Today, we'll finally talk about summarise! It's very similar to mutate, but instead of adding or altering a variable in a dataset, it aggregates your data, creating a new tibble with the columns containing your requested summary data. The number of rows will be equal to the number of groups from group_by (if you don't specify any groups, your tibble will have one row that summarizes your entire dataset).

These days, when I want descriptive statistics from a dataset, I generally use summarise, because I can specify the exact statistics I want in the exact order I want (for easy pasting of tables into a report or presentation).

Also, if you're not a fan of the UK spelling, summarize works exactly the same. The same is true of other R/tidyverse functions, like color versus colour.

Let's load the reads2019 dataset and start summarizing!

```
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------- tidyverse
1.3.0 --
```

```
##  ggplot2 3.2.1      purrr   0.3.3
##  tibble  2.1.3      dplyr   0.8.3
##  tidyr   1.0.0      stringr 1.4.0
##  readr   1.3.1      forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------
tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
reads2019 <- read_csv("~/Downloads/Blogging A to Z/SaraReads2019_allrated.csv",
                      col_names = TRUE)
```

```
## Parsed with column specification:
## cols(
##   Title = col_character(),
##   Pages = col_double(),
##   date_started = col_character(),
##   date_read = col_character(),
##   Book.ID = col_double(),
##   Author = col_character(),
##   AdditionalAuthors = col_character(),
##   AverageRating = col_double(),
##   OriginalPublicationYear = col_double(),
##   read_time = col_double(),
##   MyRating = col_double(),
##   Gender = col_double(),
##   Fiction = col_double(),
##   Childrens = col_double(),
##   Fantasy = col_double(),
##   SciFi = col_double(),
##   Mystery = col_double(),
##   SelfHelp = col_double()
## )
```

First, we could use summarise to give us some basic descriptives of the whole dataset. If we want to save the results to a tibble, we would give it a new name, or we could just have it display those results and not save them. Here's what happens when I request a summary without saving a new tibble.

```
reads2019 %>%
  summarise(AllPages = sum(Pages),
            AvgLength = mean(Pages),
            AvgRating = mean(MyRating),
            AvgReadTime = mean(read_time),
            ShortRT = min(read_time),
            LongRT = max(read_time),
            TotalAuthors = n_distinct(Author))
```

```
## # A tibble: 1 x 7
##   AllPages AvgLength AvgRating AvgReadTime ShortRT LongRT TotalAuthors
##
## 1    29696      341.      4.14        3.92       0     25           42
```

Now, let's create a summary where we do save it as a tibble. And let's have it create some groups for us. In the dataset, I coded author gender, with female authors coded as 1, so I can find out how many women writers are represented in a group by summing that variable. I also want to fill in a few missing publication dates, which seems to happen for Kindle version of books or books by small publishers. This will let me find out my newest and oldest books in each group; I just arrange by publication year, then request last and first, respectively. Two books were published in 2019, so I'll replace the others based on title, then have R give the remaining NAs a year of 2019.

```
reads2019 %>%
  filter(is.na(OriginalPublicationYear)) %>%
  select(Title)
```

```
## # A tibble: 5 x 1
##   Title
##
## 1 Empath: A Complete Guide for Developing Your Gift and Finding Your Sense of
S…
## 2 Perilous Pottery (Cozy Corgi Mysteries, #11)
## 3 Precarious Pasta (Cozy Corgi Mysteries, #14)
## 4 Summerdale
## 5 Swarm Theory
```

```
reads2019 <- reads2019 %>%
  mutate(OriginalPublicationYear = replace(OriginalPublicationYear,
                                           Title == "Empath: A Complete Guide
for Developing Your Gift and Finding Your Sense of Self", 2017),
         OriginalPublicationYear = replace(OriginalPublicationYear,
                                           Title == "Summerdale", 2018),
         OriginalPublicationYear = replace(OriginalPublicationYear,
                                           Title == "Swarm Theory", 2016),
         OriginalPublicationYear = replace_na(OriginalPublicationYear, 2019))

genrestats <- reads2019 %>%
  filter(Fiction == 1) %>%
  arrange(OriginalPublicationYear) %>%
  group_by(Childrens, Fantasy, SciFi, Mystery) %>%
  summarise(Books = n(),
            WomenAuthors = sum(Gender),
            AvgLength = mean(Pages),
            AvgRating = mean(MyRating),
            NewestBook = last(OriginalPublicationYear),
            OldestBook = first(OriginalPublicationYear))
```

Now let's turn this summary into a nicer, labeled table.

```
genrestats <- genrestats %>%
  bind_cols(Genre = c("General Fiction",
                  "Mystery",
                  "Science Fiction",
                  "Fantasy",
                  "Fantasy SciFi",
                  "Children's Fiction",
                  "Children's Fantasy")) %>%
  ungroup() %>%
  select(Genre, everything(), -Childrens, -Fantasy, -SciFi, -Mystery)

library(expss)

##
## Attaching package: 'expss'

## The following objects are masked from 'package:stringr':
##
##     fixed, regex

## The following objects are masked from 'package:dplyr':
##
##     between, compute, contains, first, last, na_if, recode, vars

## The following objects are masked from 'package:purrr':
##
##     keep, modify, modify_if, transpose

## The following objects are masked from 'package:tidyr':
##
##     contains, nest

## The following object is masked from 'package:ggplot2':
##
##     vars

as.etable(genrestats, rownames_as_row_labels = NULL)
```

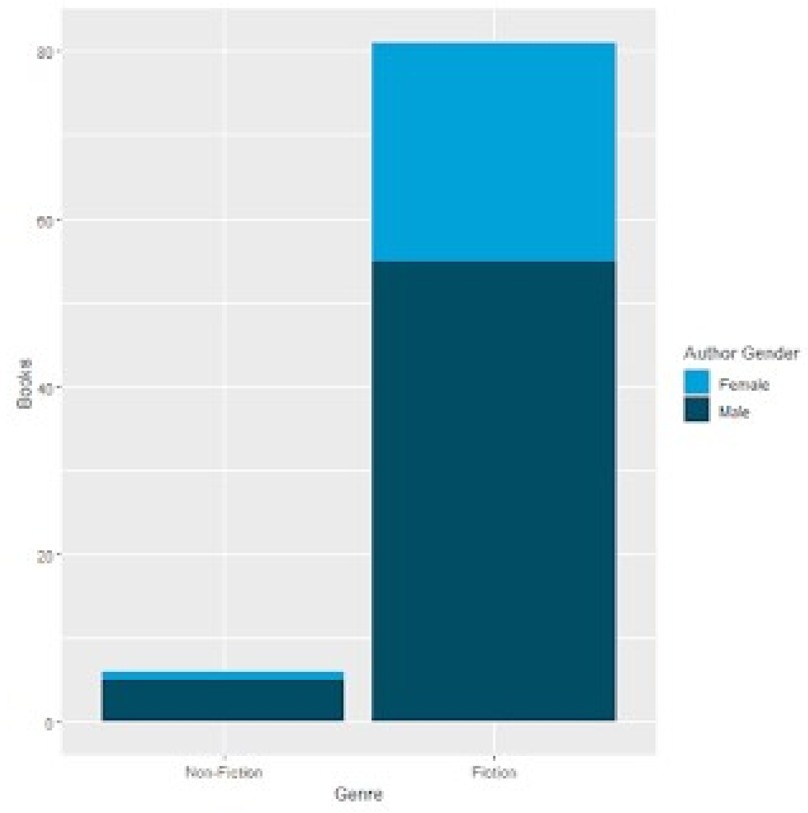
Genre	Books	WomenAuthors	AvgLength	AvgRating	NewestBook	OldestBook
General Fiction	15	10	320.1	4.1	2019	1941
Mystery	9	8	316.3	3.8	2019	1950
Science Fiction	19	4	361.4	4.4	2019	1959
Fantasy	19	3	426.3	4.2	2019	1981
Fantasy SciFi	2	0	687.0	4.5	2009	2006
Children’s Fiction	1	0	181.0	4.0	2016	2016
Children’s Fantasy	16	1	250.6	4.2	2008	1900

I could have used other base R functions in my summary as well – such as sd, median, min, max, and so on. You can also summarize a dataset and create a plot of that summary in the same code.

```
library(ggthemes)

## Warning: package 'ggthemes' was built under R version 3.6.3

reads2019 %>%
  mutate(Gender = factor(Gender, levels = c(0,1),
                         labels = c("Male",
                                    "Female")),
         Fiction = factor(Fiction, levels = c(0,1),
                          labels = c("Non-Fiction",
                                     "Fiction"),
                                ordered = TRUE)) %>%
  group_by(Gender, Fiction) %>%
  summarise(Books = n()) %>%
  ggplot(aes(Fiction, Books)) +
  geom_col(aes(fill = reorder(Gender, desc(Gender)))) +
  scale_fill_economist() +
  xlab("Genre") +
  labs(fill = "Author Gender")
```


Books
Non-Fiction
Fiction
Genre
Author Gender
Female
Male