

# Manipulating data in the **tidyverse**

EC 103–02

Marcio Santetti

# RStudio cloud

# RStudio cloud

For students struggling with installing RStudio, *RStudio cloud* may be an alternative:

<https://rstudio.cloud/>

Set up an account and you can start working with RStudio from your browser.

# Some important **tidyverse** functions

# Some important **tidyverse** functions

Data wrangling and manipulation are common tasks when doing empirical work.

Even though it has been easier and easier to access high-quality data, we may need to perform some *cleaning*, *filtering*, and *organizing* before we proceed.

The **tidyverse** has a wide array of functions, of which we will study a few.

# Some important **tidyverse** functions

To see these functions in practice, let us keep working on the “toy” data set we saw in the last session.

Recall:

```
1 library(tidyverse)
2
3 my_data <- read_csv("toy_data.csv")
```

# Some important tidyverse functions

```
1 my_data
```

```
# A tibble: 9 × 5
  name      tip  age weight movie
<chr> <dbl> <dbl> <dbl> <chr>
1 John    1.5    17    150 drama
2 Anna    2.5    17    160 comedy
3 Xavier  3.5    21    165 drama
4 Walter  4      25    140 horror
5 Bailey  5      21    170 horror
6 David  10     28    175 drama
7 Anna   18     18    160 comedy
8 Anna    3      19    160 drama
9 Walter  1.25   24    140 comedy
```

# The pipe operator



# The pipe operator

The tidyverse has a very useful operator, known as the **pipe** operator, that facilitates data wrangling.

- `%>%`

If you are using RStudio, you can use the pipe operator using the following keyboard shortcut:

- `Cmd+Shift+M` (macOS);
- `Ctrl+Shift+M` (Windows).

# The `select()` function

# The `select()` function

The first function we will look at is the `select()` function.

In practice:

```
1 my_data %>%  
2   select(name, movie)
```

```
# A tibble: 9 × 2  
  name    movie  
  <chr>  <chr>  
1 John   drama  
2 Anna   comedy  
3 Xavier drama  
4 Walter horror  
5 Bailey horror  
6 David  drama  
7 Anna   comedy  
8 Anna   drama  
9 Walter comedy
```

# The `select()` function

```
1 my_data %>%  
2   select(name, age, weight)
```

```
# A tibble: 9 × 3  
  name      age weight  
  <chr>  <dbl>  <dbl>  
1 John      17     150  
2 Anna      17     160  
3 Xavier    21     165  
4 Walter    25     140  
5 Bailey    21     170  
6 David     28     175  
7 Anna      18     160  
8 Anna      19     160  
9 Walter    24     140
```

# The `select()` function

When working with a pipeline, R will not automatically update your data set.

In case you want to **store** a modified data set, you simply assign your pipeline to a **new object**.

```
1 my_data_subset <- my_data %>%  
2   select(name, age, weight)
```

# The `filter()` function

# The `filter()` function

In practice:

```
1 my_data %>%  
2   filter(movie %in% "drama")
```

```
# A tibble: 4 × 5  
  name      tip  age weight movie  
  <chr>  <dbl> <dbl>  <dbl> <chr>  
1 John     1.5    17    150 drama  
2 Xavier   3.5    21    165 drama  
3 David   10     28    175 drama  
4 Anna     3     19    160 drama
```

# The `filter()` function

In practice:

```
1 my_data %>%  
2   filter(age > 20)
```

```
# A tibble: 5 × 5
```

	name	tip	age	weight	movie
	<chr>	<dbl>	<dbl>	<dbl>	<chr>
1	Xavier	3.5	21	165	drama
2	Walter	4	25	140	horror
3	Bailey	5	21	170	horror
4	David	10	28	175	drama
5	Walter	1.25	24	140	comedy



# The `filter()` function

In practice:

```
1 my_data %>%  
2   filter(tip < 5)
```

```
# A tibble: 6 × 5
```

	name	tip	age	weight	movie
	<chr>	<dbl>	<dbl>	<dbl>	<chr>
1	John	1.5	17	150	drama
2	Anna	2.5	17	160	comedy
3	Xavier	3.5	21	165	drama
4	Walter	4	25	140	horror
5	Anna	3	19	160	drama
6	Walter	1.25	24	140	comedy

# The `filter()` function

In practice:

```
1 my_data %>%  
2   filter(name %in% "Anna" | name %in% "David") # or
```

```
# A tibble: 4 × 5  
  name    tip  age weight movie  
  <chr> <dbl> <dbl> <dbl> <chr>  
1 Anna   2.5   17   160 comedy  
2 David  10    28   175 drama  
3 Anna   18    18   160 comedy  
4 Anna   3     19   160 drama
```

# The `filter()` function

In practice:

```
1 my_data %>%  
2   filter(name %in% "Anna" & tip > 5) # and
```

# A tibble: 1 × 5

	name	tip	age	weight	movie
	<chr>	<dbl>	<dbl>	<dbl>	<chr>
1	Anna	18	18	160	comedy

# The `filter()` function

In case you want to store a modified data set, just assign to a new object:

```
1 my_data_filter <- my_data %>%  
2   filter(name %in% "Anna" & tip > 5)
```

# The mutate() function

# The mutate() function

In practice:

```
1 my_data %>%  
2   mutate(age_months = age * 12)
```

```
# A tibble: 9 × 6
```

	name	tip	age	weight	movie	age_months
	<chr>	<dbl>	<dbl>	<dbl>	<chr>	<dbl>
1	John	1.5	17	150	drama	204
2	Anna	2.5	17	160	comedy	204
3	Xavier	3.5	21	165	drama	252
4	Walter	4	25	140	horror	300
5	Bailey	5	21	170	horror	252
6	David	10	28	175	drama	336
7	Anna	18	18	160	comedy	216
8	Anna	3	19	160	drama	228
9	Walter	1.25	24	140	comedy	288

# The mutate() function

In practice:

```
1 my_data %>%  
2   mutate(weight_kg = weight * 0.453592)
```

```
# A tibble: 9 × 6  
  name      tip  age weight movie  weight_kg  
  <chr> <dbl> <dbl> <dbl> <chr>    <dbl>  
1 John    1.5    17    150 drama    68.0  
2 Anna    2.5    17    160 comedy   72.6  
3 Xavier  3.5    21    165 drama    74.8  
4 Walter  4      25    140 horror   63.5  
5 Bailey  5      21    170 horror   77.1  
6 David  10     28    175 drama    79.4  
7 Anna   18     18    160 comedy   72.6  
8 Anna    3     19    160 drama    72.6  
9 Walter  1.25   24    140 comedy   63.5
```

# The mutate() function

A way of mutating, but adding new columns:

```
1 fav_birds <- c("kestrel", "quail", "albatross", "hummingbird",
2               "american robin", "eastern bluebird", "hummingbird",
3               "california quail", "blue jay")
4
5 my_data %>%
6   add_column(fav_birds)
```

```
# A tibble: 9 × 6
```

	name	tip	age	weight	movie	fav_birds
	<chr>	<dbl>	<dbl>	<dbl>	<chr>	<chr>
1	John	1.5	17	150	drama	kestrel
2	Anna	2.5	17	160	comedy	quail
3	Xavier	3.5	21	165	drama	albatross
4	Walter	4	25	140	horror	hummingbird
5	Bailey	5	21	170	horror	american robin
6	David	10	28	175	drama	eastern bluebird
7	Anna	18	18	160	comedy	hummingbird
8	Anna	3	19	160	drama	california quail
9	Walter	1.25	24	140	comedy	blue jay



# A pipeline

```
1 my_data %>%  
2   add_column(fav_birds) %>%  
3   filter(fav_birds %in% "hummingbird") %>%  
4   select(name)
```

```
# A tibble: 2 × 1  
  name  
  <chr>  
1 Walter  
2 Anna
```

# The `group_by()` function

# The `group_by()` function

```
1 my_data %>%  
2   group_by(name) %>%  
3   summarize(mean_tip = mean(tip)) # compute average tip ($) given
```

```
# A tibble: 6 × 2  
  name      mean_tip  
  <chr>    <dbl>  
1 Anna      7.83  
2 Bailey     5  
3 David     10  
4 John      1.5  
5 Walter    2.62  
6 Xavier    3.5
```