Package ‘gtreg’

October 18, 2022

Title Regulatory Tables for Clinical Research

Version 0.2.0

Description Creates tables suitable for regulatory agency submission by leveraging the ‘gtsummary’ package as the back end. Tables can be exported to HTML, Word, PDF and more. Highly customized outputs are available by utilizing existing styling functions from ‘gtsummary’ as well as custom options designed for regulatory tables.

License GPL (>= 3)

URL https://github.com/shannonpileggi/gtreg,
https://shannonpileggi.github.io/gtreg/

BugReports https://github.com/shannonpileggi/gtreg/issues

Depends R (>= 3.4)

Imports broom.helpers (>= 1.5.0), cli (>= 3.1.0), dplyr (>= 1.0.7), forcats (>= 0.5.1), glue (>= 1.6.0), gtsummary (>= 1.6.1), purrr (>= 0.3.4), rlang (>= 1.0.0), stringr (>= 1.4.0), tibble (>= 3.1.6), tidyr (>= 1.1.4)

Suggests covr (>= 3.5.1), gt (>= 0.3.1), knitr (>= 1.37), rmarkdown, spelling (>= 2.2), testthat (>= 3.1.1)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.2.1

NeedsCompilation no

Author Shannon Pileggi [aut, cre, cph]
  (https://orcid.org/0000-0002-7732-4164),
  Daniel D. Sjoberg [aut] (https://orcid.org/0000-0003-0862-2018)

Maintainer Shannon Pileggi <shannon.pileggi@gmail.com>

Repository CRAN

Date/Publication 2022-10-18 15:25:09 UTC
R topics documented:

.complete_ae_data .................................................. 2
add_overall_tbl_ae .................................................. 3
df_adverse_events .................................................. 5
df_patient_characteristics ......................................... 6
inline_text_tbl_ae .................................................... 7
selectors ................................................................. 8
style_xxx ............................................................... 9
tbl_ae ................................................................. 10
tbl_ae_count .......................................................... 12
tbl_ae_focus ........................................................... 13
tbl_listing .............................................................. 15
tbl_reg_summary ....................................................... 17

Index 19

---------------------------------------------------------------

.complete_ae_data  Create a complete and expanded data frame for tabulating adverse events

---------------------------------------------------------------

Description

Returns a data frame that has an observation for each patient in the study, with combinations for each ID, SOC, and AE. The returned data frame includes new logical columns "..ae.." and "..soc.." indicating whether that row should be included when tabulating the AE table. When multiple AEs of the same type are observed, the AE with the largest by= value is the observation to be used in the tabulation.

Usage

.complete_ae_data(
  data,
  id,
  ae,
  soc = NULL,
  by = NULL,
  strata = NULL,
  id_df = NULL,
  by_values = NULL,
  missing_text = "Unknown",
  missing_location = "first"
)
Arguments

- **data**: Data frame
- **id**: String variable name of the patient ID
- **ae**: String variable name of the adverse event column
- **soc**: Optional string variable name of the system organ class column
- **by**: Optional string variable to split results by, e.g. report AEs by grade or attribution
- **strata**: Optional string variable to stratify results by, e.g. report AEs summaries by treatment group
- **id_df**: Optional data frame of complete id values and strata to achieve correct base n for the situation in which not all subjects experience adverse events
- **by_values**: Optional vector of complete by values, listed in desired order, to achieve correct table structure for the situation in which an adverse event of a certain grade is not observed for a given soc
- **missing_text**: String that will be shown for missing levels of by=, Default is "Unknown"
- **missing_location**: location where the column summarizing values with missing levels by= will be located in the final table. Must be one of c("first", "last", "hide"). Default is "first"

Value

- a tibble

Examples

```r
df_adverse_events %>%
  complete_ae_data(
    id = "patient_id",
    ae = "adverse_event",
    soc = "system_organ_class",
    by = "grade",
    strata = "trt"
  )
```

---

Tabulate Overall Summary

Description

Tabulate Overall Summary
Usage

## S3 method for class 'tbl_ae'
add_overall(x, across = NULL, ...)

## S3 method for class 'tbl_ae_count'
add_overall(x, across = NULL, ...)

## S3 method for class 'tbl_ae_focus'
add_overall(x, across = NULL, ...)

Arguments

x Object of class "tbl_ae", "tbl_ae_count", or "tbl_ae_focus"

across Specify the type of overall statistics to include.
• "both" adds summaries across both the by= and strata= levels
• "by" adds summaries across the by= levels
• "strata" adds summaries across the strata= levels
• "overall-only" adds a single overall column Default is all possible over-
all types.

... Not used

Value

Summary object of same input class

Notes

If the spanning headers are modified prior to the call of add_overall(), the ordering of the columns may not be correct.

Example Output

Examples

# Example 1 -----------------------------------------------------------------
add_overall_ex1 <-
df_adverse_events %>%
tbl_ae_count(
  ae = adverse_event,
  soc = system_organ_class,
  by = grade,
  strata = trt
) %>%
add_overall() %>%
modify_header(all_ae_cols() ~ "**Grade (by)**") %>%
bold_labels()
df_adverse_events

# Example 2 -----------------------------------------------------------------
add_overall_ex2 <-
  df_adverse_events %>%
  tbl_ae(
    id = patient_id,
    ae = adverse_event,
    soc = system_organ_class,
    by = grade
  ) %>%
  add_overall(across = 'by') %>%
  modify_header(all_ae_cols() ~ "**Grade (by)**") %>%
  bold_labels()

# Example 3 -----------------------------------------------------------------
add_overall_ex3 <-
  df_adverse_events %>%
  tbl_ae_focus(
    id = patient_id,
    include = c(any_complication, grade3_complication),
    ae = adverse_event,
    strata = trt
  ) %>%
  add_overall(across = 'strata')

# Example 4 -----------------------------------------------------------------
add_overall_ex4 <-
  df_adverse_events %>%
  tbl_ae(
    id = patient_id,
    ae = adverse_event,
    soc = system_organ_class,
    by = grade,
    strata = trt
  ) %>%
  add_overall(across = 'overall-only') %>%
  modify_header(all_ae_cols() ~ "**Grade (by)**") %>%
  bold_labels()

---

df_adverse_events  Simulated Adverse Event Database

Description
A data set containing reported AEs from a trial.

Usage
  df_adverse_events
df_patient_characteristics

Format

A data frame with 100 rows–one row per patient per AE

patient_id  Patient ID  
trt  Treatment Group  

system_organ_class  System Organ Class  

adverse_event  Adverse Event  

grade  Grade  


drug_attribution  Drug Attribution  

any_complication  Any Grade Complication  

grade3_complication  Grade 3+ Complication  


Simulated Patient Characteristics Database

Description

Simulated Patient Characteristics Database

Usage

df_patient_characteristics

Format

A data frame with 100 rows–one row per patient

patient_id  Patient ID  
trt  Treatment Group  

age  Patient Age  


marker  Biological Marker  

status  Study Status  

discontinued  Discontinued from Study  

off_trt_ae  Off Treatment Adverse Event
Description

Function allows users to report formatted and styled results from gtreg tables in-line.

Usage

```r
## S3 method for class 'tbl_ae'
inline_text(x, row, column = NULL, ...)

## S3 method for class 'tbl_ae_count'
inline_text(x, row, column = NULL, ...)

## S3 method for class 'tbl_ae_focus'
inline_text(x, row, column = NULL, ...)
```

Arguments

- **x**: an object of class `tbl_ae()`, `tbl_ae_count()`, `tbl_ae_focus()`
- **row**: string indicating the AE or SOC to report
- **column**: column name of cell to report. Use `show_header_names(x)` to print all column names beside the current header.
- **...**: not used

Value

- string

Examples

```r
tbl <-
df_adverse_events %>%
tbl_ae(
  id = patient_id,
  ae = adverse_event,
  soc = system_organ_class,
  by = grade
)
show_header_names(tbl)

inline_text(tbl, "Anaemia", column = stat_5)
```
Column Selectors

**Description**

See the Table modifications article for examples.

- `all_ae_cols(overall, unknown)` selects all columns summarizing AE statistics. By default, unknown and overall columns are not selected.
- `all_cols_in_strata(strata)` selects all columns from specified stratum.
- `all_overall_cols()` selects all overall columns
- `all_unknown_cols()` selects all unknown columns

**Usage**

```r
all_ae_cols(overall = FALSE, unknown = FALSE)
all_cols_in_strata(strata)
all_overall_cols()
all_unknown_cols()
```

**Arguments**

- `overall` logical indicating whether to include the overall columns. Default is `FALSE`
- `unknown` logical indicating whether to include the unknown or missing columns. Default is `FALSE`
- `strata` character vector of the selected stratum

**Value**

selected columns

**Example Output**

**See Also**

`gtsummary::all_stat_cols()`
**Examples**

```r
selectors_ex1 <-
  df_adverse_events %>%
  dplyr::mutate(grade = ifelse(dplyr::row_number() == 1L, NA, grade)) %>%
  tbl_ae(
    id = patient_id,
    ae = adverse_event,
    soc = system_organ_class,
    by = grade
  ) %>%
  add_overall(across = 'by') %>%
  modify_header(
    all#ae_cols() ~ "**Grade {by}**",
    all_overall_cols() ~ "**Total**",
    all_unknown_cols() ~ "**Unknown Grade**"
  )
```

---

**style_xxx**

*Style numbers as x’s*

**Description**

The purpose of `style_xxx()` is to convert numeric values in summary tables to x’s of consistent length for mock tables. See the [Table shells vignette](#) for detailed examples.

**Usage**

```r
style_xxx(x, width = digits + 2, digits = 0)
```

**Arguments**

- `x`: a numeric or character vector
- `width`: the width of output field of x’s, including the decimal place
- `digits`: the number of digits displayed after the decimal place

**Value**

a character vector

**Examples**

```r
style_xxx(7:10, digits = 0)
style_xxx(7:10, digits = 1)
style_xxx(7:10, width = 2, digits = 0)
style_xxx(7:10, width = 5, digits = 2)
```
**Description**

The function tabulates adverse events. One AE per ID will be counted in the resulting table. If a by= variable is passed and a patient experienced more than one of the same AE, the AE associated with the highest by= level will be included. For example, if a patient has two of the same AE and by = grade, the AE with the highest grade will be included. Similarly, if tabulations within system organ class are requested, the AE within SOC associated with the highest grade will be tabulated.

**Usage**

```r
tbl_ae(
  data,
  id,
  ae,
  soc = NULL,
  by = NULL,
  strata = NULL,
  id_df = NULL,
  statistic = "{n} ({p})",
  by_values = NULL,
  digits = NULL,
  sort = NULL,
  zero_symbol = "\U2014",
  missing_location = c("first", "last", "hide")
)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Data frame</td>
</tr>
<tr>
<td>id</td>
<td>Variable name of the patient ID</td>
</tr>
<tr>
<td>ae</td>
<td>Variable name of the adverse event column</td>
</tr>
<tr>
<td>soc</td>
<td>Variable name of the system organ class column</td>
</tr>
<tr>
<td>by</td>
<td>Variable to split results by, e.g. report AEs by grade</td>
</tr>
<tr>
<td>strata</td>
<td>Variable to stratify results by, e.g. report AEs summaries by treatment group</td>
</tr>
<tr>
<td>id_df</td>
<td>Optional data frame of complete id values and strata to achieve correct base n for the situation in which not all subjects experience adverse events. See df_patient_characteristics for an example id_df that pairs with df_adverse_events.</td>
</tr>
<tr>
<td>statistic</td>
<td>String indicating the statistics that will be reported. The default is &quot;{n} ({p})&quot;</td>
</tr>
<tr>
<td>by_values</td>
<td>Optional vector of complete by values, listed in desired order, to achieve correct table structure for the situation in which an adverse event of a certain grade is not observed for a given soc</td>
</tr>
</tbody>
</table>
digits

Specifies the number of decimal places to round the summary statistics. By default integers are shown to zero decimal places, and percentages are formatted with `style_percent()`. If you would like to modify either of these, pass a vector of integers indicating the number of decimal places to round the statistics. For example, if the statistic being calculated is `{n} ({p}%)` and you want the percent rounded to 2 decimal places use `digits = c(0, 2)`. User may also pass a styling function: `digits = style_sigfig`.

sort

Controls order of AEs and SOCs in output table. The default is `NULL`, where AEs and SOCs are sorted alphanumerically (and factors sorted according to their factor level). Use `sort = "ae"` to sort AEs in decreasing frequency order, `sort = "soc"` to sort SOCs in decreasing order, and `sort = c("ae", "soc")` to sort both. AEs are sorted within SOC.

zero_symbol

String used to represent cells with zero counts. Default is the em-dash ("\U2014"). Using `zero_symbol = NULL` will print the zero count statistics, e.g. "0 (0)"

missing_location

location where the column summarizing values with missing levels by= will be located in the final table. Must be one of `c("first", "last", "hide")`. Default is "first"

Value

a ‘tbl_ae’ object

Example Output

Examples

```r
# Example 1 -----------------------------------------------------------------
tbl_ae_ex1 <-
  df_adverse_events %>%
  tbl_ae(
    id = patient_id,
    ae = adverse_event,
    soc = system_organ_class,
    by = grade,
    strata = trt
  ) %>%
  modify_header(all_ae_cols() ~ "**Grade {by}**")

# Example 2 -----------------------------------------------------------------
tbl_ae_ex2 <-
  df_adverse_events %>%
  tbl_ae(
    id = patient_id,
    ae = adverse_event,
    by = grade
  ) %>%
```

modify_header(all_ae_cols() ~ "**Grade (by)**")

### tbl_ae_count

**Tabulate Raw AE Counts**

**Description**

Create a table counting all AEs.

**Usage**

```r
tbl_ae_count(
  data,
  ae,
  soc = NULL,
  by = NULL,
  strata = NULL,
  by_values = NULL,
  digits = NULL,
  sort = NULL,
  zero_symbol = "\U2014",
  missing_location = c("first", "last", "hide")
)
```

**Arguments**

- **data**
  Data frame
- **ae**
  Variable name of the adverse event column
- **soc**
  Variable name of the system organ class column
- **by**
  Variable to split results by, e.g. report AEs by grade
- **strata**
  Variable to stratify results by, e.g. report AEs summaries by treatment group
- **by_values**
  Optional vector of complete by values, listed in desired order, to achieve correct table structure for the situation in which an adverse event of a certain grade is not observed for a given soc
- **digits**
  Specifies the number of decimal places to round the summary statistics. By default integers are shown to zero decimal places, and percentages are formatted with `style_percent()`. If you would like to modify either of these, pass a vector of integers indicating the number of decimal places to round the statistics. For example, if the statistic being calculated is `{n} ({p}%)` and you want the percent rounded to 2 decimal places use `digits = c(0, 2)`. User may also pass a styling function: `digits = style_sigfig`
- **sort**
  Controls order of AEs and SOCs in output table. The default is NULL, where AEs and SOCs are sorted alphanumerically (and factors sorted according to their factor level). Use `sort = "ae"` to sort AEs in decreasing frequency order, `sort = "soc"` to sort SOCs in decreasing order, and `sort = c("ae", "soc")` to sort both. AEs are sorted within SOC.
`tbl_ae_focus`  

`zero_symbol`  

String used to represent cells with zero counts. Default is the em-dash ("\u2014"). Using `zero_symbol = NULL` will print the zero count statistics, e.g. "0 (0)"

`missing_location`  

Location where the column summarizing values with missing levels by= will be located in the final table. Must be one of c("first", "last", "hide"). Default is "first"

**Details**

`tbl_ae_count` counts all AEs (whereas `tbl_ae` counts by maximum grade). Thus, `tbl_ae_count` does not provide percentages as multiple AEs can be counted per subject.

**Value**

A 'tbl_ae_count' object

**Example Output**

**See Also**

`tbl_ae`

**Examples**

```r
# Example 1 -------------------------------------------------------------
tbl_ae_count_ex1 <-
tbl_ae_count(
  data = df_adverse_events,
  ae = adverse_event,
  soc = system_organ_class,
  strata = trt,
  by = grade
) %>%
  modify_header(all_ae_cols() ~ "**Grade {by}**")
```

---

<table>
<thead>
<tr>
<th><code>tbl_ae_focus</code></th>
<th><strong>Tabulate AE Focused (Dichotomous) Summaries</strong></th>
</tr>
</thead>
</table>

**Description**

Summarize dichotomous AE data. For example, report the rate of patients that have an AE of Grade 3 or higher.
Usage

```r
tbl_ae_focus(
  data,
  include,
  id,
  ae,
  soc = NULL,
  strata = NULL,
  label = NULL,
  id_df = NULL,
  statistic = "{n} ({p})",
  digits = NULL,
  sort = NULL,
  zero_symbol = "\U2014"
)
```

Arguments

data          Data frame
include       Vector of column names to summarize. Column names may be quoted or un-
              quoted. All columns must be class 'logical'.
id            Variable name of the patient ID
ae            Variable name of the adverse event column
soc           Variable name of the system organ class column
strata        Variable to stratify results by, e.g. report AEs summaries by treatment group
label         A named list of labels that will be applied in the resulting table. Names must be
              those passed in `include=`. Default is NULL, and either the label attribute or the
              column name will be used.
id_df         Optional data frame of complete id values and strata to achieve correct base
              n for the situation in which not all subjects experience adverse events. See
              `df_patient_characteristics` for an example `id_df` that pairs with `df_adverse_events`.
statistic     String indicating the statistics that will be reported. The default is "\{n\} \{(p)\"
digits        Specifies the number of decimal places to round the summary statistics. By
              default integers are shown to zero decimal places, and percentages are formatted
              with `style_percent()`. If you would like to modify either of these, pass a
              vector of integers indicating the number of decimal places to round the statistics.
              For example, if the statistic being calculated is "\{n\} \{(p)\%\}" and you want the
              percent rounded to 2 decimal places use `digits = c(0, 2)`. User may also pass
              a styling function: `digits = style_sigfig`
sort          Controls order of AEs and SOCs in output table. The default is NULL, where
              AEs and SOCs are sorted alpha-numerically (and factors sorted according to their
              factor level). Use `sort = "ae"` to sort AEs in decreasing frequency order, `sort = "soc"
              ` to sort SOCs in decreasing order, and `sort = c("ae", "soc")` to sort
              both. AEs are sorted within SOC.
zero_symbol   String used to represent cells with zero counts. Default is the em-dash ("\U2014").
              Using `zero_symbol = NULL` will print the zero count statistics, e.g. "0 (0)"
Value

a `tbl_ae_focus` object

Example Output

Examples

```r
# Example 1  ________________________________________________________________
tbl_ae_focus_ex1 <-
df_adverse_events %>%
tbl_ae_focus(
  include = c(any_complication, grade3_complication),
  id = patient_id,
  ae = adverse_event,
  soc = system_organ_class,
  label =
    list(any_complication = "Any Grade Complication",
         grade3_complication = "Grade 3+ Complication")
) %>%
bold_labels()
```

---

tbl_listing & Data Listing Table

Description

Function creates a gtsummary-class listing of data. Column labels are used as column headers, when present. The listing prints observations in the order of the input data.

Usage

tbl_listing(data, group_by = NULL, bold_headers = TRUE)

Arguments

data a data frame
group_by Single variable name indicating a grouping variable. Default is NULL for no grouping variable. When specified, a grouping row will be added to the first column. See details below.
bold_headers logical indicating whether to bold column headers. Default is TRUE

Value
gtsummary data listing
group_by

The grouping column and the first column in the table will be combined and the type/class may be converted to common type/class for both columns. However, if either the group_by= column or the first column are factors, the factor column(s) will first be converted to character.

The groups are ordered according to the grouping variable’s type (i.e., character, numeric, or factor).

Details

The purpose of tbl_listing() is to add support for printing data frames, while taking advantage of the {gtsummary} defaults, e.g. ability to print to most output formats, using print themes to have a common style to all tables in a document, etc.

While the output of tbl_listing() is class 'gtsummary', these tables are not meant to be merged with other 'gtsummary' tables with tbl_merge(), or reporting table contents with inline_text(). The reason is that a proper 'gtsummary' contains additional hidden structure not present in the result of tbl_listing(). If you do need to report the results of tbl_listing() in-line, it’s recommended to convert the table to a data frame, then extract the needed cell, e.g.

```r
tbl_listing() |> as_tibble(col_names = FALSE) |> dplyr::slice(1) |> dplyr::pull(colname)```

Example Output

Examples

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

tbl_listing_ex1 <-
  head(df_adverse_events, n = 10) %>%
  select(system_organ_class, adverse_event, grade, drug_attribution, patient_id) %>%
  arrange(adverse_event, desc(grade)) %>%
  tbl_listing(group_by = system_organ_class) %>%
  bold_labels()

set.seed(11234)
tbl_listing_ex2 <-
df_patient_characteristics %>%
  dplyr::slice_sample(n = 10) %>%
  select(patient_id, status, discontinued, off_trt_ae) %>%
  tbl_listing() %>%
  as_gt() %>%
  gt::opt_row_striping()```
Data Summary Table

Description

Function wraps `gtsummary::tbl_summary()` to create a data summary table often seen in regulatory submissions. Continuous variable summaries are shown on multiple lines with additional summary statistics and percentages are shown for categorical variables; precision levels estimated based on values observed.

Usage

```r
tbl_reg_summary(
  data,
  by = NULL,
  label = NULL,
  statistic = NULL,
  digits = NULL,
  type = NULL,
  value = NULL,
  missing = c("no", "yes", "ifany"),
  missing_text = NULL,
  sort = NULL,
  percent = NULL,
  include = everything()
)
```

Arguments

- **data**: A data frame
- **by**: A column name (quoted or unquoted) in `data`. Summary statistics will be calculated separately for each level of the `by` variable (e.g. `by = trt`). If `NULL`, summary statistics are calculated using all observations.
- **label**: List of formulas specifying variables labels, e.g. `list(age ~ "Age", stage ~ "Path T Stage")`. If a variable’s label is not specified here, the label attribute `attr(data$age, "label")` is used. If attribute label is `NULL`, the variable name will be used.
- **statistic**: List of formulas specifying types of summary statistics to display for each variable.
- **digits**: List of formulas specifying the number of decimal places to round summary statistics. If not specified, `tbl_summary` guesses an appropriate number of decimals to round statistics. When multiple statistics are displayed for a single variable, supply a vector rather than an integer. For example, if the statistic being calculated is `{mean} (`{sd}`)` and you want the mean rounded to 1 decimal place, and the SD to 2 use `digits = list(age ~ c(1, 2))`. User may also pass a styling function: `digits = age ~ style_sigfig`
**type**  
List of formulas specifying variable types. Accepted values are `c("continuous", "continuous2", "categorical", "dichotomous")`. e.g. `type = list(age ~ "continuous", female ~ "dichotomous")`. If type not specified for a variable, the function will default to an appropriate summary type.

**value**  
List of formulas specifying the value to display for dichotomous variables. `gtsummary` selectors, e.g. `all_dichotomous()`, cannot be used with this argument.

**missing**  
Indicates whether to include counts of NA values in the table. Allowed values are "no" (never display NA values), "ifany" (only display if any NA values), and "always" (includes NA count row for all variables). Default is "ifany".

**missing_text**  
String to display for count of missing observations. Default is "Unknown".

**sort**  
List of formulas specifying the type of sorting to perform for categorical data. Options are `frequency` where results are sorted in descending order of frequency and `alphanumeric`, e.g. `sort = list(everything() ~ "frequency")`

**percent**  
Indicates the type of percentage to return. Must be one of "column", "row", or "cell". Default is "column".

**include**  
variables to include in the summary table. Default is `everything()`

**Value**

a ‘tbl_reg_summary’ object

**Example Output**

**See Also**

See `gtsummary::tbl_summary()` help file

See vignette for detailed tutorial

**Examples**

```r
tbl_reg_summary_ex1 <-
df_patient_characteristics %>%
tbl_reg_summary(by = trt, include = c(marker, status))
```
Index

* datasets
  - df_adverse_events, 5
  - df_patient_characteristics, 6
  - complete_ae_data, 2

add_overall.tbl_ae
  (add_overall_tbl_ae), 3
add_overall.tbl_ae_count
  (add_overall_tbl_ae), 3
add_overall.tbl_ae_focus
  (add_overall_tbl_ae), 3
add_overall_tbl_ae, 3
all_ae_cols (selectors), 8
all_cols_in_strata (selectors), 8
all_overall_cols (selectors), 8
all_unknown_cols (selectors), 8

df_adverse_events, 5, 10, 14
df_patient_characteristics, 6, 10, 14

inline_text.tbl_ae
  (inline_text_tbl_ae), 7
inline_text.tbl_ae_count
  (inline_text_tbl_ae), 7
inline_text.tbl_ae_focus
  (inline_text_tbl_ae), 7
inline_text_tbl_ae, 7
selectors, 8
style_xxx, 9

tbl_ae, 10, 13
tbl_ae_count, 12
tbl_ae_focus, 13
tbl_listing, 15
tbl_reg_summary, 17