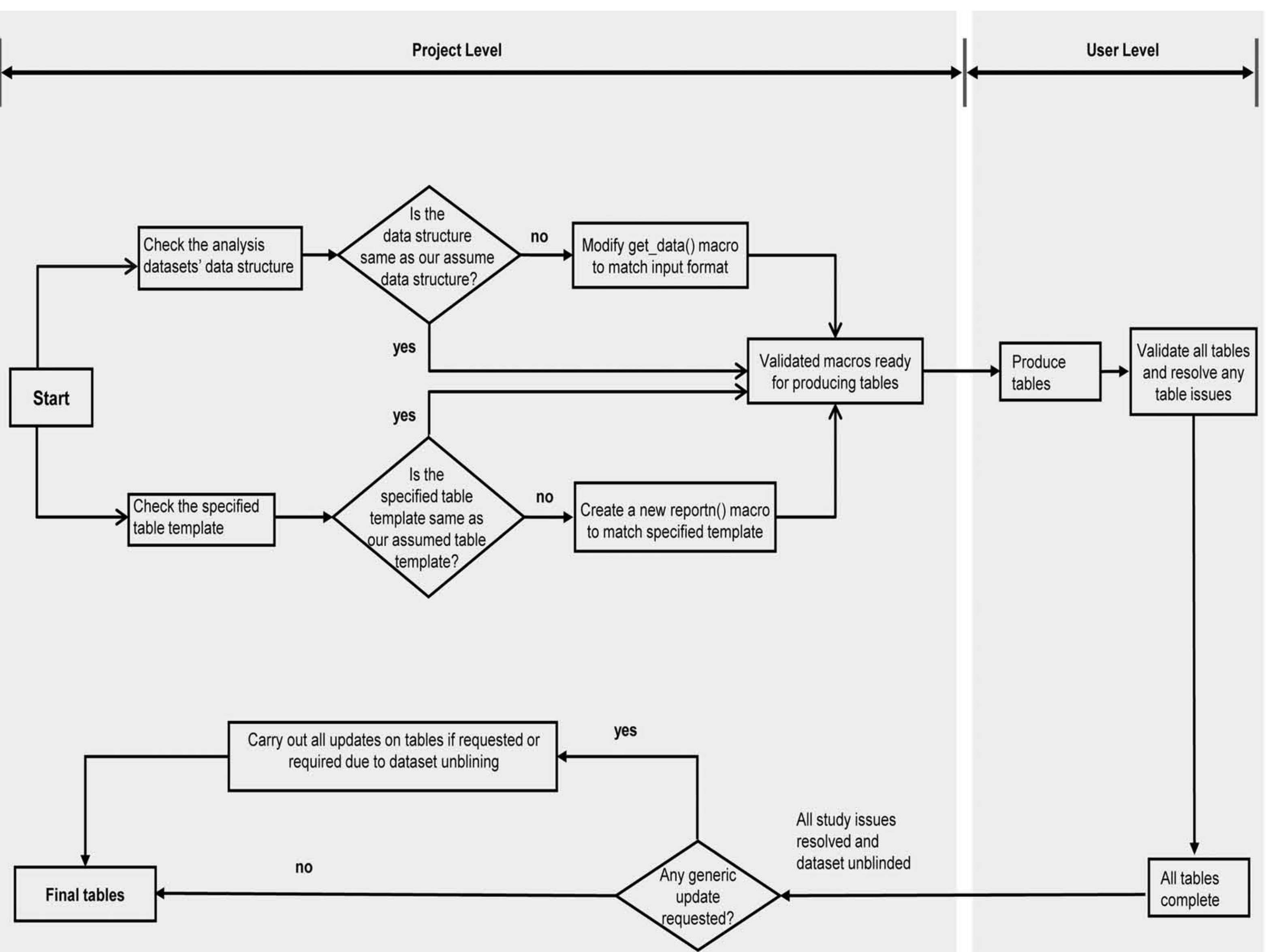


DIAGRAM 4

Please feel free to contact us for further information: www.ShafiConsultancy.com