Scooby-Doo

Danielle van Westbroek-Stibbe

2023-10-24

Scooby-Doo

The data used this week is collected on ScoobyPedia, an encyclopedia on the hit cartoon series Scooby-Doo. This is a quote from their website:

The show follows the iconic mystery solving detectives, know as Mystery Inc., as they set out to solve crime and unmask criminals, bent on revenge or committing criminal acts for their own personal gain.

Titular character, Scooby, is followed by his best pal Shaggy as both vie for Scooby Snacks on their adventures! Velma brings her extra intellect and initiative to them, setting out plans to catch criminals. Fred is the team's leader while Daphne is bold and full of personality.

The data we will be using has been coded for research purposes and demonstrated by the Tidy Tuesday Initiative. The data is described here:

https://github.com/rfordatascience/tidytuesday/blob/master/data/2021/2021-07-13/readme.md

Package loading

```
library(dplyr)     # For data wrangling
```

```
## Warning: package 'dplyr' was built under R version 4.2.3
```

```
library(ggplot2)   # For plotting
```

```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```
library(readr)     # For loading data
```

```
## Warning: package 'readr' was built under R version 4.2.3
```

```
library(stringr)   # For string operations
```

```
## Warning: package 'stringr' was built under R version 4.2.3
```

```
library(tibble)    # For easily transforming data to tibble
```

```
## Warning: package 'tibble' was built under R version 4.2.3
```

```
library(tidyverse) # For tidy work
```

```
## Warning: package 'tidyverse' was built under R version 4.2.3
```

```
## Warning: package 'tidyr' was built under R version 4.2.3
```

```
## Warning: package 'purrr' was built under R version 4.2.3
```

```
## Warning: package 'forcats' was built under R version 4.2.3
```

```
## Warning: package 'lubridate' was built under R version 4.2.3
```

Data loading

We can load the data directly from Github.

```
scooby_doo <- read_csv("https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2021/
```

```
## Rows: 603 Columns: 75
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (60): series_name, network, season, title, imdb, engagement, format, mo...
## dbl   (5): index, run_time, monster_amount, suspects_amount, culprit_amount
## lgl   (9): unmask_other, caught_other, caught_not, door_gag, batman, scooby_...
## date  (1): date_aired
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Inspect the data structure:

```
scooby_doo %>%
  glimpse()
```

```
## Rows: 603
## Columns: 75
## $ index                   <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14~
## $ series_name             <chr> "Scooby Doo, Where Are You!", "Scooby Doo, Wh~
## $ network                 <chr> "CBS", "CBS", "CBS", "CBS", "CBS", "CBS", "CB~
## $ season                  <chr> "1", "1", "1", "1", "1", "1", "1", "1", "1", ~
## $ title                   <chr> "What a Night for a Knight", "A Clue for Scoo~
## $ imdb                    <chr> "8.1", "8.1", "8", "7.8", "7.5", "8.4", "7.6"~
## $ engagement              <chr> "556", "479", "455", "426", "391", "384", "35~
## $ date_aired              <date> 1969-09-13, 1969-09-20, 1969-09-27, 1969-10-~
## $ run_time                <dbl> 21, 22, 21, 21, 21, 21, 21, 21, 21, 21, 21, 2~
## $ format                  <chr> "TV Series", "TV Series", "TV Series", "TV Se~
## $ monster_name            <chr> "Black Knight", "Ghost of Cptn. Cuttler", "Ph~
## $ monster_gender          <chr> "Male", "Male", "Male", "Male", "Female", "Ma~
## $ monster_type            <chr> "Possessed Object", "Ghost", "Ghost", "Ancien~
## $ monster_subtype         <chr> "Suit", "Suit", "Phantom", "Miner", "Witch Do~
```

```
## $ monster_species          <chr> "Object", "Human", "Human", "Human", "Human",~
## $ monster_real             <chr> "FALSE", "FALSE", "FALSE", "FALSE", "FALSE", ~
## $ monster_amount           <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 1, 2, 1, 1, ~
## $ caught_fred              <chr> "FALSE", "FALSE", "FALSE", "TRUE", "FALSE", "~
## $ caught_daphnie           <chr> "FALSE", "FALSE", "FALSE", "FALSE", "FALSE", ~
## $ caught_velma             <chr> "FALSE", "FALSE", "FALSE", "FALSE", "FALSE", ~
## $ caught_shaggy            <chr> "TRUE", "TRUE", "FALSE", "FALSE", "FALSE", "F~
## $ caught_scooby            <chr> "TRUE", "FALSE", "TRUE", "FALSE", "TRUE", "FA~
## $ captured_fred            <chr> "FALSE", "TRUE", "FALSE", "FALSE", "FALSE", "~
## $ captured_daphnie         <chr> "FALSE", "TRUE", "FALSE", "FALSE", "FALSE", "~
## $ captured_velma           <chr> "FALSE", "TRUE", "FALSE", "FALSE", "FALSE", "~
## $ captured_shaggy          <chr> "FALSE", "FALSE", "FALSE", "FALSE", "FALSE", ~
## $ captured_scooby          <chr> "FALSE", "FALSE", "FALSE", "FALSE", "TRUE", "~
## $ unmask_fred              <chr> "FALSE", "TRUE", "TRUE", "TRUE", "FALSE", "TR~
## $ unmask_daphnie           <chr> "FALSE", "FALSE", "FALSE", "FALSE", "FALSE", ~
## $ unmask_velma             <chr> "FALSE", "FALSE", "FALSE", "FALSE", "FALSE", ~
## $ unmask_shaggy            <chr> "FALSE", "FALSE", "FALSE", "FALSE", "FALSE", ~
## $ unmask_scooby            <chr> "TRUE", "FALSE", "FALSE", "FALSE", "TRUE", "F~
## $ snack_fred               <chr> "TRUE", "FALSE", "TRUE", "FALSE", "FALSE", "T~
## $ snack_daphnie            <chr> "FALSE", "FALSE", "FALSE", "TRUE", "TRUE", "F~
## $ snack_velma              <chr> "FALSE", "TRUE", "FALSE", "FALSE", "FALSE", "~
## $ snack_shaggy             <chr> "FALSE", "FALSE", "FALSE", "FALSE", "FALSE", ~
## $ snack_scooby             <chr> "FALSE", "FALSE", "FALSE", "FALSE", "FALSE", ~
## $ unmask_other             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ caught_other             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ caught_not               <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ trap_work_first          <chr> "NULL", "FALSE", "FALSE", "TRUE", "NULL", "TR~
## $ setting_terrain          <chr> "Urban", "Coast", "Island", "Cave", "Desert",~
## $ setting_country_state    <chr> "United States", "United States", "United Sta~
## $ suspects_amount          <dbl> 2, 2, 0, 2, 1, 2, 1, 2, 1, 1, 1, 1, 2, 2, 1, ~
## $ non_suspect              <chr> "FALSE", "TRUE", "TRUE", "FALSE", "FALSE", "F~
## $ arrested                 <chr> "TRUE", "TRUE", "TRUE", "TRUE", "TRUE", "TRUE~
## $ culprit_name             <chr> "Mr. Wickles", "Cptn. Cuttler", "Bluestone th~
## $ culprit_gender           <chr> "Male", "Male", "Male", "Male", "Male", "Male~
## $ culprit_amount           <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, ~
## $ motive                   <chr> "Theft", "Theft", "Treasure", "Natural Resour~
## $ if_it_wasnt_for          <chr> "NULL", "NULL", "NULL", "NULL", "NULL", "NULL~
## $ and_that                 <chr> "NULL", "NULL", "NULL", "NULL", "NULL", "NULL~
## $ door_gag                 <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ number_of_snacks         <chr> "2", "1", "3", "2", "2", "4", "4", "0", "1", ~
## $ split_up                 <chr> "1", "0", "0", "1", "0", "0", "1", "0", "0", ~
## $ another_mystery          <chr> "1", "0", "0", "0", "1", "0", "0", "0", "0", ~
## $ set_a_trap               <chr> "0", "0", "0", "0", "0", "0", "1", "1", "0", ~
## $ jeepers                  <chr> "0", "0", "0", "0", "0", "1", "0", "0", "0", ~
## $ jinkies                  <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", ~
## $ my_glasses               <chr> "1", "0", "0", "0", "1", "0", "0", "1", "0", ~
## $ just_about_wrapped_up    <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", ~
## $ zoinks                   <chr> "1", "3", "1", "2", "0", "2", "1", "0", "0", ~
## $ groovy                   <chr> "0", "0", "2", "1", "0", "0", "1", "0", "0", ~
## $ scooby_doo_where_are_you <chr> "0", "1", "0", "0", "1", "0", "0", "1", "0", ~
## $ rooby_rooby_roo          <chr> "1", "0", "0", "0", "0", "1", "1", "1", "1", ~
## $ batman                   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ scooby_dum               <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ scrappy_doo              <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
```

```
## $ hex_girls                <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ blue_falcon              <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ fred_va                  <chr> "Frank Welker", "Frank Welker", "Frank Welker~
## $ daphnie_va               <chr> "Stefanianna Christopherson", "Stefanianna Ch~
## $ velma_va                 <chr> "Nicole Jaffe", "Nicole Jaffe", "Nicole Jaffe~
## $ shaggy_va                <chr> "Casey Kasem", "Casey Kasem", "Casey Kasem", ~
## $ scooby_va                <chr> "Don Messick", "Don Messick", "Don Messick", ~
```

Q1: What can we learn about the data from glimpsing it? What is the unit of observation? Are all variables coded properly? How can we fix it?

1. One episode per row.
2. Meta-data about episode: series name, network, season, title, IMDB rating, engagement sore, date aired, run time, format, voice actors, and sound effects.
3. Data about episode content: about monsters, catching/captured/unmasking characters, about the suspects and whether the suspects were the culprits, and snacks
4. It appears that the data is "wide", with one observation (=episode) per row.
5. Null values are treated as character, which means that R doesn't understand they are null. We can fix it by adding an argument to the data loading:

```
scooby_doo <- read_csv("https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2
```

```
## Rows: 603 Columns: 75
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (24): series_name, network, season, title, format, monster_name, monste...
## dbl  (18): index, imdb, engagement, run_time, monster_amount, suspects_amoun...
## lgl  (32): monster_real, caught_fred, caught_daphnie, caught_velma, caught_s...
## date  (1): date_aired
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
glimpse(scooby_doo)
```

```
## Rows: 603
## Columns: 75
## $ index                    <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14~
## $ series_name              <chr> "Scooby Doo, Where Are You!", "Scooby Doo, Wh~
## $ network                  <chr> "CBS", "CBS", "CBS", "CBS", "CBS", "CBS", "CB~
## $ season                   <chr> "1", "1", "1", "1", "1", "1", "1", "1", "1", ~
## $ title                    <chr> "What a Night for a Knight", "A Clue for Scoo~
## $ imdb                     <dbl> 8.1, 8.1, 8.0, 7.8, 7.5, 8.4, 7.6, 8.2, 8.1, ~
## $ engagement               <dbl> 556, 479, 455, 426, 391, 384, 358, 358, 371, ~
## $ date_aired               <date> 1969-09-13, 1969-09-20, 1969-09-27, 1969-10-~
## $ run_time                 <dbl> 21, 22, 21, 21, 21, 21, 21, 21, 21, 21, 21, 2~
## $ format                   <chr> "TV Series", "TV Series", "TV Series", "TV Se~
## $ monster_name             <chr> "Black Knight", "Ghost of Cptn. Cuttler", "Ph~
## $ monster_gender           <chr> "Male", "Male", "Male", "Male", "Female", "Ma~
## $ monster_type             <chr> "Possessed Object", "Ghost", "Ghost", "Ancien~
```

```
## $ monster_subtype          <chr> "Suit", "Suit", "Phantom", "Miner", "Witch Do~
## $ monster_species          <chr> "Object", "Human", "Human", "Human", "Human",~
## $ monster_real             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ monster_amount           <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 1, 2, 1, 1, ~
## $ caught_fred              <lgl> FALSE, FALSE, FALSE, TRUE, FALSE, TRUE, TRUE,~
## $ caught_daphnie           <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ caught_velma             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ caught_shaggy            <lgl> TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE~
## $ caught_scooby            <lgl> TRUE, FALSE, TRUE, FALSE, TRUE, FALSE, FALSE,~
## $ captured_fred            <lgl> FALSE, TRUE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ captured_daphnie         <lgl> FALSE, TRUE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ captured_velma           <lgl> FALSE, TRUE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ captured_shaggy          <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ captured_scooby          <lgl> FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, FALS~
## $ unmask_fred              <lgl> FALSE, TRUE, TRUE, TRUE, FALSE, TRUE, FALSE, ~
## $ unmask_daphnie           <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ unmask_velma             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ unmask_shaggy            <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, TRU~
## $ unmask_scooby            <lgl> TRUE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE~
## $ snack_fred               <lgl> TRUE, FALSE, TRUE, FALSE, FALSE, TRUE, FALSE,~
## $ snack_daphnie            <lgl> FALSE, FALSE, FALSE, TRUE, TRUE, FALSE, FALSE~
## $ snack_velma              <lgl> FALSE, TRUE, FALSE, FALSE, FALSE, FALSE, TRUE~
## $ snack_shaggy             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ snack_scooby             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ unmask_other             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ caught_other             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ caught_not               <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ trap_work_first          <chr> NA, "FALSE", "FALSE", "TRUE", NA, "TRUE", "FA~
## $ setting_terrain          <chr> "Urban", "Coast", "Island", "Cave", "Desert",~
## $ setting_country_state    <chr> "United States", "United States", "United Sta~
## $ suspects_amount          <dbl> 2, 2, 0, 2, 1, 2, 1, 2, 1, 1, 1, 1, 2, 2, 1, ~
## $ non_suspect              <lgl> FALSE, TRUE, TRUE, FALSE, FALSE, FALSE, FALSE~
## $ arrested                 <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FAL~
## $ culprit_name             <chr> "Mr. Wickles", "Cptn. Cuttler", "Bluestone th~
## $ culprit_gender           <chr> "Male", "Male", "Male", "Male", "Male", "Male~
## $ culprit_amount           <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, ~
## $ motive                   <chr> "Theft", "Theft", "Treasure", "Natural Resour~
## $ if_it_wasnt_for          <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "thes~
## $ and_that                 <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "dog"~
## $ door_gag                 <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ number_of_snacks         <chr> "2", "1", "3", "2", "2", "4", "4", "0", "1", ~
## $ split_up                 <dbl> 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, ~
## $ another_mystery          <dbl> 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ set_a_trap               <dbl> 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, ~
## $ jeepers                  <dbl> 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ jinkies                  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ my_glasses               <dbl> 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, ~
## $ just_about_wrapped_up    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ zoinks                   <dbl> 1, 3, 1, 2, 0, 2, 1, 0, 0, 0, 0, 6, 3, 5, 8, ~
## $ groovy                   <dbl> 0, 0, 2, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, ~
## $ scooby_doo_where_are_you <dbl> 0, 1, 0, 0, 1, 0, 0, 1, 0, 2, 0, 0, 0, 1, 0, ~
## $ rooby_rooby_roo          <dbl> 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 3, 0, 0, 0, ~
## $ batman                   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ scooby_dum               <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
```

```
## $ scrappy_doo             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ hex_girls               <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ blue_falcon             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL~
## $ fred_va                 <chr> "Frank Welker", "Frank Welker", "Frank Welker~
## $ daphnie_va              <chr> "Stefanianna Christopherson", "Stefanianna Ch~
## $ velma_va                <chr> "Nicole Jaffe", "Nicole Jaffe", "Nicole Jaffe~
## $ shaggy_va               <chr> "Casey Kasem", "Casey Kasem", "Casey Kasem", ~
## $ scooby_va               <chr> "Don Messick", "Don Messick", "Don Messick", ~
```

Filtering

Say we only want to use data from the original series "Scooby Doo, Where Are You!", and later work with that dataset.

Exercise A: create a new dataset called `scooby_doo_orig` by filtering the original dataset.

Q2: How many episode did the original series have?

```
scooby_doo_orig <- scooby_doo %>%
  filter(series_name == "Scooby Doo, Where Are You!")

scooby_doo_orig %>%
  nrow()
```

```
## [1] 25
```

Selecting

For a later exercise, we wish to subset the dataset of the original series to include each episode's index, title, and monster data.

Exercise B: Create this dataset and call it `scooby_doo_orig_monster`. Check that you have 9 columns in the new dataset.

```
scooby_doo_orig_monster <- scooby_doo_orig %>%
  select(index, title, monster_name:monster_amount)

scooby_doo_orig_monster <- scooby_doo_orig %>%
  select(index, title, starts_with("monster")) # this also works

scooby_doo_orig_monster %>%
  ncol()
```

```
## [1] 9
```

What is the average IMDB rating per season of each Scooby-Doo series?

For this, we will use the full dataset again.

Exercise C: Inspect the number of episodes in each season per series (hint: try grouping the appropriate variables and counting the resulting number of cases):

```
scooby_doo %>%
  group_by(series_name, season) %>%
  count()
```

```
## # A tibble: 47 x 3
## # Groups:   series_name, season [47]
##    series_name                     season        n
##    <chr>                           <chr>     <int>
##  1 A Pup Named Scooby-Doo          1            13
##  2 A Pup Named Scooby-Doo          2             8
##  3 A Pup Named Scooby-Doo          3             4
##  4 A Pup Named Scooby-Doo          4             5
##  5 Be Cool, Scooby-Doo!            1            26
##  6 Be Cool, Scooby-Doo!            2            27
##  7 Dynomutt, Dogwonder             Crossover     3
##  8 Hanna-Barbera Superstars 10     Special       3
##  9 Harvey Birdman, Attorney at Law Crossover     1
## 10 Johnny Bravo                    Crossover     1
## # i 37 more rows
```

There are 47 unique series-season combination. Inspecting the results a little tells us that there are some special and crossover episodes from series that are not Scooby Doo-centered. There are also a few movies. Say we are only interested in regular Scooby Doo episodes.

NOTE: You can do more things with group_by, for example adding a new column where you enumerate the episodes:

```
scooby_doo <- scooby_doo %>%
  group_by(series_name, season) %>%
  mutate(ep_number = row_number()) %>%
  ungroup()

scooby_doo %>%
  select(series_name, season, ep_number) %>%
  head(n = 20)
```

```
## # A tibble: 20 x 3
##    series_name                season ep_number
##    <chr>                      <chr>      <int>
##  1 Scooby Doo, Where Are You! 1              1
##  2 Scooby Doo, Where Are You! 1              2
##  3 Scooby Doo, Where Are You! 1              3
##  4 Scooby Doo, Where Are You! 1              4
##  5 Scooby Doo, Where Are You! 1              5
##  6 Scooby Doo, Where Are You! 1              6
##  7 Scooby Doo, Where Are You! 1              7
##  8 Scooby Doo, Where Are You! 1              8
##  9 Scooby Doo, Where Are You! 1              9
## 10 Scooby Doo, Where Are You! 1             10
## 11 Scooby Doo, Where Are You! 1             11
## 12 Scooby Doo, Where Are You! 1             12
## 13 Scooby Doo, Where Are You! 1             13
## 14 Scooby Doo, Where Are You! 1             14
```

```
## 15 Scooby Doo, Where Are You! 1             15
## 16 Scooby Doo, Where Are You! 1             16
## 17 Scooby Doo, Where Are You! 1             17
## 18 Scooby Doo, Where Are You! 2              1
## 19 Scooby Doo, Where Are You! 2              2
## 20 Scooby Doo, Where Are You! 2              3
```

When we use group_by to create a new variable, we should remember to ungroup, so that new operations don't only happen to grouped data.

Exercise D: Create a new dataset called `scooby_doo_ep` that only include Scooby Doo TV Series and segmented TV series.

The following lines all give the same results!

```
scooby_doo_ep <- scooby_doo %>%
  filter(season != "Crossover" & season != "Special" & season != "Movie")

scooby_doo_ep <- scooby_doo %>%
  filter(! is.na(season %>% as.numeric()))
```

```
## Warning: There was 1 warning in `filter()`.
## i In argument: `!is.na(season %>% as.numeric())`.
## Caused by warning in `season %>% as.numeric()`:
## ! NAs introduced by coercion
```

```
scooby_doo_ep <- scooby_doo %>%
  filter(format %>% str_detect("TV Series") &
           season != "Special")

scooby_doo_ep %>%
  group_by(series_name, season) %>%
  count()
```

```
## # A tibble: 31 x 3
## # Groups:   series_name, season [31]
##    series_name                season     n
##    <chr>                      <chr>  <int>
##  1 A Pup Named Scooby-Doo     1         13
##  2 A Pup Named Scooby-Doo     2          8
##  3 A Pup Named Scooby-Doo     3          4
##  4 A Pup Named Scooby-Doo     4          5
##  5 Be Cool, Scooby-Doo!       1         26
##  6 Be Cool, Scooby-Doo!       2         27
##  7 Laff-a-Lympics             1         32
##  8 Laff-a-Lympics             2         16
##  9 Scooby Doo, Where Are You! 1         17
## 10 Scooby Doo, Where Are You! 2          8
## # i 21 more rows
```

Exercise E: Summarize the average IMDB rating per season of each series:

```
scooby_doo_ep %>%
  group_by(series_name, season) %>%
  summarize(mean_imdb = mean(imdb))
```

```
## `summarise()` has grouped output by 'series_name'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 31 x 3
## # Groups:   series_name [15]
##    series_name                season mean_imdb
##    <chr>                      <chr>      <dbl>
##  1 A Pup Named Scooby-Doo     1           7.45
##  2 A Pup Named Scooby-Doo     2           7.62
##  3 A Pup Named Scooby-Doo     3           7.08
##  4 A Pup Named Scooby-Doo     4           6.84
##  5 Be Cool, Scooby-Doo!       1           7.54
##  6 Be Cool, Scooby-Doo!       2           7.33
##  7 Laff-a-Lympics             1           6.69
##  8 Laff-a-Lympics             2           6.59
##  9 Scooby Doo, Where Are You! 1           8.12
## 10 Scooby Doo, Where Are You! 2           8.09
## # i 21 more rows
```

How well did Scooby's team do on solving crime?

There are several questions we can answer with this data. For these questions, we will use the `scooby_doo_ep` dataset.

What percentage of the real culprits were unsuspected? We can find this out by summing up the number of "TRUE" in the column `non_suspect`. We can then compare this number to the number of culprits who were suspected (marked by "FALSE").

```
non_sus <- scooby_doo_ep %>%
  select(non_suspect) %>%
  sum(na.rm = TRUE)

sus <- scooby_doo_ep %>%
  filter(! non_suspect) %>%
  nrow()

non_sus * 100 / (non_sus + sus)
```

```
## [1] 10.23018
```

10% of the culprits in the show were unsuspected!

Who caught the most culprits? To answer this question, we need to count the number of "TRUE" in each column containing the term "caught".

1. Select columns containing the term "caught":

```
caught <- scooby_doo_ep %>%
  select(starts_with("caught"))

glimpse(caught)
```

```
## Rows: 540
## Columns: 7
## $ caught_fred    <lgl> FALSE, FALSE, FALSE, TRUE, FALSE, TRUE, TRUE, FALSE, FA~
## $ caught_daphnie <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,~
## $ caught_velma   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,~
## $ caught_shaggy  <lgl> TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, T~
## $ caught_scooby  <lgl> TRUE, FALSE, TRUE, FALSE, TRUE, FALSE, FALSE, FALSE, FA~
## $ caught_other   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,~
## $ caught_not     <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,~
```

We have a dataset containing one column per character, with a logical value for each episode (caught/not). By summing up each column, we can find out the number of times "TRUE" is coded for each character.

Step 2: Sum up each column to find the number of "TRUE" incidences

```
caught_total <- caught %>%
  colSums(na.rm = TRUE) %>% # Results in a named vector
  enframe()                 # Nicely transforms the named vector to a dataframe.

caught_total
```

```
## # A tibble: 7 x 2
##   name           value
##   <chr>          <dbl>
## 1 caught_fred      113
## 2 caught_daphnie    21
## 3 caught_velma      31
## 4 caught_shaggy     66
## 5 caught_scooby    135
## 6 caught_other      68
## 7 caught_not        29
```

Step 3: Sort caught_total in a descending order to easily find the best monster catcher.

```
caught_total %>%
  arrange(value %>% desc())
```

```
## # A tibble: 7 x 2
##   name           value
##   <chr>          <dbl>
## 1 caught_scooby    135
## 2 caught_fred      113
## 3 caught_other      68
## 4 caught_shaggy     66
## 5 caught_velma      31
## 6 caught_not        29
## 7 caught_daphnie    21
```

```
# It looks like Scooby is our champion!
```

Who unmasked the most monsters? **Exercise G**: Repeat the steps above with the appropriate variables to find out.

```
unmasked <- scooby_doo_ep %>%
  select(starts_with("unmask"))

unmasked_total <- unmasked %>%
  colSums(na.rm = TRUE) %>%
  enframe()

unmasked_total %>%
  arrange(value %>% desc())
```

```
## # A tibble: 6 x 2
##   name           value
##   <chr>          <dbl>
## 1 unmask_fred       88
## 2 unmask_velma      71
## 3 unmask_other      34
## 4 unmask_daphnie    29
## 5 unmask_scooby     21
## 6 unmask_shaggy     11
```

```
# Fred unmasked the most monsters!
```

Who was captured the most times? **Exercise H**: Same task!

```
captured <- scooby_doo_ep %>%
  select(starts_with("captured"))

captured_total <- captured %>%
  colSums(na.rm = TRUE) %>%
  enframe()

captured_total %>%
  arrange(value %>% desc())
```

```
## # A tibble: 5 x 2
##   name             value
##   <chr>            <dbl>
## 1 captured_daphnie    72
## 2 captured_shaggy     68
## 3 captured_scooby     66
## 4 captured_velma      55
## 5 captured_fred       54
```

```
# Daphnie was captured the most!
```

Can we break a myth?

Scooby-Doo is known for unmasking many monsters to be fake. Of the non-human monsters, how many were real?

Let's first inspect the different categories under the variable `monster_type`:

```
scooby_doo_ep %>%
  group_by(monster_type) %>%
  count()
```

```
## # A tibble: 112 x 2
## # Groups:   monster_type [112]
##    monster_type                                                                n
##    <chr>                                                                   <int>
##  1 Ancient                                                                    18
##  2 Ancient,Animal                                                              2
##  3 Ancient,Ghost,Extraterrestrial,Animal,Disguised,Ghost,Possessed Object~     1
##  4 Ancient,Possessed                                                           1
##  5 Ancient,Possessed Object                                                    1
##  6 Animal                                                                     68
##  7 Animal,Ancient                                                              1
##  8 Animal,Ancient,Ancient,Mechanical,Plant                                     1
##  9 Animal,Animal                                                               7
## 10 Animal,Animal,Animal                                                        2
## # i 102 more rows
```

We quickly notice that some cells are coded with multiple monster_type values. That is because the dataset is structured in a *wide* format, with one episode per row. If we wish to analyze the different mosters, we need to pivot the dataset to a longer format, analyzing one monster per row. For simplicity, we will only focus on monster_type, although more variables are structured the same way.

```
scooby_doo_ep_longtype <- scooby_doo_ep %>%
  separate_longer_delim(cols = monster_type, delim = ",")


scooby_doo_ep_longtype
```

Longer dataset by deliminator

```
## # A tibble: 977 x 76
##    index series_name   network season title  imdb engagement date_aired run_time
##    <dbl> <chr>         <chr>   <chr>  <chr> <dbl>      <dbl> <date>        <dbl>
##  1     1 Scooby Doo, ~ CBS     1      What~   8.1        556 1969-09-13       21
##  2     2 Scooby Doo, ~ CBS     1      A Cl~   8.1        479 1969-09-20       22
##  3     3 Scooby Doo, ~ CBS     1      Hass~   8          455 1969-09-27       21
##  4     4 Scooby Doo, ~ CBS     1      Mine~   7.8        426 1969-10-04       21
##  5     5 Scooby Doo, ~ CBS     1      Deco~   7.5        391 1969-10-11       21
##  6     6 Scooby Doo, ~ CBS     1      What~   8.4        384 1969-10-18       21
##  7     7 Scooby Doo, ~ CBS     1      Neve~   7.6        358 1969-10-25       21
```

```
##  8     8 Scooby Doo, ~ CBS     1      Foul~   8.2        358 1969-11-01       21
##  9     9 Scooby Doo, ~ CBS     1      The ~   8.1        371 1969-11-08       21
## 10    10 Scooby Doo, ~ CBS     1      Bedl~   8          346 1969-11-15       21
## # i 967 more rows
## # i 67 more variables: format <chr>, monster_name <chr>, monster_gender <chr>,
## #   monster_type <chr>, monster_subtype <chr>, monster_species <chr>,
## #   monster_real <lgl>, monster_amount <dbl>, caught_fred <lgl>,
## #   caught_daphnie <lgl>, caught_velma <lgl>, caught_shaggy <lgl>,
## #   caught_scooby <lgl>, captured_fred <lgl>, captured_daphnie <lgl>,
## #   captured_velma <lgl>, captured_shaggy <lgl>, captured_scooby <lgl>, ...
```

Exercise I: filter to select "real" monsters only and count the number of incidences for each monster type left.

```
scooby_doo_ep_longtype %>%
  filter(monster_real) %>%
  group_by(monster_type) %>%
  count()
```

```
## # A tibble: 15 x 2
## # Groups:   monster_type [15]
##    monster_type           n
##    <chr>              <int>
##  1 " Disguised"           2
##  2 "Ancient"              7
##  3 "Animal"              30
##  4 "Disguised"          127
##  5 "Disugised"            1
##  6 "Dr. Trebal"           2
##  7 "Extraterrestrial"     8
##  8 "Ghost"               29
##  9 "Magician"            13
## 10 "Mechanical"           9
## 11 "Mythical"            18
## 12 "Plant"                6
## 13 "Possessed Object"     7
## 14 "Super-Villain"       90
## 15 "Undead"              20
```

It appears that at least some of the mythical monsters were real!

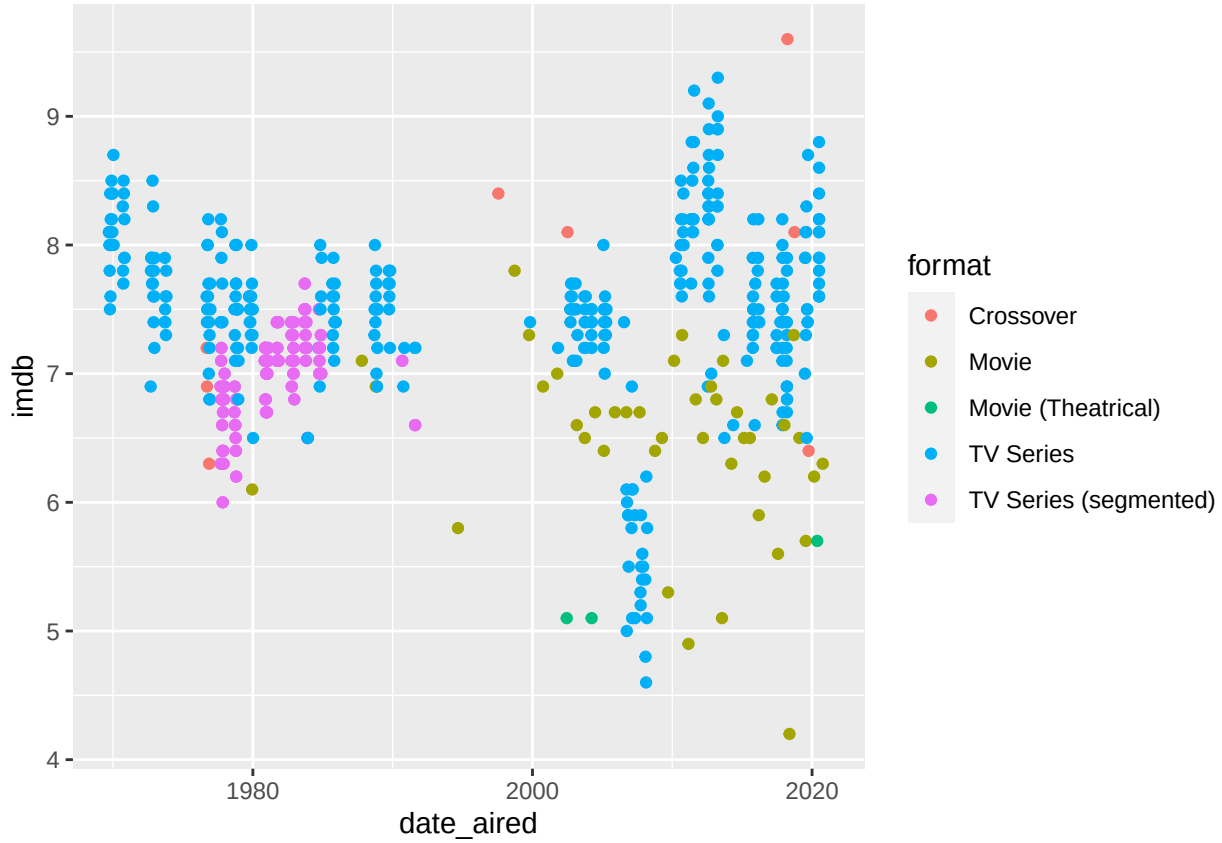
Data visualization

In case we have time left at the workshop today, let's try plotting some of the things we analyzed!

Plot 1: IMDB Rating per date, for different formats For GGPlot, we always have to specify an aesthetic argument, where we specify what should be treated as the x, y, or group variable. After we specify that, we can add new arguments by using the "+" sign and adding more and more layers to the plot. For example, if we want to produce a point-plot, we can do so with:

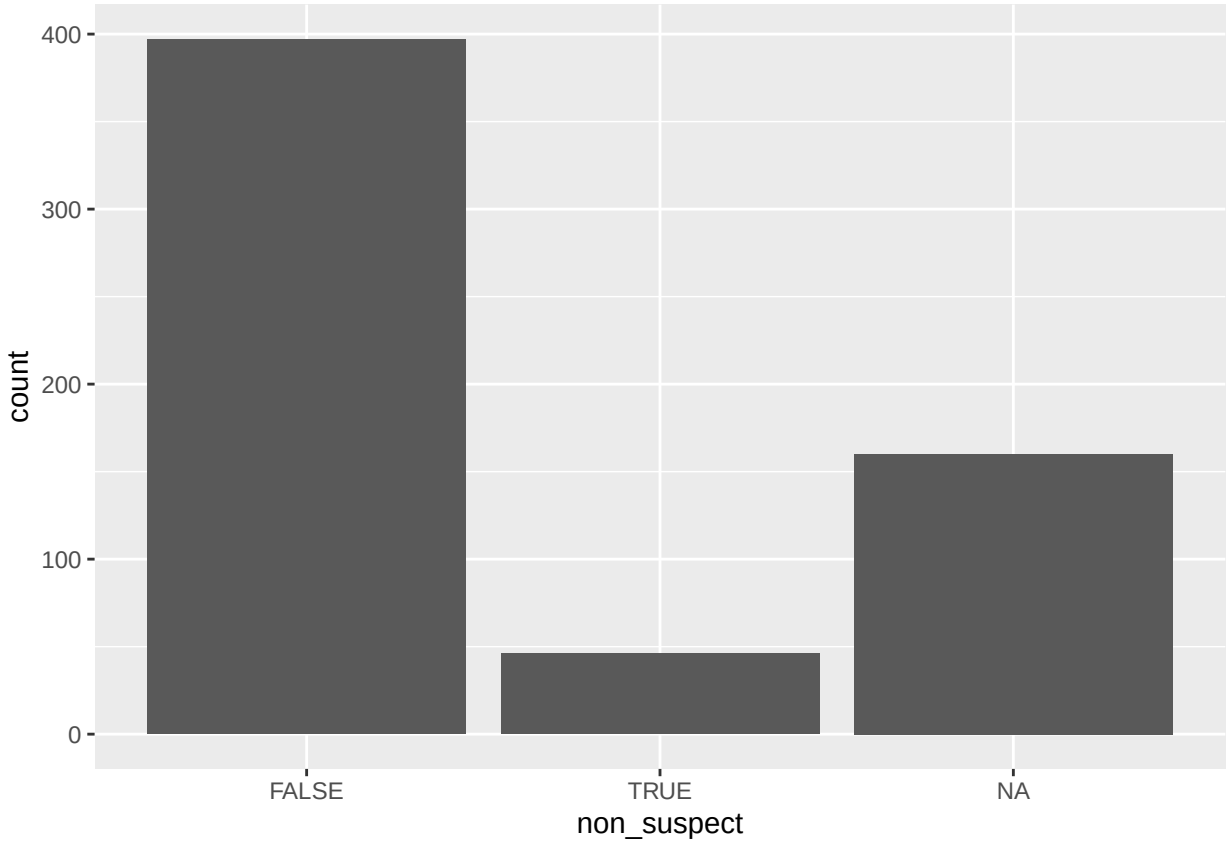
```
scooby_doo %>%
  ggplot(aes(x = date_aired,
             y = imdb,
             color = format)
         ) +
  geom_point()
```

```
## Warning: Removed 15 rows containing missing values (‘geom_point()‘).
```



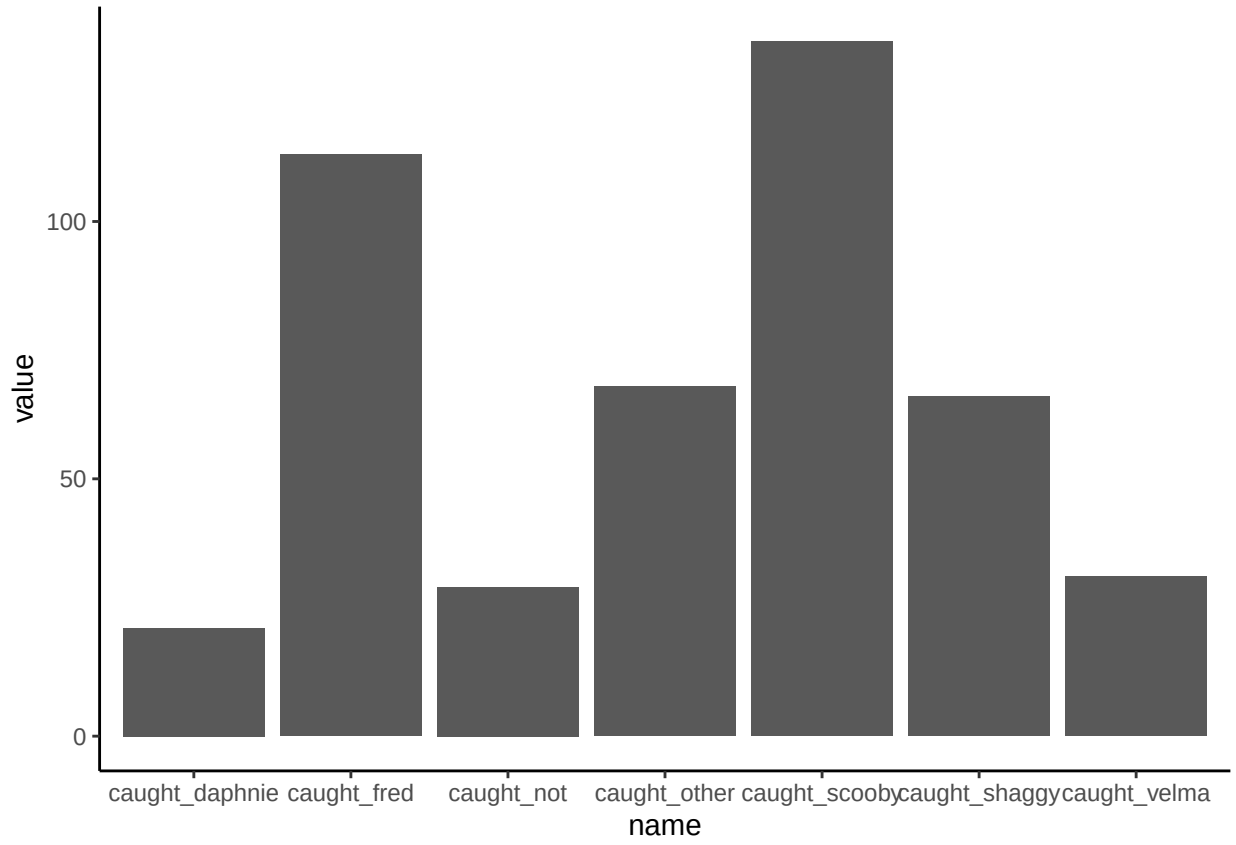
Plot 2: Suspected vs unsuspected culprits in different formats For a bar chart, we only need to specify an x or a y variable:

```
scooby_doo %>%
  ggplot(aes(x = non_suspect)) +
  geom_bar()
```



Plot 3: Who caught the most culprits? We can plot a summary table like the one we created for "caught_total" with a geom_col() command.

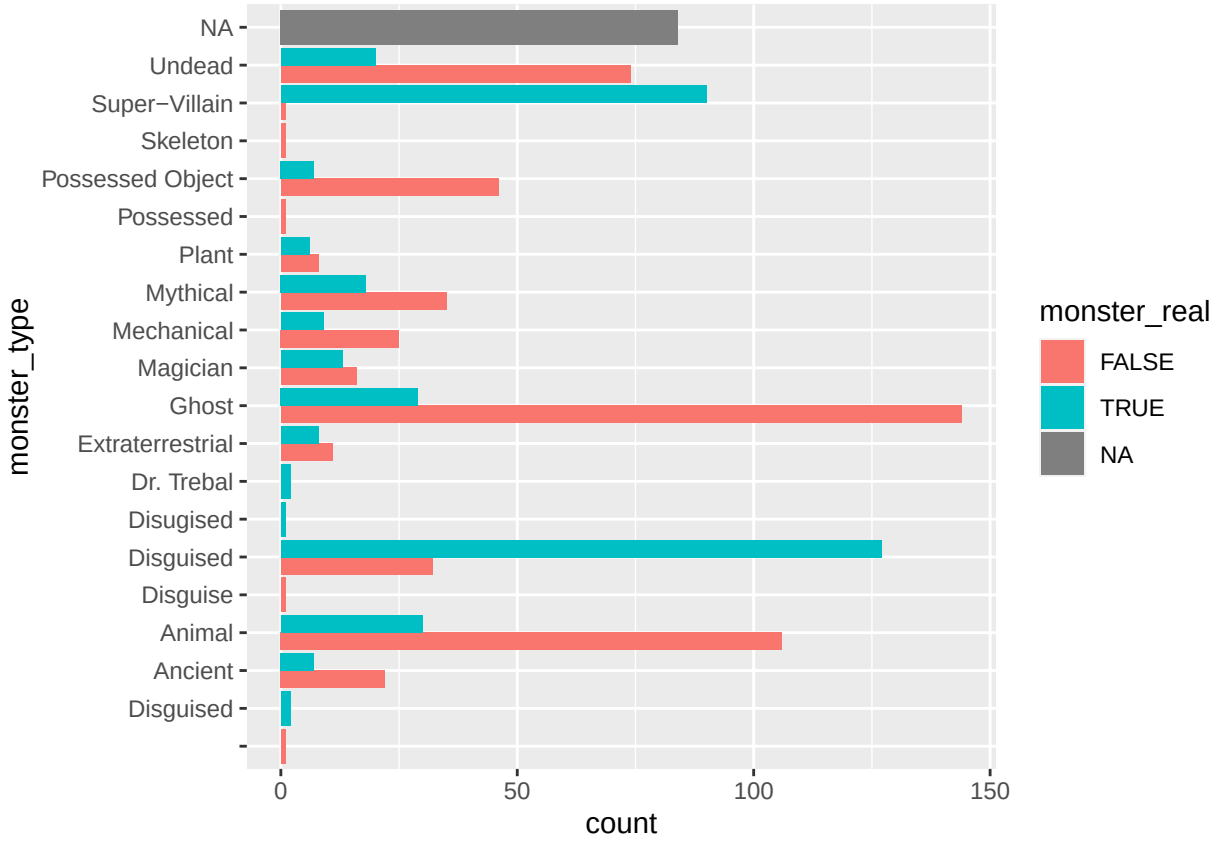
```
caught_total %>%
  ggplot(aes(x = name, y = value)) +
  geom_col() +
  theme_classic()
```



Plot 4: Per monster type, how many real vs unreal monsters were uncovered? Earlier, we saw how we can create a bar chart with ggplot. But what if we want to add another layer to the data, for example different categories? We can do so by adding an aesthetic "fill".

Note: Fill refers to the color of the chart, while "color" refers to its border. Therefore, for some chart types (e.g. point, line), we should specify a "color" aesthetic instead of "fill" for illustrating different categories. Try both on the code below!

```
scooby_doo_ep_longtype %>%
  ggplot(aes(y = monster_type,
             fill = monster_real)
         ) +
  geom_bar(position = "dodge")
```


monster_type
NA
Undead
Super−Villain
Skeleton
Possessed Object
Possessed
Plant
Mythical
Mechanical
Magician
Ghost
Extraterrestrial
Dr. Trebal
Disugised
Disguised
Disguise
Animal
Ancient
Disguised
0
50
100
150
count
monster_real
FALSE
TRUE
NA