

1: Manipulating Data with dplyr

RStudio interface showing a lesson on manipulating data with dplyr. The console displays a progress bar and a quiz question. The Environment pane shows variables like cran, cran2, and cran3. The Help pane shows the documentation for Relational Operators.

```
11/9 Df[,m]=tmp <- (X+Z); log2(tmp+5)} #Crea columna m con los valores de log base 2
11/9 Df[,m]=tmp <- (X+Z); log2(tmp+5)} #Crea columna m con los valores de log base 2
12011 (Top Level)
```

Console output:

```
1 844086.5

| That's the answer I was looking for.

===== 95%

| That's not particularly interesting. summarize() is most useful when working with data that has been
| grouped by the values of a particular variable.

...

===== 97%

| We'll look at grouped data in the next lesson, but the idea is that summarize() can give you the
| requested value FOR EACH group in your dataset.

...

===== 98%

| In this lesson, you learned how to manipulate data using dplyr's five main functions. In the next
| lesson, we'll look at how to take advantage of some other useful features of dplyr to make your life
| as a data analyst much easier.

...

===== 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: Yes
2: No

Selection:
```

Environment pane:

Variable	Value
cran	225468 obs. of 11 variables
cran2	225468 obs. of 8 variables
cran3	225468 obs. of 3 variables

Help pane: Relational Operators

Description

Binary operators which allow the comparison of values in atomic vectors.

Usage

```
x < y
x > y
x <= y
x >= y
x == y
x != y
```

2: Grouping and Chaining with dplyr

RStudio interface showing a lesson on grouping and chaining with dplyr. The console displays a progress bar and a quiz question. The Environment pane shows variables like result1, result2, result3, setup, setup_ss, top_countries, and top_counts. The Help pane shows the documentation for group_by().

```
1 # arrange() the result by size_mb, in descending order.
1027 (Top Level)
```

Console output:

```
8 13 DE ipred 186685 0.178036690
9 14 US mnormt 36204 0.034528825
10 16 US iterators 289972 0.276538849
# ... with 142,011 more rows

| Excellent work!

===== 96%

| And finish it off.

> submit()

| Sourcing your script...

| That's the answer I was looking for.

===== 98%

| In this lesson, you learned about grouping and chaining using dplyr. You combined some of the things
| you learned in the previous lesson with these more advanced ideas to produce concise, readable, and
| highly effective code. Welcome to the wonderful world of dplyr!

...

===== 100%

| Would you like to receive credit for completing this course on Coursera.org?

1: Yes
2: No

Selection:
```

Environment pane:

Variable	Value
result1	46 obs. of 5 variables
result2	46 obs. of 5 variables
result3	46 obs. of 5 variables
setup	132 obs. of 6 variables
setup_ss	25 obs. of 5 variables
top_countries	46 obs. of 5 variables
top_counts	61 obs. of 5 variables

Help pane: group_by()

Arguments

.data a tibble

... variables to group by. All tibbles accept variable names, some will also accept functions of variables. Duplicated groups will be silently dropped.

add By default, when add = FALSE, group_by will override existing groups. To instead add to the existing groups, use add = TRUE

.dots Used to work around non-standard evaluation. See vignette("nse") for details.

Tibble types

3: Tidying Data with tidyr

RStudio

File Edit Code View Plots Session Build Debug Tools Help

Go to file/function Addins

script1.R script2.R script3.R script4.R script5.R script6.R script7.R script8.R script9.R

Run Source

15:10 (Top Level)

R Script

Console

```

> source("local_data_frame_130_x_bj")
Groups: part, sex [6]

  score_range part sex count total prop
<chr> <chr> <chr> <int> <int> <dbl>
1 700-800 read male 40151 776092 0.05173485
2 600-690 read male 121950 776092 0.15713343
3 500-590 read male 227141 776092 0.29267278
4 400-490 read male 242554 776092 0.31253253
5 300-390 read male 113568 776092 0.14633317
6 200-290 read male 30728 776092 0.03959324
7 700-800 read fem 38898 883955 0.04400450
8 600-690 read fem 126084 883955 0.14263622
9 500-590 read fem 259553 883955 0.29362694
10 400-490 read fem 296793 883955 0.33575578
# ... with 26 more rows

| Great job!

| ----- | 98%

| In this lesson, you learned how to tidy data with tidyr and dplyr. These tools will help you spend
| less time and energy getting your data ready to analyze and more time actually analyzing it.
|
| ----- | 100%

| Would you like to receive credit for completing this course on Coursera.org?
1: No
2: Yes
Selection:

```

Environment History

Global Environment

- failed 6 obs. of 4 variables
- gradebook 10 obs. of 4 variables
- mydf 225468 obs. of 11 variables
- pack_sum 6023 obs. of 5 variables
- passed 4 obs. of 4 variables
- res 20 obs. of 3 variables
- result1 46 obs. of 5 variables
- result2 46 obs. of 5 variables

Files Plots Packages Help Viewer

R: Spread a key-value pair across multiple columns. Find in Topic

spread (tidyr)

R Documentation

Spread a key-value pair across multiple columns.

Description

Spread a key-value pair across multiple columns.

Usage

```
spread(data, key, value, fill = NA, convert = FALSE, sep = NULL)
```

Arguments

4: Dates and Times with lubridate

RStudio

File Edit Code View Plots Session Build Debug Tools Help

Go to file/function Addins

script1.R script2.R script3.R script4.R script5.R script6.R script7.R script8.R script9.R

Run Source

15:10 (Top Level)

R Script

Console

```

> source("local_data_frame_130_x_bj")
Groups: part, sex [6]

| You nailed it! Good job!

| ----- | 95%

| This is where things get a little tricky. Because of things like leap years, leap seconds, and
| daylight savings time, the length of any given minute, day, month, week, or year is relative to when
| it occurs. In contrast, the length of a second is always the same, regardless of when it occurs.
|
| ----- | 97%

| To address these complexities, the authors of lubridate introduce four classes of time related
| objects: instants, intervals, durations, and periods. These topics are beyond the scope of this
| lesson, but you can find a complete discussion in the 2011 Journal of Statistical Software paper
| titled "Dates and Times Made Easy with lubridate".
|
| ----- | 98%

| This concludes our introduction to working with dates and times in lubridate. I created a little timer
| that started running in the background when you began this lesson. Type stopwatch() to see how long
| you've been working!

> stopwatch()
[1] "19M 58.86352109909065"

| you are amazing!

| ----- | 100%

| Would you like to receive credit for completing this course on Coursera.org?
1: No
2: Yes
Selection:

```

Environment History

Global Environment

- top_unique_... 60 obs. of 5 variables

Values

Variable	Value
arrive	2016-11-12 22:24:56
depart	2016-11-11 17:34:56
dt1	"2014-08-23 17:23:02"
dt2	chr [1:3] "2014-03-14" "2014-09-22"
how_long	Formal class Interval
last_time	

Files Plots Packages Help Viewer

R: Utilities for creation and manipulation of 'Interval' ... Find in Topic

interval (lubridate)

R Documentation

Utilities for creation and manipulation of Interval objects.

Description

interval creates an [Interval-class](#) object with the specified start and end dates. If the start date occurs before the end date, the interval will be positive. Otherwise, it will be negative.

%--% Creates an interval that covers the range spanned by two dates. It replaces the original behavior of lubridate, which created an interval by default whenever two date-times were subtracted.

int_start and int_end are accessors the start date of an interval. Note that changing the start date of an interval will change the