



Tidytlg: An R Package for Clinical Reporting Using Tidyverse

Sheng-Wei Wang (Janssen R&D) & Eli Miller (Atorus Research)

Presented at PhUSE US Connect 2023

March 5-8, 2023

Sebastian Ferreira, *Untitled 1*

Artwork from the National Art Exhibitions
of the Mentally Ill, Inc (NAEMI).

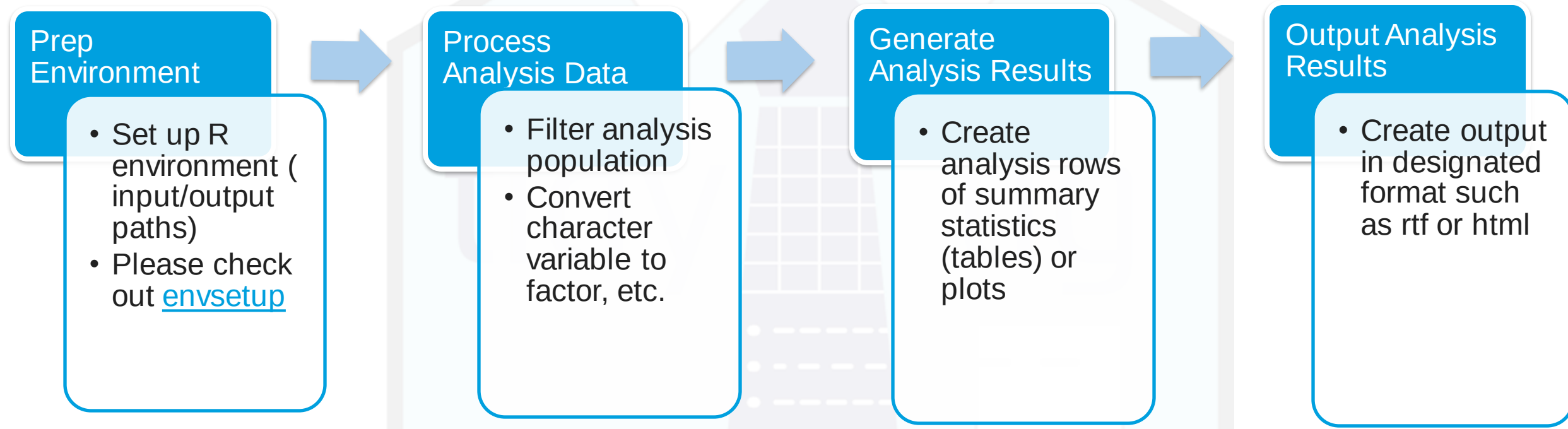
Outline

- Motivation: Why?
- TLG programming workflow
- Tidytlg core functions: `univar`, `freq`, `nested_freq`, `bind_table`, `gentlg`
- Example outputs walkthrough:
 - Demographic Table
 - Adverse Events Table
 - Laboratory Summary Table
 - Listing
 - Graph
- Metadata-driven table creation
- Summary

Motivation

- Open-source technologies have been rapidly growing in recent years, and more and more pharmaceutical companies are starting to use **R as an alternative to SAS** or additional software for clinical trial reporting
- **To ease the transition** from SAS to R for programmers, we developed the **tidytlg package for R** to create the tables, listings, and graphs (**TLG**) for clinical study reports
- Please check our public announcement on [Medium](#) / [LinkedIn](#) and package [documentation](#) for more details

TLG Programming Workflow



- This workflow can be implemented with tidytlg:
 - Functional method: build a custom script step by step for each TLG
 - Metadata method: create a generic script that utilizes column and table metadata for each TLG

Tidytlg Core Functions

- Generate analysis results of descriptive summary statistics
 - **univar**: generate univariate statistics for numeric variables
 - **freq**: generate counts and percentages for character variables
 - **nested_freq**: summarize counts and percentages in a nested structure (counts within counts) such as body system by preferred terms of Adverse Events data
- Output analysis results
 - **bind_table**: combine analysis results with formatting variables
 - **gentlg**: convert analysis results to rtf or html files

Key Tidytlg Analysis Function Arguments

- df: analysis dataframe containing records to summarize
- colvar: column variable within df to summarize (such as the treatment variable)
- rowvar: analysis variable within df for producing summary statistics
- statlist: object of statistics list to display (e.g., statlist(c("N", "MEANS", "MEDIAN", "RANGE", "IQRANGE")))
- row_header: text string to add as a header for the row summary
- rowtext: text string to display for a single row summary
- rowbyvar: variable within df to repeat the rowvar analysis
- tablebyvar: variable within df to repeat entire table analysis

Example 1: Demographic Table

```
library(dplyr, warn.conflicts = FALSE)
library(tidytlg)

# Note cdisc_adsl is built into the package for use
ittpop <- cdisc_adsl %>%
  filter(ITTFL == "Y")

# frequency of Intend-to-Treat patients by planned treatment
tbl1 <- freq(ittpop,
  rowvar = "ITTFL",
  statlist = statlist("n"),
  colvar = "TRT01P",
  rowtext = "Analysis Set: Intend-to-Treat Population",
  subset = ITTFL == "Y")

# N, MEAN (SD), MEDIAN, RANGE, IQ Range of age by planned treatment
tbl2 <- univar(ittpop,
  rowvar = "AGE",
  colvar = "TRT01P",
  row_header = "Age (Years)")

# frequency of gender by planned treatment
tbl3 <- freq(ittpop,
  rowvar = "SEX",
  statlist = statlist(c("N", "n (x.x%)")),
  colvar = "TRT01P",
  row_header = "Gender, n(%)")

# combine results together
tbl <- bind_table(tbl1, tbl2, tbl3)
```

- Analysis results are created step-by-step:
 - Call freq to create analysis population counts
 - Call univar to generate univariate statistics for age
 - Call freq to create gender counts and percentages
- Use bind_table function to combine analysis results with formatting variables (indentme, newrows, newpage)

	label	Placebo	Xanomeline High Dose	Xanomeline Low Dose	row_type	anbr	indentme	roworder	newrows	newpage
1	Analysis Set: Intend-to-Treat Population	5	5	5	HEADER	1	0	1	0	0
2	Age (Years)				HEADER	2	0	1	1	0
3	N	5	5	5	N	2	1	2	0	0
4	Mean (SD)	69.60 (14.398)	72.20 (9.230)	75.60 (6.731)	VALUE	2	2	3	0	0
5	Median	64.00	75.00	74.00	VALUE	2	2	4	0	0
6	Range	(52.0; 85.0)	(57.0; 81.0)	(68.0; 84.0)	VALUE	2	2	5	0	0
7	IQ range	(63.00; 84.00)	(71.00; 77.00)	(71.00; 81.00)	VALUE	2	2	6	0	0
8	Gender, n(%)				HEADER	3	0	1	1	0
9	N	5	5	5	N	3	1	2	0	0
10	F	2 (40.0%)	3 (60.0%)	1 (20.0%)	VALUE	3	2	3	0	0
11	M	3 (60.0%)	2 (40.0%)	4 (80.0%)	VALUE	3	2	4	0	0

Example 1: Demographic Table

- Create rtf output in the gentlg function call

```
# Create rtf output -----
gentlg(huxme      = tbl ,
       orientation = "portrait",
       file       = "TDEMO01",
       title      = "Summary of demographics",
       footers    = "Note: SD: standard deviation; IQ: interquartile",
       colspan     = list(c("", "", "Xanomeline", "Xanomeline")),
       colheader   = c("", "Placebo", "High", "Low"),
       wcol       = .30)
```

TDEMO01: Summary of demographics				
		Xanomeline		
		Placebo	High	Low
Analysis Set:	Intend-to-Treat Population	5	5	5
Age (Years)				
N		5	5	5
Mean (SD)		69.60 (14.398)	72.20 (9.230)	75.60 (6.731)
Median		64.00	75.00	74.00
Range		(52.0; 85.0)	(57.0; 81.0)	(68.0; 84.0)
IQ range		(63.00; 84.00)	(71.00; 77.00)	(71.00; 81.00)
Gender, n(%)				
N		5	5	5
F		2 (40.0%)	3 (60.0%)	1 (20.0%)
M		3 (60.0%)	2 (40.0%)	4 (80.0%)
Note: SD: standard deviation; IQ: interquartile				
[tdemo01.rtf][C:/phuse2023_tidytlg/tdemo01.R] 26JAN2023, 14:57				

- For html output, add two arguments in the gentlg call: format = "HTML", print.hux = FALSE

Example 2: Adverse Events Summary Table

Counting Subjects with AEs by Body System and Preferred term

```
library(dplyr, warn.conflicts = FALSE)
library(stringr)
library(tidytlg)

# filter treatment emergent AE records for safety population
adae <- cdisc_adae %>%
  filter(SAFFL == "Y", TRTEMFL == "Y") %>%
  # filter one body system for illustration
  filter(AEBODSYS == "GENERAL DISORDERS AND ADMINISTRATION SITE CONDITIONS") %>%
  # convert AEBODSYS and AEDECOD to sentence case
  mutate(AEBODSYS = str_to_sentence(AEBODSYS), AEDECOD = str_to_sentence(AEDECOD)) %>%
  rename(TRT01AN = TRTAN)

# filter safety population
safetypop <- cdisc_adsl %>%
  filter(SAFFL == "Y")

# frequency of safety population
tbl1 <- freq(safetypop,
  rowvar = "SAFFL",
  statlist = statlist("n"),
  colvar = "TRT01AN",
  rowtext = "Analysis Set: Safety Population",
  subset = SAFFL == "Y")

# nested frequency of AEBODSYS by AEDECOD
tbl2 <- nested_freq(adae,
  denom_df = safetypop,
  colvar = "TRT01AN",
  rowvar = "AEBODSYS*AEDECOD",
  statlist = statlist("n (x.x%)"),
  descending_by = "81",
  row_header = "System organ class \\n Preferred term")

# combine results together
tbl <- bind_table(tbl1, tbl2) %>%
  select(-AEBODSYS)
```

- Process ADAE data for selecting analysis records
- Denominator data (denom_df) are from AD_SL for calculating percentages
- Use nested_freq function to summarize nested counts and percentages by AEBODSYS and AEDECOD
 - rowvar = "AEBODSYS*AEDECOD"
 - descending_by argument is used to specify the column for sorting the counts of (a) AEBODSYS (b) AEDECOD within each AEBODSYS in descending order
 - additional arguments of cutoff and cutoff_stat are available to filter out low frequency of AEDECOD terms for reporting

Example 2: Adverse Events Summary Table

```
# Create rtf output -----
gentlg(huxme      = tbl ,
       orientation = "portrait",
       file        = "TAE01",
       title       = "Summary of treatment-emergent adverse events",
       footers     = "Note: subjects are counted only once for any given event",
       colspan     = list(c("", "", "Xanomeline", "Xanomeline")),
       colheader   = c("", "Placebo", "High", "Low"),
       wcol        = .50)
```

TAE01: Summary of treatment-emergent adverse events			
		Xanomeline	
	Placebo	High	Low
Analysis Set: Safety Population	5	5	5
System organ class			
Preferred term			
General disorders and administration site conditions	2 (40.0%)	3 (60.0%)	5 (100.0%)
Application site pruritus	1 (20.0%)	2 (40.0%)	5 (100.0%)
Application site erythema	1 (20.0%)	1 (20.0%)	4 (80.0%)
Fatigue	0	1 (20.0%)	2 (40.0%)
Application site irritation	0	1 (20.0%)	1 (20.0%)
Application site pain	0	0	1 (20.0%)
Application site vesicles	0	1 (20.0%)	1 (20.0%)
Application site dermatitis	0	1 (20.0%)	0
Application site urticaria	0	1 (20.0%)	0
Pyrexia	1 (20.0%)	0	0

Note: subjects are counted only once for any given event

[tae01.rtf][C:/R/phuse2023_tidytlg/tae01.R] 23JAN2023, 16:40

Example 3: Laboratory Summary Table

```
library(dplyr, warn.conflicts = FALSE)
library(tidytlg)

# filter safety population records
adlb <- cdisc_adlb %>%
  filter(SAFFL == "Y") %>%
  # filter ALT and AST records for illustration
  filter(PARAMCD %in% c("ALT", "AST")) %>%
  filter(AVISIT %in% c("Baseline", "Week 4", "Week 8", "Week 12"))

# convert AVISIT to factor based on AVISITN
adlb$AVISIT <- char2factor(adlb, "AVISIT", "AVISITN")

# call univar to summarize descriptive stats by PARAM and AVISIT
tbl1 <- univar(adlb,
  colvar = "TRTAN",
  tablebyvar = "PARAM",
  rowvar = "AVAL",
  rowbyvar = "AVISIT",
  statlist = statlist(c("N", "MEANS", "MEDIAN", "RANGE")))

# bind analysis results with formatting variables
tbl <- bind_table(tbl1, tablebyvar = "PARAM", rowbyvar = "AVISIT")
```

- Process ADLB data: select analysis records and convert the analysis visit (AVISIT) variable to a factor
- Call univar function to summarize univariate descriptive statistics of the lab measurements for each parameter and analysis visit
 - Use rowbyvar = “AVISIT” to repeat descriptive analysis for each visit within a parameter
 - Use tablebyvar = “PARAM” to repeat the entire by visit analysis for each parameter

Example 3: Laboratory Summary Table

```
# Create rtf output -----
gentlg(huxme      = tbl,
       orientation = "portrait",
       file        = "TLB01",
       title       = "Summary of descriptive statistics for lab values",
       footers     = "Note: SD: standard deviation",
       colspan      = list(c("", "", "Xanomeline", "Xanomeline")),
       colheader    = c("", "Placebo", "High", "Low"),
       wcol         = .30)
```

TLB01: Summary of descriptive statistics for lab values				
		Xanomeline		
		Placebo	High	Low
Alanine Aminotransferase (U/L)				
Baseline				
N		5	5	5
Mean (SD)		20.40 (5.273)	18.80 (3.114)	22.60 (8.019)
Median		22.00	18.00	23.00
Range		(15.0; 27.0)	(16.0; 23.0)	(15.0; 34.0)
Week 4				
N		5	4	5
Mean (SD)		21.80 (9.230)	16.75 (4.500)	24.40 (7.537)
Median		18.00	15.50	21.00
Range		(15.0; 38.0)	(13.0; 23.0)	(16.0; 35.0)
Week 8				
N		3	2	4
Mean (SD)		15.67 (6.028)	17.00 (1.414)	26.00 (7.703)
Median		15.00	17.00	27.50
Range		(10.0; 22.0)	(16.0; 18.0)	(16.0; 33.0)
Week 12				
N		3	1	4
Mean (SD)		18.67 (7.371)	14.00 (-)	26.00 (10.231)
Median		16.00	14.00	24.50
Range		(13.0; 27.0)	(14.0; 14.0)	(16.0; 39.0)

Example 4: Listing of Adverse Events

```
library(dplyr, warn.conflicts = FALSE)
library(stringr)
library(tidytlg)

# process AE data for listing
tbl <- cdisc_adae %>%
  filter(SAFFL == "Y" & TRTEMFL == "Y") %>%
  mutate(SAEFL = if_else(AESER == "Y", "Yes", "No"),
         DTHFL = if_else(AEOUT == "FATAL", "Yes", "No"),
         AETERM = str_to_sentence(AETERM)) %>%
  # filter records of 2 subjects for illustration
  filter(USUBJID %in% c("01-701-1015", "01-701-1023")) %>%
  # select columns to display in listing
  select(TRTA, USUBJID, ASTDY, AETERM, SAEFL, DTHFL)

# Create rtf output -----
gentlg(huxme      = tbl,
      tlf         = "l",
      orientation = "landscape",
      file        = "LSAE01",
      title       = "Listing of Adverse Events",
      idvars      = c("TRTA", "USUBJID"),
      wcol        = 0.10,
      colheader   = c("Treatment Group",
                      "Subject ID",
                      "Study Day of AE",
                      "Verbatim Term",
                      "Serious",
                      "Fatal"))
```

- Prepare data for listing and assign it to tbl
- In the gentlg call:
 - Use tlf = “l” for creating listing
 - Use idvars argument to remove repeated values

LSAE01: Listing of Adverse Events					
Treatment Group	Subject ID	Study Day of AE	Verbatim Term	Serious	Fatal
Placebo	01-701-1015	2	Application site erythema	No	No
		2	Application site pruritus	No	No
	01-701-1023	8	Diarrhoea	No	No
		3	Erythema	No	No
		3	Erythema	No	No
		22	Atrioventricular block second degree	No	No
		3	Erythema	No	No

[lsae01.rtf][C:/R/phuse2023_tidytlg/lsae01.R] 25JAN2023, 15:18

Example 5: Graph Output

- Create plot object using ggplot2
- In the gentlg call:
 - Use huxme = plot
 - Use tlf = "f"
 - Specify title and file name

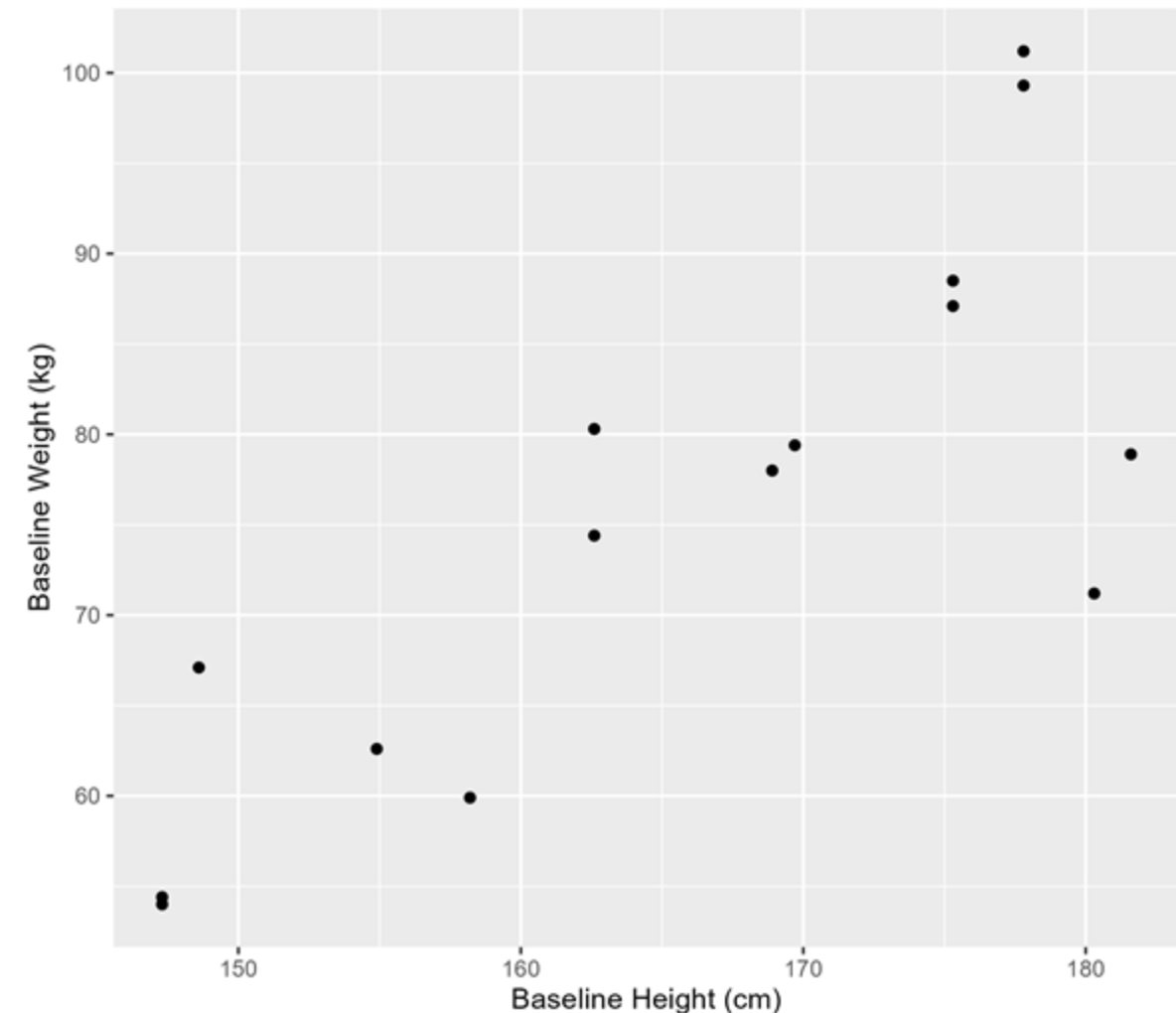
```
library(dplyr, warn.conflicts = FALSE)
library(ggplot2)
library(tidytlg)

# process data
adsl <- cdisc_adsl %>%
  filter(ITTFL == "Y")

# generate plot
plot <- ggplot(data = adsl, aes(x = HEIGHTBL, y = WEIGHTBL)) +
  geom_point() +
  labs(x = "Baseline Height (cm)",
       y = "Baseline Weight (kg)")

# Create rtf output -----
gentlg(huxme      = plot,
      tlf         = "f",
      orientation = "landscape",
      file        = "Graph01",
      title       = "Scatter plot of baseline body weight and height")
```

Graph01: Scatter plot of baseline body weight and height



[graph01.rtf][C:/R/phuse2023_tidytlg/graph01.R] 25JAN2023, 16:13

Metadata-driven Table Creation

- Table metadata: a dataframe describing the data, functions, and arguments needed to produce the table results
- Use generate_results function with inputs of table metadata and column metadata to create the tbl dataframe
- Column metadata: column structure of table layout defined by coldef (column variable), decode (column header), span1...3 (spanning header)

	A	B	C	D	E	F
1	tbltype	coldef	decode	span1	span2	span3
2	type1	0	Placebo			
3	type1	54	Low Dose	Xanomeline		
4	type1	81	High Dose	Xanomeline		
5	type2	0	Placebo			
6	type2	54	Low Dose	Xanomeline		
7	type2	81	High Dose	Xanomeline		
8	type2	54+81	Combined	Xanomeline		
9	type3	0	Placebo			
10	type3	54	Low Dose	Xanomeline		
11	type3	81	High Dose	Xanomeline		
12	type3	54+81	Combined	Xanomeline		
13	type3	0+54+81	Total			

```
# define table metadata
table_metadata <- tibble::tribble(
  ~func, ~df, ~rowvar, ~rowtext, ~row_header, ~statlist, ~subset,
  "freq", "adsl", "ITTFL", "Analysis set: ITT", NA, statlist("n"), "ITTFL == 'Y'",
  "univar", "adsl", "AGE", NA, "Age (Years)", NA, NA,
  "freq", "adsl", "SEX", NA, "Gender", statlist(c("N", "n (x.x%)")), NA
) %>%
  mutate(colvar = "TRT01PN")

# Generate results
tbl <- generate_results(table_metadata,
  column_metadata_file = system.file("extdata/column_metadata.xlsx", package = "tidytlg"),
  tbltype = "type1")
```

- The same tbl of the demographic table is produced

	label	col1	col2	col3	row_type	func	anbr	indentme	roworder	newrows
1	Analysis set: ITT	5	5	5	HEADER	freq	1	0	1	0
2	Age (Years)				HEADER	univar	2	0	1	1
3	N	5	5	5	N	univar	2	1	2	0
4	Mean (SD)	69.60 (14.398)	75.60 (6.731)	72.20 (9.230)	VALUE	univar	2	2	3	0
5	Median	64.00	74.00	75.00	VALUE	univar	2	2	4	0
6	Range	(52.0; 85.0)	(68.0; 84.0)	(57.0; 81.0)	VALUE	univar	2	2	5	0
7	IQ range	(63.00; 84.00)	(71.00; 81.00)	(71.00; 77.00)	VALUE	univar	2	2	6	0
8	Gender				HEADER	freq	3	0	1	1
9	N	5	5	5	N	freq	3	1	2	0
10	Male	3 (60.0%)	4 (80.0%)	2 (40.0%)	VALUE	freq	3	2	3	0
11	Female	2 (40.0%)	1 (20.0%)	3 (60.0%)	VALUE	freq	3	2	4	0

Summary

- Tidytlg is developed to increase the adoption of R for clinical reporting by reducing the learning curve from SAS to R
- TLG creation can be implemented in multiple ways with tidytlg:
 - Functional method (intuitive and easy to learn)
 - Metadata method (efficient for generic table creation)
- Tidytlg has been open-sourced in [Pharmaverse Github](#), and we welcome your feedback on bug fixes, functionality improvements, and contributions to improving this package

Acknowledgments

We would like to acknowledge **Nicholas Masel** for your guidance, support from **Sumesh Kalappurakal**, and thank to **Aidan Ceney, Nathan Kosiba, Steven Haesendonckx, Alexandra Papadopoulou** for all your contributions to the tidytlg package.