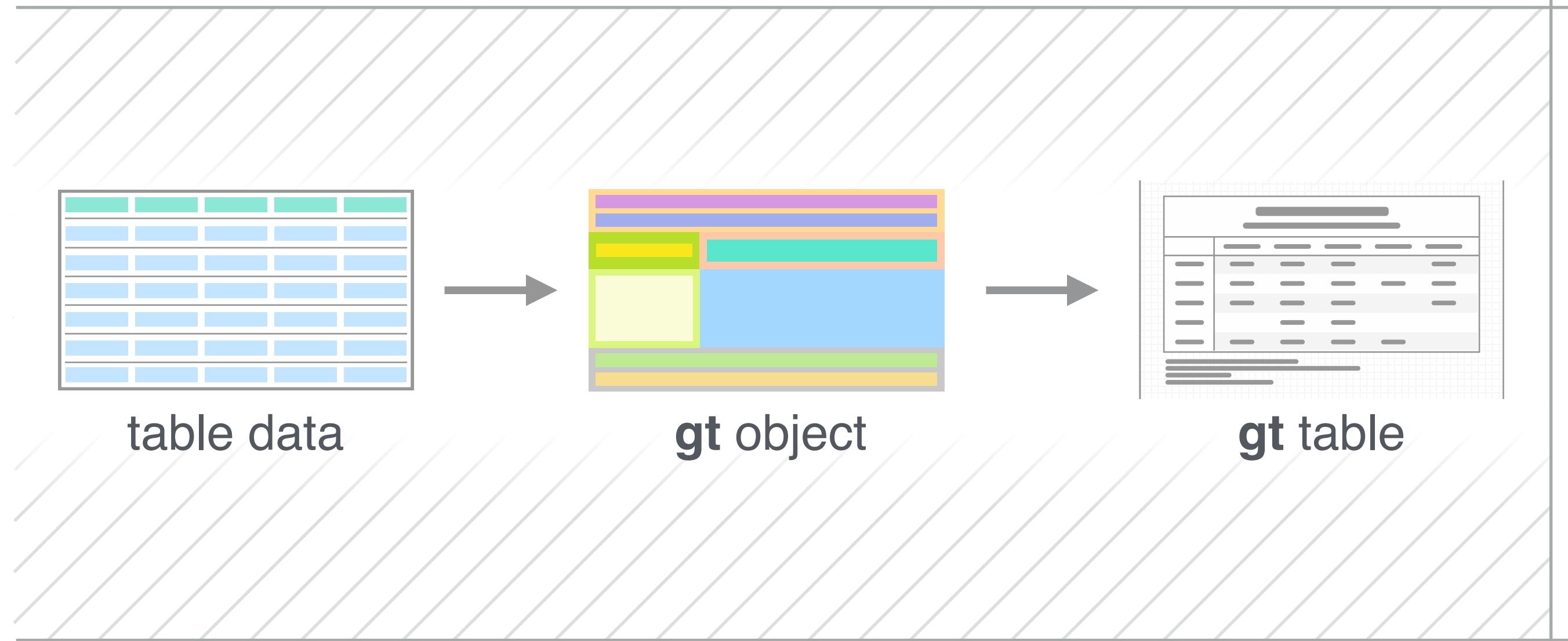


The gt Package

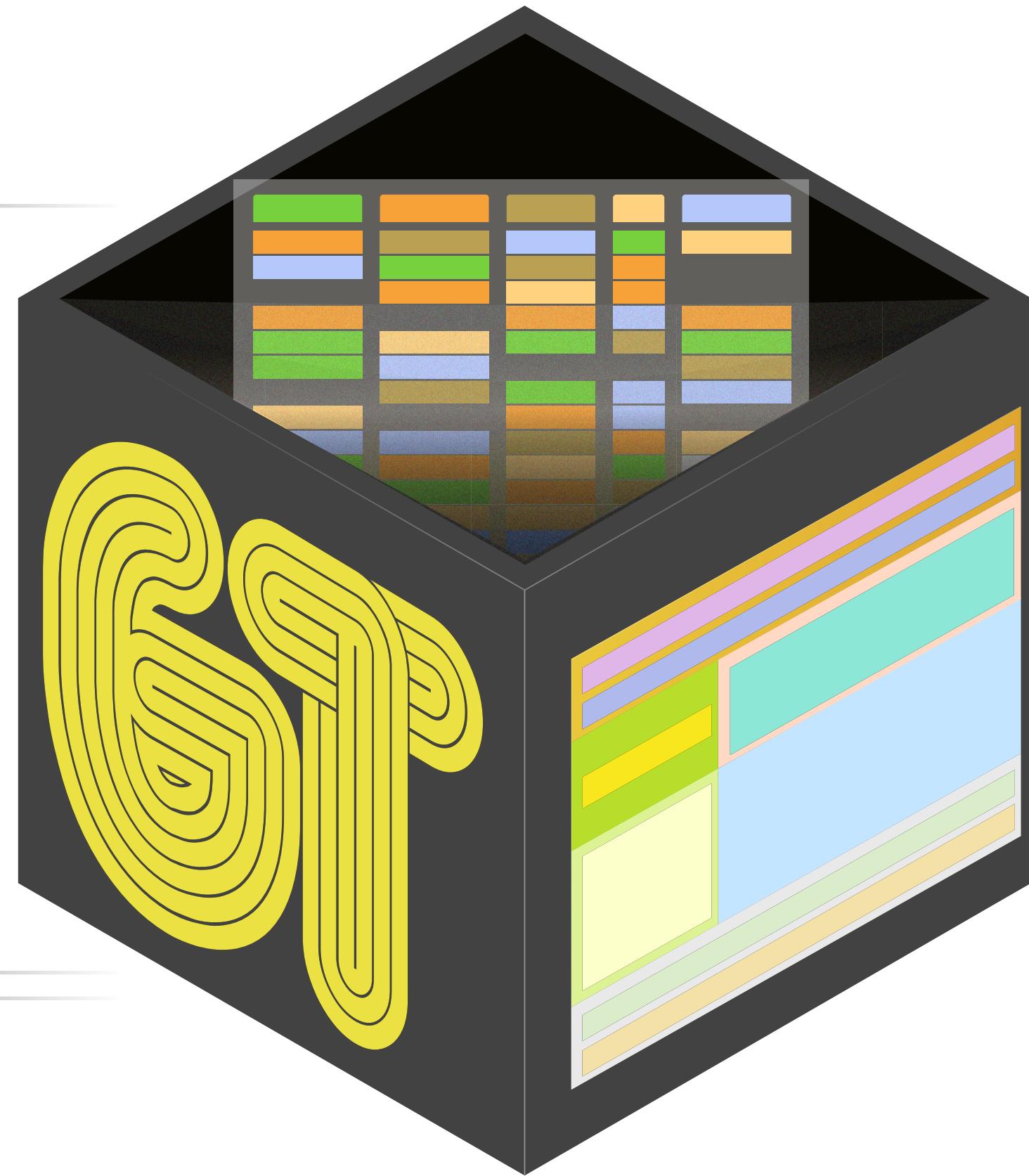
Introduction and Demo



 rich-iannone

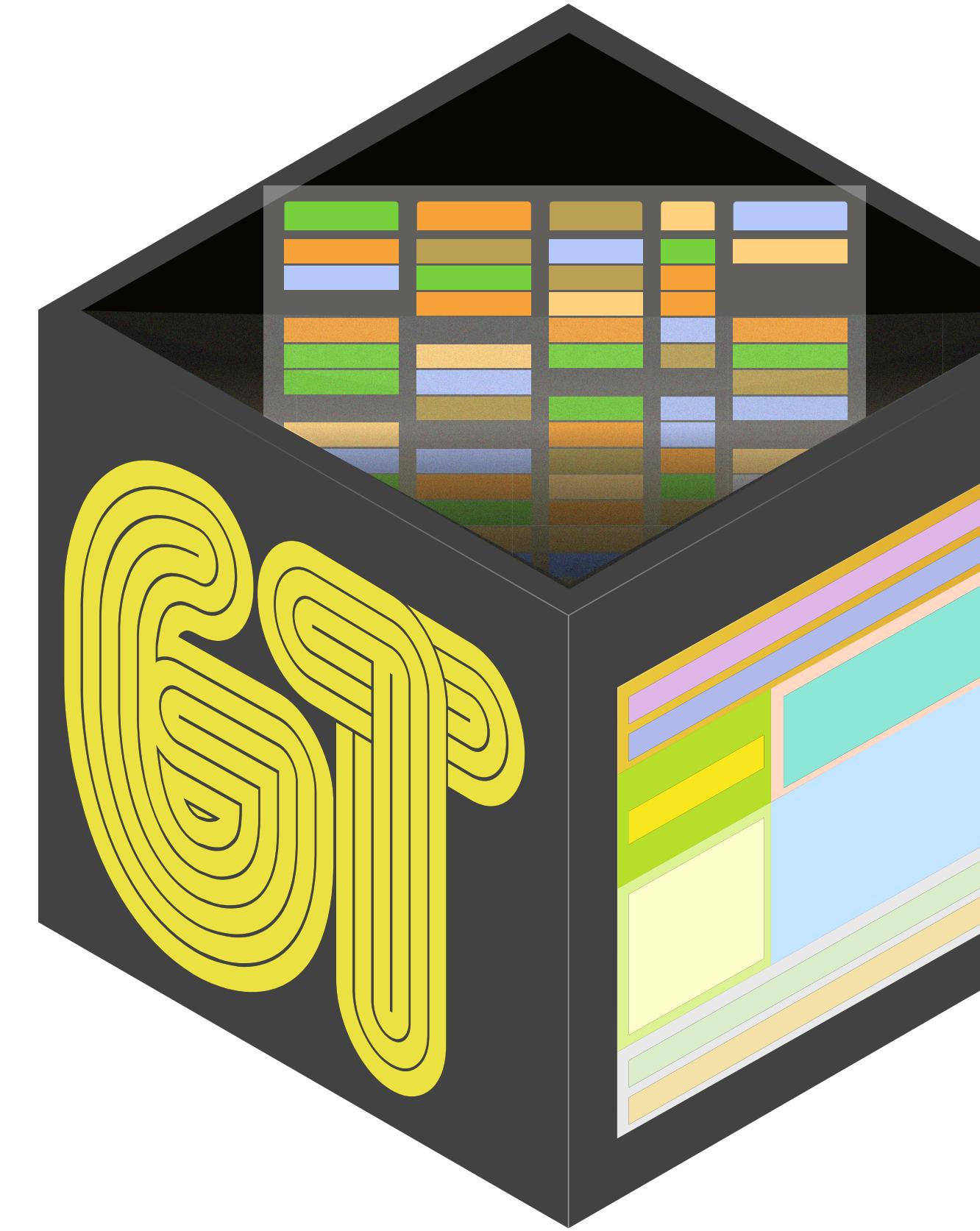
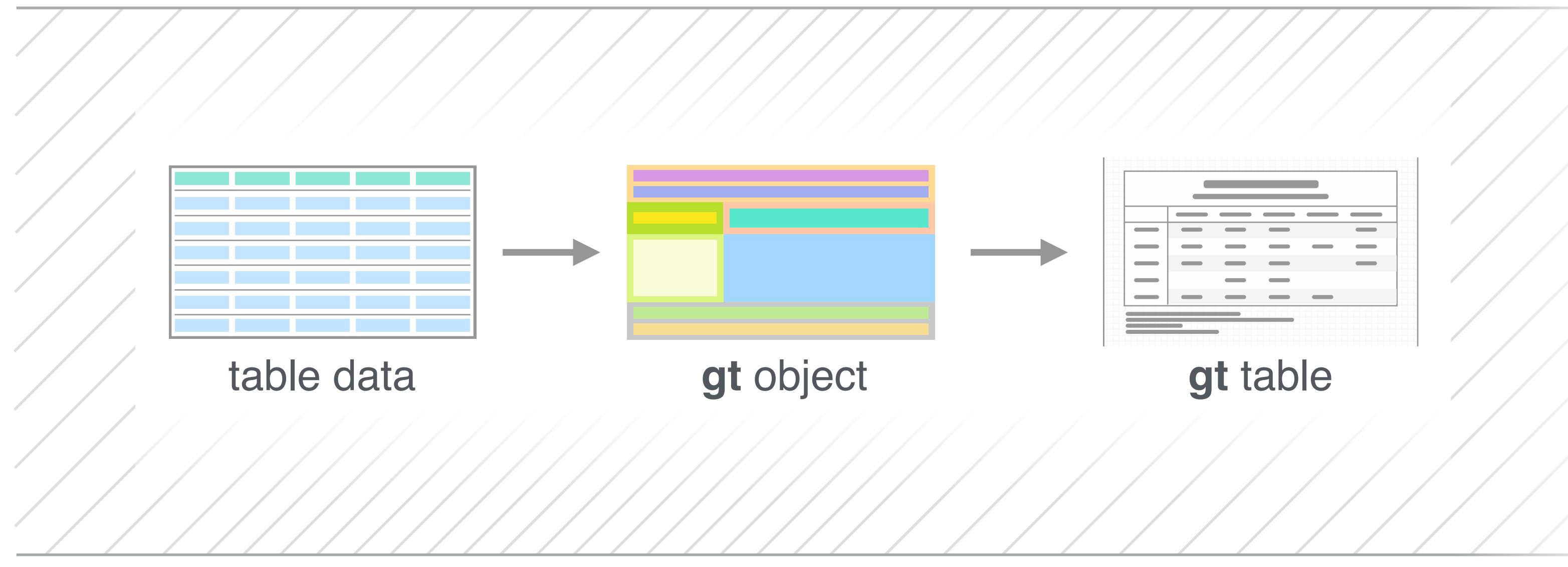
 @gt_package

 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.

```
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_bytes()`

	- format as integers - change separators via locale codes	UNFORMATTED			- use % signs - don't scale by 100 force + sign		- format as currency - remove separators - accounting notation		- decorate formatted values using a pattern
	Value	Value	Value	Value	Value	Value	Value	Value	Value
	1.20	1	1.20	1.2%	0.0012	£1.20	1 B	<1 B>	
	30.30	30	3.03×10^1	30.3%	0.0303	£30.30	30 B	<30 B>	
	1,023.00	1.023	1.02×10^3	1023%	1023	£1023.00	1 kB	<1 kB>	
	34,502.40	34.502	3.45×10^4	34502.4%	34502.4	£34502.40	34.5 kB	<34.5 kB>	
	-7,900,345.00	-7.900.345	-7.90×10^6	-79000345%	-7900345	(£7900345.00)	-7.9 MB	<-7.9 MB>	
	9.23	9	9.23	9.29.2%	9.23	£9.23	9 B	<9 B>	
	- format to an exact number of decimal places	- use scientific notation			- scale values manually	- express values in bytes			

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 coupe¹

GT coupe¹

¹These coupes

gt preserves the same footnote mark

Multiple footnotes are allowed at the same location.

Base coupe^{1,2,3}

GT coupe^{1,4,5}

¹These coupes can h

²Base models tend to

³This is the only optic

⁴Although labeled as

⁵Final 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

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	EURO0.44

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

gt works well within R Markdown and Quarto.

```
16
17 ````{r exibble_gt}
18 exibble %>%
19   gt(
20     rowname_col = "row",
21     groupname_col = "group"
22   ) %>%
23   fmt_number(
24     columns = num,
25     decimals = 2
26   ) %>%
27   fmt_date(
28     columns = date,
29     date_style = 6
30   ) %>%
31   fmt_time(
32     columns = time,
33     time_style = 4
34   ) %>%
35   fmt_datetime(
36     columns = datetime,
37     date_style = 6,
38     time_style = 4
39   ) %>%
40   fmt_currency(
41     columns = currency,
42     currency = "EUR"
43   )
44 ````
```

The screenshot shows an RStudio interface. On the left, there is an R code chunk with syntax highlighting for R code. On the right, the output of the code is displayed as a gt table. The table has a header row with columns labeled 'num', 'char', 'fctr', 'date', 'time', 'datetime', and 'currency'. Below the header, there are two sections: 'grp_a' and 'grp_b', each containing four rows of data. The data includes various values for each column type, such as numerical values, character strings, factors, dates, times, datetimes, and currency amounts.

	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

.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"
  )
```

This screenshot shows the rendered HTML output of the R Markdown document. It displays the same gt table as the RStudio code chunk, but it is presented as a standard HTML table with a light gray background and white text. The table structure and data are identical to the one shown in the RStudio screenshot.

	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:

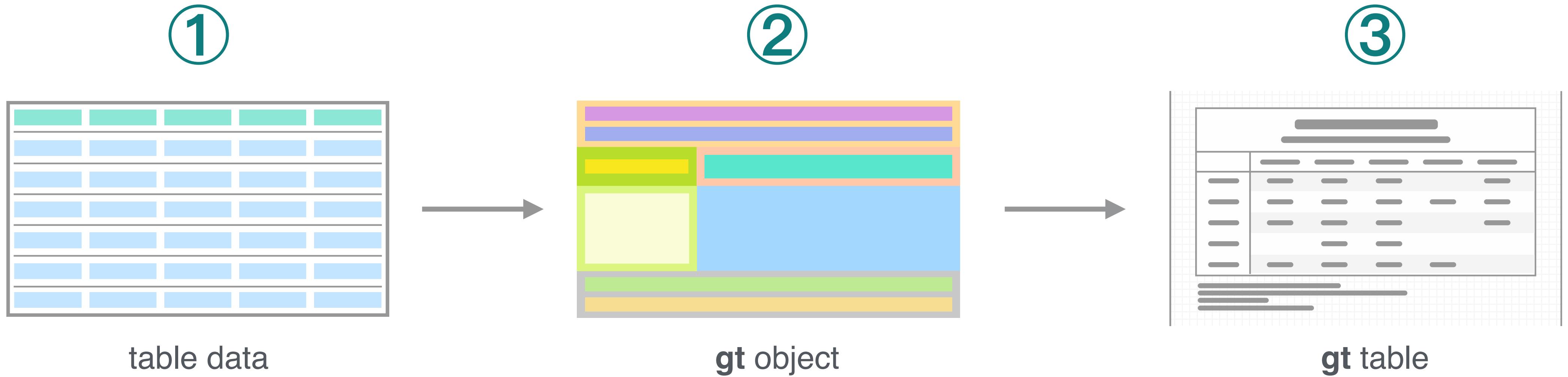
```
exibble %>%
  gt(
    rowname_col = "row",
    groupname_col = "group"
  ) %>%
  fmt_number(
    columns = num,
    decimals = 2
  ) %>%
  fmt_date(
    columns = date,
    date_style = 6
  ) %>%
  fmt_time(
    columns = time,
    time_style = 4
  ) %>%
  fmt_datetime(
    columns = datetime,
 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

This screenshot shows the rendered HTML output of the Quarto document. The table structure and data are identical to the ones in the RStudio and R Markdown screenshots. The table is displayed as a standard HTML table with a light gray background and white text.

	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

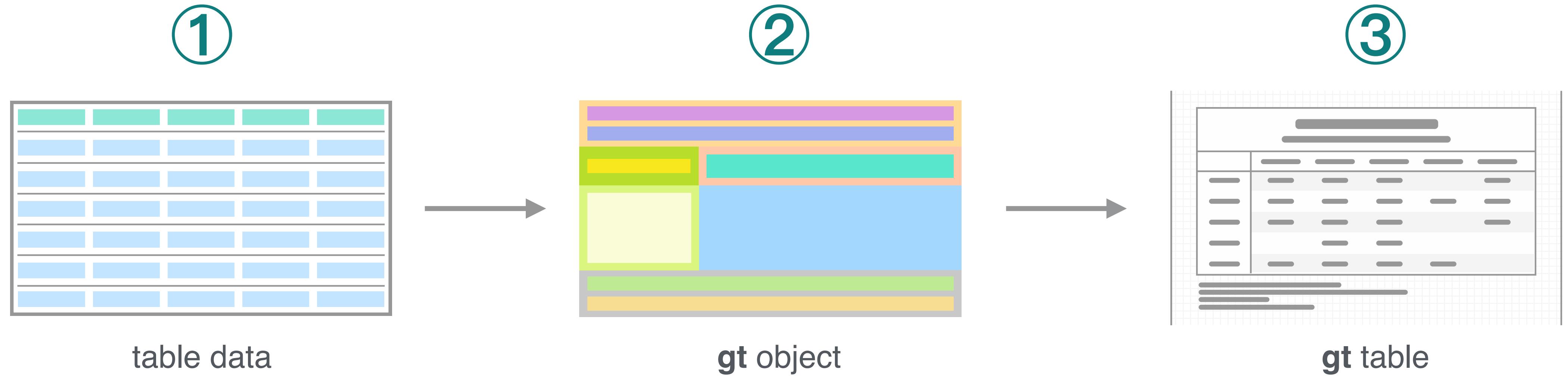
The Typical Workflow for Making Tables with **gt**.



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

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

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



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**.

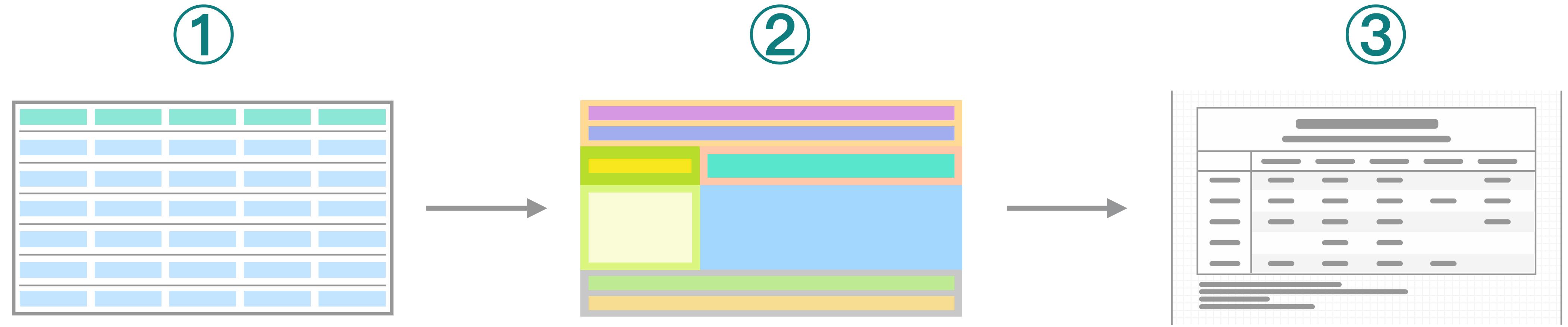


table data

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.

②

gt object

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

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

③

gt table

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

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

Understanding the different parts of a table.

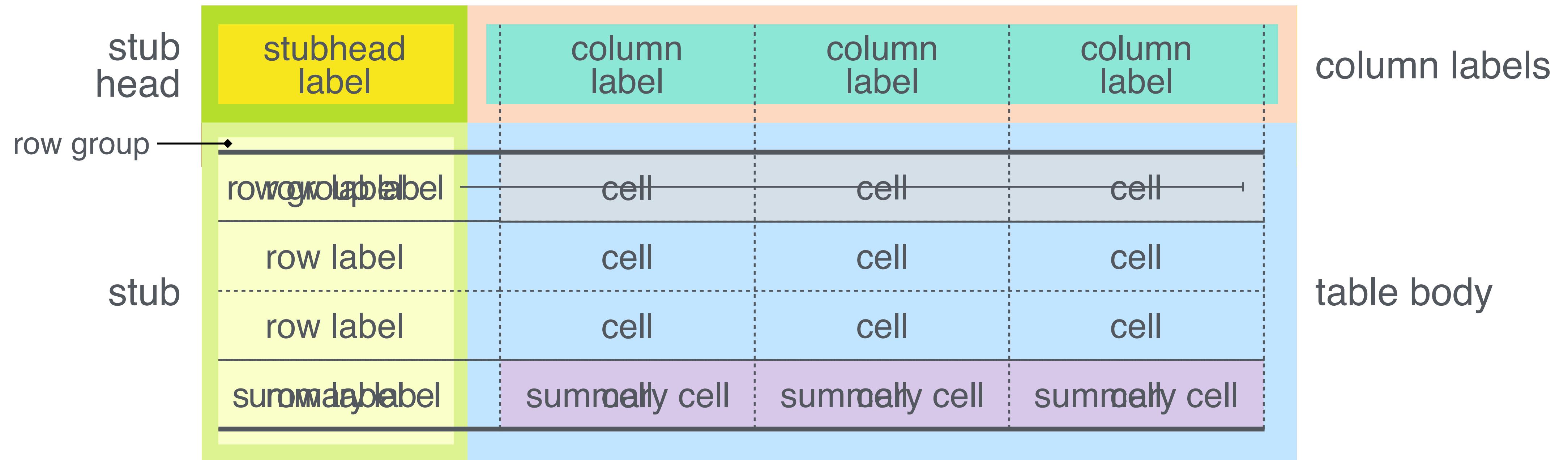
This is the most basic form of a **gt** table:

column label	column label	column label
cell	cell	cell

column labels

table body

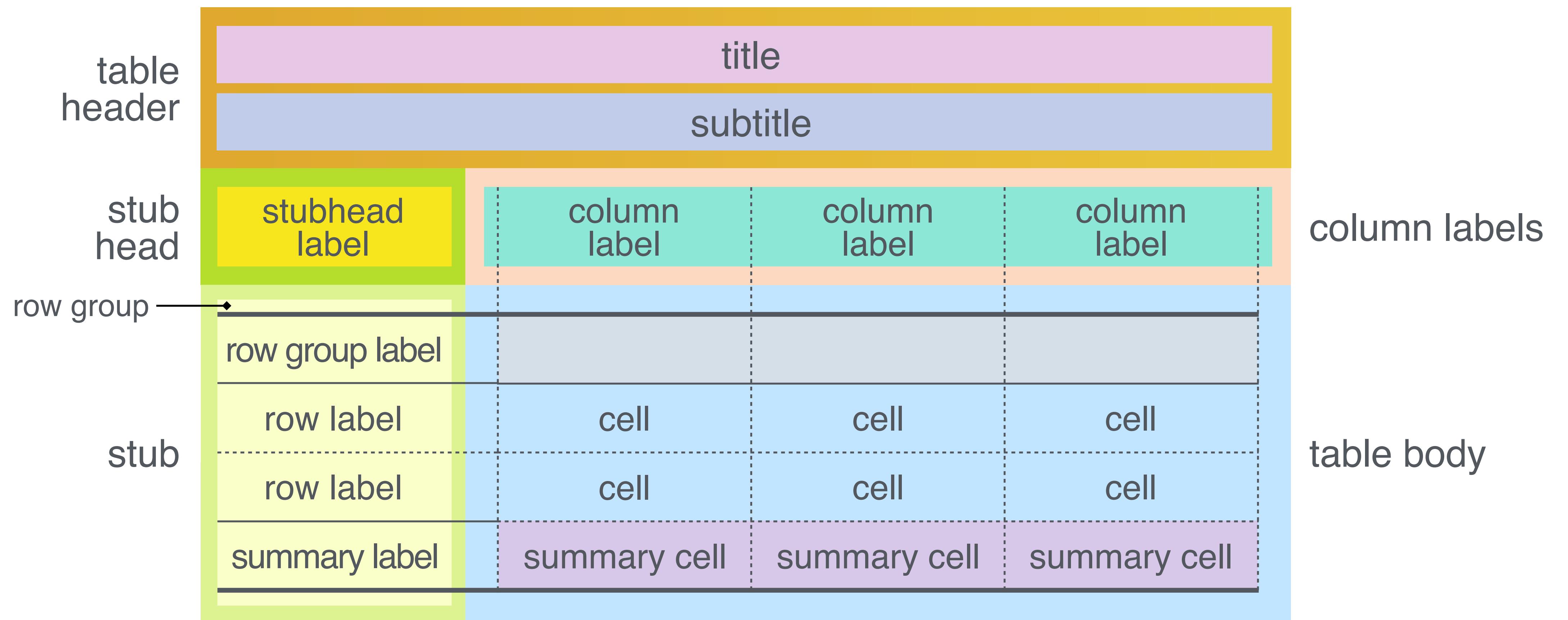
Understanding the different parts of a table.



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.

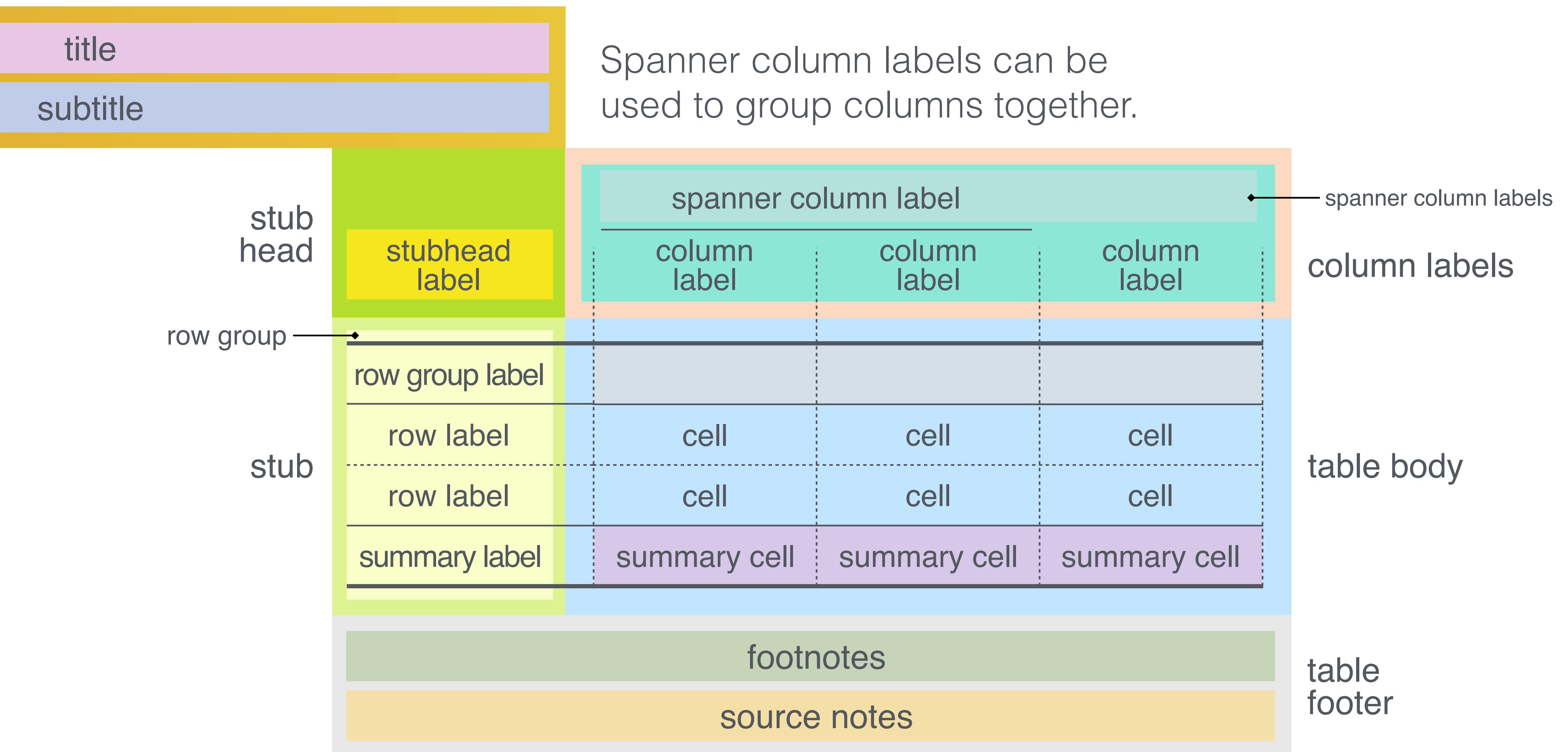
The diagram illustrates a complex table structure with the following components:

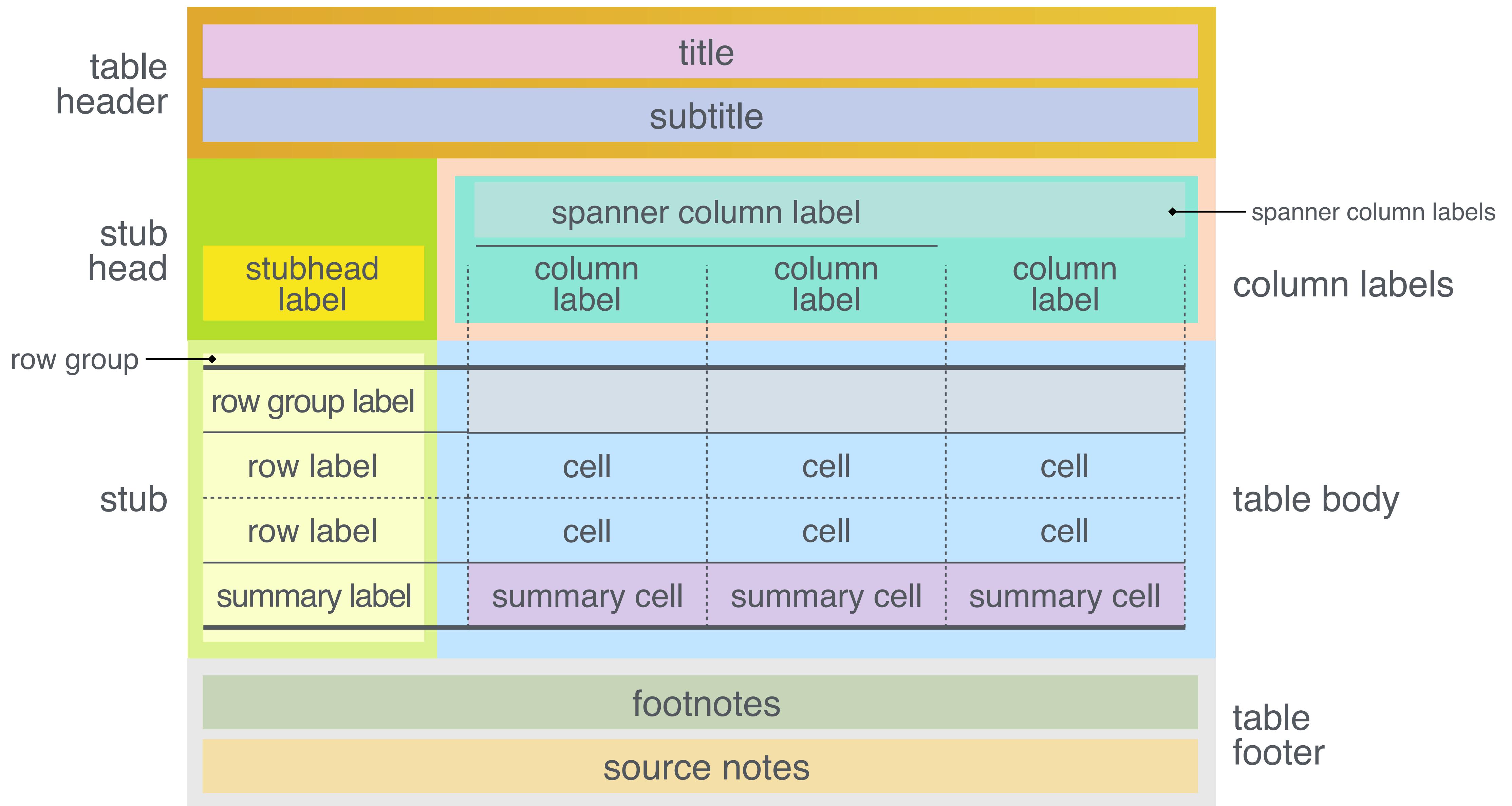
- table header**: The top-most section, containing the **title** (pink) and **subtitle** (light blue).
- stub head**: A row containing the **stubhead label** (yellow).
- row group**: A row containing the **row group label** (green).
- stub**: A row containing two **row label**s (yellow).
- table body**: The main data area, consisting of four columns of **cell**s (blue).
- summary label**: A row containing four **summary cell**s (purple).
- table footer**: The bottom-most section, containing the **footnotes** (light green) and **source notes** (yellow).

A horizontal arrow points from the label "row group" to the "row group label" cell. The "column labels" label is positioned to the right of the first column, and the "table body" label is positioned to the right of the second column.

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

Footnotes and source notes serve as useful annotations.





Demo

More information on **gt**.

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



Posit Cloud

gt Test Drive

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**



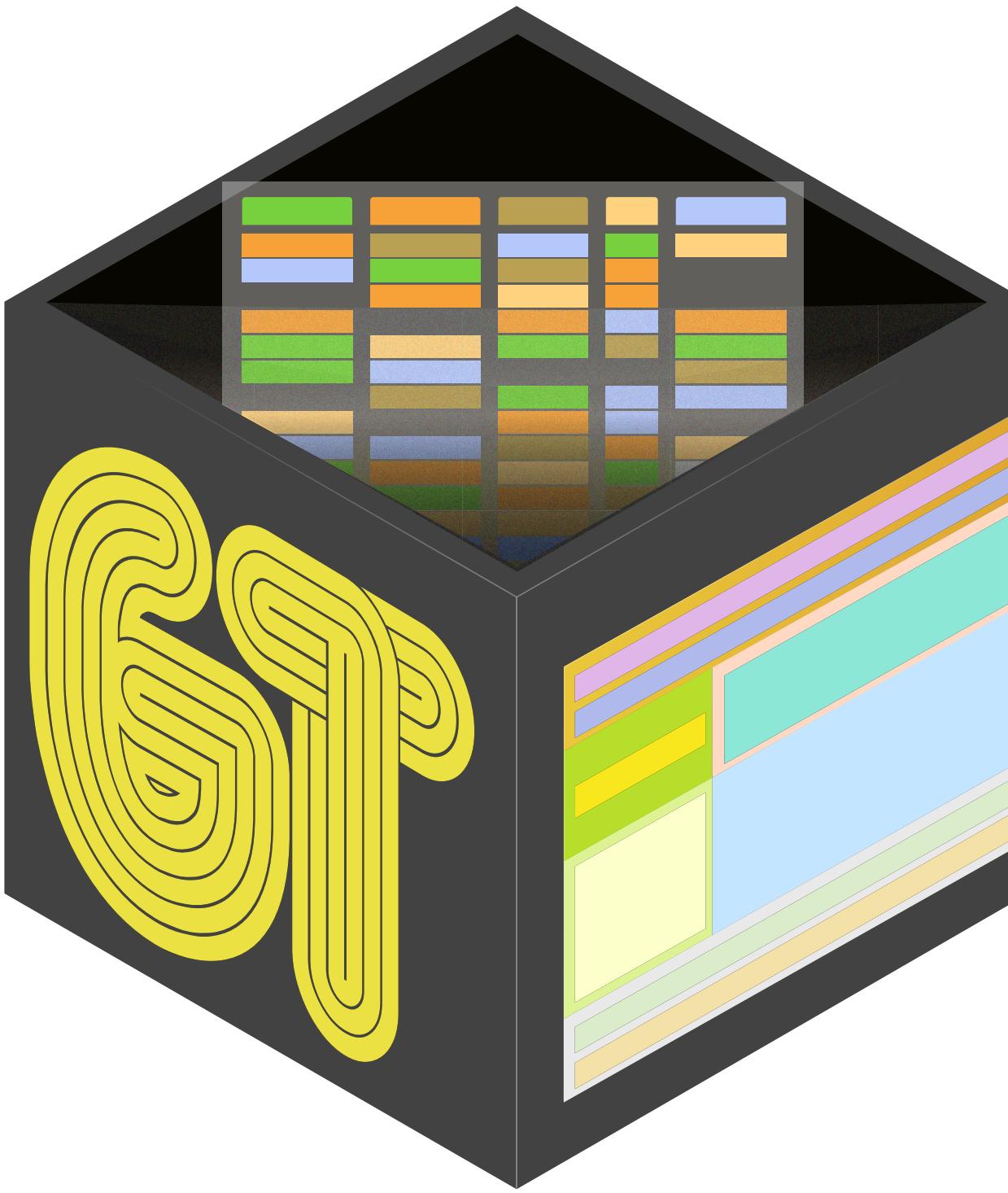
The link is available in the package README and the project website

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