Data wrangling

Tame your data

Daniela Palleschi

4/13/23

Table of contents

```
## play sound if error encountered
### from: https://sejohnston.com/2015/02/24/make-r-beep-when-r-markdown-finishes-or-when-i
options(error = function(){    # Beep on error
  beepr::beep(sound = "wilhelm")
  Sys.sleep(2) #
  }
 )
## and when knitting is complete
.Last <- function() {          # Beep on exiting session
  beepr::beep(sound = "ping")
  Sys.sleep(6) # allow to play for 6 seconds
  }
```

```
# Create references.json file based on the citations in this script
# make sure you have 'bibliography: references.json' in the YAML
rbbt::bbt_update_bib("_wrangling.qmd")
```

```
Wrote 2 references to './references/references.json'
```

```
knitr::opts_chunk$set(eval = T, # change this to 'eval = T' to reproduce the analyses; mak
                      echo = T, # 'print code chunk?'
                      message = F, # 'print messages (e.g., warnings)?'
                      error = F,
                      warning = F)
```

'wrangle' defined

/ raŋ l/

noun

a dispute or argument, typically one that is long and complicated. “an insurance wrangle is holding up compensation payments”

verb

1. have a long, complicated dispute or argument. “the bureaucrats continue wrangling over the fine print”
2. NORTH AMERICAN round up, herd, or take charge of (livestock). “the horses were wrangled early”

Wrangler

- Jeep Wrangler 

- Wrangler Jeans 

- Cowboys 

Data Wrangling

- data wrangling = tidying + transforming
- an often long, arduous stage of analysis

Tidy

- re-shaping
  - e.g., from wide to long data
- outcome:
  - each column = a variable
  - each row = an observation

Transform

- filtering
- creating new variables based on observations (e.g., reaction times)
- computing summary statistics (e.g., means)

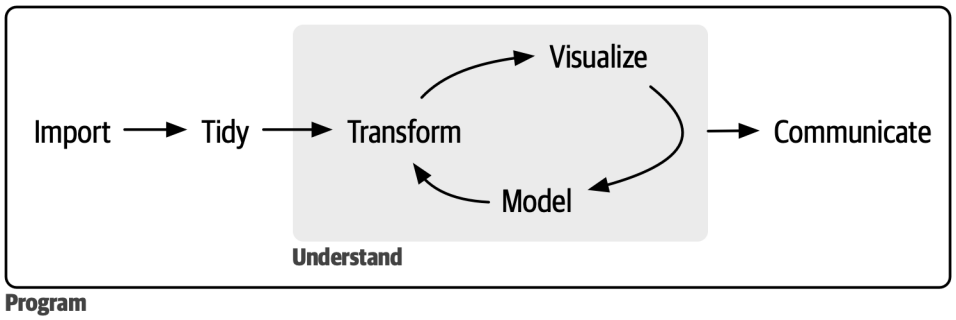

Figure 1: Image source: Wickham, Çetinkaya-Rundel, and Grolemund (n.d.) (all rights reserved)

Why tidy data?

- helps future you
  - and collaborators
- facilitates sharing your data *and* code (Laurinavichyute, Yadav, and Vasishth 2022)
- in short: facilitates reproducibility!

What does tidy data look like?

Three rules (Wickham, Çetinkaya-Rundel, and Grolemund, n.d.):

1. Each variable is a column, each column is a variable
2. Each observation is a row, each row is an observation
3. Each value is a cell, each cell is a single value

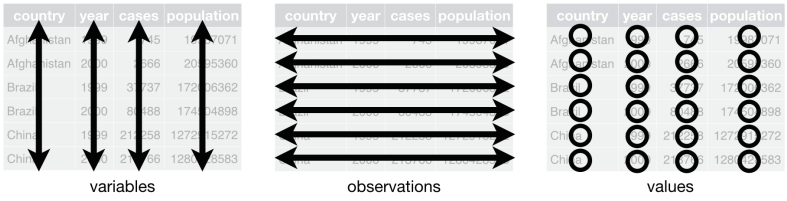

Figure 2: Image source: Wickham, Çetinkaya-Rundel, and Grolemund (n.d.) (all rights reserved)

- N.B., how you define a *variable* or *observation* is relative to what you want to do
  - for now, let's consider a single trial per participant as an observation

the `tidyverse`

- a collection of R packages for tidy data
- you need to load a package at the beginning of every session
- today we will mostly use functions from the `dplyr` package, but if you load the `tidyverse` you don't need to also load `dplyr`

```
# load tidyverse
library(tidyverse)
```

package versions

- you can check the package version with:

```
packageVersion("tidyverse")
```

```
[1] '2.0.0'
```

- need to update?

```
# update a single package
install.packages("tidyverse")
```

- what about your other packages?

```
# which packages need updating?
old.packages()
# update all old packages
update.packages()
```

the magitrr pipe %>%

- takes the object before it and feeds it into the next command
  - the pipe could be read as "and then"
  - N.B., there's a new pipe in town! The R native `|>` (Ctrl/Cmd+Shift+M)

```
# take data frame and then...
iris %>%
```

```
3    # print the head
4    head()
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

Figure 3: Image source: magittr documentation (all rights reserved)

load our data

```
# load lifetime data
readr::read_csv(here::here("data/data_lifetime_pilot.csv"))
```

```
# A tibble: 4,431 x 28
   RECORDING_SESSION_LABEL TRIAL_INDEX EYE_USED IA_DWELL_TIME
   <chr>                         <dbl> <chr>            <dbl>
 1 px7                               1 RIGHT                0
 2 px7                               2 RIGHT                0
 3 px7                               3 RIGHT                0
 4 px7                               3 RIGHT                0
 5 px7                               3 RIGHT                0
 6 px7                               3 RIGHT                0
 7 px7                               3 RIGHT                0
 8 px7                               3 RIGHT                0
 9 px7                               4 RIGHT                0
10 px7                               5 RIGHT                0
# i 4,421 more rows
# i 24 more variables: IA_FIRST_FIXATION_DURATION <dbl>,
#   IA_FIRST_RUN_DWELL_TIME <dbl>, IA_FIXATION_COUNT <dbl>, IA_ID <dbl>,
#   IA_LABEL <chr>, IA_REGRESSION_IN <dbl>, IA_REGRESSION_IN_COUNT <dbl>,
#   IA_REGRESSION_OUT <dbl>, IA_REGRESSION_OUT_COUNT <dbl>,
#   IA_REGRESSION_PATH_DURATION <dbl>, KeyPress <dbl>, rt <dbl>, bio <chr>,
#   critical <chr>, gender <chr>, item_id <dbl>, list <dbl>, match <chr>, ...
```

- was anything added to the Environment pane?

save dataframe as object

- object_name <- code_output_to_be_saved_as_object_name

```
# load lifetime data
df_lifetime <- readr::read_csv(here::here("data/data_lifetime_pilot.csv"),
                               locale = readr::locale(encoding = "latin1") # for special c
                               )
```

- you should now see the object `df_lifetime` in the Environment pane

A note on annotation

- annotation is continuous: provide useful comments to describe your code
- you always have at least one collaborator: future you!
  - try to write useful comments to help future you/collaborators follow

```
library(tidyverse) # for tidy data wrangling
```

Tidyverse verbs

- verbs are functions from the `tidyverse` package
- for data tidying and transforming we'll mostly use verbs from the `dplyr` package, which is part of the `tidyverse`

`rename()`

- let's start by re-naming some of our variables
  - e.g., `RECORDING_SESSION_LABEL` is a long way of saying 'participant'

```
# load tidyverse
df_lifetime %>%
  rename("px" = RECORDING_SESSION_LABEL)
```

```
# A tibble: 4,431 x 28
   px    TRIAL_INDEX EYE_USED IA_DWELL_TIME IA_FIRST_FIXATION_DURATION
   <chr>       <dbl> <chr>            <dbl>                       <dbl>
 1 px7             1 RIGHT                0                           0
 2 px7             2 RIGHT                0                           0
 3 px7             3 RIGHT                0                           0
 4 px7             3 RIGHT                0                           0
 5 px7             3 RIGHT                0                           0
 6 px7             3 RIGHT                0                           0
 7 px7             3 RIGHT                0                           0
 8 px7             3 RIGHT                0                           0
 9 px7             4 RIGHT                0                           0
10 px7             5 RIGHT                0                           0
# i 4,421 more rows
# i 23 more variables: IA_FIRST_RUN_DWELL_TIME <dbl>, IA_FIXATION_COUNT <dbl>,
```

```
#   IA_ID <dbl>, IA_LABEL <chr>, IA_REGRESSION_IN <dbl>,
#   IA_REGRESSION_IN_COUNT <dbl>, IA_REGRESSION_OUT <dbl>,
#   IA_REGRESSION_OUT_COUNT <dbl>, IA_REGRESSION_PATH_DURATION <dbl>,
#   KeyPress <dbl>, rt <dbl>, bio <chr>, critical <chr>, gender <chr>,
#   item_id <dbl>, list <dbl>, match <chr>, condition <chr>, name <chr>, ...
```

```
# load tidyverse
df_lifetime <- df_lifetime %>%
  rename("px" = RECORDING_SESSION_LABEL,
         "trial" = TRIAL_INDEX)
```

Exercise

Change the following names:

`EYE_USED` to `eye` `IA_DWELL_TIME` to `tt` `IA_FIRST_FIXATION_DURATION` to `ff` `IA_FIXATION_COUNT` to `fix_count` `IA_FIRST_RUN_DWELL_TIME` to `fp` `IA_ID` to `region_n` `IA_LABEL` to `region_text`

`IA_REGRESSION_IN` to `reg_in` `IA_REGRESSION_IN_COUNT` to `reg_in_count` `IA_REGRESSION_OUT` to `reg_out` `IA_REGRESSION_OUT_COUNT` to `reg_out_count` `IA_REGRESSION_PATH_DURATION` to `rpd` `name_vital_status` to `lifetime`

```
names(df_lifetime)
```

```
 [1] "px"            "trial"         "eye"           "tt"
 [5] "ff"            "fp"            "fix_count"     "region_n"
 [9] "region_text"   "reg_in"        "reg_in_count"  "reg_out"
[13] "reg_out_count" "rpd"           "KeyPress"      "rt"
[17] "bio"           "critical"      "gender"        "item_id"
[21] "list"          "match"         "condition"     "name"
[25] "lifetime"      "tense"         "type"          "yes_press"
```

`mutate()`

Make some change

- new columns

```r
df_lifetime <- df_lifetime %>%
  mutate(new_column = "new")
```

- change existing column

```r
df_lifetime <- df_lifetime %>%
  mutate(new_column = px,
         trial = trial + 5)
```

- but let's undo that...

```r
df_lifetime <- df_lifetime %>%
  mutate(trial = trial - 5)
```

`if_else()`

- can be used inside `mutate()`
  - change values based on some logical condition
  - can be used to change an existing column, or create a new one

```r
df_lifetime <- df_lifetime %>%
  mutate(new_column = if_else(name=="Aaliyah","Aaliyah","not Aaliyah"))
```

`case_when()`

- can be used inside `mutate()`
  - change values based on multiple logical conditions
  - can be used to change an existing column, or create a new one

```r
df_lifetime <- df_lifetime %>%
  mutate(newer_column = case_when(
    name=="Aaliyah" & trial > 104 ~ "Aaliyah 2nd half",
    name=="Beyoncé" & (px == "px01" | px == "px04") ~ "Beyoncé px04 or px06",
    TRUE ~ "otherwise"))
```

Exercise

:::{.column width = "100%"} 1. Create a new variable `accept` that checks whether the button pressed (`KeyPress`) equals the button that corresponds to an acceptance (`yes_press`) + if `KeyPress` and `yes_press` are the same, `accept` should be `1`. If not, `accept` should be `0` + hint: you will need `if_else()` or `case_when()`

2. Create a new variable `accuracy` where:

- if `match` has the value `yes` and `accept` has the value `1`, `accuracy` is `1`
- if `match` has the value `no` and `accept` has the value `0`, `accuracy` is `1`
- if `match` has the value `yes` and `accept` has the value `0`, `accuracy` is `0`
- if `match` has the value `no` and `accept` has the value `1`, `accuracy` is `0` :::

:::{.column width = "50%"}

```
mean(df_lifetime$accept)
```

```
[1] 0.6068608
```

```
df_lifetime |>
  select(accept) |>
  mutate(accept = as_factor(accept)) |>
  summary()
```

```
 accept
 0:1742
 1:2689
```

::: :::{.column width = "50%"}

```
mean(df_lifetime$accuracy)
```

```
[1] 0.6267208
```

```
summary(as_factor(df_lifetime$accuracy))
```

```
   0    1
1654 2777
```

:::

`group_by()` and `ungroup()`

Group data by certain variable(s) + then perform some mutation + then ungroup the data

```
df_lifetime <- df_lifetime |>
  group_by(px) |>
  mutate(px_accuracy = mean(accuracy)) %>%
  ungroup()
```

```
round(
  range(df_lifetime$px_accuracy),
  2)
```

```
[1] 0.26 0.90
```

`select()`

- keep only certain column(s)

```
df_lifetime %>%
  select(px)
```

```
# A tibble: 4,431 x 1
   px
   <chr>
 1 px7
 2 px7
 3 px7
 4 px7
 5 px7
 6 px7
 7 px7
 8 px7
 9 px7
10 px7
# i 4,421 more rows
```

```
df_lifetime %>%
  select(px, trial)
```

```
# A tibble: 4,431 x 2
   px    trial
   <chr> <dbl>
 1 px7       1
 2 px7       2
 3 px7       3
 4 px7       3
 5 px7       3
 6 px7       3
 7 px7       3
 8 px7       3
 9 px7       4
10 px7       5
# i 4,421 more rows
```

`select()`

- or remove certain columns

```
df_lifetime %>%
  select(-px, -trial)
```

```
# A tibble: 4,431 x 31
   eye      tt    ff    fp fix_count region_n region_text    reg_in reg_in_count
   <chr> <dbl> <dbl> <dbl>     <dbl>    <dbl> <chr>           <dbl>        <dbl>
 1 RIGHT     0     0     0         0        1 He owned innu~      0            0
 2 RIGHT     0     0     0         0        1 She is a moth~      0            0
 3 RIGHT     0     0     0         0        1 She                 0            0
 4 RIGHT     0     0     0         0        2 will perform        0            0
 5 RIGHT     0     0     0         0        3 in prestigiou~      0            0
 6 RIGHT     0     0     0         0        4 in the future,      0            0
 7 RIGHT     0     0     0         0        5 as reported i~      0            0
 8 RIGHT     0     0     0         0        6 as reported i~      0            0
 9 RIGHT     0     0     0         0        1 He interviewe~      0            0
10 RIGHT     0     0     0         0        1 She                 0            0
# i 4,421 more rows
# i 22 more variables: reg_out <dbl>, reg_out_count <dbl>, rpd <dbl>,
#   KeyPress <dbl>, rt <dbl>, bio <chr>, critical <chr>, gender <chr>,
#   item_id <dbl>, list <dbl>, match <chr>, condition <chr>, name <chr>,
#   lifetime <chr>, tense <chr>, type <chr>, yes_press <dbl>, new_column <chr>,
#   newer_column <chr>, accept <dbl>, accuracy <dbl>, px_accuracy <dbl>
```

Exercise

Let's get rid of the example variables we created with `mutate`: `new_column` and `newer_column`

```
names(df_lifetime)
```

```
 [1] "px"            "trial"         "eye"           "tt"
 [5] "ff"            "fp"            "fix_count"     "region_n"
 [9] "region_text"   "reg_in"        "reg_in_count"  "reg_out"
[13] "reg_out_count" "rpd"           "KeyPress"      "rt"
[17] "bio"           "critical"      "gender"        "item_id"
[21] "list"          "match"         "condition"     "name"
[25] "lifetime"      "tense"         "type"          "yes_press"
[29] "accept"        "accuracy"      "px_accuracy"
```

`filter()`

- select certain rows based on certain criteria (`==`, `!=`, `>`, `<`, `|`)
  - N.B. when testing logical conditions `==` is needed

```
df_lifetime %>%
  filter(trial == 1)
```

```
# A tibble: 8 x 31
  px    trial eye      tt    ff    fp fix_count region_n region_text      reg_in
  <chr> <dbl> <chr> <dbl> <dbl> <dbl>     <dbl>    <dbl> <chr>              <dbl>
1 px7       1 RIGHT     0     0     0         0        1 He owned innume~       0
2 px6       1 RIGHT  1603   145  1603         9        1 She is a mother~       0
3 px2       1 RIGHT  1224   147  1224         5        1 He is a father ~       0
4 px1       1 RIGHT  1829    84  1829         7        1 She made innume~       0
5 px8       1 RIGHT  2456   138  2456         9        1 In the '70s, he~       0
6 px4       1 RIGHT  1708   160  1708         8        1 Beloved morning~       0
7 px3       1 RIGHT   806   220   806         3        1 She was a mothe~       0
8 px5       1 LEFT   3557   171  3557        16        1 In the '70s, he~       0
# i 21 more variables: reg_in_count <dbl>, reg_out <dbl>, reg_out_count <dbl>,
#   rpd <dbl>, KeyPress <dbl>, rt <dbl>, bio <chr>, critical <chr>,
#   gender <chr>, item_id <dbl>, list <dbl>, match <chr>, condition <chr>,
#   name <chr>, lifetime <chr>, tense <chr>, type <chr>, yes_press <dbl>,
#   accept <dbl>, accuracy <dbl>, px_accuracy <dbl>
```

filter()

```r
df_lifetime %>%
  filter(px_accuracy > .5)
```

```
# A tibble: 2,776 x 31
   px    trial eye      tt    ff    fp fix_count region_n region_text     reg_in
   <chr> <dbl> <chr> <dbl> <dbl> <dbl>     <dbl>    <dbl> <chr>            <dbl>
 1 px7       1 RIGHT     0     0     0         0        1 He owned innum~      0
 2 px7       2 RIGHT     0     0     0         0        1 She is a mothe~      0
 3 px7       3 RIGHT     0     0     0         0        1 She                  0
 4 px7       3 RIGHT     0     0     0         0        2 will perform         0
 5 px7       3 RIGHT     0     0     0         0        3 in prestigious~      0
 6 px7       3 RIGHT     0     0     0         0        4 in the future,       0
 7 px7       3 RIGHT     0     0     0         0        5 as reported in~      0
 8 px7       3 RIGHT     0     0     0         0        6 as reported in~      0
 9 px7       4 RIGHT     0     0     0         0        1 He interviewed~      0
10 px7       5 RIGHT     0     0     0         0        1 She                  0
# i 2,766 more rows
# i 21 more variables: reg_in_count <dbl>, reg_out <dbl>, reg_out_count <dbl>,
#   rpd <dbl>, KeyPress <dbl>, rt <dbl>, bio <chr>, critical <chr>,
#   gender <chr>, item_id <dbl>, list <dbl>, match <chr>, condition <chr>,
#   name <chr>, lifetime <chr>, tense <chr>, type <chr>, yes_press <dbl>,
#   accept <dbl>, accuracy <dbl>, px_accuracy <dbl>
```

```r
df_lifetime %>%
  filter(px == "px3")
```

```
# A tibble: 551 x 31
   px    trial eye      tt    ff    fp fix_count region_n region_text     reg_in
   <chr> <dbl> <chr> <dbl> <dbl> <dbl>     <dbl>    <dbl> <chr>            <dbl>
 1 px3       1 RIGHT   806   220   806         3        1 She was a moth~      0
 2 px3       2 RIGHT  2167   160  2167         7        1 He discovered ~      0
 3 px3       3 RIGHT   164   164   164         1        1 She                  0
 4 px3       3 RIGHT   999   432   432         3        2 has appeared         1
 5 px3       3 RIGHT   546   170   334         3        3 in popular mus~      0
 6 px3       3 RIGHT   728   569   569         2        4 in the past,         1
 7 px3       3 RIGHT   412   219   412         2        5 so Wikipedia s~      0
 8 px3       4 RIGHT  1066   166  1066         5        1 Many pieces we~      0
 9 px3       5 RIGHT     0     0     0         0        1 He                   0
```

```
10 px3       5 RIGHT   631   165   315         3        2 will perform         1
# i 541 more rows
# i 21 more variables: reg_in_count <dbl>, reg_out <dbl>, reg_out_count <dbl>,
#   rpd <dbl>, KeyPress <dbl>, rt <dbl>, bio <chr>, critical <chr>,
#   gender <chr>, item_id <dbl>, list <dbl>, match <chr>, condition <chr>,
#   name <chr>, lifetime <chr>, tense <chr>, type <chr>, yes_press <dbl>,
#   accept <dbl>, accuracy <dbl>, px_accuracy <dbl>
```

```r
df_lifetime %>%
  filter(px == "px3" | trial == "3")
```

```
# A tibble: 589 x 31
   px    trial eye      tt    ff    fp fix_count region_n region_text     reg_in
   <chr> <dbl> <chr> <dbl> <dbl> <dbl>     <dbl>    <dbl> <chr>             <dbl>
 1 px7       3 RIGHT     0     0     0         0        1 She                   0
 2 px7       3 RIGHT     0     0     0         0        2 will perform          0
 3 px7       3 RIGHT     0     0     0         0        3 in prestigious~       0
 4 px7       3 RIGHT     0     0     0         0        4 in the future,        0
 5 px7       3 RIGHT     0     0     0         0        5 as reported in~       0
 6 px7       3 RIGHT     0     0     0         0        6 as reported in~       0
 7 px6       3 RIGHT   190   190   190         1        1 She                   1
 8 px6       3 RIGHT   321   175   175         2        2 has worked            1
 9 px6       3 RIGHT  1723   154   154         9        3 with important~       0
10 px6       3 RIGHT   672   160   283         5        4 in the past,          0
# i 579 more rows
# i 21 more variables: reg_in_count <dbl>, reg_out <dbl>, reg_out_count <dbl>,
#   rpd <dbl>, KeyPress <dbl>, rt <dbl>, bio <chr>, critical <chr>,
#   gender <chr>, item_id <dbl>, list <dbl>, match <chr>, condition <chr>,
#   name <chr>, lifetime <chr>, tense <chr>, type <chr>, yes_press <dbl>,
#   accept <dbl>, accuracy <dbl>, px_accuracy <dbl>
```

```r
df_lifetime %>%
  filter(px == "px3" & trial != "3")
```

```
# A tibble: 546 x 31
   px    trial eye      tt    ff    fp fix_count region_n region_text     reg_in
   <chr> <dbl> <chr> <dbl> <dbl> <dbl>     <dbl>    <dbl> <chr>             <dbl>
 1 px3       1 RIGHT   806   220   806         3        1 She was a moth~       0
 2 px3       2 RIGHT  2167   160  2167         7        1 He discovered ~       0
 3 px3       4 RIGHT  1066   166  1066         5        1 Many pieces we~       0
```

```
 4 px3       5 RIGHT     0     0     0         0        1 He                  0
 5 px3       5 RIGHT   631   165   315         3        2 will perform        1
 6 px3       5 RIGHT  1086   291   553         5        3 in numerous mu~     0
 7 px3       5 RIGHT   422   228   228         2        4 in the future,      0
 8 px3       5 RIGHT   110   110   110         1        5 so news report~     0
 9 px3       5 RIGHT   196   196   196         1        6 so news report~     0
10 px3       6 RIGHT  2233   160  2233        10        1 Innumerable re~     0
# i 536 more rows
# i 21 more variables: reg_in_count <dbl>, reg_out <dbl>, reg_out_count <dbl>,
#   rpd <dbl>, KeyPress <dbl>, rt <dbl>, bio <chr>, critical <chr>,
#   gender <chr>, item_id <dbl>, list <dbl>, match <chr>, condition <chr>,
#   name <chr>, lifetime <chr>, tense <chr>, type <chr>, yes_press <dbl>,
#   accept <dbl>, accuracy <dbl>, px_accuracy <dbl>
```

Logical operators

- symbols used to describe a logical condition
- `==` is idential (`1 == 1`)
- `!=` is not identical (`1 != 2`)
- `>` is greater than (`2 > 1`)
- `<` is less than (`1 < 2`)
- `&` and also (for multiple conditions)
- `|` or (for multiple conditions)

Exercise

1. Create a new dataframe `df_crit` that includes only critical trials (tip: trial type is stored in the column `type`).
2. Create a new dataframe `df_fill` that includes only filler trials

::: {.column width = "50%"}

```
df_crit |> select(type) |> head()
```

```
# A tibble: 6 x 1
  type
  <chr>
1 critical
2 critical
3 critical
4 critical
5 critical
6 critical
```

:::

::: {.column width = "50%"}

```
df_fill |> select(type) |> head()
```

```
# A tibble: 6 x 1
  type
  <chr>
1 filler
2 filler
3 filler
4 filler
5 filler
6 filler
```

:::

`distinct()`

- like `filter()`, but for *distinct values* of a variable
  - "select rows with distinct values for some row(s)"

```
df_crit %>%
  distinct(px)
```

```
# A tibble: 8 x 1
  px
  <chr>
```

```
1 px7
2 px6
3 px2
4 px1
5 px8
6 px4
7 px3
8 px5
```

```
df_crit %>%
  distinct(px, name)
```

```
# A tibble: 639 x 2
   px    name
   <chr> <chr>
 1 px7   Edith Piaf
 2 px7   Aaliyah
 3 px7   David Beckham
 4 px7   Jana Novotna
 5 px7   Grace Kelly
 6 px7   Nigella Lawson
 7 px7   Coco Chanel
 8 px7   Ben Kingsley
 9 px7   Jim Carrey
10 px7   Judy Garland
# i 629 more rows
```

```
df_crit %>%
  distinct(px, name,
           .keep_all=T)
```

```
# A tibble: 639 x 31
   px    trial eye      tt    ff    fp fix_count region_n region_text reg_in
   <chr> <dbl> <chr> <dbl> <dbl> <dbl>     <dbl>    <dbl> <chr>        <dbl>
 1 px7       3 RIGHT     0     0     0         0        1 She              0
 2 px7       5 RIGHT     0     0     0         0        1 She              0
 3 px7       8 RIGHT     0     0     0         0        1 He               0
 4 px7      10 RIGHT     0     0     0         0        1 She              0
 5 px7      13 RIGHT     0     0     0         0        1 She              0
 6 px7      16 RIGHT     0     0     0         0        1 She              0
```

```
 7 px7      18 RIGHT     0     0     0         0        1 She              0
 8 px7      21 RIGHT     0     0     0         0        1 He               0
 9 px7      23 RIGHT     0     0     0         0        1 He               0
10 px7      26 RIGHT     0     0     0         0        1 She              0
# i 629 more rows
# i 21 more variables: reg_in_count <dbl>, reg_out <dbl>, reg_out_count <dbl>,
#   rpd <dbl>, KeyPress <dbl>, rt <dbl>, bio <chr>, critical <chr>,
#   gender <chr>, item_id <dbl>, list <dbl>, match <chr>, condition <chr>,
#   name <chr>, lifetime <chr>, tense <chr>, type <chr>, yes_press <dbl>,
#   accept <dbl>, accuracy <dbl>, px_accuracy <dbl>
```

arrange()

- sort column(s) in ascending or descending order
  - this is really just for ease of reading

```
# default: ascending order (A-Z)
df_crit %>%
  distinct(px, trial, name, condition) %>%
  arrange(px, trial)
```

```
# A tibble: 639 x 4
   px    trial name                    condition
   <chr> <dbl> <chr>                   <chr>
 1 px1       3 Angela Merkel           livingPP
 2 px1       5 Jennifer Lawrence       livingPP
 3 px1       8 Chimamanda Ngozi Adichi livingPP
 4 px1      10 Kurt Cobain             deadPP
 5 px1      13 Charlie Chaplin         deadSF
 6 px1      16 Edith Piaf              deadPP
 7 px1      18 Bridgette Bardot        deadSF
 8 px1      21 Grace Kelly             deadPP
 9 px1      23 Martin Luther King Jr.  deadSF
10 px1      26 Janet Jackson           livingPP
# i 629 more rows
```

```
# descending order (Z-A)
df_crit %>%
  distinct(px, trial, name, condition) %>%
  arrange(desc(px), trial)
```

```
# A tibble: 639 x 4
   px    trial name              condition
   <chr> <dbl> <chr>             <chr>
 1 px8       3 Daniel Radcliffe  livingSF
 2 px8       5 Maya Angelou      deadSF
 3 px8       8 Margaret Thatcher deadSF
 4 px8      10 Taylor Swift      livingSF
 5 px8      13 Mariah Carey      livingSF
 6 px8      16 James Cameron     livingPP
 7 px8      18 Kate Middleton    livingSF
 8 px8      21 Abraham Lincoln   deadPP
 9 px8      23 Ingrid Bergman    deadSF
10 px8      26 Helen Mirren      livingSF
# i 629 more rows
```

`separate()`

- create new columns from a single column

```
df_crit<- df_crit %>%
  separate(name,
           sep=" ",
           into = c("First","Last"),
           remove = F, # don't remove original column (name)
           extra = "merge") # if extra chunks, combine in 'Last' (von der...)
```

- opposite: `unite()`

`pivot_wider()`

`pivote_longer()`

Save your tidy data

- once your data is nice and tidy, save it with a **new filename**
  - this way you always have the same starting point for your data exploration/analyses

```
write.csv(df_crit, here::here("data/tidy_data_lifetime_pilot.csv"))
```

Session Info

```
sessionInfo()
```

```
R version 4.2.3 (2023-03-15)
Platform: aarch64-apple-darwin20 (64-bit)
Running under: macOS Ventura 13.2.1

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRlapack.dylib

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] lubridate_1.9.2 forcats_1.0.0   stringr_1.5.0   dplyr_1.1.0
 [5] purrr_1.0.1     readr_2.1.4     tidyr_1.3.0     tibble_3.2.1
 [9] ggplot2_3.4.2   tidyverse_2.0.0

loaded via a namespace (and not attached):
 [1] pillar_1.9.0     compiler_4.2.3   tools_4.2.3      bit_4.0.5
 [5] digest_0.6.31    timechange_0.2.0 jsonlite_1.8.4   evaluate_0.20
 [9] lifecycle_1.0.3  gtable_0.3.3     pkgconfig_2.0.3  png_0.1-8
[13] rlang_1.1.0      cli_3.6.0        rstudioapi_0.14  parallel_4.2.3
[17] curl_5.0.0       yaml_2.3.7       xfun_0.37        fastmap_1.1.1
[21] withr_2.5.0      httr_1.4.5       knitr_1.42       generics_0.1.3
[25] vctrs_0.6.1      fs_1.6.1         hms_1.1.3        bit64_4.0.5
[29] tidyselect_1.2.0 rprojroot_2.0.3  grid_4.2.3       glue_1.6.2
[33] here_1.0.1       R6_2.5.1         fansi_1.0.4      vroom_1.6.1
[37] rmarkdown_2.20   tzdb_0.3.0       magrittr_2.0.3   scales_1.2.1
[41] htmltools_0.5.4  rbbt_0.0.0.9000  colorspace_2.1-0 utf8_1.2.3
[45] stringi_1.7.12   munsell_0.5.0    crayon_1.5.2
```

References

Laurinavichyute, Anna, Himanshu Yadav, and Shravan Vasishth. 2022. "Share the Code, Not Just the Data: A Case Study of the Reproducibility of Articles Published in the Journal of

Memory and Language Under the Open Data Policy." *Journal of Memory and Language* 125: 12.

Wickham, Hadley, Mine Çetinkaya-Rundel, and Garrett Grolemund. n.d. *R for Data Science*. 2nd ed. https://r4ds.hadley.nz/.