nbs02-03c_teacher-notes

AUTHOR
Jeremy Mikecz

Introduction to Tidyverse

Tidyverse is a system of R packages for data wrangling and analysis. It provides a different method and syntax for working with data tables (often called "data frames" in data science; but, tidyverse dataframes are known as "tibbles") from base R.

For additional help learning how to use R's tidyverse system of packages, see notebook 02 or slideshow.

Types of 2d data in R

- **dataframes (core R)**: columns with column names; rows with indexes
- **tibbles (tidyverse)**: a type of dataframe used with the tidyverse collection of packages
- **data.table**: optimized for speed with large datasets

In these lessons, we will work with tidyverse a collection of packages designed for data science. Tidyverse works with **tibbles**, a customized and newer form of dataframes.

[PRES: pipes, tidyverse syntax, tidy data, tribbles]

1. Setup

```
## install tidyverse with:
#install.packages("tidyverse")

## update tidyverse with:
#tidyverse_update()

## import tidyverse with:
library(tidyverse)
```

```
Warning: package 'dplyr' was built under R version 4.4.2

── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
✓ dplyr     1.1.4     ✓ readr     2.1.5
✓ forcats   1.0.0     ✓ stringr   1.5.1
✓ ggplot2   3.5.1     ✓ tibble    3.2.1
✓ lubridate 1.9.3     ✓ tidyr     1.3.1
```

```
✓ purrr     1.0.2
── Conflicts ─────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
## see what packages are included with tidyverse:
tidyverse_packages()
```

```
 [1] "broom"         "conflicted"    "cli"           "dbplyr"
 [5] "dplyr"         "dtplyr"        "forcats"       "ggplot2"
 [9] "googledrive"   "googlesheets4" "haven"         "hms"
[13] "httr"          "jsonlite"      "lubridate"     "magrittr"
[17] "modelr"        "pillar"        "purrr"         "ragg"
[21] "readr"         "readxl"        "reprex"        "rlang"
[25] "rstudioapi"    "rvest"         "stringr"       "tibble"
[29] "tidyr"         "xml2"          "tidyverse"
```

3b. preloaded datasets

We have already used the preloaded `starwars` dataset. But is it a tibble, a dataframe, or something else?

3c. Dataframes vs. tibbles

```
wfz df <- read.csv("../data/census1970.csv")}
```

Convert a dataframe into a tibble

[PRES: tidyverse verbs]

4b. ARRANGE: Sorting

4c. FILTER (rows)

```
wfz starwars |>   filter(species=="Droid")}
```

4d. SELECT: Subset (columns)

For a range of columns

4e. RENAME columns

```
wfz starwars |>   rename(character = name, planet = homeworld)}
```

4f. MUTATE: create new columns

```
wfz starwars |>     mutate(taller_than122cm = ifelse(height > 122, TRUE, FALSE))}
```

```
wfz starwars |>     mutate(in_new_hope = map_lgl(films, ~"A New Hope" %in% .x)) |>
filter(in_new_hope == TRUE)}
```

Exercise

Using what you learned above:

1. Subset the Star Wars dataset, keeping only characters' name, homeworld, and species.
2. Sort the dataset by species.
3. Then, create a new column that identifies whether the character is an organic lifeform or not.

5. Split-Apply-Combine

A very common data science technique is to split a dataset into groups, perform some action on each of those groups, and then combine the results from each group into one new dataset. For example, we can use this technique to calculate the average height of Star Wars characters by species using the **group_by()** and **summarise()** functions.

```
wfz starwars |>    group_by(species) |>   summarise(avg_height = mean(height, na.rm = TRUE))}
```

```
wfz starwars |>       group_by(species) |>      summarise(avg_height = mean(height, na.rm = TRUE))
|>     arrange(avg_height)}
```

Exercise: Split-Apply-Combine

Using the split-apply-combine technique and other methods learned above, do at least one of the following:

1. Identify the most common species for each planet.
2. Identify the most common eye color for each species.
3. Identify the average and maximum age (so two columns) for each species.
4. answer a similar question of your own