

---

# Introducing pmtables

**Author:** Metrum Research Group

**Date:** 2021-03-06

**Version:** 0.3.2.9100

**Introduction:** This is a simple introduction to the pmtables package for R. I hope this will be useful for those who are new to the package and those who just need a reminder on the syntax.

---

## Contents

<b>1</b>	<b>A simple table: stable</b>	<b>4</b>
1.1	Syntax	4
1.2	Basics	4
1.3	Annotate with file names	5
1.4	Saving your stable	6
1.5	Align columns	7
1.5.1	Fixed column widths	8
1.6	Manipulating columns and names	8
1.6.1	Rename columns	8
1.6.2	Hide a column name	8
1.6.3	Unmask column names	9
1.6.4	Make column names bold	9
1.6.5	Drop a column from the table	9
1.7	Other customizations	10
1.7.1	Notes	10
1.7.2	Units	11
1.7.3	Multi-line column headers	11
1.7.4	Insert horizontal lines	12
<b>2</b>	<b>Group table rows with panel</b>	<b>14</b>
2.1	Syntax	14
2.2	Basics	14
2.3	panel: important points	15
<b>3</b>	<b>Group table columns with spanners</b>	<b>16</b>
3.1	Syntax	16
3.2	Basics	16
3.3	Multiple spanners	16
3.3.1	Using pipe syntax	17
3.4	Breaking span title	17

<b>4</b>	<b>Tables that span multiple pages: longtable</b>	<b>18</b>
4.1	Syntax . . . . .	18
4.2	Basics . . . . .	18
4.3	Inserting longtable into your latex document . . . . .	18
<b>5</b>	<b>The pipe interface</b>	<b>19</b>
5.1	Basics . . . . .	19
5.2	Simple table . . . . .	20
5.3	Long table . . . . .	20
5.4	Annotate with file names . . . . .	20
5.5	Save a table . . . . .	20
5.6	Align columns . . . . .	20
5.7	Rename columns . . . . .	21
5.8	Blank columns . . . . .	21
5.9	Drop columns . . . . .	21
5.10	Notes . . . . .	21
5.11	Units . . . . .	22
5.12	panel . . . . .	22
5.13	span . . . . .	22
5.14	hlines - at . . . . .	23
5.15	hlines - from . . . . .	23
5.16	hlines - pattern . . . . .	23
<b>6</b>	<b>Options for previewing the table</b>	<b>24</b>
6.1	st2report . . . . .	24
6.2	st2viewer . . . . .	24
6.3	st2article . . . . .	24
6.4	st2doc . . . . .	24
<b>7</b>	<b>A word about sanitizing table contents</b>	<b>25</b>
7.1	Notes . . . . .	25
7.2	File names . . . . .	25
7.3	Column names . . . . .	25
7.4	Main table contents . . . . .	25
7.5	Span titles . . . . .	26
7.6	Panel names . . . . .	26
7.7	cols_extra input . . . . .	26

<b>8</b>	<b>pmtable</b>	<b>27</b>
8.1	Principles	27
8.2	Rename cols	27
8.3	Data inventory tables	27
8.3.1	Stacked by endpoint	29
8.3.2	Paneled	30
8.3.3	Grouped (by study)	31
8.4	Wide categorical table	32
8.4.1	Paneled (limited utility, IMO)	33
8.4.2	Grouped (by male / female)	34
8.4.3	Paneled and grouped	35
8.4.4	No summary	36
8.5	Long categorical table	37
8.5.1	Ungrouped	37
8.5.2	Grouped (by formulation)	38
8.5.3	Summary on bottom and right	39
8.5.4	No summary	40
8.6	Wide continuous table	41
8.6.1	Ungrouped	41
8.6.2	Paneled	41
8.6.3	Grouped (by study)	43
8.6.4	Paneled and grouped	44
8.7	Long continuous table	45
8.7.1	Ungrouped	45
8.7.2	Paneled	46

# 1 A simple table: stable

## 1.1 Syntax

Pass your `data.frame` into `stable()`

```
stable(data)
```

Other formal arguments include:

- **align** to set column alignment
- **panel** to create groups of rows under a “panel” header
- **span** to group columns under a “spanner” header
- **notes** to create table notes
- **sumrows** to insert summary rows
- **units** that get placed below the corresponding column name
- **drop** to omit certain columns from the table
- **sizes** to set different table size attributes
- **escape\_fun** a function to sanitize table items

You can also pass a bunch of other arguments through `...` to further format the table (see `?stable` for details)

## 1.2 Basics

`stable()` is the name of the workhorse function that is used to turn `data.frames` into TeX tables. This chapter will introduce the `stable()` function and how to use it to create basic tables.

To illustrate usage and features of `stable()`, we will use the `stdata` data set that comes with `pmtables`

```
data <- stdata()
```

```
head(data)
```

```
. # A tibble: 6 x 9
.   STUDY      DOSE  FORM    N    WT   CRCL  AGE  ALB  SCR
.   <chr>      <chr> <chr>  <chr> <chr> <chr> <chr> <chr> <chr>
. 1 12-DEMO-001 100 mg tablet  80   71.4  104   33.7  4.20  1.06
. 2 12-DEMO-001 150 mg capsule 16   89.4  122   24.4  4.63  1.12
. 3 12-DEMO-001 150 mg tablet  48   81.7  104   34.4  3.83  0.910
. 4 12-DEMO-001 150 mg troche  16   94.0  93.2   27.4  4.94  1.25
. 5 12-DEMO-001 200 mg tablet  64   67.9  100   27.5  4.25  1.10
. 6 12-DEMO-001 200 mg troche  16   76.6  99.2   22.8  4.54  1.15
```

We can turn this data frame into a TeX table by passing it into `stable()`.

```
out <- stable(data)
```

```
head(out, n = 10)
```

```

. [1] "\\setlength{\\tabcolsep}{5pt} "
. [2] "\\begin{threeparttable}"
. [3] "\\renewcommand{\\arraystretch}{1.3}"
. [4] "\\begin{tabular}[h]{l}{l11111111}"
. [5] "\\hline"
. [6] "STUDY & DOSE & FORM & N & WT & CRCL & AGE & ALB & SCR \\\\"
. [7] "\\hline"
. [8] "12-DEMO-001 & 100 mg & tablet & 80 & 71.4 & 104 & 33.7 & 4.20 & 1.06 \\\\"
. [9] "12-DEMO-001 & 150 mg & capsule & 16 & 89.4 & 122 & 24.4 & 4.63 & 1.12 \\\\"
. [10] "12-DEMO-001 & 150 mg & tablet & 48 & 81.7 & 104 & 34.4 & 3.83 & 0.910 \\\\"

```

Note that we have shown the raw latex code that is generated by `stable()`. That is to say: the output from `stable()` is a character vector of latex code for the table. Note also that this character vector has a special class associated with it: `stable`. That means we can write functions that recognize this character vector as output from `stable()` and we can have those functions process the character vector in special ways.

We can render that table in TeX **in the current Rmarkdown document** by passing the text to `st_asis()`.

```
out %>% st_asis()
```

STUDY	DOSE	FORM	N	WT	CRCL	AGE	ALB	SCR
12-DEMO-001	100 mg	tablet	80	71.4	104	33.7	4.20	1.06
12-DEMO-001	150 mg	capsule	16	89.4	122	24.4	4.63	1.12
12-DEMO-001	150 mg	tablet	48	81.7	104	34.4	3.83	0.910
12-DEMO-001	150 mg	troche	16	94.0	93.2	27.4	4.94	1.25
12-DEMO-001	200 mg	tablet	64	67.9	100	27.5	4.25	1.10
12-DEMO-001	200 mg	troche	16	76.6	99.2	22.8	4.54	1.15
12-DEMO-002	100 mg	capsule	36	61.3	113	38.3	4.04	1.28
12-DEMO-002	100 mg	tablet	324	77.6	106	29.9	4.31	0.981
12-DEMO-002	50 mg	capsule	36	74.1	112	37.1	4.44	0.900
12-DEMO-002	50 mg	tablet	324	71.2	106	34.1	4.63	0.868
12-DEMO-002	75 mg	capsule	36	72.4	105	38.2	3.89	0.900
12-DEMO-002	75 mg	tablet	288	71.6	98.9	34.2	4.49	0.991
12-DEMO-002	75 mg	troche	36	73.6	103	49.2	4.52	0.930

Remember to only call `st_asis()` when you are rendering tables inline in an Rmd document. If you are sending table code to a TeX report, then you will save them to a file and then include them into your report.

The remaining sections of this chapter will show you how to modify and enhance this output in the more basic ways. We will implement separate chapters for more complicated table manipulations.

### 1.3 Annotate with file names

`pmtables` can track and annotate your table with the filenames of the R code that generated the table (`r_file`) as well as the output file where you write the the table `.tex` code (`output_file`).

To have `pmtables` annotate your table with these file names, pass them in with the `r_file` and `output_file` arguments

```
out <- stable(data, r_file = "tables.R", output_file = "tables.tex")
```

When we look at the rendered table, these names will show up as annotations at the bottom of the table

```
out %>% st_asis()
```

STUDY	DOSE	FORM	N	WT	CRCL	AGE	ALB	SCR
12-DEMO-001	100 mg	tablet	80	71.4	104	33.7	4.20	1.06
12-DEMO-001	150 mg	capsule	16	89.4	122	24.4	4.63	1.12
12-DEMO-001	150 mg	tablet	48	81.7	104	34.4	3.83	0.910
12-DEMO-001	150 mg	troche	16	94.0	93.2	27.4	4.94	1.25
12-DEMO-001	200 mg	tablet	64	67.9	100	27.5	4.25	1.10
12-DEMO-001	200 mg	troche	16	76.6	99.2	22.8	4.54	1.15
12-DEMO-002	100 mg	capsule	36	61.3	113	38.3	4.04	1.28
12-DEMO-002	100 mg	tablet	324	77.6	106	29.9	4.31	0.981
12-DEMO-002	50 mg	capsule	36	74.1	112	37.1	4.44	0.900
12-DEMO-002	50 mg	tablet	324	71.2	106	34.1	4.63	0.868
12-DEMO-002	75 mg	capsule	36	72.4	105	38.2	3.89	0.900
12-DEMO-002	75 mg	tablet	288	71.6	98.9	34.2	4.49	0.991
12-DEMO-002	75 mg	troche	36	73.6	103	49.2	4.52	0.930

Source code: tables.R

Source file: tables.tex

## 1.4 Saving your stable

Saving your stable **can** be as easy as sending it into `writeLines()`

```
writeLines(out, con = tempfile(tmpdir = '.', fileext = ".tex"))
```

But remember that we passed in the `output_file` argument to `stable()` and we can use that data to save the table code to the file we named in that argument.

Note that our `stable` object has another attribute now called `stable_file`

```
attributes(out)
```

```
. $class
. [1] "stable"
.
. $stable_file
. [1] "tables.tex"
```

This has the value that we passed in as `output_file`. To save our table to `stable_file`, we call `stable_save()`

```
stable_save(out)
```

There is a `dir` argument to `stable_save()` that we can use to select the directory where the file will be saved

```
stable_save(out, dir = tempdir())
```

And if you look at the default value for `dir` in `?stable_save`, you'll see that this is associated with an option called `pmtables.dir`; you can set that option to your default output directory and your tables will be saved there until you change that

```
options(pmtables.dir = tempdir())
stable_save(out)
```

## 1.5 Align columns

Use the `align` argument to align column data to the left, center or right. Use a `cols_*` function to specify the default alignment for all columns

```
tmp <- tibble(AB = 1, CDEFGHIJ = 2, KL = 3)
stable(tmp, align = cols_center()) %>% st_asis()
```

AB	CDEFGHIJ	KL
1	2	3

You can pass in exceptions to the default

```
stable(tmp, align = cols_center(CDEFGHIJ = "r")) %>% st_asis()
```

AB	CDEFGHIJ	KL
1	2	3

Or you can pass an alignment directive and the columns that are bound by that directive

```
stable(tmp, align = cols_center(.1 = "AB,KL")) %>% st_asis()
```

AB	CDEFGHIJ	KL
1	2	3

A special directive called `.outer` lets you specify the alignment of the first and last column in the table. For example, this code puts the first column to the left and the last column to the right.

```
stable(tmp, align = cols_center(.outer = "lr")) %>% st_asis()
```

AB	CDEFGHIJ	KL
1	2	3

### 1.5.1 Fixed column widths

Use `col_ragged(size)` to force a column to be a fixed size.

```
stable(tmp, align = cols_center(AB = col_ragged(2))) %>% st_asis()
```

AB	CDEFGHIJ	KL
1	2	3

By default, the unit is cm so that the first column (AB) has a width of 2 cm regardless of the contents.

See `cols_align()` help topic for more information and argument descriptions.

## 1.6 Manipulating columns and names

### 1.6.1 Rename columns

You can change the name that appears in the rendered table with `cols_rename`

```
data %>%
  slice(1:3) %>%
  stable(cols_rename = c(Age = "AGE", Weight = "WT")) %>%
  st_asis()
```

STUDY	DOSE	FORM	N	Weight	CRCL	Age	ALB	SCR
12-DEMO-001	100 mg	tablet	80	71.4	104	33.7	4.20	1.06
12-DEMO-001	150 mg	capsule	16	89.4	122	24.4	4.63	1.12
12-DEMO-001	150 mg	tablet	48	81.7	104	34.4	3.83	0.910

Note that the rename syntax follows the tidyselect convention of putting the new name on the left and the old name on the right.

### 1.6.2 Hide a column name

You can also “erase” the name of a column in the output

```
data %>%
  slice(1:3) %>%
  stable(cols_blank = "WT,ALB,SCR") %>%
  st_asis()
```



STUDY	DOSE	FORM	N	WT	CRCL	AGE	ALB	SCR
12-DEMO-001	100 mg	tablet	80	71.4	104	33.7	4.20	1.06
12-DEMO-001	150 mg	capsule	16	89.4	122	24.4	4.63	1.12
12-DEMO-001	150 mg	tablet	48	81.7	104	34.4	3.83	0.910

### 1.6.3 Unmask column names

In tibbles, you can't have duplicate column names. The `cols_split` argument lets you unmask the names when duplicate names are prefixed with a tag and a delimiter

```
tmp <- tibble(a.A = 1, b.A = 2, c.A = 3)
```

```
tmp %>% stable(cols_split = '.') %>% st_asis()
```

A	A	A
1	2	3

### 1.6.4 Make column names bold

```
data %>% slice(1:2) %>% stable(cols_bold = TRUE) %>% st_asis()
```

<b>STUDY</b>	<b>DOSE</b>	<b>FORM</b>	<b>N</b>	<b>WT</b>	<b>CRCL</b>	<b>AGE</b>	<b>ALB</b>	<b>SCR</b>
12-DEMO-001	100 mg	tablet	80	71.4	104	33.7	4.20	1.06
12-DEMO-001	150 mg	capsule	16	89.4	122	24.4	4.63	1.12

### 1.6.5 Drop a column from the table

If we want to prevent a column from appearing in the output table (e.g. FORM)

```
head(data)
```

```
. # A tibble: 6 x 9
.   STUDY      DOSE  FORM    N    WT  CRCL  AGE  ALB  SCR
.   <chr>      <chr> <chr>  <chr> <chr> <chr> <chr> <chr> <chr>
. 1 12-DEMO-001 100 mg tablet  80   71.4  104   33.7  4.20  1.06
. 2 12-DEMO-001 150 mg capsule 16   89.4  122   24.4  4.63  1.12
. 3 12-DEMO-001 150 mg tablet  48   81.7  104   34.4  3.83  0.910
. 4 12-DEMO-001 150 mg troche 16   94.0  93.2  27.4  4.94  1.25
. 5 12-DEMO-001 200 mg tablet  64   67.9  100   27.5  4.25  1.10
. 6 12-DEMO-001 200 mg troche 16   76.6  99.2  22.8  4.54  1.15
```

list the column name as drop

```
stable(data, drop = "FORM") %>% st_asis()
```

STUDY	DOSE	N	WT	CRCL	AGE	ALB	SCR
12-DEMO-001	100 mg	80	71.4	104	33.7	4.20	1.06
12-DEMO-001	150 mg	16	89.4	122	24.4	4.63	1.12
12-DEMO-001	150 mg	48	81.7	104	34.4	3.83	0.910
12-DEMO-001	150 mg	16	94.0	93.2	27.4	4.94	1.25
12-DEMO-001	200 mg	64	67.9	100	27.5	4.25	1.10
12-DEMO-001	200 mg	16	76.6	99.2	22.8	4.54	1.15
12-DEMO-002	100 mg	36	61.3	113	38.3	4.04	1.28
12-DEMO-002	100 mg	324	77.6	106	29.9	4.31	0.981
12-DEMO-002	50 mg	36	74.1	112	37.1	4.44	0.900
12-DEMO-002	50 mg	324	71.2	106	34.1	4.63	0.868
12-DEMO-002	75 mg	36	72.4	105	38.2	3.89	0.900
12-DEMO-002	75 mg	288	71.6	98.9	34.2	4.49	0.991
12-DEMO-002	75 mg	36	73.6	103	49.2	4.52	0.930

Of course some tidyverse could accomplish the same thing

```
data %>% select(-FORM) %>% stable()
```

## 1.7 Other customizations

### 1.7.1 Notes

Arbitrary notes can get added to any table using the `notes` argument.

```
data %>%
  slice(1:3) %>%
  stable(notes = "Showing just the first three rows") %>%
  st_asis()
```

STUDY	DOSE	FORM	N	WT	CRCL	AGE	ALB	SCR
12-DEMO-001	100 mg	tablet	80	71.4	104	33.7	4.20	1.06
12-DEMO-001	150 mg	capsule	16	89.4	122	24.4	4.63	1.12
12-DEMO-001	150 mg	tablet	48	81.7	104	34.4	3.83	0.910

Showing just the first three rows

The appearance of the notes can be controlled by calling `noteconf()` and passing the result as `note_config`. See `?tab_notes()` for more details.

### 1.7.2 Units

pmtables can automatically place units underneath the appropriate column. To do this, generate a list with names that match the column names you want to label with units.

```
u <- list(
  WT = "kg", CRCL = "ml/min", AGE = "year", ALB = "g/dL",
  SCR = "mg\\%"
) %>% map(~paste0("(", .x, "%"))
```

Then pass that list as units to `stable()`

```
stable(data, units = u) %>% st_asis()
```

STUDY	DOSE	FORM	N	WT (kg)	CRCL (ml/min)	AGE (year)	ALB (g/dL)	SCR (mg%)
12-DEMO-001	100 mg	tablet	80	71.4	104	33.7	4.20	1.06
12-DEMO-001	150 mg	capsule	16	89.4	122	24.4	4.63	1.12
12-DEMO-001	150 mg	tablet	48	81.7	104	34.4	3.83	0.910
12-DEMO-001	150 mg	troche	16	94.0	93.2	27.4	4.94	1.25
12-DEMO-001	200 mg	tablet	64	67.9	100	27.5	4.25	1.10
12-DEMO-001	200 mg	troche	16	76.6	99.2	22.8	4.54	1.15
12-DEMO-002	100 mg	capsule	36	61.3	113	38.3	4.04	1.28
12-DEMO-002	100 mg	tablet	324	77.6	106	29.9	4.31	0.981
12-DEMO-002	50 mg	capsule	36	74.1	112	37.1	4.44	0.900
12-DEMO-002	50 mg	tablet	324	71.2	106	34.1	4.63	0.868
12-DEMO-002	75 mg	capsule	36	72.4	105	38.2	3.89	0.900
12-DEMO-002	75 mg	tablet	288	71.6	98.9	34.2	4.49	0.991
12-DEMO-002	75 mg	troche	36	73.6	103	49.2	4.52	0.930

### 1.7.3 Multi-line column headers

If the column header is long, you can break it across multiple lines. By default, use `...` in the column name

```
tibble(`First line ... Second line` = 123456789) %>%
  stable() %>% st_asis()
```

First line
Second line
123456789

The break can be introduced through the rename mechanism

```
tibble(a = 1) %>%
  stable(cols_rename = c(`First ... Second` = "a")) %>%
  st_asis()
```

First Second
1

Look at the `?tab_cols` help topic for the `cols_break` argument; this lets you change the character sequence used for the break.

#### 1.7.4 Insert horizontal lines

Pass `hlines_at` to insert horizontal lines above specific rows. This can be either logical vector with the same length as the number of rows in the table or a vector of integers.

```
stable(stdata(), hline_at = c(3,5)) %>% st_asis()
```

STUDY	DOSE	FORM	N	WT	CRCL	AGE	ALB	SCR
12-DEMO-001	100 mg	tablet	80	71.4	104	33.7	4.20	1.06
12-DEMO-001	150 mg	capsule	16	89.4	122	24.4	4.63	1.12
12-DEMO-001	150 mg	tablet	48	81.7	104	34.4	3.83	0.910
12-DEMO-001	150 mg	troche	16	94.0	93.2	27.4	4.94	1.25
12-DEMO-001	200 mg	tablet	64	67.9	100	27.5	4.25	1.10
12-DEMO-001	200 mg	troche	16	76.6	99.2	22.8	4.54	1.15
12-DEMO-002	100 mg	capsule	36	61.3	113	38.3	4.04	1.28
12-DEMO-002	100 mg	tablet	324	77.6	106	29.9	4.31	0.981
12-DEMO-002	50 mg	capsule	36	74.1	112	37.1	4.44	0.900
12-DEMO-002	50 mg	tablet	324	71.2	106	34.1	4.63	0.868
12-DEMO-002	75 mg	capsule	36	72.4	105	38.2	3.89	0.900
12-DEMO-002	75 mg	tablet	288	71.6	98.9	34.2	4.49	0.991
12-DEMO-002	75 mg	troche	36	73.6	103	49.2	4.52	0.930

or

```
stable(stdata(), hline_at = data$FORM == "tablet") %>% st_asis()
```

STUDY	DOSE	FORM	N	WT	CRCL	AGE	ALB	SCR
12-DEMO-001	100 mg	tablet	80	71.4	104	33.7	4.20	1.06
12-DEMO-001	150 mg	capsule	16	89.4	122	24.4	4.63	1.12
12-DEMO-001	150 mg	tablet	48	81.7	104	34.4	3.83	0.910
12-DEMO-001	150 mg	troche	16	94.0	93.2	27.4	4.94	1.25
12-DEMO-001	200 mg	tablet	64	67.9	100	27.5	4.25	1.10
12-DEMO-001	200 mg	troche	16	76.6	99.2	22.8	4.54	1.15
12-DEMO-002	100 mg	capsule	36	61.3	113	38.3	4.04	1.28
12-DEMO-002	100 mg	tablet	324	77.6	106	29.9	4.31	0.981
12-DEMO-002	50 mg	capsule	36	74.1	112	37.1	4.44	0.900
12-DEMO-002	50 mg	tablet	324	71.2	106	34.1	4.63	0.868
12-DEMO-002	75 mg	capsule	36	72.4	105	38.2	3.89	0.900
12-DEMO-002	75 mg	tablet	288	71.6	98.9	34.2	4.49	0.991
12-DEMO-002	75 mg	troche	36	73.6	103	49.2	4.52	0.930

Pass `hlines_from` to derive hline locations based on non-repeating values in a table column. Notice how this behaves.

```
stable(stdata(), hline_from = "DOSE") %>% st_asis()
```

STUDY	DOSE	FORM	N	WT	CRCL	AGE	ALB	SCR
12-DEMO-001	100 mg	tablet	80	71.4	104	33.7	4.20	1.06
12-DEMO-001	150 mg	capsule	16	89.4	122	24.4	4.63	1.12
12-DEMO-001	150 mg	tablet	48	81.7	104	34.4	3.83	0.910
12-DEMO-001	150 mg	troche	16	94.0	93.2	27.4	4.94	1.25
12-DEMO-001	200 mg	tablet	64	67.9	100	27.5	4.25	1.10
12-DEMO-001	200 mg	troche	16	76.6	99.2	22.8	4.54	1.15
12-DEMO-002	100 mg	capsule	36	61.3	113	38.3	4.04	1.28
12-DEMO-002	100 mg	tablet	324	77.6	106	29.9	4.31	0.981
12-DEMO-002	50 mg	capsule	36	74.1	112	37.1	4.44	0.900
12-DEMO-002	50 mg	tablet	324	71.2	106	34.1	4.63	0.868
12-DEMO-002	75 mg	capsule	36	72.4	105	38.2	3.89	0.900
12-DEMO-002	75 mg	tablet	288	71.6	98.9	34.2	4.49	0.991
12-DEMO-002	75 mg	troche	36	73.6	103	49.2	4.52	0.930

See the `?tab_hlines` help topic for more info. See also `st_hline()` for the pipe equivalent with additional feature.

## 2 Group table rows with panel

### 2.1 Syntax

To panel a table by STUDY

```
stable(stdata(), panel = "STUDY")
```

To set a prefix for the panel header:

```
stable(stdata(), panel = as.panel("STUDY", prefix = "Study: "))
```

### 2.2 Basics

Paneling your table is a way to group sets of rows together into a “panel” with a panel header rendered in bold font. For example, we can panel a table of mtcars by carb. We will be working with an abbreviated version of mtcars:

smcars

```
.
      name mpg cyl  disp  hp drat   wt  qsec vs am gear
. Datsun 710      Datsun 710 22.8   4 108.0  93 3.85 2.320 18.61  1  1    4
. Hornet 4 Drive Hornet 4 Drive 21.4   6 258.0 110 3.08 3.215 19.44  1  0    3
. Valiant          Valiant 18.1   6 225.0 105 2.76 3.460 20.22  1  0    3
. Fiat 128          Fiat 128 32.4   4  78.7  66 4.08 2.200 19.47  1  1    4
. Toyota Corolla Toyota Corolla 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4
. Toyota Corona   Toyota Corona 21.5   4 120.1  97 3.70 2.465 20.01  1  0    3
. Fiat X1-9        Fiat X1-9 27.3   4  79.0  66 4.08 1.935 18.90  1  1    4
. Merc 240D        Merc 240D 24.4   4 146.7  62 3.69 3.190 20.00  1  0    4
. Merc 230         Merc 230 22.8   4 140.8  95 3.92 3.150 22.90  1  0    4
. Honda Civic      Honda Civic 30.4   4  75.7  52 4.93 1.615 18.52  1  1    4
.
      carb
. Datsun 710      1
. Hornet 4 Drive  1
. Valiant          1
. Fiat 128          1
. Toyota Corolla   1
. Toyota Corona    1
. Fiat X1-9         1
. Merc 240D         2
. Merc 230          2
. Honda Civic       2
```

Then we pass into `stable()` and name the paneling column:

```
smcars %>% stable(panel = "carb") %>% st_asis()
```

name	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear
<b>1</b>										
Datsun 710	22.8	4	108	93	3.85	2.32	18.61	1	1	4
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3
Valiant	18.1	6	225	105	2.76	3.46	20.22	1	0	3
Fiat 128	32.4	4	78.7	66	4.08	2.2	19.47	1	1	4
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.9	1	1	4
Toyota Corona	21.5	4	120.1	97	3.7	2.465	20.01	1	0	3
Fiat X1-9	27.3	4	79	66	4.08	1.935	18.9	1	1	4
<b>2</b>										
Merc 240D	24.4	4	146.7	62	3.69	3.19	20	1	0	4
Merc 230	22.8	4	140.8	95	3.92	3.15	22.9	1	0	4
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4

Now, all of the `carb==1` rows are grouped with the heading 1 and similarly the `carb==2` rows are grouped with the heading 2 in bold.

This is ok, but a more informative heading would be helpful. To do this, we'll call `as.panel()` to both name the panel column and set some options:

```
smcars %>% stable(panel = as.panel("carb", prefix = "carb: ")) %>% st_asis()
```

name	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear
<b>carb: 1</b>										
Datsun 710	22.8	4	108	93	3.85	2.32	18.61	1	1	4
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3
Valiant	18.1	6	225	105	2.76	3.46	20.22	1	0	3
Fiat 128	32.4	4	78.7	66	4.08	2.2	19.47	1	1	4
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.9	1	1	4
Toyota Corona	21.5	4	120.1	97	3.7	2.465	20.01	1	0	3
Fiat X1-9	27.3	4	79	66	4.08	1.935	18.9	1	1	4
<b>carb: 2</b>										
Merc 240D	24.4	4	146.7	62	3.69	3.19	20	1	0	4
Merc 230	22.8	4	140.8	95	3.92	3.15	22.9	1	0	4
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4

Note that the prefix is completely specified by the user (including any spaces or a colon).

## 2.3 panel: important points

1. Most of the time, the data frame should be sorted by the `panel` column
2. `pmtables` creates panels by non-repeating values in the `panel` column; there will be an error if duplicate panel names are found and this can be overridden by passing `duplicates_ok` to `as.panel()`.

### 3 Group table columns with spanners

#### 3.1 Syntax

Set the `span` argument to the output of `as.span()`. The key arguments for `as.span()` are the `spanner title` and the names of the columns over which you want the `spanner` to run

```
stable(stdata(), span = as.span("Covariates", WT:SCR))
```

The equivalent pipe syntax is

```
st_new(stdata()) %>% st_span("Covariates", WT:SCR)
```

#### 3.2 Basics

A column `spanner` puts a horizontal line over a sequence of column names and places a title above that line forming a column group.

As a trivial example:

```
data <-
  tibble(
    Tariffville = "06081", Connecticut = "CT",
    Minnesota = "MN", Minneapolis = "55455"
  )

data %>% stable(span = as.span("States", Connecticut:Minnesota)) %>% st_asis()
```

	States		
Tariffville	Connecticut	Minnesota	Minneapolis
06081	CT	MN	55455

#### 3.3 Multiple spanners

Multiple `spanners` can be added to a table by specifying the `level` for any `spanner` that you want to be placed above the lowest level `spanner`. For example,

```
sp <- list(
  as.span("States", Connecticut:Minnesota),
  as.span("The Universe", Tariffville:Minneapolis, level = 2)
)

data %>% stable(span = sp) %>% st_asis()
```

	The Universe		
	States		
Tariffville	Connecticut	Minnesota	Minneapolis
06081	CT	MN	55455



Note that to specify multiple spanners, we pass a list of span objects. I've simplified the code a bit here by creating that list as a standalone object and then passing the whole list as `span`.

### 3.3.1 Using pipe syntax

For problems like this, it might be preferable to use the pipe syntax

```
data %>%
  st_new() %>%
  st_span("States", Connecticut:Minnesota) %>%
  st_span("The Universe", Tariffville:Minneapolis, level = 2) %>%
  stable() %>%
  st_asis()
```

The Universe			
States			
Tariffville	Connecticut	Minnesota	Minneapolis
06081	CT	MN	55455

## 3.4 Breaking span title

We can make the title of the span break across multiple lines by using `...`

```
stable(data, span = as.span("Example ... States", Connecticut:Minnesota)) %>%
  st_asis()
```

Example States			
Tariffville	Connecticut	Minnesota	Minneapolis
06081	CT	MN	55455

## 4 Tables that span multiple pages: longtable

### 4.1 Syntax

To create a long table from a data frame

```
stable_long(stdata())
```

To create a long table from pipeline

```
st_new(data) %>% stable_long()
```

To create a long table from pmtable

```
pt_cont_long(data, cols = "WT,EGFR") %>% stable_long()
```

### 4.2 Basics

You can create longtables that span multiple pages of your pdf document. Tables using the `longtable` environment are very different than the basic table from `stable()` which are built using `tabular` environment.

### 4.3 Inserting longtable into your latex document

Once you have written your long table out to a file, you can source it into your latex document with a simple input command

```
\input{my-table.tex}
```

You should not wrap your longtable input code in `\begin{table}/\end{table}` as you would with a regular tabular table.

## 5 The pipe interface

### 5.1 Basics

Mostly working with this data; but some others come in later to illustrate certain features.

```
data <- pmt_summarized
head(data)
```

```
. # A tibble: 6 x 9
.   STUDY      DOSE  FORM    N    WT   CRCL  AGE  ALB  SCR
.   <chr>      <chr> <chr>  <chr> <chr> <chr> <chr> <chr> <chr>
. 1 12-DEMO-001 100 mg tablet  80   71.4  104   33.7  4.20  1.06
. 2 12-DEMO-001 150 mg capsule 16   89.4  122   24.4  4.63  1.12
. 3 12-DEMO-001 150 mg tablet  48   81.7  104   34.4  3.83  0.910
. 4 12-DEMO-001 150 mg troche 16   94.0  93.2  27.4  4.94  1.25
. 5 12-DEMO-001 200 mg tablet  64   67.9  100   27.5  4.25  1.10
. 6 12-DEMO-001 200 mg troche 16   76.6  99.2  22.8  4.54  1.15
```

You start out a pipeline by passing your data frame into `st_new()`

```
data %>% st_new() %>% class
```

```
. [1] "stobject"      "environment"
```

This creates an object that gets revised by subsequent steps in the pipeline, adding features and styling as you go.

For the final step in the pipeline, we'll send the object to `stable()` to create the table

```
data %>% st_new() %>% stable() %>% head(n=9)
```

```
. [1] "\\setlength{\\tabcolsep}{5pt} "
. [2] "\\begin{threeparttable}"
. [3] "\\renewcommand{\\arraystretch}{1.3}"
. [4] "\\begin{tabular}[h]{l|lllllllll}"
. [5] "\\hline"
. [6] "STUDY & DOSE & FORM & N & WT & CRCL & AGE & ALB & SCR \\\\"
. [7] "\\hline"
. [8] "12-DEMO-001 & 100 mg & tablet & 80 & 71.4 & 104 & 33.7 & 4.20 & 1.06 \\\\"
. [9] "12-DEMO-001 & 150 mg & capsule & 16 & 89.4 & 122 & 24.4 & 4.63 & 1.12 \\\\"
```

## 5.2 Simple table

You can terminate the pipeline and create a tabular table by sending to `stable()`

```
data %>% st_new() %>% stable()
```

Equivalent call

```
data %>% stable()
```

## 5.3 Long table

You can also pipe to `stable_long()` to make a long table

```
data %>% st_new() %>% stable_long()
```

## 5.4 Annotate with file names

```
st_new(data) %>% st_files(r = "foo.R", output = "foo.tex")
```

Equivalent call

```
stable(data, r_file = "foo.R", output_file = "foo.tex")
```

Note that in the pipe version, we already have `file` in the function name so that gets dropped from the argument name.

## 5.5 Save a table

First convert with `stable()` or `stable_long()` then save with `stable_save()`

```
st_new(data) %>%  
  st_files(output = "foo.tex") %>%  
  stable() %>%  
  stable_save()
```

## 5.6 Align columns

- `st_center(...)`
- `st_left(...)`
- `st_right(...)`
- `st_align(...)`

```
st_new(data) %>%  
  st_center() %>%  
  stable()
```

Equivalent call

```
stable(data, align = cols_center())
```

## 5.7 Rename columns

```
st_new(data) %>%  
  st_rename(Weight = "WT") %>%  
  stable()
```

Equivalent call

```
stable(data, cols_rename = c(Weight = "WT"))
```

## 5.8 Blank columns

```
st_new(data) %>% st_blank(WT, ALB, DOSE) %>% stable()
```

Equivalent call

```
stable(data, cols_blank = "WT,ALB,DOSE")
```

## 5.9 Drop columns

```
st_new(data) %>% st_drop(WT, ALB, DOSE) %>% stable()
```

Equivalent call

```
stable(data, drop = "WT, ALB, DOSE")
```

## 5.10 Notes

- Multiple calls are allowed; notes will accumulate

```
st_new(data) %>%  
  st_notes("first note") %>%  
  st_nates("second note") %>%  
  stable()
```

Equivalent call

```
stable(  
  data,  
  notes = c("first note", "second note")  
)
```

## 5.11 Units

```
st_new(data) %>%  
  st_units(WT = "kg", AGE = "years") %>%  
  stable()
```

Note that `st_units()` will automatically add parens to your units; this can be suppressed with the `parens` argument.

Units can also be added as a list

```
u <- list(WT = "kg", AGE = "years")  
  
st_new(data) %>%  
  st_units(u) %>%  
  stable()
```

Equivalent call

```
stable(data, units = u)
```

## 5.12 panel

```
st_new(data) %>%  
  st_panel("STUDY") %>%  
  stable()
```

Equivalent call

```
stable(data, panel = "STUDY")
```

## 5.13 span

```
st_new(data) %>%  
  st_span("Covariates", c(WT, ALB, EGFR)) %>%  
  stable()
```

Equivalent call

```
stable(data, span = as.span("Covariates", c(WT, ALB, EGFR)))
```

## 5.14 hlines - at

To put horizontal lines at specific rows

```
st_new(data) %>% st_hline(at = c(2,4,6))
```

Equivalent call

```
stable(data, hline_at = c(2,4,6))
```

## 5.15 hlines - from

To calculate hlines based on data frame column values (for example, to break the table by changing values of STUDYf)

```
st_new(data) %>% st_hline(from = "STUDYf")
```

Equivalent call

```
stable(data, hline_from = "STUDYf")
```

## 5.16 hlines - pattern

To search the table for a pattern and add hlines above matches

```
st_new(data) %>% st_hline(pattern = "All", cols = "Summary")
```

```
. # A tibble: 13 x 9
.   STUDY      DOSE  FORM    N    WT    CRCL  AGE    ALB    SCR
.   <chr>      <chr> <chr>  <chr> <chr> <chr> <chr> <chr> <chr>
.  1 12-DEMO-001 100 mg tablet  80   71.4  104   33.7  4.20  1.06
.  2 12-DEMO-001 150 mg capsule 16   89.4  122   24.4  4.63  1.12
.  3 12-DEMO-001 150 mg tablet  48   81.7  104   34.4  3.83  0.910
.  4 12-DEMO-001 150 mg troche 16   94.0  93.2   27.4  4.94  1.25
.  5 12-DEMO-001 200 mg tablet  64   67.9  100   27.5  4.25  1.10
.  6 12-DEMO-001 200 mg troche 16   76.6  99.2   22.8  4.54  1.15
.  7 12-DEMO-002 100 mg capsule 36   61.3  113   38.3  4.04  1.28
.  8 12-DEMO-002 100 mg tablet 324   77.6  106   29.9  4.31  0.981
.  9 12-DEMO-002 50 mg capsule 36   74.1  112   37.1  4.44  0.900
. 10 12-DEMO-002 50 mg tablet 324   71.2  106   34.1  4.63  0.868
. 11 12-DEMO-002 75 mg capsule 36   72.4  105   38.2  3.89  0.900
. 12 12-DEMO-002 75 mg tablet 288   71.6  98.9   34.2  4.49  0.991
. 13 12-DEMO-002 75 mg troche 36   73.6  103   49.2  4.52  0.930
```

The cols argument limits the search to the Summary column; omit this argument to search the whole table.

There is no equivalent call for this using straight stable().

## 6 Options for previewing the table

### 6.1 st2report

Take a table or a list of tables and render them in a report-like document with a table of contents and a (faux) caption for each table

```
data %>% stable() %>% st2report()
```

You might try passing `ntex` to force the document to build more than once (sometimes the layout settles down after the second build)

```
data %>% stable() %>% st2report(ntex = 2)
```

Pass in a list of tables and you will get one table on each page with a listing of tables in the table of contents

```
list(table1, table2, table3) %>% st2report()
```

### 6.2 st2viewer

This function relies on `texPreview()` to render your table and display it as a graphic in the viewer window

```
data %>% stable() %>% st2viewer()
```

This method is more convenient because the tables always go to the viewer. But the rendering will not be like what you will see in the report.

### 6.3 st2article

Like `st2report()` but less report-like. You should use `st2report()` instead.

```
data %>% stable() %>% st2article()
```

### 6.4 st2doc

The original. Rather than building a TeX article, it runs the table in a Rmd document via pandoc. Not recommended; it is much slower to get the preview because there has to be a call to pandoc

```
data %>% stable() %>% st2doc()
```



## 7 A word about sanitizing table contents

### 7.1 Notes

```
x <- ptdata() %>% st_new(notes = "EDA_summary = TRUE") %>%  
  st_make(inspect = TRUE) %>%  
  get_stable_data()  
  
x$notes
```

```
. [1] "EDA\\_summary = TRUE"
```

### 7.2 File names

```
x <- ptdata() %>% st_new() %>%  
  st_files(r = "my_script.R") %>%  
  st_make(inspect = TRUE) %>%  
  get_stable_data()  
  
x$notes
```

```
. [1] "Source code: my\\_script.R"
```

### 7.3 Column names

```
out <-  
  tibble(a_1 = 5) %>%  
  stable(inspect = TRUE) %>%  
  get_stable_data()  
  
out$cols_tex
```

```
. [1] "a\\_1 \\\\"
```

### 7.4 Main table contents

```
out <-  
  tibble(a = "5_2") %>%  
  stable(inspect = TRUE) %>%  
  get_stable_data()  
  
out$tab
```

```
. [1] "5\\_2 \\\\"
```

## 7.5 Span titles

```
out <-
  ptdata() %>%
  stable(inspect = TRUE, span = colgroup("foo_this", WT:SCR)) %>%
  get_stable_data()

out$span_data$tex

. [1] "\\multicolumn{4}{c}{ } & \\multicolumn{5}{c}{foo\\_this} \\\\"
. [2] "\\cmidrule(lr){5-9}"
```

## 7.6 Panel names

```
data <- tibble(a = c("a_1", "a_1", "a_1", "a_2", "a_2"),
              b = letters[1:5])

out <- stable(data, panel = "a")
out[grepl("multicolumn", out)]

. [1] "\\multicolumn{1}{l}{\\textbf{a\\_1}}\\\\"
. [2] "\\hline \\multicolumn{1}{l}{\\textbf{a\\_2}}\\\\"
```

## 7.7 cols\_extra input

```
x <- letters[1:5]
data <- tibble(a = x, b = x, c = x)
xtra <- tibble(a = "foo%", b = "$\\mu$g", c = "1234 \\% %")
out <- stable(data, cols_extra = xtra)
out[grepl("%", out, fixed = TRUE)]

. [1] "foo\\% & $\\mu$g & 1234 \\% % \\\\"
```

## 8 pmtable

### Some setup

```
units = yspec::ys_get_unit(yspec::ys_help$spec(), parens = TRUE)
```

```
data <- pmt_first
data_pk <- pmt_pk
data_all <- pmt_obs
```

### 8.1 Principles

These functions expect that the user passes in all data that is to be summarized and nothing more. We will not filter your data.

### 8.2 Rename cols

When you select columns to summarize, you can generally pass in alternate (nicer) names that you want to show up in the table. For example, if I have a column called WT in the data frame and I want it to show up as Weight this can be accomplished during the call

```
pt_cont_wide(data, cols = c(Weight = "WT")) %>%
  stable(notes = NULL) %>% st_asis()
```

Weight
70.7 (12.8) [157]

Alternatively, you can use the table argument to enter rename info. Note that table is a list that should have names that match up with columns in the data frame and values that are the new names

```
tab <- list(SEXf = "Sex", ASIANf = "Race group")

pt_cat_wide(data, cols = "SEXf,ASIANf", table = tab) %>%
  stable() %>% st_asis()
```

	Sex		Race group	
n	male	female	Asian	non-Asian
160	80 (50.0)	80 (50.0)	66 (41.2)	94 (58.8)

Summary is count (percent)  
n: number of records summarized

### 8.3 Data inventory tables

- Count number of
  - individuals

- observations
  - BQL observations
  - missing values
- Calculate the percent of observations or BQL in different sub groups

### 8.3.1 Stacked by endpoint

- The stacked plot creates multiple independent tables to summarize different endpoints; there is no single overall summary for the table because we are summarizing different endpoints

```
out <- pt_data_inventory(
  data_all,
  by = c(Study = "STUDYf"),
  panel = as.panel("SEQf", prefix = "Endpoint: "),
  stacked = TRUE
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_axis()
```

Study	Number				Percent	
	SUBJ	MISS	OBS	BQL	OBS	BQL
<b>Endpoint: DEMO PK</b>						
12-DEMO-001	30	8	427	15	13.9	0.5
12-DEMO-002	50	10	1152	38	37.4	1.2
11-DEMO-005	40	10	920	30	29.9	1.0
13-DEMO-001	40	7	582	11	18.9	0.4
<i>Group Total</i>	160	35	3081	94	100.0	3.1
<b>Endpoint: ESTRDIOL</b>						
11-DEMO-005	40	0	40	0	50.6	0.0
13-DEMO-001	40	1	39	0	49.4	0.0
<i>Group Total</i>	80	1	79	0	100.0	0.0
<b>Endpoint: BMD</b>						
11-DEMO-005	40	9	111	0	49.1	0.0
13-DEMO-001	40	5	115	0	50.9	0.0
<i>Group Total</i>	80	14	226	0	100.0	0.0

SUBJ: subjects

BQL: below quantitation limit

MISS: missing observations (not BQL)

OBS: observations

Source code: test.R

Source file: test.tex

### 8.3.2 Paneled

- Just summarize a single endpoint

```
out <- pt_data_inventory(
  data_pk,
  by = c(Study = "STUDYf"),
  panel = "ASIANf"
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()
```

Study	Number				Group percent		Overall percent	
	SUBJ	MISS	OBS	BQL	OBS	BQL	OBS	BQL
<b>Asian</b>								
12-DEMO-001	17	4	241	10	19.5	0.8	7.8	0.3
12-DEMO-002	18	4	414	14	33.4	1.1	13.4	0.5
11-DEMO-005	16	5	366	13	29.5	1.0	11.9	0.4
13-DEMO-001	15	3	218	4	17.6	0.3	7.1	0.1
<b>non-Asian</b>								
12-DEMO-001	13	4	186	5	10.1	0.3	6.0	0.2
12-DEMO-002	32	6	738	24	40.1	1.3	24.0	0.8
11-DEMO-005	24	5	554	17	30.1	0.9	18.0	0.6
13-DEMO-001	25	4	364	7	19.8	0.4	11.8	0.2
<b>All data</b>	160	35	3081	94	—	—	100.0	3.1

SUBJ: subjects

BQL: below quantitation limit

MISS: missing observations (not BQL)

OBS: observations

Source code: test.R

Source file: test.tex

### 8.3.3 Grouped (by study)

```
out <- pt_data_inventory(  
  data_pk,  
  by = c(Study = "STUDYf")  
)  
  
out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()
```

Study	Number				Percent	
	SUBJ	MISS	OBS	BQL	OBS	BQL
12-DEMO-001	30	8	427	15	13.9	0.5
12-DEMO-002	50	10	1152	38	37.4	1.2
11-DEMO-005	40	10	920	30	29.9	1.0
13-DEMO-001	40	7	582	11	18.9	0.4
<b>All data</b>	160	35	3081	94	100.0	3.1

SUBJ: subjects

BQL: below quantitation limit

MISS: missing observations (not BQL)

OBS: observations

Source code: test.R

Source file: test.tex

## 8.4 Wide categorical table

- Summary of categorical data in wide format
- The summary is number (percent within group)
- Wide refers to the fact that the covariates go across the table

### 8.4.0.1 Ungrouped

```
out <- pt_cat_wide(  
  data = data,  
  cols = vars(Formulation = FORMf, Sex = SEXf, "Race group" = ASIANf)  
)  
  
out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()
```

n	Formulation			Sex		Race group	
	tablet	capsule	troche	male	female	Asian	non-Asian
160	130 (81.2)	15 (9.4)	15 (9.4)	80 (50.0)	80 (50.0)	66 (41.2)	94 (58.8)

Summary is count (percent)

n: number of records summarized

Source code: test.R

Source file: test.tex



### 8.4.1 Paneled (limited utility, IMO)

- Provided here for completeness

```
out <- pt_cat_wide(
  data = data,
  cols = vars(Formulation = FORMf, Sex = SEXf, "Race group" = ASIANf),
  panel = as.panel("STUDYf", prefix = "Study: ")
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()
```

n	Formulation			Sex		Race group	
	tablet	capsule	troche	male	female	Asian	non-Asian
<b>Study: 12-DEMO-001</b>							
30	25 (83.3)	3 (10.0)	2 (6.7)	10 (33.3)	20 (66.7)	17 (56.7)	13 (43.3)
<b>Study: 12-DEMO-002</b>							
50	42 (84.0)	6 (12.0)	2 (4.0)	18 (36.0)	32 (64.0)	18 (36.0)	32 (64.0)
<b>Study: 11-DEMO-005</b>							
40	30 (75.0)	3 (7.5)	7 (17.5)	29 (72.5)	11 (27.5)	16 (40.0)	24 (60.0)
<b>Study: 13-DEMO-001</b>							
40	33 (82.5)	3 (7.5)	4 (10.0)	23 (57.5)	17 (42.5)	15 (37.5)	25 (62.5)
<b>All data</b>							
160	130 (81.2)	15 (9.4)	15 (9.4)	80 (50.0)	80 (50.0)	66 (41.2)	94 (58.8)

Summary is count (percent)

n: number of records summarized

Source code: test.R

Source file: test.tex

#### 8.4.2 Grouped (by male / female)

```
out <- pt_cat_wide(
  data = data,
  by = c(Sex = "SEXf"),
  cols = vars(Formulation = FORMf, "Race group" = ASIANf)
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()
```

Sex	n	Formulation			Race group	
		tablet	capsule	troche	Asian	non-Asian
male	80	62 (77.5)	7 (8.8)	11 (13.8)	28 (35.0)	52 (65.0)
female	80	68 (85.0)	8 (10.0)	4 (5.0)	38 (47.5)	42 (52.5)
<b>All data</b>	160	130 (81.2)	15 (9.4)	15 (9.4)	66 (41.2)	94 (58.8)

Summary is count (percent)

n: number of records summarized

Source code: test.R

Source file: test.tex

## 8.4.3 Paneled and grouped

```

out <- pt_cat_wide(
  data = data,
  cols = vars(Formulation = FORMf, Sex = SEXf, "Race group" = ASIANf),
  panel = as.panel("STUDYf", prefix = "Study: "),
  by = c("RF Group" = "RFf")
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()

```

RF Group	n	Formulation			Sex		Race group	
		tablet	capsule	troche	male	female	Asian	non-Asian
Study: 12-DEMO-001								
normal	30	25 (83.3)	3 (10.0)	2 (6.7)	10 (33.3)	20 (66.7)	17 (56.7)	13 (43.3)
Study: 12-DEMO-002								
normal	50	42 (84.0)	6 (12.0)	2 (4.0)	18 (36.0)	32 (64.0)	18 (36.0)	32 (64.0)
Study: 11-DEMO-005								
normal	10	9 (90.0)	0 (0.0)	1 (10.0)	7 (70.0)	3 (30.0)	3 (30.0)	7 (70.0)
mild	10	7 (70.0)	2 (20.0)	1 (10.0)	7 (70.0)	3 (30.0)	5 (50.0)	5 (50.0)
moderate	10	6 (60.0)	0 (0.0)	4 (40.0)	8 (80.0)	2 (20.0)	6 (60.0)	4 (40.0)
severe	10	8 (80.0)	1 (10.0)	1 (10.0)	7 (70.0)	3 (30.0)	2 (20.0)	8 (80.0)
Study: 13-DEMO-001								
normal	40	33 (82.5)	3 (7.5)	4 (10.0)	23 (57.5)	17 (42.5)	15 (37.5)	25 (62.5)
All data	160	130 (81.2)	15 (9.4)	15 (9.4)	80 (50.0)	80 (50.0)	66 (41.2)	94 (58.8)

Summary is count (percent)

n: number of records summarized

Source code: test.R

Source file: test.tex

## 8.4.4 No summary

```

out <- pt_cat_wide(
  data = data,
  summarize = "none",
  cols = vars(Formulation = FORMf, Sex = SEXf, "Race group" = ASIANf),
  panel = as.panel("STUDYf", prefix = "Study: "),
  by = c("RF Group" = "RFf")
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()

```

RF Group	n	Formulation			Sex		Race group	
		tablet	capsule	troche	male	female	Asian	non-Asian
Study: 12-DEMO-001								
normal	30	25 (83.3)	3 (10.0)	2 (6.7)	10 (33.3)	20 (66.7)	17 (56.7)	13 (43.3)
Study: 12-DEMO-002								
normal	50	42 (84.0)	6 (12.0)	2 (4.0)	18 (36.0)	32 (64.0)	18 (36.0)	32 (64.0)
Study: 11-DEMO-005								
normal	10	9 (90.0)	0 (0.0)	1 (10.0)	7 (70.0)	3 (30.0)	3 (30.0)	7 (70.0)
mild	10	7 (70.0)	2 (20.0)	1 (10.0)	7 (70.0)	3 (30.0)	5 (50.0)	5 (50.0)
moderate	10	6 (60.0)	0 (0.0)	4 (40.0)	8 (80.0)	2 (20.0)	6 (60.0)	4 (40.0)
severe	10	8 (80.0)	1 (10.0)	1 (10.0)	7 (70.0)	3 (30.0)	2 (20.0)	8 (80.0)
Study: 13-DEMO-001								
normal	40	33 (82.5)	3 (7.5)	4 (10.0)	23 (57.5)	17 (42.5)	15 (37.5)	25 (62.5)

Summary is count (percent)

n: number of records summarized

Source code: test.R

Source file: test.tex

## 8.5 Long categorical table

- Categorical table in long format
- Long indicates that the covariates go down the table

### 8.5.1 Ungrouped

```
out <- pt_cat_long(
  data = data,
  cols = vars(Study = STUDYf, Sex = SEXf, "Race group" = ASIANf, "Child-Pugh" = CPf)
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()
```

	Summary n = 160
<b>Study</b>	
12-DEMO-001	30 (18.8)
12-DEMO-002	50 (31.2)
11-DEMO-005	40 (25.0)
13-DEMO-001	40 (25.0)
<b>Sex</b>	
male	80 (50.0)
female	80 (50.0)
<b>Race group</b>	
Asian	66 (41.2)
non-Asian	94 (58.8)
<b>Child-Pugh</b>	
Score=0	130 (81.2)
Score=1	10 (6.2)
Score=2	10 (6.2)
Score=3	10 (6.2)

Summary is count (percent)  
n: number of records summarized

Source code: test.R

Source file: test.tex

## 8.5.2 Grouped (by formulation)

```

out <- pt_cat_long(
  data = data,
  cols = vars(Study = STUDYf, Sex = SEXf, "Race group" = ASIANf, "Child-Pugh" = CPf),
  span = c(Formulation = "FORMf")
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()

```

	Formulation			Summary n = 160
	tablet n = 130	capsule n = 15	troche n = 15	
Study				
12-DEMO-001	25 (19.2)	3 (20.0)	2 (13.3)	30 (18.8)
12-DEMO-002	42 (32.3)	6 (40.0)	2 (13.3)	50 (31.2)
11-DEMO-005	30 (23.1)	3 (20.0)	7 (46.7)	40 (25.0)
13-DEMO-001	33 (25.4)	3 (20.0)	4 (26.7)	40 (25.0)
Sex				
male	62 (47.7)	7 (46.7)	11 (73.3)	80 (50.0)
female	68 (52.3)	8 (53.3)	4 (26.7)	80 (50.0)
Race group				
Asian	53 (40.8)	7 (46.7)	6 (40.0)	66 (41.2)
non-Asian	77 (59.2)	8 (53.3)	9 (60.0)	94 (58.8)
Child-Pugh				
Score=0	106 (81.5)	12 (80.0)	12 (80.0)	130 (81.2)
Score=1	7 (5.4)	1 (6.7)	2 (13.3)	10 (6.2)
Score=2	8 (6.2)	1 (6.7)	1 (6.7)	10 (6.2)
Score=3	9 (6.9)	1 (6.7)	0 (0.0)	10 (6.2)

Summary is count (percent)

n: number of records summarized

Source code: test.R

Source file: test.tex

### 8.5.3 Summary on bottom and right

```
out <- pt_cat_long(
  data = data,
  summarize = "both",
  cols = vars(Formulation = FORMf, Sex = SEXf, "Race group" = ASIANf),
  span = vars(Study = STUDYf)
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()
```

	Study				Summary n = 160
	12-DEMO-001 n = 30	12-DEMO-002 n = 50	11-DEMO-005 n = 40	13-DEMO-001 n = 40	
<b>Formulation</b>					
tablet	25 (83.3)	42 (84.0)	30 (75.0)	33 (82.5)	130 (81.2)
capsule	3 (10.0)	6 (12.0)	3 (7.5)	3 (7.5)	15 (9.4)
troche	2 (6.7)	2 (4.0)	7 (17.5)	4 (10.0)	15 (9.4)
<b>Sex</b>					
male	10 (33.3)	18 (36.0)	29 (72.5)	23 (57.5)	80 (50.0)
female	20 (66.7)	32 (64.0)	11 (27.5)	17 (42.5)	80 (50.0)
<b>Race group</b>					
Asian	17 (56.7)	18 (36.0)	16 (40.0)	15 (37.5)	66 (41.2)
non-Asian	13 (43.3)	32 (64.0)	24 (60.0)	25 (62.5)	94 (58.8)

Summary is count (percent)

n: number of records summarized

Source code: test.R

Source file: test.tex

## 8.5.4 No summary

```

out <- pt_cat_long(
  data = data,
  summarize = "none",
  cols = vars(Formulation = FORMf, Sex = SEXf, "Race group" = ASIANf),
  span = vars(Study = STUDYf)
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()

```

	Study			
	12-DEMO-001	12-DEMO-002	11-DEMO-005	13-DEMO-001
<b>Formulation</b>				
tablet	25 (83.3)	42 (84.0)	30 (75.0)	33 (82.5)
capsule	3 (10.0)	6 (12.0)	3 (7.5)	3 (7.5)
troche	2 (6.7)	2 (4.0)	7 (17.5)	4 (10.0)
<b>Sex</b>				
male	10 (33.3)	18 (36.0)	29 (72.5)	23 (57.5)
female	20 (66.7)	32 (64.0)	11 (27.5)	17 (42.5)
<b>Race group</b>				
Asian	17 (56.7)	18 (36.0)	16 (40.0)	15 (37.5)
non-Asian	13 (43.3)	32 (64.0)	24 (60.0)	25 (62.5)

Summary is count (percent)

n: number of records summarized

Source code: test.R

Source file: test.tex



## 8.6 Wide continuous table

- Continuous table in wide format
- Wide means that the covariates go across the table

### 8.6.1 Ungrouped

```
out <- pt_cont_wide(
  data = data,
  cols = "WT,SCR,AGE,ALB,HT",
  units = units
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()
```

WT (kg)	SCR (mg/dL)	AGE (years)	ALB (g/dL)	HT (cm)
70.7 (12.8) [157]	1.36 (0.986) [160]	33.7 (8.83) [160]	4.20 (0.793) [156]	179 (17.7) [160]
Summary is mean (sd) [count]				
Source code: test.R				
Source file: test.tex				

### 8.6.2 Paneled

```
out <- pt_cont_wide(
  data = data,
  cols = "WT,SCR,AGE,ALB,HT",
  panel = c(Study = "STUDYf"),
  units = units
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()
```

WT (kg)	SCR (mg/dL)	AGE (years)	ALB (g/dL)	HT (cm)
<b>Study 12-DEMO-001</b>				
72.2 (14.3) [29]	1.03 (0.155) [30]	32.0 (9.19) [30]	4.28 (0.474) [29]	180 (19.3) [30]
<b>Study 12-DEMO-002</b>				
72.4 (11.5) [49]	0.971 (0.161) [50]	35.0 (8.20) [50]	4.47 (0.468) [50]	182 (15.4) [50]
<b>Study 11-DEMO-005</b>				
68.9 (14.5) [39]	2.52 (1.43) [40]	32.8 (8.48) [40]	4.41 (0.537) [39]	175 (19.2) [40]
<b>Study 13-DEMO-001</b>				
69.4 (11.6) [40]	0.950 (0.165) [40]	34.2 (9.67) [40]	3.58 (1.15) [38]	179 (17.2) [40]
<b>All data</b>				
70.7 (12.8) [157]	1.36 (0.986) [160]	33.7 (8.83) [160]	4.20 (0.793) [156]	179 (17.7) [160]
Summary is mean (sd) [count]				
Source code: test.R				
Source file: test.tex				

### 8.6.3 Grouped (by study)

```
out <- pt_cont_wide(
  data = data,
  cols = "WT,SCR,AGE,ALB,HT",
  by = c(Study = "STUDYf"),
  units = units
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()
```

Study	WT (kg)	SCR (mg/dL)	AGE (years)	ALB (g/dL)	HT (cm)
12-DEMO-001	72.2 (14.3) [29]	1.03 (0.155) [30]	32.0 (9.19) [30]	4.28 (0.474) [29]	180 (19.3) [30]
12-DEMO-002	72.4 (11.5) [49]	0.971 (0.161) [50]	35.0 (8.20) [50]	4.47 (0.468) [50]	182 (15.4) [50]
11-DEMO-005	68.9 (14.5) [39]	2.52 (1.43) [40]	32.8 (8.48) [40]	4.41 (0.537) [39]	175 (19.2) [40]
13-DEMO-001	69.4 (11.6) [40]	0.950 (0.165) [40]	34.2 (9.67) [40]	3.58 (1.15) [38]	179 (17.2) [40]
<b>All data</b>	70.7 (12.8) [157]	1.36 (0.986) [160]	33.7 (8.83) [160]	4.20 (0.793) [156]	179 (17.7) [160]

Summary is mean (sd) [count]

Source code: test.R

Source file: test.tex

## 8.6.4 Paneled and grouped

```

out <- pt_cont_wide(
  data = data,
  cols = "WT,SCR,AGE,ALB,HT",
  by = c(Study = "STUDYf"),
  panel = c(Formulation = "FORMf"),
  units = units
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()

```

Study	WT (kg)	SCR (mg/dL)	AGE (years)	ALB (g/dL)	HT (cm)
<b>Formulation tablet</b>					
12-DEMO-001	71.0 (14.2) [24]	1.01 (0.157) [25]	32.6 (9.23) [25]	4.22 (0.459) [24]	179 (19.7) [25]
12-DEMO-002	72.2 (11.8) [41]	0.966 (0.166) [42]	34.0 (7.93) [42]	4.49 (0.495) [42]	182 (15.9) [42]
11-DEMO-005	68.8 (15.2) [29]	2.48 (1.47) [30]	33.2 (8.73) [30]	4.37 (0.568) [29]	173 (19.7) [30]
13-DEMO-001	69.4 (11.0) [33]	0.967 (0.163) [33]	33.7 (9.67) [33]	3.53 (1.14) [31]	178 (16.5) [33]
<b>Formulation capsule</b>					
12-DEMO-001	72.9 (17.3) [3]	1.12 (0.0700) [3]	32.2 (12.0) [3]	4.49 (0.593) [3]	184 (23.0) [3]
12-DEMO-002	70.9 (10.3) [6]	1.03 (0.146) [6]	37.7 (7.59) [6]	4.38 (0.354) [6]	181 (15.4) [6]
11-DEMO-005	73.9 (11.1) [3]	3.06 (2.19) [3]	31.8 (4.99) [3]	4.65 (0.240) [3]	181 (16.4) [3]
13-DEMO-001	58.4 (4.04) [3]	0.973 (0.195) [3]	36.5 (6.69) [3]	3.09 (1.50) [3]	167 (8.88) [3]
<b>Formulation troche</b>					
12-DEMO-001	85.3 (12.4) [2]	1.20 (0.0707) [2]	25.1 (3.28) [2]	4.74 (0.283) [2]	194 (0.163) [2]
12-DEMO-002	79.7 (8.61) [2]	0.910 (0.0283) [2]	48.0 (1.79) [2]	4.49 (0.0354) [2]	182 (10.9) [2]
11-DEMO-005	66.8 (13.9) [7]	2.45 (1.05) [7]	31.4 (9.34) [7]	4.49 (0.509) [7]	177 (19.8) [7]
13-DEMO-001	77.4 (15.9) [4]	0.795 (0.0777) [4]	37.3 (12.9) [4]	4.32 (0.994) [4]	193 (22.4) [4]
<b>All data</b>	70.7 (12.8) [157]	1.36 (0.986) [160]	33.7 (8.83) [160]	4.20 (0.793) [156]	179 (17.7) [160]

Summary is mean (sd) [count]

Source code: test.R

Source file: test.tex

## 8.7 Long continuous table

- Continuous summary table in long format
- Long indicates that covariates go down the table

### 8.7.1 Ungrouped

```
out <- pt_cont_long(  
  data = data,  
  cols = "WT,SCR,AGE",  
  units = units  
)  
  
out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()
```

Variable	n	Mean	Median	SD	Min / Max
WT (kg)	157	70.7	70.0	12.8	43.6 / 97.2
SCR (mg/dL)	160	1.36	1.04	0.986	0.710 / 5.59
AGE (years)	160	33.7	33.4	8.83	18.9 / 49.5

n: number of records summarized

SD: standard deviation

Min: minimum; Max: maximum

Source code: test.R

Source file: test.tex

## 8.7.2 Paneled

```

out <- pt_cont_long(
  data = data,
  cols = "WT,SCR,AGE",
  panel = vars(Study = STUDYf),
  units = units
)

out %>% stable(r_file = "test.R", output_file = "test.tex") %>% st_asis()

```

Variable	n	Mean	Median	SD	Min / Max
<b>Study 12-DEMO-001</b>					
WT (kg)	29	72.2	70.0	14.3	50.9 / 97.2
SCR (mg/dL)	30	1.03	1.04	0.155	0.740 / 1.30
AGE (years)	30	32.0	28.0	9.19	19.9 / 47.8
<b>Study 12-DEMO-002</b>					
WT (kg)	49	72.4	72.1	11.5	51.5 / 96.6
SCR (mg/dL)	50	0.971	0.970	0.161	0.720 / 1.30
AGE (years)	50	35.0	36.0	8.20	20.3 / 49.2
<b>Study 11-DEMO-005</b>					
WT (kg)	39	68.9	65.4	14.5	43.6 / 92.8
SCR (mg/dL)	40	2.52	2.33	1.43	0.720 / 5.59
AGE (years)	40	32.8	33.4	8.48	19.2 / 49.5
<b>Study 13-DEMO-001</b>					
WT (kg)	40	69.4	68.1	11.6	50.7 / 96.6
SCR (mg/dL)	40	0.950	0.975	0.165	0.710 / 1.26
AGE (years)	40	34.2	35.2	9.67	18.9 / 49.5
<b>All data</b>					
WT (kg)	157	70.7	70.0	12.8	43.6 / 97.2
SCR (mg/dL)	160	1.36	1.04	0.986	0.710 / 5.59
AGE (years)	160	33.7	33.4	8.83	18.9 / 49.5

n: number of records summarized

SD: standard deviation

Min: minimum; Max: maximum

Source code: test.R

Source file: test.tex