Creating Tables with kable()

Andrew Zieffler

This work is licensed under a
Creative Commons Attribution
4.0 International License.

Create an R Project and an RMD File

- Create a new R Project by creating a new directory. (`New Project... > New Directory`)
  - Select New Project
  - Give the project a name
  - Use Browse... to determine where this directory will be saved on your computer

- Create a new RMD document that will render to HTML (`File > New File... > R Markdown...`)
  - Change the YAML to include a title, your name as author, and the date.
  - Delete the pre-populated main part of the RMD document.

- Save the RMD file in your newly created directory.

- Knit the document.

As you work through the remainder of these notes, you will be adding content to the main part of this document.

Tables are Rectangular Arrangements

Any table is essentially a rectangular layout (rows and columns) of information. Below I show two common tables of statistical output. I have also added guide-lines to show the rectangular display of information **within** each of these tables.

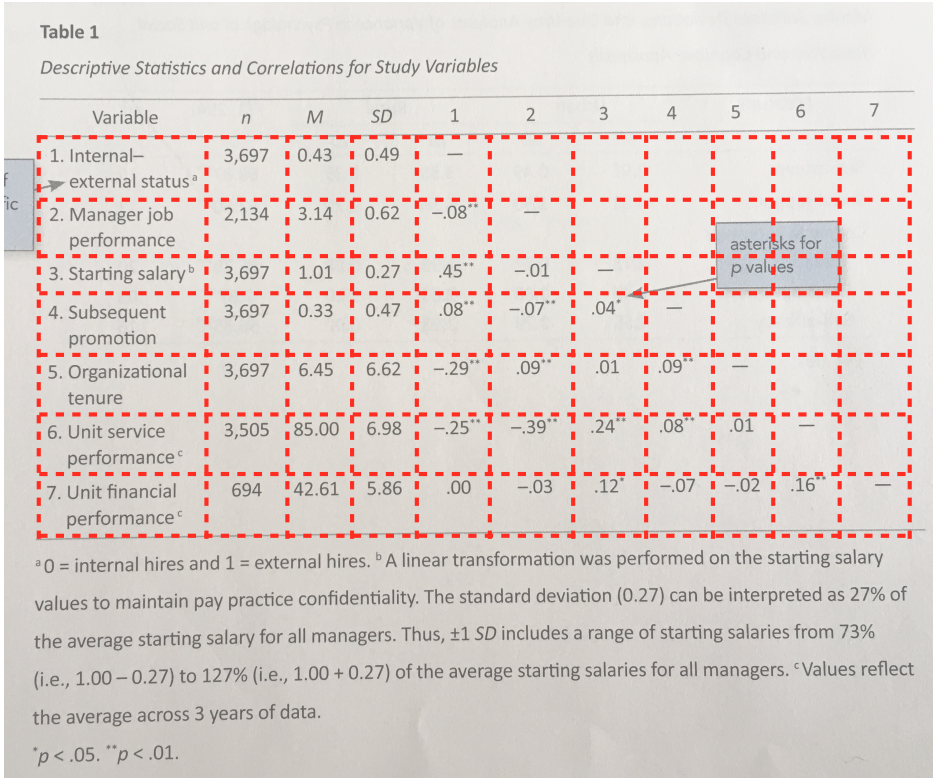


Table 1

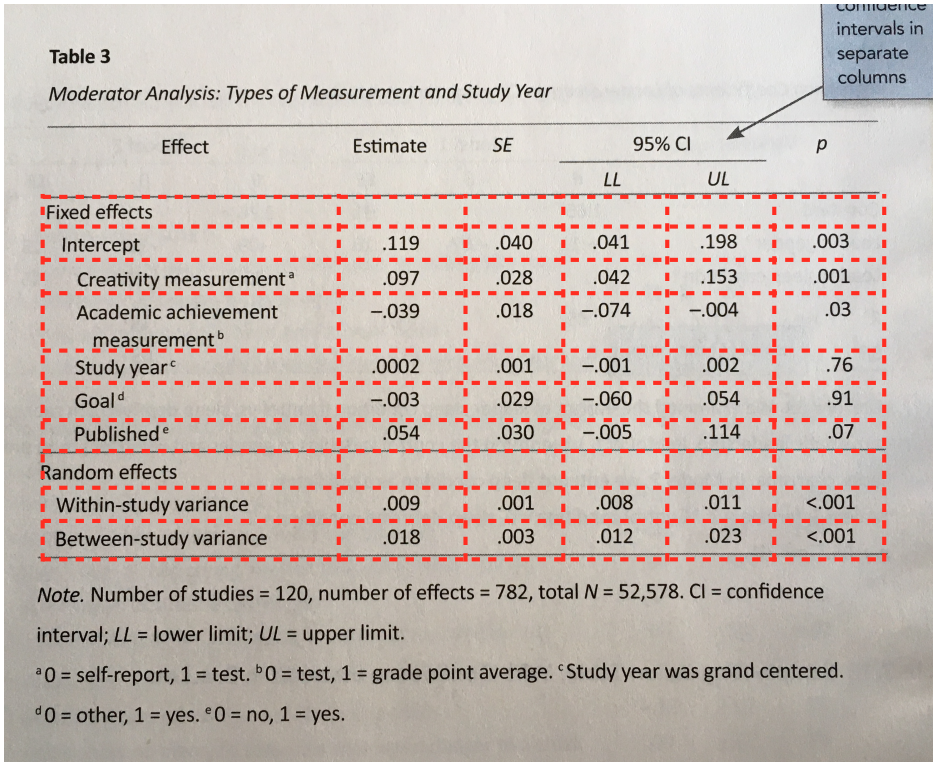
Descriptive Statistics and Correlations for Study Variables

Variable	*n*	*M*	*SD*	1	2	3	4	5	6	7
1. Internal–external status[a]	3,697	0.43	0.49	—						
2. Manager job performance	2,134	3.14	0.62	−.08**	—					
3. Starting salary[b]	3,697	1.01	0.27	.45**	−.01	—				
4. Subsequent promotion	3,697	0.33	0.47	.08**	−.07**	.04*	—			
5. Organizational tenure	3,697	6.45	6.62	−.29**	.09**	.01	.09**	—		
6. Unit service performance[c]	3,505	85.00	6.98	−.25**	−.39**	.24**	.08**	.01	—	
7. Unit financial performance[c]	694	42.61	5.86	.00	−.03	.12*	−.07	−.02	.16**	—

asterisks for *p* values

[a] 0 = internal hires and 1 = external hires. [b] A linear transformation was performed on the starting salary values to maintain pay practice confidentiality. The standard deviation (0.27) can be interpreted as 27% of the average starting salary for all managers. Thus, ±1 *SD* includes a range of starting salaries from 73% (i.e., 1.00 – 0.27) to 127% (i.e., 1.00 + 0.27) of the average starting salaries for all managers. [c] Values reflect the average across 3 years of data.

*$p < .05$. **$p < .01$.



confidence intervals in separate columns

Table 3

Moderator Analysis: Types of Measurement and Study Year

Effect	Estimate	*SE*	95% CI		*p*
			LL	*UL*	
Fixed effects					
Intercept	.119	.040	.041	.198	.003
Creativity measurement[a]	.097	.028	.042	.153	.001
Academic achievement measurement[b]	−.039	.018	−.074	−.004	.03
Study year[c]	.0002	.001	−.001	.002	.76
Goal[d]	−.003	.029	−.060	.054	.91
Published[e]	.054	.030	−.005	.114	.07
Random effects					
Within-study variance	.009	.001	.008	.011	<.001
Between-study variance	.018	.003	.012	.023	<.001

Note. Number of studies = 120, number of effects = 782, total *N* = 52,578. CI = confidence interval; *LL* = lower limit; *UL* = upper limit.

[a] 0 = self-report, 1 = test. [b] 0 = test, 1 = grade point average. [c] Study year was grand centered. [d] 0 = other, 1 = yes. [e] 0 = no, 1 = yes.

Photo Credit: Table images taken from American Psychological Association (2020). *Publication manual of the American Psychological Association* (7th ed.). Author.

Anatomy of a Table

To illustrate how to create a table in the RMD document, we will attempt to recreate the following table. I have identified different aspect of this table that I will refer to throughout the notes.

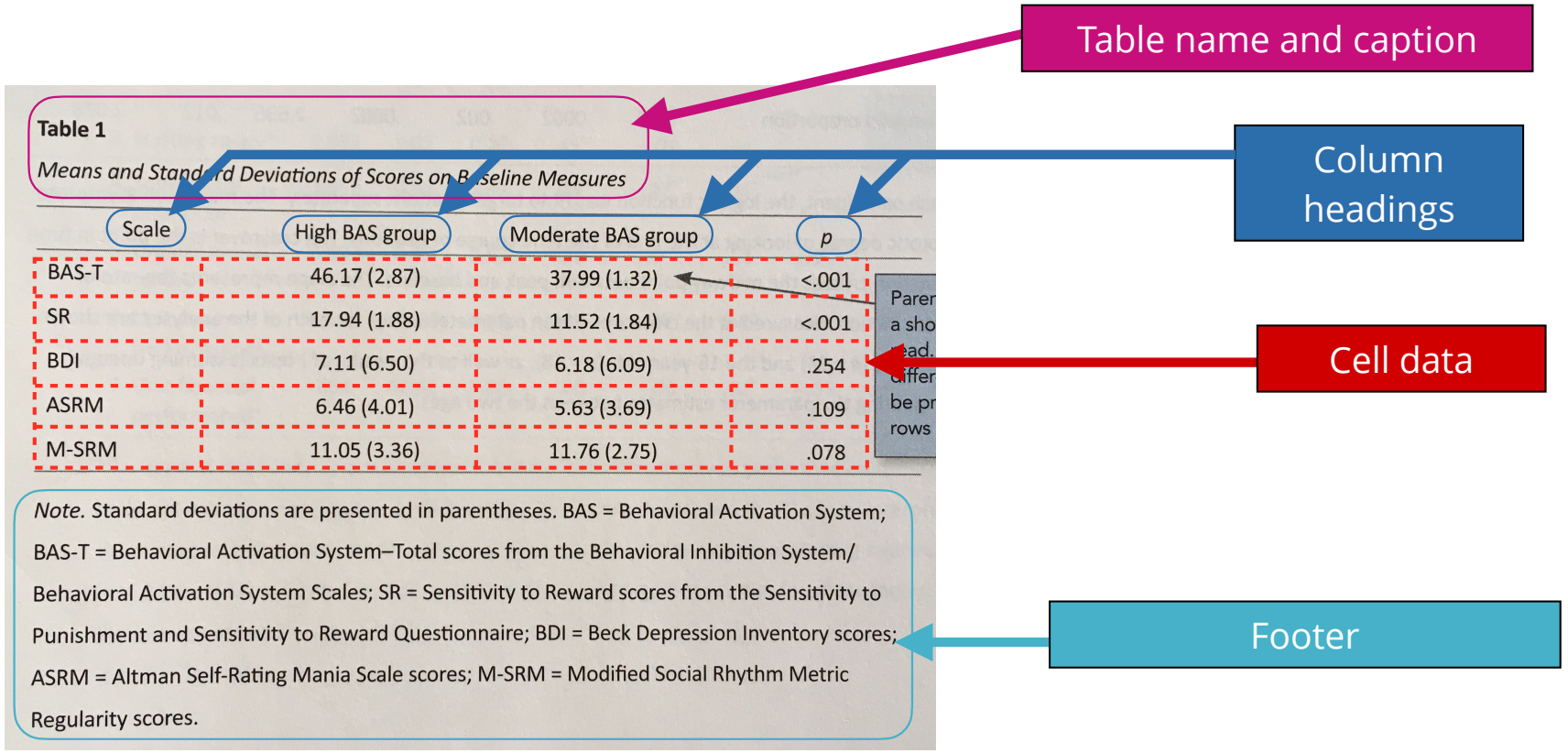


Photo Credit: Table images taken from American Psychological Association (2020). *Publication manual of the American Psychological Association* (7th ed.). Author.

Creating a Table

We can use the fact that **data frames** are also rectangular displays of information to create a table in the RMD document. The `data.frame()` function is used to enter the cell information from each column. This is, of course, done in a code chunk.

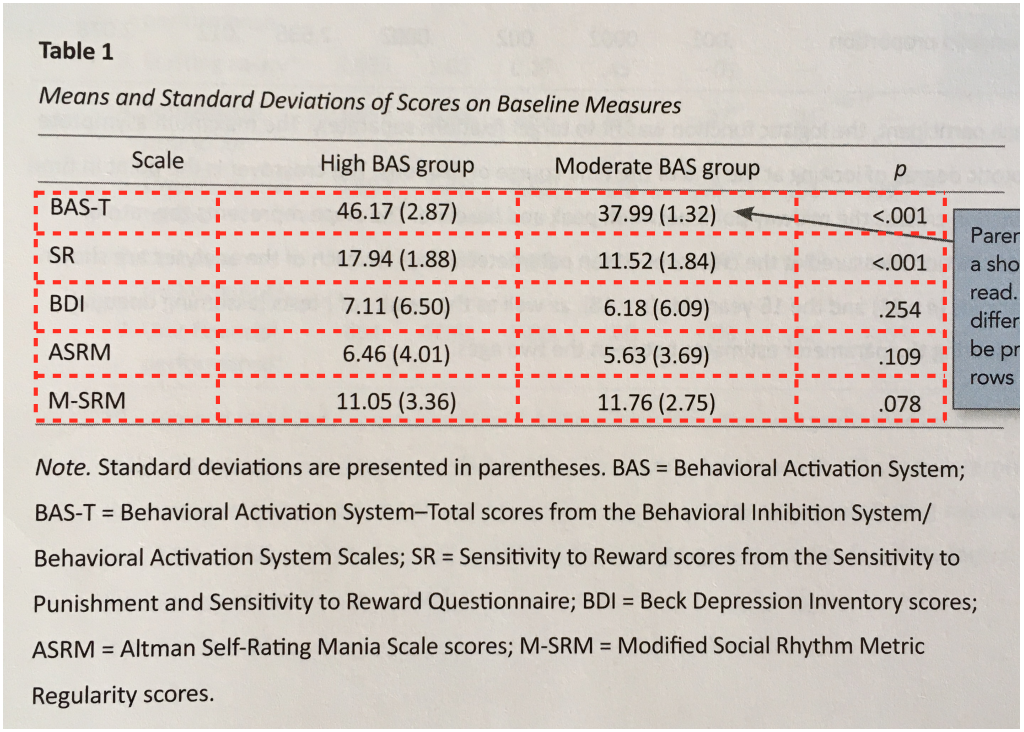


Table 1

Means and Standard Deviations of Scores on Baseline Measures

Scale	High BAS group	Moderate BAS group	*p*
BAS-T	46.17 (2.87)	37.99 (1.32)	<.001
SR	17.94 (1.88)	11.52 (1.84)	<.001
BDI	7.11 (6.50)	6.18 (6.09)	.254
ASRM	6.46 (4.01)	5.63 (3.69)	.109
M-SRM	11.05 (3.36)	11.76 (2.75)	.078

Paren
a sho
read.
differ
be pr
rows

Note. Standard deviations are presented in parentheses. BAS = Behavioral Activation System; BAS-T = Behavioral Activation System–Total scores from the Behavioral Inhibition System/ Behavioral Activation System Scales; SR = Sensitivity to Reward scores from the Sensitivity to Punishment and Sensitivity to Reward Questionnaire; BDI = Beck Depression Inventory scores; ASRM = Altman Self-Rating Mania Scale scores; M-SRM = Modified Social Rhythm Metric Regularity scores.

```
```{r table_01}
# Input cell information
tab_01 = data.frame(
  scale = c("BAS-T", "SR", "BDI", "ASRM", "M-SRM"),
  high = c("46.17 (2.87)", "17.94 (1.88)",
           "7.11 (6.50)", "6.46 (4.01)", "11.05 (3.36)"),
  mod = c("37.99 (1.32)", "11.52 (1.84)",
          "6.18 (6.09)", "5.63 (3.69)", "11.76 (2.75)"),
  p = c("<.001", "<.001", ".254", ".109", ".078")

)

# Show output
tab_01
```
```

```
##   scale          high          mod     p
## 1 BAS-T 46.17 (2.87) 37.99 (1.32) <.001
## 2    SR 17.94 (1.88) 11.52 (1.84) <.001
## 3   BDI  7.11 (6.50)  6.18 (6.09)  .254
## 4  ASRM  6.46 (4.01)  5.63 (3.69)  .109
## 5 M-SRM 11.05 (3.36) 11.76 (2.75)  .078
```

Photo Credit: Table images taken from American Psychological Association (2020). *Publication manual of the American Psychological Association* (7th ed.). Author.

Once you have the cell information in a data frame, **all the rest** of the syntax is formatting, adding a caption, changing column names, etc. There are several R packages that can be used to add this formatting. Here I will illustrate the use of the `kable()` function from the `{knitr}` package.

````
```{r table_01}
# Load package
library(knitr)
library(tidyverse) #To use %>%


# Input cell information
tab_01 = data.frame(
  scale = c("BAS-T", "SR", "BDI", "ASRM", "M-SRM"),
  high = c("46.17 (2.87)", "17.94 (1.88)",
           "7.11 (6.50)", "6.46 (4.01)", "11.05 (3.36)"),
  mod = c("37.99 (1.32)", "11.52 (1.84)",
          "6.18 (6.09)", "5.63 (3.69)", "11.76 (2.75)"),
  p = c("<.001", "<.001", ".254", ".109", ".078")

)


# Use kable() to format output
tab_01 %>%
  kable(
    format = "html"
)
```
````

scale	high	mod	p
BAS-T	46.17 (2.87)	37.99 (1.32)	<.001
SR	17.94 (1.88)	11.52 (1.84)	<.001
BDI	7.11 (6.50)	6.18 (6.09)	.254
ASRM	6.46 (4.01)	5.63 (3.69)	.109
M-SRM	11.05 (3.36)	11.76 (2.75)	.078

The `format="html"` argument to the `kable()` function formats the table using HTML. We can add additional arguments to the `kable()` function to further format the output.

Photo Credit: Table images taken from American Psychological Association (2020). *Publication manual of the American Psychological Association* (7th ed.). Author.

Changing Column Names

To change the column names, we use the `col.names=` argument. We provide this with a vector of column names using the `c()` function.

```
# Use kable() to format output
tab_01 %>%
  kable(
    format = "html",
    col.names = c("Scale", "High BAS group", "Moderate BAS group", "*p*")
)
```

Note that we can use Markdown syntax to format column names as well. Here we italicized the name of the last column.

Scale	High BAS group	Moderate BAS group	*p*
BAS-T	46.17 (2.87)	37.99 (1.32)	<.001
SR	17.94 (1.88)	11.52 (1.84)	<.001
BDI	7.11 (6.50)	6.18 (6.09)	.254
ASRM	6.46 (4.01)	5.63 (3.69)	.109
M-SRM	11.05 (3.36)	11.76 (2.75)	.078

Adding a Title and Caption

To add a title and caption, we use the `caption=` argument. We provide this with a character string.

```
# Use kable() to format output
tab_01 %>%
  kable(
    format = "html",
    col.names = c("Scale", "High BAS group", "Moderate BAS group", "*p*"),
    caption = "**Table 1** <br /> *Means and Standard Deviations of Scores on Baseline Measures*"
)
```

The `<br />` is HTML syntax that includes a line break.

Table 1

Means and Standard Deviations of Scores on Baseline Measures

Scale	High BAS group	Moderate BAS group	*p*
BAS-T	46.17 (2.87)	37.99 (1.32)	<.001
SR	17.94 (1.88)	11.52 (1.84)	<.001
BDI	7.11 (6.50)	6.18 (6.09)	.254
ASRM	6.46 (4.01)	5.63 (3.69)	.109
M-SRM	11.05 (3.36)	11.76 (2.75)	.078

Column Alignment

To change the alignment of the information in the table, we use the `align=` argument. We provide this with a vector of alignment values (`"l"`, `"c"`, or “`r"`) using the `c()` function.

```
# Use kable() to format output
tab_01 %>%
  kable(
    format = "html",
    col.names = c("Scale", "High BAS group", "Moderate BAS group", "*p*"),
    caption = "**Table 1** <br /> *Means and Standard Deviations of Scores on Baseline Measures*",
    align = c("l", "c", "c", "c")
)
```

Although it is difficult to see, the alignment operates on both the cell information and the column name. For example, the label “*p*” and the associated *p*-values in the last column are all centered.

Table 1

Means and Standard Deviations of Scores on Baseline Measures

Scale	High BAS group	Moderate BAS group	*p*
BAS-T	46.17 (2.87)	37.99 (1.32)	<.001
SR	17.94 (1.88)	11.52 (1.84)	<.001
BDI	7.11 (6.50)	6.18 (6.09)	.254
ASRM	6.46 (4.01)	5.63 (3.69)	.109
M-SRM	11.05 (3.36)	11.76 (2.75)	.078

Super-Formatting with kableExtra

The {kableExtra} package includes a lot of functionality to enhance tables created with the kable() function. One of those functions is the kable_styling() function. We pipe the output of the kable() function into the kable_styling() function. (Don't forget to load the {kableExtra} package first.) The full syntax (so far) in our code chunk is shown below.

```
```{r table_01}
# Load package
library(knitr)
library(kableExtra)
library(tidyverse) #To use %>%


# Input cell information
tab_01 = data.frame(
  scale = c("BAS-T", "SR", "BDI", "ASRM", "M-SRM"),
  high = c("46.17 (2.87)", "17.94 (1.88)",
           "7.11 (6.50)", "6.46 (4.01)", "11.05 (3.36)"),
  mod = c("37.99 (1.32)", "11.52 (1.84)",
          "6.18 (6.09)", "5.63 (3.69)", "11.76 (2.75)"),
  p = c("<.001", "<.001", ".254", ".109", ".078")

)


# Use kable() to format output
tab_01 %>%
  kable(
    format = "html",
    col.names = c("Scale", "High BAS group", "Moderate BAS group", "*p*"),
    caption = "**Table 1** <br /> *Means and Standard Deviations of Scores on Baseline Measures*"
) %>%
  kable_styling()
```
```

Table 1

Means and Standard Deviations of Scores on Baseline Measures

Scale	High BAS group	Moderate BAS group	*p*
BAS-T	46.17 (2.87)	37.99 (1.32)	<.001
SR	17.94 (1.88)	11.52 (1.84)	<.001
BDI	7.11 (6.50)	6.18 (6.09)	.254
ASRM	6.46 (4.01)	5.63 (3.69)	.109
M-SRM	11.05 (3.36)	11.76 (2.75)	.078

The table output is now spread out and is looking pretty good. It should also be easier to see the column alignment.

Using the Classic Theme

APA wants horizontal lines above and below the column names and also below the bottom row. Rather than using the `kable_styling()` function, we can use the `kable_classic()` function. This will add these lines. Note that the outputted table spans across the entire width of the HTML document.

```
# Use kable() to format output
tab_01 %>%
  kable(
    format = "html",
    col.names = c("Scale", "High BAS group", "Moderate BAS group", "*p*"),
    caption = "**Table 1** <br /> *Means and Standard Deviations of Scores on Baseline Measures*"
) %>%
  kable_classic()
```

Table 1

Means and Standard Deviations of Scores on Baseline Measures

Scale	High BAS group	Moderate BAS group	*p*
BAS-T	46.17 (2.87)	37.99 (1.32)	<.001
SR	17.94 (1.88)	11.52 (1.84)	<.001
BDI	7.11 (6.50)	6.18 (6.09)	.254
ASRM	6.46 (4.01)	5.63 (3.69)	.109
M-SRM	11.05 (3.36)	11.76 (2.75)	.078

Changing the Table Width

To adjust the table width we can use the `table.attr=` argument in the `kable()` function. This argument essentially adds elements into the `<table>` tag in the HTML output. We will add the `style=` HTML attribute, which allows us to include CSS syntax (which formats the HTML) in an HTML tag. Here the CSS syntax, `width:40%;` sets the table width to 40% of the document width. Note that the CSS syntax needs to be quoted. This is why we have two sets of quotes in the `table.attr=` argument.

```
# Use kable() to format output
tab_01 %>%
  kable(
    format = "html",
    col.names = c("Scale", "High BAS group", "Moderate BAS group", "*p*"),
    caption = "**Table 1** <br /> *Means and Standard Deviations of Scores on Baseline Measures*",
    table.attr = "style='width:40%;'"
) %>%
  kable_classic()
```

Table 1

Means and Standard Deviations of Scores on Baseline Measures

Scale	High BAS group	Moderate BAS group	*p*
BAS-T	46.17 (2.87)	37.99 (1.32)	<.001
SR	17.94 (1.88)	11.52 (1.84)	<.001
BDI	7.11 (6.50)	6.18 (6.09)	.254
ASRM	6.46 (4.01)	5.63 (3.69)	.109
M-SRM	11.05 (3.36)	11.76 (2.75)	.078

Matching the Document Font

The `kable_classic()` function uses the **Arial Narrow** font by default. It is good typesetting to use the same font in your table as in the rest of your document. The default font used for the text and headings of a knitted HTML document is **Helvetica Neue**. To change the table font we can use the `html_font=` argument in the `kable_classic()` function. This will use CSS to change the font. Because the font name has two words (i.e., a space) in its name, the font name needs to be quoted in the CSS. (Again, two sets of quotes, one to designate the R argument and another to designate the font name.) To ensure that this inner set of quotation marks is interpreted correctly in the CSS, we need to escape them using a backslash (\).

```
# Use kable() to format output
tab_01 %>%
  kable(
    format = "html",
    col.names = c("Scale", "High BAS group", "Moderate BAS group", "*p*"),
    caption = "**Table 1** <br /> *Means and Standard Deviations of Scores on Baseline Measures*",
    table.attr = "style='width:40%;'"
) %>%
  kable_classic(html_font = "\"Helvetica Neue\"")
```

Table 1

Means and Standard Deviations of Scores on Baseline Measures

Scale	High BAS group	Moderate BAS group	*p*
BAS-T	46.17 (2.87)	37.99 (1.32)	<.001
SR	17.94 (1.88)	11.52 (1.84)	<.001
BDI	7.11 (6.50)	6.18 (6.09)	.254
ASRM	6.46 (4.01)	5.63 (3.69)	.109
M-SRM	11.05 (3.36)	11.76 (2.75)	.078

Centering the Column Names

If you really want to be APA compliant, we also need to center all the column names in the header row (but not necessarily the cell information in the column). To do this we will pipe the output of our current table into the `row_spec()` function. This function allows us to format individual rows in the table. To specify the header row we use the argument `row=0` and also include `align="c"` to center the contents of the header row.

```
# Use kable() to format output
tab_01 %>%
  kable(
    format = "html",
    col.names = c("Scale", "High BAS group", "Moderate BAS group", "*p*"),
    caption = "**Table 1** <br /> *Means and Standard Deviations of Scores on Baseline Measures*",
    table.attr = "style='width:40%;'"
) %>%
  kable_classic(html_font = "\"Helvetica Neue\"") %>%
  row_spec(row = 0, align = "c")
```

Table 1

Means and Standard Deviations of Scores on Baseline Measures

Scale	High BAS group	Moderate BAS group	*p*
BAS-T	46.17 (2.87)	37.99 (1.32)	<.001
SR	17.94 (1.88)	11.52 (1.84)	<.001
BDI	7.11 (6.50)	6.18 (6.09)	.254
ASRM	6.46 (4.01)	5.63 (3.69)	.109
M-SRM	11.05 (3.36)	11.76 (2.75)	.078

Adding a Footnote

To add a footnote, we will pipe the output of our current table into the `footnote()` function. There are many arguments we can use to adapt the footnote. Here we will use `general=` to include the text of our footnote, `general_title=` to change the footnote title to *Note.*, and `footnote_as_chunk=TRUE` to display the footnote as a chunk of text rather than as a list

```
# Use kable() to format output
tab_01 %>%
  kable(
    format = "html",
    col.names = c("Scale", "High BAS group", "Moderate BAS group", "*p*"),
    caption = "**Table 1** <br /> *Means and Standard Deviations of Scores on Baseline Measures*",
    table.attr = "style='width:40%;'"
) %>%
  kable_classic(html_font = "\"Helvetica Neue\"") %>%
  row_spec(row = 0, align = "c") %>%
  footnote(
    general = "Standard deviations are presented in parentheses. BAS = Behavioral Activation System; BAS-T = Behavioral
               Activation System-Total scores from the Behavioral Inhibition System/Behavioral Activation System Scales;
               SR = Sensitivity to Reward scores from the Sensitivity to Punishment and Sensitivity to Reward
               Questionnaire; BDI = Beck Depression Inventory scores; ASRM = Altman Self-Rating Mania Scale scores; M-SRM
               = Modified Social Rhythm Metric Regularity scores.",
    general_title = "Note.",
    footnote_as_chunk = TRUE
    )
```

Table 1

Means and Standard Deviations of Scores on Baseline Measures

Scale	High BAS group	Moderate BAS group	*p*
BAS-T	46.17 (2.87)	37.99 (1.32)	<.001
SR	17.94 (1.88)	11.52 (1.84)	<.001
BDI	7.11 (6.50)	6.18 (6.09)	.254
ASRM	6.46 (4.01)	5.63 (3.69)	.109
M-SRM	11.05 (3.36)	11.76 (2.75)	.078

Note. Standard deviations are presented in parentheses. BAS = Behavioral Activation System; BAS-T = Behavioral Activation System-Total scores from the Behavioral Inhibition System/Behavioral Activation System Scales; SR = Sensitivity to Reward scores from the Sensitivity to Punishment and Sensitivity to Reward Questionnaire; BDI = Beck Depression Inventory scores; ASRM = Altman Self-Rating Mania Scale scores; M-SRM = Modified Social Rhythm Metric Regularity scores.

Final Remarks and Learning More

Some Remarks

When we initially created the data frame for the table contents, we used character strings to enter the data. This is necessary for columns that are actual strings (e.g., the Scale column), but is not the best choice for columns that are inherently numeric (e.g., means, standard deviations, and *p*-values we are providing in the remaining columns).

```
tab_01 = data.frame(
  scale = c("BAS-T", "SR", "BDI", "ASRM", "M-SRM"),
  high = c("46.17 (2.87)", "17.94 (1.88)",
           "7.11 (6.50)", "6.46 (4.01)", "11.05 (3.36)"),
  mod = c("37.99 (1.32)", "11.52 (1.84)",
          "6.18 (6.09)", "5.63 (3.69)", "11.76 (2.75)"),
  p = c("<.001", "<.001", ".254", ".109", ".078")

)
```

Using strings often makes more work for you. For example, if the data changes, you would need to recalculate the means/SDs and then update the contents of the strings. It also opens up the probability of having an error in your document; every time you transfer the results of a computation to text, there is a chance that you will make an error (e.g., typo, transposition of digits).

A more computationally-thoughtful alternative (if you had the raw data) would be to use R functions to compute the means/SDs directly in the data frame. We will demonstrate this with examples throughout the course.

As you begin your journey in using R Markdown, don't worry about this immediately. If it is easier to initially use strings to include the content, use strings. As you get more comfortable, you can try making numerical content more computational in nature.

Learning More

There are many resources to help you with creating tables. Here are a couple:

- The kableExtra package documentation is incredible. Among other things, it includes extensive vignettes on creating tables for HTML documents and for LaTeX documents. For example, you can learn how to include images and plots into your table, color cells, add spanned headers, add tooltips (e.g. popover message, hover highlighting), scrollboxes, and more!

- The `{kable}` and `{kableExtra}` packages are not the only show in town for creating tables (see here). I introduced these not because they are better than others, but because they are the packages I use and am most familiar with.