Dynamic documents in R

Making tables in Quarto

2023-04-28

Working with descriptive tables

```r
descriptives <- diabetes |>
  filter(!is.na(glyhb)) |>
  mutate(
    diabetic = case_when(
      glyhb >= 6.5 ~ "Diabetic",
      glyhb < 6.5 ~ "Healthy",
      NA ~ NA_character_
    ),
    bmi = (weight / height^2) * 703
  ) |>
  group_by(diabetic) |>
  summarise(
    n = n(),
    across(
      c(glyhb, bmi, age),
      \(.x) mean(.x, na.rm = TRUE)
    ))
```

Descriptive statistics table

Standard `tibble` display

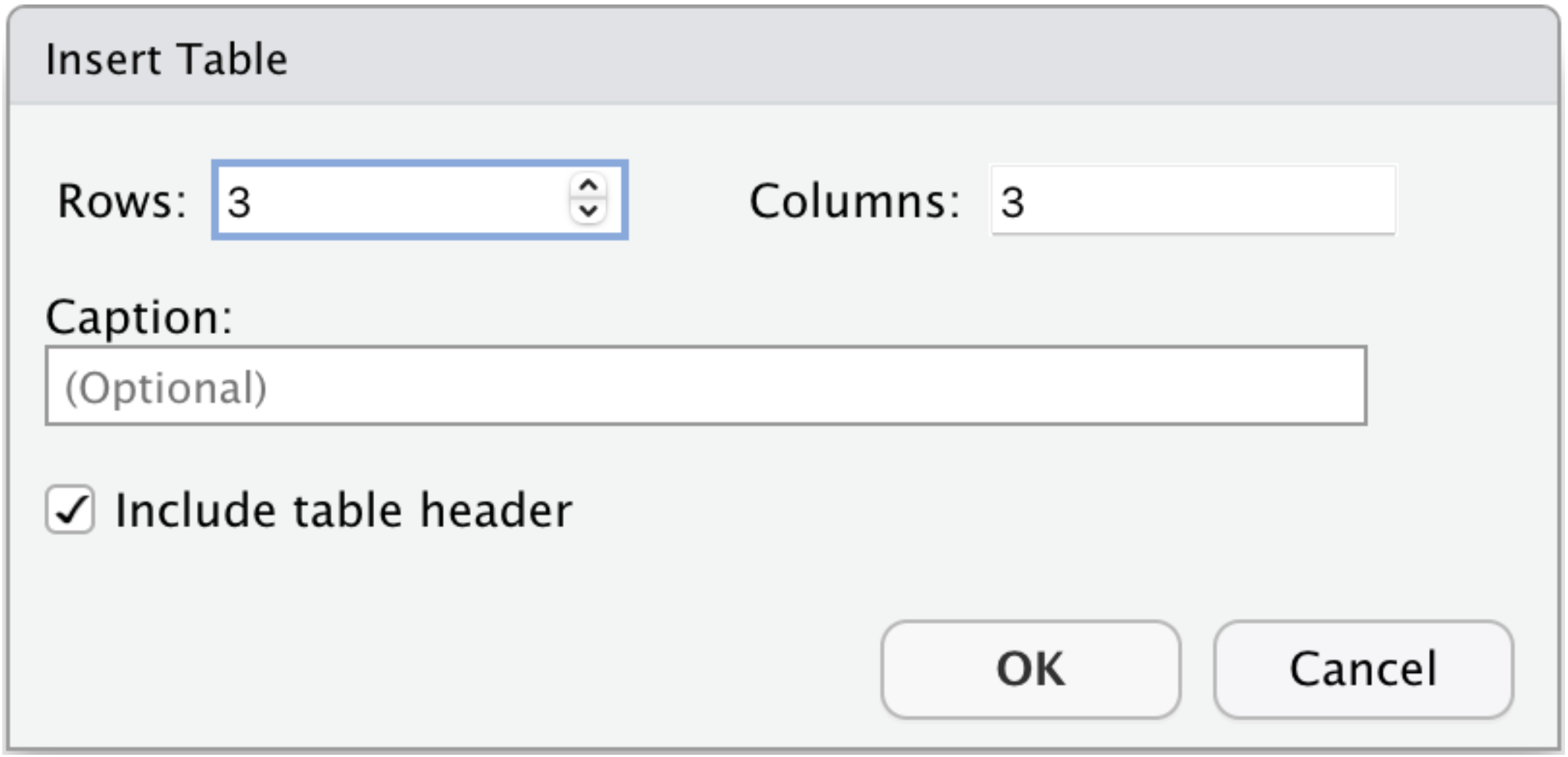
```
1 descriptives
```

```
# A tibble: 2 × 5
  diabetic      n glyhb   bmi   age
  <chr>     <int> <dbl> <dbl> <dbl>
1 Diabetic     65  9.89  30.8  58.4
2 Healthy     325  4.73  28.4  44.4
```

Useful in console, but less so for reports or presentation

Visual Editor tables

Visual Editor tables



Your turn 1

Using Visual Editor, create a markdown table that represents these data:

The Physicians' Health Study enrolled over 22,000 male physicians to study the effect of low-dose aspirin on myocardial infarctions (heart attacks). Of those who took aspirin, 129 had heart attacks, while 10,898 did not. Of those who took the placebo, 239 had heart attacks and 10,795 did not.

knitr::kable()

`kable()` creates formatted tables from rectangular objects (data.frames, matrices, and tibbles)

```
1 kable(descriptives)
```

diabetic	n	glyhb	bmi	age
Diabetic	65	9.886615	30.80235	58.43077
Healthy	325	4.730400	28.38114	44.44308

`kable()` options: column names

Supply a vector of new names with `col.names` argument

```
1 kable(
2   descriptives,
3   col.names = c("Diabetes Status", "N", "A1c", "BMI", "Age")
4 )
```

Diabetes Status	N	A1c	BMI	Age
Diabetic	65	9.886615	30.80235	58.43077
Healthy	325	4.730400	28.38114	44.44308

`kable()` options: number format

Use `digits` for decimal place

```
1 kable(
2   descriptives,
3   digits = 1
4 )
```

diabetic	n	glyhb	bmi	age
Diabetic	65	9.9	30.8	58.4
Healthy	325	4.7	28.4	44.4

Your turn 2

Using the `tidy()` function from `broom`, turn `response_model` and `marker_model` into dataframes, binding them together with `bind_rows()`

Turn the resulting dataframe into a table using `kable()`

Set the `eval` chunk option to `true`

Render

```
response_model <- glm(response ~ age + stage, data = trial,
                      family = binomial)
marker_model <- lm(marker ~ trt + stage + age, data = trial)
```

Your turn 2

```
1 models <- bind_rows(tidy(response_model), tidy(marker_model))
2 kable(models)
```

term	estimate	std.error	statistic	p.value
(Intercept)	-1.4862242	0.6202284	-2.3962530	0.0165637
age	0.0193911	0.0114681	1.6908683	0.0908620
stageT2	-0.5414264	0.4400027	-1.2305071	0.2185073
stageT3	-0.0595348	0.4504203	-0.1321761	0.8948450
stageT4	-0.2310863	0.4482284	-0.5155549	0.6061653
(Intercept)	0.8336800	0.2497219	3.3384338	0.0010317
trtDrug B	-0.2009397	0.1290364	-1.5572330	0.1212427
stageT2	0.3841096	0.1734658	2.2143255	0.0281131

term	estimate	std.error	statistic	p.value
stageT3	0.2838452	0.1895362	1.4975777	0.1360653
stageT4	0.1987877	0.1818389	1.0932073	0.2758227
age	-0.0004592	0.0044877	-0.1023233	0.9186185

gtsummary

Create publication-ready analytic and summary tables

Good support for wide variety of output formats

Built on the gt package

tbl_cross()

Creates a cross-tabulation of two categorical variables

```
library(gtsummary)
tbl_cross(
  data,
  row = x,
  col = y
)
```

tbl_cross()

```r
table_data <- diabetes |>
  mutate(
    # create diabetic category based on A1c and calculate bmi
    diabetic = case_when(
      glyhb >= 6.5 ~ "Diabetic",
      glyhb < 6.5 ~ "Healthy",
      NA ~ NA_character_
    ), bmi = (weight / height^2) * 703
  ) |>
  select(diabetic, age, gender, bmi)
```

tbl_cross()

```
1 tbl_cross(table_data, row = diabetic, col = gender)
```

	gender		
	female	male	Total
diabetic			
Diabetic	36	29	65
Healthy	192	133	325
Unknown	6	7	13
Total	234	169	403

`tbl_cross()`: variable names

```
1 library(labelled)
2 var_label(table_data$diabetic) <- "Diabetes Status"
3 var_label(table_data$diabetic)
```

```
[1] "Diabetes Status"
```

`tbl_cross()`: variable names

Set multiple variable lables with `list()`

```
var_label(table_data) <- list(
  gender = "Gender",
  diabetic = "Diabetes Status"
)
```

`tbl_cross()`: variable names

```
1  tbl_cross(table_data, diabetic, gender)
```

	Gender		
	female	male	Total
Diabetes Status			
Diabetic	36	29	65
Healthy	192	133	325
Unknown	6	7	13
Total	234	169	403

Your turn 3

Create a contingency table of `trial` (a dataset from gtsummary) using `tbl_cross()`: set `trt` to the rows and `response` to the columns.

Add a p-value with `add_p()`

Render

Your turn 3

```
trial |>
  tbl_cross(row = trt, col = response) |>
  add_p()
```

	Tumor Response				
	0	1	Unknown	Total	p-value[1]
Chemotherapy Treatment					0.7
Drug A	67	28	3	98	
Drug B	65	33	4	102	
Total	132	61	7	200	

[1] Fisher's exact test

`tbl_summary()`

Calculates descriptive statistics

Can split calculations by groups (i.e. categorical or dichotomous variables)

Sensible defaults; easily customized

tbl_summary()

```r
var_label(table_data) <- list(
  gender = "Gender",
  diabetic = "Diabetes Status",
  age = "Age",
  bmi = "BMI"
)

tbl_summary(table_data)
```

tbl_summary()

Characteristic	N = 403[1]
Diabetes Status	
Diabetic	65 (17%)
Healthy	325 (83%)
Unknown	13
Age	45 (34, 60)
Gender	
female	234 (58%)
male	169 (42%)
BMI	28 (24, 32)
Unknown	6

[1] n (%); Median (IQR)

`tbl_summary()`: split by groups

```
1 table_data |>
2   select(diabetic, age, gender, bmi) |>
3   tbl_summary(by = diabetic)
```

`tbl_summary()`: split by groups

Characteristic	Diabetic, N = 65[1]	Healthy, N = 325[1]
Age	59 (51, 65)	41 (32, 55)
Gender		
female	36 (55%)	192 (59%)
male	29 (45%)	133 (41%)
BMI	30 (26, 33)	27 (23, 32)
Unknown	2	4

[1] Median (IQR); n (%)

tbl_summary():digits

```
table_data |>
  select(diabetic, age, gender, bmi) |>
  tbl_summary(
    by = diabetic,
    digits = all_continuous() ~ 1
  )
```

tbl_summary():digits

Characteristic	Diabetic, N = 65[1]	Healthy, N = 325[1]
Age	59.0 (51.0, 65.0)	41.0 (32.0, 55.0)
Gender		
female	36 (55%)	192 (59%)
male	29 (45%)	133 (41%)
BMI	29.9 (26.4, 33.2)	27.5 (23.4, 31.8)
Unknown	2	4

[1] Median (IQR); n (%)

tbl_summary():missing_text

```
table_data |>
  select(diabetic, age, gender, bmi) |>
  tbl_summary(
    by = diabetic,
    missing_text = "(Missing)"
  )
```

tbl_summary():missing_text

Characteristic	Diabetic, N = 65[1]	Healthy, N = 325[1]
Age	59 (51, 65)	41 (32, 55)
Gender		
female	36 (55%)	192 (59%)
male	29 (45%)	133 (41%)
BMI	30 (26, 33)	27 (23, 32)
(Missing)	2	4

[1] Median (IQR); n (%)

`tbl_summary()`: tests & p-values

```
table_data |>
  select(diabetic, age, gender, bmi) |>
  tbl_summary(by = diabetic) |>
  add_p()
```

`tbl_summary()`: tests & p-values

Characteristic	Diabetic, N = 65[1]	Healthy, N = 325[1]	p-value[2]
Age	59 (51, 65)	41 (32, 55)	<0.001
Gender			0.6
female	36 (55%)	192 (59%)	
male	29 (45%)	133 (41%)	
BMI	30 (26, 33)	27 (23, 32)	0.003
Unknown	2	4	

[1] Median (IQR); n (%)

[2] Wilcoxon rank sum test; Pearson's Chi-squared test

tbl_summary(): statistic

```
tbl_summary(
  table_data,
  statistic = list(all_continuous() ~ "{mean} ({sd})")
)
```

Characteristic	N = 403[1]
Diabetes Status	
Diabetic	65 (17%)
Healthy	325 (83%)
Unknown	13
Age	47 (16)
Gender	
female	234 (58%)
male	169 (42%)
BMI	29 (7)
Unknown	6

[1] n (%); Mean (SD)

`tbl_summary()`: statistic

Access variables by name or type (`all_continuous()`, `all_categorical()`, `all_dichotomous()`, etc.)

Add other elements to table

Function	Description
`add_p()`	add p-values to the output comparing values across groups
`add_overall()`	add a column with overall summary statistics
`add_n()`	add a column with N (or N missing) for each variable
`add_difference()`	add column for difference between two group, confidence interval, and p-value
`add_stat_label()`	add label for the summary statistics shown in each row
`add_stat()`	generic function to add a column with user-defined values
`add_q()`	add a column of q values to control for multiple comparisons

gtsummary functions to format table

Function	Description
`modify_header()`	update column headers
`modify_footnote()`	update column footnote
`modify_spanning_header()`	update spanning headers
`bold_labels()`	bold variable labels
`bold_levels()`	bold variable levels
`italicize_labels()`	italicize variable labels
`italicize_levels()`	italicize variable levels
`bold_p()`	bold significant p-values

Your Turn 4

Create a summary table of `trial` by the `trt` variable.

Modify the label for `grade` to say "Tumor Grade"

Add an overall column and a p-value

Modify the table to use Mean (SD) and n (%) via the `statistic` argument: `list(all_continuous() ~ "{mean} ({sd})", all_categorical() ~ "{n} ({p}%)")`

Bold the labels

Modify the header with `modify_spanning_header(c("stat_1", "stat_2") ~ "**Treatment Received**")`

Your Turn 4

```
trial |>
  tbl_summary(
    by = trt,
    label = grade ~ "Tumor Grade",
    statistic = list(
      all_continuous() ~ "{mean} ({sd})",
      all_categorical() ~ "{n} ({p}%)"
    )
  ) |>
  bold_labels() |>
  modify_spanning_header(
    c("stat_1", "stat_2") ~ "**Treatment Received**"
  ) |>
  add_overall() |>
  add_p()
```

Your Turn 4

Characteristic	Overall, N = 200[1]	Treatment Received		p-value[2]
		Drug A, N = 98[1]	Drug B, N = 102[1]	
Age	47 (14)	47 (15)	47 (14)	0.7
Unknown	11	7	4	
Marker Level (ng/mL)	0.92 (0.86)	1.02 (0.89)	0.82 (0.83)	0.085
Unknown	10	6	4	
T Stage				0.9
T1	53 (26%)	28 (29%)	25 (25%)	
T2	54 (27%)	25 (26%)	29 (28%)	
T3	43 (22%)	22 (22%)	21 (21%)	
T4	50 (25%)	23 (23%)	27 (26%)	
Tumor Grade				0.9
I	68 (34%)	35 (36%)	33 (32%)	
II	68 (34%)	32 (33%)	36 (35%)	
III	64 (32%)	31 (32%)	33 (32%)	
Tumor Response	61 (32%)	28 (29%)	33 (34%)	0.5
Unknown	7	3	4	
Patient Died	112 (56%)	52 (53%)	60 (59%)	0.4
Months to Death/Censor	19.6 (5.3)	20.2 (5.0)	19.0 (5.5)	0.14

[1] Mean (SD); n (%)

[2] Wilcoxon rank sum test; Pearson's Chi-squared test

Other `tbl_summary()` arguments

See the other options with *`?tbl_summary()`*

Argument	Description
`label`	specify the variable labels printed in table
`type`	specify the variable type (e.g. continuous, categorical, etc.)
`statistic`	change the summary statistics presented
`digits`	number of digits the summary statistics will be rounded to
`missing`	whether to display a row with the number of missing observations
`missing_text`	text label for the missing number row
`sort`	change the sorting of categorical levels by frequency
`percent`	print column, row, or cell percentages
`include`	list of variables to include in summary table

tbl_regression()

```
model_data <- diabetes |>
  mutate(bmi = (weight / height^2) * 703)
glyhb_model <- lm(glyhb ~ bmi + age, data = model_data)
tbl_regression(glyhb_model)
```

Characteristic	Beta	95% CI[1]	p-value
bmi	0.04	0.01, 0.08	0.006
age	0.05	0.03, 0.06	<0.001

[1] CI = Confidence Interval

`tbl_regression()`: variable labels

Add labels just like `tbl_summary()`

```
var_label(model_data) <- list(
  bmi = "BMI",
  glyhb = "A1c",
  age = "Age"
)
glyhb_model <- lm(glyhb ~ bmi + age, data = model_data)
tbl_regression(glyhb_model)
```

Characteristic	Beta	95% CI[1]	p-value
BMI	0.04	0.01, 0.08	0.006
Age	0.05	0.03, 0.06	<0.001

[1] CI = Confidence Interval

`tbl_regression()`: other options

Argument	Description
`label`	modify variable labels in table
`exponentiate`	exponentiate model coefficients
`include`	names of variables to include in output. Default is all variables
`show_single_row`	By default, categorical variables are printed on multiple rows. If a variable is dichotomous and you wish to print the regression coefficient on a single row, include the variable name(s) here.
`conf.level`	confidence level of confidence interval
`intercept`	indicates whether to include the intercept
`estimate_fun`	function to round and format coefficient estimates
`pvalue_fun`	function to round and format p-values
`tidy_fun`	function to specify/customize tidier function

`tbl_regression()`: Model statistics

Add model statistics from `broom::glance()`

```
1 tbl_regression(glyhb_model) |>
2   add_glance_source_note()
```

Characteristic	Beta	95% CI[1]	p-value
BMI	0.04	0.01, 0.08	0.006
Age	0.05	0.03, 0.06	<0.001

[1] CI = Confidence Interval

R^2 = 0.132; Adjusted R^2 = 0.127; Sigma = 2.06; Statistic = 28.9; p-value = <0.001; df = 2; Log-likelihood = -821; AIC = 1,650; BIC = 1,666; Deviance = 1,617; Residual df = 381; No. Obs. = 384

`tbl_regression()`: other add options

Function	Description
`add_global_p()`	adds the global p-value for a categorical variables
`add_glance_source_note()`	adds statistics from `glance()` as source note
`add_vif()`	adds column of the variance inflation factors (VIF)
`add_q()`	add a column of q values to control for multiple comparisons

tbl_regression(): format options

```
1 tbl_regression(glyhb_model) |>
2   add_glance_source_note() |>
3   bold_p() |>
4   bold_labels()
```

Characteristic	Beta	95% CI[1]	p-value
BMI	0.04	0.01, 0.08	**0.006**
Age	0.05	0.03, 0.06	**<0.001**

[1] CI = Confidence Interval

R^2 = 0.132; Adjusted R^2 = 0.127; Sigma = 2.06; Statistic = 28.9; p-value = <0.001; df = 2; Log-likelihood = -821; AIC = 1,650; BIC = 1,666; Deviance = 1,617; Residual df = 381; No. Obs. = 384

Adding captions to tables

Use the `tbl-cap` chunk option

````
```{r}
#| label: tbl-with-caption
#| tbl-cap: "Association of BMI and A1c adjusted for age"
tbl_regression(glyhb_model) |>
  add_glance_source_note() |>
  bold_p() |>
  bold_labels()
```
````

Your turn 5

Create a regression table for `response model`; exponentiate the output with `exponentiate = TRUE`

Add a global P-value

Add the model statistics via `add_glance_source_note()`

Italicize the variable levels

Merge the table you just made with `marker_table` using `tbl_merge()`

Your turn 5

```
response_table <- response_model |>
  tbl_regression(
    exponentiate = TRUE
  ) |>
  add_global_p() |>
  add_glance_source_note() |>
  italicize_levels()

marker_table <- marker_model |>
  tbl_regression() |>
  add_global_p()

both_tables <- list(response_table, marker_table)
tbl_merge(both_tables)
```

Your turn 5

	Table 1			Table 2		
Characteristic	**OR**[1]	**95% CI**[1]	**p-value**	**Beta**	**95% CI**[1]	**p-value**
Age	1.02	1.00, 1.04	0.087	0.00	-0.01, 0.01	>0.9
T Stage			0.6			0.2
T1	—	—		—	—	
T2	0.58	0.24, 1.37		0.38	0.04, 0.73	
T3	0.94	0.39, 2.28		0.28	-0.09, 0.66	
T4	0.79	0.33, 1.90		0.20	-0.16, 0.56	
Chemotherapy Treatment						0.12
Drug A				—	—	
Drug B				-0.20	-0.46, 0.05	

[1] OR = Odds Ratio, CI = Confidence Interval

Null deviance = 229; Null df = 182; Log-likelihood = -112; AIC = 234; BIC = 250; Deviance = 224; Residual df = 178; No. Obs. = 183

Cross-referencing tables

1. A table caption (`#| tbl-cap: "Title"`)
2. A labeled code chunk prefixed with `tbl-` (`#| label: tbl-chunk-name`)
3. Reference with `@tbl-chunk-name`

Your Turn 6

Add a caption to the merged table above

Reference that table below using this format:
`@tbl-name-of-chunk`

Render

Your Turn 6

```
1 ```{r}
2 #| label: table-two
3 #| tbl-cap: "Regression models of participant markets and respor
4 tbl_merge(both_tables)
5 ```
```

Your Turn 6

Regression models of participant markets and response

	Table 1			Table 2		
Characteristic	**OR[1]**	**95% CI[1]**	**p-value**	**Beta**	**95% CI[1]**	**p-value**
Age	1.02	1.00, 1.04	0.087	0.00	-0.01, 0.01	>0.9
T Stage			0.6			0.2
T1	—	—		—	—	
T2	0.58	0.24, 1.37		0.38	0.04, 0.73	
T3	0.94	0.39, 2.28		0.28	-0.09, 0.66	
T4	0.79	0.33, 1.90		0.20	-0.16, 0.56	
Chemotherapy Treatment						0.12
Drug A				—	—	
Drug B				-0.20	-0.46, 0.05	

[1] OR = Odds Ratio, CI = Confidence Interval

Null deviance = 229; Null df = 182; Log-likelihood = -112; AIC = 234; BIC = 250; Deviance = 224; Residual df = 178; No. Obs. = 183

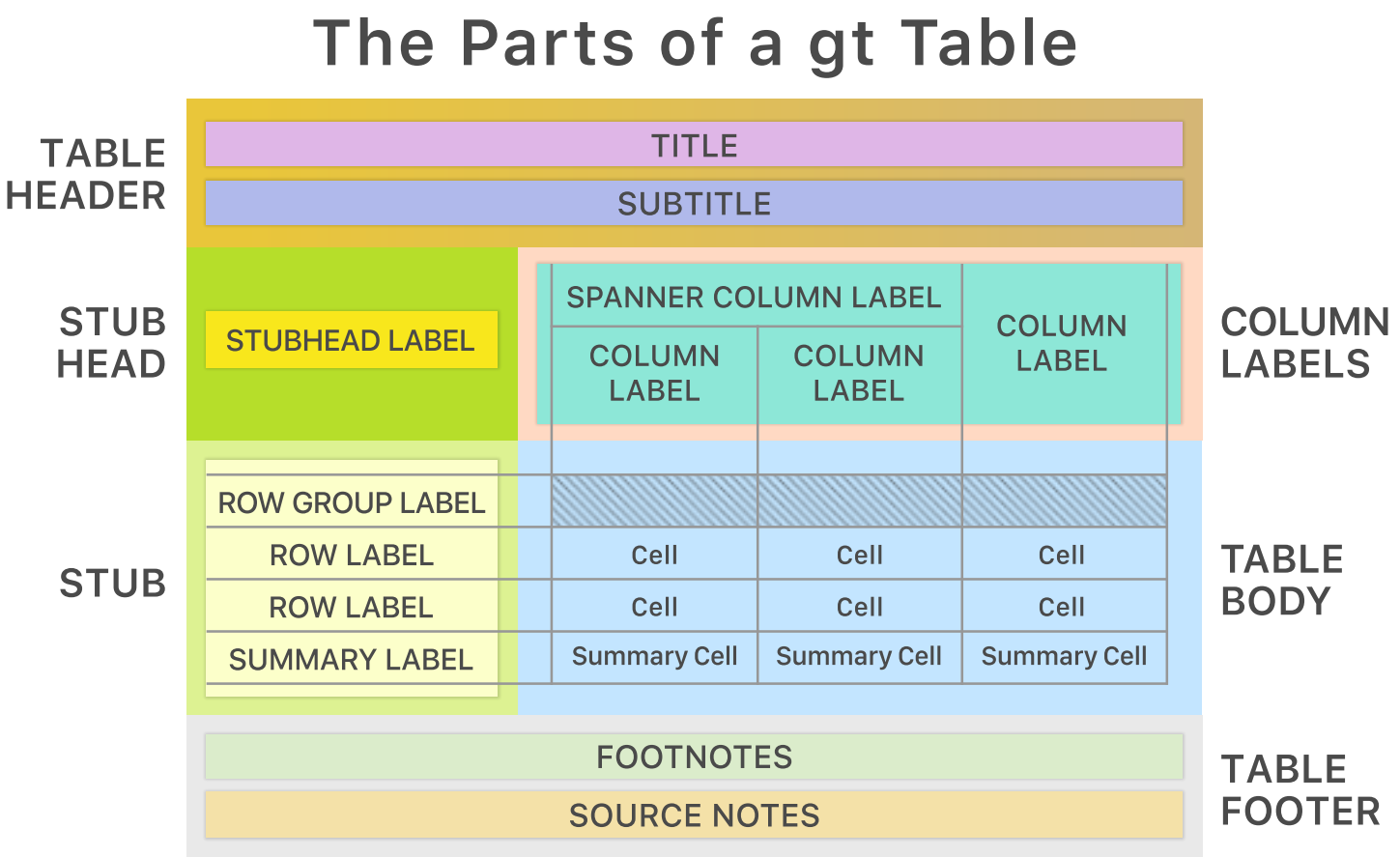
In Table 1, we show two regression models: a linear model of participant markets and a logistic regression model of participant response.

gtsummary: Output formats

Print Engine	Function	HTML	Word	PDF	RTF
gt	`as_gt()`				
flextable	`as_flex_table()`				
huxtable	`as_hux_table()`				
kableExtra	`as_kable_extra()`				
kable	`as_kable()`				
tibble	`as_tibble()`				

Also check out the gt package!

“construct a wide variety of useful tables with a cohesive set of table parts”



Resources

Quarto: Tables: Quarto documentation on tables

gtsummary Website: Many vignettes to learn more about gtsummary

gt Website: Vignettes and a short course on learning gt