Sidekicks of the Tidyverse

Andrew B. Collier
&
Bianca Peterson

22 April 2023
satRday
King's College London

Fathom

I have a confession.

I really like Python.

I  R more.

♥ ⩾ really like.


dplyr
forcats
ggplot2
purrr
readr
stringr
TIBBLE
tidyr

Heroes

of the Tidyverse

`dplyr::mutate()` `dplyr::select()`

`dplyr::filter()` `dplyr::group_by()`

`dplyr::summarise()` `purrr::map()`

`tidyr::pivot_longer()` `tidyr::nest()`

`tidyr::pivot_wider()` `tidyr::unnest()`

The titans! Doing all the heavy lifting.

Villains

of the Tidyverse

Sidekicks

of the Tidyverse

Not as flashy or glamorous as the heroes.

Dashing Duo #1

`map()`

`imap()`

```
map(.x, .f, ...)
```

where

- `.x` — a list or vector; and
- `.f` — a function of one variable.

```
imap(.x, .f, ...)
```

where

- `.x` — a list or vector; and
- `.f` — a function of *two* variables.

```
height
```

```
Luke Skywalker          C-3PO          R2-D2
           172            167             96
```

```
# Iterate vector; return list.
#
map(height, function(h) {h})
```

```
$`Luke Skywalker`
[1] 172

$`C-3PO`
[1] 167

$`R2-D2`
[1] 96
```

```
imap(height, function(h, i) {
  paste(i, h)
})
```

```
$`Luke Skywalker`
[1] "Luke Skywalker 172"

$`C-3PO`
[1] "C-3PO 167"

$`R2-D2`
[1] "R2-D2 96"
```

What to do when "data" is in the list index?

```
head(people, n = 4)
```

```
$`Luke Skywalker`
  id height mass gender
1  1    172   77   male

$`C-3PO`
  id height mass gender
1  2    167   75    n/a

$`R2-D2`
  id height mass gender
1  3     96   32    n/a

$`Darth Vader`
  id height mass gender
1  4    202  136   male
```

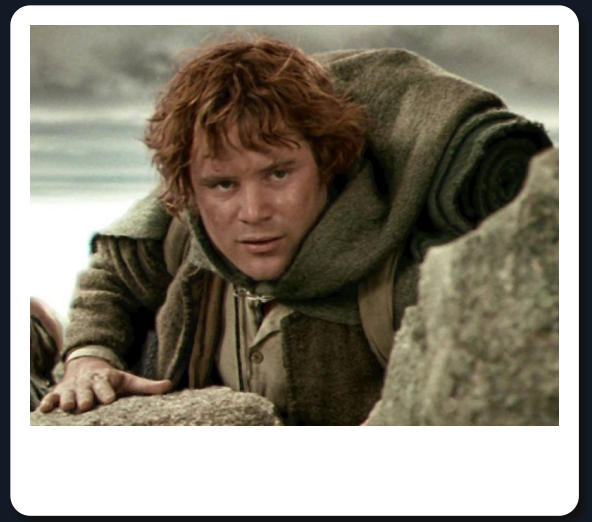
```r
people <- imap(people, function(data, index) {
  data %>% mutate(name = index)
}) %>%
  bind_rows() %>% # Avoid use of imap_dfr() — it's deprecated.
  head(4)

people
```

```
  id height mass gender            name
1  1    172   77   male Luke Skywalker
2  2    167   75    n/a           C-3PO
3  3     96   32    n/a           R2-D2
4  4    202  136   male    Darth Vader
```

The columns are in the "wrong" order!

Dashing Duo #2

`select()`



`relocate()`

```
people
```

```
  id height mass gender           name
1  1    172   77   male Luke Skywalker
2  2    167   75    n/a          C-3PO
3  3     96   32    n/a          R2-D2
4  4    202  136   male    Darth Vader
```

```
# Shuffle order and drop gender.
people %>% select(id, name, height, mass)
```

```
  id           name height mass
1  1 Luke Skywalker    172   77
2  2          C-3PO    167   75
3  3          R2-D2     96   32
4  4    Darth Vader    202  136
```

```r
people %>%
  # Moving to beginning is easy.
  select(name, everything())
```

```
             name id height mass gender
1 Luke Skywalker  1    172   77   male
2           C-3PO  2    167   75    n/a
3           R2-D2  3     96   32    n/a
4     Darth Vader  4    202  136   male
```

```r
people %>%
  # Moving to end requires fancy footwork.
  select(-id, everything(), id)
```

```
  height mass gender              name id
1    172   77   male Luke Skywalker  1
2    167   75    n/a           C-3PO  2
3     96   32    n/a           R2-D2  3
4    202  136   male     Darth Vader  4
```

What about a position that's not at beginning or end?

```r
people %>%
  # Must specify all columns in required order...
  select(id, height, name, mass, gender)
```

```
  id height           name mass gender
1  1    172 Luke Skywalker   77   male
2  2    167          C-3PO   75    n/a
3  3     96          R2-D2   32    n/a
4  4    202    Darth Vader  136   male
```

```
people %>%
  # ... or at least everything up to the column you're moving.
  select(id:height, name, everything())
```

```
  id height             name mass gender
1  1    172 Luke Skywalker   77   male
2  2    167            C-3PO   75    n/a
3  3     96            R2-D2   32    n/a
4  4    202      Darth Vader  136   male
```

Surely there must be a better way?

```
people %>%
  relocate(name, .after = id)
```

```
  id              name height mass gender
1  1 Luke Skywalker    172   77   male
2  2           C-3PO    167   75    n/a
3  3           R2-D2     96   32    n/a
4  4     Darth Vader    202  136   male
```

```
people %>%
  relocate(name, .before = height)
```

```r
# Moving column to end.
people %>% relocate(id, .after = last_col())
```

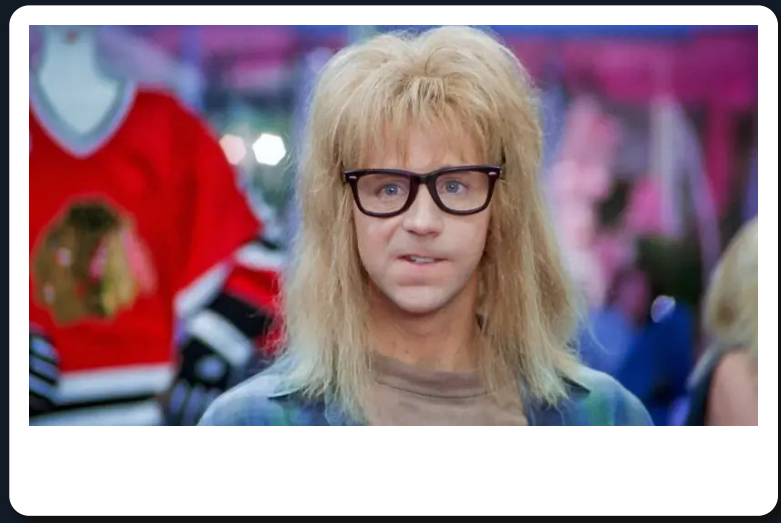
```
  height mass gender            name id
1    172   77   male Luke Skywalker  1
2    167   75    n/a           C-3PO  2
3     96   32    n/a           R2-D2  3
4    202  136   male    Darth Vader  4
```

```r
# Moving range of columns.
people %>% relocate(height:gender, .after = last_col())
```

```
  id            name height mass gender
1  1 Luke Skywalker    172   77   male
2  2           C-3PO    167   75    n/a
3  3           R2-D2     96   32    n/a
4  4    Darth Vader    202  136   male
```

Dashing Duo #3

`case_when()`



`case_match()`

```
genders
```

```
             name gender
1 Luke Skywalker   male
2           C-3PO    n/a
3           R2-D2    n/a
4    Darth Vader   male
```

The "n/a" values should be replaced with NA.

```
genders %>% mutate(
  gender = ifelse(gender == "n/a", NA, gender)
)
```

```
             name gender
1 Luke Skywalker   male
2           C-3PO   <NA>
3           R2-D2   <NA>
4     Darth Vader   male
```

```
genders
```

```
             name gender
1 Luke Skywalker   male
2           C-3PO    n/a
3           R2-D2    n/a
4    Darth Vader      M
```

```r
genders %>% mutate(
  gender = ifelse(gender == "n/a", NA, gender),
  gender = ifelse(gender == "M", "male", gender)
)
```

```
             name gender
1 Luke Skywalker   male
2           C-3PO   <NA>
3           R2-D2   <NA>
4    Darth Vader   male
```

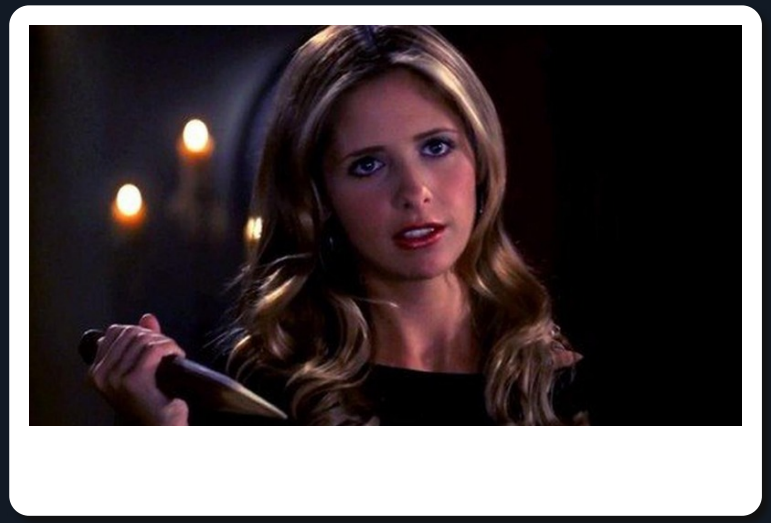
```
genders %>% mutate(
  gender = case_when(
    # Give target per pattern.
    gender == "n/a" ~ NA,
    gender == "M" ~ "male",
    .default = gender
  )
)
```

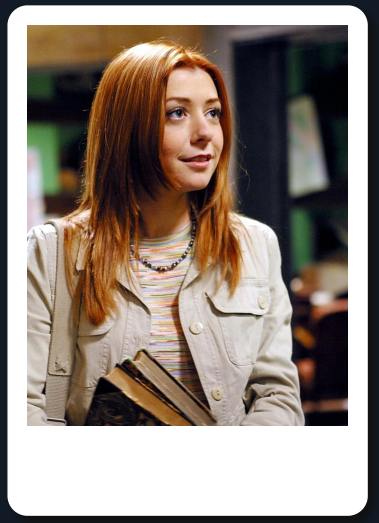
```
            name gender
1 Luke Skywalker   male
2          C-3PO   <NA>
3          R2-D2   <NA>
4    Darth Vader   male
```

```
genders %>% mutate(
  gender = case_match(
    gender,
    "n/a" ~ NA,
    c("M", "m") ~ "male",
    .default = gender
  )
)
```

```
            name gender
1 Luke Skywalker   male
2          C-3PO   <NA>
3          R2-D2   <NA>
4    Darth Vader   male
```

Dashing Duo #4





`str_flatten()`

`str_flatten_comma()`

```
names
```

```
[1] "Luke Skywalker" "C-3PO"          "R2-D2"          "Darth Vader"
```

```
paste(names)
```

```
[1] "Luke Skywalker" "C-3PO"          "R2-D2"          "Darth Vader"
```

Nope!

Why?

```
paste("Luke Skywalker", "C-3PO", "R2-D2", "Darth Vader")
```

```
[1] "Luke Skywalker C-3PO R2-D2 Darth Vader"
```

```
paste(names, collapse = " ")
```

```
[1] "Luke Skywalker C-3PO R2-D2 Darth Vader"
```

```
paste(names, collapse = ", ")
```

```
[1] "Luke Skywalker, C-3PO, R2-D2, Darth Vader"
```

Success!

```
str_flatten(names)
```

```
[1] "Luke SkywalkerC-3POR2-D2Darth Vader"
```

No `collapse` required.

```
str_flatten(names, collapse = ", ")
```

```
[1] "Luke Skywalker, C-3PO, R2-D2, Darth Vader"
```

```
str_flatten_comma(names)
```

```
[1] "Luke Skywalker, C-3PO, R2-D2, Darth Vader"
```

And it handles the Oxford comma.

```
str_flatten_comma(names, last = ", and ")
```

```
[1] "Luke Skywalker, C-3PO, R2-D2, and Darth Vader"
```

Dashing Duo #5

`safely()`

`insistently()`

```python
try:
  # Might generate an error.
except ValueError:
  # Do something.
except RuntimeError:
  # Do something else.
except:
  # Do something different.
else:
  # Do this if no error.
```

```r
tryCatch(
  {
    # Might generate an error.
  },
  error = function(error) {
    # Do something.
  },
  finally = {
    # Do this ALWAYS.
  }
)
```

```r
sheldon_is_hero <- function() {
  message("Is Sheldon the real hero?")
  stop("Sheldon is not the hero.")
}

sheldon_is_hero()
```

```
Error in sheldon_is_hero() : Amy is the hero.
In addition: Warning message:
In sheldon_is_hero() : Are you sure?
```

```r
amy_is_hero <- function() {
  TRUE
}

amy_is_hero()
```

```
[1] TRUE
```

Using `try()` at least stops a script from *breaking*.

```
hero <- try(sheldon_is_hero(), silent = TRUE)
```

No output. Look at the result.

```
hero
```

```
[1] "Error in sheldon_is_hero() : Amy is the hero.\n"
attr(,"class")
[1] "try-error"
attr(,"condition")
<simpleError in sheldon_is_hero(): Amy is the hero.>
```

But the result returned is not ideal.

```
sheldon_is_hero_safely <- safely(sheldon_is_hero)
```

```
hero <- sheldon_is_hero_safely()

hero
```

```
$result
NULL

$error
<simpleError in .f(...): Sheldon is not the hero.>
```

```
amy_is_hero_safely <- safely(amy_is_hero)
```

```
hero <- amy_is_hero_safely()

hero
```

```
$result
[1] TRUE

$error
NULL
```

What if we just want something simpler?

```r
sheldon_is_hero_possibly <- possibly(
  sheldon_is_hero,
  otherwise = FALSE
)
```

```r
hero <- sheldon_is_hero_possibly()

hero
```

```
[1] FALSE
```

What if we want to try again on failure?

```r
sheldon_is_hero_insistently <- insistently(
  sheldon_is_hero,
  rate = rate_backoff(max_times = 4, jitter = FALSE)
)

sheldon_is_hero_insistently()
```

```
# 2023-04-21 10:10:48
Is Sheldon the real hero?
# 2023-04-21 10:10:50 (2s delay)
Is Sheldon the real hero?
# 2023-04-21 10:10:54 (4s delay)
Is Sheldon the real hero?
# 2023-04-21 10:11:02 (8s delay)
Is Sheldon the real hero?
Error in `rate_sleep()`:
! Request failed after 4 attempts.
```

```
sheldon_is_hero_insistently_possibly <- possibly(
  insistently(
    sheldon_is_hero,
    rate = rate_backoff(pause_base = 1, max_times = 4, jitter = FALSE)
  ),
  otherwise = FALSE
)

sheldon_is_hero_insistently_possibly()
```

```
Is Sheldon the real hero?
Is Sheldon the real hero?
Is Sheldon the real hero?
Is Sheldon the real hero?
Is Sheldon the real hero?
[1] FALSE
```

No matter how much you might insist, Sheldon is not the hero.

They are all heroes.

Andrew B. Collier

andrew@fathomdata.dev

https://github.com/datawookie
https://twitter.com/datawookie
https://www.linkedin.com/in/datawookie/

Bianca Peterson

bianca@fathomdata.dev

https://github.com/binxiepeterson
https://twitter.com/BinxiePeterson
https://www.linkedin.com/in/bianca-peterson/

Fathom

www.fathomdata.dev

{emayili}

Slides at https://tinyurl.com/satrday-tidyverse-sidekicks.