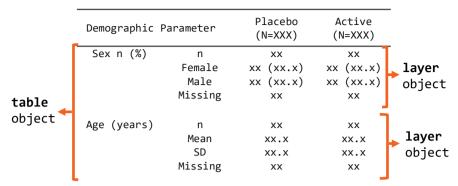
# Traceability Focused Clinical Data Summary with Tplyr: : CHEAT SHEET



# **CREATING THE TABLE OBJECT**

tplyr\_table(target, treat\_var, where=TRUE, cols=vars()) - used to create the
table object

## ADDING LAYERS TO A TABLE

add\_layer(parent, layer, name=NULL) - Constructs the layer within the call to the function.

add layers (parent, ...) - Attaches layers that have already been constructed.

### CREATING LAYER OBJECTS

```
group_<type>(parent, target_var, by=vars(), where=TRUE, ...) - family of functions used to create layers.
       group_count(t, SEX, by="Sex n (%)")
                                                       group_shift(t, vars(row=BNRIND, column=ANRIND),
                                                                                                                 group desc(t, AGE, by="Age (years)"))
                                                       by=vars(PARAM, AVISIT))
                                                                                                                   Age (years)
          Sex n (%)
                            53 (61.6%)
                                                                       VISIT
                                                                                                                                Mean (SD)
                                                                                                                                           75.2 (8.59)
                                                              PARAM 1 VISIT 1
                                                                                                                                 Median
                                                                                                                                              76.0
                            33 (38.4%)
                                                                                     0
                                                                                         0 1
                                                                                                                                 Q1, Q3
                                                                                                                                            69.2, 81.8
                                                                                     3
                                                                                        1
                                                                                            0
                                                                                                                                Min, Max
                                                                                                                                             52, 89
                                                                                     0
                                                                                         7
```

# **BUILDING AND USING METADATA**

**build**(x, metadata) – Triggers the execution of the tplyr\_table and optionally the associated metadata.

```
add_Layer(group_count(RACE))
t %>%
build(metadata = TRUE)

row_id row_label1 var1_Placebo var1_Treated
c1_1 ASIAN 0 (0.0%) 2 (20.0%)
c2 1 WHITE 9 (100.0%) 8 (80.0%)
```

t <- tplyr table(adsl, TRT01P, where = SAFFL == "Y") %>%

get\_meta\_subset(x, row\_id, column, add\_cols=vars(USUBJID), ...) Extracts the subset of data based on result metadata.

```
get_meta_subset(t, 'c1_1', 'var1_Treated', add_cols = vars(USUBJID, SEX))

USUBJID SEX TRT01P SAFFL RACE
004 F Treated Y ASIAN
007 M Treated Y ASIAN
```

**get\_meta\_result**(x, row\_id, column, ...) – Extracts the result metadata of a tplyr\_table.

```
get_meta_result(t, 'c1_1', 'var1_Treated')

#> tplyr_meta: 3 names, 3 filters

#> Names:

#> TRT01P, SAFFL, RACE

#> Filters:

#> TRT01P == c("Treated"), SAFFL == "Y", RACE == c("ASIAN")
```

# **TEMPLATES AND TABLE FORMATS**

new\_layer\_template(name, template) - Creates a layer template.
use\_template(name, ..., add\_params=NULL) - Uses a layer template.



set\_<type>\_layer\_formats(obj, ...) - Sets default format
strings for layers type.

## **GENERAL STRING FORMATTING**

The f\_str() object controls the numbers reported.

_	•
xx (xx.x% 8 (53.3%) 1 ( 6.7%)	::_:::::::::::::::::::::::::::::::
xx (XX.x% 8 (53.3%) 1 (6.7%)	• INTEGERMITA / SNACES AND I DECIMALNIACE

- Decimals round to the specified length.
- Integers will not truncate. If an integer exceeds the set length, Tplyr will push the number over.

# **POST PROCESSING**

str\_indent\_wrap(x, width=10, tab\_width=5) - Wrap strings to a specific
width with hyphenation while preserving indentation.

apply\_row\_masks(dat, row\_breaks=FALSE, ...) - Replace repeating row label variables with blanks in preparation for display and optionally inserts row breaks.

apply\_conditional\_format(string, format\_group, condition, replacement, full\_string=FALSE) - Applies conditional formatting of a pre-populated string of numbers.

```
i.e. "0 (0.0%)" -> "0" or "1 (0.004%)" -> "1 (<0.1%)"
```

str\_extract\_fmt\_group(string, format\_group) - Extracts format group
strings.

```
string <- c(" 5 (5.8%)", " 8 (9.3%)", "78 (90.7%)")
str_extract_fmt_group(string, 1)
#> [1] " 5" " 8" "78"
str_extract_fmt_group(string, 2)
#> [1] "(5.8%)" "(9.3%)" "(90.7%)"
```

str\_extract\_num(string, format\_group) - Extracts format group numbers.
apply\_formats(format\_string, ..., empty=c(.overall="")) - Applies
format strings outside of a tplyr\_table.

# SORTING

Ordering helpers are columns added into Tplyr tables.

#### **SORTING THE LAYERS**

Layers are indexed using the variable ord\_layer\_index by the order in which they were added to the table using add\_layer() or add\_layers().

Tplyr

row label2 var1 Placebo

<chr>

14 ( 16.3%)

42 ( 48.8%)

30 ( 34.9%)

86 (100.0%)

<chr>

65-80

>80

<65

#### **SORTING THE BY VARIABLES**

Each by variable gets an ord\_layer\_<n> column. The order variables will calculate based on the first applicable method:

- Use factor levels if variable is a factor
- Use a matching variable name suffixed by N from the dataset if available (i.e. RACE and RACEN)
- Use alphanumeric sorting of variable values

#### **SORTING COUNT LAYER RESULTS**

Count layers get an ord\_layer\_<n> column based on the sort method specified in set\_order\_count\_method().

set\_order\_count\_method("byfactor") - Use
factor levels. If variable is not a factor,
alphanumeric sorting will be used. This is the
default method and

set\_order\_count\_method() does not need to
be called.

set\_order\_count\_method("byvarn") - Use a
matching variable name suffixed by N from
the dataset if available (i.e. RACE and RACEN)

set\_order\_count\_method("bycount") - Sort
based on counts in a particular column.
Requires the use of additional helper
functions:

- set\_ordering\_cols(e, ...) Specifies the treat\_var and cols= value(s) from tplyr\_table() to determine the column from which the ordering should be based. set\_ordering\_cols("High", "WHITE")
- set\_result\_order\_var(e, result\_order\_var) - Specifies the occurrence or proportion variable on which the ordering should be based. set\_result\_order\_var(n)

# SORTING DESCRIPTIVE STATISTICS LAYER RESULTS

Descriptive statistics layers get an ord\_layer\_<n> column based on the order in which the f\_str() objects are created through set\_format\_strings().

# Traceability Focused Clinical Data Summary with Tplyr: : CHEAT SHEET

## **COUNT AND SHIFT LAYERS**

#### **CALCULATING PERCENTAGES**

set\_denoms\_by(e, ...) - Specifies variable(s) to use to calculate
percentages. If not called, uses treat\_var and cols= from
tplyr\_table().

set\_denom\_where(e, denom\_where) - Specifies denominator
subset. If not called, uses where= from group <type>()....

#### MISSING COUNTS PRESENTATION

set\_missing\_count(e, fmt=NULL, sort\_value=NULL, .....
denom\_ignore=FALSE, ...) - Controls how missing counts are handled.

#### ADDING A 'TOTAL' ROW

add\_total\_row(e, fmt=NULL, count\_missings=TRUE,
sort\_value=NULL) - Adds a row presenting the total counts (i.e.,
the n's that are summarized).

set\_total\_row\_label(e, total\_row\_label) - Specifies a row
label for the total row. If not called, default text will be "Total".

#### **NESTED COUNTS**

When calculating **nested counts** use dplyr::vars() to specify 2 variables for target\_var.

#### **DISTINCT VS EVENT COUNTS**

**set\_distinct\_by**(e, distinct\_by) - Specifies variable(s) to use to calculate distinct occurrences.

#### **FORMATTING**

set\_format\_strings() and f\_str() are used to specify the occurrence and proportion variables and how they will be presented.

```
t <- tplyr_table(adsl, TRT01P, where = SAFFL == "Y") %>%
   add_total_group() %>%
   add_treat_grps('Treated' = c("High Dose", "Low Dose")) %>%
   add_layer(
   group_count(AGEGR1, by = RACE, where = SEX == "F") %>%
   set_denoms_by(TRT01P, RACE) %>%
   set_denom_where(TRUE) %>%
   set_missing_count(f_str("xx", n), Missing = NA, denom_ignore = TRUE) %>%
   add_total_row(f_str("xx", n), count_missings = FALSE) %>%
   set_total_row_label("n") )

t %>%
   build()
```

```
t <- tplyr_table(adae, TRTA, where = AESER == "Y") %>%
    set_pop_data(ads1) %>%
    set_pop_treat_var(TRTA) %>%
    set_pop_where(SAFFL == "Y") %>%
    add_layer(
        group_count(vars(AEBODSYS, AEDECOD)) %>%
        set_distinct_by(USUBJID) %>%
        set_format_strings(f_str("xx (XX.x) [XX]", distinct_n, distinct_pct, n))
        t %>%
        build()
```

# group\_shift() is an abstraction of group\_count() and can be used with many of the same functions

row label2 var1 Placebo

<chr>

14 ( 16.3%) 42 ( 48.8%)

30 ( 34.9%)

86 (100.0%)

<chr>

<65

65-80

# TABLE LEVEL FUNCTIONS

#### **ADDING TREATMENT GROUPS**

add\_treat\_grps(table, ...) - Create new treatment groups by combining existing treatment groups from the values within treat var.

add\_total\_group(table, group\_name="Total")
- Create total treatment group by combining
all treatment groups from the values within
treat var.

#### ADDING A POPULATION DATASET

If target does not include the entire necessary population, the **population functions** can provide population information.

set\_pop\_data(table, pop\_data) - Specifies a
population dataset.

set\_pop\_treat\_var(table, pop\_treat\_var) Specifies a treatment variable from the
population dataset. If not called, uses
treat var from tplyr table().

set\_pop\_where(obj, where) - Specifies a
population subset. If not called, uses where=
from tplyr table().

# **DESCRIPTIVE STATISTIC LAYERS**

#### **BUILT-IN SUMMARIES**

Description	Variable Name		
N	n		
Mean	mean		
Standard Deviation	sd		
Median	median		
Variance	variance		
Minimum	min		
Maximum	max		
Interquartile Range	iqr		
Q1	q1		
Q3	q3		
Missing	missing		

#### **CUSTOM SUMMARIES**

Custom summaries allow any function to be used in a descriptive statistics layer.

**set\_custom\_summaries**(e, ...) – Allows user to define custom summaries that will be performed in dplyr::summarize. Use .var as the variable name being summarized.

#### FORMATTING AND PERFORMING SUMMARIES

set\_format\_strings() and f\_str() are used to specify
the summaries that will be performed and how they will
be presented.

- On the left side of the equal sign the user inputs text that becomes the row label.
- On the right side the user specifies how the numbers will be displayed and lists the descriptive statistic summaries that will be performed.

The empty parameter of f\_str() specifies what to display if an element or elements in a cell produce NA values.

**Auto precision** is used to format numeric summaries based on the precision of the data collected.

- Use a/A instead of x/X
- Use a+n/A+n where n is the number of additional spaces you wish to add
- Use the cap parameter to cap the length allotted for integers and decimals

