PR12 - Data Wrangling

김서준

2023-12-01

1. Data Wrangling with `tidyverse`

- Data Wrangling이란, 분석을 진행하기 위해 날것(raw)의 데이터를 분석에 적합한 형태로 정형화시키는 작업이다.
- R에서는 `tidyverse` 라는 패키지를 구성하고 있어서, 일관성있고 쉬운 작업을 가능하게 한다.

```
#install.packages("tidyverse")
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────────────── tidy
verse 2.0.0 ──
## ✓ dplyr     1.1.3     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.4.4     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.0
## ✓ 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 beco
me errors
```

2. `tidyr`

- `tidyr` 은 Hadley wickham이 만든 데이터의 포맷을 변경하기 위한 패키지

`tidyr` 의 주요함수

함수	설명
`gather()`	데이터를 wide에서 long 포멧으로 변경
`spread()`	데이터를 long에서 wide 포맷으로 변경
`separate()`	단일 열(column)을 복수 열들로 분리
`unite()`	복수 열(column)들을 단일 열로 결합

`tidyr` 실습 데이터: `cases` in `EDAWR`

- Dataset to support the Expert Data Analysis with R: `EDAWR`

```
library(devtools)
```

```
## 필요한 패키지를 로딩중입니다: usethis
```

```
#devtools::install_github("rstudio/EDAWR", force = TRUE)
library(EDAWR)
```

```
##
## 다음의 패키지를 부착합니다: 'EDAWR'
```

```
## The following object is masked from 'package:dplyr':
##
##     storms
```

```
## The following objects are masked from 'package:tidyr':
##
##     population, who
```

```
head(cases)
```

```
##   country  2011  2012  2013
## 1      FR  7000  6900  7000
## 2      DE  5800  6000  6200
## 3      US 15000 14000 13000
```

```
head(pollution)
```

```
##       city  size amount
## 1 New York large     23
## 2 New York small     14
## 3   London large     22
## 4   London small     16
## 5  Beijing large    121
## 6  Beijing small     56
```

```
head(storms)
```

```
##     storm wind pressure       date
## 1 Alberto  110     1007 2000-08-03
## 2    Alex   45     1009 1998-07-27
## 3 Allison   65     1005 1995-06-03
## 4     Ana   40     1013 1997-06-30
## 5  Arlene   50     1010 1999-06-11
## 6  Arthur   45     1010 1996-06-17
```

2.1. gather() 함수

- wide 포맷의 데이터를 원하는 조건에 맞게 long 포맷으로 변환하는 함수
- gather(데이터, 키(key), 값(value), ...)
    - 키(key): **새로운데이터**에 변수로 표시될 열이름

- 값(value): **새로운데이터**에 변수의 값이 표시될 열이름 +...: **원데이터**로 부터 모으기(gather)가 진행될 열들의 범위

```
gather(cases, year, n, 2:4)
```

```
##   country year     n
## 1      FR 2011  7000
## 2      DE 2011  5800
## 3      US 2011 15000
## 4      FR 2012  6900
## 5      DE 2012  6000
## 6      US 2012 14000
## 7      FR 2013  7000
## 8      DE 2013  6200
## 9      US 2013 13000
```

2.2. spread() 함수

- long 포맷의 데이터를 원하는 조건에 맞게 long 포맷으로 변환하는 함수
- `separate(데이터, 키(key), 값(value), ~)`
  - 키(key): 복수개의 열로 spread될 기존 long 포맷의 열이름
  - 값(value): 복수개의 열로 spread 되어 값이 될 기존 long 포맷의 열이름

```
spread(pollution, size, amount)
```

```
##       city large small
## 1  Beijing   121    56
## 2   London    22    16
## 3 New York    23    14
```

2.2. separate() 함수

- 하나의 열을 특정 조건에 따라 여러개의 열로 나누어 주는 함수입니다.
- `separate(data, col, into, sep, ~)`
  - col: 조건에 따른 분할을 진행할 열이름
  - into: 분할될 결과가 저장될 각 열들의 이름
  - sep: 분할 조건

```
storms2 <- separate(storms, date, c("year", "month", "day"), sep = "-")
storms2
```

```
## # A tibble: 6 × 6
##   storm    wind pressure year  month day
##   <chr>   <int>    <int> <chr> <chr> <chr>
## 1 Alberto   110     1007 2000  08    03
## 2 Alex       45     1009 1998  07    27
## 3 Allison    65     1005 1995  06    03
## 4 Ana        40     1013 1997  06    30
## 5 Arlene     50     1010 1999  06    11
## 6 Arthur     45     1010 1996  06    17
```

2.4. unite() 함수

- 여러개로 나누어진 열을 특정 조건에 따라 결합해주는 함수입니다.
- `unite(data, col, ..., sep)`
  - `col` : 조건에 따라 결합된 결과가 저장될 열이름
  - `...` : 합쳐질 열이름들
  - `sep` : 결합시 구분자

```
unite(storms2, "date", year, month, day, sep = "-")
```

```
## # A tibble: 6 × 4
##   storm    wind pressure date
##   <chr>   <int>    <int> <chr>
## 1 Alberto   110     1007 2000-08-03
## 2 Alex       45     1009 1998-07-27
## 3 Allison    65     1005 1995-06-03
## 4 Ana        40     1013 1997-06-30
## 5 Arlene     50     1010 1999-06-11
## 6 Arthur     45     1010 1996-06-17
```

3. dplyr

- `dplyr` 은 Hadley Wickham이 만든 데이터 핸들링을 위한 패키지
- `dplyr` 은 C++로 작성되어 기존 데이터핸들링 패키지보다 빠른 데이터조작이 가능
- 각종 데이터베이스 지원(**MYSQL**, **postgreSQL**, **SQLite**, **BigQuery**)
- R의 기본문법과 프로그래밍능력만으로도 데이터의 조작이 가능하지만, `dplyr` 패키지를 활용하면 통일된 문법양식으로 데이터조작이 가능함
- 체인연산자를 지원합으로(`%>%`) 앞부분의 연산결과를 뒤에 오는 함수의 입력값으로 사용할 수 있음

dplyr 의 주요함수

dplyr 실습데이터: nycflights13

- 미국 휴스턴에서 출발하는 모든 비행기의 이착륙기록

```
#install.packages("nycflights13")
library(nycflights13)
library(dplyr)
head(flights)
```

```
## # A tibble: 6 × 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
## 1  2013     1     1      517            515         2      830            819
## 2  2013     1     1      533            529         4      850            830
## 3  2013     1     1      542            540         2      923            850
## 4  2013     1     1      544            545        -1     1004           1022
## 5  2013     1     1      554            600        -6      812            837
## 6  2013     1     1      554            558        -4      740            728
## # ℹ 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

3.1 fliter() 함수

- 데이터에서 원하는 조건에 따라 행을 추출하는 함수
- fliter(데이터, 조건1 | 조건2) : 조건1 또는 조건2 둘중 한가지를 충족하는 데이터 추출
- fliter(데이터, 조건1 & 조건2) : 조건1과 조건2 모두 충족하는 데이터 추출
- 조건을 작성할때 쉼표',‘는 AND,'|'는 OR와 같음

```
filter(flights, month == 1 | day == 1) #37198row
```

```
## # A tibble: 37,198 × 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ℹ 37,188 more rows
## # ℹ 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
filter(flights, month == 1, day == 1) #842row
```

```
## # A tibble: 842 × 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ℹ 832 more rows
## # ℹ 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
filter(flights, month == 1, day == 1, year == 2013) #832
```


```
## # A tibble: 842 × 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ℹ 832 more rows
## # ℹ 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```


3.2 ’arrange()` 함수

- 데이터를 원하는 조건에 따라 정렬해주는 함수
- arrange(데이터, 정렬기준컬럼1, 정렬기준컬럼2, 정렬기준컬럼3)
- 내림차순으로 정렬시 desc함수 사용 : arrange(데이터, desc(정렬기준컬럼1))

```
arrange(flights, year, month, day) #ArrDelay, Month, Year 순으로 정렬
```

```
## # A tibble: 336,776 × 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ℹ 336,766 more rows
## # ℹ 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
arrange(flights, desc(month)) #Month컬럼기준으로 내림차순으로 정렬
```

```
## # A tibble: 336,776 × 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013    12     1       13           2359        14      446            445
##  2  2013    12     1       17           2359        18      443            437
##  3  2013    12     1      453            500        -7      636            651
##  4  2013    12     1      520            515         5      749            808
##  5  2013    12     1      536            540        -4      845            850
##  6  2013    12     1      540            550       -10     1005           1027
##  7  2013    12     1      541            545        -4      734            755
##  8  2013    12     1      546            545         1      826            835
##  9  2013    12     1      549            600       -11      648            659
## 10  2013    12     1      550            600       -10      825            854
## # ℹ 336,766 more rows
## # ℹ 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

3.3 select() 함수

- select함수는 원하는 열(column)을 추출
- select(데이터, 컬럼1, 컬럼2, 컬럼3)
- select(데이터, 컬럼1:컬럼3)
- 컬럼명을 변경할수 있음

```
select(flights, year, month, day)
```

```
## # A tibble: 336,776 × 3
##     year month   day
##    <int> <int> <int>
##  1  2013     1     1
##  2  2013     1     1
##  3  2013     1     1
##  4  2013     1     1
##  5  2013     1     1
##  6  2013     1     1
##  7  2013     1     1
##  8  2013     1     1
##  9  2013     1     1
## 10  2013     1     1
## # ℹ 336,766 more rows
```

```
select(flights, year:day)
```

```
## # A tibble: 336,776 × 3
##     year month   day
##    <int> <int> <int>
##  1  2013     1     1
##  2  2013     1     1
##  3  2013     1     1
##  4  2013     1     1
##  5  2013     1     1
##  6  2013     1     1
##  7  2013     1     1
##  8  2013     1     1
##  9  2013     1     1
## 10  2013     1     1
## # ℹ 336,766 more rows
```

```
select(flights, -(year:day))
```

```
## # A tibble: 336,776 × 16
##    dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
##       <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>
##  1      517            515         2      830            819        11 UA
##  2      533            529         4      850            830        20 UA
##  3      542            540         2      923            850        33 AA
##  4      544            545        -1     1004           1022       -18 B6
##  5      554            600        -6      812            837       -25 DL
##  6      554            558        -4      740            728        12 UA
##  7      555            600        -5      913            854        19 B6
##  8      557            600        -3      709            723       -14 EV
##  9      557            600        -3      838            846        -8 B6
## 10      558            600        -2      753            745         8 AA
## # ℹ 336,766 more rows
## # ℹ 9 more variables: flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

3.4 distinct() 함수

- 중복항목을 제외한 데이터를 확인 할 수 있음(unique함수와 동일)
- distinct(데이터, 컬럼명)

```
distinct(select(flights, tailnum))
```

```
## # A tibble: 4,044 × 1
##    tailnum
##    <chr>
##  1 N14228
##  2 N24211
##  3 N619AA
##  4 N804JB
##  5 N668DN
##  6 N39463
##  7 N516JB
##  8 N829AS
##  9 N593JB
## 10 N3ALAA
## # ℹ 4,034 more rows
```

```
distinct(select(flights, origin, dest))
```

```
## # A tibble: 224 × 2
##    origin dest
##    <chr>  <chr>
##  1 EWR    IAH
##  2 LGA    IAH
##  3 JFK    MIA
##  4 JFK    BQN
##  5 LGA    ATL
##  6 EWR    ORD
##  7 EWR    FLL
##  8 LGA    IAD
##  9 JFK    MCO
## 10 LGA    ORD
## # ℹ 214 more rows
```

3.5 mutate() 함수

- 기존 데이터 프레임에 새로운 열을 추가해줌
- 데이터프레임 내의 변수들을 활용해 새로운 변수를 만들매 효과적임
- 새로 생성한 변수를 해당 함수내에서 바로 활용이 가능

```
#arr_delay - dep_delay값으로 gain컬럼 추가
mutate(flights, gain = arr_delay - dep_delay)
```

```
## # A tibble: 336,776 × 20
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ℹ 336,766 more rows
## # ℹ 12 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>, gain <dbl>
```

```
#gain컬럼을 만드는 동시에 gain컬럼을 이용해 다른 변수를 생성가능
mutate(flights,
      gain = arr_delay - dep_delay,
      gain_per_hour = gain/(air_time/60))
```

```
## # A tibble: 336,776 × 21
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ℹ 336,766 more rows
## # ℹ 13 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>, gain <dbl>, gain_per_hour <dbl>
```

3.6 summarise() 함수

- mean(), sd(), var(), median()함수를 활용해 기술통계량을 확인
- 결과를 데이터프레임으로 반환함

```
summarise(flights, delay = mean(dep_delay, na.rm = TRUE))
```

```
## # A tibble: 1 × 1
##   delay
##   <dbl>
## 1  12.6
```

3.7 group_by() 함수

- 변수의 레벨에 따라 자료를 그룹화해줌
- 그룹에 따른 수치자료를 산출하고 싶을때 편리함
- summarize함수와 함께 사용시 aggregate함수와 같은 기능
- ex)직급에 따른 평균 연봉과 사용가능한 연차일수(휴가)를 구하고 싶을때

```
#비행기별로 그룹만들기
by_tailnum <- group_by(flights, tailnum) #비행기별로 그룹만들기
#비행기별 비행회수, 비행거리평균, 연착시간평균 산출
delay <- summarise(by_tailnum, count = n(), dist = mean(distance, na.rm = TRUE),
delay = mean(arr_delay, na.rm = TRUE))
#회수가 20회이상 , 거리가 2000이하인 비행기만 추출
delay <- filter(delay, count > 20, dist < 2000)
```

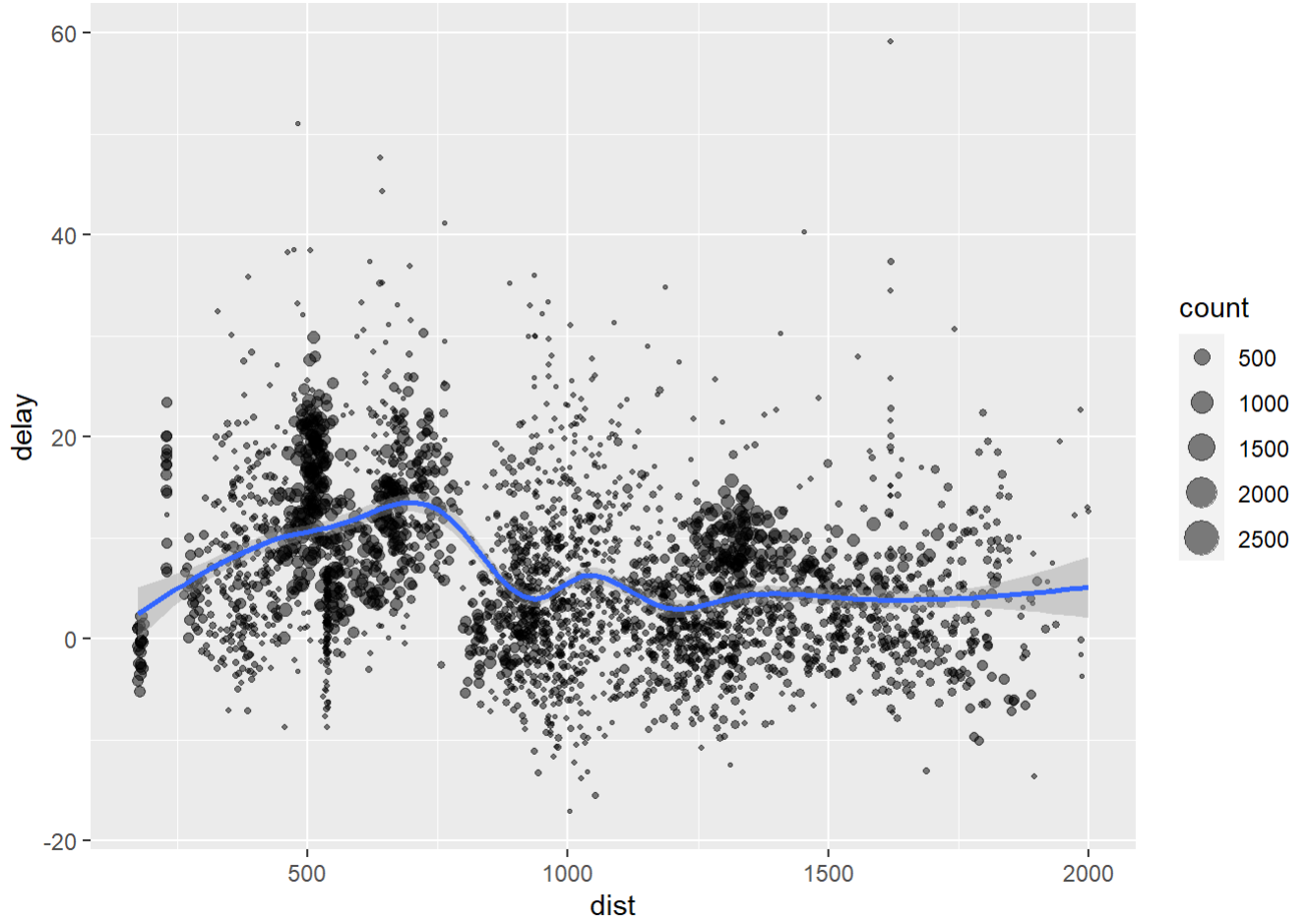
- 위에서 만든 delay데이터로 시각화

```
library(ggplot2)
ggplot(delay, aes(dist, delay)) +
geom_point(aes(size = count), alpha = 1/2) +
geom_smooth() +
scale_size_area()
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 1 rows containing missing values (`geom_point()`).
```



3.8. `join()` 함수 - join(x, y) 또는 join(x, y, by="기준열") 형태 - 조인의 기준이 되는 단일 컬럼이 존재하는 경우 별도 by인수를 지정하지 않아도됨 - 단일 칼럼이 존재하지 않는 경우 by=c(기준열1 = 기준열2)와 같이 설정을 해주어야 함. - 조인의 기준이 되는 컬럼이 여러개이거나, 여러가지 컬럼을 동시에 활용해야하는 경우 by인수를 사용

```
#install.packages("readr")
library(readr)
#join 실습 데이터 생성
superheroes <- "
name, alignment, gender, publisher
Magneto, bad, male, Marvel
Storm, good, female, Marvel
Mystique, bad, female, Marvel
Batman, good, male, DC
Joker, bad, male, DC
Catwoman, bad, female, DC
Hellboy, good, male, Dark Horse Comics
"
publishers <- "
publisher, yr_founded
DC, 1934
Marvel, 1939
Image, 1992
"
superheroes <- read_csv(superheroes, trim_ws = TRUE, skip = 1)
```

```
## Rows: 7 Columns: 4
## ── Column specification ────────────────────────────────────────────────────────
──────────────────────────────
## Delimiter: ","
## chr (4): name, alignment, gender, publisher
##
## 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.
```

```
publishers <- read_csv(publishers, trim_ws = TRUE, skip = 1)
```

```
## Rows: 3 Columns: 2
## ── Column specification ────────────────────────────────────────────────────────
──────────────────────────────
## Delimiter: ","
## chr (1): publisher
## dbl (1): yr_founded
##
## 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.
```

```
getwd()
```

```
## [1] "C:/UnivStudy/UnivLectures/23-2/R-programming/Works/PR12()"
```

- inner_join, left_join, full_join, anti_join, semi_join 각각의 출력값확인하기

```
inner_join(superheroes, publishers) # X, Y의 교집합
```

```
## Joining with `by = join_by(publisher)`
```

```
## # A tibble: 6 × 5
##   name     alignment gender publisher yr_founded
##   <chr>    <chr>     <chr>  <chr>          <dbl>
## 1 Magneto  bad       male   Marvel          1939
## 2 Storm    good      female Marvel          1939
## 3 Mystique bad       female Marvel          1939
## 4 Batman   good      male   DC              1934
## 5 Joker    bad       male   DC              1934
## 6 Catwoman bad       female DC              1934
```

```
left_join(superheroes, publishers) # X기준(왼쪽)으로 머징
```

```
## Joining with `by = join_by(publisher)`
```

```
## # A tibble: 7 × 5
##   name     alignment gender publisher         yr_founded
##   <chr>    <chr>     <chr>  <chr>                   <dbl>
## 1 Magneto  bad       male   Marvel                   1939
## 2 Storm    good      female Marvel                   1939
## 3 Mystique bad       female Marvel                   1939
## 4 Batman   good      male   DC                       1934
## 5 Joker    bad       male   DC                       1934
## 6 Catwoman bad       female DC                       1934
## 7 Hellboy  good      male   Dark Horse Comics          NA
```

```
full_join(superheroes, publishers) #X, Y의 합집합
```

```
## Joining with `by = join_by(publisher)`
```

```
## # A tibble: 8 × 5
##   name     alignment gender publisher         yr_founded
##   <chr>    <chr>     <chr>  <chr>                   <dbl>
## 1 Magneto  bad       male   Marvel                   1939
## 2 Storm    good      female Marvel                   1939
## 3 Mystique bad       female Marvel                   1939
## 4 Batman   good      male   DC                       1934
## 5 Joker    bad       male   DC                       1934
## 6 Catwoman bad       female DC                       1934
## 7 Hellboy  good      male   Dark Horse Comics          NA
## 8 <NA>     <NA>      <NA>   Image                    1992
```

```
anti_join(superheroes, publishers) #X의 컬럼만 유지하며 머징
```

```
## Joining with `by = join_by(publisher)`
```

```
## # A tibble: 1 × 4
##   name    alignment gender publisher
##   <chr>   <chr>     <chr>  <chr>
## 1 Hellboy good      male   Dark Horse Comics
```

```
semi_join(superheroes, publishers) #Y의 여집합
```

```
## Joining with `by = join_by(publisher)`
```

```
## # A tibble: 6 × 4
##   name     alignment gender publisher
##   <chr>    <chr>     <chr>  <chr>
## 1 Magneto  bad       male   Marvel
## 2 Storm    good      female Marvel
## 3 Mystique bad       female Marvel
## 4 Batman   good      male   DC
## 5 Joker    bad       male   DC
## 6 Catwoman bad       female DC
```

4. magrittr

- magrittr 패키지는 면산자(operator)들의 집합들을 제공합니다.
- 데이터 연산을 왼쪽에서 오른쪽 순서로 구조화,
- nested 함수 호출을 피함,
- 지역 변수 및 함수의 정의의 필요성을 최소화,
- 연산 순서 내에서 어디서나 추가 step을 만을 수 있음
- f(x)를 x %>% f()로 대체할 수 있음
- 이 연산자가 main operator(chaining)인데 해당 기능이 의미 없이 보이시겠지만 여러가지 기능을 결합할 때 그 이점이 더욱 명확해집니다.
- dplyr을 불러오면 자동으로 불러와지게 됩니다.

4.1 main operator (chaining; %>%)

- 여러단계의 함수나 연산을 연결하여 한번에 수행할 때 사용
- 앞의 함수의 결과는 바로 뒤에오는 함수의 입력값이 됨
- 데이터를 여러객체에 할당하지 않아도 되기매문에 메모리 관리에 유리함

체인연산 사용하지 않을때

```
a1 <- group_by(flights, year, month, day)
a2 <- select(a1, year:day, arr_delay)
a3 <- summarise(a2, arr = mean(arr_delay, na.rm = TRUE))
```

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

```
a4 <- filter(a3, arr > 30)
a4
```

```
## # A tibble: 42 × 4
## # Groups:   year, month [11]
##     year month   day   arr
##    <int> <int> <int> <dbl>
##  1  2013     1    16  34.2
##  2  2013     1    31  32.6
##  3  2013     2    11  36.3
##  4  2013     2    27  31.3
##  5  2013     3     8  85.9
##  6  2013     3    18  41.3
##  7  2013     4    10  38.4
##  8  2013     4    12  36.0
##  9  2013     4    18  36.0
## 10  2013     4    19  47.9
## # ℹ 32 more rows
```

체인연산 사용했을때

```
flights %>%
  group_by(year, month, day) %>%
  select(arr_delay) %>%
  summarise(
    arr = mean(arr_delay, na.rm = TRUE)
    ) %>%
  filter(arr > 30)
```

```
## Adding missing grouping variables: `year`, `month`, `day`
## `summarise()` has grouped output by 'year', 'month'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 42 × 4
## # Groups:   year, month [11]
##     year month   day   arr
##    <int> <int> <int> <dbl>
##  1  2013     1    16  34.2
##  2  2013     1    31  32.6
##  3  2013     2    11  36.3
##  4  2013     2    27  31.3
##  5  2013     3     8  85.9
##  6  2013     3    18  41.3
##  7  2013     4    10  38.4
##  8  2013     4    12  36.0
##  9  2013     4    18  36.0
## 10  2013     4    19  47.9
## # ℹ 32 more rows
```

4.2. .의 역할

- “.”의 역할에 대해서 알아봅시다.
- 일반적으로 '%>%`연산자만 사용하시게 되면 제일 첫 인수에 자동으로 배정이 됩니다.

```
head(iris, 3)
```

```
##   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
```

```
iris %>% head(3) # = head(., 3)
```

```
##   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
```

- 데이터를 넘겨줘야 할 인수의 위치가 첫번패가 마닐 경우 다음과 같은 에러를 확인할 수 있음
- 'gsub() 는 찾아 바꾸는 함수로써, 사용방법은 'gsub(찾을문자나 숫자,바꿀문자나 숫자, 데이터)

```
a <- c("bannananana", "an apple")
gsub("n","l",a)
```

```
## [1] "ballalalala" "al apple"
```

```
a %>% gsub("n", "l")
```

```
## Warning in gsub(., "n", "l"): 인자 'pattern'는 반드시 길이가 1 보다 커야 하고,
## 오로지 첫번째 요소만이 사용될 것입니다
```

```
## [1] "l"
```

- 이러한 상황에서, “.”을 원하는 위치에 넣어주시면 해당 위치에 데이터가 넘어가게 됨
- “.”은 magrittr나 dplyr에만 속해 있는 것이 마니라 R의 base에 정해진 규칙으로 `.~ cyl` 의 사용법과 같습니다.

```
gsub("n","1",a)
```

```
## [1] "ba11a1a1a1a" "a1 apple"
```

```
a %>% gsub("n","1",.)
```

```
## [1] "ba11a1a1a1a" "a1 apple"
```

4.3. chaining 예제

4.3.1. mtcars aggregate

```
library(magrittr)
```

```
##
## 다음의 패키지를 부착합니다: 'magrittr'
```

```
## The following object is masked from 'package:purrr':
##
##     set_names
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
car_data <-
  mtcars %>%
  subset(hp > 100) %>%
  aggregate(. ~ cyl, data = ., FUN = . %>% mean %>% round(2)) %>%
  transform(kpl = mpg %>% multiply_by(0.4251)) %>% #4
  print
```

```
##   cyl   mpg   disp     hp drat   wt  qsec   vs   am gear carb       kpl
## 1   4 25.90 108.05 111.00 3.94 2.15 17.75 1.00 1.00 4.50 2.00 11.010090
## 2   6 19.74 183.31 122.29 3.59 3.12 17.98 0.57 0.43 3.86 3.43  8.391474
## 3   8 15.10 353.10 209.21 3.23 4.00 16.77 0.00 0.14 3.29 3.50  6.419010
```

- 예제 해석
  - mtcars 데이터셋을 (#1)
  - hp를 기준으로 100보다 큰 데이터만 추출한 후(#2)
  - cy1를 기준으로 각 변수들의 평균을 구한 다음에 소수점 둘째 자리까지 반올림을 한 후(#3)
  - kpl(kilometer per liter) 열을 만들어 mpg*0.4251을 수행하고(#4)
  - 만들어진 데이터를 출력(#5)과 동시에 car_data에 할당하는 과정입니다.
- 체인연산없이 실행

```
car_data <-
  transform(aggregate(. ~ cyl,
                      data = subset(mtcars, hp > 100),
                      FUN = function(x) round(mean(x), 2)),
            kpl = mpg*0.4251)
car_data
```

```
##   cyl   mpg   disp     hp drat   wt  qsec   vs   am gear carb       kpl
## 1   4 25.90 108.05 111.00 3.94 2.15 17.75 1.00 1.00 4.50 2.00 11.010090
## 2   6 19.74 183.31 122.29 3.59 3.12 17.98 0.57 0.43 3.86 3.43  8.391474
## 3   8 15.10 353.10 209.21 3.23 4.00 16.77 0.00 0.14 3.29 3.50  6.419010
```

4.3.2. 예제 변환

- ** 2.1 .** 예제 tidyr의 함수들도 chaining 연산과 함께 사용하면 직관적으로 사용할 수 있습니다.

```
cases %>% gather (Year, n, 2:4)
```

```
##   country Year     n
## 1      FR 2011  7000
## 2      DE 2011  5800
## 3      US 2011 15000
## 4      FR 2012  6900
## 5      DE 2012  6000
## 6      US 2012 14000
## 7      FR 2013  7000
## 8      DE 2013  6200
## 9      US 2013 13000
```

- ** 3.7.** 예제 dplyr에서도 함께 쓰여 데이터를 그룹화하고 수치를 요약하는 등의 작업에 특화되어 있습니다.

```
#비행기별 비행회수, 비행거리평균, 면착시간평균 산출
flights %>%
  group_by(tailnum) %>%
  summarise(
    count = n(),
    dist = mean(distance, na.rm = TRUE),
    delay = mean(arr_delay, na.rm = TRUE)
    )
```

```
## # A tibble: 4,044 × 4
##    tailnum count  dist  delay
##    <chr>   <int> <dbl>  <dbl>
##  1 D942DN      4  854. 31.5
##  2 N0EGMQ    371  676.  9.98
##  3 N10156    153  758. 12.7
##  4 N102UW     48  536.  2.94
##  5 N103US     46  535. -6.93
##  6 N104UW     47  535.  1.80
##  7 N10575    289  520. 20.7
##  8 N105UW     45  525. -0.267
##  9 N107US     41  529. -5.73
## 10 N108UW     60  534. -1.25
## # ℹ 4,034 more rows
```

5. tibble

- 'tibble'은 tidyverse 생태계에서 데이터 프레임을 대신하여 편리한 기능들 및 동작을 포함한 자료형입니다.
- factor 자동 변환
- 일부값만 출력
- 출력시 자료형 명시
- 데이터 프레임과 비교 | 작업유형 | 데이터프레임 명령어 | 티블 명령어
  ——————————————— | 생성 | 'data.frame()' | 'data_frame()',
  'tibble()`, `"tribble() I | 강제변환 (Coercion) | 'as.data.frame() | 'as_tibble() I | 데이터 불러오기 |
  `read .* ()` | `read_delim()`, `read_csv()`, `read_csv2()`, `read_tsv()` | ## 5.1. 'tibble` 생성

tibble()

```
tibble(
  x = 1:5,
  y = 1,
  z = x^2 + y
)
```

```
## # A tibble: 5 × 3
##       x     y     z
##   <int> <dbl> <dbl>
## 1     1     1     2
## 2     2     1     5
## 3     3     1    10
## 4     4     1    17
## 5     5     1    26
```

tribble()

- 코드 단계에서 데이터를 입력받도록 하기 위해 존재하는 함수입니다.

```
tribble(
  ~x, ~y, ~Z,
# -- | -- | ---
  "a", 2, 3.6,
  "b", 1, 8.5
)
```

```
## # A tibble: 2 × 3
##   x         y     Z
##   <chr> <dbl> <dbl>
## 1 a         2   3.6
## 2 b         1   8.5
```

as_tibble()

- 기존의 데이터 프레임을 tibble 형으로 전환 합니다.

```
iris_tibble <- as_tibble(iris) # 기존의 데이터 프레임을 tibble로

print(class(iris)) # 기존 데이터 프레임 클래스
```

```
## [1] "data.frame"
```

```
print(class(iris_tibble)) # 새롭게 정의된 tibble 클래스 (데이터 프레임도)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
head(iris_tibble)
```

```
## # A tibble: 6 × 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>
## 1          5.1         3.5          1.4         0.2 setosa
## 2          4.9         3            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           3.6          1.4         0.2 setosa
## 6          5.4         3.9          1.7         0.4 setosa
```

5.2. 데이터 불러오기

- 데이터를 읽어올 때, dataframe이 아닌 tibble로 읽어오기 위해서, 동일한 'tidyverse' 생태계에 속한 'readr' 패키지의 함수들을 필요로합니다. 이미 'tidyverse`를 library하였으므로 바로 이용 가능합니다.

read_csv(file)

- 기존의 데이터 불러오기와 동일하게 파일명을 지정하여 해당 파일을 tibble로 읽어올 수 있습니다.

```
read_csv("traffic.csv")
```

```
## New names:
## Rows: 500 Columns: 12
## ── Column specification
## ────────────────────────────────────────────────────────────────────────
────────────
## Delimiter: "," chr (4): rpt.id, rpt.contents, info.tp, info.tit dbl (5): ...1,
## start.pos.x, start.pos.y, end.pos.x, end.pos.y date (2): occ.dtime, end.dtime
## time (1): reg.dtime
## 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.
## • `` -> `...1`
```

```
## # A tibble: 500 × 12
##     ...1 rpt.id   rpt.contents  info.tp info.tit occ.dtime  reg.dtime end.dtime
##    <dbl> <chr>    <chr>         <chr>   <chr>    <date>     <time>    <date>
##  1     1 01149328 서부간선도로… A4      단순정보 2014-06-15 17:17     2014-06-15
##  2     2 01149327 동부간선도로… A4      단순정보 2014-06-15 17:16     2014-06-15
##  3     3 01149326 북부간선도로… A4      단순정보 2014-06-15 17:16     2014-06-15
##  4     4 01149325 올림픽대로 …  A4      단순정보 2014-06-15 17:15     2014-06-15
##  5     5 01149324 강변북로 (일… A1      단순사고 2014-06-15 17:15     2014-06-15
##  6     6 01149323 내부순환로 …  A4      단순정보 2014-06-15 17:14     2014-06-15
##  7     7 01149322 평택-시흥간…  A4      단순정보 2014-06-15 17:14     2014-06-15
##  8     8 01149321 서울-춘천간…  A4      단순정보 2014-06-15 17:13     2014-06-15
##  9     9 01149320 천안-논산간…  A4      단순정보 2014-06-15 17:13     2014-06-15
## 10    10 01149319 영동고속도로… A4      단순정보 2014-06-15 17:12     2014-06-15
## # ℹ 490 more rows
## # ℹ 4 more variables: start.pos.x <dbl>, start.pos.y <dbl>, end.pos.x <dbl>,
## #   end.pos.y <dbl>
```

read_csv(csv_url)

- 외부에서 공개된 csv 파일도 바로 읽어올 수 있습니다.
    - github, gist, google drive

```
file_url <- "https://gist.githubusercontent.com/theoroe3/8bc989b644adc24117bc66f50c292fc8/raw/f
677a2ad811a9854c9d174178b0585a87569af60/tibbles_data.csv"
read_csv(file_url)
```

```
## Rows: 4 Columns: 4
## ── Column specification ────────────────────────────────────────────────────────
──────────────────────────────
## Delimiter: ","
## chr (1): name
## dbl (3): <-, 8, %
##
## 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.
```

```
## # A tibble: 4 × 4
##    `<-`   `8`   `%` name
##   <dbl> <dbl> <dbl> <chr>
## 1     1     2  0.25 t
## 2     2     4  0.25 h
## 3     3     6  0.25 e
## 4     4     8  0.25 o
```

locale 설정

- 한글이 포함된 데이터를 읽어올 매, 'read.csv'에서 'fileEncdoing'으로 조정을 하였습니다.
- 'read_csv' 에서는 주로 `locale` 인자를 설정해 주어야 하는데, 통상적으로 `locale('ko',encoding='euc-kr')` 와 같이 설정해줍니다.
- 예제는 아래의 연습문제에서 데이터를 불러오는 것으로 알아보겠습니다.

5. 3. 결측값 처리

- 결측값을 처리하는 방법으로 결측값이 있는 행을 삭제하거나, 다른 값으로 치환하는 방법이 있습니다.
- `drop_na()` 는 결측값이 있는 행을 삭제하는 함수입니다.
- `fill()` 은 인접한 값들을 이용해서 결측값을 치환하는 방법입니다.
- `replace_na()` 는 특정한 값을 이용해서 결측값을 치환하는 방법입니다.

drop_na()

```
library(dplyr)
df <- tibble(x = c(1, 2, NA), y = c("a", NA, "b"))
df %>% drop_na()
```

```
## # A tibble: 1 × 2
##       x y
##   <dbl> <chr>
## 1     1 a
```

```
df %>% drop_na(x)
```

```
## # A tibble: 2 × 2
##       x y
##   <dbl> <chr>
## 1     1 a
## 2     2 <NA>
```

```
vars <- "y"
df %>% drop_na(x, any_of(vars))
```

```
## # A tibble: 1 × 2
##       x y
##   <dbl> <chr>
## 1     1 a
```

fill()

```
sales <- tibble::tribble(
~quarter, ~year, ~sales,
"Q1", 2000, 66013,
"Q2", NA, 69182,
"Q3", NA, 53175,
"Q4", NA, 21001,
"Q1", 2001, 46036,
"Q2", NA, 58842,
"Q3", NA, 44568,
"Q4", NA, 50197,
"Q1", 2002, 39113,
"Q2", NA, 41668,
"Q3", NA, 30144,
"Q4", NA, 52897,
"Q1", 2004, 32129,
"Q2", NA, 67686,
"Q3", NA, 31768,
"Q4", NA, 49094)
sales %>% fill(year)
```

```
## # A tibble: 16 × 3
##    quarter  year sales
##    <chr>   <dbl> <dbl>
##  1 Q1       2000 66013
##  2 Q2       2000 69182
##  3 Q3       2000 53175
##  4 Q4       2000 21001
##  5 Q1       2001 46036
##  6 Q2       2001 58842
##  7 Q3       2001 44568
##  8 Q4       2001 50197
##  9 Q1       2002 39113
## 10 Q2       2002 41668
## 11 Q3       2002 30144
## 12 Q4       2002 52897
## 13 Q1       2004 32129
## 14 Q2       2004 67686
## 15 Q3       2004 31768
## 16 Q4       2004 49094
```

‘fill(.direction=“up”)’

```
tidy_pets <- tibble::tribble(
~rank, ~pet_type, ~breed,
1L, NA, "Boston Terrier",
2L, NA, "Retrievers (Labrador)",
3L, NA, "Retrievers (Golden)",
4L, NA, "French Bulldogs",
5L, NA, "Bulldogs",
6L, "Dog", "Beagles",
1L, NA, "Persian",
2L, NA, "Maine Coon",
3L, NA, "Ragdoll",
4L, NA, "Exotic",
5L, NA, "Siamese",
6L, "Cat", "American Short")
tidy_pets %>%
fill(pet_type, .direction = "up")
```

```
## # A tibble: 12 × 3
##     rank pet_type breed
##    <int> <chr>    <chr>
##  1     1 Dog      Boston Terrier
##  2     2 Dog      Retrievers (Labrador)
##  3     3 Dog      Retrievers (Golden)
##  4     4 Dog      French Bulldogs
##  5     5 Dog      Bulldogs
##  6     6 Dog      Beagles
##  7     1 Cat      Persian
##  8     2 Cat      Maine Coon
##  9     3 Cat      Ragdoll
## 10     4 Cat      Exotic
## 11     5 Cat      Siamese
## 12     6 Cat      American Short
```

replace_na()

```
df <- tibble(x = c(1, 2, NA), y = c("a", NA, "b"))
df %>% replace_na(list(x = 0, y = "unknown"))
```

```
## # A tibble: 3 × 2
##       x y
##   <dbl> <chr>
## 1     1 a
## 2     2 unknown
## 3     0 b
```

```
df %>% dplyr::mutate(x = replace_na(x, 0))
```

```
## # A tibble: 3 × 2
##       x y
##   <dbl> <chr>
## 1     1 a
## 2     2 <NA>
## 3     0 b
```

PR12 연습문제

```
data1 <- read.csv('data1.csv', fileEncoding = 'EUC-KR')
data2 <- read.csv('data2.csv', fileEncoding = 'EUC-KR')
```

문제 1

data1.csv에는 지역별 온실가스 배출량 정보가 있으며, data2.csv에는 지역별 기업수가 있다. 두 변수를 '시도'를 기준으로 하나의 데이터프레임으로 만드시오 조건1. wide형태를 long으로 바꾸어야 함 (gather 이용) 조건2. join을 진행하여야함. 조건3. head(, 10)을 통해 상위 10개만 출력하시오.

```
data2_long <- data2 %>% gather(광역시도명, 규모, -등록현황 )
```

```
joindata <- left_join(data1, data2_long, by = "광역시도명")
```

```
## Warning in left_join(data1, data2_long, by = "광역시도명"): Detected an unexpected many-to-m
any relationship between `x` and `y`.
## ℹ Row 1 of `x` matches multiple rows in `y`.
## ℹ Row 1 of `y` matches multiple rows in `x`.
## ℹ If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
head(joindata, 10)
```

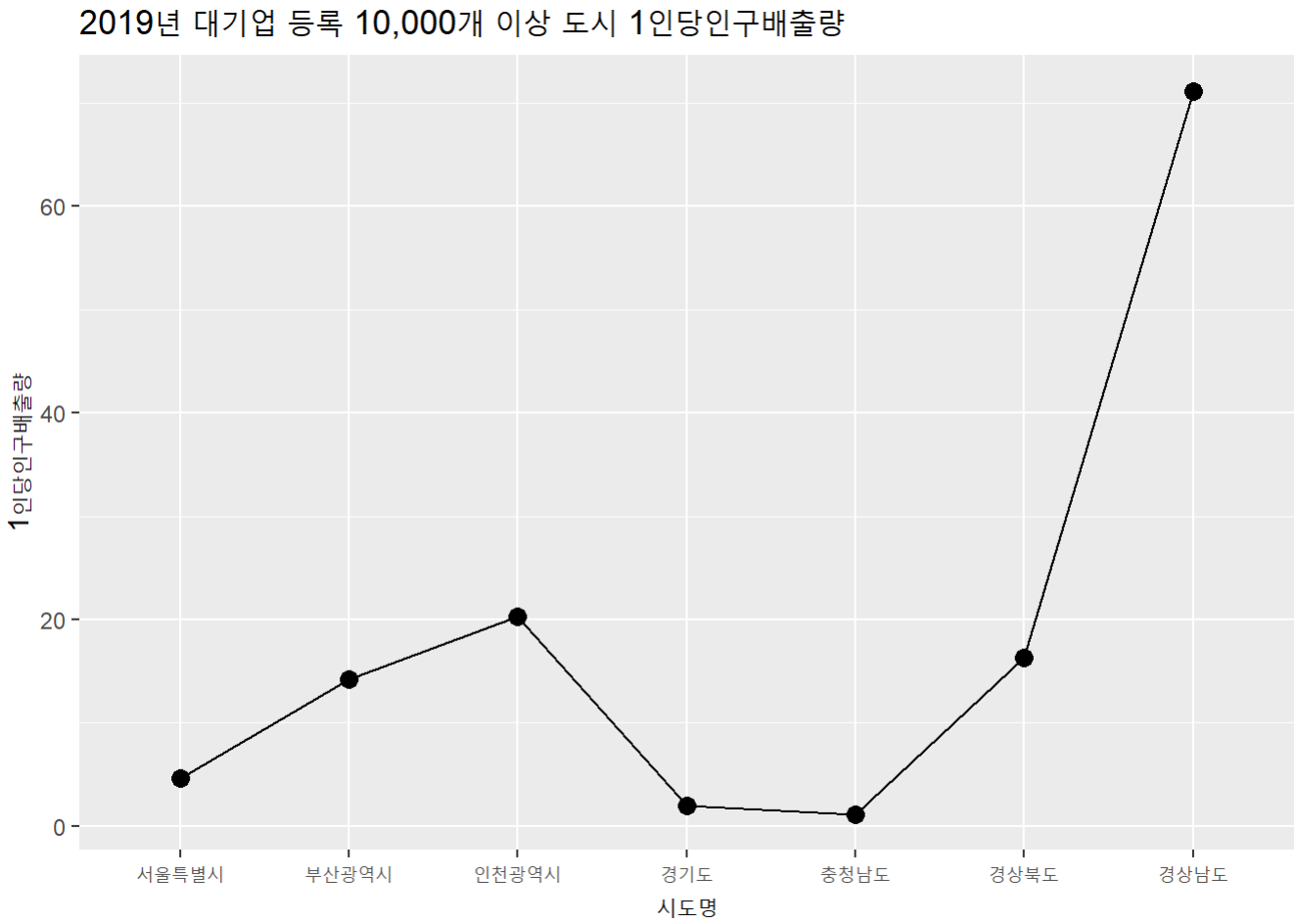
```
##    년도 광역시도명 전체온실가스배출량 X1인당인구배출량
## 1  2019 서울특별시           11366609            1.168
## 2  2019 서울특별시           11366609            1.168
## 3  2019 서울특별시           11366609            1.168
## 4  2019 서울특별시           11366609            1.168
## 5  2019 부산광역시            6747556            1.977
## 6  2019 부산광역시            6747556            1.977
## 7  2019 부산광역시            6747556            1.977
## 8  2019 부산광역시            6747556            1.977
## 9  2019 대구광역시            4003434            1.642
## 10 2019 대구광역시            4003434            1.642
##    관리업체1개당.온실가스배출량 사업장1개당온실가스.배출량       등록현황  규모
## 1                      35744.05                  3305.208   규모(소기업)    19
## 2                      35744.05                  3305.208   규모(중기업)   363
## 3                      35744.05                  3305.208   규모(대기업) 11206
## 4                      35744.05                  3305.208 규모(중견기업)     9
## 5                      47518.00                  9032.873   규모(소기업)    33
## 6                      47518.00                  9032.873   규모(중기업)   527
## 7                      47518.00                  9032.873   규모(대기업) 10202
## 8                      47518.00                  9032.873 규모(중견기업)     1
## 9                      33642.30                  8087.745   규모(소기업)    30
## 10                     33642.30                  8087.745   규모(중기업)   438
```

문제 2

```
filtered_data <- filter(joindata, 년도 == 2019, 등록현황 == '규모(대기업)', 규모 > 10000) %>%
  group_by(광역시도명)
filtered_data
```

```
## # A tibble: 7 × 8
## # Groups:   광역시도명 [7]
##    년도 광역시도명 전체온실가스배출량 X1인당인구배출량 관리업체1개당.온실가스…¹
##   <int> <chr>                 <int>           <dbl>                   <dbl>
## 1  2019 서울특별시          11366609            1.17                   35744.
## 2  2019 부산광역시           6747556            1.98                   47518
## 3  2019 인천광역시          48301096           16.3                   345008.
## 4  2019 경기도              61854822            4.67                  173263.
## 5  2019 충청남도           151149802           71.2                   767258.
## 6  2019 경상북도            54013530           20.3                   275579.
## 7  2019 경상남도            47656268           14.2                   264757.
## # ℹ abbreviated name: ¹ 관리업체1개당.온실가스배출량
## # ℹ 3 more variables: 사업장1개당온실가스.배출량 <dbl>, 등록현황 <chr>,
## #   규모 <int>
```

```
ggplot(filtered_data, aes(x = 광역시도명, y = X1인당인구배출량, group = 1)) +
  geom_line() +                       # 플롯에 선 추가
  geom_point(size = 3) +              # 플롯에 점 추가
  labs(title = "2019년 대기업 등록 10,000개 이상 도시 1인당인구배출량",   # 플롯 제목 설정
       x = "시도명",                  # x축 레이블 설정
       y = "1인당인구배출량") +  # y축 레이블 설정
  scale_x_discrete(labels = filtered_data$광역시도명)
```




```
 # x축에 대해 사용자 정의 레이블 설정
#applied_data <- joindata %>% group_by(광역시도명)
```

12 도전문제

1.

```
# 'corona.csv' 파일을 EUC-KR 인코딩으로 읽음
corona <- read.csv('corona.csv', fileEncoding = 'EUC-KR')

# '통계.기준일자' 열에서 " 00:00:00"을 빈 문자열로 대체
corona_time_cleaned <- corona %>%
  mutate(통계.기준일자 = sub(" 00:00:00", "", 통계.기준일자))

# 데이터의 상단을 확인.
head(corona_time_cleaned)
```

```
##   전일까지의.누적.통계..1차.접종. 전일까지의.누적.통계..2차.접종. 통계.기준일자
## 1                        11040300                        10576253    2021-11-21
## 2                        11027472                        10556185    2021-11-20
## 3                        11017006                        10539606    2021-11-19
## 4                        11006655                        10521937    2021-11-18
## 5                        10997612                        10507681    2021-11-17
## 6                        10981866                        10481876    2021-11-16
##   당일.통계.1차.접종. 당일.통계.2차.접종. 지역명칭 전체.누적.통계.1차.접종.
## 1                8359               12491   경기도                 11048659
## 2               11727               18983   경기도                 11039199
## 3                9659               15686   경기도                 11026665
## 4                9544               16745   경기도                 11016199
## 5                8134               13148   경기도                 11005746
## 6               14680               24600   경기도                 10996546
##   전체.누적.통계.2차.접종.
## 1                 10588744
## 2                 10575168
## 3                 10555292
## 4                 10538682
## 5                 10520829
## 6                 10506476
```

```
# 라이브러리 lubridate를 불러옴.
library(lubridate)

# 날짜 컬럼을 추출하기 위해 통계의 기준일자를 date_to_day 데이터프레임에 저장.
date_to_day <- data.frame(date = as.Date(corona_time_cleaned$통계.기준일자))

# date_to_day 데이터프레임의 날짜 컬럼을 이용하여 요일을 계산, corona_time_cleaned 데이터프레임에 weekday 컬럼을 추가.
corona_time_cleaned$weekday <- weekdays(date_to_day$date)

head(corona_time_cleaned)
```

```
##   전일까지의.누적.통계..1차.접종. 전일까지의.누적.통계..2차.접종. 통계.기준일자
## 1                        11040300                        10576253    2021-11-21
## 2                        11027472                        10556185    2021-11-20
## 3                        11017006                        10539606    2021-11-19
## 4                        11006655                        10521937    2021-11-18
## 5                        10997612                        10507681    2021-11-17
## 6                        10981866                        10481876    2021-11-16
##   당일.통계.1차.접종. 당일.통계.2차.접종. 지역명칭 전체.누적.통계.1차.접종.
## 1                8359               12491   경기도                  11048659
## 2               11727               18983   경기도                  11039199
## 3                9659               15686   경기도                  11026665
## 4                9544               16745   경기도                  11016199
## 5                8134               13148   경기도                  11005746
## 6               14680               24600   경기도                  10996546
##   전체.누적.통계.2차.접종. weekday
## 1                10588744   일요일
## 2                10575168   토요일
## 3                10555292   금요일
## 4                10538682   목요일
## 5                10520829   수요일
## 6                10506476   화요일
```

```
# separate를 통해 통계.기준일자의 날짜를 - 를 기준으로 각각 year, month, day 컬럼을 새롭게 만듦
corona_time_cleaned <- corona_time_cleaned %>% separate(통계.기준일자, c("year", "month", "da
y"))
head(corona_time_cleaned)
```

```
##   전일까지의.누적.통계..1차.접종. 전일까지의.누적.통계..2차.접종. year month
## 1                        11040300                        10576253 2021    11
## 2                        11027472                        10556185 2021    11
## 3                        11017006                        10539606 2021    11
## 4                        11006655                        10521937 2021    11
## 5                        10997612                        10507681 2021    11
## 6                        10981866                        10481876 2021    11
##   day 당일.통계.1차.접종. 당일.통계.2차.접종. 지역명칭 전체.누적.통계.1차.접종.
## 1  21                8359               12491   경기도                  11048659
## 2  20               11727               18983   경기도                  11039199
## 3  19                9659               15686   경기도                  11026665
## 4  18                9544               16745   경기도                  11016199
## 5  17                8134               13148   경기도                  11005746
## 6  16               14680               24600   경기도                  10996546
##   전체.누적.통계.2차.접종. weekday
## 1                10588744   일요일
## 2                10575168   토요일
## 3                10555292   금요일
## 4                10538682   목요일
## 5                10520829   수요일
## 6                10506476   화요일
```

```
# corona_time_cleaned 데이터프레임을 사용하여 요일별로 1차 백신 접종과 2차 백신 접종 통계의 평균을 계산.
vaccination_day <- corona_time_cleaned %>%
  group_by(weekday) %>%
  summarise(mean_1st_vaccination = mean(당일.통계.1차.접종.), mean_2nd_vaccination = mean(당일.통계.2차.접종.))

# vaccination_day 데이터프레임 출력
vaccination_day
```

```
## # A tibble: 7 × 3
##   weekday mean_1st_vaccination mean_2nd_vaccination
##   <chr>                  <dbl>                <dbl>
## 1 금요일                49303.               50253.
## 2 목요일                38265.               38389.
## 3 수요일                47424.               46058
## 4 월요일                 3057.                2910.
## 5 일요일                31347.               30308.
## 6 토요일                58545.               66108.
## 7 화요일                57428.               45164.
```

```
weekday_order <- c("월요일", "화요일", "수요일", "목요일", "금요일", "토요일", "일요일")

# weekday를 순서대로 정렬
data_sorted <- vaccination_day %>%
  arrange(factor(weekday, levels = weekday_order))

# 결과 출력
print(data_sorted)
```

```
## # A tibble: 7 × 3
##   weekday mean_1st_vaccination mean_2nd_vaccination
##   <chr>                  <dbl>                <dbl>
## 1 월요일                 3057.                2910.
## 2 화요일                57428.               45164.
## 3 수요일                47424.               46058
## 4 목요일                38265.               38389.
## 5 금요일                49303.               50253.
## 6 토요일                58545.               66108.
## 7 일요일                31347.               30308.
```

```
# corona_time_cleaned 데이터셋을 month로 그룹화하여,


vaccination_month <- corona_time_cleaned %>%
  group_by(month) %>%
  summarise(mean_1st_vaccination = mean(당일.통계.1차.접종.), # 당일.통계.1차.접종.의 평균인 mean_1st_vaccination과
            mean_2nd_vaccination = mean(당일.통계.2차.접종.)) # 당일.통계.2차.접종.의 평균인 mean_2nd_vaccination을 계산하여
vaccination_month # vaccination_month 데이터프레임을 생성합니다.
```

```
## # A tibble: 9 × 3
##   month mean_1st_vaccination mean_2nd_vaccination
##   <chr>                <dbl>                <dbl>
## 1 03                   3685.                 30.8
## 2 04                  15705.                953.
## 3 05                  16075.              13630.
## 4 06                  72608.              21053.
## 5 07                  33046.              12184.
## 6 08                  81634.              63118.
## 7 09                  93295.              86108.
## 8 10                  18793.             121137.
## 9 11                  13765.              22842.
```

2.Tidy data

```
library(tidyverse)
```

```
table1
```

```
## # A tibble: 6 × 4
##   country      year  cases population
##   <chr>       <dbl>  <dbl>      <dbl>
## 1 Afghanistan  1999    745   19987071
## 2 Afghanistan  2000   2666   20595360
## 3 Brazil       1999  37737  172006362
## 4 Brazil       2000  80488  174504898
## 5 China        1999 212258 1272915272
## 6 China        2000 213766 1280428583
```

```
#> # A tibble: 6 × 4
#>   country      year  cases population
#>   <chr>       <dbl>  <dbl>      <dbl>
#> 1 Afghanistan  1999    745   19987071
#> 2 Afghanistan  2000   2666   20595360
#> 3 Brazil       1999  37737  172006362
#> 4 Brazil       2000  80488  174504898
#> 5 China        1999 212258 1272915272
#> 6 China        2000 213766 1280428583
table2
```

```
## # A tibble: 12 × 4
##    country      year type            count
##    <chr>       <dbl> <chr>           <dbl>
##  1 Afghanistan  1999 cases             745
##  2 Afghanistan  1999 population   19987071
##  3 Afghanistan  2000 cases            2666
##  4 Afghanistan  2000 population   20595360
##  5 Brazil       1999 cases           37737
##  6 Brazil       1999 population  172006362
##  7 Brazil       2000 cases           80488
##  8 Brazil       2000 population  174504898
##  9 China        1999 cases          212258
## 10 China        1999 population 1272915272
## 11 China        2000 cases          213766
## 12 China        2000 population 1280428583
```

```
#> # A tibble: 12 × 4
#>   country      year type           count
#>   <chr>       <dbl> <chr>          <dbl>
#> 1 Afghanistan  1999 cases            745
#> 2 Afghanistan  1999 population  19987071
#> 3 Afghanistan  2000 cases           2666
#> 4 Afghanistan  2000 population  20595360
#> 5 Brazil       1999 cases           37737
#> 6 Brazil       1999 population 172006362
#> # ℹ 6 more rows
table3
```

```
## # A tibble: 6 × 3
##   country      year rate
##   <chr>       <dbl> <chr>
## 1 Afghanistan  1999 745/19987071
## 2 Afghanistan  2000 2666/20595360
## 3 Brazil       1999 37737/172006362
## 4 Brazil       2000 80488/174504898
## 5 China        1999 212258/1272915272
## 6 China        2000 213766/1280428583
```

```
#> # A tibble: 6 × 3
#>   country      year rate
#>   <chr>       <dbl> <chr>
#> 1 Afghanistan  1999 745/19987071
#> 2 Afghanistan  2000 2666/20595360
#> 3 Brazil       1999 37737/172006362
#> 4 Brazil       2000 80488/174504898
#> 5 China        1999 212258/1272915272
#> 6 China        2000 213766/1280428583

# Spread across two tibbles
table4a  # cases
```

```
## # A tibble: 3 × 3
##   country     `1999` `2000`
##   <chr>        <dbl>  <dbl>
## 1 Afghanistan    745   2666
## 2 Brazil       37737  80488
## 3 China       212258 213766
```

```
#> # A tibble: 3 × 3
#>   country     `1999` `2000`
#>   <chr>        <dbl>  <dbl>
#> 1 Afghanistan    745   2666
#> 2 Brazil       37737  80488
#> 3 China       212258 213766
table4b  # population
```

```
## # A tibble: 3 × 3
##   country         `1999`     `2000`
##   <chr>            <dbl>      <dbl>
## 1 Afghanistan   19987071   20595360
## 2 Brazil       172006362  174504898
## 3 China       1272915272 1280428583
```

```
#> # A tibble: 3 × 3
#>   country         `1999`     `2000`
#>   <chr>            <dbl>      <dbl>
#> 1 Afghanistan   19987071   20595360
#> 2 Brazil       172006362  174504898
#> 3 China       1272915272 1280428583
```

```
# Compute rate per 10,000
table1 %>%
  mutate(rate = cases / population * 10000)
```

```
## # A tibble: 6 × 5
##   country      year  cases population  rate
##   <chr>       <dbl>  <dbl>      <dbl> <dbl>
## 1 Afghanistan  1999    745   19987071 0.373
## 2 Afghanistan  2000   2666   20595360 1.29
## 3 Brazil       1999  37737  172006362 2.19
## 4 Brazil       2000  80488  174504898 4.61
## 5 China        1999 212258 1272915272 1.67
## 6 China        2000 213766 1280428583 1.67
```

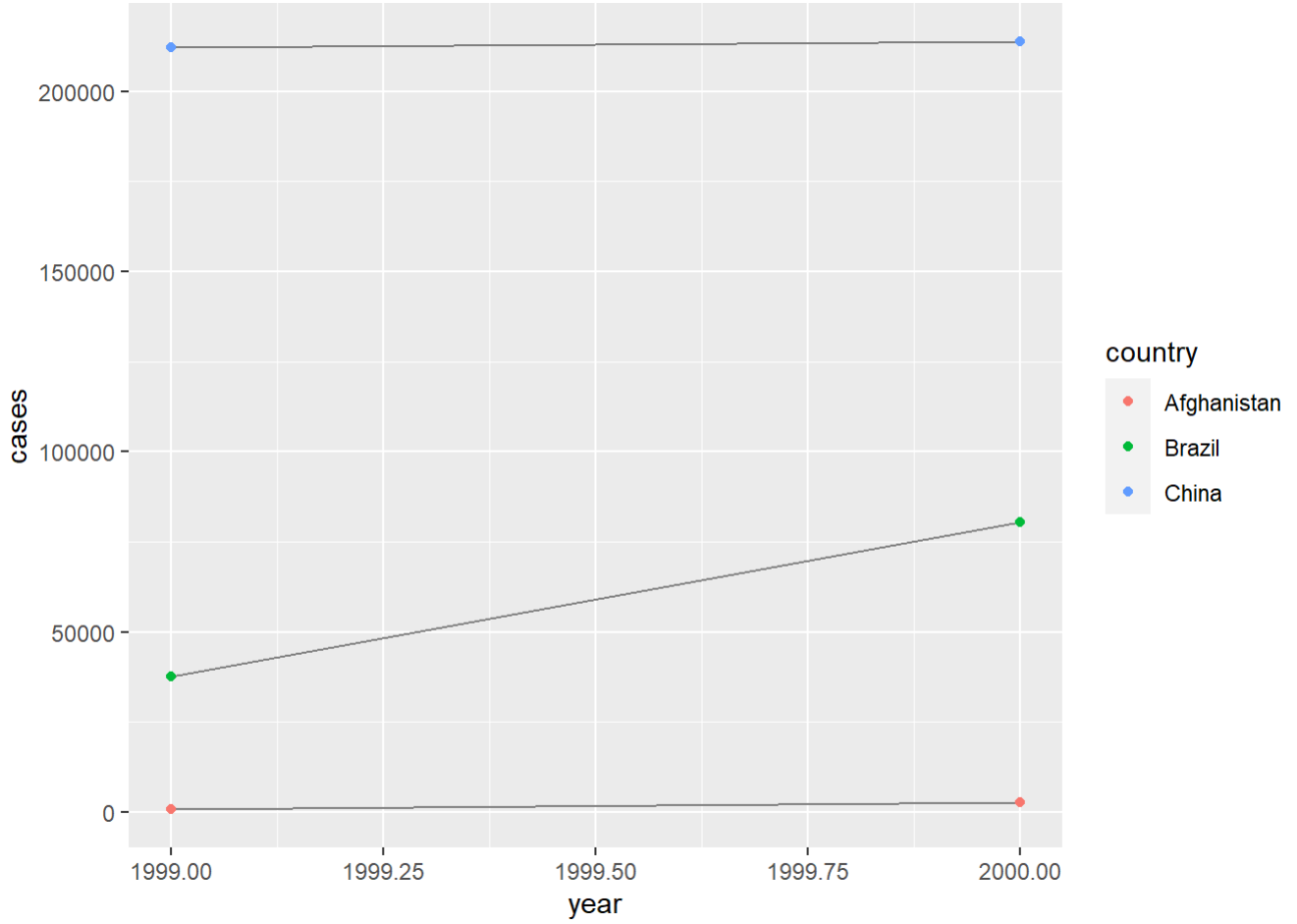
```
#> # A tibble: 6 × 5
#>   country      year  cases population  rate
#>   <chr>       <dbl>  <dbl>      <dbl> <dbl>
#> 1 Afghanistan  1999    745   19987071 0.373
#> 2 Afghanistan  2000   2666   20595360 1.29
#> 3 Brazil       1999  37737  172006362 2.19
#> 4 Brazil       2000  80488  174504898 4.61
#> 5 China        1999 212258 1272915272 1.67
#> 6 China        2000 213766 1280428583 1.67

# Compute cases per year
table1 %>%
  count(year, wt = cases)
```

```
## # A tibble: 2 × 2
##    year      n
##   <dbl>  <dbl>
## 1  1999 250740
## 2  2000 296920
```

```
#> # A tibble: 2 × 2
#>    year      n
#>   <dbl>  <dbl>
#> 1  1999 250740
#> 2  2000 296920

# Visualise changes over time
library(ggplot2)
ggplot(table1, aes(year, cases)) +
  geom_line(aes(group = country), colour = "grey50") +
  geom_point(aes(colour = country))
```




```
table4a
```

```
## # A tibble: 3 × 3
##   country     `1999` `2000`
##   <chr>        <dbl>  <dbl>
## 1 Afghanistan    745   2666
## 2 Brazil       37737  80488
## 3 China       212258 213766
```

```
#> # A tibble: 3 × 3
#>   country     `1999` `2000`
#>   <chr>        <dbl>  <dbl>
#> 1 Afghanistan    745   2666
#> 2 Brazil       37737  80488
#> 3 China       212258 213766
```

```
table4a %>%
  pivot_longer(c(`1999`, `2000`), names_to = "year", values_to = "cases")
```

```
## # A tibble: 6 × 3
##   country     year   cases
##   <chr>       <chr>  <dbl>
## 1 Afghanistan 1999     745
## 2 Afghanistan 2000    2666
## 3 Brazil      1999   37737
## 4 Brazil      2000   80488
## 5 China       1999  212258
## 6 China       2000  213766
```

```
#> # A tibble: 6 × 3
#>   country     year   cases
#>   <chr>       <chr>  <dbl>
#> 1 Afghanistan 1999     745
#> 2 Afghanistan 2000    2666
#> 3 Brazil      1999   37737
#> 4 Brazil      2000   80488
#> 5 China       1999  212258
#> 6 China       2000  213766
```

```
table4b %>%
  pivot_longer(c(`1999`, `2000`), names_to = "year", values_to = "population")
```

```
## # A tibble: 6 × 3
##   country     year  population
##   <chr>       <chr>      <dbl>
## 1 Afghanistan 1999    19987071
## 2 Afghanistan 2000    20595360
## 3 Brazil      1999   172006362
## 4 Brazil      2000   174504898
## 5 China       1999  1272915272
## 6 China       2000  1280428583
```

```
#> # A tibble: 6 × 3
#>   country     year  population
#>   <chr>       <chr>      <dbl>
#> 1 Afghanistan 1999    19987071
#> 2 Afghanistan 2000    20595360
#> 3 Brazil      1999   172006362
#> 4 Brazil      2000   174504898
#> 5 China       1999  1272915272
#> 6 China       2000  1280428583
```

```
tidy4a <- table4a %>%
  pivot_longer(c(`1999`, `2000`), names_to = "year", values_to = "cases")
tidy4b <- table4b %>%
  pivot_longer(c(`1999`, `2000`), names_to = "year", values_to = "population")
left_join(tidy4a, tidy4b)
```

```
## Joining with `by = join_by(country, year)`
```

```
## # A tibble: 6 × 4
##   country     year   cases population
##   <chr>       <chr>  <dbl>      <dbl>
## 1 Afghanistan 1999     745   19987071
## 2 Afghanistan 2000    2666   20595360
## 3 Brazil      1999   37737  172006362
## 4 Brazil      2000   80488  174504898
## 5 China       1999  212258 1272915272
## 6 China       2000  213766 1280428583
```

```
#> Joining with `by = join_by(country, year)`
#> # A tibble: 6 × 4
#>   country     year   cases population
#>   <chr>       <chr>  <dbl>      <dbl>
#> 1 Afghanistan 1999     745   19987071
#> 2 Afghanistan 2000    2666   20595360
#> 3 Brazil      1999   37737  172006362
#> 4 Brazil      2000   80488  174504898
#> 5 China       1999  212258 1272915272
#> 6 China       2000  213766 1280428583
```

```
head(table2)
```

```
## # A tibble: 6 × 4
##   country      year type           count
##   <chr>       <dbl> <chr>          <dbl>
## 1 Afghanistan  1999 cases            745
## 2 Afghanistan  1999 population  19987071
## 3 Afghanistan  2000 cases           2666
## 4 Afghanistan  2000 population  20595360
## 5 Brazil       1999 cases          37737
## 6 Brazil       1999 population 172006362
```

```
#> # A tibble: 12 × 4
#>   country      year type           count
#>   <chr>       <dbl> <chr>          <dbl>
#> 1 Afghanistan  1999 cases            745
#> 2 Afghanistan  1999 population  19987071
#> 3 Afghanistan  2000 cases           2666
#> 4 Afghanistan  2000 population  20595360
#> 5 Brazil       1999 cases          37737
#> 6 Brazil       1999 population 172006362
#> # ℹ 6 more rows
```

```
table2 %>%
    pivot_wider(names_from = type, values_from = count)
```

```
## # A tibble: 6 × 4
##   country      year  cases population
##   <chr>       <dbl>  <dbl>      <dbl>
## 1 Afghanistan  1999    745   19987071
## 2 Afghanistan  2000   2666   20595360
## 3 Brazil       1999  37737  172006362
## 4 Brazil       2000  80488  174504898
## 5 China        1999 212258 1272915272
## 6 China        2000 213766 1280428583
```

```
#> # A tibble: 6 × 4
#>   country      year  cases population
#>   <chr>       <dbl>  <dbl>      <dbl>
#> 1 Afghanistan  1999    745   19987071
#> 2 Afghanistan  2000   2666   20595360
#> 3 Brazil       1999  37737  172006362
#> 4 Brazil       2000  80488  174504898
#> 5 China        1999 212258 1272915272
#> 6 China        2000 213766 1280428583
```

```
stocks <- tibble(
  year   = c(2015, 2015, 2016, 2016),
  half  = c(   1,    2,     1,    2),
  return = c(1.88, 0.59, 0.92, 0.17)
)
stocks %>%
  pivot_wider(names_from = year, values_from = return) %>%
  pivot_longer(`2015`:`2016`, names_to = "year", values_to = "return")
```

```
## # A tibble: 4 × 3
##    half year  return
##   <dbl> <chr>  <dbl>
## 1     1 2015    1.88
## 2     1 2016    0.92
## 3     2 2015    0.59
## 4     2 2016    0.17
```

```
#table4a %>%
  #pivot_longer(c(1999, 2000), names_to = "year", values_to = "cases")
#> Error in `pivot_longer()`:
#> ! Can't subset columns past the end.
#> ℹ Locations 1999 and 2000 don't exist.
#> ℹ There are only 3 columns.
```

```
people <- tribble(
  ~name,             ~names,  ~values,
  #-----------------|--------|------
  "Phillip Woods",   "age",       45,
  "Phillip Woods",   "height",   186,
  "Phillip Woods",   "age",       50,
  "Jessica Cordero", "age",       37,
  "Jessica Cordero", "height",   156
)
```

```
preg <- tribble(
  ~pregnant, ~male, ~female,
  "yes",     NA,    10,
  "no",      20,    12
)
```

```
table3
```

```
## # A tibble: 6 × 3
##   country      year rate
##   <chr>       <dbl> <chr>
## 1 Afghanistan  1999 745/19987071
## 2 Afghanistan  2000 2666/20595360
## 3 Brazil       1999 37737/172006362
## 4 Brazil       2000 80488/174504898
## 5 China        1999 212258/1272915272
## 6 China        2000 213766/1280428583
```

```
#> # A tibble: 6 × 3
#>   country      year rate
#>   <chr>       <dbl> <chr>
#> 1 Afghanistan  1999 745/19987071
#> 2 Afghanistan  2000 2666/20595360
#> 3 Brazil       1999 37737/172006362
#> 4 Brazil       2000 80488/174504898
#> 5 China        1999 212258/1272915272
#> 6 China        2000 213766/1280428583
```

```
table3 %>%
  separate(rate, into = c("cases", "population"))
```

```
## # A tibble: 6 × 4
##   country      year cases  population
##   <chr>       <dbl> <chr>  <chr>
## 1 Afghanistan  1999 745    19987071
## 2 Afghanistan  2000 2666   20595360
## 3 Brazil       1999 37737  172006362
## 4 Brazil       2000 80488  174504898
## 5 China        1999 212258 1272915272
## 6 China        2000 213766 1280428583
```

```
#> # A tibble: 6 × 4
#>   country      year cases  population
#>   <chr>       <dbl> <chr>  <chr>
#> 1 Afghanistan  1999 745    19987071
#> 2 Afghanistan  2000 2666   20595360
#> 3 Brazil       1999 37737  172006362
#> 4 Brazil       2000 80488  174504898
#> 5 China        1999 212258 1272915272
#> 6 China        2000 213766 1280428583
```

```
table3 %>%
  separate(rate, into = c("cases", "population"), sep = "/")
```

```
## # A tibble: 6 × 4
##   country      year cases  population
##   <chr>       <dbl> <chr>  <chr>
## 1 Afghanistan  1999 745    19987071
## 2 Afghanistan  2000 2666   20595360
## 3 Brazil       1999 37737  172006362
## 4 Brazil       2000 80488  174504898
## 5 China        1999 212258 1272915272
## 6 China        2000 213766 1280428583
```

```
table3 %>%
  separate(rate, into = c("cases", "population"), convert = TRUE)
```

```
## # A tibble: 6 × 4
##   country      year  cases population
##   <chr>       <dbl>  <int>      <int>
## 1 Afghanistan  1999    745   19987071
## 2 Afghanistan  2000   2666   20595360
## 3 Brazil       1999  37737  172006362
## 4 Brazil       2000  80488  174504898
## 5 China        1999 212258 1272915272
## 6 China        2000 213766 1280428583
```

```
#> # A tibble: 6 × 4
#>   country      year  cases population
#>   <chr>       <dbl>  <int>      <int>
#> 1 Afghanistan  1999    745   19987071
#> 2 Afghanistan  2000   2666   20595360
#> 3 Brazil       1999  37737  172006362
#> 4 Brazil       2000  80488  174504898
#> 5 China        1999 212258 1272915272
#> 6 China        2000 213766 1280428583
```

```
table3 %>%
  separate(year, into = c("century", "year"), sep = 2)
```

```
## # A tibble: 6 × 4
##   country     century year  rate
##   <chr>       <chr>   <chr> <chr>
## 1 Afghanistan 19      99    745/19987071
## 2 Afghanistan 20      00    2666/20595360
## 3 Brazil      19      99    37737/172006362
## 4 Brazil      20      00    80488/174504898
## 5 China       19      99    212258/1272915272
## 6 China       20      00    213766/1280428583
```

```
#> # A tibble: 6 × 4
#>   country     century year  rate
#>   <chr>       <chr>   <chr> <chr>
#> 1 Afghanistan 19      99    745/19987071
#> 2 Afghanistan 20      00    2666/20595360
#> 3 Brazil      19      99    37737/172006362
#> 4 Brazil      20      00    80488/174504898
#> 5 China       19      99    212258/1272915272
#> 6 China       20      00    213766/1280428583
```

```
table5 %>%
  unite(new, century, year)
```

```
## # A tibble: 6 × 3
##   country     new   rate
##   <chr>       <chr> <chr>
## 1 Afghanistan 19_99 745/19987071
## 2 Afghanistan 20_00 2666/20595360
## 3 Brazil      19_99 37737/172006362
## 4 Brazil      20_00 80488/174504898
## 5 China       19_99 212258/1272915272
## 6 China       20_00 213766/1280428583
```

```
#> # A tibble: 6 × 3
#>   country     new   rate
#>   <chr>       <chr> <chr>
#> 1 Afghanistan 19_99 745/19987071
#> 2 Afghanistan 20_00 2666/20595360
#> 3 Brazil      19_99 37737/172006362
#> 4 Brazil      20_00 80488/174504898
#> 5 China       19_99 212258/1272915272
#> 6 China       20_00 213766/1280428583
```

```
table5 %>%
  unite(new, century, year, sep = "")
```

```
## # A tibble: 6 × 3
##   country     new   rate
##   <chr>       <chr> <chr>
## 1 Afghanistan 1999  745/19987071
## 2 Afghanistan 2000  2666/20595360
## 3 Brazil      1999  37737/172006362
## 4 Brazil      2000  80488/174504898
## 5 China       1999  212258/1272915272
## 6 China       2000  213766/1280428583
```

```
#> # A tibble: 6 × 3
#>   country     new   rate
#>   <chr>       <chr> <chr>
#> 1 Afghanistan 1999  745/19987071
#> 2 Afghanistan 2000  2666/20595360
#> 3 Brazil      1999  37737/172006362
#> 4 Brazil      2000  80488/174504898
#> 5 China       1999  212258/1272915272
#> 6 China       2000  213766/1280428583
```

```
tibble(x = c("a,b,c", "d,e,f,g", "h,i,j")) %>%
  separate(x, c("one", "two", "three"))
```

```
## Warning: Expected 3 pieces. Additional pieces discarded in 1 rows [2].
```

```
## # A tibble: 3 × 3
##   one   two   three
##   <chr> <chr> <chr>
## 1 a     b     c
## 2 d     e     f
## 3 h     i     j
```

```
tibble(x = c("a,b,c", "d,e", "f,g,i")) %>%
  separate(x, c("one", "two", "three"))
```

```
## Warning: Expected 3 pieces. Missing pieces filled with `NA` in 1 rows [2].
```

```
## # A tibble: 3 × 3
##   one   two   three
##   <chr> <chr> <chr>
## 1 a     b     c
## 2 d     e     <NA>
## 3 f     g     i
```

```
stocks <- tibble(
  year   = c(2015, 2015, 2015, 2015, 2016, 2016, 2016),
  qtr    = c(   1,    2,    3,    4,    2,    3,    4),
  return = c(1.88, 0.59, 0.35,   NA, 0.92, 0.17, 2.66)
)
```

```
stocks %>%
  pivot_wider(names_from = year, values_from = return)
```

```
## # A tibble: 4 × 3
##     qtr `2015` `2016`
##   <dbl>  <dbl>  <dbl>
## 1     1   1.88  NA
## 2     2   0.59   0.92
## 3     3   0.35   0.17
## 4     4  NA      2.66
```

```
#> # A tibble: 4 × 3
#>     qtr `2015` `2016`
#>   <dbl>  <dbl>  <dbl>
#> 1     1   1.88  NA
#> 2     2   0.59   0.92
#> 3     3   0.35   0.17
#> 4     4  NA      2.66
```

```
stocks %>%
  pivot_wider(names_from = year, values_from = return) %>%
  pivot_longer(
    cols = c(`2015`, `2016`),
    names_to = "year",
    values_to = "return",
    values_drop_na = TRUE
  )
```

```
## # A tibble: 6 × 3
##     qtr year  return
##   <dbl> <chr>  <dbl>
## 1     1 2015    1.88
## 2     2 2015    0.59
## 3     2 2016    0.92
## 4     3 2015    0.35
## 5     3 2016    0.17
## 6     4 2016    2.66
```

```
#> # A tibble: 6 × 3
#>     qtr year  return
#>   <dbl> <chr>  <dbl>
#> 1     1 2015    1.88
#> 2     2 2015    0.59
#> 3     2 2016    0.92
#> 4     3 2015    0.35
#> 5     3 2016    0.17
#> 6     4 2016    2.66
```

```
stocks %>%
  complete(year, qtr)
```

```
## # A tibble: 8 × 3
##    year   qtr return
##   <dbl> <dbl>  <dbl>
## 1  2015     1   1.88
## 2  2015     2   0.59
## 3  2015     3   0.35
## 4  2015     4  NA
## 5  2016     1  NA
## 6  2016     2   0.92
## 7  2016     3   0.17
## 8  2016     4   2.66
```

```
#> # A tibble: 8 × 3
#>    year   qtr return
#>   <dbl> <dbl>  <dbl>
#> 1  2015     1   1.88
#> 2  2015     2   0.59
#> 3  2015     3   0.35
#> 4  2015     4  NA
#> 5  2016     1  NA
#> 6  2016     2   0.92
#> # ℹ 2 more rows
```

```
stocks %>%
  complete(year, qtr)
```

```
## # A tibble: 8 × 3
##    year   qtr return
##   <dbl> <dbl>  <dbl>
## 1  2015     1   1.88
## 2  2015     2   0.59
## 3  2015     3   0.35
## 4  2015     4  NA
## 5  2016     1  NA
## 6  2016     2   0.92
## 7  2016     3   0.17
## 8  2016     4   2.66
```

```
#> # A tibble: 8 × 3
#>    year   qtr return
#>   <dbl> <dbl>  <dbl>
#> 1  2015     1   1.88
#> 2  2015     2   0.59
#> 3  2015     3   0.35
#> 4  2015     4  NA
#> 5  2016     1  NA
#> 6  2016     2   0.92
#> # ℹ 2 more rows
```

```
treatment <- tribble(
  ~ person,           ~ treatment, ~response,
  "Derrick Whitmore", 1,           7,
  NA,                 2,           10,
  NA,                 3,           9,
  "Katherine Burke",  1,           4
)
```

```
treatment %>%
  fill(person)
```

```
## # A tibble: 4 × 3
##   person           treatment response
##   <chr>                <dbl>    <dbl>
## 1 Derrick Whitmore         1        7
## 2 Derrick Whitmore         2       10
## 3 Derrick Whitmore         3        9
## 4 Katherine Burke          1        4
```

```
#> # A tibble: 4 × 3
#>   person           treatment response
#>   <chr>                <dbl>    <dbl>
#> 1 Derrick Whitmore         1        7
#> 2 Derrick Whitmore         2       10
#> 3 Derrick Whitmore         3        9
#> 4 Katherine Burke          1        4
```

```
head(who)
```

```
## # A tibble: 6 × 60
##   country   iso2  iso3   year new_sp_m014 new_sp_m1524 new_sp_m2534 new_sp_m3544
##   <chr>     <chr> <chr> <int>       <int>        <int>        <int>        <int>
## 1 Afghanis… AF    AFG    1980          NA           NA           NA           NA
## 2 Afghanis… AF    AFG    1981          NA           NA           NA           NA
## 3 Afghanis… AF    AFG    1982          NA           NA           NA           NA
## 4 Afghanis… AF    AFG    1983          NA           NA           NA           NA
## 5 Afghanis… AF    AFG    1984          NA           NA           NA           NA
## 6 Afghanis… AF    AFG    1985          NA           NA           NA           NA
## # ℹ 52 more variables: new_sp_m4554 <int>, new_sp_m5564 <int>,
## #   new_sp_m65 <int>, new_sp_f014 <int>, new_sp_f1524 <int>,
## #   new_sp_f2534 <int>, new_sp_f3544 <int>, new_sp_f4554 <int>,
## #   new_sp_f5564 <int>, new_sp_f65 <int>, new_sn_m014 <int>,
## #   new_sn_m1524 <int>, new_sn_m2534 <int>, new_sn_m3544 <int>,
## #   new_sn_m4554 <int>, new_sn_m5564 <int>, new_sn_m65 <int>,
## #   new_sn_f014 <int>, new_sn_f1524 <int>, new_sn_f2534 <int>, …
```

```
who <- who
#> # A tibble: 7,240 × 60
#>   country   iso2  iso3   year new_sp_m014 new_sp_m1524 new_sp_m2534 new_sp_m3544
#>   <chr>     <chr> <chr> <dbl>       <dbl>        <dbl>        <dbl>        <dbl>
#> 1 Afghanis… AF    AFG    1980          NA           NA           NA           NA
#> 2 Afghanis… AF    AFG    1981          NA           NA           NA           NA
#> 3 Afghanis… AF    AFG    1982          NA           NA           NA           NA
#> 4 Afghanis… AF    AFG    1983          NA           NA           NA           NA
#> 5 Afghanis… AF    AFG    1984          NA           NA           NA           NA
#> 6 Afghanis… AF    AFG    1985          NA           NA           NA           NA
#> # ℹ 7,234 more rows
#> # ℹ 52 more variables: new_sp_m4554 <dbl>, new_sp_m5564 <dbl>,
#> #   new_sp_m65 <dbl>, new_sp_f014 <dbl>, new_sp_f1524 <dbl>,
#> #   new_sp_f2534 <dbl>, new_sp_f3544 <dbl>, new_sp_f4554 <dbl>,
#> #   new_sp_f5564 <dbl>, new_sp_f65 <dbl>, new_sn_m014 <dbl>,
#> #   new_sn_m1524 <dbl>, new_sn_m2534 <dbl>, new_sn_m3544 <dbl>,
#> #   new_sn_m4554 <dbl>, new_sn_m5564 <dbl>, new_sn_m65 <dbl>, …
```

```
who1 <- who %>%
  pivot_longer(
    cols = 5:40,
    names_to = "key",
    values_to = "cases",
    values_drop_na = TRUE
  )
head(who1)
```

```
## # A tibble: 6 × 26
##   country  iso2  iso3   year new_ep_f1524 new_ep_f2534 new_ep_f3544 new_ep_f4554
##   <chr>    <chr> <chr> <int>        <int>        <int>        <int>        <int>
## 1 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 2 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 3 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 4 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 5 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 6 Afghani… AF    AFG    1997           NA           NA           NA           NA
## # ℹ 18 more variables: new_ep_f5564 <int>, new_ep_f65 <int>,
## #   new_rel_m014 <int>, new_rel_m1524 <int>, new_rel_m2534 <int>,
## #   new_rel_m3544 <int>, new_rel_m4554 <int>, new_rel_m5564 <int>,
## #   new_rel_m65 <int>, new_rel_f014 <int>, new_rel_f1524 <int>,
## #   new_rel_f2534 <int>, new_rel_f3544 <int>, new_rel_f4554 <int>,
## #   new_rel_f5564 <int>, new_rel_f65 <int>, key <chr>, cases <int>
```

```
#> # A tibble: 76,046 × 6
#>   country     iso2  iso3   year key          cases
#>   <chr>       <chr> <chr> <dbl> <chr>        <dbl>
#> 1 Afghanistan AF    AFG    1997 new_sp_m014      0
#> 2 Afghanistan AF    AFG    1997 new_sp_m1524    10
#> 3 Afghanistan AF    AFG    1997 new_sp_m2534     6
#> 4 Afghanistan AF    AFG    1997 new_sp_m3544     3
#> 5 Afghanistan AF    AFG    1997 new_sp_m4554     5
#> 6 Afghanistan AF    AFG    1997 new_sp_m5564     2
#> # ℹ 76,040 more rows
```

```
head(who1 %>%
  count(key))
```

```
## # A tibble: 6 × 2
##   key              n
##   <chr>        <int>
## 1 new_ep_f014   1032
## 2 new_ep_m014   1038
## 3 new_ep_m1524  1026
## 4 new_ep_m2534  1020
## 5 new_ep_m3544  1024
## 6 new_ep_m4554  1020
```

```
#> # A tibble: 56 × 2
#>   key              n
#>   <chr>        <int>
#> 1 new_ep_f014   1032
#> 2 new_ep_f1524  1021
#> 3 new_ep_f2534  1021
#> 4 new_ep_f3544  1021
#> 5 new_ep_f4554  1017
#> 6 new_ep_f5564  1017
#> # ℹ 50 more rows
```

```
who2 <- who1 %>%
  mutate(key = stringr::str_replace(key, "newrel", "new_rel"))
head(who2)
```

```
## # A tibble: 6 × 26
##   country  iso2  iso3   year new_ep_f1524 new_ep_f2534 new_ep_f3544 new_ep_f4554
##   <chr>    <chr> <chr> <int>        <int>        <int>        <int>        <int>
## 1 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 2 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 3 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 4 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 5 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 6 Afghani… AF    AFG    1997           NA           NA           NA           NA
## # ℹ 18 more variables: new_ep_f5564 <int>, new_ep_f65 <int>,
## #   new_rel_m014 <int>, new_rel_m1524 <int>, new_rel_m2534 <int>,
## #   new_rel_m3544 <int>, new_rel_m4554 <int>, new_rel_m5564 <int>,
## #   new_rel_m65 <int>, new_rel_f014 <int>, new_rel_f1524 <int>,
## #   new_rel_f2534 <int>, new_rel_f3544 <int>, new_rel_f4554 <int>,
## #   new_rel_f5564 <int>, new_rel_f65 <int>, key <chr>, cases <int>
```

```
#> # A tibble: 76,046 × 6
#>   country     iso2  iso3   year key          cases
#>   <chr>       <chr> <chr> <dbl> <chr>        <dbl>
#> 1 Afghanistan AF    AFG    1997 new_sp_m014      0
#> 2 Afghanistan AF    AFG    1997 new_sp_m1524    10
#> 3 Afghanistan AF    AFG    1997 new_sp_m2534     6
#> 4 Afghanistan AF    AFG    1997 new_sp_m3544     3
#> 5 Afghanistan AF    AFG    1997 new_sp_m4554     5
#> 6 Afghanistan AF    AFG    1997 new_sp_m5564     2
#> # ℹ 76,040 more rows
```

```
who3 <- who2 %>%
  separate(key, c("new", "type", "sexage"), sep = "_")
head(who3)
```

```
## # A tibble: 6 × 28
##   country  iso2  iso3   year new_ep_f1524 new_ep_f2534 new_ep_f3544 new_ep_f4554
##   <chr>    <chr> <chr> <int>        <int>        <int>        <int>        <int>
## 1 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 2 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 3 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 4 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 5 Afghani… AF    AFG    1997           NA           NA           NA           NA
## 6 Afghani… AF    AFG    1997           NA           NA           NA           NA
## # ℹ 20 more variables: new_ep_f5564 <int>, new_ep_f65 <int>,
## #   new_rel_m014 <int>, new_rel_m1524 <int>, new_rel_m2534 <int>,
## #   new_rel_m3544 <int>, new_rel_m4554 <int>, new_rel_m5564 <int>,
## #   new_rel_m65 <int>, new_rel_f014 <int>, new_rel_f1524 <int>,
## #   new_rel_f2534 <int>, new_rel_f3544 <int>, new_rel_f4554 <int>,
## #   new_rel_f5564 <int>, new_rel_f65 <int>, new <chr>, type <chr>,
## #   sexage <chr>, cases <int>
```

```
#> # A tibble: 76,046 × 8
#>   country     iso2  iso3   year new   type  sexage cases
#>   <chr>       <chr> <chr> <dbl> <chr> <chr> <chr>  <dbl>
#> 1 Afghanistan AF    AFG    1997 new   sp    m014       0
#> 2 Afghanistan AF    AFG    1997 new   sp    m1524     10
#> 3 Afghanistan AF    AFG    1997 new   sp    m2534      6
#> 4 Afghanistan AF    AFG    1997 new   sp    m3544      3
#> 5 Afghanistan AF    AFG    1997 new   sp    m4554      5
#> 6 Afghanistan AF    AFG    1997 new   sp    m5564      2
#> # ℹ 76,040 more rows
```

```
who3 %>%
  count(new)
```

```
## # A tibble: 1 × 2
##   new       n
##   <chr> <int>
## 1 new   67355
```

```
#> # A tibble: 1 × 2
#>   new       n
#>   <chr> <int>
#> 1 new   76046
who4 <- who3 %>%
  select(-new, -iso2, -iso3)
```

```
who5 <- who4 %>%
  separate(sexage, c("sex", "age"), sep = 1)
who5
```

```
## # A tibble: 67,355 × 26
##    country      year new_ep_f1524 new_ep_f2534 new_ep_f3544 new_ep_f4554
##    <chr>       <int>        <int>        <int>        <int>        <int>
##  1 Afghanistan  1997           NA           NA           NA           NA
##  2 Afghanistan  1997           NA           NA           NA           NA
##  3 Afghanistan  1997           NA           NA           NA           NA
##  4 Afghanistan  1997           NA           NA           NA           NA
##  5 Afghanistan  1997           NA           NA           NA           NA
##  6 Afghanistan  1997           NA           NA           NA           NA
##  7 Afghanistan  1997           NA           NA           NA           NA
##  8 Afghanistan  1997           NA           NA           NA           NA
##  9 Afghanistan  1997           NA           NA           NA           NA
## 10 Afghanistan  1997           NA           NA           NA           NA
## # ℹ 67,345 more rows
## # ℹ 20 more variables: new_ep_f5564 <int>, new_ep_f65 <int>,
## #   new_rel_m014 <int>, new_rel_m1524 <int>, new_rel_m2534 <int>,
## #   new_rel_m3544 <int>, new_rel_m4554 <int>, new_rel_m5564 <int>,
## #   new_rel_m65 <int>, new_rel_f014 <int>, new_rel_f1524 <int>,
## #   new_rel_f2534 <int>, new_rel_f3544 <int>, new_rel_f4554 <int>,
## #   new_rel_f5564 <int>, new_rel_f65 <int>, type <chr>, sex <chr>, age <chr>, …
```

```
#> # A tibble: 76,046 × 6
#>   country      year type  sex   age   cases
#>   <chr>       <dbl> <chr> <chr> <chr> <dbl>
#> 1 Afghanistan  1997 sp    m     014       0
#> 2 Afghanistan  1997 sp    m     1524     10
#> 3 Afghanistan  1997 sp    m     2534      6
#> 4 Afghanistan  1997 sp    m     3544      3
#> 5 Afghanistan  1997 sp    m     4554      5
#> 6 Afghanistan  1997 sp    m     5564      2
#> # ℹ 76,040 more rows
```

```
head(who %>%
  pivot_longer(
    cols = 5:40,
    names_to = "key",
    values_to = "cases",
    values_drop_na = TRUE
  ) %>%
  mutate(
    key = stringr::str_replace(key, "newrel", "new_rel")
  ) %>%
  separate(key, c("new", "var", "sexage")) %>%
  select(-new, -iso2, -iso3) %>%
  separate(sexage, c("sex", "age"), sep = 1))
```

```
## # A tibble: 6 × 26
##   country  year new_ep_f1524 new_ep_f2534 new_ep_f3544 new_ep_f4554 new_ep_f5564
##   <chr>   <int>        <int>        <int>        <int>        <int>        <int>
## 1 Afghan…  1997           NA           NA           NA           NA           NA
## 2 Afghan…  1997           NA           NA           NA           NA           NA
## 3 Afghan…  1997           NA           NA           NA           NA           NA
## 4 Afghan…  1997           NA           NA           NA           NA           NA
## 5 Afghan…  1997           NA           NA           NA           NA           NA
## 6 Afghan…  1997           NA           NA           NA           NA           NA
## # ℹ 19 more variables: new_ep_f65 <int>, new_rel_m014 <int>,
## #   new_rel_m1524 <int>, new_rel_m2534 <int>, new_rel_m3544 <int>,
## #   new_rel_m4554 <int>, new_rel_m5564 <int>, new_rel_m65 <int>,
## #   new_rel_f014 <int>, new_rel_f1524 <int>, new_rel_f2534 <int>,
## #   new_rel_f3544 <int>, new_rel_f4554 <int>, new_rel_f5564 <int>,
## #   new_rel_f65 <int>, var <chr>, sex <chr>, age <chr>, cases <int>
```