

# Data manipulation with dplyr

Completed by Leon Woltermann

## Contents

Aims of this worksheet . . . . .	1
Selecting columns ( <code>select()</code> ) . . . . .	3
Filtering rows ( <code>filter()</code> ) . . . . .	4
Creating new columns ( <code>mutate()</code> ) . . . . .	5
Sorting columns ( <code>arrange()</code> ) . . . . .	6
Split-apply-combine ( <code>group_by()</code> ) . . . . .	7
Summarizing or aggregating data ( <code>summarize()</code> ) . . . . .	10

**NB:** The worksheet has been developed and prepared by Lincoln Mullen. Source: Lincoln A. Mullen, *Computational Historical Thinking: With Applications in R (2018-)*: <http://dh-r.lincolnmullen.com>. Minor modifications added by Maxim Romanov (loading `methodists` dataset).

The best way to learn R or computational history is to practice. These worksheets contain a series of questions designed to teach you about R or different computational methods. The worksheets are R Markdown documents that include text and code together. The places where you are expected to answer questions are marked like this.

(Q) Can you make a plot from this dataset?

Beneath each question is a space to either create a code block or write an answer.

## Aims of this worksheet

One of the key reasons to use R is to be able to manipulate data with ease. After completing this worksheet you will be able to work with the most commonly used data manipulation verbs provided by the `dplyr` and `tidyr` packages.

We will begin by loading the necessary packages and data. We will use the `methodists` dataset from the `historydata` package. This dataset contains membership figures for Methodist meetings (which were organized into districts, which were in turn organized into conferences) for the early nineteenth century.

```
library(tidyverse)
library(historydata)
#data(methodists)
#methodists
```

### `methodists` data (MGR)

**NB:** It seems like there are some errors in the `historydata` package, and the `methodists` dataset does not properly load. The package itself is available on GitHub (<https://github.com/ropensci/historydata>), so we can try a different way of getting the data that we need for the worksheet. Specifically, if we know the exact address of the data file (url), we can open it with the `read.csv` command, like shown below (you need to be connected to Internet, of course):

```
methodists <- read.csv("https://raw.githubusercontent.com/ropensci/historydata/master/data-raw/methodists")
head(methodists)
```

```
##   minutes_year minutes_date minutes_location conference district      meeting
## 1          1773   1773-06-01   Philadelphia                New York
## 2          1773   1773-06-01   Philadelphia      Philadelphia
## 3          1773   1773-06-01   Philadelphia      New Jersey
## 4          1773   1773-06-01   Philadelphia      Maryland
## 5          1773   1773-06-01   Philadelphia      Virginia
## 6          1774   1774-05-25   Philadelphia      New York
##   state members_general members_white members_colored members_indian
## 1          180             NA             NA             NA
## 2          180             NA             NA             NA
## 3          200             NA             NA             NA
## 4          500             NA             NA             NA
## 5          100             NA             NA             NA
## 6          222             NA             NA             NA
##                                     url notes
## 1 http://hdl.handle.net/2027/nyp.33433069134967?urlappend=%3Bseq=13
## 2 http://hdl.handle.net/2027/nyp.33433069134967?urlappend=%3Bseq=13
## 3 http://hdl.handle.net/2027/nyp.33433069134967?urlappend=%3Bseq=13
## 4 http://hdl.handle.net/2027/nyp.33433069134967?urlappend=%3Bseq=13
## 5 http://hdl.handle.net/2027/nyp.33433069134967?urlappend=%3Bseq=13
## 6 http://hdl.handle.net/2027/nyp.33433069134967?urlappend=%3Bseq=14
```

This data file, however, is slightly different from what we need, so some minor modifications will be necessary. You do not need to be concerned about the code in the next chunk, just run it.

```
#library(dplyr)

replace_na <- function(x, val = 0L) {
  ifelse(is.na(x), val, x)
}

methodists <- methodists %>%
  as_tibble() %>%
  filter(minutes_year != 1778,
         minutes_year != 1779,
         minutes_year != 1785) %>%
  filter(minutes_year >= 1786, minutes_year <= 1834) %>%
  dplyr::rename(members_black = members_colored,
               year = minutes_year) %>%
  mutate(members_indian = as.integer(members_indian)) %>%
  mutate(members_white = replace_na(members_white),
         members_black = replace_na(members_black),
         members_indian = replace_na(members_indian)) %>%
  rowwise() %>%
  mutate(members_total = sum(members_general, members_white, members_black,
                           members_indian, na.rm = TRUE)) %>%
  ungroup() %>%
  select(year, conference, district, meeting, state, members_total,
        starts_with("members_"), url)
```

(1) You are welcome to try to interpret the code above and describe it in your own words (*hard*):

## Selecting columns (`select()`)

The first data manipulation verb that we are going to use is `select()`. This function lets us pass the names of the columns that we want to keep.

```
methodists %>%  
  select(year, meeting, members_total)
```

```
## # A tibble: 20,241 x 3  
##   year meeting      members_total  
##   <int> <chr>          <int>  
## 1  1786 Portsmouth        356  
## 2  1786 Sussex          488  
## 3  1786 Brunswick        364  
## 4  1786 Amelia          412  
## 5  1786 Mecklenburg      429  
## 6  1786 Bedford         540  
## 7  1786 Orange          449  
## 8  1786 Williamsburg     178  
## 9  1786 Alleghany        368  
## 10 1786 Berkley         166  
## # ... with 20,231 more rows
```

Notice that we have not actually changed the data stored in `methodists` until we assign the changed data to a variable.

Read the documentation for this function, `?select`.

- (2) Select the columns for `year`, `meeting`, as well as all columns that begin with the word `members_`.

```
methodists %>%  
  select(year, meeting, starts_with("members_"))
```

```
## # A tibble: 20,241 x 7  
##   year meeting      members_total members_general members_white members_black  
##   <int> <chr>          <int>          <int>          <int>          <int>  
## 1  1786 Portsmouth        356             NA           330           26  
## 2  1786 Sussex          488             NA           416           72  
## 3  1786 Brunswick        364             NA           305           59  
## 4  1786 Amelia          412             NA           382           30  
## 5  1786 Mecklenburg      429             NA           392           37  
## 6  1786 Bedford         540             NA           524           16  
## 7  1786 Orange          449             NA           374           75  
## 8  1786 Williamsburg     178             NA           167           11  
## 9  1786 Alleghany        368             NA           350           18  
## 10 1786 Berkley         166             NA           140           26  
## # ... with 20,231 more rows, and 1 more variable: members_indian <int>
```

- (3) Remove the column `url`.

```
select(methodists, -url)
```

```
## # A tibble: 20,241 x 10  
##   year conference district meeting      state members_total members_general  
##   <int> <chr>      <chr>    <chr>    <chr>          <int>          <int>  
## 1  1786 ""      ""      Portsmouth ""          356           NA  
## 2  1786 ""      ""      Sussex    ""          488           NA  
## 3  1786 ""      ""      Brunswick ""          364           NA  
## 4  1786 ""      ""      Amelia    ""          412           NA
```

```
## 5 1786 "" "" Mecklenburg "" 429 NA
## 6 1786 "" "" Bedford "" 540 NA
## 7 1786 "" "" Orange "" 449 NA
## 8 1786 "" "" Williamsburg "" 178 NA
## 9 1786 "" "" Alleghany "" 368 NA
## 10 1786 "" "" Berkley "" 166 NA
## # ... with 20,231 more rows, and 3 more variables: members_white <int>,
## #   members_black <int>, members_indian <int>
```

## Filtering rows (`filter()`)

The `select()` function lets us pick certain columns. The `filter()` function lets select certain rows based on logical conditions. For example, here we get the only the meetings where the total number of members is at greater than 1,000.

```
methodists %>%
  filter(members_total > 1000)
```

```
## # A tibble: 1,891 x 11
##   year conference district meeting state members_total members_general
##   <int> <chr>      <chr>    <chr>    <chr>      <int>      <int>
## 1 1786 ""      ""      Kent      ""      1013      NA
## 2 1787 ""      ""      Kent      ""      1211      NA
## 3 1787 ""      ""      Talbot    ""      1601      NA
## 4 1788 ""      ""      Sussex    ""      1611      NA
## 5 1788 ""      ""      Brunswick ""      1604      NA
## 6 1788 ""      ""      Mecklenburg ""      1109      NA
## 7 1788 ""      ""      Calvert   ""      1347      NA
## 8 1788 ""      ""      Baltimore ""      1219      NA
## 9 1788 ""      ""      Kent      ""      1600      NA
## 10 1788 ""      ""      Talbot    ""      1601      NA
## # ... with 1,881 more rows, and 4 more variables: members_white <int>,
## #   members_black <int>, members_indian <int>, url <chr>
```

(4) Get just the rows from New York in 1800.

```
filter(methodists, state == "New York", year == 1800)
```

```
## # A tibble: 18 x 11
##   year conference district meeting state members_total members_general
##   <int> <chr>      <chr>    <chr>    <chr>      <int>      <int>
## 1 1800 ""      ""      Albany City New ~      40      NA
## 2 1800 ""      ""      Albany Circuit New ~      708      NA
## 3 1800 ""      ""      Brooklyn    New ~      54      NA
## 4 1800 ""      ""      Cambridge    New ~      703      NA
## 5 1800 ""      ""      Chenango     New ~      227      NA
## 6 1800 ""      ""      Columbia     New ~      144      NA
## 7 1800 ""      ""      Delaware     New ~      380      NA
## 8 1800 ""      ""      Dutchess     New ~      321      NA
## 9 1800 ""      ""      Herkimer     New ~      294      NA
## 10 1800 ""      ""      Long Island  New ~      390      NA
## 11 1800 ""      ""      Mohawk       New ~      242      NA
## 12 1800 ""      ""      Newburg      New ~      359      NA
## 13 1800 ""      ""      New Rochelle a~ New ~      744      NA
## 14 1800 ""      ""      New York     New ~      776      NA
## 15 1800 ""      ""      Oneida and Cay~ New ~      209      NA
```

```
## 16 1800 "" "" Plattsburg New ~ 107 NA
## 17 1800 "" "" Saratoga New ~ 444 NA
## 18 1800 "" "" Seneca New ~ 221 NA
## # ... with 4 more variables: members_white <int>, members_black <int>,
## # members_indian <int>, url <chr>
```

(5) Which Methodist meetings had only black members?

```
filter(methodists, members_total == members_black & members_total != 0)
```

```
## # A tibble: 40 x 11
##   year conference district meeting state members_total members_general
##   <int> <chr>      <chr>    <chr> <chr>      <int>      <int>
## 1 1786 ""        ""      Antigua ""      1000      NA
## 2 1787 ""        ""      Bladen ""       30      NA
## 3 1819 "New York" "New York" Zion a~ ""      791      NA
## 4 1820 "New York" "New York" Zion a~ ""      690      NA
## 5 1823 "New York" "New York" Asbury~ ""      134      NA
## 6 1830 "South Carolina" "Augusta" Missio~ ""      240      NA
## 7 1830 "South Carolina" "Charlest~ Missio~ ""      107      NA
## 8 1830 "South Carolina" "Charlest~ Missio~ ""      310      NA
## 9 1831 "Georgia" "Athens" Missio~ ""      165      NA
## 10 1831 "South Carolina" "Charlest~ Missio~ ""      440      NA
## # ... with 30 more rows, and 4 more variables: members_white <int>,
## # members_black <int>, members_indian <int>, url <chr>
```

## Creating new columns (mutate())

Very often one will want to create a new column based on other columns in the data. For instance, in our Methodist data, there is a column called `year`, but that column represents the year that the minutes were reported. The membership figures are actually for the previous year. Here we create a new column called `year_recorded`, where each value is one less than in `year`.

```
methodists %>%
  mutate(year_recorded = year - 1) %>%
  select(year, year_recorded, meeting)
```

```
## # A tibble: 20,241 x 3
##   year year_recorded meeting
##   <int>      <dbl> <chr>
## 1 1786      1785 Portsmouth
## 2 1786      1785 Sussex
## 3 1786      1785 Brunswick
## 4 1786      1785 Amelia
## 5 1786      1785 Mecklenburg
## 6 1786      1785 Bedford
## 7 1786      1785 Orange
## 8 1786      1785 Williamsburg
## 9 1786      1785 Alleghany
## 10 1786      1785 Berkley
## # ... with 20,231 more rows
```

Notice that we chained the data manipulation functions using the pipe (`%>%`). This lets us create a pipeline where we can do many different manipulations in a row.

(6) Create two new columns, one with the percentage of white members, and one with the percentage of black members.

```
methodists %>%
  mutate(percentage_white = members_white/members_total * 100) %>%
  mutate(percentage_black = members_black/members_total * 100) %>%
  select(year, members_total, percentage_white, percentage_black)
```

```
## # A tibble: 20,241 x 4
##   year members_total percentage_white percentage_black
##   <int>      <int>      <dbl>      <dbl>
## 1 1786         356        92.7         7.30
## 2 1786         488        85.2        14.8
## 3 1786         364        83.8        16.2
## 4 1786         412        92.7         7.28
## 5 1786         429        91.4         8.62
## 6 1786         540        97.0         2.96
## 7 1786         449        83.3        16.7
## 8 1786         178        93.8         6.18
## 9 1786         368        95.1         4.89
## 10 1786         166        84.3        15.7
## # ... with 20,231 more rows
```

## Sorting columns (arrange())

Often we want to sort a data frame by one of its columns. This can be done with the verb `arrange()`. By default `arrange()` will sort from least to greatest; we can use the function `desc()` to sort from greatest to least. In this example, we sort the data frame to get the meetings with the highest number of white members.

```
methodists %>%
  arrange(desc(members_white))
```

```
## # A tibble: 20,241 x 11
##   year conference district meeting      state members_total members_general
##   <int> <chr>      <chr>    <chr>    <chr>      <int>      <int>
## 1 1825 Ohio      Lancaster Muskingum ""         7775      NA
## 2 1832 New York  New York  New York ""         5355      NA
## 3 1831 New York  New York  New York ""         4953      NA
## 4 1830 New York  New York  New York ""         3955      NA
## 5 1829 New York  New York  New York ""         3839      NA
## 6 1831 Baltimore Baltimore Baltimore ""         3764      NA
## 7 1834 Baltimore Baltimore Baltimore city ""         3740      NA
## 8 1833 Baltimore Baltimore Baltimore city ""         3700      NA
## 9 1832 Baltimore Baltimore Baltimore ""         3622      NA
## 10 1828 New York  New York  New York ""         3477      NA
## # ... with 20,231 more rows, and 4 more variables: members_white <int>,
## #   members_black <int>, members_indian <int>, url <chr>
```

- (7) Which meetings had the highest number of black members? Select only the necessary columns so that the results print in a meaningful way.

```
methodists %>%
  arrange(desc(members_black)) %>%
  select(year, conference, district, meeting, members_total, members_black)
```

```
## # A tibble: 20,241 x 6
##   year conference      district meeting      members_total members_black
##   <int> <chr>      <chr>    <chr>      <int>      <int>
## 1 1817 South Carolina Edisto    Charleston    6049      5699
```

```
## 2 1816 South Carolina Edisto Charleston 5658 5313
## 3 1815 South Carolina Edisto Charleston 4058 3796
## 4 1832 South Carolina Charleston Charleston 4252 3629
## 5 1813 South Carolina Edisto Charleston 3925 3604
## 6 1814 South Carolina Edisto Charleston 3697 3418
## 7 1831 South Carolina Charleston Charleston 3967 3354
## 8 1834 South Carolina Charleston Charleston 3902 3249
## 9 1833 South Carolina Charleston Charleston 3860 3221
## 10 1830 South Carolina Charleston Charleston 3711 3163
## # ... with 20,231 more rows
```

(8) Which meetings had the high percentage of black members without being entirely black?

```
methodists %>%
  mutate(percentage_black = members_black/members_total * 100) %>%
  arrange(desc(percentage_black)) %>%
  filter(percentage_black != 100)
```

```
## # A tibble: 20,120 x 12
##   year conference district meeting state members_total members_general
##   <int> <chr> <chr> <chr> <chr> <int> <int>
## 1 1834 "South Carolina" "Lincolnt~ Combah~ "" 1172 NA
## 2 1799 "" "" George~ "Sou~ 188 NA
## 3 1798 "" "" George~ "Sou~ 187 NA
## 4 1816 "South Carolina" "Pee Dee" George~ "" 1743 NA
## 5 1800 "" "" George~ "Sou~ 233 NA
## 6 1833 "South Carolina" "Charlest~ Missio~ "" 1109 NA
## 7 1801 "" "" George~ "Sou~ 302 NA
## 8 1815 "South Carolina" "Pedee" George~ "" 1646 NA
## 9 1805 "South Carolina" "Camden" Wilmin~ "" 427 NA
## 10 1828 "South Carolina" "Fayettev~ George~ "" 1401 NA
## # ... with 20,110 more rows, and 5 more variables: members_white <int>,
## # members_black <int>, members_indian <int>, url <chr>,
## # percentage_black <dbl>
```

## Split-apply-combine (group\_by())

Notice that in the example above the `arrange()` function sorted the entire data frame. So when we looked for the circuits with the largest number of members, we got rows from 1825, then 1830, then 1829, then 1830, and so on. What if we wanted to get the biggest circuit from each year?

We can solve this kind of problem with what Hadley Wickham calls the “split-apply-combine” pattern of data analysis. Think of it this way. First we can *split* the big data frame into separate data frames, one for each year. Then we can *apply* our logic to get the results we want; in this case, that means sorting the data frame. We might also want to get just the top one row with the biggest number of members. Then we can *combine* those split apart data frames into a new data frame.

Take a simple example using the `top_n()` function, which returns the top `n` (in this case, top 1) results for a particular column. After selecting a few columns, we get the row in the data frame which has the highest value for `members_black`.

```
methodists %>%
  select(year, meeting, members_total, members_black) %>%
  top_n(1, members_black)
```

```
## # A tibble: 1 x 4
##   year meeting members_total members_black
```

```
##   <int> <chr>           <int>           <int>
## 1  1817 Charleston      6049           5699
```

We can change how that code works by using the `group_by()` function. Now we get the one row for each unique year in the dataset.

```
methodists %>%
  select(year, meeting, members_total, members_black) %>%
  group_by(year) %>%
  top_n(1, members_black)
```

```
## # A tibble: 49 x 4
## # Groups:   year [49]
##   year meeting members_total members_black
##   <int> <chr>           <int>           <int>
## 1  1786 Antigua         1000           1000
## 2  1787 Kent            1211            604
## 3  1788 Calvert         1347            842
## 4  1789 Calvert         1852            909
## 5  1790 Calvert         1984           1170
## 6  1791 Calvert         2089           1329
## 7  1792 Calvert         1900           1200
## 8  1793 Surry           1769            955
## 9  1794 Calvert         1784           1102
## 10 1795 Calvert         1526            924
## # ... with 39 more rows
```

We get the same results more concisely and reliably, though the steps of “split-apply-combine” are perhaps somewhat less easy to see.

(9) For each year, which was the biggest circuit?

```
methodists %>%
  group_by(year) %>%
  top_n(1, members_total)
```

```
## # A tibble: 49 x 11
## # Groups:   year [49]
##   year conference district meeting state members_total members_general
##   <int> <chr>           <chr>   <chr>   <chr>           <int>           <int>
## 1  1786 ""           ""      Kent    ""             1013            NA
## 2  1787 ""           ""      Talbot  ""             1601            NA
## 3  1788 ""           ""      Sussex  ""             1611            NA
## 4  1789 ""           ""      Calvert  ""             1852            NA
## 5  1790 ""           ""      Calvert  ""             1984            NA
## 6  1791 ""           ""      Calvert  ""             2089            NA
## 7  1792 ""           ""      Calvert  ""             1900            NA
## 8  1793 ""           ""      Surry    ""             1769            NA
## 9  1794 ""           ""      Sussex  ""             2354            NA
## 10 1795 ""           ""      Calvert  ""             1526            NA
## # ... with 39 more rows, and 4 more variables: members_white <int>,
## #   members_black <int>, members_indian <int>, url <chr>
```

(10) For each year, which church had the biggest percentage of black members without being entirely black?

```
methodists %>%
  mutate(percentage_black = members_black/members_total * 100) %>%
  filter(percentage_black != 100) %>%
```



```
group_by(year) %>%
  top_n(1, percentage_black)
```

```
## # A tibble: 50 x 12
## # Groups:   year [49]
##   year conference district meeting      state members_total members_general
##   <int> <chr>      <chr>    <chr>      <chr>      <int>      <int>
## 1 1786 ""          ""      Calvert    ""          611        NA
## 2 1787 ""          ""      Charleston ""          87         NA
## 3 1788 ""          ""      Calvert    ""         1347        NA
## 4 1789 ""          ""      Mecklenburg ""          790        NA
## 5 1790 ""          ""      Annapolis  ""          307        NA
## 6 1791 ""          ""      Charleston ""          185        NA
## 7 1792 ""          ""      Georgetown ""          149        NA
## 8 1793 ""          ""      Prince George's ""          290        NA
## 9 1793 ""          ""      Georgetown ""          232        NA
## 10 1794 ""          ""      Charleston ""          280        NA
## # ... with 40 more rows, and 5 more variables: members_white <int>,
## #   members_black <int>, members_indian <int>, url <chr>,
## #   percentage_black <dbl>
```

(11) For the year 1825, what was the biggest meeting in each conference? In each district?

```
methodists %>%
  filter(year == 1825) %>%
  group_by(conference, district) %>%
  top_n(1, members_total)
```

```
## # A tibble: 75 x 11
## # Groups:   conference, district [75]
##   year conference district meeting      state members_total members_general
##   <int> <chr>      <chr>    <chr>      <chr>      <int>      <int>
## 1 1825 Ohio      Ohio      Youngstown ""          701        NA
## 2 1825 Ohio      Portland Mansfield  ""          785        NA
## 3 1825 Ohio      Lancaster Muskingum  ""         7775        NA
## 4 1825 Ohio      Muskingum Barnesville ""         1090        NA
## 5 1825 Ohio      Scioto    Deer Creek ""         1022        NA
## 6 1825 Ohio      Lebanon  Mad River  ""         1419        NA
## 7 1825 Ohio      Miami    Madison    ""          906        NA
## 8 1825 Kentucky Kenhawa  Little Kenh~ ""          648        NA
## 9 1825 Kentucky Augusta  Fleming    ""          930        NA
## 10 1825 Kentucky Green River Christian ""          734        NA
## # ... with 65 more rows, and 4 more variables: members_white <int>,
## #   members_black <int>, members_indian <int>, url <chr>
```

(12) For each year, what was the biggest church in the Baltimore conference?

```
methodists %>%
  filter(conference == "Baltimore") %>%
  group_by(year) %>%
  top_n(1, members_total)
```

```
## # A tibble: 33 x 11
## # Groups:   year [33]
##   year conference district meeting state members_total members_general
##   <int> <chr>      <chr>    <chr>    <chr>      <int>      <int>
```

```
## 1 1802 Baltimore Baltimore Calvert "" 1612 NA
## 2 1803 Baltimore Baltimore Calvert "" 2052 NA
## 3 1804 Baltimore Baltimore Calvert "" 2457 NA
## 4 1805 Baltimore Baltimore Calvert "" 2531 NA
## 5 1806 Baltimore Baltimore Calvert "" 2421 NA
## 6 1807 Baltimore Baltimore Calvert "" 2544 NA
## 7 1808 Baltimore Baltimore Calvert "" 2273 NA
## 8 1809 Baltimore Baltimore Calvert "" 2182 NA
## 9 1810 Baltimore Baltimore Calvert "" 2303 NA
## 10 1811 Baltimore Baltimore Calvert "" 2198 NA
## # ... with 23 more rows, and 4 more variables: members_white <int>,
## #   members_black <int>, members_indian <int>, url <chr>
```

## Summarizing or aggregating data (`summarize()`)

In the examples using `top_n()` we performed a very simple kind of data summary, where we took the single row with the biggest value in a given column. This essentially boiled many rows of a data frame down into a single row. We would like to be able to summarize or aggregate a data frame in other ways as well. For instance, we often want to take the sum or the mean of a given column. We can do this using the `summarize()` function in conjunction with the `group_by()` function.

In this example, we group by the year the minutes were taken. Then we find the total number of white members for each year.

```
methodists %>%
  group_by(year) %>%
  summarize(total_members_white = sum(members_white, na.rm = TRUE))
```

```
## # A tibble: 49 x 2
##   year total_members_white
##   <int>         <int>
## 1 1786         18291
## 2 1787         21949
## 3 1788         30557
## 4 1789         34425
## 5 1790         45983
## 6 1791         50580
## 7 1792         52079
## 8 1793         51486
## 9 1794         52794
## 10 1795         48121
## # ... with 39 more rows
```

Notice that we get one row in the recombined data frame for each group in the original data frame. The value in the new column is the result of a function (in this case, `sum()`) applied to the columns in each of the split apart data frames.

There is also a special case where we might want to know how many rows were in each of the split apart (or grouped) data frames. We can use the special `n()` function to get that count. (This is such a common thing to do that dplyr provides the special function `count()` to do this in an abbreviated way. You can look up that function's documentation to see how it works.)

```
methodists %>%
  group_by(year) %>%
  summarize(total_meetings = n())
```

```
## # A tibble: 49 x 2
```

```
##   year total_meetings
##   <int>         <int>
## 1 1786           51
## 2 1787           55
## 3 1788           76
## 4 1789           84
## 5 1790           99
## 6 1791          126
## 7 1792          135
## 8 1793          141
## 9 1794          146
## 10 1795          136
## # ... with 39 more rows
```

(13) How many meetings were there in each conference in each year since 1802?

```
methodists %>%
  filter(year >= 1802) %>%
  group_by(conference, year) %>%
  summarize(total_meetings = n())
```

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

```
## # A tibble: 389 x 3
## # Groups:   conference [26]
##   conference year total_meetings
##   <chr>      <int>         <int>
## 1 ""        1805             1
## 2 "Alabama" 1833             22
## 3 "Alabama" 1834             28
## 4 "Baltimore" 1802             30
## 5 "Baltimore" 1803             30
## 6 "Baltimore" 1804             37
## 7 "Baltimore" 1805             39
## 8 "Baltimore" 1806             39
## 9 "Baltimore" 1807             46
## 10 "Baltimore" 1808             42
## # ... with 379 more rows
```

(14) What is the average number of white, black, Indian and total members for each year since 1786?

```
methodists %>%
  filter(year >= 1786) %>%
  group_by(year) %>%
  summarize(total_white = mean(members_white, na.rm = TRUE), total_black = mean(members_black, na.rm = TRUE),
            total_indian = mean(members_indian, na.rm = TRUE))
```

```
## # A tibble: 49 x 4
##   year total_white total_black total_indian
##   <int>      <dbl>      <dbl>      <dbl>
## 1 1786      359.       56.7         0
## 2 1787      399.       70.6         0
## 3 1788      402.      105.         0
## 4 1789      410.      105.         0
## 5 1790      464.      118          0
## 6 1791      401.      104.         0
```

```
## 7 1792      386.      103.      0
## 8 1793      365.      102.      0
## 9 1794      362.       95.2      0
## 10 1795      354.       89.5      0
## # ... with 39 more rows
```

Being able to create summaries like these is essential for visualizing the data.