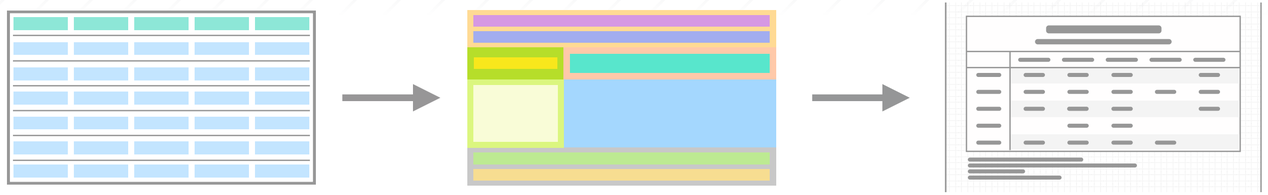
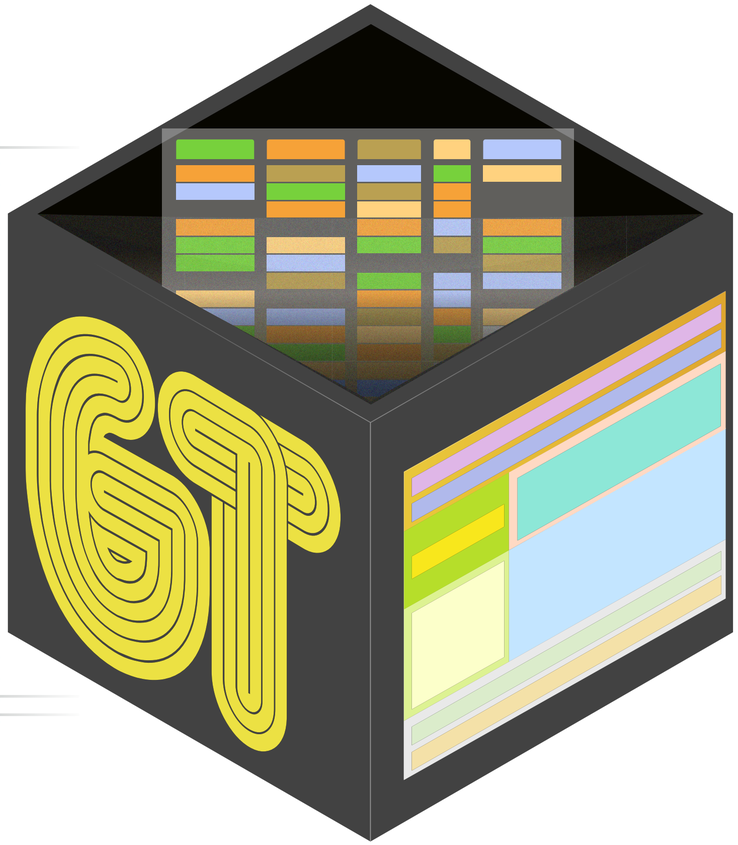
The **gt** Package

Introduction and Demo





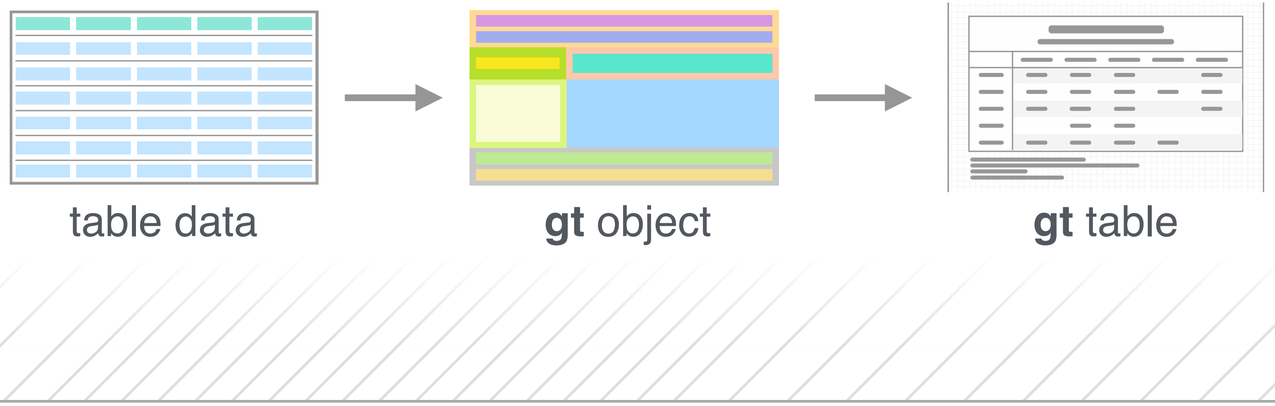
rich-iannone

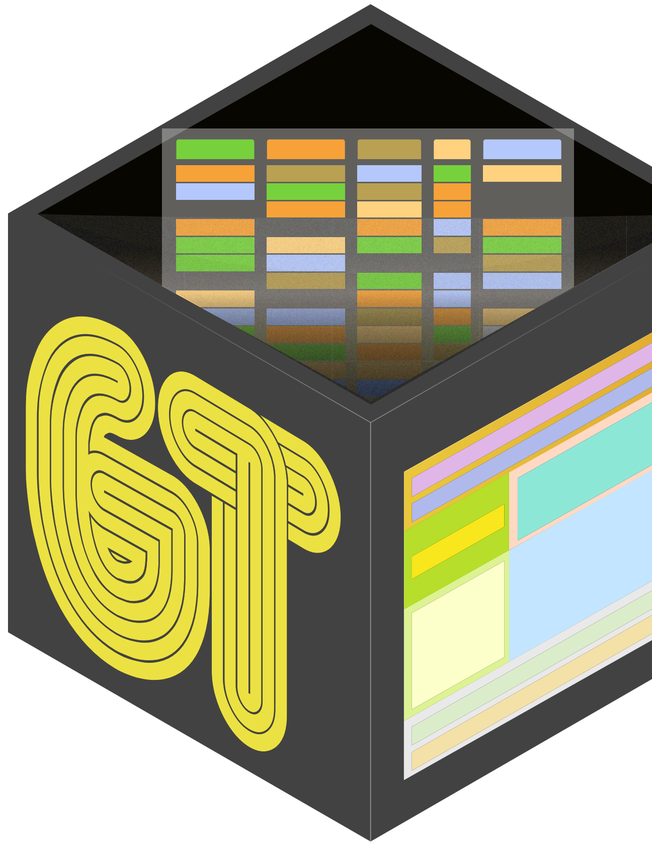
@gt_package

rich@posit.co

The **gt** Package

Introduction and Demo





The **gt** package lets us create *display tables* with a declarative interface, allowing us to fine-tune the final appearance.

We can integrate the tables in publishing workflows and **Shiny** apps.

Features of the **gt** Package.

1 // A declarative but forgiving API.

```
library(gt)

exibble %>%
  gt(rowname_col = "row", groupname_col = "group") %>%
  tab_source_note(source_note = "Source note.") %>%
  tab_footnote(
    footnote = "This is a footnote.",
    locations = cells_body(columns = 1, rows = 1)
  ) %>%
  tab_header(
    title = "The title of the table",
    subtitle = "The table's subtitle"
  )
```

Should always start with `gt()`.

Subsequent statements can usually be expressed in any order.

Each function acts as instructions. **gt** should then take all those instructions and figure out what to render.

Features of the **gt** Package.

2 // Table generation in multiple output types with the same API.

```r
library(gt)

exibble %>%
  gt(rowname_col = "row", groupname_col = "group") %>%
  tab_source_note(source_note = "Source note.") %>%
  tab_footnote(
    footnote = "This is a footnote.",
    locations = cells_body(columns = 1, rows = 1)
  ) %>%
  tab_header(
    title = "The title of the table",
    subtitle = "The table's subtitle"
  )
```

The set of statements supplied to **gt** doesn't have to change depending on type of table output you want.

HTML
LaTeX
RTF
Word

FLEXIBLE OUTPUTS, SAME API

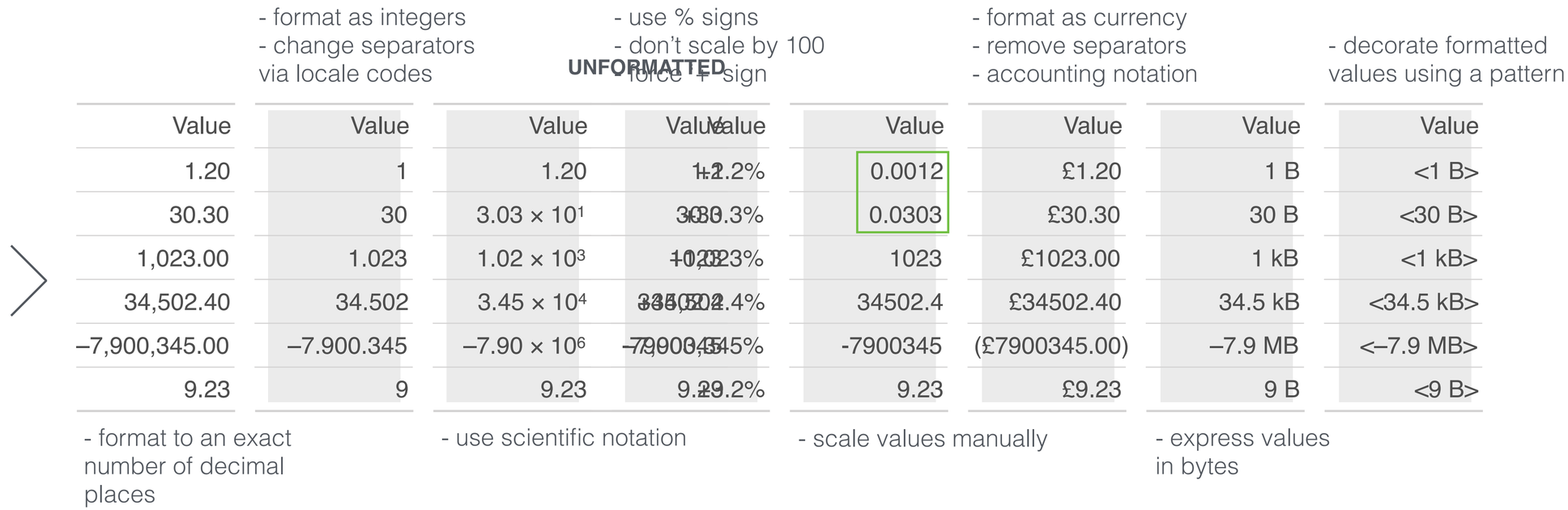
gt should always work when taking the same code used for HTML table over to a LaTeX **R Markdown** document.

Features of the **gt** Package.

3 // Useful formatting functions for cell values.

We can format numbers, dates, and strings with a large set of very flexible and easy-to-use functions.

```
fmt_number() fmt_integer() fmt_scientific() fmt_engineering() fmt_percent() fmt_partsper() fmt_fraction() fmt_currency() fmt
```



Features of the **gt** Package.

4 // Methods for restructuring table data.

We are able to express how **gt** tables are structured.
Some rearrangements happen automatically but manual control is available.

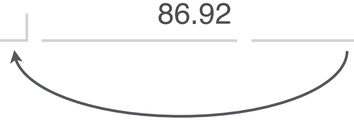
- columns gathered together when placed under a column spanner

Column 1	Column 2	Column 3
23.42	–	15.24
63.90	21.34	43.70
—	61.93	26.00
1.29	17.60	15.58
–28.02	–10.55	–5.23
86.92	65.23	47.25

Column Spanner		
Column 1	Column 3	Column 2
23.42	15.24	–
63.90	43.70	21.34
—	26.00	61.93
1.29	15.58	17.60
–28.02	–5.23	–10.55
86.92	47.25	65.23

- move columns manually

Column 1	Column 2
23.42	–
63.90	21.34
—	61.93
1.29	17.60
–28.02	–10.55
86.92	65.23



Features of the **gt** Package.

5 // Easy-to-use footnotes that self-organize.

THE BASICS

It is straightforward to define table footnotes in **gt**.

Column 1	Column 2[a]	Column 3
23.42	–	15.24
63.90	21.34	[a] 143.70
—	61.93	26.00
1.29	17.60	15.58
–28.02	–10.55	–5.23
86.92	65.23	47.25

[a] We, unfortunately, cannot explain this value.

gt always expresses the ordering of footnotes automatically.

Column 1	Column 2[a]	Column 3
23.42	–	15.24
63.90	21.34	[b] 143.70
—	61.93	26.00
1.29	17.60	15.58
[c] –28.02	[c] –10.55	[c] –5.23
86.92	65.23	47.25

[a] This is the column in the middle. The 2nd one.

[b] We, unfortunately, cannot explain this value.

[c] We don't expect negative values, yet, here they are.

Features of the **gt** Package.

5 // Easy-to-use footnotes that self-organize.

THE BASICS

It is straightforward to define table footnotes in **gt**.

gt always expresses the ordering of footnotes automatically.

ADVANCED HANDLING

We are able to apply the same footnote to multiple locations.

Base coupe[1]
GT coupe[1]
[1] These coupe

gt preserves the same footnote mark

Multiple footnotes are allowed at the same location.

Base coupe[1,2,3]
GT coupe[1,4,5]
[1] These coupes can h
[2] Base models tend to
[3] This is the only optic
[4] Although labeled as
[5] Final year in which th

gt handles complex footnote marks in the expected manner

You can create output tables in four different formats.

	num	char	fctr	date	time	datetime	currency
grp_a							
row_1	0.11	apricot	one	Jan 15, 2015	1:35 PM	Jan 1, 2018 2:22 AM	€49.95
row_2	2.22	banana	two	Feb 15, 2015	2:40 PM	Feb 2, 2018 2:33 PM	€17.95
row_3	33.33	coconut	three	Mar 15, 2015	3:45 PM	Mar 3, 2018 3:44 AM	€1.39
row_4	444.40	durian	four	Apr 15, 2015	4:50 PM	Apr 4, 2018 3:55 PM	€65,100.00
grp_b							
row_5	5,550.00	NA	five	May 15, 2015	5:55 PM	May 5, 2018 4:00 AM	€1,325.81
row_6	NA	fig	six	Jun 15, 2015	NA	Jun 6, 2018 4:11 PM	€13.26
row_7	777,000.00	grapefruit	seven	NA	7:10 PM	Jul 7, 2018 5:22 AM	NA
row_8	8,880,000.00	honeydew	eight	Aug 15, 2015	8:20 PM	NA	€0.44

HTML

You can create output tables in four different formats.

	num	char	fctr	date	time	datetime	currency
grp_a							
row_1	0.11	apricot	one	Jan 15, 2015	1:35 PM	Jan 1, 2018 2:22 AM	EUR49.95
row_2	2.22	banana	two	Feb 15, 2015	2:40 PM	Feb 2, 2018 2:33 PM	EUR17.95
row_3	33.33	coconut	three	Mar 15, 2015	3:45 PM	Mar 3, 2018 3:44 AM	EUR1.39
row_4	444.40	durian	four	Apr 15, 2015	4:50 PM	Apr 4, 2018 3:55 PM	EUR65,100.00
grp_b							
row_5	5,550.00	NA	five	May 15, 2015	5:55 PM	May 5, 2018 4:00 AM	EUR1,325.81
row_6	NA	fig	six	Jun 15, 2015	NA	Jun 6, 2018 4:11 PM	EUR13.26
row_7	777,000.00	grapefruit	seven	NA	7:10 PM	Jul 7, 2018 5:22 AM	NA
row_8	8,880,000.00	honeydew	eight	Aug 15, 2015	8:20 PM	NA	EUR0.44

LaTeX

You can create output tables in four different formats.

	num	char	fctr	date	time	datetime	currency
grp_a							
row_1	0.11	apricot	one	Jan 15, 2015	1:35 PM	Jan 1, 2018 2:22 AM	EUR49.95
row_2	2.22	banana	two	Feb 15, 2015	2:40 PM	Feb 2, 2018 2:33 PM	EUR17.95
row_3	33.33	coconut	three	Mar 15, 2015	3:45 PM	Mar 3, 2018 3:44 AM	EUR1.39
row_4	444.40	durian	four	Apr 15, 2015	4:50 PM	Apr 4, 2018 3:55 PM	EUR65,100.00
grp_b							
row_5	5,550.00	NA	five	May 15, 2015	5:55 PM	May 5, 2018 4:00 AM	EUR1,325.81
row_6	NA	fig	six	Jun 15, 2015	NA	Jun 6, 2018 4:11 PM	EUR13.26
row_7	777,000.00	grapefruit	seven	NA	7:10 PM	Jul 7, 2018 5:22 AM	NA
row_8	8,880,000.00	honeydew	eight	Aug 15, 2015	8:20 PM	NA	EUR0.44

RTF

You can create output tables in four different formats.

num	char	fctr	date	time	datetime	currency	row	group
1.111e-01	apricot	one	2015-01-15	13:35	2018-01-01 02:22	49.950	row_1	grp_a
2.222e+00	banana	two	2015-02-15	14:40	2018-02-02 14:33	17.950	row_2	grp_a
3.333e+01	coconut	three	2015-03-15	15:45	2018-03-03 03:44	1.390	row_3	grp_a
4.444e+02	durian	four	2015-04-15	16:50	2018-04-04 15:55	65100.000	row_4	grp_a
5.550e+03	NA	five	2015-05-15	17:55	2018-05-05 04:00	1325.810	row_5	grp_b
NA	fig	six	2015-06-15	NA	2018-06-06 16:11	13.255	row_6	grp_b
7.770e+05	grapefruit	seven	NA	19:10	2018-07-07 05:22	NA	row_7	grp_b
8.880e+06	honeydew	eight	2015-08-15	20:20	NA	0.440	row_8	grp_b

Word

gt works well within **R Markdown** and **Quarto**.

````
```{r exibble_gt}
exibble %>%
  gt(
    rowname_col = "row",
    groupname_col = "group"
  ) %>%
  fmt_number(
    columns = num,
    decimals = 2
  ) %>%
  fmt_date(
    columns = date,
    date_style = 6
  ) %>%
  fmt_time(
    columns = time,
    time_style = 4
  ) %>%
  fmt_datetime(
    columns = datetime,
    date_style = 6,
    time_style = 4
  ) %>%
  fmt_currency(
    columns = currency,
    currency = "EUR"
  )
```
````

	num	char	fctr	date	time	datetime	currency
grp_a							
row_1	0.11	apricot	one	Jan 15, 2015	1:35 PM	Jan 1, 2018 2:22 AM	€49.95
row_2	2.22	banana	two	Feb 15, 2015	2:40 PM	Feb 2, 2018 2:33 PM	€17.95
row_3	33.33	coconut	three	Mar 15, 2015	3:45 PM	Mar 3, 2018 3:44 AM	€1.39
row_4	444.40	durian	four	Apr 15, 2015	4:50 PM	Apr 4, 2018 3:55 PM	€65,100.00
grp_b							
row_5	5,550.00	NA	five	May 15, 2015	5:55 PM	May 5, 2018 4:00 AM	€1,325.81
row_6	NA	fig	six	Jun 15, 2015	NA	Jun 6, 2018 4:11 PM	€13.26
row_7	777,000.00	grapefruit	seven	NA	7:10 PM	Jul 7, 2018 5:22 AM	NA

.Rmd / .qmd

code chunks within the document. You can embed an R code chunk like this:

```
exibble %>%
  gt(
    rowname_col = "row",
    groupname_col = "group"
  ) %>%
  fmt_number(
    columns = num,
    decimals = 2
  ) %>%
  fmt_date(
    columns = date,
    date_style = 6
  ) %>%
  fmt_time(
    columns = time,
    time_style = 4
  ) %>%
  fmt_datetime(
    columns = datetime,
    date_style = 6,
    time_style = 4
  ) %>%
  fmt_currency(
    columns = currency,
    currency = "EUR"
  )
```

	num	char	fctr	date	time	datetime	currency
grp_a							
row_1	0.11	apricot	one	Jan 15, 2015	1:35 PM	Jan 1, 2018 2:22 AM	€49.95
row_2	2.22	banana	two	Feb 15, 2015	2:40 PM	Feb 2, 2018 2:33 PM	€17.95
row_3	33.33	coconut	three	Mar 15, 2015	3:45 PM	Mar 3, 2018 3:44 AM	€1.39
row_4	444.40	durian	four	Apr 15, 2015	4:50 PM	Apr 4, 2018 3:55 PM	€65,100.00
grp_b							
row_5	5,550.00	NA	five	May 15, 2015	5:55 PM	May 5, 2018 4:00 AM	€1,325.81
row_6	NA	fig	six	Jun 15, 2015	NA	Jun 6, 2018 4:11 PM	€13.26
row_7	777,000.00	grapefruit	seven	NA	7:10 PM	Jul 7, 2018 5:22 AM	NA
row_8	8,880,000.00	honeydew	eight	Aug 15, 2015	8:20 PM	NA	€0.44

R Markdown HTML

document. You can embed an R code chunk like this:

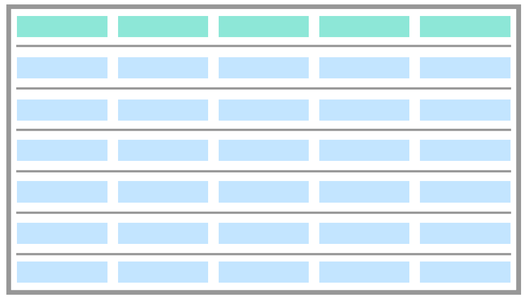
```
exibble %>%
  gt(
    rowname_col = "row",
    groupname_col = "group"
  ) %>%
  fmt_number(
    columns = num,
    decimals = 2
  ) %>%
  fmt_date(
    columns = date,
    date_style = 6
  ) %>%
  fmt_time(
    columns = time,
    time_style = 4
  ) %>%
  fmt_datetime(
    columns = datetime,
    date_style = 6,
    time_style = 4
  ) %>%
  fmt_currency(
    columns = currency,
    currency = "EUR"
  )
```

	num	char	fctr	date	time	datetime	currency
grp_a							
row_1	0.11	apricot	one	Jan 15, 2015	1:35 PM	Jan 1, 2018 2:22 AM	€49.95
row_2	2.22	banana	two	Feb 15, 2015	2:40 PM	Feb 2, 2018 2:33 PM	€17.95
row_3	33.33	coconut	three	Mar 15, 2015	3:45 PM	Mar 3, 2018 3:44 AM	€1.39
row_4	444.40	durian	four	Apr 15, 2015	4:50 PM	Apr 4, 2018 3:55 PM	€65,100.00
grp_b							
row_5	5,550.00	NA	five	May 15, 2015	5:55 PM	May 5, 2018 4:00 AM	€1,325.81
row_6	NA	fig	six	Jun 15, 2015	NA	Jun 6, 2018 4:11 PM	€13.26
row_7	777,000.00	grapefruit	seven	NA	7:10 PM	Jul 7, 2018 5:22 AM	NA
row_8	8,880,000.00	honeydew	eight	Aug 15, 2015	8:20 PM	NA	€0.44

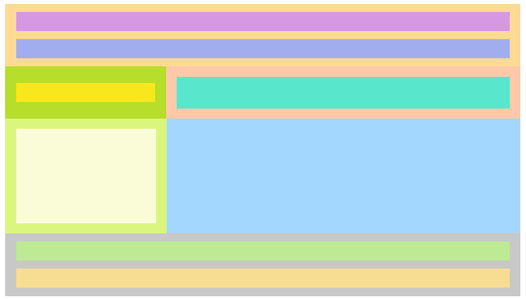
Quarto HTML

The Typical Workflow for Making Tables with **gt**.

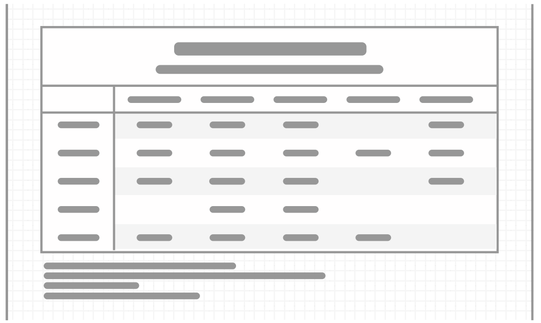
① table data



② **gt** object



③ **gt** table



Put your data in a form that's reasonably close to the expected form of the display table.

Use **dplyr** and **tidyr** and other great Tidyverse 📦s.

The Typical Workflow for Making Tables with **gt**.

① ② ③

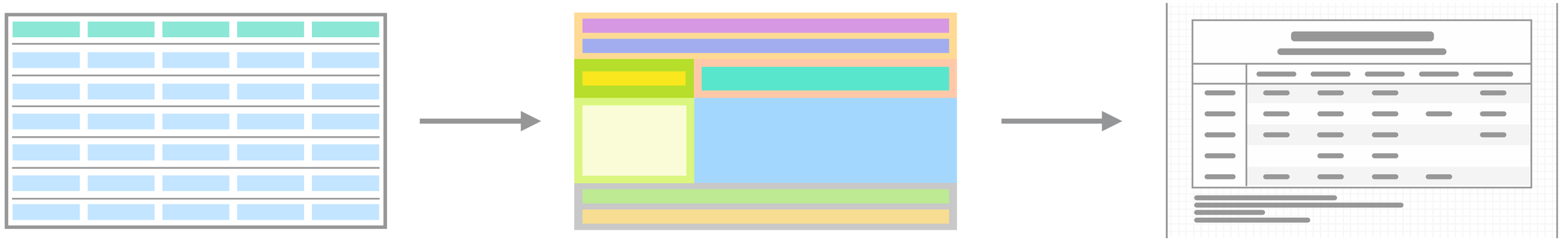


table data | **gt** object | **gt** table

Put your data in a form that's reasonably close to the expected form of the display table.

Use **dplyr** and **tidyr** and other great Tidyverse 📦s.

Add table components, group rows together, add spanner labels, footnotes, format cells, add styles…

Use **gt**'s functions to build. Preview in **RStudio**.

The Typical Workflow for Making Tables with **gt**.

① ② ③

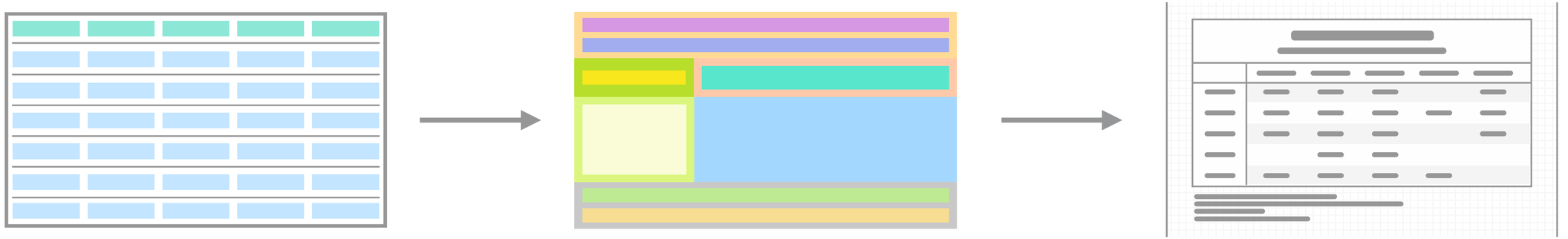


table data | **gt** object | **gt** table

Put your data in a form that's reasonably close to the expected form of the display table.

Add table components, group rows together, add spanner labels, footnotes, format cells, add styles…

Output the table to **HTML**, save an image. **RTF** and **LaTeX** output is possible as well.

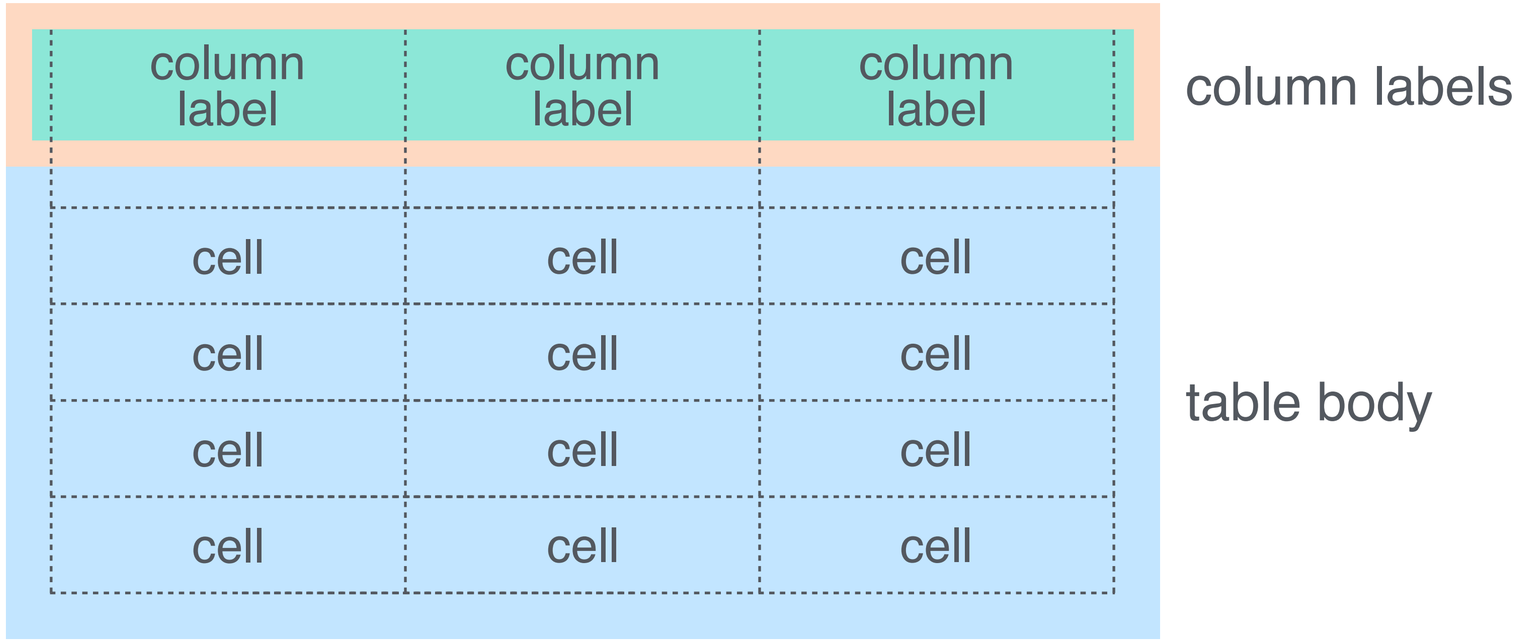
Use **dplyr** and **tidyr** and other great Tidyverse 📦s.

Use **gt**'s functions to build. Preview in **RStudio**.

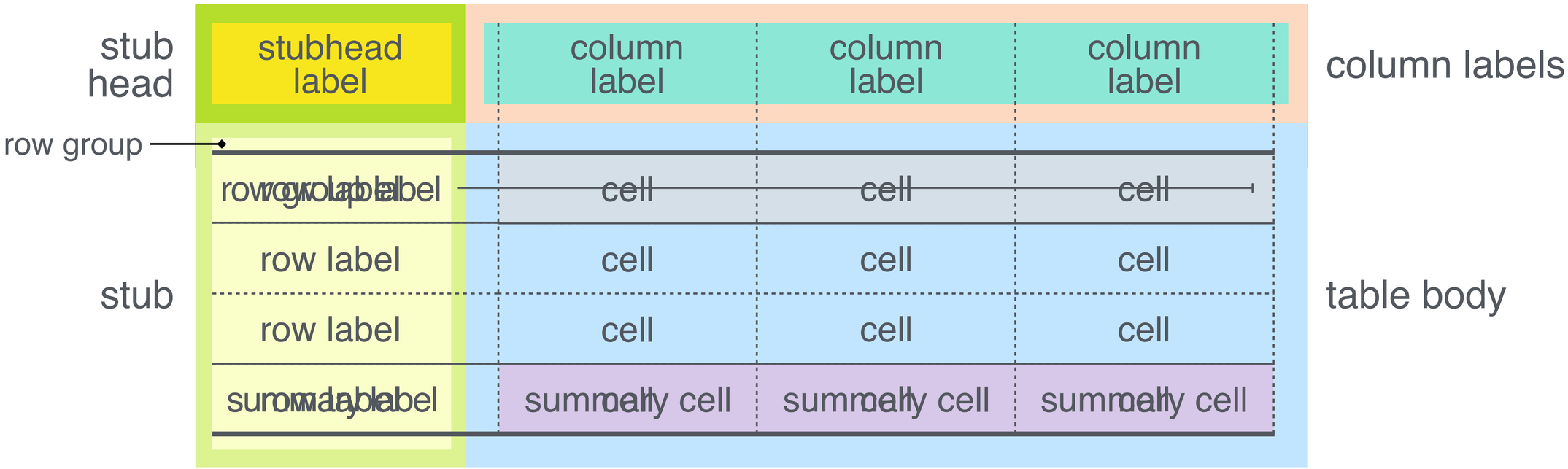
Use tables in **Shiny** apps, reports, packages, etc.

Understanding the different parts of a table.

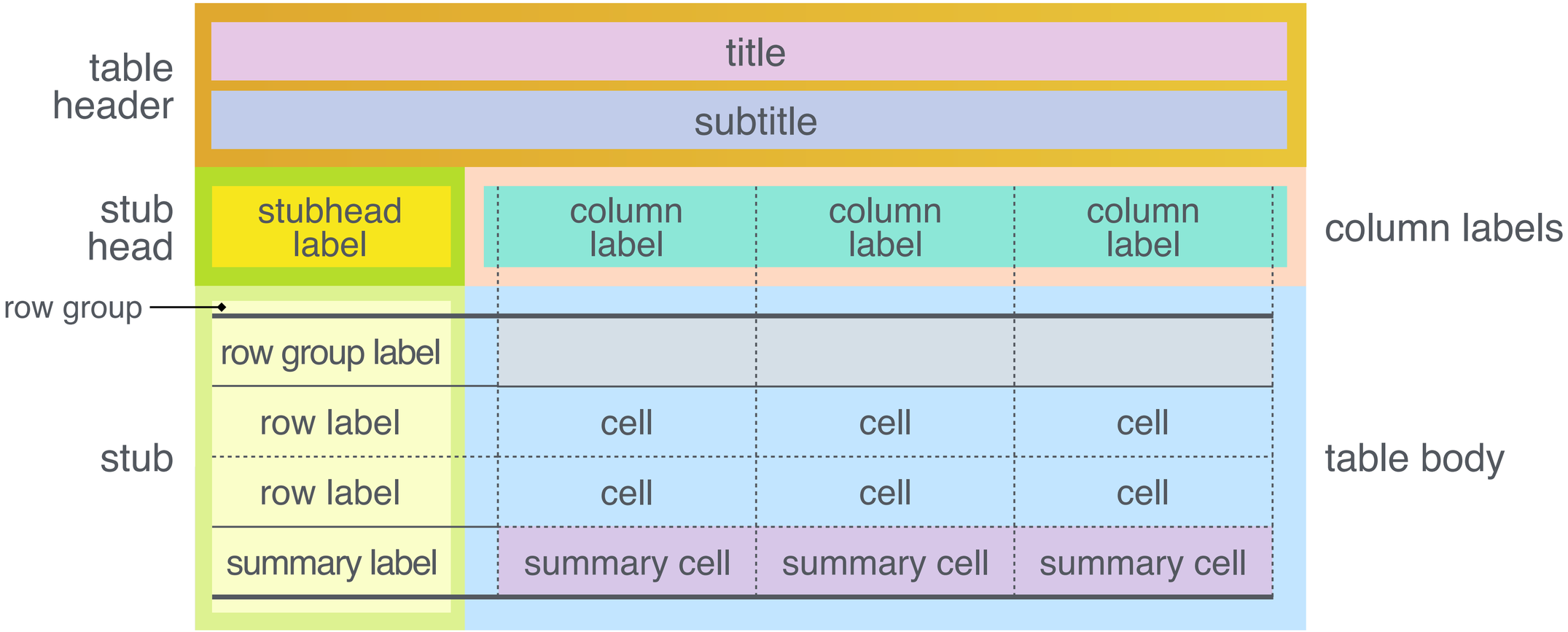
This is the most basic form of a **gt** table:



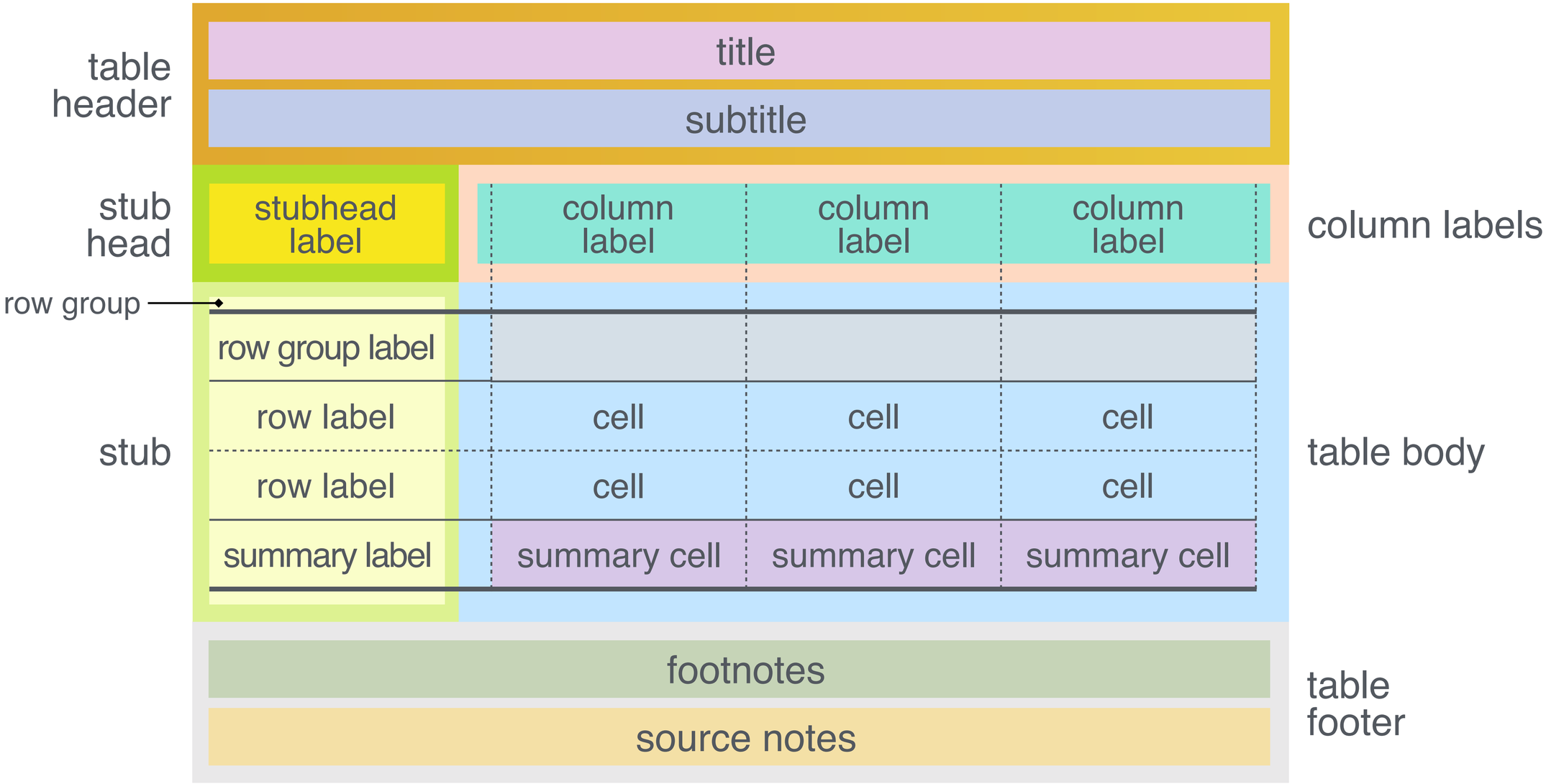
Understanding the different parts of a table.



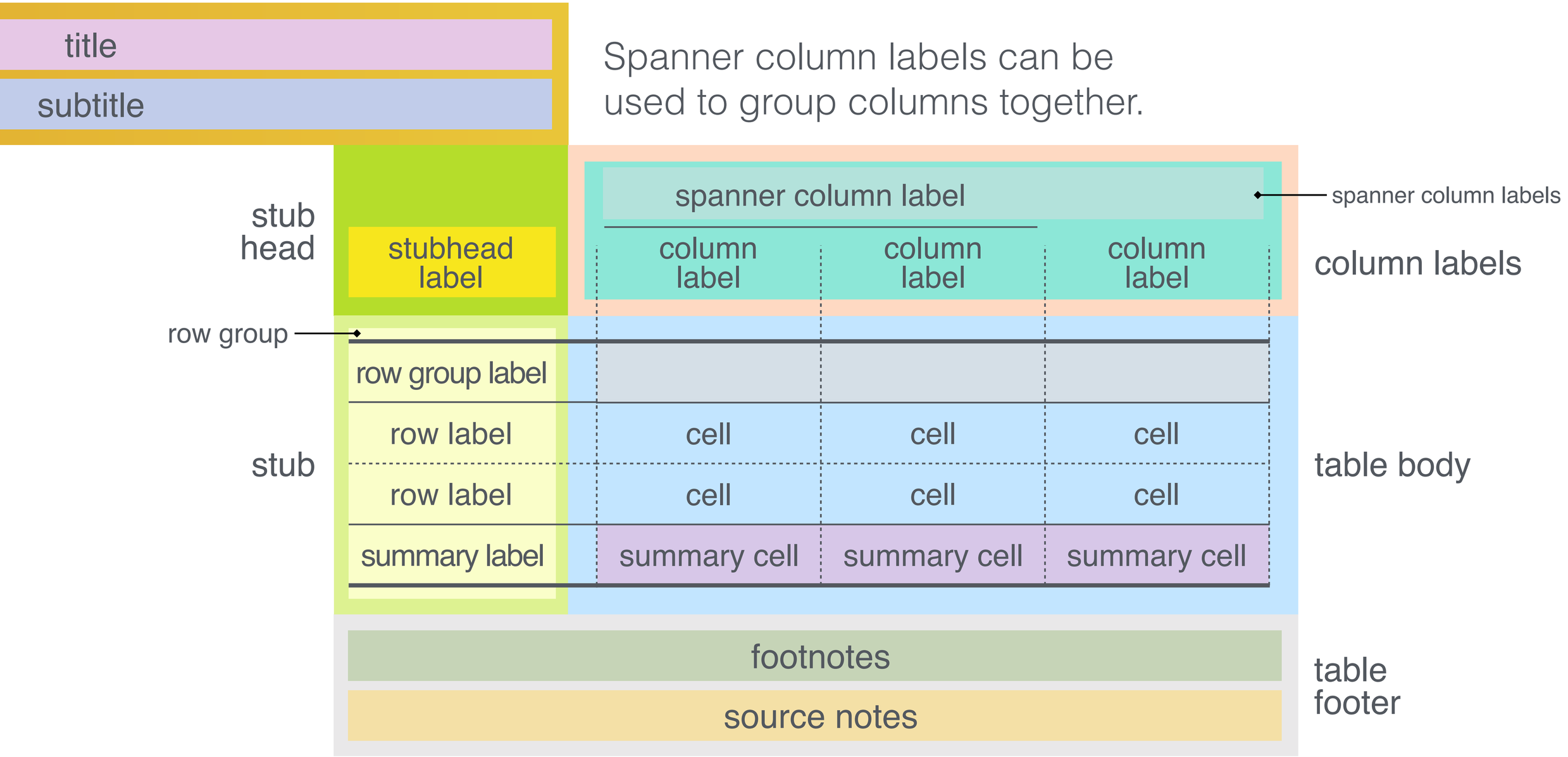
A table stub is not always needed but it can be useful.
Rows can be grouped, and they can have labels.
Summary rows can be added to groups (or, we can have a *grand summary*).



A table header is a great place to add a title and a subtitle.



Footnotes and source notes serve as useful annotations.


title
subtitle
Spanner column labels can be used to group columns together.
stub head
stubhead label
spanner column label
spanner column labels
column label
column label
column label
column labels
row group
row group label
row label
cell
cell
cell
stub
row label
cell
cell
cell
summary label
summary cell
summary cell
summary cell
table body
footnotes
source notes
table footer

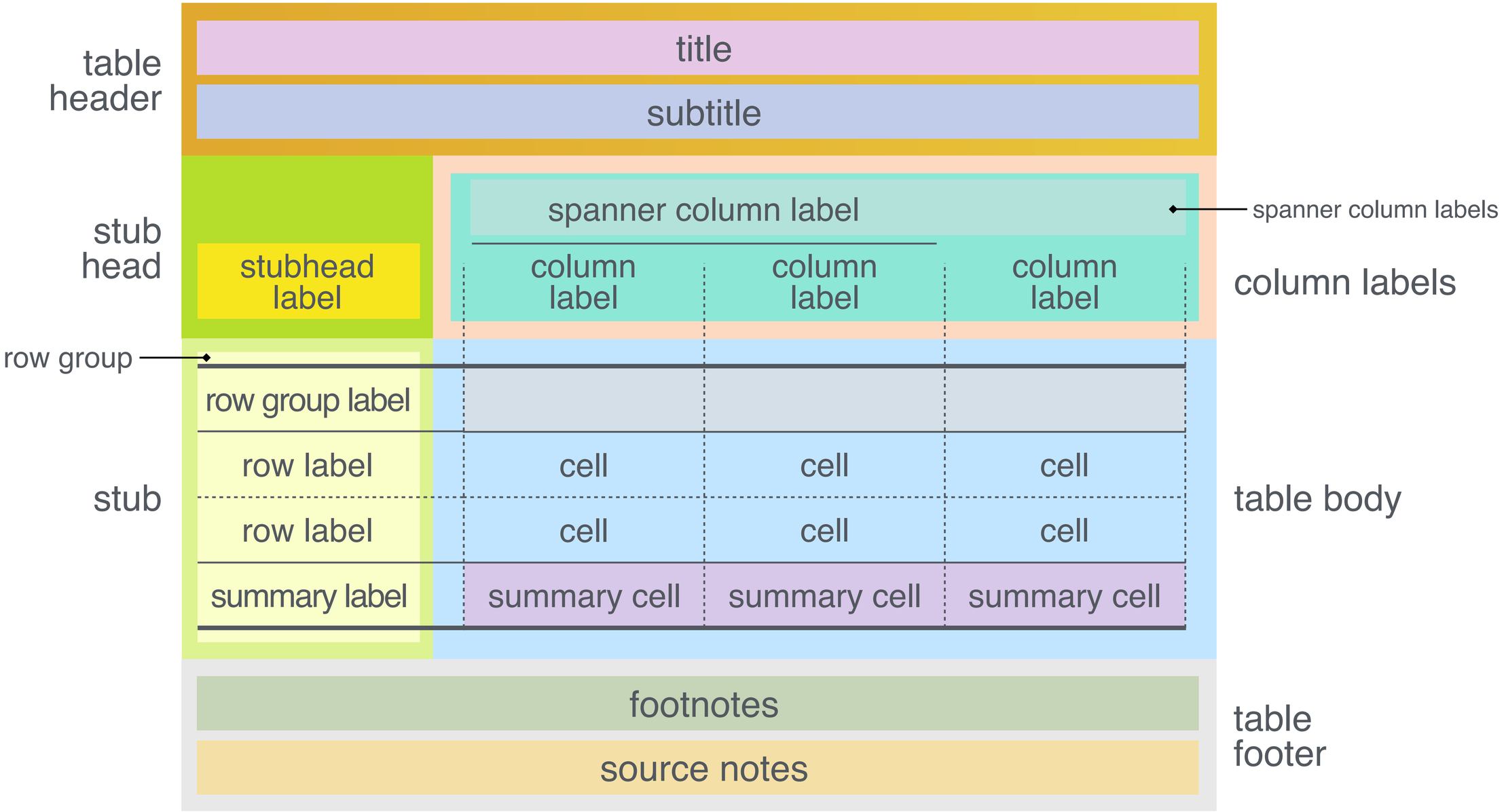


table header
title
subtitle
stub head
spanner column label
spanner column labels
stubhead label
column label
column label
column label
column labels
row group
row group label
stub
row label
cell
cell
cell
row label
cell
cell
cell
summary label
summary cell
summary cell
summary cell
table body
footnotes
source notes
table footer

Demo

More information on **gt**.

You can try out dozens of **gt** examples in **Posit Cloud**

The link is available in the package README and in the project website

github.com/rstudio/gt

gt.rstudio.com

More information on **gt**.

You can try out dozens of **gt** examples in **Posit Cloud**

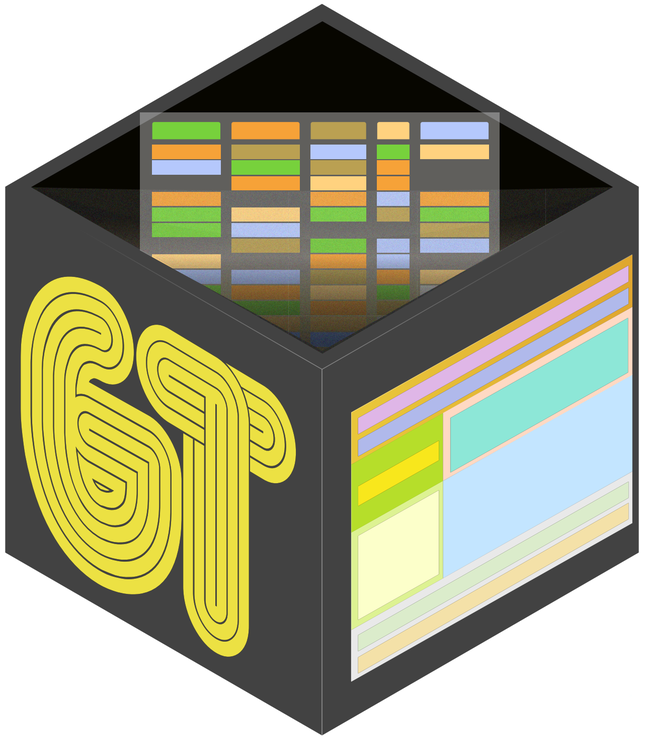
The link is available in the package README and the project website

github.com/rstudio/gt gt.rstudio.com

---

gt's *Function Reference* section has per-function info

gt.rstudio.com/reference



https://github.com/rich-iannone/presentations

rich-iannone @gt_package rich@posit.co